Paper 046-2007

Turning the data around: PROC TRANSPOSE and alternative approaches

Erik W. Tilanus, independent consultant, Driebergen, the Netherlands

ABSTRACT

PROC TRANSPOSE can be used to "rotate" (transpose) SAS® data sets. I.e. the values of one given variable become variable names and variables in the input data set become observations in the output.
The answers to three simple questions suffice to define the specification statements of PROC TRANSPOSE.
This paper poses these questions and demonstrates how PROC TRANSPOSE can be handled using them.

INTRODUCTION

In reporting it is common to have a time dimension in the horizontal direction, like the figures you see in Table 1. For analysis with SAS it is often more appropriate to have these figures "rotated", i.e. a variable YEAR and variables for the various equipment types, like in Table 2.

That is exactly what PROC TRANSPOSE can do for you.

In this paper we will discuss the basics of PROC TRANSPOSE and demonstrate a number of special techniques with it. We also look at alternatives, using a DATA step.

Product group	2000	2001	2002	2003	2004	2005
CD players	23043	24543	22865	21982	20637	19494
MP3 players	8345	10468	12270	13846	15029	17404
Cassette players	12593	11470	10343	9317	8204	6971
DVD players	15390	17489	19153	20458	21725	23094
DVD recorders	3046	3382	3574	3760	4501	5036
VHS recorders	14590	13695	13583	11459	9851	7466

Table 1: Units sold figures of some electronics store

YEAR	CD players	MP3 players	Cassette players	DVD players	DVD recorders	VHS recorders
2000	23043	8345	12593	15390	3046	14590
2001	24543	10468	11470	17489	3382	13695
2002	22865	12270	10343	19153	3574	13583
2003	21982	13846	9317	20458	3760	11459
2004	20637	15029	8204	21725	4501	9851
2005	19494	17404	6971	23094	5036	7466

Table 2: The same data, but rotated, is often easier to analyse with SAS

PROC TRANSPOSE, THE BASICS

In the introduction we mentioned already that PROC TRANSPOSE "rotates" a SAS data set, which is to say that TRANSPOSE turns variables into observations and observations into variables. With TRANSPOSE you can create a certain level of independence between program and data. Now processing and presentation can be optimized separately from defining the data set for optimal efficiency.

SPECIFICATION IN THREE EASY QUESTIONS

PROC TRANSPOSE is controlled by three specification statements: the ID statement, the VAR statement and the BY statement. The variables to be declared in these statements can be determined with the help of a few simple questions:

1. *Which variable in the input data set contains (after formatting!) the variable names of the output data set?*

This variable is specified in the ID statement. From the question it is clear that this variable must have a unique value in each observation (or per BY group) after formatting, since this becomes the name of the transposed variable. If the contents of the variable does not comply with the rules for variable names, SAS will adapt it by, among other things, replacing unacceptable characters with underscores. If the ID variable is numeric it is necessary to attach a prefix to its value, to change for example the value 3 into SCORE3 or the value 12 into SCORE12. Such a prefix is declared in the PROC statement with the PREFIX= option.

2. *Which variable(s) in the input data set contains the values to be transposed?*

These variables are declared in the VAR statement. Now, in order to find out from which original variable in the input data set the value originates, SAS adds the variable _NAME_. Should another variable name be desired here, the name must be given in the option NAME= in the PROC statement.
If the VAR statement is left out, then all numeric variables which do not as yet have any other task (i.e. declared in an ID or BY statement) will be transposed.

3. *For which group of observations is the value of the ID variable unique (forms a 'block' to be transposed)?*

This group of observations is designated in the BY statement. The presence of the BY statement means that the data set will not be transposed as a whole, but transposing will take place per BY group.

FIRST EXAMPLE

Now let us apply this to the data in the introduction. (Table 1) Let us assume that the table is in SAS data set format, with the variables Product_Group and Y2000 – Y2005.

Question 1: The variable Product_Group contains the names for the new variables, so we enter:

```
ID Product_Group;
```

Question 2: The variables Y2000 – Y2005 contain the information to be transposed. Since these are all the numeric variables, you could leave out the VAR statement, but is a good habit to include it anyway:

```
VAR Y2000-Y2005;
```

Question 3: There is no BY group processing in this example. All values of the variable Product_Group are different. So we don't need a BY statement.

We add one more element: we replace the default _NAME_ variable by Year, using the NAME= option in the PROC statement. So the total program now looks like this:

```
LIBNAME Forum "E:\SASForum";
PROC TRANSPOSE DATA=Forum.Horizontal1 OUT=Forum.Vertical1 NAME=Year;
   ID Product_Group;
   VAR Y2000-Y2005;
RUN;
```

The result of running this code is a data set like in Table 3.

Obs	YEAR	CD_ players	MP3_ players	Cassette_ players	DVD_ players	DVD_ recorders	VHS_ recorders
1	Y2000	23043	8345	12593	15390	3046	14590
2	Y2001	24543	10468	11470	17489	3382	13695
3	Y2002	22865	12270	10343	19153	3574	13583
4	Y2003	21982	13846	9317	20458	3760	11459
5	Y2004	20637	15029	8204	21725	4501	9851
6	Y2005	19494	17404	6971	23094	5036	7466

Table 3: PROC PRINT output of data set Vertical1.

EXTENSION OF THE EXAMPLE

Table 4 contains an extension of Table 1 . Next to the variables in Table 1 it contains also the variable Outlet. Now we have the situation that the variable that contains the new variable names has a repetition of values. According to Question 3, the variable Outlet should be in the BY statement, since per outlet the values in Product_Group are unique. Note that not all product groups are present in all outlets. That is not a problem. It will simply lead to some missing values in the output.

Obs	Outlet	Product_Group	Y2000	Y2001	Y2002	Y2003	Y2004	Y2005
1	Electronics Market	CD players	18980	23967	17709	29189	19898	15886
2	Electronics Market	MP3 players	10843	13141	14294	19972	22290	17645
3	Electronics Market	DVD players	13938	16236	16370	22404	21302	20905
4	Electronics Market	DVD recorders	976	999	1171	1683	1211	3035
5	Electronics Market	VHS recorders	26527	29310	31817	20272	22458	11522
6	Media Center	CD players	20730	25086	20884	25192	21777	19933
7	Media Center	MP3 players	13034	16677	18018	21522	23254	25197
8	Media Center	Cassette players	15733	14618	14153	12402	8236	7781
9	Media Center	DVD players	14875	17290	17702	20379	23450	21668
10	Media Center	VHS recorders	15890	16240	18348	12067	10525	7652
11	Music Store	CD players	24563	26837	20464	29353	19852	19087
12	Music Store	MP3 players	9888	12785	12454	18457	20102	18328
13	Music Store	Cassette players	19690	15647	17392	16572	10864	10301
14	Music Store	DVD players	11880	13779	15845	16016	15518	19112
15	Music Store	DVD recorders	1097	1151	1537	1999	1565	3097
16	Music Store	VHS recorders	22934	23917	25693	18607	16236	10666
17	Video and more	CD players	5467	7364	5932	10399	5656	4491
18	Video and more	Cassette players	55303	34352	40553	33534	21303	24863
19	Video and more	DVD players	17026	18208	15774	27595	22277	23571
20	Video and more	DVD recorders	835	884	1050	1558	1089	2568
21	Video and more	VHS recorders	29794	37609	40375	22168	31374	16402

Table 4: Extension of table 1: inclusion of the outlet name

With the BY statement the program would look like this:

```
LIBNAME Forum "E:\SASForum";
PROC TRANSPOSE DATA=Forum.Horizontal4 OUT=Forum.Vertical4 NAME=Year;
   ID Product_Group;
   VAR Y2000-Y2005;
   BY Outlet;
RUN;
```

The result is presented in Table 5.

DOUBLE TRANSPOSITION

There are situations where the data cannot be transposed at once in the way we have seen so far. Let us assume that we would not only have the units sold in the table, but also the related turnover. Table 6 shows part of the new input table. We renamed the Y2000-Y2005 variables to Q2000-Q2005 (Q for quantity) and added the variables V2000-V2005 (V for Value, the turnover).

In our output we would like to see two variables related to each product category: the units sold (e.g. Q_CD_Players) and the turnover (e.g. (V_CD_Players). This cannot be achieved with a simple transposition.

There are two ways to approach this problem.

The first approach is to perform two standard transpositions, one for each series of variables to be transposed. Then the two resulting output data sets are merged (one to one) to achieve the desired result.

The second approach is a single transposition of all variables, followed by a data step where you merge the transposed data set with itself, filtering either the Q-related or V-related observations with a WHERE= clause.

Obs	outlet	YEAR	CD_ players	MP3_ players	DVD_ players	DVD_ recorders	VHS_ recorders	Cassette_ players
1	Electronics Market	Y2000	18980	10843	13938	976	26527	.
2	Electronics Market	Y2001	23967	13141	16236	999	29310	.
3	Electronics Market	Y2002	17709	14294	16370	1171	31817	.
4	Electronics Market	Y2003	29189	19972	22404	1683	20272	.
5	Electronics Market	Y2004	19898	22290	21302	1211	22458	.
6	Electronics Market	Y2005	15886	17645	20905	3035	11522	.
7	Media Center	Y2000	20730	13034	14875	.	15890	15733
8	Media Center	Y2001	25086	16677	17290	.	16240	14618
9	Media Center	Y2002	20884	18018	17702	.	18348	14153
10	Media Center	Y2003	25192	21522	20379	.	12067	12402
11	Media Center	Y2004	21777	23254	23450	.	10525	8236
12	Media Center	Y2005	19933	25197	21668	.	7652	7781
13	Music Store	Y2000	24563	9888	11880	1097	22934	19690
14	Music Store	Y2001	26837	12785	13779	1151	23917	15647
15	Music Store	Y2002	20464	12454	15845	1537	25693	17392
16	Music Store	Y2003	29353	18457	16016	1999	18607	16572
17	Music Store	Y2004	19852	20102	15518	1565	16236	10864
18	Music Store	Y2005	19087	18328	19112	3097	10666	10301
19	Video and more	Y2000	5467	.	17026	835	29794	55303
20	Video and more	Y2001	7364	.	18208	884	37609	34352
21	Video and more	Y2002	5932	.	15774	1050	40375	40553
22	Video and more	Y2003	10399	.	27595	1558	22168	33534
23	Video and more	Y2004	5656	.	22277	1089	31374	21303
24	Video and more	Y2005	4491	.	23571	2568	16402	24863

Table 5: The result of transposing the data in table Table 4.

DOUBLE TRANSPOSE: FIRST APPROACH, USING 2 TRANSPOSITIONS

The first approach uses two transpositions, one for the Q2000-Q2005 variables and one for the V2000-V2005 variables. To rename the output variables from e.g. CD_Players to Q_CD_Players, we use the PREFIX= option in the PROC statement. We merge the data sets together using a one to one merge. This is possible since the observations in the two data sets match perfectly.

Off course there can only be one variable Year in the output data set. According to the rules of the MERGE statement, this variable will contain the values from the last mentioned data set. But we don't need the prefix befor the year anymore, therefore we strip the prefix with the SUBSTR function.

This is the code for the first approach:

```
LIBNAME Forum "E:\SASForum";
PROC TRANSPOSE DATA=Forum.Horizontal6 OUT=Forum.Vertical6A NAME=Year PREFIX=Q_;
   ID Product_Group;
   VAR Q2000-Q2005;
   BY Outlet;
RUN;
PROC TRANSPOSE DATA=Forum.Horizontal6 OUT=Forum.Vertical6B NAME=Year PREFIX=V_;
   ID Product_Group;
   VAR V2000-V2005;
   BY Outlet;
RUN;
```

```
DATA Forum.Vertical6;
   MERGE Forum.Vertical6A
         Forum.Vertical6B;
   Year=SUBSTR(Year,2,4);
RUN;
```

Obs	Outlet	Product_Group	Q2000	Q2001	Q2002	Q2003
1	Electronics Market	CD players	18980	23967	17709	29189
2	Electronics Market	MP3 players	10843	13141	14294	19972
4	Electronics Market	DVD players	13938	16236	16370	22404
5	Electronics Market	DVD recorders	976	999	1171	1683
6	Electronics Market	VHS recorders	26527	29310	31817	20272
7	Media Center	CD players	20730	25086	20884	25192
8	Media Center	MP3 players	13034	16677	18018	21522
…	…	…	…	…	…	…

Obs	Q2004	Q2005	V2000	V2001	V2002	V2003	V2004	V2005
1	19898	15886	3302520	3834720	2603223	3940515	2487250	1826890
2	22290	17645	2016798	2076278	1915396	2276808	2162130	1464535
4	21302	20905	4947990	4724676	3912430	4391184	3429622	2759460
5	1211	3035	505568	455544	469571	594099	376621	831590
6	22458	11522	3289348	3487890	3627138	2229920	2358090	1163722
7	21777	19933	3607020	4013760	3069948	3400920	2722125	2292295
8	23254	25197	2424324	2634966	2414412	2453508	2255638	2091351
…	…	…	…	…	…	…	…	…

Table 6: Part of the table with units sold (Q2000-Q2005) and turn over (V2000-V2005)

DOUBLE TRANSPOSE: SECOND APPROACH, USING 1 TRANSPOSITION

In the second approach we use one transposition, transposing all variables at once. This will lead to an output data set with observations for each value of the Year variable: Q2000-Q2005 and V2000-V2005. Then we merge the data set to itself, filtering either the Q- or the V- values. At the same time we rename the transposed variables.

```
LIBNAME Forum "E:\SASForum";
PROC TRANSPOSE DATA=Forum.Horizontal6 OUT=Forum.Vertical6X NAME=Year;
   ID Product_Group;
   VAR Q2000-Q2005 V2000-V2005;
   BY Outlet;
RUN;
DATA Forum.Vertical6Y;
   MERGE Forum.Vertical6X(WHERE=(Year:"Q")
                          RENAME=(CD_Players=Q_CD_players
                                  MP3_Players=Q_MP3_Players
                                  Cassette_Players=Q_Cassette_Players
                                  DVD_Players=Q_DVD_Players
                                  DVD_Recorders=Q_DVD_Recorders
                                  VHS_Recorders=Q_VHS_Recorders))
         Forum.Vertical6X(WHERE=(Year:"V")
                          RENAME=(CD_Players=V_CD_players
                                  MP3_Players=V_MP3_Players
                                  Cassette_Players=V_Cassette_Players
                                  DVD_Players=V_DVD_Players
                                  DVD_Recorders=V_DVD_Recorders
                                  VHS_Recorders=V_VHS_Recorders));
   Year=SUBSTR(Year,2,4);
RUN;
```

A part of the resulting data set of both approaches is presented in Table 7.

Obs	outlet	Year	Q_CD_ players	Q_MP3_ players	Q_DVD_ players	Q_DVD_ recorders	Q_VHS_ recorders
1	Electronics Market	2000	18980	10843	13938	976	26527
2	Electronics Market	2001	23967	13141	16236	999	29310
3	Electronics Market	2002	17709	14294	16370	1171	31817
4	Electronics Market	2003	29189	19972	22404	1683	20272
5	Electronics Market	2004	19898	22290	21302	1211	22458
6	Electronics Market	2005	15886	17645	20905	3035	11522
7	Media Center	2000	20730	13034	14875	.	15890
8	Media Center	2001	25086	16677	17290	.	16240
…	…	…	…	…	…	…	…

Obs	Q_Cassette_ players	V_CD_ players	V_MP3_ players	V_DVD_ players	V_DVD_ recorders	V_VHS_ recorders	V_Cassette_ players
1	.	3302520	2016798	4947990	505568	3289348	.
2	.	3834720	2076278	4724676	455544	3487890	.
3	.	2603223	1915396	3912430	469571	3627138	.
4	.	3940515	2276808	4391184	594099	2229920	.
5	.	2487250	2162130	3429622	376621	2358090	.
6	.	1826890	1464535	2759460	831590	1163722	.
7	15733	3607020	2424324	5280625	.	1970360	2265552
8	14618	4013760	2634966	5031390	.	1932560	2031902
…	…	…	…	…	…	…	…

Table 7: The result of the double transposition

ANOTHER EXAMPLE: BUILDING A TWO-WAY DISTANCE TABLE

A distance table for a road map often contains city names both in the columns and the rows, so that a distance can always be determined in both directions. By means of PROC TRANSPOSE, this can be realized quite simply from a straight distance data set. Table 8 shows the input data set and Table 9 shows the desired 2-way distance table.

The first step is to generate additional observations in such way that there is a two-way pair of all cities.

```
DATA Expand(DROP=temp);
   SET Distance;
   * First write the original observation;
   OUTPUT;
   * Exchange To and From;
   temp = From;
   From = To;
   To = temp;
   OUTPUT;
RUN;
```

Next we use PROC TRANSPOSE to create the table. Let us follow the three questions again:

1. Which variable contains the names of the output variables?
The variable To. So To will be specified in the ID statement. The city name however may not meet the naming conventions for SAS variable names, e.g. New York. That is why we declare this variable also in an IDLABEL statement. This way the variable name may be adjusted, its original value is still in the label.

2. Which variable contains the values to be transposed?
The values to be transposed are the miles. So the variable Mile should be in the VAR statement.

3. Which variable marks the groups which will form the blocks to be transposed?
To is unique for each From value, but not across From values. Therefore we put From into the BY statement.

So this is the required PROC TRANSPOSE:

```
PROC TRANSPOSE OUT=Table(DROP=_NAME_);
  ID To;
  IDLABEL To;
  VAR Mile;
  BY From;
RUN;
```

Obs	From	To	Mile
1	New York	Toronto	495
2	New York	Boston	215
3	Boston	Toronto	549
4	Washington	New York	236
5	Washington	Boston	443
6	Washington	Toronto	554

Table 8: The input for the 2-way distance table.

From	Boston	New York	Toronto	Washington
Boston	.	215	549	443
New York	215	.	495	236
Toronto	549	495	.	554
Washington	443	236	554	.

Table 9: The desired 2-way distance table.

TRANSPOSING WITHOUT PROC TRANSPOSE

TRANSPOSING IN A DATA STEP

If you are a real DATA step die-hard, you may program a transposition within a DATA step, using arrays. The basic principle is not difficult: you fill a two-dimensional array row by row with the observations to transpose and then you build the output observations by reading the array column by column. But the pain is in the details. It might become reasonably complex coding. The code presented below will perform the same transposition as presented above, with Table 6 as input and Table 7 as output.

The program consists of two DATA steps. The first step is to determine a number of characteristics of the input data set and store them in macro variables. The reason to split it into two steps is simply to prevent hard coded values or variable names in the process, where they can be derived from the input data set.

In array NewVar we construct all the new variable names, like 'Q_CD_Players' adding any values that we encounter in the input. While reading the values of Product_Group, we also change them into valid SAS variable names. Once we have read the total data set, we generate a macro variable VarString that contains all the new variable names, a macrovariable Nobs with the number of obsservations and a macro variable NewVarCount with the total number of new variable names.

The second DATA step does the real work. It starts reading the input data set and stores the information into (temporary) array's, row by row. The dimension of the array is not critical, as long as they are big enough. That is why we use the variable count and observation count macro variables.

With the BY statement and the FIRST. and LAST. special variables we monitor the beginning and end of the observations of each outlet.

Once the last observation of an outlet is read we start constructing the output observation in the array outputvars. This time we read by column, storing the quantity and value information for each product group for one year in the output array. When we processed all rows (= product groups) for one year we write an observation to the output and continue with the next year.

```
LIBNAME Forum "E:\SASForum";
DATA _NULL_;
   RETAIN LastNewVar 0;
   LENGTH VarString $32000;
   SET forum.horizontal6 NOBS=nobs END=finish;
   Array NewVar {1000} $32 _TEMPORARY_;
   DO i=1 to LastNewVar;
      IF TRANSLATE(TRIM(Product_Group),'_',' ') = NewVar{i} THEN LEAVE;
   END;
   IF i = LastNewVar+1 THEN DO;
      NewVar{i} = TRANSLATE(TRIM(Product_Group),'_',' ');
      LastNewVar + 1;
   END;
   IF finish THEN DO;
      DO i=1 TO LastNewVar;
         VarString = CATX(' ',VarString,'Q_'||NewVar{i},'V_'||NewVar{i});
      END;
      CALL SYMPUT('NewVar',VarString);
      CALL SYMPUTX('NewVarCount',LastNewVar*2);
      CALL SYMPUTX('nobs',nobs);
   END;
RUN;
DATA Forum.TransposeDataStep(KEEP= Outlet Year &newvar);
   SET forum.horizontal6 END=finish;
   BY Outlet;
   ARRAY charvar {&newvarcount} $18 _TEMPORARY_;
   ARRAY numvar  {&nobs,100} _TEMPORARY_;
   ARRAY years {*} Q2000-Q2005 V2000-V2005;
   ARRAY outputvars {*} &newvar;
   IF FIRST.Outlet then obscount=0;
   obscount+1;
   charvar {obscount} = TRANSLATE(TRIM(product_group),'_',' ');
   DO i= 1 TO DIM(years);
      numvar{obscount,i} = years{i};
   END;
   IF LAST.Outlet THEN DO;
      DO col = 1 to DIM(Years)/2;
         DO row = 1 TO obscount;
            DO outpos = 1 TO &newvarcount;
               IF 'Q_'||charvar{row} = SCAN("&newvar",outpos,' ') THEN LEAVE;
            END;
            outputvars{outpos} = numvar{row,col};
            outputvars{outpos+1} = numvar{row,DIM(years)/2 + col};
         END;
         Year = 1999+col;
         OUTPUT;
         * Clear output array;
         DO n=1 to DIM(outputvars);
            outputvars{n} = .;
         END;
      END;
   END;
RUN;
```

PROC REPORT AND PROC TABULATE

If the only purpose of the transposition is to format the data in such way that it prints nicely, think again. You may not need to transpose the data at all. By specifying the proper rows and columns in PROC REPORT or PROC TABULATE, you may well be able to get all the reporting you need!.

CONCLUSION

Data analysis within SAS is almost always executed by variable across all observations. If your input data is not structured that way, you can use PROC TRANSPOSE to rotate the data an make it more appropriate for analysis. Using PROC TRANSPOSE is relatively simpel, if you keep the three questions in your mind that have been presented in this paper.

If the only reason that you want to transpose your data is the sequence in a report, you may not need to transpose at all, just use PROC REPORT or PROC TABULATE.

REFERENCES

More information about PROC TRANSPOSE can be found in:
SAS Publishing: Base SAS 9.1 Procedures Guide, Volumes 1-4, 2004
SAS Publishing: Base SAS 9.1.3 Procedures Guide, Second Edition, Volumes 1-4, 2006
More information about handling array's can be found in:
SAS Publishing: SAS 9.1.3 Language Reference: Concepts, Third Edition, Volumes 1 and 2, 2005
SAS Publishing: SAS 9.1.3 Language Reference: Dictionary, Fifth Edition, Volumes 1-4, 2006
All information can also be found in SAS OnlineDoc® 9.1.3

RECOMMENDED READING

PROC TRANSPOSE has been subject of several presentations at SUGI conferences. Following is a selection of these papers. A complet overview can be found on http://support.sas.com/events/sasglobalforum/previous/online.html.

An Alternative Method of Transposing Data Without the Transpose Procedure. Sunil K. Gupta, SUGI 22, paper 94, updated for SUGI 30, paper 039
The TRANSPOSE Procedure or How to Turn it Around. Janet Stuelpner, SUGI 31, paper 234
Changing the shape of your data: PROC TRANSPOSE vs. Arrays. Bob Vergile, SUGI 24, paper 60
Some uses (and handy Abuses) of Proc Transpose. Ralph W. Leighton, SUGI 27, paper 16, updated for SUGI 29, paper 267.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Erik W. Tilanus
Horstlaan 51
3971 LB Driebergen
the Netherlands
email: erik.tilanus@planet.nl
fax: +31 343 517007