PAPER PO05

PROC TRANSPOSE for Better Listings View

Jules van der Zalm, OCS Consulting, Rosmalen, the Netherlands

ABSTRACT

Listings are common within clinical trails, either for internal review or for reporting to the authorities; however it is not always that straightforward to generate a readable listing when the information does not fit properly on a page.

When we present, for example, a vital signs listing with the parameters across the top and the subjects row by row we may end up repeating the subject over more than one page, losing the ability to easily compare the vital sign parameters for a single subject. A preferred method would be to place the subjects across the top and to present the parameters on rows. This way the information for a single patient remains on the same page, improving the readability.

This paper will present a dynamic solution that will allow the programmer to easily create a listing that will fit on the page and will improve the readability.

INTRODUCTION

In clinical trials, all data should be processed into tables, figures and listings for review by the authorities. In particular the data listings can be pretty hard to display nicely on regular Letter or A4 paper sizes, especially when it comes to outputting large amounts of variables as often is the case in demographics, vital signs or laboratory output.

There are many workarounds for this problem, which all have their pros and cons. You can split the data up into multiple listings, so that each set of CRF items is presented in a separate listing. You can abbreviate repeating values and explain the abbreviations in a footnote, which will allow you to use smaller columns resulting in more columns on one page. You can present a grouped listing so the grouped information is only displayed once, and there are probably infinite other methods.
The major drawback of most of these methods is that they make it more difficult to look up data for a single subject, being it either because the data is harder to read due to the abbreviations and the large amount of information on the page, or on the other hand the data is spread over multiple pages so you will have to look up several different pages to get all the information of one subject.

This paper presents a method that has one big pro and only one minor con: it provides excellent readability of your output without concessions to the data, on the cost of just some extra paper.

THE METHOD

In regular listings all subjects are shown below each other, with their variables (the CRF items) shown in columns, like in the example on the following page. The example shows a regular listing of a demographic dataset that complies with CDISC standards. Because of the limited space, some columns like date/time columns are very compact making it hard to distinguish which data belongs to which column. Increasing the space between those columns might solve this problem but then the variables at the very right of the page would drop off. In fact, some variables have already been dropped because they wouldn't fit on the page.

The method that is presented in this paper introduces a completely different method: the variables, or CRF items, are shown below each other. Each subject has its own column, giving more room for long texts and providing flexibility as to how the data is presented. The output is easier on the eye and provides more detail as can be seen in the example on page 2.

REGULAR LISTING DISPLAY

Listing 1.2-A Demographics
All subjects by subject identifier.

Subject	Subj. id	Drug start date	Drug end date	Site	Birth date	Age	Age unit	Sex	Race	Ethnicity	Trt. arm code	Trt. arm	Coun-try	Collection date/time	Coll. day
OCS-001-0001	0001	2009-07-17 22:47:51	2009-08-15 03:27:08	001	1988-01-31 04:00:00	21	YEARS	M	WHITE	NOT HISPANIC OR LATINO	GRP1	GROUP 1	USA	2009-07-18 19:55:33	1
OCS-001-0002	0002	2009-07-29 09:03:33	2009-08-26 05:31:25	001	1990-01-05 09:30:00	19	YEARS	F	WHITE	NOT HISPANIC OR LATINO	GRP1	GROUP 1	USA	2009-07-29 15:26:58	1
OCS-004-0003	0003	2009-07-31 01:24:23	2009-08-28 01:34:08	004	1986-03-28 04:00:00	23	YEARS	M	WHITE	NOT HISPANIC OR LATINO	GRP1	GROUP 1	USA	2009-07-31 19:38:22	1
OCS-002-0004	0004	2009-07-15 04:25:12	2009-08-12 05:36:39	002	1988-06-13 04:45:00	21	YEARS	F	WHITE	NOT HISPANIC OR LATINO	GRP1	GROUP 1	USA	2009-07-15 13:32:19	1
OCS-001-0005	0005	2009-07-18 09:55:43	2009-08-15 13:35:02	001	1989-07-11 04:45:00	20	YEARS	F	WHITE	NOT HISPANIC OR LATINO	BLND	BLINDED	USA	2009-07-19 03:07:26	1
OCS-003-0006	0006	2009-07-19 19:41:53	2009-08-16 16:52:44	003	1990-07-04 05:30:00	19	YEARS	M	WHITE	NOT HISPANIC OR LATINO	GRP2	GROUP 2	USA	2009-07-20 07:19:26	1
OCS-003-0007	0007	2009-07-31 23:44:11	2009-08-29 02:27:41	003	1987-06-14 22:30:00	22	YEARS	F	WHITE	NOT HISPANIC OR LATINO	BLND	BLINDED	USA	2009-08-01 17:19:26	1
OCS-001-0008	0008	2009-07-26 16:28:24	2009-08-24 13:13:05	001	1987-02-14 21:00:00	22	YEARS	M	WHITE	NOT HISPANIC OR LATINO	BLND	BLINDED	USA	2009-07-27 15:22:00	1
OCS-004-0009	0009	2009-07-17 21:04:29	2009-08-15 05:58:33	004	1986-12-14 11:00:00	22	YEARS	M	WHITE	NOT HISPANIC OR LATINO	GRP2	GROUP 2	USA	2009-07-18 15:53:24	1
OCS-004-0010	0010	2009-07-19 16:24:32	2009-08-16 05:04:12	004	1989-01-27 05:15:00	20	YEARS	M	WHITE	NOT HISPANIC OR LATINO	GRP1	GROUP 1	USA	2009-07-19 23:17:44	1

Listing 1.2-A Demographics
All subjects by subject identifier.

Unique Subject No.	OCS-001-0001	OCS-001-0002	OCS-004-0003	OCS-002-0004
Study Identifier	OCS-5241	OCS-5241	OCS-5241	OCS-5241
Domain Abbreviation	DM	DM	DM	DM
Subject ID	0001	0002	0003	0004
Site Number	001	001	004	002
Planned Arm Code	GRP1	GRP1	GRP1	GRP1
Planned Arm Desc.	GROUP 1	GROUP 1	GROUP 1	GROUP 1
Country	USA	USA	USA	USA
Start Date	2009-07-17T23:04:42	2009-07-29T09:20:24	2009-07-31T01:41:14	2009-07-15T04:42:03
End Date	2009-08-15T03:43:59	2009-08-26T05:48:16	2009-08-28T01:50:59	2009-08-12T05:53:30
Date/Time of Birth	1988-01-31T04:15:00	1990-01-05T09:45:00	1986-03-28T04:15:00	1988-06-13T05:00:00
Age At Entry	21	19	23	21
Age Units	YEARS	YEARS	YEARS	YEARS
Sex	M	F	M	F
Race	WHITE	WHITE	WHITE	WHITE
Ethnicity	NOT HISPANIC OR LATINO	NOT HISPANIC OR LATINO	NOT HISPANIC OR LATINO	NOT HISPANIC OR LATINO
Collection Date/Time	2009-07-18T20:12:24	2009-07-29T15:43:49	2009-07-31T19:55:13	2009-07-15T13:49:10
Collection Day	1	1	1	1
Investigator	GWJ	ASD	REH	REH
Investigator Name	DR. JENKINS	DRS. DAVIS	DR. R. HUNT	DR. R. HUNT

TERMS DEFINITIONS

In order to improve the readability of this paper the most important terms that are used throughout this document are described here.

- CRF item: a question or item on the CRF, usually saved as a variable in a (SAS®) dataset
- Variable: column in a dataset
- Record: row in a dataset
- Reporting item: that what unique identifies the record (i.e. a subject or a visit)

STARTING POINT

The starting point of this method is the dataset for which the listing has to be created. The dataset should have only one record per subject or for example one record per visit, depending on the type of data. If you are displaying, for example, laboratory data which has a record per laboratory measurement per visit, then you should first transpose the data so that there is only one record per laboratory visit and the measurements are turned into variables. The techniques that are used to prepare such a dataset are considered as basic knowledge and are not explained in this paper. Once the starting point dataset has been created and sorted in the required order – probably by subject or by subject and visit – it is best to get rid of all variables that should not be presented in the listing.

The next steps will explain how to define the number of reporting items (being it subjects or visits) shown on a page, how to define the order of the CRF items on the output and how to setup PROC REPORT for the actual output. The examples are based on an example program that is referenced at the end of this document. This program uses a demographics dataset which complies with the CDISC standards. Because this demographics dataset is large with many columns, the examples on the paper are based on the SASHELP.CLASS dataset.

STEP 1: ADD PAGE NUMBERING

The first and most important step is the addition of page numbering to the records in the source dataset. By doing this the number of reporting items per page is defined. The page number is a number starting from 1 on the first page, incrementing by 1 on each following page. Each page has a certain number of records on it; the example on page 3 has got four. In the source dataset, all records that are displayed on the same page have the same page number, so in the dataset the page number increments after each four records.

This means that, when assigning page numbers to records, you directly define on which page the record will be shown. That is why it is so important that the dataset is sorted correctly. The code used to generate the page number may look like the following.

```
data work.class;
  retain pageno 0;
  set sashelp.class;

  if (mod(_n_, 4) = 1) then pageno + 1;

run;
```

In the example that is shown above every block of four records will have the same page number, meaning that there will be four reporting items on a page. Simply change the 4 that is in the `mod()`-function into a different number and the number of reporting items on a page will be increased or decreased. If you have many long fields that you don't want to have wrapped to the next line you may want to decrease the number of reporting items per page. You can increase the number of reporting items if there is much space between one item and the next. By increasing the number of items you will decrease the total number of pages, and vice versa. The number of reporting items per page is limited by the total width required per item and the available space on the physical page.

STEP 2: TRANSPOSE THE DATA

The next step is to transpose the data. When transposing it, all variables in the dataset are turned into records, and all the records (subjects, or subject visits) are turned into variables. Because the PROC TRANSPOSE is given a BY PAGENO parameter, the CRF items are repeated for each page. The result is a dataset that contains one record per page, per CRF item.

The code to perform this transpose may look like this:

```
proc transpose data=work.class
               out =work.class2;
  by pageno;
  var _all_;
run;
```

And if this is the dataset that is transposed by the PROC TRANSPOSE …

PAGENO	NAME	AGE	SEX
1	Alfred	14	M
1	Alice	13	F
2	Barbara	13	F

…. then this is what the result looks like:

PAGENO	_NAME_	COL1	COL2
1	Name	Alfred	Alice
1	Age	14	13
1	Sex	M	F
2	Name	Barbara	
2	Age	13	
2	Sex	F	

This is where the effect of this method becomes visible. The form of the dataset is now defined; the remaining steps will consist of fine-tuning and outputting.

STEP 3: DEFINING VARIABLE ORDER

By default, the order of the CRF items (that are now records) is the same as the order of the variables in the original dataset. This order may be as you require, but it is most likely that it is not, so to improve readability the order must be changed. We will also include white lines to separate related blocks of data.

The order of the output dataset will be defined in a dataset which is called the order definition dataset. This dataset contains the names of all variables in the source dataset. Each of these variable names will be assigned a sequence number. Additionally, variables can be given *options* or a new label. Giving *options* means that the variable is given a special code that, for example, implicates an empty line should follow the variable in the output. Yet this requires some custom coding, which is explained in Step 4.

Subsequently the sequence numbers in this order definition dataset can be merged onto the data, and the data can be sorted by this sequence number.

For better readability it is important that, instead of the variable names, labels that describe the actual variable are displayed in the output. The labels of the CRF items that are displayed in the output are those that were assigned to the variables in the source dataset. If there are many different lengths or these labels are incorrect, you can assign new labels in the order definition dataset. If there is a label assigned in the order definition dataset then this label will be used in the output. If there is none defined then the source variable label is displayed.

With respect to the example above, the code that is written to create the order definition dataset could look like this.

```
data work.orddef;
  infile datalines;
  length sequence 8 varname $8 options $4 label $30;

  input sequence 1-6
        varname  7-14
        options  16-19
        label    21-50;
datalines;
1   NAME     W
2   AGE
3   SEX          Gender
;
run;
```

When using this example data step in the output NAME will be shown on top, followed by AGE and SEX. The variable labels are taken from the variables in the source dataset except for variable SEX, for which the label is changed to 'Gender'.

STEP 4: CUSTOM CODE

After the order of the variables has been defined there is the possibility to apply some extra modifications to the data before outputting it. Modifications can be for example the addition of extra titles in between the variable labels, the insertion of printer codes for bold or italic output or applying indentation to the labels. This part requires custom programming that will not be described in this paper.

For numerical variables it may be necessary to remove leading spaces added when transposing the dataset in Step 2. Removing these leading spaces is also done in the custom code.

The custom code in the example program uses the *options* that are defined in the order definition dataset to insert white lines between two variables:

```
data work.class;
  set work.class;

  array cols col:;

  /* Strip leading and trailing spaces. */
  do i = 1 to dim(cols);
    cols[i] = strip(cols[i]);
  end;

  /* Insert white line if specified. */
  output;
  if index(options, 'W') > 0 then do;
    call missing(of col:, label);
    output;
  end;
run;
```

STEP 5: OUTPUTTING

By performing the steps above, the source dataset has been transformed into a dataset that is ready for output using PROC REPORT. Since there are only a few variables in the dataset, the PROC REPORT step is very short and generic.

There are only a few relevant variables in the dataset, which are:

- PAGENO defining the order of the subjects.
- SEQUENCE defining the order of the variables on each page.
- LABEL containing the original label of the source dataset's variable, or the label as defined in the order definition dataset if specified.
- COL1 - COLn variables as created by PROC TRANSPOSE. These contain the actual subject data.

The PROC REPORT step then may look like the following.

```
proc report data=work.class headline headskip nowindows noheader;
  column pageno sequence label col:;

  define pageno   / order noprint;
  define sequence / order noprint;
  define label    / display width=20;
  define col:     / display width=25 flow;

  break after pageno / page;
run;
```

The variables PAGENO and SEQUENCE are defined to determine the exact order of the data. They are not printed on the output. Although the data is probably already sorted as required it is good to have PROC REPORT define the order once again, just in case.

Then the variable LABEL is defined, with a width that equals the longest variable label. Finally the COL variables are defined using COL:. The semi-colon ensures that all available COL-variables are presented, so if you add more columns (in Step 1) then these are automatically processed by the PROC REPORT step.

Note that the option NOHEADER is specified. As the variable labels are presented below each other there is no need for column headers above each column.

The final output now looks like this.

```
Name    Alfred   Alice

Age     14       13
Gender  M        F
------ page break ------
Name    Barbara

Age     13
Gender  F
```

CONCLUSION

Although it requires some additional programming – that I'm sure you'll get used to – the method presented on this paper provides a solid method for presenting large amounts of data in a way that keeps the information perfectly accessible. Opposed to other methods data is not split up in separate sections and are not abbreviated to save space on the page, which are all methods that have a big impact on the readability of the listing.

REFERENCES

Navigate to the following location for the example program: http://www.ocs-consulting.com/phuse/papers/transpose, or send an e-mail to the author of this paper.

ACKNOWLEDGEMENTS

Thanks to my colleagues Bas van Bakel and Raymond Ebben for creative input and reviewing this paper, and to the PhUSE organisation for giving me the opportunity to present my paper at PhUSE 2009 in Basel.

CONTACT INFORMATION

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

Author name	:	Jules van der Zalm
Company	:	OCS Consulting
Address	:	P.O. Box 490 5240 AL Rosmalen The Netherlands
Work phone	:	+31 (0)73 523 6000
Facsimile	:	+31 (0)73 523 6600
E-mail	:	sasquestions@ocs-consulting.com
Website	:	www.ocs-consulting.nl