PhUSE 2016

Paper PP02

Efficiently Converting Datasets to Vertical Structure using Metadata

Thevaki Mahadevan, Syne Qua Non, Norfolk, UK

ABSTRACT

Data collected from a single CRF page, at an analysis time point, holds data for many analysis parameters per subject within an observation, which means raw data comes in as horizontal data structure. This often then gets converted to vertical structure by programmers for analysis purposes. This paper looks at how we can utilize the common variable prefixes and metadata details obtained via the proc contents procedure to automate converting horizontal datasets into vertical structure. Generally proc transpose or arrays are used for this purpose but the advantages of the proc contents approach is that it avoids the need for manually entering variable names and their labels, which certainly save time and improve quality in particular when there are many parameters to transpose. Once the unique parameters needing to be transposed are identified, they can be macro parametrized and looped through, in a data step, to convert them to vertical structure.

INTRODUCTION

Regulatory authorities ask for better data transparency and propose that data sets should be identified in such a way that they can be linked to protocols, publications, trial registries and patients. Data collected for each study will vary depending on the nature of the study and its objectives. The way to process that data further will also vary depending on its Metadata. Therefore a closer look at some of the approaches to accessing that Metadata adds significance when choosing the best.

TRANSPOSING HORIZONTAL RAW DATASET TO VERTICAL ANALYSIS DATASET

TRENDS IN aCRF MAPPING

Variable names within a domain tend to have the same prefix letters. E.g. AIM domain shown below in the aCRF page has the same prefix letters, AIMxxx, for all the domain specific variables.

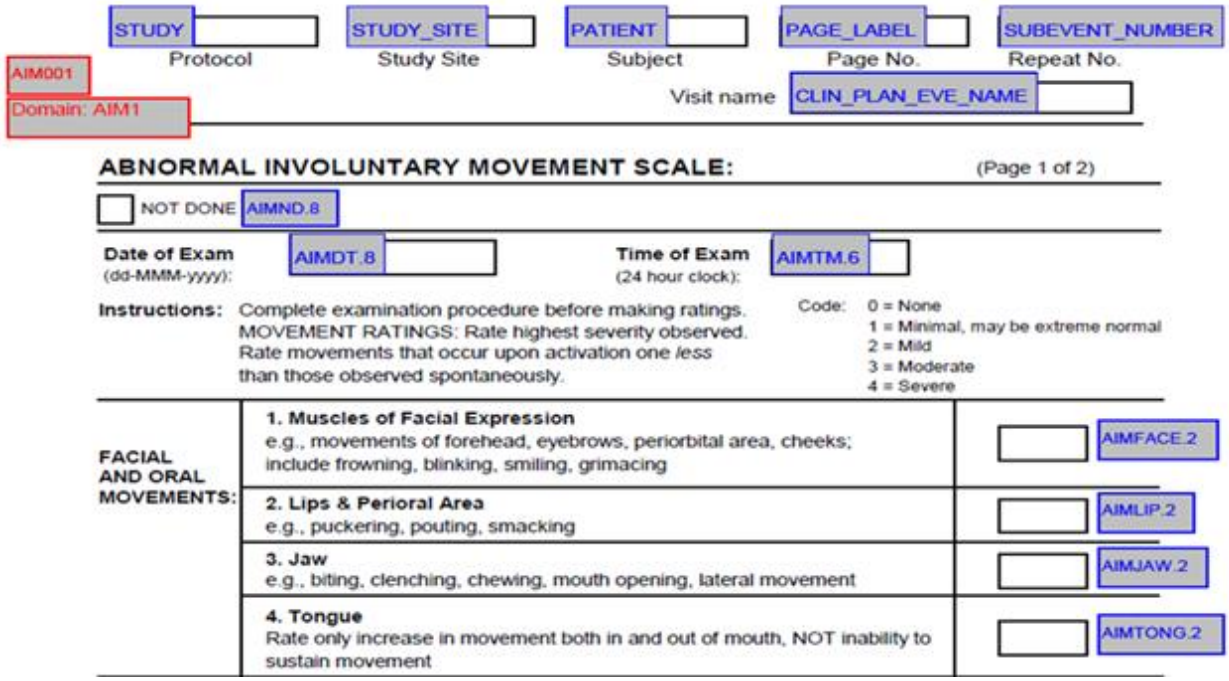


Figure 1: Example of an aCRF domain, showing the trends in variable names when responding to each questions

HORIZONTAL RAW DATASET

Same variable names, as mapped in the aCRF, are kept in the horizontally structured raw dataset and support an understanding of the relationship of aCRF to raw dataset, aiding traceability. Generally as part of the mapping process, each variable collected is formatted to give two/three forms; a numeric, character and a long/full (decoded/formatted) version.

SUBJID	CPEVENT	AIMND	AIMFACE	AIMFACEF	AIMLIP	AIMLIPF	AIMJAW	AIMJAWF	AIMTONG	AIMTONGF	AIMUPPR	AIMUPPRF	AIMLOW	AIMLOWF
10011001	DAY 1		0	0	0	0	0	0	0	0	0	0	0	0
10011001	DAY 8		0	0	0	0	0	0	0	0	0	0	0	0
10011001	DAY15		0	0	0	0	0	0	0	0	0	0	0	0
10011001	DAY22		0	0	0	0	0	0	0	0	0	0	0	0
10011001	END_OF_TREATMENT		0	0	0	0	0	0	0	0	0	0	0	0
10011001	FOLLOW_UP	NOT DONE	.		.		.		.		.		.	
10011002	DAY 1		0	0	0	0	0	0	0	0	0	0	0	0
10011002	DAY 8		0	0	0	0	0	0	0	0	0	0	0	0
10011002	DAY15		0	0	0	0	0	0	0	0	0	0	0	0
10011002	DAY22		0	0	0	0	0	0	0	0	0	0	0	0
10011002	END_OF_TREATMENT		0	0	0	0	0	0	0	0	0	0	0	0
10011002	FOLLOW_UP	NOT DONE	.		.		.		.		.		.	
10011004	DAY 1		0	0	0	0	0	0	0	0	0	0	0	0
10011004	DAY 8		0	0	0	0	0	0	0	0	0	0	0	0
10011004	DAY15		0	0	0	0	0	0	0	0	0	0	0	0

Figure 2: Example of a horizontally structured raw dataset

VERTICAL ANALYSIS DATASET

The vertical structure lends itself to flexibility:

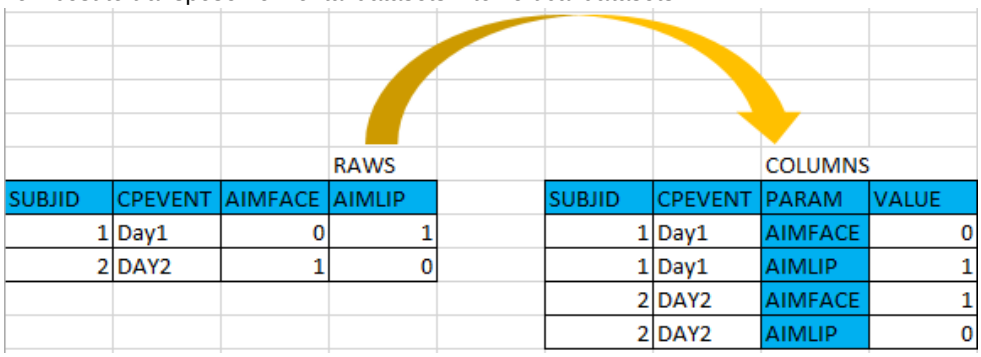
- Each row holds one question and its response.
- Easy to add or remove questions, even if over time as new questions are added to CRFs, or to derive different endpoints.
- Easy to apply an algorithm across all the questions, and across questions from different CRFs.
- Different data standards will have the same structure, so this simplifies the learning curve for additional data standards.

CPEVENT	EFDT	EFND	EFQUES	EFQUESN	EFRAW	EFRSLT	SUBJID	EFTM
DAY 1	21AUG2014		AIMFACE	1	0	0	10011001	17:01:00
DAY 1	21AUG2014		AIMLIP	2	0	0	10011001	17:01:00
DAY 1	21AUG2014		AIMJAW	3	0	0	10011001	17:01:00
DAY 1	21AUG2014		AIMTONG	4	0	0	10011001	17:01:00
DAY 1	21AUG2014		AIMUPPR	5	0	0	10011001	17:01:00
DAY 8	28AUG2014		AIMFACE	1	0	0	10011001	14:10:00
DAY 8	28AUG2014		AIMLIP	2	0	0	10011001	14:10:00
DAY 8	28AUG2014		AIMJAW	3	0	0	10011001	14:10:00
DAY 8	28AUG2014		AIMTONG	4	0	0	10011001	14:10:00
DAY 8	28AUG2014		AIMUPPR	5	0	0	10011001	14:10:00
DAY15	04SEP2014		AIMFACE	1	0	0	10011001	15:02:00
DAY15	04SEP2014		AIMLIP	2	0	0	10011001	15:02:00
DAY15	04SEP2014		AIMJAW	3	0	0	10011001	15:02:00
DAY15	04SEP2014		AIMTONG	4	0	0	10011001	15:02:00
DAY15	04SEP2014		AIMUPPR	5	0	0	10011001	15:02:00
DAY22	10SEP2014		AIMFACE	1	0	0	10011001	16:20:00
DAY22	10SEP2014		AIMLIP	2	0	0	10011001	16:20:00
DAY22	10SEP2014		AIMJAW	3	0	0	10011001	16:20:00
DAY22	10SEP2014		AIMTONG	4	0	0	10011001	16:20:00
DAY22	10SEP2014		AIMUPPR	5	0	0	10011001	16:20:00
END_OF_TREATMENT	19SEP2014		AIMFACE	1	0	0	10011001	8:02:00
END_OF_TREATMENT	19SEP2014		AIMLIP	2	0	0	10011001	8:02:00
END_OF_TREATMENT	19SEP2014		AIMJAW	3	0	0	10011001	8:02:00
END_OF_TREATMENT	19SEP2014		AIMTONG	4	0	0	10011001	8:02:00
END_OF_TREATMENT	19SEP2014		AIMUPPR	5	0	0	10011001	8:02:00
FOLLOW_UP	26SEP2014	NOT DONE	AIMFACE	1	.	.	10011001	.
FOLLOW_UP	26SEP2014	NOT DONE	AIMLIP	2	.	.	10011001	.
FOLLOW_UP	26SEP2014	NOT DONE	AIMJAW	3	.	.	10011001	.
FOLLOW_UP	26SEP2014	NOT DONE	AIMTONG	4	.	.	10011001	.
FOLLOW_UP	26SEP2014	NOT DONE	AIMUPPR	5	.	.	10011001	.

Figure 3: Example of a vertically structured analysis dataset

TYPICAL TRANSPOSING APPROACHES

How best to transpose horizontal datasets into vertical datasets?



RAWS

SUBJID	CPEVENT	AIMFACE	AIMLIP
1	Day1	0	1
2	DAY2	1	0

COLUMNS

SUBJID	CPEVENT	PARAM	VALUE
1	Day1	AIMFACE	0
1	Day1	AIMLIP	1
2	DAY2	AIMFACE	1
2	DAY2	AIMLIP	0

Figure 4: Sample transposing of horizontally structured raw dataset to vertically structured dataset

Typical approaches to transposing datasets are by;

1. proc transpose
2. Arrays

Some of the disadvantages of these approaches include;
1) Requiring manual entry of the variables and their attributes
2) Will consume more time
3) Has more chances for human error
4) Transparency of the data is manually influenced

PROC TRANSPOSE APPROACH

An example code of a proc transpose approach is shown below. Here, each transposing variable is first grouped as a numeric, character or short name variable type first. Then each individual group variables are passed through proc transpose, one at a time, and then the transposed datasets within each groups are set together, before merging each group data altogether.

```
proc transpose data=hv_data out=hvn&_i.&var;
          by &_bygroup;
          var  &&g_&var._sv&_i;
run;

/* When &&g_&var._sv&_i resolves to AIMFACE*/
proc transpose data=hv_data out=hvn1numvar;
by INV INVSITE PID PROJCODE PROTNO PT QUALIFYV REPEATSN SID STUDY SUBEVE TRIALNO VISIT
SUBJID COLLDATE CPEVENT EFDT EFDTF EFTM EFTMF EFND EFNDN EFDTP;
var AIMFACE;
run;

Data numvar;
set hvn1numvar hvn2numvar….;
Run;

Data final;
merge numvar charvar shortvar;
by INV INVSITE PID PROJCODE PROTNO PT QUALIFYV REPEATSN SID STUDY SUBEVE TRIALNO VISIT
SUBJID COLLDATE CPEVENT EFDT EFDTF EFTM EFTMF EFND EFNDN EFDTP _flag ;
rename short_n=efques short_lab=efquesf;
Run;

 /* Numeric Variables List */
_numvar =
%str(AIMFACE/AIMLIP/AIMJAW/AIMTONG/AIMUPPR/AIMLOW/AIMNECK/AIMSEV/AIMINC/AIMAWAR/AIMTEE
N/AIMDENTN),

 /* Character Variables List */
_charvar =
%str(AIMFACEF/AIMLIPF/AIMJAWF/AIMTONGF/AIMUPPRF/AIMLOWF/AIMNECKF/AIMSEVF/AIMINCF/AIMAW
```

```
ARF/AIMTEEF/),

/* Shortname Variables List */
_shortvar =
%str(AIMFACE/AIMLIP/AIMJAW/AIMTONG/AIMUPPR/AIMLOW/AIMNECK/AIMSEV/AIMINC/AIMAWAR/AIMTEE
N/AIMDENTN),
```

AN ARRAY APPROACH

An example code of an array transposing approach is shown below. Here, each short name variables, their labels and the results holding variables are identified first. Then their values or variables representing their values, in the case with results, are manually entered in an array statement before transposing them in a do loop.

```
data e5d_quesn;
   attrib  EFQUES    length=$50  label="Efficacy parameter short name"
           EFQUESF   length=$200 label="Efficacy parameter long name"
           EFQUESN   length=8    label="Efficacy parameter number"
           EFRSLT    length=8    label="Efficacy result for analysis - num"
           EFRSLTF   length=$100 label="Efficacy result for analysis - char"
           EFRAW     label="Efficacy raw result - num" length=8
           EFRAWF    label="Efficacy raw result - char" length = $80;

   set e5d_2;
   array ques{6} $ 8.    ('E5D_UTIL', 'E5D5MOB', 'E5D5SEL', 'E5D5ACT', 'E5D5DIS',
'E5D5ANX');
   array ques_1{6}       (1 2 3 4 5 6);
   array quesf{6} $ 200. ('EQ-5D Utility Index Score', 'Mobility', 'Self-Care',
'Usual Activities', 'Pain/Discomfort','Anxiety/Depression');
   array quesn_1{6}       utscore E5D5MOBN E5D5SELN E5D5ACTN E5D5DISN E5D5ANXN;
   array rawf_1{6} $     utf E5D5MOB E5D5SEL E5D5ACT E5D5DIS E5D5ANX;

    do i = 1 to dim(ques);
      efques  = ques(i);
      efquesn = ques_1(i);
      efquesf = quesf(i);
      EFRSLT  = quesn_1(i);
      EFRSLTF = rawf_1(i);
      efraw   = quesn_1(i);
      efrawf  = rawf_1(i);
       output;
   end;
run;
```

METADATA TRANSPOSING APPROACH

RAW DATASET METADATA

First the content is output into a dataset, to be able to access variable names and their labels, and to analyse the metadata closer. Only one unique variable name per question, its label and type are macro parametrised. Then the correlation among its duplicate associated variables are identified. Finally a do loop is programmed to process each unique set of variables using the macro parameters and their associated duplicate variable's correlations.

```
proc contents data=aim out=acont noprint;  /** Understand the RAW DATASET contents **/
run;
```

MEMNAME	NAME	TYPE	LENGTH	VARNUM	LABEL	FORMAT	FORMATL	INFORMAT	NPOS	NOB
AIM	AIMAWAR	1	8	52	Awareness of Abnormal Movements		2		168	386
AIM	AIMAWARF	2	200	53	Awareness of Abnormal Movements-FUL	$	200	$	3162	386
AIM	AIMDENT	2	3	59	Wears Dentures	$	3	$	3765	386
AIM	AIMDENTF	2	200	61	Wears Dentures-FUL	$	200	$	3968	386
AIM	AIMDENTL	2	200	60	Wears Dentures-DLV	$	200	$	3768	386
AIM	AIMDENTN	1	8	58	Wears Dentures-DVN		10		184	386
AIM	AIMDT	1	8	2	Actual Date of Visit/Collection/Exam	DATE	9		8	386
AIM	AIMDTF	2	200	33	Actual Date of Visit/Collection/Exam-FUL	$	200	$	1162	386
AIM	AIMFACE	1	8	34	Muscles of Face		2		96	386
AIM	AIMFACEF	2	200	35	Muscles of Face-FUL	$	200	$	1362	386
AIM	AIMINC	1	8	50	Incapacity Due Abnormal Movements		2		160	386

Figure 5: Example of a proc content output dataset

```
/****** Identify unique variables needed as macro parameters ******************/
proc sort data=acont out=aconts;
by name type;
run;

data pacont;
set aconts;
by name type;
 /**Remove variables not needed for transposing, e.g. date/time/not done **/
where name ? 'AIM' and name not in
('AIMND','AIMNDN','AIMNDF','AIMTM','AIMTMF','AIMDT','AIMDTF',
'AIMTEEL', 'AIMDENTL');
 /****** Keep only unique variables and  remove duplicate vars ****/
if STRIP(lag(name))||'F' eq  name then delete;
/******Handle original character separately   ******************/
if STRIP(lag(name))||'N' eq  name then delete;
run;

   data _null_;
    set pacont end=last;
    if last then  call symput('tot',_n_);
    call symput('NAME'||left(_N_), trim(left(name)));
    call symput('NAML'||left(_N_), trim(left(LABEL)));
    call symput('TYPE'||left(_N_), trim(left(TYPE)));
   run;
 %put &tot &NAME1 &NAML1 &TYPE1;

/******Convert to vertical structure ******************/
%macro vertical;
data AIMq2 (DROP=AIM:);
set AIMq1;
 length efquesn 8 efques $8 efquesf $80 efraw 8 efrawf $80 efrslt 8 efrsltf $80;
  label EFQUESN = "Efficacy parameter number"
        EFQUES  = "Efficacy parameter short name"
        EFQUESF = "Efficacy parameter long name"
        EFRAW   = "Efficacy raw result - num"
        EFRAWF  = "Efficacy raw  result - char"
        EFRSLT  = "Efficacy result for analysis - num"
        EFRSLTF = "Efficacy result for analyses - char"   ;
%do i=1 %to &tot;
EFQUESN = &i;                          /* parameter number */
EFQUES  = "&&name&i";           /* parameter short name"  */
%if &&type&i eq 2 %then %do; EFRAW   = &&name&i..N; %end;      /* raw result - num */
%else %if &&type&i eq 1 %then %do; EFRAW   = &&name&i; %end;   /* raw result - num */
EFRAWF  = &&name&i..F;          /* raw result - char */
EFQUESF = "&&naml&i";           /*parameter long name */
EFRSLT = efraw; EFRSLTF = efrawf;
output;
%end;

run;
%mend;
%vertical;
```

CONCLUSION

Several ways to transpose a horizontally structured dataset to vertical, but they all require a good understanding of the dataset metadata. A data dependent automated approach demonstrated in this paper has many advantages over other manually inputted counterpart approaches, particularly when there are many variables to transpose.

CONTACT INFORMATION

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

Thevaki Mahadevan
Syne Qua Non
Gostling House
Diss Business Park
Diss
Norfolk
IP22 4GT
UK
Email:Thevaki.Mahadevan@synequanon.com
Web: http://www.synequanon.com