Creating Graphical Patient Profiles Using SAS
William Garner, Gilead Sciences, Inc., Foster City, CA

ABSTRACT

Patient profiles bring together various sources of related data for a subject participating in a clinical trial. They are especially useful when the data is dynamic, e.g. additional tests are done based on the finding of a radiological scan.

This paper describes the development of such a profile for a clinical trial as well as general macros made which utilizes general inputs from a standardized dataset structure. The profile is broken down into a series of 3 objectives, each with their own numbered steps.

INTRODUCTION

A graphical patient profile can display the relationship between various data sources in a way that is easy for the human eye to process. In this paper, we present a graphical profile using SAS 9.2 Statistical Graphics (SG) Procedures. The profile is generated through the use of two macros and a standard dataset structure similar to the CDISC ADaM Basic Data Structure (BDS). We begin with a discussion of the study that motivated the development of these profiles. It should be noted that the profiles presented here were used in conjunction with supporting text-based listings, as the listings allow for additional information to be provided that cannot easily fit on a single graph.

MOTIVATION

The motivation for these profiles stems from a clinical trial investigating invasive fungal infections (IFI). The European Organization for Research and Treatment of Cancer/Invasive Fungal Infections Cooperative Group and the National Institute of Allergy and Infectious Diseases Mycoses Study Group (EORTC/MSG) has published specific criteria for assessing the presence of an IFI. The assessment of an IFI is made by an external efficacy committee.

Previously, text-based listings for all tests performed were sent to the committee for review. This resulted in dozens of pages of output per subject and required the reviewer to flip through multiple outputs to assess relationships between variables. This process could be streamlined with a single graphical timeline that showed key variables used in the diagnosis.

After numerous consultations with the study clinician, the profile in Figure 1 (below) was generated and sent to the members of the committee for their review. There are five main points of interest in the profile.

1. Demographic information appears across the top, including the reason the committee is reviewing this subject
2. Each variable appears as a unique row with specified labels
3. A legend appears on the bottom to describe the abbreviations and colors used
4. Study days appear across the bottom while calendar dates appear across the top
5. This cluster shows key data related to the assessment of IFI

It is also worth noting the presence of the yellow strips labeled “No Data to Report” on rows 20 and 22. This indicates that this subject did not have data from those tests. However, other subjects did have such data. The reason for this was to ensure that every subject had the same 22 rows. This facilitates quick comparisons across subjects.

BEHIND THE SCENES

At its core, the profile requires four inputs which are listed below and demonstrated in Table 1 (below).

1. Subject-level demographic information
2. Labels to state what is being displayed
3. Information regarding how to display the data (and what to display)
4. Information regarding where to display the data (including which row and where along the timeline)
Figure 1. Patient Profile for Diagnosis of IFI According to EORTC Criteria

**Demographic Information**

- **Subject ID:** AAA
- **Site Number:** AAA
- **Age:** 68 yr
- **Sex:** M
- **Race:** White
- **Subject Died:** No
- **Stopped Study Drug & Started Antifungals:** Yes
- **RI End Date (Reason):** 31JAN2012 (Consolidation)
- **Achieved Complete RI:** Yes

**Review Reason:** Took antifungals, investigator checked IFI criteria; radiological scan done; positive beta-D-glucan test

**Study Day**

- **Drug Admin:**
  - Observed Admin
  - Antifungal To Admin
  - Antimicrobial Agent

- **Temperature:**
  - Normal

- **WBC:**
  - 1.84

- **Leukopenia:**
  - None

- **Radiology:**
  - Chest CT

- **Other:**
  - Normal

- **Lung Culture:**
  - Normal

- **Blood Culture:**
  - Normal

- **Urine Culture:**
  - Normal

- **Gastrointestinal Criteria:**
  - Normal

- **Mycological Criteria:**
  - Normal

- **App. GM (Serum):**
  - Positive

- **App. GM (BAL):**
  - Positive

- **Sputum:**
  - Positive

**Test Results**

- **Cultured:**
  - Positive

- **Beta-D-Glucan:**
  - Positive

- **Test Result:**
  - Positive

**Data Extraction Date:** 01MAR13:07:04:37
The initialization of the profile can be broken down into 3 steps:

1. Determine the study day for all dates
2. Determine the data extraction date (for the footer)
3. Determine the number of lines to print

CREATE THE GRAPHICAL PROFILE

There are three main steps generating the profile:

1. Initialize the profile (using data from all subjects)
2. Prepare an individual subject’s data
3. Display the subject’s data

INITIALIZE THE PROFILE

The initialization of the profile can be broken down into 7 steps:

1. Determine the data extraction date (for the footer)
2. Load the data and add abbreviations to the display types
3. Determine the number of lines to print
4. Create a blank dataset to fill in any holes for the subject if data is not present for a particular row
5. Use the blank dataset to assign formats to the y-values (the labels that correspond with the line numbers)
6. Define the colors and fonts to use in the profile
7. Determine the study day for all dates

Table 1. Example Input Dataset

<table>
<thead>
<tr>
<th>Subj</th>
<th>Age</th>
<th>Sex</th>
<th>Race</th>
<th>Trta</th>
<th>Fdosdt</th>
<th>Label</th>
<th>Line</th>
<th>Stdtn</th>
<th>Endtn</th>
<th>Value</th>
<th>Value2</th>
<th>Dtype</th>
<th>Lgrp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Drug1</td>
<td>1</td>
<td>08JAN12</td>
<td>08JAN12</td>
<td>Drug1</td>
<td>DOT</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Drug1</td>
<td>1</td>
<td>16JAN12</td>
<td>16JAN12</td>
<td>Drug1</td>
<td>DOT</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Drug1</td>
<td>1</td>
<td>28JAN12</td>
<td>28JAN12</td>
<td>Drug1</td>
<td>DOT</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Drug1</td>
<td>1</td>
<td>07FEB12</td>
<td>07FEB12</td>
<td>Drug1</td>
<td>DOT</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Drug1</td>
<td>1</td>
<td>13JAN12</td>
<td>02FEB12</td>
<td>Drug2</td>
<td>INTERVAL</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Fever</td>
<td>3</td>
<td>31JAN12</td>
<td>01FEB12</td>
<td><em>ABN</em></td>
<td>INTERVAL</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Fever</td>
<td>3</td>
<td>28JAN12</td>
<td>28JAN12</td>
<td><em>ABN</em></td>
<td>INTERVAL</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab1</td>
<td>5</td>
<td>08JAN12</td>
<td>08JAN12</td>
<td>11.1</td>
<td>AS_IS</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab1</td>
<td>5</td>
<td>18JAN12</td>
<td>18JAN12</td>
<td>10.8</td>
<td>AS_IS</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab1</td>
<td>5</td>
<td>28JAN12</td>
<td>28JAN12</td>
<td>4.2</td>
<td><em>ABN</em></td>
<td>AS_IS</td>
<td>1</td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab2</td>
<td>6</td>
<td>08JAN12</td>
<td>08JAN12</td>
<td>250</td>
<td>AS_IS</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab2</td>
<td>6</td>
<td>18JAN12</td>
<td>18JAN12</td>
<td>300</td>
<td>AS_IS</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab2</td>
<td>6</td>
<td>28JAN12</td>
<td>28JAN12</td>
<td>500</td>
<td>AS_IS</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab3</td>
<td>7</td>
<td>13JAN12</td>
<td>13JAN12</td>
<td><em>ABN</em></td>
<td>INTERVAL</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>Lab3</td>
<td>7</td>
<td>02FEB12</td>
<td>02FEB12</td>
<td>TRACE</td>
<td>INTERVAL</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>AE1</td>
<td>8</td>
<td>17JAN12</td>
<td>20JAN12</td>
<td>PT1</td>
<td>MILD</td>
<td>INTERVAL</td>
<td>3</td>
</tr>
<tr>
<td>1001</td>
<td>25</td>
<td>M</td>
<td>WH</td>
<td>A</td>
<td>08JAN12</td>
<td>AE2</td>
<td>9</td>
<td>15JAN12</td>
<td>26JAN12</td>
<td>PT2</td>
<td>SEVERE</td>
<td>INTERVAL</td>
<td>3</td>
</tr>
</tbody>
</table>
PREPARE AN INDIVIDUAL SUBJECT’S DATA

The preparation of an individual subject’s data can be broken down into 11 steps:

P1. Subset the subject from the entire dataset
P2. Find the minimum/maximum dates and days
P3. Create a dataset of all subject-level variables
P4. Remove end dates/days from everything except intervals
P5. Append legend group and display type to start/end days
P6. Merge this back with subject-level variables
P7. Merge the blank dataset to fill in any holes
P8. Display “abnormal” results differently with new variables
P9. Set min/max days for “no data” and “date missing” records
P10. Set different variables for the different display types
P11. Sort the data and pre-populate legend groups as needed

Note: For the profiles generated in Figure 1, the days (x-axis) were not held constant across all subjects. It is easy to move Step P2 to the initialization of the profile (as Step I8) to ensure consistent x- and y-axes for all subjects.

The result of these steps leads to a dataset that has additional variables, mostly filled with empty fields. The reason for this is how we will display the profile in the next section. Essentially, since each variable type is drawn differently, it must appear in its own column. Consequently, a particular subject will have a lot of “holes” in their profile as 20+ columns exist to display only one data point.

DISPLAY THE SUBJECT’S DATA

The display of the subject’s data can be broken down into 6 steps:

D1. Create header and footer datasets
D2. Initialize the PDF using the defined template
D3. Display title information for the profile
D4. Display study-specific demographic information
D5. Draw the patient profile, displaying each type of data differently
D6. Display the footer

Step D4 is the most complicated portion of the profile, as it requires the user to modify the SAS code. The rest of the code can be run based off of the prescribed input dataset.

The results of the above dataset are shown in Figure 2 (below).

ADDITIONAL FEATURES

Presenting the profiles to other clinicians, two requests came up consistently: Show trends in addition to numeric values for lab values and split the profile across multiple pages. The latter is especially useful when there is a long duration of data or multiple observations occur close together and appear illegible.

These suggestions were incorporated in the general code that appears below.

LIMITATIONS

The code used is implemented in SAS 9.2. As such, there can be issues with the colors of the graph and the entries in the legend. This is discussed in depth by Cheng and Flavin (2013). A nice solution in SAS 9.3 is the use of a discrete attribute map, which can be incorporated via the DATTRMAP option. For the code below, the user needs to manually update the code in Step P11.
Data Extraction Date: 25JUN13:13:38:43
THE CODE

/* (I1) Read the date of the dataset to determine the data extraction date */
data lastmod(keep=modate);
  set sashelp.vtable;
  where libname = "STATDATA" and memname = "PP_EXAMPLE";
run;
data _null_
  set lastmod;
  call symputx("last_mod", put(modate, DATETIME16.));
run;
/* (I2) Load the input dataset and add abbreviations for the display types */
data final;
  set statdata.pp_example;
  /* NOTE: Update this dataset as necessary */
  DISP_ = substr(DTYPE,1,1);
run;
proc sort data = final;
  by subjid line stdtn;
run;
/* (I3) Determine the number of lines to print in the profile */
proc sort data = final out = numlines nodupkey;
  by descending line;
run;
data numlines;
  set numlines;
  if _N_ = 1;
  call symputx("nline", line);
  call symputx("maxy", line-1);
run;
/* (I4) Create a blank dataset to fill in holes for subjects */
data final;
  set final;
  stdyn = stdtn - fdosdt; endyn = endtn - fdosdt;
run;
/* (I5) Use the blank dataset to assign formats to the y-values */
data blank;
  set blank;
  length value $200. dtype $15.
  Value = "No Data To Report"; DTYPE = "NO_DATA"; y = &nline. - line + 0;
run;
/* (I6) Define Custom style for colors and fonts */
proc template;
  define Style styles.PatientProfile;
    parent = styles.printer;
    ... /* NOTE: The user needs to specify their own styles here */
end;
run;
/* (I7) Populate Study Day for all of the dates */
data final;
  set final;
run;
/* The following macro will create the patient profile for a subject */
%macro prep_sub(subjid=,min=,max=);
/* (P1) Subset the data for the specific subject */
data sub&subjid.;
   set final;
   where SUBJID = "&subjid.";
run;
/* (P2) Find the minimum and maximum dates/days */
data _null_;  
   retain dateMin dateMax dayMin dayMax;
   set sub&subjid.
      end=last;
   %if &min.^= %then %do;
      dayMin = &min.;
      dateMin = &min. + fdosdtn;
      if STDYN < &min. then do;
         STDTN = .;
         STDYN = .;  end;
      if ENDYN < &min. then do;
         ENDTN = .;
         ENDYN = .;  end;
   %end;
   %if &max.^= %then %do;
      dayMax = &max.;
      dateMax = &max. + fdosdtn;
      if STDYN > &max. then do;
         STDTN = .;
         STDYN = .;  end;
      if ENDYN > &max. then do;
         ENDTN = .;
         ENDYN = .;  end;
   %end;
   dateMin=min(STDTN, dateMin);
   dateMax=max(STDTN, ENDTN, dateMax);
   dayMin=min(STDYN, dayMin);
   dayMax=max(STDYN, ENDTN, dayMax);
%global mindate maxdate minday maxday;
   if last then do;
      call symputx('mindate', strip(dateMin-1));
      call symputx('maxdate', strip(dateMax+1));
      call symputx('minday', strip(dayMin-1));
      call symputx('maxday', strip(dayMax+1));
   end;
run;
/* (P3) Create a dataset of all of the subject-level variables */
data sub&subjid._sl(drop=stdtn stdyn endtn endyn label line valu: dtype lgrp disp_);
   set sub&subjid.
      by subjid;
   if first.subjid;
run;
/* (P4) Remove the end dates/days for all Display Types except Intervals */
data sub&subjid.;
   set sub&subjid.
      if DTYPE ^= "INTERVAL" and STDTN = ENDTN and STDYN = ENDYN then do;
      ENDTN = .;
      ENDYN = .;  end;
run;
/* (P5) Reshape the data using transpose, appending lgrp to stdyn (endyn) */
proc sort data=sub&subjid. out = sub&subjid.;
   by subjid line label value value2 dtype disp_ stdtn endtn lgrp;
run;
proc transpose data=sub&subjid. out = sub&subjid. ;
   var stdyn endyn;
   by subjid line label value value2 dtype disp_ stdtn endtn lgrp;
run;
proc sort data=sub&subjid.;
   by subjid line label value value2 dtype disp_ stdtn endtn;
run;
data sub&subjid.;
   set sub&subjid.;
   _name1_ = strip(_name_)||strip(LGRP)||"_"||strip(DISP_);
run;
proc transpose data=sub&subjid. out = sub&subjid. (drop=_name_ ENDYN0_A ENDYN0_D ENDYN1:
   ENDYN2:);
   var coll;
   by subjid line label value value2 dtype stdtn endtn;
   id _name1_;  
run;
/* (P6) Merge the subject-levels variables back */
data sub&subjid.;
   merge sub&subjid._sl sub&subjid.;
by subjid;
run;
/* (P7) Merge the blank dataset with the subject dataset */
data blank;
  set blank;
  subjid = "&subjid.";
run;
data sub&subjid.;
merge blank sub&subjid.(drop=LABEL);
by subjid line;
run;
/* (P8a) If the interval is flagged as "_ABN_", display it differently */
data sub&subjid.;
  set sub&subjid.;
  if stdyn0_I ^= . and strip(upcase(value2)) = "_ABN_" then do;
    stdyn0_I_ab = stdyn0_I;
    endyn0_I_ab = endyn0_I;
    stdyn0_I = .;
    endyn0_I = .;
  end;
run;
/* (P8b) If single values are flagged as "_ABN_", display them differently */
data sub&subjid.;
  set sub&subjid.;
  if stdyn1_A ^= . and strip(upcase(value2)) = "_ABN_" then do;
    stdyn1_A_ab = stdyn1_A;
    stdyn1_A = .;
  end;
run;
/* (P9) Set min/max days for "NO_DATA" and "D_MISS" and center text to display */
data sub&subjid.;
  set sub&subjid.;
  if stdyn0_A ^= . and endyn0_A = then do;
    stdyn0_M = &minday.;
    endyn0_M = &maxday.;
    mid_M = (&minday. + &maxday.)/2;
  end;
  else if DTYPE = "NO_DATA" then do;
    stdyn0_N = &minday.;
    endyn0_N = &maxday.;
    mid_N = (&minday. + &maxday.)/2;
  end;
run;
/* (P10a) Display "SPARK" type differently */
data _null_;  
  set sub&subjid.;
  by line;
  retain sparkmin sparkmax;
  if first.line then do;
    sparkmin = .;
    sparkmax = .;
  end;
  if stdyn1_S ^= . then do;
    sparkmin = min(sparkmin,value);
    sparkmax = max(sparkmax,value);
  end;
  if last.line then do;
    call symputx("min"||strip(put(line,2.)),sparkmin);
    call symputx("max"||strip(put(line,2.)),sparkmax);
  end;
run;
data sub&subjid.;
  set sub&subjid.;
  by line;
  retain sparkmin sparkmax sparkran;
  if first.line then do;
    sparkmin = symget("min"||strip(put(line,2.)));
  end;
  if stdyn1_S ^= . then do;
    sparkmax = symget("max"||strip(put(line,2.)));
    sparkran = sparkmax - sparkmin;
    y_spark = y - 0.4 + 0.8*(value*1-sparkmin)/sparkran;
  end;
run;
/* (P10b) Create separate variables for "value" to displayed for each type */
data sub&subjid.;
  set sub&subjid.;
  if stdyn0_A ^= . then stdyn0_A_value = value; /* AS_IS values that are not coded */
  if stdyn1_S ^= . then do; /* SPARK values (both abnormal and not abnormal) */
    if strip(upcase(value2)) = "_ABN_" then stdyn1_S_ab_value = value;
    else stdyn1_S_value = value;
  end;
  if mid_M ^= . then mid_M_value = strip(value)||" (No Date)"; /* D_MISS values */
  if mid_N ^= . then mid_N_value = value; /* NO_DATA values */
if stdyn1_A ^= . then stdyn1_A_value = value; /* AS_IS values that *are* coded */
if stdyn1_A_ab ^= . then stdyn1_A_ab_value = value;
if stdyn2_C ^= . then stdyn2_C_value2 = value2; /* C_DOT values */
if stdyn3_I ^= . then stdyn3_I_value2 = value2; /* INTERVAL values for AEs */
/* NOTE: Additional groups would need to be updated/added here */
run;
/* (P11a) Sort the data and create a final dataset for the subject */
proc sort data = sub&subjid. out = sub&subjid._final;
by line label stdtn;
run;
/* (P11b) Manually populate legend groups to clean the legend */
data sub&subjid._final;
set sub&subjid._final;
if stdyn2_C_value2 = "" then stdyn2_C_value2 = "NEGATIVE"; /* Colored Dot */
if stdyn3_I_value2 = "" then do; /* AE Intervals */
  if _N_ = 1 then stdyn3_I_value2 = "MILD";
  else if _N_ = 2 then stdyn3_I_value2 = "MODERATE";
  else stdyn3_I_value2 = "SEVERE";
end;
run;
%mend prep_sub;

/* Create the graphical patient profile */
%macro disp_sub(subjid=,min=,max=,num=);
  /* (D1a) Create a patient header */
  proc sort data = sub&subjid._sl out = header_info nodupkey;
    by subjid;
  run;
  %macro insert1(label=,var=,inhibit=LR);
    adj.format_cell(data: "&label: "||strip(&var), overrides: "just=l", inhibit: "&inhibit.");
  %mend insert1;
  %macro insert2(label=,var=,inhibit=,unit=);
    adj.format_cell(data: "&label: "||strip(&var)||" &unit", overrides: "just=l", inhibit: "&inhibit.");
  %mend insert2;
  /* (D1b) Create a patient footer */
  proc sort data = sub&subjid._sl out = footer_info(keep=subjid) nodupkey;
    by subjid;
  run;
  data footer_info;
    set footer_info;
    extrcdtm = "&last_mod.";
  run;
  /* (D2) Initialize the PDF using the defined template */
  ods listing close; ods escapechar="";
  options orientation=landscape nonumber nodate;
  ods pdf notoc startpage=no %if &num. ^= %then %do; file="&out.\&subjid.-Patient-Profile-&num..pdf" %end; %else %do; file="&out.\&subjid.-Patient-Profile.pdf" %end;
  style=styles.PatientProfile; title; footnote;
  data _null_; set header_info;
    declare odsout adj();
  /* (D3a) Add the title of the patient profile */
    adj.table_start(overrides: "width=100pct borderwidth=0"); adj.row_start();
    adj.format_cell(data: " Patient Profile for Study XXX-XXXX ", split: "|",
      overrides: "just=c font_size=13pt font_weight=bold");
    adj.row_end(); adj.table_end();
  /* (D3b) Title for the header data */
    adj.table_start(overrides: "width=100pct borderwidth=2"); adj.row_start();
    adj.format_cell(data: " Demographic Information ", overrides: "just=c
      font_size=11pt font_weight=bold", inhibit: "BLR");
    adj.row_end(); adj.table_end();
  /* (D4) First and Second row of header data */
    adj.table_start(overrides: "width=100pct borderwidth=2"); adj.row_start();
Creating Graphical Patient Profiles Using SAS®, continued

%insert1(label=Subject ID, var=Subjid, inhibit=TLR);
%insert2(label=Age, var=Age, unit=yr, inhibit=TLR);
%insert1(label=Sex, var=Sex, inhibit=TLR);
%insert1(label=Race, var=put(Race,$RACE.), inhibit=TLR);
%insert1(label=Ethnicity, var=put(Ethnic,$ETHNIC.), inhibit=TLR);
%insert1(label=Treatment, var=trta, inhibit=TLR);
adj.row_end(); adj.table_end();
run;

/* (D5) Draw the profile */
ods listing style=PatientProfile image_dpi=100;
ods graphics / reset width=10in height=4.5in imagename="Sub&_subjid._PP_Timeline"
border=off;

proc sgplot data=sub&subjid._final noautolegend nocycleattrs;
  /* Draw the duration events */
  vector x=endyn0_I y=y / xorigin=stdyn0_I yorigin=y noarrowheads
  lineattrs=(thickness=7px);
  vector x=endyn0_I_ab y=y / xorigin=stdyn0_I_ab yorigin=y noarrowheads
  lineattrs=(thickness=7px color=red);
  /* Draw start and end events */
  scatter x=stdyn0_I y=y / markerattrs=(size=10px symbol=diamondfilled);
  scatter x=stdyn0_I_ab y=y / markerattrs=(size=10px symbol=diamondfilled color=red);
  scatter x=endyn0_I y=y / markerattrs=(size=10px symbol=diamondfilled);
  scatter x=endyn0_I_ab y=y / markerattrs=(size=10px symbol=diamondfilled);
  /* Plot entries with No Data to display */
  vector x=endyn0_N y=y / xorigin=stdyn0_N yorigin=y noarrowheads
  lineattrs=(thickness=11px color=lightyellow);
  scatter x=mid_N y=y / markerchar=mid_N_value makercharattrs={family='Lucida Console' size=7};
  /* Plot data with missing dates */
  scatter x=mid_M y=y / markerchar=mid_M_value markercharattrs={family='Lucida Console' size=7};
  /* Add a reference line at Study Day 0 */
  reline 0 / axis=x lineattrs=(thickness=1 color=lightgrey);
  /* Plot colored dot tests */
  scatter x=stdyn2_C y=y / markerattrs=(size=10px symbol=circlefilled)
  group=stdyn2_C_value2 name='legend2';
  /* Plot non-numeric events */
  scatter x=stdyn0_D y=y / markerattrs=(size=10px symbol=circlefilled);
  /* Plot numeric/character results */
  scatter x=stdyn0_A y=y / markerchar=stdyn0_A_value markercharattrs={family='Lucida Console' size=7 color=black};
  /* Plot (normal/abnormal) numeric/character results */
  scatter x=stdyn1_A y=y / markerchar=stdyn1_A_value markercharattrs={family='Lucida Console' size=7 color=black} name='legend11' legendlabel='Normal';
  scatter x=stdyn1_A_ab y=y / markerchar=stdyn1_A_ab_value
  markercharattrs={family='Lucida Console' size=7 color=red} name='legend12'
  legendlabel='Abnormal';
  /* Plot spark lines and (abnormal) numeric results */
  series x=stdyn1_S y=y_spark / lineattrs=(color=lightgrey pattern=solid);
  scatter x=stdyn1_S y=y_spark / markerchar=stdyn1_S_value
  markercharattrs={family='Lucida Console' size=7 color=black} name='legend11'
  legendlabel='Normal';
  scatter x=stdyn1_S y=y_spark / markerchar=stdyn1_S_ab_value
  markercharattrs={family='Lucida Console' size=7 color=red} name='legend12'
  legendlabel='Abnormal';
  /* Draw the adverse events */
  vector x=endyn3_I y=y / xorigin=stdyn3_I yorigin=y noarrowheads
  lineattrs=(thickness=7px pattern=solid) group=stdyn3_I_value2 name='legend3';
  scatter x=stdyn3_I y=y / markerattrs=(size=10px symbol=diamondfilled)
  group=stdyn3_I_value2;
  scatter x=endyn3_I y=y / markerattrs=(size=10px symbol=diamondfilled)
  group=stdyn3_I_value2;
CONCLUSION

Patient profiles are an excellent way to convey relationships between various sources of data in a compact manner. The code introduced in this paper is aimed at facilitating the creation of a graphical patient profile in SAS® 9.2 using the SG procedures. The profiles are generated based on a standard dataset structure, BDS, with additional variables to control how the data is to be presented. These display types can be implemented with minimal modification. The profiles use common rows to facilitate subject comparisons.

In practice, the patient profiles helped reviewers to quickly evaluate subjects, but additional information was needed and provided in the form of text-based listings. The main utility of these profiles was to focus attention on specific data worth exploring in more detail.

REFERENCES

Matange, S. Patient Profile Graphs Using SAS®. PharmaSUG. 2013; Paper MS14-SAS.

CONTACT INFORMATION

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

Name: William Garner  
Company: Gilead Sciences, Inc.  
Address: 333 Lakeside Dr  
City, State ZIP: Foster City, CA 94404  
Work Phone: (650) 522-5691  
E-mail: will.garner@gilead.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.