

**PharmaSUG China 2018 – Paper DV-52**  
**Multi-Arm CONSORT Diagrams with SAS**

Sanjay Matange, SAS Institute Inc.

Prashant Hebbar, SAS Institute Inc.

## **ABSTRACT**

In Clinical trials, Consolidated Standards of Reporting Trials (CONSORT) flow diagrams are an important part of the randomized trials report. These diagrams present a bird's eye view of the flow of patients through the different stages of the trial.

The SG Procedures do not support a statement for drawing these diagrams. But with some data processing steps we can draw commonly used CONSORT diagrams using the SGPLOT procedure. In this paper, we show how to harness power of the SGPLOT Procedure to create these diagrams.

## **INTRODUCTION**

A CONSORT diagram shows the flow of subjects through each stage in a clinical trial. Typical flow progression is enrollment, allocation to treatment, follow-up or disposition status and analysis. Such a flow diagram is considered an important tool for assessing a trial. (Hopewell *et al*, 2011).

Previously published work for drawing these diagrams with SAS primarily used Rich Text Format (RTF) templates (Carpenter and Fisher, 2012; Mallavarapu and Shults, 2016). Here we show how to use the SGPLOT procedure to draw commonly used multi-Arm CONSORT diagrams. We assume that the requisite counts used in the diagram have already been computed. Some DATA step pre-processing is needed to create the data set for the graph.

A program to create CONSORT diagrams using SGPLOT procedure was previously published in the *Graphically Speaking* blog post titled "Outside-the-box: CONSORT diagram". After that, the code in the program was cleaned and improved and presented in the paper "CONSORT Diagrams with SGPLOT Procedure" at the 2018 PharmaSUG conference.

In this paper, we simplify the process for many standard multi-arm layouts that are commonly used in the industry. We also use some new SGPLOT features released with SAS 9.4M5. However, the program can be easily modified to use the features provided with older 9.4 releases.

## **CONSORT DIAGRAM DATA AND STRUCTURE**

The process used in this paper assumes a standard structure and layout for CONSORT diagrams commonly used in the industry. Examination of various such diagrams on the web shows the use of a standard flow for the diagrams. These typically start with the Enrollment stage that includes Assessment, Exclusion and Randomization. This is followed by the Allocation, Follow-Up and Analysis stages. The last three stages have multiple columns, one for each arm of the study. While the details of some diagrams may vary, the basic steps are essentially the same.

For our purpose, we have used the format shown in Mallavarapu and Shults (2016) which is for a 4-Arm study.

The CONSORT diagram is essentially composed of boxes linked by directed lines. The text in these boxes may be horizontal or vertical with center or left alignment.

## **PLOT STATEMENTS**

To implement these elements in PROC SGPLOT, we use the following plot statements:

- POLYGON statement to draw the boxes (with background fill and without).
- SERIES statement to draw the links and arrows.
- TEXT statement to render the various text elements.

## **FOUR ARM CONSORT DIAGRAM**

To describe how the program and macros work, it is useful to start with the end result in mind. Figure 1 shows the output for a 4-ARM CONSORT diagram. To make the production of such a diagram easier, we are assuming that the structure of these diagrams is very standard as shown in Figure 1. Note the features of the diagram as follows:

1. The diagram shows four "Stages", viz., Enrollment, Allocation, Follow-Up and Analysis.

2. Equal vertical space is provided for each stage, with a header on the left showing the stage name.
3. The first stage, Enrollment, has three nodes with custom layout and links.
4. The bottom three stages, viz. Allocation, Follow-Up and Analysis all have evenly spaced columns, one for each ARM.
5. Between the Enrollment and Allocation stages, the diagram forks to the multiple arms.

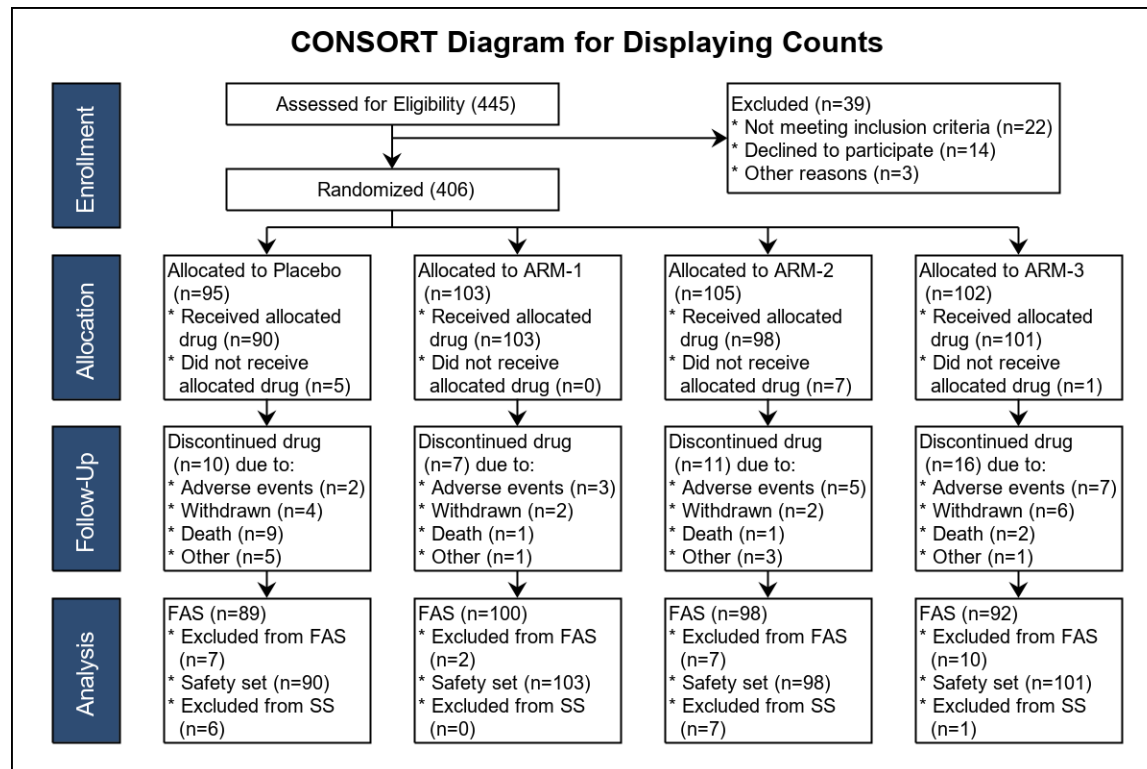


Figure 1 –Four-Arm diagram

## DATA PREPARATION

For preparing the data for the diagram we use a drawing area of [0,1] horizontal (X axis) space and [0, 1] vertical (Y axis) space, with the [0, 0] origin at the top left. Data sets for the link and box elements are created in this space. The layout of the diagram is set as a 5 x 5 grid, top to bottom and left to right as shown in Figure 2.

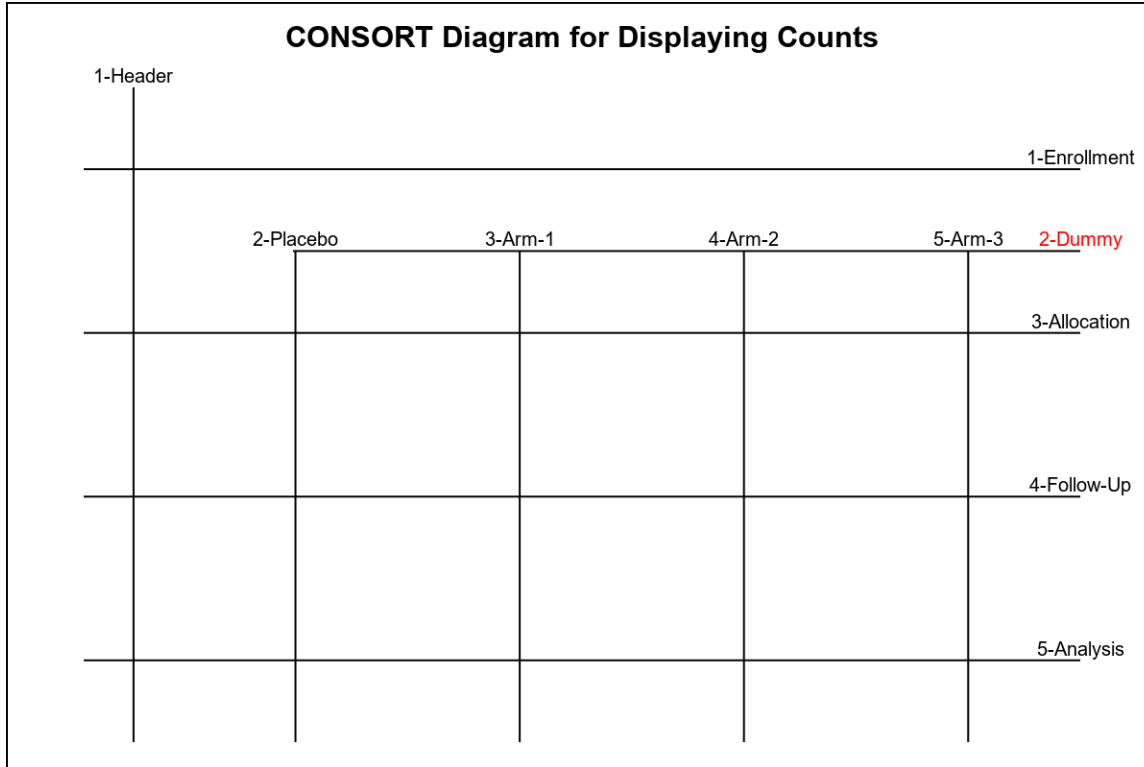
Vertically, each stage of the diagram is allocated 25% of the height, starting from the top. Figure 2 shows the positioning of the boxes for Enrollment, Allocation, Follow-Up and Analysis, from top to bottom. The centerline for each of these stages is shown on the right in Figure 2.

Note, there is an extra Dummy stage between the top Enrollment stage and the Allocation stage. The centerline for this stage is shown in Figure 2 and the label is shown in red. This information is used to draw the horizontal line connecting all 4 arms to the Randomized node.

The Enrollment stage has three boxes for “Assessed”, “Excluded” and “Randomized” that have customized placement. The input data part of the program uses macro variables to define the vertical positions of the three nodes as a fraction of the full diagram height. The horizontal locations of these nodes are generally positioned between the lower sets of columns. The header is positioned on the left aligned with other headers.

Each of the lower three stages is located at equal vertical spacing of 0.25 of the diagram height as shown in Figure 2. The centerline of each stage is shown along with a label on the right. A macro variable is used to set the spacing. Horizontally, a header for each stage is displayed at the left. The rest of the space is equally allocated to 4 arms.

Note, for the last 4 stages of “Dummy”, “Allocation”, “Follow-Up” and “Analysis”, the columns are all on the grid shown in Figure 2, Their centerlines are located by “Header”, “Placebo”, “Arm-1”, “Arm-2” and “Arm-3”. For Dummy stage, the node sizes have effective zero height, so they are not displayed.



**Figure 2 – Layout grid for a 4-Arm diagram**

**PROGRAM STEPS**

The program has three parts that are used in sequence:

- Data input.
- Macro for computing the diagram data.
- Macro to rendering the diagram using the SGPLOT procedure.

**DATA INPUT**

The program only requires the user to provide the (Row, Col) indexes for each node, along with the values for the data to be shown in each box (n1 to n5). Index values for rows is 1 to 5 from top to bottom. Index values for columns is 1 to 5 from left to right as shown in Figure 2. For the Enrollment stage, the column index values 1, 2, 3 and 4 are for Header, Assessed, Excluded and Randomized node each. Partial input data is shown below in Figure 3.

```
input Row Col Stage $ Case $ n1 n2 n3 n4 n5;
1 1 Enrollment Header . . . . .
1 2 Enrollment Assessed 445 . . . .
1 3 Enrollment Excluded 39 22 14 3 .
1 4 Enrollment Randomized 406 . . . .
2 1 Dummy Header . . . . .
2 2 Dummy Placebo . . . . .
2 3 Dummy ARM-1 . . . . .
2 4 Dummy ARM-2 . . . . .
2 5 Dummy ARM-3 . . . . .
3 1 Allocation Header . . . . .
3 2 Allocation Placebo 95 90 5 . .
3 3 Allocation ARM-1 103 103 0 . .
```

**Figure 3 – Input data for CONSORT layout.**

Note, the character variables “Stage” and “Case” correspond to Row and Column. These character columns are included in the input data as an aid to understand the data. The program can be modified to eliminate these variables.

In the data input part of the program, node widths and heights are set using some macro variables. These are saved in the data set generated in the data input part of the program. A value “WF” is also computed and saved in a macro variable that is used for customizing the Excluded note. We will discuss this later in the paper. After the data input part of the program, we call the macro for computing the diagram data, followed by the macro to render the diagram.

```

%let rowSpacing=0.25;
%let pad=0.02;
%let headerWidth=0.1;
%let nColumns=4;
%let yAssess=0.2;
%let yRandom=0.7;
%let yExclude=0.4;
%let yHeader=0.5;

/*--Diagram Layout--*/
data layout;
  length Stage $12 case $12;
  format Row row. Col col.;
  input Row Col Stage $ Case $ n1-n5;

  /*--Pad to increase the width of the Exclude box when columns=3--*/
  wf=(4-&nColumns)*0.05;
  call symput("wf", wf);

  h=&rowSpacing;
  if Stage='Dummy' then h=2*&pad;
  dx=(1-&headerWidth)/&nColumns;

  select (case);
    when ('Header') w=&headerWidth;
    when ('Randomized', 'Assessed') do; w=(&nColumns-1)*dx/2; h=0.1; end;
    when ('Excluded') do; w=wf+(&nColumns-1)*dx/2; h=0.2; end;
    otherwise w=dx;
  end;

  /* Update last 5 columns for n1-n5 counts */
  datalines;
1  1  Enrollment  Header      .      .      .      .
<other data rows>
;
run;

```

**Figure 4 – Input data program**

## MACRO FOR COMPUTING DIAGRAM DATA

The **%consortData (inData=, arms=, outData=)** macro is used to compute the data for the diagram. It has the following parts to compute all the data needed to render the diagram:

- A data step to compute the center of the nodes for all stages of the diagram based on the grid indices provided.
- A data step to compute the polygon vertices for each node.
- Utility macros are used to create the text for each node using the n1-n5 parameters provided.
- A data step to compute the links between the nodes.
- The node and link data are then combined into one data set.

Separate polygons are created for the filled and unfilled nodes. Columns (xf, yf) are used for the filled polygons and columns (x, y) for the unfilled ones. For the node text, again two separate lists are created. One for the vertical label of the headers in the column “vLabel”, and one for the text in each node in the column “hLabel”. The utility macros are used to generate the code needed to create the standard node labels including the values n1 to n5 provided by the user in the Input Data part of the program. See the full code in Appendix-1.

## MACRO FOR RENDERING THE DIAGRAM

The `%consortDiagram (data=, fillColor=)` macro is used to render the diagram. The macro uses the data set created by the `%consortData ()` macro and uses the SGPLOT procedure. The entire code for the macro including the SGPLOT procedure step is shown in Figure 5.

```
%macro consortDiagram(data= , fillColor=stgb);

proc sgplot data=consort noborder noautolegend;
  /*--Filled boxes--*/
  polygon id=pid x=xf y=yf / fill outline
    fillattrs=(color=&fillcolor) lineattrs=graphdatadefault;

  /*--vertical text--*/
  text x=xl y=yl text=vlabel / rotate=90 textattrs=(size=9 color=white);

  /*--Empty boxes--*/
  polygon id=pid x=x y=y / lineattrs=graphdatadefault;

  /*--horizontal text, both left and center aligned--*/
  text x=xl y=yl text=hLabel / splitchar='.' splitpolicy=splitalways
    position=position;

  /*--Links with arrow heads--*/
  series x=xlink y=ylink / group=lid lineattrs=graphdatadefault
    arrowheadpos=end arrowheadshape=barbed arrowheadscales=0.4;

  /*--Links without arrow heads--*/
  series x=xlink2 y=ylink2 / group=lid2 lineattrs=graphdatadefault;

  xaxis display=none min=0 max=1 offsetmin=0 offsetmax=0;
  yaxis display=none min=0 max=1 offsetmin=0 offsetmax=0 reverse;
run;

%mend consortDiagram;
```

**Figure 5 – Input data program**

The details of the SGPLOT program are very straightforward, as seen in Figure 5. The coordinates for the diagram are in normalized 0.0 – 1.0 space in both width and height, with the origin at the top left. For that, it is necessary to set the `offsetMin` and `offsetMax` for the axes and also use the `REVERSE` option on the `YAXIS` statement.

## MULTIPLE LAYOUTS

The program supports different layouts of the CONSORT diagram. Commonly used CONSORT diagrams have two, three or four arms. The program uses the `nColumns` macro variable to set the number of columns to be used for the layout of the lower part of the diagram. The macro parameter `arms=` is used to inform the macro of the actual number of columns in the data.

Using a combination of these macro variable and parameter, we can create CONSORT diagrams for the following different cases. Note the settings for the macro parameter `Arms` and the macro variable `nColumns`. The data must always include all 5 stages (including `Row=2` for Dummy). When `arms < 4`, any data for `Col > arms`, it is ignored.

- `Arms=4, nColumns=4`: Figure 1
- `Arms=3, nColumns=4`: Figure 6
- `Arms=3, nColumns=3`: Figure 7
- `Arms=2, nColumns=4`: Figure 8
- `Arms=2, nColumns=3`: Figure 9.

Note: When using a 4-column layout, the nodes in the Enrollment stage are positioned in the center of the lower pair of columns. Also, the widths of the nodes are set to 1.5 times the width of the lower nodes. However, when using a 3-column layout, the above does not work well. In this case we use the `WF` macro variable to adjust the nodes position and widths. Various diagrams are shown in Figures 6, 7, 8 and 9.

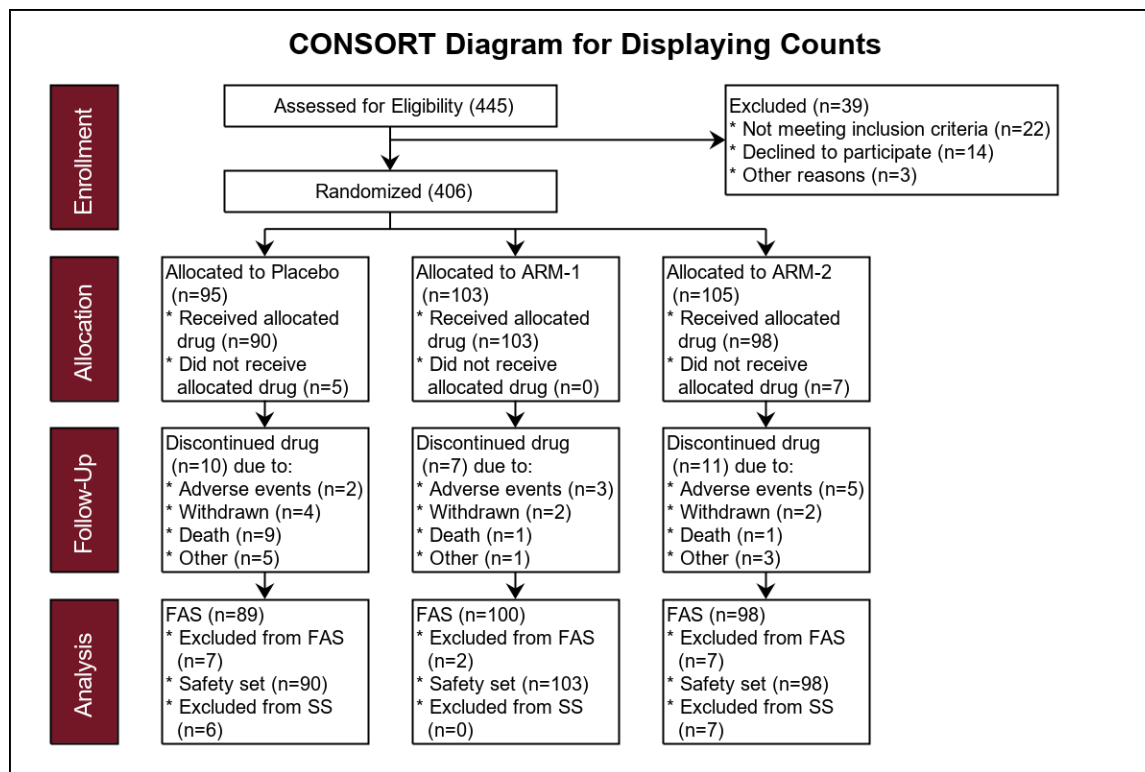


Figure 6 – 3 Arms, 4 Columns – with empty space on the right

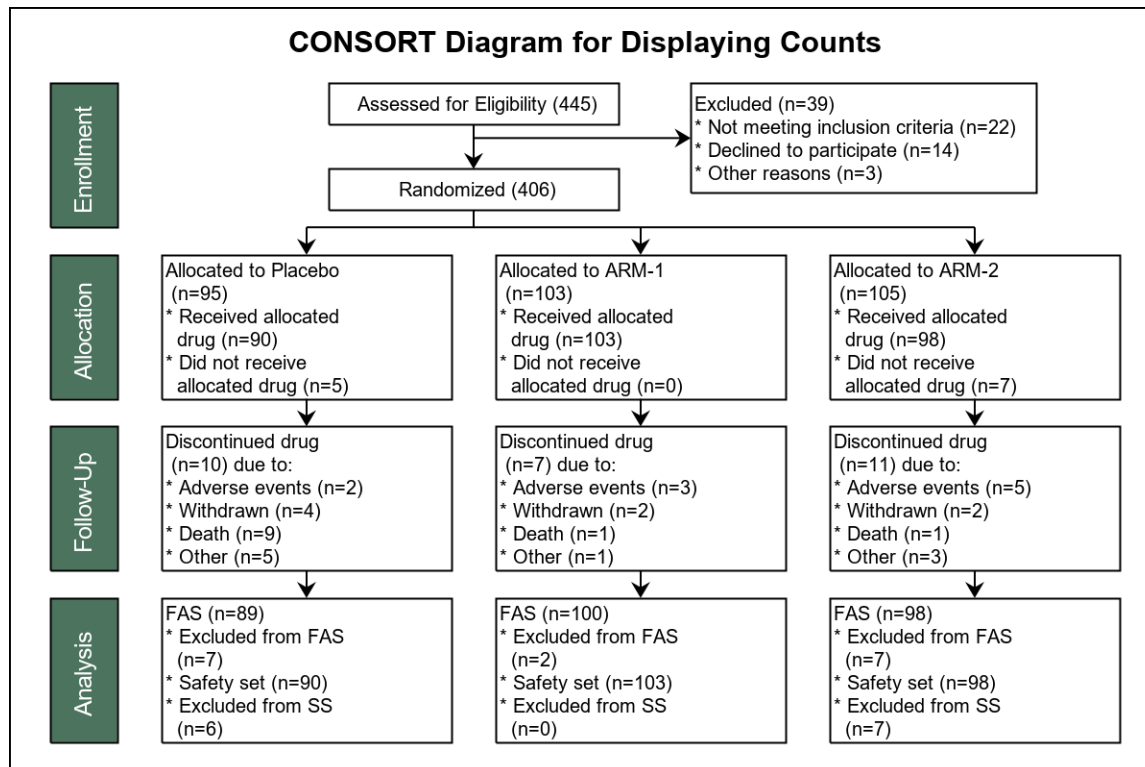


Figure 7 – 3 Arms, 3 Columns –no empty space on the right

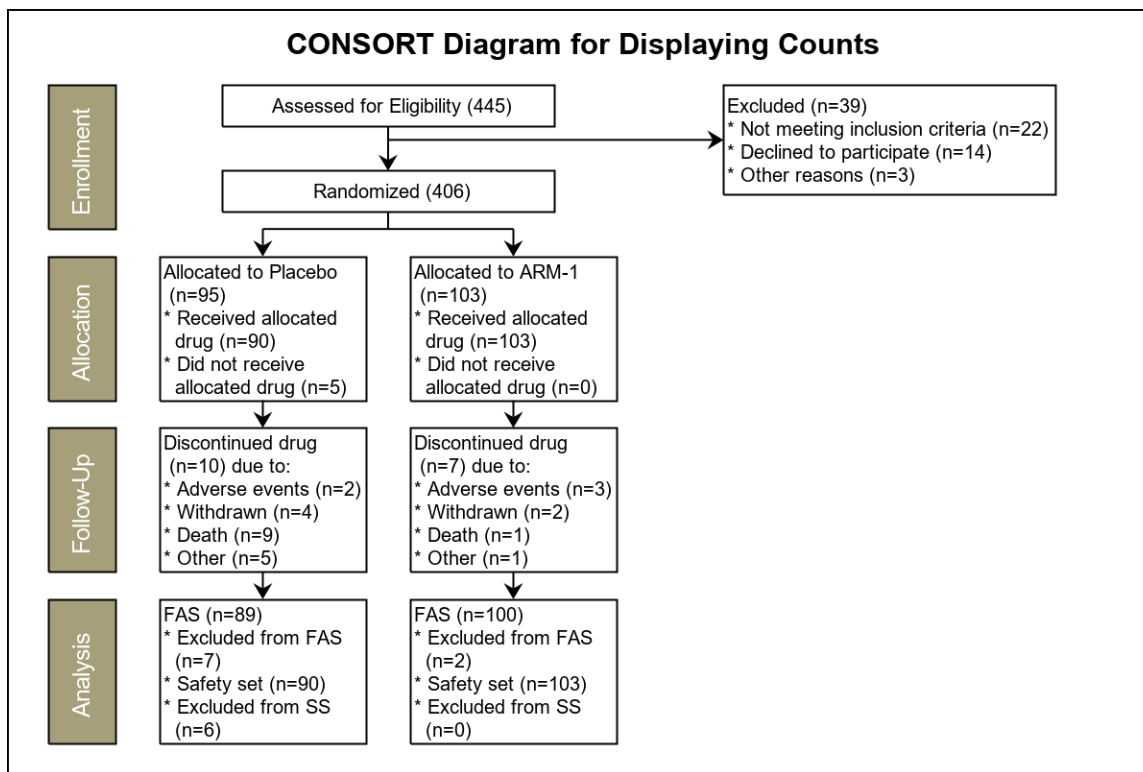


Figure 8 – 2 Arms, 4 Columns – Lot of empty space on the right

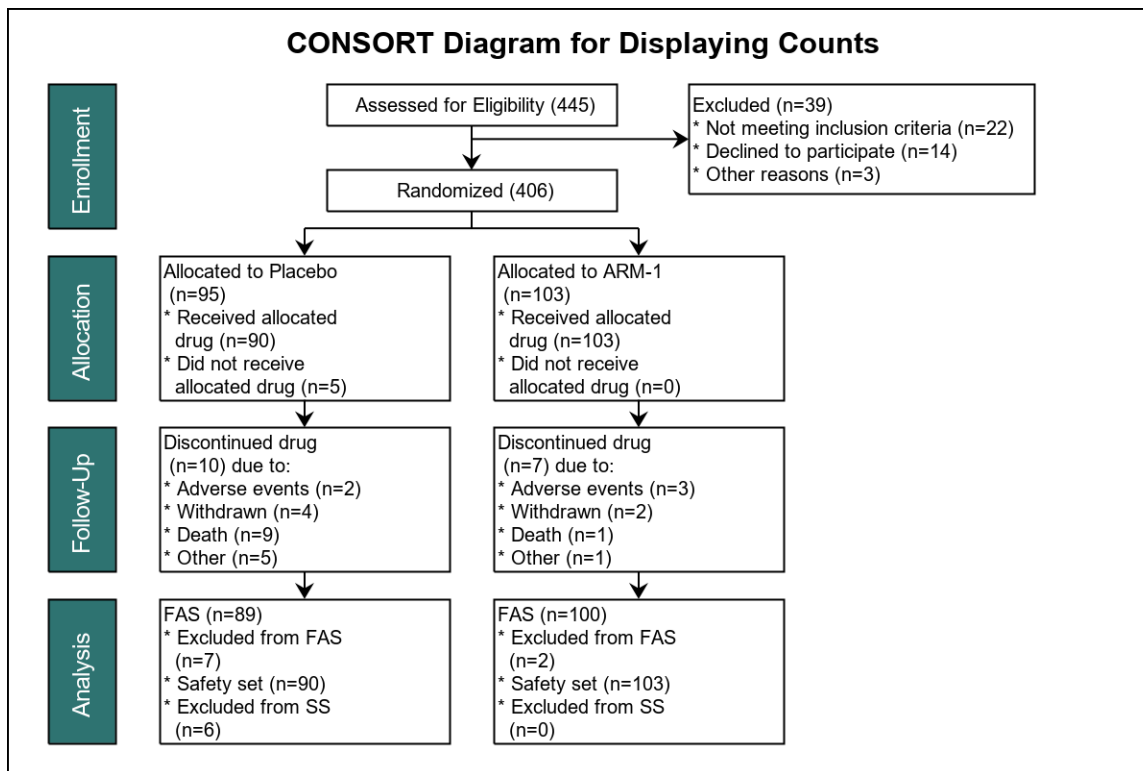


Figure 9 – 2 Arms, 3 Columns – Some empty space on the right.

## CONCLUSION

We have demonstrated that a CONSORT diagram can be created using only SAS code, thus reducing the complexity of using additional applications or template files. The output can be generated in a variety of formats supported by ODS Graphics. The computing of the data for the consort diagram is separated from the rendering. This allows the diagram to be computed once, and then rendered to different destinations without the need to recompute the data.

The diagrams can be laid out in a 4-Column or a 3-Column layout. Some users may prefer to render a 3-Arm diagram in a 3-Column layout to fill the space. Also note that since PROC SGPLOT is based on GTL, this graph can be similarly created using a GTL template and the SGRENDER procedure.

## REFERENCES

- Hopewell, S., *et al.* 2011. "Reporting of participant flow diagrams in published reports of randomized trials." *Trials*, 12:253.
- Carpenter, A. and Fisher D. G. 2012. "Reading and Writing RTF Documents as Data: Automatic Completion of CONSORT Flow Diagrams." *Proceedings of PharmaSUG 2012*, TF16. Available at <https://www.pharmasug.org/proceedings/2012/TF/PharmaSUG-2012-TF16.pdf>
- Mallavarapu, A. and Shults, D. 2016. "CONSORT Diagram: Doing it with SAS." *Proceedings of PhUSE 2016*, Poster PP03. Available at <http://www.phusewiki.org/docs/Conference%202016%20PP%20Papers/PP03.pdf>
- Hebbar, P. and Matange, S. 2018. "CONSORT Diagrams with SGPLOT Procedure." *Proceedings of PharmaSUG 2018*, Available at <https://www.pharmasug.org/proceedings/2018/DV/PharmaSUG-2018-DV24.pdf>
- Matange, S. and Heath, D., 2011. *Statistical Graphics Procedures by Example: Effective Graphs Using SAS®*. Cary, NC: SAS Institute Inc.
- Matange, Sanjay. "Graphically Speaking." Available at <https://blogs.sas.com/content/graphicallyspeaking/2016/10/20/outside-box-consort-diagram/>.

## ACKNOWLEDGMENTS

Thanks to Shane Rosanbalm of Rho, Inc. Chapel Hill, North Carolina, for his insights and suggestions.

## RECOMMENDED READING

- *SAS® 9.4 ODS Graphics: Procedures Guide*

## CONTACT INFORMATION

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

Sanjay Matange  
SAS Institute, Inc.

[sanjay.matange@sas.com](mailto:sanjay.matange@sas.com)

Prashant Hebbar  
SAS Institute, Inc.

[prashant.hebbar@sas.com](mailto:prashant.hebbar@sas.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.

## APPENDIX – FULL SOURCE CODE

### Macros to compute diagram data:

```
/*--Utility macros--*/  
%macro enrollExcluded(n1=, n2=, n3=, n4=);  
cat ("Excluded (n=", &n1.,  
    ")* Not meeting inclusion criteria (n=", &n2.,  
    ")* Declined to participate (n=", &n3.,  
    ")* Other reasons (n=", &n4.,  
    ")");  
%mend enrollExcluded;
```



CONSORT Diagram using SGPLOT Procedure, continued

```

%macro allocation(arm=, n1=, n2=, n3=);
cat ("Allocated to ", &arm.,
    ". (n=", &n1.,
    ").* Received allocated. drug (n=", &n2.,
    ").* Did not receive. allocated drug (n=", &n3.,
    ")");
%mend allocation;

%macro followup(n1=, n2=, n3=, n4=, n5=);
cat ("Discontinued drug. (n=",&n1.,
    ") due to:.* Adverse events (n=", &n2.,
    ").* Withdrawn (n=", &n3.,
    ").* Death (n=",&n4.,
    ").* Other (n=", &n5.,
    ")");
%mend followup;

%macro analysis(n1=, n2=, n3=, n4=);
cat ("FAS (n=",&n1.,
    ").* Excluded from FAS. (n=",&n2.,
    ").* Safety set (n=", &n3.,
    ").* Excluded from SS. (n=",&n4.,
    ")");
%mend analysis;

/*--Compute all the box and link data, given the counts data--*/
%macro consortData (inData=, arms=, outData=);

    /*--Consort diagram Node locations--*/
    data node_info;
    set &inData.;

    /*--Compute node center (x, y)--*/
    select (Row);
    /*--Analysis, Follow-Up and Allocation--*/
    when (5, 4, 3) do; /*--Analysis, Follow-Up and Allocation--*/
    select (Col);
    when (1) do;
        ny=(row-1.5)*&rowSpacing; nx=&headerWidth/2; output; end;
    when (2) do;
        ny=(row-1.5)*&rowSpacing; nx=&headerWidth+w/2; output; end;
    when (3) do;
        ny=(row-1.5)*&rowSpacing; nx=&headerWidth+1.5*w; output; end;
    when (4) if &arms > 2 then do;
        ny=(row-1.5)*&rowSpacing; nx=&headerWidth+2.5*w; output; end;
    when (5) if &arms > 3 then do;
        ny=(row-1.5)*&rowSpacing; nx=&headerWidth+3.5*w; output; end;
    otherwise;
    end;
end;

/*--Dummy Row to draw the top arrows from horz line--*/
when (2) do;
    select (Col);
    when (1) do;
        ny=(row-1)*&rowSpacing-1*&pad; nx=&headerWidth/2; output; end;
    when (2) do;
        ny=(row-1)*&rowSpacing-1*&pad; nx=&headerWidth+w/2; output; end;
    when (3) do;
        ny=(row-1)*&rowSpacing-1*&pad; nx=&headerWidth+1.5*w; output; end;
    when (4) if &arms > 2 then do;
        ny=(row-1)*&rowSpacing-1*&pad; nx=&headerWidth+2.5*w; output; end;
    when (5) if &arms > 3 then do;

```

CONSORT Diagram using SGPLOT Procedure, continued

```

        ny=(row-1)*&rowSpacing-1*&pad; nx=&headerWidth+3.5*w; output; end;
    otherwise;
end;
end;

/*--Enrollment--*/
when (1) do;
    select (Col);
        when (1) do; ny=&yHeader*&rowSpacing; nx=&headerWidth/2; output; end;
        when (4) do; ny=&yRandom*&rowSpacing; nx=&headerWidth+dx; output; end;
        when (2) do; ny=&yAssess*&rowSpacing; nx=&headerWidth+dx; output; end;
        when (3) do; ny=&yExclude*&rowSpacing;
            nx=&headerWidth+(&nColumns-1)*dx+&wf; output; end;
    otherwise;
end;
end;

    otherwise;
end;
run;

/*--Consort diagram Node polygon data--*/
data nodes;
    length vLabel $10 hLabel $200 position $10;
    retain pid 0;
    set node_info;

    nh=h-2*&pad; nw=w-2*&pad;
    pid+1;

/*--Compute node polygons for non-Dummy rows--*/
if row ne 2 then do;
    if col eq 1 then do; /*--Is a Header--*/
        yf=ny-nh/2; xf=nx-nw/2; vlabel=''; output;
        yf=ny-nh/2; xf=nx+nw/2; vlabel=''; output;
        yf=ny+nh/2; xf=nx+nw/2; vlabel=''; output;
        yf=ny+nh/2; xf=nx-nw/2; vlabel=''; output;
        yl=ny; xl=nx; vlabel=stage; output;
    end;
    else do; /*--Not a Header--*/
        y=ny-nh/2; x=nx-nw/2; hlabel=''; output;
        y=ny-nh/2; x=nx+nw/2; hlabel=''; output;
        y=ny+nh/2; x=nx+nw/2; hlabel=''; output;
        y=ny+nh/2; x=nx-nw/2; hlabel=''; output;
        yl=ny;
    end;

/*--set the text for the nodes--*/
if row=5 then do;
    hlabel=%analysis(n1=n1, n2=n2, n3=n3, n4=n4);
    xl=nx-nw/2; position='Right'; end;
if row=4 then do;
    hlabel=%followup(n1=n1, n2=n2, n3=n3, n4=n4, n5=n5);
    xl=nx-nw/2; position='Right'; end;
if row=3 then do;
    hlabel=%allocation(arm=case, n1=n1, n2=n2, n3=n3);
    xl=nx-nw/2; position='Right'; end;

/*--set the text for the nodes in Enrollment stage--*/
if row=1 then do;
    if col=2 then do;
        hlabel=cat ("Assessed for Eligibility (", n1, ")");
        xl=nx; position='Center'; end;
    if col=3 then do;

```

CONSORT Diagram using SGPLOT Procedure, continued

```

        hlabel=%enrollExcluded(n1=n1, n2=n2, n3=n3, n4=n4);
        xl=nx-nw/2; position='Right'; end;
    if col=4 then do;
        hlabel=cat ("Randomized (", n1, ")"); xl=nx; position='Center'; end;
    end;
    output;
end;
end;
run;

/*--Reorder node_info by X & Y--*/
proc sort data=node_info out=nodes_Sorted;
by col row;
run;

/*--Compute the Consort diagram Links between non-Header nodes--*/
data links;
keep stage case lid lid2 xlink ylink xlink2 ylink2;
retain lid 0 xp yp;
set nodes_Sorted (where=(stage ne 'Enrollment' and case ne 'Header')) end=last;
by col row;

lid2=.;

/*--Upper end of link as "previous" (xp, yp)--*/
if first.col then do;
    nh=h-2*&pad;
    yp=ny+nh/2; xp=nx;
end;

/*--Bottom end of link, output both ends, and compute "previous"--*/
else do;
    nh=h-2*&pad;
    ylink=yp; xlink=xp; output;
    ylink=ny-nh/2; xlink=nx; output;
    yp=ny+nh/2; xp=nx; lid+1;
end;

/*--Create the custom links in the Enrollment stage--*/
if last then do;
    call missing (stage, case, xlink, ylink);
    w=&headerWidth;
    dx=(1-w)/&nColumns;
    call missing (lid2, xlink2, ylink2);

    /*--Assessed to Randomize link--*/
    ny=&yAssess*&rowSpacing; nx=&headerWidth+dx; h=0.1; nh=h-2*&pad;
    ylink=ny+nh/2; xlink=nx; output;
    ny=&yRandom*&rowSpacing; nx=&headerWidth+dx; h=0.1; nh=h-2*&pad;
    ylink=ny-nh/2; xlink=nx; output;
    lid+1;

    /*--Horizontal link to Excluded--*/
    ny=&yExclude*&rowSpacing; nx=&headerWidth+1*dx;
    w=(&nColumns-1)*dx/2; nw=w-2*&pad;
    ylink=ny; xlink=nx; output;
    ny=&yExclude*&rowSpacing; nx=&headerWidth+(&nColumns-1)*dx+&wf;
    w=&wf+(&nColumns-1)*dx/2; nw=w-2*&pad;
    ylink=ny; xlink=nx-nw/2; output;
    lid=.;

    /*--Horizontal link over 4 ARMs--*/
    nx=w+dx/2; w=dx;

```

## CONSORT Diagram using SGPLOT Procedure, continued

```
    lid2=0; ylink2=&rowSpacing-&pad; xlink2=&headerWidth+dx/2; output;
    xlink2=&headerWidth+(&arms-0.5)*dx; output;

    /*--Randomize to Horz link--*/
    ny=&yRandom*&rowSpacing; nx=&headerWidth+dx; h=0.1; nh=h-2*&pad;
    lid2=1; ylink2=ny+nh/2; xlink2=nx; output;
    ylink2=&rowSpacing-&pad; output;
end;
run;

/*--combine node and link data--*/
data &outData.;
  set nodes links;
run;

%mend consortData;
```

### Macro to render the diagram:

```
%macro consortDiagram(data= , fillColor=stgb);

proc sgplot data=consort noborder noautolegend;
  /*--Filled boxes--*/
  polygon id=pid x=xf y=yf / fill outline
    fillattrs=(color=&fillcolor) lineattrs=graphdatadefault;

  /*--vertical text--*/
  text x=xl y=yl text=vlabel / rotate=90 textattrs=(size=9 color=white);

  /*--Empty boxes--*/
  polygon id=pid x=x y=y / lineattrs=graphdatadefault;

  /*--horizontal text, left aligned--*/
  text x=xl y=yl text=hLabel / splitchar='.' splitpolicy=splitalways
    position=position;

  /*--Links--*/
  series x=xlink y=ylink / group=lid lineattrs=graphdatadefault
    arrowheadpos=end arrowheadshape=barbed arrowheadscale=0.4;

  /*--Links without arrow heads--*/
  series x=xlink2 y=ylink2 / group=lid2 lineattrs=graphdatadefault;

  xaxis display=none min=0 max=1 offsetmin=0 offsetmax=0;
  yaxis display=none min=0 max=1 offsetmin=0 offsetmax=0 reverse;
run;

%mend consortDiagram;
```

### Data Input and driver:

```
/*--Macro variables for layout--*/
%let rowSpacing=0.25;
%let pad=0.02;
%let headerWidth=0.1;
%let nColumns=4;
%let yAssess=0.2;
%let yRandom=0.7;
%let yExclude=0.4;
%let yHeader=0.5;
```

CONSORT Diagram using SGPLOT Procedure, continued

```

proc format;
  value row
    5='Analysis'
    4='Follow-up'
    3='Allocation'
    2='Dummy'
    1='Enrollment';

  value col
    1='Header'
    2='Placebo'
    3='Arm-1'
    4='Arm-2'
    5='Arm-3';
run;

/*--Diagram Layout--*/
data layout;
  length Stage $12 case $12;
  format Row row. Col col.;
  input Row Col Stage $ Case $ n1-n5;

  /*--Pad to increase the width of the Exclude box when columns=3--*/
  wf=(4-&nColumns)*0.05;
  call symput("wf", wf);

  /*--Node widths and heights--*/
  h=&rowSpacing;
  if Stage='Dummy' then h=2*&pad;
  dx=(1-&headerWidth)/&nColumns;

  select (case);
    when ('Header')
      w=&headerWidth;
    when ('Randomized', 'Assessed')
      do; w=(&nColumns-1)*dx/2; h=0.1; end;
    when ('Excluded')
      do; w=wf+(&nColumns-1)*dx/2; h=0.2; end;
    otherwise
      w=dx;
  end;

  /*--Update your n1-n5 counts here--*/
datalines;
1 1 Enrollment Header . . . . .
1 2 Enrollment Assessed 445 . . . .
1 3 Enrollment Excluded 39 22 14 3 .
1 4 Enrollment Randomized 406 . . . .
2 1 Dummy Header . . . . .
2 2 Dummy Placebo . . . . .
2 3 Dummy ARM-1 . . . . .
2 4 Dummy ARM-2 . . . . .
3 1 Allocation Header . . . . .
3 2 Allocation Placebo 95 90 5 . .
3 3 Allocation ARM-1 103 103 0 . .
3 4 Allocation ARM-2 105 98 7 . .
4 1 Follow-Up Header . . . . .
4 2 Follow-Up Placebo 10 2 4 9 5
4 3 Follow-Up ARM-1 7 3 2 1 1
4 4 Follow-Up ARM-2 11 5 2 1 3
5 1 Analysis Header . . . . .

```

CONSORT Diagram using SGPLOT Procedure, continued

```
5 2 Analysis Placebo 89 7 90 6 .
5 3 Analysis ARM-1 100 2 103 0 .
5 4 Analysis ARM-2 98 7 98 7 .
;
run;

/*--Create the Diagram data, given the counts data set --*/
%consortData (inData=layout, outData=consort, arms=3);

%let gpath=C:\Work\Sugi\PharmaSUG_China\2018\Papers\Consort\Image;
%let dpi=200;

ods _all_ close;
ods listing gpath=&gpath image_dpi=&dpi;
ods graphics / reset width=6in height=4in imagename='Consort_3_Arms_4_Col';

/*--Draw the Consort diagram--*/
title 'CONSORT Diagram for Displaying Counts';
%consortDiagram(data=consort, fillColor=darkmagenta);
```