Paper 3794-2019

Don’t leave it to the discretion of ODS Graphics: Use Discrete Attribute Map

Shilpakala Vasudevan, Ephicacy Lifescience Analytics

ABSTRACT

When working with visualization of grouped data, SAS® ODS defines default attributes (colors, symbols, patterns) for each group in the graph. For example, if we were to plot line plots with markers for different groups in a graph: ODS styles would assign by default a line color, a symbol for the marker, and a marker color for each of the groups by itself. Although this may seem convenient, however, if there is an update in the data, the values for the groups might change or there might be missing groups in the data. This might cause the attributes for the groups to differ each time the graph is run. To aid in uniformity, we can use discrete attribute maps that assign set attributes to each group value. This consistency in the patterns and attributes then makes it easier to understand and review the graphs.

In this paper, we will be seeing how to implement discrete attribute map sets with examples, and how it ensures an easy way to consistency in display of outputs.

INTRODUCTION

ODS Graph procedures help produce a variety of graphs useful in statistical analysis. An important part of graphs are the visual attributes: colors, patterns, lines, sizes and fonts. Attributes easily help us to identify the corresponding values in the data. Group analysis is a very common feature in statistics. When working with groups, it is important that consistency is maintained when they are represented in graphs. This enables the reviewers to quickly recognize and distinguish any changes, ensures accuracy in the review and saves time as well, in avoiding errors.

WHY DISCRETE ATTRIBUTE MAPS?

Let us say we are working on a study that checks for the health index of patients, and there are three treatment groups in this study: A, B, C. We will be comparing the three treatment groups to see which one can create a positive impact on the health indices of patients. For this purpose, we will be computing the % change in health index values.

The following is part of a sample dataset of the study:

Table 1. Sample data for spider plot

Obs	Treatment	Week	% change in Health Index
1	A	Week 1	5
2	A	Week 2	7
3	B	Week 1	10
4	B	Week 2	13
5	C	Week 1	-7
6	C	Week 2	-10

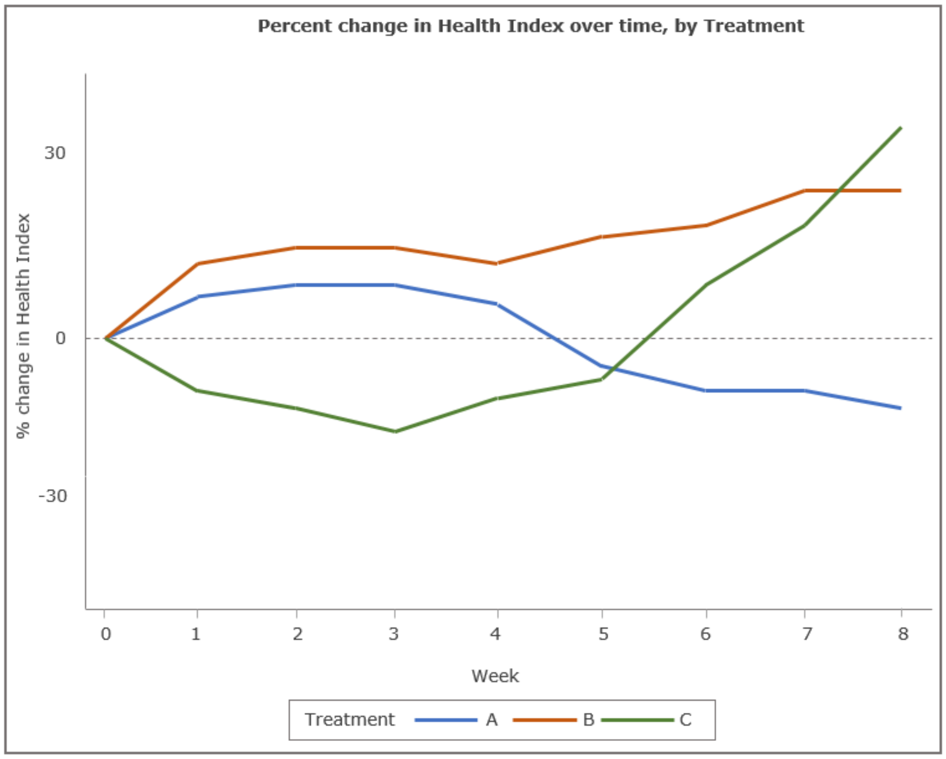
To visually compare, we will produce a spider plot, that will plot the %change in health index with time. A spider plot typically has a reference line, from which all group values have their starting value.

To create a spider plot, we have the below proc sgplot code:

```
title "Percent change in Health Index over time, by Treatment";
ods graphics / reset width=5in height=4in;
PROC SGPLOT data=spider;
  refline 0 / lineattrs=(pattern=dash);
  series x=week y=change / group=treatment;
run;
```

With this code, the below plot is generated:

Figure 1. Spider plot



We can see the Time (Week) plotted in X axis, and % change in Health Index plotted in Y axis. The reference line is 0, and at the beginning all 3 treatment groups start from 0. Here the treatment groups are color coded as:

Treatment A – Blue
Treatment B – Red
Treatment C – Green

This color coding is done based on the default working mechanisms of SAS. By default, SAS picks up attributes from the GraphData1-GraphDataN style elements. In the sample dataset, the first occurring treatment is A, so the linecolor is allocated the first style element color as blue. The second occurring treatment among the observations in the dataset, is B, and this is assigned the second style element color, red. And in a similar manner, the third occurring treatment is C which is assigned green color.

WITHOUT DISCRETE ATTRIBUTE MAP

Let us suppose there was a modification in the data or an update in the data. In our example, let us say, we modify the data, by sorting the data by %change in health index:

```
PROC SORT data=spider;
  by change;
run;
```

This updates the data to this order:

Table 2. Sorted sample data

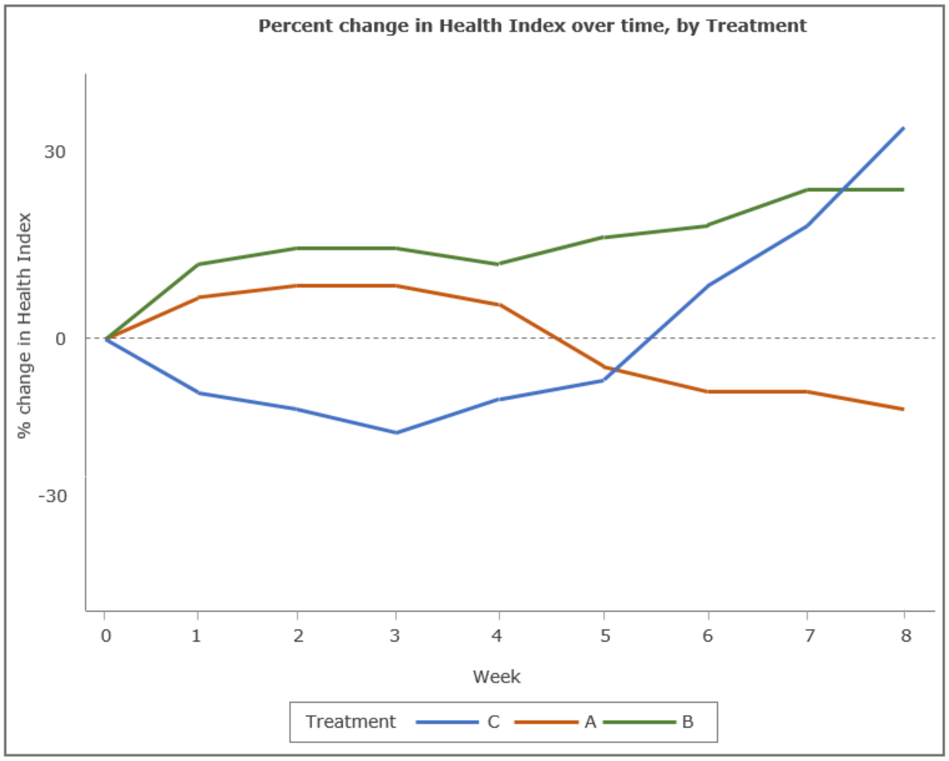
Obs	Treatment	Week	% change in Health Index
1	C	Week 2	-10
2	C	Week 1	-7
3	A	Week 1	5
4	A	Week 2	7
5	B	Week 1	10
6	B	Week 2	13

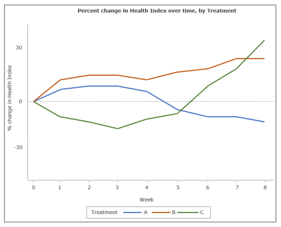
As we can see, the order of the data has changed on the sort. Treatment C occurs first in the dataset, followed by A and then B.

Now with this updated dataset, if we create the same plot, we will see the order of the colors assigned to treatments will change. As seen in the below graph, the order of the color for treatment groups has changed to:

Treatment C – Blue (Green in graph before)
Treatment A – Red (Blue in graph before)
Treatment B – Green (Red in graph before)

Figure 2. Spider plot after an update in data





Spider Plot before update

In certain cases, we might consider only 2 treatments to be displayed in a plot, instead of displaying all 3.

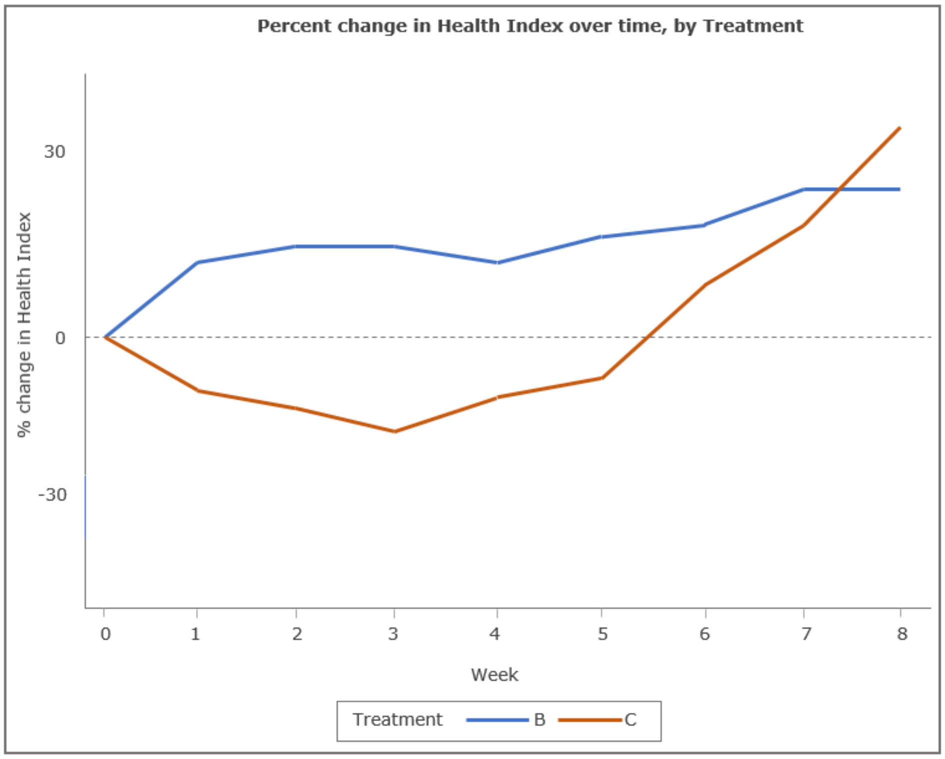
For example, if we are only selecting Treatments B and C for comparison:

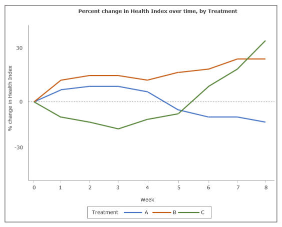
Table 3. Missing group in sample data

Obs	Treatment	Week	% change in Health Index
1	B	Week 1	10
2	B	Week 2	13
3	C	Week 1	-7
4	C	Week 2	-10

If we now plot the graph with the same code, the colors allotted to the groups get changed again. The absence of treatment A makes the first style element attributes to be transferred to the second occurring treatment group B, and the second style element ones to the third treatment group C.

Figure 3. Spider plot with a missing group





Spider Plot before update

The colors are therefore modified every time the data is updated, or if there is any missing data for the group. Every time plot is created, the attributes might change depending on the modification that happens to the data.

This might cause difficulties whenever data needs to be compared from time to time, or different studies are compared with each other or when the reviewer wants to check for updates.

WITH DISCRETE ATTRIBUTE MAP

With discrete attribute maps, we can specify the attributes that each group should have. This ensures the graphs have the same attributes every time the plot is run.

HOW TO USE DISCRETE ATTRIBUTE MAPS?

There are 2 steps to create and use a discrete attribute map:

1. Creating a discrete attribute map data set
2. Associating the data set to the graph procedures

1. CREATING A DISCRETE ATTRIBUTE MAP DATA SET

First we create a dataset that would map attributes to every data value of the group variable.

The data set should have:

- ID variable – Attribute map identifier variable that can be used to identify the attribute map we will be using.
- Value variable – Indicates the group values for which attributes will be assigned
- Attribute variables – visual attributes that we will assign for the group values. The visual attributes can be related to line styles (linecolor, linepattern, linestyle, linethickness), fill styles (fillcolor, fillstyle, filltransparency), marker styles (markercolor, markersize, markerstyle, markersymbol, markertransparency) text styles (textcolor, textfamily, textstyle, textstyleelement, textweight)

Example of a map data set is as follows:

```
DATA attrmap1;
  length linecolor fillcolor $ 15;
  input ID $ value $ linecolor $ fillcolor $;
  datalines;
    id1  A  yellow      yellow
    id1  B  pink        pink
    id1  C  lightgreen  lightgreen
  ;
run;
```

Here “attrmap1” is the attribute map data set name. “id1” is the ID variable that will be used to identify the attributes for the treatment group. For every value in the group, an attribute (linecolor, fillcolor) is assigned.

Table 4. Discrete Attribute Map data set

Obs	ID	value	linecolor	fillcolor
1	id1	A	yellow	yellow
2	id1	B	pink	pink
3	id1	C	lightgreen	lightgreen

2. ASSOCIATING THE DATA SET TO THE GRAPH PROCEDURES

Once we have created the data set, we need to associate this to the graph procedures we use.

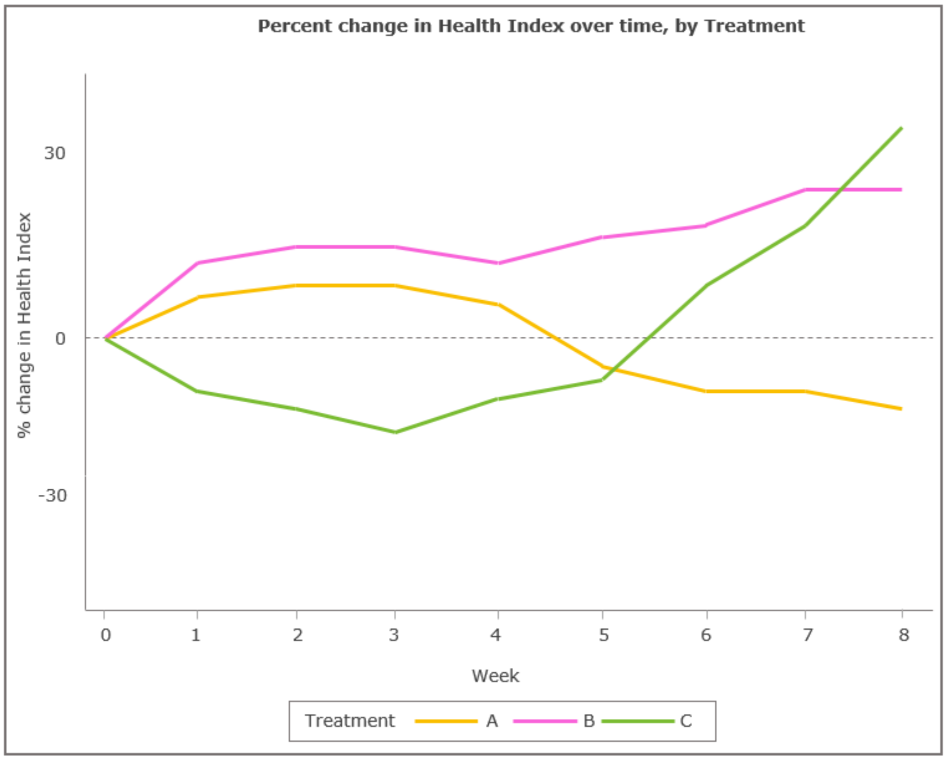
The graph procedure should have:

- DATTRMAP= option – To reference the attribute map data set in the procedure
- GROUP= option – To indicate which group variable is used in the procedure
- ATTRID= option – To indicate which ID variable from the attribute data set, is used for associating attributes to the group values

In the below example, we are using proc sgplot. The dattrmap indicates the attribute data set that was created, the attrid indicates the id variable values associated with the group.

```
title "Percent change in Health Index over time, by Treatment";
ods graphics / reset width=5in height=3in;
PROC SGPLOT data=spider dattrmap=attrmap1;
  refline 0 / lineattrs=(pattern=dash);
  series x=week y=change / group=treatment attrid=id1;
run;
```

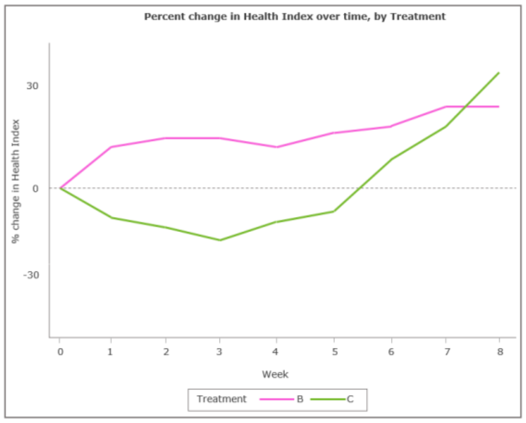
Figure 4. Spider plot using discrete attribute map



As we defined in the attribute map, treatment A is now shown as yellow, Treatment B is shown as pink, and Treatment C is light green.

This will ensure consistency of the attributes of the group even if data is changed. For example, if one of the treatment groups was missing, the same color coding would be followed, even if the missing group was not present.

Figure 5. Spider plot using discrete attribute map, with missing group



APPLYING DISCRETE ATTRIBUTE MAPS IN OTHER WAYS

a) Using Discrete Attribute Map for Multiple Groups

Let us assume in addition to the treatment groups, we also want to plot any events of interest in our plot.

Table 5. Multiple groups data set

Obs	Treatment	Week	% change in Health Index	Event of interest
1	A	Week 1	5	noevent
2	A	Week 2	7	noevent
3	B	Week 1	10	event
4	B	Week 2	13	event
5	C	Week 1	-7	event
6	C	Week 2	-10	noevent

We will need to create a separate id in our dataset for the 'event of interest' group. Now we have id1 and id2 to create attributes for 2 groups.

```
DATA attrmap2;
  length linecolor fillcolor markercolor markersymbol $ 15;
  input ID $ value $ linecolor $ fillcolor $
                   markercolor $ markersymbol $ markertransparency;
  datalines;
    id1  A        yellow      yellow
    id1  B        pink        pink
    id1  C        lightgreen  lightgreen
    id2  event    brown       brown       brown   TriangleFilled   0
    id2  noevent  brown       brown       brown   TriangleFilled   1
  ;
run;
```

Whenever, we have multiple groups in our dataset, we need to make sure the data set is sorted by the id, before it is taken to the procedure.

```
PROC SORT data=attrmap2;
  by id;
run;
```

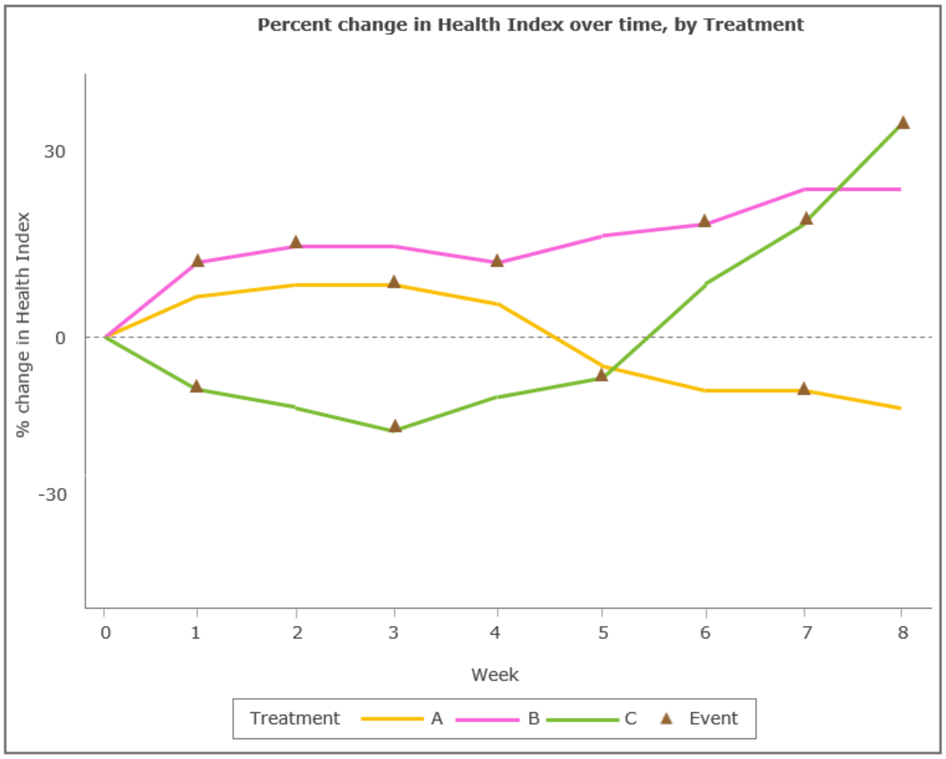
Table 6. Discrete Attribute Map data set for multiple groups

Obs	ID	value	linecolor	fillcolor	markercolor	markersymbol	markertransparency
1	id1	A	yellow	yellow			
2	id1	B	pink	pink			
3	id1	C	lightgreen	lightgreen			
4	id2	event	brown	brown	brown	brown	0
5	id2	noevent	brown	brown	brown	brown	1

In out sgplot procedure, we are having a series plot where we use 'id1' for mapping attributes to treatment. We also have scatter plot, where we will use 'id2' to map attributes to event of interest.

```
title "Percent change in Health Index over time, by Treatment";
ods graphics / reset width=5in height=4in;
PROC SGPLOT data=spider dattrmap=attrmap2;
  refline 0 / lineattrs=(pattern=dash);
  series x=week y=change / group=treatment name="legend1" attrid=id1;
  scatter x=week y=change / group=event name="legend2" attrid=id2;
  keylegend "legend1" "legend2" / title="Treatment" exclude=("noevent");
run;
```

Figure 6. Plot with using multiple groups, using discrete attributes

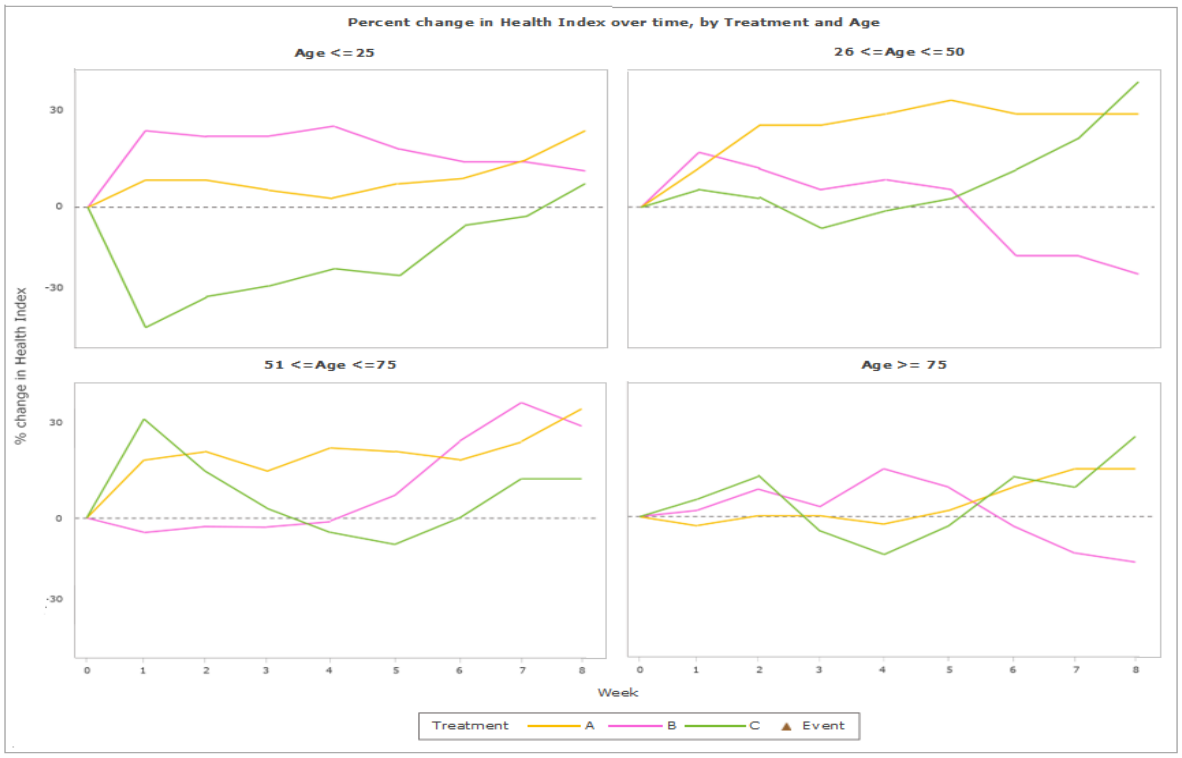


b) Using Discrete Attribute Map for Creating Multipanel Graphs

We can apply the same method to proc sgpanel, to create multipanel graphs:

```
title "Percent change in Health Index over time, by Treatment";
ods graphics / reset width=5in height=3in;
PROC SGPANEL data=spider dattrmap=attrmap1;
  panelby age;
  refline 0 / lineattrs=(pattern=dash);
  reg x=week y=change / group=treatment attrid=id1;
run;
```

Figure 7. Multipanel plots using discrete attribute map



c) Using Discrete Attribute Map with PROC TEMPLATE

Alternatively, we can use discrete attribute maps, with proc template and sgrender. Proc template is more robust and can prove to be more flexible than the ODS procedures, especially when there are missing data involved.

In proc template, we have the discreteattrmap block in the template, where we assign the attributes for each group. The group variable and the id for the map is defined in the discreteattrvar statement. The plotting of the values and the groups with the attributes, defined in the template, is then referenced in proc sgrender.

```
proc template;
define statgraph attrmap1;
begingraph;
  entrytitle " Percent change in Health Index over time, by Treatment ";
```

```
    discreteattrmap name="id1" / ignorecase=true;
      value "A" / lineattrs=(color=yellow);
      value "B" / lineattrs=(color=pink);
      value "C" / lineattrs=(color=lightgreen);
    enddiscreteattrmap;

    discreteattrvar attrvar=attreatment var=treatment attrmap="id1";
    layout overlay;
      seriesplot x=week y=change / group=attreatment;
      discretelegend "id1" / title="Treatment";
    endlayout;

  endgraph;
  end;
  run;

  proc sgrender data=spider template=attrmap1;
  run;
```

CONCLUSION

Discrete attribute maps are the next step in simplification of developing efficient graphs. Normally when working with groups, SAS takes the default style elements for assigning attributes. With discrete attribute maps, there is an easy way to customize the graphs as per our requirements, and to ensure uniformity.

Another way of providing further enhancements to our graphs are the range attribute maps, that help in mapping attributes to ranges of values. The attribute maps have different functionalities when made well use of will provide more clarity and precision to the graphs.

CONTACT INFORMATION

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

Shilpakala Vasudevan
Ephicacy Lifescience Analytics
shilpakala.vasudevan@ephicacy.in