PharmaSUG China 2024 - Paper DV-10021

Heat Maps of Clinical Data Using SAS® and R

Ternence Liu, Sanofi Corporation.

ABSTRACT

A heat map is a graphical representation of data that displays the relative intensity of values in a matrix as colors. It is commonly used to visualize complex datasets and identify patterns, correlations, and outliers. Heat maps are especially useful when dealing with large datasets or data with high dimensionality, as they can provide an overview of the data briefly. Heat maps are increasingly used in clinical trials to visually represent data and provide insights into various aspects of the trial, such as treatment response, patient characteristics, safety profiles, geographic distribution, etc. This paper demonstrates how to visualize clinical data with heat maps using SAS® (PROC SGPLOT, Graph Template Language (GTL)) and R (ggplot2) through a practical example.

INTRODUCTION

Graphs can be effective in visualizing the data, which might be difficult to see in tabular form. If you want to display as much information as possible in as little space, ways like heat maps are highly recommended. Heat maps consist of a grid of cells, where each cell represents a data point in the matrix. The color of each cell represents the value of the data point, with a gradient of colors used to indicate increasing or decreasing values. Usually, a darker color or shade represents a higher or greater quantity of the value represented in the heat map. This paper introduces Figure 1 as an example and shows how to generate this heat map step by step using SAS® and R.

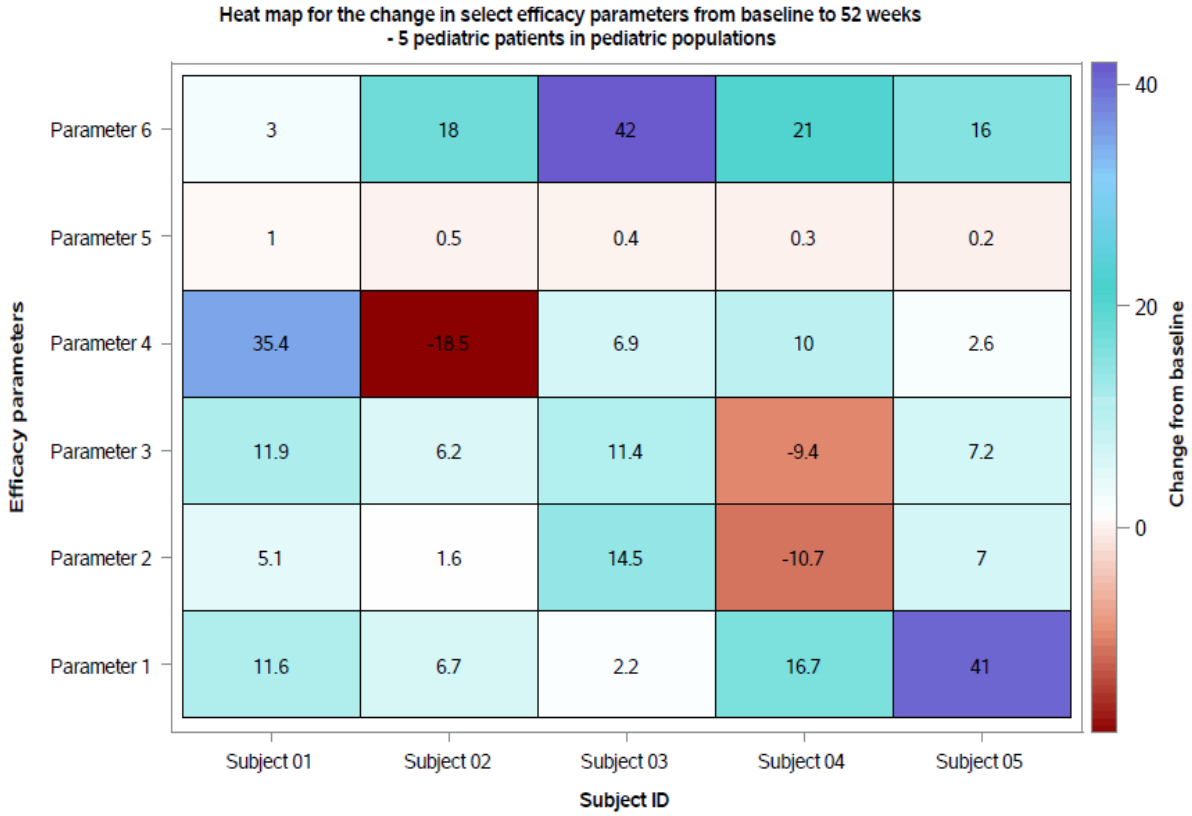

Figure 1. Sample Figure

Figure 1 shows the change from baseline at Week 52 for six efficacy parameters of each pediatric subject. Through the heat map, the data can be visualized in a better way.

In Figure 1:

- **The X-axis** represents five pediatric subjects (SUBJID).
- **The Y-axis** represents six different efficacy parameters (PARAM).
- **The values in the matrix** represent the change from baseline at Week 52 of the parameters (CHG).

A dummy dataset was used for Figure 1 and other figures in this paper. The SAS code to generate the dummy dataset is shown below:

```
data dummy_data;
     length SUBJID $25. PARAM AVISIT $20.;
     input SUBJID &:$ PARAM &:$ AVISIT &:$ AVAL  BASE CHG;
     cards;
   Subject 01   Parameter 1   WEEK 52   38.6   27   11.6
   Subject 01   Parameter 2   WEEK 52   51.1   46   5.1
   Subject 01   Parameter 3   WEEK 52   57.8   45.9   11.9
   Subject 01   Parameter 4   WEEK 52   103.3   67.9   35.4
   Subject 01   Parameter 5   WEEK 52   -1.3   -2.3   1
   Subject 01   Parameter 6   WEEK 52   59   56   3
   Subject 02   Parameter 1   WEEK 52   68.4   61.7   6.7
   Subject 02   Parameter 2   WEEK 52   76.2   74.6   1.6
   Subject 02   Parameter 3   WEEK 52   86.4   80.2   6.2
   Subject 02   Parameter 4   WEEK 52   93.8   112.3   -18.5
   Subject 02   Parameter 5   WEEK 52   -1   -1.5   0.5
   Subject 02   Parameter 6   WEEK 52   120   102   18
   Subject 03   Parameter 1   WEEK 52   67.2   65   2.2
   Subject 03   Parameter 2   WEEK 52   81.5   67   14.5
   Subject 03   Parameter 3   WEEK 52   84.8   73.4   11.4
   Subject 03   Parameter 4   WEEK 52   103.7   96.8   6.9
   Subject 03   Parameter 5   WEEK 52   -1.3   -1.7   0.4
   Subject 03   Parameter 6   WEEK 52   134   92   42
   Subject 04   Parameter 1   WEEK 52   76.9   60.2   16.7
   Subject 04   Parameter 2   WEEK 52   82.2   92.9   -10.7
   Subject 04   Parameter 3   WEEK 52   82.5   91.9   -9.4
   Subject 04   Parameter 4   WEEK 52   100   90   10
   Subject 04   Parameter 5   WEEK 52   -1.7   -2   0.3
   Subject 04   Parameter 6   WEEK 52   80   59   21
   Subject 05   Parameter 1   WEEK 52   85.6   44.6   41
   Subject 05   Parameter 2   WEEK 52   91   84   7
   Subject 05   Parameter 3   WEEK 52   87.7   80.5   7.2
   Subject 05   Parameter 4   WEEK 52   102.6   100   2.6
   Subject 05   Parameter 5   WEEK 52   -2.1   -2.3   0.2
   Subject 05   Parameter 6   WEEK 52   50   34   16
   ;
run;
```

GENERATE HEAT MAPS USING SAS®

As of SAS® 9.4, statements like HEATMAPPARM and HEATMAP in SGPLOT and SGPANEL procedures, Graph Template Language (GTL), etc. can be used to create heat maps.

This section introduces how to generate the sample heat map (Figure 1) using SAS® step by step. Some common options and statements will be introduced. Two methods will be introduced in this paper:

- Method 1: HEATMAPPARM in PROC SGPLOT
- Method 2: Graph Template Language (GTL)

PROC SGPLOT

HEATMAPPARM statement in SGPLOT creates a heat map, which represents the values of a response variable with different colors. This statement requires X, Y, and either one of COLORRESPONSE or COLORGROUP arguments. The **COLORRESPONSE** argument only supports numeric variables and colors the cells continuously using a gradient of colors. The COLORGROUP argument is required when the response variable is of type discrete.

Step 1: Figure 2 - Create a basic heat map by using the following code:

```
proc sgplot data=dummy_data;
     heatmapparm x=SUBJID y=PARAM colorresponse=CHG;
run;
```

Note that:

- If X= specifies a numeric variable and the X-axis type is discrete, then you must specify **DISCRETEX**. Otherwise, the heat map might not be drawn correctly. If X= specifies a character variable, then this option is ignored, and the X-axis is considered to be discrete.
- If Y= specifies a numeric variable and the Y-axis type is discrete, then you must specify **DISCRETEY**. Otherwise, the heat map might not be drawn correctly. f Y= specifies a character variable, then this option is ignored, and the Y axis is considered to be discrete.

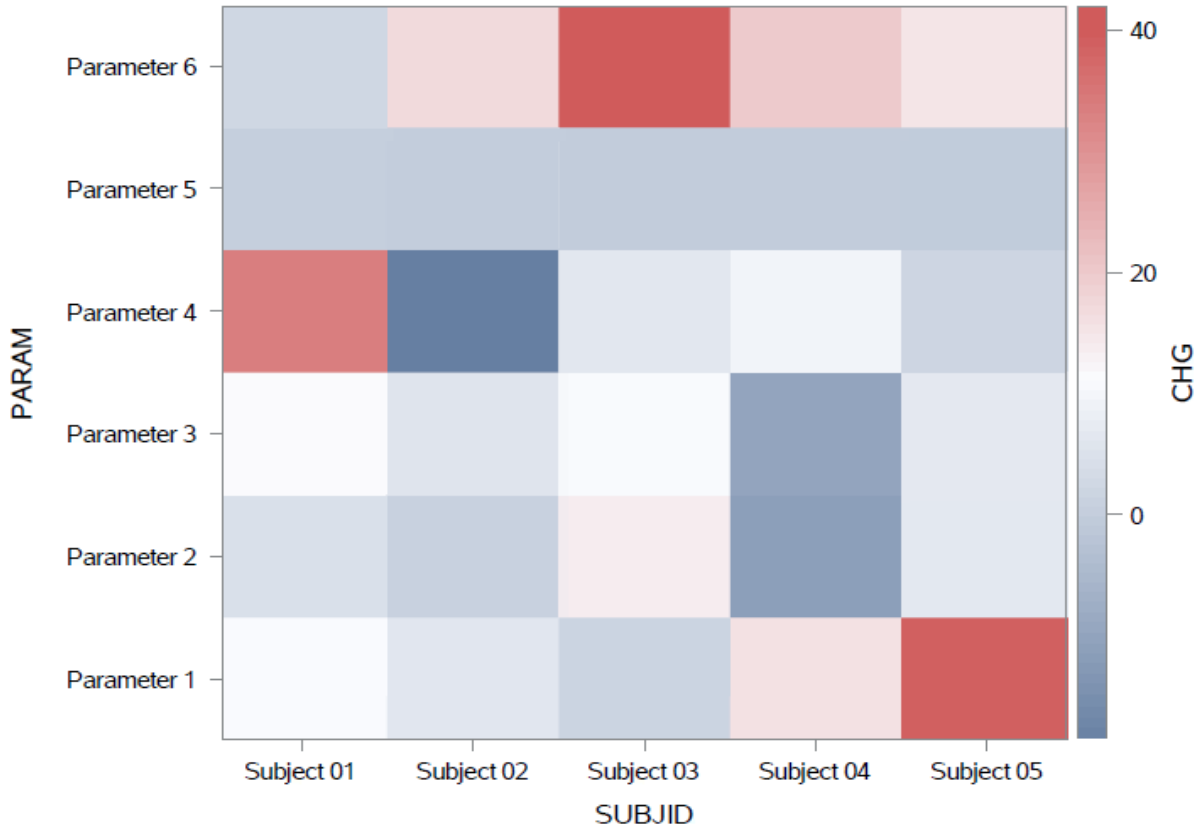

Figure 2. SGPLOT – Heat Map with default color assignments and auto legend

Step 2: Figure 3 - Displays an outline around each colored region using the **OUTLINE** option**.** Use the **TEXT** statement to display the associated text values at (X, Y) locations in the graph. The text can be numbers or characters. The **STRIP** option specifies that leading and trailing blanks should be stripped from the marker text before it is displayed.

The code for Figure 3 is shown below:

```
proc sgplot data=dummy_data;
    heatmapparm x=SUBJID y=PARAM colorresponse=CHG / outline;
    text x=SUBJID y=PARAM text=CHG / strip;
run;
```

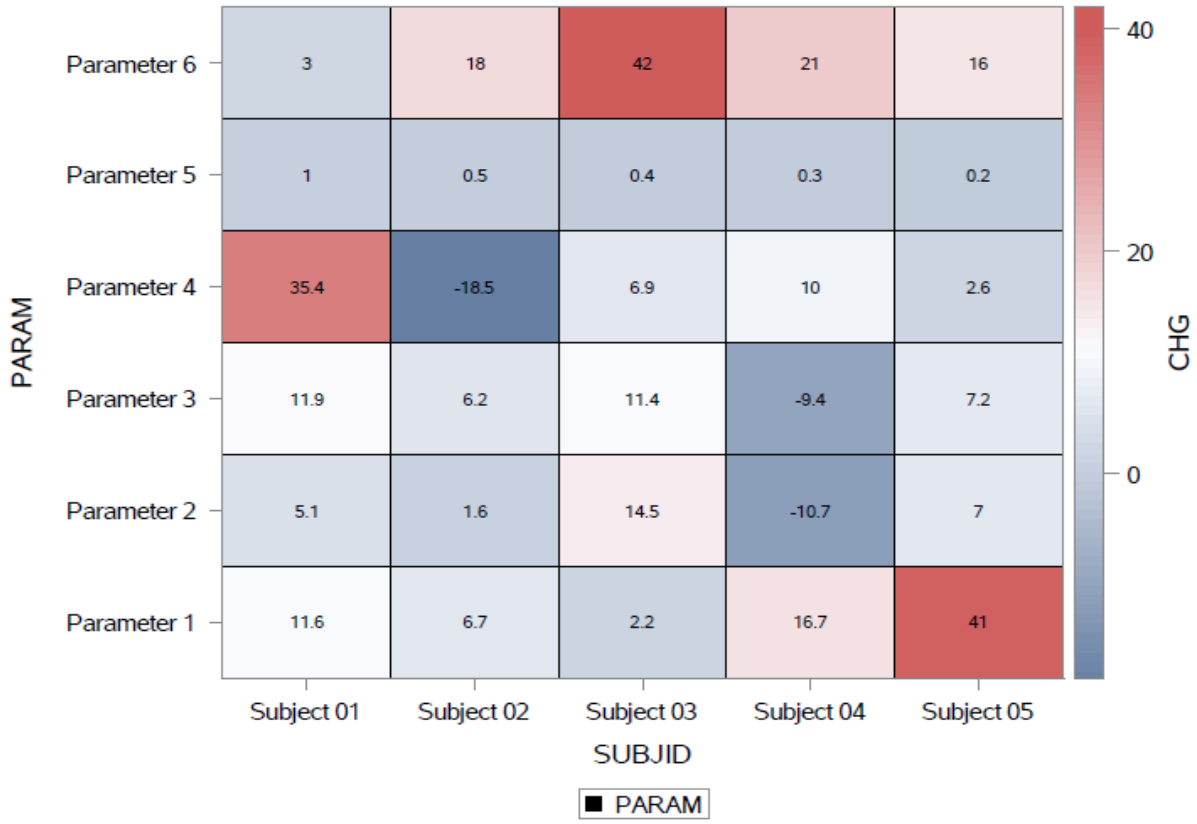

Figure 3. SGPLOT – Heat Map using OUTLINE option and TEXT statement

Step 3: Figure 4 - Setting the label for the X-axis, Y-axis, and legend. Use the COLORMODEL= option to specify a color ramp to use. Instead of auto legend, we use **GRADLEGEND** here, which is the gradient legend that can be used only in conjunction with the COLORRESPONSE option. Note that if the COLORGROUP is used rather than the COLORRESPONSE, then the KEYLEGEND statement can be used to display the range values as the GRADLEGEND is not supported by the COLORGROUP option.

The code for Figure 4 is shown below:

```
proc sgplot data=dummy_data;
  heatmapparm x=SUBJID y=PARAM colorresponse=CHG / outline
colormodel=(DarkRed DarkSalmon White PaleTurquoise MediumTurquoise
LightSkyBlue SlateBlue);
  text x=SUBJID y=PARAM text=CHG / strip;
  xaxis labelattrs=(size=8 weight=bold) label="Subject ID" offsetmin=0.11
offsetmax=0.11;
  yaxis labelattrs=(size=8 weight=bold) label="Efficacy parameters"
offsetmin=0.1 offsetmax=0.1;
  gradlegend / titleattrs=(size=8 weight=bold) title="Change from baseline";
run;
```

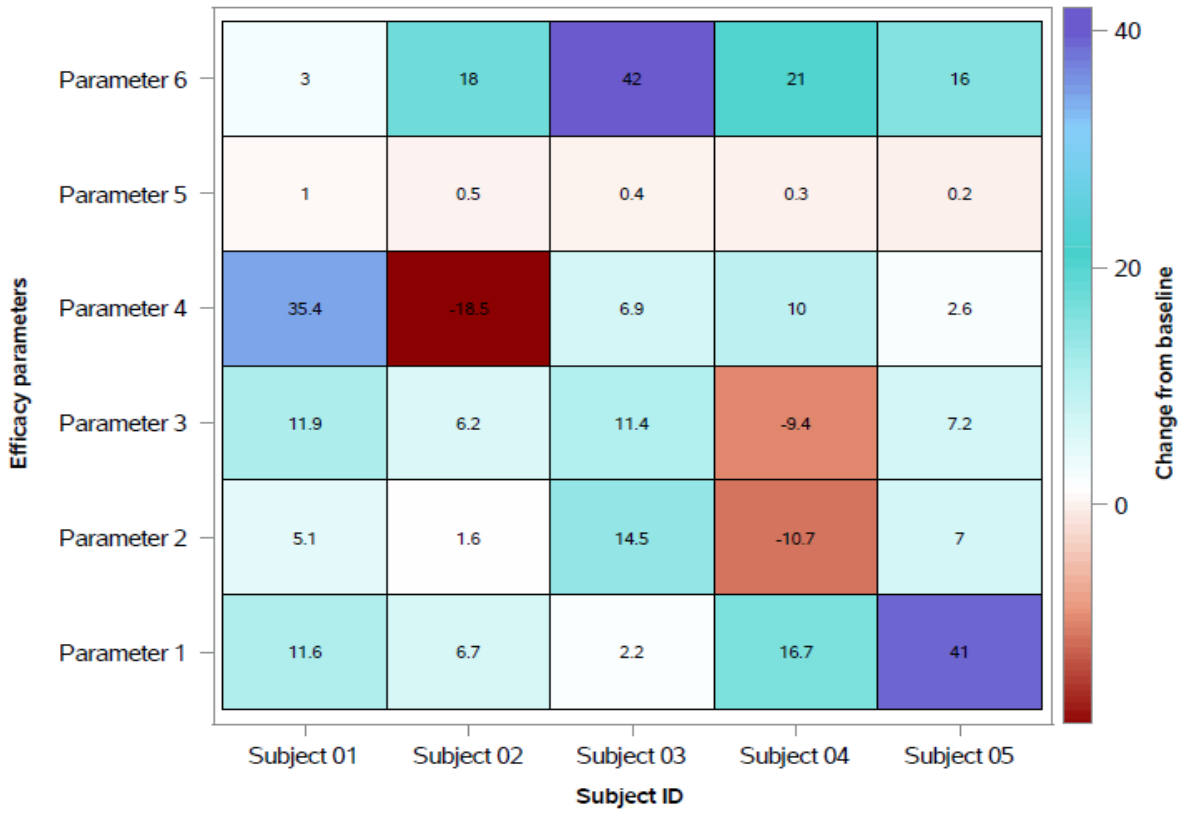

Figure 4. SGPLOT – Heat Map using COLORMODEL option and GRADLEGEND statement

Summary: By using the HEATMAPPARM statement in SGPLOT, you can create a basic heat map, and you can use the options (OUTLINE, COLORMODEL, etc.) and statements (TEXT, GRADLEGEND, etc.) to enhance the figures to better visualize the data. For more details about the PROC SGPLOT HEATMAPPARM statement, you can refer to the SAS help document: SAS Help Center: Syntax: PROC SGPLOT HEATMAPPARM Statement.

GRAPH TEMPLATE LANGUAGE (GTL)

By using GTL, the statements **HEATMAP** and **HEATMAPPARM** can be used to create heat maps. In this paper, we will create the heat map by using the **HEATMAPPARM** statement as an example.

Step 1: Figure 5 - Create a basic heat map by using the following code:

```
proc template;
  define statgraph heatmap;
   begingraph;
     layout overlay;
       heatmapparm x=SUBJID y=PARAM colorresponse=CHG /
       name="heatmap";
       continuouslegend "heatmap";
     endlayout;
   endgraph;
  end;
```

```
run;

proc sgrender data=dummy_data template=heatmap;
run;
```

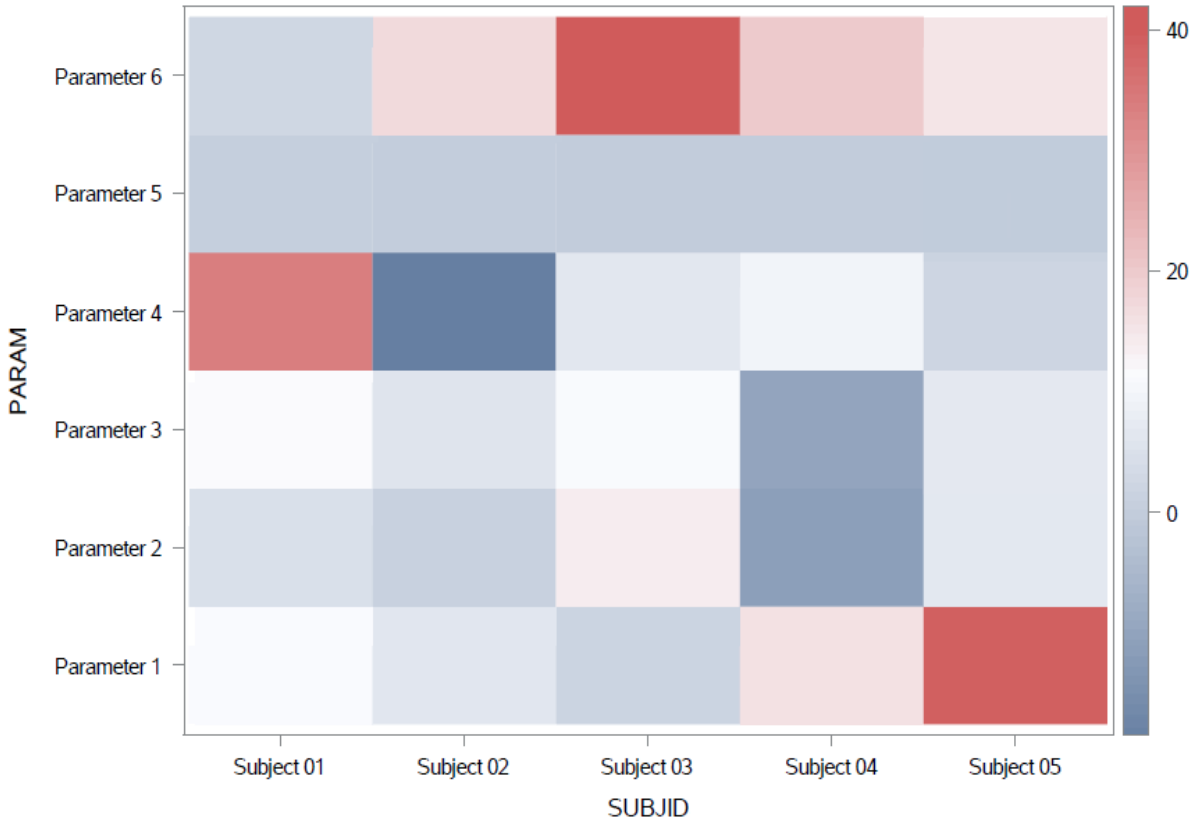

Figure 5. GTL – Heat Map with default color assignments and auto legend

Step 2: Figure 6 - Displays an outline around each colored region using the **DISPLAY=ALL** option. Use the **OUTLINEATTRS=** option to specify the outline attributes. Use the **TEXTPLOT** statement to display text values at specific X and Y locations in the graph. The **STRIP=TRUE** option specifies leading and trailing blanks that should be stripped from the text-marker text before it is displayed.

The code for Figure 6 is shown below:

```
proc template;
  define statgraph heatmap;
    begingraph;
      layout overlay;
        heatmapparm x=SUBJID y=PARAM colorresponse=CHG /
          name="heatmap" display=all outlineattrs=(pattern=solid thickness=1
          color=black);
        textplot x=SUBJID y=PARAM text=CHG / strip=true;
        continuouslegend "heatmap";
      endlayout;
    endgraph;
  end;
run;

proc sgrender data=dummy_data template=heatmap;
run;
```

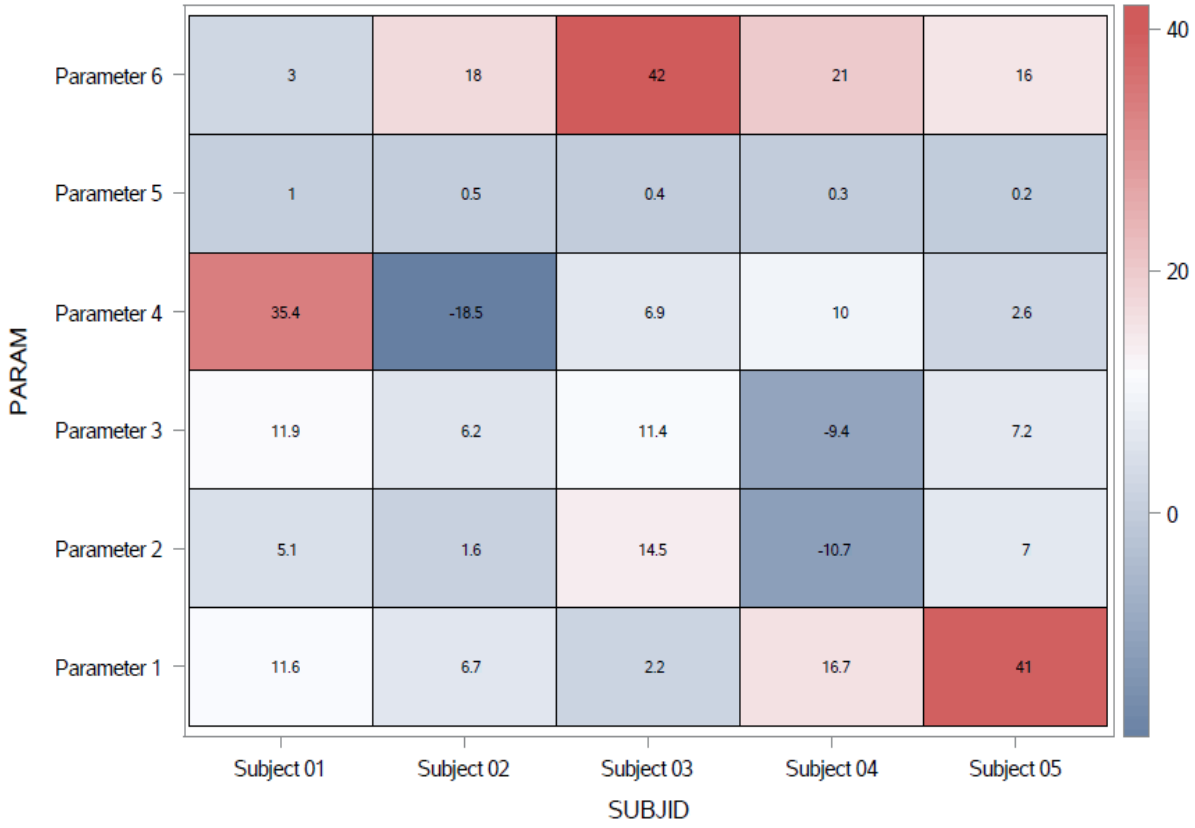

Figure 6. GTL – Heat Map using DISPLAY=ALL option and TEXTPLOT statement

Step 3: Figure 7 - Setting the label for the X-axis, Y-axis, and legend. Use the **COLORMODEL**= option to specify a color ramp to use.

The code for Figure 7 is shown below:

```
proc template;
  define statgraph heatmap;
    begingraph;
      layout overlay / xaxisopts=(label="Subject ID" labelattrs=(size=8
weight=bold)) yaxisopts=(label="Efficacy parameters" labelattrs=(size=8
weight=bold));
        heatmapparm x=SUBJID y=PARAM colorresponse=CHG /
          name="heatmap" display=all outlineattrs=(pattern=solid
          thickness=1 color=black) colormodel= (DarkRed DarkSalmon White
          PaleTurquoise MediumTurquoise LightSkyBlue SlateBlue);
         textplot x=SUBJID y=PARAM text=CHG / strip=true;
         continuouslegend "heatmap" / title="Change from baseline";
       endlayout;
    endgraph;
  end;
run;

proc sgrender data=dummy_data template=heatmap;
run;
```

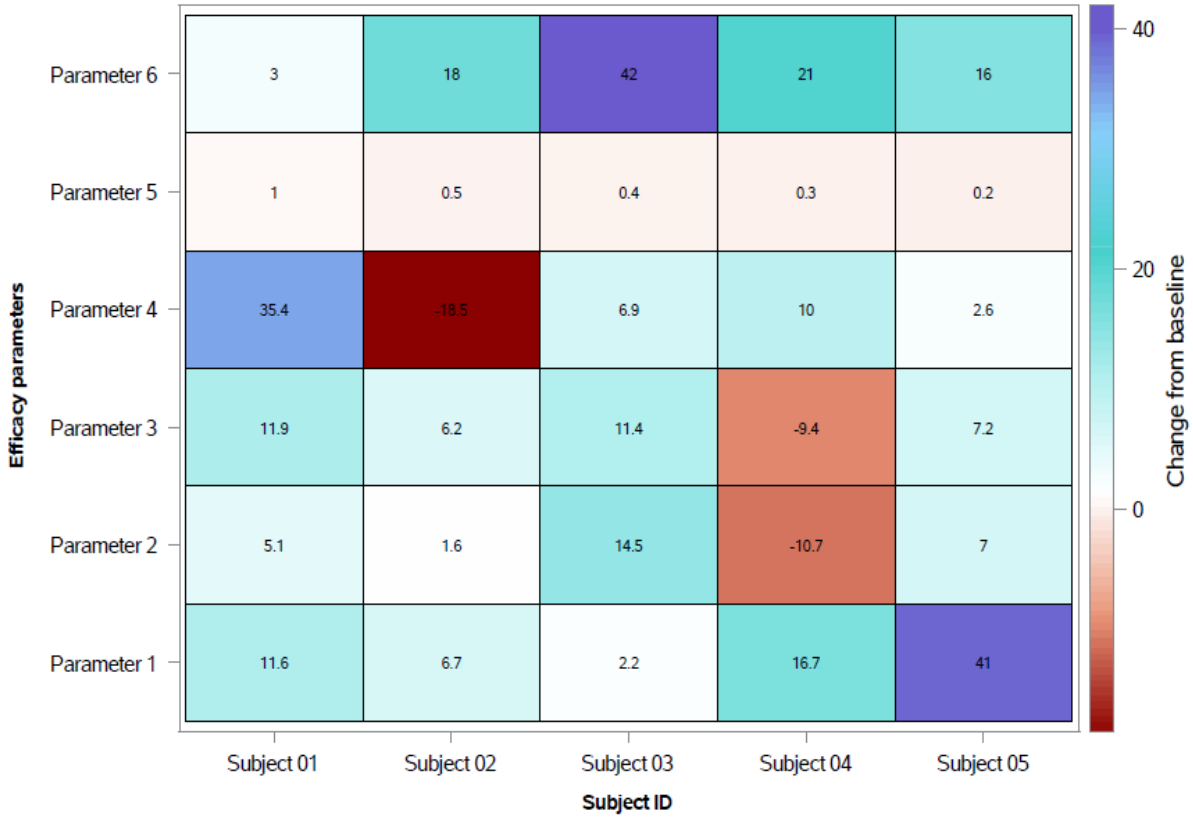

Figure 7. GTL – Heat Map using COLORMODEL option

Summary: By using the HEATMAPPARM statement in GTL, you can create a basic heat map, and you can use the options (DISPLAY=ALL, COLORMODEL, etc.) and statements (TEXTPLOT, CONTINUOUSLEGEND, etc.) to enhance the figures to better visualize the data. For more details about the GTL HEATMAPPARAM statement, you can refer to the SAS help document: SAS Help Center: HEATMAPPARM Statement.

GENERATE HEAT MAPS USING R

In R, there are several methods to create heat maps, depending on the specific needs and data characteristics. Such as the ***heatmap()*** function of the **stats** package, the ***pheatmap()*** function of the **pheatmap** package, the ***geom_tile()*** function of the **ggplot2** package, etc.

In this paper, the method using the ***geom_tile()*** function of the **ggplot2** package will be introduced. A heat map in ggplot2 can be created with geom_tile, passing the categorical variables to x and y arguments and the continuous variable to fill the argument of ***aes***. You can customize the border color, line width, and line style of the tiles with ***color***, ***lwd*** *and* ***linetype***, respectively.

Note: R version 4.2.1 is used for this paper.

The code for Figure 8 is shown below:

```
install.packages("ggplot2")
library (ggplot2)
ggplot(dummy_data, aes(x = SUBJID, y = PARAM, fill = CHG)) +
       geom_tile(color = "black",
       lwd = 0.15,
       linetype = 1)
```

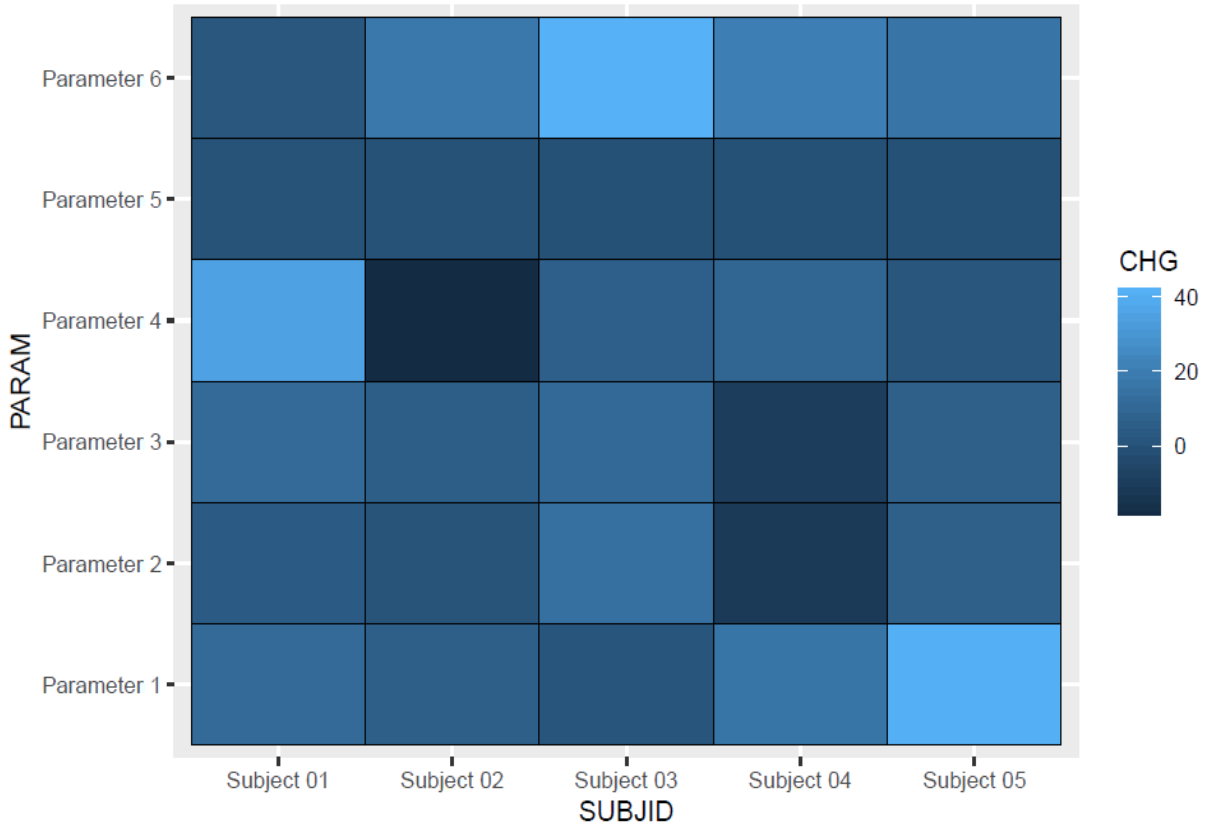

Figure 8. ggplot2 – Heat Map using geom_tile() function

Adding the values

In addition, you can add the values over the tiles with ***geom_text()*** function, passing the numerical variable to the label argument of the ***aes()*** function.

The code for Figure 9 is shown below:

```
# install.packages("ggplot2")
library(ggplot2)
ggplot(dummy_data, aes(x = SUBJID, y = PARAM, fill = CHG)) +
       geom_tile(color = "black",
                 lwd = 0.15,
                 linetype = 1) +
       geom_text(aes(label = CHG), color = "black", size = 4)
```

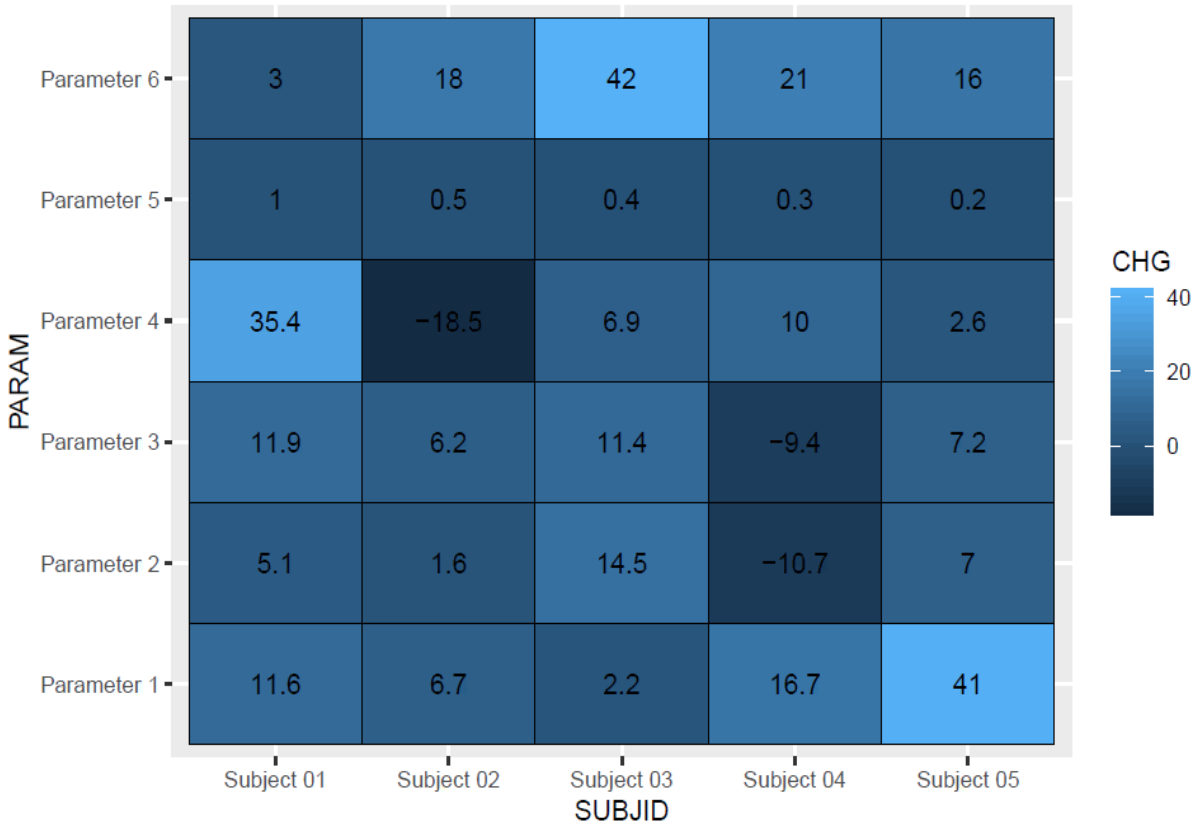

Figure 9. ggplot2 – Heat Map using geom_text() function

Change the label of x-axis, y-axis and legend

labs() function can modify axis, legend, and plot labels. And the ***guide_colourbar()*** function can set the attributes of the legend.

The code for Figure 10 is shown below:

```
# install.packages("ggplot2")
library(ggplot2)
```

```
ggplot(dummy_data, aes(x = SUBJID, y = PARAM, fill = CHG)) +
       geom_tile(color = "black",
                 lwd = 0.15,
                 linetype = 1) +
       geom_text(aes(label = CHG), color = "black", size = 4) +
       labs(x = "Subject ID",
            y = "Efficacy parameters",
            fill = "Change from baseline") +
       guides(fill=guide_colourbar (title.position = "right",
                                    barheight = 19,
                                     barwidth = 0.8,
                                    title.hjust = 0.5,
                                    title.theme = element_text(angle = 90)
              ))
```

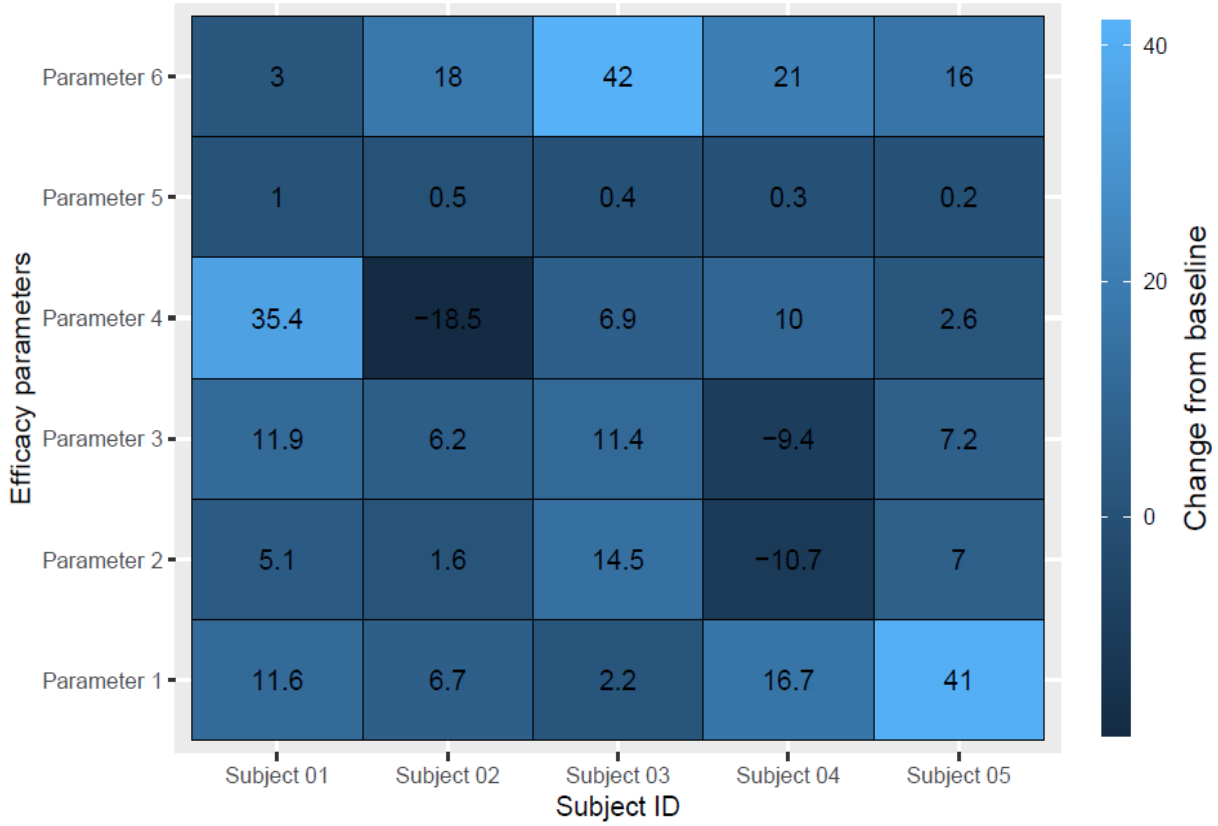

Figure 10. ggplot2 – Heat Map using labs() and guide_colourbar() function

Setting the attributes of plot

Modify a single plot's theme using the ***theme()*** function. Themes are a powerful way to customize the non-data components of your plots: i.e. titles, labels, fonts, background, gridlines, and legends. The ***scale_fill_gradientn()*** function can be used to set the gradient color, which takes a vector of reference *colours()* as its argument, and constructs a scale that linearly interpolates between the specified values.

The code for Figure 11 is shown below:

```
# install.packages("ggplot2")
library(ggplot2)
ggplot(dummy_data, aes(x = SUBJID, y = PARAM, fill = CHG)) +
    geom_tile(color = "black",
                lwd = 0.15,
                linetype = 1) +
     scale_fill_gradientn(colours = c("DarkRed", "DarkSalmon",  "White",
            "PaleTurquoise", "MediumTurquoise", "LightSkyBlue",
            "SlateBlue")) +
     geom_text(aes(label = CHG), color = "black", size = 3) +
     labs(x = "Subject ID",
          y = "Efficacy parameters",
          fill = "Change from baseline") +
     theme(panel.background = element_rect(color = "grey", linewidth = 0.5,
                fill = "white"),
           axis.ticks = element_line (color = "grey"),
           axis.line = element_line (color = "grey")) +
           guides(fill=guide_colourbar (title.position = "right",
                  barheight = 19,
                  barwidth = 0.8,
                  title.hjust = 0.5,
                  title.theme = element_text(angle = 90)
  ))
```

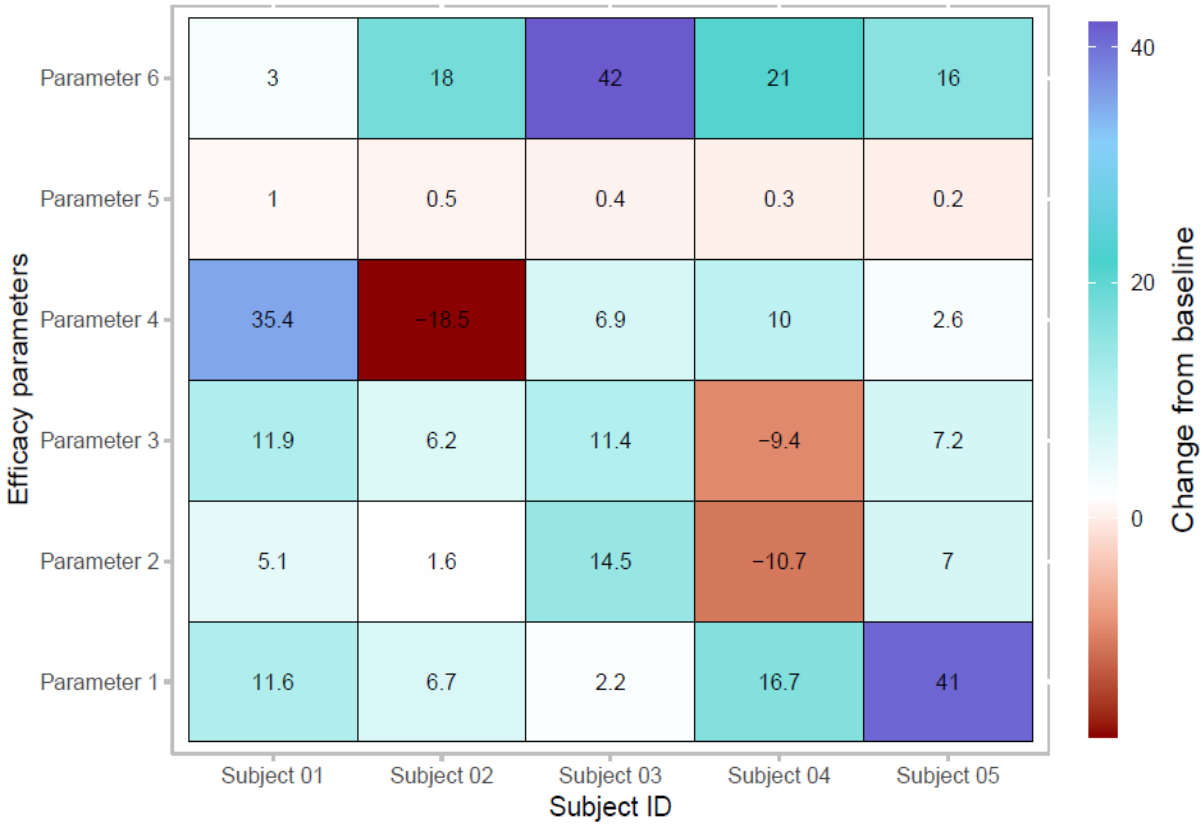

Figure 11. ggplot2 – Heat Map using theme() and scale_fill_gradientn() function

Summary: The *geom_tile()* function of the *ggplot2* package is introduced to generate the sample heat map, and you can use the functions (*geom_text(), labs(), guide_colourbar(), theme(), scale_fill_gradientn(),* etc.) to enhance the figures to better visualize the data.

CONCLUSION

This paper introduced using SAS and R to generate a specific heat map. The HEATMAPPARM statement in SGPLOT and the HEATMAPPARM statement in GTL can generate the same plot. The common statements and options were introduced to enhance the plot. The geom_tile() function of the ggplot2 package by R showed how to generate the same heat map. We can choose an appropriate method to visualize clinical data with heat maps according to our data visualization and reporting purposes.

REFERENCES

Pavani Potuganti, and Sree Nalluri, Eliassen Group Biometrics and Data Solutions.2021. “Graphical Representation of Clinical Data using Heat Map Graphs”. PharmaSUG 2021 - Paper DV-078. Available at https://www.lexjansen.com/pharmasug/2021/DV/PharmaSUG-2021-DV-078.pdf.

Kristen Much, Chapel Hill, Nicole Bader, Chapel Hill, 2017. “Building Heat Maps for Data Cleaning and Beyond”. Southeast SAS User Group Conference (SESUG) Paper RIV-138-2017. Available at https://www.lexjansen.com/sesug/2017/RIV-138.pdf.

SAS® 9.4 ODS Graphics: Procedures Guide, Sixth Edition

SAS® 9.4 Graph Template Language: Reference, Fifth Edition

José Carlos Soage. “Heat map in ggplot2”. Available at https://r-charts.com/correlation/heat-map-ggplot2.

ACKNOWLEDGMENTS

The author would like to thank Fan Zhang and Alice Chen for their advice and support.

CONTACT INFORMATION

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

Ternence Liu
Ternence.Liu@sanofi.com