Paper RIV52

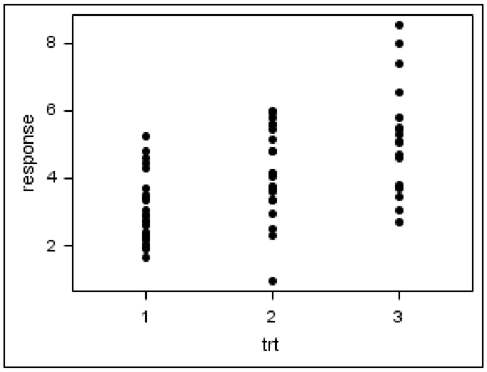
A Strip Plot Gets Jittered into a Beeswarm

Shane Rosanbalm, Rho, Inc.

ABSTRACT

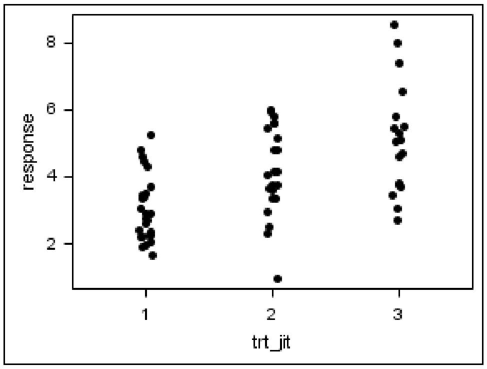
The beeswarm is a relatively new type of plot and one that SAS® does not yet produce automatically (as of version 9.4). For those unfamiliar with beeswarm plots, they are related to strip plots and jittered strip plots. Strip plots are scatter plots with a continuous variable on the vertical axis and a categorical variable on the horizontal axis (e.g., systolic blood pressure vs. treatment group). The strip plot is hamstrung by the fact that tightly packed data points start overlaying one another, obscuring the story that the data are trying to tell. A jittered strip plot seeks to remedy this problem by randomly moving data points off of the categorical center line. Depending on the volume of data and the particular sequence of random jitters, this technique does not always eliminate all overlays. In order to guarantee no overlays we must adopt a non-random approach. This is where the beeswarm comes in. The beeswarm approach is to plot data points one at a time, testing candidate locations for each new data point until one is found that does not conflict with any previously plotted data points. The macro presented in this paper performs the preliminary calculations necessary to avoid overlays and thereby produce a beeswarm plot.

BACKGROUND



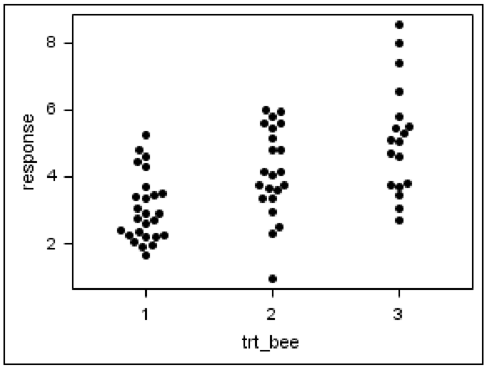
A **strip plot** is a scatter plot with a continuous variable on the vertical axis and a categorical variable on the horizontal axis. All of the data points within each category are vertically aligned.

One disadvantage of this vertical alignment is that a significant number of data points overlay one another.



A **jittered strip plot** attempts to reduce overlays by randomly moving data points by small amounts to the left and right.

Because the movements of the data points are random, some overlays still occur. Furthermore, many data points are moved unnecessarily.



A **beeswarm plot** is a relatively new type of plot that uses a computational algorithm to avoid overlays. The algorithm only moves data points when necessary, and even then the data point is only moved the minimum distance necessary to avoid overlays.

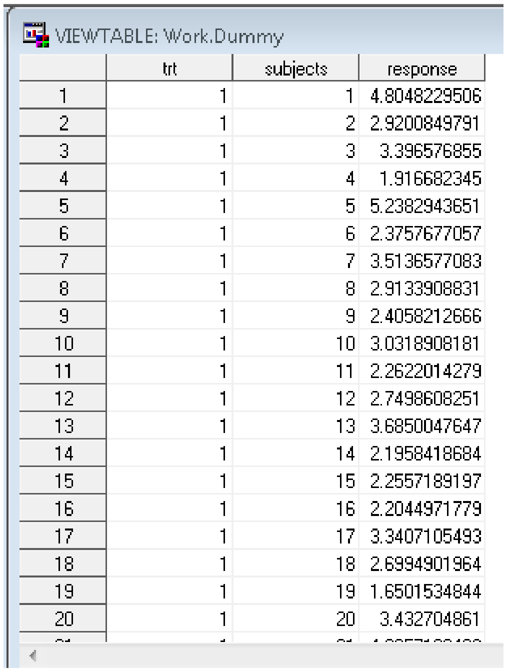
The macro presented in this paper implements the beeswarm algorithm.

SNEAKING UP ON A BEESWARM

In order to understand how to produce a beeswarm plot, it will be instructive to first look at how the strip plot and jittered strip plots are typically produced.

The random number function RANNOR is used to generate the simulated dataset that will be used throughout this paper. The code is included only to allow you to reproduce the results that follow. Attempting to read and understand this data step would not be time well spent.

```
data dummy;
   do trt = 1 to 3;
      do subjects = 1 to 30 - 4*trt by 1;
         response = sqrt(trt)*(rannor(1)+3);
         output;
      end;
   end;
run;
```

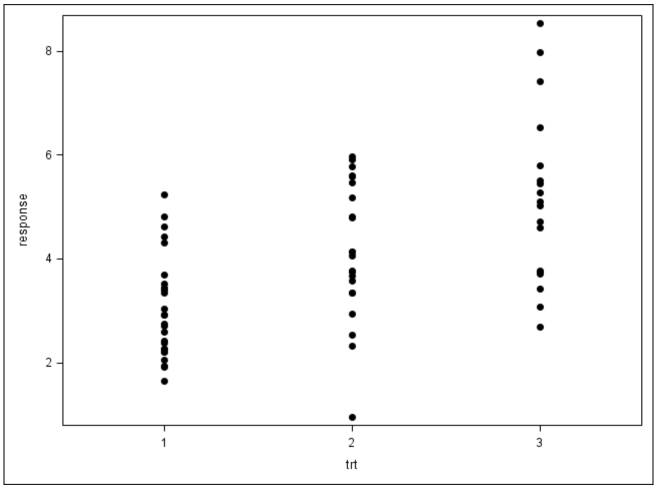


VIEWTABLE: Work.Dummy

	trt	subjects	response
1	1	1	4.8048229506
2	1	2	2.9200849791
3	1	3	3.396576855
4	1	4	1.916682345
5	1	5	5.2382943651
6	1	6	2.3757677057
7	1	7	3.5136577083
8	1	8	2.9133908831
9	1	9	2.4058212666
10	1	10	3.0318908181
11	1	11	2.2622014279
12	1	12	2.7498608251
13	1	13	3.6850047647
14	1	14	2.1958418684
15	1	15	2.2557189197
16	1	16	2.2044971779
17	1	17	3.3407105493
18	1	18	2.6994901964
19	1	19	1.6501534844
20	1	20	3.432704861

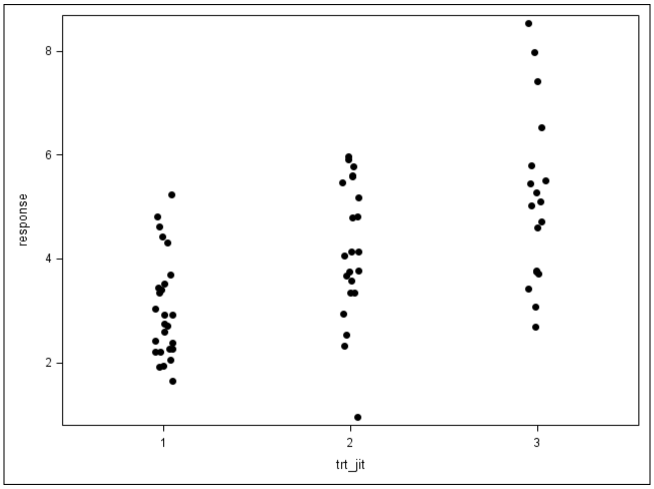
We begin with a strip plot. The code used to produce a strip plot is straightforward. Simply tell the SCATTER statement what your X= and Y= variables are, add some modest formatting options, and viola.

```
proc sgplot data=dummy;
   scatter x=trt y=response /
      markerattrs=(symbol=circlefilled);
   xaxis min=0.5 max=3.5 integer;
run;
```



We next turn to a jittered strip plot. The code used to produce a jittered strip plot represents only a slight alteration of the code used to produce a non-jittered strip plot. The key difference is that the SGPLOT code is preceded by a data step. In this data step a randomly jittered version of the original x-axis variable is created (trt_jit). This new variable is then used as the X= variable in the SCATTER statement.

```
/* new x-axis variable trt_jit is created in a data step */
data jitter;
   set dummy;
   trt_jit = trt - 0.05 + 0.1*ranuni(1);
run;

proc sgplot data=jitter;
   scatter x=trt_jit y=response /
      markerattrs=(symbol=circlefilled);
   xaxis min=0.5 max=3.5 integer;
run;
```



As we will see in the next section of this paper, the code used to produce a beeswarm plot is not substantially different from the code used to produce a jittered strip plot. The key difference will be that the preliminary data step is replaced by a preliminary macro call.

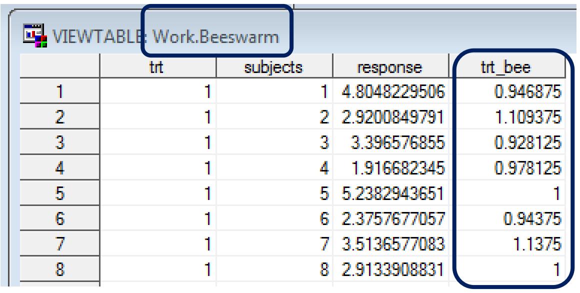
BEESWARM MACRO BASICS

In order to produce a beeswarm plot we will need to create an alternate version of the x-axis variable (like trt_jit, but non-random). This is accomplished by calling the beeswarm macro. The macro has 3 required parameters:

- data= (input dataset)
- respvar= (response/continuous variable)
- grpvar= (grouping/categorical variable)

The macro works by creating a new version of the original x-axis variable (*grpvar*_bee). For instance, the following call to the beeswarm macro creates a new variable named trt_bee.

```
%beeswarm(data=dummy
         ,respvar=response
         ,grpvar=trt
         );
```

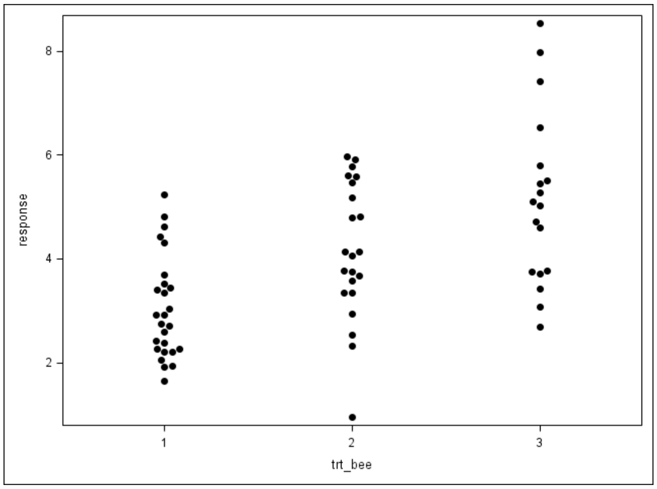


VIEWTABLE: Work.Beeswarm

	trt	subjects	response	trt_bee
1	1	1	4.8048229506	0.946875
2	1	2	2.9200849791	1.109375
3	1	3	3.396576855	0.928125
4	1	4	1.916682345	0.978125
5	1	5	5.2382943651	1
6	1	6	2.3757677057	0.94375
7	1	7	3.5136577083	1.1375
8	1	8	2.9133908831	1

Once the trt_bee variable has been created by the macro, adapting the jittered strip plot code to produce the beeswarm plot is straightforward; simply update the X= variable name in the SCATTER statement.

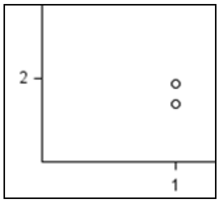
```
/* new x-axis variable trt_bee is created in a macro call */
%beeswarm(data=dummy
         ,respvar=response
         ,grpvar=trt
         );

proc sgplot data=beeswarm;
   scatter x=trt_bee y=response /
      markerattrs=(symbol=circlefilled);
   xaxis min=0.5 max=3.5 integer;
run;
```



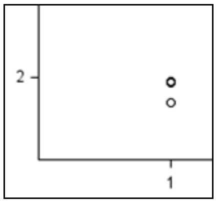
And that's all there is to it!

BEHIND THE CURTAIN

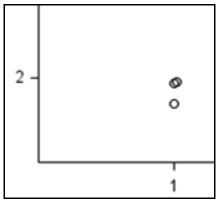
The algorithm that the beeswarm macro uses is based on the distance formula. Responses are first sorted from smallest to largest within each group. The smallest data point in the first group is then placed directly over the x-axis value of 1.



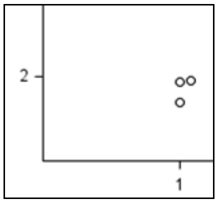
The second data point in the group is then evaluated to see whether or not it too can be placed directly over the x-axis value of 1 (i.e., whether or not its marker will overlay the first data point's marker). In the dataset used for this paper, the second data point does not overlay the first.



Moving on to the third data point in the group, we evaluate to see whether or not it too can be placed directly over the x-axis value of 1. Unfortunately, the y-value of the third data point is nearly identical to that of the second data point, resulting in significant overlay between the two data points.



When overlay is detected the initial candidate location is abandoned and a new candidate location is selected for evaluation. This new candidate location is given an x-value that is slightly larger than the original x-value. In this particular instance the new location still results in significant overlay.



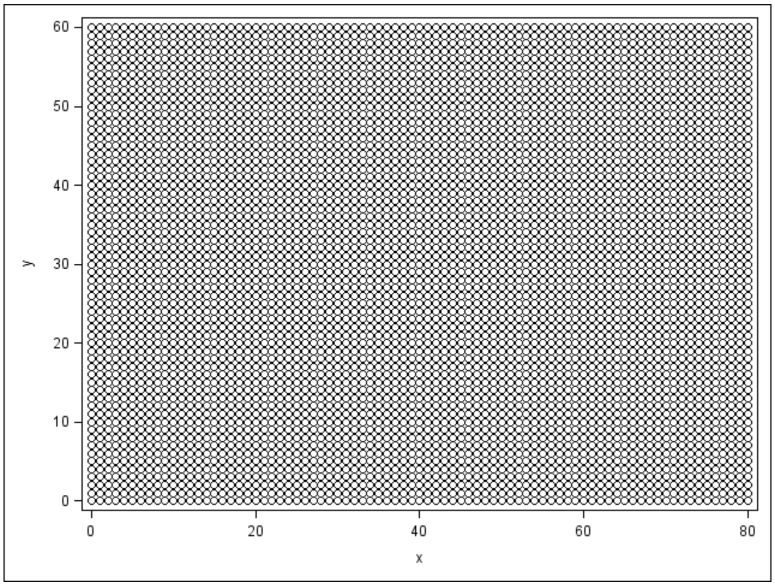
The process of evaluating new candidate locations continues until a location is found that puts the third data point a suitable distance from the second data point. In this particular instance the 5th candidate locations proves to be sufficiently far from the second data point to avoid an overlay.

The algorithm continues until every data point within a group has been found a location that does not overlay any other data point within the group.

NON-DEFAULT DIMENSIONS

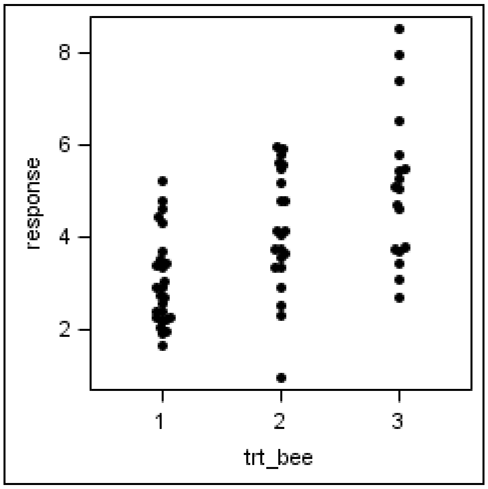
The default dimensions for SGPLOT output are 480px tall by 640px wide. A SCATTER plot produced with these default dimensions leaves enough space for approximately 60 markers vertically and 80 markers horizontally.

```
data fill;
   do y = 0 to 60;
      do x = 0 to 80;
         output;
      end;
   end;
run;

ods graphics /
   reset=all;

proc sgplot data=fill;
   scatter x=x y=y;
run;
```



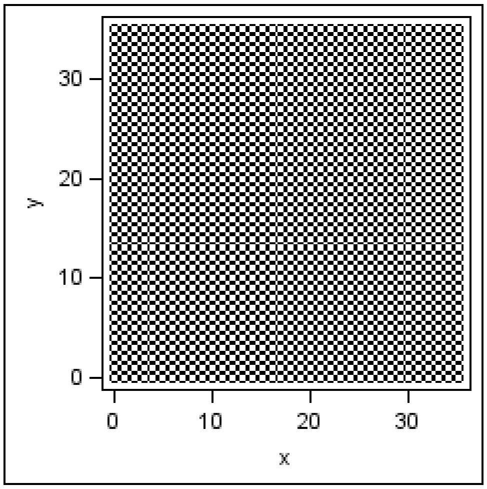
The beeswarm macro assumes these values of 60 and 80 in calculating whether or not data points will overlay one another. However, oftentimes we need to produce graphs with non-default dimensions. For instance, suppose that we need to produce a graph for a journal article with dimensions 2.5in tall by 2.5in wide. Calling the beeswarm macro as before will result in many of the data points overlaying one another.

```
%beeswarm(data=dummy
         ,respvar=response
         ,grpvar=trt
         );

ods graphics /
   width=2.5in height=2.5in;

proc sgplot data=beeswarm;
   scatter x=trt_bee y=response /
      markerattrs=(symbol=circlefilled);
   xaxis min=0.5 max=3.5 integer;
run;
```



These overlays occur because the beeswarm macro has assumed that there was room for 60 markers vertically and 80 markers horizontally. In reality there was room for far fewer markers in the 2.5in by 2.5in graph. Minimal experimentation quickly reveals that there is room for roughly 35 markers in both the vertical and horizontal directions in a graph that is 2.5in by 2.5 in.

```
data fill;
   do y = 0 to 35;
      do x = 0 to 35;
         output;
      end;
   end;
run;

ods graphics /
   width=2.5in height=2.5in;

proc sgplot data=fill;
   scatter x=x y=y;
run;
```

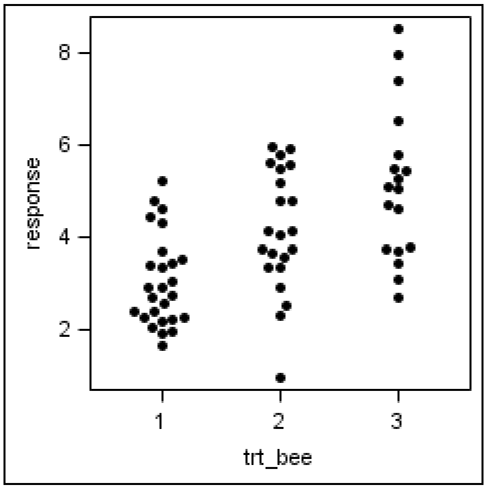


These vertical and horizontal fit values can be passed to the macro using the following optional parameters:

- rmarkers= (number of markers that will fit in the response/continuous direction; default=60)
- gmarkers= (number of markers that will fit in the grouping/categorical direction; default=80)

Specifying values of rmarkers=35 and gmarkers=35 allows the macro to adjust to the new dimensions and avoid overlays.

```
%beeswarm(data=dummy
         ,respvar=response
         ,grpvar=trt
         ,rmarkers=35
         ,gmarkers=35
         );

ods graphics /
   width=2.5in height=2.5in;

proc sgplot data=beeswarm;
   scatter x=trt_bee y=response /
      markerattrs=(symbol=circlefilled);
   xaxis min=0.5 max=3.5 integer;
run;
```



In summary, whenever you need to produce a beeswarm plot with non-default dimensions, begin by experimenting to see how many markers will fit vertically and horizontal. Then pass these values to the macro using the optional parameters rmarkers= and gmarkers=.

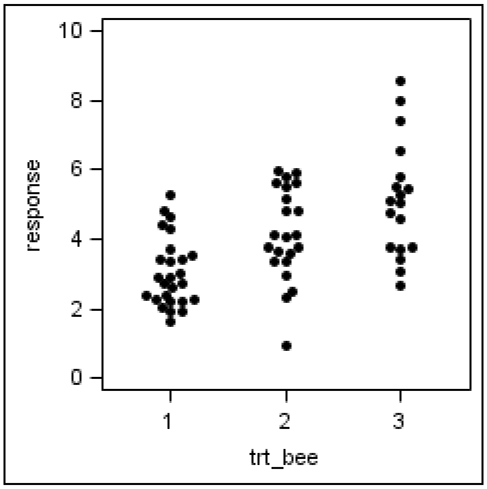
Note: non-default marker sizes would be handled similarly. Experiment to determine how many larger or smaller markers will fit vertically and horizontal, and then use rmarkers= and gmarkers= to pass this information to the macro.

NON-DEFAULT AXES

The default axis behavior in ODS Graphics is to sometimes allow the min and max data points to be unbounded by tick marks (except when the data points get very close to the next tick mark, in which case ODS Graphics reluctantly allows the data points to be bounded by a tick mark). Correspondingly, the beeswarm macro assumes that the min and max data points will extend to the edge of the data space.

However, in some situations we will have bounding tick marks (i.e., the data points will not extend to the edge of the data space). For example, suppose our sponsor requests that we extend the y-axis range in our above graph to cover the interval [0, 10]. This request crams the data points into a smaller space, resulting in a small amount of undesirable overlays.

```
%beeswarm(data=dummy
         ,respvar=response
         ,grpvar=trt
         ,rmarkers=35
         ,gmarkers=35
         );

ods graphics / width=2.5in height=2.5in;

proc sgplot data=beeswarm;
   scatter x=trt_bee y=response /
      markerattrs=(symbol=circlefilled);
   xaxis min=0.5 max=3.5 integer;
   yaxis min=0 max=10;
run;
```

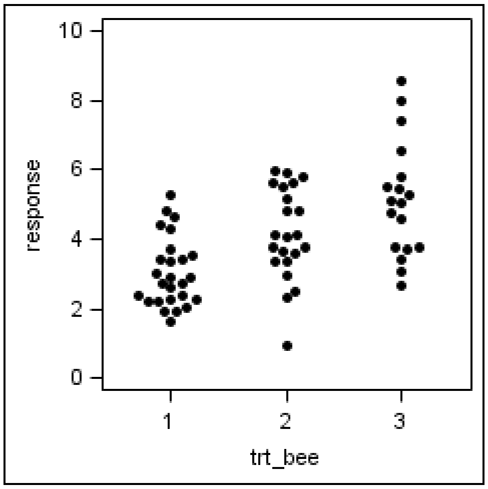


The extended y-axis min and max values can be passed to the macro using the following optional parameters:

- rmin= (response axis minimum)
- rmax= (response axis maximum)

Specifying values of rmin=0 and rmax=10 allows the macro to adjust to the new axis range and avoid overlays.

```
%beeswarm(data=dummy
         ,respvar=response
         ,grpvar=trt
         ,rmarkers=35
         ,gmarkers=35
         ,rmin=0
         ,rmax=10
         );

ods graphics /
   width=2.5in height=2.5in;

proc sgplot data=beeswarm;
   scatter x=trt_bee y=response /
      markerattrs=(symbol=circlefilled);
   xaxis min=0.5 max=3.5 integer;
   yaxis min=0 max=10;
run;
```



In summary, whenever you need to produce a beeswarm plot with a y-axis range that extends beyond the observed min or max data values, pass the extended range to the macro using the optional parameters rmin= and rmax=.

CONCLUSION

The beeswarm plot improves upon the jittered strip plot by moving data points based on a non-random algorithm. Data points are only moved when necessary, and the distance moved is only the minimum necessary to avoid overlays.

The beeswarm macro does not produce a plot directly. The macro works by creating a new x-axis variable, and that variable is then used as the X= option in a SCATTER statement.

The beeswarm macro assumes room for 60 markers vertically and 80 markers horizontally. Optional parameters rmarkers= and gmarkers= are used to adjust for non-default dimensions.

The beeswarm macro assumes that markers extend to the edge of the data space. Optional parameters rmin= and rmax= are used to adjust for the presence of bounding tick marks.

MACRO SOURCE CODE

The beeswarm macro source code is available for direct download at graphics.rhoworld.com/tools/beeswarm. Alternatively, send email requests to graphics@rhoworld.com.

CONTACT INFORMATION

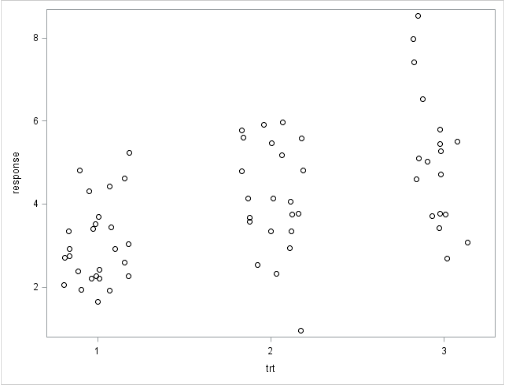
Your comments and questions are valued and encouraged. Contact the author at:

Shane Rosanbalm
Rho, Inc
6330 Quadrangle Dr.
Chapel Hill, NC 27517
919.595.6273
E-mail: shane_rosanbalm@rhoworld.com
Web: www.rhoworld.com

APPENDIX 1: THE SAS 9.4 JITTER OPTION

In SAS 9.4 there is a JITTER option that has been added to the SCATTER statement within SGPLOT. When we add this option to our original strip plot code we get the following.

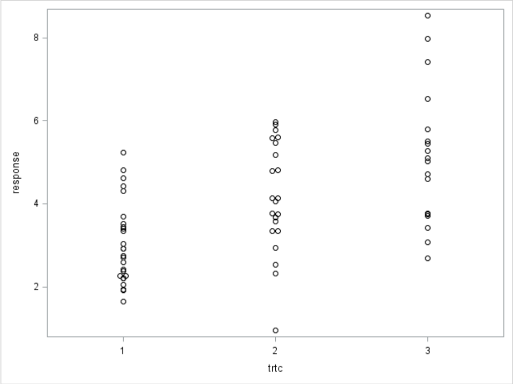
```
proc sgplot data=dummy;
   scatter x=trt y=response / jitter
      markerattrs=(color=black);
   xaxis min=0.5 max=3.5 integer;
run;
```



This is reminiscent of the jittered strip plot that we produced earlier, but without the added fuss of the preliminary data step. If you want a jittered strip plot, use the JITTER option.

Switching from a numeric x-axis variable to a character x-axis variable, the JITTER option gives us something slightly different.

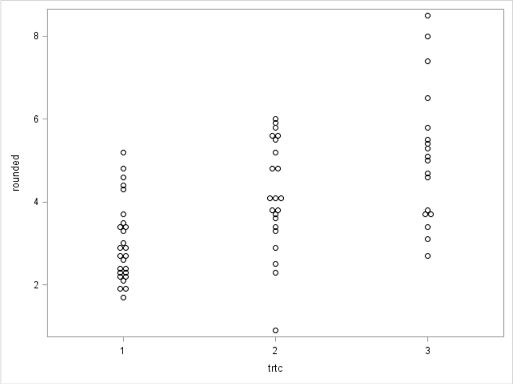
```
data dummyc;
   set dummy;
   trtc = put(trt,1.);
run;

proc sgplot data=dummy;
   scatter x=trtc y=response / jitter
      markerattrs=(color=black);
run;
```



Notice how this is starting to behave like the beeswarm plot, but is still quite a ways off. A few data points are pushed out to the side, but only when their y-values are nearly identical. Many significant overlays still remain.

Now, suppose that our original dummy data had been rounded to the nearest tenth. With this data, the JITTER option will actually give us something very beeswarm-like.

```
data rounded;
   set dummyc;
   rounded = round(response,0.1);
run;

proc sgplot data=rounded;
   scatter x=trtc y=rounded / jitter
      markerattrs=(color=black);
run;
```



This illustration is not meant to suggest that you should round your data to produce a beeswarm plot. However, what it does suggest is, if you happen to have a sufficiently discrete response variable (e.g., count data, temperature in degrees Fahrenheit, etc.), the JITTER option could be used to achieve a beeswarm-like effect.

APPENDIX 2: PANELED GRAPHS

The beeswarm macro can be used with SGPANEL as well. The general strategy is as follows:

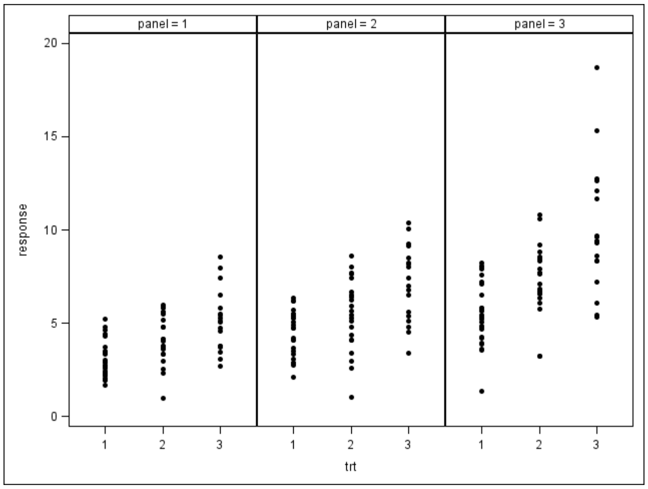
- Experiment with SGPANEL to deduce the best values for rmarkers= and gmarkers= within a panel.
- Use a macro DO-loop to pass data to the beeswarm macro one panel at a time.
- Stack these panel-specific beeswarm datasets back together into one large beeswarm dataset.

Let’s walk through an example. We begin by adding an outer DO-loop to the code that generates the dummy data for this paper. This gives us the same data as before for panel=1, plus additional data for panel=2 and panel=3.

```
data dummy_panel;
   do panel = 1 to 3;
      do trt = 1 to 3;
         do subjects = 1 to 30 - 4*trt*floor(sqrt(panel)) by 1;
            response = sqrt(trt)*sqrt(panel)*(rannor(1)+3);
            output;
         end;
      end;
   end;
run;
```

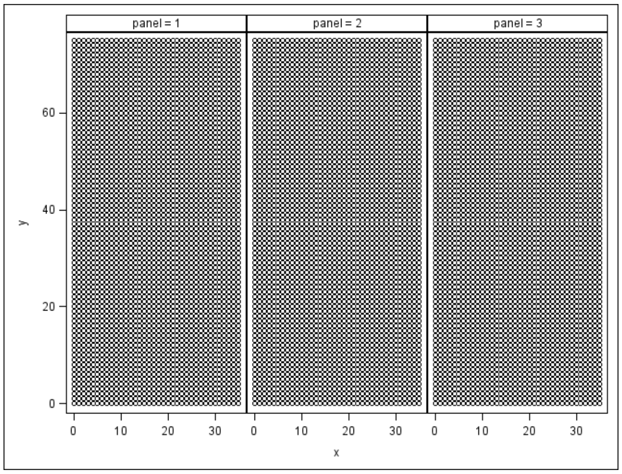
A paneled strip plot of this data is straightforward to produce. We will use SGPANEL with the COLUMNS= option on the PANELBY statement.

```
proc sgpanel data=dummy panel;
   panelby panel / columns=3;
   scatter x=trt y=response /
      markerattrs=(symbol=circlefilled);
   colaxis min=0.5 max=3.5 integer;
run;
```



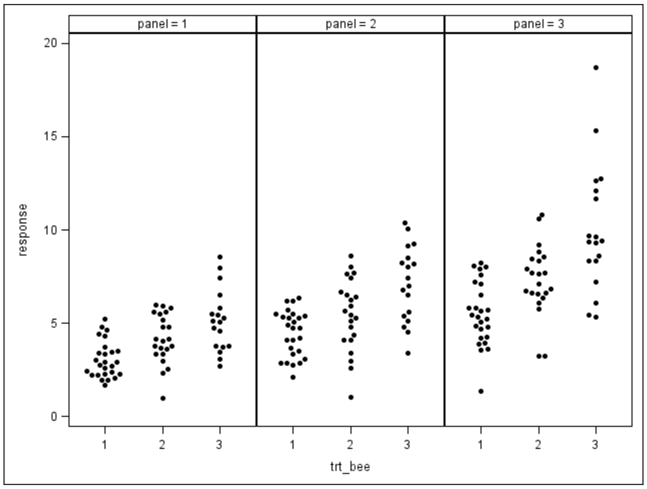
Before calling the beeswarm macro, first experiment to see how many markers will fit into a panel. Using default dimensions and marker sizes, a 3-column panel graph has room for 75 markers vertically and 35 markers horizontally.

```
data fill;
   do panel = 1 to 3;
      do y = 0 to 75;
         do x = 0 to 35;
            output;
         end;
      end;
   end;
run;

proc sgpanel data=fill;
   panelby panel /
      columns=3;
   scatter x=x y=y;
   colaxis thresholdmin=0
      thresholdmax=0;
   rowaxis thresholdmin=0
      thresholdmax=0;
run;
```



Next, use a macro DO-loop to pass data to the beeswarm macro one panel at a time. Note that in addition to specifying rmarkers=75 and gmarkers=35 in the macro call, we also specify rmin=0 and rmax=20. The rmin= and rmax= are necessary because of the bounding tick marks produced by SGPANEL.

```
%macro beeswarm_by_panel;

   /* pass data to the beeswarm macro one panel at a time */
   %do i = 1 %to 3;

      data panel&i;
         set dummy_panel;
         where panel eq &i;
      run;

      %beeswarm(data=panel&i
               ,respvar=response
               ,grpvar=trt
               ,rmarkers=75
               ,gmarkers=35
               ,rmin=0
               ,rmax=20
               ,out=beeswarm&i
               );

   %end;

   /* stack individual beeswarm datasets back together */
   data beeswarm;
      set %do i = 1 %to 3; beeswarm&i %end;;
   run;

%mend beeswarm_by_panel;


%beeswarm_by_panel;


proc sgpanel data=beeswarm;
   panelby panel / columns=3;
   scatter x=trt_bee y=response /
      markerattrs=(symbol=circlefilled);
   colaxis min=0.5 max=3.5 integer;
run;
```


panel = 1
panel = 2
panel = 3
response
trt_bee