The attached graph was generated using SAS/GRAPH running on a VAX/VMS computer system. PROC GLSIDE was executed using the ANNOTATE option to direct the creation of the plot from an annotate data set. The data set was created in a data step using as input the mean attribute values for each sweetener as well as the p-values for the test of no difference between sweeteners for each attribute as generated in a PROC GLM step. The graphics output was printed on a DEC LN03 laser printer using a VMS PRINT command.

The purpose of the plot is to be able to compare two sweeteners with respect to their (multivariate) attributes as perceived by a group of trained sensory evaluation judges. This type of plot, often called a "spider plot" because of its web-like appearance, is in common use for the graphical display of sensory profile information. Although the SAS/GRAPH procedure GCHART has a STAR plotting feature, it plots multiple classes (in this case, sweeteners) as separate "stars" instead of overlaying the classes on the same star. It is much easier to see the differences between the sweeteners when they are overlaid on the same set of axis rays.

An additional feature of the "spider" plot as implemented using ANNOTATE, is that the origin (zero mean) is a polygon with a fixed, non-zero radius rather than a single point. When a sweetener does not have a given attribute present in its profile a single-point origin can tend to make it very hard to see what is happening with the "web" of the sweetener.

Because spider plots are used frequently for displaying this type of profile data, the determination of the required axis ray length scaled to fit the data, the implementation of a varying number of attributes and sweeteners, and the capture of the p-values from the GLM hypothesis tests were automated into a single SAS job.
Attribute Means of Two Sweeteners
Tasted in a Lemonade Beverage
(Evaluated by 13 Judges Using a 0–15 Scale)

For bold attributes the samples are significantly different at $\alpha=0.10$
Most Creative Use of the Software

**Kermon M. Gibes**

Statistician

The NutraSweet Company

601 E. Kensington

Mount Prospect, IL 60056

Source program for generating a "spider" plot

**LANGUAGE:** SAS 5.16

**SYSTEM:** VAX/VMS

**INPUT:** ASCII data file 'SUGI13 PLOT. DAT', 

**OUTPUT:** BASE.log, .lis and .gaf-files

---

```
&
SUGI 13 SAS/GRAPH Software Competition Entry

FILESPEC: [GIBES.SAS|SUGI13 PLOT.SAS]

SYSTEM: VAX/VMS

LANGUAGE: SAS 5.16

ABSTRACT: Source program for generating a "spider" plot

DATE: March 1, 1988

INPUT: ASCII data file 'SUGI13 PLOT. DAT'

/ *

options linesize=80 nodate;

title1 'I

title2 'Sweetener Profile -- SENSORY EVALUATION -- Descriptive';

title3 'Comparison of two sweeteners';

*** set formats for sweeteners and attribute scales

proc format;

value sample

203,764 = 'Sweetener A'

412,628 = 'Sweetener B'

value scale

1 = 'Orange Complex'

2 = 'Fresh'

3 = 'Cooked'

4 = 'Peel Oil'

5 = 'Distilled oil'

6 = 'Other Citrus'

7 = 'Lime'

8 = 'Terpene'

9 = 'Grapefruit'

10 = 'Tangerine'

11 = 'Sweet'

12 = 'Sour'

13 = 'Bitter'

14 = 'Astringent'

15 = 'Throat Burn'

16 = 'Lemon/Lime Aftertaste'

17 = 'Sweet Aftertaste'

18 = 'Sour Aftertaste'

19 = 'Bitter Aftertaste'

run;

*** create new treatment variable

```

```

* .... create new treatment variable;

treat = put( sample, sample. );

run;

* *** create one variable (score) from the attribute variables

```

```

data work;

set input;

array rarray{*} r1-r19;

do scale = 1 to dim( rarray );

score = rarray[ scale ];

output;

end;

keep judge sample treat scale score;

run;

** ** sort the data by scale and capture the GLM output using PRINTTO

```

```

proc sort data=work;

by scale;

run;

proc printto unit=20 new;

proc glm data=work;

by scale;

classes judge sample;

model score = judge sample;

format scale scale. sample sample. ;

title5 'Repeated Measures ANOVA of Attribute data';

run;

proc printto;

* *** parse the GLM output to get the p-values

```

```

```

data pvalue;

length keyword $ 80;

infile for020 I

input @2 keyword $ @;

retain flag 0;

if keyword _ 'SOURCE' then do;

input. charI $ char2 $ type $;

if type = 'III' then flag = 1;

end;

else if keyword _ 'SAMPLE' and flag then do;

input num1 num2 num3 pvalue;

scale + 1; flag = 0;

output;

end;

else input;

keep scale pvalue;

run;

* *** merge the p-values back into the raw data

```

```

```

proc printto;

* ...
case the GLM output to get the p-values

```

```

data pvalue;

length keyword $ 80;

infile for020 I

input @2 keyword $ @;

retain flag 0;

if keyword = 'SOURCE' then do;

input char1 $ char2 $ type $;

if type = 'III' then flag = 1;

end;

else if keyword = 'SAMPLE' and flag then do;

input num1 num2 num3 pvalue;

scale + 1;

flag = 0;

output;

end;

else input;

keep scale pvalue;

run;

* *** merge the p-values back into the raw data

```

```

```

data work;

merge pvalue work;

by scale;

run;
```

---

```
**SAS Code**

```sas
proc sort data=work; by treat scale; run;

*** get the means by treatment group and scale ***************
 proc univariate data=work noprint; by treat scale; var score pvalue; output out=means mean=score pvalue; run;

*** get the number of scales per treatment group **************
 proc freq data=means; table treat / noprint out=Freq;
 run;

*** determine and save the number of treatments and scales ********
data parms;
i = 1;
set freq nobs=n point=i;
flag = 1;
ntreat = n;
scale = count;
do i = 2 to n;
set freq nobs=n point=i;
if count ne nscale then flag=0;
end;
i = n;
if flag=1 then output;
keep ntreat nscale;
stop;
run;

*** get the maximum mean over all treatments and scales ***********
 proc univariate data=means noprint; var score; output out=maxs max=max;
 run;

*** add the maximum mean info to the PARMS data set **************
data parms;
if n = 1 then set maxs;
set parms;
run;

*** now merge all parameter info into the MEANS data set ************
data means;
if n = 1 then set parms;
set means;
run;

*** generate plot command data sets from means **********************
(note: commands to generate a polygon must be in grouped observations,
hence to insure this the inner polygon, the data polygons, the legend
* and the rays/labels are output to separate data sets and combined
* later;)

data legend axis inner polygon;
set means;

set annotate variable types;
length angle x y line rotate size $; length xsys ysys hsys position $ 11 length function style $ 8; length text $ 200;

set parameter values;
if n = 1 then do;
shrink = 196/259;
arc = 2*acos(-1)/nscale;
axis_l = ceil( maxs );
end;

retain shrink / shrinkage factor applied to y-axis to obtain
uniform metric on LN03 printers (plot area is 196cm by 259cm) */
arc / angle between adjacent rays */
asix_l / ray length */
xcenter 50 / x-axis center */
ycenter 50 / y-axis center */
fs radius 30 / radius of the axis rays */
fr radius 31 / radius position for start of ray labels */
fr radius 5 / radius of the inner polygon */
axes_w 0.65 / axis label character size */
legd ls 0.60 / legend label character size */
flagl 1 / flag variable initially set to 1 */
alpha 0.10 / alpha level for significance */

xsys = "3"; / absolute screen units for x-axis */
ysys = "3"; / absolute screen units for y-axis */
hsys = "4"; / absolute screen cell units for size */
if n = 1 then do;
/* draw axis scale information one time only */
fraction = "LABEL"; text = "Ray length"; compress(put( axis_l, 3. ) || " pts" );
style = "COMPLEX";
size = 0.75;
x = 90;
y = 5;
position = 4;
xsys = "5";
ysys = "5";
output legend;
end;

if first. treat then do;
/* increment line style and draw legend entry for each treat */
line = 1;
legends = 5;
legendx = 3;
xsys = "5";
ysys = "5";
line = "FRAME";
x = 5 + legends;
y = legendy;
output legend;
function = "DRAW";
```

**Note:** The above code is a fragment of SAS code used for statistical analysis, specifically for generating graphical outputs. It includes sorting, univariate analysis, frequency tables, parameter determination, maximum mean calculation, and plotting commands.
**......need only variables used by annotate:**

```
keep
  style hsys ysys x y function line size
  angle rotate text position;
run;
```

**......combine all plot command data sets ******************************************

```
data spider;
set legend axis inner polygon;
run;
```

**......set graphics options for use on LNO3 printer ******************************

```
options
gsname=replace
gfname=plot
device=tekln01
rotate
nosprompt
nodisplay
```

**......run GELIDE using the ANNOTATE option to draw the plot ***************

```
proc gslide anno_spider;
title1 h=0.75 f=complex y=c w=2 'Kernon Gibes Most Creative Use of the Software Monochromatic'
title2 h=1 f=complex
title3 'Attribute Means of Two Sweeteners' 'Tasted in a Lemonade Beverage' 'Evaluated by 3 Judges Using a 0-15 Scale'
```

**......run only variables used by annotate:**

```
xsys='3';
ysys='3';
scale=3;
line='line';
style='"';
if first. treat then i=0;
if first.scale then i=1;
  radius = r.radius + macore * (r.radius-l.radius) / axis_l;
  x = xcenter + radius * cos( (axisno-l)*arc );
y = ycenter + shrink * r.radius * sin( (axisno-l)*arc );
else function = 'LABEL';
```

**......for each scale, increment the axis number, draw the ray and its label only once (by the sorting, there will be more than one "first.scale", but the flag variable insures this is done one time only):**

```
axijson;
line = 1;
ysys='3';
yysys='3';
function='MOVE';
x = xcenter + r.radius * cos( (axijson-l)*arc );
y = ycenter + shrink * r.radius * sin( (axijson-l)*arc );
```

**......output commands to draw inner polygon:**

```
if axijson then function = 'POLY';
else function = 'POLYCONT';
```

**......assumed SCALE was numeric (for sorting), but format was supplied for labeling the rays:**

```
text = put( scale, scale.);
size = axis lsi;
```

**......if significant, switch to bold font:**

```
if pvalue < alpha then style = 'XSWISS';
else style = 'COMPLEX'
```

**......if angle > 90 and angle < 270 then do:**

```
position = '4';
angle = angle - 180;
end;
```

**......assumed SCALE was numeric (for sorting), but format was supplied for labeling the rays:**

```
text = put( scale, scale.);
size = axis lsi;
```

**......draw the next edge of polygon for this treat:**

```
xsys='3';
ysys='3';
scale=3;
line='line';
style='"';
if first. treat then i=0;
if first.scale then i=1;
  radius = r.radius + macore * ( r.radius-l.radius ) / axis_l;
  x = xcenter + radius * cos( ( i-1)*arc );
y = ycenter + shrink * radius * sin( ( i-1)*arc );
else function = 'LABEL';
```

**......if significant, switch to bold font:**

```
if pvalue < alpha then style = 'XSWISS';
else style = 'COMPLEX'
```

**......if angle > 90 and angle < 270 then do:**

```
position = '4';
angle = angle - 180;
end;
```

**......display the next edge of polygon for this treat:**

```
xsys='3';
ysys='3';
scale=3;
line='line';
style='"';
if first. treat then i=0;
if first.scale then i=1;
  radius = r.radius + macore * ( r.radius-l.radius ) / axis_l;
  x = xcenter + radius * cos( ( i-1)*arc );
y = ycenter + shrink * radius * sin( ( i-1)*arc );
else function = 'LABEL';
```

**......combine all plot command data sets ******************************************