To Macro or Not to Macro

That is the question

Mike Atkinson
Acko Systems Consulting Inc

Macro Variables vs Actual Macros

- I highly recommend using macro **variables**, which can be assigned and referenced within bare SAS code
- I admit there are *some* situations where SAS macros are useful
- *But in my opinion*, SAS macros are often used in situations that could be handled without macros (usually by organising the data differently)

Advantages of using Macro Variables

- Can make SAS code easier to modify and reuse if %let statements are used at the top of the code to assign values to macro variables
- Reduces chance of error – e.g. running code with wrong parameter value(s) – if all values that need to change are right at the top of the code: requiring the use of macro variables

Ways to assign values to macro variables

- %let statement(s) to assign stuff to macro variables, outside of a data step
- proc SQL, select values and put into a macro variable, including lists of values
- call symput (and call symputx)

Example of setting macro variables

```
* Set fiscal year start date;
%let startdt     = 20150401;



* Automatically generate other dates based on start date;
%let startdt_sas = %sysfunc(inputn(&startdt, yymmdd8.));
%let enddt_sas   = %sysfunc(intnx(month, &startdt_sas, 11, E));

%let startdt_dt    = %sysfunc(putn(&startdt_sas, date9.)):00:00:00;
%let enddt_plus1_dt  = %sysfunc(putn(&enddt_sas, date9.)):00:00:00;
```

Disadvantages of using SAS Macros

```
%macro stuff(blah, blah);
     %do something;
%mend stuff;

%stuff(blarg, blarg2);
```

In practice:

- using SAS macros usually makes it more difficult to debug and solve problems
- Extensive use of SAS macros make it more difficult to determine what SAS code is doing

Why are there SAS Macros?

- In the beginning, SAS did not have macros
  - Only within a data step was there the capability for looping and branching
  - Programmers of other languages feel thwarted by SAS logic (or perceived lack thereof)
- Macros were added in SAS V5 (early 80's)
  - Looping and branching could then be done outside the data step
  - Programmers from other languages happy
  - Other SAS programmers scared

When I have used SAS macros

- Utilities, that will be used many times, such as:
  - Dropping a table if it exists (but doing nothing if it doesn't)
  - Checking whether two datasets have the same number of rows (and throwing an error if not)
- Creating a series of multi-part reports
  - E.g. multiple proc print, proc report, proc tabulate, etc steps must be called in succession to generate each report

  (Although I would accomplish the data preparation first without using SAS macros)
- When creating SAS macro variables – at the top of a program – so that %IF and %ELSE could be used
  - (although I try to set macro variables with %let statements if possible)
- Some other situations when I could not accomplish something with base SAS code

What I do not recommend (as an alternative to using SAS macros)

- ✖ Copying and pasting code, such a SAS data step, resulting in multiple similar steps

What I do recommend (as an alternative to using SAS macros)

- ✓ Rewriting the code to use classification variables, with BY statement and joins or merges, to eliminate the need for macros

Demo using SAS University Edition

The End

- or “oh oh, something broke!”

SASHELP.FISH

View: Column names Filter: (none)

Total rows: 159 Total columns: 7

	Species	Weight	Length1	Length2
1	Bream	242	23.2	25.4
2	Bream	290	24	26.3
3	Bream	340	23.9	26.5
4	Bream	363	26.3	29
5	Bream	430	26.5	29
6	Bream	450	26.8	29.7
7	Bream	500	26.8	29.7
8	Bream	390	27.6	30
9	Bream	450	27.6	30
10	Bream	500	28.5	30.7
11	Bream	475	28.4	31
12	Bream	500	28.7	31
13	Bream	500	29.1	31.5

SASHELP.FISH × | Fish_1.sas ×

CODE | LOG | RESULTS

```
1
2 * Work with Bream only;
3
4 data bream;
5     set sashelp.fish;
6     where species = 'Bream'
7       and weight is not null;
8 run;
9
10
11 * Get mean;
12
13 proc sql  noprint;
14     select mean(weight) into :mn_wgt
15     from bream;
16 quit;
17
18
19 * Compare bream with bream mean weight;
20
21 data bream_compare;
22     set bream;
23     if (weight < &mn_wgt)        then lt_mn = 1;
24     if (weight > &mn_wgt)        then gt_mn = 1;
25 run;
```

```
* Count them;

proc sql;
    create table bream_stats as
    select count(*)          as n_bream,
           sum(lt_mn)        as cnt_lt_mn,
           sum(gt_mn)        as cnt_gt_mn
    from bream_compare;
quit;


* Calculate percentages;

proc sql;
    create table bream_pct as
    select cnt_lt_mn / n_bream          as pct_lt_mn        format=percent7.,
           cnt_gt_mn / n_bream          as pct_gt_mn        format=percent7.
    from bream_stats;
quit;
```

SASHELP.FISH × | Fish_1.sas ×

CODE | LOG | RESULTS | OUTPUT DATA

Table: WORK.BREAM_PCT | View: Column names | Filter: (none)

Total rows: 1 Total columns: 2

	pct_lt_mn	pct_gt_mn
1	53%	47%

SASHELP.FISH × | Fish_2.sas ×

CODE | LOG | RESULTS

```
* Get the mean weight over the four fish combined;

proc sql  noprint;
    select mean(weight) into :mn_wgt
    from sashelp.fish
    where species in ('Bream', 'Pike', 'Perch', 'Whitefish');
quit;



* Make a macro to look at one species at a time;


%macro pct_fish(species);


* Look at just the one species;

data &species;
    set sashelp.fish;
    where upcase(species) = upcase("&species")
      and weight is not null;
run;
```

SASHELP.FISH × | Fish_2.sas ×

CODE | LOG | RESULTS

```
* Get mean;

proc sql  noprint;
    select mean(weight) into :mn_wgt_&species
    from &species;
quit;


* Compare bream with bream mean weight and overall mean;

data &species._compare;
    set &species;
    if (weight < &mn_wgt)            then lt_mn = 1;
    if (weight > &mn_wgt)            then gt_mn = 1;
    if (weight < &mn_wgt_&species)   then lt_mn_species = 1;
    if (weight > &mn_wgt_&species)   then gt_mn_species = 1;
run;
```

SASHELP.FISH × | Fish_2.sas ×

CODE | LOG | RESULTS

```
* Count them;

proc sql;
    create table &species._stats as
    select count(*)            as n_species,
           sum(lt_mn)          as cnt_lt_mn,
           sum(gt_mn)          as cnt_gt_mn,
           sum(lt_mn_species)  as cnt_lt_mn_species,
           sum(gt_mn_species)  as cnt_gt_mn_species
    from &species._compare;
quit;


* Calculate percentages;

proc sql;
    create table &species._pct as
    select "&species"                      as species,
           cnt_lt_mn / n_species           as pct_lt_mn          format=percent7.,
           cnt_gt_mn / n_species           as pct_gt_mn          format=percent7.,
           cnt_lt_mn_species / n_species   as pct_lt_mn_species  format=percent7.,
           cnt_gt_mn_species / n_species   as pct_gt_mn_species  format=percent7.
    from &species._stats;
quit;
```

SASHELP.FISH | Fish_2.sas

CODE | LOG | RESULTS

```
            cnt_lt_mn_species / n_species as pct_lt_mn_species  format=percent7.,
            cnt_gt_mn_species / n_species as pct_gt_mn_species  format=percent7.
    from &species._stats;
quit;

%mend pct_fish;


* Run the macro for each fish species;

%pct_fish(bream);
%pct_fish(pike);
%pct_fish(perch);
%pct_fish(whitefish);

* Combine the four fish species info;

data fish_final;
    length species $ 20;
    set bream_pct
        pike_pct
        perch_pct
        whitefish_pct;
run;
```

SASHELP.FISH × | Fish_2.sas ×

CODE | LOG | RESULTS | OUTPUT DATA

Errors, Warnings, Notes

Errors (8)

ERROR 22-322: Syntax error, expecting one of the following: a name, a quoted string, a numeric constant, a datetime constant,

ERROR 22-322: Syntax error, expecting one of the following: a name, a quoted string, a numeric constant, a datetime constant,

ERROR 22-322: Syntax error, expecting one of the following: a name, a quoted string, a numeric constant, a datetime constant,

ERROR 22-322: Syntax error, expecting one of the following: a name, a quoted string, a numeric constant, a datetime constant,

ERROR 22-322: Syntax error, expecting one of the following: a name, a quoted string, a numeric constant, a datetime constant,

```
WARNING: Apparent symbolic reference MN_WGT_ not resolved.
NOTE 137-205: Line generated by the invoked macro "PCT_FISH".
130         quit;   * Compare bream with bream mean weight and overall mean;  data &species._comp
130       ! (weight < &mn_wgt)           then lt_mn = 1;     if (weight > &mn_wgt)           then
130       ! &mn_wgt_&species)  then

            _
            22
ERROR 22-322: Syntax error, expecting one of the following: a name, a quoted string, a numeric co
              a missing value, bitstring, INPUT, PUT.

WARNING: Apparent symbolic reference MN_WGT_ not resolved.
NOTE: Line generated by the invoked macro "PCT_FISH".
130                                          &mn_wgt_&species)  then gt_mn_species = 1; run;
```

SASHELP.FISH | *Fish_2.sas

CODE | LOG | RESULTS | OUTPUT DATA

Table: WORK.FISH_FINAL | View: Column names | Filter: (none)

Total rows: 4 Total columns: 5 Rows 1-4

	species	pct_lt_mn	pct_gt_mn	pct_lt_mn_species	pct_gt_mn_species
1	bream	38%	62%	53%	47%
2	pike	53%	47%	65%	35%
3	perch	68%	32%	66%	34%
4	whitefish	50%	50%	50%	50%

SASHELP.FISH × | Fish_2_more_macros.sas ×

CODE | LOG | RESULTS

```
* Get the mean weight over the four fish combined;

proc sql  noprint;
    select mean(weight) into :mn_wgt
    from sashelp.fish
    where species in ('Bream', 'Pike', 'Perch', 'Whitefish');
quit;


* Perhaps taking macros too far, so that code becomes less readable?;

%macro sumit(var);
    sum(&var) as cnt_&var
%mend sumit;

%macro pctit(var);
    cnt_&var / n_species as pct_&var
%mend pctit;


* Make a macro to look at one species at a time;

%macro pct_fish(species);
```

SASHELP.FISH × | Fish_2_more_macros.sas ×

CODE | LOG | RESULTS

```
* Compare bream with bream mean weight and overall mean;

data &species._compare;
    set &species;
    if (weight < &mn_wgt)            then lt_mn = 1;
    if (weight > &mn_wgt)            then gt_mn = 1;
    if (weight < &&mn_wgt_&species) then lt_mn_species = 1;
    if (weight > &&mn_wgt_&species) then gt_mn_species = 1;
run;


* Count them;

proc sql;
    create table &species._stats as
    select count(*)             as n_species,
           %sumit(lt_mn),
           %sumit(gt_mn),
           %sumit(lt_mn_species),
           %sumit(gt_mn_species)
    from &species._compare;
quit;
```

SASHELP.FISH × | Fish_2_more_macros.sas ×

CODE | LOG | RESULTS | OUTPUT DATA

Table: WORK.FISH_FINAL | View: Column names | Filter: (none)

Total rows: 4 Total columns: 5 Rows 1-4

	species	pct_lt_mn	pct_gt_mn	pct_lt_mn_species	pct_gt_mn_species
1	bream	38%	62%	53%	47%
2	pike	53%	47%	65%	35%
3	perch	68%	32%	66%	34%
4	whitefish	50%	50%	50%	50%

SASHELP.FISH × | Fish_3.sas ×

CODE | LOG | RESULTS

```
* Get the median weight over the four fish combined;

proc sql  noprint;
    select mean(weight) into :mn_wgt
    from sashelp.fish
    where species in ('Bream', 'Pike', 'Perch', 'Whitefish');
quit;


* Find mean for each species;

proc sql;
    create table species_means as
    select species,
           mean(weight) as mn_wgt_species
    from sashelp.fish
    where species in ('Bream', 'Pike', 'Perch', 'Whitefish')
    group by species;
quit;
```

SASHELP.FISH × | Fish_3.sas ×

CODE | LOG | RESULTS

```
22 * Compare each with mean(s);
23
24 proc sql;
25     create table species_cnts as
26     select a.species,
27            n(weight)                                                    as n_species,
28            sum(case when a.weight < &mn_wgt then 1 else 0 end)         as cnt_lt_mn,
29            sum(case when a.weight > &mn_wgt then 1 else 0 end)         as cnt_gt_mn,
30            sum(case when a.weight < b.mn_wgt_species then 1 else 0 end)  as cnt_lt_mn_sp
31            sum(case when a.weight > b.mn_wgt_species then 1 else 0 end)  as cnt_gt_mn_sp
32     from sashelp.fish (where=(species in ('Bream', 'Pike', 'Perch', 'Whitefish') and
33                               weight is not null))
34                                 a,
35          species_means          b
36     where a.species = b.species
37     group by a.species;
38 quit;
39
40
```

SASHELP.FISH × | Fish_3.sas ×

CODE | LOG | RESULTS

```
30            sum(case when a.weight < b.mn_wgt_species then 1 else 0 end)  as cnt_lt_mr
31            sum(case when a.weight > b.mn_wgt_species then 1 else 0 end)  as cnt_gt_mr
32     from sashelp.fish (where=(species in ('Bream', 'Pike', 'Perch', 'Whitefish') and
33                               weight is not null))
34                                 a,
35          species_means          b
36     where a.species = b.species
37     group by a.species;
38 quit;
39
40
41 * Calculate percentages;
42
43 proc sql;
44     create table species_pct as
45     select species,
46            cnt_lt_mn / n_species          as pct_lt_mn          format=percent7.,
47            cnt_gt_mn / n_species          as pct_gt_mn          format=percent7.,
48            cnt_lt_mn_species / n_species  as pct_lt_mn_species  format=percent7.,
49            cnt_gt_mn_species / n_species  as pct_gt_mn_species  format=percent7.
50     from species_cnts
51     order by species;
52 quit;
```

SASHELP.FISH | Fish_3.sas

CODE | LOG | RESULTS | OUTPUT DATA

Table: WORK.SPECIES_PCT | View: Column names | Filter: (none)

Total rows: 4 Total columns: 5 Rows 1-4

	Species	pct_lt_mn	pct_gt_mn	pct_lt_mn_species	pct_gt_mn_species
1	Bream	38%	62%	53%	47%
2	Perch	68%	32%	66%	34%
3	Pike	53%	47%	65%	35%
4	Whitefish	50%	50%	50%	50%