A Macro-Driven Approach for Systematically Testing Variables Against a Base Regression Model
Jennifer Alessandro & Bryan Harmon
Management Science Associates, Inc., Pittsburgh, PA

ABSTRACT
Have you ever had a base regression model and wanted to test what happens when you add a variable such as television advertising to that base model? What if you have 10 different television variables, each with four different possible retention rates? You may even want to try some of the advertising variables in combination with one another. In the end, only one or a small subset of these variables should be added to the base model. All of them need to be tested, but testing them in the model all at once will cause collinearity issues. This paper will provide a macro you can use to test a large set of variables quickly and efficiently. The output will provide a summary showing how each candidate variable worked in the base model and which combinations of candidate variables worked well together. It will also go into detail on how to easily generalize the macro, so it can be used for many projects.

INTRODUCTION
Modeling requires a great deal of work. As much as the modeler may want to sit back while a program iterates through variables to find a final model that scenario is just not possible. Each variable has to be tested for significance, correlation with other model variables, and common sense. For example, should an increase in disposable income positively or negatively impact my sales? Although you cannot have the work done for you, SAS® macros can help ease the modeler’s burden.

Since a macro can be tailored to whatever you need, it can be great for allowing you to still have control over your model while unloading some of the grunt work. Sometimes you just need something simple that would test many different variables in a base model and display the results. Variable patterns must be identified and decisions made for which could or should be added into the base model. Results sometimes need to be reviewed and shared across teams or with a client. This paper begins by demonstrating a simple macro that calls the regression model each time a new variable or set of variables need to be tested. The approach is then tailored into something more sophisticated that help narrow down the results and possibly determine a new base model candidate.

BASE EXAMPLE
In the code, the REGRESSION procedure is called multiple times for each new model candidate being explored. One variable can be added at a time, or multiple variables within the base model can be tested. Several statistics, including the coefficient, t-value, and variance inflation, are being output using the Output Delivery System. These statistics can help you make a decision as to whether this is a good candidate for a new base model.

The code in Figure 1.0 is the general set up before the macro is run to call the regression. It first sets a macro variable called, base_vars, containing a list of the variables in the base model. The second macro variable, new_vars, is a list of the new variables you want to try. Here each model to try is separated by a tilde (~). The very last model that will be run has five additional variables to test at the same time.

Here the PROC REG runs just the base model, and an output dataset, base_model, containing the variables with their estimates is created. See Figure 1.1 for a partial list of the output displaying the parameter estimates. In the next section, the estimates for each variable from the base model will be compared to the estimates from the new model candidates to determine if there are any coefficient changes.
Figure 1.0
The Variable Lists & Base Model
/*listing the base variables*/
%let base_vars = CCI competitive_sales disposable_income sample_units seasonality store_count;

/*listing the variables to try in the base, models are separated by a ~*/
%let new_vars = Media_OOH_Spend_prodA ~ Media_OOH_Spend_prodB ~ Media_Print_Spend_prodA ~ Media_Print_Spend_prodB ~ Media_TV_GRPs_prodA ~ Media_TV_GRPs_prodB ~ Media_TV_GRPs_prodC ~ Media_TV_GRPs_prodD ~ Media_TV_GRPs_prodE ~ Media_TV_GRPs_prodF;

/*using the REG procedure and ODS to create a an output data set with the base model estimates called base_model*/
ods output "Parameter Estimates" = base_model(keep= Variable Estimate rename=(Estimate=Base_Estimate));

/*the Base_Model: notation assigns a name to the model*/
proc reg data = nesug.model_data;
Base_Model: model sales = &base_vars;
run;
quit;

Figure 1.1
Parameter Estimates for Base Model

| Variable         | DF | Parameter Estimate | Standard Error | t Value | Pr > |t| |
|------------------|----|--------------------|----------------|---------|------|---|
| Intercept        | 1  | 3607341            | 330596         | 10.91   | <.0001 |
| CCI              | 1  | -17616             | 3277.27740     | -5.38   | <.0001 |
| competitive_sales| 1  | -1040789           | 175182         | -5.94   | <.0001 |
| disposable_income| 1  | 40750              | 6164.94056     | 6.61    | <.0001 |
| sample_units     | 1  | 1.20009            | 0.09472        | 12.67   | <.0001 |
| seasonality      | 1  | -3426204           | 277392         | -12.35  | <.0001 |
| store_count      | 1  | 2749.32958         | 15.41748       | 178.33  | <.0001 |

Figure 1.2 contains the macro that will run the new models when called. Datasets, all with a prefix of parest, containing information about the variables are being output. The do while loop will continue to run the regression until the last set of variables is pulled from the new_vars macro variable, so that parameter estimates are created for each new model being tried.

The datasets can all be set together using a dataset list, available in SAS® 9.2. If you are working in an earlier version of SAS®, code can be added within the macro to create the file. The end result is one dataset, parest_all, that has the model details for all 22 models that are run.

Figure 1.3 shows partial output of this dataset for the first four models. Output can be compared to determine which models, if any, show promise as the next base model or if patterns become noticeable. For example, the
variable, Media_OOH_Spend_prodA, added in Model 1, is statistically significant. However, disposable income in this model is no longer statistically significant and actually flips signs. Perhaps this might be something worth looking into further. Maybe the two variables are correlated with one another and some sort of transformation can be made to one or both of the variables, so that they can be put in the model together.

Other useful patterns that may emerge are trends in significance. Such as, the statistical significance of a retained variable will usually get better or worse as the retention rate increases. If each version of the retention rate is printed, a pattern might emerge that you did not know existed. Perhaps adding in a higher/lower retention rate might get the variable to come with a significant t-stat into the model. The base code in Figures 1.0 and 1.2 is a simple and quick way to run many models. The results printed out so that they could be easily viewed. In the next section, the list of variables with promise will be shrunk down to a more manageable size.

---

Fig. 1.2
The Regression Macro

```sas
%macro multiple_models;
/*setting macro variables model (used as a model counter) and add_var (uses the scan function to pull the set of variables being tried)*/
   %let model = 1;
   %let add_var = %scan(&new_vars,1,~);
/*do while loop will continue to call the regression and output estimate datasets as long as add_var is not null*/
   %do %while (&add_var ne);
      %if &model lt 10 %then %do; %let name = Model_00&model; %end;
      %else if &model lt 100 %then %do; %let name = Model_0&model; %end;
      %else %do; %let name = Model_&model; %end;
      ods output "Parameter Estimates" = parest&name(keep= Model Variable DF Estimate StdErr tValue Probt StandardizedEst VarianceInflation);
      proc reg data = nesug.model_data
         outest=coeff&name(keep= MODEL_ AIC_ BIC_);
         &name.: model sales = &base_var &add_var
         / vif stb rsquare dw pcorr2 AIC BIC;
         output out=preds;
      run;
      %let model = %eval(&model + 1);
      %let add_var = %scan(&new_vars,&model,~);
   %end;
%mend;
%multiple_models;
/*sets all the parest datasets together using the colon data set list function*/
data parest_all;
   length variable $32.;
   set parest:;
run;
proc print data = parest_all noobs width=min;
   var variable Model Estimate StdErr tValue Probt;
run;
```
Perhaps your goal is to only see the combination of variables that are working. This will help you determine which of the 22 models could be the next base model. Additional code can be added that can reduce the amount of output.

This code joins the base model with the candidate models using the SQL procedure and then allows for comparisons to be made. It checks for significance using a 90% confidence level and looks at whether the sign of a variable’s coefficient changed between the candidate model and the base model. Since the expected coefficient of media support is positive, it also checks that the variables being tested have the correct sign.

The PROC SQL is then used to determine which models are “bad”. Only significant variables with the correct sign and those that did not impact the base model negatively are kept in possible_models, the table created in the PROC SQL. Figure 2.1 shows the five remaining candidates that met all the criteria, reducing the number of potential model candidates by over 75%.
The code which is already versatile can be further generalized to be used across multiple projects and modelers. Setting a macro variable for the dataset, dependent variable, and/or the output files can help make the code easier to update. Additional tests on the value of a model candidate can also be added to the code. For example, perhaps the model has been showing correlations and it is important that the variance inflation factor be considered as well.
A model meeting the client’s specific needs is one that includes as many of the marketing activities and programs they need to explain. This need will generally override a model deemed the best model statistically. A client must understand the model’s implications and have audience buy-in because they ultimately have to defend the results to their superiors and different departments. In many cases, they will have an opinion on which variable is more important or necessary to include in a model, regardless of whether or not another variable is a better choice.

As statisticians, however, we would prefer to choose the best model whenever possible. There are several techniques available in SAS®, so that direct comparisons can be made across candidates. One technique is demonstrated in Figure 3.0. It prints out information criteria associated with a model, which can be used to determine the best model. In Figure 2.0, the estimates for each model are output to a dataset with a prefix of coeff in the PROC REG macro code. Two measures, Akaike’s Information Criteria (AIC) and Sawa’s Bayesian Information Criteria (BIC) are kept. Now in Figure 3.0, similar to the parameter estimates, they are set together using a dataset list. The measures are merged on to the possible_models dataset. The model with the smallest AIC or BIC is deemed the best model.

If you have the flexibility to make variable choices based on a best model approach, then this is a great tool to narrow down the list to a single model. Figure 3.1 is the output from the PRINT procedure for the dataset score_aic_bic. Model 2, has the lowest AIC and BIC score, identifying it as the best candidate model across either measure.

CONCLUSION
Creating the perfect model can be a very iterative process. However, using a few tools from SAS® can streamline it significantly.

REFERENCES:
For more information on the data set lists and other procedures mentioned in this paper, refer to SAS Institute Inc.

ACKNOWLEDGMENTS
SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the authors at:

Jennifer Alessandro & Bryan Harmon
Management Science Associates, Inc.
6565 Penn Avenue
Pittsburgh, PA 15206
Work Phone: 412-362-2000
Web: www.msa.com

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *