Paper 1822-2018

Penalized Variable Selection and Quantile Regression in SAS®: An Overview

Russ Lavery, contractor, Bryn Mawr, PA

Great thanks to: Peter Flom, Peter Flom Consulting, New York, New York

ABSTRACT

This friendly, and picture oriented, paper is about some new PROCs for modeling using penalized variable selection and some PROCS for building models that are a richer description of your data than you can get from OLS. The four PROCS we will cover are: Reg, GLMSelect, QuantReg and QuantSelect. The paper explains theory and gives examples of SAS code and output for four PROCS. The new penalized methods that are discussed in this paper help, only help, find a parsimonious model. They help us by using algorithms that are more stable, continuous and computationally efficient than stepwise methods.

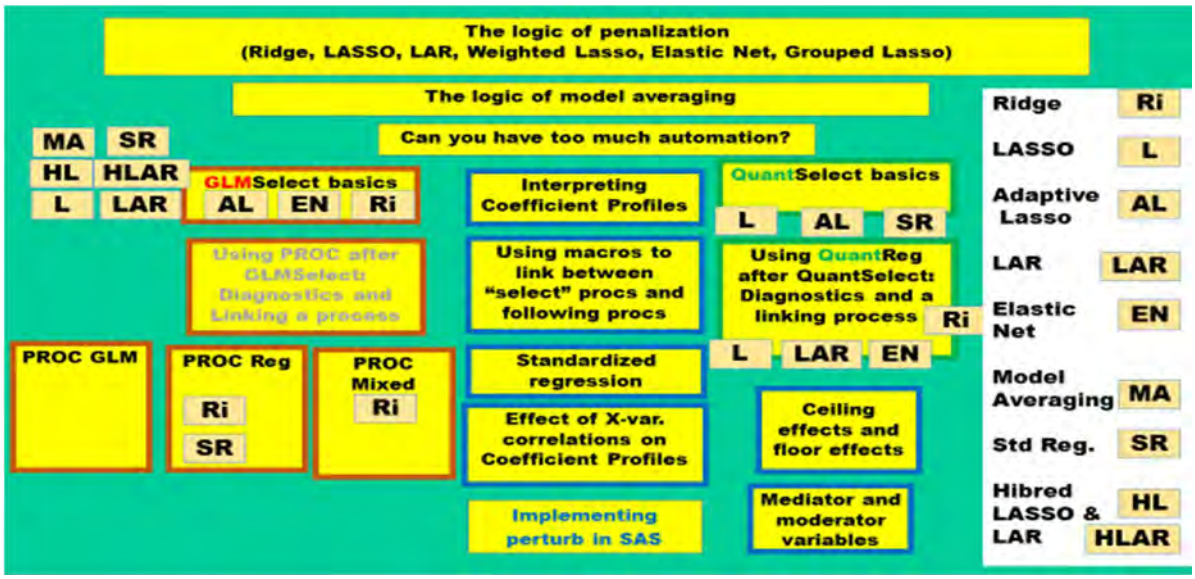

Figure 1

INTRODUCTION

A historical goal of regression analysis in the last century has been to determine, from one data set, THE ONE model that best describes the relationship between the dependent and independent variables for new data. This process creates ONE line (predicting the conditional mean response). Much of this work has involved the use of maximum likelihood estimation (MLE). Recent advances in computer power allow us to expand this goal in two ways: First: Methods of penalized regression allow us to solve some of the problems of MLE. Second, quantile regression lets us create multiple, not one, conditional estimates of Y.

Maximum likelihood estimation has three major problems:

Collinearity: When the independent variables are collinear, the β values are unstable.

Overfitting: When the sample size (N) is not much larger than the number of variables (p), resulting models will be over fit and will not generalize well to new data.

Parsimony: Einstein said that the goal should be to make everything "as simple as possible, but not simpler". Much of model building has focused on the attaining the ideal degree of parsimony. However, the older methods (e.g. bivariate screening, stepwise selection, forward selection) are flawed.

Statistics given in the output from regression (p values, parameter estimates, R^2 and so on) were developed for testing one specific model and are not justified when used multiple times as they are in stepwise.

This paper started out as a much smaller topic and it grew and grew. It started out as a talk about only PROC QuantReg. QuantReg works fine on its own if there is a single model that we wish to test and it is specified a priori. However, if we want to use model selection methods with a QuantReg, we need to understand PROC QuantSelect.

But PROC QuantSelect requires an understanding of penalized regression and the techniques of: Ridge, LASSO, LAR, Weighted LASSO, Elastic Net and Grouped LASSO. These methods are easier to understand in the more familiar environment of ordinary regression, which led us to PROC GLMSelect, which is similar to PROC QuantSelect. We suggest that a study all the four PROCS, and theory, is better than studying one PROC. PROC GLMSelect can also be used to select variables for PROC GLM and PROC Mixed but we will not show examples of these in this paper.

Figure 1 shows the products we examined in creating this paper and some of the advanced, machine learning, options one can request from the PROCs.

BACKGROUND

Model selection is difficult and is an important topic in statistics. For a long time, stepwise methods have been known to have problems but we used/taught them because we lacked alternatives for the automatic creation of parsimonious models.

Model selection, establishing the relationship between Y and elements of a set of X variables, is difficult for several reasons. A nonlinear relationship between Y and several X is might not show up in a marginal plot of Y versus X – even a partial regression plot. Scatterplots show the marginal relationship between Y and the X variables and cannot show anything about relationships between a Y and several Xs (although coplots and scatterplot matrices can do some of this). A strong relationship between Y and one X may be outperformed by a relationship between Y and a group of X variables that are, individually, poor predictors. A lack of the marginal relationship between Y and an X variable does not mean that it the X variable is not useful in the model. The X variable might be needed as a moderator or mediator in a model that includes other X variables. In addition, theory often suggests some variables “should” be related to the dependent variable. In these cases, finding a weak relationship is interesting.

Most unsettlingly, nearly all the results of regression (F and chi-square tests, parameter estimates, p values and so on), assume that the model is specified beforehand and that is not the case with stepwise methods. Results are incorrect when stepwise or similar methods are used.

Stepwise algorithms are fast because they are greedy. They make the best choice at each step, regardless of the effects of future X variables and this is often not the way to find “TRUTH”.

All-Subsets regression examines all subsets of the X variables for each particular number of X variables (best 1 X-variable model, best 2 X-variable model etc.). An advantage of all subsets is that the best set of two predictors need not include the X variable that was the best one predictor model. However the biases of all-subsets β values are much, much greater than in stepwise. All-Subsets creates combinations of X variables, so, even with only a relatively small model with 10 independent variables, there are over 1000 models to examine (and adjust for). The chance of an α error is large.

The new penalized methods that are discussed in this paper help, only help, find a parsimonious model. They help us by using algorithms that are more stable, continuous and computationally efficient than stepwise methods. However; thought is still required for three reasons. Firstly; output is still complex. Secondly; some penalized penalties are more appropriate in a particular situation than others. Thirdly, any automatic method will be inadequate because (as noted above) there are considerations other than simple model fit to consider. No automatic method is a substitute for substantive knowledge and thought.

Figure 2 starts a story. Hopefully, that story explains some of the logic behind, and steps in, Ridge regression. We know that OLS can, if the number of variables is large or if the X variables are collinear, produce highly unstable β estimates and this slide starts an illustration of that problem.

In the white box on the left, a data step creates two variables that are highly correlated and a Y variable that is created by the formula Y = 4 + 1X_1 + 1X_2 + error. A PROC GPlot shows the extreme correlation between X_1 and X_2. Finally a PROC Reg builds a model using these two highly correlated X variables. (It should be noted that correlation and collinearity are not the same and that there can be collinearity among a set of variables none of which are have a high individual correlation with Y. However, in our example case, with only 2 IVs, a high degree of correlation does imply a high degree of collinearity. The best method of detecting collinearity is condition indexes, available in PROC REG.

The results of the model are shown on the right of Figure 2. In the top box you can see that the model predicts very well. In the bottom box you can see that neither the X_1 nor the X_2 variable is significant. This is a very common occurrence in regression when X variables are correlated. Notice, in the formula that creates the variable true_Y, that the βs for X_1 and X_2 sum to two (1 + 1 =2). In the parameter estimates box on the right-hand side the βs also sum approximately to two (24.78 + -22.88), though the values are very different from the β in the formula for true_Y.

Pause for a minute and consider this. If the X'X matrix has problems, SAS uses a generalized inverse and the βs become unstable. If X_1 and X_2 are as highly correlated as the plot shows, then we can say (practically) that X_1 *IS* X_2. We could re-write the formula as Y= 2*X_1 + 0 X_2. We could re-write the formula as Y= 0X_1 + 2*X_2.

INTRODUCTION TO THE PROBLEM:

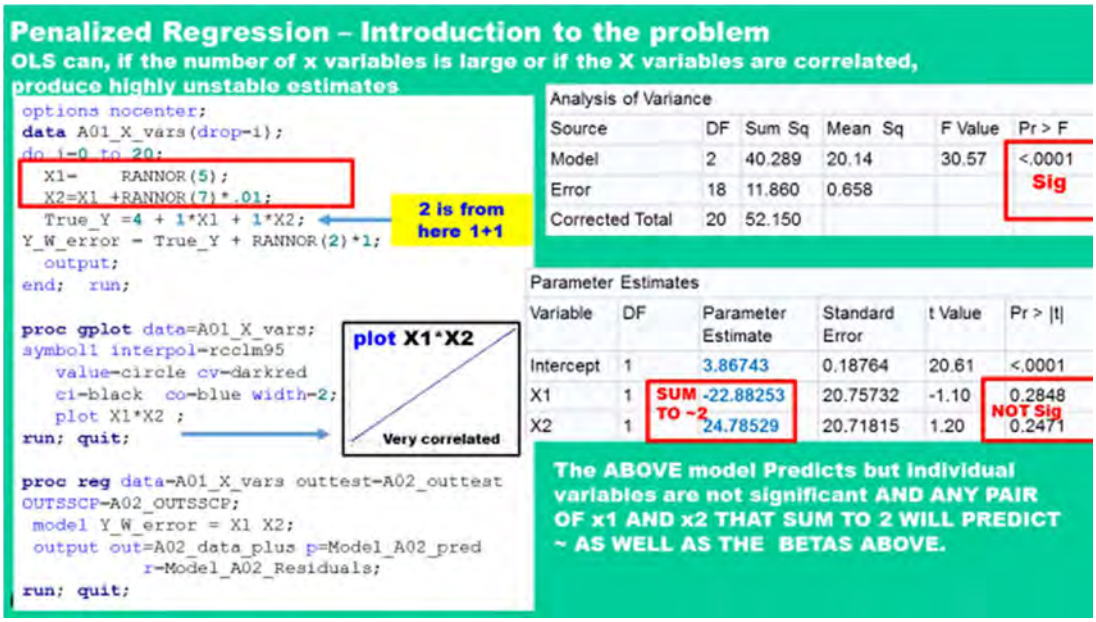

Figure 2

The code in Figure 3 manually calculates sum squared error for many different "models". The characteristics of all the models is that the values of β_1+ β_2 sum to two (-10 + 12 =2 , 4 -2 =2, etc.). Since X_1 and X_2 are so correlated, X_1 and X_2 are really the "same" variable. So, any combination of X_1 and X_2 where the βs add up to the number two should give the about the same answer (we did add random error). The squared residual number in red (11.86) is from the SAS PROC Reg.

The calculations in Figure 3 (differing only because of rounding) are approximately the same number. A key conclusion is this: in the presence of very high correlations, many combinations of β values will give the same answer.

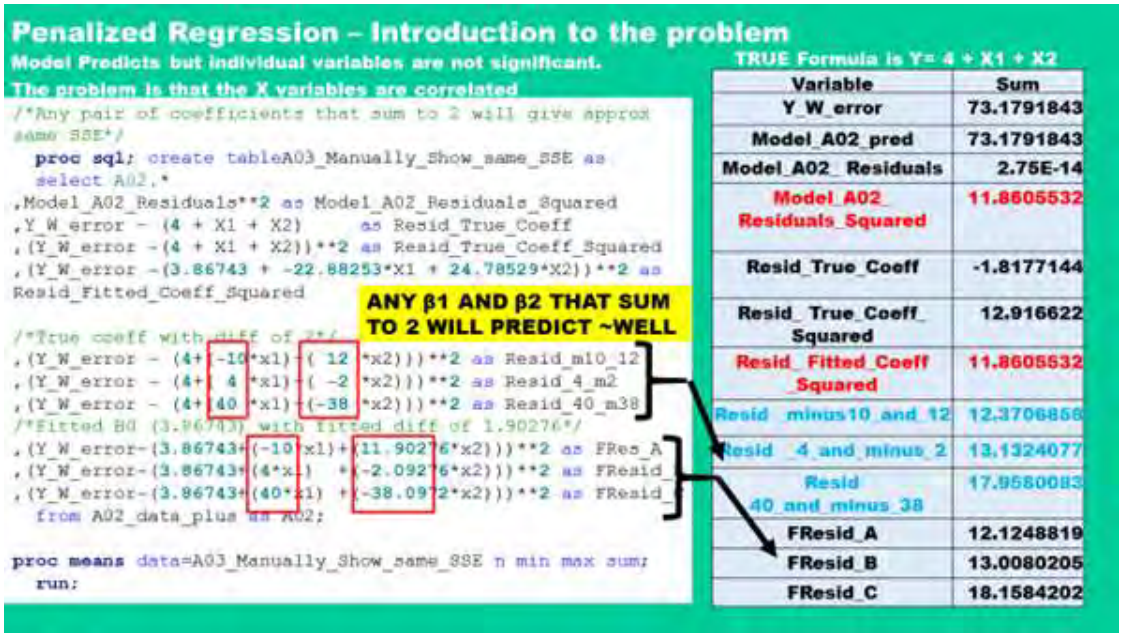

Figure 3

Since we get the same model result from many different models we have to ask “how do we choose a model (a set of beta values)” from the many models (sets of βs that predict equally well).

The logic for answering the question is simple. Consider that if these two models predict about as well:

$$Y = 4 + 1 X_1 + 1 X_2$$

and

$$Y = 4 + 101 X_1 - 99 X_2$$

Occam’s razor can be used to select a model. Occam’s razor suggests that we pick a simpler mode, and in this case, Occam’s razor suggests that we should pick models with small β values. Unless there is evidence to the contrary, it is more likely that $Y = 4 + 1 X_1 + 1 X_2$ than $Y = 4 + 101 X_1 - 99 X_2$. Applying Occam’s razor, models with small β values, and models with few βs and models with lower exponents on the Xs, are considered simpler and more likely.

We now have a rule for picking models when X variables are correlated and β values are unstable. When multicollinearity is present, we prefer small βs. This is the underlying logical reason for Ridge regression. In the presence of multicollinearity, we want to shrink β values and Ridge will be our first method.

As a small digression, it does not make sense to shrink the intercept towards zero and none of the stat packages will shrink β_0. Additionally, because X variables often have greatly different units of measure, all of the packages will standardize the X variables so that shrinkage is applied equally regardless of the natural units of the X variables. Packages also un-standardize for user convenience.

A FRIENDLY (MADE-UP) STORY ABOUT RIDGE REGRESSION:

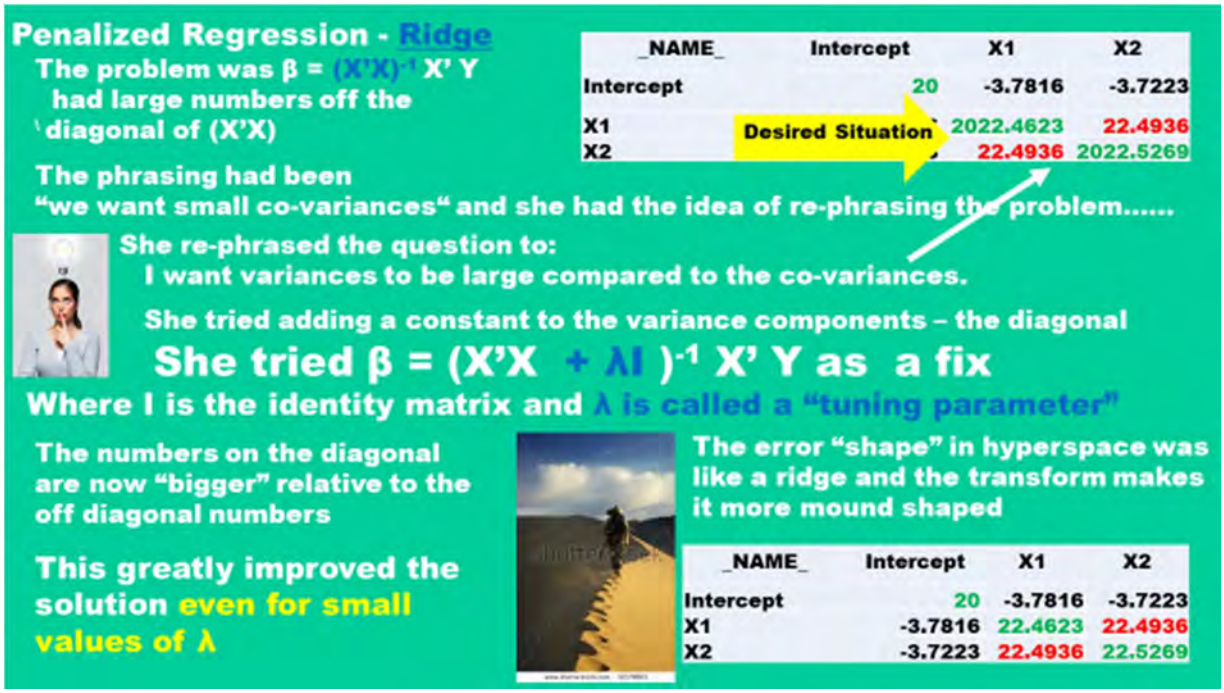

Figure 4

Once upon a time a young statistician was confronted with the problems of multicollinearity as she was building a model. She knew the regression formula was $\beta = (X'X)^{-1} X' Y$ and that (X'X) was the sums of squares and cross products matrix. "Good" regression problems had (X'X) matrices where the off diagonal values were small. She knew that when the X variables were highly correlated the (X'X) matrix had off diagonal values that were large compared to the values on the diagonals.

The common way of describing the multicollinearity problem was "we want small covariates" in (X'X). She had the idea of rephrasing the problem. She rephrased the problem to "I want variances that are large compared to the covariates".

She tried using $(X'X + \lambda I)$ to add to the diagonal values. Adding λI makes the diagonal values larger relative to the off-diagonal values. This greatly improved the solution, even for small values of λ. With her technique the numbers on the diagonals were larger than the numbers on the off-diagonals, though this is not the ridge for which the technique is named. The inventor thought that his technique, as it moved up it's gradient, was moving up a ridge. There is a small picture of a man walking on a ridge in Figure 4.

It is now time to make a useful, though slightly tortured, analogy concerning inverses.

Think of two values 5 and 10. 10 is bigger than 5. The inverse of 10 is 1/10 and is smaller than the inverse of 5 (1/5). We conclude that as numbers get bigger, their inverses get smaller and note that $(X'X)^{-1}$ is an inverse.

I say I will torture the analogy because the inverse of a matrix is not a number but another matrix. If λ is greater than zero $(X'X + \lambda I)$ is "larger than" (X'X) and so $(X'X + \lambda I)^{-1}$ is smaller than $(X'X)^{-1}$. As λ gets larger $(X'X + \lambda I)^{-1}$ gets "smaller" because it is an inverse (sorry about that).

Since the $\beta = (X'X + \lambda I)^{-1} X'Y$, we are multiplying an inverse by a fixed amount (the X'Y). As λ gets large the βs get small. We have found a way to shrink βs towards zero and to implement Occam's razor in selecting variables.

But, nothing is free and there is a complication for this technique. The variances of the βs can be calculated using: var(β-hat) = σ^2 (X'X) $^{-1}$ …. where σ^2 is unknown but can be calculated by RSS(β-hat) / (N-P) RSS(β-hat) . P is the number of X variables and N is the number of obs.

As λ gets larger the decreasing inverse causes the variance of the βs to decrease – but only for a while. There is a hidden upward pressure on the variance that is exerted through the σ^2 term. As λ gets large it will affect the formula for the βs by making the βs → 0. As βs → 0 the predicted values of Y become less accurate and RSS goes up. As the βs move away from their true value, RSS gets larger because the model no longer predicts as well. Eventually the effect of poor predicting overwhelms the effect of λ on the inverting and σ^2 (X'X) $^{-1}$ increases. The link between the formulas produces a characteristic U-shaped curve as λ increases.

Let us do two thought experiments about the formula (X'X + λI) $^{-1}$. As λ gets close to zero our λI term disappears and we are left with OLS regression. As λ gets large it makes the diagonal of (X'X + λI) $^{-1}$ matrix get large, the inverse small. In the general case, the βs go to zero because (X'X + λI) is large and the inverse term in β = (X'X + λI) $^{-1}$ X'Y is small. However, it is possible for individual β to increase or decrease or even change sign if λ increases but is less than infinity.

There is another way to think about the effect of increasing λ. Points, from correlated X variables, can be thought of as forming a "cylinder" in the hyperspace defined by the X variables. Points from independent X variables form a swarm of bees in the hyperspace defined by the X variables. The fitted hyperplane, from your model, rests on the observed points in X-space.

If the points form a narrow cylinder then small changes in the observations can allow the plane to tilt. Think of trying to balance a dinner plate on a rolling pin. However, if the points are spread out in hyperspace then the hyperplane rest on points that are spread widely apart and the hyperplane is stable. Think of the data swarm as supporting the hyperplane in a manner similar to a plate resting on a table. The λI term can be thought of as spreading out the points in X-space and providing better support for the hyperplane.

In summary; Ridge implements Occam's razor when it shrinks βs.

ANOTHER VIEW OF RIDGE

Mathematically, β = (X'X + λI) $^{-1}$ X' Y can be converted to
$SSE_\lambda\ (\beta) = \sum_{i=1}^{n}(Y_i - \sum_{j=1}^{p-1} X_{ij}\ \beta_j)^2 + \boldsymbol{\lambda \sum_{j=1}^{p-1}(\beta_j)^2}$ which is like the OLS formula with a penalty (blue term) for having large (or for having many) βs. The second power penalty is easy to optimize using calculus. The Ridge penalty is often called an L2 penalty because of the second power in the exponent of the penalty term. λ is often called the shrinkage parameter. Large β values, or models with more β values, are interpreted as more complex models and, according to Occam, less likely to occur in nature. Models with large βs, and models many βs, are both penalized by the Ridge technique.

LASSO (Least Absolute Values Shrinkage and Selection)

The penalty for LASSO is very similar to the penalty formula for Ridge. The LASSO penalty term is the absolute value of the βs raised to the first power (that is, unchanged). The LASSO penalty is often called an L1 penalty because of the first power in the penalty term.

$$SSE_\lambda\ (\beta) = \sum_{i=1}^{n}(Y_i - \sum_{j=1}^{p-1} X_{ij}\ \beta_j)^2 + \boldsymbol{\lambda \sum_{j=1}^{p-1}|\beta_j|^1}$$

Since LASSO has an absolute value penalty, it cannot be differentiated. LASSO optimization is done by operations research techniques (simplex, etc.) and the original algorithms were fairly slow.

Comparing the shrinkage of Ridge to LASSO you can see a few important differences. β values in a Ridge never reach exactly zero (until λ= infinity) while LASSO can shrink β values to exactly zero..

Some people like LASSO because the β values go to exactly zero and are, in LASSO, removed from the model. These people claim the fact that LASSO sets β values to exactly zero is a positive characteristic for LASSO and these people use LASSO as a replacement for stepwise.

Some people like Ridge regression because, in Ridge, β values do NOT go to zero. These people say that there is an over-emphasis on the P value in much of statistics. People in this camp feel that the true purpose of statistics is to estimate an "effect size" and they like Ridge regression because Ridge will leave relatively unimportant variables in the model with a small β values (small effect sizes).

Figure 5

Ridge shrinks the coefficients for large β values, the important variables, more than it does small β values. This is correct when the problem is collinearity, since one problem with collinearity is that some parameter estimates are much too large. However, when the problem is overfitting, this is not the case. Thus, LASSO is designed to deal with overfitting, (and does not deal that well with collinearity) and Ridge is designed to deal with collinearity (and does not solve problems of overfitting).

Figure 5 illustrates the difference between β plots for Ridge and LASSO and shows how easy it is to code these two techniques in SAS. As λ increases the β values change – in both Ridge and LASSO. In a Ridge the values change smoothly and will asymptotically approach zero. In Ridge, Unstable β values can cross zero and remain in the model.

LASSO plots are really a series of points connected by lines – just to make the relationship among the points is easier to see. You only interpret LASSO plots at the points where an X variable enters or leaves the model. As you can see the LASSO plot for age goes exactly to zero. In LASSO, when a β value is zero, the variable has been removed from the model.

ADAPTIVE LASSO

Ridge regression shrinks β values for important variables more than it shrinks βs for unimportant variables. Lasso shrinks all β values by the same amount. The idea of Adaptive LASSO is to shrink the βs for important variables less than the β values for unimportant variables – keeping important variables in the model "longer".

Adaptive LASSO is a two-step process because it needs a starting point for the βs. An OLS regression is often the starting point for β values for Adaptive LASSO (and may therefore be incorrect when there is collinearity). Shrinkage values are created based on variable importance and final model selection still takes some human insight and skill.

ELASTIC NET

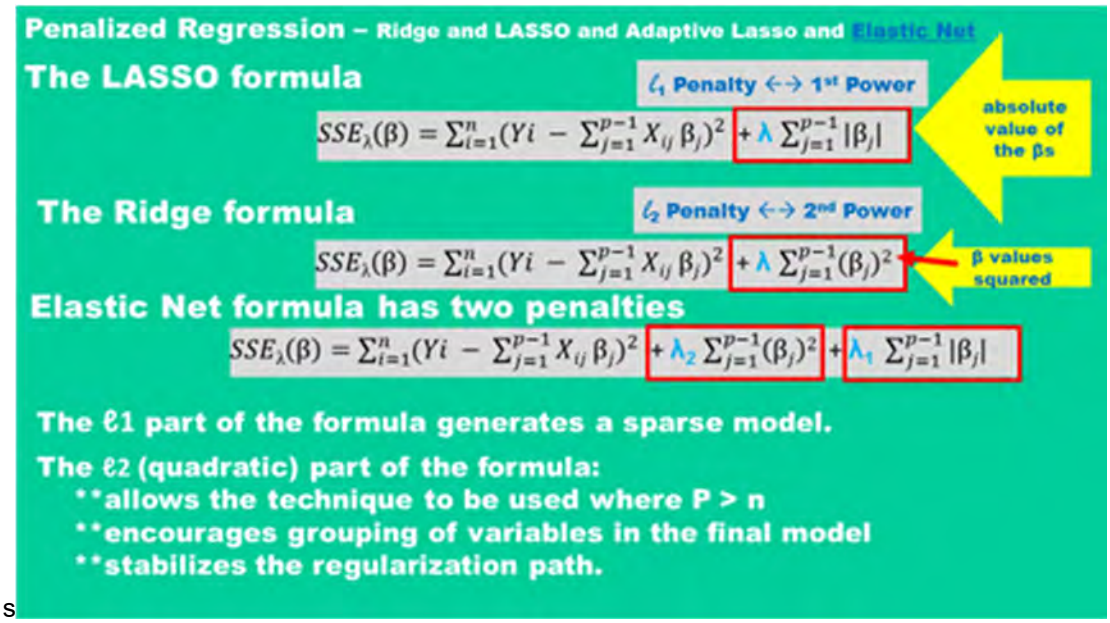

Figure 6

Elastic Net, as you can see in Figure 6, has both of the L1 and L2 penalties.

The L1 part of the formula allows Elastic Net to create a model with few variables – a sparse model. The L2 part of the formula allows Elastic Net to be used when the number of variables is greater than the number of observations. This allows Elastic Net to be used in situations, like biological testing, where there might be thousands of variables and only a few hundred observation.

Elastic Net also overcomes the issue that LASSO has with correlated "groups of X variables". If your data set has groups of X variables that are highly correlated (height, weight, shoe size, hat size, etc.) LASSO will only select one X variable from a group of highly correlated X variables. Elastic Net can bring several of those correlated X variables into the model and keep them there longer. The L2 parameter also stabilizes the path of the β values.

However, Elastic Net has a problem because it has two penalties. Occam's razor says we should look for simple models and Elastic Net has two penalties that it applies to more complicated models. Some modelers think that applying two penalties is too harsh. SAS has an option that lets you request a scaling correction to reduce the "double penalty" but keep many of the positive characteristics of Elastic Net.

FORWARD STAGEWISE

Forward Stagewise is a technique that was important in algorithm development, but is no longer used. Forward Stagewise picks the same initial X variable as forward stepwise, but it only changes the β value by a small amount and then calculates residuals. Foreword Stagewise then picks the X variable with the highest correlation to the residuals (might be the same variable as previous) and makes a small change in the β for that variable – then calculates residuals. Forward Stagewise takes many small steps to reach a solution. Variables can be added and removed at any step.

LAR (LEAST ANGLE REGRESSION)

LAR is an improvement on Forward Stagewise and is so fast that it is used, with modifications, as an algorithm for other techniques. LAR does not take small steps towards a solution but makes a big jump in a β value. LAR has a speed advantage, because variables are only added to, and not removed from, the model.

When you are stopping short of a fully saturated model (assume you have 200 X variables in your data set and you might only want to consider models with a maximum of 50 X variables) LAR has a definite speed advantage. Limiting the number of X variables in a model is often done because business people do not have resources to deal with a 200 variable model. Often, there are no practical advantages to a model containing hundreds of variables over a model containing 50 variables. Details of LAR are shown in Figure 7.

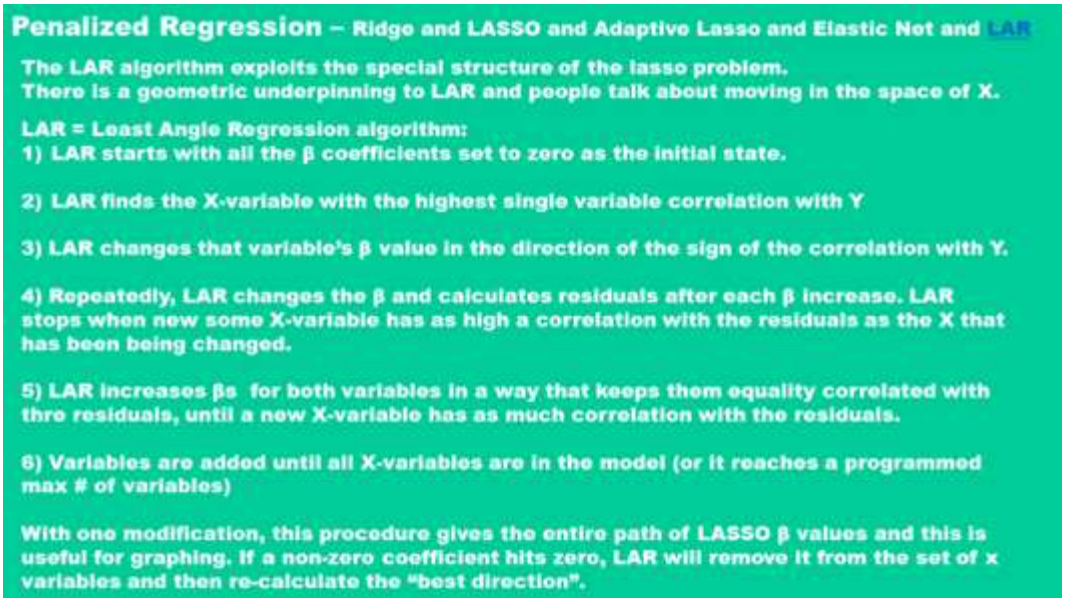

Figure 7

EXTERNAL VALIDITY: GENERALIZING TO NEW DATA: MAKING ONE DATA SET LOOK LIKE MANY DATA SETS

It is common knowledge that a model performs best on the data set that was used to create the model. This means that measures of goodness of fit, like R^2, are biased upwards on the data set that was used to create the model. On new data, a model will likely predict less well. There are techniques that will allow a modeler, with only one data set, to create a model that better generalizes to new data.

If the original data set is large it can be split into two or three parts. The parts are called: train, test, validate, if there are three parts. If there are two parts, the data sets are usually call train and test.

The idea is to create a model using the training data and see how well it would project by predicting a data set containing new observations. This works well if the original data set is large enough to split into two or three parts. If it is not large enough to split, another technique must be used.

A K-fold validation is one technique that is useful for moderately sized data sets. A "fold" is another term for a "split" and each split should be large enough to have a "reasonable chance" of being similar to the population to which we will generalize. Most modelers use a five or 10 split–though there is no great reason for selecting any number. The idea is to take different subsets of the data, build multiple models and then average the models.

Each model will use one Kth of the data as a test and develop the model using K-1 parts of the data. Results of the K different models are averaged (many averaging algorithms exist) to produce a result that is more generalizable than a model built on the whole data set.

Think a bit more about the issue of small N. If N is *moderate* (say N=200 observations) a fivefold split will create a training set with 160 observations and the training set is likely to approximate the next data set encountered. However a K=5 split produces a *test* data set that only has 40 observations. It is likely that this test set will, due to randomness, differ from the data to which you wish to generalize. Forty is likely to be too small to be a good test set.

If N is *small* (N = 50 observations) a fivefold split will send forty observations to the training data set and ten observations to the test data set. It is likely that neither of these is sufficiently large to be representative– even after averaging many different models. There is some theoretical basis for saying a fivefold or a tenfold split produces models that contain excess numbers of X variables.

Another technique is bootstrapping. There seems to have been many different techniques developed under the general heading of bootstrap, but descriptions, when given, seem to point to a very similar algorithm being commonly used.

Most commonly, people talk about the algorithm that follows. Start with a data set with N observations and pick a sample of size N – allowing replacements. The bootstrap sample will have the same number of observations as the original data but, on average, only 63.2% of the observations in the raw data will be in any particular bootstrap sample. Some observations will not be in a sample and some observations will be in a sample multiple times.

The bootstrap sample is not a better representation of the data than your original data and cannot be used as a replacement data source for building one model. What is done is to take many bootstrap samples and to build a model on each of the bootstrap samples and then average the results of the models. You can create many bootstrap samples from one moderately sized data set.

The next step up in complexity is to create models that use either different parts of the data or different methods. These are called ensemble techniques. Common ensemble techniques are K-means, bagging and boosting and the “bucket of models” method. An ensemble is defined as a group of items viewed as a whole rather than individually. Another definition is: all the parts of a thing taken together, so that each part is considered only in relation to the whole.

Ensemble methods are a change from our historical practice of building one best model on a data set.

Figure 8 is a graphic representing the ensemble modeling process. The fact that we are building multiple models is not as much of a problem today, as it had been in the past, because of the prevalence of multiple CPU machines. Now models can be created in parallel. Figure 8 shows a bunch of tables labeled “subsets of observations”. There are a great many methods to take subsets of the master data. Some techniques will take a subset of the observations and some techniques will take a subset of the observations and a subset of the X variables.

Models are built on each of the subsets and then combined. It is important that the models be as independent as possible. “Independent models” mean that the models misclassify different observations. If the models all misclassify the same observations, then averaging will not help model accuracy.

There are many methods used to combine the predictions and some combining algorithms will feed back, into the model building process, information about how accurate that model had been.

If Y is continuous, it is possible to predict Y values using each of the models and then average the Y values. It is also possible to average the β values from the individual models and use the averaged β values to predict Y.

If the Y value is categorical (yes, no or High, Med, Low) you can just allow each model to “vote” and pick the classification that has the greatest number of votes.

As a general principle, the more models you average the better your prediction, especially if the models misclassify different observations.

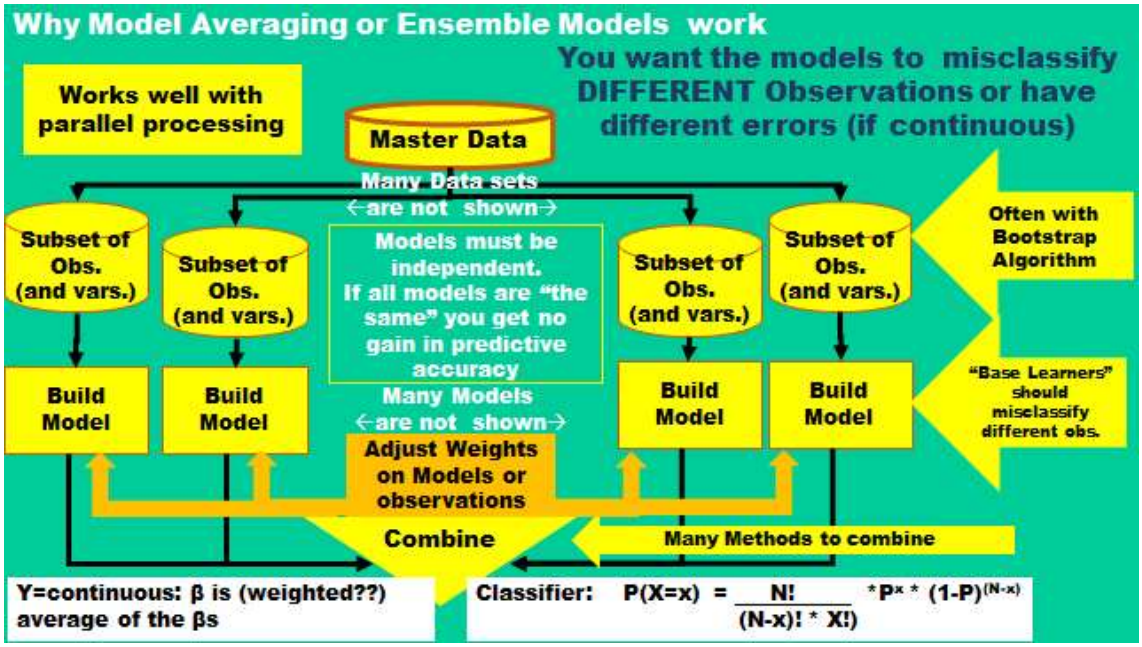

Figure 8

BAGGING (BOOTSTRAP AGGREGATION)

Bagging is a two-step process, invented in 1996, and is a very common way of creating and then combining models. It is a general term describing resampling and averaging of predictions over a collection of bootstrap samples and will reduce prediction variation.

A general description of Bagging (Bootstrap Aggregation) is:

1. Start with a training set S, with N observations, made up of observations ($x_1,y_1, \ldots x_n y_n$)

2. Create NS bootstrap samples

For each of the NS samples, fit a model and get an equation that predicts Y from the X vector

Average the predictions from each of the NS bootstrap models.

There are many averaging methods and they can be complex. Two major philosophical schools are: Bayesian averaging and Frequentist averaging. There are multiple methods within each major philosophy.

QUANTILE REGRESSION

We are now in the second part of the talk. We have finished with the section on model selection theory and will see some of the logic for, and examples of, Quantile regression.

As Figure 9 mentions, OLS minimizes the squared deviation from the fitted line and often uses maximum likelihood to find the β coefficients that will minimize the squared deviation. Quantile regression minimizes the weighted sum of the signs of the deviation from the fitted line and the β are calculated using algorithms from operations research.

The points on the lower left corner of Figure 9, illustrate a 10% quantile calculation. If a point is above the line it has a weight of 1. If a point is below the line it has a weight of -9. In our example we have 10 points at each level of X but this is not a requirement in real life. The red points below the line, are multiplied by their weights (-9) and the black points above the line are multiplied by their rates (+1). An operations research technique is used to set the slope of the line to minimize the sum of the weighted deviation.

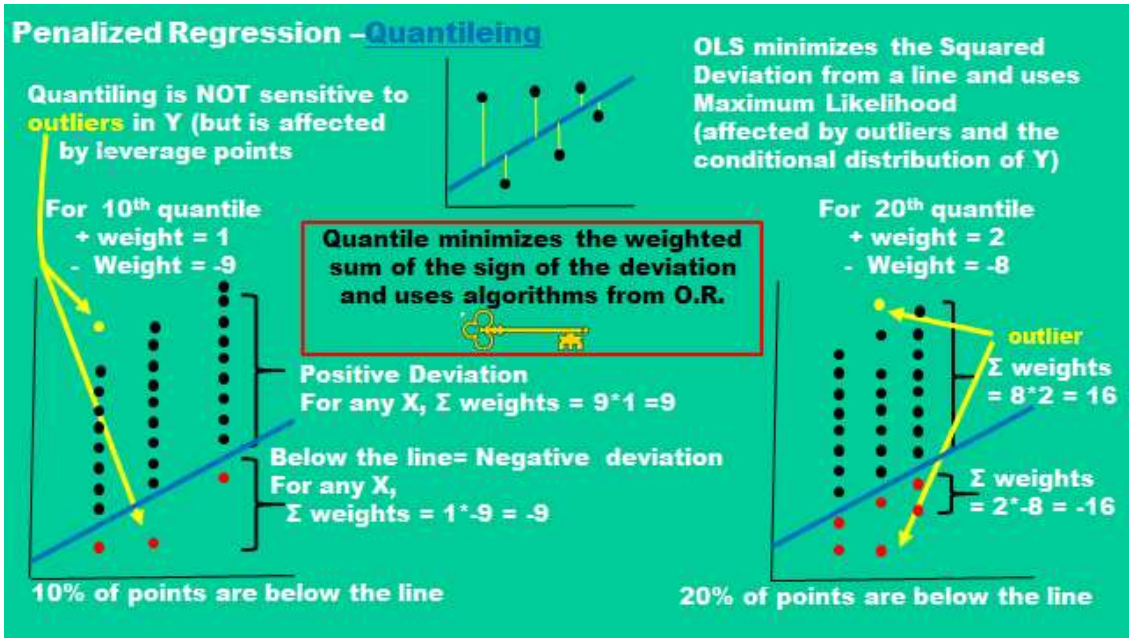

Figure 9

You can see that the summed weights of the black points, for any particular X, sum to positive nine and the sum of the weights for the red points below the line, for any particular X, sum to -9. The blue line, in this picture, minimizes the weighted sums from the line.

On the lower right-hand part of Figure 9 we see a 20% quantile regression. The data set is the same as on the left. The difference between a 10th percentile quartile calculation and a 20th percentile quartile calculation is the weights assigned to observations above and below the fitted line. In the 20th percentile calculation, if a point is below the fitted line, the weight is -8. If a point is above the fitted line, the weight is +2. An algorithm will find β values that will minimize the sum of the weights.

Looking at Figure 9 we see, given enough points, we can use the same data set to calculate lines for as many percentiles as we want. You can also see that Quantile regression is not sensitive to outliers, though it is sensitive to high leverage points.

Figure 10 shows a “compare and contrast” between the correct quantile logic and a common misunderstanding of the quantile logic. Please, focus on the white box in the center. In that box, someone has taken the bottom 20% of the MARGINAL Y values, put them into a data set, and then done a regression on that subset of observations.

A regression line on a subset based on the ***marginal values of Y*** is NOT a Quantile regression. Fitting a regression line to the bottom 20% of the marginal Y values is a mistake. Quantile regression fits a line bottom 20% of the ***conditional*** **values of Y** and results will be different.

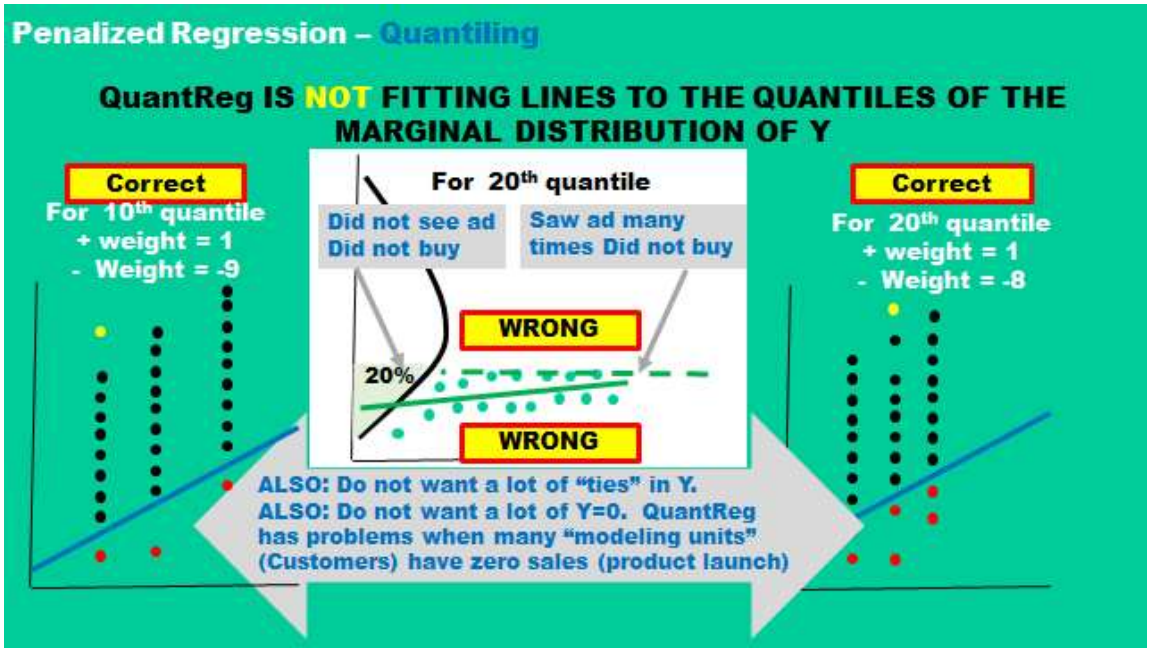

Figure 10

MAIN EXAMPLES OF THE FOUR PROCS (REG GLMSELECT QUANTREG AND QUANTSELECT)

This paper will now focus on four procedures that allow us to use the techniques we have been discussing above (Reg, GLMSelect, QuantReg and QuantSelect). These techniques are run in pairs. Modelers run PROC GLMSelect then any of the following: PROC Reg, PROC GLM or PROC Mixed.

We will use SASHelp.cars as our data set. You can see that it has problems (that we will not fix here) with class levels with small frequencies. No technique is "bullet proof" and a data set, for a real project, should be cleaned before modeling. We will not do that in this example.

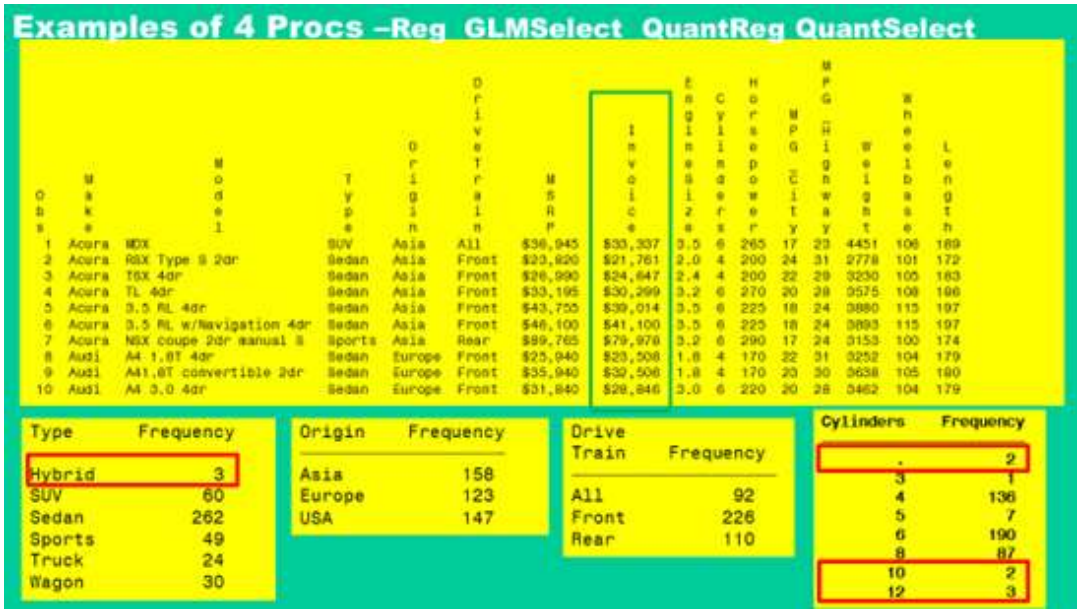

Figure 11

MODELERS RUN PROC GLMSELECT AND THEN PROC REG OR PROC QUANTREG (WE WILL REVERSE ORDER)

The procedures with “select” in the name implement penalized regression techniques but do not have very good diagnostics. Modelers use PROC Reg, PROC Mixed , PROC GLM and PROC QuantReg for diagnostics and for creating plots.

The workflow to be used, in practice, is to run one of the “select” procedures to identify an interesting model, or models, and then to rerun those models using PROC Reg, PROC GLM, PROC Mixed or PROC QuantReg in order to get diagnostics.

This paper will discuss these PROCS in “reverse use” order because we think it might be easier to understand the PROCs in this order: PROC Reg, then PROC GLMSelect then PROC QuantReg and finally PROC QuantSelect.

SASHelp.cars, will be used in all examples. However, variables used in models will differ in the different examples. As you can see in the red rectangles in Figure 10, this data has problems (that we will ignore today). In actuality, techniques discussed below cannot be used without data cleaning and thought.

PROC REG: RIDGE REGRESSION

PROC Reg does Ridge regression and the syntax required is simple.

There is option in the PROC Reg statement that creates Ridge regression (it is shown in red to the right). A plot option produces useful output. This model statement has eight variables from SASHelp.cars and we will use these variables many times.

```
Proc Reg Data=sashelp.cars
        Plots(maxpoints=100 5000)
        OutEst=Needed4RidgePlot
        ridge=0 to .02 by .002 ;
  Model Invoice = EngineSize  Cylinders Horsepower
                  MPG_City MPG_Highway
                  Weight Wheelbase length
                  /lackfit ;
run;quit;
```

Examples of 4 Procs –Reg (8) GLMSelect QuantReg QuantSelect

Parameter Estimates

Variable	Label	DF	B Est.	Standard Error	t Value	Pr > |t|	
Intercept	Intercept	1	3148.60	9241.60	0.34	0.7335	
EngineSize	Engine Size (L)	1	-4702.50	1168.91	-4.02	<.0001	
Cylinders		1	2662.93	726.98	3.66	0.0003	
Horsepower		1	223.52	11.64	19.20	<.0001	
MPG_City	MPG (City)	1	-107.36	277.76	-0.39	0.6993	Non-Sig.
MPG_Highway	MPG (Highway)	1	622.62	272.34	2.29	0.0227	
Weight	Weight (LBS)	1	6.036	1.41495	4.27	<.0001	
Wheelbase	Wheelbase (IN)	1	-542.116	134.32	-4.04	<.0001	
Length	Length (IN)	1	3.79871	73.73	0.05	0.9589	Non-Sig.

Figure 12

A quick examination of names of variables in Figure 12 suggests that several of the variables are collinear and, as expected, some of these variables are not significant.

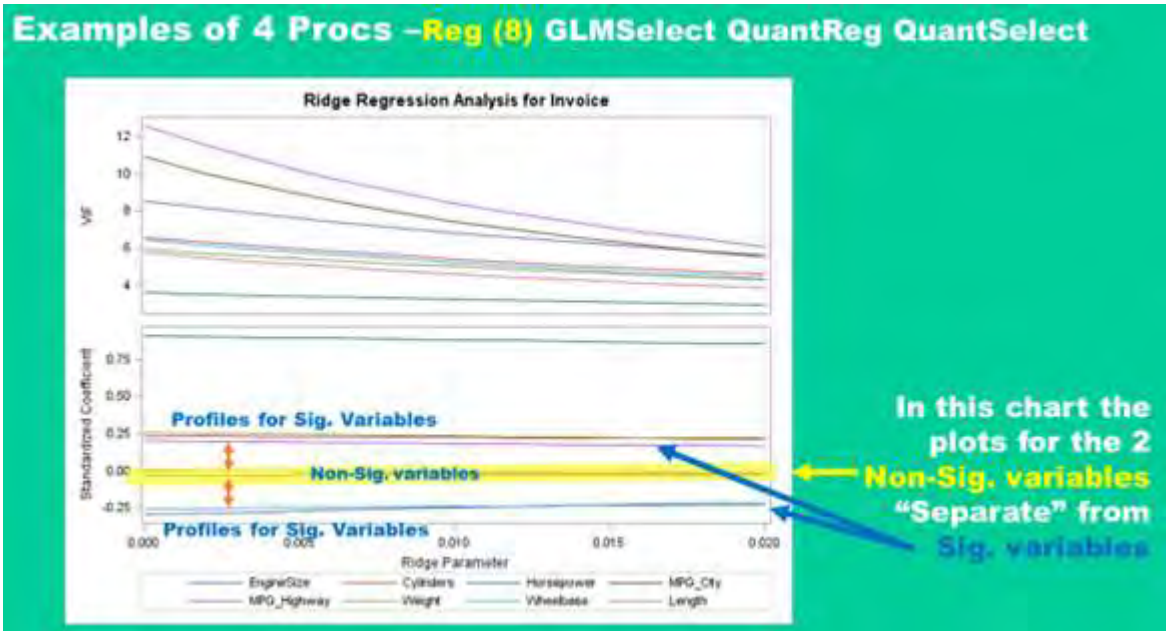

Figure 13

In Figure 13, the X variables are a mixture of predicting and non-predicting and there are gaps between the two types of variables. When variables in the model statement are a mixture of significant and insignificant variables, Ridge regression can produce Ridge plots, with gaps, like the one in Figure 13. This model is a mixture of significant variables and variables that have no predictive power (plots close to zero). The Ridge plots suggest that some of these variables can be removed in our efforts to make a parsimonious model. The usual modeling process is PROC GLMSelect followed by PROC Reg because we want to take advantage of the diagnostics in PROC Reg.

PROC GLMSELECT

This example of GLMSelect uses the LASSO penalization method. There are many options, and they can be easily coded, to split the data sets.

```
Proc GLMSelect Data=sashelp.cars
               Plots=all;
  Partition fraction(validate=.3);
  Model Invoice =
  EngineSize  Cylinders Horsepower
  MPG_City MPG_Highway
  Weight Wheelbase length
  /Selection=LASSO (choose=validate)
                  ;
run;quit;
```

GLMSelect only has options that will allow you to split the incoming data set into two or three partitions. Once you split the data, SAS senses the split, and changes the type of output so that it is appropriate for your modeling process.

GLM select calculates at least eight different measures of goodness of fit. It allows you to specify which measure of goodness of fit you would like to have used as the judgment of overall model quality. The code above specifies that the incoming data set is partitioned into 70% training and 30% validate and that the model quality be judged on the statistic from the validate data set. However it does not specify which of the eight statistics should be used as the judgment of model quality.

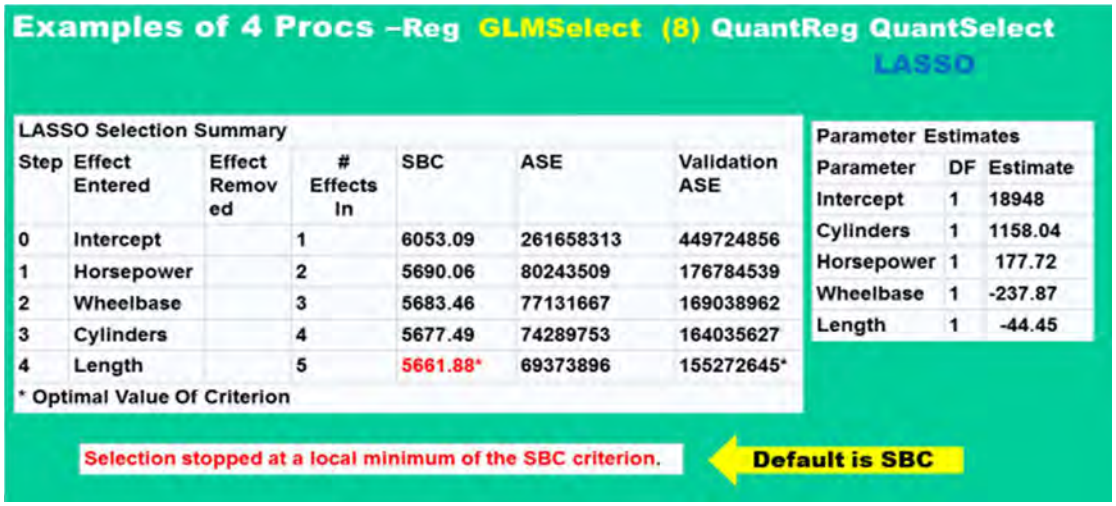

Figure 14

Figure 14 shows the model selection summary and that SBC was used as a test statistic.

The process of:

partitioning the data

and selecting the proper goodness of fit measure

and coding so that the goodness of fit measure is applied to the data partition you intend

is beyond the scope of this paper but hopefully will be covered in the future paper on GLMSelect.

We will now look at another example from GLMSelect. This time with fewer X variables in the model (see white box below).

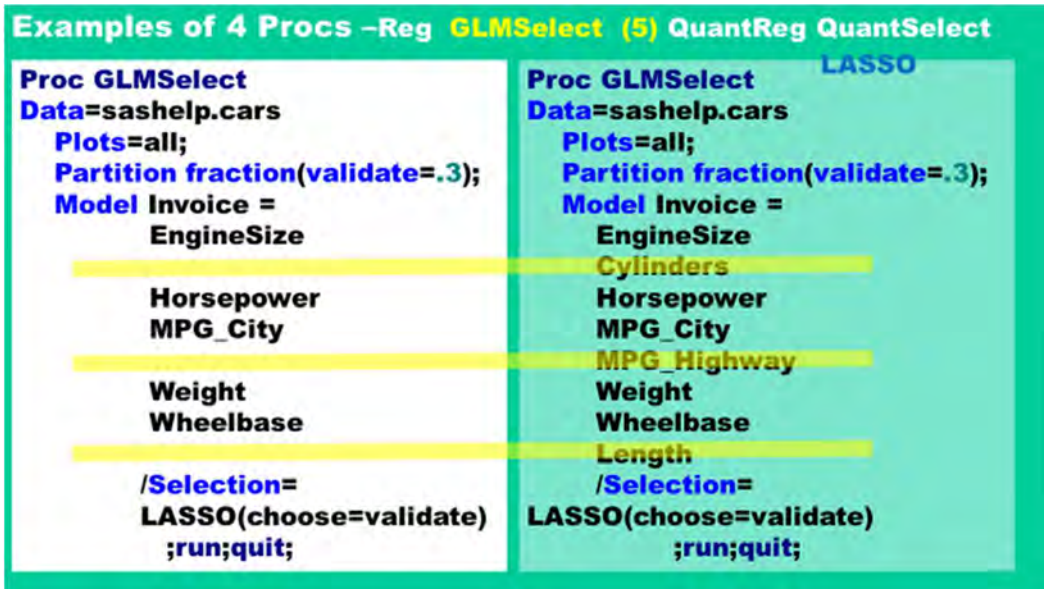

Figure 15

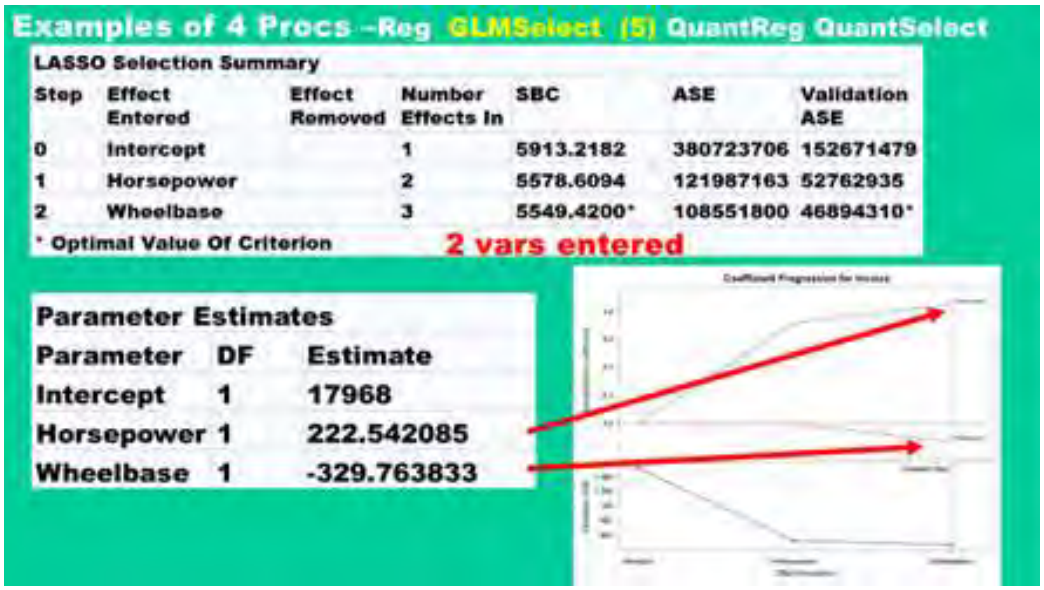

Figure 16

The upper box in Figure 16 shows the LASSO selection summary. Only two variables were added. In the bottom right of Figure 16 you can see a two-part chart. The top part shows coefficient progression as variables enter the model. The bottom part shows change in Validation ASE as variables entered the model.

The LASSO coefficient progression charts should only be interpreted at the points where variables enter or leave the model. They show the estimates of the parameters as points. In order to make the chart easier to read these points are collected by lines. One should not attempt to interpret the lines between points where you can see variables enter/leave.

However, one can get some idea of the stability of a variable by tracking lines on the chart. It seems the β for horsepower is relatively stable as variables entered the model. The bottom chart shows that adding a second variable did relatively little to the predictive power of the model. Horsepower is the biggest predicting X that we have tried in our models.

It also shows the model selection summary and the final βs. In the right-lower corner we see the coefficient progression plot and a plot of validation ASE as a function of variables entered.

Figure 17 shows that PROC GLMSelect will do model Averaging. It is the only one of the four PROCs, covered in this paper, that will do Model Averaging. Model averaging, very simple in SAS, is usually considered a data mining or machine learning technique. It comes free with SAS stat.

The bottom part of the slide shows that 100 samples were built from the raw data set SASHelp.cars and the sampling method was with replacement-suggesting that this was a bootstrap sample.

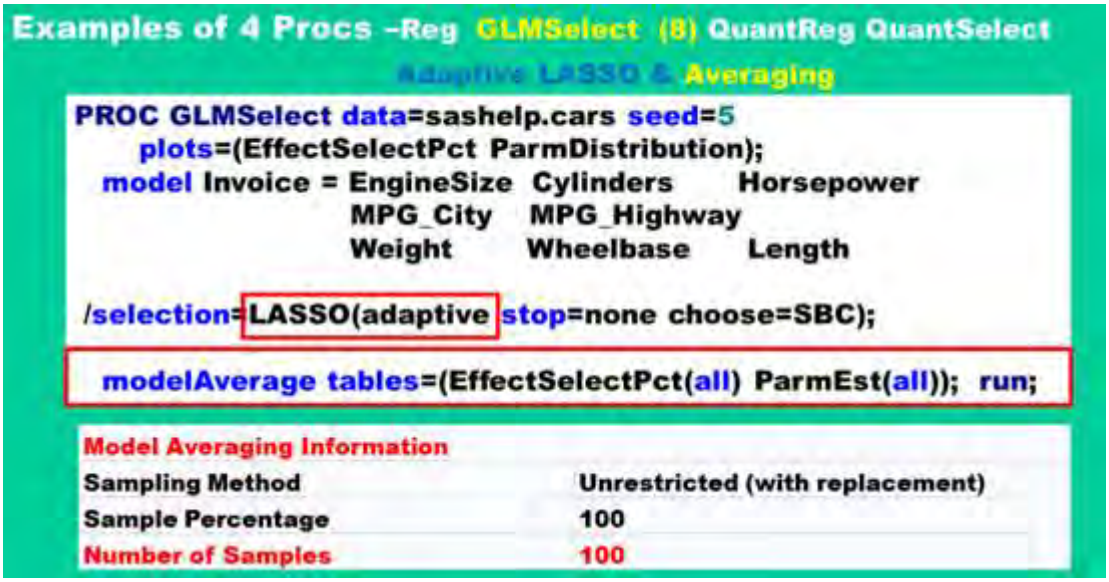

Figure 17

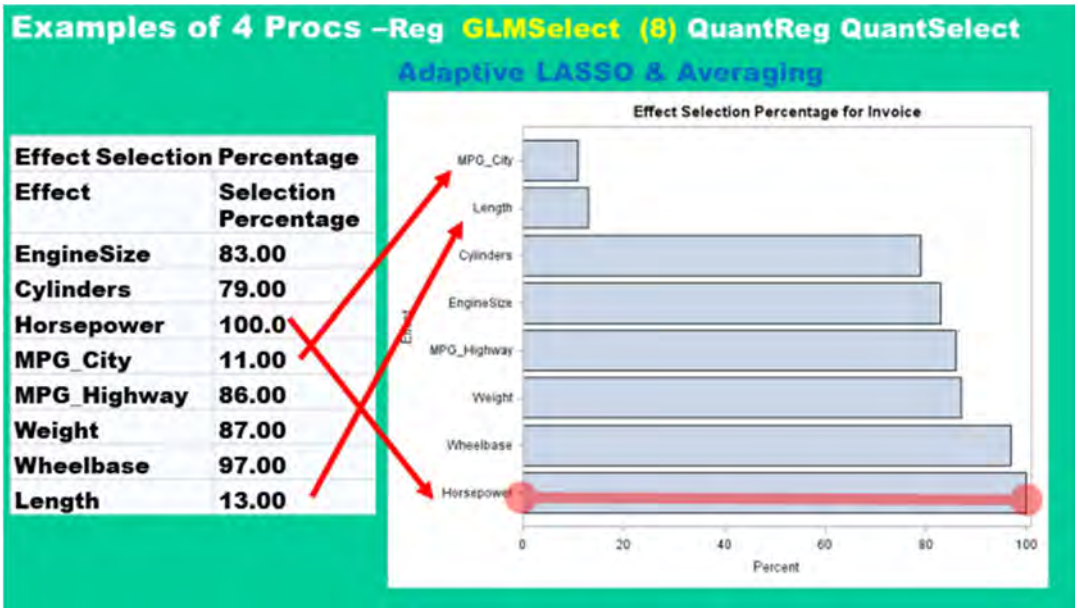

Figure 18

Figure 18 shows some of the standard output from PROC GLMSelect with Model Averaging. This output is an indication of variable importance - a variable importance chart. It shows that horsepower was selected in 100% of the 100 models created. Other variables, like “length of car” and “miles per gallon city”, were selected infrequently. SAS built 100 models from randomly selected observations from SASHelp.cars and “length of car” and “miles per gallon city” were in only a few of the 100 models.

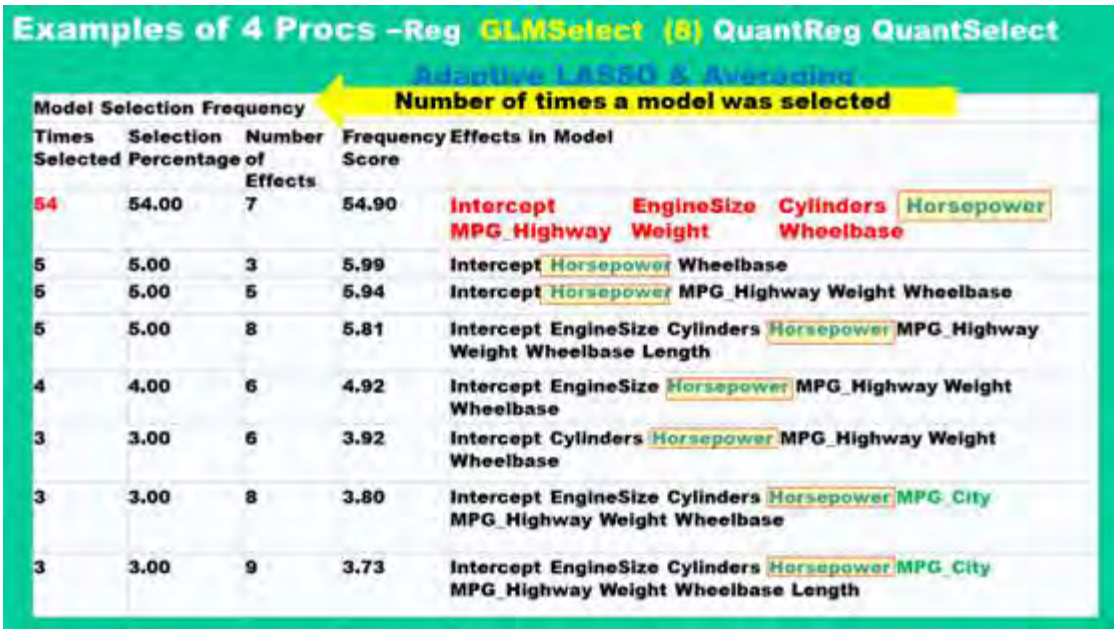

Figure 19

Figure 19 is a different summary of the 100 models that were produced - a model importance chart. This is a model summary, as opposed to the variable summary in Figure 19.

In Figure 19, we see the seven variable model, in red text, was selected fifty-four percent of the time. This means that from SASHelp.cars, when observations were selected randomly, this model was selected most of the time as the model that best described the data. This model is insensitive to the outliers in Y.

Figure 19 does not show all of the models that were created, there is not enough space on a PowerPoint slide for that. However, with this truncated output, you can determine that horsepower was selected in all of the models *that we can see*. This is consistent with horsepower being selected in all models in the chart in Figure 18. "Miles per gallon city" (shown in green) appears relatively infrequently this is also consistent with Figure 18.

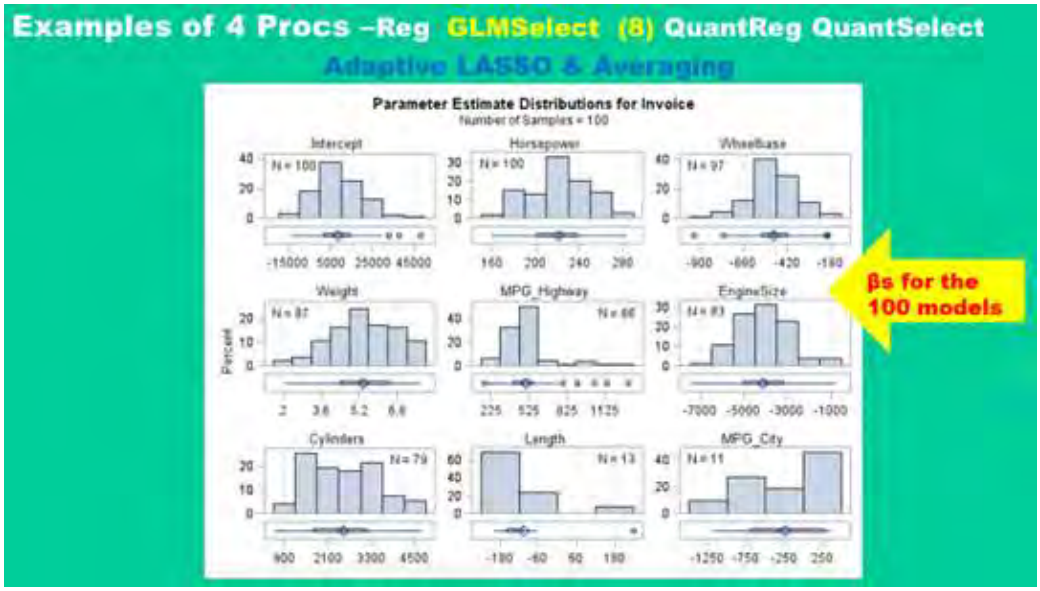

Figure 20

Figure 20 shows the distribution of the β values for the hundred models built.

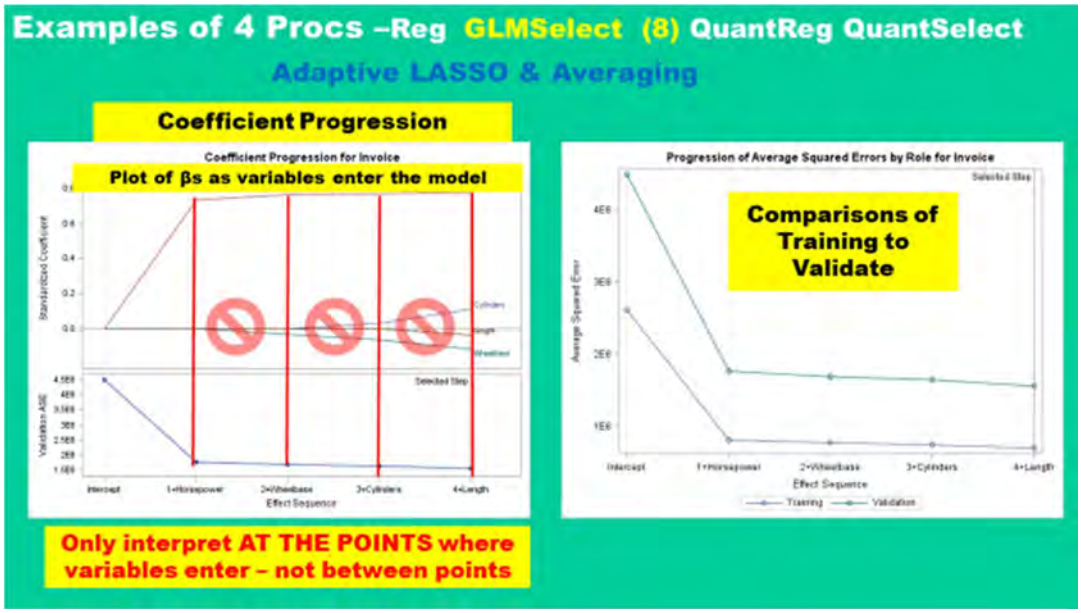

Figure 21

Figure 21 shows the coefficient progression for variables entering the model. This example shows model averaging and the requested technique was LASSO, so SAS produced a LASSO plot. Parameters should only be read where variables enter/leave the model (red lines) and not at points between (red international no sign).

The chart on the right compares the error in the training and validate data sets. Where the plots for the training and test data sets start to diverge is a common indication of best model.

QUANTILE REGRESSION

It is relatively easy to get SAS to produce a Quantile Regression.

Quantile is an option on the model statement and all you have to do is list the quantiles you want. Here we request a 10th percentile, a 50th percentile and a 90th percentile.

Figure 22 shows that some of these variables are not significant for any quantile. The figure also shows that the models produced by quantile regression are very different for the different quantiles.

```
Proc QuantReg Data=sashelp.cars
Algorithm=interior
                Plots=all
                CI=resampling;
Model Invoice = EngineSize  Cylinders
                  Horsepower MPG_City
                  MPG_Highway Weight
                  Wheelbase length
                  /quantile = .1 .5 .9
                   ;
run;quit;
```

Horsepower is significant and its β value changes by a factor of two between quantile .1 and .9. Wheelbase is significant and its β value changes by a factor of two. "Miles per gallon highway", while not significant, has β values that vary greatly. Change in βs for "MPG Highway" and "MPG City" could be caused by either, or both, of two reasons. The change could be caused by the parameter estimates being different for the subjects in different quantiles or because the two MPG figures are highly collinear and, therefore, parameter estimates are inherently unstable. In a "real" analysis only one of these variables would be used, unless there was interest in the effect of one, conditioning on the other. In that case, one might use the difference between them as a predictor.

Examples of 4 Procs –Reg GLM Select QuantReg (8) QuantSelect

We use QuantReg because QuantReg has T & P values

by Quantile

B values for different Quantiles are different:

Parameter Estimates Quantile=.1

Parameter	DF	Estimate	Pr > |t|
Intercept	1	751.5983	0.9168
EngineSize	1	-2491.69	0.0279
Cylinders	1	1582.503	0.0193
Horsepower	1	110.2486	<.0001
MPG_City	1	-324.288	0.2573
MPG_HWay	1	566.3684	0.0150
Weight	1	5.0636	0.0002
Wheelbase	1	-205.891	0.0587
Length	1	-43.8742	0.4381

Parameter Estimates Quantile=.5

Parameter	DF	Estimate	Pr > |t|
Intercept	1	-302.685	0.9620
EngineSize	1	-4585.34	<.0001
Cylinders	1	2249.790	<.0001
Horsepower	1	169.8368	<.0001
MPG_City	1	-88.5550	0.5857
MPG_HWay	1	441.8109	0.0026
Weight	1	6.5756	<.0001
Wheelbase	1	-293.968	0.0019
Length	1	-44.1013	0.3018

Parameter Estimates Quantile=.9

Parameter	DF	Estimate	Pr > |t|
Intercept	1	11310.57	0.3449
EngineSize	1	-3244.67	0.1597
Cylinders	1	3307.171	0.0016
Horsepower	1	241.7681	<.0001
MPG_City	1	269.8456	0.4901
MPG_HWay	1	272.7960	0.4152
Weight	1	8.1504	<.0001
Wheelbase	1	-445.094	0.0013
Length	1	-138.519	0.1250

Figure 22

The idea that makes Quantile Regression so attractive is that the different Quantiles can represent different types of people - different market segments.

The exciting idea is that Quantile Regression, with the same dataset , lets us produce equations for people who buy very little of our product (10th percentile buyers) and people who buy a lot of our product (90th percentile buyers). This can help in creating more effective marketing campaigns. Additionally, this technique is not limited to business and can be applied a number of scientific disciplines.

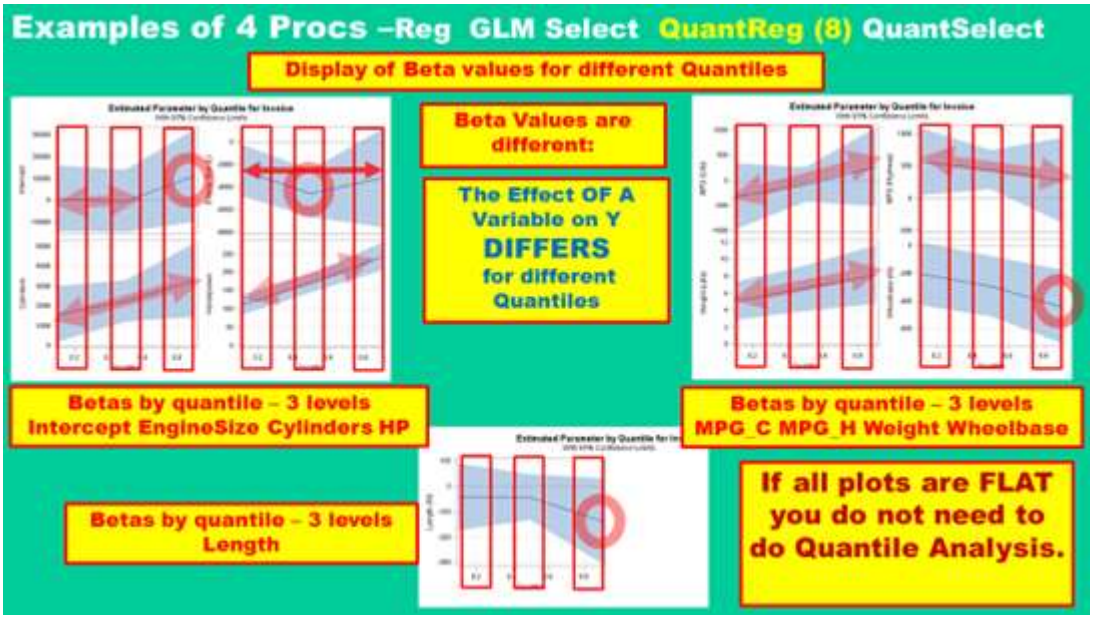

Figure 23

The table in Figure 22 is difficult to interpret. There are a lot of numbers to read and the numbers are spread over several pages. SAS uses a graphical way to help analysts interpret the large number of β values

SAS created the output in Figure 23 to make it easier on modelers. Figure 23 shows plots of the β (Y axis) values by Quantile (X axis). Remember, the model only requested three quartiles and modelers are only allowed to look at points, on the plot, immediately above the quantile "mark" on the X axis (see red rectangles). Our code requested three quantiles and there are only three points to be read on each of these plots.

Where the β values are all equal, where the plots are flat, the effect of the X variable on the Y variable is the same for all Quantiles. Where the plots are flat over all quantiles, a modeler does not need to do Quantile analysis. There is no differential response to the variable

This paper does not show much in the way of diagnostics on this model. The model has non-significant variables so this example is presented as a teaching tool. It is not presented as an example of best practice in modeling. It only allows us to discuss features of these plots.

Many of these variables seem to have constantly increasing βs. Looking at cylinders (left most plot – bottom left section), it seems that cylinders is associated with increasing invoice price across all quantiles. It seems that people buying expensive cars are willing to pay more for having more cylinders. People buying inexpensive cars do not care about the number of cylinders.

These plots provide a great amount of information.

As was mentioned before, the “select” procedures do not have an abundance of diagnostics. A modeler must run PROC GLM, PROC Reg, PROC QuantReg or PROC Mixed in order to get diagnostics.

This slide shows some of the existing diagnostics for PROC QuantReg.

Please search other SAS papers for discussions of model diagnostics.

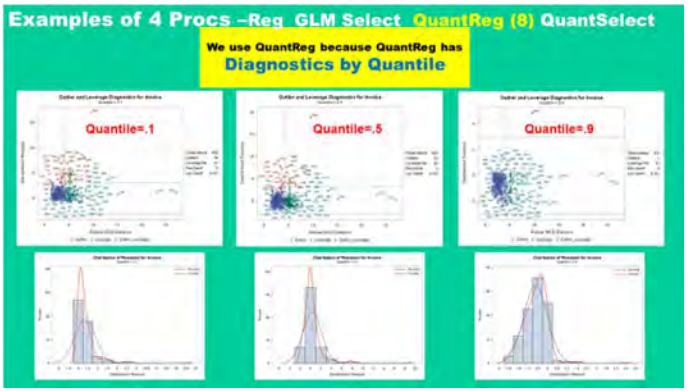

Figure 24

A PROC QUANTREG FOR ONE X VARIABLE

This is a second example for PROC QuanttReg and it uses only one X variable, though that X variable is raised to the first, second and third power.

As you can see in Figure 25, the βs differ substantially among the three quantiles. Also notice the effect of one of the SAS options being ignored by the coder.

```
Proc QuantReg data=Sashelp.cars
plots=all;
model
     Invoice
      = Horsepower
        Horsepower*Horsepower
        Horsepower*Horsepower*Horsepower
       /quantile = .2 .5 .8;
       ;run; quit;
```

As you create interactions the labels for the interaction that longer and there is an option that lets you set the length of the variable. Truncation occurred automatically.

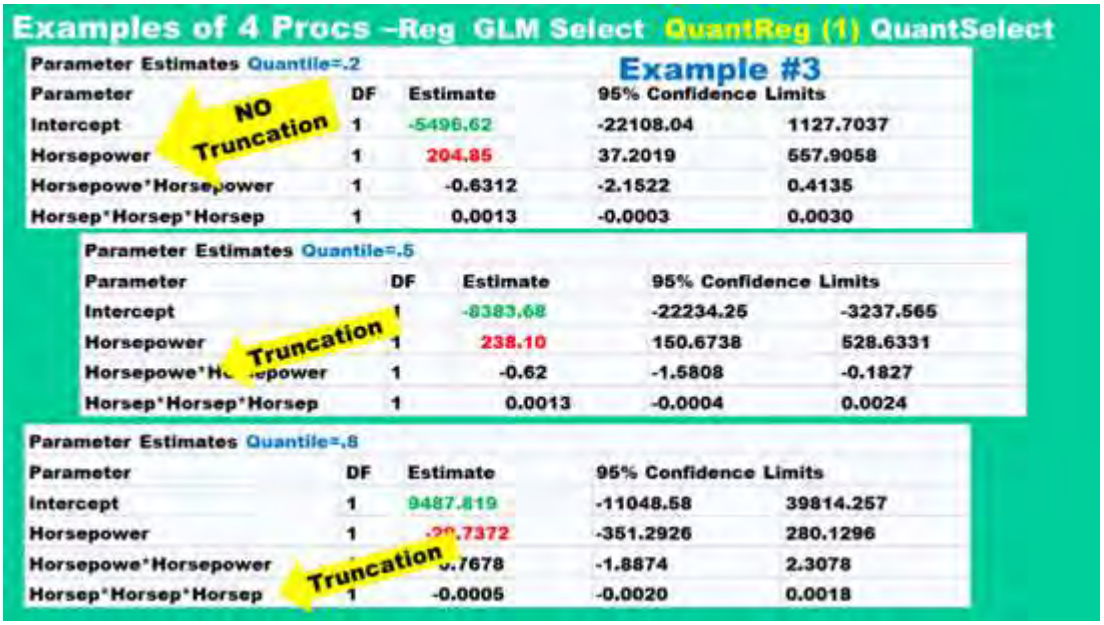

Figure 25

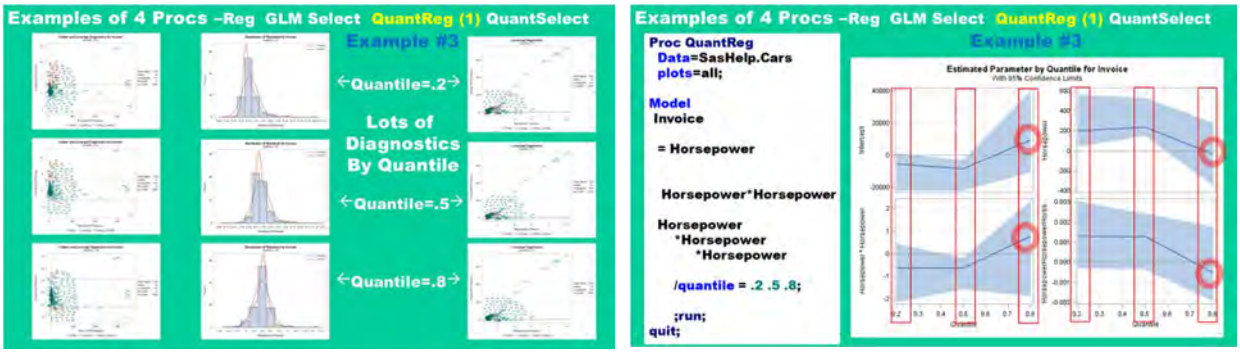

Figure 26 (Two parts)

Figure 26 shows some of the diagnostics that can come out of PROC QuantReg and also the profile plots.

If there is only one X variable PROC QuantReg will automatically create a profile plot, like the one in Figure 27. Plots like this can be produced for models with more than one X variable but such plots require coding and use of SAS graphics.

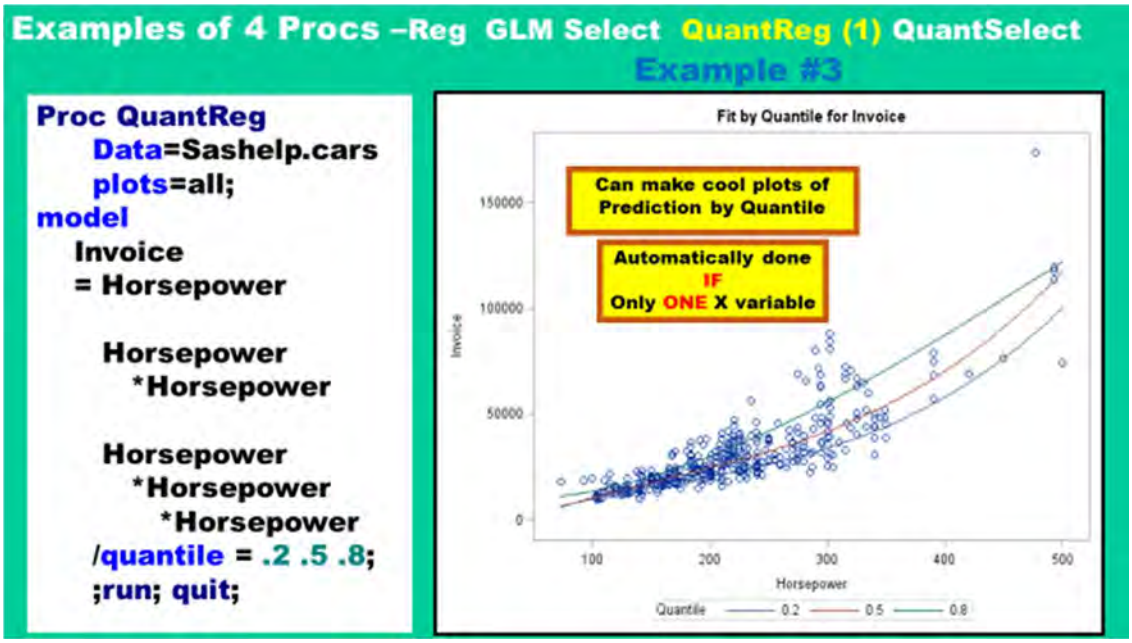

Figure 27

PROC QUANTSELECT

This example uses QuantSelect and the LASSO penalization method to create Quantile models with 11 X variables.

Figure 28 shows that the 10th percentile and 50th percentile models do not look very much like the 90th percentile model.

This slide, and output like it from other data sets, has raised doubts about the ability of an OLS model, with just one equation, to effectively model how X variables affect a Y variable. We can see that the number of variables differs for the different quartiles this suggests that models with "one line" might not be the best.

```
Proc QuantSelect Data=sashelp.cars
Plots=all ;
  partition fraction(Validate=0.3) ;
  Class origin  DriveTrain type /split;
  Model Invoice =
          Origin DriveTrain Type
          EngineSize  Cylinders
          Horsepower
          MPG_City MPG_Highway
          Weight Wheelbase length
          /quantile = .1 .5 .9
          Selection=LASSO  ;
run;
%put _user_;
```

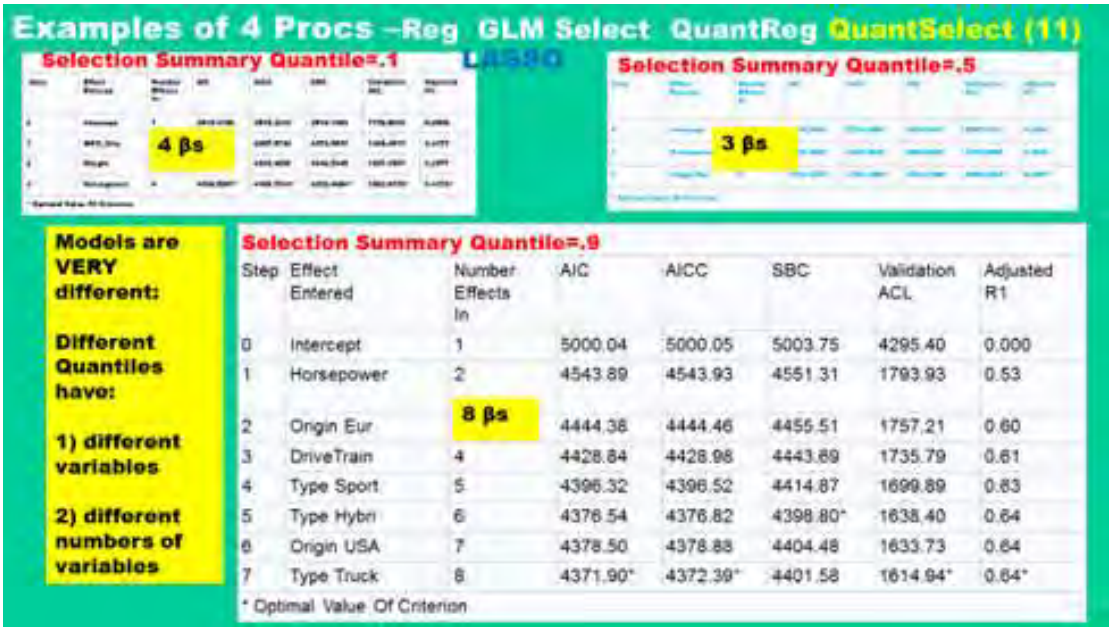

FIGURE 28

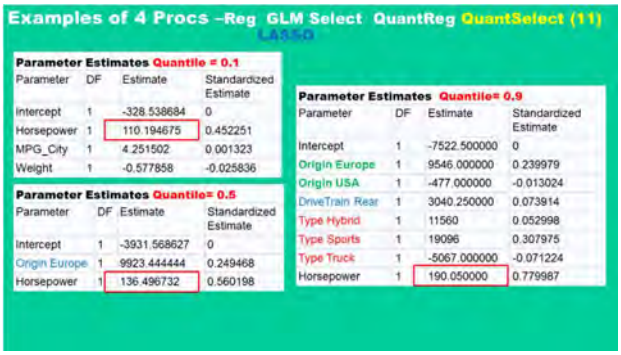

FIGURE 29 (TWO PARTS)

Figure 29 (left) shows the β values for the different quantiles and we can see that the coefficient for horsepower is fairly different for the three quantiles.

Figure 29 (Right) gives a very high level view of some of the charts that are produced in PROC QuantSelect. While these pictures are small, you can see that the charts are not very similar. This supports the idea that modeling the mean response to an X vector, using OLS, might not adequately describes the underlying process in the data.

FIGURE 30

Modelers will want to avoid typing by having PROC QuantSelect “communicate” the selected models to other PROCs. SAS allows this “communication between PROCs” via automatically created Macro variables. Figure 30 shows a list of the variables that were created in this example. These were created to allow a programmer to use PROC QuantSelect to create a number of models and store the variables in the models in macro variables.

As a second step, a SAS programmer would use macros, to loop over all of these macro variables and create diagnostics for each model with minimal typing.

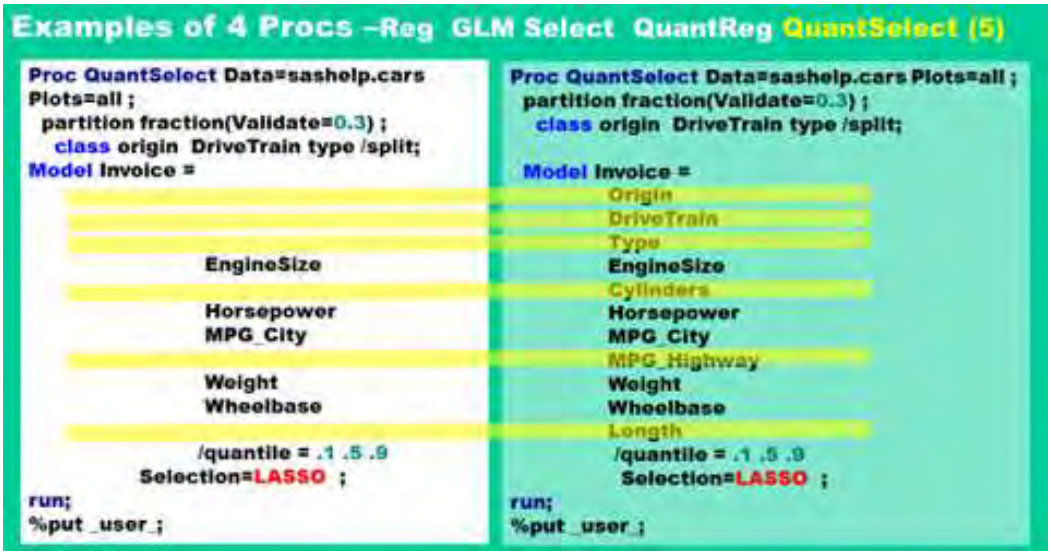

Figure 31

Figure 31 shows the code for a second example of PROC QuantSelect. It has the same options as the previous example but uses fewer variables (see white box).

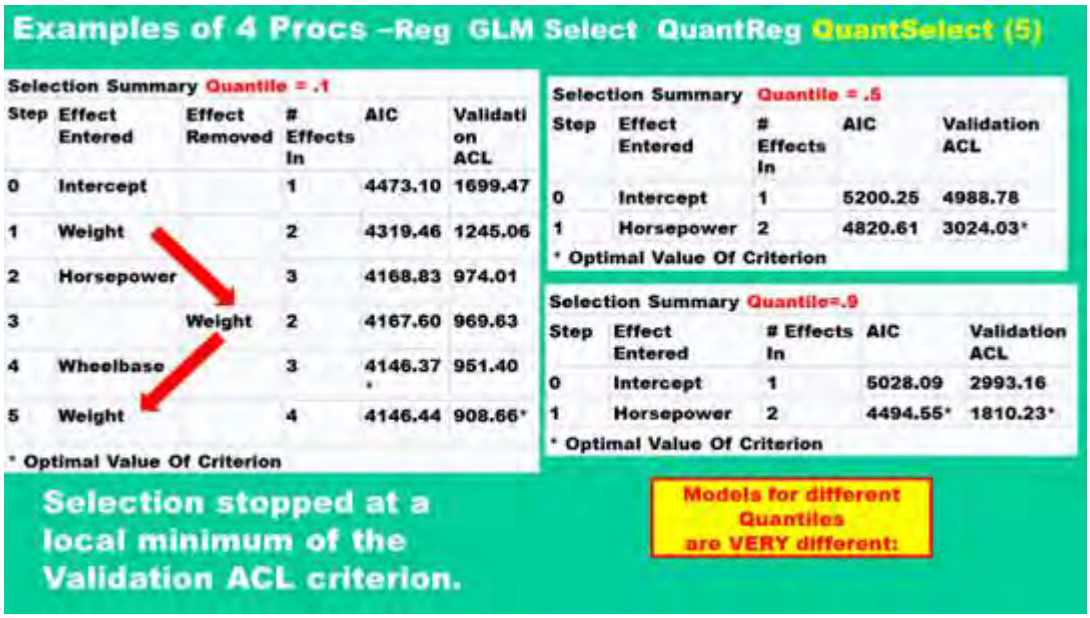

Figure 32

Figure 32 shows the selection summary for the three different quantiles. You can see from the plots that the process differs for the three different quantiles. In this case, the process for creating a model for the 10th percentile is more complicated than the process for the 50th or the 90th percentile. The fact that different processes are used to create models for different quantiles lends support to the idea that OLS, with only one model equation that predicts the conditional mean response, is not rich enough to describe the true relationship between a set of X variables and Y.

Figure 33 (two parts)

In Figure 33, the left-hand slide shows how the profile plots differ along the three quartiles and the right-hand chart shows how the β values differ among the quartiles

Summary.

This is been an overview of penalized regression in SAS. These new techniques are exciting and have potential to improve modeling efforts with relatively little additional work. This paper is intended to show how several different statistical procedures are related to each other and how they might be used. The idea is that a modeler uses a select procedure to find a parsimonious model and then a follow-up procedure to produce diagnostics.

We plan that this paper is the lead-off paper, and an overview paper, for a small series of papers on penalized regression and Quantile regression. If you like this paper you might want to look for others in the series.

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

Russ Lavery russ.lavery@verizon.net
(thanks to Peter Flom PeterFlomConsulting@mindspring.co)