

Alternative methods of regression when OLS is not right

Peter L. Flom Peter Flom Consulting, LLC

ABSTRACT

Ordinary least square regression is one of the most widely used statistical methods. However, it is a parametric model and relies on assumptions that are often not met. Alternative methods of regression for continuous dependent variables relax these assumptions in various ways. This paper will explore PROCs such as QUANTREG, ADAPTIVEREG and TRANSREG for these data.

Keywords: Regression.

INTRODUCTION

Ordinary least squares (OLS) regression is the default regression method for continuous dependent variables, partly because it was one of the first models developed. More recent regression methods are often better suited to particular problems and may give insights that OLS cannot give. In addition, some newer methods relax or eliminate assumptions that may be violated in many regression problems.

QUANTILE REGRESSION AND PROC QUANTREG

MOTIVATION

There are at least two motivations for quantile regression: Suppose our dependent variable is bimodal or multimodal that is, it has multiple humps. If we knew what caused the bimodality, we could separate on that variable and do stratified analysis, but if we don't know that, quantile regression might be good. OLS regression will, here, be as misleading as relying on the mean as a measure of centrality for a bimodal distribution.

If our DV is highly skewed as, for example, income is in many countries we might be interested in what predicts the median (which is the 50th d percentile) or some other quantile; just as we usually report median income rather than mean income.

One more example is where our substantive interest is in people at the highest or lowest quantiles. For example, if studying the spread of sexually transmitted diseases, we might record number of sexual partners that a person had in a given time period. And we might be most interested in what predicts people with a great many partners, since they will be key parts of spreading the disease.

THEORY

A quantile is ordinarily thought of as an order statistic. One type of quantile is the percentile, or 100-quantile. The p th (sample)/(population) percentile is the value that is higher than $p\%$ of all the values in the (sample)/(population). More formally, the τ th quantile of X is defined as

$$F^{-1}(\tau) = \inf[x : F(x) > \tau]$$

where F is the distribution function of X .

The key bit of theory, as noted by Koenker and originally developed by Fox and Rubin is that this problem of sorting can be converted into one of optimization. Specifically, the problem is to minimize

$$E\rho_{\tau}(X - \hat{x}) = (\tau - 1) \int_{-\infty}^{\hat{x}} (x - \hat{x})dF(x) + \tau \int_{\hat{x}}^{\infty} (x - \hat{x})dF(x)$$

This allows relatively simple extension of the problem of ordinary least squares regression to quantile regression. For details, see Koenker.

PROC QUANTREG

Here I outline the basic syntax of PROC QUANTREG and do not go over every detail. For that, you can always see the documentation.

```

PROC QUANTREG <options> ;
CLASS variables ; *SAME AS OTHER PROCS;
MODEL response = independents </ options> ;
OUTPUT <OUT= SAS-data-set> <options> ;
PERFORMANCE <options> ;

```

As usual, the first statement invokes the procedure. There are also BY, ID, TEST, EFFECT and WEIGHT statements, all of which operate similarly to other statistical procedures. The PROC QUANTREG statement has some options that are dissimilar to other procedures. You can choose the algorithm and the method for calculating confidence intervals, but, as usual, SAS® has sensible defaults. Several of the algorithms need starting points, and you can specify these using the INEST statement. There are many plotting options, dealt with below.

The key statement is the model statement. The usual syntax applies, but the options are different. The key option is the QUANTILE option, the syntax of which is

```

QUANTILE=number-list | PROCESS

```

This option specifies the quantile levels for the quantile regression. You can specify any number of quantile levels in the number list. You can also compute the entire quantile process by specifying the PROCESS option. Only the simplex algorithm is available for computing the quantile process. The default is a median regression, which corresponds to QUANTILE=0.5. The PROCESS option calculates the entire quantile process.

ODS GRAPHICS AND PROC QUANTREG

Graphics are always important tools for evaluating models, but this is especially true in quantile regression. The volume of printed output can become overwhelming, because if you (for example) run quantile regressions on the .05, .1095 quantile, that is 19 regressions, and there will be approximately the same amount of output as running 19 PROC GLMs. Fortunately, SAS now offers excellent graphics that can be obtained relatively easily. They can also be modified using the graph template language (GTL) but I will not discuss that in this paper.

EXAMPLE: BIRTHWEIGHT DATA

Predicting low birth weight is important because babies born at low weight are much more likely to have health complications than babies of more typical weight. The usual approaches to this are either to model the mean birth weight as a function of various factors using OLS regression, or to dichotomize or otherwise categorize birth weight and then use some form of logistic regression (either 'normal' or ordinal). Both these are inadequate. Modeling the mean is inadequate because, as we shall see, different factors are important in modeling the mean and the lower quantiles. We are often interested in predicting which mothers are likely to have the lowest weight babies, not the average birth weight of a particular group of mothers. Categorizing the dependent variable is rarely a good idea, principally because it throws away useful information and treats people within categories as the same. A typical cutoff value for low birth weight is 2.5 kg. Yet this implies that a baby born at 2.49 kg is the same as a baby born at 1.0 kg, while one born at 2.51 kg is the same as one who is 4 kg. This is clearly not the case.

SAS provides birthweight data that is useful for illustrating PROC QUANTREG. I added an ID variable to the data set provided by SAS:

```

data new;
  set sashelp.bweight;
  count + 1;
run;

```

A very simple model In the SAS documentation for PROC QUANTREG, there is a program with a reasonable model for a set of birth weight data. However, for illustrative purposes, it will be clearer to first look at an unrealistically simple model, with only one independent variable. One continuous variable is maternal weight gain. Perhaps the first graph to look at is a graph of

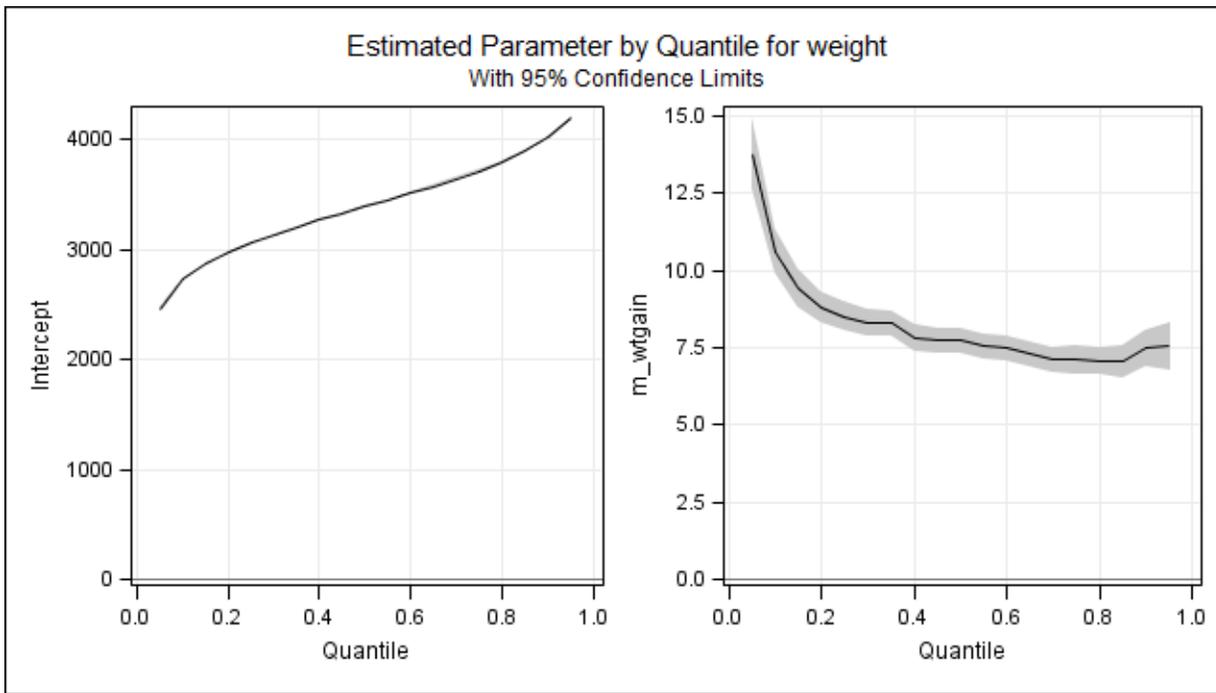


Figure 1: Parameters by quantile

the importance of the parameters at each quantile. The code for such a model and graph is

```
proc quantreg ci=sparsity/iid algorithm=interior(tolerance=1.e-4)
    data=new;
    class visit ed;
    model weight = MomWtGain/quantile= 0.05 to 0.95 by 0.05
        plot=quantplot;
run;
```

this produces figure 1

The left portion of this plot shows the predicted birth weight for each quantile, if the mother gains no weight. Not surprisingly, it is monotone upwards, and roughly like a normal distribution. But the main interest is in the panel on the right. Maternal weight gain makes much more difference in the lower quantiles than the upper ones (at least, in this oversimplified model). For example, at the .1 quantile, each kg of weight gained by the mother relates to about 12 g gained by the baby. But at the upper quantiles, it relates to only about 7.5 g.

Another graph is the fit plot (figure 2), available when there is a single, continuous IV. This allows a more detailed look at the relationship between the IV and the DV at different quantiles.

A FULLER MODEL

The fuller model used in the SAS example and adapted from Koenker includes the child's sex, the mother's marital status, mother's race, the mother's age (as a quadratic), her educational status, whether she had prenatal care, and, if so, in which trimester, whether she smokes, and, if so, how many cigarettes a day, and her weight gain (as a quadratic).

Mother's marital status was coded as married vs. not married; race was either Black or White (it is not clear if mothers of other races were simply excluded), mother's education was coded as either less than high school (the reference category), high school graduate, some college, or college graduate. Prenatal care was coded as none, first trimester (the reference category), second trimester or third trimester. Mother's weight gain and age were centered on the means. The SAS code for this model is

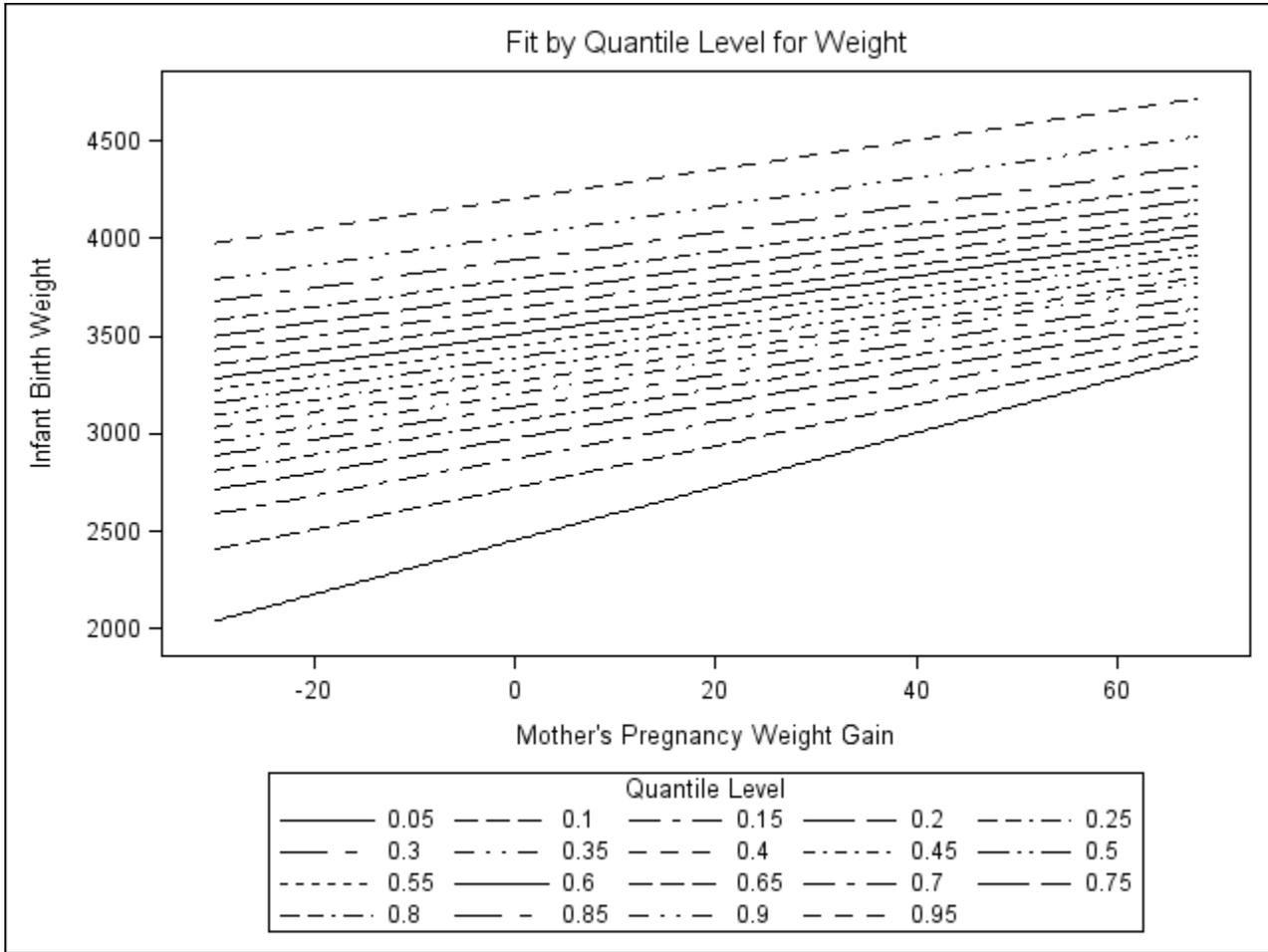


Figure 2: Parameters by quantile

```

proc quantreg ci=sparsity/iid algorithm=interior(tolerance=1.e-4)
  data=new;
  class visit MomEdLevel;
  model weight = black married boy visit MomEdLevel MomSmoke
    cigspersday MomAge MomAge*MomAge
    MomWtGain MomWtGain*MomWtGain/
    quantile= 0.05 to 0.95 by 0.05
    plot=quantplot;
run;

```

The quantile plots for this model are shown in figures 3 through 6.

Figure 3 shows the effect of the intercept, the mother being Black, the mother being married and the child being a boy. The intercept is the mean birth weight for each quantile for a baby girl born to an unmarried White woman who has less than high school education, does not smoke, is the average age and gains the average amount of weight. Just about 5% of these babies weigh less than the usual cut-off weight of 2,500 grams. Babies born to Black women are lighter than those born to White women, and this effect is greater at the low end than elsewhere - the difference is about 280 grams at the 5%tile, 180 grams at the median, and 160 grams at the 95%tile. Babies whose mothers were married weigh more than those whose mothers were not, and the effect is relatively constant across quantiles. Boys weigh more than girls, and this effect is larger at the high end: At the 5%tile boys weigh about 50 grams more than girls, but at the 95%tile the difference is over 100 grams.

Figure 4 shows the effects of prenatal care, and the first part of education, figure 5 shows the other education effects and the effects of smoking. Finally, figure 6 shows the effects of maternal age and weight gain. These last two are somewhat harder to interpret, as is always the case with quadratic effects compared to linear effects. One way to ameliorate this confusion is to plot the predicted birth weight of babies for different maternal ages or weight gain, holding other variables constant at their means or most common values. First, we get the predicted values by coding:

```

proc quantreg ci=sparsity/iid algorithm=interior(tolerance=1.e-4)
  data=new;
  class visit MomEdLevel;
  model weight = black married boy visit MomEdLevel MomSmoke
    cigspersday MomAge MomAge*MomAge
    MomWtGain MomWtGain*MomWtGain/
    quantile= 0.05 to 0.95 by 0.05;
  output out = predictquant p = predquant;
run;

```

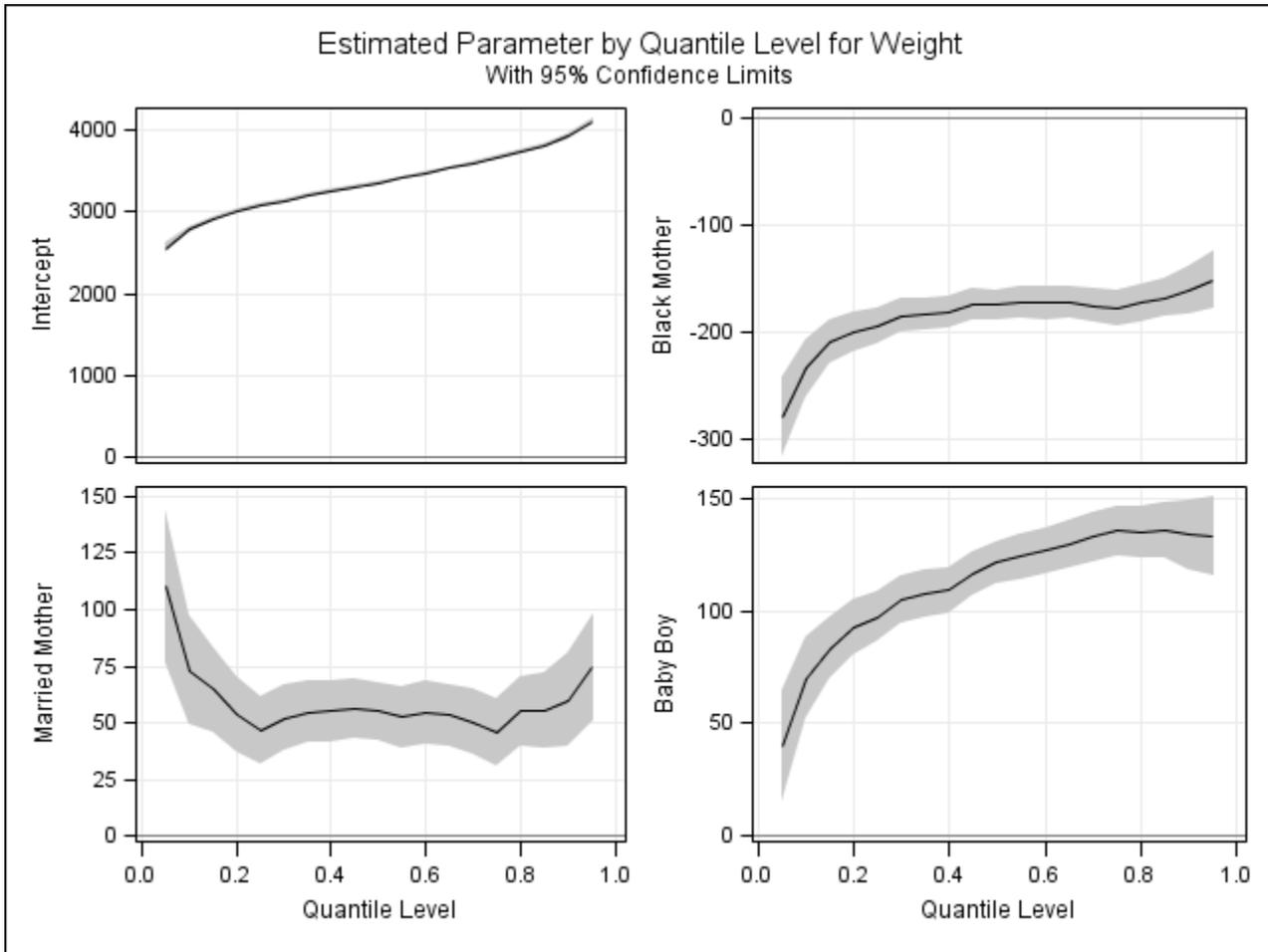


Figure 3: Parameters by quantile

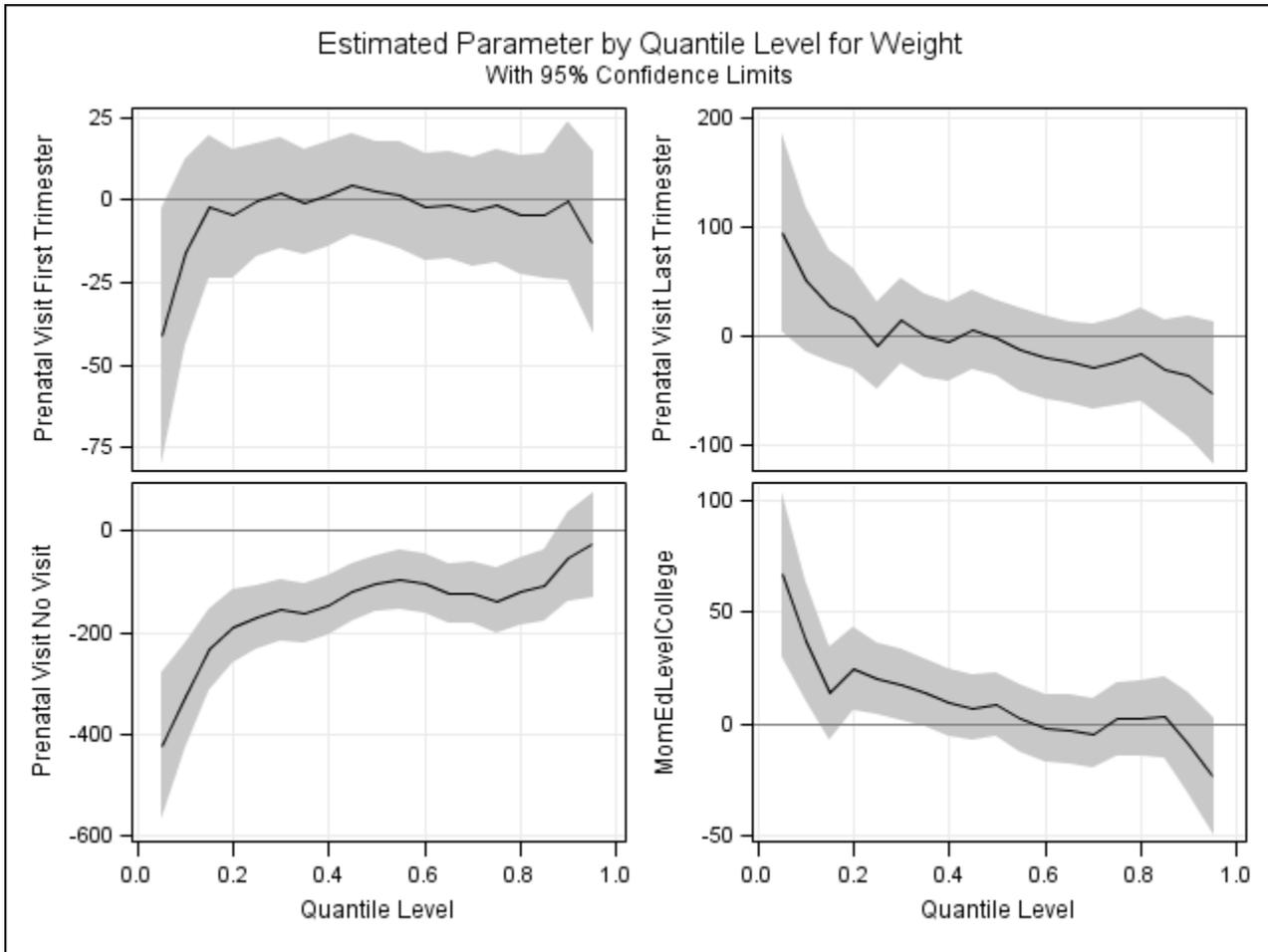


Figure 4: Parameters by quantile, part 2

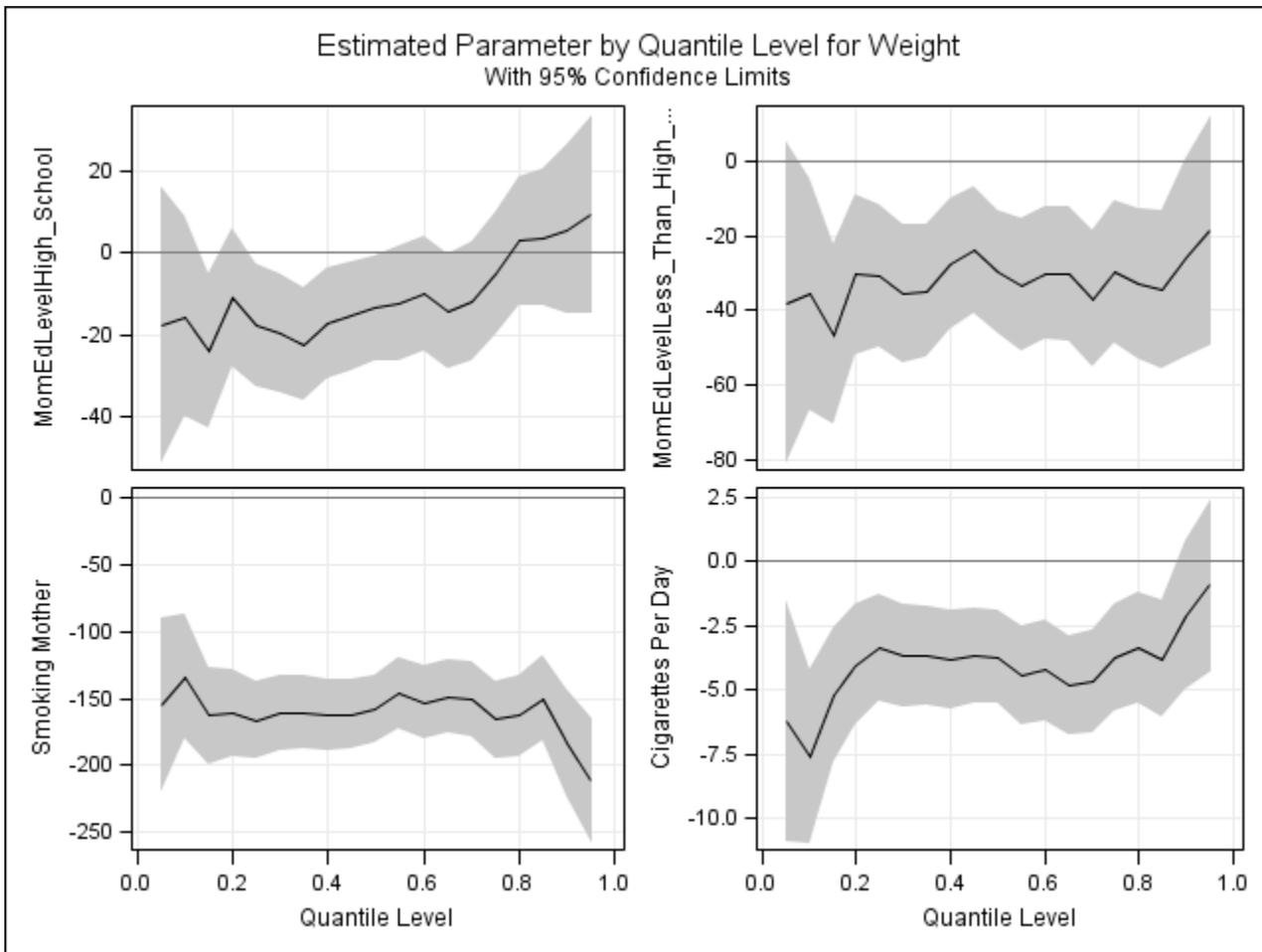


Figure 5: Parameters by quantile, part 3

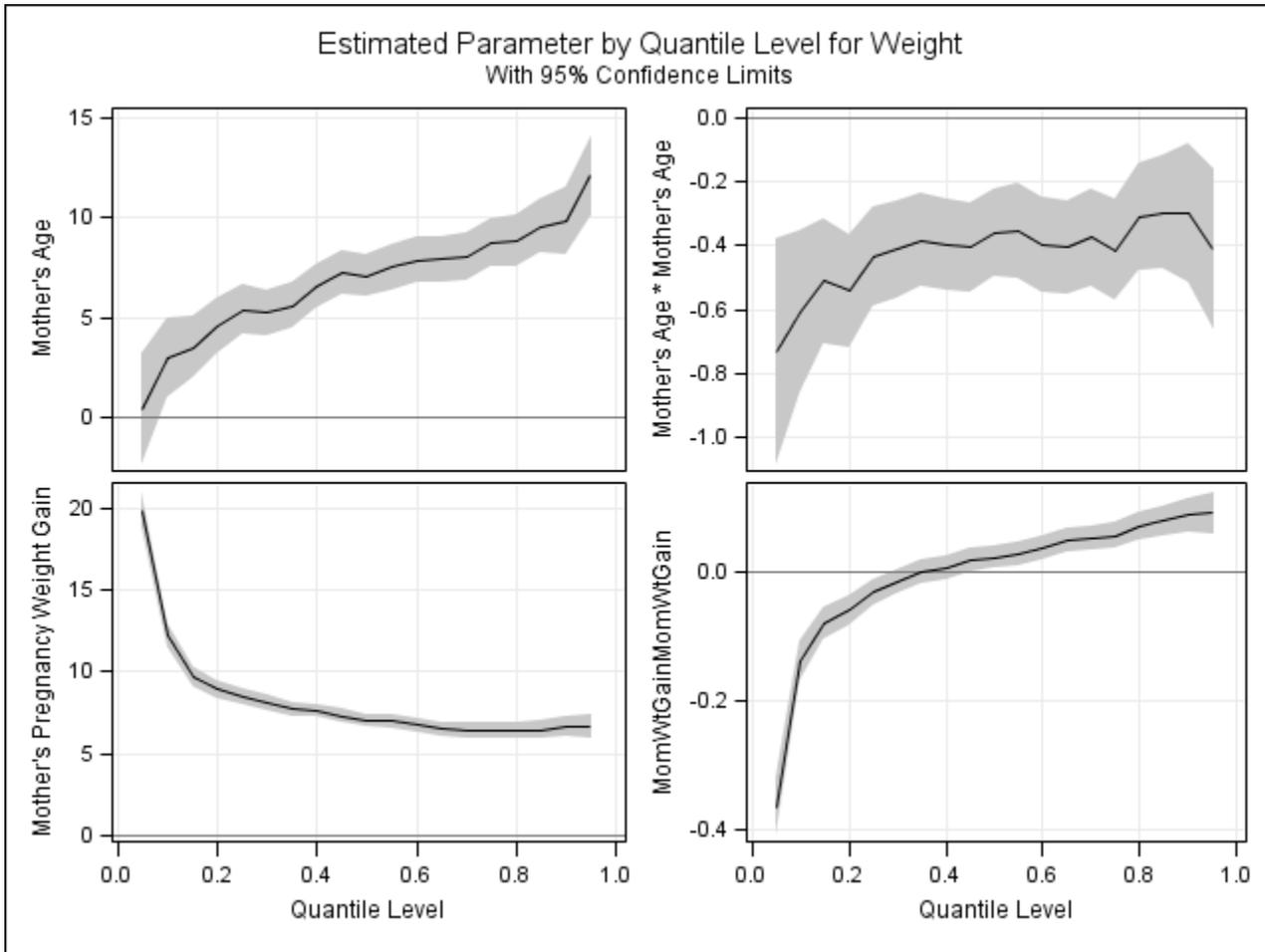


Figure 6: Parameters by quantile, part 4

then we subset this to get only the cases where the other values are their means or modes. First, for maternal age:

```
data mwtgainingraph;  
  set predictquant;  
  where black = 0 and married = 1 and boy = 1 and MomAge = 0 and MomSmoke = 0 and visit = 3 and  
run;
```

Then sort it:

```
proc sort data = mwtgainingraph;  
by MomWtGain;  
run;
```

Then graph it.

```
proc sgplot data = mwtgainingraph;  
  title 'Quantile fit plot for maternal weight gain';  
  yaxis label = "Predicted birth weight";  
  series x = MomWtGain y = predquant1 /curvelabel = "5 %tile";  
  series x = MomWtGain y = predquant2/curvelabel = "10 %tile";  
  series x = MomWtGain y = predquant5/curvelabel = "25 %tile";  
  series x = MomWtGain y = predquant10/curvelabel = "50 %tile";  
  series x = MomWtGain y = predquant15/curvelabel = "75 %tile";  
  series x = MomWtGain y = predquant18/curvelabel = "90 %tile";  
  series x = MomWtGain y = predquant19/curvelabel = "95 %tile";  
run;
```

which creates figure 7.

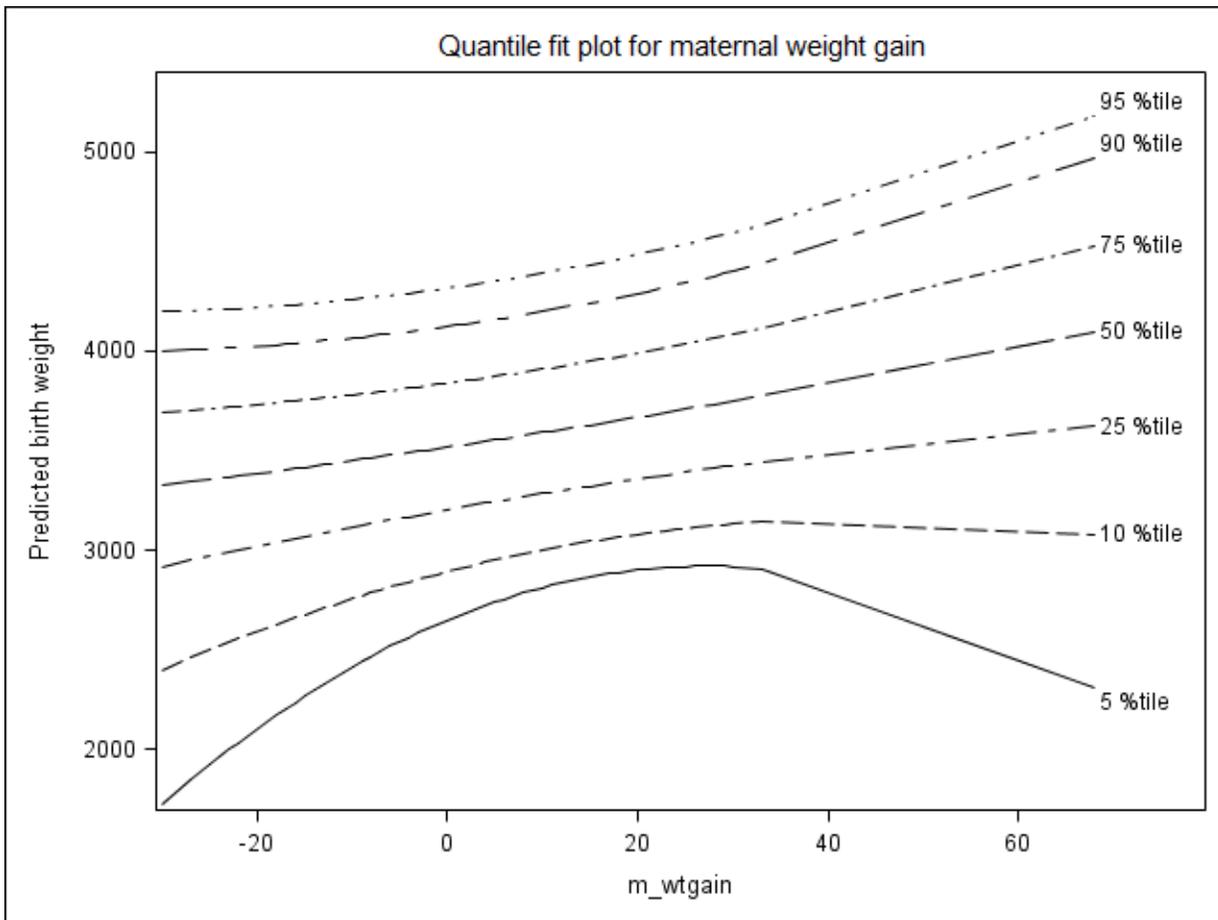


Figure 7: Predicted birth weight by maternal weight gain

This is a fascinating graph! Note that the extreme quantiles are the ones where the quadratic effect is prominent. Further note that mothers who either lose weight or gain a great deal of weight have much higher chances of having low birth weight babies than women who gain a moderate amount. In addition, women who gain a great deal have higher chances of having extremely large babies. This sort of finding confirms medical opinion, but is not something we could find with ordinary least squares regression.

We can do the same thing for maternal age but the effect of age is not that huge, and the quadratic effect is so small that we might consider simplifying the model by eliminating it. On the other hand, if the literature says that there should be strong quadratic effects of maternal age, then either there is something odd about this data set or we have evidence counter to that claim. One thing to note is that this data set spans a limited range of ages - all mothers were 18 to 45 years old. There might be strong effects that occur at younger and older ages.

MULTIPLE ADAPTIVE REGRESSION SPLINES AND PROC ADAPTIVEREG

INTRODUCTION

SAS offers several choices for nonparametric regression including TPSSPLINE, LOESS and GAM, but the first two are limited to relatively low dimensions and the last can involve very long computation times and is not guaranteed to converge for nonnormally distributed dependent variables. PROC ADAPTIVEREG implements multivariate adaptive regression splines (MARS), which were developed by Jerome Friedman in the early 1990s.

WHAT IS A MARS MODEL?

It is possible to view a MARS model in several ways but the simplest is probably to view it as a generalization of a regression tree. Regression trees start with all the data in one node and then split it into two daughter nodes, each of which is as homogenous as possible with regard to the dependent variable. Each daughter node is then split again. Although regression trees are very useful, they have some drawbacks:

1. They are discontinuous. That is, the predicted value of the dependent variable will jump at particular levels of the independent variables. This is hard to justify substantively in most cases
2. They cannot model a purely additive model very well.
3. They do not work well when there are either no interactions or when one or two interactions are dominant.

These drawbacks pose severe limits on the accuracy of the results (see Friedman, 1991).

One algorithm for generating regression trees is a multiply looped algorithm that has a step function in the innermost loop. MARS replaces this step function with a truncated power spline function. For full details, see Friedman, 1991. It turns out that the MARS model can also be viewed in the following form:

$$\hat{f}(x) = \alpha_0 + \sum_{K_m=1} f_i(x_i) + \sum_{K_m=2} f_{i,j}(x_i, x_j) + \sum_{K_m=3} f_{i,j,k}(x_i, x_j, x_k) + \dots$$

that is, a sum of a sum of all basis functions that involve one variable, two variables, three variables and so on. Further details of MARS models, including model selection criteria, lack of fit criteria and computational considerations, are in Friedman (1991) section 3. Their implementation in ADAPTIVEREG is in the SAS documentation.

USES OF MARS MODELS

MARS models are most useful in high dimensional spaces where there is little substantive reason to assume linearity or a low-level polynomial fit. They combine very flexible fitting of the relationship between independent and dependent variables with model selection methods that can sharply reduce the dimension of the model. SAS' implementation of these models extends them to dependent variables in the exponential family. One potential problem with any method that is so data-driven and flexible is that of fitting that takes too much advantage of the particular data set being analyzed (although Friedman, 1991 shows some evidence that this is not as much of a problem with MARS models as might be supposed). The usual ways to ameliorate this are either dividing the data into training, test and (sometimes) validation subsets or using crossvalidation. ADAPTIVEREG allows both.

PROC ADAPTIVEREG

The basic syntax for ADAPTIVEREG is similar to other regression PROCs:

```

PROC ADAPTIVEREG <options>;
  BY variables;
  CLASS variables </ options>;
  FREQ variable;
  MODEL dependent <(options)> = <effects></ options>;
  OUTPUT <OUT=SAS-data-set> <keyword <(keyword-options )>
    <=name>> <keyword <(keyword-options )> <=name>>;
  PARTITION <options>;
  SCORE <DATA=SAS-data-set> <OUT=SAS-data-set><keyword <=name>><keyword <=name>>;
  WEIGHT variable;
RUN;

```

Unfamiliar portions include the options on the PROC statement and the MODEL statement and whole of the PARTITION and SCORE statements. Full coverage is in the SAS documentation, here I give only some highlights.

On the PROC statement, you can specify three different data sets: DATA =, TESTDATA = and VALDATA = . You can specify which plots you want and you can specify the number of threads (if you have that ability on your computer). You can also specify computational methods, although (as usual) the defaults are generally good. Unfamiliar options on the MODEL statement include ADDITIVE (which specifies a model with no interactions), ALPHA (which controls knot selection), KEEP = (this allows you to force certain variables into the model), MAXBASIS and MAXORDER (which limit the number of basis functions and order of interactions). The PARTITION statement provides a number of methods for dividing the data into training, test and validation sets. The SCORE statement is something like OUTPUT.

EXAMPLE: MILEAGE DATA

Kuhfeld & Cai (2013) analyzed automobile mileage data provided by the University of California at Irvine Machine Learning Data Depository (Asuncion & Newman, 2007). I provide a slightly different and further analysis of that data set. The first step is to get the data:

```

%let base = http://archive.ics.uci.edu/ml/machine-learning-databases;

proc format;
  invalue q '?' = .;
run;

data autoMPG;
  infile "&base/auto-mpg/auto-mpg.data" device = url expandtabs lrecl = 120 pad;
  input MPG Cylinders Displacement HP :q. weight acceleration year origin name $50.;
  name = propcase(left(tranwrd(name, '"', '')));
run;

```

Initial analysis showed that there were a very few cars with either 3 or 5 cylinders while the vast majority had an even number; in addition, these cars with an odd number of cylinders appeared different from the other in terms of gas mileage; they were deleted from the data for all further analyses. Two MARS models were considered: An additive model and a model allowing 2 way interactions to enter. The latter provided only a tiny increase in R^2 and therefore the former is used henceforth:

```

proc adaptivereg data = autompg2 plots = all details = bases seed = 12345;
  class origin;
  model mpg = cylinders displacement weight acceleration year origin/additive;
  output out = MPGADAPtointer2 p = predADAPtointer2 r = residADAPtointer2;
  partition fraction (test= .3);
run;

```

The additive model had an R^2 of 0.91; four variables were included in the final model: Year, weight, acceleration and displacement. The first two were the most important by far. Year had 4 knots (at 1973, 79, 80 and 81), weight had 2 knots (at 2070 and 3449 pounds), acceleration had 1 knot at 20.5 and displacement had 1 knot at 122.

One potential problem with computer intensive models is that of overfitting. PROC ADAPTIVEREG allows us to check for this by using training and test data sets. One way to do this is to include the PARTITION statement (as above). We can then compare the absolute values of the residuals of the two models.

```

data validate;
  set mpgadaptnointer2;
  absresidadaptnointer2 = abs(residadaptnointer2);
run;

```

```

proc ttest data = validate;
  class _ROLE_;
  var absresidadaptnointer2;
run;

```

The mean absolute residual for the training data was 1.84 and for the test data it was 2.02, which is both small and nonsignificant ($t = 0.98$, $df = 389$, $p = 0.33$).

The model was

$$17.61 + 0.005724 * \max(3449 - \text{weight}, 0) + 0.90 * \max(\text{year} - 73, 0) + 1.45 * \max(\text{accel} - 20.5, 0) + 0.90 * \max(122 - \text{displacement}, 0) - 0.86 * \max(\text{year} - 80, 0) + 4.30 * \max(\text{year} - 79, 0) + 4.63 * \max(\text{year} - 81, 0) - 0.0028 * \max(\text{weight} - 2700, 0)$$

Needless to say, this is not intuitively clear. The first term is simply the intercept, each succeeding term includes an indicator. So the second term says to add $0.0057 * (3449 - \text{weight})$, but only if weight is less than 3449 pounds; the last term also involves weight and says to subtract $0.0028 * (\text{weight} - 2700)$ but only if weight is below 2700 pounds. SAS Output includes a chart of the effect of each variable, but I don't find that chart that helpful. An alternative is to generate a data set that covers the entire range of the data on the independent variables that are in the model and then score it:

```

data score;
  do displacement = 60 to 460 by 10;
    do acceleration = 8 to 25 ;
      do year = 70 to 82;
        do weight = 2223 to 3609 by 100;
          output;
        end;
      end;
    end;
  end;
run;

```

```

proc adaptivereg data = autompg2 plots = all details = bases;
  class origin;
  model mpg = cylinders displacement weight acceleration year origin/additive;
  score data = score out = scoreout;
run;

```

we can then use this to create graphs that are 'sliced'. Since, in this case, the two most important variables were year and weight, we make contour graphs of the predicted value on those two variables, sliced by the other variables. This requires use of the graph template language:

```

%let off0 = offsetmin=0 offsetmax=0 linearopts=(thresholdmin=0 thresholdmax=0);
proc template;
  define statgraph surface;
    dynamic _title _x _y;
    begingraph / designwidth=defaultDesignHeight;
      entrytitle _title;
      layout overlay / xaxisopts=(&off0) yaxisopts=(&off0);
        contourplotparm z=pred y=_y x=_x / gridded=FALSE;
      endlayout;
    endgraph;
  end;
run;

```

```
end;  
run;
```

to create a template. We then slice the data by the knots of the other two variables, creating 4 data sets:

```
data scoreout1;  
set scoreout;  
where displacement < 122 and acceleration < 20.5;  
run;
```

```
data scoreout2;  
set scoreout;  
where displacement < 122 and acceleration ge 20.5;  
run;
```

```
data scoreout3;  
set scoreout;  
where displacement ge 122 and acceleration < 20.5;  
run;
```

```
data scoreout4;  
set scoreout;  
where displacement ge 122 and acceleration ge 20.5;  
run;
```

then running SGRENDER on each data set to create four graphs, e.g.

```
title;  
proc sgrender data = scoreout1 template = surface;  
dynamic _title = "Pred MPG where disp. < 122, accel < 15.5"  
_x = "weight" _y = "year";  
run;
```

and similarly for other combinations, generating figures 8 to 11

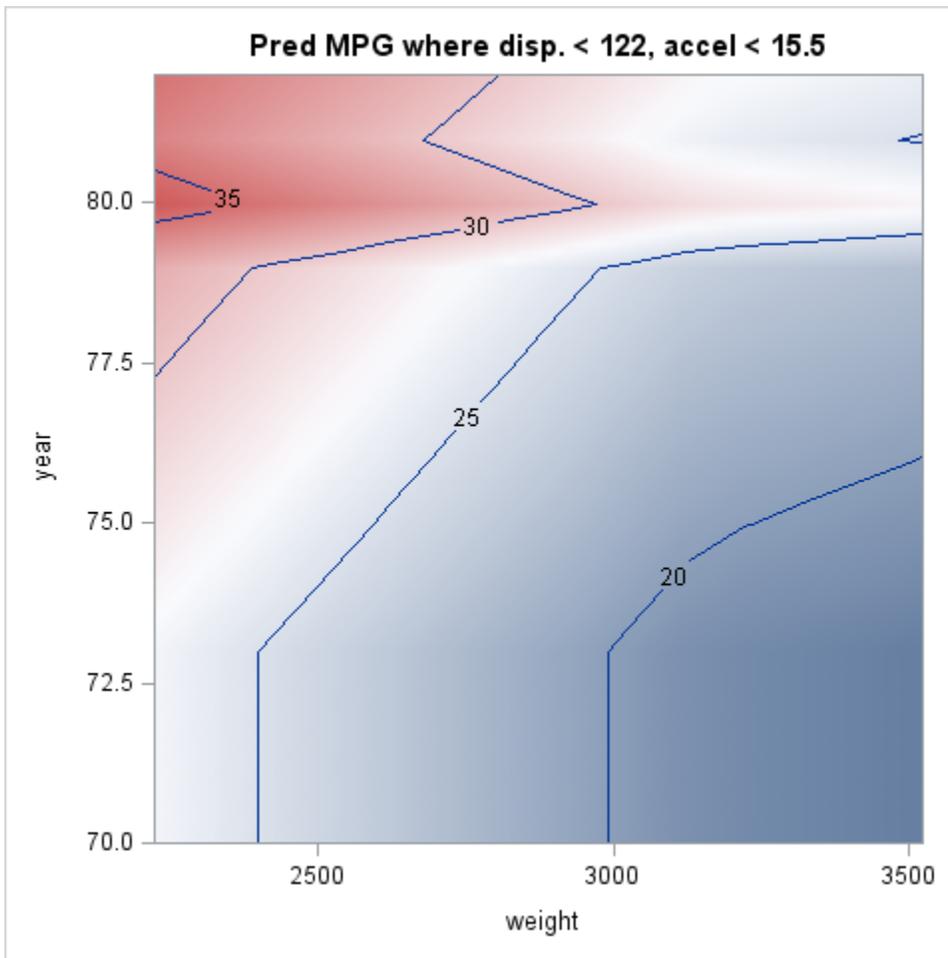


Figure 8: MARS model for low displacement and low acceleration

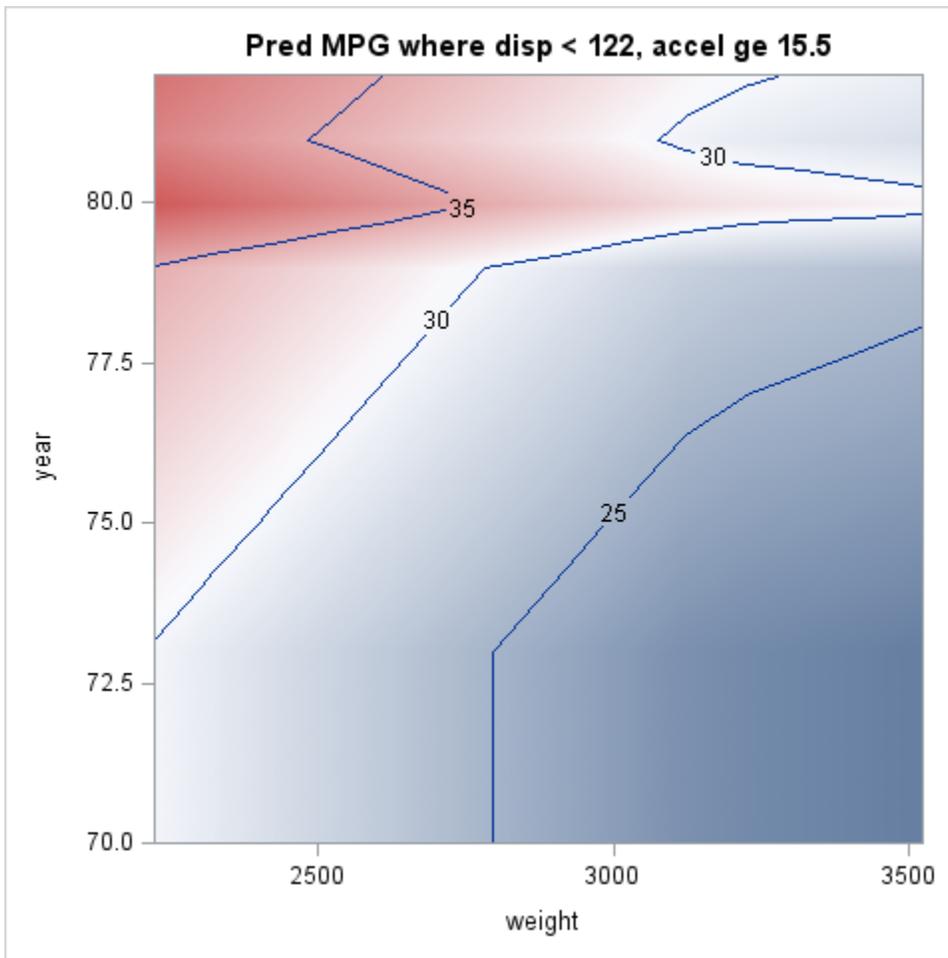


Figure 9: MARS model for low displacement and high acceleration

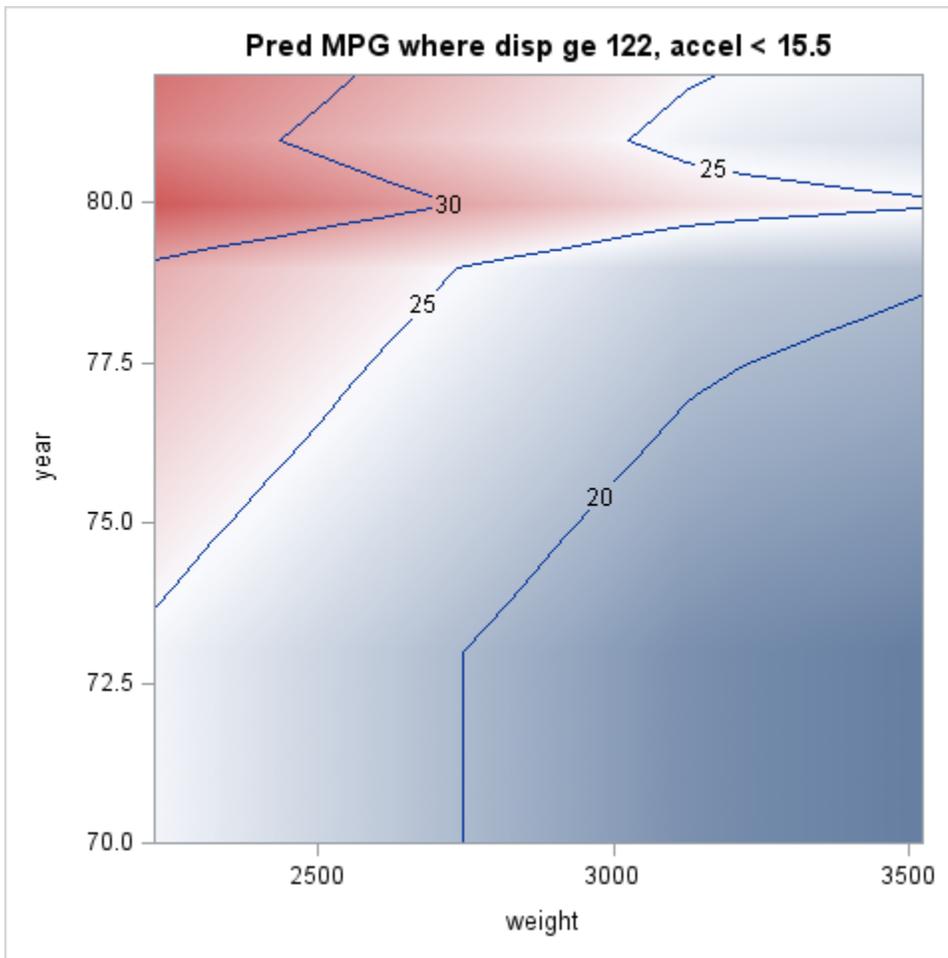


Figure 10: MARS model for high displacement and low acceleration

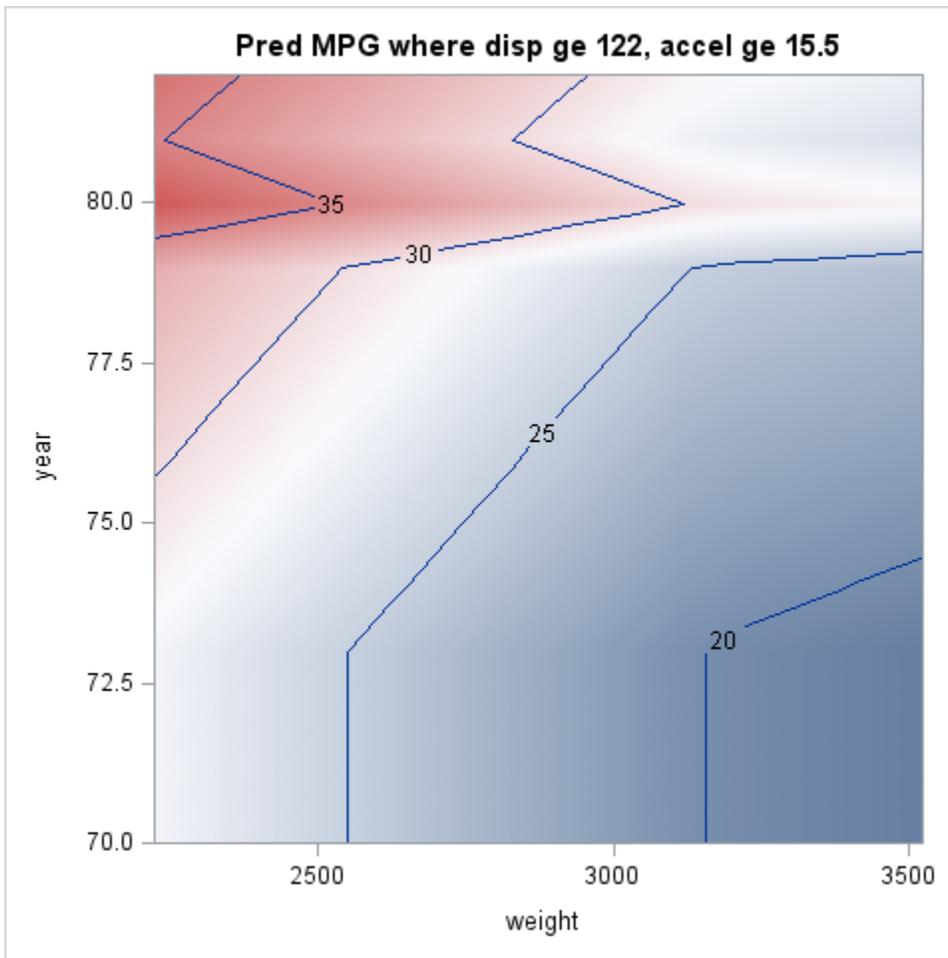


Figure 11: MARS model for high displacement and high acceleration

TRANSFORMING VARIABLES AND PROC TRANSREG

VARIOUS TRANSFORMATIONS

Sometimes it makes sense to transform one or more variables. Of course, it is possible to define new variables in the data step and then use the transformed variables in a regression, but PROC TRANSREG offers many options and allows automation of some tasks.

PROC TRANSREG

PROC TRANSREG is very versatile and has many uses beyond regression. Here I will only discuss its use in regression models with continuous dependent variables. PROC TRANSREG allows a huge variety of transformations of the dependent and independent variables. One particular strength is its options for ordinal independent variables (a very under-examined area). Another is optimal scoring of nominal independent variables.

The basic syntax for TRANSREG is

```
PROC TRANSREG <DATA=SAS-data-set><PLOTS=(plot-requests)>
<OUTTEST=SAS-data-set> <a-options> <o-options> ;
  MODEL <transform(dependents </ t-options>)>
  <transform(dependents </ t-options>) ...=>
  transform(independents </ t-options>)
  <transform(independents </ t-options>) ...> </ a-options> ;
  OUTPUT <OUT=SAS-data-set> <o-options> ;
  ID variables ;
  FREQ variable ;
  WEIGHT variable ;
  BY variables ;
RUN;
```

Most of these function in the usual way; the essential new parts of TRANSREG are the transforms in the MODEL statement. There are too many of these to cover here; see the SAS documentation.

EXAMPLE

Using the same mileage data set we can run:

```
proc transreg data = autompg2 plots = all maxiter = 200;
  id name;
model identity(mpg) = spline(displacement weight acceleration) opscore(origin year cylinders);
output out = mpgtransreg predicted residuals ;
run;
```

This uses splines for the continuous independent variables (displacement, weight and acceleration) and optimal scoring for the discrete variables (origin, year and cylinders). The plots of the transformations (see figures 12 and 13 for continuous and discrete variables, respectively) show that displacement has a nonmonotonic relationship with mileage and that it matters most at lower levels of displacement; weight has a monotonic relationship, but matters very little at higher weights; acceleration has a slightly nonmonotonic relationship but matters at higher levels (less acceleration). The optimal scoring leaves origin as is, but cylinders is nonmonotonic with 6 cylinder engines better than 4 (but 8 much worse than either). Year is nearly linear and could probably be put in as a continuous variable. These are shown in the following two plots.

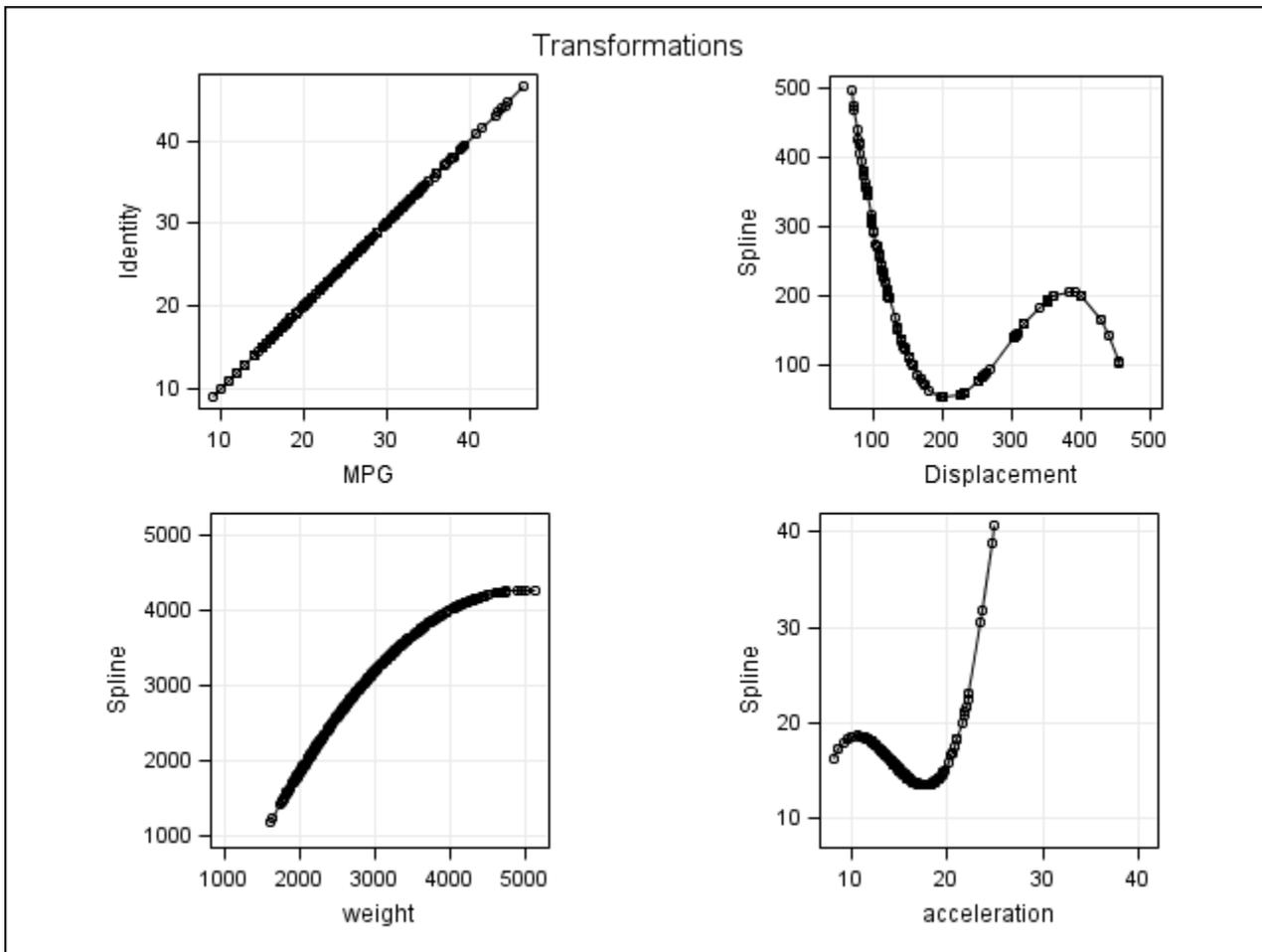


Figure 12: Transformations of continuous variables

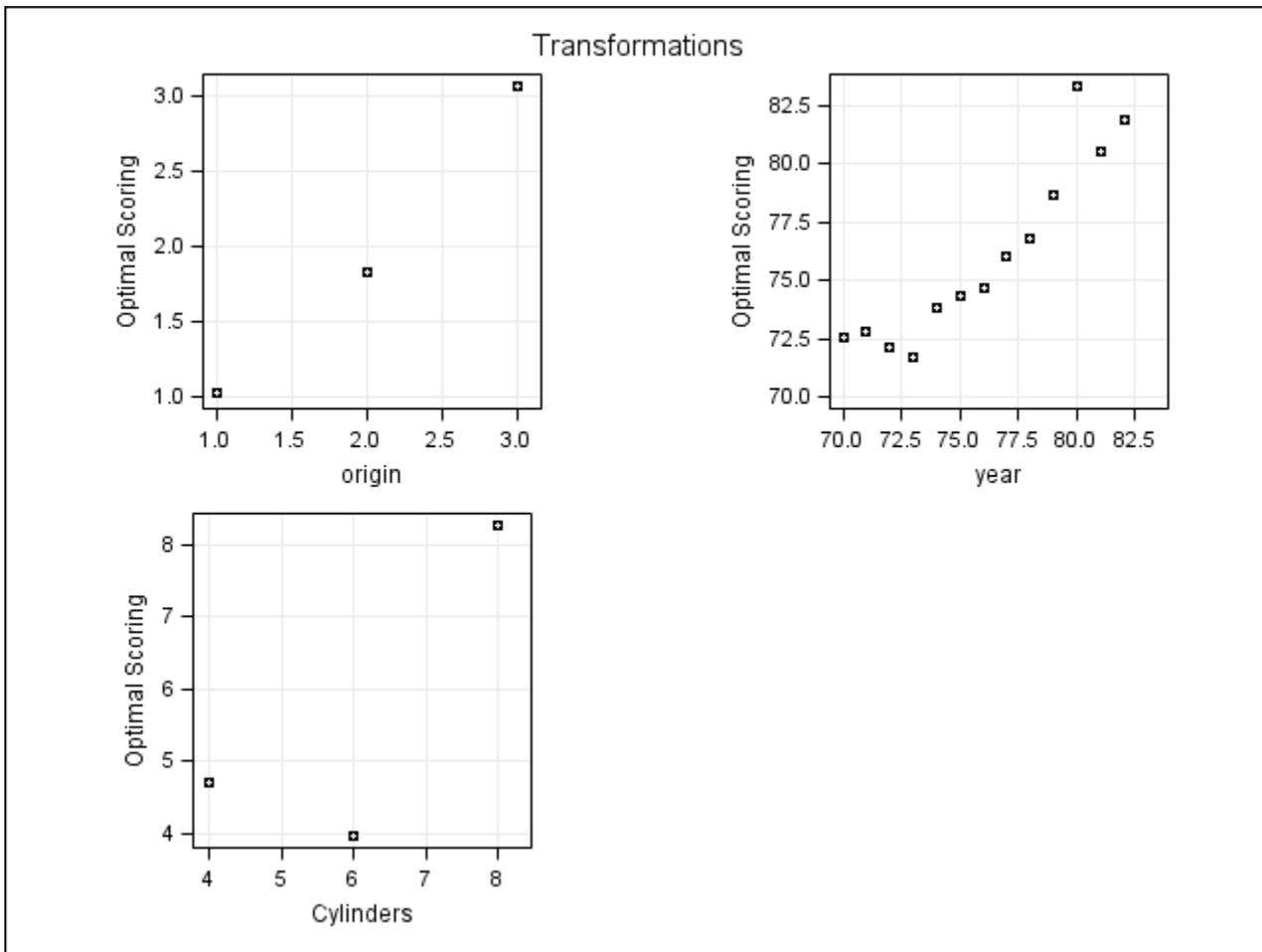


Figure 13: Transformations of categorical variables

SUMMARY AND CONTACT INFO

SUMMARY

Ordinary least squares regression is often useful, but alternatives exist that make fewer assumptions or answer questions that are sometimes more interesting. These methods have become much more practical due to the increasing power and ubiquity of computers. They ought to be more widely used. SAS makes them available in a straightforward manner.

CONTACT INFORMATION

Peter L. Flom
 Peter Flom Consulting, LLC
 515 West End Ave.
 New York, NY 10024
 Phone: (917) 488-7176
 peterflomconsulting@mindspring.com
 Personal webpage: <http://www.statisticalanalysisconsulting.com/>

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. Other brand names and product names are registered trademarks or trademarks of their respective companies.