ABSTRACT

Linear models frequently include terms to reduce bias in parameter estimation and extraneous variation. Large data sets typically contain observations from multiple sources, such as different locations, time periods or product types. If one or more of the model variables is a classification (CLASS) variable, then the required computations might overwhelm memory capacity and disable computation of model coefficients. This problem can be partially solved by using the ABSORB statement in the GLM procedure, which enables estimation of coefficients of non-classification variables. But it does not permit computation of linear combinations involving coefficients of CLASS variables, such as predicted values or comparisons of levels of among CLASS variables. This paper demonstrates a computational method that can be carried out in data steps that accomplishes the same objectives as the ABSORB statement, but enables computation of predicted values and residuals. An illustrative data set contains prices of machine products that were sold to multitudes of customers. The data were collected to estimate the effects of product cost, demand and a possible change in economic environment on price, adjusted for product effects. (Data for the example were simulated to represent a real data.) The HPmixed procedure has capabilities that overcome the shortcomings of the GLM ABSORB statement, but uses more computational resources, as is briefly shown.

INTRODUCTION

Econometric models are commonly used in antitrust litigation to provide evidence of collusive and illegal price fixing. In one such case data were collected from hundreds of thousands of financial transactions in sales of machine parts. Data from each transaction included a price of the item sold, identification of the type of product, date of the transaction, economic indices of cost to the seller and of demand in the product marketplace. Typically, litigation is the result of a “class action” lawsuit brought by a group of customers against sellers of the products. Usually, the illegal collusive price fixing is alleged to have occurred during a prescribed time segment, called the DAMAGES period. Evidence of collusion is obtained by comparing the actual prices during the damages period with prices that would have occurred had there been no collusion. The latter are called But-for prices; that is, prices that would have been charged in a free market with no collusive activity. Two statistical models of price are constructed, corresponding to the actual prices and the But-for prices. Ordinarily, linear models are used to represent both the actual and But-for prices. The two models can be simultaneously represented by one linear equation, wherein a term or set of terms, distinguishes the representation of the actual prices from the But-for prices. Here are the two models in standard statistical terms. First is the so-called But-for model representing the economic relationship between the prices and effects due to cost and demand factors in a fair market.

But-for model:

\[ y_{ij} = \alpha_i + \beta_{\text{cost}} \cdot x_{\text{cost},ij} + \beta_{\text{demand}} \cdot x_{\text{demand},ij} + e_{ij}, \]

where \( y_{ij} \) = logarithm of price of product \( i \) in transaction \( j \), and

- \( \alpha_i \) = intercept of the regression for Product \( i \),
- \( \beta_{\text{cost}} \) = expected effect of cost index on log price
- \( x_{\text{cost},ij} \) = value of cost index at the time of transaction \( j \),
- \( \beta_{\text{demand}} \) = expected effect of demand index on log price
- \( x_{\text{demand},ij} \) = value of demand index at the time of transaction \( j \),
- \( e_{ij} \) = random error.
Second is the so-called Damages model that includes, in addition to all the terms in the But-for model, a set of terms that represent price changes (damages) due to presumed collusive activity during the class period.

\[ y_{ij} = \alpha_i + \beta_{\text{cost}i} \cdot \text{cost}_{i,j} + \beta_{\text{demand}i} \cdot \text{demand}_{i,j} + \tau \cdot d + \pi_{\text{cost}i} \cdot \text{cost}_{i,j} + \pi_{\text{demand}i} \cdot \text{demand}_{i,j} + e_{ij}, \]

where (in addition to the terms in the But-for model),

\[ d = 1 \text{ if the transaction is in the damages period and } d = 0 \text{ if the transaction is not in the damages period}, \]

\[ \tau = \text{the basal effect of collusion}, \]

\[ \pi_{\text{cost}} = \text{increase in effect of cost index due to collusion}, \]

\[ \pi_{\text{demand}} = \text{increase in effect of demand index due to collusion}. \]

A statistical model for the data in matrix notation is

1. \[ Y = D\alpha + d\delta + X\beta + e, \]

where

- \( Y \) is the vector of LogPrice values,
- \( D \) is the matrix of indicator variables corresponding the Products,
- \( d \) is the vector of indicator variable for presence \((d = 1)\) or absence \((d = 0)\) in the time periods of econometric change,
- \( X \) is the matrix of values of CostIndex and DemandIndex,

and

- \( e \) is the vector of random errors.

The model (1) can be written more compactly using partitioned matrices as

2. \[ Y = U\theta + e, \]

where \( U = [D; \: d; \: X] \) and \( \theta' = [\alpha'; \: \delta'; \: \beta'] \).

The method of Ordinary Least Squares (OLS) is most commonly used to obtain estimates of the parameters in linear models. Pertaining to model (2) the estimates of the parameters in equation (1), would be given by evaluating

3. \[ T = (U'U)^{-1} U' Y. \]

The central topic in this paper concerns making the computation in equation 3 when the matrix \( U'U \) is “large,” generally meaning \( D \) has 3500 or more rows and columns. Inversion of such a large matrix is a huge computational task which GLM cannot accomplish. Instead, GLM reverts to the “absorption,” which does not actually invert \( U'U \) but rather “absorbs” the variation associated with the \( D \) portion of \( U \), which contains the columns associated with the transactions. In this example, there are 81,090 columns in \( D \) one in \( d \), and four in \( X \). Once the columns in \( D \) are absorbed the remaining portion of \( U'U \) only has five rows and five columns.

**METHODOLOGY**

The GLM procedure makes computations for analysis of variance and regressions analysis. These are two of the most important methods for statistical data analysis, and are under the broader grouping of general linear models, hence the term General Linear Model. GLM is one of the oldest procedures in SAS, dating back to the mid-seventies. HPMIXED is much newer and capable of making computations still in the realm of general linear models,
but utilizing computational methods unknown when GLM was developed. Three methods are illustrated; two using GLM and one using HPMIXED.

**GLM WITH THE ABSORB STATEMENT**

A statistical model for the data in terms of SAS proc GLM code is:

```
(1) proc glm data = absorb;
class product;
model LogPrice = Product Period CostIndex Period*CostIndex DemandIndex Period*DemandIndex;
run;
```

The variables names in the model statement are:

- **LogPrice**: Logarithm of price in a transaction
- **Product**: an identification of a particular part
- **Period**: 0 or 1 depending on whether the transaction date was during the period of economic change
- **CostIndex**: is a combined measure of product costs
- **DemandIndex**: is a combined measure of product demand

Output 1 is a printout of the first five observations of the SAS data ABSORB:

<table>
<thead>
<tr>
<th>Obs</th>
<th>Product</th>
<th>LogPrice</th>
<th>Period</th>
<th>CostIndex</th>
<th>CostIndex_DM</th>
<th>DemandIndex</th>
<th>DemandIndex_DM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P00001</td>
<td>9.41222</td>
<td>1</td>
<td>77.0496</td>
<td>11.0348</td>
<td>-1.40432</td>
<td>-1.71539</td>
</tr>
<tr>
<td>2</td>
<td>P00001</td>
<td>8.71628</td>
<td>0</td>
<td>-1.9908</td>
<td>-2.3362</td>
<td>1.57576</td>
<td>-0.41131</td>
</tr>
<tr>
<td>3</td>
<td>P00001</td>
<td>8.99793</td>
<td>0</td>
<td>-1.9534</td>
<td>-2.2987</td>
<td>1.65133</td>
<td>-0.33573</td>
</tr>
<tr>
<td>4</td>
<td>P00001</td>
<td>9.43539</td>
<td>1</td>
<td>90.3732</td>
<td>24.3584</td>
<td>-0.11193</td>
<td>-0.42300</td>
</tr>
<tr>
<td>5</td>
<td>P00002</td>
<td>9.30849</td>
<td>1</td>
<td>43.6825</td>
<td>-22.3323</td>
<td>1.81546</td>
<td>1.50439</td>
</tr>
</tbody>
</table>

There being 81090 Products presents a computational difficulty. The CLASS statements would generate 81090 indicator variables and result in the matrix $D$ having 81095 columns which would cause the GLM procedure to fail. To deal with this issue, GLM has the ABSORB statement that partially solves the problem by “absorbing” effects attributable to the variables in the ABSORB statement.

These statements would be used to invoke the absorption process:

```
(2) proc glm data = absorb;
    absorb Product;
    model LogPrice = Period CostIndex Period*CostIndex DemandIndex Period*DemandIndex / solution;
run;
```

The Absorb statement enables GLM to compute estimates of $\alpha$ and the $\delta$ vectors. But GLM will not compute parameter estimates for the indicator variables created by the ABSORB statement. Consequently, predicted values and residuals cannot be computed because the Product coefficients are not available.
Output 2 is a printout from submitting the statements in (2) showing parameter estimates for the estimated modes:

```
Output 2: Using the ABSORB Statement

The GLM Procedure

Dependent Variable: LogPrice

Source DF Sum of Squares Mean Square F Value Pr > F
Model 81095 587369.2142 7.2430 177.11 <.0001
Error 492552 20142.8397 0.0409
Corrected Total 573647 607512.0539

R-Square Coeff Var Root MSE LogPrice Mean
0.966844 2.411284 0.202225 8.386600
```

```
Source DF Type I SS Mean Square F Value Pr > F
Product 81090 577109.3803 7.1169 174.03 <.0001
Period 1 7226.4284 7226.4284 176708 <.0001
CostIndex 1 2949.5777 2949.5777 72125.9 <.0001
Period*CostIndex_DM 1 65.8201 65.8201 1609.50 <.0001
DemandIndex 1 4.1005 4.1005 100.27 <.0001
Period*DemandIndex_DM 1 13.9072 13.9072 340.07 <.0001
```

```
Source DF Type III SS Mean Square F Value Pr > F
Period 1 26.54776049 26.54776049 649.17 <.0001
CostIndex 1 6.61124018 6.61124018 161.66 <.0001
Period*CostIndex_DM 1 32.25536070 32.25536070 788.74 <.0001
DemandIndex 1 17.39974480 17.39974480 425.48 <.0001
Period*DemandIndex_DM 1 13.90717669 13.90717669 340.07 <.0001
```

```
Parameter Estimate Standard t Value Pr > |t|
Error
Period 0.1773744129 0.00696164 25.48 <.0001
CostIndex 0.0014294476 0.00011242 12.71 <.0001
Period*CostIndex_DM 0.0032004609 0.00011396 28.08 <.0001
DemandIndex 0.0159861182 0.00077501 20.63 <.0001
Period*DemandIndex_DM -0.0150221659 0.00081461 -18.44 <.0001
```

Notice that estimates of Product parameters in the vector \(\alpha\) are not printed due to the statement:

Absorb Product;

Notice that estimates of Product parameters in the vector \(\alpha\) are not printed due to the statement:

Absorb Product;
Those estimates usually are not of inherent interest. However, they would be needed to compute predicted values from the estimated regression model. In fact, predicted values and residuals are not available from PROC GLM when using the ABSORB statement, as is clearly stated in SAS documentation of PROC GLM. Closer inspection of the Type I SS output provides insight into the function of the statement. Look at the portion labelled “Type I SS” and notice the line labelled “Product.” It shows DF = 81090, the number of columns in D. This is the only line of the output pertaining to Products.

The value of the Type I SS = 577,109 is not computed in the same manner as the other SS shown in the output. SAS GLM documentation states it is computed in a manner similar to those used by PROC NESTED; that is, standard analysis of variance computations. Specifically, let \( a_i \) be the mean of the LogPrice values from Product 1 (see Output 1),

\[
a_1 = (9.412 + 8.716 + 8.998 + 9.435)/4 = 9.140
\]

and similarly for Product 2, 3, ..., 80190. Then (uncorrected) Type I SS = \( \sum (s_1^2 + \ldots + s_{8117822}) \) = 577,109, where \( s_i^2 = n_i a_i^2 \). You can think of \( n_i a_i^2 \) as being the SS associated with the “intercept” for Product i, which is technically correct, and thus the degrees of freedom for Product is the number of Products (10810).

GLM with Long-hand Absorption (LHA):

The purpose of this article is to present a method (referred to as “LHA” that basically duplicates what the ABSORB statement accomplishes, but retains information in a way that enables computation of predicted values and residuals. Predicted values require estimates of all the parameters in the model, namely those in the vectors \( \alpha, \delta \), and \( \beta \). Estimates of parameters in \( \alpha \) are not computed due to the ABSORB statement. But those estimates would be given by the Product means, \( a_1, a_2, \ldots, a_{10810} \). The LHA process utilizes preliminary computations in data steps that essentially transform the indicators that are “absorbed,” but does not require inversion of a matrix. Basically, in terms of values used in the previous section, it explicitly computes quantities of the form:

\[
9.412 - 9.140 = 0.272, 8.716 - 9.140 = -0.424, 8.998 - 9.140 = -0.142, 9.435 - 9.140 = 0.295, 
\]

in a data step. These values are residuals that would result from regressing LogPrice on the Product indicator variables. The illustrated process pertains only to 0-1 indicator variables, such as those created for a variable in a CLASS statement.

Output 3 shows a data set with indicator variables for the first three products:

<table>
<thead>
<tr>
<th>Obs</th>
<th>LogPrice</th>
<th>Product</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>Period</th>
<th>CostIndex</th>
<th>DemandIndex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9.41222</td>
<td>P00001</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>77.0496</td>
<td>-1.40432</td>
</tr>
<tr>
<td>2</td>
<td>8.71628</td>
<td>P00001</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1.9908</td>
<td>1.57576</td>
</tr>
<tr>
<td>3</td>
<td>8.99793</td>
<td>P00001</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1.9534</td>
<td>1.65133</td>
</tr>
<tr>
<td>4</td>
<td>9.43539</td>
<td>P00001</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>90.3732</td>
<td>-0.11193</td>
</tr>
<tr>
<td>5</td>
<td>9.30849</td>
<td>P00002</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>43.6825</td>
<td>1.81546</td>
</tr>
</tbody>
</table>

The technique used to compute the parameter estimates is based on a well-known two-step process. Consider a statistical model with a dependent variable \( y \) and two sets of independent variables, denoted collectively in matrices \( D \) and \( T \), with corresponding parameter vectors \( \alpha \) and \( \tau \). The model equation is
\[ y = D\alpha + T\tau + e. \]

Relative to model 1. \((Y = D\alpha + d\delta + X\beta + e)\), \(T\tau\) in model 4. Is equal to \(d\delta + X\beta\) in model 1.

The first step is to regress the variable \(y\) and the variables in \(T = \{t_0, \ldots, t_l\}\) on the set of variables in \(D = \{d_0, \ldots, d_k\}\), and collect the residuals from those regressions. (This is the step that is not explicitly carried out by the ABSORB statement.) It turns out, since each of the columns in \(D\) consist of 0’s and 1’s, that these residuals can be computed directly without explicitly fitting a linear regression model. In fact, the values 0.272, -0.424, -0.142 and 0.295 in 5. are the residuals corresponding to Product P00001. In words, the coefficients for the regression on the dummy variables would be means of the variable being regressed (regressors), such as the number 0.272, for Product P00001. In turn, the residuals would be the values of the regressor minus the means of the values of the regressors corresponding to the Product group containing the value of the regressor, such as the numbers 9.412, 8.716, 8.998 and 9.453.

Assume the residuals from the regression have been obtained and entered into two sets of variables \(\{\text{Res}_y\}\) and \(\{\text{Res}_{t_1}, \ldots, \text{Res}_{t_n}\}\). The second step is to regress \(\text{Res}_y\) on \(\text{Res}_{t_1}, \ldots, \text{Res}_{t_m}\). The coefficients of this regression will be equal to the coefficient estimates for the variables in \(T\) as if the entire model (5) had been fitted, although no coefficients for the variables in \(D\) will be explicitly produced.

These SAS statements create variables needed to perform the computations in SAS data steps are:

```sas
data Absorb; set Absorb ;
    Per_CstInd=Period*CostIndex;
    Per_DemInd=Period*DemandIndex;
run;
```

The MEANS procedure is used to compute Product means of model variables and save the means in a data set named Prodmeans:

```sas
proc means data=Absorb noprint;
    by Product ;
    var LogPrice Period CostIndex Per_CstInd DemandIndex Per_DemInd;
    output out=Prodmeans mean=LogPr_Avg Period_Avg CstInd_Pavg Per_CstInd_Avg DemInd_Pavg Per_DemInd_Avg n=n;
run;
```

Output 4 shows a data set containing the means of all the quantitative variables in the model (note the suffix “Avg” on all these variable names):

<table>
<thead>
<tr>
<th>Ob</th>
<th>Product</th>
<th>LogPr_Av</th>
<th>Period_Av</th>
<th>CstInd_Av</th>
<th>Per_CstInd_dm_Av</th>
<th>DemInd_Av</th>
<th>Per_DemInd_dm_Av</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>P00001</td>
<td>9.14045</td>
<td>0.50000</td>
<td>40.8697</td>
<td>8.84831</td>
<td>0.42771</td>
<td>-0.53460</td>
</tr>
<tr>
<td>2</td>
<td>P00002</td>
<td>9.25691</td>
<td>0.71429</td>
<td>52.9647</td>
<td>4.91834</td>
<td>0.90442</td>
<td>0.15567</td>
</tr>
<tr>
<td>3</td>
<td>P00003</td>
<td>9.18200</td>
<td>0.25000</td>
<td>14.2963</td>
<td>0.02548</td>
<td>0.87530</td>
<td>-0.68906</td>
</tr>
<tr>
<td>4</td>
<td>P00004</td>
<td>9.05492</td>
<td>0.57143</td>
<td>40.7255</td>
<td>3.62713</td>
<td>1.26628</td>
<td>0.26018</td>
</tr>
<tr>
<td>5</td>
<td>P00005</td>
<td>8.68597</td>
<td>0.66667</td>
<td>43.6838</td>
<td>0.71913</td>
<td>1.64128</td>
<td>0.43219</td>
</tr>
</tbody>
</table>
Next, use these statements to merge the data set Prodmeans with the original data set Absorb to obtain a new data set named Obsmeans:

```plaintext
data Obsmeans; merge Absorb Prodmeans; by Product;
run;
```

The first 5 observations of the data set Obsmeans are shown in Output 5:

<table>
<thead>
<tr>
<th>Ob st</th>
<th>LogPr_Avg</th>
<th>Period_Avg</th>
<th>CstInd_Avg</th>
<th>Per_CstInd_dm_Avg</th>
<th>DemInd_Avg</th>
<th>Per_DemInd_dm_Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 P00001</td>
<td>9.14045</td>
<td>0.50000</td>
<td>40.8697</td>
<td>8.84831</td>
<td>0.42771</td>
<td>-0.53460</td>
</tr>
<tr>
<td>2 P00001</td>
<td>9.14045</td>
<td>0.50000</td>
<td>40.8697</td>
<td>8.84831</td>
<td>0.42771</td>
<td>-0.53460</td>
</tr>
<tr>
<td>3 P00001</td>
<td>9.14045</td>
<td>0.50000</td>
<td>40.8697</td>
<td>8.84831</td>
<td>0.42771</td>
<td>-0.53460</td>
</tr>
<tr>
<td>4 P00001</td>
<td>9.14045</td>
<td>0.50000</td>
<td>40.8697</td>
<td>8.84831</td>
<td>0.42771</td>
<td>-0.53460</td>
</tr>
<tr>
<td>5 P00002</td>
<td>9.25691</td>
<td>0.71429</td>
<td>52.9647</td>
<td>4.91834</td>
<td>0.90442</td>
<td>0.15567</td>
</tr>
</tbody>
</table>

Now create a new data set named Resids whose observations contain values in Absorb minus the corresponding value in Obsmeans using this code:

```plaintext
data Resids; set Obsmeans;
  LogPr_Res=LogPrice-LogPr_Avg;
  PeriodRes=Period-Period_Avg;
  CstIndRes=CostIndex-CstInd_Avg;
  PeriodCstIndRes=Per_CstInd-Per_CstInd_Avg;
  DemIndRes=DemandIndex-DemInd_Avg;
  PeriodDmdIndRes=Per_DemInd-Per_DemInd_Avg;
run;
```

Output 6 contains the first five of the data set Resids:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P00001</td>
<td>0.27176</td>
<td>0.50000</td>
<td>36.1799</td>
<td>2.1865</td>
<td>-1.83203</td>
<td>-1.18079</td>
</tr>
<tr>
<td>2</td>
<td>P00001</td>
<td>-0.42418</td>
<td>-0.50000</td>
<td>-42.8604</td>
<td>-8.8483</td>
<td>1.14805</td>
<td>0.53460</td>
</tr>
<tr>
<td>3</td>
<td>P00001</td>
<td>-0.14252</td>
<td>-0.50000</td>
<td>-42.8230</td>
<td>-8.8483</td>
<td>1.22362</td>
<td>0.53460</td>
</tr>
<tr>
<td>4</td>
<td>P00001</td>
<td>0.29494</td>
<td>0.50000</td>
<td>49.5035</td>
<td>15.5101</td>
<td>-0.53964</td>
<td>0.11159</td>
</tr>
<tr>
<td>5</td>
<td>P00002</td>
<td>0.05158</td>
<td>0.28571</td>
<td>-9.2822</td>
<td>-27.2506</td>
<td>0.91104</td>
<td>1.34872</td>
</tr>
</tbody>
</table>

Finally, regress LogPr_Res on the other residual values using the GLM statements

```plaintext
(3) Proc GLM data=Resids;
    output out=Pred p=LogPr_ResHat r=LogPr_ResRes;
Run;
```
Output 7 shows results from statements (3):

### Output 7: Results of the regression of LogPr_Res on RHS residuals

The GLM Procedure

Dependent Variable: LogPr_Res

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Sum of Squares</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>5</td>
<td>10259.83390</td>
<td>2051.96678</td>
<td>58437.4</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Error</td>
<td>573642</td>
<td>20142.83972</td>
<td>0.03511</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>573647</td>
<td>30402.67362</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R-Square  Coef Var  Root MSE  LogPr_Res Mean
0.337465  3.3577E16  0.187387  5.5808E-16

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Type I SS</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeriodRes</td>
<td>1</td>
<td>7226.428400</td>
<td>7226.428400</td>
<td>205799</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>CstIndRes</td>
<td>1</td>
<td>2949.577744</td>
<td>2949.577744</td>
<td>84000.2</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>PeriodCstIndRes</td>
<td>1</td>
<td>65.820116</td>
<td>65.820116</td>
<td>1874.47</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>DemIndRes</td>
<td>1</td>
<td>4.100460</td>
<td>4.100460</td>
<td>116.78</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>PeriodDmdIndRes</td>
<td>1</td>
<td>13.907177</td>
<td>13.907177</td>
<td>396.06</td>
<td>&lt;.0001</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Type III SS</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeriodRes</td>
<td>1</td>
<td>26.54776049</td>
<td>26.54776049</td>
<td>756.05</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>CstIndRes</td>
<td>1</td>
<td>6.61124018</td>
<td>6.61124018</td>
<td>188.28</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>PeriodCstIndRes</td>
<td>1</td>
<td>32.25536071</td>
<td>32.25536071</td>
<td>918.59</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>DemIndRes</td>
<td>1</td>
<td>17.39974480</td>
<td>17.39974480</td>
<td>495.52</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>PeriodDmdIndRes</td>
<td>1</td>
<td>13.90717669</td>
<td>13.90717669</td>
<td>396.06</td>
<td>&lt;.0001</td>
</tr>
</tbody>
</table>

| Parameter             | Estimate  | Standard Error | t Value | Pr > |t| |
|-----------------------|-----------|----------------|---------|------|---|
| Intercept             | 0.00000000000 | 0.00024741 | 0.00  | 1.0000 |
| PeriodRes             | 0.1773744129 | 0.00645085 | 27.50  | <.0001 |
| CstIndRes             | 0.0014294476 | 0.00010418 | 13.72  | <.0001 |
| PeriodCstIndRes       | 0.0032004609 | 0.00010560 | 30.31  | <.0001 |
| DemIndRes             | 0.0159861182 | 0.00071814 | 22.26  | <.0001 |
| PeriodDmdIndRes       | -0.0150221659 | 0.00075484 | -19.90 | <.0001 |

Compare the output from statements (4) with that from statements (6), and see that parameter estimates are identical, with a minor caveat. Output from (4) shows no intercepts due to the use of the Absorb statement. As noted earlier, that is because the 81090 indicator variables are implicitly in the model. Each of those indicator variables corresponds to one and only one Product. The indicator equals 0 for all observations except those corresponding to the one Product for which the indicator has values of 1. (See output 3). Thus, if all the 81090
indicator variables were added together, the result would have the value 1 in each observation, as would an
“indicator” for an intercept. All you see in Output 2 is the single row labeled “Product” that tells you there are
81090 Products each with 1 DF. If you added all these columns related to Products, the sum would be a column
with a 1 in all 81090 rows. So that’s a long explanation of why no other “intercept” is in the model. All information
in Output 2 below the row just discussed is identical to that found in Output 7. But Output 7 contains one row not
found in Output 2), the row labeled “Intercept” whose value is zero out to ten decimal places. The correct value is
exactly 0 because all the variables in the model are residuals, each of has mean 0.

Moreover, predicted values and residuals are available in the data set Pred. The GLM code in (3) above contains
the statement in the GLM code (5) that creates the predicted values and residuals, namely

```
output out=Pred p=LogPr_ResHat r=LogPr_ResRes;
```

which produces predicted values and residuals and outputs them to a new data set named Pred. The data set
named Resids contains the variables named LogPr_ResHat and LogPr_ResRes. These respectively contain predicted
values and residuals from the regression. These variables are not computed when using the ABSORB statement,
which reveals the major benefit of the methods presented in this paper. A prime use is to construct graphs of the
type commonly presented to show results of the regression analysis. The predicted values and residuals are found
in the variables LogPr_ResHat and LogPr_ResRes. These variable names derive from the name of the name of the
dependent variable in the regression, LorPr_Res. “Res” in the variable name created from the computations
leading up to Table 6.

---

Output 8: Data set Pred

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P00001</td>
<td>36.1799</td>
<td>2.1865</td>
<td>-1.83203</td>
<td>-1.18079</td>
<td>0.13585</td>
<td>0.13591</td>
</tr>
<tr>
<td>2</td>
<td>P00001</td>
<td>-42.8604</td>
<td>-8.8483</td>
<td>1.14805</td>
<td>0.53460</td>
<td>-0.16795</td>
<td>-0.25623</td>
</tr>
<tr>
<td>3</td>
<td>P00001</td>
<td>-42.8230</td>
<td>-8.8483</td>
<td>1.22362</td>
<td>0.53460</td>
<td>-0.16669</td>
<td>0.02417</td>
</tr>
<tr>
<td>4</td>
<td>P00001</td>
<td>49.5035</td>
<td>15.5101</td>
<td>-0.53964</td>
<td>0.11159</td>
<td>0.19879</td>
<td>0.09615</td>
</tr>
<tr>
<td>5</td>
<td>P00002</td>
<td>-9.2822</td>
<td>-27.2506</td>
<td>0.91104</td>
<td>1.34872</td>
<td>-0.05550</td>
<td>0.10708</td>
</tr>
</tbody>
</table>

Output 8 shows the list of variables in Output 6 that were used in the regression analysis displayed in Output7, and
two variables that were produced by the regression analysis in Output 7, namely LogPr_ResHat and LogPr_ResRes.
Please pardon the complicated notation for these variables. Just remember that LogPr_Res is the name of the
dependent variable in the analysis from Output 7. Then LogPr_ResHat is the predicted value from that regression
and LogPr_ResRes is the residual value from the regression.

Predicted values and residuals are largely used to construct graphs and compute diagnostic measures of model fit
and validity.
Output 9a shows a plot of LogPr_ResHat versus the variable YYMM, whose values are of month and year. This plot reveals features of the predicted values over the years 2000 – 2018. Predicted values follow a trend that changes only modestly over years 2000 - 2007, increases steeply over 2008 - 2012 and then changes jaggedly over 2013 – 2018.
Output 9b shows a plot of LogPr_ResRes versus YYMM. It reveals features of the variability in the residuals. A perfectly fitting model would produce residuals that (1) follow about the same range of vertical variation over the years and (2) also have a symmetric vertical pattern over the years. Output 9b shows the residuals meet criterion (1) but not quite as well criterion (2) because of being more densely packed along the lower edge of the plot than along the upper edge.

**Methodology: HPMixed**

The HPMixed procedure is much newer than GLM. It can be used to obtain all these results illustrated in the previous two sections; GLM with ABSORB statement and GLM with longhand absorption. HPMixed also has mixed model methods and more, but at the cost of much more cpu time. For example, the relatively small analysis shown here executed in 12 minutes by HPMixed, compared with only a few seconds by either of the GLM-based methods. The difference increases quickly as more levels of the absorbed variables increase. About 50% more levels results in cpu of more than an hour of cpu by HPMixed, but only a few more seconds by the method presented here. The difference is due to the necessity of essentially inverting an enormous matrix by HPMixed, but the complex computations using the present method increase only minutely due to only using simple arithmetic for the data step manipulations. In the end, the choice comes down to the nature of the problem at hand.

Here are statements for using HPMixed in the present problem:

```plaintext
Proc HPmixed Data= Absorb ;
   Class Product;
   model LogPrice = Product Period CostIndex
                   Period*CostIndex DemandIndex Period*DemandIndex ;
   estimate 'Period' Period 1;
```

![Output 9b: Plot of LogPrice Residuals vs Year and Month](image)
estimate 'CostIndex' CostIndex 1;
estimate 'Period*CostIndex' Period*CostIndex 1;
estimate 'DemandIndex' DemandIndex 1;
estimate 'Period*DemandIndex' Period*DemandIndex 1;
Run;

Notice the use of "estimate" statements to obtain the desired parameter estimates.

Output 10 shows results from the HPMixed procedure.

Output 10: HPMixed

The HPMIXED Procedure

Model Information
Data Set WORK.ABSORB
Response Variable LogPrice
Estimation Method Restricted Maximum Likelihood (REML)
Degrees of Freedom Method Residual

Class Level Information

Class Levels Values
Product 81091 P00001 P00002 P00003 P00004 P00005 P00006 P00007 P00008 P00009 P00010 P00011 P00012 P00013 P00014 P00015 P00016 P00017 P00018 P00019 P00020 ...

Number of Observations Read 573648
Number of Observations Used 573648

Dimensions
G-side Cov. Parameters 0
R-side Cov. Parameters 1
Columns in X 81097
Columns in Z 0
Subjects (Blocks in V) 1

Analysis of Variance

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Sum of Squares</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>81095</td>
<td>587369</td>
<td>7.242977</td>
<td>177.11</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Error</td>
<td>492552</td>
<td>20143</td>
<td>0.040895</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>573647</td>
<td>607512</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Covariance Parameter Estimates

<table>
<thead>
<tr>
<th>Cov Parm</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residual</td>
<td>0.04089</td>
</tr>
</tbody>
</table>
Fit Statistics

-2 Res Log Likelihood: -24569
AIC (Smaller is Better): -24567
AICC (Smaller is Better): -24567
BIC (Smaller is Better): -24556
CAIC (Smaller is Better): -24555
HQIC (Smaller is Better): -24564
R-Square: 0.96684
Root MSE: 0.20222
Coeff Var: 2.41128

Estimates

| Label                | Estimate | Standard Error | DF | t Value | Pr > |t|   |
|----------------------|----------|----------------|----|---------|-------|-----|
| Period               | 0.1774   | 0.006962       | 493E3 | 25.48   | <.0001 |
| CostIndex            | 0.001429 | 0.000112       | 493E3 | 12.71   | <.0001 |
| Period*CostIndex_dm  | 0.003200 | 0.000114       | 493E3 | 28.08   | <.0001 |
| DemandIndex          | 0.01599  | 0.000775       | 493E3 | 20.63   | <.0001 |
| Period*DemandIndex_dm| -0.01502 | 0.000815       | 493E3 | -18.44  | <.0001 |

Results from HPMixed are very close to those of GLM with the ABSORB statement and GLM with the LHA computations. Given the reason for not using GLM with the ABSORB statement is the need to gain access to predicted values and values, the choice is between GLM with Long-hand Absorption and HPMixed. If the user is only about convenience of running the code, then HPMixed clearly is best. But if execution time is a concern and the user has time to invest the overhead in adapting the then is viable, particularly if the user wished to run the code frequently. Relative to the example illustrated in this paper, GLM with LHA used 0.53 seconds and HPMixed used 10 minutes and 39 seconds.

One other aspect of the choice between the options concerns there are two or more class variables. This is no problem with HPMixed, but the execution time may be greatly increased. GLM with LHA can be used, but the class variables would have to be combined into one variable. For example, it the two variables are A B, then and new variable could be constructed, say named AB, with its levels being the combinations of all levels of A with all levels of B. This is relatively easy to do, but it creates a large number of levels. The combined DF for the variable AB would be ab, where a is the number of levels of a and b is the number of levels of b. That is an increase of ab - a - b = (a -1)(b - 1) -1, which is the degrees of freedom for A*B interaction plus 1. If interaction is “significant” in whatever manner the user considers important, then it should be accommodated. If interaction is not significant, then ideally it should not be in the model. Having interaction in the model when it is not significant is not harmful except for wasting degrees of freedom. And a shortage of degrees of freedom is not a concern if the number of levels is large.

On a final note regarding absorption: This topic is not treated extensively in statistics courses. But, at least in earlier times, it was of keen interest in fields like animal breeding. The interested reader might consult “Nellie Landbloom’s Copybook for Beginners in Research Work.”

Acknowledgements: Thanks to Joe Bloom and Paul Manning for preparation of the data set and helpful discussions about the manuscript.