Why Scanner Data are Valuable to Marketing

Scanner data are beginning to have a large impact on the marketing of frequently purchased packaged goods. There are at least four good reasons that this is so. First, scanner data are timely. The lag between the time of purchase and the time that information is in the marketer's hands is shorter than with previous marketing data.

Second, scanner data are objective, and not subject to many of the types of errors associated with surveys or panels. Consumer's memories are notoriously fallible when it comes to reconstructing previous purchasing activity. In addition, there are powerful demands of prestige seeking and politeness which are placed on the consumer when we ask him or her what she has been doing or what he thinks of our products. In contrast, when the trade collects scanner data, they do not ask anybody anything. They simply record behavior.

Third, scanner data are numerous. It is possible to purchase data from many weeks and from many stores, where each of those stores may witness the sale of 1,000s of units of your products per week. Huge sample sizes can be brought to bear on marketing decisions.

Fourth, an excellent software platform exists for the analysis of scanner data - SAS. In fact, I believe SAS is peculiarly well-suited to dealing with scanner data. Scanner data must often be extensively transformed, sorted, and aggregated; things at which SAS excels. Conceptually, scanner data are rectangular arrays in two or more dimensions, where each cell of the array has a number of variables such as number of units sold and price. Such data are ideal for PROCs and DATA step processing. And finally, quantitative models must be used to reveal the marketing effects which are latent in the data. Again, an admittedly ideal application for SAS.

Scanner Data Can Help Assess the Marketing Program

Due to the timeliness of the data, scanner records can provide a constant real time assessment of the marketing program. Depending on the service subscribed to, a sudden dip in share can be detected within two weeks of its occurrence at the check stand. As the year progresses, one can tell if goals are being met, exceeded, or perhaps not being met. This use of scanner data is descriptive. SAS PROC SUMMARY, PROC FREQ, PROC TABULATE and other procedures can be used to perform this task. As a simple example, let us say we want to calculate dollar share for each brand in a particular market for which we have data. We could use the DATA step to calculate dollar volume for each brand by multiplying units by price. We could then use PROC FREQ, say, with dollar volume treated as a WEIGHT variable. The TABLE statement would specify brand as the variable to be analyzed. To look at unit share instead of dollar share, we need only use units as the WEIGHT variable.

Scanner Data for Marketing Strategy

Due to the objectivity of the data, scanner records are quite valuable for strategic reasons. The numbers represent actual sales in the market, so naturally they can be quite revealing about the nature of each brand, and the nature of the competition between brands. The mathematical tool we use to reveal the nature of the brands and their competitive structure is called a market response model. A further summary of competitive effects can be achieved by using multidimensional scaling to produce a competitive map, often referred to as a product space. The rest of this paper will be concerned with building a sales response model and deriving a competitive map.

Market Share Models

While classical economics has bequeathed us a set of techniques which deal with demand for commodities and other goods which are infinitely divisible, packaged goods are sold as whole units. Thus the consumer purchase decision is a discrete one: one either buys a bottle of Inglenook or Almaden. The store manager ordinarily does not allow consumers to fill a plastic bottle with a little of each. It is also true that in many markets total industry demand does not change much from one year to another. Marketing battles are therefore fought over share, which is a sum constrained to lie between 0 and 1 and which must total 1 for the market as a whole. The importance of the discrete nature of the data, and the share constraint, is that techniques based on ordinary least squares can lead to potentially inappropriate marketing decisions.

There are, however, techniques which are especially well suited to modeling market share. One such technique is based on seminal work by R. Duncan Luce (1959). Work by McFadden (1984) and Theil (1970) and others has resulted in the popularization of the Luce approach. To describe this class of models we will start with what Bell, Keeny and Little (1975) have called the fundamental market share theorem:

\[ S_i = \sum_j \frac{A_j}{A_i} \]

Here, \( S_i \) is the share for brand \( i \), \( A_i \) is the attractiveness of brand \( i \), and the summation in the denominator is taken over all brands in a particular consumer's choice set. The attraction of each brand may itself be a function of marketing instruments such as price,
specials, or features. The $A_i$ may also contain a component which merely varies randomly, and a component which is constant despite changes in marketing variables.

A number of functions have been used to model utility, including what Nakanishi and Cooper (1974) have termed a Multiplicative Competitive Interaction (MCI) model. Another popular option for modeling the $A_i$ is called the MultiNomial Logistic (MNL) model. In that case,

$$A_i = \exp \left( a_i + \sum_k X_{ik} b_{ik} \right).$$

The notation here has $a_i$ as a brand-specific intercept term which captures the level of attraction even in the absence of marketing activity. $X_{ik}$ represents the level of the $k$-th instrument for the $i$-th brand, such as brand $i$'s price.

The particular variant of MNL model above is called a simple effects model, because each marketing instrument for each brand only impacts its own brand, and that impact is the same for each brand. As an example, let us say there is only one marketing instrument; price. The above model holds that all the brands have the same sensitivity to price, $B$. Further, a change in price for one brand does not impact the attraction for any other brand, even though the shares of all brands may change due to the sum constraint.

A somewhat more complex version of (2) is called the differential effects model. In this model, the impact of a marketing action on attractiveness may differ from one brand to the next.

$$A_i = \exp \left( a_i + \sum_k X_{ij} b_{ik} \right).$$

In order to fit either of these models, that is; (1) and (2) or (1) and (3); we need to use PROC DLM or PROC CATMOD. A more complex version of the MNL model has differential effects for each brand, as well as cross- or competitive effects from every brand to every other brand. In such a case the representation of the $A_i$ is

$$A_i = \exp \left( a_i + \sum_k \sum_j X_{ijk} b_{ikj} \right).$$

Here, the $X$ values are defined as before, but there are unique $B$ values for the cross effect of every brand on every other brand. $B_{kij}$ would represent the impact of brand $i$'s $k$-th instrument on its own attraction. This model is called the extended model (Bultez & Naert 1975) or the cross competitive model (Carpenter, Cooper, Hanssens & Midgley 1987).

Even though these market share models are nonlinear, it can be shown that by taking the log of the ratio of the market share for two brands, called a logit, the computation of the unknown parameters, $B$, can be made quite efficient. One of the brands is chosen as the denominator for all the other brands. Choice of the base brand is arbitrary, and CATMOD uses the last brand for this purpose. However, that means that the $B$ coefficients for the last brand are not computed. In any case, attraction is not directly observed, so that one brand needs to be assigned a and $B$ coefficients of zero in order to set its metric.

In the next section, we will show how to prepare data for the models described by (2) and (3).

Preparing Data for the MNL Model

The retail trade sells scanner data to companies such as A. C. Nielson or Information Resources who in turn sell it to consumer goods companies. The amount and type of data purchased is subject to negotiation, but there are two broad classes of data: store level data and aggregate data. With aggregate data, it is important to note that share by itself is not sufficient for modeling. The raw number of units (or dollars) is needed in order to assess the relative amount of sampling error which is invariably present in share. Store level data are quite useful for modeling, but adds a complicating set of factors to the discussion. To use store level data, a way must be found to remove the bias associated with stock-outs, distribution must be handled by the model in some way, and dummy variables (or some other method) must be used to account for variation from one store to another.

Rather than deal with these issues in a short paper, I will illustrate a market response model which uses aggregated data. The data set which will serve as the example has 14 relevant brands. A typical form for all scanner data consists of observations with the following variables: UPC, WEEK, STORE, UNITS, PRICE, FEATURE, and DISPLAY. If you have store level data, the first step is to aggregate the data up to the week level. To do this, you could use the following code which uses PROC SUMMARY. The example starts off by summing sales across all stores by week and by brand. In the second PROC SUMMARY invocation, average price is calculated by week and by brand. In the first PROC SUMMARY, units is given on the VAR statement. In the second, units is used as the WEIGHT variable in order to calculate average price.

```sql
proc summary data=xscanner.allbrand nway;
   class week brand;
   var units dollars;
   output out=usums sum = units dollars;
   * Price summarized by week, by brand;
proc summary data=xscanner.allbrand nway;
   class week brand;
   var price;
   freq units;
   output out=preprice mean = aprice;
```

If you have store level data, the first step is to aggregate the data up to the week level. To do this, you could use the following code which uses PROC SUMMARY. The example starts off by summing sales across all stores by week and by brand. In the second PROC SUMMARY invocation, average price is calculated by week and by brand. In the first PROC SUMMARY, units is given on the VAR statement. In the second, units is used as the WEIGHT variable in order to calculate average price.

```sql
proc summary data=xscanner.allbrand;
   class week brand;
   var units dollars;
   output out=usums sum = units dollars;
   * Price summarized by week, by brand;
proc summary data=xscanner.allbrand nway;
   class week brand;
   var price;
   freq units;
   output out=preprice mean = aprice;
```
TRANSPOSE can preprice, and transpose it in such a way that it will prove convenient to take the SAS dataset the 14 brands become 14 variables. PROC TRANSPOSE can be used for this.

Assume that TRANSPOSE creates a data set "prices". The same process needs to be performed for the USUMS dataset, although here we need to divide the brand sales by the weekly totals in order to determine the market share for each brand for each week. Weekly totals will have a missing value for brand, so sorting will facilitate this task:

```sas
proc sort data=usum ;
by week brand ;
data pretrans ;
keep week brand share ;
sst sum ;
if first.week then weektot = units ;
else do ;
share = units / weektot ;
output ;
end;
proc transpose data=pretrans out=response ;
by week ;
var share ;
```

One convenient way to fit the models (2) and (3) is to use IML. The IML modules appear below:

```sas
/* SCANNED DATA ANALYSIS PROGRAMS - GLS LOGIT MODEL AND SUPPORTING ROUTINES */

DR. CHARLES HOFACKER V1.00 (2/88) 904/644-4091

PROC IML ;

START SCAMMOn ;
Q)
proc sort data=usum ;
by week brand ;
data pretrans ;
keep week brand share ;
sst sum ;
if first.week then weektot = units ;
else do ;
share = units / weektot ;
output ;
end;
proc transpose data=pretrans out=response ;
by week ;
var share ;
```

It will prove convenient to take the SAS dataset preprice, and transpose it in such a way that the 14 brands become 14 variables. PROC TRANSPOSE can be used for this.

One convenient way to fit the models (2) and (3) is to use IML. The IML modules appear below:

```sas
/* SCANNED DATA ANALYSIS PROGRAMS - GLS LOGIT MODEL AND SUPPORTING ROUTINES */

DR. CHARLES HOFACKER V1.00 (2/88) 904/644-4091

PROC IML ;

START SCAMMOn ;
Q)
```
The modules ASV, GEN, POLY, and ASC are used to build a design matrix for input to the SCANMOD module. A sample use is given below using the data sets which we just created. We start by creating an IML module to build the design matrix. We will have occasion to build the design matrix more than once:

**START SCR ;**
* CREATE ASCS ;
RUN ASC ;
* CREATE DIFFERENTIAL EFFECTS FOR PRICE ;
X = C ;
DIF = 1:14 ;
RUN DIF ;
X - X II D ;
FINISH ;
**START ELCONS ;**
**ROUTINE TO CALCULATE DERIVATIVES AND ELASTICITIES.**
WHEN IVS ARE CONSTANT ACROSS BRANDS (INCOME TYPE VARS) ARGUMENT LIST IS IDENTICAL TO ELVARY

**The ASC module produces Alternative Specific Constants in a matrix C, which are the "a" values in (2) through (4). The DIF module produces a matrix D which will allow a separate beta for each brand (except the last), as in (3). The SCANMOD module expects the independent variables to be in a matrix X, so as the design matrices are created, they are placed in X.**

Next, we get the response data set containing the week by brand matrix with market share being the entries in the matrix.

**USE RESPONSE ;**
READ ALL INTO N ;
**The same READ process follows for the prices dataset, which was produced by TRANSPOSE from "preprice," and which also has weeks for rows and the 14 brands for columns. Here, however, the entries of the dataset consist of average prices.**

**USE PRICES ;**
READ ALL INTO Z ;

Next we pick off the first years worth of data, and produce some other input required by SCAMMOD: the number of observations (N), the number of brands (K) Q=R-1. The vector BRANDS is used to pretty up the output; the names have been changed so as not to reveal any trade secrets.

**N = N(1:52, );**
**Z = Z(1:52, );**
**S = NROW(N);**
**R = NCOL(N);**
**Q = R - 1 ;**
**BRANDS = "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" ;**

Next, we run the SCR module to actually build the design matrix, then we fit the GLS model using SCANMOD.
The output contains the GLS beta coefficients along with the t probabilities to test whether individual betas are null. This module also produces the Wald statistic, which is used to test the null hypothesis that the model fits. A chi square test is also provided which tests the null hypothesis that all predictors other than the "a" terms are null.

At this point it would also be possible to use the HYPOTH module to test further hypotheses of interest, such as whether all price betas are equal.

It is frequently more convenient to look at the derivatives of market share with respect to the marketing instruments, or the elasticities, than it is to look at B coefficients (Steinberg 1987). The elasticity of share for brand i as a function of marketing instrument k for brand j is defined as

$$ e_k^{(i,j)} = \frac{S_j}{x_j^k} \left( \frac{x_j^k}{S_j} \right) $$

In our example, we have been using price as the only marketing instrument, so that these elasticities would be price elasticities of share.

There are two IML modules for calculating elasticities. ELVARY is used to produce elasticities for any independent variables which are allowed to vary across brands. ELCONS is used to produce elasticities for any independent variables which are constant across brands, and are allowed to vary across brands. For our example, we have PRICE, which is clearly one of the former independent variables, not the latter. In the code which follows, assume that APRICE contains 14 observations containing the average price across the time period studied for the 14 brands. We begin by reading in APRICE into XAVG, then transposing the vector into the form required by ASC and DIF.

```
USE APRICE;
READ ALL INTO XAVG;
XAVG = XAVG';
Z = XAVG;
S = 1; R = 14; Q = 13;
RUN SCR;
```

Now we have created the appropriate design matrix for the betas which we have previously calculated with SCANNOD. The predict module wants the input vector to be called XO.

```
X0 = X;
RUN PREDICT;
```

```
PRINT 'PROBABILITIES PREDICTED FROM AVG VALS', ,
PP[FORMAT=5.1 ROWNAME=BRANDS]);
```

ELVARY can only handle a single independent variable at a time. It requires that the input matrix, called B, contain Beta values produced by SCANNOD for a particular independent variable (like price). B is square, and in our case will be 14 by 14. The rows of B correspond input effects (brands) and the columns correspond to output effects (brands). For model (2), B will be a scalar matrix with a constant diagonal element. For (3), B will be a diagonal matrix, while for (4), B will be a symmetric full matrix. To summarize, b_{ij} contains the impact of brand i on the attraction of brand j. ELVARY also requires the price point at which to evaluate the elasticities in the vector X0.

```
B = (DIAG(BETA(16:28))); J(13,1,0) //
J(1,14,0) ;
X0 = AVG;
```

We might note here that the B matrix needs to be augmented with a row and column of zeroes. This is because the model (3) does not contain a beta for the last brand. Here we go:

```
RUN ELVARY;
PRINT 'DERIVATIVES',B,ELASTICITIES',X;
```

The elasticities contain two types of information of use for marketing decisions: diagonal elements and off-diagonal elements. Diagonal values reflect the sensitivity of a brand to its own marketing actions. In general terms, diagonal price elasticities are almost always negative. In theory, the closer they are to zero, the more profitable the brand can be, for raising its price does not lead to much loss of share. In practice, however, large profitable brands can have very negative elasticities because consumers will be more inclined to stockpile them.

Display and feature elasticities tend to be positive. The larger these elasticities, the stronger the suggestion that the brand will benefit from marketing activity.

The off-diagonal cross-elasticities reveal the competitive structure of a market (Cooper 1987). For example, for price, most competing brands will tend to have positive cross-elasticities. As brand i raises its price, brand j will benefit. If brand j benefits a lot, that implies that consumers switch off to brand j, and the two compete directly. If brand j benefits hardly at all, that implies that the two are dissimilar. These ideas suggest that price elasticities provide important clues for the nature of the product space. Brands with high cross-elasticities are next to each other in the market but brands with low cross-elasticities are far apart.

```
PROC ALSCAL, which is available from the SUGI Supplemental Library, can produce such a map. In fact, ALSCAL has a special option which can handle asymmetric input matrices such as a matrix of elasticities.
```

Running PROC CATMOD

```
It is possible to use CATMOD to calculate the simple effects or differential model. Bill Stanish of the SAS Institute has pointed out that the trick here is to utilize the R = option of the model statement to input the entire design matrix. One can use the DATA step to produce the design matrix, output it to a file, and then
```

Let us talk about the fully extended model. In order to trick CATMOD into fitting this model, the 14 variable price vector created above in "price" must be duplicated for each of the 14 brands. This will be accomplished below easily enough as the MERGE statement will duplicate records from the shorter SAS dataset. We need to merge the price data set with the output dataset the 14 variable price vector created above for each of the 14 brands. This will be accomplished below easily enough as the MERGE statement will duplicate records from the shorter SAS dataset.

Recall that "prices" came from "preprice" by way of TRANSPOSE. At this point we are ready to run CATMOD. CATMOD requires a set of 'populations' and a set of 'responses'. In our case, the populations are the 52 weeks and the brand choices are the responses. Here is the code.

```sas
proc catmod data=xscanner.catmod;
  direct price1-price14;
  population week;
  weight units;
  model brand = price1-price14 / prob;
```

The DIRECT statement is used so that the prices are used as is, rather than a design matrix being created. The POPULATION statement identifies the populations to be analyzed, which here is necessary because our cases consist of brand-weeks, rather than weeks. The WEIGHT statement tells CATMOD that the variable UNITS gives a frequency count for each response, which is to say each brand.

CATMOD will provide a test of model residual, as well as tests of significance for each independent variable. In our case we have 14 independent variables, each with 13 degrees of freedom. CATMOD will also print the individual B values, sorted by independent variables, and within independent variables, sorted by response function. In our case there are 13 response functions consisting of the logit for each brand's share divided by the last brand's share. Thus there are 14*13 B values, and each one can be assessed with a single degree of freedom chi square test.

Suggestions for Enhancements to SAS/STAT

By repeating the vector of independent variables for each response, CATMOD will fit the fully competitive model for scanner data. With some effort, CATMOD can be made to fit the simple and differential effects model often called the conditional logit model. CATMOD is primarily designed to deal with variables which do not change from one response to the next, which is to say variables which are constant for the population. Another inconvenience about CATMOD is that it cannot calculate elasticities subject to user input of a relevant "price" point.

CATMOD is already quite general, and it is not clear what the syntax for these new models would be if they were added to it. Instead, I would propose that a new procedure be developed which could certainly utilize most of the routines already programmed for CATMOD. This new procedure would specialize in logistic regression, allowing both maximum likelihood and weighted least squares estimation. In addition, "income" type variables would be allowed as well as price type variables in a conditional logit specification.

References


