

Consumer Demand Preference Patterns: Using PROC IML* to Forecast Future Demand

Daric Brummett, University of Notre Dame
Lawrence Marsh, University of Notre Dame

Economic models of consumer demand generally involve assumptions of economic rationality, permanent tastes and action only in accordance with a single underlying smooth demand function. This method assumes two extremely important, and yet all too often ignored, facts. First, that tastes are permanent, and second, that a theorist will choose the proper functional form for demand. Most of us have experienced a taste change at one time or another. However, in economic theory, this fact is difficult to model. Additionally, in usual economic analysis, the demand function is chosen by the theorist from a limited number of well defined, well behaved and well known functional forms. But when forecasting consumer demand, a method which allowed taste changes to be modeled as well as allowing the data to choose the proper functional form, would be desirable. This paper discusses a solution to the problem of taste changes and functional choice utilizing a program written in PROC IML.

The solution presented here is twofold. First, IML is used to define a revealed preference index which ranks all bundles of goods purchased in the period under study. Taste changes may be accommodated in such a system. Second, IML is used to develop a neural network architecture which is capable of forecasting without a prior specified functional form. This paper utilizes a data set of a single consumer purchasing four goods over twenty periods. However, because the program is written entirely in adaptive matrix terms, any size data set may be accommodated.

The theory of revealed preference is simple enough. If an individual uses income in one period to purchase a basket of goods and in a later period that same bundle is again affordable and is *not* chosen, then the bundle chosen in the later period is revealed preferred to the bundle chosen in the first period. Therefore, the first task for the program is to compare each bundle chosen in each period to all other bundles chosen in different periods. The program creates a matrix of the income required to purchase all bundles at all prices. That matrix is then compared with the actual income available in each period. If the income available in a period is greater than that needed to purchase other bundles, then the bundle chosen is issued a '1' signifying a preference for each time it is preferred. Through this method, a binary square matrix, with dimension equal to the number of observations, is created. Then the program rowsums the binary matrix giving a preference index which reveals an individuals ranking of each bundle.

In IML:

```
BINARY = (INCOME * J(1,N,1)) > INCALL;
INDEX  = BINARY * J(N,1,1) + J(N,1,1);
```

Where N = number of periods of observations, INCOME is an Nx1 vector of the income actually available in each period, and INCALL is an NxN matrix of income necessary to purchase the bundles of the jth period at the prices of the ith period. Therefore, the comparison operator '>' yields a binary NxN matrix. INDEX is then an Nx1 rowsum of Binary. The vector of ones is added to INDEX to make its range from 1 to N rather than from 0 to (N-1).

The revealed preference index portion of the program solves the problem which arises when tastes change. The program simply takes, as input data, actual events. This way, any taste changes which occur, may be fully incorporated into the index. If a moving twenty period interval is utilized, as time goes by the influence of past tastes will decline as does the likelihood of returning to those taste patterns.

Having addressed the problem of taste changes, the problem of functional form is now at hand. Thus far, the program has the original data set containing past consumption prices and bundles chosen in each period. Additionally, the preference index is now available as data. Forecasting future consumption in response to changing prices will require a function to be fit to the index points. The question of functional form is upon us. Should the function be linear or nonlinear, standard or improvised? Fortunately, neural networks allow for such decisions to be made by the data itself.

The neural network structure allows all interactions of the explanatory variables to emerge as the data warrants. The neural network utilized here is three layers deep. Its structure is as follows.

i = 1...N N = number of observations
j = 1...G G = number of goods in bundle

$$Z_{Aij} = [1 + \text{Exp}(-(\alpha_j * I_i + \beta_j * P_{ij} + C_j * I_i * P_{ij}))]^{(-1)}$$

$$Z_{B1i} = [1 + \text{Exp}(-(\psi_{a1} * Q_i * Z_{Ai1} + \psi_{b1} * Q_i * Z_{Ai2} + \dots + \psi_{g1} * Q_i * Z_{Aig}))]^{(-1)}$$

$$Z_{B2i} = [1 + \text{Exp}(-(\psi_{a2} * Q_i * Z_{Ai1} + \psi_{b2} * Q_i * Z_{Ai2} + \dots + \psi_{g2} * Q_i * Z_{Aig}))]^{(-1)}$$

$$Z_{B3i} = [1 + \text{Exp}(-(\psi_{a3} * Q_i * Z_{Ai1} + \psi_{b3} * Q_i * Z_{Ai2} + \dots + \psi_{g3} * Q_i * Z_{Aig}))]^{(-1)}$$

$$X_{ij} = W_{j0} + W_{j1} * Z_{B1} + W_{j2} * Z_{B2} + W_{j3} * Z_{B3}$$

In this system, Z_A transfers each observation onto a sigmoid zero-one interval. The model in the Z_A equation is the base model and the terms for income (I), price (P) and their interaction (I*P) are accounted for. Each good (j) in each consumption bundle (i) has a separate Z_A value. The Z_A values are allowed to interact in the Z_B equations (also sigmoidal). The Q_i

parameter in the ZB equations represents the revealed preference index value corresponding to the i th bundle. The ZB values are, in turn, utilized as explanatory variables for the estimates of all goods in each bundle, X_{ij} .

After all parameters have been set to initial values and ZA, ZB, X values have been initially calculated, the system must be trained. Training simply means altering the parameters of the system so that they more closely estimate the actual amounts purchased in each bundle. Training is accomplished by calculating the difference between the commodity estimates X_{ij} and the actual commodity values of the i th bundle (call it Δ_i). The parameters are then adjusted by the reciprocal of their derivatives with respect to X multiplied by Δ_i and divided by the number of parameters to be adjusted.

For example the adjustments to the ZA parameters would be:

$$\begin{aligned}\alpha_{\text{new}} &= \alpha_{\text{old}} + [\Delta_i * (dX/d\alpha)^{-1} * (\text{NUMPARAM})^{-1}] \\ \beta_{\text{new}} &= \beta_{\text{old}} + [\Delta_i * (dX/d\beta)^{-1} * (\text{NUMPARAM})^{-1}] \\ C_{\text{new}} &= C_{\text{old}} + [\Delta_i * (dX/dC)^{-1} * (\text{NUMPARAM})^{-1}]\end{aligned}$$

This process is repeated multiple times for all bundles in the period of observation until the convergence criterion is satisfied (convergence criterion is discussed below).

If all data is used in training, there is the obvious problem of conforming too closely to the sample and losing predictive power. A natural solution is to train the system with only a fraction of the available data. Prior to entering PROC IML, in the data step, a random binary variable may be created and added to the data observations. This is accomplished via:

```
INPUT DATA1 DATA2 DATA3 @@;
B = RANUNI(SEED) * F;
B = ROUND(B,1);
CARDS;
```

Where B is the binary variable and $F+1$ is the inverse of the fraction of data to be utilized in training the neural network. (For example, if $F=2$ then 1/3 of the data available will be utilized in training.)

After PROC IML has been invoked, it is a simple task to separate observations into two groups. The group in which $B = 0$ would be the data kept to train the neural network. Otherwise the observation would be included in the out of sample data group.

The training termination criterion is the minimization of the out of sample error sum of squares. Here the out of sample prices and quantities are put into the neural network to generate estimates of commodity amounts. The difference between the estimates of the out of sample quantities and the actual out of sample quantities are the errors. When the sum of squared out of sample errors is at a minimum, training is terminated.

For this small example, there are four goods and twenty observations.
The data follows:

```
Price1 1 1 1 1 2 1 1 1 1 2 2 1 2 1 2 2 1 1
Price2 3 4 3 4 3 4 5 4 5 3 4 3 3 4 3 5 3 3 4 4
Price3 5 5 5 4 3 2 2 1 2 1 2 1 2 3 3 2 4 4 5 6
Price4 6 5 6 6 6 6 6 6 6 6 6 6 7 6 5 6 6 6 7 7 7

Good1 10 10 10 10 9 7 8 8 8 7 8 7 8 10 12 12 11 13 9 8
Good2 4 4 5 5 5 4 4 3 4 3 4 5 4 4 5 6 6 4 4 3
Good3 9 7 6 5 5 6 7 7 9 9 10 8 9 10 8 9 9 9 8 9
Good4 2 2 2 3 2 2 2 1 2 3 3 3 3 2 4 3 1 2 3 3
```

The INDEX generated by the revealed preference portion of the program is as follows (ranking for bundle 1...N):

```
INDEX 14 16 14 11 3 11 6 2 5 1 4 3 5 1 4 3 5 10 12 6 16 18 19 20
```

When splitting the data, F was set at one, therefore, half the data was utilized in training the neural network and the other half was utilized in calculating the out of sample error sum of squares. The seed value was set to 1000. The observations (columns of the above data) which were issued a value of '1' (signifying inclusion in the training group), and '0' (signifying exclusion from the training group) were:

```
IN    1 3 5 7 8 9 10 12 17 18 19 20
OUT   2 4 6 11 13 14 15 16
```

The training routine was then called and the minimum out of sample error sum of squares determined. In this example the minimum error sums of squares occurs at the 359th iteration of training at a value of 17.477085. At that iteration of training, the coefficients were as follows:

ZACOEFF	α_j	β_j	C_j
	-3252.490	-3264.570	-3250.194
	-93828.81	-70430.76	-94684.27
	125903.55	125902.94	125903.62
	-192764.7	-180352.6	-192955.7

ZBCOEFF	ψ_{aj}	ψ_{bj}	ψ_{cj}	ψ_{dj}
	-1.553E43	-442894.1	-9387528	-442894.1
	-1.323E51	-48859213	-8.165E13	-48859213
	-3.233E55	-1.203E12	-1.995E18	-1.203E12

W	Wj0	Wj1	Wj2	Wj3
	-3249.741	-5.783E46	7.2609E46	7.2609E46
	-94968.42	-1.689E48	2.1212E48	2.1212E48
	-151526	-2.696E48	3.3844E48	3.3844E48
	-193145.6	-3.436E48	4.3141E48	4.3141E48

The columns of the ZACOEFF matrix correspond to the parameters α β C, while the rows signify each different commodity present in the baskets (four in this case). The columns of the ZBCOEFF matrix correspond to the parameters $\psi_a \dots \psi_g$, while the rows correspond to the parameters belonging to ZB1, ZB2 or ZB3. The columns of the W matrix correspond to the four W coefficients needed for each commodity estimate. The rows of the W matrix correspond to each of the goods in each basket (four in this case).

Forecasting new consumption patterns resulting from price changes is a straightforward process. After the system has been trained and parameter estimates (ZACOEFF, ZBCOEFF, W) computed, the new price levels and current (or anticipated) income levels are inserted into the ZAj equations. From the resultant ZAj values, ZB1, ZB2, and ZB3 values may be obtained. The estimated consumption bundle will emerge when the ZB values are inserted into the Xj equations.

This brief paper demonstrates that forecasting consumer behavior without reliance on taste permanence or assumed functional form is not as daunting as may have been previously thought. All that is necessary is a revealed preference procedure and a neural network architecture suitable to the problem at hand. The procedure IML makes such forecasting considerably easier and greatly increases its flexibility by providing an environment which allows highly specialized applications to be programmed by those who will ultimately utilize them.

*SAS/IML is a registered trademark of SAS Institute Inc., Cary, NC, USA.

Any questions or comments may be addressed to the authors at the following address:

Department of Economics
University of Notre Dame
Notre Dame, IN. 46656

If BITNET is more convenient:

BRUMME@IRISHMVS
LMARSH@IRISHMVS

If telephone is more convenient:

D. Brummett 219-287-1288
L. Marsh 219-287-0458