The Induction of Consumer Preference Models using Evolutionary Computation

Bruce Edmonds, Scott Moss and Andrew Lock
Centre for Policy Modelling,
The Business School,
Manchester Metropolitan University,
Aytoun Building, Aytoun Street, Manchester, M1 3GH, UK.
Tel: +44 161 247 6479 Fax: +44 161 247 6802
web: http://www.cpm.mmu.ac.uk/ email: b.edmonds@mmu.ac.uk

Abstract

A technique is described whereby context-dependent consumer preference models can be automatically induced using algorithms taken from the field of evolutionary computation. In order to make this possible an abstract meta-model is constructed to relate classes of preference models to aggregate price and sales data. This meta-model is broadly consistent with existing consumer preference models, but its prime purpose is to provide an appropriate framework for the technique. The technique proceeds as following: a marketing practitioner specifies the relevant market attributes and the perceived values of these attributes for each product; the algorithm then induces models within this framework that explain the aggregate data. We tested this technique on markets for alcoholic beverage. We found good fits with the data using relatively short sequences of in-sample data, but more importantly it gave qualitative information about the possible contexts of consumer purchases.

Keywords: brand choice, choice models, buyer behaviour, market structure, (consumer preferences, modelling, evolutionary computation, purchasing context, judgement, aggregate data)

1 Introduction

A common procedure in modelling for marketing analysis is to specify the structural equations to enable the application of robust statistical algorithms to available data. The algorithms determine the values of the parameters of the structural equations. The purpose of this paper is to demonstrate a technique for the induction of part of the *structure* of marketing models from the data. The overall idea is to use techniques from the field of evolutionary computation to automatically find models of consumer preferences that are in closest agreement with aggregate sales and price data. To do this effectively we get practitioners to specify some elements that constrain the space of possible models. We have found that this technique produces models that produce a good mapping to actual sales data from a relatively small training sets of data.

The method reported here is an application of techniques from the field of evolutionary computation (see below). The algorithm induces from the data a set of models that lie within a space of possible models defined by the modeller. The constraints on the search space that determine which models are possible are derived from judgements about the relevant variables representing product attributes for a particular market and the perception of consumers of the values of these attributes for each product[1]. This framework is essential as it both ensures that the induced models have a natural marketing interpretation in terms of the qualitative attributes of groups of consumer purchases and that the search space is sufficiently constrained so as to make the induction practical using moderate computational resources. A meta-model is necessary in order to structure this process.

2 Structuring the Search for Consumer Preference Models

The point of our approach is to get a computer to induce as much of the model structure as possible. We thus seek to *delay* as much of the model specification as possible: we specify a generic framework for relating a certain type of consumer preference models to aggregate data; a marketting practitioner specifies some information to constrain the possibilities to those that are relevant to a particular market; and the computer then induces models within these constraints to explain the aggregate data. Thus the purpose of this paper is not to exhibit a particular model but rather a *method* for inducing models – a method which uses both judgemental information and aggregate quantitative data. The purpose of the meta-model is twofold: *firstly* to provide a framework for the integration of judgemental and quantitative information and *secondly* to make possible the automatic induction of meaningful consumer preference models.

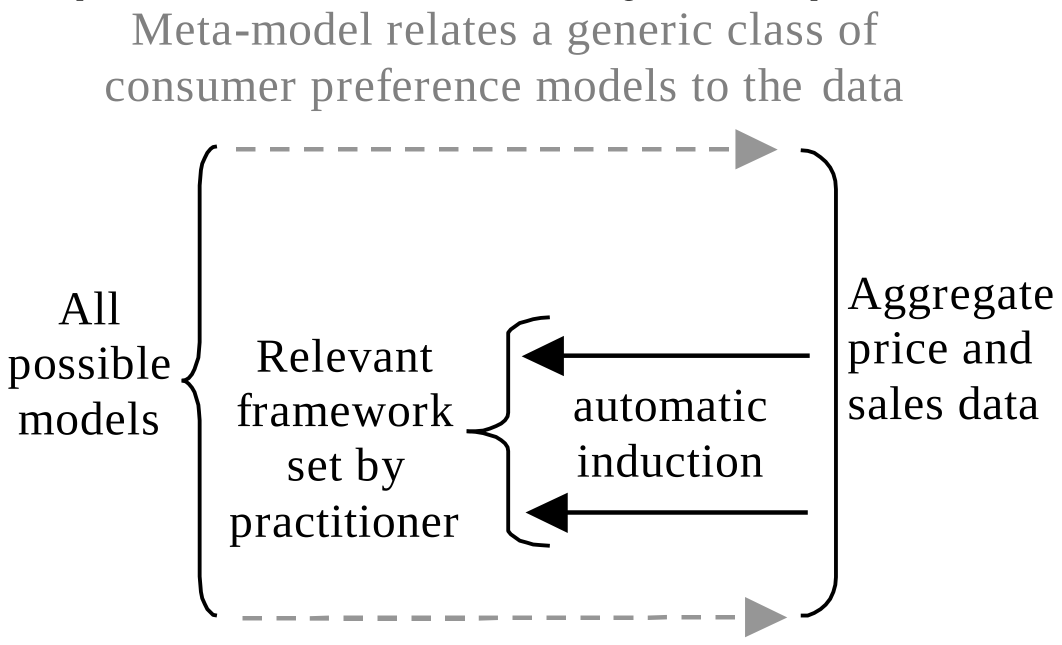

Meta-model relates a generic class of consumer preference models to the data
All possible models
Relevant framework set by practitioner
automatic induction
Aggregate price and sales data

Figure 1. The Structure of the Technique

Figure 1, immediately above, illustrates the overall process. A meta-model relates, in an abstract way, a set of variables relating relevant product attributes (size, relative price, quality, etc.), a set of products with perceived values of these attributes and a set of purchasing clusters (each with their own ideals, tolerance to deviation from this ideal and

[1] Throughout this paper we will count different sizes of products as separate, and also include 'un-producted' items under pseudo-products such as 'own-label' for supermarket producted products. Seperating out different sizes is important as different sizes are bought in different purchasing contexts. For example a customer might buy a large bottle of spirit for a party but only a small bottle if it is a self-reward.

criticality of the attribute) to the aggregate prices and market shares. A marketing practitioner or other expert specifies which attributes may be relevant and what values the various products have of these attributes (in terms of consumer perception) – this constrains the possible preference models to those that deal with these attributes. The algorithm then induces models each composed of an indefinite number of purchasing clusters with their preferences that match the observed aggregate price and market share data.

The meta-model, although designed *primarily* with a view to supporting this process, inevitably has *some* content of its own. The same technique could be used with a variety of meta-models, as long as they met the criteria described in section 5. From our point of view the technique is more important than particular details of the meta-model.

However, the described meta-model is broadly consistent with well-established models. It is predicated on the assumption of an attribute space in which products are located and, secondly, the assumption that consumers' perceptions of the positioning of products are homogeneous (Blattberg and Hoch, 1990). Despite the homogeneity of perceptions, consumers' preferences are assumed to be heterogeneous and dependent on the consumption context. It would, of course, be possible in principle to model the perceptual space and attribute preferences conditioned by consumption context using, for example, non-metric multidimensional scaling, conjoint measurement and related techniques. However, this preclude the automatic induction of models using the methods described here and be unlikely to result in models that are meaningful to the marketing practitioners that use them[2].

Consumers are not directly represented but rather *clusters of purchasing decisions* made by consumers are represented in terms of their individual ideal values of the attributes (along with their tolerance to deviation from these and the criticality of the attribute). These ideals etc. are conditional upon the intended consumption context. The propensity of purchasing a particular product is inversely related to the distance of each product from the consumer's ideal point. The basic model is consistent with market attraction models where a log-odds transformation of purchase probability is related to perceptual distances from the ideal point (Morgan-Jones and Zufryden, 1980) and broader optimal positioning models (Horsky and Rao, 1984). The exact *shape* of the preference curve was not found to be critical to the success of the technique, for it was found that other plausible shapes did almost as well

Apart from price and weekly aggregate sales for each product, the models were derived automatically from practitioner judgement within a literature-derived framework. The use of expert judgement has been widely studied. Early work in psychology (e.g. Meehl, 1954) started by taking statistical models as a floor to show the additional predictive power of expert judgement. However as Dawes (1972) put it, "the floor turned out to be the ceiling". Since then there has been a widespread literature showing poor judgemental performance for a variety of tasks and settings, and a less extensive literature which appears to show that, for tasks with explicit feedback and for which expertise is specifically developed, expert judges perform well and their models are well calibrated

[2] It should be noted, however, that the purpose of the method reported here is to co-evolve models and strategy within a framework based on the recognition that, for many purposes, commercial and business environments are too complex and poorly understood for forecasting models to be credible and reliable.

(c.f. Murphy and Brown, 1984). Blatternberg and Hoch (1990) present an example of conditioning database models with managerial judgement in a marketting context. An important difference between previous studies and the results reported here is that, in many of the experimental cases previously reported, subjects, whether expert or inexpert, were asked to estimate the value of a dependent or target variable while, in the development of the models reported here, the practitioners were asked to specify those aspects of a market they were most certain of: the relevant product attributes and the perception of products in terms of these attributes[3]. There were then a number of iterations refining the models and then refining estimates in the light of the output of early model runs.

If no models can be found that are in reasonable agreement with the data, this would be an indication that these elements are false. Thus the technique provides a weak consistency test of descriptions of the structure of purchasing preferences against data models of the resulting sales, and a method of inducing better models as a result of a practitioner-simulation interaction.

3 The Data

The input that this technique needs falls into two halves: aggregate quantitative data and qualitative judgemental input.

The quantitative data we used is aggregate weekly EPOS data for a region over two years for the volume of each product (in each size) and the volume of sales. In fact the technique can be used on much shorter runs of data, since we typically used short sequences of in-sample data (e.g. 5-20 weeks).

The qualitative data consisted of specifying the attributes that might be relevant for a particular market and the approximate perceived value of that attribute that each product (in each size) is perceived to posses on a five point Likehart Scale (high, above average, average, below average, low).

4 Some Techniques of Evolutionary Computation

Over the last 30 years computer science and artificial intelligence have increasingly looked to biology as a productive source of ideas. In particular evolutionary and genetic processes have been taken as a paradigm for "robust" search procedures, i.e. ones that can produce reasonable solutions for difficult problems (as opposed to optimal solutions for easier problems). The computational techniques that have resulted are called "evolutionary algorithms" and the field has come to be known as "evolutionary computation".

4.1 Genetic Algorithms

The most well-known of these techniques was introduced by John Holland in the seventies, called a Genetic Algorithm (GA) (Holland, 1975). In a GA each possible solution to a problem is encoded as a fixed-length string of symbols. Each such string thus represents a (better or worse) way of solving the target problem. In the GA a large

[3] In our trials the practitioners had only to rank the attributes on a five point scale: high, above average, average, below average and low. These categories were mapped into numbers. As with the shape of the preference curves, the exact values were not critical to the results.

collection of such solution strings are collected together into a population. Initially this population is generated at random. The population is then “evolved” over many generations in a manner directly analogous to DNA in living organisms. That is, in each generation the bit strings which encode the better solutions (the “fitter” ones) are propagated more often into the next generation. Thus better solutions tend to predominate and “push out” the less successful solutions as time progresses. Variation is continually introduced into this population by the two mechanisms of mutation and crossover. Mutation acts upon the solutions to randomly change some of the symbols in the strings to other symbols. Crossover mimics the sexual recombination of genes – two strings are selected, a random point is chosen, both strings are cut at this point and the material after the crossover point of each gene is swapped over to form two new “child” solutions. The action of mutation and crossover are illustrated in Figure 2.

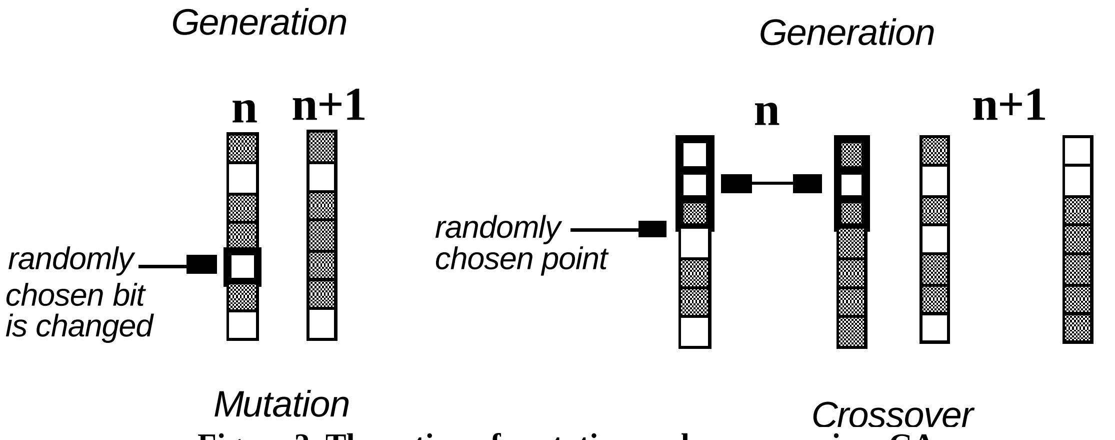


Figure 2. The action of mutation and crossover in a GA

Mutation enables the introduction of novel solution patterns while crossover allows the recombination of existing ones. The fitness of the solutions (and hence the extent the string or its “children” enter the next generation) is usually determined by a function that the programmer specifies, this is called the “fitness function”. It is this fitness function that determines the goal for the whole process. It is designed so that it attributes the maximum value to the best possible solution. The whole process is illustrated below in Figure 3.

The resulting process is a search procedure which is good at finding acceptable solutions to difficult problems. It does not necessarily find the best possible solution but is “robust” in the sense that it is very unlikely that it will get stuck at a bad solution that happens to be a local optimum. Such algorithms are easy to apply without a great deal of domain knowledge and also allow for a considerable degree of parallelism in the computation. For these reasons GA’s have become quite a widely used tool. There are now many variations on the basic techniques and applications of it. A good and practical general introductions to GAs is (Goldberg, 1989).

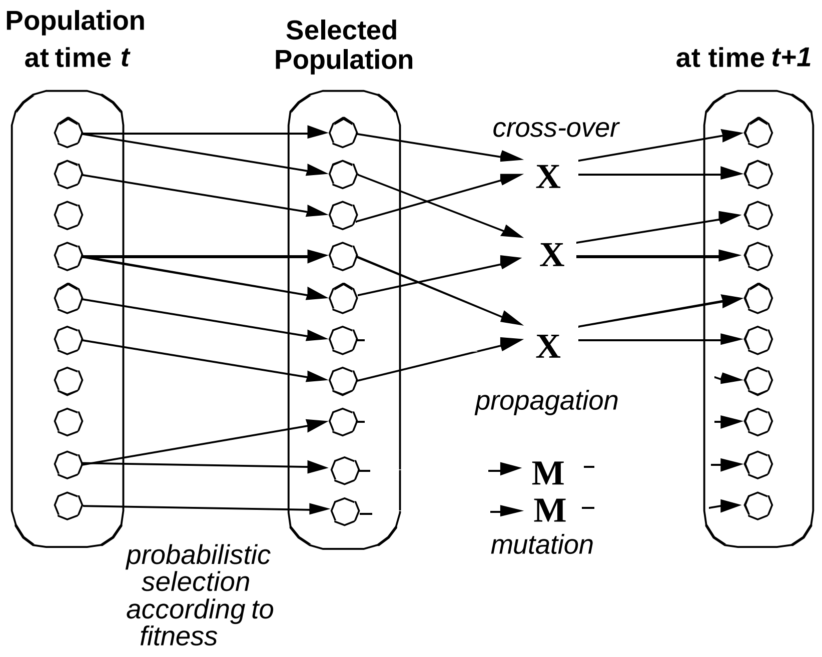


Figure 3 The operations of propagation, mutation and crossover in a GA

4.2 Genetic Programming

A major limitation of GAs is the requirement for a fixed length bit string with a finite number of symbols. Although this restriction makes for efficient computation it makes GAs difficult to apply in situations where an open-ended solution is required such as a program, a network or an expression.

The genetic programming paradigm (GP) was developed by John Koza (1992) to get around this kind of limitation. The technique is basically the same as for GAs but each solution is encoded as a tree-structure instead of by a fixed-length symbol string. The possible tree-structures are determined by a formal grammar which determines what nodes and terminals etc. it can have. The tree-structures can be of any depth, although sometimes a maximum is imposed for reasons of computational efficiency. An example tree-structure is shown in representing an arithmetic function.

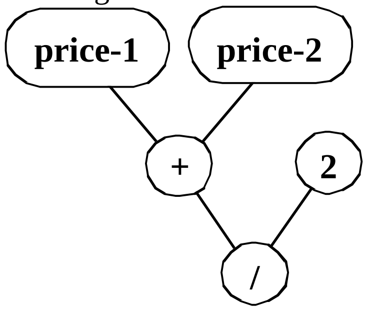


Figure 4 An example of a solution encoded as a tree-structures

The difference in structure of potential solutions means that there are also some minor differences in the way the algorithm is implemented. In particular mutation is not always used and crossover has to be a bit different: instead of choosing a random position in a string, a random node from each of the two “parent” trees are chosen and the sub-trees which are rooted at these nodes are then cut, swapped and “grafted” onto the other to form the “child” trees. This is illustrated in Figure 4.

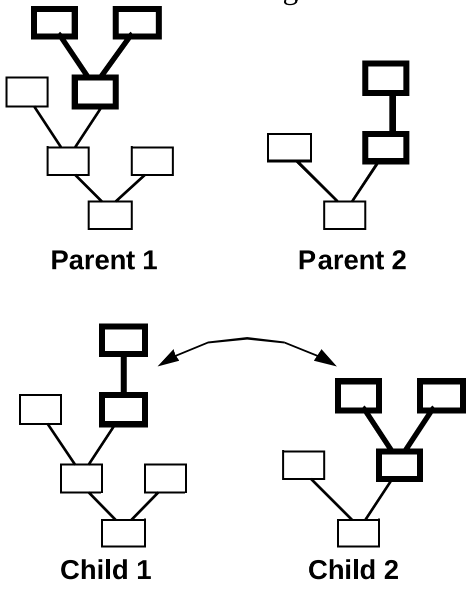


Figure 5 The action of crossover in GP

The increased expressiveness of the encoded solutions in GP means that it can be applied with less effort than is needed to find suitable encodings into fixed-length strings. It also means that the algorithm can come up with genuinely surprising and innovative solutions that the human programmer might never have thought of. There are several variations on the basic technique and many applications. A good introduction to the basic technique is (Koza, 1992), and more recent developments are covered by (Kinnear 1994, Angeline and Kinnear 1996, Spector et al. 1999).

5 Structuring the Search Space

In order to be able successfully to apply the algorithms described above to the problem of finding suitable models we have to devise a suitable encoding that meets a number of criteria. The models must be: formal; flexible; sufficiently restrictable; interpretabe and quickly evaluated. We discuss these in turn.

5.1 Formality

The models have to be formally and unambiguously specified. That is for each intended model there must be only one formal representation. This does not mean that every encoded model will be distinguishable in terms of its effect on sales but that each distinct encoding has a distinct and unambiguous interpretation.

5.2 Flexibility

The encoding has to be flexible enough to include the sort of models we want to find. If one restricts the sort of models too much the all one is doing is finding the best parameterization for a model one has already designed. The strength of the techniques we describe below is that it enables some of the structure of the models to be discovered.

5.3 Sufficiently Restrictable

The space of all possible models has to be restricted sufficiently to give the algorithm a reasonable chance of success using available computational resources. If one has a flexible encoding then the number of possible models will be huge. Evidently, there is a tradeoff between the number of possible models (due to the flexibility of the encoding), and restricting the number of models in the interests of tractability. The virtue of evolutionary algorithms is that they allow for a more efficient trade-off between flexibility and search difficulty than do previous search techniques.

There are theoretical reasons why there is no completely general search algorithm that is in some way better than all the others for any search space (Wolpert and Macready, 1995). In other words, the use of domain knowledge is essential for efficient and hence practical search for models. The trick is to design an encoding that uses domain knowledge to fix those elements that one is reasonably certain about. This is the motivating force behind the basic structure of the encoding we present below in section 4, where we fix those elements that practitioners feel more certain about (the relevant attributes and the brand's values of these) but allow flexibility over elements subject to palpably geater uncertainty (the nature and number of purchasing clusters).

5.4 Interpretable

The models generated by the algorithm must be interpretable by marketing practitioners. To this end, the encoded solutions must correspond to a language of talking about such markets that makes sense to them. The system we describe is most profitably used when there can be an effective dialogue between the expert and the model outputs. This aspect is touched upon in section 7.

5.5 Quickly evaluated

Each model has to be able to be evaluated against the data quite quickly. Evolutionary algorithms work by the rapid but "dumb" parallel evolution of a population of solutions. Typical runs of such algorithms involve populations of thousands evolving over thousands of generations. The computational time taken by evolutionary algorithms is totally dominated by the time to evaluate the fitness of each solution. Thus these algorithms are only effective if the time to evaluate each potential model (encoded as a solution) is reasonably fast.

6 An Encoding of Consumer Preferences to Allow their Automatic Evolution

Below we specify the framework for potential models that we have developed for the algorithm to work upon, i.e. the meta-model in figure 1. These are designed to be as credible as possible in marketing terms given the above criteria.

As a result of this compromise the framework will be flexible in many ways but somewhat artificially constrictive in others. This is especially true of the exact functional form of some of the equations, which are necessary to ensure the computational efficiency of evaluating potential solutions but which do have relevant shapes that are sufficiently parameterised so as to be able to be appropriately fitted.

6.1 General structure and assumptions on the meta-model

Assumption 1: *In each market there will be a number of dimensions representing the relevant attributes that the consumer uses to decide amongst competing products.*

There can be any number of these both of numerical and binary type. These dimensions are selected by the practitioner concerned as those that are relevant. It is important that all the dimensions that are important for distinguishing products are included but it is not critical if some irrelevant ones are included[4]. Price is almost always included as a relevant dimension. These attributes are intended to be those as perceived by customers – they are not necessarily linked to physically measurable attributes and can be quite abstract. For example one dimension could be whether a beer is imported or not and another how expensive the product is perceived to be.

Assumption 2: *The perceived values of the attributes of the products are known.*

That is, once the relevant dimensions have been decided then the rough position of the products in terms of their perception by consumers in terms of these attributes are known.

Assumption 3: *There are meaningful clusters of purchasing decisions that will have broadly similar goals in terms of these dimensions.*

These clusters correspond with potential customer purchase contexts rather than customers. For example a customer could desire different attributes of a bottle of spirits if buying for a social event and when buying for themselves as a reward for some achievement. There can be any number of these clusters.

Assumption 4: *The desirability of the product (in the absence of competition) is strongly related to the price and the extent to which the product meets the ideals of the cluster on the relevant dimensions separately.*

This is not such a restrictive assumption as it may seem as the dimensions act independently and they could be quite abstract such as *expensiveness* (which could be modelled as the average price) or *the extent to which an item is a bargain* (which could be modelled as the extent to which the current price is less than its average price). However it does rule out extremely non-linear combinations of dimensions, for example where *a single cluster* desires one attribute on one dimension and another on a second dimension *but not both together*. In the application to a market for spirits reported below, for example, the clusters induced could be characterised as: social, functional and reward. The attributes of the products sold in that market which were specified by the marketing professionals as uniqueness, specialness and expensiveness. Expensiveness is not the same as price or relative price since an "expensive" drink can sometimes be acquired (relatively) cheaply in a sales promotion. Also expensiveness might have an upward-sloping demand curve as opposed to a typical downward-sloping curve for price.

The next assumption concerns the distance between competing products.

Assumption 5: *The extent to which products compete is strongly (and inversely) related to the "distance" between products expressed in terms of their attributes in terms of these key dimensions.*

[4] The algorithm will discover this irrelevance, resulting in clusters that are extremely tollerant to deviations in these attributes. The inclusion of irrelevant attribute dimensions will, however, make the search harder as each extra attribute dimension expands the search space.

That is to say that items that are perceived as having very different attributes in these relevant dimensions will not be strongly competing with each other and ones more similar will be more strongly competing. There are more assumptions about the relevant attributes and shape of this distance function which we discuss below.

Assumption 6: *When products are sufficiently close to each other (in terms of perceived distance), the extent to which sales can be "poached" will depend on the closeness, the price and the extent to which the product meets the ideals of the cluster.*

Note that these effects do not have to be independent of each other; for price can also be a attribute used to judge distance between products and also their general desirability in a non-decreasing way.

6.2 Parameterising the preference functions of groups

We make several assumptions about the preferences of these groups

Assumption 7: *Each group has an ideal, such that it desires the product more the closer the attribute of that product matches its ideal.*

Note that we do *not* assume that these ideals are very important in all dimensions, hence the next assumption.

Assumption 8: *Groups will have different tolerances to deviation from this ideal in the extent to which their desire for the product decreases with this deviation.*

Assumption 9: *Groups will have different residual desires for products when there is a large deviation from their ideal.*

That is to say that some dimensions will be of the nature of an 'optional extra' so that even if the desired attribute is not at all present they still have a high basic desire for the product. For other groups, products and attributes there will be a sharp 'cut-off' point beyond which the group would not consider purchasing the product at all.

We have chosen a space of transformed normal curves as the basis for our preference functions in any one dimension for a group. This has the advantage that there is a region around the ideal where small deviations from the ideal are not perceptible and that the effect of deviation can drop of slowly with large deviations. In trials it was found that the exact shape of this function was not critical for the overall technique, for example it was found that a triangular distribution about the ideal did almost as well.

The preference function (for each cluster for each attribute) is thus parameterised by a triple of real numbers: an *ideal* value, a index of *tolerance* for deviations from that ideal and an index of the extent to which that attribute is critically important for that cluster, called here the *criticality* index. In Figure 5, the effect of different tolerances on the preference function are shown. In this figure, p_S on the vertical axis is the preference index of attribute value c for the purchasing clusters in the cluster s. The domain of p_S is the unit interval. The value of c is represented on the horizontal axis. The range of actually occurring values of c (for the products concerned) is mapped onto the unit interval [0,1] for convenience – thus it does not cover all possible relevant values of the attribute, and so appears truncated. Clearly, for a given ideal attribute value c^*, the dashed preference distribution entails more tolerance to deviations from the ideal than the solid-lined distribution.

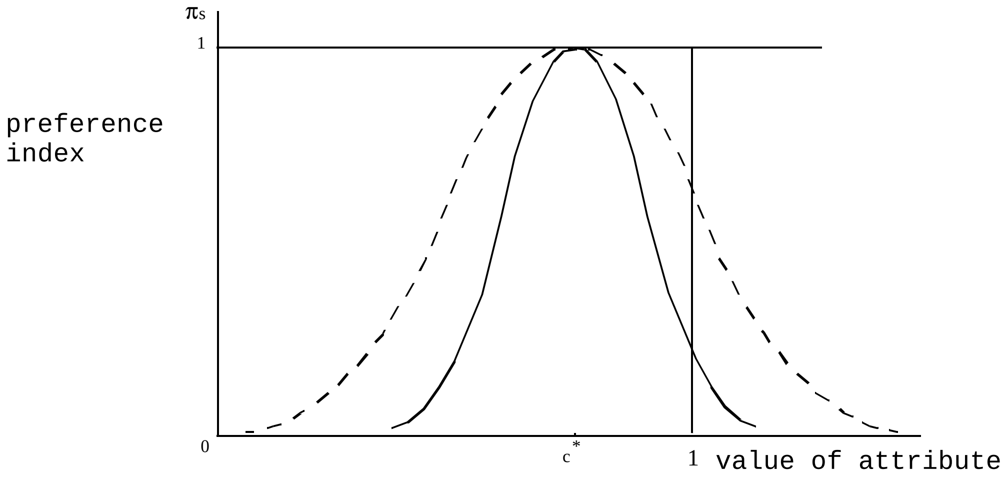


Figure 6 Preference distribution (same attribute ideal, different tolerances)

In figure 6, we show some distributions differ only in their degrees of 'criticality'. The distribution has the least 'criticality', this function is such that even if there is a large deviation from the ideal there is a considerable residual desire for that value of the attribute, corresponding to the situation where a attribute can effect decisions but will not rule out certain purchases. Preference functions with a criticality of 1 (the dashed line in figure 6) will drop to zero slowly like a normal distribution (although not all of the range will be relevant to the current choice of products). A preference function with a criticality of more than one will drop to zero with a certain deviation from the ideal, in which case the product would not be considered for purchase regardless of the values of the product in other ways – this allows the delimitation of markets by attributes.

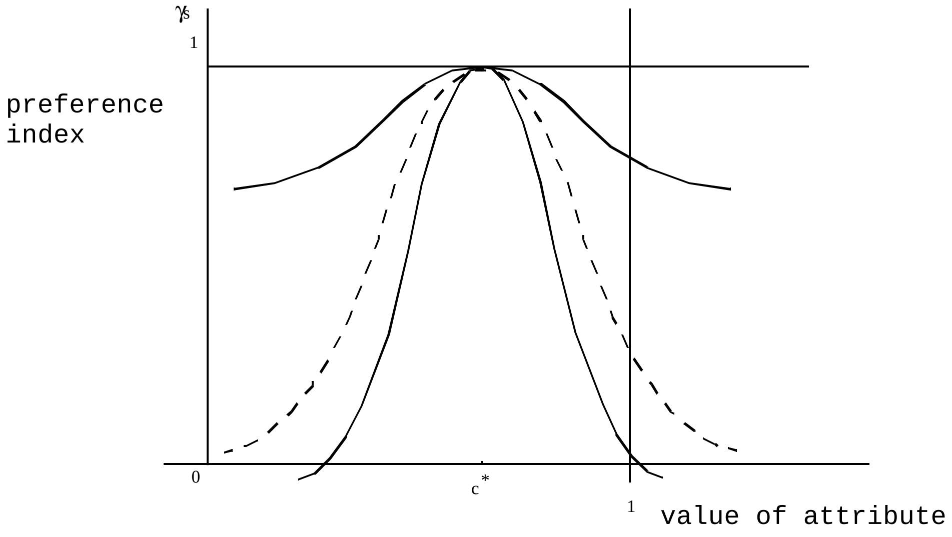


Figure 7: Preference distribution (same attribute ideal, same tolerances, different degrees of criticality)

The mathematical form of this function is:

(1) $$g = \frac{2m_c e^{-36\left(\frac{c^* - c}{t_c}\right)^2} + 1 - m_c}{1 + m_c}$$

where c is the value of attribute C, c^* is the ideal value, m_c is the index of the criticality and t_c is the corresponding tolerance index. This is merely a parameterised version of the equation of a general normal curve. In practice, we have found that we get good empirical results with these models by mapping tolerance indices into the unit interval and criticality indices into the [0, 2]-interval.

This parameterisation allows an sufficient range of functional shapes, representing most of the natural approximations one would require for this granularity of modelling, but at the same time it is easy to compute and has a natural interpretation in terms understandable by marketing practitioners.

6.3 Combining the effects of preference functions for different attribute dimensions

One of our concerns in specifying this framework was that separate markets should be able to emerge. Hence we want the combination of the desirability along different dimensions to combine so that there may be no overlap between the preference functions of different clusters. For this reason we decided to combine the preference functions for individual attributes multipicatively. This means that if any of the individual preference functions drops to zero for the value of any particular attribute of a product then the overall preference index for that product also drops to zero. For example in the market for spirits, if the perceived strength (as in alcohol content) was sufficiently low it probably would not be considered by common purchasing clusters for spirits regardless of its other charms – it would simply just not count as a spirit.

The preference index corresponding to the cluster *s* for product *b,* denoted G_{sb}, is the product of the preference indices for actual attribute value associated with the product. Formally,

(2) $$\Gamma_{sb} = \prod_{c \in C} g_{sc}$$

where *C* is the set of defined attributes. These indexes are then scaled for all the products in the set being considered so they sum to one. Thus, the *strength* of product *b* w.r.t. cluster *s* is

(3) $$\breve{s}_{sb} = \frac{\Gamma_{sb}}{\sum_j \Gamma_{sj}}$$

6.4 The distance metric

Now we come to the part of the model framework that concerns the interaction of competing products on the purchasing clusters. To make the above assumption concerning distance between products operational we need a distance metric. The basic distance metric used was the Euclidian distance in the space of attributes. The exact distance function was not found to be critical to the models (probably because there is already implicit scaling of the attributes to an appropriate scale by the range of existing product attributes available) – it would be possible to introduce different scaling parameters for each dimension, but this would increase the search space for models in return for little practical benefit in terms of model fit.

If two products are both very different from a third, how different they are from one another is not usually relevant to the consumers' product choices. We therefore used a squashing function giving us a distance measure which made increases in small distances more important than the same increases in large distances. The function used in the model reported here was:

(4) $$d_{ij} = \tanh\left(2\left|\Theta_i - \Theta_j\right|\right)$$

The shape of this is illustrated below in figure 6. The 'squashing' factor of 2 was chosen on entirely pragmatic grounds in that it gave us the best fit to the data. The 'tahn' function is merely a convieniant squashing function, for the exact shape is not critical to the results.

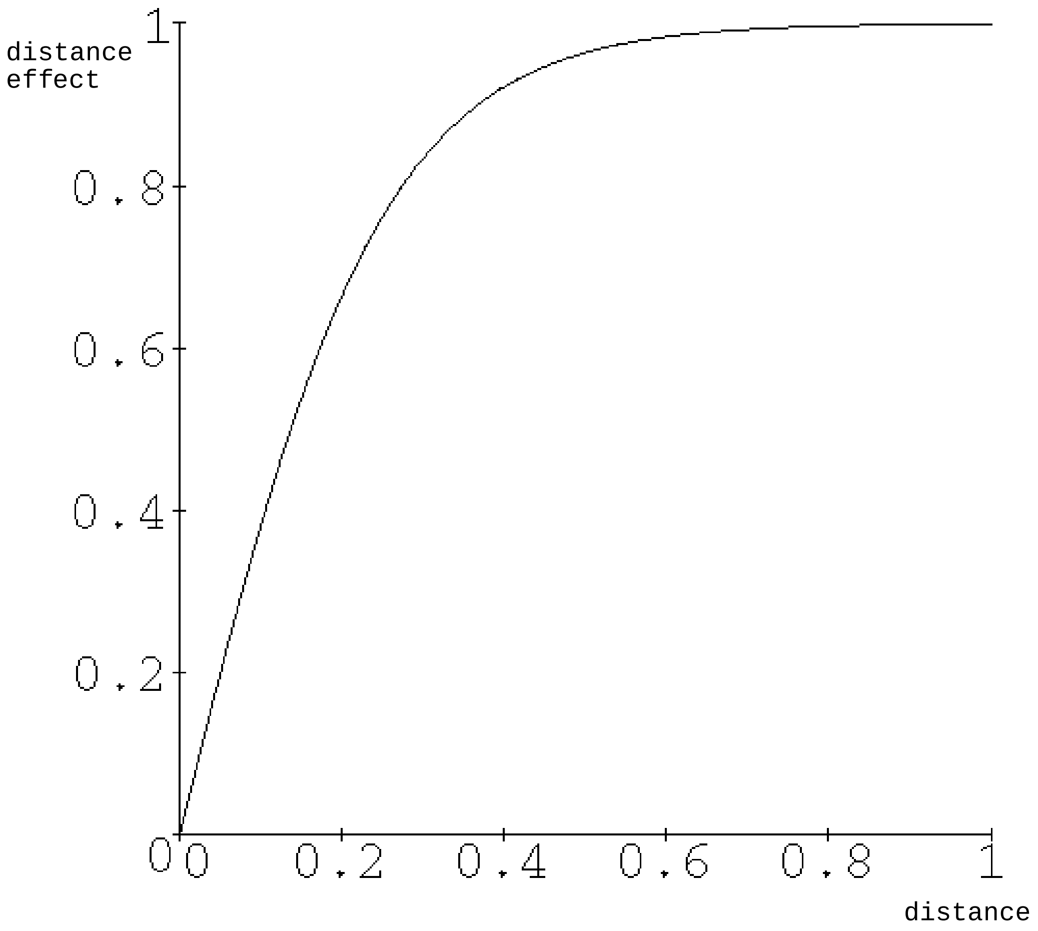


Figure 8: Effect of the squashing function on the distance metric

Because product differentiation need not have the same impact in all markets, we specify the differentiation effect as being determined by the distance between the products in attribute space and a *differentiation value parameter* (DIP) to be denoted as I_d. The differentiation effect expression is

(5) $$d_{ij} = e^{-(I_d d_{ij})^2}$$

where δ_{ij} is the distance between products *i* and *j* in attribute space. This is a scaled index of relevance of one product to another, the shape of the function is illustrated in figure 8.

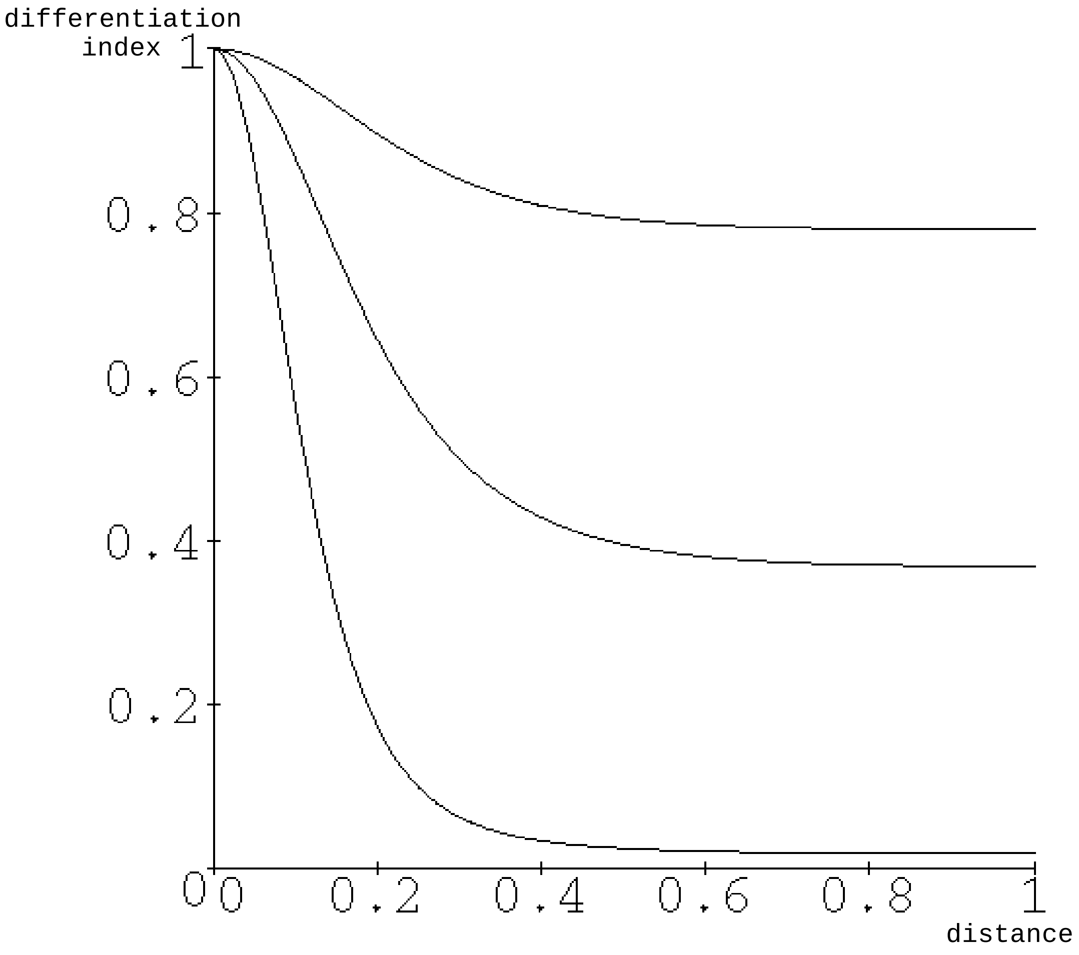


Figure 9: : Graph of the differentiation index in terms of distance for different DIPs

6.5 The reach function

So far the picture being built up is fairly static. The preferences functions only depend upon the extent to which a products characteristics match those of the consumers preferences. We now add the effect of competition. The idea is that one product is *vulerable* to a second if the second is close enough to be a competitor of the first but is superior in terms of either price or by having more desirable characteristics in terms of the attributes. The vulnerability of a product with respect to another results in sales lost to that product. In this we have broadly followed on from the ideas in (Bronnenberg and VanHonacker 1996).

In order to capture these ideas in a model we define a function which we call *reach*. This is an index of the share which one product takes from another. *Reach* is larger the greater the relative market strength and the lower the relative price. But the effect of either market strength or price is greater if the products are similar (in terms of the *distance* between them).

Denote by r_{ij} the *reach* of the *i*th with respect to the *j*th of a set of *n* products. Since we intend to use this concept of reach to determine the volume shares of the various products, all *n* of the products must be similar in the sense that their quantities can be measured in some common unit such as litres or grams or, in the case of non-financial services, person-hours.

Formally,

(6) $$r_{ij} = r_{\mathrm{ij}}\left(\frac{s_i}{s_j}, \frac{p_i}{p_j}\right)$$

where

(7) $$\frac{\partial r_{ij}}{\partial\left(\frac{s_i}{s_j}\right)} = r_s\left(\frac{s_i}{s_j}, \left|\Theta_i - \Theta_j\right|\right) > 0$$

(8) $$\frac{\partial r_{ij}}{\partial\left(\frac{p_i}{p_j}\right)} = r_p\left(\frac{p_i}{p_j}, \left|\Theta_i - \Theta_j\right|\right) < 0$$

(9) $$\frac{\partial r_s}{\partial\left(\left|\Theta_i - \Theta_j\right|\right)} < 0$$

(10) $$\frac{\partial r_p}{\partial\left(\left|\Theta_i - \Theta_j\right|\right)} < 0$$

Verbally, the reach of one product with respect to another is determined by their relative strengths and relative prices. Naturally, reach increases with relative strength (inequality (8)) and diminishes with relative price (inequality (9)). The sensitivity of reach with respect to relative strengths and to relative prices diminishes as the products are less similar (inequalities (10) and (11)).

Because we represent the value of each attribute for each product as a real number in the unit interval, the coordinates representing the position of a product in attribute space is always in the unit hypercube of dimensionality equal to the number of attributes. The maximum distance between any two points (corresponding to the diagonal of the hypercube) is the square root of its dimensionality — in this case the square root of the number of attributes. It is therefore natural to normalize the distances between products' positions on the square root of the number of attributes. In this way, the model is not sensitive to the size of the chosen attribute set.

We thus split the reach function into components: the price effect and the strength effect. Given that in different markets the price effect and the product strength will have different strengths, we introduce two new parameters: the strength value parameter and the price effect parameter.

The price effect is a standard economic demand function. The effect of the relative prices is

$$\Pi_{ij} = e^{-\frac{p_i}{p_j} d_{ij} I_p} \tag{11}$$

where I_p is the price value parameter (PIP).

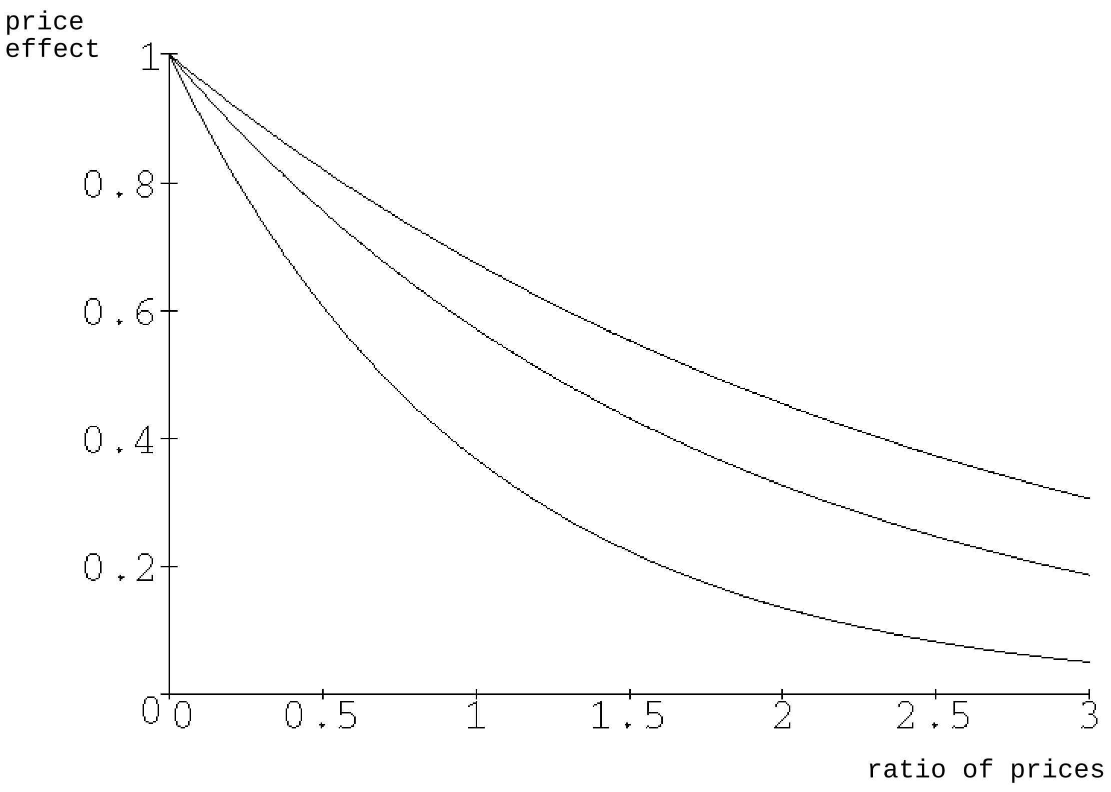


Figure 10 Graph of the price effect in terms of the price ratio for different distances

The effect of the relative strengths of two products with respect to a cluster, *s*, is

$$\Sigma_{sij} = \frac{\tanh\left(\left(\frac{S_{si}}{S_{sj}} - 1\right) d_{ij} I_s\right) + \tanh\left(d_{ij} I_s\right)}{1 + \tanh\left(I_s\right)} \tag{12}$$

where I_s is the *strength value parameter* (SIP) and is the distance effect index introduced above. The larger the value of the SIP, the higher the value of Σ_{sij} for any value of the strength ratio. The strength effect is just a shifted logistic curve, its range is from 0 to 1 and it gets steeper when the distance effect is greater (i.e. products are closer to each other). This is shown in Figure 7.

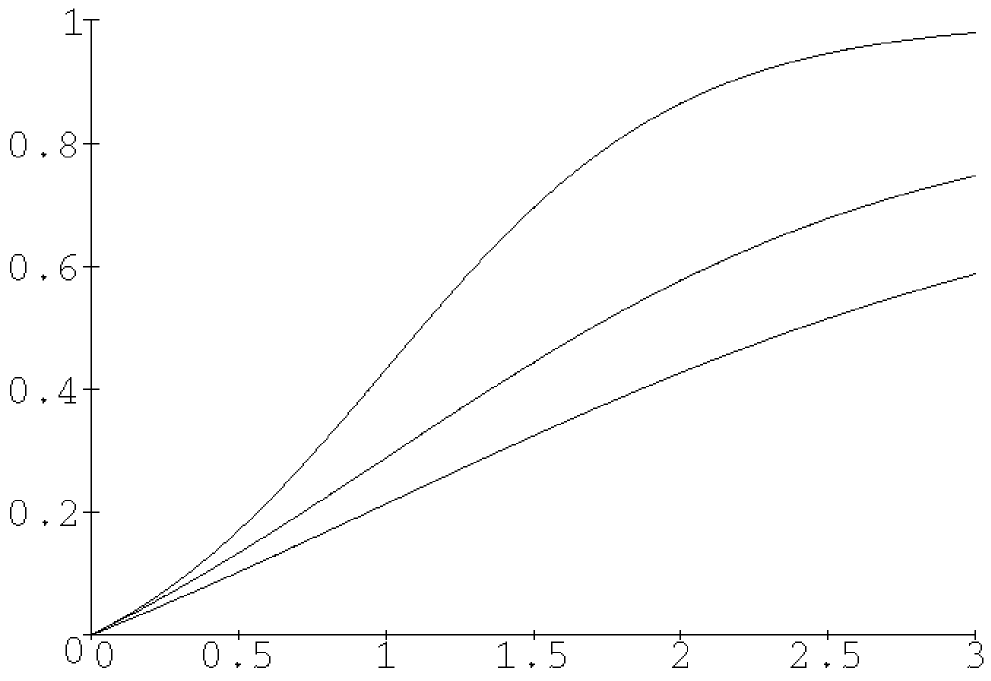


Figure 11: Graph of the strength effect in terms of the strength ratio for different distances

Finally the price and strength effect are multiplied together to give the reach of one product over another:

(13) $$r_{sij} = \Sigma_{sij} \Pi_{ij}$$

The overall effect of the ratio of prices and market strengths on the reach is shown in figure 11 and figure 12, for two different distances between products. In the first (where the products are almost identical in terms of attributes) we see how the reach of one over the other increases sharply with an increase in the ratio of the products' prices or desirabilities. In the second the reach is much reduced due to the distance between the products.

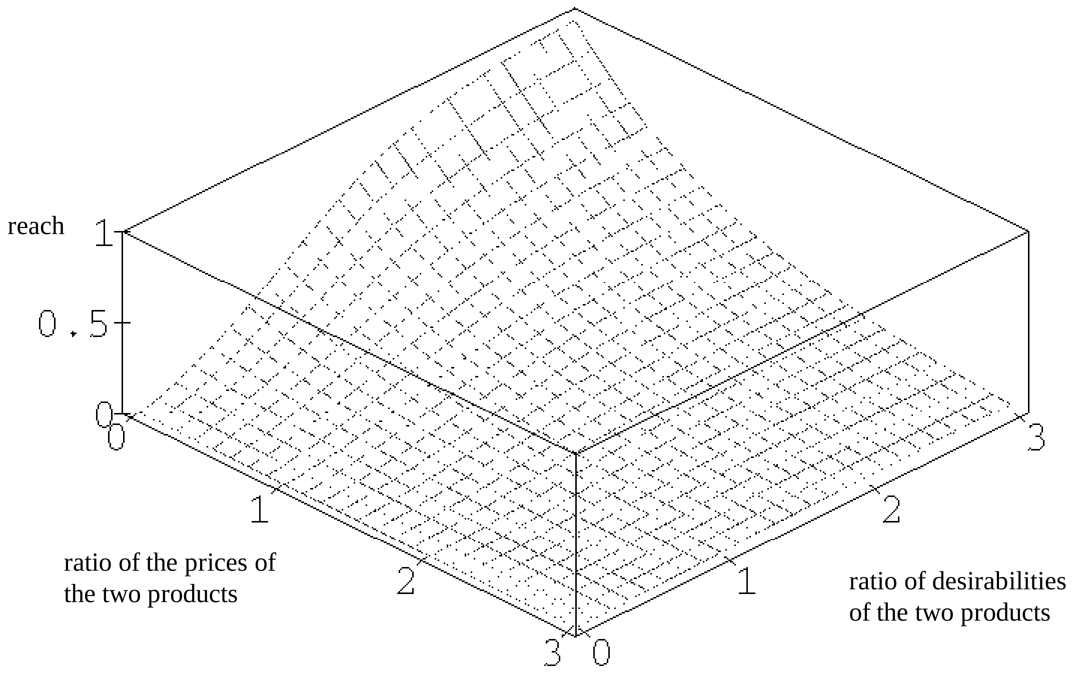


Figure 12: Graph of the reach, with distance=0

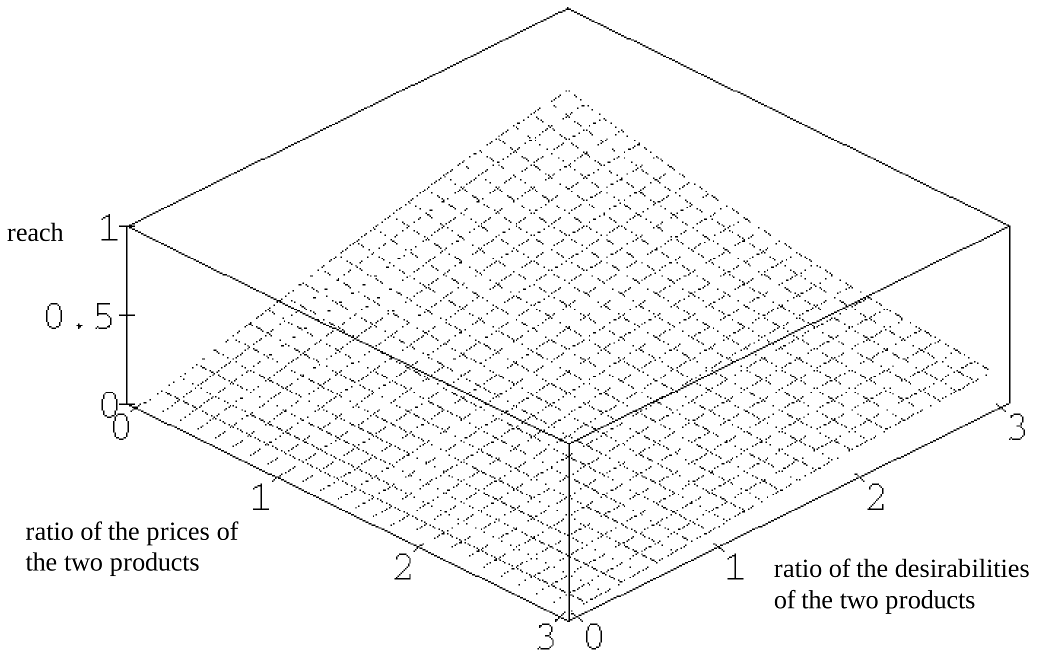


Figure 13: Graph of the reach, with distance=0.5

Finally the reach function of product *i* over product *j* is the weighted sum of the reaches over all the clusters. This represents the total reach of one product over another, in other words this represents the proportion of sales lost to each of the other products.

(14) $$r_{ij} = \sum_{s \in S} w_s \, r_{sij}$$

where *S* is the set of purchasing clusters.

6.6 Market shares

The notional demand index of each product product has two components: the first represents the number of sales generated by the product product's *strength* with respect to each cluster, but this is modified by the proportion of these notional sales lost to other products according to their *reach* over it.

Thus the proportion of sales lost to all other products is given by the product of the proportion lost to each of the others:

(15) $$R_b = \prod_{j \neq b} r_{jb}$$

Now the notional demand is the sum over clusters of the strength (weighted by their size) reduced by this loss rate.

(16) $$\Delta_b = R_b \sum_s w_s \, \hat{s}_{sb}$$

Finally, to get the simulated market shares we normalise these notional demands to the size of the market.

(17) $$m_b = \frac{\Delta_b}{\sum_i \Delta_i}$$

This gives the model's prediction of the market shares of each product at each time.

6.7 Specification of a model

In order to specify a complete context-dependent attribute preference model (CDAP model) within this framework, the following have evidently to be determined:

- PIP, SIP and DIP parameters for the market (equations:)
- The relevant attributes to consumer choice in this market *
- The products *
    - for each product:
        - the perceived value of that product of each of the attributes *
- the purchasing clusters
    - for each cluster:
        - a weight, indicating the size of the cluster
        - for each attribute
            - its ideal value of that attribute
            - its tolerance to deviations from the ideal of that attribute
            - the criticality of that attribute to its choice

Of these, the marketing practitioners are typically more confident of the relevant attributes, the products and their perceived attributes (i.e. those marked with an asterix above). They seem less sure of the anything pertaining to the purchasing clusters. The three market parameters are unknown, being model scaling constants.

Thus in our algorithm we got involved practitioners to input the attributes and the perceived values of these attributes for each product on artificial scales. Then the algorithm described below looks for models to fit this information and the aggregate market and price data. It is feasible for any of the above data to be specified and the computer to search for models by specifying the other data, but this seems to be the way that best combines a match with the degree of certainty to which the practitioners attach to their knowledge and sufficiently restricting the search space to ensure a reasonable output from the algorithm.

7 Implementation details of the algorithms employed

If it is the case that the domain experts are dissatisfied with their present set of CDAP models, they may want some new models to adapt and work from. Typically these domain experts are fairly sure about the relevant properties in a particular market and the perceived attributes of these properties for each product, what they are uncertain about is the number, identity and preferences of their customers. Thus there is a need for an algorithm which, given the relevant product attributes, automatically searches for CDAP models that are consistent with the known data. An algorithm which we have found to be effective is described below - the Automatic CDAP Honing Engine (ACHE).

The heart of the procedure for determining a credible CDAP model from the sales data is a genetic programming algorithm. This can be made more robust with a random search front-end to ensure a viable initial population of possible models and then a final hill-climbing algorithm afterwards to tune the models found. Here we will just describe the basic tecniques.

7.1 Genetic programming module

Genetic programming (GP) differs from the familiar genetic algorithms in that the gene is a labelled tree rather than a string. The basic GP algorithm is:

1) Specify the possible branching and terminal nodes that the trees can be built from and the fitness function for evaluating them.
2) Generate an initial population of random trees of a given depth using these nodes.
3) Evaluate this population using the fitness function.
4) Find the best gene and, if it is good enough, stop.
5) Otherwise generate a new population of trees using one of two methods (according to a fixed proportion determined by the programmer):
   a) drawing pairs of trees randomly from the current population with a probability related to their fitness and producing two new offspring by choosing a random node in each and swapping the sub-trees that are rooted at these nodes (tree-crossover) or,
   b) randomly choosing trees with fitness-related probabilities for propagation to the new population.
6) Go to step 3.

In our case the tree-structure covered possible CDAP models. A gene was an instance of the following specification:

```
gene := IP list, weight list, CDAP list,

IP list := price value parameter, strength value parameter,
differentiation parameter

weight list := list of non-negative numbers (of same length as list of
CDAP states)

CDAP list := list of CDAP specifications, one for each CDAP state

      CDAP specification := list of preference specifications, one for
      each property

            preference specification := a triple of numbers: the ideal
            value, its criticality and the tolerance to variation
```

This corresponds to the list of data enumerated in section 4.7, above.

The fitness function was the RMSE error of the predicted market shares compared to the actual shares over a sample period for the competitive set with a small discount to bias the algorithm in favour of models with fewer CDAP states.

Our crossover operator was constrained to produce only well-formed genes, i.e. if one chosen sub-tree was a preference specification the other would be also. Also if the domain expert had previously entered any trial CDAP models, these would be seeded into the initial population, so that variations of these would be tried along side the randomly generated ones.

7.2 Competitive set front-end

One problem we encountered is that although practitioners may know the total set of products in any market and, for their own products, have strong and well articulated views about the main competing products, they are uncertain about the effects of other products on their own. Putting the complete range (perhaps 2500 products for which data is available) into the algorithm would needlessly waste computational time since most of these products have little effect on one another. Moreover, practitioners frequently want a model centred around a particular product. For these entirely pragmatic reasons, we added a front end which applies statistical algorithms to the full EPOS data set to filter out products characterised by insignificant (though not typically symmetrical) cross price elasticities with a designated *focus* product.

A three-stage filtering algorithm was developed to identify the set if competitors of any, arbitrarily focus product. At each stage, the marketing practitioners were able to retain products discarded by the algorithm or discard retained products. The stages were:

1. OLS regression of market share of the focus product on the relative price in logs of each of the other products for which data is held as well as regressing the shares of the other products on the same price variable of the focus product. Products were retained if both t-ratios exceeded a critical value and discarded otherwise. The critical value was chosen to yield anough degrees of freedom for the second stage.
2. Multiple OLS regressions on the logs of the prices of all remaining products and the log of total sales volume. This was an iterative process in which the product with the lowest standard error on its price coefficient at each iteration was discarded until the coefficients on all of the log prices were significant at the 99% confidence level. The regression equation used in this stage was taken from the Deaton-Muellbauer AIDS algorithm but without the symmetry restriction. In general, the marketing professionals were interested in the half-dozen or so most important competitors. Leaving 15 to 20 products in the competitive set at this stage gave the practitioners confidence that all of the most important six to eight competitors were included for the third stage.
3. Further elimination of products from the competitive set together with analysis of the changes in competitive structures over the data period was based on a non-linear generalization of the second stage based on the local regression algorithm of Cleveland and Devlin (1988).[5] The particular advantage of this stage was that it yielded a time series of cross-price elasticities indicating that some products appeared significant in linear regressions because of a few large and systematic fluctuations in volumes and prices due to special offers or other ephemeral events during the observation period. A rulebase was developed to identify such products and also to identify products that were becoming less competitive with the focus product. All such products were discarded from the analysis.

8 Applying the Technique on Markets for Liquor

The technique was initially tested on data from the UK market for liquor. This consisted of the aggregate volume and average price for 4 products[6] in each of 2 sizes

[5] We actually used a refinement of this technique developed by our collaborator on this project, Michael Campbell. A full description of the algorithm used is reported by Campbell *et al.* (1997).

[6] One of these 'products' is the pseudo product of all the 'Own Label' products conflated together.

over 96 weeks. A marketting practioner suggested that the relevant attributes could include: relative price, expensiveness (a long-term average of the price), size, 'specialness' (ranging from the well-known to a special treat), and 'uniqueness' (ranging from the ordinary to the starkly different). The practioner estimated the percieved values of the specialness and uniqueness attributes for each product on a Likehart scale of (low, below average, average, above average, and high) which was mapped onto the values [0.1, 0.3, 0.5, 0.7, 0.9]. The other attributes were scaled so that their values mapped onto the [0,1] interval (except price which was mapped onto the [0,0.1] interval so as not to contribute much to the percieved distance between products).

The plot for the real and simulated market shares for these five products is shown in figure 15, below. The model was learnt on only the first 21 weeks of data. We see that the simulated shares track the actual shares well for well over a year. The RMS error was 2.8% on the 76 weeks of out-of-sample data over all five products.

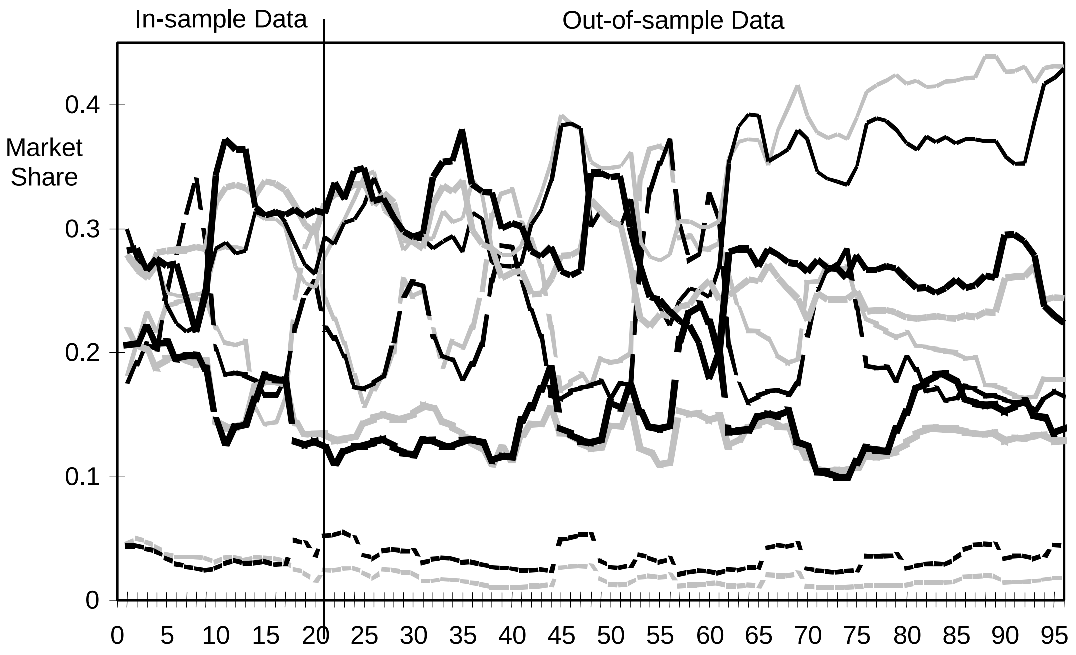


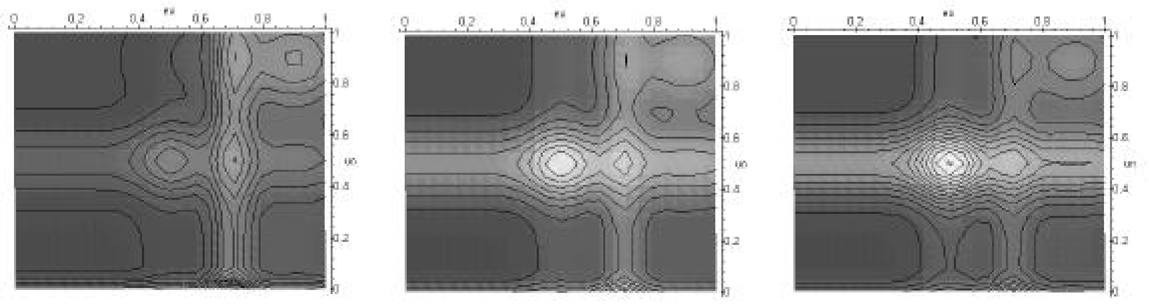
Figure 14: Real vs. Predicted Market Share (black=actual, grey=model generated)

Table 1 shows the attributes for the three clusters found for this market. The entries are scaled from 0 to 10 for ease of reading. The bold entries are those where the cluster response was particularly sharp. The blank entries are where the customer response was too flat to be important. We see from this that almost half the market is determined by a cluster (cluster B) that values a low relative price and slightly larger bottles. Cluster A likes above average expense along with no uniqueness – in other words the value a well-known product that they see as being a little expensive. Cluster C has less definite preferences, but seems to be biased towards expensive and unique products. As it turns out the abstract attribute of specialness turns out not to be very important, probably because this attribute is dominated by how expensive the product is percieved to be.

Cluster	Relative Price	Expensiveness	Size	Specialness	Uniqueness
A (21%)	1	**7**	6		**0**
B (49%)	**1**	5	**8**		5
C (29%)	2	9		3	9

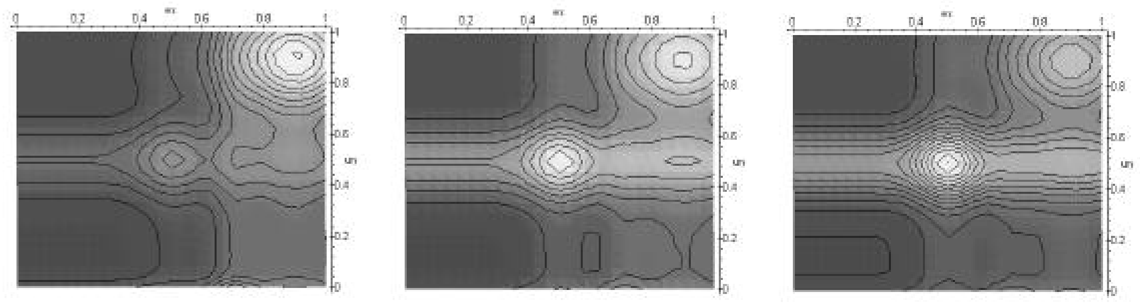
Table 1. The attributes of the three clusters found

It is not easy to visualise the underlying CDAP models. We attempt this in two ways.The first method is by plotting an index of the intrinsic desirability of various values of the attributes to the CDAP clusters, which is the weighted sum of the desirability of a product with notional attributes. This is done in figures 15 and 16.



Low Relative Price Average Relative Price High Relative Price

Figure 15. Index of the Intrinsic Desirability of various values of the attributes expensiveness and uniqueness, for three prices, size 70cl (white= high desirability, black=low desirability).

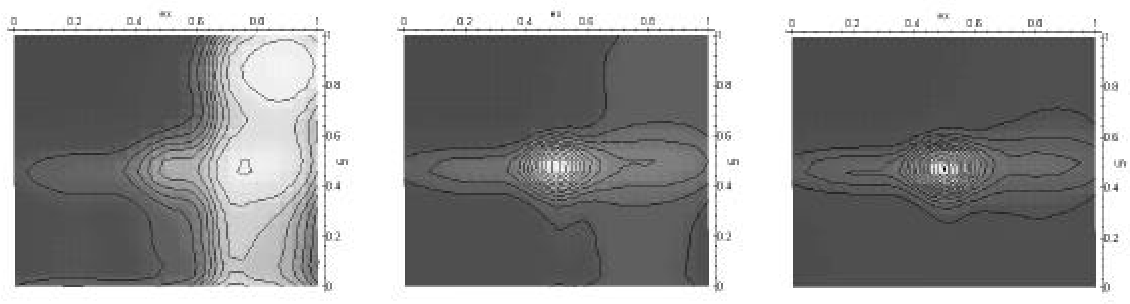


Low Relative Price Average Relative Price High Relative Price

Figure 16. Index of the Intrinsic Desirability of various values of the attributes expensiveness and uniqueness, for three prices, size 1L (white= high desirability, black=low desirability).

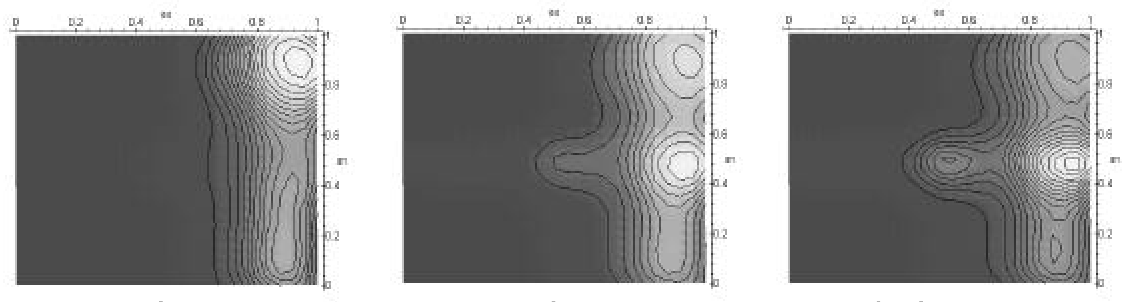
These diagrams should be interpreted with caution. *Firstly*, they represent one set of models that is consistent with the judgemental information and the aggregate data, there may be others as well. *Secondly*, the the models will only be accurately induced around the values of the attributes that were input (either by the practitioner or in the aggregate date) – for example, if all the products had the same value for some attribute, then the results for other values of this attribute would be arbitrary. This is not surprising as no induction method (automatic or otherwise) can work in regions where there is no data. *Thirdly*, the diagrams in figures 15 and 16 do not take into account the effects of the other products in the market – it would be no use positioning a product with attributes that are desirable by this index if this region was already oversupplied with products.

The second method is by plotting what demand there would be for a notional new product that was positioned with various different attributes values in such a market. Figures 17 and 18 show the equivalent plots to figures 15 and 16 but for demand rather than an index of desirability.



Low Relative Price Average Relative Price High Relative Price

Figure 17. Notional demand for a new product with various values of the attributes expensiveness and uniqueness, for three prices, size 70cl (white= high demand, black=low demand).



Low Relative Price Average Relative Price High Relative Price

Figure 18. Notional demand for a new product with various values of the attributes expensiveness and uniqueness, for three prices, size 1L (white= high demand, black=low demand).

Thus the first figure in Figure 17 indicates that give we have two 70cl bottles of liquor both having the same relatively low price, then there will be higher demand for the product percieved as more expensive. That is people buying such products like a bargain – buying an expensive bottle at a cheap price. What is perhaps more surprising is that at higher relative prices there would be more demand for a product with a medium level of uniqueness (and for 70cl bottles).

Again a great deal of caution is needed to avoid overinterpreting these plots. They represent the notional demand for a new product positioned only against the products selected by the competative set filter, on the assumption that the underlying discovered CDAP model is correct and that this new product does not change the preceptions of the existing products. But they are useful indications that might guide further research into these markets and it does illustrate how the technique can give a result that is readily interperable in terms that are meaningful to marketting practioners. The fact that the framework is somewhat specified by the practioners and the models mapped into this framework ensures this.

9 Further Research

There is obviously a lot of futher work that could be done regarding this technique. In no particular order this includes:

- A comparison of the output of the algorithm against the preferences revealed by focus groups or panel data;
- The application of the technique to compare the different models that are induced at different times or for different regions of the same market[7];
- Extending the algorithm so that it can induce against a background of changing attributes other than price;
- Further incremental testing and improvement of the speed and robustness of the underlying algorithm.

Although there is not space here, the technique has been tested on serveral other markets for alcoholic beverage, including ones with hundreds of products, on different continents, for beers as well as liquor. We intend to apply the technique in the near future to the consumption of tap water.

10 Conclusion

We have shown how techniques from evolutionary computation can be used to effectively induce consumer preference models from a framework provided by practitioners and the aggregate sales and volume data for the relevant products and it does this at the expense of only moderate computational resources. Furthermore it produces models which are readily interpretable by practitioners in terms of the attriutes of the clusters of purchase decisions. The technique thus bridges the gap between aggregate numerical data and the attributes of purchasing clusterss.

This approach has four virtues not shared by approaches which pre-specify the structure of models.

One is that the model structure owes more to the application domain and less to requirements of the statistical algorithms. Consequently, the analysis is driven to a greater extent by the data rather than the method.

A second virtue is that the technique relies to a lesser extent on the assumption that the underlying data generating processes are not changing. This feature supports the analysis of markets in which, for example, product positioning is changing for reasons that are distinct from pricing issues. A particular example where changing structures are palpably important is the market for alcoholic beverages in any of the transition or other rapidly developing economies. And in some cases the purpose of the market analysis is to determine ways in which to change the structural relations that have been observed in the past by repositioning products in their markets. In yet other cases, inadvertent repositioning has taken place by virtue of apparently unrelated decisions.[8]

A third virtue of the method reported here is its support for the coevolution of models and policy for commercial and business environments that are too complex and poorly understood for forecasting models to be credible and reliable. Markets for

[7] One practioner even went so far as to suggest comparing the results for data derived from different tills in the same supermarket!

[8] A case reported to the authors involves a product of gin the production of which was moved a few miles which chanced to include the boundary of Greater London. A competitor noted that it was no longer a "London gin" and used this change in its advertising.

consumers' goods in transition economies are a key example of such circumstances.

Finally, it permits the combination of managerial judgement with available data. For example, manufacturers of fast moving consumer goods may have retail audit data for their product class, but may not have access to disaggregated EPOS data which would permit modelling of the kind which Guadagai and Little (1998) described. By the same token it is capable of being applied in industrial or business to business marketting settings where tradittionally data has been less readilly available.

11 Acknowledgements

The system and results reported here were obtained as part of the Intelligent Marketing Integrated System (IMIS) Project conducted in collaboration with United Distillers PLC as part of the Information Systems Integration Programme of the UK Engineering and Physical Sciences Research Council and the Department of Trade and Industry under contract number IED/4/8022. The programming language used in developing IMIS is SDML, designed and implemented by the Centre for Policy Modelling using the VisualWorks 2.5 development environment supplied by Sincom Systems Ltd.. The data was provided by United Distillers who have generously supported IMIS from its inception. Clive Sims of Competitive Analysis International suggested basing demand analysis on need-states which we formalized as CDAPs. We thank our colleague Steve Wallis and our collaborators in IMIS, Michael Campbell and Paul Ormerod for their encouragement and help in establishing the programme of which this paper is a result.

12 References

Angeline, P. J. and Kinnear, K. E. Jr. (eds.) (1996) *Advances in Genetic Programming, Volume 2*. Cambridge, MA: MIT Press.

Blattberg, C.R. and Hoch, J.S. (1990). Database models and managerial intuition - 50% model + 50%. *Management Science*, **36**:887-899.

Cleveland W. S., Devlin S. J., Grosse E. (1988) Regression By Local Fitting - Methods, Properties, And Computational Algorithms. *Journal Of Econometrics* **37**: 87-114.

Bronnenberg B. J., VanHonacker W. R. (1996) Limited choice sets, local price response, and implied measures of price competition. *Journal of Marketing Research* **33**:163-173.

Dawes, R. M. (1972). *Fundamentals of attitude measurement*. New York: Wiley.

Deaton A. and Muellbauer J. (1980). An almost ideal demand system. *American Economic Review*, **70**:312-326

Goldberg, D. E. (1989) *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.

Guadagni, P.M., Little, J.D.C. (1998). When and what to buy: a nested logit model of coffee purchase. *Journal of Forecasting*, **17**:303-326

Holland, J. (1975). *Adaptation In Natural and Artificial Systems*. Ann Arbour: The University of Michigan Press,.

Horsky, D. and Rao, M. R. (1984). Estimation Of Attribute Weights From Preference, *Management Science*, **30**:801-822.

Kinnear, K. E. Jr. (ed.) (1994) *Advances in Genetic Programming*. Cambridge, MA: MIT Press.

Koza, J. R. (1992) *Genetic Programming: on the programming of computers by means of natural selection*. Cambridge, MA: MIT Press.

Meehl, P. E. (1954). *Clinical versus statistical prediction : a theoretical analysis and a review of the evidence*. Minneapolis: University of Minnesota Press.

Morgan-Jones, J. and Zufryden, F. S. (1980) Adding Explanatory Variables to a Consumer Purchase Behavior Model: An Exploratory Study. *Journal of Marketing Research*, **17**:323-334.

Moss, S., Gaylard, H., Wallis, S. and Edmonds, B. (1998). SDML: A Multi-Agent Language for Organizational Modelling. *Computational and Mathematical Organization Theory*, **4**:43-69.

Murphy A. H., Brown B. G. (1984) A Comparative-Evaluation Of Objective And Subjective Weather Forecasts In The United-States. *Journal of Forecasting* **3**:369-393.

Spector, L., Langdon, W. B., O'Reilly, U. and Angeline, P. J. (eds.) (1999) *Advances in Genetic Programming, Volume 3*. Cambridge, MA: MIT Press.

Wolpert, D. H., and Macready, W. G. (1995) No Free Lunch Theorems for Search. Technical Report, Santa Fe Institute, Number SFI-TR-95-02-010. (http://www.santafe.edu/sfi/publications/Working-Papers/95-11-101.ps)