SMRT ADVANCED SIMULATION MODULE for Product Optimization

Version 1

Sawtooth Software, Inc.
Sequim, WA
http://www.sawtoothsoftware.com

About Technical Support

We've designed this manual to teach you how to use our software and to serve as a reference to answer your questions. If you still have questions after consulting the manual, we offer telephone support.

When you call us, please be at your computer and have at hand any instructions or files associated with your problem, or a description of the sequence of keystrokes or events that led to your problem. This way, we can attempt to duplicate your problem and quickly arrive at a solution.

For customer support, contact our Sequim, Washington office at 360/681-2300, email: support@sawtoothsoftware.com, (fax: 360/681-2400).

Outside of the U.S., contact your Sawtooth Software representative for support.

Sawtooth Software, Inc.
Sequim, WA
http://www.sawtoothsoftware.com

Table of Contents

Chapter 1

Installation Procedure

Introduction

The Advanced Simulation Module (ASM) is a component within Sawtooth Software's Suite of Marketing Research Tools (SMRT). Depending on the license(s) you purchase within SMRT, different capabilities are enabled within the software. At the time of this release, the other main components within SMRT are ACA (Adaptive Conjoint Analysis), CBC (Choice-Based Conjoint), and CVA (Traditional Ratings-Based Conjoint).

Before installing SMRT, we suggest that you terminate any programs that are already running. Installing (or re-installing) SMRT will not delete or overwrite any existing SMRT data or study-related files.

Installing SMRT

1. Insert the CD-ROM installation disk into your CD-ROM drive.

2. The installation program should automatically start. If it does not, browse to your CD-ROM folder and double-click **AUTORUN.EXE**.

3. Select SMRT from the menu. By default, SMRT is installed to the following directory: **c:\Program Files\Sawtooth Software\SMRT**. You can change this directory if you want.

4. Start the software by clicking ***Start | Programs | Sawtooth Software | Sawtooth Software SMRT.*** Click ***Help | About SMRT/User ID…*** and click the ***Add User ID…*** button. Type your User ID information (usually attached to the CD's protective case and/or the Certificate of Authenticity) *exactly* as written, including upper and lower case. Click ***OK*** when finished.

Running SMRT

Click ***Start | Programs | Sawtooth Software | Sawtooth Software SMRT***, and the main menu is displayed.

Removing SMRT from Your PC

Windows® operating systems do not let you remove software just by deleting a folder. Windows applications leave traces of themselves in many places, and to remove them you should use the Uninstall procedure. However, if you try to uninstall SMRT while it is still running, the uninstall utility will not be able to do a complete job, and many files will be left on your disk.

IMPORTANT NOTE:
Before uninstalling SMRT, make sure it is not currently running.

Uninstalling SMRT

After you have exited SMRT:

1. Click ***Start* | *Settings* | *Control Panel***

2. Double-click ***Add/Remove Programs***

3. Select ***Sawtooth Software SMRT***

4. Click ***Add/Remove...***, and follow the instructions.

If you created questionnaires or developed market simulation scenarios before uninstalling SMRT, the questionnaire files, simulation settings, study data, and other related study files will *not* be removed by the uninstall program. This protects you from losing important data or files related to studies you may be working on, or studies you've already completed. If you no longer want these files, you can delete them manually.

Chapter 2

New Capabilities in Advanced Simulation Module (ASM)

The Advanced Simulation Module extends the capabilities of the standard market simulator within Sawtooth Software's SMRT platform. New capabilities are listed below.

1. **Optimization Searches.** In standard market simulations, researchers create market scenarios and estimate shares of preference for fixed alternatives within those scenarios. The ASM allows the researcher to search for optimal products based on utility, share, purchase likelihood, revenue, profit or cost minimization. These searched products can be optimized vis-à-vis a set of fixed competitors, or without regard to a competitive set. For profit or cost minimization searches, the ASM includes a new area for specifying costs associated with levels in the study. The new functionalities leverage the existing point-and-click interface within SMRT. No "programming" is involved to set up and run sophisticated product search scenarios.

2. **Improved Sensitivity Simulations Area.** Sensitivity simulations allow the researcher to execute multiple simulation runs at once, usually to vary a product across all levels of, say, price, and report the shares at each price point. With the ASM, you can additionally choose multiple products within sensitivity runs. With the choice of multiple products and multiple attributes, there are two new modes for running sensitivity analyses: exhaustive, and parallel. There is also a new area for creating "price associations." These are fixed incremental increases or decreases to the price of a product based on other levels in the study. For example, the researcher may want to always increase the price of a vehicle by $1,500 if a V6 engine size is included rather than a V4.

3. **Simulation Capacity Increased.** The number of products that may be entered in a market simulation scenario has been increased from 30 to 100.

4. **New Rescaling Options for Average Utility Reporting.** The ASM includes the ability to choose from Raw (no rescaling), Points, Diffs, and Zero-Centered Diffs scaling for presenting the part worth utilities in the simulation output report. This new option is available by clicking the ***Advanced Settings…*** button, from the ***Scenario Specification*** dialog.

Additional Notes: SMRT v4.5 (the standard simulator) and the ASM both include an improved random number generator for use in the Randomized First Choice engine. As a result, the simulated shares will differ slightly from earlier versions of SMRT if using the RFC simulation method. With the increase in the number of product alternatives that could be included in a simulation, the auto-calibrating attribute error function used in RFC required updating, which can lead to small differences in the resulting shares relative to previous versions.

Chapter 3

Product Optimization: Motivation and Methods

Introduction

Conjoint analysis is used by many marketing organizations to assess the likely degree of success of potential new products. Conjoint analysis assumes that an individual's liking for a product can be approximated as the sum of "part worths" for its separate attribute levels. A conjoint questionnaire conducts a designed experiment for each individual, providing data from which it is possible to estimate his or her part worths.

Those estimated part worths can be used in choice (market) simulations. The simplest simulation specifies several competitive products in terms of their attribute levels, and then predicts which of those products each respondent would prefer. Such results may be used to estimate market share for hypothetical new or modified products, as well as their potential revenue and likely profitability. In the absence of competitive products, conjoint data (calibrated using purchase likelihood data) can also be used to simulate respondents' likelihood of purchasing specific products.

Conjoint analysis has been enormously successful since its introduction in marketing research more than 30 years ago (Green and Rao, 1971), largely because of its simulation capability. Simulators use part worth data, which can be difficult to understand for many managers, and convert them into product shares of preference—resembling market shares—that are easy to understand and immensely practical for managers. Simulators represent the best conjoint has to offer in terms of assessing attribute importance and sensitivities, complex interaction or substitution effects, and the likely success of products given certain competitive conditions.

Toward Optimization

Marketers often turn the question around: rather than ask "How good would ***this*** product be?" they often ask "What product would be ***best***?" The Advanced Simulator Module (ASM) has been developed to answer such questions.

Sawtooth Software is by no means the first to consider how one might search for products that optimize market share or profitability. Green and Krieger (1993) considered the same problem years ago, proposing a method similar in many ways to what we have done.

Conjoint simulators provide the best means to date for product optimization. They can take into account the characteristics of currently-available products as well as the desires of a heterogeneous population of potential buyers. Subject to reasonable caveats about the quality of respondent sampling and questionnaire design, conjoint simulations can accurately assess likely product success long before a product is ready for test market. The ASM can optimize based on utility, purchase likelihood, market share, revenue or profitability; however, profitability optimization requires additional user-provided information about feature costs. If you include feature costs, the ASM can also perform cost minimization searches, searching for products that meet some threshold of utility, share, revenue, or profit while minimizing cost.

Finding the best product by manually specifying many simulations would be difficult because there can be such a large number of potential products to evaluate. Suppose products are described by 10 attributes, each with 5 levels. Then the number of possible products (ignoring the possibility of interpolation between levels) would be 5 to the 10th power, almost 10 million. Although each of those possible products could be simulated, examining all combinations would take too long to be workable. When we consider that one may wish to optimize multiple products simultaneously, even situations with few attributes would be infeasible for manual search.

Another possible approach would be to configure a product with attribute levels that are most desirable on average. But that would fail in two ways. First, it would almost certainly choose an unprofitable product because it would select products with many desirable features and the lowest possible price. The second reason such a method would fail is that it does not recognize heterogeneity in the desires of the market. Marketers know that the most successful products are those that appeal to buyers who are not already satisfied by existing products, and hence existing products must be taken into account.

The ASM uses heuristic search strategies to find the optimum (or near-optimum) product or a portfolio of products. It can optimize several kinds of objectives, including estimates of market share, total revenue, profitability, purchase likelihood, and total utility. It does this by exploring the "response surface" of the objective, such as share, corresponding to attribute levels for the product(s) of interest.

Although most response surfaces are multi-dimensional, it is useful to imagine there are only two independent variables. Suppose a product category has two attributes, both continuous, and that we are interested in market share. Then we could represent share estimates from many simulations by a three-dimensional model. The product attributes would be represented by the X and Y axes, and the shares from simulations of each combination of attributes would be represented by the third dimension, the height of a surface. Our task is to find the highest point on that surface.

One strategy would be to imagine a grid underlying the surface and to measure its height at every grid point. This is a reasonable approach if there are few dimensions, but quickly becomes unworkable as the number of dimensions increases and the number of grid points multiplies. If the surface has a single peak, another strategy would be to pick a point on the surface, find the direction in which the surface rises most rapidly, and move in that direction to find a higher point, repeating the process until the highest point is found. Such "steepest ascent" hill-climbing methods are efficient with surfaces having single peaks, but can be misled if there are multiple peaks. Because different optimization strategies are more effective with different kinds of response surfaces, the ASM provides several from which to choose.

The ASM can be used to analyze conjoint data from any of Sawtooth Software's conjoint systems (ACA, CBC, or CVA). It can be used for full-profile, partial-profile, and alternative-specific designs. Estimation can include linear terms and interaction terms. It can also be used to analyze conjoint or preference data provided by the researcher that was not necessarily generated by Sawtooth Software's systems. The ASM can also be used with aggregate conjoint data, though it is most effective with individual data.

The paragraphs above have provided a general introduction without details; but several topics require closer examination. These include the types of product simulations available, the specific optimization methods provided, and the ways cost information is used in the calculation of profitability.

Simulation Methods

The Advanced Simulation Module provides several methods for simulating preferences. With the exception of Utility search which requires no additional description, these are all available in the regular Sawtooth Software SMRT simulation software, and are fully described elsewhere. Here we shall provide just a brief description of each, ranging from simplest and fastest to slowest but most useful.

First Choice or Maximum Utility: This is the simplest simulation method. Each respondent's utility for each product is estimated by summing the appropriate part worths. The utilities for all products are compared, and the respondent is assumed to choose the product with maximum utility. Another way to say this is that we assume all of a respondent's choice likelihood accrues to his "first choice" product, regardless of the magnitude of difference in utility between that product and the others. The estimate of a product's share of market is simply the percentage of respondents for whom it has highest utility.

This method has some desirable characteristics. It is simple and easy to understand. Also it is fast, so optimizations done with First Choice simulations proceed quickly. Most important, First Choice simulations are not vulnerable to difficulties caused by the inclusion of similar products, such as when evaluating portfolios of similarly branded alternatives. We'll describe this problem shortly.

However, first choice simulations also have some undesirable properties. They tend to exaggerate the shares of popular products and underestimate the shares of unpopular products. Further, unlike other methods, there is no way to "tune" them to compensate for this characteristic. A second shortcoming is that since all of a respondent's choice likelihood is allocated to a single product, the standard errors of the resulting shares are larger than with other methods that distribute a respondent's choice likelihood across several products.

At the current stage of simulation technology, the First Choice method is mainly of historical interest, and we would not advocate using it in optimizations unless there are special circumstances, such as a compelling need for computational speed, the benefit of exceptionally large sample sizes, and confirmation that the relative scaling of share results is appropriate.

Share of Preference: This method is only slightly more complicated than the First Choice method. As with that method, each respondent's utility is computed for each product. However, rather than assigning all of a respondent's choice likelihood to the product with maximum utility, we allocate choice likelihood among products by first exponentiating all products' utilities (converting them to positive numbers) and then percentaging the results so that they sum to 100. (An example appears below.) This is equivalent to employing a logit model for product choice.

The Share of Preference method is nearly as fast as the First Choice method, and has the additional benefit that the results can be "tuned" so the ratio of maximum to minimum estimated product shares can be adjusted as desired. This is accomplished by multiplying the part worths by a positive constant (the "exponent"). A large constant causes more extreme share predictions, and in the limit the Share of Preference method approaches the First Choice method. A small constant causes shares to be more nearly equal, and with a very small constant the predicted shares will all become nearly equal.

However, as with all logit models, the Share of Preference method is vulnerable to what are known as "IIA problems." (IIA is short for "independence from irrelevant alternatives"). A simple example can demonstrate this. Suppose there are two quite different products (A and B) for which a respondent has utilities of 0 and 1.0. Then Share of Preference estimates of the respondent's choice likelihoods would be as below:

Product	Utility	Exponentiated Utility	Share Estimate
A	0.0	1.00	26.9
B	1.0	2.72	73.1

		3.72	

Now suppose we introduce another product, A', which has characteristics identical to those of product A. Then in a three-way simulation we obtain the following estimates:

Product	Utility	Exponentiated Utility	Share Estimate
A	0.0	1.00	21.2
A'	0.0	1.00	21.2
B	1.0	2.72	57.6

		4.72	

We have nearly doubled the total estimated share of the A products merely by including a second copy. We could drive the estimated total share of the A products as high as we like simply by including many of them. But we know this is not a realistic simulation of real world conditions, where a respondent who prefers traveling via car to a bus is likely to choose the car no matter how many buses are available.

By contrast, the First Choice method assigns the respondent to product B no matter how many copies of A are included, demonstrating much more reasonable behavior.

With logit models, a newly introduced product takes share from existing products in proportion to their current shares. (The ratio of original shares 26.9 / 73.1 = 0.368 is the same as the ratio of new shares 21.2 / 57.6 = 0.368.) This property is useful in some circumstances, but presents problems in our context because a newly introduced product will probably ***not*** take share from others proportionally to their shares. For example, an additional package size for a soft drink may be expected to take more business from other sizes of its own brand than from other brands. We refer to this important property as differential substitution.

This IIA property of logit models is troubling, because we may wish to estimate the total market share for a portfolio of products that are somewhat similar to one another. We may wish to know how many more units of a make of car will be purchased if we offer a convertible in addition to a sedan, or how many more ounces of cereal will be purchased if we offer a jumbo package in addition to a regular sized one. In the case of aggregate models, such as a main effects logit model, the Share of Preference method is obviously inappropriate for questions like these.

IIA problems are substantially reduced when dealing with individual rather than aggregate simulations. Individual simulations often allocate nearly all of a respondent's preference to a single product, so they become more like First Choice simulations. Thus, Share of Preference simulations are likely to be less misleading when used at the individual level.

The Share of Preference method should provide good results when all of the products in a simulation are equally similar to one another. However this is a hard condition to ensure, so it is more prudent to use a method that is not vulnerable to this difficulty.

Share of Preference with Correction for Similarity: This simulation method was implemented by Sawtooth Software many years ago. It corrects for product similarity by decreasing estimated shares of products in proportion to their similarity to others. At the time of its introduction it provided an improvement, compared to unadjusted Share of Preference simulations. However, though in the right direction, the corrections are somewhat arbitrary. Technological advances in the last few years have provided another method which is superior.

Randomized First Choice (RFC): This is the preferred method for simulations involving choices among competing products. It is slower than other methods, but overcomes the previously described shortcomings.

Results for each respondent are simulated many times in "sampling iterations." The part worths are perturbed randomly for each iteration. The perturbations are of two types. First, the part worths themselves are perturbed (by adding "attribute error"), and the modified part worths are used to sum utilities for each product. Then the utility sums themselves may be further perturbed (by adding "product error"). For each iteration the respondent is allocated to the product with highest (perturbed) utility. The results for all iterations are averaged to produce the final estimate of choice share for each respondent.

It is clear that real respondents are somewhat inconsistent when making choices. This suggests that the values they ascribe to product features actually vary from moment to moment, and the RFC procedure is an attempt to mimic that variability.

This method is much slower than the preceding ones, because each respondent's choices are simulated many times rather than just once. However the method has compensating advantages.

The default for the RFC method is to add only "attribute" rather than "product" error. In that case IIA problems are avoided and similar products do not receive inflated shares. This means that RFC simulations are appropriate for a wide range of occasions, including estimation of the value of line extensions and portfolios of similar products.

Also, unlike simple First Choice simulations, Randomized First Choice simulations can be tuned to provide differing ratios of extremity between shares for popular and unpopular products. The magnitudes of perturbations of part worths and utilities can be adjusted. In general, as magnitude of perturbations is increased, predicted shares become more similar. In the limit, with very large perturbations and many sampling iterations, all products' shares would become equal. It is commonly found that First Choice simulation results are too extreme. One reason is that in the real world not every product is always available, and occasionally a buyer must accept a product that would not otherwise be his or her first choice. Like Share of Preference simulations, and unlike First Choice simulations, Randomized First Choice simulations can be tuned (by adjusting the magnitude of perturbations) to more closely mimic actual market shares.

Purchase Likelihood Simulations: Sometimes a product category is so new that there are no competitive products to which it may be compared. At such times it is useful to model purchase likelihood rather than share of market. Some conjoint methods produce part worths that are scaled so that utility sums can easily be converted to estimates of purchase likelihood. In that case a simulation could involve a single product, and if there are multiple products, their results are unaffected by one another (for example, all products could have high likelihoods, or low likelihoods).

When simulating multiple products with Purchase Likelihood simulations the researcher may be less interested in the sum of the purchase likelihoods than in their maximum for a given respondent. The researcher may seek a portfolio that has something for everyone, and may not care how attractive each respondent finds his/her second-and-third-choice products. For this reason the option is provided of averaging only the likelihoods for highest-likelihood products in a portfolio.

Purchase Likelihood simulations can be very fast. But we caution the user not to interpret the results literally. Respondents are notoriously unable to estimate likelihoods of any kind, and likelihoods of purchase of new products are no exception. Purchase Likelihood estimates should never be interpreted as more than directional indicators of buyer preference. Their absolute levels are probably meaningless.

Recommendation: We recommend Randomized First Choice as the preferred method in nearly all circumstances. If a much faster method is required, the First Choice method has the advantage of not inflating shares for similar products, but its estimated market shares will probably be too extreme and their standard errors will be relatively large.

The Share of Preference method is also fast, and can be tuned to provide the desired amount of variation in product shares, but may inflate the shares of products that are similar to others. This difficulty can be partially ameliorated by using Share of Preference with Correction for Similarity, but the correction is less accurate than that provided by Randomized First Choice.

If the product of interest is unique or too new for there to be a competitive product category, then Purchase Likelihood simulations may be appropriate, but only to provide a relative ranking of products rather than to provide accurate estimates of actual purchase likelihoods.

All simulation methods can be run either with respondents weighted equally or with respondents weighted by a variable selected by the user. Furthermore, external effects can be included, and the Exponent (scale factor) may be tuned, when appropriate.

Optimization Methods

To conduct product searches in the ASM, the user specifies several items:

- One or more products for which optimal attribute levels are to be discovered.
- Competitive products that will be held constant throughout the analysis (unless searches are to maximize utility or purchase likelihood for a single offering).
- The objective to be optimized (estimated market share, purchase likelihood, profitability, revenue, cost minimization, or total utility) whether for a single product or across a portfolio of products.
- For each attribute, the range of levels permitted, as well as an "interpolation range" for any attributes that are to be treated as continuous rather than discrete variables. With interpolation ranges, an interpolation step value may be specified. With an interpolation step of unity, only the levels measured are explored. One can explore levels in between those measured, by specify fractional increments such as 1/2, 1/5, etc., where the step value is the denominator. The user may also specify ranges within attributes that will not be explored.
- If profitability searches or cost minimization are used, product cost information is also required.

Five optimization algorithms are available, which differ in several ways. We now describe each of them briefly.

Exhaustive Search is the simplest of the algorithms. It examines every combination of permitted levels of all attributes. For example, if an attribute has three levels (levels 1, 2, 3) and an interpolation step value of 2 is used, there would be 5 levels of that attribute to explore (1, 1.5, 2, 2.5, 3).

The main strength of Exhaustive Search is that it is guaranteed to find the best solution (from among the domain specified). If the response surface has multiple peaks, Exhaustive Search will evaluate all of them and choose the best one.

The main shortcoming of Exhaustive Search is that the number of combinations to be evaluated can become very large. For example, with 10 attributes, each having 5 possible levels, the number of solutions to explore would be 5 to the 10^{th} power, or nearly 10 million.

Thus Exhaustive Search will probably not be used until faster methods have first reduced the size of the problem. For example, other methods might quickly determine that interest should be focused on narrower ranges of several attributes. If a particular level appears always to be present in the winning product, then you can hold that level constant, letting other attributes vary. After holding several attributes constant, Exhaustive Search could be used to find the best combination of levels within a reduced domain of possibilities.

Grid Search is much faster than Exhaustive Search. The attribute levels to be explored are specified in exactly the same way, but rather than examining all potential combinations, Grid Search proceeds heuristically. A starting solution is chosen, or may be specified by the user. Then several iterations are conducted. Within each iteration the attributes are selected in random order, all permitted levels of each attribute are examined (with all other attributes held constant) and the best level is retained. In each iteration the attributes are examined in a different random order. The process continues until an iteration fails to find a better solution. That is to say, Grid Search stops after a solution has been found that cannot be improved by changing any single attribute.

The main strength of Grid Search is its speed. The number of combinations to be evaluated increases only linearly with the number of attributes and levels, rather than as a product of the numbers of levels raised to the power of the number of attributes. If the response surface is single-peaked, Grid Search is guaranteed to find the optimum. If there are multiple peaks, then repeated runs from different starting points are very likely to find it. Even when run several times from the same starting point Grid Search is likely to provide different results, because the order of adjustment of attributes is randomized differently each time.

Its main shortcoming is that Grid Search is not guaranteed to find the global optimum if there are several peaks. However, in our experience Grid Search is very effective at quickly finding good solutions, and thus can be used to reduce the domain required for further exploration by Exhaustive Search.

Gradient Search works by finding a combination of attributes to change simultaneously, using a "steepest ascent" method to find the top of a peak in the response surface. For attributes that cannot be interpolated, Gradient Search uses a procedure identical to Grid Search (but restricted to whole-number values). Gradient Search works like this:

- Start with an initial solution either obtained randomly or as specified by the user.

- Make a small change in each interpolable attribute, one-at-a-time, and measure the resulting gain or loss in the objective. Decide on a direction for changing all interpolable attributes simultaneously which results in the greatest improvement per unit of change. This is the "gradient," and is the direction of locally steepest ascent for the response surface.

- Conduct a "line search" starting with the current solution and moving in the direction specified by the gradient. The first move is very small, and each successive move is twice as far from the starting point. This process must eventually go too far, as revealed by a drop in the objective function.

- Use results from the final three points to fit a quadratic curve to the response surface, and find the point at which the quadratic function is maximized. Evaluate the response surface at that point. Retain that solution if better than the best previous one.

- When some attributes are interpolable and others are not, then each iteration consists of a line search for the interpolable attributes and Grid Search for those not interpolable.

- Iterations are terminated when there is no improvement from one iteration to the next.

The main strength of Gradient Search is that it can find good solutions quickly, and is not restricted to pre-specified interpolation steps. Gradient Search treats interpolable attributes as continuous rather than discrete variables.

The main shortcoming of Gradient Search is that, like Grid Search, it is only guaranteed to find local optima. Because there may be multiple peaks in the response surface, it would be desirable to run Gradient Search several times from different starting points.

A second difficulty with Gradient Search is that it may have trouble with discontinuities in the response surface. For example, price could be specified as having level values of [$1000-$3000, $6000], indicating continuity between $1000 and $3000, but excluding values between $3000 and $6000. In this case interpolation is permitted among some of its levels, but not between $3000 and $6000. Gradient Search is likely to be unsatisfactory with such attributes.

Stochastic Search uses a very simple concept, similar in many ways to Grid Search. The basic idea is that a variable is chosen at random, and its value is changed at random. If that results in an improvement, the change is accepted, and otherwise rejected. This is repeated many times. Surprisingly, this simple procedure is very effective. Stochastic Search only uses this procedure for interpolable variables. Those which cannot be interpolated are handled with the procedure used in Grid Search.

Stochastic Search also makes use of a procedure known as "snapping." We have found that the optimal solution often occurs at whole-numbered attribute levels. To accelerate convergence, if any attribute level is close to a whole number we "snap" it by setting it exactly equal to that whole number. In the first iteration we use a broad snapping interval, converting any value within 0.2 of a whole number to that whole number. The interval is decreased in each subsequent iteration, and has little impact in later iterations.

A starting solution is chosen just as in Gradient Search. Several iterations are done, and within each iteration we try adjusting all interpolable variables in random order and repeat that process many times. The amount by which each interpolable attribute is adjusted is a random normal variable with standard deviation reduced in each successive iteration. We try adjusting each non-interpolable variable using a procedure like Grid Search. Iterations stop when there is no improvement from one iteration to the next.

The main strengths of Stochastic Search are its speed, in which it is comparable to Grid Search and Gradient Search, and the fact that it does not require continuity, and thus is not hampered by constraints.

Its main shortcoming is that it does not guarantee finding the global maximum. Like all other methods except Exhaustive Search, it should be run several times from different starting points to ensure that the best answer is found. However, we have found it to be very effective at finding the best solution.

Genetic Search is our implementation of the procedure described by Balakrishnan and Jacob (1996). We start with a population of size 300, consisting of potential solutions obtained randomly. In each iteration ("generation") the least "fit" half of the population is replaced with new members obtained by random "mating" of the most fit 150 members. We assume that the most fit "parents" are most likely to produce fit "children." Parents for each new member are chosen randomly, with probability related to their fitness.

Each child has a combination of the parents' characteristics. For interpolable variables the child's value is a random one, rectangularly distributed between the parents' values. Also, the child's value is subjected to "mutation" by adding a normal random variable. As with some other algorithms, we "snap" values to whole numbers, aggressively at first but less so in each subsequent generation. For non-interpolable variables, the child receives the value of one parent or the other with equal probability.

As a measure of success we use the objective value for the best member of that generation. It is difficult to say when one should stop iterating. Lack of improvement in one generation does not necessarily mean that there will be no improvement in future generations. The default is to stop when three successive generations have failed to show improvement, although the user may modify that setting.

One strength of Genetic Search is that, like Stochastic Search and Grid Search, it makes no assumptions about the shape of the response surface, and is therefore not hindered by discontinuities due to constraints. Another is that it is theoretically possible that the final population may include near-optimal members who are high on different peaks, and in these cases Genetic Search is less vulnerable to local optima. In our initial work, we have seen this only occasionally, and for these unusual cases Genetic Search consistently returns good solutions. For the most part we find that the response surface is more generally unimodal, and the faster "steepest assent" methods also consistently achieve near-optimal solutions.

Genetic Search's main shortcoming is slowness. We have found it to take several times as long as the faster methods.

Recommendations: As a robust general-purpose approach, it might make most sense to start by running either Grid Search or Stochastic Search from different starting points. (Each time you request a search procedure in the ASM, it uses different random starting points or random orders for attribute changes.) If the same answer is always obtained, that is probably the optimum. If not, then the experience obtained should permit reducing the size of unknown domain substantially, so it may become feasible to run Exhaustive Search in that reduced domain.

If there are quantitative variables that are continuous rather than categorical, then Gradient Search should be most appropriate for them. Remember that if you haven't specified any interpolable ranges, Grid, Gradient and Stochastic search are identical.

Genetic Search is an interesting algorithm that has the potential of finding good solutions when conditions limit the capabilities of other methods, such as when the response surface is very irregular with multiple peaks. It takes much longer, but in certain cases achieves superior results.

For both Genetic and Exhaustive Search the user can specify the number of solutions that should be reported, and that many of the best solutions are displayed in the report window. This capability seems most likely to be useful when the response surface has multiple peaks. In that case the researcher may be able to use expert opinion to evaluate a number of near-optimal solutions.

We have seen situations where searches seem to capitalize on "reversals" in part worths to produce less useful solutions. It may, therefore, make sense to use part worths that have been constrained during estimation to avoid reversals.

Introducing Product Costs and Estimating Profitability

Profit is the universal measure of success for most businesses, and as such is usually the most valuable search criterion. Unless constrained to search within relatively profitable spaces of opportunity, Utility, Share of Preference, or Purchase Likelihood simulations mostly produce trivial solutions where the best features are delivered at the cheapest prices. Moreover, revenue searches focus solely on revenue without regard to profit. We recognize that a firm may have a specific goal in mind for a particular product line, such as maximizing penetration. But these strategies are generally the exception rather than the rule.

If profitability is to be optimized, the user must provide information about the unit cost for features. Such information is often hard to obtain, but we urge users to avail themselves of cost data, or to approximate costs wherever possible.

Cost information may be specified for attributes independently, or may vary depending on other variables. For example, if we were studying a pharmaceutical product, one attribute might be type of container, and another might be size of container. The costs of different types of container can depend on their sizes, and on several other variables as well, if desired.

The arithmetic of the profitability computation is very simple. For each product we have not only part worths reflecting respondent values, but we also have part-costs indicating the contribution to cost of different product attributes. We sum those part-costs to get the cost of one unit of product. We subtract that cost from the price of the product to get a unit margin, and we multiply unit margin by estimated market share to get a measure of relative profitability. The user may specify the total number of units sold in the market, in which case estimated revenue and profitability can be stated in actual monetary amounts.

Caveats

One problem in the early years of conjoint analysis was that simulation results looked so much like actual market shares that managers sometimes forgot they were only estimates. The same difficulty applies to product optimizations.

Users must remember that optimization results are only the results of statistical exercises. There are numerous ways things can go wrong. To name just a few, (1) the sample of survey respondents must be chosen appropriately and be of sufficient size, (2) the conjoint questionnaire must be designed well (include the right attributes and levels), (3) the competitive context must be specified accurately, and (4) the most robust methods should be used to estimate part worths.

Results concerning revenue and profitability are especially troublesome. They depend critically on proper estimation of price coefficients. We know from years of experience that many conjoint methods do not do a good job of estimating price sensitivity. Of the methods provided by Sawtooth Software, CBC is the most effective and ACA least effective for pricing studies. However, as effective as CBC can be in the hands of a knowledgeable researcher, it is easy to describe attribute levels in vague and unclear ways that lead to misleading results.

Finally, profitability computations require not only buyer preference information, but also cost information. That information is hard to come by, and may simply be unavailable in many organizations. Needless to say, the accuracy of profitability estimation depends critically on the accuracy of cost data.

As a final note, it is important to recognize that even if the caveats above are dealt with appropriately, the results of an optimization are only valid inasmuch as the assumed fixed competition does not react. When competitors react, the previous solution is usually no longer optimal.

How Well Does It Work?

We have tested the optimization routines with several conjoint data sets volunteered by Sawtooth Software users. We have been able to assume cost information, so our tests have involved all types of optimization criteria supported by the ASM.

The results with all data sets and search criteria have been similar, so we shall describe only one set of results in detail. This data set consists of conjoint part worths for 546 respondents on 15 attributes having a total of 83 levels. We shall disguise the product category.

With this data set we constructed six hypothetical competitive products and sought characteristics for an optimal seventh product. Realizing that the highest-share product would necessarily have the lowest price, we constrained the price of the new product to a fixed level suggested as reasonable by the provider of the data set. One other attribute was also constrained, which dealt with the “form factor” of the product, since the owner of the data set already had a clear idea of what form factor he wished to explore.

We have subjected this data set to hundreds of optimizer runs, both to explore the comparative effectiveness of the several optimization procedures, and to “tune” default settings for several user-specifiable details.

We assess the effectiveness of each simulation method by two measures: (1) how reliably it obtains what we believe to be the best answer, and (2) how much computational effort is required. Since we have done our testing on computers with varying clock speeds, we standardize our time measurements by reporting the number of actual product simulations required.

Even after constraining two of the attributes to constant levels, there remain approximately 143 million possible product configurations. Even with the fastest of the simulation methods and on the fastest desktop computer available today the Exhaustive Search method would require nearly five years of continuous computation, clearly not a feasible alternative.

By contrast, to see how quickly a solution could be obtained, we ran Grid Search with Share of Preference simulations, limiting the domain explored to whole-number levels (using interpolation steps of 1). That computation required only 97 simulations, completed in 30 seconds using a computer with a 2.8 GHz processor. The solution differed in only one attribute from the best solution found in hundreds of optimization runs using RFC simulations.

All further results we shall report use RFC simulations. The default number of independent sampling iterations per respondent is about 90 (to achieve about 50,000 total iterations across all respondents) with this data set, but since we planned so many runs we used a more modest value of 30. (This leads to a total of 16,380 sampling iterations across all respondents, 546 x 30 = 16,380) We report results for seven runs of each optimization method, all from different random starting points.

We judged five attributes to be interpolable; that is to say, we explored the entire region between their maximum and minimum levels, not limiting our exploration to whole-numbered levels. Eight other attributes were judged not to be interpolable, and we explored only whole-numbered levels. Two attributes, price and form factor, were locked at fixed levels.

The Grid, Gradient, and Stochastic search algorithms got the same answer each time, which is also the best answer we have found by any method. Interestingly, it consisted of only whole-numbered levels, even for attributes treated as continuous. The median number of evaluations required by those methods were 129 for Gradient, 187 for Grid, and 253 for Stochastic. Using a computer with 2.8 GHz processor, these runs required an average of about 5 minutes, a little less for Gradient and a little more for Stochastic.

Genetic Search never got exactly that same answer, although it came close several times. The worst of its seven results had a product share 70% as high as the apparent optimum, and two of the seven runs had product shares 98% as high as the apparent optimum. Letting it run longer rather than halting after three generations without improvement might have yielded better answers, but at the cost of longer run times. As it was, the median number of simulations required by Genetic Search was 1600, implying run times about ten times as long as the average of the other methods.

As a final check on the optimality of the apparently best solution, we ran Exhaustive Search on a restricted set of attribute levels, including all levels that had appeared in any of the solutions described so far, plus all whole-numbered levels between those. That run required 3600 simulations and about an hour and a half on the same computer. The best solution it found was identical to that produced by the other methods.

Thus we can be quite confident that the best solution found was indeed optimal, although absolute certainty would require an impossibly long Exhaustive Search. The encouraging result is that this solution was also found seven times out of seven by Grid, Gradient, and Stochastic Search, and the average length of those runs was only about 5 minutes. In practice, you will probably want to use at least 50,000 sampling iterations for RFC simulations during your final searches, so your run times would be three times longer or more.

A problem with 15 attributes and 83 levels may be smaller than some that users of the software will encounter, but the computational time for those three algorithms tends to increase only linearly with the number of total attribute levels. Therefore problems with much larger numbers of attributes and levels (or including optimization of multiple products simultaneously) should still be solvable in reasonable time.

We should note that the findings from this one data set will not apply generally to all data sets. We have seen other data sets for which genetic search consistently provides near-optimal answers, and the “steepest ascent” techniques sometimes return noticeably sub-standard answers. Thus, we recommend you alternate between different methods, first starting with quicker methods/settings and then working toward longer run times as you gain familiarity with the data set and/or reduce the search space.

References

Balakrishnan, P. V. and Varghese S. Jacob (1996), “Genetic Algorithms for Product Design.,” *Management Science*, 42 (8),1105-1117.

Green, Paul E. and V. R. Rao (1971), “Conjoint Measurement for Quantifying Judgmental Data,” *Journal of Marketing Research*, 8 (August), 355-363.

Green, Paul E. and Abba M. Krieger (1993), "Conjoint Analysis with Product Positioning Applications", Handbooks in Operations Research and Management Sciences", Volume 5, J. Eliashberg and G. L. Lilien (eds.). Amsterdam: North-Holland, Chapter 10, pp. 467-515.

Chapter 4

Specifying Product Searches in SMRT

Introduction

We assume that you already are comfortable with the basics of using the market simulator in SMRT. If you are not, we strongly recommend that you review the tutorial chapter (chapter 4) for the ACA, CBC or CVA software for SMRT. Also, please refer to other chapters in our conjoint analysis manuals dealing specifically with the fundamental functionality of the market simulator.

The Advanced Simulation Module (ASM) extends the functionality of the market simulator to search for optimal products based on the criteria of share, purchase likelihood, revenue, profit, cost minimization or utility. Searches are conducted using one of five different search algorithms. (If searching based on cost or profit, you must supply additional cost information, by attribute.) No "programming" is required to perform these searches. You can set up fairly complex searches using the point-and-click interface.

Assign Level Values

The first step in conducting a product optimization search is to make sure you have assigned any level values to quantitative (continuous) attributes, such as price, speed, weight, or other variables to which you wish to assign convenient level value codes. In the case of price, you'll almost certainly want to specify level values if you plan to search based on revenue or profit, otherwise the monetary scale will not be correct.

For example, you may wish to associate level values with the levels of price for your study:

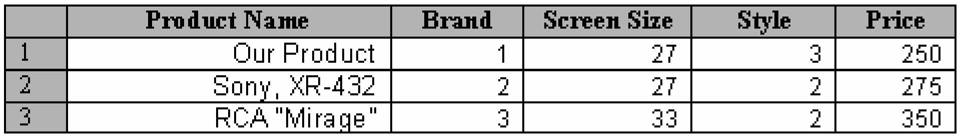
Level Value	Level Number	Level Text
100	1	$100 per year
200	2	$200 per year
400	3	$400 per year

From the ***Market Simulator*** dialog, click ***Assign Level Values…*** to associate the value "100" with level 1, "200" with level 2, etc. That way, any revenue/profit calculations have appropriate scaling. Also, assigning level values makes it easier to interpolate between values and interpret the output. For example, rather than specifying $300 as price "2.5" (half-way between levels 2 and 3), if you've specified level values as above, you simply specify the price for the product directly as "300."

Specifying Dynamic (Searched) Products

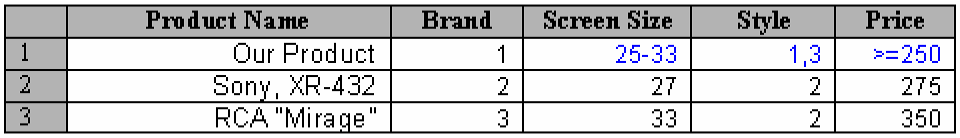
From the ***Scenario Specification*** dialog, in the *Operating Mode* control, select *Product Search*.

When using the market simulator in typical simulation mode, you specify multiple products using a spreadsheet-like grid within the ***Scenario Specification*** dialog, accessed by clicking ***Add…*** from the ***Market Simulator*** dialog. For example, you might specify three products as following:



	Product Name	Brand	Screen Size	Style	Price
1	Our Product	1	27	3	250
2	Sony, XR-432	2	27	2	275
3	RCA "Mirage"	3	33	2	350

However, when searching for optimal products, you enter *ranges* of levels rather than fixed levels for the product to be searched. For example, in the grid below we specify that “Our Product” must be brand level 1 (our brand), but all the other values can vary.



	Product Name	Brand	Screen Size	Style	Price
1	Our Product	1	25-33	1,3	>=250
2	Sony, XR-432	2	27	2	275
3	RCA "Mirage"	3	33	2	350

Notice that we’ve used three kinds of syntax for specifying the levels that can vary:

25-33
“The levels of Screen Size can vary from 25 to 33”

1,3
“The Style levels can be either 1 or 3”

>=250
“Price must be greater than or equal to 250”

When you use expressions such as these for searched products, the text turns blue. You cannot extrapolate when conducting product searches. If you type an extrapolated value, it turns red, and before closing this window you’ll receive an error message asking you to correct the problem. Note also that for continuous variables to be interpolable in product searches, you must assign interpolable ranges under ***Mode Settings…*** and using the *Attribute Interpolations* tab.

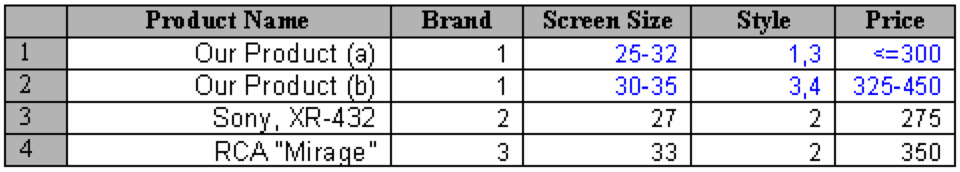
Only these three types of expressions may be used: “-“ (dash), “,” (comma), “>=” (greater than or equal to), or “<=” (less than or equal to). Here are more examples of valid product search entries:

15-20,25
“The levels can vary from 15 to 20, or can also be 25”

1,3,5,6,7
“The levels can be either 1, 3, 5, 6, or 7”

<=45,>=60
“The levels can be less than or equal to 45, or greater than or equal to 60”

You can search for multiple products simultaneously, such as finding the best 2 products to offer. In the example below, the product search will seek to maximize the *sum* of the shares, revenues, profits, or utilities (whichever we specify under ***Mode Settings…***) of products 1 and 2.



	Product Name	Brand	Screen Size	Style	Price
1	Our Product (a)	1	25-32	1,3	<=300
2	Our Product (b)	1	30-35	3,4	325-450
3	Sony, XR-432	2	27	2	275
4	RCA "Mirage"	3	33	2	350

Each of these two searched products is constrained to be brand level 1 (our brand). “Our Product (a)” is constrained generally to have smaller screen sizes and lower prices, whereas “Our Product (b)” is constrained to have larger screen size and higher prices. Both products can be of style 3, but styles 1 and 4 are unique to each searched product. However, it is not necessary to constrain products to have mostly unique searchable ranges. We could easily have let the two products vary freely over the entire range of all possible levels.

When conducting product searches, you can also specify that certain static (fixed) products should be included in the objective function (the value to be maximized, whether share, revenue, profit or utility). For example, you may have a current product on the market that will not change, but you want to search for another product to offer, such that the profit for the existing product and the new product are maximized. Use the ***Mode Settings…*** button to specify any static products to be included in the search criterion.

Note that either at least one fixed product must be left out of the objective function or the "None" share must be enabled by setting the None weight to a value greater than 0 under the ***Advanced Settings…*** button.

Prohibiting Combinations of Levels

In the previous example, we showed how to constrain searched products to have certain characteristics (and to prohibit other characteristics). For example, we didn't permit any searched products to have our brand and also style 2. Style 2 was reserved for our competitors, and our brand could only appear with styles 1, 3, or 4. You will probably find that most prohibitions you wish to enforce can be done in the same manner as we've shown, using only the product entry grid.

If you need to specify additional prohibited combinations, you can use the *Prohibitions* tab, accessed from the ***Scenario Specification*** dialog by clicking ***Mode Settings…***. Also, keep in mind that any prohibitions you specified when developing your experimental design or composing your conjoint interview are *not* carried forward into analysis. The market simulator pays no attention to the earlier prohibitions you may have used in composing a conjoint questionnaire.

Entering Cost Information

To perform profitability searches, cost minimization searches (exhaustive search only) or other searches (e.g. share, utility) subject to the criterion that cost cannot exceed some threshold, you are required to specify cost information. In the ASM, costs are associated with specific levels in your study design. It is not necessary to specify costs for all attributes in your study, or for all levels within a particular attribute. Some attributes may not have associated costs, and some levels may be irrelevant to your particular search (i.e. your competitor's costs).

Costs can be tied to the levels of a single attribute, or on the joint combination of up to four attributes. To specify costs, from the market simulator's ***Scenario Specification*** dialog, select *Product Search* as the *Operating Mode*. Then, click the ***Mode Settings…*** button right below *Operating Mode*. The ***Product Search Settings Dialog*** is displayed: Click the ***Cost, Revenue & Profit Information…*** tab. In the *Attribute-Based Costs* area, click ***Add Table…*** to add incremental costs associated with different attributes.

Chapter 5

Practical Optimization Examples

Introduction

In this chapter, we'll illustrate some common optimization problems that can be solved using the ASM. In the first example, we specify step-by-step how to use the software to perform the searches. Following the first example, we'll describe additional types of problems more generally, to give you a greater understanding of the flexibility within the program. We'll refer to example data sets installed with the ASM.

The examples covered include:

- New Product Introduction without Competition (Utility, Purchase Likelihood Search)
- New Product Introduction with Competition (Share, Revenue Search)
- Searching for Multiple Products Simultaneously
- New Product Introduction with Competition (Profitability Search)
- TURF-Like Problems
- Maximize Appeal Subject to a Maximum Cost
- Minimize Cost Subject to Meeting a Performance Threshold

Example #1: New Product Introduction without Competition (Utility, Purchase Likelihood Search)

We'll begin with a simple illustration that doesn't involve a "market simulation" at all, but focuses on finding a single product configuration to maximize average utility for the population. This, admittedly, is a problem that doesn't require an optimization routine. But for the purpose of illustration, we'll begin with the simple case.

Imagine you are creating a new product to introduce to a market without any established competition. We'll refer to the "Tires1" data set that ships with SMRT and is used in the ACA documentation. The Tires1 data set deals with tires for mountain bikes. You can find this study in your **Tutorial** folder. We invite you to start the SMRT software, and open this project, using ***File | Open...*** and by browsing to your **\Program Files\Sawtooth Software\SMRT\Tutorial** directory.

After opening the Tires1 project, click ***Analysis | Market Simulator***, and the main ***Market Simulator*** dialog opens. You should have a single existing utility run, with no simulation scenarios yet specified.

The average raw part worths for the 30 respondents in this data set are:

Brand Name:

Michelin	-0.12
Tioga	-0.00
Electra	-0.31

Tire Type/Firmness:

Puncture-proof solid foam core. Very firm (best for pavement)	-0.25
Puncture-proof solid foam core. Somewhat firm (for pavement or dirt)	-0.11

Puncture-proof solid foam core. Less firm (for best traction in dirt)	-0.24
Standard inner tube tire. Adjustable firmness (for pavement or dirt)	0.03

Tread Type:

Off-road tread	-0.24
On-road tread	-0.16
On-road / Off-road tread	-0.03

Weight:

Tire weight: 2 lbs.	0.31
Tire weight: 3 lbs.	-0.08
Tire weight: 4 lbs.	-0.66

Tread Wear Durability:

Tread lasts for 1000 miles	-0.42
Tread lasts for 2000 miles	-0.16
Tread lasts for 3000 miles	0.15

*The choice of part worth scaling (raw, points, diffs, zero-centered diffs) for reporting the average part worths can be selected under the **Advanced Settings…** button from the **Scenario Specification** dialog. This is a new feature with the ASM.*

With this array of attributes and levels, there are 3x4x3x3x3= 324 possible product combinations. Let's assume your brand is fixed: you are Michelin. With brand fixed, there are now 4x3x3x3=108 remaining combinations.

If the goal is to provide the single, most appealing product to the market (assuming no competition) without regard to cost, the optimal product is trivial. After brand (Michelin), one selects the most preferred level on average from each remaining attribute:

Level	Utility
Michelin	-0.12
Standard inner tube tire. Adjustable firmness (for pavement or dirt)	0.03
On-road / Off-road tread	-0.03
Tire weight 2 lbs.	0.31
Tread lasts for 3000 miles	0.15

Total:	0.34

Even though the answer is trivial, we can use the ASM to search for this result. From the ***Market Simulator*** dialog, click ***Add…*** to add a new *Simulation Scenario*. Specify a simulation scenario name in the upper-left field. Under *Operating Mode*, select *Product Search* using the drop-down control. Click ***Insert Product…*** and specify a product name in the first column, labeled *Product Name*. Type the following specifications:

Product Name	Brand Name	Tire Type	Tread Type	Weight	Tread Wear
Optimal	1	1-4	1-3	1-3	1-3

Note that we have specified both fixed levels (for brand) and also dynamic ranges (e.g. 1-3) for attributes. Dynamic ranges indicate the range of values that the search routine can use to maximize some criterion, yet to be specified. In the standard market simulator in SMRT, only fixed levels can be specified; there is no search capability.

We have yet to specify the simulation method to employ. Under *Simulation Method*, use the drop-down box to select *Utility*. Because this is such a simple problem with just 108 possible combinations (and we are using the extremely fast method of examining utilities only), let's select exhaustive search. In cases where there are few possible combinations, exhaustive search is feasible, and it *guarantees* finding the global optimum. Again, from the *Search Settings* tab, select *Exhaustive Search* as the *Search Method*.

When you choose *Exhaustive Search*, a ***Settings…*** button becomes available. Click this button to review the settings for exhaustive search. The dialog shows that the 12 top products will be reported (sorted by our search criterion). Exhaustive Search and Genetic Search are the two search methods that search a wide enough variety of near-optimal solutions, that reporting the top n products found can be useful for further review.

Click ***OK*** a few times to return to the main simulator dialog, and then click ***Compute!*** to perform the search.

Below the standard report of average utilities and importances, the following report is given (we show only the top three results, of twelve):

Product Search Results Summary

Number of simulations	108
Total time elapsed	0:00:02
Seconds per simulation	0.02

Product Search Result #1

Product	Utility	Std Err	Brand Name	Tire Type/Firmness	Tread Type	Weight	Tread Wear Durability
Optimum	0.34	0.24	1	4	3	1	3

Product Search Result #2

Product	Utility	Std Err	Brand Name	Tire Type/Firmness	Tread Type	Weight	Tread Wear Durability
Optimum	0.22	0.24	1	4	2	1	3

Product Search Result #3

Product	Utility	Std Err	Brand Name	Tire Type/Firmness	Tread Type	Weight	Tread Wear Durability
Optimum	0.19	0.23	1	2	3	1	3

The exhaustive search tried all 108 combinations, and the total elapsed time was 2 seconds. All the searches in this chapter were performed using a 1.8 GHz processor. The optimal product is the one we previously identified by selecting (other than Michelin) the most preferred level from each attribute. As we calculated from the report of average utilities, the average utility for this optimal product is 0.34.

The next-best product alternatives are shown, in sort order by total utility, below the top result. The next-best product has a utility of 0.22. The only difference for this product is that the tread type is a level 2 instead of level 1. The ASM reports the standard error for each objective function, whether utility (as in this case), share, purchase likelihood, revenue or profit.

(Hint: You can easily cut-and-paste the results from the report window into a spreadsheet application such as Excel™. Using the Data, Sort features in Excel and sorting by the objective function (share, etc.), you can filter (collect) the rows associated just with the product specifications, and also perform secondary sorting. Also note that you can cut and paste information (such as an array of attribute levels for an optimal product) from the report window into other parts of the Market Simulator dialogs, such as to another place within the current report, or to the scenario specifications and cost checker areas.)

When searching based on raw part worths, each respondent may not receive equal weight (if the scaling of the raw part worths is more extreme for one respondent than another). If you want to use normalized part worths, such as zero-centered diffs, to conduct the search, you can export the part worths through the *Run Manager* to a text-only .hbu file as zero-centered diffs and then import that .hbu file through the *Run Manager* as a new *Utility Run*. The search results may differ slightly, depending on how the part worths are scaled.

Purchase Likelihood Optimizations

Searching for products that maximize Purchase Likelihood is essentially the same as a utility maximization search, except that the utility for product alternatives is transformed to a purchase likelihood scale. To use the Purchase Likelihood scale, select *Purchase Likelihood* as the search method, from the ***Scenario Specification*** dialog. Purchase Likelihood simulations are proper when the part worths have been calibrated using purchase likelihood questions from ACA or CVA. When part worths are calibrated to purchase likelihoods, the equation below yields least-squares fit to respondents' purchase likelihood scale in the survey instrument.

$$P_x = e^x/(1+e^x)$$

The data for Tires1 came from ACA, and are scaled to produce purchase likelihood values to fit the likelihoods respondents stated during the ACA survey.

If using the Purchase Likelihood simulation with individual-level CBC data that have zero-centered logit scaling (such as with latent class or HB), the interpretation of the purchase likelihood scale is the probability of selecting this alternative relative to an alternative with mid-scale preference (utility of 0) based on the levels defined in the study.

Example #2: New Product Introduction with Competition (Share Search)

This example builds on principles covered in the previous section. For the sake of brevity, we spend less time specifying each click/step than in the previous example.

The previous section dealt with a simple—if not trivial—problem. In this example, we'll introduce the element of competition. When multiple product alternatives are considered, a share simulation is appropriate. For most share simulation cases, we'd generally recommend using the Randomized First Choice simulation method. But, simulations using Randomized First Choice take considerably longer than the other techniques. For the purposes of illustration, so you don't lose unnecessary time if following along in the software, we'll employ First Choice simulations.

For this illustration, we'll use the "TV" data set, also available in your Tutorial directory. This is an actual data set, with 352 respondents, collected using a computerized CBC interview in the late 1990s. The subject matter is features of mid-size television sets for the home. The attributes, levels, and average part worths (zero-centered diffs scaling) are:

Brand:	
JVC	-28.56
RCA	-1.47
Sony	30.03
Screen Size:	
25" screen	-28.37
26" screen	0.11
27" screen	28.26
Sound Quality:	
Mono sound	-71.47
Stereo sound	23.81
Surround sound	47.66
Channel Blockout:	
No channel blockout	-24.10
Channel blockout	24.10
Picture-in-picture:	
No picture in picture	-34.15
Picture in picture	34.15
Price:	
$300	53.80
$350	31.60
$400	-19.73
$450	-65.67

Let’s assume an existing market, with the following four product offerings, and existing relative market shares (shares of First Choice):

Product #1 8.2%	**Product #2** 11.9%	**Product #3** 21.3%	**Product #4** 58.5%
JVC 25” screen Mono sound No blockout No PIP $300	JVC 26” screen Stereo sound Blockout No PIP $375	Sony 26” screen Stereo sound No blockout PIP $400	Sony 27” screen Surround sound Blockout PIP $450

We see that JVC offers a stripped-down, least expensive, model (Product #1), Sony is currently offering a full-featured expensive version (Product #4) that captures the highest share, and both JVC and Sony are offering mid-range offerings (Products #2 and #3).

Let’s imagine that you represent RCA, and RCA wishes to enter this market. RCA is urging you to price the new offering at $390. You must answer two questions:

> Given the existing competition, and a price of $390, what is the optimal product to maximize share?
>
> Assuming a total market size of 1MM units sold, what is the optimal product to maximize revenue (Relative Share x Price)?

Open the TV data set, located in your **Tutorial** folder. First, from the ***Market Simulator*** dialog, click the ***Assign Level Values…*** button, and associate the values 300, 350, 400 and 450 with the four levels of price. When we perform the Revenue search, it is important that we specify appropriate monetary units associated with price (as revenue is computed by multiplying by the level values associated with price). Assigning level values will also make it easier to perform interpolations for price (such as specifying $390).

Next, specify the four existing (fixed) products within the ***Scenario Specification*** dialog. Specify the fifth product to be searched. The specifications include the constraints that it must be RCA and priced at $390. The other attributes should be specified as searched ranges, across all possible levels. For example, screen size should be levels “1-3.” (If you are confused regarding how to specify “searched” products, please refer to the previous example using the Tires1 data set.)

First, we’ll deal with the question, “Which product can RCA offer for $390 to maximize share?” This is a tiny problem, with only 36 remaining combinations to be searched (after holding brand and price constant). Therefore, *Exhaustive Search* is appropriate. Again, recall that we are using First Choice as the simulation method. Make sure to highlight the *HB Run* in the *Utility Runs* list when performing this simulation.

When you click ***Compute!*** from the ***Market Simulator*** dialog, ASM tries all 36 possible combinations, and returns the following optimal product configuration:

> RCA
> 27” screen
> Surround sound
> Blockout
> PIP
> $390
>
> Share= 52.27%

We again see that the search has returned a very trivial solution. If RCA charges $390, it should provide all the best remaining features to maximize share. It is important to note that there are many studies in which attributes do not have such clearly "most preferred" levels. In such cases, the answer will no longer be so obvious.

Optimizing Revenue

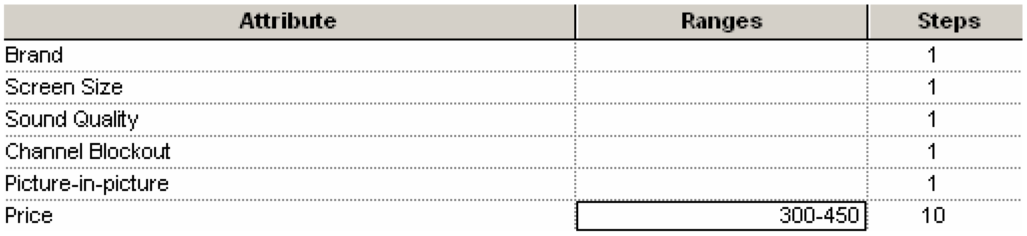
The second question posed earlier was how to price an optimal RCA-branded TV to maximize overall revenue. Using the standard simulation mode with the market simulator, we can compute shares of First Choice for the "optimal" product found in the previous section at various prices. And, assuming a total number of units sold of 1MM to the market, we can calculate total revenue to RCA at each price point (revenue = price x share x units sold).

Price	Share	Revenue
$300	69.32%	$207,960,000
$325	67.05%	$217,912,500
$350	64.20%	$224,700,000
$375	57.67%	$216,262,500
$400	46.31%	$185,240,000
$425	36.65%	$155,762,500
$450	24.43%	$109,935,000

If evaluating every $25 increment, the highest revenue is achieved at $350. However, we can also use the ASM to search for this automatically. Furthermore, we can have the ASM investigate many more price points along the price continuum of $300 to $450.

The ASM's search methods deal with interpolation differently. For Stochastic, Genetic, and Gradient search methods, it treats the range of interpolation as truly continuous. For Grid and Exhaustive search methods, it searches a discrete number of equally spaced points along the continuum. The number of points used is controlled using the interpolation step value. The largest step value permitted is 10, meaning that 10 steps are simulated from one level to the next. Since our price levels represent $50 differences in this study, a step value of 10 indicates that the ASM will use a $5 increment.

This revenue search with interpolation would take some time for exhaustive search. For a first attempt, let's use the ***Grid*** search method. The Grid search doesn't treat interpolable attributes as truly continuous, but relies on the step values to indicate how precisely to search across the interpolated range. First, from the ***Scenario Specification*** dialog, change the Price specification for the RCA product from a fixed price of 390, to a searched range of 300-450. Then, click the ***Mode Settings…*** button. From the ***Product Search Settings*** dialog, choose *Grid* as the *Search method*. Next, we need to specify interpolation rules. Click the *Attribute Interpolations* tab. For price, specify the *Ranges* as 300-450, and *steps* as 10. The grid within the dialog should look like this when you are finished:



Attribute	Ranges	Steps
Brand		1
Screen Size		1
Sound Quality		1
Channel Blockout		1
Picture-in-picture		1
Price	300-450	10

From the *Search Settings* tab of the ***Product Search Settings*** dialog, select to search based on *Revenue*. Select the ***Cost, Revenue & Profit Information...*** tab. Specify the *market size* as 1000000 total units, and indicate which is the *Price* attribute (ASM must know which attribute is associated with price). (If using alternative-specific designs, it is possible to have multiple price attributes.) Click ***OK*** to close the dialog.

You should have already associated appropriate level values with the price attribute (using the ***Assign Level Values…*** button) to ensure that the resulting revenue calculation results in the correct monetary output.

Now we are ready to run the product search. Before doing so, it is important to note that with the exception of the exhaustive search method, the results depend upon the random starting seed that the ASM uses. Each time you run a market simulation, a new random seed is selected. Therefore, we urge you to repeat a search multiple times before accepting a final solution.

(Hint: if you would like to have the ASM automatically repeat the same search multiple times, simply copy the simulation scenario within the Simulation Scenarios list, and check all replicated scenarios you wish to run prior to clicking ***Compute!****. You can copy a simulation scenario by clicking* ***Add…****, and selecting that the new simulation is to be a copy of a previous one.)*

Click ***Compute!*** to search for the product that maximizes revenue using Grid search. You find that the optimal price is $355 for the RCA product to maximize revenue of $224,900,568. This is slightly higher than the revenue achieved earlier at $350, when we were limited to less precise $25 increments. Also note that the same product configuration was found as before. In this case, the same product configuration (other than price) that maximized share also maximized revenue.

The Stochastic, Gradient and Genetic search routines treat price as truly linear. You may wish to attempt a few runs using these search methods instead of Grid. Be warned, however, that Genetic may take a few minutes to compute. Especially when treating interpolated attributes as truly linear, you will nearly always get a different answer each time you repeat the simulation. Moreover, since we are using First Choice simulations, the response surface is not as smooth as if using Share of Preference or Randomized First Choice. This may lead to slightly less stability in this solution.

We have run this problem a few times, using different search methods. A near-optimal solution for maximizing revenue for RCA is found at a price of $353.5813 (revenue= $226,010,771). By examining the accompanying estimated standard error for revenue ($9,050,413), one can determine that there isn't a statistically significant difference between this result and the total revenue of $224,900,568 at a price of $355 that we found using the Grid search and interpolation at $5 intervals, or the earlier result when interpolating at $25 increments.

Example #3: Searching for Multiple Products Simultaneously

For illustration, let's make the search space a bit more complex. Let's assume that RCA is interested in offering *two* different television sets. What are the best two offerings to bring to the market to optimize revenue for RCA, again under the previous assumptions of competition and market size?

From the ***Scenario Specification*** grid, add another RCA product, with dynamic ranges for all other remaining attributes. If we employ an interpolation step value of 10, there are 31 unique prices to investigate over the $300-$450 range. With two products to search simultaneously, the total possible combinations is equal to $(3x3x2x2x31)^2$ = 1,245,456. On our 1.8 GHz PC, this would take about 69 hours to search using Exhaustive Search.

Select the search technique you wish to use. We've repeated this simulation many times. Results that are probably near the global optimum are:

Product	Revenue	Std Err	Product Shares of Preference	Std Err	Brand	Screen Size	Sound Quality	Channel Blockout	Picture -in- picture	Price
Search 1	78386740	7521567.7	23.579545	2.2625657	2	3	2	2	2	332.44
Search 2	161276388	9309337	46.022727	2.6565642	2	3	3	2	2	350.43
Total	239663128	11968197	69.602273	3.4894894						

We can see that as with the one-product optimization solution, an RCA at about $350 with all the best features for screen size, sound quality, channel blockout and PIP is chosen (product "Search 2" above). In addition, a bit more revenue might be derived if the same television were simultaneously offered with the lesser quality Stereo Sound for $18 less, rather than with Surround Sound. Recall that the best one-product simulation achieved revenue of about $226MM, whereas the revenue with two product offering for RCA is about $240MM.

All the simulations conducted this point have lacked the important element of cost information. In the previous example, we don't know whether RCA could stay in business manufacturing and selling these two television sets. Even though we have specified a revenue maximization strategy, RCA may very well lose money on every unit sold if the total costs for these television sets exceed the prices indicated.

As we've emphasized before, having cost information is key to obtaining the most value from product optimization searches. Without key cost data, the search results often return trivial answers that are not in the best interest of a company. In the next section, we'll extend this example by including cost information and conducting a profitability search.

A Note on Standard Errors

For ease of computation, we calculate the standard error of any net revenue, share, utility, or profit for multiple products using the formula for "pooled standard error." For example, considering products *a* and *b*, the standard error of the sum (shares, revenues, etc.) of products *a* and *b* is equal to:

$$SE_{a+b} = \text{sqrt}\,(SE_a^2 + SE_b^2)$$

Because the shares for *a* and *b* are usually negatively correlated in market simulations, this estimate provides a more conservative (slightly larger) value than if the standard error were computed on a new variable representing the net sum for *a* and *b*.

Example #4: New Product Introduction with Competition (Profitability Search)

In the previous example, RCA was interested in creating mid-sized televisions to compete with JVC and Sony, where each competitor already offered two products. We searched for an optimal configuration for the best single and pair of RCA televisions to offer, relative to the competition, to maximize Revenue. However, the fundamental weakness of the approach is that there is no guarantee that following the search recommendations is a profitable strategy for RCA.

In this example, we include the critical aspect of costs. Let's assume that RCA is able to provide a basic fixed cost structure for including various features in these mid-sized television sets:

Base Cost:	$180
Screen Size:	
25" screen	+$0
26" screen	+$20
27" screen	+$35
Sound Quality:	
Mono sound	+$0
Stereo sound	+$25
Surround sound	+$40
Channel Blockout:	
No channel blockout	+$0
Channel blockout	+$15
Picture-in-picture:	
No picture in picture	+$0
Picture in picture	+$30

To specify this cost information, from the ***Scenario Specification*** dialog, click ***Method Settings…*** to reveal the ***Product Search Settings*** dialog. Select the ***Cost, Revenue & Profit information…*** tab. Click ***Add Table…*** to add a "cost table." A cost table lets you specify costs for one attribute at a time, or up to four attributes at a time, if the costs interact with multiple attributes (e.g. if the cost for PIP depends on what screen size is offered). In our example above, there are no interdependencies for costs.

Specify five cost tables:

Table #1: select brand, and specify cost of 180 for RCA, but leave the other two brands blank
Table #2: select screen size, and specify the incremental cost values
(repeat for sound quality, channel blockout, and picture-in-picture).

Once you have specified your cost tables, you can use the ***Cost Checke*****r…** feature on this same dialog to have the ASM add the net cost for product configurations, given your tables. This is useful to check your cost entries prior to spending time conducting searches that may be erroneous.

From the *Search Settings* tab of the ***Product Search Settings*** dialog, select to search for products based on *Profit.*

You are now ready to run a profit search. Click ***Compute!*** from the main ***Market Simulator*** dialog. We've run this search multiple times, using different search methods. A near-optimal solution is shown below:

Product	Profit	Std Err	Product Shares of Preference	Std Err	Revenue	Std Err	Brand	Screen Size	Sound Quality	Channel Blockout	Picture -in-picture	Price
RCA 1	27761476	2505940.6	25.852273	2.3336029	92392157	8339947.8	2	1	2	2	2	357.39
RCA 2	31665338	2879666.4	25.568182	2.3251871	108369883	9855227.7	2	3	3	2	2	423.85
Total	59426813	3817357.3	51.420455	3.2942674	200762040	12910470						

Profit is maximized at $59,426,813 by offering a full-featured TV at $423.85 (slightly undercutting the price for Sony's identically full-featured TV selling for $450) and another less-fully featured TV at $357.39. The compromises in the less-fully featured TV are to reduce the screen size to level 1 (25") and to offer Stereo rather than Surround Sound.

Example #5: TURF-Like Problems

A classic case is often presented of stocking a shelf with the optimal subset of ice cream flavors to maximize the number of people who find at least *n* of the flavors appealing. The method of analysis often used to solve this problem is called TURF (Total Unduplicated Reach and Frequency). TURF initially found use in media research, but we can apply aspects of the approach to standard market simulation problems.

Imagine that a store owner can only choose 6 flavors of ice cream to stock from 50 possible flavors. He wants to choose the 6 flavors that maximize the likelihood that customers will find a flavor that they would like to purchase. One possibility is to survey respondents and identify the 6 flavors that on average have the highest preference. In the absence of heterogeneity of tastes, this strategy works very well. However, it is possible that there are segments of the market that strongly prefer niche flavors, and are less attracted by flavors that appeal to the majority. Even though these flavors lack general appeal, including a niche flavor may increase the likelihood that each customer can find a flavor in the store owner's freezer that will satisfy him or her enough to purchase.

These same kinds of problems can be extended to more complex products, composed of many attributes, such as in conjoint analysis. The ASM can solve these kinds of TURF-like problems. The approach for ASM is to configure a minimum number of product alternatives (as specified by the analyst), such that respondents find at least one of the alternatives highly desirable. More specifically, the objective function for each respondent is the maximum utility or purchase likelihood for any product in the set (rather than the sum or average across the products). After all, we are typically more concerned that just one product alternative satisfy the customer rather than in maximizing the *average* appeal of our offerings for that person.

To illustrate, consider the case of three product alternatives and three respondents:

Utilities for Different Product Alternatives

	Alternative A	Alternative B	Alternative C	Maximum Utility (A, B, C)
Respondent 1	10	20	40	40
Respondent 2	30	10	20	30
Respondent 3	10	20	20	20

			Average:	30

If product alternatives A, B, and C were offered to Respondents 1 through 3, the average utility for the most preferred alternative within the set would be 30. The ASM seeks to find ideal product alternatives A, B, and C, such that this value is maximized.

TURF-like searches can only be performed in the ASM using Purchase Likelihood or Utility simulation methods. All products included in the simulation scenario must be included in the objective function, though all do not need to be dynamic (searched) products.

To implement this feature within the ASM, use the Purchase Likelihood or Utility simulation methods. Select the ***Method Settings…*** button from the ***Scenario Specification*** dialog. On the ***Multiple Searched Product Settings*** dialog, select *Optimize based on the most preferred within the set (TURF-like method).*

An example report is shown below, assuming the objective function is Purchase Likelihood.

Product	Avg Best Purchase Likelihood	Preferred by % Respondents	Brand Name	Tire Type/ Firmness	Tread Type	Weight	Tread Wear Durability
Product 1	52.56	50.00	1	4	3	1	1
Product 2	48.20	33.33	1	1	2	2	2
Product 3	49.34	16.67	1	2	1	3	3
Total	50.57	100.00					

The objective function is to maximize the average purchase likelihood for the most preferred alternative within the set. This is shown as 50.57 (Total Avg. Best Purchase Likelihood) in the report above. For each product alternative, we have summarized the purchase likelihood among those for whom this product had the highest purchase likelihood. We also report what percent of respondents preferred each product (first choice rule). For example, 50% of the respondents preferred product A, and the average purchase likelihood among those individuals for product A was 52.56. The Total Avg. Best Purchase Likelihood is thus the weighted average (weighted by percent of people preferring each alternative) of the corresponding values for each alternative.

To keep TURF-like optimizations from returning trivial results (such as always including the best alternatives at the lowest price), you can fix a subset of the levels to reflect realistic combinations (as we've done above for weight and tread wear durability). You can also include cost information and implement a cost threshold beyond which alternatives cannot be considered.

If you request a TURF-like purchase likelihood simulation to maximize revenue (or profit), the objective function for each respondent is the greatest purchase likelihood among the set of alternatives multiplied by the price (less cost) of that most preferred product times the number of total units sold apportioned to this respondent (based on the respondent's weight). Note that we assume a first choice rule: the respondent is assumed to choose only the most preferred option, with probability equal to the purchase likelihood for that option.

If you are interested in estimating shares of preference using another choice rule for the set of products found using this TURF-like procedure, you can easily specify those products in a standard Randomized First Choice or Share of Preference share simulation.

Example #6: Maximize Appeal Subject to a Maximum Cost

The previous examples dealt with products sold for a price, leading to concrete measures of revenue and profits. However, there are other situations in which price, revenues and profits are either not applicable or only indirectly tied to the optimization problem at hand.

For example, one might use conjoint analysis for determining what type of health insurance plan a large company should adopt. One can enter costs into the ASM, and use these as constraints in the optimization. For example, one might seek to optimize the “Purchase Likelihood” or utility of the health plan (as perceived by the employees), subject to the cost not exceeding some threshold.

You can specify cost thresholds when using any of the search criteria (utility, share, purchase likelihood, revenue, or profit). From the ***Product Search Settings*** dialog, select the ***Cost, Revenue & Profit Information*** tab. On that tab, you can specify to exclude products from the search that cost above a certain threshold, or that costs are limited to a certain valid range (not below x or above y).

Example #7: Minimize Cost Subject to Meeting a Performance Threshold

The researcher can search for products that minimize cost, subject to meeting a certain utility, purchase likelihood, share, revenue, or profit threshold. Building upon the previous example, the problem may be to minimize the cost of a health plan such that the average desirability (purchase likelihood model) of that plan is at least 75%. Such searches can only be done using the Exhaustive search method.

To perform cost minimization searches, from the ***Product Search Settings*** dialog, *Search Settings* tab, select to use *Exhaustive* search. On that same dialog, select to search for products based on *Cost.* (When you search based on cost, the ASM always assumes that lower costs are better, as opposed to all other search criteria, where higher values are better.) Click the ***Settings…*** button, and a dialog appears to let you exclude products according to certain performance thresholds (depending on your simulation method, you can select to filter based on utility, purchase likelihood, share, revenue or profit).

Appendix

How the ASM Module Interpolates Cost Data

The Advanced Simulation Module lets you input cost information for levels of attributes for use in profitability optimizations. Costs can be associated with the levels of a single attribute. Optionally, if the costs for levels of an attribute vary based on levels of other attributes (for example, the cost of an engine size upgrade depends on the make of the vehicle) then the simulator permits you to specify that costs can be conditional on up to four attributes.

When dealing with costs based on just one attribute at a time, the software does linear interpolation between adjacent levels. That is a straightforward procedure that requires no explanation. However, when dealing with two-way or higher interactions for costing data, the situation is considerably more complex. This appendix explains and justifies the procedure used in interpolation for two-way and higher costing interactions.

With one-dimensional interpolation we first find a line containing the two extreme points, and then choose a result from that line. With two-dimensional interpolation the situation is different. We have a total of four points, composing the four corners of a square. For convenience we might label those points as follows:

		Attribute 2	
		Level 1	Level 2
Attribute 1	Level 1	A	B
	Level 2	C	D

In this scheme, point A represents the cost associated with the combination of the first level for both attributes, D represents the combination of level 2 for both attributes, and B and C each represent combinations of level 1 on one attribute and level 2 on the other.

By analogy with the one-dimensional case, we might try to fit a plane to all four points, and then do our interpolation within that plane. Although a plane can always be found to fit three points, it will not generally be true that a plane will fit four points very well. Our circumstance is particularly troublesome because the surface required to fit all four points would be quite warped. If both attributes happen to have only two levels, then the row and column sums of the 2 x 2 array are zero, implying that A = -B = D = -C. If both dimensions have more than two levels the relationship is not so precise, but the surface containing the four points still tends to be saddle-shaped.

Accordingly, we use a nonlinear interpolation procedure, which we describe with a numerical example. Suppose a product has specifications of 1.2 for attribute 1, and 1.6 for attribute 2. Also suppose the costs associated with the four points are:

		Attribute 2	
		Level 1	Level 2
Attribute 1	Level 1	A = $10	B = $15
	Level 2	C = $5	D = $2

Finally, let p1 = the proportion of the distance between the two levels of Attribute 1 for which we would interpolate (0.2 in this example) and p2 = the proportion of the distance between the two levels of Attribute 2 for which we would interpolate (0.6 in this example).

We interpolate by forming the convex sum of four corner points:

$$\begin{array}{rllllll} \text{Result} = & & (1-p1) & * & (1-p2) & * & A \\ & + & (1-p1) & * & p2 & * & B \\ & + & p1 & * & (1-p2) & * & C \\ & + & p1 & * & p2 & * & D \end{array}$$

This procedure describes a surface that includes all four corner points as well as their centroid. Notice that if p1 = p2 = 0, then the result is equal to A. If p1 = p2 = 1, the result is equal to D. If p1 = 0 and p2 = 1, the result is equal to C, and if p1 = 1 and p2 = 0, the result is equal to D. If both attributes have integer specification, this method of interpolation therefore gives the same answer as if no interpolation were used.

This procedure has the desirable property that there are no discontinuities in the regions of the four corner points, as there would be if we tried to fit the points with a plane.

If one attribute has an integer specification but the other requires interpolation, then interpolation is automatically linear for the one attribute requiring it. For example, if p1 = 0, then the result is (1-p2)*A + p2*B, a linear interpolation between A and B. Similarly, if p1 = 1, then the result is a linear interpolation between C and D.

Also, if p1 = p2 = 0.5, the result is (A+B+C+D)/4. Therefore the result at the center of the square is the average of the four corners. If both attributes require interpolation, then the surface is quite regular, with only as much curvature as required to fit the four corners.

For our numerical example, the result is $11.04. The center of the square has a value of $8, but the location corresponding to p1 = 0.2 and p2 = 0.6 is upward and to the right of the center, toward the corner that has a value of $15.

Since our procedure is nonlinear, it would be hazardous to extrapolate beyond the known points. Therefore, the software gives stern warnings if the user attempts to extrapolate beyond measured levels of the attributes.

The same technique for interpolation in two-space can be expanded to three- and four-space cost interpolations. For example, for three-space, interpolation between adjacent levels for three attributes involves 8 points to consider (the 8 corners of a cube). Considering those corners as points A through H, the equation is expanded to:

$$\begin{array}{rllllllll} \text{Result} = & & (1-p1) & * & (1-p2) & * & (1-p3) & * & A \\ & + & (1-p1) & * & p2 & * & (1-p3) & * & B \\ & + & p1 & * & (1-p2) & * & (1-p3) & * & C \\ & + & p1 & * & p2 & * & (1-p3) & * & D \\ & + & (1-p1) & * & (1-p2) & * & p3 & * & E \\ & + & (1-p1) & * & p2 & * & p3 & * & F \\ & + & p1 & * & (1-p2) & * & p3 & * & G \\ & + & p1 & * & p2 & * & p3 & * & H \end{array}$$