

MBC v1.1

**Software for Menu-Based
Choice Analysis**

(Updated June 16, 2016)



**Sawtooth Software, Inc.
Orem, UT**

<http://www.sawtoothsoftware.com>

In this manual, we refer to product names that are trademarked. Windows, Windows 95, Windows 98, Windows 2000, Windows 7, Windows XP, Windows Vista, Windows NT, Excel, PowerPoint, and Word are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

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 mouse clicks 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 Orem, Utah office at 801/477-4700, email: support@sawtoothsoftware.com.

Table of Contents

Foreword

Acknowledgements	1
Prologue	2
Overview	3

Introduction

Quick-Start and Introduction	5
Case Study: TGI Friday's Restaurant	14
Case Study: Ford Fusion	15

Designing MBC Questionnaires

Introduction	17
Constructing Factors and Levels	18
Fractional Factorial Design Strategies	20
Number of Price Levels	22
Single-Block, Multiple-Block and Randomized Plans	24
Premium Pricing and Bundle Pricing	27
Design Testing	29

Preparing Data Files for MBC

Preparing Data Files for MBC	31
Labels File	33
Selecting Independent and Dependent Variables	35
Missing Values	36
Segmentation Variables and Weights	37

Counting Analysis

Introduction	39
Investigating Significant Predictors of Choice	41
Linear or Non-Linear Relationships	45
Positive or Negative Slope Relationships	48
Comparing Counts to Final Market Simulations	49
Which Combinations of Items Were Selected Most Often?	50
Substitutes or Complements?	53

Conceptualizing MBC Models

Conceptualizing MBC Models	55
Example #1	56
Example #2	59
Example #3	62
Combinatorial Dependent Variable Coding Suggestions	65
Conceptualizing the Model Summary	66
Independent Variables: Generic or Alternative-Specific	67

Building Models (Specify Models Tab)

Specify Models Tab	71
Importing Model Specifications	74
Exhaustive Alternatives (Combinatorial) Coding Approach	76
A More Complex Fast-Food Example	78

Analyzing MBC Data Using Aggregate Logit

Analyzing MBC Data Using Aggregate Logit	85
Acknowledgments and Limitations	86
Testing Different Model Specifications with Aggregate Logit	87
Using the MBC Software to Perform Logit Analysis	89
Logit Settings	90
Estimate Models	91
Managing Your Logit Runs	94

Analyzing MBC Data Using HB

Analyzing MBC Data Using HB	95
Using the MBC Software to Perform HB Analysis	96
HB Settings	97
Estimate Models	99
Managing Your HB Runs	102

Simulating Respondent Choices

Simulating Respondent Choices	103
Aggregate vs. Individual-Level Predictions	104
Market Simulation Math	105
The Simulations Tab	106
Simulator Validation	111
Netting Dependent Variables	112
Sensitivity Analysis	113
Excel Simulator	121

Simulating Combinatorial Choice Outcomes

Simulating Combinatorial Choice Outcomes	123
HB Draws and Combinatorial Choice Predictions	124
Simulating Combinatorial Outcomes	125
Running Combinatorial Simulations in MBC	127
2006 Fast-Food Menu Study Results	129
2011 Fast-Food Menu Study Results	130

Tutorial: Configuring an Automobile

Tutorial: Configuring an Automobile	133
---	-----

Polytomous Logit

Polytomous Logit	147
------------------------	-----

Appendix: Technical Details

Chi-Square Computations 151
Relationship Chi-Square 152
Non-Linearity Chi-Square 154
Coding Choice Tasks 157
Prior Covariance Matrix for HB Estimation 159
Coding Linear and Log-Linear Functions 160
Utility Constraints 163

References

References 165

Index **167**

1 Foreword

1.1 Acknowledgements

Many people have had a role in inspiring, consulting, programming, testing, and otherwise helping with this documentation and with the MBC software itself. For inspiration, we owe much to David Bakken, John Liechty, and Steve Cohen for their published articles on this subject. We've borrowed ideas from their work and have cited them in this booklet. Tom Eagle, Peter Lenk, Jeffrey Dotson, John Howell, and Aaron Hill provided ideas and guidance as we formulated our plans for the software and our direction on the analytical procedures. Since the release of MBC, Paulo Cordella, Carlo Borghi, Kees van der Wagt, Gerard Losschilder, Christian Neuerberg, and Chris Moore have undertaken and published methodological work comparing the methods we use in MBC to other competing models of menu choices (including sampling of alternatives), validating the usefulness of our approaches. We should note that not all of these contributors endorse the approaches we have taken, but their insight has been very helpful. During the development of the software over 180 people signed on as "development partners" and agreed to respond to periodic feedback emails that we sent. This proved very useful as we made decisions about philosophical direction and software functionality.

Software of course needs quality control and testing, and Murray Milroy deserves heaps of credit for his careful and often inventive work (attempting to do sometimes nonsensical things that the typical human would never try). Walt Williams is the lone master programmer who cranked the software out from start to finish, and is known to create the fastest HB code on the planet. We are ever so grateful to the many customers who have served as beta testers for the software. And, of course, Rich Johnson deserves a great deal of credit for hours spent counseling us regarding our strategic direction and on some of the technical details of MBC software.

Excel and Word are trademarks of Microsoft Corporation.

1.2 Prologue

For over 35 years, Sawtooth Software has created software tools to solve quantitative market research problems, especially those involved with predicting consumer behavior. The real client problem of predicting demand for complex products and services led our founder, Rich Johnson, to develop ACA (Adaptive Conjoint Analysis), released as a commercial software program in 1985.

In the early 1990s, we noticed that certain pricing research problems weren't being served well by ACA. This led to the development of the first version of CBC software in 1993. Johnson certainly didn't invent the technique, as Dan McFadden had laid the groundwork in discrete choice methods in the 70s, and Jordan Louviere had championed discrete choice experiments within the marketing community in the 80s.

Over the last few decades, we have worked closely with our software customers in technical support, training, and consulting. When we see common problems that don't seem to be solved well by our existing tools, it naturally leads us to ponder whether something new could be developed to meet the challenge.

For example, in the late 1990s, we heard of ACA users using 20 to 30 binary (two-level, on/off) attributes, when what they really needed was either the Method of Paired Comparisons or MaxDiff (originally invented by Jordan Louviere in the late 1980s). Steve Cohen (In4mation Insights) had written some fine articles on MaxDiff (based on Jordan's work) that were winning industry awards. MaxDiff soon followed as a Sawtooth Software product.

Most recently, we observed many of our clients wanting to do menu-based choice tasks, but struggling to get CBC and MaxDiff software to handle the problem. Prior to 2012, we tackled some MBC projects on our own using power tricks involving CBC and CBC/HB, a lot of data processing, and Excel simulators. The undertaking was satisfying, but ultimately very time consuming. Those experiences led to the first version of Menu-Based Choice (MBC) software, released in 2012.

We do not claim to be world's leading experts in menu-based choice. Nor do we claim that the approach we've taken is always the best possible solution. But, a great deal of accumulated evidence so far strongly suggests that we've chosen an approach that is practical, robust, and scalable, to solve varied and often demanding menu-based problems.

1.3 Overview

Menu-Based Choice (MBC) is software for analyzing a variety of menu-based and discrete choice problems in survey-based market research. It does not design the questionnaires or collect the data. It does, however, analyze results and provide a market simulator.

This software requires more expertise to use properly than our other conjoint analysis tools. *The user should have solid background in CBC and multivariate statistical modeling, especially in terms of building models (regression, MNL) and the theory behind coding independent variables.* While the software manages most details involving the data processing, independent variable coding, model estimation and simulations, the user must understand and direct the process intelligently.

The steps to use the program are as follows:

1. Using your own means, you design the questionnaire and collect the data. This may be done using our CBC software's Complete Enumeration or Shortcut designs), blocked, or "fixed" designs (such as using our CVA system's capable designer).
2. Using other tools, such as Excel or SPSS, you prepare the data in .csv format. The data include CaseIDs, independent variables, and dependent variables. A separate demographics file may be prepared, which may include weights and segmentation (covariate) variables.

The final steps all are done using the MBC software:

3. Open the data file and identify the independent and dependent variables. If an optional demographics file is available, point MBC to that file as well.
4. Use Counting Analysis to examine the effect of independent variables on the menu choices (the dependent variables). This allows you to develop hypotheses regarding which variables (such as prices for menu items) affect which choices on the menu, and the shape of utility functions (linear or non-linear).
5. Use Counting Analysis to report the most commonly selected bundles of items on the menu. This identifies which combinations were most commonly selected, accumulated across the various manipulations of your menu design.
6. With the aggregate logit routine provided, you can formally test which variables are significant predictors of choice and the appropriate functional form for price on different menu choices (e.g. linear, non-linear). You may use the aggregate logit routine for building final models as well, though many researchers prefer HB estimation.
7. Using the hierarchical Bayes (HB) routine provided, you estimate utility weights for the independent variables (attribute levels) that are predictive of respondent choices on the menu. Typically, you build multiple sub-models, where each sub-model predicts a specific portion of the menu.

8. MBC includes a [market simulator](#) to predict the proportion of respondents expected to select each item on the menu, given specific menu prices or realizations of other independent variables. If you've employed HB estimation, the simulator can also predict the combinatorial selections of items.

MBC also can create an Excel-based simulator that you may share with clients without fee.

Technical Support

As with our other software systems, we provide free technical support during business hours. Technical support is intended to respond to difficulties in making the software function. It is not intended to provide free consulting help to assist with the many design and conceptual aspects of analyzing MBC studies.

2 Introduction

2.1 Quick-Start and Introduction

See our [Tutorial](#) for a click-by-click guide using a sample data set. Or, follow the Quick-Start instructions directly below:

Quick-Start to Using MBC Software

Assuming you have already fielded your MBC study and collected data:

1. Using a program like Excel, create a .csv data file (one row per choice task) that contains CaseID (first column) followed by independent variables and dependent variables. [Click here](#) for more details.
2. Using a program like Excel, create a .csv labels file that provides labels and level values associated with the independent and dependent variables for your study. The labels are for your convenience in reading analytics output. The level values are used as input to estimation of linear or log-linear coefficients. [Click here](#) for more details.
3. Start the MBC software, and use the Project Wizard to point to your data file and optional [segmentation/weights](#) file.
4. Use the Variables tab to specify which variables are independent and dependent variables, and to import your labels and level values from the labels file.
5. Analyze the data using [Counting Analysis](#), [Logit](#), and [HB](#).
6. Conduct [Market Simulations](#).

Increasingly, stated preference choice projects involve Menu-Based Choice scenarios (MBC) where respondents can select one to multiple options from a menu. This is not surprising, given the fact that buyers commonly are allowed to customize products and services (mass customization). Examples include choosing options to put on an automobile, selections from a restaurant menu, banking options, configuring an insurance policy, or purchasing bundled vs. *a la carte* services including mobile phones, internet, and cable.

Here is a very simple menu, where the respondent chooses options and a total price is shown:

Which of the following options would you buy? Select as many as you wish, or none of the items.

- Option A \$12
- Option B \$24
- Option C \$7
- Option D \$55
- Option E \$3

Total Price of Selected Options: \$22

Figure 1.1

Respondents can select from zero to five choices on the menu in Figure 1.1. This particular respondent has selected three items, for a total price of \$22. There are $2^5=32$ possible ways respondents can complete this menu. We designed the questionnaire so that some or all of the prices vary across respondents, or even across repeated menus given to the same respondent.

If we vary prices across menu questions, we can observe whether changing prices influences what respondents pick. Economic theory, of course, suggests that as the price of a menu item *increases*, its likelihood of choice will *decrease*. But, is that relationship fairly linear? Does reducing the price for an item cause a different item on the menu to be chosen more likely (or even *less* likely)? In other words, are items on the menus *substitutes* or *complements*? Menu-Based Choice (MBC) experiments (yet another form of conjoint analysis) can investigate such issues.

A second example below involves the choice of a base model followed by configuration of options on that base model:

Which of the following would you buy? Select a Base Model, and then any add-on options you wish.

- Base Model 1 \$200
- Base Model 2 \$275
- Base Model 3 \$550

- Option A \$12
- Option B \$24
- Option C \$7
- Option D \$55
- Option E \$3

Total Price of Selections: \$297

Figure 1.2

There are $(3)(2^5) = 96$ possible combinations of selections for this menu. Prices might vary from menu to menu, or maybe some options (such as Base Model 3) are not always available. With larger menus, there are often thousands or even millions of possible ways to make choices from the menu.

Over the past two decades, there has been increasing interest in designing and analyzing menu-based choice questionnaires, as they often more realistically reflect real-world buying situations than standard Choice-Based Conjoint (CBC) or ratings-based conjoint. Examples in the literature include articles by Liechty *et al.* (2001), and Cohen and Liechty (2007).

Cohen and Liechty (2007) write: "While choiceboards and menus have become more attractive, how should the practicing researcher design studies and analyze the data to better understand mass customization with menus? ... At first glance, understanding product bundles—which consist of discrete features—seems like something that conjoint analysis should be able to handle. But, the complexity of the menu situation renders traditional conjoint analysis wholly inadequate."

A typical example of a menu-based problem is the classic "BYO" or "Configurator" task. Much like the Dell website of today, respondents can be shown different computer features and prices, and are asked to configure their ideal computer (see Figure 1.3). As respondents make choices such as for brands, hard drive sizes, and processor speeds, the total price for the configured product is shown. After respondents are satisfied with the configuration and total price, they moved to the next question.

<i>Please select the PC you'd be most likely to purchase:</i>
<input checked="" type="radio"/> Dell (\$500) <input type="radio"/> IBM (\$600) <input type="radio"/> Compaq (\$550) <input type="radio"/> Acer (\$525)
<input type="radio"/> 100 GB Hard Drive (\$0) <input checked="" type="radio"/> 200 GB Hard Drive (\$60) <input type="radio"/> 500 GB Hard Drive (\$90)
<input type="radio"/> 1 MB RAM (\$0) <input type="radio"/> 2 MB RAM (\$80) <input checked="" type="radio"/> 4 MB RAM (\$150)
<input checked="" type="radio"/> Base Processor (\$0) <input type="radio"/> Enhanced Processor (\$250)
<input type="radio"/> 17-inch screen (\$0) <input checked="" type="radio"/> 19-inch screen (\$40) <input type="radio"/> 21-inch screen (\$90)
<input type="radio"/> No office (\$0) <input type="radio"/> Office (\$200) <input checked="" type="radio"/> Office + Access (\$240)
<input type="radio"/> 90-day warranty (\$0) <input type="radio"/> 180-day warranty (\$50) <input checked="" type="radio"/> 365-day warranty (\$100)
Total Price: \$1090

Figure 1.3 (BYO)

The Sawtooth Software community was made aware of menu-based choice experiments through articles given at the Sawtooth Software conference by Bakken and Bayer in 2001, and Bakken and Bond in 2004. The 2001 paper involved BYO experiments, whereas the 2004 paper involved a menu-based choice of bundles vs. *a la carte* items. (Sawtooth Software conference proceedings including these articles may be downloaded for free from our website at <http://www.sawtoothsoftware.com/education/techpap.shtml>.)

The 2004 paper by Bakken and Bond was particularly interesting, and here are some examples from that paper.

In the first example, respondents choose a restaurant to visit, and whether to buy a bundled item or *a la carte* items.

Below are three different restaurant options with menu items and their respective prices. Please select what you would typically choose from the items from one restaurant.

Please keep in mind that you cannot choose items from more than one restaurant.

McDonalds	Burger King	Wendy's
<input type="checkbox"/> Big Mac \$2.25	<input type="checkbox"/> Whopper \$3.25	<input type="checkbox"/> Classic Double \$2.75
<input type="checkbox"/> Large French Fries \$1.75	<input type="checkbox"/> Large French Fries \$1.49	<input type="checkbox"/> Biggie Fries \$1.49
<input type="checkbox"/> Large Drink \$1.49	<input type="checkbox"/> Medium Drink \$0.99	<input type="checkbox"/> Biggie Drink \$1.45
<input type="checkbox"/> Big Mac Extra Value Meal \$4.99	<input type="checkbox"/> Whopper Value Meal \$5.25	<input type="checkbox"/> Classic Double Combo Meal \$5.25

Figure 1.4

In the second example from Bakken and Bond, respondents chose among different options for purchasing scalpels, forceps, and sutures. Respondents could buy them together as a package, or separately from two different suppliers.



Please review the information below and make your selection by either selecting each of the products individually from one manufacturer or the other, or selecting all products from one brand as a bundle (all from Brand A or Brand B).						
	Brand A			Brand B		
Frequency of Sales Rep Contact:	Weekly			Monthly		
Telephone Technical Support Availability:	12 hours a day, 5 days a week			24 hours a day, 7 days a week		
Contract Compliance Requirement:	50%			90%		
	Scalpels	Forceps	Sutures	Scalpels	Forceps	Sutures
Overall Quality:	Superior	Expected	Expected	Superior	Expected	Superior
End User Preference:	Good	Preferred	Good	Preferred	Good	Good
Individual Product Price:	5% Less	5% higher	Current	5% Higher	Current	10% Lower
To Make Your Selection...						
Choose one of each product here 	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
OR	Price for Scalpels, Forceps, and Sutures together from Brand A			Price for Scalpels, Forceps, and Sutures together from Brand B		
	20% Lower			10% Lower		
Select all products from one brand here 	<input type="radio"/>			<input type="radio"/>		

Figure 1.5

Bakken and Bond used logit-based model estimation, and they posited that respondents approached the task in Figure 1.5 with two-step decision making. First, the respondent chooses

any one of the bundled offers or rejects all bundles. If all bundles are rejected, then the buyer chooses one or more *a la carte* items from the menu.

The separate logit models making up the two stages (estimated via HB and having individual-level utilities) were linked via a first-choice market simulator. First, respondents were predicted to pick a bundled offer or not. If respondents were projected to not choose a bundle in the first stage, then their most likely choices among the *a la carte* selections were predicted. The simulator accumulated the choices across respondents to create market predictions.

While we have not chosen to build simulators for our MBC software in precisely the same way, we employ a principle that Bakken and Bond advocated in their paper: specifically, the idea that decisions for some menus may be modeled as having a nested, or staged structure. We also employ logit-based models, but use the logit equation to predict likelihoods of choice rather than the first choice rule.

Others have advocated multivariate probit analysis for menu-based problems (Liechty *et al.* 2001, Cohen and Liechty 2007). Multivariate probit seems to provide a more theoretically complete model, directly incorporating the idea that items on the menu can be substitutes or complements. The researcher runs a single model, rather than a sequence of separate logit models. The logit-based models seem to provide good simulated share predictions of the likelihood of the sample selecting different items on the menu. But, multivariate probit may particularly have an advantage when the goal is to predict the *combinations* of menu selections that each individual makes. Other researchers such as Eagle have also suggested nested logit for menu choices. Perhaps in the future we will investigate multivariate probit or nested logit for MBC. For now, we are more familiar with logit analysis, and find that it tends to work quite well. We worry about the scalability of multivariate probit to the more complex types of menus that our users may wish to attempt. The logit framework we've chosen is very flexible, and can handle from small to large menuing problems.

Starting in about 2006, we included an MBC example in our advanced CBC tutorials. To provide data for that tutorial, we fielded an Lighthouse Studio questionnaire (our program for web-based interviewing) where we programmed (using Free Format questions and Javascript verification) a simplified version of a fast-food restaurant menu (a generic combination of the McDonald's and Burger King menus), as shown in Figure 1.6.

Menu Scenario #1: Please imagine you pulled into a fast-food restaurant to order dinner for <u>just yourself</u> . If this were the menu, what (if anything) would you purchase?		
<input type="radio"/> Deluxe Hamburger Value Meal -Deluxe Hamburger -Medium fries -Medium drink \$3.99	<input type="radio"/> Chicken Sandwich Value Meal -Chicken Sandwich -Medium fries -Medium drink \$5.59	<input type="radio"/> Fish Sandwich Value Meal -Fish Sandwich -Medium fries -Medium drink \$3.99
(Only order sandwiches, fries or drinks from this area if you did not pick a value meal above.) Sandwiches: <input type="radio"/> Deluxe Hamburger \$1.99 <input type="radio"/> Chicken Sandwich \$3.59 <input type="radio"/> Fish Sandwich \$1.99 Fries: <input type="radio"/> Small \$0.79 <input type="radio"/> Medium \$1.49 <input type="radio"/> Large \$1.69 Drinks: <input type="radio"/> Small \$0.99 <input type="radio"/> Medium \$1.69 <input type="radio"/> Large \$2.19		Salads: <input type="radio"/> Cobb dinner salad \$4.79 <input type="radio"/> Grilled chicken salad \$4.39 Healthy Sides: <input type="radio"/> Carrots/Celery with Ranch dressing \$1.19 <input type="radio"/> Apple slices/Grapes with dipping sauce \$0.99 Desserts: <input type="radio"/> Apple/Cherry/Berry pie \$0.99 <input type="radio"/> Cookies \$1.19 Total Price: \$_____
<input type="radio"/> I wouldn't buy anything from this menu. <input type="radio"/> I'd drive to a different restaurant, or do something else for dinner.		

Figure 1.6

681 respondents each completed 10 choice tasks like the one in Figure 1.6. Within each category, the selections were mutually exclusive. As with the Bakken and Bond example (Figure 1.5), this is a bundling vs. *a la carte* example (mixed bundling). The value meal bundles were always offered at a discount compared to the cost of purchasing the separate items *a la carte*. We also employed some special pricing rules, to ensure that the prices for larger size fries or drinks were always more expensive than the smaller sizes.

The next example is one that all CBC users should find familiar:

If these were the only three options, which <i>one</i> would you choose?		
Brand A	Brand B	Brand C
Low Performance	Medium Performance	High Performance
Average Service	Average Service	Superior Service
\$100	\$200	\$300
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure 1.7

Figure 1.7 is the standard CBC task (with brands sorted in natural order), which incidentally may also be analyzed using MBC software. Sawtooth Software's CBC program only considers the main effects and first-order interactions for CBC questions during HB estimation. MBC software can model those (as the user specifies the coding properly). Although it's probably unnecessary for a standard CBC design analyzed under HB, MBC additionally can incorporate cross-effects into its models (such as the effect of Brand A's characteristics on the choice likelihood of Brand B). And, if you needed to extend the CBC example (Figure 1.8) to use a check-box (select all that apply) response, MBC could be used to analyze the data, whereas our standard CBC software would treat the data as constant-sum allocation.

Which of the following would you choose? Select all that apply (or None of the boxes, if none appeal to you).		
Brand A	Brand B	Brand C
Low Performance	Medium Performance	High Performance
Average Service	Average Service	Superior Service
\$100	\$200	\$300
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 1.8

In pharma studies, we often hear of projects involving CBC-looking tasks with multiple drugs and a patient profile in the prologue. The task involves showing a patient profile with varying characteristics along with an array of drugs with varying characteristics, where we ask doctors to select which drug or combination of drugs they would prescribe to a patient with those characteristics.

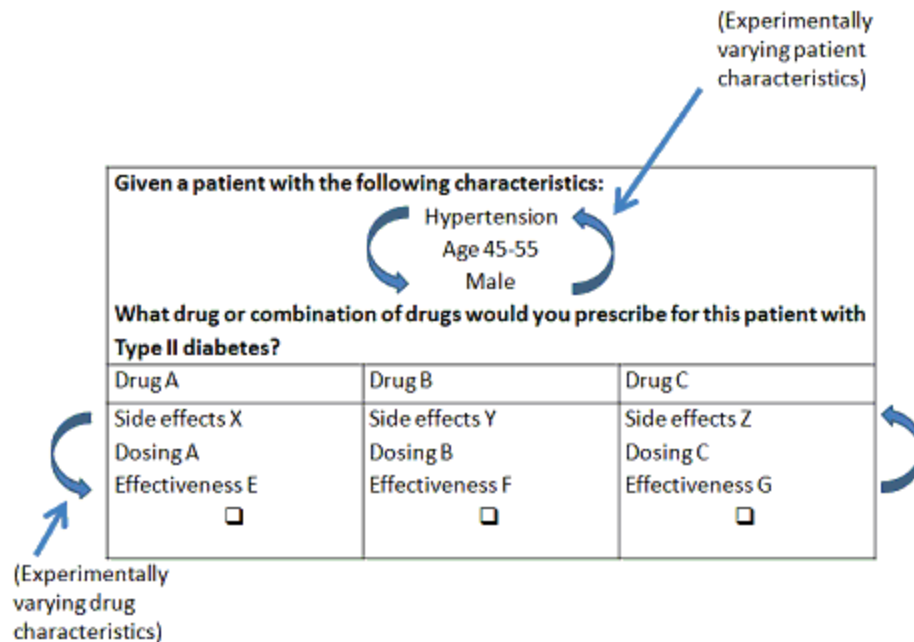


Figure 1.9

Such CBC-looking studies cannot be analyzed with our standard CBC software, but they can with our MBC tool. The likelihood of selecting each drug is modeled as a function of that drug's characteristics as well as the characteristics of the patient profile.

MBC software may also be used to estimate aggregate polytomous logit (situational choice) models, such as a situation where the researcher wishes to study which demographic characteristics of unwed teenage mothers are drivers of three possible outcomes for the pregnancy: a) terminate pregnancy, b) keep the baby, or c) put the baby up for adoption. One of the alternatives is held as the reference (0-utility) option, and we fit alternative-specific coefficients for the demographic characteristics to predict the other two outcomes. The resulting simulator predicts the likelihood of all three outcomes based on any combination of demographic characteristics for a given teenage mother.

As you can see, there are myriad possibilities within the family of discrete choice and multi-check menu choice questionnaires. Many questionnaires are hybrids, with characteristics of both discrete choices among mutually-exclusive options and multi-check menus. Some also involve hierarchical (staged) processes, where respondents first choose among multiple options, and then select among other options conditional on the first-stage selection. MBC software is flexible enough to handle the examples in this chapter, and many more possibilities that you or your clients might envision.

2.2 Case Study: TGI Friday's Restaurant

Adapted from "Analysing Pick 'n Mix Menus via Choice Models to Optimise the Client Portfolio—Theory and Case Study" by Chris Moore, GfK, NOP, 2010 Sawtooth Software Conference Proceedings.

At the 2010 Sawtooth Software Conference, Chris Moore (GfK, NOP) described an MBC project conducted by his firm for TGI Friday's restaurants. The study design followed a two-stage process (both stages within the same questionnaire). In the first stage, a CBC study was used to predict which of several restaurants (including TGI Friday's) respondents would choose to visit, given changes in the main aspects of the menu. In the second stage, a menu-based choice study was used to predict what buyers would pick from the TGI Friday's menu, given price changes just for TGI Friday's items. 1,500 online respondents completed the survey.

For the first-stage model, Chris employed analysis similar to a CBC study. The second-stage TGI Friday's menu tasks were analyzed with an approach nearly identical to that employed by our MBC software. The market simulator was used (together with an Excel-based optimization program called Crystal Ball) to optimize pricing for key dishes to ultimately increase net profit for the client.

The model predicted holdout scenarios well, with an R-squared better than 0.95. But, the proof of the pudding was whether the firm could increase profits by following the researcher's recommendations and implementing the optimized menu. That they did, and after six months of sales data, the test store showed 15% higher profit than the control stores.

2.3 Case Study: Ford Fusion

The next case study we present below (Ford Fusion) as far as we know did not use any of our software tools, but shows how menu-based configurators can be used to collect valuable information to direct product design and pricing efforts.

Source: "Net Gives Ford Fusion Pricing Clues: A bit of con-fusion on the Fusion Web site" (July 10, 2005) by Paul A. Eisenstein, The Car Connection, downloaded from: http://www.thecarconnection.com/tips-article/1007615_net-gives-ford-fusion-pricing-clues.

When Ford announced plans to launch its critical new Fusion sedan at \$17,995, it took many observers by surprise, especially when compared with competitors such as the Hyundai Sonata, which starts \$500 higher.

In today's hotly competitive market, the launch price of a new vehicle is critical, but so are other factors, including the standard equipment package and option prices.

So the automaker has taken an unusual approach to getting consumers to tell what they expect to pay for the new four-door, and what sort of options they expect on the car. It "added a layer of real-time research," using a specially designed Web site.

It's common these days for manufacturers to set up dedicated Web sites well before a new product hits the road. That provides a way to communicate with the curious, and keep the attention of so-called "hand-raisers," especially likely prospects. The Fusion site initially featured several subtle differences that only the most observant visitors might have noticed.

The most critical feature was built into the Fusion pricing calculator. Prior to setting the \$17,995 base, the automaker programmed in a range of possible prices, Geist reveals, between the \$13,600 price of the smaller Ford Focus and the \$22,700 for the full-size Five Hundred.

"We call it randomized pricing," explains Geist "It changed every time a consumer would go to the Web site. We were trying to see if the take (likely purchase) rate changed if the price was higher or lower."

The system also created a randomized list of standard and option features, helping the automaker get a better sense of what web visitors expected on the Fusion. One thing they asked for was all-wheel drive. Ford had expected that option to have a much lower rate of demand on the smaller Fusion than on the Five Hundred, said Geist, "but they came in pretty darn close."

Unfortunately for Ford, it made an early decision not to make the AWD package available at launch. But it has been able to adjust other option features in time for Fusion's upcoming debut. Indeed, with the low base price, the Web site research indicated buyers will "want a richer mix of premium options," says Geist, than originally forecast in-house.

The automaker got a lot more feedback than it had anticipated. Of the first 50,000 visitors to the Fusion site, a full 30,000, or 60 percent, "built" a vehicle to their specifications. Part of the challenge for Ford was to then interpret that data.

"When you're not stroking a check, you may be designing a vehicle that's a little more expensive than what you'd actually buy at the dealership," cautions Geist. But he said the automaker was able to compare the Web data with other research for Fusion. There have been the usual clinics and focus groups, but the online approach has been especially helpful in making last-minute tweaks.

"If they can configure the car correctly," says Dan Gorrell, chief analyst with the California research firm, Strategic Visions, "it will help Ford improve customer satisfaction, and generate more buzz."

"This is a logical progression" in the effort to get customers involved in the development of a new vehicle, says Geist, hinted that in the future, projects like this could be put in place "even earlier in the process."

3 Designing MBC Questionnaires

3.1 Introduction

We've design the MBC software to analyze a variety of menu-based choice problems, samples of which we showed in the previous section. However, the experimental design for these projects as well as the creation of the questionnaire and data collection is left for you to manage using other tools (described below).

There are common elements involved in typical MBC questionnaires:

1. Respondents choose among menu items, each typically featuring item-specific prices (also known as alternative-specific prices).
2. Respondents make from zero to multiple selections per choice task (but do not specify volume/quantity of purchase).
3. The price of the total ticket is shown, and is updated as respondents "buy" more items on the menu (it isn't necessary to show the total price, but it is common to do so).
4. There often are constraints on the combinations of items that can be chosen.
5. There often are constraints on pricing (e.g. medium size drinks must be higher price than small size; price of the bundle must be less than purchasing the same items *a la carte*).

You may do questionnaire development and data collection with any questionnaire instrument (even paper-and-pencil) or web survey tool of your choice, including Sawtooth Software's Lighthouse Studio program. We generally recommend using (controlled) randomized design strategies (such as the Complete Enumeration or Shortcut strategies in our CBC software), as they are robust and convenient to use for MBC studies. If you use Lighthouse Studio, you will need to customize the questions using Free Format and typically some customized Javascript code that you must write on your own.

The focus of these next sections is on experimental design, and we assume the reader already is proficient in conjoint and CBC analysis.

3.2 Constructing Factors and Levels

In conjoint analysis we consider multiple factors (attributes) where each attribute has at least two levels. Menu-based choice problems also involve multiple factors, each having multiple levels. Whereas we often think of a CBC question as being composed of multiple product concepts (cards), we should think of the entire MBC menu question being represented by a *single* card. This allows researchers to use the familiar tools for conjoint design with MBC experiments (CBC or CVA Software, Warren Kuhfeld's SAS routines), except that the number of factors for MBC experiments will often be much larger than for traditional conjoint or CBC.

Consider an example introduced previously:

Which of the following would you buy? Select a Base Model, and then any add-on options you wish.

- Base Model 1 \$200
- Base Model 2 \$275
- Base Model 3 \$550

- Option A \$12
- Option B \$24
- Option C \$7
- Option D \$55
- Option E \$3

- Total Price of Selections: \$297

Figure 2.1

Let's imagine that Base Model 3 was sometimes available and sometimes not (controlled by a binary "on/off" factor). Furthermore, each option on the menu can vary in terms of its prices. We could draw truly continuous random prices for each option (within a given range), but it makes it easier for both design and analysis if we select a limited number of discrete prices, such as five per option.

The nine factors in our experiment, with their levels are as follows:

- | | |
|----------------------------------|--|
| 1) Prices for Base Model 1: | \$150, \$175, \$200, \$225, \$250 |
| 2) Prices for Base Model 2: | \$250, \$275, \$300, \$350, \$400 |
| 3) Availability of Base Model 3: | Available, not Available |
| 4) Prices for Base Model 3: | \$400, \$500, \$550, \$650, \$750 |
| 5) Prices for Option A: | \$10, \$12, \$14, \$16, \$20 |
| 6) Prices for Option B: | \$16, \$20, \$24, \$28, \$36 |
| 7) Prices for Option C: | \$3, \$4, \$5, \$6, \$7 |
| 8) Prices for Option D: | \$50, \$55, \$60, \$70, \$80 |
| 9) Prices for Option E: | \$2.00, \$2.50, \$3.00, \$3.50, \$4.00 |

The total number of possible combinations (full-factorial) is: $5^8 \times 2^1 = 781,250$. Thankfully, respondents are not required to respond to all 781,250 combinations to allow us to estimate the effect of these variables on choices. We can use a subset (fractional factorial) of the possible combinations to include in the questionnaire.

3.3 Fractional Factorial Design Strategies

With the very earliest form of conjoint analysis (card-sort conjoint), analysts used design catalogues or specialized software to select a very limited number of cards to display to respondents. The formula to determine the *minimum* number of cards to enable utility estimation (main effects) is: $\text{Total_Levels} - \text{Total_Attributes} + 1$.

Consider a conjoint study with 6 attributes each with 3 levels; the minimum number of cards needed is $18 - 6 + 1 = 13$. Using the minimum is not recommended in practice, since there are no degrees of freedom and therefore response error may result in severe errors of estimation. For traditional (card-sort) conjoint, many analysts would want to use 1.5x to 2x as many cards as parameters to be estimated. 18, 21, or 24 cards might be advantageous choices in this case, as all are divisible by 3 (each attribute has 3 levels), leading to the possibility of finding a level balanced, near-orthogonal plan. That said, designs that sacrifice a modest degree of level balance and orthogonality may still be quite adequate in practice. Sawtooth Software's CVA program has a very good designer for working with traditional conjoint designs (and also MBC designs, as we'll show later).

As we just illustrated with just a few attributes each with a few levels often about 18 cards would be enough to support precise estimates of all main effects. Since card-sort conjoint interviewing was typically done on paper, it was easiest when there was just one deck of cards to replicate across respondents. Each respondent saw the same set of cards (but perhaps in random order). This represents a one-version plan.

Traditional conjoint design software (such as Sawtooth Software's CVA) can be used effectively for MBC questionnaires. Each card defines an MBC task. For the MBC example in Figure 2.1, we have 8 attributes with 5 levels, and 1 attribute with 2 levels. The minimum number of cards is $42 - 9 + 1 = 34$. To obtain 1.5x as many cards as parameters to estimate (giving us some degrees of freedom), we'd have $(34)(1.5) = 51$ cards. To obtain a level-balanced plan, we would probably try either 50 or 60 cards (divisible both by 5 and 2). This is probably too many MBC tasks to ask any one respondent, so we would consider dividing the cards into different blocks of questions, such as 5 or 6 blocks each with 10 questions.

Designs became more complex with CBC, as researchers were interested not only in main effects, but sometimes in interactions and cross-effects. These higher-order effects required larger numbers of cards (concepts) and choice tasks to estimate the parameters of interest with reasonable precision—typically more choice tasks than any one respondent could complete. For example, if the experimental design called for 72 total choice tasks (each made up of multiple concepts), researchers might divide these into six blocks of 12 questions. Each respondent would complete just one of the blocks, and all the effects accommodated by the design could be estimated by pooling the analysis across respondents.

Why all this discussion of historical methods for designing conjoint and CBC? If you plan to collect the data using a method that requires you to hard-code the specific questions respondents will see, then you will be striving for MBC questionnaires that can be fielded using a limited number of questions and question blocks (our CVA software could be quite useful in designing such studies).

However, if you use a computerized interviewing method that can dynamically display the MBC questionnaire based on a database of pre-designed tasks, or based on designs generated on-the-fly using randomized methods (where each respondent gets a unique questionnaire version), you will have greater flexibility in handling complex designs and MBC questionnaires. Also, the useful counting analysis built into MBC software is best supported by randomized plans (where respondents are randomly assigned to receive one of many available blocks of the design).

3.4 Number of Price Levels

To study price sensitivity requires that we vary prices and observe respondent reaction. At minimum we need to have two price points to study price sensitivity. But, if we use just two price points, we haven't observed how respondents react to prices in between those two points. We can only assume some function to estimate reactions to interior prices (such as linear). With three price points (e.g. \$200, \$300, and \$400), we have an opportunity to capture some non-linearity in the price function, but with just one kink. More price points lead to greater opportunity to capture non-linearity in the price function, and to assess which specific points along the function are associated with regions of non-linearity (elbows, or dramatic changes in slope). It would be tempting to test dozens of price points along the range, but this approach comes with some drawbacks, which we'll discuss further below.

It is not a requirement in MBC to use equidistant spacing across price levels. For example, prices: \$200, \$250, \$300, \$400, \$600 are reasonable, as well as: \$200, \$300, \$400, \$500, \$600. Both price level specifications measure five price points between \$200 and \$600. But, the first probes more granular spacing at the lower prices. If at all possible, try to include specific price levels that the client is interested in taking to market, or that might represent psychological break-points (thresholds of resistance) for different segments of respondents.

We suggest you maintain symmetry across factors of the design, in terms of number of levels. It wouldn't make sense to have one item vary across 3 prices and another item vary across 10. The conjoint literature is replete with cautions about the "Number of Levels" effect, and we should expect this could lead to bias in MBC experiments as well.

We generally recommend including about 5 levels of price variation per menu item (no fewer than 4 and no greater than about 9). The decision regarding number of price levels to include per menu item for a given study depends on a number of factors:

- The type of design you plan to use: fixed (limited block) or randomized plan
- The utility function you plan to specify for each price effect
- The sample size

We'll discuss each of these points in turn.

If you plan to generate a minimal-sized design (with as few cards and blocks as possible), then increasing the number of price levels per item on the menu will make it harder to generate a plan with a reasonably small total number of cards (MBC questions). You'll therefore want to use relatively few price levels per menu item. If, however, you are able to use randomized design strategies (e.g. unique questionnaire version for each respondent) and questionnaires built on-the-fly via computerized interviewing, this gives you greater flexibility to study more levels of price.

Many researchers will want to estimate non-linear functions for price, using the "part-worth" functional specification (dummy-coding). With dummy-coding, a k-level price attribute is estimated as k-1 parameters in the model. If five levels of price are used for a menu item, the effect of its price can be captured as 5-1 = 4 parameters. However, 20 total price levels would require 19 parameters to estimate. Not only would this tax the estimation, but it would likely lead to utility reversals among adjacent price level estimates.

Many readers will recognize that using dummy-coding and effects-coding lead to either identical or nearly identical model fit and predictions. Our CBC and CBC/HB software employ effects-coding. Our CVA software employs dummy-coding. We have decided to employ dummy coding for ease of developing MBC and user understanding. With dummy-coding, the reference level is set to have a utility of 0.

The sample size also affects the ability to investigate non-linear functions for price using the dummy-coded "part-worth" functional form. We should ensure that enough occurrences of each price point are available in the data to lead to reasonably stable estimates. MBC experiments might involve 800 respondents, each having seen 8 menu tasks. This leads to 6,400 total completed menus. If 20 price levels are defined for a particular menu item, this means that each price level will have been seen about $6,400/20 = 320$ times. While this wouldn't pose any problem for linear or log-linear estimation, it can lead to troubles for part-worth (dummy-coded) functions. There would only be 320 support points for each discrete price level, and the aim of part-worth functions is to independently estimate the utility at each discrete price point. Our general rule of thumb is that each level of independent variable (attribute) in a choice experiment occur at *bare minimum* 500 times, and preferably 1000 times. For a study involving 6,400 total completed menu tasks in the data file, this means that one should probably not go much beyond about 6 levels for a price attribute.

Since this section has been pretty lengthy, we summarize the main points:

- We generally recommend using about 5 levels of price per menu item
- Symmetry: the number of price levels per menu item should be the same across items in the design
- Price levels don't need to be equidistant
- The $[\#respondents \times \#menu_tasks]/price_levels$ should be around 1000. In other words, each price point should appear about 1000x across all respondents and menu tasks.

3.5

Single-Block, Multiple-Block and Randomized Plans

We've already discussed how for traditional card-sort conjoint, typically just one version of the cards was printed, and all respondents got the same set of cards. With CBC, often there were more tasks in the design than any one respondent could answer. If 72 total tasks were required, these could be split into 6 blocks of 12 questions. Each respondent would be randomly assigned to complete one of the blocks.

Over the years, researchers have begun to favor larger designs (especially for CBC), with greater numbers of blocks. In the limit, each respondent can receive his/her own unique version (block) of the plan. In the Sawtooth Software literature, these are referred to as "randomized" plans. That is to say that each respondent is randomly assigned to receive one of many available versions of the questionnaire, where each version has been designed carefully according to standard design principles such as level balance and orthogonality.

Having multiple blocks not only can increase design efficiency, but it reduces order and context effects, which can contaminate utility estimation.

Although we've spent a good deal of space discussing carefully designed plans according to catalogues or specialized design software, it turns out that for computerized interviewing, and relatively large sample sizes (both likely for MBC projects), *randomized plans are convenient and quite robust.*

Sawtooth Software's CBC software can generate designs that you employ in MBC, and the fast-food example described earlier (Figure 1.6) was designed using CBC's Shortcut design methodology. However, purely random designs (such as using "list building" in Lighthouse Studio or other randomizing methodologies provided by other computer interviewing software) are quite suitable for aggregate analysis such as aggregate logit. It is best to try to control for level balance when using random designs, but if the random number generator is good, there will be quite reasonable level balance across hundreds of respondents and thousands of tasks. Exact level balance is not required.

Although some researchers might feel that using randomized designs with no explicit control for level balance or orthogonality seems haphazard and unscientific, it turns out that such designs are often about 80% as efficient (when considering aggregate analysis) as the best designed plans available. For some practitioners, randomized plans are especially alluring because they are easy to implement and can help hold development costs down. But, if individual-level analysis via HB is to be used, we recommend more careful design procedures that lead to excellent level balance and near-orthogonality within each respondent's tasks.

Our preference is to use CBC software with its controlled randomization strategies to design MBC questionnaires, and to field the studies using Lighthouse Studio and the "Free Format" questions with Javascript verification.

Individual Item Price Variation or Lock-Step Category Price Variation

In the example shown earlier in this documentation (Figure 1.6), we varied the price of each menu item independently. Drinks and fries posed a special problem, since we wanted to measure the price sensitivity separately for the small, medium, and large varieties. We imposed some modest prohibitions in the design, to avoid displaying illogical price comparisons (e.g. large fries costing less than medium fries). However, another approach to investigating respondent choices in the face of changing prices is to vary the price of all items within the category in a correlated (lock-step) fashion.

For example, rather than draw a random price variation for each size of fries, we could draw a single variation to apply to *all* fries. For example, we could establish average prices for three different sizes of fries:

Average Prices:

\$1.09 Small fries
\$1.59 Medium fries
\$1.99 Large fries

Then, for a given choice task, we might reduce *all prices* by \$0.20:

Prices all decreased by \$0.20:

\$0.89 Small fries
\$1.39 Medium fries
\$1.79 Large fries

Rather than having three separate independent variables that control prices for the three sizes of fries, we can use a single independent variable to control all prices for fries. That variable might take on 5 levels: -\$0.40, -\$0.20, +\$0.00, +\$0.20, +\$0.40.

We still model the choice of each size of fries as a function of price changes, but price changes overall for the category rather than for each item independently. In general, quantity demanded of fries (net as a category) should increase as the category price decreases for fries. But, we shouldn't necessarily expect the quantity demanded of *each item* within the category to increase as the category's price decreases. For example, as the category price for fries uniformly decreases for all three sizes, respondents might shift away from the smaller fries in favor of medium and large fries. So, small fries could show a *decrease* in quantity demanded as category price for fries decreases. Conversely, as category price increases, small fries might show an *increase* in quantity demanded, as people shift away from more expensive larger fries.

Each area of the restaurant menu shown in Figure 1.6 could be designed using category price variations rather than individual-item price variations: Sandwiches, Fries, Drinks, Salads, Healthy Sides, and Desserts.

Which approach should you take: individual-item variation or lock-step category pricing variation? It depends chiefly on the aims of the research, but we'd recommend trying to use lock-step category pricing variation whenever it makes sense. Lock-step category price variation means

fewer independent variables, and can be less demanding on the model in terms of numbers of parameters to estimate. If the client is just interested in the choice of items due to overall category price changes rather than in estimating the independent price sensitivity of each item within the category, then it would favor the lock-step category pricing approach. In many real-world situations, the store frequently increases or decreases prices for all or most items in a category, so lock-step category price changes may actually be more realistic.

3.6 Premium Pricing and Bundle Pricing

One of the practical hurdles that researchers will probably face with MBC questionnaires is the restriction that some prices on the menu must always be higher than other prices. For example, consider basic and premium items on the menu, and the premium item ("Ultra") should always be 50% more expensive than the basic item:

- Basic \$100
- Ultra \$150

Perhaps the Basic price level variations are \$80, \$100, and \$120. In this case, we only include that 3-level factor in the design. The price of ultra is simply computed as 1.5x the Basic price (this is the same notion as lock-step category pricing). With such a design, the likelihood of choosing Ultra will be expected to be a function of the price of the Basic package.

Sometimes, the design needs to be a bit more complex, such as Ultra being \$20, \$40, or \$60 more expensive than Basic. For that case, we would have two factors in the design to control the prices for Basic and Ultra:

<u>1) Basic Price</u>	<u>2) Premium for Ultra</u>
\$80	\$20
\$100	\$40
\$120	\$60

So, if a design card features \$80 for the Basic Price and \$40 as the premium to apply when computing Ultra's price, the display is:

- Basic \$80
- Ultra \$120

You simply add the Basic price plus the premium to compute Ultra's price on the menu. During analysis, the likelihood of picking Ultra is modeled as a function of the Basic price and separately the premium price.

Bundle pricing is also handled using the same trick. Consider the fast-food example introduced earlier.

Menu Scenario #1: Please imagine you pulled into a fast-food restaurant to order dinner for <u>just yourself</u> . If this were the menu, what (if anything) would you purchase?		
<input type="radio"/> Deluxe Hamburger Value Meal -Deluxe Hamburger -Medium fries -Medium drink \$3.99	<input type="radio"/> Chicken Sandwich Value Meal -Chicken Sandwich -Medium fries -Medium drink \$5.59	<input type="radio"/> Fish Sandwich Value Meal -Fish Sandwich -Medium fries -Medium drink \$3.99
(Only order sandwiches, fries or drinks from this area if you did not pick a value meal above.) Sandwiches: <input type="radio"/> Deluxe Hamburger \$1.99 <input type="radio"/> Chicken Sandwich \$3.59 <input type="radio"/> Fish Sandwich \$1.99 Fries: <input type="radio"/> Small \$0.79 <input type="radio"/> Medium \$1.49 <input type="radio"/> Large \$1.69 Drinks: <input type="radio"/> Small \$0.99 <input type="radio"/> Medium \$1.69 <input type="radio"/> Large \$2.19		Salads: <input type="radio"/> Cobb dinner salad \$4.79 <input type="radio"/> Grilled chicken salad \$4.39 Healthy Sides: <input type="radio"/> Carrots/Celery with Ranch dressing \$1.19 <input type="radio"/> Apple slices/Grapes with dipping sauce \$0.99 Desserts: <input type="radio"/> Apple/Cherry/Berry pie \$0.99 <input type="radio"/> Cookies \$1.19
<input type="radio"/> I wouldn't buy anything from this menu. <input type="radio"/> I'd drive to a different restaurant, or do something else for dinner.		

Figure 2.2

In this example, we wanted to make sure that the value meals shown at the top of the menu were always priced lower than the sum of their parts if ordered separately from the *a la carte* portion of the menu. Furthermore, we wanted to test respondents' sensitivity to differing amounts of discounts. We tested the following value meal bundling discounts: \$0.38, \$0.78, \$1.18, and \$1.58, which were treated as an additional 4-level factor in the design. So, to determine the price for the Value Meal, we added the prices shown on the current menu for the components making up the value meal, minus one of the bundling discounts. In the example above, the Deluxe Value Meal price was computed as follows:

Deluxe Hamburger	\$1.99
+ Medium fries	\$1.49
+ Medium drink	\$1.69
- Bundling discount	<u>\$1.18</u>
Total:	\$3.99

When we modeled the likelihood of choosing the Deluxe Hamburger Value Meal, we modeled it as a function of the prices of its separate components (hamburger price, medium fries price, medium drink price), as well as its bundling discount.

3.7 Design Testing

Readers accustomed to using Sawtooth Software's CBC program know how easy it is to use CBC's Test Design procedure to generate random respondent data and compute the standard errors of logit utilities via aggregate logit. MBC doesn't provide an advanced test capability. But, you can create your .csv datafiles using your experimental design, and use Excel's "Randbetween (minvalue,maxvalue)" function to generate random respondent answers to the menu tasks. Then, you submit the data to MBC software's aggregate logit routine to examine the size of the standard errors of the utility parameters. For example, you can examine the size of the standard errors for designs without prohibitions compared to a design you are working on that requires some prohibitions. Our experience from standard CBC studies is that utility parameters from random response data should have standard errors of .05 or smaller. (The standard errors are of course related to the magnitude of the independent variables in the X matrix. The way we scale the X matrix in MBC gives each column comparable scaling, allowing us to directly compare the magnitude of standard errors using these norms.)

For MBC studies, if you find that you must impose prohibitions in the design, or if you have other design elements that might degrade design efficiency, you certainly should test your design. Failure to adequately test your design may result in unusable data. There are enough moving parts and complication to some MBC studies, that such details cannot be overlooked.

When generating random-responders for test data, it's helpful to consider the choice likelihoods. If an item is likely to be chosen about 5% of the time, then it would overstate the precision of the parameters involved (such as that item's prices) if choices were assigned with about 50% likelihood.

4 Preparing Data Files for MBC

4.1 Preparing Data Files for MBC

We have made it very easy to prepare data files for MBC software. We use the .csv text format, which may be saved from Excel and by most software that is used for collecting and processing data. The top row contains labels, but all data in the remaining rows must be numeric only (or missing/blank).

Consider a very simple menu where respondents can choose among three options, each with varying prices:

- Hamburger \$1.99
- Fries \$1.39
- Drink \$1.19

To code the data file, a leading row of labels should first be provided (shown in row 1 in Figure 3.1 below). The respondent ID (must be unique) should be found in the first column (column A), with independent variables and dependent variables in the other columns (columns B through G). Each menu task is formatted *as a single row*. Thus, a respondent with 5 menu tasks will span 5 rows of the data file. An example is given below (the first 8 rows, as might be displayed by Excel):

	A	B	C	D	E	F	G
1	CaseID	Pr_Ham	Pr_Fries	Pr_Drink	Ch_Ham	Ch_Fries	Ch_Drink
2	1001	3	1	2	2	1	2
3	1001	1	2	3	1	1	2
4	1001	5	2	1	2	1	2
5	1001	4	4	2	1	1	2
6	1001	1	3	5	1	1	2
7	1002	2	2	2	2	2	1
8	1002	1	4	3	2	2	1

Figure 3.1 (Example Data File as Displayed in Excel)

The independent variables in this data file (columns B through D) are the prices for each of the items on the menu (**Pr_Ham**, **Pr_Fries**, and **Pr_Drink**). In this example, we have varied each menu item's price across 5 alternative-specific prices. The actual prices shown for the five possible prices of hamburger (variable **Pr_Ham**) might have been:

<u>Level#</u>	<u>Price Shown</u>
1	\$1.59
2	\$1.79
3	\$1.99
4	\$2.19
5	\$2.39

But, we code the prices shown as integers 1-5 in the data file (*you should code most of your independent variables as consecutive integers beginning with 1*). This gives MBC maximum

flexibility to use this variable as a categorical (dummy-coded part-worth function) or as a linear term (when you assign values to the categories of each independent variable).

When this spreadsheet is saved to a .csv format, it looks like the following when opened with a text editor like Notepad:

```
CaseID,Pr_Ham,Pr_Fries,Pr_Drink,Ch_Ham,Ch_Fries,Ch_Drink
1001,3,1,2,2,1,2
1001,2,5,3,2,1,2
1001,5,2,1,2,1,2
1001,4,4,2,1,1,2
1001,1,3,5,1,1,2
1002,2,2,4,2,2,1
1002,1,4,4,2,2,1
```

It's easy to open your .csv-formatted data file in MBC software. From the *Data Files* tab, you simply browse to your .csv file.

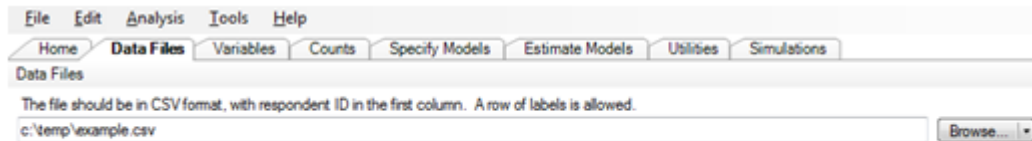


Figure 3.2 (Selecting Data File from Data Files Tab)

4.2 Labels File

After you have specified which .csv file to use for your project, the next step is to provide labels for the levels (variable categories) used in the study, as well as values associated with each independent variable level if you want to specify models involving linear or log-linear terms. The *Variables* tab is where you do this, and this dialog supports cutting and pasting from programs such as Word or Excel, so you don't have to re-type labels and values that you have written elsewhere:

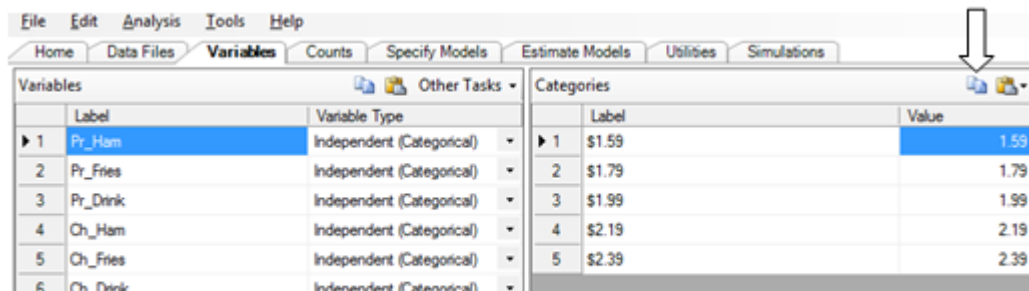


Figure 3.3 (Establishing Variable Labels and Values on Variables Tab)

On the *Variables* tab, you specify the labels used for the categories of each of the independent and dependent variables in your study. In Figure 3.3, we show how the labels have been specified for each of the five levels of hamburger price (**Pr_Ham**), as well as the values associated with each label. The labels are used in MBC's reports and also the market simulator (so we suggest concise labels).

Because there are many labels and values associated with variables in MBC projects, it is helpful to define these in a separate **my_labels.csv** file, using a program like Excel. You may **Import Category Labels and Values from File** by clicking the icon indicated by the arrow in Figure 3.3. The .csv file containing labels must have four columns: attribute#, level#, label, and value (see Figure 3.4).

Figure 3.4 shows an example labels file, as displayed in Excel.

	A	B	C	D
1	1	1	\$1,500	1500
2	1	2	\$1,750	1750
3	1	3	\$2,000	2000
4	1	4	\$2,500	2500
5	2	1	\$500	500
6	2	2	\$700	700
7	2	3	\$900	900
8	2	4	\$1,200	1200
9	3	1	\$300	300
10	3	2	\$400	400
11	3	3	\$500	500
12	3	4	\$600	600
13	4	1	\$600	600
14	4	2	\$800	800
15	4	3	\$1,000	1000
16	4	4	\$1,200	1200
17	5	1	\$150	150
18	5	2	\$200	200
19	5	3	\$250	250
20	5	4	\$350	350

Figure 3.4 (Example Labels File as Displayed by Excel)

For this dataset, after the independent variables (**Pr_Ham**, **Pr_Fries**, **Pr_Drink**) follow the dependent variables: the choices of items on the menu (**Ch_Ham**, **Ch_Fries**, **Ch_Drink**). (It is not a requirement that independent variables be provided prior to dependent variables; they can come in any order.)

Notice that the final column (values) contains prices on the scale in which they were shown to respondents (e.g. 1500, 1750, etc.). If you specify that these independent variables are Categorical, MBC software will automatically [transform](#) these to have values within the design matrix that are zero-centered with a range of 1 (to provide faster convergence during estimation).

In this data file, we have coded menu choices (dependent variables) as 1=chosen, 2=not chosen (***all dependent variables should be consecutive integers, starting with 1***). Therefore, in Row #2 of the data file (respondent 1001's first menu task, as shown earlier), this respondent did not choose a hamburger (**Ch_Ham=2**) or drink (**Ch_Drink=2**), but did choose fries (**Ch_Fries=1**). For the second choice task seen by this respondent (formatted on row 3), the respondent chose both hamburger and French fries from the menu.

Some menus will have dependent variables that take on more than just two levels indicating on/off. The responses are coded as consecutive integers starting with 1.

4.3 Selecting Independent and Dependent Variables

From the *Variables* tab, you need to indicate which variables are dependent variables using the *Variable Type* column drop-down control. Figure 3.5 shows the three dependent variables (**Ch_Ham**, **Ch_Fries**, **Ch_Drink**) with *Variable Type* set to "Dependent." Notice also that we can specify which (if any) category of the dependent variable indicates "not selected" (off state), which makes the later counting report more streamlined and easy to read. For this experiment, all dependent variables have an off state, so we indicate for each dependent variable that code=2 indicates the off state (not selected checkbox).

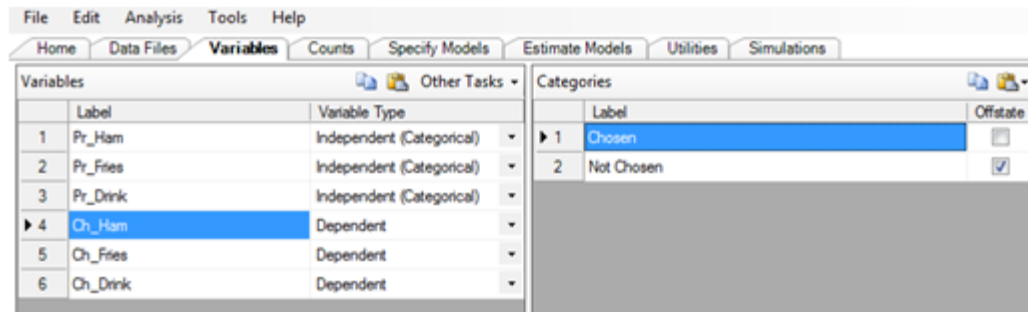


Figure 3.5 (Marking Dependent Variables and Indicating any Off state)

Other data may be provided as well in the file, such as continuous independent variables that you treat as "user-specified" for MBC to take directly into the independent variable matrix as a single column without any further modification. These variables may include decimal places of precision, and include positive and negative numbers.

4.4 Missing Values

Missing Tasks

If a respondent did not complete all tasks, you may simply delete any missing task(s) (rows) from the data file. Each respondent does not need to have the same number of tasks (rows) present in the data file.

Missing Dependent Variables

When dependent variables are conditional on the state of other dependent or independent variables, this leads to missing data. For example, if you cannot pick Dependent Variable 2 unless Dependent Variable 1 was chosen, then Dependent Variable 2 will be missing whenever Dependent Variable 1 is not chosen. A dependent variable may also be missing due to the state of a triggering independent variable. For example, if an availability variable in the design triggers whether an item is available or not, then the choice of that item will be missing whenever the availability variable's value is false.

Missing values appear as blank fields when opened with a program like Excel. If viewed with a text editor, the missing values are indicated by consecutive delimiters in the .csv file.

Missing Independent Variables

Independent variables can be missing due to a conditional dependent variable, as described above. Independent variables can also be missing due to the state of a triggering independent variable. For example, if an availability variable in the design triggers whether an item is available or not, then the price of that item will be missing whenever the availability variable's value is false. The missing value is indicated as a blank field.

There are other cases involving independent variables that are only applicable to specific alternatives in the design matrix (alternative-specific effects). In these cases, you may specify a not-applicable independent variable with a code of zero. The zero carries forward into the design matrix for parameter estimation.

4.5 Segmentation Variables and Weights

If you have segmentation variables and weights, these should be provided in a separate "demographics" .csv file, again with CaseID as the first column. The CaseIDs must match the CaseIDs provided in the main data file. You select the demographics file from the *Data Files* tab:

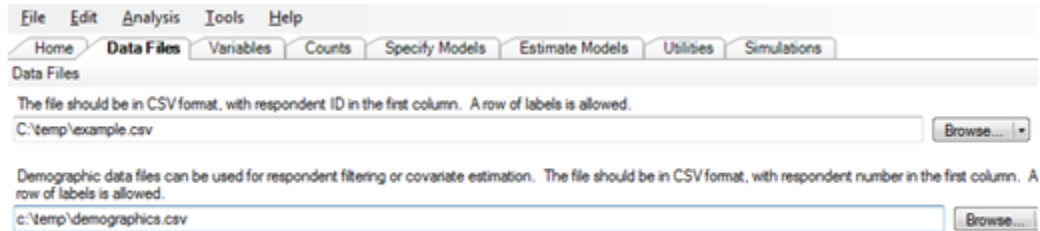


Figure 3.6 (Selecting Demographics File from Data Files tab)

Missing values are not permitted in the demographics file. If you have a missing value, you should set it to a numeric code distinguishable from the other numeric values in the file.

Weights are applied during counts and aggregate logit.

Weights are not applied during HB analysis, but rather are applied post hoc during market simulations, to differentially weight respondents' predicted choices.

5 Counting Analysis

5.1 Introduction

Although it is not requisite for MBC projects, counting analysis can help you learn about important relationships in your data *before* you spend time building logit or HB models. After investigation via counting analysis, you will be better prepared to build well-specified models: effective combinations of independent variables with appropriate functional forms, and accurate market simulators.

In counting analysis, we compare how many times something was *available to be chosen* to how many times it was *actually chosen*. If a hamburger at \$1.99 was available to be chosen 1000 times (across all respondents and menu tasks), and it was selected 100 of those times, its Counts percentage is $100/1000 = 10\%$.

Randomized designs, as described previously, provide robust data for counting analysis (except when prohibitions would bias the results). For orthogonal or near-orthogonal designed plans that are also level balanced, larger plans featuring multiple blocks (versions) also provide data suitable for counting analysis.

Counting analysis provides many benefits:

- For each choice option on the menu, you can develop hypotheses regarding which independent variables are the strongest drivers of those choices.
- Counting analysis allows you to consider whether the effects of continuous variables (such as price) might be best specified as linear, curvilinear (log-linear), or non-linear (part-worth function).
- You can observe whether continuous variables indicate positive or negative slope relationships with choice. Own-price effects are typically negative and cross-effects for substitutable items are typically positive.
- For typical menu situations, the independent variables are prices for the menu items. But independent variables can also be nominal/categorical in nature, such as the availability of an item on the menu (if the item is sometimes on or off the menu), different orderings or layouts for the menu, or variables involving the choice context, such as "purchasing for a child vs. purchasing for oneself." Counting analysis lets you examine the effect of these nominal or categorical independent variables on menu choices.
- You can examine the most commonly selected *combinations* of choices across all tasks in your MBC questionnaire. For example, the most commonly-picked menu bundle for a fast-food study might be the combination: Hamburger + Medium Fries + Medium Drink.
- Counting analysis can provide an excellent cross-check against the final market simulator output, to give you assurance that there aren't any glaring errors in your simulator. If you specify a market simulation wherein all items on the menu are held at their average price (or "typical" level, if considering a non-price independent variable), the resulting shares of choice should closely resemble the marginal counts proportions. When you change an item's price from its lowest to highest values (and all other items on the menu are held at the constant average price), the resulting demand function for that item should closely resemble the own-price demand function suggested by counts. Cross-elasticity price functions may also be compared for simulated results vs. counts. If you have made a significant error in building your models, comparisons to counts most times will point out the problem.

- You can investigate whether items on the menu appear to be complements or substitutes via counting analysis. You can compare the actual joint likelihood of selection to the expected joint likelihood.

In the next sections, we will describe some of these benefits in more detail.

5.2 Investigating Significant Predictors of Choice

Counting analysis helps you investigate which independent variables (such as price variations) seem to be significant drivers of menu choices. For example, the choice of hamburger on a fast-food restaurant menu is almost certainly related to the price charged for that hamburger (an own-price effect). And, the price of the chicken sandwich may also have an effect on the choice likelihood for hamburger (a cross-effect). Counting analysis allows you to investigate the strength of such relationships using a Chi-Square statistic.

It would seem to be covering all bases if we built models where choice of hamburger was a function of its own desirability, its own price, and *all* other menu prices or other independent variables. Such a model could capture *all* potential own- and cross-effects (a "kitchen sink" model). This is in the spirit of *mother logit* models that were popular thirty or forty years ago under aggregate analysis, such as aggregate logit.

While such models are conceptually complete, they are probably not the best for the MBC models this software estimates via logit and HB, for three main reasons:

- Such models include a lot of parameters to be estimated (many independent variables) and might take a very long time to run under Hierarchical Bayes (HB) estimation. In addition to long runtimes, the ratio of number of parameters to be estimated (independent variables) to information provided (choice tasks) at the individual level may be relatively high. HB models that estimate individual-level models do best when the data are not especially sparse.
- Models that include *all* possible effects, whether significant or not, run the risk of overfitting. While the fit of the model to the choices in the calibration tasks (the tasks used to estimate models) may slightly increase with the inclusion of non-meaningful independent variables, the ability to predict new choice scenarios (outside the calibration tasks) often degrades due to parameter weights taking advantage of within-sample random relationships.
- When market simulator models use all possible own- and cross-effects, these often demonstrate noisy reversals that can be unsettling. For example, lowering the price for carrots on the menu might *ever so slightly* increase the simulated likelihood of purchasing a hamburger, when such relationships are non-significant and lack face validity. While it may be explained that such weak relationships are non-significant and due to random error, many people prefer tidy simulation results that don't reflect such irregularities. Pruning non-significant cross-effects will lead to cleaner simulators with fewer oddities to explain to clients.

Because of these reasons, we encourage you to prune your MBC models by dropping non-significant independent variables. Most own-effects are significant drivers of choice and should be included in the model. But, many cross-effects are not significant. In our experience, often 75% or more of potential cross-effects will turn out to be non-significant.

For each potential menu selection (dependent variable), significant *potential* drivers of that choice may be identified via counting analysis. For example, we have been discussing a fast-food restaurant example. We expect that the choice of hamburger from the *a la carte* menu is certainly related to its price. Counting analysis allows us to examine the strength of that effect (in terms of Chi-Square), as well as investigate other potential drivers.

Most readers are probably familiar with counting analysis from Sawtooth Software's CBC programs, and the counting analysis we employ is very similar for MBC. Counting analysis simply computes the percent of times an item on the menu was chosen when specific levels (indications or prices) of an independent variable were given. For example, we can count the percent of times that the hamburger was chosen from the *a la carte* menu when it was offered at different discrete price levels.

The *Counts* tab within the MBC software includes a dialog where you can request counts for each dependent variable at a time. In Figure 4.1 we have selected to run a counting analysis that analyzes the *Effect of variables on choices* of the dependent variable **Sandwich_Choice**.

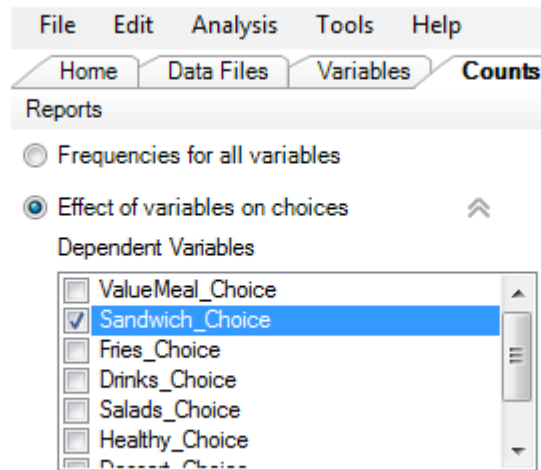


Figure 4.1 (Examining Effect of Independent Variables on Sandwich Choices)

When we generate the report, we can look at (for example) the effect of changing prices of the hamburger (**Burger_Price**) on the likelihood of choosing the hamburger.

Dependent variable: Sandwich_Choice, Hamburger	
Choice Probability	9.99 %

Figure 4.2 (Counts Report for Probability of Burger Choice)

Overall, hamburgers were chosen 9.99% of the time across all tasks in the MBC questionnaire. But, the price for the hamburgers varied according to a randomized experimental design. We can tabulate the percent of times hamburger was chosen when it was shown at each price point in the design.

Choice probability by categories of Burger_Price	
\$1.99	12.33
\$2.29	9.69
\$2.59	8.74
\$2.89	9.18
Relationship Chi-Square	10.66
Degrees of freedom	3
P-value	0.01
Non-Linearity Chi-Square	2.86
Degrees of freedom	2
P-value	0.24
Suggested Functional Form	Log-Linear

We see that as the price of the hamburger increases, its likelihood of choice generally decreases. However, there is a slight (non-significant) "reversal" for the probabilities at \$2.59 (8.75%) and \$2.89 (9.18%). The probabilities suggest respondents prefer Burgers at \$2.89 over \$2.59 (which is probably not true). Counting analysis is not quite as accurate as parameters estimated via logit and HB analysis, so perhaps once utilities are estimated, the reversal will go away. However, if it persists, fitting a linear, log-linear or part-worth function (with utility constraints) could resolve the reversal if you felt it was necessary to eliminate it.

Note that we have computed a "Relationship Chi-Square" statistic with accompanying likelihood (p-value) that quantifies the likelihood that the strength of this relationship as observed would occur by chance (0.01). If the choice likelihood of hamburger was essentially unchanged when it was offered at different price levels, the Relationship Chi-Square statistic would be very small, and the p-value would be very high (near 1.0). Many researchers take p-values of 0.05 (95% confidence) or even 0.10 (90% confidence) as indications that an independent variable should be included in a model. We would recommend that you not be overly selective at this stage of the analysis. Any independent variable with a p-value of about 0.20 or less (80% confidence or higher) might be included in our logit and HB models unless there is clearly no reasonable behavioral explanation for that relationship.

MBC reports the count proportions for all discrete levels of the independent variables predicting each category of each dependent variable. Because this can be a lot of data to digest, MBC software provides a summary table listing the strength of each independent variable relationship (in terms of the p-values) on choice for each menu item.

Dependent variable: Sandwich_Choice, Hamburger			
Independent Variable	Relationship P-Value	Non-linearity P-Value	Suggested Functional Form
IV1 Bundle Discount	0.00	0.80	Linear
IV2 Deluxe Burger Price	0.01	0.24	Log-Linear
IV3 Chicken Sandwich Price	0.19	0.13	Log-Linear
IV4 Fish Sandwich Price	0.18	0.18	Linear
IV5 Small Fries Price	0.34	0.21	Linear
IV6 Medium Fries Price	0.19	0.95	Linear
IV7 Large Fries Price	0.04	0.62	Linear
IV8 Small Drink Price	0.87	0.90	Log-Linear
IV9 Medium Drink Price	0.02	0.15	Log-Linear
IV10 Large Drink Price	0.63	0.42	Log-Linear
IV11 Cobb Salad Price	0.40	0.32	Linear
IV12 Grilled Chicken Salad Price	0.13	0.25	Linear
IV13 Carrot Price	0.88	0.83	Linear
IV14 Fruit Slices	0.06	0.18	Log-Linear
IV15 Pie Price	0.90	0.79	Log-Linear
IV16 Cookie Price	0.95	0.83	Linear

This allows you to scan the summary table and quickly note which independent variables are potential drivers of choice for each level of each dependent variable. (It's easiest to click **Show results in popup button** or to cut-and-paste these summary tables into another document or spreadsheet for later reference as you build your models.) When you eventually build models that predict each dependent variable, you'll want to refer to this summary table as a possible guide for which variables to include, and recommendations for appropriate functional forms (linear, log-linear, or part-worth).

Notes: Counts results provide a guide for understanding relationships in your data. However, many researchers approach experimental design and subsequent model building with a priori beliefs regarding which variables to include, and model specification. It is appropriate to rely on the strength of the experimental design approach and to apply those prior beliefs when building your models and simulating choices, even if many of the effects seem not to have statistically significant relationships. In fact, such an approach sets you on more formally correct ground than the post hoc approach of looking for statistically significant effects (which requires you to select a degree of confidence) and including all those that seem to meet that threshold of confidence.

If using Combinatorial Coding of dependent variables (a recommended approach), the combinatorial dependent variable often takes on a dozen or more categorical states. Many of those states involve very small bases, leading to imprecise counting results and lengthy Counts reports to account for each of often dozens of categories. It is better to approach model building for combinatorial dependent variables using a priori beliefs combined with knowledge learned from Counting analysis prior to combining multiple (e.g. binary) dependent variables into combinatorial outcomes.

5.3 Linear or Non-Linear Relationships

Many econometric models used in academia and in practice assume linear or log-linear functions for price or other continuous variables, such as shown below.

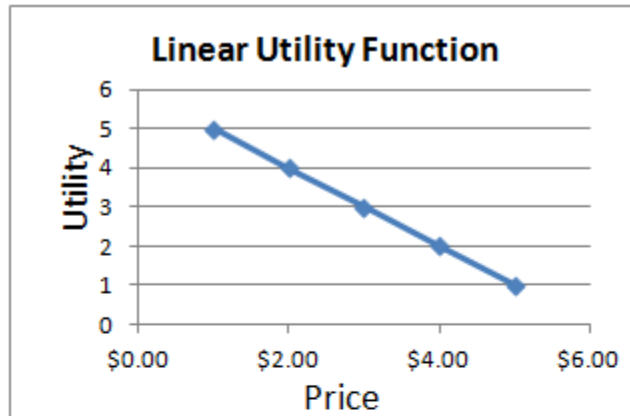


Figure 4.5 (Linear Utility Function)

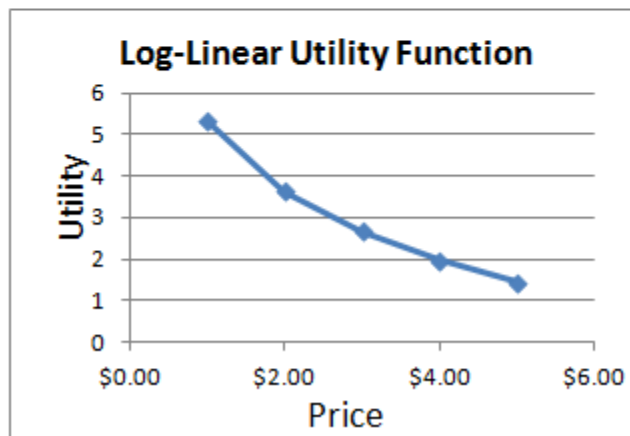


Figure 4.6 (Log-Linear Utility Function)

In MBC, we accept the possibility that price functions are not necessarily so smooth and predictable as shown in Figures 4.5 and 4.6. We suggest you be on the lookout for non-linear relationships that may not be captured well using linear or log-linear specifications—and counting analysis provides one way to investigate this. It is quite possible that respondents have certain psychological thresholds that may lead to quite non-linear effects of price on choice for certain menu items. ***If you were to specify a linear relationship for a price variable and the true relationship is decidedly non-linear, you may just have smoothed away an important finding of the research and hindered the accuracy of your simulation model.*** MBC software lets you customize for each independent variable whether its effect on specific menu choices is linear, log-linear, or non-linear (part-worth).

Counting analysis helps you identify *potential* non-linear relationships (remember that counting analysis isn't as accurate as estimating the effects via logit or HB). If you design your questionnaire so that price variations take on a reasonably small number of discrete (rather than continuous)

price variations (such as five levels of price: \$10, \$12.50, \$15, \$17.50, and \$20), you put yourself in a good position to investigate non-linear effects using counting analysis and later to more formally model non-linear effects (via dummy-coded part-worths) in subsequent logit and HB analysis.

In general, we suggest you investigate via counting analysis whether *own-price* effects are linear or non-linear. Cross-effects are typically less strong and may more generally be captured (without much loss in model accuracy) as linear or log-linear effects.

For example, we can examine the effect of hamburger price on choice of hamburger (a strongly significant own-effect). We've plotted the relationship using Excel:

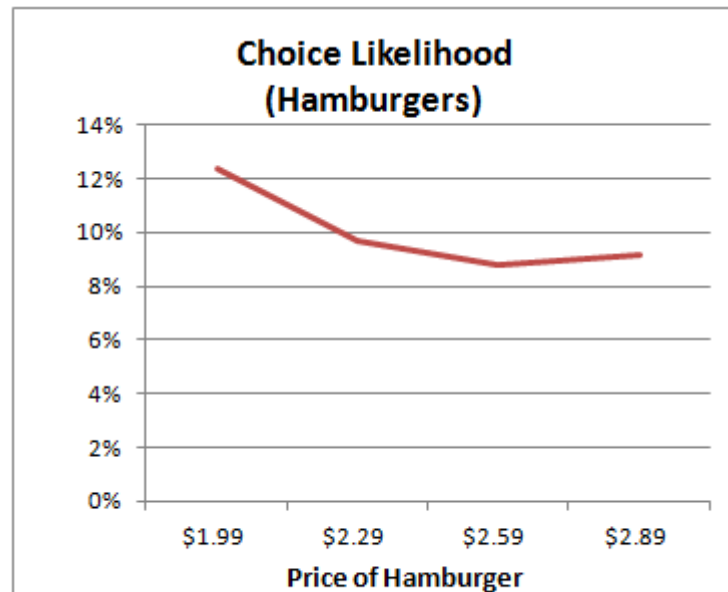


Figure 4.7 (Effect of Price of Hamburger on Hamburger Choice)

It can be difficult to judge whether there is enough evidence that a relationship is non-linear or linear just by visually inspecting such plots. The preference function might zigzag or look to have distinct bends and curves, when really those bends are due to noise resulting from sampling error.

If we assume independence of choice tasks, we can compute the standard error of each proportion and also a confidence interval. If each price point in the chart is supported by an n of about 1000 (where n is the number of choice tasks shown with that price across the sample), we know that the 95% confidence interval is roughly $\pm 3\%$ or less for each count proportion. Considering the margins of error helps us intuit ("eyeball") whether the non-linear patterns we see could be due to chance variations in the proportions or are probably repeatable and significant. But, MBC's counts analysis provides a more formal Chi-Square test to assist you in deciding whether there is enough evidence in the data to justify a non-linear function. In the counts report, a "[Non-Linearity Chi-Square](#)" test is shown, together with a p-value (see Figure 4.3). (Details of this test are given in the Appendix.) If the non-linearity p-value is 0.05 or lower, it indicates 95% confidence or better that the relationship is non-linear (a part-worth function). The Chi-Square test for this particular relationship leads to a recommendation that a log-linear function fits the data better than linear, but that a part-worth function (dummy-coded parameters) is not justified by the data.

Some researchers may plot counts curves and see a relationship that looks decidedly non-linear, yet the Chi-square test for non-linearity may recommend against using a part-worth function. That is because there needs to be a large enough count size (enough respondent choices; enough evidence) to justify that the frequencies observed are different enough from the frequencies expected from a linear function.

For linear or log-linear functions, just one parameter is estimated to capture the relationship. *Reducing the number of parameters to be estimated is beneficial*, especially for HB analysis. The log-linear model is done by transforming the quantitative variable (such as price) by the natural log in the design matrix (note: price values must be positive to support log-linear functions as the log of a negative value is undefined). For non-linear models that are not fit well using either a linear or log-linear function, we can employ dummy-coding (part-worth function). With dummy-coding, a price variable taking on 5 discrete levels is modeled using 5-1 or 4 parameters.

5.4 Positive or Negative Slope Relationships

Counting analysis can help you identify whether an independent variable (such as price for a menu item) has a positive or negative effect on the choice of a menu item. The effect of the price of a menu item on its own choice likelihood is expected to be negative: as price increases, the choice likelihood decreases. For items viewed as *substitutes*, the price of one item is *positively* related to the choice likelihood of the other item. For *complementary* items, the price of one item is *negatively* related to the other item.

Sometimes, you or your client may have some theories regarding the positive or negative effect of variables such as prices on menu choices. The counting module can help you assess whether the data seem to conform to those assumptions. Of course, you should more formally test the appropriate signs of utility relationships using the aggregate logit module. Formally testing the appropriate signs would be important prior to imposing utility (monotonicity) constraints during utility estimation. You wouldn't want to impose a utility constraint with a sign that contradicts a strongly significant sign relationship.

5.5 Comparing Counts to Final Market Simulations

In an earlier section of this manual, we stressed how important counts are in helping to diagnose glaring problems in the market simulator. MBC market simulators can become quite complex, involving multiple logit-based models acting in concert to predict choice likelihoods for menu items given specific prices (or other specific realizations of independent variables). Before delivering such what-if simulator tools to a client, you should do a healthy amount of testing and checking to ensure that it is working properly. Comparing simulations to counts results can help you assess systematically whether there is a significant problem with the market simulator. Of course, holdout choice tasks are also valuable in this regard. But, with MBC studies, *counts are probably even more valuable than holdouts* for assessing the quality of the market simulator and diagnosing specific problem areas. A holdout gives you a snapshot of how respondents would react to a *specific* set of prices (or realization of other independent variables) on the menu. Counts results capture how respondents reacted to the *full range of price changes*, in terms of the average choice likelihood of menu items, and the own-effects and cross-effects of menu prices on choice. Thus, counts results provide a more complete reference for testing the output of market simulators.

5.6

Which Combinations of Items Were Selected Most Often?

Often, researchers are interested in which specific combinations of menu items were selected most often. Consider a menu with 12 items that can be checked or not checked. There are $2^{12} = 4,096$ possible combinations of checks that may be made. Combinatorial counting analysis reports the top n combinations that were selected, and the percent of total choices accounted for by each combination.

From the **Counts** tab, you can select to perform a counting analysis on the *Most frequently chosen combinations*.

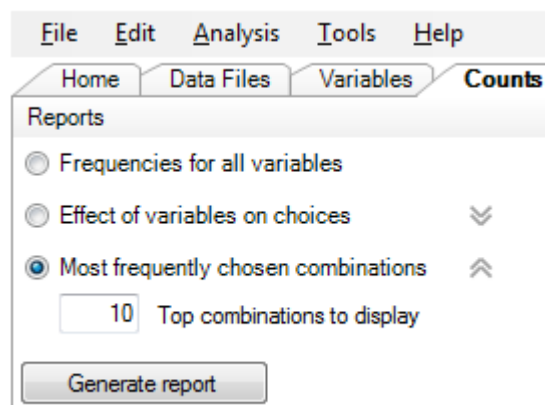


Figure 4.8 (Counts Reporting Options)

You may also specify how many top combinations to display (10 are shown by default). When you click **Generate report**, MBC first reports the one-way choice likelihoods of items:

Menu Combinations Counts Report		
Respondents Included		681
Total Number of Choice Tasks		5448
Average Number of Choice Tasks per Respondent		8.00
Total Number of Choice Tasks (weighted)		5448.00
Average Number of Choice Tasks (weighted)		8.00
Tasks Included		All
Respondents Weighted by		None
Number of Results to Display		10
Weighted Respondents		681.00
One-Way Probabilities of Choice		
Dependent Variable	Category	Probability
DV1 Value Meal Choice	Deluxe Hamburger Value Meal Chosen	27.42
	Chicken Sandwich Value Meal Chosen	10.99
	Fish Sandwich Value Meal Chosen	10.65
DV2 Sandwich Choice	Deluxe Hamburger Chosen	9.99
	Chicken Sandwich Chosen	7.38
	Fish Sandwich Chosen	5.62

Figure 4.9 (One-Way Counts Report)

For example, the Hamburger Meal was chosen 27.42% of the time.

MBC also reports two-way choice likelihoods of items (how often each combination of two levels was chosen).

'DV1 Value Meal Choice' by 'DV7 Dessert Choice'		
	Pie chosen	Cookie chosen
Deluxe Hamburger Value Meal Chosen	2.57	0.75
Chicken Sandwich Value Meal Chosen	0.88	0.18
Fish Sandwich Value Meal Chosen	1.34	0.26

Figure 4.10 (Two-Way Counts Report)

In Figure 4.10, we see that the combination Deluxe Hamburger Meal + Pie was chosen 2.57% of the time.

Next, the top 10 full combinations selected are reported (the first seven in this report are shown in Figure 4.11):

Combination	Probability
[DV1 Value Meal Choice] Deluxe Hamburger Value Meal Chosen	23.16
[DV1 Value Meal Choice] Not chosen	11.60
[DV2 Sandwich Choice] Not chosen	
[DV3 Fries Choice] Not chosen	
[DV4 Drinks Choice] Not chosen	
[DV5 Salad Choice] Not chosen	
[DV6 Healthy Side Choice] Not chosen	
[DV7 Dessert Choice] Not chosen	
[DV1 Value Meal Choice] Chicken Sandwich Value Meal Chosen	9.69
[DV1 Value Meal Choice] Fish Sandwich Value Meal Chosen	8.74
[DV5 Salad Choice] Grilled chicken salad chosen	3.19
[DV4 Drinks Choice] Med drink chosen	2.44
[DV5 Salad Choice] Grilled chicken salad chosen	
[DV1 Value Meal Choice] Deluxe Hamburger Value Meal Chosen	2.31
[DV7 Dessert Choice] Pie chosen	

Figure 4.11 (Combinatorial Selections Counts Report)

The most common selection was to choose just the Hamburger Meal (23.16% of all choices). The 2nd most likely choice was to choose nothing (11.60% of choices). The 6th most common selection was to choose a Medium Drink + Grilled Chicken Salad (2.44% of choices). Although all top 10 most likely combinations are not shown in Figure 4.11 (to save space), it is interesting to note that largely due to the inclusiveness and popularity of the value meals, those top 10 combinations represent 65% of all choices made in the questionnaire.

5.7 Substitutes or Complements?

Counting analysis also can provide a simple way to investigate whether items on the menu are substitutes or complements. Consider a simple menu with a dozen items that are either selected or not. If item #1 is selected 20% of the time and item #2 is selected 30% of the time, and if these items were neither substitutes nor complements, we would expect that the likelihood of the two being selected together would be $0.2 \times 0.3 = 0.06$. MBC reports both one-way and two-way counts. If we find that the two-way joint likelihood of item #1 and item #2 being selected together is much larger than 0.06, then we would take this as evidence that these two items are complements. If the joint likelihood is much smaller than 0.06, then that is evidence that the items are substitutes. This suggests a simple measure for testing the synergies between items is to divide the actual joint likelihood (2-way counts probability) by the expected joint likelihood (product of the one-way counts probabilities). Values much larger than 1.0 suggest complements; values much less than 1.0 suggest substitutes. (For a more formal statistical test, the actual frequency of the joint occurrence may be compared to the expected frequency, and a Chi-square test may be performed with d.f. of 1.)

6 Conceptualizing MBC Models

6.1 Conceptualizing MBC Models

The MBC software is a flexible, scalable tool for analyzing menu-based, multi-check choice problems. Unlike Sawtooth Software's CBC, where the development of attribute lists with levels and estimation of utilities follows a similar pattern from project to project, MBC problems require more forethought, ingenuity, and conceptual know-how on your part. *Using MBC requires much more experience and expertise in statistics and econometric modeling than using CBC software!*

While designing the menu, you must constantly be thinking about how you plan to use MBC software to analyze the resulting data. You should generate designs and, *prior* to fielding the study, estimate the models using dummy (random) respondent data. Keep a close eye on the number of parameters you are estimating and whether you have enough respondents x choice tasks to estimate the desired effects with reasonable precision. After collecting the data, you'll make decisions about how to build the models (series of sub-models) that provide good fit to the calibration data and accurately predict how respondents will make choices given new menu scenarios.

Before we continue, it is useful to review some concepts and terminology:

- Dependent variables can be binary (on/off) or mutually-exclusive categorical: 1=choose red, 2=choose green, 3=choose blue, 4=choose no color (where one and only one category may be selected from the menu).
- The effect that a menu item's price has upon its own choice likelihood is termed an *own price effect*. The effect of a menu item's price on the choice likelihood of a *different* menu item is termed a *cross price effect*.
- We refer to *logit-based* models, which can mean aggregate logit models as well as HB models, as the HB estimation we employ uses the logit equation.
- We refer to *sub-models*, which are models that predict the choice for a specific item (or multiple items) on the menu, but not *all* items on the menu. By using multiple sub-models, we can "divide and conquer" complex menu problems.

6.2 Example #1

Consider a simple menu-based task for configuring additional options on a new automobile:

Which of the following options would you buy? Select as many as you wish, or none of the items.

Alloy Wheels \$2,000
 Moon roof \$1,200
 XM Radio \$300 (+\$13/month)
 Navigation System (in dash) \$1,600

Total Price of Selected Options: \$3,900

Figure 5.1

With four binary dependent variables, there are $2^4=16$ unique ways that respondents could complete the menu.

There are two main ways this model can be conceptualized using MBC software:

Option 1:

Construct four separate logit-based sub-models, where each sub-model predicts the likelihood that an option is chosen. Each model has a form identical to a binary logit model, where the dependent variable takes on two states: "chosen" and "not chosen."

- Sub-Model 1: Choice of Alloy Wheels (or not) is a function of its desirability, its price, and all other significant cross effects
- Sub-Model 2: Choice of Moon roof (or not) is a function of its desirability, its price, and all other significant cross effects
- Sub-Model 3: Choice of XM Radio (or not) is a function of its desirability, its price, and all other significant cross effects
- Sub-Model 4: Choice of Navigation System (or not) is a function of its desirability, its price, and all other significant cross effects

Given a new menu scenario with specific item prices, we can predict the likelihood that respondents would pick each item from the menu. To make the predictions, we run simulations for all four sub-models.

Option 2:

Since there are just 16 possible ways that respondents can complete the menu task, you could recode your dependent variable (in your .csv data file) from a series of four binary variables to a single categorical variable taking on values 1-16. One possible ordering of those 16 unique combinations is as follows:

- 1) Alloy Wheels, Moon roof, XM Radio, Navigation System
- 2) Alloy Wheels, Moon roof, XM Radio
- 3) Alloy Wheels, Moon roof, Navigation System

- 4) Alloy Wheels, Moon roof
- 5) Alloy Wheels, XM Radio, Navigation System
- 6) Alloy Wheels, XM Radio
- 7) Alloy Wheels, Navigation System
- 8) Alloy Wheels
- 9) Moon roof, XM Radio, Navigation System
- 10) Moon roof, XM Radio
- 11) Moon roof, Navigation System
- 12) Moon roof
- 13) XM Radio, Navigation System
- 14) XM Radio
- 15) Navigation System
- 16) (None)

With this formulation of the dependent variable, you could estimate a single logit-based (MNL) model, rather than four separate models. In effect, we are assuming a data generation process where the respondent considered all 16 possible combinations, together with the prices associated with those outcomes, and selected the *one* alternative with the highest total utility.

- Model: Choice of alternative (out of 16) is a function of its desirability and the prices of the items involved in that alternative.

Given a new menu scenario with specific prices, we can predict the choice likelihood (across the sample) of each of the 16 discrete outcomes. You separately will need to net the choice probability for each of the four options on the menu by summing the probability shares across alternatives that included that option. For example, choice of Alloy Wheels was an element of alternatives 1-8. By summing the probabilities across alternatives 1-8, we accumulate the likelihood that Alloy Wheels was selected on the menu. To predict the likelihood that Moon roof was selected on the menu, we sum the likelihood of alternatives 1, 2, 3, 4, 9, 10, 11, and 12. (MBC software allows you to create [Netted Variables](#) within the Market Simulator that automatically accumulate these netted probabilities, so that you can have the simulator report the probability of choice for each original item on the menu.)

Which way of formulating the model is better, Option 1 (four separate models) or Option 2 (a single unified model)? Based on our experience comparing these kinds of models, we'd recommend that if the number of possible ways that respondents can complete the menu is a reasonably small number, such as about 36 or fewer, building a single unified model will likely perform better. It is conceptually more complete to have a single model that predicts all possible combinatorial outcomes than to build a sequence of models that predicts each separate choice on the menu (but does not specifically consider combinatorial outcomes). But, obviously, such complete models are not always feasible. As the number of possible combinatorial outcomes becomes too large (such as 75 or more), the problem can become too big for HB to handle in reasonable time, the model tends to become too sparse at the individual level, and overfitting becomes a serious concern. *However, it is possible to think about **logical subsections of the menu** that may be modeled in this way, rather than having to*

model all possible outcomes in the menu using a single dependent variable. This is a recommended approach.

6.3 Example #2

One of the examples we showed earlier in this documentation was a choice of *a la carte* menu items vs. bundled value meals. We modify it slightly below to place restrictions on the way the choices may be made.

Below are three different restaurant options with menu items and their respective prices. Please select what you would choose, given these prices.

If you choose a value meal, you may only select one, and you cannot add additional items.

If you pick individual items (rather than a value meal), you may select multiple items from multiple restaurants in any combination.

McDonalds	Burger King	Wendy's
<input type="checkbox"/> Big Mac \$2.25	<input type="checkbox"/> Whopper \$3.25	<input type="checkbox"/> Classic Double \$2.75
<input type="checkbox"/> Large French Fries \$1.75	<input type="checkbox"/> Large French Fries \$1.49	<input type="checkbox"/> Biggie Fries \$1.49
<input type="checkbox"/> Large Drink \$1.49	<input type="checkbox"/> Medium Drink \$0.99	<input type="checkbox"/> Biggie Drink \$1.45
<input type="checkbox"/> Big Mac Extra Value Meal \$4.99	<input type="checkbox"/> Whopper Value Meal \$5.25	<input type="checkbox"/> Classic Double Combo Meal \$5.25

Figure 5.2

Imagine that there are logical restrictions on the way the menu may be completed: if respondents choose a value meal, they are only allowed to select one, and they aren't allowed to select any additional *a la carte* items from the menu. If they decide to purchase *a la carte* items (rather than a value meal), they may mix their *a la carte* orders across restaurant chains in any combination.

It turns out that there are too many possible outcomes ($3+2^9=515$) to model this well using HB and an exhaustive combinatorial outcomes approach.

Because the choice of value meal was mutually exclusive (only one value meal could be chosen), we code this as a single categorical dependent variable with 4 possible outcomes (1=1st value meal chosen, 2=2nd value meal chosen, 3=3rd value meal chosen, 4= no value meal chosen). Modeling mutually exclusive, categorical outcomes using a single categorical dependent variable (MNL formulation) generally will be more accurate than treating that single choice as a series of binary dependent variables (binary logits).

There are at least three ways to use MBC to model Example 2.

Option 1:

The easiest way to conceptualize the model is that the choice of each item on the menu is a function of its desirability, its own-effect of price, and any significant cross-effects from other item's prices from the menu. A separate sub-model is built to predict each *a la carte* menu item (9 models) and a sub-model is built to predict the likelihood of picking one of the value meals vs. no value meal (1 model), for a total of 10 separate logit models.

The weakness of this series of sub-models is that there is no recognition that it is logically inconsistent for a respondent to pick both an *a la carte* menu item and a value meal. This could lead to nonsensical individual-level predictions and harm predictive accuracy for the aggregate choice likelihoods of menu items.

Option 2:

This model assumes a hierarchical (nested) decision structure. Respondents are viewed first to consider whether they will purchase a value meal or not. If respondents do not choose a value meal, then they decide which of the *a la carte* items to select. A sub-model is built for each of the 9 *a la carte* items, predicting the likelihood that respondents will select each from the menu. As with option 1, there are 10 separate logit models, but with a key difference discussed below.

The nesting structure in MBC is accomplished by retaining only the choice tasks where value meals were *not* selected when estimating the 9 sub-models accounting for the choices of *a la carte* items. In MBC software, choice of a dependent variable can be specified to be conditional on the choice of another dependent variable. We specify that the choice of each of the 9 *a la carte* items is conditional on *no value meal* being selected.

Setting up a nesting structure formally informs (via simulations) that respondents cannot pick both a value meal and an *a la carte* option. It restricts simulated predictions to the space of feasible outcomes.

During simulations, and at the individual level, the likelihood that respondents would not pick a value meal is multiplied by the likelihood of selecting each of the *a la carte* items given non-choice of the value meal.

Option 3:

This approach is just like Option 2, except that rather than model the *a la carte* choices as a series of 9 binary logits, we collapse these binary outcomes into three dependent variables representing all feasible combinations of specific items:

Dependent variable 1: (all combinations of burger choices)

Dependent variable 2: (all combinations of fries choices)

Dependent variable 3: (all combinations of drinks choices)

Note that each of these collapsed dependent variables has $2^3 = 8$ possible outcomes. By using counting analysis, we may find that some of these 8 outcomes were never chosen (for example, it's highly unlikely that any respondent would choose three burgers, one from each restaurant). If that is the case, we may decide we have enough evidence to code a collapsed dependent variable with fewer than 8 outcomes. This restricts the model to only the feasible outcomes, and should lead to improved accuracy. Note that by doing so we assume the non-observed outcomes in the sample also have zero likelihood of choice within a new sample.

This approach simplifies the menu to 4 dependent variables in total (instead of 10): the three directly above, as well as the primary choice of whether to purchase a value bundle or not.

6.4 Example #3

In the previous section on counting analysis, we described a way to examine whether pairs of items on the menu appear to be substitutes or complements (by comparing the actual joint likelihood of choice to the expected joint likelihood). We discuss this topic further in this example.

Imagine you were at the grocery store to buy food for a Saturday afternoon barbeque for 4 people. Please assume you currently have no food items at your house. If the following items were the only things available, which would you choose? (Select no items if you wouldn't pick anything.)

Main:

- Hamburgers (1 dozen patties) (\$6.99)
- Hamburger buns (1 dozen buns) (\$1.59)
- Hot Dog Buns (10-pack) (\$1.59)
- Hot Dogs (8-pack) (\$2.59)
- Steaks (4 steaks, 3 lbs total) (\$24.99)
- Ribs (4 lbs total) (\$14.99)
- Chicken Breasts (1/2 dozen) (\$8.99)

Chips:

- Potato Chips (\$1.89)
- Tortilla Chips (1.79)

Beverages:

- Milk (1-quart) (\$1.20)
- Milk (1-gallon) (\$3.00)
- Orange Juice (1 quart) (\$2.00)
- Orange Juice (1 gallon) (\$4.50)
- Soda (6-pack) (\$2.29)
- Soda (12-pack) (\$4.00)
- Beer (6-pack) (\$3.50)
- Beer (12-pack) (\$6.00)

Condiments:

- Salsa (\$2.29)
- Relish (\$1.89)
- Mustard (\$2.69)
- Ketchup (\$1.59)
- Steak Sauce (\$2.99)

One way to code the data for this menu is to consider that there are 22 separate checkboxes. Thus, we have 22 dependent variables each assuming either code 1 or 2 (1=chosen; 2=not chosen). We could build 22 separate binary logit/HB models, where each model predicted the outcome of

one checkbox as a function of the desirability of the food item for that checkbox, its price, and any significant cross-effects.

However, if we realize (either *a priori* or by examining the data via counts) that there are options on the menu that are very strong complements, then it may make sense (after collecting the data) to recode a few binary dependent variables items into a single dependent variable representing all feasible combinatorial outcomes. For example, hamburger patties and hamburger buns are *very* complementary. It would seem very strange for a person to purchase hamburger patties without also buying the buns. The new 4-level dependent variable is coded as follows:

- 1 = Hamburger patties
- 2 = Hamburger buns
- 3 = Hamburger patties and hamburger buns
- 4 = No hamburgers or buns

The act of grouping strongly complementary menu items into categories of a *single* dependent variable can lead to more accurate market simulation models. It more formally captures the complementary synergies among items.

Imagine that we performed counting analysis and across 800 respondents and 8,000 tasks find that no task (or very few tasks) ever reflected code 1 of the collapsed 4-level dependent variable directly above (Hamburger patties only). If we were willing to make the assumption that the likelihood of observing that outcome was zero in future market simulations, we could recode the dependent variable with just three outcomes (and delete as outliers any tasks that didn't result in these three codes):

- 1 = Hamburger buns
- 2 = Hamburger patties and hamburger buns
- 3 = No hamburgers or buns

One could extend the same idea to capture the synergies among 3 or 4 items that were strong complements *and/or substitutes*. But, once the number of categories of a collapsed dependent variable exceeds about 36 or so, this approach could tax the model in terms of numbers of parameters to be estimated and probably is counterproductive.

You might wonder what would happen if two items were strong substitutes or complements, and we neglected to recognize this and collapse the items into a single dependent variable. When running counts analysis, you would probably recognize that there was a strong cross-effect for the price of hamburger patties affecting the choice of buns (and perhaps vice-versa). When including those cross-effects in the HB models, the substitution patterns would be enhanced. But, perhaps more importantly, HB provides utility parameters at the individual level, and respondents who purchase hamburger patties probably also purchase hamburger buns. Respondents who don't purchase hamburger patties also probably don't purchase hamburger buns. Capturing such heterogeneity via HB helps ensure that the market simulator reflects complementary and substitution effects among items on the menu.

There are many other pairs or triads of complementary items that could be considered in addition to the one we identified:

Strong Complements:

Hotdogs and hotdog buns
Tortilla chips and salsa
Mustard and ketchup and relish (a triad)
Steaks and steak sauce

Strong Substitutes:

1 quart milk and 1 gallon milk
6-pack soda and 12-pack soda
1 quart orange juice and 1 gallon orange juice
6-pack beer and 12-pack beer

We should point out that if a menu item is included as an element of multiple categories of the *same* dependent variable, when market simulation predictions are made, the total likelihood of choosing that item is the sum of the probabilities of the categories of the dependent variable that include that item. Therefore, *it would be incorrect to represent an item as an element of categories of two or more dependent variables! This would lead to double-counting the probability of choice for that item in MBC.*

6.5

Combinatorial Dependent Variable Coding Suggestions

We have discussed the benefits of considering if groups of items on your menu seem to be complementary or substitutable, and to consider collapsing multiple binary choices (from either the entire menu *or a subset of the menu*) into a single dependent variable. For example, eight separate items on a subset of a restaurant menu all from the sandwiches category might be collapsed into a single dependent variable. Although there are $2^8 = 256$ possible ways to choose among eight sandwiches on the menu, perhaps only 10 combinations of sandwich choices were ever chosen with enough frequency to worry about accounting for in the model. Perhaps there are a *very few* observations of other combinations beyond those 10 most frequent combinations. In this example, it's quite probable that the 10 most frequent combinations of choices of sandwiches represent more than 98% of all sandwich combinations ever chosen. What we might think about doing is to treat these *very infrequent* remaining sandwich choice combinations as outliers, and to drop these tasks altogether. It's quite possible that many or most of the dropped tasks actually were outliers: responses made at random or due to individual response error. Researchers regularly discard outliers in other market research data and obtain better representation of the true population distributions. Perhaps by discarding combinatorial choice outliers (by dropping such tasks) we can leverage the power of combinatorial dependent variable coding, keep the number of categories in each collapsed dependent variable to a much more manageable number, remove outlier tasks and noise, and thus obtain better overall predictions.

When using combinatorial dependent variable coding, it is helpful for reporting purposes within the market simulator to aggregate across multiple categories using [Netted Variables](#), thereby constituting the choices back to their original categories from the menu. Thus, rather than reporting that the choice of Fries is accounted for by multiple categories of a combinatorial dependent variable, a single Fries netted variable is reported that automatically sums across those multiple categories.

6.6 Conceptualizing the Model Summary

Keep these guidelines in mind:

- If there are mutually exclusive choices on the menu (e.g. radio buttons forcing exclusive choices among multiple options), make sure these are coded as a single categorical dependent variable. For example, the choice among three value meals vs. no value meals in Example 2 should be coded as a single dependent variable with four outcomes. It would be a mistake to code the mutually exclusive choice among the three value meals as three binary (on/off) dependent variables.
- Enumerate the total possible ways that respondents can complete the menu (or subsets of a menu). If there are about 36 or fewer possible ways that were actually observed in the data with significant frequency, you should consider modeling the menu selection as a single model that assumes that the respondent viewed the menu (or subset of the menu) as a discrete choice (MNL) among these possible alternatives.
 - o Consider whether some items on the menu are strong complements or substitutes with other items on the menu (counting analysis helps assess this). If so, consider recoding the data into a single dependent variable representing the unique possible combinations of those menu items. Again, if none (or very few) of the respondent tasks ever reflect certain chosen combinations, you might decide to omit those categories in the collapsed dependent variable and drop such outlier responses.
- If the choice of an item on the menu is conditional on another item either being chosen or not chosen, then it can improve the accuracy of the resulting simulator if a nesting structure is specified. In a nesting structure, some dependent variables are conditional on respondents first selecting or rejecting another item on the menu. In Example 2 shown previously, we may specify that the choice of *a la carte* items was conditional upon respondents first rejecting all value meals. Imposing nesting structures can ensure that the model formally recognizes that certain menu choice combinations are not feasible.

6.7

Independent Variables: Generic or Alternative-Specific

In the previous section on counting analysis, we discussed coding independent variables as linear, log-linear, and part-worth (dummy coded). In this section, we discuss whether independent variables should be parameterized as *generic* or *alternative-specific*. We describe it conceptually here, and then show how to use MBC software to perform the steps in the Specify Models section that follows.

To help us describe the difference between *generic* and *alternative specific* independent variables, let's consider a simple CBC problem, with three sandwich options (Hamburger, Chicken Sandwich, and Fish Sandwich) at different prices (Figure 5.4). (Our discussion of this CBC problem is directly applicable to MBC analysis) For ease of discussion, let's assume that the prices for all three sandwiches take on just three variations:

- Level 1: \$2.00
- Level 2: \$2.50
- Level 3: \$3.00

Further assume that we have decided to model price as a linear function. In CBC and MBC analysis, we recommend centering quantitative variables such as price when estimating linear terms (and giving the centered prices a range of 1.0). Therefore, the values -0.5, 0.0 and 0.5 are used to represent \$2.00, \$2.50, and \$3.00.

Which of the following would you choose?		
Hamburger	Chicken Sandwich	Fish Sandwich
\$2.00	\$3.00	\$3.00

Figure 5.4

Under the standard MNL model formulation for this CBC task, the simplest model (with fewest parameters to estimate) assumes main effects. A generic price function is captured, indicating respondents' sensitivity to price changes (holding sandwich type constant). This is coded in the independent variable (X matrix) as three rows (alternatives) with three columns (independent variables):

A	B	C
0	0	-0.5
1	0	0.5
0	1	0.5

We've labeled the columns A, B, and C. Column A is a dummy-coded column that when submitted to logit estimation captures the utility weight of Chicken Sandwich, and column B is a dummy-coded column to capture the utility of Fish Sandwich. To avoid linear dependency, the utility of Hamburger has reference value of zero.


C is a column reflecting the zero-centered price for the sandwich alternatives, and captures a *generic* price effect (that applies commonly across the three sandwich types).

It's common for researchers to investigate (via 2 log-likelihood tests in aggregate logit) whether the assumption of a generic price effect holds, or whether the model fit may be significantly improved by capturing *alternative-specific* price effects (e.g. the independent price effect for each sandwich type). Alternative-specific price effects may be captured by coding price separately (as unique columns) for each alternative:

A	B	C	D	E
0	0	<u>-0.5</u>	0.0	0.0
1	0	0.0	<u>0.5</u>	0.0
0	1	0.0	0.0	<u>0.5</u>

Columns A and B are unchanged from our earlier example, as the only differences will be found in the way we parameterize the prices. Rather than capturing price as a single column, we have distributed price across three separate columns (C, D, and E), corresponding to each of the three sandwich alternatives. We've bolded and underlined the three prices for each alternative in column C. In the first row, column C contains the price for Hamburger (-0.5). Column D captures the price effect for Chicken Sandwich, and it does not apply to row 1 (the Hamburger row). When a price doesn't apply, it is specified at its center of gravity (0).

Specifying price variables (or other independent variables other than price) as alternative-specific typically increases the fit of the model, but at the expense of requiring more parameters to estimate. When you use MBC software to specify models, such as the choice of sandwich on the menu, you may choose whether to assume that independent variables have generic effects on alternatives or alternative-specific effects. We strongly encourage you to investigate (via 2 log-likelihood tests using the aggregate logit routine included in MBC) whether the increased model fit due to alternative-specific price effects is large enough to justify the increased number of parameters in the model. Logit models often benefit from many alternative-specific effects; whereas HB models sometimes are hindered by them, leading to overfitting.

For each dependent variable, MBC asks you to specify which independent variables to include in the model, and how each independent variable should be coded: linear, log-linear, or part-worth coded. For dependent variables that are multi-level categorical, such as the choice of sandwich (1=Hamburger, 2=Chicken Sandwich, 3=Fish Sandwich), MBC asks you to pick the independent variables to use, and the way in which you specify the effects indicates whether they are generic or alternative-specific. In a subsequent step, you are given the opportunity to ask MBC to collapse the effect of different independent variables into a common variable and generic effect. (Using the  Group & Collapse... icon).

For example, consider that we have used MBC to model the choice of sandwich (Hamburger, Chicken Sandwich, or Fish Sandwich). For each category of the dependent variable (1=Hamburger, 2=Chicken Sandwich, 3=Fish Sandwich) MBC has asked us to indicate which independent variables are to be included, along with their functional form (we've specified linear). We indicate **Price_Hamburger**, **Price_ChickenSandwich**, **Price_FishSandwich** as the independent variables associated with each of the categories of the dependent variable. MBC initializes the following

design matrix, with alternative-specific price effects (you can click the *Preview Design* button to view how MBC codes the independent variable matrix):

A	B	C	D	E
0	0	-0.5	0.0	0.0
1	0	0.0	0.5	0.0
0	1	0.0	0.0	0.5

Let's say that we would like to assume that the price effect doesn't differ between Hamburger and Chicken Sandwich (alternatives 1 and 2), but is different for Fish sandwich (alternative 3). We select columns C and D of the independent variable matrix and ask MBC to collapse those two columns as a generic effect. First, MBC makes sure the columns are coded in the same way (linear, log-linear, or part-worth). Because they are both linear in this case, MBC looks at the assigned level values associated with Hamburger and Chicken Sandwich, and re-centers the prices for Hamburgers and Chicken Sandwiches on a common zero-centered scale (they already were on the same scale in our example, so no change is necessary), and then MBC collapses the values of the rescaled elements for columns C and D:

A	B	CD	E
0	0	-0.5	0.0
1	0	0.5	0.0
0	1	0.0	0.5

We have labeled the collapsed column "CD." If we wanted to capture price as a generic factor across all three sandwich alternatives, we could ask MBC to collapse columns C, D, and E. MBC software first puts the prices for Hamburgers, Chicken Sandwiches, and Fish Sandwiches on a common zero-centered scale, and then collapses the corresponding (rescaled) elements of columns C, D, and E.

A	B	CDE
0	0	-0.5
1	0	0.5
0	1	0.5

We have described how MBC software can parameterize all independent variables as alternative-specific. This is an expensive way to build models (large number of parameters to be estimated), but it leads to the potential of greater model fit if indeed the effect of independent variables is different for each category of the dependent variable. MBC allows the researcher to select which columns of the X matrix to collapse as generic effects across specific categories of the dependent variable. One can investigate different ways to parameterize the model that will result in very little loss in model fit, but reduce the number of parameters to be estimated. This work is a relatively manual process, done using MBC's aggregate logit routine, and with 2 log-likelihood testing.

7 Building Models (Specify Models Tab)

7.1 Specify Models Tab

In this section, we take the theory we introduced in the last section and describe how to put it into practice: how to use MBC software to specify (build) different models that predict choices on the menu. You specify models involving independent variables that predict each dependent variable. For each dependent variable, you'll select which independent variables to use, and how they should be coded (linear, log-linear, or part-worth).

We'll begin with the small example introduced earlier. Respondents can choose among three options, each with varying prices across the tasks:

- Hamburger \$1.99
- Fries \$1.39
- Drink \$1.19

The data format is shown below (Figure 6.1). The three independent variables are **Pr_Ham** (price of the hamburger), **Pr_Fries** (price of fries), and **Pr_Drink** (price of drink). These price variables can each take on 5 possible values, from lowest to highest price. Following the independent variables in the data file are the three dependent variables: **Ch_Ham** (choice of hamburger), **Ch_Fries** (choice of fries), and **Ch_Drink** (choice of drink). The dependent variables are coded as 1=chosen and 2=not chosen.

	A	B	C	D	E	F	G
1	CaseID	Pr_Ham	Pr_Fries	Pr_Drink	Ch_Ham	Ch_Fries	Ch_Drink
2	1001	3	1	2	2	1	2
3	1001	1	2	3	1	1	2
4	1001	5	2	1	2	1	2
5	1001	4	4	2	1	1	2
6	1001	1	3	5	1	1	2
7	1002	2	2	2	2	2	1
8	1002	1	4	3	2	2	1

Figure 6.1 (Simple Data File)

After selecting this data file within MBC, you then will have classified and labeled the independent variables and dependent variables from the *Variables* tab (Figure 6.2). These steps were described previously in this documentation.

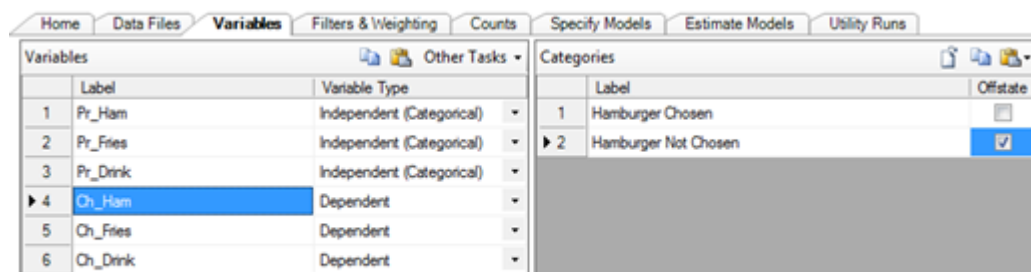


Figure 6.2 (Variables Tab)

Because there are three dependent variables, we will build three separate models: one model (a sub-model) per dependent variable. When you navigate to the *Specify Models* tab, you see the following dialog (Figure 6.3):

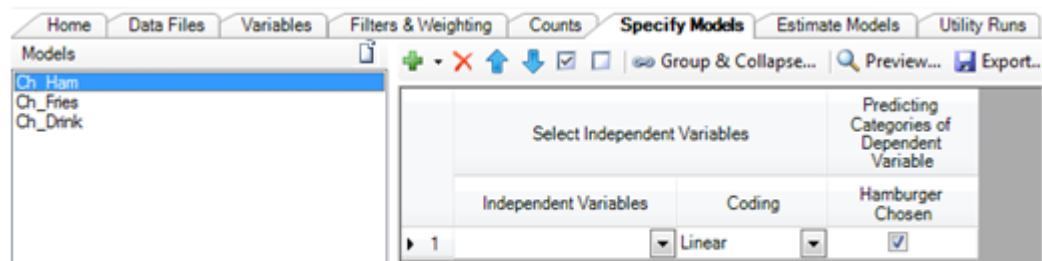


Figure 6.3

Along the left, you see the three dependent variables (**Ch_Ham**, **Ch_Fries**, and **Ch_Drink**). We'll begin with the first sub-model (predicting choice of hamburger). We highlight the **Ch_Ham** label with the mouse, and begin adding independent variables (predictor variables) to the sub-model predicting choice of hamburger.

(In reality, such a simple menu should be modeled using a single dependent variable that coded all 8 possible combinations that could be selected (as shown in the subsequent section of this chapter). But, for illustration, we first demonstrate how three sub-models could be built, one predicting choice of each of the three menu items.)

Let's imagine that after studying counts analysis, we believe choice of hamburger depends on the price of the hamburger (**Pr_Ham**) as well as the price of the fries (**Pr_Fries**), but *not* the price of the drink. Click the **Add Row** icon (+) twice (see Figure 6.4, below) to specify the two independent variables. Then, select the coding for each of the two independent variables (specify *Linear* to keep things simple).

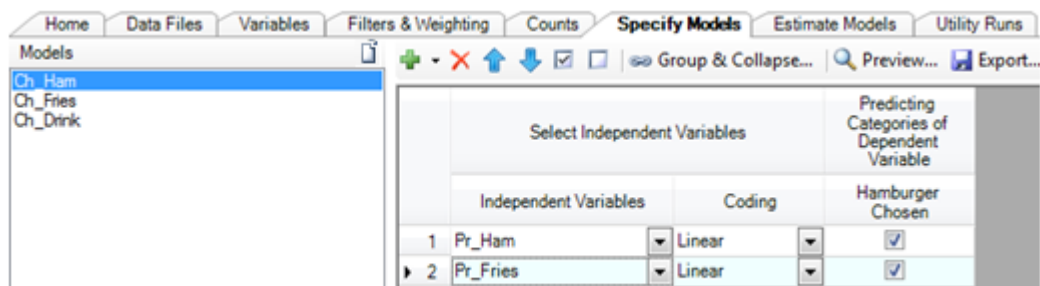


Figure 6.4 (Selecting Independent Variables in a Model)

To see how the specified coding within Figure 6.4 is actually implemented in the design matrix, click the **Preview...** (**Preview Design Matrix**) button.

Preview Design Matrix

← Respondent: 1001 →

	Task/Alternative	ASC (Hamburger Chosen)	Pr_Ham [Linear]	Pr_Fries [Linear]	Response
1	Hamburger Chosen	1	0	-0.5	0
1	Hamburger Not Chosen	0	0	0	1
2	Hamburger Chosen	1	-0.5	-0.167	1
2	Hamburger Not Chosen	0	0	0	0
3	Hamburger Chosen	1	0.5	-0.167	0
3	Hamburger Not Chosen	0	0	0	1
4	Hamburger Chosen	1	0.25	0.5	1
4	Hamburger Not Chosen	0	0	0	0
5	Hamburger Chosen	1	-0.5	0.167	1
5	Hamburger Not Chosen	0	0	0	0

Figure 6.5 (Preview Design Matrix)

The ability to preview the design matrix keeps you informed and in control of the mechanics of model building in MBC, absent the hassle of so much data processing work. Analysts using MBC software should already be familiar with coding independent variable matrices for econometric models like regression and MNL. **Preview Design Matrix** makes the entire process transparent and reassuring, because you can see exactly how the design matrix is being coded.

In Figure 6.5, the design matrix for the first respondent is shown (Respondent #1001). You can click the right arrow at the upper-left of the dialog to display the design for the next respondent, if you wish.

The rows are alternatives in the choice task. Since the dependent variable has two possible outcomes: chosen/not chosen, each task is formatted with two alternatives. The first alternative in each set represents the chosen state, and the second represents the non-chosen (off) state. The off state is coded as a row of zeros. The chosen state is coded with levels describing the characteristics of the menu item (in this case, the hamburger).

We capture the desirability of the menu item itself via a dummy-coded Alternative-Specific Constant (ASC). Then, the remaining independent variables (prices for hamburger and fries) are coded as single columns (because we requested linear coding), zero-centered with a range of 1.0. Because the prices in this example were equidistant and symmetric, the lowest price (level 1) is represented by -0.5, and the highest price (level 5) with 0.5.

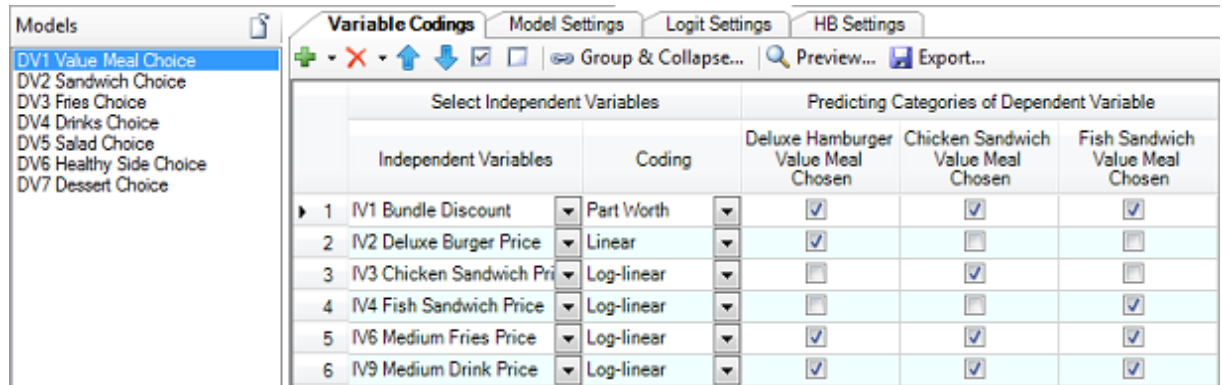
(If we had chosen to code the price variables as part-worth functions, there would be $5-1 = 4$ dummy-coded columns to code the 5 price levels.)


The final column records the respondent choice (the dependent variable), where 1=chosen, and 0=not chosen.

7.2 Importing Model Specifications

The *Variable Codings* dialog allows you to specify models by pointing and clicking. However, some researchers may prefer to import their model specifications from a .CSV file to better automate the procedure—especially if a great many dependent variables are involved.

Consider the following model specification that has been manually specified within the MBC software:



You can create a .CSV file that when imported (by clicking the  icon in the upper-left of the *Variable Codings* dialog) generates this same model specification (as if you had clicked all these settings within the dialog). Here is the format that generates this model specification.

DV_Index	IV_Index	Coding	Category_1	Category_2	Category_3
17	1	1	1	1	1
17	2	2	1	0	0
17	3	3	0	1	0
17	4	3	0	0	1
17	6	3	1	1	1
17	9	3	1	1	1

The first row (labels) is optional in the .CSV file.

- DV_Index** is an index number that indicates which Dependent Variable model is being specified. The first variable displayed in the *Variables* tab (whether an independent variable or dependent variable) is index #1, the second variable in the data file is index #2, etc. In this data file, the first dependent variable is actually variable #17). More than one dependent variable index may be included in the file and may be specified in any order.

- **IV_Index** is an index number that indicates which Independent Variable is being included in the model. As with the dependent variables, we refer to them using indices indicating their order in the data file (refer to the *Variables* tab to see the list of variables and their indices). In this particular data file, the first Independent Variable is index #1, the second Independent Variable is index #2, etc.
- **Coding** indicates which coding to use, where 1="Part Worth", 2="Linear", 3="Log-Linear" and 4="Excluded". (If the independent variable is a Continuous variable, then the coding is automatically set to "User-Specified" unless you specified it is 4="Excluded").
- **Category_X** indicates which column number (category) in the Variable Codings dialog is being predicted by the independent variable. This information is only needed if more than one category exists in the Variable Codings dialog for the dependent variable. A "1" indicates selected and a "0" means not selected. The number of columns is equal to the number of categories for the dependent variable (number - 1 if a category has been marked as the off-state).

Multiple dependent variable models may be imported by continuing to append additional rows to the .CSV file, for additional dependent variables.

Any imported models will overwrite existing models.

7.3

Exhaustive Alternatives (Combinatorial) Coding Approach

Let's again use the simple example introduced near the beginning of this documentation:

- Hamburger \$1.99
- Fries \$1.39
- Drink \$1.19

It is probably better to model this tiny menu as a single dependent variable with eight possible outcomes rather than three separate binary dependent variables. Coding the data as a single dependent variable explicitly considers the *combinations* of items respondents select. The dependent variable may be coded with eight categories as follows:

- 1= Hamburger & Fries & Drink
- 2= Hamburger & Fries
- 3= Hamburger & Drink
- 4= Fries & Drink
- 5= Hamburger
- 6= Fries
- 7= Drink
- 8= Nothing (off state)

The *Variables* dialog should look like Figure 6.6, with **Choice** as the dependent variable, with eight categories, and three independent variables **Pr_Ham** (price of hamburger), **Pr_Fries**, and **Pr_Drink**.

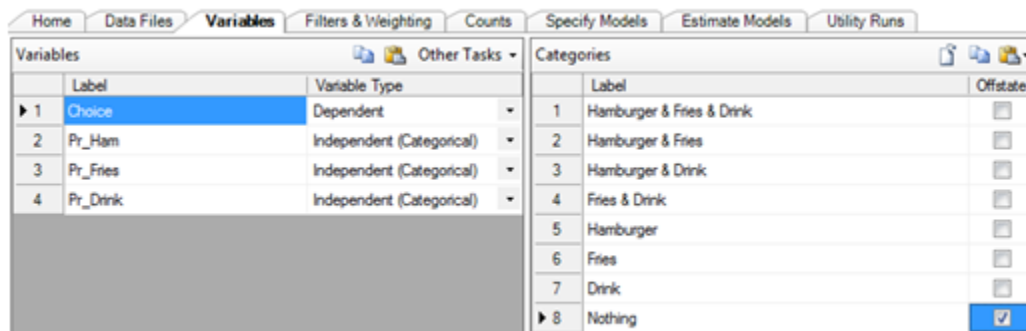


Figure 6.6

A recommended way to code the independent variables is to estimate separate price sensitivities for each of the three prices of menu items (hamburger price, fries price, and drinks price). The *Variable Codings* grid should look like the following:

Select Independent Variables			Predicting Categories of Dependent Variable						
	Independent Variables	Coding	Hamburger & Fries & Drink	Hamburger & Fries	Hamburger & Drink	Fries & Drink	Hamburger	Fries	Drink
1	Pr_Ham	Linear	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	Pr_Fries	Linear	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	Pr_Drink	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Figure 6.7

Preview Design Matrix displays:

Task/Alternative		ASC (1)	ASC (2)	ASC (3)	ASC (4)	ASC (5)	ASC (6)	ASC (7)	Hamburger Pr [Linear]	Fries_Pr [Linear]	Drink_Pr [Linear]	Response
1	Hamburger & Fries & Drink	1	0	0	0	0	0	0	0.25	-0.25	0.25	0
1	Hamburger & Fries	0	1	0	0	0	0	0	0.25	-0.25	0	0
1	Hamburger & Drink	0	0	1	0	0	0	0	0.25	0	0.25	1
1	Fries & Drink	0	0	0	1	0	0	0	0	-0.25	0.25	0
1	Hamburger	0	0	0	0	1	0	0	0.25	0	0	0
1	Fries	0	0	0	0	0	1	0	0	-0.25	0	0
1	Drink	0	0	0	0	0	0	1	0	0	0.25	0
1	Nothing	0	0	0	0	0	0	0	0	0	0	0

Figure 6.8 (Design Matrix)

Alternatively, you could code the price for each of the eight alternatives as eight "user-defined" continuous variables (within your .csv data file). You could then read the data file into MBC, and code these eight independent variables as alternative-specific price variables. This would lead to 15 parameters to estimate, as opposed to the more parsimonious 10 parameters as shown in Figure 6.8.

While it is true that most every MBC project you do will have too many possible alternatives to code the *entire* menu as a single dependent variable with all exhaustive combinations as categories, **you should consider tackling logical sections of a larger menu with the *Exhaustive Alternatives* coding approach.** This particularly can be useful if collapsing menu items that are strong substitutes into all feasible combinatorial outcomes.

7.4 A More Complex Fast-Food Example

In the next example, we'll use a restaurant study menu, and a model predicting a dependent variable with 4 mutually-exclusive outcomes. Consider the fast-food menu shown in Figure 6.9:

Menu Scenario #1: Please imagine you pulled into a fast-food restaurant to order dinner for <u>just yourself</u> . If this were the menu, what (if anything) would you purchase?		
<input type="radio"/> Deluxe Hamburger Value Meal -Deluxe Hamburger -Medium fries -Medium drink \$3.99	<input type="radio"/> Chicken Sandwich Value Meal -Chicken Sandwich -Medium fries -Medium drink \$5.59	<input type="radio"/> Fish Sandwich Value Meal -Fish Sandwich -Medium fries -Medium drink \$3.99
(Only order sandwiches, fries or drinks from this area if you did not pick a value meal above.) Sandwiches: <input type="radio"/> Deluxe Hamburger \$1.99 <input type="radio"/> Chicken Sandwich \$3.59 <input type="radio"/> Fish Sandwich \$1.99		Salads: <input type="radio"/> Cobb dinner salad \$4.79 <input type="radio"/> Grilled chicken salad \$4.39 Healthy Sides: <input type="radio"/> Carrots/Celery with Ranch dressing \$1.19 <input type="radio"/> Apple slices/Grapes with dipping sauce \$0.99 Desserts: <input type="radio"/> Apple/Cherry/Berry pie \$0.99 <input type="radio"/> Cookies \$1.19
Fries: <input type="radio"/> Small \$0.79 <input type="radio"/> Medium \$1.49 <input type="radio"/> Large \$1.69 Drinks: <input type="radio"/> Small \$0.99 <input type="radio"/> Medium \$1.69 <input type="radio"/> Large \$2.19		
<input type="radio"/> I wouldn't buy anything from this menu. <input type="radio"/> I'd drive to a different restaurant, or do something else for dinner.		

Figure 6.9 (A Fast-Food Menu)

We'll demonstrate how to build the model that predicts the choice of sandwiches from the *a la carte* section of the menu (indicated by the arrow). The four possible outcomes of that dependent variable (**Sandwich_Choice**) are:

- 1=choose Deluxe Hamburger
- 2=choose Chicken Sandwich
- 3=choose Fish Sandwich
- 4=no choice

When we click the *Specify Models* tab, and select the dependent variable **Sandwich_Choice**, we see the display in Figure 6.10:

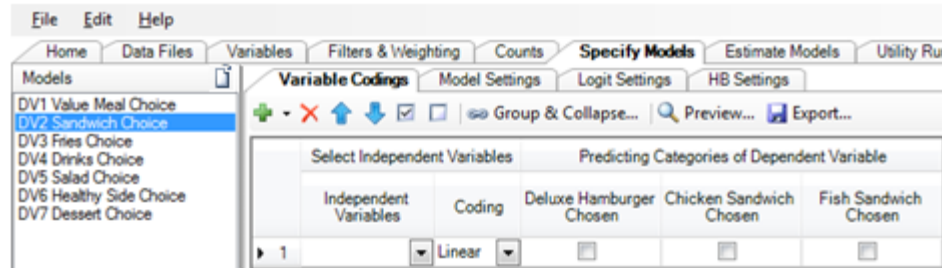


Figure 6.10 (Variable Codings display)

Figure 6.10 displays a grid, where we select different independent variables to predict categorical outcomes of the dependent variable (**Sandwich_Choice**). The three categories of the dependent variable shown are the *non-off state* categories. The off state is the constant, and is represented by a row of zeros in the design matrix, and thus doesn't require any additional coding specification here.

Let's imagine the after studying the counts report, we hypothesized that the choice of sandwiches was affected by the following independent variables:

- Bundle_Discount** (the discount for the value meal combo)
- Deluxe_Burger_Price** (the price of the deluxe hamburger)
- Chicken_Sandwich_Price** (the price of the chicken sandwich)
- Fish_Sandwich_Price** (the price of the fish sandwich)

Our task now is to add those independent variables to the grid, and specify which functional form to use for each. To keep things simple, we'll use linear coding for the price variables.

Specifying a Generic Effect across Three Alternatives

First, let's assume that **Bundle_Discount** has a generic effect on the preference for each of the *a la carte* sandwiches. If the value meal discount is relatively small, there will be an increased preference for the *a la carte* sandwiches. Because we are choosing a generic effect, we are assuming that the effect of **Bundle_Discount** on choice of *a la carte* sandwich does not differ across the sandwiches, but leads to a uniform increase or decrease in utility for the three options. Specify the following as shown in Figure 6.11:

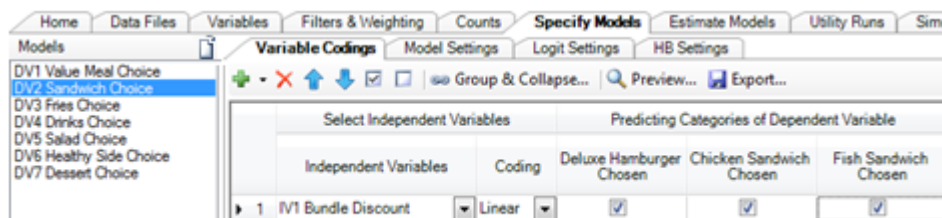


Figure 6.11 (Generic Effect on Preference for *a la Carte* Sandwich)

To see the independent variable matrix, click **Preview Design Matrix**.

Preview Design Matrix						
← Respondent: 1 →						
	Task/Alternative	ASC (Deluxe Hamburger Chosen)	ASC (Chicken Sandwich Chosen)	ASC (Fish Sandwich Chosen)	IV1 Bundle Discount [Linear]	Response
1	Deluxe Hamburger Chosen	1	0	0	0.167	0
1	Chicken Sandwich Chosen	0	1	0	0.167	0
1	Fish Sandwich Chosen	0	0	1	0.167	0
1	Not chosen	0	0	0	0	1
2	Deluxe Hamburger Chosen	1	0	0	0.5	0
2	Chicken Sandwich Chosen	0	1	0	0.5	0
2	Fish Sandwich Chosen	0	0	1	0.5	0
2	Not chosen	0	0	0	0	1

Figure 6.12 (Generic Effect on Choice of *a la Carte* Sandwich)

In Figure 6.12, each task is coded with four rows (alternatives). The fourth row is the *no choice* alternative (the off state), and is coded as a row of zeros across the independent variables. The desirability of each of the three sandwiches is captured using dummy-coded Alternative-Specific Constants (ASCs) labeled ASC(Deluxe Hamburger Chosen), ASC(Chicken Sandwich Chosen), and ASC(Fish Sandwich Chosen).

The next column, **Bundle_Discount (Linear)** is a generic effect (because only one column is being specified to capture a single utility weight that affects all three non-off state categories of the dependent variable). The prices are coded as zero-centered, with a range of 1.0. Thus, the relative price 0.167 applied to the value meals in the first task affects the choice (in a generic way) of the three sandwich alternatives.

Specifying Alternative-Specific Effects across Three Alternatives

Let's imagine that we believed that the effect of **Bundle_Discount** on choice of sandwich was not generic, but differed depending on the sandwich option. We could specify three alternative-specific effects rather than a single generic effect:

Home Data Files Variables Filters & Weighting Counts Specify Models Estimate Models Utility Runs Simu						
Models						
Variable Codings Model Settings Logit Settings HB Settings						
Select Independent Variables Predicting Categories of Dependent Variable						
	Independent Variables	Coding	Deluxe Hamburger Chosen	Chicken Sandwich Chosen	Fish Sandwich Chosen	
1	IV1 Bundle Discount	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	IV1 Bundle Discount	Linear	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	IV1 Bundle Discount	Linear	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 6.13 (Alternative-Specific Effects)

Note that we have asked for three independent variables to be coded in the design matrix, and that each one applies only to one of the sandwich options. When we click **Preview Design Matrix**, we see the resulting alternative-specific coding:

Preview Design Matrix

← Respondent: 1 →

Task/Alternative	ASC (Deluxe Hamburger Chosen)	ASC (Chicken Sandwich Chosen)	ASC (Fish Sandwich Chosen)	IV1 Bundle Discount [Linear]	IV1 Bundle Discount [Linear] (2)	IV1 Bundle Discount [Linear] (3)	Response
1 Deluxe Hamburger Chosen	1	0	0	0.167	0	0	0
1 Chicken Sandwich Chosen	0	1	0	0	0.167	0	0
1 Fish Sandwich Chosen	0	0	1	0	0	0.167	0
1 Not chosen	0	0	0	0	0	0	1
2 Deluxe Hamburger Chosen	1	0	0	0.5	0	0	0
2 Chicken Sandwich Chosen	0	1	0	0	0.5	0	0
2 Fish Sandwich Chosen	0	0	1	0	0	0.5	0
2 Not chosen	0	0	0	0	0	0	1

Figure 6.14 (Design Matrix for Alternative-Specific Effects)

Note that the **Bundle_Discount** is represented in three columns, and each column captures a unique utility weight associated with the effect of value meal discount on each sandwich option.

Collapsing Separate Variables as a Generic Effect

Sometimes, you will want to take multiple independent variables involved in alternative-specific effects and collapse them into a single generic effect (to reduce the number of parameters estimated). We described this back in Chapter 5.

Let's imagine we are predicting the choice of sandwich (Deluxe Hamburger, Chicken Sandwich, or Fish Sandwich) using three independent variables:

- Deluxe_Burger_Price**
- Chicken_Sandwich_Price**
- Fish_Sandwich_Price**

To specify the separate own-effect of price of each sandwich on each sandwich choice (alternative-specific effects), we'd use the following configuration within the *Variable Codings* dialog:

Select Independent Variables			Predicting Categories of Dependent Variable		
	Independent Variables	Coding	Deluxe Hamburger Chosen	Chicken Sandwich Chosen	Fish Sandwich Chosen
1	IV2 Deluxe Burger Price	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	IV3 Chicken Sandwich Price	Linear	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	IV4 Fish Sandwich Price	Linear	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Figure 6.15 (Specifying Effect of Price on Sandwich Choices)

Clicking the **Preview Design Matrix** displays the resulting alternative-specific coding:

Preview Design Matrix								
← Respondent: 1 →								
	Task/Alternative	ASC (Deluxe Hamburger Chosen)	ASC (Chicken Sandwich Chosen)	ASC (Fish Sandwich Chosen)	IV2 Deluxe Burger Price [Linear]	IV3 Chicken Sandwich Price [Linear]	IV4 Fish Sandwich Price [Linear]	Response
1	Deluxe Hamburger Chosen	1	0	0	-0.5	0	0	0
1	Chicken Sandwich Chosen	0	1	0	0	0.167	0	0
1	Fish Sandwich Chosen	0	0	1	0	0	-0.5	0
1	Not chosen	0	0	0	0	0	0	1
2	Deluxe Hamburger Chosen	1	0	0	-0.167	0	0	0
2	Chicken Sandwich Chosen	0	1	0	0	-0.167	0	0
2	Fish Sandwich Chosen	0	0	1	0	0	0.5	0
2	Not chosen	0	0	0	0	0	0	1

Figure 6.16 (Preview Design Matrix for Alternative-Specific Own Price Effects)

Note in Figure 6.16 that we are estimating separate utility effects for each of the sandwich prices (**Deluxe_Burger_Price**, **Chicken_Sandwich_Price**, and **Fish_Sandwich_Price**) on each choice of sandwich type. But, if the effect of price on the preference for sandwich types could be assumed to be essentially equivalent, we could consider saving two degrees of freedom and collapse the three columns into a single column as a generic price effect.

To collapse separate independent variable columns into a single column, we select **Group and Collapse Variables** from the *Variable Codings* tab (see Figure 6.4). The *Collapse & Group Variables* dialog is displayed (Figure 6.17).

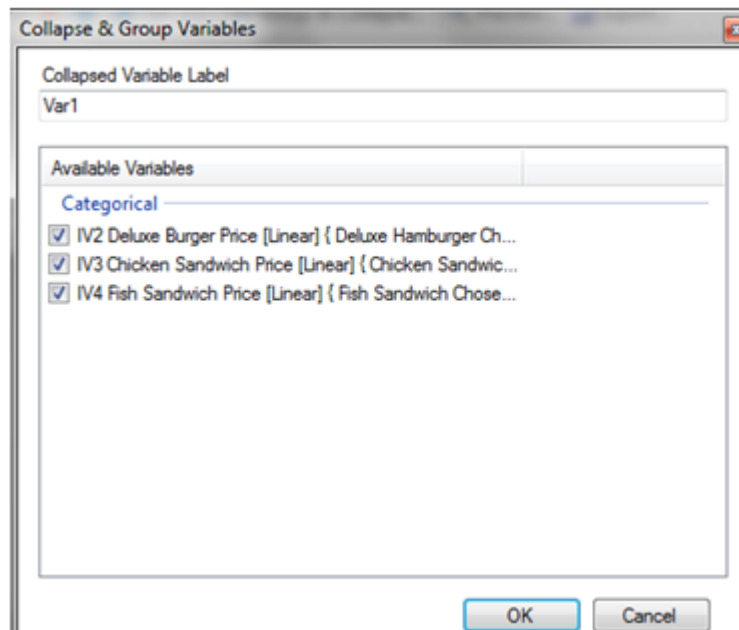


Figure 6.17 (Collapse & Group Variables)

The software suggests a name for the collapsed independent variable (Var1). To keep things simple, we'll use that label. Next, select all three variables from the list to collapse (as we've shown in Figure 6.17). Then, click **OK**.

The variables that have been grouped into a collapsed factor are shaded grey in the *Variable Codings* dialog, and a bracket with the collapsed variable name is shown at the right.

Select Independent Variables		Predicting Categories of Dependent Variable		
Independent Variables	Coding	Deluxe Hamburger Chosen	Chicken Sandwich Chosen	Fish Sandwich Chosen
1 IV2 Deluxe Burger Price	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2 IV3 Chicken Sandwich Price		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3 IV4 Fish Sandwich Price		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Figure 6.18 (Collapsed Variables in the Variable Codings Dialog)

(To un-collapse Var1, click on the bracket at the far right of the display, and a red-colored "x" appears that you can select to remove the collapsed variable relationship.)

When you click **Preview Design Matrix**, you see that the three variables have been collapsed into a single continuous variable named **Var1**:

Preview Design Matrix					
← Respondent: 1 →					
Task/Alternative	ASC (Deluxe Hamburger Chosen)	ASC (Chicken Sandwich Chosen)	ASC (Fish Sandwich Chosen)	Var1 [Linear]	Response
1 Deluxe Hamburger Chosen	1	0	0	-0.412	0
1 Chicken Sandwich Chosen	0	1	0	0.43	0
1 Fish Sandwich Chosen	0	0	1	-0.412	0
1 Not chosen	0	0	0	0	1
2 Deluxe Hamburger Chosen	1	0	0	-0.254	0
2 Chicken Sandwich Chosen	0	1	0	0.272	0
2 Fish Sandwich Chosen	0	0	1	0.061	0
2 Not chosen	0	0	0	0	1

Figure 6.19 (Collapsed Variable within Design Matrix)

Since the prices differed across the sandwiches (alternative-specific pricing), the prices are mapped to a common scale and rescaled to be zero-centered with a range of 1.0.

Collapsing variables to save degrees of freedom is generally recommended for HB analysis, especially if the separate utility weights don't differ much. For aggregate logit analysis, alternative-specific weights and larger models often perform better.

8 Analyzing MBC Data Using Aggregate Logit

8.1 Analyzing MBC Data Using Aggregate Logit

Throughout this section, we assume the reader is already familiar with standard CBC and aggregate logit analysis. Logit analysis uses a gradient search procedure to find a vector of aggregate utilities that yields a maximum likelihood fit to the data. In other words, we find a set of average utilities for the sample that predicts as accurately as possible what respondents actually chose in the questionnaire. The independent variables in MBC studies are typically the prices for the menu items, and the choices on the menu represent the dependent variables.

Logit assumes IIA (Independence of Irrelevant Alternatives) and a choice of a *single* alternative from a set of multiple alternatives. This would seem to make it inadequate for MBC where respondents can pick from 0 to many options on the menu. But, we employ some strategies to enable logit to be applied within an MBC context. Foremost, we develop a separate logit model to predict each item's choice. Thus, if there are 20 items on the menu, we can develop 20 separate logit models, each with two mutually exclusive outcomes (checked or not checked). The likelihood of selecting each item is modeled not only as a function of its own desirability and price, but as a function of other items' prices as well. Thus, the series of logit models are interrelated.

(That said, we often look for opportunities to account for multiple items' choices using a single dependent variable. For example, the choice among 3 binary items on the questionnaire could be captured using 8 categories of a single dependent variable.)

We should note that some MBC menus involve options that have three or more mutually exclusive categories from which to select. Such dependent variables are readily analyzed using logit. Other menus may feature some items that can only be chosen if *another* item on the menu has been selected first (a two-stage process). We may also use logit to model such conditional dependencies. Only the tasks where respondents have selected the "trigger" category of the dependent variable are used to estimate parameters for the conditional dependent variable.

8.2 Acknowledgments and Limitations

We do not claim that the logit equation with its IIA assumption is the best approach for all MBC studies. However, the logit model has proven robust over four decades of application within CBC and related work within the discrete choice paradigm (e.g. scanner data econometric modeling), especially when employing disaggregate analysis (i.e. Latent Class and HB). Our experience so far is that logit is robust and flexible, able to handle a wide variety of simple to complex MBC studies.

Elite practitioners and academics might employ other models that outperform our logit-based approach for certain projects and data conditions. But, our aim with this MBC software is to supply a *practical* tool that empowers skilled conjoint practitioners to do MBC studies, saves them a great deal of time in the process, is scalable to handle complex MBC problems, and produces good results each time.

We don't expect this software to handle *all* potential MBC modeling situations (just as our CBC software doesn't handle *all* potential CBC-like problems), but we hope that it addresses the vast majority of them. There may be situations in which consultants will be called upon to conduct MBC studies that are beyond the reach of this commercial software program. We encourage our MBC users to study this manual, receive training, and become skillful in using the software. When the situation exceeds the capabilities of the software, we recommend that you turn to other tools, or be ready to hire able consultants to assist you.

8.3

Testing Different Model Specifications with Aggregate Logit

Even if you plan to use HB for your final simulation models, we recommend you test preliminary specifications using MBC's aggregate logit tool. Aggregate logit is *much* faster than HB, and it provides well-known statistical tests for investigating which variables to include, and whether to specify quantitative variables as linear, log-linear, or part-worth functional forms. Counting analysis provides guidance regarding which variables and functional forms to try in aggregate logit. Once you have tested different specifications, you can employ the best models within HB estimation.

T-test

Each logit run is accompanied by a report showing the effects (utilities or coefficients) for each independent variable, the standard error, and a t-ratio. The t-ratio is equal to the effect divided by the standard error, and a magnitude of 1.96 indicates a significant parameter at 95% confidence. A t-ratio of magnitude 1.645 or greater reflects at least 90% confidence.

For part-worth coded variables, $k-1$ parameters are estimated, *where the reference level is the first level of the attribute*. The reference level is set at zero, and thus the t-ratios for levels of a part-worth coded variable test the utility differences between each level (other than the first) vs. the first level.

Each aggregate logit solution involves an overall fit statistic called log-likelihood. Log-likelihood is the sum of the natural logs of the likelihoods from each choice task. The best possible log-likelihood is zero (if each respondent's choices were perfectly predicted by the aggregate logit utilities). Generally, there are thousands of choice tasks, and the log-likelihood will be a large magnitude negative number, such as -5,258.23.

2 Log-Likelihood Test

The 2 log-likelihood test is commonly used to compare two logit models in terms of fit to the data. Let's imagine that we are debating whether to include an independent variable such as a particular cross-effect in one of our sub-models. We could run the model (a) without the cross-effect (record the log-likelihood: LL_a) and then run it again (b) including the cross-effect (again recording the log-likelihood: LL_b).

Two times the difference in log-likelihoods between the models:

$$2(LL_b - LL_a)$$

is distributed as Chi-Square, with degrees of freedom equal to the number of *additional* parameters (terms estimated) included in the second model (b). If the cross-effect is a price variable with 5 levels, then this would involve $5-1 = 4$ additional parameters to

estimate (dummy-coded, part-worth functional form). With a linear or log-linear form, the inclusion of the cross-effect adds just a single parameter to the model (DF=1). To see if the Chi-Square value obtained indicates a statistically significant improvement in the model, we may refer to a standard Chi-Square table available in most any statistical text book. Or, we may also use Excel's Chi-Square Formula:

=CHIDIST(Chi-Value, DF)

For example, let's assume that two times the difference in log-likelihood for the two models above is 10, and there are 4 degrees of freedom in the test. If we specify the following in Excel:

=CHIDIST(10,4)

Excel returns: 0.040428, which is the p-value, meaning that there is a 4.04% likelihood that an improvement in fit this large would be observed by chance. In other words, we are $100\% - 4.04\% = 95.96\%$ confident that the difference in fit would *not* have occurred by chance.

Researchers are accustomed to using 95% as the standard for determining significance, usually because they fear declaring a difference/relationship when a difference/relationship does not exist (a "false positive"). But, with MBC modeling, we are typically testing whether to include an additional variable in the model or we are comparing the difference in fit between two functional forms. Including a variable in the model that actually turns out to be worthless is actually not such a devastating thing for market simulation accuracy. It will add a little bit of time to the estimation process, and may add a little bit of noise if the variable is indeed worthless. *Therefore, we encourage you to relax the 95% confidence rule to something like 90% or 80%.* If the data suggest with 80% confidence that a parameter increases the fit of the model (and it has logical face validity, with the coefficients pointing in the right preference direction), it probably makes sense to include it in the model. You stand better odds that it will add value than not (4:1 odds, given 80% confidence). And, if it doesn't add value, it typically won't harm you much in terms of predictive accuracy.

8.4 Using the MBC Software to Perform Logit Analysis

For each Dependent Variable in your dataset you should already have specified a model (selected the Independent Variables that predict Dependent Variables) as described previously. During the model specification process, you should have selected the appropriate functional form (part-worth, linear, log-linear) and you should have previewed the design matrices to check the suitability of the coding.

Under the **Specify Models** tab, there are two sub-tabs with dialogs to be concerned about when preparing to run Logit Analysis: **Model Settings**, and **Logit Settings**.

Model Settings

Constraints: When you want to force a utility (effect) to be positive or negative, or if you want to force level 2 to have higher utility than level 1 (for a part-worth function), you specify constraints. Constraints can be useful to ensure that the model conforms to expectations, and can be used to avoid relationships that lack face validity. But, constraints almost always reduce the fit of your model (if an out-of-order relationship needs to be constrained). Sometimes, constraints can reduce the predictive validity of your simulation models, so you should be careful when imposing constraints.

Include Alternative-Specific Constants: By default, MBC software inserts a part-worth (dummy-coded) categorical variable to account for the intrinsic value of the menu options (holding prices constant). These terms are labeled "ASC" in the Preview Design Matrix and in the logit utility report. Occasionally you might need to remove the ASCs from your model, for example, with *unlabeled* designs. An unlabeled design is where the product concepts have randomized positions within the task, and there is no fixed description that identifies each item. Rather, other independent variables taking on varying levels (of brand, color, speed, etc.) fully describe the differences among concepts. The classic example of an unlabeled design is the common CBC task.

8.5 Logit Settings

Logit uses a gradient-search procedure to maximize the fit or likelihood of the data. You will rarely need to modify the default settings.

Step Size: The gradient search procedure has a default (1) that controls how quickly to "step" (modify the utilities from the previous iteration) in the direction specified by the gradient vector. Under rare occasions, you may need to decrease the step size to ensure proper convergence.



Maximum Number of Iterations: Typically, aggregate logit for MBC problems converges within about 4 to 20 iterations. If you reduce the Step Size, it will take more steps to converge, but the resulting solution might be *slightly* more precise. You should rarely need to change this setting.

Convergence Limit: The gradient search process stops when the last iteration fails to improve the fit to the data by enough to make it worthwhile to continue. The convergence limit of 0.00001 leads to a high degree of precision. Improvements in the log-likelihood beyond this degree of precision will have essentially no practical effect on your models.

Use respondent weights: If you have selected a .csv file containing weights on the *Data Files* tab, then you may use these to weight respondents differentially. Respondents with relatively larger weights will exert greater influence on the final utilities (effects). The weights are applied *during the utility estimation process* for aggregate logit.

8.6 Estimate Models

After you have specified *Model Settings* and *Logit Settings* you are ready to estimate your logit models. MBC datasets typically involve choices of multiple items. You estimate a separate logit model to predict the outcomes of each Dependent Variable. Thus, you will typically have more than one model to estimate.

When you click the *Estimate Model* tab, you see a list of Dependent Variables, with a logit  or an HB  icon displayed next to each.









Model	Progress	Status
Logit		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	
HB		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	

Figure 7.1 (The Estimate Model display)

For example, to run the Logit model to predict Dependent Variable **Ch1**, click the first *Estimate* link shown in the first row of this table. To run the Logit model to predict Dependent Variable **Ch3**, click the third *Estimate* link in the third row of this table (indicated with arrows below).









Model	Progress	Status
Logit		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	
HB		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	

Figure 7.2 (Estimate Links for Logit Runs)

After you click the *Estimate* link, estimation begins. Logit is extremely fast, and a report is almost immediately displayed to the screen:

```

Iteration 1 Log-likelihood = -2632.83840 Chi Sq = 3673.14923 RLH = 0.66477
Iteration 2 Log-likelihood = -2565.68713 Chi Sq = 3807.45177 RLH = 0.67173
Iteration 3 Log-likelihood = -2564.51340 Chi Sq = 3809.79925 RLH = 0.67185
Iteration 4 Log-likelihood = -2564.51267 Chi Sq = 3809.80070 RLH = 0.67185
Iteration 5 Log-likelihood = -2564.51267 Chi Sq = 3809.80070 RLH = 0.67185

```

*Converged after 0.20 seconds.

```

Log-likelihood for this model = -2564.51267
Log-likelihood for null model = -4469.41302

```

```

Difference = 1904.90035

```

```

Percent Certainty = 42.62082
Consistent Akaike Info Criterion = 5148.56839
Chi Square = 3809.80070
Relative Chi Square = 1904.90035

```

	Effect	Std Err	t Ratio	Variable
1	-1.85200	0.03666	-50.51534	ASC (1)
2	-0.46861	0.07721	-6.06912	Pr1 [Log-linear]

The report shows that it took just 5 iterations and 0.20 total seconds to obtain convergence. (HB runs will typically take 40,000 iterations and about 5 to 15 minutes to finish, so this is comparably fast indeed!)

The log-likelihood for this model and the null model (which assumes no information, or utility scores of zero) are given. Rather than take time here to describe each of the statistics listed in the report, we refer the interested reader to our Latent Class documentation (which may be downloaded for free from our Technical Papers library), which describes RLH, Percent Certainty, Consistent Akaike Information Criterion, Chi-Square, and Relative Chi-Square.

There are a few key statistics to pay attention to:

Log-likelihood for this model. This fit statistic is useful for comparing models that employ different variables or different functional forms of variables. Log-likelihood fits that are closer to zero are better when comparing models. If models you are comparing include a different number of parameters estimated, then you will need to apply the 2 log-likelihood test, as described earlier in this section.

Percent Certainty. Intuitively, this measure feels quite a bit like the R-squared from regression analysis. Percent Certainty indicates the percent of the way between the Null model and the perfect model (where the Null model is one that assumes flat utilities with zero information). If your Percent Certainty is near zero, then this indicates that your model is not doing a very good job at predicting respondents' choices.

Effect. This is the utility or coefficient associated with each term in your model. If using linear or log-linear price terms, then we generally expect negative signs for own-effects and positive signs for cross-effects between items that are substitutable. If using part-worth specifications, then K-1 effects are given for a K-level independent variable (resulting from

dummy-coding). The utility of the first level of the independent variable is not shown, and is set to zero. The effects of the second through Kth levels are shown (with respect to the first level set to zero). Thus, an effect of 0.1 for the second level of a part-worth coded independent variable means that it is worth 0.1 utilities more than level 1.

Standard Error. This quantifies the precision of each of the effects for the independent variables. Precision is improved when there is little to no correlation between independent variables in the design matrix. With most MBC studies, standard errors from aggregate logit should be about 0.10 or less. If your standard errors are much larger than this, you should investigate why you are obtaining relatively low precision of the utility estimates. Common problems include small sample sizes and poor experimental designs (with correlation among independent variables).

T-ratio. This is simply the Effect divided by the Standard Error. It indicates whether the Effect is significantly different from zero. T-ratios with absolute magnitude of 1.96 or greater reflect at least 95% confidence that the effect of the independent variable is non-zero. A t-ratio of magnitude 1.645 or greater reflects at least 90% confidence. You may wish to prune your models of Effects that have less than 80% significance, to avoid overfitting (but this recommendation is subject to debate and circumstances).

8.7 Managing Your Logit Runs

Each logit run you estimate is stored within your project folder in a subfolder named *Utilities*, and may be viewed and managed from the **Utility Runs** tab. The run is labeled with a combination of the Dependent Variable name, logit or HB, and the date the run was created (you may rename each run if you like).

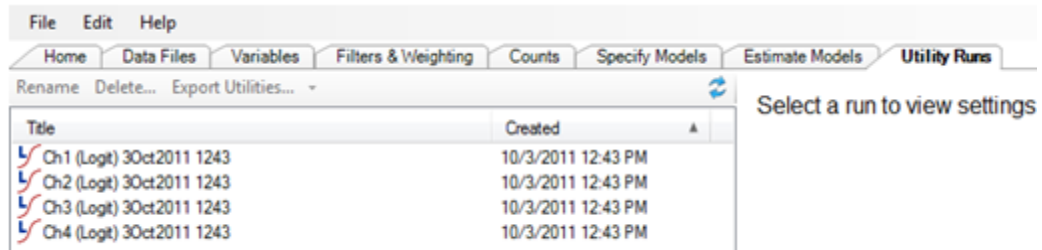


Figure 7.3 (Utility Runs tab with Logit Runs Displayed)

You can **Rename** and **Delete** runs. You can review the settings and results (average utilities, fit, standard errors, etc.) by highlighting a run in the table. You can also export the contents of a utility run to a .csv file by clicking **Export Utilities...**

When you conduct market simulations, you select a utility run to use for predicting categories of each dependent variable. Thus, you could have multiple candidate runs stored and available for using in the market simulator.

9 Analyzing MBC Data Using HB

9.1 Analyzing MBC Data Using HB

In the previous section, we illustrated how to use aggregate logit to analyze MBC studies. We recommended that you use the faster logit routine to investigate different model specifications prior to running the longer HB estimation. We also acknowledged that a logit-based approach may not be the best overall way to analyze MBC data. In this section, we illustrate how hierarchical Bayes (HB) may be used to analyze MBC data. The HB approach we use is a disaggregate, logit-based approach that estimates parameters at the individual level. It utilizes the same algorithms as our popular CBC/HB software program.

Researchers familiar with CBC know that the literature is replete with evidence that HB is superior to aggregate logit for conjoint analysis. Our research with MBC so far suggests that HB may have a *slight* edge over aggregate logit with MBC data, but that both approaches lead to good results. Given our preliminary findings, it would be tempting to opt for aggregate logit as the utility files are tiny, and the market simulations using aggregate are therefore extremely fast. But, HB has two key benefits for MBC studies:

1. If market simulations are to be done by different market segments (banner points), the availability of individual-level data means we can simply "slice and dice" the sample during simulations and easily report the results by segment. With aggregate logit, one would have to go back and re-estimate separate models for different market segments, then conduct separate simulations for each segment.
2. HB utilities are required for [combinatorial choice simulations](#). Combinatorial simulations predict the most likely *combinations* of items that are picked together, rather than just describing what percent of the sample is predicted to pick each menu item.

To summarize, we recommend using aggregate logit to investigate different model specifications (i.e. testing which independent variables to include, and selecting functional form), but using HB results for conducting market simulations. But, if you are not going to make separate predictions by segment, and if you have no need for combinatorial choice simulations, then aggregate logit will often serve you quite well for MBC analysis and simulations.

(Note: The model specification most appropriate for logit analysis is not necessarily the specification that works best under HB analysis. Logit analysis often benefits from larger models with more terms, whereas HB often benefits from more parsimonious model specification. One often selects generic terms, or linear or log-linear specifications when using HB.)

9.2 Using the MBC Software to Perform HB Analysis

For each Dependent Variable in your dataset you should already have specified a model (selected the Independent Variables that predict Dependent Variables). During the model specification process, you should have selected the appropriate functional form (part-worth, linear, log-linear) and you should have previewed the design matrices to check the suitability of the coding.

Under the ***Specify Models*** tab, there are two sub-tabs with dialogs to be concerned about when preparing to run HB Analysis: ***Model Settings***, and ***HB Settings***.

Model Settings

Constraints: When you want to force a utility (effect) to be positive or negative, or if you want to force level 2 to have higher utility than level 1 (for a part-worth function), you specify constraints. Constraints can be useful to ensure that the model conforms to expectations, and can be used to avoid relationships that lack face validity. But, constraints frequently reduce the fit of your model (if an out-of-order relationship needs to be constrained). Sometimes, constraints can reduce the predictive validity of your simulation models, so you should be careful when imposing constraints.

Include Alternative-Specific Constants: By default, MBC software inserts a part-worth (dummy-coded) categorical variable to account for the intrinsic value of the menu options (holding prices constant). These terms are labeled "ASC" in the Preview Design Matrix and in the logit utility report. Occasionally you might need to remove the ASCs from your model, for example, with *unlabeled* designs. An unlabeled design is where the product concepts have randomized positions within the task, and there is no fixed description that identifies each item. Rather, other independent variables taking on varying levels (of brand, color, speed, etc.) fully describe the differences among concepts. The classic example of an unlabeled design is the common CBC task.

9.3 HB Settings

HB involves a number of settings that govern the computation:

Number of iterations before using results: HB usually requires thousands of iterations to obtain good estimates of respondent utilities and of the population means and covariances. The preliminary iterations are the "burn-in" iterations that are not actually used in the final utility run, but are undertaken as a means to obtain convergence. Typically, 20000 or more iterations should be performed before convergence is assumed in MBC. Only after a run is completed can one examine the history of part-worths estimated across the iterations to assess if the model indeed has converged to a relatively stable solution. The part-worth utilities should oscillate randomly around their true values, with no indication of trend. If convergence is not obtained, then you will want to re-run the HB estimation and specify a larger number of initial iterations. (*Note: larger models, particularly those that use part-worth coded price parameters can take a long time to converge, sometimes more than the default settings in the software.*)

Number of draws to be used for each respondent: Each iteration, after convergence is assumed, leads to a candidate set of utilities for each respondent. With the MBC software, we have chosen to provide the *option* to simulate respondent choices based on these individual draws. Given current hardware, we have found that using more than about 200 draws per respondent can result in lengthy market simulation runs, and the extra time required to employ more draws is probably not worth the extra effort. Therefore, we suggest a default number of draws per respondent of 200.

Skip factor for saving random draws: The skip factor is a way of compensating for the fact that successive draws of the betas are not independent. A skip factor of k means that results will only be used for each k th iteration. It is useful to use every k th draw for each respondent to capture more completely the distribution of draws that characterize each respondent's preferences. The utilities can oscillate in rather lengthy wave-like patterns, with cycles measured in the thousands or tens of thousands of iterations, so we recommend using a large skip factor for saving draws. Convergence in MBC may sometimes take longer than standard CBC problems, so we recommend saving periodic iterations over the duration of about 20000 or more candidate iterations (after obtaining convergence). By default we suggest skipping every 100th iteration for HB runs within MBC software (note that the recommendations within CBC/HB software are different, since we typically focus on point estimates rather than draws within CBC/HB).

Skip factor for displaying in graph: Indicates the skip factor employed when deciding for how many draws the results should be displayed on the graph.

Skip factor for printing in log file: This controls the amount of detail that is saved in the `studname.log` file to record the history of the iterations.

Prior degrees of freedom: This value is the additional degrees of freedom for the prior covariance matrix (not including the number of parameters to be estimated), and can be set from 2 to 100000. The higher the value, the greater the influence of the prior variance



and more data are needed to change that prior. The scaling for degrees of freedom is relative to the sample size. If you use 50 and you only have 100 subjects, then the prior will have a big impact on the results. If you have 1000 subjects, you will get about the same result if you use a prior of 5 or 50. As an example of an extreme case, with 100 respondents and a prior variance of 0.1 with prior degrees of freedom set to the number of parameters estimated plus 50, each respondent's resulting part worths will vary relatively little from the population means. We urge users to be careful when setting the prior degrees of freedom, as large values (relative to sample size) can make the prior exert considerable influence on the results.

Prior variance: The default is 1 for the prior variance for each parameter, but users can modify this value. You can specify any value from 0.1 to 99. Increasing the prior variance tends to place more weight on fitting each individual's data, and places less emphasis on "borrowing" information from the population parameters. The resulting posterior estimates are relatively insensitive to the prior variance, except a) when there is very little information available within the unit of analysis relative to the number of estimated parameters, and b) the prior degrees of freedom for the covariance matrix (described above) is relatively large. (Note that CBC/HB software uses a default prior variance of 1. We also employ a prior variance of 1 in MBC studies.

Note: we use the prior covariance matrix as recommended by Professor Peter Lenk and described in the CBC/HB software documentation, Appendix G. Part-worth and ASC parameters are dummy-coded, and follow Lenk's recommended prior covariance structure.

9.4 Estimate Models

After you have specified *Model Settings* and *HB Settings* you are ready to estimate your HB models. MBC datasets typically involve choices of multiple items. You estimate a separate HB model to predict the outcomes of each Dependent Variable. Thus, you will typically have more than one model to estimate.

When you click the *Estimate Model* tab, you see a list of Dependent Variables, with a logit  or an HB  icon displayed next to each.









Model	Progress	Status
Logit		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	
HB		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	

Figure 8.1 (The Estimate Model Display)

For example, to run the HB model to predict Dependent Variable **Ch1**, click the first *Estimate* link shown in the HB section of this table. To run the HB model to predict Dependent Variable **Ch3**, click the third *Estimate* link in the HB section of this table (indicated with arrows below).











Model	Progress	Status
Logit		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	
HB		
 Ch1	Estimate	
 Ch2	Estimate	
 Ch3	Estimate	
 Ch4	Estimate	

Figure 8.2 (Estimate Links for HB Runs)

After you click the *Estimate* link, the estimation begins. Multiple estimations can be run simultaneously (by clicking on multiple *Estimate* links), and this will take advantage of your hardware's multi-core processor.

Warning: When multiple HB runs are running simultaneously (requires multi-core preprocessors), it is quite possible for some hardware configurations to experience higher than normal processor temperature, sometimes leading to a sudden shutdown of your system or system freeze. This is more likely to occur on laptops, that typically have limited ventilation and smaller heat sinks than towers/desktops. On some systems, there are power options or battery saving options (in the case of laptops) that can help you slow the processor rate. Consult your hardware manuals and IT manager for more information.

HB estimation takes *much* longer than logit estimation, and it typically will take from 5 to 15 minutes to complete an HB estimation for each of your models.

Model	Progress	Status
Logit		
Ch1	Estimate	Finished at 12:43 PM
Ch2	Estimate	Finished at 12:43 PM
Ch3	Estimate	Finished at 12:43 PM
Ch4	Estimate	Finished at 12:43 PM
HB		
Ch1	<div style="width: 50%; background-color: green;"></div>	Cancel iteration 11870 of 40000
Ch2	<div style="width: 50%; background-color: green;"></div>	Cancel iteration 11660 of 40000
Ch3	<div style="width: 50%; background-color: green;"></div>	Cancel iteration 11729 of 40000
Ch4	<div style="width: 50%; background-color: green;"></div>	Cancel iteration 11740 of 40000

Figure 8.3 (Estimate Models Display with Four HB Runs in the Queue)

In the example above, four HB runs are currently simultaneously running. A *Cancel* link is provided to cancel any run. Previously, four logit models were completed (one for each dependent variable).

There are a few key statistics to pay attention to:

Pct. Cert. and *RLH*. These two measures are derived from the same likelihood computation, so they capture the same thing, but on a different scale. The RLH is the "Root Likelihood" and is the geometric mean of the likelihoods across all tasks and all respondents. Thus an RLH of .50 means that the logit model predicts individual choices at a 50% likelihood. However, if there were only two alternatives in the model (chosen vs. not chosen), then this prediction rate would be no better than chance, since the likelihood of a model with no information would typically be $\frac{1}{2}$ or 0.5. Percent Certainty indicates the percent of the way between the Null model and the perfect model (where the Null model is one that assumes flat utilities with zero information). If your Percent Certainty is near zero, then this indicates that your model is not doing a very good job at predicting respondents' choices.

The Pct. Cert. and RLH measures convey essentially the same information, and both are good indicators of goodness of fit of the model to the data. The choice between them is a matter of personal preference.

Avg Variance and *Parameter RMS*: The final two statistics, "Avg Variance" and "Parameter RMS," are also indicators of goodness of fit, though less directly so. With a logit model the scaling of the part worths depends on goodness of fit: the better the fit, the larger the

estimated parameters. Thus, the absolute magnitude of the parameter estimates can be used as an indirect indicator of fit. "Avg Variance" is the average of the current estimate of the variances of part worths, across respondents. "Parameter RMS" is the root mean square of all part worth estimates, across all part worths and over all respondents. As iterations progress, all four values (Pct. Cert., RLH, Avg Variance, and Parameter RMS) tend to increase for a while and then level off, thereafter oscillating randomly around their final values. Their failure to increase may be taken as evidence of convergence. However, there is no good way to identify convergence until long after it has occurred. For this reason we suggest planning a large number of initial iterations, such as 10,000 or more, and then examining retrospectively whether these four measures have been stable for the last several thousand iterations.

Viewing the history of Mean Betas across the burn-in and used iterations is one of the best way to assess how well the model has converged. In the graphic below, the first 10000 iterations were initial/burn-in iterations, and the second 10000 (in the white region at the right) are the used iterations. Although there is some oscillation in the betas, there is certainly no remaining trend after the first 1000 or so iterations have been completed.

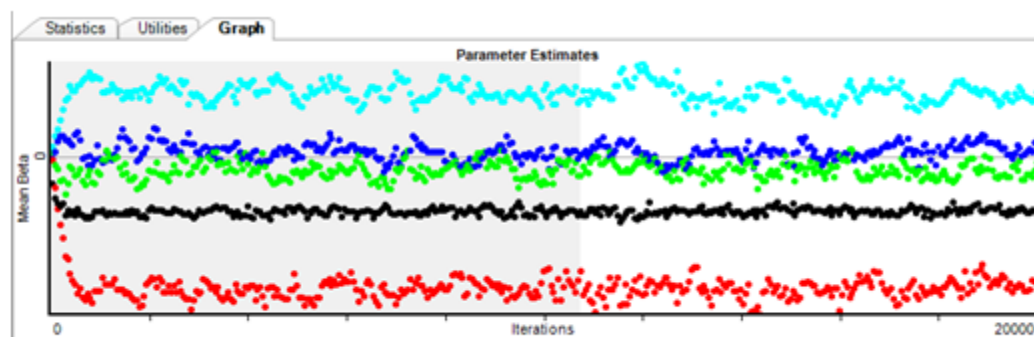
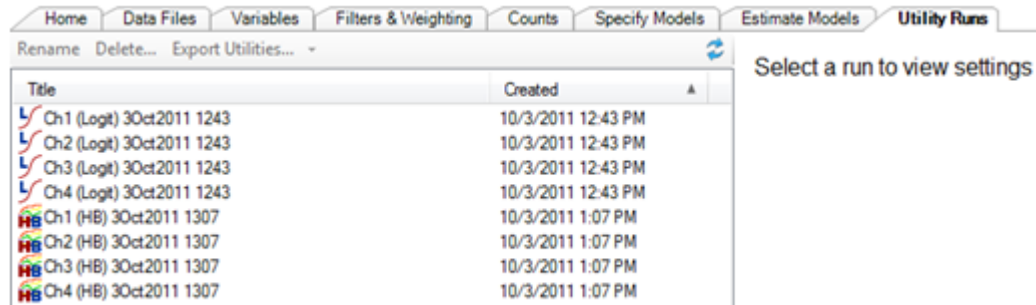


Figure 8.4 (Graph of Mean Beta History)

A graph is available for each of the runs you have run in the queue.

9.5 Managing Your HB Runs

Each HB run you estimate is stored within your project folder in a subfolder named *Utilities*, and may be viewed and managed from the **Utility Runs** tab. The run is labeled with a combination of the Dependent Variable name, logit or HB, and the date the run was created (you may rename each run if you like).



Title	Created
Ch1 (Logit) 30Oct2011 1243	10/3/2011 12:43 PM
Ch2 (Logit) 30Oct2011 1243	10/3/2011 12:43 PM
Ch3 (Logit) 30Oct2011 1243	10/3/2011 12:43 PM
Ch4 (Logit) 30Oct2011 1243	10/3/2011 12:43 PM
Ch1 (HB) 30Oct2011 1307	10/3/2011 1:07 PM
Ch2 (HB) 30Oct2011 1307	10/3/2011 1:07 PM
Ch3 (HB) 30Oct2011 1307	10/3/2011 1:07 PM
Ch4 (HB) 30Oct2011 1307	10/3/2011 1:07 PM

Figure 8.5 (Utility Runs Display with Logit and HB Runs)

You can **Rename** and **Delete** runs. You can review the settings and results (average utilities, fit, variance statistics, etc.) by highlighting a run in the table. You can also export the contents of a utility run to a .csv file by clicking **Export Utilities...**

When you conduct market simulations, you select a utility run to use for predicting categories of each dependent variable. Thus, you could have multiple candidate runs stored and available for using in the market simulator.

10 Simulating Respondent Choices

10.1 Simulating Respondent Choices

The previous two sections dealt with building models (aggregate logit and HB) to predict respondent choices. An MBC study typically involves multiple models (sub-models), where each sub-model predicts whether a given option on the menu was selected. Those sub-models estimate utility coefficients that characterize the preferences for menu items and the utilities of other aspects that vary on the menu, such as prices. Given those utility coefficients, we can use the series of sub-models to predict which options respondents will choose on the menu given a specific menu scenario. This process is typically referred to as *market simulation*, but it goes by other names as well including *what-if analysis* or just *simulations*.

10.2 Aggregate vs. Individual-Level Predictions

Market simulations may be done either using utilities estimated via aggregate logit or individual-level HB. If using aggregate logit there is just a single vector of coefficients for each sub-model that characterizes the average preferences for the sample. In effect, aggregate logit treats the data as if all respondent choices had been made by a single respondent who had taken a very long survey. Because there is just a single vector of utilities for each sub-model, the data storage requirements are miniscule, and the simulation process works extremely fast. The logit rule (also known in other Sawtooth Software documentation as *Share of Preference*) is used within each sub-model to estimate the likelihood that the sample would select each menu option.

If using HB analysis, MBC software stores 200 draws per respondent as well as point estimates (the average of the draws for each respondent). So, if there are 1000 respondents, there will be 200,000 vectors of utilities for each sub-model. As when using aggregate logit, the logit rule is applied, but we do it either for each point estimate (once per respondent) or each draw (200 times per respondent) to predict the likelihood of selecting items on the menu. But, given our example, the data storage requirement is 200,000x larger than aggregate logit, and 200,000 predictions are made (if choosing to simulate using the draws) rather than just 1. The results are averaged across all 200,000 predictions to summarize the probabilities of choice for the sample.

With simulations from HB, we may also report standard errors for the predicted probabilities of choice. We can also report choice probabilities for each respondent. Neither of these options is available through aggregate logit.

10.3 Market Simulation Math

For each of the independent variables included in the dataset, the researcher specifies which level (or value in the case of quantitative variable) characterizes the specific market scenario. With MBC studies, the independent variables often refer to the prices for each option (though other independent variables are possible, such as availability effects, or other promotion factors).

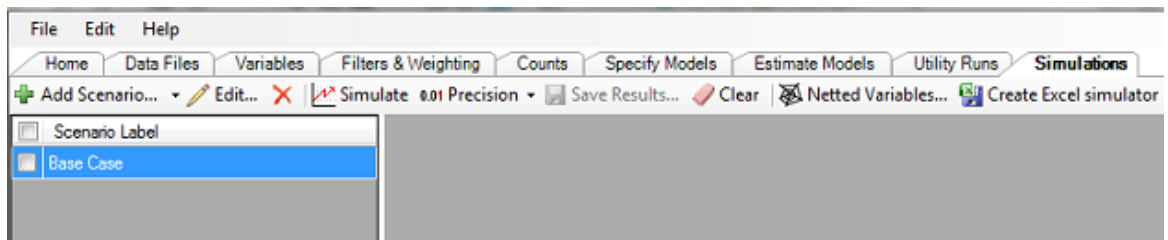
During the model-building process, you specified which independent variables to include in each sub-model and how they were to be coded (part-worth, linear, log-linear). The simulation process simply repeats the same coding procedure (as described earlier) to build an independent variable matrix representing the levels within the specific market scenario. For each row of the independent variable matrix, each element is simply multiplied by the corresponding coefficient in the vector of utilities within the sub-model, and the sum of those products reflects the total utility of that alternative. For binary alternatives (chosen or not chosen), there are two rows in the independent variable matrix, and we compute the utility of the menu option and the utility of not selecting the menu option (the constant or "None" alternative). The antilog of the total utilities is proportional to choice within the set. This is the logit rule, and we use it to predict the likelihood that items will be selected on the menu.

If selecting a dependent variable is conditional on the choice of a different dependent variable, then an additional step is required. As an example, perhaps selecting an item *a la carte* is only an option if the respondent has first rejected the value meal bundles. In that case, the likelihood of the respondent *not* selecting a value meal is multiplied by the likelihood of the respondent selecting each item *a la carte*. (Recall that the sub-model that predicts choice of the *a la carte* item has been built only using the choice tasks where the conditional dependent variable was chosen. Therefore, the likelihood of the *a la carte* selection is the likelihood of its choice *given* that the conditional dependent variable has been chosen.)

There is another exception that should be described. Perhaps an item is only able to be selected if it is available on the menu (availability design). The independent variable signals whether the option is available or not. If the independent variable (which must be specified as part of the market scenario) indicates that the option is not available, then the choice likelihood is simply set to zero.

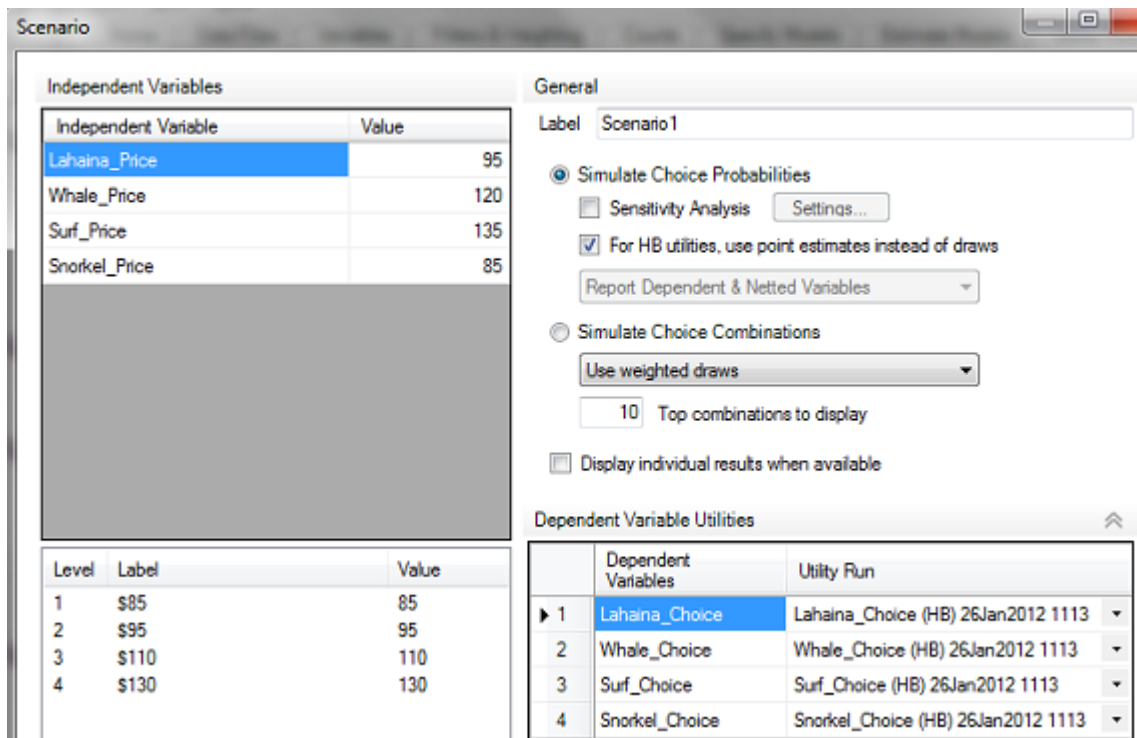
10.4 The Simulations Tab

After you have estimated logit and/or HB models, you are ready to simulate respondent choices to specific menu scenarios using the Simulation tab (the market simulator, also known as the what-if simulator).



When you first click the Simulations tab, a Scenarios panel is shown (currently blank) along with the **Add Scenario...**, **Edit...**, **X (Remove)**, **Simulate**, **Precision**, **Save Results...**, **Clear**, **Netted Variables...**, and **Create Excel Simulator** buttons.

Click the **Add Scenario...** button to add a Simulation Scenario. A Simulation Scenario is a specific menu situation (described by levels/prices of your independent variables). The Scenario dialog is displayed (Figure 9.2).



The Scenario dialog has a number of settings:

Simulate Choice Probabilities: Choosing this option tells the MBC software to simulate what percent of the respondents would select each category of each dependent variable included in the simulations. It returns a Choice Probability across the sample for each

dependent variable, based on the levels (e.g. prices) you have specified for the independent variables.

Sensitivity Analysis: Allows you to run multiple simulations in automated mode for generating demand curve charts.

Simulate Choice Combinations (Combinatorial Simulations): Selecting this option tells the MBC software to simulate the specific combinations of choices predicted to be selected by each respondent, given the levels you have specified for the independent variables.

Top combinations to display and Population weighting: These settings affect the Combinatorial Simulations.

Save individual results: If you are using HB runs to predict respondent choices, you can ask the simulator to display those results by respondent in the report and save the choice probabilities of menu selections for each respondent to a .csv file.

Independent Variables: These are the levels of the independent variables (typically the prices) that you specify for the Simulation Scenario. You can specify values such as 1, 2, 3, and 4 (as in this example). You may also specify values between levels, such as 1.3 or 3.75, and MBC software will perform interpolation. Values outside the valid range (extrapolation) are not permitted.

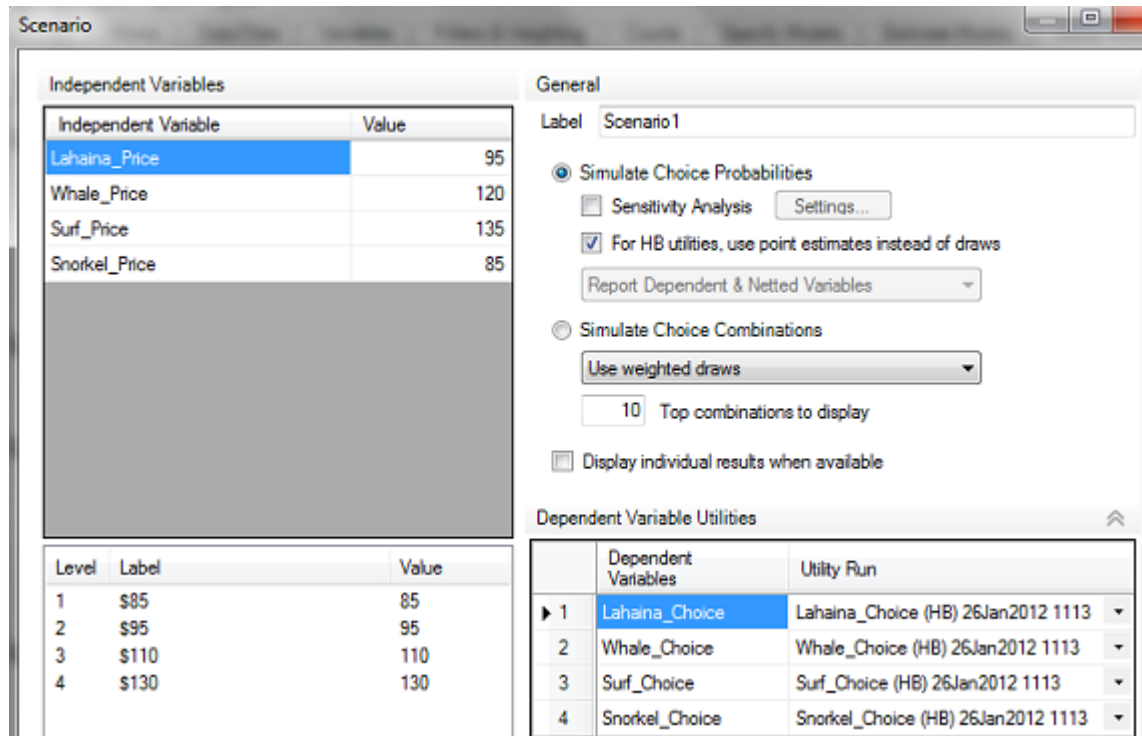
Dependent Variable Utilities: For each dependent variable you want to include in the market simulation, you need to select an available utility run to use. Either logit or HB runs may be selected. (For combinatorial simulations, HB runs are required).

Respondent Filters: If you have read-in a *Demographics* file (from the **Data Files** tab) and are using HB utility runs (containing individual-level data), you can select which respondents to include in the simulation run.

Respondent Weights: If you have read-in a *Demographics* file (from the **Data Files** tab) and are using HB utility runs (containing individual-level data), you can specify which variable in that file should be applied as respondent weights.

Exponent: If you are using the *Simulate Choice Probabilities* simulation method, you can set the Scale Factor multiplier (Exponent) to use. The default scale is 1.0. Values greater than 1.0 cause predicted choice probabilities to become more extreme. Values less than 1.0 cause predicted choice probabilities to become less extreme. The Exponent has no effect on the *Simulate Choice Combinations* approach.

Figure 9.2 below shows a simulation run with specific prices selected for the four menu items, and HB runs selected for use in predicting each of the dependent variable outcomes.



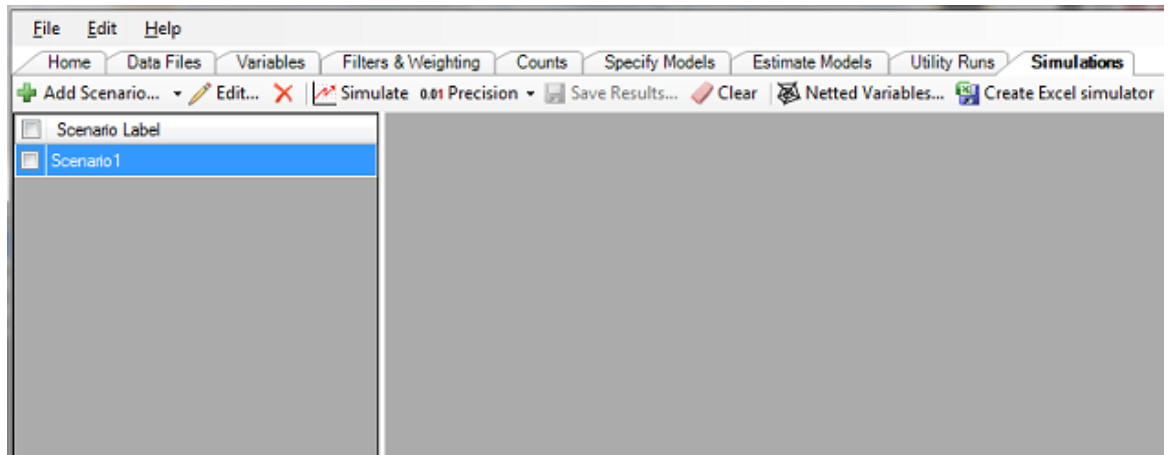
In this example, variables **Lahaina_Price** (price for menu item 1) through **Snorkel_Price** (price for menu item 4) have been specified as:

Lahaina_Price = 95
 Whale_Price = 120
 Surf_Price = 135
 Snorkel_Price = 85

You may specify prices in-between categorical prices actually used in the survey. This is called *interpolation*.

For each of the Dependent Variables, the user has selected which run to use when predicting respondent choices. Because we are requesting individual-level results to file, we must select HB runs. The drop-down controls are used to select available runs as appearing under the *Utility Runs* tab.

Click **Save** to save the run and return to the *Simulations* tab.



Once you return to the *Simulations* tab, you see the Scenario you just created saved in the list (you can have multiple scenarios saved on this dialog). To simulate respondent choices for this scenario, click the **Simulate** button at the top of the dialog. After typically a minute or so of processing, the results are displayed to the screen:

The screenshot shows the same 'Simulations' dialog box, but now the 'Simulate' button is highlighted, and the right side of the dialog displays a table of results. The table has four columns labeled A, B, C, and D. The results are organized into sections: 'Independent Variable Specification' (rows 11-15), 'Simulation Results:' (row 17), and three choice categories: 'Lahaina_Choice' (rows 19-23), 'Whale_Choice' (rows 24-28), and 'Surf_Choice' (rows 29-30). Each choice category shows 'Share' and 'Std Err' values for 'Chosen' and 'Not chosen' states.

	A	B	C	D
11	Independent Variable Specification:			
12	Lahaina_Price	95		
13	Whale_Price	120		
14	Surf_Price	135		
15	Snorkel_Price	85		
16				
17	Simulation Results:			
18				
19	Lahaina_Choice	Share		Std Err
20				
21	Lahaina Chosen		0.3506	0.0035
22	Not chosen		0.6494	0.0035
23				
24	Whale_Choice	Share		Std Err
25				
26	Whale Chosen		0.2640	0.0033
27	Not chosen		0.7360	0.0033
28				
29	Surf_Choice	Share		Std Err
30				

The predicted probability that the sample would select each category of the dependent variables is shown on the screen, with the standard errors for each. For example, Lahaina Chosen would be selected 35% of the time.

To see predictions for each respondent (given as probabilities of choice, according to the logit rule), click the *Individual Results* tab at the bottom of the display. (Note, you must have clicked the option: *Save Individual Results* when you edit the scenario to get individual-level predictions.)

	A	B	C	D	E	F	G	H
1	Respondent	Lahaina Chosen	Not chosen	Whale Chosen	Not chosen	Surf Chosen	Not chosen	Snorkel Chose
2	1	0.3126	0.6874	0.2046	0.7954	0.4905	0.5095	0.47
3	2	0.2941	0.7059	0.2391	0.7609	0.5142	0.4858	0.46
4	3	0.3633	0.6367	0.3321	0.6679	0.3647	0.6353	0.31
5	4	0.2742	0.7258	0.2490	0.7510	0.5786	0.4214	0.45
6	5	0.3431	0.6569	0.1958	0.8042	0.4248	0.5752	0.41
7	6	0.4348	0.5652	0.2384	0.7616	0.3743	0.6257	0.45
8	7	0.3086	0.6914	0.2289	0.7711	0.3869	0.6131	0.44
9	8	0.3391	0.6609	0.2747	0.7253	0.4033	0.5967	0.53
10	9	0.3039	0.6961	0.2365	0.7635	0.3664	0.6336	0.25
11	10	0.3320	0.6680	0.2181	0.7819	0.3828	0.6172	0.37
12	11	0.4636	0.5364	0.3016	0.6984	0.3133	0.6867	0.44

Figure 9.5 (Individual Prediction Results Shown)

Results are displayed for each respondent, indicating the predicted probability that each option on the menu would be selected, given the prices specified in the simulation scenario.

Create Excel Simulator...

Click this to create an Excel simulator for standard choice probability simulations (not combinatorial simulations). A .xlsx file is created, with macros enabled. The simulator has multiple sheets, which probably should be hidden when you deliver the file to your client. Most users will add their own attractive front sheet that provides a user interface for their client, and a place to display results and/or graphs/charts. That customized front sheet will interact via linked formulas with the supporting sheets that are automatically written out to the .xlsx file.

Click [Here](#) for more information.

10.5 Simulator Validation

Researchers should be concerned about how well their choice models can predict new scenarios, and especially choices made by a different set of people than used as respondents. It is common to include holdout scenarios, but probably less common (and certainly more expensive!) to include holdout respondents in conjoint-related studies, including MBC. We recommend including a holdout task or two in your MBC studies, which you can use to compare the results of different models and verify that your market simulation models are working well to predict choices.

One source for checking the quality of your simulator that you should not overlook is the Counts data (assuming you have used a good-quality randomized design). You should compare the own- and self-elasticity curves from Counting analysis to the curves that may be generated using sensitivity analysis within the simulator. When the two sets of curves are in good agreement, this is another indication that your models are working well to predict choice behavior.

10.6 Netting Dependent Variables

This dialog lets you create *netted variables*: new variables for reporting purposes that simply sum across categories of dependent variables. Netted variables are particularly useful within the simulator when you are using the recommended approach of collapsing dependent variables via [combinatorial coding](#). (Combinatorial coding leads to fewer and often more accurate models.)

Consider the case where a combinatorial dependent variable with eight categories has been coded to represent binary choices across three separate dependent variables. The three original variables were:

1. Hamburger (1=choose; 2=not choose)
2. Fries (1=choose; 2=not choose)
3. Drink (1=choose; 2=not choose)

And, the new combinatorial dependent variable (recoded in the data file and being used in modeling to stand in place of the three binary variables) has eight categories:

- 1 = Hamburger & Fries & Drink
- 2 = Hamburger & Fries
- 3 = Hamburger & Drink
- 4 = Fries & Drink
- 5 = Hamburger
- 6 = Fries
- 7 = Drink
- 8 = Nothing (off state)

The probabilities of choice across these eight categories sum to 1.0.

Three new netted variables for reporting purposes can be created for use within the simulator based on outcomes of the 8-category combinatorial dependent variable.

Hamburger = category1+category2+category3+category5
Fries = category1+category2+category4+category6
Drink = category1+category3+category4+category7

If you are interested in having the simulator tell you what percent of respondents are projected to choose the Hamburger, then it is much easier to refer to the netted variable that directly reports that information. Otherwise, you'll get eight separate probabilities for the eight categories, and you'd need to add across four categories to report the percent expected to pick hamburger.

Note: The [Excel simulator](#) option doesn't move your netted variables to the simulator spreadsheet. But, you can easily create new cells with Excel formulas that sum across multiple categories in the Front Page of your Excel simulator.

10.7 Sensitivity Analysis

Sensitivity Analysis allows you to run dozens or hundreds of simulations in an automated mode, typically with the goal of developing demand curves. It saves you a great deal of time, because you don't have to manually set up and run each simulation separately.

For example, you can develop a demand curve for how a particular price affects choice of a particular item on the menu. To develop this curve, you specify that Sensitivity Analysis should compute the likelihood of respondents choosing this item, given each price point of interest.

You can ask sensitivity analysis to simulate across the price levels included in your design, or even for price points in between those levels (interpolation).

Note: *Combinatorial Simulations* are not supported.

Example #1, One Demand Curve

Consider a hypothetical MBC study where four activities on the island of Maui were offered to tourists (each at varying prices):

Shopping in Lahaina:	\$85, \$95, \$110, \$130
Whale Watching:	\$100, \$110, \$120, \$140
Surfing Lessons:	\$120, \$130, \$140, \$160
Snorkeling:	\$80, \$90, \$100, \$120

Respondents could choose which (up to 4) activities to purchase.

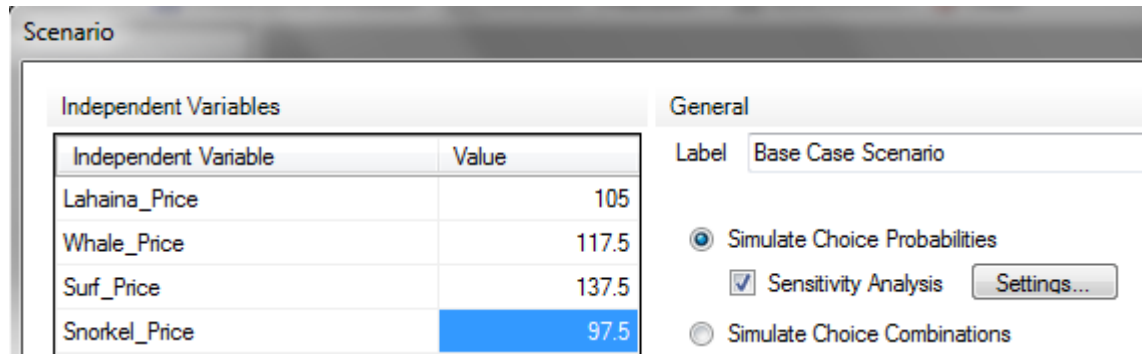
After you have built your logit or HB models for this menu, you should specify a simulation scenario. When conducting Sensitivity Analysis it is typical (but not required) to specify a base case scenario featuring each item on the menu at its middle (or average) price.

Base Case Scenario

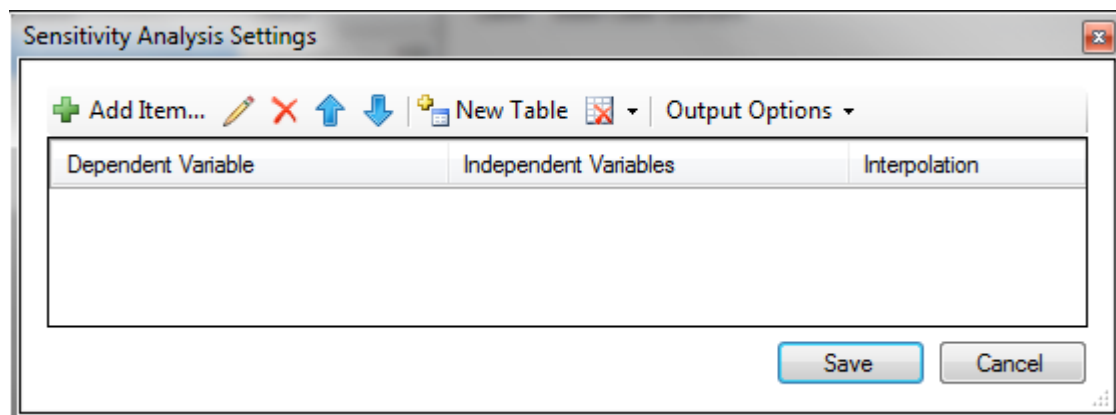
(items at average prices:)

Shopping in Lahaina:	\$105
Whale Watching:	\$117.50
Surfing Lessons:	\$137.50
Snorkeling:	\$97.5

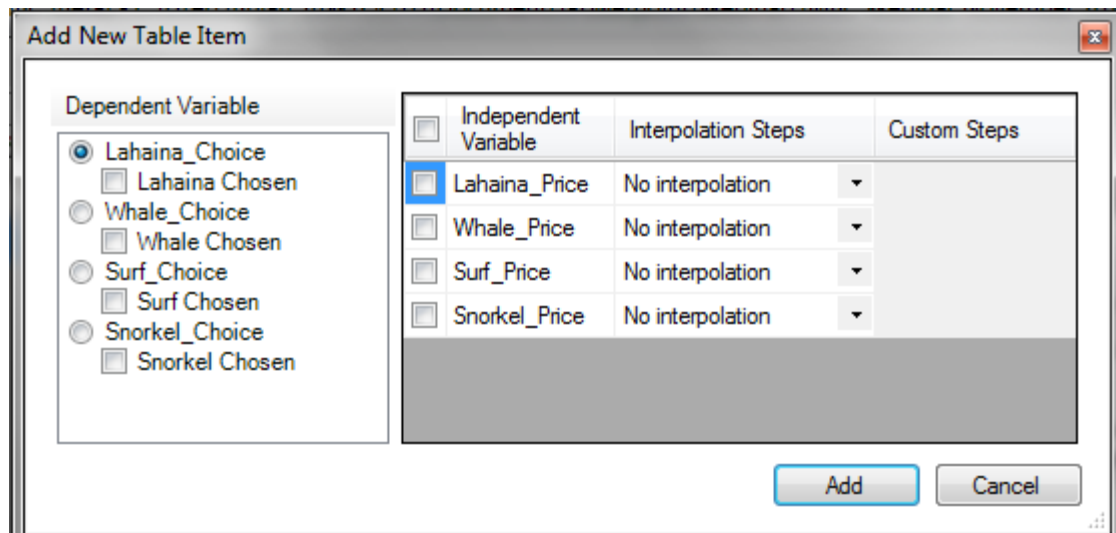
From the **Simulations** tab, you next *Add a Scenario* and on the *Scenario* dialog specify the base case prices for the items. Next, you click the *Sensitivity Analysis* checkbox as shown below:



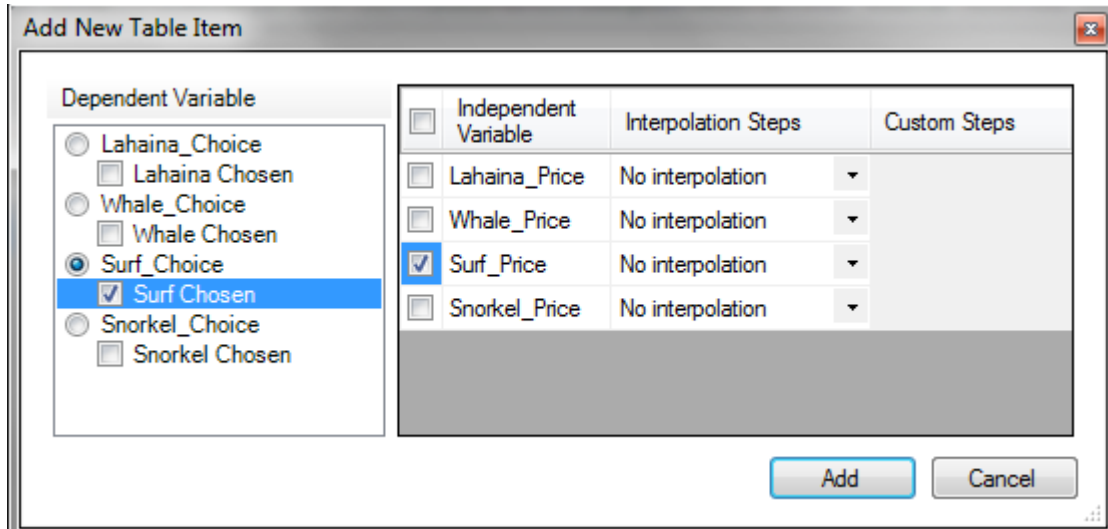
Next, you click the **Settings...** button that is next to the *Sensitivity Analysis* checkbox. The following is displayed:



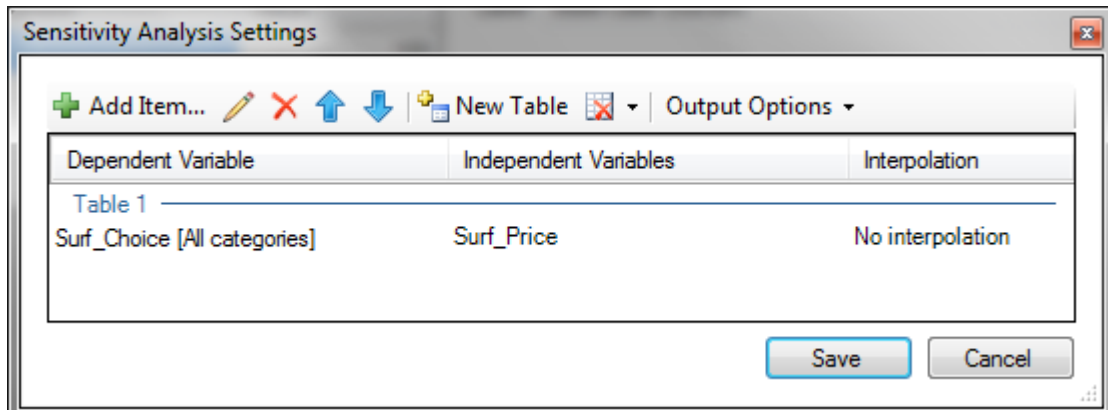
Click the **Add Item...** icon in the upper-left to add a sensitivity analysis specification.



A table of results will end up being reported, so the MBC software asks you to add a new "item" to that table of results. Because we want to see how choosing surfing lessons is affected by the price for surfing lessons, we select **Dependent Variable = Surf Chosen** along with **Independent Variable = Surf_Price**, as shown below:



If we wanted to specify that prices in between the four prices included in the experimental design should be simulated, we would specify *Interpolation Steps*. For this example, we won't do that. Clicking the **Add** button adds this sensitivity item to the table of results that will be produced:

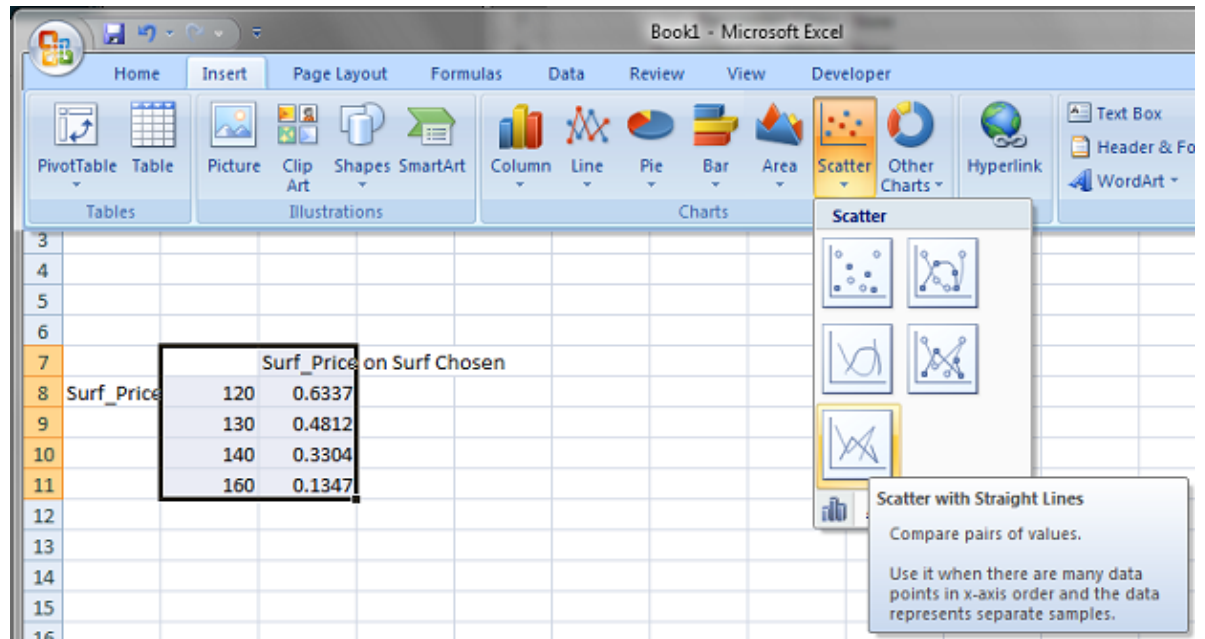


Click **Save** to return to the *Scenario* dialog. Then, click **Save** again to return to the *Simulations* tab. When you click the *Simulate* button, four separate simulations are run (one for each price point), and the results are shown:

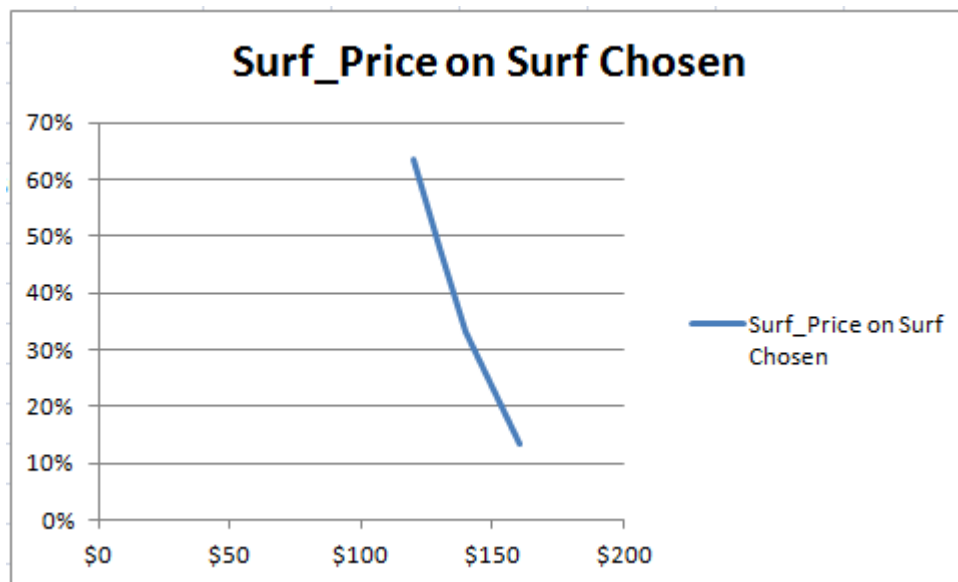
Table 1		
		Surf_Price on Surf Chosen
Surf_Price	120	0.6337
	130	0.4812
	140	0.3304
	160	0.1347

This table shows that for price point "120 dollars" the likelihood of choosing Surf Lessons is 0.6337, etc.

You can paste these results into an Excel Spreadsheet, to create a demand curve. Within Excel below, we have highlighted a portion of the results table (as shown), clicked *Insert + Scatter Plot* and chosen "Scatter with Straight Lines":



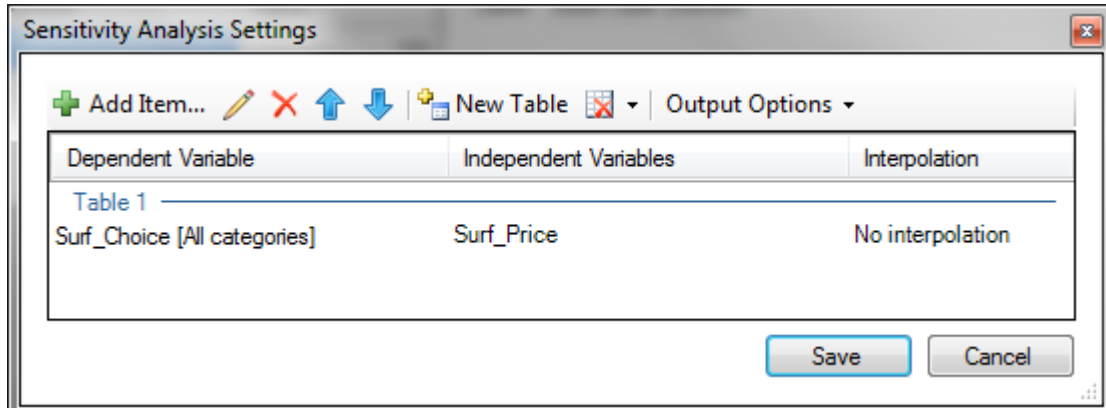
The result is then plotted within Excel:



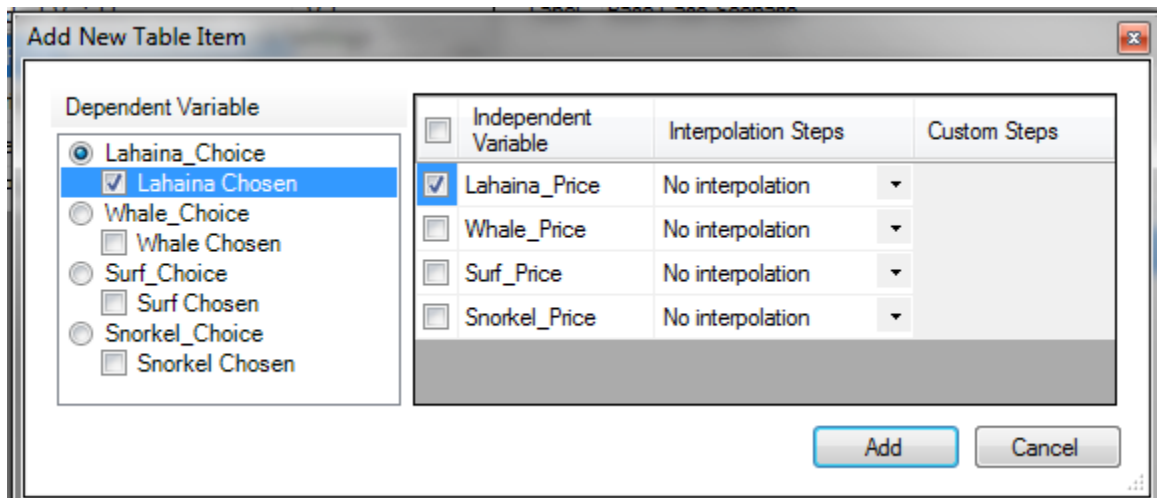
Example 2, Multiple Demand Curves

In this example, we add more items to the Sensitivity Table from the previous example. In addition to the demand curve for Surfing Lessons, we want to plot (on the same chart) the demand curves for the other three items: Shopping in Lahaina, Whale Watching, and Snorkeling.

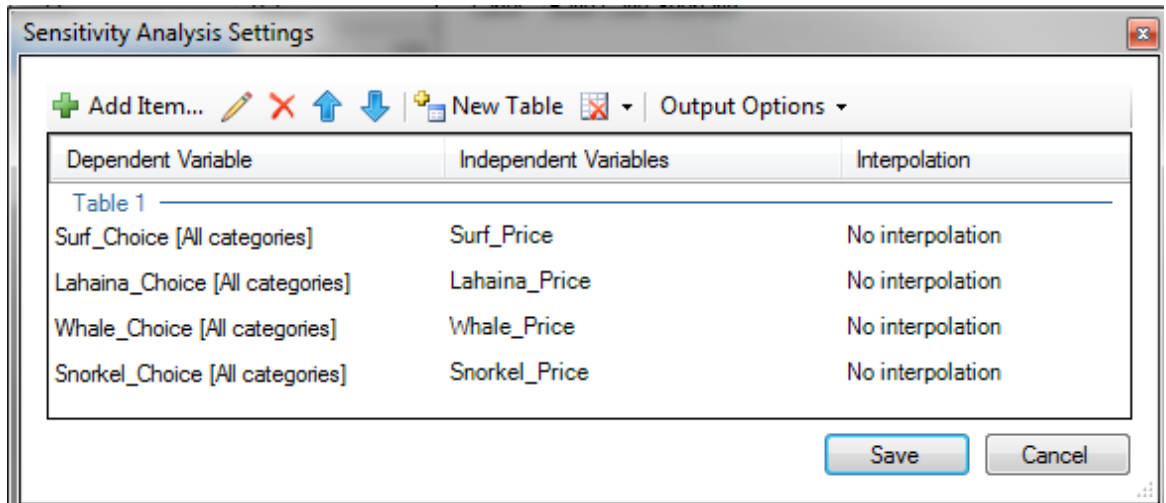
From the **Simulations** tab, we edit the *Base Case Scenario*, and click the **Settings** button (right next to the *Sensitivity Analysis* checkbox). The Sensitivity Analysis Table as we had built before (for Example #1) is shown:



Click the **Add Item...** icon at the upper-left of the dialog. Then, specify that you also want to simulate the choice likelihood of **Lahaina_Chosen** as a function of **Lahaina_Price**:



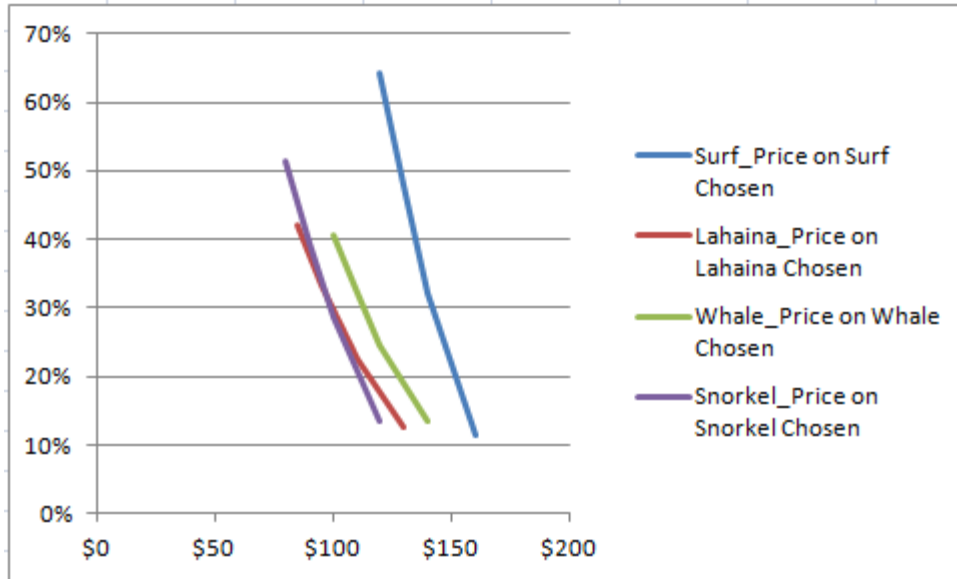
Click the **Add** button. Then, repeat the same process to add the demand curves for **Whale_Choice** and **Snorkel_Choice**:



Click **Save**, and then return to the *Simulations* tab to run the simulation scenario. The following table is produced (as shown when copied into Excel):

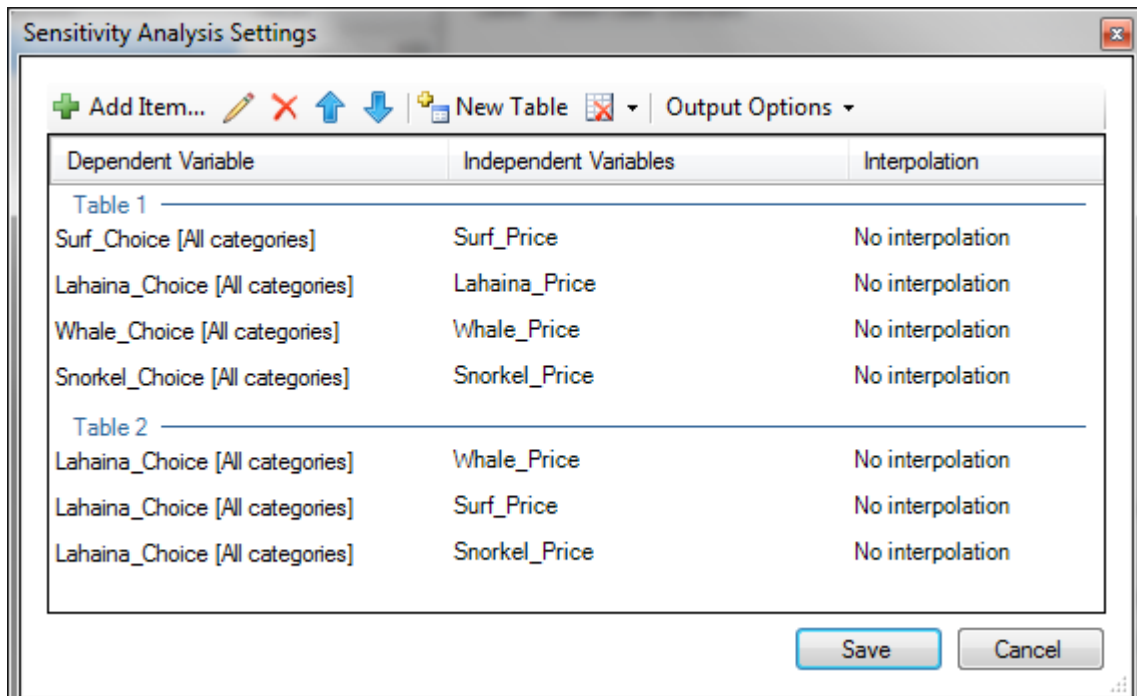
	Surf_Price	Lahaina_Price	Whale_Price	Snorkel_Price	
Surf_Choice	120	0.6425			
	130	0.4802			
	140	0.322			
	160	0.1138			
Lahaina_Choice	85		0.4192		
	95		0.3341		
	110		0.2252		
	130		0.1246		
Whale_Choice	100			0.4075	
	110			0.3217	
	120			0.2464	
	140			0.1358	
Snorkel_Choice	80				0.5154
	90				0.3949
	100				0.2857
	120				0.1343

Within Excel, if you highlight the area as shown above and click *Insert + Scatter Plot + Scatter with Straight Lines*, you get the following chart:



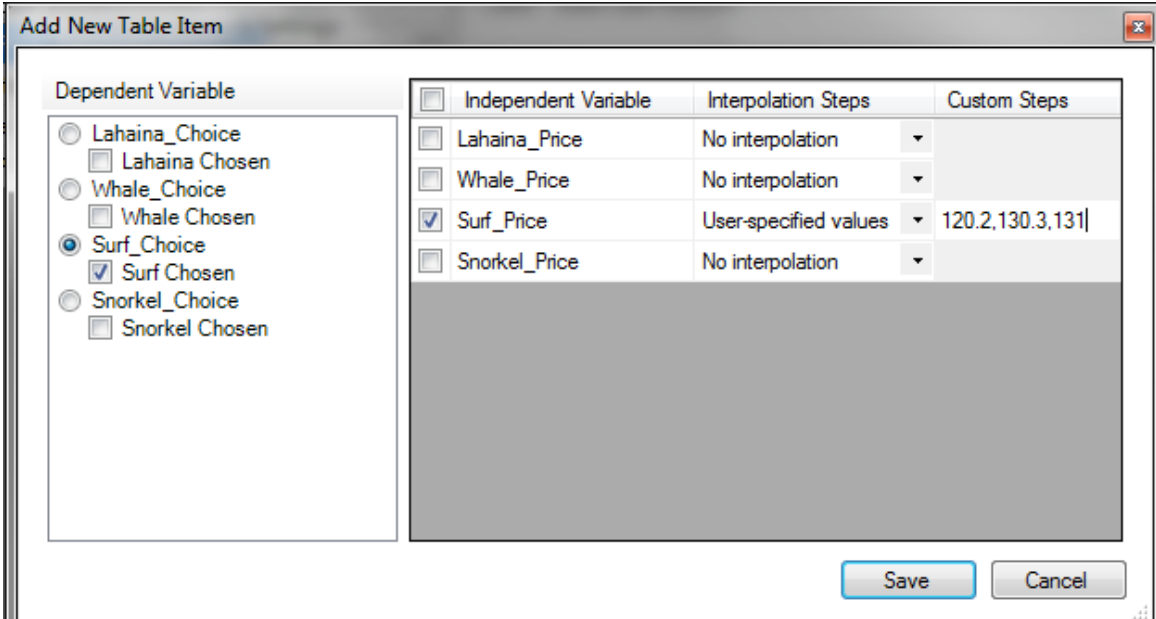
Multiple Tables

If you keep adding items to the same table, they will be formatted as new columns in the output. If you want to start a new table in the output, do so by clicking the *New Table* icon. In the example below, we have started a new table, which will contain the cross-elasticity curves for how other items' prices affect the likelihood of choosing the Lahaina Shopping Trip.



Custom Steps

If you don't like the interpolation steps that MBC chooses, and you want to specify certain price points, you can click *User-specified values* for the *Interpolation Steps*, and then type the values you want to use (separated by commas) within the *Custom Steps* field, as shown below:



The screenshot shows the 'Add New Table Item' dialog box. On the left, under 'Dependent Variable', there are radio buttons for 'Lahaina_Choice', 'Whale_Choice', 'Surf_Choice', and 'Snorkel_Choice', each with a corresponding 'Chosen' checkbox. 'Surf_Choice' is selected, and 'Surf Chosen' is checked. On the right, a table lists independent variables with their interpolation steps and custom steps.

<input type="checkbox"/> Independent Variable	Interpolation Steps	Custom Steps
<input type="checkbox"/> Lahaina_Price	No interpolation	
<input type="checkbox"/> Whale_Price	No interpolation	
<input checked="" type="checkbox"/> Surf_Price	User-specified values	120.2,130.3,131
<input type="checkbox"/> Snorkel_Price	No interpolation	

Buttons: Save, Cancel

Prices \$120.20, \$130.30, \$131, etc. will be used in the sensitivity run.

10.8 Excel Simulator

MBC can automatically create an Excel simulator containing the utility runs and necessary mathematical functions to perform standard Choice Simulations (not combinatorial choice simulations). The .xlsx file (Microsoft Office Excel 2007 Workbook) format is used. Segments and weights may be included. The resulting simulator file is usually less than 3MB and may be emailed to others. It produces results exactly equal to the results of the standard MBC simulator.

To create the Excel simulator file, go to the **Simulations** tab. Click the *Create Excel simulator* icon, and a wizard steps you through the process.

The .xlsx file contains VBA code, so when opening the simulator file, you'll be prompted regarding whether you want to enable the macros that are present within the file. You should specify to enable their functionality.

Only point estimates are supported (draws are not supported). We have found that draws (typically hundreds of separate estimates of the utilities for each respondent) would lead to too much information and too many formulas to manage within the .xlsx files. Just opening an Excel simulator based on draws can take 5 minutes or more, which would not be tolerated by users or their clients. Therefore, ***if you know you will be delivering an Excel simulator to your client, you should make sure that the initial work you do for your client using the MBC simulator applies the point estimates rather than the draws.*** This setting (to use point estimates rather than draws) is on the *Scenario* dialog.

The simulator file has multiple sheets, which probably should be hidden when you deliver the file to your client. Most users will add their own attractive front sheet that provides an attractive user interface (often mimicking the look of the original menu within the questionnaire), and a place to display results and/or graphs/charts. That customized front sheet should be linked via pointing formulas to the basic front sheet that has been automatically written within the .xlsx file. That basic front sheet in turn points to the result of formulas within deeper sheets within the .xlsx file.

The simulator file contains VBA code that takes the independent variable levels you have specified on the front sheet, examines the settings involved in the utility runs (model specifications), and determines how to code the simulation scenario into design matrices appropriate for each dependent variable model. A matrix multiply step is then performed for each model and each respondent (if using HB utilities), multiplying the design matrix by the respondent beta vector. Then, the logit rule is performed, as described in the earlier section on [Market Simulation Math](#).

11 Simulating Combinatorial Choice Outcomes

11.1 Simulating Combinatorial Choice Outcomes

The previous section described how MBC software can predict the likelihood of menu items being selected across the sample of respondents using (typically) separate, but interrelated, logit-based models. The results were expressed in terms of percent of respondents expected to choose each of many items on the menu, given specific menu prices. Some researchers and clients also require a prediction of the most likely *combinatorial selections* made by respondents. For example, results from a combinatorial prediction simulator could reveal that the combination *Hamburger + Medium Fries + Small Drink* is the most likely to be chosen on the menu, given specific menu prices. The percent of respondents projected to choose that specific combination of menu items can be shown, as well as the next n most likely combinations with their respective likelihoods. The attentive reader will notice that this output has the same format as the counting report of combinatorial choices described earlier and shown in Figure 4.11. Whereas the counting report nets the combinatorial choices over all price variations shown in the experiment, the combinatorial simulation predicts the most likely combinatorial choices *given a specific set of menu prices* as specified in the simulation scenario.

11.2 HB Draws and Combinatorial Choice Predictions

To make combinatorial predictions with our MBC software, one must be using individual-level HB models. And, we recommend using combinatorial coding for dependent variables, for either the entire menu (if feasible) or logical sections of the menu.

With Sawtooth Software's market simulation tools for standard CBC, we have advocated collapsing the draws for each respondent into a single point estimate, representing the average of each respondent's utility draws. Under this approach, choice likelihoods for items are estimated by applying the logit rule to the point estimates. MBC can do this as well, and it is faster than operating on individual draws. But we recommend simulating combinatorial choices at the level of the draws. This is partly due to the speed of modern computers. But the main reason we recommend operating on the level of the draws is to make possible a special way of simulating combinatorial choices called *weighted draws* (described later in this section).

11.3 Simulating Combinatorial Outcomes

Now we will describe the mechanics of simulating combinatorial choices. For each item on the menu, we have an HB model (with multiple draws per respondent) that may be used to compute the likelihood that the respondent would pick that item (given a user-specified set of prices in the simulation scenario).

(This of course assumes the simple approach of coding each menu selection as a separate dependent variable. If you have collapsed multiple selections into a single dependent variable with multiple codes, such as three binary variables coded as a single 8-level variable, then each dependent variable actually reflects a combination of selections from the menu.)

The likelihood is computed using the logit rule, and is a value ranging from 0.0 to 1.0. If we have, say, eight items in the menu (each with two categories: chosen or not chosen), we may use eight separate HB models to predict the likelihood that each item was selected by this respondent. For example, for the first respondent, we might compute likelihoods of selecting the first category (choice) of the eight items as follows: 0.42, 0.26, 0.87, 0.92, 0.01, 0.02, 0.61, 0.07. Given those probabilities, the most likely prediction of what the respondent would choose can be discretized as 2, 2, 1, 1, 2, 2, 1, 2, where "1" indicates chosen and "2" indicates not chosen. For each dependent variable, the category with the highest probability is projected to be chosen (similar to the notion of the *first choice* simulation rule used in traditional conjoint simulators).

We suggest using draws for combinatorial predictions, and we use each of those draws to make first-choice combinatorial choice predictions (as if each respondent made multiple shopping trips). We could simply tally these combinatorial selections across all respondents and draws to achieve a combinatorial simulation result and distribution (listing the *n* most likely combinations, each followed by the likelihood of choice across the sample). If we had 1000 respondents, we would have 200,000 combinatorial predictions. However, many researchers and clients prefer to have a single individual-level prediction (rather than 200), for classifying respondents and computing hit rates (for holdout menu choices). Therefore, we select the *one most likely* combinatorial choice for each respondent for reporting purposes.

If operating at the level of the draws, one way to select the most likely combinatorial selection for each respondent is to count which *one* combination of projected first choices occurs the most often across a respondent's draws. This is one of the options we have done in MBC software, but we have provided an additional variation called "weighted draws". A series of separate logit-based HB models might make a prediction regarding a combinatorial choice that is not very likely to be made by respondents. To reduce the likelihood of this, we can incorporate additional information about the combinatorial choices *actually made* by respondents to inform the simulation procedure about likely and unlikely choices.

For each respondent we examine the tasks used in developing the models (the calibration tasks, of which there are often 8 or more), and we tally the frequency that each combination of menu items was selected. For example, if eight items were included on the menu, the combination 2, 2, 1, 1, 2, 2, 1, 2 (indicating that items 3, 4, and 7 were chosen) might have been selected 3 times. We can use the log of this frequency (to avoid taking the natural log of 0, we add 2 to the frequency tallies before taking the natural log) as a weight when computing which of the 200 combinatorial

predictions for each respondent is the most likely (assuming the default 200 draws). So, rather than simply tallying the 200 predictions and selecting the most likely combination, we do a weighted count that favors combinations that are actually observed to be more likely. This type of market simulation integrates combinatorial likelihood information directly from each respondent. Thus, if two items are highly substitutable and were never chosen together by the respondent, if the simulator tries to predict that this respondent will choose those two items together, such a result would be discouraged. Only if a relatively large number of draws for a respondent indicate the joint choice of those two items would such a combinatorial prediction be allowed.

Weighting the combinatorial predictions for the draws by a log transform of the frequency as observed in the calibration tasks is a *post hoc* simulation approach that is not theoretically grounded. We do this to help overcome weaknesses in the HB-logit approach to combinatorial predictions. And, we have empirical evidence that this procedure can indeed improve the accuracy of individual-level predictions. As we collect further evidence from more datasets, we'll learn even more regarding whether this is a robust procedure that will yield consistently better results in practice. If weighting the draws in this manner bothers you, you can choose the simulation option that doesn't apply population frequency weights to the predictions from the draws. You may also select the quicker approach of projecting combinatorial choices using the point estimates.

11.4 Running Combinatorial Simulations in MBC

Running Combinatorial Simulations in MBC

Simulate Choice Probabilities
 Sensitivity Analysis
 For HB utilities, use point estimates instead of draws

➔ Simulate Choice Combinations

Top combinations to display
 Display individual results when available

Dependent Variable Utilities		
	Dependent Variables	Utility Run
▶ 1	Lahaina_Choice	Lahaina_Choice (HB) 26Jan2012 1113
2	Whale_Choice	Whale_Choice (HB) 26Jan2012 1113
3	Surf_Choice	Surf_Choice (HB) 26Jan2012 1113
4	Snorkel_Choice	Snorkel_Choice (HB) 26Jan2012 1113

Figure 10.1 (Simulation Scenario with Combinatorial Simulation Selected)

To simulate combinatorial choices, you simply edit a simulation scenario and then select the *Simulate Choice Combinations* option, as shown in Figure 10.1. You can specify how many of the top (most frequently predicted) combinations to display in the report. By default, the 10 most commonly predicted combinatorial choices will be shown on the screen. If you select *Display individual results when available*, each respondent's predicted combinatorial selection will also be given, available in a separate tab of the report.

In the previous section, we described how we can operate on the level of the draws and apply individual-level information regarding the frequency of specific combinations of items selected (across all relevant tasks) to improve individual-level predictions from the simulator. The three options supported by the software are:

Use Weighted Draws: (Default Method) The draws are weighted by the natural log of how often the combination suggested by the "first choice" rule for that draw was observed to be selected by the respondent.

Use Draws: No additional weighting is used to influence the individual combinatorial choice predictions.

Use Point Estimates: Point estimates are used to predict the individual combinatorial choice predictions (faster method, but usually inferior).

Recommendation: we have found that the method of weighted draws works better for a few datasets we have examined. Weighted draws is therefore the recommended method.

11.5 2006 Fast-Food Menu Study Results

In 2006, we conducted an MBC research study involving 681 respondents (Opinionology sample). The exercise involved selecting items on a fast-food menu (as shown in figure 1.6). Each respondent completed 10 menu tasks, and tasks 4 and 5 were held out from utility estimation. There were 1809 total possible combinations that respondents could select within the menu. If we built a simulator that made random predictions of menu choices for each respondent, we would expect a hit rate of $1/809 = 0.06\%$, which would not be very good at all.

For this study, tasks 4 and 5 were held out from utility estimation, and used solely for purposes of measuring internal predictive validity. A simple way to predict choices is to assume that respondents will pick the same thing in tasks 4 and 5 that they picked in task 3. It turns out that using task 3's choices for each respondent to predict responses to tasks 4 and 5 (where the menu prices changed in each task) results in a hit rate of 42.2%, which is almost 800 times the chance level! But, we have interest in predicting combinations using our logit-based models rather than assuming that respondents will select simply what they most recently selected, irrespective of price changes.

Using MBC software's approach of a series of HB models (unconstrained) with linear alternative-specific own-effects plus significant linear cross-effects, we used MBC's simulator (on the draws) and achieved an average predictive hit rate over tasks 4 and 5 of 50.6%. If we weight the draws according to the likelihood of combinations occurring across the tasks used in utility estimation, the hit rate increases slightly to 53.0%. Unfortunately, we don't have test-retest reliability figures for this study, so we don't know if these predictions are about as good as we possibly could expect. If we had repeated tasks 4 and 5 at a later point in the survey, then we could see how likely respondents were to choose the same menu options when presented with the same menu scenarios. After all, respondents will not be perfectly consistent from task to task, so there is a theoretical upper limit to predictive validity significantly below 100%. Perhaps our hit rate of 53% is approaching the test-retest reliability rate, and is thus about as good as we could expect to do. Again, we cannot know for this study. But, a second study we conducted (described in the next section) did include test-retest reliability.

We also used MBC's ability to simulate shares of choice on the margin (for each item individually) and predicted the marginal shares of choices for holdout tasks 4 and 5. Regressing the actual likelihoods on the predicted resulted in an R-square of 0.96.

For this restaurant dataset, the *a la carte* choices were not available to respondents unless they had first rejected the value meal combos. This would seem to justify a two-stage simulation with conditional dependent variables for the three *a la carte* choices, triggered by respondents first rejecting the combo meals. But, we actually achieved better predictions when *not* specifying conditional dependent variables. We don't know if this result will hold for other datasets, but mention it here to report what model specification performed best for this dataset.

11.6 2011 Fast-Food Menu Study Results

In 2011, we fielded another fast-food restaurant menu study, this time choosing a design that was thought would be especially challenging for our logit-based approach. Rather than having mutually-exclusive choices within seven categories (as we did in the 2006 study), we had 21 binary choices on the menu, with no restrictions on the ways respondents could complete the menu.

Menu Scenario (4 of 12)		
Imagine you were visiting a fast-food restaurant for lunch, and the following menu and prices were offered. You are hungry, and are going to buy something.		
What would you purchase just for yourself? Pick as many or as few items as you wish.		
Combo Meals (Comes with Medium Fries and Medium Fountain/Soft Drink)	Sandwiches	
<input type="checkbox"/> Hamburger Value Meal \$ 5.19 <input type="checkbox"/> Cheeseburger Value Meal \$ 5.39 <input type="checkbox"/> Chicken Sandwich Value Meal \$ 5.29 <input type="checkbox"/> Chicken Nuggets Value Meal \$ 5.69	<input type="checkbox"/> Hamburger \$ 2.69 <input type="checkbox"/> Cheeseburger \$ 2.99 <input type="checkbox"/> Chicken Sandwich \$ 2.79 <input type="checkbox"/> Chicken Nuggets \$ 3.19	
Drinks	Sides	Extra Value Menu
<input type="checkbox"/> Fountain/Soft Drink (Large) \$ 1.59 <input type="checkbox"/> Fountain/Soft Drink (Medium) \$ 1.09 <input type="checkbox"/> Coffee \$ 1.09 <input type="checkbox"/> Shake (Medium) \$ 1.49 <input type="checkbox"/> Milk \$ 0.89	<input type="checkbox"/> Fries (Medium) \$ 1.69 <input type="checkbox"/> Onion Rings \$ 1.89 <input type="checkbox"/> Salad topped with chicken \$ 3.49 <input type="checkbox"/> Chili \$ 1.69	<input type="checkbox"/> Junior Cheeseburger \$ 1.09 <input type="checkbox"/> Junior Chicken Sandwich \$ 1.19 <input type="checkbox"/> Small Fries \$ 1.39 <input type="checkbox"/> Side Salad \$ 0.99
Total Price of Your Order: \$ 0.00		

Figure 10.2 (2011 Restaurant Study Menu)

There were $2^{21}=2,097,152$ total possible combinations that respondents could select. 1698 respondents were interviewed (again, Opinionology sample), with 850 respondents used for building the model (calibration sample) and another 848 holdout respondents randomly selected to serve as validation sample. The holdout respondents received one of two versions of the questionnaire, each with 12 menu tasks. Calibration respondents received 12 tasks as well, where 8 of the tasks came from a (controlled) randomized design. The other 4 tasks were fixed holdouts, with tasks 4 and 9 being replicates, and tasks 5 and 10 also replicates (to assess test-retest reliability).

Another interesting aspect of this study is that the "Extra Value Menu" section in the bottom right (with its four menu items) was only available in half of the choice tasks.

Using models developed from 8 tasks completed by the calibration respondents, we compared the share predictions for individual items to the actual choice likelihoods observed in the holdout sample (a total of 456 predicted choice likelihoods for menu items across 24 menu tasks). Regressing the actual likelihoods on the predicted likelihoods resulted in an R-square of 0.88. We shouldn't expect to achieve perfect prediction (R-Square of 1.0) due to sampling error, since we are using 850 respondents to predict responses from two *different* sets of 424 respondents. Based on simulations using random draws of synthetic respondents (samples of $n=850$ and $n=424$), with expected proportions as observed in the data, we estimate the expected R-squared for prediction for this situation (if the only error is due to sampling) is 0.91. So, we conclude that our prediction model is doing almost as well as it should theoretically do.

Calibration respondents received four holdout menu tasks, where tasks 4 and 9 were repeats, and tasks 5 and 10 were also repeats. The average test-retest reliability (how often respondents could answer the same menu task in the same way) was 53.9%. The *weighted draws* approach to combinatorial simulations yielded a predictive hit rate accuracy of 49.6%. The non-weighted draws approach yielded hit rate accuracy of 48.2%. Both results are approaching test-retest reliability, or the maximum predictive accuracy that is theoretically possible.

We tried two ways of coding the dependent variables: 21 separate binary dependent variables vs. four combinatorial dependent variables (based on logical groupings of items from the menu). The combinatorial dependent variable coding was more successful in predictive accuracy (we've reported its results above), and was also a *quicker* model to estimate and simulate. Formulating the problem as 21 binary variables led to predictive accuracy of shares of 0.86 and hit rates of 46.1% using "weighted draws" (compared to 0.88 and 49.6% for the combinatorial dependent variables model).

12 Tutorial: Configuring an Automobile

12.1 Tutorial: Configuring an Automobile

In this section, we give a detailed description of how to use MBC software to model choices for a real dataset that involved respondents configuring automobiles. While there are many ways to model these data, we'll focus on two approaches a researcher might select. The first one follows the "serial cross-effects" models that treat each item on the menu as a separate model. The second approach breaks the menu into three logical subsections and examines all possible patterns of checked options within those subsections. Both models use a two-stage approach for this particular dataset, where the dependent variables in a second model are conditional upon choices made to the dependent variable in the first stage model.

A total of 1629 respondents completed a web-based questionnaire, fielded using Opinionology panel, during 2010. Each respondent completed 8 menu-based choice tasks as shown below. 806 respondents are included in the dataset, and represent calibration respondents. 823 additional respondents were interviewed and became holdout respondents used for validation.

Respondent Task:

If you were deciding between the following three cars, and the prices were as shown, which car would you select, at which options would you add to it?

<input type="radio"/> Honda Accord	<input type="radio"/> BMW 3-Series	<input type="radio"/> Hyundai Elantra
Base Price: \$28,000	Base Price: \$27,000	Base Price: \$20,000
<input type="checkbox"/> \$2,000 Alloy Wheels	<input type="checkbox"/> \$1,750 Alloy Wheels	<input type="checkbox"/> \$1,500 Alloy Wheels
<input type="checkbox"/> \$1,200 Moonroof/Sunroof	<input type="checkbox"/> \$900 Moonroof/Sunroof	<input type="checkbox"/> \$700 Moonroof/Sunroof
<input type="checkbox"/> \$300 XM Radio (+ \$13/month)	<input type="checkbox"/> \$600 XM Radio (+ \$13/month)	<input type="checkbox"/> \$500 XM Radio (+ \$13/month)
<input type="checkbox"/> \$1,600 Navigation system (in dash)	<input type="checkbox"/> \$1,000 Navigation system (in dash)	<input type="checkbox"/> \$2,000 Navigation system (in dash)
Total: \$28,000	Total: \$27,000	Total: \$20,000

Figure 11.1 (The Respondent Task)

Respondents chose a column (the radio button), and then configured (added) any options within the column they wished (the check boxes). As they added options, the total price of the vehicle shown at the bottom of the selected column dynamically updated. Note that respondents were allowed to add options only for the *one* vehicle they had chosen.

Experimental Design:

Respondents saw their top three considered vehicles within each choice set (as previously specified in the questionnaire). The top considered vehicle always appeared in the first concept/column. The 2nd considered vehicle always appeared in the second column, and the 3rd considered vehicle always in the third column. The base prices shown were set to the price respondents expected to pay for each vehicle (as previously specified in the questionnaire). The prices for each of the options (alloy wheels, sunroof, XM Radio, or navigation system) could take on 4 discrete values, as shown below. Additionally, the base price for each vehicle was varied experimentally by -\$3,000, \$0, or +\$3,000.

Alloy Wheels

\$1,500
\$1,750
\$2,000
\$2,500

Sunroof

\$500
\$700
\$900
\$1,200

XM Radio

\$300
\$400
\$500
\$600

Navigation System

\$1,000
\$1,300
\$1,600
\$2,000

Base Price

-\$3,000
+\$0
+\$3,000

We employed a randomized design, using CBC's Balanced Overlap design strategy.

Conceptualizing the Models

Referring back to Figure 11.1, we can conceptualize the respondent task as a two-stage choice process, as follows: Each respondent views the three cars on the screen (together with the option prices) and first decides which column (car) to select. Once the respondent selects the car, the respondent then configures the selected choice with optional alloy wheels, sunroof, XM radio, and navigation system. Obviously, options can only be configured on a car that has been chosen, so this represents a series of conditional dependent variables for the second-stage model.

The Data File:

Three sample datasets were installed with your MBC software, and are located in **.../My Documents/Sawtooth Software/MBC Samples**. We will first focus on the dataset named **CarBinaryDVs.csv**. (If you are using the MBC software in Demo mode, it will only analyze the first 50 respondents, so your results will be different from ours shown below).

The first few rows of the data file look like:

ID	----Independent Variables----	---Dependent Variables---
1	3 4 1 3 2 1 2 3 3 3 1 3 4 1 4	1 2 1 2 2 2 2 2 2 2 2 2 2
1	3 1 4 2 3 2 3 3 4 1 2 4 1 2 1	3 2 2 2 2 2 2 2 2 2 2 1 2 2
1	2 2 4 3 2 1 1 1 1 1 3 2 3 2 4	3 2 2 2 2 2 2 2 2 2 2 1 2 2
1	1 1 2 3 2 3 4 1 2 1 2 3 2 1 4	1 1 1 2 1 2 2 2 2 2 2 2 2 2
1	1 3 1 4 4 4 1 2 2 3 3 3 1 4 4 3	1 2 1 2 2 2 2 2 2 2 2 2 2 2
1	2 1 3 4 4 3 3 4 1 2 2 4 2 1 2	3 2 2 2 2 2 2 2 2 2 2 1 2 1
1	1 1 3 2 3 2 2 1 3 3 3 4 3 1 1	1 2 1 2 1 2 2 2 2 2 2 2 2 2
1	3 4 2 4 1 2 3 1 2 2 2 2 4 3 4	1 2 1 2 1 2 2 2 2 2 2 2 2 2
2	2 3 2 4 1 1 4 4 2 4 3 2 2 1 2	1 2 2 1 2 2 2 2 2 2 2 2 2 2
2	1 3 1 4 3 2 4 4 1 1 3 1 3 3 2	1 2 2 1 2 2 2 2 2 2 2 2 2 2
2	1 2 4 4 3 2 3 3 2 4 1 1 2 3 1	1 2 2 1 2 2 2 2 2 2 2 2 2 2
2	3 4 3 3 2 2 2 3 1 4 3 3 2 2 3	2 2 2 2 2 2 2 1 2 2 2 2 2 2
2	3 3 2 4 1 1 1 1 2 2 2 4 4 3 4	2 2 2 2 2 2 2 1 2 2 2 2 2 2
2	2 2 1 1 1 1 3 1 3 3 3 4 3 4 3	1 2 2 1 2 2 2 2 2 2 2 2 2 2

The Data layout is as follows:

Col#	Name:	Description:	Values:
1	RespID	Case ID	1-893
<i>(Independent Variables):</i>			
2	Concept1_Pr_Base	Base Price Concept #1	1-3
3	Concept1_Pr_Alloy	Alloy Wheels Price Concept #1	1-4
4	Concept1_Pr_Sunroof	Sunroof Price Concept #1	1-4
5	Concept1_Pr_Radio	XM Radio Price Concept #1	1-4
6	Concept1_Pr_Nav	Navigation System Price Concept #1	1-4
7	Concept2_Pr_Base	Base Price Concept #2	1-3
8	Concept2_Pr_Alloy	Alloy Wheels Price Concept #2	1-4
9	Concept2_Pr_Sunroof	Sunroof Price Concept #2	1-4
10	Concept2_Pr_Radio	XM Radio Price Concept #2	1-4
11	Concept2_Pr_Nav	Navigation System Price Concept #2	1-4
12	Concept3_Pr_Base	Base Price Concept #3	1-3
13	Concept3_Pr_Alloy	Alloy Wheels Price Concept #3	1-4
14	Concept3_Pr_Sunroof	Sunroof Price Concept #3	1-4
15	Concept3_Pr_Radio	XM Radio Price Concept #3	1-4
16	Concept3_Pr_Nav	Navigation System Price Concept #3	1-4
<i>(Dependent Variables):</i>			
17	Ch_Auto_Concept	Choice of Column/Concept	1-3
18	Ch_Alloy1	Choice of Alloy Wheels Concept#1	1-2
<i>("1" indicates chosen; "2" not , etc.)</i>			
19	Ch_Roof1	Choice of Sunroof Concept#1	1-2
20	Ch_Radio1	Choice of XM Radio Concept#1	1-2
21	Ch_Nav1	Choice of Nav System Concept#1	1-2
22	Ch_Alloy2	Choice of Alloy Wheels Concept#2	1-2
23	Ch_Roof2	Choice of Sunroof Concept#2	1-2
24	Ch_Radio2	Choice of XM Radio Concept#2	1-2
25	Ch_Nav2	Choice of Nav System Concept#2	1-2
26	Ch_Alloy3	Choice of Alloy Wheels Concept#3	1-2
27	Ch_Roof3	Choice of Sunroof Concept#3	1-2
28	Ch_Radio3	Choice of XM Radio Concept#3	1-2
29	Ch_Nav3	Choice of Nav System Concept#3	1-2

In this data layout, we've coded the dependent variables each as separate variables. There are 13 dependent variables, and we will be building 13 separate models (later in this section, we'll show

how to code the model with 4 combinatorial dependent variables, and we'll build 4 separate models).

Reading the Data File into MBC Software

First, we open the MBC software and click **File + Open...** and browse to the data file (**.../My Documents/Sawtooth Software/MBC Samples/CarBinaryDVs.csv**). We have already defined our category labels and values in the **CarBinaryDVs Labels.csv**, so we can click the **Import Category Labels and Values from File** icon (as indicated by the black arrow in Figure 11.2) to import those automatically. (Click [here](#) to review the labels file layout.)

Next, on the **Variables** tab, we specify (using drop-down boxes) which variables are the dependent variables, and for the twelve dependent variables representing the configuration of options we specify that the second category for each of those is the "Not Selected" Off state. Once that is done, the **Variables** tab will look like Figure 11.2.

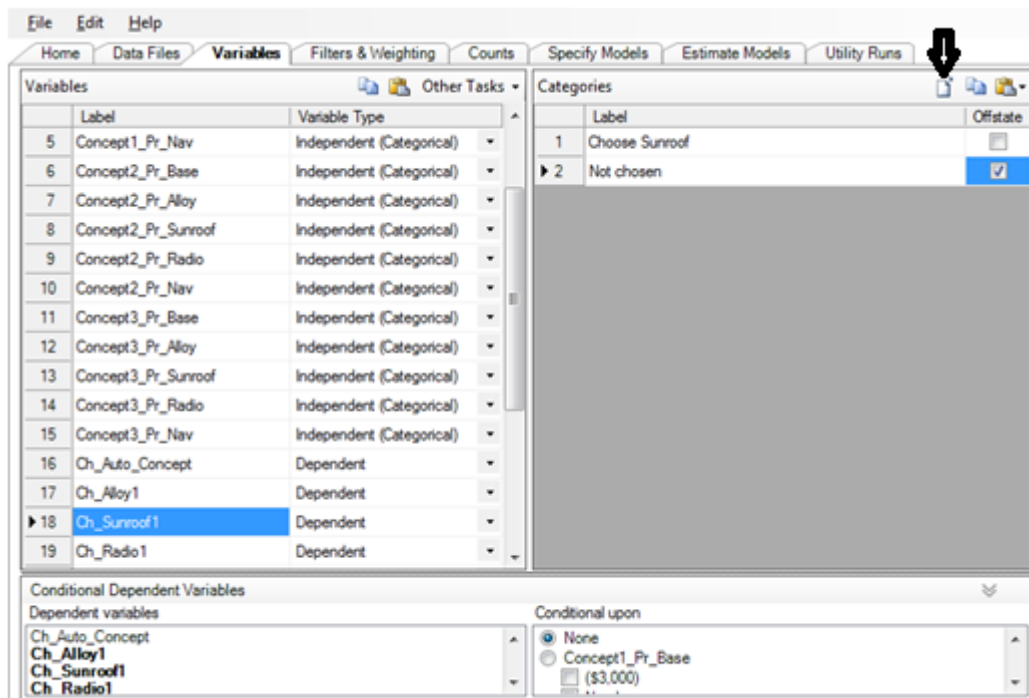


Figure 11.2 (The Variables Tab)

Next, we need to specify that variables **Ch_Alloy1** through **Ch_Nav3** are *Conditional Dependent Variables*. In the section at the bottom of the **Variables** dialog (as shown in Figure 11.2), we click each of these variables and indicate that each is dependent on categories of **Ch_Auto_Concept**. **Ch_Alloy1** through **Ch_Nav1** (configurations for the automobile in the left column of the questionnaire) are only valid if **Ch_Auto_Concept** = 1. **Ch_Alloy2** through **Ch_Nav2** (configurations for the automobile in the middle column of the questionnaire) are only valid if **Ch_Auto_Concept** = 2. And, **Ch_Alloy3** through **Ch_Nav3** (configurations for the automobile in the right column of the questionnaire) are only valid if **Ch_Auto_Concept** = 3.

Counting Analysis

It is typical to use counting analysis to examine the frequencies of each independent and dependent variable, and to examine the relationship between independent variables and dependent variables. We can examine which variables appear to be significant drivers of choice of dependent variables, and the best-fitting functional forms (linear, log-linear, or part-worth).

These steps are described in the section on [Counts](#), and we won't take the space here to describe them. Counting analysis suggests that the biggest driver of choice for **Ch_Auto_Concept** (the discrete choice of column in the questionnaire) is the base price of the automobile. Only a few of the options' prices under each automobile seem to be significant drivers of choice according to the chi-square counting tests. Even so, most of these relationships appear to show negative relationships between price and choice likelihood (as prices go down, likelihood of choice goes up).

Despite the fact that very few of these demonstrate a statistically significant relationship at the 80% confidence or higher via counting analysis, we will consider putting them into the logit-based models. They conceptually make a lot of sense, and especially if we collapse alternative-specific variables into generic predictors of column choice, their stability and precision increase.

Specifying Models

For help in describing the models, we again show a task from the questionnaire:

If you were deciding between the following three cars, and the prices were as shown, which car would you select, and which options would you add to it?

<input type="radio"/> Honda Accord	<input type="radio"/> BMW 3-Series	<input type="radio"/> Hyundai Elantra
Base Price: \$28,000	Base Price: \$27,000	Base Price: \$20,000
<input type="checkbox"/> \$2,000 Alloy Wheels	<input type="checkbox"/> \$1,750 Alloy Wheels	<input type="checkbox"/> \$1,500 Alloy Wheels
<input type="checkbox"/> \$1,200 Moonroof/Sunroof	<input type="checkbox"/> \$900 Moonroof/Sunroof	<input type="checkbox"/> \$700 Moonroof/Sunroof
<input type="checkbox"/> \$300 XM Radio (+ \$13/month)	<input type="checkbox"/> \$600 XM Radio (+ \$13/month)	<input type="checkbox"/> \$500 XM Radio (+ \$13/month)
<input type="checkbox"/> \$1,600 Navigation system (in dash)	<input type="checkbox"/> \$1,000 Navigation system (in dash)	<input type="checkbox"/> \$2,000 Navigation system (in dash)
Total: \$28,000	Total: \$27,000	Total: \$20,000

Figure 11.3 (The Respondent Task)

As mentioned previously, we conceptualize this menu as a two-stage choice procedure. First, respondents examine the three columns (the three automobiles) in the questionnaire and decide which one they will choose. This first-stage model is essentially identical to a standard CBC setup: respondents view the characteristics of each automobile and select which one to buy. Each automobile is characterized by variation in its base price, and variation in the prices of its four options. There is a single dependent variable (**Ch_Auto_Concept**) taking on mutually-exclusive categories 1, 2, or 3. We may model this as a standard MNL, where the utility of the category (column of the questionnaire) is a function of the desirability of that column's characteristics. As the characteristics of a column improve (such as its prices lowering), the likelihood of choice for this category (column) increases and the likelihood of choice of the other categories (columns) decreases. Referring to Figure 11.3, as the pricing characteristics of Honda Accord and its options improve, the projected likelihood of choice for BMW 3-Series and Hyundai Elantra decrease.

It is also possible to consider additional cross-effect terms, wherein the utility of a column is not only a function of that column's characteristics, but the characteristics of competing columns. However, this additional series of explanatory terms is probably not needed, since substitution/competition among the columns is already largely accounted for in the standard MNL model (especially using heterogeneous models, such as HB).

As a starting-point model, let's consider that the choice of column in the questionnaire (automobile) is a function of all its characteristics (specified as unique relationships to each column, meaning alternative-specific effects). From Sawtooth Software's CBC software perspective, this is the same thing as specifying a logit model with 2-way interaction terms between "attribute 1 (ASCs)" and the other four attributes (price of alloy wheels, price of sunroof, price of radio, and price of navigation system). On the **Specify Models + Variable Codings** tab, this specification looks as follows (we've kept things simple throughout this section by coding all independent variables as *Linear* functional forms):

Select Independent Variables		Predicting Categories of Dependent Variable			
Independent Variables	Coding	Choose Left Concept	Choose Middle Concept	Choose Right Concept	
1 Concept1_Pr_Base	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2 Concept1_Pr_Alloy	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
3 Concept1_Pr_Sunroof	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
4 Concept1_Pr_Radio	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
5 Concept1_Pr_Nav	Linear	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
6 Concept2_Pr_Base	Linear	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
7 Concept2_Pr_Alloy	Linear	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
8 Concept2_Pr_Sunroof	Linear	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
9 Concept2_Pr_Radio	Linear	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
10 Concept2_Pr_Nav	Linear	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
11 Concept3_Pr_Base	Linear	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
12 Concept3_Pr_Alloy	Linear	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
13 Concept3_Pr_Sunroof	Linear	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
14 Concept3_Pr_Radio	Linear	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
15 Concept3_Pr_Nav	Linear	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 11.4 (Alternative-Specific Coding)

The **Variable Codings** dialog makes it easy to navigate and add/remove variables. But, you should keep in mind that the rows and columns are flipped from the actual design matrix orientation in the estimation procedure. In terms of utility estimation, the rows in this dialog (the Independent Variables) really represent the *columns* in the design matrix.

If we click **Preview Design Matrix**, we can preview the design matrix for each respondent (each of the price variables has been zero-centered, and given a range of 1.0, for more information on design coding, see the [appendix](#)). The first three choice tasks for Respondent #1 are shown in Figure 11.5. Notice there are 17 predictor columns (independent variable parameters) and the final column is the dependent variable.

← Respondent: 1 →

Task/Alternative	ASC (1)	ASC (2)	1_Pr_Base	1_Pr_Alloy	1_Pr_Sunroof	1_Pr_Radio	1_Pr_Nav [Lines]	2_Pr_Base	2_Pr_Alloy	2_Pr_Sunroof	2_Pr_Radio	2_Pr_Nav	3_Pr_Base	3_Pr_Alloy	3_Pr_Sunroof	3_Pr_Radio	3_Pr_Nav	Response	
1 Choose Left Concept	0	0	0.5	0.563	-0.464	0.167	-0.175	0	0	0	0	0	0	0	0	0	0	0	1
1 Choose Middle Concept	1	0	0	0	0	0	0	-0.5	-0.188	0.107	0.167	0.125	0	0	0	0	0	0	0
1 Choose Right Concept	0	1	0	0	0	0	0	0	0	0	0	0	0	-0.5	0.063	0.536	-0.5	0.525	0
2 Choose Left Concept	0	0	0.5	-0.438	0.536	-0.167	0.125	0	0	0	0	0	0	0	0	0	0	0	0
2 Choose Middle Concept	1	0	0	0	0	0	0	0	0.063	0.107	0.5	-0.475	0	0	0	0	0	0	0
2 Choose Right Concept	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0.563	-0.464	-0.167	-0.475	1
3 Choose Left Concept	0	0	0	-0.188	0.536	0.167	-0.175	0	0	0	0	0	0	0	0	0	0	0	0
3 Choose Middle Concept	1	0	0	0	0	0	0	-0.5	-0.438	-0.464	-0.5	-0.475	0	0	0	0	0	0	0
3 Choose Right Concept	0	1	0	0	0	0	0	0	0	0	0	0	0.5	-0.188	0.107	-0.167	0.525	1	

Figure 11.5 (Preview Design Matrix, Alternative-Specific Coding)

If we are interested how well this model fits the data, we can run an aggregate logit model by clicking the *Estimate Models* tab and clicking the *Estimate* link for the *Ch_Auto_Select* dependent variable, logit run. The results are as follows:

```

Log-likelihood for this model =      -5506.70344
Log-likelihood for null model =     -7083.85204
-----
Difference =                        1577.14859

Percent Certainty =                 22.26400
Consistent Akaike Info Criterion = 11179.52282
Chi Square =                        3154.29719
Relative Chi Square =                185.54689
    
```

	Effect	Std Err	t Ratio	Variable
1	-0.59056	0.03601	-16.39993	ASC (1)
2	-0.81352	0.03869	-21.02597	ASC (2)
3	-1.85597	0.07501	-24.74143	Concept1_Pr_Base [Linear]
4	-0.07083	0.07917	-0.89467	Concept1_Pr_Alloy [Linear]
5	-0.20071	0.07789	-2.57701	Concept1_Pr_Sunroof [Linear]
6	-0.03315	0.07616	-0.43525	Concept1_Pr_Radio [Linear]
7	-0.24988	0.07853	-3.18184	Concept1_Pr_Nav [Linear]
8	-1.89078	0.08423	-22.44913	Concept2_Pr_Base [Linear]
9	-0.05958	0.08338	-0.71455	Concept2_Pr_Alloy [Linear]
10	-0.28357	0.08398	-3.37652	Concept2_Pr_Sunroof [Linear]
11	-0.09987	0.08353	-1.19559	Concept2_Pr_Radio [Linear]
12	-0.11465	0.08360	-1.37138	Concept2_Pr_Nav [Linear]
13	-1.82219	0.08951	-20.35628	Concept3_Pr_Base [Linear]
14	-0.11670	0.08828	-1.32196	Concept3_Pr_Alloy [Linear]
15	-0.14773	0.08949	-1.65077	Concept3_Pr_Sunroof [Linear]
16	0.08330	0.08823	0.94403	Concept3_Pr_Radio [Linear]
17	-0.19957	0.08900	-2.24245	Concept3_Pr_Nav [Linear]

This particular model leads to a log-likelihood fit of -5506.70344, with 17 parameters in the model.

We note a few problematic issues from this run. First, we expect all coefficients to be negative, since lowering prices should probably always increase choice likelihood for each alternative. **Concept3_Pr_Radio** has a slightly positive coefficient (though not statistically significant from zero, given its t-ratio of 0.94403). We also note that many of the coefficients are relatively close to zero. This may be expected, since not all the variables will have a strong effect on choice. Our personal opinion is that we'd like to see coefficients with the expected sign, and we'd like the t-tests to

suggest somewhere around 80% or higher significance for these coefficients (t-ratio of 1.28 absolute magnitude or higher). We could decide to re-run the model and specify a sign constraint (monotonicity constraint) on variable 16 to remove the reversal (or we could simply drop variable 16 from the model).

Also, it's interesting to note how similar the coefficients are for variables predicting their respective columns. For example, change to the base price (-\$3,000 to +\$3,000) has a coefficient of -1.86 when predicting concept 1, -1.89 when predicting concept 2, and -1.82 when predicting concept 3. Examining the other alternative-specific variables also indicates that their slopes are quite similar, irrespective of which concept they predict. This is a good indicator that different coefficients to predict each alternative (alternative-specific effects) may not be needed or justified.

We can return to the *Specify Models* tab to re-specify the model as generic effects rather than alternative-specific effects. To collapse alternative-specific into generic effects, we select the *Group and Collapse Variables...* button. We collapse the **Concept1_Pr_Base**, **Concept2_Pr_Base**, and **Concept3_Pr_Base** variables into a new variable that we'll call **Pr_Base** (see Figure 11.6).

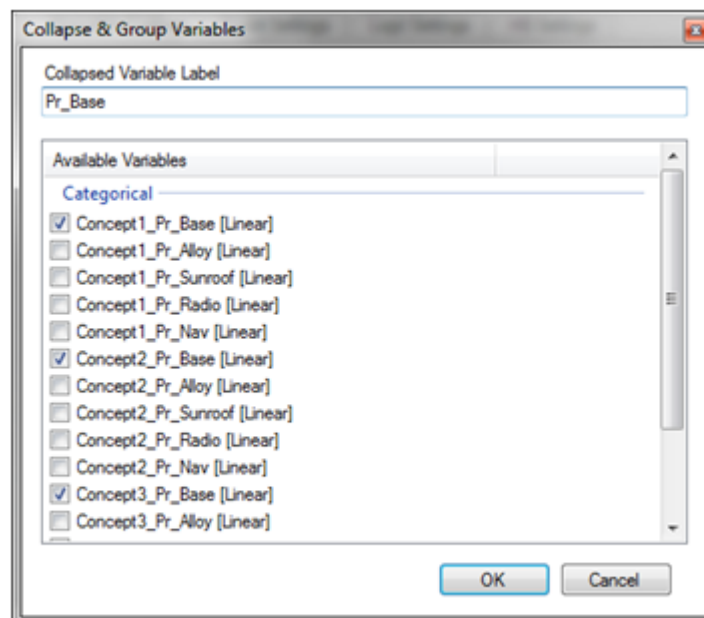


Figure 11.6 (Collapsing Alternative-Specific Variables into Generic Effects)

We similarly collapse other triads of variables into **Pr_Alloy**, **Pr_Sunroof**, **Pr_Radio**, and **Pr_Nav**. When we have finished collapsing these variables, the *Variable Codings* dialog looks like Figure 11.6 (note the grouping brackets with the new variable names in the grey margin at the far right side of the grid).

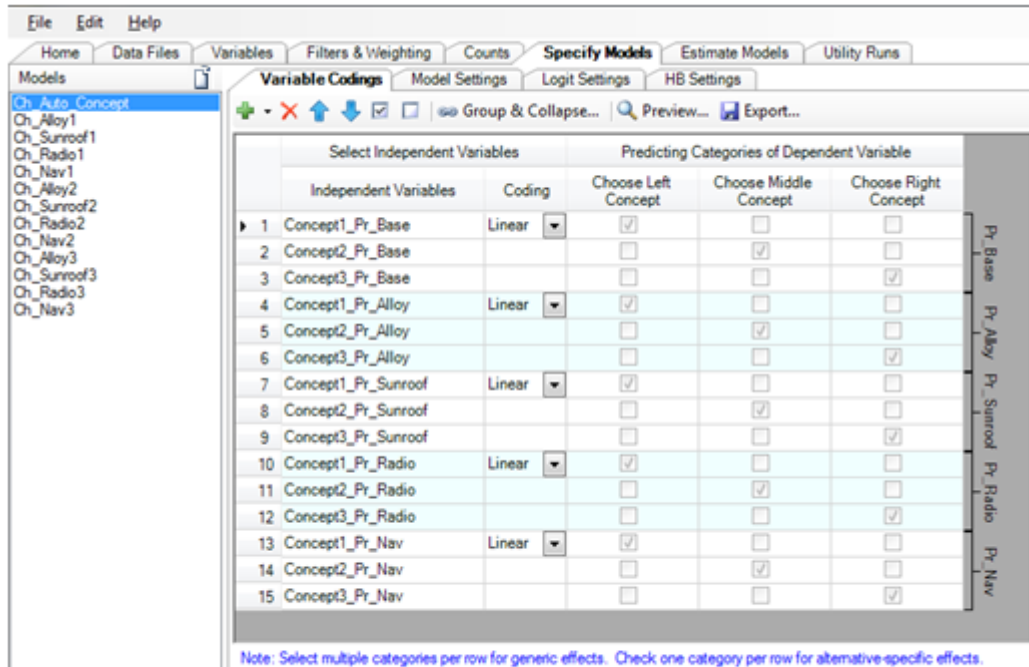


Figure 11.7 (The Variable Codings Dialog after Collapsing Alternative-Specific Variables into Generic Effects)

When we click the **Preview Design Matrix...** button we can see the more compact (parsimonious) generic effects model (Figure 11.8).

Preview Design Matrix

← Respondent: 1 →

Task/Alternative	ASC (1)	ASC (2)	Pr_Base [Linear]	Pr_Alloy [Linear]	Pr_Sunroof [Linear]	Pr_Radio [Linear]	Pr_Nav [Linear]	Response
1 Choose Left Concept	0	0	0.5	0.563	-0.464	0.167	-0.175	1
1 Choose Middle Concept	1	0	-0.5	-0.188	0.107	0.167	0.125	0
1 Choose Right Concept	0	1	-0.5	0.063	0.536	-0.5	0.525	0
2 Choose Left Concept	0	0	0.5	-0.438	0.536	-0.167	0.125	0
2 Choose Middle Concept	1	0	0	0.063	0.107	0.5	-0.475	0
2 Choose Right Concept	0	1	0	0.563	-0.464	-0.167	-0.475	1

Figure 11.8 (Preview Design Matrix, Generic Terms Model Specification)

Now, rather than 17 terms in the model, we have 7 terms. If we re-run the aggregate logit estimation, we obtain the following report:

```

Log-likelihood for this model =      -5509.26442
Log-likelihood for null model =      -7083.85204
                                     -----
                                   Difference =      1574.58761

Percent Certainty                    =      22.22784
Consistent Akaike Info Criterion    =     11086.92952
Chi Square                           =      3149.17523
Relative Chi Square                   =      449.88218

      Effect      Std Err      t Ratio      Variable
1      -0.58363      0.03352     -17.41184     ASC (1)
2      -0.81842      0.03581     -22.85611     ASC (2)
3      -1.85835      0.04104     -45.28187     Pr_Base [Linear]
4      -0.08272      0.04336     -1.90785     Pr_Alloy [Linear]
5      -0.21154      0.04359     -4.85240     Pr_Sunroof [Linear]
6      -0.02171      0.04309     -0.50377     Pr_Radio [Linear]
7      -0.19159      0.04317     -4.43844     Pr_Nav [Linear]

```

This condensed model leads to a log-likelihood fit of -5509.26442 (with 7 parameters estimated) which compares to the more expensive alternative-specific effects model with a fit of -5506.70344 (with 17 degrees of freedom). Following the 2 log-likelihood test, 2x the difference in log-likelihood is distributed as chi-square: $2 \times (5509.26442 - 5506.70344) = 5.12$, 10 d.f. (the difference in degrees of freedom between the two models). Referring to a Chi-square table, the critical value for 95% confidence is 18.31, meaning the Chi-square we observe from our 2 log-likelihood test falls well short of the critical value. We have not observed enough increase in fit from the alternative-specific model specification to justify the additional 10 parameters added to the model. The generic-effects model with just 7 terms provides about equally good fit, and happily shows no sign reversals (all coefficients are negative). Furthermore, the standard errors of the terms have decreased quite a bit from the alternative-specific effects model (precision has increased).

Respondent Choice of Options (Stage 2 Models)

In the previous section, we modeled stage 1 respondent choices. But, we need to specify models to account for what respondents select in terms of configured options (alloy wheels, sunroof, radio, and navigation system) *given* that they have selected a column (automobile). Earlier in this section, we already used the **Variables** dialog to specify that each automobile option (sunroof, alloy wheels, etc.) is conditional on the choice of the column (**Ch_Auto_Concept**). So, when we build models to predict which options the respondent will buy, we will only use those tasks in which the respondent has first selected the given column.

It is tempting to think that cross-effects (between-alternative effects) will be needed when modeling which options the respondent will configure on each automobile. After all, the likelihood of choosing alloy wheels for car #1 might be affected by the price of alloy wheels on car #2. But, we should remember that those competitive effects of alloy wheels were already principally accounted for in the first stage model. These second-stage models only use the tasks in the data file *given* that the respondent has selected the alternative/column of interest.

To review the characteristics of our dataset, we have 12 separate dependent variables that indicate whether each option was chosen (category 1) or not (category 2, off state). Let's consider the first of those dependent variables **Ch_Alloy1** (choice of alloy wheels on car #1). One conceptually reasonable model is that the likelihood of choosing alloy wheels for car #1 (concept #1) is a function of the desirability of alloy wheels (captured as an ASC), the base price for car #1, the price

of alloy wheels in car #1, and perhaps also the prices of the sunroof, radio, and navigation system for car #1. This model's results are given below:

```

Log-likelihood for this model =      -1488.29018
Log-likelihood for null model =      -2108.55372
                                     -----
Difference =                          620.26355

Percent Certainty                    =       29.41654
Consistent Akaike Info Criterion    =      3030.70198
Chi Square                           =      1240.52709
Relative Chi Square                   =       206.75452

```

	Effect	Std Err	t Ratio	Variable
1	-1.47158	0.05410	-27.20074	ASC (1)
2	-0.16387	0.12672	-1.29310	Concept1_Pr_Base [Linear]
3	-0.40714	0.13134	-3.09984	Concept1_Pr_Alloy [Linear]
4	0.29877	0.12296	2.42992	Concept1_Pr_Sunroof [Linear]
5	0.04719	0.12373	0.38143	Concept1_Pr_Radio [Linear]
6	-0.16616	0.12644	-1.31420	Concept1_Pr_Nav [Linear]

Only tasks in which Concept 1 (car #1) was chosen are included in this run (because we earlier indicated that **Ch_Alloy1** was conditional upon (only relevant when) **Ch_Auto_Concept=1**). As we examine the coefficients, we see the expected result that as the price of alloy wheels for car #1 increases, the likelihood of selecting alloy wheels on car #1 decreases (-0.40714). But, interesting to note is that alloy wheels and sunroof seem like substitutes, because the coefficient for the price of sunroof is a positive 0.29877 with a relatively large t-ratio of 2.42992 (suggesting that such a strong relationship is probably not due to chance). We have scant evidence that the price of radio is a significant effect (0.04719, t-ratio 0.38143), and whether in truth the effect is negative or positive. But, price of navigation system has a negative coefficient (-0.16616, indicating a possible complementary good) with a t-ratio of -1.31420 (better than 80% confidence). This is not a particularly strong effect, but it suggests a probable relationship.

A big challenge with these second-stage models is that we do not know ahead of time (other than for the own-effect) whether the price effects for options should be negative or positive. We don't know ahead of time their magnitude. So, it is challenging to know whether to aggressively prune the models and delete some of the variables or to just leave all the variables in, due to conceptual hypotheses and consistency of application across all 12 models. This model only has 6 parameters to estimate, and conceptually it makes sense that the likelihood of configuring options for car #1 depends on all car #1 option prices. Therefore, one could make an argument to retain all such variables in each model.

Eleven additional models would be specified and estimated for the remaining items on the menu. For sake of space, we don't display their logit estimation results here.

Combinatorial Second-Stage Model Specification

*The data set **CarCombinatorialDVs.csv** should be used for this section of the tutorial. The labels file for this study is named **CarCombinatorialDVs Labels.csv**.*

If a goal of our research was to predict the combinatorial selections of items that would be made by respondents, given specific pricing scenarios, then it would especially make sense to collapse

some of the dependent variables via combinatorial coding. For example, with four options under alternative 1 (car #1), there are $2^4 = 16$ possible ways to configure car #1.

- Alloy Wheels
- Moonroof/Sunroof
- XM Radio
- Navigation System

One possible listing of these 16 possible combinations is shown below:

	(1=chosen, 2=not chosen)			
Code	Alloy	Sunrf	Radio	NavSys
1	1	1	1	1
2	1	1	1	2
3	1	1	2	1
4	1	1	2	2
5	1	2	1	1
6	1	2	1	2
7	1	2	2	1
8	1	2	2	2
9	2	1	1	1
10	2	1	1	2
11	2	1	2	1
12	2	1	2	2
13	2	2	1	1
14	2	2	1	2
15	2	2	2	1
16	2	2	2	2

So, rather than code four separate dependent variables for **Ch_Alloy1**, **Ch_Sunroof1**, **Ch_Radio1**, and **Ch_Nav1**, we could recode them in the dataset as a single dependent variable **Ch_Options1** with 16 possible categories, where the 16th category is the not-chosen/off state.

Employing combinatorial coding of the choice of options within each alternative (column) leads to 4 rather than 13 total models to estimate:

- Stage 1 prediction of **Ch_Auto_Concept** (3 categories)
- Stage 2 prediction of **Ch_Options1** (16 categories)
- Stage 2 prediction of **Ch_Options2** (16 categories)
- Stage 2 prediction of **Ch_Options3** (16 categories)

The first (Stage 1) model is the same as we showed before. But, the remaining three Stage 2 models would be more complex than before, with more terms involved. This additional complexity would likely come with a payoff: the power to predict *specific combinatorial choices* made by respondents with greater accuracy. (Individual combinatorial choice predictions are only possible in MBC software if employing HB estimation).

One useful way to specify which independent variables predict categories of the combinatorial dependent variables is as follows:

<u>Variable</u>	<u>Predicting Categories of Ch Options</u>
Concept1_Pr_Alloy	1-8
Concept1_Pr_Sunroof	1-4, 9-12
Concept1_Pr_Radio	1-2, 5-6, 9-10, 13-14
Concept1_Pr_Nav	1, 3, 5, 7, 9, 11, 13, 15

Thus, for each of the three Stage 2 models, the independent variable matrix would be composed of 16-1 ASCs, plus 4 generic price effects, each relevant to predicting 8 categories of the dependent variable. Such models should work quite well, especially with HB estimation, for predicting the specific combinations of items each respondent will pick.

Results

For our presentation at the 2010 Sawtooth Software conference (Orme 2010), we reported predictive results for models extremely similar to the ones we've described in this section. The models built based on the 806 calibration respondents were able to predict aggregate choice likelihoods for the 823 holdout respondents with an R-squared of about 0.96.

13 Polytomous Logit

13.1 Polytomous Logit

In this section, we describe a type of discrete choice analysis called *Polytomous Logit* that MBC supports, but is not supported by our CBC software.

A polytomous logit discrete choice study involves a categorical dependent variable taking on three or more states. Often, for a given choice task, the independent variables are held constant across the different categories (alternatives) of the dependent variable, for example when the independent variables are respondent characteristics. In the model specification, alternative-specific beta weights are associated with each categorical outcome. So that the model converges with stable utility estimates, one category (outcome) of the dependent variable has its utility set to zero (an offstate for that category).

Consider the following example. Imagine you are a social scientist who is studying an existing data set involving choices that young unmarried women have made regarding pregnancies. You are researching the question of what will young unmarried women aged 21 and younger (with other varying demographic characteristics) do when facing a decision regarding pregnancy?

- a) Terminate the pregnancy
- b) Keep the baby
- c) Adopt out the baby

The data set consists of thousands of young unmarried women aged 21 and younger who became pregnant. In addition to the three categories of outcomes described above (the dependent variable), we have other independent variables describing each young woman in the data set:

Dependent Variable:

- 1= Terminate the pregnancy (Off_State)
- 2= Keep the baby
- 3= Adopt out the baby

IV1: Age

- 1= <17 years
- 2= 17-18 years
- 3= 19-21 years

IV2: IQ

(zero-centered coding, with 0= average (100) IQ score. To be treated as continuous variable)

IV3: Employment

- 1= Not employed
- 2= Part-time employed
- 3= Full-time employed

IV4: Parent_Status

1= Divorced/Separated

2= Married/Together

IV5: HH_Income_Parents

1= <\$25K

2= \$26K-\$40K

3= \$41K-\$79K

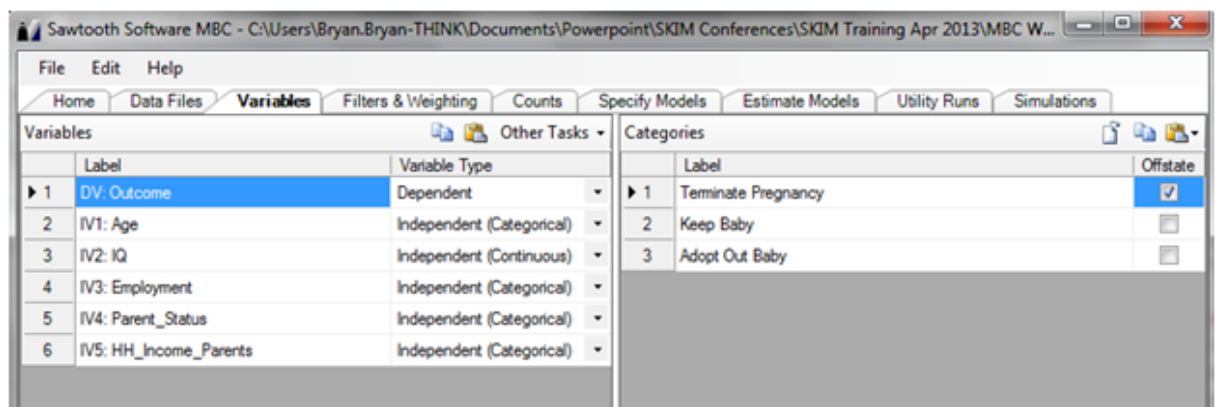
4= \$80K+

We display the beginning of the data set below, where each case (young woman) consists of a single row in the file.

A	B	C	D	E	F	G	H
CASEID	DV: Outcome	IV1: Age	IV2: IQ	IV3: Employment	IV4: Parent_Status	IV5: HH_Income_Parents	
1	1	2	-1	2	1	3	
2	1	2	-15	1	2	1	
3	3	1	-13	1	1	1	
4	2	2	-12	2	2	1	
5	1	2	21	3	1	3	
6	2	3	-5	3	1	1	
7	1	1	-30	1	2	1	
8	3	2	3	2	1	4	

Note that IV2 is the zero-centered IQ score for each young woman. Since the average IQ is 100, CASEID #1 has an IQ of 99. CASEID #2 has an IQ of 85.

Within the MBC software, the **Variables** tab should look as follows:



Note that the dependent variable is specified to include an offstate (for identification of the model and to obtain convergence). The offstate is the outcome category *Terminate Pregnancy* and is constrained during utility estimation to have a 0 utility. (We can assign any one of the three categories as the offstate; the model fit is the same.) Also note that we treat IV2: IQ as a *Continuous* independent variable (what we call User *Specified* variable within our popular CBC/HB software). We treat the other nominal independent variables as categorical.

In the *Specify Models* tab, we specify the following:

	Select Independent Variables		Predicting Categories of Dependent Variable	
	Independent Variables	Coding	Keep Baby	Adopt Out Baby
1	IV1: Age	▼ Part Worth ▼	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	IV2: IQ	▼ User-specified ▼	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	IV3: Employment	▼ Part Worth ▼	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	IV4: Parent_Status	▼ Part Worth ▼	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	IV5: HH_Income_Parents	▼ Part Worth ▼	<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	IV1: Age	▼ Part Worth ▼	<input type="checkbox"/>	<input checked="" type="checkbox"/>
7	IV2: IQ	▼ User-specified ▼	<input type="checkbox"/>	<input checked="" type="checkbox"/>
8	IV3: Employment	▼ Part Worth ▼	<input type="checkbox"/>	<input checked="" type="checkbox"/>
9	IV4: Parent_Status	▼ Part Worth ▼	<input type="checkbox"/>	<input checked="" type="checkbox"/>
▶ 10	IV5: HH_Income_Parents	▼ Part Worth ▼	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Note that the independent variables are entered *twice* within the design matrix. Separate sets of betas predict the category categories *Keep Baby* and *Adopt Out Baby*. *Terminate pregnancy* is the reference state and has a utility of zero for identification.

For this data set, since there is only one observation per case, aggregate logit is the appropriate model.

If the data consist of multiple observations per case, then we could consider running HB. Any respondent characteristics would be constant across tasks, so these may be used as covariates in HB. Only independent variables that vary across tasks for the respondent should be estimated as betas within HB.

For the pregnancy example we've been describing, once we run the aggregate logit model, we can run a market simulation that could answer questions such as the following:

What's the probability of each outcome for a young lady with the following characteristics who became pregnant?

- 17 year old girl
- IQ of 105 (specify "5" in simulator)
- Not employed
- Parents married/together
- Parents' HH income \$50K

The simulator reports predicted probabilities of the different decisions for a young lady with these characteristics, such as shown below:

Simulation Settings:	
Utility Runs:	DV Outcome (Logit) 3Jul2013 1548
HB Runs Use:	N/A
Respondent Filters:	N/A
Respondent Weighted by:	N/A
Exponent:	1.0000
Independent Variable Specification:	
IV1: Age	2
IV2: IQ	5
IV3: Employment	1
IV4: Parent_Status	2
IV5: HH_Income_Parents	3
Simulation Results:	
DV: Outcome	Share
Terminate Pregnancy	0.4500
Keep Baby	0.2880
Adopt Out Baby	0.2620

45.00% Terminate pregnancy
 28.80% Keep baby
 26.20% Adopt out baby

Although we have shown an example here involving the social sciences and decisions regarding pregnancy, there are many marketing applications. Traditional conjoint designs may be fielded where the respondent is shown multiple product concepts (one at a time) and for each product concept selects from three or more categories of action, such as:

Buy today for myself
 Buy today for another person
 Buy today for myself and another person
 Not buy, but would recommend to others
 Not buy, would not recommend to others

14 Appendix: Technical Details

14.1 Chi-Square Computations

MBC's counting routines use the Chi-Square statistic to test the strength of relationship between independent variables and choices on the menu (dependent variables). A Chi-Square statistic also is used to test whether such relationships are potentially non-linear. The Chi-Square statistic quantifies the amount of deviation between actual and expected frequencies of choice. If the differences are strong, then the likelihood of observing such differences due to chance is small.

Consider the following example for the Fast-Food study we've cited a number of times in this documentation. 681 respondents each completed 8 menu tasks, for a total of 5,448 choices. In 544 (9.99%) of those menu tasks, respondents selected Hamburger from the *a la carte* section of the menu.

The counts report within MBC reports the following:

Respondents included: 681
Total tasks included: 5,448

Dependent Var: Sandwich, Hamburger
Choice probability: 9.99%

Choice probability by levels of Indep Var: Hamburger_Prices

\$1.99	12.33%
\$2.29	9.69%
\$2.59	8.74%
\$2.89	9.18%

Relationship Chi Square	10.66
D.F.	3
P-value	0.01

Non-Linearity Chi Square	2.86
D.F.	2
P-value	0.24

14.2 Relationship Chi-Square

The "Relationship Chi-Square" is the Chi-Square statistic associated with the relationship between the independent variable and the dependent variable. If this value is high, it means that the price of the hamburger seems to have a strong influence on the choice likelihood of hamburger. We describe the computation below.

Chi-Square statics are computed using actual frequencies, rather than probabilities. So, we first examine the actual frequencies of the choices of hamburger in each category of prices for hamburger:

	Occurrence In Design	Actual Choice Frequencies
\$1.99	1,362	168
\$2.29	1,362	132
\$2.59	1,362	119
\$2.89	1,362	125

The random design we employed in the study (CBC's Shortcut design strategy) led to perfect balance in the number of times the hamburger was shown at each of the price points (1,362 times). When hamburger was offered at \$1.99 (1,362 total times), it was chosen 168 times ($168/1,362 = 12.3\%$), which leads to the 0.123 probability reported by MBC software.

Next, we compute the expected frequencies, given the null hypothesis (that there is no relationship between levels of the independent variable and choice of the dependent variable). If there were no relationship, we'd expect the choice frequencies of hamburger by level of hamburger price just to be proportional to their actual occurrences in the design. Since the levels of hamburger price occurred an equal number of times in the design, the expected choice frequency for each level is simply the total number of choices divided by four ($(168+132+119+125)/4 = 136$):

	Actual Frequencies	Expected Frequencies	$(f_a - f_e)^2/f_e$
\$1.99	168	136	7.52
\$2.29	132	136	0.12
\$2.59	119	136	2.13
\$2.89	125	136	0.89
		Sum:	10.66

Next, we compute $(f_a - f_e)^2/f_e$ for each row, where f_a is the actual frequency and f_e is the expected frequency. Summing across the rows, we obtain a total of 10.66, which is the Chi-Square statistic.

Next, we compute a p-value associated with that Chi-Square statistic, indicating the likelihood of observing a Chi-Square of that size due to chance. The p-value is 0.014, obtained from established Chi-Square tables, given 3 Degrees of Freedom (D.F. is equal to rows-1=3 degrees of freedom). If in truth there were no relationship between the price of the hamburger and choice of hamburger, and we fielded the study repeated times each time using a new sample, we would expect to see such a strong Chi-Square statistic only 1.4% of the time. In other words, we are 100% - 1.4% or 98.6% certain that the price of the hamburger has some sort of effect on choice likelihood of hamburger.

14.3 Non-Linearity Chi-Square

MBC software reports a second Chi-Square statistic, which tests whether the appropriate functional form of the logit utility for predicting choice is linear or not. If the test suggests that linearity is an appropriate assumption, then specifying a linear term will save degrees of freedom and lead to more parsimonious models. If the test suggests that it is probable that a non-linear function is more appropriate, then we suggest you investigate the log-linear or the part-worth functional forms for parameterizing the independent variable.

Prior to explaining how the Non-Linearity Chi-Square is calculated, we should first review principles of the logit model. If a linear utility is used, the outcome of the logit rule for predicted likelihood of choice will be non-linear. For example, consider the following linear utilities for price:

Price	Utility
\$1	1.0
\$2	0.5
\$3	0.0
\$4	-0.5
\$5	-1.0

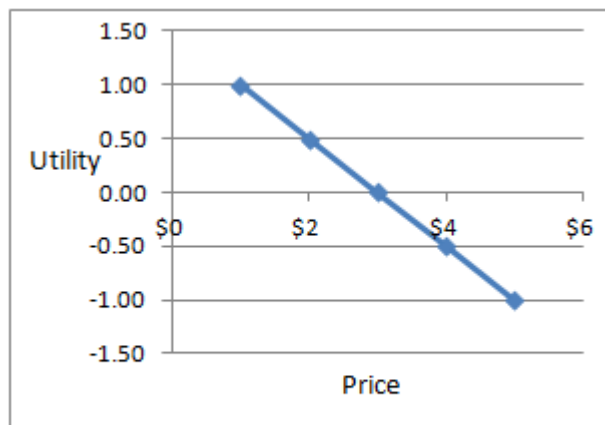


Figure A.1

When we use the logit rule to predict probabilities of choice at the different price points, we see that the resulting predicted choice probabilities are non-linear:

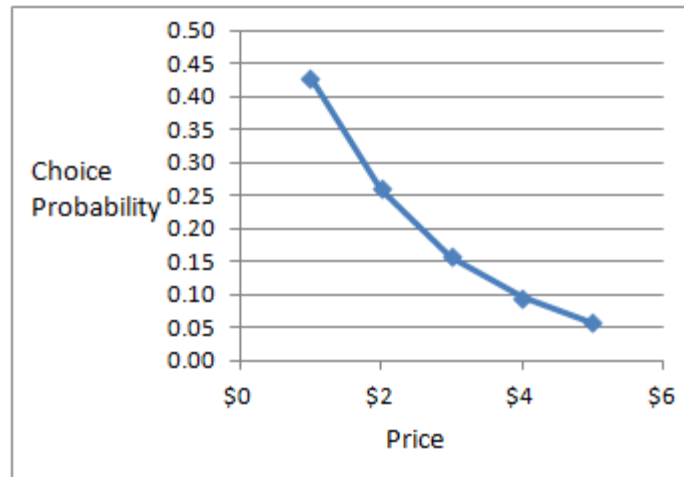


Figure A.2

Two statements about the relationship between logit utilities and predicted choice likelihoods are useful to remember:

- The antilog of utility is proportional to choice likelihood
- Utilities are linearly related to the natural log of choice likelihood

If you use MBC's counts module, plot counts probabilities for the effect of price on choice, and see a curve like Figure A.2, it is very likely that a *simple linear function* for the price variable is the correct functional form. If, however, you see a linear-looking counts probabilities plot, it is likely that a *non-linear* specification for the price variable is appropriate. This seems counterintuitive, but the non-linear transformation involved with applying the logit rule to predict probabilities of choice leads to this outcome.

Now that we have laid some groundwork, we are ready to describe the computation of the non-linearity Chi-Square statistic in MBC's counts module.

First, we calculate the choice likelihood at each of the levels of the independent variable:

Price Levels	Occurrence In Design	Actual Choice Frequencies	Choice Likelihood
\$1.99	1,362	168	$168/1362 = 0.1233$
\$2.99	1,350	132	$132/1350 = 0.0978$
\$2.59	1,356	119	$119/1356 = 0.0878$
\$2.89	1,397	125	$125/1397 = 0.0895$

The Expected Frequencies assuming a linear model are calculated by fitting a linear regression, where Y is a vector containing the choice likelihoods, $\ln(Y)$ contains the natural log of the values in Y , and the vector X contains the prices. We regress $\ln(Y)$ on X . Using the estimated slope and intercept from that linear regression, we can compute the expected probabilities (Y) at each price

point, and then multiply those expected probabilities by the occurrences in the design to compute the expected frequencies.

The Actual Frequencies, the Expected Frequencies (from the regression, assuming a linear relationship) and the Chi-Square computation are as follows:

	Actual Frequencies (X)	Expected Frequencies (Y)	$(f_a - f_e)^2/f_e$
\$1.99	168	157.78	0.66
\$2.29	132	140.50	0.51
\$2.59	119	126.79	0.48
\$2.89	125	117.36	0.50
		Sum:	2.15

Referring to the Chi-Square table (this time with $4-2 = 2$ D.F., because two degrees of freedom are used for fitting the earlier regression step), we compute a p-value of 0.34, indicating that we are only $100\% - 34\% = 66\%$ confident that appropriate functional form for the utility of price is non-linear (part-worth form). So, it appears that fitting with a simple linear model may be appropriate.

We can also use the procedure just described to compare the Expected Frequencies assuming a log-linear functional form to the actual frequencies (we regress $\ln(Y)$ on $\ln(X)$). This can help us determine whether a log-linear functional form may suffice in fitting non-linear data instead of the more costly part-worth function. Using a log-linear functional form requires just one estimated parameter, and can fit modestly concave curve functions.

14.4 Coding Choice Tasks

Each choice task is submitted to logit or HB estimation as a matrix, where the rows represent categories of the dependent variables (choice alternatives) and the columns represent independent variables that describe the alternatives.

Many dependent variables have a "not chosen" state. There are multiple ways to code this, and we have chosen to code the "not chosen" state as the default/reference level and to set its utility to zero. To do this, we employ dummy coding.

With a binary dependent variable where one level is the "not chosen" state, we construct the task as if there were two choice alternatives: a choice of the item (such as a hamburger) vs. choice of the "not chosen" alternative. In regular CBC, we would refer to the "not chosen" alternative as the "None." We capture the alternative specific constant (ASC) for the "chosen" state (preference for a hamburger) using a dummy coded independent variable column, labeled **var1** below.

	var1
Alternative #2 (hamburger)	1
Alternative #1 (None)	0

Other independent variables may describe the hamburger, such as its price and the prices of other items. These are coded as additional columns in the design matrix (such as var2 through v4 below), with the "None" alternative always containing zero for all columns. For example:

	var1	var2	var3	var4
Alternative #2 (hamburger)	1	-0.667	0.500	-0.234
Alternative #1 (None)	0	0.000	0.000	0.000

Some menus involve dependent variables that take on more than one possible mutually exclusive state. Consider the choice of one size of French fries, vs. the choice of no French fries at all. The dependent variable is coded in the original data file (the .CSV file) as:

- 1 = No French fries
- 2 = Small French fries
- 3 = Medium French fries
- 4 = Large French fries

When MBC software codes the alternative-specific constants for these choice alternatives, it is coded as k-1 columns in the independent variable matrix, to avoid linear dependency. The 4-1=3 columns look like:

	var1	var2	var3
Small French fries	1	0	0
Medium French fries	0	1	0
Large French fries	0	0	1
No French fries (None)	0	0	0

Other independent variables may describe the alternatives, such as the prices for the different sizes of French fries, and the prices of other items on the menu. These are coded as additional

columns in the design matrix (such as var4 and var5 below), with the "None" alternative always containing zero for all columns.

	var1	var2	var3	var4	var5
Small French fries	1	0	0	-0.242	0.000
Medium French fries	0	1	0	0.375	0.000
Large French fries	0	0	1	0.000	0.655
No French fries (None)	0	0	0	0.000	0.000

In the example above, var4 is a price effect that is common (generic) to both Small and Medium French fries. Var5 is an alternative-specific effect to Large French fries.

Note that we choose zero-centered coding for continuous independent variables such as price (as described earlier in this appendix). If independent variables are coded as "part-worth" functional form, we employ dummy coding, where the reference level is coded as a row of 0s. The utility of the reference level is therefore set to zero. We choose the "off state" as the reference level if an off state level is indicated. Otherwise, we arbitrarily choose the first level of the independent variable as the reference level.

14.5 Prior Covariance Matrix for HB Estimation

In our work with the CBC/HB system, we have found that for categorical independent variables, there can be a bias in the utility estimate of the "reference" level under HB estimation. The greater the number of levels of the independent variable, and the fewer choice tasks available, the greater the bias (Orme and Lenk, 2004, Lenk and Orme, 2009). Peter Lenk provided a solution that we have used very successfully within CBC/HB, and it involves some adjustments to the Prior Covariance Matrix. We employ those same adjustments for the dummy-coded independent variables within the MBC software (and also for the Alternative-Specific Constants, ASCs). This happens automatically, and we refer the interested reader to the relevant documentation on the prior covariance matrix within the CBC/HB documentation.

14.6 Coding Linear and Log-Linear Functions

MBC software can estimate the effect of continuous independent variables (such as price) as linear or log-linear functions. If the true shape of the preference function is essentially linear or log-linear, then coding the function in this manner reduces the number of independent variables to be estimated in the model, cuts the time to estimate models, can reduce the likelihood of observing reversals, and can lead to models with higher predictive validity.

Estimating linear or log-linear functions leads to a single parameter (beta coefficient) to be estimated rather than multiple parameters (dummy-coded part-worth functions) that estimate the utility at each price level. In the case of a linear function, the prices for the menu items might be entered as a single column of values in the design matrix, with values such as 1.50, 1.75, 2.00, and 2.25 (MBC further transforms such price values as described further below). For log-linear functions, the natural log of the price values may be used in the design matrix. MBC manages the design matrix for the user, so the data processing and natural log transformations are done automatically (with further zero-centering and normalization steps described below).

In standard regression or logit models, the absolute scaling of independent variables is not consequential to the convergence, fit of the model, or simulator predictions. Whether we use price values of 10, 20, or 30 in the independent variable matrix or 1.0, 2.0, or 3.0, the fit of the model will be the same; only the magnitude of the coefficient will be different, by a factor of 10. But, when using HB, the absolute scaling of the independent variables will affect the convergence of the model and accuracy of the coefficients. Because our HB routine assumes prior means of 0 and prior variances that are uniform across the parameters, it makes it difficult for HB to converge properly if the coded values for independent variables have vastly different variance. So that you can obtain consistently good results and relatively fast convergence, MBC software automatically rescales the linear and log-linear values in the independent variable matrix to be zero-centered, with a range of 1.

For example, consider an independent variable (price) that was shown to respondents as \$10000, \$20000, and \$30000. The user specifies that certain *Level Values* are to be associated with those prices and used in the market simulator to specify menu scenarios. Let's assume the user has associated the values 10000, 20000, 30000 with the three levels of price.

Linear Coding Example

Level #	Shown to Respondents	Assigned Level Values	Transformed Values for IV Matrix
1	\$10,000	10000	-0.5
2	\$20,000	20000	0.0
3	\$30,000	30000	0.5

When the internal files are prepared (built) for HB analysis, values of -0.5, 0.0, and 0.5 are used in the independent variable (IV) matrix rather than 10000, 20000, and 30000. (Accomplished by zero-centering the values and giving them a range of 1.0.) This will obtain faster convergence and unbiased estimates for HB. During market simulations, the user specifies values for menu items on the original 10000 to 30000 scale, but the simulator knows to transform those values back to the -

0.5 to 0.5 scale prior to multiplying the X values by the utility coefficients and using those utilities for logit rule simulation computations.

Below, we show the final transformed values used in the independent variable matrix when log-linear coding is selected:

Log-Linear Coding Example

Level #	Shown to Respondents	Assigned Level Values	Transformed Values for IV Matrix
1	\$10,000	10000	-0.544
2	\$20,000	20000	0.088
3	\$30,000	30000	0.457

The transformed values used in the independent variable matrix are computed as follows. First, we take the natural log of the assigned level values:

Assigned Level Values	Natural Log
10000	9.210
20000	9.903
30000	10.309

Next, we zero-center the transformed values, by subtracting the mean (9.808) from each of the values:

Assigned Level Values	Natural Log	Centered Values
10000	9.210	-0.597
20000	9.903	0.096
30000	10.309	0.501

Last, we have found through years of working with HB on CBC datasets that our HB routine converges quicker if the range of the independent variables is a small number, such as 1.0. Currently, the range is 0.501- -0.597 = 1.099. So, we can obtain the desired range if we multiply the centered values above by a factor of 1.0/1.099, leading to the following Final Rescaled Values:

Assigned Level Values	Natural Log	Centered Values	Final Rescaled Values
10000	9.210	-0.597	-0.544
20000	9.903	0.096	0.088
30000	10.309	0.501	0.457

The Final Rescaled Values remain zero-centered (sum to zero), and they have a range of 0.457 - -0.544 = 1.0. (There is some rounding error in representing these with 3 decimal places of precision.)

Using transformed values with a range exactly of 1.0 isn't an absolute requirement. In our work, any range scaled in the single digits works reasonably well within our HB estimation (where we've assumed means of zero and uniform prior variance across the parameters). A range of 1.0 seems logical, since any dummy-coded parameters also range from 0 to 1 and have a range of 1.0.

14.7 Utility Constraints

MBC studies frequently include independent variables for which almost everyone would agree regarding whether it has a negative or positive relationship. However, estimated parameters sometimes turn out not to have those expected orders. This can be a problem, since independent variables with the wrong signs, are likely to yield nonsense results and can undermine users' confidence.

MBC provides the capability of enforcing constraints on orders of part worth (dummy-coded) functions, on signs of linear (or loglinear) coefficients, and between coefficients from continuous (user-specified) coding. The same constraints are applied for all respondents, so constraints should only be used for variables that have unambiguous a-priori preference orders.

Evidence to date suggests that constraints can be useful when the researcher is primarily interested in the prediction of individual choices, as measured by hit rates for holdout choice tasks. However, constraints appear to be less useful, and sometimes can be harmful, if the researcher is primarily interested in making aggregate predictions, such as predictions of shares of choices.

Wittink (2000) pointed out that constraints can be expected to reduce variance at the expense of increasing bias. He observed that hit rates are sensitive to both bias and variance, so trading a large amount of variance for a small amount of bias is likely to improve hit rates. He also observed that aggregate share predictions are mostly sensitive to bias since random error is likely to average out, and share predictions are therefore less likely to be improved by constraints.

With MBC software, we employ the same method of constraints (Simultaneous Tying) as has been employed for over a decade within CBC/HB software.

Simultaneous Tying

This method features a change of variables between the "upper" and "lower" parts of the HB model. For the upper model, we assume that each individual has a vector of (unconstrained) utility parameters, with distribution:

$$\mathbf{b}_i \sim \text{Normal}(\mathbf{a}, \mathbf{D})$$

where:

\mathbf{b}_i = unconstrained parameters for the i th individual

\mathbf{a} = means of the distribution of unconstrained parameters

\mathbf{D} = variances and covariances of the distribution of unconstrained parameters

For the lower model, we assume each individual has a set of constrained parameters, \mathbf{b}_i where \mathbf{b}_i is obtained by recursively tying each pair of elements of \mathbf{b}_i that violate the specified order constraints (or setting to zero a coefficient with the wrong sign), and the probability of the i th individual choosing the k th alternative in a particular task is

$$p_k = \exp(x_k' b_i) / \sum_j \exp(x_j' b_i)$$

With this model, we consider two sets of parameters for each respondent: unconstrained and constrained. The unconstrained parameters are assumed to be distributed normally in the population, and are used in the upper model. However, the constrained parameters are used in the lower model to evaluate likelihoods.

For dummy-coded functions, we speak of "recursively tying" because, if there are several levels within an attribute, tying two values to satisfy one constraint may lead to the violation of another. The algorithm cycles through the constraints repeatedly until they are all satisfied.

When constraints are in force, the estimates of population means and covariances are based on the unconstrained parameters. However, since the constrained parameters are of primary interest, we report the average of the constrained parameters on-screen.

When constraints are employed, two kinds of changes can be expected in the on-screen output:

Measures of fit (Pct. Cert. and RLH) will be **decreased**. Constraints always decrease the goodness-of-fit for the sample in which estimation is done. This is accepted in the hope that the constrained solution will work better for predictions in new choice situations.

Measures of scale (Avg. Variance and Parameter RMS), which are based on unconstrained parameters, will be **increased**. The constrained parameters have less variance than the unconstrained parameters, because they are produced by tying unconstrained values. Since constrained parameters are used to assess the fit of the model to the data (by computing likelihood), the constrained values take on the "correct" scaling, and the unconstrained values therefore have greater variance.

You may impose constraints on dummy-coded, linear, or loglinear coded independent variables.

15 References

15.1 References

Bakken, David G. and Len Bayer (2001), "Increasing the Value of Choice-Based Conjoint with 'Build Your Own' Configuration Questions," Sawtooth Software Conference Proceedings, pp 99-110.

Bakken, David G. and Megan Kaiser Bond (2004), "Estimating Preferences for Product Bundles vs. *a la carte* Choices," Sawtooth Software Conference Proceedings, pp 123-134.

Cohen, Steven H. and John C. Liechty (2007), "Have it Your Way: Menu-based conjoint analysis helps marketers understand mass customization," Marketing Research Magazine, Fall 2007, pp 28-34.

Lenk, Peter, and Bryan Orme (2009), "The Value of Informative Priors in Bayesian Inference with Sparse Data," Journal of Marketing Research, Volume 46, Number 6.

Liechty, John, Venkatram Ramaswamy, and Steven H. Cohen (2001), "Choice Menus for Mass Customization: An Experimental Approach for Analyzing Customer Demand with an Application to a Web-Based Information Service," Journal of Marketing Research, May 2001, pp 183-196.

Moore, Chris (2010), "Analysing Pick 'n Mix Menus via Choice Models to Optimise the Client Portfolio—Theory and Case Study," Sawtooth Software Conference Proceedings, Sequim, WA.

Orme, Bryan and Peter Lenk, (2004), "HB Estimation for "Sparse" Data Sets: The Priors Can Matter," Advanced Research Techniques Forum, American Marketing Association.

Orme, Bryan (2010), "Menu-Based Choice Modeling Using Traditional Tools," Sawtooth Software Conference Proceedings, Sequim, WA.

Index**- A -**

ACA 2
Alternative specific constants (ASCs) 71, 89, 96, 157
Alternative specific effects 67, 78, 133
Alternative-specific prices 17
Availability designs 18
Avg variance 99

- B -

Blocks 20, 24
Bundling 5, 27
BYO 5

- C -

CBC 2, 5, 29, 67
Chi-square 87, 151, 152
Collapsing variables 78, 133
Combinatorial counting analysis 50
Combinatorial dependent variables 41, 56, 65, 76, 112, 130, 133
Combinatorial simulations 123, 124, 125, 127
Complements 5, 39, 48, 53, 62
Conditional dependent variables 36, 59, 85, 105, 129, 133
Configurator 5
Constraints 89, 96, 163
Continuous variables 35
Convergence 90, 91, 97, 99
Counting analysis 39, 41, 111, 133
Cross effects 5, 41, 45, 48, 49, 55
csv layout 31
CVA 20

- D -

Data file format 31
Dependent variables 33, 35, 41, 55, 67, 91, 99, 125
Design testing 55
Draws 97, 104, 124, 125, 163
Dummy coding 22, 45, 157

- E -

Effect 91
Estimate models 99
Excel simulator 106, 121

Exhaustive alternatives 76, 133
Experimental design 18
Exponent 106

- F -

First choice rule 125
Fractional factorial 18, 20

- G -

Generic effects 67, 78, 133

- H -

HB 95, 96, 97
Hit rates 163
Holdout tasks 49, 111, 129

- I -

Independent variables 31, 33, 35, 41, 67
Individual results 106

- L -

Labels file 33
Linear functions 45, 160
Lock step pricing 24
Log likelihood 91
Log linear functions 45, 160
Logit 5, 85, 87, 89, 90, 91, 94, 104, 133
Logit rule 154
Log-likelihood 87

- M -

Market simulations 103, 104, 105, 106
MaxDiff 2
Missing values 31, 36

- N -

Netting dependent variables 112
Non-linearity 22
Nonlinearity Chi-square 45, 154

- O -

Off state 35, 71, 78
Overfitting 41
Overheating 99
Own effects 41, 45, 48, 49, 55

- P -

Parameter RMS 99, 163
Part worth functions 45
Percent certainty 91, 99
Pharma studies 5
Point estimates 104, 124
Polytomous logit 147
Premium pricing 27
Preview design matrix 71
Prior covariance matrix 159
Prior degrees of freedom 97
Prior variance 97
Probit 5
Prohibitions 29

- R -

Randomized design plans 24
References 165
Relationship Chi-square 41, 152
Reversals 41, 133
RLH 99, 163
Run manager 94

- S -

Sample size 22
Scale Factor 106
Segmentation 95
Segmentation variables 37
Sensitivity analysis 113
Serial cross effects model 133
Simulator for Excel 121
Simultaneous tying 163
Skip factor 97
Standard error 91
Step size 90
Substitutes 5, 39, 48, 53, 62
Synthetic data 29, 55

- T -

T test 87, 91, 133
Testing designs 29
Thermal event 99
Tutorial 133
Two log-likelihood test 87
Two-stage models 5, 14, 59, 133

- U -

Utility constraints 48, 89, 163

- V -

Validation 14, 39, 49, 111, 129, 130
Variance 163

- W -

Weighted draws 125, 127, 129, 130
Weights 37