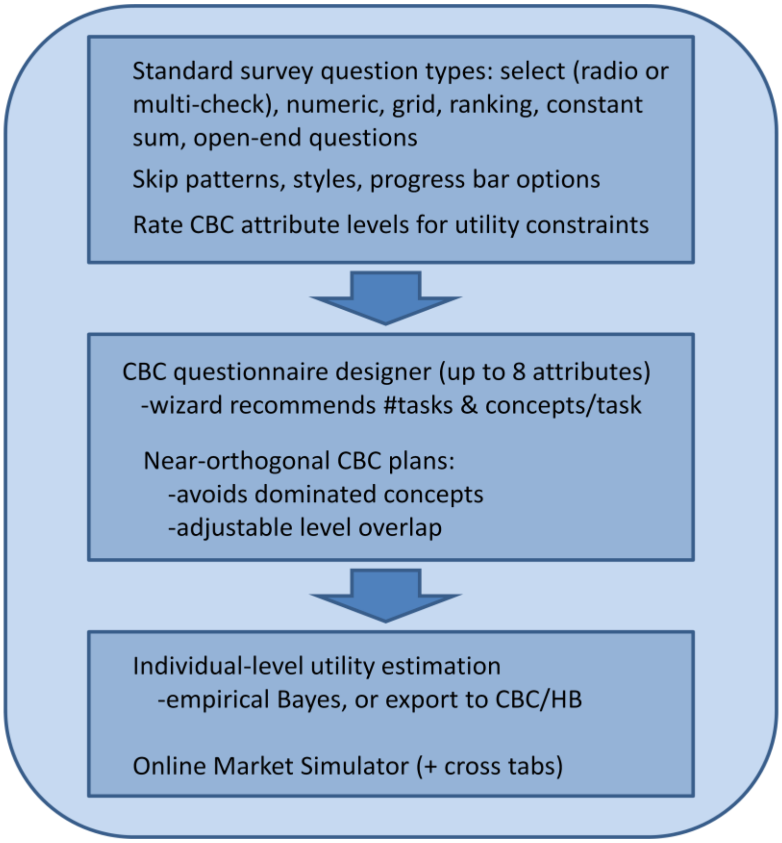
Discover-CBC: How and Why It Differs from Lighthouse Studio's CBC Software

Abstract and Overview

Sawtooth Software's web-based (SaaS--Software as a Service) questionnaire authoring tool is streamlined, attractive, and user-friendly. We're calling this web-based platform "Discover" and the CBC component within it "Discover-CBC." Discover-CBC includes the essential aspects of Sawtooth Software's Lighthouse Studio CBC software and also includes some new features for experimental design. Users will find Discover-CBC easy to use, they'll be able to collaborate better in teams, and the results should be nearly indistinguishable from our CBC package in Lighthouse Studio. All aspects, from questionnaire authoring, designing CBC tasks, fielding the study, and analyzing the results are managed within the intuitive, browser-based interface.

Discover-CBC Capabilities and Project Flow



Standard survey question types: select (radio or multi-check), numeric, grid, ranking, constant sum, open-end questions

Skip patterns, styles, progress bar options

Rate CBC attribute levels for utility constraints

CBC questionnaire designer (up to 8 attributes)
-wizard recommends #tasks & concepts/task

Near-orthogonal CBC plans:
-avoids dominated concepts
-adjustable level overlap

Individual-level utility estimation
-empirical Bayes, or export to CBC/HB

Online Market Simulator (+ cross tabs)

Background and Motivation

Over the last 15 years, CBC (Choice-Based Conjoint) has become the most popular conjoint-related methodology worldwide. Conjoint analysis is considered by many the premier market research technique for new product design, pricing, repositioning, line-extension, and market segmentation. It also finds wide application in fields such as health and environmental economics, transportation planning, employee research, and even litigation.

Software-as-a-Service (SaaS) is convenient and has advantages for many applications and markets (SaaS is also known as on-demand or browser-based software). We released our SaaS-delivered CBC tool, called Discover-CBC, based on the same theory and using similar algorithms as our earlier Windows-based CBC product (within Lighthouse Studio).

We have taken best practices from our Lighthouse Studio CBC product and incorporated them within Discover-CBC. We've also been able to incorporate some new things that we've learned over the years that aren't available within the Lighthouse Studio CBC software. So, in many ways, Discover-CBC represents a step forward.

We assume the reader already is familiar with CBC, so we skip the general theory and details that are outlined in our CBC Technical Paper, available for download at:
http://www.sawtoothsoftware.com/download/techpap/cbctech.pdf

A Web-Based (SaaS) Software Platform

Our SaaS software (Software as a Service) resides on a web server provided by a reliable third party. The user does not need to install any software and simply logs into the service using a browser connected to the internet. All questionnaire authoring, data collection, and analysis (including tabulations and market simulations) occur on the web server. Results (such as raw data and conjoint utilities) may be downloaded to the user's local device and opened in a program such as Excel.

Discover-CBC Questionnaires

The software lets the questionnaire author compose standard questions in addition to the CBC tasks, such as text screens (requiring no response), select-questions (radio and check box), numeric, grid-type, rank, constant sum, and open-end questions[1]. The researcher can add skip patterns, a progress bar, and choose from available styles. We also provide graphical tabulation analysis[2] for summarizing and cross-tabbing the results for these non-conjoint questions. These questions are automatically made

[1] The Perl code that formats the questionnaire for browsers and stores the data is the same as used in our Lighthouse Studio system.

[2] Lighthouse Studio's cross-tabulation functionality as offered in the Online Data Management Module is used within the Discover platform.

available as filters and banner points within the market simulator[3] (the what-if simulator for the CBC data).

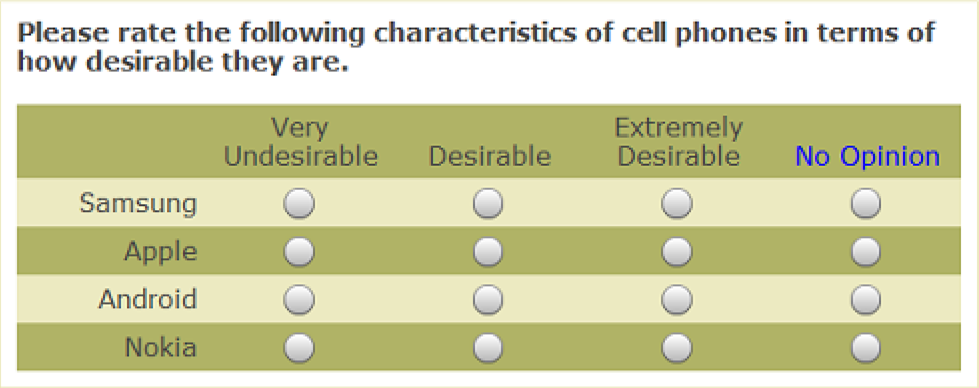
The CBC questions are limited to:

- 8 attributes (factors)
- 15 levels per attribute
- No more than 8 concepts (profiles) per choice task (set)
- No more than 30 tasks (sets) per respondent

Most questionnaire authors will want to hone their attribute lists to cover about two to six attributes with no more than about two to six levels per attribute. If you follow these guidelines, most CBC questionnaires will require about 10 to 20 choice tasks.

Ratings of Levels within Attributes

Upfront, the questionnaire author should specify which attributes have known (*a priori*) utility order (such as higher speeds always being preferred to lower speeds or lower prices always preferred to higher prices). For attributes *without* known utility order (brand, color, style, flavor), the software adds ratings questions prior the CBC tasks.



Please rate the following characteristics of cell phones in terms of how desirable they are.

	Very Undesirable	Desirable	Extremely Desirable	No Opinion
Samsung	○	○	○	○
Apple	○	○	○	○
Android	○	○	○	○
Nokia	○	○	○	○

We use a rating scale with only a few scale points as recommended by Kevin Lattery (2009). This scale allows respondents to differentiate among levels using three broad categories of preference. It also offers a “no opinion” rating, so we don’t force respondents to rate levels they have no opinion about.

The *a priori* and stated preference orders of levels within attributes:

1. Serve as utility constraints (monotonicity constraints) to permit robust individual utility estimation (Logit with data augmentation via empirical Bayes).

[3] We use the same simulator as provided by our Online Market Simulator, but it is seamlessly integrated within Discover-CBC.

2. Provide individual-level preference information so we can try to avoid[4] dominated concepts within the on-the-fly experimental design. (The definition of dominated concepts is described in Appendix B).

Note that we do *not* ask respondents to rate the overall *importance* of attributes as was done with the original ACA (Adaptive Conjoint Analysis) system. The importance of attributes is estimated entirely based on the choices within the CBC tasks.

Choosing the Number of CBC Tasks and Concepts per Task

Discover-CBC includes a recommendation wizard that suggests an appropriate number of tasks and concepts per task to ask, given the specific attribute and level array specified by the questionnaire author. The recommendations are based on Logit theory (specifically, the computation of standard errors), knowledge drawn from a large variety of experimental designs for CBC, plus our many years of experience regarding what works with real respondents. The author may override the suggestions.

By default, a "None" concept is added to each choice task. The questionnaire author may remove the None option if desired.

Experimental Design

Our Discover-CBC designs have an adaptive element, since we try to avoid dominated concepts (based on *a priori* rankings of attributes plus the additional rating information respondents provide about attributes such as brand and color). Thus, we wait until we observe the respondent's preferences before generating the designs on-the-fly. A respondent's design is typically generated in less than ½ a second, even when there is substantial load on the server. Our designs use a random starting point followed by many iterations of level-relabeling and swapping across all attribute levels, concepts, and tasks to satisfy multiple simultaneous goals: one-way level balance within attributes, two-way level balance between attributes, target degree of level overlap (specified by the author), avoiding dominated concepts, and avoiding prohibited pairs of levels between attributes (specified by the author). These designs are near-orthogonal and have high relative D-efficiency, because designs that are one- and two-way level-balanced also are near-orthogonal and statistically efficient.

Discover-CBC supports prohibitions between attributes. But, we place limits on the number and pattern of prohibitions that may be specified so that design efficiency is not overly compromised. If prohibitions are specified, the software will recommend a few additional choice tasks be asked to counteract the loss in statistical efficiency. Discover-CBC doesn't include a *test design* step, since the designs are generated on-the-fly during data collection. Rather, we display a warning if the user tries to specify too many prohibitions or a particularly damaging pattern of prohibitions. We also warn the user if too few tasks

[4] The experimental designer simultaneously targets multiple goals, one of which is to avoid dominated concepts. If there are conflicting goals that cannot be resolved, some dominated concepts may result. The presence of dominated concepts does not invalidate the results. Standard CBC designs typically include many dominated concepts for individuals.

are specified. *Essentially, we make it nearly impossible for users to field a study that would yield poor quality utility estimates.*

New with this software is the ability for the author to specify how much “level overlap” to use per attribute. By default, all attributes have a modest degree of level overlap (level repeating within a task). Modest level overlap has been found to be beneficial for the following reasons:

1. Allowing levels to repeat on occasion within the choice task more deeply probes respondents’ preferences when respondents employ non-compensatory (cut-off) rules—which researchers have found to be a typical respondent behavior. If using minimal overlap designs (where levels never repeat within task), respondents who focus on just a few “must have” levels are not challenged to provide very deep information about their preferences. For example, a respondent who must have a “red” product can easily find the one red product to pick per task if there is no level overlap. In that case, we only learn about the respondent’s preference for red, since (given our design) all other attribute levels cannot affect choice. However, if overlap is imposed and two red products are shown within a task, the respondent then must decide which *other* attribute levels matter beyond the must-have red characteristic.

2. Allowing levels to repeat improves the statistical efficiency of interaction effects[5].

If there are fewer levels within an attribute than concepts to show per task, level overlap necessarily must occur. The author can request an additional measure of level overlap on such attributes if desired.

Comparison of Discover-CBC Experimental Designs to Lighthouse Studio’s CBC

It’s natural to wonder how well the fast relabeling/swapping procedure implemented in Discover-CBC compares to Lighthouse Studio’s CBC designer. After all, Lighthouse Studio’s CBC designer (Complete Enumeration algorithm) has been in use since the first release of CBC in 1993. Various researchers have done comparative tests of design quality for it versus other going methods (SAS, JMP, R, etc.). Results have generally been favorable for CBC’s designs in terms of D-efficiency (a common standard for evaluating the efficiency of CBC designs). The Complete Enumeration approach nears optimal efficiency for main effects estimation. Even though Discover-CBC can do new things the standard Lighthouse Studio version of CBC cannot (e.g. adjust the amount of level overlap per attribute, avoid dominated concepts at the individual level), we sought an apples-to-apples comparison. Therefore, we tested minimal overlap designs without avoiding dominated concepts (we set the prior level ratings to have equal preferences for Discover-CBC). We tested four different designs across the two design algorithms:

1. Symmetric design with no prohibitions:
   5x5x5x5x5x5 design, 5 concepts per task, 8 tasks

2. Symmetric design with prohibitions:

[5] Discover-CBC software will not automatically estimate interaction effects. However, advanced users may export their data to a CBC/HB ready data file for more advanced analysis (under the Advanced menu).

5x5x5x5x5x5 design, 5 concepts per task, 8 tasks
Prohibitions: (1,2 & 2,1), (1,2 & 2,2), (2,1 & 3,1), (2,1 & 3,2), (2,2 & 3,1), (2,2 & 3,2), (3,1 & 4,1), (3,1 & 4,2)

3. Asymmetric design with no prohibitions:
   2x3x4x5x6x7 design, 5 concepts per task, 8 tasks

4. Asymmetric design with prohibitions:
   2x3x4x5x6x7 design, 5 concepts per task, 8 tasks
   Prohibitions: (1,2 & 2,1), (2,1 & 3,1), (2,1 & 3,2), (2,2 & 3,2), (3,1 & 4,1), (3,1 & 4,2)

To ensure stable results, we generated 15 individual-respondent (1-version) designs using unique design seeds. We averaged the relative D-efficiency across the 15 replicates. We indexed the results so that the Lighthouse Studio CBC designer (Complete Enumeration) had a relative D-efficiency of 100% for comparisons.

Relative Efficiency of Designs

(Where Lighthouse Studio CBC is indexed to 100)

Design	Lighthouse CBC	Discover-CBC
Symmetric design, no prohibitions	100	100
Symmetric design, prohibitions	100	105
Asymmetric design, no prohibitions	100	101
Asymmetric design, prohibitions	100	104

The results are very favorable for the quick method that Discover-CBC implements. The relabeling/swapping routine matches or beats the design efficiency of Lighthouse Studio's CBC (Complete Enumeration) in all four cases. The two designs involving prohibitions led to the greatest advantage, where it is between 4 to 5% more efficient than the Lighthouse Studio CBC designer.

The results initially surprised us, because the Lighthouse Studio Complete Enumeration routine examines *all* possible combinations of attribute levels within the task to maximize level balance and orthogonality. However, Discover-CBC uses a heuristic relabeling/swapping routine that doesn't involve exhaustively examining all possible combinations. So, how can it beat Complete Enumeration? The secret of the success of the relabeling/swapping routine is that it goes backward and forward across all tasks in the design to investigate improvements that could be made by changing levels. In contrast, the Complete Enumeration routine only marches forward. Once a task has been designed, it never revisits that task to investigate other level combinations. Only the current task being designed is manipulated across all possible combinations, given the *fixed* aspects of the tasks already set in place.

In terms of speed, the Complete Enumeration and relabeling/swapping routines are both extremely fast for small to mid-sized designs, generating a design for each respondent in a fraction of a second. However, with bigger designs, the Complete Enumeration strategy becomes extremely slow:

Comparison of Speed to Generate Individual Respondent Designs

with 8 Tasks, 5 Concepts per Task

Design	Lighthouse CBC	Discover-CBC
3x3x3x3	<0.5 seconds	<0.5 seconds
4x4x4x4x4x4	<0.5 seconds	<0.5 seconds
5x5x5x5x5x5x5	8 seconds	<0.5 seconds
6x6x6x6x6x6x6	32 seconds	<0.5 seconds
7x7x7x7x7x7x7x7	10 minutes	<0.5 seconds

For implementing customized designs on the fly (that can avoid dominated concepts), the rapid relabeling/swapping routine implemented within Discover-CBC is very advantageous. It is just as good or better than CBC's Complete Enumeration Strategy and much faster for the largest designs. Plus, the author can adjust the amount of level overlap desired per attribute.

One might also wonder how the Discover-CBC designer compares to Lighthouse Studio's CBC designer under Balanced Overlap (the default CBC design method). We can make such a direct comparison only after adjusting the degree of level overlap per attribute in Discover-CBC to approximate the same degree of level overlap as implemented in Balanced Overlap. Balanced Overlap uses the same algorithm as Complete Enumeration; but relaxes the penalty involved if overlap is observed. So far, we have found that the same advantages outlined above for the relabeling/swapping routine also apply for designs with level overlap. The Discover-CBC designs with level overlap tend to match or be slightly more efficient than the Balanced Overlap routine. And, its designs more consistently achieve a specific degree of level overlap for each attribute.

Utility Estimation

To avoid the severe IIA troubles due to aggregate logit, we avoid using pooled aggregate logit analysis. Rather, we use individual-level estimation, which is the favored approach for reducing IIA troubles, providing a continuous representation of market heterogeneity and improving the accuracy of market simulations. We use maximum likelihood estimation via individual-level logit, subject to *monotonicity constraints*, with Bayesian smoothing toward population parameters via *empirical Bayes*.

Monotonicity constraints are imposed based on the *a priori* ratings of levels within ordered attributes plus any respondent ratings of unordered attributes. In each iteration of the individual-level logit gradient search procedure, if two part-worths within an attribute are seen to violate the known preference order, we average the part-worths. Then, we add .01 to the more preferred of the two levels and subtract .01 from the less preferred level. In each iteration, we use the constrained solution to evaluate the likelihood and take it as the starting vector of utilities for the next iteration. Idiosyncratic monotonicity constraints for non-ordered attributes can improve the recovery of true heterogeneity. Both kinds of utility constraints (for ordered and non-ordered attributes) make utility estimation more robust in the face of limited sample availability as well as prohibitions (if these have been specified by the author) and usually improve the predictive validity of the part-worths.

Various methods of purely individual-level utility estimation have been advocated by Jordan Louviere[6], one of the leading researchers in discrete choice methods (Islam , Louviere, and Pihlens 2009) and under the proper conditions (enough information available at the respondent level) have been shown to work nearly as well as HB in recent applications (Marshall *et al.* 2010, Johnson and Orme 2012). Using maximum likelihood estimation at the individual level while shrinking individual estimates somewhat toward the population means via *empirical Bayes* has also been found to perform nearly as well as full HB (Frischknecht *et al.* 2013b). Our implementation of empirical Bayes involves first computing an aggregate logit solution across the respondent population using all choice data. We then temper each respondent's answer to each CBC choice task by average population preferences (see Appendix A). This shrinks the parameters toward the population parameters, avoiding *separation* problems (Frischknecht *et al.* 2013a) and aiding robust convergence. Shrinking the parameters toward the population parameters avoids assuming too much certainty (too large of scale factor) at the individual level in the face of usually sparse data conditions. It also leverages information across the sample to potentially improve the quality of individual-level estimates (nearly as effectively as HB).

Why Not HB?

One naturally would wonder why we don't employ HB (hierarchical Bayesian) estimation of utilities within the base capabilities of this software. After all, our CBC/HB routine is considered a gold standard and consistently produces excellent results. We considered using HB, but decided with this first version of the software to use individual-level logit made more robust by a combination of monotonicity constraints and empirical Bayes. If you insist on HB estimation, you may export the choice data from this software to a file compatible with our CBC/HB system for HB estimation (under the *Advanced* menu option in Discover-CBC).

The main reasons against using HB for this particular software application were:

1. HB places a much larger computational requirement on the servers than our method of empirical Bayes. If hundreds of users submitted HB jobs simultaneously, the entire system could potentially grind to a crawl.
2. Purely individual-level estimation via logit (when leveraging within-attribute ranking information of levels via monotonicity constraints and employing empirical Bayes) can be nearly as accurate as HB models (see evidence of this in Appendix A).

Market Simulations

Users analyze Discover-CBC data using the same Online Market Simulator as provided to our Lighthouse Studio CBC clients. This is a streamlined, easy-to-use, and attractive web-based (SaaS) application. For tabulating average utilities, utilities are normalized to zero-centered diffs scaling. Zero-centered diffs normalizes each person's utilities to have equivalent range (average range of 100 utiles across

[6] We should note that our method of individual-level logit plus augmented choice data via empirical Bayes is not exactly the same as Louviere's (and colleagues') approaches, but shares many essential characteristics.

attributes), so that it is more proper to combine and compare respondents and respondent groups. We still use the raw utilities to estimate shares of preference, under the Randomized First Choice Rule.

References

Frischknecht, Bart, Christine Eckert, John Geweke, and Jordan J. Louviere (2013a), "Incorporating Prior Information to Overcome Complete Separation Problems in Discrete Choice Model Estimation," working paper downloaded from www.censoc.uts.edu.au/pdfs/geweke_papers/gp_working_10.pdf.

Frischknecht Bart, Christine Eckert, John Geweke, Jordan Louviere (2013b), "Estimating Individual-level Choice Models," 2013 ART Forum, 9-12 June 2013, Chicago.

Islam, Towhidul, Jordan Louviere, and David Pihlens (2009), "Aggregate Choice and Individual Models: A Comparison of Top-Down and Bottom-Up Approaches," Sawtooth Software Conference Proceedings, pp 291-298.

Johnson, Richard and Bryan Orme (2012), "Can We Shorten CBC Interviews without Sacrificing Quality?" SKIM/Sawtooth Software European Conference, Berlin, Germany.

Lattery, Kevin (2009), "Coupling Stated Preferences with Conjoint Tasks to Better Estimate Individual-Level Utilities," Sawtooth Software Conference Proceedings, pp 171-184.

Marshall, Don , Siu-Shing Chan, and Joseph Curry, "A Head-to-Head Comparison of the Traditional (Top-Down) Approach to Choice Modeling with a Proposed Bottom-Up Approach," Sawtooth Software Conference Proceedings, pp 309-320.

Sawtooth Software (2013), "CBC Technical Paper," Technical Paper Series, downloaded from www.sawtoothsoftware.com.

Appendix A

Empirical Bayes Data Augmentation

We augment each respondent's choice data by assuming normative responses based on the population parameters. To wit, we add new choice tasks to each respondent's data, making it appear as if a respondent with average population preferences has also answered each individual respondent's tasks. A weighting factor, λ, is selected that governs the contribution of the population preferences to individual data based on the researcher's experience regarding how much shrinkage should be done toward the population means. We have tried various weighting factors with two high-quality CBC datasets and have selected a default weighting factor that is conservative and works very well. Earlier in this document we described reasons for using this rapid procedure for parameter estimation. Empirical Bayes is not as complete as a full HB estimation, but the researcher may export the data for estimation under our locally-installed CBC/HB software if full HB estimation is desired.

Although it would seem natural to replicate each respondent's choice tasks and use the normative responses for those replicated tasks, it is actually more compact and equally appropriate to format the choice tasks once within the data and to leverage the population information by assuming continuous choice probability responses (chip allocation). An illustration of such data augmentation for a single choice task with three alternatives A, B, and C is shown below, for a λ of 0.25. The individual respondent in this illustration has chosen alternative A. Using the logit rule, we may compute the likelihood of the population picking each alternative for that same choice task (shown below as 42%, 38%, 20%).

	Individual's	Population	Augmented		Normalized to 100%	
	Choice	Likelihood	Respondent Choice		Augmented Choice	
A	100%	42%	(100)+ 0.25(42)	= 110.5	110.5/125	= 88.4%
B	0%	38%	(0)+ 0.25(38)	= 9.5	9.5/125	= 7.6%
C	0%	20%	(0)+0.25(20)	= 5.0	5.0/125	= 4.0%
			Sum	= 125.0	Sum	= 100.0%

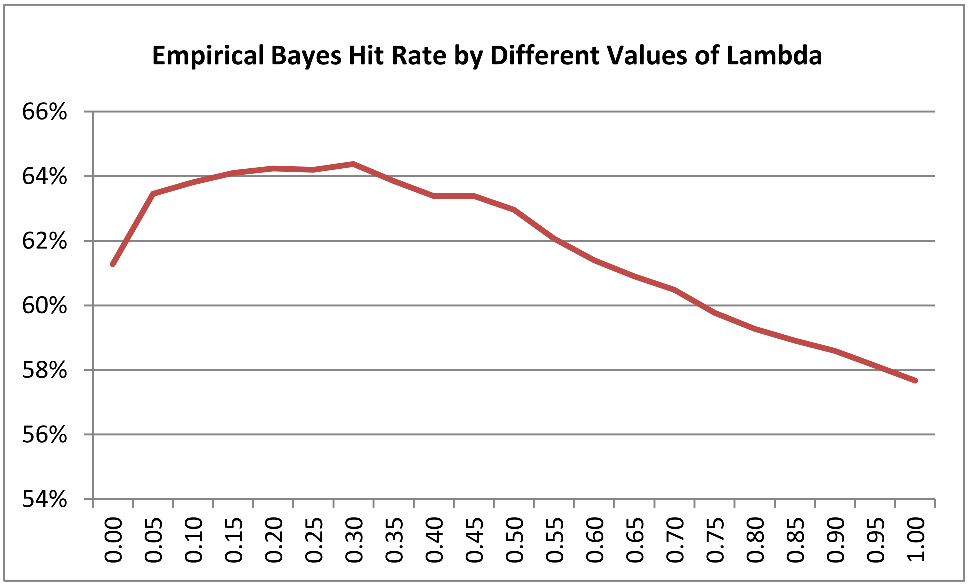
After augmenting the individual choice with population information, the new responses for this choice task reflect a chip allocation of 88.4 for alternative A, 7.6 for B, and 4.0 for C. We use the augmented data to estimate parameters for each respondent using maximum likelihood logit. If any parameter grows to an absolute magnitude of 5 during maximum likelihood estimation, we terminate the iterations early. This, for the extreme cases, shrinks the parameters toward zero (essentially a naïve empirical Bayes prior).

It may seem like a λ of 0.25 would imply that one-quarter of the information is due to the population priors and three-quarters weight to the individual. But, the population weight is actually less, since 100 of the 125 points in the example task above are attributed to the individual-level responses (80%).   Plus, one must remember that additional purely individual-level information is incorporated via the monotonicity constraints. Thus, the influence of population priors on the individual's choices across all tasks is less than 20% when using a λ of 0.25.

Empirical Bayes Results for Two Example Data Sets

In 1997, we collected over 400 CBC interviews (via mall intercept) involving mid-sized TV sets. We've used this data set over the years in multiple R&D applications and have found it to have excellent qualities. The survey contains 18 CBC tasks, each with 5 concepts (without a None alternative). The survey also contains 8 CBC holdout tasks. We cleaned the data to delete respondents who were speeding or answering very inconsistently, leaving 352 final respondents. The design had six attributes, with a 3x3x3x2x2x4 level structure (11 parameters to estimate, after effects coding).

If using CBC/HB software with default settings (and no utility constraints), the hit rate is 64.81%. Using empirical Bayes, we estimated utilities using different λ values from 0.00 to 1.00 in increments of 0.05. A λ of 0.00 implies purely individual-level logit (no Bayesian shrinkage) and a λ of 1.0 implies equal weight for individual and population data. The results are shown in the chart below:

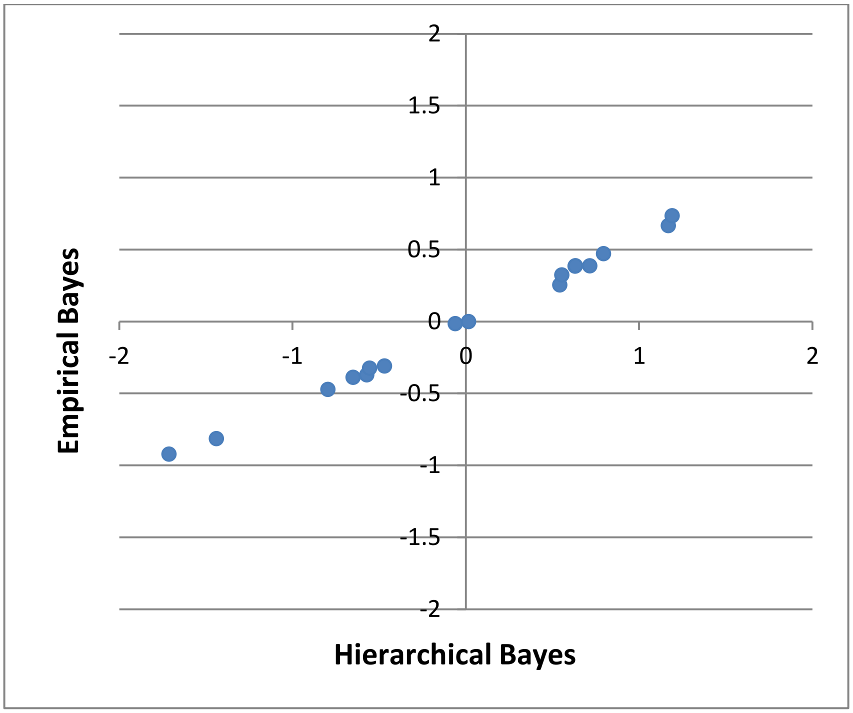


The hit rate is maximized (64.48%) at λ = 0.30, just barely below HB's hit rate of 64.81%. Estimation for empirical Bayes took about 3 seconds in total, compared to about 3 minutes for HB.

We further can compare the HB and empirical Bayes utilities for this data set. Below, we plot the average utilities (HB vs. empirical Bayes) across the sample:

HB vs. Empirical Bayes Utilities (Average across Sample)

(Correlation = 0.998)



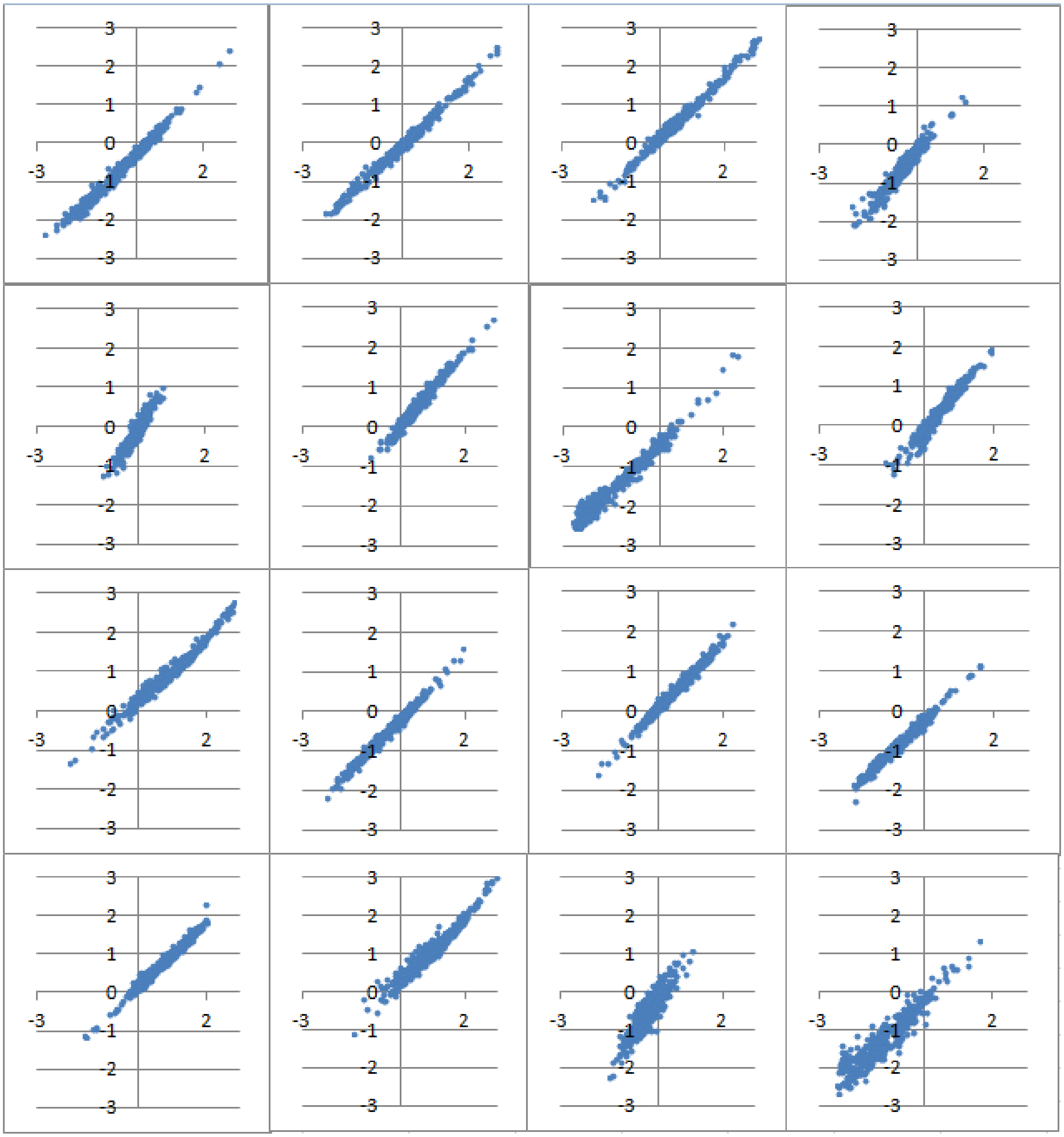
The average utilities are essentially identical, except for a uniform stretching of the scale for HB. HB utilities have larger scale than empirical Bayes by a factor of almost double (1.74).

We also can display individual-level scatter plots for each attribute level, comparing HB utilities to empirical Bayes utilities (we've plotted sixteen of the seventeen part-worths in the following chart; the seventeenth is omitted for symmetry purposes in the display only). Each scatter plot below involves 352 data points, representing the 352 respondents in this data set[7].

[7] We normalized the vectors of utilities (HB and empirical Bayes) within each respondent to each have a variance of 1.0 prior to plotting these data.

HB vs. Empirical Bayes Utilities

(Individual Respondent Scatter Plots)



In general, there is a remarkably high correlation of individual-level part-worths between empirical Bayes and HB estimation. The heterogeneity captured across respondents (patterns of individual preferences) is quite similar. We should note that this data set has a very high ratio of individual-level data relative to parameters to estimate. Most commercial datasets today are not so adequately blessed.

In September 2013, we collected a new dataset (supplied by SSI panel) to compare the results of Discover-CBC to our standard CBC software (Lighthouse Studio). This study was more demanding (statistically) than the data set we previously described. It included six attributes, with a 6x2x4x2x4x5 level structure plus a None alternative (18 parameters to estimate, after effects coding). A little over 300 respondents completed Discover-CBC software questionnaire, made up of 14 choice sets (the advisor built into the software recommended we ask between 10-16 choice tasks), each comprising 4 product concepts plus a None alternative. 20% of respondents' choices were of the None concept. We cleaned the data for speeders and for any respondents who answered more than 10 None choices, leaving 286 final respondents for analysis. The subject matter was vacation package options for summer travel. We included 8 holdout choice tasks, each with four product concepts (and no None concept). Prior to completing the 14 choice questions, we also asked respondents to rate the levels of 4 of the non-ordered 6 attributes (using the 4-point scale as previously described). The other 2 attributes had assumed *a priori* rank order.

We used both HB and empirical Bayes to analyze the data. For respondents completing the Discover-CBC version of the questionnaire, the hit rate for HB was 57.8% (only using the 14 CBC tasks, and global constraints for ordered attributes 4 and 6, as would be standard practice for CBC questionnaires in the industry). The hit rate for empirical Bayes (using the 14 tasks and also leveraging the ratings of levels within attributes as individual-level utility constraints) was maximized at 59.6 under λ = 0.45. Without the individual-level utility constraints, empirical Bayes achieved a hit rate 56.6%, slightly below that of HB (57.8%). We conclude that the individual-level ratings information is indeed helpful for this data set for improving hit rates. Empirical Bayes + individual-level utility constraints slightly edges out standard HB estimation, though the difference was not statistically significant. We also examined the accuracy of share of preference simulations for fitting holdout shares of choice, finding that empirical Bayes + individual-level utility constraints slightly edges out HB estimation (using global constraints for attribute 4 and 6). The MAE (Mean Absolute Error) of prediction for holdout choice shares was 2.12 for Empirical Bayes and 2.21 for HB.

What made this project even more interesting is that a randomly selected, additional group of 319 respondents (after cleaning) simultaneously received a standard CBC version of the same study as implemented using our Lighthouse Studio software. The 14 CBC tasks were preceded by ratings questions for attribute levels, just as we did with for the respondents who received the Discover-CBC questionnaire version (to give these respondents the same training task as the other respondents).

2013 Summer Travel Data Results

	Hit Rate	**Share Prediction Error (MAE)**
Lighthouse version of CBC (HB with global constraints on just attributes 4 & 6) n=319	57.8%	2.82
Discover-CBC (empirical Bayes with full individual-level utility constraints) n=286	59.6%	2.12

For this group of 319 respondents who received a standard CBC questionnaire, we computed part-worth utilities using the standard approach with HB (with global constraints on attributes 4 and 6). The hit rate was 57.8%, which ties the hit rate for Discover-CBC when also analyzed via HB (without the benefit of the supplemental level ratings within attributes questions). This suggests that the new Discover-CBC software's strategy for experimental design does at least as well as the Lighthouse Studio version of CBC. For this data set, including ratings of levels within attributes as individual-level constraints further improves the predictive validity. Of course, it can be argued that the time spent in completing those level ratings questions (1 minute, 7 seconds median time for this study) could have been spent completing a few more CBC questions. Yet, those ratings of levels within attributes provide a good training task to introduce the attribute levels, make the survey more varied (less monotonous) for respondents, and when leveraged as individual-level utility constraints add useful information for part-worth utility estimation.

In summary, Empirical Bayes is a shortcut procedure, but it does nearly as well as full HB for these two data sets. Based on our results, where λ of 0.30 maximized the predictive validity for the first project (without individual-level utility constraints) and λ of 0.45 maximized the predictive validity of the second (with individual-level utility constraints), we have decided to set the default λ to 0.35 for Discover-CBC software[8].

A topic for future research in empirical Bayes is to investigate a bootstrap sampling procedure that systematically holds out single tasks in the design and estimates the utilities using the remaining tasks under different values of lambda. In this way, we could calibrate the appropriate value for lambda for any given dataset. A second topic for future research is to use an initial latent class solution rather than a single-group aggregate solution for the shared information component across respondents. We should also note that further advances in hardware technology (including load balancing) may eventually lead us to include the full HB utility estimation routines within the Discover-CBC software.

[8] In July 2014, we conducted yet another spit-sample test to compare CBC (with HB estimation) vs. Discover-CBC with empirical Bayes estimation (using λ of 0.35). The subject matter was smart phone choices on 7 attributes. Sample was provided by SSI. Separate groups received standard CBC, Discover-CBC, and a holdout sample completed 12 CBC tasks (single block plan). Average utilities for CBC/HB and Discover-CBC (empirical Bayes) were correlated at 0.92. Holdout predictability was essentially tied for standard CBC and Discover-CBC.

Appendix B

Dominated Concept Definition

The experimental designer in Discover-CBC seeks to avoid dominated concepts for each individual. A dominated concept is one that is logically inferior, in terms of preference, to another one within the same task. However, avoiding dominated concepts is one of multiple goals, so it is possible in some circumstances for it not to be able to avoid all dominated concepts.

Below are a few examples of dominated (and not-dominated) concepts. All concepts within the task are compared, though for simplicity we will just illustrate the cases using two concepts.

Let's assume the following ratings and *a priori* rank orders are assigned to the following attribute levels:

<u>Attribute 1: Brand</u>

- Brand A (respondent rated as "1" in preference)
- Brand B (respondent rated as a "2" in preference)
- Brand C (respondent rated as a "2" in preference)
- Brand D (respondent rated as "No Opinion")

<u>Attribute 2: Speed</u>

- 20 pages per minute (*a priori* assigned a "1" in preference)
- 25 pages per minute (*a priori* assigned a "2" in preference)
- 30 pages per minute (*a priori* assigned a "3" in preference)
- 35 pages per minute (*a priori* assigned a "4" in preference)

<u>Attribute 3: Price</u>

- $100 (*a priori* assigned a "4" in preference)
- $120 (*a priori* assigned a "3" in preference)
- $140 (*a priori* assigned a "2" in preference)
- $160 (*a priori* assigned a "1" in preference)

Dominated Concept Example #1:

Brand A (1)	Brand B (2)
25 pages per minute (2)	30 pages per minute (3)
$140 (2)	$120 (3)

(Every attribute for the right-hand concept has higher preference than the left concept, leading to a clearly dominated concept.)

Dominated Concept Example #2:

Brand A (1)	Brand A (1)
25 pages per minute (2)	30 pages per minute (3)
$140 (2)	$120 (3)

(The concepts share the same attribute level for attribute #1. Every other attribute for the right-hand concept has higher preference than the left concept, leading to a clearly dominated concept.)

<u>Not</u> Dominated Concept Example #1:

Brand C (2)	Brand A (1)
25 pages per minute (2)	30 pages per minute (3)
$140 (2)	$120 (3)

(The right-hand concept is better on two attributes but worse on one attribute. We don't know the importance of attributes for this respondent, so we cannot determine that one concept dominates the other.)

<u>Not</u> Dominated Concept Example #2:

Brand B (2)	Brand C (2)
25 pages per minute (2)	30 pages per minute (3)
$140 (2)	$120 (3)

(We only use a 3-point scale, providing rough discrimination on preference. Thus, we don't believe we have enough information to know for certain that the right-hand concept clearly dominates the left.)

<u>Not</u> Dominated Concept Example #3:

Brand D (No opinion)	Brand C (2)
25 pages per minute (2)	30 pages per minute (3)
$140 (2)	$120 (3)

(Whenever an attribute level is rated as "no opinion", then we don't have any information to determine for certain that one concept is preferred to the other. The exception to this is if two concepts share the same "no opinion" level, but all other aspects lead to a clear domination.)