Replication notes for: The Probit Choice Model under Sequential Search with an Application to Online Retailing

Jun B. Kim   Paulo Albuquerque   Bart J. Bronnenberg

May 5, 2016

DISCLAIMER

Following the guidelines set by Management Science INFORMS, we provide these data files and programs "as-is" without any support. In particular, by supplying these replication files, we do not grant permission to use the data for other purposes than replication. We are under no obligation to answer questions about the code or provide further support for the code or implementation.

1 Overview

The estimation of our model involves two key steps. First, given a set of candidate parameters, we simulate individual-level optimal search and choice decisions for each consumer. Second, we aggregate search sets and choices across consumers and compute the aggregate-level predictions. During the optimization process, we maximize the likelihood of the model for the combined data set of view ranks, conditional choice shares, and sales ranks, subject to matching the fraction of consumers who browse but do not purchase.

The major computational challenge during the estimation process is that our likelihood function may be non-smooth due to the use of the selection rule when simulating individual-level optimal search and choice. Note that sorting of products in descending reservation utility values is a discrete operation and two different parameter sets can lead to two different sorting orders when we simulate an individual consumer. For instance, a parameter vector of β may lead to the optimal search sequence of $< A,B,C >$ while $\beta + \Delta\beta$ to $< B,A,C >$. Such a reversal in optimal search sequence may introduce discontinuities on search and choice probabilities since their computations are conditional on the search sequence. This means that the surface of our objective function may have jumps. In the next section, we provide an estimation strategy to deal with this.

2 Estimation

2.1 Overview

In this section, we provide a general approach for estimating the proposed model using aggregate search and choice data. We first extensively explored the parameter space in order to locate the promising parameter subspace using a stochastic search algorithm. Next, within this subspace, we used a gradient-based algorithm. We provide the steps below.

2.2 Step 1: Explore

The goal in the first step is to find the region of parameter space that may contain the global optimum point. In order to fully explore the parameter space, we recommend using a stochastic search algorithm. We used the differential evolution (DE) algorithm.[1] At the end of the exploration stage, this returns the best parameter set from the parameter vector population in DE.

2.3 Step 2: Optimize

Using the best parameter estimates from the step 1 as the initial starting point, we estimated the empirical model using the Matlab optimization toolbox.

3 Replication Instruction

Steps for replication:

1. Unzip the file "`matlab data and code`" into a folder. This folder contains both matlab data and program files. The matlab program files are annotated. For data structure, please refer to comments in matlab file `main.m`.

2. In order to run the program, one needs to install the parallel computing toolkit in matlab (www.mathworks.com/product computing/)

3. Start the file `main.m` to execute the replication process.

4. Note that due to the stochastic nature of the Differential Evolution Algorithm in the first step in section 3.1, in combination with your version of Matlab, and your machine specifications, there is a chance that your estimates show small differences from the parameter estimates we report in the paper.[2]

5. Our program was tested on Matlab R2015b (8.6.0.267246) released on Aug 20, 2015.

[1] For details, please refer to http://www1.icsi.berkeley.edu/~storn/code.html

[2] In addition, the numerical round-off issues in multithreaded or multicore computational environment such as matlab parallel processing toolkit may lead to small differences in parameter estimates. (e.g., see http://blogs.mathworks.com/loren/2009/12/04/comparing-single-threaded-vs-multithreaded-floating-point-calculations).