Comparison of Nonlinear Optimization Methods on a Multinomial Logit-Model in R

Hannes PUCHER
Institute for Information Business, Vienna University of Economics and BA
Vienna, 1090, Austira

and

Volker STIX (corresponding author)
Institute for Information Business, Vienna University of Economics and BA
Augasse 2-6
Vienna, 1090, Austria
volker.stix@wu-wien.ac.at

ABSTRACT

In this article we analyze several optimization methods of a real-life nonlinear optimization problem in R. The optimization problem results from fitting a multinomial logit-model. This is part of the so called Brand Simulator, a tool to forecast and simulate possible marketing mixes. In the existing optimization procedure, simple derivation-free techniques are used. The objective of this paper is to apply several promising optimization techniques in order to improve optimization with respect to time and quality. Hence a testbed was implemented in R. Results show that the proposed techniques outperform existing optimization procedures in most of the cases.

Keywords: marketing-mix analysis, Multinomial logit-model, logistic regression, optimization, quasi-Newton methods, maximum-likelihood estimator.

1. INTRODUCTION

The "Brand Simulator" (BS) is a software tool for optimization and simulation. The major purpose of the BS is the simulation and analysis of the impact of different marketing-mix concepts on the behavior of households in Germany in the fast-moving-consumer-goods market.
Whereas the BS consists of several optimization steps and modules, we focus in this paper on the optimization of a sub-model denoted as brand model (BM). The BM is based on a multinomial logit-model (MNL). In the BS the involved parameters are estimated by linear-least square estimation through data drawn from a consumer panel. Least-square estimation of a MNL results in a nonlinear optimization problem, which must be solved by nonlinear optimization techniques. In addition, several parameters of the model are subject to linear constraints in order to meet marketing-specific needs. In the BS the optimization is done using derivation-free techniques such as Nelder-Mead technique and Golden Section Search [1] so far. Constraints are incorporated by a linear penalty function.
In order to improve existing optimization procedures, the following alternative methods were applied to the problem: derivation-based quasi-Newton methods, a genetic algorithm, simulated annealing, and a combination of a quasi-Newton method with the genetic algorithm. In doing so, parameters are simultaneously optimized instead of separate optimization. Moreover, alternative ways of incorporating the linear constraints such as quadratic penalty function, barrier function and box-constraints are compared. Additionally, maximum-likelihood estimation is applied.
To verify the introduced experiments, the optimization problem was implemented in R. In view of the numerous experiments to be solved, the problem was embedded in a testbed. Whereas the testbed was implemented in pure R code, the core modules of the objective function computation are mainly implemented in C subroutines. The quality of optimization and simulation is evaluated by several goodness-of-fit measures.
In the remainder of this article we will present the optimization problem in short. We introduce the special features of the BM, input data, optimization procedure, and how optimization quality is measured. In Section 3, we briefly describe the methods used for optimization. Section 4 presents experimental settings and implementation of the testbed and in Section 5 we discuss the results derived from experiments.

2. OPTIMIZATION PROBLEM

Features of the brand model
The BM represents a purchase situation in which an individual faces a set of alternatives. The specification of the BM is based on an MNL. The MNL belongs to the discrete choice models in which the set of two or more alternatives is discrete and finite [1]. In the model, the explained variable is the set of brands and the explanatory variables are marketing-mix variables such as price or promotion. The parameters of the model describe the impact of the explanatory variables on the explained variable. From a technical point of view, the BM estimates the purchase probability of each brand i within a set C of alternative j subject to explanatory variables X and its parameters θ. The MNL guarantees that probabilities for each brand within a set of alternatives add up to 1. The purchase probability of a brand is given by

$$P_i = \frac{\exp^{\theta_i X_i}}{\sum_{j \in C} \exp^{\theta_j X_j}} \quad . \tag{1}$$

Compared to a traditional MNL, the BM differs in the structure of alternatives and the parameters. Due to panel properties, a set of alternatives in a certain purchase situation cannot be observed directly. Therefore, alternatives are identified and calculated on the basis of information

extracted from employed data [15]. The sets of alternatives in a certain purchase situation may vary in size, explanatory variables and, consequently, in the number of parameters. In the BM, the various sets of alternatives are denoted as contentions. The number of parameters for estimation depends mainly on the number of brands involved in the analyzed data set C.

Input data

Data for this study was provided by the GfK Group drawn from a consumer panel maintained by the company. The panel records purchase acts of households on a weekly basis. A purchase act contains information such as time, place, subject, or quantity of purchase. Received data was reassessed and edited by means of statistical principles. From this data, sets are extracted related to specific commodity groups containing different brands. For this work, a large and a small data set are used. The large data set includes 445,896 purchase data records, whereas the small data set includes 31,041 records.

Original optimization procedure

Using the least-square method for optimizing the parameters of the BM, the problem is given as

$$\min_{\theta} \sum_{i \in C} (y_i - P_i)^2 \quad \text{subject to} \quad c_l(\theta) \geqslant 0 \,, \tag{2}$$

where $c_l(\theta)$ denotes l linear constraints restricting the parameters, y_i refers to the observed purchase acts and can only take the values 0 or 1, whereas P_i takes values between 0 and 1. Parameters are optimized separately. Hence optimization is organized in five runs [15]. Each run optimizes a different set of parameters. For multidimensional optimization, a Nelder-Mead technique, also called Downhill-Simplex (DS), is adopted and a Golden-Section Search (GS) is applied to the one-dimensional case [1]. Separate optimization is used because the Nelder-Mead technique cannot manage the optimization of simultaneous variables. The optimization is terminated by stopping criteria such as a maximum number of iterations or a drop below a minimum change in the objective value. Linear constraints of the problem are incorporated by a linear penalty function.

Optimization quality

Optimization quality is measured by a goodness-of-fit measure developed by Estrella [3]. Its purpose is to provide a goodness-of-fit for discrete choice models comparable to the R^2 calculated in traditional regression analysis. Table 1 shows the performance of the existing optimization procedure subject to employed data sets. Due to the small data, optimization took about 4 hours resulting in an Estrella's R^2 of 0.034. Considering the small data, optimization took 9 minutes resulting in an Estrella's R^2 of 0.062. In both cases optimization quality is relatively poor.

Table 1. Optimization of BM in the Brand Simulator

Characteristics	Small data set	Large data set
Number of parameters	71	165
Number of constraints	58	134
Computation time in minutes	9	234
Starting objective value	7,770.894	142,098.827
Objective value after optimization	5,828.917	117,983.932
Estrella's R^2	0.062	0.034

3. PROPOSED METHODS

In order to improve optimization procedure, several derivation-based methods and meta-heuristics are implemented. This includes 5 quasi-Newton methods, a version of a genetic algorithm, and a combination of the genetic algorithm with a quasi-Newton method. In addition to that, a Nelder-Mead technique [10] comparable to that of the Brand Simulator is applied as a benchmark. In our experiments, all parameters are optimized simultaneously.

Quasi-Newton methods

The set of quasi-Newton methods proposed for the optimization problem includes three types of conjugate gradient techniques (CG) [9], three implementations of the Broyden-Fletcher-Goldfarb-Shannon procedure (BFGS) [4,6,9], a limited BFGS with box-constraints (L-BFGS-B) [12], the Levenberg-Marquardt (LEV-MAR) [8] and the Brandon-Hall-Hall-Hausmann method (BHHH) [5]. Among the CG, a Fletcher-Reeves version (CG1), a Polak-Ribière version (CG2) and a Hestenes-Stiefel version (CG3) was applied. All techniques are available in the standard distribution of R or in additional packages [14]. All proposed methods use

$$\theta_{k+1} = \theta_k + \alpha_k p_k \,, \tag{3}$$

where subscript k denotes the k-th iteration, p_k defines the search direction and α_k defines the step length. The proposed methods differ in varied computations of the search direction and step length.

First-order derivations (gradient) and the various approximation of second-order derivations are calculated by finite differentiation [13]. The calculation of derivations lead to further computation time at each iteration, the number of iterations, however, for achieving an acceptable optimum is expected to decrease.

Genetic algorithm

Instead of calculate derivations, a genetic algorithm (GA) uses special search operators to compute new trial solutions. The GA we propose for this problem is based on the GENOUD program [7]. It is highly scalable and incorporates nine search operators. Those include four crossover operators, four mutation operators and one operator for cloning. Additionally, the GA implements an elite strategy to improve convergence property [11]. Moreover, the GA offers a combination with quasi-Newton methods. In this study, the GA was combined with the BFGS version of the PORT routines which enables box-constraints (GA-PORT). This feature exists mainly for improving local optimization performance as GA are known to perform inefficiently near an optimum [11].

Simulated annealing

We implemented simulated annealing (SA) for this problem as described in [2]. There the SA method uses a logarithm temperature schedule dependent on the current iteration, a starting temperature and the maximum number of function evaluations. Starting temperature and maximum number of evaluations can be set by the user.

Maximum-likelihood estimation

In addition to the least-square method for optimization, the maximum-likelihood estimation (MLE) is used. In general, MLE is considered to be a standard approach for estimating a logit-model [15]. The MLE objective function is given by

$$\max_{\theta} \sum_{i \in M} \ln(P_i) \quad \text{subject to} \quad c_l(\theta) \geq 0 \,, \tag{4}$$

where the set M covers all observations where $y_i = 1$.

Constraints handling procedures

In addition to the linear penalty function, a quadratic penalty function and a logarithm barrier function are evaluated. Compared to the linear penalty function, an individual penalization is pursued as each constraint violation is penalized individually. Penalty or barrier methods augment the objective function and hence transform a constrained problem to an unconstrained problem. Penalty procedures count among dual methods, whereas barrier functions are related to inter-point procedures. Considering the penalty as well as the barrier procedure, outer iterations are necessary with respect to a declining penalty parameter. Theoretically, a decline in the penalty parameter forces the optimizers to become feasible [13]. There is, however, no guarantee that a final solution achieves feasibility. For comparison, the simpler box-constraints feature available in the GA, BFGS and L-BFGS-B are used as well. In contrast to penalty and barrier procedures, box-constraints are integrated in the method's algorithm and are counted among primal methods [13]. The adjusted algorithm guarantees feasibility during the whole optimization and no outer iterations are required.

4. EXPERIMENTAL SETTINGS AND IMPLEMENTATION

The challenge of applying different techniques to a problem is to make the performance of the techniques comparable. In doing so, equivalent controlling parameters among the techniques were identified and synchronized to enable similar conditions. This includes stopping criteria with respect to maximum iterations, to relative changes in objective or gradient. Different types of techniques, however, are accompanied by different controlling parameters. As a consequence, those parameters were adjusted in such a way that the optimization effort was comparable as exactly as possible. Optimization time was measured in terms of number of objective function evaluations. These values were derived from several test runs.

As a result, the maximum number of iterations was set to 10 for all methods except for the LEVMAR, Nelder-Mead, SA and GA. LEVMAR only allows the adjustment of a maximum number of function evaluations per iteration. This parameter was set to 1,000 in the event of outer iterations and 1,700 without outer iterations. Similarly, maximum iterations of Nelder-Mead and SA were limited to 10,400 in the event of outer iterations and otherwise to 1,300. Implementation-based parameters were set to default. Hence it is assumed that in general all techniques perform best with its default values. One Exception was found in the LEVMAR algorithm. Here, test runs suggested to increase the accuracy of finite differentiation to 1e-5 instead of 1e-3 (default).

For the implementation of the constraints, the type of penalization of the quadratic penalty and logarithm barrier procedure had to be adjusted. We decided to realize a strict and a mild penalization. This was done by setting the starting penalty parameter μ_0 to 1 (strict) and 2 (mild). The decline σ in the penalty parameter at each outer iteration was fixed to 10% of the previous value. The maximum number of outer iterations was set to 8 in order to keep the number of total iterations low. For comparison, the linear penalty used in the original Brand Simulator was implemented as a benchmark. A summary of all configurations and their reference number is given in Table 2.

Configurations 1-1, 1-2, 2-1, 2-2 and 3 were used with CGs, BFGS, L-BFGS-B, LEVMAR, BHHH, as well as SA procedures and the box-constraints were used with BFGS, L-BFGS-B and the GA. For the latter, *only* the box constraints method was deployed. In general, the proposed penalty and barrier procedures are considered to be inefficient with GA, especially in the event of many constraints. Thus it is preferable to use primal methods [11].

Table 2. Configurations of constraints handling procedures

ID	Procedure	μ_0	σ	Maximum outer iterations
1-1	quadratic penalty	1	0.1	8
1-2	quadratic penalty	2	0.1	8
2-1	logarithm barrier	1	0.1	8
2-2	logarithm barrier	2	0.1	8
3	linear penalty	-	-	-
4	box-constraints	-	-	-

Both the least-square and maximum-likelihood procedures were applied to all settings. By its nature, the LEVMAR could solely be used with the least-square procedure. Similarly, the BHHH could only be used with the MLE. The study is conducted in three phases. First, all experiments are processed on basis of a small data set. Second, the most promising methods in combination with constrained handling procedures are applied to a large data set to verify performance. In summary, 234 experiments (or optimization runs) are conducted due to the small data set, 11 experiments on the basis of the large data set.

For implementation the software package R is chosen. This is because it is freely available, it is easy and flexible to implement the special features of the BM and the proposed optimization techniques are available. The testbed compounds several R functions. One of them is the main function in which experiments can be set centrally. Central settings prevent faulty input information.

Specified settings as well as automatically pre-calculated information are passed as input to selected optimization techniques. This ensures that input information is attuned to each interface to ensure reliability and validation of the results due to applied technique. After optimization, results are processed and recorded in a table including information about experiment settings, achieved least-square function value, computation time recorded in minutes, Estrella's R^2, the efficiency measure and feasibility of solution.

The major challenge was to reduce computation time of the objective function. Therefore, loops for allocating parameters for the corresponding data sets and the denominator of (1) were coded as C subroutines. In addition, information about contentions structure, calculated prior, was used to diminish the loop size and matrix operations were abandoned in favor of vector operations, which are highly efficient in R.

5. RESULTS

A first exploration of the results revealed that the barrier function did not work well with finite differentiation. Therefore, those results were excluded for further analysis. So the number of results reduced from 234 to 158. To focus on the best results, feasible results exceeding the average are

selected. Table 4 shows these results from small data set in descending order with respect to Estrella's R^2, where "o.f." stands for "objective function".

For comparison, we included the optimization result of the original Brand Simulator (#29). The results indicate that optimization in the original Brand Simulator was clearly outperformed in quality and performance. The DS integrated in R was inferior to quasi-Newton methods and meta-heuristics. Among applied techniques, LEVMAR (#1 with 0.590260) and BHHH (#2 with 0.545580) performed best while LEVMAR was far more efficient (ratio of 10.97469 to 5.18091). This outcome is not surprising, as both techniques exploit the special structure of the objective function due to least-square estimation and maximum-likelihood estimation respectively. Furthermore, the combination of the GA and the BFGS-PORT proved to be efficient (#3 with 0.435517) and outperforms the respective methods GA (#16 with 0.378640) and the best BFGS-PORT (#9 with 0.396960). The BFGS-PORT, however, was faster (ratio of 11.15226 to 7.04040). Considering time, the BFGS-PORT in combination with NLS turned out to be the most efficient technique. The achieved optimization quality of applied SA configurations was below the average. One reason could be that penalty function combined with SA prevent the SA to achieve better optimization quality. Furthermore, the default values of the SA may not be adequate for the considered optimization problem.

Considering estimation methods, no clear evidence is given whether the least-square or maximum-likelihood estimation is more qualified. There is only a slight tendency that maximum-likelihood lead to better results in average.

Table 3 shows the robustness of the constraints handling configurations. As expected, the most robust configuration is the box-constraint configuration. No infeasible solution has been generated. Among the penalty functions, the linear penalty function turned out to be more robust. The linear penalty function, however, led to inferior optimization quality. A milder penalization should be preferred to a stricter penalization amongst quadratic penalty functions.

Table 3. Robustness of constraints handling procedures

Configuration	feasible optimizers	infeasible optimizers	Sum
1-1	17	21	38
1-2	23	15	38
3	31	7	38
4	44	0	44
Sum	**115**	**43**	**158**

We found similar results using the large data set (see Table 5). Again, LEVMAR (#1) outperformed all other techniques (including results from the original Brand Simulator). In addition, it turned out to be the most efficient technique (ratio of 0.16817). The results on BHHH are not available since BHHH was not able to manage the involved large vectors. L-BFGS-B turned out to be competitive in terms of optimization quality but suffers from relatively high computation time. Sticking to computation time, most of the tested techniques outperform the Brand Simulator with respect to computational time. LEVMAR e.g. is 1.5 times faster than the Brand Simulator. This is surprising, as implementation in the Brand Simulator bases completely on C sources and is therefore expected to be much faster, especially due to storage management. The performance of the Levenberg-Marquardt could be further enhanced by using analytical derivations. This would reduce multiple function evaluations at each iteration.

Considering the optimization program, the maximum-likelihood optimization led to better results with respect to the L-BFGS-B, BFGS-PORT and GA-PORT and confirmed the tendency of the small data set. Again, the combination of GA and BFGS-PORT performed well.

Table 4. Results from the small data set

#	o.f.	technique	constraints	least-square value	minutes	R^2	R^2/time
1	nls	levmar	1-2	5418.15738039334	5.37833	0.590260	10.97469
2	mle	bhhh	1-2	5571.69891647763	10.53050	0.545580	5.18091
3	mle	ga-port	4	5659.41900000000	6.18595	0.435517	7.04040
4	nls	levmar	1-1	5599.98244185570	4.88083	0.432380	8.85883
5	mle	lbfgsb	4	5670.46027991221	7.61433	0.431530	5.66733
6	mle	lbfgsb	1-2	5757.33995396882	11.89933	0.418290	3.51527
7	nls	lbfgsb	1-1	5745.67169594246	10.88433	0.415260	3.81523
8	mle	cg3	1-1	5808.06856585466	6.93417	0.410200	5.91568
9	mle	port	4	5687.64024771080	3.55950	0.396960	11.15226
10	mle	lbfgsb	1-1	5790.85404069227	11.50100	0.393850	3.42449
11	mle	ds	1-1	5790.27101869178	5.97900	0.390790	6.53611
12	nls	lbfgsb	4	5685.20097477194	6.38767	0.390600	6.11489
13	nls	port	4	5629.13152252270	3.22817	0.389690	12.07160
14	mle	cg3	1-2	5835.35858095708	7.05167	0.385660	5.46900
15	mle	bfgs	1-2	5792.71262403588	6.65750	0.378850	5.69052
16	mle	ga	4	6003.44310000000	6.12500	0.378640	6.18123
17	nls	cg1	1-2	5766.36808461207	5.72400	0.377810	6.60052
18	mle	cg2	1-2	5792.99466547177	6.72183	0.377400	5.61460
19	nls	cg2	1-1	5784.42395161786	5.93467	0.377160	6.35516
20	nls	port	1-2	5774.24620164565	3.13133	0.376920	12.03709
21	nls	cg1	1-1	5776.05192006425	5.84650	0.376610	6.44168
22	mle	cg2	1-1	5798.36050816819	6.78650	0.376470	5.54733
23	nls	cg2	1-2	5780.94833128411	5.88967	0.376130	6.38623
24	nls	cg3	1-2	5779.91307155361	5.89733	0.376000	6.37571
25	mle	cg1	1-1	5813.19163108785	6.58133	0.369300	5.61129
26	mle	ds	3	5875.18662272825	5.92333	0.360470	6.08566
27	nls	ds	1-1	5840.84537844609	5.05150	0.357580	7.07873
28	nls	ds	1-2	5831.23310004912	5.04183	0.350280	6.94744
29	NLS	DS+GS	3	5828.91720000000	9.00000	0.062410	0.69340

Table 5. Results from large data set

#	o.f.	technique	constraints	least-square value	minutes	R^2	R^2/time
1	nls	levmar	1-2	113800.015040000	159.67520	0.26850	0.16817
2	mle	lbfgsb	4	116922.048554040	234.13267	0.21294	0.09095
3	mle	port	4	117175.673136832	117.98750	0.19835	0.16811
4	mle	ga-port	4	117230.692295613	167.24683	0.18095	0.10820
5	nls	lbfgsb	4	118002.606827692	242.59233	0.06840	0.02820
-	NLS	DS+GS	3	117983.932000000	247.00000	0.03400	0.01380
6	nls	port	4	119648.266411146	108.15150	0.00168	0.00155
7	nls	cg1	1-2	124307.200000000	210.25000	n.a.	n.a.
8	mle	cg3	1-1	124554.3	247.176	n.a.	n.a.
9	mle	ga	4	127247.102757183	103.68117	n.a.	n.a.
10	mle	ds	1-1	129139.7	89.95	n.a.	n.a.
11	nls	ds	1-1	130141.7	81.6	n.a.	n.a.

6. CONCLUSION

Several quasi-Newton methods and meta-heuristics were applied to a nonlinear optimization problem subject to linear constraints. Linear and quadratic penalty functions as well as box-constraints were tested to incorporate the constraints. In addition, least-square and maximum-likelihood optimization models were compared. Results show that a Levenberg-Marquardt and a combination approach seem to be the most promising technique for this problem. The newly proposed techniques significantly outperform the existing optimization procedure in terms of optimization quality and computational time.

REFERENCES

[1] M. Avriel, **Nonlinear Programming: Analysis and Methods**, Dover Publishing, 2003.

[2] C. J. P. Bélisle, Convergence theorems for a class of simulated annealing algorithms on $\mathbb{R}^d$. **Journal of Applied Probability**, Vol. 29, No. 4, 1992, pp. 885-859.

[3] A. Estrella, A new measure of t for equations with dichotomous dependent variables. **Journal of Business and Economic Statistics**, Vol. 16, No. 2, 1998, pp. 198-205.

[4] D. M. Gay, Usage summary for selected optimization routines. **Computing Science Technical Report No. 153**, Murray Hill: AT&T Bell Laboratories, 1990.

[5] S. M. Goldfeld and R. E. Quandt, **Nonlinear Methods in Econometrics**, Amsterdam: North-Holland, 1972.

[6] J. E. Koonatz et al., A modular system of algorithms for unconstrained minimization. **ACM Trans. Math. Software**, Vol. 11, No. 4, 1985, pp. 419-440.

[7] W. R. Mebane and J. S. Sekhon: R version of Genetic Optimization Using Derivatives. **R package version 5.1-9**, 2007.

[8] J.J. Moré, The Levenberg-Marquardt algorithm: implementation and theory, **Lecture Notes in Mathematics,** 630: Numerical analysis (Watson, G. A. ed.), 1978, pp. 105-116.

[9] J. C. Nash, **Compact Numerical Methods for Computers: Linear Algebra and Function Minimization**. Bristol: Adam Hilger, 1990.

[10] J. A. Nelder and R. Mead, A simplex algorithm for function minimization. **Computer Journal**, Vol. 7, 1965, pp. 308-313.

[11] A. E. Eiben and J. E. Smith, **Introduction to Evolutionary Computing**. Berlin: Springer, 2003.

[12] J. Nocedal et al, A limited memory algorithm for bound constrained optimization. **SIAM J. Sci. Comput**., Vol. 16, No. 5, 1995, pp. 1190-1208.

[13] J. Nocedal and S. J. Wright, **Numerical Optimization**. New York: Springer, 1999.

[14] R-Project for statistical computing, URL: http://www.r-project.org/, 1.12.2008

[15] K. E. Train, **Discrete Choice Methods with Simulation**. Cambridge: Cambridge Univ. Press, 2003.