View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Repository of the University of Namur

UNIVERSITÉ DE NAMUR

Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche

researchportal.unamur.be

RESEARCH OUTPUTS / RÉSULTATS DE RECHERCHE

Application of an adaptive Monte Carlo algorithm to mixed logit estimation

Bastin, Fabian; Cirillo, Cinzia; Toint, Philippe

Published in:
Transportation Research Part B : Methodological

DOI:
10.1016/j.trb.2005.09.002

Publication date:
2006

Document Version
Early version, also known as pre-print

Link to publication

Citation for pulished version (HARVARD):
Bastin, F, Cirillo, C & Toint, P 2006, 'Application of an adaptive Monte Carlo algorithm to mixed logit estimation' Transportation Research Part B : Methodological, vol. 40, no. 7, pp. 577-593. https://doi.org/10.1016/j.trb.2005.09.002

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 21. May. 2019

Application of an adaptive Monte Carlo algorithm to Mixed Logit estimation

Fabian Bastin [a,*,1] , Cinzia Cirillo [b] , Philippe L. Toint [b]

[a] *Department of Mathematics, University of Namur, Rempart de la Vierge 8, Namur, Belgium.*

[b] *Transportation Research Group, Department of Mathematics, University of Namur, Rempart de la Vierge 8, Namur, Belgium*

Abstract

This paper presents the application of a new algorithm for maximizing the simulated likelihood functions appearing in the estimation of Mixed Multinomial logit (MMNL) models. The method uses Monte Carlo sampling to produce the approximate likelihood function and dynamically adapts the number of draws on the basis of statistical estimators of the simulation error and simulation bias. Its convergence from distant starting points is ensured by a trust-region technique, in which improvement is ensured by locally maximizing a quadratic model of the objective function. Simulated data is first used to assess the quality of the results obtained and the relative performance of several algorithmic variants. These variants involve, in particular, different techniques for approximating the model's Hessian and the substitution of the trust-region mechanism by a linesearch. The algorithm is also applied to a real case study arising in the context of a recent Belgian transportation model. The performance of the new Monte Carlo algorithm is shown to be competitive with that of existing tools using low discrepancy sequences.

Key words: Discrete choice; Mixed logit model; Trust-region methods; Adaptive Monte Carlo samplings.

* Corresponding author. Cerfacs, Av. G. Coriolis 42, 31057 Toulouse Cedex, France. Tel.: +33 5 6119 3014, fax: +33 5 6119 3000.

Email addresses: fabian.bastin@fundp.ac.be (Fabian Bastin), cinzia.cirillo@fundp.ac.be (Cinzia Cirillo), philippe.toint@fundp.ac.be (Philippe L. Toint).

[1] Research Fellow of the National Fund for Scientific Research (FNRS)

1 Introduction

Transport models based on discrete choice methods have dramatically changed in the last ten years (Bhat, 2003). Researchers have been proposing more flexible model formulations and practitioners are now starting to use them for a better understanding and representation of complex decisional processes in travel demand modeling. Among others, mixed logit models (MMNL) offer the possibility to overcome most of the limitations of multinomial and nested logit models (MNL and NL). They are mainly used to estimate randomly distributed explanatory variables, to measure correlation across alternatives and to account for state dependency across observations. Although the conceptual structure of MMNL was known from the late 70s (Electric Power Research Institute, 1977), the numerical cost associated with their evaluation delayed their applicability to real case studies until the mid nineties (Hensher and Greene, 2003). The non-closed mathematical form of MMNL indeed imposes the resolution of multidimensional integrals to calculate the relative choice probabilities. This is done by means of simulations, which can be in real large-scale model estimation very time consuming or sometimes infeasible even on fast machines. As a consequence, current research has turned to the cheaper quasi-Monte Carlo approaches, based on low discrepancy sequences, which has been shown to produce more accurate integration approximations than classical Monte Carlo samplings, when the number of draws is fixed, for instance in the study of physics problem (Morokoff and Caflish, 1995). Bhat (2001) and Train (1999) have proposed the use of Halton sequences (Halton, 1960) for mixed logit model and have found that they perform much better than pure random draws in simulation estimation. Garrido (2003) has explored the use of Sobol sequences, while Sándor and Train (2004) have compared randomized Halton draws and (t, m, s)-nets. This trend is not without drawbacks. For instance, Bhat (2001) pointed out that the coverage of the integration domain by Halton sequences rapidly deteriorates for high integration dimensions and consequently proposed a heuristic based on the use of scrambled Halton sequences. He also randomized these sequences in order to allow the computation of the simulation variance of the model parameters. Hess et al. (2005) have proposed the use of modified Latin hypercube sampling, whose performance has been assessed in two recent papers (see Bastin et al. (2005a) and Sivakumar et al. (2005)).

Quasi-Monte Carlo methods are not the only way to decrease the numerical cost involved in mixed logit estimation. In this paper we attempt to capitalize on the desirable aspects of pure Monte Carlo techniques while significantly improving their efficiency. Monte Carlo techniques benefit from a credible theory for the convergence of the calibration process, as well as of stronger statistical foundations (see for instance Fishman 1996 for a general review, Rubinstein and Shapiro (1993) and Shapiro 2000; 2003 for application to stochastic pro-

gramming, Bastin et al. (2005c) for more specific developments in the context of non-linear programming and mixed logit problems). In particular, statistical inference on the objective function is possible, while the quality of the results can only be estimated in practice, for quasi-Monte Carlo procedures, by repeating the calibration process on randomized samples (L'Ecuyer and Lemieux, 2002).

Our approach is to propose a new algorithm for stochastic programming using Monte Carlo methods, that is based on the trust-region technique. Trust-region methods are well-known in non-linear non-convex/non-concave optimization, and have been proved to be reliable and efficient for both constrained and unconstrained problems. Moreover, the associated theoretical corpus is extensive (Conn et al., 2000). Our efficiency objective led us to adapt the traditional deterministic trust-region algorithm to handle stochasticity and, more importantly, to allow an adaptive variation of the used number of draws at each iteration, as initially presented in Bastin et al. (2003). It additionally uses information on the bias and improved algorithmic safeguards, and has been proved to be convergent from any starting point in a companion paper (Bastin et al., 2005b). The technique results in an algorithm whose execution time is competitive with existing tools for mixed logit models, while giving more information to the practitioners, and achieving similar solution accuracy. We also aim to underline the importance of the optimization algorithm choice when looking for numerical performances and show that exploitation of statistical inference is valuable, by presenting applications of the method to simulated and real datasets.

The paper is organized as follows. We briefly review the mixed logit problem and some of its properties in Section 2. We then introduce our new algorithm in Section 3. Section 4 presents our numerical experimentations on simulated data. The discussion is then enlarged to a real case model estimation. Some conclusions and perspectives are finally outlined in Section 6.

2 The Mixed Logit model

2.1 The problem and its approximation

Discrete choice models provide a description of how individuals perform a selection among a finite set of alternatives. Let I be the population size and $\mathcal{A}(i)$ the set of available alternatives for individual i, $i = 1, \ldots, I$, with cardinal $|\mathcal{A}(i)|$. For each individual i, each alternative A_j ($j = 1, \ldots, |\mathcal{A}(i)|$) has an associated utility that depends on the individual characteristics and the relative attractiveness of the alternative. The utility is assumed to have the

additive form

$$U_{ij} = V_{ij} + \boldsymbol{\epsilon}_{ij}, \tag{1}$$

where $V_{ij} = V_{ij}(\beta_j, x_{ij})$ is a function of the model parameters vector β_j and of x_{ij}, the observed attributes of alternative A_j, while $\boldsymbol{\epsilon}_{ij}$ is a random term reflecting the unobserved part of the utility.

Assuming that individual i selects the alternative maximizing his/her utility, the probability that he/she chooses alternative A_j is given by

$$P_{ij} = P\left[\epsilon_{il} \leq \epsilon_{ij} + (V_{ij} - V_{il}), \forall A_l \in \mathcal{A}(i)\right]. \tag{2}$$

The particular form of this choice probability depends on the random terms $\boldsymbol{\epsilon}_{ij}$ in (1). If we assume that they are independently Gumbel distributed with mean zero and scale factor one, Equation (2) can be expressed with the logit formula

$$L_{ij}(\beta) = \frac{e^{V_{ij}(\beta)}}{\sum_{l=1}^{|\mathcal{A}(i)|} e^{V_{il}(\beta)}}, \tag{3}$$

where we have simplified our notation by dropping the explicit dependence on the known observations x_{ij}. Formula (3) characterizes the classical multinomial logit model.

Mixed logit models relax the assumption that the parameters β are the same for all individuals, by assuming instead that individual explanatory variables vectors $\beta(i)$ $(i = 1, \ldots, I)$ are realizations of a random vector $\boldsymbol{\beta}$. We then assume that $\boldsymbol{\beta}$ is itself derived from a random vector $\boldsymbol{\xi}$ and a parameter vector θ, which we write as $\boldsymbol{\beta} = \beta(\boldsymbol{\xi}, \theta)$, and therefore $L_{ij}(\boldsymbol{\beta})$ becomes $L_{ij}(\boldsymbol{\xi}, \theta)$. The probability choice is then given by

$$P_{ij}(\theta) = E_P\left[L_{ij}\left(\boldsymbol{\xi}, \theta\right)\right] = \int L_{ij}(\xi, \theta) P(d\xi) = \int L_{ij}(\xi, \theta) f(\xi) d\xi, \tag{4}$$

where P is the probability measure associated with ξ and $f(\cdot)$ its distribution function.

The vector of parameters θ is then estimated by maximizing the log-likelihood function, i.e. by solving the program

$$\max_{\theta} LL(\theta) = \max_{\theta} \frac{1}{I} \sum_{i=1}^{I} \ln P_{ij_i}(\theta), \tag{5}$$

where j_i is the alternative choice made by the individual i. This involves the computation of $P_{ij_i}(\theta)$ for each individual i, $i = 1, \ldots, I$, which is impractical since it requires the evaluation of one multidimensional integral per individual. The value of $P_{ij_i}(\theta)$ is therefore replaced by some approximation, obtained in

the Monte Carlo setting by sampling over ξ, and given by

$$SP_{ij_i}^R(\theta) = \frac{1}{R}\sum_{r=1}^{R} L_{ij_i}(\xi_r, \theta), \tag{6}$$

where R is the number of random draws ξ_r. As a result, θ is now computed as the solution of the simulated log-likelihood problem

$$\max_\theta SLL^R(\theta) = \max_\theta \frac{1}{I}\sum_{i=1}^{I} \ln SP_{ij_i}^R(\theta). \tag{7}$$

We will denote by θ_R^* a solution of this last approximate problem (often called the Sample Average Approximation, or SAA), while θ^* will represent a solution of the true problem (5).

When the same individual delivers several observations, rather than only one, these repeated choices are usually correlated, and we must consider the probability of the individual's choice sequence, instead of the particular observations. Typically, the tastes of a given decision-maker are then assumed to remain constant across choice situations for each particular respondent, such that tastes vary across individuals, but not across observations for the same individual. With j_{it} giving the alternative chosen by decision-maker i at time t ($t = 1, \ldots, T_i$), conditional on the realization ξ, the probability of the choice sequence is then

$$L_i^{T_i}(\xi, \theta) = \prod_{t=1}^{T_i} L_{ij_{it}}(\xi, \theta).$$

This leads to a new version of the log-likelihood function, given by

$$LL(\theta) = \frac{1}{N}\sum_{i=1}^{I} \ln \int L_i^{T_i}(\xi, \theta) f(\xi) d\xi,$$

where we use the number of observations $N \geq I$ as weight factor. The simulated log-likelihood function has a corresponding form.

2.2 Convergence of approximations and useful estimators

Bastin et al. (2005c) have shown that the mixed logit problem can be viewed as a generalization of a classical class of stochastic programming problems, allowing the extension of associated convergence results, as the the number of draws R tends to infinity, for a fixed population size (as is the case in most real applications), and taking an independently and identically distributed sample for each individual. They deduce in particular that for almost every sequence of random draws, there exists some limit point θ^* of (θ_R^*) that is first (second)-order critical for the true log-likelihood function under some

reasonable assumptions, if θ_R^* $(R = 1, \ldots)$ are first (second)-order critical for the corresponding SAA problem.

It is furthermore possible to estimate the error made by using the SAA problem (7) instead of the true problem (5), since it can be show that

$$N\left(LL(\theta) - SLL^R(\theta)\right) \Rightarrow \mathcal{N}\left(0, \sum_{i=1}^{I} \frac{\sigma_{ij_i}^2(\theta)}{R\left(P_{ij_i}(\theta)\right)^2}\right) \tag{8}$$

as R tends to infinity, where $\Rightarrow$ means convergence in distribution and where σ_{ij_i} is the standard deviation of $P_{ij_i}(\theta)$. Therefore, $SLL^R(\theta)$ is an asymptotically unbiased estimator of $LL(\theta)$, and the asymptotic radius of its confidence interval is

$$\epsilon_\delta^R(\theta) = \alpha_\delta \frac{1}{N} \sqrt{\sum_{i=1}^{I} \frac{\sigma_{ij_i}^2(\theta)}{R\left(P_{ij_i}(\theta)\right)^2}}, \tag{9}$$

where α_δ is the quantile of a $N(0,1)$, associated to some level of signification δ. We have set δ to 0.9 in our tests, leading to $\alpha_{0.9} \approx 1.64$. $\epsilon_\delta^R(\theta)$ can be numerically evaluated by replacing $\sigma_{ij_i}(\theta)$ and $P_{ij_i}(\theta)$ by their corresponding statistical estimators $\sigma_{ij_i}^R(\theta)$ and $P_{ij_i}^R(\theta)$.

Finally, the simulation bias for finite R can be approximated by

$$B^R(\theta) := E[SLL^R(\theta)] - LL(\theta) = -\frac{I^2\left(\epsilon_\delta^R(\theta)\right)^2}{2N\alpha_\delta^2}. \tag{10}$$

The validity of these estimators for practical purposes is discussed in Section 4.2. It should be noted that error and bias can be calculated analytically only for Monte Carlo sequences. The extension to quasi-Monte Carlo sequences is difficult because of their deterministic nature. However, by introducing some randomness in low discrepancy sequences, one can use statistical methods for error analysis (see for instance (Bastin et al., 2004) for a preliminary study in the context of MMNL models estimation).

3 A new algorithm for solving the SAA problem

The maximization of the simulated log-likelihood remains an expensive task since, as pointed out in the introduction, the population size I can be large as can the number of multidimensional integrals in the expression of the objective function. To perform this maximization, we consider here a trust-region algorithm that exploits statistical inference to limit the number of draws needed in the early iterations, away from the solution. The basic idea is to generate a sample set prior the optimization process, with $R_{\max}$ i.i.d. random draws per individual. At iteration k, only a (possibly small) subset of this sample set

will be used, by selecting R_k of the $R_{\max}$ random draws for each individual (for simplicity, the first R_k draws).

3.1 A trust-region approach

As other optimization techniques, a trust-region method is an iterative procedure for maximizing an objective function. The basic principle is as follows. Consider a current iterate θ_k at iteration k. A trial point $\theta_k + s_k$, is then calculated by maximizing a model m_k of the objective function inside a neighborhood $\mathcal{B}_k$ of θ_k, called the trust-region (see Step 3 in Algorithm (1) below). The trust-region is mathematically defined as

$$\mathcal{B}_k \stackrel{def}{=} \{\theta \in \mathbb{R}^m \,|\, \|\theta - \theta_k\| \leq \psi_k\} \,.$$

where ψ_k is called the trust-region radius. The model m_k is selected in such a way that it coincides to the true objective function as the trust-region radius tends to zero. The predicted and actual increases in objective function values are then compared (Step 4) in order to check if the model adequately represents the objective function for our maximization purpose. If the agreement is sufficiently good, the trial point becomes the new iterate (Step 5) and the trust-region radius is (possibly) enlarged (Step 6); the iteration is then said to be successful. If this agreement is poor, the trust region is shrunk in order to improve this correspondence between the model and the objective function. In addition, if the model predicted increase is large compared to the error of the simulated log-likelihood (which is dependent on the number of draws), while being a good approximation of this objective, we surmise that a less accurate approximation could be sufficient and therefore reduce the number of draws (Step 4). On the other hand, if the model prediction is poor compared to the accuracy of the objective function, we increase the number of draws in an attempt to correct this deficiency.

Since Monte Carlo simulation is used to compute our simulated likelihood objective function, a crucial ingredient to make our algorithm efficient is the technique which adapts the number of draws used at each iteration. Its main goal is to balance the number of draws (which should be kept as small as possible to reduce algorithmic costs) with bias and error in the value of the simulated likelihood. Its details are given in the Appendix.

A description of our algorithm follows. We will refer to it as the BTRDA algorithm, for basic trust-region with dynamic accuracy, by analogy with the basic trust-region (BTR) algorithm (Conn et al., 2000, Chapter 6). The two algorithms indeed coincide if we fix R_k to $R_{\max}$ ($k = 1, \ldots, \infty$). Some details are skipped for the sake of clarity, but can be found in Appendix.

Algorithm 1: Trust-region maximization algorithm (BTRDA)

Step 0. Initialization. Let *tol* be a (user-defined) tolerance. An initial point θ_0 and an initial trust-region radius ψ_0 are given. The minimum number of draws $R^0_{\min}$, the (user-defined) maximum number of draws $R_{\max}$ and the initial number of draws $R_0 \in [R^0_{\min}, R_{\max}]$ are also given. Compute $SLL^{R_0}(\theta_0)$ and set k, the iteration index, and t, the number of successful iterations, both to 0.

Step 1. Stopping test. Stop if the maximum number of draws has been reached and the relative gradient (see Appendix) is less than the tolerance. In order to avoid objective function increases insignificant compared to simulation noise, also stop if the relative gradient is less than a fraction of the simulation error (9).

If $R_k < R_{\max}$, safeguard against premature convergence to inaccurate first-order critical points and increase the minimum number of draws if no significant increase of the log-likelihood has been observed for small numbers of draws (see Algorithm 2 in Appendix).

Step 2. Model definition. Define a model $m_k^{R_k}$ of $SLL^{R_k}(\theta)$ in the trust region $\mathcal{B}_k$

Step 3. Step calculation. Compute a step s_k that sufficiently increases the model $m_k^{R_k}$ and such that $\theta_k + s_k \in \mathcal{B}_k$; set $\Delta m_k^{R_k} = m_k^{R_k}(\theta_k + s_k) - m_k^{R_k}(\theta_k)$.

Step 4. Comparison of actual and predicted increases. Set $R^- := R_k$ and compute a new number of draws $R^+ \geq R^k_{\min}$ (see Algorithm 3 in Appendix). Define

$$\rho_k := \frac{SLL^{R^+}(\theta_k + s_k) - SLL^{R^-}(\theta_k)}{\Delta m_k^{R_k}}. \tag{11}$$

If $\rho_k < 0.01$ and $R_k \neq R^+$, modify R^- or the candidate number of draws R^+ to take account of simulation bias and error differences, and update ρ_k (see Algorithm 4 in Appendix).

Step 5. Acceptance of the trial point. If $\rho_k < 0.01$, define the next iterate θ_{k+1} as θ_k, and set $R_{k+1} := R^-$. Otherwise accept the candidate iterate by defining $\theta_{k+1} := \theta_k + s_k$, set $R_{k+1} := R^+$ and increment t by one.

Step 6. Trust-region radius update. Set

$$\psi_{k+1} = \begin{cases} \min\{10^{20}, \max(2\|s_k\|, \psi_k)\} & \text{if } \rho_k \geq 0.75, \\ \frac{1}{2}\psi_k & \text{otherwise.} \end{cases}$$

Increment k by 1 and go to Step 1.

Note that the algorithm only consider models where the population heterogeneity ensures that the maximum number of draws $R_{\max}$ is always at-

tained (Bastin et al., 2005b). This is not necessary in the simpler multinomial logit case, and the procedure can be stopped in Step 1 in this case provided both simulation error and relative gradient are less than the pre-defined tolerance, even if the maximum number of draws has not been reached.

It can be proved (Bastin et al., 2005b) that the BTRDA Algorithm is globally convergent, i.e., under suitable smoothness assumptions, that the algorithm converges to a local first-order critical point of the SAA problem (7) from any starting point (and not just from starting points that are close enough to the local solution).

4 Numerical assessment

4.1 AMLET

The validation of the proposed methodology has been performed with our software AMLET (for Another Mixed Logit Estimation Tool), that is available in open-source at `http://www.grt.be/amlet`. We analyze the package on synthetic and real data, and in this last case, we compare obtained results to those delivered by Gauss 5.1 and the MaxLik module (Schoenberg, 2001) (in which we have used Halton and Monte Carlo sequences) and the codes written by Train (1999). All results were obtained on a Pentium IV 3Ghz with 1GByte of memory, under the Windows 2000 environment, and Cygwin for AMLET. We first assess the validity of the bias and error asymptotic estimators. We next compare different algorithmic options in the optimization process, in order to validate the choice of the trust-region framework. We then conclude the experimental framework by evaluating the package on a real dataset.

We always use linear utility functions. Furthermore, in experiments using simulated data, the attribute values are drawn from a standard univariate normal distribution $N(0,1)$ and the coefficient of each independent variable is also drawn from an univariate normal distribution $N(\frac{1}{2},1)$. The error term is generated from an extreme value (Gumbel) distribution, ensuring that the conditional choice probability follows the logit formula (3). This allows us to compute the utility of each alternative, including observed and unobserved terms. The individual choice is then identified for each observation as the alternative with the highest utility. The optimization starting point is defined by (arbitrarily) setting all initial parameter values to 0.1 and the initial number of draws is set to $\lceil 0.1 R_{\max} \rceil$. The tolerance for convergence was set to 10^{-6} (see Appendix).

Since the bias and error estimates (10) and (9) are only valid asymptotically as R increases, it is useful to assess them numerically for practical purposes, in particular when the simulation bias is significant. They could indeed be quite useless if the computed informations were poor for smaller numbers of draws. In order to assess these estimates, we consider a synthetic population of 5000 individuals facing 5 alternatives (one of which is the null alternative) for which the utilities involve 5 explanatory variables. We use 500, 1000, 2000, 3000, 4000, 5000 random draws per individual. For each fixed size of the draws set, we minimize 36 different SAA approximations (constructed on the basis of 36 independent samples of random draws, denoted by R_s, $s = 1, \ldots, 36$), resulting in 36 slightly different solutions (with non-negative components). We then compute the mean of these 36 optimal values $SLL_*^{R_s}$ to obtain a "mean optimal value", which we will denote by $\overline{SLL_*^R}$, and which can be viewed as an estimator of $E[SLL_*^R]$ with

$$SLL_*^R = \max_{\theta \geq 0} SLL^R(\theta). \tag{12}$$

Note that the variance of this $\overline{SLL_*^R}$ to $\sigma^2/36$, where σ^2 is the variance of SLL_*^R. For each of the 36 solutions, we furthermore estimate the standard deviation due to the sampling effect by recomputing the log-likelihood at the corresponding solution with 36 new, independent, samples. The mean of these newly estimated log-likelihood values is then an estimator of $E[SLL^R]$ at the considered solution, that we will denote by $\hat{E}_s[SLL^R]$ $(s = 1, \ldots, 36)$. The estimators $\hat{E}_s[SLL^R]$ $(s = 1, \ldots, 36)$ are, by construction, less sensitive to the simulation error than the corresponding values $SLL_*^{R_s}$, while they present the same expected bias with respect to the true objective value. This suggests to use them in order to exhibit the bias differences between the various numbers of draws, so we also compute their average:

$$\overline{\hat{E}[SLL^R]} = \frac{1}{36} \sum_{s=1}^{36} \hat{E}_s[SLL^R]. \tag{13}$$

while the optimal solutions slightly differ. Results are reported in Table 1, where we have indexed the quantities by the relevant equation reference. We also indicate the differences in mean optimal values, bias and $\overline{\hat{E}[SLL^R]}$, when the number of draws is increased. We then reproduce the experiment on a panel dataset, made of 1000 individuals, each one delivering 5 observations, for sampling sizes varying from 500 to 6000 random draws per individual. Corresponding results are summarized in Table 2.

We first observe that estimated and numerical standard deviations are similar, suggesting that the approximation (9) is adequate, even when the simulation

Table 1. Validation of error and bias estimation (cross-sectional data)

Number of draws	**500**	**1000**	**2000**	**3000**	**4000**	**5000**
Mean opt. value (12)	-1.4426396	-1.4420646	-1.4418132	-1.4416672	-1.4416542	-1.4416458
Mean opt. value diff.	-	0.0005750	0.0002514	0.0001460	0.0000131	0.0000083
Estimated std deviation (8)	0.0006931	0.0004914	0.0003485	0.0002843	0.0002459	0.0002206
Numerical std deviation	0.0006856	0.0004864	0.0003533	0.0002855	0.0002479	0.0002242
Estimated error (9)	0.0011401	0.0008088	0.0005729	0.0004682	0.0004055	0.0003628
Estimated bias (10)	-0.0012011	-0.0006044	-0.0003033	-0.0002025	-0.0001519	-0.0001216
Bias diff.	-	0.0005967	0.0003011	0.0001007	0.0000506	0.0000303
Average $\hat{E}[SLL^R]$ (13)	-1.4427775	-1.4421316	-1.4418430	-1.4417303	-1.4416928	-1.4416572
Average $\hat{E}[SLL^R]$ **diff.**	-	0.0006459	0.0002886	0.0001127	0.0000375	0.0000356

Table 2. Validation of error and bias estimation (panel data)

Number of draws	**500**	**1000**	**2000**	**3000**	**4000**	**5000**	**6000**
Mean opt. value (12)	-1.3115668	-1.3096944	-1.3089731	-1.3083892	-1.3084261	-1.3082348	-1.3082011
Mean opt. value diff.	-	0.0018723	0.0007213	0.0005839	-0.0000369	0.0001913	0.0000337
Estimated std deviation (8)	0.0011209	0.0008258	0.0006019	0.0004820	0.0004220	0.0003806	0.0003458
Numerical std deviation	0.0011856	0.0008246	0.0006197	0.0004877	0.0004184	0.0003790	0.0003524
Estimated error (9)	0.0018622	0.0013550	0.0009738	0.0008031	0.0006967	0.0006240	0.0005703
Estimated bias (10)	-0.0032046	-0.0016966	-0.0008764	-0.0005960	-0.0004485	-0.0003598	-0.0003005
Bias diff.	-	0.0015080	0.0008202	0.0002804	0.0001475	0.0000887	0.0000592
Average $\hat{E}[SLL^R]$ (13)	-1.3114255	-1.3097113	-1.3088024	-1.3085533	-1.3083735	-1.3082790	-1.3082355
Average $\hat{E}[SLL^R]$ **diff.**	-	0.0017142	0.0009089	0.0002491	0.0001798	0.0000945	0.0000435

bias is of the same order or even larger (for instance for the case of 500 draws). The evolution of the estimated bias (10) reflects well that of $\overline{\hat{E}[SLL^R]}$ (13), as indicated by the reported differences with increasing number of draws, but the parallelism is less clear between the bias and the mean optimal value $\overline{SLL_*^R}$ (note the decrease in this quantity when the number of draws changes from 3000 to 4000 in Table 2), as expected. This can indeed be explained by the standard error of the mean estimator since, for instance, the confidence interval radius at level 0.95 for $\overline{SLL_*^R}$ can be estimated to be 0.00014 and 0.00012 for the cases of 3000 and 4000 random draws, respectively. The bias diminution can therefore be dominated by the variance of the mean optimal value SLL_*^R. If all our θ values were identical (and in the neighborhood of the calculated 36 values), the associated confidence interval radii at level 0.95 associated to $\overline{\hat{E}[SLL^R]}$ would be approximately 0.000026 and 0.000022 for 3000 and 4000 random draws respectively, and the influence of bias can then be detected. In practice, the simulation bias is therefore difficult to quantify accurately when only approximate optimal values are available, since it can be masked by the variance of the simulated log-likelihood values.

Table 9 also exhibits the slow convergence in $O(\sqrt{R})$ of Monte Carlo approximations. However we observe that the bias decreases in absolute value faster than the error, this last reduction being in $O(R)$, as predicted by equation (10). An important additional observation is that bias and error are significantly higher with panel data than with cross-sectional ones. For instance if 1000 Monte-Carlo draws are used in the cross-sectional situation, we have to take 3000 draws per individual for the panel data in order to observe similar error and bias. This can be explained by the more complicated expression of the probability choice. Note however that the total number of draws is smaller in the panel case. This is also an indication that the number of draws that is necessary to achieve a satisfying accuracy is dependent of the model formulation.

4.3 Algorithmic options for optimization

Solving (7) requires the use of numerical optimization procedures, among which one of the most popular for calibrating discrete choices models is the BFGS linesearch method (Train, 2003, pages 225–226). This method is based on using a quasi-Newton direction where the Hessian of the log-likelihood function is approximated by the well-known BFGS variable-metric formula. A suitable step length is then computed along this direction to yield the final step. This method is acknowledged to be efficient whenever the function to optimize is concave. As this is not the case for mixed logit models, it is thus interesting to evaluate the relevance of trust-region methods, the contending methodology described in Section 3, which specializes in non-concave

problems.

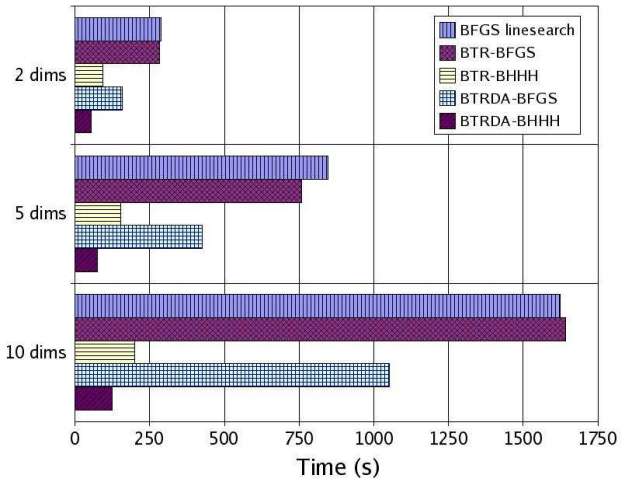
For comparison purposes, we therefore coded a BFGS algorithm, as described by Nocedal and Wright (1999), using the efficient linesearch technique by Moré and Thuente (1994) and associated code. Note that the BFGS algorithm in Gauss uses the StepBT line search, as described in Dennis and Schnabel (1983, Chapter 6). We then simulated three populations of 5000 individuals facing five alternatives (one of which is the null alternative), with associated utilities involving respectively two, five and ten explanatory variables. We used 2000 random draws and compared the BFGS, BTR and BTRDA algorithms for solving the SAA problem (7). In the trust-region framework, we use a quadratic model, defined as

$$m_k^R(\theta_k + s) = SLL^R(\theta_k) + \langle \nabla_\theta SLL^R(\theta_k), s \rangle + \frac{1}{2}\langle s, H_k s \rangle, \qquad (14)$$

where H_k is a symmetric approximation of $\nabla^2_{\theta\theta} SLL^R(\theta_k)$, chosen to guarantee convergence under some technical conditions (Bastin et al., 2005b). We have tested two different Hessian approximations, namely the BFGS and the BHHH (Berndt et al., 1974) Hessian updates. This leads to a total of 5 considered methods, whose optimization times are reported in Figure 1. For each dimension, the procedure was repeated for 10 different samplings. The three algorithms give similar optimal log-likelihood values for a particular sampling, that are indistinguishable in view of the simulation error. While the BTR algorithm equipped with the BFGS Hessian update gives similar optimization times to the BFGS line search, the use of BHHH significantly decreases them. We will however show in Section 5 that these conclusions cannot be extended to real data, since the BFGS update then performs better than the BHHH strategy. Unless otherwise stated, we will thus use the BFGS Hessian update in the next sections. We nevertheless see here that the trust-region approach is, in all cases, at least competitive with the BFGS linesearch method, and that the adaptive strategy always reduces the optimization time, independently of the Hessian update scheme. The interest of trust-region methods compared to linesearch techniques has also been observed by Bastin et al. (2005a).

5 Performance on a real data set

We finally test our algorithm on real data in order to evaluate its performance on a large-scale model system (other applications can be found in Bastin et al. (2003) and Bastin et al. (2005a)). The set, collected in Autumn 2002 in Brussels (Belgium), is a stated preference exercise. One of the main objectives of the survey was to test the propensity to switch from car to a more efficient Public Transport network, with better access, new high-speed lanes and improved comfort. It is only one part of a large survey conceived for the estimation



Method	Hessian update	2 dimensions	5 dimensions	10 dimensions
BFGS	BFGS	$287s$	$847s$	$1625s$
BTR	BFGS	$283s$	$757s$	$1641s$
	BHHH	$93s$	$153s$	$200s$
BTRDA	BFGS	$157s$	$426s$	$1051s$
	BHHH	$55s$	$74s$	$124s$

Figure 1. Computational times with various optimization algorithms

of the new transport regional model, called IRIS II. Car users (mainly commuters) were intercepted on the ring from 5:00 a.m. to 10:00 p.m. and were asked to fill out a questionnaire under the direct assistance of interviewers. They analyzed up to three scenarios based on their current trip and expressed their choices. The design contains 7 variables, of which 6 present 3 levels of variations and 1 has two levels of variation. In order to reduce the size of the total set we adopted an orthogonal design. The levels of variation depend on the total car distance from origin to destination; we distinguish three classes: less than 10 km, between 10 and 30 km, greater than 30 km. In summary, the variables and the levels were chosen as follows (in brackets we indicate value levels for classes of distance between 10-30 km and greater than 30 km):

Car Time: +5(10,20) min., +10(20,30) min., +20(30,50) min. compared to actual car time;
Car Cost: 0, +0.06 Euro/km compared to actual car cost;
Toll: 1, 3, 7 euros;
Delayed Departure Time: -45 min., +30 min., +60 min. on the actual departure time;
PT Time: -10(-20,-30) min., +5(0,-5), +15(+10,+10) min., compared to actual car time;
PT Cost (ticket + parking / per month): 25(40,50), 45(70,85), 75(105,130)

euros;

Comfort: no seats available-very crowded, no seats available-not crowded, seats available.

We selected, for the model presented here, only trips with work as final destination. After data cleaning, a total number of 2602 observations from 871 individuals were entered into the model. Four choice options were available to respondents: car, car with delayed departing time, car on a High Occupancy Vehicle dedicated lane (HOV) and Public Transport (PT). Each option was specified with a different utility for car drivers (CD) and for passengers (CP), giving a total of eight alternatives; in particular the High Occupancy Vehicle lane was toll free when at least two passengers shared the same car.

A total of 18 exogenous variables were included in the final specification, of which 4 alternative specific constants (car passenger with delayed departure time, car as driver on HOV lane, shared car on HOV and Public Transport), 7 level of service variables (congested and free flow time, cost, HOV toll, origin-destination distance, comfort on two levels of variations), 3 departure time variables, 2 variables representing socio-economic characteristics (being manager or self-employed) and the remaining describing trip characteristics (trip frequency per week, dummy for stopping to pick up/drop off children). Seven of the explanatory variables are randomly distributed, with two of them assumed to be lognormal (congested and free flow time coefficients) and the remaining five assumed to be normal. Results are summarized in Table 3.

Since, for this study, we were particularly interested in taste variation over new alternatives (HOV and shared car on HOV) and over departure time switches, we allowed those coefficients to vary normally. Although the mean value had the same sign (negative) as the correspondent value estimated with a multinomial logit model, we found a significant part of the population with a positive value for those parameters. In particular, 13% and 12% of the population have positive alternative specific constant for HOV and shared car; while about 15 % of the respondents are willing to leave home earlier or 30 minutes later. The average value of willingness to pay is 20.3 euros (median 13.8) during peak-hours and 17.3 euros (median 11.5) out-of peak; those values are consistent to what we found in a previous study conducted by our group on revealed preference data (Bernard et al., 2003). These results correspond to the best of a series of model variants tested. However, since these details are out of the scope of this paper, we refer the interested reader to Cirillo and Axhausen (2005) for more discussions on modeling issues.

The model has been estimated both with Gauss and AMLET, using the same starting point. The comparison with the most used commercial software is given to attest the efficiency of our method. Results are summarized in Table 3, where solutions obtained with Monte Carlo draws have been averaged

over ten simulations in order to evaluate the mean performance of the software, with different sequences, while in practice we often only use one simulation in order to derive the parameters. The alternatives associated to non-generic parameters are indicated between brackets, next to the parameter name. AMLET results are similar to those obtained by Gauss with 250 Halton sequences (125 Halton draws seem to be insufficient for this problem with seven dimensions of integration). The optimization time of Gauss heavily depends on the optimization routine chosen. Ruud's routine is the fastest with only 194 seconds and 125 Halton draws, as indicated in Table 5, but results are then unacceptable, as can be seen by comparing the various columns of the table. Using 250 draws, it took 627 seconds and gave adequate parameters compared to the Monte Carlo ones. We also have to keep in mind that this routine is basically a Newton-Raphson approach, where the BHHH Hessian update is used. As such, the convergence, while superlinear, is only local, not global. As a consequence, the method can only be used with care since it may actually fail to converge. This problem has indeed been encountered in experiments with other model specifications, for instance when three lognormally-distributed explanatory variables were included. Similar problems were also encountered with other real datasets (Cirillo and Axhausen, 2005). A linesearch globalization of the method can be found in the MaxLik module, which additionally uses the Dennis-Schnabel stopping test (see Appendix), while Ruud's routine stops for a small H^{-1} norm of the gradient. The algorithm is then much slower, giving optimization times similar with the BTRDA approach (combined to the BHHH Hessian update), and 8 times more draws. In our tests, running times dramatically increase in Gauss when Monte Carlo draws are used instead of Halton sequences. We report an average optimization time of approximately 42 minutes for Paul Ruud's routine, and 3h46 for MaxLik-BHHH optimization routine.

The BTRDA time decreases when the BFGS Hessian update is used, at variance with the tests performed on synthetic data. Figure 2 shows that the optimization time crucially depends on the sample for the BHHH case, while the algorithm is stabler with BFGS. This can be explained by the information identity (see for instance Train (2003), Section 8.7): for a correctly specified model at the true parameters, the covariance of the scores is equal to the negative of the expected Hessian. Unfortunately, perfect information remains elusive, and we only maximize an approximation of the log-likelihood. The BHHH update therefore may not converge to the Hessian of the objective, sometimes leading to poor performances close to the solution. This is clearly visible for BTR with BHHH, where three of ten runs have been especially time consuming, while seven have converged very fast. A similar phenomenon has been observed with Gauss and 1000 Monte Carlo draws, since two runs have badly performed with the Paul Ruud's routine, and one with MaxLik. We guess that, as is often the case, the function defined by our practical problem is more difficult to maximize than a synthetic data likelihood, and that

the BFGS Hessian update then recovers local concavity better. Finally, the BFGS MaxLik did not converge from our starting point, and numerical difficulties arose when using AMLET with line search: first iterations produced poor iterates far from the solution, leading to slow subsequent convergence, a phenomenon also pointed out by Bastin et al. (2005a). We have even occasionally observed a breakdown of the linesearch procedure (computing the step to the next iterate), preventing any additional progress. The step computation always starts from the steepest direction in our trust-region framework, avoiding such difficulties. The behavior of the BFGS algorithm could possibly be circumvented with adequate preconditioning or hybrid Hessian update strategies, but more investigations are needed to assess these strategies.

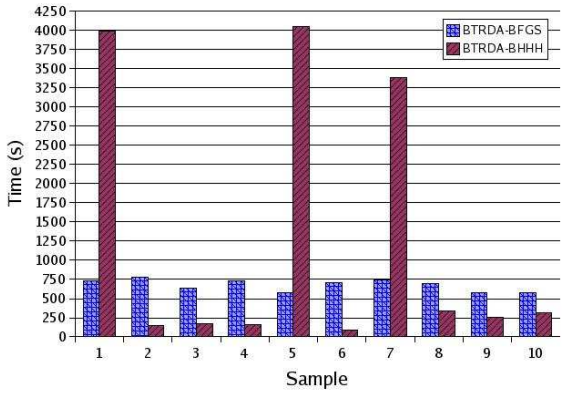


Figure 2. Optimization time variations (IRIS survey; 1000 MC)

The beneficial effect of the variable sample size strategy is illustrated in Figure 3, giving the evolution of the sample size R_k with the iteration index k on the left size, and with the objective value in the right graph. In particular we can see that the number of draws increases toward its maximum value only when the objective function's value is near to its maximum.

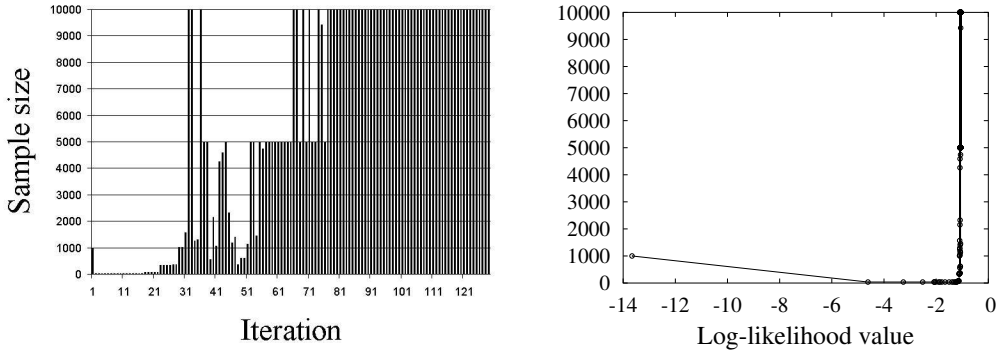


Figure 3. Number of draws variation

Table 3. IRIS survey model

Variable		Gauss			AMLET	
		125 Halton	250 Halton	1000 MC	1000 MC	10000 MC
Car Passenger (CP)	μ	-1.4369 (3.8)	-1.2927 (3.3)	-1.1122 (2.9)	-1.1431 (3.3)	- 1.1528 (3.2)
HOV (HOV)	μ	-5.2177 (9.1)	-5.1967 (10.0)	-5.1805 (10.18)	-5.1874 (9.6)	-5.2413 (9.9)
HOV (HOV)	σ	4.9080 (9.9)	4.7893 (9.9)	4.8834 (9.5)	4.8241 (10.4)	4.8988 (10.1)
Shared Car on HOV (HOVs)	μ	-6.8784 (12.6)	-7.1402 (12.3)	-7.1388 (11.7)	-7.1178 (11.9)	-7.2271 (11.5)
Shared Car on HOV (HOVs)	σ	6.0884 (11.0)	6.3635 (11.9)	6.2331 (10.4)	6.2356 (10.36)	6.3396 (10.0)
Public Transport (PT)	μ	-0.6515 (1.2)	-0.7498 (1.3)	0.9292 (1.5)	-0.9264 (1.76)	-0.9138 (1.7)
Congested Travel Time (LN)	μ	-0.0910 (22.5)	-0.0967 (21.9)	0.0903 (21.2)	-0.0959 (21.7)	-0.0960 (21.3)
Congested Travel Time (LN)	σ	0.1204 (10.8)	0.1243 (10.4)	0.1047 (8.9)	0.1123 (6.7)	0.1231 (5.73)
Free-Flow Travel Time (LN)	μ	-0.0769 (21.7)	-0.0910 (20.2)	-0.0893 (20.3)	-0.0894 (20.5)	-0.0890 (20.2)
Free-Flow Travel Time (LN)	σ	0.1029 (6.7)	0.1401 (10.1)	0.1316 (9.3)	0.1296 (7.0)	0.1283 (6.4)
Cost	μ	-0.2809 (4.8)	-0.2545 (4.1)	-0.2773 (4.3)	-0.2702 (5.0)	-0.2817 (5.15)
Toll (HOV)	μ	-0.4920 (8.0)	-0.4951 (8.1)	-0.5062 (8.0)	-0.5030 (9.8)	-0.5081 (9.80)
Dist. (CD,CP,CDs,CPs,HOV,HOVs)	μ	0.1893 (9.0)	0.2200 (9.5)	0.2145 (8.9)	0.2102 (10.9)	0.2131 (10.3)
Dist. (CD,CP,CDs,CPs,HOV,HOVs)	σ	0.2114 (8.7)	0.2570 (10.7)	0.2497 (9.8)	0.2471 (11.91)	0.2508 (11.3)
Trip Frequency - once a week (PT)	μ	2.6087 (2.5)	2.9988 (2.6)	3.1035 (2.7)	2.7604 (2.6)	3.2541 (3.0)
Comfort no-seats (PT,PTs)	μ	-0.9989 (3.9)	-1.1855 (4.3)	-1.1551 (4.2)	-1.1405 (4.4)	-1.1683 (4.44)
Comfort no seats, crowded (PT,PTs)	μ	-1.6903 (5.9)	-1.9029 (6.5)	-1.8638 (6.3)	-1.8732 (7.9)	-1.8787 (7.8)
Earlier Departure Time (CP,CPs)	μ	-3.2713 (7.7)	-3.2713 (7.7)	-3.1649 (7.8)	-3.1599 (7.7)	-3.2146 (7.4)
Earlier Departure Time (CP,CPs)	σ	2.4496 (5.3)	2.9170 (6.2)	2.8021 (5.6)	2.7707 (5.9)	2.8356 (5.8)
Later Departure Time (CP,CPs)	μ	-2.2247 (6.4)	-2.3160 (6.6)	-2.4354 (6.6)	-2.3985 (6.3)	-2.4569 (6.14)
Later Departure Time (CP,CPs)	σ	2.7758 (5.1)	2.3328 (4.7)	2.5768 (4.7)	2.4720 (4.4)	2.6486 (4.54)
Much later Departure Time (CP,CPs)	μ	-2.4615 (10.6)	-2.6428 (10.7)	-2.6667 (10.4)	-2.6820 (11.6)	-2.6795 (11.3)
Self-employed (CD,HOV)	μ	1.4941 (3.3)	1.9910 (3.8)	1.8023 (3.3)	1.8392 (3.9)	1.8631 (3.9)
Manager (HOV)	μ	1.4163 (2.1)	1.2797 (1.9)	1.2431 (1.8)	1.3295 (2.0)	1.2443 (1.8)
Number of cars - 3 per HHLD (CD)	μ	2.2339 (3.8)	2.0608 (3.3)	1.9836 (3.3)	2.0271 (3.3)	2.0732 (3.3)
Log-likelihood		-1.06036	-1.05651	-1.05882	-1.05853	-1.0577
Error		Not available	Not available	Not available	0.00204	0.00067
Bias		Not available	Not available	Not available	-0.00201	-0.00022

Table 4
Optimization times (IRIS survey)

Method	125 Halt.	250 Halt.	1000 MC
BHHH Ruud's rout. [1]	194s	627s	2540s
BHHH Max-Lik rout.	1084s	2265s	13576s
BFGS Max-Lik rout.	failure	failure	failure

[1]failure with other model specifications

Method	Hessian update	1000 MC	2000 MC
BFGS	BFGS	2292s [2]	3720s [3]
BTR	BFGS	1421s	2815s
	BHHH	4237s	7306s
BTRDA	BFGS	675s	1486s
	BHHH	1291s	1935s

[2]1 failure over 10 runs
[3]2 failures over 10 runs

6 Conclusion

We have developed a new algorithm for mixed logit model estimation. The proposed unconstrained stochastic programming method uses statistical inference to accelerate computation and has been implemented in the AMLET package. Numerical experimentations show that a trust-region method can handle the non-concavity of the problem better than a more traditional line-search scheme, suggesting that the choice of the optimization framework is of crucial importance. Results also show that a strategy that uses a variable number of draws in the estimation of the choice probabilities gives significant gains in the optimization time compared to usual approach with a fixed number of draws, while giving additional information on the adequation between of the Monte Carlo approximation and the true function, and not suffering of non-uniform coverage in high integration dimensions.

However, an extension of the method to low discrepancy sequences is not immediate, since error and bias are difficult to quantify in practice, even if they are usually lower than with Monte-Carlo sequences for the same number of draws. On the other hand, the use of Monte Carlo draws protects against the loss of uniform coverage in high dimensional problems, which makes the MC methods often more robust, both theoretically and numerically. Our procedure can therefore be seen as a good compromise between the necessity to speed up the estimation process and the exploitation of statistical information. Ongoing studies are trying to efficiently assess the simulation error obtained with quasi-

Monte Carlo techniques (Bastin et al., 2004).

Several research directions remain open. First of all, comparisons with more complex quasi-Monte Carlo methods are desirable (this issue has been partially investigated by Bastin et al. (2005a)). Secondly, further improvements of the algorithm are not impossible, possibly yielding additional computational gains. In particular, the impact of the Hessian update used in the optimization may be investigated further, as it appears to be computationally significant. However, since our observations differ between the synthetic and real data cases, no clear-cut practical conclusions can be stated at this point. More research is needed to investigate this issue, in particular by considering hybrid strategies, combining BHHH with other update strategies. Finally, the statistical error on the estimated parameters also deserves further analysis.

Acknowledgments

The authors would like to express their gratitude to Thierry Duquenne from Région de Bruxelles Capitale for granting access to the IRIS data set. Our thanks go also to Marcel Rémon for his helpful comments on statistical theory, and to the Belgian National Fund for Scientific Research for the grant that made this research possible for the first author and for its support of the third author during a sabbatical mission.

References

Bastin, F., Cirillo, C., Hess, S., 2005a. Evaluation of optimisation methods for estimating mixed logit models. Forthcoming in Transportation Research Record.

Bastin, F., Cirillo, C., Toint, Ph. L., 2003. Numerical experiments with AMLET, a new Monte-Carlo algorithm for estimating mixed logit models. Paper presented at the 10th International Conference on Travel Behaviour Research.

Bastin, F., Cirillo, C., Toint, Ph. L., 2004. Estimating mixed logit models with quasi-monte carlo sequences allowing practical error estimation. Paper presented at the European Transport Conference, Strasbourg, France.

Bastin, F., Cirillo, C., Toint, Ph. L., 2005b. An adaptive monte carlo algorithm for computing mixed logit estimators. Forthcoming in Computational Management Science.

Bastin, F., Cirillo, C., Toint, Ph. L., 2005c. Convergence theory for nonconvex stochastic programming with an application to mixed logit. Forthcoming in Mathematical Programming, Series B.

Bernard, P.-Y., Cirillo, C., Cornélis, E., Masquilier, B., 2003. Intégration du transport public au modèle de transport de personnes de la région wallonne. Tech. Rep. 2003/23, Transportation Research Group, University of Namur, Namur, Belgium.

Berndt, E. K., Hall, B. H., Hall, R. E., Hausman, J. A., 1974. Estimation and inference in nonlinear structural models. Annals of Economic and Social Measurement 3/4, 653–665.

Bhat, C. R., 2001. Quasi-random maximum simulated likelihood estimation of the mixed multinomial logit model. Transportation Research Part B 35 (7), 677–693.

Bhat, C. R., 2003. Econometric models of choice: Formulation and estimation. Paper presented at the 10th International Conference on Travel Behaviour Research.

Cirillo, C., Axhausen, K. W., 2005. Evidence on the distribution of values of travel time savings from a six-week diary. Forthcoming in Transportation Research Part A.

Conn, A. R., Gould, N. I. M., Toint, Ph. L., 2000. Trust-Region Methods. SIAM, Philadelphia, USA.

Dennis, J. E., Schnabel, R. B., 1983. Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice-Hall, Englewood Cliffs, New Jersey, USA.

Electric Power Research Institute, 1977. Methodology for predicting the demand for new electricity-using goods. Final Report EA-593, Project 488-1, Electric Power Research Institute, Palo Alto, California, USA.

Fishman, G. S., 1996. Monte Carlo: Concepts, Algorithms and Applications. Springer Verlag, New York, USA.

Garrido, R. A., 2003. Estimation performance of low discrepancy sequences in stated preferences. Paper presented at the 10th International Conference on Travel Behaviour Research.

Halton, J. H., 1960. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numerische Mathematik 2, 84–90.

Hensher, D. A., Greene, W. H., 2003. The mixed logit model: The state of practice. Transportation 30 (2), 133–176.

Hess, S., Train, K., Polak, J., 2005. On the use of a modified latin hypercube sampling (mlhs) approach in the estimation of a mixed logit model for vehicle choice. Forthcoming in Transportation Research Part B.

L'Ecuyer, P., Lemieux, C., 2002. Recent advances in randomized quasi-Monte Carlo methods. Vol. 46 of Internat. Ser. Oper. Res. Management Sci., 419–474. Kluwer Academic Publisher.

Moré, J. J., Thuente, D. J., 1994. Line search algorithms with guaranteed sufficient decrease. ACM Transactions on Mathematical Software 20 (3), 286–307.

Morokoff, W. J., Caflish, R. R., 1995. Quasi-Monte Carlo integration. Journal of Computational Physics 122 (2), 218–230.

Nocedal, J., Wright, S. J., 1999. Numerical Optimization. Springer, New York, USA.

Rubinstein, R. Y., Shapiro, A., 1993. Discrete Event Systems. John Wiley & Sons, Chichester, England.

Sándor, Z., Train, K., 2004. Quasi-random simulation of discrete choice models. Transportation Research Part B 38 (4), 313–327.

Schoenberg, R., 2001. Optimization with the quasi-Newton method. Aptech Systems, Inc., Maple Valley, WA, USA.

Shapiro, A., 2000. Stochastic programming by Monte Carlo simulation methods. SPEPS.

Shapiro, A., 2003. Monte Carlo sampling methods. In: Shapiro, A., Ruszczyński, A. (Eds.), Stochastic Programming. Vol. 10 of Handbooks in Operations Research and Management Science. Elsevier, 353–425.

Sivakumar, A., Bhat, C. R., Ökten, G., 2005. Simulation estimation of mixed discrete choice models using randomized quasi-monte carlo sequences: A comparison of alternative sequences, scrambling method, and uniform-to-normal variate transformation techniques. Forthcoming in Transportation Research Record.

Train, K., 1999. Halton sequences for mixed logit. Working paper No. E00-278, Department of Economics, University of California, Berkeley.

Train, K., 2003. Discrete Choice Methods with Simulation. Cambridge University Press, New York, USA.

A BTRDA algorithm details

We give here some further practical details of the BTRDA algorithm described in Section 3.

A.1 Stopping test

The stopping criterion used in Step 1 of Algorithm 1 is a modification of the classical test based on the relative gradient (Dennis and Schnabel, 1983, Chapter 7). The algorithm is terminated as soon as

$$g_{rel}(\theta) \stackrel{def}{=} \max_c \left\{ \frac{|[\nabla_\theta SLL^R(\theta)]_c| \ \max\{|\theta_c|, 1.0\}}{\max\{|SLL^R(\theta)|, 1.0\}} \right\} \leq tol, \qquad \text{(A.1)}$$

where v_c is the c-th component of the vector v. This standard test however usually leads to final iterations exhibiting increases so small that the gain in the objective function cannot be discriminated against the simulation noise. This leads us to adapt the condition to this situation by only requiring that the left-hand side of (A.1) is less than a fraction of the error. In practice, we stop if $R_k = R_{\max}$ and if $g_{rel}(\theta) \leq \max\{tol, 0.1\epsilon^R_{0.9}\}$.

A.2 Safeguarding against premature convergence

As stated in Step 2 of Algorithm 1, safeguards are applied in avoid convergence to "phantom" first-order critical points that only appear at intermediate numbers of draws, but not at higher ones. In particular, we have to exclude the pathological case in which θ_k is a first-order critical point for SLL^{R_k}, with $R_k \neq R_{\max}$. If $\epsilon^{R_k}_\delta(\theta_k) > tol$, the algorithm does not stop, but since the model (14) is quadratic, no increase is achieved if $-H_k$ is positive definite, and the algorithm then breaks down. In order to circumvent this problem, we force an increase of R_{k+1}, should this situation occur. We nevertheless point out that this feature was never triggered in our experiments. Indeed, the gradient norm usually changes slowly in the vicinity of such a critical point, and a small gradient typically leads to a small model increase, which itself then causes the number of draws to increase with the effect that $R_{\max}$ is always reached in practice before the safeguard is activated.

We also increase the minimum number of draws when the adaptive strategy does not provide sufficient numerical gains, as a safeguard ensuring that $R_{\max}$ will be reached during the final iterations.

These safeguards are formally described in Algorithm 2 below. We first define two $R_{\max}$-dimensional vectors v and l. The first of these vectors is used to remember the value of the simulated likelihood at the start of the last set of consecutive iterations where the corresponding number of draws was last used. At iteration $k = 0$, $v(R_0)$ is set to $SLL^{R_0}(z_0)$ and all other components to $-\infty$. At the beginning of iteration $k > 0$, $v(i)$ then contains $SLL^i\left(\theta_{h(i)}\right)$ where $h(i)$ is the index of the last iteration for which $R_{h(i)-1} \neq R_{h(i)} = i$ if size i has already been used, or $-\infty$ if the size i has not been used yet. The vector l contains in position i the number of successful iterations since iteration $h(i)$ (included), or -1 if size i has not been used yet. (At iteration $k = 0$, $l(R_0)$ is set to 0 and all other components to -1.) Recall also that t contains the total number of successful iterations encountered until the current iteration k (included).

Algorithm 2: Safeguard against premature convergence

If the relative gradient is smaller than the tolerance but the number of draws is strictly less than $R_{\max}$, increase R_k to some value not exceeding $R_{\max}$, to ensure that the relative gradient is strictly larger than the tolerance if $R_{k+1} \neq R_{\max}$.

If $R_k = R_{k+1}$ or if

$$SLL^{R_{k+1}}(R_{k+1}) - v(R_{k+1}) \geq \frac{1}{2}\nu_1[t - l(R_{k+1})]\epsilon_\delta^{R_{k+1}}(\theta_k), \tag{A.2}$$

set $R_{\min}^{k+1} = R_{\min}^k$. Otherwise increase the minimum number of draws by setting

$$\begin{cases} R_{\min}^{k+1} := \min\left\{\left\lceil \frac{R_k + R_{k+1}}{2} \right\rceil, R_{\max}\right\} & \text{if } R_k < R_{k+1}, \\ R_{\min}^{k+1} := R_{k+1} + 1 & \text{otherwise.} \end{cases}$$

If $R_k \neq R_{k+1}$, set $l(R_{k+1}) := t$ and $v(R_{k+1}) := SLL^{R_{k+1}}(\theta_{k+1})$.

The initial $R_{\min}^0$ has (arbitrarily) been set to 36 during our tests.

It can be shown that the right size of (A.2) is always positive (Bastin et al., 2005b), meaning that a violation of this condition corresponds to a poor increase of the objective function, for R_{k+1} draws. We again apply a different strategy if the number of draws decreases or increases. In the first case, bias difference and accuracy loss can explain a decrease or a too small increase of the SAA objective, but it is numerically cheaper to keep small sample sizes. In the second case, we use a more conservative approach in order to avoid poor increases of the SAA objective associated with large sample sizes resulting in important numerical costs.

We conclude the detailed description of our algorithm by indicating how the number of draws is balanced with bias and error in the value of the simulated likelihood.

The choice of R^+ in Step 3 of Algorithm 1 first attempts to trade number of draws and accuracy, and is described below.

Algorithm 3: Selection of the candidate number of draws

Define $\nu_1 \in (0,1)$. Use (9) to estimate the number of draws needed to equalize accuracy and model increase, that is

$$R^s := \min\left\{\max\left(R^k_{\min}, \left\lceil \frac{\alpha^2_\delta}{(I\Delta m_k^{R_k})^2} \sum_{i=1}^{I} \frac{\left(\sigma^{R_k}_{ij_i}(\theta)\right)^2}{\left(P^{R_k}_{ij_i}(\theta)\right)^2} \right\rceil\right), R^k_{\max}\right\}.$$

Compute the ratio between the model improvement and the estimated error,

$$\tau_1^k = \frac{\Delta m_k^{R_k}}{\epsilon_\delta^{R_k}(\theta_k)},$$

and the ratio between the current number of draws and the candidate size R^s for the next iteration:

$$\tau_2^k = \frac{R_k}{R^s}.$$

Then define

$$R' := \begin{cases} \min\left\{\lceil R_{\max}/2\rceil, \lceil R^s\rceil\right\} & \text{if } \tau_1^k \geq 1, \\ \min\left\{\lceil R_{\max}/2\rceil, \lceil \tau_1^k R^s\rceil\right\} & \text{if } \tau_1^k < 1 \text{ and } \tau_1^k \geq \tau_2^k, \\ \lceil R_{\max}/2\rceil & \text{if } \nu_1 \leq \tau_1^k < 1 \text{ and } \tau_1^k < \tau_2^k, \\ R_{\max} & \text{if } \tau_1^k < \nu_1 \text{ and } \tau_1^k < \tau_2^k. \end{cases}$$

Set $R^+ = \max\{R', R^k_{\min}\}$.

This somewhat technical part of the algorithm is motivated as follows.

(1) If $\tau_1^k \geq 1$, the model increase dominates the estimated error: we then reduce the number of draws to the minimum between $\lceil R^s\rceil$ and $\lceil R_{\max}/2\rceil$, in an attempt to increase numerical performance.

(2) If $\tau_1^k < 1$ the improvement is dominated by the inaccuracy. However, a small but repeated improvement over several consecutive iterations may lead to a global increase that is significant compared to the log-likelihood error, while keeping the computational costs lower than if $R_{\max}$ draws were used. In an attempt to exploit this fact, we then consider two cases.

(a) If $\tau_1^k \geq \tau_2^k$, the ratio between the current number of draws and the potential next one is less than the ratio between the model increase and the estimated error. A larger sampling would result in an error decrease, and so in a larger τ_1^k for a similar model increase. We therefore capitalize on τ_1^k by computing a sample size lower than R^s, such that an increase of order $\epsilon_{0.9}^{R_k}(\theta_k)$ would be reached in approximately $\lceil \tau_1^k \rceil$ iterations if τ_1^j is similar to τ_1^k for j close to k.

(b) If $\tau_1^k < \tau_2^k$, it may nevertheless be cheaper to continue to work with a smaller number of draws, defined again as $\lceil R_{\max}/2 \rceil$, as long as τ_1^k exceeds some threshold $\nu_1 > 0$ (set to 0.2 in our tests). Below to this threshold, we consider that the increase is too small to allow a less than maximum number of draws.

If R^+ is not equal to R_k, an additional complication unfortunately occurs in that the computation of

$$SLL^{R^+}(\theta_k + s_k) - SLL^{R_k}(\theta_k)$$

in Algorithm 1 is affected by the change both in simulation bias and error. This may in turn lead to a small or negative ratio ρ_k and an unsuccessful iteration, even when the model $m_k^{R_k}$ is a good approximation of the objective function for a number of draws equal to R^k. In particular, $SLL^{R^+}(\theta)$ may exceed $SLL^{R_k}(\theta_k)$ for all θ in a neighborhood of θ_k. This imposes to possibly redefine ρ_k in Step 5 of Algorithm 1, using a procedure that distinguishes between the cases where $R^+ > R_k$ and $R^+ < R_k$. In this latter case, the absolute value of the bias increases, irrespective of variations of the objective due to changes in simulation error, and the objective function is usually worse for a fixed θ. We then introduce a new candidate number of draws R^b in an attempt to equilibrate bias and model increase, as described below.

Algorithm 4: Number of draws revision when $\rho_k < 0.01$ and $R_k \neq R^+$

If the candidate number of draws R^+ is less than R_k, compute the number of draws R^b that gives, according to (10), a bias equal to the predicted increase, that is

$$R^b := \left\lceil \frac{1}{2\Delta m_k^{R_k} I} \sum_{i=1}^{I} \frac{\left(\sigma_{iji}^{R_k}(\theta)\right)^2}{\left(P_{iji}^{R_k}(\theta)\right)^2} \right\rceil .$$

If $R^+ < R^b < R_k$, set $R^+ := R^b$ and recompute ρ_k from (11).

If the (possibly recomputed) ρ_k is still strictly less than 0.01, compare R^+ and R_k. If $R^+ > R_k$, compute $SLL^{R^+}(\theta_k)$, $\Delta m_k^{R^+}$ and $\epsilon_\delta^{R^+}(\theta_k)$, else compute $SLL^{R_k}(\theta_k + s_k)$. Set R^- to $\max\{R_k, R^+\}$, and update ρ_k using (11).