I T L S

WORKING PAPER
ITLS-WP-06-14

Approximation of Bayesian Efficiency in Experimental Choice Designs

By

Michiel CJ Bliemer[1,2], John M Rose[*] & Stephane Hess[1]

[1]Faculty of Economics and Business
Institute of Transport and Logistics Studies
The University of Sydney 2006 Australia

[2] Delft University of Technology
Faculty of Civil Engineering & Geosciences,
Transport & Planning Department
Delft The Netherlands

[*] Corresponding author

June 2006

ISSN 1832-570X

INSTITUTE of TRANSPORT and LOGISTICS STUDIES

The Australian Key Centre in Transport and Logistics Management

The University of Sydney

Established under the Australian Research Council's Key Centre Program.

NUMBER: Working Paper ITLS-WP-06-14

TITLE: Approximation of Bayesian Efficiency in Experimental Choice Designs

ABSTRACT: This paper compares different types of simulated draws over a range of number of draws in generating Bayesian efficient designs for stated choice studies. The paper examines how closely pseudo Monte Carlo, quasi Monte Carlo and polynomial cubature methods are able to replicate the true levels of Bayesian efficiency for SC designs of various dimensions. The authors conclude that the predominantly employed method of using pseudo Monte Carlo draws is unlikely to result in leading to truly Bayesian efficient SC designs. The quasi Monte Carlo methods analyzed here (Halton, Sobol, and Modified Latin Hypercube Sampling) all clearly outperform the pseudo Monte Carlo draws. However, the polynomial cubature method examined in this paper, incremental Gaussian quadrature, outperforms all, and is therefore the recommended approximation method for the calculation of Bayesian efficiency of stated choice designs.

KEY WORDS: ***Experimental design, Gaussian quadrature, Halton sequences, Modified Latin Hypercube Sampling, Sobol sequences, Stated Choice experiments***

AUTHORS: Michiel CJ Bliemer[1,2], John M Rose & Stephane Hess[1]

CONTACT: Institute of Transport and Logistics Studies **(C37)**
An Australian Key Centre
The University of Sydney NSW 2006 Australia

Telephone: +61 9351 0071
Facsimile: +61 9351 0088
E-mail: itlsinfo@itls.usyd.edu.au
Internet: http://www.itls.usyd.edu.au

DATE: June 2006

1. Introduction

The generation of stated choice (SC) experiments has evolved to become an increasingly important, but complex component of SC studies (Burgess and Street 2003; Carlsson and Martinsson 2003; Ferrini and Scarpa 2006; Huber and Zwerina 1996; Kanninen 2002; Kessels et al. 2006; Kuhfeld et al. 1994; Lazari and Anderson 1994; Sándor and Wedel 2001, 2002, 2005; Street and Burgess 2004; Street et al. 2001). Typically, SC experiments present sampled respondents with a number of hypothetical scenarios (known as choice situations) consisting of a universal but finite number of alternatives that differ on a number of attribute dimensions. These respondents are then asked to specify their preferred alternative from the set of alternatives presented within each choice situation based on the attribute levels shown. These responses are then pooled both over hypothetical choice scenarios and respondents before being used to estimate parameter weights for each of the design attributes (or in some cases, even attribute levels). Depending on the type of experiment conducted, researchers may obtain estimates of the direct or cross elasticities (or marginal effects) of the alternatives as well as the marginal rates of substitution respondents are willing to make in trading between two attributes (i.e., willingness to pay measures).

Traditionally, researchers have relied upon the use of orthogonal experimental designs to populate the hypothetical choice situations shown to respondents (see Louviere et al., 2000, for a review of orthogonal designs). More recently however, some researchers have begun to question the relevance of orthogonal designs when applied to SC experiments (e.g., Huber and Zwerina 1996; Kanninen 2002; Kessels et al. 2006; Sándor and Wedel 2001, 2002, 2005). Generally, the argument against the use of orthogonality as a design criterion in the construction process is that the property of orthogonality is unrelated to the desirable properties of the econometric models used to analyse SC data (i.e., logit and probit models). The orthogonality (or otherwise) of an experimental design relates to the correlation structure between the attributes of the design with designs in which all between attribute correlations are zero being said to be orthogonal (in some cases, this definition of an orthogonal design may be relaxed to define orthogonality as occurring when all attribute correlations are zero within alternatives but not necessarily between alternatives; see Louviere et al. (2000) for a discussion on sequential versus simultaneous generation of orthogonal designs). Whilst orthogonality is an important criterion to determine independent effects in linear models, discrete choice models are not linear (Train 2003). In models of discrete choice, the correlation structure between the attributes is not what is of importance. Rather, given the derivation of the models, it is the correlations of the differences in the attributes which should be of concern.

Huber and Zwerina (1996) took the important step of relating the statistical properties of the SC experiments to the econometric models estimated on such data. In their paper, Huber and Zwerina showed that designs that let go of orthogonality as a consideration in generating SC experiments and which attempt to reduce the asymptotic standard errors of the parameter estimates (i.e., the square roots of the diagonal elements of the asymptotic variance-covariance (AVC) matrix) will generally result in designs that either (i) improve the reliability of the parameters estimated from SC data at a fixed sample size or (ii) reduce the sample size required to produce a fixed level of reliability in the parameter estimates with a given experimental design. The linking of the experimental design generation process to attempts to reduce the asymptotic standard

errors of the parameter estimates has resulted in a class of designs known as 'efficient designs' where designs that produce smaller asymptotic standard errors are thought of as being more efficient.

In order to calculate the AVC matrix for a SC design, the analyst requires *a priori* knowledge of the utility functions for that design. This is because the values of the AVC matrix are directly dependent upon both the attribute levels and the choice probabilities of the alternatives contained within each of the designs choice situations. The choice probabilities for a given design are in turn a function of the attribute levels of the alternatives as well as the parameter weights associated with each of these attributes. As such, the parameter values play a key role in determining the level of efficiency of a design. Unfortunately, the exact parameter values are unlikely to be known at the design construction phase, and as such, the researcher may have to make certain assumptions as to what values (termed priors) these will be in order to generate an efficient design.

Three different approaches have been used in the past regarding the parameter priors assumed in generating efficient SC experiments. In the first approach, researchers have made the strong assumption that all parameter priors for the design are simultaneously equal to zero (e.g., Burgess and Street 2003; Huber and Zwerina 1996; Street and Burgess 2004; Street et al. 2001). Whilst such an assumption is likely to aid in locating truly optimal designs (i.e., the most efficient design), optimality will only exist under the assumption of zero parameter estimates. Furthermore, the assumption of zero parameter priors is unlikely to hold in reality (and if it does, then there exist significant implications in terms of the attributes and/or levels used in the SC study). Thus, the efficiency of a design generated under such an assumption is highly unlikely to translate through to the data if the true parameter estimates are not zero in reality. A second approach that has sometimes been used is to assume that the parameter priors are non-zero and known with certainty (e.g., Carlsson and Martinsson 2003; Rose and Bliemer 2005). In such an approach, a single fixed prior is assumed for each attribute. Whilst the assumption of perfect certainty is a strong one, the design generation process is such that researchers are able to test its impact on a design's efficiency assuming misspecification of the priors. Sándor and Wedel (2001) introduced a third approach by relaxing the assumption of perfect *a priori* knowledge of the parameter priors through adopting a Bayesian approach to the design generation process. Rather than assume a single fixed prior for each attribute, the efficiency of a design is now determined over a number of draws taken from prior parameter distributions assumed by the researcher. Different distributions may be associated with different population moments representing different levels of uncertainty with regards to the true parameter values[1].

The Bayesian approach to constructing efficient SC experiments requires that the efficiency of a design be evaluated over numerous different draws taken from the prior parameter distributions assumed in generating the design. The Bayesian efficiency of a design is then calculated as the expected value of whatever measure of efficiency is assumed over all the draws taken. The Bayesian approach therefore necessitates the use of simulation methods to approximate the expectations for differing designs.

[1] For example, Sándor and Wedel (2001, 2002, 2005) assume normal distributions with different means and standard deviations, whilst Kessels et al. (2006) assume uniform distributions ranging between -1 and 1; the latter assumes no knowledge about the magnitude of the true parameter value other than it being between the suggested range, nor about the direction of the parameter.

A number of different simulation procedures are available to researchers, with the simplest being the use of pseudo random draws. In using pseudo random draws (often referred to as pseudo Monte Carlo, or PMC, draws), points from a distribution are randomly selected. Whilst simple in practice to implement, results obtained using PMC draws are susceptible to being specific to the particular draws taken from whatever distribution is assumed, with different sets of random draws likely to produce different coverage over the distribution space, possibly leading to widely different results when calculating the expectations. This risk is especially high with the use of a low number of draws. The precision of simulation processes may potentially be improved by using a more systematic approach in selecting points when sampling from a distribution. Such techniques are commonly referred to within the literature as quasi random Monte Carlo draws (see, for example, Bhat 2001, 2003; Hess et al. 2005; Sándor and Train 2003). The potential to provide better coverage of the distribution space for each prior parameter distribution should theoretically result in a lower approximation error in calculating the simulated choice probabilities for a given design. This in turn will result in greater precision in generating the design's AVC matrix, resulting in greater precision in terms of the Bayesian efficiency measure of that design. Other methods, such as polynomial cubature, also aim to minimize the approximation error when calculating the Bayesian efficiency.

Independent of the type of draws used, the researcher must decide on the number of draws to use. If too few draws are taken, it is probable that the resulting Bayesian measure of efficiency will be far from the true efficiency for a given design. If too many draws are used, the computation time in generating an efficient design will be unnecessarily high. The issue therefore becomes one of how many draws should be used before the Bayesian measure of efficiency will converge to the true efficiency level for a given design, or alternatively, fall within some acceptable error range around the true value. Unfortunately, the answer to this question will likely depend on the dimensions of the design itself, the number of Bayesian priors assumed, the population of the prior distributions, the type of econometric model used, as well as the type of draws employed.

The purpose of this paper is to examine over a range of draws, the performance of various forms of draws in approximating the true level of efficiency for a number of different designs. This paper compares the performance of the PMC method to three different types of quasi random Monte Carlo draws, namely Halton, Sobol, and Modified Latin Hypercube Sampling (MLHS) draws, and one polynomial cubature method, namely Gauss-Hermite approximation. In making our comparisons, we vary not only the number of draws but also the dimensions of the designs. In doing so, we are able to make recommendations as to what are the best types of draws to use as well as how many to use when generating designs of different dimensions.

The remainder of the paper is as follows. In the following section, we define efficiency as related to SC experimental designs. Section 3 further details Bayesian efficiency for SC experiments and discusses each of the approximation methods in more detail. Section 4 provides case studies in which we compare the performance of the types of draws varying the number of draws taken over a range of different experimental designs. Section 5 provides a discussion and conclusion to the paper.

2. Efficiency of Experimental Designs for Discrete Choice Models

Historically, efficiency when dealing with SC studies has generally been related to how statistically *reliable* the parameters in a discrete choice model will likely be when estimated using data obtained from a SC experiment. Reliability of the parameters has been defined in terms of the asymptotic standard errors of the model to be estimated where improvements in reliability suggest a reduction in the asymptotic standard errors and hence an increase in the asymptotic *t*-ratios of the model estimates. As such, the use of more efficient designs leads to an expectation that a lower number of respondents will be required to produce statistically significant parameter estimates for a given SC study when compared to less efficient designs.

Measurement of the (in)efficiency of a SC experimental design is typically expressed in terms of some form of *error* (e.g., D-error, A-error) derived from the AVC matrix for the design. Both the dimensions of the AVC matrix of a design and the values that populate it will influence the (in)efficiency of the design. In turn, the AVC of a design will depend on the following:

(a) *Econometric form of the discrete choice model estimated*

Different discrete choice models (e.g., multinomial logit (MNL), nested logit (NL), mixed logit (ML) models) lead to different AVC matrices;

(b) *Experimental design*

Different choice situations (i.e., different combinations of attribute levels in each choice situation) lead to different AVC matrices; and

(c) *Prior parameter values*

Different assumptions made regarding the *true* values of the parameter estimates result in different AVC matrices.

Let the AVC matrix be denoted by $\Omega(\tilde{\beta} \mid X)$, where $\tilde{\beta}$ represents the prior parameter values and X the attribute levels in the underlying experimental design. This matrix can be determined (analytically or by simulation) for each of the discrete choice models, see Appendix A. The D-error, describing the inefficiency of a design, can be expressed as

$$f(\tilde{\beta} \mid X) = \det\left(\Omega(\tilde{\beta} \mid X)\right)^{1/K}, \tag{1}$$

where K is the number of parameters[2]. The lower this D-error, the higher the overall efficiency of the design will be. Hence, given the prior parameters and the discrete choice model, the aim in creating the experimental design is to find attribute levels X such that this D-error is as low as possible. The design with the lowest D-error is termed D-optimal. Other (in)efficiency measures exist which may be substituted for D-error.

[2] In Rose and Bliemer (2005), it is suggested that the rows and columns for the parameters representing constants in the model should be removed from the AVC matrix when computing the D-error, as they may dominate the D-error while having no clear efficiency meaning in a stated choice experiment.

For example, some researchers prefer A-error, where Equation (1) now becomes (replacing the determinant by the trace of the matrix and normalizing it by dividing by the number of parameters):

$$f(\tilde{\beta} \mid X) = \frac{\mathrm{tr}\left(\Omega(\tilde{\beta} \mid X)\right)}{K}. \tag{2}$$

Up to this point, it has been assumed that the prior parameter values are fixed and known. Typically however, the parameters are unknowns to be estimated in the model using the data collected from some underlying SC experiment. The literature has suggested some starting points for identifying prior parameter values that may be useful in constructing efficient SC experiments such as obtaining them from previous studies, focus groups, pilot studies, managers, etc. (Sándor and Wedel 2001). Nevertheless, priors obtained using these methods will likely exhibit a certain degree of uncertainty. Unfortunately, an efficient experimental design is only efficient for the specified prior parameter values assumed; hence, if the priors are incorrectly specified, the experimental design may become less efficient[3]. In order to generate a more robust experimental design incorporating uncertainty in the parameter priors assumed (i.e., the design generation process does not solely depend on fixed priors), a Bayesian approach has been proposed within the literature (Sandor and Wedel 2001). Using this approach, (a subset of) prior parameters are assumed to have random distribution(s) rather than fixed values. Such designs are known as Bayesian efficient designs if the *expected* efficiency is high (or the associated expected error low). Let $\phi(\tilde{\beta} \mid \theta)$ denote the multivariate probability density function of the priors $\tilde{\beta}$, where θ are the corresponding parameters of this distribution (e.g., if $\tilde{\beta}$ follows a normal distribution, then θ represents the means and standard deviations of this multivariate normal distribution). The Bayesian D-error (or A-error) can then be written as the expected D-error (A-error),

$$E(f) = \int_{\tilde{\beta}} f(\tilde{\beta} \mid X)\phi(\tilde{\beta} \mid \theta)d\tilde{\beta}. \tag{3}$$

Minimizing the Bayesian D-error (denoted D_b-error, as opposed to the D_z-error and the D_p-error assuming zero and fixed priors, respectively) will yield a D_b-optimal experimental design. Unfortunately, computation of the above integral is complex as it cannot be calculated analytically. Therefore, it has to be approximated, typically by simulation. Approximation of this integral can be time consuming, especially given the fact that in general, millions of experimental designs may need to be evaluated when searching for a (Bayesian) efficient design, and where each evaluation requires simulation. For realistically sized experimental designs with many randomly distributed parameters, this may not be feasible if the D_b-error cannot be computed quickly. In the next section, different approximations are outlined and discussed. We show that there are much better (and faster) approximation methods available than those currently used by most researchers.

[3] In some cases, a design may actually be more efficient under mis-specified prior values.

3. Approximation of Bayesian Efficiency

In this section we will describe several different methods for approximating the D_b-error as stated in Equation (3). Three main types of approximations are considered, namely (a) pseudo-random Monte Carlo (PMC) simulation, (b) quasi-random Monte Carlo simulation, and (c) polynomial cubature. The most common method is PMC simulation, which is currently used by all but few researchers[4].

Independent of the method, the principles in generating efficient SC experiments remain the same:

1) first, *R* values from the random distribution of the prior parameter values are drawn;
2) next the D-error is evaluated for each one of the values; and
3) an average D-error is computed over these values (giving the D_b-error).

The PMC and quasi-random MC methods all take a simple (unweighted) average of the different D_b-errors, but differ in the way they take the draws from the random distribution. In the PMC method, these draws are completely *random*, whereas in the quasi-random MC methods they are intelligent and structured, and in most cases deterministic. The polynomial cubature methods construct intelligent and deterministic draws as well, but also determine specific weights for each draw and compute a weighted average.

Rather than drawing from a multivariate distribution, all methods generally use independent draws from univariate distributions for each random prior $\tilde{\beta}_k$, under the assumption that all parameters are independent. Under this assumption, Equation (3) can be written as

$$E(f) = \int_{\tilde{\beta}} f(\tilde{\beta} \mid X)\phi(\tilde{\beta} \mid \theta)d\tilde{\beta} = \int_{\tilde{\beta}_1} \cdots \int_{\tilde{\beta}_K} f(\tilde{\beta} \mid X)\phi_k(\tilde{\beta}_k \mid \theta_k)d\tilde{\beta}_1 \cdots d\tilde{\beta}_K. \quad (4)$$

Equation (4) also allows priors to have different forms of random distributions, such as mixing priors with a normal and a uniform distribution. The distribution parameters θ_k will determine the mean prior value and the standard deviation (uncertainty) of that prior. Hence, one can include uncertainty for each prior parameter by specifying the corresponding random distribution.

Below, each of the approximation methods will be outlined.

[4] Sándor and Wedel (2002, 2005) adopt a quasi random Monte Carlo approach; orthogonal array-based Latin hypercube sampling, and randomly shifted good lattice points, respectively. All other papers reviewed appear to use PMC methods.

3.1 Pseudo-Random Monte Carlo (PMC) Simulation

In PMC simulation, for each of the *K* parameters, *R* independent draws are taken from their given prior distributions. For each of these *R* draws of the prior parameters, the D_b-error is computed. Finally, the average is taken of all computed D-errors. Let $\tilde{\beta}^{(r)} = [\tilde{\beta}_1^{(r)}, \ldots, \tilde{\beta}_K^{(r)}]$ denote draw *r*, $r = 1, \ldots, R$, from the corresponding prior random distributions described by the probability density functions $\phi_k(\tilde{\beta}_k \mid \theta_k)$. The approximation of the D_b-error can be formalized as

$$E(f) \approx \frac{1}{R} \sum_{r=1}^{R} f(\tilde{\beta}^{(r)} \mid X). \tag{5}$$

The total number of D-error evaluations is equal to *R*. In order to determine the draws $\tilde{\beta}_k^{(r)}$, we let the computer generate for each parameter *R* pseudo-random numbers $u_k^{(r)}$ which are uniformly distributed on the interval [0,1], and then compute the draws by

$$\tilde{\beta}_k^{(r)} = \Phi_k^{-1}\left(u_k^{(r)}\right), \tag{6}$$

where $\Phi_k(\tilde{\beta}_k \mid \theta_k)$ denotes the cumulative distribution function corresponding to the probability density function $\phi_k(\tilde{\beta}_k \mid \theta_k)$.

3.2 Quasi-Random Monte Carlo Simulation

Randomness of the draws is not a prerequisite in the approximation of the integral in Equation (3); rather, it has been argued in the literature that (a) correlation between draws for different dimensions has a positive effect on the approximation, and (b) one should aim for the draws to be distributed as uniformly as possible over the area of integration. Hence, the draws can be selected deterministically so as to minimize the integration error. Quasi-random MC simulation methods for approximating the D_b-error are almost identical to the PMC simulation method, except that they use deterministic draws for $\tilde{\beta}_k^{(r)}$ (as opposed to purely random draws). In fact, Equations (5) and (6) are still valid, but instead of generating pseudo-random numbers $u_k^{(r)} \sim U(0,1)$, these numbers $u_k^{(r)}$ are taken from different intelligent quasi-random sequences, also called low discrepancy sequences. In this paper, we examine three different sequences. MLHS aims to distribute the draws uniformly, while maintaining randomness between different dimensions. Halton and Sobol sequences provide a certain degree of uniformity in the distribution of the draws, but also introduce correlations between the sequences in different dimensions. We now look at these three approaches in turn.

3.2.1 Modified Latin Hypercube Sampling (MLHS)

The MLHS method (Hess et al. 2005) produces multi-dimensional sequences by combining randomly shuffled versions of one-dimensional sequences made up of uniformly spaced points. Formally, the individual one-dimensional sequences of length R are constructed as:

$$u_k^{(r)} = \frac{r-1}{R} + \xi_k, \;\; r = 1, \ldots, R, \tag{7}$$

where ξ_k is a random number drawn between 0 and $1/R$, and where a different random draw is used in each of the K different dimensions. In the resulting sequence, the distances between adjacent draws are all equal to $1/R$, satisfying the condition of equal spacing. Multi-dimensional sequences are constructed by simple combination of randomly shuffled one-dimensional sequences, where the shuffling disrupts the correlation between individual dimensions.

3.2.2 Halton Sequences

Halton sequences (Halton 1960) are constructed according to a deterministic method based on the use of prime numbers, dividing the 0-1 space into p_k segments (with p_k giving the prime used as the base for parameter k), and by systematically filling in the empty spaces, using cycles of length p_k that place one draw in each segment. Formally, the r^{th} element in the Halton sequence based on prime p_k is obtained by taking the radical inverse of integer r in base p_k by reflection through the radical point, such that

$$r = \sum_{\ell=0}^{L} b_\ell^{(r)} p_k^\ell, \tag{8}$$

where $0 \le b_\ell^{(r)} \le p_k - 1$ determines the L digits used in base p_k in order to represent r (i.e., solving Equation (8)), and where the range for L is determined by $p_k^L \le r < p_k^{L+1}$. The draw is then obtained as:[5]

$$u_k^{(r)} = \sum_{\ell=0}^{L} b_\ell^{(r)} p_k^{-\ell-1}. \tag{9}$$

To allow for the computation of a simulation error, the deterministic Halton sequence can be randomized in several ways. Here, we use the approach discussed by amongst others Tuffin (1996), where the modified draws are obtained by adding a random draw ξ_k to the individual draws in dimension k, and by subtracting one from any draws that now fall outside the 0-1 interval. A different random draw is used for each dimension.

[5] As an example, consider the 5th draw using 2 (the first prime number) as base. Then $r = 5$ can be expressed using three digits as 101 in base 2, because $5 = 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2$. Using Equation (9) the 5th draw is then given by $1 \cdot 2^{-0-1} + 0 \cdot 2^{-1-1} + 1 \cdot 2^{-2-1} = 0.5 + 0 + 0.125 = 0.625$.

3.2.3 Sobol Sequences

The main problem with Halton sequences is the fact that the individual sequences are highly correlated, leading to problems with poor multi-dimensional coverage in higher dimensions. Aside from various transformations of the standard Halton sequence and other advanced methods (cf. Hess et al. 2005), one approach that has received exposure in the area of discrete choice modeling is the Sobol sequence, used amongst others by Garrido (2003). Like Halton sequences, Sobol sequences are based on Van der Corput sequences (cf. Niederreiter 1992). However, rather than in a *K*-dimensional problem using the first *K* primes (as in Halton sequences), Sobol sequences are based on prime 2 in each dimension, where different permutations are used to ensure that the resulting *K*-dimensional sequence obtains good coverage. We will use a randomized version of the Sobol sequences equivalent to the randomization in the Halton sequences by adding a random component to each of the draws in each dimension.

3.3 Polynomial cubature

Polynomial cubature methods aim to approximate integrals using orthogonal polynomials. Gaussian quadrature is the best-known method. In case of a single variable, the use of *R* draws yields an exact approximation if the integrand is a polynomial up to degree (2*R*-1). General functions can be approximated by (high order) polynomials, hence the higher the degree (yielding more draws), the more accurate the approximation will be.

The principle of Gaussian quadrature is that not only the draws $\tilde{\beta}_k^{(r)}$ for the priors are selected intelligently, but also that weights $w_k^{(r)}$ are associated with each draw. The approximation of the D_b-error using Gaussian quadrature can be formalized as

$$E(f) \approx \sum_{r_1=1}^{R_1} \cdots \sum_{r_K=1}^{R_K} w_1^{(r_1)} \cdots w_K^{(r_K)} f(\tilde{\beta}_1^{(r_1)}, \ldots, \tilde{\beta}_K^{(r_K)} \mid X). \tag{10}$$

The draws for the priors and the associated weights depend on the random distribution. Different draws for each individual parameter are called abscissas. In the case where $\tilde{\beta}_k \sim N(\mu_k, \sigma_k)$, the abscissas and weights can be computed using so-called Hermite polynomials. If $\tilde{\beta}_k \sim U(a_k, b_k)$, the abscissas and weights can be computed using so-called Legendre polynomials. The abscissas and weights for both situations are listed in Table 1 for up to 10 draws for each individual parameter. The weights always sum up to one, i.e., $\sum_{r=1}^{R} w_k^{(r)} = 1$ for each *k*. For each of the *K* parameters, the number of abscissas used, R_k, can be different.

Table 1: Abscissas and weights for Gauss-Hermite and Gauss-Legendre integration

	Normal distribution $N(\mu_k, \sigma_k)$ $\tilde{\beta}_k^{(r)} = \mu_k + x^{(r)}\sqrt{2}\sigma_k$		**Uniform distribution** $U(a_k, b_k)$ $\tilde{\beta}_k^{(r)} = \frac{1}{2}(a_k + b_k) + \frac{1}{2}(b_k - a_k)x^{(r)}$	
R_k	$x^{(r)}$	$w_k^{(r)}$	$x^{(r)}$	$w_k^{(r)}$
1	0.0000000000	1.0000000000	0.0000000000	1.0000000000
2	±0.7071067812	0.5000000000	±0.5773502692	0.5000000000
3	0.0000000000	0.6666666667	0.0000000000	0.4444444444
	±1.2247448714	0.1666666667	±0.7745966700	0.2777777778
4	±1.6506801239	0.0458758548	±0.3399810400	0.3260725750
	±0.5246476233	0.4541241452	±0.8611363100	0.1739274250
5	0.0000000000	0.5333333333	0.0000000000	0.2844444450
	±2.0201828705	0.0112574113	±0.5384693100	0.2393143350
	±0.9585724646	0.2220759220	±0.9061798500	0.1184634450
6	±2.3506049737	0.0025557844	±0.2386191800	0.2339569650
	±1.3358490740	0.0886157460	±0.6612093900	0.1803807850
	±0.4360774119	0.4088284696	±0.9324695100	0.0856622450
7	0.0000000000	0.4571428571	0.0000000000	0.2089795900
	±2.6519613568	0.0005482689	±0.4058451500	0.1909150250
	±1.6735516288	0.0307571240	±0.7415311900	0.1398526950
	±0.8162878829	0.2401231786	±0.9491079100	0.0647424850
8	±2.9306374203	0.0001126145	±0.1834346400	0.1813418900
	±1.9816567567	0.0096352201	±0.5255324100	0.1568533250
	±1.1571937125	0.1172399077	±0.7966664800	0.1111905150
	±0.3811869902	0.3730122577	±0.9602898600	0.0506142700
9	0.0000000000	0.4063492063	0.0000000000	0.1651196775
	±3.1909932018	0.0000223458	±0.3242534234	0.1561735385
	±2.2665805845	0.0027891413	±0.6133714327	0.1303053482
	±1.4685532892	0.0499164068	±0.8360311073	0.0903240803
	±0.7235510188	0.2440975029	±0.9681602395	0.0406371942
10	±3.4361591188	0.0000043107	±0.1488743400	0.1477621100
	±2.5327316742	0.0007580709	±0.4333953900	0.1346333600
	±1.7566836493	0.0191115805	±0.6794095700	0.1095431800
	±1.0366108298	0.1354837030	±0.8650633700	0.0747256750
	±0.3429013272	0.3446423349	±0.9739065300	0.0333356700

Note that the total number of D-error evaluations is equal to $R = \prod_{k=1}^{K} R_k$, that is, the total number of all combinations of abscissas in all dimensions. This number of D-error evaluations grows exponentially if the number of random priors increases.[6] Therefore, Gaussian quadrature is typically not suitable for integrals of high dimensionality, although it is extremely powerful for low-dimensional problems.

[6] The minimum number of draws is typically two, such that with 10 random parameters, the minimum number of draws possible using Gaussian quadrature is 2^{10} = 1,024. Using three draws per random parameter increases this number to 3^{10} = 59,049.

4. Case studies

4.1 Model and experimental design description

We consider six different discrete choice models with the number of parameters ranging from two to 14 (cf. Table 2). They are all of the multinomial logit (MNL) type, although a similar analysis could be performed for nested logit (NL) and mixed logit (ML) by replacing the AVC matrix, see Appendix A. The constants in the model are assumed to have fixed priors (the constants are essentially design parameters in a stated choice experiment), where the uncertainty about the other parameters translates into random prior parameter values. In our case studies, each prior parameter $\tilde{\beta}_k$ is assumed to be normally distributed with a mean μ_k and a standard deviation σ_k,

$$\tilde{\beta}_k \sim N(\mu_k, \sigma_k). \tag{11}$$

Table 2: Model specifications

Model	*Utility functions*	*Comments*
M1	$U_1 = \beta_1 x_{11} + \beta_2 x_{12}$ $U_2 = \beta_1 x_{21} + \beta_2 x_{22}$	0 constants 2 generic par. 0 alt.-spec. par.
M2	$U_1 = \beta_{01} + \beta_1 x_{11} + \beta_2 x_{12} + \beta_3 x_{13}$ $U_2 = \beta_1 x_{21} + \beta_2 x_{22} + \beta_4 x_{23}$	1 constant 2 generic par. 2 alt.-spec. par.
M3	$U_1 = \beta_{01} + \beta_1 x_{11} + \beta_2 x_{12} + \beta_3 x_{13}$ $U_2 = \beta_{02} + \beta_1 x_{21} + \beta_2 x_{22} + \beta_4 x_{23}$ $U_3 = \beta_1 x_{31} + \beta_2 x_{32} + \beta_5 x_{33} + \beta_6 x_{34}$	2 constants 2 generic par. 4 alt.-spec. par.
M4	$U_1 = \beta_{01} + \beta_1 x_{11} + \beta_2 x_{12} + \beta_3 x_{13} + \beta_4 x_{14}$ $U_2 = \beta_{02} + \beta_1 x_{21} + \beta_2 x_{22} + \beta_5 x_{23} + \beta_6 x_{24}$ $U_3 = \beta_1 x_{31} + \beta_2 x_{32} + \beta_7 x_{33} + \beta_8 x_{34}$	2 constants 2 generic par. 6 alt.-spec. par.
M5	$U_1 = \beta_{01} + \beta_1 x_{11} + \beta_2 x_{12} + \beta_3 x_{13} + \beta_4 x_{14} + \beta_5 x_{15}$ $U_2 = \beta_{02} + \beta_1 x_{21} + \beta_2 x_{22} + \beta_6 x_{23} + \beta_7 x_{24} + \beta_8 x_{25}$ $U_3 = \beta_1 x_{31} + \beta_2 x_{32} + \beta_9 x_{33} + \beta_{10} x_{34}$	2 constants 2 generic par. 8 alt.-spec. par. (1 dummy)
M6	$U_1 = \beta_{01} + \beta_1 x_{11} + \beta_2 x_{12} + \beta_3 x_{13} + \beta_4 x_{14} + \beta_5 x_{15} + \beta_6 x_{16}$ $U_2 = \beta_{02} + \beta_1 x_{21} + \beta_2 x_{22} + \beta_3 x_{23} + \beta_7 x_{24} + \beta_8 x_{25} + \beta_9 x_{26} + \beta_{10} x_{27}$ $U_3 = \beta_1 x_{31} + \beta_2 x_{32} + \beta_3 x_{33} + \beta_{11} x_{34} + \beta_{12} x_{35}$	2 constants 3 generic par. 9 alt.-spec. par. (5 dummies)

The means μ_k are listed in Table 3 while the standard deviations are taken as a function of the mean,

$$\sigma_k = \alpha |\mu_k|, \qquad \alpha \geq 0. \tag{12}$$

We will consider two situations, namely a small uncertainty about the priors using $\alpha = 0.1$ and a large uncertainty about the priors using $\alpha = 0.3$.

Table 3: Prior parameter mean values

model	β_{01}	β_{02}	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}	β_{11}	β_{12}
M1	-	-	-0.09	-0.3	-	-	-	-	-	-	-	-	-	-
M2	1.2	-	-0.09	-0.3	0.5	0.8	-	-	-	-	-	-	-	-
M3	3.0	1.4	-0.09	-0.3	0.5	0.9	0.3	0.7	-	-	-	-	-	-
M4	-1.2	0.8	-0.09	-0.3	0.5	0.6	0.9	1.2	0.3	0.7	-	-	-	-
M5	-3.0	-1.5	-0.09	-0.3	0.5	0.9	0.6	0.3	0.8	1.2	0.3	0.8	-	-
M6	-3.3	1.0	-0.09	-0.06	-0.3	0.5	0.9	0.6	0.3	0.8	1.2	-0.3	0.3	0.8

The designs used for assessing the D_b-errors with different approximations are listed in Appendix B. In the next subsection the D_b-errors calculated using the different approximation methods will be compared for each model/design.

4.2 Comparison of approximation methods

For each design we calculate the D_b-error using the five different approximations: PMC, MLHS, Halton sequences, Sobol sequences, and Gauss-Hermite. For each design, all approximation outcomes are compared to the true value of the D_b-error, obtained by using a very large number of draws (all methods converged to the same true D_b-error in the limit). The D_b-errors are computed for different numbers of draws, from 20 draws up to 10,000–40,000 draws (depending on the model/design). The deviation from the true D_b-error is computed as a percentage.

Rather than computing a single percentage for the deviation from the true D_b-error for a given number of draws in each approximation, 50 deviations are computed by changing the draws 50 times randomly, which is trivial for PMC and MLHS and is described for Halton and Sobol in Section 3. Using these values, we determine the 95 percent confidence intervals for the D_b-errors. Since Gaussian quadrature is completely deterministic, these draws cannot be randomized; hence, there is no need to compute a confidence interval in this case. In Guassian quadrature, the number of draws cannot be chosen arbitrarily, as the number of draws should be a multiple of the number of abscissas used. We increase the number of draws each time by increasing the number of abscissas for a single parameter prior. Note that we do not require that each individual parameter prior has the same number of abscissas. Instead, we use different numbers of abscissas for each prior, depending on the impact this prior has on the utility. That is, if a prior has a large effect on the utility and has a large standard deviation, then we require more information on this prior in order to calculate the D_b-error more accurately. The priors are ranked in decreasing order of the mean value multiplied with the corresponding average attribute level. Starting with a single abscissa for each prior, the prior with the highest order will face an increase in the number of abscissas first, then the second in order, etc., until all priors have two abscissas each. Then the procedure starts all over again by increasing the number of abscissas for each prior in the same

order. The total number of draws used in the Gauss-Hermite approximation is equal to the product of all prior abscissas, as mentioned in Section 3.3.

Figure 1 depicts confidence intervals for the deviations from the true D_b-errors for the different approximation methods for design D3 using different numbers of draws *R* and large standard deviations of the priors ($\alpha = 0.3$). Note that this design corresponds to model M3 having six random priors (the constants are assumed to have fixed parameter priors). The figures for the other designs show very similar results and as such are not reproduced here. A summary of the confidence intervals for all designs with small and large standard errors are given in Table 4 and Table 5, respectively. From ***Figure 1***, we can conclude that PMC has the widest confidence interval (roughly -4 to +4 percent using 1,000 draws), while all other methods produce more narrow confidence intervals. Halton and Sobol perform quite well, while MLHS is mainly performing better than PMC for small *R* and but less well with higher *R*. The single line for the Gauss-Hermite approximation in the figure can be regarded as the 100 percent confidence interval. Clearly, the Gauss-Hermite approximation outperforms all other methods.

Table 4: Deviation from the true D_b-error (in %), $\alpha = 0.1$

		PMC		MLHS		Halton		Sobol			Gauss
design	*R*	Low	high	low	high	low	high	low	high	*R*	-
D1	*40*	-1.59	1.55	-0.44	0.43	-0.64	0.57	-0.57	0.56	*36*	0.00
	100	-1.04	0.94	-0.18	0.18	-0.29	0.24	-0.27	0.23	*100*	0.00
	200	-0.73	0.73	-0.12	0.11	-0.15	0.10	-0.17	0.12	*196*	0.00
	500	-0.39	0.39	-0.04	0.06	-0.07	0.04	-0.07	0.05	*506*	0.00
	1000	-0.26	0.31	-0.02	0.02	-0.04	0.03	-0.04	0.03	*992*	0.00
	2000	-0.18	0.22	-0.02	0.02	-0.02	0.01	-0.02	0.02	*1980*	0.00
D2	*40*	-1.50	1.59	-0.85	0.98	-0.79	0.83	-0.81	0.83	*36*	0.00
	100	-0.79	0.91	-0.40	0.48	-0.42	0.36	-0.44	0.41	*108*	0.00
	200	-0.54	0.67	-0.35	0.36	-0.25	0.23	-0.32	0.32	*192*	0.00
	500	-0.38	0.41	-0.22	0.23	-0.14	0.15	-0.16	0.15	*500*	0.00
	1000	-0.26	0.27	-0.15	0.13	-0.08	0.08	-0.08	0.06	*1080*	0.00
	2000	-0.16	0.17	-0.12	0.11	-0.04	0.04	-0.04	0.04	*2058*	0.00
	5000	-0.14	0.14	-0.06	0.08	-0.02	0.02	-0.02	0.03	*5184*	0.00
	10000	-0.08	0.10	-0.06	0.05	-0.02	0.02	-0.02	0.02	*10000*	0.00
D3	*40*	-1.67	1.59	-0.98	1.01	-1.05	1.24	-1.06	0.89	*32*	-1.86
	100	-1.06	0.93	-0.48	0.60	-0.56	0.57	-0.55	0.48	*96*	-0.01
	200	-0.68	0.69	-0.40	0.36	-0.32	0.33	-0.38	0.34	*216*	-0.01
	500	-0.43	0.41	-0.26	0.25	-0.17	0.18	-0.17	0.14	*486*	-0.01
	1000	-0.30	0.28	-0.18	0.19	-0.10	0.11	-0.11	0.10	*972*	0.00
	2000	-0.22	0.21	-0.16	0.13	-0.05	0.07	-0.07	0.07	*1728*	0.00
	5000	-0.14	0.11	-0.09	0.08	-0.03	0.03	-0.03	0.03	*5120*	0.00
	10000	-0.11	0.10	-0.05	0.07	-0.01	0.02	-0.02	0.02	*10000*	0.00
D4	*40*	-2.13	2.59	-1.57	1.77	-1.47	1.53	-1.95	1.46	*32*	-2.42
	100	-1.20	1.32	-0.83	0.80	-0.92	1.07	-0.91	0.77	*128*	-1.98
	200	-0.79	0.89	-0.76	0.71	-0.52	0.60	-0.52	0.46	*256*	0.00
	500	-0.64	0.62	-0.47	0.44	-0.24	0.31	-0.26	0.23	*576*	0.00
	1000	-0.41	0.36	-0.30	0.27	-0.14	0.20	-0.14	0.15	*864*	-0.01
	2000	-0.32	0.32	-0.21	0.20	-0.10	0.12	-0.12	0.11	*1944*	-0.01
	5000	-0.21	0.15	-0.14	0.12	-0.05	0.05	-0.05	0.05	*4374*	0.00
	10000	-0.16	0.12	-0.09	0.09	-0.03	0.03	-0.03	0.03	*11664*	0.00
	20000	-0.11	0.09	-0.04	0.06	-0.02	0.02	-0.02	0.02	*20736*	0.00
D5	*40*	-1.96	2.31	-0.77	0.63	-1.93	1.62	-1.29	1.14	*32*	-0.48
	100	-1.44	1.56	-0.46	0.44	-0.85	0.82	-0.56	0.45	*128*	-0.13
	200	-1.30	1.17	-0.27	0.25	-0.49	0.43	-0.31	0.27	*256*	-0.05

	500	-0.79	0.67	-0.13	0.12	-0.21	0.22	-0.15	0.09	*512*	-0.03
	1000	-0.59	0.46	-0.09	0.09	-0.11	0.12	-0.09	0.08	*1024*	-0.03
	2000	-0.39	0.38	-0.08	0.07	-0.07	0.08	-0.03	0.03	*2304*	0.01
	5000	-0.22	0.22	-0.03	0.04	-0.04	0.04	-0.02	0.02	*5184*	0.00
	10000	-0.20	0.16	-0.03	0.03	-0.02	0.02	-0.01	0.01	*11664*	0.00
	20000	-0.16	0.15	-0.02	0.02	-0.01	0.01	-0.01	0.01	*17496*	0.00
	40000	-0.11	0.10	-0.01	0.01	-0.01	0.01	-0.01	0.01	*39366*	0.00
D6	*40*	-1.15	1.11	-0.33	0.39	-0.94	0.90	-0.60	0.57	*32*	-0.29
	100	-0.68	0.82	-0.21	0.22	-0.51	0.44	-0.36	0.29	*128*	-0.18
	200	-0.58	0.64	-0.14	0.14	-0.35	0.27	-0.23	0.23	*256*	-0.14
	500	-0.31	0.32	-0.10	0.10	-0.19	0.17	-0.06	0.08	*512*	-0.02
	1000	-0.27	0.18	-0.07	0.06	-0.06	0.05	-0.05	0.06	*1024*	-0.01
	2000	-0.20	0.17	-0.04	0.04	-0.03	0.03	-0.05	0.06	*2048*	0.01
	5000	-0.13	0.09	-0.03	0.04	-0.02	0.01	-0.02	0.02	*4096*	0.00
	10000	-0.07	0.07	-0.02	0.02	-0.01	0.01	-0.01	0.01	*9216*	0.00
	20000	-0.04	0.05	-0.02	0.02	-0.01	0.01	-0.01	0.01	*20736*	0.00
	40000	-0.03	0.04	-0.01	0.01	-0.00	0.00	-0.00	0.01	*46656*	0.00

Table 5: Deviation from the true Db-error (in %), $\alpha = 0.3$

		PMC		MLHS		Halton		Sobol			Gauss
design	*R*	low	high	low	high	low	high	low	high	*R*	-
D1	*40*	-5.04	5.14	-1.16	1.55	-1.91	1.77	-1.92	1.91	*36*	0.00
	100	-3.43	3.22	-0.76	0.74	-0.88	0.73	-0.88	0.71	*100*	0.00
	200	-2.27	2.31	-0.44	0.45	-0.50	0.36	-0.55	0.43	*196*	0.00
	500	-1.21	1.22	-0.26	0.25	-0.23	0.15	-0.24	0.18	*506*	0.00
	1000	-0.86	1.02	-0.18	0.17	-0.14	0.10	-0.12	0.11	*992*	0.00
	2000	-0.58	0.73	-0.09	0.09	-0.06	0.06	-0.07	0.06	*1980*	0.00
D2	*40*	-16.23	17.39	-9.64	9.93	-11.89	12.92	-9.96	9.95	*36*	-0.33
	100	-7.94	9.75	-5.97	6.36	-5.83	6.00	-5.43	4.59	*108*	-0.04
	200	-5.27	6.96	-4.54	4.51	-3.57	3.21	-3.98	3.72	*192*	-0.02
	500	-4.04	4.39	-2.62	2.56	-2.27	2.34	-2.35	1.85	*500*	-0.00
	1000	-3.04	3.16	-2.12	2.27	-1.43	1.20	-1.25	0.89	*1080*	-0.00
	2000	-2.00	1.96	-1.37	1.25	-0.81	0.75	-0.76	0.75	*2058*	-0.00
	5000	-1.44	1.52	-0.91	0.78	-0.45	0.51	-0.48	0.47	*5184*	-0.00
	10000	-0.92	1.12	-0.58	0.66	-0.31	0.27	-0.35	0.31	*10000*	-0.00
D3	*40*	-11.22	10.72	-7.62	6.75	-7.84	8.81	-7.85	6.33	*32*	-12.21
	100	-7.23	6.13	-4.64	5.66	-4.39	4.56	-4.39	3.74	*96*	-0.21
	200	-5.00	4.83	-3.37	3.56	-2.83	3.01	-3.04	2.55	*216*	-0.19
	500	-3.18	3.00	-2.23	2.01	-1.68	1.77	-1.52	1.07	*486*	-0.14
	1000	-2.10	1.79	-1.37	1.40	-0.92	0.99	-1.02	0.75	*972*	-0.14
	2000	-1.58	1.36	-0.89	0.80	-0.51	0.60	-0.65	0.55	*1728*	-0.14
	5000	-1.00	0.69	-0.66	0.53	-0.33	0.31	-0.35	0.26	*5120*	-0.02
	10000	-0.82	0.67	-0.46	0.36	-0.18	0.16	-0.25	0.19	*10000*	-0.02
D4	*40*	-19.41	24.28	-16.32	16.94	-19.81	19.28	-18.87	13.95	*32*	-17.26
	100	-11.35	13.30	-10.50	10.68	-12.82	15.46	-10.37	8.82	*128*	-15.32
	200	-9.02	10.37	-7.67	5.97	-7.23	8.95	-6.41	5.12	*256*	-2.11
	500	-6.62	6.16	-5.10	5.30	-3.67	4.21	-4.00	3.24	*576*	-0.72
	1000	-4.49	3.62	-3.22	2.95	-2.46	2.95	-2.37	2.40	*864*	-0.65
	2000	-3.27	3.10	-2.49	1.79	-1.67	1.68	-2.02	1.77	*1944*	-0.63
	5000	-2.07	1.44	-1.22	1.20	-0.93	0.83	-0.87	0.67	*4374*	-0.60
	10000	-1.44	1.01	-1.19	0.99	-0.59	0.46	-0.48	0.43	*11664*	-0.06
	20000	-1.20	0.95	-0.68	0.66	-0.46	0.25	-0.42	0.27	*20736*	-0.06
D5	*40*	-11.90	11.36	-5.13	4.84	-10.36	9.07	-7.66	7.00	*32*	-2.37
	100	-9.07	8.92	-3.54	3.54	-4.14	4.22	-3.69	2.96	*128*	-1.93
	200	-6.67	5.64	-2.78	2.50	-2.68	2.18	-2.04	1.64	*256*	-1.74
	500	-4.11	3.42	-1.46	1.37	-1.51	1.69	-1.29	0.92	*512*	-1.70
	1000	-2.89	2.40	-1.08	1.03	-0.77	0.95	-1.11	0.90	*1024*	-1.64
	2000	-1.88	1.75	-0.86	0.59	-0.59	0.67	-0.42	0.33	*2304*	-0.07

	5000	-1.20	1.02	-0.45	0.40	-0.31	0.31	-0.32	0.22	*5184*	-0.02
	10000	-1.02	0.83	-0.36	0.31	-0.19	0.18	-0.20	0.16	*11664*	-0.01
	20000	-0.78	0.73	-0.23	0.24	-0.13	0.09	-0.16	0.10	*17496*	-0.01
	40000	-0.53	0.48	-0.12	0.13	-0.08	0.05	-0.12	0.07	*39366*	-0.01
D6	*40*	-7.10	7.16	-3.30	4.27	-4.97	6.24	-4.35	4.54	*32*	-2.07
	100	-3.65	4.62	-1.85	2.27	-2.82	3.19	-2.94	3.17	*128*	-1.33
	200	-3.11	3.90	-1.19	2.08	-1.65	1.99	-1.92	2.65	*256*	-0.96
	500	-1.85	2.51	-0.51	0.81	-0.75	1.25	-0.45	1.09	*512*	-0.04
	1000	-1.63	1.60	-0.44	0.82	-0.26	0.81	-0.26	0.88	*1024*	-0.02
	2000	-1.07	1.42	-0.22	0.64	0.06	0.45	-0.18	0.81	*2048*	-0.15
	5000	-0.58	0.83	0.07	0.45	0.11	0.39	0.07	0.46	*4096*	-0.16
	10000	-0.24	0.74	0.12	0.48	0.15	0.35	0.15	0.38	*9216*	-0.02
	20000	-0.06	0.58	0.13	0.36	0.20	0.33	0.19	0.32	*20736*	-0.00
	40000	0.01	0.55	0.17	0.36	0.23	0.30	0.21	0.30	*46656*	-0.00

In Table 4 and Table 5, the 95 percent confidence intervals (from low to high) of the deviations are indicated for PMC, MLHS, Halton, and Sobol for different numbers of draws *R*, while the deterministic deviations from Gauss-Hermite is indicated in the last column. As the number of draws for Gauss-Hermite in general does not match the number of draws from the other methods, the value of *R* that is as close as possible to the number of draws for the other methods are shown.

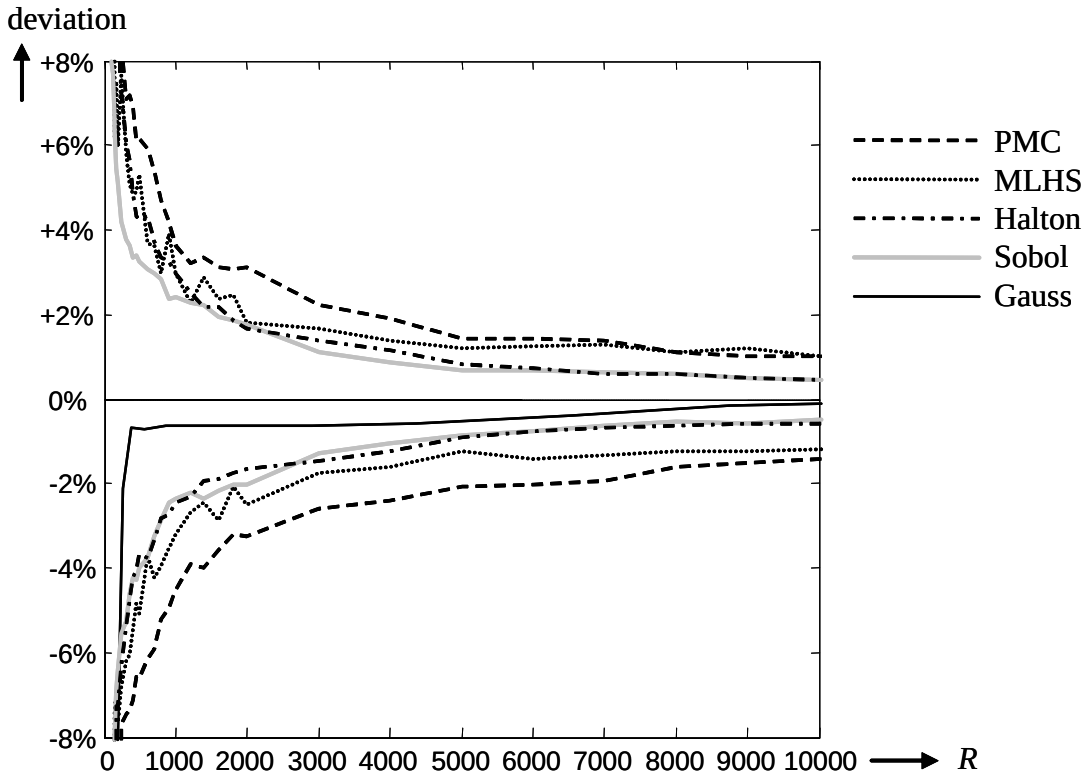

Figure 1: Confidence intervals for deviation of the true D_b-error of design D3 $(\alpha = 0.3)$

Comparing Table 4 and Table 5, larger standard deviations result in greater difficulty in approximating the D_b-error than smaller standard deviations. Only a few draws are needed in order to have the 95 percent confidence interval of the deviation within ±1 percent. Even though PMC is outperformed by all other methods (particularly by Gauss-

Hermite) in the case of small standard deviations, it can be concluded that the approximation method is of particular importance in cases where the standard deviations are larger, making it more difficult to compute the D_b-error. As such, we will focus mainly on the results dealing with large standard deviations in the priors.

As expected, Gauss-Hermite is preferred in designs with lower dimensions (designs D1 and D2 have 2 and 4 random priors, respectively) as indicated in Table 5, where even with low numbers of draws, the true D_b-error is accurately reproduced. In designs with higher numbers of dimensions (design D3–D6 have 6, 8, 10, and 12 random priors, respectively) Gauss-Hermite has slightly more problems with computing the D_b-error, but still performs well. This result is somewhat surprising as some researchers have found in the past that Gauss-Hermite typically only works well with very low dimensions (1, 2, or 3) due to the exponentially growing number of draws needed (Bhat 2001). However, in this paper we use a smarter approach (which we will term incremental Gaussian quadrature) in which not all priors are given the same number of abscissas, which significantly reduces the number of draws needed for higher-dimensional problems. Overall, our findings suggest that Gauss-Hermite outperforms the other methods considered within this paper. However, it is worth noting that, mainly with designs D3 and D4, Gauss-Hermite has difficulty when a small numbers of draws (R<100) is used. This is due to the fact that with a small number of draws, the Gauss-Hermite method is unable to pick up enough variance in the prior parameters. Nevertheless, it performs well with more draws.

Halton and Sobol perform similarly, clearly outperforming PMC. It is interesting to note that in all designs, MLHS performs well compared to PMC and similar to Halton and Sobol when low numbers of draws are used. However, whenever the number of draws increases, the D_b-error from MLHS does not converge as rapidly to the true value as Halton and Sobol do. This may be explained by the way the sequences are constructed. PMC lacks both a uniform spread of the integration area and correlation between the draws in different dimensions, which were argued to have a positive effect on the accuracy of the approximation (see Section 3). MLHS has a uniform spread by definition, but correlation is removed as much as possible by randomizing the order of the draws in each dimension. Halton and Sobol are less uniform in their spread (particularly with low number of draws), but the correlation between the dimensions has positive effect on the outcomes (at least for smaller dimensions). With small R, MLHS produces more uniform sequences than Halton or Sobol. However, with larger R this uniformity plays less of a role and the importance of the correlation between the sequences may become more important.

Rather than looking at the percentage deviation from the true D_b-error for different numbers of draws, we can consider the reverse by looking at the number of draws needed in order to ensure (with 95 percent certainty; and 100 percent certainty in case of Gauss-Hermite) that the deviation is not more than a certain percentage. The results are shown in Table 6, where the numbers of draws have been determined by inverting the lines in

Figure 1 (and using linear interpolation).

Table 6: Number of draws for different allowed maximum deviations

Design 1, $\alpha = 0.1$						Design 1, $\alpha = 0.3$				
Dev.	PMC	MLHS	Halton	Sobol	Gauss	PMC	MLHS	Halton	Sobol	Gauss
5.0%	<20	<20	<20	<20	1	45	<20	<20	<20	1
4.0%	<20	<20	<20	<20	1	73	<20	<20	<20	2
3.0%	<20	<20	<20	<20	1	126	<20	27	27	4
2.0%	26	<20	<20	<20	1	242	29	39	39	4
1.0%	111	<20	26	22	1	1,033	74	78	85	4
0.5%	352	37	52	51	2	>2,000	139	204	252	4
Design 2, $\alpha = 0.1$						Design 2, $\alpha = 0.3$				
Dev.	PMC	MLHS	Halton	Sobol	Gauss	PMC	MLHS	Halton	Sobol	Gauss
5.0%	<20	<20	<20	<20	1	294	116	115	110	16
4.0%	<20	<20	<20	<20	1	577	293	150	199	16
3.0%	<20	<20	<20	<20	2	1,115	348	288	327	24
2.0%	21	<20	<20	<20	2	2,016	1,400	645	629	24
1.0%	89	39	33	32	4	>10,000	4,543	1,592	1,389	24
0.5%	293	85	88	83	8	>10,000	>10,000	3,947	4,810	36
Design 3, $\alpha = 0.1$						Design 3, $\alpha = 0.3$				
Dev.	PMC	MLHS	Halton	Sobol	Gauss	PMC	MLHS	Halton	Sobol	Gauss
5.0%	<20	<20	<20	<20	2	200	112	76	86	64
4.0%	<20	<20	<20	<20	2	297	161	121	115	64
3.0%	<20	<20	<20	<20	4	539	275	158	204	64
2.0%	31	<20	<20	<20	32	1,415	647	420	351	64
1.0%	106	64	56	44	64	4,987	1,943	997	1,247	64
0.5%	367	154	119	111	64	>10,000	8,956	2,594	2,825	64
Design 4, $\alpha = 0.1$						Design 4, $\alpha = 0.3$				
Dev.	PMC	MLHS	Halton	Sobol	Gauss	PMC	MLHS	Halton	Sobol	Gauss
5.0%	<20	<20	<20	<20	4	871	441	390	320	256
4.0%	<20	<20	<20	<20	4	1,171	581	657	499	256
3.0%	34	<20	<20	<20	8	2,398	1,087	984	769	256
2.0%	65	32	<20	27	128	6,342	2,676	1,703	2,034	384
1.0%	179	85	106	89	256	>20,000	11,525	4,749	4,278	384
0.5%	691	270	243	210	256	>20,000	>20,000	14,343	9,758	6,561
Design 5, $\alpha = 0.1$						Design 5, $\alpha = 0.3$				
Dev.	PMC	MLHS	Halton	Sobol	Gauss	PMC	MLHS	Halton	Sobol	Gauss
5.0%	<20	<20	<20	<20	1	300	48	86	79	16
4.0%	<20	<20	<20	<20	1	540	75	107	95	16
3.0%	24	<20	<20	<20	16	970	144	186	138	16
2.0%	59	<20	37	<20	32	1,696	233	419	220	128
1.0%	278	35	83	57	32	8,634	1,096	954	1,076	1,536
0.5%	1,392	88	193	112	32	>40,000	4,604	2,820	1,624	1,536
Design 6, $\alpha = 0.1$						Design 6, $\alpha = 0.3$				
Dev.	PMC	MLHS	Halton	Sobol	Gauss	PMC	MLHS	Halton	Sobol	Gauss
5.0%	<20	<20	<20	<20	1	87	36	67	33	8
4.0%	<20	<20	<20	<20	1	187	45	85	55	8
3.0%	<20	<20	<20	<20	1	401	91	105	114	8
2.0%	<20	<20	26	<20	2	656	145	139	290	64
1.0%	57	<20	39	<20	2	4,192	481	768	860	256
0.5%	294	34	102	60	8	>40,000	4,452	1,888	4,129	512

With small standard deviations ($\alpha = 0.1$), the number of draws required to be within one percent from the true D_b-error is typically not larger than 100 for all designs.

However, in the case where the prior parameter distributions are assumed to have large standard deviations ($\alpha = 0.3$), a much larger number of draws is necessary. Compare the outcomes for design D1 using large standard deviations. If one would like to be with 95 percent probability within 0.5 percent from the true D_b-error, more than 2,000 PMC draws are needed, while MLHS, Halton, and Sobol only require 139, 204, and 252, respectively. Moreover, Gauss-Hermite approximation needs only four draws (two abscissas per prior parameter) to be within that 0.5 percent. This pattern repeats itself for other designs. For example, for design D2, PMC and MLHS need more than 10,000 draws to be within the 0.5 percent range, Halton and Sobol require almost 4,000 and 5,000, respectively, while Gauss-Hermite requires only 36 draws. As expected, in higher dimensions, Gauss-Hermite requires significantly more draws, but still less than when using the other methods. For example, in design D5, PMC needs more than 40,000 draws, MLHS, Halton, and Sobol need approx. 4,600, 2,800, and 1,600, respectively, whilst Gauss-Hermite requires approx. 1,500 draws. Nevertheless, it is to be expected that Gauss-Hermite approximation in larger models with more than 10 random prior parameters (as in model M6) will need significantly more draws, which may become prohibitive. It should be pointed out however, that the other methods may require more draws as well, meaning that choosing for Halton or Sobol may not necessarily provide better results than Gauss-Hermite with the same number of draws. Nonetheless, the number of draws in Gauss-Hermite approximation is dictated by the product of the prior abscissas and choosing a small value may therefore be impossible in a large design. In that case, there is always the option of using Halton or Sobol, as the number of draws can be selected arbitrarily, although one should realize that the approximated D_b-error may deviate largely from the true value.

5. Conclusions and discussion

This paper compares the performance of PMC draws to several types of quasi random Monte Carlo draws, as well as to a single polynomial cubature method, when using Bayesian methods to generate efficient SC designs. The quasi random Monte Carlo draws include Halton, Sobol and MLHS draws whilst the polynomial cubature method examined is Gauss-Hermite. Performance comparisons are made for six SC designs with various design dimensions (attributes and alternatives) as well as over different assumptions regarding the standard deviations of the prior parameter distributions. In all but a few cases involving an extremely small number of draws, Gauss-Hermite approximation appears to outperform all other methods in reproducing the true level of a design's level of efficiency, whilst the PMC method appears to perform worst in nearly all cases. When the standard deviations of the prior parameter distributions are relatively small (i.e., the researcher is more certain about the true parameter value), Halton, Sobol and MLHS draws appear to perform equally well. However, with larger standard deviations in the prior parameter distributions and with higher numbers of design dimensions, the performance of MLHS is slightly worse than both Halton and Sobol sequences. Furthermore, all approximation methods need more draws if the standard deviations of the priors are larger.

Our findings call into question the predominant use within the literature of PMC draws to generate Bayesian efficient SC designs. Our findings suggest that designs generated using PMC methods are unlikely to be truly efficient under the assumptions made by the researcher (that is the population moments of the prior parameter distributions)

unless an impractically large number of draws are used. The results of this paper suggest that whilst quasi random Monte Carlo methods perform much better than the PMC method, better approximation to the true level of efficiency of a design may be achieved using polynomial cubature methods. This result conflicts with evidence offered in other areas using simulation methods (mainly in estimating the random parameters in mixed logit models) which suggest that Halton draws outperform Gauss-Hermite approximations (Bhat 2001) in obtaining more correct results. In this paper, we have used an incremental Gauss-Hermite approximation, which is a more intelligent technique than used elsewhere in determining how many draws to use, which may partly explain these conflicting results.

Of course, as we have noted on several occasions, as the number of dimensions requiring simulation increases, so does the number of draws required when using Gauss-Hermite approximations. Unlike with other methods, this cannot be avoided. That is, whilst the researcher can determine the number of draws to employ when using PMC or quasi random Monte Carlo methods, thus accepting a lower level of accuracy in return for lower computational cost, the number of draws required when using polynomial cubature methods is determined by the abscissas for a given design. Whilst the number of draws may be reduced using methods such as that used here, there still remains a minimum number of draws that must be used when employing polynomial cubature methods. No such limits exist for the other methods. It appears however, that for a given level of accuracy involving designs with large numbers of dimensions, the number of draws required when using polynomial cubature methods represents the minimum number of draws, independent of the type of draws taken. As such, whilst the researcher may rely on less draws when using say Halton draws, the reduction in the number of draws comes at the price of less accurate results.

One limitation within the research presented here is that we have only examined the case of Bayesian efficient designs assuming the multinomial logit model form. The theory presented in this paper is still valid for other discrete choice models (such as nested logit and mixed logit). Whilst we would expect the results to hold for these other models, this is still to be confirmed. An interesting case to examine is the mixed logit model, whereby simulation is required not only for the Bayesian prior distributions, but also the random parameter distributions as well. Sándor and Wedel (2002, 2005) do report results for Bayesian efficient designs developed using mixed logit models, adopting a quasi random Monte Carlo approach; orthogonal array-based Latin hypercube sampling, and randomly shifted good lattice points, respectively. They report in a footnote (Sándor and Wedel 2005) that some exploration of the number of draws was undertaken but we call for a more structured examination of the issue, similar to that presented here.

Additionally, the analyses presented only consider Normal distributions for the prior parameters. Additional research is required to investigate the impact of the different approximation methods when other probability distributions are assumed.

Acknowledgments

We would like to thank Andrew Collins for his work on the Ngene software, and Fabian Bastin for his help in creating the Sobol sequences.

References

Bastin, Fabian, Cinzia Cirillo and Philippe L. Toint (2004), "Estimating mixed logit models with quasi-Monte Carlo sequences allowing practical error estimation," Proceedings European Transport Conference, Strasbourg, France.

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

----- (2003), "Simulation estimation of mixed discrete choice models using randomized and scrambled Halton sequences," *Transportation Research B*, 37(9), 837-855.

Bliemer, Michiel C.J., John M. Rose, and David A. Hensher (2005), "Constructing efficient stated choice experiments allowing for differences in error variances across subsets of alternatives," Submitted for publication to *Transportation Research B*.

Burgess, Leonie and Deborah J. Street (2003), "Optimal Designs for 2k Choice Experiments," *Communications in Statistics, Theory and Methods*, 32(11), 2185–2206.

Carlsson, Fredrik and Peter Martinsson (2002), "Design Techniques for Stated Preference Methods in Health Economics," *Health Economics*, 12 (June), 281-294.

Ferrini, Silvia and Riccardo Scarpa (2006), "Designs with a-priori information for nonmarket valuation with choice-experiments: a Monte Carlo study," Submitted to *Journal of Environmental Economics and Management*.

Garrido, Rodrigo A. (2003), "Estimation performance of low discrepancy sequences in stated preferences," paper presented at the 10th International Conference on Travel Behaviour Research, Lucerne, Switzerland.

Halton, John (1960), "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numerische Mathematik*, 2, 84-90.

Hensher, David A. and William H. Greene (2002), "Specification and estimation of the nested logit model: alternative normalisations," *Transportation Research B*, 36 (1), 1-17.

Hess, Stephane, Kenneth E. Train and John W. Polak (2005), "On the use of a Modified Latin Hypercube Sampling (MLHS) approach in the estimation of a Mixed Logit model for vehicle choice," *Transportation Research B*, 40(2), 147-163.

Huber, Joel and Klaus Zwerina (1996), "The Importance of Utility Balance in Efficient Choice Designs," *Journal of Marketing Research*, 33 (August), 307-317.

Joe, Stephen and Frances Y. Kuo (2003), "Remark on algorithm 659: Implementing Sobol's quasirandom sequence generator," *ACM Transactions on Mathematical Software*, 29(1), 49-57.

Kanninen, Barbara J. (2002), "Optimal Design for Multinomial Choice Experiments," *Journal of Marketing Research*, 39 (May), 214-217.

Kessels, Roselinde, Peter Goos, and Martina Vandebroek (2006), "A comparison of criteria to design efficient choice experiments," Forthcoming in *Journal of Marketing Research*.

Kuhfeld, Warren F., Randall D. Tobias and Mark Garratt (1994), "Efficient Experimental Design with Marketing Research Applications," *Journal of Marketing Research*, 21 (November), 545-557.

Lazari, Andreas G. and Don A. Anderson (1994), "Designs of Discrete Choice Experiments for Estimating Both Attribute and Availability Cross Effects," *Journal of Marketing Research*, 31(3), 375-383.

Louviere, Jordan J., David A. Hensher and Joffre D. Swait (2000), *Stated Choice Methods—Analysis and Application*. Cambridge, UK: Cambridge University Press.

McFadden, Dan (1974), "Conditional logit analysis of qualitative choice behaviour," in *Frontiers of econometrics*, Zarembka, Paul, ed., Academic Press, New York, 105-142.

Niederreiter, Harald (1992), Random number generation and quasi-Monte Carlo methods. CBMS-NFS Regional Conference Series in Applied Mathematics, 63, SIAM, Philadelphia, PA.

Rose, John M. and Michiel C.J. Bliemer (2005), "Constructing efficient choice experiments," Report ITLS-WP-05-07, Institute of Transport and Logistics Studies, University of Sydney.

Sándor, Zsolt and Michel Wedel (2001), "Designing Conjoint Choice Experiments Using Managers' Prior Beliefs," *Journal of Marketing Research*, 38 (November), 430-444.

---- and ---- (2002), "Profile Construction in Experimental Choice Designs for Mixed Logit Models," *Marketing Science*, 21(4), 455-475.

---- and ---- (2005), "Heterogeneous conjoint choice designs," *Journal of Marketing Research*," 42 (May), 210-218.

---- and Kenneth Train (2004), Quasi-random simulation of discrete choice models. *Transportation Research B*, 38 (November), 313-327.

Street, Deborah J., David S. Bunch, and Beverly J. Moore (2001), "Optimal Designs for 2^k Paired Comparison Experiments," *Communications in Statistics, Theory and Methods*, 30(10), 2149–71.

---- and Leonie Burgess (2004), "Optimal and Near Optimal Pairs for the Estimation of Effects in 2-Level Choice Experiments," *Journal of Statistical Planning and Inference*, 118(1–2), 185–99.

Train, Kenneth (2003), *Discrete Choice Methods with Simulation*, Cambridge University Press: Cambridge.

Tuffin, Bruno (1996), "On the use of low-discrepancy sequences in Monte Carlo methods," *Monte Carlo Methods and Applications*, 2(4), 295-320.

Appendix A. Deriving the asymptotic (co)variance matrix

Consider an experimental design with alternatives (indexed by *j*) with associated attributes (indexed by k). In each choice situation s, we assume that the levels of the attributes are given by x_{jks}. Let the utility of alternative *j* in choice situation *s* be given by

$$U_{js} = V_{js} + \varepsilon_{js}, \tag{13}$$

where V_{js} denotes the observed utility and ε_{js} denotes the unobserved utility.

Denote the complete experimental design by $X \equiv [x_{jks}]$. Let the observed utility of alternative *j* in choice situation *s* be given by

$$V_{js}(X \mid b) = \sum_{k \in K_j} b_k x_{jks}, \tag{14}$$

where $b \equiv [b_k]$ denotes the vector of attribute weights, which are typically the unknown parameters to be estimated. Depending on the set of attributes appearing in each alternative K_j, both generic and alternative-specific weights can be present. In the generic case, parameter b_k appears in multiple utility functions of different alternatives, while in the alternative-specific case, the parameter only appears in the corresponding alternative.

Let $P_{js}(X \mid b)$ denote the probability of choosing alternative *j* in choice situation *s*, and let y_{js} denote the outcome of the stated choice experiment based on the experimental design (assuming a single respondent), where y_{js} equals one if alternative *j* is chosen in choice situation *s*, and zero otherwise. The log-likelihood function can be written as

$$L(b \mid X) = \sum_s \sum_j y_{js} \log\left[P_{js}(X \mid b) \right]. \tag{15}$$

Assuming that β are the true parameter values, the Fisher information matrix can be written as

$$I(\beta \mid X) = \frac{\partial^2 L(\beta \mid X)}{\partial b \partial b'}. \tag{16}$$

The asymptotic variance-covariance (AVC) matrix can be computed as the negative inverse of the Fisher information matrix:

$$\Omega(\beta \mid X) = -I^{-1}(\beta \mid X) \tag{17}$$

The probability $P_{js}(X \mid b)$ depends on the assumptions regarding the unobserved components ε_{js}. Different assumptions yield different models. We will discuss three model types:

(a) *Multinomial logit (MNL) model* – Assumes that all ε_{js} are distributed identically and independently following a type I extreme value distribution. The estimated taste parameters are assumed to be nonrandom;
(b) *Nested logit (NL) model* – Assumes that the ε_{js} follow a joint extreme value distribution, such that the errors are still distributed identically, but no longer independently. The estimated taste parameters are still assumed to be nonrandom;
(c) *Mixed logit (ML) model* – Allows for additional errors on top of the type I extreme value errors, where no independence or identicality assumptions apply.

In the following, we will show the formulae for the probabilities in each of these models. Only these probabilities are different, the theory in the paper therefore holds for each of these models.

A.1 Multinomial logit (MNL) model

In the MNL model, we assume that the parameters *b* are nonrandom. Assuming prior parameter values $\tilde{\beta}$ for these parameters yields the following probability of choosing alternative *j* in choice situation *s* (see McFadden, 1974):

$$P_{js}(X \mid \tilde{\beta}) = \frac{\exp\left(V_{js}(X \mid \tilde{\beta})\right)}{\sum_i \exp\left(V_{is}(X \mid \tilde{\beta})\right)}. \tag{18}$$

The resulting Fisher information matrix and the AVC matrix can be computed analytically. See McFadden (1974) for the Fisher information matrix in case all parameters are generic, and see Rose and Bliemer (2005) in case the parameters are alternative-specific or there is a mix of generic and alternative-specific parameters.

A.2 Nested logit (NL) model

In the NL model, the error terms for some of the alternatives are correlated, leading to heightened substitution between these alternatives (cf. Train, 2003). The situation can be represented most easily with the help of a tree, where the choice of an alternative is preceded by the choice of a nest that this alternative belongs to (with alternatives that have correlated errors sharing a nest). It is important to stress that this is an econometric notion, and not a behavioral one; the model does not imply a sequential choice process by the respondent.

The simplest example is given by a two-level case, with composite nests at the upper level, and elementary alternatives at the lower level. Assume that the scales in the NL model are normalized at the lower level (corresponding to the RU1 model specification, see Hensher and Green, 2002) and that the scales of each nest *m* is given by λ_m. Assume that model parameters *b* are nonrandom and that the prior parameter values for

these parameters and the scale parameters are given by $\tilde{\beta}$ and $\tilde{\lambda}$, respectively. Then the probability of choosing alternative *j* that belongs to nest *m* is given by (see Bliemer et al., 2005):

$$P_{js}(X \mid \tilde{\beta}, \tilde{\lambda}) = \frac{\left(\sum_{i \in J_m} \exp\left(V_{is}(X \mid \tilde{\beta})\right)\right)^{\tilde{\lambda}_m}}{\sum_n \left(\sum_{i \in J_n} \exp\left(V_{is}(X \mid \tilde{\beta})\right)\right)^{\tilde{\lambda}_n}} \frac{\exp\left(V_{js}(X \mid \tilde{\beta})\right)}{\sum_i \exp\left(V_{is}(X \mid \tilde{\beta})\right)}. \tag{19}$$

The Fisher information and AVC matrix can be computed analytically for this two-level NL model as shown in Bliemer et al. (2005).

A.3 Mixed logit (ML) model

Now instead of assuming fixed parameters *b* we assume that the parameters follow a random distribution with distribution parameters β (e.g., explaining the heterogeneity of agents in the discrete choice model). Suppose that the probability density function of *b* is given by $\psi(b \mid \beta)$. Assuming prior values $\tilde{\beta}$ of the distribution parameters, the (expected) probability of choosing alternative *j* in choice situation *s* is

$$P_{js}(X \mid \tilde{\beta}) = \int_b \frac{\exp\left(V_{js}(X \mid b)\right)}{\sum_i \exp V_{is}(X \mid b)} \psi(b \mid \tilde{\beta}) db. \tag{20}$$

Note that computing this probability requires the evaluation of an integral, similar to Equation (3). An analytical solution to solving this integral does not exist. The same approximation methods discussed in this paper can be applied for solving Equation (20). Comparisons of different approximation methods (pseudo-random, Halton, MLHS, (*t,m,s*)-nets, Gaussian quadrature) in the ML model can be found in Bhat (2001), Sándor and Train (2004), and Hess et al. (2005). Note that the computation of the D_b-error for ML models means that for each draw of the prior parameters, a simulation of the probability in Equation (20) is needed. Different simulation methods can be used in combination, for example, one could use Halton draws for simulating the probability, while using Gaussian quadrature approximations for the computation of the D_b-error. The computation time for evaluating a Bayesian efficient design for estimating an ML model is very high, making it only feasible to find Bayesian efficient designs for ML models with a limited number of random parameters.

Appendix B. Experimental designs

The designs used in this paper are D-efficient (not necessarily D-optimal) designs using fixed priors created using the Ngene[7] software. As an example, the syntax for generating the design for model M2 is given below. It generates a D-efficient design for the specified MNL model with six choice situations. The prior parameter values are given between brackets for each parameter b and the attribute levels are given between brackets for each attribute x. Note that b1 and b2 are generic since they have the same name in both utility functions for the two alternatives, while b0, b3 and b4 are alternative-specific. Tables B.1 through B.6 list the experimental designs used for the analyses in the paper.

```
Efficient
;mtype = mnl
;alts = alt1, alt2
;rows = 6
;eff = d
;model:
U(alt1) = b0 [1.2] + b1[-0.09] * x1[0,20,30] + b2[-0.3] * x2[1,3,5]  + b3[0.5] * x3[2,4,6] /
U(alt2) = b1 * x1 + b2 * x2 + b4[0.8] * x4[2,4,6]$
```

Table B.1: Experimental design D1 (for model M1)

s	x_{11}	x_{12}	x_{21}	x_{22}
1	20	3	10	1
2	20	1	20	3
3	30	5	30	1
4	30	1	10	5
5	10	3	20	3
6	10	5	30	5

D-error $(\alpha = 0)$ = 0.029186
Bayesian D-error $(\alpha = 0.1)$ = 0.029352
Bayesian D-error $(\alpha = 0.3)$ = 0.030671

Table B.2: Experimental design D2 (for model M2)

s	x_{01}	x_{11}	x_{12}	x_{13}	x_{21}	x_{22}	x_{23}
1	1	10	3	6	30	3	6
2	1	30	5	6	20	1	2
3	1	20	1	4	20	5	6
4	1	30	3	4	10	3	4
5	1	20	1	2	10	5	2
6	1	10	5	2	30	1	4

D-error $(\alpha = 0)$ = 0.093658
Bayesian D-error $(\alpha = 0.1)$ = 0.097006
Bayesian D-error $(\alpha = 0.3)$ = 0.132040

[7] Ngene is currently in prototype status and is being developed at the University of Sydney.

Table B.3: Experimental design D3 (for model M3)

s	x_{01}	x_{11}	x_{12}	x_{13}	x_{02}	x_{21}	x_{22}	x_{23}	x_{31}	x_{32}	x_{33}	x_{34}
1	1	30	1	6	1	10	3	2	10	5	3	6
2	1	10	5	2	1	20	1	2	30	1	3	4
3	1	10	1	2	1	30	5	6	30	3	5	4
4	1	10	5	6	1	30	1	6	20	3	5	4
5	1	20	3	2	1	10	1	2	30	5	7	6
6	1	20	1	4	1	20	3	4	10	5	3	8
7	1	30	3	6	1	20	5	4	10	1	5	2
8	1	30	5	4	1	30	5	4	10	1	7	2
9	1	30	1	2	1	10	3	2	20	3	5	2
10	1	20	3	4	1	10	5	6	20	1	3	8
11	1	20	5	4	1	30	1	6	20	5	7	6
12	1	10	3	6	1	20	3	4	30	3	7	8

D-error $(\alpha = 0)$ = 0.056300
Bayesian D-error $(\alpha = 0.1)$ = 0.059608
Bayesian D-error $(\alpha = 0.3)$ = 0.086397

Table B.4: Experimental design D4 (for model M4)

s	x_{01}	x_{11}	x_{12}	x_{13}	x_{14}	x_{02}	x_{21}	x_{22}	x_{23}	x^{*}_{24}	x_{31}	x_{32}	x_{33}	x_{34}
1	1	20	5	4	7	1	10	1	2	1	30	3	3	8
2	1	10	3	4	4	1	30	1	6	0	10	5	5	4
3	1	10	5	2	7	1	30	5	4	1	10	1	3	4
4	1	30	1	6	4	1	20	5	4	0	20	1	7	2
5	1	10	3	2	7	1	20	3	2	0	30	3	7	6
6	1	20	1	2	10	1	10	5	6	0	30	3	5	6
7	1	30	5	4	7	1	10	3	2	1	20	1	7	2
8	1	30	3	6	10	1	20	5	6	1	20	1	3	6
9	1	20	3	4	10	1	30	1	6	1	10	5	7	8
10	1	30	5	6	10	1	10	1	4	0	20	3	5	4
11	1	10	1	6	4	1	30	3	4	1	30	5	5	8
12	1	20	1	2	4	1	20	3	2	0	10	5	3	2

D-error $(\alpha = 0)$ = 0.096534
Bayesian D-error $(\alpha = 0.1)$ = 0.10423
Bayesian D-error $(\alpha = 0.3)$ = 0.17769
* Attribute x_{24} is dummy-coded.

Table B.5: Experimental design D5 (for model M5)

s	x_{01}	x_{11}	x_{12}	x_{13}^{*}	x_{14}^{*}	x_{15}	x_{02}	x_{21}	x_{22}	x_{23}^{*}	x_{24}^{*}	x_{25}^{*}	x_{31}	x_{32}	x_{33}	x_{34}^{*}
1	1	10	5	0	0	7	1	20	1	0	0	1	10	5	5	0
2	1	10	3	0	1	4	1	30	5	1	0	0	30	3	7	0
3	1	20	1	0	0	10	1	10	3	0	0	1	20	3	5	1
4	1	30	5	1	0	4	1	30	1	0	1	0	10	5	3	0
5	1	20	3	0	1	4	1	30	3	0	0	0	20	5	3	1
6	1	20	1	1	0	7	1	30	1	1	0	1	10	5	7	1
7	1	10	1	1	0	4	1	20	5	0	0	1	30	5	5	1
8	1	10	3	0	0	7	1	30	1	0	1	1	30	3	7	0
9	1	30	5	0	0	10	1	30	3	0	1	0	20	3	7	1
10	1	30	1	0	1	10	1	20	3	0	1	1	20	3	5	1
11	1	20	5	0	1	10	1	20	1	1	0	1	30	1	5	0
12	1	20	3	1	0	10	1	10	5	0	1	1	10	1	7	0
13	1	20	3	0	0	7	1	10	5	1	0	0	20	3	3	0
14	1	30	5	1	0	10	1	20	1	1	0	0	30	1	3	1
15	1	30	5	0	1	7	1	10	3	0	0	0	20	5	5	0
16	1	10	1	0	0	7	1	10	3	0	0	0	10	1	3	1
17	1	10	3	1	0	4	1	10	5	0	1	0	30	1	7	1
18	1	30	1	0	1	4	1	20	5	1	0	1	10	1	3	0

D-error $(\alpha = 0)$ = 0.28606
Bayesian D-error $(\alpha = 0.1)$ = 0.49787
Bayesian D-error $(\alpha = 0.3)$ = 0.63800
* Attributes (x_{13}, x_{14}), (x_{23}, x_{24}), x_{25} and x_{34} are dummy-coded.

Table B.6: Experimental design D6 (for model M6)

s	x_{01}	x_{11}	x_{12}	x_{13}	x_{14}^{*}	x_{15}^{*}	x_{16}	x_{02}	x_{21}	x_{22}	x_{23}	x_{24}^{*}	x_{25}^{*}	x_{26}^{*}	x_{27}	x_{31}	x_{32}	x_{33}	x_{34}	x_{35}^{*}
1	1	10	5	3	0	0	10	1	10	5	3	1	0	1	1	10	10	1	7	1
2	1	20	10	5	0	1	7	1	10	5	3	0	1	1	2	10	10	1	5	1
3	1	20	5	3	0	0	7	1	20	10	1	0	1	0	3	10	5	5	3	1
4	1	20	10	3	1	0	7	1	10	15	5	1	0	1	1	20	5	1	7	0
5	1	10	5	5	0	0	4	1	10	5	3	0	0	0	1	10	15	3	3	0
6	1	10	15	3	0	0	10	1	20	15	1	0	1	1	1	30	10	3	5	0
7	1	10	15	3	0	1	4	1	20	15	5	1	0	1	2	30	5	3	5	1
8	1	20	15	1	0	1	4	1	30	15	3	0	1	0	2	20	15	5	3	1
9	1	20	10	1	0	1	4	1	30	5	5	1	0	0	3	30	10	3	7	0
10	1	30	15	1	1	0	10	1	30	5	5	0	1	1	2	20	15	1	5	0
11	1	10	10	5	0	1	4	1	20	15	3	0	0	1	3	30	5	3	7	0
12	1	30	15	1	1	0	7	1	30	10	1	0	0	0	2	20	15	5	3	1
13	1	30	5	3	1	0	7	1	30	15	1	1	0	0	2	10	5	5	5	1
14	1	30	5	5	0	0	10	1	10	5	1	0	1	0	1	10	10	5	7	0
15	1	10	10	1	1	0	4	1	20	10	3	0	0	1	3	30	15	1	3	1
16	1	30	10	5	0	1	10	1	10	10	1	0	0	0	3	30	10	5	7	1
17	1	30	5	5	1	0	10	1	30	10	5	1	0	1	3	20	15	1	5	0
18	1	20	15	1	0	0	7	1	20	10	5	0	0	0	1	20	5	3	3	0

D-error $(\alpha = 0)$ = 0.26361
Bayesian D-error $(\alpha = 0.1)$ = 0.27163
Bayesian D-error $(\alpha = 0.3)$ = 0.33554
* Attributes (x_{14}, x_{15}), (x_{24}, x_{25}), x_{26} and x_{35} are dummy-coded.