Likelihood approximation by numerical integration on sparse grids

Florian Heiss, Viktor Winschel

Likelihood approximation by numerical integration on sparse grids

Florian Heiss, Viktor Winschel

PII: S0304-4076(07)00255-2
DOI: doi:10.1016/j.jeconom.2007.12.004
Reference: ECONOM 2998

To appear in: *Journal of Econometrics*

Received date: 29 May 2007
Revised date: 15 November 2007
Accepted date: 12 December 2007

Cite this article as: Florian Heiss and Viktor Winschel, Likelihood approximation by numerical integration on sparse grids, *Journal of Econometrics* (2008), doi:10.1016/j.jeconom.2007.12.004

Likelihood Approximation by Numerical Integration on Sparse Grids*

Florian Heiss† and Viktor Winschel‡

November 15, 2007

The calculation of likelihood functions of many econometric models requires the evaluation of integrals without analytical solutions. Approaches for extending Gaussian quadrature to multiple dimensions discussed in the literature are either very specific or suffer from exponentially rising computational costs in the number of dimensions. We propose an extension that is very general and easily implemented, and does not suffer from the curse of dimensionality. Monte Carlo experiments for the mixed logit model indicate the superior performance of the proposed method over simulation techniques.

JEL codes: C15, C35, C63

Keywords: Likelihood Simulation; Multivariate Quadrature; Mixed Logit

*We would like to thank Alexander Ludwig, Axel Börsch-Supan, Daniel McFadden, Paul Ruud, Arthur van Soest, and Joachim Winter for valuable comments and suggestions.

†University of Munich, Department of Economics, Ludwigstr. 28 RG, 80539 Munich, Germany, mail@florian-heiss.de

‡University of Mannheim, Department of Economics, L7, 3-5, 68163 Mannheim, Germany, winschel@rumms.uni-mannheim.de

1 Introduction

Many econometric models imply likelihood functions that involve multidimensional integrals without analytically tractable solutions. This problem arises frequently in microeconometric latent dependent variable (LDV) models in which all or some of the endogenous variables are only partially observed. Other sources include unobserved heterogeneity in nonlinear models and models with dynamically optimizing agents.

There are different approaches for numerical integration. It is well known that Gaussian quadrature can perform very well in the case of one-dimensional integrals of smooth functions as suggested by Butler and Moffit (1982). Quadrature can be extended to multiple dimensions. The direct extension is a tensor product of one-dimensional quadrature rules. However, computing costs rise exponentially with the number of dimensions and become prohibitive for more than four or five dimensions. This phenomenon is also known as the curse of dimensionality of deterministic numerical integration.

The main problem of this product rule is that the class of functions in which it delivers exact results is not restricted to polynomials of a given total order. Unlike in the univariate case, general efficient quadrature rules for this class are much harder to directly derive and often intractable (Cools 2003, Judd 1998, Ch. 7.5). They are therefore usually considered impractical for applied research (Bhat 2001, Geweke 1996).

These problems led to the advancement and predominant use of simulation techniques for the numerical approximation of multidimensional integrals in the econometric literature, see for example McFadden (1989) or Börsch-Supan and Hajivassiliou (1993) and Geweke, Keane and Runkle (1997). Hajivassiliou and Ruud (1994) and Geweke (1996) provide an overview over the general approaches of simulation and Train (2003) provides a textbook treatment with a focus on discrete choice models, one of the major classes of models for which these methods were developed and frequently used.

This paper proposes and investigates the performance of a different approach to this class of problems. It can be traced back to Smolyak (1963) who provides a general rule

how to extend univariate operators to multiple dimensions. In the economics literature, Krüger and Kübler (2004) have used this approach for the approximation of macroeconomic models. Winschel (2005) uses a Smolyak-based strategy for a nonlinear Kalman filter. Integration based on Smolyak's rule has been advanced in recent research in numerical mathematics, for an overview see Bungartz and Griebel (2004).

The class of functions for which it is exact is confined to polynomials of a given total order. This dramatically decreases computational costs in higher dimensions compared to Gaussian quadrature extended to multiple dimensions by the product rule. It is based on one-dimensional quadrature but extends it to higher dimensions in a more careful way than the tensor product rule. This implies that it is easily implemented and very general since only one-dimensional quadrature nodes and weights have to be derived.

We start by introducing notation and presenting the problem of likelihood approximation and solutions commonly used in the literature in section 2. Section 3 introduces integration on sparse grids and discusses its features and implementation. Section 4 presents the Monte Carlo design and results and section 5 concludes.

2 Common Approaches to Likelihood Approximation

2.1 The problem

The likelihood function of many econometric models cannot be calculated analytically but is instead expressed as an expected value over one or several random variables. Leading examples are limited dependent or other nonlinear models for panel data with unobserved heterogeneity, different error components or multinomial choice models.

Let $\mathbf{x} = [x_1, ..., x_D]$ denote a vector of random variables and write a general integration problem as

$$I_D[g] \quad = \quad \int_{\Omega_1} \cdots \int_{\Omega_D} g(\mathbf{x})\, \tilde{w}(\mathbf{x})\, dx_D \cdots dx_1, \tag{1}$$

where $\tilde{w}(\mathbf{x})$ represents the joint p.d.f. of $\mathbf{x}$, and $g(\mathbf{x})$ is a function which is in the nontrivial case nonlinear in $\mathbf{x}$. The integral $I_D[g]$ represents the expected value of g.

It will be convenient to assume that the integral is expressed in a way so that the weight function $\tilde{w}(\mathbf{x})$ can be decomposed as

$$\tilde{w}(\mathbf{x}) = \prod_{d=1}^{D} w(x_d) \tag{2}$$

and where $\Omega_d = \Omega$ for all $d = 1, ..., D$. In the interpretation with $\mathbf{x}$ representing a vector of random variables and $\tilde{w}(\mathbf{x})$ their joint p.d.f., this restriction is equivalent to assuming that the random variables are independently and identically distributed. Independence is crucial for the remainder of this paper, while identical distributions are merely assumed for notational convenience to save on another subscript.

This structure of the weighting function is less restrictive than it might seem. If independence is violated in the original formulation of the problem, a change of variables often leads to such a structure. If for example $\mathbf{z}$ denotes a vector of jointly normally distributed random variables with mean $\boldsymbol{\mu}$ and covariance matrix Σ, then $\mathbf{x} = L^{-1}(\mathbf{z} - \boldsymbol{\mu})$ is distributed i.i.d. standard normal if L is the Cholesky decomposition of Σ such that $LL' = \Sigma$. Note, however, that the problem of the calculation of joint posterior distributions for Bayesian inference cannot in general be easily expressed in this fashion. This is the reason why for these problems, it is not possible to directly draw values from the joint distribution but one has to resort to MCMC or similar methods.

Since for nonlinear functions $g(x)$ the integral has in general no closed-form solution, it has to be approximated numerically.

2.2 Simulation

Monte Carlo simulation is the most commonly used technique in the econometric literature for the numerical approximation of integrals of the form (1) in the multivariate case $D > 1$. Given a number R of replications, a set of random numbers or "nodes" $[\mathbf{x}_1, ..., \mathbf{x}_R]$ is

generated such that each $\mathbf{x}_r$ is a draw from the distribution characterized by $\tilde{w}(\mathbf{x})$. The simulated integral is then equal to

$$S_{D,R}[g] = \frac{1}{R}\sum_{r=1}^{R} g(\mathbf{x}_r). \tag{3}$$

Under very weak conditions, the simulated value is unbiased and $\sqrt{R}$-consistent by a law of large numbers, independent of the dimension D of the problem. Hajivassiliou and Ruud (1994) discuss approaches and properties of simulation-based estimation.

There are different ways to generate the set of draws $[\mathbf{x}_1, ..., \mathbf{x}_R]$. Restriction (2) is useful, since the draws can be made independently across the dimensions. The generation of pseudo-random draws is implemented in all major software packages. The resulting values are independent across draws. Quasi-Monte Carlo methods and antithetic sampling algorithms distribute the nodes more evenly and create some sort of negative correlation between draws. Therefore, they generally achieve both a better approximation quality with a given number of replications and in many cases also faster convergence rates. Sándor and András (2004) discuss various approaches in the setting of likelihood simulation for multinomial probit models.

2.3 Univariate Quadrature

In cases where the integral is univariate, Gaussian quadrature and related approaches are potentially powerful alternatives to simulation. A leading example are random effects (RE) models like the RE probit model discussed by Butler and Moffit (1982). In this case, the scalar random variable x represents the individual random effect and $g(x)$ is the probability of the sequence of observed outcomes conditional on the explanatory variables, parameters and x. The integral $I_1[g]$ is the marginal (with respect to x) probability.

Quadrature rules depend on the distribution of x and deliver the exact value of the integral if $g(x)$ is a polynomial of a given order. Define a sequence of quadrature rules $V = \{V_i : i \in \mathbb{N}\}$ so that the order of polynomial exactness increases with i. Each rule V_i

specifies a set of R_i nodes $\mathbb{X}_i = [x_1, ..., x_R]$ and a corresponding weight function $w : \mathbb{X}_i \to \mathbb{R}$ that is appropriate for the distribution of x characterized by w and Ω.

The quadrature approximation of $I_1[g]$ by V_i is then given as

$$V_i[g] = \sum_{x \in \mathbb{X}_i} g(x) w_i(x). \tag{4}$$

Given nodes and weights, quadrature rules are straightforward to implement, since equation (4) merely requires to calculate a weighted sum of function values. While they are not trivial to determine, for the most common cases they are tabulated in the literature and efficient software is available, see for example Press, Flannery, Teukolsky and Vetterling (1993) and Miranda and Fackler (2002).

Gaussian quadrature rules are especially efficient in univariate integration. If V_i is a Gaussian rule with R_i nodes, then $V_i[g] = I_1[g]$ if g is a polynomial of order $2R_i - 1$ or less. The Weierstrass approximation theorem states that any function can be approximated by a polynomial arbitrarily closely under mild regularity conditions. This makes Gaussian quadrature attractive for general functions g. The sequence of approximations $V_i[g]$ converges to $I_1[g]$ under weak assumptions as the number of function evaluations rises with i. A sufficient condition is that g is bounded and Riemann-integrable. There are various results concerning the speed of convergence for additional smoothness properties. For example if g has n bounded derivatives, many Gaussian quadrature approximations converge to the true value at a rate of R^{-n}. This is much better than the $\sqrt{R}$-consistency of Monte Carlo simulation if $g(x)$ is sufficiently smooth. A more detailed discussion of Gaussian quadrature can be found in the literature, see for example Davis and Rabinowitz (1984).

It will be useful below to use nested sequences of quadrature rules in the sense that the set of nodes used by some rule is a subset of those used by one with a higher accuracy, so that $\mathbb{X}_i \subseteq \mathbb{X}_j$ if $i < j$. This is not the case for classical Gaussian rules – they generally use completely different nodes for each accuracy level i. An example for nested sequences of univariate quadrature rules are Kronrod-Patterson sequences (Patterson 1968). A Kronrod-Patterson rule with accuracy level i adds a number of points to the set of

nodes $\mathbb{X}_{i-1}$ of the preceding accuracy level and updates the weights. So by design, they are nested. The additional nodes are chosen such that a maximum polynomial exactness is achieved. Because of the restriction that all nodes in $\mathbb{X}_{i-1}$ are to be reused, Kronrod-Patterson rules generally require a higher number of nodes to achieve the same univariate polynomial exactness as Gaussian quadrature rules which optimally choose the nodes without the requirement of nested sets.[1]

2.4 Multivariate Quadrature

In multiple dimensions, the general approach to approximate an integral with a rule which is exact for polynomials of a given order works the same way as in the univariate case. The definition of the order of a multivariate polynomial deserves some qualification since it is less obvious than in the univariate case. The most commonly used definition is the total order, see Judd (1998, Ch. 6.12). Consider a D-variate polynomial

$$g(x_1, ..., x_D) = \sum_{t=1}^{T} a_t \prod_{d=1}^{D} x_d^{j_{t,d}} \tag{5}$$

for some $T \in \mathbb{N}$, $[a_1, ..., a_T] \in \mathbb{R}^T$ and $[j_{t,1}, ..., j_{t,D}] \in \mathbb{N}^D$ for all $t = 1, ..., T$. The total order of g is defined as the maximal sum of exponents $\max_{t=1,...,T} \sum_{d=1}^{D} j_{t,d}$.

Multivariate polynomials with a bounded total order are also known as complete polynomials. They play an important role in function approximation – for example multivariate Taylor series approximations are complete polynomials. Consider a second-order Taylor approximation in two dimensions. It involves the terms x_1, x_2, x_1^2, x_2^2, and $x_1 x_2$. It is a polynomial of total order 2, since the sums of exponents do not exceed 2.

Unlike in the univariate case, general efficient quadrature rules for complete polynomials in the multivariate case are much harder to directly derive, Judd (1998, Ch. 7.5) provides

[1] With one or three integration nodes, the Kronrod-Patterson rule and the Gaussian rule coincide. With $2^R - 1$ nodes for $R > 1$, Gaussian rules are exact for polynomials up to order $2(2^R - 1) - 1$, whereas Kronrod-Patterson rules are only exact for polynomials up to order $3 \cdot 2^{R-1} - 1$. So the ratio of both approaches $3/4$ as m rises.

an introduction to this topic. One of the earlier and best known examples is the paper of Radon (1948) which provides a two-dimensional rule which is exact for polynomials of total order 5 or less on a rectangular region with uniform weight. Since then, a large literature on very specific problems for different numbers of dimensions, orders of polynomial exactness, integration region, and weight function has evolved. For a collection of these rules, see Stroud (1971) and Cools (2003). Because of their limitation to narrowly defined problems, they are usually considered impractical for applied econometric research (Bhat 2001, Geweke 1996).

A much simpler approach is to view a multivariate integral as a sequence of nested univariate integrals and combine univariate quadrature rules in a tensor product fashion. Define the tensor product of univariate quadrature rules with potentially different accuracy levels in each dimension indicated by the multi-index $\mathbf{i} = [i_1, ..., i_D]$ as

$$(V_{i_1} \otimes \cdots \otimes V_{i_D})[g] \quad = \quad \sum_{x_1 \in \mathbb{X}_{i_1}} ... \sum_{x_D \in \mathbb{X}_{i_D}} g(x_1, ..., x_D) \prod_{d=1}^{D} w_{i_d}(x_d), \tag{6}$$

where the nodes $\mathbb{X}_{i_1}, ...\mathbb{X}_{i_D}$ and weights $w_{i_1}, ..., w_{i_D}$ are those implied by the underlying one-dimensional quadrature rules $V_{i_1}, ..., V_{i_D}$. The product rule $T_{D,k}$ for D-variate Gaussian quadrature with accuracy level k is simply this tensor product with the same accuracy in each dimension $(V_k \otimes \cdots \otimes V_k)[g]$.

This rule is widely known and often taken to be *the* multivariate quadrature rule. For likelihood approximations, it has been used for example by Naylor and Smith (1988). A well known fact is that the product rule suffers from a "curse of dimensionality": It evaluates the function g at the full grid of points $\mathbb{X}_k \otimes \cdots \otimes \mathbb{X}_k$. In D dimensions, the product rule therefore requires R^D evaluations of the function g if the underlying univariate rule V_k is based on R nodes. This exponential growth of computational costs with the number of dimensions makes the product rule inefficient in a moderate number of dimensions and infeasible in high dimensions. While for example Gaussian quadrature exactly evaluates a univariate polynomial of order 7 with 4 function evaluations, the corresponding product rule with 20 dimensions requires $4^{20} = 1,099,511,627,776$ evaluations.

The reason for this exponential growth lies in the fact that the product rule is not exact in a class of polynomials of a bounded total order but for a tensor product of univariate polynomials. Coming back to the two-dimensional example with order two, in addition to complete polynomials of order two involving the terms x_1, x_2, x_1^2, x_2^2, and x_1x_2, it is also exact for polynomials involving the higher-order terms $x_1^2x_2$, $x_1x_2^2$, and $x_1^2x_2^2$. The total number of these terms rises exponentially which drives the "curse of dimensionality". At the same time, the intuition from Taylor series approximation suggests that the additional terms do not increase the speed of convergence of the approximated to the true function since they vanish at a higher rate.

Integration on sparse grids will be discussed in section 3. It shares advantages from specifically designed multivariate quadrature rules and the product rule. Since it combines univariate rules, it is as general and simple to use as the product rule. Since it aims to be exact in the class of complete polynomials instead of tensor products of univariate polynomials, it does not suffer from exponentially growing computational costs.

2.5 Efficient Change of Variables

If the integrand in a numerical integration problem is not well-behaved, all algorithms can perform poorly. For an example of such problems of simulation, see Lee (1997) and for a similar problem of univariate quadrature see Lee (2000). For all these problems, a change of variables can make the integrand better suited for numerical analysis.

There are different approaches to automatically analyze the integration problem and perform such a change of variables which can improve the approximation performance considerably. If the integral is simulated, this corresponds to an importance sampling algorithm with a clever choice of the sampling distribution, see Richard and Zhang (2005). A similar trick can be used in deterministic numerical integration, see Naylor and Smith (1988), Liu and Pierce (1994), and Rabe-Hesketh, Skrondal and Pickles (2005).

For the remainder of this paper, these improvements are not used. The main reason is to level the playing field between different algorithms by ensuring that relative performance

differences can be attributed to the algorithms and not to potentially different integration problems.

3 Quadrature on Sparse Grids

3.1 The algorithm

The integration rule discussed in this section will be exact for complete polynomials of a given order in multiple dimensions. Like the product rule, it combines univariate quadrature rules, so it is very general and easy to implement. But unlike the product rule, its computational costs do not rise exponentially but considerably slower. The basic idea goes back to Smolyak (1963) and is a general method for multivariate extensions of univariate operators.

The Smolyak approach has attracted attention in numerical mathematics. For a survey of this literature, see Bungartz and Griebel (2004). In the economics literature, the Smolyak construction has been used for the numerical approximation of macroeconomic models by Krüger and Kübler (2004) and Winschel (2005).

For deterministic integration, the construction can be defined as follows. For an underlying sequence of univariate quadrature rules, define $V_0[g] = 0$ and the difference of the approximation when increasing the level of accuracy from $i-1$ to i as

$$\Delta_i[g] = V_i[g] - V_{i-1}[g] \quad \forall i \in \mathbb{N}. \tag{7}$$

With $\mathbf{i} = [i_1, ..., i_D]$, define for any nonnegative integer q

$$\mathbb{N}_q^D = \left\{ \mathbf{i} \in \mathbb{N}^D : \sum_{d=1}^{D} i_d = D + q \right\} \tag{8}$$

and $\mathbb{N}_q^D = \emptyset$ for $q < 0$. For example, $\mathbb{N}_2^2 = \{[1,3],[2,2],[3,1]\}$. The Smolyak rule with accuracy level $k \in \mathbb{N}$ for D-dimensional integration is defined as

$$A_{D,k}[g] \;=\; \sum_{q=0}^{k-1} \sum_{\mathbf{i} \in \mathbb{N}_q^D} \left(\Delta_{i_1} \otimes \cdots \otimes \Delta_{i_D}\right)[g]. \tag{9}$$

The rule starts with a crude product rule – note that $A_{D,1}[g] = (V_1 \otimes \cdots \otimes V_1)[g]$. With a higher value of k, it sequentially incorporates the effect of increasing the accuracy in each dimensions in a particular way.

It is instructive to express $A_{D,k}[g]$ directly in terms of the univariate quadrature rules instead of their differences. Wasilkowski and Woźniakowski (1995) show that it can be written as

$$A_{D,k}[g] \quad = \quad \sum_{q=k-D}^{k-1} (-1)^{k-1-q} \binom{D-1}{k-1-q} \sum_{\mathbf{i} \in \mathbb{N}_q^D} (V_{i_1} \otimes \cdots \otimes V_{i_D})[g]. \tag{10}$$

This rule is a weighted sum of product rules with different combinations of accuracy levels $\mathbf{i} = [i_1, ..., i_D]$. Their sum is bounded which has the effect that the tensor product rules with a relatively fine sequence of nodes in one dimension are relatively coarse in the other dimensions. This is analogous to the bound on the sum of exponents for multivariate polynomials of a total order.

Figure 1 demonstrates the construction of the sparse grid by the Smolyak rule for a simple example with $D = 2$ and $k = 3$. The nodes for a sequence of univariate quadrature rules $\mathbb{X}_1, \mathbb{X}_2$, and $\mathbb{X}_3$ are shown in the top of the figure. The product rule $\mathbb{X}_3 \otimes \mathbb{X}_3$ evaluates the function at all two-dimensional combinations of nodes prescribed by $\mathbb{X}_3$ which are shown in the upper right part of the figure. As equation (10) shows, the sparse grids rule combines tensor products of lower order $\mathbb{X}_i \otimes \mathbb{X}_j$ such that $3 \leq i + j \leq 4$. The nodes of these products as well as the resulting sparse grid are shown in the lower part of the figure.

3.2 Properties

Quadrature on sparse grids is exact for polynomials of a given total order, as discussed for example by Bungartz and Griebel (2004) and stated in Theorem 1. This is of interest for general integrands since any well-behaved functions can be approximated by polynomials arbitrarily closely.

Theorem 1 *Assume that the sequence of univariate quadrature rules $V = \{V_i : i \in N\}$ is defined such that V_i is exact for Ω, w, and all univariate polynomials of order $2i-1$ or less. This implies the Smolyak rule $A_{D,k}$ using V as the univariate basis sequence is exact for D-variate polynomials of total order $2k-1$ or less.*

A proof of this theorem is provided in the appendix.

The set of nodes used by the sparse grids rule (10) can be written as

$$\mathbb{X}_{D,k} = \bigcup_{q=k-D}^{k-1} \bigcup_{\mathbf{i} \in \mathbb{N}_q^D} (\mathbb{X}_{i_1} \otimes \cdots \otimes \mathbb{X}_{i_D}). \tag{11}$$

The number of nodes in $\mathbb{X}_{D,k}$ depends on the univariate nodes $\mathbb{X}_1, ..., \mathbb{X}_k$ and can in general not be easily calculated – in general, the number of nodes increases polynomially in the number of dimensions (Bungartz and Griebel 2004). We discuss the case if Gaussian quadrature is used for the sequence of underlying univariate rules. Remember that the product rule $T_{D,k}$ with underlying Gaussian quadrature rules which uses k nodes in each dimension needs a total number of k^D nodes so that the logarithm of the nodes is of order $O(D)$ as $D \to \infty$.

Theorem 2 *Consider the sparse-grids rule $A_{D,k}$ with underlying Gaussian quadrature rules $V = \{V_i : i \in N\}$ such that each $\mathbb{X}_i$ used by V_i has i nodes. For a given accuracy k and rising D, the logarithm of nodes in $\mathbb{X}_{D,k}$ is of order $O(\log(D))$.*

We give a proof in the appendix. At least asymptotically, the number of nodes (its logarithm) does not rise exponentially (linearly) as for the product rule, but only polynomially (logarithmically). This is of course only of limited use in practice since realistic values for D are far from infinity. We therefore give precise numbers for different dimensions below. Before, we discuss alternatives to Gaussian quadrature as the underlying univariate rules.

Theorem 1 requires that in the sequence of quadrature rules $V_1, V_2...$ each V_i is exact for all univariate polynomials of order $2i-1$ or less. As discussed above, Gaussian quadrature rules achieve this requirement on univariate exactness with a minimal number of i nodes

for each V_i. Obviously, a low number of univariate quadrature nodes helps to obtain a low total number of nodes in the sparse grids rule.

In the example presented in Figure 1, the sets of univariate nodes are nested in the sense that $\mathbb{X}_i \subseteq \mathbb{X}_j$ if $i \leq j$. Because the nodes are nested, the sets $\mathbb{X}_1 \otimes \mathbb{X}_2$ and $\mathbb{X}_2 \otimes \mathbb{X}_1$ do not add any distinct nodes to the sparse grid and also the other sets share a substantial number of points. This makes the union of the tensor products a much smaller set than in the other extreme case in which each set contains different nodes so $\mathbb{X}_i \cap \mathbb{X}_j = \emptyset$ if $i \neq j$. Gaussian quadrature rules are close to the latter case – generally, only the midpoint is shared by rules with an odd number of nodes.

As discussed above, Kronrod-Patterson rules have nested sets of nodes. While they are less efficient in one dimension, this feature makes them more efficient for the use in quadrature on sparse grids. Petras (2003) discusses this problem for the case of unweighted integration (Gauss-Legendre equivalent) and Genz and Keister (1996) for the normal p.d.f. weights (Gauss-Hermite equivalent).

Table 1 shows the number of function evaluations required by different multivariate integration rules to achieve a given degree of polynomial exactness. The product rule suffers from the curse of dimensionality. The number of nodes for the Smolyak rule also rises with the number of dimensions, but substantially slower. As discussed, while in one dimension Gaussian quadrature is more efficient, in higher dimensions the Kronrod-Patterson rules need fewer nodes. As already mentioned in section 2.4, there are deterministic integration rules specifically designed for combinations of the number of dimensions, degree of polynomial exactness, integration region, and weight function. These can be more efficient than the Smolyak rule. For example, Stroud (1971) provides rules for a level 7 of polynomial exactness which need 141 and 9,961 nodes in 5 and 20 dimensions, respectively. These numbers are slightly smaller than the corresponding 151 and 10,001 nodes needed by the Kronrod-Patterson version of the Smolyak rule.

3.3 Implementation

Sparse grids integration can be easily implemented in practice. Equation (10) can be written explicitly with the definition in (6) as

$$A_{D,k}[g] = \sum_{q=k-D}^{k-1} \sum_{\mathbf{i}\in\mathbb{N}_q^D} \sum_{x_1\in\mathbb{X}_{i_1}} \dots \sum_{x_D\in\mathbb{X}_{i_D}} g(x_1,\dots,x_D)(-1)^{k-1-q}\binom{D-1}{k-1-q}\prod_{d=1}^{D} w_{i_d}(x_d). \tag{12}$$

This formula boils down to a weighted sum of function evaluations $g(\mathbf{x})$. The sets of nodes $\mathbf{x} = [x_1, ..., x_D]$ are determined by the relevant combinations of nodes of univariate quadrature rules V_i, where the levels of accuracy in each dimension are determined by $\mathbf{i} \in \mathbb{N}_q^D$ and $k - D \leq q < k$. The corresponding weights are $(-1)^{k-1-q}\binom{D-1}{k-1-q}\prod_{d=1}^{D} w_{i_d}(x_d)$. Especially if nested quadrature rules are used, the same vectors $\mathbf{x}$ will appear multiple times for different combinations of values of $\mathbf{i}$. Instead of evaluating g several times for the same values, it suffices to do this once and sum up the respective weights beforehand.

To come back to the simple example with $D = 2$ and $k = 3$, equation 12 works out as follows: q takes the values 1 and 2. For $q = 1$, $\mathbb{N}_q^D = \{[1, 2], [2, 1]\}$. So the nodes of the quadrature rule with accuracy level 1 in dimension 1 and level 2 in dimension 2 are combined as well the other way around. The weight for each of these combined nodes is the product of the one-dimensional quadrature weights times $(-1)^{k-1-q}\binom{D-1}{k-1-q} = -1$. For $q = 2$, $\mathbb{N}_q^D = \{[1, 3], [2, 2], [3, 1]\}$. The nodes resulting from a combination of the univariate rule of accuracy level 1 in dimension 1 with level 3 in dimension 2, those from a combination of the levels 2 and 2, and those from a combination of levels 3 and 1 are added to the set of nodes used for the Smolyak rule. Their weight is obtained as the product of the corresponding univariate weights times $(-1)^{k-1-q}\binom{D-1}{k-1-q} = 1$.

In the application for estimation, integrals have to be evaluated many times. Since the nodes and weights do not depend on g, they can be calculated once for the relevant underlying quadrature rule, number of dimensions D, and accuracy level k. The result is a set of nodes $x_{r,d}$ and weights w_r with $d = 1, ..., D$ and $r = 1, ..., R$, where R denotes

the number of nodes after removing duplicates. For each evaluation of the integral, simply calculate

$$A_{D,k}[g] \quad = \quad \sum_{r=1}^{R} g(x_{r,1}, ..., x_{r,D}) w_r. \tag{13}$$

In practice, the only difference to using the product rule is the way the nodes and weights are determined. The only additional difference to simulation is, that weighted instead of unweighted means have to be calculated.

We provide the readily calculated matrices of nodes and weights for Gaussian and nested quadrature rules both for unweighted integrals and integrals with Gaussian weights and for many combinations of the number of dimensions and accuracy levels. Furthermore, we provide code for the software packages Matlab and Stata to generate these for general problems. All downloads can be found at `http://www.sparse-grids.de`. This makes integration on sparse grids straightforward to implement.

3.4 Discussion and Extensions

As univariate quadrature, all multivariate quadrature rules rely on a polynomial approximation of the integrand. In all cases, ill-behaved integrands such as functions with discontinuities can be expected to hamper a satisfactory approximation. Simulation methods do not require smoothness of the integrand, although with a finite number of simulation draws, smooth integrands also help to improve the approximation performance (Stern 1992). Transformations of an ill-behaved integrand like an adaptively determined change of variables can be expected to work as well for sparse grids integration as for the product rule for which it is suggested by Naylor and Smith (1988).

While the computational costs do only rise polynomially instead of exponentially, this increase is noticeable with a high number of dimensions. The *speed of convergence* of simulation approximations does not depend on the number of dimensions. This however does not imply that the *level* of simulation error given a number of simulation draws is independent of the number of dimensions. With a high number of dimensions, also

accurate approximation by simulation can be computationally very costly, see for example Lee (1997).

One potential drawback of the integration on sparse grids approach is that – as equation (10) shows – some of the weights implied by the Smolyak rule can become negative. This feature is shared with many specially designed multivariate quadrature rules (Cools 2003). While we did not encounter this problem in our simulations, it is theoretically possible that the approximated integral becomes negative even if the integrand is positive everywhere. This effect can be seen as evidence that the approximation is extremely crude for the given level of accuracy and should disappear for higher accuracy levels.

Quadrature on sparse grids as it is defined above takes as an input an accuracy level for all dimensions which translates into an order of polynomial exactness. It can be easily changed to be more accurate in some dimensions than in others which might be useful if it is clear a priori that accuracy in some dimensions is especially crucial for exact likelihood approximations. The simplest approach is to use different sequences of underlying univariate quadrature rules for each dimension.

There is no general rule which accuracy level is needed for a specific problem. Instead of setting it to a fixed number, the error of approximation can be estimated by comparing the results with different accuracy levels. This is especially efficient with nested univariate rules since the set of nodes for a given accuracy level is a subset of those for a higher level. When increasing the accuracy, $g(\cdot)$ has to evaluated at the additional nodes only and the other values obtained during the calculations for the lower level can be reused. Gerstner and Griebel (2003) suggest a 'dimension adaptive' algorithm based on sparse grids integration which uses a similar approach but searches for a sufficient accuracy level in each dimension separately.

4 Monte Carlo Simulations: Mixed Logit Models

4.1 The Model

In this section we present Monte Carlo experiments to assess the relative performance of the numerical integration algorithms. Different random parameters logit or mixed multinomial logit (MML) models are implemented. This model is widely used in applied econometrics for studying choices between a finite set of alternatives, see for example Revelt and Train (1998) and Brownstone and Train (1999). McFadden and Train (2000) provide an introduction to this model and a discussion of its estimation by simulation methods. Chiou and Walker (2007) discuss the identification and estimation of the MML model. This model has also been used before to study the performance of different general simulation methods (Bhat 2001, Hess, Train and Polak 2006).

Consider a random sample of N individuals. The data has a panel structure, so that for each of the subjects T choices are observed. In each of these choice situations, the individual is confronted with a set of J alternatives and chooses one of them. These alternatives are described by K strictly exogenous attributes. The $(K \times 1)$ vectors $\mathbf{x}_{itj}$ collect these attributes of alternative $j = 1, ..., J$ in choice situation $t = 1, ..., T$ of individual $i = 1, ..., N$.

Random utility maximization (RUM) models of discrete choices assume that the individuals pick the alternative which results in the highest utility. The researcher obviously does not observe these utility levels. They are modeled as latent variables for which the observed choices provide an indication. Let the utility that individual i attaches to alternative j in choice situation t be represented by the random parameters specification

$$U_{itj} \quad = \quad \mathbf{x}'_{itj}\boldsymbol{\beta}_i + e_{itj}. \tag{14}$$

It is given by a linear combination of the attributes of the alternative, weighted with individual-specific taste levels $\boldsymbol{\beta}_i$. These individual taste levels are distributed across the population according to a parametric joint p.d.f. $f(\boldsymbol{\beta}_i; \boldsymbol{\theta})$ with support $\Psi \subseteq \mathbb{R}^K$. The i.i.d.

random variables e_{itj} capture unobserved utility components. They are assumed to follow an Extreme Value Type I (or Gumbel) distribution.

Our goal is to estimate the parameters $\boldsymbol{\theta}$ of the taste level distribution. Let y_{itj} denote an indicator variable that has the value 1 if individual i chooses alternative j in choice situation t and 0 otherwise. Denote the vector of observed individual outcomes as $\mathbf{y}_i = [y_{itj}; t = 1, ..., T, j = 1..., J]$ and the matrix of all strictly exogenous variables as $\mathbf{x}_i = [\mathbf{x}_{itj}; t = 1, ..., T, j = 1..., J]$. The probability that the underlying random variable $\mathbf{Y}_i$ equals the observed realization $\mathbf{y}_i$ conditional on $\mathbf{x}_i$ and the individual taste levels $\boldsymbol{\beta}_i$ can be expressed as

$$P_i^*(\boldsymbol{\beta}_i) = \Pr(\mathbf{Y}_i = \mathbf{y}_i | \mathbf{x}_i, \boldsymbol{\beta}_i) \quad = \quad \prod_{t=1}^{T} \frac{\prod_{j=1}^{J} \exp(\mathbf{x}'_{itj}\boldsymbol{\beta}_i)^{y_{itj}}}{\sum_{j=1}^{J} \exp(\mathbf{x}'_{itj}\boldsymbol{\beta}_i)}. \tag{15}$$

The likelihood contribution of individual i is equal to the joint outcome probability as a function of $\boldsymbol{\theta}$. It can be written as

$$P_i(\boldsymbol{\theta}) = \Pr(\mathbf{Y}_i = \mathbf{y}_i | \mathbf{x}_i, \boldsymbol{\theta}) \quad = \quad \int_{\Psi} P_i^*(\boldsymbol{\beta}_i) f(\boldsymbol{\beta}_i; \boldsymbol{\theta}) \, d\boldsymbol{\beta}_i. \tag{16}$$

A solution for this K-dimensional integral does in general not exist in closed form and has to be approximated numerically.

4.2 Approximation of Outcome Probabilities

For illustration purposes, we start with a simple case of the general model. Let $J = 2$ so that the model simplifies to a binary choice model. Also let there only be $K = 1$ attribute of the alternatives for which $x_{it2} - x_{it1} = 1$ for all t. Consequently, the individual taste parameter β_i is a scalar. Assume it is normally distributed over the population with mean 1 and variance σ^2. Let the individual have chosen T_1 times alternative 1 and T_2 times alternative 2, so that there are in total $T_1 + T_2$ observations. The likelihood contribution in equation (16) can then be simplified to

$$\begin{aligned} P_i(\theta) \quad &= \quad \int_{-\infty}^{\infty} P_i^*(z)\phi(z) \, dz \\ \text{with } P_i^*(z) \quad &= \quad (1 + \exp(-1 - \sigma z))^{-T_1} (1 + \exp(1 + \sigma z))^{-T_2}. \end{aligned} \tag{17}$$

This univariate integral can either be simulated or approximated using standard Gaussian quadrature. Figure 2 shows the function $P_i^*(z)$ for two different cases and depicts the numerical approaches to its integration. The simulated probability with R simulation draws can be represented as the sum of R rectangles each of which has a width of $1/R$ and a height that corresponds to the function value at randomly chosen points. Quadrature exactly integrates a polynomial of a given degree that represents an approximation to the integrand.

How well $P_i^*(z)$ is approximated by a low-order polynomial depends on the parameters. With high T and σ^2, the function has large areas in which it is numerically zero (or unity). These areas create a problem for the polynomial fit. In Figure 2, two cases are presented. In the simple case of Model 1 with $T_1 = 0, T_2 = 1$, and $\sigma^2 = 1$, the function $P_i^*(z)$ – and therefore its integral – is already well approximated by a third-order polynomial. A ninth-order polynomial is indistinguishable from the original function. In order to integrate this ninth-order polynomial exactly, Gaussian quadrature rules only need $R = 5$ function evaluations.

In the second model with $T = 20, T_1 = 8, T_2 = 12$, and $\sigma^2 = 5$, the problem of large tails with zero conditional probability is evident. A third-order polynomial does a poor job in approximating the function and there are noticeable differences between the original function and its ninth-order polynomial approximation. A 19^{th}-order polynomial for which Gaussian quadrature needs 10 function evaluations however is again indistinguishable from the true function. This can of course be arbitrarily problematic with even higher σ^2 and T. With a sharp and narrow peak, approximation by simulation can have poor properties, too. Intuitively, this is since only a small fraction of simulation draws are within the nonzero area. As discussed in section 2.5, a change of variables can help to make the function better suited for numerical integration regardless of the integration method.

For the two models depicted in Figure 2 and two more extreme cases, Table 2 presents performance measures of simulation and Gaussian quadrature approximations of the choice probabilities (17). The numbers presented are absolute errors for the quadrature approx-

imations and root mean squared errors for simulations which have been performed 1,000 times for each model and number of draws. All errors are defined relative to the value obtained by a Riemann sum with 10,000 grid points. For all models, Gaussian quadrature with 10 nodes performs better than simulation with 1,000 draws.

To study more complex models, we turn to a setup where $J = T = 5$. The number of explanatory variables K determines the dimensionality of the integration problem. We chose $K = 3, 5, 10$, and 20 for the Monte Carlo studies reported in Table 3. The individual taste levels are specified as i.i.d. normal random variables with mean 1 and variance $2/K$ to hold the total variance of U_{itj} constant as K changes. Instead of using one predefined data set, we draw 1,000 samples from the joint distribution of $\mathbf{y}_i$ and $\mathbf{x}_i$, where the x_{itj} are specified as independent uniform random variables and the conditional distribution of the Bernoulli random variables $\mathbf{y}_{itj}$ is given in equation (15). For each of these draws, we approximate the joint outcome probability using simulation and Smolyak integration with different numbers of nodes. For the calculation of the mean squared errors, we approximate the true value by simulation with 200,000 draws.

The rows denoted as "simulation" represent simulated probabilities using a standard random number generator. They perform worst in all cases. The "quasi Monte Carlo" results are obtained using modified latin hypercube sequences (MLHS) which are shown to work well for the estimation of MML models by Hess et al. (2006).[2] This method works much better than the standard simulation. The product rule performs better than both simulation methods in low dimensions, especially $K = 3$. In five dimensions, its advantage disappears and in ten dimensions, it is clearly the worst method. For $K = 20$, we did not obtain results since it is not computationally feasible. In all cases, the sparse grids method clearly outperforms all other methods. Table 5 in the appendix shows results for the more difficult case $\sigma^2 = 5/K$ and $J = T = 5$. While all errors rise, the relative performances remain unchanged.

[2]We also experimented with Halton sequences which do not seem to make too much of a difference compared to MLHS. Results can be requested from the authors.

4.3 MSL Estimation

One of the main reasons why approximations of the outcome probabilities are interesting is that they are required for most estimators of the parameters $\boldsymbol{\theta}$. We discuss maximum simulated (or approximated) likelihood estimation such that the estimators are defined as

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \sum_i \log\left(\tilde{P}_i(\boldsymbol{\theta})\right),$$

where $\tilde{P}_i(\boldsymbol{\theta})$ is some approximation of the individual joint outcome probability $P_i(\boldsymbol{\theta})$. Alternatively, estimators could be based on simulated scores or moments. For a discussion of the various approaches, see for example Hajivassiliou and Ruud (1994). We chose this estimator since it is easy to implement and by far the most widely used for these kinds of models, see for example Brownstone and Train (1999), McFadden and Train (2000), and Chiou and Walker (2007).

It is intuitive that the quality of approximation of $\tilde{P}_i(\boldsymbol{\theta})$ translates into the properties of the estimators. We specify a number of different models and estimate the parameters using simulation, antithetic sampling, and deterministic integration using different degrees of accuracy. As a starting point, a reference model is specified with $N = 1000$, $T = 5$, $J = 5$, $K = 10$, $\mu = 1$, and $\sigma = 0.5$. These numbers are chosen to represent a typical application in applied choice analysis. In order to cover a wide range of realistic values and to assess their impact on the approximation errors of the different methods, each of these numbers is varied separately. For each of these settings, estimates were obtained for 100 artificial data sets. The K-dimensional vectors of properties of the alternatives $\mathbf{x}_{itj}$ were drawn from a standard uniform distribution. The model parameters μ and σ are constrained to be equal for all properties to simplify the estimation and estimated for each data set. We used the same methods as discussed in the previous section pseudo-random Monte Carlo (PMC), quasi-random Monte Carlo (QMC) and sparse grids integration (SGI).

Table 4 shows results for different dimensions of integration. The simulation-based estimates are much better for μ than for σ. This can be explained by the fact that while the simulated probabilities $\tilde{P}_i(\mu, \sigma)$ are unbiased for the true values $P_i(\mu, \sigma)$, the log

transformation introduces downward bias. This bias depends on the simulation variance which in turn depends on σ. This tends to bias $\hat{\sigma}$ downwards. As predicted from the results for the approximated probabilities, standard simulation is dominated by quasi-random simulation. SGI is again clearly the best method and for example in ten dimensions requires only 21 nodes for the same accuracy for which QMC needs 201 and PMC 1201 function evaluations.

In the appendix, results for other model parameter choices are presented. Table 6 shows variations of μ and σ and Table 7 varies N, T, and J. The basic findings are unaffected by these changes. As σ increases, the approximation error rises and therefore all methods perform worse.[3] With a larger number of i.i.d. cross-sectional observation units N or longitudinal observations T, the estimators improve. Their relative advantages remain unaffected.

5 Conclusions

Smolyak (1963) introduced a general rule for extending univariate operators to multivariate problems. This approach has attracted recent interest in the numerical mathematics literature for function approximation and numerical integration.

We discuss this approach for the prevalent problem of likelihood approximation by numerical integration in multiple dimensions. The main advantage of this sparse grids integration (SGI) rule over the well known product rule extension of univariate quadrature is that it does not impose exponentially increasing computational costs with a rising number of dimensions. As opposed to quadrature rules explicitly derived for multivariate integration problems, this approach is very general and straightforward to implement.

After introducing the method and discussing its properties, we present extensive Monte Carlo evidence for the likelihood approximation of the mixed logit model. The results

[3] If σ has a very large value, all methods fail to give reasonable estimation results. As discussed above, adaptive rescaling might solve this problem.

suggest that SGI works very well for a wide range of model parameterizations. The computational costs to achieve a negligible approximation error are considerably lower than with simulation estimators. Together with suggested refinements, SGI is a promising candidate for the efficient likelihood approximation of a large class of econometric models.

Appendix

Proof of Theorem 1

Note that

$$A_{D,k}\left[\sum_{t=1}^{T} a_t \prod_{d=1}^{D} x_d^{j_{t,d}}\right] = \sum_{t=1}^{T} a_t A_{D,k}\left[\prod_{d=1}^{D} x_d^{j_{t,d}}\right].$$

Therefore, it suffices to establish polynomial exactness for any of the T monomials. Consider $g = \prod_{d=1}^{D} x_d^{j_d}$ for some sequence $j_1, ..., j_D$ with

(i) $\sum_{d=1}^{D} j_d \leq 2k-1$.

The theorem states that this implies $A_{D,k}[g] = I_D[g]$. For the sequence of underlying univariate quadrature rules $V_1, V_2, ...$ we have by assumption

(ii) $V_k[x^j] = I_1[x^j]$ if $j \leq 2k-1$.

For the univariate case $D = 1$, the sparse grids rule simplifies to the univariate quadrature rule:

$$A_{1,k}[g] \;=\; \sum_{i=1}^{k} (V_i[g] - V_{i-1}[g]) = V_k[g], \tag{18}$$

so the theorem follows immediately from (ii). For the multivariate case, a proof is presented via induction over D. Suppose that polynomial exactness has been established for $D-1$ dimensions:

(iii) $A_{D-1,\tilde{k}}\left[\prod_{d=1}^{D-1} x_d^{j_d}\right] = I_{D-1}\left[\prod_{d=1}^{D-1} x_d^{j_d}\right]$ if $\sum_{d=1}^{D-1} j_d \leq 2\tilde{k}-1$

It remains to be shown that this implies polynomial exactness for D dimensions.

First note that because of the multiplicative structure of the monomial integrand,

$$(\Delta_{i_1} \otimes \cdots \otimes \Delta_{i_D})\left[\prod_{d=1}^{D} x_d^{j_d}\right] = \prod_{d=1}^{D} \Delta_{i_d}[x_d^{j_d}].$$

Rewrite the general Smolyak rule (9) by separating the sum over the D^{th} dimension:

$$A_{D,k}[g] \;=\; \sum_{i_D=1}^{k} \sum_{\tilde{q}=0}^{k-i_D} \sum_{\mathbf{i}\in\mathbb{N}_{\tilde{q}}^{D-1}} (\Delta_{i_1} \otimes \cdots \otimes \Delta_{i_D})[g].$$

Combining these two expressions, we get

$$A_{D,k}\left[\prod_{d=1}^{D} x_d^{j_d}\right] = \sum_{i_D=1}^{k} \Delta_{i_D}\left[x_D^{j_D}\right] A_{D-1,k-i_D+1}\left[\prod_{d=1}^{D-1} x_d^{j_d}\right].$$

By (ii), we know that whenever $2(i_D-1) > j_D$, $V_{i_D}\left[x_D^{j_D}\right] = V_{i_D-1}\left[x_D^{j_D}\right] = I\left[x_D^{j_D}\right]$. Therefore, $\Delta_{i_D}\left[x_D^{j_D}\right]$ and the summands are zero unless $j_D \geq 2(i_D-1)$. This in turn implies together with (i) that for nonzero summands $\sum_{d=1}^{D-1} j_d \leq 2(k-i_D+1)-1$ and therefore $A_{D-1,k-i_D+1}\left[\prod_{d=1}^{D-1} x_d^{j_d}\right] = I_{D-1}\left[\prod_{d=1}^{D-1} x_d^{j_d}\right]$ by (iii).

With $i_D^* = .5(j_D+1)$ for odd j_D and $i_D^* = .5j_D$ for even j_D, it follows that

$$\begin{aligned} A_{D,k}\left[\prod_{d=1}^{D} x_d^{j_d}\right] &= I_{D-1}\left[\prod_{d=1}^{D-1} x_d^{j_d}\right] \sum_{i_D=1}^{i_D^*} \Delta_{i_D}\left[x_D^{j_D}\right] \\ &= I_{D-1}\left[\prod_{d=1}^{D-1} x_d^{j_d}\right] V_{i_D^*}\left[x_D^{j_D}\right] \end{aligned}$$

By (ii), $V_{i_D^*}\left[x_D^{j_D}\right] = I_1\left[x_D^{j_D}\right]$ and the theorem follows.

Proof of Theorem 2

Let $R_{D,k}$ denote the number of distinct nodes in $\mathbb{X}_{D,k}$. For $D \geq k$, it can be bounded as

$$R_{D,k} \leq \sum_{q=0}^{k-1} \sum_{\mathbf{i}\in\mathbb{N}_q^D} \prod_{d=1}^{D} i_d, \tag{19}$$

since the univariate quadrature rule V_i has i nodes. The inequality comes from the fact that the midpoints appear repeatedly in the underlying quadrature rules. Note that the average element of $\mathbf{i} \in \mathbb{N}_q^D$ is $\frac{D+q}{D}$ and that the product is maximized if all elements have the same value. Therefore

$$\prod_{d=1}^{D} i_d \leq \left(\frac{D+q}{D}\right)^D \quad \forall \mathbf{i} \in \mathbb{N}_q^D.$$

The number of vectors $\mathbf{i} \in \mathbb{N}_q^D$ is $\binom{D-1+q}{D-1}$. So

$$R_{D,k} \leq \tilde{R}_{D,k} = k\binom{D-1+k}{D-1}\left(\frac{D+k}{D}\right)^D.$$

As $D \to \infty$, $\left(\frac{D+k}{D}\right)^D \to \exp(k)$, $\binom{D-1+k}{D-1} \to \frac{D^k}{k!}$ and therefore $\log\left(\tilde{R}_{D,k}\right) \to k - \log((k-1)!) + k\log(D) = O(\log(D))$.

References

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

Börsch-Supan, Axel, and Vassilis Hajivassiliou (1993) 'Smooth unbiased multivariate probability simulators for maximum likelihood estimation of limited dependent variable models.' *Journal of Econometrics* 58, 347–368

Brownstone, David, and Kenneth Train (1999) 'Forecasting new product penetration with flexible substitution patterns.' *Journal of Econometrics* 89, 109–129

Bungartz, Hans-Joachim, and Michael Griebel (2004) 'Sparse grids.' *Acta Numerica* 13, 147–269

Butler, J. S., and Robert Moffit (1982) 'A computationally efficient quadrature procedure for the one-factor multinomial probit model.' *Econometrica* 50(3), 761–764

Chiou, Lesley, and Joan L. Walker (2007) 'Masking identification of discrete choice models under simulation methods.' *Journal of Econometrics*

Cools, Ronald (2003) 'An encyclopedia of cubature formulas.' *Journal of Complexity* 19, 445–453

Davis, Philip J., and Philip Rabinowitz (1984) *Methods of Numerical Integration,* 2nd ed. (New York: Academic Press)

Genz, Alan, and B. D. Keister (1996) 'Fully symmetric interpolatory rules for multiple integrals over infinite regions with Gaussian weight.' *Journal of Computational and Applied Mathematics* 71, 299–309

Gerstner, T., and M. Griebel (2003) 'Dimension-adaptive tensor-product quadrature.' *Computing* 71, 65–87

Geweke, John (1996) 'Monte Carlo simulation and numerical integration.' In *Handbook of Computational Economics Vol. 1,* ed. Hans M. Amman, David A. Kendrick, and John Rust (Amsterdam: Elsevier Science) pp. 731–800

Geweke, John F., Michael P. Keane, and David E. Runkle (1997) 'Statistical inference in the multinomial multiperiod probit model.' *Journal of Econometrics* 80, 125–165

Hajivassiliou, Vassilis A., and Paul A. Ruud (1994) 'Classical estimation methods for LDV models using simulation.' In *Handbook of Econometrics Vol. 4,* ed. Robert F. Engle and Daniel L. McFadden (New-York: Elsevier) pp. 2383–2441

Hess, Stephane, Kenneth E. Train, and John W. Polak (2006) 'On the use of a modified latin hypercube sampling (MLHS) method in the estimation of a mixed logit model for vehicle choice.' *Transportation Research Part B* 40, 147–163

Judd, Kenneth L. (1998) *Numerical Methods in Economics* (Cambridge, Mass.: MIT Press)

Krüger, D., and F. Kübler (2004) 'Computing equilibrium in OLG models with stochastic production.' *Journal of Economic Dynamics and Control* 28, 1411–1436

Lee, Lung-Fei (1997) 'Simulated maximum likelihood estimation of dynamic discrete choice statistical models: Some Monte Carlo results.' *Journal of Econometrics* 82, 1–35

Lee, Lung-fei (2000) 'A numerically stable quadrature procedure for the one-factor random-component discrete choice model.' *Journal of Econometrics* 95, 117–129

Liu, Qing, and Donald A. Pierce (1994) 'A note on Gauss–Hermite quadrature.' *Biometrika* 81, 624–629

McFadden, Daniel (1989) 'A method of simulated moments for estimation of discrete choice models without numerical integration.' *Econometrica* 57, 995–1026

McFadden, Daniel, and Kenneth Train (2000) 'Mixed MNL models for discrete response.' *Journal of Applied Econometrics* 15, 447–470

Miranda, Mario J., and Paul L. Fackler (2002) *Applied Computational Economics and Finance* (Cambridge MA: MIT Press)

Naylor, J.C., and A.F.M. Smith (1988) 'Econometric illustrations of novel numerical integration strategies for Bayesian inference.' *Journal of Econometrics* 38, 103–125

Patterson, Thomas N. L. (1968) 'The optimum addition of points to quadrature formulae.' *Mathematics of Computation* 22, 847–856

Petras, Knut (2003) 'Smolyak cubature of given polynomial degree with few nodes for increasing dimension.' *Numerische Mathematik* 93, 729–753

Press, William H., Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling (1993) *Numerical Recipes in C, 2nd Edition* (Cambridge University Press)

Rabe-Hesketh, Sophia, Anders Skrondal, and Andrew Pickles (2005) 'Maximum likelihood estimation of limited and discrete dependent variable models with nested random effects.' *Journal of Econometrics* 128, 301–323

Radon, Johann (1948) 'Zur mechanischen kubatur.' *Monatshefte für Mathematik* 52(4), 286–300

Revelt, David, and Kenneth Train (1998) 'Mixed logit with repeated choices: Households' choices of appliance efficiency level.' *Review of Economics and Statistics* 80, 647–657

Richard, Jean-François, and Wei Zhang (2005) 'Efficient high-dimensional importance sampling.' Working Paper, University of Pittsburgh, Dept. of Economics

Sándor, Zsolt, and P. Péter András (2004) 'Alternative sampling methods for estimating multivariate normal probabilities.' *Journal of Econometrics* 120, 207–234

Smolyak, Sergey A. (1963) 'Quadrature and interpolation formulas for tensor products of certain classes of functions.' *Soviet Mathematics Doklady* 4, 240–243

Stern, Steven (1992) 'A method for smoothing simulated moments of discrete probabilities in multinomial probit models.' *Econometrica* 60(4), 943–952

Stroud, Arthur Howard (1971) *Approximate calculation of multiple integrals* (Englewood Cliffs, N.J.: Prentice-Hall)

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

Wasilkowski, Grzegorz W., and Henryk Woźniakowski (1995) 'Explicit cost bounds of algorithms for multivariate tensor product problems.' *Journal of Complexity* 8, 337–392

Winschel, Viktor (2005) 'Solving, estimating and selecting nonlinear dynamic economic models without the curse of dimensionality.' mimeo, University of Mannheim

Tables

Table 1: Number of function evaluations

	Product rule	Smolyak rule	
Dimensions	**Gaussian**	**Gaussian**	**KP**
Level $k = 2$, Polynomial exactness = 3			
$D = 1$	2	2	3
$D = 5$	32	11	11
$D = 10$	1,024	21	21
$D = 20$	1,048,576	41	41
Level $k = 3$, Polynomial exactness = 5			
$D = 1$	3	3	3
$D = 5$	243	61	51
$D = 10$	59,049	221	201
$D = 20$	3,486,784,401	841	801
Level $k = 4$, Polynomial exactness = 7			
$D = 1$	4	4	7
$D = 5$	1,024	241	151
$D = 10$	1,048,576	1,581	1,201
$D = 20$	1,099,511,627,776	11,561	10,001
Level $k = 5$, Polynomial exactness = 9			
$D = 1$	5	5	7
$D = 5$	3,125	781	391
$D = 10$	9,765,625	8,761	5,281
$D = 20$	95,367,431,640,625	120,321	90,561

Table 2: Approximating probabilities in one dimension: RMSE

	$R = 2$	$R = 5$	$R = 10$	$R = 100$	$R = 1000$
Model 0: $T = 1, T_0 = 0, T_1 = 1, \sigma^2 = .25$					
Simulation	0.0944	0.0569	0.0421	0.0130	0.0043
Quadrature	0.0046	0.0008	0.0002	0.0000	–
Model 1: $T = 1, T_0 = 0, T_1 = 1, \sigma^2 = 1$					
Simulation	0.1846	0.1111	0.0822	0.0253	0.0085
Quadrature	0.0102	0.0012	0.0002	0.0000	–
Model 2: $T = 20, T_0 = 8, T_1 = 12, \sigma^2 = 4$					
Simulation	1.0206	0.6745	0.4841	0.1516	0.0484
Quadrature	0.8115	0.2472	0.0024	0.0000	–
Model 3: $T = 30, T_0 = 10, T_1 = 20, \sigma^2 = 9$					
Simulation	1.5222	0.9658	0.6591	0.2121	0.0677
Quadrature	1.0000	0.6336	0.0402	0.0000	–

The reported numbers are root mean squared errors relative to the "true" value

Table 3: RMSE of probabilities: $\sigma^5 = 2/K, J = T = 5$

$K = 3, R =$	7	8	87	125	495	512
Simulation	0.3373	0.2971	0.0879	0.0776	0.0372	0.0382
Quasi MC	0.2287	0.1802	0.0382	0.0331	0.0161	0.0152
Product rule		0.0669		0.0112		0.0048
Sparse grids	0.0303		0.0050		0.0020	
$K = 5, R =$	11	32	151	243	903	1024
Simulation	0.2663	0.1448	0.0705	0.0536	0.0303	0.0278
Quasi MC	0.1486	0.0796	0.0310	0.0257	0.0127	0.0130
Product rule		0.0567		0.0277		0.0171
Sparse grids	0.0243		0.0049		0.0043	
$K = 10, R =$	21	201	1024	1201		
Simulation	0.1987	0.0654	0.0284	0.0255		
Quasi MC	0.1076	0.0317	0.0135	0.0128		
Product rule			0.0420			
Sparse grids	0.0173	0.0211		0.0035		
$K = 20, R =$	41	801	10001			
Simulation	0.1428	0.0324	0.0095			
Quasi MC	0.0795	0.0156	0.0048			
Sparse grids	0.0125	0.0160	0.0027			

The reported numbers are root mean squared errors relative to the "true" value

Table 4: Errors of the estimated parameters with different K

	RMSE ($\hat{\mu}$)			RMSE ($\hat{\sigma}$)		
R	PMC	QMC	SGI	PMC	QMC	SGI
Dimension $K = 2$						
9	0.0486	0.0458	0.0448	0.3648	0.1648	0.1177
45	0.0458	0.0452	0.0448	0.1712	0.1246	0.1177
961	0.0449	0.0449	0.0448	0.1179	0.1170	0.1177
Dimension $K = 4$						
9	0.0411	0.0361	0.0340	0.3857	0.1985	0.0902
81	0.0341	0.0340	0.0339	0.1319	0.0963	0.0923
1305	0.0338	0.0339	0.0339	0.0921	0.0916	0.0923
Dimension $K = 10$						
21	0.0481	0.0353	0.0272	0.2951	0.1554	0.0668
201	0.0298	0.0277	0.0272	0.0874	0.0708	0.0654
1201	0.0276	0.0271	0.0271	0.0691	0.0662	0.0654
Dimension $K = 14$						
29	0.0507	0.0348	0.0240	0.2493	0.1321	0.0601
393	0.0252	0.0247	0.0252	0.0665	0.0618	0.0632
3361	0.0251	0.0252	0.0249	0.0629	0.0637	0.0631
Dimension $K = 20$						
41	0.0655	0.0465	0.0290	0.2323	0.1390	0.0620
801	0.0298	0.0285	0.0276	0.0634	0.0599	0.0564
10001	0.0289	0.0291	0.0291	0.0594	0.0585	0.0585

The reported numbers are RMSEs relative to the true value

Table 5: RMSE of probabilities: $\sigma^2 = 5, J = T = 10$

$K = 3, R =$	7	8	87	125	495	512
Simulation	0.7284	0.8184	0.2093	0.1860	0.0902	0.0923
Quasi MC	0.6234	0.5849	0.1498	0.1147	0.0636	0.0600
Product rule		0.2060		0.0480		0.0221
Sparse grids	0.2696		0.0217		0.0055	
$K = 5, R =$	11	32	151	243	903	1024
Simulation	0.6628	0.3921	0.2264	0.1652	0.0807	0.0726
Quasi MC	0.5226	0.3517	0.1550	0.1106	0.0653	0.0564
Product rule		0.1828		0.0917		0.0566
Sparse grids	0.2434		0.0567		0.0151	
$K = 10, R =$	21	201	1024	1201		
Simulation	0.6708	0.2025	0.0950	0.0816		
Quasi MC	0.5278	0.1783	0.0784	0.0713		
Product rule			0.1574			
Sparse grids	0.1798	0.1058		0.0573		
$K = 20, R =$	41	801	10001			
Simulation	0.4928	0.1169	0.0348			
Quasi MC	0.3971	0.1176	0.0286			
Product rule						
Sparse grids	0.1034	0.0756	0.0395			

The reported numbers are root mean squared errors relative to the "true" value

Table 6: Errors of the estimated parameters with different μ and σ

	RMSE ($\hat{\mu}$)			RMSE ($\hat{\sigma}$)		
R	PMC	QMC	SGI	PMC	QMC	SGI
Parameters $\mu = 0.5, \sigma = 0.5$						
21	0.0268	0.0226	0.0208	0.2922	0.1533	0.0625
201	0.0211	0.0208	0.0209	0.0813	0.0638	0.0603
1201	0.0209	0.0209	0.0209	0.0631	0.0625	0.0605
Parameters $\mu = 2, \sigma = 0.5$						
21	0.0842	0.0598	0.0460	0.3050	0.1545	0.0712
201	0.0510	0.0470	0.0475	0.1034	0.0767	0.0738
1201	0.0478	0.0475	0.0472	0.0803	0.0732	0.0735
Parameters $\mu = 1, \sigma = 0.25$						
21	0.0264	0.0253	0.0257	0.1759	0.0984	0.0919
201	0.0250	0.0252	0.0255	0.0929	0.0888	0.0893
1201	0.0255	0.0256	0.0256	0.0868	0.0955	0.0923
Parameters $\mu = 1, \sigma = 1$						
21	0.1070	0.0867	0.0490	0.4095	0.3053	0.1754
201	0.0409	0.0363	0.0343	0.0994	0.0824	0.0746
1201	0.0316	0.0309	0.0303	0.0614	0.0597	0.0553

The reported numbers are RMSEs relative to the true value

Table 7: Errors of the estimated parameters with different N, T, and J

	RMSE ($\hat{\mu}$)			RMSE ($\hat{\sigma}$)		
R	PMC	QMC	SGI	PMC	QMC	SGI
$N = 500$						
21	0.0511	0.0441	0.0417	0.2993	0.1792	0.1101
201	0.0416	0.0414	0.0418	0.1284	0.1102	0.1126
1201	0.0413	0.0417	0.0417	0.1023	0.1109	0.1121
$N = 2000$						
21	0.0417	0.0275	0.0165	0.3124	0.1530	0.0547
201	0.0185	0.0171	0.0166	0.0769	0.0575	0.0515
1201	0.0168	0.0166	0.0166	0.0548	0.0517	0.0512
$T = 3$						
21	0.0513	0.0414	0.0374	0.3254	0.1836	0.1157
201	0.0378	0.0369	0.0383	0.1283	0.1220	0.1253
1201	0.0380	0.0381	0.0381	0.1208	0.1240	0.1234
$T = 10$						
21	0.0195	0.0189	0.0281	0.2284	0.1375	0.0457
201	0.0253	0.0269	0.0291	0.0572	0.0466	0.0377
1201	0.0284	0.0288	0.0294	0.0393	0.0385	0.0376
$J = 3$						
21	0.0619	0.0456	0.0363	0.2996	0.1485	0.0845
201	0.0389	0.0362	0.0370	0.1080	0.0931	0.0978
1201	0.0376	0.0372	0.0371	0.0972	0.0960	0.0964
$J = 10$						
21	0.0339	0.0271	0.0214	0.2638	0.1415	0.0540
201	0.0226	0.0215	0.0216	0.0705	0.0563	0.0561
1201	0.0215	0.0215	0.0215	0.0569	0.0558	0.0559

The reported numbers are RMSEs relative to the true value

Figures

Figure 1: Construction of the sparse grid in two dimensions

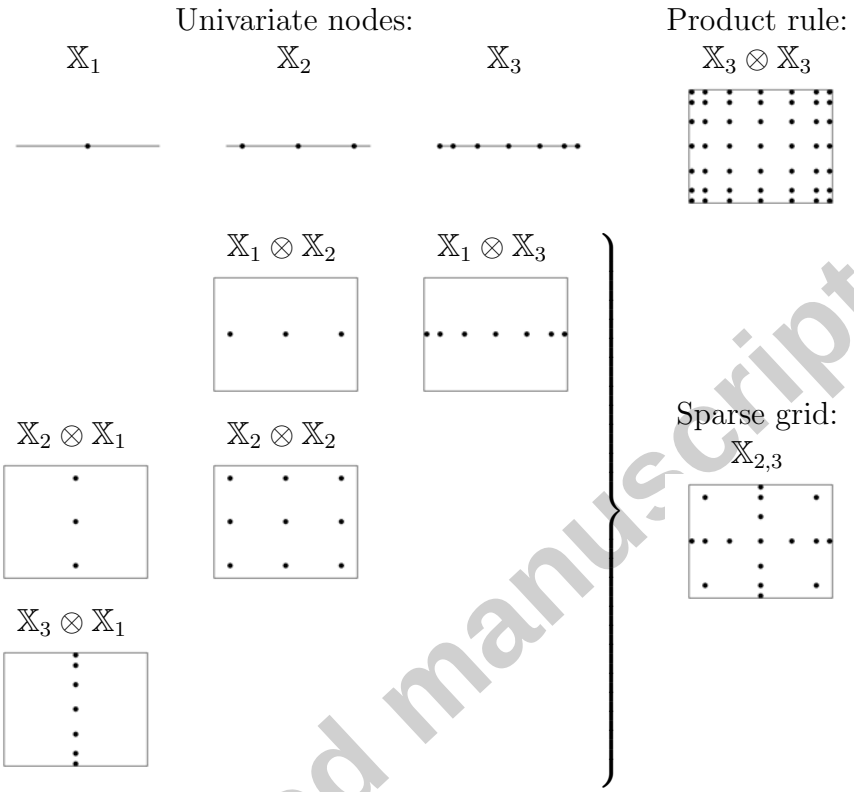
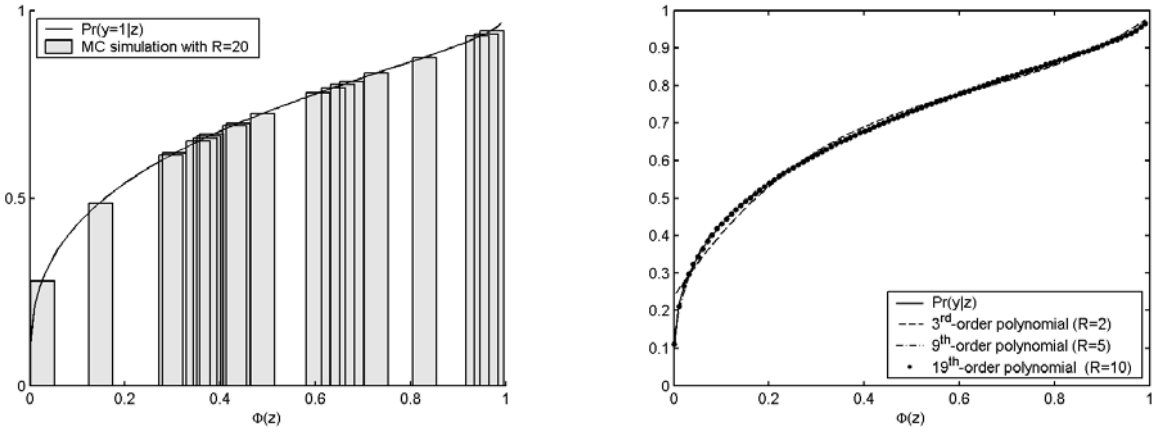


Figure 2: Approximating probabilities in one dimension

Model 1: $T = 1, T_1 = 0, T_2 = 1, \sigma^2 = 1$

Simulation $(R = 20)$ Quadrature $(R = 2, 5, 10)$



Model 2: $T = 20, T_1 = 8, T_2 = 12, \sigma^2 = 4$

Simulation $(R = 20)$ Quadrature $(R = 2, 5, 10)$