Time Series Simulation with Quasi–Monte Carlo Methods

Jenny X. Li [*] Peter Winker [†]

May 31, 2000

Abstract

The purpose of this paper is to compare the quasi–Monte Carlo methods, in particular the so–called (t, m, s)–nets , with classical Monte Carlo approaches for the simulation of econometric time series models. Quasi–Monte Carlo methods have found successful applications in many fields such as in physics, image processing, and the evaluation of finance derivatives. However, they are rarely used in the field of econometrics. In this paper, we apply both traditional Monte Carlo and quasi–Monte Carlo simulation methods to time series models which typically arise in macroeconometrics. The numerical experiments demonstrate that quasi–Monte Carlo methods outperform the traditional Monte Carlo for all models we investigate.

1 Introduction

The traditional Monte Carlo method is widely used in many research fields due to its simplicity. Indeed, it is computationally easy and its convergence rate is independent of the dimension of the problem. However, this method exhibits also several disadvantages. In particular, it can be computationally onerous to achieve a high level of accuracy, because its convergence rate is

[*]Departments of Mathematics and Economics, The Pennsylvania State University, University Park, PA, 16802, USA *Email:* `<li@math.psu.edu>`

[†]Department of Economics, University of Mannheim, D–68131 Mannheim, Germany. *Email:* `<Peter.Winker@vwl.uni-mannheim.de>`

only $O(N^{-1/2})$ for N sample paths. For example, to reduce the error by a factor of 10, the number of simulations would have to increase by a factor of 100. Hence, high accuracy requirements may lead to long computation times. Furthermore, it might be difficult to obtain "randomly" generated sample paths in high dimension (Ripley, 1987, pp. 23ff).

A quasi–Monte Carlo simulation uses more uniformly distributed deterministic sequences. It can provide a considerably improved convergence rate, close to $O(N^{-1})$ or even $O(N^{-3/2})$ in some special cases (?) (?).[1] This improvement in convergence rate has the potential for dramatic gains both in computational time and in the range of applications of simulation methods for econometric problems.

We will review both the Monte Carlo and the quasi–Monte Carlo method, and also introduce the quasi–Monte Carlo sequences such as: Fauré, Halton, Sobol and the most recently developed (t, m, s)–nets (Niederreiter, 1992). Special emphasis will be given to discussing the problems that may be encountered in implementing quasi–Monte Carlo methods and comparing their performance with the traditional Monte Carlo. Particularly we will apply both methods on time series models as they typically arise in the econometric modeling of macroeconomic data. We are especially interested in the uni– and multivariate autoregressive models (AR–, VAR–models), error correction models (ECM and VECM), and non–linear models as they appear, e.g. in form of the minimum condition or an aggregate matching function in models of temporary equilibrium (Franz *et al.*, 2000).

The rest of the paper is arranged as follows. In Section 2, we provide a basic description of Monte Carlo and quasi–Monte Carlo simulation techniques. In Section 3, we introduce the econometric time series models we investigate, and in Section 4 we discuss the results of an extensive numerical comparison study. Some final remarks are given in Section 5.

[1]See also Fang *et al.* (2000) on related results for latin hypercube designs.

2 Classical Monte Carlo, Low–Discrepancy Sequences and (t, m, s)–Nets

2.1 The Monte Carlo Simulation Framework

There exists a large literature on Monte Carlo simulation (Gentle, 1998). Therefore, we provide here only a brief overview and concentrate the discussion on applications to econometric time series models.

The problem of simulating time series models may be illustrated using the simplest case of a stochastic univariate autoregressive process

$$y_t = \alpha_0 + \alpha_1 y_{t-1} + \varepsilon_t \,.$$

For a given initial value y_0 and given parameter (estimates) α_0 and α_1, a sample path for y_t of length T can be simulated based on a sequence of ε_t of length T. However, usually a single realization of such a sample path is not sufficient, since one might be interested in the distribution of some y_t or some statistics derived from this distribution. Of course, such results are always conditional on the given or assumed joint distribution of all ε_t. Typical examples of statistics on y_t, which are of interest in econometric research, include its expectation and variance or confidence bands. In a simple linear setting with normally distributed error terms ε_t, these statistics can be obtained analytically. In more general settings, however, these statistics have to be approximated using the empirical distribution of the y_t obtained by simulating a large number N of sample paths. The resulting empirical distribution of the y_t is taken as Monte Carlo approximation of the true distribution.

The sample paths ε_{it} $(i = 1, \ldots, N,\ t = 1, \ldots, T)$ are obtained in three steps by classical Monte Carlo. First, a pseudo–random number generator is used to generate numbers uniformly distributed in the interval $(0, 1)$. Second, these uniformly distributed numbers are transformed to normally distributed numbers by some transformation method, e.g. Box–Muller or inverse method (Ripley, 1987, pp. 54ff). Finally, the vectors ε_{it} of length T for each $i = 1, \ldots, N$ are formed by using T consecutive numbers. If the inverse method is used for the transformation to normal deviates,[2] the problem of generating "good" ε's is equivalent to the problem of generating "good" random numbers

[2]Different transformation methods, such as the Box–Muller method, should be avoided in this context, as they might introduce artificial correlation (Ripley, 1987, pp. 54ff).

in the T–dimensional unit cube. By "good", we mean that for each time period, the distribution of the simulated ε's closely approximates $\mathcal{N}(0, I)$, where I denotes the T–dimensional identity matrix. Correspondingly, "good" random numbers in the unit cube $(0,1)^T$ should approximate the uniform distribution as close as possible.

For multivariate and non–linear time series models, the situation is similar. However, the dimension of the sample paths ε_{it} increases. For example, in a two–dimensional autoregressive models, sample paths of length $2T$ have to be generated: one of length T for the first equation and one of length T for the second. As the time series models usually assume that these sample paths are independent, this requirement has to be satisfied by Monte Carlo or quasi–Monte Carlo methods as well.

So, the key to Monte Carlo methods is to generate "good" random points. However, we can only generate pseudo–random sequences, which, in particular in higher dimension, might lead to clumping of points and, therefore, limits their uniformity. The cause of this clumping is the fact that points in a pseudo–random sequence are almost, but not completely independent. Thus, they have a chance of landing very close to each other.[3] The accuracy of Monte Carlo methods can be improved by using more uniformly distributed pseudo–random sequences. One straightforward method consists of the use of antithetic variates, i.e. with a random vector ε also $-\varepsilon$ is used for the simulation. In linear models, this method reduces the bias of the estimation of the mean to zero. However, the effect of antithetic variates on estimates of variance and in non–linear models is not clear.[4] Hence, quasi–Monte Carlo or low–discrepancy sequences methods strive for uniformly distributed point sets in a more general setting. Instead of using pseudo–random sequences, deterministic point sets are applied, thereby minimizing clustering and improving accuracy. In fact, by uniformly picking the points, higher accuracy may be achieved with a smaller number of simulations. However, what the uniformity really means in the context of time series simulation and which measure of uniformity is adequate in this case has not been addressed yet.

[3]Some bad examples are provided in Ripley (1987, pp. 23f).

[4]The simulation results in Section 4 provide some evidence.

2.2 Measuring the Uniformity of Point Sets

Intuitively, for a uniformly distributed sequence $\mathbf{x}_n$ in the s–dimensional unit cube $I^s = [0,1]^s$, we would expect to see exactly the same number of points located in all subsets of I^s which have the same volume. Actually, we can measure the uniformity of $\mathbf{x}_n$ in terms of its discrepancy. This is simply defined by considering the number of points in the subsets of I^s. Thus, for N points $\{\mathbf{x}_n\}_1^N$ in the s–dimensional unit cube $I^s = [0,1]^s$, $s \geq 0$, and a subset J of I^s the *local discrepancy* $D(J;N)$ is defined by

$$D(J;N) = A(J;N) - V(J)N\,,$$

where $A(J;N)$ is the number of $n, 1 \leq n \leq N$, with $\mathbf{x}_n \in J$ and $V(J)$ is the volume of the subinterval J. Hence, if the N points are uniformly distributed then the local discrepancy should be very small for all J's. This leads to a global definition, the *discrepancy*[5] $\Delta(N)$ of the N points, which is defined to be

$$\Delta(N) = \sup_J |D(J;N)|,$$

where the supremum is extended over all subsets J of the form $J = \prod_{i=1}^s [0, u_i)$. Unfortunately, it seems impossible to calculate this discrepancy for a point set if the number of points is large and the dimension s moderate (L'Ecuyer and Hellekalek, 1998, p. 230).[6] Furthermore, this measure of discrepancy has some shortcomings from a theoretical point of view, since it is not invariant to some natural transformations of the unit cube. Alternative measures can be derived by replacing the supremum norm, e.g. by the L_2 norm, and the considered subsets J (Hickernell, 1998). For this paper, we employ the centred L_2–discrepancy (CL_2) proposed by Hickernell (1998), which according to results reported in Fang *et al.* (2000) outperforms the discrepancy when searching for uniform designs for at least two reasons. First, it is more sensitive than the usual discrepancy, i.e. designs with identical discrepancy might differ markedly in their CL_2–discrepancy. Second, employing the CL_2–discrepancy results in low discrepancy for all lower dimensional projections of the point sets. Finally, Fang *et al.* (2000) establish a connection between low CL_2–discrepancy and orthogonality of designs.

[5]This measure of discrepancy based on the supremum norm is also known as *star–discrepancy* in the literature.

[6]An approximation based on the optimization heuristic threshold accepting is proposed in Winker and Fang (1997).

2.3 Low–Discrepancy Sequences and (t, m, s)–Nets

Any point set, which has very "small" discrepancy, is called low–discrepancy point set. There are many well–know low–discrepancy sets which are uniformly distributed in $(0, 1)^s$, for example Fauré–, Halton–, and Sobol–sequences (Press *et al.*, 1992, pp. 300ff) and (t, m, s)–nets . Halton–sequences describe a class of multidimensional infinite sequences that fill the interval $[0, 1)$. To generate a multidimensional Halton–sequence is quite easy. First, we begin with a consecutive sequence of non–negative integers, such as $n = 0, 1, 2, \cdots N - 1$ if N replications are required. For each integer n, convert it to its representation in the base p number system, where p is any prime number and $p \geq 2$ (e.g. 100 is the base 2 representation of the integer 4). Thereby, for each dimension, a different base p is chosen. Secondly, transform the base p representation into a number in the interval $[0, 1)$ by reflection about the decimal point. Interested readers are referred to **?** for a very detailed example. Fauré–sequences essentially are permutations of Halton–sequences, and the Sobol–sequence is a reordering of the Halton–sequence, too (Bratley and Fox, 1988). The low–discrepancy sequences proposed by Halton, Fauré, and Sobol are all called (t, s)–sequences, for which certain finite segments form (t, m, s)–nets . A (t, m, s)–net is a point set in I^s, for which the local discrepancy equals zero for many subsets. More specifically, let $0 \leq t \leq m$ be integers. A (t, m, s)–net in base b is a point set $\mathbf{x_n}$ of b^m points in $[0, 1)^s$ such that every elementary interval E in base b of volume $1/b^{m-t}$ contains exactly b^t points, where an *elementary interval* in base b is a subinterval E of $[0, 1)^s$ of the form

$$E = \prod_{i=1}^{s} [a_i b^{-d_i}, (a_i + 1) b^{-d_i}),$$

with integers $a_i, d_i \geq 0$, and $0 \leq a_i < b^{d_i}$ for $1 \leq i \leq s$.

There are many methods to construct (t, m, s)–nets ; see the survey by **?**. The most commonly used methods include: direct constructions using various properties of finite fields and polynomials over finite fields; error–correcting codes including both linear and nonlinear codes such as Kerdock codes; combinatorial methods including generalized orthogonal arrays; and a method which uses linear combinations of the rows of a so–called *generator matrix*, see Bierbrauer and Edel (1999) and Li and Mullen (2000) for more details.

Obviously, (t, s)–sequences and (t, m, s)–nets are closely related to each other. Indeed, (t, s)–sequences provide an effective way to construct (t, m, s)–nets , since the existence of a (t, s)–sequence in base b implies the existence of a $(t, m, s + 1)$–net for all $m \geq t$. However by using techniques other than (t, s)–sequences, we can often construct a net with a smaller value of t, which means a net with more uniformly distributed points.

For example, using the method described as construction 18 in Clayman *et al.* (1999), it is known that there is a $(6, 9)$–sequence in base 2 (and there is no known (5,9)–sequence). Hence there is a $(6, m, 10)$–net in base 2 for all $m > 6$. In particular, there is a $(6, 14, 10)$–net in base 2. However using the construction method indicated in Bierbrauer and Edel (1999) and discussed in considerable detail in Sections 4 and 5 of that paper, using generator matrices one can have a $(5, 14, 10)$–net in base 2. A $(5, 14, 10)$–net has a more uniform distribution of points than does a $(6, 14, 10)$–net.

2.4 Uniformity of Pseudo–Random and Quasi–Monte Carlo Point Sets

In this section we consider a number of pseudo–random and quasi–Monte Carlo points sets. The pseudo–random points sets will be used are: GAUSS, UNIF, ESSL and quasi–Monte Carlo points sets are: FAURE, HALTON, SOBOL, TMS. The sources of the generation are listed below.

GAUSS	Uniform pseudo–random number generator of GAUSS 3.2.4
UNIF	Uniform pseudo–random number generator from Bratley and Fox (1988)
ESSL	Uniform pseudo–random number generator from ESSL: $s_n = (a(s_{n-1}))\text{mod}m = (a^n s_0)\text{mod}m \quad x_n = s_n/m$, where s_0 is the initial seed, $a = 16807$ and $m = 2^{31} - 1$
FAURE	Fauré sequence generator from Bratley and Fox (1988)
HALTON	Halton sequence generator from Bratley and Fox (1988)
SOBOL	Sobol sequence generator from Bratley and Fox (1988)
TMS	TMS net generator from Li and Mullen (2000)

All those points sets provide good approximations to the uniform distribution. While the pseudo–random number generators are univariate by construction, the quasi–Monte Carlo methods provide multivariate point sets explicitly. For the pseudo–random number generators, higher dimensional vectors are obtained by stacking the corresponding number of drawings in one vector. The quality of the approximation of the uniform distribution is measured using the centred L_2–discrepancy. Table 1 shows the CL_2–discrepancy of some point sets obtained by pseudo–random number generators and quasi–Monte Carlo methods. As the point sets provided by pseudo–random number generators depend on the initial seed, the mean of 10 different point sets is reported for these generators. For some instances, different (t, m, s)–nets are available. Then, the table provides the smallest discrepancy found for all (t, m, s)–nets used in the simulation study in section 4.

Table 1: Centered L_2–discrepancy of Point Sets

	s = 10				
Method	N=1024	N=4096	N=16384	N=32768	N=65536
GAUSS	$0.715 \cdot 10^{-2}$	$0.181 \cdot 10^{-2}$	$0.452 \cdot 10^{-3}$	$0.216 \cdot 10^{-3}$	$0.107 \cdot 10^{-3}$
UNIF	$0.699 \cdot 10^{-2}$	$0.170 \cdot 10^{-2}$	$0.425 \cdot 10^{-3}$	$0.194 \cdot 10^{-3}$	$0.107 \cdot 10^{-3}$
ESSL	$0.698 \cdot 10^{-2}$	$0.170 \cdot 10^{-2}$	$0.425 \cdot 10^{-3}$	$0.194 \cdot 10^{-3}$	$0.107 \cdot 10^{-3}$
FAURE	$0.121 \cdot 10^{-2}$	$0.148 \cdot 10^{-3}$	$0.195 \cdot 10^{-4}$	$0.845 \cdot 10^{-5}$	$0.344 \cdot 10^{-5}$
HALTON	$0.133 \cdot 10^{-2}$	$0.204 \cdot 10^{-3}$	$0.250 \cdot 10^{-4}$	$0.825 \cdot 10^{-5}$	$0.197 \cdot 10^{-5}$
SOBOL	$0.832 \cdot 10^{-3}$	$0.137 \cdot 10^{-3}$	$0.194 \cdot 10^{-4}$	$0.812 \cdot 10^{-5}$	$0.288 \cdot 10^{-5}$
TMS	$0.364 \cdot 10^{-2}$	$0.207 \cdot 10^{-3}$	$0.186 \cdot 10^{-4}$	–	$0.509 \cdot 10^{-5}$

In Table 1, s is the dimension and N the number of points of the the sample point sets. The results for these instances clearly indicate that the CL_2–discrepancy of all quasi–Monte Carlo point sets is smaller than the expected value for point sets obtained by Monte Carlo methods.[7] A clear ranking of the quasi–Monte Carlo methods is not provided by this evidence. Only the (t, m, s)–net with 16 384 points has a slightly smaller centred L_2–discrepancy than the other quasi–Monte Carlo point sets. However, Niederreiter (1992) points out that (t, m, s)–nets yield the smallest discrepancy bound and therefore by the Koksma–Hlawka inequality the smallest error bound (within the class of functions of bounded variation in the sense of Hardy and Krause) among all known constructions of point sets. Especially, within the class of functions with rapidly converging Walsh series, Larcher and Traunfellner (1994) have shown that digital (t, m, s)–nets yield an error

[7]See also Fang *et al.* (2000) for a formal derivation of expectation values for CL_2 of Monte Carlo and quasi–Monte Carlo point sets.

bound of the optimal order of magnitude. Thus, the findings of table 1 have to be attributed either to the use of the centred L_2–discrepancy instead of the star–discrepancy or to some shortcomings of the specific construction of (t, m, s)–nets for these comparatively small sets.

3 Time Series Models

In the previous section we discussed different Monte Carlo and quasi–Monte Carlo methods for generating stochastic error paths for time series models. Furthermore, a comparison of the generated point sets based on a measure of discrepancy has been provided. In this section, we add evidence on the performance of the methods in both linear and non linear stochastic time series models as they typically appear, e.g., in macroeconometric modeling (Winker, 1999). While it is possible to obtain analytical solutions for the linear models, the benchmark solutions for the non linear models have to be obtained by a huge number of replications either in a standard Monte Carlo framework or — and this is the approach followed in this paper — in a quasi–Monte Carlo framework. We will discuss the number of replications we use and accuracy problems in Section 4.

Tables 2 and 3 provide an overview of the models used in our simulation setup. While models (1)–(6) are linear models, the last three models (7)–(9) exhibit some nonlinearities. Thus, analytical solutions for the mean response and its variance can be obtained for the first models, while the benchmarks for the latter are obtained using Sobol sequences with a large number of replications N. All one dimensional models are simulated over 10 time periods and all two dimensional models over 5 or 10 periods, respectively, fixing initial values to one. Thus, for the one dimensional models, the dimension of the error space is 10, hence 10–dimension pseudo–random or low discrepancy points sets are needed for the simulation. Similarly, 10– or 20–dimensional point sets will be used for the two dimensional models.

The numbers of replications used in the simulations are powers of 2, which is a result of the construction of (t, m, s)–nets in base 2. Thus, results are provided for 1024, 4096, 16 384, and 65 536 points, respectively. When antithetic variates are employed, only half the number of points is generated, and the other half is obtained by multiplying each vector ε with minus one.

Of course, the accuracy of the approximations could be improved using a larger number of replications. However, in macroeconometric modeling, the

Table 2: Models used for Simulation – I –

No.	Model	Equation	Parameters
(1)	AR(1)	$x_t = \alpha_0 + \alpha_1 x_{t-1} + \varepsilon_t$	$\alpha_0, \alpha_1, \sigma_\varepsilon$
(2)	VAR(1) 2–dim.	$x_t = \alpha_0 + \alpha_1 x_{t-1} + \alpha_2 y_{t-1} + \varepsilon_{1,t}$ $y_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 y_{t-1} + \varepsilon_{2,t}$	$\alpha_0, \alpha_1, \alpha_2, \sigma_{\varepsilon_1},$ $\beta_0, \beta_1, \beta_2, \sigma_{\varepsilon_2}$
(3)	AR(2)	$x_t = \alpha_0 + \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \varepsilon_t$	$\alpha_0, \alpha_1, \alpha_2, \sigma_\varepsilon$
(4)	VAR(2) 2–dim.	$x_t = \alpha_0 + \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \alpha_3 y_{t-1} + \alpha_4 y_{t-2} + \varepsilon_{1,t}$ $y_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 x_{t-2} + \beta_3 y_{t-1} + \beta_4 y_{t-2} + \varepsilon_{2,t}$	$\alpha_0, \alpha_1, \alpha_2,$ $\alpha_3, \alpha_4, \sigma_{\varepsilon_1},$ $\beta_0, \beta_1, \beta_2,$ $\beta_3, \beta_4, \sigma_{\varepsilon_2}$
(5)	ECM(1)	$\Delta x_t = \alpha_0 + \alpha_1 \Delta x_{t-1} + \alpha_2 \Delta y_{t-1} + \lambda(x_{t-1} - \beta y_{t-1}) + \varepsilon_t$	$\alpha_0, \alpha_1, \alpha_2,$ $\lambda, \beta, \sigma_\varepsilon$
(6)	VECM(1) 2–dim.	$\Delta x_t = \alpha_0 + \alpha_1 \Delta x_{t-1} + \alpha_2 \Delta y_{t-1} + \lambda_1(x_{t-1} - \gamma y_{t-1}) + \varepsilon_{1,t}$ $\Delta y_t = \beta_0 + \beta_1 \Delta x_{t-1} + \beta_2 \Delta y_{t-1} + \lambda_2(x_{t-1} - \gamma y_{t-1}) + \varepsilon_{2,t}$	$\alpha_0, \alpha_1, \alpha_2,$ $\lambda_1, \gamma, \sigma_{\varepsilon_1}$ $\beta_0, \beta_1, \beta_2,$ $\lambda_2, \sigma_{\varepsilon_2}$

solution of the model for a given set of errors is often quite time consuming. Therefore, the number of replications is limited by available computer resources. The actual number of replications used in stochastic policy simulation is often rather in the range 1000 to 2000 than larger than 10000 (Franz *et al.*, 2000). Furthermore, the gain of switching from Monte Carlo to quasi–Monte Carlo methods is often much larger than the gain of increasing the number of replications in a Monte Carlo setting.

4 Simulation

Already for the rather limited set of models presented in Tables 2 and 3 a high number of qualitatively different parameter settings could be considered. Both constraints of space and computing resources requires some selection of models. Furthermore, the outcome of model simulation can be evaluated

Table 3: Models used for Simulation – II –

No.	Model	Equation	Parameters
(7)	VAR(1)	$x_t = \alpha_0 + \alpha_1 x_{t-1} + \alpha_2 y_{t-1} + \varepsilon_{1,t}$	$\alpha_0, \alpha_1, \alpha_2, \sigma_{\varepsilon_1}$,
	2–dim.	$y_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 y_{t-1} + \varepsilon_{2,t}$	$\beta_0, \beta_1, \beta_2, \sigma_{\varepsilon_2}$
	min/max	$z_t = \min\{x_t, y_t\}$	
(8)	VAR(1)	$x_t = \alpha_0 + \alpha_1 x_{t-1} + \alpha_2 y_{t-1} + \varepsilon_{1,t}$	$\alpha_0, \alpha_1, \alpha_2, \sigma_{\varepsilon_1}$,
	2–dim.	$y_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 y_{t-1} + \varepsilon_{2,t}$	$\beta_0, \beta_1, \beta_2, \sigma_{\varepsilon_2}$
	CES	$z_t = \{x_t^{-\rho} + y_t^{-\rho}\}^{-\frac{1}{\rho}}$	ρ
(9)	VECM(1)	$\Delta x_t = \alpha_0 + \alpha_1 \Delta x_{t-1} + \alpha_2 \Delta y_{t-1}$	$\alpha_0, \alpha_1, \alpha_2$,
		$+\lambda_1(x_{t-1} - \gamma y_{t-1}) + \varepsilon_{1,t}$	$\lambda_1, \gamma, \sigma_{\varepsilon_1}$
	2–dim.	$\Delta y_t = \beta_0 + \beta_1 \Delta x_{t-1} + \beta_2 \Delta y_{t-1}$	$\beta_0, \beta_1, \beta_2$,
		$+\lambda_2(x_{t-1} - \gamma y_{t-1}) + \varepsilon_{2,t}$	$\lambda_2, \sigma_{\varepsilon_2}$
	CES	$z_t = \{x_t^{-\rho} + y_t^{-\rho}\}^{-\frac{1}{\rho}}$	ρ

using different measures. For this first explorative analysis, we concentrate on the bias of $\mathrm{E}(x_T)$ for models (1) to (6) and $\mathrm{E}(z_T)$ for models (7) to (9), where T denotes the last simulated period.[8] Furthermore, the bias of the estimated variance of x_T and z_T is also considered. In future research, we will also include measures like the bias of estimated 10– and 90–percent quantiles or the MSFE discussed in the paper by Ericcsson and Marquez (1998).

The number of time periods T defines the dimensionality of the error process used for the simulations. So far, we use only 10–dimensional processes. Therefore, $T = 10$ for the univariate models and $T = 5$ for the models with two stochastic shock components. Finally, the simulation outcomes depend on the number of replications. While a high number of replications eventually results in high quality results for all methods, we concentrate on rather small numbers of replications as they typically appear when simulating econometric models or estimators. Therefore, we report results for $N = 4096$, $N = 16384$ and $N = 65536$, respectively. Table 4 shows the bias for $\mathrm{E}(x_{10})$ in percent for model (1). For the pseudo–random generators, the mean of the absolute

[8] A similar approach is chosen by Acworth *et al.* (1998), when comparing the performance of Monte–Carlo and quasi–Monte Carlo methods for option pricing.

bias of ten different runs is reported.

Table 4: Bias (p.c.) of simulated $\mathrm{E}(x_{10})$ for Model (1) ($\alpha_0 = 0.1$, $\sigma_\varepsilon = 0.2$)

		Pseudo Random Numbers			Quasi–Monte Carlo Methods			
α_1	N	GAUSS	UNIF	ESSL	FAURE	HALTON	SOBOL	TMS
-0.99	4096	0.5001	0.5084	1.4363	0.0706	-0.1199	0.0149	0.0000
-0.99	16384	0.4605	0.4610	0.3922	0.0972	-0.0459	0.0083	0.0000
-0.99	65536	0.1501	0.1485	0.1607	0.0054	-0.0066	-0.0012	0.0000
-0.90	4096	1.0157	1.0340	2.3534	0.1150	-0.2730	0.0181	0.0000
-0.90	16384	0.6601	0.6631	0.7683	0.1358	-0.1084	0.0127	0.0000
-0.90	65536	0.2422	0.2412	0.2765	0.0007	-0.0167	-0.0029	0.0000
-0.50	4096	2.6063	2.6622	5.7335	0.2872	-1.4954	0.0990	0.0000
-0.50	16384	1.2873	1.2970	2.4894	0.0860	-0.6421	0.0370	0.0000
-0.50	65536	0.7355	0.7407	1.2406	-0.0391	-0.1133	-0.0072	0.0000
0.00	4096	1.7426	1.7329	2.7782	0.0861	-1.4771	0.1090	0.0000
0.00	16384	0.7956	0.7927	1.2153	-0.0124	-0.5336	0.0224	0.0000
0.00	65536	0.6722	0.6731	0.5722	-0.0177	-0.1208	-0.0009	0.0000
0.50	4096	1.4046	1.4059	1.1047	0.0179	-1.4133	0.0576	0.0000
0.50	16384	0.5487	0.5494	0.7799	-0.0207	-0.4590	0.0095	0.0000
0.50	65536	0.4142	0.4141	0.2831	-0.0241	-0.1293	0.0012	0.0000
0.90	4096	0.5447	0.5485	0.4930	-0.0368	-0.6581	0.0036	0.0000
0.90	16384	0.2138	0.2159	0.2554	-0.0180	-0.2164	-0.0003	0.0000
0.90	65536	0.1636	0.1631	0.1298	-0.0232	-0.0653	0.0002	0.0000
0.99	4096	0.4082	0.4111	0.3987	-0.0385	-0.4599	-0.0029	0.0000
0.99	16384	0.1845	0.1848	0.2113	-0.0186	-0.1510	-0.0014	0.0000
0.99	65536	0.1206	0.1201	0.1130	-0.0192	-0.0456	-0.0002	0.0000

The results for the estimated bias of $\mathrm{E}(x_{10})$ in model (1) are clearly in favour of quasi–Monte Carlo methods. In particular, (t, m, s)–nets provide unbiased estimates without using antithetic variates. Therefore, in the sequel antithetic variates are used only for the pseudo–random numbers and the other quasi–Monte Carlo methods. Furthermore, the speed of convergence, i.e. the decrease in the bias, when the number of replications N is increased, is much higher for the quasi–Monte Carlo methods mirroring the findings for the discrepancy of these point sets. Both for the pseudo–random number generators and the Halton sequences, the bias becomes smaller, when autocorrelation is high. For the pseudo–random numbers this finding can be explained by the fact that artificial correlation stemming from their construction becomes relatively smaller when autocorrelation of the process increases.

Table 5: Bias (p.c.) of simulated variance of (x_{10}) for Model (1) ($\alpha_0 = 0.1$, $\sigma_\varepsilon = 0.2$)

		Pseudo Random Numbers			Quasi–Monte Carlo Methods			
α_1	N	GAUSS	UNIF	ESSL	FAURE	HALTON	SOBOL	TMS
-0.99	4096	1.5927	1.6104	2.4209	-0.2865	0.1226	-0.1347	-0.3434
-0.99	16384	0.8573	0.8663	2.0180	-0.2887	0.0407	-0.4396	0.2600
-0.99	65536	0.6623	0.6592	0.7449	0.0102	-0.0096	0.0238	-0.0824
-0.90	4096	1.5637	1.4916	2.7374	-0.4077	0.2308	-0.4753	-0.5611
-0.90	16384	0.9292	0.9283	1.9133	-0.2374	0.0086	-0.7404	0.0575
-0.90	65536	0.4530	0.4513	0.6688	0.0302	-0.0174	0.0458	0.0318
-0.50	4096	1.1716	1.1780	2.9215	-0.1082	-0.3710	-1.6932	-0.0431
-0.50	16384	1.0441	1.0436	1.7506	-0.0969	-0.1426	-1.4697	-0.1200
-0.50	65536	0.5836	0.5831	0.5430	0.0445	-0.0341	-0.0077	0.0832
0.00	4096	2.3790	2.3593	2.8036	0.1360	-0.3785	-0.6623	-0.0076
0.00	16384	1.1960	1.2007	1.5163	-0.1688	-0.2354	-0.1870	-0.0020
0.00	65536	0.3141	0.3161	0.5072	0.0242	-0.0344	-0.0582	-0.0005
0.50	4096	3.2600	3.2365	2.1190	0.2575	-0.9234	0.1708	-0.0827
0.50	16384	0.9213	0.9015	1.1403	-0.1175	-0.5971	1.1219	0.1205
0.50	65536	0.3308	0.3296	0.5459	-0.0505	-0.1182	-0.0970	-0.0637
0.90	4096	2.2593	2.2643	2.7785	0.9080	-3.5490	-2.1190	-0.7205
0.90	16384	1.0401	1.0462	1.5611	0.0807	-1.3127	-0.1957	0.5963
0.90	65536	0.4219	0.4223	0.8896	-0.0735	-0.4119	-0.2254	0.2277
0.99	4096	1.3713	1.4312	2.7272	0.9600	-3.9162	-2.8039	-1.7599
0.99	16384	0.8311	0.8470	1.5107	0.1454	-1.3521	-0.6654	0.6146
0.99	65536	0.3943	0.3949	0.9864	-0.0441	-0.4352	-0.2664	0.4673

Table 5 shows the bias of simulated variance of x_{10} for model (1) using antithetic variates for all methods except (t, m, s)–nets . The qualitative results do not change when employing original sequences.[9] However, the biases for the other quasi–Monte Carlo methods become smaller without antithetic variates, in particular for a small number of replications. For high negative autocorrelation HALTON sequences outperform the other methods, while for

[9]Results are available on request.

small negative autocorrelation FAURE and SOBOL sequences seem competitive. Only for no autocorrelation of the data generating process, (t, m, s)–nets clearly outperform the other methods, while for positive autocorrelation FAURE sequences result in the smallest bias of the simulated variance.

Because of the strong dependency on the degree of autocorrelation, simulation results for the other models are reported for three parameter combinations corresponding to the cases of high negative autocorrelation (AC=-1), almost negligible autocorrelation (AC=0) and high positive autocorrelation (AC=1). The complete set of parameters for all 27 simulated models is provided in Tables 12 and 13 in the appendix. As the results for the different pseudo–random number generators do not differ much, only results for the ESSL generator are reported for models (2) – (9). Finally, for the non linear models, when analytical solutions for the true estimates are not available, reference values for expected value and variance are obtained by simulation. Therefore, SOBOL–sequences of increasing length are used until the change in these values when doubling the length becomes smaller than 10^{-5}.[10]

For models (1) – (6) (linear models), the bias of expectation is zero, when using antithetic variates. Therefore, the bias of simulated $\mathrm{E}(x_{10})$ is reported for models (7) – (9) only in Tables 6, 7 and 8. Furthermore, only results for $N = 4096$, $N = 16\,384$ and $N = 65\,536$ are reported.

Table 6: Bias (p.c.) of simulated $\mathrm{E}(x_{10})$ for Models (7)–(9), $N = 4096$

Model	AC	ESSL	FAURE	HALTON	SOBOL	TMS
7	-1	0.0640	-0.0412	-0.0105	-0.0175	0.1322
7	0	1.6698	0.4347	0.3782	-0.2699	0.0497
7	1	0.2461	-0.4441	-0.7357	-0.2401	0.6317
8	-1	0.3990	0.0315	-0.0538	0.2870	1.1276
8	0	1.0199	0.0062	-0.0289	0.0702	0.0497
8	1	0.5591	0.4045	-0.0866	-0.2812	-0.5542
9	-1	0.0375	0.0153	0.0126	-0.0060	0.0267
9	0	0.0160	0.0256	0.0318	0.0059	-0.0079
9	1	0.1246	-0.2850	-0.2208	0.0283	-0.0752

[10]In order to obtain this high accuracy up to several 100 million replications are required depending on the model and the parameter set, while, typically, some million replications are sufficient.

It is clear from the numerical experiment, for $N = 4096$ the bias of simulated $\mathrm{E}(x_1 0)$ is smaller than 1 percent for almost all instances and point sets. Since it differs considerably across methods for a given model, it is impossible to derive some general conclusion. However, we observe from Table 6, that for $N = 4096$ the SOBOL–sequence always results in a bias smaller than 0.3 percent, while this upper bound is 0.45 percent for FAURE, 0.75 for HALTON, 1.15 for TMSand 1.7 for the mean of 10 replications of the Monte Carlo sequence. Since the results of the Monte Carlo simulation are stochastic, biases of the order of magnitude of more than 1 percent may appear quite frequently in this setting. It should be noted that simulated effects in a macroeconometric simulation setup are often smaller than one percent. Consequently, the bias of Monte Carlo simulation can be larger than the simulated effect! The SOBOL–sequences seem to be most robust in avoiding large biases, which is confirmed by the findings for $N = 16\,384$ and $N = 65\,536$ with a single exception for model (8), $AC = 1$ and $N = 16\,384$. Again, the (t, m, s)–net sometimes provides high quality approximations, for the just mentioned instance even the overall best approximation, but fails for other instances resulting in biases even larger than the mean for the pseudo–random number generator. This puzzling result requires further analysis of the implementation of (t, m, s)–nets to the simulation of this kind of time series processes.

Table 7: Bias (p.c.) of simulated $\mathrm{E}(x_{10})$ for Models (7)–(9), $N = 16\,384$

Model	AC	ESSL	FAURE	HALTON	SOBOL	TMS
7	-1	0.0168	0.0141	0.0154	-0.0137	0.0330
7	0	0.6915	0.0932	0.0007	-0.0883	0.1062
7	1	0.0936	-0.1616	-0.2463	-0.0138	0.3392
8	-1	0.2033	-0.1071	0.0795	0.2360	-0.0359
8	0	0.5587	0.0022	-0.0397	-0.0743	0.0724
8	1	0.4347	0.1517	-0.1858	-0.0037	-0.0259
9	-1	0.0178	0.0025	0.0023	0.0006	0.0074
9	0	0.0084	0.0093	0.0114	-0.0011	-0.0055
9	1	0.0578	-0.0300	-0.0040	-0.0022	0.0628

Tables 9, 10 and 11 show the results of the numerical experiments for the bias of simulated variance of x_{10} or z_{10}, respectively. Those results are based

Table 8: Bias (p.c.) of simulated $\mathrm{E}(x_{10})$ for Models (7)–(9), $N = 65\,536$

Model	AC	ESSL	FAURE	HALTON	SOBOL	TMS
7	-1	0.0202	-0.0012	-0.0045	-0.0180	-0.0049
7	0	0.2621	0.0053	0.0563	-0.0434	-0.0374
7	1	0.0589	-0.0331	-0.0373	-0.0262	0.1186
8	-1	0.1231	0.0003	-0.0199	0.0090	0.0453
8	0	0.1158	-0.0065	0.0051	-0.0319	-0.0532
8	1	0.1675	-0.0300	-0.0466	-0.0483	0.1146
9	-1	0.0054	0.0001	-0.0003	-0.0002	0.0006
9	0	0.0029	0.0027	0.0032	-0.0004	-0.0096
9	1	0.0193	0.0075	0.0065	0.0021	0.0623

on a larger set of instances, as the use of antithetic variates does not preclude a bias of the simulated variance. In fact, the results for ESSLindicates a high and persistent bias of a order of magnitude of 2 to 3 percent for $N = 4096$, which decreases only slowly to about 1 to 1.5 percent for $N = 16\,384$ and 0.5 to 0.75 percent for $N = 65\,536$ corresponding to the slow convergence rate of Monte Carlo methods.

Comparing among the quasi–Monte Carlo methods, again no clear ranking is provided. However, as the number of replications is increased, SOBOL–sequences obtain the best approximation in more than half of all cases, while the (t, m, s)–net obtains the best approximation only in about 20 percent of cases for $N \leq 16\,384$ and only twice for $N = 65\,536$. The higher convergence rate of quasi–Monte Carlo methods is also mirrored by the simulation results, as the typical bias of the variance estimate for the SOBOL–sequences decreases from 0.3 to 0.6 percent for $N = 4096$ to 0.02 to 0.06 percent for $N = 65\,536$.

5 Conclusion

In this paper, the use of quasi–Monte Carlo methods for the purpose of simulating time series processes is analyzed. Based on theoretical results on the discrepancy and integration error bounds, different quasi–Monte Carlo methods are compared to standard pseudo–random number generators. In partic-

Table 9: Bias (p.c.) of simulated variance of x_{10} for Models (1)–(9), $N = 4096$

Model	AC	ESSL	Faure	Halton	Sobol	TMS
1	-1	2.7944	-0.4764	0.1896	-0.4088	-0.5622
1	0	2.7065	-0.6616	-0.2256	0.1367	-0.0069
1	1	2.8995	-2.1174	-2.7766	0.9096	-0.7189
2	-1	2.8032	-0.2243	-0.3940	-0.3516	0.0854
2	0	2.6641	-0.7424	-0.6673	0.4079	0.1208
2	1	3.0373	-2.0949	-2.0193	0.5902	-0.7501
3	-1	2.5023	-1.4554	-0.0608	-0.3141	-0.2188
3	0	2.8204	-0.1021	-0.0163	-0.2141	-0.2052
3	1	2.4063	-2.1100	-2.7776	0.9185	-0.8170
4	-1	2.5210	-0.1623	-0.2787	0.3218	0.7929
4	0	2.9123	-0.2537	-0.3728	0.2819	0.3145
4	1	2.1302	-0.7898	-0.7596	1.3037	1.3810
5	-1	1.9437	-1.0747	-1.7976	0.0816	-0.4028
5	0	2.8738	0.1567	-0.7299	0.2325	-0.0390
5	1	2.5935	-2.2214	-2.5135	0.9678	1.4545
6	-1	2.9619	-1.3924	-1.2981	0.1421	-1.5619
6	0	1.7689	-1.8637	-1.8625	0.6184	-0.6617
6	1	2.9302	-1.9023	-1.9872	-0.0194	-2.7328
7	-1	2.4558	-0.0108	-0.2348	-0.3215	-0.5198
7	0	2.3938	-0.5135	-0.3293	0.0109	-0.0099
7	1	2.0748	0.0279	-0.0609	0.1218	0.2749
8	-1	2.0178	1.5756	-0.0097	-0.0473	0.0509
8	0	1.0777	0.1009	0.0581	-0.7772	-0.2259
8	1	1.8841	-1.7413	-0.5016	0.5591	1.0325
9	-1	2.6707	-0.8669	-0.6990	0.2843	-1.4769
9	0	2.7834	-2.8761	-3.8193	-0.4740	-0.9318
9	1	2.2133	0.4736	0.1313	-0.1828	-3.0125

Table 10: Bias (p.c.) of simulated variance of x_{10} for Models (1)–(9), $N =$ 16 384

Model	AC	ESSL	FAURE	HALTON	SOBOL	TMS
1	-1	0.7462	-0.7415	-0.0133	-0.2385	0.0564
1	0	0.7390	-0.1862	-0.1944	-0.1681	-0.0012
1	1	1.3374	-0.1941	-1.1119	0.0823	0.5979
2	-1	0.6537	-0.0922	-0.0881	-0.2609	0.1592
2	0	0.7441	-0.2050	-0.2109	-0.1957	0.0140
2	1	1.1397	-0.6124	-0.5988	0.0651	0.2778
3	-1	1.1345	-1.4008	-0.0858	-0.1101	-0.2342
3	0	1.3188	-0.4005	0.0080	-0.2745	0.2667
3	1	1.0918	-0.1932	-1.1160	0.0699	0.5860
4	-1	1.3799	-0.0790	-0.0353	-0.1380	0.2750
4	0	1.5713	-0.1333	-0.0915	-0.1500	0.4165
4	1	1.0020	-0.1802	-0.1946	-0.0463	-0.0260
5	-1	0.7699	-0.4100	-0.5203	-0.1328	0.0751
5	0	1.0681	1.2140	-0.5749	-0.0575	0.2207
5	1	0.9761	-0.5779	-0.8773	0.1437	0.4211
6	-1	1.5863	-0.4540	-0.4479	0.0346	0.5390
6	0	0.9082	-0.5431	-0.5339	0.0189	0.2639
6	1	1.5390	-0.6099	-0.5841	0.1899	0.2885
7	-1	1.0582	-0.0675	-0.1829	-0.2564	0.0070
7	0	1.1878	-0.0742	-0.1980	-0.2844	0.1307
7	1	1.4323	0.2132	-0.2731	0.1257	0.7701
8	-1	0.5488	1.4788	-1.1372	-0.0829	0.5663
8	0	0.6057	0.2424	-0.1896	-0.2358	0.0861
8	1	1.0049	-0.0213	0.3270	0.3963	-0.0940
9	-1	1.1215	-0.2163	-0.1980	-0.0081	0.1600
9	0	1.5695	-1.2004	-1.4697	0.2222	0.8150
9	1	0.9568	-0.1876	-0.3028	0.2036	-0.3860

Table 11: Bias (p.c.) of simulated variance of x_{10} for Models (1)–(9), $N =$ 65 536

Model	AC	ESSL	Faure	Halton	Sobol	TMS
1	-1	0.6406	0.0447	-0.0216	0.0291	0.0307
1	0	0.3248	-0.0574	-0.0489	0.0249	0.0002
1	1	0.5642	-0.2238	-0.3705	-0.0719	0.2294
2	-1	0.4866	-0.0398	-0.0237	-0.0523	-0.2566
2	0	0.3073	-0.0626	-0.0523	-0.0317	-0.0089
2	1	0.4159	-0.1764	-0.1980	-0.0138	0.3009
3	-1	0.5937	0.0307	-0.0430	0.0456	0.1257
3	0	0.7290	0.0351	-0.0134	0.0077	-0.0414
3	1	0.5063	-0.2223	-0.3681	-0.0684	0.2431
4	-1	0.9000	-0.0458	-0.0168	-0.0464	-0.2188
4	0	0.8512	-0.0587	-0.0286	-0.0462	-0.1984
4	1	0.5283	-0.0434	-0.0687	-0.0488	-0.1970
5	-1	0.5683	-0.0409	-0.1865	0.0153	0.1487
5	0	0.3823	-0.1079	-0.1522	-0.0719	-0.0829
5	1	0.4603	-0.1981	-0.3198	-0.0713	0.1010
6	-1	0.4780	-0.1380	-0.1322	-0.0166	0.3662
6	0	0.6381	-0.1587	-0.1777	-0.0155	0.2494
6	1	0.7500	-0.1524	-0.1933	0.0606	0.6797
7	-1	0.5035	-0.0296	-0.0117	0.0259	-0.2425
7	0	0.2954	-0.0802	-0.0136	-0.0532	-0.0981
7	1	0.5986	-0.2285	-0.2099	0.0366	0.4385
8	-1	0.4029	0.2797	-0.3062	-0.3630	0.0904
8	0	0.1983	-0.0462	0.0532	0.0213	-0.3812
8	1	0.4418	-0.0346	0.0885	0.2135	0.0917
9	-1	0.4034	-0.0505	-0.0310	-0.0115	0.2127
9	0	0.8070	-0.3937	-0.4912	0.0954	1.3545
9	1	0.7233	-0.1114	-0.1502	0.0477	0.1447

ular, the use of (t, m, s)–nets is motivated on a strong theoretical background. However, the centred L_2–discrepancy does not indicate a clear ranking among the quasi–Monte Carlo methods, nevertheless they all behave much better than Monte Carlo methods.

The application of Monte Carlo and quasi–Monte Carlo methods to the simulation of linear and non linear time series models provides further evidence of the superiority of quasi–Monte Carlo methods. In the case of linear models, the bias of simulated expectation can be reduced to zero by the use of antithetic variates. However, (t, m, s)–nets provide unbiased results in this case without using antithetic variates. For the bias of expectation for the non linear models and the bias of the simulated variance, the quasi–Monte Carlo methods are clearly superior to the Monte Carlo approaches. However, again there is no clear ranking of the quasi–Monte Carlo methods. A superiority of (t, m, s)–nets as could have been expected on theoretical grounds is not found.[11] Here, it is rather SOBOL–sequences, which provide the overall best approximations.

We only concentrate on the simulated distribution of x_{10}, y_{10} and z_{10}, respectively, in this paper. However, further simulation might consider extending the simulation period and, in due course, the dimensionality of the error space. Besides, investigation of the improvement of (t, m, s)–nets is important, as one of the advantage of (t, m, s)–nets is that with "nice" tweaking the uniformality can be improved while all the good theoretical properties remain the same. Such testing is now underway.

[11]Hellekalek (1998, pp. 68f) reports on results from a comparison of good lattice point sets and (t, m, s)–nets for numerical integration. He also finds that despite of the nice theoretical properties of (t, m, s)–nets , no superiority can be detected, quite to the contrary.

A Parameters of Simulated Models

Table 12: Parameters of Simulated Models – I –

Model	AC	Parameters
(1)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.9, \sigma_\varepsilon = 0.2$
	0	$\alpha_0 = 0.1, \alpha_1 = 0, \sigma_\varepsilon = 0.2$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.9, \sigma_\varepsilon = 0.2$
(2)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.8, \alpha_2 = -0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05$
	0	$\alpha_0 = 0.1, \alpha_1 = 0.1, \alpha_2 = -0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.8, \alpha_2 = 0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05$
(3)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.8, \alpha_2 = -0.1, \sigma_\varepsilon = 0.2$
	0	$\alpha_0 = 0.1, \alpha_1 = -0.5, \alpha_2 = 0.5, \sigma_\varepsilon = 0.2$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.8, \alpha_2 = 0.1, \sigma_\varepsilon = 0.2$
(4)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.8, \alpha_2 = -0.1, \alpha_3 = 0.5, \alpha_4 = 0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.3, \beta_2 = 0.2, \beta_3 = -0.5, \beta_4 = -0.3, \sigma_{\varepsilon_2} = 0.05$
	0	$\alpha_0 = 0.1, \alpha_1 = -0.5, \alpha_2 = 0.4, \alpha_3 = 0.5, \alpha_4 = 0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = 0.5, \beta_2 = -0.1, \beta_3 = -0.2, \beta_4 = 0.3, \sigma_{\varepsilon_2} = 0.05$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.8, \alpha_2 = 0.1, \alpha_3 = -0.5, \alpha_4 = -0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = -0.1, \beta_1 = 0.3, \beta_2 = -0.2, \beta_3 = 0.5, \beta_4 = 0.3, \sigma_{\varepsilon_2} = 0.05$
(5)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.9, \alpha_2 = 0, \lambda = -0.1, \beta = 0.5, \sigma\varepsilon = 0.2$
	0	$\alpha_0 = 0.1, \alpha_1 = 0.1, \alpha_2 = 0, \lambda = -0.4, \beta = 1.0, \sigma\varepsilon = 0.2$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.9, \alpha_2 = 0, \lambda = -0.1, \beta = 0.5, \sigma\varepsilon = 0.2$
(6)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.8, \alpha_2 = -0.1, \lambda_1 = -0.1, \gamma = 0.5, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.1, \beta_2 = -0.8, \lambda_2 = 0.2, \sigma_{\varepsilon_2} = 0.05$
	0	$\alpha_0 = 0.1, \alpha_1 = 0.2, \alpha_2 = -0.2, \lambda_1 = -0.5, \gamma = 1.0, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.2, \lambda_2 = 0.2, \sigma_{\varepsilon_2} = 0.05$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.8, \alpha_2 = 0.1, \lambda_1 = -0.1, \gamma = 0.5, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.1, \beta_2 = -0.8, \lambda_2 = 0.2, \sigma_{\varepsilon_2} = 0.05$

Table 13: Parameters of Simulated Models – II –

Model	AC	Parameters
(7)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.8, \alpha_2 = -0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05$
	0	$\alpha_0 = 0.1, \alpha_1 = 0.1, \alpha_2 = -0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.8, \alpha_2 = 0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05$
(8)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.8, \alpha_2 = -0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05, \rho = 10$
	0	$\alpha_0 = 0.1, \alpha_1 = 0.1, \alpha_2 = -0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05, \rho = 10$
	1	$\alpha_0 = 0.1, \alpha_1 = 0.8, \alpha_2 = 0.1, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.6, \sigma_{\varepsilon_2} = 0.05, \rho = 10$
(9)	-1	$\alpha_0 = 0.1, \alpha_1 = -0.8, \alpha_2 = -0.1, \lambda_1 = -0.2, \gamma = 0.5, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = -0.7, \lambda_2 = 0.05, \sigma_{\varepsilon_2} = 0.05, \rho = 10$
	0	$\alpha_0 = 0.1, \alpha_1 = -0.1, \alpha_2 = 0.2, \lambda_1 = -0.1, \gamma = 0.2, \sigma_{\varepsilon_1} = 0.1$
		$\beta_0 = 0.1, \beta_1 = -0.2, \beta_2 = 0.1, \lambda_2 = 0.05, \sigma_{\varepsilon_2} = 0.05, \rho = 10$
		$\alpha_0 = 0.1, \alpha_1 = 0.8, \alpha_2 = 0.1, \lambda_1 = -0.2, \gamma = 0.5, \sigma_{\varepsilon_1} = 0.2$
		$\beta_0 = 0.1, \beta_1 = 0.2, \beta_2 = 0.7, \lambda_2 = 0.05, \sigma_{\varepsilon_2} = 0.05, \rho = 10$

References

Acworth, P., M. Broadie and P. Glasserman (1998). A comparison of some monte carlo and quasi-monte carlo methods 1996. In: *Lecture Notes in Statistics* (H. Niederreiter, P. Hellekalek, G. Larcher and P. Zinterhof, Eds.). Vol. 127. pp. 1–18.

Bierbrauer, J. and Y. Edel (1999). *Some good binary (t,m,s)–nets.* Preprint.

Bratley, P. and B. L. Fox (1988). Sobol's quasirandom sequence generator for multivariate quadrature and optimization. *ACM TOMS* **14**(1), 88–100.

Clayman, A. T., K. M. Lawrence, G. L. Mullen, H. Niederreiter and N. J. A. Sloane (1999). Updated tables of parameters of (t, m, s) – nets. *Journal of Combinatorial Designs.*

Ericcsson, N. R. and J. Marquez (1998). A framework for economic forecasting. *Econometrics Journal* **1**, C228–C266.

Fang, K.-T., C.-X. Ma and P. Winker (2000). Centered l_2–discrepancy of random sampling and latin hypercube design, and construction of uniform designs. *Mathematical Computation* p. forthcomming.

Franz, W., K. Göggelmann, M. Schellhorn and P. Winker (2000). Quasi-monte carlo methods in stochastic simulations. *Empirical Economics* p. forthcoming.

Gentle, J. E. (1998). *Random Number Generation and Monte Carlo Methods.* Springer. New York.

Hellekalek, P. (1998). On the assessment of random and quasi–random point sets. In: *Random and Quasi-Random Point Sets. Lecture Notes in Statistics* (P. Hellekalek and G. Larcher, Eds.). Vol. 138. Springer. New York. pp. 49–108.

Hickernell, F. J. (1998). A generalized discrepancy and quadrature error bound. *Mathematics of Computation* **67**, 299–322.

Larcher and Traunfellner (1994). ???. *Math. Comp.* **63**, 277–291.

L'Ecuyer, P. and P. Hellekalek (1998). Random number generators: Selection criteria and testing. In: *Random and Quasi-Random Point Sets. Lecture Notes in Statistics* (P. Hellekalek and G. Larcher, Eds.). Vol. 138. Springer. New York. pp. 223–265.

Li, J. and G. Mullen (2000). Parallel computing of a quasi–monte carlo algorithm for valuting derivatives. *Parallel Computing* **26**(5), 641–653.

Niederreiter, H. (1992). *Random Number Generation and Quasi-Monte Carlo Methods.* SIAM. Philadelphia, PE.

Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery (1992). *Numerical Recipes in Fortran 77.* 2nd ed.. Cambridge University Press. New York, NY.

Ripley, B. D. (1987). *Stochastic Simulation.* Wiley. New York.

Winker, P. (1999). Quasi-monte carlo policy simulations in a macroeconometric disequilibrium model. In: *Proceedings of the 14th World Congress International Federation of Automatic Control.* Elsevier. Oxford. pp. 45–50.

Winker, P. and K.-T. Fang (1997). Application of threshold accepting to the evaluation of the discrepancy of a set of points. *SIAM Journal on Numerical Analysis* **34**(5), 2028–2042.