

Tilburg University

Supercomputers, Monte Carlo simulation and regression analysis

Kleijnen, J.P.C.; Annink, B.

Publication date:
1989

[Link to publication in Tilburg University Research Portal](#)

Citation for published version (APA):

Kleijnen, J. P. C., & Annink, B. (1989). *Supercomputers, Monte Carlo simulation and regression analysis*. (Research memorandum / Tilburg University, Department of Economics; Vol. FEW 402). Unknown Publisher.

General rights

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

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

Take down policy

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

CBM
R



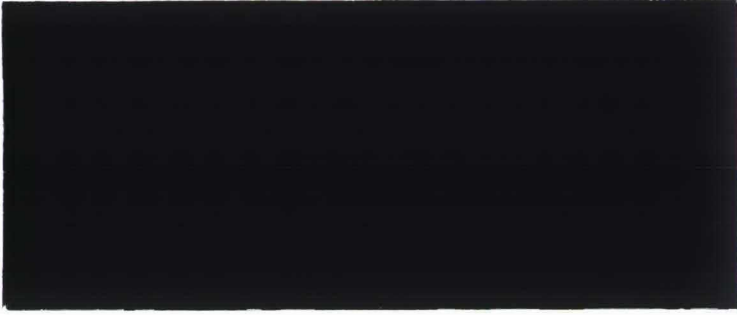
UNIVERSITEIT
BRABANT

POSTBOX 90153
5000 LE TILBURG
THE NETHERLANDS

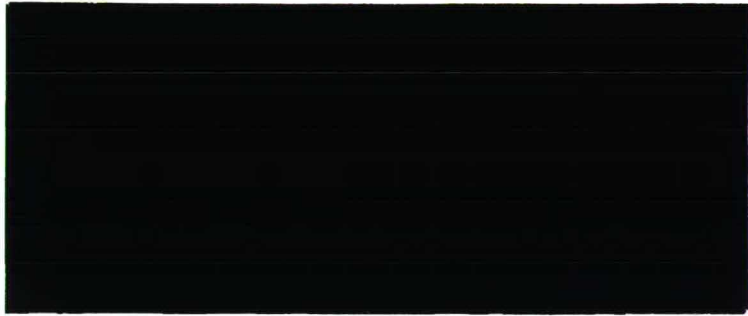



* C I N O 1 1 6 1 *

7626
1989
402



DEPARTMENT OF ECONOMICS
RESEARCH MEMORANDUM



 K.U.B.
BIBLIOTHEEK
TILBURG

**SUPERCOMPUTERS, MONTE CARLO SIMULATION
AND REGRESSION ANALYSIS**

Jack P.C. Kleijnen
Ben Annink

FEW 402

1984
1985
1986

SUPERCOMPUTERS, MONTE CARLO SIMULATION,
AND REGRESSION ANALYSIS

Jack P.C. Kleijnen
and
Ben Annink

August 1989

Correspondence should be directed to: Prof. J.P.C. Kleijnen, Department of Information Systems and Auditing, School of Business and Economics, Catholic University Brabant (Katholieke Uiversiteit Brabant), 5000 LE Tilburg, Netherlands. FAX: 013-663019. E-mail: t435klei@htikub5.

SUPERCOMPUTERS, MONTE CARLO SIMULATION, AND REGRESSION ANALYSIS

Jack P.C. Kleijnen

and

Ben Annink

Department of Information Systems and Auditing, School of Business and Economics, Catholic University Brabant (Katholieke Universiteit Brabant), 5000 LE Tilburg, Netherlands. E-mail: t435klei@htikub5.

Supercomputers provide a new tool for management scientists. The application of this tool requires thinking in parallel or vector mode. This mode is examined in the context of Monte Carlo simulation experiments with multivariate regression models. The parallel mode needs to exploit a specific dimension of the Monte Carlo experiment (namely the replicates of that experiment). Then Ordinary Least Squares on a CYBER 205 takes only 1.4% of the time needed on a VAX 8700. Estimated Generalized Least Squares, however, is slower on the CYBER 205 because it requires matrix inversion.

(ADDITIONAL KEYWORDS: DISTRIBUTION SAMPLING; MULTIVARIATE DISTRIBUTION; COMMON SEEDS; METAMODEL)

1. Introduction

This paper focuses on the use of supercomputers in Monte Carlo experiments with multivariate regression analysis. Regression models can be used to determine a metamodel of a simulation model; see Kleijnen (1987, 1988). So this paper may be of interest to management scientists for several reasons:

(i) Regression analysis is often used by management scientists to analyse simulation data and real-world data.

(ii) The study shows how supercomputers can be applied in Monte Carlo experiments, which are related to stochastic simulation: both use pseudo-random numbers, but simulation is dynamic (a case in point is queuing simulation) whereas Monte Carlo experiments are not; see Teichrow (1965). So Monte Carlo experiments are simpler. Our study may challenge other researchers to apply supercomputers to Monte Carlo and simulation models.

A new development in management science is the advent of *supercomputers* or vector computers such as the CYBER 205. Their pipelined architecture distinguishes these computers from traditional scalar computers (for example, the IBM 370 and the VAX series) and from truly parallel computers (such as the HYPERCUBE). The challenge now is to think in the "parallel" mode; that is, the problem is to formulate mathematical models such that the vector mode of the supercomputer can be applied. Parallel thinking can be exemplified by the innerproduct of two matrices $\tilde{v}_1 \tilde{v}_2 = \sum_{j=1}^n v_{1j} v_{2j}$. This computation requires n scalar operations $v_{1j} v_{2j}$; these n multiplications can be done in parallel; that is, to compute the product $v_{1j} v_{2j}$ the computer does not need $v_{1(j-1)} v_{2(j-1)}$. The pipeline architecture means that a supercomputer works as an assembly line: efficiency improves drastically if a large number of identical operations can be executed independently of each other; see Levine (1982).

Our paper is organized as follows. In § 2 we summarize the well-known multivariate linear regression model and its application in simulation experiments with common pseudorandom numbers. This regression model is to be studied, using Monte Carlo experimentation. In § 3 we show how the Monte Carlo program can be "vectorized" so that it can be run in parallel; we discover a "third dimension" of Monte Carlo experiments. § 4 gives conclusions. References and appendices complete the paper.

2. Multivariate Regression Models and Simulation

Consider the well-known linear regression model

$$E(\underline{y}) = \underline{X} \underline{\beta} \quad (2.1)$$

with $\underline{y} = (y_1, \dots, y_i, \dots, y_n)'$, $\underline{\beta} = (\beta_1, \dots, \beta_j, \dots, \beta_Q)'$ and $\underline{X} = (x_{ij})$ where $i = 1, \dots, n$ and $j = 1, \dots, Q$. This model is multivariate if the errors $\underline{e} = (e_1, \dots, e_i, \dots, e_n)'$ are mutually dependent (the errors are also called disturbance or noise). We assume additive errors:

$$\underline{y} = \underline{X} \underline{\beta} + \underline{e} \quad (2.2)$$

We further assume that \underline{e} is multivariate normally (MN) distributed:

$$\underline{e} \in MN(\underline{\mu}_e, \underline{\Omega}_e) \quad (2.3)$$

where $\underline{\mu}_e = \underline{0}$ (a column of n zero's) and $\underline{\Omega}_e$ equals $\underline{\Omega}_y$ because of (2.2); $\underline{\Omega}_y$ is assumed to be non-singular. When this model is applied to simulation data, (2.1) is called the metamodel (the regression equation is a model of the simulation computer program; see Kleijnen, 1987); in (2.3) $\underline{\Omega}_y$ is non-diagonal if common seeds are used for the pseudorandom number generator of the simulation model; see Kleijnen (1988).

We consider experimental design situations only: we assume that the independent regression variables \underline{X} in (2.1) follow from an experimental design for k factors: $\underline{D} = (d_{ih})$ with $h = 1, \dots, k$. For example for $i = 1, \dots, n$ we may have $x_{i1} = 1$ (dummy), $x_{i2} = d_{i1}$ (factor 1), $x_{i3} = \log d_{i2}$ (factor 2 on log scale), $x_{i4} = d_{i1}d_{i3}$ (interaction between factor 1 and 3). Counterexamples are provided by econometrics where \underline{X} can be observed only, not controlled; see Kleijnen (1987, p. 159). Moreover, in well designed experiments it is possible to replicate specific factor combinations; that is, row i of \underline{X} or $\underline{x}'_i = (x_{i1}, \dots, x_{ij}, \dots, x_{iQ})$ can be observed $m_i \geq 2$ times. For example, in simulation the combination i of the simulation parameters d_h is run m_i times (a terminating simulation is repeated with m_i independent pseudorandom number streams; in non-terminating or steady-state simulations m_i subruns are obtained; see Kleijnen, 1987, pp.

8-10, 63-83). If all combinations of simulation parameters use the same seed for the pseudorandom number generator, then obviously m_i becomes a constant m . Outside a simulation context, Rao (1959) assumes m independent observations on the n -variate vector y . His assumption agrees with the simulation context of Table 1, which assumes independent seeds. These observations yield the following unbiased estimators of $\sigma_{ii'} = \text{cov}(y_i, y_{i'}) = \text{cov}(y_{ir}, y_{i'r})$:

$$\hat{\sigma}_{ii'} = \frac{\sum_{r=1}^m (y_{ir} - \bar{y}_i)(y_{i'r} - \bar{y}_{i'})}{m-1} \quad (i, i' = 1, \dots, n) (m \geq 2) \quad (2.4)$$

with the averages $\bar{y}_i = \sum_{r=1}^m y_{ir}/m$; by definition we have $\sigma_{ii} = \sigma_i^2$. In matrix notation (2.4) becomes

$$\hat{\hat{Q}}_y = \tilde{Y} \tilde{Y}' / (m-1) - \bar{y} \bar{y}' m \quad (2.5)$$

with $\hat{\hat{Q}}_y = (\hat{\sigma}_{ii'})$, $\tilde{Y} = (y_{ir})$ and $\bar{y} = (\bar{y}_i)$.

TABLE 1
Regression Data

Combination i (effects: $\beta_1 \dots \beta_j \dots \beta_Q$)	Responses y_{ir} (seed 1) ... (seed r) ... (seed m)	Average response \bar{y}_i	Estimated (co)variances $\hat{\sigma}_{ii'}$
$x_{11} \dots x_{1j} \dots x_{1Q}$ $x_{21} \dots x_{2j} \dots x_{2Q}$	$y_{11} \quad \dots \quad y_{1r} \quad \dots \quad y_{1m}$ $y_{21} \quad \dots \quad y_{2r} \quad \dots \quad y_{2m}$	\bar{y}_1 \bar{y}_2	$\hat{\sigma}_1^2 \hat{\sigma}_{12} \dots \hat{\sigma}_{1n}$ $\hat{\sigma}_2^2 \dots \hat{\sigma}_{2n}$
$x_{i1} \dots x_{ij} \dots x_{iQ}$	$y_{i1} \quad \dots \quad y_{ir} \quad \dots \quad y_{im}$	\bar{y}_i	$\hat{\sigma}_i^2 \dots \hat{\sigma}_{in}$
$x_{n1} \dots x_{nj} \dots x_{nQ}$	$y_{n1} \quad \dots \quad y_{nr} \quad \dots \quad y_{nm}$	\bar{y}_n	σ_n^2

Kleijnen (1988, p.67) proposes two different point estimators for the regression parameters β . The first estimator uses Ordinary Least Squares or OLS:

$$\hat{\beta} = (\tilde{X}'\tilde{X})^{-1}\tilde{X}'\tilde{y}, \quad (2.6)$$

which assumes $n > Q$. The second estimator uses Estimated Generalized Least Squares or EGLS:

$$\hat{\beta} = (\tilde{X}'\hat{\Omega}^{-1}\tilde{X})^{-1}\tilde{X}'\hat{\Omega}^{-1}\tilde{y}, \quad (2.7)$$

which assumes that $\hat{\Omega}_y$ is non-singular; also see (2.3). The estimated covariance matrices of these two estimators are

$$\hat{\Omega}_{\hat{\beta}} = (\tilde{X}'\tilde{X})^{-1}\tilde{X}'\hat{\Omega}_y\tilde{X}(\tilde{X}'\tilde{X})^{-1}/m \quad (2.8)$$

and

$$\hat{\Omega}_{\hat{\beta}} \approx (\tilde{X}'\hat{\Omega}_y^{-1}\tilde{X})^{-1}/m \quad (2.9)$$

where the symbol \approx means that the equality holds only asymptotically. Obviously we have

$$\hat{\Omega}_y^{-1} = \hat{\Omega}_y/m \quad (2.10)$$

Monte Carlo experimentation enables us to study the statistical behavior of different regression procedures. For example, we may estimate the α and β errors of a test devised to detect a misspecified regression model; see Kleijnen (1988, p. 71). In this paper, however, we focus on the computer generation of the observations $Y = (y_{ir})$ and the estimators $\hat{\beta}$ and $\hat{\beta}$; see (2.2) through (2.7). The computation of other statistics is then straightforward. The Monte Carlo experiment is replicated L times, say $L = 100$ (taking $L = 100$ means that estimated α and β errors have standard errors smaller than 0.05, since $\hat{\alpha}$ and $\hat{\beta}$ are binomially distributed).

3. Parallel Design of the Monte Carlo Program

In § 1 we emphasized that supercomputers work efficiently only if many parallel operations can be identified. An individual element y_{ir} in Table 1, defined by (2.2), can be computed in vector mode, but this mode is inefficient since typical values for n and Q are as small as 4 and 3. Alternatively, we may consider the parallel computation of either n rows (factor combinations) or m columns (independent observations or seeds). Let us first consider these two dimensions and then a third dimension.

The errors within a column are statistically dependent: they are n -variate normal. We first discuss programming for $n = 2$. We then sample the independent univariate standard normal variates z_1 and z_2 , and compute the linear transformation $e_1 = \sigma_1 z_1$ and $e_2 = \sigma_2(\rho z_1 + (1-\rho^2)^{\frac{1}{2}} z_2)$. For general n the sampling subroutine for multivariate normal \underline{e} with covariance matrix $\underline{\Omega}_{\underline{y}}$ is

$$\underline{e} = \underline{C} \underline{z}, \quad (3.1)$$

where $\underline{z} = (z_1, \dots, z_1, \dots, z_n)'$ with independent $z_i \in N(0,1)$, and with \underline{C} a lowertriangular matrix defined by

$$\underline{C} \underline{C}' = \underline{\Omega}_{\underline{y}}, \quad (3.2)$$

which is computed by Choleski's technique; see Naylor et al. (1966, pp. 97-99) and standard software libraries such as IMSL and NAG. But (3.1) defines a *recursive* relation, and such relations are not efficiently handled by supercomputers. So we do not compute this dimension in parallel.

The columns in Table 1 are statistically independent, by definition (see the text above (2.4)). Hence a supercomputer can calculate these m observations in parallel. But it is well-known that supercomputers become efficient only if the number of parallel operations is "large", say, $m \geq 50$. In the Monte Carlo experiment we wish to study m equal to 2, 10, 25 and 50. So in most cases, parallel computation would be slower than scalar computation; also see Levine (1982) and SARA (1984).

But there is a *third dimension* in this problem! The Monte Carlo experiment is repeated 100 times (see § 2). We speak of "MC replicates" &

with $\ell = 1, \dots, L$ and $L = 100$ (which must be distinguished from the replicates $r = 1, \dots, m$). These replicates are statistically independent and can be computed in parallel, as we shall see. The more replicates we wish to obtain (higher L), the more efficient the vector computer becomes. We may visualize our problem as follows. There is a three-dimensional box to be filled in parallel with errors $e_{i r \ell}$ with $i = 1, \dots, n$; $r = 1, \dots, m$; $\ell = 1, \dots, L$. This box is filled in steps 1 through 3 below. In step 4 statistics such as $\hat{\Omega}_{\tilde{y}}$ are computed.

Step 1: Sample pseudorandom numbers x in parallel

Kleijnen (1989) evaluates several procedures for the parallel generation of pseudorandom number $x \in U(0,1)$. Kleijnen and Annink (1989) recommend the following standard scalar generator. Take a multiplicative congruential generator, since the statistical properties of such a generator are well-known. To initialize the parallel version of this generator, first generate - in vector mode - a vector of J successive pseudorandom integers $\underline{x} = (x_0, x_1, x_2, \dots, x_{J-2}, x_{J-1})$ with seed x_0 and $x_j = (a x_{j-1}) \bmod m$ for $j = 1, 2, \dots, J-1$. Once and for all compute a scalar multiplier: $(a^J) \bmod m$. Multiplication of the vector \underline{x} with this scalar multiplier gives a new vector: $(x_J, x_{J+1}, \dots, x_{2J-2}, x_{2J-1})'$. In this way the pseudorandom numbers are generated in exactly the same order as they would have been produced in scalar mode. At the end of the Monte Carlo experiment the vector of the last J numbers should be stored, so that the experiment may be continued later on.

In § 1 we mentioned that supercomputers become more efficient as the number of parallel operations increases. For the CYBER 205, however, there is a technical upper limit, since this computer uses 16 bits for addressing; see SARA (1984, p. 26). Therefore we take $J = 2^{16} - 1 = 65,535$.

There is a computational problem: overflow occurs when computing $(a^J) \bmod m$. This problem is solved, using controlled integer overflow and the CYBER 205's two's complement representation of negative integers. Appendix 1 gives the computer program.

Step 2: Sample independent standard normal variates z in parallel

There are several techniques for generating $z \in N(0,1)$; see Devroye (1986). We take a procedure that fits a vector computer:

$$z_1 = (-2 \ln x_1)^{\frac{1}{2}} \cos 2\pi x_2 \quad (3.3.a)$$

$$z_2 = (-2 \ln x_1)^{\frac{1}{2}} \sin 2\pi x_2, \quad (3.3.b)$$

where the mutually independent pair x_1 and x_2 yields the mutually independent pair z_1 and z_2 . To compute the functions \ln , \cos and \sin for a vector of numbers, we use FORTRAN 200's vector functions VLN, VCOS, and VSIN. So, given a vector of L independent pseudorandom numbers x , we use the first half to compute L/2 independent, parallel realizations of $\ln x_1$, and the second half to compute $\cos(2\pi x_2)$ and $\sin(2\pi x_2)$: Figure 1 gives a pseudo-FORTRAN program where π is computed through the arccosine; see SARA (1984, p. 13). To convert this pseudo-FORTRAN into a FORTRAN 200 program, we can replace DO loops by the special syntax of FORTRAN 200; the supercomputer can also automatically translate the FORTRAN program of Figure 1 (provided we add CONTINUE statements); see CDC (1986), SARA (1984, p. 17).

FIGURE 1

Parallel computation of L variates $z \in N(0,1)$.

```

L2 = L/2; PI = ACOS(-1.0); C = 2 * PI
DO 20 LL = 1, L2
20  HELP1(LL) = SQRT(-2 * LOG(X(LL)))
DO 30 LL = 1, L2
HELP2(LL) = COS(X(LL + L2) * C)
HELP3(LL) = SIN(X(LL + L2) * C)
Z(LL) = HELP1(LL) * HELP2(LL)
30  Z(L2 + LL) = HELP1(LL) * HELP3(LL)

```

Note that Petersen (1988) generates z in parallel, using not (3.3) but Teichroew's procedure described in Naylor et al. (1966, p. 94).

Above we saw that we wish to fill a three-dimensional "box" with $e_{i,r,l}$. So we store the vectors \underline{z} (with L elements) of Figure 1 into a three-dimensional array $Z(i,r,l)$.

Step 3: Sample n-variate \underline{e}

The error vector $\underline{e} = (e_1, \dots, e_1, \dots, e_n)'$ is multivariate normal with mean zero and covariance matrix $\underline{\Omega}_y$; see (2.3). To generate \underline{e} we linearly transform the n -variate vector of independent standard normal variates $\underline{z} = (z_1, \dots, z_n)'$; see (3.1). This transformation uses the lower triangular matrix C of (3.2). This yields

$$e_i = \sum_{j=1}^i c_{ij} z_j \quad (i=1, \dots, n). \quad (3.3)$$

To obtain M observations and L Monte Carlo replicates of \underline{e} , we might apply the naive FORTRAN program of Figure 2, where M denotes the maximum value of m in the experiment (here $M = 50$) and $E(I,R,LL)$ is zero initially. Note that \underline{C} or $C(I,J)$ does not vary over seeds (R) and Monte Carlo replicates (LL); it does vary over the Monte Carlo experiments defined by $\underline{\Omega}_y$.

FIGURE 2

Naive FORTRAN program for \underline{e} .

```

DO      10      LL = 1,L
      DO      10      R = 1,M
            DO      10      I = 1,N
                  DO      10      J = 1,I
10      E(I,R,LL) = E(I,R,LL) + C(I,J) * Z(J,R,LL)

```

To vectorize this naive program we should make the inner DO loop long; therefore we move the LL loop; moreover we should store the columns of the array columnwise; see SARA (1984, pp. 15, 20-21, 33). These two

guidelines yield Figure 3. (Note that the inner loop forms a so-called "linked triad"; hence it can be vectorized; see SARA, 1984, pp. 18-19.)

FIGURE 3

Vectorized FORTRAN program for \underline{e} .

```

DO 20 I = 1,N
  DO 20 J = 1,I
    DO 20 R = 1,M
      DO 20 LL = 1,L
20          E(LL,R,I) = E(LL,R,I) + C(I,J) * Z(LL,R,J)

```

We point out that m and n vary with the Monte Carlo experiments. So an experiment may use only part of the pseudorandom numbers stored in the "box" $E(LL,R,I)$. Implementing Figure 3 not only saves computer time, but it also runs experiments with common seeds.

Note that we generate $M*L$ (instead of L) elements in parallel, if we replace two loops - namely the loops for R and LL - in Figure 3 by a single loop - namely $LR = 1, \dots, M*L$ - which yields the two-dimensional array $E(LR,I)$. Then, however we have to rearrange this array into the three-dimensional array $E(LL,R,I)$ because the latter array is needed for the computation of statistics such as $\hat{\underline{Q}}_y$, as we see now.

Step 4: Compute statistics $\hat{\underline{Q}}_y$, $\hat{\underline{\beta}}$ and $\hat{\underline{\beta}}$

Once we have the three-dimensional array \underline{E} , we can easily compute estimates such as $\hat{\underline{Q}}_y$ defined in (2.5). This equation can also be computed as

$$\hat{\underline{Q}}_y = \underline{e} \underline{e}' / (m-1) - \bar{\underline{e}} \bar{\underline{e}}' m, \quad (3.4)$$

FIGURE 4

Vectorizable FORTRAN program for $\bar{\tilde{e}}$.

```

DENOM = 1.0/m
DO 10 I = 1,N
  DO 10 R = 1,M
    DO 10 LL = 1,L
10      EBAR(LL,I) = EBAR(LL,I) + E(LL,R,I)
      DO 20 I = 1,N
        DO 20 LL = 1,L
20          EBAR(LL,I) = EBAR(LL,I) * DNOM

```

where $\bar{\tilde{e}} = (\bar{e}_1, \dots, \bar{e}_1, \dots, \bar{e}_n)'$ with $\bar{e}_i = \sum_{r=1}^m e_{ir}/m$. Figure 4 shows the vectorizable FORTRAN program for the computation of $\bar{\tilde{e}}$. This program can be compiled and vectorized automatically. Alternatively we can use special FORTRAN 200 instructions such as Q8SSUM that computes sums like $\sum e_{ir}$. The computation of $\hat{\tilde{\Omega}}_y$ in (3.4) can be programmed analogous to Figure 4. Alternatively we can program innerproducts ($e'e$ and $\bar{e}\bar{e}'$) through the special function Q8SDOT; see SARA (1984, pp. 22,30).

A problem arises when computing the *inverse* $\hat{\tilde{\Omega}}_y^{-1}$, which is needed to compute the EGLS estimator $\hat{\tilde{\beta}}$ in (2.7). The trick in the preceding steps was to make the inner loop long; that is, we made the LL loop the inner loop. The instruction in that loop is executed in parallel, provided that instruction contains *no function or subroutine references* except for basic functions such as sine. So the computer cannot calculate L inverses in parallel, since calculating an inverse requires a subroutine call; see SARA (1984, p. 23).

So $\hat{\tilde{\Omega}}_y^{-1}$ must be computed in scalar mode. Once this inverse is available, some matrix multiplications follow (such as $\hat{\tilde{\Omega}}_y^{-1} X$), but these matrices are small; hence parallellization is not efficient. To quantify these ideas, we compute the OLS and the EGLS point estimators of (2.6) and (2.7). Into (2.6) we substitute

$$\tilde{W} = (\tilde{X}' \tilde{X})^{-1} \tilde{X}' \quad (3.5)$$

and into (2.7)

$$\tilde{V} = (\tilde{X}' \hat{\tilde{\Omega}}_y^{-1} \tilde{X})^{-1} \tilde{X}' \hat{\tilde{\Omega}}_y^{-1} . \quad (3.6)$$

\tilde{W} needs to be computed only once, but V needs to be computed $L = 100$ times (since $\hat{\tilde{\Omega}}_y$ changes every time). For the computations we select $n = 4$, $Q = 3$, and $m = 10$. To improve the accuracy of our timing data we repeat the computation 100 times. Appendix 2 gives the computer program. This yields Table 2.

TABLE 2

Total CPU times (in microseconds)
($n=4, Q=3, m=10, L=100$)

Computer	Estimator of $\tilde{\beta}$	
	OLS	EGLS
VAX 8700	920	25,580
CYBER 205		
scalar mode	734	33,206
vector mode	13	27,613

Table 2 shows that computation of inverses (using the NAG routine F01AAF) is inefficient on the CYBER 205; this supercomputer is even slower than the VAX 8700! If no subroutine calls interfere with parallelization, then the CYBER 205 is very fast: the OLS estimator $\hat{\tilde{\beta}}$ requires only 1.4% of the time needed on the VAX 8700. (Appendix 3 gives some more programming tricks for improving the efficiency of supercomputers.)

4. Conclusions

Supercomputers provide a new challenge for management scientists, since their application requires a new way of thinking, namely "thinking in parallel mode". This paper examined supercomputing in Monte Carlo experiments with multivariate regression models. Because the matrix of independent variables \tilde{X} is relatively small, supercomputers are inefficient if applied straightforwardly. Monte Carlo experiments, however, are replicated many times, say 100 times. Exploiting this dimension of the problem makes supercomputers efficient in some applications, for example, in Ordinary Least Squares. If, however, matrix inversion is needed - as is the case in Estimated Generalized Least Squares - then supercomputers seem slower than scalar computers such as the VAX 8700.

Appendix: FORTRAN 200 program for the pseudorandom number generator

```

PROGRAM VARIANT4
  IMPLICIT REAL (U-Z), INTEGER (A-T)
  PARAMETER (N1=5,N4=65535,K=1)
  PARAMETER (A3=37772072706109)
  INTEGER MVA5T
  BIT BVAST
  DESCRIPTOR MVA5T, BVAST
  DIMENSION T(N4), S1(N1)
  DIMENSION X1(N1)
  DATA MINT / X'0000800000000000' /
  CALL RANSET (K)
  DO 5 I=1,N4
    U=RANF()
    CALL RANGET(T(I))
5 CONTINUE C ! N=5
C ! SCALAR
  S1(1;N1)=T(1;N1)
  ZPU1=SECOND()
  DO 10 I=1,N1
    S1(I)=A1*S1(I)
    IF (S1(I).LT.0) S1(I)=S1(I)-MINT
    X1(I)=S1(I)/MINT
10 CONTINUE
  ZPU2=SECOND()
  U1=ZPU2-ZPU1
C ! VECTOR
  ASSIGN MVA5T, .DYN.N1
  ASSIGN BVAST, .DYN.N1
  S1(1;N1)=T(1;N1)
  ZPU1=SECOND()
  S1(1;N1)=A1*S1(1;N1)
  BVAST=S1(1;N1).LT.0
  MVA5T=S1(1;N1)-MINT
  S1(1;N1)=Q8VCTRL(MVA5T,BVAST;S1(1;N1))

```

```

X1(1;N1)=S1(1;N1)/MINT
ZPU2=SECOND()
Z1=ZPU2-ZPU1
FREE
PRINT *, 'BEGIN: GEVEKTORISEERD SCALAR
PRINT *, 'N= 5 '.Z1,' ',U1
END

```

Appendix 2: FORTRAN 200 program for the OLS and EGLS estimators

OLS ESTIMATOR FOR BETA

```

CALL MXM(XT,X,XTX)
CALL INVERSE(XTX,XTXI)
CALL MXM(XTXI,XT,W)
DO 5 I=1,N
DO 5 J=1,M
YGEM(1,I;LL)=YGEM(1,I;LL)+Y(1,J,I;LL)
5 CONTINUE
DO 10 I=1,R
DO 10 J=1,N
BETA(1,I;LL)=BETA(1,I;LL)+W(I,J)*YGEM(1,J;LL)
10 CONTINUE

```

EGLS ESTIMATOR FOR BETA

```

DO 5 I=1,N
DO 5 J=1,M
YGEM(1,I;LL)=YGEM(1,I;LL)+Y(1,J,I;LL)
5 CONTINUE
DO 10 I=1,N
DO 10 J=1,N
DO 10 K=1,M
S(1,I,J;LL)=S(1,I,J;LL)+((Y(1,K,I;LL)-
YGEM(1,I;LL))*(Y(1,K,J;LL)-YGEM(1,J;LL)))
10 CONTINUE

```

```

DO 15 I=1,
  DO 15 J=1,N
    S(1,J,I;LL)=S(1,I,J;LL)
15 CONTINUE
DO 18 K=1,LL
  DO 20 I=1,N
    DO 20 J=1,N
      DU4(J,1)=S(K,J,I)
20 CONTINUE
  CALL INVERSE(DU4,MY4,N)
  DO 25 I=1,N
    DO 25 J=1,N
      SI(K,J,I)=MY4(J,I)
25 CONTINUE
18 CONTINUE
DO 30 I=1,R
  DO 30 J=1,N
    DO 30 K=1,N
      XTSI(1,I,J;LL)=XTSI(1,I,J;LL)+XT(I,K*SI(1,K,J;LL))
30 CONTINUE
DO 35 I=1,R
  DO 35 J=1,R
    DO 35 K=1,N
      XTSIX(1,I,J;LL)=XTSIX(1,I,J;LL)+XTSI(1,I,K;LL)*X(K,J)
35 CONTINUE
DO 40 K=1,LL
  DO 45 I=1,R
    DO 45 J=1,R
      DU3(J,I)=XTSIX(K,J,I)
45 CONTINUE
  CALL INVERSE(DU3,MY3,R)
  DO 50 I=1,R
    DO 50 J=1,R
      XTSIXI(K,J,I)=MY3(J,I)
50 CONTINUE
40 CONTINUE

```

```

DO 55 I=1,R
  DO 55 J=1,N
    DO 55 K=1,R
      V(1,I,J;LL)=V(1,I,J;LL)+XTSIXI(1,I,K;LL)*XTSI(1,K,J;LL)
55  CONTINUE
  DO 60 I=1,R
    DO 60 K=1,N
      BETA(1,I;LL)=BETA(1,I;LL)+V(1,I,K;LL)*YGEM(1,K;LL)
60  CONTINUE

```

Appendix 3: Programming tricks

There are several "tricks" for improving the efficiency of supercomputers. These tricks should be applied in any computer program, not only Monte Carlo experiments:

1. Scalar divides take relatively much time (54 cycles versus 5 cycles for multiplication; 1 cycle takes 20 nanoseconds); the computation of denominators like $1/m$ (see Figure 4) and $1/(m-1)$ (see eq. 3.4) should therefore be separated by several lines of code; SARA (1984, pp. 5,7).
2. Double precision is slow and excludes vector mode; SARA (1984, p. 6).
3. There are special vectorized instructions so-called V-functions and Q8-functions. We saw some examples above; also see SARA (1984, pp. 27,30).
4. The compiler can optimize the standard FORTRAN program; next special programs (like SPY and CIA) can measure which parts of the program take most time during execution and are candidates for customized optimization.

Acknowledgement

The first author was sponsored by the Supercomputer Visiting Scientist Program at Rutgers University, The State University of New Jersey, during July 1988. In 1989 computer time on the CYBER 205 in Amsterdam was made available by SURF/NFS.

References

- CDC, *FORTRAN 200 Version 1 Reference Manual*, Publicatio no. 60480200, Control Data Corporation, Sunyvale, California 94088-3492, December 1986.
- Devroye, L., *Non-Uniform Random Variate Generation*, Springer-Verlag, New York, 1986.
- Kleijnen, J.P.C., *Statistical Tools for Simulation Practitioners*, Marcel Dekker, Inc., New York, 1987.
- Kleijnen, J.P.C., Analyzing Simulation Experiments with Common Random Numbers, *Management Science*, 34, 1(1988), 65-74.
- Kleijnen, J.P.C., Pseudorandom number generation on supercomputers, *Supercomputer (1989) (accepted for publication)*.
- Kleijnen, J.P.C. and B. Annink, *Pseudorandom number generators revisited*, Katholieke Universiteit Brabant (Catholic University Brabant), May 1989.
- Levine, R.D., Supercomputers, *Scientific American*, January 1982, 112-125.
- Naylor, T.H., J.L. Balintfy, D.S. Burdick and K. Chu, *Computer Simulation Techniques*. John Wiley & Sons, New York, 1966.
- Petersen, W.P., Some vectorized random number generators for uniform, normal, and Poisson distributions for CRAY X-MP. *The Journal of Supercomputing*, 1. (1988), 327-335.
- Rao, C.R., Some problems involving linear hypotheses in multivariate analysis, *Biometrika*, 46 (1959), 49-58.

SARA, *Cyber 205 user's guide; part 3, Optimization of FORTRAN programs.*

SARA (Stichting Academisch Rekencentrum Amsterdam/ Foundation Academic Computer Centre Amsterdam), Amsterdam, Nov. 1984.

Teichroew, D., A history of distribution sampling prior to the era of the computer and its relevance to simulation. *Journal American Statistical Association*, March 1965, 27-49.

IN 1988 REEDS VERSCHENEN

- 297 Bert Bettonvil
Factor screening by sequential bifurcation
- 298 Robert P. Gilles
On perfect competition in an economy with a coalitional structure
- 299 Willem Selen, Ruud M. Heuts
Capacitated Lot-Size Production Planning in Process Industry
- 300 J. Kriens, J.Th. van Lieshout
Notes on the Markowitz portfolio selection method
- 301 Bert Bettonvil, Jack P.C. Kleijnen
Measurement scales and resolution IV designs: a note
- 302 Theo Nijman, Marno Verbeek
Estimation of time dependent parameters in linear models
using cross sections, panels or both
- 303 Raymond H.J.M. Gradus
A differential game between government and firms: a non-cooperative
approach
- 304 Leo W.G. Strijbosch, Ronald J.M.M. Does
Comparison of bias-reducing methods for estimating the parameter in
dilution series
- 305 Drs. W.J. Reijnders, Drs. W.F. Verstappen
Strategische bespiegelingen betreffende het Nederlandse kwaliteits-
concept
- 306 J.P.C. Kleijnen, J. Kriens, H. Timmermans and H. Van den Wildenberg
Regression sampling in statistical auditing
- 307 Isolde Woittiez, Arie Kapteyn
A Model of Job Choice, Labour Supply and Wages
- 308 Jack P.C. Kleijnen
Simulation and optimization in production planning: A case study
- 309 Robert P. Gilles and Pieter H.M. Ruys
Relational constraints in coalition formation
- 310 Drs. H. Leo Theuns
Determinanten van de vraag naar vakantiereizen: een verkenning van
materiële en immateriële factoren
- 311 Peter M. Kort
Dynamic Firm Behaviour within an Uncertain Environment
- 312 J.P.C. Blanc
A numerical approach to cyclic-service queueing models

- 313 Drs. N.J. de Beer, Drs. A.M. van Nunen, Drs. M.O. Nijkamp
Does Morkmon Matter?
- 314 Th. van de Klundert
Wage differentials and employment in a two-sector model with a dual
labour market
- 315 Aart de Zeeuw, Fons Groot, Cees Withagen
On Credible Optimal Tax Rate Policies
- 316 Christian B. Mulder
Wage moderating effects of corporatism
Decentralized versus centralized wage setting in a union, firm,
government context
- 317 Jörg Glombowski, Michael Krüger
A short-period Goodwin growth cycle
- 318 Theo Nijman, Marno Verbeek, Arthur van Soest
The optimal design of rotating panels in a simple analysis of
variance model
- 319 Drs. S.V. Hannema, Drs. P.A.M. Versteijne
De toepassing en toekomst van public private partnership's bij de
grote en middelgrote Nederlandse gemeenten
- 320 Th. van de Klundert
Wage Rigidity, Capital Accumulation and Unemployment in a Small Open
Economy
- 321 M.H.C. Paardekooper
An upper and a lower bound for the distance of a manifold to a nearby
point
- 322 Th. ten Raa, F. van der Ploeg
A statistical approach to the problem of negatives in input-output
analysis
- 323 P. Kooreman
Household Labor Force Participation as a Cooperative Game; an Empiri-
cal Model
- 324 A.B.T.M. van Schaik
Persistent Unemployment and Long Run Growth
- 325 Dr. F.W.M. Boekema, Drs. L.A.G. Oerlemans
De lokale produktiestructuur doorgelicht.
Bedrijfstakverkenningen ten behoeve van regionaal-economisch onder-
zoek
- 326 J.P.C. Kleijnen, J. Kriens, M.C.H.M. Lafleur, J.H.F. Pardoel
Sampling for quality inspection and correction: AOQL performance
criteria

- 327 Theo E. Nijman, Mark F.J. Steel
Exclusion restrictions in instrumental variables equations
- 328 B.B. van der Genugten
Estimation in linear regression under the presence of heteroskedasticity of a completely unknown form
- 329 Raymond H.J.M. Gradus
The employment policy of government: to create jobs or to let them create?
- 330 Hans Kremers, Dolf Talman
Solving the nonlinear complementarity problem with lower and upper bounds
- 331 Antoon van den Elzen
Interpretation and generalization of the Lenke-Howson algorithm
- 332 Jack P.C. Kleijnen
Analyzing simulation experiments with common random numbers, part II: Rao's approach
- 333 Jacek Osiewalski
Posterior and Predictive Densities for Nonlinear Regression. A Partly Linear Model Case
- 334 A.H. van den Elzen, A.J.J. Talman
A procedure for finding Nash equilibria in bi-matrix games
- 335 Arthur van Soest
Minimum wage rates and unemployment in The Netherlands
- 336 Arthur van Soest, Peter Kooreman, Arie Kapteyn
Coherent specification of demand systems with corner solutions and endogenous regimes
- 337 Dr. F.W.M. Boekema, Drs. L.A.G. Oerlemans
De lokale produktiestructuur doorgelicht II. Bedrijfstakverkenningen ten behoeve van regionaal-economisch onderzoek. De zeescheepsnieuwbouwindustrie
- 338 Gerard J. van den Berg
Search behaviour, transitions to nonparticipation and the duration of unemployment
- 339 W.J.H. Groenendaal and J.W.A. Vingerhoets
The new cocoa-agreement analysed
- 340 Drs. F.G. van den Heuvel, Drs. M.P.H. de Vor
Kwantificering van ombuigen en bezuinigen op collectieve uitgaven 1977-1990
- 341 Pieter J.F.G. Meulendijks
An exercise in welfare economics (III)

- 342 W.J. Selen and R.M. Heuts
A modified priority index for Günther's lot-sizing heuristic under capacitated single stage production
- 343 Linda J. Mittermaier, Willem J. Selen, Jeri B. Waggoner, Wallace R. Wood
Accounting estimates as cost inputs to logistics models
- 344 Remy L. de Jong, Rashid I. Al Layla, Willem J. Selen
Alternative water management scenarios for Saudi Arabia
- 345 W.J. Selen and R.M. Heuts
Capacitated Single Stage Production Planning with Storage Constraints and Sequence-Dependent Setup Times
- 346 Peter Kort
The Flexible Accelerator Mechanism in a Financial Adjustment Cost Model
- 347 W.J. Reijnders en W.F. Verstappen
De toenemende importantie van het verticale marketing systeem
- 348 P.C. van Batenburg en J. Kriens
E.O.Q.L. - A revised and improved version of A.O.Q.L.
- 349 Drs. W.P.C. van den Nieuwenhof
Multinationalisatie en coördinatie
De internationale strategie van Nederlandse ondernemingen nader beschouwd
- 350 K.A. Bubshait, W.J. Selen
Estimation of the relationship between project attributes and the implementation of engineering management tools
- 351 M.P. Tummers, I. Woittiez
A simultaneous wage and labour supply model with hours restrictions
- 352 Marco Versteijne
Measuring the effectiveness of advertising in a positioning context with multi dimensional scaling techniques
- 353 Dr. F. Boekema, Drs. L. Oerlemans
Innovatie en stedelijke economische ontwikkeling
- 354 J.M. Schumacher
Discrete events: perspectives from system theory
- 355 F.C. Bussemaker, W.H. Haemers, R. Mathon and H.A. Wilbrink
A $(49,16,3,6)$ strongly regular graph does not exist
- 356 Drs. J.C. Caanen
Tien jaar inflatieneutrale belastingheffing door middel van vermogensaftrek en voorraadaftrek: een kwantitatieve benadering

- 357 R.M. Heuts, M. Bronckers
A modified coordinated reorder procedure under aggregate investment
and service constraints using optimal policy surfaces
- 358 B.B. van der Genugten
Linear time-invariant filters of infinite order for non-stationary
processes
- 359 J.C. Engwerda
LQ-problem: the discrete-time time-varying case
- 360 Shan-Hwei Nienhuys-Cheng
Constraints in binary semantical networks
- 361 A.B.T.M. van Schaik
Interregional Propagation of Inflationary Shocks
- 362 F.C. Drost
How to define UMVU
- 363 Rommert J. Casimir
Infogame users manual
Rev 1.2 December 1988
- 364 M.H.C. Paardekooper
A quadratically convergent parallel Jacobi-process for diagonal
dominant matrices with nondistinct eigenvalues
- 365 Robert P. Gilles, Pieter H.M. Ruys
Characterization of Economic Agents in Arbitrary Communication
Structures
- 366 Harry H. Tigelaar
Informative sampling in a multivariate linear system disturbed by
moving average noise
- 367 Jörg Glombowski
Cyclical interactions of politics and economics in an abstract
capitalist economy

IN 1989 REEDS VERSCHENEN

- 368 Ed Nijssen, Will Reijnders
"Macht als strategisch en tactisch marketinginstrument binnen de distributieketen"
- 369 Raymond Gradus
Optimal dynamic taxation with respect to firms
- 370 Theo Nijman
The optimal choice of controls and pre-experimental observations
- 371 Robert P. Gilles, Pieter H.M. Ruys
Relational constraints in coalition formation
- 372 F.A. van der Duyn Schouten, S.G. Vanneste
Analysis and computation of (n,N) -strategies for maintenance of a two-component system
- 373 Drs. R. Hamers, Drs. P. Verstappen
Het company ranking model: a means for evaluating the competition
- 374 Rommert J. Casimir
Infogame Final Report
- 375 Christian B. Mulder
Efficient and inefficient institutional arrangements between governments and trade unions; an explanation of high unemployment, corporatism and union bashing
- 376 Marno Verbeek
On the estimation of a fixed effects model with selective non-response
- 377 J. Engwerda
Admissible target paths in economic models
- 378 Jack P.C. Kleijnen and Nabil Adams
Pseudorandom number generation on supercomputers
- 379 J.P.C. Blanc
The power-series algorithm applied to the shortest-queue model
- 380 Prof. Dr. Robert Bannink
Management's information needs and the definition of costs, with special regard to the cost of interest
- 381 Bert Bettonvil
Sequential bifurcation: the design of a factor screening method
- 382 Bert Bettonvil
Sequential bifurcation for observations with random errors

- 383 Harold Houba and Hans Kremers
Correction of the material balance equation in dynamic input-output models
- 384 T.M. Doup, A.H. van den Elzen, A.J.J. Talman
Homotopy interpretation of price adjustment processes
- 385 Drs. R.T. Frambach, Prof. Dr. W.H.J. de Freytas
Technologische ontwikkeling en marketing. Een oriënterende beschouwing
- 386 A.L.P.M. Hendrikx, R.M.J. Heuts, L.G. Hoving
Comparison of automatic monitoring systems in automatic forecasting
- 387 Drs. J.G.L.M. Willems
Enkele opmerkingen over het inversificerend gedrag van multinationale ondernemingen
- 388 Jack P.C. Kleijnen and Ben Annink
Pseudorandom number generators revisited
- 389 Dr. G.W.J. Hendrikse
Speltheorie en strategisch management
- 390 Dr. A.W.A. Boot en Dr. M.F.C.M. Wijn
Liquiditeit, insolventie en vermogensstructuur
- 391 Antoon van den Elzen, Gerard van der Laan
Price adjustment in a two-country model
- 392 Martin F.C.M. Wijn, Emanuel J. Bijnen
Prediction of failure in industry
An analysis of income statements
- 393 Dr. S.C.W. Eijffinger and Drs. A.P.D. Gruijters
On the short term objectives of daily intervention by the Deutsche Bundesbank and the Federal Reserve System in the U.S. Dollar - Deutsche Mark exchange market
- 394 Dr. S.C.W. Eijffinger and Drs. A.P.D. Gruijters
On the effectiveness of daily interventions by the Deutsche Bundesbank and the Federal Reserve System in the U.S. Dollar - Deutsche Mark exchange market
- 395 A.E.M. Meijer and J.W.A. Vingerhoets
Structural adjustment and diversification in mineral exporting developing countries
- 396 R. Gradus
About Tobin's marginal and average q
A Note
- 397 Jacob C. Engwerda
On the existence of a positive definite solution of the matrix equation $X + A^T X^{-1} A = I$

- 398 Paul C. van Batenburg and J. Kriens
Bayesian discovery sampling: a simple model of Bayesian inference in auditing
- 399 Hans Kremers and Dolf Talman
Solving the nonlinear complementarity problem
- 400 Raymond Gradus
Optimal dynamic taxation, savings and investment
- 401 W.H. Haemers
Regular two-graphs and extensions of partial geometries

Bibliotheek K. U. Brabant



17 000 01086029 5