J. Basic. Appl. Sci. Res., 2(8)7645-7661, 2012
© 2012, TextRoad Publication

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Computational Correlation Coefficient and Numerically Stability for Testing Independent Random Sequences in Lattice and QMC

D. Rostamy, M. Jabbari

Department of Mathematics, Imam Khomeini International University, Qazvin, Iran

ABSTRACT

In descriptive statistics, there are two computational algorithms for determining the correlation coefficient, if we have two sets of observations $\{x_i\}_{i=1}^{k}$, $\{y_j\}_{j=1}^{m}$:

$$\text{Algorithm 1:r} = \frac{\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}(x_i-\bar{x})(y_j-\bar{y})}{\sqrt{\sum_{i=1}^{k} n_i(x_i-\bar{x})^2 \sum_{j=1}^{m} n_j(y_j-\bar{y})^2}}, \qquad \text{and Algorithm 2:r} = \frac{(\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}x_i y_j)-n\overline{x}\overline{y}}{\sqrt{\sum_{i=1}^{k} n_i(x_i-\bar{x})^2 \sum_{j=1}^{m} n_j(y_j-\bar{y})^2}}$$

Where $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, and the correlation coefficient formula will be numerically stable only if it is derived from the division of covariance over variance for they are numerically stable. It is interesting to discuss, which of the above formulas is numerically more trustworthy in the machine numbers set and lattice rules? We prove that the first algorithm is the better than the second algorithm. Numerical experiments show the efficiency of Algorithm 1.
KEYWORDS: Quasi -Monte Carlo methods, lattice rules, computational statistics, round-off error, Lattice-Nyström.

INTRODUCTION

Random sequence is at the heart of the lattice rules and quasi-Monte Carlo (QMC) methods [8, 15]. Acceptable random sequence of uniform random points for integration lattices and the digital pass a variety of such tastes:

1. Serial correlation
2. Uniformity test
3. Gap test
4. Run test
5. Permutation test.

Also, the correlation coefficient as an important concept from statistics is a measure of how well trends in the predicted values follow trends in past actual values. It is a measure of how well the predicted values from a forecast model "fit" with the real-life data. On the other hand, we observe that there are different methods for computing correlation coefficient. It is interesting to discuss, which of the methods is numerically more trustworthy in the machine numbers set?

In this paper, we want to discuss the numerically stability of serial correlation. This is a rather weak test for interdependence between two sequences. If the serial correlation coefficient is very small, then two sequences are most independent.

On the other hand, there are some aspects strategies for error analysis were investigated by [1-7,9,11,13,14,17-21]. Assessing the accuracy of the results of calculations is a paramount goal in numerical analysis. One distinguishes several kinds of errors which may limit this accuracy:

1) Errors in the input data,
2) Round of error,
3) Approximation errors.

The stability analysis of a numerical method is related to the above sources of error if the amount of the total error is very small [10,16].

In the following, we recall some basic definitions for this investigation from [11].

Definition 1. We define the set of machine numbers by

***Corresponding author**: D. Rostamy Department of Mathematics, Imam Khomeini International University, Qazvin, Iran

$$IF = IF(\beta, m, L, U) = \{0\} \cup \left\{ x \in \mathbb{R} : x = (-1)^s \beta^e \sum_{i=1}^{m} \alpha_i \beta^{-i} \right\},$$

such that the set of floating point numbers with m significant digits , base $\beta \geq 2$, $\alpha_1 \neq 0$, $0 \leq \alpha_i \leq \beta - 1$, $i = 1, \dots, m$, $s = 0,1$ and the range of (L, U) with $L \leq e \leq U$, $e \in Z$, $L \in Z$ and $U \in Z$, also, we have:

$$Card\, IF = 2(U - L + 1)(\beta - 1)\beta^{m-1} + 1.$$

Moreover, we recall that underflow and overflow are obtained if $e < L$ and $e > U$, respectively.
If we consider an algorithm same as $y = \varphi(x)$, such that function $\varphi: D \mapsto \mathbb{R}^m, D \subseteq \mathbb{R}^n, x$ and y are input and output of φ, respectively. Therefore, we can give the following decomposition for φ :

$$\varphi = \varphi^{(r)} \circ \varphi^{(r-1)} \circ \cdots \circ \varphi^{(0)}$$
$$\varphi^{(i)}: D_i \mapsto D_{i+1},\ D_j \subseteq \mathbb{R}^{n_j},\ D_0 = D, D_{r+1} \subseteq \mathbb{R}^{n_{r+1}} = \mathbb{R}^m,$$

where its sequence of elementary operations gives rise to a decomposition of φ into a sequence of elementary maps and leads from $x^{(0)} := x$ via a chain of intermediate results

$$x = x^{(0)} \mapsto \varphi^{(0)}(x^{(0)}) = x^{(1)} \mapsto \ldots \mapsto \varphi^{(r)}(x^{(r)}) = x^{(r+1)} = y,$$

to the result y. Now let us denote $\psi^{(i)}$ the ``remainder map'' by:

$$\psi^{(i)} = \varphi^{(r)} \circ \varphi^{(r-1)} \circ \cdots \circ \varphi^{(i)}: D \mapsto \mathbb{R}^m \quad i = 0, \dots, r,$$

then $\psi^{(0)} \equiv \varphi$ with floating-point arithmetic.

On the other hand, if we assume that every $\varphi^{(i)}$ is continuously differentiable on D_i. Then, we arrive at the following formula which describes the effect of the input errors Δx and the round off errors α_i on the result $y = x^{(r+1)} = \varphi(x)$:

$$\Delta y = \Delta x^{(r+1)} \doteq D\varphi(x)\Delta x + D\varphi^{(1)}(x^{(1)})\alpha_1 + \cdots + D\varphi^{(r)}(x^{(r)})\alpha_r + \alpha_{r+1}, \qquad (1)$$

where

$$E_{i+1} = \begin{pmatrix} \varepsilon_1 & 0 & 0 & \cdots & 0 \\ 0 & \varepsilon_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \varepsilon_n \end{pmatrix} \quad , |\varepsilon_j| \leq eps \ , \text{ and}$$

$$\alpha_{i+1} = E_{i+1} x^{(i+1)}.$$

Therefore, we consider $D\varphi\Delta x$ where the Jacobian of the matrix $D\psi^{(i)}$, which measures the propagation or *total effect of rounding error* is,

$$D\psi^{(1)}\alpha_1 + \ldots + D\psi^{(r)}\alpha_r + \alpha_{r+1}. \qquad (2)$$

Here, we recall the following mapping:

$$rd : D \rightarrow IF,\ \ D \subset \mathbb{R},$$

where $rd(x) = x(1+\varepsilon), |\varepsilon| \leq eps$ for all $x \in D$ and $eps = [\frac{\beta}{2}] \times \beta^{-t}$ (we call *eps* as the machine epsilon or the machine precision).

Definition 2. An algorithm is numerically more trustworthy than another algorithm if, for a given data set x , the total effect of rounding error (2) smaller for the first algorithm compared to that of the second one.

We convert the computing of the correlation coefficient formula to the following algorithms. Based on the following algorithms, since the numerically stability variance has been estimated [11], so, here, we only consider the numerically stability covariance:

$$Algorithm\ \ 1: C = \mathrm{cov}(x,y) = \frac{\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij} x_i y_j}{n} - \overline{x}\,\overline{y},$$

$$Algorithm\ \ 2: r = \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}(x_i - \overline{x})(y_j - \overline{y}).$$

Computation of DφΔx for two algorithms

Proposition 1.The terms DφΔx are changed for two algorithms.

Proof: In Algorithm 1 for $x=\left(x_1,\cdots,x_k\right), y=\left(y_1,\cdots,y_m\right)$, we have the following statement:

$$\Delta C = D\varphi.\Delta x = \frac{1}{n}\begin{bmatrix} \left(\sum_{j=1}^{m} y_j n_{1j} - \overline{y}\right)\Delta x_1 + \cdots + \left(\sum_{j=1}^{m} y_j n_{kj} - \overline{y}\right)\Delta x_k \\ + \left(\sum_{i=1}^{k} x_i n_{i1} - \overline{x}\right)\Delta y_1 + \cdots + \left(\sum_{i=1}^{k} x_i n_{im} - \overline{x}\right)\Delta y_m \end{bmatrix}^T .$$

If we write,

$$\varphi = C = \underbrace{\left(\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij} x_i y_j\right)}_{F} - \underbrace{\overline{x}\,\overline{y}}_{g},$$

$$F = \frac{1}{n}(n_{11}x_1y_1 + \cdots + n_{1m}x_1y_m + n_{21}x_2y_1 + \cdots + n_{2m}x_2y_m + \cdots + n_{k1}x_ky_1 + \cdots + n_{km}x_ky_m),$$

$$\frac{\partial F}{\partial x} = \frac{1}{n}\begin{bmatrix} n_{11}y_1 + \cdots + n_{1m}y_m \\ n_{21}y_1 + \cdots + n_{2m}y_m \\ \vdots \\ n_{k1}y_1 + \cdots + n_{km}y_m \end{bmatrix}^T = \frac{1}{n}\begin{bmatrix} \sum_{j=1}^{m} n_{1j}y_j \\ \sum_{j=1}^{m} n_{2j}y_j \\ \vdots \\ \sum_{j=1}^{m} n_{kj}y_j \end{bmatrix}^T ,$$

$$\frac{\partial F}{\partial y} = \frac{1}{n}\begin{bmatrix} n_{11}x_1 + \cdots + n_{k1}x_k \\ n_{12}x_1 + \cdots + n_{k2}x_k \\ \vdots \\ n_{1m}x_1 + \cdots + n_{km}x_k \end{bmatrix}^T$$

and

$$g = \overline{x}\,\overline{y} = \frac{\sum_{i=1}^{k} x_i}{n}.\frac{\sum_{j=1}^{m} y_j}{n} = \frac{1}{n^2}\sum_{i=1}^{k}\sum_{j=1}^{m} x_i y_j$$

$$= \frac{1}{n^2}\begin{bmatrix} x_1y_1 + \ldots + x_1y_m \\ x_2y_1 + \ldots + x_2y_m \\ \vdots \\ x_ky_1 + \ldots + x_ky_m \end{bmatrix}^T ,$$

$$\frac{\partial g}{\partial x} = \frac{1}{n^2}\begin{bmatrix} y_1 + \ldots + y_m \\ \vdots \\ y_1 + \ldots + y_m \end{bmatrix}^T = \frac{1}{n^2}\begin{bmatrix} \sum_{j=1}^{m} y_j \\ \vdots \\ \sum_{j=1}^{m} y_j \end{bmatrix}^T ,$$

and

$$\frac{\partial g}{\partial y} = \frac{1}{n^2}\begin{bmatrix} x_1 + \ldots + x_k \\ \vdots \\ x_1 + \ldots + x_k \end{bmatrix}^T = \frac{1}{n^2}\begin{bmatrix} \sum_{i=1}^{k} x_i \\ \vdots \\ \sum_{i=1}^{k} x_i \end{bmatrix}^T ,$$

On the other hand we have:

$$\frac{\partial\varphi}{\partial x}=\frac{1}{n}\begin{bmatrix}\sum_{j=1}^{m} n_{1j}y_j\\ \sum_{j=1}^{m} n_{2j}y_j\\ \vdots\\ \sum_{j=1}^{m} n_{kj}y_j\end{bmatrix}^T-\frac{1}{n^2}\begin{bmatrix}\sum_{j=1}^{m} y_j\\ \sum_{j=1}^{m} y_j\\ \vdots\\ \sum_{j=1}^{m} y_j\end{bmatrix}^T .$$

For derivation related to x_1, we can write:

$$\begin{aligned}&\left(\frac{1}{n}n_{11}y_1-\frac{1}{n^2}y_1\right)+\left(\frac{1}{n}n_{12}y_2-\frac{1}{n^2}y_2\right)+\cdots+\left(\frac{1}{n}n_{1m}y_m-\frac{1}{n^2}y_m\right)\\ &=\frac{1}{n}\left[y_1\left(n_{11}-\frac{1}{n}\right)+y_2\left(n_{12}-\frac{1}{n}\right)+\cdots+y_m\left(n_{1m}-\frac{1}{n}\right)\right]\\ &=\frac{1}{n}\sum_{j=1}^{m}y_j\left(n_{1j}-\frac{1}{n}\right).\end{aligned}$$

Therefore, for all the other derivations of $x=(x_1,\ldots,x_k)$ we have:

$$\frac{\partial\varphi}{\partial x}=\frac{1}{n}\begin{bmatrix}\sum_{j=1}^{m} y_j\left(n_{1j}-\frac{1}{n}\right)\\ \sum_{j=1}^{m} y_j\left(n_{2j}-\frac{1}{n}\right)\\ \vdots\\ \sum_{j=1}^{m} y_j\left(n_{kj}-\frac{1}{n}\right)\end{bmatrix}^T ,$$

and for $y=(y_1,\cdots,y_m)$

$$\frac{\partial\varphi}{\partial y}=\frac{1}{n}\begin{bmatrix}n_{11}x_1+\cdots+n_{k1}x_k\\ \vdots\\ n_{1m}x_1+\cdots+n_{km}x_k\end{bmatrix}^T-\frac{1}{n^2}\begin{bmatrix}x_1+\ldots+x_k\\ \vdots\\ x_1+\ldots+x_k\end{bmatrix}^T .$$

For simplifying, we can write:

$$\begin{aligned}&\left(\frac{1}{n}n_{11}x_1-\frac{1}{n^2}x_1\right)+\left(\frac{1}{n}n_{21}x_2-\frac{1}{n^2}x_2\right)+\cdots+\left(\frac{1}{n}n_{k1}x_k-\frac{1}{n^2}x_k\right)\\ &=\frac{1}{n}x_1\left(n_{11}-\frac{1}{n}\right)+\frac{1}{n}x_2\left(n_{21}-\frac{1}{n}\right)+\cdots+\frac{1}{n}x_k\left(n_{k1}-\frac{1}{n}\right)=\frac{1}{\mathrm{n}}\sum_{i=1}^{k}\mathrm{x}_i\left(n_{i1}-\frac{1}{n}\right).\end{aligned}$$

Therefore, for all the other derivations of $y=(y_1,\ldots,y_m)$ we have:

$$\frac{\partial\varphi}{\partial y}=\frac{1}{n}\begin{bmatrix}\sum_{i=1}^{k} x_i\left(n_{i1}-\frac{1}{n}\right)\\ \vdots\\ \sum_{i=1}^{k} x_i\left(n_{im}-\frac{1}{n}\right)\end{bmatrix}^T .$$

Then we have:

$$\Delta C=D\varphi.\Delta x=\frac{1}{n}\begin{bmatrix}\left(\sum_{j=1}^{m}y_j\left(n_{1j}-\frac{1}{n}\right)\right)\Delta x_1+\cdots+\left(\sum_{j=1}^{m}y_j\left(n_{kj}-\frac{1}{n}\right)\right)\Delta x_k\\ +\left(\sum_{i=1}^{k}x_i\left(n_{i1}-\frac{1}{n}\right)\right)\Delta y_1+\cdots+\left(\sum_{i=1}^{k}x_i\left(n_{im}-\frac{1}{n}\right)\right)\Delta y_m\end{bmatrix}^T .$$

We simplify this statement once more to reach the relation DφΔx of the first algorithm:

$$\Delta C = D\varphi.\Delta x = \frac{1}{n}\begin{bmatrix} \left(\sum_{j=1}^{m} y_j n_{1j} - \overline{y}\right)\Delta x_1 + \cdots + \left(\sum_{j=1}^{m} y_j n_{kj} - \overline{y}\right)\Delta x_k \\ + \left(\sum_{i=1}^{k} x_i n_{i1} - \overline{x}\right)\Delta y_1 + \cdots + \left(\sum_{i=1}^{k} x_i n_{im} - \overline{x}\right)\Delta y_m \end{bmatrix}^T.$$

In algorithm 2, we have the following statement:

$$\Delta C = D\varphi.\Delta x$$

$$= \frac{1}{n}\begin{bmatrix} \left(\left(-\frac{1}{n}\right)\sum_{i=2}^{k}\sum_{j=1}^{m} n_{ij}\left(y_j - \overline{y}\right) + \left(1-\frac{1}{n}\right)\sum_{j=1}^{m} n_{1j}\left(y_j - \overline{y}\right)\right)\Delta x_1 + \cdots \\ + \left(\left(-\frac{1}{n}\right)\sum_{i=1}^{k-1}\sum_{j=1}^{m} n_{ij}\left(y_j - \overline{y}\right) + \left(1-\frac{1}{n}\right)\sum_{j=1}^{m} n_{kj}\left(y_j - \overline{y}\right)\right)\Delta x_k \\ + \left(1-\frac{1}{n}\right)\sum_{i=1}^{k} n_{i1}\left(x_i - \overline{x}\right) + \left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=2}^{m} n_{ij}\left(x_i - \overline{x}\right)\Delta y_1 \\ + \cdots + \left(1-\frac{1}{n}\right)\sum_{i=1}^{k} n_{im}\left(x_i - \overline{x}\right) + \left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=1}^{m-1} n_{ij}\left(x_i - \overline{x}\right)\Delta y_m \end{bmatrix}^T.$$

If we write,

$$\varphi = \frac{1}{n}\begin{bmatrix} n_{11}\left(x_1 - \frac{x_1+\ldots+x_k}{n}\right)\left(y_1 - \frac{y_1+\ldots+y_m}{n}\right) + \cdots \\ + n_{1m}\left(x_1 - \frac{x_1+\ldots+x_k}{n}\right)\left(y_m - \frac{y_1+\ldots+y_m}{n}\right) \\ + n_{21}\left(x_2 - \frac{x_1+\ldots+x_k}{n}\right)\left(y_1 - \frac{y_1+\ldots+y_m}{n}\right) + \cdots \\ + n_{2m}\left(x_2 - \frac{x_1+\ldots+x_k}{n}\right)\left(y_m - \frac{y_1+\ldots+y_m}{n}\right) + \cdots \\ + n_{k1}\left(x_k - \frac{x_1+\ldots+x_k}{n}\right)\left(y_1 - \frac{y_1+\ldots+y_m}{n}\right) + \cdots \\ + n_{km}\left(x_k - \frac{x_1+\ldots+x_k}{n}\right)\left(y_m - \frac{y_1+\ldots+y_m}{n}\right) \end{bmatrix}^T$$

$$\frac{\partial\varphi}{\partial x_1} = \frac{1}{n}\begin{bmatrix} n_{11}\left(1-\frac{1}{n}\right)\left(y_1 - \overline{y}\right) + \cdots + n_{1m}\left(1-\frac{1}{n}\right)\left(y_m - \overline{y}\right) \\ + n_{21}\left(-\frac{1}{n}\right)\left(y_1 - \overline{y}\right) + \cdots + n_{2m}\left(-\frac{1}{n}\right)\left(y_m - \overline{y}\right) + \cdots \\ + n_{k1}\left(-\frac{1}{n}\right)\left(y_1 - \overline{y}\right) + \cdots + n_{km}\left(-\frac{1}{n}\right)\left(y_m - \overline{y}\right) \end{bmatrix}^T$$

$$= \frac{1}{n}\left[\left(-\frac{1}{n}\right)\sum_{i=2}^{k}\sum_{j=1}^{m} n_{ij}\left(y_j - \overline{y}\right) + \left(1-\frac{1}{n}\right)\sum_{j=1}^{m} n_{1j}\left(y_j - \overline{y}\right)\right]$$

$$\vdots$$

$$\frac{\partial\varphi}{\partial x_k} = \frac{1}{n}\begin{bmatrix} n_{11}\left(-\frac{1}{n}\right)\left(y_1 - \overline{y}\right) + \cdots + n_{1m}\left(-\frac{1}{n}\right)\left(y_m - \overline{y}\right) \\ + n_{21}\left(-\frac{1}{n}\right)\left(y_1 - \overline{y}\right) + \cdots + n_{2m}\left(-\frac{1}{n}\right)\left(y_m - \overline{y}\right) + \cdots \\ + n_{k1}\left(1-\frac{1}{n}\right)\left(y_1 - \overline{y}\right) + \cdots + n_{km}\left(1-\frac{1}{n}\right)\left(y_m - \overline{y}\right) \end{bmatrix}^T$$

$$= \frac{1}{n}\left[\left(-\frac{1}{n}\right)\sum_{i=1}^{k-1}\sum_{j=1}^{m} n_{ij}\left(y_j - \overline{y}\right) + \left(1-\frac{1}{n}\right)\sum_{j=1}^{m} n_{kj}\left(y_j - \overline{y}\right)\right],$$

And we can write

$$\frac{\partial\varphi}{\partial y_1}=\frac{1}{n}\begin{bmatrix} n_{11}\left(1-\frac{1}{n}\right)\left(x_1-\overline{x}\right)+n_{12}\left(-\frac{1}{n}\right)\left(x_1-\overline{x}\right)+\cdots \\ +n_{1m}\left(-\frac{1}{n}\right)\left(x_1-\overline{x}\right)+n_{21}\left(1-\frac{1}{n}\right)\left(x_2-\overline{x}\right) \\ +n_{22}\left(-\frac{1}{n}\right)\left(x_2-\overline{x}\right)+\cdots+n_{2m}\left(-\frac{1}{n}\right)\left(x_2-\overline{x}\right)+\cdots \\ +n_{k1}\left(1-\frac{1}{n}\right)\left(x_k-\overline{x}\right)+n_{k2}\left(-\frac{1}{n}\right)\left(x_k-\overline{x}\right)+\cdots \\ +n_{km}\left(-\frac{1}{n}\right)\left(x_k-\overline{x}\right)\end{bmatrix}^T$$

$$=\frac{1}{n}\left[\left(1-\frac{1}{n}\right)\sum_{i=1}^{k}n_{i1}\left(x_i-\overline{x}\right)+\left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=2}^{m}n_{ij}\left(x_i-\overline{x}\right)\right]$$

$$\vdots$$

$$\frac{\partial\varphi}{\partial y_m}=\begin{bmatrix} n_{11}\left(-\frac{1}{n}\right)\left(x_1-\overline{x}\right)+\cdots+n_{1m}\left(1-\frac{1}{n}\right)\left(x_1-\overline{x}\right) \\ +n_{21}\left(-\frac{1}{n}\right)\left(x_2-\overline{x}\right)+\cdots+n_{2m}\left(1-\frac{1}{n}\right)\left(x_2-\overline{x}\right)+\cdots \\ +n_{k1}\left(-\frac{1}{n}\right)\left(x_k-\overline{x}\right)+\cdots+n_{km}\left(1-\frac{1}{n}\right)\left(x_k-\overline{x}\right)\end{bmatrix}^T$$

$$=\frac{1}{n}\left[\left(1-\frac{1}{n}\right)\sum_{i=1}^{k}n_{im}\left(x_i-\overline{x}\right)+\left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=1}^{m-1}n_{ij}\left(x_i-\overline{x}\right)\right].$$

So, for derivation related to $x=(x_1,\ldots,x_k)$,we can write:

$$\frac{\partial\varphi}{\partial x}=\frac{1}{n}\begin{bmatrix}\left(-\frac{1}{n}\right)\sum_{i=2}^{k}\sum_{j=1}^{m}n_{ij}\left(y_j-\overline{y}\right)+\left(1-\frac{1}{n}\right)\sum_{j=1}^{m}n_{1j}\left(y_j-\overline{y}\right) \\ \vdots \\ \left(-\frac{1}{n}\right)\sum_{i=1}^{k-1}\sum_{j=1}^{m}n_{ij}\left(y_j-\overline{y}\right)+\left(1-\frac{1}{n}\right)\sum_{j=1}^{m}n_{kj}\left(y_j-\overline{y}\right)\end{bmatrix}^T.$$

Also, for all the other derivations of $y=(y_1,\ldots,y_m)$:

$$\frac{\partial\varphi}{\partial y}=\frac{1}{n}\begin{bmatrix}\left(1-\frac{1}{n}\right)\sum_{i=1}^{k}n_{i1}\left(x_i-\overline{x}\right)+\left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=2}^{m}n_{ij}\left(x_i-\overline{x}\right) \\ \vdots \\ \left(1-\frac{1}{n}\right)\sum_{i=1}^{k}n_{im}\left(x_i-\overline{x}\right)+\left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=1}^{m-1}n_{ij}\left(x_i-\overline{x}\right)\end{bmatrix}^T.$$

Then we have:

$$\Delta C=D\varphi.\Delta x$$

$$=\frac{1}{n}\begin{bmatrix}\left(\left(-\frac{1}{n}\right)\sum_{i=2}^{k}\sum_{j=1}^{m}n_{ij}\left(y_j-\overline{y}\right)+\left(1-\frac{1}{n}\right)\sum_{j=1}^{m}n_{1j}\left(y_j-\overline{y}\right)\right)\Delta x_1+\cdots \\ +\left(\left(-\frac{1}{n}\right)\sum_{i=1}^{k-1}\sum_{j=1}^{m}n_{ij}\left(y_j-\overline{y}\right)+\left(1-\frac{1}{n}\right)\sum_{j=1}^{m}n_{kj}\left(y_j-\overline{y}\right)\right)\Delta x_k \\ +\left(1-\frac{1}{n}\right)\sum_{i=1}^{k}n_{i1}\left(x_i-\overline{x}\right)+\left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=2}^{m}n_{ij}\left(x_i-\overline{x}\right)\Delta y_1 \\ +\cdots+\left(1-\frac{1}{n}\right)\sum_{i=1}^{k}n_{im}\left(x_i-\overline{x}\right)+\left(-\frac{1}{n}\right)\sum_{i=1}^{k}\sum_{j=1}^{m-1}n_{ij}\left(x_i-\overline{x}\right)\Delta y_m\end{bmatrix}^T.$$

Having compared this relation DφΔx for two algorithms, it is demonstrated that this amount for the second algorithm is more than of the first one.
Hence, proof is completed.

Total effect of rounding for algorithm 1

Proposition 2. A bound for the total effect of rounding error for algorithm 1 is as follows:

$$|A| \le \left[\frac{2}{n}\left(\sum_{i=1}^{k}\sum_{j=1}^{m} x_i y_j n_{ij} + \overline{xy}\right) + 2y\right] eps.(\mathrm{I})$$

Here, we assume A: = total effect of rounding error for algorithm 1 , such that

$$A = \left(\frac{1}{n}\sum_{j=1}^{m}\sum_{i=1}^{k} x_i y_j n_{ij}\varepsilon_1 + \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m} y_j x_i n_{ij}\varepsilon_2 - \overline{x}\,\overline{y}\varepsilon_3 - \overline{x}\,\overline{y}\varepsilon_4\right)$$
$$+\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m} x_i y_j n_{ij}\varepsilon_5 - \overline{x}\,\overline{y}\varepsilon_6 + \left(\frac{1}{n}\left(\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij} x_i y_j\right) - \overline{x}\,\overline{y}\right)\varepsilon_7.$$

Proof: If we consider algorithm 1 then for simplify we can write:

$$\varphi^{(o)}\left(x^{(0)}\right) = x^{(1)} = \left(\sum_{i=1}^{k} x_i, \sum_{j=1}^{m} y_i, \overline{x}, \overline{y}\right),$$
$$\varphi^{(1)}\left(x^{(1)}\right) = \left(x^{(2)}\right) = \left(\sum_{i=1}^{k}\sum_{j=1}^{m} x_i y_j n_{ij}, \overline{x}\,\overline{y}\right),$$
$$\varphi^{(2)}\left(x^{(2)}\right) = \left(x^{(3)}\right) = \frac{1}{n}\left(\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij} x_i y_j\right) - \overline{x}\,\overline{y}.$$

Therefore, we have:

$$\varphi = \varphi^{(2)} o \varphi^{(1)} o \varphi^{(o)},$$
$$\Delta y = \Delta x^{(3)} = D\varphi.\Delta x + D\psi^{(1)}\left(x^{(1)}\right)\alpha_1 + D\psi^{(2)}\left(x^{(2)}\right)\alpha_2 + \alpha_3$$

and we can conclude

$$x^{(0)} = \begin{pmatrix} x_1 \\ \vdots \\ x_k \\ y_1 \\ \vdots \\ y_m \end{pmatrix} \xrightarrow{\varphi^{(o)}} x^{(1)} = \begin{pmatrix} \sum_{i=1}^{k} x_i \\ \sum_{j=1}^{m} y_i \\ \overline{x} \\ \overline{y} \end{pmatrix} \xrightarrow{\varphi^{(1)}} x^{(2)} = \begin{pmatrix} \sum_{i=1}^{k}\sum_{j=1}^{m} x_i y_j n_{ij} \\ \overline{x}\,\overline{y} \end{pmatrix}$$
$$\xrightarrow{\varphi^{(2)}} x^{(3)} = \frac{1}{n}\left(\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij} x_i y_j\right) - \overline{x}\,\overline{y}.$$

Hence, we can write

$$\varphi^{(o)}\left(x_1, \ldots, x_k, y_1, \ldots, y_m\right) = \begin{pmatrix} \sum_{i=1}^{k} x_i \\ \sum_{j=1}^{m} y_j \\ \overline{x} \\ \overline{y} \end{pmatrix},$$
$$\varphi^{(1)}(u, v, t, s) = \begin{pmatrix} u v n_{ij} \\ ts \end{pmatrix},$$
$$\varphi^{(2)}(p, q) = \frac{p}{n} - q,$$
$$\psi^{(1)}\left(x^{(1)}\right) = \psi^{(1)}(u, v, t, s) = \varphi^{(2)} o \varphi^{(1)}(u, v, t, s) = \varphi^{(2)}\left(u v n_{ij}, ts\right) = \frac{u v n_{ij}}{n} - ts,$$

$$D\psi^{(1)}(u,v,t,s)=\left(\frac{1}{n}vn_{ij},\frac{1}{n}un_{ij},-s,-t\right),$$
$$D\psi^{(1)}\left(x^{(1)}\right)=\left(\frac{1}{n}\sum_{j=1}^{m}y_j n_{ij},\frac{1}{n}\sum_{i=1}^{k}x_i n_{ij},-\overline{y},-\overline{x}\right),$$
$$\psi^{(2)}\left(x^{(2)}\right)=\psi^{(2)}(p,q)=\varphi^{(2)}(p,q)=\frac{p}{n}-q,$$
$$D\psi^{(2)}\left(x^{(2)}\right)=\left(\frac{1}{n},-1\right).$$

Moreover, we have:

$$\alpha_1=\overline{\varphi^{(0)}x^{(0)}}-\varphi^{(0)}x^{(0)}=\begin{pmatrix}\sum_{i=1}^{k}x_i\varepsilon_1\\ \sum_{j=1}^{m}y_j\varepsilon_2\\ \overline{x}\varepsilon_3\\ \overline{y}\varepsilon_4\end{pmatrix}^T,$$

$$\alpha_2=\overline{\varphi^{(1)}x^{(1)}}-\varphi^{(1)}x^{(1)}=\begin{pmatrix}\sum_{i=1}^{k}\sum_{j=1}^{m}x_i y_j n_{ij}\varepsilon_5\\ \overline{x}\,\overline{y}\varepsilon_6\end{pmatrix}^T,$$

$$\alpha_3=\overline{\varphi^{(2)}x^{(2)}}-\varphi^{(2)}x^{(2)}=\left(\frac{1}{n}\left(\sum_{i=1}^{k}\sum_{j=1}^{m}n_{ij}x_i y_j\right)-\overline{x}\,\overline{y}\right)\varepsilon_7.$$

So, we can write:

$$\Delta y=\Delta x^{(3)}=D\varphi.\Delta x+D\psi^{(1)}\left(x^{(1)}\right)\alpha_1+D\psi^{(2)}\left(x^{(2)}\right)\alpha_2+\alpha_3=D\varphi.\Delta x$$
$$+\left(\frac{1}{n}\sum_{j=1}^{m}y_j n_{ij},\frac{1}{n}\sum_{i=1}^{k}x_i n_{ij},-\overline{y},-\overline{x}\right)\begin{pmatrix}\sum_{i=1}^{k}x_i\varepsilon_1\\ \sum_{j=1}^{m}y_j\varepsilon_2\\ \overline{x}\varepsilon_3\\ \overline{y}\varepsilon_4\end{pmatrix}+\left(\frac{1}{n},-1\right)$$
$$\begin{pmatrix}\sum_{i=1}^{k}\sum_{j=1}^{m}x_i y_j n_{ij}\varepsilon_5\\ \overline{xy}\varepsilon_6\end{pmatrix}+\left(\frac{1}{n}\left(\sum_{i=1}^{k}\sum_{j=1}^{m}x_i y_j n_{ij}\right)-\overline{xy}\right)\varepsilon_7,$$

Therefore we have

$$\Delta y=D\varphi.\Delta x+\left(\frac{1}{n}\sum_{j=1}^{m}\sum_{i=1}^{k}x_i y_j n_{ij}\varepsilon_1+\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}y_j x_i n_{ij}\varepsilon_2-\overline{x}\,\overline{y}\varepsilon_3-\overline{x}\,\overline{y}\varepsilon_4\right)$$
$$+\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}x_i y_j n_{ij}\varepsilon_5-\overline{x}\,\overline{y}\varepsilon_6+\left(\frac{1}{n}\left(\sum_{i=1}^{k}\sum_{j=1}^{m}n_{ij}x_i y_j\right)-\overline{x}\,\overline{y}\right)\varepsilon_7.$$

Therefore, the proof is completed.

Total effect of rounding error for algorithm 2

Proposition 3: A bound for the total effect of rounding error for algorithm 2 is as follows:

$$|B| \leq \left[\frac{2}{n} \left(\sum_{i=1}^{k} \sum_{j=1}^{m} x_i y_j n_{ij} + \overline{xy} \right) + 2y \right] eps. (\mathrm{II})$$

Here, we assume B: = Total effect of rounding error for algorithm 2, such that

$$B = \left(-\frac{1}{n}\right)\bar{x}\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}\left(y_j - \bar{y}\right)\varepsilon + \left(-\frac{1}{n}\right)\bar{y}\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}\left(x_i - \bar{x}\right)\varepsilon' + \frac{1}{n}\left(x_1 - \bar{x}\right)\varepsilon_1$$
$$\sum_{j=1}^{m} n_{1j}\left(y_j - \bar{y}\right) + \cdots + \frac{1}{n}\left(x_k - \bar{x}\right)\varepsilon_k \sum_{j=1}^{m} n_{kj}\left(y_j - \bar{y}\right) + \frac{1}{n}\left(y_1 - \bar{y}\right)\varepsilon'_1 \sum_{i=1}^{k} n_{i1}\left(x_i - \bar{x}\right)$$
$$+ \cdots + \frac{1}{n}\left(y_m - \bar{y}\right)\varepsilon'_m \sum_{i=1}^{k} n_{im}\left(x_i - \bar{x}\right) + \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i - \bar{x}\right)\left(y_j - \bar{y}\right) n_{ij}\varepsilon_{k+1}$$
$$+ \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i - \bar{x}\right)\left(y_j - \bar{y}\right) n_{ij}\varepsilon'_{m+1} + \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i - \bar{x}\right)\left(y_j - \bar{y}\right)\varepsilon''.$$

Proof: If we consider algorithm 2 then for simplify we can write:

$$\varphi^{(o)}\left(x^{(0)}\right) = x^{(1)} = \begin{pmatrix} x_1 \\ \vdots \\ x_k \\ y_1 \\ \vdots \\ y_m \\ \bar{x} \\ \bar{y} \end{pmatrix},$$

$$\varphi^{(1)}\left(x^{(1)}\right) = \left(x^{(2)}\right) = \begin{pmatrix} \left(x_1 - \bar{x}\right) \\ \vdots \\ \left(x_k - \bar{x}\right) \\ \left(y_1 - \bar{y}\right) \\ \vdots \\ \left(y_m - \bar{y}\right) \end{pmatrix},$$

$$\varphi^{(2)}\left(x^{(2)}\right) = x^{(3)} = \begin{pmatrix} \sum_{i=1}^{k}\left(x_i - \bar{x}\right) \\ \sum_{j=1}^{m}\left(y_j - \bar{y}\right) \end{pmatrix},$$

$$\varphi^{(3)}\left(x^{(3)}\right) = x^{(4)} = \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i - \bar{x}\right)\left(y_j - \bar{y}\right).$$

Therefore, we have:

$$\varphi = \varphi^{(3)} o \varphi^{(2)} o \varphi^{(1)} o \varphi^{(o)}$$
$$\Delta y = \Delta x^{(4)} = D\varphi.\Delta x + D\psi^{(1)}\left(x^{(1)}\right)\alpha_1 + D\psi^{(2)}\left(x^{(2)}\right)\alpha_2 + D\psi^{(3)}\left(x^{(3)}\right)\alpha_3 + \alpha_4$$

we can conclude

$$x^{(0)}=\begin{pmatrix}x_1\\ \vdots\\ x_k\\ y_1\\ \vdots\\ y_m\end{pmatrix}\xrightarrow{\varphi^{(o)}}x^{(1)}=\begin{pmatrix}x_1\\ \vdots\\ x_k\\ y_1\\ \vdots\\ y_m\\ \overline{x}\\ \overline{y}\end{pmatrix}\xrightarrow{\varphi^{(1)}}x^{(2)}=\begin{pmatrix}\left(x_1-\overline{x}\right)\\ \vdots\\ \left(x_k-\overline{x}\right)\\ \left(y_1-\overline{y}\right)\\ \vdots\\ \left(y_m-\overline{y}\right)\end{pmatrix}$$

$$\xrightarrow{\varphi^{(2)}}x^{(3)}=\begin{pmatrix}\sum_{i=1}^{k}\left(x_i-\overline{x}\right)\\ \sum_{j=1}^{m}\left(y_j-\overline{y}\right)\end{pmatrix}\xrightarrow{\varphi^{(3)}}x^{(4)}=\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i-\overline{x}\right)\left(y_j-\overline{y}\right).$$

Hence, we have

$$\varphi^{(o)}\left(x_1,\cdots,x_k,y_1,\cdots,y_m\right)=\left(x_1,\cdots,x_k,y_1,\cdots,y_m,\overline{x},\overline{y}\right),$$

$$\varphi^{(1)}\left(u_1,\cdots,u_k,w_1,\cdots,w_m,u,w\right)=\left(\left(u_1-u\right),\cdots,\left(u_k-u\right),\left(w_1-w\right),\cdots,\left(w_m-w\right)\right);$$

$$\varphi^{(2)}\left(v_1,\cdots,v_k,\gamma_1,\cdots,\gamma_m\right)=\left(\sum_{i=1}^{k}v_i,\sum_{i=1}^{k}\gamma_i\right),$$

$$\varphi^{(3)}\left(p,q\right)=\frac{1}{n}pqn_{ij},$$

$$\psi^{(1)}\left(x^{(1)}\right)=\varphi^{(3)}o\varphi^{(2)}o\varphi^{(1)}\left(u_1,\cdots,u_k,w_1,\cdots,w_m,u,w\right)=$$

$$\varphi^{(3)}o\varphi^{(2)}\left(u_1-u\right),\cdots,\left(u_k-u\right),\left(w_1-w\right),\cdots,\left(w_m-w\right)$$

$$=\varphi^{(3)}\left(\sum_{i=1}^{k}\left(u_i-u\right),\sum_{j=1}^{k}\left(w_j-w\right)\right)$$

$$=\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(u_i-u\right)\left(w_j-w\right)n_{ij},$$

On the other hand we have

$$D\psi^{(1)}=\frac{1}{n}\begin{bmatrix}\left(\sum_{j=1}^{m}\left(w_j-w\right)n_{1j},\cdots,\sum_{j=1}^{m}\left(w_j-w\right)n_{kj}\right)\\ ,\left(\sum_{i=1}^{k}\left(u_i-u\right)n_{i1},\cdots,\sum_{i=1}^{k}\left(u_i-u\right)n_{im}\right)\\ ,-\sum_{i=1}^{k}\sum_{j=1}^{m}\left(w_j-w\right)n_{ij},-\sum_{i=1}^{k}\sum_{j=1}^{m}\left(u_i-u\right)n_{ij}\end{bmatrix},$$

$$D\psi^{(1)}\left(x^1\right)=\frac{1}{n}\begin{bmatrix}\left(\sum_{j=1}^{m}\left(y_j-\overline{y}\right)n_{1j},\cdots,\sum_{j=1}^{m}\left(y_j-\overline{y}\right)n_{kj}\right)\\ ,\left(\sum_{i=1}^{k}\left(x_i-\overline{x}\right)n_{i1},\cdots,\sum_{i=1}^{k}\left(x_i-\overline{x}\right)n_{im}\right)\\ ,-\sum_{i=1}^{k}\sum_{j=1}^{m}\left(y_j-\overline{y}\right)n_{ij},-\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i-\overline{x}\right)n_{ij}\end{bmatrix},$$

$$\psi^{(2)} = \varphi^{(3)} o \varphi^{(2)} \left(v_1, \cdots, v_k, \gamma_1, \cdots, \gamma_m\right) = \varphi^{(3)} \left(\sum_{i=1}^{k} v_i, \sum_{j=1}^{m} \gamma_j \right) = \frac{1}{n} \sum_{i=1}^{k} \sum_{j=1}^{m} v_i \gamma_i n_{ij}$$

moreover, we have

$$D\psi^{(2)} = \frac{1}{n} \left(\sum_{j=1}^{m} n_{1j} \gamma_j, \cdots, \sum_{j=1}^{m} n_{kj} \gamma_j, \sum_{i=1}^{k} n_{i1} v_1, \cdots, \sum_{i=1}^{k} n_{im} v_i \right),$$

$$D\psi^{(2)}\left(x^2\right) = \frac{1}{n} \left(\begin{array}{l} \sum_{j=1}^{m} n_{1j} \left(y_j - \bar{y}\right), \cdots, \sum_{j=1}^{m} n_{kj} \left(y_j - \bar{y}\right), \\ \sum_{i=1}^{k} n_{i1} \left(x_i - \bar{x}\right), \cdots, \sum_{i=1}^{k} n_{im} \left(x_i - \bar{x}\right) \end{array} \right),$$

$$\psi^{(3)} = \varphi^{(3)} = \frac{1}{n} pqn_{ij},$$

$$D\psi^{(3)} = \left(\frac{1}{n} qn_{ij}, \frac{1}{n} pn_{ij} \right) = \frac{1}{n} n_{ij} \left(q, p\right),$$

$$D\psi^{(3)}\left(x^{(3)}\right) = \frac{1}{n} n_{ij} \left(\sum_{j=1}^{m} \left(y_j - \bar{y}\right), \sum_{i=1}^{k} \left(x_i - \bar{x}\right) \right).$$

Also, we know that

$$\alpha_1 = \overline{\varphi^{(0)} x^{(0)}} - \varphi^{(0)} x^{(0)} = \begin{pmatrix} \left.\begin{matrix} 0 \\ \vdots \\ 0 \end{matrix}\right\} k \\ \left.\begin{matrix} 0 \\ \vdots \\ 0 \end{matrix}\right\} m \\ \bar{x}\varepsilon \\ \bar{y}\varepsilon' \end{pmatrix}$$

Therefore we can write

$$\alpha_2 = \overline{\varphi^{(1)} x^{(1)}} - \varphi^{(1)} x^{(1)} = \begin{pmatrix} \left(x_1 - \bar{x}\right)\varepsilon_1 \\ \vdots \\ \left(x_k - \bar{x}\right)\varepsilon_k \\ \left(y_1 - \bar{y}\right)\varepsilon'_1 \\ \vdots \\ \left(y_m - \bar{y}\right)\varepsilon'_m \end{pmatrix},$$

$$\alpha_3 = \overline{\varphi^{(2)} x^{(2)}} - \varphi^{(2)} x^{(2)} = \begin{pmatrix} \sum_{i=1}^{k} \left(x_i - \bar{x}\right)\varepsilon_{k+1} \\ \sum_{j=1}^{m} \left(y_j - \bar{y}\right)\varepsilon'_{m+1} \end{pmatrix},$$

$$\alpha_4 = \overline{\varphi^{(3)} x^{(3)}} - \varphi^{(3)} x^{(3)} = \frac{\sum_{i=1}^{k} \sum_{j=1}^{m} n_{ij} \left(x_i - \bar{x}\right)\left(y_j - \bar{y}\right)}{n} \varepsilon''.$$

So, we can write

$$\begin{aligned}
\Delta y &= D\varphi.\Delta x + D\psi^{(1)}\left(x^{(1)}\right)\alpha_1 + D\psi^{(2)}\left(x^{(2)}\right)\alpha_2 + D\psi^{(3)}\left(x^{(3)}\right)\alpha_3 + \alpha_4,\\
\Delta y &= D\varphi.\Delta x + \left(-\frac{1}{n}\right)\bar{x}\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}\left(y_j-\bar{y}\right)\varepsilon + \left(-\frac{1}{n}\right)\bar{y}\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}\left(x_i-\bar{x}\right)\varepsilon'\\
&+\frac{1}{n}\left(x_1-\bar{x}\right)\varepsilon_1\sum_{j=1}^{m} n_{1j}\left(y_j-\bar{y}\right)+\cdots+\frac{1}{n}\left(x_k-\bar{x}\right)\varepsilon_k\sum_{j=1}^{m} n_{kj}\left(y_j-\bar{y}\right)\\
&+\frac{1}{n}\left(y_1-\bar{y}\right)\varepsilon_1'\sum_{i=1}^{k} n_{i1}\left(x_i-\bar{x}\right)+\cdots+\frac{1}{n}\left(y_m-\bar{y}\right)\varepsilon_m'\sum_{i=1}^{k} n_{im}\left(x_i-\bar{x}\right)\\
&+\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i-\bar{x}\right)\left(y_j-\bar{y}\right)n_{ij}\varepsilon_{k+1}+\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i-\bar{x}\right)\left(y_j-\bar{y}\right)n_{ij}\varepsilon_{m+1}'\\
&+\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{m}\left(x_i-\bar{x}\right)\left(y_j-\bar{y}\right)\varepsilon''.
\end{aligned}$$

It is obviously seen that II>I and this is achieved simply through the opening of the above relations, so, the one algorithm is much more reliable numerically.
Therefore, the proof is completed.

Computation of the total effect of rounding error for algorithms

In this section our aim is to compute covariance with the use of elementary maps. If we consider algorithm 1, then we can state it by the following decomposition

$$\mathrm{C=cov(x,y)}=\frac{\sum_{i=1}^{k}\sum_{j=1}^{m} n_{ij}x_i y_j}{n}-\bar{x}\,\bar{y}$$

Therefore, we have:

$$\begin{aligned}
x^{(0)} &= \left(x_1,\cdots,x_k,y_1,\cdots,y_m\right)\\
x^{(1)} &= \left(x_1,\cdots,x_k,y_1,\cdots,y_m,x_1+x_2\right)\\
&\vdots\\
x^{(k)} &= \left(x_1,\cdots,x_k,y_1,\cdots,y_m,\bar{x}\right)\\
x^{(k+1)} &= \left(x_1,\cdots,x_k,y_1,\cdots,y_m,\bar{x},y_1+y_2\right)\\
&\vdots\\
x^{(k+m)} &= \left(x_1,\cdots,x_k,y_1,\cdots,y_m,\bar{x},\bar{y}\right)\\
x^{(k+m+1)} &= \left(x_1+x_2,y_1,\cdots,y_m,\bar{x},\bar{y}\right)\\
&\vdots\\
x^{(2k+m-1)} &= \left(\sum_{i=1}^{k}x_i,y_1,\cdots,y_m,\bar{x},\bar{y}\right)\\
x^{(2k+m)} &= \left(\sum_{i=1}^{k}x_i,y_1+y_2,\bar{x},\bar{y}\right)\\
&\vdots\\
x^{(2k+2m-2)} &= \left(\sum_{i=1}^{k}x_i,\sum_{j=1}^{m}y_j,\bar{x},\bar{y}\right)\\
x^{(2k+2m-1)} &= \left(\sum_{i=1}^{k}\sum_{j=1}^{m}x_i y_j,\bar{x},\bar{y}\right)
\end{aligned}$$

$$x^{(2k+2m)} = \left(\sum_{i=1}^{k} \sum_{j=1}^{m} x_i y_j n_{ij}, \overline{x}, \overline{y} \right)$$

$$x^{(2k+2m+1)} = \left(\sum_{i=1}^{k} \sum_{j=1}^{m} x_i y_j n_{ij}, \overline{xy} \right)$$

$$x^{(2k+2m+2)} = \left(\frac{1}{n} \sum_{i=1}^{k} \sum_{j=1}^{m} x_i y_j n_{ij}, \overline{xy} \right)$$

$$x^{(2k+2m+3)} = \left(\frac{1}{n} \sum_{i=1}^{k} \sum_{j=1}^{m} x_i y_j n_{ij} - \overline{xy} \right).$$

Also, the following decomposition is given for algorithm 2, if we consider

$$\text{C=cov(x,y)=}\frac{1}{n} \sum_{i=1}^{k} \sum_{j=1}^{m} n_{ij} (x_i - \overline{x})(y_j - \overline{y})$$

$$x^{(0)} = \left(x_1, \cdots, x_k, y_1, \cdots, y_m \right)$$

$$x^{(1)} = \left(x_1, \cdots, x_k, y_1, \cdots, y_m, x_1 + x_2 \right)$$

$$\vdots$$

$$x^{(k)} = \left(x_1, \cdots, x_k, y_1, \cdots, y_m, \overline{x} \right)$$

$$x^{(k+1)} = \left(x_1, \cdots, x_k, y_1, \cdots, y_m, \overline{x}, y_1 + y_2 \right)$$

$$\vdots$$

$$x^{(k+m)} = \left(x_1, \cdots, x_k, y_1, \cdots, y_m, \overline{x}, \overline{y} \right)$$

$$x^{(k+m+1)} = \left(x_1 - \overline{x}, x_2, \cdots, x_k, y_1, \cdots, y_m, \overline{x}, \overline{y} \right)$$

$$\vdots$$

$$x^{(2k+m)} = \left(x_1 - \overline{x}, \cdots, x_k - \overline{x}, y_1, \cdots, y_m, \overline{x}, \overline{y} \right)$$

$$x^{(2k+m+1)} = \left(x_1 - \overline{x}, \cdots, x_k - \overline{x}, y_1 - \overline{y}, y_2, \cdots, y_m, \overline{y} \right)$$

$$\vdots$$

$$x^{(2k+2m)} = \left(x_1 - \overline{x}, \cdots, x_k - \overline{x}, y_1 - \overline{y}, \cdots, y_m - \overline{y} \right)$$

$$x^{(2k+2m+1)} = \left(\left(x_1 - \overline{x} \right) + \left(x_1 - \overline{x} \right), \cdots, \left(x_k - \overline{x} \right), y_1 - \overline{y}, \cdots, y_m - \overline{y} \right)$$

$$\vdots$$

$$x^{(3k+2m-1)} = \left(\sum_{i=1}^{k} \left(x_i - \overline{x} \right), y_1 - \overline{y}, \cdots, y_m - \overline{y} \right)$$

$$x^{(3k+2m)} = \left(\sum_{i=1}^{k} \left(x_i - \overline{x} \right), \left(y_1 - \overline{y} \right) + \left(y_2 - \overline{y} \right), \cdots, y_m - \overline{y} \right)$$

$$\vdots$$

$$x^{(3k+3m-2)} = \left(\sum_{i=1}^{k} \left(x_i - \overline{x} \right), \sum_{j=1}^{m} \left(y_j - \overline{y} \right) \right)$$

$$x^{(3k+3m-1)} = \left(\sum_{i=1}^{k} \sum_{j=1}^{m} \left(x_i - \overline{x} \right) \left(y_j - \overline{y} \right) \right)$$

$$x^{(3k+3m)} = \left(\sum_{i=1}^{k} \sum_{j=1}^{m} (x_i - \overline{x})(y_j - \overline{y}) n_{ij} \right)$$
$$x^{(3k+3m+1)} = \left(\frac{1}{n} \sum_{i=1}^{k} \sum_{j=1}^{m} (x_i - \overline{x})(y_j - \overline{y}) n_{ij} \right).$$

It is concluded that if we have $\{y_j\}_{j=1}^{m}$ and, $\{x_i\}_{i=1}^{k}$ then by using Algorithm 1 , we have (2k+2m+3) elementary maps. Also, for Algorithm 2 we have (3k+3m+1) elementary maps.

Therefore, the second algorithm has also much more stages rather then the first algorithm, and has much more errors.

Now that we have achieved the more reliable algorithm for the estimation of the covariance and since we knew that which algorithm has been much reliable for the variance, so we can reach the preferred algorithm for the correlation coefficient.

Therefore, the following algorithm is much more reliable numerically for the correlation coefficient:

$$r = \frac{(\sum_{i=1}^{k} \sum_{j=1}^{m} n_{ij} x_i y_j) - n\overline{xy}}{\sqrt{\sum_{i=1}^{k} n_i (x_i - \overline{x})^2 \sum_{j=1}^{m} n_j (y_j - \overline{y})^2}}.$$

RESULTS AND DISCUSSION

In this follows we run two algorithms on two sets of data. Also, we assume that the set of numbers is IF (10,4,L,U). Our aim is to calculate the relative errors of the two algorithms in the set IF(10,6,L,U).

Remark 1.e_1 and e_2 denote the relative error for algorithm 1and 2, respectively.

Example 1. We assume that the set of numbers and observations are as follows:

$x_1 = 0.1345, x_2 = 0.2123, x_3 = 0.3104.$

$y_1 = 0.1254, y_2 = 0.2213, y_3 = 0.3041, n = 3.$

In the set IF (10,4,L,U) we obtain:

$\overline{x} = 0.2191$ and $\overline{y} = 0.2169$ and in the set IF (10,6,L,U)we obtain

$\overline{x} = 0.219066$ and $\overline{y} = 0.216933$

In the set IF (10,4,L,U) we obtain for Algorithm 1 in Table 1.1:

Table 1.1 Result of Algorithm 1for IF (10,4,L,U).

x_i	y_j	x_1y_j	x_2y_j	x_2y_j	n_{1i}	n_{2i}	n_{3i}
0.1345	0.1254	0.0169	0.0266	0.0389	1	2	1
0.2123	0.2213	0.0297	0.0470	0.0686	2	4	2
0.3104	0.3041	0.0409	0.0646	0.0944	3	1	4

also, in this set for Algorithm 2 we have in Table 1.2:

Table 1.2 Result of Algorithm 2 for IF (10,4,L,U).

x_i	$x_i - \overline{x}$	y_j	$y_j - \overline{y}$	n_{1i}	n_{2i}	n_{3i}
0.1345	–0.0846	0.1254	–0.0915	1	2	1
0.2123	–0.0068	0.2213	0.0044	2	4	2
0.3104	0.0913	0.3041	0.0872	3	1	4

and in the set IF (10,6,L,U) we obtain for Algorithm 1 in Table 1.3:

Table 1.3 Result of Algorithm 1 for IF (10,6,L,U).

x_i	y_j	$x_1 y_j$	$x_2 y_j$	$x_2 y_j$	n_{1i}	n_{2i}	n_{3i}
0.1345	0.1254	0.016866	0.026622	0.038924	1	2	1
0.2123	0.2213	0.029765	0.046981	0.068691	2	4	2
0.3104	0.3041	0.040901	0.064560	0.094392	3	1	4

And finally, in this set we can obtain the quantities of Algorithm 2 in Table1.4:

Table 1.4 Result of Algorithm 2 for IF (10,6,L,U).

x_i	$x_i - \overline{x}$	y_j	$y_j - \overline{y}$	n_{1i}	n_{2i}	n_{3i}
0.1345	–0.08566	0.1254	–0.091533	1	2	1
0.2123	–0.006766	0.2213	0.004367	2	4	2
0.3104	0.091334	0.3041	0.087167	3	1	4

We obtain the relative errors in Table 1.5 for two algorithms:

Table 1.5 Comparison between two algorithms for Example 1.

	Alg. 1	Alg. 2
Covariance of observations in IF (10,4,L,U)	$\bar{C}$= 0.3053	$\bar{C}$ = 0.0032
Covariance of observations in IF (10,6,L,U)	C = 0.242736	C = 0.005991
Relative error in IF (10,6,L,U)	e_1=-0.6947	e_2=0.9968

Example 2.We assume that the set of numbers and observations are as follows:

$$x_1 = 0.1123, x_2 = 0.1234, x_3 = 0.1356.$$

$$y_1 = 0.1113, y_2 = 0.1124, y_3 = 0.1135, n = 3.$$

In the set IF (10,4,L,U) we have:

$\bar{x}$ = 0.1238 and $\bar{y}$ = 0.1124 and in the set IF (10,6, L, U)we obtain

$\bar{x}$ = 0.123767 and $\bar{y}$ = 0.112400

In the set IF (10,4,L,U) we obtain for Algorithm 1in Table 2.1:

Table 2.1 Result of Algorithm 1for IF (10,4,L,U).

x_i	y_j	$x_1 y_j$	$x_2 y_j$	$x_2 y_j$	n_{1i}	n_{2i}	n_{3i}
0.1123	0.1113	0.0125	0.0137	0.0151	3	2	1
0.1234	0.1124	0.0126	0.0139	0.0152	2	0	2
0.1356	0.1356	0.1523	0.0167	0.0184	1	1	2

also in this set for Algorithm 2 we have in Table 2.2:

Table 2.2 Result of Algorithm 2 for IF (10,4,L,U).

x_i	$x_i - \overline{x}$	y_j	$y_j - \overline{y}$	n_{1i}	n_{2i}	n_{3i}
0.1123	–0.0115	0.1113	–0.0011	3	2	1
0.1234	–0.0004	0.1124	0	2	0	2
0.1356	0.0118	0.1356	0.0011	1	1	2

and in the set IF (10,6,L,U) we obtain for Algorithm 1in Table 2.3:

Table 2.3 Result of Algorithm 1 for IF (10,6,L,U).

x_i	y_j	$x_1 y_j$	$x_2 y_j$	$x_2 y_j$	n_{1i}	n_{2i}	n_{3i}
0.1123	0.1113	0.012499	0.013734	0.015092	3	2	1
0.1234	0.1124	0.012623	0.013870	0.015241	2	0	2
0.1356	0.1356	0.152279	0.016733	0.018387	1	1	2

And finally, in this set we can obtain the quantities of Algorithm 2 in Table2.4:

Table 2.4 Result of Algorithm 2 for IF (10,6,L,U).

x_i	$x_i - \overline{x}$	y_j	$y_j - \overline{y}$	n_{1i}	n_{2i}	n_{3i}
0.1123	−0.011467	0.1113	−0.0011	3	2	1
0.1234	−0.000367	0.1124	0	2	0	2
0.1356	0.011833	0.1356	0.0011	1	1	2

We obtain the relative errors in Table 2.5 for two algorithms:

Table 2.5 Comparison between two algorithms for Example 2.

	Alg. 1	Alg. 2
Covariance of observations in IF(10,4,L,U)	$\bar{C}$ = 0.002647	$\bar{C}$ = 0
Covariance of observations in IF(10,6,L,U)	C = 0.0026	C = 0.000013
Relative error in IF(10,6,L,U)	e_1=-0.0180	e_2=1

Conclusion

Computational correlation coefficient is playing on ones more important role in the testing independent random sequences in lattice and QMC, therefore, we try to use this work for the spatial numerical results that reminds a challenge in lattice-Nyström and sigmoidal-Nyström methods. If deserves special attention and will be consideration elsewhere (see[12,13]).

Acknowledgments

The authors are very grateful to anonymous reviewers for valuable comments.

REFERENCES

[1] R. L. Ashenhurst & N. Metropolis, Un-normalized floating-point arithmetic, J. Assoc. Comput. Math. 6(1959) , 415-428.

[2] F. L. Bauer , Computational graphs and rounding error, SIAM J. Numer. Anal. 11 (1974), 87-96.

[3] F. L. Bauer , J. Heinhold, K. Samelson & R. Sauer, Modern Rechenanlagen, Teubner, Stuttgart, 1965.

[4] H. Henrici, Error Propagation for Difference Methods, Wiley, New York, 1963.

[5] D. E. Knuth, The Art of Computer Programming, Semi-numerical Algorithms, vol. 2, Addision-Wesley, Readig, MA, 1969.

[6] U. Kulisch, Grundzgüe der Intervallrechning, in: D. Lauugwitz (Ed.) , Überblicke Mathematik, Vol. 2, Bibliographiches Institut, Mannheim, (1969), 51-98.

[7] R. E. Moore, Interval Analysis, Prentice-Hall, Englewood Cliff, NJ, 1966.

[8] H. Niederreiter, Random number generation and quasi-Monte Carlo methods, Siam, 1992.

[9] H. A. Rademacher , On the accumulation of errors in processes of integration on the high-speed calculating machines, in Proceedings of a Symposium on Large Scale Digital Calculating Machinery, Annals Comput. Labor. Harvard University, 16 (1948), 176-185.

[10] D. Rostamy V.F., A Stochastic Partial Differential Equation for Computational Algorithms, Appl. Math. Comput., 159 (2004), 429-434.

[11] D. Rostamy and E. Yaghesh, Comparison of numerically stability of two algorithms for calculation of variance, Journal of Sci. Islamic Rep. of Iran, 21(2010), 265-272.

[12] D. Rostamy, M. Jabbari & M. Gadirian, Lattice-Nyström Method for Solving Integro-Differential Equations with the Initial Conditions, Submitted paper.

[13] D. Rostamy, and M. Jabbari, Sigmoidal- Nyström for solving an aerofoil equation, Applied Mathematics and Computation, 218(2012) 8492-8502

[14] O. Stauning , Automatic Validation of Numerical Solutions, Ph. D. Thesis, Technical University of Denmark, Lyngby, 1997.

[15] V. Sinescu, Shifted lattice rules based on a general weighted discrepancy for integrals over Euclidean space, J. Comput Appl. Math 232 (2009) , no. 2, 240-251.

[16] J. Stoer & R. Bulirsch, Introduction to Numerical Analysis, Springer-Verlag, New York, 1993.

[17] H. M. Taylor& S. Karlin, An Introduction to Stochastic Modeling, Academic Press, New York, 1984.

[18] S. R. Varadhan, Multiplicational Differential Processes, Springer-Verlag, New York,1996.

[19] J. H. Wilkinson, Error analysis of floating-point computation, Numer. Math. 2 (1960), 219-340.

[20] J. H. Wilkinson, Rounding Errors in Algebraic Processes, Wiley, New York, 1963.

[21] J. H. Wilkinson, The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1965.