Vol.2017-ARC-225 No.19
Vol.2017-SLDM-179 No.19
Vol.2017-EMB-44 No.19
2017/3/9

Stochastic Computingにおける相関の許容によるSNGの削減

渡邉 朗弘[1,a)]　山下 茂[2,b)]

概要：Stochastic Computing（SC）は．従来の二進法の演算に代わる低コストな計算手法である．SC は，Stochastic Number（SN）というビット列で演算を行うが，SN の間に相関がある場合，正しい演算結果が得られない．SN 生成器（SNG）を共有すると相関が発生するため．生成したい SN につき，それぞれ独立した SNG を用いる．しかし，誤差が発生したとしても，最終的な出力に与える影響が非常に小さい箇所ならば，SNG を共有することができると考えられる．本稿では，相関の発生によって生じる誤差の大小に作用する要因を分析し，出力の誤差を抑えつつ，SNG を共有することで SNG の数を削減した SC 回路を実現する手法を提案する．実験の結果，本手法を用いることで，ランダムに SNG を削減した場合に比べて，誤差を平均約 0.327 倍に抑えつつ，SNG を削減できた．また，SNG を共有しない回路の出力誤差に近い値を保つことができた．つまり，本手法により，許容誤差内に収めながら，回路面積を削減できることを示した．

1. はじめに

Stochastic Computing（以降，SC）は，1960 年代に提案された，従来の二進法の演算に代わる低コストな計算手法である [1]．SC の利点として，従来の計算手法よりも低面積，低消費電力な回路の設計が可能であることや，ソフトエラー耐性に優れていることなどが挙げられる [2][3]．従来の回路の微細化に伴い，上記のような利点を持つ SC が再注目されている [4][5]．また，応用分野として，画像処理やニューラルネットワークといった分野での利用が考えられている [6][7]．

SC では，Stochastic Number（以降，SN）と呼ばれるビット列における 1 の存在確率によって表現される数値を用いて演算を行う．SC の演算において，ゲートに入力されるビット列の間に相関がある場合，正しい演算結果が得られないという問題がある [8]．そのため，SC ではできるだけ相関の発生を抑え，演算誤差が生じないように回路を設計する必要がある．例えば，同一の SN 生成器を再利用すると，その生成器から生成された SN 同士には相関が発生するため，生成したい SN につき，それぞれ独立した SN 生成器を用意し，相関の発生を回避する．

しかし，この相関による演算誤差の問題は，一般にゲート単体レベルでしか考慮されていない．そのため，回路全体で見たとき，誤差を許容できる箇所が存在する可能性があり，そのような場合には，相関の発生を無視した設計をすることで，回路面積を削減できると考えられる．

本稿では，相関を許容しながら出力の誤差を抑えた SC 回路の実現手法を提案する．SC で行いたい計算において，演算の種類や入力変数の各々の出現数，SN の値の大小などによって相関が発生した際の影響は異なる．相関が存在したとしても誤差が小さい条件を調査し，誤差を抑えつつ SN 生成器の削減を図る．

2. Stochastic Computing

2.1 Stochastic Number

SC では，数値はビット列内の 1 の存在確率によって表現される．例えば，ビット列 01001100 は，8 ビットのうち 1 の数が 3 個存在するので 3/8 を表す．SC において，このように数値を表現したビット列を Stochastic Number（SN）と呼ぶ．SN は，ビット列の並びが異なっていても，ビット列内の 1 の存在確率が等しければ同じ値を表す．例えば，01001100 と 11100000 という SN は，どちらも 3/8 を表す．一般的に，SN はビット列の長さが長いほど，数値表現の精度がよくなる [9]．

SN の生成方法として，乱数発生器から出力された乱数と二進数の値を比較することで行う方法がある．図 1 に，乱数発生器として Linear Feedback Shift Register（以降，

1 立命館大学大学院情報理工学研究科
2 立命館大学情報理工学部
a) mineral@ngc.is.ritsumei.ac.jp
b) ger@cs.ritsumei.ac.jp

Vol.2017-ARC-225 No.19
Vol.2017-SLDM-179 No.19
Vol.2017-EMB-44 No.19
2017/3/9

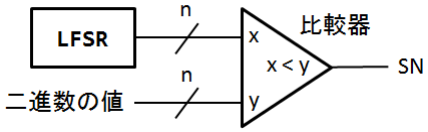

図 1　LFSR を用いた SNG

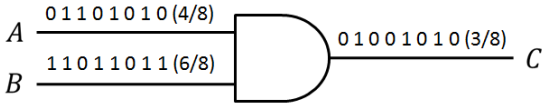

図 2　SC における乗算の例

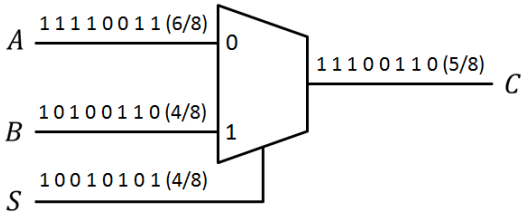

図 3　SC における加算の例

LFSR）を用いた，SN 生成器の例を示す．LFSR は，周期 2^n-1 で一回の動作につき n ビットの異なる値を発生させる．比較器は，LFSR から出力される値よりも二進数の値の方が大きければ 1 を，そうでなければ 0 を出力する．したがって，2^n-1 回動作させることで，n ビットの二進数の値から 2^n-1 ビットの SN を得ることができる．また，SN 生成器のことを Stochastic Number Generator（以降，SNG）と呼ぶ．

以降では，数値 x, y を値とする SN を，それぞれ対応する大文字の X, Y で表す．また，X, Y 内の 1 の存在確率をそれぞれ $Prob(X=1), Prob(Y=1)$ で表す．このとき，$Prob(X=1)=x, Prob(Y=1)=y$ となる．

2.2　SC における演算

本節では，SC における演算方法について述べる．SC の演算は簡単な論理ゲートで実現することができる．

まず，乗算について述べる．乗算は AND ゲート 1 つで実現することができる．例を図 2 に示す．図 2 では，AND ゲートに $a=4/8, b=6/8$ である A, B を入力した結果，$c=3/8$ である C が出力されており，$a \times b = c$ が実現できている．

次に，加算について述べる．加算はマルチプレクサ 1 つで実現することができる．マルチプレクサの制御入力に 1/2 を表す SN を入力することで，1/2 にスケールされた加算結果を得ることができる．例を図 3 に示す．図 3 では，マルチプレクサに $a=6/8, b=4/8, s=4/8$ である A, B, S を入力した結果，$c=5/8$ である C が出力されており，$1/2(a+b)=c$ が実現できている．

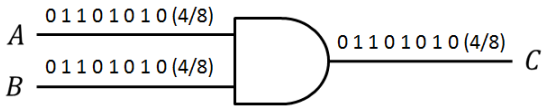

図 4　相関が存在する乗算の例

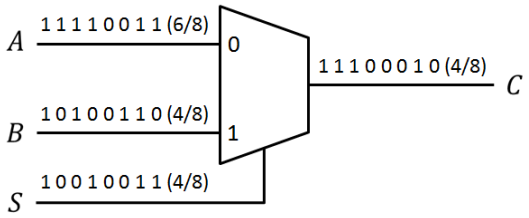

図 5　相関が存在する加算の例

2.3　SC における相関による問題

SC では，上記に示したように，簡単な論理ゲートで演算を行うことができる．しかし，論理ゲートに入力される SN に相関が存在すると，正しい演算結果が得られないという問題がある．乗算における例を図 4 に示す．図 4 では，$4/8 \times 4/8$ という乗算を行っているが，乗算結果は正しい演算結果の 2/8 ではなく 4/8 という誤った値となっている．また，加算における例を図 5 に示す．加算は，マルチプレクサの制御入力の SN である S と，入力の SN である A, B に相関が存在すると，正しい結果が得られない [10][11]．図 5 では，$6/8+4/8$ という加算を行っているが，加算結果は正しい演算結果の 5/8 ではなく 4/8 という誤った値となっている．ただし，入力の SN である A, B 間の相関は，加算結果に影響を及ぼさないことが知られている [10][11]．

上記の問題を解消するため，生成したい SN につき，それぞれ独立した SNG を用意する必要がある．これは，同一の SNG を再利用して SN を生成した場合，その SNG から生成された SN 間には相関が存在するからである．例えば，構造も初期値も同一の LFSR である SNG を用いてある数値，$x, y(x<y)$ の SN を生成することを考える．このとき，x と y は同一の乱数値と比較されるため，x を入力した SNG の出力が 1 のとき，y においても SNG の出力は必ず 1 となる．このため，x, y から得られる SN には相関が存在する．このように，同一の SNG を再利用すると，相関の発生につながる．したがって，独立した SNG を用いることで，相関の発生を抑制し，演算精度の低下の問題を緩和する．

しかし，相関による演算誤差の問題は，一般にゲート単体レベルでしか考慮されていない．そのため，回路全体で見たとき，相関による最終的な出力結果への影響の大小は，場合によって異なると考えられる．すなわち，最終的な出力結果への誤差による影響が非常に小さい所が存在し，そこでは相関を許容できる可能性がある．このことを利用することによって SNG を減らし，回路面積を削減できると考えられる．

Vol.2017-ARC-225 No.19
Vol.2017-SLDM-179 No.19
Vol.2017-EMB-44 No.19
2017/3/9

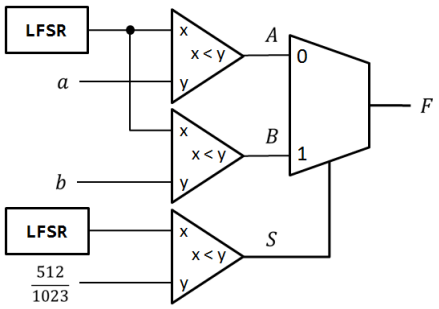

図 6 A と B を同一の LFSR で生成する場合

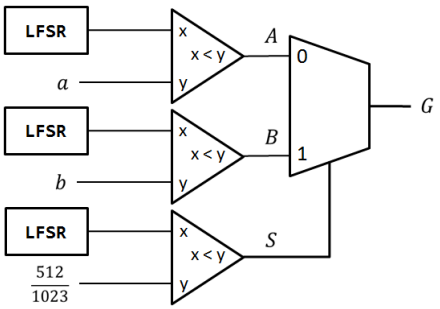

図 7 全て独立の場合

3. 相関を許容した SC の実現手法

相関を許容できる箇所を見つけ，SNG の削減を実現するため，相関によって発生する誤差の大小に作用する要因を調査した．本章では，その要因について説明する．

3.1 マルチプレクサの入力の相関による出力への影響力

2.3 節で述べたように，マルチプレクサは，加算したい値が入る入力（以降，データ入力）の SN 間の相関は，加算結果に影響を及ぼさないことが知られている [10][11]．これにより，データ入力の SN は，異なる SNG から生成しても，同一の SNG から生成しても，どちらでもかまわないという認識が一般的となっている．しかし，本研究において，異なる SNG よりも同一の SNG を用いてデータ入力の SN を生成すべきであることを発見した．

以下では，図 6 と図 7 の 2 つの場合において，$A+B$ の加算を考える．図 6 では，A と B を構造と初期値が同一の LFSR を用いて生成する．図 7 では，A と B を異なる LFSR を用いて生成する．ここで，LFSR は図 8 の 10bit 構造で，1 から 1023 までの乱数を生成できるとする．また，図 6 の a と図 7 の a，図 6 の b と図 7 の b は同一の値とする．さらに，図 6 の S を生成するための LFSR と図 7 の S を生成するための LFSR の構造と初期値も同一とする．このとき，ある $a, b(1/1023 \leq a, b \leq 1023/1023)$ における $F = A + B, G = A + B$ の計算を，生成に用いる LFSR の初期値が異なる，1023 通りの S すべてにおいて行う．この 1023 回の計算において，A, B は同一とする．そして，上記の加算を，A, B の値と A, B の生成時に用いる LFSR の初期値を変え，100 回行う．実験結果を図 9 に示す．実線は，F の誤差の方が G の誤差よりも小さい場合を示す．破線は，F の誤差の方が G の誤差よりも大きい場合を示す．横軸は 100 通りのテストケースの番号を示し，縦軸は上記の場合の発生数を示す．この図より，A と B を構造と初期値が同一の LFSR を用いて生成する方が，A と B を異なる LFSR を用いて生成するよりも，誤差が小さくなる確率が高いことがわかる．つまり，マルチプレクサのデータ入力において，回路面積の削減だけでなく，誤差の減少の観点からも，LFSR を共有するべきであるといえる．

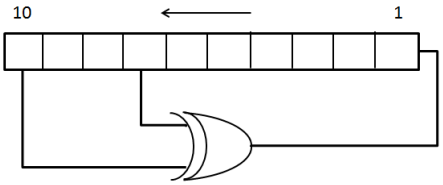

図 8 10bitLFSR

これは，マルチプレクサの構造によるものであると推察される．図 10 にマルチプレクサの構造を示す．ここで，NOT ゲートによりビットが反転したとしても，A と S の間の相関の有無に変化はない．そのため，A と B を同一の SNG で生成した場合，S と A に相関が存在しなければ，S と B も必ず相関が存在しない．しかし，A と B を異なる SNG で生成した場合，S と A に相関が存在しなくとも，S と B に相関が存在する可能性がある．つまり，データ入力の SN を異なる SNG で生成した場合，同一の SNG で生成した場合に比べて，データ入力の SN と制御入力の SN の間に相関が生じる可能性が高くなると考えられる．したがって，上記のような結果に至ったと考えられる．

また，マルチプレクサに複雑な SN が入力されることを考える．例として，$AB + CD$ について考える．A と C を同一の LFSR で，B と D を同一の LFSR で生成する場合の出力を F とし，A, B, C, D を異なる LFSR で生成する場合の出力を G とする．また，上記以外の SN の値や S，LFSR の構造は，前述の実験と同様に F, G において同一とする．このとき，ある $a, b, c, d(1/1023 \leq a, b, c, d \leq 1023/1023)$ における $F = AB + CD, G = AB + CD$ の計算を，生成に用いる LFSR の初期値が異なる，1023 通りの S すべてにおいて行う．この 1023 回の計算において，A, B, C, D は同一とする．そして，上記の加算を，A, B, C, D の値と A, B, C, D の生成時に用いる LFSR の初期値を変え，100 回行う．実験の結果を図 11 に示す．実線は，F の誤差の方が G の誤差よりも小さい場合を示す．破線は，F の誤差の方が G の誤差よりも大きい場合を示す．横軸は 100 通りのテストケースの番号を示し，縦軸は上記の場合の発生数を示す．この図より，F, G において差は無いことがわかる．つまり，何か演算を行った結果の SN を加算する場合，他の入力の SN と演算を行っていない SN 同士を加算する

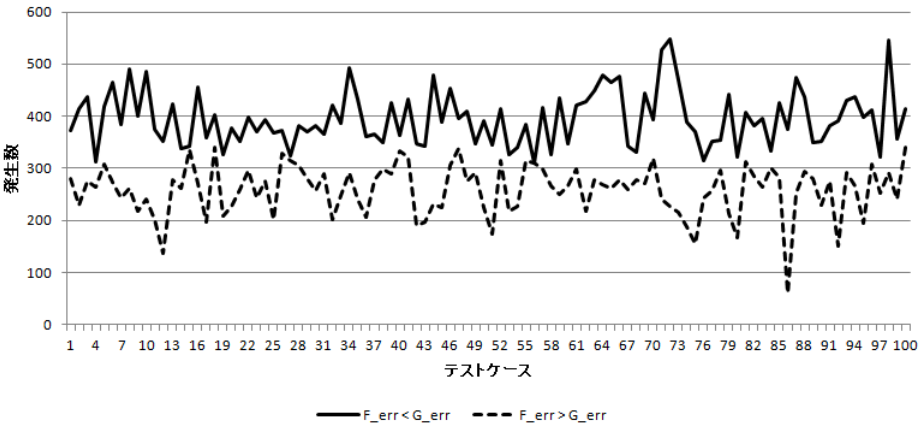

図 9 誤差発生の数の比較

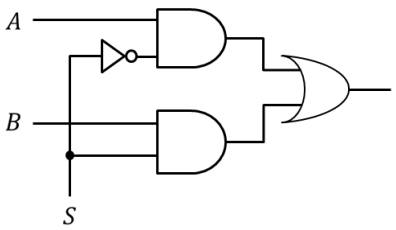

図 10 マルチプレクサの構造

場合に比べて，誤差の減少に関する影響力は小さいと考えられる．

3.2 SN の値による出力への影響力

この節では，SN の値が与える出力への影響について述べる．相関の存在する SN を用いて乗算や加算を行うとき，その SN の値に応じた誤差の変化について考える．

まず，乗算 $A \times B$ について考える．ここで，A, B を同一の SNG で生成するとする．また，$1/15 \leq a, b \leq 14/15$ とし，SNG における LFSR は図 12 とする．最大値が $15/15 = 1$ でないのは，1 を掛ける場合は必ず相関がないため，除いているからである，このとき，a, b のとりうる値の組み合わせ $14 \times 14 = 196$ 通りすべての乗算を行う．実験結果を図 13 に示す．横軸は a の値を示し，縦軸はその a と 1/15 から 14/15 までの b との乗算の結果のうち，最大の誤差を示す．この図より，乗算に用いられる値が小さくなる，もしくは大きくなるにしたがって，相関による誤差は小さくなることがわかる．つまり，非常に小さい値，もしくは非常に大きい値を用いた乗算が行われる場合，SNG を共有化することができる可能性が高いといえる．

次に，加算 $A + B$ について考える．ここで，データ入力 A, B と制御入力 S を同一の SNG で生成するとする．また，$1/15 \leq a, b \leq 15/15$ とする．このとき，a, b のとりうる値の組み合わせ $15 \times 15 = 225$ 通りすべての加算を行う．実験結果を図 14 に示す．横軸は a の値を示し，縦軸はその a と 1/15 から 15/15 までの b との加算の結果のうち，最大の誤差を示す．加算は 2.2 節に述べたように，1/2 倍にスケールされて行われるため，正確な結果を得るためには 2 倍する必要がある．そのため，誤差も 2 倍になってしまう．これにより，乗算の誤差に比べて，加算は誤差が小さくなりにくくなっている．また，A と S のみを同一の SNG で生成する場合も同様に実験を行った結果．B を生成する SNG によって誤差の大小は左右されるものの，やはり乗算に比べて，誤差は大きかった．

以上の結果より，加算においては相関を発生させるべきではない．また，乗算においては非常に小さい値，もしくは非常に大きい値が用いられることがあらかじめ判明している場合ならば，相関を許容できる可能性があることがわかる．

3.3 入力変数の積項の出現回数による出力への影響力

この節では，SC の計算式において，とりうる積の組の各々の出現回数と出力への影響力の関係性を述べる．前提として，同じ値をとる SN を独立に生成することは無いこととする．これは，例えば，$F = A(B + C)$ と $G = AB + AC$ を考えたとき，G における A を $A_1, A_2 (Prob(A_1 = 1) = Prob(A_2 = 1) = a)$ のように同じ値でも異なる SN としたとしても，出力 F と G の結果は同じになるため，余分な SNG が必要になるだけであるからである．

$X = AB(C + D)$ という式について考える．ここで，2 変数がとりうる積の種類とその出現回数は，AB が 2 回，AC，BC，AD，BD が 1 回である．このとき，A と B を同一の SNG で生成する場合と，A と C を同一の SNG で生成する場合の 2 つの構造を考える．ここで，前者における出力を X_0，後者における出力を X_1 とし，入力値の範囲は 1/1023 から 1023/1023 とする．また，両方において，同一の SNG で生成する SN 以外の SN は，すべて異なる SNG で生成し，SNG における LFSR は図 8 とする．X_0, X_1 において，100 回ランダムに値を入力し，計算を行ったところ，74 回 X_0 の方が X_1

Vol.2017-ARC-225 No.19
Vol.2017-SLDM-179 No.19
Vol.2017-EMB-44 No.19
2017/3/9

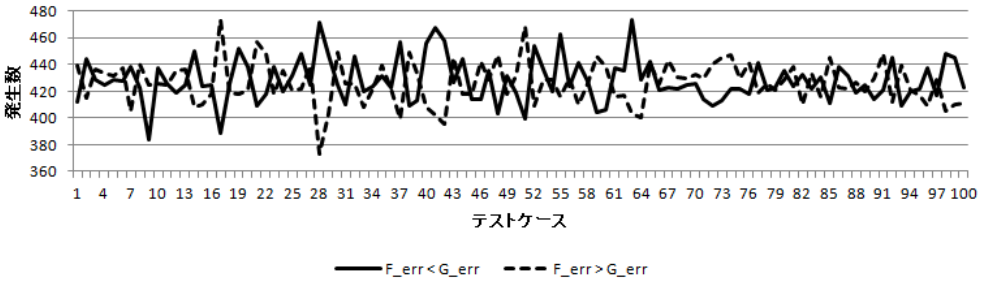

図 11 $AB + CD$ における誤差発生の数の比較

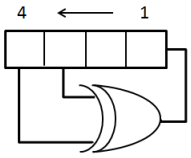

図 12 4bitLFSR

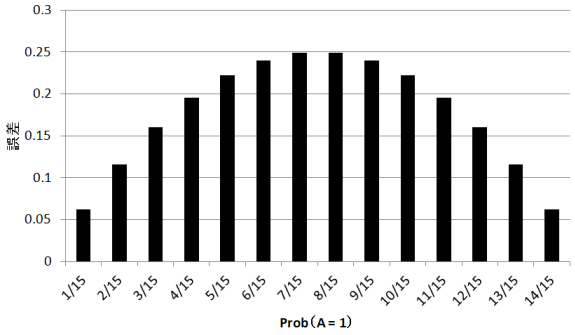

図 13 乗算における SNG 共有化による誤差

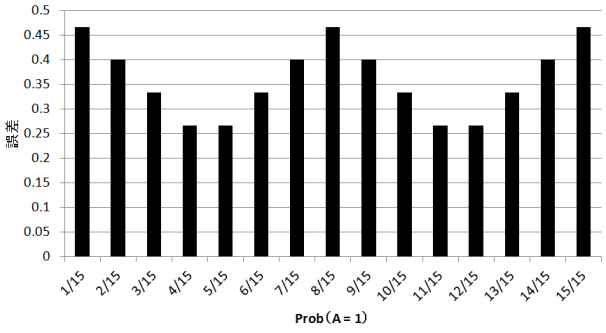

図 14 $A + B$ におけるすべての SNG 共有化による誤差

よりも誤差が大きかった．また，$Y = AB(C+D+E+F)$，$Z = AB(C+D+E+F+G+H+I+J)$ についても同様の構造で同様の計算を行った．Y において，AB は 4 回，AC は 1 回，Z において，AB は 8 回，AC は 1 回出現する．実験結果を表 1 に示す．$ERR_{AB} > ERR_{AC}$ は，A と B を同一の SNG で生成した場合における出力誤差が，A と C を同一の SNG で生成した場合における出力誤差よりも大きかった回数を表す．$ERR_{AB} < ERR_{AC}$ は，その逆を表し，$ERR_{AB} = ERR_{AC}$ は，差がなかった回数を表す．

この表 1 より，計算式において 2 変数がとりうる積の組

表 1 積項の出現回数による出力誤差の変化

計算式	$ERR_{AB} > ERR_{AC}$	$ERR_{AB} < ERR_{AC}$	$ERR_{AB} = ERR_{AC}$
X	74	23	3
Y	83	13	4
Z	89	9	2

のうち，出現回数が多い組ほど相関が存在したときの誤差が大きくなると考えられる．つまり，上記の組において，出現回数が多い組は異なる SNG を用いて SN を生成するべきであり，出現回数が少ない組ほど同一の SNG を用いて SN を生成できる可能性があるといえる．

4. 実験結果と考察

4.1 評価方法

以下の式

$$ab(\frac{20}{1023}c + d) + (e + f) + g(\frac{480}{1023}h + ijk) + (l + m)$$

を計算する回路に対して，SNG を共有しない場合，3 章で述べた手法をもとに SNG を共有した場合，ランダムに SNG を共有した場合の 3 通りの場合において計算を行い，それぞれの出力誤差を比較する．ここで，変数の値の範囲は 1/1023 から 1022/1023 とする．0 や 1 は，必ず相関が存在せず，SNG の削減による誤差の測定の妨げになると考えられるからである．また，マルチプレクサの制御入力の SN はすべて独立の SNG で生成することとし，入力変数の SN を生成する SNG のみを共有可能とする．SNG における LFSR は図 8 とする．この比較により，3 章で述べた手法が，誤差を抑えつつ SNG を削減することに有用であることを示す．

4.2 実験結果と考察

上記の式に対して，マルチプレクサのデータ入力の SN 間の相関は無視できることを踏まえながら，提案手法を適用した．その結果，$a, e, f, 480/1023, i, j, l, m$ を値とする SN を生成する SNG の共有，b, h, k を値とする SN を生成する SNG の共有，$20/1023, c, d, g$ を値とする SN を生成する SNG の共有を行うことができた．これにより，15 個の SNG を 3 個に削減した．

上記の提案手法を適用した場合，全く SNG を共有しな

Vol.2017-ARC-225 No.19
Vol.2017-SLDM-179 No.19
Vol.2017-EMB-44 No.19
2017/3/9

表 2　出力誤差の平均の比較

テストケース	独立（15）	提案（3）	ランダム（3）
1	0.007638	0.006411	0.040363
2	0.006361	0.007958	0.020377
3	0.007388	0.009112	0.022077
4	0.007928	0.006362	0.011855
5	0.007717	0.007496	0.028987
6	0.007012	0.006429	0.032122
7	0.005643	0.005875	0.018473
8	0.007857	0.007644	0.032956
9	0.007448	0.005826	0.011501
10	0.007709	0.008670	0.022686
11	0.006303	0.005780	0.019718
12	0.006210	0.006622	0.014970
13	0.006396	0.007976	0.021445
14	0.005859	0.007042	0.020804
15	0.009140	0.007534	0.018628
16	0.004536	0.005699	0.013791
17	0.004659	0.006549	0.020695
18	0.007980	0.007644	0.018804
19	0.005651	0.005012	0.019543
20	0.005269	0.006167	0.011767
平均	0.006735	0.006890	0.021078

い場合，ランダムに SNG を共有して 3 個に削減した場合において，入力変数にランダムな値を入力した計算を 100 回行い，それぞれの場合における出力誤差の平均を算出した．そして，この計算 1 回を 1 回のテストとし，LFSR の初期値を変えて 20 回のテストを行った．

実験結果を表 2 に示す．「独立」は全く SNG を共有していない場合を，「提案」は 3 章で述べた手法をもとに SNG を共有した場合を，「ランダム」はランダムに SNG の共有箇所を決定した場合を表し，付随の括弧付きの数字は，それぞれの場合における SNG の数を表す．また，最後の行は，1 から 20 のテストケースの平均値を表す．表 2 より，提案手法で SNG を共有した方が，ランダムに SNG を共有した場合よりも出力誤差が小さくなることがわかる．提案手法の場合による誤差は，ランダムの場合による誤差よりも，平均約 0.327 倍小さい．また，提案手法の場合による誤差と独立の場合による誤差の差は平均 0.000155 であり，ランダムの場合による誤差と独立の場合による誤差の差は平均 0.014343 である．つまり，誤差を抑えながら SNG を削減できていることがわかる．さらに，独立の場合よりも提案の場合のほうが誤差が小さいテストケースがある．これは，3.1 節で述べた考えにより，誤差が減少しているからであると考えられる．

5. おわりに

本論文では，相関の発生による誤差の大小に作用する要因を調査し，その要因を利用することで，相関を許容し，SNG を削減しながら出力の誤差を抑えた SC 回路の実現手法を提案した．実験では，ある計算式において，提案手法を用いることで，相関による誤差を抑制しつつ，SNG を削減できるかどうかを確認した．実験の結果，20 回行ったテスト全てにおいて，提案手法を用いて SNG を削減した回路の方が．ランダムに SNG を削減した回路よりも，出力誤差が小さく，平均約 0.326 倍の小ささであった．また，SNG を削減していない回路と比べても，出力誤差は近い値を保っていた．これより，提案手法は，誤差を抑えながら SNG を削減する方法として有用であると考えられる．しかし，3.1 節から 3.3 節の手法のどれを優先的に適応させるべきかについてが考えられていない．そのため，どの手法を優先した SNG の共有を行うかによって，削減できる SNG の数や誤差が異なると考えられる．したがって，3.1 節から 3.3 節の手法の優先度を調査し，SNG の削減量と誤差の抑制の両方を考えたときの最適解を得られるようにすることが，今後の課題である．

参考文献

[1] Gaines, B.: Stochastic computing systems, *Advances in information systems science*, pp. 37–172 (1969).
[2] Borkar, S., Karnik, T. and De, V.: Design and reliability challenges in nanometer technologies, *Proceedings of the 41st annual Design Automation Conference*, p. 75 (2004).
[3] Qian, W., Li, X., Riedel, M. D., Bazargan, K. and Lilja, D. J.: An architecture for fault-tolerant computation with stochastic logic, *IEEE Transactions on Computers*, Vol. 60, No. 1, pp. 93–105 (2011).
[4] Alaghi, A. and Hayes, J. P.: Survey of stochastic computing, *ACM Transactions on Embedded Computing Systems*, Vol. 12, No. 2, p. 92 (2013).
[5] Li, P.: Analysis, design, and logic synthesis of finite-state machinebased Stochastic computing, Phd thesis, The University of Minnesota (2013).
[6] Alaghi, A., Li, C. and P.Hayes, J.: Stochastic Circuits for Real-Time Image-Processing Applications, *Design Automation Conference (DAC)*, No. 136, pp. 1–6 (2013).
[7] Kim, K., Kim, J., Yu, J., Seo, J., Lee, J. and Choi, K.: Dynamic energy-accuracy trade-off using stochastic computing in deep neural networks, *Design Automation Conference (DAC)*, No. 124, pp. 1–6 (2016).
[8] Parhi, M., Riedel, M. D. and Parhi, K. K.: Effect of bit-level correlation in stochastic computing, *Digital Signal Processing (DSP), 2015 IEEE International Conference on*, IEEE, pp. 463–467 (2015).
[9] Qian, W. and Riedel, M. D.: The synthesis of robust polynomial arithmetic with stochastic logic, *2008 45th ACM/IEEE Design Automation Conference*, pp. 648–653 (2008).
[10] Ichihara, H., Ishii, S., Sunamori, D., Iwagaki, T. and Inoue, T.: Compact and accurate stochastic circuits with shared random number sources, *IEEE 32nd International Conference on Computer Design (ICCD)*, pp. 361–366 (2014).
[11] 石井章太，砂盛大貴，市原英行，　岩垣剛，井上智生：相関を持つストカスティック数の演算精度に与える影響に関する考察，電子情報通信学会技術研究報告. VLD, VLSI 設計技術，　Vol. 113, No. 454, pp. 79–84 (2014).