Twisted GFSR Generators II

Makoto Matsumoto
Research Institute for Mathematical Sciences
Kyoto University, Kyoto 606, Japan
and
Yoshiharu Kurita
National Research Laboratory of Metrology
Tsukuba 305, Japan

December 2, 1992

Abstract

The twisted GFSR generators proposed in a previous paper have a defect in k-distribution for k larger than the order of recurrence. In this follow-up paper we introduce and analyze a new TGFSR variant having better k-distribution property. An efficient algorithm to obtain the order of equidistribution is provided, together with a tight upper bound on the order. A method to search for generators attaining this bound is discussed, and some such generators are listed. The upper bound turns out to be (sometimes far) less than the maximum order of equidistribution for a generator of that period length, but far more than that for a GFSR with a working area of the same size.

1 Introduction

In the previous paper[9], we introduced a random-number-generating algorithm, the *twisted GFSR generator* (TGFSR).

Definition 1. A sequence $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots$ of w-bit integers is *a TGFSR sequence* with parameters (w, n, m, A) $(n > m :$ positive integers) if it satisfies

$$\mathbf{x}_{l+n} = \mathbf{x}_{l+m} \oplus \mathbf{x}_l A \quad (l = 0, 1, 2, \ldots), \tag{1}$$

where $\mathbf{x}_i$ are regarded as row vectors of bits, $\oplus$ denotes the bitwise exclusive-or operation, A is a $w \times w$ matrix with components in GF(2), and $\mathbf{x}_i A$ denotes the multiplication between a row vector and a matrix over GF(2).

If A is an identity matrix, then the sequence is a GFSR sequence based on a characteristic trinomial. As shown in [6, §§3.7, §§3.8], both GFSR and TGFSR can be viewed as implementation approaches of digital matrix generators, and TGFSR generators can also be implemented as large GFSRs (based on a characteristic polynomial of order wn).

Categories and Subject Descriptors: G. 2.1. [Discrete Mathematics] : Combinatorics–*recurrences and difference equations*; G. 3. [Probability and Statistics]–*random number generation.*

General Terms: Algorithms, Theory, Experimentation

Additional Key Words and Phrases: GFSR, TGFSR, m-sequences, k-distribution, finite fields

With a suitable choice of (w, n, m, A), the sequence attains the maximal period $2^{nw} - 1$. Here we treat only TGFSR with maximal periods, and simply call them TGFSR. In the previous paper, we dealt with the case where A is of rational normal form, as below, because it permits an efficient implementation of the recurrence (1).

Definition 2. A TGFSR sequence with

$$A = R := \begin{pmatrix} & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \\ a_0 & a_1 & \cdots & \cdots & a_{w-1} \end{pmatrix}$$

is called *a TGFSR sequence of rational normal form* (TGFSR(R)).

Unfortunately, a TGFSR(R) has a defect from the viewpoint of *k-distribution to v-bit accuracy*[11], defined as follows.

Definition 3. A pseudorandom sequence $\mathbf{x}_i$ of w-bit integers of period P satisfying the following condition is said to be *k-distributed to v-bit accuracy*: let $\mathrm{trunc}_v(\mathbf{x})$ denote the number formed by the leading v bits of $\mathbf{x}$, and consider the kv-bit vectors

$$(\mathrm{trunc}_v(\mathbf{x}_i), \mathrm{trunc}_v(\mathbf{x}_{i+1}), \ldots, \mathrm{trunc}_v(\mathbf{x}_{i+k-1})) \quad (0 \leq i < P).$$

Then, each of the 2^{kv} possible combinations of bits occurs the same number of times in a period, except for the all-zero combination that occurs once less often.

Let $\mathbf{x}_0, \mathbf{x}_1, \ldots$ be a sequence of w-bit integers and let P be its period. For each $v = 1, 2, \ldots, w$, let $k(v)$ denote the maximum number such that the sequence is $k(v)$-distributed to v-bit accuracy. Clearly we have the inequality $2^{k(v)v} - 1 \leq P$, since at most P patterns can occur in one period. In the case of TGFSR, $P = 2^{nw} - 1$ holds, hence we have $k(v) \leq \lfloor nw/v \rfloor$ with n being the number of words. However, as Tezuka[10] pointed out, TGFSR(R) is only n-distributed to 2-bit accuracy, far smaller than the upper bound $k(v) \leq \lfloor nw/v \rfloor$. (Generators attaining this upper bound for every v $(1 \leq v \leq w)$ are called *asymptotically random*[11].) This led us to consider k-distribution of TGFSR with other types of A instead of R. The purpose of this paper is to introduce a new feasible variant of TGFSR with better k-distribution.

It turns out that TGFSR has a tighter upper bound than the one deduced above, namely $k(v) \leq n\lfloor w/v \rfloor$. Consequently, a TGFSR is never asymptotically random. However, we could find an efficient algorithm to obtain A attaining this bound and an efficient implementation of the corresponding TGFSR. We shall list some TGFSR generators attaining these upper bounds $n\lfloor w/v \rfloor$ simultaneously for all v. They are much better than the $\lfloor n/v \rfloor$ achieved by a GFSR of the same size (i.e., n words of w-bit integers). One may still insist that a GFSR of the same *period* $2^{nw} - 1$ (consuming w times memory area of TGFSR) may achieve an asymptotically random distribution $k(v) = \lfloor nw/v \rfloor$, and consequently that Theorem 1 is a negative result. This is not necessarily the case, as shown in the following comparison with an asymptotically random GFSR of N-words $(N \sim nw)$ with $k(v) = \lfloor N/v \rfloor$.
(i) To obtain a TGFSR whose $k(v)$ exceeds $\lfloor N/v \rfloor$ for all v, it is sufficient to take $n = \lceil 2N/(w+1) \rceil$ words $(\ll N)$. (This follows from a simple calculation.) Thus, a TGFSR needs much less memory than a GFSR of the same k-distribution property. Note that in a multi-task system, a memory-consuming program is sometimes time-consuming, because of swapping of memories.

(ii) In the GFSR case, to obtain $k(v)$ for a given initial value, one must calculate the rank of an $N \times N$ matrix for each v, while any initial value attains the upper bound in the case of TGFSR.
(iii) Even an asymptotically random GFSR is rejected by the weight-distribution test, if it is based on a trinomial (see Section 4. See also [9]).

A brief sketch of this paper is as follows. In Section 2, we provide an efficient algorithm to obtain $k(v)$ through simple operations on matrix A (Theorem 1), which also shows that TGFSR has the upper bound $k(v) \leq n\lfloor w/v \rfloor$. In Section 3, we discuss how to search for the matrix A which satisfies the above bounds at once for all v and also allows for an efficient implementation. In Subsection 3.1, we analyze the bad correlation in TGFSR(R) by applying Theorem 1. In Subsection 3.2, based on this analysis, we discuss a method to modify the output sequence of a TGFSR(R) by a simple linear transformation into a sequence of TGFSR satisfying the bound of Theorem 1. This modification requires only a few instructions to be added to the previous TGFSR(R) program. In Subsection 3.3, we discuss an efficient way to determine a modifying parameter. In Section 4, we list some efficient generators attaining these bounds. We conduct empirical tests on these generators and the old TGFSR(R), and we dismiss the latter type.

2 Criterion for equidistribution

The next theorem provides an efficient algorithm to obtain $k(v)$ and its tight upper bound for the general TGFSR.

Theorem 1. Let (w, n, m, A) be the parameters of a TGFSR. Let $\mathbf{d}_j^{(i)}$ denote the i-th column vector of A^j. Consider the sequence of vectors

$$\mathbf{d}_0^{(0)}, \mathbf{d}_0^{(1)}, \ldots, \mathbf{d}_0^{(v-1)}, \mathbf{d}_1^{(0)}, \mathbf{d}_1^{(1)}, \ldots, \mathbf{d}_1^{(v-1)}, \mathbf{d}_2^{(0)}, \ldots.$$

Let $\mathbf{d}_{j_0}^{(i_0)}$ be the first vector that is GF(2)-linearly dependent with the preceding vectors. Then we have

$$k(v) = nj_0.$$

Corollary 1.

$$n|k(v) \text{ and } k(v) \leq n\lfloor w/v \rfloor \quad (v = 1, 2, \ldots, w).$$

Proof (by R. Couture). Fix k and v. Let $V_w := GF(2)^w$ be the space of w-bit integers regarded as a row vector space.

Define

$$\Omega_w := \Omega'_w := V_w^k,$$

and identify Ω_w with the space of $k \times w$ matrices. Similarly define V_v, Ω_v, and Ω'_v.

Let $\rho : V_w^n \to \Omega_w$ map $(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1})$ to the first k values $(\mathbf{x}_0, \ldots, \mathbf{x}_{k-1})$ of the TGFSR sequence $(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_j, \ldots)$ with initial value $(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1})$. Let trunc : $V_w \to V_v$ denote the truncation map defined in Definition 3, that is, the multiplication by the $w \times v$-matrix $Q := \begin{pmatrix} \mathrm{I}_v \\ 0 \end{pmatrix}$ from the right. We denote the multiplication from the right by $\times Q$ and from the left by $Q\times$. Now the k-distribution to v-bit accuracy is equivalent to the surjectivity of the composition map

$$V_w^n \xrightarrow{\rho} \Omega_w \xrightarrow{\times Q} \Omega_v,$$

since the state vector assumes all nonzero values in V_w^n in one period. Let $\tau : V_w^n \to \Omega'_w$ be the map defined by

$$(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}) \mapsto (\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}, \mathbf{x}_0 A, \mathbf{x}_1 A \ldots, \mathbf{x}_{n-1} A, \mathbf{x}_0 A^2, \ldots, \mathbf{x}_0 A^q, \ldots, \mathbf{x}_{r-1} A^q),$$

where r and q are the residue and the quotient of k/n, respectively.

Let us consider the linear recurrence

$$y_{l+n} = y_{l+m} + y_l X \quad (l = 0, 1, 2, \ldots),$$

where X is an indeterminate, $y_0, \ldots, y_{n-1}$ are indeterminates, and y_l $(l \geq n)$ is a polynomial of these indeterminates. Then, for any integer N, y_N can be written as a linear combination of

$$\{y_i X^j | i = 0, 1, \ldots, n-1, i + jn \leq N\},$$

and the coefficient of $y_i X^j$ for unique (i, j) with $N = i + jn$ does not vanish. By substituting $y_l := \mathbf{x}_l$ and $X := A$, we see that there is a regular lower-half triangular $k \times k$-matrix T such that the composition

$$V_w^n \xrightarrow{\tau} \Omega'_w \xrightarrow{T\times} \Omega_w$$

coincides with ρ. Now we have a commutative diagram

$$\begin{tikzcd} V_w^n \arrow[r, "\tau"] \arrow[d, equal] & \Omega'_w \arrow[r, "\times Q"] \arrow[d] & \Omega'_v \arrow[d] \\ V_w^n \arrow[r, "\rho"] & \Omega_w \arrow[r, "\times Q"] & \Omega_v \end{tikzcd},$$

where the two vertical maps are the isomorphisms $T\times$. Hence, it is sufficient to consider the surjectivity of the upper row, which is the linear transformation

$$(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}) \mapsto (\mathbf{x}_0 Q, \mathbf{x}_1 Q, \ldots, \mathbf{x}_{n-1} Q, \mathbf{x}_0 AQ, \mathbf{x}_1 AQ, \ldots, \mathbf{x}_{n-1} AQ, \mathbf{x}_0 A^2 Q, \ldots, \mathbf{x}_{r-1} A^q Q),$$

and this splits to a direct sum according to $V_w^n = V_w \oplus V_w \oplus \cdots \oplus V_w$. Thus the surjectivity of the upper row reduces to the surjectivity of

$$\begin{array}{ccc} V_w^n & \to & V_v^{q+1} \\ \mathbf{x} & \mapsto & \mathbf{x}U, \end{array}$$

where U is a $w \times (q+1)v$-matrix

$$U := (Q, AQ, A^2 Q, \ldots, A^q Q).$$

Thus, k-distribution of the v-bit accuracy is equivalent to the rank of U being $(q+1)v$. In other words, to the independence of columns of U. The maximum q equals j_0 in the statement of the theorem, and since $q = \lfloor k/n \rfloor$, the theorem follows. Corollary 1 is an immediate consequence of the theorem.

Remark 1. The j_0 in the theorem coincides with the order of equidistribution to v-bit accuracy of the matrix linear congruential sequence defined by $z_{i+1} = z_i A$, if $\varphi_A(t)$ is primitive. It is worth noting that the condition for a TGFSR with parameters (w, n, m, A) to achieve the upper bound in Corollary 1 depends only on A, and is independent of n and m, unlike the case of GFSR[11][3]. This implies that one good A serves for any n and m, provided that (w, n, m, A) is a tuple of parameters of a (maximal period) TGFSR generator.

3 How to attain the bound

We want to find matrices A satisfying the upper bound in Corollary 1 and permitting an efficient implementation. We will show that TGFSR(R)s cannot reach that upper bound, and then propose a way of constructing matrices A that satisfy those conditions.

3.1 Bad correlation in TGFSR(R)

We interpret j_0 in Theorem 1 as the degree of the minimal-degree relation between some polynomials, in order to investigate bad correlations in TGFSR(R). Let us fix one set of parameters (w, n, m, A) providing a TGFSR. Let η be an eigenvalue of A. Then η generates GF(2^w) over GF(2) and is of multiplicity one, because the characteristic polynomial of A is irreducible. Thus, the corresponding row eigenvector can be taken in GF$(2^w)^w$. Let $(\phi_1, \dots, \phi_w)$ be such a (row) eigenvector of A, namely,

$$(\phi_1, \dots, \phi_w)A = (\eta\phi_1, \dots, \eta\phi_w), \ \phi_i \in \mathrm{GF}(2^w). \tag{2}$$

We can state Theorem 1 in terms of degree, as follows.

Theorem 2. Let $\eta \in$ GF(2^w) be an eigenvalue of A and $(\phi_1, \dots, \phi_w)$ be a corresponding row eigenvector. Let us define the degree of an element of GF(2^w) as the minimal degree of its representations as a nonzero polynomial in η. (Thus, degree of 0 is w.) Let v be an integer with $1 \le v \le w$. For a linear relation $\sum_{i=1}^{v} \gamma_i\phi_i = 0$ in GF(2^w), we define the degree of the linear relation as the maximum degree of γ_i $(i = 1, \dots, v)$.

Then, $k(v)$ equals n times the degree of the minimal-degree relation between $\{\phi_1, \phi_2, \dots, \phi_v\}$.

Proof. First we claim that $\{\phi_1, \phi_2, \dots, \phi_w\}$ is linearly independent over GF(2). Because, if not, they satisfy a linear relation over GF(2), and hence all Galois conjugates of the set $\{\phi_1, \phi_2, \dots, \phi_w\}$ satisfy the same linear relation. Thus, the Galois conjugates of the vector $(\phi_1, \phi_2, \dots, \phi_w)$ (there are exactly w conjugates) are linearly dependent. This contradicts that each of these vectors lies in an eigenspace with distinct eigenvalues.

By this and Theorem 1, $k(v)$ is n times the minimum j such that the components in the $(j+1)v$-dim GF(2^w)-vector

$$(\phi_1, \phi_2, \dots, \phi_v)(Q, AQ, A^2Q, \dots, A^jQ)$$

(Q: the $w \times v$ matrix defined in the proof of Theorem 1) are linearly dependent over GF(2), in other words, the set

$$\{\phi_1, \phi_2, \dots, \phi_v, \eta\phi_1, \eta\phi_2, \dots, \eta\phi_v\eta^2\phi_1, \dots, \eta^j\phi_1, \dots, \eta^j\phi_v\}$$

being linearly dependent. Thus, j is nothing but the degree of the minimal-degree relation between $\{\phi_1, \dots, \phi_v\}$.

We shall apply this theorem to TGFSR(R). A direct calculation shows the following.

Lemma 1. Let $A := R$ be the matrix in Definition 2. Then, a GF(2^w) vector $(\phi_1, \phi_2, \dots, \phi_w)$ is a (row) eigenvector of R if and only if it satisfies the equations

$$\phi_i = \eta\phi_{i+1} + a_i\phi_w \quad (i = 1, \dots, w-1) \tag{3}$$

and

$$a_0\phi_w = \eta\phi_1. \tag{4}$$

Because $a_0 = 1$, equations (3) and (4) in the above lemma show that

$$\phi_i = \eta(\phi_{i+1} + a_i\phi_1), \quad (i = 1, \ldots, w-1). \tag{5}$$

Thus, in the case of $i = 1$, this shows that the degree of the minimal-degree linear relation between ϕ_1 and ϕ_2 is one, and hence Theorem 2 shows that $k(2) = n \times 1$, recovering a result of Tezuka[10]. Next, we consider consecutive three bits ϕ_i, ϕ_{i+1}, and ϕ_{i+2}. If a_i is zero, then by (3) the minimal-degree linear relation between ϕ_i and ϕ_{i+1} has degree one; hence the i-th bit and $i+1$-th bit are at most n-distributed. If a_{i+1} is zero, the same is true between the $i+1$-th and the $i+2$-th bits. If $a_i = a_{i+1} = 1$, then one can eliminate ϕ_w from equation (3) to obtain

$$\phi_i - \phi_{i+1} = \eta\phi_{i+1} - \eta\phi_{i+2},$$

which asserts the following.

Proposition 1. In the TGFSR sequence, any three consecutive bits of $\mathbf{x}_i$ are at most n-distributed.

From (5) we obtain a recurring formula

$$\phi_{i+1} = \eta^{-1}\phi_i + a_i\phi_1,$$

and by solving this inductively, it is easy to see that

$$\phi_i = \Phi_i(\eta^{-1})\phi_1$$

holds for some polynomial $\Phi_i(t) \in$GF(2)$[t]$ of degree $i-1$ ($i \le w$). Thus, any GF(2)-linear combination of $\phi_1, \ldots, \phi_s$ can be written in the form of $\Psi(\eta^{-1})\phi_1$ with $\Psi(t)$ of degree $\le s-1$. Any two such linear combinations $\Psi(\eta^{-1})\phi_1$ and $\Psi(\eta^{-1})'\phi_1$ have a linear relation of degree $\le s-1$, namely, $(\Psi(\eta^{-1})'\eta^{s-1}) \cdot \Psi(\eta^{-1})\phi_1 = (\Psi(\eta^{-1})\eta^{s-1}) \cdot \Psi(\eta^{-1})'\phi_1$. By Theorem 2 we have proven the following.

Proposition 2. Any two linear combinations of $(\phi_1, \ldots, \phi_s)$ has a linear relation of degree $s-1$. Thus, for a TGFSR(R) sequence, any two linear combinations of the most significant s bits are at most $n(s-1)$-distributed.

3.2 Tempering TGFSR(R)

We shall provide an efficient method to search for a good A based on the above analysis of TGFSR(R). The idea is the following modification (called *tempering*) of TGFSR(R) generators, which is equivalent to considering a general A.

Let $\{\mathbf{x}_i\}$ be a TGFSR(R) sequence, and define a sequence $\{\mathbf{z}_i\}$ by putting

$$\mathbf{z}_i := \mathbf{x}_i P, \tag{6}$$

where P is a regular GF(2)-matrix. By deleting $\mathbf{x}$ from (1) using (6), we realize that $\{\mathbf{z}_i\}$ is a TGFSR sequence with parameters $(w, n, m, P^{-1}RP)$. Since the characteristic polynomial of A

of a (maximal period) TGFSR is irreducible, A is similar to a (unique) rational normal form R, that is, $A = P^{-1}RP$ holds for certain P and R. Hence, any (maximal period) TGFSR can be obtained in this way. Thus, what we should do is to choose a simple P such that $\{z_i\}$ attains the bound in Theorem 1.

Let $(\phi_1, \ldots, \phi_w)$ be the row eigenvector of R, as in §3.1. Then, the eigenvector of $A = P^{-1}RP$ is

$$(\phi'_1, \ldots, \phi'_w) = (\phi_1, \ldots, \phi_w)P,$$

and hence Theorem 2 can be applied.

We shall investigate a feasible transform $\mathbf{x} \mapsto \mathbf{x}P$ permitting an efficient implementation. The linear operations with respect to GF(2) existing in a usual instruction set are exclusive-or, bit shift, and bitwise AND with a constant bitmask. Proposition 1 suggests that bitwise AND will be necessary in some way, since the others do not cut off the adjacency of the consecutive bits. Proposition 2 asserts that a P such as $\begin{pmatrix} U & V \\ 0 & W \end{pmatrix}$, where U is an $s \times 2$ matrix and $s - 1 < \lfloor w/2 \rfloor$, never attains the bound on $k(2)$ since $k(2) \leq n(s-1)$. To attain the bound, P should add the s-th bit of $\mathbf{x}$ with $s - 1 \geq \lfloor w/2 \rfloor$ to one of the most significant two bits. This observation and the limitation in the instruction set lead us to a transformation

$$\mathbf{x} \mapsto \mathbf{x} \oplus (\mathbf{x} << s - 2), \tag{7}$$

with $s - 1 \geq \lfloor w/2 \rfloor$, where $\mathbf{x} << s - 2$ denotes the $(s-2)$-bit shiftleft operation on $\mathbf{x}$.

Taking Proposition 1 into account, we contrive a transformation

$$\mathbf{y} := \mathbf{x} \oplus ((\mathbf{x} << \text{ some bits })\&\text{BITMASK}), \tag{8}$$

where & represents the bitwise AND operation and BITMASK is a suitable bitmask.

A more complicated combinatorial analysis with many case divisions shows that this transform is not yet sufficient. We extensively examined several alternatives, and finally found a feasible transformation, namely, applying the transform (8) twice:

Transform 1. We define P to be the transform $\mathbf{x} \to \mathbf{x}P = \mathbf{z}$ given by

$$\begin{aligned} \mathbf{y} &:= \mathbf{x} \oplus ((\mathbf{x} << s)\&\mathbf{b}); \\ \mathbf{z} &:= \mathbf{y} \oplus ((\mathbf{y} << t)\&\mathbf{c}); \end{aligned} \tag{9}$$

where s and t are integers, $\mathbf{b}$ and $\mathbf{c}$ are suitable bitmasks of word size, $(\mathbf{x} << s)$ indicates the s-bit shiftleft, and & means the bitwise AND operation.

Note that $0 \geq s, t \geq w - 1$. To attain the bound on $k(2)$, it is necessary to satisfy $s + t \geq \lfloor w/2 \rfloor - 1$, since P is easily seen to have the form $\begin{pmatrix} U & V \\ 0 & W \end{pmatrix}$ with U of size $(s + t + 2) \times 2$. Empirically, all TGFSR(R) that we have found can be tempered into TGFSR which attains the bounds by using Transform 1.

3.3 How to find parameters

In this section, a strategy to determine the bitmasks $\mathbf{b}$ and $\mathbf{c}$ is described. Take s and t satisfying the condition stated after Transform 1. Experimentally, we could find $\mathbf{b}$ and $\mathbf{c}$ attaining the bound if we choose t near $\lfloor w/2 \rfloor - 1$ and s near $t/2$ (see Table 1). Now we fix such s and t, and explain a way to find such $\mathbf{b}$ and $\mathbf{c}$.

First set $\mathbf{b} := 0$; $\mathbf{c} := 0$. Assume that the converted sequence $\{z_i\}$ shows the optimal order of equidistribution up to $(v-1)$-bit accuracy in the sense of the Corollary 1. Now we shall optimize the order of equidistribution to v-bit accuracy. The bits of $\mathbf{b}$ and $\mathbf{c}$ possibly affecting the v-th bit of $\mathbf{z}$ are

$$b^{(v)}, c^{(v)}, b^{(v+t)},$$

where $b^{(v)}$ denotes the v-th bit of $\mathbf{b}$, etc. The easiest bit to control is $c^{(v)}$. This bit does not affect more significant bits in z. Bit $b^{(v)}$ influences $z^{(v-t)}$ if $v-t>0$ and $c^{(v-t)}=1$, and hence should not be changed in this case. Bit $b^{(v+t)}$ does not affect more significant bits, but if $c^{(v)}=0$, then it does not affect $z^{(v)}$. In summary, we may change the bitmasks under the following restrictions.

$$\begin{aligned}
c^{(v)} &= \begin{cases} 0 & \text{if } v > w-t \\ (0,1) & \text{otherwise,} \end{cases} \\
b^{(v)} &= \begin{cases} 0 & \text{if } v > w-s \\ \text{as it was} & \text{if } c^{(v-t)}=1 \text{ and } v-t>0 \\ (0,1) & \text{otherwise,} \end{cases} \\
b^{(v+t)} &= \begin{cases} 0 & \text{if } v+t > w-s \text{ or } c^{(v)}=0 \\ (0,1) & \text{otherwise.} \end{cases}
\end{aligned}$$

By checking $k(v)$ in each case, we determine these three bits. We use the backtracking technique to find a parameter satisfying the bounds on $k(v)$, $v=0,1,\ldots,w-1$.

Note that independence of some columns of $P^{-1}R^jP$, as in Theorem 1 (see also Definition 5), is equivalent to that of R^jP. Instead of using columns of R^jP, it is somewhat easier to use rows of ${}^tP{}^tR^j$. The i-th row $\mathbf{p}_i$ of ${}^tP$ is obtained by

$$\begin{aligned}
\mathbf{y} &:= \mathbf{e}_i \oplus ((\mathbf{e}_i \& \mathbf{c}) >> t); \\
\mathbf{p}_i &:= \mathbf{y} \oplus ((\mathbf{y} \& \mathbf{b}) >> s);
\end{aligned}$$

where $\mathbf{e}_i$ is the i-th unit (row) vector. The transform $\mathbf{x} \mapsto \mathbf{x}{}^tR$ is obtained by $\mathbf{x}{}^tR = \text{shift left}(\mathbf{x}) \oplus q$, where q is 0 or $\mathbf{e}_w$ according to the parity of the number of 1s in $\mathbf{x}\&\mathbf{a}$.

4 Practical generators and their tests

Table 1 lists some examples of TGFSR obtained by the method in §3.3. In the table, TT*** is obtained by tempering T*** in the previous paper[9]. Hence tempering can easily be implemented by adding two lines (9) to the original programs for T***. T800 is an original TGFSR(R) generator without tempering, used here for comparison. G607 is an asymptotically random GFSR generator proposed in [11].

Observe that $k(v)$ coincides with $n\lfloor w/v \rfloor$ for the tempered TGFSR, which is the upper bound given in the Corollary 1. Note also that any TGFSR satisfies $k(v) \geq n$, as proven in [9].

Generator		The order of equidistribution							
ID	Parameters	$k(1)$	$k(2)$	$k(3)$	$k(4)$	$k(5)$	$k(6)$	$k(7)$	$k(8)$
		$k(9)$	$k(10)$	$k(11)$	$k(12)$	$k(13)$	$k(14)$	$k(15)$	$k(16)$
		$k(17)$	$k(18)$	$k(19)$	$k(20)$	$k(21)$	$k(22)$	$k(23)$	$k(24)$
		$k(25)$	$k(26)$	$k(27)$	$k(28)$	$k(29)$	$k(30)$	$k(31)$	$k(32)$
TT400	$(w,n,m) = (16,25,11)$	400	200	125	100	75	50	50	50
	a= `A875`	25	25	25	25	25	25	25	25
	s=2, **b**= `6A68`	*	*	*	*	*	*	*	*
	t=7, **c**= `7500`	*	*	*	*	*	*	*	*
TT403	$(w,n,m) = (31,13,2)$	403	195	130	91	78	65	52	39
	a=`6B5ECCF6`	39	39	26	26	26	26	26	13
	s=8, **b**=`102D1200`	13	13	13	13	13	13	13	13
	t=14, **c**=`66E50000`	13	13	13	13	13	13	13	*
TT775	$(w,n,m) = (31,25,8)$	775	375	250	175	150	125	100	75
	a= `6C6CB38C`	75	75	50	50	50	50	50	25
	s=6, **b**=`1ABD5900`	25	25	25	25	25	25	25	25
	t=14, **c**=`776A0000`	25	25	25	25	25	25	25	25
TT800	$(w,n,m) = (32,25,7)$	800	400	250	200	150	125	100	100
	a=`8EBFD028`	75	75	50	50	50	50	50	50
	s=7, **b**=`2B5B2500`	25	25	25	25	25	25	25	25
	t=15, **c**=`DB8B0000`	25	25	25	25	25	25	25	25
T800	$(w,n,m) = (32,25,7)$	800	25	25	25	25	25	25	25
	a=`8EBFD028`	25	25	25	25	25	25	25	25
		25	25	25	25	25	25	25	25
		25	25	25	25	25	25	25	25
G607	$(w,n,m) = (23,607,334)$	607	303	202	151	121	101	86	75
	(an asymptotically random GFSR	67	60	55	50	46	43	40	37
	of 607 words[11])	35	33	31	30	28	27	26	*
		*	*	*	*	*	*	*	*

Table 1*. k-distribution of four tempered TGFSR(R), one plain TGFSR(R), and one GFSR

In Table 1, **a**, **b**, and **c** are written in hexadecimal. Thus, for example, the TT775 31-bit pseudorandom-integer generators can be implemented as follows (C-like notations are used for bit operations).

Define four integer constants $n := 25$, $m := 8$, $s := 6$, and $t := 14$, and three 32-bit integers in the hexadecimal notation $\mathbf{a} :=$ `6C6CB38C`, $\mathbf{b} :=$ `1ABD5900`, and $\mathbf{c} :=$ `776A0000`.

Let $x[n]$ be an array of n 32-bit integers, y be a 32-bit integer variable, and l be an integer variable.

```
Step 1. $l \leftarrow 0$

Step 2. Set $x[0]$, $x[1]$, …, $x[n-1]$ to suitable nonzero initial values (with most significant bit 0).

Step 3. $y \leftarrow x[l] \oplus ((x[l] << s)\&\mathbf{b})$; $y \leftarrow y \oplus ((y << t)\&\mathbf{c})$; output $y$.

Step 4. $x[l] \leftarrow x[(l+m) \bmod n] \oplus \text{shiftright}(x[l]) \oplus \begin{cases} 0 & \text{if LSB of } x[l]=0 \\ \mathbf{a} & \text{if LSB of } x[l]=1, \end{cases}$

Step 5. $l \leftarrow (l+1) \bmod n$

Step 6. Goto Step 3.
```

*If $w = 31$, the most significant bit is always zero in 32-bit words.

Statistical Tests.

To verify the improvement due to tempering, the following statistical tests, *weight distribution tests*[9], are performed. These tests are designed for a statistical treatment of the deviation of weights of trinomial-based m-sequences, which was pointed out by Fredricsson[1].

Using a randomly selected initial seed, we generate $N \times r$ uniformly distributed random numbers $x_1, x_2, ..., x_{N\times r} \in [0,1]$. This sequence is divide into r pieces: $y_1 := (x_1, x_2, \ldots, x_N), y_2 := (x_{N+1}, x_{N+2}, \ldots, x_{2N}), \ldots, y_r := (x_{(r-1)N+1}, x_{(r-1)N+2}, \ldots, x_{rN})$. For each y_i $(1 \le i \le r)$, let X_i be the number of components of y_i greater than R, where R is a fixed constant $0 < R < 1$. The observations X_i $(1 \le i \le r)$ are expected to conform to the binomial distribution: $P(X = k) = \binom{N}{k} R^{N-k}(1-R)^k$. Then we compare the empirical distribution of these r observations to the theoretical distribution in a goodness-of-fit test of hypothesis. To be precise, we divided the interval $[0, N]$ into eight intervals so that the probability of X falling in each interval is roughly equal to each other. Then we count the number of X_i falling in each of eight categories, and get the chi-square statistic[4].

The above procedure is iterated t times with randomly sampled initial seeds, and thus we get t chi-square statistics, $u_1, u_2, ..., u_t$. These $\{u_i\}$ $(1 \le i \le t)$ are expected to conform to the chi-square distribution with 7 degrees of freedom. Then we measure the difference between this empirical distribution and chi-square distribution using Kolmogorov-Smirnov statistics. We get two observations KS^+ and KS^-, which are expected to conform to the KS-distribution. We denote by $KS^\pm(\%)$ the corresponding percentile values, namely, $KS^+(\%)$ (resp. $KS^-(\%)$) is the probability that the random variable K_t^+ (resp. K_t^-) will be KS^+ (resp. KS^-) or less (see [4]).

Table 2 lists the results of the above test for various generators with $R = 1/4,\ N = 256,\ r = 8192,\ t = 64$. Since $R = 1/4$, only the most significant two bits are tested. This table shows that the tempered TGFSR generators and LM * , G607 † and G1563 ‡ are not rejected, while the plain TGFSR(R) generators are rejected. The cpu-time consumed by each generators is also listed. Tempering causes approximately ten percent loss in speed.

As a by-product of this test, we also calculate the third moment M_3 of the r observations $\{X_i\ (1 \le i \le r)\}$, t times. Then we get the mean value of M_3 for these t values: $[M_3] = \sum_{\tau=1}^{t} \{M_3\}_\tau / t$ (about the deviation of M_3 of trinomial GFSRs, see [7]). The theoretical value is $NR(1-R)(2R-1)$, which in this case is -24. The authors do not know the distribution function of M_3, and hence cannot perform a formal test of hypothesis. We can, however, clearly distinguish two groups, one with the mean value within -24 ± 3 and the other with that less than -40.

In addition to weight distribution tests with different parameters ($R = 1/8,\ 1/3,\ 2/3,\ 3/4$ etc.), we performed extensively two other types of tests, the run-test and the KS-test (for details, see [9]) for various generators and these tempered TGFSR sequences always passed. However, trinomial-based GFSR generators such as G607 and G1563 are rejected in the weight distribution test with $R = 1/2,\ N = 4096,\ r = 8192,\ t = 64$, as shown in Table 3.

Table 2. The result of weight distribution test
with $R = 1/4,\ N = 256,\ r = 8192,\ t = 64$

* Lehmer's congruential method, proposed by the authors[9]

† based on a primitive trinomial $X^{607} + X^{273} + 1$, proposed by Tootill[11]

‡ based on the recursion $x_n = x_{n-3p} + x_{n-3q}$, where $p = 521$ and $q = 32$, proposed by Fushimi[2]

Generator	T400	T403	T775	T800	TT400	TT403	TT775	TT800	LM	G607	G1563
KS^+(%)	100	100	100	100	91	54	32	4	84	81	79
KS^-(%)	0	2	0	0	3	26	49	85	22	45	3
$[M_3]$	-44	-46	-46	-44	-27	-25	-23	-24	-22	-23	-23
cpu time ($''$)	74	63	72	72	79	70	79	80	92	55	174

Table 3. The result of weight distribution test
with $R = 1/2,\ N = 4096,\ r = 8192,\ t = 64$

Generator	T400	T403	T775	T800	TT400	TT403	TT775	TT800	LM	G607	G1563
KS^+(%)	80	37	57	39	80	18	18	26	3	100	100
KS^-(%)	10	65	12	76	10	86	79	42	87	0	0
$[M_3]$	41	40	102	-72	41	76	17	71	95	-5974	-2730

Remark 2. The generator G1563 was suggested by one of the referees as a more contemporary serious competitor. We should emphasize that this generator is one of the trinomial-based GFSRs, and any of them has a serious deviation in weights for N more than twice of the order of recurrence, as deduced from the warnings in [7] and [1]. The characteristic polynomial of G1563 has many terms, but it divides a trinomial of degree 1563 (which should be avoided according to [7]), and the generating algorithm of G1563 shows that the criticizing argument in [1] is again valid in this case.

Remark 3. Recently L'Ecuyer[5] tested some class of pseudorandom-number generators and reported that a TGFSR sequence failed in one nearest-pair test. This result was possibly caused by the linear relation between the consecutive three bits (see Proposition 1).

Acknowledgement. The authors would like to express their sincere gratitude to R. Couture, who informed us of the simple proof of Theorem 1 and pointed out its equivalence to Theorem 2. Thanks are also due to the anonymous referees for invaluable comments.

Appendix: a C-program.

Here is a C-program implementing the generator TT800. The function `genrand()` returns a uniformly distributed real pseudorandom number (double precision) between 0 and 1. Note that the initial seed (the initial value of `x`, a 25-dimensional array of 32-bit integers) can be chosen arbitrarily except all-zero, and this makes the code shorter than that of GFSR generators (c.f. the code in the appendix of [2]).

```
/* A C-program for TT800 */
#include <stdio.h>
#define N 25
#define M 7

double
genrand()
{
    unsigned long y;
    static int k = 0;
    static unsigned long x[N]={ /* initial seeds: N=25 words */
        0x95f24dab, 0x0b685215, 0xe76ccae7, 0xaf3ec239, 0x715fad23,
        0x24a590ad, 0x69e4b5ef, 0xbf456141, 0x96bc1b7b, 0xa7bdf825,
        0xc1de75b7, 0x8858a9c9, 0x2da87693, 0xb657f9dd, 0xffdc8a9f,
        0x8121da71, 0x8b823ecb, 0x885d05f5, 0x4e20cd47, 0x5a9ad5d9,
        0x512c0c03, 0xea857ccd, 0x4cc1d30f, 0x8891a8a1, 0xa6b7aadb
    };
    if (k==N) { /* generate N words at one time */
        int kk;
        for (kk=0;kk<N-M;kk++) {
            if (x[kk] % 2 == 0) { x[kk] = x[kk+M] ^ (x[kk] >> 1); }
            else { x[kk] = x[kk+M] ^ (x[kk] >> 1) ^ 0x8ebfd028; } /* a */
        }
        for (; kk<N;kk++) {
            if (x[kk] % 2 == 0) { x[kk] = x[kk+(M-N)] ^ (x[kk] >> 1); }
            else { x[kk] = x[kk+(M-N)] ^ (x[kk] >> 1) ^ 0x8ebfd028; } /* a */
        }
        k=0;
    }
    y = x[k++];
    y ^= (y << 7) & 0x2b5b2500;  /* s and b */
    y ^= (y << 15) & 0xdb8b0000; /* t and c */
    return( (double) y / (unsigned long) 0xffffffff);
}
```

References

[1] Fredricsson, S. A. Pseudo-randomness Properties of Binary Shift Register Sequences. *IEEE Trans. Inform. Theory* IT-21 (1975), 115–120.

[2] Fushimi, M. Random number generation with the recursion $X_t = X_{t-3p} \oplus X_{t-3q}$. *J. Comp. and Appl. Math.* 31 (1990), 105–118.

[3] Fushimi, M. and Tezuka, S. The k-distribution of generalized feedback shift register pseudorandom numbers. *Commun. ACM* 26(1983), 516–523.

[4] Knuth, D. E. *The art of Computer Programming*, Vol 2: *Seminumerical Algorithms*, 2nd ed. Addison-Wesley, Reading, Mass., 1981.

[5] L'Ecuyer, P. Testing random number generators. *Proceedings of the 1992 Winter Simulation Conference*, IEEE Press (1992), 305–313.

[6] L'Ecuyer, P. Uniform random number generation. *Annals of Operations Research*, to appear.

[7] Lindholm, J. H. An analysis of the pseudo-randomness properties of subsequences of long m-sequences. *IEEE Trans. Inform. Theory*, IT-14(July 1968), 569–576.

[8] Marsaglia, G. and Zaman, A. A new class of random number generators. *Annals of Applied Probability* 1(1991), 462–480.

[9] Matsumoto, M. and Kurita, Y. Twisted GFSR generators. *ACM Trans. on Modelling and Computer Simulation* 2(1992), 179–194.

[10] Tezuka, S. A unified view of long-period random number generators. *A manuscript.*

[11] Tootill, J. P. R., Robinson, W. D. and Eagle, D. J. An asymptotically random Tausworthe sequence. *J. ACM* 20, 3(July 1973), 469–481.