Table 7.4 Generating Polynomials

Degree of polynomial	Quaternary generating polynomials	Octal generating polynomials
1	1G	1A
1	1H	1B
2	11G	11C
2	11H	11E
3	111G	101A
3	111H	101B
4	101HG	1001C
4	10GGG	1001E
5	10001G	10011C
5	10001H	10011E

If α is a root of this polynomial, the field elements are defined by (0, 1, G, H), where the nonzero elements can be identified as

$$0 \sim (00), \qquad \alpha^0 = 1 \sim (01), \qquad \alpha^1 = G \sim (10), \qquad \alpha^2 = \alpha + 1 = H \sim (11) \tag{7.3.11}$$

Table 7.4 lists generating polynomials of degree m for quaternary and octal sequences[11–13] These are suitable for use on channels with memory of m symbols, m = 1, 2, 3, 4, 5.

As in the case of binary PN sequences, an additional 0 can be inserted into a nonbinary PN sequence with a period of $M^m - 1$ to produce a deBruijn sequence with a period equal to M^m.

7.4. Generation of Correlated Random Sequences

In many applications there is often the need to generate a random sequence which has a specified autocorrelation function $R(k)$ or $R(\tau)$ or power spectral density $S(f)$. For example, in simulating a randomly time-varying mobile communication channel, the time variations are modeled by a random process with the so-called Jakes power spectral density

$$S(f) = \frac{1}{\sqrt{1 - (f/f_D)^2}}, \quad |f| \le f_D \tag{7.4.1}$$

where f_D is the maximum Doppler frequency.

Another example of this situation occurs when we seek an equivalent representation for phase noise in a communication system where the phase noise might be characterized as a Gaussian process with an arbitrary power spectral density, which in many cases might be specified empirically based on measurements.

When $R(\tau) \neq 0$ for $\tau = kT_s$, $k = \pm 1, \pm 2, \ldots$, the process is correlated at multiples of the sampling interval and hence samples of the random process will be correlated. In order to

simulate the sampled values of such a process, we need algorithms for generating correlated sequences of random numbers with a given autocorrelation function $R(kT_s)$.

There are also many occasions when we have to generate sampled values of multiple random processes which are correlated. For example, when we represent a bandpass Gaussian random process with a power spectral density that is nonsymmetric around the center frequency, the two random processes that represent the real and imaginary components of the complex lowpass-equivalent representation will be correlated. Another example occurs in modeling of multipath propagation in radio channels where the time-varying nature of each path is modeled as a random process. When this model is implemented in a tapped delay line form, the tap gains become a set of correlated random processes, and in order to simulate this model, we need to generate sampled values of a set of correlated processes with a given power spectral density.

Multiple random processes can be represented as a vector-valued random process. Each component of the vector-valued random process is a scalar-valued process with an arbitrary autocorrelation function. The components of the vector-valued random process may be correlated as discussed in the preceding examples. Whereas the correlation along the time axis is temporal in nature and it leads to the notion of power spectral density, the correlation between the components of a vector-valued random process represents a correlation in a different dimension (often referred to as spatial correlation) as shown in Figure 7.14

In the first part of this section we will focus our attention on algorithms for generating scalar-valued random sequences with a given autocorrelation or power spectral density. We will then turn our attention to the generation of vector sequences with a given temporal correlation and spatial correlation. For the Gaussian case we will see that these algorithms will consist in finding and applying a linear transformation either temporally ("horizontally" along the time axis in Figure 7.14) or spatially (applying the transformation "vertically" in Figure 7.14).

The non-Gaussian case will involve nonlinear transformations and it is in general difficult to find these transformations.

7.4.1. Correlated Gaussian Sequences: Scalar Case

An uncorrelated Gaussian sequence can be transformed into a correlated Gaussian sequence through a linear transformation or filtering which preserves the Gaussian distribution, but alters the correlation properties. The coefficients of the linear transformation can be obtained from the specified correlation function of the output. We present below two approaches, a time-domain approach based on the correlation function, and a frequency-domain approach based on the power spectral density. The time-domain approach uses a class of discrete time models called autoregressive and moving average (ARMA) processes.

7.4.1.1. Autoregressive and Moving Average (ARMA) Models

A correlated Gaussian sequence $Y(n)$ can be generated from an uncorrelated Gaussian sequence $X(n)$ using an autoregressive moving average [ARMA (p, q)] model of the form

$$Y(n) = \underbrace{\sum_{i=1}^{p} \phi_{pi} Y(n-i)}_{\text{Autoregressive part}} + \underbrace{\sum_{j=1}^{q} \theta_{qj} X(n-j)}_{\text{Moving average part}} + X(n) \tag{7.4.2}$$

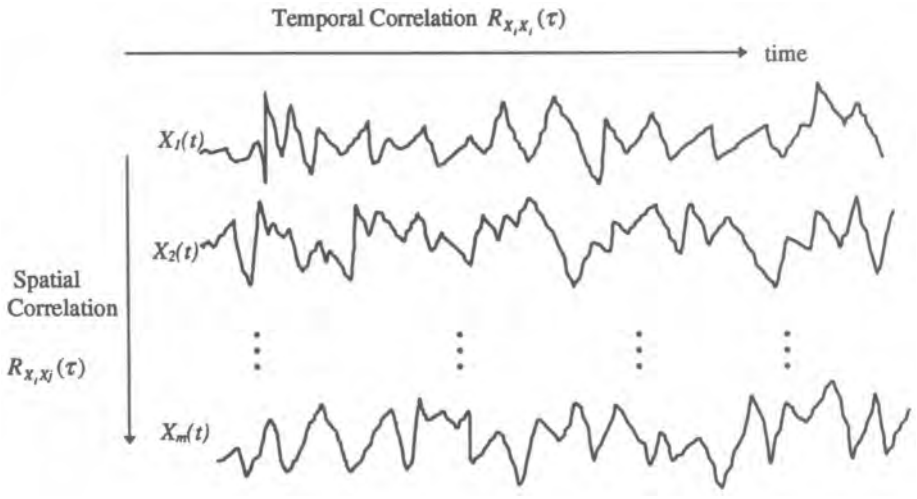

Figure 7.14. Example of a vector-valued random process.

where $X(n)$ is the "input" sequence that is an uncorrelated Gaussian sequence with zero mean and variance σ_X^2, $Y(n)$ is the "output" sequence, and $\phi_{pi}, i = 1, 2, ..., p$, and $\theta_{qj}, j = 1, 2, \ldots, q$, are the parameters of the autoregressive and the moving average parts of the model, respectively. The sampling time is normalized to 1 for convenience. The model given in Equation (7.4.2) is a linear time-invariant discrete-time filter with a transfer function

$$H(f) = \frac{1 + \sum_{i=1}^{p} \phi_{pi} \exp(-j2\pi fi)}{1 - \sum_{i=1}^{q} \theta_{qi} \exp(-j2\pi fi)} \tag{7.4.3}$$

and produces an output power spectral density of the form

$$S_{YY}(f) = S_{XX}(f)|H(f)|^2 = \sigma_X^2 \left| \frac{1 + \sum_{i=1}^{p} \phi_{pi} \exp(-j2\pi fi)}{1 - \sum_{i=1}^{q} \theta_{qi} \exp(-j2\pi fi)} \right|^2 \quad \text{for} \quad |f| < \frac{1}{2} \tag{7.4.4}$$

A simpler version of the model, called the autoregressive AR(p) model, results if the coefficients of the moving average part of the model are zero:

$$Y(n) = \sum_{i=1}^{p} \phi_{pi} Y(n - i) + X(n) \tag{7.4.5}$$

For the AR(p) model, it can be shown that the autocorrelation function $Y(n)$ is given by[14]

$$R_{YY}(0) = \sum_{i=1}^{p} \phi_{pi} R_{YY}(i) + \sigma_X^2 \tag{7.4.6}$$

and

$$R_{YY}(k) = \sum_{i=1}^{p} \phi_{pi} R_{XX}(k-i), \qquad k \geq 1 \tag{7.4.7}$$

We can express Equation (7.4.7) for $k = 1, 2, \ldots, p$ in a matrix form,

$$\begin{bmatrix} R_{XX}(1) \\ R_{XX}(2) \\ \vdots \\ R_{XX}(p) \end{bmatrix} = \begin{bmatrix} R_{XX}(0) & R_{XX}(1) & \cdots & R_{XX}(p-1) \\ R_{XX}(1) & R_{XX}(0) & \cdots & R_{XX}(p-2) \\ \vdots & \vdots & \vdots & \vdots \\ R_{XX}(p-1) & R_{XX}(p-2) & \cdots & R_{XX}(0) \end{bmatrix} \begin{bmatrix} \phi_{p1} \\ \phi_{p2} \\ \\ \phi_{pp} \end{bmatrix} \tag{7.4.8}$$

Equation (7.4.8) is called the Yule–Walker equation and it relates the values of the autocorrelation function of the output and the parameters of the model. Given $R_{XX}(k)$, $k = 0, 1, 2, \ldots, p$, we can obtain the coefficients ϕ_{pi}, $i = 1, 2, \ldots, p$, by inverting Equation (7.4.8) and the value of σ_X^2 from Equation (7.4.7). Recursive procedures for determining the model order p and for solving the Yule–Walker equation can be found in Ref. 15. Once the model parameters have been obtained, they are substituted in Equation (7.4.5) to transform an uncorrelated Gaussian sequence into a correlated Gaussian sequence.

While it is easy to obtain the parameters of an AR(p) model, the relationship between the coefficients of the ARMA(p, q) model and the autocorrelation function of the output are more complex and nonlinear in nature and hence it is very difficult to solve for the coefficients of the model, given the autocorrelation values. While a large body of literature exists on this topic, the most commonly used procedure involves converting an ARMA(p, q) model into an equivalent AR model of order $\gg p + q$ and solving for the parameters of the AR model using the Yule–Walker equation. Another approach solves for the AR part of the ARMA model first and then obtains the coefficients of the MA part. The ARMA model is used in situations where the power spectral density for the output sequence has a complex structure with many peaks and valleys. From Equation (7.4.3) it can be seen that the AR parts influence the poles of the transfer function (and hence the peaks in the output power spectral density) and the MA part corresponds to the zeros of the transfer function (and hence affects the valleys in the psd).

When we implement an ARMA model using Equation (7.4.2) we need p initial values of $Y(k)$ to get the recursion started. If these initial conditions are chosen arbitrarily, then the output will have a transient in the beginning and the first few samples (of the order of $10p$) should be ignored.

An example of the use of ARMA model as an approximation of a phase noise process is presented in Chapter 12.

7.4.1.2. Spectral Factorization Method

We can also design a filter in the frequency domain to transform an uncorrelated Gaussian sequence into a correlated sequence. This approach is particularly useful if the power spectral density of the output process is given in closed form as a ratio of polynomials in f^2. When we pass an independent Gaussian sequence through a filter, the output Gaussian process will have a power spectral density given by

$$S_{YY}(f) = S_{XX}(f)|H(f)|^2 \tag{7.4.9}$$

where

$$S_{XX}(f) = \sigma_X^2 \qquad \text{for} \quad |f| < 1/2 \tag{7.4.10}$$

and hence if we choose $\sigma_X^2 = 1$, we have

$$S_{YY}(f) = |H(f)|^2 \tag{7.4.11}$$

Equation (7.4.9) shows that by choosing the filter transfer function carefully, we can produce the desired output psd (autocorrelation function). The transfer function of a realizable and stable filter can be synthesized by applying the transformation $s = 2\pi jf$ to Equation (7.4.10), *factoring* the given $S_{YY}(f)$ and hence $|H(f)|^2$ into a product of the form $H(s)H(-s)$. The transfer function of the filter is chosen as $H(s)$, which has all of its poles in the left half of the s plane. The example given below illustrates this.

■ *Example 7.4.1.* Find the transfer function of a filter to generate zero-mean Gaussian noise with the PSD

$$S_{YY}(f) = \frac{(2\pi f)^2}{a^2 + (2\pi f)^2}$$

Assume that the input is a zero-mean Gaussian process with $S_{XX}(f) = 1$.

Solution:

$$|H(f)|^2 = \frac{(2\pi f)^2}{a^2 + (2\pi f)^2}$$

Substituting $s = j2\pi f$, we obtain

$$|H(s)|^2 = \frac{-s^2}{a^2 - s^2} = \left(\frac{s}{a+s}\right)\left(\frac{-s}{a-s}\right)$$

The first factor, $s/(a + s)$, is the transfer function of the realizable filter. That is,

$$H(s) = \frac{s}{a+s}, \qquad H(f) = \frac{2\pi jf}{a + 2\pi jf}$$

which yields

$$|H(f)|^2 = \frac{(2\pi f)^2}{a^2 + (2\pi f)^2}$$ ■

Once the transfer function of the filter is obtained via spectral factorization, it can be implemented as an IIR or FIR filter.

In many cases the output spectral density may be given in empirical form or it may be in an analytical form that is not conducive to factoring (for example, the Jakes spectrum). In these cases, one of two approaches can be taken.

1. Empirically fit an analytical form (as a ratio of polynomials in f^2) to the given psd

and then apply the spectral factorization.

2. Directly synthesize an FIR filter by setting the filter transfer function to be equal to the square root of the given power spectral density (the phase of the filter is not important since it does not affect the power spectral density).

The second method can be used to produce a Gaussian sequence with the Jakes spectrum given in Equation (7.4.1) by setting

$$H(f) = \sqrt{S_{YY}(f)} = [1 - (f/f_d)^2]^{-1/4} \tag{7.4.12a}$$

This filter has an impulse response

$$h(t) = 1.457 f_d^{-1/4} J_{1/4}(x), \qquad x = 2\pi f_d t \tag{7.4.12b}$$

where $J_{1/4}(x)$ is the fractional Bessel function. The filter given in Equation (7.4.12) can be implemented as an IIR or FIR filter.

7.4.2. Correlated Gaussian Vector Sequences

7.4.2.1. Special Case

We now consider the problem of generating sampled values of m zero-mean Gaussian random processes $Y_1(t), Y_2(t), \ldots, Y_m(t)$ with arbitrary power spectral densities and arbitrary correlation between them. We begin with the simpler case where the autocorrelation function of each of these processes is the same except for a scale factor and the cross-correlation function has the same functional form as the autocorrelation function. (This assumption is justified for the modeling multipath channels discussed in Chapter 9). The sampled values of these m scalar-valued processes can be treated as components of a vector-valued process $\vec{\mathbf{Y}}(k)$,

$$\vec{\mathbf{Y}}(k) = \begin{bmatrix} Y_1(k) \\ Y_2(k) \\ \vdots \\ Y_m(k) \end{bmatrix} \tag{7.4.13}$$

with

$$R_{Y_i Y_j}(n) = E\{Y(k)Y(k+n)\} = \sigma_{ij} R(n) \tag{7.4.14}$$

where σ_{ij} is the covariance *between the components* of the process at a given time, and $R(n)$ describes the *temporal correlation.* In this model the spatial and temporal correlations are separable.

We can generate a sequence of Gaussian vectors $\vec{\mathbf{Y}}(k)$ with a given set of covariances *between its components* by transforming a sequence of Gaussian vectors $\vec{\mathbf{X}}(k)$ with uncorrelated (and hence independent) Gaussian components where each component has the temporal correlation specified by $R(n)$. We first generate m independent sequences (components of the vector $\vec{\mathbf{X}}$) with a given temporal correlation $R(n)$ using say the ARMA method, and then transform the uncorrelated components of the vector $\vec{\mathbf{X}}$ into correlated components using a memoryless linear transformation. This problem is an easy one and is restated as

follows (we can drop the time index k here since the memoryless transformation is applied to the components of $\vec{\mathbf{X}}$ for each value of k).

Given a random vector $\vec{\mathbf{X}}$ with a multivariate Gaussian distribution with mean zero and covariance matrix

$$\mathbf{\Sigma}_{\vec{\mathbf{X}}} = \vec{\mathbf{I}} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix} \tag{7.4.15}$$

we want to transform it into another Gaussian vector $\vec{\mathbf{Y}}$ with zero mean and covariance matrix

$$\mathbf{\Sigma}_{\vec{\mathbf{Y}}} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1m} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{m1} & \sigma_{m2} & \cdots & \sigma_{mm} \end{bmatrix} \tag{7.4.16}$$

by applying a linear transformation of the form

$$\vec{\mathbf{Y}} = \mathbf{A}\vec{\mathbf{X}} \tag{7.4.17}$$

The linear transformation given above transforms the covariance matrix of $\vec{\mathbf{X}}$, which is the identity matrix given in Equation (7.4.15), into

$$\mathbf{\Sigma}_{\vec{\mathbf{Y}}} = \mathbf{A}\mathbf{\Sigma}_{\vec{\mathbf{X}}}\mathbf{A}^T = \mathbf{A}\mathbf{A}^T \tag{7.4.18}$$

For a given covariance of matrix $\vec{\boldsymbol{Y}}$, Equation (7.4.18) can be used to find the transformation matrix $\mathbf{A}$ by *factoring* the covariance matrix. If $\mathbf{\Sigma}_{\vec{\boldsymbol{Y}}}$ is positive-definite, then there exists a unique lower diagonal matrix $\mathbf{A}$ of the form

$$\mathbf{A} = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} \end{bmatrix} \tag{7.4.19}$$

such that

$$\mathbf{\Sigma}_{\vec{\mathbf{Y}}} = \mathbf{A}\mathbf{A}^T \tag{7.4.20}$$

This decomposition is called the Cholesky decomposition, and the elements of $\mathbf{A}$ can be obtained from the elements of $\mathbf{\Sigma}_{\vec{\mathbf{Y}}}$ according to

$$a_{ij} = \frac{\sigma_{ij} - \sum_{k=1}^{j-1} a_{ik}a_{jk}}{\left[\sigma_{ij} - \sum_{k=1}^{i-1} a_{jk}^2\right]^{1/2}}, \qquad \text{where} \quad \sum_{k=1}^{0} a_{ik}a_{jk} = 0, \quad 1 \le j \le i \le m \tag{7.4.21}$$

It is also possible to use an eigenvector decomposition to find the linear transformation. In any case, after the transformation is found, the algorithm for generating a sequence of correlated Gaussian vectors is shown in Figure 7.15

7.4.2.2. General Case

In the general case, the spatial and temporal correlation could be arbitrary and each component of $\vec{\mathbf{Y}}$ could have a different temporal autocorrelation function and the cross-

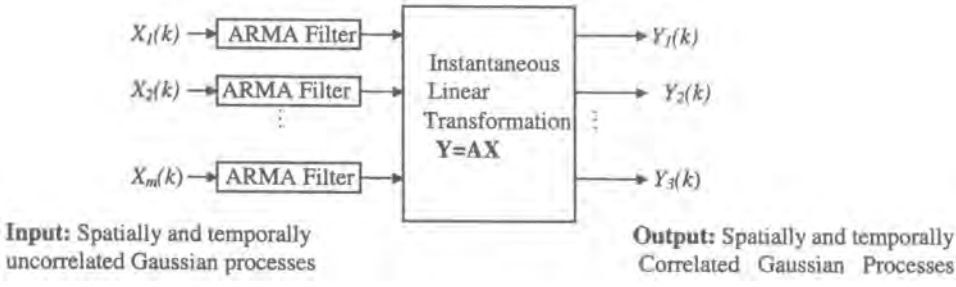

Figure 7.15. Algorithm for generating correlated Gaussian vectors.

correlation functions can also have different functional forms. The process $\vec{\mathbf{Y}}$ is now specified by a sequence of covariance matrices $\boldsymbol{\Sigma}_{\vec{\mathbf{Y}}}(j)$ for each time lag j and it can be generated by transforming $\vec{\mathbf{X}}$ using a vector-valued ARMA process similar to the scalar version discussed in Section 7.4.1.1,

$$\vec{\mathbf{Y}}(n) = \sum_{i=1}^{p} \vec{\boldsymbol{\phi}}_{pi}\vec{\mathbf{Y}}(n-i) + \sum_{j=1}^{q} \vec{\boldsymbol{\theta}}_{qj}\vec{\mathbf{X}}(n-j) + \vec{\mathbf{X}}(n) \tag{7.4.22}$$

Note that the coefficients of the model are themselves matrices. The coefficients of the AR version of the vector model can be determined, using a vector version of the Yule–Walker equations. For detail, the reader is referred to Reference 15.

7.4.3. Correlated Non-Gaussian Sequences

Generation of random sequences with an arbitrary probability density function and autocorrelation is more difficult than generating correlated Gaussian sequences. In the Gaussian case a linear transformation when applied to an uncorrelated Gaussian sequence *preserves* the Gaussian pdf while producing the desired correlation. This is not the case in the non-Gaussian case. For example, if $Y(n) = X(n-1) + X(n)$, where $X(n)$ is an independent sequence of uniform random variables, the sequence $Y(n)$ will be correlated, but its pdf will be triangular. Thus a linear transformation alters both the correlation function and the pdf (except in the Gaussian case, where the form of the pdf is preserved by a linear transformation).

It is possible, however, to control both the pdf and the correlation function using a combination of linear and nonlinear transformation as shown in Figure 7.16.

The nonlinearities $f(\cdot)$ and $g(\cdot)$ and the filter response $h(\cdot)$ are chosen to produce the desired output pdf and PSD. The procedure used to f, g, and h is rather complex and is detailed in Ref. 16 Briefly, $f(\cdot)$ maps the uniform uncorrelated sequence X to an uncorrelated Gaussian sequence V. The filter h maps the uncorrelated Gaussian sequence V to a correlated Gaussian sequence W, which is then mapped to the desired pdf by the nonlinear transformation $g(\cdot)$, which also modifies the predistorted PSD to the desired PSD. These transformations cannot be chosen independently since they affect both the pdf and the PSD. They have to be chosen together subject to the requirements of the output pdf and PSD.

The multidimensional version of this problem is much more difficult to solve and no general solutions are available.

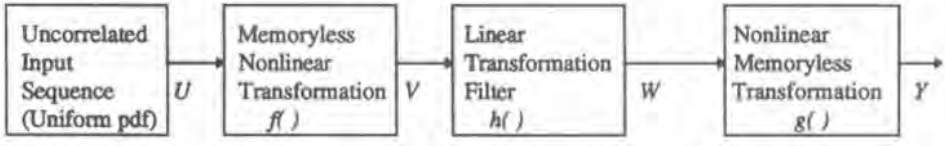

Figure 7.16. Generation of a random sequence with an arbitrary pdf and autocorrelation (PSD).

7.5. Testing of Random Number Generators

Outputs of random number generators (RNGs) used to drive long Monte Carlo simulations must be tested to ensure that they have the "right" statistical properties (temporal and distributional). While it is not necessary to perform exhaustive tests on all RNGs that are part of a simulation package, it is necessary to test at least the uniform and Gaussian RNGs since other RNGs are usually derived from these.

There are two general sets of tests that are normally used to verify the statistical properties of the outputs of RNGs. The first set checks the temporal properties of the outputs of RNGs, the most important temporal properties being stationarity and independence. The second set verifies the distributional properties. This set of tests includes simple checks on the values of parameters such as means and variances of the output sequences to, goodness-of-fit tests, which provide indications of how closely the pdfs of the actual outputs of the RNGs fit the distributions they are supposed to produce.

Tests might range from producing various plots derived from the output sequences and inspecting them visually to see if they "look right", to more sophisticated (statistical) hypothesis tests at a desired level of confidence. We describe below a few typical tests that are useful and easy to implement. Additional tests may be found in Refs. 1, 3, 13, and 16–20.

7.5.1. Stationarity and Uncorrelatedness

7.5.1.1. Introduction

The output of RNGs should be stationary and uncorrelated (except when we want to generate correlated sequences; even in this case, the input sequence used to generate a correlated sequence is usually white, i.e., uncorrelated). Simple tests for stationarity involve the following steps:

1. Generate a long sequence of N samples.
2. Divide it into m nonoverlapping segments and compute the means and variances of each segment.
3. Test for the equality of the means and variances (and other parameters).

For a stationary process, the mean, variances, and other parameters computed from different segments of the data should be equal, within the limits of statistical variations. Hypothesis tests for equality of means, variances, etc., can be found in Ref. 14.

One of the more serious problems with RNGs is that the output sequence might be correlated and in the worst case be periodic with a period shorter than the simulation length. Estimated values of the normalized autocovariance function can be used to test the uncor-