

INPUT MODELING TOOLS FOR COMPLEX PROBLEMS

Barry L. Nelson
Michael Yamnitsky

Department of Industrial Engineering & Management Sciences
Northwestern University
Evanston, Illinois 60208-3119, U.S.A.

ABSTRACT

A simulation model is composed of inputs and logic; the inputs represent the uncertainty or randomness in the system, while the logic determines how the system reacts to the uncertain elements. Simple input models, consisting of independent and identically distributed sequences of random variates from standard probability distributions, are included in every commercial simulation language. Software to fit these distributions to data is also available. In this tutorial we describe input models that are useful when the input modeling problem is more complex.

1 INTRODUCTION

Input models are used to represent the uncertainty or randomness in a simulation. *Input modeling*—choosing the representation—is often characterized as “picking a probability distribution.” And it may be that simple if the following approximations are reasonable:

- The input process can be represented as a sequence of independent random variables having a common (identical) distribution; in other words, a sequence of i.i.d. random variables.
- The common distribution is one of the standard families that are included in nearly all commercial simulation languages: beta, empirical, Erlang, exponential, gamma, lognormal, normal, Poisson, triangular, uniform or Weibull.
- Data are available from which to select and fit the distribution using methods such as maximum likelihood or moment matching.
- A standard distribution provides a good fit to the data, as verified by a visual inspection or a goodness-of-fit test.

There are a number of software packages to support simple input modeling, including ExpertFit, the Arena Input Processor, Stat:Fit and BestFit. Unfortunately, simple models often fail for one of the following reasons:

- The limited shapes represented by the standard families of distributions are not flexible enough to represent some characteristics of the observed data or some known aspects of the process.
- The input process is not inherently independent, either in time sequence or with respect to other input processes in the simulation.
- The input process changes over time.
- No data are available from which to select a family or assess the fit.

This tutorial describes models and techniques that are useful when simple models fail. We emphasize recent advances for which there exists some software support, even if the software is research software rather than commercial software. The related issue of random-variate generation is also discussed.

The paper is organized according to univariate input models (Section 3), arrival-counting processes (Section 4) and multivariate input models (Section 5). Section 2 defines notation that is used throughout the paper, and Section 6 gives directions for obtaining the software.

2 NOTATION

The generic univariate input random variable is denoted by X , with cdf F_X , and density function f_X or mass function p_X . The mean of a random variable is denoted by μ , variance by σ^2 , and correlations between random variables by ρ . Subscripts are added as needed.

A sequence of i.i.d. inputs is $X_i, i = 1, 2, \dots$, while a time-series input process is $\{X_t; t = 0, 1, 2, \dots\}$. The

term “time series” means that the random variables may be dependent in sequence. If we have a sample X_1, X_2, \dots, X_n , then the order statistics (sorted values) are denoted $X_{(1)} < X_{(2)} < \dots < X_{(n)}$.

When the inputs are random vectors, then boldface type is used; e.g.,

$$\mathbf{X}_i = \begin{pmatrix} X_{i1} \\ X_{i2} \\ \vdots \\ X_{ik} \end{pmatrix}$$

is a $k \times 1$ random vector with joint cdf $F_{\mathbf{X}}$.

Greek letters, such as β, θ and α , denote parameters of input models. We use $\hat{\cdot}$ to indicate an estimate, or $\bar{\cdot}$ if the estimate is an average; e.g., $\hat{\beta}$ and \bar{X} .

3 UNIVARIATE INPUT MODELS

In this section we consider alternative input models for univariate distributions. These models are useful for representing i.i.d. sequences $X_i, i = 1, 2, \dots$, and are most often needed when the process data has unusual characteristics (e.g., more than one mode), or when we have no data and want to construct a distribution that has certain properties (e.g., moments or percentiles). The input models presented here range from a flexible family (Section 3.1), to a method for modifying any standard family (Section 3.2), to a method for constructing a distribution with nearly any desired properties (Section 3.3).

3.1 Johnson Family

In the case of modeling data with an unknown distribution, an alternative to using a standard family of distributions is to use a more flexible system of distributions, such as the Johnson translation system (Johnson 1949). One method for fitting target distributions from Johnson’s translation system is via least-squares estimation, which is implemented in a software program called FITTR1 developed by Swain, Venkatraman and Wilson (1988). See Schmeiser and Deutsch (1977) for another flexible family that is easy to use in simulation.

The Johnson translation system is defined by the cdf

$$F(x) = \Phi \{ \gamma + \delta g[(x - \xi)/\lambda] \}, \quad -\infty < x < \infty \quad (1)$$

where Φ is the standard normal cdf, γ and δ are shape parameters, ξ is the location parameter, λ is the scale parameter, and g is one of the following transformations:

$$g(x) = \begin{cases} \log(x) & \text{for the lognormal family} \\ \sinh^{-1}(x) & \text{for the unbounded family} \\ \log[x/(1-x)] & \text{for the bounded family} \\ x & \text{for the normal family.} \end{cases}$$

The appropriate transformation is chosen by estimating the skewness and kurtosis from a random sample X_1, X_2, \dots, X_n and finding the unique Johnson cdf that matches the pair.

The basis for the least-squares fitting procedure implemented in FITTR1 is to minimize a quadratic form of the differences between each parametrically approximated uniformized order statistic, R_j , and its corresponding expected value, η_j , where $R_j = F(X_{(j)})$ and $\eta_j = j/(n+1)$. The $E[R_j] = j/(n+1)$ if F is the true distribution of X . The transformed variate R_j can be written as $R_j = \eta_j + \varepsilon_j$, where the $\{\varepsilon_j\}$ are translated uniform order statistics with mean zero. Then for $\varepsilon' = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n)$ and some $n \times n$ weight matrix \mathbf{W} , the least squares estimation problem is

$$\min_{\gamma, \delta, \xi, \lambda} \varepsilon' \mathbf{W} \varepsilon$$

subject to

$$\begin{aligned} & \delta > 0 \\ \lambda & \begin{cases} > 0 & \text{unbounded family} \\ > X_{(n)} - \xi & \text{bounded family} \\ = 1 & \text{lognormal, normal families} \end{cases} \\ \xi & \begin{cases} < X_{(1)} & \text{lognormal, bounded families} \\ = 0 & \text{normal family} \end{cases} \end{aligned}$$

Swain, Venkatraman and Wilson (1988) investigate various choices for the weight matrix, including the identity matrix (which leads to ordinary least squares), a diagonal matrix with i th diagonal element $1/\text{Var}[\varepsilon_i]$ (which leads to diagonally weighted least squares), and a matrix that is the inverse of the matrix with i, j th element $\text{Cov}[\varepsilon_i, \varepsilon_j]$ (which leads to weighted least squares). They conclude that diagonally weighted least squares is a good overall choice.

The FITTR1 software is available in Fortran or as an MS-DOS executable. In addition to the various least-squares fits, FITTR1 will estimate parameters via moment matching, percentile matching, or minimum L_1 or L_∞ norm. FITTR1 does not produce graphics, but the user can export data to generate plots in standard graphics programs.

Random-variate generation can be accomplished by transforming a standard-normal variate Z (generated in any way) into $X = \xi + \lambda g^{-1}[(Z - \gamma)/\delta]$, where

$$g^{-1}(a) = \begin{cases} e^a & \text{for the lognormal family} \\ (e^a - e^{-a})/2 & \text{for the unbounded family} \\ 1/(1 + e^{-a}) & \text{for the bounded family} \\ a & \text{for the normal family.} \end{cases}$$

For situations when no data are available, DeBroya et al. (1989) describe VISIFIT software for matching a Johnson bounded distribution to subjective information.

The Johnson bounded distribution provides a more flexible alternative to the uniform, triangular and beta distributions often used to represent subjective information about a random quantity. The user provides VISIFIT with an upper and lower bound, plus any one of the following:

- mode and width of central 95% of distribution
- mode and standard deviation
- mode and percentile (fractile)
- mode and median (unless mode = median)
- mode and mean (unless mode = mean)
- median and width
- median and mean (unless median = mean)
- median and standard deviation
- two percentiles (one of which may be the median)
- mean and width
- mean and standard deviation
- mean and percentile
- percentile and width
- percentile and standard deviation
- two beta parameters

The implied Johnson bounded distribution is then displayed graphically, and the user may change its shape in a variety of ways until an acceptable model is attained. The VISIFIT software is available as an MS-DOS executable.

3.2 Inverse Distribution with a Polynomial Filter

To generate random variates from an unknown continuous distribution, the inverse cdf, fitted to a sample of data, is often used; i.e., $X = F_X^{-1}(U)$, where $U \sim U(0, 1)$ and F_X is the chosen distribution. The philosophy behind “inverse distribution with a polynomial filter” (IDPF) is to improve the fit by modifying the transformation to $X = F_X^{-1}(q(U))$, where q is a polynomial in U .

Let $X_i, i = 1, 2, \dots, n$ be an i.i.d. sample. The typical first step in input modeling is to decide what general family—such as gamma, exponential or Johnson—provides the shape that best matches the empirical distribution. The IDPF method makes no assumption about the initial input model F_X , called the reference distribution, except that it has a continuous density function. Once a reference distribution is selected, the second step is to establish the set of parameters that best fit the general family to the data. The third step is to determine the quality of the fit. If visual inspection or a goodness-of-fit test show either a local or general problem with the fit, then IDPF can be applied.

The IDPF procedure, as a fourth step, creates a modified F_X^{-1} that improves the fit compared to the reference distribution. IDPF was developed by Avramidis and Wilson (1994) and is an update to a method originally proposed by Hora (1983). The method is to replace U

with an r th order polynomial $q(U)$,

$$q(U) = b_1U + b_2U^2 + \dots + b_rU^r.$$

The $\{b_i; i = 1, 2, \dots, r\}$ are chosen in such a way that $F_X^{-1}(q(U))$ remains a legitimate inverse cdf, which is equivalent to the statement that $q(U)$ is strictly increasing in U with the boundary conditions $q(0) = 0$ and $q(1) = 1$.

Estimation of the b_i for the IDPF procedure is formulated as a least squares estimation problem. The ordinary-least-squares distance is the distance between the empirical inverse cdf—represented by the order statistics of the sample—and the modified inverse reference distribution. Specifically,

$$\hat{\epsilon}_{\text{ols}}^2 = \min_{b_1, \dots, b_r} \sum_{i=1}^n \left\{ X_{(i)} - F_X^{-1} \left[q \left(\frac{i - 0.5}{n} \right) \right] \right\}^2.$$

Avramidis and Wilson (1994) also develop a weighted-least-squares formulation that compensates for the differences in variability of the order statistics.

The IDPF software is available in FORTRAN or MS-DOS executable. The software assumes the reference distribution is from the Johnson family, and uses the techniques described in Section 3.1 to obtain the initial fit. The key output consists of the four parameters of the reference distribution from the Johnson family, followed by the set of \hat{b}_i s that define the r th-order polynomial. These two sets of parameters provide all that is necessary to generate variates. Variate generation is accomplished in the obvious way by first generating $U \sim U(0, 1)$, then returning $X = F_X^{-1}(q(U))$.

3.3 Univariate Bézier Distributions

Univariate Bézier distributions provide a flexible alternative to standard distributions (Wagner and Wilson 1996a). The univariate Bézier distribution is a special case of a spline curve and is constructed by fitting a curve to a specified number of points called control points. Let $\mathbf{p}_i = (x_i, z_i)'$ be the i th control point for $i = 0, 1, 2, \dots, n$. The control points are not data points; instead, they act as “anchors” for the Bézier cdf and can be moved so as to alter the shape of the distribution. Typically, the Bézier cdf is a continuous distribution. The Bézier cdf interpolates the first and last control points exactly (and is thus a bounded distribution) but might only pass in close proximity to control points $1, 2, \dots, n - 1$.

A Bézier distribution with $n + 1$ control points is defined as

$$\mathbf{P}(t) = \begin{pmatrix} x(t) \\ F_X(x(t)) \end{pmatrix} = \sum_{i=0}^n B_{n,i}(t) \mathbf{p}_i$$

for $t \in [0, 1]$, where $B_{n,i}(t)$ is the Bernstein polynomial

$$B_{n,i}(t) = \begin{cases} \binom{n}{i} t^i (1-t)^{n-i}, & \text{for } t \in [0, 1] \\ 0, & \text{otherwise.} \end{cases}$$

Although the Bernstein polynomial may initially seem complex, it is helpful to think of it in terms of the probability mass function (pmf) of a binomial random variable. For a binomial random variable Y , the probability of i successes in n trials is

$$\Pr\{Y = i\} = \binom{n}{i} p^i (1-p)^{n-i}$$

for $i = 0, 1, \dots, n$, where p is the probability of success. The parameter p is fixed, and probability statements are made about the random variable Y for different values of i . The Bernstein polynomial differs from the pmf of a binomial random variable in that it fixes i and varies t (the continuous analog of p) within the interval $[0, 1]$. Thus, for t equal to p , the value of the Bernstein polynomial is equal to the value of the pmf of a binomial random variable evaluated at i .

At any value of t in the interval $[0, 1]$, the value of the Bézier distribution is simply a weighted average of the control points. To observe this, notice that the Bézier distribution can be written as

$$\mathbf{P}(t) = \begin{pmatrix} x(t) \\ F_X(x(t)) \end{pmatrix} = \sum_{i=0}^n \binom{n}{i} t^i (1-t)^{n-i} \mathbf{p}_i$$

for $t \in [0, 1]$. For any fixed value of t the sum of the weights is 1.

PRIME is a software tool used to construct univariate Bézier distributions with or without data. It is an interactive, graphical software program that runs on a PC under Windows.

PRIME's main workspace is a window displaying the X and F_X coordinate axes along with the Bézier cdf. The initial starting point for PRIME is a cdf consisting of six control points arranged in a straight line. The shape of the distribution is changed by repositioning the control points within the window. Adjustments to the position of a control point are made by clicking and dragging the point to a new location on the screen. Control points can be both added and deleted. As control points are repositioned, added, or deleted, the shape of the Bézier distribution is updated. Furthermore, PRIME has the ability to detect infeasible cdfs (i.e., ones that are not nondecreasing) and highlights an invalid distribution in red.

PRIME also allows the user to view and manipulate the probability density function. Just like the cdf, the pdf is updated as the control points are moved. Although it is possible to manipulate the pdf, it is not as intuitive

as adjusting the cdf. Because the total area under the pdf must remain equal to 1, adjusting one control point simultaneously adjusts both adjacent control points (unless one of the adjacent control points is the last control point) in order to satisfy this constraint.

Another useful feature of PRIME is that it allows a comparison between the first four moments of the Bézier distribution and an empirical data set. In the absence of data, this comparison is useful for constructing an input model having any first four moments. PRIME also allows the user to insure that the Bézier distribution has certain fixed percentiles.

In addition to allowing the user to construct a cdf interactively, PRIME has the ability to apply standard statistical estimation procedures to Bézier distributions. Specifically, maximum likelihood, moment matching, percentile matching, least squares, L_1 norm and L_∞ norm estimation can be accessed when data has been imported. PRIME will also determine the number of control points to use by sequentially evaluating the impact on fit of an additional control point via a likelihood ratio test (Wagner and Wilson 1996b).

Bézier random-variates are generated via the inverse transform method. Given a random number $U \sim U(0, 1)$, a search procedure is implemented to find the associated value of the parameter t such that $F_X(x(t)) = U$. Once t is determined, $X = x(t)$.

4 ARRIVAL-COUNTING PROCESSES

Arrival-counting processes are among the most important of all input models because simulations of industrial and service systems are typically driven by the arrivals of customers, orders, materials, information, etc. Renewal arrival-counting processes, in which the times between arrivals are i.i.d. random variables, are a standard feature of commercial simulation languages. A marginal distribution that is often chosen is the exponential distribution with mean $1/\lambda$, which implies that the renewal arrival process is a Poisson arrival process with constant arrival rate λ arrivals per unit time.

In many practical situations there is need for an arrival process whose rate is a function of time, $\lambda(t)$. The nonhomogeneous Poisson process (NHPP) is a well-known generalization of the Poisson process that allows a time-dependent arrival rate. However, fitting the function $\lambda(t)$ to data is a very difficult problem. Lee, Wilson, and Crawford (1991) consider fitting an NHPP with an exponential rate function containing a polynomial component (for long term trend) and a trigonometric component (for cyclic behavior). Kuhl, Wilson and Johnson (1997) extended these ideas to allow multiple periodicities.

Let $\{N(t) : t \geq 0\}$ be a nonnegative integer valued stochastic process representing the cumulative number of arrivals up to time t . Consider a sequence of n arrivals for this process at times $t_1 < t_2 < \dots < t_n$ over a fixed time interval $(0, S]$. A reasonable starting point for modeling such a process would be an NHPP with a rate function, $\lambda(t)$, that could capture any cyclic or long term trends in the data. Kuhl, Wilson and Johnson (1997) consider the rate function

$$\lambda(t) = \exp \left\{ \sum_{i=0}^m \alpha_i t^i + \sum_{k=1}^p \beta_k \sin(\omega_k t + \phi_k) \right\}. \quad (2)$$

The vector, Θ , of unknown parameters consists of $m + 1$ coefficients for the polynomial component, and the p amplitudes (β_k), frequencies (ω_k) and phase shifts (ϕ_k) for the trigonometric component. For ease of exposition, we focus on the case $p = 1$ cyclic component.

Using the properties of an NHPP gives the following log-likelihood function of Θ in the special case $p = 1$:

$$L(\Theta) = \sum_{i=0}^m \alpha_i T_i + \beta \sum_{j=1}^n \sin(\omega t_j + \phi) - \int_0^S \lambda(z) dz$$

where $T_i = \sum_{j=1}^n t_j^i$. Notice that the degree m of the polynomial component is also unknown. Since m must be a nonnegative integer, it is difficult to determine via maximum likelihood. Therefore, Lee, Wilson and Crawford (1991) recommend that estimation of Θ be conditioned on a fixed value of m , and the final value of m be determined by a likelihood ratio test. Differentiating the log-likelihood function with respect to each unknown parameter individually results in a system of $m + 4$ nonlinear equations which can be solved numerically. See Johnson, Lee and Wilson (1994a) for details on parameter estimation.

Johnson, Lee, and Wilson (1994b) describe two software programs to estimate parameters for, and simulate, an NHPP with rate function (2) in the special case of $p = 1$ cyclic component. The program NPPMLE computes maximum-likelihood estimates for the rate function parameters given a set of arrival epochs. The program requires additional input from the user regarding the length of the observation interval, number of arrivals, maximum degree of the polynomial, significance level for the likelihood ratio test to select m , and other items concerning the trigonometric component. The output provides estimated values for all parameters.

The program NPPSIM uses rate function parameters (such as estimates from NPPMLE) to simulate arrivals by the method of thinning with a piecewise linear majorizing function. Inputs required by NPPSIM are the rate function parameters and the length of the observation interval. The output of the program is a series of arrival epochs written

to a file, and the piecewise linear majorizing function which is written to the terminal. This rate function is displayed with an upper limit, slope, and intercept for each interval. Both of the software programs are written in FORTRAN 77. For the case $p \geq 1$, Kuhl, Wilson and Johnson (1997) have developed mp3mle and mp3sim, extensions of NPPMLE and NPPSIM that allow multiple periodicities.

5 MULTIVARIATE INPUT MODELS

In this section we consider input models for dependent random variables, either random vectors or time-series processes. Standard models for these two cases are the multivariate normal distribution (denoted MVN) for random vectors, and the Gaussian autoregressive order- p process (denoted AR(p)) for time-series input. The univariate marginal distribution in both cases is normal, limiting the usefulness of these models in simulation applications. Nevertheless, since many alternative input models are based on the MVN or AR(p) models, we review each of them briefly.

The standard multivariate normal distribution defines a $k \times 1$ random vector \mathbf{Z} with mean vector $\boldsymbol{\mu} = (0, 0, \dots, 0)'$ and correlation matrix

$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & \rho_{12} & \cdots & \rho_{1k} \\ \rho_{21} & 1 & \cdots & \rho_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{k1} & \rho_{k2} & \cdots & 1 \end{pmatrix}$$

such that the i th element Z_i is distributed $N(0, 1)$ and $\rho_{ij} = \text{Corr}[Z_i, Z_j]$. The parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ uniquely specify a multivariate normal distribution. For multivariate vectors with other marginal distributions, however, this is not enough information to determine a unique joint distribution.

The standard AR(p) process is a time series $\{Z_t; t = 0, 1, 2, \dots\}$ defined by the recursion

$$Z_t = \sum_{h=1}^p \alpha_h Z_{t-h} + \varepsilon_t \quad (3)$$

where $\{\varepsilon_t\}$ is a sequence of i.i.d. $N(0, \sigma_\varepsilon^2)$ random variables. Let $\rho_Z(h) = \text{Corr}[Z_t, Z_{t+h}]$, the autocorrelation at lag h . If this process is appropriately initialized, the parameters $\alpha_1, \alpha_2, \dots, \alpha_p$ satisfy certain conditions, and $\sigma_\varepsilon^2 = 1 - \sum_{h=1}^p \alpha_h \rho_Z(h)$, then each Z_t is marginally $N(0, 1)$ with autocorrelations that are a known function of $\alpha_1, \alpha_2, \dots, \alpha_p$. Similar to the case of random vectors, alternative time-series input models with other marginal distributions are not uniquely determined by their autocorrelation structure.

5.1 Time-Series Input Processes

The goal is to construct a stationary time series $\{X_t; t = 0, 1, 2, \dots\}$ with given marginal distribution F_X and given first p autocorrelations $\rho_X(h), h = 1, 2, \dots, p$. There are two basic approaches:

1. Construct such a process using properties specific to the particular marginal distribution of interest F_X .
2. Construct a process $\{U_t; t = 0, 1, 2, \dots\}$ with $U(0,1)$ marginals and whose autocorrelations are easily controlled. Then form the input process via the transformation $X_t = F_X^{-1}(U_t)$.

Because of its generality, we only discuss the second approach here. In addition to the method we describe, see Melamed, Hill and Goldsman (1992), Song, Hsiao and Chen (1996) and Willemain and Desautels (1993) for alternatives.

AutoRegressive To Anything (ARTA) processes define a $U(0,1)$ time series via the transformation $U_t = \Phi(Z_t)$, where $\{Z_t; t = 0, 1, 2, \dots\}$ is a stationary, standard $AR(p)$ process and Φ is the standard normal cdf. Cario and Nelson (1996) have shown that the lag- h autocorrelation of the time-series input process defined by

$$X_t = F_X^{-1}(U_t) = F_X^{-1}[\Phi(Z_t)] \tag{4}$$

is a continuous, nondecreasing function of the lag- h autocorrelation of the Z_t process. Therefore, the problem of fixing the first p autocorrelations of the X_t process decomposes into p independent problems of determining the value $\rho_Z(h) = \text{Corr}[Z_t, Z_{t+h}]$ that maps into the desired autocorrelation $\rho_X(h) = \text{Corr}[X_t, X_{t+h}]$, for $h = 1, 2, \dots, p$. Since $\rho_X(h)$ is a continuous, nondecreasing function of $\rho_Z(h)$, these problems are easily solved numerically (Cario and Nelson 1998). Variate generation is accomplished by generating a stationary, standard $AR(p)$ process Z_t by any method, then applying equation (4).

ARTAFACTS (ARTA Fitting Algorithm for Constructing Time Series) is software available in FORTRAN to fit ARTA processes for nine marginal distributions (normal, student's t , continuous uniform, exponential, gamma, Weibull, lognormal, Johnson unbounded, Johnson bounded, empirical cdf and discrete distribution with finite support) and up to 5 autocorrelations. ARTAFACTS will also estimate the autocorrelation function of a data set. As input the user must provide the parameters of the desired marginal distribution and the autocorrelations to be matched. ARTAGEN (ARTA GENERation algorithm) takes the output of ARTAFACTS and generates a stationary ARTA process as input to a simulation.

5.2 Random Vectors

A number of methods have been proposed for representing and generating random vectors with specific distributions. See Devroye (1986) and Johnson (1987) for general references. In addition, multivariate extensions to the Johnson family have been developed (Johnson 1987, Chapter 5, Stanfield, et al. 1996), and a bivariate Bézier distribution has been defined and implemented in PRIME (Wagner and Wilson 1995, 1996b).

A general method for obtaining random vectors with arbitrary marginal distributions and correlation matrix is described by Cario and Nelson (1997). The central idea is to transform a standard multivariate normal vector into the desired random vector. These vectors are referred to as having a NORTA (NORmal To Anything) distribution. Specifically, let

$$\mathbf{X} = \begin{pmatrix} F_{X_1}^{-1}[\Phi(Z_1)] \\ F_{X_2}^{-1}[\Phi(Z_2)] \\ \vdots \\ F_{X_k}^{-1}[\Phi(Z_k)] \end{pmatrix}$$

where $\mathbf{Z} = (Z_1, Z_2, \dots, Z_k)'$ is a standard MVN vector with correlation matrix Σ , and $F_{X_1}, F_{X_2}, \dots, F_{X_k}$ are the desired marginal distributions. The problem then becomes finding a Σ that implies the desired correlation matrix for \mathbf{X} . The results of Cario and Nelson (1997) suggest that this is not a difficult numerical problem. The procedure for finding Σ is an extension of the ARTA method for time-series input.

The NORTA method provides a general-purpose approach for modeling and generating dependent input processes. The generality, however, comes at the cost of computational efficiency. The fitting process is time consuming, although this expense is incurred only once for each input model. And even though the marginal time for generating each NORTA variate can be longer than the fastest available method for a particular distribution (as there is a need to evaluate the composite function $F_X^{-1}[\Phi(Z)]$), the NORTA method will yield acceptable results in system simulation environments where event processing and animation account for large fraction of the execution time.

NORTA has been implemented by Chen and Jeng (1998) in their Random Vector software package. The software, which is written in C, generates a given number of k -dimensional random vectors having a desired correlation structure and given marginals. The inputs to the program include the initial random seed, desired number of vectors to be generated, the dimension of the vectors, marginals for each vector component, and the upper triangular form

of the correlation matrix excluding the diagonal elements. Marginal distributions can be selected by the user from eleven commonly used distributions, including normal, lognormal, exponential, beta, gamma, Weibull, uniform, and others. For each distribution, the user is asked to enter the necessary distribution parameters, such as mean and standard deviation for the normal marginal.

Random Vector software uses the method of retrospective optimization in order to match the desired correlation structure iteratively. Specifically, a stochastic root-finding algorithm is implemented in this software.

The output of the program consists of the intermediate output of the stochastic root-finding procedure, as the sample sizes are being increased, and the final result, consisting of the desired number of random vectors and their statistics. Statistics include mean and variance of each marginal within a vector, as well as the resulting correlation matrix. The code's accuracy and efficiency strongly depend on the dimension of random vectors and their marginal distributions. Computational complexity increases greatly as the vector dimension increases.

For a different approach to fitting NORTA distributions that is useful for obtaining simulation input whose marginal distribution and *sample correlation matrix* closely matches given marginals and correlation matrix, see Lurie and Goldberg (1998).

6 OBTAINING THE SOFTWARE

FITTR1, VISIFIT and IDPF may be obtained from <http://www.ie.ncsu.edu/people/faculty/wilson/>.

PRIME may be obtained from Mary Ann Flanigan Wagner at maflanig@osf1.gmu.edu.

mp3mle and mp3sim may be obtained from <http://www.ie.ncsu.edu/people/faculty/wilson/>.

ARTAFACTS and ARTAGEN may be obtained from <http://www.iems.nwu.edu/~nelsonb/ARTA/>.

Random Vector software may be obtained from Huifen Chen at huifen@aries.dyu.edu.tw.

ACKNOWLEDGMENTS

This paper is an update of Nelson, B. L., P. Ware, M. C. Cario, C. A. Harris, S. A. Jamison, J. O. Miller, J. Steinbugl and J. Yang. 1995. Input modeling when simple models fail. In *Proceedings of the 1995 Winter Simulation Conference*, ed. C. Alexopoulos, K. Kang, W. R. Lilegdon and D. Goldsman, 93–100. Institute of Electrical and Electronics Engineers, Arlington, Virginia.

REFERENCES

- Avramidis, A. N. and J. R. Wilson. 1994. A flexible method for estimating inverse distribution functions in simulation experiments. *ORSA Journal on Computing* 6:342–355.
- Cario, M. C. and B. L. Nelson. 1996. Autoregressive to anything: Time-series input processes for simulation. *Operations Research Letters* 19:51–58.
- Cario, M. C. and B. L. Nelson. 1997. Modeling and generating random vectors with arbitrary marginal distributions and correlation matrix. Technical Report, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois.
- Cario, M. C. and B. L. Nelson. 1998. Numerical methods for fitting and simulating autoregressive-to-anything processes. *INFORMS Journal on Computing* 10:72–81.
- Chen, H. and C. Jeng. 1998. RA algorithms for generation of multivariate random vectors. Working paper, Department of Industrial Engineering, Da-Yeh University, Chang-Hwa, Taiwan.
- DeBroya, D. J., R. S. Dittus, S. D. Roberts and J. R. Wilson. 1989. Visual interactive fitting of bounded Johnson distributions. *Simulation* 52:199–205.
- Devroye, L. 1986. *Non-Uniform Random Variate Generation*. New York: Springer-Verlag.
- Hora, S. C. 1983. Estimation of the inverse function for random variate generation. *Communications of the ACM* 26:590–594.
- Johnson, M. A., S. Lee and J. R. Wilson. 1994a. Experimental evaluation of a procedure for estimating nonhomogeneous Poisson processes having cyclic behavior. *ORSA Journal on Computing* 6:356–368.
- Johnson, M. A., S. Lee and J. R. Wilson. 1994b. NPPMLE and NPPSIM: Software for estimating and simulating nonhomogeneous Poisson processes having cyclic behavior. *Operations Research Letters* 15:273–282.
- Johnson, M. E. 1987. *Multivariate Statistical Simulation*. New York: John Wiley.
- Johnson, N. L. 1949. Systems of frequency curves generated by methods of translation. *Biometrika* 36:297–304.
- Kuhl, M. E., J. R. Wilson and M. A. Johnson. 1997. Estimating and simulating Poisson processes having trends or multiple periodicities. *IIE Transactions* 29:201–211.
- Lurie, P. M. and M. S. Goldberg. 1998. An approximate method for sampling correlated random variables from partially-specified distributions. *Management Science* 44:203–218.
- Lee, S., J. R. Wilson and M. M. Crawford. 1991. Modeling and simulation of a nonhomogeneous Poisson process having cyclic behavior. *Communications in Statistics—Simulation and Computation* 20:777–809.

- Melamed, B., J. Hill and D. Goldsman. 1992. The TES methodology: Modeling empirical stationary time series. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J. J. Swain, D. Goldsman, R. C. Crain and J. R. Wilson, 135–144. Institute of Electrical and Electronics Engineers, Washington, D.C.
- Schmeiser, B. W. and S. J. Deutsch. 1977. A versatile four parameter family of probability distributions suitable for simulation. *IIE Transactions* 9:176–181.
- Song, W. T., L. Hsiao and Y. Chen. 1996. Generation of pseudorandom time series with specified marginal distributions. *European Journal of Operational Research* 93:194–202.
- Stanfield, P. M., J. R. Wilson, G. A. Mirka, N. F. Glasscock, J. P. Psihogios and J. R. Davis. 1996. Multivariate input modeling with Johnson distributions. In *Proceedings of the 1996 Winter Simulation Conference*, , ed. J. M. Charnes, D. J. Morrice, D. T. Brunner and J. J. Swain, 1457–1464. Institute of Electrical and Electronics Engineers, Coronado, California.
- Swain, J. J., S. Venkatraman and J. R. Wilson. 1988. Least-squares estimation of distribution functions in Johnson's translation system. *Journal of Statistical Computation and Simulation* 29:271–297.
- Wagner, M. A. F. and J. R. Wilson. 1995. Graphical interactive simulation input modeling with bivariate Bézier distributions. *ACM TOMACS* 5:163–189
- Wagner, M. A. F. and J. R. Wilson. 1996a. Using univariate Bézier distributions to model simulation input processes. *IIE Transactions* 28:699–711.
- Wagner, M. A. F. and J. R. Wilson. 1996b. Recent developments in input modeling with Bézier distributions. In *Proceedings of the 1996 Winter Simulation Conference*, ed. J. M. Charnes, D. J. Morrice, D. T. Brunner and J. J. Swain, 1448–1456. Institute of Electrical and Electronics Engineers, Coronado, California.
- Willemain, T. R. and P. A. Desautels. 1993. A method to generate autocorrelated uniform random numbers. *Journal of Statistical Computation and Simulation* 45:23–31.
- MICHAEL YAMNITSKY** is a Ph.D. student in the Department of Industrial Engineering and Management Sciences at Northwestern University. His research interests include systems simulation, input modeling and stochastic processes. He has also done project work and has work experience in the area of warehousing, logistics, and supply chain management.

AUTHOR BIOGRAPHIES

BARRY L. NELSON is a Professor in the Department of Industrial Engineering and Management Sciences at Northwestern University, and is Director of the Master of Engineering Management program there. He is interested in the design and analysis of computer simulation experiments, particularly statistical efficiency, multivariate output analysis and input modeling. He was program chair of the 1997 Winter Simulation Conference.