INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

An Algorithm for Fast Generation of Bivariate Poisson Random Vectors

Kaeyoung Shin, Raghu Pasupathy,

To cite this article:
Kaeyoung Shin, Raghu Pasupathy, (2010) An Algorithm for Fast Generation of Bivariate Poisson Random Vectors. INFORMS Journal on Computing 22(1):81-92. http://dx.doi.org/10.1287/ijoc.1090.0332

INFORMS Journal on Computing
Vol. 22, No. 1, Winter 2010, pp. 81–92
ISSN 1091-9856 | EISSN 1526-5528 | 10 | 2201 | 0081

DOI 10.1287/ijoc.1090.0332

An Algorithm for Fast Generation of Bivariate Poisson Random Vectors

Kaeyoung Shin, Raghu Pasupathy
Grado Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061 {pman93@vt.edu, pasupath@vt.edu}

We present the "trivariate reduction extension" (TREx)—an exact algorithm for the fast generation of bivariate Poisson random vectors. Like the normal-to-anything (NORTA) procedure, TREx has two phases: a preprocessing phase when the required algorithm parameters are identified, and a generation phase when the parameters identified during the preprocessing phase are used to generate the desired Poisson vector. We prove that the proposed algorithm covers the entire range of theoretically feasible correlations, and we provide efficient-computation directives and rigorous bounds for truncation error control. We demonstrate through extensive numerical tests that TREx, being a specialized algorithm for Poisson vectors, has a preprocessing phase that is uniformly a hundred to a thousand times faster than a fast implementation of NORTA. The generation phases of TREx and NORTA are comparable in speed, with that of TREx being marginally faster. All code is publicly available.

Key words: statistics; simulation; random variable generation; multivariate distribution; correlation
History: Accepted by Marvin Nakayama, Area Editor for Simulation; received March 2008; revised September 2008, March 2009; accepted March 2009. Published online in *Articles in Advance* August 18, 2009.

1. Introduction

The problem of generating random variables from a Poisson distribution with given parameter $\lambda > 0$ is well studied, and there currently exist very fast general procedures for implementation on a digital computer. See Schmeiser and Kachitvichyanukul (1981) and Devroye (1986) for overviews, and Chen (1994), Kronmal and Peterson (1979), Atkinson (1979a, b), and Kemp and Kemp (1991) for specific algorithms.

In this paper, a preliminary version of which was published as Shin and Pasupathy (2007), we consider the bivariate generalization of the above problem—given $\lambda > 0$, $\lambda' > 0$, and $-1 \le \rho \le 1$, generate a bivariate Poisson random vector (X_1, X_2) with the stipulation that X_1, X_2 are Poisson distributed with means λ, λ', respectively, and $\mathrm{Corr}(X_1, X_2) = \rho$. Our motivation is a setting where there is a need for a Poisson random vector generation algorithm that is exact, exhibits fast setup and generation times, and is able to handle any "fair" problem. Applications seem widespread—see Johnson et al. (2005, Chapter 4) and Johnson et al. (1997, Chapter 37) for a long list of references on Poisson models.

We will use the following measures in assessing the quality of our solution procedure: (i) exactness of the procedure; (ii) the fraction of the feasible set of correlations that can be handled by the procedure; (iii) execution time for the preprocessing phase, if any, within the procedure; and (iv) execution time for the generation phase within the procedure. Whereas the measures (i), (iii), and (iv) are self-explanatory, what we mean by (ii) will become clear in §2, where we elaborate on the notion of the set of *feasible correlations* for a given pair of marginal distributions. For now, it suffices to note that specifying the marginal distributions of X_1 and X_2 automatically imposes a maximum feasible correlation $\rho^+(\lambda, \lambda')$, and a minimum feasible correlation $\rho^-(\lambda, \lambda')$, that is achievable between X_1 and X_2. The interval $[\rho^-(\lambda, \lambda'), \rho^+(\lambda, \lambda')] \subseteq [-1, 1]$ is thus the largest set of correlations that any procedure can hope to handle. Therefore, (ii) is measured as the fraction of the set of correlations $[\rho^-(\lambda, \lambda'), \rho^+(\lambda, \lambda')]$ that can be handled by the given procedure.

1.1. Traditional Solutions

Traditionally, the problem of generating bivariate Poisson random vectors is approached using one of two methods: (i) trivariate reduction (TR) or (ii) the "normal-to-anything" (NORTA) procedure. In what follows, we provide a brief discussion of each of these.

1.1.1. Trivariate Reduction. TR (Mardia 1970) is a well-known procedure where three independent Poisson random variables are combined appropriately to form two correlated random variables. Specifically, to generate the Poisson random variables X_1, X_2 with the respective parameters λ, λ', and correlation $\rho > 0$,

TR first generates three independent Poisson random variables Y_1, Y_2, and Y_{12}, with parameters λ_1, λ_2, and λ_{12}, respectively. The generated random variables are then combined as

$$X_1 = Y_1 + Y_{12},$$
$$X_2 = Y_2 + Y_{12},$$

to obtain X_1 and X_2. Because the sum of independent Poisson random variables is itself a Poisson random variable, the resulting random variables X_1, X_2 are each Poisson with parameters $\lambda_1+\lambda_{12}$ and $\lambda_2+\lambda_{12}$, respectively. The parameters λ_1, λ_2, and λ_{12} are chosen to match the target means λ, λ', and the target correlation ρ, by solving the following system:

$$\begin{aligned} \lambda &= \lambda_1 + \lambda_{12}, \\ \lambda' &= \lambda_2 + \lambda_{12}, \\ \rho &= \frac{\lambda_{12}}{\sqrt{(\lambda_1+\lambda_{12})(\lambda_2+\lambda_{12})}}. \end{aligned} \tag{1}$$

Solving the system (1) gives us

$$\lambda_{12} = \rho\sqrt{\lambda\lambda'}, \qquad \lambda_1 = \lambda - \lambda_{12}, \qquad \lambda_2 = \lambda' - \lambda_{12}. \tag{2}$$

Although elegant, TR has two important drawbacks that frequently render it unusable:

D.1. TR cannot be used when the target correlation ρ is negative; and

D.2. Even when the target correlation ρ is positive, the vector (X_1, X_2) obtained through TR may not be able to attain the target correlation while also achieving the specified marginal distributions.

The disadvantage D.1 is evident since the covariance $\mathrm{Cov}(X_1, X_2) = \mathrm{Var}(Y_{12}) = \lambda_{12} > 0$. To see disadvantage D.2, we notice that the solution (2), to the system of equations in (1), implies that λ, λ', and ρ should satisfy $\lambda \geq \rho\sqrt{\lambda\lambda'}$ and $\lambda' \geq \rho\sqrt{\lambda\lambda'}$, or equivalently, $\rho \leq \sqrt{k}$ where $k = \mathrm{Min}(\lambda, \lambda')/\mathrm{Max}(\lambda, \lambda')$. Otherwise, one of λ_1 and λ_2 will be negative, implying that TR cannot be used to generate the vector (X_1, X_2) with the desired marginal distributions and correlation. This points to a rather serious problem in TR: as the discrepancy between the desired means λ and λ' increases, the range of correlations that can be handled by TR shrinks. For example, if $\lambda_1 = 0.9$ and $\lambda_2 = 9$, the maximum possible correlation that can be handled by TR is $\sqrt{0.9/9} = 0.316$. The region $\rho \in (0, 0.316)$ that can be handled by TR for this particular example is depicted in Figure 1 as the dotted line segment joining B and C. The *entire* feasible region is $\rho \in (-0.8733, 0.9187)$ and is depicted in Figure 1 as the dotted line segment joining A and D.

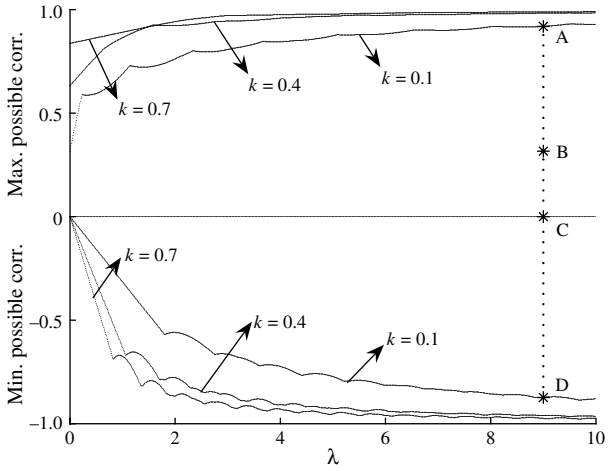

Figure 1 Maximum and Minimum Achievable Correlations Between Two Poisson Random Variables

Note. In the figure, λ is the larger of the two desired means, and k is the ratio of the smaller desired mean to the larger desired mean.

1.1.2. NORTA Procedure. NORTA is, by far, the most general and popular method of generating correlated random vectors. The general principle at work in NORTA is to first generate the "appropriate" normal random variables and then transform them to obtain the required random variables. Specifically, to generate a random vector $X = (X_1, X_2, \ldots, X_d)$ having marginals $(F_1, F_2, \ldots, F_d)$ and correlation matrix $\rho = (\rho_{ij})_{d\times d}$, use the following procedure.

1. Generate a multivariate normal random vector $Z(\rho^*) = (Z_1(\rho^*), Z_2(\rho^*), \ldots, Z_d(\rho^*))$, where $Z_i(\rho^*)$, $i = 1, 2, \ldots, d$, are standard normal random variables, and the vector $Z(\rho^*)$ has the correlation matrix $\rho^* = (\rho^*_{ij})_{d\times d}$.

2. Deliver $X = (X_1, X_2, \ldots, X_d)$ through $X_i = F_i^{-1}(\Phi(Z_i(\rho^*)))$, where $F_i^{-1}(u) = \inf\{u\colon F(x) \geq u\}$ and Φ is the univariate standard normal distribution.

The preprocessing phase in NORTA thus involves the identification of the correlation matrix $\rho^* = (\rho^*_{ij})_{d\times d}$, to be used in Step 1 of the NORTA procedure, so that the correlation matrix of the returned vector X in Step 2 is $\rho = (\rho_{ij})_{d\times d}$. For the current context, $d = 2$ and the marginal distributions are Poisson with parameters λ_1 and λ_2. Therefore, the preprocessing phase in NORTA amounts to finding a scalar ρ^* such that

$$\begin{aligned} &\mathrm{Corr}(F_1^{-1}(\Phi(Z_1(\rho^*))), F_2^{-1}(\Phi(Z_2(\rho^*)))) \\ &= \frac{\mathrm{E}[F_1^{-1}(\Phi(Z_1(\rho^*)))F_2^{-1}(\Phi(Z_2(\rho^*)))] - \lambda_1\lambda_2}{\sqrt{\lambda_1\lambda_2}} = \rho. \end{aligned} \tag{3}$$

The NORTA procedure is very general and works particularly well when the support of the required marginals is continuous (Chen 2001; Cario and Nelson 1997, 1998). However, when the support of one or more of the random variables $X_i = F_i^{-1}(\Phi(Z_i(\rho^*)))$,

$i = 1, 2$, is denumerable (countably infinite) as in the current context, the root-finding problem (3) turns out to be nontrivial and difficult to solve efficiently. As elaborated in Avramidis et al. (2009), the main difficulty lies in efficiently and accurately computing the function $g(\rho^*) = \mathrm{E}[F_1^{-1}(\Phi(Z_1(\rho^*)))F_2^{-1}(\Phi(Z_2(\rho^*)))]$, which, in the denumerable case, turns out to be an infinite double sum with the summand being a bivariate normal tail probability. We say more on this in §5.2, where we discuss a fast implementation of NORTA and compare it with that of the proposed algorithm.

1.2. Contributions

In this paper, we extend TR appropriately to propose TREx—a tailored algorithm for generating bivariate Poisson random vectors. The following are specific contributions of this work.

1. We characterize and depict the theoretical limits of the feasible range of correlations for a given pair of Poisson distributions (§2, Propositions 1–5).

2. We describe and list TREx, an algorithm for generating correlated Poisson random vectors (§3). The algorithm is exact and covers all theoretically feasible correlations in two dimensions. We detail a fast algorithm for solving the preprocessing phase in TREx, along with directives on implementation (§4).

3. We provide rigorous bounds for truncation error control (§4.2, Propositions 9 and 10) useful for TREx implementation.

4. A minor contribution is obtaining bounds for the error incurred in truncating the double infinite sum when computing $\mathrm{E}[F_1^{-1}(\Phi(Z_1(\rho^*)))F_2^{-1}(\Phi(Z_2(\rho^*)))]$ within NORTA (Proposition 11). Although we present this bound for the Poisson case in §5, it may be useful in other NORTA contexts where the marginal distributions are denumerable.

5. All code is available at https://filebox.vt.edu/users/pasupath/pasupath.htm. Specifically, the website provides (i) a fast module ("maxcorr") that calculates the maximum and minimum allowable correlation between any two Poisson random variables, and (ii) an implementation of TREx that incorporates the error bounds detailed in the paper.

1.3. Organization

The remainder of the paper is organized as follows. In §2 we characterize the structure of the feasible region of correlations between two Poisson random variables as a function of the Poisson parameters λ_1, λ_2. In §3, we present a detailed description and listing of the TREx algorithm. This is followed by §4, where we provide an algorithm for executing the preprocessing phase of TREx, including directives on implementation. Section 5 describes results from extensive numerical tests on the preprocessing phases of TREx and NORTA. We provide concluding remarks in §6.

2. Structure of the Feasible Region

In this section, we depict the feasible set of correlations between two Poisson random variables X_1, X_2 with respective parameters λ, λ'. We first present the following definition introduced by Ghosh and Henderson (2003).

Definition 1. A product-moment (rank) correlation matrix Σ is *feasible* for a given set of marginal distributions $F_1, F_2, \ldots, F_d$ if there exists a random vector X with marginal distributions $F_1, F_2, \ldots, F_d$ and product-moment (rank) correlation matrix Σ.

To illustrate feasible correlation matrices, consider two Poisson random variables X_1 and X_2 having respective means $\lambda = 0.5$ and $\lambda' = 0.5$. It can be shown that the largest achievable positive correlation between X_1 and X_2 is 1, and the largest achievable negative correlation between X_1 and X_2 is -0.5. Therefore, any correlation matrix

$$\Sigma = \begin{pmatrix} 1 & r \\ r & 1 \end{pmatrix},$$

with r values in the interval $[-0.5, 1]$ is a *feasible* correlation matrix for the vector (X_1, X_2). The matrix Σ is a correlation matrix, but not feasible, if r lies in the interval $[-1, -0.5)$.

More generally, as shown in Whitt (1976), if random variables X_1 and X_2 have cumulative distribution functions (cdfs) $F(x)$ and $G(x)$, respectively, and U is a random variable that is uniformly distributed between 0 and 1, then $\mathrm{Corr}(F^{-1}(U), G^{-1}(U))$ is the maximum achievable and $\mathrm{Corr}(F^{-1}(U), G^{-1}(1-U))$ the minimum achievable correlations between X_1 and X_2, respectively. Therefore the *feasible* set of correlations between the random variables X_1 and X_2 is $[\mathrm{Corr}(F^{-1}(U), G^{-1}(1-U)), \mathrm{Corr}(F^{-1}(U), G^{-1}(U))]$.

Figure 1 depicts this feasible set when X_1 and X_2 are Poisson random variables. The figure is plotted as a function of the larger desired mean λ (assumed without loss of generality) of X_1 and X_2, and the ratio k of the smaller to the larger desired means of X_1 and X_2. Thus, for a given λ and k, a vertical line between the corresponding upper and lower curves depicts the range of feasible correlations.

Five properties of the curves depicted in Figure 1 are noteworthy.

—Each curve is continuous everywhere in $\lambda \in (0, \infty)$ (Proposition 1). Furthermore, the set of points where each curve is nondifferentiable has Lebesgue measure zero (Proposition 7).

—There is an initial linear region for every negative correlation curve (see the bottom half of Figure 1). This corresponds to the region $\{\lambda: F_\lambda^{-1}(u)F_{k\lambda}^{-1}(1-u) = 0$ for all $u \in [0, 1]\}$.

—For a given ratio of the two parameters, i.e., for fixed k, the maximum positive and maximum negative correlations tend to 1 and -1, respectively, as $\lambda \to \infty$ (Proposition 3).

—The curves are in the form of "scallops" with the ends of the scallops corresponding to the points of nondifferentiability.

—The curves become "approximately linear" when multiplied by the factor $\lambda\sqrt{k}$.

Propositions 1 through 5 characterize the limiting behavior of these curves rigorously. As stated earlier, assume X_1 and X_2 are Poisson random variables with means λ and λ'. Also assume, without loss in generality, that $\lambda \geq \lambda'$. Denote the maximum and minimum achievable correlation between X_1 and X_2 as $\rho^+(\lambda, \lambda')$ and $\rho^-(\lambda, \lambda')$, respectively. Also, denote $k = \lambda'/\lambda$. Proofs for Propositions 1 through 5 are provided in the Online Supplement (available at http://joc.pubs.informs.org/ecompanion.html).

Proposition 1. *Functions* $\rho^+(\lambda, k\lambda)$, $\rho^-(\lambda, k\lambda)$ *are continuous in* $(\lambda, k) \in (0, \infty) \times (0, 1]$.

Proposition 2. *For fixed* k,

$$\lim_{\lambda\to 0} \rho^+(\lambda, k\lambda) = \sqrt{k}; \qquad \lim_{\lambda\to 0} \rho^-(\lambda, k\lambda) = 0.$$

Proposition 3. *For fixed* k,

$$\lim_{\lambda\to \infty} \rho^+(\lambda, k\lambda) = 1; \qquad \lim_{\lambda\to \infty} \rho^-(\lambda, k\lambda) = -1.$$

Proposition 4. *For fixed* λ,

$$\lim_{k\to 0} \rho^+(\lambda, k\lambda) = 0; \qquad \lim_{k\to 0} \rho^-(\lambda, k\lambda) = 0.$$

Proposition 5. *For fixed* λ,

$$\lim_{k\to 1} \rho^+(\lambda, k\lambda) = \rho^+(\lambda, \lambda); \qquad \lim_{k\to 1} \rho^-(\lambda, k\lambda) = \rho^-(\lambda, \lambda).$$

3. TREx—Algorithm Description

Recall that the objective is to generate the random vector (X_1, X_2) such that X_1 has a Poisson distribution with mean λ, X_2 has a Poisson distribution with mean λ', and $\mathrm{Corr}(X_1, X_2) = \rho$, where $\lambda, \lambda' > 0$ and $\rho \in (-1, 1)$ are given.

Our assumption about $\rho \in (-1, 1)$ creates the possibility of the desired correlation being infeasible; i.e., $\rho > \rho^+(\lambda, k\lambda)$ or $\rho < \rho^-(\lambda, k\lambda)$. This problem of infeasibility is not a complication because it is automatically detected at the end of the preprocessing step. In other words, the proposed algorithm is such that nothing special needs to be done to check for an infeasible problem.

Denote $F_\lambda^{-1}(y) = \inf\{x : F_\lambda(x) > y\}$, where $F_\lambda(x)$ is the Poisson cdf with mean λ. Let U be a random variable that is uniformly distributed between 0 and 1. Then the proposed algorithm takes the following form:

$$\begin{aligned} &X_1 = Y_1 + F_{\lambda^*}^{-1}(U), \quad X_2 = Y_2 + F_{k\lambda^*}^{-1}(U) \quad \text{if } \rho > 0; \\ &X_1 = Y_1 + F_{\lambda^*}^{-1}(U), \quad X_2 = Y_2 + F_{k\lambda^*}^{-1}(1-U) \quad \text{if } \rho < 0. \end{aligned} \tag{4}$$

We draw attention to three aspects of the proposed operations. First, when $\rho > 0$, i.e., when positive correlation between X_1 and X_2 is sought, we use common random numbers as in TR. When $\rho < 0$, we use antithetic variates to induce negative correlation between X_1 and X_2. Second, we note that for both cases, $\rho > 0$ and $\rho < 0$, unlike TR, there is no "common random variable." Instead, the random variables inducing correlation are obtained through inversion of two different Poisson cdfs. The means of these Poisson cdfs are in the same ratio as the target means λ and λ'. Third, the value of λ^* needs to be determined as part of the preprocessing step so that the resulting random variables X_1, X_2 attain the target means and the target correlation.

3.1. TREx—Algorithm Listing

We list the operations involved in TREx as Algorithm 1. We discuss Step 7 (preprocessing step) in detail in §4. Inverting a Poisson cdf, required in various steps, can be done efficiently through existing random variate generation routines (Kemp and Kemp 1991, Schmeiser and Kachitvichyanukul 1981, Devroye 1986).

Algorithm 1 (TREx)

Require: $\lambda > 0$, $\lambda' > 0$, $\rho \in (-1, 1)$

1: **if** $\rho = 0$ **then**
2: Generate $U_1 \sim U(0,1)$, $U_2 \sim U(0,1)$, independently
3: $X_1 \leftarrow F_\lambda^{-1}(U_1)$
4: $X_2 \leftarrow F_{\lambda'}^{-1}(U_2)$
5: **return** (X_1, X_2)
6: **end if**
7: Solve for λ^* {preprocessing step}
8: Generate $U_1 \sim U(0,1)$, $U_2 \sim U(0,1)$, $U_3 \sim U(0,1)$ independently
9: $Y_1 \leftarrow F_{\lambda-\lambda^*}^{-1}(U_1)$
10: $Y_2 \leftarrow F_{\lambda'-k\lambda^*}^{-1}(U_2)$
11: $Y_{12} \leftarrow F_{\lambda^*}^{-1}(U_3)$
12: **if** $\rho > 0$ **then**
13: $Y'_{12} \leftarrow F_{k\lambda^*}^{-1}(U_3)$
14: **else**
15: $Y'_{12} \leftarrow F_{k\lambda^*}^{-1}(1-U_3)$
16: **end if**
17: $X_1 \leftarrow Y_1 + Y_{12}$
18: $X_2 \leftarrow Y_2 + Y'_{12}$
19: **return** (X_1, X_2)

3.2. Rationale

It is clear that TREx addresses the disadvantage D.1 in TR. What is not immediately evident is the fact that TREx fully addresses disadvantage D.2 as well. To see this, consider the $\rho > 0$ operation in (4). It is clear from construction that the random variables Y_1, Y_2, $F_{\lambda^*}^{-1}(U)$, and $F_{k\lambda^*}^{-1}(U)$ are each Poisson distributed with respective means $\lambda - \lambda^*$, $\lambda' - k\lambda^*$, λ^*, and $k\lambda^*$. Therefore,

the random variables X_1 and X_2 will have the correct marginal distributions, provided the quantities $\lambda - \lambda^*$ and $\lambda' - k\lambda^*$ remain positive. This, however, can be ensured by restricting λ^* to the interval $[0, \lambda]$, after recalling that $\lambda \geq \lambda^*$ and $k \leq 1$. A similar argument holds for the $\rho < 0$ case as well.

What range of correlations are covered if we restrict λ^* to the interval $[0, \lambda]$? To answer this question, again consider the $\rho > 0$ case in (4). As $\lambda^* \to 0$, we have $\mathrm{Corr}(X_1, X_2) \to 0$, giving us the trivial uncorrelated case. On the other extreme, as $\lambda^* \to \lambda$, we have $\lambda - \lambda^* \to 0$, $\lambda' - k\lambda^* \to 0$, and $k\lambda^* \to \lambda'$. These three implications together mean that Y_1 and Y_2 vanish, and $\mathrm{Corr}(X_1, X_2) \to \mathrm{Corr}(F_\lambda^{-1}(U), F_{k\lambda}^{-1}(U)) = \rho^+(\lambda, k\lambda)$. Furthermore, it can be shown that $\mathrm{Corr}(X_1, X_2)$ is a continuous function of λ^*. These three facts—$\mathrm{Corr}(X_1, X_2) \to 0$ as $\lambda^* \to 0$, $\mathrm{Corr}(X_1, X_2) \to \rho^+(\lambda, k\lambda)$ as $\lambda^* \to \lambda$, and the continuity of $\mathrm{Corr}(X_1, X_2)$ as a function of λ^*—ensure that the entire range of positive correlations $[0, \rho^+(\lambda, k\lambda)]$ can be achieved through TREx. Similar arguments for the $\rho < 0$ case imply that TREx achieves the entire range of negative correlations $[\rho^-(\lambda, k\lambda), 0]$ as well.

Before we state the above arguments formally through Proposition 6, we also note in passing that we can achieve a similar effect, i.e., obtaining the entire range of feasible correlations, through

$$\begin{aligned} &X_1 = Y_1 + F_{\lambda_1^*}^{-1}(U), \quad X_2 = Y_2 + F_{\lambda_2^*}^{-1}(U) \quad \text{if } \rho > 0; \\ &X_1 = Y_1 + F_{\lambda_1^*}^{-1}(U), \quad X_2 = Y_2 + F_{\lambda_2^*}^{-1}(1-U) \quad \text{if } \rho < 0; \end{aligned} \tag{5}$$

instead of (4). The operation (5), however, provides no advantages over (4), at least in two dimensions. It does have the disadvantage of making the preprocessing step a two-dimensional search, as opposed to the one-dimensional search afforded by (4). The proof of Proposition 6 is a simple consequence of Proposition 1.

Proposition 6. *Let Y_1, Y_2 be Poisson random variables with means $\lambda - \lambda^*$ and $\lambda' - k\lambda^*$, respectively, where $0 < \lambda^* \leq \lambda$ and $k = \lambda'/\lambda \leq 1$. Let U be a random variable that is mutually independent of Y_1 and Y_2, and uniformly distributed between 0 and 1. Then*

(i) *the functions* $\mathrm{Corr}(Y_1 + F_{\lambda^*}^{-1}(U), Y_2 + F_{k\lambda^*}^{-1}(U))$, $\mathrm{Corr}(Y_1 + F_{\lambda^*}^{-1}(U), Y_2 + F_{k\lambda^*}^{-1}(1-U))$ *are continuous in* $\lambda^* \in (0, \lambda]$;

(ii) $\lim_{\lambda^* \to \lambda} \mathrm{Corr}(Y_1 + F_{\lambda^*}^{-1}(U), Y_2 + F_{k\lambda^*}^{-1}(U)) = \rho^+(\lambda, k\lambda)$; *and*

(iii) $\lim_{\lambda^* \to \lambda} \mathrm{Corr}(Y_1 + F_{\lambda^*}^{-1}(U), Y_2 + F_{k\lambda^*}^{-1}(1-U)) = \rho^-(\lambda, k\lambda)$.

4. TREx Preprocessing Step (Solving for λ^*)

We see from (4) that X_1 and X_2 have the correct marginal distributions. The more challenging question is that of identifying λ^* so that the target correlation ρ is attained. In this section, we detail a fast numerical procedure that can be used to identify λ^*.

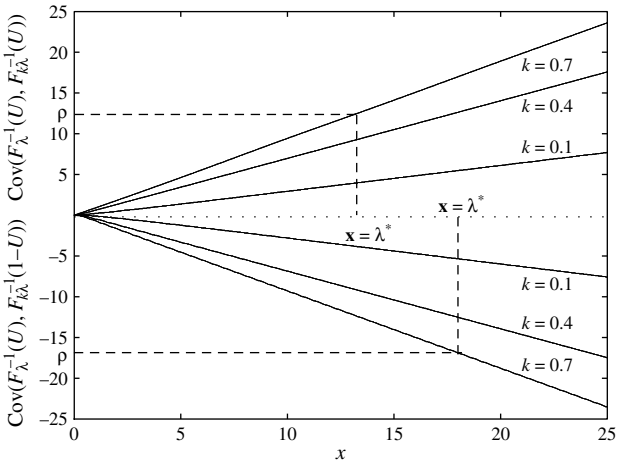

Figure 2 **Preprocessing Through One-Dimensional Root-Finding Search on the Covariance Function $h(x)$**

From (4), the correlation $\mathrm{Corr}(X_1, X_2)$ as a function of λ, k, and λ^* is given by

$$\mathrm{Corr}(X_1, X_2) = \begin{cases} \dfrac{1}{\lambda\sqrt{k}}(\mathrm{E}[F_{\lambda^*}^{-1}(U)F_{k\lambda^*}^{-1}(U)] - k\lambda^{*2}) \\ \quad \text{if } \rho > 0, \\ \dfrac{1}{\lambda\sqrt{k}}(\mathrm{E}[F_{\lambda^*}^{-1}(U)F_{k\lambda^*}^{-1}(1-U)] - k\lambda^{*2}) \\ \quad \text{if } \rho < 0. \end{cases}$$

From the above expression for $\mathrm{Corr}(X_1, X_2)$, identifying λ^* satisfying $\mathrm{Corr}(X_1, X_2) = \rho$ amounts to solving the following generic root-finding problem, as depicted in Figure 2. Given λ, k, ρ,

$$\text{find } x = \lambda^* \text{ satisfying } h(x) = \rho\lambda\sqrt{k}, \tag{6}$$

where

$$h(x) = \begin{cases} \mathrm{E}[F_x^{-1}(U)F_{kx}^{-1}(U)] - kx^2 & \text{if } \rho > 0, \\ \mathrm{E}[F_x^{-1}(U)F_{kx}^{-1}(1-U)] - kx^2 & \text{if } \rho < 0. \end{cases} \tag{7}$$

The existence of a solution to the problem in (6) is evident from the following facts and the intermediate value theorem (Bartle 1976, p. 153): (i) $h(x)$ is continuous (see Proposition 1), (ii) $\lim_{x \to 0} h(x) = 0$ (see Proposition 2), and (iii) $\lim_{x \to \infty} h(x) = \infty$ if $\rho > 0$ and $-\infty$ if $\rho < 0$ (see Proposition 3). There is overwhelming numerical evidence in support of $h(x)$ being strictly monotone. We have, however, been unable to prove this rigorously.

4.1. Recursion, Function, and Derivative Computation

In this section, we present a solution for the root-finding problem in (6). After noting that the function

h is differentiable almost everywhere through Proposition 7, we detail the efficient computation of $h(x)$ and its derivative $h'(x)$, and provide rigorous directives on safely truncating the summations that appear during their computation.

PROPOSITION 7. *The real-valued function*

$$h(x) = \begin{cases} E[F_x^{-1}(U)F_{kx}^{-1}(U)] - kx^2 & \text{if } \rho > 0, \\ E[F_x^{-1}(U)F_{kx}^{-1}(1-U)] - kx^2 & \text{if } \rho < 0 \end{cases}$$

is differentiable almost everywhere.

PROOF. Since the products $F_x^{-1}(U)F_{kx}^{-1}(U)$ and $F_x^{-1}(U)F_{kx}^{-1}(1-U)$ are each nondecreasing in x (for fixed U), the functions $E[F_x^{-1}(U)F_{kx}^{-1}(U)]$ and $E[F_x^{-1}(U)F_{kx}^{-1}(1-U)]$ are both nondecreasing in x. This implies, however, that $E[F_x^{-1}(U)F_{kx}^{-1}(U)]$ and $E[F_x^{-1}(U)F_{kx}^{-1}(1-U)]$ are differentiable almost everywhere (Royden 1988, p. 100). Conclude that $h(x)$ is differentiable almost everywhere. □

For solving the root-finding problem (6), we use a Newton recursion on $h(x)$:

$$x = x + \frac{1}{h'(x)}(\rho\lambda\sqrt{k} - h(x)). \tag{8}$$

In what follows, we elaborate on the efficient computation of $h(x)$, $h'(x)$ appearing in (8).

Case $\rho < 0$: When $\rho < 0$, we note that $E[F_x^{-1}(U) \cdot F_{kx}^{-1}(1-U)] = 0$ when x is small enough, i.e., if $F_x(0) + F_{kx}(0) = e^{-x} + e^{-kx} \geq 1$. In such a case, $h(x) = -kx^2$ implies that $\lambda^* = \sqrt{-\rho\lambda/\sqrt{k}}$. Otherwise, we compute $h(x) = \int_0^1 F_x^{-1}(u)F_{kx}^{-1}(1-u)\,du - kx^2$ starting from the "middle region" of the integral and progressively summing out to the tails.

To do this, we first compute $m_x = \text{Min}\{k \in Z^+: F_x(k) \geq 0.5\}$, $m_{kx} = \text{Min}\{k \in Z^+: F_{kx}(k) \geq 0.5\}$, and the corresponding cumulative probabilities $F_x(m_x)$, $F_{kx}(m_{kx})$, where $Z^+ = \{0, 1, 2, \ldots\}$ denotes the set of nonnegative integers. An efficient way to compute these is through J-fraction approximations given in Kemp (1988). These approximations are highly accurate analytic expressions for $F_x(r)$, where r is the "round-off" value of x, i.e., the integer that satisfies $r + \alpha = x$, $-0.5 \leq \alpha < 0.5$. We then express

$$h(x) = \int_0^{0.5} F_x^{-1}(u)F_{kx}^{-1}(1-u)\,du + \int_0^{0.5} F_{kx}^{-1}(u)F_x^{-1}(1-u)\,du - kx^2. \tag{9}$$

The first of the integrals on the right-hand side of (9) is 0 if $m_x = 0$. Otherwise,

$$\int_0^{0.5} F_x^{-1}(u)F_{kx}^{-1}(1-u)\,du = S + \sum_{j=m_x-1,\, m_x-2,\, \ldots,\, 1} S_j, \tag{10}$$

where

$$l_j = \text{Min}\{n \in Z^+ : 1 - F_{kx}(n) \leq F_x(j)\}, \quad \text{for } j = 1, 2, \ldots;$$

$$S_j = \begin{cases} jl_j(F_x(j) - F_x(j-1)), & \text{if } 1 - F_{kx}(l_j) < F_x(j-1); \\ B_j + C_j + \sum_{i=l_j}^{l_{j-1}-2} T_{ij}, & \text{otherwise;} \end{cases}$$

$$B_j = jl_j(F_x(j) - 1 + F_{kx}(l_j));$$
$$C_j = jl_{j-1}(1 - F_{kx}(l_{j-1} - 1) - F_x(j-1));$$
$$T_{ij} = (i+1)j(F_{kx}(i+1) - F_{kx}(i));$$

$$S = \begin{cases} m_x m_{kx}(0.5 - F_x(m_x - 1)), & \text{if } 1 - F_{kx}(m_{kx}) \leq F_x(m_x - 1); \\ B + C + \sum_{i=m_{kx}}^{l_{m_x-1}-2} T_{im_x}, & \text{otherwise;} \end{cases}$$

$$B = m_x m_{kx}(F_{kx}(m_{kx}) - 0.5);$$
$$C = m_x l_{m_x-1}(1 - F_{kx}(l_{m_x-1} - 1) - F_x(m_x - 1)).$$

The expression in (10) is obtained upon noting that the function $F_x^{-1}(u)$ takes the value $m_x - 1$ in the interval $[F_x(m_x - 1), 0.5]$, $m_x - 2$ in the interval $[F_x(m_x - 2), F_x(m_x - 1))$, $m_x - 3$ in the interval $[F_x(m_x - 3), F_x(m_x - 2))$, and so on. Similarly, the function $F_{kx}^{-1}(1-u)$ takes the value m_{kx} in the interval $[1 - F_{kx}(m_{kx}), 0.5]$, $m_{kx} + 1$ in the interval $[1 - F_{kx}(m_{kx} + 1), 1 - F_{kx}(m_{kx}))$, $m_{kx} + 2$ in the interval $[1 - F_{kx}(m_{kx} + 2), 1 - F_{kx}(m_{kx} + 1))$, and so on. The integral on the left-hand side of (10) can thus be expressed as a summation by splitting the interval $[0, 0.5]$ into sub-intervals starting from 0.5 and obtained by arranging the numbers $0.5, F_x(m_x - 1), 1 - F_{kx}(m_{kx}), F_x(m_x - 2), 1 - F_{kx}(m_{kx} + 1), \ldots$, in descending order. The first such subinterval gives rise to the summand S as defined, and the $(j+1)$th subinterval gives rise to the summand S_j as defined. (The summation on the right-hand side of (10) has only a finite number of terms because $F_x^{-1}(u) = 0$ for $u \in [0, F_x(0)]$.) A similar calculation applies to the second integral appearing on the right-hand side of (9) and for the expressions in (12) and (13) below.

Case $\rho > 0$: For this case, unlike the $\rho < 0$ case, there exists no linear portion of the curves in Figure 1. Again, we first compute $m_x = \text{Min}\{k \in Z^+: F_x(k) \geq 0.5\}$, $m_{kx} = \text{Min}\{k \in Z^+: F_{kx}(k) \geq 0.5\}$, and the corresponding cumulative probabilities $F_x(m_x)$, $F_{kx}(m_{kx})$ using the J-fraction approximations in Kemp and Kemp (1991). We again express $h(x)$ in two parts as

$$h(x) = \int_0^{0.5} F_x^{-1}(u)F_{kx}^{-1}(u)\,du + \int_{0.5}^1 F_x^{-1}(u)F_{kx}^{-1}(u)\,du - kx^2. \tag{11}$$

The first of the integrals in (11) is 0 if either $m_x = 0$ or $m_{kx} = 0$. Otherwise,

$$\int_0^{0.5} F_x^{-1}(u)F_{kx}^{-1}(u)\,du = S + \sum_{j=m_x-1,\, m_x-2,\, \ldots,\, 1} S_j, \tag{12}$$

where

$$l_j = \text{Max}\{n \in Z^+ \colon F_{kx}(n) \le F_x(j)\}, \quad \text{for } j = 1, 2, \ldots;$$

$$S_j = \begin{cases} j(l_j+1)(F_x(j) - F_x(j-1)), & \text{if } F_{kx}(l_j) < F_x(j-1); \\ B_j + C_j + \sum_{i=l_j}^{l_{j-1}+2} T_{ij}, & \text{otherwise;} \end{cases}$$

$$B_j = j(l_j+1)(F_x(j) - F_{kx}(l_j));$$

$$C_j = j(l_{j-1}+1)(F_{kx}(l_{j-1}+1) - F_x(j-1));$$

$$T_{ij} = ji(F_{kx}(i) - F_{kx}(i-1));$$

$$S = \begin{cases} m_x m_{kx}(0.5 - F_x(m_x - 1)), \\ \qquad \text{if } F_{kx}(m_{kx} - 1) \le F_x(m_x - 1); \\ B + C + \sum_{i=m_{kx}-1}^{l_{m_x-1}+2} T_{im_x}, \quad \text{otherwise;} \end{cases}$$

$$B = m_x m_{kx}(0.5 - F_{kx}(m_{kx} - 1));$$

$$C = m_x(l_{m_x-1}+1)(F_{kx}(l_{m_x-1}+1) - F_x(m_x - 1)).$$

Similarly, the second integral on the right-hand side of (11) can be written as

$$\int_{0.5}^{1} F_x^{-1}(u) F_{kx}^{-1}(u)\, du = S + \sum_{j=m_x+1}^{\infty} S_j, \tag{13}$$

where

$$l_j = \text{Max}\{n \in Z^+ \colon F_{kx}(n) \le F_x(j)\}, \quad \text{for } j = 1, 2, \ldots;$$

$$S_j = \begin{cases} j(l_j + 1)(F_x(j) - F_x(j-1)), & \text{if } F_{kx}(l_j) < F_x(j-1); \\ B_j + C_j + \sum_{i=l_{j-1}+1}^{l_j - 1} T_{ij}, & \text{otherwise;} \end{cases}$$

$$B_j = j(l_j + 1)(F_x(j) - F_{kx}(l_j));$$

$$C_j = j(l_{j-1} + 1)\left(F_{kx}(l_{j-1} + 1) - F_x(j-1)\right);$$

$$T_{ij} = j(i+1)(F_{kx}(i+1) - F_{kx}(i));$$

$$S = \begin{cases} m_x m_{kx}(F_x(m_x) - 0.5), & \text{if } F_{kx}(m_{kx}) \ge F_x(m_x); \\ B + C + \sum_{i=m_{kx}}^{l_{m_x}-1} T_{im_x}, & \text{otherwise;} \end{cases}$$

$$B = m_x(l_{m_x} + 1)(F_x(m_x) - F_{kx}(l_{mx}));$$

$$C = m_x m_{kx}(F_{kx}(m_{kx}) - 0.5).$$

The derivative $h'(x)$ can be obtained through direct differentiation of the summation expressions for $h(x)$, after noting the derivatives (with respect to x) $F_x'(0) = -e^{-x}$, $F_{kx}'(0) = -ke^{-kx}$, and $F_x'(i) = -P_x(i)$, $F_{kx}'(i) = -kP_{kx}(i)$ for $i = 1, 2, \ldots$.

4.2. Bounds on Truncation Error

As described in §4.1, computing $h(x)$ is based on a summation involving a potentially large number of tail probabilities from specified Poisson distributions. From a computational standpoint, it would be useful to truncate this summation, while making sure that the terms excluded add to less than a prespecified tolerance ϵ. In this section, we present results that provide directives for such safe truncation. We first note the following identities related to the moments of the Poisson distribution. See the Online Supplement for a proof.

Proposition 8. *Let P_x and F_x denote the probability mass function and cumulative distribution function, respectively, of the Poisson distribution with mean x. Then,*

(i) $\sum_{j=s}^{\infty} jP_x(j) = x(1 - F_x(s-2))$, $s \in Z$;

(ii) $\sum_{j=s}^{\infty} j^2 P_x(j) = x^2(1 - F_x(s-3)) + x(1 - F_x(s-2))$, $s \in Z$;

(iii) $\sum_{j=0}^{s} j^2 P_x(j) = x^2(F_x(s-2)) + x(F_x(s-1))$, $s \in Z$.

Recall that when $\rho < 0$, we wrote $h(x) = \int_0^{0.5} F_x^{-1}(u) F_{kx}^{-1}(1-u)\, du + \int_0^{0.5} F_{kx}^{-1}(u) F_x^{-1}(1-u)\, du$. We also expressed each of these integrals through a double summation (10) that starts from the center, i.e., from $u = 0.5$, and sums outward to $u = 0$. Proposition 9, proved in the Online Supplement, provides a bound on the error because of truncating each of these summations.

Proposition 9. *Let F_x and F_{kx} represent Poisson cdfs with respective means x and kx. Then,*

$$\left| \int_0^{0.5} F_x^{-1}(u) F_{kx}^{-1}(1-u)\, du - \int_{\delta}^{0.5} F_x^{-1}(u) F_{kx}^{-1}(1-u)\, du \right| \le F_x^{-1}(\delta) kx (1 - F_{kx}(F_{kx}^{-1}(1-\delta) - 3)). \tag{14}$$

In illustrating the usefulness of Proposition 9, suppose that we have summed to $u = \delta > 0$ and that our required tolerance in computing $h(x)$ is ϵ. Then, apply the error bound in (14) to each integral comprising $h(x)$ individually by stopping the summation when the right-hand side of (14) falls below $\epsilon/2$. Because every term in the right-hand side of (14) is known, checking the error bound is also computationally cheap.

We next present a similar truncation error bound for the $\rho > 0$ case. Recall that for the $\rho > 0$ case, $h(x) = \int_0^{0.5} F_x^{-1}(u) F_{kx}^{-1}(u) + \int_{0.5}^{1} F_x^{-1}(u) F_{kx}^{-1}(u)$, with the individual integrals being expressed as double summations shown in (12) and (13). Proposition 10 provides separate truncation error bounds for these, with a proof in the Online Supplement.

PROPOSITION 10. *Let F_x and F_{kx} represent Poisson cdfs with respective means x and kx. Then,*

(i)

$$\left|\int_0^{0.5} F_x^{-1}(u)F_{kx}^{-1}(u)\,du - \int_\delta^{0.5} F_x^{-1}(u)F_{kx}^{-1}(u)\,du\right| \le \sqrt{I(x, F_x^{-1}(\delta))I(kx, F_{kx}^{-1}(\delta))},$$

where $I(x,y) = x^2F_x(y-2) + xF_x(y-1)$.

(ii)

$$\left|\int_{0.5}^1 F_x^{-1}(u)F_{kx}^{-1}(u)\,du - \int_{0.5}^{1-\delta} F_x^{-1}(u)F_{kx}^{-1}(u)\,du\right| \le \sqrt{I'(x, F_x^{-1}(1-\delta)-1)I'(kx, F_{kx}^{-1}(1-\delta)-1)},$$

where $I'(x,y) = x^2(1-F_x(y-3)) + x(1-F_x(y-2))$.

Proposition 10, in a fashion similar to Proposition 9, suggests that we do not have to include all the summands appearing in (12) and (13). Instead, if ϵ is the prescribed tolerance, stop the summation for $\int_0^{0.5} F_x^{-1}(u)F_{kx}^{-1}(u)du$ when $I(x, F_x^{-1}(\delta))$ and $I(kx, F_{kx}^{-1}(\delta))$ each fall below $\sqrt{\epsilon/2}$. Similarly, stop the summation for $\int_{0.5}^1 F_x^{-1}(u)F_{kx}^{-1}(u)\,du$ when $I'(x, F_x^{-1}(1-\delta)-1)$ and $I'(kx, F_{kx}^{-1}(1-\delta)-1)$ each fall below $\sqrt{\epsilon/2}$.

4.3. Initial Guess

Motivated by Figure 2, the initial guess x_0 for the recursion (8) is obtained through a linear approximation $l(x)$ to the function $h(x)$. From Proposition 3, we see that for fixed k,

$$\lim_{x\to\infty} \frac{\mathrm{E}[F_x^{-1}(U)F_{kx}^{-1}(U)] - kx^2}{x} = \sqrt{k},$$
$$\lim_{x\to\infty} \frac{\mathrm{E}[F_x^{-1}(U)F_{kx}^{-1}(1-U)] - kx^2}{x} = -\sqrt{k}.$$

Thus, for a given problem instance, the initial guess x_0 for the recursion (8) is obtained by solving for x from the equation $l(x) = \mathrm{sign}(\rho)\sqrt{k}\,x + c = \rho\lambda\sqrt{k}$, to obtain $x_0 = (\rho\lambda\sqrt{k} - c)/(\mathrm{sign}(\rho)\sqrt{k})$. The intercept c of the linear approximation $l(x)$ is set heuristically. For instance, we recommend $c = 0$ for $\rho > 0$, and $c = \sqrt{k}b_m - kb_m^2$ for $\rho < 0$, where b_m is the boundary of the "initial linear region" for the negative correlation case. Recall that the boundary b_m is the solution to the equation $\mathrm{e}^{-b_m} + \mathrm{e}^{-kb_m} = 1$ and can be obtained rapidly through a Newton search.

We conclude this section with Algorithm 2—a formal algorithm listing of the preprocessing step in TREx.

Algorithm 2 (TREx preprocessing step)

Require: $\lambda_1 > 0$, $\lambda_2 > 0$, $-1 \le \rho \le 1$, $\epsilon > 0$

if $|\rho| \le \epsilon$ **then**
  return 0 {i.e., generate independently}
end if
$\lambda \leftarrow \mathrm{Max}(\lambda_1, \lambda_2)$
$k \leftarrow \mathrm{Min}(\lambda_1, \lambda_2)/\lambda$
$\mathrm{s} \leftarrow \mathrm{sign}(\rho)\sqrt{k}$
if $\rho < 0$ **then**
  Solve for b_m to within tolerance ϵ {i.e., find b_m satisfying $\mathrm{e}^{-b_m} + \mathrm{e}^{-kb_m} = 1$}
  if $\rho\sqrt{k}\lambda \ge -kb_m^2$ **then**
    return $\sqrt{-\rho\lambda/\sqrt{k}}$ {solution lies in the initial linear region}
  end if
  $c \leftarrow \sqrt{k}b_m - kb_m^2$
else
  $c \leftarrow 0$
end if
$\lambda^* = (\rho\sqrt{k}\lambda - c)/s$ {initial guess}
$\hat{\rho} \leftarrow 0$
while $|\hat{\rho} - \rho| > \epsilon$ **do**
  Calculate ρ^*, ρ_1^*

$$\rho^* = \rho^+(\lambda^*, k\lambda^*) \quad \text{if } \rho > 0;$$
$$= \rho^-(\lambda^*, k\lambda^*) \quad \text{if } \rho < 0;$$
$$\rho_1^* = \frac{d\rho^*}{d\lambda}.$$

  $\hat{\rho} = \rho^*\lambda^*/\lambda$
  $h'^{-1} \leftarrow \rho_1^*\sqrt{k}\lambda^* + \rho^*\sqrt{k}$
  $\lambda^* \leftarrow \lambda^* + h'^{-1}(\rho\sqrt{k}\lambda - \hat{\rho}\sqrt{k}\lambda^*)$ {Newton update}
end while

5. Computational Experience

Recall that both TREx and NORTA have two phases: (i) a preprocessing phase where the parameters required within the algorithm are identified, and (ii) a generation phase where the identified parameters are used appropriately to generate the required Poisson random vector. In this section, we report detailed results on execution times for (i). Our emphasis is on (i) because, as Table 1 demonstrates, TREx and NORTA are quite comparable in terms of execution times for (ii), with TREx being slightly faster.

Recall from §1 that the Poisson random variate generation problem has three problem parameters: the means λ_1, λ_2 of the marginal distributions, and a desired correlation ρ. A problem is thus characterized uniquely by the three parameters $(\lambda_1, \lambda_2, \rho)$, or equivalently by (λ, k, ρ), where $\lambda = \mathrm{Max}(\lambda_1, \lambda_2)$ and $k = \mathrm{Min}(\lambda_1, \lambda_2)/\mathrm{Max}(\lambda_1, \lambda_2)$. In assessing performance, a large number of pairs (λ, ρ) were randomly generated at each of a set of fixed k values in $(0, 1]$, and phase (i) of both TREx and NORTA were executed in MATLAB. The stipulated tolerance was set at 10^{-4}, and the tests were performed on an Intel 1.67 GHz processor. CPU execution times were recorded using the "tic toc" function in MATLAB. All MATLAB code used in the numerical experiments

Table 1 A Brief Comparison of the Generation Phases in the TREx and NORTA Algorithms

(λ_1, λ_2)	ρ	TREx	NORTA	(λ_1, λ_2)	ρ	TREx	NORTA
(1, 1)	0.01	0.96	1.93	(10, 25)	0.01	0.90	1.88
	0.20	0.89	1.85		0.20	1.13	1.88
	0.50	1.11	1.86		0.50	1.13	1.88
	0.90	0.89	1.85		0.90	1.13	1.88
(1, 10)	0.01	0.89	1.86	(10, 100)	0.01	1.02	1.89
	0.20	1.01	1.86		0.20	1.14	1.89
	0.50	1.01	1.86		0.50	1.14	1.89
	0.90	0.89	1.87		0.90	1.14	1.89
(1, 25)	0.01	1.01	1.88	(25, 25)	0.01	0.90	1.88
	0.20	1.01	1.87		0.20	1.12	1.88
	0.50	1.01	1.88		0.50	1.13	1.88
	0.90	1.01	1.91		0.90	1.13	1.88
(1, 100)	0.01	1.02	1.88	(25, 100)	0.01	1.02	1.90
	0.20	1.02	1.88		0.20	1.14	1.90
	0.50	1.02	1.88		0.50	1.14	1.90
	0.90	1.02	1.88		0.90	1.14	1.89
(10, 10)	0.01	0.90	1.88	(100, 100)	0.01	1.12	1.89
	0.20	1.11	1.88		0.20	1.14	1.89
	0.50	1.11	1.88		0.50	1.14	1.92
	0.90	1.11	1.88		0.90	1.13	1.89

Notes. The columns titled "TREx" and "NORTA" show the time, measured in seconds and excluding the preprocessing phase, required to generate 10,000 two-dimensional Poisson random vectors with desired means (λ_1, λ_2) and desired correlation ρ. As can be seen, both algorithms are comparable in terms of generation times, with TREx being marginally faster. Our focus in this paper is more on the preprocessing phases of the two algorithms.

is available for download at https://filebox.vt.edu/users/pasupath/pasupath.htm.

5.1. TREx Preprocessing Times

For assessing the efficiency of the preprocessing phase in TREx, roughly 500,000 pairs (λ, ρ) were generated randomly (uniformly) from the space $(0, 1{,}000] \times (-1, 1)$ for each of the values $k = 0.05, 0.10, \ldots, 1$. At each k value, the preprocessing phase in TREx was then executed, and the recorded CPU times were used to estimate various quantiles. As noted earlier, we do not report generation times here, i.e., the time taken to execute Steps 8 through 20 in the algorithm listing shown in §3.1. Results from the numerical experiments on TREx are depicted in Figure 3, where the 25th, 50th, 75th, and 99th percentiles of execution times are plotted as a function of k.

As can be seen from Figure 3, TREx exhibits uniformly fast preprocessing times. A majority of the generated problems are solved to stipulated tolerance within 8×10^{-4} CPU seconds. Among the roughly 10 million problems that we generated in total, the preprocessing phase for no problem took more than 5×10^{-3} seconds and eight iterations. Figure 3 also suggests that problems that are symmetric in the means of the required marginal distributions, i.e., $k \approx 1$, are somewhat easier to solve. This is because the shape of the correlation function is such that for values of k close to 1, TREx's initial guess detailed in §4.3 turns out to be quite accurate.

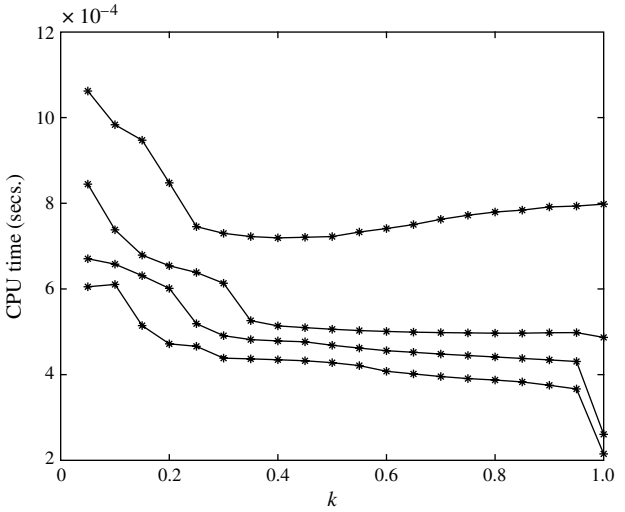

Figure 3 Performance of the Preprocessing Phase in TREx

Notes. The curves show the 25th, 50th, 75th, and 99th percentile preprocessing times estimated by randomly generating five hundred thousand problems for each $k = \text{Min}(\lambda_1, \lambda_2)/\text{Max}(\lambda_1, \lambda_2)$. The reported CPU times were obtained from execution through a MATLAB compiler on an Intel 1.67 GHz processor.

Although we did not generate λ values greater than 1,000, we do not see any reason why the proposed algorithm will not work efficiently for larger λ values. In such cases, however, it is worthwhile investigating whether a normal approximation to the Poisson is a more efficient alternative.

5.2. NORTA Preprocessing Times

As suggested in §1.1.2, the NORTA procedure is very general and works particularly well when the support of the required marginal distributions is continuous (Chen 2001; Cario and Nelson 1997, 1998). However, as elaborated in Avramidis et al. (2009), when the support of one or more of the marginal distributions is denumerable, the root-finding problem (3) becomes nontrivial because of the difficulty in efficiently computing the function $g(\rho^*) = \mathrm{E}[F_1^{-1}(\Phi(Z_1(\rho^*)))F_2^{-1}(\Phi(Z_2(\rho^*)))]$. Avramidis et al. (2009) alleviate this situation by noting that $g(\rho^*)$ and $g'(\rho^*)$ can be computed as

$$g(\rho^*) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \bar{\Phi}_{\rho^*}(z_i, z_j),$$
$$g'(\rho^*) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \phi_{\rho^*}(z_i, z_j), \tag{15}$$

where $\Phi_{\rho^*}(x, y)$ is the bivariate standard normal distribution function and $\bar{\Phi}_{\rho^*}(x, y) = \Phi_{\rho^*}(-x, -y)$, $\phi_{\rho^*}(x, y)$ is the bivariate normal density function with correlation ρ^*, $z_i = \Phi^{-1}(F_1(i))$, $z_j = \Phi^{-1}(F_2(j))$, and $\Phi(x)$ is the standard normal cdf.

Various algorithms are outlined in Avramidis et al. (2009) for solving NORTA's preprocessing phase. We report results from the execution of one of these algorithms—NI3—with which we had the most success. The NI3 algorithm, in essence, is the Newton iteration (Ortega and Rheinboldt 1970):

$$\rho_{n+1} = \rho_n - \frac{f(\rho_n)}{f'(\rho_n)}, \quad f(\rho_n) = \frac{g(\rho_n) - k\lambda^2}{\sqrt{k}\lambda} - \rho, \qquad (16)$$

with appropriate safeguards introduced to ensure that the iterates stay within the stipulated range. We also tried two other algorithms—NI2A and NI2B—outlined in Avramidis et al. (2009), but we had much less success primarily because of difficulties in reliably setting parameters within the embedded numerical integration procedure.

In carrying out the recursion (16), we need to safely approximate $g(\rho^*)$ and $g'(\rho^*)$ by truncating the infinite sums appearing in (15). We used the following result in deciding the number of summands to use in approximating $g(\rho^*)$.

Proposition 11. *Let $\Phi(x)$ denote the univariate standard normal cdf, let $\Phi_{\rho^*}(x, y)$ denote the bivariate standard normal cdf with correlation ρ^*, and let $\bar{\Phi}_{\rho^*}(x, y) = \Phi_{\rho^*}(-x, -y)$. Also, let $z_i = \Phi^{-1}(F_1(i))$ and $z_j = \Phi^{-1}(F_2(j))$, where F_1, F_2 are Poisson cdfs with parameters λ_1, λ_2, respectively. Then,*

(i) *if $\rho^* < 0$,*

$$\left| \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \bar{\Phi}_{\rho^*}(z_i, z_j) - \sum_{i=0}^{N}\sum_{j=0}^{N} \bar{\Phi}_{\rho^*}(z_i, z_j) \right| \leq e^{-(\lambda_1+\lambda_2)} \frac{s^N}{1-s} (e^{e\lambda_1} + e^{e\lambda_2})$$

for all s, N satisfying $1 > s \geq \text{Max}(e\lambda_1/N, e\lambda_2/N)$;

(ii) *if $\rho^* > 0$ and $\lambda_1 \geq \lambda_2$ (without loss of generality),*

$$\left| \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} \bar{\Phi}_{\rho^*}(z_i, z_j) - \sum_{i=0}^{N}\sum_{j=0}^{N} \bar{\Phi}_{\rho^*}(z_i, z_j) \right| \leq 2e^{-N} \left(\left(\frac{1}{\lambda_2}(2N + 3 - \lambda_2)^2 + \ln(4) + 2\lambda_1 + \frac{4}{\lambda_2} \right) e^{2\lambda_2} + (2N+3)e^{2\lambda_1} \right) + e^{\lambda_2} \frac{4}{\lambda_2} e^{-\sqrt{(N-1)\lambda_2 - 2\lambda_1\lambda_2}} \cdot (1 + \sqrt{(N-1)\lambda_2 - 2\lambda_1\lambda_2}).$$

A bound on truncation error has been difficult to establish for $g'(\rho^*)$, although computing $g'(\rho^*)$ accurately is not needed for the Newton iterates in (16) to converge. The bivariate cdf tail appearing as the summand in (15) was calculated using the algorithm by Genz (2004), which appears to be the fastest among those available.

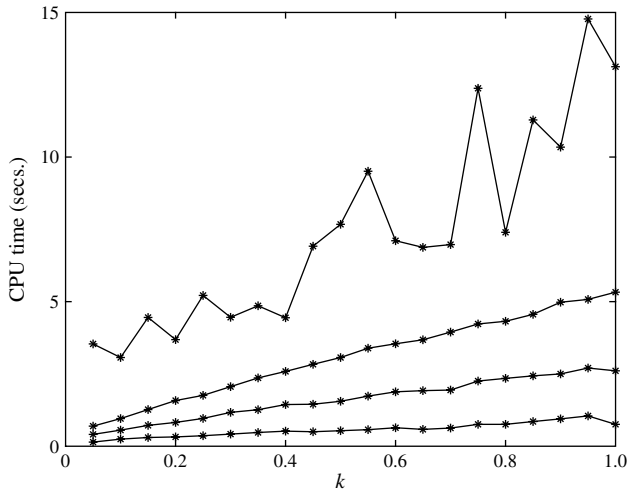

Figure 4 **Performance of the Preprocessing Phase in NORTA Using the NI3 Algorithm in Avramidis et al. (2009)**

Notes. The curves show the 25th, 50th, 75th, and 99th percentile preprocessing times estimated by randomly generating about a thousand problems for each $k = \text{Min}(\lambda_1, \lambda_2)/\text{Max}(\lambda_1, \lambda_2)$. The reported CPU times were obtained from execution through a MATLAB compiler on an Intel 1.67 GHz processor.

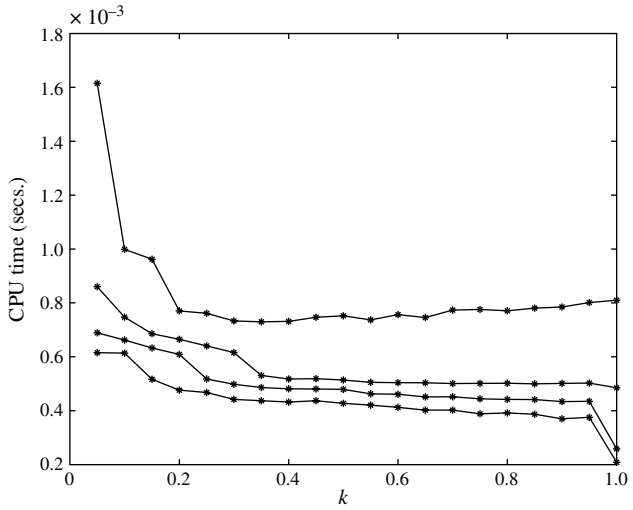

Figure 5 **The 25th, 50th, 75th, and 99th Percentile Preprocessing Times in TREx on the Same Set of Problems Used in Assessing NORTA, and Depicted in Figure 4**

For assessing NORTA's preprocessing step, we generated, for each $k = 0.05, 0.10, \ldots, 1$, more than a thousand pairs (λ, ρ) uniformly drawn from the space $(0, 100] \times (-1, 1)$. The preprocessing step for each of these problems was "solved" using the iteration (16), with $g(\rho^*)$ approximated using the truncation bounds specified by Proposition 11. The execution times, again recorded using the tic toc function in MATLAB, were used to construct the 25th, 50th, 75th, and 99th percentiles as shown in Figure 4. The performance of TREx on the same set of problems in shown is Figure 5.

As can be seen from Figure 4, NORTA performs reasonably well with a majority of the generated problems being solved to stipulated tolerance within 5.0 CPU seconds. Of the roughly 20,000 problems that we generated, the minimum and the maximum time taken by NORTA's preprocessing step were 0.05 and 16.14 seconds, respectively. TREx, however, performed much faster on the same set of problems. It can be seen from Figure 5 that the preprocessing step in TREx is generally between 100 and 1,000 times faster than in NORTA. In fact, we were unable to find even a single problem where TREx's preprocessing step executed slower than that in NORTA.

6. Summary and Concluding Remarks

In this paper, we present a specialized, exact method for generating correlated Poisson random variables. The proposed method TREx, like trivariate reduction, uses extra random variables to induce the required correlation. The extra random variables use common random numbers to induce positive correlation, antithetic variates to induce negative correlation, and have appropriately scaled means identified through a preprocessing step involving a fast one-dimensional recursive search. We also identify rigorous theoretical bounds for truncation error control during implementation.

Unlike trivariate reduction, and like NORTA, TREx has complete coverage in two dimensions. Furthermore, extensive numerical testing reveals that the proposed preprocessing step in TREx, in an overwhelming majority of the cases, takes less than 5×10^{-3} seconds when executed through MATLAB on an Intel 1.67 GHz processor. The corresponding step in a fast implementation of NORTA seems significantly slower, as revealed through experiments on a common set of problems.

It is likely that at least a part of the speed gain that TREx provides over NORTA may be diminished through the use of more efficient numerical quadrature within NORTA. (For instance, the times reported in Avramidis et al. 2009 seem to be faster than those reported here, although the test problems and stopping criteria are different.) The fact remains, however, that evaluating $g(\rho^*)$ in NORTA is expensive—it involves computing a double-infinite sum of bivariate normal tail probabilities, each of whose arguments is a normal inverse of a Poisson cdf. By contrast, the corresponding computation in TREx involves an infinite sum of Poisson cdf inverse products. Computing each of the Poisson inverse products is made very efficient through Kemp's (1988) analytic approximation to the Poisson median probability. The actual generation times in the two methods are comparable.

Two additional remarks relating to future research are now in order.

(i) How should TREx be extended to higher dimensions? As in most random vector generation algorithms, there is a direct extension of TREx from two to higher dimensions by simply breaking the n-dimensional problem into $n(n-1)/2$ two-dimensional problems (Cario and Nelson 1997, Chen 2001). Although simple and direct, it is far from clear that this is the most efficient way of extending TREx to higher dimensions. The primary issue is that, since each of the $n(n-1)/2$ problems "receive" their own extra random variables, there is wastage in terms of computing time and also possible reduction in the coverage area.

(ii) The negative binomial distribution can be characterized as a "Poisson distribution whose parameter λ is Gamma distributed," i.e., as a mixture of the Poisson and Gamma distributions. This close relationship leads to interesting possibilities of developing specialized methods for generating correlated negative binomial random variables, especially by exploiting existing fast methods for Gamma random variate generation. Such methods may be particularly useful, considering that the negative binomial distribution has become increasingly popular as a more flexible alternative to the Poisson distribution (Johnson et al. 1997).

Acknowledgments

The authors thank Bruce Schmeiser, whose original ideas formed the basis of this research. They also thank the anonymous referees, whose comments and suggestions have significantly improved the content and presentation of various parts of this paper. This work was supported in part by the Office of Naval Research through Grant N00014-08-1-0066.

References

Atkinson, A. C. 1979a. Recent developments in the computer generation of Poisson random variables. *Appl. Statist.* **28**(3) 260–263.

Atkinson, A. C. 1979b. The computer generation of Poisson random variables. *Appl. Statist.* **28**(1) 29–35.

Avramidis, A. N., N. Channouf, P. L'Ecuyer. 2009. Efficient correlation matching for fitting discrete multivariate distributions with arbitrary margins and normal-copula dependence. *INFORMS J. Comput.* **21**(1) 88–106.

Bartle, R. G. 1976. *The Elements of Real Analysis*. John Wiley & Sons, New York.

Cario, M. C., 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, IL.

Cario, M. C., B. L. Nelson. 1998. Numerical methods for fitting and simulating autoregressive-to-anything processes. *INFORMS J. Comput.* **10**(1) 72–81.

Chen, H. 1994. Stochastic root finding in system design. Ph.D. thesis, School of Industrial Engineering, Purdue University, West Lafayette, IN.

Chen, H. 2001. Initialization for NORTA: Generation of random vectors with specified marginals and correlations. *INFORMS J. Comput.* **13**(4) 312–331.

Devroye, L. 1986. *Non-Uniform Random Variate Generation*. Springer, New York.

Genz, A. 2004. Numerical computation of rectangular bivariate and trivariate normal t probabilities. *Statist. Comput.* **14**(3) 251–260.

Ghosh, S., S. G. Henderson. 2003. Behavior of the NORTA method for correlated random vector generation as the dimension increases. *ACM TOMACS* **13**(3) 276–294.

Johnson, N. L., A. W. Kemp, S. Kotz. 2005. *Univariate Discrete Distributions*. John Wiley & Sons, New York.

Johnson, N. L., S. Kotz, N. Balakrishnan. 1997. *Discrete Multivariate Distributions*. John Wiley & Sons, New York.

Kemp, A. W. 1988. Simple algorithms for the Poisson modal cumulative probability. *Comm. Statist.: Simulation Comput.* **17**(4) 1495–1508.

Kemp, C. D., A. W. Kemp. 1991. Poisson random variate generation. *Appl. Statist.* **40**(1) 143–158.

Kronmal, R. A., A. V. Peterson Jr. 1979. On the alias method for generating random variables from a discrete distribution. *Amer. Statistician* **33**(4) 214–218.

Mardia, K. V. 1970. *Families of Bivariate Distributions*. Griffin, London.

Ortega, J. M., W. C. Rheinboldt. 1970. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York.

Royden, H. 1988. *Real Analysis*. Prentice Hall, New York.

Schmeiser, B. W., V. Kachitvichyanukul. 1981. Poisson random variate generation. Technical report, School of Industrial Engineering, Purdue University, West Lafayette, IN.

Shin, K., R. Pasupathy. 2007. A method for fast generation of bivariate Poisson random vectors. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, R. R. Barton, eds. *Proc. 2007 Winter Simulation Conf.*, Institute of Electrical and Electronics Engineers, Piscataway, NJ, 472–479.

Whitt, W. 1976. Bivariate distributions with given marginals. *Ann. Statist.* **4**(6) 1280–1289.