NUCLEAR SCIENCE AND ENGINEERING: **173**, 207–221 (2013)

Random Sampling from the Klein-Nishina Distribution: Efficiency, Parsimony, and Speed

Kirk Mathews*

Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433-7765

Received August 12, 2011
Accepted June 6, 2012

Abstract—*Many Monte Carlo photon transport codes draw samples of the scattering cosine, and corresponding energy loss, for Compton scatter from the Klein-Nishina differential scattering cross section. We present new schemes for sampling by rejection that offer advantages in run time, parsimony in use of random numbers, and/or acceptance efficiency. These are compared with Kahn's method, which was recommended by Lux and Koblinger in 1991. We conclude with recommendations for both algorithmic and table-based approaches.*

I. INTRODUCTION

Many Monte Carlo photon transport codes model Compton scattering using the Klein-Nishina (K-N) differential scattering cross section:

$$\frac{d\sigma}{d\mu} = (\alpha_{fsc} r_e)^2 \left(\frac{1}{1+\alpha(1-\mu)}\right)^2 \times \left(1+\alpha(1-\mu)+\frac{1}{1+\alpha(1-\mu)}-(1-\mu^2)\right) , \tag{1}$$

where

$\alpha = E_\gamma / m_e c^2$ = relative energy of the photon

α_{fsc} = fine structure constant

$r_e = \hbar/(m_e c)$ = electron wavelength

$\mu = \cos\theta = \hat{\Omega}' \cdot \hat{\Omega}$ = cosine of the angle θ between the unit-length direction of the motion vector of an incident photon, $\hat{\Omega}$, and that of the photon after scattering, $\hat{\Omega}'$.

The distribution of cross-sectional area over the scattering cosine is specified by the differential scattering cross section; it is the cross-sectional-area density (i.e., per unit μ). This distribution, after normalizing by dividing by the cross-sectional area for all directions σ, forms the probability density function (pdf) for the distribution of the scattering cosine. The K-N cross section can be multiplied by a form factor that accounts for the binding energy of the electrons in atoms, and some codes provide options to do so. We do not address form factors here.

Nearly every function involved in this work has a parametric dependence on α, so it would not be inappropriate to indicate this, for example, as $q(\mu;\alpha)$. However, in the interests of concise notation, we omit this dependence except where it is useful to emphasize it. For convenience, we standardize the differential cross section, defining q such that $q(1) = 2$ for all α, by dropping the leading scaling constants:

$$q(\mu) = \left(\frac{1}{1+\alpha(1-\mu)}\right)^2 \times \left(\frac{1}{1+\alpha(1-\mu)}+\alpha(1-\mu)+\mu^2\right) . \tag{2}$$

The pdf for this distribution is obtained by normalizing q:

$$f_{KN}(\mu) = \frac{q(\mu)}{A(\alpha)} , \tag{3}$$

*E-mail: Kirk.Mathews@afit.edu

where

$$A(\alpha) = \int_{-1}^{1} q(\mu;\alpha)\, d\mu \ . \tag{4}$$

Unfortunately, although $A(\alpha)$ can be obtained in closed form, the formula is badly conditioned for computation with small α,

$$A(\alpha) = \frac{2\alpha(2+\alpha(1+\alpha)(8+\alpha)) - (1+2\alpha)^2(2+(2-\alpha)\alpha)\log(1+2\alpha)}{\alpha^3(1+2\alpha)^2} \ , \tag{5}$$

and would require numerical quadrature. Fortunately, sampling by rejection is independent of scaling; as will be seen, $A(\alpha)$ cancels out.

The cumulative distribution function (CDF) for the distribution of μ is $F_{KN}(\mu) = \int_{-1}^{\mu} f_{KN}(\mu')\, d\mu'$. Although it is available in closed form, this CDF is as badly behaved as $A(\alpha)$, or worse. Furthermore, it is not invertible without numerical root solving. (Gerts[1] used inversion by nonlinear root solving in Mathematica. His experience is that "it is quite inefficient, though.") Therefore, we choose rejection techniques to avoid requiring either f_{KN} or F_{KN}; we seek to use only q.

The ratio of energies, before scatter to after scatter, is

$$x \equiv \frac{E_\gamma}{E'_\gamma} = \frac{\alpha}{\alpha'} = 1 + \alpha(1-\mu) \ . \tag{6}$$

(Some authors use x as the reciprocal ratio, for which we use y.) The differential cross section can be expressed in terms of this variable. We standardize the distribution of x by multiplying q by α^2 (to remove singularities), substituting x, and rearranging to obtain

$$\frac{d\sigma}{dx} \propto Q(x) = 1 + \frac{\alpha^2 - 2\alpha - 2}{x} + \frac{x_{\max}}{x^2} + \frac{\alpha^2}{x^3} \ , \tag{7}$$

where $x_{\max} = 1 + 2\alpha$. Some methods found in the literature sample x or $y = 1/x$, which then determines μ. Sampling x or y, however, results in bad conditioning in the computation $\mu = 1 - (x-1)/\alpha$ or $\mu = 1 - (1-y)/\alpha y$ for small α. However, Eq. (6) is well-conditioned for computing x from μ with any positive α. Therefore, we choose to sample for μ.

Blomquist and Gelbard[2] assess several methods, including isotropic sampling and rejection and Horowitz et al.'s linear rejection function[3]—both for sampling from the distribution of α'/α. These have efficiencies of 2/3 at best. Blomquist and Gelbard reject tabular methods and various approximation schemes and instead recommend Koblinger's direct random-term sampling scheme,[4] which uses two random numbers per sample, for $\alpha \geq 1 + \sqrt{3}$, combined with Kahn's long-overlooked random-term sampling and rejection scheme,[5] which uses three random numbers per attempt, for $\alpha < 1 + \sqrt{3}$. Rather than using two-dimensional tables of piecewise-linear approximations to the CDF, Lux and Koblinger recommend this same combination.[6] Following these recommendations, we do not consider approximate methods; however, we do consider a tabular rejection scheme that is both fast and exact. We refer to methods as either algorithmic or tabular.

One thing that these previous approaches have in common is that they all sample an energy ratio. To explore new ground, we developed methods for sampling from the distribution of the cosine of the photon scattering angle μ. Not only does this avoid loss of precision, but also the distribution in μ has different shapes than the distribution in x, offering new opportunities for rejection sampling functions.

The optimization of numeric processors within current microcomputer CPUs has changed the relative costs of various arithmetic operations and of Fortran intrinsic functions. Therefore, we explore methods that use exponentials, logarithms, hyperbolic functions, and so on, as well as strictly algebraic functions. Similarly, the presence of substantial on-chip caches should make tabular approaches more effective than in the past. We evaluate each method on the basis of three goals that often conflict:

1. efficiency: acceptance efficiency of rejection schemes
2. parsimony: use of fewer random numbers per sample
3. speed: random samples per unit CPU time.

II. SAMPLING TECHNIQUES

In this section, we review a variety of sampling techniques in the context of the K-N distribution.

II.A. Direct Sampling

II.A.1. Direct Inversion of the CDF

Consider a pdf $\varphi(\mu)$ with CDF $\Phi(\mu)$, which has the inverse function Φ^{-1}. Sampling by direct inversion of the CDF uses one random (pseudorandom) number ξ

sampled from a uniform distribution on the interval [0,1], to generate a sample

$$\mu = \Phi^{-1}(\xi) \tag{8}$$

from the pdf φ. In modern Fortran, the algorithm is simply

```
Call Random_Number(xi)
mu = PhiInverse(xi) .
```

II.A.2. Inversion of the CDF by Root Solving

If the CDF is not invertible in closed form, it can be inverted by numerically root solving

$$\Phi(\mu) = \xi \tag{9}$$

for μ. Presuming that $\varphi(\mu) > 0$ for $\mu \in [-1,1]$, which is the case for the distributions used here, $\Phi(\mu)$ is strictly monotone increasing so that the root-solving problem is well behaved.

II.A.3. Linear Inverse Interpolation of the CDF

If the CDF is not available in closed form, it can be obtained by numerical quadrature of the pdf. However, the combination of numerical quadrature and numerical root solving is typically prohibitively expensive in computational cost. As an approximate sampling scheme, a table of CDF values could be obtained by numerical quadrature. Then, the inverse CDF would be approximated by inverse linear interpolation. Efficiency is enhanced if the points are precomputed and are uniformly spaced in Φ, rather than in μ. This eliminates the need to search for the segment of Φ that contains ξ. This approach is equivalent to a piecewise-constant approximation of φ. Unfortunately, not only are interpolation methods approximations, they are also computationally expensive when the table entries cannot be precomputed because they are functions of another parameter, to wit, α in the case of the K-N distribution.

II.A.4. Random-Term Technique

The pdf of a variable that is conditioned upon which of several possible events occurs is

$$f(x) = \sum_{i=1}^{n} f(x|\text{event}_i)P(\text{event}_i) \ . \tag{10}$$

A sample could be drawn by inverting $F(x) = \xi$, but it may be easier to first select a random event, as weighted by their probabilities, and then invert the conditional CDF, i.e., solve $F(x|\text{event}_i) = \xi$. Sampling from a function that is a sum of terms, $g(u) = \sum_{i=1}^{n} g_i(u)$, can be done in this way by casting it in this form by normalizing each term individually:

$$f_g(u) = \sum_{i=1}^{n} \left(\frac{g_i(u)}{\int_{-\infty}^{\infty} g_i(u)\,du} \right) \left(\frac{\int_{-\infty}^{\infty} g_i(u)\,du}{\sum_{i=1}^{n} \int_{-\infty}^{\infty} g_i(u)\,du} \right) . \tag{11}$$

Koblinger called this a mixed probabilities method and applied it to the four terms of $Q(x)$ in Eq. (7). In that case, rejection is not needed, but all the terms must be nonnegative, which requires that $\alpha \geq 1 + \sqrt{3}$. This scheme uses two random numbers per sample: one to choose the term, and one to draw the sample from that term.

II.B. Rejection Methods

Perhaps the most intuitive form of sampling by rejection is sampling by geometric rejection. For example, to choose random points in France, uniformly distributed by area, take an area-preserving map of the world, choose random points uniformly distributed over its surface, and disregard (reject) those that are not in France, but use (accept) those that are in France. Because the area of France is small compared to that of the Earth, relatively few points will be accepted. Note that the scale of the map is irrelevant. Large or small, the same fraction of samples drawn will be accepted. (This is usually called the efficiency of the rejection method. We call it acceptance efficiency and use the symbol η.) The efficiency cannot be improved by scaling but can be improved by trimming area out of the rejection region. Thus, it would be more efficient to use a map of Europe than one of the world. On the other hand, the sampling region must include the entire acceptance region.

II.B.1. Geometric Rejection Applied to a Distribution Function

Lewis and Miller[7] describe rejection as follows (paraphrased and with some notation changes): Let $0 \leq f(x) \leq f_{\max}$ be the pdf of x in the domain $a \leq x \leq b$. Rescale both to fit in a unit square: Let $u = (x - a)/(b - a)$ and $\tilde{f}(u) = f(x(u))/f_{\max}$, where $x(u) = a + (b - a)u$. Then, $0 \leq u \leq 1$ and $0 \leq \tilde{f} \leq 1$. To choose a random point uniformly distributed by area within the square, let u and v be random numbers uniformly distributed in $(0,1)$, as are provided by two draws from a typical pseudorandom number generator. Accept the point if $v \leq \tilde{f}(u)$.

Applying the unit-square approach to the K-N distribution,

$$\tilde{f}(u) = \frac{f(\mu(u))}{f_{\max}} = \frac{q(\mu(u))/A(\alpha)}{q_{\max}/A(\alpha)} = \frac{q(\mu(u))}{q_{\max}} , \quad (12)$$

where

$$\mu(u) = -1 + 2u . \quad (13)$$

Thus, $A(\alpha)$ is not needed, and because of the standardization of q, $q_{\max} = q(1) = 2$, independent of α.

The first step of generalization is to use the natural scale of q, rather than rescaling to the unit square. Let `Random()` be a pseudorandom generator function. It returns a new random number from the uniform distribution over the interval [0,1) each time it is used. The following algorithm is equivalent:

```
qMax = 2.
Do
  mu = 2. * Random() - 1.
  If ( qMax * Random() < = q(mu) ) Exit
    !  Accept mu
 End do .
```

This method uses $q_{\max}$ as the upper boundary of the rejection region because it is easy to draw a uniformly distributed sample and test for acceptance. (This is an example of a constant sampling function, presented in Sec. III.A.1.) It is very fast per attempt. However, this naïve approach can be inefficient; as shown in Fig. 1 for $\alpha = 1$, acceptance efficiency is only 28.7%.

This approach is equivalent to sampling μ from the pdf $\varphi(\mu)$, obtained by normalizing the sampling function $s(\mu) = q_{\max}$,

$$\varphi(\mu) = \frac{s(\mu)}{\int_{-1}^{1} s(\mu)\, d\mu} = \frac{2}{\int_{-1}^{1} 2\, d\mu} = \frac{1}{2} , \quad (14)$$

integrating it to obtain the CDF,

$$\Phi(\mu) = \int_{-1}^{\mu} \frac{1}{2}\, d\mu = \frac{\mu + 1}{2} , \quad (15)$$

setting it equal to ξ, and solving for μ,

$$\mu = 2\xi - 1 . \quad (16)$$

Note that this sampling function is an upper bound on q: $q_{\max} = s(\mu) \geq q(\mu)$.

II.B.2. Generalized Rejection

The efficiency of sampling by rejection (perhaps it should be called sampling by acceptance) can now be improved by choosing a sampling function $s(\mu)$ that trims away some of the rejection region while leaving all the acceptance region present, just as the map of Europe trimmed rejection area from the map of the world but did not remove any of France. As an example, consider a straight line between $(-1,q(-1))$ and $(+1,q(+1))$, as shown in Fig. 2. (This is an example of a linear sampling function, presented in Sec. III.A.2.) The efficiency is improved to 48.5% for this value of α.

Thus, the methodology for developing a scheme for sampling by rejection from a positive distribution is as follows:

1. Choose a sampling function s with the property

$$s(\mu) \geq q(\mu) > 0 \quad \text{for } -1 \leq \mu \leq 1 , \quad (17)$$

and such that it is easy to sample μ from the pdf $\varphi(\mu)$ that is obtained by normalizing $s(\mu)$.

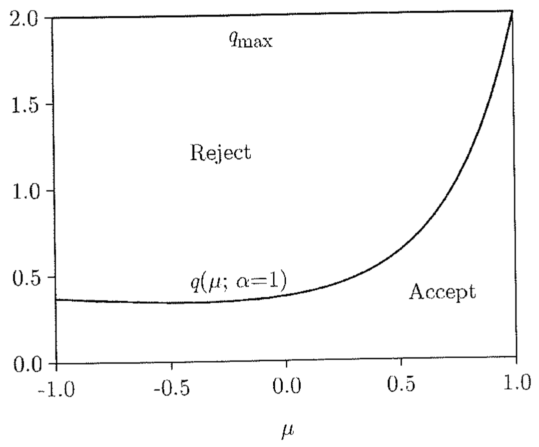

Fig. 1. Geometric rejection applied to $q(\mu;\alpha = 1)$.

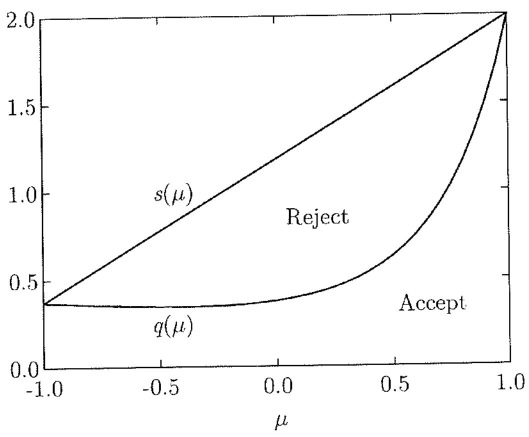

Fig. 2. Improved rejection sampling for $\alpha = 1$.

2. Sample μ from the pdf $\varphi(\mu)$.

3. Then, accept the value of μ thus obtained with probability $0 < q(\mu)/s(\mu) \leq 1$ (or go back to step 2 if sample is rejected).

A Fortran-like pseudocode for this algorithm is

```
Do
  mu = SampleFromS(Random())
  ! method depends on choice of sampling
    function s(mu)
  If (s(mu) * Random() < = q(mu)) Exit
         !  Accept mu.
 End do  ! Otherwise, try again .
```

II.B.3. Acceptance Efficiency, Usage of Random Numbers, and Cost

Denoting the probability of an event as P(event description), the probability that a value of μ in $d\mu$ at μ will be sampled and accepted in a single trial, $p_1(\mu)\,d\mu$, is the product of (a) the probability that a value of μ in $d\mu$ at μ will be sampled,

$$P(\text{draw } \mu \in [\mu, \mu + d\mu]) = \frac{s(\mu)\,d\mu}{\int_{-1}^{1} s(\mu')\,d\mu'} = \varphi(\mu)\,d\mu \ , \tag{18}$$

and (b) the conditional probability that it will then be accepted,

$$P(\text{accept } \mu | \mu \text{ was drawn}) = \frac{q(\mu)}{s(\mu)} \ . \tag{19}$$

Note that the requirements that $s(\mu) \geq q(\mu)$ [Eq. (17)] and $q(\mu) \geq 0$, which is implicit in that q is proportional to a pdf, are together both necessary and sufficient to ensure that $0 \leq P(\text{accept } \mu | \mu \text{ was drawn}) \leq 1$.

Thus,

$$p_1(\mu)\,d\mu = \frac{s(\mu)\,d\mu}{\int_{-1}^{1} s(\mu')\,d\mu'} \frac{q(\mu)}{s(\mu)} = \frac{q(\mu)\,d\mu}{\int_{-1}^{1} s(\mu')\,d\mu'} \ . \tag{20}$$

The sampling efficiency η is the probability of accepting a value (any value) of μ in one trial:

$$\eta = \int_{-1}^{1} p_1(\mu)\,d\mu = \frac{\int_{-1}^{1} q(\mu)\,d\mu}{\int_{-1}^{1} s(\mu)\,d\mu} \ . \tag{21}$$

This is simply the area under q divided by the area under s, just as the efficiency of the map technique for sampling a point in France is the area of France divided by the area of the region covered by the map.

The expected number of trials used to get an acceptance is

$$n_{trials}(\eta) = \sum_{j=1}^{\infty} (j(1-\eta)^{j-1}\eta) = \frac{1}{\eta} \ . \tag{22}$$

The expected number of random numbers used to obtain a sample by rejection is thus

$$n_{randoms}(\eta) = \frac{n_{randoms\ per\ trial}}{\eta} \ , \tag{23}$$

where the number of random numbers used in one trial, $n_{randoms\ per\ trial}$, is determined by the algorithm (two or three for most rejection schemes).

A scheme is efficient (high acceptance probability) if s does not exceed q by much and parsimonious if it is efficient and uses few random numbers per trial, while the computational cost is the product of the expected number of trials and the computational cost per trial.

Next, we prove that this rejection approach does provide the correct distribution of sampled direction cosines. The probability that a value of μ in $d\mu$ will be sampled and accepted on the k'th attempt is the product of the probability that each prior trial results in rejection, $(1-\eta)^{k-1}$, and the probability that such a value will be accepted in one trial (the k'th trial). The probability that such a value will be accepted at any trial (before some other value is accepted) is the sum over all possible numbers of trials:

$$\begin{aligned} &P(\text{accept } \mu \in [\mu, \mu + d\mu]) \\ &= \sum_{k=1}^{\infty} (1-\eta)^{k-1} p_1(\mu)\,d\mu \\ &= [p_1(\mu)\,d\mu] \sum_{j=0}^{\infty} (1-\eta)^j = \left[\frac{q(\mu)\,d\mu}{\int_{-1}^{1} s(\mu')\,d\mu'} \right] \frac{1}{\eta} \\ &= \frac{q(\mu)\,d\mu}{\int_{-1}^{1} s(\mu')\,d\mu'} \frac{1}{\left[\frac{\int_{-1}^{1} q(\mu)\,d\mu}{\int_{-1}^{1} s(\mu)\,d\mu} \right]} \\ &= \frac{q(\mu)\,d\mu}{\int_{-1}^{1} q(\mu')\,d\mu'} = f_{KN}(\mu)\,d\mu \qquad \text{Q.E.D} \ . \end{aligned} \tag{24}$$

Finally, there is the question of how to draw a sample from $s(\mu)$. Ideally, $s(\mu)$ is readily normalized to form the pdf $\varphi(\mu)$, which is readily integrated to form the CDF $\Phi(\mu)$:

$$\Phi(\mu) = \int_{-1}^{\mu} \varphi(\mu')\, d\mu' \ . \tag{25}$$

Then, μ is obtained by drawing a uniform pseudorandom number $\xi \in [0,1]$ and solving $\Phi(\mu) = \xi$ for μ. Again ideally, this solution would be obtainable in closed form such that $\mu = \Phi^{-1}(\xi)$. To be usable, the resulting formulas must be numerically well-conditioned, e.g., without excessive loss of precision in subtractions. In Sec. III, we examine several choices of sampling function $s(\mu)$. Some are ideal, as described here. However, it may be that greater acceptance efficiency (using an s that is a tighter bound on q) offsets a more complicated or expensive process for sampling from that s. Therefore, we first review other schemes for sampling.

II.C. Combining Rejection and Random-Term Techniques

Two ways to combine rejection and random-term techniques consist of choosing one technique for sampling and the other technique for inversion of the CDF.

II.C.1. Rejection Using Random-Term Sampling

Consider a function from which to sample, for example, $q(\mu)$, and a rejection sampling function, $s(\mu)$ that is a sum of positive terms, $s_i(\mu) \geq 0$ such that

$$s(\mu) = \sum_{i=1}^{n} s_i(\mu) \geq q(\mu) \ . \tag{26}$$

Rather than inverting the CDF created from s, we can use the random-term technique to draw a sample from s by using s as g in Eq. (11). Let

$$S_i = \int_{-1}^{1} s_i(\mu)\, d\mu \ ; \tag{27}$$

then, the probabilities used to select a term are

$$P_i = S_i \bigg/ \sum_{j=1}^{n} S_j \ . \tag{28}$$

This uses two random numbers: one to select the term, and one to sample the term. Then, the sample is tested for acceptance using a third random number, ξ_3:

$$\begin{array}{ll} \text{Accept} & \xi_3 \leq q(\mu)/s(\mu) \\ \text{Reject} & \text{else} \ . \end{array} \tag{29}$$

Rejection is being used as an alternative to inversion of the CDF, and the sample is drawn from the sampling function by random-term sampling. Analytic inversion of the CDF for the selected term is desirable, so each $s_i(\mu)$ should be amenable to that. Failing that, it may still be efficient if the terms that require more expensive sampling methods have small enough probabilities P_i.

If the sample is rejected, one must draw three new random numbers and repeat the whole procedure until a sample is accepted.

II.C.2. Random-Term Sampling by Rejection

Rejection with random-term sampling does not require q to be a sum of terms, and it does not require any integration of q. However, if q can be partitioned into terms, and if it is practical to integrate each over its domain, then the nesting of methods can be reversed to be random-term sampling by rejection. This does not require that it be practical to construct and directly invert the CDF for each term. This approach is more parsimonious in the use of random numbers, because the term to be sampled is randomly selected only once; hence, this need not be drawn again after a rejection.

Consider a function from which to sample, for example $q(\mu)$, that is a sum of positive terms:

$$q(\mu) = \sum_{i=1}^{n} q_i(\mu) \ . \tag{30}$$

Use q as g in Eq. (11). For this method, the probabilities for the terms are based on q, not s. Some of the terms may produce CDFs that are usefully inverted analytically. Those that are not can be sampled by rejection: Choose sampling functions such that

$$s_i(\mu) \geq q_i(\mu) \geq 0 \ , \tag{31}$$

each of which $s_i(\mu)$ can be sampled by inverting the CDF created from it.

The process has three steps: (a) choose the random term (using one random number), (b) sample from its sampling function by analytic inversion (using a second random number), and (c) test for acceptance (using a third random number):

$$\begin{array}{ll} \text{Accept} & \xi_3 \leq q_i(\mu)/s_i(\mu) \\ \text{Reject} & \text{else} \ . \end{array} \tag{32}$$

After a rejection, it is only necessary to return to step (b), rather than step (a). This can save some random numbers.

If precision is lost in computing $q(\mu)$, this technique may allow an algebraic reformulation from which it may be partitioned into terms that do not (individually) have this difficulty. Then, only one of the $q_i(\mu)$ need be computed. The technique may also be more efficient because only one term of q and one term of s need be evaluated.

II.D. Tabular Rejection

Linear inverse interpolation is an approximate technique in which the actual pdf is approximated as being piecewise constant. A piecewise-linear CDF yields its piecewise-constant pdf by differentiation. However, a sampling function for rejection sampling of a given pdf is readily constructed by partitioning the domain of the pdf (as conveniently rescaled, i.e., q) into intervals that tile the domain and by setting the sampling function in each interval to equal the maximum value of the pdf within that interval. Actually, s can be chosen to exceed the maximum value in the interval, with a loss of rejection efficiency, if need be. It is this property that makes a tabular method practical for the K-N distribution.

This approach is exact because the rejection test uses q, not an approximation to it. Construction of the partition of the domain so that each interval of s is equally likely to contain the random ξ enhances efficiency by eliminating the need to search for the interval that contains ξ.

II.E. Properties of $q(\mu)$

The shape of $q(\mu)$ varies dramatically with α, as shown in Fig. 3. For $\alpha \geq 1 + \sqrt{3}$, direct sampling of q is possible using Koblinger's method. The curve for 2.74, which is slightly greater than $1 + \sqrt{3}$, shows the shape at this juncture.

For $\alpha = 0$, $q(\mu) = 1 + \mu^2$, and a quadratic upper bound is efficient for small α. A hyperbolic cosine has a similar shape and is also efficient for some α. For larger values, an inverse linear function with a pole to the right of the diagram captures the shape of q rather well. An exponential curve that fits q at each end was also examined.

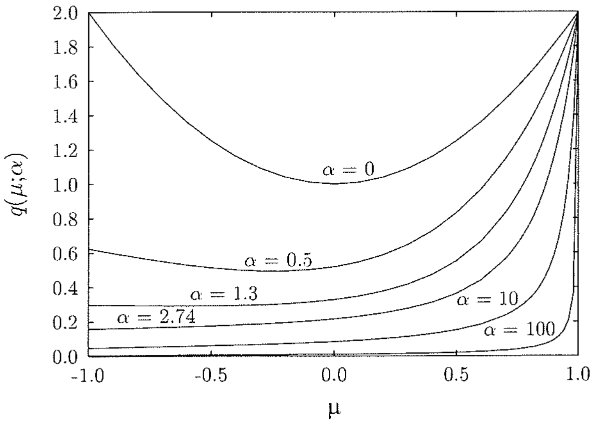

Fig. 3. $q(\mu;\alpha)$ for various energy ratios.

Particular values are needed in fitting some of the sampling functions: the values for forward and backward scattering,

$$q_F = q(+1) = 2$$

and

$$q_B = q(-1) = \frac{2 + 4\alpha + 4\alpha^2}{(1 + 2\alpha)^3} \ , \tag{33}$$

and at the minimum, $q_{\min} = q(\mu_{\min})$, where

$$\mu_{\min} = \frac{(-\alpha)(4 + 8\alpha + 2\alpha^2 + 4\alpha^3)}{(2 + 8\alpha + 10\alpha^2 + 2\alpha^3 - 4\alpha^4) + (2 + 4\alpha)\sqrt{1 + 4\alpha + 10\alpha^2 + 6\alpha^3 - 3\alpha^4}} \ . \tag{34}$$

Equation (34) has been algebraically manipulated into a form that has negligible loss of precision from subtractions within this range. Another useful value is α_B, the value of α for which $\mu_{\min} = -1$:

$$\alpha_B = \frac{1}{3} + \frac{8}{3\sqrt[3]{98 + 18\sqrt{17}}} + \frac{\sqrt[3]{98 + 18\sqrt{17}}}{6} \approx 1.73991 \ . \tag{35}$$

The formula for $q(\mu_{\min})$ does not simplify in any helpful way. The slopes of $q(\mu)$ at the ends of the curve are also useful; they simplify to

$$q'_F = \left. \frac{d}{d\mu} q(\mu) \right|_{\mu=1} = 2 + 4\alpha \tag{36}$$

and

$$q'_B = \frac{-2 - 4(\alpha + \alpha^2 - \alpha^3)}{(1 + 2\alpha)^2} \ . \tag{37}$$

III. SAMPLING FUNCTIONS

We considered a wide variety of possible sampling functions. A few are presented because they are particularly simple or obvious. Others are of interest because of their performance in at least one of the objectives in at least some range of α.

III.A. Polynomial Sampling Functions

Polynomials of low order are easy to construct and integrate to form their CDFs. For low-enough order, these can be inverted in closed form, or in any case by random-term methods.

III.A.1. Constant Sampling Function

With a polynomial of order zero, all μ are equally likely. Thus, this scheme is also known as isotropic sampling with rejection. The tightest bound is $s(\mu) = q_F = 2$. This yields isotropic sampling: $\mu = 2\xi_1 - 1$; acceptance is per Eq. (32). This is fast per trial but inefficient except for very small α.

III.A.2. Linear Sampling Function

The tightest order 1 polynomial bound interpolates between the endpoints of q:

$$s_{Linear}(\mu) = \frac{q_F + q_B}{2} + \frac{q_F - q_B}{2}\mu \ . \tag{38}$$

Integrating produces a quadratic CDF, which yields the following computationally well-conditioned formula (the subtraction in the numerator being benign):

$$\mu = \frac{4(q_F + q_B)\xi_1 - q_F - 3q_B}{q_F + q_B + 2\sqrt{q_F^2\xi_1 + q_B^2(1-\xi_1)}} \ , \tag{39}$$

and acceptance is per Eq. (32). This is somewhat more efficient than isotropic sampling and rejection but also slower to compute.

III.A.3. Quadratic Sampling Function

At $\alpha = 0$, q is the simple quadratic $1 + \mu^2$. So, at least for small α, a quadratic sampling function should be very efficient. The tightest bounding quadratic sampling function is constructed to osculate q at its minimum and to match q at $\mu = 1$:

$$s_{quadratic}(w) = q_{\min} + (2 - q_{\min})w^2 \ ,$$

$$\text{where} \quad w = \frac{\mu - \mu_{\min}}{1 - \mu_{\min}}$$

$$\text{and} \quad w_B = \frac{-1 - \mu_{\min}}{1 - \mu_{\min}} \le w \le 1 \ . \tag{40}$$

We tested three techniques for inverting the CDF of w to draw a sample: root solving with Newton's method, the analytic solution, and random-term sampling (labeled "quadratic two terms" in Sec. V). Newton's method is straightforward but slow compared to the other methods.

III.A.4. Quadratic Rejection Using Analytic Inversion

The sampling equation $F_{quadratic}(w) = \xi$ can be written as

$$w^3 + 3\beta w = 2\gamma \ ,$$

$$\text{where} \quad \beta = \frac{q_{\min}}{2 - q_{\min}}$$

$$\text{and} \quad \gamma = \frac{\xi_1(1 + 3\beta) + (1 - \xi_1)w_B(w_B^2 + 3\beta)}{2} \ . \tag{41}$$

In this form, the analytic solution is elegant, even in the following form, which is well-conditioned for computation (for either sign of γ) because $0 < \beta \le \frac{1}{2}$:

$$w = \left(z - \frac{\beta}{z}\right)\operatorname{sgn}(\gamma) \ ,$$

$$\text{where} \quad z = \sqrt[3]{|\gamma| + \sqrt{\beta^3 + \gamma^2}} \ . \tag{42}$$

Then, $\mu = \mu_{\min} + w(1 - \mu_{\min})$ is acceptance tested as per Eq. (32).

III.A.5. Quadratic Rejection Using Random-Term Sampling

Applying the random-term sampling method to the constant and w^2 terms in $s_{quadratic}(\mu)$ yields

$$\mu = \begin{cases} 2\xi_2 - 1 & \xi_1 \le \left[1 + \dfrac{(2 - q_{\min})(1 - w_B^3)}{3q_{\min}(1 - w_B)}\right]^{-1} \\ \mu_{\min} + (1 - \mu_{\min})[\xi_2 + (1 - \xi_2)w_B^3]^{1/3} & \text{Else} \ , \end{cases} \tag{43}$$

and ξ_3 is used to test for rejection or acceptance as in Eq. (32).

III.B. Inverse Polynomial Rejection

For $\alpha > \alpha_B$, the shape of q suggests an inverse power with the pole at some location $\mu > 1$. Therefore, we consider inverse powers with a shifted origin. Inverse linear and inverse square functions were constructed to satisfy the constraints of Eq. (17).

III.B.1. Inverse Linear Sampling Function

The inverse linear sampling function is fitted to q by matching its value at $\mu = -1$ and its slope at $\mu = 1$:

$$s_{IL}(\mu) = \left(\frac{b+2}{b+1-\mu}\right) q_B \ , \tag{44}$$

where

$$b = \frac{1+\sqrt{1+8q_F'/q_B}}{2q_F'/q_B} \ . \tag{45}$$

The inverse linear sampling function is readily integrated to obtain its CDF, which is inverted in closed form to sample μ:

$$\mu = 1 - b\left[\left(1+\frac{2}{b}\right)^{\xi_1} - 1\right] \ . \tag{46}$$

Then, the test for acceptance uses ξ_2. This method can be applied for any $\alpha \geq 0$.

III.B.2. Inverse Square

Kahn's method obtains simplicity of the sampling and rejection functions at the expense of modest efficiency and the use of random-term sampling with rejection. It yields speed at the cost of high usage of random numbers. The inverse square sampling function obtains equivalent simplicity without needing random-term sampling. The sampling function is

$$s_{IS}(\nu) = \frac{1}{a(2+\alpha\nu)^2} \ , \tag{47}$$

where $\nu = 1 - \mu$ is always positive and

$$a = \left(\frac{2}{16+\alpha}\right) \tag{48}$$

ensures that the constraints of Eq. (17) are satisfied for $0 \leq \alpha \leq 1 + \sqrt{3}$. Normalizing and integrating produces an invertible CDF, yielding

$$\nu = \frac{2\xi_1}{1+\alpha(1-\xi_1)} \ . \tag{49}$$

For this sampling function, the test for acceptance in Eq. (32) simplifies to

$$\xi_2 \leq p = a(y+1)^2(b+y+\mu^2) \ , \tag{50}$$

where

$$b = \alpha\nu \ ; \quad x = 1+b \ ; \quad y = 1/x \ ; \quad \mu = 1-\nu \ . \tag{51}$$

After acceptance, the energy is reduced by $\alpha = \alpha y$.

Like Kahn's method, the simplicity of this method makes it fast enough per sampling attempt to compensate for its moderate efficiency. However, it is more parsimonious than Kahn's method because it uses only two random numbers per attempt, rather than the three per attempt required by Kahn's method.

III.C. Transcendental Sampling Functions

In the testing done by Bloomquist and Gelbard, transcendental functions were so slow to compute on the hardware of the day (CDC 7600, IBM 370/195, and IBM 3033) that methods using them were not presented. However, with current microprocessors, things may have changed. The issue is whether increased efficiency and the corresponding parsimony can compensate for run time per attempt or justify modest increases in run time. An exponential interpolation between q_B and q_F was simple but lacked efficiency. Attempts to combine two exponentials led to a bound using the hyperbolic cosine.

III.C.1. Exponential

The exponential function

$$s_{\exp}(\mu) = be^{\mu\log(c)} = bc^{\mu} \ , \tag{52}$$

where $b = \sqrt{q_F q_B}$ and $c = \sqrt{q_F/q_B}$, matches q at both ends and is concave up. This improves efficiency compared to linear interpolation. It normalizes and integrates to form an invertible CDF, yielding

$$\mu = \frac{\log[(1-\xi_1)/c + \xi_1 c]}{\log(c)} \ . \tag{53}$$

Then, ξ_2 is used to test for acceptance as per Eq. (32).

III.C.2. Hyperbolic Cosine

The hyperbolic sampling function

$$s_{\cosh}(\mu) = q_{\min}\cosh\left(\frac{\mu-\mu_{\min}}{1-\mu_{\min}}\beta\right) \ ,$$

$$\text{where} \quad \beta = \cosh^{-1}(q_F/q_{\min}) \ , \tag{54}$$

meets the same conditions as the quadratic. It is usable only for α in the range $0.20285 \le \alpha < 1 + \sqrt{3}$, in which it is both defined and an upper bound to q. It integrates in closed form, yielding an invertible CDF. Thus,

$$\mu = \mu_{\min} + \frac{(1-\mu_{\min})}{\beta} \sinh^{-1}\left(\xi_1 \sinh(\beta) + (1-\xi_1)\sinh\left[\left(\frac{-1-\mu_{\min}}{+1-\mu_{\min}}\right)\beta\right]\right) . \tag{55}$$

Then, ξ_2 is used to test for acceptance. Our testing used the intrinsic hyperbolic functions in Intel Fortran 95 and reduced cost using

$$\sinh(\beta) = [(q_F/q_{\min})^2 - 1]^{1/2} . \tag{56}$$

III.D. Rejection Using the Positive Terms of $Q(x)$

Three terms in $Q(x)$ are positive for all $\alpha \ge 0$. Omitting the other term when it is negative provides an upper bound sampling function for rejection:

$$s_{three\ terms}(x) = 1 + \frac{x_{\max}}{x^2} + \frac{\alpha^2}{x^3} . \tag{57}$$

The remaining term is negative for $\alpha < 1 + \sqrt{3}$ but is zero at that value. Thus, this method approaches 100% efficiency as $\alpha \to 1 + \sqrt{3}$. Because most of the other methods decrease in efficiency as α increases, this scheme may have utility for at least some range of α.

With this rejection sampling function, x can be obtained by random-term sampling. However, $\mu = 1 - (x-1)/\alpha$ is badly conditioned for small α; therefore, we reduce the formulas for x algebraically to well-conditioned formulas for μ:

$$\mu = \begin{cases} 1 - \dfrac{(1-\xi_2)b}{1 + \alpha\xi_2 b + x_{\max}\sqrt{1+\alpha\xi_2 b}} \\ \qquad 0 \le \xi_1 < p_1 = c_1/c_2 \\ \mu = \dfrac{2\xi_2(1+\alpha)-1}{1+2\xi_2\alpha} \\ \qquad p_1 \le \xi_1 < p_2 = p_1 + \dfrac{x_{\max}^2}{c_2} \\ \mu = 2\xi_2 - 1 \quad p_2 \le \xi_1 \le 1 , \end{cases} \tag{58}$$

where

$$\begin{aligned} x_{\max} &= 1 + 2\alpha ; \\ b &= 4(1+\alpha) ; \\ c_1 &= \alpha^2(1+\alpha) ; \\ c_2 &= c_1 + 2x_{\max}^2 . \end{aligned} \tag{59}$$

Then, x is computed from μ, and both are accepted if $\xi_3 \le Q(x)/s_{three\ terms}(x)$.

IV. TABULAR METHODS: STEP SAMPLING WITH REJECTION

For any given $\mu < 2$, $q(\mu;\alpha)$ is a monotone-decreasing function of α for all $\alpha \ge 0$. Therefore, a step (piecewise-constant) sampling function $s_k(\mu)$ constructed for a particular value α_k is usable for any $\alpha \ge \alpha_k$, but with monotone-decreasing acceptance efficiency. With an appropriate set of such sampling functions for $k = 1, 2, \ldots, K$, choose k such that $\alpha_k \le \alpha < \alpha_{k+1}$, where $\alpha_{K+1} = 1 + \sqrt{3}$. The challenges are to construct the set of α_k and the corresponding steps in μ and sample function values for each step in such a way as to make the scheme fast and efficient for drawing samples. (The cost of creating these data tables is not of concern.)

IV.A. Constructing a Step-Sampling Function for Given α

For a specified table value α_k, we partition the domain of μ into N equally likely intervals in the following way. The end values bound the range of μ, so $\mu_{0,k} = -1$ and $\mu_{N,k} = +1$. The solution proceeds iteratively starting with the approximation

$$A_k \approx \int_{-1}^{1} q(\mu;\alpha_k)\, d\mu . \tag{60}$$

If $\mu_{\min}(\alpha_k) > -1$, we determine step segments sequentially inward from both ends toward $\mu_{\min}$; otherwise, we work downward from $+1$. Because q is monotone decreasing as we work inward,

$$s_{n,k} = \max_{\mu_{n-1,k} \le \mu \le \mu_{n,k}} (q(\mu)) = q(\mu_{outer}) , \tag{61}$$

where μ_{outer} is the value at the outer end, which is already known. Then, the area of the interval and the value of s just obtained determines μ_{inner}:

$$\mu_{inner} = \mu_{outer} \pm \frac{A_i}{s_{interval} N} , \tag{62}$$

where the plus applies when moving inward from left to right and the minus applies when moving inward from right to left. When all the cosines have been found, the value of s for the interval between the innermost points from each side is the larger of the values of q at the ends of the interval. The area under the resulting step-sampling function is larger than the area used to construct it. So, we iterate on the area, letting

$$A_k = \sum_{n=1}^{N} (\mu_{n,k} - \mu_{n-1,k}) s_{n,k} \tag{63}$$

and reconstructing the sampling function, until the area converges to some tolerance. We require that the area match the previous iteration's area to at least ten digits and that the area of each segment match A_k/N to at least ten digits.

For testing this method, we let $N = 32$, for which the acceptance efficiency in sampling μ from this s and testing for rejection using $q(\mu;\alpha_k)$ exceeds 96% for $0 \le \alpha_k \le 1 + \sqrt{3}$.

IV.B. Partitioning the Domain of α

Start with a worst-case efficiency goal, $\eta_{\min}$; we chose 90%. Then, construct a table of piecewise-constant sampling functions for each of a set α_k. Starting at $\alpha_0 = 0$, construct the piecewise-constant $s_0(\mu)$ for that α as described in Sec. IV.A. The acceptance efficiency (for 32 segments) is $\eta_0 \approx 0.978$ for $\alpha = \alpha_0$. (For convenience and conservatism, we used the worst-case efficiency instead.) The efficiency at a larger α is

$$\eta(\alpha,\alpha_k) = \frac{\int_{-1}^{1} q(\mu;\alpha)\,d\mu}{\int_{-1}^{1} s_k(\mu)\,d\mu} = \eta_k \frac{\int_{-1}^{1} q(\mu;\alpha)\,d\mu}{\int_{-1}^{1} q(\mu;\alpha_k)\,d\mu} . \tag{64}$$

Sequentially, for $k = 1,2,\ldots,$ root solve for α_{k+1} such that $\eta(\alpha_{k+1},\alpha_k) = \eta_{\min}$. We approximated this conservatively by choosing α_{k+1} such that

$$\eta(\alpha_{k+1},\alpha_k) = 0.96 \frac{\int_{-1}^{1} q(\mu;\alpha_{k+1})\,d\mu}{\int_{-1}^{1} q(\mu;\alpha_k)\,d\mu} . \tag{65}$$

When $\alpha_{k+1} \ge \alpha_{\max}$, the table can end at $K = k$. We used $\alpha_{\max} = 1 + \sqrt{3}$.

Finally, the complete tabular sampling function is

$$s(\mu;\alpha) = s_{n,k} \ni \begin{cases} \alpha_k = \max\limits_{0\le k'\le K} (\alpha_{k'} \le \alpha) \\ \mu_{n,k} = \min\limits_{1\le n'\le N} (\mu_{n',k} \ge \mu) , \end{cases} \tag{66}$$

where K is the largest value of k. For our choices, $K = 20$. Note that while the interval boundary values run from $\mu_0 = -1$ through $\mu_N = +1$, the intervals between adjacent boundaries are indexed from $n = 1$ for the interval $\mu_0 \le \mu \le \mu_1$ and in general $\mu_{n-1} < \mu \le \mu_n$, up through $n = N$, for the interval $\mu_{N-1} < \mu \le \mu_N$.

IV.C. Efficient Implementation

The first step is to use the input α to find the index k. To avoid the cost of a search when drawing a sample, and having constructed a good partition as in Sec. IV.B, find a function that approximates these partition values. The approximation need not be highly accurate; it is good enough if it results in an efficient table for drawing samples. It is equally important that it be inexpensive to compute. We chose $K = 20$ and the rational polynomial

$$\kappa(\alpha) = \frac{4 + \alpha(132 + 45\alpha)}{4 + \alpha(8 + \alpha)} . \tag{67}$$

Then, root solve to find α_k for which $\kappa(\alpha_k) = k$ for $k = 1, 2, \ldots, K$. Then, build new data tables using these α_k. In sampling, given some α, the k to use in the data table so constructed is the floor of κ: $\lfloor \kappa(\alpha) \rfloor$. Two random numbers are needed for each attempt until a sample is accepted and the energy is reduced using the accepted μ.

Adding a table of the minimum value of q in each bin,

$$q_{n,k}^{\min} = \min_{\substack{\mu_{n-1,k}\le\mu\le\mu_{n,k} \\ \alpha_k\le\alpha<\alpha_{k+1}}} q(\mu;\alpha) , \tag{68}$$

where $\alpha_{K+1} = 1 + \sqrt{3}$, allows acceptance (usually) without the cost of computing `q(mu, alpha)`. The usual test for acceptance may be preceded by a pretest using $q_{n,k}^{\min}$. The algorithm, in Fortran syntax, is

```
k = Floor(kappa(alpha))
Do
  xi1 = 32.0 * Random()
  n = Floor(xi1)
  fp = xi1 - Real(n)  ! i.e. Fractional
    part of xi
  mu = muData(n-1, k) + fp * (muData(n, k)-
    muData(n-1, k))
  xi2 = Random()
  If (qMin(n, k) < s(n, k) * xi2) Exit
    ! PRETEST, optional
  If (q(mu, alpha) < s(n, k) * xi2) Exit
End do
alpha = alpha / (1.0 + alpha * (1.0 -
    mu)) .
```

V. PERFORMANCE

The methods were implemented in Fortran 95. The resulting code was used to measure run times to the millisecond for 10^8 repetitions of each routine for each plotted value of α (to obtain timing accuracy and reduce random variability). Acceptance efficiency and random number usage were evaluated by numerical and/or analytical integration in Mathematica 7.

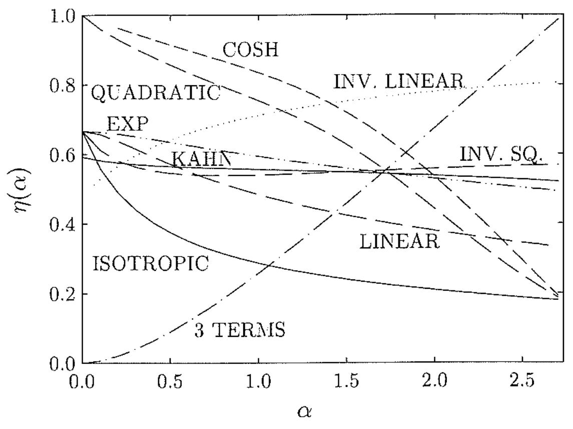

Fig. 4. Acceptance efficiency of algorithmic methods.

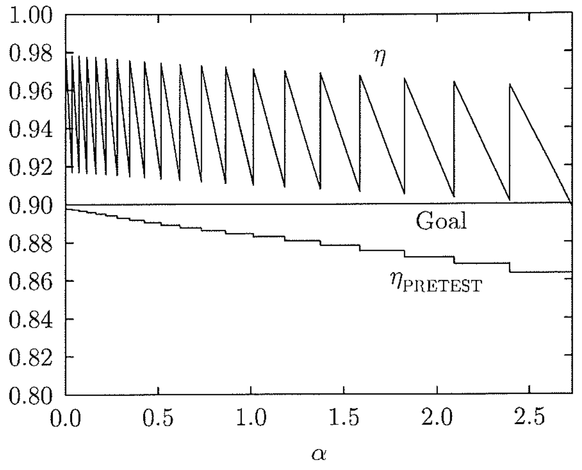

Fig. 5. Acceptance efficiency of step sampling with rejection.

All testing was conducted using Intel Visual Fortran[a] with integration into the Microsoft Visual Studio development environment.[b] Derivations, efficiency calculations, and plots used Wolfram Mathematica.[c] The algorithms were tested using a Hewlett-Packard computer[d] with an Intel i7 processor[e] running the Microsoft Windows 7 operating system.[f]

V.A. Efficiency

On the basis of acceptance efficiency of the analytic rejection schemes alone, a combination of quadratic, hyperbolic, inverse linear, and three-terms methods is optimal, as can be seen in Fig. 4. Note that the acceptance efficiency of rejection does not depend on the method of sampling from the quadratic, so only one curve is shown for the quadratic sampling function.

The efficiency of the table-based step rejection method could be made as high as one likes by using sufficiently large tables. Figure 5 shows the acceptance efficiency for the pretest and the overall acceptance efficiency of the two tests combined, which is equivalent in efficiency to the method without the pretest. Discontinuities occur at the tabulated values of α. The first test is passed in 86.4% to 89.8% of trials, and the combined acceptance efficiency ranges from 89.8% to 97.8%. Although the combination approach is more efficient for very small α and for α very close below $1 + \sqrt{3}$, the tabular method is most efficient overall. (In comparing Figs. 4 and 5, note the difference in scale.)

Maximum efficiency without tables is achieved by the combination

$$\begin{array}{l}\text{Maximum}\\ \text{Efficiency}\\ \text{Method}\end{array} = \begin{cases} \text{Quadratic Analytic} & 0 \le \alpha < 0.20285 \\ \text{Cosh} & 0.20285 \le \alpha < 1.40585 \\ \text{Inverse Linear} & 1.40585 \le \alpha < 2.25917 \\ \text{Three Terms} & 2.25917 \le \alpha < 1 + \sqrt{3} \\ \text{Koblinger's} & 1 + \sqrt{3} \le \alpha \ . \end{cases} \tag{69}$$

V.B. Parsimony

As can be seen in Fig. 6, the most parsimonious analytic method is a combination of quadratic analytic or root solving, hyperbolic cosine, and inverse linear:

$$\begin{array}{l}\text{Maximum}\\ \text{Parsimony}\\ \text{Method}\end{array} = \begin{cases} \text{Quadratic Analytic} & 0 \le \alpha < 0.20285 \\ \text{Cosh} & 0.20285 \le \alpha < 1.40585 \\ \text{Inverse Linear} & 1.40585 \le \alpha < 1 + \sqrt{3} \\ \text{Koblinger's} & 1 + \sqrt{3} \le \alpha \ . \end{cases} \tag{70}$$

Kahn's method requires about twice as many random numbers per sample as does this combination.

The tabular step method requires only two random numbers per trial. For the table used here, with acceptance efficiency of 90% to 98%, the number of random numbers used ranges from 2.04 to 2.23. This is the most parsimonious method for $\alpha > 0.55$, nearly as good for

[a]Intel® Visual Fortran Compiler, 11.1.051., Integration for Microsoft Visual Studio 2008, 11.1.3468.2008, © 2002–2009 Intel Corporation.

[b]Microsoft Visual Studio 2008, Version 9.0.21022.8 RTM, © 2007 Microsoft Corporation with Microsoft .NET Framework, Version 3.5, © 2007 Microsoft Corporation.

[c]Wolfram Mathematica 7.0.1.0 for Microsoft Windows (64 bit), © 1988–2009 Wolfram Research, Inc.

[d]Hewlett-Packard Pavilion dv7 Notebook personal computer.

[e]Intel® Core™ i7 CPU, Q720 at 1.60 GHz.

[f]Microsoft Windows 7 Home Premium, © 2009 Microsoft Corporation, Service Pack 1.

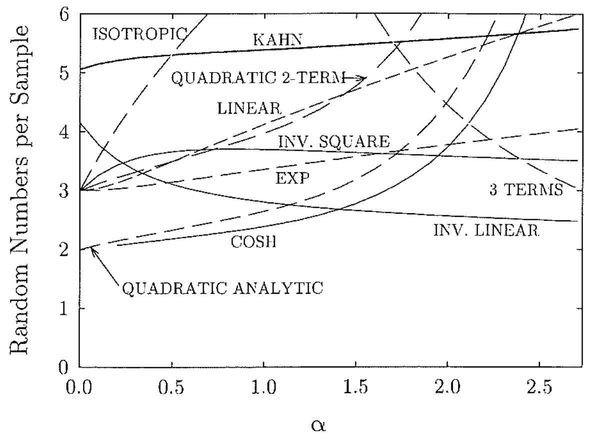

Fig. 6. Parsimony of algorithmic methods.

lower energies, and the most parsimonious on average. Kahn's method uses 2.3 to 2.8 times as many random numbers per sample.

V.C. Computational Speed

Computational speeds (samples per microsecond CPU time) are listed in Table I. These results are minimum, mean, and maximum for $\alpha = 0.003, 0.103, \ldots, 2.703$, computed on an Intel i7 laptop computer. On this basis alone, the best analytic method was inverse square rejection, which at its slowest is faster than Kahn's method. (For energies below ~50 keV, isotropic sampling is somewhat faster than inverse square sampling.)

Because the efficiencies of many of the methods vary substantially with α, as shown in Fig. 4, the speeds of the methods inherit this dependence. Consequently, the average speed of a method depends on how it is used. Table II lists average speeds, as ratios to the corresponding speed of Kahn's method, for three tests. The second column shows results when values of α were chosen uniform randomly in the interval $0.2 \leq \alpha \leq 2.73$. This is disadvantageous for the tabular methods because it stresses the memory caching for the lookup tables. The third column starts at $\alpha = 2.73$ and reduces α through the random samples drawn for each subsequent sample until $\alpha < 0.01$. This is repeated until 10^8 samples have been drawn. This test emulates usage in a transport code and results in better performance of the tabular methods (better than random α) because table entries can be reused for several successive samples. For the algorithmic methods, the third column has much greater weighting of low values of α, because gammas retain almost all their energy in K-N scatters at low energy. The fourth column is derived from the values in the mean column of Table I. Thus, it weights α uniformly.

The performances of the four fastest methods are shown as functions of α in Fig. 7. These speeds are obtained by drawing 10^8 samples with each method at each of $\alpha = 0.003, 0.103, \ldots, 2.73$. The pretest, which usually avoids computing $q(\mu; a)$ in the step method (see Fig. 5), does speed the rejection test. With it, the step method is nearly twice as fast as Kahn's method everywhere in the range of α for which Koblinger's method cannot be used. However, in the more realistic testing presented in Table II, its advantage is reduced by the extra table lookup required to use it.

TABLE I

Speed of Methods*

Method	Minimum	Mean	Maximum
Step with pretest	6.990	7.235	7.428
Step without pretest	6.152	6.359	6.430
Inverse square	4.559	4.723	5.560
Kahn	3.669	4.091	4.294
Maximum efficiency	2.336	3.135	4.310
Maximum parsimony	2.304	3.096	4.237
Inverse linear	2.852	3.913	4.262
Linear	2.387	3.192	4.376
Exponential	2.486	2.815	3.175
Isotropic	1.716	2.726	5.892
Quadratic two terms	0.854	2.547	4.099
Three terms	0.575	2.531	4.305
Quadratic analytic	0.900	2.478	3.653
Hyperbolic	0.846	2.122	4.034
Quadratic root solving	0.718	2.027	2.873

*In samples per microsecond.

V.D. Additional Performance Considerations

The optimal choice of methods depends on (a) the code designer's relative weighting of the three objectives—efficiency, parsimony, and speed—and (b) the performance (in each objective) of the methods as delivered by the computer system to be used.

While efficiency and parsimony are properties of the algorithm only (and of the table sizes for the tabular method), relative speeds of the methods are machine dependent because parsimony affects speed in a machine-dependent way. The testing reported here used Intel Fortran's intrinsic (serial) random number generator in serial code. Use of a vectorized multicore parallel random number generator, such as is available in the Intel MKL (math kernel library), in an otherwise serial code would nearly eliminate the cost of random numbers, so that parsimony would have less influence on speed. However, as we move to petascale computing, high-quality 128-bit random number generators are needed, and presumably, no processors will be dedicated to generating random numbers. We anticipate that parsimony will have increased influence on speed in such systems due to the cost of random number generators with the necessary cycle lengths (numbers of random numbers generated

TABLE II

Ratios of Speeds

Method	Ratio of Method Speed to Speed of Kahn's Method		
Test Scheme: Selection of Values for α	Random in [0.2,2.73]	Decreasing by Scatters: 2.731 to 0.001	Various, Repeated 0.003 to 2.723
Step with pretest	1.65	1.78	1.80
Step without pretest	1.61	1.75	1.57
Inverse square	1.10	1.37	1.16
Kahn	1	1	1
Maximum efficiency	0.70	0.90	0.77
Maximum parsimony	0.71	0.91	0.76
Quadratic two terms	0.49	0.96	0.62
Quadratic root solving	0.40	0.70	0.49
Quadratic analytic	0.49	0.89	0.60
Exponential	0.66	0.79	0.68
Inverse linear	0.91	0.75	0.96
Three terms	0.48	0.96	0.62
Hyperbolic	0.46	0.59	0.51
Linear	0.71	1.07	0.78
Isotropic	0.57	1.37	0.66

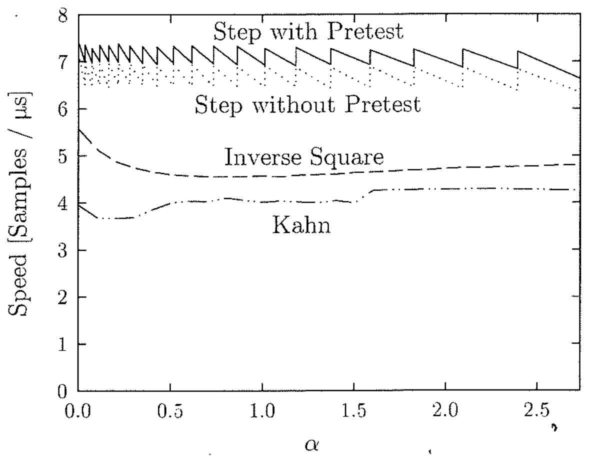

Fig. 7. Speed of fastest methods.

before starting to repeat the pseudorandom sequence). Acceptance efficiency is a consideration in its own right, because of the variability in the time required to obtain a sample when efficiency is low. Also, for computations that run into the tens of thousands of central processor unit hours, even a modest speedup can be significant.

For step sampling and rejection with the resolution tested here, the tables (including the pretest table) required 1940 reals. Higher resolution in both dimensions would produce marginal gains in efficiency and parsimony while requiring more memory. The latter would increase the incidence of cache misses, reducing speed, while the higher efficiency would increase speed. This presents an opportunity for optimization that we did not pursue because the optimum is implementation, compiler, and machine dependent.

VI. RECOMMENDATIONS

Step sampling with rejection, including the pretest, is the overall best choice for $0 \leq \alpha \leq 1 + \sqrt{3}$. It is, overall, ~75% faster than Kahn's method, with acceptance efficiency of 90% to 98% and only two random numbers used per trial. We did not test it above that range, because the needed resolution in α increases as α increases, and there is no obvious upper bound for α. Therefore, we recommend the combination of step sampling with rejection and pretest for $\alpha \leq 1 + \sqrt{3}$ and Koblinger's method for $\alpha > 1 + \sqrt{3}$.

For applications in which minimal memory usage is more important than speed, the inverse square method provides the best speed without tables and is a simple algorithm. It is faster and more parsimonious than Kahn's method while being of comparable efficiency. For $\alpha \leq 1 + \sqrt{3}$, inverse square rejection is recommended as a general replacement for Kahn's method, with Koblinger's method used for $\alpha > 1 + \sqrt{3}$, for such applications.

For the simplest implementation with reasonable performance, we recommend inverse linear rejection for all α. The simplicity is offset by its use of a square root, a logarithm, and an exponential. It uses ~25% more run time (in our testing) and ~10% to 25% more random numbers than Koblinger's method for $\alpha > 1 + \sqrt{3}$.

However, if computational cost were dominated by the random number generator, then parsimony would determine speed, making the maximum parsimony method in Eq. (70) the best choice.

ACKNOWLEDGMENTS

This work was conducted at the Air Force Institute of Technology as part of the official duties of the author.

REFERENCES

1. D. W. GERTS, Senior Scientist, Idaho National Laboratory, Personal Communication (2011).

2. R. N. BLOMQUIST and E. M. GELBARD, "An Assessment of Existing Klein-Nishina Monte Carlo Sampling Methods," *Nucl. Sci. Eng.*, **83**, 380 (1983).

3. Y. S. HOROWITZ, A. DUBI, and S. MORDECHAI, "The Monte Carlo Nonuniform Sampling Technique Applied to the Klein-Nishina Probability Density Function," *Nucl. Sci. Eng.*, **60**, 461 (1976).

4. L. KOBLINGER, "Direct Sampling from the Klein-Nishina Distribution for Photon Energies Above 1.4 MeV," *Nucl. Sci. Eng.*, **56**, 218 (1975).

5. H. KAHN, "Applications of Monte Carlo," AECU-3259, National Technical Information Service (1954).

6. I. LUX and L. KOBLINGER, *Monte Carlo Particle Transport Methods: Neutron and Photon Calculations*, CRC Press, Boca Raton, Florida (1991).

7. E. E. LEWIS and W. F. MILLER, Jr., *Computational Methods of Neutron Transport*, American Nuclear Society, La Grange Park, Illinois (1993).