# Supplying Local Microphysics Parameterizations with Information about Subgrid Variability: Latin Hypercube Sampling

VINCENT E. LARSON

*Atmospheric Science Group, Department of Mathematical Sciences, University of Wisconsin—Milwaukee, Milwaukee, Wisconsin*

JEAN-CHRISTOPHE GOLAZ

*National Research Council, Naval Research Laboratory, Monterey, California*

HONGLI JIANG AND WILLIAM R. COTTON

*Department of Atmospheric Science, Colorado State University, Fort Collins, Colorado*

(Manuscript received 12 October 2004, in final form 2 June 2005)

ABSTRACT

One problem in computing cloud microphysical processes in coarse-resolution numerical models is that many microphysical processes are nonlinear and small in scale. Consequently, there are inaccuracies if microphysics parameterizations are forced with grid box averages of model fields, such as liquid water content. Rather, the model needs to determine information about subgrid variability and input it into the microphysics parameterization.

One possible solution is to assume the shape of the family of probability density functions (PDFs) associated with a grid box and sample it using the Monte Carlo method. In this method, the microphysics subroutine is called repeatedly, once with each sample point. In this way, the Monte Carlo method acts as an interface between the host model's dynamics and the microphysical parameterization. This avoids the need to rewrite the microphysics subroutines.

A difficulty with the Monte Carlo method is that it introduces into the simulation statistical noise or variance, associated with the finite sample size. If the family of PDFs is tractable, one can sample solely from cloud, thereby improving estimates of in-cloud processes. If one wishes to mitigate the noise further, one needs a method for reduction of variance. One such method is Latin hypercube sampling, which reduces noise by spreading out the sample points in a quasi-random fashion.

This paper formulates a sampling interface based on the Latin hypercube method. The associated family of PDFs is assumed to be a joint normal/lognormal (i.e., Gaussian/lognormal) mixture. This method of variance reduction has a couple of advantages. First, the method is general: the same interface can be used with a wide variety of microphysical parameterizations for various processes. Second, the method is flexible: one can arbitrarily specify the number of hydrometeor categories and the number of calls to the microphysics parameterization per grid box per time step.

This paper performs a preliminary test of Latin hypercube sampling. As a prototypical microphysical formula, this paper uses the Kessler autoconversion formula. The PDFs that are sampled are extracted diagnostically from large-eddy simulations (LES). Both stratocumulus and cumulus boundary layer cases are tested. In this diagnostic test, the Latin hypercube can produce somewhat less noisy time-averaged estimates of Kessler autoconversion than a traditional Monte Carlo estimate, with no additional calls to the microphysics parameterization. However, the instantaneous estimates are no less noisy. This paper leaves unanswered the question of whether the Latin hypercube method will work well in a prognostic, interactive cloud model, but this question will be addressed in a future manuscript.

*Corresponding author address:* Vincent E. Larson, Dept. of Mathematical Sciences, University of Wisconsin—Milwaukee, P.O. Box 413, Milwaukee, WI 53201.
E-mail: vlarson@uwm.edu

## 1. Introduction: Microphysical parameterization in the presence of small-scale variability

Numerical models of the atmosphere have difficulty representing small-scale cloud microphysical processes, such as drizzle formation. This would be difficult even if microphysical formulas were known with perfect accuracy. This is because it is also difficult for a host model to provide accurate fields with which to drive the microphysics. Often a model cannot achieve good results simply by inputting the grid box averages into a microphysical parameterization (Sommeria and Deardorff 1977; Rotstayn 2000; Pincus and Klein 2000; Larson et al. 2001). This is because of two reasons: first, computational constraints preclude the use of grid boxes that are small enough to fully resolve cloud microphysical processes; and second, microphysical formulas are often nonlinear. The coarse resolution means that grid-box-sized volumes contain significant subgrid variability. The nonlinearity means that, in order to compute the microphysics accurately, the subgrid variability about the average must be taken into account. For example, a microphysics parameterization may need to be informed that a grid-box-sized region is filled partially with clouds, or entirely with uniform clouds, or entirely with clouds of variable liquid water, etc.

When we account for subgrid variability, we desire a methodology that is compatible with many types of parameterizations. These parameterizations may range from simplified analytic formulas to complex numerical subroutines. They may represent processes such as autoconversion of cloud droplets to drizzle drops, activation of cloud condensation nuclei, or even atmospheric chemistry. Also, a researcher may wish to compare simulations that use the same host model but two different parameterizations. In all these cases, it is more convenient to account for subgrid variability once for all parameterizations, rather than using separate methods for each parameterization.

One general method of accounting for subgrid variability is the Monte Carlo method (Johnson 1987; Gentle 2003; Pincus et al. 2003; Räisänen et al. 2004; Räisänen and Barker 2004). The idea is to draw a separate random sample from each grid box and time step. The microphysics parameterization is then called once per sample point using the values of vertical velocity $w$, cloud water mixing ratio $r_c$, droplet number mixing ratio $N_c$, and so forth associated with each sample point. Finally, the microphysical calculations based on each sample point are suitably averaged. In this way, a Monte Carlo scheme acts as a general interface between the model dynamics and the microphysics, that is, between model-prognosed quantities like $r_c$ and the microphysics parameterization. In principle, this allows one to insert new microphysics parameterizations in an almost plug-and-play fashion (Pincus et al. 2003).

A problem is that at each time step, the sampling introduces statistical noise into the model. The statistical noise is reduced in a time average by cancellations between time steps. It is desirable that the noise be reduced as rapidly as possible, since cloud fields change over time. This can be accomplished by a so-called variance reduction method (Räisänen and Barker 2004).

This paper applies to microphysics parameterization a variance reduction method known as Latin hypercube sampling (McKay et al. 1979; Owen 2003, hereafter OW03; Press et al. 1992). The Latin hypercube method reduces variance by preventing the sample points from clumping together in sample space, as can happen with purely random points. The Latin hypercube method can be used with an arbitrary number of variates (e.g., $w$, $r_c$, $N_c$, etc.). The Latin hypercube method can use several sample points per grid box and time step. But even if only one sample point is used, it can still reduce the time-averaged noise (but not instantaneous noise) as compared with traditional Monte Carlo sampling. We will consider only processes that occur locally within a single grid box, such as autoconversion, rather than processes like radiative transfer that depend on cloud overlap between vertically displaced grid boxes. The cloud overlap problem has been addressed by Räisänen et al. (2004) and Räisänen and Barker (2004). To reduce sampling noise, we will sample only from the cloudy portion of the grid box, although our algorithm can be easily modified to handle processes that occur throughout the entire grid box.

This paper is structured as follows. Section 2 provides background on methods to account for subgrid variability. Section 3 describes the Latin hypercube method and how to implement it for cloud microphysical problems. The method requires one to choose a family of probability density functions. We choose a mixture (i.e., sum) of two normal/lognormal functions. In section 4, we present diagnostic tests of statistical errors that arise from Latin hypercube sampling. As an example of a microphysical formula, we will discuss the Kessler autoconversion formula. Although the Kessler formula has the defect of neglecting variations in droplet number mixing ratio (Cotton and Anthes 1989), it is commonly used and it permits one to construct analytic solutions for purposes of comparison. In section 5, we present a summary and conclusions.

## 2. Background: The problem, stated mathematically

For concreteness, suppose that we want to compute autoconversion $A$, averaged over a grid box, in a way that accounts for subgrid variability. The essential problem, mathematically expressed, is to compute an integral,

$$\overline{A(r_c)} = \int_0^\infty dr_c P(r_c) A(r_c). \quad (1)$$

Here $\overline{()}$ denotes a grid box average, and $\overline{A(r_c)}$ represents the grid box average of the autoconversion rate, $A$. Here $A$ is assumed to be a function of cloud water $r_c$ alone. In simple parameterizations, $A$ is an analytic function; in more complex parameterizations, $A$ is a numerical algorithm. Here $P(r_c)$ is the probability density function (PDF) of $r_c$ within the grid box. The integral states that the average autoconversion rate is a weighted sum of autoconversion rates. That is, to compute the average, we must compute the autoconversion rate for each value of $r_c$ within the grid box, weight by the probability of occurrence of that value of $r_c$, and sum all the values.

The fastest and most accurate way to compute $\overline{A}$ is to integrate (1) analytically. This is possible only if both $P$ and $A$ are simple, analytic functions. Furthermore, when $A$ is a simple function of only one variable, it costs relatively little to perform straightforward numerical integration.

The cost escalates if $A$ is a microphysics parameterization that must be computed numerically, and the parameterization depends on several variables. For instance, suppose $A$ were a function of $d$ variables, and we wanted to evaluate it using a computational mesh with $b$ points in each direction in sample space (not physical space). A straightforward integration would involve $b^d$ evaluations. For example, if $A$ were a general microphysics package that depended on 5 variables, and one desired an integration over 4 mesh points per variable, one would need to evaluate $A$ at 1024 mesh points!

Calling a microphysics parameterization 1024 times in each grid box and time step of a simulation is prohibitively expensive. Pincus et al. (2003) addresses this problem by making clever use of the traditional Monte Carlo method, that is, drawing sample points randomly from the PDF, $P(r_c)$. A small number of sample points, say 1, can be chosen per grid box and time step, and those points are used to evaluate the microphysics. At the next time step, a new set of sample points is chosen for the grid box, and the microphysics is reevaluated. After a period of time, the hope is that a suitably accurate microphysical average will emerge, even though the microphysics contains statistical noise at each time step (Pincus et al. 2003; Räisänen et al. 2004; Räisänen and Barker 2004). A related approach is to use the aforementioned straightforward integration, evaluate at 1 of the $b^d = 1024$ mesh points per time step, and achieve a time-averaged answer after 1024 time steps. This is a lot of time steps! However, if one insists on evaluating at all mesh points, one should at least scramble the order in which the mesh points are evaluated, in order to reduce the number of time steps that occur before a reasonably unbiased estimate emerges.

Instead, we advocate Latin hypercube sampling, which combines advantages of straightforward integration with those of the traditional Monte Carlo method. Latin hypercube sampling chooses a particular kind of scrambled-order sample consisting of $b$ points from the $b^d$ mesh points, and perturbs those points by a small random factor.

## 3. The Latin hypercube algorithm

This section describes our proposed method of variance reduction, Latin hypercube sampling. Latin hypercube sampling forms an interface between the dynamical part of the model and a microphysical parameterization. The interface generates a set of input values that can be fed into the microphysical parameterization.

### a. Overview of Latin hypercube sampling

For concreteness, our exposition will assume that the microphysics depends on five variables: total water mixing ratio (vapor + cloud water) $r_t$, liquid water potential temperature $\theta_l$, vertical velocity $w$, cloud droplet number mixing ratio $N_c$, and drizzle mixing ratio $r_r$. The variables $N_c$ and $r_r$ are included because they are useful for characterizing general drizzle processes not discussed in this paper. Cloud water mixing ratio, $r_c$, is not included because it can be diagnosed from $r_t$ and $\theta_l$ if we assume that supersaturation leads instantaneously to condensation. However, the method can be generalized to any number of variables, that is, predictands.

The Latin hypercube method requires assuming a functional form for the family of PDFs representing variability in a grid box. For the part of the PDF family that represents $r_t$, $\theta_l$, and $w$, we have chosen a mixture of two Gaussians. This permits both positive and negative skewness. For $N_c$ and $r_r$, we assume a single lognormal form. Lognormal PDFs have a positive domain and are always positively skewed, that is, they contain many small values and few large values. Lognormal

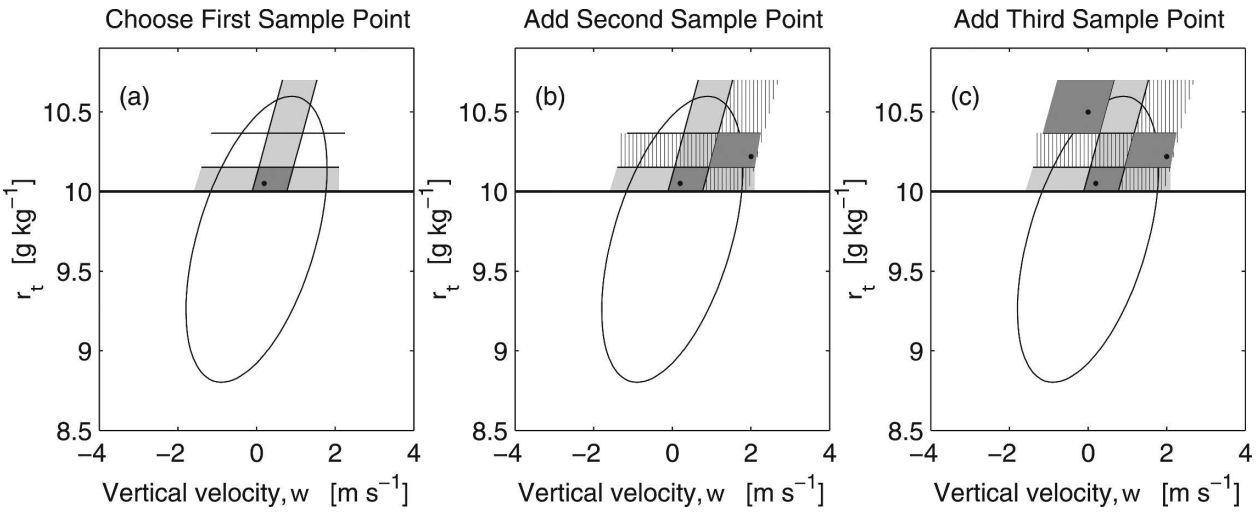


FIG. 1. A three-point Latin hypercube sample taken from the cloudy portion ($r_t > 10$) of a bivariate ($w \times r_t$) single Gaussian PDF. The PDF has been divided into three "rows" i.e., strips of nearly constant $r_t$, and three "columns," i.e., strips of roughly constant $w$. (a) A "square" is chosen at random (dark shading) and within it is randomly chosen the first sample point (dot). The square's column and row (light shading) are excluded when selecting subsequent squares. (b) The second sample point is chosen, and its row and column are excluded (vertical hatching). (c) The third point is selected randomly from within the only square that remains. All squares are associated with equal probability (figure not drawn to scale).

PDFs have been used by many authors. Their chief advantage is simplicity. An unlimited number of log-normally distributed hydrometeor categories may be added to the PDF without mathematical difficulty.

The inputs to the Latin hypercube interface are statistical moments of the variables ($r_t$, $\theta_l$, $w$, $N_c$, $r_r$). Specifically, for the Gaussian-mixture variables ($r_t$, $\theta_l$, $w$), the input is the means and covariance matrix for *each* Gaussian, plus the mixture fraction, $a$, which determines the relative magnitude of the two Gaussians. This is a total of 19 independent inputs. For the log-normal variables ($N_c$, $r_r$), the input is the *grid box* means and covariance matrix. Also needed are covariances between ($r_t$, $\theta_l$, $w$) and ($N_c$, $r_r$). The output of the Latin hypercube interface is a user-chosen number of $d$-dimensional sample points from the distribution.

The goal of Latin hypercube sampling is to spread out the sample points, so that low and high and moderate values of each variate are all contained in the sample. To illustrate the method, suppose that we want to generate a sample of three points from a bivariate PDF of $w$ and $r_t$. Suppose furthermore that we want to sample only cloudy points, assumed here to have $r_t > 10$ g kg$^{-1}$ for simplicity. To use the Latin hypercube method, we divide the area in $w \times r_t$ space covered by the PDF into a checkerboard of nine "squares" or "hypercubes" (see Fig. 1). Each square encompasses equal probability, which means that these squares may be distorted in shape and/or unequal in area. We then choose a square at random. Within this square, we choose a point randomly according to the PDF. Then we strike out the square's row and column, and from the remaining rows and columns, choose another square at random. From within this second square, we choose a point randomly. Finally, we iterate once more, thereby selecting the third point (Press et al. 1992, p. 315). In the example in Fig. 1, the three sample points are chosen from the following squares: point 1 has ($w$, $r_t$) $\leftrightarrow$ (medium value, low value), point 2 has ($w$, $r_t$) $\leftrightarrow$ (high value, medium value), point 3 has ($w$, $r_t$) $\leftrightarrow$ (low value, high value).

We see that any three Latin hypercube sample pairs ($w$, $r_t$) are guaranteed to contain small, medium, and large values of both $w$ and $r_t$. This is useful for driving a microphysics package containing many schemes, each of which varies primarily with one variable (OW03). For instance, a cloud droplet activation scheme may vary strongly with $w$, whereas an autoconversion scheme may depend mostly on $r_t$ (i.e., $r_c$). If the microphysics package is fed a Latin hypercube sample, the activation scheme will sample a full range of $w$ values, and the autoconversion scheme will fully sample $r_t$. This is advantageous. In general, Latin hypercube sampling performs well whenever the quantity to be averaged is a sum of univariate functions (OW03).

In contrast, for schemes that depend strongly on the correlation of two variables, Latin hypercube sampling has little advantage over traditional Monte Carlo sampling (Press et al. 1992). However, Latin hypercube is guaranteed not to perform much worse. Specifically,

Owen (1997) has proven that the variance of a $n_t$-point Latin hypercube sample, $\mathcal{V}_{\mathrm{LHS}}$, is related to the variance of a traditional $n_t$-point Monte Carlo sample, $\mathcal{V}_{\mathrm{MC}}$, by

$$\mathcal{V}_{\mathrm{LHS}} \leq \frac{n_t}{n_t - 1}\,\mathcal{V}_{\mathrm{MC}}, \quad \text{where} \quad n_t > 1. \tag{2}$$

In other words, a Latin hypercube sample with $n_t > 1$ points is never worse, on average, than a Monte Carlo sample with one fewer point. The proof holds for any square-integrable (autoconversion) integrand.

### b. Generating a uniformly distributed Latin hypercube sample

Most random number generators generate a series of numbers between 0 and 1 that are distributed uniformly. For this reason, it is simplest to draw the Latin hypercube sample from a uniform PDF with independent variates. Later we transform the sample points to points from the PDF of interest, in this case a normal/lognormal mixture. We choose $n_t$ Latin hypercube points from a $(d + 1)$-variate uniform PDF. The first $d$ variates represent, in our case, our $d = 5$ variables ($r_t$, $\theta_l$, $w$, $N_c$, $r_r$). The $(d + 1)$th variate, the mixture variate, is used to select one of the two mixture components or "plumes" from the binormal/lognormal PDF, as described below.

To construct the Latin hypercube sample, we use the following algorithm (OW03). We first generate $(d + 1)$ independent columns consisting of random permutations of the integers $[0, \ldots, (n_t - 1)]$ (OW03; Devroye 1986). These column vectors form a $n_t \times (d + 1)$ matrix, $\Pi_{ij}$. This matrix is used to select the squares in the sample space. Next, we choose the location of each sample point randomly within its square. To do so, we form a matrix $\mathbf{U}$ whose elements are $n_t \times (d + 1)$ real numbers drawn independently and randomly from a uniform distribution varying between (but not including) 0 and 1. Our random number generator is that of Press et al. (1992, 281–282), based on L'Ecuyer (1988). Finally, the set of $n_t$ Latin hypercube samples is given by the $n_t \times (d + 1)$ matrix $\mathbf{V}$,

$$V_{ij} = \frac{1}{n_t}\Pi_{ij} + \frac{1}{n_t}U_{ij}. \tag{3}$$

Each row of $\mathbf{V}$ is a $(d + 1)$-variate Latin hypercube sample, with each column corresponding to a different variable: $r_t$, $\theta_l$, $w$, $N_c$, $r_r$, or the mixture variate. Now consider the first column, which corresponds to $n_t$ sample points for $r_t$. Because the first column of $\Pi$ (and all the others) consists of a permutation of $0, 1, \ldots, (n_t - 1)$, the values of the first column of $\mathbf{V}$ are spread out or stratified, with one value between 0 and $1/n_t$, another between $1/n_t$ and $2/n_t$, and so forth. This leads to an improvement in the estimates of functions of $r_t$, relative to Monte Carlo sampling without stratification.

We do not need to use all $n_t$ sample points for a given grid box during one time step. Instead, we may save the values of the $n_t$ sample points between time steps, using without replacement $n < n_t$ of the sample points at a time, so that after $n_t/n$ time steps all $n_t$ points are used once (and only once). Then a new batch of $n_t$ points may be generated. Here $n_t/n$ must be an integer. Below we shall focus on Latin hypercube samples with ($n = 1$, $n_t = 12$), ($n = 2$, $n_t = 12$), and ($n = 1$, $n_t = 1$). The latter case is simply traditional Monte Carlo sampling. The computational cost of ($n = 1$, $n_t = 12$) sampling is no more than traditional Monte Carlo ($n = 1$, $n_t = 1$) except for the small overhead associated with constructing the Latin hypercube matrix $V_{ij}$. The cost of ($n = 2$, $n_t = 12$) is double the other two sample types. We have chosen $n_t = 12$ only because it allows one to choose $n = 3$ if one so desires in the future. Other values of $n_t$ give similar results.

### c. A change of variables to facilitate sampling of cloudy points

Now we transform from the model-produced variables $r_t$ and $\theta_l$ to new variables $s$ and $t$. The purpose is to allow us to sample solely from the cloudy part of the grid box. The variable $s$ approximates $r_c$ when $s > 0$. The variable $t$ is orthogonal to $s$. When $t$ varies, $r_c$ remains unchanged. The variables $s$ and $t$ are approximated as linear combinations of $r_t$ and $\theta_l$. The transformation is a translation of the origin of the coordinates plus rotation and stretching.

Mathematically, the transformation between ($r_t$, $\theta_l$) and ($s$, $t$) is defined by the following equations. First, we divide $s$ and $t$ into mean $\overline{(\,)}$ and perturbation $(\,)'$ parts

$$s = \bar{s} + s', \tag{4}$$

$$t = \bar{t} + t'. \tag{5}$$

Consider the translation of the origin of coordinates. For simplicity, we let

$$\bar{t} = 0. \tag{6}$$

This entails no loss of generality. The location of $\bar{s}$ is set by [see Lewellen and Yoh (1993); also Sommeria and Deardorff (1977); Mellor (1977)]

$$\bar{s} \equiv \overline{r_t} - r_s(\overline{T_l}, p)\,\frac{[1 + \beta_1(\overline{T_l})\overline{r_t}]}{[1 + \beta_1(\overline{T_l})r_s(\overline{T_l}, p)]}, \tag{7}$$

where $r_t$ is the total water mixing ratio (vapor plus liquid). We have employed liquid water temperature, defined as (Sommeria and Deardorff 1977)

$$T_l \equiv T - \frac{L}{c_p} r_c = \theta_l \left(\frac{p}{p_0}\right)^{R_d/c_p}, \tag{8}$$

where $T$ is temperature, $L$ is the latent heat of vaporization, $R_d$ is the gas constant for dry air, $c_p$ is the specific heat of dry air at constant pressure, and $p_0$ is a reference pressure. The variable $T_l$ is approximately conserved under condensation, but unlike the liquid water potential temperature, $\theta_l$, $T_l$ is not conserved under changes in pressure. We have also defined a function, $\beta_1(T_l)$

$$\beta_1 = \beta_1(T_l) = \frac{R_d}{R_v}\left(\frac{L}{R_d T_l}\right)\left(\frac{L}{c_p T_l}\right), \tag{9}$$

where $R_v$ is the gas constant for water vapor. The function $\beta_1$ is dimensionless and ranges from about 150 at $T = 300$ K to about 337 at $T = 200$ K. Also, the saturation mixing ratio evaluated at $T = T_l$, $r_s(T_l, p)$, is given by

$$r_s(T_l, p) = \frac{R_d}{R_v} \frac{e_s(T_l)}{p - e_s(T_l)}, \tag{10}$$

where $p$ is pressure, and $e_s$ is the saturation vapor pressure over a liquid water surface.

The rotation and stretching of the coordinates is given by

$$s' = c_{r_t} r_t' - c_{\theta_l} \theta_l', \tag{11}$$

$$t' = c_{r_t} r_t' + c_{\theta_l} \theta_l', \tag{12}$$

where we have defined

$$c_{r_t} = \frac{1}{1 + \beta(\overline{T}_l) r_s(\overline{T}_l, p)}$$

and

$$c_{\theta_l} = \frac{1 + \beta(\overline{T}_l)\overline{r}_t}{[1 + \beta(\overline{T}_l) r_s(\overline{T}_l, p)]^2} \frac{c_p}{L} \beta(\overline{T}_l) r_s(\overline{T}_l, p) \left(\frac{p}{p_0}\right)^{R_d/c_p}.$$

To construct the PDF below, we need the covariance matrix of $s$, $t$, $w$, $N_c$, and $r_r$ for each Gaussian component. For simplicity, we will write the covariance matrix for $s$, $t$, and $w$ (the extension to the full covariance matrix, which includes $N_c$ and $r_r$, is straightforward) as follows:

$$\begin{bmatrix} s' \\ t' \\ w' \end{bmatrix} [s' \quad t' \quad w'] = \begin{bmatrix} \overline{s'^2} & \overline{s't'} & \overline{w's'} \\ \overline{s't'} & \overline{t'^2} & \overline{w't'} \\ \overline{w's'} & \overline{w't'} & \overline{w'^2} \end{bmatrix}. \tag{13}$$

To write this in terms of $r_t'$, $\theta_l'$, and $w'$, we substitute (11) and (12) for $s'$ and $t'$

$$\begin{bmatrix} s' \\ t' \\ w' \end{bmatrix} = \begin{bmatrix} c_{r_t} & -c_{\theta_l} & 0 \\ c_{r_t} & c_{\theta_l} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_t' \\ \theta_l' \\ w' \end{bmatrix} \equiv \mathbf{T} \begin{bmatrix} r_t' \\ \theta_l' \\ w' \end{bmatrix}. \tag{14}$$

Substituting (14) into (13), we find

$$\begin{bmatrix} \overline{s'^2} & \overline{s't'} & \overline{w's'} \\ \overline{s't'} & \overline{t'^2} & \overline{w't'} \\ \overline{w's'} & \overline{w't'} & \overline{w'^2} \end{bmatrix} = \mathbf{T} \begin{bmatrix} \overline{r_t'^2} & \overline{r_t'\theta_l'} & \overline{w'r_t'} \\ \overline{r_t'\theta_l'} & \overline{\theta_l'^2} & \overline{w'\theta_l'} \\ \overline{w'r_t'} & \overline{w'\theta_l'} & \overline{w'^2} \end{bmatrix} \mathbf{T}^{\mathrm{T}}. \tag{15}$$

Here $\mathbf{T}^{\mathrm{T}}$ is the transpose of $\mathbf{T}$.

### d. *Our assumed family of PDFs: A mixture of normal/lognormals*

Before we can transform the uniformly distributed sample points to the PDF of interest, we must construct our assumed PDF family, which is a mixture (i.e., sum) of two multivariate normal components, $G_1$ and $G_2$:

$$\begin{aligned} P[s, t, w, N_{c0} \ln(N_c/N_{c0}), r_{r0} \ln(r_r/r_{r0})] \\ = aG_1[s, t, w, N_{c0} \ln(N_c/N_{c0}), r_{r0} \ln(r_r/r_{r0})] \\ + (1 - a)G_2[s, t, w, N_{c0} \ln(N_c/N_{c0}), r_{r0} \ln(r_r/r_{r0})], \end{aligned} \tag{16}$$

where $0 \le a \le 1$ is the mixture fraction, that is, the weight of each Gaussian. Here $N_{c0}$ is a constant with a value equal to unit number mixing ratio in whatever units are desired, for example, 1 $g^{-1}$. Similarly, $r_{r0}$ is the constant unit drizzle mixing ratio. Because $N_{c0} \ln(N_c/N_{c0})$ and $r_{r0} \ln(r_r/r_{r0})$ are distributed according to a Gaussian mixture PDF, $N_c$ and $r_r$ are distributed according to a double lognormal PDF. For mathematical simplicity, we will reduce the double lognormal PDFs to single lognormal PDFs below.

We have chosen this particular PDF family because it is general enough to represent both cumulus and stratocumulus boundary layers, yet is mathematically tractable. Although the mixture fraction, $a$, is the same for all variables, skewness can differ for different variables because the width and means of the individual Gaussians can differ for different variables. All of the PDF's defining parameters, such as $a$, must be provided to the Latin hypercube sampler by the dynamical model. This can be done using the parameterization of Larson et al. (2002) and Golaz et al. (2002). However, one can choose other PDF families depending on the physical

system to be modeled and the level of simplicity desired.

To write an explicit form for $G_1$ and $G_2$, we introduce some notation. Let $\mathbf{X}$ be a $d$-component column vector representing the different random variables. For our case, $d = 5$ and $\mathbf{X}^{\mathrm{T}} = [s, t, w, N_{c0} \ln(N_c/N_{c0}), r_{r0} \ln(r_r/r_{r0})]$. Let $\mu_1$ be a $d$-component column vector that represents the means of the $d$ different variables for Gaussian mixture component 1. Likewise, $\mu_2$ represents the means of component 2. Next we let $\mathbf{\Sigma}_1$ be a $d \times d$ covariance matrix for component 1 whose $ij$ element is

$$\Sigma_{1ij} = \overline{(\mathbf{X}_i - \mu_{i1})(\mathbf{X}_j - \mu_{j1})};$$

$\mathbf{\Sigma}_2$ for component 2 is defined analogously.

To obtain a single lognormal in $N_c$ and $r_r$, rather than a mixture of two lognormals, we set the means and variances of $N_c$ equal for both mixture components as follows:

$$\mu_1(4) = \mu_2(4) \equiv \mu_{N_cG}, \tag{17}$$

and

$$\Sigma_1(4, 4) = \Sigma_2(4, 4) \equiv \sigma^2_{N_cG}, \tag{18}$$

and similarly for $r_r$.

Here, $\mu_{NcG}$ is the mean of the Gaussian variable $N_{c0} \ln(N_c/N_{c0})$. It is related to the mean ($\overline{N_c}$) and variance ($\overline{N_c'^2}$) of the lognormal variable $N_c$ by Garvey (2000)

$$\mu_1(4) = \mu_2(4) \equiv \mu_{N_cG} = N_{c0} \ln\left[\frac{\overline{N_c}}{N_{c0}}\left(1 + \frac{\overline{N_c'^2}}{\overline{N_c}^2}\right)^{-1/2}\right]. \tag{19}$$

Also, the variance of the Gaussian variable $N_{c0} \ln(N_c/N_{c0})$ is (Garvey 2000)

$$\Sigma_1(4, 4) = \Sigma_2(4, 4) \equiv \sigma^2_{N_cG} = N^2_{c0} \ln\left(1 + \frac{\overline{N_c'^2}}{\overline{N_c}^2}\right). \tag{20}$$

Finally we may write $G_k$, where $k = 1$ or 2, as (Johnson 1987, p. 50)

$$G_k = (2\pi)^{-d/2}|\Sigma_k|^{-1/2} \exp\left[-\frac{1}{2}(\mathbf{X} - \mu_{\mathrm{k}})^{\mathrm{T}}\Sigma_{\mathrm{k}}^{-1}(\mathbf{X} - \mu_{\mathrm{k}})\right]. \tag{21}$$

Equations (16)–(21) are written in abstract notation. To present the PDF more concretely, we present various marginal distributions of the full PDF. A marginal distribution of some variables is obtained by integrating the PDF over all the other variables. For instance, if we integrate over $N_c$ and $r_r$, we are left with the marginal distribution of $s$, $t$, and $w$. In the simplified binormal PDF that we use (Larson et al. 2002; Golaz et al. 2002), there is no within-component correlation of $w$ with $s$ or $t$. Then we may write for the $k$th mixture component,

$$\begin{aligned} G_k(s', t', w') = {} & \frac{1}{(2\pi)^{3/2}\sigma_{wk}\sigma_{sk}\sigma_{tk}(1 - r^2_{stk})^{1/2}} \\ & \times \exp\left\{-\frac{1}{2}\left[\frac{w' - (w_k - \overline{w})}{\sigma_{wk}}\right]^2\right\} \\ & \times \exp\left\{-\frac{1}{2(1 - r^2_{stk})}\left[\left(\frac{s' - (s_k - \overline{s})}{\sigma_{sk}}\right)^2\right.\right. \\ & + \left(\frac{t' - (t_k - \overline{t})}{\sigma_{tk}}\right)^2 - 2r_{stk}\left(\frac{s' - (s_k - \overline{s})}{\sigma_{sk}}\right) \\ & \left.\left. \times \left(\frac{t' - (t_k - \overline{t})}{\sigma_{tk}}\right)\right]\right\}. \end{aligned}$$

Here, $w_k$, $s_k$, and $t_k$ are the means of $w$, $s$, and $t$ for the $k$th mixture component. The quantities $\overline{w}$, $\overline{s}$, and $\overline{t}$ are grid box averages that include contributions from both mixture components, and $w'$, $s'$, and $t'$ are the deviations from the grid box averages. Also, $\sigma_{wk}$, $\sigma_{sk}$, and $\sigma_{tk}$ are the standard deviations of $w$, $s$, and $t$ for mixture component $k$. Furthermore, $-1 < r_{stk} < 1$ is the within-component correlation between $s$ and $t$.

The marginal distribution of $N_c$ is lognormal

$$\begin{aligned} P(N_c) = {} & \frac{1}{\sqrt{2\pi}(\sigma_{N_cG}/N_{c0})N_c} \\ & \times \exp\left\{-\frac{1}{2}\left[\frac{N_{c0} \ln(N_c/N_{c0}) - \mu_{N_cG}}{\sigma_{N_cG}}\right]^2\right\}. \end{aligned} \tag{22}$$

The marginal distribution of $s$ and $N_c$ for the $k$th mixture component is joint normal/lognormal (Garvey 2000)

$$\begin{aligned} P_k(s, N_c) = {} & \frac{1}{(2\pi)\sigma_{sk}(\sigma_{N_cG}/N_{c0})(1 - r^2_{sN_c})^{1/2}N_c} \\ & \times \exp\left\{-\frac{1}{2(1 - r^2_{sN_c})}\left[\left(\frac{s - s_k}{\sigma_{sk}}\right)^2\right.\right. \\ & + \left(\frac{N_{c0} \ln(N_c/N_{c0}) - \mu_{N_cG}}{\sigma_{N_cG}}\right)^2 \\ & - 2r_{sN_c}\left(\frac{s - s_k}{\sigma_{sk}}\right) \\ & \left.\left. \times \left(\frac{N_{c0} \ln(N_c/N_{c0}) - \mu_{N_cG}}{\sigma_{N_cG}}\right)\right]\right\}. \end{aligned}$$

Here $r_s N_c$ is the within-component correlation of $s$ and $N_{c0} \ln(N_c/N_{c0})$, assumed equal for both components.

*e. Transforming a uniform distribution to a binormal/lognormal mixture*

Now that we have found an adequate PDF family, the next step is to transform the Latin hypercube sample (3) from an independent, uniform distribution to our correlated, binormal/lognormal distribution. We want the transformed sample points to be spread out, as was the original uniform sample. Therefore, we choose a transformation method that does not scramble or eliminate any sample points but rather preserves their relative ordering. Specifically, we choose the so-called conditional distribution approach. The steps in this approach are the following (Johnson 1987, p. 43):

1) Find a value of $s$ using the marginal distribution of $s$.
2) Find a value of $t$ using the conditional distribution of $t$ given the value of $s$.
3) Find a value of $w$ using the conditional distribution of $w$ given the values of $s$ and $t$.

We iterate for as many variates as needed. This method reduces the problem of finding a sample from a multivariate PDF to the problem of finding a series of sample points from successive univariate PDFs.

To generate a point from a univariate PDF, we use the inverse cumulative distribution function method (Johnson 1987, 19–20). The cumulative distribution function (CDF), denoted $F(x)$, is the probability that a value of a variable, selected at random from the PDF $P(x)$, is less than $x$ (Boas 1983, p. 715). The CDF and PDF are related by

$$F(x) = \int_{\text{Lower limit}}^{x} P(x')\, dx'. \tag{23}$$

That is, $F(x)$ is the area under $P(x)$ to the left of $x$; $F(x)$ is strictly increasing from 0 to 1 for both binormal and lognormal PDFs.

To generate a variate with CDF $F$, we first take a Latin hypercube sample point $V$, which is drawn from a uniform distribution over the range (0, 1). Then we note that a random variate $X$ from the CDF of interest, $F$, is related to $V$ by

$$F(X) = V. \tag{24}$$

We invert the function $F$ to solve for what we want, $X$:

$$X = F^{-1}(V). \tag{25}$$

The transformation is illustrated graphically in Fig. 2. To generate our binormal/lognormal PDF, we need to use a numerical algorithm to compute the inverse of a Gaussian CDF, $F_G^{-1}$. We use that of Acklam (available online at http://home.online.no/~pjacklam).

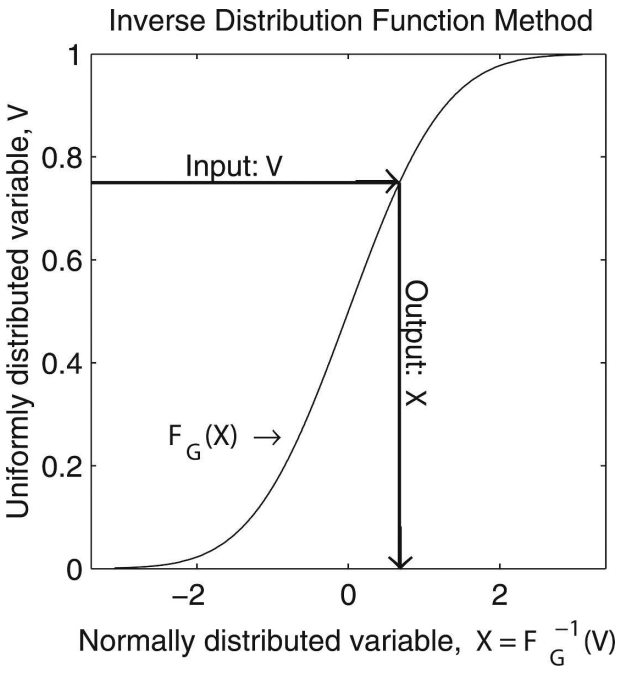


FIG. 2. An illustration of the inverse distribution function method for a single Gaussian PDF. First a point $V$ is chosen randomly from a uniform distribution. Then the cumulative distribution function $F_G$ is used to map $V$ into a point $X$ from a single Gaussian PDF.

Armed with the conditional distribution approach, we are prepared to select a value of $s$ from its marginal distribution, which is a univariate mixture of two Gaussians in $s$. First we must choose whether to draw the sample point from the first or second Gaussian mixture component. To do so, we use the $(d + 1)$th variate, denoted $MV$ for mixture variate, from the Latin hypercube sample. Here $MV$ is distributed uniformly on (0, 1). We choose the first mixture component if $MV$ is less than the probability of finding a point in the first component, namely, $aC_1/C$. Otherwise, we choose the second mixture component. Here $a$ is the mixture fraction [see Eq. (16)], $C_1$ is the fraction of component 1 that is cloudy, $C_2$ is the fraction of component 2 that is cloudy, and the total cloud fraction from both components is $C = aC_1 + (1 - a)C_2$. We select values from within cloud because autoconversion and many other microphysical processes occur exclusively in cloud. If $C = 0$, we do not select a sample point or call the autoconversion scheme for that grid box and time step.

Having chosen the mixture component, we need to select a point from the cloudy part of the Gaussian component PDF, $G(s)$. Mathematically, this means choosing from the part that has $s > 0$. Therefore, we are

choosing a point from a truncated Gaussian PDF, $G_t(s)$, which is related to $G(s)$ by

$$G_t(s) = \begin{cases} \frac{1}{C} G(s) & \text{if} \quad s > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{26}$$

The corresponding CDFs are related by

$$F_{G_t}(s) = \begin{cases} \frac{1}{C}[F_G(s) - (1 - C)] & \text{if} \quad s > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{27}$$

The relationship between $G$, $F_G$, and $F_{tG}$ is illustrated in Fig. 3. Since we want to transform our uniform sample point, $V$, to a truncated Gaussian sample point, $s$, we set $F_{Gt}(s) = V$. Substituting this into (27), we invert to find the desired sample point, $s$:

$$s = F_G^{-1}[VC + (1 - C)]. \tag{28}$$

It is trivial to adjust the scheme to choose points from both clear and cloudy regions: we merely keep the full Gaussian PDF, rather than truncating it. Then we find $s = F_G^{-1}(V)$.

Now we may choose $t$ using the conditional distribution of $t$, conditional on the value of $s$ we have just chosen. Afterward, we may choose a value of $w$ given $s$ and $t$. We continue in this way for the remaining variables.

To use this conditional distribution approach, before we can find the marginal and conditional distributions of a Gaussian-mixture PDF, we need to do so for a single-Gaussian PDF. We use the symbols $\mathbf{X} \sim \mathcal{N}_d(\mu, \Sigma)$ to denote a $d$-variate random sample $\mathbf{X}$ that is drawn from a single-normal $d$-variate PDF $\mathcal{N}_d$ with mean $\mu$ and covariance matrix $\mathbf{\Sigma}$. That is, $\mathbf{X}$ is a particular set of values, $[s, t, w, N_{c0} \ln(N_c/N_{c0}), r_{r0} \ln(r_r/r_{r0})]$, that is drawn randomly from a 5-dimensional single-Gaussian PDF, $\mathcal{N}_d$; $\mathcal{N}_d$ represents either of the two Gaussians, $G_1$ or $G_2$, that make up our double-Gaussian PDF (16). We divide the variables into a set consisting of $k$ variables, denoted by a subscript $\alpha$, and a set consisting of the remaining $d - k$ variables, denoted by subscript $\beta$. For instance, if $k = 2$, $\alpha$ corresponds to $s$ and $t$, and $\beta$ might correspond to the remaining variables, $w$, $N_{c0} \ln(N_c/N_{c0})$, and $r_{r0} \ln(r_r/r_{r0})$. We define the joint normal PDF as

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_\alpha \\ \mathbf{X}_\beta \end{bmatrix}, \tag{29}$$

with means

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_\alpha \\ \boldsymbol{\mu}_\beta \end{bmatrix}. \tag{30}$$

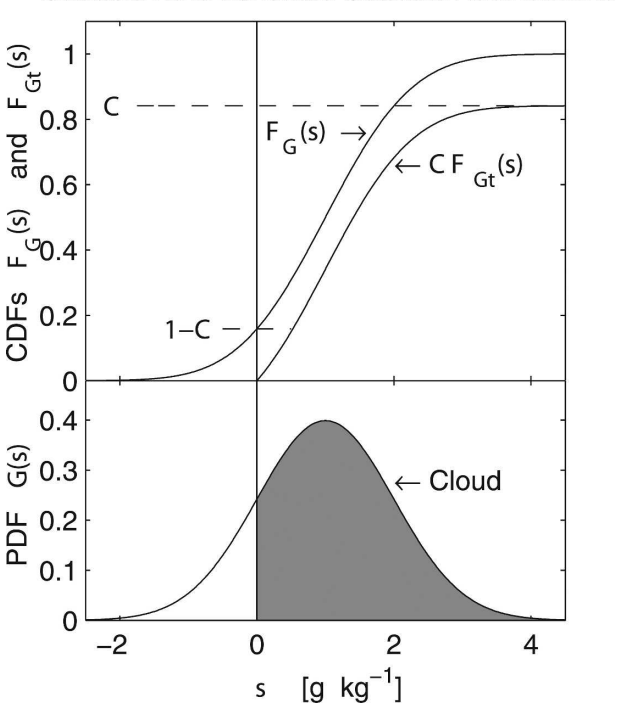


FIG. 3. A depiction of a single Gaussian PDF, $G(s)$, its corresponding cumulative distribution function, $F_G(s)$, and the corresponding cumulative distribution function associated with the cloudy part ($s > 0$), $F_{Gt}(s)$. Here $F_{Gt}(s)$ is a truncated Gaussian, and $C$ denotes cloud fraction.

So, for instance, if $\alpha = 2$, then $\boldsymbol{\mu}_\alpha$ is a column vector representing the first two means of the Gaussian, that is, $(s_1, t_1)$; $\boldsymbol{\mu}_\beta$ is a column vector representing the other means, that is, $[w_1, N_{c0} \ln(N_{c1}/N_{c0}), r_{r0} \ln(r_{r1}/r_{r0})]$. The covariance matrix is given by

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{\alpha\alpha} & \mathbf{\Sigma}_{\alpha\beta} \\ \mathbf{\Sigma}_{\beta\alpha} & \mathbf{\Sigma}_{\beta\beta} \end{bmatrix} \tag{31}$$

with dimensions

$$\begin{bmatrix} k \times k & k \times (d - k) \\ (d - k) \times k & (d - k) \times (d - k) \end{bmatrix}. \tag{32}$$

Then the marginal PDF of $\mathbf{X}_\alpha$ is $\mathcal{N}_k(\boldsymbol{\mu}_\alpha, \mathbf{\Sigma}_{\alpha\alpha})$. The marginal PDF of $\mathbf{X}_\beta$ is $\mathcal{N}_{d-k}(\boldsymbol{\mu}_\beta, \mathbf{\Sigma}_{\beta\beta})$ (Johnson 1987, p. 50). The conditional distribution of $\mathbf{X}_\beta$ given $\mathbf{X}_\alpha = \mathbf{x}_\alpha$ is

$$\mathcal{N}_{d-k}\big[\boldsymbol{\mu}_\beta + \mathbf{\Sigma}_{\beta\alpha}\,\mathbf{\Sigma}_{\alpha\alpha}{}^{-1}(\mathbf{x}_\alpha - \boldsymbol{\mu}_\alpha), \\ \mathbf{\Sigma}_{\beta\beta} - \mathbf{\Sigma}_{\beta\alpha}\,\mathbf{\Sigma}_{\alpha\alpha}{}^{-1}\mathbf{\Sigma}_{\alpha\beta}\big]. \tag{33}$$

We continue until we have drawn a complete sample of $[s, t, w, N_{c0} \ln(N_c/N_{c0}), r_{r0} \ln(r_r/r_{r0})]$ from a Gaussian mixture PDF. At this point, we exponentiate

$X_{N_c} = N_{c0} \ln(N_c/N_{c0})$ and $X_{r_r} = r_{r0} \ln(r_r/r_{r0})$ in order to generate lognormal distributions for $N_c$ and $r_c$. That is, we compute

$$N_c = N_{c0} \exp(X_{N_c}/N_{c0}) \quad r_r = r_{r0} \exp(X_{r_r}/r_{r0}) . \quad (34)$$

To summarize, we transform a uniform distribution to a binormal/lognormal mixture via the following steps. First, we select either the first or second Gaussian. Then, from this individual Gaussian, we choose a value of $s$ from within cloud using the inverse cumulative distribution function. Finally, we successively choose values of $[t, w, N_{c0} \ln(N_c/N_{c0}), r_{r0} \ln(r_r/r_{r0})]$ for this individual Gaussian using the aforementioned conditional distribution formulas for a single Gaussian.

### f. Computing a grid box average

The foregoing steps generate $n$ sample points, $(s_i, t_i, w_i, N_{ci}, r_{ri})$, for the current grid box and time step, out of a total of $n_t$ points in the sample. We need to input the $n$ points successively into a microphysical parameterization and find an average for the current time step. To choose the mixture component for *all n* points, we use the $(d + 1)$th variate, that is, the mixture variate, corresponding to the *first* of the $n$ sample points. Suppose the microphysical parameterization of interest is denoted $A$. Then the grid box average of $A$ is given by

$$\overline{A} = \frac{C}{n} \sum_{i=1}^{n} A(s_i, t_i, w_i, N_{ci}, r_{ri}). \quad (35)$$

Multiplying by cloud fraction $C$ ensures that we have a grid box average, not a within-cloud average.

In a prognostic cloud parameterization, it is important that water mass be conserved. This can be achieved with a Latin hypercube scheme if only rates (i.e., tendencies), such as autoconversion rates, are directly altered, and if whatever is added to drizzle mixing ratio is simultaneously removed from cloud mixing ratio, and similarly for other processes.

## 4. Results: Does Latin hypercube sampling perform better than traditional Monte Carlo sampling?

In this section, we present a preliminary test of Latin hypercube sampling. Namely, we investigate whether Latin hypercube sampling can lead to a more accurate calculation of autoconversion of cloud droplets to drizzle drops. Autoconversion formulas are typically highly nonlinear, and hence autoconversion modeling should benefit from an accurate representation of subgrid variability.

The parameterization we choose is the Kessler autoconversion formula, $A_K$ (Kessler 1969):

$$A_K = K(r_c - r_{\text{crit}})H(r_c - r_{\text{crit}}). \quad (36)$$

Here, $H$ is the Heaviside step function, equal to 1 when its argument is positive and 0 when negative. Here also, $K$ is an inverse time constant, set to $10^{-3}$ s$^{-1}$ (Grell et al. 1994). The constant $r_{\text{crit}}$ is a critical threshold below which no autoconversion occurs. The Kessler formula is piecewise linear, but because each piece differs in slope, it is nonlinear at $r_{\text{crit}}$. We set $r_{\text{crit}} = 0.3$ g kg$^{-1}$, a value within the range of liquid water mixing ratios encompassed by the clouds that we consider, so that our cloud cases "see" the nonlinearity.

The Kessler formula has the deficiency that it depends only on the cloud water mixing ratio, $r_c$, and not the droplet number concentration, $N_c$ (Cotton and Anthes 1989, p. 92). This is not a disadvantage for us, because our immediate interest is not accurate autoconversion prediction, but merely a simple nonlinear function with which to test Latin hypercube sampling. The main reason we use the Kessler formula, besides its ubiquity, is that it permits analytic solutions with which we can directly compare the numerical ones. However, because the Kessler formula depends on only one variable, it cannot be used to test how well Latin hypercube sampling handles two or more correlated variables. For correlated variables, theory predicts that Latin hypercube sampling will perform about as well as traditional Monte Carlo sampling.

For purposes of comparison, we need analytic formulas for two quantities that are obtained by integrating the Kessler formula or its square over the Gaussian-mixture PDF (16). The first is the grid box average Kessler autoconversion,

$$\overline{A_K} = aA_{K1} + (1 - a)A_{K2}, \quad (37)$$

where $a$ is the mixture fraction, and $A_{K1}$ and $A_{K2}$ are the autoconversion rates associated with plumes (i.e., mixture components) 1 and 2, given by

$$A_{Kk} = K \frac{\sigma_{sk}}{\sqrt{2\pi}} \exp\left[ -\frac{1}{2} \left( \frac{s_k - r_{\text{crit}}}{\sigma_{sk}} \right)^2 \right] + K(s_k - r_{\text{crit}})C_{r_{\text{crit}},k}. \quad (38)$$

The subscript $k = 1, 2$ denotes the plume. The quantity $C_{r_{\text{crit}},k}$ is analogous to the plume cloud fraction, $C_k$, except that it denotes the fraction of the $k$th plume that contains liquid water content greater than $r_{\text{crit}}$

$$C_{r_{\text{crit}},k} \equiv \frac{1}{2} \left\{ 1 + \text{erf}\left[ \frac{1}{\sqrt{2}} \left( \frac{s_k - r_{\text{crit}}}{\sigma_{sk}} \right) \right] \right\}. \quad (39)$$

The second quantity we need is the grid box standard deviation of Kessler autoconversion, defined here as

$$\mathrm{std}(A_K) = \sqrt{\mathrm{Var}(A_K)}. \tag{40}$$

Here the grid box centered variance of Kessler autoconversion, $\mathrm{Var}(A_K)$, is given by

$$\mathrm{Var}(A_K) = a[(A_{K1} - \overline{A_K})^2 + \mathrm{VarPlume}(A_{K1})] + (1 - a)[(A_{K2} - \overline{A_K})^2 + \mathrm{VarPlume}(A_{K2})]. \tag{41}$$

In this equation, VarPlume ($A_{Kk}$) is the within-plume centered variance associated with plume $k$,

$$\mathrm{VarPlume}(A_{Kk}) = K(s_k - r_{\mathrm{crit}})A_{Kk} + K^2\sigma_{sk}^2 C_{r_{\mathrm{crit}},k} - A_{Kk}^2. \tag{42}$$

For completeness, we list the formula for the within-cloud centered variance of Kessler autoconversion

$$\overline{A_K^2}^{\mathrm{cld}} - (\overline{A_K}^{\mathrm{cld}})^2 = \frac{\mathrm{Var}(A_K) + (\overline{A_K})^2}{C} - \frac{(\overline{A_K})^2}{C^2}. \tag{43}$$

Here the within-cloud average Kessler autoconversion is

$$\overline{A_K}^{\mathrm{cld}} = \frac{\overline{A_K}}{C}. \tag{44}$$

To test the Latin hypercube method, we have seen that we need a prototypical nonlinear formula (we have chosen the Kessler formula). We also need plausible surrogate samples of atmospheric subgrid variability. For this we use large-eddy simulations (LES). We examine simulations of two very different boundary layer cloud regimes: the Barbados Oceanographic and Meteorological Experiment (BOMEX) trade wind cumulus case (Siebesma et al. 2003), and the First International Satellite Cloud Climatology Project (ISCCP) Regional Experiment (FIRE) stratocumulus case (Moeng et al. 1996). These simulations were set up according to Global Energy and Water Experiment (GEWEX) Cloud System Study (GCSS) intercomparison specifications, which are based loosely on observations. The observations showed little drizzle formation, and the simulations deliberately shut off drizzle and all other precipitation processes. Nonetheless, it is still reasonable to use these cases to test the Latin hypercube simulations, because they contain realistic variability and because we can still diagnostically compute what the autoconversion rate would have been had it been turned on in the simulation.

Our test of Latin hypercube sampling is diagnostic. That is, we use the LES to compute offline all the parameters needed to specify the binormal/lognormal PDF for each time of interest and altitude in the LES domain. The parameters are the mixture fraction, $a$, and the means and covariance matrix for each individual Gaussian. The formula we test depends on only two variables, $r_t$ and $\theta_l$, which leads to 11 independent parameters. These are obtained as in Larson and Golaz (2005). For instance, $a$ is obtained from the skewness of vertical velocity, $\mathrm{Sk}_w = \overline{w'^3}/(\overline{w'^2})^{3/2}$:

$$a = \frac{1}{2}\left\{1 - \mathrm{Sk}_w\left[\frac{1}{4(1 - \tilde{\sigma}_w^2)^3 + \mathrm{Sk}_w^2}\right]^{1/2}\right\}, \tag{45}$$

where $\tilde{\sigma}_w^2 \cong 0.4$.

Given each PDF, we compute Kessler autoconversion analytically and estimate it using Monte Carlo and Latin hypercube sampling. We use LES instead of observational data because LES readily provides all the needed input moments.

The Naval Research Laboratory's Coupled Ocean/Atmosphere Mesoscale Prediction System (COAMPS®; Hodur 1997) is the basis of the LES model we use, COAMPS–LES (Golaz et al. 2005). COAMPS–LES solves the compressible equations of motion (Klemp and Wilhelmson 1978) and includes a turbulent kinetic energy (TKE) subgrid-scale model (Deardorff 1980). Second-order advection is used for the momentum variables, and a positive definite advection scheme with second-order polynomial interpolation (Bott 1989) is used for the scalars. This LES model has compared well with results from several GCSS intercomparisons of boundary layer stratocumulus and cumulus layers, including some not discussed here (Brown et al. 2002; Stevens et al. 2001, 2005; Golaz et al. 2005). Both the BOMEX and FIRE cases were run with specified or idealized radiative cooling. For BOMEX, the grid box size is 100 m × 100 m × 40 m and the domain size is 6400 m × 6400 m × 3000 m. For FIRE, the grid box size is 50 m × 50 m × 25 m and the domain size is 3000 m × 3000 m × 1200 m.

Our results are shown in Figs. 4, 5, and Table 1. These are based on several calculated quantities. The first (thick solid line) is analytic calculations of grid box mean (37) and standard deviation (40) of Kessler autoconversion. The second (circles) is estimates of these quantities using the traditional Monte Carlo method, with one sample point per grid box and time step. Each new sample point is drawn independently of the prior ones. The third (triangles) is a Latin hypercube estimate with 12 total sample points ($n_t = 12$), with one sample point used per grid box and time step ($n = 1$). After 12 time steps, a new batch of sample points is chosen. This method does not call the autoconversion parameterization any more often than traditional

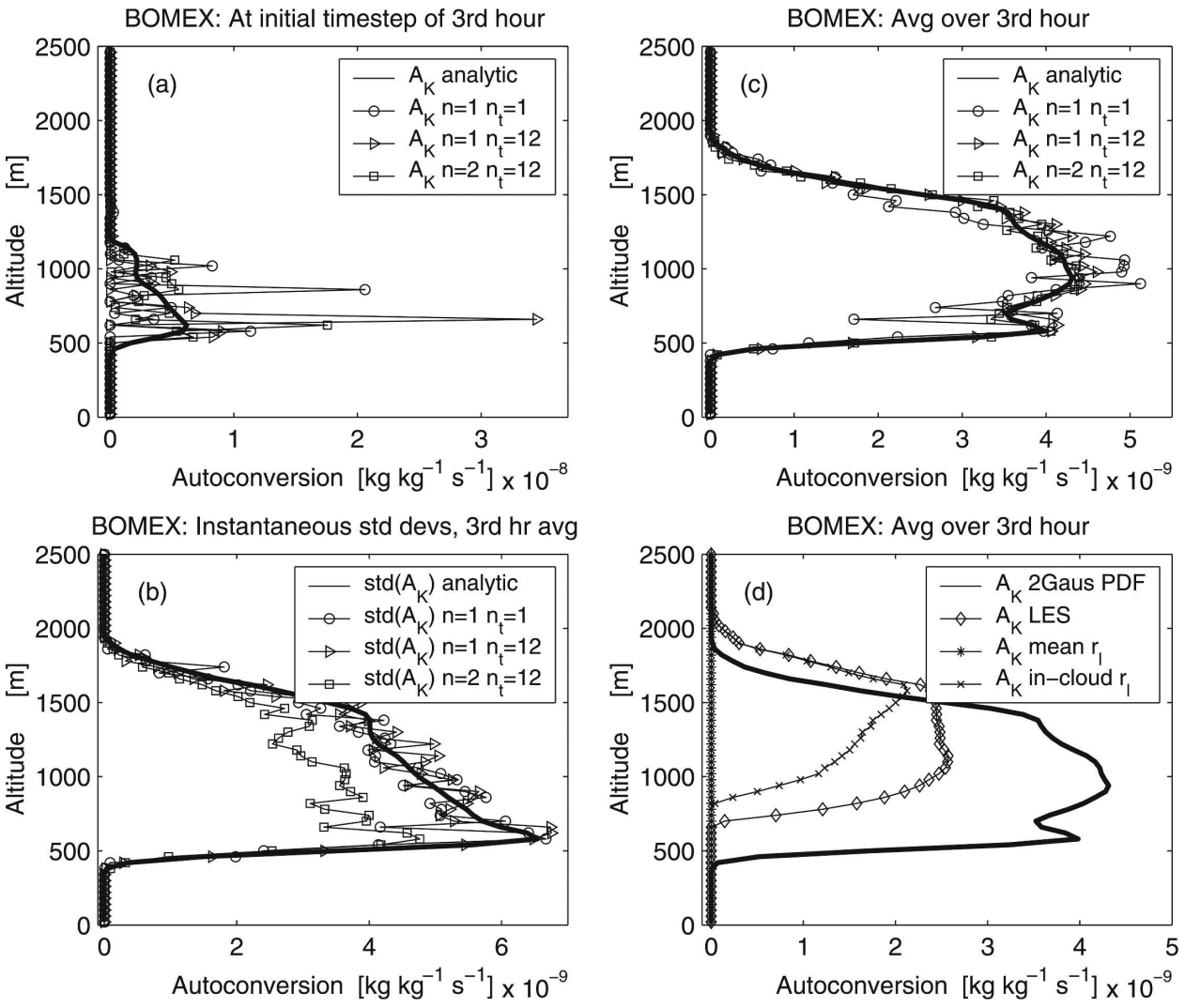


FIG. 4. Estimates of grid box average Kessler autoconversion for the BOMEX cumulus simulation. Estimates are based on analytic Eqs. (37) and (40) (thick solid line), traditional Monte Carlo ($n = 1$, $n_t = 1$) (circles), Latin hypercube sampling with ($n = 1$, $n_t = 12$) (triangles), and Latin hypercube sampling with ($n = 2$, $n_t = 12$) (squares). Here $n$ denotes the number of sample points per grid box and time step; $n_t$ denotes the total number of points in the Latin hypercube sample. (a) An *instantaneous snapshot* of the estimates: using two sample points per grid box and time step ($n = 2$, squares) reduces the noise somewhat. (b) The std devs: using two sample points gives lower values, i.e., reduces noise, as desired. (c) Time-averaged autoconversion: a Latin hypercube sample ($n = 1$, $n_t = 12$) (triangles) is less noisy than traditional Monte Carlo (circles), with no additional calls to the microphysics. (d) The "true" LES Kessler autoconversion (diamonds) along with analytic approximations based on a binormal PDF (thick solid line), within-cloud liquid water (x marks), and grid box average liquid water (asterisks); the binormal PDF approximation is somewhat in error, and the within-cloud and grid box average liquid water give underestimates, as expected.

Monte Carlo. The fourth calculated quantity (squares) is a Latin hypercube estimate with 12 total sample points ($n_t = 12$), but with two sample points used per grid box and time step ($n = 2$), thereby doubling the computational cost associated with microphysics.

First consider the BOMEX cumulus results, presented in Fig. 4. Figure 4a presents a snapshot of Kessler autoconversion at one time step. There is large variability for Monte Carlo and Latin hypercube with $n = 1$, with autoconversion rates ranging from 0 to 4 times the analytic value. We find, as expected, that for a particular snapshot, a Latin hypercube sample with $n = 1$ is no less noisy than a Monte Carlo sample for a particular snapshot; it is only after averaging at least $n_t = 12$ time steps that Latin hypercube with $n = 1$ reduces the noise. Less noisy is the Latin hypercube sample with $n = 2$ points per grid box and time step, although it still exhibits considerable noise.

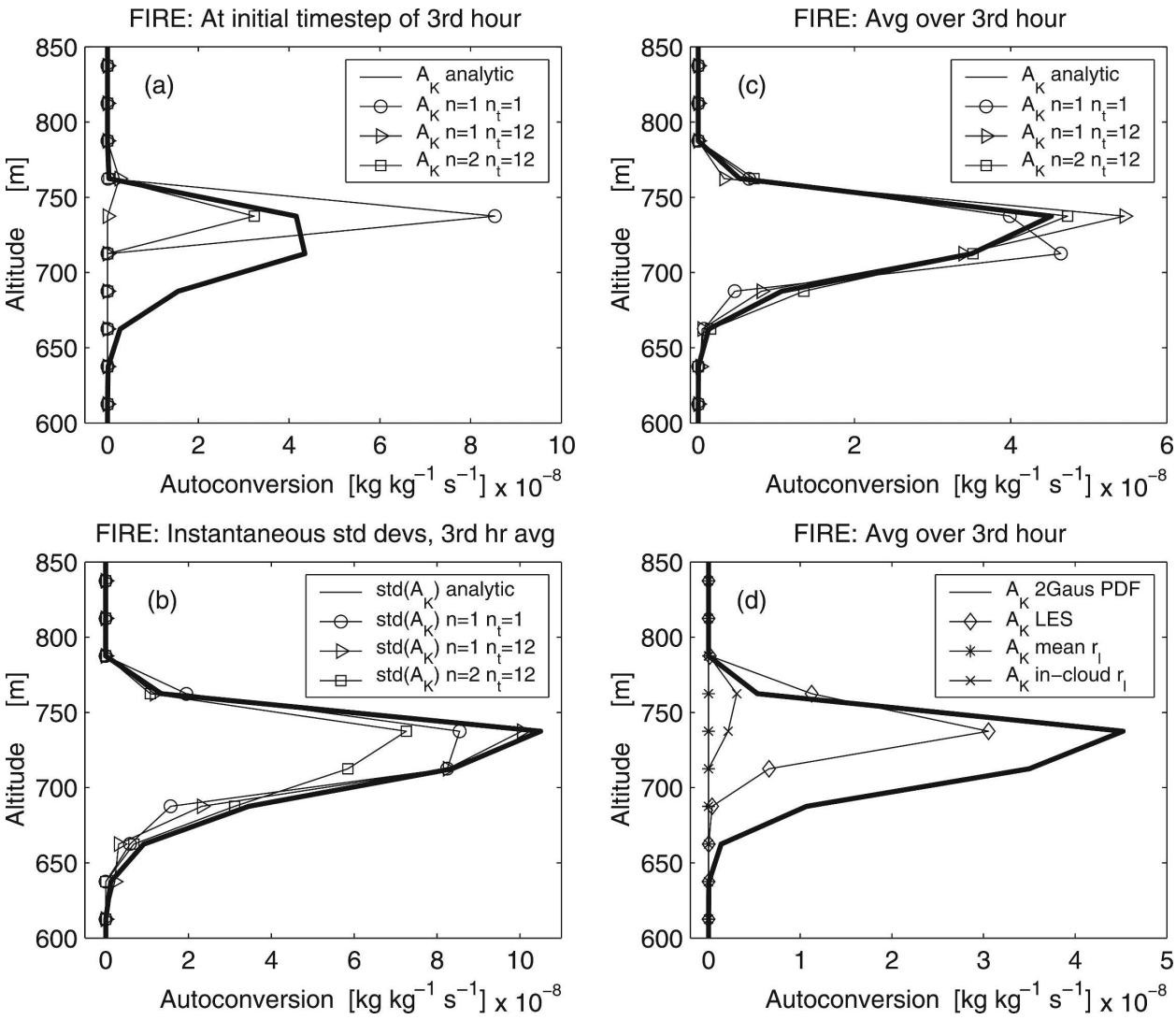


FIG. 5. Same as in Fig. 4, but for the FIRE stratocumulus simulation.

Figure 4b, which shows instantaneous standard deviations about the mean, tells a similar story. These deviations represent variability from time step to time step; small values are desirable. The Monte Carlo and $n = 1$ Latin hypercube standard deviations match the analytic Eq. (40). The $n = 2$ Latin hypercube sample has a lower standard deviation, demonstrating its reduced instantaneous noise, in conformity with Fig. 4a.

Figure 4c displays a 1-h average of autoconversion. Sixty snapshots are included in the average, with one snapshot per minute. Even after this time period, the Monte Carlo method (circles) exhibits considerable noise. Noticeably less noisy is the $n = 1$ Latin hypercube sample, and even less noisy is the $n = 2$ Latin hypercube sample.

Figure 4d addresses the question of whether the solution to which Monte Carlo and Latin hypercube methods converge is accurate. When using these methods, before drawing a random sample, first we must fit a binormal/lognormal PDF to the "true" PDF produced by the LES. If the binormal/lognormal PDF is a poor fit, the end result will be poor, regardless of how well we choose the sample points. Figure 4d shows that the PDF fit is moderately accurate but could use improvement. Specifically, the Kessler autoconversion calculated analytically from the fitted binormal/lognormal PDF (solid line) exceeds that computed directly from the true LES PDF (diamonds). This is because of errors in the shape of the assumed PDF family. Also plotted is Kessler autoconversion computed using Eq. (36), but with the local cloud water $r_c$ replaced by the grid box

TABLE 1. Errors in Kessler autoconversion as computed by traditional Monte Carlo (TMC) ($n = 1$, $n_t = 1$), Latin hypercube (LH) sampling with one sample point per altitude and time step ($n = 1$, $n_t = 12$), and Latin hypercube sampling with two sample points per altitude and time step ($n = 2$, $n_t = 12$). The latter has double the computational cost of the former two. Errors are computed based on LES of the BOMEX cumulus case and the FIRE stratocumulus case. Errors are root-mean-square deviations from the analytic grid box average autoconversion rates given by (37). The errors are averaged over the third hour of the simulations and the altitudes that contain some autoconversion. Instantaneous deviations are instantaneous rms errors that are subsequently time- and vertically averaged. Time-averaged deviations are computed by averaging autoconversion over an hour, calculating the deviation from the analytic autoconversion rate averaged over the same hour, taking the absolute values, and vertically averaging. Numbers in parentheses are normalized by the traditional Monte Carlo values. The time-averaged deviations are lower for LH sampling than for TMC sampling. The instantaneous deviations are lower only when there are $n = 2$ sample points per altitude and time step, as expected. Errors are lower for BOMEX because cloud fraction in BOMEX is small.

| | BOMEX | | FIRE | |
|---|---|---|---|---|
| | Time-averaged deviations ($10^{-9}$ s$^{-1}$) | Instantaneous deviations ($10^{-9}$ s$^{-1}$) | Time-averaged deviations ($10^{-9}$ s$^{-1}$) | Instantaneous deviations ($10^{-9}$ s$^{-1}$) |
| TMC ($n = 1$, $n_t = 1$) | 0.45 (1) | 3.4 (1) | 4.1 (1) | 35 (1) |
| LH ($n = 1$, $n_t = 12$) | 0.15 (0.33) | 3.6 (1.1) | 2.7 (0.66) | 37 (1.1) |
| LH ($n = 2$, $n_t = 12$) | 0.10 (0.22) | 2.5 (0.74) | 1.2 (0.29) | 30 (0.86) |

average cloud water $\overline{r_c}$ (asterisks) or the within-cloud average cloud water $\overline{r_c}^{\text{cld}}$ (x marks). It can be proven that both these approximations systematically underpredict the true autoconversion (Larson et al. 2001), as illustrated in Fig. 4d. In fact, using grid box average cloud water $\overline{r_c}$ (asterisks) yields zero everywhere, because cloud fraction is small and therefore $\overline{r_c} \ll r_{\text{crit}}$. However, some autoconversion would occur if we were to reduce the autoconversion threshold, $r_{\text{crit}} = 0.3$ g kg$^{-1}$. As an aside, we note that *prognosing* $\overline{r_c}^{\text{cld}}$ accurately is also difficult (Larson 2004).

Results from the FIRE stratocumulus case are shown in Fig. 5. The overall pattern is similar to that for BOMEX: there is considerable variability for a given snapshot, but time averaging reduces the variability more for Latin hypercube sampling than for traditional Monte Carlo sampling. The improvement, however, is milder than for BOMEX. Substituting the average within-cloud cloud water into the Kessler formula (x marks) leads to a severe underestimate of the true autoconversion (diamonds) because $\overline{r_c}^{\text{cld}} < r_{\text{crit}}$. A less severe underestimate would occur if we were to lower the critical threshold for autoconversion, here taken to be $r_{\text{crit}} = 0.3$ g kg$^{-1}$.

Table 1 quantifies the above discussion. Shown are error estimates averaged over an hour of time (60 snapshots, each separated by 1 min) and over the vertical extent of the cloud. Latin hypercube sampling is more accurate than traditional Monte Carlo, noticeably so for BOMEX and less so for FIRE.

We now compare the reduction of variance produced by Latin hypercube sampling versus the optimal spectral sampling technique of Räisänen and Barker (2004). Optimal spectral sampling exploits physical insight about atmospheric radiation to reduce noise in radiative transfer calculations. In the examples listed in Table 1 of Räisänen and Barker (2004), optimal spectral sampling costs 50% more than traditional Monte Carlo but, impressively, roughly halves instantaneous standard deviations as compared with Monte Carlo sampling from the cloud alone. In contrast, our Latin hypercube sample with ($n = 2$, $n_t = 12$) costs double and only reduces *instantaneous* deviations by a factor of 1.2 to 1.4. However, our Latin hypercube sample with ($n = 1$, $n_t = 12$) cost negligibly more than traditional Monte Carlo from cloud points and reduces *time-averaged* deviations by a factor of 1.5 to 3. Latin hypercube sampling with ($n = 2$, $n_t = 12$) reduces the time-averaged noise further. Again, this shows that Latin hypercube sampling reduces time-averaged noise more effectively than instantaneous noise. Räisänen and Barker (2004) demonstrate that physical insight can be used very effectively to reduce variance for radiative transfer. It is our hope that Latin hypercube sampling can benefit *microphysical* calculations without requiring knowledge of the details of the physics. For microphysical calculations, optimal spectral sampling is not directly applicable.

## 5. Summary and conclusions

This paper has addressed a particular problem in numerical modeling of cloud microphysics: namely, that of driving a local, nonlinear microphysical parameterization when the model resolution is coarse. When this problem arises, inaccurate results are generated by simply driving the microphysical parameterization with grid box averages of model-produced fields, such as liquid water content. Rather, the host model needs to feed the microphysical parameterization information about subgrid variability.

The particular solution we have investigated is a Monte Carlo method called Latin hypercube sampling (McKay et al. 1979; OW03). Compared with traditional Monte Carlo sampling, Latin hypercube sampling accomplishes a reduction of variance by choosing sample points quasi randomly, rather than entirely randomly.

Microphysics may be computed using the Latin hypercube algorithm as follows. The input is information about the distribution of fields within a particular grid box at a particular time step. The output is an estimate of the grid box average of a microphysical process at that time step. The steps of the algorithm are

1) Choose points from Latin hypercube sample for a uniform distribution [Eq. (3)].
2) Transform the uniformly distributed sample points to the actual distribution that represents the grid box's subgrid variability (section 3e).
3) Input the transformed sample points into the microphysics parameterization; that is, call the microphysics subroutine once for each of the sample points (section 3f).
4) Average the results of the microphysical calculations to yield a grid box average (section 3f).

All steps are then repeated for all other grid boxes and time steps.

Latin hypercube sampling has the following advantage over traditional Monte Carlo sampling:

- *It can reduce statistical noise inherent in Monte Carlo methods.* Latin hypercube sampling spreads a sample of $n_t$ points throughout the sample space so that the points do not, by random chance, cluster in one subvolume. Therefore, averages over $n_t$ Latin hypercube points are often more accurate than averages over $n_t$ randomly sampled points.

The exception occurs when variables are strongly correlated. Then Latin hypercube sampling is unlikely to be more accurate than random sampling (Press et al. 1992), but it is also guaranteed to be not much less accurate (Owen 1997). Our sampling method can handle correlations because we have chosen a general yet tractable joint PDF family. This is important because many microphysical processes depend on such correlations. For instance, activation of cloud condensation nuclei depends jointly on $w$ and $r_t$.

Latin hypercube sampling, in conjunction with our joint binormal/lognormal PDF, shares the following advantages and disadvantages with traditional Monte Carlo sampling:

- *It is compatible with a large variety of local parameterizations, both numerical and analytic.* This makes it possible, with a single interface, to account for subgrid variability in many local processes including those in microphysics and even potentially atmospheric chemistry. This generality is made possible by the fact that the Latin hypercube method does not rely on properties of a particular parameterization to reduce variance. This is useful for numerical models, such as the Weather Research and Forecasting model, that contain many options for microphysics parameterization. These existing parameterizations would require little, if any, modification to be used with Latin hypercube sampling. Additionally, it may be possible to combine Latin hypercube sampling with other methods that exploit the physics of a particular parameterization to reduce variance. Nonlocal processes like radiative transfer cannot be handled with the algorithm in this paper unless an assumption about cloud overlap is made.
- *It can be extended to any number of variates.* The number of input variables differs for different microphysics parameterizations. Some microphysics parameterizations require the input of only one predicted "hydrometeor," cloud/ice water content (Tiedtke 1993). Other microphysical parameterizations require the prediction of many hydrometeor categories such as cloud droplets, raindrops, pristine ice, graupel, and so on (Cotton et al. 2003). For ease of exposition, the present paper has discussed a 5-variate input sample in total water mixing ratio $r_t$, liquid water potential temperature $\theta_l$, vertical velocity $w$, droplet number mixing ratio $N_c$, and drizzle mixing ratio $r_r$. However, this can be easily generalized to more variables because our assumed binormal/lognormal PDF is sufficiently simple.
- *It can sample either within-cloud areas exclusively, or both cloudy and clear areas.* It is wasteful to choose sample points in clear areas for processes like autoconversion that occur only in cloud. Therefore, in this paper, we have sampled only the cloudy portion of a grid box. However, for processes that occur in both cloudy and clear areas, our algorithm can be easily adapted.
- *It allows the modeler to choose the PDF family.* Different PDFs may be suited to different physical processes. If the joint binormal/lognormal PDF that we discuss is not adequate for a particular application, a more appropriate PDF may be substituted with relative ease.
- *It allows the modeler to choose the number of calls to the microphysics per grid box and time step.* The size and implementation of the Latin hypercube sample can be varied based on the computational resources available and the importance of reducing statistical noise. For instance, suppose one chooses to use a

12-point Latin hypercube sample. If the microphysical parameterization is computationally expensive and the subgrid variability small, one may select 1 point from the 12-point sample for each time step, and repeat after 12 time steps. If the parameterization is cheap and the variability large, one may instead select 2 points per time step and thereby complete the 12-point sample in 6 time steps.

We tested Latin hypercube sampling against large-eddy simulations (LES). We simulated two different boundary layer cloud regimes: the BOMEX trade wind cumulus case, and the FIRE stratocumulus case. Our tests were diagnostic: that is, we used LES output to fix the PDF at each altitude and time of interest, and then, from these PDFs, we drew Latin hypercube and Monte Carlo samples. We used these samples to compute the Kessler autoconversion rate from cloud droplets to drizzle drops. Since the Kessler formula depends only one variable (cloud water), it cannot be used to test sampling of correlated variables, although for correlated formulas, theory predicts that Latin hypercube and Monte Carlo sampling perform comparably. We chose Kessler autoconversion as a test formula because it is widely used and, importantly, is analytically tractable.

In our first Latin hypercube sampling strategy, we chose 12 sample points, and selected one sample point per altitude and time step. This involves no additional calls to the microphysics beyond traditional Monte Carlo sampling. Nevertheless, the time-averaged Latin hypercube estimates were better by a factor of 3 for BOMEX and a factor of 1.5 for FIRE. However, the noise in the instantaneous profiles was not ameliorated, as expected. In simulations in which this level of noise is problematic, one may maintain the 12-point Latin hypercube sample, but choose two sample points per altitude and time step, which doubles the microphysical cost. We found that this reduces the instantaneous noise somewhat, although instantaneous autoconversion rates can still range from zero to three times the true value. Whether or not an interactive simulation can accept this magnitude of noise cannot be determined by our diagnostic tests.

In summary, we found Latin hypercube sampling to be a very general method that offers the prospect of moderately reducing the time-averaged sampling noise in Monte Carlo estimates at negligible extra computational cost.

*Acknowledgments.* V. E. Larson is grateful for financial support provided by Grant ATM-0239982 from the National Science Foundation, and by subaward G-7424-1 from the DoD Center for Geosciences/Atmospheric Research at Colorado State University via Cooperative Agreement DAAD19-02-2-0005 with the Army Research Laboratory. William Cotton and Hongli Jiang acknowledge financial support for this research from the National Science Foundation under Grant ATM-0215367. This research was performed while J.-C. Golaz held a National Research Council Research Associateship Award at the Naval Research Laboratory, Monterey, California. COAMPS® is a registered trademark of the Naval Research Laboratory.

## REFERENCES

Boas, M. L., 1983: *Mathematical Methods in the Physical Sciences.* 2d ed. John Wiley and Sons, 793 pp.

Bott, A., 1989: A positive definite advection scheme obtained by nonlinear renormalization of the advective fluxes. *Mon. Wea. Rev.,* **117,** 1006–1015.

Brown, A. R., and Coauthors, 2002: Large-eddy simulation of the diurnal cycle of shallow cumulus convection overland. *Quart. J. Roy. Meteor. Soc.,* **128,** 1075–1093.

Cotton, W. R., and R. A. Anthes, 1989: *Storm and Cloud Dynamics.* Academic Press, 884 pp.

——, and Coauthors, 2003: RAMS 2001: Current status and future directions. *Meteor. Atmos. Phys.,* **82,** 5–29.

Deardorff, J. W., 1980: Stratocumulus-capped mixed layers derived from a three-dimensional model. *Bound.-Layer Meteor.,* **18,** 495–527.

Devroye, L., 1986: *Non-Uniform Random Variate Generation.* Springer-Verlag, 843 pp.

Garvey, P. R., 2000: *Probability Methods for Cost Uncertainty Analysis.* Marcel Dekker, 401 pp.

Gentle, J. E., 2003: *Random Number Generation and Monte Carlo Methods.* 2d ed. Springer, 381 pp.

Golaz, J.-C., V. E. Larson, and W. R. Cotton, 2002: A PDF-based model for boundary layer clouds. Part I: Method and model description. *J. Atmos. Sci.,* **59,** 3540–3551.

——, S. Wang, J. D. Doyle, and J. M. Schmidt, 2005: Second and third moment vertical velocity budgets derived from COAMPS-LES. *Bound.-Layer Meteor.,* in press.

Grell, G. A., J. Dudhia, and D. R. Stauffer, 1994: A description of the fifth-generation Penn State/NCAR Mesoscale Model (MM5). Tech. Rep. NCAR TN-398 STR, 138 pp.

Hodur, R. M., 1997: The Naval Research Laboratory's Coupled Ocean/Atmosphere Mesoscale Prediction System (COAMPS). *Mon. Wea. Rev.,* **125,** 1414–1430.

Johnson, M. E., 1987: *Multivariate Statistical Simulation.* John Wiley and Sons, 230 pp.

Kessler, E., 1969: *On the Distribution and Continuity of Water Substance in Atmospheric Circulation. Meteor. Monogr.,* No. 10, Amer. Meteor. Soc., 84 pp.

Klemp, J. B., and R. B. Wilhelmson, 1978: The simulation of three-dimensional convective storm dynamics. *J. Atmos. Sci.,* **35,** 1070–1095.

Larson, V. E., 2004: Prognostic equations for cloud fraction and liquid water, and their relation to filtered density functions. *J. Atmos. Sci.,* **61,** 338–351.

——, and J.-C. Golaz, 2005: Using probability density functions to derive consistent closure relationships among higher-order moments. *Mon. Wea. Rev.,* **133,** 1023–1042.

——, R. Wood, P. R. Field, J.-C. Golaz, T. H. Vonder Haar, and

W. R. Cotton, 2001: Systematic biases in the microphysics and thermodynamics of numerical models that ignore subgrid-scale variability. *J. Atmos. Sci.,* **58,** 1117–1128.

——, J.-C. Golaz, and W. R. Cotton, 2002: Small-scale and mesoscale variability in cloudy boundary layers: Joint probability density functions. *J. Atmos. Sci.,* **59,** 3519–3539.

L'Ecuyer, P., 1988: Efficient and portable combined random number generators. *Commun. ACM,* **31,** 742–774.

Lewellen, W. S., and S. Yoh, 1993: Binormal model of ensemble partial cloudiness. *J. Atmos. Sci.,* **50,** 1228–1237.

McKay, M. D., R. J. Beckman, and W. J. Conover, 1979: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics,* **21,** 239–245.

Mellor, G. L., 1977: The Gaussian cloud model relations. *J. Atmos. Sci.,* **34,** 356–358.

Moeng, C.-H., and Coauthors, 1996: Simulation of a stratocumulus-topped planetary boundary layer: Intercomparison among different numerical codes. *Bull. Amer. Meteor. Soc.,* **77,** 261–278.

Owen, A. B., 1997: Monte Carlo variance of scrambled equidistribution quadrature. *SIAM J. Numer. Anal.,* **34,** 1884–1910.

——, 2003: Quasi–Monte Carlo techniques. *Siggraph 2003*, *Course 44*, San Diego, CA, Association for Computing Machinery.

Pincus, R., and S. A. Klein, 2000: Unresolved spatial variability and microphysical process rates in large-scale models. *J. Geophys. Res.,* **105,** 27 059–27 065.

——, H. W. Barker, and J.-J. Morcrette, 2003: A fast, flexible, approximate technique for computing radiative transfer in inhomogeneous cloud fields. *J. Geophys. Res.,* **108,** 4376, doi:10.1029/2002JD003322.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, 1992: *Numerical Recipes in C: The Art of Scientific Computing.* 2d ed. Cambridge University Press, 994 pp.

Räisänen, P., and H. W. Barker, 2004: Evaluation and optimization of sampling errors for the Monte Carlo Independent Column Approximation. *Quart. J. Roy. Meteor. Soc.,* **130,** 2069–2085.

——, ——, M. F. Khairoutdinov, J. Li, and D. A. Randall, 2004: Stochastic generation of subgrid-scale cloudy columns for large-scale models. *Quart. J. Roy. Meteor. Soc.,* **130,** 2047–2067.

Rotstayn, L. D., 2000: On the "tuning" of autoconversion parameterizations in climate models. *J. Geophys. Res.,* **105,** 15 495–15 507.

Siebesma, A. P., and Coauthors, 2003: A large eddy simulation intercomparison study of shallow cumulus convection. *J. Atmos. Sci.,* **60,** 1201–1219.

Sommeria, G., and J. W. Deardorff, 1977: Subgrid-scale condensation in models of nonprecipitating clouds. *J. Atmos. Sci.,* **34,** 344–355.

Stevens, B., and Coauthors, 2001: Simulations of trade wind cumuli under a strong inversion. *J. Atmos. Sci.,* **58,** 1870–1891.

——, and Coauthors, 2005: Evaluation of large-eddy simulations via observations of nocturnal marine stratocumulus. *Mon. Wea. Rev.*, **133**, 1443–1462.

Tiedtke, M., 1993: Representation of clouds in large-scale models. *Mon. Wea. Rev.,* **121,** 3040–3061.