Projective Integration Methods for Distributions.*

C. W. Gear†

November 26, 2001

Abstract

Projective methods were introduced in an earlier paper. In this paper we consider their application to the output of simulations of a collection of randomly moving objects. In the limit (of a large number of these objects) the solution would be a time varying probability distribution of the objects in space. A key idea in this paper is the representation of the probability distribution, a representation that permits the application of the projective method.

Keywords Integration, probability density functions, cumulative distribution functions

1 Introduction

The motivation for this paper comes from some biological models, but the method is applicable to a large class of problems. The biological models of interest have the following general structure. The system consists of a set of cells. Each cell, C_i, has an internal state, S_i that evolves following a differential equation whose driving term(s) are the values of external stimuli. Part of the internal state is whether the cell is moving or not, and if it is moving, in which direction it is moving. (This is often assumed to be a constant direction until there is a discrete change.) There is some probability that the cell will change the part of its state concerned with the motion, either by stopping if it is currently moving or starting to move (usually in some random direction) if it is stationary. The length of time that it remains in a particular moving state is dependent on other internal state variables, and can thus be influenced not only by the value of external stimuli, but their time gradient *as observed by the moving cell.* Clearly such a mechanism can bias the random walk taken by the cell in favor of preferring directions in which certain stimuli increase (or decrease). Models for this are discussed in [4] and [5].

In earlier papers ([1], [2]) we considered projective methods for stiff problems with gaps in their spectra. In the projective method, a numerical solution is computed at a

*The author would like to acknowledge helpful discussions with Ioannis Kevrekidis and Sima Setayeshgar of Princeton.

†NEC Research Institute, retired

sequence of relatively closely spaced points (in time) and then a "giant" step is taken using polynomial extrapolation from the last few of those computed points. The extrapolation computation is explicit and fast, and its purpose was to use large, explicit steps even in the presence of stiffness. In [3] we applied the projective method to problems that were described at a "microscopic" level, that is, by a high-dimensional model requiring a small time step. The projective integration step was then done on a *restriction* of the solution of the microscopic model to a macroscopic model in a much lower dimension. An example would be the restriction of a Lattice Boltzman model of a system to one involving velocities and densities on, for example, a finite element discretization.

In this paper we apply the projective methods to restrictions of the results of a stochastic simulation of moving particles. Because the objective of this paper is to examine the numerical method, and not the physical problem, a particularly simple biased random walk is chosen as the model problem. This is described in the next section.

The simulation of the biased random walk yields a finite distribution of positions at each time step. This has to be represented in some compact (low-dimensional) manner by a restriction operator so that the projective method can be applied. It is natural to think in terms of probability density functions (PDFs), but it is difficult to get "reasonable" functions from discrete samples without a priori knowledge of the nature of the distribution - something we do not want to require. For plotting purposes, one usually resorts to histograms. These are either to coarse (if few bins are used) or too jagged (if many bins are used) for projective integration. What we can plot easily is the cumulative distribution function (CDF). Suppose that the simulation of N cells yields the set of positions $\mathbf{x} = \{x_i\}, i = 1, 2, \cdots, N$ for the cells at time t_n. Assume that these have been sorted into ascending order ($x_i \leq x_{i+1}$). Simply plot the graph consisting of the points $(x_i, (i - 0.5)/N), i = 1, \cdots, N$. Any sort of interpolation can be used between the points, but if N is not small - say 500 or more - the graphical presentation could use just the points or a simple linear interpolant. As we will see, we will not need to perform any sort of interpolation in the algorithm to be presented, so interpolation does not introduce accuracy issues.

The reader can easily see that this produces a reasonable approximation to a CDF by executing the Matlab statement

plot(sort(randn(1,N)),((1:N)-0.5)/N)

having set N to some largish value such as 1,000. The result is good in an error norm such as L_2 or L_∞. However, note that if we wanted to plot the PDF, we have to differentiate the CDF, and this means that we have to specify what interpolation is used. In a way, the reason that the PDF is difficult to generate is that it is the derivative of a numerically-specified function - and numerical differentiation is poorly conditioned. We will work with low-dimensional representations of CDFs in this paper.

The most complete representation of a CDF is the sorted set $\{x_i\}$, but that is not a concise representation. We will approximate a CDF using a set of orthogonal basis functions. However, we first "turn the graph on its side." That is, we will approximate x by a weighted sum of orthogonal functions of y. The easiest way to visualize this is that we first reflect the CDF in the line $y = x$ so that the y axis is horizontal and the x axis is vertical. We then choose a set of basis functions, $\phi(y)$ on [0,1] with respect to

some suitable weight and use the approximation

$$x(y) \approx \sum_{j=0}^{q} \alpha_j \phi_j(y) \tag{1}$$

However, note that we are only interested in approximating $x(y)$ on the discrete point set $\mathbf{y} = \{(i - 0.5)/N\}$. Therefore, we choose an orthogonal basis on this discrete set.

As with any approximation problem, the best weight function depends on the problem. In limited experiments on the example discussed later, we have found that a uniform weight and polynomial ϕ_i work well. Thus $\phi_i(y)$ is a polynomial of degree i. Computationally, the $\phi_i, i = 0, \cdots, q$ are represented by their values on the point set $\mathbf{y}$. We have seldom found it necessary to use more than the first six basis functions ($q = 5$). They can be pre-computed (they depend only on N) and form a $(q+1)$ by N matrix, Φ. Then the coefficients $\alpha = \{\alpha_i\}$ in eq. (1) are computed from $\alpha = \Phi\mathbf{x}$. The approximation to $\mathbf{x}$ can then be computed from $\mathbf{x} \approx \Phi^T\alpha$ when needed.

Figures 1, 2, and 3 show the results of three approximations to a random sample. In all three cases, there are 250 points in the sample (the small number was chosen to amplify the deviation from the limiting distributions). The samples are plotted as points, the approximant by a line. In Figure 1 all points are drawn from an N(0,1) distribution, and the approximation is via 5-th degree polynomials. Figures 2 and 3 consists of 125 samples each from the two distributions N(2,0) and N(-2,0) (normal with unit variance and means -2 and +2). Figure 2 also uses 5-th degree polynomials, and, as might be expected, the fit is not good. Because of symmetry, all even coefficients except for the constant term are zero except for noise, so the 5-th degree polynomial really only provides four degrees of freedom. A 13-th degree polynomial provides eight degrees of freedom - and we might expect to need twice as many degrees of freedom to fit a combination of two nearly disjoint distributions. That fit is shown in Figure 3.

Two points are worth noting in Figure 1: the approximation "smoothes out" the irregularities in the sample (although this doesn't show that it is necessarily any closer to the underlying CDF), and the tails of the curve do not fit well. The latter problem is to be expected with polynomial approximants to a distribution with long tails. If it is known that there are long tails, then the basis functions should be changed. For example, the error function could be included as a basis function, and either combined with polynomials, or polynomials could be built on it. (This is not a significant cost because the basis functions are pre-computed.) Another suggestion, due to Sima Setayeshgar of Princeton, is to use a non-linear transformation of the independent variable (y in this case). She proposed using the inverse sine transformation: $z = 0.5 - \arcsin(1 - 2y)/\pi$ and then building a polynomial basis on z. It gives results that match the tails well. The user should be aware that any of these approaches will tend to add tails to the approximant even when the sample doesn't have them!

2 A Biased Random Walk Example

The example treated in this paper is the following problem: time is discretized into intervals of length Δt and $t_n = n\Delta t$. A collection of cells, $\{c_i\}$, occupy positions $x(i,t)$ on the real line. At each time step, t_n, cell c_i moves a distance Δx left or right. The probability of a left move is $p(x_i) = (1 + f(x(i,t_n)))/2$ where $-1 \leq f(x) \leq +1$. The

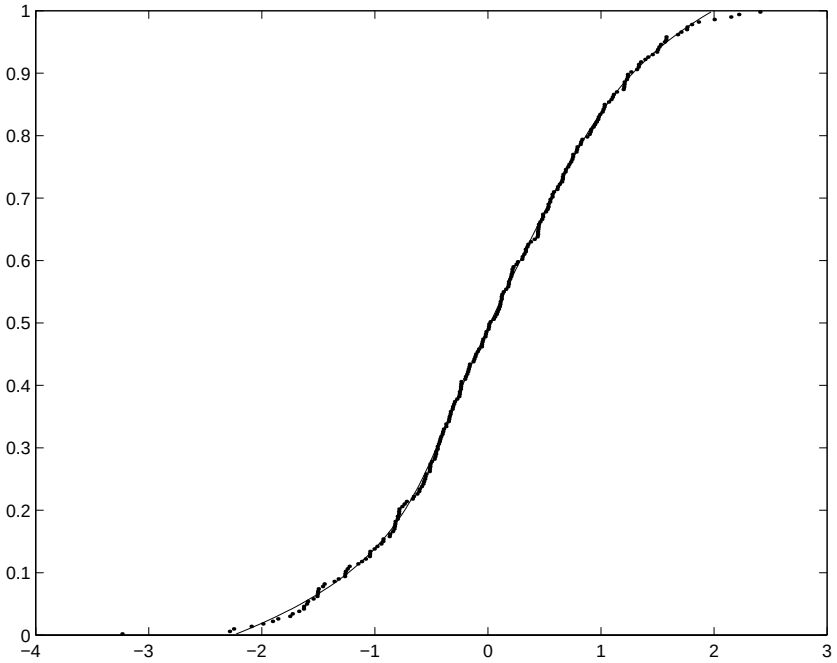

Figure 1: 5-th Degree Polynomial Approximation of Gaussian CDF

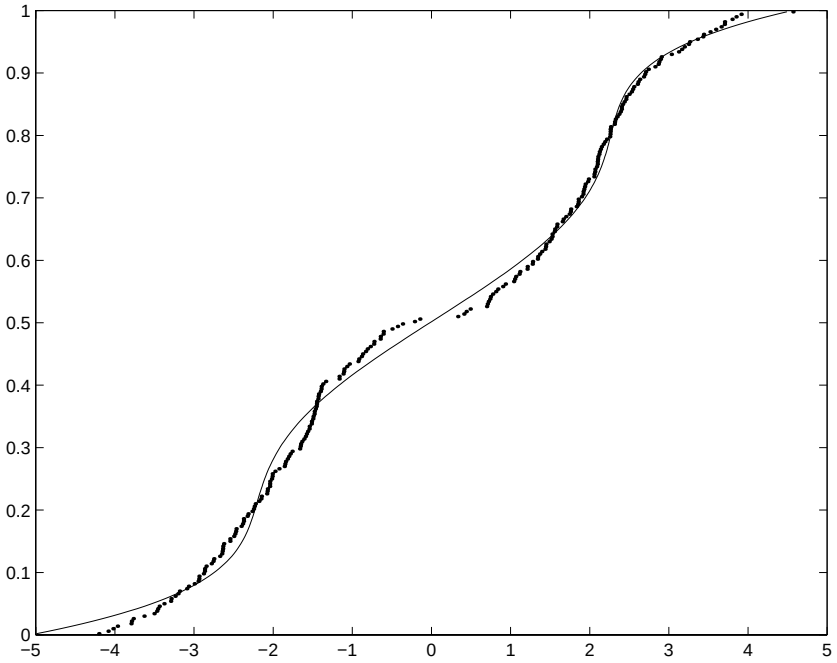

Figure 2: 5-th Degree Polynomial Approximation of Double Gaussian CDF

probability of a right move is, therefore, $(1 - f(x(i, t_n)))/2$. The function $f(x)$ is monotonic non-decreasing. If $f(x) = -1$ for $x \leq x_L$ and $f(x) = +1$ for $x \geq x_R$ it is clear that after a finite time, any initial distribution will be constrained to the open interval $(x_L - \Delta x, x_R + \Delta x)$. If, in addition, $x_R - x_L = M\Delta x$ and all particles are initially assigned to the discrete point set $x_m = x_L + m\Delta x, m = 0, 1, \cdots, M$ then they remain on that set. If, in addition, f is linear (in its range from -1 to 1), M is even, and the number of cells initially assigned to odd values of m is the same as the number initially assigned to even values of m, then the probability distribution of the cells converges to the binomial distribution. (The odd/even condition is required because in the model it is clear that each cell alternates between an even- and an odd-numbered position. This could be avoided by providing for a finite probability, p, of a cell remaining in place at each step and adjusting the left and right probabilities appropriately, but it slows the simulation while adding no value to the model's illustrative capability.)

In the limit as the number of cells becomes infinite, and as Δx goes to zero and $f(x)$ goes to zero as $f(x) = \Delta x q(x)$, the distribution satisfies the partial differential equation:

$$\frac{\partial u}{\partial t} = \frac{(\Delta x)^2}{\Delta t}(1 - p)\left[\frac{\partial (qu)}{\partial x} + \frac{1}{2}\frac{\partial^2 u}{\partial x^2}\right]$$

where p is the probability of remaining in place.

Note that if $q(x)$ is linear in x, say $q(x) = \alpha x$ with $\alpha > 0$, then a steady-state solution is

$$u(x) = Ae^{-\alpha x^2}$$

which is a normal distribution. A general solution is

$$u(x, t) = Ae^{-\gamma(t)(x - z(t))^2 + \beta(t)}$$

where γ, z, and β satisfy the ODEs

$$\frac{d\gamma}{dt} = 2\sigma\gamma(t)(\alpha - \gamma(t))$$

$$\frac{dz}{dt} = -\sigma\alpha z$$

and

$$\frac{d\beta}{dt} = \sigma(\alpha - \gamma(t))$$

with

$$\sigma = \left(\frac{(\Delta x)^2}{\Delta t}\right)(1 - p)$$

3 Projective Integration of CDFs

The basic idea of projective integration, described in [1], is to compute the solution (by the "inner integration" method) at a sequence of $k + q$ time steps and then to perform a q-th degree polynomial extrapolation on the results of the last $q + 1$ steps. The first k

time steps serve to damp the stiff components (or to equilibrate statistically unreasonable initial conditions). The next q provide the information for the extrapolation forward in time (called the *projective step*).

In the context of this stochastic example, we do the following starting from any initial random distribution of cells on the real line:

1. Perform the random simulation for k time steps.
2. Perform the random simulation for a further q time steps.
3. *Restrict* the outputs from the k through $k+q$-th time steps to the orthogonal basis representations, α_n, for $n = k, k+1, \cdots, k+q$. (This requires that the components of $\mathbf{x}_n$ be sorted into ascending order before forming $\alpha_n = \Phi \mathbf{x}_n$.)
4. Perform the projective step (a polynomial extrapolation) through the α_n for $n = k$ to $k+q$ to get α_{k+q+M}.
5. Compute the value of $\mathbf{x}_{k+q+M} = \Phi^T \alpha_{k+q+M}$.

This process is then repeated as often as necessary to cover the total interval of integration. Note that after step 5 the values in the vector $\mathbf{x}$ may no longer be ordered. This does not matter as all we are concerned about is that we have a sample of N values. They will be resorted before further restriction operations.

In the earlier paper the projective step was a q-th order extrapolation through the last $q+1$ points. In the stochastic case, such as this example, the results at each time step differ from the limiting distribution by random "noise." This can be reduced by increasing the sample size. Its effect can also be reduced by increasing q but not increasing the order of the polynomial extrapolation. In our tests, for example, we typically used $q = 10$, but did a first or second order polynomial extrapolation for the projective step based on a least-squares fit. This was done separately for each component of α. As might be expected, the amount of noise in the components α_i of α_n increases with i and we found that it was best to use a lower-order extrapolants for larger i's. In fact, for modest N (2000) we found that second degree for α_0, first degree for α_1 and zero-th degree for the other extrapolants was adequate.

Figures 4, 5, and 6 show the results of a sample run of this problem for the coefficients α_0, α_1, and α_3. N was 2000, k was 0 (it is not necessary to equilibrate after the projective step in this problem), q was 10, and M was 10. The figures show the output of direct simulation (points), the output of the simulation steps used to form the projective step (x marks), and the projections (lines). Five "outer" steps are performed, each consisting of 10 simulation steps followed by an extrapolation over 10 steps for a total distance of 100 steps. Δt was 0.1 - this is an arbitrary choice since it plays no part in anything but the asymptotic differential equation which is not used in the simulation. The random numbers used in the direct simulation and the projective one were the same - that is, when a simulation step is done preparatory to a projection step, the random numbers that are used to decide whether to go right or left are the same ones as those used in the direct simulation step at the same time level.

The initial conditions were drawn from a Gaussian with mean and variance significantly different from the steady state so the solution consists of samples from a moving Gaussian (its mean and variance are changing with time). The value of α_0 is the mean, while the value of α_1 is proportional to the standard deviation.

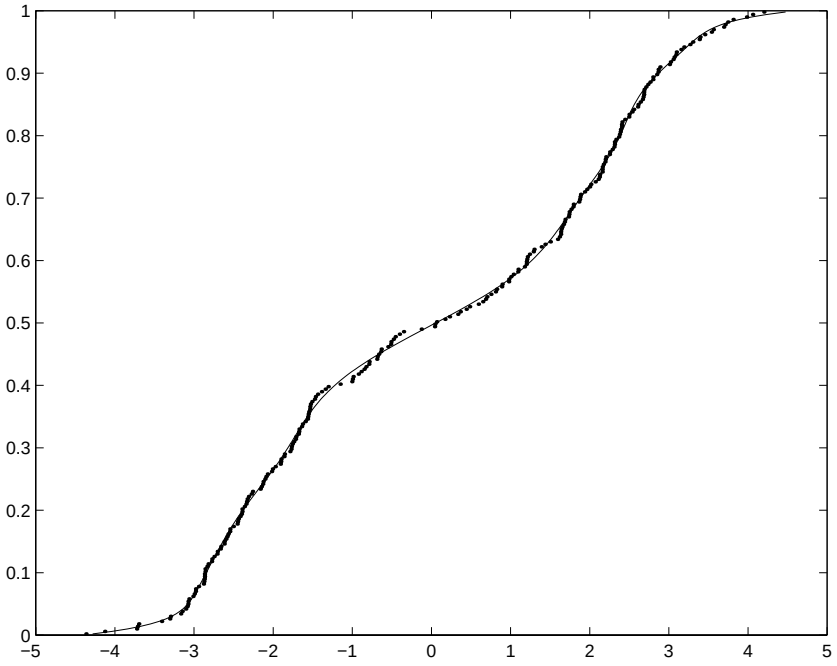

Figure 3: 13-th Degree Polynomial Approximation of Double Gaussian CDF

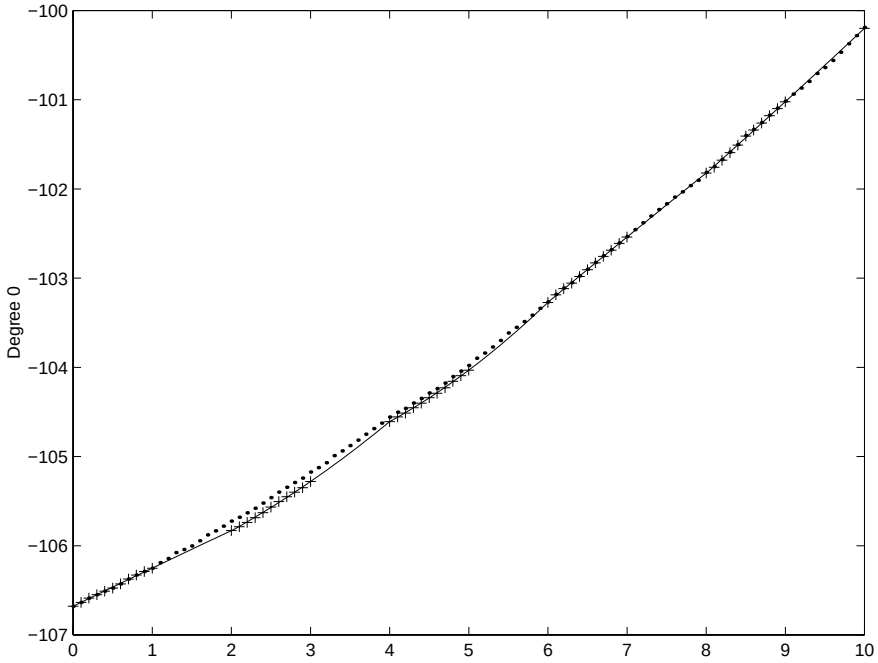

Figure 4: Coefficient α_0

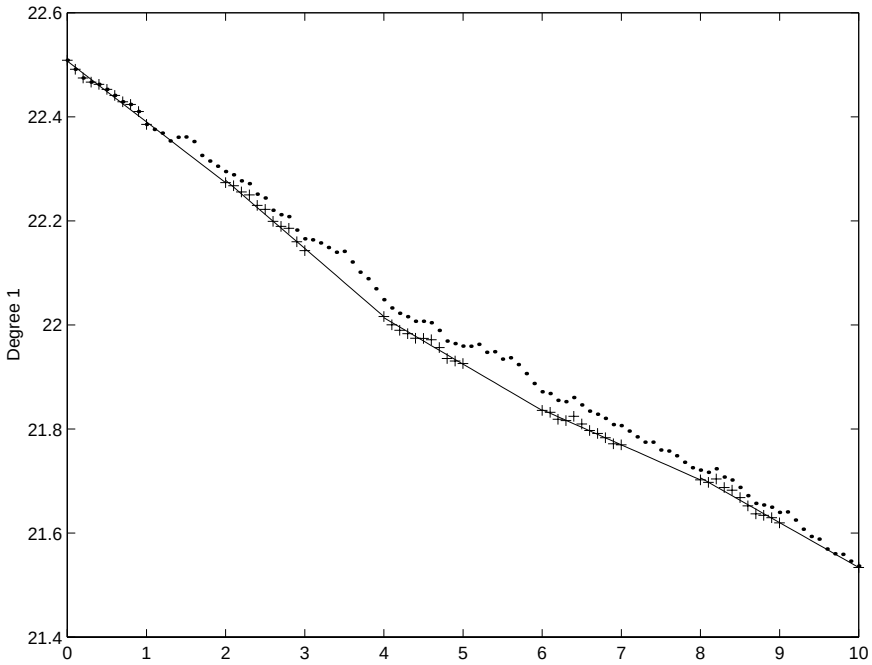

Figure 5: Coefficient α_1

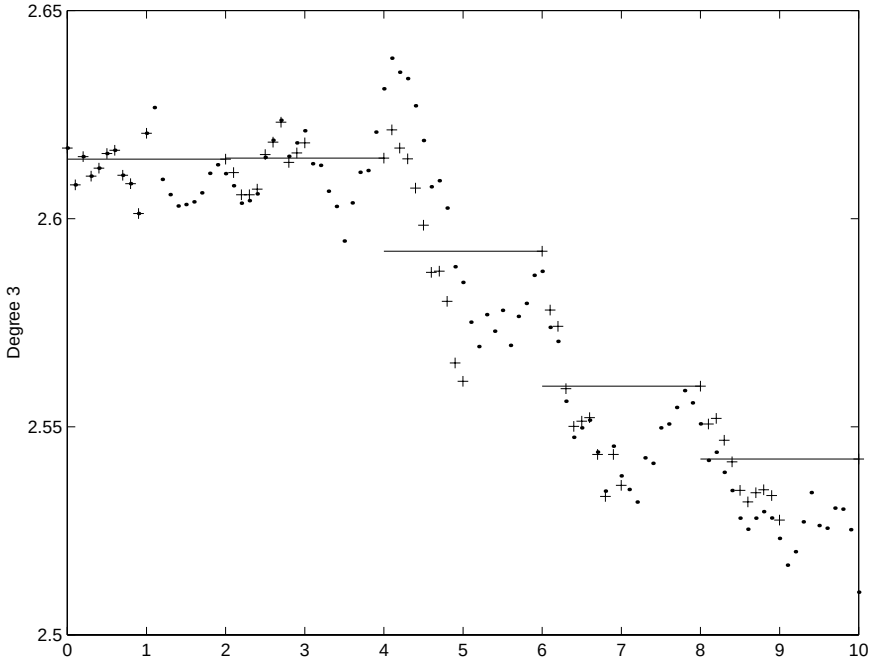

Figure 6: Coefficient α_3

The second-order extrapolation for α_0 is not appreciably better than first order would have been. As can be seen in Figure 6 the computed values of α_3 contain a lot of noise, so using higher than zero-th order extrapolation would be folly unless N and/or q were increased significantly. We do not show α_2 since it is similar to α_3. Also, since the CDF of the Gaussian "on its side" is a constant plus an odd function about the midpoint of the interval, the large N limit solution for α_2 (and all the higher even coefficients) is zero. The noise in the projective solutions is not significantly different in size from the noise in the direct simulation coefficients.

The projective method uses half the number of simulation steps of the direct simulation approach in this example (because $k + q = 10$ and $M = 10$). If the simulation of each individual cell is computationally expensive, the projective method could be roughly twice as fast.

4 Comments

We have shown how the projective integration method can be applied to a stochastic simulation. The two key ideas in this paper, in our view, are the way in which the distribution is represented (the CDF "on its side") and the use of a least squares approximation in the projective step. The former avoids all of the difficulties associated with the representation of the PDF, while the latter reduces the sample size needed to get sufficient accuracy in the projective step.

References

[1] Gear, C. W. and Kevrekidis, I. G, Projective Methods for Stiff Differential Equations: *problems with gaps in their eigenvalue spectrum*, NEC Research Institute Report 2001-029, submitted to SISC.

[2] Gear, C. W. and Kevrekidis, I. G, Telescopic Projective Methods for Stiff Differential Equations, NEC Research Institute Report 2001-122, submitted to JCP.

[3] Gear, C. W., Kevrekidis, I. G., and Theodoropoulos, C., "Coarse" Integration/Bifurcation Analysis via Microscopic Simulators: micro-Galerkin Methods, NEC Research Institute Report 2001-106, to appear, Computers and Chemical Engineering.

[4] Othmer, H. G. and Schaap, P., Oscillatory cAMP Signaling in the Development of Dictyostelium Discoideum, *Comments on Theoretical Biology*, 5, pp175-282 (1998).

[5] Spiro, P. A., Parkinson, J. S., and Othmer, H. G., A model of excitation and adaptation in bacterial chemotaxis, PNAS, 94 (1997), pp. 7263-7268.