Coarse KMC methods for ODEs: *Stationary states, limit cycles, and reverse integration*

R. Rico-Martinez[1,2], C. W. Gear[1,3], Ioannis G. Kevrekidis[1]
[1]Princeton University, Dept of Chemical Engineering
[2]Instituto Tecnológico de Celaya, Dpto. de Ingeniería Química
[3]NEC Laboratories (retired)

March 29, 2003

Abstract

In "equation-free" modeling, the model is given at a microscopic level yet we believe the macroscopic properties can be described by equations involving only coarse variables (typically various low moments of the microscopic model). If we do not know those equations we are forced to model at the microscopic level, even if we only need to know the macroscopic behavior.

This paper considers the problems of computing the macroscopic limit cycles and stationary points (especially ones that are saddle points or unstable) for systems whose microscopic models are stochastic. In particular, we use projective multi-step integration to perform *reverse integration* (backwards in time to an initial stationary point if there is one) even though the stochastic microscopic simulation can move only forwards in time.

Keywords

1 Introduction

In previous work we have used *coarse time-stepping* as a tool for computing the properties of the macroscopic description of a process even when we only know the model of the microscopic process. A coarse time-stepper is based on a pair of transformations between the microscopic and macroscopic descriptions: *lifting*, μ, which takes the macroscopic description into the (usually higher-dimensional) microscopic description and *restriction*, $\mathcal{M}$, which goes in the other direction. A coarse time step in a method of computing the change in the macroscopic description over a time step $\Delta T = n\delta t$ where δt is the time step of the microscopic model. One coarse time step starting from the macro value $Y(T)$ consists of

1. Lift to an initial value at the microscopic level: $y(T) = \mu Y(T_0)$

2. Integrate forward at the microscopic level n time steps to $y(T + n\delta t)$.

3. Restrict the final answer to the coarse variables $Y(T + \Delta T) = \mathcal{M}y(T + n\delta t)$.

Applied directly to long time integration, the coarse time-stepper would do nothing to reduce the cost of computing at the microscopic level. Neither would there be any point to performing the lifting operations after the initial one because the lifted quantities are available from the previous coarse time step. It is when coarse time-stepping is used in conjunction with other techniques that it provides computational and analytical benefits. These techniques are based on the observation that the derivative of the coarse solution can be estimated from the results of the coarse time-stepper - or more precisely, the chord of the solution over a time step $N\Delta T$ can be computed for any integral N. Thus, the stationary state problem can be solved by finding an initial state, Y^* such that the chord slope is zero. The chords can be used as inputs to an *outer integrator* of the macroscopic variables in a process we called *projective integration*[1]. A limit cycle can be found by solving a two-point boundary value problem at the macroscopic level.

In computing the chord slope, we do not, usually, take just one coarse step, but perform several preliminary ones after the initial lifting operation, and then compute the chord slope of the final one. This is done to allow the initial lifted values to "heal" - that is, to settle to values such that their higher moments have become functionals of the lower ones to be used as coarse variables. (If this does not happen, the system does not close into one describable in terms of the coarse variables.) Since the values of the macroscopic variables are only needed for the chord computation, the restriction operation only need be performed at the two points needed to compute the chord. Thus the chord calculation process to find the slope, S, of a chord of the solution through the points $Y(T + n_1\delta t)$ and $Y(T + n_2\delta t)$ starting from $Y(T)$ is:

1. Lift from $Y(T)$ to $y(T)$

2. Perform n_1 microscopic simulation steps to get $y(T + n_1\delta t)$.

3. Restrict to get $Y(T + n_1\delta t)$.

4. Perform $n_2 - n_1$ more microscopic simulation steps to get $y(T + n_2\delta t)$.

5. Restrict to get $Y(T + n_2\delta t)$.

6. Compute $S = (Y(T + n_2\delta t) - Y(T + n_1\delta t))/((n_2 - n_1) * \delta t)$.

If the microscopic model is stochastic, the chord slope of the coarse time-stepper has stochastic noise. The variance of this noise can be reduced in several ways: the size (number of particles or other components) of the microscopic model can be increased, multiple copies of the microscopic model can be run (in parallel!) and the results averaged, or - the method we will use here - the chord can be computed by fitting a straight line to the output of a number of (coarse) time steps. For this the chord calculation process is

1. Lift from $Y(T)$ to $y(T)$

2. Perform n_1 microscopic simulation steps to get $y(T + n_1\delta t)$.

3. Restrict to get $Y(T + n_1\delta t)$.

4. Repeat the next two steps for $q = 1, 2, \cdots, m$
5. Perform n_3 more microscopic simulation steps to get $y(T + (n_1 + qn_3)\delta t)$.
6. Restrict to get $Y(T + (n_1 + qn_3)\delta t)$.
7.
8. Compute S as the slope of the least-squared linear fit to $Y(T + n_1 + qn_3\delta t)$ for $q = 0, 2, \cdots, m$.

The proc his paper we will use this form of cl step integration and limit cycles, re

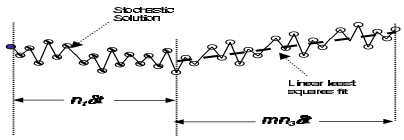

Figure 1: Computing the chord slope.

One can find stationary points in which all of the eigenvalues are in the positive half plane (in other words, completely unstable stationary points rather than saddle points) by integrating backwards in time. In [2] we showed that the properties of projective integration that damp highly stable components combined with taking the projective step backwards in time allows us to find stationary saddle points if the local eigenvalues in the negative half plane have large negative real parts while those in the positive half plane have small real parts. In this process the *inner integrator* proceeds in the forward (in time) direction while the outer (projective) step id backwards. When the microscopic model is stochastic, it is usually inherently uni-directional in time. For example, there is no difference in a random walk with "negative" directions than in one with positive ones. In either direction it describes the evolution in forward time of the microscopic process. In Section 4 of this paper we will combine a stochastic microscopic model with a reverse projective integrator to find a stationary saddle point of a system.

The model problem used here is naturally one for which we know the macroscopic equations so that comparisons can be made between a stochastic integration based on a microscopic model and the deterministic macroscopic equations. It one of the examples used by [3], and is a kinetic Monte Carlo description of a simple surface reaction model for which the mean field evolution equation for the coverages (θ_i) of the species are known [3].

$$\begin{aligned}
\frac{d\theta_A}{dt} &= \alpha\theta_* - \gamma\theta_A - 4k_r\theta_A\theta_B \\
\frac{d\theta_B}{dt} &= 2\beta\theta_*^2 - 4k_r\theta_A\theta_B \\
\frac{d\theta_A}{dt} &= \mu\theta_* - \eta\theta_C
\end{aligned}$$

where $\theta_* = 1 - \theta_A - \theta_B - \theta_C$. For our examples, the values of the parameters are set as follows: $\mu = 0.36$, $\eta = 0.016$, $\alpha = 1.6$, $\gamma = 0.04$, $k_r = 1.0$, and β was varied as cited in the text.

The KMC simulations were performed as described in [3] using $N = (1000)^2$ adsorption sites and computing the average over several realizations, typically 100. Comparisons were made with numerical integration of eq. (1) using ODESSA ?????citation??.

2 Forward Projective Integration

Projectiv orm when used with the estimatio own in Figure 2.

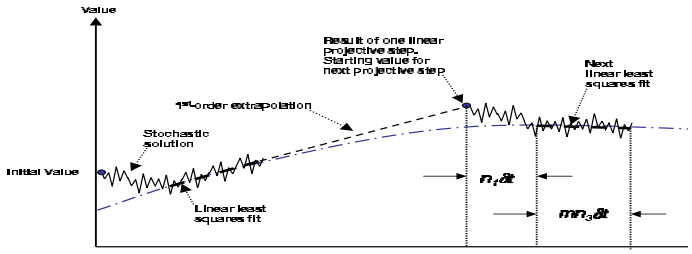

Figure 2: First-order projective integration.

In practice we need to use a higher order method. Here we will use an explicit multistep method similar to the Adams-Bashforth method. The Adams-Bashforth method can be found in any standard text book[4]. The third-order method is

$$y(T + H) = y(T) + \frac{H}{12}(23\dot{y}(T) - 16\dot{y}(T - H) + 5\dot{y}(T - 2H))$$

where H is the step size. This formula assumes that we have estimates of the derivatives $\dot{y}$ that are at least second-order accurate at three places: T, $T-H$, and $T-2H$. Higher order methods require correspondingly more derivative estimates of correspondingly higher accuracy (p-th order accurate estimates for an order $p+1$ integration method). The formula also assumes that the spacing, H, between the integration points is fixed. Neither of these assumptions is strictly correct in the stochastic chord slope estimator we are using. Referring to Figure 1 we see that if the microscopic simulation starts at the point T, we will actually estimate the slope of the chord between $T+n_1\delta t$ and $T+(n_1+mn_3)\delta t$. This is a first-order accurate estimate of the derivative at $T + (n_1 + mn_3/2)\delta t$, not at T. Hence, we need to modify the Adams-Bashforth method in two ways.

To discuss this, define $T_{n+1} = T_n + H$ where H is the "projective step size," that is, the distance between successive groups of inner steps. Let T_n be the center of the chord computed via a least-squares fit. Let s_n be the computed slope of that chord. Then we need to integrate from T_n to $T_n + H - (n_1 + mn_3/2)\delta t$ to get an approximation to the y value at the start of the next stochastic computation as shown in Figure 1. This is the first modification to the Adams-Bashforth method. The first order method, which

is Euler's method, is straightforward:

$$y(T_n + H_0) \approx y(T_n) + H_0 s_n$$

where $H_0 = H - (n_1 + mn_3/2) * \delta t$. As long as s_n is a zeroth-order accurate approximation to $\dot{y}(T_n)$, this is a first-order method. Second and higher-order methods must reflect the fact that the final step is less than the spacing, H, between the past points. For example, the second-order method is

$$y(T_n + H_0) \approx y(T_n) + H_0((s_n + \frac{H_0}{2H_1})(s_n - s_{n-1})).$$

If s_n and s_{n-1} are first-order accurate approximations of the derivative, this is a second-order method. Similar modifications apply to higher-order methods, and can be found via routine algebra.

We do not compute approximations of the derivatives, but of the slopes of a least-squares linear fit. It is clear that these are first-order approximations to the derivative at the center, so no further adjustment is needed to these formulae through second order. However, the order of the difference between s_n and $\dot{y}(T_n)$ is $O(mn_3\delta t)$. The error introduced is of order $O(H^2(mn_3\delta t/H))$. In practice, whether or not an additional correction is needed depends on the size of $mn_3\delta t/H$. If this is small, the additional error is unimportant, but if the projective step is not moderately large compared to the length of microscopic integration used to estimate the chord slope, additional corrections to the integration formula are needed. Because the size of $(H^2(mn_3\delta t/H))$ in the following is ?????????, we will ignore this correction.

Figures 3 and 4 compare the results of projective integration using ?????????(details) with an accurate integration using ODESSA. The projective integrator parameters were chosen via the comparisons between different order methods and different projection steps, as shown in Figures eq. (5) and 6. The length of the "settle" time was based on the size of fastest time constant whose eigenvalue is around -6.

3 Limit Cycles

When we wish to compute the limit cycle from the output of a coarse time-stepper, the main challenges are the noisy nature of the trajectories given by the coarse time-stepper and the fact that the time stepper can only be iterated forward in time.

In order to compute the cycle, we will look at the Poincaré map of the trajectory. The Poincaré map is defined by the successive intersections of the time-stepper trajectory with a plane $H(\vec{Y})$ transversal to the flow; Under the Poincaré map, a simple stable periodic trajectory of the original flow becomes an attracting fixed point.

Assuming that H is defined by $\{\vec{Y} \in R^n : Y_i = C\}$, a crossing on the Poincaré section can be detected when the quantity $(C - Y_i)$ changes sign [5]. Since one is only interested in returns to the surface in the same direction, one wants the slope of the transverse intersection to the surface to have the same sign all the time. As it will be discussed below, for noisy systems, these simple criteria require additional steps to prevent the detection of false crossings.

Uncertainty in any state variable, makes the computation of sign changes on such variables unreliable. However, since we assume closure of the time-stepper trajectory,

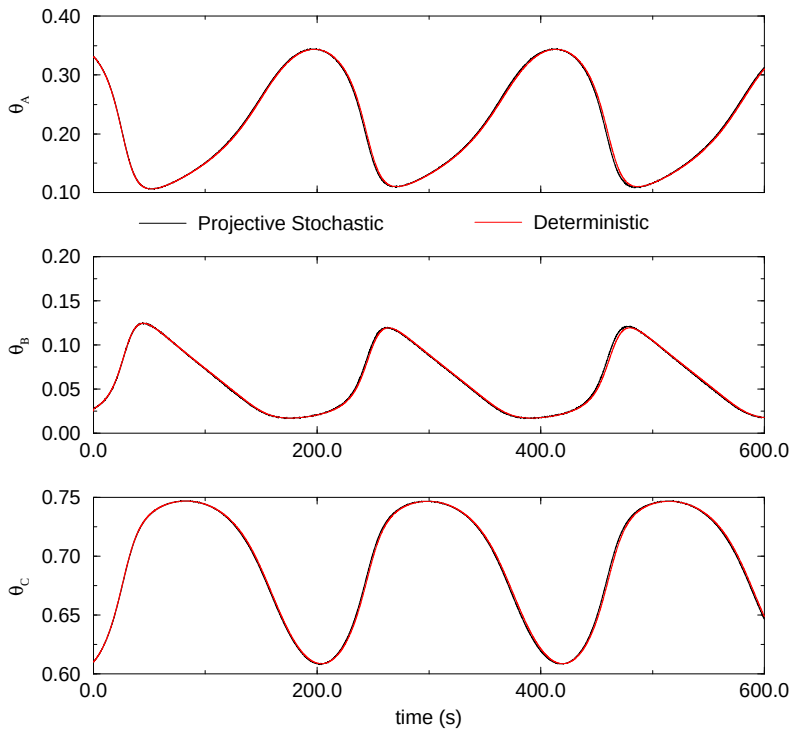

Figure 3: Comparison of time series. Model 3. Time step is 0.04. Stochastic code was used for half the time step and the remaining was "projected" using Adams-Bashforth.

we could expect at least local smoothness of the trajectory near the Poincaré plane. Accordingly we devise the following algorithm to detect crossing at the Poincaré surface

1. Monitor the difference $|Y_i - C|$ along the trajectory.
2. If $|Y_i - C| < \delta$ at some time ($t = t_0$), gather a fixed number of points (M, until $t = t_M$) along the trajectory. With these points, approximate the trajectory of the state variables ($j = 1, 2, ..., n$) via a linear mapping: $Y_j = a_j t + b_j$.

   RAMIRO: why do we look at a linear approx to all state variables, and not just Y_i here?
3. Assisted by this local model, confirm that a crossing as occurred. That is, $Y_i = C$ for some $t \in [t_0, t_M]$. If this condition is not met, return to step 1.

Note that this algorithm also gives information about the direction of the crossing (the sign of the slope of the local model for x_i); thus, the crossings at the Poincaré surface

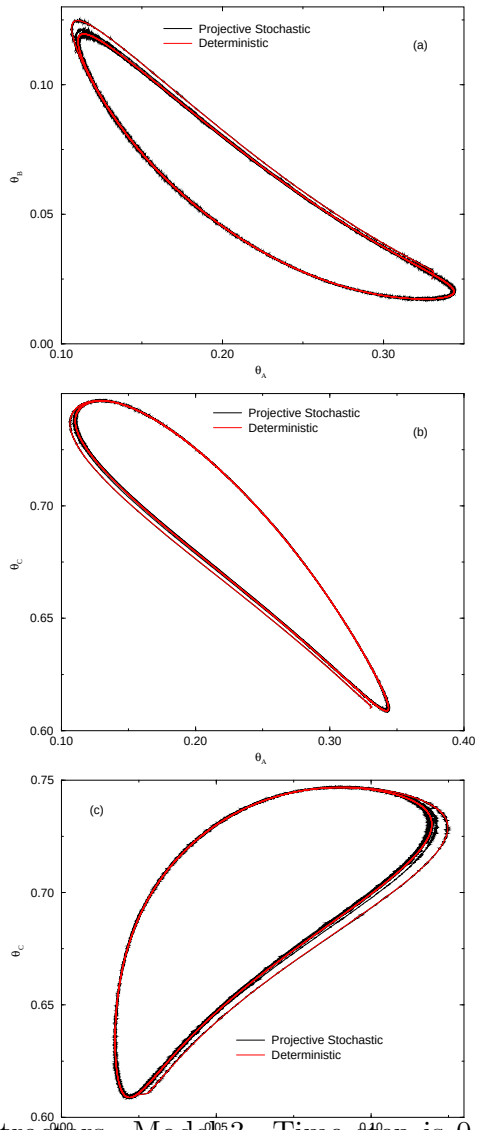

Figure 4: Comparison of attractors. Model 3. Time step is 0.04. Stochastic code was used for half the time step and the remaining was "projected" using Adams-Bashforth.

are fully determined by requiring, in addition to condition 3 above, that the sign of a_i is the same for all crossings.

In the neighborhood of a simple stable periodic trajectory, a properly defined Poincaré section will translate as a slowly convergent sequence. If one imposes a Newton-Raphson contraction mapping on the Poincaré section, the convergence to the fixed point of the sequence (i.e., to the periodic trajectory of the flow) can be accelerated. This contraction map is defined as follows. Let $\mathbf{x}$ be the vector of components of Y excluding Y_i used to define the Poincaré section.

$$\mathbf{x}^{k+1} = \mathbf{x}^k = -J(\mathbf{x}^k)^{-1}(\mathbf{x}^k - \mathbf{x}^{k-1})$$

where J is the Jacobian of the linearization of the Poincaré map.

The Jacobian can be estimated by applying perturbations around the current Poincaré crossing. The usual differences schemes may be very sensitive to the noisy output given by the time-stepper; In order to reduce such sensitivity, we used a "centered" ensemble

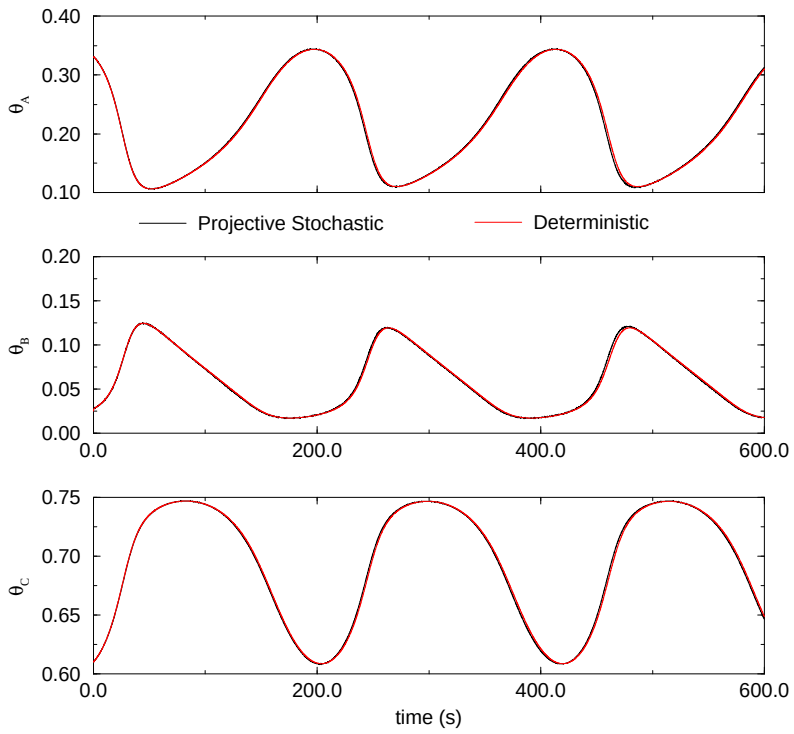

Figure 5: Time evolution of θ_B for several projective integrator parameters. WE NEED MORE DETAILS WHEN WE GET THE ACTUAL PLOTS

of perturbations (we apply several symmetrical perturbations around the current point).

3.1 Results

The KMC simulations were performed as described in Section2 using $N = (1000)^2$ adsorption sites and computing the average over 100 realizations. The deterministic (mean-field approximation) system exhibits a periodic trajectory. If looked upon in a Poincarè framework, using $\theta_A = 0.33$ as the pinned variable and considering only crossings with negative slope, the fixed point of the Poincarè section is found at $(\theta_B, \theta_c) = (0.027947, 0.61173)$. It is indicated with a filled square in Figure 7). The leading multiplier is 0.72147 while the second multiplier 2.92×10^{-8}. The the return time 183.89 time units.

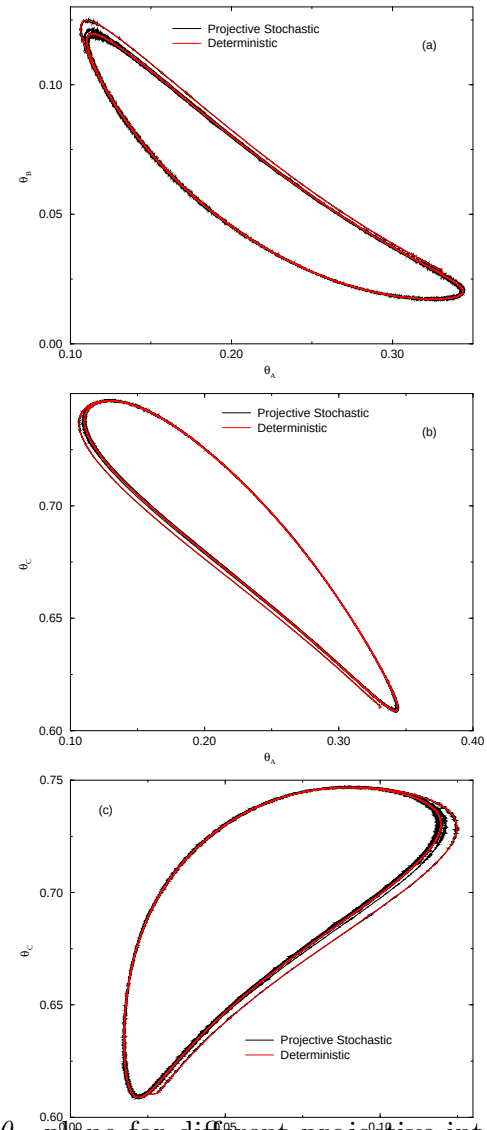

Figure 6: Attractors in θ_A-θ_C plane for different projective integrator parameters. WE NEED MORE DETAILS WHEN WE GET THE ACTUAL PLOTS

3.2 Converging on a Limit cycle

Note the difference on several orders of magnitude of the multipliers on the limit cycle indicating that perturbations along one of the directions (on the two-dimensional Poincaré section) decay very quickly. In order to avoid large numerical sensitivity during the Jacobian estimation, we resort to a one-dimensional approach leading to the slaving of θ_C, at the Poincarè crossing, to θ_B. We use the eigenvectors of the linearization around the fixed point for the deterministic (mean-field) system. The resulting line equation (see Fig. 7 all points of the NR iteration lie on it) is:

$$\theta_C = (1.0 - 2.63149316\theta_B)/1.51747895$$

This approach gives an error $O(10^{-4})$, commensurable with the expected uncertainty level from the KMC simulation. The convergence criteria, maximum deviation to declare convergence, was set to 0.0001 on θ_B. The NR converges after 5 iterations (point 0 on Fig. 7 is the initial point). The derivatives of the map were estimated using three

perturbations around the current point and fitting a least-squares line (3 points), the perturbations were 0.0005, 0.0, and -0.0005 on θ_B. The estimated leading multiplier was 0.6832 and the return time 177.68 time units. ‘ The location of the fixed point is estimated to be at $(0.028050, 0.610346)$.

4 Reverse Projective Integration

Reverse projective integration, described in [2], is a version of projective integration in which the inner integrator proceeds in a forward direction while the projective step is in the reverse direction. It was used in [2] because of its unusual stability properties that enabled us to find stationary saddle points. Here we use it to integrate in the reverse direction even though the coarse time-stepper can only move forward in time because it is based on a microscopic simulator that only is defined for forward time. It is illustrated in Figure 8 with a linear projective step.

There is a further simplification of eq. (1) to a single ODE that has an unstable stationary point (its sole eigenvalue is positive), given in [3] as

*****Put Alexi model one equation in here

(1)

In this case, the eq. (1) can be integrated in reverse time to find the stationary point. Reverse projective integration can also be used to find that stationary point. Fig. 9 shows the comparison between the reverse deterministic and projective-stochastic integrations for one initial condition. The parameter k_r was set to 5. The trajectories (starting at time=0.0) approach the unstable fixed point. The deterministic trajectory was obtained from ODESSA using a time step of -0.02 units. The projective stochastic was used using the stochastic code forward for 0.02 units of time and then the Adams-Bashforth formulae, as described above, with a time step of -0.12 time units.

*****We need to say somewhere what the microscopic step size is, or equivalent information.

Figures 10 and 13 show similar integrations for eq. (1). The trajectories approach a saddle stationary point. The eigenvalues of the linearization of the flow along the deterministic trajectory are presented in Fig. 11. In this case, the deterministic trajectory also was calculated using reverse projective integration with ODESSA as the inner integrator in the forward direction. (This was necessary because of the saddle nature of the stationary point.) ODESSA integrated for on unit of time in the forward direction and projective Adams-Bashforth was used in the reverse direction for -2 units of time. The projective stochastic solution was calculated with the same steps, but the forward integration was carried out with the stochastic code. Seemingly there is significant numerical instability along the stochastic trajectory. This “noise” can be partially explained by looking at the integration of the linearization associated with the eigenvalues closer to zero. A small perturbation along the trajectory (say δ) is amplified in a significant part of the trajectory according to the factors plotted in Fig. 12. These factor were calculated as the exponential of the real part of the pair of eigenvalues closer to zero multiplied by the time steps (forward and backward). Even when such perturbations (that are a natural part of the stochastic simulation results) are not amplified, they diminish very slowly. The choice of time-steps is critical as we should use a large forward step to prevent the forward stable direction from producing a blow up of the backward trajectory. However, Fig. 13 shows that the trajectories are similar when plotted in phase-space.

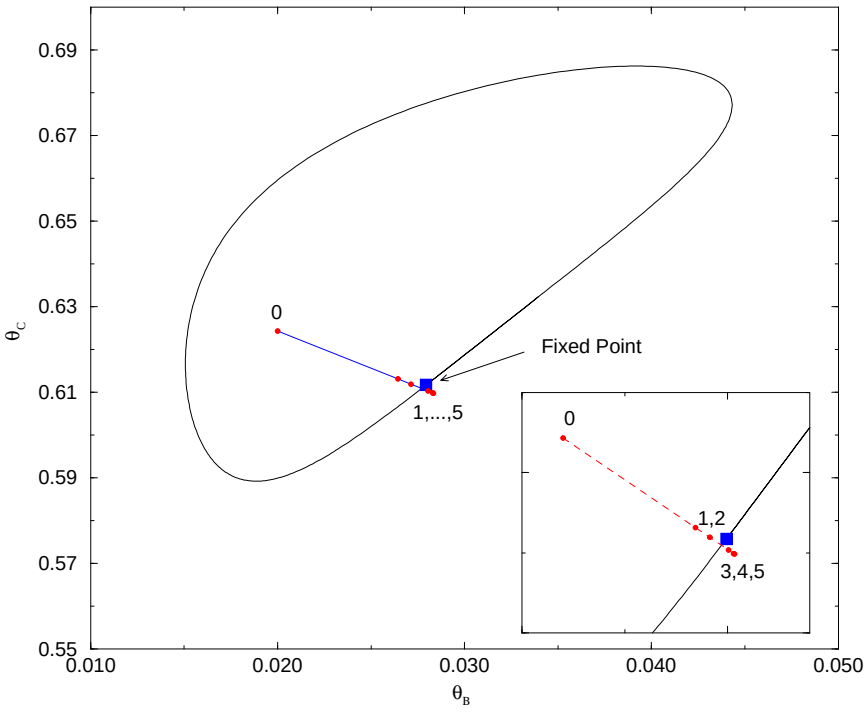

Figure 7: Newton Raphson trajectory of the convergence to the limit cycle. The initial point is marked 0, remaining points are given by the Newton Iteration formula.

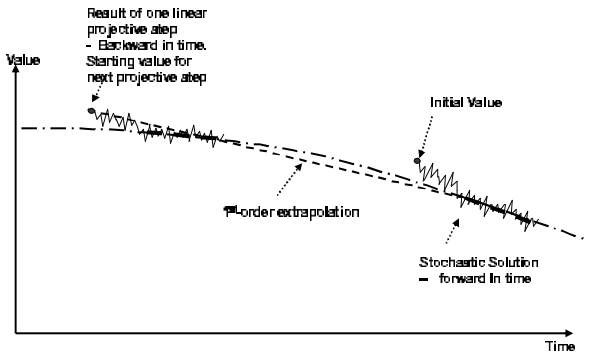

Figure 8: Reverse Projective Integration.

Figures 14, 15, 16 and 17 show the same results when the trajectory is restarted closer to the saddle point, after the region where the perturbations are amplified (at the point of the deterministic trajectory of Fig. 10 where time=-180 s). The agreement is better, but still significant "noise" can be observed.

5 Conclusion

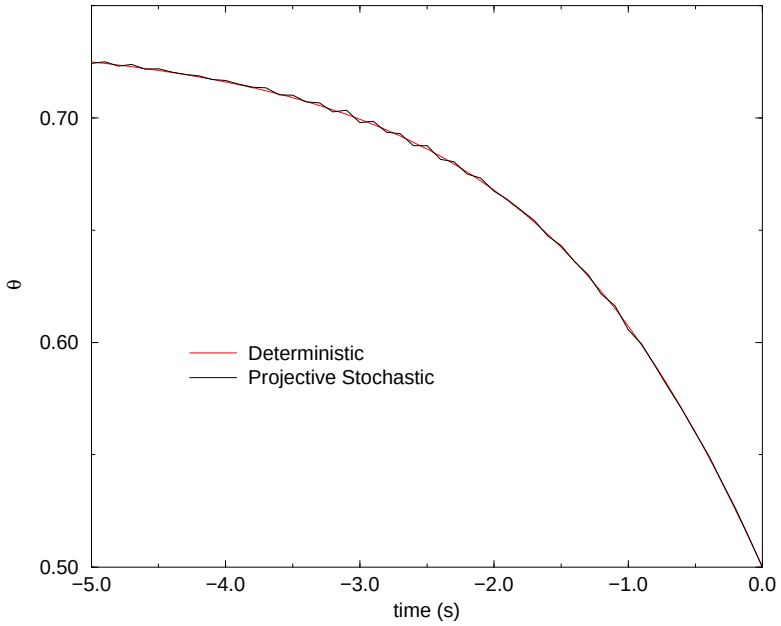

Figure 9: Backward integration for Model 1. The trajectory approaches a unstable fixed point.

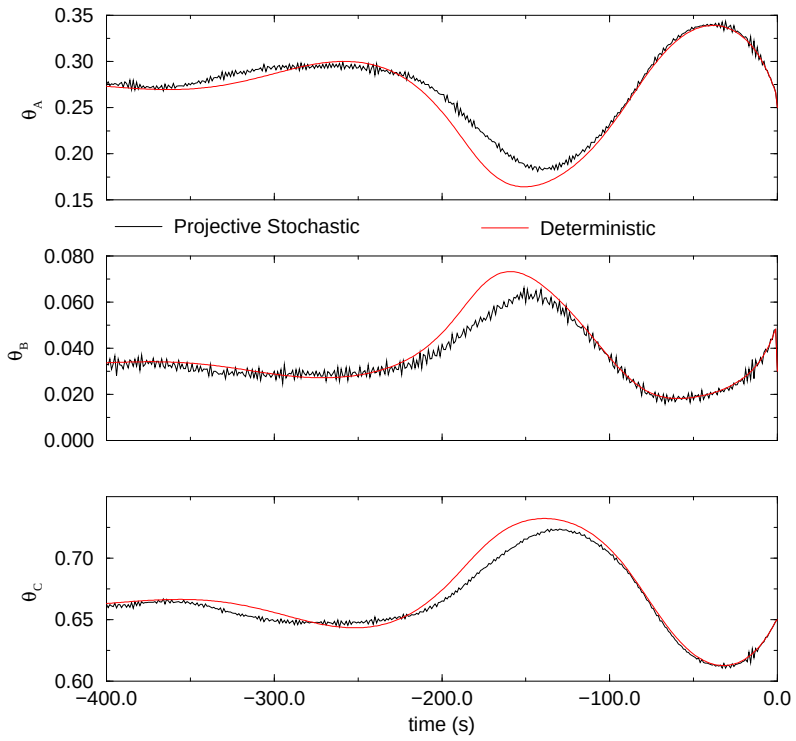

Figure 10: Backward integration for model 3, time series.

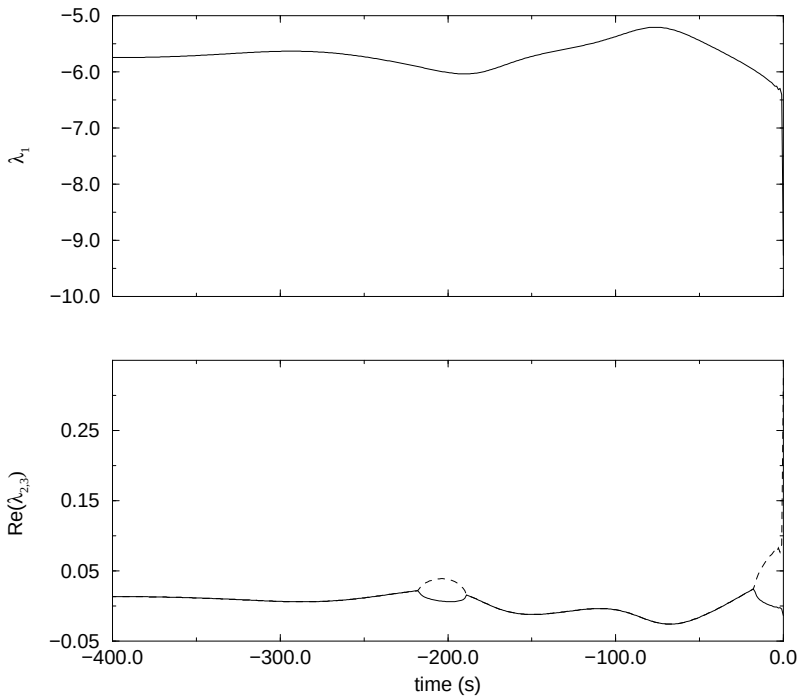

Figure 11: Backward integration for model 3, eigenvalues along the trajectory.

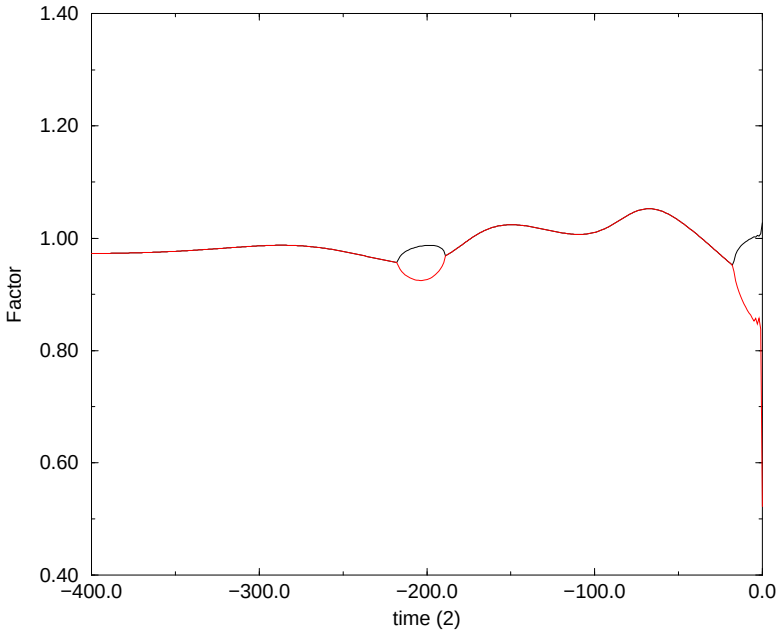

Figure 12: Backward integration for model 3, factor that amplifies (> 1) or diminishes (< 1) the perturbations along the trajectory.

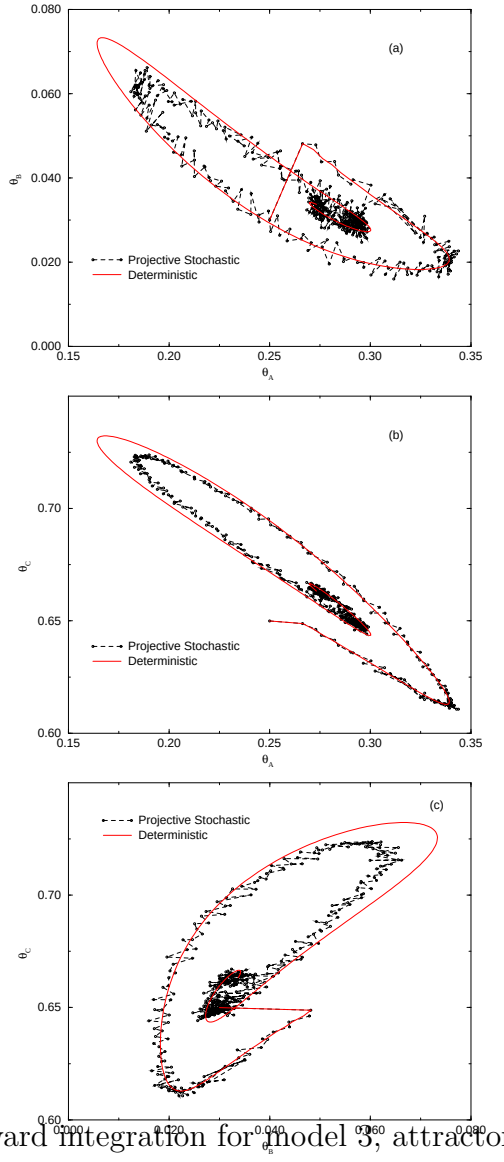

Figure 13: Backward integration for model 3, attractor projections.

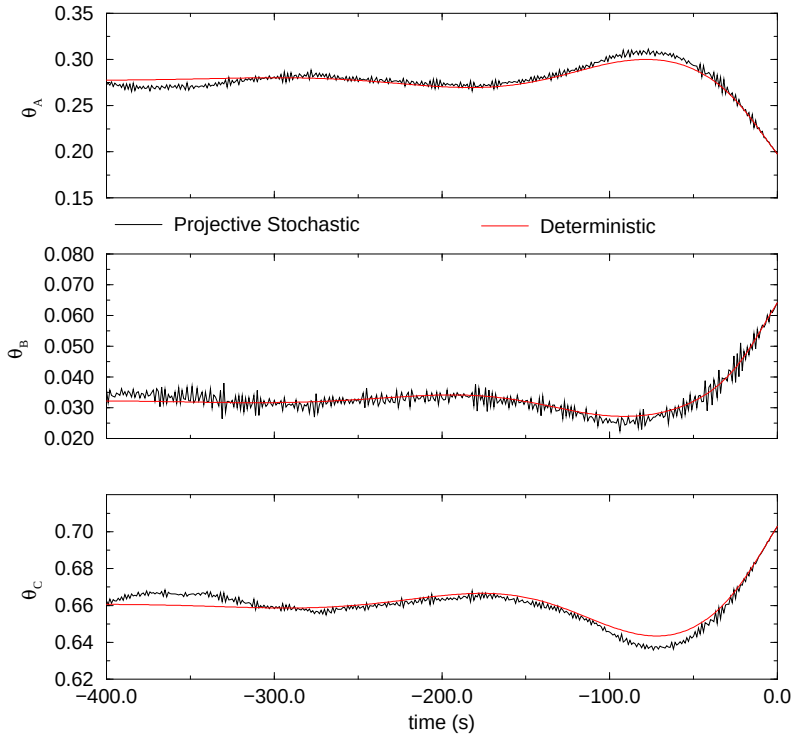

Figure 14: Backward integration for model 3, time series.

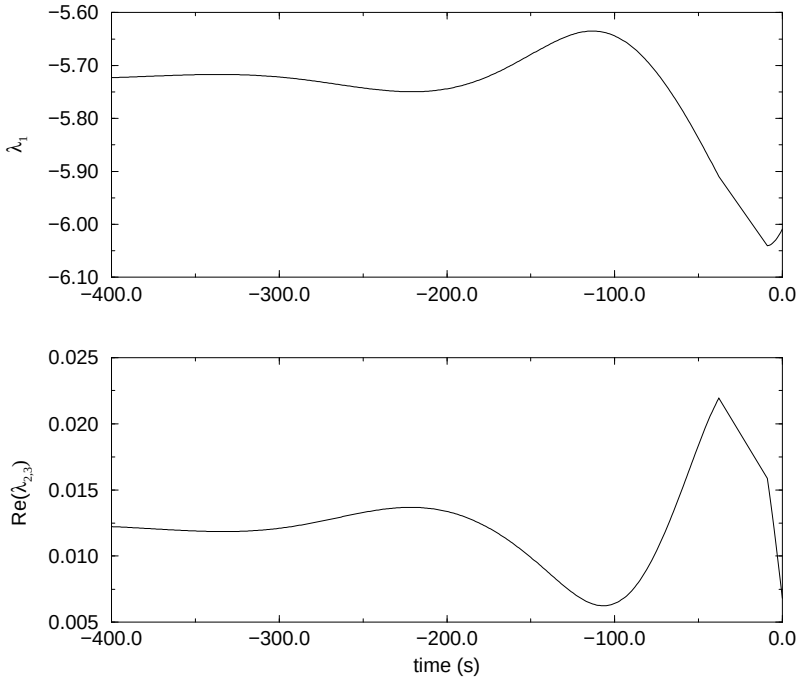

Figure 15: Backward integration for model 3, eigenvalues along the trajectory.

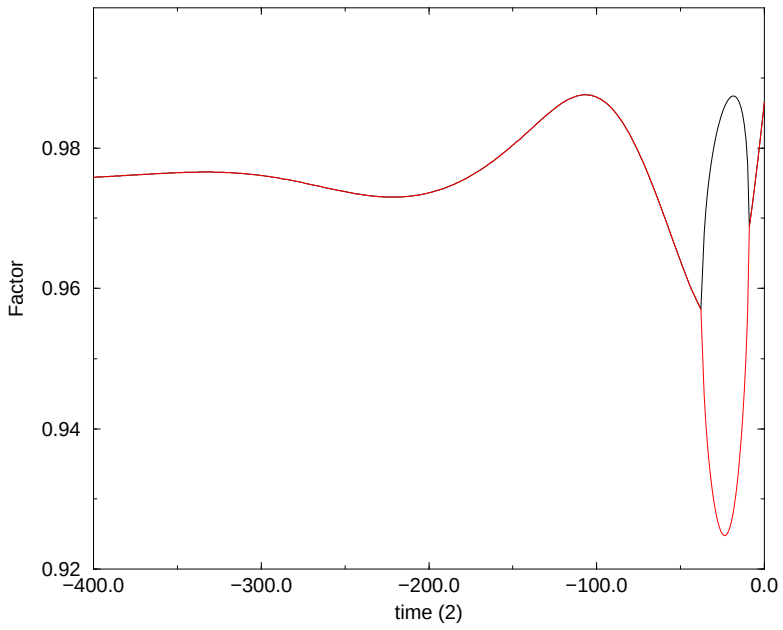

Figure 16: Backward integration for model 3, factor that amplifies (> 1) or diminishes (< 1) the perturbations along the trajectory.

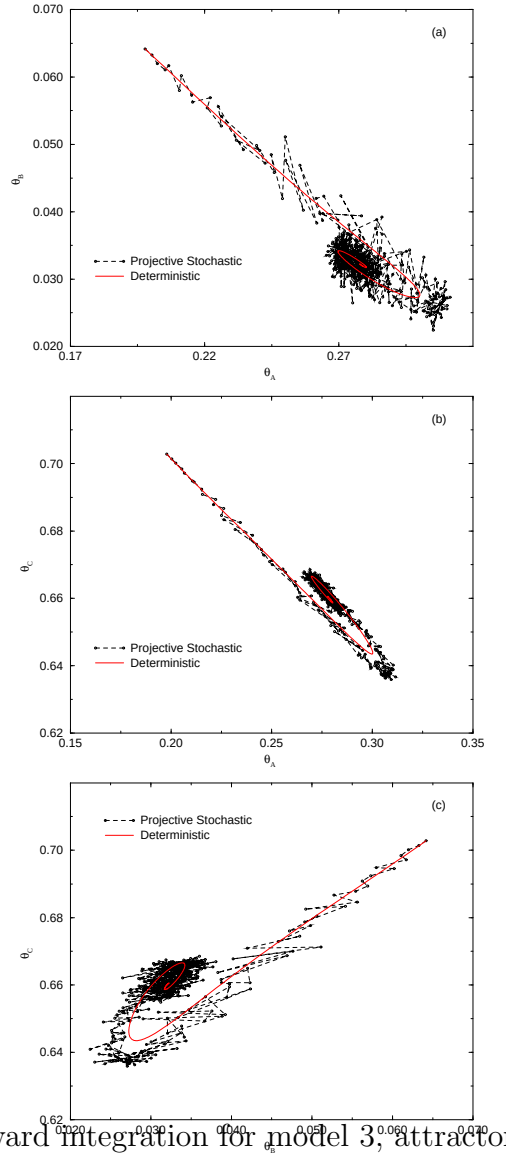

Figure 17: Backward integration for model 3, attractor projections.

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, to appear SISC.

[2] Gear, C. W. and Kevrekidis, I. G, Computing Stationary Points of Unstable Stiff Systems. (Paper on reverse intergation in archives)

[3] A.G. Makeev, D. Maroudas and I.G. Kevrekidis, "Coarse" stability and bifurcation analysis using stochastic simulators: Kinetic Monte Carlo Examples, *J. Comp. Physics*, **116**:10083–10091 (2002).

[4] Gear, C. W., Numerical Initial Value Problems in Ordinary Differential Equations, Prentice Hall, Englewood Cliffs, NJ, 1971.

[5] W. Tucker, Computing accurate Poincaré maps, *Physica D*, **171**:127–137 (2002).

[6] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg and K. Theodoropoulos, Equation-free multiscale computation: enabling microscopic simulators to perform system-level tasks", Submitted to *Comm. Math. Sciences*. (also NEC Technical Report, NEC-TR 2002-010N, http://www.neci.nj.nec.com/homepages/cwg/), August 2002..

[7] A.G. Makeev, D. Maroudas, A. Z. Panagiotopoulos and I.G. Kevrekidis, Coarse bifurcation analysis of Kinetic Monte Carlo Simulations: A lattice-gas model with lateral interactions, *J. Comp. Physics*, in press.

[8] C. I. Siettos, A. Armaou, A. G. Makeev and I. G. Kevrekidis, Microscopic/Stochastic timesteppers and Coarse Control: A kinetic Monte Carlo example, submitted to *AIChE J.*, July 2002.

[9] C. Theodoropoulos, Y. H. Qian and I. G. Kevrekidis, Coarse stability and bifurcation analysis using time-steppers: A reaction-diffusion example, *PNAS*, **97**:9840–9843 (2000).