An equation-free computational approach for extracting population-level behavior from individual-based models of biological dispersal

Radek Erban [a,1], Ioannis G. Kevrekidis [b], Hans G. Othmer [c,*]

[a] *School of Mathematics, University of Minnesota, Minneapolis, MN 55455, USA*
[b] *Princeton University, Department of Chemical Engineering, PACM & Mathematics, Engineering Quadrangle, Olden Street, Princeton, NJ 08544, USA*
[c] *School of Mathematics and Digital Technology Center, University of Minnesota, Minneapolis, MN 55455, USA*

Abstract

The movement of many organisms can be described as a random walk at either or both the individual and population level. The rules for this random walk are based on complex biological processes and it may be difficult to develop a tractable, quantitatively-accurate, individual-level model. However, important problems in areas ranging from ecology to medicine involve large collections of individuals, and a further intellectual challenge is to model population-level behavior based on a detailed individual-level model. Because of the large number of interacting individuals and because the individual-level model is complex, classical direct Monte Carlo simulations can be very slow, and often of little practical use. In this case, an equation-free approach [I.G. Kevrekidis, C.W. Gear, J.M. Hyman, P. Kevrekidis, O. Runborg, K. Theodoropoulos, Equation-free, coarse-grained multiscale computation: enabling microscopic simulators to perform system-level analysis, Commun. Math. Sci. 1 (4) (2003) 715–762] may provide effective methods for the analysis and simulation of individual-based models. In this paper we analyze equation-free coarse projective integration. For analytical purposes, we start with known partial differential equations describing biological random walks and we study the projective integration of these equations. In particular, we illustrate how to accelerate explicit numerical methods for solving these equations. Then we present illustrative kinetic Monte Carlo simulations of these random walks and show that a decrease in computational time by as much as a factor of a thousand can be obtained by exploiting the ideas developed by analysis of the closed form PDEs. The illustrative biological example here is chemotaxis, but it could be any random walker that biases its movement in response to environmental cues.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Chemotaxis; Individual-based models; Coarse integration; Projective integration

1. Introduction

In current complex systems modeling practice, we are often presented with a model at a *fine* level of description (atomistic, stochastic, individual-based), while we want to study the behavior at a macroscopic *coarse-grained* (continuum, population) level. This situation frequently arises in the modeling of biological dispersal, where significant progress is being made in modeling at the individual organism/cell level, while the derivation of the corresponding closed, macroscopic population level equations remains very difficult, and lags far behind in development. The example here is bacterial chemotaxis, for which much is known about signal transduction and motor behavior of individual cells, but only in a limited number of cases can one rigorously derive equations describing behavior of bacterial populations [11,12]. Usually one can develop a suitable cell-based stochastic model, and would like to obtain population-level information without having a coarse-grained evolution equation. Computational methods for obtaining an approximation to the macroscopic evolution without explicitly obtaining equations have been developed [25,16,18, 38]. The main idea is to use short bursts of appropriately-initialized computations using the detailed, fine-scale model, followed

---

* Corresponding author. Tel.: +1 612 624 8325; fax: +1 612 626 2017.
 E-mail addresses: erban@maths.ox.ac.uk (R. Erban), yannis@princeton.edu (I.G. Kevrekidis), othmer@math.umn.edu (H.G. Othmer).
 [1] Present address: Mathematical Institute, University of Oxford, 24–29 St. Giles', Oxford OX1 3LB, United Kingdom.

by processing of the results to obtain estimates of the desired macroscopic quantities such as the spatial distribution of the number density, time derivatives, and various measures of the sensitivity of the solution with respect to parameters.

The first, and probably most important step of this equation-free approach is to determine what are the appropriate variables in terms of which one could hope to close macroscopic evolution equations. Typically these variables are a few slowly-evolving lower moments of the many-particle distribution function (e.g., cell density for chemotactic movement [11], species concentrations for reaction–diffusion problems [15], or density and momentum fields, the zeroth and first moments of the distribution of molecules in velocity space, for the Navier–Stokes equations [6,7]). In most cases, knowledge of the level of closure (the number and identity of variables with which one can write a deterministic model for the process) comes from extensive experimental experience and observation, long before it is rigorously justified by theory. In the equation-free approach, the simplest conceptual path for selecting the appropriate *observables* as state variables is to perform a homotopy between conditions at which an accurate closed equation is known and validated, and the conditions of interest, when this is possible. Model reduction in general is based on the assumption that, after rapid initial transients, higher order moments of the evolving distributions can be approximately-represented as functionals of the slow, "master" ones — the ones in terms of which we write the closed equations. The closure is thus embodied in a "slow manifold": a graph of a function (in moment space) which, given the values of the few governing lower moments, provides the "slaved" higher order moment values. Separation of time scales between the rapid equilibration of the higher, slaved moments, and the slow evolution of the "master" ones underpins the derivation of closed, reduced, macroscopic equations. The idea then is to design computational experiments with the fine scale simulator that test for this separation of time scales, and suggest variables capable of parametrizing the slow manifold [38,31].

As is discussed in more detail in [38], it is possible, using matrix-free iterative linear algebra methods, to estimate, using direct simulation, characteristic relaxation time scales for the problem (at least in the neighborhood of a particular equilibrium). These time scales, and the eigendirections corresponding to them, can guide the modeler in deciding the *number*, and even the selection of variables capable of parametrizing (at least locally) this slow manifold. In this process, homotopy and knowledge of the appropriate parametrizing variables in some region of the operating parameter space, gives us a starting point for variable selection. In the more general, and much more difficult case in which we begin a completely new problem and have no initial knowledge of what might be good "order parameters" in terms of which to attempt to close macroscopic equations, the alternative is to use data processing techniques on extensive experimental (or computational experimental) runs, to try and develop a reasonable reduction hypothesis. Algorithms for data compression, from the more traditional principal component analysis to the more modern sparse kernel and diffusion map feature analysis may be useful here [39,31]. This is, however, a separate and active research subject in itself, and we will not pursue it here.

In this paper we will assume that we have enough knowledge of the problem to identify a set of variables in terms of which to write a closed equation. In that spirit we study the coarse integration of simple models for chemotaxis of cells, and we assume that the slow dynamics of the system are parametrized by cellular density. The main goal is to illustrate the computational gain of equation-free methods, by which we mean a large speed up of the stochastic simulation for a class of biologically-motivated problems involving slow dispersal of organisms/cells.

The paper is organized as follows. In Section 2, we present a brief overview of equation-free methods with emphasis on coarse projective integration. We present the main strategy which we will use for the analysis of coarse integration — namely the deterministic projective integration of partial differential equations (PDEs). Moreover, we show how the results of this paper can be interpreted in terms of equation-free *coarse* projective integration for kinetic Monte Carlo (kMC) simulations of random walks; and we define the gain of these methods. In Section 3 we present partial differential equations modeling the dispersal of cells, and we provide two biological motivations of the chemotaxis system studied later. We also discuss the main mathematical properties of these equations. Finally, we introduce a test family of spatial signal profiles which are used in the computational examples in Sections 4 and 5. In Section 4, we study the efficiency of projective integration for different discretizations of the macroscopic PDE equations. We obtain a measure of efficiency (gain) of the method for different choices of the "inner integrator". We demonstrate a stable, signal-independent method, i.e., a method which has the same gain for all mathematically admissible environmental changes. We also study more accurate inner integrators, for which the efficiency depends on the size of the environmental signal (concentration gradients). Section 4.5 contains illustrative numerical results; here we provide computations illustrating the analysis in Section 4 and give examples for which the method leads to a significant reduction in the computational time required. In Section 5 we return to the original random walk problem. We discuss the application of our approach to accelerating the Monte Carlo simulations and present a case in which the computational time is reduced by a factor of 10^3. Finally, in Section 6 we summarize the results, and mention significant generalizations. We conclude by reiterating the main elements of the equation-free approach as a "wrapper" around a usually slow, cell- or organism-based stochastic simulator, aimed at assisting in the efficient study of emergent, population-level behavior.

2. Equation-free methods — coarse integration

Consider a large collection of randomly-walking individuals for which we have a microscopic model, and suppose that we want to know the time evolution of the macroscopic density N of the individuals. One approach is to derive partial differential equation(s)
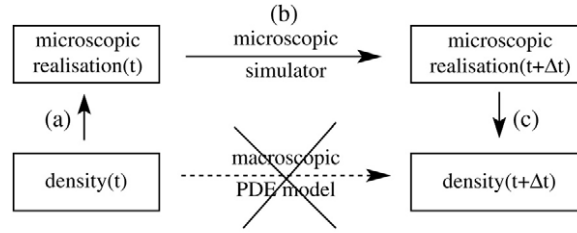
Fig. 1. Schematic of timestepper (a)–(c).

for macroscopic observables, such as the density N, and then compute the solution of the PDE(s) using standard numerical methods. This entails a choice of algorithm, a time step Δt, and a routine which computes the density $N(t + \Delta t)$ from the density $N(t)$.

If explicit macroscopic equations are not available, we can still compute the density of individuals at time $t + \Delta t$ from the density of individuals at time t using Monte Carlo simulation of the microscopic model. This can be done as follows.

(a) Given the macroscopic initial density $N(t)$, construct consistent microscopic initial conditions (initialize each individual so that the density is $N(t)$).
(b) Evolve the system using the microscopic Monte Carlo simulator for time Δt.
(c) Compute the density of individuals $N(t + \Delta t)$ from the microscopic data at time $t + \Delta t$.

Steps (a)–(c) provide an alternative path to computing $N(t + \Delta t)$ from $N(t)$ as illustrated in Fig. 1. The main goal is to compute the long time evolution of cellular density N, and to that end we could simply use step (b) many times, i.e., we could in principle run the microscopic simulator only. However, since the biological models are often complex, step (b) can be very computationally intensive. Thus the key constraint is that we are in fact able to run the microscopic simulator only for short times. Since we seek the long time evolution, we have to combine (a)–(c) with another step which can be formulated in many ways, e.g.,

(d) Using the macroscopic data for N computed in (a)–(c), estimate the time derivative $\frac{\partial N}{\partial t}(t + \Delta t)$. Because of fluctuations due to the stochastic nature of the simulation, we may require several independent microscopic realizations of $N(t)$ in part (a) to be able to accurately estimate the expected density $N(t + \Delta t)$ and its time derivative. We then take advantage of the assumed smoothness (in time) of the trajectory of the unavailable macroscopic evolution, and take a large projective step by estimating the density $N(t + \Delta t + T)$ for some $T > 0$ as

$$N(t + \Delta t + T) \approx N(t + \Delta t) + T \frac{\partial N}{\partial t}(t + \Delta t).$$

The density $N(t + \Delta t + T)$ is then used as a new initial condition in (a).

The algorithm (a)–(d) is called *coarse projective integration*, specifically, coarse projective forward Euler, and it can be formulated in many ways [16,18,25]. For example we can use different methods to estimate the time derivative of N in (d), or we can extrapolate using other macroscopic variables in part (d), e.g., with flux profiles as opposed to density profiles. In any case, the actual projective step is performed on some spatial discretization of these macroscopic variable profiles e.g., finite difference, finite element, or spectral decompositions of the profiles.

The algorithm (a)–(d) can speed up the computations provided that we can safely (in terms of stability and accuracy) choose $T \gg \Delta t$ and provided that the so-called "lift–run–restrict procedure" (a)–(c) does not require excessive computation to estimate the time derivative of N. In particular, the more time is spent in parts (a)–(c) of the algorithm, the larger $T/\Delta t$ in part (d) must be chosen to have the potential for computational gain. Since we also study modifications of (a)–(d), we will define the gain $\mathcal{G}$ of the coarse projective integration method as

$$\mathcal{G} = \frac{\text{time to compute the evolution of the system by running a Monte Carlo simulator only}}{\text{time to compute the evolution of the system by coarse projective integration (a)–(d)}}. \tag{2.1}$$

For example, if we need k realizations of the Monte Carlo evolution in steps (a)–(c) to compute the evolution of the system in the interval $[t, t + \Delta t]$, and if we assume that the computational time of step (d) is negligible, then the gain $\mathcal{G}$ can be simply estimated as $\mathcal{G} = \frac{T + \Delta t}{k \Delta t}$. On the other hand, one might argue that scaling by k may be too severe, since the equation we are evolving is not one for the single fluctuating realization, but for the expected density profile estimated, for example, as the average of k copies.

As a first, *illustrative* step in the analysis of the gain of coarse integration, we will replace the stochastic part (a)–(c) by a deterministic operator as shown in Fig. 2. This means that we assume that we *do* know, at least for some parameter regime, a closed macroscopic equation for the expected density profile of the particular kinetic Monte Carlo simulation. We then replace steps (a)–(c) by a short deterministic integration (i). We run this deterministic integrator only for a short time Δt, and process its results to obtain an extrapolation in time (ii); we then repeat the process. In this deterministic setup, we can more easily study the dependence of the gain on the parameters of the model and, in particular, the gap between slow and fast eigenvalues in the spectrum of the equation.
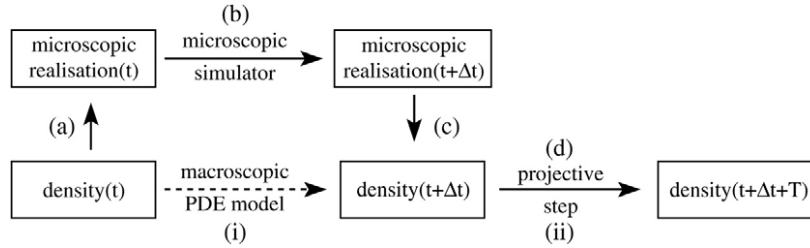
Fig. 2. We first analyse (i)–(ii) in parameter regimes where macroscopic equations are available.
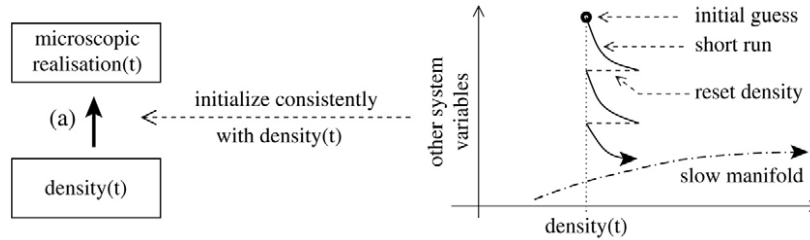

Fig. 3. Initialization of other state variables in step (a).

Assuming that most of the computational time is spent in part (i), we can rewrite the definition (2.1) in the deterministic setting (i)–(ii) as follows:

$$\mathcal{G} = \frac{T + \Delta t}{\Delta t}.$$

(2.2)

In the following section we introduce biologically-motivated problems for which the corresponding macroscopic equations are known for some, or for all, parameter regimes.

Finally, let us mention that step (a) requires that initialization of all system variables be done consistently with the density profile $N(t)$. This means that we initialize all individuals in such a way that the macroscopic density is equal to $N(t)$. There are many ways to accomplish this. Ideally, we would like to initialize the remaining macroscopic system observables (e.g. higher moments of the cell distribution function than density, the 0-th moment) on a slow manifold parametrized by the density profile $N(t)$ — that is, we would like to initialize them "slaved[2] to" $N(t)$. One possible procedure (which we use in Section 5) is schematically illustrated in Fig. 3. Here we make an initial guess for the remaining state variables and we run the Monte Carlo simulator for a short time only, then we reset the position of each individual to its initial value, keeping all other state variables unchanged. Repeating this procedure several times, we can find the initial condition close to the slow manifold of the system [20,21].

3. Chemotaxis

Many organisms that move in a random walk fashion respond to environmental signals by biasing their rules of movement. If we consider chemical signals in the environment, the corresponding motility behavior is called *chemotaxis* or *chemokinesis*, depending on whether the organism senses the direction of signal gradients directly, or responds by changing its speed or the frequency of turning. We will not distinguish between different terminologies, and we will call *chemotaxis* any alteration of behavior caused by the environmental cues; chemotaxis will be the illustrative biological example in this paper. At the population level, chemotaxis can lead to aggregation, travelling waves and pattern formation (see e.g., [5] for *E. coli*, [1,9] for *Dictyostelium discoideum*) and an important task is to explain population-level behavior in terms of individual-based models. To do that, equation-free methods may be suitable [37]. However our purpose here is to use the strategy for analysis outlined in Fig. 2, and to this end we choose a chemotactic example for which the macroscopic equations are known. First, in Section 3.1, we describe the simplified model of bacterial chemotaxis for which the macroscopic equations were derived in some parameter regimes [11,12]. Next, in Section 3.2, we present an even simpler random walk, which involves directional sensing, and is more suitable for modeling of certain eukaryotic organisms. Here the macroscopic equations can be derived for any choice of parameters. These equations have the same structure as in the bacterial case. In Section 4, we report the results of projective integration of the chemotaxis equations.

---

[2] The underlying idea is that the set of moments of the cell distribution constitutes a singularly perturbed system, characterized by time scale separation: higher order moments are assumed to quickly become functionals of the low, slow, governing ones, like density (i.e. they quickly approach a slow manifold parametrized by density). The same key assumption also underlies the analytical derivation of coarse-grained, macroscopic equations.

3.1. Bacterial chemotaxis

Flagellated bacteria, the best studied of which is *E. coli*, have two different modes of motile behavior that are determined by the rotation of their flagella. When rotated counterclockwise, the flagella coalesce into a propulsive bundle that produces a relatively straight "run". When rotated clockwise they fly apart and the bacterium "tumbles" without significant translocation. Hence, a bacterium runs at a constant velocity for a random length of time, then tumbles for a random length of time, chooses a new direction at random, and repeats the process. In order to find food or avoid noxious substances, a bacterium increases its runs in favorable directions and decreases them when going in an unfavorable direction. The run length is controlled by a complex biochemical network [2,40] that involves signal transduction and alteration of an intracellular protein called CheY that controls the direction of rotation of the flagellar motors, and consequently changes the movement of the bacterium.

In the absence of an extracellular signal the duration of both runs and tumbles are exponentially distributed, with means of 1 s and 10^{-1} s, respectively [3], and in a gradient of attractant the cell increases or decreases the run time, depending on whether it moves in a favorable or unfavorable direction. Since the tumbling time is small compared to the typical running time, we can describe the motion of *E. coli* as a *velocity jump process* [32], which means that a bacterium runs in some direction and, at random instants of time changes its velocity according to a Poisson process with mean turning rate γ. The turning rate is altered by CheY [8], so we can write $\gamma = \gamma(y_1)$ where y_1 denotes the concentration of the phosphorylated form of CheY.

Let $y = (y_1, y_2, \ldots, y_m) \in \mathbb{R}^m$ denote the intracellular variables, which can include the concentration of proteins, receptors, etc., and let $S(x, t) = (S_1, S_2, \ldots, S_M) \in \mathbb{R}^M$ denote the signals in the environment. Then existing deterministic models of bacterial signal transduction pathways can be cast in the form of a system of ordinary differential equations that describe the evolution of the intracellular state, forced by the extracellular signal. Thus

$$\frac{dy}{dt} = f(y, S) \tag{3.1}$$

where $f : \mathbb{R}^m \times \mathbb{R}^M \to \mathbb{R}^m$ describes the particular model. Eq. (3.1) is integrated along the trajectory of each cell, and the y_1 component of the solution together with the turning rate $\gamma = \gamma(y_1)$ defines the random walk of each bacterium.

As was noted in [11,12], existing models for signal transduction and models of flagellar motor behavior involve tens of chemical species, which makes the problem very complicated for analysis. However, the essential aspects of the dynamics can be captured by a much simpler "cartoon" model which involves just two variables. For the "cartoon" model, one can derive closed macroscopic equations for some parameter regimes (see [11] in 1D, see [12] in 2D/3D). In [11,12], Eq. (3.1) and equation for $\gamma(y_1)$ read as follows

$$\frac{dy_1}{dt} = \frac{g(S(x, t)) - (y_1 + y_2)}{t_e}, \quad \frac{dy_2}{dt} = \frac{g(S(x, t)) - y_2}{t_a}, \quad \gamma = \gamma_0 - \beta y_1, \tag{3.2}$$

where $t_e \ll t_a$ are constants, x is the current position of a cell, $S : \mathbb{R}^n \times [0, \infty) \to [0, \infty)$ is the concentration of the chemoattractant, and $g : [0, \infty) \to [0, \infty)$ models the first step of signal transduction. The constant $\gamma_0 > 0$ is the turning rate if no chemoattractant is present, which is changed by the linear function $-\beta y_1$, with $\beta > 0$, if attractant gradients are present.

In this paper, we restrict the random walks to *movement along the real line*, which means that individuals move to the left or to the right with constant speed s and, at random instants of time, change their direction with turning frequency γ. In this case, using (3.2) in suitable parameter regimes, one can derive a macroscopic partial differential equation for the density of individuals $N \equiv N(x, t)$ of the following form [11]

$$\frac{\partial^2 N}{\partial t^2} + 2\gamma_0 \frac{\partial N}{\partial t} = \frac{\partial}{\partial x} \left(s^2 \frac{\partial N}{\partial x} - g'(S(x)) \frac{2\beta s^2 t_a}{(1 + 2\gamma_0 t_a)(1 + 2\gamma_0 t_e)} S'(x) N \right). \tag{3.3}$$

The macroscopic equation (3.3) is valid for shallow gradients of the signal (small $S'(x)$) and for a suitable order of magnitude of the parameters involved (see [11] for details).

Since bacteria are too small to sense spatial gradients of the chemoattractant over their body lengths, they alter their turning rates as described above, to achieve the desired response to changes in chemoattractant concentration. On the other hand, eukaryotic unicellular organisms like *Dictyostelium discoideum* are large enough to sense directly the chemical gradients and respond to them appropriately. Motivated by this observation, in the following section we present a simple example of a 1D random walk of individuals such that a cell can sense directly the gradient of chemoattractant $S'(x)$ and respond with changes of its direction according to the gradient seen by the cell (see also [44]).

3.2. Chemotaxis with directional sensing

We consider the random movement of individuals which reduce their probability of changing direction when moving in a favorable direction, e.g., in the direction of increasing attractant. We suppose as above that a particle moves along the *x* axis

at a constant speed s, but that at random instants of time it reverses its direction according to a Poisson process with turning frequency

$$\gamma = \gamma_0 \pm bS'(x) \tag{3.4}$$

where b is a positive constant and the sign depends on the direction of the particle movement: plus for particles moving to the left, and minus for particles moving to the right. Let $R(x, t)$ (resp. $L(x, t)$) be the density of particles at (x, t) which are moving to the right (resp. left): then $R(x, t)$ and $L(x, t)$ satisfy the equations

$$\frac{\partial R}{\partial t} + s\frac{\partial R}{\partial x} = -(\gamma_0 - bS'(x))R + (\gamma_0 + bS'(x))L, \tag{3.5}$$

$$\frac{\partial L}{\partial t} - s\frac{\partial L}{\partial x} = (\gamma_0 - bS'(x))R - (\gamma_0 + bS'(x))L. \tag{3.6}$$

Equations of this type have been studied by many authors, and for a discussion of previous work see [33,22].

The density of particles at (x, t) is given by the sum $N(x, t) = R(x, t) + L(x, t)$, and the flux is $sR(x, t) - sL(x, t)$. We are primarily interested in the evolution of the macroscopic density N, and therefore we rewrite Eqs. (3.5) and (3.6) as the equations for the variables N and J given by

$$N = R + L, \quad J = R - L \quad \Leftrightarrow \quad R = \frac{N + J}{2}, \quad L = \frac{N - J}{2}, \tag{3.7}$$

where J is a rescaled flux.

Then adding and subtracting (3.5) and (3.6) gives

$$\frac{\partial N}{\partial t} + s\frac{\partial J}{\partial x} = 0, \tag{3.8}$$

$$\frac{\partial J}{\partial t} + s\frac{\partial N}{\partial x} = -2\gamma_0 J + 2bS'(x)N. \tag{3.9}$$

Thus the random walk can be described by the closed system of two equations (3.8) and (3.9) with given initial conditions $N(\cdot, 0)$ and $J(\cdot, 0)$.

Finally, assuming sufficient smoothness, we can convert (3.8) and (3.9) into a second order damped hyperbolic equation for N, namely

$$\frac{\partial^2 N}{\partial t^2} + 2\gamma_0 \frac{\partial N}{\partial t} = s^2 \frac{\partial^2 N}{\partial x^2} - 2bs\frac{\partial}{\partial x}\left(S'(x)N\right). \tag{3.10}$$

This is a hyperbolic version of the classical Keller–Segel equation [23,24]. Note that (3.10) has the same structure as (3.3), which can also be written as a system of two equations of the form (3.8) and (3.9). Therefore, the system (3.8) and (3.9) can also be viewed as a macroscopic description of bacterial chemotaxis.

3.3. Scaling and mathematical formulation of main problems

If we consider the system (3.8) and (3.9) as a description of the collective movement of bacteria *E. coli*, then we can give biologically realistic values for the parameters s and γ_0. The speed of a bacterium is $s \simeq 10 \; \mu m/s$ and the turning frequency is $\gamma_0 \simeq 1 \; s^{-1}$. To nondimensionalize equations (3.8) and (3.9), we choose the characteristic time scale $T_0 = \gamma_0^{-1}$, denote the characteristic space scale as L_0, and the characteristic concentration as N_0. Define

$$\hat{s} = \frac{sT_0}{L_0}, \quad \hat{S}'(x) = \frac{bS'(x)T_0}{L_0}, \quad \hat{N} = \frac{N}{N_0}, \quad \hat{J} = \frac{J}{N_0}, \quad \hat{t} = \frac{t}{T_0}, \quad \hat{x} = \frac{x}{L_0}. \tag{3.11}$$

Then the nondimensionalized equations (3.8) and (3.9) have the form

$$\frac{\partial \hat{N}}{\partial \hat{t}} + \hat{s}\frac{\partial \hat{J}}{\partial \hat{x}} = 0, \quad \frac{\partial \hat{J}}{\partial \hat{t}} + \hat{s}\frac{\partial \hat{N}}{\partial \hat{x}} = -2\hat{J} + 2\hat{S}'(x)\hat{N}, \tag{3.12}$$

and to simplify notation, we drop the hats in (3.12) and obtain the nondimensionalized system

$$\text{(NJ)} \begin{cases} \dfrac{\partial N}{\partial t} + s\dfrac{\partial J}{\partial x} = 0 \\ \dfrac{\partial J}{\partial t} + s\dfrac{\partial N}{\partial x} = -2J + 2S'(x)N. \end{cases}$$

Here we have one dimensionless parameter s and one dimensionless function $S'(x)$, and we estimate the orders of them as follows. In a typical macroscopic bacterial experiment the characteristic length scale L_0 is 1 cm, and since the characteristic time scale is $T_0 = \gamma_0^{-1} = 1$ s, we have $s \simeq 10^{-3}$. If the characteristic length scale is 10 cm then $s = 10^{-4}$, and in either case s is a small parameter. A realistic choice of $S'(x)$ must ensure that the turning rate (3.4) is positive, i.e., $|S'(x)| \leq 1$. Hence, we will assume throughout that

$$s \ll 1, \quad |S'(x)| \leq 1. \tag{3.13}$$

The system (NJ) is a linear hyperbolic system of two equations with nonconstant coefficients, which can be rewritten in diagonal form as a system of two equations for the right and left fluxes (cf. (3.5) and (3.6)). Thus, (NJ) can be rewritten as

$$(RL) \begin{cases} \dfrac{\partial R}{\partial t} + s \dfrac{\partial R}{\partial x} = -[1 - S'(x)]R + [1 + S'(x)]L \\ \dfrac{\partial L}{\partial t} - s \dfrac{\partial L}{\partial x} = [1 - S'(x)]R - [1 + S'(x)]L. \end{cases}$$

We also know that the system (NJ) can be written as a single second order equation for N (compare with (3.10)), or as the following system for the variables N and U

$$(NU) \begin{cases} \dfrac{\partial N}{\partial t} = U \\ \dfrac{\partial U}{\partial t} = s^2 \dfrac{\partial^2 N}{\partial x^2} - 2s \dfrac{\partial}{\partial x} \left(S'(x)N \right) - 2U. \end{cases}$$

In the following sections we study the system (NJ) or its equivalent formulations (RL) and (NU). We will restrict our computations to the finite interval [0, 2] with no flux boundary conditions which, in the formulation (NJ), can be written in the form

$$J(0, t) = \frac{\partial N}{\partial x}(0, t) = S'(0) = 0, \quad \text{and} \quad J(2, t) = \frac{\partial N}{\partial x}(2, t) = S'(2) = 0, \quad \text{for } t \geq 0. \tag{3.14}$$

As indicated here we also impose no-flux boundary conditions on the signal.

Finally, let us identify the dimensionless times of interest. The characteristic time scale was set as to the mean turning time, i.e., $T_0 = 1$ s, since that characterizes the microscopic dynamics, but the macroscopic times of interest in pattern formation experiments are several hours or days. From the mathematical point of view we are interested in the long term dynamics and steady states, and therefore we want to develop methods to compute the density profile $N(x, t)$ for dimensionless times $t \gg 1$.

3.4. Slow and fast variables and the slow manifold

In this section we consider spatial regions where the signal derivative is either zero or maximal possible (to assure a nonnegative turning rate). We show that in such regions the fluxes relax to functionals of the density for large times, i.e. the memory of the initial flux decays quickly. Thus the long-term dynamics can be described by a single first-order in time equation for the density N. Similar conclusions can be also made about systems (RL) and (NU). For example, in the case of (RL), we could characterize the long-term dynamics using only the right flux R, only the left flux L, or any linear combination of R and L (e.g., the density N). Knowing the density N, we can compute either (or both of) the right and left fluxes — alternatively, these fluxes quickly evolve to functionals of the density field; this constitutes our "slow manifold". The choice of the "right" observables can be made by the modeler; for historical (as well as practical) reasons we will use the density N in the following as a description of the slow variables.

3.4.1. Special choices of $S'(x)$
If $S'(x) = 0$, then system (NJ) can be rewritten as a second order damped wave equation

$$\frac{\partial^2 N}{\partial t^2} + 2 \frac{\partial N}{\partial t} = s^2 \frac{\partial^2 N}{\partial x^2}. \tag{3.15}$$

It is well-known [43] that the asymptotic behavior of the solution of (3.15) under the boundary conditions (3.14) is given by the corresponding diffusion equation

$$\frac{\partial N}{\partial t} = \frac{s^2}{2} \frac{\partial^2 N}{\partial x^2}. \tag{3.16}$$

Consequently, the long-term, slow dynamics can be described by this first-order in time equation for density only.

Next consider a spatial region where the signal gradient is the maximum possible, i.e., $S'(x) = 1$. If the region with maximal signal gradient is large enough, then (RL) in this region reduces to

$$\frac{\partial R}{\partial t} + s\frac{\partial R}{\partial x} = 2L, \quad \frac{\partial L}{\partial t} - s\frac{\partial L}{\partial x} = -2L. \tag{3.17}$$

Since the leftward flux L decays exponentially according to the second equation, the long-term behavior (in large spatial regions with $S'(x) = 1$) is given by the rightward flux only. Since $N = R + L$ and $L \to 0$, the long time dynamics is simply described by the first order transport equation

$$\frac{\partial N}{\partial t} + s\frac{\partial N}{\partial x} = 0. \tag{3.18}$$

A similar transport equation holds for the minimal possible signal gradient $S'(x) = -1$. Of course the boundary conditions (3.14) require that we cannot choose $S'(x) = 1$ in the whole domain of interest, and consequently, (3.18) only gives a good approximation of the behavior of cellular density in large spatial regions with maximal signal gradient. On the other hand, if we consider the random walk in a finite domain and we look for long term dynamics/stationary state then the no-flux boundary conditions (3.14) have to be taken into account.

3.4.2. (NJ) for general signals

For general signals, the behavior is just a combination of transport and diffusion as given by the second order equation (3.10). The steady state of (NJ) under no-flux boundary conditions is given by

$$s^2\frac{\partial^2 N}{\partial x^2} - 2s\frac{\partial}{\partial x}\left(S'(x)N\right) = 0,$$

and it follows that

$$N_s(x) = C\exp\left(\frac{2}{s}S(x)\right) \tag{3.19}$$

where the constant C is given by the initial condition for N. The interesting question is whether the behavior of (NJ) can indeed be described by a single first order equation for large times. The simplest choice is to use a parabolic counterpart of (3.10), given in dimensionless form as

$$\frac{\partial N}{\partial t} = \frac{s^2}{2}\frac{\partial^2 N}{\partial x^2} - s\frac{\partial}{\partial x}\left(S'(x)N\right). \tag{3.20}$$

Eq. (3.20) has the same steady state as (NJ), and moreover it reduces to (3.16) for constant signals. On the other hand, if $S'(x) = 1$, then Eq. (3.20) differs from (3.18) by the term $\frac{s^2}{2}\frac{\partial^2 N}{\partial x^2}$ which adds artificial diffusion to the system [41]. Consequently, if we have extended spatial regions where $S'(x) = 1$, then Eq. (3.20) gives different transient behavior than (NJ) but finally leads to the same steady state as (NJ). It is important to note that a major issue in equation-free computation is how many independent variables are needed in order to close with a *first-order in time* system, because it may be difficult to initialize microscopic variables consistently with given macroscopic observables and *their history* (e.g. their first order time derivatives). In regimes where at least a second-order in time equation is needed for closure, initializing the density is not enough; the time derivative of density must also be prescribed. In such a case we would use an alternative initialization for equation-free computations: we would prescribe right- and left-going fluxes R and L, which would be sufficient to start a particle-based simulation, because it is much easier to initialize particles based on more than one independent variable rather than based on the "history" of a single variable.[3]

3.5. Test family of signal functions

In later sections, several numerical computations are presented. Here we introduce the test family of signal functions which we will use in these illustrative examples. In all examples, we consider the problem (NJ) on the interval [0, 2] with no-flux boundary conditions, where the signal belongs to the one-parameter test family of signal functions given by

$$S_\alpha(x) = \alpha\overline{S}(x), \quad \text{for } \alpha \in [0, 1], \tag{3.21}$$

where $\overline{S}(x)$ is a fixed signal function and α scales the strength of the signal. The signal function $\overline{S}(x)$ is chosen in the following form (see also Fig. 4):

---

[3] In doing projective integration based on simulations over the entire spatial domain, the spatial order of the equation does not play a crucial role. If, however, one tries to use equation-free techniques such as the gaptooth scheme and patch dynamics [25,19,36], implementing effective matching conditions between patches becomes important, and that is crucially affected by the spatial order of the effective evolution equation. The design of computational experiments to determine the spatial order of an unknown (in closed form) equation is an interesting subject, discussed in part in [27].
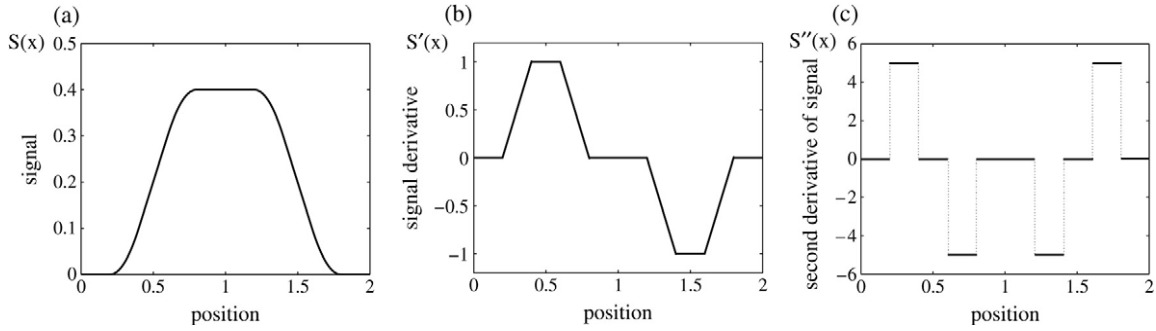
Fig. 4. (a) Graph of "hat-profile" signal function $\overline{S}(x)$. (b) Graph of $\overline{S}'(x)$. (c) Graph of $\overline{S}''(x)$.

Interval	$\left[0, \frac{1}{5}\right]$	$\left[\frac{1}{5}, \frac{2}{5}\right]$	$\left[\frac{2}{5}, \frac{3}{5}\right]$	$\left[\frac{3}{5}, \frac{4}{5}\right]$	$\left[\frac{4}{5}, \frac{6}{5}\right]$	$\left[\frac{6}{5}, \frac{7}{5}\right]$	$\left[\frac{7}{5}, \frac{8}{5}\right]$	$\left[\frac{8}{5}, \frac{9}{5}\right]$	$\left[\frac{9}{5}, 2\right]$
$\overline{S}(x)$	0	$\frac{(5x-1)^2}{10}$	$x - \frac{3}{10}$	$\frac{4-(5x-4)^2}{10}$	$\frac{4}{10}$	$\frac{4-(5x-6)^2}{10}$	$-x + \frac{17}{10}$	$\frac{(5x-9)^2}{10}$	0
$\overline{S}'(x)$	0	$5x - 1$	1	$4 - 5x$	0	$6 - 5x$	-1	$5x - 9$	0
$\overline{S}''(x)$	0	5	0	-5	0	-5	0	5	0

Since the maximal absolute value of the derivative $\overline{S}'(x)$ is equal to 1, the assumption (3.13) requires that $\alpha \in [0, 1]$, and $\alpha = 1$ means that the signal derivative $S'_\alpha(x)$ is maximal possible in some subintervals of the domain [0, 2]. For the $\overline{S}(x)$ chosen, the signal gradient $S'_\alpha(x)$ is zero in the intervals $[0, \frac{1}{5}]$, $[\frac{4}{5}, \frac{6}{5}]$ and $[\frac{9}{5}, 2]$, so the behavior will be similar to the diffusion equation there (for any α). If $\alpha = 1$ in (3.21), then the signal derivative $S'_1(x)$ is maximal possible, equal to 1, in the interval $[\frac{2}{5}, \frac{3}{5}]$; consequently, the right moving individuals will never turn in this interval and the corresponding coarse equation is a transport equation (3.18) there. Similarly, the signal gradient is minimal, equal to -1, in the interval $[\frac{7}{5}, \frac{8}{5}]$; consequently, the left moving individuals will never turn in this interval and the corresponding coarse equation is again the transport equation there.

4. Projective integration

The next objective is to study the so-called projective integration of the system (NJ), or its equivalent forms (RL) and (NU). To that end, we first summarize results from [17] about the *projective forward Euler method*. Suppose that we want to solve the initial value problem for the linear system of ordinary differential equations

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \mathcal{L}y, \quad y(0) = y_0, \tag{4.1}$$

where y is a n-dimensional vector and $\mathcal{L}$ is a $n \times n$ matrix of real numbers. Given constants k and M and step size δt, the projective forward Euler method (P_k^M) can be described as follows [17]:

(P_k^M-1) Use the forward Euler method[4] to integrate the system (4.1) over k time steps of the length δt to compute $y(t + k\delta t)$ from $y(t)$;
(P_k^M-2) perform one more integration step to compute $y(t + k\delta t + \delta t)$ from $y(t + k\delta t)$;
(P_k^M-3) perform an extrapolation over M steps, using $y(t + k\delta t + \delta t)$ and $y(t + k\delta t)$ to estimate $y(t + k\delta t + \delta t + M\delta t)$ as $y(t + k\delta t + \delta t + M\delta t) = (M + 1)y(t + k\delta t + \delta t) - My(t + k\delta t)$.

Thus, the procedure (P_k^M-1)–(P_k^M-3) integrates the system over the $(k + 1 + M)$ steps of the length δt. Next, we have the following result [17].

Lemma 1. *Method* (P_k^M-1)–(P_k^M-3) *for solving* (4.1) *is stable provided that the error amplification given by*

$$\sigma(\lambda \delta t) = [(M + 1)(1 + \lambda \delta t) - M](1 + \lambda \delta t)^k \tag{4.2}$$

satisfies $|\sigma(\lambda \delta t)| \leq 1$ *for all* λ *in the spectrum of the matrix* $\mathcal{L}$ *of system* (4.1).

Proof. See [17] where a more general linear stability analysis for systems of nonlinear ODEs is done. □

The absolute stability region in the complex $\lambda \delta t$-plane, which is plotted in Figs. 5 and 6(a), is the area inside the curve $|\sigma(\lambda \delta t)| = 1$. We see that the region splits into two parts for large M. Consequently, the constant M can be large if the spectrum is concentrated

---

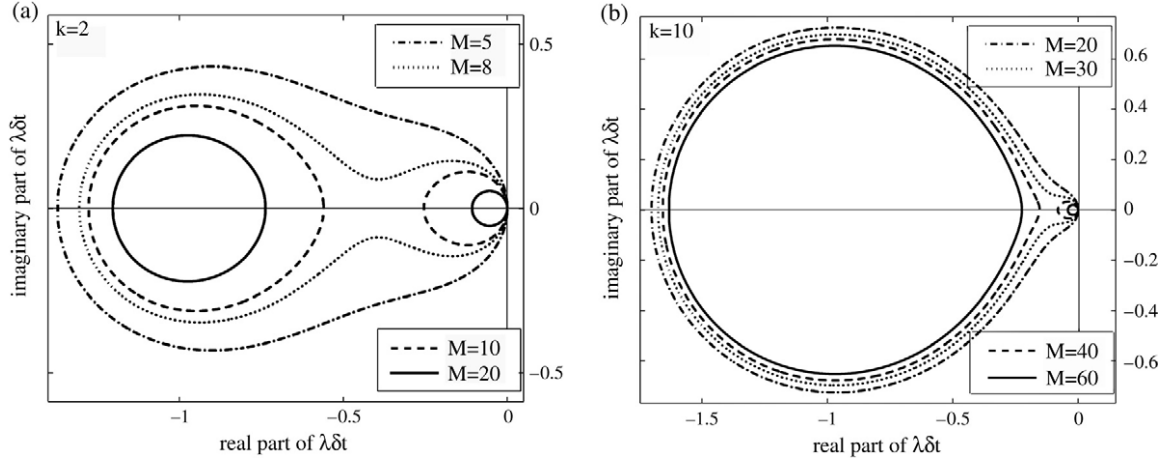[4] In fact any other integration scheme can be used here.

Fig. 5. (a) The regions of absolute stability of P_k^M methods for $k = 2$ and $M = 5$ (dot–dashed line), $M = 8$ (dotted line), $M = 10$ (dashed line) and $M = 20$ (solid line). (b) The regions of absolute stability of P_k^M methods for $k = 10$ and $M = 20$ (dot–dashed line), $M = 30$ (dotted line), $M = 40$ (dashed line) and $M = 60$ (solid line).
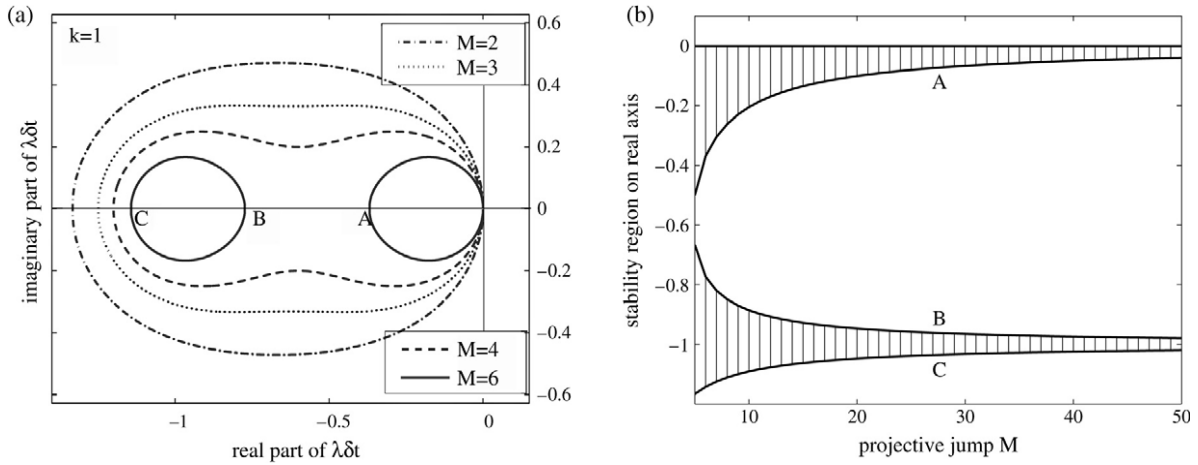

Fig. 6. (a) The regions of absolute stability of P_k^M methods for $k = 1$ and $M = 2$ (dot–dashed line), $M = 3$ (dotted line), $M = 4$ (dashed line) and $M = 6$ (solid line). (b) Intersection of stability region from part (a) with real axis plotted as a function of M. The equations for boundary curves $A(M)$, $B(M)$ and $C(M)$ are given in Lemma 2.

into two widely-separated regions corresponding to the fast and slow components. We also see that we increase the part of the stability region corresponding to fast components if we increase the number of inner integration steps k. The stability region for $k = 1$ is given in Fig. 6(a), for $k = 2$ in Fig. 5(a) and for $k = 10$ in Fig. 5(b).

In the following sections, we discretize the PDEs using the method of lines. Some of the systems we study will have a real-valued spectrum for parameter values of interest. Consequently, the interesting part of the stability region from Fig. 6(a) is its intersection with the real axis. For large M, the real stability region comprises the union of two intervals given by Lemma 2 for $k = 1$, and plotted in Fig. 6(b).

Lemma 2. *Suppose that the eigenvalues of the matrix $\mathcal{L}$ are all real. Then the procedure $(P_k^M\text{-}1)$–$(P_k^M\text{-}3)$ with $k = 1$ and $M \geq 5$ for solving (4.1) is stable provided that*

$$\lambda \delta t \in (C, B) \cap (A, 0), \quad \text{for all } \lambda \text{ in the spectrum of } \mathcal{L}, \tag{4.3}$$

where $C < -1 < B < A < 0$ are given by

$$C = -1 - \frac{1}{M+1}, \quad B = -\frac{M+2+\sqrt{(M-2)^2-8}}{2(M+1)}, \quad \text{and} \quad A = -\frac{M+2-\sqrt{(M-2)^2-8}}{2(M+1)}. \tag{4.4}$$

Proof. This is an easy consequence of Lemma 1. ☐

From Fig. 6(b) we see that (in the case of a real spectrum) one can choose a large projective jump M provided that the spectrum of $\mathcal{L}$ lies in two small intervals, separated by a spectral gap. Later, we will see such linear systems arising in our simulations; the

natural question then is: if we know (or can estimate) the spectrum of $\mathcal{L}$, what is the maximal possible choice of M such that the P_k^M method is stable? The answer is given in the following lemma.

Lemma 3. *Suppose that eigenvalues of the matrix $\mathcal{L}$ are all real. Let $-2 < c < -1 < b < a < 0$ and $\delta t > 0$ be given constants such that*

$$\lambda \delta t \in (c, b) \cap (a, 0), \quad \text{for all } \lambda \text{ in the spectrum of } \mathcal{L}. \tag{4.5}$$

Then $(P_k^M\text{-}1)$–$(P_k^M\text{-}3)$ is stable for all M satisfying the inequality

$$M \leq \min \left\{ \left| \frac{1 + (1+a)^{k+1}}{a(1+a)^k} \right|, \left| \frac{1 + (1+b)^{k+1}}{b(1+b)^k} \right|, \left| \frac{1 - (1+c)^{k+1}}{c(1+c)^k} \right| \right\}. \tag{4.6}$$

Proof. The amplification factor (4.2) is given by the formula

$$\sigma(\lambda \delta t) = M \lambda \delta t (1 + \lambda \delta t)^k + (1 + \lambda \delta t)^{k+1}.$$

In order to have a stable method, the following three inequalities must be satisfied simultaneously:

$$\sigma(a) \geq -1, \quad \sigma(b) \geq -1, \quad \text{and} \quad \sigma(c) \leq 1.$$

Solving for M, we obtain (4.6). $\quad \square$

Finally, let us note that the results of this section could be also viewed as results of linear stability analysis of projective integration of general nonlinear systems of ODEs of the form $y' = F(y)$, $y(0) = y_0$, where y is a n-dimensional vector and $F : \mathbb{R}^n \to \mathbb{R}^n$ [17].

4.1. Projective integration of chemotaxis systems (NJ), (RL) and (NU)

Before we implement coarse projective integration, we illustrate the use of projective integration and the factors affecting its implementation and effectiveness through the use of discretizations of the chemotaxis equations themselves. In this context, it is convenient to think that we only have available as an "inner simulator" a black-box dynamic integrator with a small, fixed time step (for example, a forward Euler simulator of a discretization of the problem), and that we are attempting to accelerate this black box code.

Given a signal profile $S(x)$, the speed s and initial conditions, we will look for the solution of (NJ) in the finite interval $[0, 2]$ with no flux boundary conditions (3.14). To do that, we will discretize (NJ) and rewrite it as a system of ordinary differential equations of the form (4.1) using the method of lines. The resulting system of ODEs is a starting point for our basic projective integration algorithm, which is little different than $(P_k^M\text{-}1)$–$(P_k^M\text{-}3)$. It is based on the sketch in Fig. 2. Choosing a suitable time step δt, and constants k and M, the algorithm is given in the following three steps $(Pr_k^M\text{-}1)$–$(Pr_k^M\text{-}3)$. Note that the steps $(Pr_k^M\text{-}1)$–$(Pr_k^M\text{-}2)$ correspond to step (i) as outlined in Fig. 2, and the last step $(Pr_k^M\text{-}3)$ corresponds to step (ii) in Fig. 2.

$(Pr_k^M\text{-}1)$ integrate system (NJ) over k time steps of length δt to compute $N(t + k\delta t)$ and $J(t + k\delta t)$ from $N(t)$ and from suitably initialized flux $J(t)$ — see (4.9) and (4.10);

$(Pr_k^M\text{-}2)$ perform one more inner integration step to compute $N(t + k\delta t + \delta t)$ and $J(t + k\delta t + \delta t)$ from $N(t + k\delta t)$ and $J(t + k\delta t)$;

$(Pr_k^M\text{-}3)$ perform an extrapolation over M steps, using $N(t + k\delta t + \delta t)$ and $N(t + k\delta t)$ to compute $N(t + k\delta t + \delta t + M\delta t) = (M + 1)N(t + k\delta t + \delta t) - MN(t + k\delta t)$.

Note that we can approximate the time derivative of N in step $(Pr_k^M\text{-}2)$ by

$$\frac{\partial N}{\partial t} = \frac{N(t + k\delta t + \delta t) - N(t + k\delta t)}{\delta t}$$

and therefore step $(Pr_k^M\text{-}3)$ is equivalent to

$$N(t + k\delta t + \delta t + M\delta t) = N(t + k\delta t + \delta t) + M\delta t \frac{\partial N}{\partial t}$$

which is the forward Euler projective step. We see that step $(Pr_k^M\text{-}3)$ is really equivalent to step (ii) in Fig. 2. It is important to notice that integrating the full system (NJ) requires initialization not only of the density N (which is prescribed) but also of the flux J, which is not; this will be discussed further below. As we mentioned in Section 2, the coarse/projective integration method is efficient provided that we can choose a large projective time T in step (d) in Fig. 2 relative to the time Δt of the steps (a)–(c) from Fig. 2 and still retain accuracy and stability. Using the notation from Section 2, we have

$$\Delta t = k\delta t + \delta t, \quad T = M\delta t, \tag{4.7}$$

and consequently, the gain $\mathcal{G}$ of the method (2.2) can be expressed as

$$\mathcal{G} = \frac{T + \Delta t}{\Delta t} = \frac{M + k + 1}{k + 1}. \tag{4.8}$$

Our goal is to make this gain as large as possible. Moreover, in order to use the scheme $(\mathrm{Pr}_k^M\text{-}1)$–$(\mathrm{Pr}_k^M\text{-}3)$, we have to specify the spatial discretization of (NJ). We study two options in Section 4.2. Finally, we also have to specify how we initialize the flux in step $(\mathrm{Pr}_k^M\text{-}1)$. There are several possibilities for doing this, the easiest of which is to use the initial flux $J(t)$ in step $(\mathrm{Pr}_k^M\text{-}1)$ given by

$$J(t) = 0. \tag{4.9}$$

We can also use as an initial guess the value of the flux computed in the previous step $(\mathrm{Pr}_k^M\text{-}2)$ corresponding to a time $M\delta t$ ago, i.e., before the projective jump. Thus we could use

$$J(t) = \text{"flux } J(t - M\delta t) \text{ which was computed in the previous step } (\mathrm{Pr}_k^M\text{-}2)\text{"}. \tag{4.10}$$

A more sophisticated flux initialization is used in Section 5, which deals with Monte Carlo simulations (see also Fig. 3).

4.2. Discretization of (RL) and (NU)

Various possibilities exist for discretizing the system (NJ) in the spatial domain; we start with one which is based on the equivalent form (RL) and on upwinding. The advantage of upwinding is that it provides a more stable scheme for problems with a significant convection component, but it introduces artificial diffusion into the problem [41]. Another possibility to spatially discretize (NJ), (RL) or (NU) is to use central differences, which leads to Eq. (4.16).

First, to solve the system (NJ) numerically, we transform it to the system (RL) of two first order equations in diagonal form. We want to solve (RL) over the interval $[0, 2]$ with boundary conditions given by (3.14). We choose a number n and a mesh size $\delta x = 2/n$, and we discretize the interval $[0, 2]$ with $n + 1$ mesh points

$$x_k = k \cdot \delta x, \quad \text{for } k = 0, \ldots, n. \tag{4.11}$$

Next, we define

$$R_i(t) = R(x_i, t), \quad L_i(t) = L(x_i, t), \quad \text{and} \quad S_i' = S'(x_i), \quad i = 0, \ldots, n.$$

The zero flux boundary conditions (3.14) simply mean that $R_0 = L_0$ and $R_n = L_n$; consequently, we have to compute the time evolution of the $2n$-dimensional vector

$$w = (R_1, R_2, \ldots R_{n-1}, R_n, L_0, L_1, L_2, \ldots, L_{n-1})^T. \tag{4.12}$$

To discretize spatial derivatives in (RL), we use upwinding, that is,

$$\frac{\partial R}{\partial x}(x_i, t) \approx \frac{R_i(t) - R_{i-1}(t)}{\delta x}, \quad \frac{\partial L}{\partial x}(x_i, t) \approx \frac{L_{i+1}(t) - L_i(t)}{\delta x}.$$

Then, the solution of (RL) with boundary conditions (3.14) is approximated by the solution of a system of ordinary differential equations

$$\frac{\mathrm{d}w}{\mathrm{d}t} = \mathcal{A}w, \quad w(0) = w_0, \tag{4.13}$$

where w_0 is a given initial condition and matrix $\mathcal{A}$ is defined by

$$\mathcal{A} = \begin{pmatrix}
-1 - \varepsilon + S_1' & 0 & \cdot\cdot & 0 & 0 & \varepsilon & 1 + S_1' & 0 & \cdot\cdot & 0 \\
\varepsilon & -1 - \varepsilon + S_2' & \cdot\cdot & 0 & 0 & 0 & 0 & 1 + S_2' & \cdot\cdot & 0 \\
\cdot & \cdot & \cdot\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot\cdot & \cdot \\
0 & 0 & \cdot\cdot & -1 - \varepsilon + S_{n-1}' & 0 & 0 & 0 & 0 & \cdot\cdot & 1 + S_{n-1}' \\
0 & 0 & \cdot\cdot & \varepsilon & -\varepsilon & 0 & 0 & 0 & \cdot\cdot & 0 \\
0 & 0 & \cdot\cdot & 0 & 0 & -\varepsilon & \varepsilon & 0 & \cdot\cdot & 0 \\
1 - S_1' & 0 & \cdot\cdot & 0 & 0 & 0 & -1 - \varepsilon - S_1' & \varepsilon & \cdot\cdot & 0 \\
0 & 1 - S_2' & \cdot\cdot & 0 & 0 & 0 & 0 & -1 - \varepsilon - S_2' & \cdot\cdot & 0 \\
\cdot & \cdot & \cdot\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot\cdot & \cdot \\
0 & 0 & \cdot\cdot & 1 - S_{n-1}' & \varepsilon & 0 & 0 & 0 & \cdot\cdot & -1 - \varepsilon - S_{n-1}'
\end{pmatrix}$$

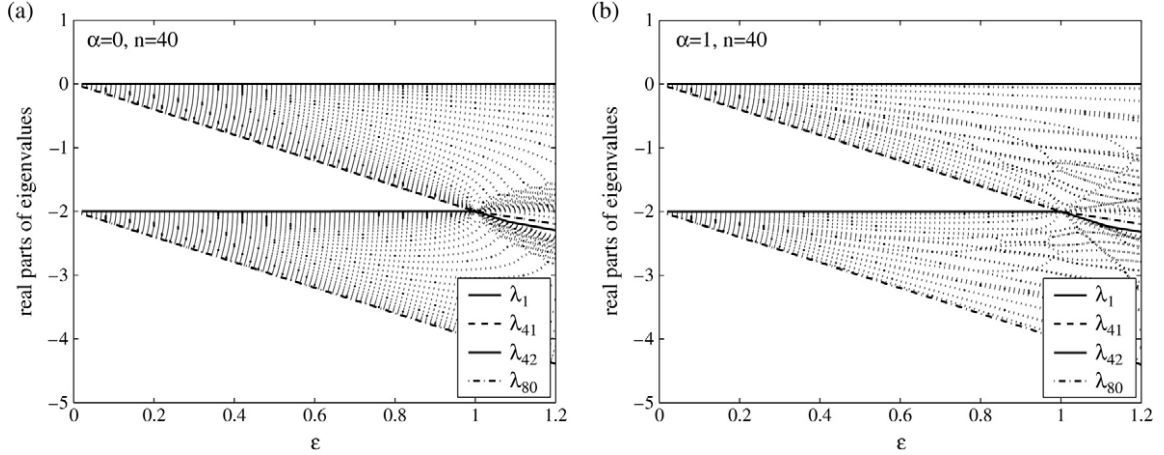where

$$\varepsilon \equiv \frac{s}{\delta x}. \tag{4.14}$$

Fig. 7. (a) Graph of real parts of eigenvalues of matrix $\mathcal{A}$ for $n = 40$ and no signal gradient in the environment, i.e., $S'_0 = S'_1 = \cdots S'_n = 0$. The eigenvalues are real for $\varepsilon \in [0, 1]$ and there is a spectral gap between λ_{41} and λ_{42}. (b) Graph of real parts of eigenvalues of matrix $\mathcal{A}$ for signal given by (3.21)) with $\alpha = 1$ and for $n = 40$.

Since we have approximated the original PDE system as a system of ordinary differential equations of the form (4.1), the results in Lemmas 1–3 can be applied. Alternatively, we can discretize the chemotaxis system in its equivalent form (NU) using standard central differences to approximate spatial derivatives in (NU). We use $n + 1$ mesh points (4.11) and we define

$$N_i(t) = N(x_i, t), \quad U_i(t) = U(x_i, t), \quad \text{and} \quad S'_i = S'(x_i), \quad i = 0, \ldots, n,$$

$$z = (N_0, N_1, N_2, \ldots, N_{n-2}, N_{n-1}, N_n, U_0, U_1, U_2, \ldots, U_{n-2}, U_{n-1}, U_n)^T, \tag{4.15}$$

$$\mathcal{D} = \begin{pmatrix} -2\varepsilon^2 & 2\varepsilon^2 - \varepsilon S'_1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \varepsilon^2 & -2\varepsilon^2 & \varepsilon^2 - \varepsilon S'_2 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \varepsilon^2 + \varepsilon S'_1 & -2\varepsilon^2 & \varepsilon^2 - \varepsilon S'_3 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \varepsilon^2 + \varepsilon S'_2 & -2\varepsilon^2 & \cdots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & \cdots & -2\varepsilon^2 & \varepsilon^2 - \varepsilon S'_{n-1} & 0 \\ 0 & 0 & 0 & 0 & \cdots & \varepsilon^2 + \varepsilon S'_{n-2} & -2\varepsilon^2 & \varepsilon^2 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 2\varepsilon^2 + \varepsilon S'_{n-1} & -2\varepsilon^2 \end{pmatrix}, \quad \mathcal{B} = \begin{pmatrix} 0 & I \\ \mathcal{D} & -2I \end{pmatrix},$$

where I is a $(n + 1) \times (n + 1)$ identity matrix and ε is given by (4.14). Then, the solution of (NU) with boundary conditions (3.14) is approximated by the solution of a system of ordinary differential equations

$$\frac{\mathrm{d}z}{\mathrm{d}t} = \mathcal{B}z, \quad z(0) = z_0, \tag{4.16}$$

where z_0 is a prescribed initial condition.

4.3. Efficiency of projective integration

First, suppose that there is no signal gradient in the domain of interest, i.e., we put $S'_0 = S'_1 = \ldots S'_n = 0$ in matrices $\mathcal{A}$ and $\mathcal{B}$. Choosing $n = 40$, the real parts of the eigenvalues of $\mathcal{A}$ and $\mathcal{B}$ as a function of ε are plotted in Figs. 7(a) and 8(a), respectively. We see that there is a clear spectral gap for small ε. The eigenvalues of $\mathcal{A}$ are all real for $\varepsilon \in [0, 1]$ and they satisfy

$$\lambda \in [-2\varepsilon, 0] \bigcup (-2 - 2\varepsilon, -2). \tag{4.17}$$

The eigenvalues of $\mathcal{B}$ are all real for $\varepsilon \in [0, 0.5]$ and for no signal in the environment and they satisfy

$$\lambda \in \left[-1 + \sqrt{1 - 4\varepsilon^2}, 0 \right] \bigcup \left[-2, -1 - \sqrt{1 - 4\varepsilon^2} \right]. \tag{4.18}$$

The spectral gap between -2ε and -2 in the case of matrix $\mathcal{A}$ is independent of the signal as can be seen from Fig. 7(b), where we use the signal profile (3.21) with $\alpha = 1$. We see that some eigenvalues changed, but that the spectral gap between λ_{41} and λ_{42} survived. The imaginary parts of the eigenvalues do not grow significantly with α, and consequently the values of the real parts determine the stability of the scheme; we can use results from Lemma 3 for matrix $\mathcal{A}$ and small $\varepsilon \in [0, 1)$. To do that, we specify
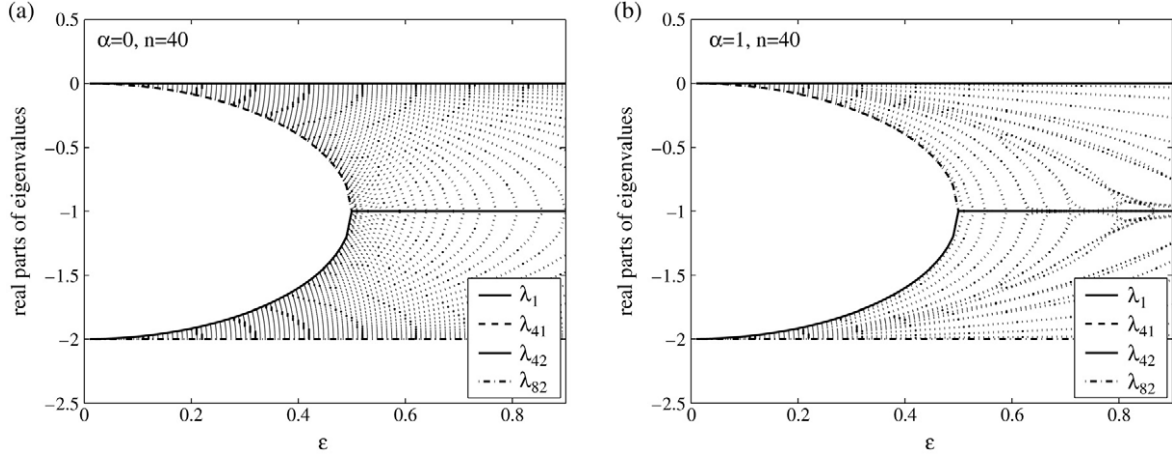
Fig. 8. (a) Graph of real parts of eigenvalues of matrix $\mathcal{B}$ for $n = 40$ and no signal gradient in the environment, i.e., $S'_0 = S'_1 = \cdots S'_n = 0$. The eigenvalues are real for $\varepsilon \in [0, 0.5]$ and there is a spectral gap between λ_{41} and λ_{42}. (b) Graph of real parts of eigenvalues of matrix $\mathcal{B}$ for signal given by (3.21) with $\alpha = 1$ and for $n = 40$.
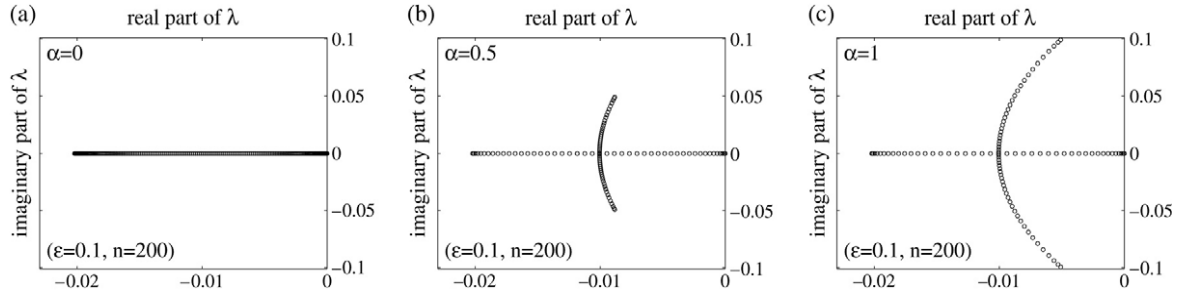

Fig. 9. A plot of "slow" eigenvalues of matrix $\mathcal{B}$ for $n = 200$ and $\varepsilon = 0.1$. We plotted only eigenvalues close to zero for different strengths of the signal α from (3.21), namely: (a) $\alpha = 0$, (b) $\alpha = 0.5$ and (c) $\alpha = 1$.

the time step δt. Since we want $\lambda \delta t$ close to -1 for eigenvalues corresponding to fast modes, we put

$$\delta t = 0.5. \tag{4.19}$$

Considering our scaling (3.11), we see that (4.19) means that δt is equal to time $1/(2\gamma_0)$. Next, if k is at least 2, then the "component" of the stability region around -1 is more extended than its second component around 0 (see Fig. 5). Consequently, using Lemma 3, the size of interval containing the slow eigenvalues determines the gain $\mathcal{G}$ of the method. Using (4.17) for the matrix $\mathcal{A}$, we have

$$\mathcal{G} = \frac{1 + (1 - \varepsilon)^{k+1}}{\varepsilon(1 - \varepsilon)^k(k + 1)}, \quad \text{which is approximately } \frac{2}{\varepsilon(k + 1)} \text{ for small } \varepsilon. \tag{4.20}$$

Note that the gain $\mathcal{G}$, given by (4.20) is *independent* of the signal strength α and it can be very large for small ε. On the other hand, as we will see in Section 4.4, the choice of small ε will decrease the *accuracy* of the upwind discretization $\mathcal{A}$ due to the strong artificial diffusion of the scheme.

Next consider the matrix $\mathcal{B}$. The real parts of its eigenvalues as functions of ε are plotted in Fig. 8. We see that the "boundary" eigenvalues -2, $-1 + \sqrt{1 - 4\varepsilon^2}$, $-1 + \sqrt{1 - 4\varepsilon^2}$, 0 of $\mathcal{B}$ are signal independent. The eigenvalues are all real for $\varepsilon \in [0, 0.5]$ and for no signal in the environment. However, if we increase the signal strength α some eigenvalues become complex, as can be seen from Fig. 9, where we plot the "slow" eigenvalues close to zero in the complex plane for $n = 200$, $\varepsilon = 0.1$, and for different signal strengths. Choosing δt by (4.19) and k, we can (for small signals) apply the results of Lemma 3 to compute the maximal possible projective jump M and, hence, to compute the gain $\mathcal{G}$ by (4.8) for small ε and for small signals. Using (4.18) and Lemma 3 for the matrix $\mathcal{B}$ for small signal gradients, we have

$$\mathcal{G} = \frac{1 + (1 - 4\varepsilon^2)^{(k+1)/2}}{(-1 + \sqrt{1 - 4\varepsilon^2})(1 - 4\varepsilon^2)^{k/2}(k + 1)}, \quad \text{which is approximately } \frac{1}{\varepsilon^2(k + 1)} \text{ for small } \varepsilon. \tag{4.21}$$

We see that discretization (4.16) results in a very large gain $\mathcal{G}$ for small ε and for small signal gradients. On the other hand, if we increase the signal gradients, then the result (4.21) is no longer true, because complex eigenvalues can appear outside the stability
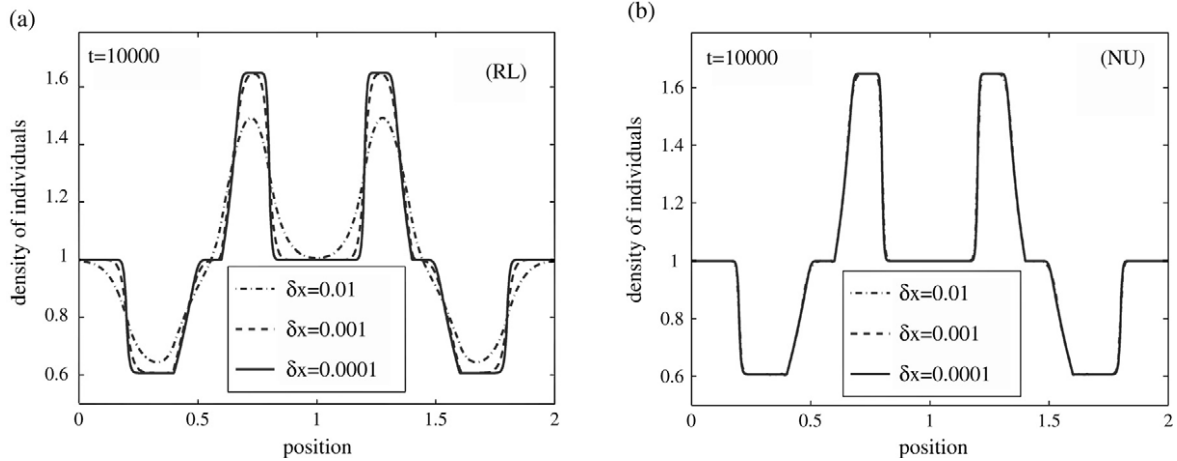
Fig. 10. (a) Graph of a solution of (RL) given by $\dot{w} = \mathcal{A}w$ at time $t = 10^4$ for different choices of δx. We used $s = 0.0001$, signal given by (3.21) with $\alpha = 0.1$, δt given by (4.19) and initial conditions (4.23). Consequently, $\delta x = 0.01$, $\delta x = 0.001$ and $\delta x = 0.0001$ correspond to $\varepsilon = 0.01$, $\varepsilon = 0.1$, and $\varepsilon = 1$, respectively. (b) Graph of a solution of (NU) given by $\dot{z} = \mathcal{B}z$ at time $t = 10^4$ for different choices of δx. The parameters are the same as in (a).

region (compare Fig. 9 and Fig. 5). For example, we see from Fig. 9(c) that the slow eigenvalues lie in the complex interval $[-0.02, 0] \times [0.1i, 0.1i]$ for $\alpha = 1$. Consequently, the absolute values of the imaginary parts of the eigenvalues are much larger than the absolute values of the real parts and the result (4.21) is not applicable for large signals.

4.4. Accuracy of projective integration

As we see in (4.14), (4.20) and (4.21), choosing a larger δx will make ε smaller and we will have a larger gain $\mathcal{G}$ for the projective integration. On the other hand, a smaller δx will increase the accuracy of the numerical method obtained by (4.13) or (4.16). The right choice of δx depends on the underlying signal. If we have signals with sharp second derivatives and if we want to capture the detailed transient behavior accurately, we have to use a sufficiently small δx. However, if we want to make use of the spectral gaps (4.17) or (4.18), we must assure that $s \ll \delta x$ to have $\varepsilon \ll 1$.

Two types of errors arise in these computations: (1) the error between the projective integration of (4.13) or (4.16) and the corresponding solutions of (4.13) or (4.16), respectively; and (2) the error between solutions of (4.13) or (4.16) and the exact solution of (NJ). The error in part (1) is sufficiently small as will be seen in Section 4.5; it is easy to estimate this error here, since the exact solution of (4.13), (4.16) or even of (NJ) can be found through careful, error-controlled computations. For microscopic simulations though, when the corresponding macroscopic equation is not known, estimating these errors becomes an important task; fortunately, numerical analysis techniques for *on line* a posteriori error estimates have been extensively developed for continuum problems, and can be naturally incorporated in equation-free computation [13]. For example, comparing results of the same computation with half the projective time step can be used to estimate the error of the scheme and control projective time step selection; comparable techniques for adaptive spatial meshing can also be used. It is, however, important to note one "twist" to traditional a posteriori numerical error estimates: errors due to the estimation scheme, e.g., due to fluctuations in stochastic simulations; this can be controlled through variance reduction schemes, either by brute force computation of several replica simulations or possibly through biasing for variance reduction [30]. Beyond adaptive time steps, adaptive mesh sizes and possibly variance reduction, we will discuss at the end of the paper the adaptive check of the *level* at which a macroscopic description closes, i.e., the number of macroscopic variables required, or the dimension of the "slow manifold".

We will now discuss errors of type (2), i.e., errors between solutions of (4.13) or (4.16) and the exact solution of (NJ). We can numerically estimate those errors by comparing the solution of (4.13) or (4.16) for different δx. Representative results can be found in Fig. 10 where we used $s = 0.0001$, a signal given by (3.21) with $\alpha = 0.1$ and δt given by (4.19). As is well known, the upwinding (4.13) discretization introduces artificial diffusion to the problem which makes the solution more dependent on δx. The central differences discretization (4.16) is more accurate here [41].

4.5. Numerical examples

Here we present illustrative numerical results. In view of (3.13), we choose

$$s = 0.0001, \quad \delta x = 0.01, \quad \varepsilon = \frac{s}{\delta x} = 0.01; \tag{4.22}$$

and we consider 201 mesh points (4.11) in the interval $[0, 2]$. The time step δt is given by (4.19) and the initial condition is
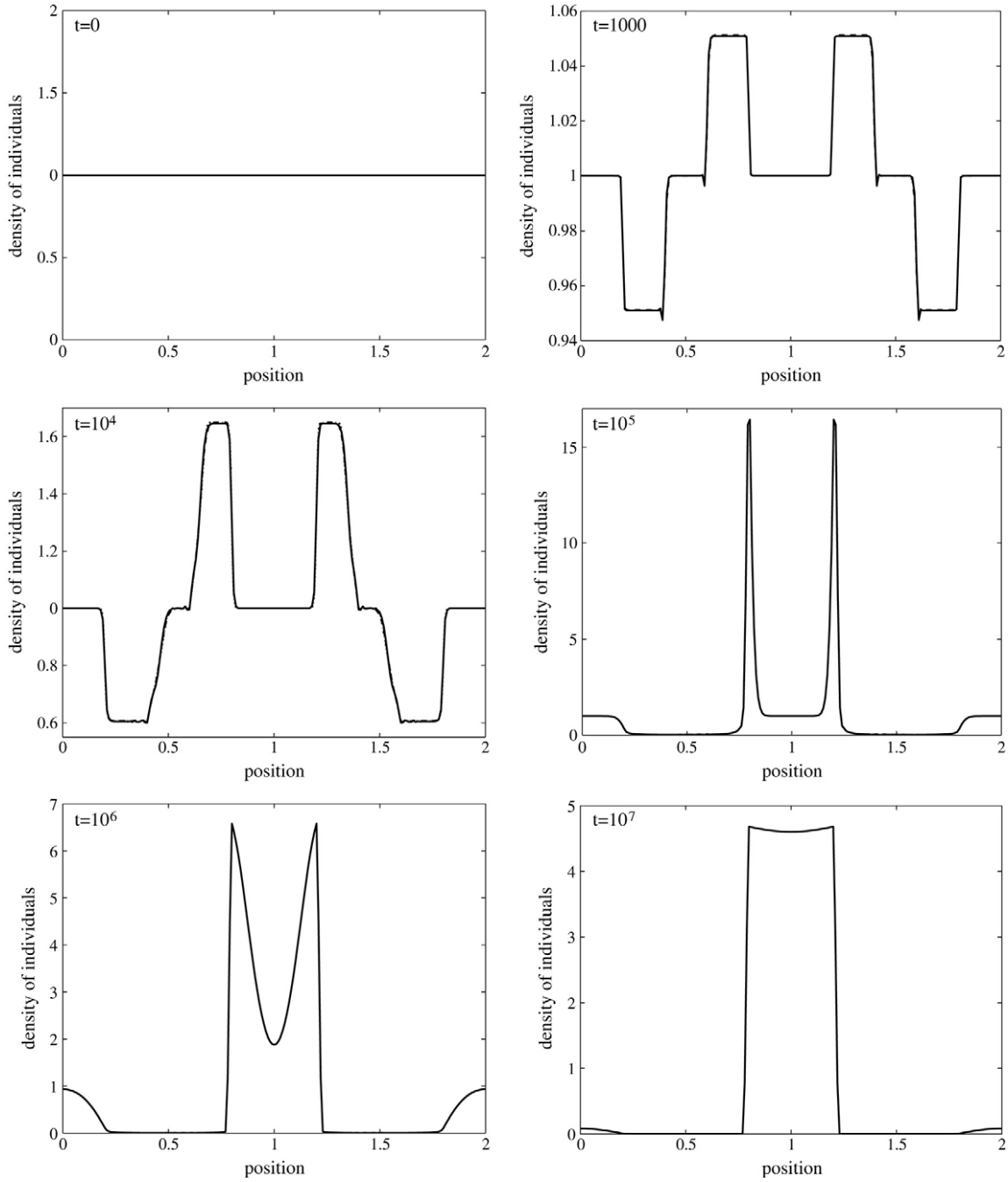
$$N(x, 0) = 1, \quad J(x, 0) = 0. \tag{4.23}$$

Fig. 11. The time evolution of the density of individuals for $s = 0.0001$ and $\alpha = 0.1$. We plot the solution of the system (4.16) (dashed line) and the solution obtained by the projective algorithm $(P_k^M$-1)–$(P_k^M$-3) for (4.16) with $k = 1$ and $M = 398$ (solid line). The gain $\mathcal{G}$ is 200. We use (4.19), (4.22) and (4.23).

We know from Section 4.4 that the discretization (4.16) gives rise to a sufficiently accurate solution of (NJ) for the choice of parameters (4.22), so we start with the discretization (4.16) first. We learned in Fig. 9 that we can have a large gain $\mathcal{G}$ of using projective integration of (4.16) if the signal gradient is small; consequently, we consider the signal (3.21) with $\alpha = 0.1$. The numerical results for $k = 1$ and $M = 398$ are given in Fig. 11. Here the gain is $\mathcal{G} = 200$ using definition (2.2). In Fig. 11, we compare the solution of system (4.16) with the projective integration of (4.16). We see that the errors between the projective integration of the system of ordinary differential equations (4.16) and the solution of (4.16) are small. Since the discretization (4.16) gives a reasonably accurate solution of (NJ), we can view also Fig. 11 as a plot of the exact solution of (NJ). Consequently, what we presented appears to be capable of significantly speeding up an explicit forward Euler method for small signal gradients (see also [17,26,29]).

The second numerical example in this section is based on the upwind discretization (4.13). We know from Section 4.4 that the discretization (4.13) provides a less accurate solution of (NJ) than (4.16) for parameters (4.22) due to the artificial diffusion of the
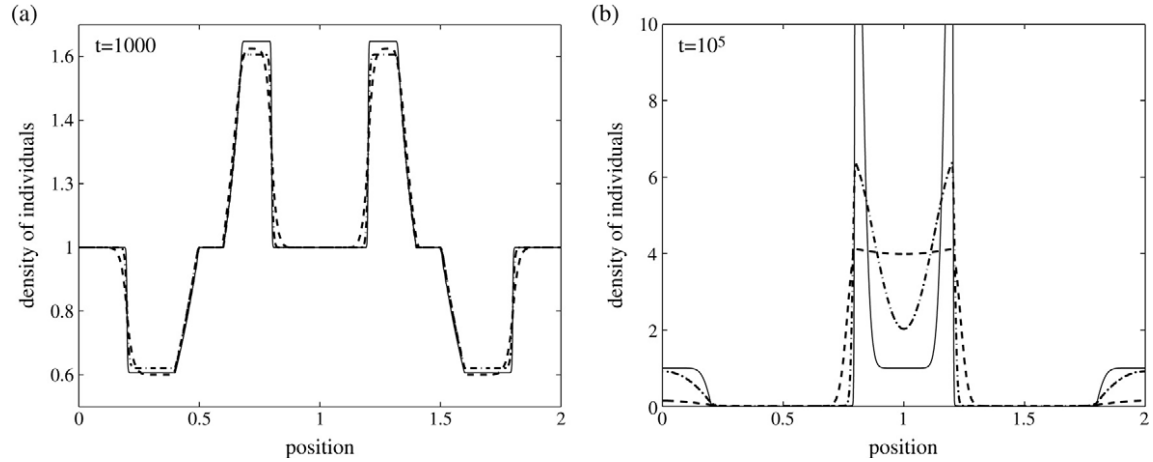
Fig. 12. (a) Density of individuals for $s = 0.0001$ and $\alpha = 1$ at time $t = 1000$. We plot the solution of (NJ) given by accurate numerical method (solid line), the solution obtained by projective algorithm $(P_k^M\text{-}1)$–$(P_k^M\text{-}3)$ for (4.13) with (4.22), $k = 1$ and $M = 198$, i.e., with $\mathcal{G} = 100$ (dashed line). We also plot the solution obtained by projective algorithm $(P_k^M\text{-}1)$–$(P_k^M\text{-}3)$ for (4.13) with (4.24), $k = 1$ and $M = 18$, i.e., with $\mathcal{G} = 10$ (dot–dashed line). In all computations, we use (4.19) and (4.23). (b) The same plots as in (a) for time $t = 10^5$.

upwinding scheme. On the other hand, the gain $\mathcal{G}$ of the projective integration method (4.13) is independent of the signal strength α. Consequently, we will present here results for $\alpha = 1$, i.e., when the signal is maximal possible. If we compare the results obtained by projective integration of (4.13) and the corresponding plots of solutions of (4.13), we again obtain small errors (results not shown) similar to those in Fig. 11. This would again support the numerical results from [16] concerning the accuracy of projective integration of ordinary differential equations. Instead, we compare the results of projective integration for two different choices of ε with an accurate solution of (NJ). We use either (4.22) or

$$s = 0.0001, \quad \delta x = 0.001, \quad \varepsilon = \frac{s}{\delta x} = 0.1. \tag{4.24}$$

Moreover, we use the initial condition (4.23) and δt given by (4.19); the results are shown in Fig. 12. We see that the long time behavior is highly influenced by the artificial diffusion of the scheme. Projective integration with small ε has large gain $\mathcal{G}$, but it will reach the steady state much faster than the exact solution of (NJ). Note that it is not an inaccuracy in projective integration per se; the inaccuracy is created by the inaccurate spatial discretization, based on upwinding with small ε.

Next, we will discuss how the ideas described so far in this paper can be used in Monte Carlo simulations of chemotaxis.

5. Coarse projective integration: A kinetic Monte Carlo example

In the previous sections we studied the gain $\mathcal{G}$ of projective integration for the system (NJ) of deterministic partial differential equations. Here we present results of Monte Carlo simulations of the underlying random walks using *coarse projective integration* [16,18,25] that will make use of the previous analysis. While in principle we simulate the evolution of the particle density profile over the entire spatial domain, we will demonstrate how to perform the computations required for coarse projective integration on a relatively small portion of the full domain. This is based on the presumed smoothness *in physical space* of the evolving density profile, which constitutes the underpinning of equation-free methods such as the *gap-tooth scheme* [19,36,25] as described below. Here we are able to speed up the kinetic Monte Carlo simulation about a thousand times.

Suppose that we have $2n_0$ random walkers in the interval $[0, 2]$, and suppose that we have only a kinetic Monte Carlo simulator to model the evolution of the system. As before, the interesting macroscopic quantity is the density of random walkers N which can be obtained as follows. We choose a macroscopic mesh size δx and we discretize the interval $[0, 2]$ using mesh (4.11). Then we obtain the (probability) density $N_{i+1/2}(t)$ at point $\frac{x_i + x_{i+1}}{2}$ as the number of particles in the interval $[x_i, x_{i+1}]$ divided by $n_0 \delta x$. We thus create a histogram of particles, which can of course be noisy.

If we have randomly walking *noninteracting* particles, the histograms obtained by using 10^9 or 10^6 random walkers appear roughly the same; the former is just less "noisy" than the latter. Consequently, we can obtain relatively accurate results quickly by simply decreasing the number of particles. However, in many interesting biological problems, cells change their environment, they consume nutrients, secrete waste, etc. Consequently, cells *interact* through environmental chemicals and then the number of cells is prescribed by the biological setup, and we cannot change it without changing the computed solution.

Therefore, in the examples of this section we will suppose that we do not know that the particles are noninteracting; we will suppose that there is a fixed number $2n_0$ of individuals in the domain of interest – the interval $[0, 2]$ – which are moving according to the rules of the random walker described in Section 3.2. We will show that in the case of a fixed number of particles the coarse
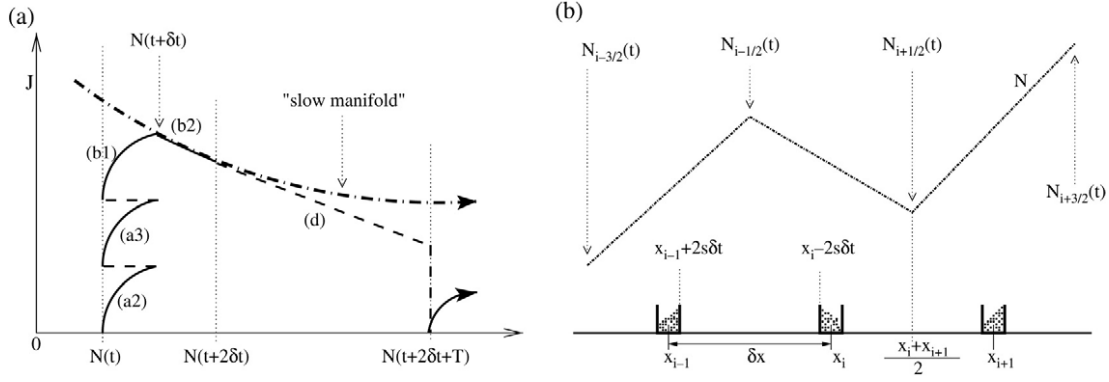
Fig. 13. (a) Schematic of steps (a1)–(d) of the coarse integration algorithm. Monte Carlo simulation is denoted by solid lines. Dashed lines denote relatively fast steps, i.e., resetting the values of density to $N(t)$ in steps (a2) and (a3) and the projective step (d). The dot–dashed line represents the evolution on the slow manifold of the system. We assume that the slow manifold can be accurately parametrized by density. (b) Schematic of three macroscopic mesh points x_{i-1}, x_i and x_{i+1} and kMC computational domains around them. Only particles close to the mesh points need be considered; the remaining particles will not leave/enter the interval $[0, x_i]$ during steps (a1)–(b2) and consequently, they do not have to be simulated. At the top, is the (piecewise linear) estimated density profile which is used in step (a1). We place particles in the small computational domains that their number is consistent with this density profile.

integration method leads to an even larger gain $\mathcal{G}$ than the projective integration method used earlier, where as before, the gain $\mathcal{G}$ is defined by (2.1). In the numerical example, we choose

$$2n_0 = 10^8, \quad s = 0.0001, \quad \delta x = 0.01, \tag{5.1}$$

i.e., we consider 201 equi-spaced mesh points (4.11) in the interval $[0, 2]$ on which we track the evolution of the macroscopic density, using the same parameters as in (4.22).

Monte Carlo simulations are performed as follows. Each particle is described by two variables — position $x \in [0, 2]$ and velocity $\pm s$. We use a small microscopic time step $dt = 0.01$, i.e., the unbiased turning frequency divided by 100, and during each time step the particle moves with speed s in the chosen direction. At the end of each time step, a random number chosen from a uniform distribution on $[0, 1]$ is generated and compared with the probability of the turn $\gamma dt = (1 \pm S'(x))dt$. If a turn occurs, the cell will move in the opposite direction during the next time step. To apply the previous results, we choose a macroscopic time step δt given by (4.19) and do kinetic Monte Carlo simulations in the interval $[t, t + \delta t]$, which means that we use the Monte Carlo simulator for $\delta t / dt$ microscopic time steps dt.

Since the histograms are noisy, we will work with the integral of the density — i.e., with the cumulative density function defined by

$$C(x, t) = \int_0^x N(x, t)dx. \tag{5.2}$$

Discretizing the interval $[0, 2]$ using mesh (4.11), we obtain

$$C_i(t) \equiv C(x_i, t) = \delta x \sum_{k=1}^{i} N_{k-1/2}(t), \quad \text{and} \quad C_i(t) - C_{i-1}(t) = N_{i-1/2}(t)\delta x. \tag{5.3}$$

In particular, the number of particles in the interval $[0, x_i]$, $i = 1, \ldots, n$ is given by

$$n_0 C_i(t) = n_0 \delta x \sum_{k=1}^{i} N_{k-1/2}(t), \quad i = 1, \ldots, n. \tag{5.4}$$

In order to use coarse integration, it is important to compute the change of $C_i(t)$ during the time interval $[t, t + \delta t]$; equivalently, we want to know the change of the number of particles in $[0, x_i]$ during the time step $[t, t + \delta t]$. Given that the speed of the particles is s, only particles which are in the small interval $[x_i - s\delta t, x_i + s\delta t]$ at time t can enter or leave the interval $[0, x_i]$. Consequently, only a small number of particles around each mesh point have to be simulated (compare Fig. 13(b)); of course we are implicitly assuming that the discretization mesh is fine enough so that interpolation between mesh points provides an accurate estimate of the evolving density profile. Using (4.19) and previous results, we choose

$$\delta t = 0.5, \quad T = 99, \tag{5.5}$$

and compute the cumulative density at time $t + 2\delta t + T = t + 100$ from the cumulative density function at time t by the following algorithm (compare with Figs. 13(a) and 2)

(a1) Given a macroscopic initial cumulative density $C(t)$ at mesh points (4.11), we compute the density $N_{i-1/2}(t)$ by the formula

$$N_{i-1/2}(t) = \frac{C_i(t) - C_{i-1}(t)}{\delta x}, \quad i = 1, \ldots, n.$$

We put $n_0 N_{i-1/2}(t)\delta x$ particles in each interval $[x_{i-1}, x_i]$ and distribute them so that the resulting probability density function is a continuous piecewise linear function with value $N_{i-1/2}(t)$ at point $\frac{x_{i-1}+x_i}{2}$, $i = 1, \ldots, n$. Thus (see Fig. 13(b))

$$N(x, t) = N_{i-1/2}(t) + \frac{N_{i+1/2}(t) - N_{i-1/2}(t)}{\delta x}\left(x - \frac{x_{i-1} + x_i}{2}\right) \quad \text{for } x \in \left[\frac{x_{i-1} + x_i}{2}, \frac{x_i + x_{i+1}}{2}\right].$$

Moreover, we assign alternating velocities to the particles, so that the initial flux is effectively zero. As we mentioned earlier, we do not have to simulate all particles in $[\frac{x_{i-1}+x_i}{2}, \frac{x_i+x_{i+1}}{2}]$; instead, we consider only particles which are inside a small interval $[x_i - 2s\delta t, x_i + 2s\delta t]$ around the macroscopic mesh point x_i (this could be thought of as analogous to the gap-tooth scheme [19], except that one does not have to formulate and impose effective smoothness boundary conditions, see Fig. 13(b)).

(a2) Evolve the system using the microscopic Monte Carlo simulator for time δt. Then return the particles to their initial position as given in (a1) but with a velocity equal to the values computed in (a2) (this and the following are *preparatory* steps to bring the microscopic initialization close to the slow manifold).[5]

(a3) Repeat (a2) again, i.e., evolve the system using the microscopic Monte Carlo simulator for time δt. Then return the position of particles to their initial values as given in (a1) keeping the velocities equal to computed velocities in (a3) (this can be repeated a few times).

(b1) Using the positions and velocities produced at the end of step (a3), evolve the system using the microscopic Monte Carlo simulator for time δt. Compute the number of particles in the interval $[0, x_i]$ at time $t + \delta t$ for $i = 1, 2, \ldots, n$.

(b2) Evolve the system using the microscopic Monte Carlo simulator for another time step δt. Compute the number of particles in the interval $[0, x_i]$ at time $t + 2\delta t$ for $i = 1, 2, \ldots, n$.

(c) Using data from (b1) and (b2), compute cumulative densities $C(t + \delta t)$ and $C(t + 2\delta t)$ at mesh points $x_0, x_1, \ldots, x_n$ (this is the *restriction* step in equation-free computation).

(d) Estimate the time derivative

$$\frac{\partial C_i}{\partial t} = \frac{C_i(t + 2\delta t) - C_i(t + \delta t)}{\delta t} \tag{5.6}$$

and take an extrapolation (projective) step

$$C_i(t + 2\delta t + T) = C_i(t + 2\delta t) + T\frac{\partial C_i}{\partial t}. \tag{5.7}$$

Then use $C_i(t + 2\delta t + T)$ as the new initial condition in step (a1).

The steps (a1)–(d) of the algorithm are illustrated in Fig. 13(a) where the slow manifold in density-flux space is shown as a dot–dashed line. Note that the steps (a1)–(a3) correspond to step (a) from Fig. 2. They are *preparatory* steps used to initialize the flux close to the slow manifold (since we assume that the flux equilibrates quickly); they qualitatively correspond to evolving the macroscopic PDE for a short time *constraining* the density profile to be the one we want to prescribe as our macroscopic initial condition. Such constrained evolution preparatory procedures (like "umbrella sampling") are standard in computational chemistry [42,35]. A more detailed description of such initialization algorithms in the case of legacy simulators can be found in [21,20].

The steps (b1)–(b2) correspond to step (b) from Fig. 2. Moreover, (b1) corresponds to step $(\Pr_k^M\text{-}1)$ and (b2) to step $(\Pr_k^M\text{-}2)$ from the projective integration algorithm of (NJ). Similarly, steps (c) and (d) can be also found in Fig. 2; moreover, steps (c) and (d) together form step $(\Pr_k^M\text{-}3)$ of the projective integration algorithm of (NJ).

If there is a small number of cells in one of the computational domains, then the straightforward application of the algorithm (a1)–(d) could give unrealistic results. For example, suppose that there are only two cells in the interval $[x_{i-1}, x_i]$ at time t, that the first cell moves to the interval $[0, x_{i-1}]$ during the time interval $[t, t + 2\delta t]$, the second cell moves outside the interval $[0, x_i]$ and that no other cell crosses mesh points x_{i-1} and x_i during time interval $[t, t + 2\delta t]$. Then the time derivative of the cumulative density function (5.6) would be negative at point x_i and positive at x_{i-1}. Moreover, the projected solution (5.7) satisfies $C_{i-1}(t + 2\delta t + T) > C_i(t + 2\delta t + T)$; consequently there is a negative number of particles in the interval $[x_{i-1}, x_i]$ at time $t + 2\delta t + T$. To avoid this problem we have to consider more realizations for each computational domain containing a small number of particles, and compute an average over this ensemble of realizations. Practically, if the number of particles n_i in the small computational domain around x_i is less than a given number m, we choose to repeat (a1)–(d) for m/n_i microscopic realizations in this computational subdomain.

Numerical results for $\delta t = 0.5$, $T = 99$, signal strength $\alpha = 0.1$ and $m = 10\,000$ are given in Fig. 14. There are two sources of gain for this method. First, we have the gain of the projective step. In one step (a1)–(d), we compute the evolution of the system

---

[5] This step annihilates the correlations between the present and the initial velocities of a particle, and at the macroscopic level the time required is essentially that in which a hyperbolic equation rather a parabolic equation is needed at the macroscopic level. This was already known to Einstein (cf. [34]).
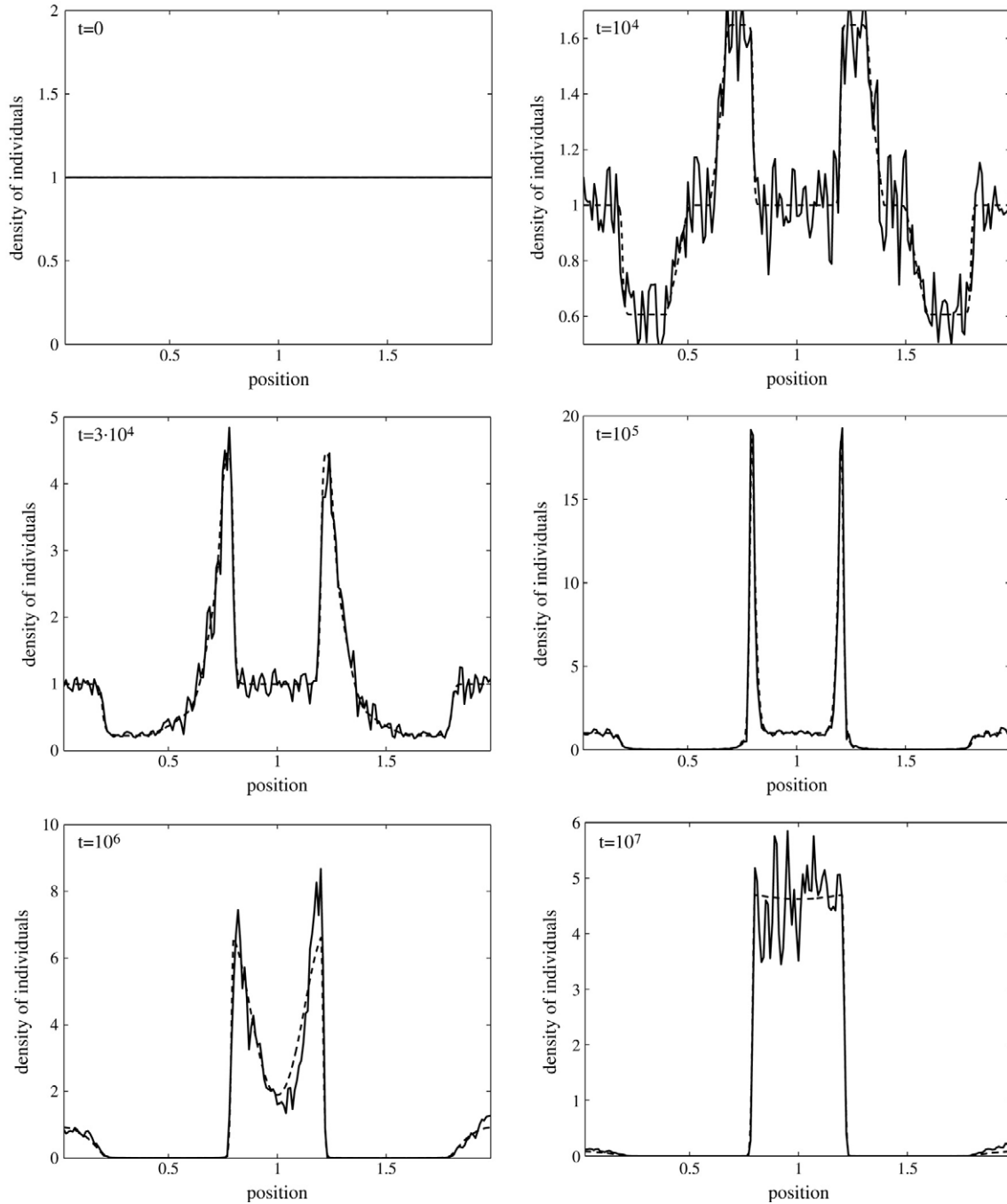
Fig. 14. The time evolution of the density of individuals for (5.1), (5.5), $\alpha = 0.1$ and initial conditions (4.23). We plot the density given by coarse integration (a1)–(d) and obtained by formula (5.8) from the computed cumulative density function $C(t)$ (solid line). Here we have gain $\mathcal{G} = 1250$. We also plot the solution of the corresponding macroscopic moment equations (dashed line).

over time $T + 2\delta t = 100$ and we run the Monte Carlo simulator for time $4\delta t = 2$ in steps (a2)–(b2). Consequently, the gain factor of the projective step is $(T + 2\delta t)/4\delta t = 50$. The second part of the gain comes from the fact that important particles (for the estimation of the evolution of a smooth macroscopic density) are only those particles which are leaving/entering the interval $[0, x_i]$ at the endpoint. From Fig. 13(b), we see that only particles which are at time t with distance less than $2s\delta t = 0.0001$ from the endpoint can leave/enter the interval $[0, x_i]$ during steps (b1)–(b2). Consequently, only the fraction $2n_0 \frac{4s\,\delta t}{\delta x} = \frac{2n_0}{50}$ of particles have to be simulated, and another factor of 50 appears in the gain.

Therefore, the combined gain of the coarse integration and reduced spatial simulation (based on macroscopic density smoothness) is $50 \times 50 = 2500$. However, 2500 is not the actual gain $\mathcal{G}$ because some computational time was lost by considering multiple microscopic realizations of domains which contained a small number of particles. In any case, we add less than $199m$ particles to the simulation where 199 is the number of "inner" computational domains and $m = 10\,000$ is the minimal number
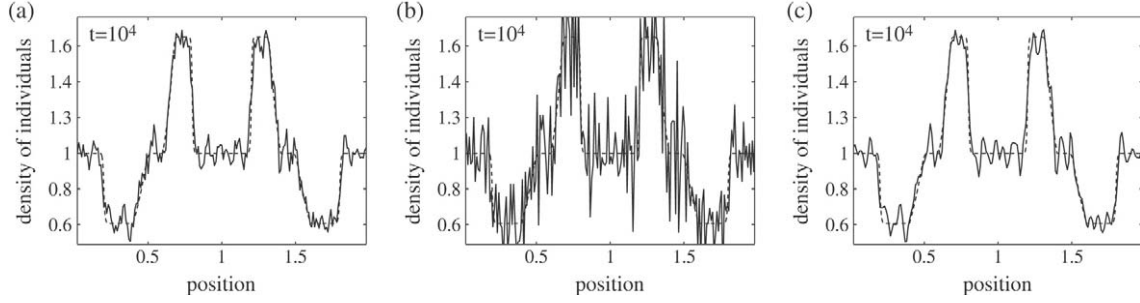
Fig. 15. Plots of the density of individuals at time $t = 10^4$ for the same cumulative distribution function as in Fig. 14, using different expressions for computing the discretized density. (a) Formula (5.9); (b) formula (5.10); (c) formula (5.11). We also plot the solution of the corresponding macroscopic moment equations (dashed line).

of particles in each of them. Consequently, we actually simulated more cells than $\frac{2n_0}{50} = 2 \times 10^6$ but, at any time, the number of simulated cells did not exceed $\frac{2n_0}{50} + 199M \sim 4 \times 10^6$. So, in the worst possible case, we slow down the computation by a factor of 2, which means that the total gain $\mathcal{G}$ of the method is at least $\mathcal{G} = 50 \times 50/2 = 1250$.

In Fig. 14, we present the time evolution of the solution given by method (a1)–(d) (solid line) compared to the solution of the macroscopic PDE equations (dashed line). Since the algorithm (a1)–(d) computes cumulative density functions $C(t)$ and we visualize the density $N(t)$ in Fig. 14, the results are noisy and the plots depend on the formula which is used to generate the density curves from the computed cumulative density data. To be precise, in Fig. 14, we show a plot of the function

$$N(x_i, t) = \frac{C_{i+1}(t) - C_{i-1}(t)}{2\delta x}. \tag{5.8}$$

Another representation of the results is given in Fig. 15, where we show results for time $t = 10^4$ using different formulas for the density function N, namely

(a) $\quad N(x_i, t) = \dfrac{C_{i+2}(t) - C_{i-2}(t)}{4\delta x},$ \hfill (5.9)

(b) $\quad N\left(\dfrac{x_i + x_{i+1}}{2}, t\right) = \dfrac{C_{i+1}(t) - C_i(t)}{\delta x},$ \hfill (5.10)

(c) $\quad N(x_i, t) = \dfrac{C_{i+2}(t) + C_{i+1}(t) - C_{i-1}(t) - C_{i-2}(t)}{6\delta x}.$ \hfill (5.11)

Comparing plots in Fig. 15 and the corresponding plot from Fig. 14, we see how the visualization of the results depends on the formula for estimating the discretized density function $N(t)$ from the discretized cumulative density function $C(t)$. In particular, if we use (5.11) instead of (5.8) in Fig. 14, then the results will look less noisy. Alternative discretizations of the particle density (namely, the use of orthogonal polynomials to represent the inverse cumulative distribution function, or ICDF) can be found in the literature [16,37]. Techniques for estimating smooth field profiles from noisy particle data have been proposed, among other places, in the computational materials science literature (e.g., the thermodynamic field estimator [28]).

One can also decrease the noise in the computation, and the resulting macroscopic field estimates, by considering multiple microscopic realizations, i.e., by increasing the value of m. However, if we increase m, then the gain $\mathcal{G}$ will decrease (obviously the "wall clock" time of the overall computation remains the same if one does these computations in parallel).

Certainly, there is a relationship between the initial number of particles $2n_0$, the minimal number of particles in each small computational domain m and the gain $\mathcal{G}$ of the method. If we have a large number of particles $2n_0$, then we can choose a large m without significantly decreasing the gain of the method. On the other hand, if we have a stochastic problem with a small number of particles $2n_0$, then it may not be appropriate to consider a closed PDE as a good model of a single system realization. In our example, m was chosen in such a way that the gain $\mathcal{G}$ of the method was decreased only by a factor of two, and thus the Monte Carlo simulation was accelerated by a factor of at least 1250. Increasing m would further decrease the gain $\mathcal{G}$ and reduce the magnitude of the fluctuations.

6. Discussion

In Section 5 we analyzed an example in which a simple coarse integration scheme was "wrapped around" a kinetic Monte Carlo simulation. The short (in time) bursts of kMC simulation were performed over only part of the full computational domain; this provides another important factor in decreasing the overall computational cost for such complex problems. The idea of reduced spatial as well as temporal simulation (the so-called "gap-tooth" scheme and its combination with projective integration in "patch

dynamics") is based on smoothness in the evolution of macroscopic observables and constitutes a hallmark of equation-free computation. Let us note that the computation of long term dynamics of our system took several days on a IBM SP 375MHz Power3 processor using algorithm (a1)–(d). Consequently, a computation using the kinetic Monte Carlo simulator would take several years and was not attempted. We estimated the accuracy of coarse projective computations by comparing to solutions of accurate macroscopic partial differential equations, which in this example happened to be known. When we do not have population level equations, we must use standard a posteriori error estimates to check accuracy and adaptively control the error of our results as discussed below.

As we saw in Fig. 9 for matrix $\mathcal{B}$, the length of the possible projective step T, as determined by stability considerations, decreases with increasing strength of the signal α. The same is true for algorithm (a1)–(d). If we increase α, then we have to decrease T in order to have a stable scheme. In order to achieve stability for larger T we could use a similar strategy to that used for the matrix $\mathcal{A}$: we could introduce artificial diffusion into the scheme which would make the scheme stable, independently of the strength of the signal α. It is not difficult to design a coarse integration scheme with artificial diffusion present; however, such a scheme would predict incorrect dynamics for the system.

A better solution to the problem of large signal gradients is to note that large signal gradients are typically localized only in small parts of the domain of interest. In fact, the problem with the coarse integration scheme begins when a large signal gradient is present and particles become highly localized in space. Then the mesh is not fine enough in certain small domains of interest (around peaks) but it is sufficiently fine in the remainder of the interval [0, 2]. Similarly, the projective step is good for most of the computational subdomains, but it would lead to instabilities because of strong signal gradients for a few of the computational domains. One could conceivably adapt the mesh, leading to a nonuniform mesh, finer in regimes with large signal gradients and coarser otherwise. Then we may need to make different projective jumps in different parts of the domain of interest; issues of this nature have been studied for nonuniform meshes in traditional continuum numerical analysis using adaptive mesh refinement (AMR) methods [4], and in hybrid situations AMAR methods [14]. Efficient implementations of such adaptive techniques may be the key to significant acceleration of our illustrative Monte Carlo scheme, since they would allow us to obtain relatively accurate results for even larger sets of signal functions and for problems where the signal is also altered by the cells.

Detailed methods have been developed for adapting the computation to the time and space scales of the problem in continuum numerical analysis. For example, a formulation of equation-free methods for conservation problems, called the heterogeneous multiscale method, has been proposed by E and Engquist [10]; it is based on estimation of fluxes and leads to a generalized Godunov scheme. Adaptive temporal stepsize selection in numerical integration and adaptive mesh refinement in spatial discretizations are indispensable components of modern software, and are typically based on a posteriori error estimates of the solution accuracy computed on line. These methods can be naturally incorporated in equation-free algorithms to control, for example, projective integration time steps to control accuracy. It should be noted that in addition to adaptive time-step selection (for coarse projective integration) and adaptive mesh selection (for gap-tooth algorithms), there is an additional type of adaptivity that arises in equation-free computation. This is the adaptive detection of the *level* of modeling, which may involve augmenting or decreasing the number of variables needed for closure. At a very qualitative level, adaptation of this "level of description" comes from the estimation of the gap between "fast" and "slow" system variables, which can be attempted using matrix-free eigensolvers. By initializing the microscopic distribution using *more* variables than the current level of modeling, one can try to estimate the characteristic relaxation times of the additional variables to functionals of the ones we need. This allows one to detect (while the level of description is still successful) whether variables that are treated as "fast" are becoming "slow", and should be included as independent variables in the modeling. A good illustration of this is the evolution of stresses in a microscopic simulator of fluid flow: for a Newtonian fluid stresses rapidly become proportional to velocity gradients, while in non-Newtonian fluids this is not true, and one must use more independent variables to model such flows. This could be considered analogous to closing bacterial chemotaxis equations with only a single field (density) which can be done for long time dynamics in some parameter regimes versus needing two independent variables (right- and left-fluxes) to successfully close the system in some other cases. In our case the flux quickly relaxes to a functional of the density, as can be seen directly from the simulations.

A summary of the steps of our computational approach is as follows.

- identify the appropriate level of closure
- apply the equation-free computational algorithm
- do a posteriori error estimation.

As we discussed above, we have to first identify the level of closure, i.e. identify the slow dynamics of the system which we want to model. Then we can do coarse projective integration by making use of the spectral gap between fast and slow modes of the system. In some cases it can be advantageous to combine coarse projective integration with gap-tooth methods, so as to exploit the smoothness in physical space by only performing the computations on relatively small subdomains. As a result one can do transient calculations much faster than by direct simulations. If one is interested in steady states and the transient dynamics are unimportant, then one can use other computational equation-free techniques (such as application of the Newton–GMRES method) to obtain steady state behavior faster or do even bifurcation analysis [18,38]. The final step is a posteriori error analysis as suggested

above. This is an important issue if one wants to apply our computational approach to problems for which macroscopic equations are unavailable.

It was shown earlier that the large gain of the coarse projective integration is governed by the large spectral gap between fast and slow eigenvalues of the system. The biological model system treated here had such a large spectral gap because the mean running distance of individuals is much smaller than the size of the domain of interest. This fact is also important for the direct reduction of the transport equation to the parabolic diffusion approximation [11,12].

The method developed here has the potential to speed up the computation of macroscopic quantities for a wide range of evolutionary problems, including a variety of biological dispersal problems having a similar separation of time scales. At present computational testing is needed to determine the applicability of the method to new problems, and further analysis is needed to rigorously determine its applicability a priori.

Acknowledgements

References

[1] F. Alcantara, M. Monk, Signal propagation during aggregation in the slime mold Dictyostelium discoideum, J. Gen. Microbiol. 85 (1974) 321–334.

[2] S. Barkai, N. Leibler, Robustness in simple biochemical networks, Nature 387 (1997) 913–917.

[3] H.C. Berg, Bacterial microprocessing, Cold Spring Harbor Symp. Quantit. Biol. 55 (1990) 539–545.

[4] M. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, J. Comput. Phys. 53 (1984) 484–512.

[5] E. Budrene, H. Berg, Complex patterns formed by motile cells of Esterichia coli, Nature 349 (1991) 630–633.

[6] C. Cercignani, The Boltzmann Equation and Its Applications, in: Applied Mathematical Sciences, vol. 67, Springer-Verlag, 1988.

[7] S. Chapman, T. Cowling, The mathematical theory of non-uniform gases: an account of the kinetic theory of viscosity, thermal conduction and diffusion in gases, Cambridge University Press, 1991.

[8] P. Cluzel, M. Surette, S. Leibler, An ultrasensitive bacterial motor revealed by monitoring signaling proteins in single cells, Science 287 (2000) 1652–1655.

[9] J.C. Dallon, H.G. Othmer, A discrete cell model with adaptive signalling for aggregation of Dictyostelium discoideum, Philos. Trans. R. Soc. Lond. B Biol. Sci. 352 (1351) (1997) 391–417.

[10] W. E, B. Engquist, The heterogeneous multiscale methods, Commun. Math. Sci. 1 (1) (2003) 87–132.

[11] R. Erban, H. Othmer, From individual to collective behaviour in bacterial chemotaxis, SIAM J. Appl. Math. 65 (2) (2004) 361–391.

[12] R. Erban, H. Othmer, From signal transduction to spatial pattern formation in *E. coli*: A paradigm for multi-scale modeling in biology, Multiscale Model. Simul. 3 (2) (2005) 362–394.

[13] D.J. Estep, A short course on duality, adjoint operators, Green's functions, and a posteriori error analysis, in: Lecture Notes, 2004.

[14] A. Garcia, J. Bell, W. Crutchfield, B. Alder, Adaptive mesh and algorithm refinement using direct simulation Monte Carlo, J. Comput. Phys. 154 (1999) 134–155.

[15] G.W. Gardiner, Handbook of Stochastic Processes for Physics, Chemistry and Natural Sciences, 2nd ed., Springer Verlag, 1985.

[16] C. Gear, Projective integration methods for distributions, NEC TR 2001-130 (2001) 1–9.

[17] C.W. Gear, I.G. Kevrekidis, Projective methods for stiff differential equations: problems with gaps in their eigenvalue spectrum, SIAM J. Sci. Comput. 24 (4) (2003) 1091–1106.

[18] C.W. Gear, I.G. Kevrekidis, C. Theodoropoulos, Coarse' integration/bifurcation analysis via microscopic simulators: micro-Galerkin methods, Comput. Chem. Eng. 26 (4) (2002) 941–963.

[19] C.W. Gear, J. Li, I.G. Kevrekidis, The gap-tooth method in particle simulations, Phys. Lett. A 316 (2003) 190–195.

[20] C.W. Gear, T.J. Kaper, I.G. Kevrekidis, A. Zagaris, Projecting on a slow manifold: Singularly perturbed systems and legacy codes, SIAM J. Appl. Dyn. Syst. 4 (3) (2005) 711–732.

[21] C.W. Gear, I.G. Kevrekidis, Constraint-defined manifolds: a legacy-code approach to low-dimensional computations, can be found as Physics/0312094 at arXiv.org, 2004.

[22] T. Hillen, A. Stevens, Hyperbolic models for chemotaxis in 1-D, Nonlinear Anal. Real World Appl. 1 (2000) 409–433.

[23] E. Keller, L. Segel, Model for chemotaxis, J. Theor. Biol. 30 (1971) 225–234.

[24] E. Keller, L. Segel, Traveling bands of chemotactic bacteria: A theoretical analysis, J. Theor. Biol. 30 (1971) 235–248.

[25] I.G. Kevrekidis, C.W. Gear, J.M. Hyman, P. Kevrekidis, O. Runborg, K. Theodoropoulos, Equation-free, coarse-grained multiscale computation: enabling microscopic simulators to perform system-level analysis, Commun. Math. Sci. 1 (4) (2003) 715–762.

[26] V.I. Lebedev, Explicit difference schemes with variable time steps for solving stiff systems of equations, in: Numerical Analysis and Its Applications, Proceedings of the Workshop on Numerical Analysis and Its Applications, WNAA 96, Rousse, Bulgaria, Springer, Berlin, 1997, pp. 274–283.

[27] J. Li, P.G. Kevrekidis, C.W. Gear, I.G. Kevrekidis, Deciding the nature of the coarse equation through microscopic simulations: The baby-bathwater scheme, Multiscale Model. Simul. 1 (3) (2003) 391–407.

[28] J. Li, D. Liao, S. Yip, Coupling continuum to molecular-dynamics simulation: Reflecting particle methods and field estimator, Phys. Rev. E 57 (6) (1998) 7259–7267.

[29] A.L. Medovikov, High-order explicit methods for parabolic equations, BIT 38 (1998) 372–390.

[30] M. Melchior, H.C. Oettinger, Variance reduced simulations of stochastic differential equations, J. Chem. Phys. 103 (1995) 9506–9509.

[31] B. Nadler, S. Lafon, R. Coifman, I. Kevrekidis, Diffusion maps, spectral clustering and the reaction coordinates of dynamical systems, Appl. Comp. Harm. Anal., 2005 (in press).

[32] H. Othmer, S. Dunbar, W Alt, Models of dispersal in biological systems, J. Math. Biol. 26 (1988) 263–298.

[33] H. Othmer, P. Schaap, Oscillatory cAMP signaling in the development of Dictyostelium discoideum, Comments Theor. Biol. 5 (1998) 175–282.

[34] H.G. Othmer, On the significance of finite propagation speeds in multicomponent reacting systems, J. Chem. Phys. 64 (1976) 460–470.

[35] J.P. Ryckaert, G. Ciccotti, H.J.C. Berendsen, Numerical integration of the cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes, J. Comput. Phys. 23 (1977) 327–341.

[36] G. Samaey, D. Roose, I.G. Kevrekidis, The gap-tooth scheme for homogenization problems, SIAM MMS 4 (1) (2005) 278–306.

[37] S. Setayeshgar, C. Gear, H. Othmer, I Kevrekidis, Application of coarse integration to bacterial chemotaxis, SIAM MMS 4 (1) (2005) 307–327.

[38] C.I. Siettos, M.D. Graham, I.G. Kevrekidis, Coarse Brownian dynamics for nematic liquid crystals: Bifurcation, projective integration, and control via stochastic simulation, J. Chem. Phys. 118 (22) (2003) 10149–10156.

[39] A.J. Smola, O.L. Mangasarian, B. Schoelkopf, Space kernel feature analysis, Data Mining Institute Technical Report 99-04, University of Wisconsin, Madison, 1999.

[40] P. Spiro, J Parkinson, H. Othmer, A model of excitation and adaptation in bacterial chemotaxis, Proc. Nat. Acad. Sci. USA 94 (1997) 7263–7268.

[41] J. Strikwerda, Finite Difference Schemes and Partial Differential Equations, Wadsworth, Inc., Pacific Grove, California, 1989.

[42] G.M. Torrie, J.P. Valleau, Monte Carlo free energy estimates using non-Boltzmann sampling: Application to the sub-critical Lennard-Jones fluid, Chem. Phys. Lett. 28 (1974) 578–581.

[43] E. Zauderer, Partial Differential Equations of Applied Mathematics, John Wiley & Sons, 1983.

[44] R. Erban, H.G. Othmer, Taxis equations for amoeboid cells, J. Math. Biol. (submitted for publication).