Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 200 (2004) 605–638

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A computational strategy for multiscale systems with applications to Lorenz 96 model

Ibrahim Fatkullin [a,*], Eric Vanden-Eijnden [b,*]

[a] *California Institute of Technology, 1200 E. California Blvd., MC 217-50, Pasadena, CA 91125, USA*
[b] *Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, USA*

Received 4 February 2004; received in revised form 26 April 2004; accepted 26 April 2004
Available online 17 June 2004

Abstract

Numerical schemes for systems with multiple spatio-temporal scales are investigated. The multiscale schemes use asymptotic results for this type of systems which guarantee the existence of an effective dynamics for some suitably defined modes varying slowly on the largest scales. The multiscale schemes are analyzed in general, then illustrated on a specific example of a moderately large deterministic system displaying chaotic behavior due to Lorenz. Issues like consistency, accuracy, and efficiency are discussed in detail. The role of possible hidden slow variables as well as additional effects arising on the diffusive time-scale are also investigated. As a byproduct we obtain a rather complete characterization of the effective dynamics in Lorenz model.
© 2004 Elsevier Inc. All rights reserved.

Keywords: Mode reduction; Averaging techniques; Effective dynamics; Heterogeneous Multiscale Method (HMM); Multiscale numerical methods; Lorenz 96 model

1. Introduction

Computational techniques for dynamical systems evolving on widely separated time-scales have received a lot of attention recently (for a review see [8]). Consider

$$\begin{cases} \dot{X}^\varepsilon = f(X^\varepsilon, Y^\varepsilon), & X^\varepsilon_{t=0} = x, \\ \dot{Y}^\varepsilon = \frac{1}{\varepsilon} g(X^\varepsilon, Y^\varepsilon), & Y^\varepsilon_{t=0} = y. \end{cases} \tag{1}$$

Here ε is a small parameter measuring the separation between time-scales, and we have assumed that the state space can be explicitly decomposed into slow variables, $X^\varepsilon \in \mathbb{R}^m$, and fast ones, $Y^\varepsilon \in \mathbb{R}^n$ (this assumption is lifted below). Systems like (1) arise from molecular dynamics, atmosphere science, materials

[*] Corresponding authors. Tel.: +1-212-998-3154; fax: +1-212-995-4121 (E. Vanden-Eijnden).
E-mail addresses: ibrahim@acm.caltech.edu (I. Fatkullin), eve2@cims.nyu.edu (E. Vanden-Eijnden).

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.04.013

science, etc. They are challenging for numerical computations because a time-step of the order of ε is necessary to resolve the fast variables Y^ε; therefore a total number of steps of the order of ε^{-1} is required to simulate the evolution of the slow variable X^ε.

In [30] a numerical procedure was designed to overcome the computational difficulties caused by the separation of time-scales. The procedure uses standard asymptotic results for systems like (1) (see e.g. [17,18,26,27,31]) which state that, in the limit as $\varepsilon \to 0$, the slow process X^ε converges to the solution of the equation

$$\dot{X} = F(X), \quad X_{t=0} = x. \tag{2}$$

Here

$$F(x) = \int_{\mathbb{R}^n} f(x,z)\mu_x(\mathrm{d}z), \tag{3}$$

where $\mu_x(\mathrm{d}z)$ is the invariant measure of the fast process considered at fixed $X^\varepsilon = x$:

$$\dot{Z}^\varepsilon(x) = \frac{1}{\varepsilon} g(x, Z^\varepsilon(x)). \tag{4}$$

Both $\mu_x(\mathrm{d}z)$ and $Z^\varepsilon(x)$ depend on x parametrically. Eq. (2) holds provided that the dynamics in (4) is ergodic and the expectation in (3) exists. Thus, the expectation in (3) can also be expressed in terms of a solution of (4) as

$$F(x) = \lim_{T\to\infty} \frac{1}{T} \int_0^T f(x, Z^\varepsilon(t,x))\mathrm{d}t. \tag{3$'$}$$

When ε is small, it is natural to try to use the simpler equation in (2) to approximate the slow process in (1). But this requires one to estimate the expectation in (3), which is nontrivial. Two possibilities come to mind. One is to make some simplifications on the dynamics of the fast variables under which the measure $\mu_x(\mathrm{d}z)$ can be estimated semi-analytically. This is the approach that was followed e.g. in [22–25]. Another possibility is to estimate $\mu_x(\mathrm{d}z)$ and the expectation in (3) via numerical simulation of (4). This idea is at the core of the method proposed in [30]. More specifically, this method proposes a class of multiscale numerical schemes with the following structure:

(1) A *macro-solver* for (2) gives the desired evolution of the slow variables $X \approx X^\varepsilon$ on the O(1)-time-scale. The choice of macro-solver is flexible, but as a rule it requires to estimate $F(X)$. Each time this is necessary, one uses:
(2) A *micro-solver* for (4), whose choice is also flexible and which gives evolution of the fast variables Z^ε on the O(ε)-time-scale; and:
(3) An *estimator* to evaluate the expectation in (3) for $F(X)$ from the data generated by the micro-solver.

Algorithms with this structure fit within the general framework of the heterogeneous multiscale method (HMM) proposed in [7] (see also [6]).

The multiscale schemes gain in efficiency over direct numerical methods for (1) because the dynamics of the fast variables has to be resolved only on a small subinterval of the total interval of time over which the dynamics of the slow variables is computed. Indeed the expectation in (3) can be estimated from the evolution of the fast variables on their own time-scale, and this calculation converges independently of the scale separation in the original dynamics. Thus the multiscale schemes have a cost independent of ε in the limit as $\varepsilon \to 0$, unlike direct numerical methods for (1) whose cost increases as ε^{-1} in this limit. Furthermore, unlike the traditional methods developed for stiff ODEs or differential algebraic equations (see e.g. [2,10–13,19]), the multiscale schemes also apply in the context of dynamical systems with stochastic effects. In [9] a thorough analysis of the convergence and accuracy properties of these schemes is presented for situations when the fast process is governed by a stochastic differential equation.

In the present paper we will investigate the usefulness of the multiscale schemes when applied to deterministic systems with chaotic behavior. Systems of this type arise commonly in applications and it is usually not known whether they meet the assumptions underlying the results in [9]. As a specific example of one such system we will study variants of a model proposed by Lorenz [20], hereafter referred to as L96. The following topics will be investigated, first in general, then using L96 as a test system.

1.1. Methodology and seamless multiscale schemes

Many systems with time-scale separation do not come in a nice form such as (1) where slow and fast variables are explicitly separated. Rather, one is often given a system of equations like

$$\dot{U}^\varepsilon = h(U^\varepsilon, \varepsilon), \quad U^\varepsilon_{t=0} = u, \tag{5}$$

where $u \in \mathbb{R}^p$ and the dependency on the small parameter ε is not as explicit as in (1) (for instance ε may simply be the reciprocal of the number of degrees of freedom). Therefore it is nice that *it is possible to devise a seamless multiscale scheme for* (5), *provided that some suitable slow variables can be defined which obey an effective dynamics as* $\varepsilon \to 0$. This requires that a mapping $\varphi(u)$ from $\mathbb{R}^p$ onto $\mathbb{R}^m$, with $m < p$ exists such that

$$D\varphi(u)h(u, \varepsilon) =: \varepsilon f(u, \epsilon) \tag{6}$$

for some function f which is O(1) in ε. This equation actually *defines* ε in the sense that the mapping φ should be such that

$$\varepsilon := ||D\varphi(u)h(u, \varepsilon)||/||h(u, \varepsilon)|| \ll 1 \tag{7}$$

for suitable norms, so that f in (6) is O(1). If the dynamics in (5) as $\varepsilon \to 0$ is ergodic on the family of hypersurfaces defined by $\varphi(u) = \text{cst}$, then the variable

$$X^\varepsilon(\tau) = \varphi(U^\varepsilon(\tau/\varepsilon)) \tag{8}$$

is a slow variable for (5) whose dynamics as $\varepsilon \to 0$ on the slow time-scale $\tau = \varepsilon t$ is governed by an equation like (2) with $F(x)$ given by

$$F(x) = \lim_{\varepsilon \to 0} \int_{\mathbb{R}^p} f(u, \varepsilon)\mu_x(\mathrm{d}u). \tag{9}$$

Here $\mu_x(\mathrm{d}u)$ is the invariant measure of (5) as $\varepsilon \to 0$ on the hypersurface $\varphi(u) = x$. In Section 2 we will show that the multiscale schemes can be readily generalized to equations like (5). Eq. (9) can be estimated in a rather seamless way at finite but small ε by constraining the dynamics in (5) on $\varphi(u) = x$ via projection. This seamless version of the multiscale scheme avoids the tedious step of having to derive explicitly equations for the fast variables Y^ε. This seamless version of the multiscale scheme will be used throughout this paper to study L96, first in parameter setting where the slow and fast variables are explicitly separated (Sections 3 and 4), then on a variant of L96 system with hidden slow variables where the use of a seamless scheme is difficult to avoid (Section 5).

1.2. Consistency, accuracy, and efficiency

How should one assess the accuracy and efficiency of the multiscale numerical schemes when applied to large deterministic systems like L96 which display chaotic behavior? Since the behavior of such systems is intrinsically stochastic, it is appropriate to use statistical criteria like the invariant measure of the slow modes or their autocorrelation functions as diagnosis for the numerical scheme. In terms of numerical analysis, such diagnoses correspond to using weak convergence criteria on infinite time intervals. Unfortunately, the properties of usual numerical schemes in this context are very poorly understood and, in particular, their rate of convergence is not known (for results in this area see [29]). However, we will show that *it is possible to understand and predict the efficiency gain of the multiscale schemes over direct numerical solvers via consistency analysis of these schemes*. This analysis will be given in Section 2 and the results confirmed in Sections 3–7 on the example of L96. In addition in the appendix we speculate on more quantitative error estimates for the multiscale schemes under the assumption that certain error estimates for the macro- and the micro-solvers are known.

The overall efficiency of the multiscale schemes depends on the efficiency of the estimator. The better the estimator, the less computations with the fast variables are necessary, and the more efficient the scheme is. Denoting by $\{Z^r(t,x)\}_{r=1}^R$ R independent realizations of the fast process, a possible (and standard) way to estimate expectations like (3) is to use

$$\tilde{F}(x) = \frac{1}{RN}\sum_{r=1}^{R}\sum_{n=N_1}^{N+N_1-1} f(x, Z^r(n\delta t, x)). \tag{10}$$

Here δt is the micro-time-step we use to simulate the fast process, R is the number of independent realizations of the fast process, N_1 is the number of steps we skip to eliminate transient, and N is the number of steps over which we perform time-averaging. The smaller these parameters, the more efficient the multiscale scheme is. It is quite remarkable that provided the fast variables are properly initialized at every macro-time-step, *the multiscale scheme converges even when* $R = 1$ (*one realization only*), $N_1 = 1$ *and* $N = 1$ (*one step of relaxation and no time-averaging*), *possibly with no loss in accuracy at fixed cost compared with a scheme with larger R, N, and* N_1. This property will be explained in Section 2 via consistency analysis and illustrated on L96 in Sections 3–5, but the key to understand it is actually quite simple. N_1 can be small and even equal to 1 because when the macro-time-step Δt is small, relaxation is short since the measures of the fast process Z conditional on the value of the slow variables X at two successive macro-time-steps are close ($O(\Delta t)$ apart). On the other hand, when $R = N = 1$, the estimate in (10) is inaccurate. But in the scheme this estimate is re-computed at every macro-time-step, meaning that it is re-computed for about $1/\Delta t$ times before the slow variables X have evolved significantly. As a result the effective number of realizations of the fast process used in the estimator (as integrated in the multiscale scheme) turns out to be proportional to $NR/\Delta t$, which is large when Δt is small even when $R = N = 1$. This shows that the quality of the estimator must be assessed as integrated in the multiscale scheme rather than taken as a tool to estimate the effective forcing at each macro-time-step.

Interestingly, the consistency analysis also indicates how to use the multiscale schemes to derive modified equations that are easier to solve than the original equations. This is especially interesting in the seamless setting where it is not obvious a priori how to modify (5).

1.3. Spatial scale separation

Besides the time-scale separation, there may also exist two separated spatial scales when the number of fast variables Y^ε is much larger than the number of slow variables X^ε. This situation is typical of multiscale systems. We will show in Sections 4 and 5 in the context of L96 that *the multiscale scheme can be used with only an* $O(1)$ *number of fast modes in situations when their original number is* $O(\varepsilon^{-1})$ (Section 4) *or* $O(\varepsilon^{-2})$ (Section 5). This permits one to increase dramatically the efficiency of the multiscale scheme.

1.4. Effective dynamics and effective forcing

During the computation, the multi-scale schemes evaluate $F(x)$ automatically. Therefore *the multiscale schemes can be used as a tool to determine the effective dynamical equation for the slow modes*. We will use this capability of the multiscale scheme in the context of L96 in Sections 3–5. This will give a rather complete characterization of the effective dynamics of the slow modes in L96, at least when the number of fast modes grows as the time-scale of their evolution becomes faster (Sections 4 and 5).

1.5. Effects on diffusive time-scale

In principle (2) holds as a limiting equation for (1) on finite time intervals only, and stochastic corrections must be included on the $O(\varepsilon^{-1})$-time-scale. For L96 in the parameter regimes investigated in Sections 3–5, it turns out that these corrections are small. The reason is that L96 displays deterministic chaos. Its behavior is already intrinsically stochastic on the O(1)-time-scale, and the small stochastic corrections arising on the $O(\varepsilon^{-1})$ time-scale have a very small effect on the long-time statistical properties of the system. This need not always be the case, though, and we can tune the parameters in L96 in such a way that the stochastic corrections on the $O(\varepsilon^{-1})$ time-scale become crucial. Such situations are explored in Section 6 where it is shown how to devise a poor man's version of the multiscale scheme which accounts for stochastic effects.

2. Methodology

Here we describe the multiscale scheme proposed in [30] and analyze some of its properties at a general level. The topics discussed in this section will be revisited in the context of L96 in Sections 3. We first discuss the multiscale scheme for systems like (1) where slow and fast variables are explicitly separated. Then we show how to devise a seamless version of the multiscale scheme which applies to equations like (5).

2.1. Basic algorithm

Let $\tilde{X}_m$ denote the numerical approximation of $X(t = m\Delta t)$, solution of (2) provided by the multiscale scheme. The simplest multiscale algorithm is the following:

Algorithm 1. (Forward Euler macro-solver and estimator by ensemble-average with R realizations.)

```
Take $\tilde{X}_0 = x$; $\{Z_{N_1,-1}\}_{r=1}^{R}$ given; $M = \lfloor T/\Delta t \rfloor$; $m = 0$;
while $m \leqslant M$
  $\tilde{F}(\tilde{X}_m) = 0$;
  for $r = 1, \dots, R$
    $Z^r_{0,m} = Z^r_{N_1,m-1}$;
    for $n = 0, 1, \dots, N_1 - 1$
    $Z^r_{n+1,m} = Z^r_{n,m} + \delta t\ \psi_g(\tilde{X}_m, Z^r_{N,m}, \delta t)$;
    end(for)
    $\tilde{F}(\tilde{X}_m) \leftarrow \tilde{F}(\tilde{X}_m) + \frac{1}{R} f(\tilde{X}_m, Z^r_{N_1,m})$;
  end(for)
  $\tilde{X}_{m+1} = \tilde{X}_m + \Delta t\ \tilde{F}(\tilde{X}_m)$;
  $m \leftarrow m + 1$;
end(while)
```

Here $[0, T]$ is the time-interval over which the evolution of the slow variables is sought, and Δt denotes the macro-time-step. $Z^r_{n,m}$ denotes the approximation of the rth independent realization of $Z(t = n\delta t, x = \tilde{X}_m)$ provided by the micro-solver whose micro-time-step is δt. Since any standard one-step method can be chosen for the micro-solver (in Sections 3–5 we will use a fourth-order Runge–Kutta), we have simply used the compact notation

$$Z^r_{n+1,m} = Z^r_{n,m} + \delta t\ \psi_g(\tilde{X}_m; Z^r_{n,m}, \delta t)$$

to denote the corresponding updating rule of the scheme. R is the number of realizations, N_1, the number of micro-time-steps we make to relax the fast process on the invariant measure $\mu_x(\mathrm{d}z)$, and the total number of micro-time-steps required per macro-time-step is therefore $R \times N_1$. Note that the algorithm requires an initial estimate for $\{Z^r_{N_1-1}\}^R_{r=1}$; in the applications, we make sure that this estimate is irrelevant by running the inner loop on r with a sufficiently large N_1 at the first macro-time-step. Subsequently, the value of N_1 can be significantly decreased as explained below.

Generalizations of Algorithm 1 to arbitrary one-step explicit schemes for the macro-solver, time-averaging for the estimator, etc. as well as a discussion of the convergence properties of these algorithms is given in the Appendix. Here we wish to discuss at a more qualitative level the properties of Algorithm 1 – these properties will then be tested in practice on L96 in Sections 3–5.

2.2. Consistency

Algorithm 1 is consistent with the limiting equation in (2) as

$$\Delta t \to 0,\ \ \delta t \to 0; \quad \text{and} \quad \varepsilon \Delta t / N_1 \delta t \to 0. \tag{11}$$

To understand why the third limit is required, note that if $\Delta t \to 0$, $\delta t \to 0$, but $\Delta t / N_1 \delta t \to \delta$, instead, then the scheme is consistent with (compare (1))

$$\begin{cases} \dot{X}^\varepsilon = \frac{1}{R} \sum_{r=1}^R f(X^\varepsilon, Z^{r,\varepsilon}, \varepsilon), & X^\varepsilon_{t=0} = x, \\ \dot{Z}^{r,\varepsilon} = \frac{1}{\varepsilon\delta} g(X^\varepsilon, Z^{r,\varepsilon}, \varepsilon), & Z^{r,\varepsilon}_{t=0} = Z^r_{N_1-1}, \quad r = 1, \dots, R. \end{cases} \tag{12}$$

This equation converges to (2) in the limit as $\varepsilon\delta \to 0$ if (1) converges to (2) in this limit, which explains why we need $\varepsilon \Delta t / N_1 \delta t = \varepsilon\delta \to 0$. In practice we therefore take Δt, δt, and $\varepsilon\delta$ sufficiently small to ensure stability and accuracy.

The consistency conditions in (11) may be somewhat surprising for two reasons: Unlike what we might have expected at first sight, neither $N_1 \delta t$ nor the number of realizations R need to be large for the scheme to converge. This is of course a good news for the multiscale scheme since the smaller N_1 and R, the lower the cost of the scheme is. Next we elucidate why this can be the case. Notice that in the explanations below it is crucial that we analyze the properties of the micro-solver/estimator (i.e. the inner loop over r in Algorithm 1) as integrated within the multiscale scheme rather than considered alone.

Consider N_1 first. Naively we might think that $N_1 \delta t$ needs to be large compared to ε because, at each macro-time-step, the R numerical approximations of the fast process $Z^\varepsilon(x)$ must relax to a sample of the invariant measure $\mu_x(\mathrm{d}z)$ unbiased by their initial conditions. Yet, on second thought, one realizes that Algorithm 1 is constructed in such a way that the initial value for $Z^r_{n,m}$ at each macro-time-step is the final value they reached at the previous macro-time-step. Therefore they already sample $\mu_{\tilde{X}_{m-1}}(\mathrm{d}z)$ initially when one let them evolve to sample $\mu_{\tilde{X}_m}(\mathrm{d}z)$. And since $\tilde{X}_m - \tilde{X}_{m-1} = \mathrm{O}(\Delta t)$, these two measures become closer and closer as $\Delta t \to 0$, and relaxation requires less and less micro-time-steps. This explains the constraint $N_1 \delta t / \Delta t \gg \varepsilon$ rather than $N_1 \delta t \gg \varepsilon$. We will see below that the same mechanism explains why ensemble-averaging is superior to time-averaging in terms of efficiency when one assesses the performance of the multiscale scheme not in approximating the slow process X (as we do now), but rather in evaluating $F(x)$ at each macro-time-step.

Consider R next. Clearly, (12) converges to (2) in the limit as $\varepsilon \to 0$ for arbitrary R, including $R = 1$. As explained in Section 1, this is because the effective number of realizations of the fast process turns out to be proportional to $R/\Delta t$ because the estimator is used $\mathrm{O}(\Delta t^{-1})$ times before the slow variables have changed significantly. To understand why this is the case from an alternative viewpoint, it is interesting compare Algorithm 1 with the following one corresponding to a macro-solver using forward Euler with time step $\Delta t/R$ and an estimator with 1 realization only. Assuming one updates the values of the slow variables every R time steps only (i.e. every Δt, just as in Algorithm 1), this can be written as:

Algorithm 2. (Forward Euler macro-solver with time step $\Delta t/R$ and estimator with 1 realization – no ensemble-average.)

```
Take $\tilde{X}_0 = x; Z_{N_1,-1}$ given; $M = \lfloor T/\Delta t \rfloor; m = 0$;
while $m \leqslant M$
  for $r = 1, \dots, R$
    $Z_{0,r} = Z_{N_1,r-1}$;
    for $n = 0, 1, \dots, N_1 - 1$
    $Z_{n+1,r} = Z_{n,r} + \delta t \ \psi_g(\tilde{X}_m, Z_{n,r}, \delta t)$
    end(for)
    $\tilde{F}(\tilde{X}_m) \leftarrow f(\tilde{X}_m, Z_{N_1,r})$;
    $\tilde{X}_m \leftarrow \tilde{X}_m + \frac{\Delta t}{R}\tilde{F}(\tilde{X}_m)$;
  end(for)
  $Z_{N_1,-1} \leftarrow Z_{N_1,R}$;
  $\tilde{X}_{m+1} \leftarrow \tilde{X}_m$;
  $m \leftarrow m + 1$;
end(while)
```

Algorithm 2 is strikingly similar to Algorithm 1. Their cost is identical and, except for the way the fast variables are initialized, the only difference is that the slow variable $\tilde{X}_m$ is updated outside the loop over r in Algorithm 1, whereas it is updated inside this loop in Algorithm 2. In particular, there is no reason to believe that Algorithm 1 is more accurate than Algorithm 2 and, in fact, the opposite may be true since Algorithm 2 uses a smaller (by a factor R) macro-time-step than Algorithm 1. This reduces the discretization error associated with the macro-solver independently of the error it makes on $\tilde{F}(x)$. Of course, such a conclusion does not account for other considerations like stability, etc. which may indicate that using more than one realization is preferable. It is also specific to a macro-solver using forward Euler; for higher order macro-solvers the optimal number of realizations may be larger than 1 – see the appendix.

2.3. Efficiency

As mentioned before, a detailed discussion of the efficiency of the multiscale scheme in the present context of chaotic systems is not possible since it would require the convergence properties of numerical schemes according to weak convergence criteria on infinite time intervals and these are not known. We speculate on this issue in the appendix by assuming that certain error estimates hold. Here we simply note that the multiscale scheme is more efficient than a direct solver for (1) because such a direct solver requires one to compute the evolution of the fast variables on the full time-interval $[0, T]$, whereas the multiscale scheme requires one to compute the dynamics of these only on a fraction $\Delta t/N_1\delta t = \delta^{-1}$ of $[0, T]$ – i.e. the multiscale scheme is δ times more efficient than a direct solver. This efficiency gain is significant when $\varepsilon \ll 1$, since it is then possible to take $\delta \gg 1$ and at the same time satisfy $\delta\varepsilon \ll 1$ required for accuracy.

It is instructive to take a closer look at this explanation. Suppose that the limiting process X is much closer – according to some suitable criterion – to X^ε than the error tolerance one accepts – if this is not the case, there is no way out of a direct computation with (1). This means that one can in principle increase the value of ε up to some optimal value $\varepsilon_{\rm opt}$ so that the discrepancy between $X^{\varepsilon_{\rm opt}}$ and X^ε is precisely within error tolerance. But if one recalls the discussion about consistency at the beginning of this section, one realizes that the multiscale scheme precisely is a way to implement such a procedure: take $R = 1$ and $\varepsilon\Delta t/N_1\delta t = \varepsilon_{\rm opt}$. In other word, the multiscale scheme may be viewed as a seamless way to compute with the optimal $\varepsilon_{\rm opt} \gg \varepsilon$ consistent with the prescribed error tolerance. Notice that this indicates that the approximation $\tilde{X}_n$ provided by the multiscale scheme may actually be closer to the original process X_ε^t than the limiting one, X solution of (2).

From this viewpoint, the multiscale scheme bears some similarity with penalty methods like the one used in computational fluid dynamics, originally introduced in [4] (see also [5]), or in Car–Parrinello ab-initio molecular dynamics [3]. In the first method, the hard constraint on incompressibility is relaxed to facilitate the computation using a slightly compressible velocity field within error tolerance of the incompressible one. In the ab-initio molecular dynamics, the mass of the electron is artificially increased to reduce the stiffness of the system.

In fact the multiscale scheme may be viewed as a tool to extend penalty methods to systems where their application is not obvious at first sight. Indeed, an advantage of the multiscale scheme is that it allows one to compute with $\varepsilon_{\rm opt}$ even in systems where it would be difficult or impossible to change explicitly the value of ε in the original equation (recall that we do not explicitly tune up ε in Algorithm 1 and $\varepsilon_{\rm opt}$ only emerges in the analysis of this algorithm). This will become fully apparent in the next section where we discuss a seamless version of the multiscale scheme.

2.4. Seamless scheme

Consider (5) appropriately rescaled

$$\dot{U}^\varepsilon = \frac{1}{\varepsilon} h(U^\varepsilon, \varepsilon), \quad \varepsilon^\varepsilon_{t=0} = u, \tag{13}$$

and suppose that the mapping $\varphi(u)$ defining the slow variables X as in (6) is known. A simple seamless multiscale algorithm to compute the dynamics of the slow variable is then the following:

Algorithm 3. (Seamless scheme with forward Euler macro-solver, unconstrained micro-solver, and estimator by ensemble-average with R realizations.)

```
Take $\tilde{X}_0 = x$; $\{U_{N_1,-1}\}_{r=1}^R$ given and consistent, i.e. $\varphi(U^r_{N_1-1}) = x$ for all $r$; $M = \lfloor T/\Delta t \rfloor$; $m = 0$;
***while*** $m \leqslant M$
$\tilde{F}(\tilde{X}_m) = 0$;
***for*** $r = 1, \dots, R$
  $U^r_{0,m} = U^r_{N_1,m-1}$;
  ***for*** $n = 0, 1, \dots, N_1 - 1$
  $U^r_{n+1,m} = U^r_{n,m} + \delta t\ \psi_h(U^r_{n,m}, \delta t)$;
  ***end(for)***
  $\tilde{F}(\tilde{X}_m) \leftarrow \tilde{F}(\tilde{X}_m) + \frac{1}{R} f(U^r_{N_1,m})$;
***end(for)***
$\tilde{X}_{m+1} = \tilde{X}_m + \Delta t\ \tilde{F}(\tilde{X}_m)$;
***for*** $r = 1, \dots, R$
  $U^r_{0,m+1} = U^r_{N_1,m} + D\varphi^{\rm T}(U^r_{N_1,m})\Lambda$
  [Here $\Lambda$ is determined so that $\varphi(U^r_{N_1,m} + D\varphi^{\rm T}(U^r_{N_1,m})\Lambda) = \tilde{X}_{m+1}$]
```

```
    end(for)
    $m \leftarrow m+1$;
  end(while)
```

Here $U^r_{n,m}$ is the approximation of the rth independent realization of $U(t = n\delta t, x = \tilde{X}_m)$ provided by the micro-solver for (9) whose updating rule is denoted as

$$U^r_{n+1,m} = U^r_{n,m} + \delta t\ \psi_h(U^r_{n,m}, \delta t).$$

The second loop for r enforces the constraint that $\varphi(U^r_{0,m+1}) = \tilde{X}_{m+1}$, i.e., the initial value in the micro-solver is consistent with the current value of the slow variables. $\Lambda \in \mathbb{R}^m$ (same dimensions as the slow variable X) needs to be determined in some way, e.g. by iteration of a variant of (18) below.

We show below that Algorithm 3 is consistent with

$$\begin{aligned}\dot{U}^r &= \frac{1}{\varepsilon\delta}\big(h(U^r,\varepsilon) - D\varphi^{\mathrm{T}}(U^r)A^{-1}(U^r)D\varphi(U^r)h(U^r,\varepsilon)\big) + \frac{1}{\varepsilon}D\varphi^{\mathrm{T}}(U^r)A^{-1}(U^r)\frac{1}{R}\sum_{r'=1}^{R}D\varphi(U^{r'})h(U^{r'},\varepsilon)\\ &= \frac{1}{\varepsilon\delta}P\ h(U^r,\varepsilon) - D\varphi^{\mathrm{T}}(U^r)A^{-1}(U^r)\frac{1}{R}\sum_{r'=1}^{R}f(U^{r'},\varepsilon)\end{aligned} \tag{14}$$

in the limit as $\Delta t \to 0, \delta t \to 0$, and $\Delta t/N_1\delta t \to \delta$. Here $A = D\varphi D\varphi^{\mathrm{T}}$ and P is the following operator mapping $\mathbb{R}^p$ onto $\mathbb{R}^p$:

$$P = I - D\varphi^{\mathrm{T}}A^{-1}D\varphi. \tag{15}$$

P projects vector fields in the full space to the tangent spaces of level sets of $\varphi(u)$. Notice that as enforced in Algorithm 3, $X = \varphi(U^r)$ for all r, and therefore this algorithm is consistent with the following equation for X:

$$\dot{X} = \frac{1}{R}\sum_{r=1}^{R}f(U^r,\varepsilon). \tag{16}$$

Thus (14) gives the same limiting equation for the slow variable X as (5) in the limit as $\varepsilon\Delta t/N_1\delta t = \varepsilon\delta \to 0$. As for Algorithm 1, R and N_1 do not need to be larger than 1 for the scheme to converge, and a gain in efficiency arises because one can take $\delta \gg 1$ and simultaneously have $\delta\varepsilon \ll 1$. This last point can also be seen upon rewriting (14) when $R = 1$ as

$$\dot{U} = \frac{1}{\varepsilon\delta}h^{\|}(U) + \frac{1}{\varepsilon}h^{\perp}(U), \tag{17}$$

where $h^{\|}(U) = P\ h(u)$ and $h^{\perp}(u) = \mathrm{O}(\varepsilon)$ denote the projections of h tangential and perpendicular to $\varphi(u) = x$, respectively. Thus, for $\delta > 1$ Algorithm 3 slows down the dynamics in the direction which does not affect the evolution of X, effectively introducing an effective $\varepsilon_{\mathrm{opt}} = \varepsilon\delta > \varepsilon$ as discussed before. This leads to the gain in efficiency. Notice that it also suggests that (17), which was derived from the multiscale scheme via consistency conditions, may be used as a starting point for other types of theoretical or numerical analyses. Indeed, given any numerical scheme, (17) can be integrated with a time-step which is δ times bigger than the time-step that one must use for the original Eq. (13). This very interesting observation allows one to generalize penalty methods to systems like (13); it will be exploited elsewhere.

It is also worth noting that it is not necessary to calculate the forcing $f(u)$ acting on the slow variables a priori and hard-code it into the program. The finite-difference approximation to the forcing can be obtained easily once the full system has been propagated by $N_1/\delta t$ micro-time-steps as

$$\tilde{F}(\tilde{X}_m) = \sum_{r=1}^{R} (\varphi(U^r_{N_1,m}) - \tilde{X}_m)/(RN_1\delta t).$$

This approximation is in the spirit of the equation-free techniques developed by Kevrekidis and collaborators [15]. It can be very useful if there is no easy access to the equation controlling the full dynamics.

Derivation of (14). This is a standard calculation and the only point worth mentioning is the calculation of Λ in the limit of $\Delta t, \delta t \to 0$. Since, by definition

$$\begin{aligned}\tilde{X}_{m+1} &= \varphi\Big(U^r_{N_1,m} + D\varphi^{\mathrm{T}}(U^r_{N_1,m})\Lambda\Big) = \varphi\left(U^r_{N_1,m-1} + \frac{N_1\delta t}{\varepsilon}h(U^r_{N_1,m-1}) + D\varphi^{\mathrm{T}}(U^r_{N_1,m-1})\Lambda\right) + \mathrm{o}(\delta t)\\ &= \tilde{X}_m + \frac{N_1\delta t}{\varepsilon}D\varphi(U^r_{N_1,m-1})h(U^r_{N_1,m-1}) + A(U^r_{N_1,m-1})\Lambda + \mathrm{o}(\delta t),\end{aligned}$$

it follows that

$$\begin{aligned}\Lambda &= A^{-1}(U^r_{N_1,m})(\tilde{X}_{m+1} - \tilde{X}_m) - \frac{N_1\delta t}{\varepsilon}A^{-1}(U^r_{N_1,m-1})D\varphi(U^r_{N_1,m-1})h(U^r_{N_1,m-1}) + \mathrm{o}(\delta t)\\ &= \Delta t A^{-1}(U^r_{N_1,m-1})\frac{1}{R}\sum_{r'=1}^{R} f(U^{r'}_{N_1,m-1}) - \frac{N_1\delta t}{\varepsilon}A^{-1}(U^r_{N_1,m-1})D\varphi(U^r_{N_1,m})h(U^r_{N_1,m-1}) + \mathrm{o}(\Delta t + \delta t).\end{aligned} \tag{18}$$

Using this result and matching properly the time-scales, it is straightforward to derive (14).

2.5. Remark on the existence of hidden slow variables

Recall that the slow variables X defined via (8) have a limiting dynamics provided that (5) at $\varepsilon = 0$ is ergodic on the hypersurface where $\varphi(u) = \mathrm{cst}$ (equivalently if slow and fast variables are explicitly known, (4) must be ergodic for every x). The question we wish to address here is the following.

Suppose that the slow variables X one identifies in (8) are incomplete, i.e. there exist hidden slow variables in the system, so that the ergodicity requirement is not met with these variables alone (notice this can happen with (1) if some combination of the Y's turn out to be also slow). What happens with the seamless multiscale scheme (Algorithm 3) if one uses this incomplete set of X's? (14) or (17) give the answer right away. Such a scheme would slow down artificially the dynamics of the slow variables that are not included, which of course can affect the evolution of all the slow variables (only the fast variables can be slowed down with no major effect on the slow ones when $\varepsilon \ll 1$ because of the existence of a limiting dynamics). Thus, it is crucial in general to include all the slow variables for the multiscale scheme to be accurate.

It is worth pointing out however that there is one exception where not including a slow variable does not affect the multiscale scheme. This is the case when this slow variable is piecewise constant in time and specifies the branch of an ergodic component for (5) at $\varepsilon = 0$ if more than one coexist and the number of branches depends on the values of the slow variables that are identified explicitly. Since such a hidden variable does not evolve except for jumps that are dictated by the current value of the slow variables explicitly accounted for in the scheme, it is easy to convince oneself that the multiscale scheme will be accurate in this case. The numerical experiments reported below suggest that L96 display such piecewise constant hidden slow variables.

Finally, we note that taking more slow variables than required (i.e. erroneously including fast variables in the set of the slow ones) does not work either. In this case indeed, the term $h^{\perp}(u)$ in (17) is $\mathrm{O}(\varepsilon^{-1})$ (instead of $\mathrm{O}(1)$ with the right choice of slow variable), and therefore to satisfy stability and accuracy criteria this equation must be solved with as small a time-step as the original Eq. (13).

2.6. Remark on ensemble- versus time-averaging

The discussion so far indicates that the multiscale scheme is fairly flexible. If evaluating $F(x)$ is one of the goals of the computation, then using many realizations may be necessary. If on the contrary our only interest is in the evolution of X, then one realization may be enough. Below we will show how to take advantage of this flexibility to study L96 via the multiscale scheme. Now we explain why ensemble-averaging is superior to time-averaging if one wants to obtain an accurate estimate of $F(x)$ at each macro-time-step. Consider the following multiscale algorithm using forward Euler as macro-solver and time-averaging:

Algorithm 4. (Forward Euler macro-solver and estimator by time-average over N micro-time-steps.)

```
Take $\tilde{X}_0 = x; Z_{N_1-1}$ given; $M = \lfloor T/\Delta t \rfloor; m = 0$;
while $m \leqslant M$
  $Z_{0,m} = Z_{N_1,m-1}; \tilde{F}(\tilde{X}_m) = 0$;
  for $n = 0, 1, \ldots, N_1 - 1$
    $Z_{n+1,m} = Z_{n,m} + \delta t\ \psi_g(\tilde{X}_m, Z_{n,m}, \delta t)$;
  end(for)
  for $n = N_1, \ldots, N_1 + N - 1$
    $Z_{n+1,m} = Z_{n,m} + \delta t\ \psi_g(\tilde{X}_m, Z_{n,m}, \delta t)$;
    $\tilde{F}(\tilde{X}_m) \leftarrow \tilde{F}(\tilde{X}_m) + \frac{1}{N} f(\tilde{X}_m; Z_{n,m})$;
  end(for)
  $\tilde{X}_{m+1} = \tilde{X}_m + \Delta t\ \tilde{F}(\tilde{X}_m)$;
  $m \leftarrow m + 1$;
end(while)
```

Here N_1 plays the same role as in Algorithm 1, and N is the number of micro-time-steps over which the time-average is actually performed; thus, Algorithm 4 requires $N_1 + N - 1$ micro-time-steps per macro-time-step. Now compare the costs of Algorithms 1 and 4 as estimators for $F(x)$.

It is reasonable to assume that the sampling error with Algorithm 1 decreases as $1/\sqrt{R}$. Therefore, if one assumes that the macro-time-step is small enough so that the relaxation errors can be neglected for the reason given before, then one needs $R = \mathrm{O}(\lambda^{-2})$ realizations to achieve an error tolerance λ. And the cost of Algorithm 1, which is $N_1 \times R$ micro-time-steps per macro-time-step (i.e. per evaluation of $F(x)$), scales as R.

In contrast the sampling error with Algorithm 4 decreases as $1/\sqrt{N\delta t}$ only (since $N\delta t$ is proportional to the effective number of realizations here), which means that one has to take $N\delta t = \mathrm{O}(\lambda^{-2})$ to achieve an error tolerance λ. And the cost of Algorithm 4, which is $N_1 + N$ micro-time-steps per macro-time-step scales as N. But since δt also has to be taken small to put the discretization errors from the micro-solver within tolerance, this means that the efficiency of Algorithm 4 deteriorates as δt decreases unlike that of Algorithm 1. Therefore Algorithm 1 beats Algorithm 4 in terms of efficiency in evaluating $F(x)$ accurately at each macro-time-step as asserted.

3. Application to Lorenz 96 (L96) system

L96 consists of K slow variables $\{X_k\}_{k=1}^{K}$ coupled to $J \times K$ fast variables $\{Y_j, k\}_{(j,k)=(1,1)}^{(J,K)}$ whose evolution is governed by

$$\begin{cases} \dot{X}_k = -X_{k-1}(X_{k-2} - X_{k+1}) - X_k + F_x + \frac{h_x}{J} \sum_{j=1}^{J} Y_{j,k}, \\ \dot{Y}_{j,k} = \frac{1}{\varepsilon}(-Y_{j+1,k}(Y_{j+2,k} - Y_{j-1,k}) - Y_{j,k} + h_y X_k). \end{cases} \tag{19}$$

The system is of the type of (1) (we dropped the subscript ε for simplicity of notations). Here both the X_k's and the $Y_{j,k}$'s are assumed to be periodic, i.e. $X_{k+K} = X_k$ and $Y_{j,k+K} = Y_{j,k}, Y_{j+J,k} = Y_{j,k+1}$. L96 was originally introduced to mimic mid-latitude weather and study the influence of multiple spatio-temporal scales on the predictability of atmospheric flows. The slow and fast variables X_k and $Y_{j,k}$ represent some atmospheric quantities discretized respectively into K and $K \times J$ sectors along the latitude circle. Each variable is driven by quadratic nonlinear interaction modeling advection, constant forcing, linear damping, and coupling between the slow variable in one sector and the J fast variables in the corresponding subsectors.

L96 was used as a tool to investigate the existence of an effective dynamics for the slow variables alone in [14,28]. In the following sections we will take this program one step further and use the multiscale scheme to compute the evolution of the slow variables and their statistics without having to fully resolve the evolution of the fast ones. This study will both demonstrate the usefulness of the multiscale scheme and as a by-product give a rather complete characterization of the effective dynamics of the slow modes in L96. For complementary analysis of the properties of (19) in various parameter settings, we refer to [1,21].

In this section, we will study (19) with $F_x = 10, h_x = -0.8, h_y = 1, K = 9, J = 8$, and $\varepsilon = 2^{-7} = 1/128$. These values of J and ε are somewhat different from the ones originally chosen by Lorenz, since $J \neq 1/\varepsilon$, but they will serve well our purpose here which is to demonstrate that the multiscale scheme is significantly more efficient than a direct scheme for (19) when the separation of time-scales is large. In Sections 4 and 5 we will consider situations with $J = O(1/\varepsilon)$ and $J = O(1/\varepsilon^2)$, respectively. The value of $\varepsilon = 1/128$ is also such that the dynamics of the slow modes is close to the limiting dynamics obtained as $\varepsilon \to 0$ – see below.

3.1. Properties of the system and existence of a limiting dynamics

In the parameter setting that we use, the solutions of (19) are chaotic. Typical time-series of a slow variable X_k and a fast variable $Y_{j,k}$ in the associated sub-sector are shown in Fig. 1; the subplot displays a

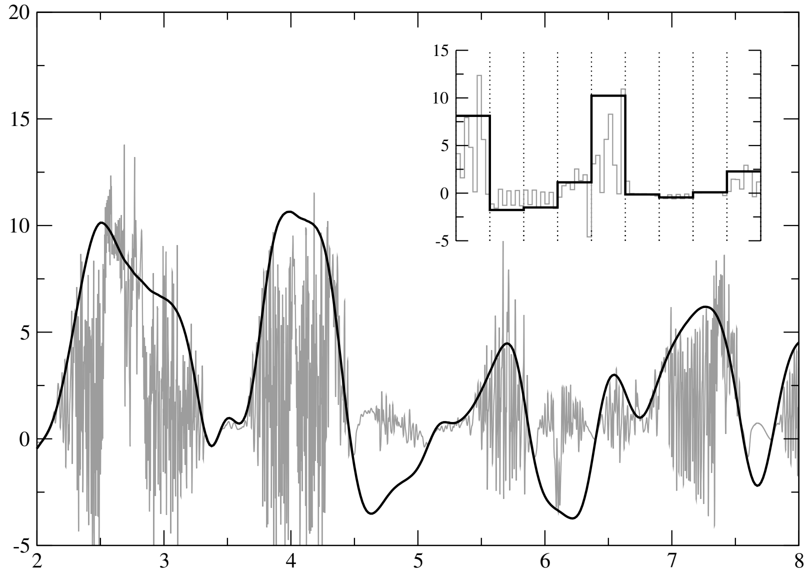

Fig. 1. Typical time-series of the slow (black line) and fast (grey line) modes; $K = 9, J = 8, \varepsilon = 1/128$. The subplot displays a typical snapshot of the slow and fast modes at a given time.

snapshot of the amplitude of the modes at a given time. The chaotic behavior can be inferred from the high sensitivity of the system to perturbations in the initial conditions, and further quantified by the statistical tests described next.

The numerical experiments show that the solutions of (19) settle on an attractor. One way to visualize (part of) this attractor is to look at the marginal probability density functions (PDFs) of the slow variable X_k (any k since the PDFs are all identical by symmetry) shown in Fig. 2. The mixing character of the dynamics can be inferred from the decay in time of the auto-correlation functions (ACFs) defined as (assuming ergodicity)

$$C_{k,k'}(t) = \lim_{T\to\infty} \frac{1}{T} \int_0^{\mathrm{T}} (X_k(t+s) - \bar{X})(X_{k'}(s) - X)\mathrm{d}s, \tag{20}$$

where

$$\bar{X} = \lim_{T\to\infty} \frac{1}{T} \int_0^{\mathrm{T}} X_k \,\mathrm{d}t, \tag{21}$$

and similarly for the fast variables – see Fig. 3. The ACFs of the slow modes X_k can be fit with great precision by

$$C_{k,k}(t) \approx C_0 \cos(\omega t)\mathrm{e}^{-\nu t}, \tag{22}$$

with appropriate $\nu, \omega,$ and $C_0 \approx C_{k,k}(0)$.

Even though there is a separation of time-scales between the slow evolution of the X_k's and the fast evolution of the $Y_{j,k}$'s since $\varepsilon = 1/128$, such separation is not obviously apparent from the correlation functions of these modes. In particular, Fig. 3 shows that, after a short transient decay, the correlation function of $Y_{j,k}$ decays and oscillates with about the same rate and frequency as the correlation function of

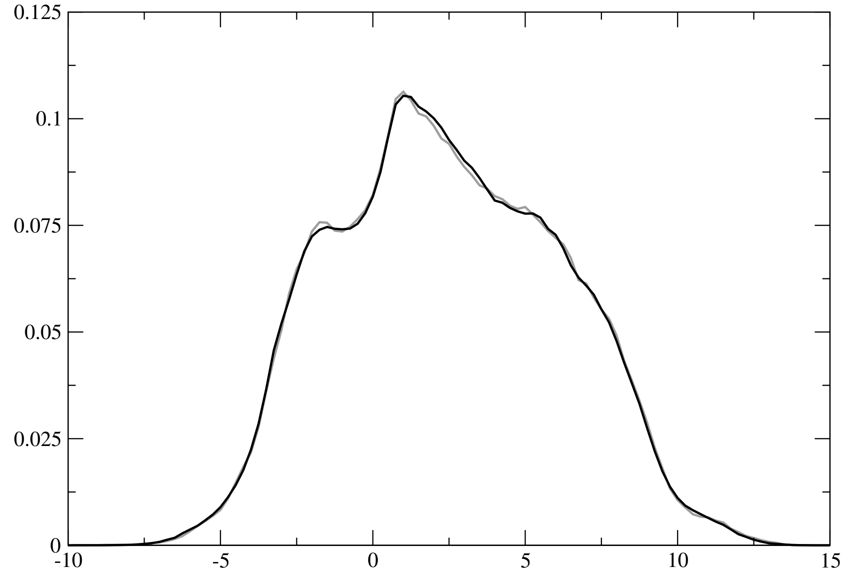

Fig. 2. PDF of the slow variable; $K = 9, J = 8$, black line: $\varepsilon = 1/128$, grey line: $\varepsilon = 1/1024$. The insensitivity in ε of the PDFs indicates that the slow variables have already converged close to their limiting behavior when $\varepsilon = 1/128$.

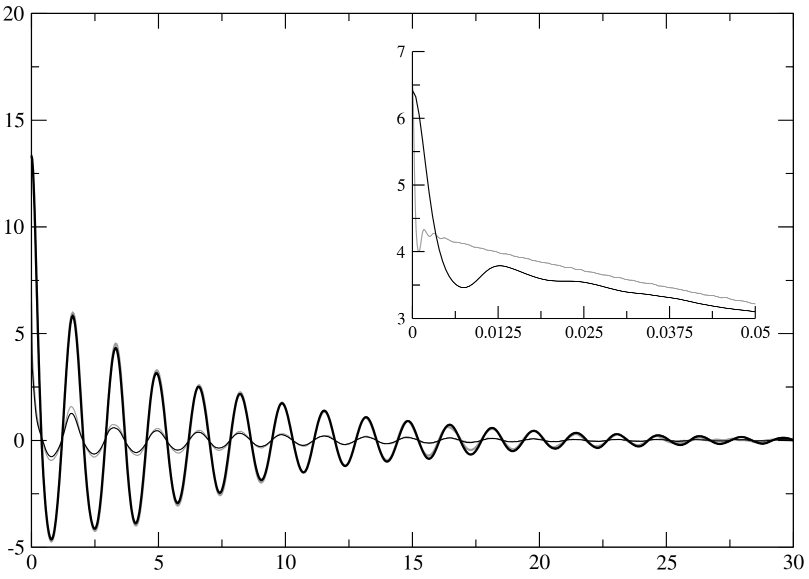

Fig. 3. ACFs of the slow (thick line) and fast (thin line) variables; $K = 9, J = 8$, black line: $\varepsilon = 1/128$, grey line: $\varepsilon = 1/1024$. The insensitivity in ε of the ACFs for the slow modes indicates that the slow variables have already converged close to their limiting behavior when $\varepsilon = 1/128$. The subplot is the zoom-in of the main graph which shows the transient decay of the ACFs of the fast modes becoming faster as ε is decreased: this is the only signature in the ACFs of the fact that the $Y_{j,k}$'s are faster.

X_k. In fact this short transient decay, which becomes shorter and shorter as ε is decreased, is the only signature on the ACFs that $Y_{j,k}$ is faster. This feature should be taken as a warning against simple procedure to identify fast modes based on computing their correlation time – here, the correlation time of the $Y_{j,k}$'s are comparable to the one of X_k and, in particular, independent of ε. In fact, the unambiguous test to determine if the $Y'_{j,k}$s are fast is to compute their ACFs at fixed $X = x$ (i.e. compute the ACFs of the variables $Z_{j,k}$'s solution of (23) below). These ACFs decay on a $O(\varepsilon)$-time-scale.

Next we check the existence of a limiting dynamics for the X_k's as $\varepsilon \to 0$. A necessary condition is that marginal PDFs and correlations functions have a limit as $\varepsilon \to 0$. This is consistent with the numerical experiments – see Figs. 2 and 3 and compare black and grey lines. This also indicates that the value we take, $\varepsilon = 1/128$, is small enough so that the statistical properties of the slow variables X_k are very close to their limit. Now, the existence of a limit for the law of the X_k's as $\varepsilon \to 0$ is necessary but not sufficient in order that these variables also have a limiting dynamics. For this we need to check the ergodicity of the fast modes at fixed $X_k = x_k$, – i.e. the solution of the following equation corresponding to the Eq. (4) which we use in the micro-solver of the multiscale scheme:

$$\dot{Z}_{j,k}(x) = \frac{1}{\varepsilon}\big(- Z_{j+1,k}(x)(Z_{j+2,k}(x) - Z_{j-1,k}(x)) - Z_{j,k}(x) + F_y + h_y x_k\big). \tag{23}$$

Fig. 4 shows the PDF of

$$\frac{h_x}{J}\sum_{j=1}^{J} Z_{j,k}(x) \tag{24}$$

for some typical values of x. This is the quantity whose average gives the effective forcing. The PDFs of (24) are robust against variations in initial conditions for $Z_{j,k}$ which confirm the ergodicity of (23). It is however

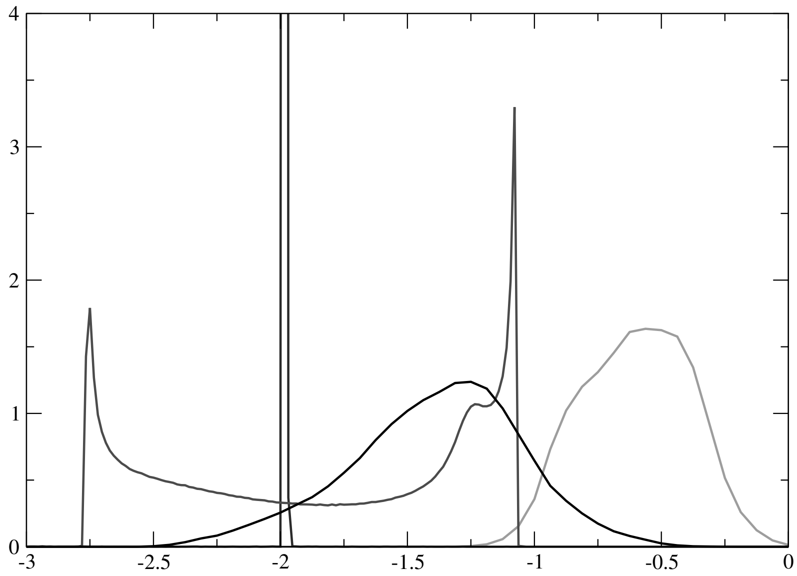

Fig. 4. Typical PDFs of the coupling term $(h_x/J)\sum_{j=1}^{J} Z_{j,k}(x)$ for various values of x. These PDFs are robust against variations in the initial conditions for (23) indicating that the dynamics of the fast modes conditional on the slow ones being fixed is ergodic. Notice however how different these PDFs look: this indicates that the feedback of the slow variables X_k on the fast ones $Y_{j,k}$ is significant in L96.

worth noting how different these PDFs look for different x, which indicates that the back reaction of the slow variables X_k on the fast ones $Y_{j,k}$ is significant in L96. This can also be seen in the time-series shown in Fig. 1: for some values of X_k, the fast variables are locked, whereas they vary widely for other values of X_k.

3.2. Direct-solvers versus the multiscale scheme

For the direct-solver we use the classical fourth-order Runge–Kutta method with time-step δt. We need to take $\delta t = 2^{-11}$ at most for stability, and at this value of δt we achieve reasonable accuracy (i.e. eyeball insensitivity of the results on the figures under further refinement of δt and changes in initial conditions). Thus, the direct simulation has a cost, taken as the number of time-steps of the fast variables per unit of time, given by

$$\text{cost(direct)} = \lfloor 1/\delta t \rfloor = 2^{11} = 2048. \tag{25}$$

To compute the PDFs and the correlation functions of the slow variables we use a total window of averaging of $T = 2^{18}$. The PDFs are computed from the time-series by bin-counting. The correlation functions are computed by direct summation:

$$C_{k,k'}(m\Delta t) = \frac{1}{M-m}\sum_{m'=1}^{M-m} X_k(m'\Delta t)X_{k'}((m'+m)\Delta t) - \bar{X}^2, \tag{26}$$

where

$$\bar{X} = \frac{1}{M}\sum_{m=1}^{M} X_k(m\Delta t), \tag{27}$$

and similarly for the fast variables. We do not utilize FFT since in our examples it does not provide significant improvements.

For the multiscale scheme we incorporate Algorithm 1 into a time-splitting scheme with a first step for the nonlinear, damping, and forcing terms in (19) using the fourth-order Runge–Kutta method, and a second step where Algorithm 1 is used to deal with the coupling term, $(h_x/J)\sum_{j=1}^{J} Y_{j,k}$. The step with the Runge–Kutta method allows us to take macro-time-steps up to $\Delta t = 2^{-4} = 1/16$ which otherwise would have to be taken much smaller to achieve stability. We test the multiscale scheme with the various values of Δt listed in Table 1. For the micro-solver for (23), we use a fourth-order Runge–Kutta with time-step $\delta t = 2^{-11}$ (i.e. the same as in the direct-solver). We check that the multiscale scheme converges and is accurate at the values $N_1 = 1$ (one micro-time-step per macro-time-step) and $R = 1$ (one realization), but we also test other values of N_1 and R as listed in Table 1. Even though this is in principle unnecessary for L96, we also use the seamless version by incorporating Algorithm 3 instead of Algorithm 1 in the time-splitting procedure described above. In all cases we estimate the cost of the multiscale scheme as the number of micro-time-steps of the fast variables per unit of time:

$$\text{cost(multiscale)} = R \times N_1 \times \lfloor 1/\Delta t \rfloor. \tag{28}$$

Even though the multiscale scheme is significantly more efficient than the direct-solver it reproduces extremely well both the PDFs and the correlations functions of the slow variables. Fig. 5 shows a run with $\Delta t = 2^{-7} = 1/128$, $N_1 = 1$, and $R = 1$, for which cost(multiscale) $= 2^7 = 128$, and hence the multiscale scheme is cost(direct)/cost(multiscale) $= 2^4 = 16$ times more efficient than the direct-solver. And this happens even though the time series for X_k that we generate with the multiscale scheme is much smaller than the one we generate in the direct simulations, since X_k is sampled every macro-time-step Δt in the former case, and every micro-time-step δt in the latter case. This simply means that even though the sample from the direct-solver is much bigger, it is not more significant statistically due to the large correlation between the slow variables at successive time-steps δt.

3.3. *Effective forcing*

The results of the last subsection clearly show that the forcing does not need to be computed accurately at each macro-time-step (which is the case since we can take $R = 1$) for the multiscale scheme to apply, as anticipated from the discussion in Section 2. Now we increase R to $R = 4096$ (a value at which the mul-

Table 1
Gain in efficiency of the multiscale scheme over a direct solver for various values of the control parameters in the multiscale algorithm

N_1	R	Δt	Gain	v	Error (%)	ω	Error (%)
$\varepsilon = 2^{-7}$		2^{-11}	–	0.135	–	3.81	–
Truncated		2^{-6}	–	0.287	113	3.57	6.3
1	4	2^{-4}	32	0.201	49	3.76	1.3
1	2	2^{-5}	32	0.171	27	3.89	2.1
1	1	2^{-6}	32	0.162	20	3.88	1.9
2	1	2^{-5}	32	0.162	20	3.88	1.9
4	1	2^{-4}	32	0.158	17	3.87	1.6
1	1	2^{-7}	16	0.137	2	3.84	0.8
2	1	2^{-7}	8	0.135	0	3.83	0.5

The error and the gain are calculated relative to the simulation with $\varepsilon = 2^{-7}$. The parameters v and ω are obtained by fitting as in (22) the ACF produced by the simulations. Notice that increasing the number of realizations R while keeping the gain fixed actually increases the error.

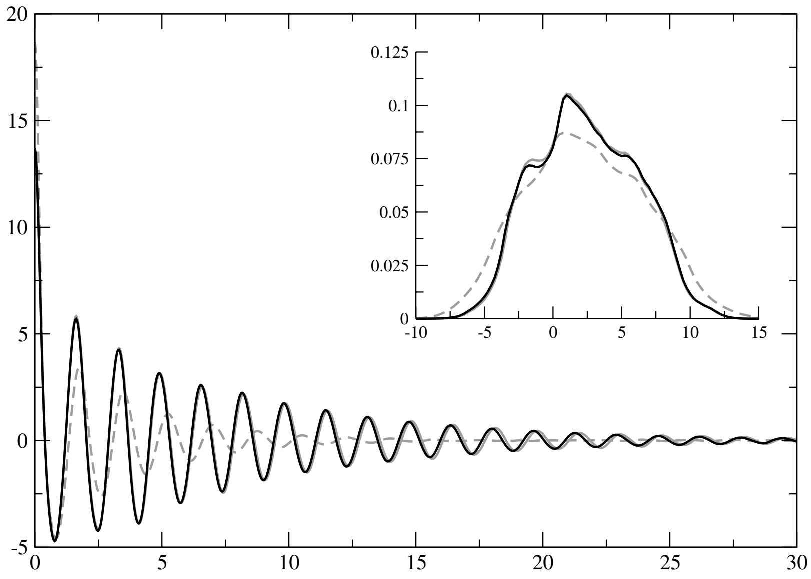

Fig. 5. Comparison between the ACF obtained via the multiscale scheme (black line) and via the direct-solver with $\varepsilon = 1/128$ (full grey line); $K = 9, J = 8$. The curves are so close that it is difficult to distinguish them. The subplot displays the PDF of the slow mode obtained via the multiscale scheme (black line) and via the direct-solver with $\varepsilon = 1/128$ (full grey line). Here $\Delta t = 2^{-7} = 1/128, N_1 = 1$, and $R = 1$. Thus cost(multiscale) $= 2^7 = 128$ and the multiscale scheme is cost(direct)/cost(multiscale) $= 2^4 = 16$ times more efficient than the direct-solver. Also shown in dashed grey are the corresponding ACF and PDF produced by the truncated dynamics where the coupling of the slow modes X_k with the fast ones, $Y_{j,k}$, is artificially switched off. The discrepancy indicates that the effect of the fast modes on the slow ones is significant in L96.

tiscale scheme is no longer more efficient than a direct scheme when $\varepsilon = 1/128$) to compute the effective forcing

$$F_k(x) = \frac{h_x}{J} \sum_{j=1}^{J} \int_{\mathbb{R}^J} z_{j,k} \mu_x(\mathrm{d}z), \tag{29}$$

where $\mu_x(\mathrm{d}z)$ is the invariant measure of (23).

A typical time series of the effective forcing $F_k(X)$ on mode X_k as it comes out of the inner loop on r in Algorithm 1 is shown in Fig. 6. Also shown is the time series of X_k itself. Since the effective forcing is the mean of the PDFs shown in Fig. 4 whose variances may be fairly large in comparison, one sees that the multiscale scheme does a good job at averaging this term to compute the effective forcing.

The effective forcing $F_k(x)$ is a vector-valued function of $K = 9$ variables $x = (x_1, \dots, x_9)$. In practice, it is not possible to run the simulation for long enough to build a sample of $F_k(x)$ which would allow one to fit this function as a whole. (Notice however that the multiscale scheme does not blindly sample $F_k(x)$ in x space but instead does it on the dynamical paths; therefore if the slow variables are dynamically constrained on a smaller subset in state space, like an attractor, the multiscale scheme automatically samples it and only it without wasting time evaluating $F_k(x)$ in regions that are not visited by the dynamics anyway). On the other hand, one may think of making additional assumptions about $F_k(x)$, the simplest of which being that it only depends on the slow variable x_k it corresponds to, i.e. $F_k(x) \approx F(x_k)$ – the next natural approximation would be to assume that $F_k(x) \approx F(x_{k-1}, x_k)$ (using the fact that the slow variables sustain wave propagating primarily from left to right), and so on. Testing $F_k(x) \approx F(x_k)$ is elementary since it amounts to verifying that the scatter-plot of $F_k(X)$ versus X_k defines a function. Such a scatter-plot is shown in Fig. 7, which

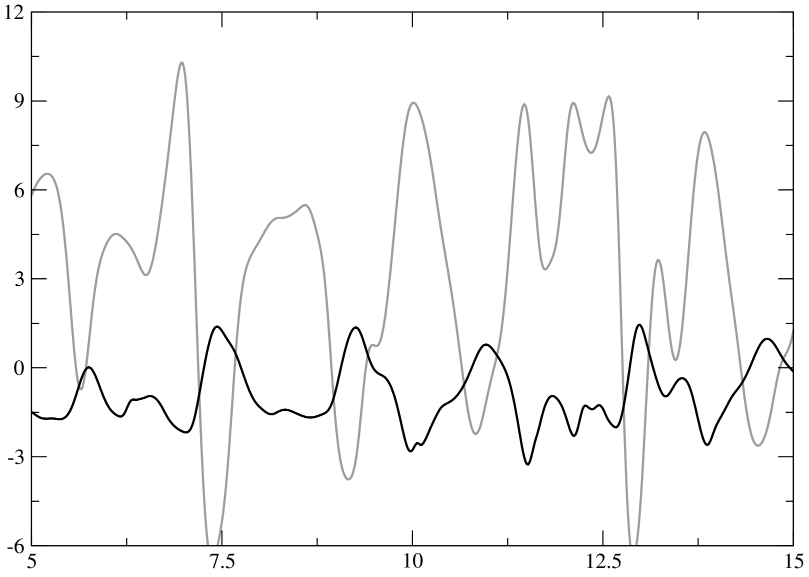

Fig. 6. Typical time-series of the slow variable (grey) and corresponding effective forcing (black) computed via ensemble averaging in the multiscale scheme with $R = 4096$ realizations; $K = 9, J = 8$. The relative smoothness of the black curve indicates that the multiscale scheme does a good job at computing the effective forcing since this is the mean of the random variable (24) whose variance is large in comparison – see Fig. 7.

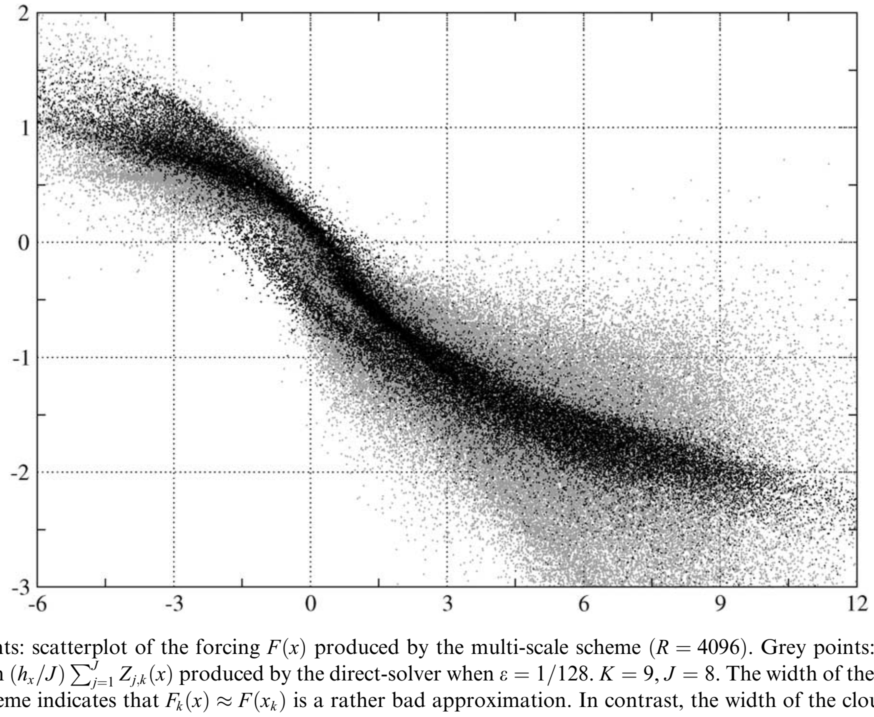

Fig. 7. Black points: scatterplot of the forcing $F(x)$ produced by the multi-scale scheme ($R = 4096$). Grey points: scatter plot of the bare coupling term $(h_x/J)\sum_{j=1}^{J} Z_{j,k}(x)$ produced by the direct-solver when $\varepsilon = 1/128$. $K = 9, J = 8$. The width of the cloud obtained via the multiscale scheme indicates that $F_k(x) \approx F(x_k)$ is a rather bad approximation. In contrast, the width of the cloud obtained via the direct-solver is more difficult to interpret since it is also due to statistical fluctuation.

clearly shows that $F_k(x) \approx F(x_k)$ is a bad approximation. Also shown is the scatter-plot of the bare forcing, which is even wider since $(h_x/J)\sum_{j=1}^{J} Y_{j,k}$ is (for all practical purposes at least) a random quantity – the width of the cloud now corresponds to statistical fluctuations in $(h_x/J)\sum_{j=1}^{J} Y_{j,k}$ which arise independently on whether its conditional average $F_k(x)$ depends or not on x_k only. This indicates that the multiscale scheme is useful in checking assumptions on the effective forcing which are more difficult to verify from direct numerical simulations due to statistical fluctuations.

3.4. Time- versus ensemble-averaging

Here we use time-averaging instead of ensemble-averaging to compute the effective forcing $F_k(x)$ and verify the assertion in Section 2 than the latter is more efficient than the former. Thus in the macro-solver we use Algorithm 4 instead of Algorithm 1. Using $N_1 = 1$ as before, the results show that it is generally (the precise value depends strongly on x) necessary to take $N = 2^{18} = 262144$ to obtain a time-series for $F_k(X)$ as accurate as the one we got with $R = 2^{12} = 4096$ using ensemble-averaging. Thus, time-averaging is about 64 times less efficient than ensemble-averaging in the present situation. The explanation for this phenomenon was given in Section 2 – in essence: with time-averaging, besides relaxing to the attractor, the $Z_{j,k}$'s have to revisit it at every macro-time-step, whereas they only need to relax to the attractor with ensemble-averaging. Yet it is useful to corroborate this explanation by looking at the convergence rate of $F_k(x)$ as a function of N, as shown in Fig. 8. When the micro-time-step δt is small, N becomes very large since the physical time of averaging is fixed and independent of δt. Also shown in Fig. 8, is the average of $(h_x/J)\sum_{j=1}^{J} Y_{j,k}$ – i.e. no constraint $X = x$ unlike what happens when one averages $(h_x/J)\sum_{j=1}^{J} Z_{j,k}$ in the multiscale scheme – which shows that constraining the dynamics as the multiscale scheme does it in the micro-solver is necessary with time-averaging (while it is not with ensemble averaging). Indeed, the

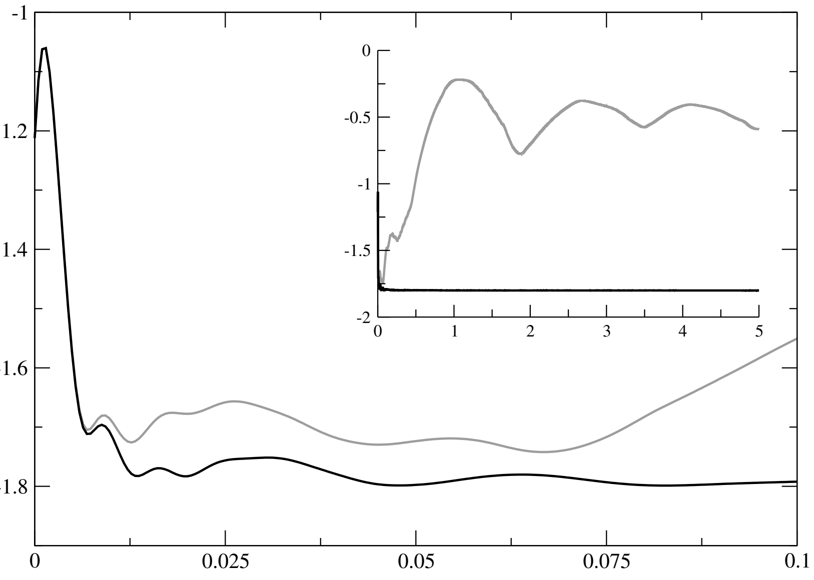

Fig. 8. Black lines: Typical accumulated time-average of the effective forcing produced by the estimator in the multiscale scheme. $K = 9$, $J = 8$, $\varepsilon = 1/128$. Grey line: Typical accumulated time-average of $(h_x/J)\sum_{j=1}^{J} Y_{j,k}$ produced by the direct solver at $\varepsilon = 1/128$. The subplot is the zoom-out of the main graph. Since $\delta t = 2^{-11}$, corresponds to $N = [t/\delta t] = 204$. Notice that due to the lack of constraint on the value of X_k in the direct solver, the average starts drifting (because the X_k's evolve) before it actually converges. Such problem does not arise in the multiscale scheme since the dynamics of the fast modes can be constrained fixing $X_k = x_k$ in micro-solver.

averaging time window is so large that, in the absence of constraint, the slow variables X_k significantly drift before the average converges which eventually would give the complete expectation of $(h_x/J)\sum_{j=1}^{J} Y_{j,k}$ rather than the conditional expectation of this quantity.

4. L96 with spatial scale separation

Here we study L96 (19) in the same parameter setting as in Section 3 except that we take $J = 1/\varepsilon = 128$. Thus there are now 128 fast modes per slow mode, and they evolve 128 times faster. This is closer to the original parameter setting used by Lorenz. Applying the multiscale scheme directly to this system as we did in Section 3 would also result in a gain in efficiency by a factor of up to 32 with no significant loss in accuracy. Here we show that we can significantly increase this gain by using the spatial scale separation in the system – i.e. the fact that the number of fast variables $Y_{j,k}$ is large. Specifically, we show how to apply the multiscale scheme by using in the micro-solver a smaller number of fast variables than in the original system – below 8 instead of 128 per slow mode. As a byproduct of this analysis, we obtain a complete characterization of the effective dynamics of L96 when $J = \mathrm{O}(1/\varepsilon)$ and we show that in this case the original system in (19) reduces to the following equation for the X_k's alone as $\varepsilon \to 0$:

$$\dot{X}_k = -X_{k-1}(X_{k-2} - X_{k+1}) - X_k + F_x + F(X_k). \tag{30}$$

Here $F(x_k)$ is an effective forcing term accounting for the effect of the $Y_{j,k}$'s on X_k which, in the present case, is a function depending only on one-variable, namely X_k. The specific shape of $F(x)$ is obtained below.

4.1. Existence and properties of a limiting dynamics

A typical snapshot and time series of the solution are shown in Fig. 9. To check that the dynamics of slow mode solution of (19) has a limit as $\varepsilon \to 0$ when $J = [1/\varepsilon]$ we proceed as in Section 3.1. First we verify

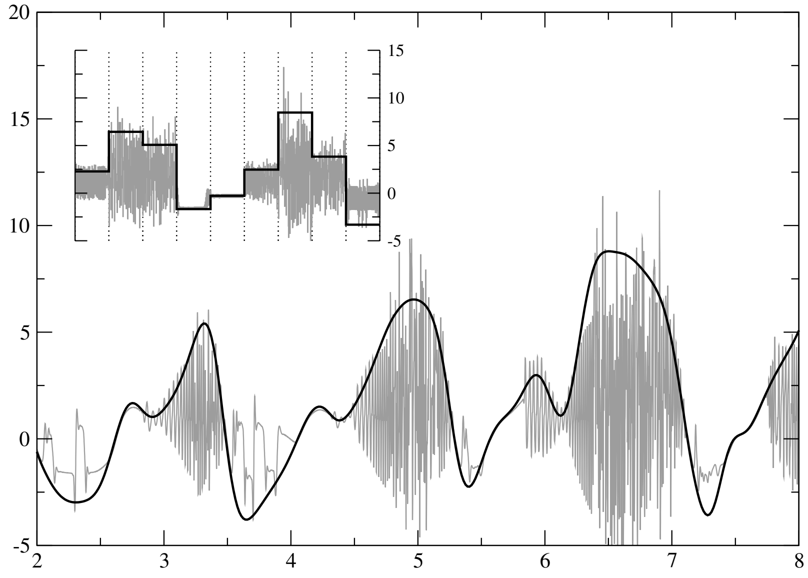

Fig. 9. Typical time-series of the slow (black line) and fast (grey line) modes; $K = 9$, $J = 1/\varepsilon = 128$. The subplot displays a typical snapshot of the slow and fast modes at a given time.

that PDF and ACF of the slow mode X_k at $J = 1/\varepsilon = 128$ and $J = 1/\varepsilon = 256$ are very similar which indicate that they have converged to their limit.

Next, we check the ergodicity of (23). Let us take a working assumption that the spatial interaction between the $Y_{j,k}$ (and hence between the $Z_{jk}(x)$ in (23)) are sufficiently weak and short-range on the average. Then, at any given time, the term $(h_x/J)\sum_{j=1}^{J} Y_{j,k}$ (and hence $(h_x/J)\sum_{j=1}^{J} Z_{j,k}(x)$) self-averages in the limit as $J \to \infty$ (i.e. $\varepsilon \to 0$ since $J = [1/\varepsilon]$) to a limit which, by the law of large numbers, satisfies

$$\lim_{\varepsilon\to 0} \varepsilon h_x \sum_{j=1}^{[1/\varepsilon]} Y_{j,k} = h_x \mathbf{E}_{X_k} Y_{j,k} \equiv F(X_k). \tag{31}$$

Here $\mathbf{E}_{X_k} Y_{j,k}$ is the conditional average of any given $Y_{j,k}$ at fixed X_k, and under the assumption of weak spatial interaction between the $Y_{j,k}$'s, it can only depend on the single X_k entering the equation for $Y_{j,k}$. Thus our working assumption implies that $F_k(x)$ depends only on this X_k, i.e. $F_k(x) = F(x_k)$. This assumption is of course non-trivial since the $Y_{j,k}$'s are nonlinearly coupled. Yet it can be verified from the scatter-plot of $F_k(X)$ versus X_k shown in Fig. 10 that this assumption holds with reasonable accuracy since this scatter-plot is rather sharp (much sharper than the one shown in Fig. 7 when $\varepsilon = 1/128$ but $J = 8$ only). Taking $J = 1/\varepsilon = 4096$ makes it even sharper which supports that this scatter-plot converges to the graph of a function as $J = [1/\varepsilon] \to \infty$. As a further test of (31) we also changed the value of h_x from the current value, $h_x = -0.8$, to $h_x = 1.2$ and we checked that this amounts to a simple rescaling of the scatter-plot by a factor 1.2/(0.8) – see the subplot of Fig. 10.

In Fig. 10, we can see that in the interval $X_k \in [-0.5, 0.9]$, F is a linear function of X_k, whereas for $X_k > 0.9$ and $X_k < -0.5$, F is a two-branch function of X_k: one branch is the continuation of the linear piece in the center interval and each of the other branches is some nonlinear function of X_k. Further analysis of

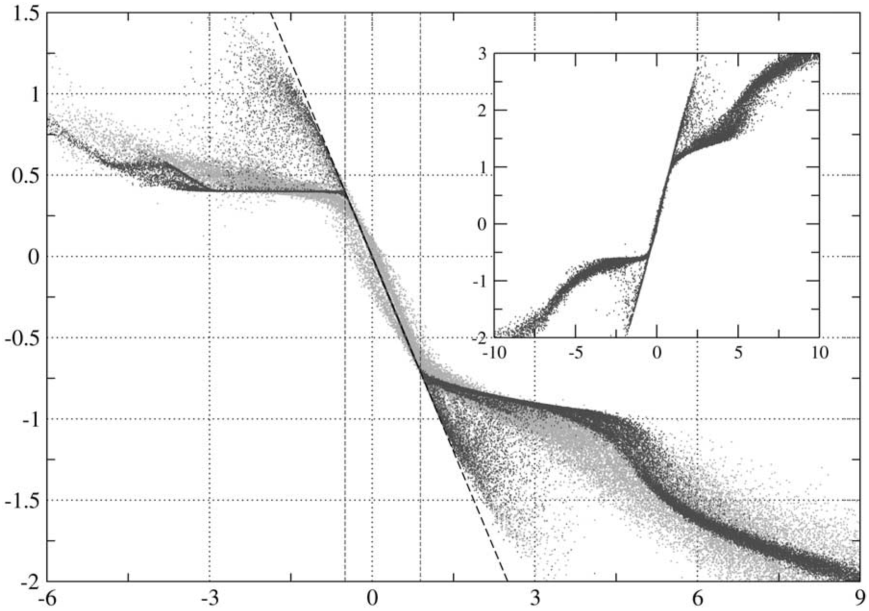

Fig. 10. Scatterplots of the bare forcing $\varepsilon h_x \sum_{j=1}^{[1/\varepsilon]} Y_{j,k}$ (no averaging here); light grey: $J = 1/\varepsilon = 128$; dark grey: $J = 1/\varepsilon = 4096$ ($K = 9$). The sharpness of these graphs confirm that bare forcing self-averages consistent with (31). The subplot: $J = 1/\varepsilon = 1024$, $h_x = 1.2$ (instead of $h_x = -0.8$ taken otherwise); it can be obtained by rescaling the forcing in the main plot respectively by 1.2/(−0.8). The thin dashed lines shows the stability band $X \in (-0.5, 8/9)$ and the corresponding forcing $F(X) = h_y h_x X$.

the dynamics indicates that when X_k leaves the center interval, F first follows the continuation of the linear branch, then jumps on the other branch. This bifurcation is in fact driven by a linear instability mechanism, as we explain next.

The simplest way to make sure that $F_k = F(X_k)$ is to change the boundary condition for the $Y_{j,k}$ in (19) and, correspondingly, the $Z_{j,k}(x)$ in (23). Instead of taking them periodic over the whole system, i.e. $Y_{j,k+K} = Y_{j,k}$, $Y_{j+J,k} = Y_{j,k+1}$, take them periodic in each subsector, i.e. $Y_{j+J,k} = Y_{j,k+1}$. Under the working assumption above, this should not change anything in the limiting dynamics for the slow variables X_k. With the new boundary conditions, it is elementary to check that the equation for $Z_{j,k}(x)$ in (23) has the following steady-state solution:

$$Z_{j,k} = h_y X_k. \tag{32}$$

This means that the measure of the $Z_{j,k}(x)$'s and therefore the conditional measure of the $Y_{j,k}$'s at fixed X_k is atomic on this value. From (31) it follows that

$$F(X_k) = h_x h_y X_k. \tag{33}$$

This function fits very well the center branch observed in the scatter-plot shown in Fig. 10. Now, let us analyze the stability of the steady-state solution (32). A standard eigenvalue analysis of the equation for $Z_{j,k}$ in (23) linearized around (32) indicates that the eigenvalues are

$$\lambda_j = -2\mathrm{i}h_y X_k \exp(\mathrm{i}\pi j/J)\sin(3\pi j/J) - 1 \quad j = 1,\dots,J. \tag{34}$$

It follows that (32) is stable if $\mathrm{Re}(\lambda_j) < 0$ or all j and unstable otherwise. As $J \to \infty$, the stable interval reduces to $X_k = (-1/2, 8/9)$. This is again in excellent agreement with the result in Fig. 10.

When (32) is unstable, we did not find any analytical argument which gives the stable branches of $F(X_k)$ seen in Fig. 10. Next, we use the multiscale scheme to compute these stable branches.

4.2. Multiscale scheme with spatial-scale separation

The discussion in the last subsection indicates that, rather than using the multiscale scheme for (19) with the original boundary conditions for $Y_{j,k}$, we should use this equation with the new boundary conditions, $Y_{j+J,k} = Y_{j,k+1}$. But in this case, the number of fast modes in each subsector can be modified as well, and in particular, can be taken much smaller than the actual value $J = 128$. Thus, in the multiscale scheme we took $J' = 8$ and checked insensitivity of the results by increasing this value to $J' = 16$ (as a theoretical justification for the value $J' = 8$, note that if $J' = 8$, the stable interval for the linear branch of $F(X_k)$ is (−1/2,1) which is already a fair approximation of the limiting interval (−1/2, 8/9) as $J \to \infty$).

With the new boundary conditions, there are two obvious ways to implement the multiscale scheme, with on on-the-fly evaluation of the effective forcing $F(x)$, or with a tabulation of this function.

4.2.1. Multiscale scheme with on-the-fly evaluation of $F(x_k)$

This proceeds exactly as in Section 3. The effective forcing is computed via the micro-solver and estimator at each time-step. We used $\Delta t = 2^{-7}$, $\delta t = 2^{-11}$ (same value as in the direct solver for (19) with the original boundary conditions), $N_1 = 1$, and $R = 1$. This gives (including the number of fast modes in the cost since it is different from the one used in the direct simulations)

$$\mathrm{cost(multiscale)} = R \times N_1 \times J' \times [1/\Delta t] = 2^{10} = 1024. \tag{35}$$

On the other hand, a direct solver for (19) has a cost

$$\mathrm{cost(direct)} = J \times [1/\delta t] = 2^{18}, \tag{36}$$

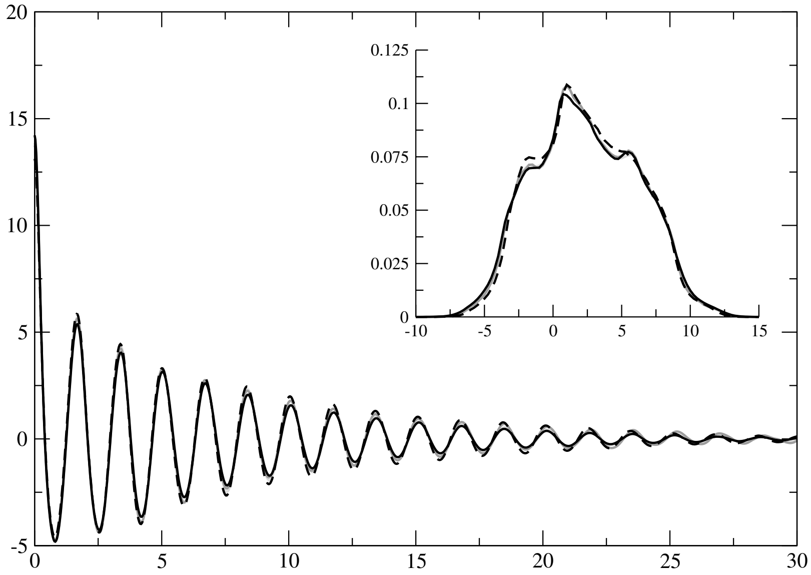

Fig. 11. Comparison of ACFs and PDFs (subplot) of the slow mode for various simulations in $J = 1/\varepsilon$ regime. Light grey: $J = 128$; dark grey: $J = 256$ (practically indistinguishable from the previous one); solid black: result from the multiscale scheme with tabulated forcing ($J = 16$); dashed black: result from the multiscale scheme with $J = 8$, $\Delta t = 2^{-7}$, $N_1 = R = 1$.

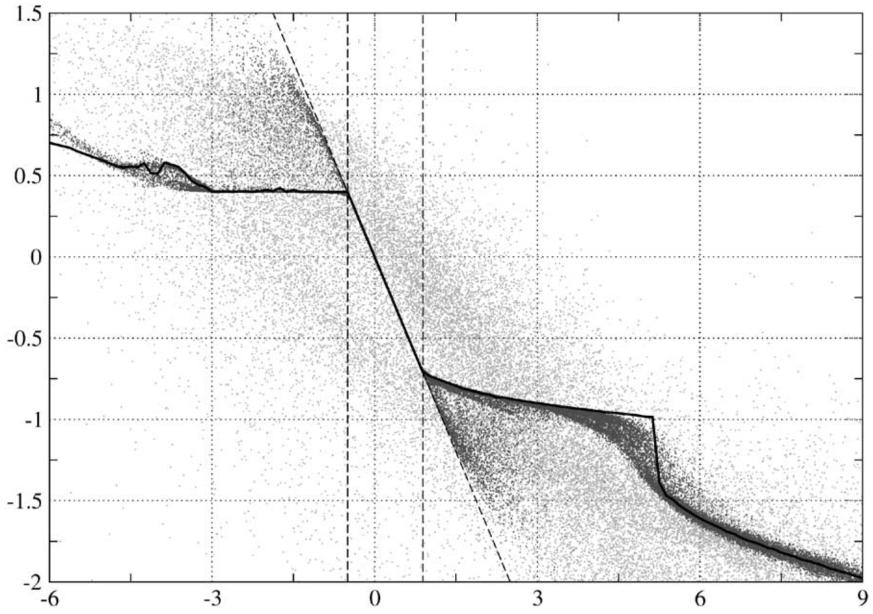

Fig. 12. Dark grey: scatterplots of the bare forcing $\varepsilon h_x \sum_{j=1}^{[1/\varepsilon]} Y_{j,k}$ with $J = 1/\varepsilon = 4096$ $(K = 9)$. Light grey: effective forcing $F_k(X)$ produced on-the-fly by the multiscale scheme with $K = 9$, $J = 8$, $N_1 = 1$, $R = 64$, $\Delta t = 2^{-7}$. Solid black line: tabulated effective forcing computed via the multiscale scheme with $K = 9$, $J = 16$.

resulting in a cost(direct)/cost(multiscale) = 256 increase in efficiency in favor of the multiscale scheme. Yet, it can be seen in Fig. 11 that the multiscale scheme performs extremely well in reproducing the functional dependence of the PDFs and ACFs of X_k. It should also be stressed that the multiscale scheme with $R = 1$ produces a forcing $F(X_k)$ which is only a poor approximation to the asymptotic limit. This can be seen in the scatterplot shown in Fig. 12. The reason is simple: Since J' is rather small in the multiscale algorithm, the forcing $F(X_k)$ does not self-average as in the original equation with $J = [1/\varepsilon]$. This reinforces a point we already made earlier. A good approximation of the effective forcing at each time-step is not necessary to obtain a good approximation of the limiting dynamics.

4.2.2. Multiscale scheme with tabulation of $F(x_k)$

In the present situation, the only advantage of using the multiscale scheme in a on-the-fly procedure as above is that it may be able to capture the hysteresis phenomena by which the fast mode remain metastable for a while on the unstable linear branch of $F(X_k)$. If one neglects this phenomena, we can simply tabulate $F(X_k)$ once and for all, for instance by saving and processing the data provided by the on-the-fly procedure. Once $F(X_k)$ has been tabulated, one can simply simulate the associated limiting equation for X_k, which results of course in an even bigger efficiency gain in favor of the multiscale scheme (infinite in fact by the criterion above since we do not have to simulate the fast variables anymore). We tabulated $F(X_k)$ for $J = 16$ and the results are also presented in Fig. 12. The results of this second procedure in terms of PDF and ACF for the slow modes, shown in Fig. 11, indicate that the hysteresis phenomena described before has a negligible influence on these quantities.

5. L96 with hidden slow variables

Even though we tested the seamless version of the multiscale scheme in Sections 3 and 4, the use of the seamless scheme was avoidable there because slow and fast variables are explicitly separated in the original set-up of L96. Here we make the seamless version unavoidable by modifying L96 so that it contains hidden slow variables. The system we will study is:

$$\begin{cases} \dot{X}_k = -X_{k-1}(X_{k-2} - X_{k+1}) - \frac{1}{\sqrt{J}} \sum_{j=1}^{J} (Y_{j,k+1}^2 - Y_{j,k-1}^2), \\ \dot{Y}_{j,k} = -\dfrac{1}{\varepsilon} Y_{j+1,k}(Y_{j+2,k} - Y_{j-1,k}) - \dfrac{1}{\sqrt{J}} Y_{i,j}(X_{k+1} - X_{k-1}). \end{cases} \tag{37}$$

We will consider these equations when the number J of fast mode scales as ε^{-2}. Besides this scaling, the main difference with (19) is that the coupling between the slow and the fast modes is quadratic in (37) instead of being linear. We have also dropped the forcing and damping terms in (37). This is unessential but it allows us to use equilibrium statistical mechanics to analyze (37) and use these results as an additional benchmark for the numerics. These results, presented next, indicate that as $\varepsilon \to 0$ with $J = \mathrm{O}(\varepsilon^{-2}) \to \infty$, (37) lead to a limiting dynamics for X_k and the following additional slow variables (hidden in (37)):

$$\bar{B}_k = \frac{1}{\sqrt{J}} \sum_{j=1}^{J} Y_{j,k}^2. \tag{38}$$

We start by analyzing the equilibrium statistical properties of (37) then report the results of our numerical experiments.

5.1. Statistical mechanics properties and existence of a limiting dynamics

Due to the absence of forcing and damping in (37), this equation conserves the energy

$$E = \sum_{k=1}^{K}\left(X_k^2 + \sum_{j=1}^{J} Y_{j,k}^2\right). \tag{39}$$

Assuming ergodicity it is reasonable to expect that the invariant measure for system is the uniform distribution on the surface of constant energy, which is just a sphere. This assumption is consistent with the numerical results, see Fig. 13. It follows that the marginal probability density of X_k is

$$\rho_J(x) = Z^{-1}(E - x^2)_+^{(K(J+1)-3)/2}, \tag{40}$$

where Z is a normalization constant and $z_+ = \max(z, 0)$. Assume that E scales as the number of modes in the system, i.e.

$$E = K(J+1)\varepsilon, \tag{41}$$

where ε is the energy density per mode. Then in the limit as $J \to \infty$, $\rho(x)$ reduces to a (zero-mean) Gaussian PDF with variance ε:

$$\rho_J(x) \to \rho(x) = \frac{\mathrm{e}^{-x^2/2\varepsilon}}{\sqrt{2\pi}} \quad \text{as } J \to \infty. \tag{42}$$

A similar argument shows that $\rho(\cdot)$ is also the limiting PDF of any $Y_{j,k}$. By the law of large numbers, this implies that

$$\frac{1}{\sqrt{J}}\sum_{j=1}^{J} Y_{j,k}^2 \to \varepsilon \quad \text{as } J \to \infty, \tag{43}$$

whereas, by the central limit theorem, we conclude that the rescaled variables

$$B_k = \frac{1}{\sqrt{J}}\left(\sum_{j=1}^{J} Y_{j,k}^2 - J\varepsilon\right), \tag{44}$$

are Gaussian random variables in the limit as $J \to \infty$ with zero mean covariance

$$\operatorname{cov}(B_k, B_{k'}) \to 2\varepsilon^2\delta_{k,k'}. \tag{45}$$

This is also confirmed by the numerics.

The variables B_k's are just re-centered versions of the $\bar{B}_k$'s in (38). We claim that the B_k's are hidden slow variables in (37). To see this, note that (37) can be written in terms of X_k and B_k as

$$\begin{cases} \dot{X}_k = -X_{k-1}(X_{k-2} - X_{k+1}) - (B_{k+1} - B_{k-1}), \\ \dot{B}_k = BT_k(Y) - 2\left(\mathscr{E} + \dfrac{1}{\sqrt{J}}B_k\right)(X_{k+1} - X_{k-1}), \end{cases} \tag{46}$$

where

$$BT_k(Y) = \frac{2}{\varepsilon\sqrt{J}}(Y_{J,k-1}Y_{1,k}Y_{2,k} - Y_{J,k}Y_{1,k+1}Y_{2,k+1}) \tag{47}$$

is a boundary term accounting for the interaction between the $Y_{j,k}$ between subsectors k. Because of this term, (46) is not a closed system for (X_k, B_k). But provided that $J/\varepsilon^2 \to J_0 \in (0, \infty)$ as $\varepsilon \to 0$, one clearly sees that the term $BT_k(Y)$ is O(1) in amplitude, implying that the variables B_k's are slow variables complementary to the X_k's. In this limit (46) reduces to the following effective system:

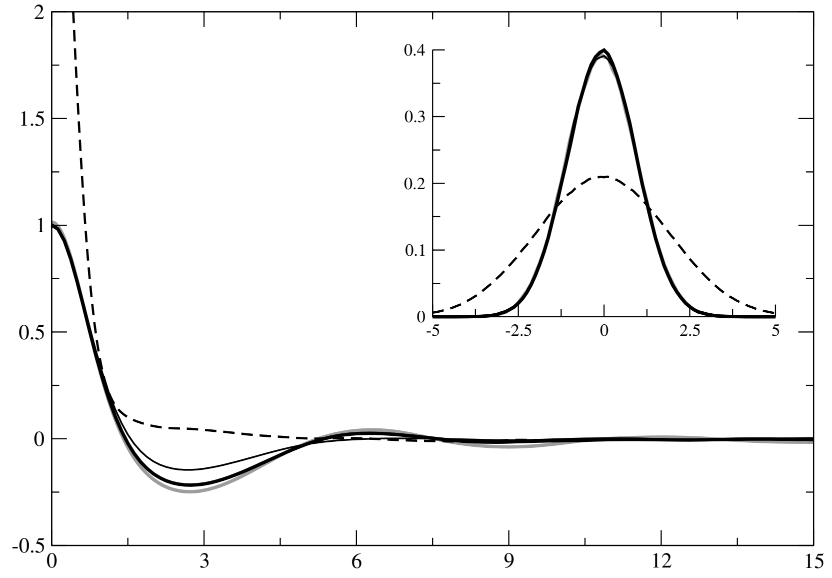

Fig. 13. ACFs and PDFs of X_k. Grey line: direct simulation with $\varepsilon = 1/64$, $J = 512$ ($\delta t = 2^{-10}$); thick solid black line: multiscale scheme with $\delta t = 2^{-8}$, $J = 32$ (efficiency gain: 64); thin black line: multiscale with $\delta t = 2^{-7}$, $J = 8$ (efficiency gain: 512). Dashed black line: multiscale scheme with $\delta t = 2^{-8}$, $J = 32$ (efficiency gain: 64) where the hidden slow variables B_k are not accounted for. The discrepancy clearly indicates that accounting for the B_k's is necessary. In all the multiscale computations $N_1 = R = 1$.

$$\begin{cases} \dot{X}_k = -X_{k-1}(X_{k-2} - X_{k+1}) - (B_{k+1} - B_{k-1}), \\ \dot{B}_k = F_k(X, B) - 2\mathscr{E}(X_{k+1} - X_{k-1}), \end{cases} \tag{48}$$

where the effective forcing $F_k(X; B)$ is the average of $BT_k(Y)$ conditional on X_k, B_k being fixed. Eq. (48) also is the limiting system for (37).

To apply a non-seamless version of the multiscale scheme to (37) would require deriving equations for a set of variables complementary to the X_k's and B_k's, and rewriting (37) in terms of these new variables. This would be a rather tedious operation to do, which we avoid by using the seamless version of the multiscale scheme. Moreover we combine this algorithm with the dimension-reduction technique discussed in the previous section, i.e., in our multiscale simulations we also decrease the value of J to compute the effective forcing.

5.2. Numerical experiments

The first numerical experiment that we perform is a full-scale direct simulation of (37) with $\varepsilon = 1/64$, $J = 1/(8\varepsilon^2) = 512$, $K = 9$. This experiment will serve us as a benchmark for the multiscale simulations. As before we use the fourth-order Runge–Kutta scheme, but we also take additional precautions to conserve the energy E. This is achieved by projecting the X_k's and $Y_{j,k}$'s onto the sphere of constant energy at every micro-time-step (note that it is not necessary to do it so often – even without this projection step the energy is conserved up to 10% throughout the whole computation). The micro-time-step that we use is $\delta t = 2^{-10}$. The results in terms of PDFs and ACFs for X_k are shown in Fig. 13. The B_k's (not shown) behave similarly.

The seamless multiscale scheme which we use is a modification of Algorithm 3. The microscopic dynamics is integrated using the fourth order Runge–Kutta scheme with the same micro-time-step as in the direct computation, $\delta t = 2^{-10}$. We take one realization only, $R = 1$, and we make one micro-time-step per macro-time-step, $N_1 = 1$. For the slow evolution we use a split-step method: fourth order Runge–Kutta scheme for the nonlinear self-interaction of the slow variables X_k and the forward Euler scheme for coupling of the latter with the fast variables. The additional slow variables B_k's are propagated via forward Euler scheme applied to the second equation in (46). We follow by a projection step where we renormalize the fast variables $Y_{j,k}$'s by multiplication so that they lie on energy spheres in y-space consistent with the current value of the B_k's, i.e. we update them as

$$Y_{j,k}^{\text{new}} = \frac{\bar{B}_k}{\sum_{j=1}^{J} (Y_{j,k}^{\text{old}})} Y_{j,k}^{\text{old}}, \tag{49}$$

where $\bar{B}_k = B_k + \varepsilon/\sqrt{J}$, see (38) and (44). This step corresponds to the second loop on r in Algorithm 3. In the present case the nonlinear system of equations which determine Λ can be solved explicitly and results in (49). Finally, we make an additional projection step where we renormalize all the variables to ensure that the total energy E is conserved.

Since the analysis in Section 5.1 shows that the effective dynamics is obtained when $J = \text{O}(\varepsilon^{-2})$, in the multiscale scheme we reduce the number of fast modes as we increase the macro-time-step. Indeed, increasing Δt amounts to increasing the value of ε to $\varepsilon' = \varepsilon\Delta t/\delta t$ (recall that $R = N_1 = 1$), as explained in Section 2.3. Consistently, we use the multiscale scheme with $\Delta t = 2^{-8}$ and $J = 32$ (corresponding to an efficiency gain ratio of 64), and with $\Delta t = 2^{-7}$ and $J = 8$ (corresponding to an efficiency gain ratio of 512). The results are displayed in Fig. 13 and show that the multiscale scheme performs extremely well even at the highest efficiency gain ratio of 512.

Finally, we also did a multiscale simulation where the B_k's are not accounted for. The results displayed in Fig. 13 clearly show that this has a dramatic influence on the PDFs and ACFs which depart significantly from their actual values in this case. The reason for this failure is actually rather easy to understand. As explained in Sections 2.3 and 2.5, the multiscale scheme gains in efficiency because in effect it slows down the fast variables, thereby allowing for a smaller number of micro-time-steps in the computation. But if some slow variables are not accounted for, the multiscale scheme will slow down these variables as well, and this will affect the dynamics of all the slow variables. In our case, using the multiscale scheme without accounting for the B_k's amounts to slowing down the evolution for the $Y_{j,k}$ by a factor of $\delta = \delta t/\Delta t$, i.e. multiplying the second equation in (37) by δ. If this happens the conserved quantity E changes to (compare (39))

$$\tilde{E} = \sum_{k=1}^{K} \left(X_k^2 + \delta \sum_{j=1}^{J} Y_{j,k}^2 \right). \tag{50}$$

Therefore the evolution proceeds on a wrong hypersurface altogether, e.g., the variance of X_k which would normally be $E/K(J+1)$ becomes $\tilde{E}/K(J+1)$, with $\tilde{E} \neq E$ determined by the initial conditions. This is precisely the effect one sees in Fig. 13.

6. Convective versus diffusive time-scales in L96

So far we have used the multiscale scheme on infinite time intervals without worrying about the fact that the underlying theoretical results assert the existence of an effective equation like (2) for (1) only on finite time-intervals. The reason why the multiscale scheme was able to capture correctly the long-time behavior in L96 via PDFs or ACFs was already explained in Section 1. L96 is an intrinsically stochastic system for which the stochastic corrections in the dynamics arising on the $\text{O}(\varepsilon^{-1})$ time-scale have a very weak effect on the long-time dynamics and quantities like the PDFs and ACFs. This is not always the case, though, and even L96 can be tuned in a way that the stochastic corrections arising on the $\mathrm{O}(\varepsilon^{-1})$ time-scale do matter. How to deal with situations of this sort and use the multiscale scheme is the subject of this section.

Consider the same generic model (1) with an additional property that the expectation in (3) is of order ε, i.e.

$$\int_{\mathbb{R}^n} f(x,z)\mu_x(\mathrm{d}z) = \varepsilon\bar{F}(x), \tag{51}$$

where $\bar{F}$ is some function O(1) in ε. Eq. (51) implies that the slow variable is frozen on the O(1) time-scale and the interesting dynamics arises on the $\mathrm{O}(\varepsilon^{-1})$ time-scale. However, this dynamics is not captured by the limiting equation $\dot{X} = \varepsilon\tilde{F}(X)$. The reason is that stochastic effects arising due to fluctuations of the effective forcing around its mean value (51) must be accounted for, and it can be shown [17,18,26,27] that the effective dynamics is in fact captured by a stochastic differential equation

$$\bar{X} = \varepsilon b(X) + \sqrt{\varepsilon}\sigma(X)\dot{W}(t), \tag{52}$$

where $W(t)$ is the standard multi-dimensional Wiener process, and the coefficients b and σ are defined via expectations similar to (though more general than) (51). Note that rescaling time as $\tau = \varepsilon t$, (52) can be written as

$$\dot{X}(\tau) = b(X(\tau)) + \sigma(X(\tau))\dot{W}(\tau). \tag{53}$$

A general multiscale procedure to compute the coefficients b and σ via micro-simulation of the full system in (1) is given in [30] (see also [9]). Here we show how to bypass this procedure using a poor man's version of the multiscale scheme much in the spirit of the penalty methods discussed in Section 2.3. The existence of the effective equation in (53) actually means that, when (51) is satisfied, the solution of the original system and the solution of (53) satisfy

$$\sup_{0\leqslant\tau\leqslant T} |\mathbf{E}\varphi(X^\varepsilon(\tau/\varepsilon)) - \mathbf{E}\varphi(X(\tau))| \to 0 \quad \text{as } \varepsilon\to 0, \tag{54}$$

where φ is a test function and $\mathbf{E}$ denotes the expectation with respect to some appropriate measures. In turns this implies that the solutions of the original equation in (1) computed at two different values of ε, say, ε and ε', satisfy

$$\sup_{0\leqslant\tau\leqslant T} |\mathbf{E}\varphi(X^\varepsilon(\tau/\varepsilon)) - \mathbf{E}\varphi(X^{\varepsilon'}(\tau\varepsilon'))| \to 0, \quad \text{as } \varepsilon,\varepsilon'\to 0. \tag{55}$$

Note the specific rescaling of time in (55) which involves the actual values of ε and is different for X^ε and $X^{\varepsilon'}$. Eq. (54) implies that

$$X^\varepsilon(t) \approx X^{\varepsilon'}(t\varepsilon/\varepsilon') \tag{56}$$

provided that ε and ε' are both sufficiently small.

The idea of a poor man's multiscale scheme for situations of this type is as follows. Since ε is small in the original equation, we can satisfy $\varepsilon \ll \varepsilon' \ll 1$ and still be in the range where (56) holds. But simulating (1) with the new ε' is then much less expensive than with the original ε. This results in efficiency gains that may not be as dramatic as what one can obtain by a full multiscale scheme where the coefficients b and σ are evaluated via micro-simulations. Yet this seamless way to implement a multiscale scheme has the advantage of its great simplicity. Next we demonstrate this on a variant of L96.

Consider

$$\begin{cases} \dot{X}_k = -\varepsilon(X_{k-1}(X_{k-2} - X_{k+1}) + X_k) + \frac{h_x}{J}\sum_{j=1}^{J}(Y_{j,k+1} - Y_{j,k-1}), \\ \dot{Y} = \frac{1}{\varepsilon}(-Y_{j+1,k}(Y_{j+2,k} - Y_{j-1,k}) - Y_{j,k} + F_y) + h_y X_k. \end{cases} \tag{57}$$

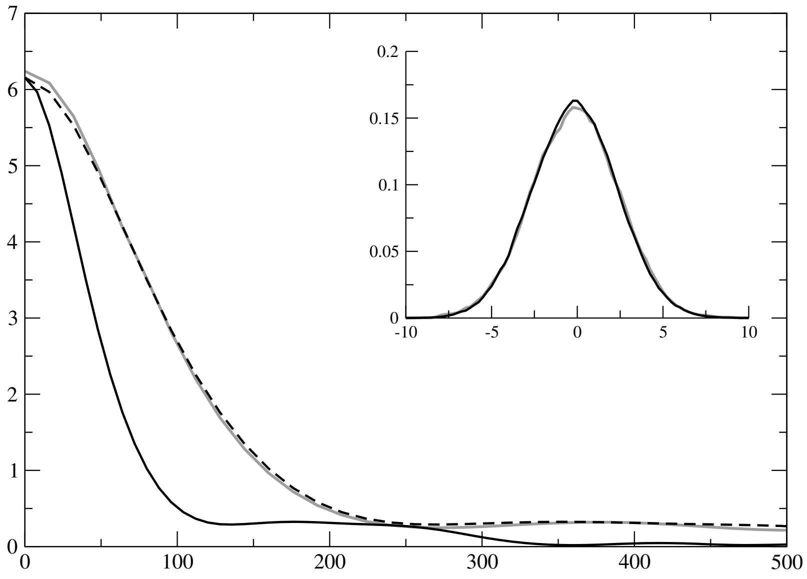

Fig. 14. ACFs and PDFs (subplot) of the slow variable X_k evolving under (57). Grey line: $\varepsilon = 1/256$; full black line: $\varepsilon = 1/128$; dashed black line: $\varepsilon = 1/128$ with time rescaled as $t \to 2t$ consistent with (56). The near perfect match confirms that evolution of X_k converges to some limiting dynamics on the $O(1/\varepsilon)$ time-scale.

With this specific scaling of the nonlinear terms in the equation for X_k, and of the coupling term with X_k in the equation for $Y_{j,k}$, we guarantee that (51) is satisfied. Indeed, the conditional measure entering (51) does not depend on x since the equation used in the microsolver is

$$\dot{Z}_{j,k} = \frac{1}{\varepsilon}(-Z_{j+1,k}(Z_{j+2,k} - Z_{j-1,k}) - Z_{j,k} + F_y). \tag{58}$$

In addition the statistics of $Z_{j,k}$ does not depend on k by periodicity and therefore the conditional averages of $Y_{j,k+1}$ and Y_{j,k_1} at x fixed are the same, and the correspond term involving their difference in (57) cancels to leading order.

We study (57) by the multiscale scheme described above by performing two simulations with $J = 2$, $K = 4$, $F_y = 10$, $h_x = -0.8$, $H_y = 1$, and $\varepsilon = 1/256$ and 1/128, and comparing the solution using the proper rescaling of time given in (56). The results are displayed in Fig. 14 and show the almost perfect agreement in PDFs and ACFs. The simulation with $\varepsilon = 1/128$ is performed with a micro-time-step which is twice as big as the one used in the simulation with $\varepsilon = 1/256$ and therefore corresponds to an efficiency gain ratio of 2. Notice that, in the present situation, the use of the multiscale scheme can be bypassed since the dependency in ε is explicit in (57). But this need not be the case, and the poor man's multiscale scheme used here can be straightforwardly generalized to systems such as the one considered in Section 5 which contain hidden slow variables.

7. Concluding remarks

We have investigated a class of numerical schemes which fit within the general framework of the Heterogeneous Multiscale Method and allow for significant computational gain in efficiency in systems with multiple spatio-temporal scales. We have applied these schemes on the example of L96 which is the prototype of a (moderately) large deterministic system displaying chaotic behavior. While precise error estimates are not currently available for systems of this type, we have shown that the analysis of the multiscale scheme in terms of its consistency with specific dynamical equations allows a rather complete understanding of the efficiency of the method in terms of the numerical parameters used. This analysis was made in general and confirmed for the specific example of L96. For this example, it was shown that the multiscale scheme can be adapted and improved by utilizing specific properties of the system so as to achieve not only considerable gain in efficiency in the computations, but also good understanding of the properties of limiting equations for the slow modes in L96. We believe that a similar strategy may be useful in more realistic examples arising from the atmospheric sciences, molecular dynamics, etc. which display the same type of multi-scale characteristic as L96 and pose similar computational and theoretical challenges.

Acknowledgements

We thank D. Cai, A.J. Chorin, W. E, D. Liu and A.J. Majda for interesting discussions. E.V.-E. is partially supported by NSF via grants DMS01-01439, DMS02-09959 and DMS02-39625.

Appendix A. More algorithms and speculative error estimates

Here we give specific multiscale algorithms and assess their performance. For simplicity we focus on using one-step explicit schemes with fixed time-steps for both the macro- and the micro-solver. The generalization to variable time-steps, or explicit multi-step schemes is straightforward; the generalization to implicit schemes for the micro-solver is straightforward as well, but using implicit schemes as macro-solver is more involved and should probably be avoided if possible. We also speculate on the performance of various estimators and insist on distinguishing (i) the error the estimator makes on evaluating $F(x)$ at each macro-time-step, and (ii) the error it induces on the slow variable evolution. We focus on using the ensemble-average as estimator; similar error estimates can be derived if time-averaging is used [9]. We stress that the error estimates given below are speculative in the sense that they depend on the convergence properties of the macro- and micro-solvers on infinite time-intervals, which we shall assume are well-behaved and known. Under these assumptions, the statements below can be proven following the lines of [9] and we refer the reader to this paper for details.

A.1. Algorithms

The generalization of Algorithm 1 to a multiscale scheme where the macro-solver uses a Q-stage one-step method with Butcher coefficients $a_{q,q'}$ and b_q can be written as:

Algorithm 5. ***(Q-macro-stage one-step macro-solver)***

```
Take $\tilde{X}_0 = x$; $M = \lceil T/\Delta t \rceil$; $m = 0$;
while $m \leqslant M$;
$\tilde{X}^\star_0 = \tilde{X}_m$;
for $q = 0, 1, \dots, Q-1$
  call $\tilde{F}(\tilde{X}^\star_q)$
  $\tilde{X}^\star_{q+1} = \tilde{X}^\star_q + \Delta t \sum_{q'=0}^{q} a_{q,q'} \tilde{F}(\tilde{X}^\star_{q'})$;
end(for)
call $\tilde{F}(\tilde{X}^\star_Q)$
$\tilde{X}_{m+1} = \tilde{X}_m + \Delta t \sum_{q=0}^{Q} b_q \tilde{F}(\tilde{X}^\star_q)$;
```

```
  $m \leftarrow m+1$;
***end(while)***
```

Here the function $\tilde{F}(x)$ invokes a subroutine which contains both the micro- solver and the estimator and can be integrated into the following subroutines to evaluate $\tilde{F}(x)$ via ensemble-averaging:

Algorithm 6. ***(micro-solver with estimator by ensemble-averaging)***

```
***function*** $\tilde{F}(\tilde{X}m)$
***given*** $\tilde{X}m, \{Z^r_{N_1,m-1}\}_{r=1}^R$:
***for*** $r = 1,\dots,R$
  $Z^r_{0,m} = Z^r_{N_1,m-1}$;
  ***for*** $n = 0,1,\dots,N_1-1$
    $Z^r_{n+1,m} = Z^r_{n,m} + \delta t \psi_g(\tilde{X}_m, Z^r_{n,m}, \delta t)$;
  ***end(for)***
  $\tilde{F}(\tilde{X}_m) \leftarrow \tilde{F}(\tilde{X}_m) + \frac{1}{R} f(\tilde{X}_m, Z^r_{N_1,m})$;
***end(for)***
***return*** $\tilde{F}(\tilde{X}_m), \{Z^r_{N_1,m}\}_{r=1}^R$.
```

It is also straightforward to implement a Q-stage generalization of the seamless algorithm (3). In this case one has to use the same procedure as described in (Section 2.4) to obtain the values of U^r at every intermediate point and therefrom calculate the macroscopic forcing and advance the slow variables.

A.2. Error estimates

A.2.1. Macro-solver

Let X_m be the numerical approximation of $X_{t=m\Delta t}$ provided by Algorithm 5 with the exact $F(x)$. We will assume that a weak error estimate of the following type holds: for all test functions $\eta(x)$ within a suitable class, there exists $\alpha > 0$ such that for Δt sufficiently small

$$\left| \lim_{M\to\infty} \frac{1}{M} \sum_{m=1}^{M} \eta(X_m) - \int_{\mathbb{R}^m} \eta(x)\mu(\mathrm{d}x) \right| \leqslant C\Delta t^\alpha, \tag{A.1}$$

where $\mu(\mathrm{d}x)$ is the invariant measure of the limiting equation in (2) (assuming it exists) and C is a generic constant. The exponent α depends on the macro-scheme, and may be less than the order of accuracy of the method over finite time-interval [29]. Thus (A.1) assumes that both the limiting equation for X in (2) and the macro-solver are ergodic; the criterion (A.1) is then tailored for assessing the performance of the scheme in approximating moments of $\mu(\mathrm{d}x)$ (in the context of SDEs, a scheme satisfying (A.1) is said to converge with respect to the ergodic criterion with order α [16]). Such a criterion is appropriate in the context of systems like L96 where both the original and the effective dynamics display deterministic chaos and for which a detailed description of a given trajectory is therefore only possible on a rather short interval of time. Of course, if such description on a short time-interval is of interest, the criterion in (A.1) can be changed into a standard ODE error estimate whose order of accuracy then simply is the order of the macro-solver; the discussion below can be straightforwardly adapted to this case.

When (A.1) holds, it is a general result of HMM [7] that the error estimate for Algorithm 5 is then

$$\left| \lim_{M\to\infty} \frac{1}{M} \sum_{r=1}^{M} \eta(\tilde{X}_m) - \int_{\mathbb{R}^m} \eta(x)\mu(\mathrm{d}x) \right| \leqslant C(\Delta t^\alpha + e(HMM)), \tag{A.2}$$

where $e(HMM)$ is an additional term accounting for the errors from the micro-solver and the estimator, which we evaluate next.

A.2.2. Estimator

Denote by $\tilde{F}(x,z)$ the approximation of $F(x)$ provided by estimating (3) by time-averaging, i.e.

$$\tilde{F}(x,z) = \frac{1}{R}\sum_{r=1}^{R} f(x, Z^r_{N_1}(x,z), \varepsilon), \tag{A.3}$$

where N_1 and R are as in Algorithm 6, and we have indicated explicitly the dependency of $\tilde{F}(x,z)$ and $Z_n(x,z)$ on the initial conditions for the micro-solver, i.e. $Z_{n=0}(x,z) = z$, since they affect the error estimates we give next.

To assess the performance of the estimator integrated within the multiscale scheme, we introduce the following error estimate for (A.3) which would be standard in the context of SDEs. We assume that there exist $\beta > 0$ such that for ε, Δx, and δt small enough, and N_1 and N, with $N_1 < N$, large enough, we have

$$\int |\tilde{F}(x,z) - F^\varepsilon(x)|\mu_{x+\Delta x}(\mathrm{d}z)\mu(\mathrm{d}x) \leqslant C\left((\delta t/\varepsilon)^\beta + \Delta x\, \mathrm{e}^{-\bar{C}N_1\delta t/\varepsilon} + \frac{1}{\sqrt{R}}\right), \tag{A.4}$$

where $C, \bar{C}$ are generic constants. Eq. (A.4) assume that measure $\mu_x(\mathrm{d}z)$ is weakly Lipschitz in x (i.e.$\int_{\mathbb{R}}^m \eta(z)\mu_x(\mathrm{d}z)$ is Lipschitz in x for all test functions η). The origin of the various terms at the right hand-side in these estimates is as follows.

The first term accounts for the discretization errors from the micro-solver and it is assumed that the micro-solver converges with respect to the ergodic criterion of order β.

The second term accounts for relaxation errors from the initial condition used in the micro-solver. This is a property of the dynamics, not of the solver; here it is assumed that the relaxation is exponential, though this is not essential and a slower algebraic decay would suffice. What is essential is that Δx is small in (A.4); this amounts to assuming that the initial condition for the micro-solver is $Z_0(x + \Delta x, z)$, i.e. it is sampled from the measure $\mu_{x+\Delta x}(\mathrm{d}z)$ which is very close to the one entering the definition of $F^\varepsilon(x)$. This is precisely what happens if one combines Algorithm 5 with 6 since the initial conditions for the fast variables at the next call of these subroutines is their final values from the last call, i.e. they already sample $\mu_{\tilde{X}_{m-1}}(\mathrm{d}z)$ initially when one let them evolve to sample $\mu_{\tilde{X}_m}(\mathrm{d}z)$, and $\tilde{X}_m - \tilde{X}_{m-1} = \mathrm{O}(\Delta t)$.

Finally, the last term account for finite sampling effects. Unlike the first two terms, for standard SDEs this term disappears if the absolute value in (A.4) is taken outside instead of inside the integral; we shall assume that this is also the case in the present context.

A.2.3. HMM

It is shown in [9] that an estimate like (A.4) is the essential information needed to evaluate $e(HMM)$ in (A.2). In particular, Algorithm 5 combined with Algorithm 6 leads to

$$e(HMM) = C\left((\delta t/\varepsilon)^\beta + \Delta t\, \mathrm{e}^{-\bar{C}N_1\delta t/\varepsilon} + \frac{\Delta t}{R}\right). \tag{A.5}$$

We have already explained why the term accounting for relaxation error in these estimates is proportional to Δt. Note also that the sampling error term enters squared compared with (A.4), and with an additional Δt, which confirms that the effective number of realizations is $R/\Delta t$, i.e. the scheme converges as $\Delta t \to 0$ even if $R = 1$.

A.3. Efficiency

From (A.2) and (A.5) one sees that the optimal cost of the multiscale scheme, taken as the total number of micro-time-steps, at error tolerance $\lambda \ll 1$ is

$$\text{cost} = \frac{TQRN_1}{\Delta t} = \mathrm{O}(\lambda^{-\min\{1/\alpha,1\}-1/\beta}\log\lambda^{-1}), \tag{A.6}$$

the choice of parameters leading to this estimate is

$$\Delta t = \mathrm{O}(\lambda^{1/\alpha}), \quad \delta t = \mathrm{O}(\varepsilon\lambda^{1/\beta}), \quad N_1 = \mathrm{O}(\lambda^{1/\beta}\log\lambda^{-1}), \quad R = \mathrm{O}(\lambda^{\max\{1/\alpha-1,0\}}). \tag{A.7}$$

Note that, for $\alpha \leqslant 1$, the optimal number of realizations is $R = 1$; for $\alpha > 1$, it scales as $\lambda^{1/\alpha-1}$. Notice also that the cost is independent of ε, which explains why the multiscale scheme is more efficient than a direct scheme for (1).

References

[1] R.V. Abramov, A.J. Majda, Quantifying uncertainty for non-Gaussian ensembles in complex systems, SIAM J. Sci. Comp., submitted for publication.
[2] U.M. Ascher, L.R. Petzold, Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations, SIAM, Philadelphia, 1998.
[3] R. Car, M. Parrinello, Unified approach for molecular dynamics and density-functional theory, Phys. Rev. Lett. 55 (1985) 2471–2474.
[4] A.J. Chorin, A numerical method for solving incompressible viscous flow problem, J. Comput. Phys. 2 (1967) 12–26.
[5] A.J. Chorin, Computational Fluid Mechanics, Academic Press, New York, 1989.
[6] W. E, Analysis of the heterogeneous multiscale method for ordinary differential equations, Commun. Math. Sci. 1 (2003) 423–436.
[7] W. E, B. Enguist, The heterogeneous multi-scale methods, Commun. Math. Sci. 1 (2003) 87–133.
[8] W. E, B. Enguist, Multiscale modeling and computations, Notices Am. Math. Soc. 50 (2003) 1062–1070.
[9] W. E, D. Liu, E. Vanden-Eijnden, Analysis of numerical techniques for multiscale stochastic dynamical systems, Commun. Pure Appl. Math., submitted for publication.
[10] C.W. Gear, Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, New York, 1971.
[11] C.W. Gear, I.G. Kevrekidis, Projective methods for stiff differential equations: problems with gap in their eigenvalue spectrum, SIAM J. Sci. Comp. 24 (2003) 1091–1106.
[12] C.W. Gear, I.G. Kevrekidis, Telescopic projective integrators for stiff differential equations, J. Comp. Phys. 187 (2003) 95–109.
[13] E. Hairer, G. Wanner, Solving Ordinary Differential Equations. II: Stiff and Differential-Algebraic Problems, Springer-Verlag, New York, 1991.
[14] W. Just, H. Kantz, C. Rödenbeck, M. Helm, Stochastic modelling: replacing fast degrees of freedom by noise, J. Phys. A 34 (2001) 3199–3213.
[15] I.G. Kevrekidis, C.W. Gear, J.M. Hyman, P.G. Kevrekidis, O. Runborg, C. Theodoropoulos, Equation-free, coarse-grained multiscale computations: enabling microscopic simulators to perform system-level analysis, Commun. Math. Sci. 1 (2003) 715–762.
[16] P.E. Kloeden, E. Platen, Numerical solutions of stochastic differential equations, in: Applications of Mathematics Series, vol. 23, Springer-Verlag, Heidelberg, 1992.
[17] T.G. Kurtz, A limit theorem for perturbed operator semigroups with applications to random evolutions, J. Functional Anal. 12 (1973) 55–67.
[18] T.G. Kurtz, Semigroups of conditioned shifts and approximations of Markov processes, Ann. Prob. 3 (1975) 618–642.
[19] V.I. Lebedev, S.I. Finogenov, Explicit methods of second order for the solution of stiff systems of ordinary differential equations, in: A. Iserles (Ed.), Acta Numerica, Cambridge University Press, Cambridge, 1997, pp. 437–484.
[20] E.N. Lorenz, Predictability A problem partly solved, in: ECMWF Seminar Proceedings on Predictability, Reading, United Kingdom, ECMWF, 1995, pp. 118.
[21] E.N. Lorenz, K.A. Emanuel, Optimal sites for supplementary weather observations: simulation with a small model, J. Atmos. Sci. 55 (1998) 399–414.

[22] A.J. Majda, I. Timofeyev, E. Vanden-Eijnden, Models for stochastic climate prediction, Proc. Natl. Acad. Sci. USA 96 (1999) 14687–14691.

[23] A.J. Majda, I. Timofeyev, E. Vanden-Eijnden, A mathematical framework for stochastic climate models, Commun. Pure Appl. Math. 54 (2001) 891–974.

[24] A.J. Majda, I. Timofeyev, E. Vanden-Eijnden, A priori tests of a stochastic mode reduction strategy, Physica D 170 (2002) 206–252.

[25] A.J. Majda, I. Timofeyev, E. Vanden-Eijnden, Systematic strategies for stochastic mode reduction in climate, J. Atmos. Sci. 60 (2003) 1705–1722.

[26] G.C. Papanicolaou, Some probabilistic problems and methods in singular perturbations, Rocky Mountain J. Math. 6 (1976) 653–674.

[27] G.C. Papanicolaou, Introduction to the asymptotic analysis of stochastic equations, in: Modern Modeling of Continuum phenomena, in: R.C. DiPrima (Ed.), Lectures in Applied Mathematics, vol. 16, American Mathematical Society, 1977.

[28] C. Rödenbeck, C. Beck, H. Kantz, Dynamical systems with time scale-separation: averaging, stochastic modeling, and central limit theorems, in: Stochastic Climate Models, in: P. Imkeller, J.-S. vonStorch (Eds.), Progress in Probability, vol. 49, Birkhäuser Verlag, Basel, 2001, p. 187209.

[29] A.M. Stuart, A.R. Humphries, Dynamical systems and numerical analysis, in: Cambridge Monographs on Applied and Computational Mathematics, vol. 2, Cambridge University Press, Cambridge, 1996.

[30] E. Vanden-Eijnden, Numerical Techniques for multiscale dynamical systems with stochastic effects, Comm. Math. Sci. 1 (2003) 385–391.

[31] V.M. Volosov, Averaging in systems of ordinary differential equations, Uspehi Mat. Nauk. 17 (1962) 3–126;
Russian Math, Surveys 17 (1962) 1–126.