# A class of robust numerical methods for solving dynamical systems with multiple time scales

Thomas Y. Hou[a], Zhongjian Wang[b], Zhiwen Zhang[b,*]

[a]*Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125, USA.*
[b]*Department of Mathematics, The University of Hong Kong, Pokfulam Road, Hong Kong SAR, China.*

## Abstract

In this paper, we develop a class of robust numerical methods for solving dynamical systems with multiple time scales. We first represent the solution of a multiscale dynamical system as a transformation of a slowly varying solution. Then, under the scale separation assumption, we provide a systematic way to construct the transformation map and derive the dynamic equation for the slowly varying solution. We also provide the convergence analysis of the proposed method. Finally, we present several numerical examples, including ODE system with three and four separated time scales to demonstrate the accuracy and efficiency of the proposed method. Numerical results verify that our method is robust in solving ODE systems with multiple time scale, where the time step does not depend on the multiscale parameters.
**Keyword**: Hamiltonian dynamical system; multiple time scales; stiff equations; convergence analysis; uniform accuracy; composition maps.
**AMS subject classifications.** 34E13, 65L04, 65P10, 65L20.

## 1. Introduction

Dynamical systems with sub-processes evolving on many different time scales are ubiquitous in applications: chemical reactions, electro-optical and neuro-biological systems, to name just a few [19, 13]. The multiple time scales in the dynamical systems pose a major problem in numerical simulations because one needs to choose small time steps for stable integration of the fast motions in the systems, which leads to large numbers of time steps required for the observation of the slow degrees of freedom and thus requires tremendous computational resources. Interested readers are referred to [8, 15] and the references therein for a detailed review.

The objective of this paper is to develop a new method to solve dynamical systems with multiple time scales; see Section 2 for the precise definition of the problems. The main idea of our method is to formally represent the solution of the multiscale dynamical system as a transformation of a slowly varying solution; see Eq.(14). By dealing with the multiscale

---

information in a dimension-by-dimension fashion, we propose a systematic way to construct a set of cumulative composition maps that capture the complicated dynamics of the problem. Based on the scale separation assumption, we successfully derive the dynamic equation for the slowly varying solution (i.e., Eq.(30)) and prove that the dynamic equation for the slowly varying solution is non-stiff; see Theorem 3.2. Thus, we can use conventional numerical methods to compute it, where the time step is independent of the multiscale parameters in the dynamical system. In addition, we analyze the error between the numerical solution obtained from our method and the exact solution in Theorem 3.4. Finally, we carry out several numerical experiments to demonstrate the accuracy and efficiency of the proposed method.

As we will demonstrate in Section 4, the proposed method can offer accurate numerical solutions to multiscale ODE systems with considerable computational savings over traditional methods, especially when the multiscale parameters are small. Numerical results (see Fig.10) show that the dynamic equation for the slowly varying solution based on the cumulative composition maps indeed capture the averaged behaviors of the solution well. While a simple averaging treatment of the original multiscale ODE systems leads to wrong results. As an analogy to this interesting finding, in the homogenization for elliptic PDEs with multiscale coefficients, a simple average of the coefficient gives a wrong result, where one needs to solve a cell problem to obtain the correct homogenization coefficient [5].

Our method is inspired by the recent development in designing uniformly accurate numerical schemes for highly oscillatory evolution equations [3, 4], where two-scale problems were solved. In [3, 4], the authors separate the two time scales into two independent variables and embed the solution of the two-scale problem into a two-variable function. Then, they derive formulations of the evolution equations for the two-variable function and prove that under certain conditions the evolution equations are solvable and non-stiff.

The novelty of our paper is that we provide a systematical way to construct a set of cumulative composition maps that allow us to correctly upscale the complicated dynamics of the problem. Notice from Eq.(25) that each map $\Phi^k$ is a perturbation of the identity operator. However, a cumulative composition of those simple maps (15) can provide an accurate approximation of the complicated dynamics of the problem. In addition, we provide a rigorous convergence analysis for the proposed method and verify the statement through numerical experiments.

Before we end this section, we give a short review of several existing methods for solving two-scale problems. When slow variables can be identified, effective equations can be obtained by averaging the instantaneous drift driving those slow variables. Two classes of numerical methods have been developed based on this observation: the equation-free method [14] and heterogeneous multiscale method (HMM) [1]. Later on, a new class of integrators for stiff ODEs as well as SDEs were developed [18], which are based on the averaging of the instantaneous flow of the hidden slow and fast variables simultaneously. Therefore, the hidden slow variables do not need to be explicitly identified. In this paper, however, we will consider problems parameterized by multiple time scales. In addition, we aim to design nu-

merical schemes that solve the multiscale dynamical problems for a wide range of multiscale parameter values with uniform accuracy.

The rest of the paper is organized as follows. In Section 2, we will derive our numerical method for solving multiscale dynamical systems and discuss its detailed implementation. In Section 3, we provide the convergence analysis for the proposed method. In Section 4, we present numerical results to demonstrate the accuracy and efficiency of our method. Concluding remarks are made in Section 5.

## 2. Numerical methods for solving multiscale dynamical systems

In this section, we will develop numerical methods to solve dynamical systems with multiple time scales. Specifically, we consider the following first-order ordinary differential equation (ODE) system to illustrate the main idea,

$$\dot{x} = f^{\epsilon}(t, x), \quad x(0) = x_0, \quad t \in [0, T], \tag{1}$$

where $x(t) \in \mathbb{R}^d$ is the solution vector, $x_0$ is the initial value, and $f^{\epsilon}(t, x)$ is a function vector field. Here $\epsilon = (\epsilon_1, ..., \epsilon_n)$ is a set of parameters, which are used to characterize different time scales in the ODE system (1). When the parameters satisfy

$$0 < \epsilon_n \ll \epsilon_{n-1} \ll \cdots \cdots \ll \epsilon_1 \ll 1, \tag{2}$$

we say that the multiscale time scales are well-separated. Given the multiscale parameters, we denote

$$f^{\epsilon}(t, x) \equiv f(\frac{t}{\epsilon_1}, \frac{t}{\epsilon_2}, \cdots, \frac{t}{\epsilon_n}, x), \tag{3}$$

where $f : \mathbb{R}^{n+d} \to \mathbb{R}^d$ is a function vector field. We assume that the first-order derivatives of $f$ are bounded, which is sufficient to guarantee the existence and uniqueness of solutions of the ODE system (1) [12, 16]. Let $t_i = \frac{t}{\epsilon_i}$, $i = 1, ..., n$. We denote $f^{\epsilon}(t, x) = f(t_1, t_2, \cdots, t_n, x)$. Moreover, we assume that $f$ is periodic with respect to its first $n$ coordinates, i.e., $t_i$, $i = 1, ..., n$. Without lost of generalities, all the periods are assumed to be 1.

*2.1. Decomposition of the multiscale function $f$*

We iteratively define the averaged functions to resolve finer scale fluctuations on coarser scales in a dimension-by-dimension fashion. We first start from the coordinate $t_n$ corresponding to the smallest-scale and define the mean function

$$\bar{f}^n(t_1, t_2, \cdots, t_{n-1}, x) = \int_0^1 f(t_1, t_2, \cdots, t_{n-1}, s, x) ds, \tag{4}$$

and the fluctuation function

$$f^n(t_1, t_2, \cdots, t_n, x) = f(t_1, t_2, \cdots, t_n, x) - \bar{f}^n(t_1, t_2, \cdots, t_{n-1}, x), \tag{5}$$

where the integration and subtraction are done in a component-wise fashion. Thus, $\bar{f}^n$ and $f^n$ are $d$-dimensional vector functions.

3

Then, based on the function $\bar{f}^n(t_1, t_2, \cdots, t_{n-1}, x)$, we define the mean function and fluctuation function corresponding to the second smallest-scale as follows

$$\bar{f}^{n-1}(t_1, t_2, \cdots, t_{n-2}, x) = \int_0^1 \bar{f}^n(t_1, t_2, \cdots, t_{n-2}, s, x) ds, \tag{6}$$

$$f^{n-1}(t_1, t_2, \cdots, t_{n-1}, x) = \bar{f}^n(t_1, t_2, \cdots, t_{n-1}, x) - \bar{f}^{n-1}(t_1, t_2, \cdots, t_{n-2}, x). \tag{7}$$

We continue this strategy and define mean functions and fluctuation functions corresponding to different time scales recursively. For instance, given the mean function $\bar{f}^{n-k+1}$, we define the mean function and fluctuation function corresponding to a coarser-scale as follows

$$\bar{f}^{n-k}(t_1, t_2, \cdots, t_{n-k-1}, x) = \int_0^1 \bar{f}^{n-k+1}(t_1, t_2, \cdots, t_{n-k-1}, s, x) ds, \tag{8}$$

$$f^{n-k}(t_1, t_2, \cdots, t_{n-k}, x) = \bar{f}^{n-k+1}(t_1, t_2, \cdots, t_{n-k}, x) - \bar{f}^{n-k}(t_1, t_2, \cdots, t_{n-k-1}, x). \tag{9}$$

Finally, we define the mean function and fluctuation function corresponding to the largest-scale as follows

$$\bar{f}^1(x) = \int_0^1 \bar{f}^2(s, x) ds, \tag{10}$$

$$f^1(t_1, x) = \bar{f}^2(t_1, x) - \bar{f}^1(x). \tag{11}$$

The above recursive formulations (4)-(11) naturally lead to a decomposition of the multiscale function $f(t_1, t_2, \cdots, t_n, x)$ into

$$f = \sum_{k=1}^n f^k + \bar{f}^1. \tag{12}$$

According to the definition, for each $k$ we have that

$$\int_0^1 f^k(t_1, \cdots, t_{k-1}, s, x) ds = 0, \quad \forall \, t_i \in [0,1], \, i = 1, ..., k-1, \text{ and } x. \tag{13}$$

*2.2. Derivation of the dynamic equation for the slowly varying solution*

We will construct a family of maps $\Phi^k = \Phi^k_{t_1, t_2, \cdots, t_k} : \mathbb{R}^d \to \mathbb{R}^d$, $k = 1, ..., n$, that allow us to represent the solution $x(t)$ of the ODE system (1) as a transformation of a slowly varying solution $y(t)$, i.e.,

$$x(t) = \Phi^n \circ \Phi^{n-1} \circ \cdots \circ \Phi^1(y(t)). \tag{14}$$

We assume that each map $\Phi^k$ is periodic with respect to $t_k$ and becomes the identical map $I_d$ when $t_k = 0$. To simplify the notation, we define a family of cumulative composition maps as follows

$$\bar{\Phi}^k = \Phi^k \circ \Phi^{k-1} \circ \cdots \circ \Phi^1, \quad k = 1, ..., n. \tag{15}$$

The rationale behind the representation (14) is that the complicated dynamics of the ODE system (1) (e.g., highly oscillatory solutions) is captured by the map $\bar{\Phi}^n$, and thus the evolution of the solution $y(t)$ is smooth. Therefore, we can compute the solution $x(t)$ (through

4

solving $y(t)$) by using numerical methods with relatively large time steps (independent of the multiscale parameters). To achieve this goal, we need to find a constructive way to obtain the map $\bar{\Phi}^n$ and to identify the dynamical equation for the solution $y(t)$.

Substituting the representation (14) into the original problem (1), we know that the solution $y(t)$ formally satisfies the following equation

$$\partial_t \bar{\Phi}^n(y) + \partial_x \bar{\Phi}^n(y)\dot{y} = f^\epsilon, \tag{16}$$

where $\partial_x \bar{\Phi}^n(y)$ is the Jacobian matrix.

It will be complicated if we directly compute $\partial_x \bar{\Phi}^n(y)$ using the chain rule. We shall adopt some approximate method to address this difficulty. Notice that the scale separation assumption on $f^\epsilon$ (see Eq.(3)) indicates that the local fluctuation of $f^\epsilon$ at $\mathcal{O}(\epsilon_n)$ scale can be resolved by the same scale part of $\partial_t \bar{\Phi}^n$. More precisely, by the chain rule, we get

$$\partial_t \bar{\Phi}^n = \frac{1}{\epsilon_n}\partial_{t_n}\Phi^n(\bar{\Phi}^{n-1}) + \sum_{i=1}^{n-1}\frac{1}{\epsilon_i}\partial_{t_i}\Phi^n(\bar{\Phi}^{n-1}) + \partial_x\Phi^n\partial_t\bar{\Phi}^{n-1}, \tag{17}$$

where the last two terms are independent of $t_n$. Compared with the decomposition of the multiscale function $f$ in (12), we can set,

$$\frac{1}{\epsilon_n}\partial_{t_n}\Phi^n(\bar{\Phi}^{n-1}) = f^n. \tag{18}$$

Since $\Phi^n$ is an identical map when $t_n = 0$, we can explicitly get,

$$\Phi^n_{t_1,t_2,\cdots,t_n}(x) = x + \epsilon_n g^n(t_1, t_2, \cdots, t_n, x), \quad \forall x, \tag{19}$$

where

$$g^n(t_1, t_2, \cdots, t_n, x) = \int_0^{t_n} f^n(t_1, t_2, \cdots, t_{n-1}, s, x)ds. \tag{20}$$

From (19), we can see that $\Phi^n = I_d + \mathcal{O}(\epsilon_n)$, which is an $\mathcal{O}(\epsilon_n)$ order perturbation of the identity operator. When the time scales are well-separated, we can dismiss the second term in Eq.(17) since

$$\sum_{i=1}^{n-1}\frac{1}{\epsilon_i}\partial_{t_i}\Phi^n(\bar{\Phi}^{n-1}) = \sum_{i=1}^{n-1}\frac{\epsilon_n}{\epsilon_i}\partial_{t_i}g^n(\bar{\Phi}^{n-1}) = \mathcal{O}(\frac{\epsilon_n}{\epsilon_{n-1}}). \tag{21}$$

Let us continue our derivation inductively with $k = n-1, \cdots, 1$. We consider the fluctuation within period $\mathcal{O}(\epsilon_k)$, and have the following observation,

$$\partial_x(\Phi^n \circ \Phi^{n-1} \circ \cdots \circ \Phi^{k+1})(\frac{1}{\epsilon_k}\partial_{t_k}\Phi^k) = f^k. \tag{22}$$

Due to the scale separation structure of $\Phi^n \circ \Phi^{n-1} \circ \cdots \circ \Phi^{k+1}$, we obtain

$$\partial_x(\Phi^n \circ \Phi^{n-1} \circ \cdots \circ \Phi^{k+1}) = I_d + \mathcal{O}(\epsilon_{k+1}), \tag{23}$$

which is an $\mathcal{O}(\epsilon_{k+1})$ order perturbation of the identity operator. Now we arrive at,

$$\frac{1}{\epsilon_k}\partial_{t_k}\Phi^k(\bar{\Phi}^{k-1}) = f^k. \tag{24}$$

Again, using the condition that $\Phi^k$ is an identical map when $t_k = 0$, we get

$$\Phi^k_{t_1,t_2,\cdots,t_k}(x) = x + \epsilon_k g^k(t_1, \cdots, t_{k-1}, t_k, x), \tag{25}$$

where

$$g^k(t_1, \cdots, t_{k-1}, t_k, x) = \int_0^{t_k} f^k(t_1, \cdots, t_{k-1}, s, x)ds, \quad k = 1, ..., n. \tag{26}$$

In the above derivation, we have used the condition that $\sum_{i=1}^{k-1}\frac{1}{\epsilon_i}\partial_{t_i}\Phi^k(\bar{\Phi}^{k-1}) = \mathcal{O}(\frac{\epsilon_k}{\epsilon_{k-1}})$.

After we obtain the explicit formulations for $\Phi^k$, $k = 1, ..., n$ and their derivatives, we are in the position to derive the dynamic equation for the solution $y(t)$. According to (16), we obtain a nested equation

$$\partial_t\Phi^n + \partial_x\Phi^n\left(\partial_t\Phi^{n-1} + \partial_x\Phi^{n-1}\left(\partial_t\Phi^{n-2} + \cdots(\partial_t\Phi^1 + \partial_x\Phi^1\dot{y}(t))\cdots\right)\right) = f^\epsilon. \tag{27}$$

From the definition of $\Phi^k$ (see Eq.(25)), we compute the derivative of $\Phi^k$ with respect to time $t$ and get,

$$\partial_t\Phi^k = \epsilon_k\sum_{r=1}^{k}\frac{1}{\epsilon_r}\partial_{t_r}g^k, \quad k = 1, ..., n. \tag{28}$$

The scale separation assumption on the multiscale parameters implies that $\partial_t\Phi^k \approx \partial_{t_k}g^k = \frac{1}{\epsilon_k}\partial_{t_k}\Phi^k$, which allows us to simplify Eq.(27) into the following form

$$\frac{1}{\epsilon_n}\partial_{t_n}\Phi^n + \partial_x\Phi^n\left(\frac{1}{\epsilon_{n-1}}\partial_{t_{n-1}}\Phi^{n-1} + \partial_x\Phi^{n-1}\cdot\right.$$
$$\left.\left(\frac{1}{\epsilon_{n-2}}\partial_{t_{n-2}}\Phi^{n-2} + \cdots(\frac{1}{\epsilon_1}\partial_{t_1}\Phi^1 + \partial_x\Phi^1\dot{\tilde{y}}(t))\cdots\right)\right) = f^\epsilon. \tag{29}$$

Here $\tilde{y}(t)$ is an approximation of $y(t)$ since Eq.(29) is an approximation of the original Eq.(27) based on the scale separation assumption on the multiscale parameters. Finally, from Eq.(29) we can get the dynamic equation for $\tilde{y}(t)$, i.e.,

$$\dot{\tilde{y}} = (\partial_x\Phi^1)^{-1}\left(\cdots(\partial_x\Phi^{n-1})^{-1}\right.$$
$$\left.\left((\partial_x\Phi^n)^{-1}(f^\epsilon - \frac{1}{\epsilon_n}\partial_{t_n}\Phi^n) - \frac{1}{\epsilon_{n-1}}\partial_{t_{n-1}}\Phi^{n-1}\right) - \cdots - \frac{1}{\epsilon_1}\partial_{t_1}\Phi^1\right) =: F(t,x). \tag{30}$$

From the above derivation, one can see that the existence of the matrices $(\partial_x\Phi^k)^{-1}$, $k = 1, ..., n$ in (30) is essential in establishing the consistency of our method. In Section 3, we will prove that the invertibility is guaranteed in the case when $\epsilon_k$ are sufficient small; see (39).

6

In addition, we will prove that the ODE system (30) is non-stiff, which will be useful for the design of uniformly accurate numerical schemes, i.e., the time step in the numerical schemes does not depend on the multiscale parameters. When we obtain the solution $\tilde{y}(t)$ of the ODE system (30), we can recover the solution of the ODE system (1) through the transform $\bar{\Phi}^n$ defined in (14)(15). The error estimate of our method will be presented later.

*Remark* 2.1. From the explicit formulations for $\Phi^k$, $k = 1, ..., n$, we know the solution of the original ODE system can be rewritten as the following form

$$x(t) = \big(I_d + \mathcal{O}(\epsilon_n)\big) \circ \big(I_d + \mathcal{O}(\epsilon_{n-1})\big) \circ \cdots \circ \big(I_d + \mathcal{O}(\epsilon_1)\big)(y(t)). \tag{31}$$

Eq.(31) clearly reveals the structure of the transformation map in our method. One can see that the transformation map is a composition of simple maps, where each of them is a perturbation of identity. Interestingly, similar ideas appear in deep neural network research; see e.g. [2, 17], where approximations of functions via compositions of near-identity functions have been used intensively and are the key to the amazing expressibility power of a deep neuron network.

*Remark* 2.2. Our method can be extended to solve an ODE system (1), where $f^\epsilon$ is a quasi-periodic function. Assume that $f^\epsilon(t, x)$ in (3) has the form

$$f^\epsilon(t, x) = f\big(\frac{a_1(t)}{\epsilon_1}, \frac{a_2(t)}{\epsilon_2}, ..., \frac{a_n(t)}{\epsilon_n}, x\big), \tag{32}$$

where $a_k(t)$, $k = 1, ..., n$ are some invertible functions in $C^2$ such that $0 < c_0 \le \|\frac{d}{dt} a_k(t)\|_\infty \le c_1 < \infty$. Then the main results stated in this section for periodic functions still hold by using the same definition of the mean function defined in (4) without any prior knowledge of $a_k(t)$, $k = 1, ..., n$. The reason is that for any smooth function $h(x, y)$ that is periodic in $y$ with period 1, one can easily show that (see [6] for an elementary proof)

$$\left| \int_a^b h\big(\frac{a_{k-1}(t)}{\epsilon_{k-1}}, \frac{a_k(t)}{\epsilon_k}\big) dt - \int_a^b \big(\int_0^1 h\big(\frac{a_{k-1}(t)}{\epsilon_{k-1}}, y\big) dy\big) dt \right| \le C \frac{\epsilon_k}{\epsilon_{k-1}}, \tag{33}$$

by using a change of variable from $t$ to $s = a_k(t)$ and the fact that the Jacobian $J(s) = (\frac{d}{dt} a_k)^{-1}$ is a smooth function of $s$.

*Remark* 2.3. For a general ODE system $\dot{x} = f(t, x)$, where $f(t, x)$ does not have an explicit form of multiscale separation parametrization, we may reparameterize and approximate $f(t, x)$ by a formal multiscale velocity field $f^\epsilon(t, x)$. For instance, we may reparameterize $f(t, x)$ into a formal two-scale structure through Fourier transform; see [11]. The limitation is that we have to compute Fourier transform of $f(t, x)$ with respect to $t$, for each fixed $x$, which involves a certain amount of computation. To develop a fast solver to address this issue will be our future work.

### 2.3. Construction of the numerical schemes

In this section, we construct efficient numerical schemes to solve Eq.(30) that are uniformly accurate with respect to $\boldsymbol{\epsilon} = (\epsilon_1, ..., \epsilon_n)$. We first discuss how to accurately and efficiently

compute the maps $\Phi^k$, $k = 1, ..., n$ defined in Eqns.(19) and (25). We observe that the maps $\Phi^k$, $k = 1, ..., n$ are explicitly defined. Therefore we can use an explicit numerical scheme to approximate them. However, such an explicit implementation is not desirable because it may destroy the structures (e.g., Hamiltonian structure) of the original problem (1).

Alternatively, we adopt an implicit midpoint scheme to approximate $\Phi^k$, i.e.,

$$\Phi^k_{t_1,t_2,\cdots,t_k}(x) = x + \epsilon_k g^k(t_1, t_2, \cdots, t_k, \frac{x + \Phi^k_{t_1,t_2,\cdots,t_k}(x)}{2}), \quad k = 1, ..., n. \qquad (34)$$

The scheme (34) still provides an $\mathcal{O}(\epsilon_k)$ approximation of Eq.(25). In practice, $\Phi^k$ in (34) can be computed by the fixed point iteration. In addition, the derivatives of $\Phi^k$ with respect to $t_k$ or $x$ involved in Eq.(30) can be computed by the fixed point iteration based on the following identities,

$$\partial_{t_k} \Phi^k(x) = \epsilon_k f^k\left(\frac{x + \Phi^k(x)}{2}\right) + \frac{\epsilon_k}{2} \partial_x g^k\left(\frac{x + \Phi^k(x)}{2}\right) \partial_{t_k} \Phi^k(x), \qquad (35)$$

$$\left(\partial_x \Phi^k(x)\right)^{-1} K = K - \frac{\epsilon_k}{2} \partial_x g^k\left(\frac{x + \Phi^k(x)}{2}\right)\left(K + (\partial_x \Phi^k(x))^{-1} K\right), \qquad (36)$$

where $K$ is a $d$-dimensional column vector. The formulae in (34)-(36) suggests an iterative scheme to calculate all the quantities that are needed to compute (30) and $\bar{\Phi}^n$. Finally, we obtain an efficient numerical scheme to solve Eq.(30) at any time $t$ and value $x$.

The detailed implementation of the proposed numerical scheme is listed in Algorithm 1, in which we introduce several variables to simply the notations. Specifically, we have $P_k = \Phi^k \circ \Phi^{k-1} \circ \cdots \circ \Phi^1(y)$, $T_k = \frac{1}{\epsilon_k} \partial_{t_k} \Phi^k \circ \Phi^{k-1} \circ \cdots \circ \Phi^1(y)$, $k = 1, ..., n$, and $D_1 = \dot{y}$.

## 3. Convergence analysis

In this section, we present the convergence analysis of the proposed method. Since our goal is to develop numerical methods to solve ODE systems with a large range of $\epsilon$-values, the following assumption appears as a natural prerequisite.

**Assumption 3.1.** *Notice that our method developed in Section 2 is a first-order method (w.r.t. $\epsilon$). We require $f^\epsilon$ and its fluctuation components $f^k$, $k = 1, ..., n$ are second-order differentiable and are bounded on some closed set $\mathbb{T}^n \times \mathcal{K}$, where $\mathcal{K} \subset \mathbb{R}^d$. In addition, we assume that the path of the solution $x(t)$ is in $\mathcal{K}$.*

*Remark* 3.1. In many cases, $f^\epsilon$ and $f^k$, $k = 1, ..., n$ are globally defined, which requires $\mathcal{K} = \mathbb{R}^d$.

First, we prove that the transformed equation (30) is non-stiff with respect to $\epsilon$, and thus it can be solved by using conventional numerical methods with relatively large time steps.

**Theorem 3.2.** *Suppose that Assumption 3.1 is satisfied and $0 < \epsilon_k < 1$, $k = 1, ..., n$ are sufficiently small. Let $F(t, x)$ denote the right hand side of the ODE system (30). Then, we have the following estimate,*

$$\left|\partial_t F(t, x)\right| \leq C_0, \qquad (37)$$

*where $\left|\cdot\right|$ is a vector norm and $C_0$ does not depend on $\epsilon_k$, $k = 1, ..., n$.*

---

**Algorithm 1** A fixed point iteration method to compute the ODE with $n$ time-scales.

1: Set $i = 0$, $P_1^{[0]} = P_2^{[0]} = \cdots = P_n^{[0]} = y$
2: **repeat**
3:     $P_0^{[i]} = y$
4:     **for** $k = 1$ to $n$ **do**
5:         $R_k^{[i]} = \frac{P_{k-1}^{[i]} + P_k^{[i]}}{2}$
6:         $P_k^{[i+1]} = P_{k-1}^{[i]} + \epsilon_k g^k(R_k^{[i]})$
7:     **end for**
8:     $i \to i + 1$
9: **until** $P_n^{[i]}$ converges.
10: Set $j = 0$, $T_1^{[0]} = T_2^{[0]} = \cdots = T_n^{[0]} = D_1^{[0]} = D_2^{[0]} = \cdots = D_n^{[0]} = 0$
11: **repeat**
12:     $D_{n+1}^{[j]} = f^\epsilon(P_n^{[i]})$
13:     **for** $k = n$ to $1$ **do**
14:         $T_k^{[j+1]} = f^k(R_k^{[i]}) + \frac{\epsilon_k}{2}\partial_x g^k(R_k^{[i]})T_k^{[j]}$
15:         $B_k^{[j]} = D_{k+1}^{[j]} - T_k^{[j+1]}$
16:         $D_k^{[j+1]} = B_k^{[j]} - \frac{\epsilon_k}{2}\partial_x g^k(R_k^{[i]})(B_k^{[j]} + D_k^{[j]})$
17:     **end for**
18:     $j \to j + 1$
19: **until** $D_1^{[j]}$ converges.

---

*Proof.* According to the definitions (25), we have the results,

$$\partial_x \Phi^k = I_d + \epsilon_k \partial_x g^k, \quad k = 1, ..., n. \tag{38}$$

When $\epsilon_k$ are sufficiently small, the inverse of $\partial_x \Phi^k$ exists and can be computed through the Neumann series expansion,

$$(\partial_x \Phi^k)^{-1} = I_d + \sum_{m=1}^{\infty} (-\epsilon_k \partial_x g^k)^m. \tag{39}$$

Taking the derivative of Eq.(39) on both sides with respect to $t$, we obtain

$$\partial_t (\partial_x \Phi^k)^{-1} = \sum_{m=1}^{\infty} \partial_t (-\epsilon_k \partial_x g^k)^m. \tag{40}$$

Moreover, we have the estimates,

$$\left|\left|(\partial_x \Phi^k)^{-1} - I_d\right|\right| \le C\epsilon_k, \tag{41}$$

$$\left|\left|\partial_t (\partial_x \Phi^k)^{-1}\right|\right| = \left|\left|\sum_{m=1}^{\infty} \partial_t (-\epsilon_k \partial_x g^k)^m\right|\right| \le C, \tag{42}$$

9

where $||\cdot||$ is a matrix norm. At the same time, we have the condition

$$\frac{1}{\epsilon_k}\partial_{t_k}\Phi^k = \partial_{t_k}g^k = f^k. \tag{43}$$

Therefore, the right hand side of the ODE system (30) can be re-written as,

$$
\begin{aligned}
F(t,x) &= \prod_{i=1}^n(\partial_x\Phi^k)^{-1}f^\epsilon - \sum_{k=1}^n\prod_{i=1}^k(\partial_x\Phi^i)^{-1}\frac{1}{\epsilon_k}\partial_{t_k}\Phi^k, \\
&= \prod_{i=1}^n(\partial_x\Phi^k)^{-1}\bar{f}^1 + \sum_{k=1}^{n-1}\prod_{i=1}^k(\partial_x\Phi^i)^{-1}\Big(\prod_{i=k+1}^n(\partial_x\Phi^i)^{-1} - I_d\Big)f^k, \\
&\equiv J_0 + \sum_{k=1}^{n-1}J_k. 
\end{aligned}
\tag{44}
$$

Taking derivative of $F(t,x)$ with respect to $t$ and using the product rule, we can easily verify that the terms $\partial_t J_0$ and $\partial_t J_k$, $k = 1, ..., n-1$ are all $\mathcal{O}(1)$. Thus, the assertion in 37 is proved. $\qquad\square$

Theorem 3.2 shows that the transformed ODE system (30) is non-stiff, which is then amenable to a standard numerical treatment. As such, we divide the time interval $[0, 1]$ by the nodes $t_m = m\Delta t$, $m = 0, ..., M$, where $\Delta t = 1/M$ is the time step and $M$ is a positive integer. For each $m$, $m = 1, ..., M$, we seek a numerical solution $\hat{y}(t_m)$ to approximate $\tilde{y}(t_m)$, which is the value of the exact solution of the ODE system (30) at time $t_m$.

Here, we use the implicit integral midpoint scheme $(Im2nd)$ to solve the ODE system (30). Between two consecutive computational times $t_m$ and $t_{m+1}$, we integrate the differential equation (30) and obtain,

$$\tilde{y}(t_{m+1}) = \tilde{y}(t_m) + \int_{t_m}^{t_{m+1}} F(s, \tilde{y}(s))ds. \tag{45}$$

Then, we approximate $\tilde{y}(s)$ by an average value and arrive at,

$$\hat{y}(t_{m+1}) = \hat{y}(t_m) + \int_{t_m}^{t_{m+1}} F\Big(s, \frac{\hat{y}(t_{m+1}) + \hat{y}(t_m)}{2}\Big)ds. \tag{46}$$

We remark that the numerical solution $\hat{y}(t_{m+1})$ can be computed by some iteration methods, such as the Newton-Raphson method or fixed point iteration method. In this paper, we choose the scheme (46) since it preserves certain intrinsic structures in the solution of the original problem; see Section 4 for more discussions.

The uniform boundedness of the first-order derivative of $F(t,x)$ (proved in Theorem 3.2) guarantees that our implicit integral midpoint scheme (46) has second-order accuracy. Furthermore, we do not need to decrease the time step $\Delta t$ when $\epsilon_k$, $k = 1, ..., n$ are small. We summarize the property of the numerical solution $\hat{y}(t_m)$ into the following lemma.

**Lemma 3.3.** *Let $\tilde{y}(t)$ be the exact solution of the transformed ODE system* (30). *And let $\hat{y}(t_m)$, $m = 1, ..., M$ be the numerical solutions obtained by the scheme* (46). *Then, we have*

$$\left| \hat{y}(t_m) - \tilde{y}(t_m) \right| = C_1(\Delta t)^2, \quad m = 1, ..., M, \tag{47}$$

*where $C_1$ does not depend on $\epsilon_k$, $k = 1, ..., n$.*

Finally, we analyze the error between the approximated solution $\bar{\Phi}^n(\hat{y})$ and the exact solution $x(t)$ of the original ODE system (1).

**Theorem 3.4.** *Let $T$ denote the final computational time. Suppose Assumption 3.1 is satisfied and $0 < \epsilon_k < 1$, $k = 1, ..., n$ are sufficiently small. For all $t \leq T$, we have the following error estimate*

$$\left| x(t) - \bar{\Phi}_t^n(\hat{y}(t)) \right| \leq C_2 \left( \max_{i=2,\cdots,n} \frac{\epsilon_i}{\epsilon_{i-1}} \right) + C_3(\Delta t)^2, \tag{48}$$

*where $C_2$ and $C_3$ are generic constants that do not depend on $\epsilon_k$, $k = 1, ..., n$ and $\Delta t$.*

*Proof.* For any given computational time $t$, we have

$$\left| x(t) - \bar{\Phi}_t^n(\hat{y}(t)) \right| \leq \left| \bar{\Phi}_t^n(y(t)) - \bar{\Phi}_t^n(\tilde{y}(t)) \right| + \left| \bar{\Phi}_t^n(\tilde{y}(t)) - \bar{\Phi}_t^n(\hat{y}(t)) \right|, \tag{49}$$

where $y(t)$ and $\tilde{y}(t)$ are the exact solutions of the ODE systems (27) and (30), respectively, and $\hat{y}(t)$ is the numerical approximation of $\tilde{y}(t)$. We shall estimate the two terms in (49) separately. First we can see that,

$$
\begin{aligned}
\dot{y} - \dot{\tilde{y}} = & \Big( \prod_{i=1}^n (\partial_x \Phi^k)^{-1} f^\epsilon - \sum_{k=1}^n \prod_{i=1}^k (\partial_x \Phi^i)^{-1} \sum_{j=1}^k \frac{\epsilon_k}{\epsilon_j} \partial_{t_j} g^k \Big) \\
& - \Big( \prod_{i=1}^n (\partial_x \Phi^k)^{-1} f^\epsilon - \sum_{k=1}^n \prod_{i=1}^k (\partial_x \Phi^i)^{-1} \partial_{t_k} g^k \Phi^k \Big), \\
= & - \sum_{k=1}^n \prod_{i=1}^k (\partial_x \Phi^i)^{-1} \sum_{j=1}^{k-1} \frac{\epsilon_k}{\epsilon_j} \partial_{t_j} g^k.
\end{aligned}
\tag{50}
$$

Using the conditions that $\partial_{t_k} g^k$ are bounded functions (see Eq.(43)) and $\left\| (\partial_x \Phi^k)^{-1} - I_d \right\| \leq C\epsilon_k$ (see Eq.(41)), we have

$$\left| \dot{y} - \dot{\tilde{y}} \right| \leq C \left( \max_{i=2,\cdots,n} \frac{\epsilon_i}{\epsilon_{i-1}} \right). \tag{51}$$

Hence for any time $t \leq T$, we have,

$$\left| y(t) - \tilde{y}(t) \right| \leq C_T \left( \max_{i=2,\cdots,n} \frac{\epsilon_i}{\epsilon_{i-1}} \right), \tag{52}$$

where the constant $C_T = \mathcal{O}(T)$. Applying the chain rule for $\bar{\Phi}^n$, we obtain

$$\partial_x \bar{\Phi}^n = \partial_x \big( \Phi^n \circ \Phi^{n-1} \circ \cdots \circ \Phi^1 \big) = \prod_{i=1}^n (I_d + \epsilon_i \partial_x g^i). \tag{53}$$

So at any time $t$, $\bar{\Phi}^n$ is Lipschitz in the variable $x$ and the Lipschitz constant is uniformly bounded when $\epsilon_i$ are small enough. Finally, combining the estimates (47), (52) and (53), we prove the statement in Theorem 3.4. □

When the multiscale parameters are well-separated; see (2), the first term in the error estimate (48) is negligible, thus our scheme has a second-order accuracy with respect to $\Delta t$. Although the above convergence analysis relies on the scale separation assumption, we can relax this assumption in some special cases. For example, when two scales collapse, i.e. $\epsilon_n = c\epsilon_{n-1}$, we can treat these two scales as a single scale and we can modify the mean function defined in (4) accordingly. More specifically, if $c = \frac{m_1}{m_2}$ is a rational number, and $h(y_1, y_2)$ is a doubly periodic function in $y_1$ and $y_2$ with period $[1, 1]$, then $h(\frac{t}{\epsilon_{n-1}}, \frac{t}{\epsilon_n}) = h(\frac{t}{\epsilon_{n-1}}, \frac{m_2 t}{m_1 \epsilon_{n-1}})$ is a periodic function of $t$ with a period $m_1 \epsilon_{n-1}$. Thus, the mean function defined in (4) can be modified accordingly as follows

$$\frac{1}{m_1} \int_0^{m_1} f(t_1, t_2, ..., t_{n-2}, s, \frac{m_2}{m_1}s, x)ds. \tag{54}$$

When $c$ is an irrational number, it is easy to show that the time average of $h(\frac{t}{\epsilon_{n-1}}, \frac{t}{\epsilon_n})$ will converge to the area in $(y_1, y_2) \in \mathbb{T}^2$ and we can modify the definition of the mean function as follows

$$\int_{[0,1]^2} f(t_1, t_2, ..., t_{n-2}, s_1, s_2, x)ds_1 ds_2. \tag{55}$$

With the above modification of the mean function, we can still prove the main results stated in this section. Our numerical results to be presented later also confirm that our method works equally well in the case when two scales collapse.

More general case can be considered as well if we have $m$ number of collapsed scales, i.e. $\epsilon_k = c_1\epsilon_{k+1} = c_2\epsilon_{k+2} = ... = c_{m-1}\epsilon_{k+m-1}$. In this case, we should consider these m-scales simultaneously and modify the definition of the mean function by using the time averaging technique for multiple scales discussed in [10]. We will not present the more general case in this paper and will leave it to our future work.

## 4. Numerical results

In this section, we present several numerical experiments to illustrate the efficiency of our method and confirm the convergence analysis. The *Im2nd* solution, or direct *Im2nd* solution, refers to the numerical solution obtained by solving the original problem (1) using the scheme (46). Moreover, we use the *UA* solution to denote the numerical solution obtained by our method. In our method, we use the scheme (46) to solve Eq.(30) and the Algorithm 1 to compute necessary quantities in the Eq.(30).

### 4.1. An ODE system with three separated time scales

The Hénon-Heiles system [9] is undoubtedly one of the most paradigmatic model potentials for time-independent Hamiltonian systems with two degrees of freedom, which is frequently used to describe the motion of stars around a galactic center. We consider a generalization

of the original Hénon-Heiles system in three degrees of freedom. The relative equilibria and bifurcations of this model were studied in [7].

Here, we assume the Hamiltonian of the three dimensional Hénon-Heiles system is parameterized by $\epsilon_1$ and $\epsilon_2$ and has the following form

$$H(\mathbf{p}, \mathbf{q}) = \frac{p_1^2}{2\epsilon_2} + \frac{q_1^2}{2\epsilon_2} + \frac{p_2^2}{2\epsilon_1} + \frac{q_2^2}{2\epsilon_1} + \frac{p_3^2}{2} + \frac{q_3^2}{2} + q_1^2 q_2 - \frac{1}{3}q_2^3 + q_2^2 q_3 - \frac{1}{3}q_3^3. \tag{56}$$

where $\mathbf{p} = (p_1, p_2, p_3)^T$ and $\mathbf{q} = (q_1, q_2, q_3)^T$. One can obtain the evolution equation for the Hamiltonian in (56) as follows,

$$\frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{q}}, \qquad \frac{d\mathbf{q}}{dt} = \frac{\partial H}{\partial \mathbf{p}}. \tag{57}$$

When $0 < \epsilon_2 \ll \epsilon_1 \ll 1$, the ODE system (57) becomes a three-scale problem and the solutions (e.g., $p_1$ and $q_1$) are highly oscillatory. Let us carry out a change of variables,

$$\begin{cases} w_1 = \cos(\frac{t}{\epsilon_2})q_1 - \sin(\frac{t}{\epsilon_2})p_1, \\ w_2 = \sin(\frac{t}{\epsilon_2})q_1 + \cos(\frac{t}{\epsilon_2})p_1, \\ w_3 = \cos(\frac{t}{\epsilon_1})q_2 - \sin(\frac{t}{\epsilon_1})p_2, \\ w_4 = \sin(\frac{t}{\epsilon_1})q_2 + \cos(\frac{t}{\epsilon_1})p_2, \\ w_5 = q_3, \\ w_6 = p_3. \end{cases} \tag{58}$$

Then, $(w_1, ..., w_6)^T$ satisfies the following ODE system

$$\begin{cases} \dot{w}_1 = 2\sin t_2 (w_1 \cos t_2 + w_2 \sin t_2)(w_3 \cos t_1 + w_4 \sin t_1), \\ \dot{w}_2 = -2\cos t_2 (w_1 \cos t_2 + w_2 \sin t_2)(w_3 \cos t_1 + w_4 \sin t_1), \\ \dot{w}_3 = \sin t_1 \big(2(w_3 \cos t_1 + w_4 \sin t_1)w_5 + (w_1 \cos t_2 + w_2 \sin t_2)^2 - (w_3 \cos t_1 + w_4 \sin t_1)^2\big), \\ \dot{w}_4 = -\cos t_1 \big(2(w_3 \cos t_1 + w_4 \sin t_1)w_5 + (w_1 \cos t_2 + w_2 \sin t_2)^2 - (w_3 \cos t_1 + w_4 \sin t_1)^2\big), \\ \dot{w}_5 = w_6, \\ \dot{w}_6 = w_5^2 - w_5 - (w_3 \cos t_1 + w_4 \sin t_1)^2, \end{cases} \tag{59}$$

where $t_2 = \frac{t}{\epsilon_2}$ and $t_1 = \frac{t}{\epsilon_1}$. Notice that the right-hand side of the ODE system (59) involves trigonometric functions and simple polynomials. Thus, all the integrals in our numerical schemes can be pre-computed analytically. We have implemented these computations with the software Mathematica.

**Verification of the convergence analysis.** We compare the error between the numerical solution obtained by our method and the reference solution. The initial value is $(w_1(0), ..., w_6(0))^T = (0.12, 0.12, 0.12, 0.12, 0.12, 0.12)^T$. The reference solution is obtained by Matlab `ode45` function applied to the ODE system (59), where the time step is $\Delta t = 0.0001$. To implement our method, we choose the second-order implicit integral midpoint scheme to

integrate the non-stiff problem (30). We find that the iteration loops needed in Algorithm 1 is about $3 - 10$ times, where the convergence threshold is set to be $10^{-14}$.

In Fig.1(a) and Fig.1(b), we show the error as a function of $\Delta t$ for different values of $\epsilon_1$ with $\epsilon_2 = \epsilon_1^2$ and $\epsilon_2 = 0.8\epsilon_1$, respectively. The magnitude of the Hamiltonian (56) is about $\frac{0.0144}{\epsilon_1^2}$. We observe a second-order convergence rate with respect to $\Delta t$ in our method. Most importantly, the error is independent of $\epsilon_1$ and $\epsilon_2$ as shown in Fig.1(a) and Fig.1(b), where the curves for different values of $\epsilon_1$ and $\epsilon_2$ are nearly identical. This confirms that our scheme is uniform accurate, which does not depend on $\epsilon_1$ and $\epsilon_2$. Notice that the numerical solution obtained by $\Delta t = 0.1$ is accurate enough to maintain an error of the order $10^{-4}$ and error of the Hamiltonian at the order of $10^{-3}$, uniformly in $\epsilon_1$ and $\epsilon_2$. For the Euler method, it is impossible since the ODE system associated with the Hamiltonian (56) becomes severely stiff when $\epsilon_1$ and $\epsilon_2$ become small. The numerical results for the Euler method were not shown here.
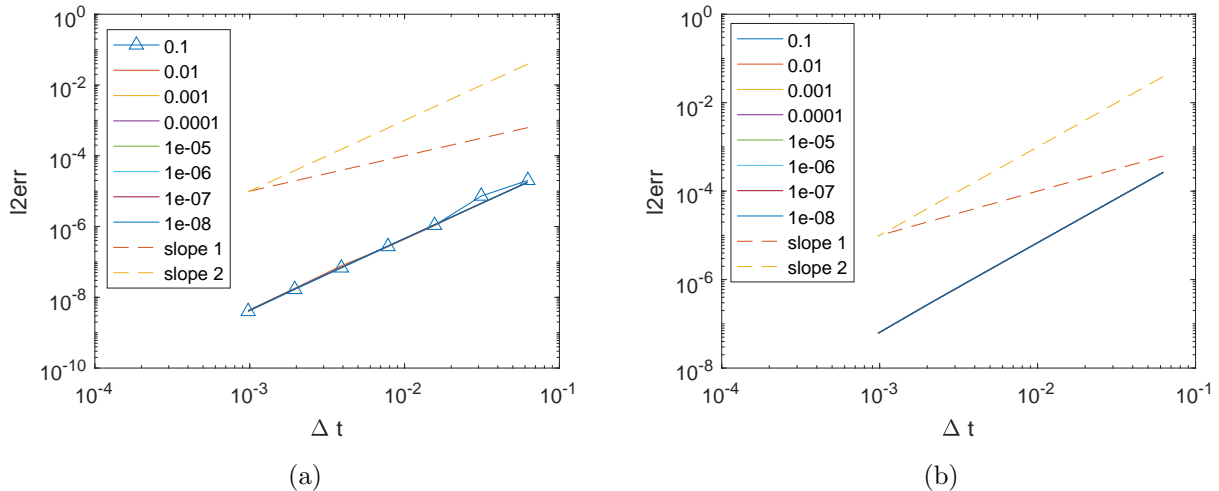


Figure 1: Error as a function of $\Delta t$ for $\epsilon_1 = 10^{-k}$, $k = 1, ..., 8$. Left: $\epsilon_2 = \epsilon_1^2$. Right: $\epsilon_2 = 0.8\epsilon_1$.

In the derivation of the numerical method and the convergence analysis, we assume the time-scales are well-separated, i.e., $0 < \epsilon_n \ll \epsilon_{n-1} \ll \cdots \cdots \ll \epsilon_1 \ll 1$. In Fig.1(b), we show the error as a function of $\Delta t$ for difference values of $\epsilon_1$ and $\epsilon_2 = 0.8\epsilon_1$ at $T = 1$. It is shown that our scheme still has a uniform accuracy, which does not depend on separation between $\epsilon_1$ and $\epsilon_2$.

Let us now verify that our method preserves the Hamiltonian of the system. In Fig.2, we plot the evolution of the error of the Hamiltonian (56) for different $\epsilon_1$ and $\epsilon_2$, where $\epsilon_2 = \epsilon_1^2$. We find that when $\epsilon_1$ is relatively large, e.g., $\epsilon_1 = 0.1$, the implicit integral midpoint scheme (46) with a large time step $\Delta t = 0.1$ and our method give similar and accurate results in computing the Hamiltonian; see Fig.2(a). However, when $\epsilon_1$ is small, e.g., $\epsilon_1 = 0.001$, the ODE system associated with the Hamiltonian (56) becomes very stiff. Directly using the scheme (46) will lose accuracy if the time step is not small enough. While our method with a large time step still maintains the same accuracy; see Fig.2(b). This result again confirms that

our scheme has a uniform accuracy in computing an ODE system with multiple time-scales, especially when the systems are stiff.
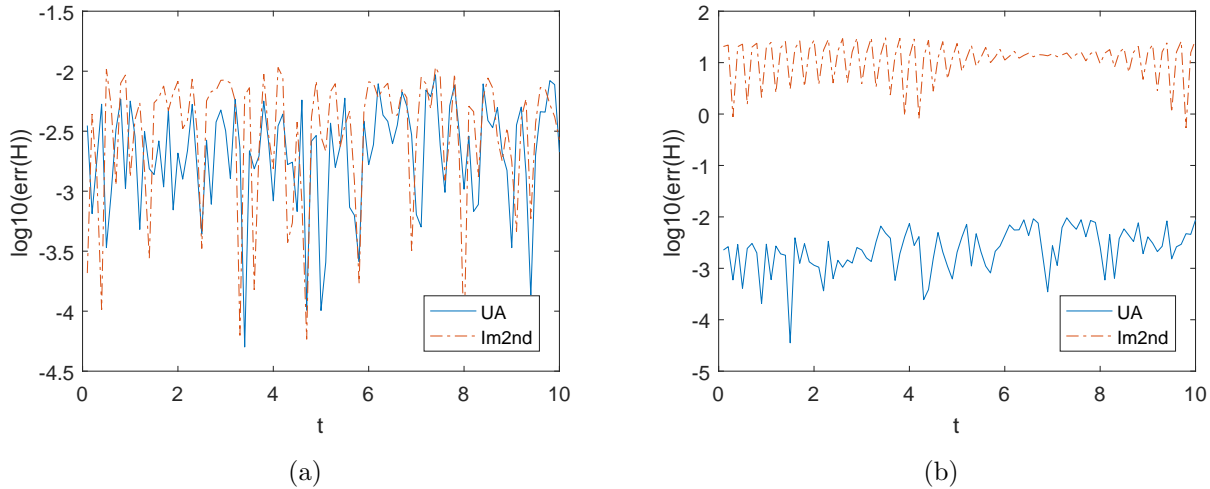


Figure 2: Evolution of the error of the Hamiltonian in the three-scale ODE system with $\epsilon_2 = \epsilon_1^2$. Left: $\epsilon_1 = 0.1$. Right: $\epsilon_1 = 0.001$. $\Delta t = 0.1$.

**Verification of the non-stiffness of the transformed equation** (30). In Theorem 3.2, we proved that under certain assumptions the transformed ODE system (30) is non-stiff. Here, we shall verify this statement numerically by solving the three-scale ODE system (59) with an initial value $(w_1(0), ..., w_6(0))^T = (0.12, 0.12, 0.12, 0.12, 0.12, 0.12)^T$. We compute the maps $\Phi^1$ and $\Phi^2$ in our method with different $t_1$ and $t_2$ for $(y_1(0), ..., y_6(0))^T = (0.20, 0.20, 0.20, 0.20, 0.20, 0.20)^T$. In addition, we record the quantities $P_2 = \Phi_{t_2}^2(\Phi_{t_1}^1(y))$, $T_1 = \frac{1}{\epsilon_1}\partial_{t_1}\Phi_{t_1}^1(y)$, $T_2 = \frac{1}{\epsilon_2}\partial_{t_2}\Phi_{t_2}^2(\Phi_{t_1}^1(y))$, and $D_1 = \dot{y}$. Recall that these quantities were defined in Section 2.3, especially in the Algorithm 1.

In Fig.3, we show the six components of the quantities $f(y) - D_1$ as functions of $t_1$ and $t_2$ (i.e., the six components of right hand side of the ODE system (59) minus their numerical counterparts), where $\epsilon_1 = 0.0001$ and $\epsilon_2 = \epsilon_1^2$. In this example, $\max|f(y)|$ is $\mathcal{O}(1)$. We also compute the cases when $\epsilon_1 = 0.01$ and $\epsilon_1 = 0.001$ with $\epsilon_2 = \epsilon_1^2$ and find the patterns of the six components of the quantities $\dot{y} - D_1$ remain almost the same as the Fig.3. Thus, we do not show them here.

In Fig.4, we show the magnitude of the quantity $\Phi_{t_2}^2(\Phi_{t_1}^1(y)) - y$ as functions of $t_1$ and $t_2$ when $\epsilon_1 = 0.01$ and $\epsilon_2 = \epsilon_1^2$. The results for $\Phi_{t_2}^2(\Phi_{t_1}^1(y)) - y$ when $\epsilon_1 = 0.01$ and $\epsilon_2 = \frac{\epsilon_1}{2}$ are shown in Fig.5. One can see that when there is no scale separation $\Phi_{t_2}^2(\Phi_{t_1}^1(y))$ are fluctuating along $t_2$ direction. This result provides numerical confirmation of our derivation. We can write $\Phi_{t_2}^2(\Phi_{t_1}^1(y))$ explicitly out as,

$$\Phi_{t_2}^2(\Phi_{t_1}^1(y)) = \Phi_{t_1}^1(y) + \epsilon_2 g^2(t_2, t_1, \Phi_{t_1}^1(y)) = y + \epsilon_1 g^1(t_1, y) + \epsilon_2 g^2(t_2, t_1, y + \epsilon_1 g^1(t_1, y)).$$
(60)

Then given $y$, when $\epsilon_1$ and $\epsilon_2$ are close, $\epsilon_2 g^2(t_2, t_1, y + \epsilon_1 g^1(t_1, y))$ is comparable to $\epsilon_1 g^1(t_1, y)$.
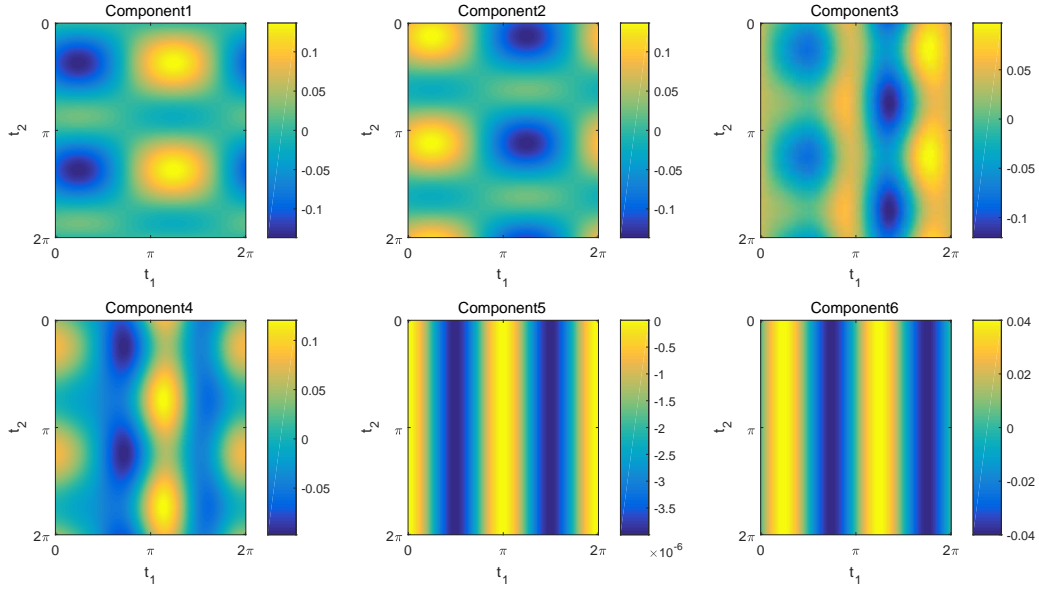
15

Figure 3: Six components of $f(t_1, t_2, y) - D_1$. Here $\epsilon_1 = 0.0001$ and $\epsilon_2 = \epsilon_1^2$.
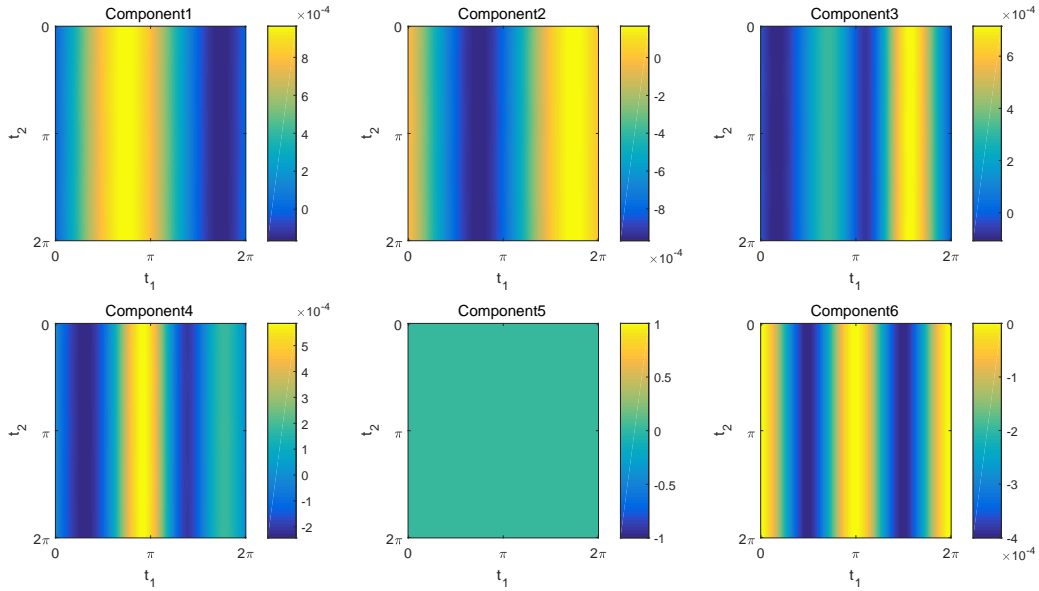


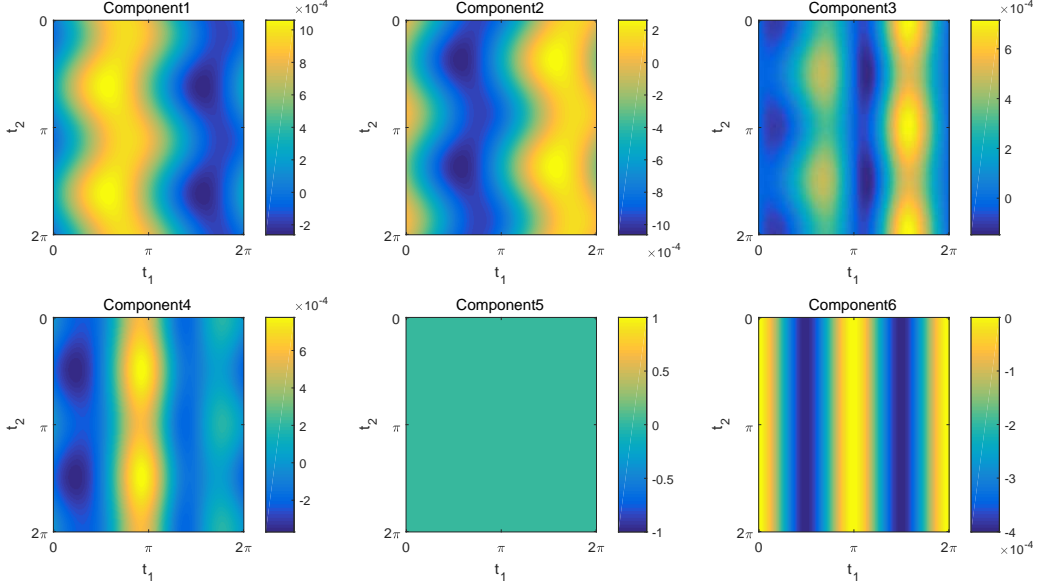Figure 4: $\Phi_{t_2}^2(\Phi_{t_1}^1(y)) - y$ when $\epsilon_1 = 0.01$ and $\epsilon_2 = \epsilon_1^2$.

Figure 5: $\Phi^2_{t_2}(\Phi^1_{t_1}(y)) - y$ when $\epsilon_1 = 0.01$ and $\epsilon_2 = \frac{\epsilon_1}{2}$.

In Fig.6 and Fig.7, we show the magnitude of the quantities $T_1$ and $T_2$ as functions of $t_1$ and $t_2$, when $\epsilon_1 = 0.0001$ and $\epsilon_2 = \epsilon_1^2$, respectively. We also compute the quantities $T_1$ and $T_2$ when $\epsilon_1$ changes (e.g. $\epsilon_1 = 0.01$ and $\epsilon_1 = 0.001$) with $\epsilon_2 = \epsilon_1^2$ and find the time derivatives of $T_1$ and $T_2$ remain almost the same magnitude as that shown in Fig.6 and Fig.7. This implies the implicit iterative scheme Eq.(35) for Eq.(34) keeps the magnitude of the non-midpoint setting, $\partial_t \Phi^k = \sum_{i=1}^k \frac{\epsilon_k}{\epsilon_i} \partial_{t_i} g^k$. Notice that component 5 and 6 do not depend on $t_2$. This is due to the fact that $f_5$ and $f_6$ are independent of $t_2$. $T_1$ is independent of $t_2$ for difference choice of $\epsilon_1$ and $\epsilon_2$, this is due to the definition of $T_1$; see Fig.6.

*4.2. An ODE system with four separated time scales*

To further study the performance of our method, we mimic the formulation of the three dimensional Hénon-Heiles system and generate a Hamiltonian system with four time scales. The Hamiltonian is given by,

$$
\begin{aligned}
H(\mathbf{q}, \mathbf{p}) =& \frac{p_1^2}{2\epsilon_3} + \frac{q_1^2}{2\epsilon_3} + \frac{p_2^2}{2\epsilon_2} + \frac{q_2^2}{2\epsilon_2} + \frac{p_3^2}{2\epsilon_1} + \frac{q_3^2}{2\epsilon_1} + \frac{p_4^2}{2} + \frac{q_4^2}{2} \\
&+ q_1^2 q_2 + q_2^2 q_3 + q_3^2 q_4 - \frac{1}{3} q_2^3 - \frac{1}{3} q_3^3 - \frac{1}{3} q_4^3,
\end{aligned}
\tag{61}
$$

where $\mathbf{p} = (p_1, p_2, p_3, p_4)^T$ and $\mathbf{q} = (q_1, q_2, q_3, q_4)^T$. The Hamiltonian in (61) is parameterized by $\epsilon_1$, $\epsilon_2$, and $\epsilon_3$. One can obtain the evolution equation for the Hamiltonian in (61) by using the relation $\frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{q}}$ and $\frac{d\mathbf{q}}{dt} = \frac{\partial H}{\partial \mathbf{p}}$. When $0 < \epsilon_3 \ll \epsilon_2 \ll \epsilon_1 \ll 1$, the associated evolution equation of the Hamiltonian in (61) becomes a four-scale ODE system. Since the formulation of the change of variables and derivation of the transformed ODE system are standard (similar as we did in Eqns.(58) and (59)), we do not show them here.
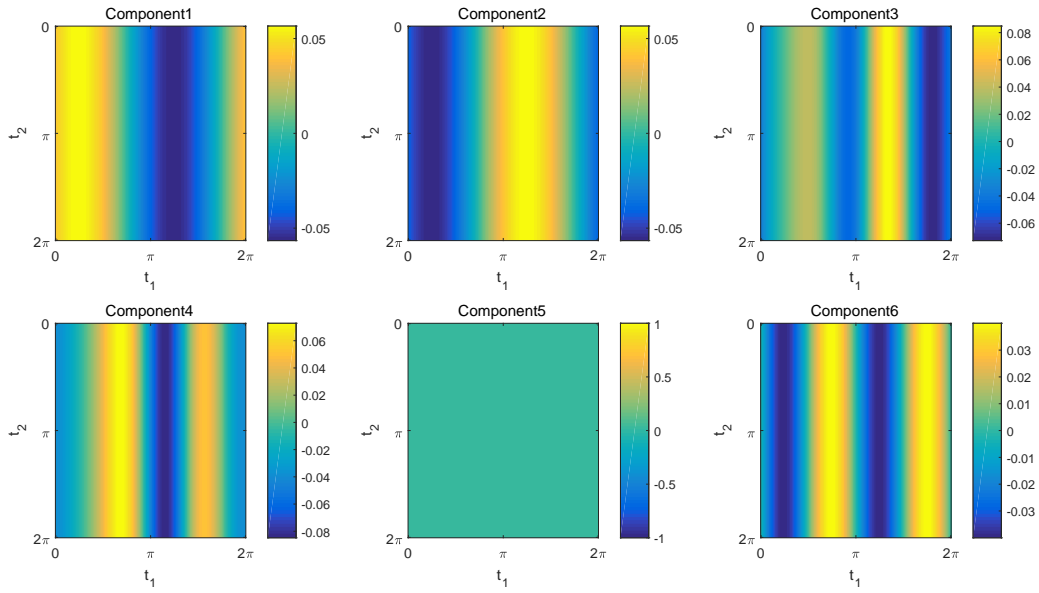
17

Figure 6: $T_1$ when $\epsilon_1 = 0.0001$ and $\epsilon_2 = \epsilon_1^2$.
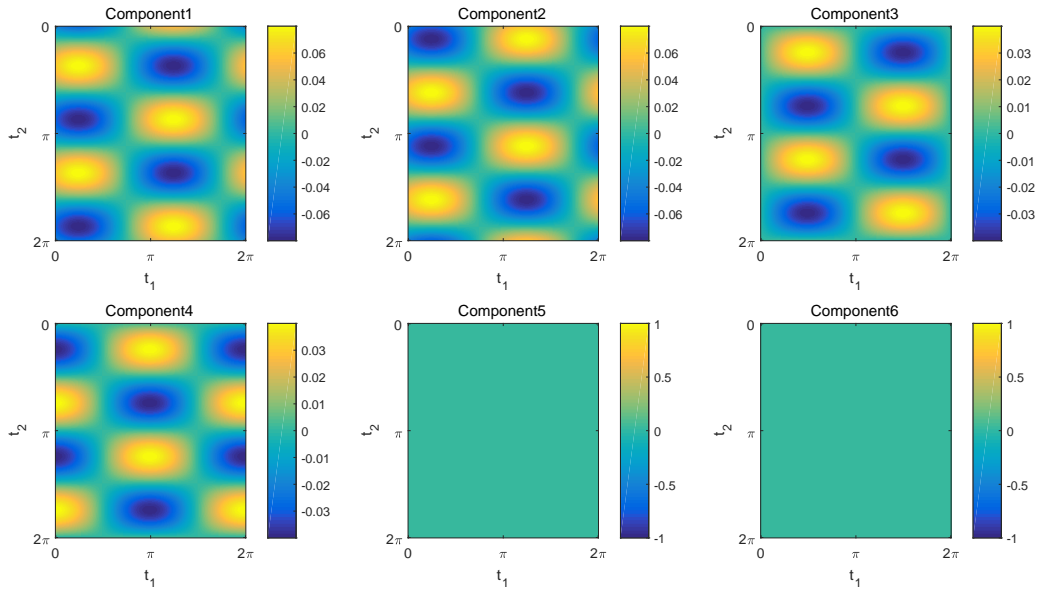


Figure 7: $T_2$ when $\epsilon_1 = 0.0001$ and $\epsilon_2 = \epsilon_1^2$.

18

**Verification of the convergence analysis.** Let us now check that our method preserves the Hamiltonian of the system. In Fig.8, we present the error of the Hamiltonian (61) as a function of time for different $\epsilon_i$, $i = 1, 2, 3$. In Fig.8(a), we set the multiscale time-scale parameters to be $(\epsilon_1, \epsilon_2, \epsilon_3) = (10^{-3}, 11 \times 10^{-5}, 3 \times 10^{-6})$ and the initial values are $w_i(0) = 0.44$ for $i = 1, \cdots, 8$. In Fig.8(b), we set $(\epsilon_1, \epsilon_2, \epsilon_3) = (0.5, 11 \times 10^{-6}, 3 \times 10^{-6})$ and the initial values are the same. Notice that the smallest period of the ODE system is about $2\pi \times \epsilon_3 \approx 1.885 \times 10^{-5}$. To resolve the oscillation in the solution, we choose the time step for the fine-scale ODE solver to be $\Delta t = 5 \times 10^{-6}$. To implement our method, we choose the time step to be $\Delta t = 10^{-1}$.

Numerical results in Fig.8 show that: (1) directly using the implicit integral midpoint scheme (46) with a coarse time step $\Delta t = 10^{-1}$ gives wrong results; (2) our method with the same coarse time step $\Delta t = 10^{-1}$ gives an accurate result that is comparable to that using the scheme (46) with a very fine time step $\Delta t = 10^{-6}$. This comparison again confirms that our scheme has a uniform accuracy in computing ODE system with multiple time-scales. In addition, from the results in Fig.8(b), where $(\epsilon_1, \epsilon_2, \epsilon_3) = (0.5, 11 \times 10^{-6}, 3 \times 10^{-6})$, we can see that when the time scales are not well separated ( i.e., $\epsilon_3$ is close to $\epsilon_2$ and $\epsilon_1$ is close to 1), our numerical method still gives an excellent performance.

In terms of the computational time, our method takes 0.163 and 0.367 seconds to compute the result shown in Fig.8(a) and Fig.8(b), respectively. While the direct *Im2nd* method with $\Delta t = 5 \times 10^{-6}$ takes about 10.48 seconds for both. Thus, our method achieves a $20 \sim 60X$ speedup over the conventional ODE solver in this example. Moreover, our method provides uniform accurate results for different values of $\epsilon_i$, $i = 1, 2, 3$. The conventional ODE solvers, such as the direct *Im2nd* method, require to choose finer time steps when we decrease $\epsilon_i$, $i = 1, 2, 3$. Therefore, it is expected that a higher speedup will be achieved when we need to solve a multiscale ODE system with much smaller $\epsilon_i$, $i = 1, 2, 3$.
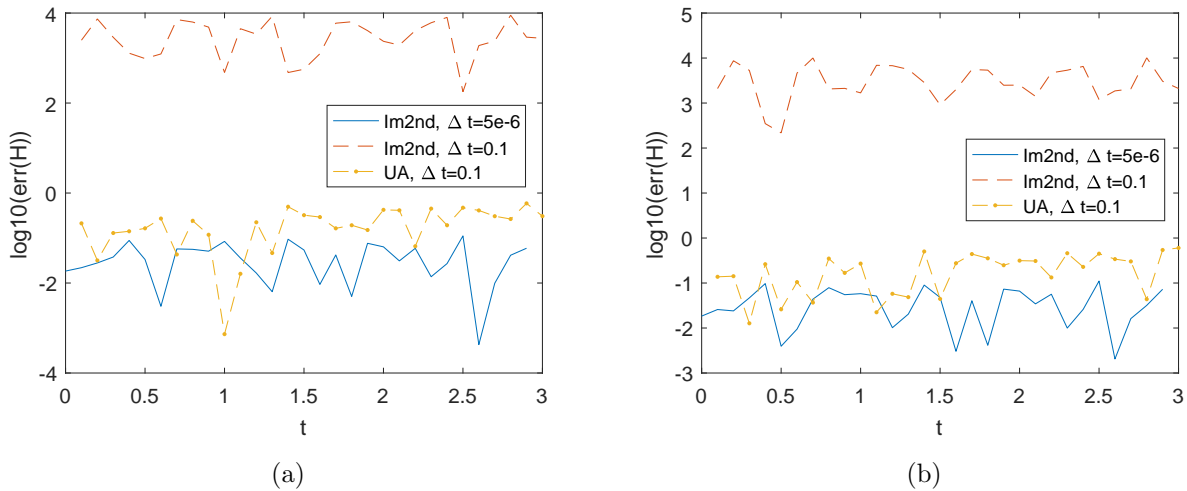


Figure 8: Evolution of the error of the Hamiltonian in the four-scale ODE system. Left: $(\epsilon_1, \epsilon_2, \epsilon_3) = (10^{-3}, 11 \times 10^{-5}, 3 \times 10^{-6})$. Right: $(\epsilon_1, \epsilon_2, \epsilon_3) = (0.5, 11 \times 10^{-6}, 3 \times 10^{-6})$.

We also investigate the convergence rate of our method with respect to the time step. In Fig.9, we show the error as a function of $\Delta t$ for two sets of values of $\epsilon_1$, $\epsilon_2$, and $\epsilon_3$ at time $T = 3$, respectively. The intial values $x_i(0)$, $i = 1, ..., 8$ and values of $(\epsilon_1, \epsilon_2, \epsilon_3)$ are the same as before. The reference solution is obtained by Matlab `ode45` function applied to the ODE system, where the time step is $\Delta t = 5 \times 10^{-7}$. We observe a second-order convergence rate with respect to $\Delta t$ in our method, which verifies the error estimate in Theorem 3.4. Most importantly, the error is independent of $\epsilon_1$, $\epsilon_2$, and $\epsilon_3$ as shown in Fig.9.
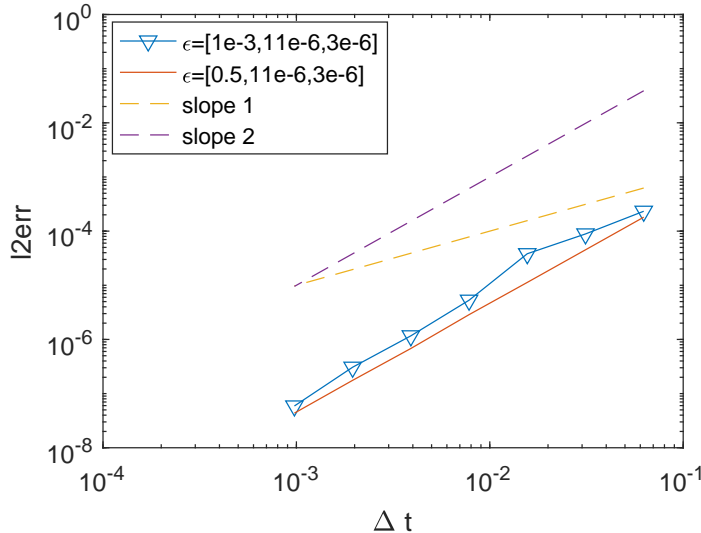


Figure 9: Error as a function of $\Delta t$ for two sets of multiscale parameters at time $T = 3$.

**Difference between $F$ and $\bar{f}^1$.** Simple calculations show that the right hand side of Eq.(30), i.e. $F(t, x)$ is approximately equal to $\bar{f}^1$. One may expect that replacing the ODE system (30) by $\dot{\tilde{y}} = \bar{f}^1$ will generate a solution that is close to the original one. We aim to investigate this issue in Fig.8(a). Here, we calculate $\bar{f}^1$ and directly solve $\dot{\tilde{y}} = \bar{f}^1$, which will be referred as the averaged method. In Fig.10(a) and Fig.10(b), we plot the first and fifth component of $\tilde{y}$, i.e., $w_1(t)$ and $w_5(t)$ obtained by several different methods.

Fig.10(a) shows that $w_1(t)$ is nearly a constant. Thus, the averaged method still performs well. Fig.10(b) shows that $w_5(t)$ has oscillations. In this case, the averaged method cannot capture the right behavior of the solution, while our method can. We find that (1) the direct *Im2nd* method with a coarse time step gives wrong results; (2) the solutions for component $w_1(t)$ obtained by the averaged method and our method agree with the reference solution; (3) the solution for component $w_5(t)$ obtained by the averaged method has large errors, while the solution for component $w_5(t)$ obtained by our method still approximates the reference solution well. This experiment shows that $F(t, x)$ in (30) indeed captures the correct dynamics of the original multiscale problem, while the direct average term $\bar{f}^1$ cannot.
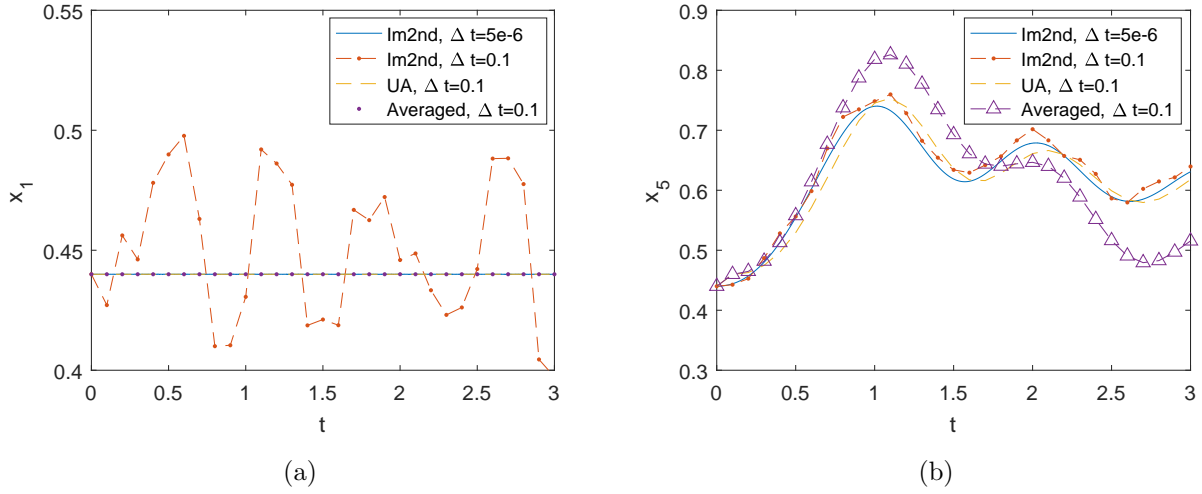
Figure 10: $w_1$ and $w_5$ obtained by using different methods.

### 4.3. An ODE with a complicated right-hand side

In the previous numerical experiments, all the integrals in our numerical schemes can be pre-computed analytically. Here, we consider a three-scale ODE, which is defined by

$$\dot{x} = (1.5 - \exp(\sin\frac{2\pi t}{\epsilon_1} + \sin\frac{2\pi t}{\epsilon_2}))x. \tag{62}$$

In this example, the right-hand side of the ODE (62) does not have analytic expressions. Thus, we use numerical quadrature rules to compute the integrals in our numerical schemes.

Since the integrands of $f^k$, $k = 1, 2$ are smooth along $x$ direction, we use 8 points in the quadrature rules to compute the integration for $\bar{f}^k$, e.g. Eqns.(4), (6), (8), and (10). To compute derivatives of $g$, we directly use a central difference scheme with $\Delta x = 10^{-3}$. We choose the initial value $x_0 = 0.48$ and $(\epsilon_1, \epsilon_2) = [7 \times 10^{-2}, 11 \times 10^{-5}]$ in the ODE (62).

In Fig.11(a), we show the numerical results obtained by different methods with different time steps. We find that: (1) the direct *Im2nd* method with the coarse time step $\Delta t = 0.1$ gives totally wrong results; (2) our method with very coarse time steps, $\Delta t = 0.1$ and $\Delta t = 0.5$, gives an accurate result that is comparable to that using the direct *Im2nd* method with a very fine time step $\Delta t = 10^{-5}$. This comparison again confirms that our method has a uniform accuracy in computing ODE system with multiple time-scales. In this experiment, our method with $\Delta t = 0.5$ costs $1.39s$, while the direct *Im2nd* method with $\Delta t = 10^{-5}$ costs $5.97s$. More savings can be achieved if the ODE (62) is described by smaller multiscale parameters $\epsilon_1$ and $\epsilon_2$.

As a byproduct of our robust numerical method (namely we can solve the complicated ODE with very large time step), we can use the representation in Eq.(14) to recover the solution of the ODE (62) in a neighborhood of the numerical solution points. In Fig.11(b), we show the recovered solution in the time domain $[0.5, 0.501]$ based on the solution of our method at $t = 0.5$. We can see that the recovered solution agrees with the reference solution and the recovered solution captures the high oscillate structure in this neighborhood.
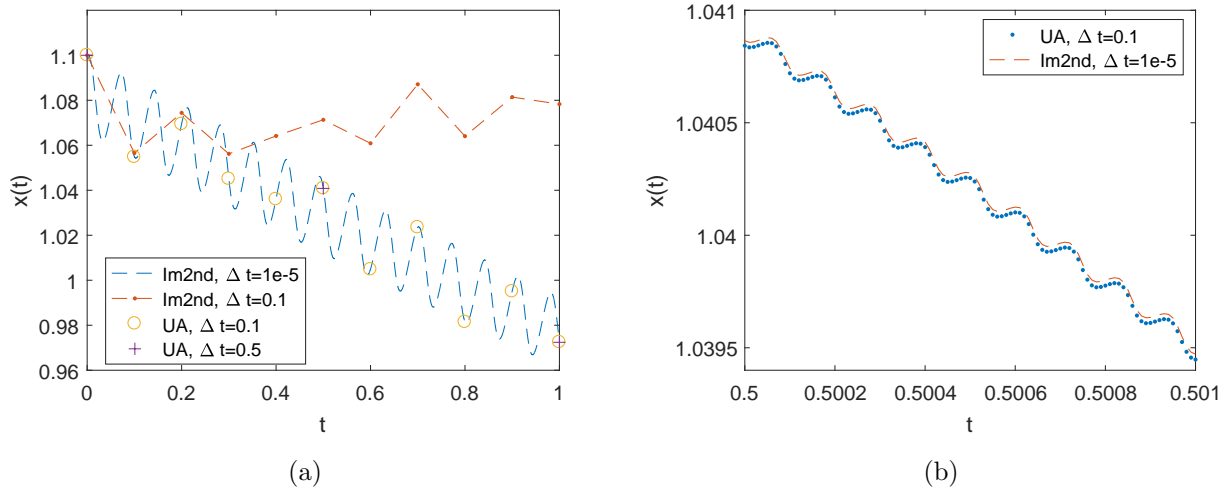
21

Figure 11: An ODE with a complicated right-hand side

## 5. Conclusions

In this paper, we have successfully developed a class of robust numerical methods to solve dynamical systems with multiple time scales. These problems are difficult to solve when the multiscale parameters are small. The essential idea of our method is to represent the solution of the dynamical systems as a transformation of a slowly varying solution. Based on the scale separation assumption, we provide an efficient way to construct the transformation map and derive the dynamic equation for the slowly varying solution. Under some mild assumptions, we obtain the convergence of the proposed method. Finally, we present several numerical examples, including ODE system with three and four separated time scales to demonstrate the accuracy and efficiency of the proposed method. Numerical results show that: (1) our method is robust and accurate in solving ODE systems with multiple time scale, where the time step does not depend on the multiscale parameters; and (2) the construction of the cumulative composition maps (which deals with the multiscale information in a dimension-by-dimension fashion) is necessary while a simple average treatment leads to wrong results.

There are two lines of work that deserve further explorations in the near future. Firstly, we shall consider to extend our proposed method to solve dynamical systems without scale separation. The idea mentioned in Remark 2.3 is a good starting point. However, we need to design fast solvers. Secondly, we are interested in extending our proposed method to solve elliptic PDEs with multiscale parameters.

## Acknowledgements

22

# References

[1] A. ABDULLE, E. WEINAN, B. ENGQUIST, AND E. VANDEN-EIJNDEN, *The heterogeneous multiscale method*, Acta Numerica, 21 (2012), pp. 1–87.

[2] S. BARTLETT, P.AND EVANS AND P. LONG, *Representing smooth functions as compositions of near-identity functions with implications for deep network optimization*, arXiv:1804.05012, (2018).

[3] P. CHARTIER, N. CROUSEILLES, M. LEMOU, AND F. MÉHATS, *Uniformly accurate numerical schemes for highly oscillatory Klein–Gordon and nonlinear Schrödinger equations*, Numerische Mathematik, 129 (2015), pp. 211–250.

[4] P. CHARTIER, M. LEMOU, F. MÉHATS, AND G. VILMART, *A new class of uniformly accurate numerical schemes for highly oscillatory evolution equations*, Found. Comput. Math., 19 (2019), pp. 1–33.

[5] Y. EFENDIEV AND T. Y. HOU, *Multiscale finite element methods. Theory and applications*, Springer-Verlag, New York, 2009.

[6] B. ENGQUIST AND T. HOU, *Particle method approximation of oscillatory solutions to hyperbolic differential equations*, SIAM journal on numerical analysis, 26 (1989), pp. 289–319.

[7] S. FERRER, M. LARA, J. PALACIAN, J. JUAN, A. VIARTOLA, AND P. YANGUAS, *The hénon and heiles problem in three dimensions. ii. relative equilibria and bifurcations in the reduced system*, International Journal of Bifurcation and Chaos, 8 (1998), pp. 1215–1229.

[8] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, vol. 31, Springer Science & Business Media, 2006.

[9] M. HÉNON AND C. HEILES, *The applicability of the third integral of motion: some numerical experiments*, The Astronomical Journal, 69 (1964), p. 73.

[10] T. HOU, *Homogenization for semilinear hyperbolic systems with oscillatory data*, Communications on pure and applied mathematics, 41 (1988), pp. 471–495.

[11] T. Hou, D. Yang, and H. Ran, *Multiscale analysis and computation for the three-dimensional incompressible Navier–Stokes equations*, Multiscale Model. Simul., 6 (2008), pp. 1317–1346.

[12] A. Iserles, *A first course in the numerical analysis of differential equations*, no. 44, Cambridge university press, 2009.

[13] C. Jones and A. Khibnik, *Multiple-time-scale dynamical systems*, vol. 122, Springer Science & Business Media, 2012.

[14] I. Kevrekidis, C. Gear, J. Hyman, P. Kevrekidid, O. Runborg, and C. Theodoropoulos, *Equation-free, coarse-grained multiscale computation: Enabling mocroscopic simulators to perform system-level analysis*, Commum. Math Sci., 1 (2003), pp. 715–762.

[15] C. Kuehn, *Multiple time scale dynamics*, vol. 191, Springer, 2015.

[16] L. Perko, *Differential equations and dynamical systems*, vol. 7, Springer Science & Business Media, 2013.

[17] Z. Shen, H. Yang, and S. Zhang, *Nonlinear approximation via compositions*, arXiv:1902.10170, (2019).

[18] M. Tao, H. Owhadi, and J. Marsden, *Nonintrusive and structure preserving multiscale integration of stiff ODEs, SDEs, and Hamiltonian systems with hidden slow dynamics via flow averaging*, Multiscale Model. Simul., 8 (2010), pp. 1269–1324.

[19] M. Tuckerman, B. Berne, and G. Martyna, *Reversible multiple time scale molecular dynamics*, J. Chem. Phys., 97 (1992), pp. 1990–2001.