SIAM J. SCI. COMPUT. © 2021 Society for Industrial and Applied Mathematics
Vol. 43, No. 1, pp. A636–A662

ERROR ESTIMATES FOR A POD METHOD FOR SOLVING VISCOUS G-EQUATIONS IN INCOMPRESSIBLE CELLULAR FLOWS[*]

HAOTIAN GU[†], JACK XIN[‡], AND ZHIWEN ZHANG[§]

Abstract. The G-equation is a well-known model for studying front propagation in turbulent combustion. In this paper, we develop an efficient model reduction method for computing regular solutions of viscous G-equations in incompressible steady and time-periodic cellular flows. Our method is based on the Galerkin proper orthogonal decomposition (POD) method. To facilitate the algorithm design and convergence analysis, we decompose the solution of the viscous G-equation into a mean-free part and a mean part, where their evolution equations can be derived accordingly. We construct the POD basis from the solution snapshots of the mean-free part. With the POD basis, we can efficiently solve the evolution equation for the mean-free part of the solution to the viscous G-equation. After we get the mean-free part of the solution, the mean of the solution can be recovered. We also provide rigorous convergence analysis for our method. Numerical results for viscous G-equations and curvature G-equations are presented to demonstrate the accuracy and efficiency of the proposed method. In addition, we study the turbulent flame speeds of the viscous G-equations in incompressible cellular flows.

Key words. viscous G-equation, Hamilton–Jacobi type equation, front speed computation, cellular flows, proper orthogonal decomposition method, POD, convergence analysis

AMS subject classifications. 65M12, 70H20, 76F25, 78M34, 80A25

DOI. 10.1137/19M1241854

1. Introduction.

Front propagation in turbulent combustion is a nonlinear and complicated dynamical process. The G-equation has been a very popular field model in combustion and physics literature for studying premixed turbulent flame propagation [31, 32, 14, 37, 34, 47]. The G-equation model is a sound phenomenological approach to study turbulent combustion, which uses the level-set formulation to study the flame front motion laws with the front width ignored. The simplest motion law is that the normal velocity of the front is equal to a constant S_l (the laminar speed) plus the projection of fluid velocity $V(\vec{x},t)$ along the normal. This gives the inviscid G-equation

$$G_t + \vec{V}\cdot\nabla G + S_l|\nabla G| = 0, \tag{1}$$

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section January 30, 2019; accepted for publication (in revised form) November 13, 2020; published electronically February 18, 2021.
https://doi.org/10.1137/19M1241854

Funding: The work of the second author was partially supported by NSF grants DMS-1211179, DMS-1522383, IIS-1632935, and DMS-1854434. The work of the third author was supported by Hong Kong RGC grants (projects 27300616, 17300817, and 17300318), the National Natural Science Foundation of China (project 11601457), the Basic Research Programme (JCYJ20180307151603959) of the Science, Technology and Innovation Commission of Shenzhen Municipality, Seed Funding Programme for Basic Research (HKU), and an RAE Improvement Fund from the Faculty of Science (HKU).

[†]Department of Mathematics, University of California at Berkeley, Berkeley, CA 94720 USA (htgu@math.berkeley.edu).
[‡]Department of Mathematics, University of California at Irvine, Irvine, CA 92697 USA (jack.xin@uci.edu).
[§]Corresponding author. Department of Mathematics, University of Hong Kong, 999077, Hong Kong SAR, China (zhangzw@hku.hk).

where the set $\{(\vec{x},t) : G(\vec{x},t) = 0\}$ corresponds to the location of the flame front at time t. The derivation of the inviscid G-equation will be elaborated in detail in section 2.1. Developing numerical methods for (1) has been an active research topic for decades; see, e.g., [35, 22, 13, 27, 16] and references therein.

As the fluid turbulence is known to cause stretching and corrugation of flames, additional modeling terms need to be incorporated into the basic G-equation. If the curvature term is added into the basic equation to model the curvature effects and the curvature term is further linearized, we will arrive at the viscous G-equation

$$G_t + \vec{V} \cdot \nabla G + S_l |\nabla G| = d S_l \Delta G. \tag{2}$$

In order to compute numerical solutions of (2), the authors of [28] first approximated the G-equations by a monotone discrete system, then applied high resolution numerical methods such as WENO (weighted essentially nonoscillatory) finite difference methods [21] with a combination of explicit and semi-implicit time stepping strategies, depending on the size and property of dissipation in the equations. However, these existing numerical methods become expensive when one needs to discretize (2) with a fine mesh or one needs to solve (2) many times with different parameters. This motivates us to exploit the low-dimensional structures of the regular solutions of viscous G-equation (2) and develop model reduction methods to solve them efficiently.

One of the successful model reduction ideas in the study of turbulent flows is the proper orthogonal decomposition (POD) method [43, 7]. The POD method uses the data from an accurate numerical simulation and extracts the most energetic modes in the system by using singular value decomposition. This approach generates low-dimensional structures that play an important role in the dynamics of the flow. The Galerkin POD method has been used to solve many types of partial differential equations, including linear parabolic equations, Burgers equations, Navier–Stokes equations, forward Kolmogorov equations, and Hamilton–Jacobi–Bellman equations; see [25, 26, 8, 45, 2, 23, 6, 24, 46, 4] and references therein for details. The interested reader is referred to [38, 6, 20] for a comprehensive introduction of the model reduction methods.

In this paper, we study the POD method to solve the viscous G-equation (2), which is a Hamilton–Jacobi type equation. To deal with the periodic boundary condition of the problem and facilitate the convergence analysis of the POD method, we decompose the solution of the viscous G-equation into a mean-free part and a mean part, where their evolution equations can be derived accordingly; see (15). We construct the POD basis from the snapshots of the mean-free part of the solutions, which can be used to solve the evolution equation for the mean-free part. Then, the mean part of the solution can be computed from the mean-free part; see (14). Notice that the bilinear form of the evolution equation for the mean-free part satisfies the coercive condition, which follows from the *Poincaré–Wirtinger inequality*. We provide rigorous convergence analysis and show that the accuracy of our method is guaranteed. We remark that the idea of decomposing the data into a mean-free part and a mean part plays an essential role in the convergence analysis of the POD method for solving the viscous G-equation. Finally, we conduct numerical experiments to demonstrate the accuracy and efficiency of the proposed method.

We find that the POD basis can be used to compute a long-time solution of the viscous G-equation or the viscous G-equations with different parameters. Moreover, we study the turbulent flame speeds of viscous G-equations in incompressible steady and time-periodic cellular flows, which help our understanding of turbulent combustion. To further reduce the numerical error, we develop an adaptive strategy to dynamically enrich the POD basis and demonstrate its effectiveness through a viscous G-equation with time-periodic fluid velocity. We remark that our POD method can easily be extended to solve other types of G-equations [28]. Though we are unable to provide rigorous convergence analysis, we find that the POD method is still efficient in solving the curvature G-equation when its solution is smooth. To the best of our knowledge, our result is the first one in the literature that develops a POD-based model reduction method to solve G-equations and compute their front speeds.

The rest of this paper will be organized as follows. In section 2, we shall give a brief derivation of G-equation models. In section 3, we show the detailed derivations of the model reduction method for G-equations. In section 4, we provide the convergence analysis of the proposed method. Our proof is based on the backward Euler–Galerkin-POD approximation scheme. The proofs for other discretization schemes, such as the Crank–Nicolson scheme, can be obtained in a similar manner. In section 5, we shall perform numerical experiments to test the performance and accuracy of the proposed method. We find that the POD method can provide considerable savings over finite difference methods in solving the G-equation while its numerical error is relatively small. Finally, concluding remarks will be given in section 6. In the appendix, we provide the derivation of a finite difference scheme to solve G-equations proposed in [28] and the procedure for constructing a POD basis.

2. Turbulent combustion and G-equations.

2.1. Derivation of the G-equations. In this section, we briefly introduce the derivation of the G-equation in turbulent combustion. In a thin reaction zone regime and the corrugated flamelet regime of premixed turbulent combustion (Chapter 2 of [37]), the flame front is modeled by a level set function, $\{(\vec{x},t) : G(\vec{x},t) = 0, \vec{x} \in R^n\}$, which is the interface between the burned area, denoted by $\{G < 0\}$, and the unburned area, denoted by $\{G > 0\}$, respectively. Therefore, one can study the propagation of the flame front by solving the dynamic equation for the level set function, namely the G-equation.

The simplest motion law for the particles on the interface is that the normal velocity of the interface is the sum of a constant S_l and the projection of fluid velocity $\vec{V}(\vec{x},t)$ along the normal direction (see Figure 1). Hence, the trajectory of a particle $\vec{x}(t)$ on the interface satisfies

$$\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = \vec{V}(\vec{x},t) + S_l\vec{n}, \tag{3}$$

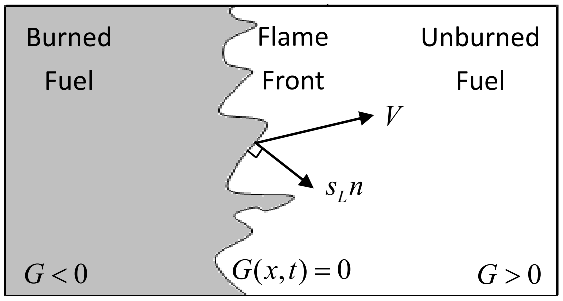


FIG. 1. *Illustration of local interface velocities in the G-equation and a flame front in a two-dimensional (2D) space.*

where S_l is the laminar flame speed and $\vec{n}$ is the normal vector. In terms of the level set function, the motion law (3) gives the inviscid G-equation,

$$G_t + \vec{V} \cdot \nabla G + S_l |\nabla G| = 0, \tag{4}$$

where ∇ denotes the gradient operator. Thus, the normal vector in (3) can be computed using $\vec{n} = \nabla G / |\nabla G|$. Notice that the set $\{G(\vec{x}, t) = 0\}$ corresponds to the location of the flame front at time t.

To take into account the effect of flame stretching and corrugation, additional terms are added into the inviscid G-equation (4) and one can obtain extended G-equation models involving curvature effects. The curvature G-equation is

$$G_t + \vec{V} \cdot \nabla G + S_l |\nabla G| = d S_l |\nabla G| \left(\nabla \cdot \frac{\nabla G}{|\nabla G|} \right), \tag{5}$$

where d is the Markstein number. The mean curvature term leads to a nonlinear degenerate second-order PDE that has been widely studied in the literature from both the analytical and numerical points of view. The curvature dependent motion is well-known; see [15, 35, 34, 9] and references therein. However, we are not aware of a POD approximation of curvature motion in the general setting. We remark that the classical boundary conditions to deal with first-order level set equations and second-order mean curvature motion equations are Neumann type boundary conditions. We are interested in computing the turbulent flame speeds of the G-equations in incompressible flows. Thus, we choose periodic boundary conditions in the G-equation (5), which will be further clarified in the numerical experiment; see section 5.7.

If the curvature term is linearized, we obtain the viscous G-equation as follows, which is the research focus in this paper:

$$G_t + \vec{V} \cdot \nabla G + S_l |\nabla G| = d S_l \Delta G. \tag{6}$$

The linearized version of the curvature G-equation, i.e., the viscous G-equation (6), allows us to carry out a convergent Galerkin approximation and POD error analysis, which is unavailable for (5). Our POD method remains efficient for smooth solutions of the curvature G-equation (5), as shown numerically later.

2.2. A periodic initial value problem. Now given a unit vector $\vec{P} \in \mathbb{R}^n$, where n is the dimension of the physical space, we shall consider the viscous G-equation (6) with planar initial condition

$$\begin{cases} G_t + \vec{V} \cdot \nabla G + S_l |\nabla G| = d S_l \Delta G & \text{in } \mathbb{R}^n \times (0, \infty), \\ G(\vec{x}, 0) = \vec{P} \cdot \vec{x} & \text{on } \mathbb{R}^n \times \{t = 0\}. \end{cases} \tag{7}$$

Here, we assume the flame front propagates in direction $\vec{P}$ with the initial front being $\{\vec{P} \cdot \vec{x} = 0\}$ and $\vec{x} = (x_1, x_2, \ldots, x_n)$. In addition, we assume $\vec{V}(\vec{x}, t)$ is spatially periodic with C^1 in $\vec{x}$ and C^0 in t. Moreover, $\vec{V}(\vec{x}, t)$ is divergence-free, i.e., $\nabla_{\vec{x}} \cdot \vec{V}(\vec{x}, t) = 0 \; \forall t$, and uniformly bounded, i.e., $||\vec{V}(\vec{x}, t)||_\infty \le A$.

If we write $G(\vec{x}, t) = \vec{P} \cdot \vec{x} + u(\vec{x}, t)$, then $u(\vec{x}, t)$ is also spatially periodic and satisfies the following periodic initial value problem:

$$\begin{cases} u_t + \vec{V} \cdot (\vec{P} + \nabla u) + S_l |\vec{P} + \nabla u| = d S_l \Delta u & \text{in } \mathbb{T}^n \times (0, \infty), \\ u(\vec{x}, 0) = 0 & \text{on } \mathbb{T}^n \times \{t = 0\}, \end{cases} \tag{8}$$

where $\mathbb{T}^n = [0,1]^n$. Hence we can reduce the problem (7) defined on the whole domain $\mathbb{R}^n$ to the problem (8) defined on $[0,1]^n$ by imposing the affine periodic condition

$$\begin{cases} G_t + \vec{V}\cdot\nabla G + S_l|\nabla G| = dS_l\Delta G & \text{in } \mathbb{T}^n\times(0,\infty),\\ G(\vec{x},0) = \vec{P}\cdot\vec{x} & \text{on } \mathbb{T}^n\times\{t=0\} \end{cases} \tag{9}$$

with the assumption $G(\vec{x}+\vec{z},t) = G(\vec{x},t) + \vec{P}\cdot\vec{z}$.

To illustrate the main idea of our method, we choose $\vec{P} = \vec{e}_1 = (1,0,\dots,0)$ and have $G(\vec{x},t) = x_1 + u(\vec{x},t)$. Let $\mathcal{A}(t)$ denote the volume of the burned area that has invaded during time interval $(0,t)$. Then, the turbulent flame speed S_T is defined as the linear growth rate of $\mathcal{A}(t)$. Notice that $G(\vec{x},0) = x_1$ and $G(\vec{x}+\vec{e}_1,t) = G(\vec{x},t)+1$. Then, the S_T can be evaluated by $G(\vec{x},t)$ or $u(\vec{x},t)$ in $[0,1]^n$,

$$S_T = \lim_{t\to\infty}\frac{-1}{t}\int_{[0,1]^n} G(\vec{x},t)d\vec{x} = \lim_{t\to\infty}\frac{-1}{t}\int_{[0,1]^n}(x_1+u(\vec{x},t))d\vec{x}. \tag{10}$$

In this work, we are interested in computing the turbulent flame speeds of the viscous G-equations in incompressible cellular flows, which is an effective (homogenized) quantity and involves a long-time computation. To this end, we impose periodic boundary conditions for (9). We will study the POD method for solving other nonlinear PDEs with other types of boundary conditions in our future research.

There are many numerical methods for solving the viscous G-equation (9) or equation (8) on a bounded physical domain $\mathbb{T}^n$. For instance, we can apply a higher-order Hamilton–Jacobi weighted essentially nonoscillatory (HJ-WENO) scheme and total variation diminishing Runge–Kutta (TVD-RK) scheme in the spatial and time discretization, respectively. See [42, 21, 34, 28] for details of the schemes. To make our paper self-contained, we show the numerical schemes proposed by [28]; see the appendix for more details.

3. Model reduction method for G-equations.

In practice, we are interested in studying the dependence of the turbulent flame speed S_T on different parameters in the viscous G-equation, such as the Markstein number d. As such, we have to solve the viscous G-equation many times, which is an expensive task. Therefore, we need to design numerical methods that allow us to efficiently and accurately solve the viscous G-equation. We shall develop an efficient model reduction method to achieve this goal. We point out that the model reduction methods also bring computational savings when one needs to solve high-dimensional problems, e.g., 3D viscous G-equations.

3.1. A decomposition strategy.

According to the definition in (10), we can either solve (8) to obtain $u(x,t)$ or solve (9) to obtain $G(x,t)$, in order to compute the turbulent flame speed S_T. We shall consider solving (8) since it is easier to deal with the boundary condition. Let us decompose the solution u of (8) into $\hat{u}+\bar{u}$, where $\hat{u}$ is the mean-free part and $\bar{u}$ is the mean of u. This decomposition means that

$$\int_{\mathbb{T}^n}\hat{u}(\vec{x},t)d\vec{x} = 0 \quad \forall t \quad \text{and} \quad \bar{u}(t) = \int_{\mathbb{T}^n} u(\vec{x},t)d\vec{x}. \tag{11}$$

Then, substituting $u = \hat{u}+\bar{u}$ into (8), we obtain

$$\hat{u}_t + \bar{u}_t + \vec{V}\cdot\left(\vec{P}+\nabla\hat{u}\right) + S_l\left|\vec{P}+\nabla\hat{u}\right| - dS_l\Delta\hat{u} = 0. \tag{12}$$

Integrating the above equation over the domain $\mathbb{T}^n$ gives

$$\int_{\mathbb{T}^n}\left[\hat{u}_t + \bar{u}_t + \vec{V}\cdot(\vec{P}+\nabla\hat{u}) - dS_l\Delta\hat{u}\right]d\vec{x} = -\int_{\mathbb{T}^n} S_l\left|\vec{P}+\nabla\hat{u}\right|d\vec{x}. \tag{13}$$

The definitions of $\bar{u}$ and $\hat{u}$ in (11) imply that $\int_{\mathbb{T}^n} \hat{u}_t = 0$ and $\int_{\mathbb{T}^n} \bar{u}_t = \bar{u}_t$. Furthermore, the periodic conditions of $\vec{V}$ and $\hat{u}$ imply $\int_{\mathbb{T}^n} \vec{V}\cdot\vec{P} = 0$, $\int_{\mathbb{T}^n} \Delta\hat{u} = 0$, and $\int_{\mathbb{T}^n} \vec{V}\cdot\nabla\hat{u} = 0$, where we have used the facts that $\vec{V}$ is divergence-free and $\hat{u}$ is the mean-free. Combining these results, we can simplify (13) as

$$\bar{u}_t = -\int_{\mathbb{T}^n} S_l\big|\vec{P} + \nabla\hat{u}\big| d\vec{x}. \tag{14}$$

Finally, we find that (8) is equivalent to

$$\begin{cases} \hat{u}_t + \vec{V}\cdot\nabla\hat{u} - dS_l\Delta\hat{u} + S_l\big|\vec{P} + \nabla\hat{u}\big| & \\ \quad - \int_{\mathbb{T}^n} S_l\big|\vec{P} + \nabla\hat{u}\big|d\vec{x} + \vec{V}\cdot\vec{P} = 0 & \text{in } \mathbb{T}^n\times(0,\infty), \\ \bar{u}_t = -\int_{\mathbb{T}^n} S_l\big|\vec{P} + \nabla\hat{u}(\vec{x},t)\big|d\vec{x} & \text{on } t\in(0,\infty), \\ \hat{u}(\vec{x},0) = 0 & \text{on } \mathbb{T}^n\times\{t=0\}, \\ \bar{u}(0) = 0. & \end{cases} \tag{15}$$

The strategy of decomposing the solution or data into a mean-free part and a mean part is often used in scientific computing and data science. However, it has not been studied in the G-equation setting. We shall demonstrate later that this decomposition plays a crucial role in the algorithm design and facilitates the convergence analysis of our proposed method.

3.2. Construction of the POD basis.

In this section, we shall present our model reduction method to solve (15). Since the evolution equation for $\bar{u}$ depends on $\hat{u}$, we first consider solving the equation for $\hat{u}$, i.e.,

$$\hat{u}_t + \vec{V}\cdot\nabla\hat{u} - dS_l\Delta\hat{u} + S_l\big|\vec{P}+\nabla\hat{u}\big| - \int_{\mathbb{T}^n} S_l\big|\vec{P}+\nabla\hat{u}\big|d\vec{x} + \vec{V}\cdot\vec{P} = 0, \text{ in } \mathbb{T}^n\times(0,\infty). \tag{16}$$

Let $H^1_{per}(\mathbb{T}^n)$ denote the Sobolev space on the domain $\mathbb{T}^n$ with a periodic boundary condition and let $\langle\cdot,\cdot\rangle$ denote the standard inner product on $L^2(\mathbb{T}^n)$. Let $H \subset H^1_{per}(\mathbb{T}^n)$ be the subspace consisting of all mean-free functions. Since H is a closed subspace of $H^1_{per}(\mathbb{T}^n)$, H itself is a Hilbert space. Let $\langle\cdot,\cdot\rangle_H$ denote the standard inner product on H. Define the bilinear form $a(\cdot,\cdot): H\times H\to\mathbb{R}$ to be

$$a(u,v) = \int_{\mathbb{T}^n} \big[(\vec{V}\cdot\nabla u)v + dS_l(\nabla u\cdot\nabla v)\big]d\vec{x}. \tag{17}$$

Also, define a nonlinear map from H to $L^2(\mathbb{T}^n)$ to be

$$F(u) = S_l\big|\vec{P}+\nabla u\big| - \int_{\mathbb{T}^n} S_l\big|\vec{P}+\nabla u\big|d\vec{x}. \tag{18}$$

The weak formulation associated with (16) is

$$\langle\hat{u}_t,\psi\rangle + a(\hat{u},\psi) + \langle F(\hat{u}),\psi\rangle = \langle -\vec{P}\cdot\vec{V},\psi\rangle \;\forall\psi\in H, \tag{19}$$

which can be solved by using numerical methods.

The POD technique for discretization of (16) requires snapshots, which can be obtained by an independent numerical method or by the appropriate technological means related to a specific application, for instance, the data from an experiment. We apply a higher-order WENO scheme and TVD-RK scheme to solve problem

(8) and extract the mean part to obtain a set of numerical solutions $\{\hat{u}(\cdot, t_k)\}$ to (16), where $t_k = k\Delta t$, $\Delta t = T/m$, $k = 0, \dots, m$. Then, we use $2m+1$ snapshots $\{\hat{u}(\cdot, t_0), \dots, \hat{u}(\cdot, t_m), \bar{\partial}\hat{u}(\cdot, t_1), \dots, \bar{\partial}\hat{u}(\cdot, t_m)\}$, where $\bar{\partial}\hat{u}(t_i) = (\hat{u}(t_i) - \hat{u}(t_{i-1}))/\Delta t$, to generate the POD basis functions. Let $S^r = \{\psi_1(\cdot), \psi_2(\cdot), \dots, \psi_r(\cdot)\}$ denote the r-dimensional orthonormal POD basis functions obtained from the $2m+1$ snapshots, which minimize the following error:

$$\begin{aligned}(20)\qquad &\frac{1}{2m+1}\sum_{i=0}^{m}\left\|\hat{u}(t_i) - \sum_{j=1}^{r}\langle \hat{u}(t_i), \psi_j\rangle_H \psi_j\right\|_H^2 \\ &+ \frac{1}{2m+1}\sum_{i=1}^{m}\left\|\bar{\partial}\hat{u}(t_i) - \sum_{j=1}^{r}\langle \bar{\partial}\hat{u}(t_i), \psi_j\rangle_H \psi_j\right\|_H^2.\end{aligned}$$

See section A.3 for the details of the POD method. By adding the terms $\bar{\partial}\hat{u}(t_i)$, $i = 1, \dots, m$, into the snapshots, we obtain more accurate POD basis functions and avoid an extra $(\Delta t)^{-2}$ factor in the convergence analysis; see section 4 for more details.

In practice, we can use experimental data or reference numerical solutions to generate solution snapshots. The choice of snapshots is critical to the quality of the POD method. The snapshots should span a linear space that approximates the solution space of the original PDEs well. However, the POD method itself gives no guidance on how to select the snapshots (page 503 of [6]). One may need some a posteriori error estimate for the solution obtained by the POD method. For time-dependent parametric PDEs, it is difficult to obtain an a posteriori error estimate for the PDEs. We propose an adaptive strategy to enrich the POD basis; see section 5.6 for more details. We refer the interested reader to [19, 40, 1, 12, 36, 39, 29, 18, 30, 11] and references therein for the adaptive techniques in the model reduction methods.

Remark 3.1. The construction of the POD basis can be costly. One can apply randomized projection methods [3] to reduce the cost in the offline stage. Once the construction is done, the POD basis can be used to solve viscous G-equations with different parameters, which brings significant computational savings in the online stage.

3.3. A backward Euler and POD-based Galerkin method. The POD basis functions provide an efficient approach to approximate the solution in the physical space. If we choose the backward Euler scheme to discretize the time space, we obtain a backward Euler and POD-based Galerkin method to solve (16). Specifically, let $\hat{U}_k \equiv \sum_{i=1}^{r} a_i(t_k)\psi_i$ denote the numerical solution at $t = t_k$, where ψ_i's are the POD basis functions. We want to find solutions $\{\hat{U}_k\}_{k=0}^{m} \subset S^r$ satisfying

$$(21)\qquad \begin{cases} \langle \bar{\partial}\hat{U}_k, \psi\rangle + a(\hat{U}_k, \psi) + \langle F(\hat{U}_k), \psi\rangle = \langle -\vec{P}\cdot\vec{V}, \psi\rangle \quad \forall \psi \in S^r, \\ \hat{U}_0 = 0, \end{cases}$$

where $\bar{\partial}\hat{U}_k = (\hat{U}_k - \hat{U}_{k-1})/\Delta t$. By choosing the test function ψ to be ψ_i, $i = 1, \dots, r$ in (21) and letting $\mathbf{a}_k = \big(a_1(t_k), \dots, a_r(t_k)\big)^T$ denote the coefficient vector, we obtain a nonlinear equation system for $\mathbf{a}_k$ as

$$(22)\qquad \mathbf{M}_1\mathbf{a}_k = \mathbf{M}_2\mathbf{a}_{k-1} + \mathbf{c} + \mathbf{f}_k,$$

where $\mathbf{M}_1, \mathbf{M}_2 \in \mathbb{R}^{r\times r}$ with $(\mathbf{M}_1)_{ij} = \langle \psi_i, \psi_j\rangle + \Delta t \cdot a(\psi_i, \psi_j)$ and $(\mathbf{M}_2)_{ij} = \langle \psi_i, \psi_j\rangle$, $\mathbf{c} \in \mathbb{R}^r$ with $c_i = -\Delta t\langle \vec{V}\cdot\vec{P}, \psi_i\rangle$, and $F_k \in \mathbb{R}^r$ with $(\mathbf{f}_k)_i = -\Delta t\big\langle F(\sum_i^r a_i(t_k)\psi_i), \psi_i\big\rangle$. The matrices $\mathbf{M}_1$, $\mathbf{M}_2$, and $\mathbf{c}$ can be precomputed and saved.

Equation (22) can be efficiently computed using Newton's method, where the solution at time t_{k-1} can be chosen as the initial guess for $\mathbf{a}_k$. The proposed method is very efficient since the number of the POD basis is small in general. For more challenging problems, such as 3D problems or convection-dominated problems (leading to an increase of the number of the POD basis), the effect of the nonlinear term becomes significant. One can choose the discrete empirical interpolation method for nonlinear model reduction [10].

We solve (22) to get the POD solution $\hat{U}_k$, which is the numerical solution of the mean-free part $\hat{u}(\vec{x},t)$ of the G-equation (16). As indicated by (15), $u(\vec{x},t)$ is recovered by adding the mean solution back,

$$u(\vec{x},t) = \hat{u}(\vec{x},t) - \int_0^t \int_{\mathbb{T}^n} S_l \big|\vec{P} + \nabla \hat{u}(\vec{x},s)\big| \mathrm{d}\vec{x}\mathrm{d}s. \tag{23}$$

Since the solution $\hat{u}(\vec{x},t)$ is smooth, we use a high-order difference scheme to compute the spatial derivatives of $\nabla \hat{u}(\vec{x},\cdot)$ and use extrapolations at boundaries to approximate the spatial derivatives to maintain the second-order accuracy. Then, we apply a composite trapezoidal rule to compute the spatial integration and temporal integration in (23). Finally, we obtain the numerical solution to $u(\vec{x},t)$ as follows:

$$U_k = \hat{U}_k - \frac{1}{2}\Delta t \sum_{i=1}^{k} \Big[\mathbb{I}\big(S_l|\vec{P} + \nabla \hat{U}_{i-1}(\vec{x})|\big) + \mathbb{I}\big(S_l|\vec{P} + \nabla \hat{U}_{i}(\vec{x})|\big) \Big], \tag{24}$$

where $\mathbb{I}(\cdot)$ denotes the numerical integrator by the composite trapezoidal rule. Moreover, we assume the mesh size of the composite trapezoidal rule is h, which is determined by the numerical method in computing the snapshots. One can also choose a wider finite difference scheme to compute the spatial derivatives and high-order numerical methods to compute the integration in (23).

4. Convergence analysis.

In this section, we shall present some convergence analysis to show that the accuracy of the numerical solution is guaranteed. Our convergence analysis follows the framework of the Galerkin finite element methods for parabolic problems [44]. To deal with the nonlinearity of the viscous G-equation, the following lemmas are useful.

LEMMA 4.1. *There exists a constant $\gamma > 0$ such that for any $u, v \in H$,*

$$||F(u) - F(v)||_{L^2} \le \gamma ||u - v||_H. \tag{25}$$

Proof. According to the definition of F in (18), the proof of this lemma can be easily obtained by using the triangle inequality. ∎

LEMMA 4.2. *The bilinear form $a(\cdot,\cdot)$ defined in (17) is continuous and coercive, which means that there exist constants $\beta > 0$ and $\kappa > 0$ such that for any $\psi, \phi \in H$*

$$a(\psi,\phi) \le \beta ||\psi||_H ||\phi||_H, \quad a(\psi,\psi) \ge \kappa ||\psi||_H^2. \tag{26}$$

Proof. Let $||\vec{V}||_\infty$ denote the maximum amplitude of the vector field $\vec{V}$. One has the estimate

$$\begin{aligned} a(\psi,\phi) &\le \int_{\mathbb{T}^n} ||\vec{V}||_\infty |\nabla\psi| \cdot |\phi| + dS_l |\nabla\psi| \cdot |\nabla\phi| \\ &\le ||\vec{V}||_\infty ||\psi||_H ||\phi||_H + dS_l ||\psi||_H ||\phi||_H. \end{aligned}$$

The last inequality follows from the Cauchy–Schwarz inequality. Moreover, since $\vec{V}$ is divergence-free, $\vec{V}\cdot\nabla$ is skew-symmetric, which means

$$a(\psi,\psi) = dS_l \int_{\mathbb{T}^n} |\nabla\psi|^2.$$

Since ψ is mean-free, the Poincaré–Wirtinger inequality implies that there exists a constant C, depending only on the domain $\mathbb{T}^n$, such that $\int_{\mathbb{T}^n} |\psi|^2 \leq C\int_{\mathbb{T}^n} |\nabla\psi|^2$. Therefore, we know that $a(\cdot,\cdot)$ is coercive, i.e.,

$$a(\psi,\psi) \geq \kappa ||\psi||_H^2,$$

where $\kappa = dS_l/(C+1) > 0$. □

In our method, we decompose the solution into a mean-free part and a mean part. The bilinear form of the evolution equation for the mean-free part satisfies the coercive condition, which plays an essential role in the convergence analysis. And the error estimate for the mean part of the solution can be obtained subsequently.

Now we define the Ritz-projection $P^r : H \to S^r$, $u \mapsto P^r u$ such that

$$a(P^r u, \psi) = a(u,\psi) \quad \forall \psi \in S^r. \tag{27}$$

Facts from functional analysis guarantee that P^r is well-defined and bounded because $a(\cdot,\cdot)$ is continuous and coercive. More specifically,

$$||P^r u||_H \leq \frac{\beta}{\kappa}||u||_H \quad \forall u \in H. \tag{28}$$

Using the same argument as in Lemmas 3 and 4 in [25], we can prove that P^r has the following approximation property. More details of the approximation property of the POD basis can be found in section A.3.

LEMMA 4.3.

$$\frac{1}{m}\sum_{k=0}^{m} ||\hat{u}(t_k) - P^r\hat{u}(t_k)||_H^2 \leq \frac{3\beta}{\kappa}\sum_{l=r+1}^{m}\lambda_l \tag{29}$$

and

$$\frac{1}{m}\sum_{k=1}^{m} ||\bar{\partial}\hat{u}(t_k) - P^r\bar{\partial}\hat{u}(t_k)||_H^2 \leq \frac{3\beta}{\kappa}\sum_{l=r+1}^{m}\lambda_l, \tag{30}$$

where λ_l is the lth largest eigenvalue of the correlation matrix K associated with the solution snapshots.

THEOREM 4.4. *Let $\hat{u}(\cdot)$ and $\{\hat{U}_k\}_{k=0}^m$ be the solutions to* (19) *and its backward Euler and POD-based Galerkin approximation, respectively. Then for sufficiently small Δt, there exists a constant $C > 0$ depending on $\hat{u}$, d, S_l, $\vec{V}$, $\vec{P}$, and T but independent of r, Δt, and m such that*

$$\frac{1}{m}\sum_{k=1}^{m} \left|\left|\hat{U}_k - \hat{u}(t_k)\right|\right|_{L^2}^2 \leq C\left((\Delta t)^2 + \sum_{l=r+1}^{m}\lambda_l\right). \tag{31}$$

Proof. For $k = 0, 1, \dots, m$, define $\vartheta_k = \hat{U}_k - P^r \hat{u}(t_k)$ and $\varrho_k = P^r \hat{u}(t_k) - \hat{u}(t_k)$. Then

$$\frac{1}{m}\sum_{k=1}^{m} ||\hat{U}_k - \hat{u}(t_k)||_{L^2}^2 \leq \frac{2}{m}\sum_{k=1}^{m} ||\vartheta_k||_{L^2}^2 + \frac{2}{m}\sum_{k=1}^{m} ||\varrho_k||_{L^2}^2, \tag{32}$$

$$\frac{2}{m}\sum_{k=1}^{m} ||\varrho_k||_{L^2}^2 \leq \frac{2}{m}\sum_{k=1}^{m} ||\varrho_k||_{H}^2 \leq \frac{6\beta}{\kappa}\sum_{l=r+1}^{m} \lambda_l. \tag{33}$$

Define $\bar{\partial}\vartheta_k = (\vartheta_k - \vartheta_{k-1})/\Delta t$. For all $\psi \in S^r$,

$$\langle \bar{\partial}\vartheta_k, \psi\rangle + a(\vartheta_k, \psi) = \langle \upsilon_k, \psi\rangle + \langle F(\hat{u}(t_k)) - F(\hat{U}_k), \psi\rangle, \tag{34}$$

where $\upsilon_k = \hat{u}_t(t_k) - P^r\bar{\partial}\hat{u}(t_k) = \hat{u}_t(t_k) - \bar{\partial}\hat{u}(t_k) + \bar{\partial}\hat{u}(t_k) - P^r\bar{\partial}\hat{u}(t_k)$. Define $\omega_k = \hat{u}_t(t_k) - \bar{\partial}\hat{u}(t_k)$ and $\eta_k = \bar{\partial}\hat{u}(t_k) - P^r\bar{\partial}\hat{u}(t_k)$. Taking $\psi = \vartheta_k \in S^r$ in the previous equality, we obtain

$$\frac{1}{\Delta t}\big(||\vartheta_k||_{L^2}^2 - \langle \vartheta_k, \vartheta_{k-1}\rangle\big) + \kappa||\vartheta_k||_{L^2}^2 \leq ||F(\hat{u}(t_k)) - F(\hat{U}_k)||_{L^2}||\vartheta_k||_{L^2} + ||\vartheta_k||_{L^2}||\upsilon_k||_{L^2}. \tag{35}$$

By using Lemma 4.1, we get

$$\begin{aligned} ||F(\hat{u}(t_k)) - F(\hat{U}_k)||_{L^2}||\vartheta_k||_{L^2} &\leq \gamma||\hat{u}(t_k) - \hat{U}_k||_H||\vartheta_k||_{L^2} \\ &\leq \gamma\big(||\varrho_k||_H + ||\vartheta_k||_H\big)||\vartheta_k||_{L^2}. \end{aligned}$$

Since $\vartheta_k \in S^r$, which is a finite dimensional space, the norms defined on S^r are equivalent. This means that there exists some constant $C_1 > 0$ such that

$$\begin{aligned} ||F(\hat{u}(t_k)) - F(\hat{U}_k)||_{L^2}||\vartheta_k||_{L^2} &\leq C_1\big(||\varrho_k||_H + ||\vartheta_k||_{L^2}\big)||\vartheta_k||_{L^2} \\ &\leq \frac{C_1}{2}||\varrho_k||_H^2 + \frac{3C_1}{2}||\vartheta_k||_{L^2}^2. \end{aligned}$$

Combining these inequalities, we have

$$||\vartheta_k||_{L^2}^2 - \langle \vartheta_k, \vartheta_{k-1}\rangle \leq \Delta t\left(||\vartheta_k||_{L^2}||\upsilon_k||_{L^2} + \frac{C_1}{2}||\varrho_k||_H^2 + \frac{3C_1}{2}||\vartheta_k||_{L^2}^2\right).$$

By using the inequality of arithmetic and geometric means, we obtain

$$\big(1 - (1 + C_1)\Delta t\big)||\vartheta_k||_{L^2}^2 \leq ||\vartheta_{k-1}||_{L^2}^2 + \Delta t\big(||\upsilon_k||_{L^2}^2 + 3C_1||\varrho_k||_H^2\big).$$

For sufficiently small Δt, there exists a constant $C_2 > 0$ such that

$$||\vartheta_k||_{L^2}^2 \leq (1 + C_2\Delta t)\left(||\vartheta_{k-1}||_{L^2}^2 + \Delta t\big(||\upsilon_k||_{L^2}^2 + 3C_1||\varrho_k||_H^2\big)\right). \tag{36}$$

By iteratively using the inequality (36), we have

$$||\vartheta_k||_{L^2}^2 \leq e^{C_2 T}\left(||\vartheta_0||_{L^2}^2 + \Delta t\sum_{j=1}^{k}\big(||\upsilon_j||_{L^2}^2 + 3C_1||\varrho_j||_H^2\big)\right). \tag{37}$$

Note that $\vartheta_0 = 0$ in our case. Therefore, we sum the inequality (37) from $k = 1$ to m and arrive at

$$\begin{aligned}\sum_{k=1}^{m} ||\vartheta_k||_{L^2}^2 &\le e^{C_2 T}\Delta t \sum_{k=1}^{m}\sum_{j=1}^{k} \left(||\upsilon_j||_{L^2}^2 + 3C_1||\varrho_j||_H^2\right)\\ &\le e^{C_2 T} T \sum_{j=1}^{m} \left(||\upsilon_j||_{L^2}^2 + 3C_1||\varrho_j||_H^2\right)\\ &\le e^{C_2 T} T \sum_{j=1}^{m} \left(2||\omega_j||_{L^2}^2 + 2||\eta_j||_{L^2}^2 + 3C_1||\varrho_j||_H^2\right).\end{aligned}$$

Therefore,

$$\begin{aligned}\frac{1}{m}\sum_{k=1}^{m} ||\hat{U}_k - \hat{u}(t_k)||_{L^2}^2 &\le \frac{2}{m}\sum_{k=1}^{m} ||\vartheta_k||_{L^2}^2 + \frac{6\beta}{\kappa}\sum_{k=r+1}^{m}\lambda_k\\ &\le \frac{2e^{C_2 T}T}{m}\sum_{j=1}^{m}\left(2||\omega_j||_{L^2}^2 + 2||\eta_j||_{L^2}^2 + 3C_1||\varrho_j||_H^2\right) + \frac{6\beta}{\kappa}\sum_{l=r+1}^{m}\lambda_l.\end{aligned}$$

From $\omega_j = \hat{u}_t(t_j) - \bar{\partial}\hat{u}(t_j) = \frac{1}{\Delta t}\int_{t_{j-1}}^{t_j}(t - t_{j-1})\hat{u}_{tt}(t)dt$, we obtain that

(38)
$$\sum_{j=1}^{m} ||\omega_j||_{L^2}^2 \le \sum_{j=1}^{m} \frac{1}{(\Delta t)^2}\int_{t_{j-1}}^{t_j}(t - t_{j-1})^2 dt \int_{t_{j-1}}^{t_j} ||\hat{u}_{tt}(t)||_{L^2}^2 dt \le \frac{\Delta t}{3}\int_0^T ||\hat{u}_{tt}(t)||_{L^2}^2.$$

Together with Lemma 4.3, we obtain

$$\frac{1}{m}\sum_{k=1}^{m} ||\hat{U}_k - \hat{u}(t_k)||_{L^2}^2 \le \left(\frac{4e^{C_2 T}}{3}\int_0^T ||\hat{u}_{tt}(t)||_{L^2}^2\right)(\Delta t)^2 + C_3\sum_{l=r+1}^{m}\lambda_l,$$

where $C_3 = \frac{3\beta}{\kappa}\left(2e^{C_2 T}T(2+3C_1)+2\right)$ and the fact $\Delta t = T/m$ is used. This completes the proof for Theorem 4.4. ◻

Theorem 4.4 provides an error estimate for the mean-free part. The mean part depends on the mean-free part and the original solution can be recovered afterward. Thus, we obtain the error estimate for the original solution as follows.

THEOREM 4.5. *Let $u(\cdot)$ and $\{U_k\}_{k=0}^m$ be the solutions to* (8) *and its numerical approximation* (24) *based on the backward Euler and POD-based Galerkin schemes, respectively. Then for sufficiently small Δt, there exists a constant $C' \ge 0$ depending on β, κ, $\hat{u}$, d, S_l, $\vec{V}$, $\vec{P}$, and T but independent of r, Δt, h, and m such that*

$$\frac{1}{m}\sum_{k=1}^{m} ||U_k - u(t_k)||_{L^2}^2 \le C'\left((\Delta t)^2 + \sum_{l=r+1}^{m}\lambda_l + h^4\right). \tag{39}$$

Proof. Let C be the constant appearing in Theorem 4.4; see (31). Recall that

$$u(t_k) = \hat{u}(t_k) + \bar{u}(t_k) = \hat{u}(t_k) - \int_0^{t_k}\int_{\mathbb{T}^n} S_l\left|\vec{P} + \nabla\hat{u}(\vec{x},t)\right| d\mathbf{x}dt \tag{40}$$

and

$$U_k = \hat{U}_k + \bar{U}_k = \hat{U}_k - \frac{1}{2}\Delta t\sum_{i=1}^{k}\left[\mathbb{I}\left(S_l|\vec{P} + \nabla\hat{U}_{i-1}(\vec{x})|\right) + \mathbb{I}\left(S_l|\vec{P} + \nabla\hat{U}_i(\vec{x})|\right)\right], \tag{41}$$

where $\mathbb{I}(\cdot)$ is the numerical integrator by the composite trapezoidal rule. We also denote $\tilde{U}_i = -\mathbb{I}\big(S_l|\vec{P}+\nabla\hat{U}_i(\vec{x})|\big)$, $i=0,1,\ldots,k$. Then, we obtain

$$\frac{1}{m}\sum_{k=1}^{m}\big|\big|U_k-u(t_k)\big|\big|_{L^2}^2 \le \frac{2}{m}\sum_{k=1}^{m}\big|\big|\hat{U}_k-\hat{u}(t_k)\big|\big|_{L^2}^2+\frac{2}{m}\sum_{k=1}^{m}\big|\bar{u}(t_k)-\bar{U}_k\big|^2. \tag{42}$$

From Theorem 4.4, we get an estimate for the first part on the right-hand side (RHS) of the inequality (42). In the sequel, we shall estimate the second part of the RHS of the inequality (42). We consider the following decomposition

$$\begin{aligned}\sum_{k=1}^{m}\big|\bar{u}(t_k)-\bar{U}_k\big|^2 \le{}& 2\sum_{k=1}^{m}\left|\bar{u}(t_k)-\sum_{j=1}^{k}\frac{1}{2}\Delta t\big(\bar{u}_t(t_{j-1})+\bar{u}_t(t_j)\big)\right|^2\\ &+2\sum_{k=1}^{m}\left|\sum_{j=1}^{k}\frac{1}{2}\Delta t\big(\bar{u}_t(t_{j-1})+\bar{u}_t(t_j)\big)-\sum_{j=1}^{k}\frac{1}{2}\Delta t\big(\tilde{U}_{j-1}+\tilde{U}_j\big)\right|^2.\end{aligned} \tag{43}$$

The first term on the RHS of the inequality (43) is bounded by

$$\begin{aligned}&2\sum_{k=1}^{m}\left|\bar{u}(t_k)-\sum_{j=1}^{k}\frac{1}{2}\Delta t\big(\bar{u}_t(t_{j-1})+\bar{u}_t(t_j)\big)\right|^2\\ &\quad\le 2(\Delta t)^2\sum_{k=1}^{m}\left|\sum_{j=1}^{k}\Big(\frac{\bar{u}(t_j)-\bar{u}(t_{j-1})}{\Delta t}-\frac{1}{2}\big(\bar{u}_t(t_{j-1})+\bar{u}_t(t_j)\big)\Big)\right|^2\\ &\quad\le 2(\Delta t)^2\sum_{k=1}^{m}k\sum_{j=1}^{k}\left|\frac{\bar{u}(t_j)-\bar{u}(t_{j-1})}{\Delta t}-\frac{1}{2}\big(\bar{u}_t(t_{j-1})+\bar{u}_t(t_j)\big)\right|^2\\ &\quad\le 2(\Delta t)^2\sum_{k=1}^{m}k\frac{(\Delta t)^2}{12}\int_0^T\big|\big|\bar{u}_{ttt}(t)\big|\big|_{L^2}^2\le\frac{T^2}{12}(\Delta t)^2\int_0^T|\bar{u}_{ttt}(t)|_{L^2}^2.\end{aligned} \tag{44}$$

The second term on the RHS of (43) is bounded by

$$\begin{aligned}&2\sum_{k=1}^{m}\left|\sum_{j=1}^{k}\frac{1}{2}\Delta t\big(\bar{u}_t(t_{j-1})+\bar{u}_t(t_j)\big)-\sum_{j=1}^{k}\frac{1}{2}\Delta t\big(\tilde{U}_{j-1}+\tilde{U}_j\big)\right|^2\\ &\quad\le(\Delta t)^2\sum_{k=1}^{m}k\sum_{j=1}^{k}\Big(\big|\bar{u}_t(t_{j-1})-\tilde{U}_{j-1}\big|^2+\big|\bar{u}_t(t_j)-\tilde{U}_j\big|^2\Big)\\ &\quad=(\Delta t)^2\sum_{k=1}^{m}k\sum_{j=1}^{k}\Bigg(\bigg|\int_{\mathbb{T}^n}S_l\Big|\vec{P}+\nabla\hat{u}(\vec{x},t_{j-1})d\mathbf{x}-\mathbb{I}\big(S_l|\vec{P}+\nabla\hat{U}_{j-1}(\vec{x})|\big)\bigg|^2\\ &\qquad\qquad+\bigg|\int_{\mathbb{T}^n}S_l\Big|\vec{P}+\nabla\hat{u}(\vec{x},t_j)d\mathbf{x}-\mathbb{I}\big(S_l|\vec{P}+\nabla\hat{U}_j(\vec{x})|\big)\bigg|^2\Bigg)\\ &\quad\le 2(\Delta t)^2m^2\sum_{j=0}^{m}\Bigg(S_l^2\int_{\mathbb{T}^n}\big|\nabla\hat{u}(\vec{x},t_j)-\nabla\hat{U}_j(\vec{x})\big|^2d\mathbf{x}\\ &\qquad\qquad+\left|\int_{\mathbb{T}^n}S_l\big|\vec{P}+\nabla\hat{U}_j(\vec{x})\big|d\mathbf{x}-\mathbb{I}\big(S_l|\vec{P}+\nabla\hat{U}_j(\vec{x})|\big)\right|^2\Bigg)\\ &\quad\le 2T^2S_l^2\sum_{j=0}^{m}\Big(||\hat{u}(t_j)-\hat{U}_j||_H^2+C_4h^4\Big)\le 2T^2S_l^2\left(\frac{3\beta}{\kappa}\sum_{l=r+1}^{m}\lambda_l+mC_4h^4\right),\end{aligned} \tag{45}$$

where C_4 comes from the error estimate of the trapezoidal rule and depends on third-order derivatives of the solution with respect to physical variables, and the last inequality follows from the fact that $\hat{u}(t_j) - \hat{U}_j \in S^r$ and norms are equivalent in a finite dimensional space. Substituting the estimate results (31), (44), and (45) into (42), we obtain

$$\frac{1}{m}\sum_{k=1}^{m} ||U_k - u(t_k)||_{L^2}^2 \leq C'\left((\Delta t)^2 + \sum_{l=r+1}^{m} \lambda_l + h^4\right), \tag{46}$$

where $C' \geq 0$ depends on β, κ, $\hat{u}$, d, S_l, $\vec{V}$, $\vec{P}$, and T but is independent of r, Δt, h, and m. ∎

Theorems 4.4 and 4.5 show that the errors depend on the sum of eigenvalues of the correlation matrix K (associated with the solution snapshots) that are not included in the POD basis. Thus, the decay of eigenvalues provides a quantitative guidance for choosing the number of the POD basis. A typical approach is to choose r so that $\sum_{l=r+1}^{m} \lambda_l \leq \min\{(\Delta t)^2, h^4\}$.

5. Numerical results.

We shall perform numerical experiments to test the performance and accuracy of the proposed method. The numerical experiments are all carried out on a laptop with a 2.5GHz quad-core Intel Core i7 processor and 8GB RAM. The Python codes are published on GitHub.[1] We consider the following viscous G-equation on $[0,1]^2$ with planar initial condition:

$$\begin{cases} G_t + \vec{V}\cdot\nabla G + S_l|\nabla G| = dS_l\Delta G & \text{in } [0,1]^2 \times (0,T), \\ G(\vec{x},0) = \vec{P}\cdot\vec{x} & \text{on } [0,1]^2 \times \{t=0\}, \end{cases} \tag{47}$$

where $\vec{x} = (x,y)$, $\vec{P}$ is the flame front propagation direction, and the assumption $G(\vec{x}+\vec{z},t) = G(\vec{x},t) + \vec{P}\cdot\vec{z}$ is used. The setting of the fluid velocity $\vec{V}(\vec{x})$ is an important issue in turbulent combustion modeling. We first consider a steady incompressible cellular flow,

$$\vec{V}(\vec{x}) = (V_1, V_2) = \nabla^{\perp}\mathcal{H} = (-\mathcal{H}_y, \mathcal{H}_x), \quad \mathcal{H} = \frac{A}{2\pi}\sin(2\pi x)\sin(2\pi y), \tag{48}$$

where A is the amplitude of the velocity filed. In the following numerical experiments on this steady flow (48), we choose $A = 4.0$, $S_l = 1$, and $\vec{P} = (1,0)$. In our comparison, the finite difference solution refers to the solution obtained by the reference method to solve (47); see section A.1 in the appendix for more details. The POD solution refers to the one obtained by our method, i.e., the solution is represented by the POD basis. We choose the method of snapshot [43] to construct the POD basis. Details were introduced in section 3.2.

In the finite difference scheme, we partition the physical domain $[0,1]^2$ into $(N_h+1)\times(N_h+1)$ grids with mesh size $h = 1/N_h$ and $N_h = 80$. For the time discretization, we choose time step $\Delta t = 1/N_t$ with $N_t = 1000$. As such, solving the viscous G-equation from 0s to Ts will generate $2TN_t + 1$ snapshots.

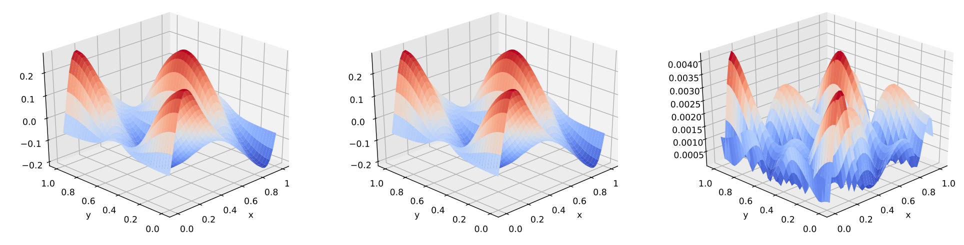
In the POD scheme, the algorithm in section 3.2 is first applied to construct a set of POD basis, denoted by $S = \{\psi_1, \ldots, \psi_r\}$, $r \leq \min\{(N_h+1)\times(N_h+1), 2TN_t+1\}$. In practice, we choose the number of the POD basis r to be the smallest integer such

[1]https://github.com/Harry970804/POD_G-Equation.

that $\sum_{l=r+1}^{N_{snap}} \lambda_l / \sum_{l=1}^{N_{snap}} \lambda_l \leq e_{POD}$, where λ_l's are the eigenvalues of the covariance matrix of the solution snapshots, N_{snap} is the number of solution snapshots, and the error threshold e_{POD} is chosen to be 0.001 in our numerical experiments. We choose the inner product in H_1 norm in computing the POD basis. Usually, r is assumed to be much smaller than the degree of freedom in the physical space discretization. In each of the following experiments, r will be given in detail later. But we want to point out in advance that, under most of the parameter settings with which we have experimented, a small r (around 10) will be good enough to capture true solutions well with a high accuracy. With the POD basis, we further apply the scheme in section 3.3 to get the POD solution. The time discretization for the POD scheme is $\Delta t_{POD} = 1/1000$.

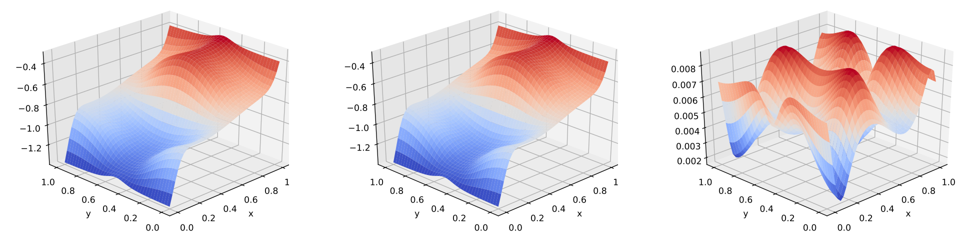
5.1. Test of the POD basis within the same computational time. In the first numerical experiment, we choose $d = 0.1$ and solve the viscous G-equation (47) from time $t = 0$ to 1s using the reference numerical scheme and obtain mean-free solution snapshots. Then, the POD basis functions are constructed from the entire snapshots. In this case, the number of basis functions is $r = 4$. Finally, we compute the (mean-free) POD solution on the same time period.

We compare the mean-free parts obtained by two difference methods at $T = 1.0$. Figure 2 shows that the POD solution agrees well with the reference solution. Moreover, based on the mean-free POD solution, we recover the numerical solution of the G-equation according to (24). In Figure 3, we show the recovered solution using our POD method and the reference solution. We can see the good performance of the POD method.



(a) Reference solution. (b) POD solution. (c) Absolute difference.

FIG. 2. *Mean-free component of the solution at $T = 1$ with $d = 0.1$.*



(a) Reference solution. (b) POD solution. (c) Absolute difference.

FIG. 3. *Solution of the viscous G-equation at $T = 1$ with $d = 0.1$.*

Table 1
The relative errors between POD solution and reference solution.

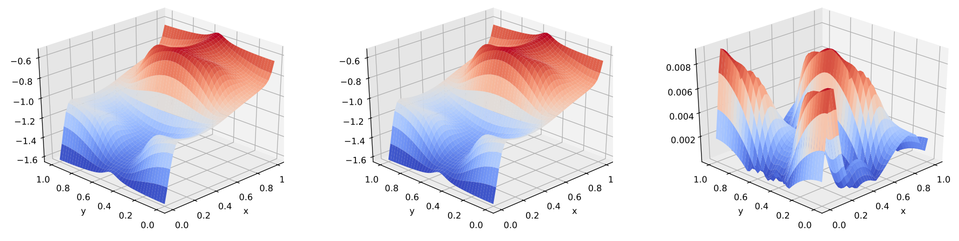
d	0.01	0.02	0.03	0.04	0.05
Rela. error, Mean-free	0.038844	0.069712	0.025673	0.021257	0.017649
Rela. error, Mean	0.020031	0.023782	0.003230	0.002196	0.002320
Rela. error, Recovered	0.020316	0.026285	0.005110	0.003020	0.003349
d	0.06	0.07	0.08	0.09	0.1
Rela. error, Mean-free	0.016216	0.014977	0.014998	0.013786	0.013793
Rela. error, Mean	0.003074	0.003258	0.004991	0.006812	0.007230
Rela. error, Recovered	0.004840	0.004997	0.005115	0.007431	0.007085

We continue our experiment by computing (47) with different choices of d from 0.01 to 0.1. We compare the errors between the POD method and the reference method in computing the mean-free part of the solution, the mean of the solution, and the recovered solution, respectively. Table 1 shows the relative error in L^2 norm of these results with different choices of d's, where "Rela. error, Mean-free," "Rela. error, Mean," and "Rela. error, Recovered" mean relative error of the mean-free part of the solution, relative error of the mean of the solution, and relative error of the recovered solution, respectively. We can see that (1) the relative error of the mean-free component is small, which shows the good performance of the POD method; (2) the relative error of the mean part is much smaller than that of the mean-free part, which we think is due to the cancellation of the error in computing the mean; and (3) the relative error of the recovered solution is smaller than that of the mean-free part since the recovered solution usually has a larger magnitude than the mean-free part.

For the dimension of the subspace spanned by the POD basis, we observe that under a fixed threshold of e_{POD}, in general, it will increase as d decreases. But it still remains small and it further implies the efficiency of the POD method. In all the experiments we have reported, the dimensions vary in the range from 4 to 10.

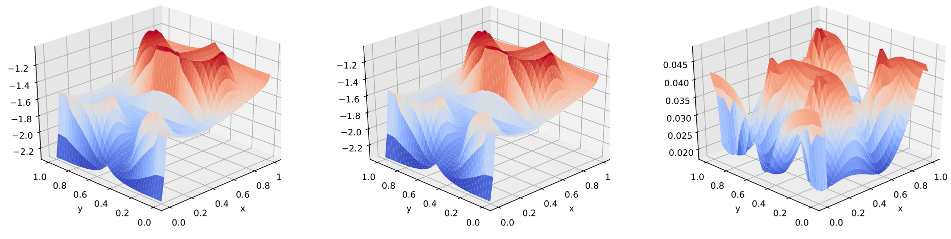
In Figures 4 and 5, we show the comparison between the recovered solutions obtained using the POD method and the reference solutions for $d = 0.05$ and $d = 0.01$, respectively. Though we observe some sharp layers appearing in the solution as d decreases, the POD basis can capture these layered structures and give accurate numerical results.

These results show that the POD method can capture the low-dimensional structures in the regular solutions of the viscous G-equations and provide an efficient model reduction method to approximate the solutions. We remark that when d is extremely small or even $d = 0$ the solutions of G-equations are not smooth and cannot



(a) Reference solution. (b) POD solution. (c) Absolute difference.

Fig. 4. *Solution of the viscous G-equation at* $T = 1$ *with* $d = 0.05$.



(a) Reference solution. (b) POD solution. (c) Absolute difference.

FIG. 5. *Solution of the viscous G-equation at $T = 1$ with $d = 0.01$.*

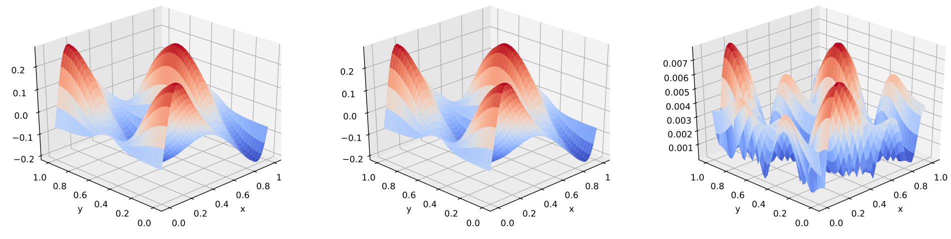
be computed by the POD-based Galerkin method. We will exploit some features of the discontinuous Galerkin method and develop new POD methods to address this challenge in our future work.

5.2. Test of the POD basis for longer time computations. In this numerical experiment, we first solve the viscous G-equation (47) from $T = 0$ to $T = 1$ using the finite difference scheme to construct the POD basis. Then, we adopt the POD method to solve the equation on a longer time from $T = 0$ to $T = 2$ using that POD basis.

We conduct the experiments for $d = 0.01, 0.02, \dots, 0.1$. The number of POD basis functions is fixed to be six across all different d.

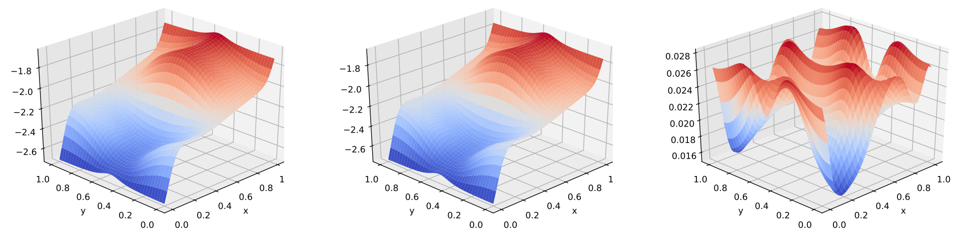
In Figures 6 and 7, the results for $d = 0.1$ are reported in detail. The POD solution agrees well with the finite difference solution, even though the basis is extracted from the first half of the whole time horizon. It is also worth pointing out that the profiles in Figures 2 and 6, i.e., the mean-free components of the solutions, are almost the same, while the profile in Figure 7 can be viewed approximately as a downward shift of the profile in Figure 3 by about 1.4 units. These interesting results show that the POD method allows us to capture the stationary solution to the G-equation, which demonstrates its ability to track the long-term behavior of the system. Table 2 shows relative errors for other choices of d.

In addition, for $d = 0.05$ and $d = 0.01$, we plot their recovered solutions obtained from both methods in Figures 8 and 9 at $T = 2$, respectively. Again, we find that the POD method performs well for those two settings. Moreover, the profiles in Figures 8 and 9 can also be viewed approximately as downward shifts of the profiles in Figures 4 and 5, respectively. The smaller the d is, the bigger shift that will be observed.



(a) Reference solution. (b) POD solution. (c) Absolute difference.

FIG. 6. *Mean-free component of the solution at $T = 2$ with $d = 0.1$.*

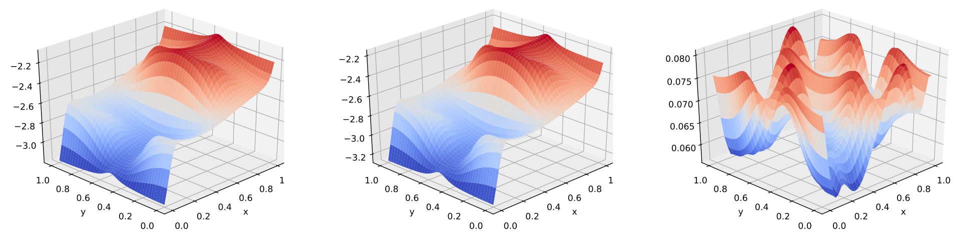


(a) Reference solution. (b) POD solution. (c) Absolute difference.

FIG. 7. *Solution of the viscous G-equation at $T = 2$ with $d = 0.1$.*

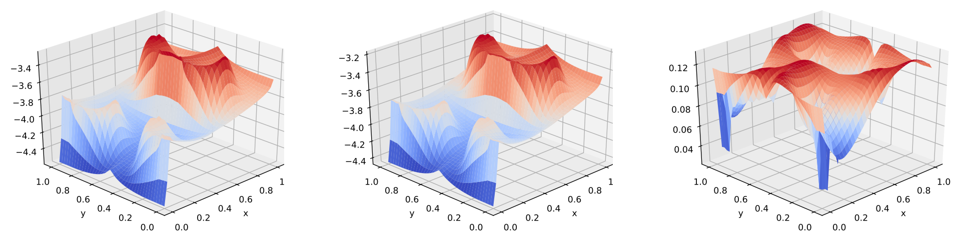
TABLE 2
The relative errors between POD solution and reference solution at $T = 2$.

d	0.01	0.02	0.03	0.04	0.05
Rela. error, Mean-free	0.082762	0.042995	0.029778	0.047371	0.042206
Rela. error, Recovered	0.028087	0.020967	0.019781	0.033270	0.026356
d	0.06	0.07	0.08	0.09	0.1
Rela. error, Mean-free	0.038563	0.03471	0.031951	0.029692	0.027939
Rela. error, Recovered	0.020541	0.016758	0.013963	0.012019	0.010648



(a) Reference solution (b) POD solution. (c) Absolute difference.

FIG. 8. *Solution of the viscous G-equation at $T = 2$ with $d = 0.05$.*



(a) Reference solution. (b) POD solution. (c) Absolute difference.

FIG. 9. *Solution of the viscous G-equation at $T = 2$ with $d = 0.01$.*

5.3. Test of the POD basis for different parameters. In this subsection, we investigate the robustness of the POD basis across different diffusion parameter d's. Specifically, we build the POD basis from the solution snapshots of the viscous G-equation with the diffusivity d_0. Then, we use that precomputed POD basis to compute solutions of the viscous G-equation with other d's.

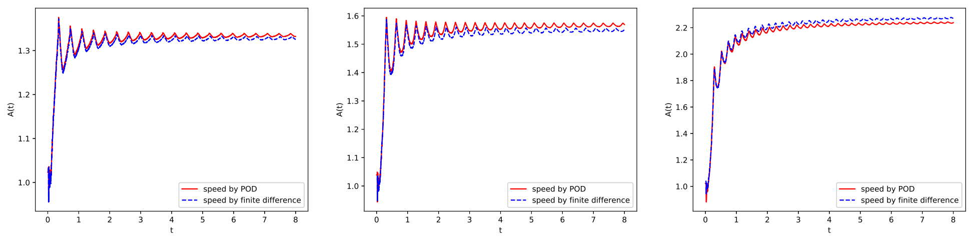
In our numerical experiment, we solve the viscous G-equation from $T = 0$ to $T = 1$ with $d_0 = 0.05$ to build the POD basis. In Table 3, we show the relative errors between the POD solutions and the reference solutions to the viscous G-equation with different d. One can find that the POD basis provides good approximations to the solution of the viscous G-equation when d is chosen from $d = 0.01$ to $d = 0.1$. The closer d is to d_0, the smaller the error will be. The fact that the precomputed POD basis can be used to compute solutions associated with different parameters with high accuracy is of great importance in practical computations.

5.4. Compute the turbulent flame speed S_T. Since the POD method is very efficient to approximately solve the viscous G-equation, we apply it to compute the turbulent flame speed S_T, which is defined in (10). In this experiment, we compare the numerical results of the turbulent flame speeds obtained by two methods. First of all, for each parameter d, we compute the solution snapshots from $T = 0$ to $T = 1$ to extract the first six POD basis functions. Then, with the POD basis, we solve the viscous G-equation for a much longer time, from $T = 0$ to $T = 8$, to obtain the approximated burned area $\mathcal{A}(t)$. Finally, we show the growth rate of the $\mathcal{A}(t)$ over time t in Figure 10. We find that the results obtained by our method agree well with the reference method. These results show that the POD basis can be used to track the long-time evolution of the turbulent flame speed. We refer the interested reader to the theoretical results on the long-time behavior of solutions to Hamilton–Jacobi equations or quasi-linear parabolic equations; see, e.g., [33, 5, 41]. Our results provide an interesting numerical illustration of these theoretical results.

5.5. Comparison of the computational time. In Table 4 we compare the CPU time of two methods when solving the viscous G-equation from time $T = 0$

TABLE 3
The relative errors between POD solutions and reference solutions.

d	0.01	0.02	0.03	0.04	0.05
Rela. error, Recovered	0.056233	0.030934	0.014142	0.008826	0.004398
d	0.06	0.07	0.08	0.09	0.1
Rela. error, Recovered	0.018682	0.022199	0.024353	0.025569	0.025963



(a) $d = 0.1$. (b) $d = 0.05$. (c) $d = 0.01$.

FIG. 10. *Numerical results of the turbulent flame speed obtained by different methods.*

TABLE 4
CPU time of two methods in solving the viscous G-equation from $T = 0$ to 1 with different d.

d	0.01	0.02	0.03	0.04	0.05
POD method	1.438894	1.296282	0.910010	0.945209	0.929022
Reference method	480.6434	474.2021	483.5336	482.6826	480.4273
d	0.06	0.07	0.08	0.09	0.1
POD method	0.873965	0.940874	0.906758	0.857222	0.786591
Reference method	488.6499	476.8122	487.3498	483.3396	476.7582

to $T = 1$ with different d. The unit of the computational time is second. One can see that computational cost of the POD method is far less than the finite difference method, which shows that model reduction method is very useful in many applications. Besides, one can find that the CPU time of the POD method slightly increases when d decreases. This is due to the fact that we are using more POD basis for small d in order to achieve the same POD truncation error threshold $e_{POD} = 0.1\%$.

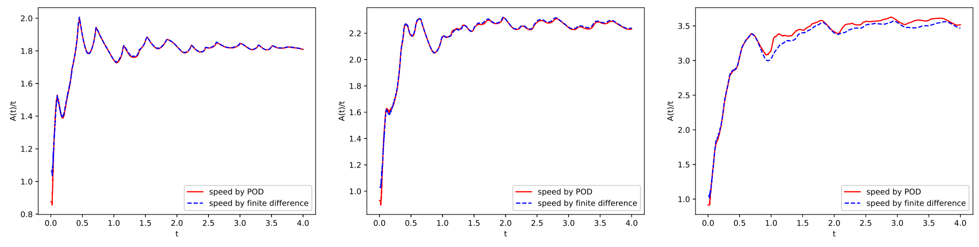
We observe that about 90% of CPU time in the POD method is used to compute the nonlinear term (see $\mathbf{f}_k$ in (22)), which may deteriorate the performance of the POD method for more challenging problems, such as 3D problems or convection-dominated problems. However, one can apply the discrete empirical interpolation method [10] to maintain the efficiency of the POD method.

5.6. Test of the POD basis for a time-periodic fluid velocity. We consider a one-parameter family of time-dependent periodic cellular flow

$$\vec{V}(\vec{x},t) = \big(V_1, V_2\big) = A\big(\cos(2\pi y), \cos(2\pi x)\big) + A\theta\cos(2\pi t)\big((\sin(2\pi y), \sin(2\pi x)\big), \tag{49}$$

where A is the amplitude of the velocity field. The first component of (49) is a steady cellular flow, while the second term of (49) is a time-periodic perturbation controlled by $\theta > 0$ that introduces an increasing amount of disorder in the flow trajectories as θ increases. In the experiments reported in Figure 11, we fix $A = 4$, $\theta = 1.0$ and test on three different d's. For each d, the POD basis is extracted from the reference solution from time $T = 0$ to $T = 1$. Here, the numbers of POD basis functions are $r = 11, 14, 27$ for $d = 0.1, 0.05, 0.01$, respectively. Then using the POD basis, we compute the POD solution from time $T = 0$ to $T = 4$. The turbulent flame speeds $\mathcal{A}(t)/t$ obtained by the POD method are compared with the reference solutions.

The relative errors for the burning speed are basically negligible. These results justify that the POD method is still efficient in solving viscous G-equation with the



(a) $d = 0.1$. (b) $d = 0.05$. (c) $d = 0.01$.

FIG. 11. *Turbulent flame speeds obtained by different methods for the time-dependent periodic cellular flow.*

time-periodic cellular flow. Compared with the results in Figure 10, the turbulent flame speed in Figure 11 has different patterns, which is caused by the mixing and chaotic features of the fluid velocity (49).

We should point out that the strategy to choose the snapshots is a crucial point in the POD method. More generally, the POD basis obtained by minimizing the projection error over a set of snapshots chosen a priori, e.g., (20) may not be able to capture the whole dynamics of equations with time-periodic or chaotic flows. Therefore, we propose an adaptive strategy to enrich the POD basis functions, which was also used in [12]. We provide the step-by-step algorithm of the adaptive strategy in section A.4. We consider the viscous G-equation with the time-dependent velocity field (49) to demonstrate the main idea, where $A = 4.0$, $\theta = 1.0$, $d = 0.1$.

We first compute the snapshots from time $T = 0$ to $T = 0.5$ and extract a set of POD basis functions from these snapshots. In this experiment, we obtain $r = 7$ basis functions, which are called the initial POD basis functions. Notice that the time period of the velocity field (49) is one. Thus, the initial POD basis functions may not be able to capture the whole time-varying dynamics of the viscous G-equation.

In our adaptive strategy, we will check the approximation ability of the current POD basis after every ΔT time period. At each checking time, we use the current POD solution as initial data and apply the finite difference scheme to compute a short period of time, say, $N\Delta t$, to obtain N new snapshots. Here, Δt is the time step in the finite difference scheme and $N > 0$ is an integer. We require $N\Delta t \ll \Delta T$. Then, we project the N snapshots onto the current POD basis and compute the norm of the projection error. If the norm of the projection error is bigger than a prescribed threshold, we enrich the POD basis from the information in the projection error.

In this experiment, we choose $\Delta T = 0.5$, $\Delta t = 0.001$, and $N = 50$. We observe that every time, one or two basis functions will be added. At the end of the experiment, there are $r = 17$ basis functions. We compare the performances of the fixed-basis POD method with the adaptive POD method, in terms of estimating the flame speed. Figure 12 clearly shows that the adaptive strategy indeed improves the performance of the POD method and obtains a more accurate estimation for the flame speed. More specifically, at the final time $T = 4$, the relative error between the speed calculated from the adaptive POD method and that from the finite difference method is 1.0676%, while the relative error for the fixed-basis POD method is 5.6648%. Meanwhile, the adaptive method still achieves high computational efficiency, as the adaptive method is

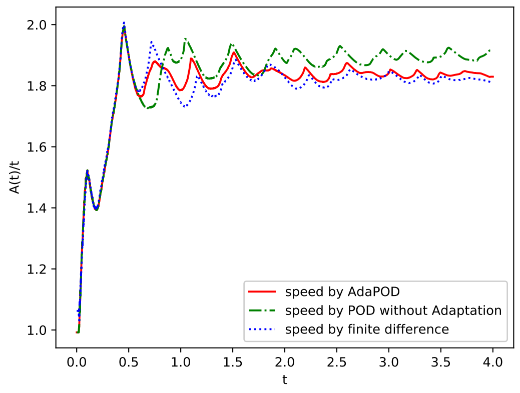


FIG. 12. *A comparison of the POD method with a fixed basis strategy and an adaptive strategy.*

running the POD scheme for most time steps and only uses the finite difference scheme for approximately 10% of the time steps. The CPU time for solving the equation from time $t = 0$ to $t = 4$ with the finite difference method is 2598.83 seconds, while for the adaptive POD method, it only takes 249.48 seconds.

Notice that Figure 12 still shows a gap between the finite difference solution and the adaptive POD solution. The reason is that we only check and enrich the POD basis every ΔT time period and we may lose a small amount of solution information between any two checking times. How to choose ΔT is important in the adaptivity of the POD method. To address this issue, one needs to obtain some a posteriori error estimate for the solution obtained by the POD method. We will study this issue in our subsequent research.

5.7. Test of the POD method for curvature G-equations. To further investigate the performance of the POD method, we solve the curvature G-equations (5) using the POD basis. Specifically, we consider the curvature G-equation with the periodic velocity field (48) and planar initial condition. If we write $G(\vec{x}, t) = \vec{P} \cdot \vec{x} + u(\vec{x}, t)$, then $u(\vec{x}, t)$ is spatially periodic and satisfies the following periodic initial value problem:

(50)
$$\begin{cases} u_t + \vec{V} \cdot (\vec{P} + \nabla u) + S_l|\vec{P} + \nabla u| = dS_l|\vec{P} + \nabla u|\Big(\nabla \cdot \frac{\vec{P}+\nabla u}{|\vec{P}+\nabla u|}\Big) & \text{in } \mathbb{T}^n \times (0, \infty), \\ u(\vec{x}, 0) = 0 & \text{on } \mathbb{T}^n \times \{t = 0\}, \end{cases}$$

where $\mathbb{T}^n = [0, 1]^n$. Again, we decompose u into the mean-free part $\hat{u}$ and the mean $\bar{u}$. The derivation is almost same as what we have done in section 3.1. The evolution equations for $\hat{u}$ and $\bar{u}$ are as follows:

$$(51)\qquad \begin{cases} \hat{u}_t + \vec{V} \cdot \nabla\hat{u} - dS_l\big|\vec{P} + \nabla\hat{u}\big|\bigg(\nabla \cdot \dfrac{\vec{P} + \nabla\hat{u}}{|\vec{P} + \nabla\hat{u}|}\bigg) \\ \quad + \displaystyle\int_{\mathbb{T}^n} dS_l\big|\vec{P} + \nabla\hat{u}\big|\bigg(\nabla \cdot \dfrac{\vec{P} + \nabla\hat{u}}{|\vec{P} + \nabla\hat{u}|}\bigg)d\vec{x} \\ \quad +S_l\big|\vec{P} + \nabla\hat{u}\big| - \displaystyle\int_{\mathbb{T}^n} S_l\big|\vec{P} + \nabla\hat{u}\big|d\vec{x} + \vec{V} \cdot \vec{P} = 0 & \text{in } \mathbb{T}^n \times (0, \infty), \\ \bar{u}_t = -\displaystyle\int_{\mathbb{T}^n} S_l\big|\vec{P} + \nabla\hat{u}(\vec{x}, t)\big|d\vec{x} + \int_{\mathbb{T}^n} dS_l|\vec{P} \\ \quad +\nabla\hat{u}|\bigg(\nabla \cdot \dfrac{\vec{P} + \nabla\hat{u}}{|\vec{P} + \nabla\hat{u}|}\bigg)d\vec{x} & \text{on } t \in (0, \infty), \\ \hat{u}(\vec{x}, 0) = 0 & \text{on } \mathbb{T}^n \times \{t = 0\}, \\ \bar{u}(0) = 0. \end{cases}$$

The implementation of the POD method is the same as we have done for the viscous G-equation in section 3. First of all, we use a reference numerical method to solve the curvature G-equation (see section A.2) and construct the POD basis by using the mean-free part of the solution snapshots. Then, we use the POD-based Galerkin method to solve the evolution equation for $\hat{u}$ in (51). The corresponding weak formulation is

$$(52)\qquad \langle \hat{u}_t, \psi\rangle + a(\hat{u}, \psi) + \langle F(\hat{u}), \psi\rangle = \langle -\vec{P} \cdot \vec{V}, \psi\rangle \ \forall\psi \in H,$$

where the bilinear form $a(\hat{u}, \psi) = \int_{\mathbb{T}^n}(\vec{V} \cdot \nabla\hat{u})\psi d\vec{x}$ and the nonlinear part $F(\hat{u}) = S_l\big|\vec{P} + \nabla\hat{u}\big| - \int_{\mathbb{T}^n} S_l\big|\vec{P} + \nabla\hat{u}\big|d\vec{x} - dS_l|\vec{P} + \nabla\hat{u}|\big(\nabla \cdot \frac{\vec{P}+\nabla\hat{u}}{|\vec{P}+\nabla\hat{u}|}\big) + \int_{\mathbb{T}^n} dS_l|\vec{P} + \nabla\hat{u}|\big(\nabla \cdot$

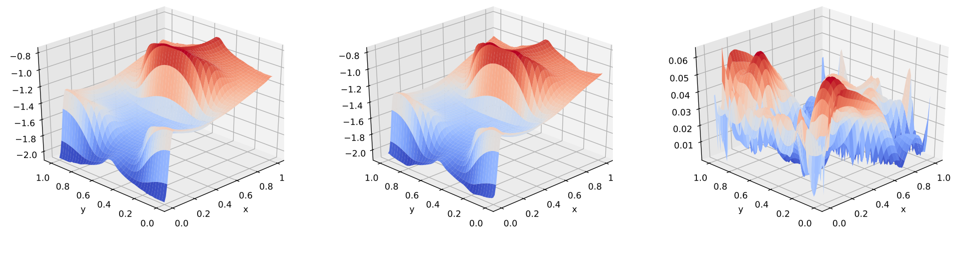

(a) Reference solution. (b) POD solution. (c) Absolute difference.

FIG. 13. *Solution of the curvature G-equation at $T = 1$ with $d = 0.1$.*

$\frac{\vec{P}+\nabla\hat{u}}{|\vec{P}+\nabla\hat{u}|})d\vec{x}$. Finally, we solve a nonlinear equation system to compute the evolution equation for $\hat{u}$. The evolution equation for the mean $\bar{u}$ can be solved subsequently.

In our numerical experiment, we use the POD method to solve (51) with the steady flow (48). We choose $A = 4.0$, $d = 0.1$, $S_l = 1$, and $\vec{P} = (1, 0)$. We first construct $r = 6$ POD basis functions from the reference solution snapshots from time $T = 0$ to $T = 1$. Then, we compute the mean-free POD solution on the same time interval using the POD-based Galerkin method. Meanwhile, we compute the mean $\bar{u}$ solution. Finally, we can recover the solution u. The relative error of the recovered solutions between the POD method and the reference method is 0.0220. Figure 13 shows the recovered solutions using the POD method and the reference method. We find that the POD method still performs well in solving the curvature G-equations.

If we compare Figure 13 with Figure 3, we can see that with the same diffusion parameter $d = 0.1$, the solution of the curvature G-equation is not very smooth. It is expected that a smaller d in the curvature G-equation will bring more challenges in the method design and convergence analysis, which will be studied in our future research.

6. Conclusion.

We have proposed an efficient model reduction method to solve viscous G-equations, which have been very popular field models in the combustion and physics literature for studying turbulent flame propagation. We constructed the POD basis based on learning the solution information from the snapshots. Then, we applied the Galerkin project method to solve the viscous G-equation and smooth solutions of the curvature G-equation by using the POD basis. We provided rigorous error analysis for our numerical methods based on a decomposition strategy, where we decomposed the solution into a mean part and a mean-free part. We showed through numerical experiments that our methods can accurately compute the various G-equations with significant computational savings. In addition, we found that the POD basis allows us to compute long-time solution of the various G-equations. Thus, we can compute the corresponding turbulent flame speeds in cellular flows. In our future work, we plan to study turbulent flame speeds of G-equations in 3D spatially or spatiotemporally periodic vortical flows. In addition, we will further investigate the adaptivity for the POD method and using the POD method to solve curvature G-equations.

Appendix A.

A.1. A reference method to solve viscous G-equations.

We first apply the finite difference scheme proposed in [28] to solve the viscous G-equation from time 0 to T seconds on the domain $D = [0, 1] \times [0, 1]$ to get the snapshots. Specifically, we employ a higher-order HJ-WENO scheme and a TVD-RK scheme in spatial and time discretization, respectively; see [35, 17, 21, 34] for more details of these schemes.

For a small d, the viscous G-equation is convection dominated and it should be treated like a hyperbolic equation. The forward Euler time discretization is given by

$$\frac{G^{n+1}-G^n}{\Delta t} + H^n(G_x^-, G_x^+, G_y^-, G_y^+) - dS_l \Delta G^n = 0, \tag{53}$$

where G_i^- and G_i^+ denote the left and right discretization of G_i in the WENO5 scheme [21]. H is a consistent and monotone numerical Hamiltonian. Then, we have

$$H(G_x^-, G_x^+, G_y^-, G_y^+) = V_1 G_x^{vel} + V_2 G_y^{vel} + S_l\sqrt{(G_x^{nor})^2 + (G_y^{nor})^2}, \tag{54}$$

where the upwinding scheme and the Godunov scheme are applied for the convection term and the nonlinear term separately [34],

$$G_x^{vel} = \begin{cases} G_x^- & \text{if } V_1 > 0, \\ G_x^+ & \text{if } V_1 < 0, \end{cases} \tag{55}$$

$$G_y^{vel} = \begin{cases} G_y^- & \text{if } V_2 > 0, \\ G_y^+ & \text{if } V_2 < 0, \end{cases} \tag{56}$$

$$(G_x^{nor})^2 = \begin{cases} (G_x^-)^2 & \text{if } V_1 > S_l, \\ \max(\max(G_x^-, 0)^2, \min(G_x^+, 0)^2) & \text{if } |V_1| \le S_l, \\ (G_x^+)^2 & \text{if } V_1 < -S_l, \end{cases} \tag{57}$$

$$(G_y^{nor})^2 = \begin{cases} (G_y^-)^2 & \text{if } V_2 > S_l, \\ \max(\max(G_y^-, 0)^2, \min(G_y^+, 0)^2) & \text{if } |V_2| \le S_l, \\ (G_y^+)^2 & \text{if } V_2 < -S_l. \end{cases} \tag{58}$$

For the diffusion term, we apply the central difference. For the time discretization, we apply the RK3 scheme [17]. The CFL condition in this case is

$$\Delta t\left(\frac{S_l + |V_1|}{\Delta x} + \frac{S_l + |V_2|}{\Delta y}\right) < 1. \tag{59}$$

When d is large, the time step size for the forward Euler scheme is very small, i.e., $\Delta t = O\big((\Delta x)^2 + (\Delta y)^2\big)$. To alleviate the stringent time step restriction, we introduce the following semi-implicit scheme:

$$\frac{G^{n+1}-G^n}{dt} + \vec{V}\cdot\nabla G^{n+1} + S_l|\nabla G^n| = dS_l\Delta G^{n+1}, \tag{60}$$

where the convection and diffusion terms are discretized by the central difference, and the normal direction term is discretized by the Godunov and the WENO5 scheme. In this case, the CFL condition is

$$\Delta t\left(\frac{S_l}{\Delta x} + \frac{S_l}{\Delta y}\right) < 1. \tag{61}$$

Numerical results in [28] show that the proposed method is efficient in solving G-equations. The limitation is that the computational cost becomes large when one needs to choose a fine mesh to discretize the problem.

A.2. A reference method to solve curvature G-equations. We also apply the finite difference scheme to solve the curvature G-equation (5), which was proposed in [28]. The spatial and the time discretization are the same as for the viscous G-equations in section A.1. The curvature term is given by

$$|\nabla G|\left(\nabla\cdot\frac{\nabla G}{|\nabla G|}\right)=\frac{G_y^2G_{xx}-G_xG_yG_{xy}+G_x^2G_{yy}}{G_x^2+G_y^2}, \tag{62}$$

where each partial derivative is approximated by the central difference. The computation of the numerical Hamiltonian is same as before; see (54)–(58). For the time discretization, we apply the RK3 scheme. The time step restriction is

$$\Delta t\left(\frac{S_l+|V_1|}{\Delta x}+\frac{S_l+|V_2|}{\Delta y}+\frac{2S_ld}{(\Delta x)^2}+\frac{2S_ld}{(\Delta y)^2}\right)<1. \tag{63}$$

When d is large, in order to avoid a small time step size $\Delta t=O((\Delta x)^2)$, the curvature term is decomposed as

$$|\nabla G|\left(\nabla\cdot\frac{\nabla G}{|\nabla G|}\right)=\Delta G-\Delta_\infty G=G_{xx}+G_{yy}-\frac{G_x^2G_{xx}+G_xG_yG_{xy}+G_y^2G_{yy}}{G_x^2+G_y^2}. \tag{64}$$

If we apply the backward Euler scheme on ΔG and forward Euler scheme on $\Delta_\infty G$, then we have the following semi-implicit time discretization scheme:

$$\frac{G^{n+1}-G^n}{dt}+\vec{V}\cdot\nabla G^{n+1}+S_l|\nabla G^n|=dS_l(\Delta G^{n+1}-\Delta_\infty G^n). \tag{65}$$

The convection and diffusion terms are discretized by the central difference, and the normal direction term is discretized by the Godunov and the WENO5 scheme. In this case, the CFL condition is same as (59).

A.3. Model reduction using the POD method. Let X be a Hilbert space equipped with the inner product $\langle\cdot,\cdot\rangle_X$ and $u(\cdot,t)\in X$, $t\in[0,T]$ be the solution of a dynamic system. In practice, we approximate the space X by a linear finite dimensional space V with $\dim(V)=N_{dof}$, where N_{dof} represents the degree of freedom of the solution space and N_{dof} can be extremely large for high-dimensional problem (consider the finite element method or finite difference method as examples). Assume a set of snapshot of solutions, denoted as $\{u(\cdot,t_1),u(\cdot,t_2),\dots,u(\cdot,t_m)\}$, where $t_1,\dots,t_m\in[0,T]$ are different time instances and m is the number of the solution snapshots.

The POD method aims to build a set of orthonormal basis $\{\psi_1(\cdot),\psi_2(\cdot),\dots,\psi_r(\cdot)\}$ with $r\le\min(m,N_{dof})$ that optimally approximates the solution snapshots. More specifically, the POD method constructs the basis by solving the following optimization problem:

$$\min_{\psi_1,\cdots,\psi_r\in X}\frac{1}{m}\sum_{i=1}^{m}\Big\|u(\cdot,t_i)-\sum_{j=1}^{r}\langle u(\cdot,t_i),\psi_j(\cdot)\rangle_X\psi_j(\cdot)\Big\|_X^2\quad s.t.\ \langle\psi_i,\psi_j\rangle_X=\delta_{ij}. \tag{66}$$

In order to solve (66), we consider the eigenvalue problem

$$Kv=\lambda v,$$

where $K \in \mathbb{R}^{m\times m}$ and $K_{ij} = \frac{1}{m}\langle u(\cdot,t_i), u(\cdot,t_j)\rangle_X$ is the snapshot correlation matrix. Let $v_k, k = 1,\dots,n$, be the eigenvectors and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m > 0$ be the positive eigenvalues. It has been shown in [43, 45] that the solution of the optimization problem (66) is given by

$$\psi_k(\cdot) = \frac{1}{\sqrt{\lambda_k}}\sum_{j=1}^{m}(v_k)_j u(\cdot,t_j),\ 1 \leq k \leq r. \tag{67}$$

It can also be shown that the following error formula holds:

$$\frac{1}{m}\sum_{i=1}^{m}\|u(\cdot,t_i) - \sum_{j=1}^{r}\langle u(\cdot,t_i),\psi_j(\cdot)\rangle_X \psi_j(\cdot)\|_X^2 = \sum_{l=r+1}^{m}\lambda_l. \tag{68}$$

Finally, let S^r denote the r-dimensional space spanned by $\{\psi_1(\cdot),\psi_2(\cdot),\dots,\psi_r(\cdot)\}$.

A.4. An adaptive strategy to dynamically enrich the POD basis. In this subsection, we formalize the proposed adaptive strategy for enriching the POD basis in Algorithm 1, where the notation has been defined before.

Algorithm 1 Adaptive strategy for enriching the POD basis.

```
1: Input: POD basis $\mathcal{S} = \{\varphi_i\}$, error threshold $\epsilon > 0$, computational time $T$, period of checking time $\Delta T$, time step of POD or finite difference method $\Delta t$, $N_T = \lceil T/\Delta t\rceil$, $N_{\text{check}} = \lceil \Delta T/\Delta t\rceil$, and snapshots number $N$ (where $N\Delta t \ll \Delta T$). Parameters in the G-equations. Solution at time $t = 0$: $U_0$.
2: for $i = 1 : N_T$ do
3:   if $i > 0$ and ($i$ mod $N_{\text{check}}$) $= 0$ then
4:     Set $\mathcal{U} = \emptyset$.
5:     for $j = 1 : N$ do
6:       Apply one step of the finite difference scheme proposed by [28] (see sections A.1 and A.2) to solve the equation from time $(i + j - 1)\Delta t$ to time $(i + j)\Delta t$.
7:       Denote the finite difference solution at time $\Delta t(i + j)$ by $u_j$.
8:       $\mathcal{U} = \mathcal{U} \cup \{u_j\}$.
9:     end for
10:    Let $P_{\mathcal{S}}$ be the projection onto the space spanned by $\mathcal{S}$. Compute $P_{\mathcal{S}}\mathcal{U} = \{P_{\mathcal{S}}u_j\}$.
11:    Compute the POD basis $\mathcal{S}'$ for $P_{\mathcal{S}}\mathcal{U}$, such that the approximation error (68) is less than $\epsilon$ (see section A.3).
12:    $\mathcal{S} = \mathcal{S} \cup \mathcal{S}'$.
13:  end if
14:  Apply one step of the backward POD-based scheme (22), with basis $\mathcal{S}$, to solve the equation from time $(i - 1)\Delta t$ to time $i\Delta t$.
15:  Denote the solution at time $i\Delta t$ by $U_i$.
16: end for
17: Output: $\{U_t\}$
```

REFERENCES

[1] A. Alla and M. Falcone, *An Adaptive POD Approximation Method for the Control of Advection-diffusion Equations*, in Control and Optimization with PDE Constraints, Springer, New York, 2013, pp. 1–17.

[2] A. Alla and M. Falcone, *A time-adaptive POD method for optimal control problems*, IFAC Proc. Vol., 46 (2013), pp. 245–250.

[3] A. Alla and J. Kutz, *Randomized model order reduction*, Adv. Comput. Math., 45 (2019), pp. 1251–1271.

[4] A. Alla and L. Saluzzi, *A HJB-POD approach for the control of nonlinear PDEs on a tree structure*, Appl. Numer. Math., 155 (2020), pp. 192–207.

[5] G. Barles and P. Souganidis, *Space-time periodic solutions and long-time behavior of solutions to quasi-linear parabolic equations*, SIAM J. Math. Anal., 32 (2001), pp. 1311–1323.

[6] P. Benner, S. Gugercin, and K. Willcox, *A survey of projection-based model reduction methods for parametric dynamical systems*, SIAM Rev., 57 (2015), pp. 483–531.

[7] G. Berkooz, P. Holmes, and J. Lumley, *The proper orthogonal decomposition in the analysis of turbulent flows*, Annu. Rev. Fluid Mech., 25 (1993), pp. 539–575.

[8] J. Borggaard, T. Iliescu, and Z. Wang, *Artificial viscosity proper orthogonal decomposition*, Math. Comput. Model., 53 (2011), pp. 269–279.

[9] K. Brakke, *The Motion of a Surface by Its Mean Curvature*, Princeton University Press, Princeton, NJ, 2015.

[10] S. Chaturantabut and D. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 32 (2010), pp. 2737–2764.

[11] J. Chen, S. Li, and Z. Zhang, *Efficient multiscale methods for the semiclassical Schrödinger equation with time-dependent potentials*, Comput. Methods Appl. Mech. Engrg., 369 (2020), 113232.

[12] M. Cheng, T. Hou, and Z. Zhang, *A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations* II*: Adaptivity and generalizations*, J. Comput. Phys., 242 (2013), pp. 753–776.

[13] B. Cockburn, G. Karniadakis, and C. Shu, *The Development of Discontinuous Galerkin Methods*, in Discontinuous Galerkin Methods, Springer, New York, 2000, pp. 3–50.

[14] P. Embid, A. Majda, and P. Souganidis, *Comparison of turbulent flame speeds from complete averaging and the G-equation*, Phys. Fluids, 7 (1995), pp. 2052–2060.

[15] L. Evans and J. Spruck, *Motion of level sets by mean curvature.* I, J. Differential Geom., 33 (1991), pp. 635–681.

[16] M. Falcone and R. Ferretti, *Semi-Lagrangian Approximation Schemes for Linear and Hamilton–Jacobi Equations*, SIAM, Philadelphia, 2013.

[17] S. Gottlieb and C. Shu, *Total variation diminishing Runge-Kutta schemes*, Math. Comp., 67 (1998), pp. 73–85.

[18] C. Grässle and M. Hinze, *POD reduced-order modeling for evolution equations utilizing arbitrary finite element discretizations*, Adv. Comput. Math., 44 (2018), pp. 1941–1978.

[19] S. Gugercin and A. C. Antoulas, *A survey of model reduction by balanced truncation and some new results*, Internat. J. Control, 77 (2004), pp. 748–766.

[20] J. Hesthaven, G. Rozza, and B. Stamm, *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*, Springer, New York, 2016.

[21] G. Jiang and D. Peng, *Weighted ENO schemes for Hamilton–Jacobi equations*, SIAM J. Sci. Comput., 21 (2000), pp. 2126–2143.

[22] S. Jin and Z. Xin, *Numerical passage from systems of conservation laws to Hamilton–Jacobi equations, and relaxation schemes*, SIAM J. Numer. Anal., 35 (1998), pp. 2385–2404.

[23] D. Kalise and A. Kröner, *Reduced-order minimum time control of advection-reaction-diffusion systems via dynamic programming.*, in Proceedings of the 21st International Symposium on Mathematical Theory of Networks and Systems, 2014, pp. 1196–1202.

[24] D. Kalise and K. Kunisch, *Polynomial approximation of high-dimensional Hamilton–Jacobi–Bellman equations and applications to feedback control of semilinear parabolic PDEs*, SIAM J. Sci. Comput., 40 (2018), pp. A629–A652.

[25] K. Kunisch and S. Volkwein, *Galerkin proper orthogonal decomposition methods for parabolic problems*, Numer. Math., 90 (2001), pp. 117–148.

[26] K. Kunisch, S. Volkwein, and L. Xie, *HJB-POD-based feedback design for the optimal control of evolution problems*, SIAM J. Appl. Dyn. Syst., 3 (2004), pp. 701–722.

[27] A. Kurganov and E. Tadmor, *New high-resolution semi-discrete central schemes for Hamilton–Jacobi equations*, J. Comput. Phys., 160 (2000), pp. 720–742.

[28] Y. Liu, J. Xin, and Y. Yu, *A numerical study of turbulent flame speeds of curvature and strain G-equations in cellular flows*, Phys. D, 243 (2013), pp. 20–31.

[29] J. Lyu, J. Xin, and Y. Yu, *Computing residual diffusivity by adaptive basis learning via spectral method*, Numer. Math., 10 (2017), pp. 351–372.

[30] J. Lyu, J. Xin, and Y. Yu, *Computing residual diffusivity by adaptive basis learning via super-resolution deep neural networks*, in International Conference on Computer Science, Applied Mathematics and Applications, Springer, New York, 2019, pp. 279–290.

[31] G. Markstein, *Nonsteady Flame Propagation*, Pergamon Press, Oxford, UK, 1964.

[32] B. Matkowsky and G. Sivashinsky, *An asymptotic derivation of two models in flame theory associated with the constant density approximation*, SIAM J. Appl. Math., 37 (1979), pp. 686–699.

[33] G. Namah and J. Roquejoffre, *Convergence to periodic fronts in a class of semilinear parabolic equations*, NoDEA Nonlinear Differential Equations Appl., 4 (1997), pp. 521–536.

[34] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Appl. Math. Sci. 153, Springer, New York, 2006.

[35] S. Osher and J. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.

[36] B. Peherstorfer and K. Willcox, *Dynamic data-driven reduced-order models*, Comput. Methods Appl. Mech. Engrg., 291 (2015), pp. 21–41.

[37] N. Peters, *Turbulent Combustion*, Cambridge University Press, New York, 2000.

[38] A. Quarteroni, A. Manzoni, and F. Negri, *Reduced Basis Methods for Partial Differential Equations: An Introduction*, Unitext, 92, Springer, New York, 2015.

[39] M.-L. Rapún, F. Terragni, and J. Vega, *Adaptive POD-based low-dimensional modeling supported by residual estimates*, Internat. J. Numer. Methods Engrg., 104 (2015), pp. 844–868.

[40] M.-L. Rapún and J. M. Vega, *Reduced order models based on local POD plus Galerkin projection*, J. Comput. Phys., 229 (2010), pp. 3046–3063.

[41] J. Roquejoffre, *Convergence to steady states or periodic solutions in a class of Hamilton–Jacobi equations*, J. Math. Pures Appl., 80 (2001), pp. 85–104.

[42] C. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, J. Comput. Phys., 77 (1988), pp. 439–471.

[43] L. Sirovich, *Turbulence and the dynamics of coherent structures.* I. *Coherent structures*, Quart. Appl. Math., 45 (1987), pp. 561–571.

[44] V. Thomée, *Galerkin Finite Element Methods for Parabolic Problems*, Springer Ser. Comput. Math. 1054, Springer, New York, 1984.

[45] S. Volkwein, *Proper Orthogonal Decomposition: Theory and Reduced-order Modelling*, Lecture notes, University of Konstanz, 2013.

[46] Z. Wang, X. Luo, S. Yau, and Z. Zhang, *Proper orthogonal decomposition method to nonlinear filtering problems in medium-high dimension*, IEEE Trans. Automat. Control, 65 (2019), pp. 1613–1624.

[47] J. Xin, *An Introduction to Fronts in Random Media*, Surv. Tutor, Appl. Math. Sci. 5, Springer, New York, 2009.