INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 4, Number 3-4, Pages 559–583

REDUCED-ORDER MODELING OF BURGERS EQUATIONS BASED ON CENTROIDAL VORONOI TESSELLATION

HYUNG-CHUN LEE, SUNG-WHAN LEE AND GUANG-RI PIAO

Dedicated to Professor Max D. Gunzburger on the occasion of his 60th birthday

Abstract. In this paper, we study a reduced-order modeling for Burgers equations. Review of the CVT(centroidal Voronoi tessellation) approaches to reduced-order bases are provided. In CVT-reduced order modelling, we start with a snapshot set just as is done in a POD(Proper Orthogonal Decomposition)-based setting. We shall investigate the technique of CVT as a procedure to determine the basis elements for the approximating subspaces. Some numerical experiments including comparison of CVT-based algorithm with numerical results obtained from FEM(finite element method) and POD-based algorithm are reported. Finally, we apply CVT-based reduced order modeling technique to a feedback control problem for Burgers equation.

Key Words. reduced-order modeling, proper orthogonal decomposition, centroidal Voronoi tessellations, Burgers equations.

1. Introduction

In the computational simulation of (nonlinear) complex, turbulent, or chaotic systems, standard discretization schemes (finite element, finite difference, etc.) may require many thousands or even millions of degrees of freedom for the accurate simulation of fluid flows. As a result, these schemes for the spatial discretization lead to sparse but very large-scale, and in general, nonlinear systems of ordinary differential equations (ODEs) and the approximate solutions using these approaches are expensive with respect to both storage and computing time. The situation is even worse for optimization problems for which multiple solutions of the complex state system are usually required or in feedback control problems for which real-time solutions of the complex state system are needed.

In order to overcome this difficulty, reduced-order modeling was introduced. Roughly speaking, this technique is to replace a given mathematical model of a system or process by a model that is much " small " than the original model, but still describes (at least approximately) certain aspects of the system or process. That is to say, the types of reduced-order models are those that attempt to determine accurate approximate solutions of a complex system using very few degrees

Received by the editors Februry 15, 2006 and, in revised form, February 23, 2006.
2000 *Mathematics Subject Classification.* 35B40, 35B37, 35Q30, 65M60.
This work was supported grant No. KRF-2005-070-C00017.

of freedom. To do so, such models have to use basis functions that are in some way intimately connected to the problem being approximated. This technique is based on projecting the dynamical system onto subspaces consisting of basis elements that contain characteristics of the expected solution. This is in contrast to the traditional numerical methods such as finite difference method (FDM) which uses grid functions as basis functions or finite elements method (FEM) which uses piecewise polynomials for this purpose.

The ideas underlying the reduced-basis method appear to have their origins in the suggestions of Almroth [1] and Nagy [13], which were developed by Noor and colleagues [14]-[16] in the context of simulations for structures and later by Peterson [18] in high Reynolds numbers incompressible viscous flow simulations. Roughly speaking, the reduced-basis method employs parameter-dependent solutions of the system to be approximated. These solutions are used to construct basis elements in the hope that solutions at other parameter values can be represented in terms of perturbations of solutions given at carefully chosen parameter values (the Lagrange basis approach) or in terms of a "moving frame" (the Taylor approach). It is important to note that the parameter-dependent solutions used as basis functions can be obtained either from full-order model numerical simulations or experimental data.

In this article, we focus on the *centroidal Voronoi tessellation* (CVT) as reduced order modelling technique which is currently an active research field. Centroidal Voronoi tessellation-based reduced-order modeling of fluid flows was developed by [8, 9]. In CVT-reduced order modelling, we start with a snapshot set just as is done in a POD-based setting. However, instead of determining a POD basis from the snapshot set, we apply our CVT methodologies to determine the generators of a CVT of the snapshot set; these generators constitute the reduced-order basis. We then use the CVT-based basis in just the same way as one uses a POD-based basis to determine a very low-dimensional approximation to the solution of a complex system. CVT also possesses an optimality property, although it is different from that possessed by POD bases. In this article, we shall investigate the CVT method as a reduced order model for the unsteady Burgers equation with appropriate initial and boundary conditions. As a matter of fact, POD and CVT may be viewed as simply different procedures to determine the basis elements for the approximating subspaces. We shall investigate the technique of CVT as a procedure to determine the basis elements for the approximating subspaces.

The plan for the rest of paper is as follows. In section 2, we give some definitions and property of CVT's, and two approaches for computing these tessellations. Section 3 is devoted to applying CVT to solve the time-dependent Burgers equation and, some numerical experiments including comparison of CVT-based algorithm with numerical results obtained from FEM and POD-based algorithm are reported in section 4. Finally, we apply CVT-based reduced-order modeling technique to a feedback control problem for Burgers equation in section 5.

2. Centroidal Voronoi Tessellation

The concept of the *centroidal Voronoi tessellations* (CVTs) has been studied in [5]. CVTs have been successfully used in several data compression settings, e.g.,

in image processing and the clustering of data. Reduced-order modeling of complex systems is another data compression setting, i.e., replacing high-dimensional approximations with low-dimensional ones.

We here consider the case where we are given a discrete set of points $\mathcal{S} = \{\mathbf{y}_i\}_{i=1}^m$ belonging to $\mathbb{R}^n$. The concept of centroidal Voronoi tessellations (CVTs) can be extended to more general sets, including regions in Euclidean space, and to more general metrics; for detailed discussions, see [5].

2.1. Definition of CVTs for discrete data sets. The definition of CVTs for discrete data sets begins with a set $\mathcal{S} = \{\mathbf{y}_i\}_{i=1}^m$ consisting of m vectors belonging to $\mathbb{R}^n$. Of course, $\mathcal{S}$ can also be viewed as a set of m points in $\mathbb{R}^n$ or a possibly complex-valued $n \times m$ matrix. In the context of CVT, it will be useful to think of the columns $\{S_{\cdot,j}\}_{j=1}^m$ of $\mathcal{S}$ as the spatial coordinate vectors of a dynamical system at time t_j. Similarly, we consider the rows $\{S_{i,\cdot}\}_{i=1}^n$ of $\mathcal{S}$ as the time trajectories of the dynamical system evaluated at the locations x_i.

Given a discrete set $\mathcal{S}$ belonging to $\mathbb{R}^n$, the set $\{T_i\}_{i=1}^l$ is called a *clustering* or a *tessellation* of the set $\mathcal{S}$ if $T_i \subset \mathcal{S}$ for $i = 1, \cdots, l$, $T_i \cap T_j = \emptyset$ for $i \neq j$, and $\cup_{i=1}^l T_i = \mathcal{S}$. Let $|\cdot|$ denote the Euclidean norm on $\mathbb{R}^n$. Given a set of points $\{\mathbf{z}_i\}_{i=1}^l$ belonging to $\mathcal{S}$ (or $\mathbb{R}^n$), the *Voronoi region* V_i corresponding to the point $\mathbf{z}_i$ is defined by

$$V_i \;=\; \{\; \mathbf{y} \in \mathcal{S} \mid \quad |\mathbf{y} - \mathbf{z}_i| \;<\; |\mathbf{y} - \mathbf{z}_j| \quad j = 1, \cdots, l,\; j \neq i\}.$$

The points $\{\mathbf{z}_i\}_{i=1}^l$ are called *generating points* or *(cluster) generators*. The set $\{V_i\}_{i=1}^l$ is known as a *Voronoi tessellation* or *Voronoi clustering* of $\mathcal{S}$ and each V_i is referred to as the *Voronoi region* or *cluster* corresponding to $\mathbf{z}_i$.

Given a density function $\rho(\mathbf{y})$ defined on $\mathcal{S}$, for each cluster V_i, we can define its *cluster centroid* $\mathbf{z}_i^*$ by

$$\mathbf{z}_i^* = \frac{\sum_{\mathbf{y} \in V_i} \mathbf{y}\rho(\mathbf{y})}{\sum_{\mathbf{y} \in V_i} \rho(\mathbf{y})} \qquad i = 1, \cdots, l.$$

Given a set $\mathcal{S}$ of m vectors in $\mathbb{R}^n$ and a positive integer $l \leq m$, a *centroidal Voronoi tessellation* (CVT) or *centroidal Voronoi clustering* of $\mathcal{S}$ is a special Voronoi tessellation satisfying

$$\mathbf{z}_i = \mathbf{z}_i^* \qquad i = 1, \cdots, l \tag{2.1}$$

i.e., the generators of the Voronoi tessellation coincide with the centroids of the corresponding Voronoi clusters. It is important to note that general Voronoi tessellations do not satisfy the CVT property (2.1) so that, for given a set $\mathcal{S}$ and positive integer l, a CVT must be constructed. Algorithms for this purpose are discussed in subsection 2.2.

Centroidal Voronoi tessellations are closely related to minimizers of an "*energy*". Specifically, let

$$\mathcal{E}(\{\mathbf{z}_i\}_{i=1}^l, \{V_i\}_{i=1}^l) \;=\; \sum_{i=1}^{l} \sum_{\mathbf{y} \in V_i} |\mathbf{y} - \mathbf{z}_i|^2 \rho(\mathbf{y}),$$

where $\{V_i\}_{i=1}^l$ is a tessellation of $\mathcal{S}$ and $\{\mathbf{z}_i\}_{i=1}^l$ are points in $\mathbb{R}^n$. No a priori relation is assumed between the V_i's and the $\mathbf{z}_i$'s. We refer to $\mathcal{E}$ as the "*cluster energy*"; in the statistics literature, it is called the *variance* or *cost*. It is easy to prove that a necessary condition for $\mathcal{E}$ to be minimized is that $\{\mathbf{z}_i, V_i\}_{i=1}^l$ is a centroidal Voronoi tessellation of $\mathcal{S}$; see [5].

The connection between CVTs and reduced-order bases is now easily made. The set $\mathcal{S}$ is obviously the snapshot set. Then, the CVT reduced basis set is the set of generators $\mathbf{z} = \{\mathbf{z}_i\}_{i=1}^l$ of a CVT of $\mathcal{S}$.

2.2. Algorithms for constructing discrete CVTs. As we have seen, the points that generate a Voronoi tessellation are not generally the centroids of the associated Voronoi regions. As a result, one is left with the following construction problem: given a region $\mathcal{S} \in \mathbb{R}^n$ and a positive integer l, determine an l-point centroidal Voronoi tessellation of $\mathcal{S}$.

There are many known methods for constructing centroidal Voronoi tessellations. We describe the two methods which are perhaps most "basic" and, certainly in the first case, the most used.

First, we have *Lloyd's method* [10] which is the straightforward iteration between constructing Voronoi tessellations and centroids.

Lloyd's method: Start with some initial set of l points $\{\mathbf{z}_i\}_{i=1}^l$ in $\mathcal{S}$, e.g., determined using random sampling;

(1) construct the Voronoi tessellation $\{V_i\}_{i=1}^l$ of $\mathcal{S}$ associated with the points $\{\mathbf{z}_i\}_{i=1}^l$;
(2) compute the centroids of the Voronoi regions $\{V_i\}_{i=1}^l$ found in Step 1; these centroids are the new set of points $\{\mathbf{z}_i\}_{i=1}^l$;
(3) go back to Step 1, or, if happy with convergence, quit.

Lloyd's method and its convergence properties have been analyzed; see [5] and [6] and also the references cited therein.

A second method is *McQueen's method* [12] which is a random sampling algorithm that doesn't require the explicit construction of Voronoi tessellations or of centroids.

McQueen's method: Start with some initial set of l points $\{\mathbf{z}_i\}_{i=1}^l$ in $\mathcal{S}$, e.g., determined using random sampling; set the integer array $J_i = 1$ for $i = 1, \cdots, l$;

(1) pick a random point $\mathbf{y} \in \mathcal{S}$;
(2) find the $\mathbf{z}_i$ closest to $\mathbf{y}$; denote the index of that $\mathbf{z}_i$ by i^*;
(3) set $\mathbf{z}_{i^*} \leftarrow \frac{J_{i^*}\mathbf{z}_{i^*}+\mathbf{y}}{J_{i^*}+1}$ and $J_{i^*} \leftarrow J_{i^*} + 1$;
(4) $\mathbf{z}_{i^*}$ along with the unchanged points $\{\mathbf{z}_i\}_{i=1, i\neq i^*}^l$ are the new set of points;
(5) go back to Step 1, or, if happy with convergence, quit.

Note that J_i keeps track of how many times the point $\mathbf{z}_i$ has been previously updated. Remarkably (since neither Voronoi tessellations or centroids appear anywhere in the definition of the algorithm), the McQueen iterations converge to a set of generators of a CVT. The convergence properties of McQueen's methods have been analyzed in [12].

One point is sampled at each McQueen iteration so that the McQueen iterations are cheap, but lots of them are needed. Lloyd's method requires relatively fewer iterations, but each iteration is expensive; a straightforward implementation requires the explicit construction of Voronoi regions and, to determine the centroids, numerical integrations on polyhedra.

3. CVT-based model reduction for the Burgers equation

3.1. Generating a snapshot. We now turn our attention to the computations. In order to generate a snapshot, we wish to numerically solve Burgers equation with homogeneous Dirichlet boundary conditions on the finite interval $[0, L]$. In particular, consider Burgers equation

$$\frac{\partial y}{\partial t}(t,x) + y(t,x)\frac{\partial y}{\partial x}(t,x) = \nu\frac{\partial^2 y}{\partial x^2}(t,x) + f(t,x) \tag{3.1}$$

with boundary condition

$$y(t,0) = y(t,L) = 0 \tag{3.2}$$

and initial condition

$$y(0,x) = y_0(x). \tag{3.3}$$

3.1.1. The Galerkin method and the weak form. First, we develop a weak formulation, from which we will derive the discretization. Multiplying both sides of the Burgers equation by a test function $v(x) \in V$, where V is a space of functions to be chosen later, and integrating over our domain $[0, L]$, we get

$$\begin{aligned}\int_0^L \frac{\partial y}{\partial t}(t,x)v(x)dx \quad &+ \quad \int_0^L y(t,x)\frac{\partial y}{\partial x}(t,x)v(x)dx \\ &= \quad \int_0^L \nu\frac{\partial^2 y}{\partial x^2}(t,x)v(x)dx + \int_0^L f(t,x)v(x)dx.\end{aligned}$$

Integrating by parts and using homogeneous Dirichlet boundary conditions yield the weak formulation of the problem,

$$\begin{aligned}&\int_0^L \frac{\partial y}{\partial t}(t,x)v(x)dx = \\ &\quad -\int_0^L y(t,x)\frac{\partial y}{\partial x}(t,x)v(x)dx - \nu\int_0^L \frac{\partial y}{\partial x}(t,x)v'(x)dx + \int_0^L f(t,x)v(x)dx.\end{aligned} \tag{3.4}$$

3.1.2. The Finite Element Approximation. Now we want to choose V such that the matrices resulting from our discretization will be as easily solvable as possible. We choose linear basis functions, $\phi_n(x)$, on the interval $[0, L]$; for details refer to [4]. The resulting matrices are tridiagonal. This sparseness given by these "hat" functions often improves computation time.

The interval of investigation, $[0, L]$, is divided into $N+1$ subintervals $[x_i, x_{i+1}]$ for $i = 0, \cdots, N$, each of length $h = L/(N+1)$. Let the $N+2$ linear basis functions be denoted by $\phi_i(x)$ for $1 \leq i \leq N$ and $0 \leq x \leq L$. But we only use the N linear basis functions, $\phi_1, \phi_2, \cdots, \phi_N$, because of the homogeneous Dirichlet boundary conditions. The approximate solution is then written as an expansion of these basis functions. In particular, we assume that

$$y^N(t,x) = \sum_{i=1}^{N} c_i(t)\phi_i(x),$$

where the coefficients $c_i(t)$ remain to be computed. We substitute $y^N(t,x)$ for $y(t,x)$ and replace $v(x)$ by our basis function $\phi_j(x)$ in (3.4). The basis functions are orthogonal when $i > j+1$ and $i < j-1$. The result is a first-order non-linear differential equation for the vector function $\mathbf{c}(t)$:

$$\begin{aligned}\int_0^L \sum_{i=1}^{N} \dot{c}_i(t)\phi_i(x)\phi_j(x)dx = &-\int_0^L \sum_{i=1}^{N} c_i(t)\phi_i(x) \sum_{k=1}^{N} c_k(t)\phi_k^{'}(x)\phi_j(x)dx \\ &- \nu\int_0^L \sum_{i=1}^{N} c_i(t)\phi_i^{'}(x)\phi_j^{'}(x)dx + \int_0^L f(t,x)\phi_j(x)dx\end{aligned} \tag{3.5}$$

The basis functions satisfy $\phi_i(x_j) = \delta_{ij}$, thus we have $y^N(t, x_i) = c_i(t)$. So the coefficients $c_i(t)$ give the value of the approximate solution at the i^{th} node.

Looking at the term on the LHS of (3.5), we have an N by N matrix with rows given by j and columns by i. We factor out the vector

$$\dot{\mathbf{c}}(t) = \begin{bmatrix} \dot{c}_1(t) \\ \dot{c}_2(t) \\ \vdots \\ \dot{c}_N(t) \end{bmatrix}_{N\times 1}$$

and call the remaining matrix the mass matrix, $[\mathbf{M}]$, due to its relation to the mass of the object being studied. The entries of the mass matrix are given by

$$m_{i,j} = \int_0^L \phi_i(x)\phi_j(x)dx.$$

The third term on the RHS of (3.5) yields the vector function

$$\mathbf{F}(t) = \begin{bmatrix} \int_0^L f(t,x)\phi_1(x)dx \\ \int_0^L f(t,x)\phi_2(x)dx \\ \vdots \\ \int_0^L f(t,x)\phi_N(x)dx \end{bmatrix}_{N\times 1}$$

Inspecting the second term on the RHS of (3.5), we have an N by N matrix with rows given by j and columns by i. We factor out the vector

$$\mathbf{c}(t) = \begin{bmatrix} c_1(t) \\ c_2(t) \\ \vdots \\ c_N(t) \end{bmatrix}_{N\times 1}$$

and call the remaining matrix the stiffness matrix, $[\mathbf{A}]$. The entries of the stiffness matrix are computed by

$$a_{i,j} = \int_0^L \phi_i^{'}(x)\phi_j^{'}(x)dx.$$

The first term on the RHS of (3.5) results in a tensor, which we call $\mathbf{N}$. The entries of the tensor are computed by

$$n_{i,k,j} = \int_0^L \phi_i(x)\phi_k^{'}(x)\phi_j(x)dx.$$

We have now reduced (3.5) to an N-dimensional matrix equation,

$$[\mathbf{M}]\dot{\mathbf{c}}(t) = -\mathbf{c}(t)^T\mathbf{N}\mathbf{c}(t) - \nu[\mathbf{A}]\mathbf{c}(t) + \mathbf{F}(t). \tag{3.6}$$

Since $[\mathbf{M}]$ is known to be invertible, we can multiply both sides of (3.6) by $[\mathbf{M}]^{-1}$ to obtain

$$\dot{\mathbf{c}}(t) = -[\mathbf{M}]^{-1}\mathbf{c}(t)^T\mathbf{N}\mathbf{c}(t) - \nu[\mathbf{M}]^{-1}[\mathbf{A}]\mathbf{c}(t) + [\mathbf{M}]^{-1}\mathbf{F}(t).$$

We now turn our attention to the initial condition. Multiplying both sides of the initial condition by $v(x)$ and integrating on the interval $[0, L]$, we get

$$\int_0^L y(0,x)v(x)dx = \int_0^L y_0(x)v(x)dx.$$

The initial condition is expanded in the linear basis functions and yields

$$\int_0^L \sum_{i=1}^N c_i(0)\phi_i(x)\phi_j(x)dx = \int_0^L y_0(x)\phi_j(x)dx, \tag{3.7}$$

for $j = 1, 2, \cdots, N$.

Inspecting the term on the LHS of (3.7), we again have an N by N matrix with the rows given by j and the columns by i. We factor out the vector

$$\mathbf{c}(0) = \begin{bmatrix} c_1(0) \\ c_2(0) \\ \vdots \\ c_N(0) \end{bmatrix}_{N\times 1}$$

and we see that the remaining matrix is again the mass matrix, $[\mathbf{M}]$.

On the RHS of (3.7) we have the N-dimensional vector

$$\mathbf{y_0^N} = \begin{bmatrix} \int_0^L y_0(x)\phi_1(x)dx \\ \int_0^L y_0(x)\phi_2(x)dx \\ \vdots \\ \int_0^L y_0(x)\phi_N(x)dx \end{bmatrix}_{N\times 1}$$

For the initial condition, we have

$$[\mathbf{M}]\mathbf{c_0} = \mathbf{y_0^N}.$$

Again multiply both sides by $[\mathbf{M}]^{-1}$ to obtain the final form of the initial condition:

$$\mathbf{c_0} = [\mathbf{M}]^{-1}\mathbf{y_0^N}.$$

We have now reached the final form of the problem. We want to solve

$$\dot{\mathbf{c}}(t) = -[\mathbf{M}]^{-1}\mathbf{c}(t)^T\mathbf{N}\mathbf{c}(t) - \nu[\mathbf{M}]^{-1}[\mathbf{A}]\mathbf{c}(t) + [\mathbf{M}]^{-1}\mathbf{F}(t). \tag{3.8}$$

given the initial condition

$$\mathbf{c_0} = [\mathbf{M}]^{-1}\mathbf{y_0^N}. \tag{3.9}$$

We are now ready to implement the computational code. In section 4, this scheme is implemented in MATLAB and the resulting ODE (3.8)–(3.9) is solved using the MATLAB function ODE113. The routine ODE113 uses the Adams-Bashforth-Moulton method to solve the ODE.

3.2. Reduced Order Modelling via CVT. Let the snapshot set arise from the discretization of the Burgers equation. Now we fix l and determine the CVT basis $\psi_1, \cdots, \psi_l$ by using the Algorithm introduced in subsection 2.2. We use the usual Euclidean distance and the uniform density function in the CVT clustering algorithm. The reduced-order solution is then written as an expansion of these basis functions. In particular, we assume that

$$y^l(t,x) = \sum_{i=1}^{l} d_i(t)\psi_i(x),$$

where the coefficients $d_i(t)$ remain to be computed. We substitute $y^l(t,x)$ for $y(t,x)$ and replace $v(x)$ by our basis function $\psi_j(x)$ in (3.4). The result is a first-order non-linear differential equation for the vector function $\mathbf{d}(t)$:

$$\begin{aligned}\int_0^L \sum_{i=1}^{l} \dot{d}_i(t)\psi_i(x)\psi_j(x)dx = &-\int_0^L \sum_{i=1}^{l} d_i(t)\psi_i(x)\sum_{k=1}^{l} d_k(t)\psi_k^{'}(x)\psi_j(x)dx \\ &- \nu\int_0^L \sum_{i=1}^{l} d_i(t)\psi_i^{'}(x)\psi_j^{'}(x)dx + \int_0^L f(t,x)\psi_j(x)dx\end{aligned} \tag{3.10}$$

Each basis, generator $\psi_i \in R^N$ of a CVT, defineds a finite element function, i.e., if

$$\psi_i = \begin{bmatrix} \psi_i^1 \\ \psi_i^2 \\ \vdots \\ \psi_i^N \end{bmatrix} \qquad \text{for } i \;=\; 1, \cdots, l,$$

we then have the corresponding finite element functions

$$\psi_i \;=\; \sum_{j=1}^{N} \psi_i^j \phi_j(x) \qquad \text{for } i \;=\; 1, \cdots, l.$$

Looking at the term on the LHS of (3.10), we have an l by l matrix with rows given by j and columns by i. We factor out the vector

$$\dot{\mathbf{d}}(t) = \begin{bmatrix} \dot{d}_1(t) \\ \dot{d}_2(t) \\ \vdots \\ \dot{d}_l(t) \end{bmatrix}_{l\times 1}$$

and call the remaining matrix the mass matrix, $[\mathcal{M}]$. The entries of the mass matrix are given by

$$\begin{aligned}\mathfrak{m}_{i,j} &= \int_0^L \psi_i(x)\psi_j(x)dx \\ &= \int_0^L \sum_{k=1}^N \psi_i^k \phi_k(x) \sum_{k=1}^N \psi_j^k \phi_k(x)dx \\ &= \psi_i(x)^T\ [\mathbf{M}]\ \psi_j(x).\end{aligned}$$

where $[\mathbf{M}]$ is the mass matrix given in subsection 3.1.

The third term on the RHS of (3.10) yields the vector function

$$\mathcal{F}(t) = \begin{bmatrix} \int_0^L f(t,x)\psi_1(x)dx \\ \int_0^L f(t,x)\psi_2(x)dx \\ \vdots \\ \int_0^L f(t,x)\psi_l(x)dx \end{bmatrix}_{l\times 1}$$

Inspecting the second term on the RHS of (3.10), we have an l by l matrix with rows given by j and columns by i. We factor out the vector

$$\mathbf{d}(t) = \begin{bmatrix} d_1(t) \\ d_2(t) \\ \vdots \\ d_l(t) \end{bmatrix}_{l\times 1}$$

and call the remaining matrix the stiffness matrix, $[\mathcal{A}]$. The entries of the stiffness matrix are computed by

$$\begin{aligned}\mathfrak{a}_{i,j} &= \int_0^L \psi_i^{'}(x)\psi_j^{'}(x)dx \\ &= \int_0^L \sum_{k=1}^N \psi_i^k \phi_k^{'}(x) \sum_{k=1}^N \psi_j^k \phi_k^{'}(x)dx \\ &= \psi_i(x)^T\ [\mathbf{A}]\ \psi_j(x).\end{aligned}$$

where $[\mathbf{A}]$ is the stiff matrix given in subsection 3.1.

The first term on the RHS of (3.10) results in a l by l by l tensor, which we call $\mathcal{N}$. The entries of the tensor are computed by

$$\begin{aligned}\mathfrak{n}_{i,k,j} &= \int_0^L \psi_i(x)\psi_k^{'}(x)\psi_j(x)dx \\ &= \int_0^L \sum_{m=1}^N \psi_i^m \phi_m(x) \sum_{m=1}^N \psi_k^m \phi_k^{'}(x) \sum_{m=1}^N \psi_j^m \phi_m(x)dx \\ &= [\psi_i(x)^T\ \mathbf{N}\ \psi_k(x)]^T \psi_j(x).\end{aligned}$$

where $\mathbf{N}$ is the tensor given in subsection 3.1.

We have now reduced (3.10) to an l-dimensional matrix equation,

$$[\mathcal{M}]\dot{\mathbf{d}}(t) = -\mathbf{d}(t)^T \mathcal{N}\mathbf{d}(t) - \nu[\mathcal{A}]\mathbf{d}(t) + \mathcal{F}(t). \tag{3.11}$$

Since $[\mathcal{M}]$ is known to be invertible, we can multiply both sides of equation (3.11) by $[\mathcal{M}]^{-1}$ to obtain

$$\dot{\mathbf{d}}(t) = -[\mathcal{M}]^{-1}\mathbf{d}(t)^T\mathcal{N}\mathbf{d}(t) - \nu[\mathcal{M}]^{-1}[\mathcal{A}]\mathbf{d}(t) + [\mathcal{M}]^{-1}\mathcal{F}(t).$$

In similar way, we can obtain the final form of the initial condition :

$$\mathbf{d}_0 = [\mathcal{M}]^{-1}\mathbf{y}_0^l.$$

where $\mathbf{y}_0^l$ is the l-dimensional vector

$$\mathbf{y}_0^l = \begin{bmatrix} \int_0^L y_0(x)\psi_1(x)dx \\ \int_0^L y_0(x)\psi_2(x)dx \\ \vdots \\ \int_0^L y_0(x)\psi_l(x)dx \end{bmatrix}_{l\times 1}$$

We have now reached the final form of the problem. We want to solve

$$\dot{\mathbf{d}}(t) = -[\mathcal{M}]^{-1}\mathbf{d}(t)^T\mathcal{N}\mathbf{d}(t) + \nu[\mathcal{M}]^{-1}[\mathcal{A}]\mathbf{d}(t) + [\mathcal{M}]^{-1}\mathcal{F}(t). \tag{3.12}$$

given the initial condition

$$\mathbf{d}_0 = [\mathcal{M}]^{-1}\mathbf{y}_0^l. \tag{3.13}$$

To implement the computational code, this scheme is also implemented in MATLAB and the resulting ODE (3.12)–(3.13) is solved using the Adams-Bashforth-Moulton method.

4. Computational Experiments

4.1. Setting Up the Problem. Consider the Burgers equation

$$\frac{\partial y}{\partial t}(t,x) - \nu\frac{\partial^2 y}{\partial x^2}(t,x) + y(t,x)\frac{\partial y}{\partial x}(t,x) = f(t,x)$$

with the homogeneous Dirichlet boundary condition

$$y(t,0) = y(t,L) = 0$$

and initial condition

$$y(0,x) = y_0(x).$$

We wish to compute approximate solutions of this problem to determine a set of snapshot vectors.

The Galerkin finite element model was described in subsection 3.1 and results in a system of ordinary differential equations :

$$[\mathbf{M}]\ \dot{\mathbf{c}}(t) + \nu\ [\mathbf{A}]\ \mathbf{c}(t) + \mathbf{c}(t)^T\ [\mathbf{N}]\ \mathbf{c}(t)\ =\ \mathbf{F}(t) \qquad in\ (0,T) \tag{4.1}$$

with the initial condition at $t = 0$

$$[\mathbf{M}]\ \mathbf{c}(0)\ =\ \mathbf{c}_0. \tag{4.2}$$

This problem is set up in MATLAB and solved by the Adams-Bashforth-Moulton method. The grid is chosen to be

$$x_i = \frac{i}{N+1} \quad \text{for} \quad i = 0, ..., N+1 \quad \text{and} \quad t_j = \frac{jT}{m} \quad \text{for} \quad j = 0, ..., m-1.$$

Most of the examples presented in this thesis employ numerical approximation schemes using $N = 119$ elements. This partitions the interval $[0, L]$ into $N+1 = 120$ subintervals of uniform length. The result is $N = 119$ nodes, making (4.1) a system of 119 ordinary differential equations.

Example: Let us present a numerical example. The programs were written in MATLAB Version 7.1 executed on a INTEL(R) PENTIUM(R) 4 Series computer. For the Burgers equation we chose the parameters $T = 1, \nu = 0.01, f = 0$ and

$$y_0(x) = \begin{cases} 1 & \text{in } (0, \frac{1}{2}], \\ 0 & \text{otherwise.} \end{cases}$$

The numerical solution to (4.1)–(4.2) in case of $N = 119$, $m = 120$ is shown in Figure 1.

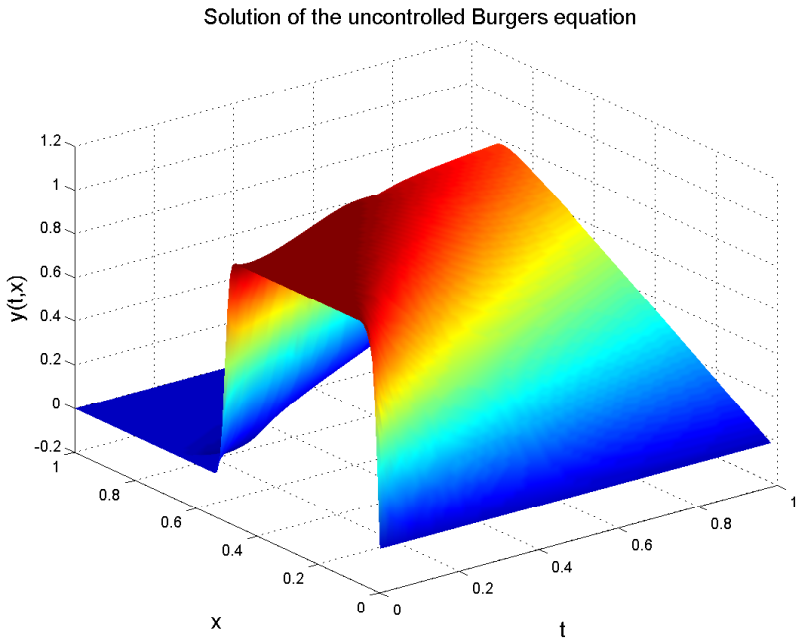

FIGURE 1. Full finite element solution of the Burgers equation.

4.2. A Comparison to FEM Solutions. In this subsection, we compare two reduced order methods (CVT and POD) of computation. The goal is to verify that the CVT scheme is returning the FEM answer as the number of basis is increased. The CVT method is compared to the POD method for $l = 4, 8$, and 16 basis. We conclude that CVT and POD have a very similar accuracy.

First of all, we apply the algorithms of subsection 2.3, the Lloyd's method, to determine the generators of a CVT of the snapshot set; a set of generators is to be used as a reduced-order basis.

Table 1, 2, and 3 compare, for the 4, 8, and 16 generator cases respectively, cluster energy, population, and range of snapshot index corresponding to each cluster. Note that the clusters are formed exactly from a sequence of data points at neighboring times.

total iterations	6		
Cluster	Cluster energy	Population	Range of snapshot index
1	43.7843	24	[1, 24]
2	37.4290	29	[25, 53]
3	35.4943	31	[54, 84]
4	34.9630	36	[85, 120]
total	151.6706	120	[1, 120]

TABLE 1. Cluster statistics for the 4 generator case.

total iterations	9		
Cluster	Cluster energy	Population	Range of snapshot index
1	7.3041	10	[1, 10]
2	5.6761	13	[11, 23]
3	4.4764	13	[24, 36]
4	5.0023	15	[37, 51]
5	5.8278	16	[52, 67]
6	6.1582	17	[68, 84]
7	5.8572	18	[85, 102]
8	4.1329	18	[103, 120]
total	44.4350	120	[1, 120]

TABLE 2. Cluster statistics for the 8 generator case.

Now, CVTs having $l = 4, 8$ and 16 generators are determined. Figures 2, 7, and 12 display, for the 4, 8, and 16 generator cases respectively, the CVT basis computed from the snapshot which is not normalized. Figures 3, 8, and 13 display, for the 4, 8, and 16 generator cases, the POD basis computed from the same snapshot data.

It is important to note that, in contrast to the POD basis set, the CVT basis set of size 8 is not built by augmenting the CVT basis set of size 4; most of the elements of the larger set seem significantly different from any of those of the smaller set.

In some computations relating CVT and POD, we have a priori knowledge of the FEM solution. In this case, we measure the numerical error introduced by the two approximating schemes, calculating the relative l_2 error at time t given by

$$\text{Relative } l_2 \text{ error } (t) = \frac{(\sum_{i=1}^{N}(y^l(t,x_i) - y^N(t,x_i))^2)^{\frac{1}{2}}}{(\sum_{i=1}^{N} y^N(t,x_i)^2)^{\frac{1}{2}}}.$$

The error is also shown graphically by computing $y^l(t,x)$ and $y^N(t,x)$ at $N+2$ evenly-spaced points in each subinterval. For the 4, 8, and 16 generator cases,

total iterations	6		
Cluster	Cluster energy	Population	Range of snapshot index
1	0.0000	1	[1]
2	0.6254	4	[2, 5]
3	0.8782	6	[6, 11]
4	0.9581	7	[12, 18]
5	0.7882	7	[19, 25]
6	0.7163	7	[26, 32]
7	0.9365	8	[33, 40]
8	0.7529	8	[41, 48]
9	0.7464	8	[49, 56]
10	0.7458	8	[57, 64]
11	0.7132	8	[65, 72]
12	0.6636	8	[73, 80]
13	0.8557	9	[81, 89]
14	1.0303	10	[90, 99]
15	0.8806	10	[100, 109]
16	0.8460	11	[110, 120]
total	12.1372	120	[1, 120]

TABLE 3. Cluster statistics for the 16 generators case.

the numerical solution and the actual error (the difference $y^l(t,x) - y^N(t,x)$) are plotted at these points for comparison. Figures 4, 9, and 14 display, for the 4, 8, and 16 generator cases respectively, CVT-based numerical solutions and corresponding actual errors. Figures 5, 10, and 15 display, for the 4, 8, and 16 generator cases respectively, POD-based numerical solutions and corresponding actual errors.

For the 4, 8, and 16 generator cases, the plots of relative l_2 errors versus time are displayed in Figures 6, 11 and 16 respectively. Note that, for both CVT and POD, relative l_2 errors decrease as the size of the CVT basis set is increased.

Table 4 compares the computational times for the CVT and POD methods using $l = 4$, 8, and 16. The speeds for the POD method at $l = 4$, and 8 are slightly faster than the speeds for the CVT method. However, the CVT method is computationally faster than the POD method, especially as l increases to 16. Figure 17 shows the relative errors versus time for the 4, 8, and 16 basis cases and both CVT- and POD-based cases.

the number of generator	method	computation time
$N = 119$	FEM	2.2488e+004
$l = 4$	POD	0.1570
	CVT	0.2180
$l = 8$	POD	0.8600
	CVT	1.0000
$l = 16$	POD	21.6410
	CVT	18.7030

TABLE 4. Computation times.

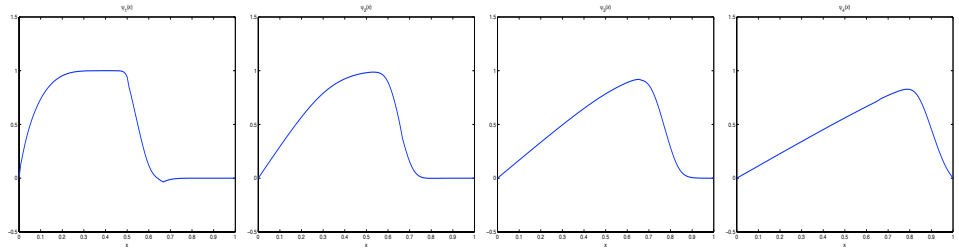

FIGURE 2. CVT basis functions for $l = 4$.

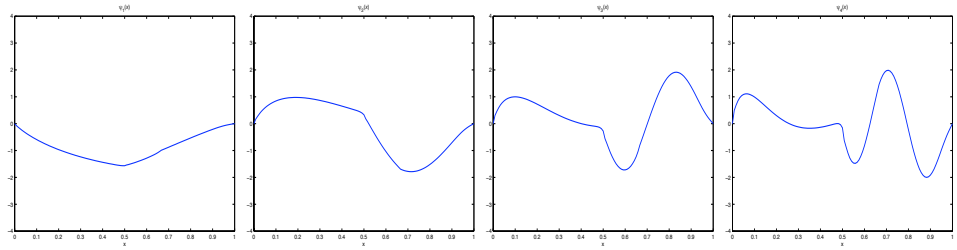

FIGURE 3. POD basis functions for $l = 4$

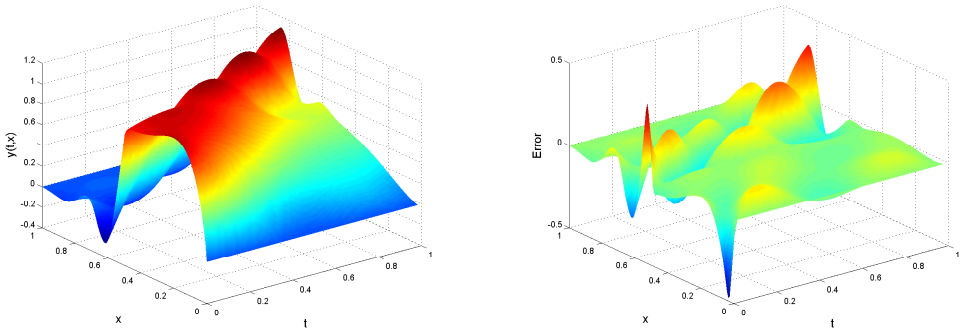

FIGURE 4. CVT-based reduced order solution (left) and actual errors using 4 generators.

5. Distributed feedback control problem

In this section, we apply the CVT-based reduced-order modeling method to a feedback control problem for the Burgers equation. The optimal control problem is to stabilize the solution to (3.1)–(3.3). The forcing term $f(t, x)$ is used to describe a distributed control. For an uncontrolled problem, $f(t, x) = 0$. For a controlled problem, the control term is assumed to have a special form of $f(t, x) = b(x)u(t)$, where $u(t)$ is a control input and $b(x)$ is a given function used to describe the control over the domain.

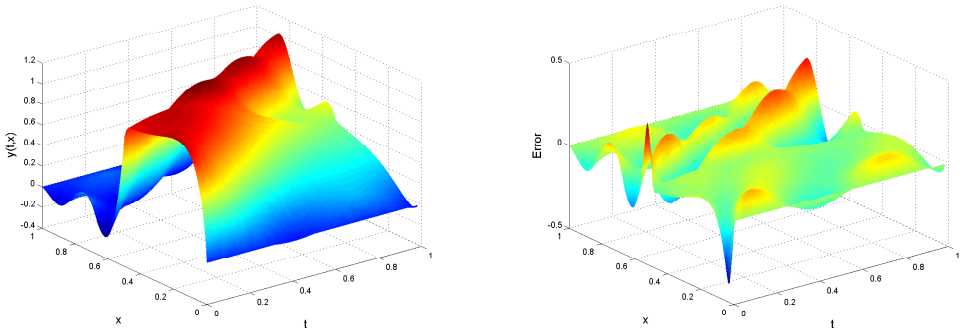

FIGURE 5. POD-based reduced order solution (left) and actual errors using 4 generators.

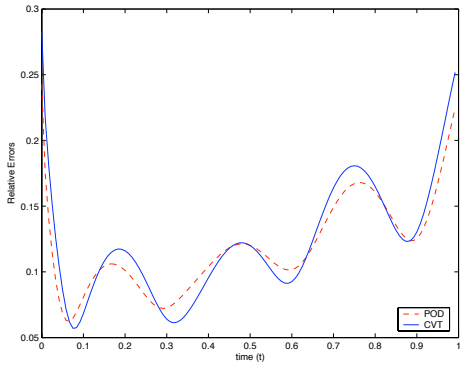

FIGURE 6. Relative errors vs. time.

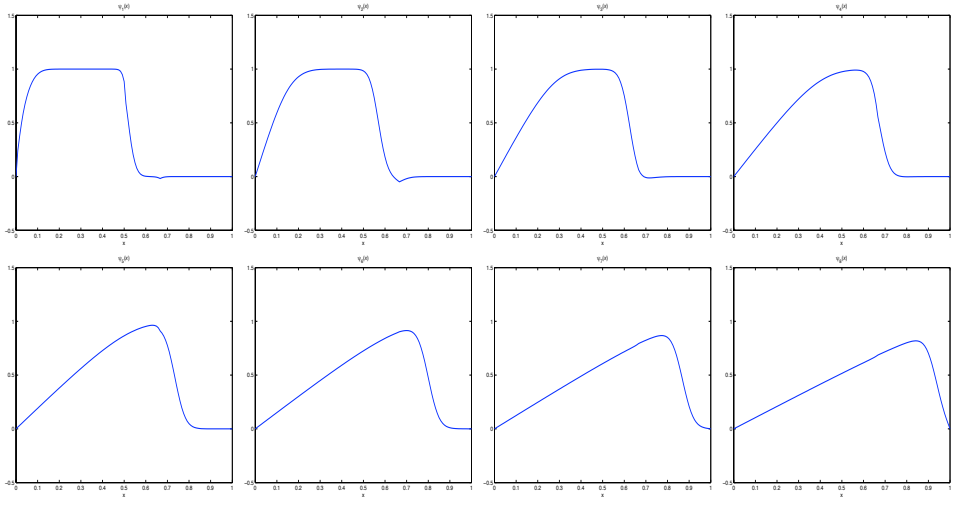

FIGURE 7. CVT basis functions for $l = 8$.

Now we describe our control problem. Find an optimal control $u(t)$ which minimizes the cost functional

$$J(u) = \int_0^\infty \left(||y(t,\cdot)||^2_{L^2(\Omega)} + |u(t)|^2 \right) dt$$

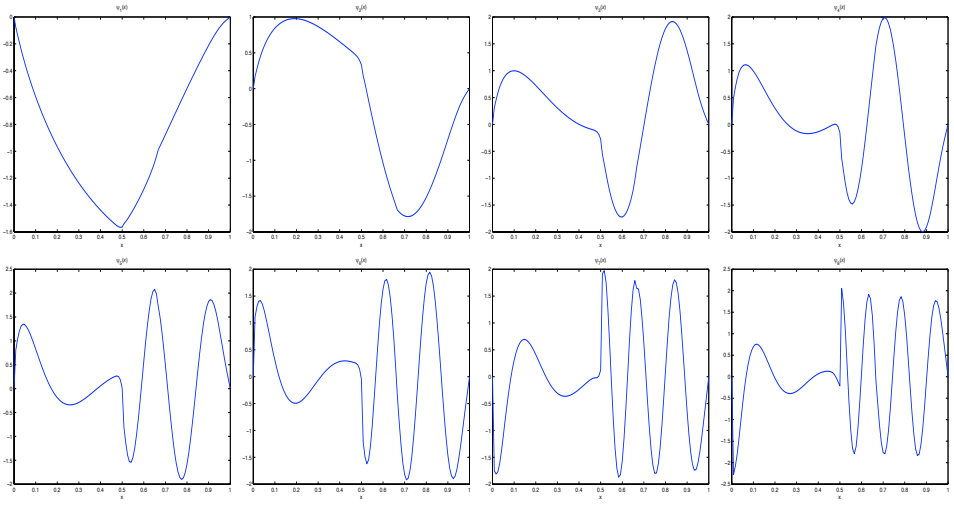

FIGURE 8. POD basis functions for $l = 8$.

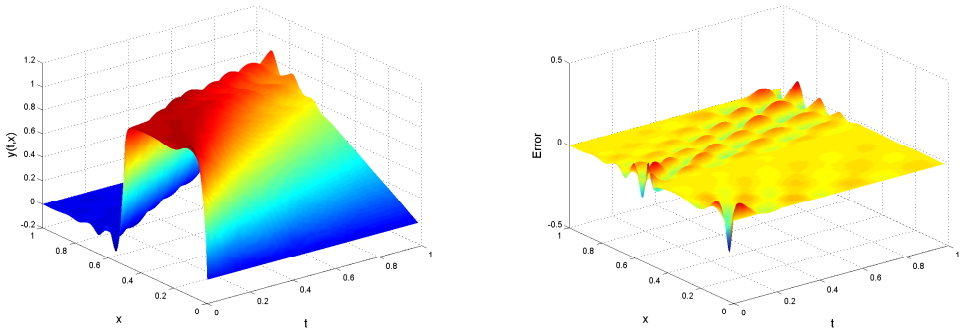

FIGURE 9. CVT-based reduced order solution (left) and actual errors using 8 generators.

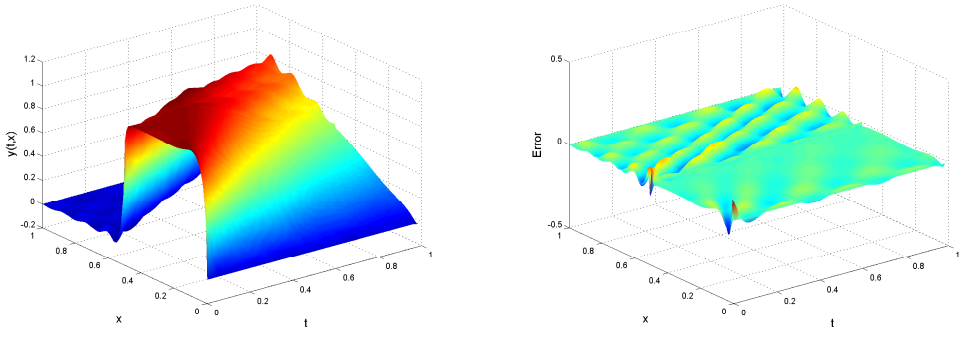

FIGURE 10. POD-based reduced order solution (left) and actual errors using 8 generators.

subject to the constraint equations

$$\frac{\partial y}{\partial t}(t,x) + y(t,x)\frac{\partial y}{\partial x}(t,x) = \nu \frac{\partial^2 y}{\partial x^2}(t,x) + u(t)b(x) \tag{5.1}$$

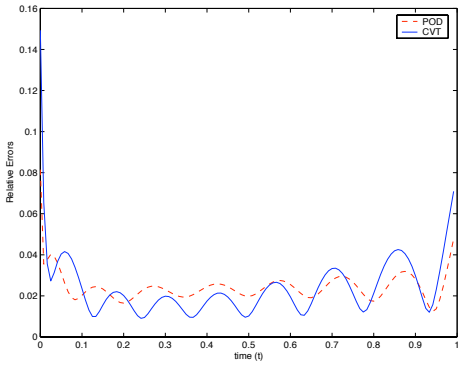

FIGURE 11. Relative errors for $l = 8$.

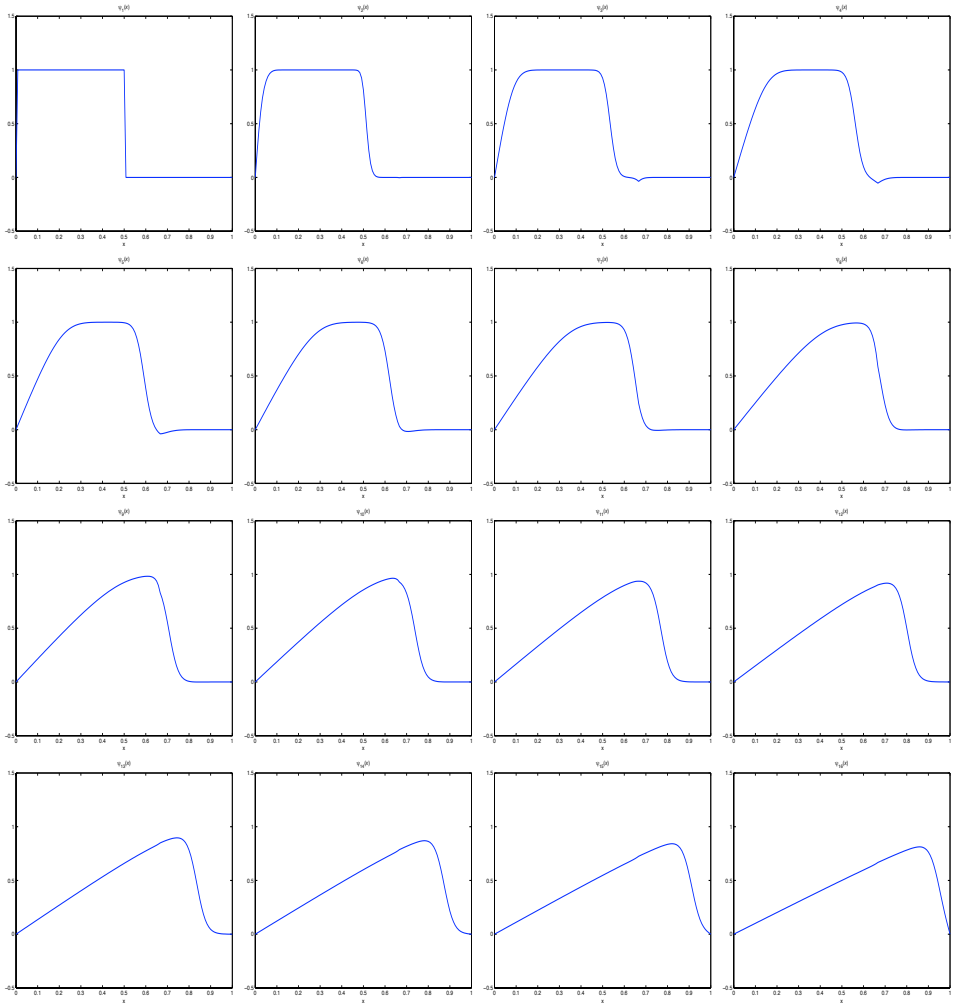

FIGURE 12. CVT basis functions for $l = 16$.

with homogeneous boundary condition

$$y(t,0) = y(t,L) = 0 \tag{5.2}$$

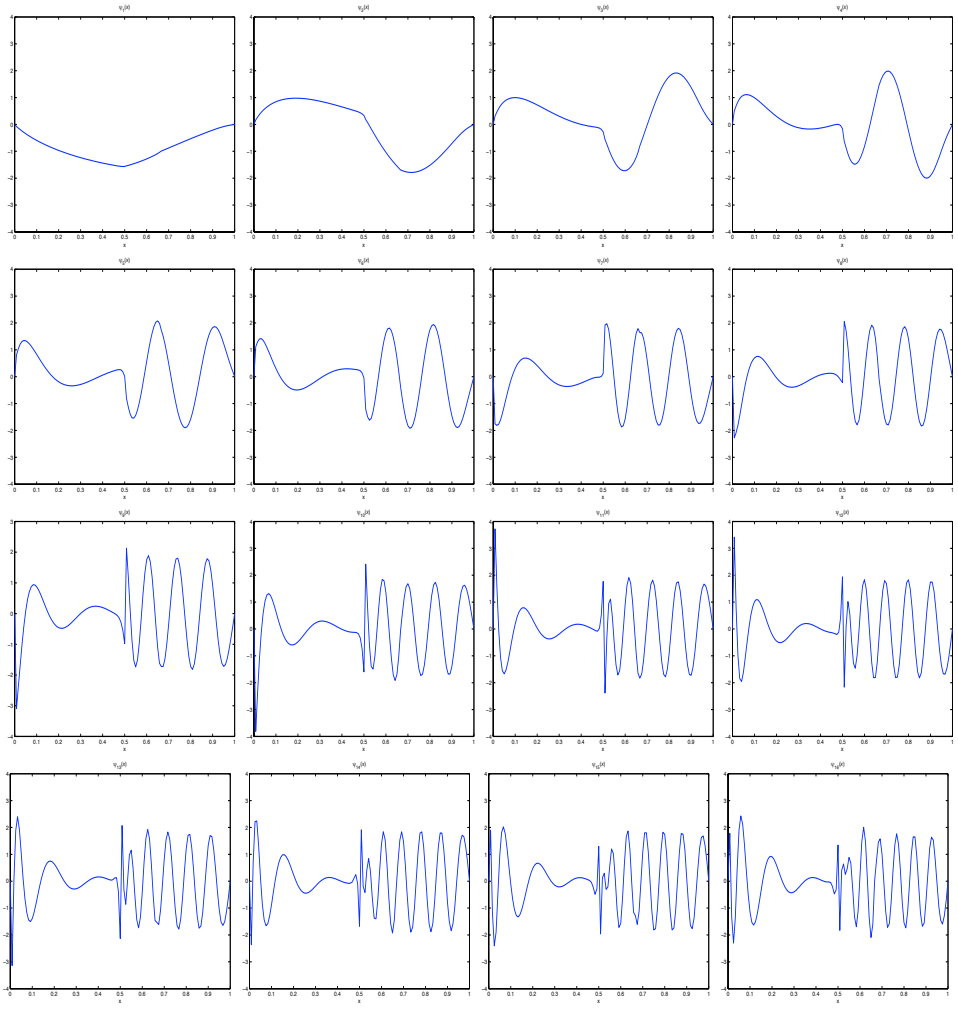

FIGURE 13. POD basis functions for $l = 16$.

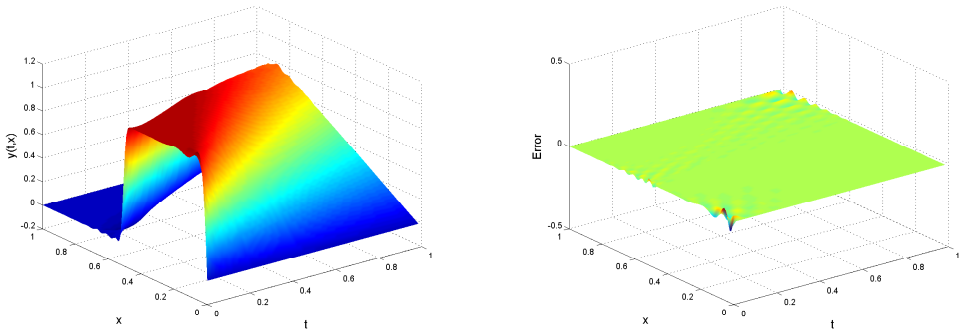

FIGURE 14. CVT-based reduced order solution (left) and actual errors using 16 generators.

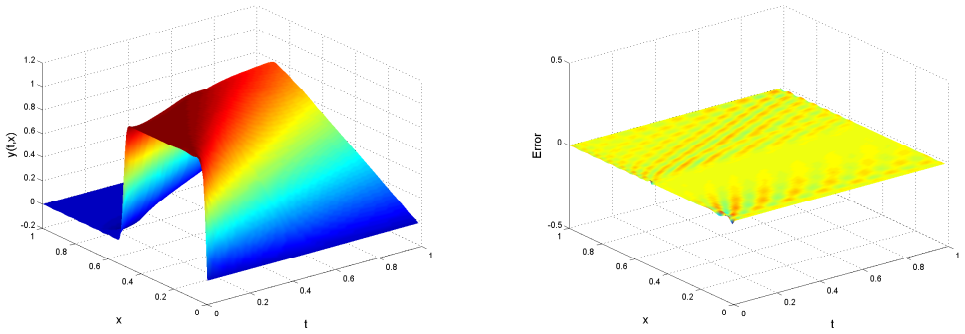

FIGURE 15. POD-based reduced order solution (left) and actual errors using 16 generators.

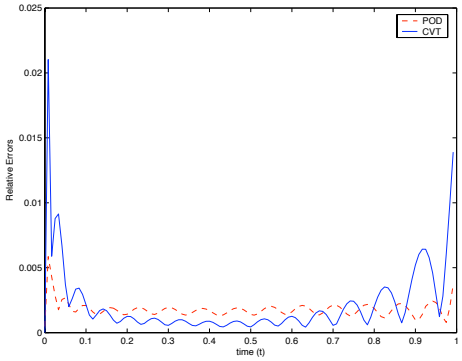

FIGURE 16. Relative errors for $l = 16$.

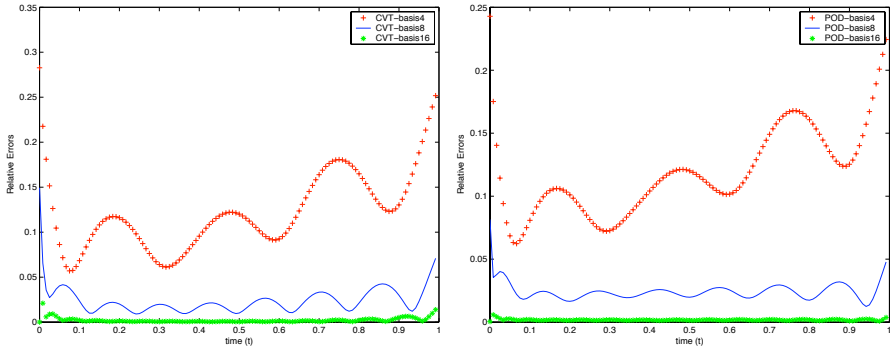

FIGURE 17. CVT(left)- and POD(right)-based relative errors vs. time.

and initial condition

$$y(0, x) = y_0(x). \tag{5.3}$$

Let $y(t) = y(t, \cdot)$ be the state in state space $L_2(\Omega)$. Let us define the linear operator $\mathcal{A}_\nu$ as $\mathcal{A}_\nu y = \nu y''$, for all $y \in \mathcal{D}(\mathcal{A}_\nu) = H_0^1(\Omega) \cap H^2(\Omega)$. The abstract form of the controlled model problem of (5.1)–(5.3) can be written as the initial value problem

$$\dot{y}(t) = \mathcal{A}_\nu y(t) + N(y(t)) + Bu(t), \quad y(0) = y_0, \quad for \quad t > 0 \tag{5.4}$$

on the space $L^(\Omega)$, where $N(y) = -yy'$ is defined on $H_0^1(\Omega)$.

5.1. Linear feedback controllers with full state feedback and state estimate feedback. Assuming the nonlinear term in the Burgers equation is small, a suboptimal feedback control u^* can be obtained by using the well-known linear quadratic regulator theory. That is, a full state state feedback control is to find an optimal control $u^* \in L^2([0,T), L^2(\Omega))$ by minimizing the cost functional

$$J(u) = \int_0^\infty \left(Qy(t,\cdot), y(t,\cdot))_{L^2(\Omega)} + (Ru(t), u(t)) \right) dt \tag{5.5}$$

subject to the constraint equations (5.1)–(5.3). Where $Q : L^2(\Omega) \to L^2(\Omega)$ is a nonnegative definite self-adjoint weighting operator for state and $R : L^2(\Omega) \to L^2(\Omega)$ is a positive definite weighting operator for the control. The optimal control $u^*(t)$ can be found as

$$u^*(t) = -\frac{1}{2} R^{-1} B^T \Pi y(t) = -Ky(t),$$

where K is called the feedback operator and Π is symmetric positive definite solution of the algebraic Riccati equation

$$\Pi \mathcal{A} + \mathcal{A}^T \Pi - \Pi B R^{-1} B^T \Pi + Q = 0. \tag{5.6}$$

Once the feedback operator K is determined, solutions to the closed loop system can be obtained from

$$\dot{y}(t) = (\mathcal{A} - BK)y(t) + N(y(t)). \tag{5.7}$$

Implementation of a controller requires a numerical approximation of the system. For PDE models of systems with complex dynamics, such as Burgers equation and Navier-Stokes equations, approximations are large scale on the order of thousands or even millions of variables. A framework for reduced order controllers is necessary and is applied to such problems by linearizing and designing the linear feedback law.

In this paper, we do not assume that we have knowledge of the full state. Instead, we assume a state measurement of the form

$$z(t) = Cy(t), \tag{5.8}$$

where $C \in \mathcal{L}(W, \mathbb{R}^m)$. We can apply the theory and results(Burns and Kang, 1991; Marrekchi, 1993;) to show that a stabilizing compensator based controller can be applied to the system. Recently Atwell and King investigate reduced order controllers for spatially distributed systems using proper orthogonal decomposition theory.

The observer design is mainly needed in order to provide the feedback control law with estimated state variables. Therefore, the control law and observer are combined together into a complete system. The combined system is called compensator. This technique assumes the availability of a limited measurement of the state. Assume we have a system in the abstract form

$$\dot{y}(t) = Ay(t) + G(y(t)) + Bu(t), \qquad y(0) = y_0, \tag{5.9}$$

where $y(t)$ is in a state space W and $u(t)$ is in a control space U.

Suppose the state measurement is given by

$$z(t) = Cy(t). \tag{5.10}$$

Given this measurement, a state estimate, $\tilde{y}(t)$, is computed by solving the observer equation

$$\dot{\tilde{y}}(t) = A\tilde{y}(t) + G(\tilde{y}(t)) + Bu(t) + L[C\tilde{y}(t) - z(t)], \qquad \tilde{y}(0) = \tilde{y}_0. \tag{5.11}$$

The feedback control law is given by

$$u(t) = -K\tilde{y}(t), \tag{5.12}$$

where K is called the feedback operator. Where functional gain operator K and estimator gain operator L are determined by linear quadratic regulator(LQR) and Kalman estimator(LQE), respectively, in usual manner.

A finite element method provides finite dimensional approximations of (5.9)–(5.10) of order N (where order refers to the freedom of finite element), given by

$$\dot{y}^N(t) = A^N y^N(t) + G^N(y^N(t)) + B^N u^N(t), \qquad y^N(0) = y_0^N, \tag{5.13}$$

$$z^N(t) = C^N y^N(t). \tag{5.14}$$

The finite dimensional approximations of the compensator equation (5.11) and control law (5.12) are given by

$$\dot{\tilde{y}}^N(t) = A^N\tilde{y}^N(t) + G^N(\tilde{y}^N(t)) + B^N u^N(t) + L^N[z^N(t) - C^N\tilde{y}^N(t)], \qquad \tilde{y}^N(0) = \tilde{y}_0^N, \tag{5.15}$$

$$u^N(t) = -K^N\tilde{y}^N(t), \tag{5.16}$$

respectively. The approximation to the closed-loop compensator system (which will henceforth be refereed to as full order) is given by

$$\begin{bmatrix} \dot{y}^N(t) \\ \dot{\tilde{y}}^N(t) \end{bmatrix} = \begin{bmatrix} A^N & -B^N K^N \\ L^N C^N & A^N - L^N C^N - B^N K^N \end{bmatrix} \begin{bmatrix} y^N(t) \\ \tilde{y}^N(t) \end{bmatrix} + \begin{bmatrix} G^N(y^N(t)) \\ G^N(\tilde{y}^N(t)) \end{bmatrix}$$

$$\begin{bmatrix} y^N(0) \\ \tilde{y}^N(0) \end{bmatrix} = \begin{bmatrix} y_0^N \\ \tilde{y}_0^N \end{bmatrix} \tag{5.17}$$

Real-time control using the full order compensator may be impossible for many physical problems in that they may require large discretized systems for adequate approximation. Therefore, a reduced order compensator is required. A "reduce-then-design" approach is has a potential drawback that is important physics or information contained in the model can be lost before obtaining the controller. Hence,in this paper, we adopt a"design-then-reduce" approach. In other words, a controller is designed based on the high order model, and then reduced.

Let $y(t) = \sum_{j=1}^{l} \alpha_j(t)\phi_j(x)$ and $\tilde{y}(t) = \sum_{j=1}^{l} \tilde{\alpha}_j(t)\phi_j(x)$ where $\phi_j(x)$'s are the reduced-order basis. The suggested control law is

$$\begin{bmatrix} \dot{y}^l(t) \\ \dot{\tilde{y}}^l(t) \end{bmatrix} = \begin{bmatrix} A^l & -B^l K^l \\ L^l C^l & A^l - L^l C^l - B^l K^l \end{bmatrix} \begin{bmatrix} y^l(t) \\ \tilde{y}^l(t) \end{bmatrix} + \begin{bmatrix} G^l(y^l(t)) \\ G^l(\tilde{y}^l(t)) \end{bmatrix}$$

$$\begin{bmatrix} y^l(0) \\ \tilde{y}^l(0) \end{bmatrix} = \begin{bmatrix} y_0^l \\ \tilde{y}_0^l \end{bmatrix} \tag{5.18}$$

In this work, reduced bases are formed using the CVT process as described in Section 2 and 3. The reduced systems given by (5.18) are compared with the full order compensator system in (5.17).

5.2. Numerical Results. For numerical computations, the viscosity coefficient was taken to be $\nu = 0.01$ and the spatial interval is taken to be $\Omega = [0, 1]$. The time interval is $[0, T]$ where $T = 2$ and $N = 1/64$. The control input operator is $B = \int_0^1 b(x)\phi(x)dx$, where $b(x) = x$ and $\phi(x)$ is a test function. The state weighting operator (used in Riccati equation calculations) $Q = M$, where M is mass matrix. We set the control weighting operator $R(1, 1) = 0.2$ and the weighting operator $\bar{Q}$ is chosen as the mass matrix M. Finally, we creating the measurement matrix C with $Cy(t, x) = 8\int_{3/4}^{7/8} y(t, x)dx$ for the state estimate feedback controller. An initial condition of $y_0(x) = 1$ in $(0, 1/2]$, otherwise 0 is applied. We obtain a standard finite element approximation of the full order PDE with $u = 0$ as in section 3. Then we generate snapshots and CVT bases as in section 4. Simulations of the full order PDE, full state feedback, reduced order full state feedback, compensator and the reduced order compensator are compared. In figure 18, the solutions of the controlled Burgers equation for full FEM and reduced order are shown where full state feedback law is used. In the same way, figure 19 shows the solutions of the controlled Burgers equation for full FEM and reduced order where compensator feedback law is used. Figure 20 shows that reduced-order feedback control methods are quit effective for both full state feedback and state estimate feedback controls. In tables 5 and 6, we report the CPU times and L^2 norms of the solution at $T = 2$ for every cases. One can see that the CPU time for reduced-order model is about 30 times less than that of full FEM with a relative error of 10^{-3}.

Dirichlet boundary feedback controllers with state feedback compensators via CVT will be discussed in another paper.

number of generators	Full FEM	4	8	16
CPU Time	123.17	3.38	5.41	10.44
$\|\|y(T, \cdot)\|\|_2$	0.0104828125	0.0109640625	0.0107000000	0.0106859375

TABLE 5. Full order control *vs*. Reduced order control, full state feedback: $\nu = 1/100$

number of generators	Full FEM	4	8	16
CPU Time	156.06	3.34	5.53	10.36
$\|\|y(T, \cdot)\|\|_2$	0.0104823278	0.0109641436	0.0106992274	0.0106859085

TABLE 6. Full order control *vs*. Reduced order control, estimate feedback: $\nu = 1/100$

References

[1] B. O. Almroth, *Automatic choice of global shape functions in structural analysis*, AIAA J., vol. 16, pp. 525–528, 1978.

[2] J. Atwell and B. King, *Reduced order controllers for spatially distributed systems via proper orthogonal decomposition*, SIAM J. Sci. Comput. Vol. 26, No. 1, pp. 128-151, 2004.

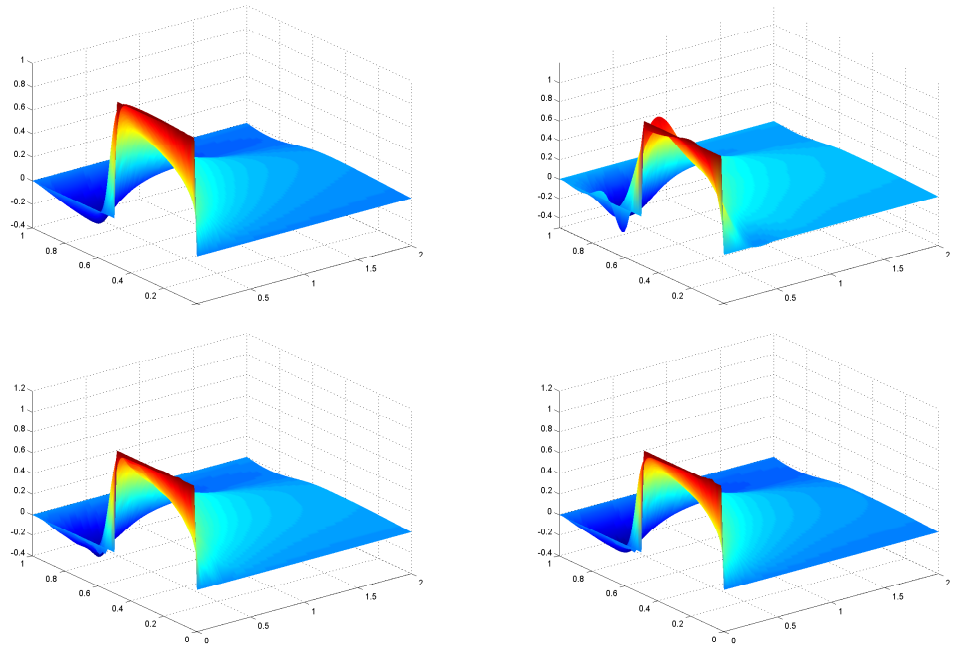

FIGURE 18. Controlled Burgers equation for full state feedback: $\nu = 1/100, , R = 0.2$, Full FEM(top left), 4 CVTs(top right), 8 CVTs(bottom left), 16 CVTs(bottom right).

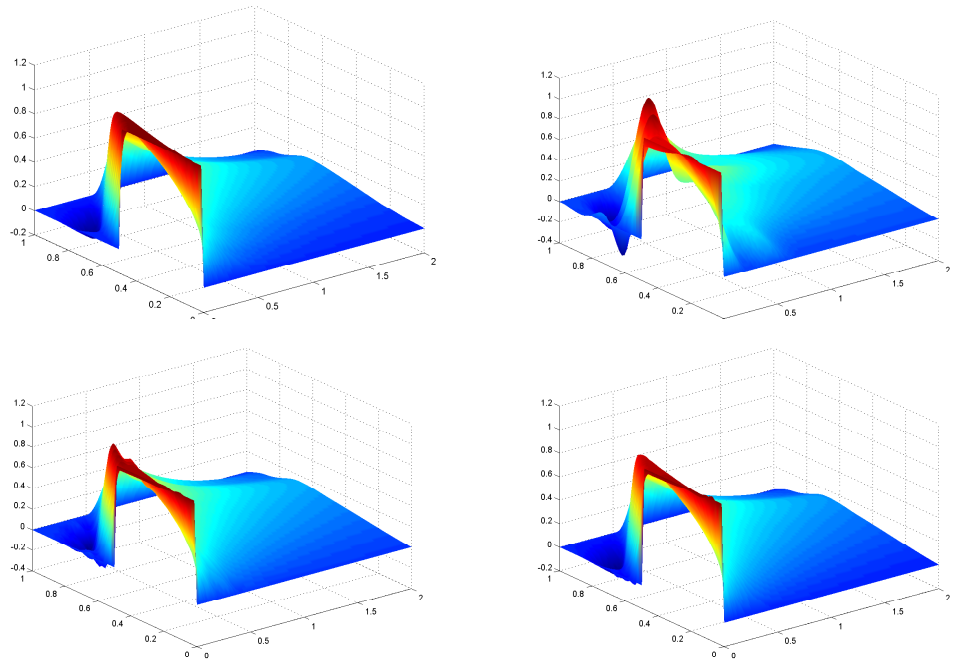

FIGURE 19. Controlled Burgers equation for state estimate feedback: $\nu = 1/100, , R = 0.2$, Full FEM(top left), 4 CVTs(top right), 8 CVTs(bottom left), 16 CVTs(bottom right).

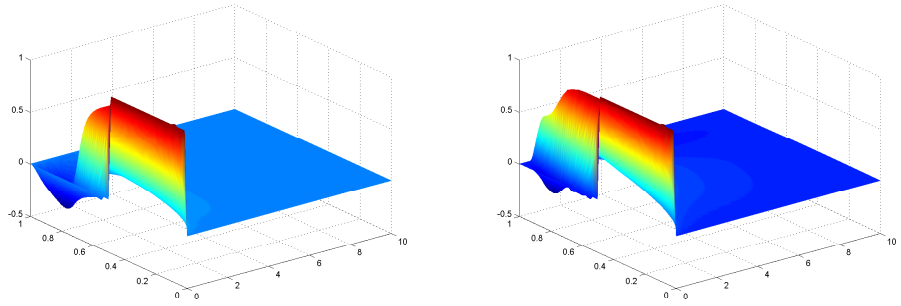

FIGURE 20. Controlled Burgers equation using 8 CVT basis: $\nu = 1/100, , R = 0.2$, $T = 10$, full state feedback(left) and state estimate feedback(right) controls.

[3] G. Berkooz, P. Holmes, and J. L. Lumley, *The proper orthogonal decomposition in the analysis of turbulent flows*, Annual Rev. Fluid Mechanics, vol. 25, pp. 539–575, 1993.

[4] S. C. Brenner and L. R. Scott, *The mathematical theory of finite element methods*, Springer-Verlag, New York, 1994.

[5] Q. Du, V. Faber, and M. Gunzburger, *Centroidal Voronoi tessellations : applications and algorithms*, SIAM Review 41, pp. 637–676, 1999.

[6] Q. Du, M. Emelianenko, and L. Ju, "*Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations*, SIAM J. Numer. Anal., Vol. 44, pp. 102- 119, 2006.

[7] K. Kunisch and S. Volkwein, *Control of Burgers equation by a reduced order approach using proper orthogonal decomposition*, JOTA, no.2, pp.345–371, 1999.

[8] H.-C. Lee, J. Burkardt, and M. Gunzburger, *Centroidal Voronoi tessellation-based reduced-order modeling of complex systems*, to appear in SIAM J. on Scientific Computing.

[9] H.-C. Lee, J. Burkardt, and M. Gunzburger, *POD and CVT-based Reduced-order Modeling of Navier-Stokes flows*, submitted.

[10] S. Lloyd, *Least squares quantization in PCM*, IEEE Trans. Infor. Theory, vol. 28, pp. 129–137, 1982.

[11] J. L. Lumley, *The structure of inhomogeneous turbulent flows*, Atmospheric Turbulence and Wave Propagation, pp. 166–178, 1967.

[12] J. MacQueen, *Some methods for classification and analysis of multivariate observations*, Proc. Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, University of California, pp. 281–297, 1967.

[13] D. A. Nagy, *Modal representation of geometrically nonlinear behavior by the finite element method*, Computers and Structures, vol. 10, pp. 683–688, 1979.

[14] A. K. Noor, *Recent advances in reduction methods for nonlinear problems*, Computers and Structures, vol. 13, pp. 31–44, 1981.

[15] A. K. Noor, C. M. Andersen, and J. M. Peters, *Reduced basis technique for collapse analysis of shells*, AIAA J., vol. 19, pp.393–397, 1981.

[16] A. K. Noor and J. M. Peters, *Reduced basis technique for nonlinear analysis of structures*, AIAA J., vol. 18, pp. 455–462, 1980.

[17] H. M. Park and M. W. Lee, *An efficient computational method of boundary optimal control problems for the Burgers equation*, Compte Methods Appl. Mech. Eng., 166(3-4), pp. 289–308, 1998.

[18] J. S. Peterson, *The reduced-basis method for incompressible viscous flow calculations*, SIAM J. Scientific and Statistical Computing, vol. 10, pp. 777–786, 1989.

[19] S. Volkwein, *Distributed control problems for the Burgers equation*, Computational Optimization and Applications, 18, pp. 115–140, 2001.

Department of Mathematics, Ajou University, Suwon, Korea 443-749

E-mail: hclee@ajou.ac.kr

URL: http://ajou.ac.kr/∼hclee/