SIAM J. Sci. Comput.
Vol. 28, No. 2, pp. 459–484

CENTROIDAL VORONOI TESSELLATION-BASED REDUCED-ORDER MODELING OF COMPLEX SYSTEMS*

JOHN BURKARDT†, MAX GUNZBURGER†, AND HYUNG-CHUN LEE‡

Abstract. A reduced-order modeling methodology based on centroidal Voronoi tessellations (CVTs) is introduced. CVTs are special Voronoi tessellations for which the generators of the Voronoi diagram are also the centers of mass (means) of the corresponding Voronoi cells. For discrete data sets, CVTs are closely related to the h-means and k-means clustering techniques. A discussion of reduced-order modeling for complex systems such as fluid flows is given to provide a context for the application of reduced-order bases. Then, detailed descriptions of CVT-based reduced-order bases and how they can be constructed from snapshot sets and how they can be applied to the low-cost simulation of complex systems are given. Subsequently, some concrete incompressible flow examples are used to illustrate the construction and use of CVT-based reduced-order bases. The CVT-based reduced-order modeling methodology is shown to be effective for these examples.

Key words. reduced-order modeling, Voronoi diagrams, incompressible flows, clustering

AMS subject classifications. 65M99, 35B30, 58F39, 93C20, 93A30, 76D05

DOI. 10.1137/5106482750342221x

1. Introduction.

Even with the use of good mesh generators, discretization schemes, and solution algorithms, the computational simulation of complex, turbulent, or chaotic systems still remains a formidable endeavor. For example, typical finite element codes may require many thousands of degrees of freedom for the accurate simulation of fluid flows. 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. The need for so many degrees of freedom results from two causes. First, phenomena occurring over very small spatial and temporal scales have to be resolved. For example, flows at even moderate values of the Reynolds number are usually turbulent. In a direct computational simulation, very small eddies have to be accurately resolved, even if one is only interested in the large-scale features of the flow. Second, the functions or grid stencils used to effect approximation have no direct relation to the solution of the complex state system. For example, the standard, locally supported, piecewise polynomial basis functions used to define a finite element approximation of the flow solution are not specially designed for a particular flow or for flows in general. Thus, in a typical finite element calculation for a turbulent flow, many thousands of degrees of freedom are needed in order to properly resolve the small eddies.

The natural question to ask is whether or not it is really necessary to have thousands of degrees of freedom in order to produce accurate approximations to solutions of complex systems. For example, for turbulent flows, one would like to identify highly persistent spatio-temporal structures using relatively inexpensive, low-dimensional approaches instead of using expensive standard techniques in the numerical solution

*Received by the editors January 31, 2003; accepted for publication (in revised form) November 3, 2005; published electronically March 31, 2006.
http://www.siam.org/journals/sisc/28-2/42221.html

†School of Computational Science and Information Technology, Florida State University, Tallahassee FL 32306-4120 (burkardt@csit.fsu.edu, gunzburg@csit.fsu.edu). The work of the second author was supported by the National Science Foundation under grant number CCR-9988303.

‡Department of Mathematics, Ajou University, Suwon 443-749, Korea (hclee@ajou.ac.kr). The work of this author was supported by KOSEF R01-2000-000-00008-0.

of partial differential equations.[1] The feedback control of complex systems provides another example; in this setting, low-dimensional state models are needed so that actuation can be determined quickly from sensed data. As a result, there have been many studies devoted to the development, testing, and use of reduced-order models for complex systems such as unsteady fluid flows.

The types of reduced-order models that we study are those that attempt to determine accurate approximate solutions of a complex system using very few degrees of freedom. To do so, such models have to use basis functions that are in some way intimately connected to the problem being approximated. Once a very low-dimensional reduced basis has been determined, one can employ it to solve the complex system by applying, e.g., a Galerkin method. In general, reduced bases are globally supported so that the discrete systems are dense; however, if the reduced basis is of very low dimension, one does not care about the lack of sparsity in the discrete system. Of course, the first natural question to ask is: How does one determine a reduced basis of very low dimension?

There have been many reduced-order modeling techniques proposed. In the structural and fluid mechanics communities, Taylor polynomial and Lagrange polynomial based methods were studied in, e.g., [23, 31] and were shown to be effective for some specific problems. In the Lagrange polynomial approach, one determines a low-dimensional basis by solving the complex system for several values of the parameters that appear in the specification of the system or at several instants in time during the evolution of the system. In the Taylor polynomial approach, the low-dimensional basis consists of the solution of the complex system and of its derivatives with respect to the parameters evaluated at fixed values of the parameters. In both cases, one uses the reduced basis to determine the solution of the complex system at other values of the parameters and, in both cases, the reduced basis is determined by employing high-dimensional discretizations that are, of course, very costly. In simulation settings, that cost is hopefully amortized over subsequent calculations for many other parameter values using the low-cost reduced basis approach. Likewise, in control settings, the cost is amortized over the many state solves necessary to determine the controls.

Today, the most popular reduced-order modeling approach for complex systems is based on proper orthogonal decomposition (POD) analysis. POD begins with a set of $\widetilde{m}$ snapshots; these could be generated by evaluating the computational solution of a transient problem at several instants of time or by evaluating the computational solution for several values of the parameters appearing in the problem description or by a combination of the two. The snapshot set can even be determined, in principle, from experiments. The computational solutions that are used to determine the snapshot set are determined using costly, large-scale, high-fidelity, e.g., finite element, codes. The $\widetilde{d}$-dimensional POD basis is then determined from the $\widetilde{d}$ eigenvectors corresponding to the $\widetilde{d}$ most dominant eigenvalues of the correlation matrix for the snapshots.[2] The POD basis is then used, usually by applying a projection procedure,

[1]The whole field of turbulence modeling, e.g., k-ϵ and large-eddy simulation (LES) models, is an attempt at reduced-order modeling. However, these approaches still result in complex systems of partial differential equations that are usually solved by employing very high-dimensional approximation schemes. The approaches to reduced-order modeling considered here are different and result in very low-dimensional models for complex systems.

[2]Perhaps more simply, a totally equivalent definition of the $\widetilde{d}$-dimensional POD basis is the first $\widetilde{d}$ left singular vectors of the matrix whose columns are the snapshot vectors. In statistical analyses, POD is known as the Karhunen–Loève analysis or the method of empirical orthogonal eigenfunctions or principal component analysis.

to determine an approximate solution for different values of the system parameters. For example, in an optimal control or optimization setting, one would generate the POD basis from snapshots that correspond to several sets of values of the optimization or control parameters. These snapshots are determined using expensive, e.g., finite element, codes having many thousands of degrees of freedom. However, the snapshots can be generated "off-line" and their costs can be hopefully amortized over several optimization calculations. When one performs the optimization or control of the system, one uses the small-dimensional POD basis to determine the multiple state solutions and any sensitivity or adjoint equations solutions needed by the optimizer or by the controller. The POD basis possesses a well-known optimization characteristic. POD-based model reduction has been applied with some success to several problems, most notably in fluid mechanics settings. For detailed discussions, one may consult [1, 2, 3, 4, 5, 15, 17, 18, 20, 21, 22, 25, 26, 27, 28, 29, 30, 32, 33, 34, 35, 36, 37, 38, 41, 42]. A few more details about how snapshots are generated and how a POD basis is used for model reduction are also given in section 1.1.

Now that we have discussed two approaches for generating reduced bases, another natural question to ask is when can one expect a reduced-order model to "work," i.e., to produce accurate, very low-dimensional approximations to a solution of a complex system. Clearly, they can work only when the low-dimensional basis "contains" all of the information needed to produce an accurate approximation, i.e., in an interpolatory setting or in the very limited settings where extrapolation can work. In the POD setting, the snapshots used to define the POD basis must contain all the information, e.g., the dynamic behavior, of the solution one is trying to obtain. The beauty of POD, when it works, is that it can reduce a set of hundreds of snapshots to a set of just a handful of POD basis vectors that contain essentially the same information. However, when the snapshot set does not contain the needed information, one has to generate new snapshots and then generate a new POD basis, often in an adaptive manner. This can happen when one uses a reduced-basis method in an optimization or control setting. Initially, the snapshots, and therefore the POD basis, can be constructed to contain the needed information, but as the optimizer or controller changes the parameters in the problem, it is often the case that the original snapshot set does not contain enough information to accurately approximate the solution at the new values of the parameters, and thus the snapshot set has to be augmented with new snapshots that reflect the new behavior. Even worse, it may be the case that the POD basis, which included most of the information contained in the snapshot set for the parameters used to generate the snapshots, ignored information in the snapshot set that may be useful for other values of the parameters. Furthermore, redoing the POD basis when the snapshot set changes requires the solution of a new eigenvalue or singular value problem which is of the size of the cardinality of the snapshot set.

Centroidal Voronoi tessellations (CVTs) have been successfully used in several data compression settings, e.g., in image processing and the clustering of data.[3] Reduced-order modeling of complex systems is another data compression setting, i.e., replacing high-dimensional approximations with low-dimensional ones. Thus, one can

[3]CVTs are useful in many other settings, e.g., just to name some, optimal quadrature rules, optimal representation and quantization, cell division, data compression, optimal distribution of resources, territorial behavior of animals, optimal placement of sensors and actuators, grid generation in two and three dimensions and on surfaces, and meshfree methods. For further discussions about the applications of CVTs, see, e.g., [6, 7, 8, 9, 10, 11, 12, 13, 14, 19, 24]. Also, like POD, CVT is closely related to well-known statistical techniques; in fact, for discrete data sets, CVT can be identified with the h-means and k-means clustering methods.

ask if CVTs can be used for the reduced-order modeling of complex systems. In CVT-reduced order modeling, 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.

The efficiency of CVT- and POD-based reduced bases depends on the their dimension, i.e., if a CVT- or POD-based basis is low-dimensional and can still approximate the state well, then approximations of the state of a complex system can be inexpensively determined. However, the ability of a CVT- or POD-based basis to approximate the state of a system is totally dependent on the information contained in the snapshot set used to generate the basis. Certainly, a CVT- or POD-based basis cannot contain more information than that contained in the snapshot set.[4] Thus, crucial to the success of both the CVT- and POD-based approaches to model reduction is the generation of "good" snapshot sets.

Unfortunately, at this time, the generation of snapshot sets is an art and not a science; it is, in fact, a rather primitive art. One recognizes that the generation of snapshot sets is an exercise in the design of experiments. For example, for stationary systems, how does one choose the sets of parameters at which the state and perhaps sensitivities are calculated using expensive, high-fidelity computations in order to generate the snapshot set? Clearly, some a priori knowledge about the types of states to be simulated or optimized using the reduced-order model is very useful in this regard. The large body of statistics literature on the design of experiments has not been used in a systematic manner. For time-dependent systems, many ad hoc measures have been invoked in the hope that they will lead to good snapshot sets. Time-dependent parameters, e.g., appearing in boundary conditions, are used to generate states that are "rich" in transients, even if the state of interest depends only on time-independent parameters. Moreover, in order to generate even "richer" dynamics, impulsive forcing is commonly used, e.g., starting the evolution impulsively with different strength impulses or introducing impulses in the midst of a simulation. In section 3, we adopt some of these strategies for the generation of snapshot sets in the concrete examples we consider. However, generating snapshot sets is not a main focus of this paper. It is clear, though, that in the future, a great deal of effort needs to be directed towards developing and justifying methodologies for generating good snapshot sets. After all, a POD or CVT basis can only be as good as the snapshot set used to generate it.

1.1. Schematic of POD- or CVT-based model reduction for simulation problems. For the sake of concreteness, we suppose that we wish to solve the problem:

$$\text{given } \theta \in \Theta, \text{ find } u(t,x) \in U \text{ such that } N(u;\theta;v) = 0 \;\forall\, v \in U, \text{ a.e.}\, t \in (0,T)\,, \tag{1.1}$$

where U and Θ are given solution[5] and parameter function spaces, respectively, and

[4]Recall that the goal of invoking POD or CVT is merely to determine a low-dimensional basis that "contains" almost all the information contained in the snapshot set.

[5]In general, the spaces of test and trial functions are not the same, i.e., in (1.1) the test functions v do not, in general, belong to the same space as do the trial functions u. For the sake of simplicity of exposition, we here assume that they are the same.

$(0,T)$ is a specified time interval.[6] If (1.1) represents a time-dependent problem, then it should be supplemented with initial conditions.[7] The mapping $N : U \times \Theta \times V$ is linear in its third argument and is generally nonlinear in its first two arguments; $N(\cdot;\cdot;\cdot)$ in general involves space and time derivatives and integrals of combinations of its arguments. In (1.1), θ denotes a set of parameters that serve to specify the problem, u denotes the desired solution, and v denotes a suitable test function. Thus, (1.1) implicitly defines a solution mapping $u(\theta) : \Theta \to U$. The object of a computational simulation is to find approximations of this mapping.

For the sake of even further concreteness, we consider finite element methods for the approximation of solutions of (1.1). To this end, one chooses a finite-dimensional subspace[8] $U^h \subset U$ parameterized by a parameter h tending to zero (e.g., the grid size) and then one poses the approximate problem:

$$\begin{aligned}&\text{given } \theta \in \Theta, \text{ find } u^h(t,x) \in U^h \text{ such that}\\ &N(u^h;\theta;v^h) = 0 \quad \forall\, v^h \in U^h, \ \text{a.e.}\, t \in (0,T)\,.\end{aligned} \tag{1.2}$$

The approximate solution is computed by choosing a basis $\{\phi_k(x)\}_{k=1}^{\widetilde{j}}$, then writing $u^h(t,x) = \sum_{k=1}^{\widetilde{j}} U_k(t)\phi_k(x)$, and then solving the problem:[9]

$$\begin{aligned}&\text{given } \theta \in \Theta, \ \text{ find } U_k(t), \ k = 1\ldots,\widetilde{j}, \text{ such that}\\ &N\left(\textstyle\sum_{k=1}^{\widetilde{j}} U_k\phi_k;\theta;\phi_j\right) = 0 \quad \text{for } j = 1,2,\ldots,\widetilde{j}, \ t \in (0,T)\,.\end{aligned} \tag{1.3}$$

In general, for complex systems of practical interest, the dimension $\widetilde{j}$ of the approximating space U^h may be in the many thousands. As a result, it may be very expensive to solve (1.3) for a single set θ of parameter values and it may be prohibitively expensive to solve (1.3) for many sets of parameter values. The latter is exactly the situation which causes the interest in reduced-order modeling so that we assume that we want to find approximations of the solutions of (1.1) for many sets of parameter values. We now describe the steps that go into defining and using POD- or CVT-type reduced-order models for finding approximations to the solutions of (1.1).

1.1.1. Generating snapshots. We begin by choosing several parameter sets $\theta_1, \theta_2, \ldots, \theta_{\widetilde{\ell}}$ and then solving (1.3) for each set, i.e., we solve the $\widetilde{\ell}$ problems:

$$\begin{aligned}&\text{for } \ell = 1,\ldots,\widetilde{\ell}, \ \text{ find } U_k^\ell, \ k = 1\ldots,\widetilde{j}, \text{ such that}\\ &N\left(\textstyle\sum_{k=1}^{\widetilde{j}} U_k^\ell\phi_k;\theta_\ell;\phi_j\right) = 0 \quad \text{for } j = 1,2,\ldots,\widetilde{j}\,.\end{aligned} \tag{1.4}$$

Note that, in general, for each ℓ, (1.4) represents the solution of a large nonlinear system of ordinary differential equations, or in the stationary case, a large system of

[6]Using (1.1) as our prototype complex system prejudices our discussion towards Galerkin-finite element and Galerkin-spectral methods. However, one could easily extend our discussion to other methods, including finite difference and finite volume methods.

[7]If (1.1) represents a stationary problem, we simply suppress all dependencies on t; for the sake of concreteness, we will assume that (1.1) represents a time-dependent problem.

[8]One may also need to approximate the parameter space in which case one also has to choose an approximating parameter subspace $\Theta^h \subset \Theta$. However, for the sake of simplicity of the exposition, we assume that this is not necessary for the problem we consider. It is also possible that the mapping N has to be replaced by an approximate mapping N^h, but we ignore that possibility as well. We note that there is no difficulty extending our discussion to the more general cases.

[9]So far, we have only effected spatial discretization. If the mapping N involves time derivatives of u, then the problem (1.3) is a system of nonlinear ordinary differential equations (ordinary differential equations) for $U_k(t), \ k = 1\ldots,\widetilde{j}$. Thus, further discretization in time is usually needed to solve (1.3).

discrete nonlinear equations. Therefore, generating snapshots is an expensive proposition.

Next, we choose several time values $t_1, t_2, \ldots, t_{\widetilde{q}}$ within the interval $[0, T]$ and we evaluate the coefficients $U_k^\ell(t)$, $k = 1 \ldots, \widetilde{j}$ and $\ell = 1, \ldots, \widetilde{\ell}$, at these time values. Using this information, we define the $\widetilde{m} = \widetilde{q}\widetilde{\ell}$ *snapshot vectors*

$$\vec{U}_m = \begin{pmatrix} U_1^\ell(t_q) \\ U_2^\ell(t_q) \\ \vdots \\ U_{\widetilde{j}}^\ell(t_q) \end{pmatrix}, \quad m = (\ell - 1)\widetilde{q} + q, \quad q = 1 \ldots, \widetilde{q}, \text{ and } \ell = 1, \ldots, \widetilde{\ell}. \tag{1.5}$$

Note that the snapshot vectors $\vec{U}_m \in \mathbb{R}^{\widetilde{j}}$, where $\widetilde{j}$ is the dimension of the finite element space being employed which may be in the many thousands. On the other hand, the number of snapshots $\widetilde{m} = \widetilde{q}\widetilde{\ell}$ is usually taken to be of the order of hundreds.

The process just described for generating snapshots is a well defined recipe except that it requires the choosing of $\widetilde{\ell}$ parameter sets $\theta_1, \ldots, \theta_{\widetilde{\ell}}$ and $\widetilde{q}$ values of time $t_1, \ldots, t_{\widetilde{q}}$. As has already been mentioned, how these choices are made is not something that is a systematic science. In section 3, we provide some examples of how these choices are made.

There are definitions other than (1.5) that have been used to define snapshot sets. For example, one could add to the snapshots in (1.5) approximations to the time derivatives of the coefficients $U_k^\ell(t)$, i.e., in addition to (1.5), we have the snapshot vectors

$$\vec{U}_m \approx \begin{pmatrix} \left.\frac{dU_1^\ell}{dt}\right|_{t_q} \\ \vdots \\ \left.\frac{dU_{\widetilde{j}}^\ell}{dt}\right|_{t_q} \end{pmatrix}, \quad m = \widetilde{\ell}\widetilde{q} + (\ell - 1)\widetilde{q} + q, \quad q = 1 \ldots, \widetilde{q}, \text{ and } \ell = 1, \ldots, \widetilde{\ell}$$

so that now we have a set of snapshots of cardinality $\widetilde{m} = 2\widetilde{\ell}\widetilde{q}$. The approximation to the time derivatives may be effected, e.g., by using finite difference quotients.

1.1.2. Generating reduced bases from the snapshot set. Snapshot sets by themselves are not always directly useful for reduced-order modeling. First, in order to include "enough" useful information about solutions of complex systems, one usually has on the order of hundreds of snapshot vectors which is too high a number for practicality. Second, snapshot sets may be linearly dependent or may be poorly conditioned as a basis so that, again, they may not be practical. On the other hand, snapshot sets often contain much redundant information so that it should be possible to define smaller basis sets of much lower cardinality that are well conditioned and that have nearly the same information content as the snapshot set. POD and CVT analyses can be applied for this purpose.

Before applying POD or CVT analyses, the snapshot set is usually modified. For example, one usually subtracts the mean of the snapshot set before further processing, e.g., instead of using (1.5), one uses the modified snapshots in which the mean of the snapshots is subtracted from every snapshot:

$$\vec{V}_m = \vec{U}_m - \overline{U} \quad \text{for } m = 1, \ldots, \widetilde{m}, \qquad \text{where} \qquad \overline{U} = \frac{1}{\widetilde{m}} \sum_{m=1}^{\widetilde{m}} \vec{U}_m\,.$$

In addition, snapshot vectors are sometimes manipulated so as to efficiently account for inhomogeneous boundary conditions. In any case, we assume that we now have in hand a satisfactory snapshot set $\mathcal{V} = \{\vec{V}_m\}_{m=1}^{\widetilde{m}}$ from which we want to define a low-dimensional reduced basis.

Let us briefly consider POD-based reduced bases. There are many equivalent ways to define POD bases corresponding to a given snapshot set. Perhaps the simplest definition is as follows. First, we form the snapshot matrix

$$S = (\vec{V}_1, \vec{V}_2, \cdots, \vec{V}_{\widetilde{m}}),$$

i.e., the matrix whose columns are the snapshot vectors; the ordering does not matter. Note that S is a $\widetilde{j} \times \widetilde{m}$ matrix so that, in practice, it has many more rows than columns. Then, the $\widetilde{d}$-dimensional POD basis consists of the first $\widetilde{d}$ left singular vectors of S. Note that the POD basis vectors belong to $\mathbb{R}^{\widetilde{j}}$ and that the POD basis is perfectly conditioned since, by definition, it is an orthonormal basis. It has also been found in many settings that POD bases of low dimension, i.e., of the order of ten or so, contain almost all the information present in snapshot sets consisting of hundreds of vectors.

Instead of POD, in this paper we consider using CVT analyses to generate a low-dimensional reduced basis from a given set of snapshots. This process is described in detail in section 2.

1.1.3. Using a reduced basis to solve a simulation problem. Suppose now that we have in hand a $\widetilde{d}$-dimensional reduced basis $\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$, be it POD-based or CVT-based. The basis vectors $\vec{Z}_i$ belong to $\mathbb{R}^{\widetilde{j}}$ and consist of the coefficients of particular (globally supported) functions belonging to the finite element trial space U^h. In fact, if we write

$$\vec{Z}_i = \begin{pmatrix} Z_{i,1} \\ Z_{i,2} \\ \vdots \\ Z_{i,\widetilde{j}} \end{pmatrix} \qquad \text{for } i = 1, \dots, \widetilde{d},$$

the corresponding finite element functions are given by

$$z_i(x) = \sum_{j=1}^{\widetilde{j}} Z_{i,j}\phi_j(x) \qquad \text{for } i = 1, \dots, \widetilde{d}, \tag{1.6}$$

where $\{\phi_j(x)\}_{j=1}^{\widetilde{j}}$ is the basis for the finite element space U^h, and the $\widetilde{d}$-dimensional reduced basis can be written in the form $\{z_i(x)\}_{i=1}^{\widetilde{d}}$. The reduced-order space $Z^h = \text{span}\,\{z_1(x), \dots, z_{\widetilde{d}}(s)\} \subset U^h$. Note that $\dim Z^h = \widetilde{d}$ which is very small compared to the dimension $\widetilde{j}$ of the finite element space used to define the discrete problem (1.2).

Having obtained a reduced basis, we now want to use it to determine an approximate solution of (1.1) for new parameter sets θ. We use exactly the same process that led to (1.2) and (1.3). Since we now seek an approximate solution $u_{ro}^h(t,x)$ belonging to the reduced-order space Z^h for all $t \in (0,T)$, we solve the problem:

$$\begin{aligned} &\text{given } \theta \in \Theta, \text{ find } u_{ro}^h(t,x) \in Z^h \text{ such that} \\ &N(u_{ro}^h; \theta; v^h) = 0 \quad \forall\, v^h \in Z^h, \ \text{a.e.}\, t \in (0,T)\,. \end{aligned} \tag{1.7}$$

Since $u^h_{ro}(t,x) \in Z^h$, we can write $u^h_{ro}(t,x) = \sum_{i=1}^{\widetilde{d}} c_i(t) z_i(x)$ for some time-dependent coefficients c_i, $i = 1,\ldots,\widetilde{d}$. Then, (1.7) is equivalent to the problem of finding the coefficients $c_i(t)$, $i = 1,\ldots,\widetilde{d}$, from the nonlinear system of ordinary differential equations:

$$\begin{aligned}&\text{given } \theta \in \Theta, \ \text{ find } c_i(t), \ i = 1\ldots,\widetilde{d}, \text{ such that}\\ &N\left(\textstyle\sum_{i=1}^{\widetilde{d}} c_i z_i; \theta; z_j\right) = 0 \quad \text{for } j = 1,2,\ldots,\widetilde{d}, \ t \in (0,T)\,.\end{aligned} \tag{1.8}$$

The reduction in size between (1.3) and (1.8) is now explicitly evident. The former is a system of $\widetilde{j}$ nonlinear ordinary differential equations for the $\widetilde{j}$ unknowns $U_k(t)$, $k = 1,\ldots,\widetilde{j}$, while (1.8) is a system of $\widetilde{d}$ ordinary differential equations for the $\widetilde{d}$ unknowns $c_i(t)$, $i = 1,\ldots,\widetilde{d}$. Thus, having produced the reduced basis $\{z_i(x)\}_{i=1}^{\widetilde{d}}$, we can now solve (1.8) for any given parameter set θ much more cheaply than if we resorted to solving (1.3).

Using (1.6), we see that the solution of (1.7) or, equivalently, of (1.8) can be written as

$$u^h_{ro}(t,x) = \sum_{k=1}^{\widetilde{j}} \left(\sum_{i=1}^{\widetilde{d}} c_i(t) Z_{i,k} \right) \phi_k(x) \tag{1.9}$$

so that clearly $u^h_{ro} \in U^h$, i.e., the solution of (1.7) lives in the finite element space. However, only $\widetilde{d}$ degrees of freedom (the c_i's) completely determine $u^h_{ro} \in U^h$. If we substitute (1.9) into (1.7) or, equivalently, if we substitute (1.6) into (1.8), we arrive at another equivalent form for the reduced-order problem:

$$\begin{aligned}&\text{given } \theta \in \Theta, \ \text{ find } c_i(t), \ i = 1\ldots,\widetilde{d}, \text{ such that}\\ &N\left(\textstyle\sum_{k=1}^{\widetilde{j}} \left(\sum_{i=1}^{\widetilde{d}} c_i Z_{i,k}\right) \phi_k(x); \theta; \sum_{j=1}^{\widetilde{j}} Z_{m,j}\phi_j(x)\right) = 0\\ &\text{for } m = 1,2,\ldots,\widetilde{d}, \ t \in (0,T)\,.\end{aligned} \tag{1.10}$$

This form of the reduced-order problem is written in terms of the standard finite element functions ϕ_j and the components of the reduced basis vectors $Z_{i,j}$; it is the form that is actually used in practice to assemble the $\widetilde{d}$ ordinary differential equations corresponding to the reduced-order problem.

To complete the definition of the reduced-order problem, one must choose a solution method for the system of ordinary differential equations (1.10).

We have now given an almost complete recipe for how a reduced basis is determined and used to find low-order approximations of complex systems; the missing step is how a reduced basis is determined from the snapshot set. We briefly described how this step is carried out in the POD setting, and in the next section, we describe how it is done in the CVT setting.

2. Centroidal Voronoi tessellation-based reduced bases.

The generation of a CVT-based reduced-order basis is the intermediate step of the model reduction algorithm. It follows the generation of the snapshot set, creating a set of basis vectors which are used in the subsequent step to determine approximations of the solution of the complex system for new physical states, i.e., new sets of parameters. We now provide a definition of a CVT-based reduced basis and then discuss how they are constructed.

2.1. Definition of centroidal Voronoi tessellations for discrete data sets. The definition of centroidal Voronoi tessellations (CVTs) for discrete data sets begins with a set $\mathcal{V} = \{\vec{V}_m\}_{m=1}^{\widetilde{m}}$ consisting of $\widetilde{m}$ vectors belonging to $\mathbb{R}^{\widetilde{j}}$. Of course, $\mathcal{V}$ can also be viewed as a set of $\widetilde{m}$ points in $\mathbb{R}^{\widetilde{j}}$. The definition also requires us to introduce the notion of clustering or tessellation, which in our context is defined to be a subdivision of the set $\mathcal{V}$ into $\widetilde{d} \leq \widetilde{m}$ subsets $\mathcal{V}_i$, $i = 1, \dots \widetilde{d}$, such that $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$ for $i \neq j$ and $\cup_{i=1}^{\widetilde{d}} \mathcal{V}_i = \mathcal{V}$. The set of subsets $\{\mathcal{V}_i\}_{i=1}^{\widetilde{d}}$ is called a *clustering* or a *tessellation* of the set $\mathcal{V}$. One way to determine a clustering is to first choose a set $\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ consisting of $\widetilde{d} \leq \widetilde{m}$ distinct vectors,[10] also belonging to $\mathbb{R}^{\widetilde{j}}$, and then choose a function $C(\vec{V}) = \vec{Z}$ that assigns each $\vec{V}_m \in \mathcal{V}$ to a particular $\vec{Z}_i \in \mathcal{Z}$, and then letting $\mathcal{V}_i = \{\vec{V}_m \in \mathcal{V}$ such that $C(\vec{V}_m) = \vec{Z}_i\}$ for $i = 1, \dots, \widetilde{d}$. We call the collection $(\mathcal{V}, C, \mathcal{Z})$ a *cluster configuration.* The vectors $\vec{Z}_i$ are called *cluster generators* and the subsets $\mathcal{V}_i$ are called *clusters.* The cardinality of cluster $\mathcal{V}_i$ is denoted by $\overline{m}_i$; clearly, $\sum_{i=1}^{\widetilde{d}} \overline{m}_i = \widetilde{m}$, the cardinality of the set $\mathcal{V}$. The vectors

$$\overline{Z}_i = \frac{1}{\overline{m}_i} \sum_{\vec{V}_m \in \mathcal{V}_i} \vec{V}_m \qquad \text{for } i = 1, \dots, \widetilde{d}$$

are called the *cluster means* or *cluster centroids.*

For a given set $\mathcal{V}$, a *Voronoi tessellation* or *Voronoi clustering* of $\mathcal{V}$ is a particular type of cluster configuration for which the cluster generators $\{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ may be chosen arbitrarily and for which the clustering rule C assigns each $\vec{V}_m \in \mathcal{V}$ to the nearest[11] $\vec{Z}_i \in \mathcal{Z}$:

$$C(\vec{V}_m) = \vec{Z}_i \quad \Longrightarrow \quad |\vec{V}_m - \vec{Z}_i| = \min_{\vec{Z}_j \in \mathcal{Z}} |\vec{V}_m - \vec{Z}_j|\,. \tag{2.1}$$

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

$$\vec{Z}_i = \overline{Z}_i\,, \tag{2.2}$$

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 (2.2) so that, for given a set $\mathcal{V}$ and positive integer $\widetilde{d}$, a CVT must be constructed. Algorithms for this purpose are discussed in section 2.2.

CVTs possess an optimization property that can be used as a basis for an alternate definition. Given a given set $\mathcal{V}$ of $\widetilde{m}$ vectors in $\mathbb{R}^{\widetilde{j}}$, a positive integer $\widetilde{d} \leq \widetilde{j}$, and a clustering configuration $(\mathcal{V}, C, \mathcal{Z})$, let

$$E(\mathcal{V}, C, \mathcal{Z}) \equiv \sum_{m=1}^{\widetilde{m}} |\vec{V}_m - C(\vec{V}_m)|^2 = \sum_{m=1}^{\widetilde{d}} \sum_{\vec{V}_m \in \mathcal{V}_i} |\vec{V}_m - \vec{Z}_i|^2 \tag{2.3}$$

[10] The vectors in the data set $\mathcal{V}$ need not be distinct.

[11] Of course, there may be more than one vector in $\mathcal{Z}$ that is nearest a $\vec{V}_m$ so that a tie-breaking rule has to be appended to (2.1). For example, among all vectors in $\mathcal{Z}$ that satisfy (2.1), we could choose the one with the smallest index value.

denote *cluster "energy."* Then, it can be shown (see [6]) that among all possible assignment rules $C(\cdot)$ and all possible cluster generators $\{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$, i.e., among all possible cluster configurations, cluster energy minimizers are CVT cluster configurations.

One easily recognizes that (2.3) defines the variance of the clustering so that, as described here, CVTs are minimum variance clusterings of discrete point sets. Thus, in the context discussed here, CVTs are well known in the statistical literature under names like k-means clustering. However, CVTs can be generalized in many ways, e.g., to continuous sets, to other metrics, to weighted energy functionals, etc. Since the latter generalization can be useful in the model reduction setting, we will briefly discuss it in section 4.

The connection between CVTs and reduced bases is now easily made. The set $\mathcal{V}$ is obviously the set of (possibly modified) snapshots. Then, the CVT reduced basis set is the set of generators $\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ of a CVT of $\mathcal{V}$.

2.2. Algorithms for constructing discrete CVTs. Even for moderate values of $\widetilde{m}$ and $\widetilde{d}$, the number of possible clusterings of the data is enormous, meaning that an exhaustive search for minimizers of the cluster energy cannot be carried out. Fortunately, there do exist good algorithms for determining CVTs of discrete point sets. We use the h-means and k-means methods (see, e.g., [16, 39, 40]) which are described in section 2.2.1. The h-means method is cheaper to use than k-means, but does not minimize the cluster energy function directly, and so it is not completely reliable. Since the two methods are compatible, an efficient hybrid technique consists of applying h-means until it can achieve no further energy reduction, followed by (presumably few) steps of k-means to drive the energy down.

The h-means and k-means algorithms have the attractive property of being *descent methods* with respect to the cluster energy. Given any cluster configuration $(\mathcal{V}, C, \mathcal{Z})$, both algorithms consider a series of modifications to the cluster assignment function C. However, a modification is only made if it is guaranteed to reduce the cluster energy $E(\mathcal{V}, C, \mathcal{Z})$. Thus, an algorithm based on taking h-means or k-means steps cannot diverge or produce a final clustering having higher energy than the initial clustering. Also, since the set $\mathcal{V}$ is finite dimensional, the algorithms are guaranteed to terminate in finitely many steps. By virtue of the nearest neighbor and centroidal generator features, the resulting clusters will be discretely convex and will tend to correspond to natural clusterings of the data.

The single step approach of the h-means and k-means methods, i.e., the fact that the methods consider one member of the data set $\mathcal{V}$ at a time, has the disadvantage of making the methods vulnerable to "local minima."[12] The k-means algorithm will never allow a one-point modification that increases the cluster energy, and so it is guaranteed to halt if it reaches a local minimum. However, for a given problem, there can be many local minima that are not global minimizers of the cluster energy. To deal with the problem of local minima, one can simply apply the CVT construction algorithm several times, using different initial configurations, and then take the final configuration having the lowest cluster energy to be the best estimate of the optimal configuration. One should also note that in other applications of CVT it has been found the local minima determined by, say, the k-means method yield very good

[12]Given a nearest-neighbor, i.e., a Voronoi, cluster configuration $(\mathcal{V}, C, \mathcal{Z})$, define a *one point change* as a change, for a single data point $\vec{V}_m$, in the value of the cluster assignment function from $C(\vec{V}_m) = \vec{Z}_i$ to $C(\vec{V}_m) = \vec{Z}_j$, followed by the replacement of $\vec{Z}_i$ and $\vec{Z}_j$ by the centroids of their new associated clusters. A *local minimum* is then a cluster configuration $(\mathcal{V}, C, \mathcal{Z})$ with the property that every one-point change results in an increase in the cluster energy.

results; this is borne out by the computational results of this paper which show that CVT reduced-order modeling can provide effective low-dimensional approximations.

2.2.1. The h-means and k-means algorithms. The h-means algorithm begins with an initial cluster configuration $(\mathcal{V}, C, \mathcal{Z})$. Then, for each point $\vec{V}_m \in \mathcal{V}$, the algorithm finds the nearest cluster generator $\vec{Z}_i \in \mathcal{Z}$; if that is not the cluster to which the data point is currently assigned, the data point is "transferred" to the cluster associated with its nearest generator. When all data points have been considered, the cluster generators $\vec{Z}$ are replaced by the centroids of the newly formed clusters. If at least one point was transferred during this process, a new sweep of all the points is carried out, and this continues until a sweep is made in which no point is transferred.

Recall that:

$\mathcal{V} = \{\vec{V}_m\}_{m=1}^{\widetilde{m}}$ denotes the fixed set of points to be clustered into a CVT consisting of $\widetilde{d}$ clusters, where $\widetilde{d}$ is also fixed;

$\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ denotes the cluster generators which can possibly change at each step of the algorithm;

$\{\mathcal{V}_i\}_{i=1}^{\widetilde{d}}$ denotes the clustering of $\mathcal{V}$ which can also possibly change at each step of the algorithm; and

$\overline{d}_i$ denotes the cardinality of the cluster $\mathcal{V}_i$ which can also possibly change at each step of the algorithm.

Then, the h-means algorithm may be formalized as follows:

```
do
    for each $\vec{V}_m \in \mathcal{V}$
        determine $\vec{Z}_i \in \mathcal{Z}$ that minimizes $|\vec{V}_m - \vec{Z}_j|$
        if $\vec{V}_m \notin \mathcal{V}_i$, then
            reassign $\vec{V}_m$ to $\mathcal{V}_i$
        end if
    end for
    if no transfers occurred, exit the loop
    for each $i = 1, \dots, \widetilde{d}$
        $\vec{Z}_i = \frac{1}{\overline{d}_i} \Sigma_{\vec{V}_m \in \vec{Z}_i} \vec{V}_m$
    end for
end do
```

The k-means algorithm is a sophisticated version of h-means. The h-means transfer test is based on the distance from a point to a cluster center. However, if a point is transferred to a new cluster, the cluster center, when it is recomputed, will move closer to the point, something that h-means misses until the step is complete, i.e., until all points in $\mathcal{V}$ are treated. Within each iteration, the k-means algorithm works with a transfer test that directly measures the change in energy that occurs when a point moves from one cluster to another and then immediately redetermines the cluster centroids before moving to the next point in $\mathcal{V}$. A single k-means sweep is computationally more expensive than a single h-means sweep. This added expense is the reason why the h-means algorithm is employed first to perform the "easy" transfers.[13] The k-means algorithm may be formalized as follows:

[13]How much more expensive is k-mean compared to h-mean depends on exactly how one carries out the steps in the two methods, e.g., how one calculates nearest neighbors. Our limited experience has indicated that it is more efficient to use k-means, especially near the end of the CVT construction process.

```
do
    for each $\vec{V}_m \in \mathcal{V}$
        evaluate the cluster energy for all possible transfers of $\vec{V}_m$ from its
            current cluster $\mathcal{V}_i$ to any of the other clusters $\mathcal{V}_j$, $j = 1, \dots, \widetilde{d}$,
            $j \neq i$
        if moving $\vec{V}_m$ from its current cluster $\mathcal{V}_i$ to the cluster $\mathcal{V}_k$ will most
            reduce the cluster energy, then
                reassign $\vec{V}_m$ to cluster $\mathcal{V}_k$
                replace the cluster generators $\vec{Z}_i$ and $\vec{Z}_k$ by the centroids of
                the newly modified clusters $\mathcal{V}_i$ and $\mathcal{V}_k$, respectively
        end if
    end for
    if no transfers occurred, exit the loop
end do
```

The main lines of a hybrid algorithm can now be made clear. Starting with a randomly initialized cluster configuration, one applies the h-means iteration (presumably many times) and the k-means iteration (hopefully a few times) to produce a locally optimal cluster configuration. One repeats the process several times, each time starting with a different random initial cluster configuration. The generators of the locally optimal configuration having the lowest cluster energy are then output as the approximation to the desired CVT reduced-order basis.

We note that there are several ancillary processing issues that arise in the implementation of CVT construction algorithms. For example, one may wish to thin, shift, or scale the snapshots so that clustering patterns are more readily exposed. One may also want to use, instead of the Euclidean one, metrics that do not differentiate between snapshot vectors which point in nearly the same direction but which have different magnitudes. Additionally, the choice of the initial set of generators can affect the efficiency of a CVT construction algorithm.

3. Computational experiments.

To demonstrate the use, efficiency, and accuracy of the CVT-based reduced-order modeling technique, two examples are considered, which we denote the *In/Out* and *T-cell* problems. Both are incompressible flow problems having boundary conditions that include an inflow shape function containing a multiplicative parameter α that controls the strength of the inflow.

3.1. In/Out problem.

The governing equations of two-dimensional incompressible viscous flow in a square region $\Omega = (0,1) \times (0,1)$ are the Navier–Stokes system

$$\frac{\partial \mathbf{u}}{\partial t} - \nu\,\Delta \mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \mathbf{0} \quad \text{in} \quad (0,T)\times\Omega, \tag{3.1}$$

$$\nabla\cdot\mathbf{u} = 0 \quad \text{in} \quad (0,T)\times\Omega \tag{3.2}$$

along with, for the specific problem we considered here, the initial and boundary conditions

$$\mathbf{u}(0,\mathbf{x}) = \mathbf{u}_0(\mathbf{x}) \quad \text{in} \quad \Omega, \tag{3.3}$$

$$\mathbf{u} = 100\,\alpha(t)\,y\,(0.2-y) \quad \text{on} \quad (0,T)\times\Gamma_i, \tag{3.4}$$

$$(\mathbf{n}\cdot\nabla)\,\mathbf{u} = 0 \quad \text{on} \quad (0,T)\times\Gamma_o, \tag{3.5}$$

$$\mathbf{u} = 0 \quad \text{on} \quad (0,T)\times\Gamma_d, \tag{3.6}$$

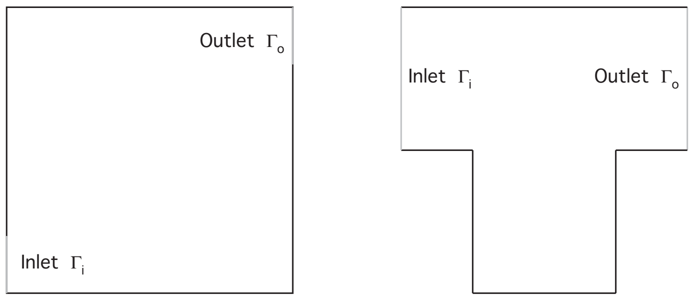

FIG. 3.1. *The flow regions for the In/Out (left) and T-cell (right) examples.*

where $\mathbf{x} = (x, y)$, $\Gamma_i = \{x = 0\,;\, 0 < y < 0.2\}$, and $\Gamma_o = \{x = 1\,;\, 0.8 < y < 1\}$ are the inflow and outflow parts of the boundary, respectively, and $\Gamma_d = \partial\Omega\backslash(\overline{\Gamma}_i \cup \overline{\Gamma}_o)$. See Figure 3.1. In (3.4), $\alpha(t)$ is a parameter that determines the strength of the parabolic inflow velocity profile. We choose $\nu = 1$ and $T = 0.08$.

Accurate Galerkin-mixed method finite element approximations of the solutions of (3.1)–(3.6) are obtained using the $P2$–$P1$ Taylor–Hood finite element pair on a grid of 1681 nodes. Finite element solutions are used for the generation of snapshots and later for comparison with CVT-based reduced-order solutions.

For the generation of snapshots, we will also solve, by the finite element method, stationary versions of (3.1)–(3.6) for which the time derivative term in (3.1) and the initial condition (3.3) are omitted and α in (3.4) is chosen independent of t.

3.1.1. Generating snapshots. We use the procedure given in section 1.1.1 to determine a set of snapshot vectors. First, the finite element approximation to the stationary version of (3.1)–(3.6) with $\alpha = 1$ is obtained. Using that steady-state solution as the initial data $\mathbf{u}_0$ in (3.3) and using $\alpha = 5$ for $0 < t < T/2 = 0.04$ and $\alpha = 1$ for $0.04 < t < T = 0.08$ in the boundary condition (3.4), we then determine a finite element approximation $\sum_{k=1}^{\widetilde{j}} U_k(t)\phi_k(\mathbf{x})$ of the solution of (3.1)–(3.6), where $\widetilde{j}$ denotes the dimension of the finite element space used for the velocity and ϕ_x denotes the basis functions for that space. This is the flow we use to generate the snapshots; it can be viewed as one for which the steady-state solution for $\alpha = 1$ is suddenly jolted, at $t = 0$, by increasing the value of α to five, and jolted again at $t = T/2 = 0.04$ by decreasing the value of α back to one. The 800 snapshot vectors

$$\vec{U}_m = \begin{pmatrix} U_1(t_m) \\ U_2(t_m) \\ \vdots \\ U_{\widetilde{j}}(t_m) \end{pmatrix}, \quad m = 1, \ldots, 800$$

are then determined by evaluating the solution of this impulsively started problem at 800 equally spaced time values t_m, $m = 1, \ldots, 800$, ranging from $t = 0$ to $t = T = 0.08$.

For subsequent use, it is convenient to modify the 800 snapshots so that they satisfy homogeneous boundary conditions. To this end, we first obtain the reference finite element approximation $\mathbf{v}(\mathbf{x}) = \sum_{k=1}^{\widetilde{j}} V_k\phi_k(\mathbf{x})$ of the stationary version of (3.1)–(3.6) with $\alpha = 3$. We then modify the first 400 snapshots by

$$\vec{U}_m \leftarrow \left(\vec{U}_m - \frac{5}{3}\vec{V}\right) \quad \text{for } m = 1, \ldots, 400$$

and, in the same way, the second 250 snapshots are modified

$$\vec{U}_m \leftarrow \left(\vec{U}_m - \frac{1}{3}\vec{V}\right) \quad \text{for } m = 401, \dots, 800,$$

where

$$\vec{V} = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_{\widetilde{j}} \end{pmatrix}.$$

In this way, all the snapshots satisfy homogeneous boundary conditions.

The snapshots we have generated by the above processes we will refer to as the *nonnormalized snapshots.* This is to contrast them with a second set of snapshots we will use, referred to as the *normalized snapshots* which are obtained from the nonnormalized snapshots by dividing by their respective length, i.e., the normalized snapshots are given by

$$\vec{U}_m \leftarrow \frac{\vec{U}_m}{|\vec{U}_m|}.$$

We will also used a *reduced snapshot set* that is determined from the nonnormalized snapshots by deleting every other snapshot from the latter, i.e., the 250 reduced snapshots are given by

$$\vec{U}_m \quad \text{for } m = 2, 4, 6, \dots, 800.$$

3.1.2. Generating CVT reduced basis. We next apply the algorithms of section 2.2 to determine the generators of a CVT of each of the snapshot sets; a set of generators is to be used as a reduced basis. CVTs having $\widetilde{d} = 4$, 8, and 16 generators are determined. Figure 3.2 displays, for the 4-generator case, the CVT basis computed from the normalized snapshots. Note that since the original boundary conditions have been "subtracted away," each basis function satisfies a zero Dirichlet boundary condition at the inlet, and a zero Neumann condition at the outlet. In the interior of the region, each basis function satisfies (discretized) continuity equation.

Figure 3.3 displays the system of 8 CVT basis functions computed from the same data. It is important to note that the set of size 8 is not built by augmenting the set of size 4; most of the elements of the larger set seem significantly different from any of those of the smaller set.

Each generator $\vec{Z}_i \in \mathbb{R}^{\widetilde{j}}$ of a CVT defines a finite element function, i.e., if

$$\vec{Z}_i = \begin{pmatrix} Z^i_1 \\ Z^i_2 \\ \vdots \\ Z^i_{\widetilde{j}} \end{pmatrix} \quad \text{for } i = 1, \dots, \widetilde{d},$$

we then have the corresponding finite element functions

$$\mathbf{z}_i = \sum_{k=1}^{\widetilde{j}} Z^i_1 \phi_j(\mathbf{x}) \quad \text{for } i = 1, \dots, \widetilde{d}.$$

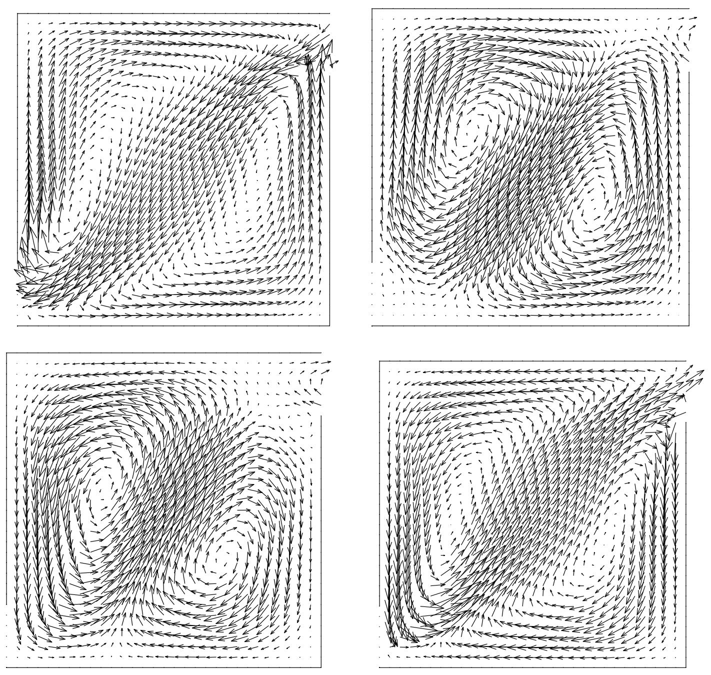
FIG. 3.2. *The CVT basis of dimension* 4 *determined from normalized snapshots.*

3.1.3. Determining CVT-based reduced-order approximations. We now use the Galerkin procedure discussed in section 1.1.3 to determine a CVT-based reduced-order approximation of the velocity field. The approximate velocity field $\mathbf{u}(\mathbf{x}, t)$ is represented as a linear combination of the CVT basis functions as follows:

$$\mathbf{u}_{cvt} = \beta(t)\mathbf{v} + \sum_{i=1}^{\widetilde{d}} a_i(t)\mathbf{z}_i(\mathbf{x}), \tag{3.7}$$

where $\mathbf{z}_i$ denotes the ith CVT basis function, $a_i(t)$ is the corresponding coefficient, $\mathbf{v}$ is the reference solution defined in section 3.1.1, $\beta(t) \equiv \alpha(t)/\alpha_r$, $\alpha_r = 3$, and $\widetilde{d}$ is the total number of CVT basis functions. In our computations, $\widetilde{d} = 4$, 8, or 16. The first term in (3.7) is included so that $\mathbf{u}$ satisfies the boundary conditions in (3.1)–(3.6).

Applying the Galerkin principle which forces the residual to be orthogonal to each of the basis functions, we obtain

$$\begin{aligned} \int_\Omega \frac{\partial}{\partial t}\mathbf{u}_{cvt} \cdot \mathbf{z}_j \, d\Omega + \nu \int_\Omega \nabla \mathbf{u}_{cvt} : \nabla \mathbf{z}_j \, d\Omega \\ + \int_\Omega (\mathbf{u}_{cvt} \cdot \nabla)\mathbf{u}_{cvt} \cdot \mathbf{z}_j \, d\Omega = 0 \qquad \text{for } j = 1, \dots, \widetilde{d}. \end{aligned} \tag{3.8}$$

Note that due to the fact that the CVT basis functions $\mathbf{z}_j$, $j = 1, \dots, \widetilde{d}$, are discretely solenoidal, the pressure does not appear in this system. Using (3.7), it is easy to see

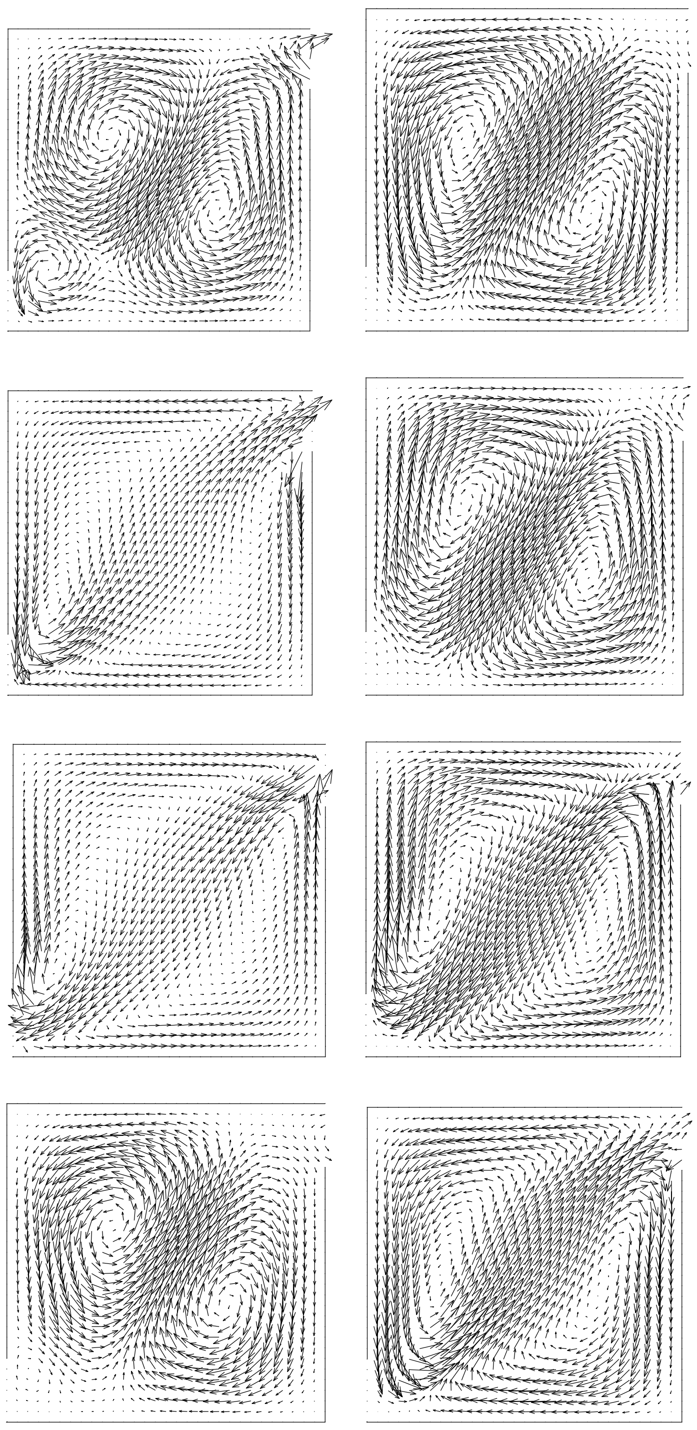

FIG. 3.3. *The CVT basis of dimension 8 function determined from normalized snapshots.*

that (3.8) is equivalent to the system of nonlinear ordinary differential equations

$$
\begin{aligned}
&\sum_{i=1}^{\widetilde{d}} \frac{d}{dt} a_i(t)\Big(\mathbf{z}_i, \mathbf{z}_j\Big) + \nu \sum_{i=1}^{\widetilde{d}} a_i(t)\Big(\nabla\mathbf{z}_i, \nabla\mathbf{z}_j\Big) \\
&+ \left(\sum_{i=1}^{\widetilde{d}} a_i(t)\mathbf{z}_i \cdot \nabla \sum_{k=1}^{\widetilde{d}} a_k(t)\mathbf{z}_k, \mathbf{z}_j\right) + \beta(t)\sum_{i=1}^{\widetilde{d}} a_i(t)\Big(\mathbf{z}_i\cdot\nabla\mathbf{v} + \mathbf{v}\cdot\nabla\mathbf{z}_i, \mathbf{z}_j\Big) \\
&+ \frac{d}{dt}\beta(t)\Big(\mathbf{v}, \mathbf{z}_j\Big) + \beta(t)\Big(\beta(t) - 1\Big)\Big(\mathbf{v}\cdot\nabla\mathbf{v}, \mathbf{z}_j\Big) = 0, \quad j = 1, \ldots, \widetilde{d},
\end{aligned}
\tag{3.9}
$$

along with the initial conditions

$$
\sum_{i=1}^{\widetilde{d}} a_i(0)\big(\mathbf{z}_i, \mathbf{z}_j\big) = \big(\mathbf{u}_0 - \beta(0)\mathbf{v}, \mathbf{z}_j\big),
\tag{3.10}
$$

where $(\cdot,\cdot)$ denotes the $L^2(\Omega)$ inner product. The set of ordinary differential equations (3.9)–(3.10) is solved by using a 4th order Runge–Kutta method.

To demonstrate robustness, the low-dimensional, CVT-based dynamic model was solved using several shapes for the inlet velocity factor $\alpha(t)$ and several different choices for the CVT basis set. The two choices for the inlet velocity function for which we present results are given by:

- *Case* 1: a hat function with respect to time (see Figure 3.4);
- *Case* 2: a sinusoid with respect to time (see Figure 3.5).

Nine different CVT bases were used: for $\widetilde{d} = 4$, 8, and 16,

- a set of $\widetilde{d}$ CVT basis functions derived from the nonnormalized snapshot set;
- a set of $\widetilde{d}$ CVT basis functions derived from the normalized snapshot set;
- a set of $\widetilde{d}$ CVT basis functions derived from the reduced snapshot set.

In addition, for both Cases 1 and 2, solutions of the system (3.1)–(3.6) were also approximated using the full finite element discretization employing thousands of unknowns.

Computing the CVT-based reduced-order solution for basis function sets of size $\widetilde{d} = 4$, 8, and 16 allowed for some study of the approximation error behavior. The CVT-based solutions are compared to the corresponding solutions of the full finite element system. The relative L^2 error was computed at each time step, based on the difference between the full finite element and CVT-based reduced-order solutions. The plots of these L^2 errors versus time are displayed in Figures 3.4 and 3.5 for Cases 1 and 2, respectively. Note that, for each case and for every time step, the overall L^2 error decreases as the size of the CVT basis set is increased.

Further comparisons are provided in Figures 3.6 and 3.7 where we display, side by side, the full finite element and CVT-based reduced-order solutions using 8 basis functions at selected time steps. Visually, the two solutions seem indistinguishable; however, the CVT-based solution was constructed from a system of 8 ordinary differential equations, while the corresponding finite element solution is derived by solving 1681 ordinary differential equations. The total running time was reduced by a factor of several thousand.

3.2. The T-cell problem. The CVT-based reduced-order modeling methodology was also applied to a second example of flow in a t-shaped region; see Figure 3.1. The flow enters from the left and exits on the right. We choose $\nu = 1$ and $T = 0.05$. A

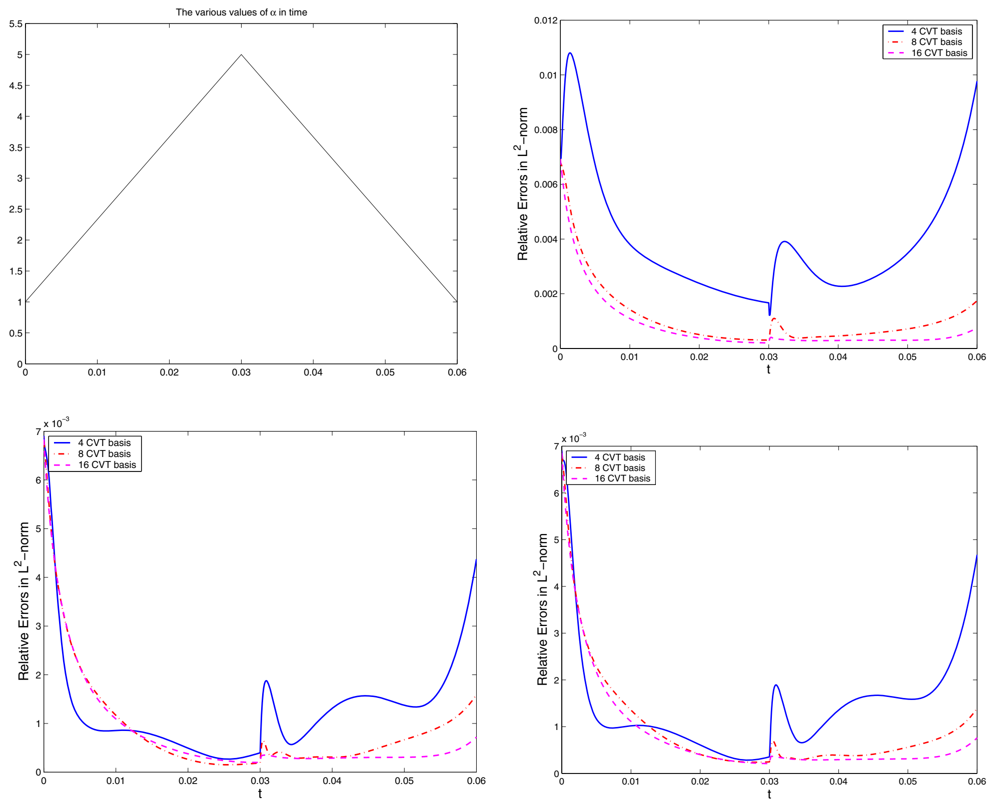

FIG. 3.4. *Multiplicative inflow strength parameter α for Case* 1 *(top left) and L^2-norm of difference between CVT-based reduced-order solutions and a full finite element simulation vs. time for the In/Out problem based on the normalized snapshot set (top right), the nonnormalized snapshot set (bottom left), and the reduced snapshot set (bottom right).*

grid of 4961 nodes was used for all finite element computations. Again, a multiplicative parameter α controls the strength of the inflow. For $0 < t < 0.025$, $\alpha = 5$ was used which represents an abrupt change at $t = 0$ from the value 1 used to generate the steady-state initial condition. For $0.025 < t < 0.05$, $\alpha = 1$ was used which represents another abrupt change at $t = 0.025$. The abrupt changes in the value of α cause sharp transient effects. This single finite element solution of (3.1)–(3.6) was sampled at 500 equally spaced time intervals to generate 500 snapshot vectors.

As for the In/Out problem, computations were performed for CVT bases of size 4 (displayed in Figure 3.8), 8 (displayed in Figure 3.9), and 16. Also, nonnormalized, normalized, and reduced snapshot sets were considered. The nine basis sets were then used to approximate the solution of a time-dependent flow in the T-shaped region. The multiplicative flow strength factor $\alpha(t)$ was chosen to be

$$\alpha(t) = \begin{cases} 160t + 1 & (0 \le t \le 0.025) \\ -160(t - 0.05) + 1 & (0.025 \le t \le 0.05). \end{cases}$$

A finite element solution of the same problem (using 4961 nodes) was also computed for comparison purposes. In Figure 3.10, the time histories of the differences between the CVT-based reduced-order solutions and the full finite element solutions, measured in the $L^2(\Omega)$-norm, are provided for the nine basis sets.

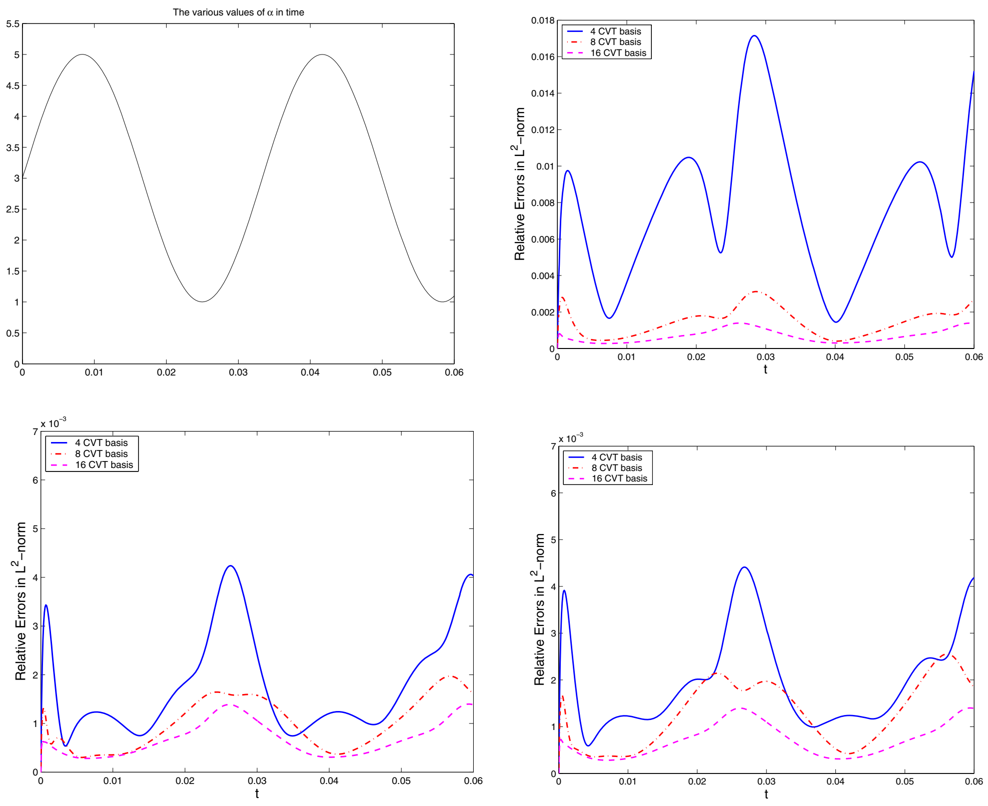

FIG. 3.5. *Multiplicative inflow strength parameter α for Case 2 (top left) and L^2-norm of difference between CVT-based reduced-order solutions and a full finite element simulation vs. time for the In/Out problem based on the normalized snapshot set (top right), the nonnormalized snapshot set (bottom left), and the reduced snapshot set (bottom right).*

3.3. Some information about CVT bases. It is of interest to mention some details about the CVT basis generation process; this discussion will consider the case of 8 CVT generators, based on nonnormalized snapshots, for the T-cell problem.

The basis generation algorithm preprocessed the data by subtracting an appropriate multiple of a steady-state solution that satisfies the boundary conditions and which had been computed with the parameter value $\alpha = 1$. The overall computation was repeated with 15 random initial generator configurations. The inner h-means and k-means algorithms were limited to a maximum of 30 iterations but always concluded before that limit was reached. The initial cluster energy of the random configurations was generally about 50,000; the final cluster energy of the 15 cases varied from 2,879.5 to 3,212.5.

For a “typical” random initial configuration, Table 3.1 shows how the cluster energy drops. Of course, the huge energy at the initial random configuration is slightly misleading. It drops tremendously after the first h-means step, as each data point is assigned to its nearest cluster center.

There was an interesting feature observed about the final cluster generators. For each cluster, the cluster elements tended to comprise all the solutions in a single time interval. In particular, for the case discussed here, the 8 cluster generators had components as listed in Table 3.2. In the interest of clarity, we’ve renumbered the cluster indices.

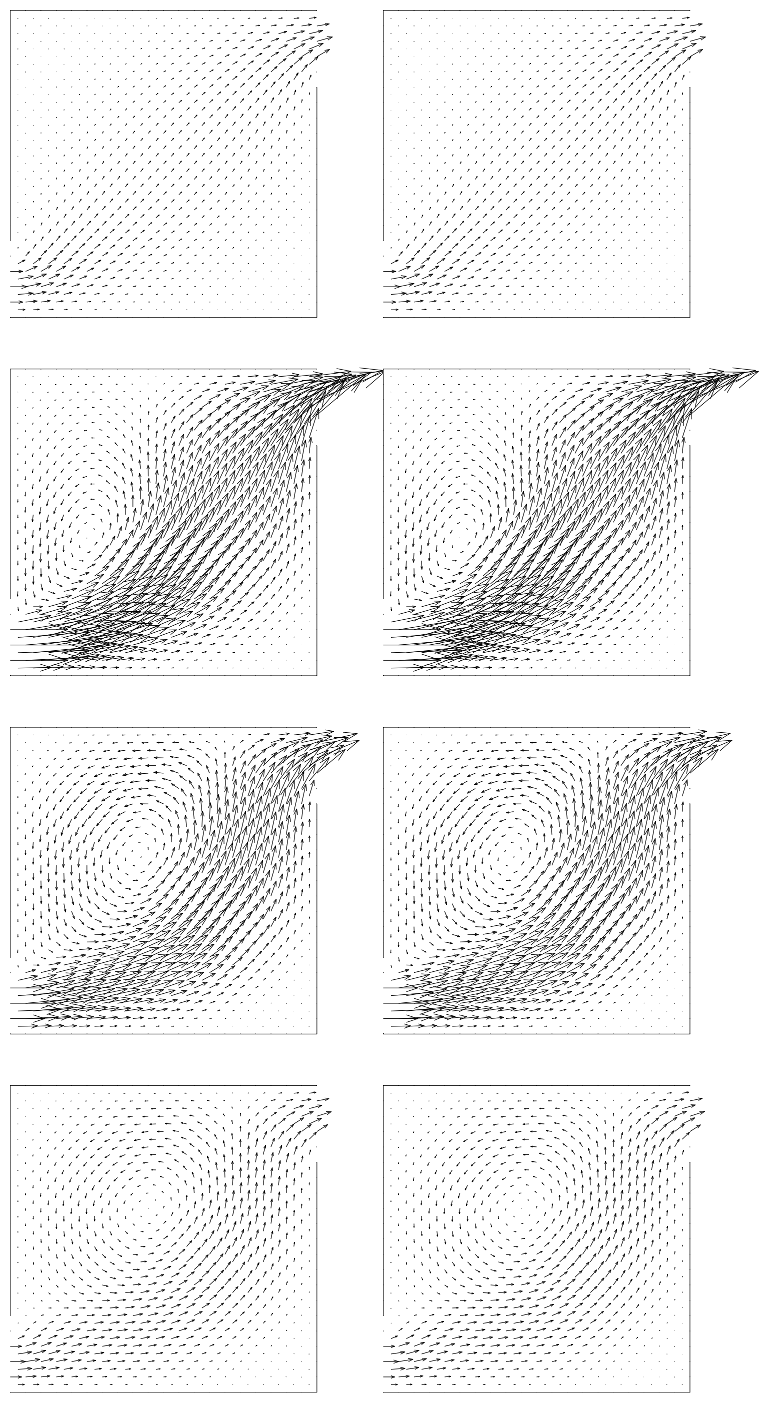

FIG. 3.6. *Full finite element approximation (left) and CVT-based reduced-order approximation of size 8 based on nonnormalized snapshots (right) for* $t = 0.1$ *(top),* $t = 1.0$ *(second),* $t = 1.5$ *(third),* $t = 2.0$ *(bottom).*

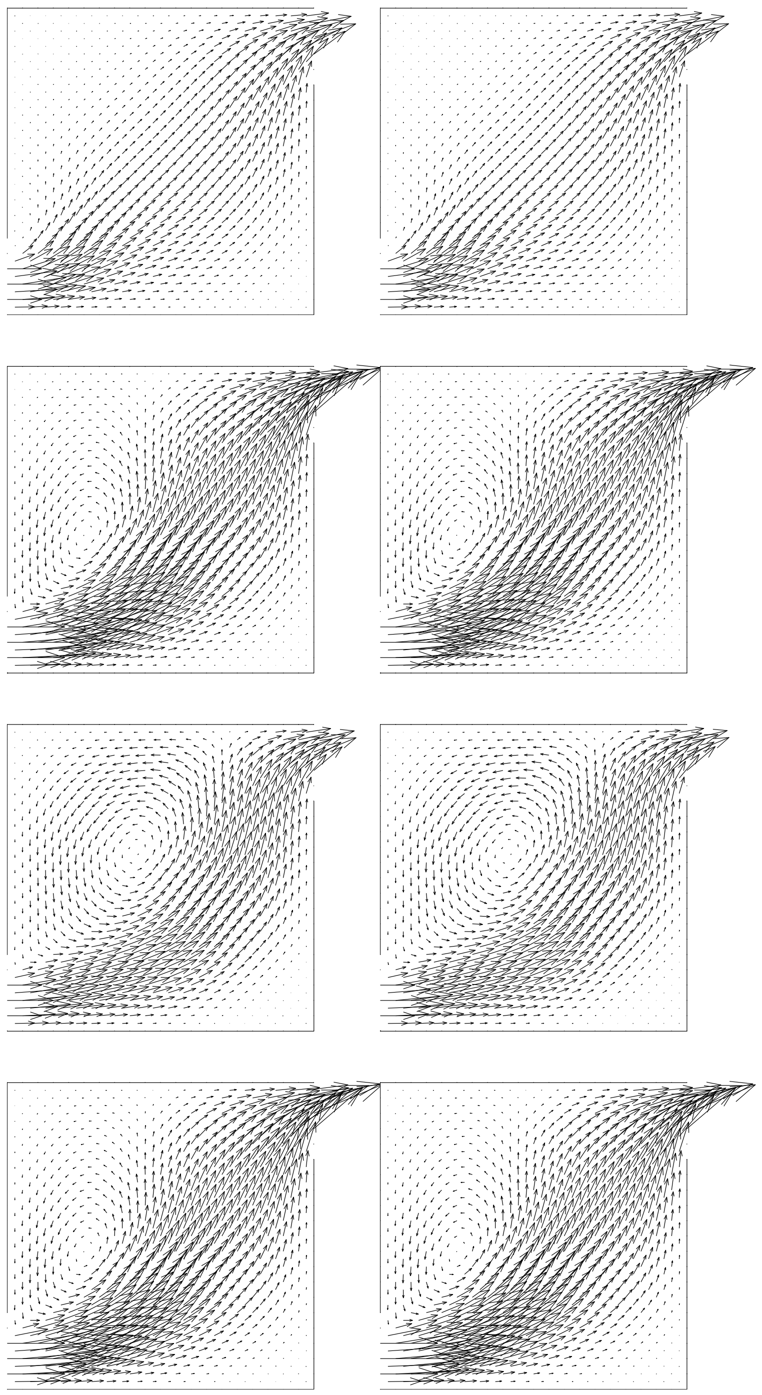

FIG. 3.7. *Full finite element approximation (left) and CVT-based reduced-order approximation of size 8 based on nonnormalized snapshots (right) for* $t = 2.5$ *(top),* $t = 3.0$ *(second),* $t = 3.5$ *(third),* $t = 5.0$ *(bottom).*

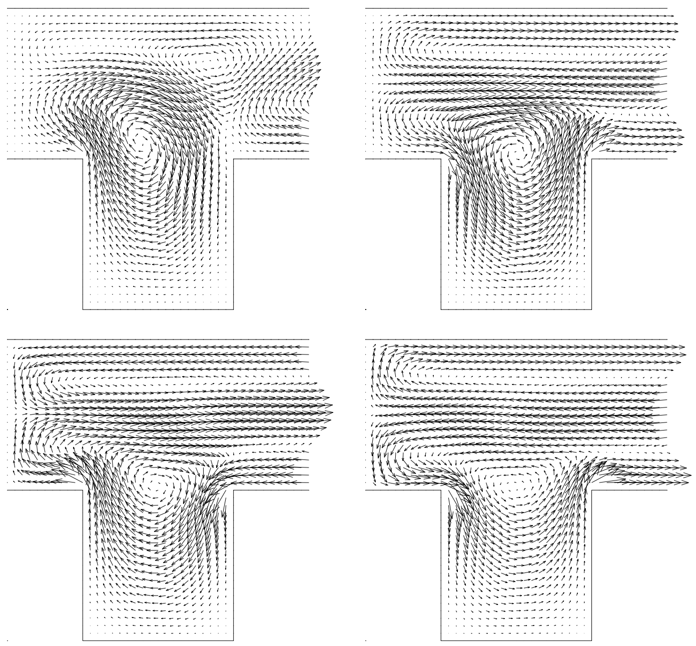

FIG. 3.8. *The CVT basis of dimension 4 determined from nonnormalized snapshots.*

Note that, except for cluster 5, the clusters are formed exactly from a sequence of data points at neighboring times. The two smallest clusters, number 1 and number 6, correspond to the initial transient and the "shock" to the system that occurs after step 250, when the system parameter α is changed. Since the solution is changing rapidly at those times, it makes sense that the clusters are small, and yet still comprise a sizable portion of the total cluster energy. Also, the one exceptional cluster, number 5, corresponds to the relatively quiescent "tails" of the two evolution processes, since it contains the solutions for steps [75, 250] and [308, 500]. This near "sequential" arrangement of snapshots within the clusters may be useful in some settings, e.g., for adapting a CVT basis.

4. Concluding remarks.

We have introduced and discussed a centroidal Voronoi tessellation based reduced-order modeling methodology and have shown that it is effective, i.e., accurate and inexpensive, on some example problems. The CVT-based approach is based on data clustering concepts. Compared to POD-based reduced-order modeling approaches, which are the ones in most common use today, the CVT-based approach is potentially less expensive, i.e., for the same number of snapshots and the same number of basis vectors, the CVT-based approach is usually less costly than POD-based approaches which require the solution of a dense eigenvalue of singular value decomposition problem. Although no direct comparisons of POD- and CVT-based results were made, the computational results provided in this paper indicate that the latter approach provides accurate results for the same type of problems for which the former approach works.

There are several avenues for the further development of CVT-based reduced-order modeling methodologies which we are currently considering. First, in [9], a discussion was given of how the CVT- and POD-based reduced-order modeling approaches can be combined to produce a hybrid method (called CVOD in that paper)

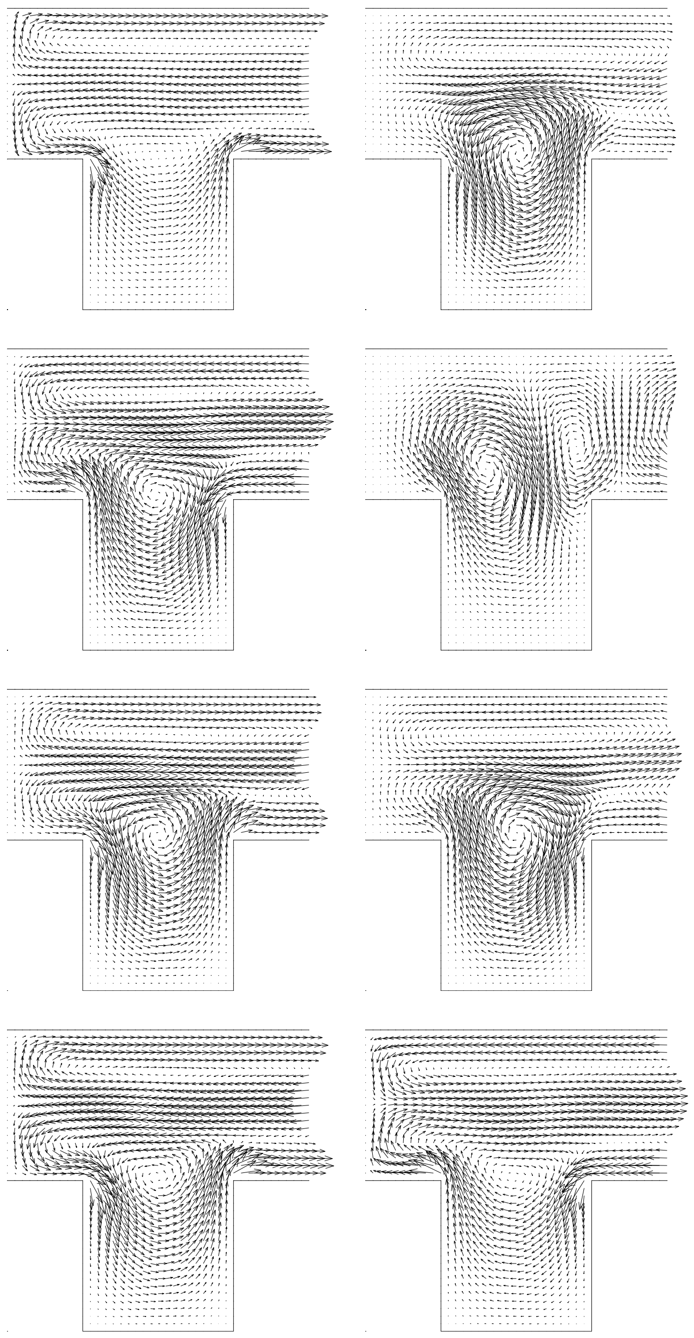

Fig. 3.9. *The CVT basis of dimension 8 determined from nonnormalized snapshots.*

that possesses the good features of both. The implementation of CVOD and experimentation with the various hybridization possibilities is a subject that is certainly deserving of attention.

Another important avenue we are pursuing is the application of CVT- and CVOD-based reduced-order modeling methodologies to control and optimization problems for

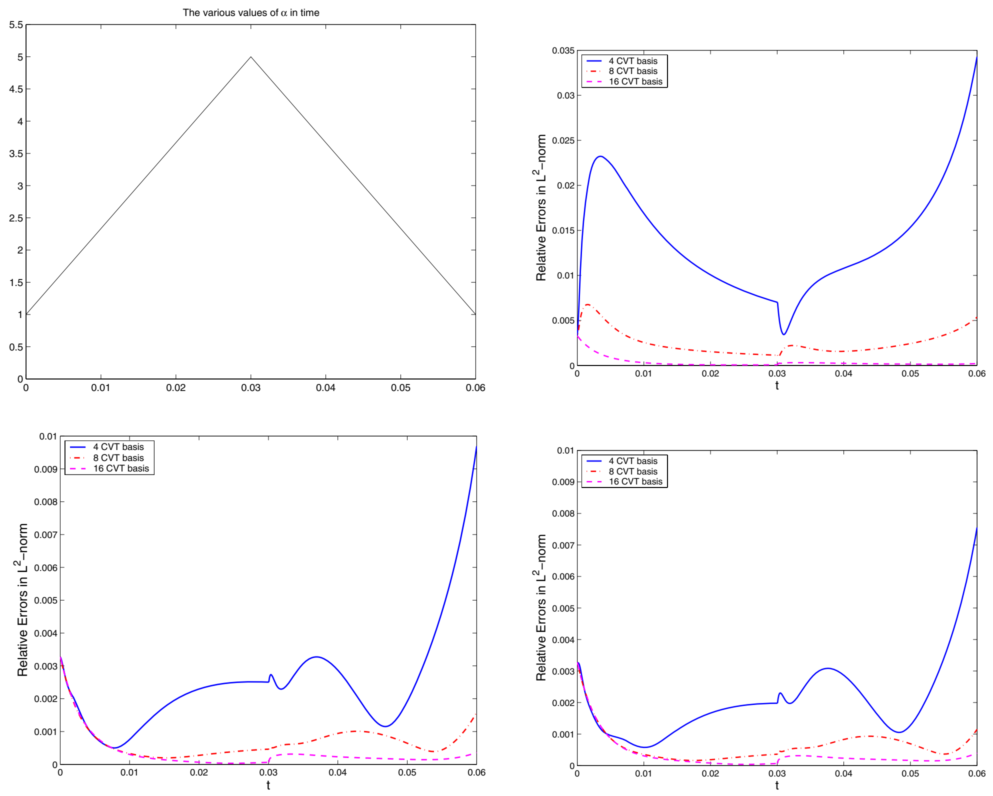

FIG. 3.10. *Multiplicative inflow strength parameter α (top left) and L^2-norm of difference between CVT-based reduced-order solutions and a full finite element simulation vs. time for the T-cell problem based on the normalized snapshot set (top right), the nonnormalized snapshot set (bottom left), and the reduced snapshot set (bottom right).*

TABLE 3.1
Energy and iteration statistics for the CVT-basis generation.

Process	Final Energy	Iterations
Random initial	49,860.5	1
h-means	3,283.4	22
k-means	3,212.5	4

TABLE 3.2
Cluster statistics for computed CVT generators.

Cluster	Cluster Energy	Population	Range
1	281.1	5	[1, 5]
2	247.5	10	[6, 15]
3	236.4	18	[16, 33]
4	215.3	41	[34, 74]
5	532.1	369	[75, 250] + [308, 500]
6	523.3	7	[251, 257]
7	428.3	15	[258, 272]
8	420.1	35	[273, 307]
Total	3,212.5	500	[1, 500]

complex systems. Many approaches to such problems require multiple or real-time state simulations so that reducing the cost of those simulations is a necessity. Indeed, much of the current interest in the reduced-order modeling of complex systems is motivated by control and optimization problems. One issue that is sure to enter these considerations is the adaptation of reduced-order bases. Adaptation is likely to be needed since, during a control or optimization process, the snapshot set that was used to generate the reduced basis my not remain adequate to represent the dynamic behavior of the system. The "sequential" aspect of CVT clustering of the snapshots may turn out to be a useful feature in this context.

A third avenue is to use the flexibility provided by CVT methodologies to advantage in the reduced-order modeling setting. For example, it is often the case that one wants a certain snapshot or a group of snapshots to have greater influence on the reduced basis than do the others. This can be easily accomplished by simply scaling the important snapshots, or equivalently, by introducing a density function defined over the snapshot set; important snapshots would have high associated densities. CVT construction algorithms would then automatically prejudice some of the generators towards the important snapshots. Furthermore, constrained CVTs can be applied which enable one to include important snapshot vectors in the reduced basis.

REFERENCES

[1] N. Aubry, W. Lian, and E. Titi, *Preserving symmetries in the proper orthogonal decomposition*, SIAM J. Sci. Comput., 14 (1993) pp. 483–505.

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

[3] G. Berkooz and E. Titi, *Galerkin projections and the proper orthogonal decomposition for equivariant equations*, Phys. Lett. A, 174 (1993), pp. 94–102.

[4] E. Christensen, M. Brons, and J. Sorensen, *Evaluation of proper orthogonal decomposition-based decomposition techniques applied to parameter-dependent nonturbulent flows*, SIAM J. Sci. Comput., 21 (2000), pp. 1419–1434.

[5] A. Deane, I. Kevrekidis, G. Karniadakis, and S. Orszag, *Low-dimensional models for complex geometry flows: Applications to grooved channels and circular cylinders*, Phys. Fluids A, 3 (1991), pp. 2337–2354.

[6] Q. Du, V. Faber, and M. Gunzburger, *Centroidal Voronoi tessellations: Applications and algorithms*, SIAM Rev., 41 (1999), pp. 637–676.

[7] Q. Du and M. Gunzburger, *Model reduction by proper orthogonal decomposition coupled with centroidal Voronoi tessellation*, in Proceedings of the Fluids Engineering Division Summer Meeting, FEDSM2002-31051, ASME, 2002.

[8] Q. Du and M. Gunzburger, *Grid generation and optimization based on centroidal Voronoi tessellations*, Appl. Math. Comput., 133 (2002), pp 591–607.

[9] Q. Du and M. Gunzburger, *Centroidal Voronoi tessellation based proper orthogonal decomposition analysis*, in Proceedings of the 8th Conference on Control of Distributed Parameter Systems, Birkhauser, Basel, Switzerland, 2002, pp. 137–150.

[10] Q. Du, M. Gunzburger, and L. Ju, *Meshfree, probabilistic determination of points, sets and support regions for meshless computing*, Comput. Methods Appl. Mech. Engrg., 191 (2002), pp. 1349–1366.

[11] Q. Du, M. Gunzburger, and L. Ju, *Constrained centroidal Voronoi tessellations for surfaces*, SIAM J. Sci. Comput., 24 (2003), pp. 1488–1506.

[12] Q. Du, M. Gunzburger, and L. Ju, *Voronoi-based finite volume methods, optimal Voronoi meshes, and PDEs on the sphere*, Comput. Methods Appl. Mech. Engrg., 192 (2003), pp. 3933–3957.

[13] A. Faulds, *Centroidal Voronoi Decompositions, Algorithms and Applications*, M.S. thesis, Department of Mathematics, Penn State University, State College, PA 2002.

[14] A. Faulds and B. King, *Sensor location in feedback control of partial differential equation systems*, in Proceedings of the 2000 IEEE CCA/CACSD, IEEE, Washington, 2000, pp. 536–541.

[15] M. Graham and I. Kevrekidis, *Pattern analysis and model reduction: Some alternative approaches to the Karhunen-Lóeve decomposition*, Comput. Chem. Engrg., 20 (1996), pp. 495–506.

[16] J. Hartigan and M. Wong, *Algorithm AS 136: A K-means clustering algorithm*, Appl. Statist., 28 (1979), pp. 100–108.

[17] P. Holmes, J. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, Cambridge University Press, Cambridge, 1996.

[18] P. Holmes, J. Lumley, G. Berkooz, J. Mattingly, and R. Wittenberg, *Low-dimensional models of coherent structures in turbulence*, Phys. Rep., 287 (1997), pp. 337–384.

[19] L. Ju, Q. Du, and M. Gunzburger, *Probablistic methods for centroidal Voronoi tessellations and their parallel implementations*, J. Parallel Comput., 28 (2002), pp. 1477–1500.

[20] K. Kunisch and S. Volkwein, *Control of Burger's equation by a reduced order approach using proper orthogonal decomposition*, JOTA, 102 (1999), pp. 345–371.

[21] K. Kunisch and S. Volkwein, *Galerkin proper orthogonal decomposition methods for parabolic problems*, Spezialforschungsbereich F003 Optimierung und Kontrolle, Projektbereich Kontinuierliche Optimierung und Kontrolle, Bericht Nr. 153, Graz, 1999.

[22] J. Lumley, *Stochastic Tools in Turbulence*, Academic Press, New York, 1971.

[23] A. Noor, *Recent advances in reduction methods for nonlinear problems*, Comput. & Structures, 13 (1981), pp. 31–44.

[24] A. Okabe, B. Boots, K. Sugihara, and S. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, 2nd ed., Wiley, New York, 2000.

[25] H. Park and D. Cho, *Low dimensional modeling of flow reactors*, Int. J. Heat Mass Transf., 39 (1996), pp. 3311–3323.

[26] H. M. Park and J. S. Chung, *A sequential method of solving inverse natural convection problems*, Inverse Problems, 18 (2002), pp. 529–546.

[27] H. M. Park and Y. D. Jang, *Control of Burger's equation by means of mode reduction*, Internat. J. Engrg. Sci., 38 (2000), pp. 785–805.

[28] H. M. Park and J. H. Lee, *Solution of an inverse heat transfer problem by means of empirical reduction of modes*, Z. Angew. Math. Phys., 51 (2000), pp. 17–38.

[29] H. M. Park and M. W. Lee, *An efficient method of solving the Navier-Stokes equations for flow control*, Internat. J. Numer. Methods Engrg., 41 (1998), pp. 1133–1151.

[30] H. M. Park and W. J. Lee, *A new numerical method for the boundary optimal control problems of the heat conduction equation*, Internat. J. Numer. Methods Engrg., 53 (2002), pp. 1593–1613.

[31] J. Peterson, *The reduced basis method for incompressible flow calculations*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 777–786.

[32] M. Rathinam and L. Petzold, *A new look at proper orthogonal decomposition*, to appear.

[33] M. Rathinam and L. Petzold, *Dynamic iteration using reduced order models: A method for simulation of large scale modular systems*, to appear.

[34] S. Ravindran, *Proper orthogonal decomposition in optimal control of fluids*, Int. J. Numer. Methods Fluids, 34 (2000), pp. 425–448.

[35] S. Ravindran, *Reduced-order adaptive controllers for fluid flows using POD*, J. Sci. Comput., 15 (2000), pp. 457–478.

[36] J. Rodríguez and L. Sirovich, *Low-dimensional dynamics for the complex Ginzburg-Landau equations*, Phys. D, 43 (1990), pp. 77–86.

[37] L. Sirovich, *Turbulence and the dynamics of coherent structures*, I-III, Quart. Appl. Math., 45 (1987), pp. 561–590.

[38] N. Smaoui and D. Armbruster, *Symmetry and the Karhunen-Loève analysis*, SIAM J. Sci. Comput., 18 (1997), pp. 1526–1532.

[39] H. Späth, *Cluster Dissection and Analysis, Theory, FORTRAN Programs, Examples*, Ellis Horwood, Chichester, 1985.

[40] D. Sparks, *Algorithm AS 58: Euclidean cluster analysis*, Appl. Statist., 22 (1973), pp. 126–130.

[41] S. Volkwein, *Optimal control of a phase field model using the proper orthogonal decomposition*, ZAMM Z. Angew. Math Mech., 81 (2001), pp. 83–97.

[42] S. Volkwein, *Proper orthogonal decomposition and singular value decomposition*, Spezialforschungsbereich F003 Optimierung und Kontrolle, Projektbereich Kontinuierliche Optimierung und Kontrolle, Bericht Nr. 153, Graz, 1999.