01 Jan 2010

Balanced POD Algorithm for Robust Control Design for Linear Distributed Parameter Systems

John R. Singler
Missouri University of Science and Technology, singlerj@mst.edu

Belinda A. Batten

Recommended Citation

J. R. Singler and B. A. Batten, "Balanced POD Algorithm for Robust Control Design for Linear Distributed Parameter Systems," *Proceedings of the 2010 American Control Conference (2010: June 30 - July 2, Baltimore, MD)*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2010.
The definitive version is available at https://doi.org/10.1109/ACC.2010.5530909

Balanced POD Algorithm for Robust Control Design for Linear Distributed Parameter Systems

John R. Singler
Department of Mathematics and Statistics
Missouri University of Science and Technology
Rolla, MO 65409
Email: singlerj@mst.edu

Belinda A. Batten
School of Mechanical, Industrial, and Manufacturing Engineering
Oregon State University
Corvallis, OR 97331–6001
Email: bbatten@engr.orst.edu

***Abstract*— A mathematical model of a physical system is never perfect; therefore, robust control laws are necessary for guaranteed stabilization of the nominal model and also "nearby" systems, including hopefully the actual physical system. We consider the computation of a robust control law for large-scale finite dimensional linear systems and a class of linear distributed parameter systems. The controller is robust with respect to left coprime factor perturbations of the nominal system. We present an algorithm based on balanced proper orthogonal decomposition to compute the nonstandard features of this robust control law. Numerical results are presented for a convection diffusion partial differential equation.**

I. Introduction

Since a mathematical model of a system is not a perfect description of the system, it is desirable for a control law to not only stabilize the mathematical model but also "nearby" systems. In this paper, we consider computing a control law for a distributed parameter system that stabilizes the nominal system and also left coprime factor perturbations of the system. The robustly stabilizing control, the central controller, is a solution to this problem and was given for finite dimensional systems by Glover and McFarlane in [1]; for infinite dimensional systems with bounded finite rank inputs and outputs, the solution can be found in [2].

Computing control laws for infinite dimensional systems often starts with discretizing the system with a convergent approximation scheme. Matrix approximations of the original system operators arise, and these matrices can be very large scale for many important applications, such as those in fluids. Much recent research has focused on developing algorithms to solve large-scale matrix Lyapunov and Riccati equations associated with control design for linear systems (among other applications); see Section IV-A below for details and references. However, the robust control law from the central controller has features that cannot directly be computed using existing algorithms for large-scale matrix equations. To address these problems, we use balanced proper orthogonal decomposition (POD) methods.

Balanced POD is an algorithm introduced by Rowley [3] for approximate balanced model reduction of linear systems. The algorithm is also related to balanced model reduction algorithms proposed by Willcox and Peraire [4] and Lall, Marsden, and Glavaški [5]. Rowley's balanced POD algorithm is similar to the method of snapshots for standard POD computations [6], however it uses two datasets. For further details on the algorithm, see the above references and also [7], [8], where we extended Rowley's algorithm to an infinite dimensional setting and proved convergence. As a side note, balanced POD can also be used to provide an optimal reconstruction of two general datasets [9] in an analogous way that standard POD can reconstruct a single dataset, e.g., [10].

In Section IV-B, we develop a balanced POD algorithm to compute the central controller for a class of linear partial differential equation (PDE) systems. The main computational cost of the algorithm is computing solution snapshots of linear PDEs. These computations can be performed with existing software and one can also take advantage of existing techniques such as special discretization schemes, domain decomposition methods, adaptive mesh refinement, and parallel algorithms. Also, since the algorithm is based on simulation data rather than matrix approximations, we bypass the potential difficulty of extracting matrices from existing simulation code. The algorithm described in this paper is new and is applicable to both large-scale finite dimensional systems and a class of infinite dimensional systems. To begin, we pose a model problem.

II. The Model Problem

To test the convergence of the algorithm, we consider a one dimensional convection diffusion equation

$$w_t(t,x) = \mu w_{xx}(t,x) - \kappa w_x(t,x) + b(x)u(t), \qquad (1)$$

$$w(t,0) = 0, \quad w(t,1) = 0, \qquad (2)$$

$$w(0,x) = w_0(x). \qquad (3)$$

System measurements are taken of the form

$$y(t) = \int_0^1 c(x)w(t,x)\,dx.$$

We assume the functions $b(x)$ and $c(x)$ are square integrable.

For the balanced POD algorithm below, we require an abstract formulation of the problem. Briefly, this can be done as follows. Let X be the Hilbert space $L^2(0,1)$ of

This work was supported in part by the Air Force Office of Scientific Research under grant FA9550-07-1-0540, and also by a grant from the University of Missouri Research Board.

square integrable functions defined on the interval $(0,1)$ with standard inner product $(f,g) = \int_0^1 f(x)g(x)\,dx$ and norm $\|f\| = (f,f)^{1/2}$. Define the convection diffusion operator $A : D(A) \subset X \to X$ by

$$[Aw](x) = \mu w_{xx}(x) - \kappa w_x(x),$$

Functions in $D(A)$ are twice differentiable and satisfy the above boundary conditions. Define $B : \mathbb{R} \to X$ and $C : X \to \mathbb{R}$ by $[Bu](x) = b(x)u$ and $Cw = (w,c)$. In this way, the PDE system can be written as the infinite dimensional system

$$\dot{w}(t) = Aw(t) + Bu(t), \quad w(0) = w_0, \quad y(t) = Cw(t),$$

where the dot denotes a time derivative.

We chose this example problem so that we could easily check for convergence of the algorithm and also compare results with computations using matrix approximations of the operators A, B, and C.

III. Robust Central Controller for Left Coprime Factor Perturbations

We now consider robust feedback control design for a general infinite dimensional system

$$\dot{x}(t) = Ax(t) + Bu(t), \quad y(t) = Cx(t),$$

holding over a Hilbert space X. We assume the operator $A : D(A) \subset X \to X$ generates a C_0-semigroup, and the control input operator $B : \mathbb{R}^m \to X$ and the observation operator $C : X \to \mathbb{R}^p$ are both bounded and finite rank.

We assume (A,B,C) is exponentially stabilizable and detectable so that the transfer function $G(s) = C(sI - A)^{-1}B$ has a normalized left coprime factorization $G = \tilde{M}^{-1}\tilde{N}$; see [2, Lemma 9.4.10]. We consider the following robust control problem: Given a robustness margin $\varepsilon > 0$, find a controller $K_c(s)$ stabilizing $G(s)$ and all "nearby" plants $G_\Delta(s)$ of the form

$$G_\Delta = (\tilde{M} + \Delta_M)^{-1}(\tilde{N} + \Delta_N),$$

where the perturbation $\Delta = [\Delta_M \ \Delta_N]$ satisfies $\|\Delta\|_\infty < \varepsilon$. Here, the applied norm is the $\mathcal{H}_\infty$ norm, which is the largest singular value of the function evaluated along the imaginary axis.

The solution to this problem can be found in Section 9.4 of Curtain and Zwart's book [2]. Let the operators $\Pi : X \to X$ and $P : X \to X$ be the unique self-adjoint, nonnegative solutions to the control and filter algebraic Riccati equations (AREs)

$$A^*\Pi + \Pi A - \Pi BB^*\Pi + C^*C = 0, \tag{4}$$

$$AP + PA^* - PC^*CP + BB^* = 0, \tag{5}$$

where the asterisk (*) denotes the Hilbert adjoint operator. There is a parameterized family of controllers solving the above problem; the central controller is given in state space form by [2, Theorem 9.4.16]

$$u(t) = -Kx_c(t), \quad \dot{x}_c(t) = A_c x_c(t) + \sigma^{-2}W^*Fy(t),$$

where

$$\begin{aligned} K &= B^*\Pi, & A_c &= A - BK - \sigma^{-2}W^*FC, \\ F &= PC^*, & W^* &= \left[I + (1-\sigma^{-2})\Pi P\right]^{-1}, \\ \sigma &= \left(1-\varepsilon^2\right)^{1/2}, & & 0 < \varepsilon < \varepsilon_{\max}. \end{aligned}$$

Here, the maximum robustness margin $\varepsilon_{\max}$ can be found exactly and is given by [2, Corollary 9.4.12]:

$$\varepsilon_{\max} = \left[1 + \lambda_{\max}(P\Pi)\right]^{-1/2}, \tag{6}$$

where $\lambda_{\max}(T)$ denotes the largest eigenvalue of the operator T. This extends the finite dimensional results of Glover and McFarlane [1] to an infinite dimensional case.

IV. Balanced POD Algorithm

For finite dimensional systems, the central controller described above can be computed by using existing solvers for the algebraic Riccati equations (4) and (5). For infinite dimensional systems, a standard approach to control design is to approximate the infinite dimensional operators by matrices and design an approximate control law; see, e.g., [11], [12], [13]. However, for many infinite dimensional equations the approximating matrices are of high dimension and conventional computational approaches are difficult, if not impossible, to apply. For the central controller design above, we have the following computational challenges for large-scale systems:

1) Compute the solutions Π and P of the algebraic Riccati equations (4) and (5)
2) Compute the maximum robustness margin $\varepsilon_{\max}$ in (6).
3) Compute $W^*F = \left[I + (1-\sigma^{-2})\Pi P\right]^{-1}F$

In this paper, we focus on using a balanced POD algorithm to treat the second and third challenges. Of course, we must first address the Riccati equations.

A. Riccati Equation Solver

Before we describe the algorithm we use to solve the Riccati equations, we briefly discuss the three main existing classes of algorithms for large-scale or infinite dimensional Riccati equations:

Three Algorithm Classes for Riccati Equations

1) Apply Newton's method to the (quadratic) Riccati equation and solve the resulting (linear) Lyapunov equations using special techniques; see, e.g., [14], [15], [16].
2) Solve the related Chandrasekhar equations, a nonlinear system of differential equations that must be integrated to steady state; see, e.g., [14], [17], [18].
3) Reduce the infinite dimensional model and solve the resulting low order matrix Riccati equation; see, e.g., [19], [20], [21].

Although all three approaches can be successful, the first approach is convergent and is generally accepted to be the most accurate; the Chandrasekhar equations can lose accuracy when integrated to steady state [14], and the "reduce-then-design" approach can fail as it has no guarantees of accuracy or convergence.

In this work, we use the first approach coupled with a trapezoid snapshot Lyapunov equation algorithm developed in [22] (see also [23], [24], [25]). We use the standard Kleinman-Newton iteration [26], [27] as opposed to the modified Kleinman-Newton iteration proposed by Banks and Ito in [14]; Feitzinger, Hylla, and Sachs have recently shown in [16] that errors can accumulate in the modified iteration if the Lyapunov equations are solved inexactly (which is unavoidable except for simple problems).

B. Balanced POD Algorithm for the Central Controller

Next, we present the balanced POD algorithm to treat the final two challenges in the central controller computations. Let X be a Hilbert space with real-valued inner product $(\cdot,\cdot)$ and corresponding norm $\|x\| = (x,x)^{1/2}$. We assume the operators (A,B,C) fall in the general framework of Section III. Since B and C are finite rank, the operators must take the form

$$Bu = \sum_{j=1}^{m} u_j b_j, \quad Cx = [\,(x,c_1),\dots,(x,c_p)\,]^T,$$

for some vectors $b_1,\dots,b_m$ and $c_1,\dots,c_p$ in X (see [28, Theorem 6.1]). As with most large-scale algorithms for feedback control gain computations, the algorithm is tractable when m and p are relatively small.

For finite dimensional systems, the Hilbert space X is taken to be $\mathbb{R}^n$, and the inner product can be taken as the standard dot product, $(a,b) = a^T b$, or a weighted dot product, $(a,b) = a^T M b$, where $M \in \mathbb{R}^{n\times n}$ is symmetric positive definite. Also, the above expressions for B and C hold where b_j is the jth column of B and c_i is the ith column of C^T.

Recall that the goals are to (a) compute the maximum robustness margin $\varepsilon_{\max}$, and (b) compute W^*F. Note that (a) requires the computation of the maximum eigenvalue of the product of the Riccati solutions, while (b) requires the inversion of an operator involving the product of the Riccati solutions. We proceed as follows.

Step 1: The first step in the algorithm is to compute the gain operators $K = B^*\Pi$ and $F = PC^*$, where Π and P solve the Riccati equations (4) and (5). More specifically, we require the *functional gains* for the operators K and F. These are defined using the above representations of B and C as follows. Since B takes the form $Bu = \sum_{j=1}^{m} u_j b_j$, it can be checked that $B^*x = [\,(x,b_1),\dots,(x,b_m)\,]^T$. Therefore, for any $x \in X$,

$$\begin{aligned} Kx &= B^*\Pi x \\ &= [\,(\Pi x, b_1),\dots,(\Pi x, b_m)\,]^T \\ &= [\,(x, \Pi b_1),\dots,(x, \Pi b_m)\,]^T, \end{aligned}$$

since Π is self-adjoint. Thus, $Kx = [\,(x,k_1),\dots,(x,k_m)\,]^T$, where $k_i = \Pi b_i \in X$ are the functional gains for K. Similarly, $Fy = \sum_{j=1}^{p} y_j f_j$, where $f_i = Pc_i$ are the functional gains for F.

Step 2: The next step is to rewrite the solutions of the Riccati equations (4) and (5) in terms of solutions of linear infinite dimensional differential equations. This is done as follows. First, as is well known, the Riccati equations can be rewritten as the Lyapunov equations

$$\begin{aligned} (A-BK)^*\Pi + \Pi(A-BK) + K^*K + C^*C &= 0, \\ (A-FC)P + P(A-FC)^* + FF^* + BB^* &= 0. \end{aligned}$$

Using the above representations of the operators B, C, K, and F, it can be shown [23], [22] that the solutions of these Lyapunov equations (i.e., the Riccati operators Π and P) take the form

$$\Pi x = \int_0^\infty \sum_{i=1}^{m+p} \big(x, z_i(t)\big)\, z_i(t)\, dt,$$

$$Px = \int_0^\infty \sum_{j=1}^{m+p} \big(x, w_j(t)\big)\, w_j(t)\, dt,$$

where each $z_i(t)$ and $w_j(t)$ are the unique solutions of the linear evolution equations

$$\dot z_i(t) = (A-BK)^* z_i(t), \quad z_i(0) = z_i^0, \tag{7}$$

$$\dot w_j(t) = (A-FC) w_j(t), \quad w_j(0) = w_j^0, \tag{8}$$

with the initial conditions given by

$$\begin{aligned} z_i^0 &= k_i,\ i = 1,\dots,m, \quad z_i^0 = c_i,\ i = m+1,\dots,m+p, \\ w_j^0 &= f_j,\ j = 1,\dots,p, \quad w_j^0 = b_j,\ i = p+1,\dots,m+p. \end{aligned}$$

Step 3: Next, use balanced POD to approximate the eigenvalues and eigenvectors of the operator products ΠP and $P\Pi$. This is done in exactly the same manner as the snapshot LQG balancing algorithm proposed by the authors in Section IV B of [25]; specifically, we require the LQG balanced POD singular values $\{\mu_k\}$ (the diagonal entries of the matrix M in step 4 of the algorithm) and the LQG balanced POD modes $\{\varphi_k, \psi_k\}$. In [9], it is shown that the nonzero eigenvalues $\{\lambda_k\}$ of ΠP and $P\Pi$ are the same, they can be expressed in terms of the singular values by $\lambda_k = \mu_k^2$, and therefore they can be ordered $\lambda_1 \geq \lambda_2 \geq \cdots > 0$.

Step 4: Finally, we approximate the maximum robustness margin $\varepsilon_{\max}$ of the central controller, and the operator product W^*F. Recall the robustness margin is given by $\varepsilon_{\max} = \big[1 + \lambda_{\max}(P\Pi)\big]^{-1/2}$. The balanced POD computations give $\lambda_{\max}(P\Pi) \approx \lambda_1$; furthermore, as the approximate solutions of the differential equations (7) and (8) converge, we have proved that λ_1 converges to $\lambda_{\max}(P\Pi)$ [8].

To compute the operator product W^*F, recall that $Fy = \sum_{j=1}^{p} y_j f_j$, where each $f_j \in X$ is a functional gain for F. Thus, $W^*Fy = \sum_{j=1}^{p} y_j (W^* f_j)$, and we need only compute the products $W^* f_j$ for $j = 1,\dots,p$ to form W^*F. We compute the products $W^* f_j$ using the following result.

Theorem 1: Let $\alpha = -(1-\sigma^{-2})$, where $\sigma = (1-\varepsilon^2)^{1/2}$ and $0 < \varepsilon < \varepsilon_{\max}$. Also, let $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ be the exact eigenvalues of $P\Pi$ and ΠP with corresponding eigenvectors $\{\varphi_k\}$ and $\{\psi_k\}$, respectively. Then for any $x \in X$, $\xi = W^*x = (I - \alpha \Pi P)^{-1} x$ is given by

$$\xi = x + \alpha \sum_{k\geq 1} \frac{\lambda_k}{1-\alpha\lambda_k} (x, \varphi_k)\, \psi_k.$$

Furthermore, let ξ_r be the rth order approximation defined by

$$\xi_r = x + \alpha \sum_{k=1}^{r} \frac{\lambda_k}{1-\alpha\lambda_k} (x, \varphi_k)\, \psi_k. \tag{9}$$

Then the approximation error is bounded by

$$\|\xi - \xi_r\| \le \alpha\, \|x\| \sum_{k>r} \frac{\lambda_k}{1-\alpha\lambda_k} \|\varphi_k\|\, \|\psi_k\| < \infty. \tag{10}$$

The proof will appear in a future work.

Notes:

- By the definition of α, we have $\alpha > 0$ and therefore the coefficient of α in the error bound is positive. Also, it can be shown that $0 < 1 - \alpha\lambda_k < 1$, and therefore $\lambda_k/(1-\alpha\lambda_k)$ is never infinite. However, as $\varepsilon \to \varepsilon_{\max}$, $1 - \alpha\lambda_1$ approaches zero and therefore the first term in the expression for ξ_r becomes infinitely large.
- All of the terms in the error bound are computable. Also, if the eigenvalues $\{\lambda_k\}$ decay quickly enough, a small value of r will give a good approximation.
- We proved convergence of approximations in [8]: as the approximate solutions of the differential equations (7) and (8) converge, each computed λ_k converges and, if λ_k is distinct, then the computed φ_k and ψ_k also converge.

The balanced POD algorithm for the central control computations may be summarized as follows:

Balanced POD Algorithm Summary:

1) Approximate the feedback gains $K = B^*\Pi$ and $F = PC^*$, where Π and P solve the AREs (4) and (5). In this work, we use the trapezoid snapshot Lyapunov algorithm from [22].
2) Compute approximate solutions of the differential equations (7) and (8).
3) Use balanced POD to approximate the eigenvalues $\{\lambda_k\}$ of $P\Pi$ and ΠP with corresponding eigenvectors $\{\varphi_k\}$ and $\{\psi_k\}$, respectively.
4) The quantity $\lambda_{\max}(P\Pi)$ is approximated by λ_1, and the operator product W^*F is approximated using Theorem 1.

V. Numerical Results

We test the convergence of the algorithm with the model problem outlined above with $\mu = 0.05$, $\kappa = 1$, control input function $b(x) = 5\sin(\pi x)$, and observation function $c(x) = -6x(x-1)\sin(\pi x/2)$.

For the snapshot algorithms, we used standard piecewise linear finite elements for the spatial discretization. For the functional gain computations, we used zero as the initial guess in the Kleinman-Newton iterations. We used a constant time step of $\Delta t = 0.01$ in the trapezoid Lyapunov solver. To approximate the solutions of the differential equations (7) and (8) in Step 2 of the algorithm, we used Matlab's `ode23s` adaptive solver. We comment further on this below.

Figures 1 and 2 show approximations to the functional gains $k(x)$ and $f(x)$ computed using 64 equally spaced finite element nodes. Further refinement produced little change in the approximations.

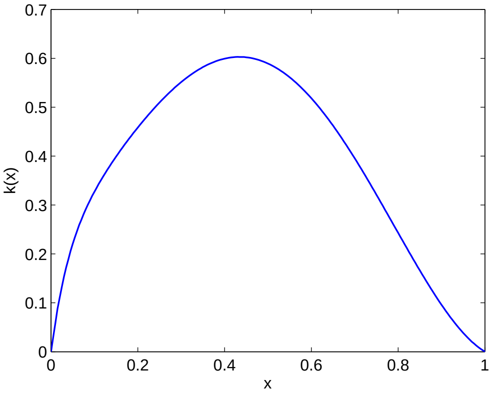

Fig. 1. Approximate control functional gain $k(x)$ computed using the snapshot algorithm with $\Delta t = 0.01$ and 64 equally spaced finite element nodes.

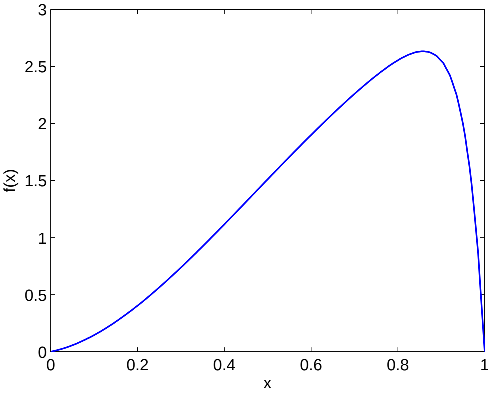

Fig. 2. Approximate observation functional gain $f(x)$ computed using the snapshot algorithm with $\Delta t = 0.01$ and 64 equally spaced finite element nodes.

The balanced POD eigenvalues are shown in Figure 3 for various numbers of equally spaced finite element nodes. The balanced POD eigenvalues decay very rapidly to zero, and they converge as the mesh is refined. The larger eigenvalues converge much more rapidly than the smaller eigenvalues. Since the eigenvalues decay to zero rapidly, the error bound (10) of Theorem 1 leads us to expect that a small value of r for ξ_r will provide a good approximation to ξ in the computations of W^*F.

One of our main goals in the central controller computation is the approximation of the maximum robustness margin $\varepsilon_{\max} = \left[1 + \lambda_{\max}(P\Pi)\right]^{-1/2}$. As discussed above, we know λ_1 converges to $\lambda_{\max}(P\Pi)$ as the approximate solutions of the differential equations (7) and (8) converge. Therefore, we approximate $\varepsilon_{\max}$ by $(1+\lambda_1)^{-1/2}$. As shown above, λ_1 converged very quickly as the mesh is refined. Table I shows the fast convergence of the approximated maximum robustness margin to 0.9188 as the finite element mesh is refined.

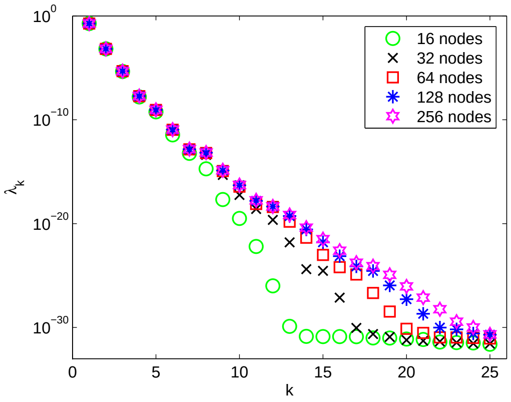

Fig. 3. Approximate balanced POD eigenvalues λ_k computed using various numbers of equally spaced finite element nodes.

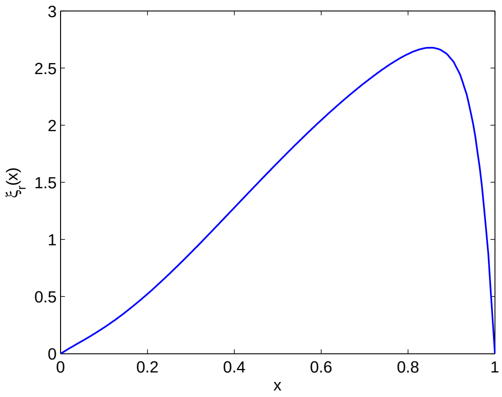

Fig. 4. The function $\xi_r(x)$ for $r = 2$ and $\varepsilon = (0.5)\varepsilon_{\max}$ as computed using 64 finite element nodes.

TABLE I

THE MAXIMUM ROBUSTNESS MARGIN $\varepsilon_{\max}$ APPROXIMATED BY $(1 + \lambda_1)^{-1/2}$ USING VARIOUS NUMBERS OF EQUALLY SPACED FINITE ELEMENT NODES.

nodes	16	32	64	128	256
$\varepsilon_{\max}$	0.9180	0.9186	0.9188	0.9188	0.9188

Next, we consider our second goal in the central controller computations: the approximation of $W^*F = (I-\alpha\Pi P)^{-1}F$, where $\alpha = -(1-\sigma^{-2})$, $\sigma = (1-\varepsilon^2)^{1/2}$, and $0 < \varepsilon < \varepsilon_{\max}$. For our model problem, $Fy = fy$, where $f \in X$ is the functional gain approximated above. As discussed above, to compute W^*F, we need only compute W^*f since $W^*Fy = (W^*f)y$. We use Theorem 1 to approximate $\xi = W^*f$. The first step in the computation is to choose a value for r so that the rth order approximation ξ_r defined in (9) is close to $\xi = W^*f$. We chose the value of r so that the value of the error bound for $\|\xi - \xi_r\|$ given in (10) is less than 10^{-4}. For all computations, we used 20 terms in the series in the error bound (10). This gave $r = 2$ for finite element meshes with 16, 32, 64, 128, and 256 equally spaced nodes and all values of ε considered.

Figure 4 shows the function $\xi_r(x)$ with $r = 2$ and $\varepsilon = (0.5)\varepsilon_{\max}$ computed using 64 equally spaced finite element nodes. For this value of ε, the function $\xi_r(x)$ is very close to the observer functional gain, $f(x)$ – see Figure 2. This is not surprising; for ε small enough, W^* is close to the identity operator and therefore $\xi = W^*f \approx f$.

As the value of ε is taken closer to $\varepsilon_{\max}$, the shape of $\xi_r(x)$ undergoes a large change. Figure 5 shows $\xi_r(x)$ again with $r = 2$ as computed using 64 equally spaced nodes, but now with $\varepsilon = (0.9)\varepsilon_{\max}$. The shape of $\xi_r(x)$ now resembles a combination of the shapes of the control and observer functional gains, $k(x)$ and $f(x)$ – see Figures 1 and 2. Further increasing ε to $(0.99)\varepsilon_{\max}$ causes the shape of $\xi_r(x)$ to resemble the control functional gain, $k(x)$; compare Figures 1 and 6. Also, as ε increases toward $\varepsilon_{\max}$, the size of $\xi_r(x)$ increases. This is expected due to the term $1/(1-\alpha\lambda_1)$ in the expression for ξ_r; as noted above, this becomes infinite as ε approaches $\varepsilon_{\max}$. For all computations, refining the finite element mesh or increasing r gave very little change in $\xi_r(x)$.

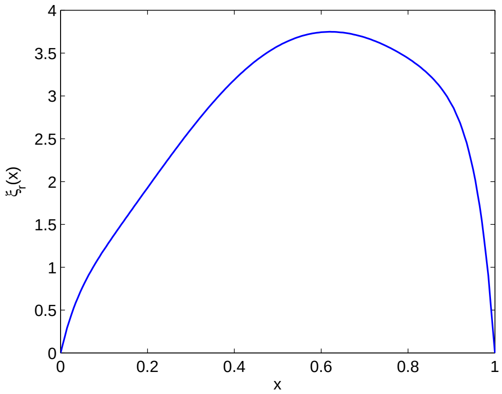

Fig. 5. The function $\xi_r(x)$ for $r = 2$ and $\varepsilon = (0.9)\varepsilon_{\max}$ as computed using 64 finite element nodes.

For all computations, the algorithm gave nearly identical results to computations using finite element matrix approximations of the operators A, B, and C. In particular, we found nearly identical maximum robustness margins and also ξ_r was very close to the function ξ computed using the matrix approximations.

In our computations, we found that using an adaptive solver for the differential equations (7) and (8) in Step 2 of the algorithm gave superior results compared to integrating these differential equations using the trapezoid rule with a constant time step. For example, the balanced POD eigenvalues computed using the adaptive solver matched the computed eigenvalues using matrix approximations very closely, while the computation with the trapezoid rule only produced accurate results for the largest few eigenvalues. Decreasing the constant time step Δt in the trapezoid rule gave slow improvement. Also, the function ξ_r computed using the trapezoid rule with $\Delta t = 0.01$ and 64 equally spaced finite element nodes gave a norm error of 0.1 compared with ξ

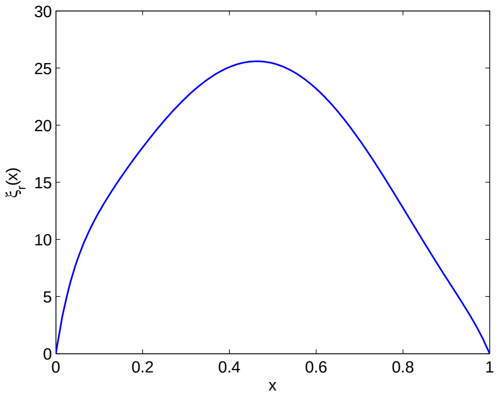

Fig. 6. The function $\xi_r(x)$ for $r = 2$ and $\varepsilon = (0.99)\varepsilon_{\max}$ as computed using 64 finite element nodes.

computed using matrix approximations; in contrast, the norm error using the adaptive solver is 2.1×10^{-3}. While these errors may not be significant for this problem, preliminary results for more complicated problems indicate that very accurate time stepping may be crucial to ensure accuracy in the balanced POD computations. We also note that the time stepping method was not an issue with the functional gain computations – the trapezoid rule Lyapunov solver with a constant time step gave excellent accuracy.

VI. Conclusions and Future Work

We presented an algorithm based on balanced POD for the computation of a robustly stabilizing control law for large-scale finite dimensional linear systems and a class of linear distributed parameter systems. Specifically, we considered the central controller, which is robust with respect to left coprime factor perturbations. We believe the algorithm can also be modified to compute the challenging features of other robust control laws, such as the standard $\mathcal{H}_\infty$ controller. Furthermore, although we used balanced POD in this work, it is likely that other balanced model reduction algorithms could be modified in a similar way for the robust control computations.

The computed central controller must be reduced in order to be implemented in real time. Also, in the reduction process, it would be desirable to keep as much robustness as possible. These topics will be considered in future work along with testing the algorithm on more challenging problems.

References

[1] K. Glover and D. McFarlane, “Robust stabilization of normalized coprime factor plant descriptions with H_∞-bounded uncertainty,” *IEEE Trans. Automat. Control*, vol. 34, no. 8, pp. 821–830, 1989.

[2] R. F. Curtain and H. J. Zwart, *An Introduction to Infinite-Dimensional Linear System Theory*. New York: Springer-Verlag, 1995.

[3] C. W. Rowley, “Model reduction for fluids, using balanced proper orthogonal decomposition,” *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, vol. 15, no. 3, pp. 997–1013, 2005.

[4] K. Willcox and J. Peraire, “Balanced model reduction via the proper orthogonal decomposition,” *AIAA Journal*, vol. 40, no. 11, pp. 2323–2330, 2002.

[5] S. Lall, J. E. Marsden, and S. Glavaški, “A subspace approach to balanced truncation for model reduction of nonlinear control systems,” *Internat. J. Robust Nonlinear Control*, vol. 12, no. 6, pp. 519–535, 2002.

[6] L. Sirovich, “Turbulence and the dynamics of coherent structures. I. Coherent structures,” *Quart. Appl. Math.*, vol. 45, no. 3, pp. 561–571, 1987.

[7] J. R. Singler and B. A. Batten, “A proper orthogonal decomposition approach to approximate balanced truncation of infinite dimensional linear systems,” *Int. J. Comput. Math.*, vol. 86, no. 2, pp. 355–371, 2009.

[8] J. R. Singler, “Snapshot algorithm for balanced model reduction of linear parabolic systems: convergence theory,” submitted.

[9] ——, “Optimality of standard and balanced proper orthogonal decomposition for data reconstruction,” submitted.

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

[11] H. T. Banks, R. C. Smith, and Y. Wang, *Smart Material Structures*. Paris: Masson and Wiley, 1996.

[12] J. A. Burns, B. B. King, and D. Rubio, “Feedback control of a thermal fluid using state estimation,” *Int. J. Comput. Fluid Dyn.*, vol. 11, no. 1-2, pp. 93–112, 1998.

[13] J. S. Gibson and A. Adamian, “Approximation theory for linear-quadratic-Gaussian optimal control of flexible structures,” *SIAM J. Control Optim.*, vol. 29, no. 1, pp. 1–37, 1991.

[14] H. T. Banks and K. Ito, “A numerical algorithm for optimal feedback gains in high-dimensional linear quadratic regulator problems,” *SIAM J. Control Optim.*, vol. 29, no. 3, pp. 499–515, 1991.

[15] K. Morris and C. Navasca, “Solution of algebraic Riccati equations arising in control of partial differential equations,” in *Control and Boundary Analysis*. Chapman & Hall/CRC, Boca Raton, FL, 2005, pp. 257–280.

[16] F. Feitzinger, T. Hylla, and E. W. Sachs, “Inexact Kleinman-Newton method for Riccati equations,” *SIAM J. Matrix Anal. Appl.*, vol. 31, no. 2, pp. 272–288, 2009.

[17] J. Borggaard, J. Burns, and L. Zietsman, “Computational challenges in control of partial differential equations,” in *Proceedings of the 2nd AIAA Flow Control Conference*, 2004, aIAA 2004-2526.

[18] J. A. Burns and K. P. Hulsing, “Numerical methods for approximating functional gains in LQR boundary control problems,” *Math. Comput. Modelling*, vol. 33, no. 1-3, pp. 89–100, 2001.

[19] J. A. Atwell and B. B. King, “Proper orthogonal decomposition for reduced basis feedback controllers for parabolic equations,” *Math. Comput. Modelling*, vol. 33, no. 1-3, pp. 1–19, 2001.

[20] H. T. Banks, R. C. H. del Rosario, and R. C. Smith, “Reduced-order model feedback control design: numerical implementation in a thin shell model,” *IEEE Trans. Automat. Control*, vol. 45, no. 7, pp. 1312–1324, 2000.

[21] G. M. Kepler, H. T. Tran, and H. T. Banks, “Reduced order model compensator control of species transport in a CVD reactor,” *Optimal Control Appl. Methods*, vol. 21, no. 4, pp. 143–160, 2000.

[22] J. R. Singler, “Convergent snapshot algorithms for infinite dimensional Lyapunov equations,” submitted.

[23] ——, “Approximate low rank solutions of Lyapunov equations via proper orthogonal decomposition,” in *Proceedings of the 2008 American Control Conference*, 2008, pp. 267–272.

[24] B. T. Dickinson, J. R. Singler, and B. A. Batten, “A snapshot algorithm for linear feedback flow control design,” in *Proceedings of the AIAA Infotech@Aerospace Conference and AIAA Unmanned...Unlimited Conference*, 2009, AIAA paper number 2009-1961.

[25] J. R. Singler and B. A. Batten, “A comparison of balanced truncation methods for closed loop systems,” in *Proceedings of the American Control Conference*, 2009.

[26] D. L. Kleinman, “On an iterative technique for Riccati equation computations,” *IEEE Trans. Autom. Control*, vol. AC-13, pp. 114–115, 1968.

[27] J. A. Burns, E. W. Sachs, and L. Zietsman, “Mesh independence of Kleinman–Newton iterations for Riccati equations in Hilbert space,” *SIAM J. Control Optim.*, vol. 47, no. 5, pp. 2663–2692, 2008.

[28] J. Weidmann, *Linear Operators in Hilbert Spaces*. New York: Springer-Verlag, 1980.