SOLUTION OF ALGEBRAIC RICCATI EQUATIONS ARISING IN CONTROL OF PARTIAL DIFFERENTIAL EQUATIONS

Kirsten Morris
Department of Applied Mathematics
University of Waterloo
Waterloo, CANADA N2L 3G1
kmorris@uwaterloo.ca

Carmeliza Navasca
Dept. of Mathematics
University of California
Los Angeles, CA 90095 USA
navasca@math.ucla.edu

Abstract Algebraic Riccati equations of large dimension arise when using approximations to design controllers for systems modelled by partial differential equations. For large model order direct solution methods based on eigenvector calculation fail. In this paper we describe an iterative method that takes advantage of several special features of these problems: (1) sparsity of the matrices (2) much fewer controls than approximation order and (3) convergence of the control with increasing model order. The algorithm is straightforward to code. Performance is illustrated with a number of standard examples.

Introduction

We consider the problem of calculating feedback controls for systems modelled by partial differential or delay differential equations. In these systems the state $x(t)$ lies in an infinite-dimensional space. A classical controller design objective is to find a control $u(t)$ so that the objective function

$$\int_0^\infty \langle Cx(t), Cx(t)\rangle + u^*(t)Ru(t)dt \tag{1}$$

is minimized where R is a positive definite matrix and the observation $C \in \mathcal{L}(X, \mathcal{R}^p)$. The theoretical solution to this problem for many infinite-dimensional systems parallels the theory for finite-dimensional systems [9, 16, 17, e.g.]. In practice, the control is calculated through approximation. This leads to solving an algebraic Riccati equation

$$A^*P + PA - PBR^{-1}B^*P = -C^*C. \tag{2}$$

for a feedback operator

$$K = -R^{-1}B'P. \tag{3}$$

The matrices A, B, C arise in a finite dimensional approximation of the infinite dimensional system. Let n indicate the order of the approximation, m the number of control inputs and p the number of observations. Thus, A is $n \times n$, B is $n \times m$ and C is $p \times n$. There have been many papers written describing conditions under which approximations lead to approximating controls that converge to

the control for the original infinite-dimensional system [3, 10, 13, 16, 17, e.g.]. In this paper we will assume that an approximation has been chosen so that a solution to the Riccati equation (2) exists for sufficiently large n and also that the approximating feedback operators converge.

For problems where the model order is small, $n < 50$, a direct method based on calculating the eigenvectors of the associated Hamiltonian works well [18]. Due the limitations of the calculation of eigenvectors for large non-symmetric matrices, this method is not suitable for problems where n becomes large.

Unfortunately, many infinite-dimensional control problems lead to Riccati equations of large order. This is particularly evident in control of systems modelled by partial differential equations with more than one space dimension. For such problems, iterative methods are more appropriate. There are two methods that may be used: Chrandrasekhar and Newton-Kleinman iterations.

In Chrandrasekhar iterations, the Riccati equation is not itself solved directly [2, 6]. A system of 2 differential equations

$$\begin{aligned} \dot{K}(t) &= -B^*L^*(t)L(t), & K(0) = 0, \\ \dot{L}(t) &= L(t)(A - BK(t)), & L(0) = C, \end{aligned}$$

is solved for $K \in R^{m\times n}$, $L \in R^{p\times n}$. The feedback operator is obtained as $\lim_{t\to-\infty} K(t)$. The advantage to this approach is that the number of controls m and number of observations p is typically much less than the approximation model order n. This leads to significant savings in storage. Furthermore, the matrices arising in approximation are typically sparse and this can be used in implementation of this algorithm. Unfortunately, the convergence of $K(t)$ can be very slow and a very accurate algorithm suitable for stiff systems must be used. This can lead to very large computation times.

Another approach to solving large Riccati equations is the Newton-Kleinman method [15]. The Riccati equation (2) can be rewritten as

$$(A - BK)^*P + P(A - BK) = -C^*C - K^*RK. \tag{4}$$

We say a matrix A_o is *Hurwitz* if $\sigma(A_o) \subset \mathbb{C}_-$ If $A - BK$ is Hurwitz, then the above equation is a Lyapunov equation. An initial feedback K_0 must be chosen so $A - BK_0$ is Hurwitz. Define $S_i = A - BK_i$, and solve the Lyapunov equation

$$S_i^*X_i + X_iS_i = -C^*C - K_i^*RK_i \tag{5}$$

for X_i and then update the feedback as $K_{i+1} = -R^{-1}B^*X_i$. If $A - BK_0$ is Hurwitz, then X_i converges quadratically to P [15]. For an arbitrary large Riccati equation, this condition may be difficult to satisfy. However, this condition is not restrictive for Riccati equations arising in control of infinite-dimensional systems. First, many of these systems are stable even when uncontrolled and so the initial iterate K_0 may be chosen as zero. Second, if the approximation procedure is valid then convergence of the feedback gains is obtained with increasing model order. Thus, a gain obtained from a lower order approximation, perhaps using a direct solution, may be used as an initial estimate, or ansatz, for a higher order approximation. This technique was used successfully in [12, 24]and later in this paper.

In this paper we use a modified Newton-Kleinman iteration first proposed by Banks and Ito [2]as a refinement for a partial solution to the Chandraskehar equation. In that paper, they partially solve the Chandrasekhar equations and then use the resulting feedback K as a stabilizing initial guess for a modified Newton-Kleinman method. Instead of the standard Newton-Kleinman form (5) above, Banks and Ito rewrote the Riccati equation in the form

$$(A - BK_i)^*X_i + X_i(A - BK_i) = -D_i^*D_i \tag{6}$$

where $X_i = P_{i-1} - P_i$, $K_{i+1} = K_i - B^TX_i$, and $D_i = K_i - K_{i-1}$. The resulting Lyapunov equation is solved for X_i. Equation (6) has fewer inhomogeneous terms than the equation in the standard

Newton-Kleinman method (5). Also, the non-homogeneous term D depends on m inputs, not the observation C. In [2]a Smith's method was used to solve the Lyapunov equations. Although convergent, this method is slow.

Solution of the Lyapunov equation is a key step in implementing either modified or standard Newton-Kleinman. The Lyapunov equations arising in the Newton-Kleinman method have several special features: (1) the model order n is generally much larger than m or p and (2) the matrices are often sparse. We use a recently developed method [19, 23]that uses these features, leading to an efficient algorithm. In the next section we describe the implementation of this Lyapunov solver. We then use this Lyapunov solver with both standard and modified Newton-Kleinman to solve a number of standard control examples, including one with several space variables. Our results indicate that modified Newton-Kleinman achieves considerable savings in computation time over standard Newton-Kleinman. We also found that using the solution from a lower-order approximation as an ansatz for a higher-order approximation significantly reduced the computation time.

1. Solution of Lyapunov Equation

Solution of a Lyapunov equation is a key step in each iteration of the Newton-Kleinman method. Thus, it is imperative to use a good Lyapunov algorithm. As for the Riccati equation, direct methods such as Bartels-Stewart [4]are only appropriate for low model order and do not take advantage of sparsity in the matrices. The Alternating Direction Implicit (ADI) and Smith methods are two well-known iterative schemes. These will be briefly described before describing a modification that leads to reduced memory requirements and faster computation.

Consider the Lyapunov equation

$$XA_o + A_o^* X = -DD^* \tag{7}$$

where $A_o \in \mathbb{R}^{n\times n}$ and $D \in \mathbb{R}^{n\times r}$. In the case of standard Newton-Kleinman, $r = m + p$ while for modified Newton-Kleinman, r is only m. If A_o is Hurwitz, then the Lyapunov equation has a symmetric positive semidefinite solution X. For $p < 0$, define $U = (A_o - pI)(A_o + pI)^{-1}$ and $V = -2p(A_o^* + pI)^{-1}DD^*(A_o + pI)^{-1}$. In Smith's method [26], equation (7) is rewritten. The solution X is found by using successive substitutions: $X = \lim_{i\to\infty} X_i$ where

$$X_i = U^* X_{i-1} U + V \tag{8}$$

with $X_0 = 0$. Convergence of the iterations can be improved by careful choice of the parameter p *e.g.* [25, pg. 197].

This method of successive substitition is unconditionally convergent, but has only linear convergence. The ADI method [20, 28]improves Smith's method by using a different parameter p_i at each step. Two alternating linear systems,

$$(A_o^* + p_i I)X_{i-\frac{1}{2}} = -DD^* - X_{i-1}(A_o - p_i I) \tag{9}$$

$$(A_o^* + p_i I)X_i^* = -DD^* - X_{i-\frac{1}{2}}^*(A_o - p_i I) \tag{10}$$

are solved recursively starting with $X_0 = 0 \in \mathbb{R}^{n\times n}$ and parameters $p_i < 0$. If all parameters $p_i = p$ then equations (9,10) reduce to Smith's method. If the ADI parameters p_i are chosen appropriately, then convergence is obtained in J iterations where $J \ll n$. Choice of the ADI parameters is discussed below.

If A_o is sparse, then the linear systems (9,10) can be solved efficiently. However, full calculation of the dense iterates X_i is required at each step. Setting $X_0 = 0$, it can be easily shown that X_i is symmetric and positive semidefinite for all i, and so we can write $X = ZZ^*$ where Z is a Cholesky factor of X [19, 23]. (A Cholesky factor does not need to be square or be lower triangular.)

Table 1. Cholesky-ADI Method

Given A_o and D
Choose ADI parameters $\{p_1, \ldots, p_J\}$ with $\Re(p_i) < 0$
Define $z_1 = \sqrt{-2p_1}(A_o^* + p_1 I)^{-1} D$
and $Z_1 = [z_1]$
For $i = 2, \ldots, J$
Define $W_i = (\frac{\sqrt{-2p_{i+1}}}{\sqrt{-2p_i}})[I - (p_{i+1} - p_i)(A_o^* + p_{i+1} I)^{-1}]$
(1) $z_i = W_i z_{i-1}$
(2) If $\|z\| > tol$
$Z_i = [Z_{i-1} \;\; z_i]$
Else, stop.

Substituting $Z_i Z_i^*$ for X_i in (9,10) and setting $X_0 = 0$, we obtain the following iterates for

$$\begin{aligned} Z_1 &= \sqrt{-2p_1}(A_o^* + p_1 I)^{-1} D \\ Z_i &= [\sqrt{-2p_i}(A_o^* + p_i I)^{-1} D, \;\; (A_o^* + p_i I)^{-1}(A_o^* - p_i I) Z_{i-1}] \end{aligned} \tag{11}$$

Note that $Z_1 \in \mathbb{R}^{n \times r}$, $Z_2 \in \mathbb{R}^{n \times 2r}$, and $Z_i \in \mathbb{R}^{n \times ir}$. Recall that for standard Newton-Kleinman, r is the sum of observations p and controls m. For modified Newton-Kleinman r is equal to m. In practice, the number of controls m (and often the number of observations p) is much less than the model order, n. This form of solution results in considerable savings in computation time and memory. The algorithm is stopped when the Cholesky iterations converge within some tolerance.

In [19]these iterates are reformulated in a more efficient form, using the observation that the order in which the ADI parameters are used is irrelevant. This leads to the algorithm shown in Table 1.

1.1 ADI Parameter Selection

As mentioned above, choice of the ADI parameters significantly affects the convergence of the ADI method. The parameter selection problem has been studied extensively [8, 20, 27, e.g.]. Optimal ADI parameters are the solution to the min-max problem

$$\{p_1, p_2, \ldots, p_J\} = \arg\min_{p_i} \max_{\lambda_j \in \sigma(A_o)} |\prod_j^J \frac{p_j - \lambda}{p_j + \lambda}|.$$

It is not feasible to solve this problem. First, solution of the Lyapunov equation arises as a step in the iterative solution of the Riccati equation and $A_o = A - BK_i$ where K_i is the feedback calculated at the $i^t h$ iterate. Thus, the matrix A_o and its spectrum changes at each iterate. Second, when A_o is large, solving this eigenvalue problem is computationally difficult.

The optimal ADI parameters are approximated in several respects. First, the spectrum of the original matrix $A - BK_0$ is used and the resulting parameters used for each subsequent Lyapunov solution. In most applications, A is Hurwitz and so we can use the spectrum of the original matrix A. If these eigenvalues are real and contained in the interval $[-b, -a]$ then the solution to (11) is known in closed form [20]. For more general problems, the selection procedure in [8]yields parameters that are approximately optimal. Let λ_i indicate the eigenvalues of $A - BK_0$. Define

$$a = \min_i(\Re\lambda_i), \tag{12}$$

$$b = \max_i(\Re\lambda_i), \tag{13}$$

$$\alpha = \tan^{-1} \max_i |\frac{\Im\lambda_i}{\Re\lambda_i}|. \tag{14}$$

These parameters determine an elliptic domain Ω that contains the spectrum and is used to determine the ADI parameters p_i. The closeness of Ω to the smallest domain containing the spectrum affects the number of iterations required for convergence of the Cholesky-ADI. The parameter α is the maximum angle between the eigenvalues and the real axis. When the spectrum contains lightly damped complex eigenvalues, α is close to $\pi/2$. In this case, Ω is a poor estimate of this domain. This point is investigated in the third example below.

2. Benchmark Examples

In this section we test the algorithm with a number of standard examples: a one-dimensional heat equation, a two-dimensional partial differential equation and a beam equation. All computations were done within MATLAB on a a computer with two 1.2 GHz AMD processors. (Shorter computation time would be obtained by running optimized code outside of a package such as MATLAB. The CPU times are given only for comparision purposes.) The relative error for the Cholesky iterates was set to 10^{-8}.

2.1 Heat Equation

Consider the linear quadratic regulator problem of minimizing a cost functional [2, 7]

$$J(u) = \int_0^\infty (|Cz(t)|^2 + |u(t)|^2)dt$$

subject to

$$\begin{aligned} \frac{\partial z(t,x)}{\partial t} &= \frac{\partial^2 z(t,x)}{\partial x^2},\ x \in (0,1), \\ z(0,x) &= \psi(x) \end{aligned} \tag{15}$$

with boundary conditions

$$\begin{aligned} \frac{\partial z(t,0)}{\partial x} &= u(t) \\ \frac{\partial z(t,1)}{\partial x} &= 0. \end{aligned} \tag{16}$$

Setting

$$Cz(t) = \int_0^1 z(t,x)dx, \tag{17}$$

and $R = 1$, the solution to the infinite-dimensional Riccati equation is

$$Kz = \int_0^1 k(x)z(x)dx$$

where $k = 1$ [3]. Thus, for this problem we have an exact solution to which we can compare the approximations.

The equations (14-17) are discretized using the standard Galerkin approximation with linear spline finite element basis on an uniform partition of $[0, 1]$. The resulting A matrix is symmetric and tridiagonal while B is a column vector with only one non-zero entry. Denote each basis element by l_i, $i = 1..n$. For an approximation with n elements, the approximating optimal feedback operator K is

$$Kz = \int_0^1 k^n(x)z(x)dx, \tag{18}$$

Newton-Kleinmann Iteration	Optimal Feedback Gain
1	50.005
2	25.0125
3	12.5262
4	6.303
5	3.2308
6	1.7702
7	1.1675
8	1.012
9	1.0001
10	1
11	1

Table 2. Heat Equation: Feedback Gain at each Newton-Kleinman Iteration

Table 3. Heat Equation: Standard Newton-Kleinman Iterations

n	Newton-Kleinman Itn's	Lyapunov Itn's	CPU time
25	11	19,22,23,26,27,29 30,31,31,31,31	0.83
50	11	24,26,28,30,32,34 35,35,35,35,35	1.2
100	11	28,31,32,35,36,38 39,40,40,40,40	3.49
200	11	33,35,37,39,41,43 44,44,44,44,44	23.1

where $k^n(x) = \sum_{i=1}^{n} k_i l_i(x)$. The solutions to the approximating Riccati equations converge [3, 13]and so do the feedback operators.

Table 2 shows the approximated optimal feedback gain at each Newton-Kleinman iteration. The data in Table 2 is identical for $n = 25,\ 50,\ 100,\ 200$ and for both Newton-Kleinman methods. The error in K versus Newton-Kleinman iteration in shown in Figure 1 for standard Newton-Kleinman and in Figure 2 for the modified algorithm. In Tables 3 and 4 we compare the number of Newton-Kleinman and Lyapunov iterations as well as the CPU time per order n. We use the ansatz $k_0(x) = 100$ for all n. With the modified algorithm, there are $1-2$ fewer Riccati loops than with the the original Newton-Kleinman iteration. Also, the modified Newton-Kleinman method requires fewer Lyapunov iterations within the last few Newton-Kleinman loops. The computation time with the modified Newton-Kleinman algorithm is significantly less than that of the original algorithm.

2.2 Two-Dimensional Example

Define the rectangle $\Omega = [0,1] \times [0,1]$ with boundary $\partial\Omega$. Consider the two-dimensional partial differential equation [7]

$$\begin{array}{lcll} \frac{\partial z}{\partial t} & = & \frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2} + 20\frac{\partial z}{\partial y} + 100z = f(x,y)u(t), & (x,y) \in \Omega \\ z(x,y,t) & = & 0, & (x,y) \in \partial\Omega \end{array} \tag{19}$$

where z is a function of x, y and t. Let

$$f(x,y) = \begin{cases} 100, & \text{if } .1 < \text{x} < .3 \text{ and } .4 < \text{y} < .6 \\ 0, & \text{else} \end{cases}.$$

Table 4. Heat Equation: Modified Newton-Kleinman Iterations

n	Newton-Kleinman Itn's	Lyapunov Itn's	CPU time
25	10	19,22,23,26,27,29 30,31,29,1	0.66
50	10	24,26,28,30,32,34 35,35,33,1	0.94
100	9	28,31,32,35,36,38 39,40,1	2.32
200	9	33,35,37,39,41,43 44,44,1	14.2

Table 5. Iterations for 2-d Equation (Newton-Kleinman)

grid	K_0	n	Newton-Kleinman Itn's	Lyapunov Itn's	CPU time
12×12	0	144	14	12,44,41,39,36,34,32 30,28,27,27,27,27,27	4.98
23×12	0	276	15	16,47,45,42,40,38,35,33 31,30,30,30,30,30,30	22.2
23×12	K_{proj}^{12x12}	276	6	29,30,30,30,30,30	10.8
23×23	0	529	16	20,51,48,46,44,41,39,37 35,34,33,33,32,32,32,32	139.
23×23	K_{proj}^{23x12}	529	5	33,32,32,32,32	65.7

Central difference approximations are used to discretize (19) on a grid of $N \times M$ points. The resulting approximation has dimension $n = N \times M$: $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times 1}$. The A matrix is sparse with at most 5 non-zero entries in any row. The B matrix is a sparse column vector. We chose $C = B^*$ and $R = 1$.

We solved the Riccati equation on a number of grids, using both standard and modified Newton-Kleinman methods. The data is shown in Tables 5 and 6. Modified Newton-Kleinman is clearly much more efficient. Fewer Lyapunov iterations are required for convergence and this leads to a reduction in computation time of nearly 50%.

We also investigated the use of non-zero initial estimates for K in reducing computation time. We first solve the Riccati equation on a 12×12 grid. Since $\sigma(A) \subset \mathbb{C}_-$, $K_0^{144} = 0$ is a possible ansatz. It required 13 Newton-Kleinman iterations and a total of 419 Lyapunov iterations to obtain a relative error in K of 10^{-11}. Linear interpolation was used to project this solution to a function on a finer grid, 23×12, where $n = 276$. Indicate this projection by $K_{proj}^{12\times12}$. On the finer grid 23×12 where $n = 276$, we used both zero and $K_{proj}^{12\times12}$ as initial estimates. As indicated in Table 6, the error of K was 10^{-12} after only 150 Lyapunov and 54 Newton-Kleinman iterations. The same procedure is applied to generate a guess K_0^{529} where the mesh is 23×23 and $n = 529$. Neglecting the computation time to perform the projection, use of a previous solution lead a total computation time over both grids of only 6.3 seconds versus 11.9 seconds for $n = 276$ and a total computation time of only 27.8 versus 79.5 for $n = 529$. Similar improvements in computation time were obtained with standard Newton-Kleinman.

3. Euler-Bernoulli Beam

Consider a Euler-Bernoulli beam clamped at one end ($r = 0$) and free to vibrate at the other end ($r = 1$). Let $w(r, t)$ denote the deflection of the beam from its rigid body motion at time t and position r. The deflection is controlled by applying a a torque $u(t)$ at the clamped end ($r = 0$).

Table 6. Iterations for 2-d Equation (Modified Newton-Kleinman)

grid	K_0	n	Newton-Kleinman Itn's	Lyapunov Itn's	CPU time
12×12	0	144	13	12,44,41,39,36,34,32 30,28,27,27,27,1	2.94
23×12	0	276	13	16,47,45,42,40,38,35 33,31,30,30,30,29	11.9
23×12	K_{proj}^{12x12}	276	4	29,30,30,29	3.36
23×23	0	529	14	20,50,48,46,44,41,39 37,35,34,33,33,33,31	79.5
23×23	K_{proj}^{23x12}	529	4	33,32,32,1	21.5

We assume that the hub inertia I_h is much larger than the beam inertia I_b so that $I_h\ddot{\theta} \approx u(t)$. The partial differential equation model with Kelvin-Voigt and viscous damping is

$$w_{tt}(r,t) + C_v w_t(r,t) + \frac{\partial^2}{\partial r^2}\left[C_d I_b w_{rrt}(x,t) + \frac{EI_r}{\rho A} w_{rr}(r,t)\right] = \frac{\rho r}{I_h} u(t), \tag{20}$$

with boundary conditions

$$\begin{aligned} w(0,t) &= 0 \\ w_r(1,t) &= 0. \\ EIw_{rr}(1,t) + C_d I_b w_{rrt}(1,t) &= 0 \\ \frac{\partial}{\partial r}[EI(1)w_{rr}(r,t) + C_d I_b w_{rrt}(r,t)]_{r=1} &= 0. \end{aligned}$$

The values of the physical parameters in Table 1.3 are as in [1].

Define H be the closed linear subspace of the Sobolev space $H^2(0,1)$

$$H = \left\{ w \in H^2(0,1) : w(0) = \frac{dw}{dr}(0) = 0 \right\}$$

and define the state-space to be $X = H \times L^2(0,1)$ with state $z(t) = (w(\cdot,t), \frac{\partial}{\partial t} w(\cdot,t))$. A state-space formulation of the above partial differential equation problem is

$$\frac{d}{dt}x(t) = Ax(t) + Bu(t),$$

where

$$A = \begin{bmatrix} 0 & I \\ -\frac{EI}{\rho}\frac{d^4}{dr^4} & -\frac{C_d I}{\rho}\frac{d^4}{dr^4} - \frac{C_v}{\rho} \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ \frac{r}{I_h} \end{bmatrix},$$

with domain

$$\begin{aligned} \mathrm{dom}\,(A) = \{(\phi,\psi) \in X : \psi \in H \ \text{ and} \\ M = EI\tfrac{d^2}{dr^2}\phi + C_d I \tfrac{d^2}{dr^2}\psi \in H^2(0,1) \ \text{ with } M(L) = \tfrac{d}{dr}M(L) = 0\}. \end{aligned}$$

We use $R = 1$ and define C by the tip position:

$$w(1,t) = C[w(x,t) \ \ \dot{w}(x,t)].$$

E	2.68×10^{10} N/m^2
I_b	1.64×10^{-9} m^4
ρ	1.02087 kg/m
C_v	1.8039 Ns/m
C_d	1.99×10^5 Ns/m
L	1 m
I_h	121.9748 $kg\ m^2$
d	.041 kg^{-1}

Table 7. Table of physical parameters.

Let $H^N \subset H$ be a sequence of finite-dimensional subspaces spanned by the standard cubic B-splines with a uniform partition of $[0, 1]$ into N subintervals. This yields an approximation in $H^N \times H^N$ [14, e.g.]of dimension $n = 2N$.

This approximation method yields a sequence of solutions to the algebraic Riccati equation that converge strongly to the solution to the infinite-dimensonal Riccati equation corresponding to the original partial differential equation description [3, 21].

The spectrum of A for various n is shown in Figure 4. For small values of n, the spectrum of A_n only contains complex eigenvalues with fairly constant angle. As n increases, the spectrum curves into the real axis. For large values of n, the spectrum shows behaviour like that of the original differential operator, and contains two branches on the real axis. For these large values of n, the ADI parameters are complex numbers. We calculated the complex ADI parameters as in [8]. Although there are methods to efficiently calculate with complex parameters by splitting the calculation into 2 real parts [19]their presence increases computation time.

Figure 5 shows the total Lyapunov iterations for various values of n. Although the number of Newton-Kleinman iteration remained at 2 for all n, the number of Lyapunov iterates increases as $n \to \infty$.

Figure 6 shows the change in the spectrum of A as C_d is varied. Essentially, increasing C_d increases the angle that the spectrum makes with the imaginary axis. Recall that the spectral bounds α, a, and b define the elliptic function domain that contains the spectrum of A [20]. The ADI parameters depend entirely on these bounds. The quantity $\frac{\pi}{2} - \alpha$ is the angle between the spectrum and the imaginary axis and so $\alpha \to \pi/2$ as C_d is decreased. Figure 7 shows the effect of varying C_d on the number of iterations required for convergence. Larger values of C_d (i.e. smaller values of α) leads to a decreasing number of iterations. Small values of C_d lead to a large number of required iterations in each solution of a Lyapunov equation.

There are several possible reasons for this. As the spectrum of A flattens with increasing C_d the spectral bounds (12-14) give sharper estimates for the elliptic function domain Ω and thus the ADI parameters are closer to optimal. Improvement in calculation of ADI parameters for problems where the spectrum is nearly vertical in the complex plane is an open problem.

Another explanation lies in the nature of the mathematical problem being solved. If $C_d > 0$ the semigroup for the original partial differential equation is parabolic and the solution to the Riccati equation converges uniformly in operator norm [16, chap.4]. However, if $C_d = 0$, the partial differential equation is hyperbolic and only strong convergence of the solution is obtained [17]. Thus, one might expect a greater number of iterations in the Lyapunov loop to be required as C_d is decreased. Any $X \in R^{n \times n}$ to the matrix Lyapunov equation is symmetric and positive semi-definite and so we can order its eigenvalues $\lambda_1 \geq \lambda_2 \geq ...\lambda_n \geq 0$. The ability to approximate X by a matrix of lower rank $\hat{X}$ is determined by the following relation [11, Thm. 2.5.2]

$$\min_{rank\tilde{X} \leq k-1} \frac{\|X - \tilde{X}\|}{\|X\|} = \frac{\lambda_k(X)}{\lambda_1(X)}.$$

Table 8. Beam: : Effect of Changing C_d (Standard Newton-Kleinman)

C_v	C_d	α	Newton-Kleinman Itn's	Lyapunov Itn's	CPU time
2	1×10^4	1.5699	–	–	–
	3×10^5	1.5661	–	–	–
	4×10^5	1.5654	3	1620;1620;1620	63.14
	1×10^7	1.5370	3	1316;1316;1316	42.91
	1×10^8	1.4852	3	744;744;744	18.01
	5×10^8	1.3102	3	301;301;301	5.32

Table 9. Beam: Effect of Changing C_d (Modified Newton-Kleinman)

C_v	C_d	α	Newton-Kleinman It's	Lyapunov It's	CPU time
2	1×10^4	1.5699	2	–	–
	3×10^5	1.5661	2	–	–
	4×10^5	1.5654	2	1620;1	24.83
	1×10^7	1.5370	2	1316;1	16.79
	1×10^8	1.4852	2	744;1	7.49
	5×10^8	1.3102	2	301;1	2.32

This ratio is plotted for several values of C_d in Figures 8 and 9. For larger values of C_d the solution X is closer to a low rank matrix than it is for smaller values of C_d. Recall that the CF-ADI algorithm used here starts with a rank 1 initial estimate of the Cholesky factor and the rank of the solution is increased at each step. The fact that the solution X is closer to a low rank matrix for larger values of C_d implies a smaller number of iterations are required for convergence. If the fundamental reason for the slow convergence with small C_d is the "hyperbolic-like" behaviour of the problem, then this convergence will not be improved by better ADI parameter selection. This may have consequences for control of coupled acoustic-structure problems where the spectra are closer to those of hyperbolic systems than those of parabolic systems.

References

[1] H.T. Banks and D. J. Inman, "On Damping Mechanisms in Beams", *ICASE Report No. 89-64*, NASA, Langley, 1989.

[2] 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, 499-515, 1991.

[3] H. T. Banks and K. Kunisch, *The linear regulator problem for parabolic systems*, SIAM J. Control Optim., vol. 22, 5, 684-698, 1984.

[4] R.H. Bartels and W. Stewart, "Solution of the matrix equation AX+XB =C" *Comm. of ACM*, Vol. 15, pg. 820-826, 1972.

[5] D. Bau, and L. Trefethen, *Numerical Linear Algebra*, SIAM, Philadephia, 1997.

[6] J.A. Burns and K. P. Hulsing, "Numerical Methods for Approximating Functional Gains in LQR Boundary Control Problems", *Mathematical and Computer Modelling*, vol. 33, no. 1, pg. 89-100, 2001.

[7] Y. Chahlaoui and P. Van Dooren, *A collection of Benchmark examples for model reduction of linear time invariant dynamical systems*, SLICOT Working Note 2002-2, http://www.win.tue.nl/niconet/

[8] N. Ellner and E. L. Wachpress, *Alternating Direction Implicit Iteration for Systems with Complex Spectra*, SIAM J. Numer. Anal., vol. 28, no. 3, 859-870, 1991.

[9] J.S. Gibson, "The Riccati Integral Equations for Optimal Control Problems on Hilbert Spaces", *SIAM J. Control and Optim.*, Vol. 17, pg. 637-565, 1979.

[10] J.S. Gibson, "Linear-quadratic optimal control of hereditary differential systems: infinite dimensional Riccati equations and numerical approximations", *SIAM J. Control and Optim.*, Vol. 21 95-139, 1983.

[11] G.H. Golub and C.F. van Loan, *Matrix Computations*, John Hopkins, 1989.

[12] J.R. Grad and K. A. Morris, "Solving the Linear Quadratic Control Problem for Infinite-Dimensional Systems", *Computers and Mathematics with Applications*, Vol. 32, No. 9, 1996, pg. 99-119.

[13] K. Ito, Strong convergence and convergence rates of approximating solutions for algebraic Riccati equations in Hilbert spaces, *Distributed Parameter Systems*, eds. F. Kappel, K. Kunisch, W. Schappacher, Springer-Verlag, 1987, 151-166.

[14] K. Ito and K. A. Morris, *An approximation theory of solutions to operator Riccati equations for H^∞ control*, SIAM J. Control Optim., 36, 82-99, 1998.

[15] D. Kleinman, *On an Iterative Technique for Riccati Equation Computations*, IEEE Transactions on Automat. Control, 13, 114-115, 1968.

[16] I. Lasiecka and R. Triggiani, *Control Theory for Partial Differential Equations: Continuous and Approximation Theories*, Part 1, Cambridge University Press, 2000.

[17] I. Lasiecka and R. Triggiani, *Control Theory for Partial Differential Equations: Continuous and Approximation Theories*, Part 2, Cambridge University Press, 2000.

[18] A.J. Laub, "A Schur method for solving algebraic Riccati equations", *IEEE Trans. Auto. Control*, Vol. 24, pg. 913-921, 1979.

[19] J. R. Li and J White, *Low rank solution of Lyapunov equations*, *SIAM J. Matrix Anal. Appl.*, Vol. 24, pg. 260-280, 2002.

[20] A. Lu and E. L. Wachpress, *Solution of Lyapunov Equations by Alternating Direction Implicit Iteration*, Computers Math. Applic., 21, 9, 43-58, 1991.

[21] K.A. Morris, "Design of Finite-Dimensional Controllers for Infinite-Dimensional Systems by Approximation", *J. Mathematical Systems, Estimation and Control*, 4, 1-30, 1994.

[22] K.A. Morris, *Introduction to Feedback Control*, Harcourt-Brace, 2001.

[23] T. Penzl, "A Cyclic Low-Rank Smith Method for Large Sparse Lyapunov Equations", *SIAM J. Sci. Comput.*, Vol. 21, No. 4, pg. 1401-1418, 2000.

[24] I. G. Rosen and C. Wang, "A multilevel technique for the approximate solution of operator Lyapunov and algebraic Riccati equations", *SIAM J. Numer. Anal.* Vol. 32 514–541, 1994.

[25] Russell, D.L., *Mathematics of Finite Dimensional Control Systems: Theory and Design,* Marcel Dekker, New York, 1979.

[26] R.A. Smith, "Matrix Equation XA + BX = C", *SIAM Jour. of Applied Math.*, Vol. 16, pg. 198-201, 1968.

[27] E. L. Wachpress, *The ADI Model Problem*, preprint, 1995.

[28] E. Wachpress, "Iterative Solution of the Lyapunov Matrix Equation", *Appl. Math. Lett.*, Vol. 1, Pg. 87-90, 1988.

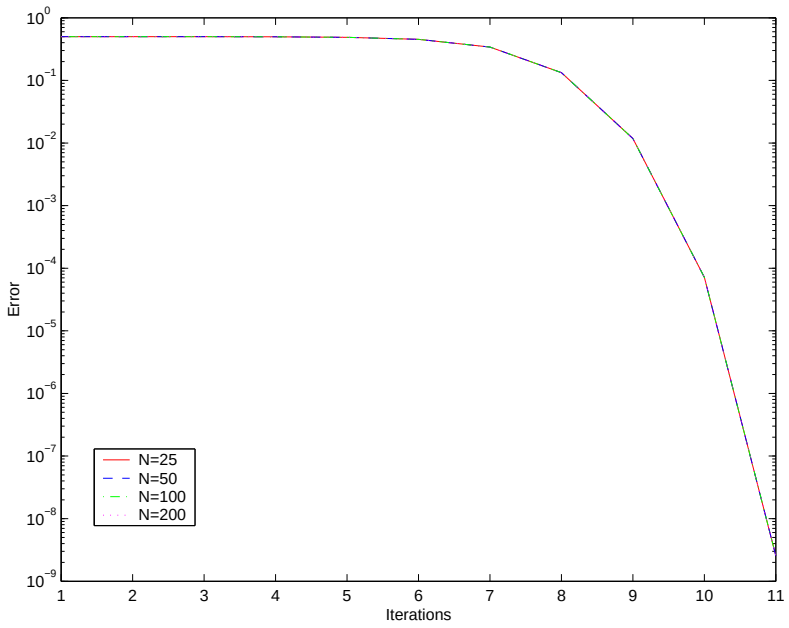

Figure 1. Heat Equation: Error versus Standard Newton-Kleinman Iterations

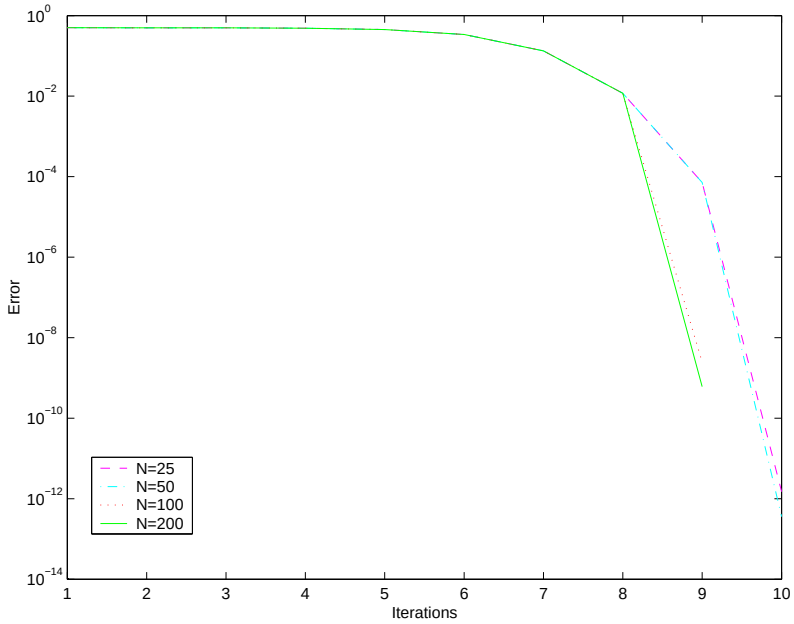

Figure 2. Heat Equation: Error versus Modified Newton-Kleinman Iterations

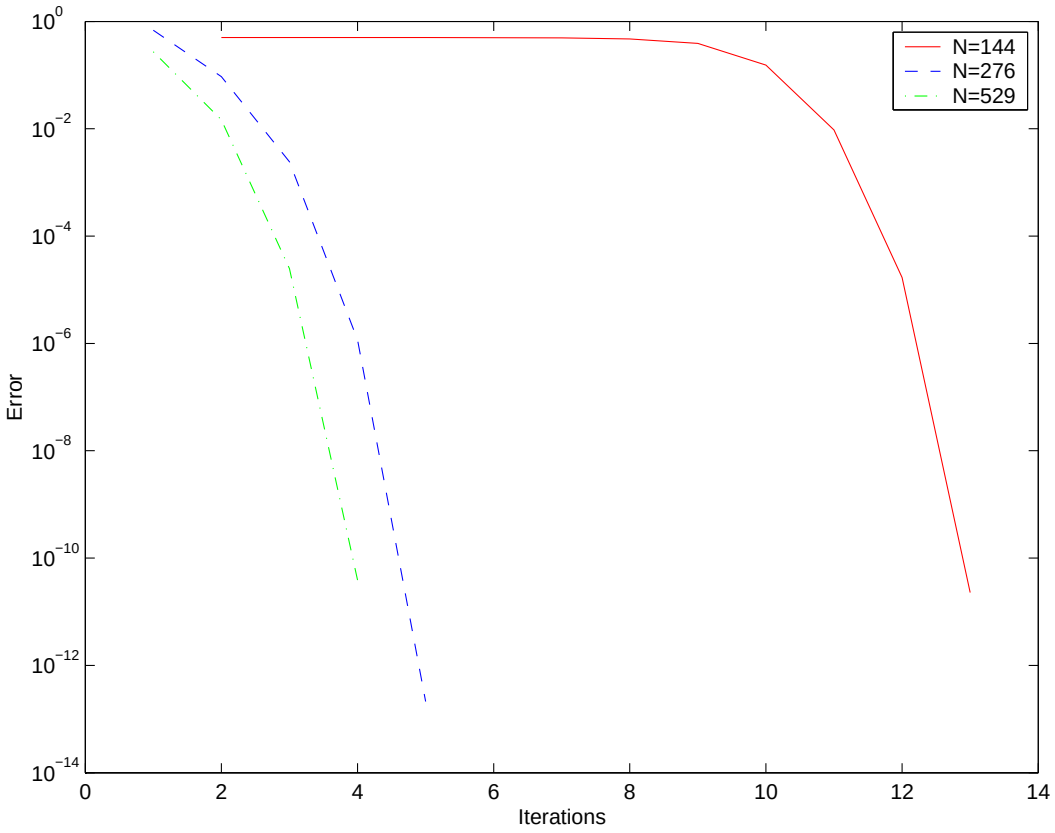

Figure 3. Convergence rate for two-d problem (modified Newton-Kleinman). Initial estimates: $K^{144} = 0$, $K^{276} = K^{144}_{proj}$, $K^{529} = K^{276}_{proj}$

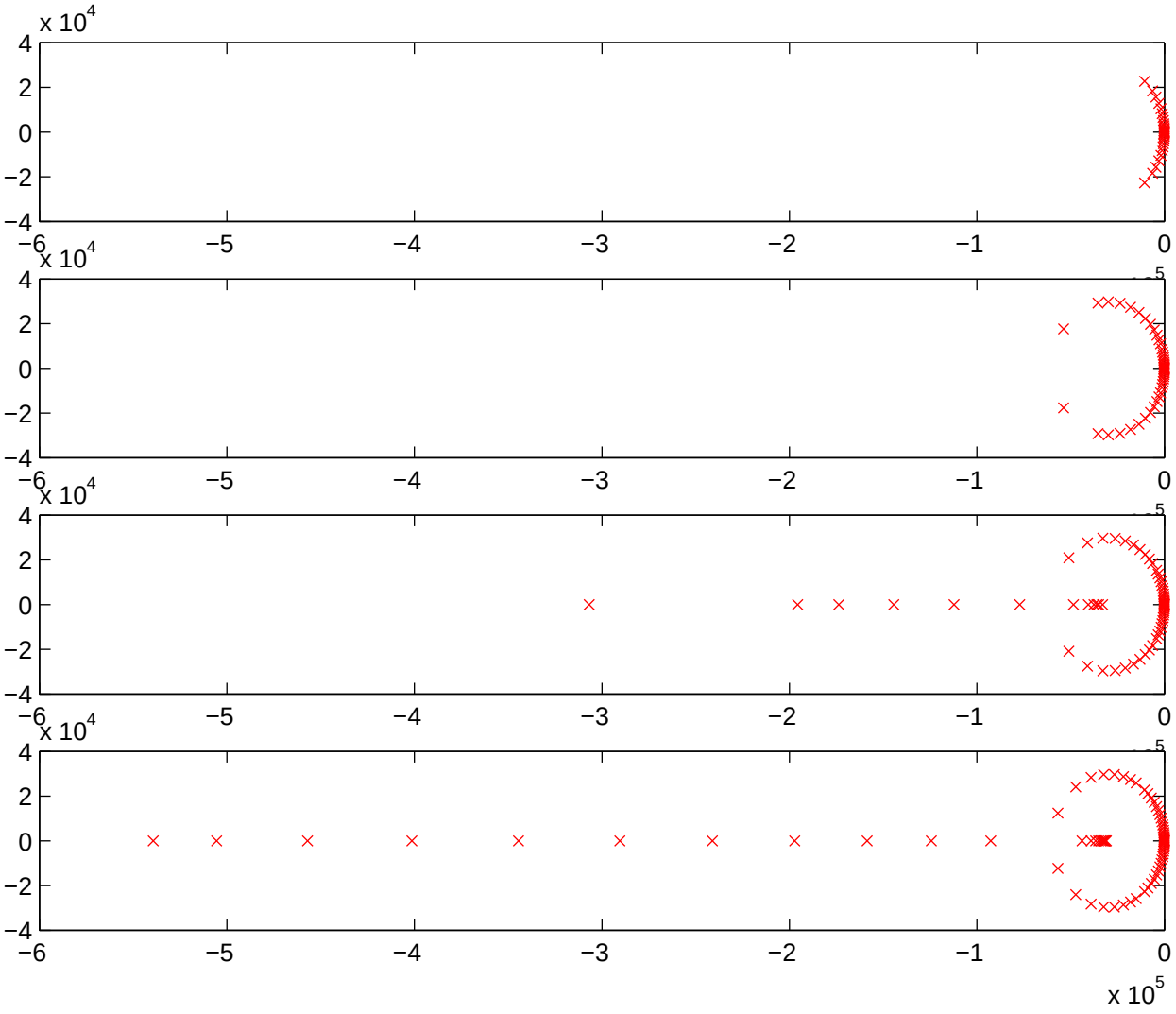

Figure 4. Beam: spectrum of A_n for $n = 48,\ 56,\ 64,\ 88$ ($C_d = 9 \times 10^5$).

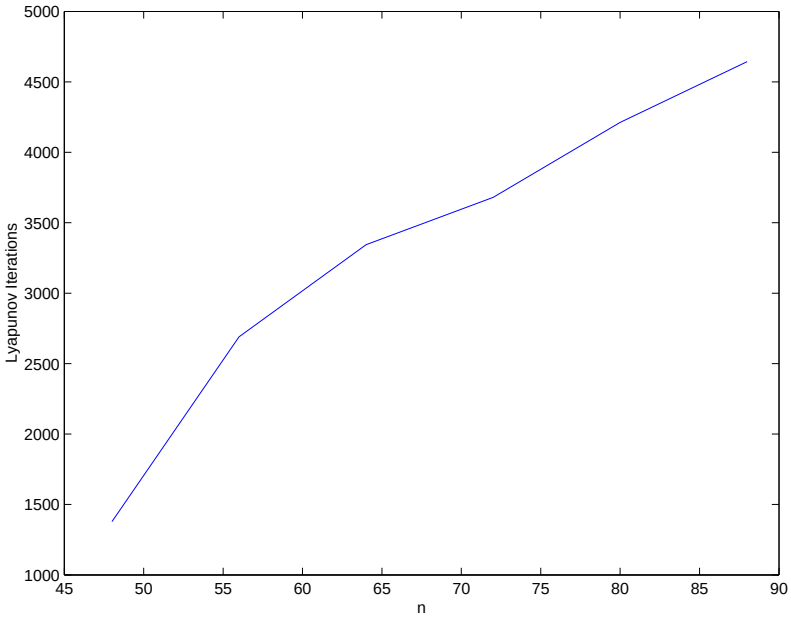

Figure 5. Beam: n vs. Lyapunov Iterations ($C_d = 9 \times 10^5$)

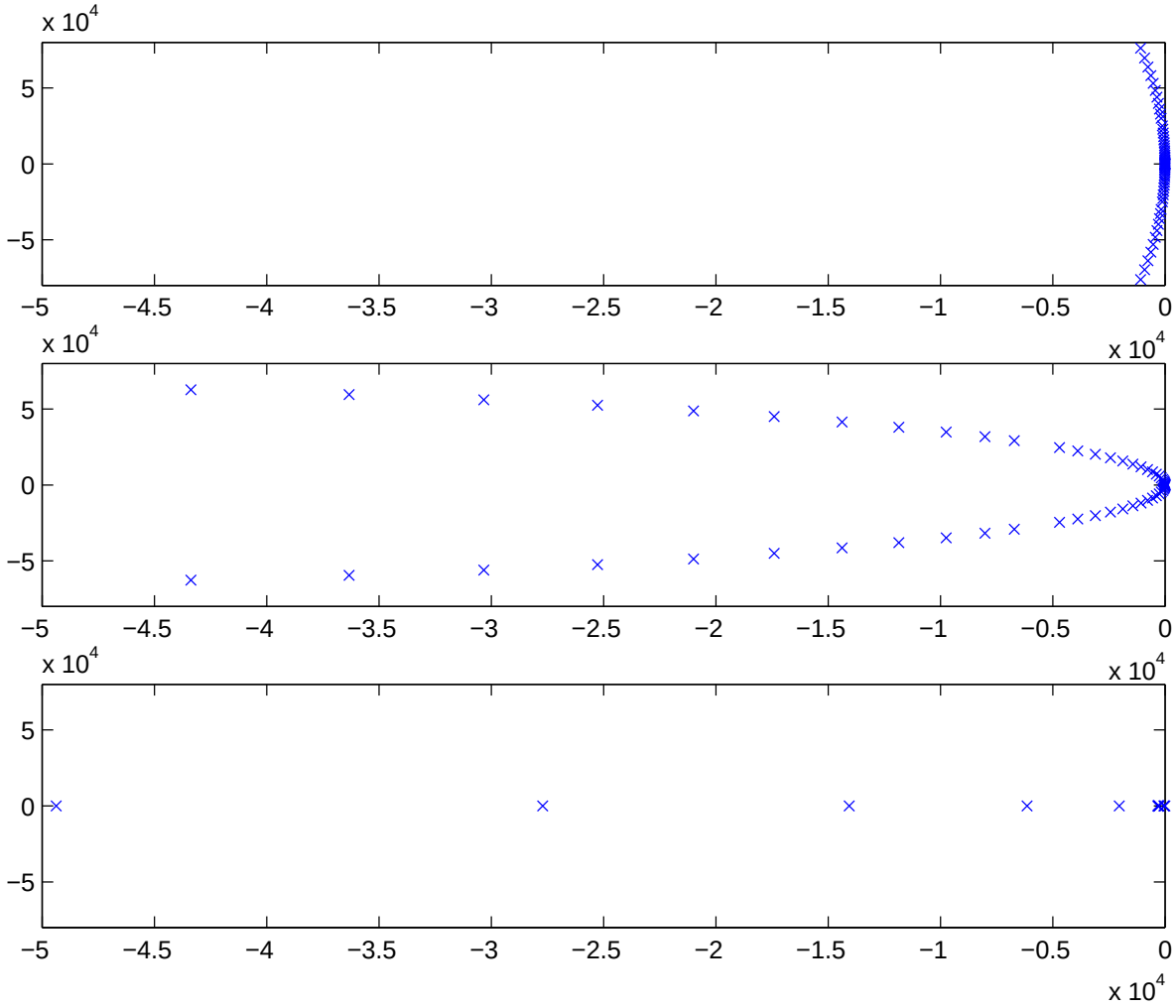

Figure 6. Beam: Spectrum of A for $C_d = 1x10^4,\ 4x10^5,\ 1x10^8$ $(n = 80)$

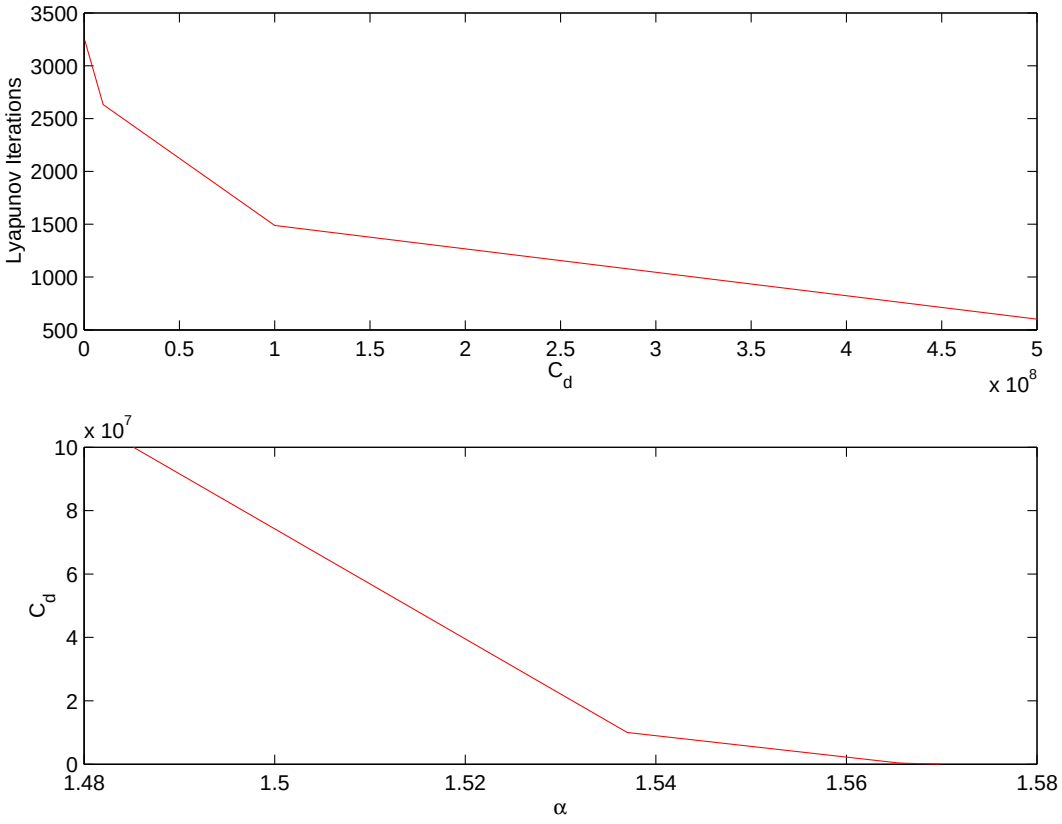

Figure 7. Beam: Effect of C_d on Lyapunov Iterations and α ($n = 80$)

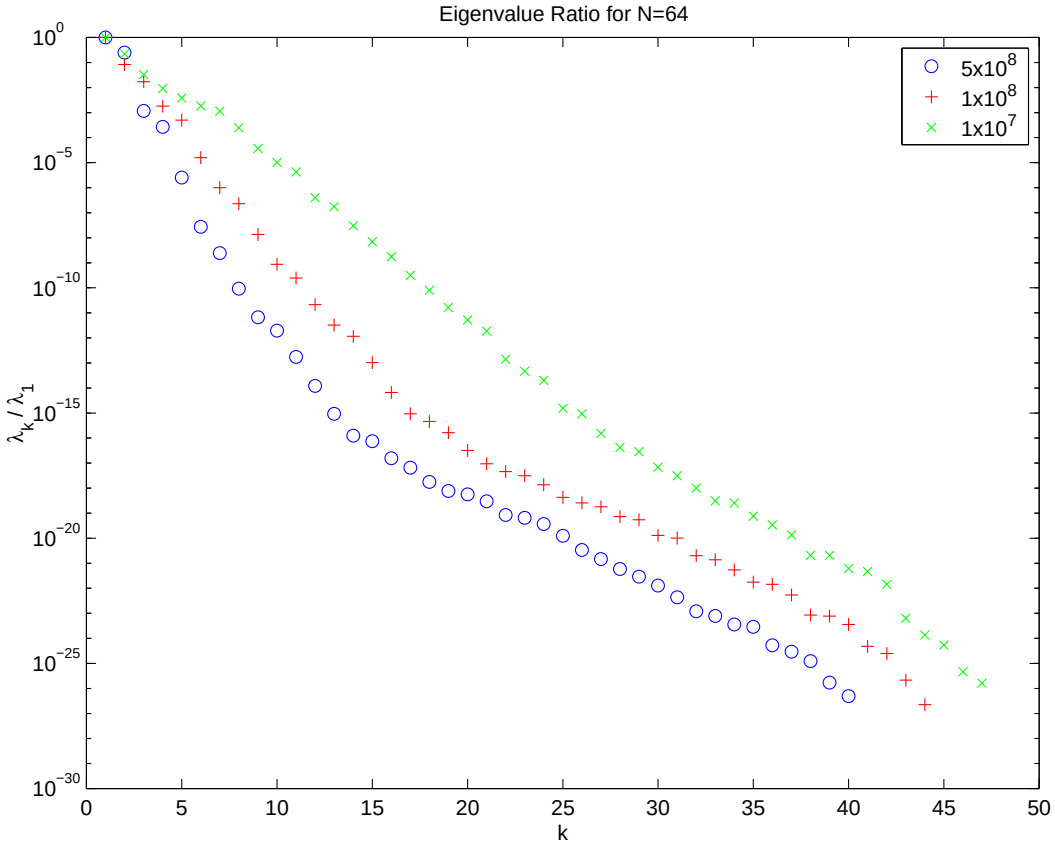

Figure 8. Beam: Effect of varying C_d on Low Rank Approximation

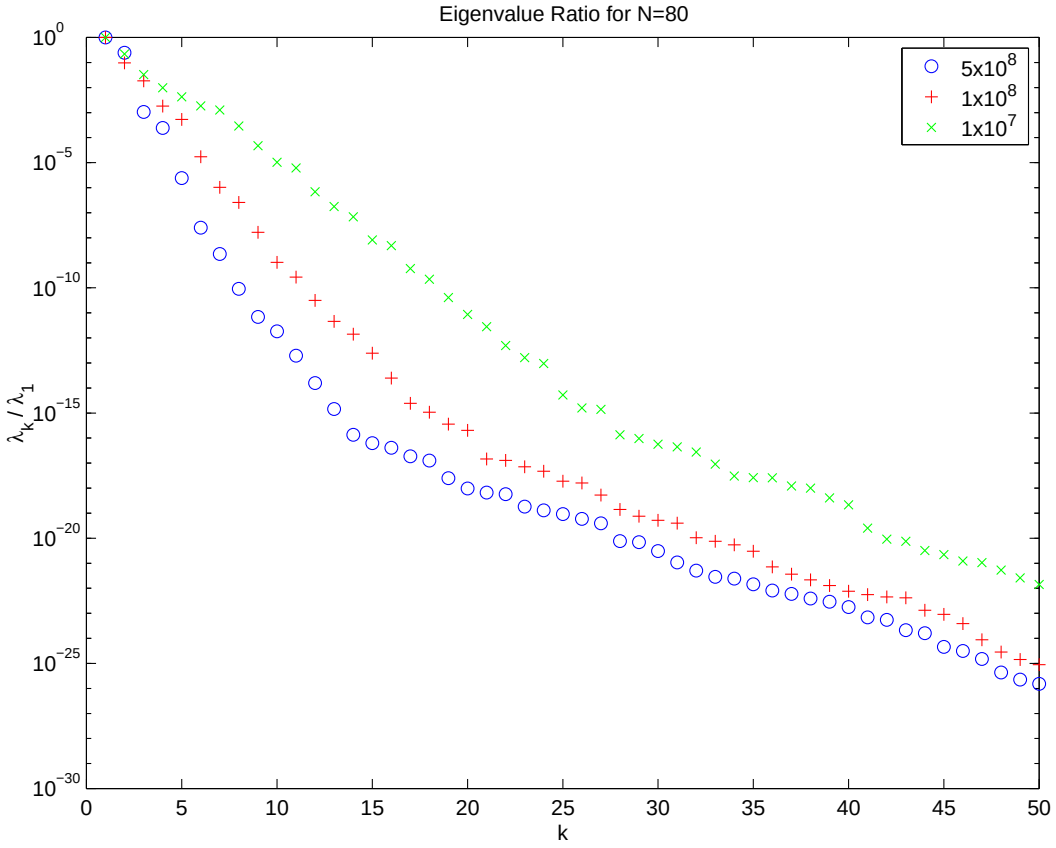

Figure 9. Beam: Effect of varying C_d on Low Rank Approximation