HOMOTOPY ALGORITHMS FOR THE H^2 AND THE COMBINED H^2/H^∞ MODEL ORDER REDUCTION PROBLEMS

by

Yuzhen Ge

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science and Applications

APPROVED:

Layne T. Watson

Joseph A. Ball

Christopher A. Beattie

April, 1993

Blacksburg, Virginia

LD
5055
U855
1993
G4
c.2

c.2

HOMOTOPY ALGORITHMS FOR THE H^2 AND THE COMBINED H^2/H^∞ MODEL ORDER REDUCTION PROBLEMS

by

Yuzhen Ge

Committee Chairman: Layne T. Watson

Computer Science

(ABSTRACT)

The problem of finding a reduced order model, optimal in the H^2 sense, to a given system model is a fundamental one in control system analysis and design. The addition of a H^∞ constraint to the H^2 optimal model reduction problem results in a more practical yet computationally more difficult problem. Without the global convergence of homotopy methods, both the H^2 optimal and the combined H^2/H^∞ model reduction problems are very difficult.

For both problems homotopy algorithms based on several formulations — input normal form; Ly, Bryson, and Cannon's 2×2 block parametrization; a new nonminimal parametrization — are developed and compared here. For the H^2 optimal model order reduction problem, these numerical algorithms are also compared with that based on Hyland and Bernstein's optimal projection equations.

Both the input normal form and Ly form are very efficient compared to the over parametrization formulation and the optimal projection equations approach, since they utilize the minimal number of possible degrees of freedom. However, they can fail to exist or be very ill conditioned. The conditions under which the input normal form and the Ly form become ill conditioned are examined.

The over-parametrization formulation solves the ill conditioning issue, and usually is more efficient than the approach based on solving the optimal projection equations for the H^2 optimal model reduction problem. However, the over-parametrization formulation introduces a very high order singularity at the solution, and it is doubtful whether this singularity can be overcome by using interpolation or other existing methods.

ACKNOWLEDGEMENTS.

The completion of this thesis owes greatly to Dr. Layne Watson, for his detail and enlightening guidance at every step of the process, for his patience and understanding when at times the progress seemed to be at a less satisfactory pace, and for his intuitive, knowledgeable, and rigorous suggestions and instructions, which made the discussions interesting and encouraging.

Emmanuel Collins and Dennis Bernstein explained to me many fine details about control theory as well as the new results in the area, which laid the foundation for the research in the thesis.

My thesis research could not have taken off so smoothly without Dragan Žigić's sharing with me his earlier results, and the software he developed.

I would also like to thank Dr. Joseph Ball and Dr. Christopher Beattie for their patience in reading this thesis and for their many valuable suggestions.

Finally I want to express my appreciation to my fiancé, Jiangeng Sun, for always being there for me.

TABLE OF CONTENTS

1. Introduction 1

   1.1. The H^2 optimal model order reduction problem 1

   1.2. The combined H^2/H^∞ model reduction problem 2

   1.3. The auxiliary minimization problem 4

2. Homotopy Algorithm Based on Input Normal Form Formulation
   for the H^2 Optimal Model Order Reduction Problem 6

   2.1. Input normal form formulations 6

   2.2. A homotopy approach based on the input normal form 8

   2.3. Numerical algorithm for input normal form homotopy 10

3. Comparison with Optimal Projection Equation Approach 13

4. Homotopy Algorithm Based on Ly's Formulation
   for the H^2 Optimal Model Order Reduction Problem 18

   4.1. Ly's formulation 18

   4.2. A homotopy approach based on the Ly formulation 19

   4.3. Numerical algorithm for Ly form homotopy 20

5. Numerical Results 21

6. Homotopy Algorithm Based on Over-Parametrization Formulation
   for the H^2 Optimal Model Order Reduction Problem 28

   6.1. Over-parametrization formulation 28

   6.2. A homotopy approach based on over-parametrization formulation 28

   6.3. Numerical algorithm for over-parametrization formulation 30

7. Comparison and Discussion 31

8. Homotopy algorithm based on the input normal form for
   the combined H^2/H^∞ model reduction problem 36

   8.1. The formulation 36

8.2. Numerical algorithm for input normal form homotopy 39

9. Homotopy algorithm based on Ly formulation for the combined H^2/H^∞ model reduction problem 41

10. Homotopy algorithm based on over-parametrization formulation for the combined H^2/H^∞ model reduction problem 44

11. H^2/H^∞ Results 47

12. Conclusions 50

References 51

Appendix A 55

Appendix B 58

Vita 87

LIST OF FIGURES

Figure 1. x_7 (OP) and θ_1 (INF) versus λ. 32

Figure 2. x_7 (OP) and θ_2 (INF) versus λ. 33

Figure 3. $\|H(s) - H_m(s)\|_\infty$, $100\,\mathcal{J}$, and $100\,J$ versus γ. 48

Figure 4. $\|H(s) - H_m(s)\|_\infty$ versus J. 49

LIST OF TABLES

Table 1. Algorithm measures for three formulations 31

Table 2. Comparison of methods 34

1. INTRODUCTION.

1.1. The H^2 optimal model order reduction problem.

The H^2 optimal model order reduction problem, i.e., the problem of approximating a higher order dynamical system by a lower order one so that a quadratic model reduction criterion is minimized, is of significant importance and is under intense study. Several earlier attempts to apply homotopy methods to the H^2 optimal model order reduction problem were not entirely satisfactory. Richter and Collins [28]–[30] devised a homotopy approach which only estimated certain crucial partial derivatives and employed relatively crude curve tracking techniques. Žigić, Bernstein, Collins, Richter, and Watson [41]–[43] formulated the problem so that numerical linear algebra techniques could be used to explicitly calculate partial derivatives, and employed sophisticated homotopy curve tracking algorithms, but the number of variables made large problems intractable. Several ways are proposed here to reduce the dimension of the homotopy map so that large problems are computationally feasible.

The problem can be formulated as: given the asymptotically stable, controllable, observable, time invariant, continuous time system

$$\begin{aligned} \dot{x}(t) &= A\,x(t) + B\,u(t), \\ y(t) &= C\,x(t), \end{aligned} \tag{1}$$

where $A \in \mathbf{R}^{n\times n}$, $B \in \mathbf{R}^{n\times m}$, $C \in \mathbf{R}^{l\times n}$, the goal is to find a reduced order model

$$\begin{aligned} \dot{x}_m(t) &= A_m\,x_m(t) + B_m\,u(t), \\ y_m(t) &= C_m\,x_m(t), \end{aligned} \tag{2}$$

where $A_m \in \mathbf{R}^{n_m\times n_m}$, $B_m \in \mathbf{R}^{n_m\times m}$, $C_m \in \mathbf{R}^{l\times n_m}$, $n_m < n$, which minimizes the cost function

$$J(A_m, B_m, C_m) \equiv \lim_{t\to\infty} \mathcal{E}\left[(y - y_m)^T R (y - y_m)\right], \tag{3}$$

where the input $u(t)$ is white noise with symmetric and positive definite intensity V and R is a symmetric and positive definite weighting matrix.

The optimal projection equations of Hyland and Bernstein [15], [16], described in Chapter 3, are basis independent and correspond to the maximum number of degrees of freedom. Richter and Collins [30] use this maximum number, and Žigić [41] reduced it somewhat. At the other extreme, the minimum number of degrees of freedom corresponds to the input normal form described in Chapter 2, and developed into a probability-one homotopy algorithm. Subtle differences between the optimal projection equations and input normal form formulations are explored in Chapter 3. Assuming a particular Jordan form for A_m leads to the minimal parameter formulation of Ly et al. [22], which is developed into a probability-one homotopy algorithm in Chapter 4. Chapter 5 gives numerical results for the input normal form and Ly form homotopies on the test set of Žigić [41].

Both the input normal form and Ly parameterization use the minimum possible number of degrees of freedom, but rely on assumptions about the structure of (A_m, B_m, C_m) that do not always hold, and therefore may not exist. Even worse, they may exist but be arbitrarily badly ill conditioned, resulting in unstable numerical algorithms. Chapter 6 explores an alternative formulation using more than the minimal number of degrees of freedom, and compares to the minimal formulations. Comparisons between the three formulations and the optimal projection equations approach are given in Chapter 7. A fundamental difference between the optimal projection equations and the other formulations is that the optimal projection equations approach solves $f(x) = 0$ where f is not the gradient of the cost functional and x is not the reduced order model, while the other three formulations solve $g(y) = 0$ where g is the gradient of the cost functional and y is the reduced order model.

1.2. The combined H^2/H^∞ model reduction problem.

In practice to simplify a plant for control design or to simplify a controller for ease of implementation, a H^∞ role must be taken into account, i.e., the order reduction approach should approximate the system frequency response to the greatest extent possible.

Several order reduction techniques have been proposed for approximating the frequency response of a given system. For example, frequency weighting has been studied in [9] in conjunction with balancing [23]. Moreover, Hankel norm reduction has been shown to have

fundamental ramifications for frequency domain approximation [1], [2], [12]. An overview and discussion of these ideas is given in [7].

The approach of [13], which is based upon a state space H^∞ formulation, is used here. In particular, by using a Riccati equation to enforce an H^∞ constraint on the norm of the reduction error in conjunction with an H^2 upper bound or entropy cost [27], it was shown in [13] that H^∞ constrained reduced order systems can be characterized by necessary conditions for optimality of the H^2 upper bound. The resulting algebraic conditions, which are a generalization of the "pure" H^2 optimality conditions given in [15], consist of nonstandard coupled Riccati and Lyapunov type matrix equations.

The purpose of the this work is to make significant progress in developing novel, stable, globally convergent numerical algorithms for solving the optimality conditions for H^2/H^∞ order reduction given in [13]. The approach we take is based on the construction of probability-one homotopy maps, similar to those developed for the H^2 order reduction problem in [11].

The problem is formulated as: given the controllable and observable, time invariant, continuous time system

$$\begin{aligned} \dot{x}(t) &= A\,x(t) + B\,Du(t), \\ y(t) &= C\,x(t), \end{aligned} \tag{4}$$

where $t \in [0, \infty)$, $A \in \mathbf{R}^{n\times n}$ is asymptotically stable, $B \in \mathbf{R}^{n\times m}$, $C \in \mathbf{R}^{l\times n}$, $D \in \mathbf{R}^{m\times p}$ $(m \leq p)$ and the input $Du(t)$ is white noise with symmetric and positive definite intensity $V \equiv DD^T$, find a n_m-th order model $(n_m < n)$

$$\begin{aligned} \dot{x}_m(t) &= A_m\,x_m(t) + B_m\,Du(t), \\ y_m(t) &= C_m\,x_m(t), \end{aligned} \tag{5}$$

where $A_m \in \mathbf{R}^{n_m\times n_m}$, $B_m \in \mathbf{R}^{n_m\times m}$, $C_m \in \mathbf{R}^{l\times n_m}$, which satisfies the following criteria:
(i) A_m is asymptotically stable;
(ii) the transfer function of the reduced order model lies within γ of the transfer function of the full order model in the H_∞ norm, i.e.,

$$\|H(s) - H_m(s)\|_\infty \leq \gamma \tag{6}$$

where

$$H(s) \equiv EC(sI_n - A)^{-1}BD, \qquad H_m(s) \equiv EC_m(sI_m - A_m)^{-1}B_mD,$$

$\gamma > 0$ is a given constant, $E \in \mathbf{R}^{q\times l}$ $(q \geq l)$ is a given constant matrix; and

(iii) the H^2 model reduction criterion

$$J(A_m, B_m, C_m) \equiv \lim_{t\to\infty} \mathcal{E}\left[(y - y_m)^T R(y - y_m)\right] \tag{7}$$

is minimized, where $\mathcal{E}$ is the expected value and $R = E^TE$ is a symmetric and positive definite weighting matrix.

1.4. The auxiliary minimization problem.

Define

$$\tilde{n} \equiv n + n_m, \qquad \tilde{E} \equiv E\tilde{C}, \qquad \tilde{D} \equiv \tilde{B}D,$$

$$\tilde{A} \equiv \begin{pmatrix} A & 0 \\ 0 & A_m \end{pmatrix}, \qquad \tilde{B} \equiv \begin{pmatrix} B \\ B_m \end{pmatrix}, \qquad \tilde{C} \equiv (C \quad -C_m), \tag{8}$$

$$\begin{aligned} \tilde{R} &\equiv \tilde{E}^T\tilde{E} = \tilde{C}^T R\tilde{C} = \begin{pmatrix} C^TRC & -C^TRC_m \\ -C_m^TRC & C_m^TRC_m \end{pmatrix}, \\ \tilde{V} &\equiv \tilde{D}\tilde{D}^T = \tilde{B}V\tilde{B}^T = \begin{pmatrix} BVB^T & BVB_m^T \\ B_mVB^T & B_mVB_m^T \end{pmatrix}. \end{aligned} \tag{9}$$

The full order system (4) and the reduced order system (5) can be written as a single augmented system

$$\begin{aligned} \dot{\tilde{x}}(t) &= \tilde{A}\,\tilde{x}(t) + \tilde{D}\,u(t), \\ \tilde{y}(t) &= \tilde{C}\,\tilde{x}(t). \end{aligned} \tag{10}$$

Using this notation the cost $J(A_m, B_m, C_m)$ can be written as

$$\begin{aligned} J(A_m, B_m, C_m) &= \lim_{t\to\infty} \mathcal{E}\left[(y - y_m)^T\,R\,(y - y_m)\right] = \lim_{t\to\infty} \mathcal{E}(\tilde{y}^T\,R\,\tilde{y}) \\ &= \lim_{t\to\infty} \mathcal{E}(\tilde{x}^T\,\tilde{C}^T\,R\,\tilde{C}\,\tilde{x}) = \lim_{t\to\infty} \mathcal{E}(\tilde{x}^T\,\tilde{R}\,\tilde{x}) = \mathrm{tr}\,(\tilde{Q}\,\tilde{R}), \end{aligned} \tag{11}$$

where $\tilde{Q}$ satisfies

$$\tilde{A}\,\tilde{Q} + \tilde{Q}\,\tilde{A}^T + \tilde{V} = 0. \tag{12}$$

LEMMA 1 [13]. *Let* (A_m, B_m, C_m) *be given and assume there exists* $\mathcal{Q} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ *satisfying*

$$\mathcal{Q} \quad \text{is symmetric and nonnegative definite} \tag{13}$$

and

$$\tilde{A}\,\mathcal{Q} + \mathcal{Q}\tilde{A}^T + \gamma^{-2}\mathcal{Q}\tilde{R}\mathcal{Q} + \tilde{V} = 0. \tag{14}$$

Then

$$(\tilde{A}, \tilde{D}) \quad \text{is stabilizable} \tag{15}$$

if and only if

$$A_m \quad \text{is asymptotically stable.}$$

Furthermore, if (15) holds, then

$$\|H(s) - H_m(s)\|_\infty \le \gamma, \tag{16}$$

$$\tilde{Q} \le \mathcal{Q} \qquad (\mathcal{Q} - \tilde{Q} \text{ is nonnegative definite}),$$

and

$$\text{tr } \tilde{Q}\tilde{R} \equiv J(A_m, B_m, C_m) \le \mathcal{J}(A_m, B_m, C_m) \equiv \text{ tr } \mathcal{Q}\tilde{R}.$$

Hence the H_∞ constraint is automatically enforced when a nonnegative definite solution to (14) is known to exist. Furthermore, the solution $\mathcal{Q}$ provides an upper bound for the actual state covariance $\tilde{Q}$ along with a bound on the H^2 model reduction.

The satisfaction of (13)–(15) leads to (i) A_m stable; (ii) a bound on the H_∞ distance between the full order and reduced order systems; and (iii) an upper bound for the H^2 model-reduction criterion. The auxiliary minimization problem is to determine (A_m, B_m, C_m) that minimizes $\mathcal{J}(A_m, B_m, C_m)$ and thus provides a bound for the actual H^2 criterion $J(A_m, B_m, C_m)$.

(A_m, B_m, C_m) is restricted to the set

$$\mathcal{S} \equiv \{(A_m, B_m, C_m) : \tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R} \text{ is asymptotically stable,}$$

$$\mathcal{Q} \text{ is symmetric positive definite,}$$

$$\text{and } (A_m, B_m, C_m) \text{ is controllable and observable } \}.$$

In Chapter 8 the input normal form homotopy approach for solving the auxiliary minimization problem is described. The Ly form and over-parametrization formulations are developed in Chapters 9 and 10. Numerical results and comparisons between different formulations are given in Chapter 11.

2. HOMOTOPY ALGORITHM BASED ON INPUT NORMAL FORM FORMULATION FOR THE H^2 OPTIMAL MODEL ORDER REDUCTION PROBLEM.

2.1. Input normal form formulations.

The following theorem is needed to present the homotopy method for the input normal form.

THEOREM 1 [17]. *Suppose $\bar{A}_m$ is asymptotically stable. Then for every minimal $(\bar{A}_m, \bar{B}_m, \bar{C}_m)$, i.e., $(\bar{A}_m, \bar{B}_m)$ is controllable and $(\bar{A}_m, \bar{C}_m)$ is observable, there exist a similarity transformation U and a positive definite matrix $\Omega = \operatorname{diag}(\omega_1, \cdots, \omega_{n_m})$ such that $A_m = U^{-1}\bar{A}_m U$, $B_m = U^{-1}\bar{B}_m$, and $C_m = \bar{C}_m U$ satisfy*

$$\begin{aligned} 0 &= A_m + A_m^T + B_m V B_m^T, \\ 0 &= A_m^T \Omega + \Omega A_m + C_m^T R C_m. \end{aligned} \tag{17}$$

In addition,

$$\begin{aligned} (A_m)_{ii} &= -\frac{1}{2}(B_m V B_m^T)_{ii}, \\ \omega_i &= \frac{(C_m^T R C_m)_{ii}}{(B_m V B_m^T)_{ii}}, \\ (A_m)_{ij} &= \frac{(C_m^T R C_m)_{ij} - \omega_j (B_m V B_m^T)_{ij}}{\omega_j - \omega_i}, \qquad \text{if } \omega_i \neq \omega_j. \end{aligned} \tag{18}$$

DEFINITION 1. The triple (A_m, B_m, C_m) satisfying (17) or (18) is said to be in *input normal form.*

Note that generically $\omega_i \neq \omega_j$ for $i \neq j$, and this is assumed henceforth. Under the assumption that a solution (A_m, B_m, C_m) in input normal form is sought, the only independent variables are B_m and C_m, and in this case the domain is

$$\{(A_m, B_m, C_m) : \ A_m \text{ is stable, } (A_m, B_m, C_m) \text{ is minimal and in input normal form}\}.$$

Assuming (A_m, B_m, C_m) is in input normal form, the cost function (3) can be written as

$$J(A_m, B_m, C_m) = \operatorname{tr}\ (\tilde{Q}\tilde{R}) \tag{19}$$

where $\tilde{Q}$ is a symmetric and positive definite matrix satisfying

$$\tilde{A}\tilde{Q} + \tilde{Q}\tilde{A}^T + \tilde{V} = 0, \tag{20}$$

and

$$\tilde{A} = \begin{pmatrix} A & 0 \\ 0 & A_m \end{pmatrix}, \qquad \tilde{R} = \begin{pmatrix} C^T R C & -C^T R C_m \\ -C_m^T R C & C_m^T R C_m \end{pmatrix}, \qquad \tilde{V} = \begin{pmatrix} BVB^T & BVB_m^T \\ B_m V B^T & B_m V B_m^T \end{pmatrix}. \tag{21}$$

$\tilde{Q}$ can be written as

$$\tilde{Q} = \begin{pmatrix} \tilde{Q}_1 & \tilde{Q}_{12} \\ \tilde{Q}_{12}^T & \tilde{Q}_2 \end{pmatrix}, \tag{22}$$

where $\tilde{Q}_1 \in \mathbf{R}^{n \times n}$, $\tilde{Q}_{12} \in \mathbf{R}^{n \times n_m}$, and $\tilde{Q}_2 \in \mathbf{R}^{n_m \times n_m}$.

The goal of minimizing (19) under the constraints (17) and (20) leads to the Lagrangian

$$\begin{aligned} L(A_m, B_m, C_m, \Omega, \tilde{Q}) = \operatorname{tr}\big[&\tilde{Q}\tilde{R} + (A_m + A_m^T + B_m V B_m^T) M_c \\ &+ (A_m^T \Omega + \Omega A_m + C_m^T R C_m) M_o + (\tilde{A}\tilde{Q} + \tilde{Q}\tilde{A}^T + \tilde{V})\tilde{P}\big], \end{aligned}$$

where the symmetric matrices M_o, M_c, and $\tilde{P}$ are Lagrange multipliers.

Setting $\partial L / \partial \tilde{Q} = 0$ gives

$$\tilde{A}^T \tilde{P} + \tilde{P}\tilde{A} + \tilde{R} = 0, \tag{23}$$

where $\tilde{P}$ is symmetric positive definite and can be partitioned as

$$\tilde{P} = \begin{pmatrix} \tilde{P}_1 & \tilde{P}_{12} \\ \tilde{P}_{12}^T & \tilde{P}_2 \end{pmatrix}. \tag{24}$$

$\partial L / \partial \Omega = 0$ and $\partial L / \partial A_m = 0$ yield

$$0 = 2M_c + 2\Omega M_o + 2(\tilde{P}_{12}^T \tilde{Q}_{12} + \tilde{P}_2 \tilde{Q}_2), \qquad 0 = (A_m M_o)_{ii}, \quad 1 \le i \le n_m.$$

A straightforward calculation shows

$$\begin{aligned} \frac{\partial L}{\partial B_m} &= 2(\tilde{P}_{12}^T B + \tilde{P}_2 B_m) V + 2 M_c B_m V, \\ \frac{\partial L}{\partial C_m} &= 2R(C_m \tilde{Q}_2 - C\tilde{Q}_{12}) + 2RC_m M_o. \end{aligned} \tag{25}$$

THEOREM 2 [8]. *The matrices M_c and M_o in (25) satisfy*

$$
\begin{aligned}
M_c &= -\left(\frac{1}{2}S + \Omega M_o\right), \\
\left(M_o\right)_{ii} &= -\frac{1}{\left(A_m\right)_{ii}} \sum_{\substack{j=1 \\ j \neq i}}^{n_m} \left(A_m\right)_{ij} \left(M_o\right)_{ji}, \\
\left(M_o\right)_{ij} &= \frac{(S)_{ij} - (S)_{ji}}{2(\omega_j - \omega_i)}, \quad \text{if } \omega_j \neq \omega_i,
\end{aligned}
\tag{26}
$$

where

$$
S = 2\left(\tilde{P}_{12}^T \tilde{Q}_{12} + \tilde{P}_2 \tilde{Q}_2\right). \tag{27}
$$

2.2. A homotopy approach based on the input normal form.

A homotopy approach based on the input normal form is now described. Let A_f, B_f, C_f , R_f, and V_f denote A, B, C, R, and V in the above and define

$$
\begin{aligned}
A(\lambda) &= A_0 + \lambda(A_f - A_0), \\
B(\lambda) &= B_0 + \lambda(B_f - B_0), \\
C(\lambda) &= C_0 + \lambda(C_f - C_0),
\end{aligned}
\qquad
\begin{aligned}
R(\lambda) &= R_0 + \lambda(R_f - R_0), \\
V(\lambda) &= V_0 + \lambda(V_f - V_0).
\end{aligned}
\tag{28}
$$

For brevity, $A(\lambda)$, $B(\lambda)$, $C(\lambda)$, $V(\lambda)$, and $R(\lambda)$ will be denoted by A, B, C, V, and R respectively in the following. Let

$$
\begin{aligned}
H_{B_m}(\theta, \lambda) &= \frac{\partial L}{\partial B_m} = 2\left(\tilde{P}_{12}^T B + \tilde{P}_2 B_m\right) V + 2 M_c B_m V, \\
H_{C_m}(\theta, \lambda) &= \frac{\partial L}{\partial C_m} = 2R\left(C_m \tilde{Q}_2 - C \tilde{Q}_{12}\right) + 2 R C_m M_o,
\end{aligned}
$$

where

$$
\theta \equiv \begin{pmatrix} \text{Vec } (B_m) \\ \text{Vec } (C_m) \end{pmatrix}
$$

denotes the independent variables B_m and C_m, M_o and M_c satisfy (26), and $\tilde{Q}$ and $\tilde{P}$ satisfy respectively (20) and (23) with partitioned forms (22) and (24). Vec(P) for a matrix $P \in \mathbf{R}^{p \times q}$ is the concatenation of its columns:

$$
\text{Vec}(P) \equiv \begin{pmatrix} P_{\cdot 1} \\ P_{\cdot 2} \\ \vdots \\ P_{\cdot q} \end{pmatrix} \in \mathbf{R}^{pq}.
$$

The homotopy map is defined as

$$\rho(\theta,\lambda) = \begin{pmatrix} \text{Vec}\,\big[H_{B_m}(\theta,\lambda)\big] \\ \text{Vec}\,\big[H_{C_m}(\theta,\lambda)\big] \end{pmatrix}, \tag{29}$$

and its Jacobian matrix is

$$D\rho(\theta,\lambda) = \big(D_\theta\rho(\theta,\lambda), D_\lambda\rho(\theta,\lambda)\big). \tag{30}$$

Define

$$\begin{aligned} \hat{H}_{B_m}\big(\tilde{P}^{(j)}, M_c^{(j)}\big) &= 2\big(\tilde{P}_{12}^{T(j)}B + \tilde{P}_2^{(j)}B_m\big)V + 2M_c^{(j)}B_mV, \\ \hat{H}_{C_m}\big(\tilde{Q}^{(j)}, M_o^{(j)}\big) &= 2R\big(C_m\tilde{Q}_2^{(j)} - C\tilde{Q}_{12}^{(j)}\big) + 2RC_mM_o^{(j)}, \end{aligned}$$

where the superscript (j) means $\partial/\partial\theta_j$: $Y^{(j)} \equiv \frac{\partial Y}{\partial\theta_j}$. Using the above definitions, we have for $\theta_j = \big(B_m\big)_{kl}$,

$$\begin{aligned} \frac{\partial H_{B_m}}{\partial(B_m)_{kl}} &= \hat{H}_{B_m}\big(\tilde{P}^{(j)}, M_c^{(j)}\big) + 2\big(\tilde{P}_2 + M_c\big)E^{(k,l)}V, \\ \frac{\partial H_{C_m}}{\partial(B_m)_{kl}} &= \hat{H}_{C_m}\big(\tilde{Q}^{(j)}, M_o^{(j)}\big), \end{aligned} \tag{31}$$

and for $\theta_j = \big(C_m\big)_{kl}$,

$$\begin{aligned} \frac{\partial H_{B_m}}{\partial(C_m)_{kl}} &= \hat{H}_{B_m}\big(\tilde{P}^{(j)}, M_c^{(j)}\big), \\ \frac{\partial H_{C_m}}{\partial(C_m)_{kl}} &= \hat{H}_{C_m}\big(\tilde{Q}^{(j)}, M_o^{(j)}\big) + 2RE^{(k,l)}\big(\tilde{Q}_2 + M_o\big), \end{aligned} \tag{32}$$

where $E^{(k,l)}$ is a matrix of the appropriate dimension whose only nonzero element is $e_{kl} = 1$. $\tilde{P}^{(j)}$ and $\tilde{Q}^{(j)}$ can be obtained by solving the Lyapunov equations

$$\begin{aligned} 0 &= \tilde{A}^{(j)}\tilde{Q} + \tilde{A}\tilde{Q}^{(j)} + \tilde{Q}^{(j)}\tilde{A}^T + \tilde{Q}\tilde{A}^{T(j)} + \tilde{V}^{(j)}, \\ 0 &= \tilde{A}^{T(j)}\tilde{P} + \tilde{A}^T\tilde{P}^{(j)} + \tilde{P}^{(j)}\tilde{A} + \tilde{P}\tilde{A}^{(j)} + \tilde{R}^{(j)}. \end{aligned} \tag{33}$$

Similarly for λ, using a dot to denote $\partial/\partial\lambda$,

$$\begin{aligned} \frac{\partial H_{B_m}}{\partial\lambda} &= \hat{H}_{B_m}\big(\dot{\tilde{P}}, \dot{M}_c\big) + 2\tilde{P}_{12}^T\big(\dot{B}V + B\dot{V}\big) + 2\big(\tilde{P}_2 + M_c\big)B_m\dot{V}, \\ \frac{\partial H_{C_m}}{\partial\lambda} &= \hat{H}_{C_m}\big(\dot{\tilde{Q}}, \dot{M}_o\big) + 2\dot{R}C_m\big(\tilde{Q}_2 + M_o\big) - 2\big(\dot{R}C + R\dot{C}\big)\tilde{Q}_{12}, \end{aligned} \tag{34}$$

where $\dot{\tilde{P}}$ and $\dot{\tilde{Q}}$ are obtained by solving the Lyapunov equations

$$0 = \dot{\tilde{A}}\tilde{Q} + \tilde{A}\dot{\tilde{Q}} + \dot{\tilde{Q}}\tilde{A}^T + \tilde{Q}\dot{\tilde{A}}^T + \dot{\tilde{V}},$$

$$0 = \dot{\tilde{A}}^T\tilde{P} + \tilde{A}^T\dot{\tilde{P}} + \dot{\tilde{P}}\tilde{A} + \tilde{P}\dot{\tilde{A}} + \dot{\tilde{R}}.$$

2.3. Numerical algorithm for input normal form homotopy.

The initial point $(\theta, \lambda) = (\theta_0, 0) = \big((B_m)_0, (C_m)_0, 0\big)$ is chosen so that the triple $\big((A_m)_0, (B_m)_0, (C_m)_0\big)$ is in input normal form and satisfies $\rho(\theta_0, 0) = 0$.

THEOREM 3 [23]. *Suppose $\bar{A}$ is asymptotically stable. Then for every minimal $(\bar{A}, \bar{B}, \bar{C})$, i.e., $\big(\bar{A}, \bar{B}\big)$ is controllable and $\big(\bar{A}, \bar{C}\big)$ is observable, there exist a similarity transformation T and a positive definite matrix $\Lambda = \operatorname{diag}\big(d_1, d_2, \cdots, d_n\big)$ with $d_i \geq d_{i+1}$ such that $A = T^{-1}\bar{A}T$, $B = T^{-1}\bar{B}$, and $C = \bar{C}T$ satisfy*

$$0 = A\Lambda + \Lambda A^T + BVB^T,$$

$$0 = A^T\Lambda + \Lambda A + C^TRC.$$

DEFINITION 2. The triple $\big(A, B, C\big)$ in the above theorem is *balanced.*

According to Moore [23], under certain conditions, the leading principal $n_m \times n_m$ block of A, the leading principal $n_m \times m$ block of B, and the leading principal $l \times n_m$ block of C in balanced form are good approximations to the reduced order model. This suggests that the initial point $(\theta_0, 0)$ be chosen as follows:

1) Transform the given triple (A_f, B_f, C_f) to balanced form (A_b, B_b, C_b).

2) Partition (A_b, B_b, C_b) as

$$A_b = \begin{matrix} \\ n_m \{ \end{matrix} \overbrace{\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}}^{n_m}, \quad B_b = \begin{matrix} \\ n_m \{ \end{matrix} \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}, \quad C_b = \overbrace{\begin{pmatrix} C_1 & C_2 \end{pmatrix}}^{n_m}.$$

3) (A_0, B_0, C_0) is chosen as

$$A_0 = \begin{pmatrix} A_{11} & 0 \\ 0 & A_{22} \end{pmatrix}, \quad B_0 = \begin{pmatrix} B_1 \\ 0 \end{pmatrix}, \quad C_0 = \begin{pmatrix} C_1 & 0 \end{pmatrix}.$$

4) The initial point for the reduced order model is chosen as

$$\bar{\theta}_0 = \begin{pmatrix} \text{Vec } (\bar{B}_m)_0 \\ \text{Vec } (\bar{C}_m)_0 \end{pmatrix} = \begin{pmatrix} \text{Vec } B_1 \\ \text{Vec } C_1 \end{pmatrix},$$

and $(\bar{A}_m)_0 = A_{11}$ by construction.

5) Transform the initial point $\big((\bar{A}_m)_0, (\bar{B}_m)_0, (\bar{C}_m)_0\big)$ to input normal form so that the initial reduced order model is

$$\big((A_m)_0, (B_m)_0, (C_m)_0\big) = \big(T^{-1}\,(\bar{A}_m)_0\,T, \quad T^{-1}\,(\bar{B}_m)_0, \quad (\bar{C}_m)_0\,T\big).$$

The initial point for the homotopy map is then $(\theta_0, 0)$, where

$$\theta_0 = \begin{pmatrix} \text{Vec } (B_m)_0 \\ \text{Vec } (C_m)_0 \end{pmatrix}.$$

(In general, the truncation to obtain the approximate reduced order model should be based on the component costs instead of on the sizes of the balanced gains d_i as done above [32]. This explains why in some cases (Examples 1 and 6) the above algorithm for choosing the initial points did not lead to a reduced order model with a minimal cost.)

Once the initial point is chosen, the rest of the computation is as follows:

1) Set $\lambda := 0$, $\theta := \theta_0$.
2) Calculate A_m from (18), $\tilde{R}$, $\tilde{V}$, and compute $\tilde{Q}$ and $\tilde{P}$ according to (20) and (23).
3) Evaluate S from (27) and M_o and M_c according to (26).
4) Evaluate the homotopy map $\rho(\theta, \lambda)$ in (29) and $D\rho(\theta, \lambda)$ in (30).
5) Predict the next point $Z^{(0)} = \big(\theta^{(0)}, \lambda^{(0)}\big)$ on the curve γ.
6) For $k := 0, 1, 2, \cdots$ until convergence do

$$Z^{(k+1)} = \big[D\rho(Z^{(k)})\big]^{\dagger} \rho\big(Z^{(k)}\big),$$

where $\big[D\rho(Z)\big]^{\dagger}$ is the Moore-Penrose inverse of $D\rho(Z)$. Let $\big(\theta_1, \lambda_1\big) = \lim_{k \to \infty} Z^{(k)}$.

7) If $\lambda_1 < 1$, then set $\theta := \theta_1$, $\lambda := \lambda_1$, and go to step 2).
8) If $\lambda_1 \geq 1$, compute the solution $\bar{\theta}$ at $\lambda = 1$. A_m is then obtained from (18).

An alternative strategy for choosing an initial point is as follows:

1) Modify A_f to $A'_f = c_1 I + c_2 A_f$, where $c_1 \leq 0$ and $c_2 \geq 0$.

1) Transform (A'_f, B_f, C_f) to balanced form and choose (A'_0, B'_0, C'_0) as before.

3) Compute the initial reduced order model $\big((A_m)_0, (B_m)_0, (C_m)_0\big)$ from the triple (A'_0, B'_0, C'_0) as before.

When $c_1 = 0$, $c_2 = 1$, this strategy reduces to the previous one. For some problems, our numerical experiments show that HOMPACK reaches $\lambda > 1$ in fewer steps with $c_1 \neq 0$ than with $c_1 = 0$. A modification to the homotopy map $\rho(\theta, \lambda)$ in (29) is

$$\rho_1(\theta, \lambda) = \lambda\rho(\theta, \lambda) + (1 - \lambda)(\theta - \theta_0),$$

where θ_0 denotes the initial value of θ at $\lambda = 0$. For some problems this homotopy map can be more efficient than the one in (29), while in other cases it can be less efficient.

3. COMPARISON WITH OPTIMAL PROJECTION EQUATIONS APPROACH.

THEOREM 4 [15] [16]. *Suppose* (A_m, B_m, C_m) *is a controllable and observable solution of the problem* (1)–(3). *Then there exist positive semidefinite* pseudogramians $\hat{Q}$, $\hat{P}$ *that are a solution to* modified Lyapunov equations

$$\begin{aligned} 0 &= \tau[A\,\hat{Q} + \hat{Q}\,A^T + B\,V\,B^T], \\ 0 &= [A^T\,\hat{P} + \hat{P}\,A + C^T\,R\,C]\,\tau, \end{aligned} \tag{35}$$

and satisfy rank conditions

$$rank\,(\hat{Q}) = rank\,(\hat{P}) = rank\,(\hat{Q}\,\hat{P}) = n_m,$$

such that the optimal model is given by

$$\begin{aligned} A_m &= \Gamma\,A\,G^T, \\ B_m &= \Gamma\,B, \\ C_m &= C\,G^T, \end{aligned} \tag{36}$$

where G *and* Γ *come from a* (G, M, Γ)-factorization *of* $\hat{Q}\hat{P}$*:*

$$\begin{aligned} \hat{Q}\,\hat{P} &= G^T\,M\,\Gamma, \\ \Gamma\,G^T &= I_{n_m}, \end{aligned} \tag{37}$$

$G, \Gamma \in \mathbf{R}^{n_m \times n}$, $M \in \mathbf{R}^{n_m \times n_m}$ *is positive semisimple and* $\tau \equiv G^T\,\Gamma$.

Equations (35) are called the optimal projection equations, which after the nontrivial algebraic manipulation described in [42], can be written in a form suitable for computation as

$$\begin{aligned} U_1\,A\,W_1\Sigma\,W_1^T + \Sigma\,W_1^T\,A^T + U_1\,B\,V\,B^T &= 0, && (n_m\ n) \\ A^T\,U_1^T\,\Sigma + U_1^T\,\Sigma\,U_1\,A\,W_1 + C^T\,R\,C\,W_1 &= 0, && (n\ n_m) \\ U_1\,W_1 - I &= 0. && (n_m^2) \end{aligned} \tag{38}$$

The unknowns are $W_1 \in \mathbf{R}^{n \times n_m}$, $U_1 \in \mathbf{R}^{n_m \times n}$ and symmetric $\Sigma \in \mathbf{R}^{n_m \times n_m}$. In terms of these new unknowns, $\hat{Q}$ and $\hat{P}$ in (37) can be written as

$$\hat{Q} = W_1 \Sigma W_1^T, \qquad \hat{P} = U_1^T \Sigma U_1.$$

Hyland and Bernstein [16] stated that the optimal projection equations can have at most $\binom{n}{n_m}$ solutions. It is shown by the following 2-dimensional example that this is not true in general.

The system [18] is given by

$$A = \begin{pmatrix} -0.05 & -0.99 \\ -0.99 & -5000.0 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 100 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 100 \end{pmatrix}. \tag{39}$$

PROPOSITION: *For the system* (1) *defined by* (39), *the solution set of the optimal projection equations contains three isolated solutions and a one-dimensional manifold parameterized by one element of either* W_1 *or* U_1.

Proof. The three isolated solutions are

$$A_m = (-0.005004234), \quad B_m = (1.000213), \quad C_m = (1.000213),$$

$$A_m = (-4998.079), \quad B_m = (100.0002), \quad C_m = (100.0002),$$

$$A_m = (-0.4659163), \quad B_m = (-1.940482), \quad C_m = (-1.940482),$$

which were obtained by both POLSYS from HOMPACK [38] and by a homotopy approach [41]–[43]. The one-dimensional manifold of solutions corresponds to

$$A_m = (-0.4851515), \quad B_m = (0.0), \quad C_m = (0.0),$$

which can be derived directly from the optimal projection equations as follows.

Let $W_1 = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, $U_1 = (x_3, x_4)$, and $\Sigma = x_5$. The optimal projection equations (38) for this problem can be written as

$$\begin{aligned}
0 &= a_{11}x_1^2x_3x_5 + a_{12}x_1x_2x_3x_5 + a_{21}x_1^2x_4x_5 + a_{22}x_1x_2x_4x_5 \\
&\quad + a_{11}x_1x_5 + a_{12}x_2x_5 + (BVB^T)_{11}x_3 + (BVB^T)_{21}x_4, \\
0 &= a_{11}x_1x_2x_3x_5 + a_{12}x_2^2x_3x_5 + a_{21}x_1x_2x_4x_5 + a_{22}x_2^2x_4x_5 \\
&\quad + a_{21}x_1x_5 + a_{22}x_2x_5 + (BVB^T)_{12}x_3 + (BVB^T)_{22}x_4, \\
0 &= a_{11}x_1x_3^2x_5 + a_{12}x_2x_3^2x_5 + a_{21}x_1x_3x_4x_5 + a_{22}x_2x_3x_4x_5 \\
&\quad + a_{11}x_3x_5 + a_{21}x_4x_5 + (C^TRC)_{11}x_1 + (C^TRC)_{12}x_2, \\
0 &= a_{11}x_1x_3x_4x_5 + a_{12}x_2x_3x_4x_5 + a_{21}x_1x_4^2x_5 + a_{22}x_2x_4^2x_5 \\
&\quad + a_{12}x_3x_5 + a_{22}x_4x_5 + (C^TRC)_{21}x_1 + (C^TRC)_{22}x_2, \\
0 &= x_1x_3 + x_2x_4 - 1.
\end{aligned} \tag{40}$$

The triple (A_m, B_m, C_m) is given by

$$\begin{aligned}
A_m &= \Gamma A G^T = (x_3\ x_4)\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\
&= x_1(a_{11}x_3 + a_{21}x_4) + x_2(a_{12}x_3 + a_{22}x_4), \\
B_m &= \Gamma B = (x_3\ x_4)\begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix} = b_{11}x_3 + b_{21}x_4, \\
C_m &= CG^T = (c_{11}\ c_{12})\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = c_{11}x_1 + c_{12}x_2,
\end{aligned} \tag{41}$$

where $\Gamma = U_1$ and $G = W_1^T$. Substituting (39) into (40) and (41), setting $B_m = x_3 + 100x_4 = 0$ and $C_m = x_1 + 100x_2 = 0$ gives $x_1 = -100x_2, x_3 = -100x_4$, and $A_m = -4852x_2x_4$. Equations (40) become

$$0 = 485200x_2^2x_4x_5 - 0.49x_2x_5, \tag{42}$$

$$0 = 485200x_2x_4^2x_5 - 0.49x_4x_5, \tag{43}$$

$$0 = 4852x_2^2x_4x_5 + 4901x_2x_5, \tag{44}$$

$$0 = 4852x_2x_4^2x_5 + 4901x_4x_5, \tag{45}$$

$$0 = 10001x_2x_4 - 1. \tag{46}$$

If $x_2 = 0$ or $x_4 = 0$, equation (46) will not be satisfied. Only the situation that $x_2 \neq 0$ and $x_4 \neq 0$ is possible. Then equations (42)–(46) can be reduced to

$$\begin{aligned} 0 &= 485200x_2x_4x_5 - 0.49x_5, \\ 0 &= 4852x_2x_4x_5 + 4901x_5, \\ 0 &= 10001x_2x_4 - 1. \end{aligned} \tag{47}$$

If $x_5 \neq 0$ then (34) becomes

$$\begin{aligned} 0 &= 485200x_2x_4 - 0.49, \\ 0 &= 4852x_2x_4 + 4901, \\ 0 &= 10001x_2x_4 - 1, \end{aligned} \tag{48}$$

which does not have a solution.

Thus $x_5 = 0$, and equation (47) reduces to

$$10001x_2x_4 - 1 = 0,$$

which gives $A_m = -4852/10001 = -0.4851515$ corresponding to a one-dimensional manifold parametrized by x_2 or x_4. Hence the solution $A_m = -0.4851515$, $B_m = 0$ and $C_m = 0$ (which is not controllable or observable) corresponds to a one-dimensional manifold of solutions of the optimal projection equations. Q. E. D.

The set of solutions of the input normal form equations contains the same set of isolated solutions as the optimal projection equations, and also a fourth isolated solution given by $A_m = B_m = C_m = 0$. Therefore the solution sets of the two formulations are different.

The input normal form equations can be rewritten as

$$\begin{aligned} 0 &= 2\big(\tilde{P}_{12}^T B + \tilde{P}_2 B_m\big)V + 2M_c B_m V, \\ 0 &= 2R\big(C_m \tilde{Q}_2 - C\tilde{Q}_{12}\big) + 2RC_m M_o. \end{aligned} \tag{49}$$

Setting $B_m = C_m = 0$, the equations become

$$\begin{aligned} 0 &= \tilde{P}_{12}^T BV, \\ 0 &= RC\tilde{Q}_{12}, \end{aligned} \tag{50}$$

where $\tilde{P}_{12}$ and $\tilde{Q}_{12}$ satisfy respectively

$$0 = A^T \tilde{P}_{12} + \tilde{P}_{12} A_m,$$

$$0 = A\tilde{Q}_{12} + \tilde{Q}_{12} A_m,$$

which has a solution $\tilde{P}_{12} = \tilde{Q}_{12} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$. A_m satisfies

$$A_m + A_m^T + B_m V B_m^T = A_m + A_m^T = 0$$

which gives $A_m = 0$.

It should be noted that the solutions to the optimal projection equations (35) that satisfy the rank conditions $\text{rank}(\hat{Q}) = \text{rank}(\hat{P}) = \text{rank}(\hat{Q}\,\hat{P}) = n_m$ characterize all controllable and observable extremals of the optimal model reduction problem. However, there are algebraic solutions to (35) that do not satisfy these rank conditions. The one-dimensional manifold of solutions of the previous proposition are such a set of solutions since for these solutions $\text{rank}(\hat{Q}) = \text{rank}(\hat{P}) = 0 \neq n_m = 1$. On the other hand, the input normal form equations characterize all extremals of the optimal model reduction problem for which the input normal form has the property that no two diagonal elements of Ω are equal. No restriction is placed on the controllability or observability of these extremals. Hence, the extremal sets that the optimal projection equations and the input normal form equations characterize are not identical. In addition, the optimal projection equations may also have algebraic solutions that characterize additional reduced-order models that are uncontrollable or unobservable and may or may not be related to the solutions of the input normal form equations by a similarity transformation. These differences in the solution sets were illustrated by the example of this section. However, it should be noted that if one considers their input-output properties, the two solution sets are equivalent.

4. HOMOTOPY ALGORITHM BASED ON LY FORMULATION FOR THE H^2 MODEL ORDER REDUCTION PROBLEM.

4.1 Ly's formulation.

Ly et al. [22] introduced another canonical form also with $n_m m + n_m l$ parameters as in the input normal form formulation. The reduced order model is represented with respect to a basis such that A_m is a 2×2 block-diagonal matrix (2×2 blocks with an additional 1×1 block if n_m is odd) with 2×2 blocks in the form

$$\begin{pmatrix} 0 & 1 \\ * & * \end{pmatrix},$$

B_m is a full matrix, and

$$C_m = \begin{pmatrix} (C_m)_1 & (C_m)_2 & \cdots & (C_m)_r \end{pmatrix}$$

where

$$(C_m)_i = \begin{pmatrix} 1 & * & \cdots & * \\ 0 & * & \cdots & * \end{pmatrix}^T,$$

$$(C_m)_r = \begin{pmatrix} 1 & * & \cdots & * \end{pmatrix}^T, \quad \text{if } n_m \text{ is odd.}$$

Let $\mathcal{S}$ be the set of indices of those elements of A_m which are parameters, i.e.,

$$\mathcal{S} \equiv \{(2,1),(2,2),\ldots,(n_m,n_m)\}.$$

To find the minimum of the cost function (19), consider the Lagrangian

$$L(A_m, B_m, C_m, \tilde{Q}) = \operatorname{tr}\big[\tilde{Q}\tilde{R} + \big(\tilde{A}\tilde{Q} + \tilde{Q}\tilde{A}^T + \tilde{V}\big)\tilde{P}\big], \tag{51}$$

where the symmetric matrix $\tilde{P}$ is a Lagrange multiplier, $\tilde{Q}$ satisfies (20), and $\tilde{A}$, $\tilde{R}$, and $\tilde{V}$ are defined in (21). Setting $\partial L/\partial \tilde{Q} = 0$ gives (23), and $\tilde{P}$ is symmetric positive definite and can be partitioned as in (24). A straightforward calculation shows

$$\begin{aligned} \frac{\partial L}{\partial (A_m)_{ij}} &= 2\big(\tilde{P}_{12}^T \tilde{Q}_{12} + \tilde{P}_2 \tilde{Q}_2\big)_{ij}, \quad (i,j) \in \mathcal{S}, \\ \frac{\partial L}{\partial B_m} &= 2\big(\tilde{P}_{12}^T B + \tilde{P}_2 B_m\big)V, \\ \frac{\partial L}{\partial (C_m)_{ij}} &= 2\frac{\partial}{\partial (C_m)_{ij}}\big[\ \operatorname{tr}\ \big(-Q_{12}^T C^T R C_m\big) + \ \operatorname{tr}\ \big(Q_2 C_m^T R C_m\big)\big] \\ &= 2R\big(C_m \tilde{Q}_2 - C\tilde{Q}_{12}\big)_{ij}, \quad i > 1. \end{aligned} \tag{52}$$

4.2. A homotopy approach based on the Ly formulation.

Let A_f, B_f, C_f , R_f, and V_f denote A, B, C, R, and V in the above and define $A(\lambda)$, $B(\lambda)$, $C(\lambda)$, $R(\lambda)$, and $V(\lambda)$ as in (28) and denote them by A, B, C, V, and R respectively in the following. Let

$$\begin{aligned} H_{A_m}(\theta,\lambda) &= \frac{\partial L}{\partial A_m} = 2\big(\tilde{P}_{12}^T\tilde{Q}_{12} + \tilde{P}_2\tilde{Q}_2\big), \\ H_{B_m}(\theta,\lambda) &= \frac{\partial L}{\partial B_m} = 2\big(\tilde{P}_{12}^T B + \tilde{P}_2 B_m\big)V, \\ H_{C_m}(\theta,\lambda) &= \frac{\partial L}{\partial C_m} = 2R\big(C_m\tilde{Q}_2 - C\tilde{Q}_{12}\big), \end{aligned} \tag{53}$$

where in H_{A_m} only those elements corresponding to the parameter elements of A_m are nonzero and

$$\theta \equiv \begin{pmatrix} (A_m)_{\mathcal{S}} \\ \text{Vec } (B_m) \\ \text{Vec } (C_m)_{\mathcal{T}.} \end{pmatrix} \tag{54}$$

denotes the independent variables, $\tilde{Q}$ and $\tilde{P}$ satisfy respectively (20) and (23), $(A_m)_{\mathcal{S}}$ is a vector consisting of those elements in A_m with indices in the set $\mathcal{S}$, i.e.,

$$(A_m)_{\mathcal{S}} = \big((A_m)_{21}, (A_m)_{22}, \cdots, (A_m)_{n_m n_m}\big)^T,$$

$(C_m)_{\mathcal{T}.}$ is the matrix obtained from rows $\mathcal{T} = \{2, \ldots, l\}$ of C_m.

The homotopy map is defined as

$$\rho(\theta,\lambda) = \begin{pmatrix} \big[H_{A_m}(\theta,\lambda)\big]_{\mathcal{S}} \\ \text{Vec } \big[H_{B_m}(\theta,\lambda)\big] \\ \text{Vec } \big[H_{C_m}(\theta,\lambda)\big]_{\mathcal{T}.} \end{pmatrix}, \tag{55}$$

and its Jacobian matrix is

$$D\rho(\theta,\lambda) = \big(D_\theta\rho(\theta,\lambda), D_\lambda\rho(\theta,\lambda)\big).$$

Define

$$\begin{aligned} \hat{H}_{A_m}\big(\tilde{P}^{(j)}, \tilde{Q}^{(j)}\big) &= 2\big(\tilde{P}_{12}^{T(j)}\tilde{Q}_{12} + \tilde{P}_{12}^T\tilde{Q}_{12}^{(j)} + \tilde{P}_2^{(j)}\tilde{Q}_2 + \tilde{P}_2\tilde{Q}_2^{(j)}\big), \\ \hat{H}_{B_m}\big(\tilde{P}^{(j)}\big) &= 2\big(\tilde{P}_{12}^{T(j)}B + \tilde{P}_2^{(j)}B_m\big)V, \\ \hat{H}_{C_m}(\tilde{Q}^{(j)}) &= 2R\big(C_m\tilde{Q}_2^{(j)} - C\tilde{Q}_{12}^{(j)}\big), \end{aligned} \tag{56}$$

where the superscript (j) means $\partial/\partial\theta_j$. Using the above definitions, we have for $\theta_j = (A_m)_{kl}$, where $(k,l) \in \mathcal{S}$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (A_m)_{kl}} &= \hat{H}_{A_m}\big(\tilde{P}^{(j)}, \tilde{Q}^{(j)}\big), \\
\frac{\partial H_{B_m}}{\partial (A_m)_{kl}} &= \hat{H}_{B_m}\big(\tilde{P}^{(j)}\big), \\
\frac{\partial H_{C_m}}{\partial (A_m)_{kl}} &= \hat{H}_{C_m}\big(\tilde{Q}^{(j)}\big),
\end{aligned} \tag{57}$$

for $\theta_j = (B_m)_{kl}$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (B_m)_{kl}} &= \hat{H}_{A_m}\big(\tilde{P}^{(j)}, \tilde{Q}^{(j)}\big), \\
\frac{\partial H_{B_m}}{\partial (B_m)_{kl}} &= \hat{H}_{B_m}\big(\tilde{P}^{(j)}\big) + 2\tilde{P}_2 E^{(k,l)} V, \\
\frac{\partial H_{C_m}}{\partial (B_m)_{kl}} &= \hat{H}_{C_m}\big(\tilde{Q}^{(j)}\big),
\end{aligned} \tag{58}$$

and for $\theta_j = (C_m)_{kl}$, where $k > 1$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (C_m)_{kl}} &= \hat{H}_{A_m}\big(\tilde{P}^{(j)}, \tilde{Q}^{(j)}\big), \\
\frac{\partial H_{B_m}}{\partial (C_m)_{kl}} &= \hat{H}_{B_m}\big(\tilde{P}^{(j)}\big), \\
\frac{\partial H_{C_m}}{\partial (C_m)_{kl}} &= \hat{H}_{C_m}\big(\tilde{Q}^{(j)}\big) + 2RE^{(k,l)}\tilde{Q}_2.
\end{aligned} \tag{59}$$

$\tilde{P}^{(j)}$ and $\tilde{Q}^{(j)}$ can be obtained by solving the Lyapunov equation (33). The derivative of the homotopy map with respect to λ can be derived in a similar fashion.

4.3. Numerical algorithm for Ly form homotopy.

The initial point $(\theta, \lambda) = (\theta_0, 0)$ is chosen so that the triple $\big((A_m)_0, (B_m)_0, (C_m)_0\big)$ is in Ly's form and satisfies $\rho(\theta_0, 0) = 0$. This can be done as follows:

1) Obtain the initial reduced order model $\big((A_m)_0, (B_m)_0, (C_m)_0\big)_b$ in balanced form in the same way as for the input normal form approach.
2) Transform the balanced $\big((A_m)_0, (B_m)_0, (C_m)_0\big)_b$ to Ly's form, and build θ_0 as described in (54).

The homotopy curve tracking computation is the same as described in Chapter 2.

5. NUMERICAL RESULTS.

In this section numerical results for both the input normal form and Ly formulations are given for eleven systems. The first nine systems have been studied and solved in [41]–[43] using the optimal projection equations approach. Comparisons are made between these two minimal formulations and the optimal projection equations in Chapter 7.

The cost J is computed for each model as tr $(\tilde{Q}\,\tilde{R})$, according to (19). For all examples $V = R = I$. Unless indicated otherwise, the solutions, given in input normal form, can be obtained by both formulations and are the same as those obtained by the optimal projection equations method.

EXAMPLE 1 [18]. The system is given by

$$A = \begin{pmatrix} -0.05 & -0.99 \\ -0.99 & -5000.0 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 100 \end{pmatrix}, \quad C = (\,1 \quad 100\,).$$

The homotopy algorithm converges to a solution corresponding to the model of order $n_m = 1$ given by

$$A_m = (\,-0.00500423\,), \quad B_m = (\,-0.100042\,), \quad C_m = (\,-10.0000\,),$$

which was not obtained by the optimal projection equation approach of [41]–[43]. This model yields the cost $J = 10000$.

In the first step of choosing an initial point, (A_f, B_f, C_f) is transformed to (A_b, B_b, C_b), where orthogonal decompositions of two matrices are needed. If the eigenvalues of one of the matrices are rearranged in ascending order, then a different solution is obtained, namely

$$A_m = (\,-4998.08\,), \quad B_m = (\,-99.9808\,), \quad C_m = (\,-100.020\,).$$

This model yields the (minimum) cost $J = 96.0781$.

EXAMPLE 2 [33]. The system is given by

$$A = \begin{pmatrix} -1 & 0 \\ 0 & -10 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 1 \\ 70 & 1 \end{pmatrix}, \quad C = (\,1 \quad -0.2\,).$$

A model of order $n_m = 1$ is

$$A_m = (-11.9794), \quad B_m = (-4.85914 \quad 0.589656), \quad C_m = (2.76076).$$

This model yields the cost $J = 0.598377$.

EXAMPLE 3 [18]. The system is given by

$$A = \begin{pmatrix} -0.25 & -0.4 \\ -0.4 & -0.72 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 1.2 \end{pmatrix}, \quad C = (1 \quad 1.2).$$

A model of order $n_m = 1$ is

$$A_m = (-0.838521), \quad B_m = (-1.29501), \quad C_m = (-1.82558).$$

This model yields the cost $J = 0.107256$.

EXAMPLE 4 [34]. The system is given by

$$A = \begin{pmatrix} -1 & 3 & 0 \\ -1 & -1 & 1 \\ 4 & -5 & -4 \end{pmatrix}, \quad B = \begin{pmatrix} -2 \\ 2 \\ 4 \end{pmatrix}, \quad C = (1 \quad 0 \quad 0).$$

A model of order $n_m = 1$ is

$$A_m = (-0.286334), \quad B_m = (0.756748), \quad C_m = (0.878161),$$

which is different from that obtained by the optimal projection equation method [41]-[43], and has a smaller cost $J = 1.22883$. A model of order $n_m = 2$ is

$$A_m = \begin{pmatrix} -0.215037 & 0.753968 \\ -2.51385 & -3.60074 \end{pmatrix}, \quad B_m = \begin{pmatrix} 0.655800 \\ 2.68356 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} 0.888877 \\ -1.09093 \end{pmatrix}.$$

This model yields the cost $J = 0.0197781$.

EXAMPLE 5 [18]. The system is given by

$$A = \begin{pmatrix} -10 & 1 & 0 \\ -5 & 0 & 1 \\ -1 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \quad C = (1 \quad 0 \quad 0).$$

A model of order $n_m = 1$ is

$$A_m = (-0.157898), \quad B_m = (0.561956), \quad C_m = (0.318537).$$

This model yields the cost $J = 0.0107792$. A model of order $n_m = 2$ is

$$A_m = \begin{pmatrix} -0.139652 & 0.100607 \\ -0.600971 & -0.448192 \end{pmatrix}, \quad B_m = \begin{pmatrix} 0.528492 \\ 0.946775 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} 0.320438 \\ -0.0961019 \end{pmatrix}.$$

This model yields the cost $J = 0.000329024$.

EXAMPLE 6 [41]. The system is given by

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -2 & -0.02 & 1 & 0.01 \\ 0 & 0 & 0 & 1 \\ 0.1 & 0.001 & -0.1 & -0.001 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad C = (0 \quad 1 \quad 0 \quad 0).$$

A model of order $n_m = 1$ is

$$A_m = (-0.353743), \quad B_m = \begin{pmatrix} -0.184397 \\ -0.820660 \end{pmatrix}, \quad C_m = (0.805197).$$

This model yields the cost $J = 285.012$.

With the input normal form, when $n_m = 2, 3$, two of the initial ωs are approximately the same, which leads to a significant numerical error in computing M_o and the numerical failure of the homotopy algorithm. Therefore this technique for choosing initial points fails, and some modification to the algorithm is needed to avoid this kind of ill conditioning. However, it is not at all clear how to systematically avoid nearly equal ωs, and this remains an open question. It can be shown that the solutions, obtained by the optimal projection equation approach, also have close ωs, which implies that changing the strategy for choosing initial points will not suffice for this example.

The solutions obtained by Ly's formulation are given in the following.

A model of order $n_m = 2$ is

$$A_m = \begin{pmatrix} 0.0 & 1.0 \\ -0.0487508 & -0.000487507 \end{pmatrix},$$

$$B_m = \begin{pmatrix} 0.0255931 & 0.499376 \\ -0.0000350605 & -0.000220956 \end{pmatrix}, \quad C_m = (1.0 \quad 0.0).$$

This model yields the cost $J = 29.2223$. A model of order $n_m = 3$ is

$$A_m = \begin{pmatrix} 0.0 & 1.0 & 0 \\ -0.0487508 & -0.000488017 & 0.0 \\ 0.0 & 0.0 & -2.48938 \end{pmatrix},$$

$$B_m = \begin{pmatrix} -0.0250401 & -0.499927 \\ 0.0000456588 & 0.000210679 \\ -1.45628 & 0.746877 \end{pmatrix}, \quad C_m = (1.0 \quad 0.0 \quad 1.0).$$

This model yields the cost $J = 28.6848$. Both of the above solutions have smaller cost than those obtained in [41]–[43].

EXAMPLE 7 [23], [39]. The system is given by

$$A = \begin{pmatrix} 0 & 0 & 0 & -150 \\ 1 & 0 & 0 & -245 \\ 0 & 1 & 0 & -1113 \\ 0 & 0 & 1 & -19 \end{pmatrix}, \quad B = \begin{pmatrix} 4 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \quad C = (0 \quad 0 \quad 0 \quad 1).$$

A model of order $n_m = 1$ is

$$A_m = (-0.495187), \quad B_m = (0.995175), \quad C_m = (0.0148426).$$

This model yields the cost $J = 4.90749 \cdot 10^{-5}$. A model of order $n_m = 2$ is

$$A_m = \begin{pmatrix} -0.437964 & -0.482612 \\ 2.84007 & -3.17242 \end{pmatrix},$$

$$B_m = \begin{pmatrix} 0.935911 \\ -2.51890 \end{pmatrix}, \quad C_m^T = (0.0149143 \quad 0.00682097).$$

This model yields the cost $J = 4.159 \cdot 10^{-7}$. A model of order $n_m = 3$ is

$$A_m = \begin{pmatrix} -0.437810 & -0.483078 & -0.0370108 \\ 2.82632 & -3.13536 & -0.612598 \\ -4.65184 & 13.1604 & -12.5542 \end{pmatrix},$$

$$B_m = \begin{pmatrix} 0.935746 \\ -2.50414 \\ 5.01082 \end{pmatrix}, \quad C_m = (0.0149143 \quad 0.00682180 \quad 0.000635413).$$

This model yields the cost $J = 4.59 \cdot 10^{-10}$.

EXAMPLE 8 [23]. The system is given by

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -50 & -79 & -33 & -5 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \quad C = (50 \quad 15 \quad 1 \quad 0).$$

A model of order $n_m = 1$ is

$$A_m = (-0.576205), \quad B_m = (1.07350), \quad C_m = (0.588692).$$

This model yields the cost $J = 0.104740$. A model of order $n_m = 2$ is

$$A_m = \begin{pmatrix} -0.532330 & -0.598751 \\ 3.80077 & -4.81512 \end{pmatrix}, \quad B_m = \begin{pmatrix} 1.03182 \\ -3.10326 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} 0.588704 \\ 0.278923 \end{pmatrix}.$$

This model yields the cost $J = 0.0269278$. A model of order $n_m = 3$ is

$$A_m = \begin{pmatrix} -0.520312 & -0.731867 & -0.162146 \\ 2.88892 & -2.23562 & -3.72129 \\ -1.08450 & 6.30540 & -0.746729 \end{pmatrix},$$

$$B_m^T = (\,1.02011 \quad -2.11453 \quad 1.22207\,), \quad C_m = (\,0.586461 \quad 0.307967 \quad 0.105043\,).$$

This model yields the cost $J = 0.00148438$.

EXAMPLE 9 [14]. The system is given by

$$A = \begin{pmatrix} -6.2036 & 15.054 & -9.8726 & -376.58 & 251.32 & -162.24 & 66.827 \\ 0.53 & -2.0176 & 1.4363 & 0 & 0 & 0 & 0 \\ 16.846 & 25.079 & -43.555 & 0 & 0 & 0 & 0 \\ 377.4 & -89.449 & -162.83 & 57.998 & -65.514 & 68.579 & 157.57 \\ 0 & 0 & 0 & 107.25 & -118.05 & 0 & 0 \\ 0.36992 & -0.14445 & -0.26303 & -0.64719 & 0.49947 & -0.21133 & 0 \\ 0 & 0 & 0 & 0 & 0 & 376.99 & 0 \end{pmatrix},$$

$$B = \begin{pmatrix} 89.353 & 0 \\ 376.99 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0.21133 \\ 0 & 0 \end{pmatrix}, \quad C = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

A model of order $n_m = 1$ is

$$A_m = -0.199272, \quad B_m = (\,0.631300 \quad -0.00187918\,), \quad C_m^T = (\,-0.187347 \quad -354.430\,).$$

This model yields the cost $J = 27632.2$. A model of order $n_m = 2$ is

$$A_m = \begin{pmatrix} -0.199608 & -0.0763006 \\ 3.33119 & -13.2758 \end{pmatrix},$$

$$B_m = \begin{pmatrix} 0.631832 & -0.00191612 \\ -5.15182 & -0.101952 \end{pmatrix}, \quad C_m = \begin{pmatrix} -0.201050 & 0.800899 \\ -354.414 & -66.1873 \end{pmatrix}.$$

This model yields the cost $J = 23262.3$. A model of order $n_m = 3$ is

$$A_m = \begin{pmatrix} -0.198769 & 0.235666 & -0.0248136 \\ -1.08739 & -0.912444 & 9.20181 \\ -0.115288 & -9.50243 & -0.0261157 \end{pmatrix},$$

$$B_m = \begin{pmatrix} -0.630503 & 0.00216112 \\ -1.350879 & -0.00377142 \\ -0.222387 & -0.0526803 \end{pmatrix}, \quad C_m = \begin{pmatrix} 0.291338 & -0.0265117 & -4.03570 \\ 354.222 & -164.479 & 26.6355 \end{pmatrix}.$$

This model yields the cost $J = 0.673079$. A model of order $n_m = 4$ is

$$A_m = \begin{pmatrix} -0.198769 & 0.235667 & -0.0248136 & 0.000915746 \\ -1.08739 & -0.912440 & 9.20181 & -0.00904508 \\ -0.115288 & -9.50243 & -0.0261155 & 0.00159031 \\ -5.46513 & -11.6984 & -1.92997 & -37.5544 \end{pmatrix},$$

$$B_m = \begin{pmatrix} -0.630503 & 0.00216112 \\ -1.35088 & -0.00377141 \\ -0.222386 & -0.0526803 \\ -8.66651 & -0.0203036 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} 0.291340 & 354.222 \\ -0.0265302 & -164.479 \\ -4.03569 & 26.6355 \\ 0.0861885 & -0.815898 \end{pmatrix}.$$

This model yields the cost $J = 3.22 \cdot 10^{-7}$.

For this example with $n_m = 3, 4$, the columns of the initial Jacobian matrices from input normal form formulations are so badly scaled that the numerical linear algebra in HOMPACK fails. Modifying HOMPACK to use the LINPACK subroutine DQRDC for the QR factorization of the initial Jacobian matrices enables HOMPACK to successfully overcome the ill conditioning and find a solution.

EXAMPLE 10 [4]. A is a 2×2 block diagonal matrix with each diagonal block being of the form

$$\begin{pmatrix} 0 & 1 \\ -\sigma_i^2 & -2y\sigma_i \end{pmatrix}, \quad i = 1, \ldots, n/2,$$

$$B = (0, b_1, 0, b_2, \ldots, 0, b_{n/2})^T, \quad C = (0, c_1, 0, c_2, \ldots, 0, c_{n/2}),$$

where $\sigma_i = i^2\pi^2$, $b_i = \sqrt{2}\sin(i\pi a)$, $c_i = \sqrt{2}\sin(i\pi s)$ and y, a, s are known parameters. This system was not studied in [41]–[43]. The input normal form approach can not give a solution when $n_m > 1$ because the initial ωs are generated in pairs.

Choosing $n = 16$, $n_m = 8$, $y = 0.001$, $a = 0.1$, $s = 0.2$, the reduced order model is

$$A_m = \text{ diag } (A_1, A_2, A_3, A_4),$$

$$A_1 = \begin{pmatrix} 0.0 & 1.0 \\ -24936.92 & -0.3158248 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0.0 & 1.0 \\ -97.40911 & -0.01973900 \end{pmatrix},$$

$$A_3 = \begin{pmatrix} 0.0 & 1.0 \\ -7890.149 & -0.1776489 \end{pmatrix}, \quad A_4 = \begin{pmatrix} 0.0 & 1.0 \\ -1558.546 & -0.07895572 \end{pmatrix},$$

$$B_m = \begin{pmatrix} 1.118022 \\ 0.3208260 \\ 0.3632675 \\ -0.007030323 \\ 1.538809 \\ -0.1661970 \\ 1.118019 \\ -0.07921770 \end{pmatrix}, \qquad C_m = \begin{pmatrix} 1.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 1.0 \\ 0.0 \end{pmatrix}^T,$$

which has cost $J = 2.59857$. Ly's formulation is very efficient for this problem.

EXAMPLE 11. The system is given by

$$A = \begin{pmatrix} -1 & 0 & 0 \\ 0.0005 & -1.000001 & 0 \\ 0.0005 & 0.0005 & -1.00001 \end{pmatrix}, \quad B = \begin{pmatrix} 1.1 \\ 1.2 \\ 1.3 \end{pmatrix}, \quad C = (2.1 \quad 2.2 \quad 2.3).$$

A model of order $n_m = 2$ with cost $J = 0.36 \cdot 10^{-14}$ is

$$A_m = \begin{pmatrix} -0.999519 & 0.00000 \\ 1.99976 & -1.00024 \end{pmatrix}, \quad B_m = \begin{pmatrix} -1.41387 \\ 1.41438 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} -5.61578 \\ 0.00000 \end{pmatrix}.$$

This system was constructed to illustrate that some problems can be solved by the input normal form formulation or the over-parametrization formulation described below but not by the Ly formulation.

6. HOMOTOPY ALGORITHM BASED ON OVER-PARAMETRIZATION FORMULATION FOR THE H^2 MODEL ORDER REDUCTION PROBLEM.

6.1. Over-parametrization formulation.

Both the input normal form formulation and Ly formulation can introduce ill conditioning, resulting from eliminating certain variables so that the minimal number of variables is used. To avoid such ill conditioning, one could use all the elements in A_m, B_m, and C_m as variables, i.e., not impose any restriction on the representation of (A_m, B_m, C_m).

The same Lagrangian as in (51) is used:

$$L(A_m, B_m, C_m, \tilde{Q}) = \mathrm{tr}\big[\tilde{Q}\tilde{R} + \big(\tilde{A}\tilde{Q} + \tilde{Q}\tilde{A}^T + \tilde{V}\big)\tilde{P}\big],$$

where the symmetric matrix $\tilde{P}$ is a Lagrange multiplier. Setting $\partial L/\partial \tilde{Q} = 0$ gives (23). A straightforward calculation shows

$$\begin{aligned}
\frac{\partial L}{\partial A_m} &= 2\big(\tilde{P}_{12}^T\tilde{Q}_{12} + \tilde{P}_2\tilde{Q}_2\big),\\
\frac{\partial L}{\partial B_m} &= 2\big(\tilde{P}_{12}^T B + \tilde{P}_2 B_m\big)V,\\
\frac{\partial L}{\partial C_m} &= 2R\big(C_m\tilde{Q}_2 - C\tilde{Q}_{12}\big).
\end{aligned} \tag{60}$$

6.2. A homotopy approach based on over-parametrization formulation.

Let A_f, B_f, C_f , R_f, and V_f denote A, B, C, R, and V in the above and define $A(\lambda)$, $B(\lambda)$, $C(\lambda)$, $V(\lambda)$ and $R(\lambda)$ as in (28) and denote them respectively by A, B, C, V, and R. Let

$$\begin{aligned}
H_{A_m}(\theta,\lambda) &= \frac{\partial L}{\partial A_m} = 2\big(\tilde{P}_{12}^T\tilde{Q}_{12} + \tilde{P}_2\tilde{Q}_2\big),\\
H_{B_m}(\theta,\lambda) &= \frac{\partial L}{\partial B_m} = 2\big(\tilde{P}_{12}^T B + \tilde{P}_2 B_m\big)V,\\
H_{C_m}(\theta,\lambda) &= \frac{\partial L}{\partial C_m} = 2R\big(C_m\tilde{Q}_2 - C\tilde{Q}_{12}\big),
\end{aligned} \tag{61}$$

where

$$\theta \equiv \begin{pmatrix} \mathrm{Vec}\ (A_m)\\ \mathrm{Vec}\ (B_m)\\ \mathrm{Vec}\ (C_m)\end{pmatrix}$$

denotes the independent variables A_m, B_m, and C_m, and $\tilde{Q}$ and $\tilde{P}$ satisfy respectively (20) and (23). Define

$$\rho(\theta,\lambda) = \begin{pmatrix} \text{Vec}\,\left[H_{A_m}(\theta,\lambda)\right] \\ \text{Vec}\,\left[H_{B_m}(\theta,\lambda)\right] \\ \text{Vec}\,\left[H_{C_m}(\theta,\lambda)\right] \end{pmatrix},$$

whose Jacobian matrix is

$$D\rho(\theta,\lambda) = \big(D_\theta\rho(\theta,\lambda),\ D_\lambda\rho(\theta,\lambda)\big).$$

Because of the over-parametrization, the Jacobian matrix of ρ is singular. The homotopy map is defined as

$$\hat{\rho}(\theta,\lambda) = \lambda\rho(\theta,\lambda) + (1-\lambda)(\theta-\theta_0), \tag{62}$$

which guarantees a well conditioned Jacobian matrix along the whole path except at the solution corresponding to $\lambda = 1$. The Jacobian matrix is given by

$$D\hat{\rho}(\theta,\lambda) = \big(\lambda D_\theta\rho(\theta,\lambda) + (1-\lambda)I, \quad \rho(\theta,\lambda) + \lambda D_\lambda\rho(\theta,\lambda) - (\theta-\theta_0)\big). \tag{63}$$

To find $D_\theta\rho(\theta,\lambda)$, define $\hat{H}_{A_m}\big(\tilde{P}^{(j)},\tilde{Q}^{(j)}\big)$, $\hat{H}_{B_m}\big(\tilde{P}^{(j)}\big)$, and $\hat{H}_{C_m}\big(\tilde{Q}^{(j)}\big)$ as in (56), where again the superscript (j) means $\partial/\partial\theta_j$. For $\theta_j = \big(A_m\big)_{kl}$,

$$\begin{aligned} \frac{\partial H_{A_m}}{\partial(A_m)_{kl}} &= \hat{H}_{A_m}\big(\tilde{P}^{(j)},\tilde{Q}^{(j)}\big) \\ \frac{\partial H_{B_m}}{\partial(A_m)_{kl}} &= \hat{H}_{B_m}\big(\tilde{P}^{(j)}\big), \\ \frac{\partial H_{C_m}}{\partial(A_m)_{kl}} &= \hat{H}_{C_m}\big(\tilde{Q}^{(j)}\big), \end{aligned} \tag{64}$$

for $\theta_j = \big(B_m\big)_{kl}$,

$$\begin{aligned} \frac{\partial H_{A_m}}{\partial(B_m)_{kl}} &= \hat{H}_{A_m}\big(\tilde{P}^{(j)},\tilde{Q}^{(j)}\big) \\ \frac{\partial H_{B_m}}{\partial(B_m)_{kl}} &= \hat{H}_{B_m}\big(\tilde{P}^{(j)}\big) + 2\tilde{P}_2E^{(k,l)}V, \\ \frac{\partial H_{C_m}}{\partial(B_m)_{kl}} &= \hat{H}_{C_m}\big(\tilde{Q}^{(j)}\big), \end{aligned} \tag{65}$$

and for $\theta_j = (C_m)_{kl}$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (C_m)_{kl}} &= \hat{H}_{A_m}\left(\tilde{P}^{(j)}, \tilde{Q}^{(j)}\right) \\
\frac{\partial H_{B_m}}{\partial (C_m)_{kl}} &= \hat{H}_{B_m}\left(\tilde{P}^{(j)}\right), \\
\frac{\partial H_{C_m}}{\partial (C_m)_{kl}} &= \hat{H}_{C_m}\left(\tilde{Q}^{(j)}\right) + 2RE^{(k,l)}\tilde{Q}_2.
\end{aligned} \tag{66}$$

$\tilde{P}^{(j)}$ and $\tilde{Q}^{(j)}$ can be obtained by solving the Lyapunov equations (33). The derivatives with respect to λ can be obtained in the same way as in Chapter 2.

6.3. Numerical algorithm for over-parametrization formulation.

The initial point $(\theta, \lambda) = (\theta_0, 0) = \left((A_m)_0, (B_m)_0, (C_m)_0, 0\right)$ is chosen so that the triple $\left((A_m)_0, (B_m)_0, (C_m)_0\right)$ is in balanced form and satisfies $\rho(\theta_0, 0) = 0$. The algorithm is similar to steps 1)–8) described in Chapter 2, except that the homotopy $\hat{\rho}$ from (62) is used.

For all the test problems except Example 6 with $n_m = 3$ and Example 9 with $n_m = 2$, 3, 4, the above algorithm gives satisfactory results by adjusting the curve tracking precision. For these exceptional cases, HOMPACK reaches $\lambda \geq 1$ very fast, but because of the high order singularity at the solution, the computed solution does not have acceptable accuracy. Although very sophisticated methods for dealing with singular endpoints of homotopy curves are known [24]–[26], these are difficult to implement in the present context, and the following simple algorithm was adequate.

1) Use the algorithm in Chapter 2 to track the curve until $\lambda \geq 1$.
2) Use the last point $\left(\tilde{\theta}, \tilde{\lambda}\right)$ before $\lambda \geq 1$ to redefine the homotopy map with $\theta_0 = \tilde{\theta}$ and set $\lambda = 0$.
3) Redo step 1.
4) Use Hermite polynomial interpolation to obtain the solution at $\lambda = 1$.

In Step 3 the new homotopy (62) has a zero curve that is nearly a straight line, and thus Hermite interpolation using points before $\lambda = 1$ and one point with $\lambda \geq 1$ is quite accurate. Care must be taken to use data points away from the singularity (lest they be inaccurate), but this is easily done by controlling the step size parameters in HOMPACK.

7. COMPARISON AND DISCUSSION.

Table 1 gives the CPU times in seconds and the number of steps needed to obtain the results for each example (a dash indicates failure). The CPU times are for a DECstation 5000/200, using double precision, IEEE arithmetic, and the MIPS RISC f77 compiler. Table 2 gives the comparison of the optimal projection equations approach and the input normal form formulation for Examples 8 and 9. The asterisks in Table 1 denote the cases requiring Hermite polynomial interpolation to obtain the solution for the over-parametrization formulation. The asterisks in Table 2 indicate cases that required special numerical linear algebra techniques to deal with severe scaling errors.

TABLE 1. ALGORITHM MEASURES FOR THREE FORMULATIONS.

		Input normal form		Ly's form		Over-parametrization	
Ex	n_m	steps	time	steps	time	steps	time
1	1	7	0.06	8	0.08	12	0.08
2	1	25	0.13	40	0.25	31	0.20
3	1	23	0.10	25	0.14	30	0.16
4	1	16	0.14	18	0.21	18	0.19
4	2	11	0.20	16	0.37	18	0.47
5	1	14	0.12	13	0.17	14	0.15
5	2	14	0.22	12	0.27	10	0.29
6	1	205	1.8	220	2.9	19	0.29
6	2	-	-	8	0.32	35	1.3
6	3	-	-	114	6.9	125*	13.
7	1	15	0.22	15	0.33	13	0.21
7	2	12	0.30	13	0.45	10	0.38
7	3	10	0.42	12	0.70	18	1.5
8	1	14	0.20	14	0.29	17	0.25
8	2	22	0.50	35	1.1	35	1.1
8	3	16	0.65	17	0.96	14	1.3
9	1	15	0.60	2339	103.	19	0.78
9	2	127	8.0	-	-	168*	13.
9	3	9	1.3	45	6.7	21*	4.2
9	4	8	1.9	59	15.	17*	7.5
10	8	-	-	19	35.	7	49.
11	2	6	0.13	-	-	16	0.42

As shown by Table 1, the input normal form homotopy can be very efficient. Also there is no need to adjust any parameter to achieve this efficiency (although to obtain

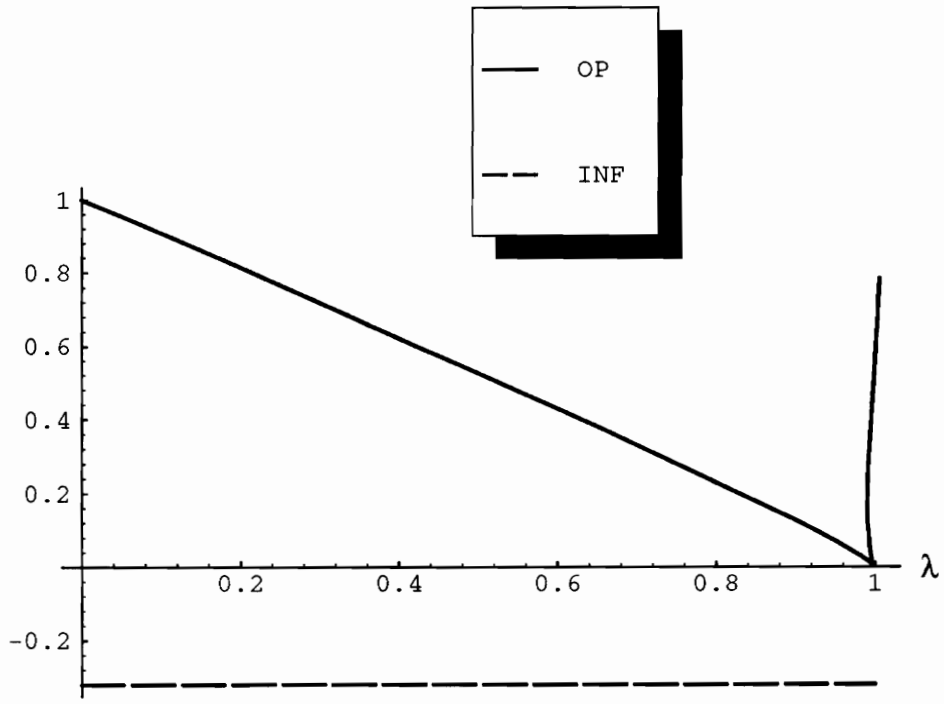

Fig. 1. x_7 (OP) and θ_1 (INF) versus λ.

the minimum solution of Example 1, some adjustment of the initial point was necessary). However, note that the potential ill conditioning of the input normal form formulation can result in failure (Examples 6 and 10) or the need for extraordinarily delicate linear algebra (Example 9).

Figures 1 and 2 show the behavior of the largest variation component with respect to λ for Example 5 at $n_m = 1$ and Example 9 at $n_m = 2$ using the input normal form formulation and the optimal projection equation formulation [41]–[43]. The figures show that component of the solution vector with the largest total amount of oscillation, corresponding to the most difficult component of the homotopy path to track. Even though Fig. 1 corresponds to a good choice of the initial point for the optimal projection equations approach, it is obviously not as efficient as the input normal form formulation. Generally speaking, since the number of variables in the input normal form and Ly formulations is much smaller than that of

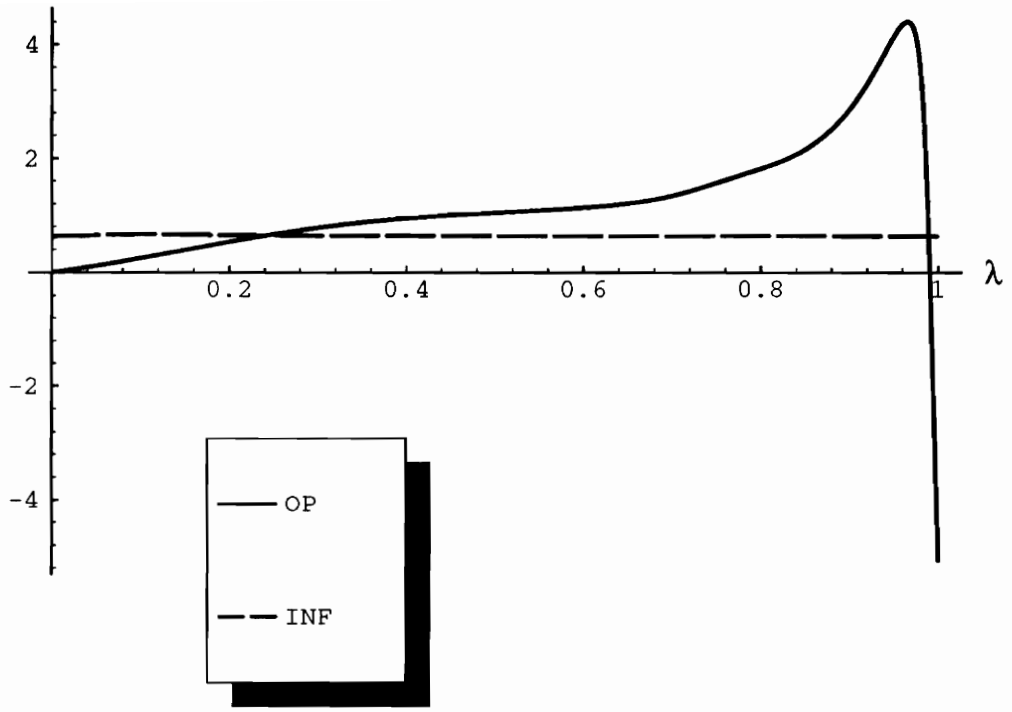

Fig. 2. x_7 (OP) and θ_2 (INF) versus λ.

the optimal projection equations formulation, and the strategy for choosing initial points uses balancing (hence giving an initial point closer to the final solution in most cases), the input normal form and Ly form homotopies are more efficient than the optimal projection equations homotopy.

For Example 9, when $n_m = 1$, the Ly's form homotopy is extremely inefficient, requiring c_1 and c_2 to be adjusted to achieve a solution. All attempts to obtain a solution when $n_m = 2$ failed. The solutions of Example 9 when $n_m = 3$ and $n_m = 4$ are singular, which accounts for the large number of steps required by Ly's form.

The optimal projection equations homotopy successfully solved *all* of the test problems, but Table 2, containing typical results, shows that the minimal parameter homotopies are much more efficient. However, when the input normal form and Ly's form are used, some

TABLE 2. COMPARISON OF METHODS.

	Example 8			
	Optimal projection		input normal form	
n_m	# steps	time (sec)	# steps	time (sec)
1	35	0.6	14	0.20
2	61	2.7	22	0.50
3	129	14.	16	0.65
	Example 9			
2	615	88	127	8.0
3	641	223	9*	1.3
4	711	518	8*	1.9

restrictions are imposed on the structure of the triple (A_m, B_m, C_m), potentially resulting in ill conditioning. For the input normal form formulation, ill conditioning occurs if two diagonal elements of Ω in (4) are approximately the same. In other words, let Q_m and P_m be the controllability and observability Gramians of the system represented by (A_m, B_m, C_m), and let

$$Q_m = W\Sigma W^T, \quad P_m = W^{-T}\Sigma W^{-1},$$

where Σ is diagonal and is the controllability and observability Gramian in balanced form. If two diagonal elements of Σ are approximately the same, then ill conditioning occurs. For Example 6, when $n_m = 2, 3$, both the initial point chosen using the given strategy and the solution obtained in [41]–[43] or by Ly's formulation are ill conditioned, i.e., two diagonal elements of Ω are approximately the same. Hence the input normal form method will not be able to solve this problem.

For Ly's formulation, ill conditioning occurs if the Jordan decomposition of A_m is ill conditioned. Precisely, if the two eigenvalues of A_m which are to be grouped into a 2×2 block are approximately the same, the transition matrix to 2×2 block diagonal form is ill conditioned. This can be clearly illustrated by observing that for $n_m = 2$, finding the Ly form is equivalent to finding the transition matrix $T \in \mathbf{R}^{2\times 2}$ such that

$$\begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{pmatrix} = \begin{pmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -\lambda_1\lambda_2 & \lambda_1 + \lambda_2 \end{pmatrix},$$

$$C_m(1,1)t_{11} + C_m(1,2)t_{21} = 1,$$

$$C_m(1,1)t_{12} + C_m(1,2)t_{22} = 0,$$

where λ_1 and λ_2 are the eigenvalues of A_m. Trivial algebra gives

$$t_{11} = -\lambda_2 C_m(1,1)^{-1}\lambda_{12}^{-1}, \qquad t_{12} = C_m(1,1)^{-1}\lambda_{12}^{-1},$$

$$t_{21} = \lambda_1 C_m(1,2)^{-1}\lambda_{12}^{-1}, \qquad t_{22} = -C_m(1,2)^{-1}\lambda_{12}^{-1},$$

$$\text{cond } T = \sigma + \sqrt{\sigma^2 - 1},$$

where

$$\lambda_{12} = \lambda_1 - \lambda_2, \qquad \tau = \frac{C_m(1,1)}{C_m(1,2)}, \qquad \sigma = \frac{1 + \tau^2 + \lambda_2^2 + \tau^2\lambda_1^2}{2|\tau\lambda_{12}|}.$$

Thus ill conditioning occurs in general when σ is large, and in particular when $\tau\lambda_{12} \approx 0$. Furthermore, note that the very existence of the Ly form is predicated on the *assumption that the Jordan form of* A_m *consists of* 2×2 *Jordan blocks*, which is a rather strong assumption.

Both the input normal form formulation and Ly's formulation can fail to exist or lead to ill conditioning and it is conceivable that both of these formulations will fail for some problems. This failure of existence in general is related to the insistence on using the minimal number of parameters $n_m m + n_m l$. The over-parametrization formulation solves the ill conditioning issue, but introduces a very high order singularity at the solution. It is doubtful whether either the Hermite interpolation used here or the techniques of [24]–[26] can handle a large problem with a singularity of order 100. A pragmatic suggestion is to try in order the input normal form, Ly's form, and the over-parametrization form, switching if ill conditioning or failure occurs. The ideal paradigm would be to have a family of minimal formulations, almost all of which exist for any given problem. The homotopy algorithm would then dynamically adjust the formulation, finding a well conditioned one and tracking its zero curve simultaneously. Such a paradigm remains an open question.

8. HOMOTOPY ALGORITHM BASED ON THE INPUT NORMAL FORM FOR THE COMBINED H^2/H^∞ MODEL REDUCTION PROBLEM.

8.1. The formulation.

To optimize $\mathcal{J}(A_m, B_m, C_m)$ over the open set $\mathcal{S}$ under the constraints that symmetric positive definite $\mathcal{Q}$ satisfies (14), and (A_m, B_m, C_m) is in input normal form, the following Lagrangian is formed:

$$\mathcal{L}(A_m, B_m, C_m, \Omega, \mathcal{Q}, \mathcal{P}, M_c, M_o) \equiv \operatorname{tr}\left[\mathcal{Q}\tilde{R} + (\tilde{A}\mathcal{Q} + \mathcal{Q}\tilde{A}^T + \gamma^{-2}\mathcal{Q}\tilde{R}\mathcal{Q} + \tilde{V})\mathcal{P}\right.$$
$$\left. + \left(A_m + A_m^T + B_m V B_m^T\right)M_c + \left(A_m^T\Omega + \Omega A_m + C_m^T R C_m\right)M_o\right],$$

where the symmetric matrices M_c, M_o, and $\mathcal{P} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ are Lagrange multipliers. $\Omega = \operatorname{diag}(\omega_1, \ldots, \omega_{n_m})$ is related to the input normal form constraint. Setting $\partial\mathcal{L}/\partial\mathcal{Q} = 0$ yields

$$0 = \left(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}\right)^T\mathcal{P} + \mathcal{P}\left(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}\right) + \tilde{R}. \tag{67}$$

Partition $\mathcal{Q}$, $\mathcal{P} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ into

$$\mathcal{Q} = \begin{pmatrix} \mathcal{Q}_1 & \mathcal{Q}_{12} \\ \mathcal{Q}_{12}^T & \mathcal{Q}_2 \end{pmatrix}, \qquad \mathcal{P} = \begin{pmatrix} \mathcal{P}_1 & \mathcal{P}_{12} \\ \mathcal{P}_{12}^T & \mathcal{P}_2 \end{pmatrix} \tag{68}$$

where $\mathcal{Q}_1$, $\mathcal{P}_1 \in \mathbf{R}^{n\times n}$ and $\mathcal{Q}_2$, $\mathcal{P}_2 \in \mathbf{R}^{n_m\times n_m}$. Define

$$\mathcal{P}\mathcal{Q} \equiv Z = \begin{pmatrix} Z_1 & Z_{12} \\ Z_{21} & Z_2 \end{pmatrix} \tag{69}$$

where

$$Z_1 \equiv \mathcal{P}_1\mathcal{Q}_1 + \mathcal{P}_{12}\mathcal{Q}_{12}^T, \qquad Z_{12} \equiv \mathcal{P}_1\mathcal{Q}_{12} + \mathcal{P}_{12}\mathcal{Q}_2,$$

$$Z_{21} \equiv \mathcal{P}_{12}^T\mathcal{Q}_1 + \mathcal{P}_2\mathcal{Q}_{12}^T, \qquad Z_2 \equiv \mathcal{P}_{12}^T\mathcal{Q}_{12} + \mathcal{P}_2\mathcal{Q}_2.$$

$\partial L/\partial\Omega = 0$ and $\partial L/\partial A_m = 0$ yield

$$0 = 2M_c + 2\Omega M_o + 2\left(\mathcal{P}_{12}^T\mathcal{Q}_{12} + \mathcal{P}_2\mathcal{Q}_2\right), \qquad 0 = \left(A_m M_o\right)_{ii}, \quad 1 \le i \le n_m.$$

A straightforward calculation shows

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial B_m} &= 2(\mathcal{P}_{12}^T B V + \mathcal{P}_2 B_m V) + 2M_c B_m V, \\
\frac{\partial \mathcal{L}}{\partial C_m} &= 2\big(RC_m \mathcal{Q}_2 - RC\mathcal{Q}_{12}\big) + 2RC_m M_o \\
&\quad + \gamma^{-2}\big[-RC(Z_1^T \mathcal{Q}_{12} + Z_{21}^T \mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12} Z_2) \\
&\quad\quad + RC_m(\mathcal{Q}_{12}^T Z_{12} + Z_{12}^T \mathcal{Q}_{12} + \mathcal{Q}_2 Z_2 + Z_2^T \mathcal{Q}_2)\big].
\end{aligned} \tag{70}$$

The matrices M_c and M_o in (70) satisfy (26).

A homotopy approach based on the input normal form is now described. Let A_f, B_f, C_f , R_f, V_f, and γ_f denote A, B, C, R, V, and γ in the above and define $A(\lambda)$, $B(\lambda)$, $C(\lambda)$, $R(\lambda)$, and $V(\lambda)$ as in (28) and define

$$\gamma(\lambda) = \gamma_0 + \lambda(\gamma_f - \gamma_0).$$

For brevity, $A(\lambda)$, $B(\lambda)$, $C(\lambda)$, $R(\lambda)$, $V(\lambda)$, and $\gamma(\lambda)$ will be denoted by A, B, C, R, V, and γ respectively in the following. Let

$$\begin{aligned}
H_{B_m}(\theta, \lambda) = \frac{\partial L}{\partial B_m} &= 2\big(\mathcal{P}_{12}^T B + \mathcal{P}_2 B_m\big)V + 2M_c B_m V, \\
H_{C_m}(\theta, \lambda) = \frac{\partial L}{\partial C_m} &= 2R\big(C_m \mathcal{Q}_2 - C\mathcal{Q}_{12}\big) + 2RC_m M_o \\
&\quad + \gamma^{-2}\big[-RC(Z_1^T \mathcal{Q}_{12} + Z_{21}^T \mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12} Z_2) \\
&\quad\quad + RC_m(\mathcal{Q}_{12}^T Z_{12} + Z_{12}^T \mathcal{Q}_{12} + \mathcal{Q}_2 Z_2 + Z_2^T \mathcal{Q}_2)\big],
\end{aligned}$$

where

$$\theta \equiv \begin{pmatrix} \text{Vec } (B_m) \\ \text{Vec } (C_m) \end{pmatrix}$$

denotes the independent variables B_m and C_m, M_o and M_c satisfy (26), and $\mathcal{Q}$ and $\mathcal{P}$ satisfy respectively (14) and (67) with partitioned forms (68). Vec(P) for a matrix $P \in \mathbf{R}^{p \times q}$ is the concatenation of its columns:

$$\text{Vec}(P) \equiv \begin{pmatrix} P_{\cdot 1} \\ P_{\cdot 2} \\ \vdots \\ P_{\cdot q} \end{pmatrix} \in \mathbf{R}^{pq}.$$

The homotopy map is defined as

$$\rho(\theta,\lambda)=\begin{pmatrix}\text{Vec}\,\big[H_{B_m}(\theta,\lambda)\big]\\ \text{Vec}\,\big[H_{C_m}(\theta,\lambda)\big]\end{pmatrix}, \tag{71}$$

and its Jacobian matrix is

$$D\rho(\theta,\lambda)=\big(D_\theta\rho(\theta,\lambda),D_\lambda\rho(\theta,\lambda)\big). \tag{72}$$

Define

$$\begin{aligned}
\hat{H}_{B_m}\big(\mathcal{P}^{(j)},M_c^{(j)}\big)&=2\big(\mathcal{P}_{12}^{T(j)}B+\mathcal{P}_2^{(j)}B_m\big)V+2M_c^{(j)}B_mV,\\
\hat{H}_{C_m}\big(\mathcal{Q}^{(j)},Z^{(j)},M_o^{(j)}\big)&=2R\big(C_m\mathcal{Q}_2^{(j)}-C\mathcal{Q}_{12}^{(j)}\big)+2RC_mM_o^{(j)}\\
&\quad-\gamma^{-2}RC\big(Z_1^{T(j)}\mathcal{Q}_{12}+Z_{21}^{T(j)}\mathcal{Q}_2+Z_1^T\mathcal{Q}_{12}^{(j)}+Z_{21}^T\mathcal{Q}_2^{(j)}\\
&\qquad+\mathcal{Q}_1^{(j)}Z_{12}+\mathcal{Q}_1Z_{12}^{(j)}+\mathcal{Q}_{12}^{(j)}Z_2+\mathcal{Q}_{12}Z_2^{(j)}\big)\\
&\quad+\gamma^{-2}RC_m\big(Z_{12}^{T(j)}\mathcal{Q}_{12}+Z_{12}^T\mathcal{Q}_{12}^{(j)}+\mathcal{Q}_{12}^{T(j)}Z_{12}+\mathcal{Q}_{12}^TZ_{12}^{(j)}\\
&\qquad+\mathcal{Q}_2^{(j)}Z_2+Z_2^{T(j)}\mathcal{Q}_2+\mathcal{Q}_2Z_2^{(j)}+Z_2^T\mathcal{Q}_2^{(j)}\big),
\end{aligned}$$

where the superscript (j) means $\partial/\partial\theta_j$: $Y^{(j)}\equiv\partial Y/\partial\theta_j$. Using the above definitions, we have for $\theta_j=\big(B_m\big)_{kl}$,

$$\begin{aligned}
\frac{\partial H_{B_m}}{\partial(B_m)_{kl}}&=\hat{H}_{B_m}\big(\mathcal{P}^{(j)},M_c^{(j)}\big)+2\big(\mathcal{P}_2+M_c\big)E^{(k,l)}V,\\
\frac{\partial H_{C_m}}{\partial(B_m)_{kl}}&=\hat{H}_{C_m}\big(\mathcal{Q}^{(j)},Z^{(j)},M_o^{(j)}\big),
\end{aligned} \tag{73}$$

and for $\theta_j=\big(C_m\big)_{kl}$,

$$\begin{aligned}
\frac{\partial H_{B_m}}{\partial(C_m)_{kl}}&=\hat{H}_{B_m}\big(\mathcal{P}^{(j)},M_c^{(j)}\big),\\
\frac{\partial H_{C_m}}{\partial(C_m)_{kl}}&=\hat{H}_{C_m}\big(\mathcal{Q}^{(j)},Z^{(j)},M_o^{(j)}\big)+2RE^{(k,l)}\big(\mathcal{Q}_2+M_o\big)\\
&\quad+\gamma^{-2}RE^{(k,l)}\big(Z_{12}^T\mathcal{Q}_{12}+\mathcal{Q}_{12}^TZ_{12}+\mathcal{Q}_2^TZ_2+Z_2^T\mathcal{Q}_2\big),
\end{aligned} \tag{74}$$

where $E^{(k,l)}$ is a matrix of the appropriate dimension whose only nonzero element is $e_{kl}=1$. $\mathcal{P}^{(j)}$ and $\mathcal{Q}^{(j)}$ can be obtained by solving the Lyapunov equations

$$\begin{aligned}
0&=\big(\tilde{A}+\gamma^{-2}\mathcal{Q}\tilde{R}\big)\mathcal{Q}^{(j)}+\mathcal{Q}^{(j)}\big(\tilde{A}+\gamma^{-2}\mathcal{Q}\tilde{R}\big)^T+\tilde{V}^{(j)}+\tilde{A}^{(j)}\mathcal{Q}+\mathcal{Q}\tilde{A}^{T(j)}+\gamma^{-2}\mathcal{Q}\tilde{R}^{(j)}\mathcal{Q},\\
0&=\big(\tilde{A}+\gamma^{-2}\mathcal{Q}\tilde{R}\big)^T\mathcal{P}^{(j)}+\mathcal{P}^{(j)}\big(\tilde{A}+\gamma^{-2}\mathcal{Q}\tilde{R}\big)+\tilde{R}^{(j)}\\
&\quad+\big(\tilde{A}^{(j)}+\gamma^{-2}\mathcal{Q}^{(j)}\tilde{R}+\gamma^{-2}\mathcal{Q}\tilde{R}^{(j)}\big)^T\mathcal{P}+\mathcal{P}\big(\tilde{A}^{(j)}+\gamma^{-2}\mathcal{Q}^{(j)}\tilde{R}+\gamma^{-2}\mathcal{Q}\tilde{R}^{(j)}\big).
\end{aligned} \tag{75}$$

Similarly for λ, using a dot to denote $\partial/\partial\lambda$,

$$\begin{aligned}
\frac{\partial H_{B_m}}{\partial\lambda} &= \hat{H}_{B_m}(\dot{\mathcal{P}}, \dot{M}_c) + 2\mathcal{P}_{12}^T(\dot{B}V + B\dot{V}) + 2(\mathcal{P}_2 + M_c)B_m\dot{V}, \\
\frac{\partial H_{C_m}}{\partial\lambda} &= \hat{H}_{C_m}(\dot{\mathcal{Q}}, \dot{Z}, \dot{M}_o) + 2\dot{R}C_m(\mathcal{Q}_2 + M_o) - 2(\dot{R}C + R\dot{C})\mathcal{Q}_{12} \\
&\quad + \gamma^{-2}\dot{R}\,h_\lambda - 2\gamma^{-3}\dot{\gamma}R\,h_\lambda - \gamma^{-2}R\dot{C}(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1Z_{12} + \mathcal{Q}_{12}Z_2),
\end{aligned} \tag{76}$$

where

$$h_\lambda = -C(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1Z_{12} + \mathcal{Q}_{12}Z_2) + C_m(\mathcal{Q}_{12}^TZ_{12} + Z_{12}^T\mathcal{Q}_{12} + \mathcal{Q}_2Z_2 + Z_2^T\mathcal{Q}_2),$$

and $\dot{\mathcal{P}}$ and $\dot{\mathcal{Q}}$ are obtained by solving the Lyapunov equations

$$\begin{aligned}
0 &= (\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R})\dot{\mathcal{Q}} + \dot{\mathcal{Q}}(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R})^T + \dot{\tilde{V}} + \dot{\tilde{A}}\,\mathcal{Q} + \mathcal{Q}\dot{\tilde{A}}^T + \gamma^{-2}\mathcal{Q}\dot{\tilde{R}}\mathcal{Q} - 2\gamma^{-3}\dot{\gamma}\mathcal{Q}\tilde{R}\mathcal{Q}, \\
0 &= (\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R})^T\dot{\mathcal{P}} + \dot{\mathcal{P}}(\tilde{A} + \gamma^{-2}\mathcal{Q}\tilde{R}) + \dot{\tilde{R}} \\
&\quad + (\dot{\tilde{A}} + \gamma^{-2}\dot{\mathcal{Q}}\tilde{R} + \gamma^{-2}\mathcal{Q}\dot{\tilde{R}} - 2\gamma^{-3}\dot{\gamma}\mathcal{Q}\tilde{R})^T\mathcal{P} + \mathcal{P}(\dot{\tilde{A}} + \gamma^{-2}\dot{\mathcal{Q}}\tilde{R} + \gamma^{-2}\mathcal{Q}\dot{\tilde{R}} - 2\gamma^{-3}\dot{\gamma}\mathcal{Q}\tilde{R}).
\end{aligned} \tag{77}$$

8.2. Numerical algorithm for input normal form homotopy.

The initial point $(\theta, \lambda) = (\theta_0, 0) = \big((B_m)_0, (C_m)_0, 0\big)$ is ideally chosen so that the triple $\big((A_m)_0, (B_m)_0, (C_m)_0\big)$ is in input normal form and satisfies $\rho(\theta_0, 0) = 0$.

This suggests that if the initial γ, i.e., γ_0, is chosen to be very large, the same approach as for the H^2 optimal model reduction problem in Chapter 2 can be used. Therefore the initial point $(\theta_0, 0)$ is chosen the same way as step 1)–5) in Chapter 2.3 while γ_0 is chosen to be very large.

Once the initial point is chosen, the rest of the computation is as follows:

1) Set $\lambda := 0$, $\theta := \theta_0$.
2) Calculate A_m from B_m and C_m, $\tilde{R}$, $\tilde{V}$, and compute $\mathcal{Q}$ and $\mathcal{P}$ according to (14) and (67).
3) Evaluate S from (27) and M_o and M_c according to (26).
4) Evaluate the homotopy map $\rho(\theta, \lambda)$ in (71) and $D\rho(\theta, \lambda)$ in (72).
5) Predict the next point $Z^{(0)} = \big(\theta^{(0)}, \lambda^{(0)}\big)$ on the homotopy zero curve using, e.g., a Hermite cubic interpolant.

6) For $k := 0, 1, 2, \cdots$ until convergence do

$$Z^{(k+1)} = \left[D\rho(Z^{(k)})\right]^{\dagger} \rho(Z^{(k)}),$$

where $\left[D\rho(Z)\right]^{\dagger}$ is the Moore-Penrose inverse of $D\rho(Z)$. Let $(\theta_1, \lambda_1) = \lim_{k \to \infty} Z^{(k)}$.

7) If $\lambda_1 < 1$, then set $\theta := \theta_1$, $\lambda := \lambda_1$, and go to step 2).

8) If $\lambda_1 \geq 1$, compute the solution $\bar{\theta}$ at $\lambda = 1$. A_m is then obtained from B_m and C_m.

9. HOMOTOPY ALGORITHM BASED ON LY FORMULATION FOR THE COMBINED H^2/H^∞ MODEL REDUCTION PROBLEM.

To optimize $\mathcal{J}(A_m, B_m, C_m)$ over the open set $\mathcal{S}$ under the constraint that symmetric positive definite $\mathcal{Q}$ satisfies (14), and (A_m, B_m, C_m) is in Ly's form, the following Lagrangian is formed:

$$\mathcal{L}(A_m, B_m, C_m, \mathcal{P}, \mathcal{Q}) \equiv \text{tr}\,\left[\mathcal{Q}\tilde{R} + (\tilde{A}\mathcal{Q} + \mathcal{Q}\tilde{A}^T + \gamma^{-2}\mathcal{Q}\tilde{R}\mathcal{Q} + \tilde{V})\mathcal{P}\right],$$

where $\mathcal{P} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ is a Lagrange multiplier. Setting $\partial\mathcal{L}/\partial\mathcal{Q} = 0$ yields (67). Partition $\mathcal{Q}$, $\mathcal{P} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ as in (68) and define $\mathcal{P}\mathcal{Q} = Z$ as in (69). The partial derivatives of $\mathcal{L}$ can be computed as

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial(A_m)_{ij}} &= 2\big(\mathcal{P}_{12}^T\,\mathcal{Q}_{12} + \mathcal{P}_2\,\mathcal{Q}_2,\big)_{ij}, \qquad (i,j) \in \mathcal{I} \\
\frac{\partial\mathcal{L}}{\partial B_m} &= 2\big(\mathcal{P}_{12}^T BV + \mathcal{P}_2 B_m V\big), \\
\frac{\partial\mathcal{L}}{\partial(C_m)_{ij}} &= 2\big(RC_m\mathcal{Q}_2 - RC\mathcal{Q}_{12}\big)_{ij} + \\
&\quad + \gamma^{-2}\big[-RC(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12}Z_2) \\
&\quad + RC_m(\mathcal{Q}_{12}^T Z_{12} + Z_{12}^T\mathcal{Q}_{12} + \mathcal{Q}_2 Z_2 + Z_2^T\mathcal{Q}_2)\big]_{ij}.
\end{aligned} \tag{78}$$

Let A_f, B_f, C_f , R_f, V_f, and γ_f denote A, B, C, R, V, and γ in the above and define $A(\lambda)$, $B(\lambda)$, $C(\lambda)$, $R(\lambda)$, $V(\lambda)$ as in (28) and

$$\gamma(\lambda) = \gamma_0 + \lambda(\gamma_f - \gamma_0)$$

and denote them by A, B, C, R, V, and γ respectively in the following. Let

$$\begin{aligned}
H_{A_m}(\theta,\lambda) &= \frac{\partial\mathcal{L}}{\partial A_m} = 2\big(\mathcal{P}_{12}^T\mathcal{Q}_{12} + \mathcal{P}_2\mathcal{Q}_2\big), \\
H_{B_m}(\theta,\lambda) &= \frac{\partial\mathcal{L}}{\partial B_m} = 2\big(\mathcal{P}_{12}^T B + \mathcal{P}_2 B_m\big)V, \\
H_{C_m}(\theta,\lambda) &= \frac{\partial\mathcal{L}}{\partial C_m} = 2R\big(C_m\mathcal{Q}_2 - C\mathcal{Q}_{12}\big) \\
&\quad + \gamma^{-2}\big[-RC(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12}Z_2) \\
&\quad + RC_m(\mathcal{Q}_{12}^T Z_{12} + Z_{12}^T\mathcal{Q}_{12} + \mathcal{Q}_2 Z_2 + Z_2^T\mathcal{Q}_2)\big],
\end{aligned}$$

where in H_{A_m} only those elements corresponding to the parameter elements of A_m are of interest and

$$\theta \equiv \begin{pmatrix} (A_m)_{\mathcal{I}} \\ \text{Vec } (B_m) \\ \text{Vec } (C_m)_{\mathcal{T}.} \end{pmatrix} \tag{79}$$

denotes the independent variables, $\mathcal{Q}$ and $\mathcal{P}$ satisfy respectively (14) and (67), $(A_m)_{\mathcal{I}}$ is a vector consisting of those elements in A_m with indices in the set $\mathcal{I}$, i.e.,

$$(A_m)_{\mathcal{I}} = \big((A_m)_{21}, (A_m)_{22}, \cdots, (A_m)_{n_m n_m}\big)^T,$$

and $(C_m)_{\mathcal{T}.}$ is the matrix obtained from rows $\mathcal{T} = \{2, \ldots, l\}$ of C_m.

The homotopy map is defined as

$$\rho(\theta, \lambda) = \begin{pmatrix} \big[H_{A_m}(\theta, \lambda)\big]_{\mathcal{I}} \\ \text{Vec } \big[H_{B_m}(\theta, \lambda)\big] \\ \text{Vec } \big[H_{C_m}(\theta, \lambda)\big]_{\mathcal{T}.} \end{pmatrix}, \tag{80}$$

and its Jacobian matrix is

$$D\rho(\theta, \lambda) = \big(D_\theta \rho(\theta, \lambda), D_\lambda \rho(\theta, \lambda)\big).$$

Define

$$\begin{aligned}
\hat{H}_{A_m}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big) &= 2\big(\mathcal{P}_{12}^{T(j)}\mathcal{Q}_{12} + \mathcal{P}_{12}^T\mathcal{Q}_{12}^{(j)} + \mathcal{P}_2^{(j)}\mathcal{Q}_2 + \mathcal{P}_2\mathcal{Q}_2^{(j)}\big), \\
\hat{H}_{B_m}\big(\mathcal{P}^{(j)}\big) &= 2\big(\mathcal{P}_{12}^{T(j)}B + \mathcal{P}_2^{(j)}B_m\big)V, \\
\hat{H}_{C_m}\big(\mathcal{Q}^{(j)}, Z^{(j)}\big) &= 2R\big(C_m\mathcal{Q}_2^{(j)} - C\mathcal{Q}_{12}^{(j)}\big) \\
&\quad - \gamma^{-2}RC\big(Z_1^{T(j)}\mathcal{Q}_{12} + Z_{21}^{T(j)}\mathcal{Q}_2 + Z_1^T\mathcal{Q}_{12}^{(j)} + Z_{21}^T\mathcal{Q}_2^{(j)} \\
&\qquad + \mathcal{Q}_1^{(j)}Z_{12} + \mathcal{Q}_1 Z_{12}^{(j)} + \mathcal{Q}_{12}^{(j)}Z_2 + \mathcal{Q}_{12}Z_2^{(j)}\big) \\
&\quad + \gamma^{-2}RC_m\big(Z_{12}^{T(j)}\mathcal{Q}_{12} + Z_{12}^T\mathcal{Q}_{12}^{(j)} + \mathcal{Q}_{12}^{T(j)}Z_{12} + \mathcal{Q}_{12}^T Z_{12}^{(j)} \\
&\qquad + \mathcal{Q}_2^{(j)}Z_2 + Z_2^{T(j)}\mathcal{Q}_2 + \mathcal{Q}_2 Z_2^{(j)} + Z_2^T\mathcal{Q}_2^{(j)}\big),
\end{aligned} \tag{81}$$

where the superscript (j) means $\partial/\partial\theta_j$. Using the above definitions, we have for $\theta_j = (A_m)_{kl}$, where $(k, l) \in \mathcal{I}$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (A_m)_{kl}} &= \hat{H}_{A_m}\big(\mathcal{P}^{(j)}, Z^{(j)}\big), \\
\frac{\partial H_{B_m}}{\partial (A_m)_{kl}} &= \hat{H}_{B_m}\big(\mathcal{P}^{(j)}\big), \\
\frac{\partial H_{C_m}}{\partial (A_m)_{kl}} &= \hat{H}_{C_m}\big(\mathcal{Q}^{(j)}, Z^{(j)}\big),
\end{aligned} \tag{82}$$

for $\theta_j = (B_m)_{kl}$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (B_m)_{kl}} &= \hat{H}_{A_m}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big), \\
\frac{\partial H_{B_m}}{\partial (B_m)_{kl}} &= \hat{H}_{B_m}\big(\mathcal{P}^{(j)}\big) + 2\mathcal{P}_2 E^{(k,l)} V, \\
\frac{\partial H_{C_m}}{\partial (B_m)_{kl}} &= \hat{H}_{C_m}\big(\mathcal{Q}^{(j)}, Z^{(j)}\big),
\end{aligned} \tag{83}$$

and for $\theta_j = (C_m)_{kl}$, where $k > 1$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (C_m)_{kl}} &= \hat{H}_{A_m}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big), \\
\frac{\partial H_{B_m}}{\partial (C_m)_{kl}} &= \hat{H}_{B_m}\big(\mathcal{P}^{(j)}\big), \\
\frac{\partial H_{C_m}}{\partial (C_m)_{kl}} &= \hat{H}_{C_m}\big(\mathcal{Q}^{(j)}, Z^{(j)}\big) + 2RE^{(k,l)}\mathcal{Q}_2 \\
&\quad + \gamma^{-2} R E^{(k,l)}\big(Z_{12}^T \mathcal{Q}_{12} + \mathcal{Q}_{12}^T Z_{12} + \mathcal{Q}_2^T Z_2 + Z_2^T \mathcal{Q}_2\big),
\end{aligned} \tag{84}$$

where $\mathcal{P}^{(j)}$ and $\mathcal{Q}^{(j)}$ can be obtained by solving the Lyapunov equation (75). Similarly for λ, using a dot to denote $\partial/\partial\lambda$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial \lambda} &= \hat{H}_{A_m}\big(\dot{\mathcal{P}}, \dot{\mathcal{Q}}\big), \\
\frac{\partial H_{B_m}}{\partial \lambda} &= \hat{H}_{B_m}\big(\dot{\mathcal{P}}\big) + 2\mathcal{P}_{12}^T\big(\dot{B}V + B\dot{V}\big) + 2\mathcal{P}_2 B_m \dot{V}, \\
\frac{\partial H_{C_m}}{\partial \lambda} &= \hat{H}_{C_m}\big(\dot{\mathcal{Q}}, \dot{Z}\big) - 2\big(\dot{R}C + R\dot{C}\big)\mathcal{Q}_{12} + 2\dot{R}C_m\mathcal{Q}_2 \\
&\quad + \gamma^{-2}\dot{R}\,h_\lambda - 2\gamma^{-3}\dot{\gamma}R\,h_\lambda - \gamma^{-2}R\dot{C}\big(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12}Z_2\big),
\end{aligned} \tag{85}$$

where

$$h_\lambda = -C\big(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12}Z_2\big) + C_m\big(\mathcal{Q}_{12}^T Z_{12} + Z_{12}^T\mathcal{Q}_{12} + \mathcal{Q}_2 Z_2 + Z_2^T\mathcal{Q}_2\big),$$

and $\dot{\mathcal{P}}$ and $\dot{\mathcal{Q}}$ are obtained by solving (77).

Choose the initial γ so that γ_0^{-2} is approximately zero. The initial point $(\theta, \lambda) = (\theta_0, 0)$ is chosen so that the triple $\big((A_m)_0, (B_m)_0, (C_m)_0\big)$ is in Ly's form and satisfies $\rho(\theta_0, 0) = 0$. This can be done as follows:

1) Obtain the initial reduced order model $\big((A_m)_0, (B_m)_0, (C_m)_0\big)_b$ in balanced form in the same way as for the input normal form approach.
2) Transform the balanced $\big((A_m)_0, (B_m)_0, (C_m)_0\big)_b$ to Ly's form, and build θ_0 as described in (79).

The homotopy curve tracking computation is the same as described in Section 8.2.

10. HOMOTOPY ALGORITHM BASED ON OVER-PARAMETRIZATION FORMULATION FOR THE COMBINED H^2/H^∞ MODEL REDUCTION PROBLEM.

To optimize $\mathcal{J}(A_m, B_m, C_m)$ over the open set $\mathcal{S}$ under the constraint that symmetric positive definite $\mathcal{Q}$ satisfies (14), the following Lagrangian is formed:

$$\mathcal{L}(A_m, B_m, C_m, \mathcal{P}, \mathcal{Q}) \equiv \text{tr}\ \left[\mathcal{Q}\tilde{R} + (\tilde{A}\mathcal{Q} + \mathcal{Q}\tilde{A}^T + \gamma^{-2}\mathcal{Q}\tilde{R}\mathcal{Q} + \tilde{V})\mathcal{P}\right]$$

where $\mathcal{P} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ is a Lagrange multiplier. Setting $\partial\mathcal{L}/\partial\mathcal{Q} = 0$ yields (67). Partition $\mathcal{Q}$, $\mathcal{P} \in \mathbf{R}^{\tilde{n}\times\tilde{n}}$ as in (68) and define $\mathcal{P}\mathcal{Q} = Z$ as in (69). A straightforward calculation shows

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial A_m} &= 2(\mathcal{P}_{12}^T\,\mathcal{Q}_{12} + \mathcal{P}_2\,\mathcal{Q}_2),\\
\frac{\partial\mathcal{L}}{\partial B_m} &= 2(\mathcal{P}_{12}^T BV + \mathcal{P}_2 B_m V),\\
\frac{\partial\mathcal{L}}{\partial C_m} &= 2\big(RC_m\mathcal{Q}_2 - RC\mathcal{Q}_{12}\big)\\
&\quad + \gamma^{-2}\big[-RC(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12}Z_2)\\
&\quad + RC_m(\mathcal{Q}_{12}^T Z_{12} + Z_{12}^T\mathcal{Q}_{12} + \mathcal{Q}_2 Z_2 + Z_2^T\mathcal{Q}_2)\big].
\end{aligned}$$

Let A_f, B_f, C_f , R_f, V_f, and γ_f denote A, B, C, R, V, and γ in the above and define $A(\lambda)$, $B(\lambda)$, $C(\lambda)$, $R(\lambda)$, and $V(\lambda)$ as in (28) and

$$\gamma(\lambda) = \gamma_0 + \lambda(\gamma_f - \gamma_0)$$

and denote them by A, B, C, R, V, and γ respectively in the following. Let

$$\begin{aligned}
H_{A_m}(\theta,\lambda) &= \frac{\partial\mathcal{L}}{\partial A_m} = 2\big(\mathcal{P}_{12}^T\mathcal{Q}_{12} + \mathcal{P}_2\mathcal{Q}_2\big),\\
H_{B_m}(\theta,\lambda) &= \frac{\partial\mathcal{L}}{\partial B_m} = 2\big(\mathcal{P}_{12}^T B + \mathcal{P}_2 B_m\big)V,\\
H_{C_m}(\theta,\lambda) &= \frac{\partial\mathcal{L}}{\partial C_m} = 2R\big(C_m\mathcal{Q}_2 - C\mathcal{Q}_{12}\big)\\
&\quad + \gamma^{-2}\big[-RC(Z_1^T\mathcal{Q}_{12} + Z_{21}^T\mathcal{Q}_2 + \mathcal{Q}_1 Z_{12} + \mathcal{Q}_{12}Z_2)\\
&\quad + RC_m(\mathcal{Q}_{12}^T Z_{12} + Z_{12}^T\mathcal{Q}_{12} + \mathcal{Q}_2 Z_2 + Z_2^T\mathcal{Q}_2)\big],
\end{aligned} \tag{86}$$

where

$$\theta \equiv \begin{pmatrix} \text{Vec } (A_m) \\ \text{Vec } (B_m) \\ \text{Vec } (C_m) \end{pmatrix} \tag{87}$$

denotes the independent variables, $\mathcal{Q}$ and $\mathcal{P}$ satisfy respectively (14) and (67).

Define

$$\rho(\theta,\lambda) = \begin{pmatrix} \text{Vec } \left[H_{A_m}(\theta,\lambda)\right] \\ \text{Vec } \left[H_{B_m}(\theta,\lambda)\right] \\ \text{Vec } \left[H_{C_m}(\theta,\lambda)\right] \end{pmatrix}, \tag{88}$$

whose Jacobian matrix is

$$D\rho(\theta,\lambda) = \big(D_\theta\rho(\theta,\lambda), D_\lambda\rho(\theta,\lambda)\big).$$

Note that θ in (87) has $n_m^2 + n_m m + n_m l$ components, more than the minimal number $n_m m + n_m l$ of the input normal form and Ly formulations. Because of this over-parametrization, the Jacobian matrix of ρ is rank deficient. The homotopy map is thus defined as

$$\hat{\rho}(\theta,\lambda) = \lambda\rho(\theta,\lambda) + (1-\lambda)(\theta - \theta_0), \tag{89}$$

which guarantees a well conditioned full rank Jacobian matrix along the whole path except at the solution corresponding to $\lambda = 1$. The Jacobian matrix of $\hat{\rho}$ is given by

$$D\hat{\rho}(\theta,\lambda) = \big(\lambda D_\theta\rho(\theta,\lambda) + (1-\lambda)I, \quad \rho(\theta,\lambda) + \lambda D_\lambda\rho(\theta,\lambda) - (\theta - \theta_0)\big). \tag{90}$$

To find $D_\theta\rho(\theta,\lambda)$, define $\hat{H}_{A_m}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big)$, $\hat{H}_{B_m}\big(\mathcal{P}^{(j)}\big)$, and $\hat{H}_{C_m}(\mathcal{Q}^{(j)}, Z^{(j)})$ as in (81). For $\theta_j = \big(A_m\big)_{kl}$,

$$\begin{aligned} \frac{\partial H_{A_m}}{\partial (A_m)_{kl}} &= \hat{H}_{A_m}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big), \\ \frac{\partial H_{B_m}}{\partial (A_m)_{kl}} &= \hat{H}_{B_m}\big(\mathcal{P}^{(j)}\big), \\ \frac{\partial H_{C_m}}{\partial (A_m)_{kl}} &= \hat{H}_{C_m}\big(\mathcal{Q}^{(j)}, Z^{(j)}\big), \end{aligned} \tag{91}$$

for $\theta_j = \big(B_m\big)_{kl}$,

$$\begin{aligned} \frac{\partial H_{A_m}}{\partial (B_m)_{kl}} &= \hat{H}_{A_m}\big(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\big), \\ \frac{\partial H_{B_m}}{\partial (B_m)_{kl}} &= \hat{H}_{B_m}\big(\mathcal{P}^{(j)}\big) + 2\mathcal{P}_2 E^{(k,l)} V, \\ \frac{\partial H_{C_m}}{\partial (B_m)_{kl}} &= \hat{H}_{C_m}\big(\mathcal{Q}^{(j)}, Z^{(j)}\big), \end{aligned} \tag{92}$$

and for $\theta_j = (C_m)_{kl}$,

$$\begin{aligned}
\frac{\partial H_{A_m}}{\partial (C_m)_{kl}} &= \hat{H}_{A_m}\left(\mathcal{P}^{(j)}, \mathcal{Q}^{(j)}\right), \\
\frac{\partial H_{B_m}}{\partial (C_m)_{kl}} &= \hat{H}_{B_m}\left(\mathcal{P}^{(j)}\right), \\
\frac{\partial H_{C_m}}{\partial (C_m)_{kl}} &= \hat{H}_{C_m}\left(\mathcal{Q}^{(j)}, Z^{(j)}\right) + 2RE^{(k,l)}\mathcal{Q}_2 \\
&\quad + \gamma^{-2} RE^{(k,l)}\left(Z_{12}^T \mathcal{Q}_{12} + \mathcal{Q}_{12}^T Z_{12} + \mathcal{Q}_2^T Z_2 + Z_2^T \mathcal{Q}_2\right),
\end{aligned} \tag{93}$$

where $\mathcal{P}^{(j)}$ and $\mathcal{Q}^{(j)}$ can be obtained by solving the Lyapunov equation (75). The derivative of the homotopy map with respect to λ is given by (85) and (90).

Choose the initial γ so that γ_0^{-2} is approximately zero. The initial point $(\theta, \lambda) = (\theta_0, 0)$ is chosen so that the triple $\left((A_m)_0, (B_m)_0, (C_m)_0\right)$ is in balanced form and satisfies $\rho(\theta_0, 0) = 0$. This can be done as follows:

1) Obtain the initial reduced order model $\left((A_m)_0, (B_m)_0, (C_m)_0\right)_b$ in balanced form in the same way as for the input normal form approach.
2) Build θ_0 from $\left((A_m)_0, (B_m)_0, (C_m)_0\right)_b$ as described in (87).

The homotopy curve tracking computation is the same as described in Section 8.2.

11. H^2/H^∞ RESULTS.

The homotopy algorithms for solving the combined H^2/H^∞ model reduction problem in Chapters 8–10 were applied to all the problems in Chapter 5. The more interesting results are shown below.

For Example 2, a model of order $n_m = 2$ when $\gamma = 10$, with cost $\mathcal{J} = 0.678376$, is

$$A_m = \begin{pmatrix} -0.117649 & -0.493522 \\ 1.10166 & -0.785869 \end{pmatrix}, \quad B_m = \begin{pmatrix} -0.485076 \\ 1.25369 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} -0.751632 \\ -0.870253 \end{pmatrix}.$$

A model of order $n_m = 2$ when $\gamma = 1.0$, with cost $\mathcal{J} = 0.723313$, is

$$A_m = \begin{pmatrix} -0.112928 & -0.507912 \\ 1.10526 & -0.789927 \end{pmatrix}, \quad B_m = \begin{pmatrix} -0.475243 \\ 1.25692 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} -0.737429 \\ -0.896272 \end{pmatrix}.$$

For Example 6, the same ill conditioning problem exists for the input normal form approach. The results are obtained by the Ly form homotopy and are therefore expressed in Ly form. A model of order $n_m = 3$ at $\gamma = 1000$, with cost $\mathcal{J} = 28.7049$, is

$$A_m = \begin{pmatrix} 0 & 1 & 0 \\ -0.0487508 & -0.000488018 & 0 \\ 0 & 0 & -2.48940 \end{pmatrix},$$

$$B_m = \begin{pmatrix} 0.0250399 & 0.499928 \\ -0.0000456816 & -0.000210656 \\ 1.45729 & -0.747393 \end{pmatrix}, \quad C_m = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}.$$

A model of order $n_m = 3$ at $\gamma = 100$, with cost $\mathcal{J} = 31.0283$, is

$$A_m = \begin{pmatrix} 0 & 1 & 0 \\ -0.0487508 & -0.000488063 & 0 \\ 0 & 0 & -2.49122 \end{pmatrix},$$

$$B_m = \begin{pmatrix} 0.0249908 & 0.499976 \\ -0.0000485435 & -0.000207832 \\ 1.58384 & -0.812299 \end{pmatrix}, \quad C_m = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}.$$

For Example 8, a model of order $n_m = 2$ at $\gamma = 10^4$, with cost $\mathcal{J} = 0.0269276$, is

$$A_m = \begin{pmatrix} -0.532330 & -0.598751 \\ 3.80077 & -4.81512 \end{pmatrix}, \quad B_m = \begin{pmatrix} 1.03182 \\ -3.10326 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} 0.588704 \\ 0.278923 \end{pmatrix}.$$

A model of order $n_m = 2$ at $\gamma = 1.0$, with cost cost $\mathcal{J} = 0.0271171$, is

$$A_m = \begin{pmatrix} -0.532190 & -0.600707 \\ 3.80359 & -4.81899 \end{pmatrix}, \quad B_m = \begin{pmatrix} 1.03169 \\ -3.10451 \end{pmatrix}, \quad C_m^T = \begin{pmatrix} 0.588683 \\ 0.279766 \end{pmatrix}.$$

A problem suggested by D. S. Bernstein is

$$A = \begin{pmatrix} -10 & -45 & -120 & -210 & -250 & -210 & -120 & -45 & -10 & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix},$$

$$B = (1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0)^T,$$

$$C = (0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1),$$

which has the transfer function

$$H(s) = \frac{1}{(1+s)^{10}}.$$

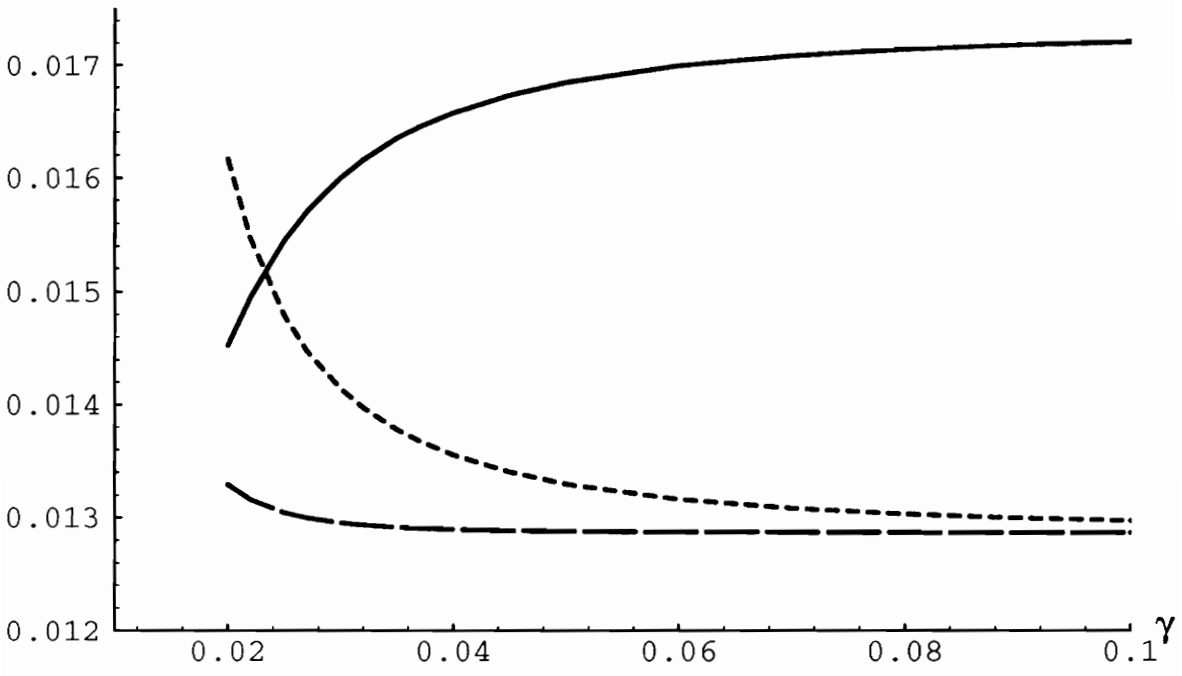

Fig. 3. $\|H(s) - H_m(s)\|_\infty$, $100\,\mathcal{J}$, and $100\,J$ versus γ.

For $n_m = 4$, the solutions are obtained for $\gamma \geq 0.02$ using the input normal form approach. Let $H_m(s)$ be the transfer function of the reduced order model obtained by minimizing $\mathcal{J}$. In Fig. 3, $\|H(s) - H_m(s)\|_\infty$, $100\,\mathcal{J}$ (dotted line), and $100\,J$ (dashed line) are plotted against γ. As shown in the figure, as γ decreases, $\|H(s) - H_m(s)\|_\infty$ also decreases while both $\mathcal{J}$ and J increase. As can be seen from the figure, $\mathcal{J}$ is a close bound for J until γ becomes very small. To show the tradeoff between the H^2 cost and the H^∞ error $\|H(s) - H_m(s)\|_\infty$, it is useful to plot $\|H(s) - H_m(s)\|_\infty$ against J (with γ as the parameter of the curve), as shown in Fig. 4.

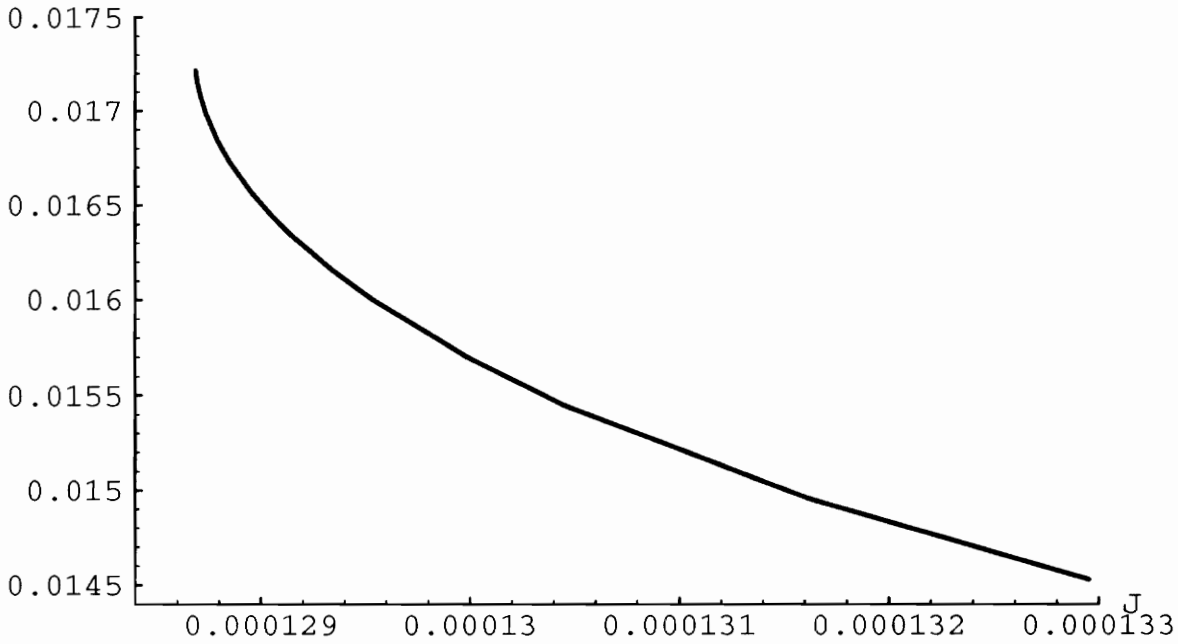

Fig. 4. $\|H(s) - H_m(s)\|_\infty$ versus J.

The same conclusion can be drawn for the input normal form homotopy algorithm and the Ly form homotopy algorithm for solving the combined H^2/H^∞ model reduction problem, i.e., they are very efficient but they may fail to exist or be very ill conditioned.

12. CONCLUSIONS.

The H^2 optimal model order reduction problem has been under intense study both theoretically and numerically. There are various approaches for solving this problem. The most significant contribution of the input normal form and the Ly form homotopy approaches is that only the minimal number of degrees of freedom is used, which gives rise to a very efficient numerical algorithm. However, the input normal form and Ly form can be ill conditioned.

The combined H^2/H^∞ model order reduction problem, resulting from the addition of a H^∞ constraint to the H^2 optimal model order reduction problem, is more complicated and more interesting. There are theoretical formulations for this problem, but there have been no serious numerical studies. Chapters 8–10 represent an attempt to design practical and efficient numerical algorithms for solving the combined H^2/H^∞ model order reduction problem. Both the input normal form homotopy and the Ly form homotopy algorithms are applicable and very efficient, while the same ill conditioning as in the homotopy algorithms for the H^2 optimal model order reduction problem may occur.

To understand the engineering implications of a H^∞ constraint on a system, more studies on the results of the combined H^2/H^∞ model order reduction problem are needed. This will be a topic for future work.

In the numerical algorithm for the combined H^2/H^∞ model order reduction problem, solving a Riccati equation takes a significant amount of computer time. To design and implement a parallel algorithm for solving the Riccati equation is necessary to improve the performance of the homotopy algorithms.

In the homotopy algorithms for both the H^2 optimal model order reduction problem and the combined H^2/H^∞ model order reduction problem, most of the time is devoted to tangent vector computation which can be divided into independent tasks. Therefore a parallel implementation of the whole homotopy algorithm can be considered in the future.

The main conclusion from this study is that the more degrees of freedom that a formulation uses, the more robust is the resulting numerical algorithm. Both the input normal form and Ly form homotopies are very efficient for both the H^2 and the combined H^2/H^∞ model order reduction problems. However, they may fail to exist or be very ill conditioned. The over-parametrization formulation solves the ill conditioning issue, but introduces singularity at the solution and may fail for a high dimensional system which will inevitably have a higher order singularity at the solution.

REFERENCES

[1] V. M. ADAMJAN, D. Z. AROV AND M. G. KREIN, "Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur-Takagi problem," *Math. USSR-Sbornik,* vol. 15, pp. 31–73, 1971.

[2] J. A. BALL AND A. C. M. RAN, "Optimal Hankel norm model reductions and Wiener-Hopf factorization I: The canonical case," *SIAM J. Contr. Optim.*, vol. 25, pp. 362–382, 1987.

[3] R. H. BARTELS AND G. W. STEWART, "Solution of the matrix equation $A\,X + X\,B = C$," *Comm. of the ACM* vol. 15, pp. 820–826, 1972.

[4] D. S. BERNSTEIN, private communication.

[5] D. S. BERNSTEIN AND W. M. HADDAD, "LQG control with an H_∞ performance Bound: a Riccati equation Approach," *IEEE Trans. Autom. Contr.*, vol. AC-34, pp. 293–305, 1989.

[6] S. BOYD, V. BALAKRISHNAN, AND P. KABAMBA, "A bisection method for computing the H_∞ norm of a transfer matrix and related problems," *Math. Contr., Sig., and Sys.*, vol. 2, pp. 207–219, 1989.

[7] R. F. CURTAIN, "L_∞-approximations of complex functions and robust controllers for large flexible space structures," NASA Technical report, PB88–186226, TW–281, 1987.

[8] L. D. DAVIS, E.G. COLLINS, JR., AND S. A. HODEL, "A parameterization of minimal plants," *Proc. 1992 American Contr. Conf.,* Chicago, IL, pp. 355–356, June 1992.

[9] D. F. ENNS, "Model reduction for control system design," Ph. D. Dissertation, Stanford University, 1984.

[10] J. D. GARDINER AND A. J. LAUB, "Parallel algorithms for algebraic Riccati equations," *Int. J. Control,* vol. 54, pp. 1317–1333, 1991.

[11] Y. Z. GE, E. G. COLLINS, L. T. WATSON, AND L. D. DAVIS, "Minimal parameter homotopies for the L^2 optimal model order reduction problem," *Tech. Rep. TR92–36,* Dept. Computer Sci., Virginia Polytechnic Inst. & State Univ., Blacksburg, VA, 1992.

[12] K. GLOVER, "All optimal Hankel-Norm approximations of linear multivariable systems and their L^∞-error bounds," *Int. J. Control,* vol. 39, pp. 1115–1193, 1984.

[13] W. M. HADDAD AND D. S. BERNSTEIN, "Combined L^2/H^∞ model reduction," *Int. J. Contr.*, vol. 49, pp. 1523–1535, 1989

[14] J. HICKIN AND N. K. SINHA, "Model reduction for linear multivariable systems," *IEEE Trans. Autom. Contr.*, vol. AC-25, pp. 1121–1127, 1980.

[15] D. C. HYLAND AND D. S. BERNSTEIN, "The optimal projection equations for fixed-order dynamic compensation," *IEEE Trans. Autom. Contr.*, vol. AC-29, pp. 1034–1037, 1984.

[16] ———, "The optimal projection equations for model reduction and the relationships among the methods of Wilson, Skelton, and Moore," *IEEE Trans. Autom. Contr.*, vol. AC-30, pp. 1201–1211, 1985.

[17] P. T. KABAMBA, "Balanced forms: canonicity and parameterization," *IEEE Trans. Autom. Contr.*, vol. AC-30, pp. 1106–1109, 1985.

[18] ———, "Balanced gains and their significance for L^2 model reduction," *IEEE Trans. Autom. Contr.*, vol. AC-30, pp. 690–693, 1985.

[19] D. KAVRANOĞLU AND M. BETTAYCB, "Characterization of the solution to the optimal H^∞ model reduction problem," *Sys. & Contr. lett.*, vol. 20, pp. 99-107, 1993.

[20] H. KWAKERNAAK AND R. SIVAN, *Linear Optimal Control Systems,* New York: Wiley-Interscience, 1972.

[21] A. J. LAUB, "A Schur method for solving algebraic Riccati equations," *IEEE Trans. Autom. Contr.*, vol. AC-24, pp. 913–921, 1979.

[22] U.-L. LY, A. E. BRYSON AND R. H. CANNON, "Design of low-order compensators using parameter optimization," *Automatica,* vol. 21, pp. 315–318, 1985.

[23] B. C. MOORE, "Principal component analysis in linear systems: controllability, observability, and model reduction," *IEEE Trans. Autom. Contr.*, vol. AC-26, pp. 17–32, 1981.

[24] A. P. MORGAN, A. J. SOMMESE, "Computing singular solutions to nonlinear analytic systems", *Research Pub. GMR-6770, General Motors Research Laboratories,* Warren, Mich. 48090, 1989.

[25] A. P. MORGAN, A. J. SOMMESE, "Computing singular solutions to polynomial systems", *Research Pub. GMR-7020, General Motors Research Laboratories,* Warren, Mich. 48090, 1989.

[26] A. P. MORGAN, A. J. SOMMESE, AND C. W. WAMPLER, "A power series method for computing singular solutions to nonlinear analytic systems", *Num. Math.,* to appear.

[27] D. MUSTAFA AND K. GLOVER, "Minimum Entropy H_∞ Control," Springer–Verlag, Berlin, 1991.

[28] S. RICHTER, "Reduced-order control design via the optimal projection approach: A homotopy algorithm for global optimality," *Proc. Sixth VPI&SU Symp. Dyn. Contr. Large Str.,* Blacksburg, VA, June 1987.

[29] ———, "A homotopy algorithm for solving the optimal projection equations for fixed-order dynamic compensation: Existence, convergence and global optimality," *Proc. American Contr. conf.,* Minneapolis, MN, pp. 1527-1531, June 1987.

[30] S. RICHTER AND E. G. COLLINS, JR., "A homotopy algorithm for reduced order compensator design using the optimal projection equations," *Proc. IEEE Conf. Dec. Contr.,* Tampa, FL, pp. 506–511, Dec. 1989.

[31] M. G. SAFONOV AND R. Y. CHIANG, "A Schur method for balanced-truncation model reduction," *IEEE Trans. Autom. Contr.,* vol. AC–34, pp. 729–733, 1989.

[32] R. E. SKELTON AND P. KABAMBA, "Comments on 'Balanced gains and their significance for L^2 model reduction'," *IEEE Trans. Autom. Contr.,* vol. AC–31, pp. 796–797, 1986.

[33] R. E. SKELTON AND A. YOUSUFF, "Component cost analysis of large scale systems," *Int. J. Contr.,* vol. 37, pp. 285–304, 1983.

[34] C. DE VILLEMAGNE AND R. E. SKELTON, "Model reductions using a projection formulation," *Int. J. Contr.,* vol. 46, pp. 2141–2169, 1987.

[35] L. T. WATSON, "Globally convergent homotopy methods: A tutorial," *Appl. Math. Comput.* vol. 31BK, pp. 369–396, 1989.

[36] L. T. WATSON, "Globally convergent homotopy algorithms for nonlinear systems of equations," *Nonlinear Dynamics* vol. 1, pp. 143–191, 1990.

[37] L. T. WATSON, "Numerical linear algebra aspects of globally convergent homotopy methods," *SIAM Rev.* vol. 28, pp. 529–545, 1986.

[38] L. WATSON, S. C. BILLUPS AND A. P. MORGAN, "HOMPACK: a suite of codes for globally convergent homotopy algorithms," *ACM Trans. Math. Software,* vol. 13, pp. 281–310, 1987.

[39] D. A. WILSON, "Optimum solution of model-reduction problem," *Proc. IEE,* vol. 117, pp. 1161–1165, 1970.

[40] W. M. WONHAM, *Linear Multivariable Control: A Geometric Approach* New York: Springer-Verlag, 1974.

[41] D. ŽIGIĆ, "Homotopy methods for solving the optimal projection equations for the reduced order model problem," M.S. thesis, Dept. of Computer Science, Virginia Polytechnic Institute and State Univ., Blacksburg, VA, June 1991.

[42] D. ŽIGIĆ, L. WATSON, E. G. COLLINS, JR., AND D. S. BERNSTEIN, "Homotopy approaches to the H^2 reduced order model problem," TR 91-24, Dept. of Computer Science, Virginia Polytechnic Institute and State Univ., 1991.

[43] ———, "Homotopy methods for solving the optimal projection equations for the H^2 reduced order model problem," TR 91-13, Dept. of Computer Science, Virginia Polytechnic Institute and State Univ., 1991.

Appendix A.

In this appendix explicit expressions for the derivatives

$$\tilde{A}^{(j)} = \frac{\partial \tilde{A}}{\partial \theta_j}, \qquad \tilde{V}^{(j)} = \frac{\partial \tilde{V}}{\partial \theta_j}, \qquad \tilde{R}^{(j)} = \frac{\partial \tilde{R}}{\partial \theta_j}, \qquad M_c^{(j)} = \frac{\partial M_c}{\partial \theta_j}, \qquad M_o^{(j)} = \frac{\partial M_o}{\partial \theta_j},$$

where $\theta_j = (B_m)_{kl}$ or $(C_m)_{kl}$, are presented. Since

$$\tilde{A} = \begin{pmatrix} A & 0 \\ 0 & A_m \end{pmatrix}, \qquad \tilde{V} = \begin{pmatrix} BVB^T & BVB_m^T \\ B_mVB^T & B_mVB_m^T \end{pmatrix}, \qquad \tilde{R} = \begin{pmatrix} C^TRC & -C^TRC_m \\ -C_m^TRC & C_m^TRC_m \end{pmatrix},$$

the derivatives of $\tilde{R}$ and $\tilde{V}$ with respect to B_m and C_m can be obtained easily as:

$$\frac{\partial \tilde{R}}{\partial B_{m,kl}} = 0,$$

$$\frac{\partial \tilde{R}}{\partial C_{m,kl}} = \begin{pmatrix} 0 & -C^TR\dfrac{\partial C_m}{\partial C_{m,kl}} \\ -\dfrac{\partial C_m^T}{\partial C_{m,kl}}RC & \dfrac{\partial C_m^T}{\partial C_{m,kl}}RC_m + C_m^TR\dfrac{\partial C_m}{\partial C_{m,kl}} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & -C^TRE^{(k,l)} \\ -E^{(l,k)}RC & E^{(l,k)}RC_m + C_m^TRE^{(k,l)} \end{pmatrix},$$

$$\frac{\partial \tilde{V}}{\partial B_{m,kl}} = \begin{pmatrix} 0 & BVE^{(l,k)} \\ E^{(k,l)}VB^T & E^{(k,l)}VB_m^T + B_mVE^{(l,k)} \end{pmatrix},$$

$$\frac{\partial \tilde{V}}{\partial C_{m,kl}} = 0.$$

Differentiating $\tilde{A}$ with respect to $(B_m)_{kl}$ and $(C_m)_{kl}$ yields

$$\frac{\partial \tilde{A}}{\partial (B_m)_{kl}} = \begin{pmatrix} 0 & 0 \\ 0 & \dfrac{\partial A_m}{\partial (B_m)_{kl}} \end{pmatrix}, \qquad \frac{\partial \tilde{A}}{\partial (C_m)_{kl}} = \begin{pmatrix} 0 & 0 \\ 0 & \dfrac{\partial A_m}{\partial (C_m)_{kl}} \end{pmatrix}.$$

Define α_{kl} and β_{kl} as

$$\alpha_{kl} \equiv -2(B_mV)_{kl}(C_m^TRC_m)_{kk}(B_mVB_m^T)_{kk}^{-2}, \qquad \beta_{kl} \equiv 2(RC_m)_{kl}(B_mVB_m^T)_{ll}^{-1}.$$

Then differentiating (18), we have

$$\frac{\partial \omega_i}{\partial (B_m)_{kl}} = \alpha_{kl}\delta_{ki}, \qquad \frac{\partial \omega_i}{\partial (C_m)_{kl}} = \beta_{kl}\delta_{li},$$

where $\delta_{ij} = \begin{cases} 1 & i = j, \\ 0 & i \neq j. \end{cases}$ is the Kronecker delta. Differentiating $(A_m)_{ii}$ and $(A_m)_{ij}$ in (18) yields

$$\frac{\partial A_{m,ii}}{\partial (B_m)_{kl}} = -(B_m V)_{kl}\delta_{ki},$$

$$\begin{aligned}\frac{\partial A_{m,ij}}{\partial (B_m)_{kl}} &= -\frac{A_{m,ij}}{(\omega_j - \omega_i)}\left[\frac{\partial \omega_j}{\partial (B_m)_{kl}} - \frac{\partial \omega_i}{\partial (B_m)_{kl}}\right] \\ &\quad - (\omega_j - \omega_i)^{-1}\left[\frac{\partial \omega_j}{\partial (B_m)_{kl}}(B_m V B_m^T)_{ij} + \omega_j \frac{\partial}{\partial (B_m)_{kl}}(B_m V B_m^T)_{ij}\right] \\ &= -(\delta_{kj} - \delta_{ki})\alpha_{kl}(\omega_j - \omega_i)^{-1} A_{m,kl} \\ &\quad - (\omega_j - \omega_i)^{-1}\left[\alpha_{kl}\delta_{kj}(B_m V B_m^T)_{ij} - \omega_j \delta_{ki}(V B_m^T)_{lj} - \omega_j (B_m V)_{il}\delta_{jk}\right],\end{aligned}$$

$$\frac{\partial A_{m,ii}}{\partial (C_m)_{kl}} = 0,$$

$$\begin{aligned}\frac{\partial A_{m,ij}}{\partial (C_m)_{kl}} &= (\omega_j - \omega_i)^{-1}\left[\frac{\partial}{\partial (C_m)_{kl}}(C_m^T R C_m)_{ij} - \frac{\partial \omega_j}{\partial (C_m)_{kl}}(B_m V B_m^T)_{ij}\right] \\ &\quad - \frac{A_{m,ij}}{(\omega_j - \omega_i)}\left[\frac{\partial \omega_j}{\partial (C_m)_{kl}} - \frac{\partial \omega_i}{\partial (C_m)_{kl}}\right] \\ &= (\omega_j - \omega_i)^{-1}\left[\delta_{li}(R C_m)_{kj} + \delta_{lj}(C_m^T R)_{ik} - \beta_{kl}\delta_{lj}(B_m V B_m^T)_{ij}\right] \\ &\quad - (\delta_{jl} - \delta_{il})\beta_{kl}(\omega_j - \omega_i)^{-1} A_{m,ij}.\end{aligned}$$

Differentiating (26) with respect to $(B_m)_{kl}$ and $(C_m)_{kl}$, we obtain

$$\begin{aligned}\frac{\partial M_{o,ii}}{\partial (B_m)_{kl}} &= (A_{m,ii})^{-2}\frac{\partial A_{m,ii}}{\partial (B_m)_{kl}}\sum_{j\neq i}^{n_m} A_{m,ij} M_{o,ji} \\ &\quad - (A_{m,ii})^{-1}\sum_{j\neq i}^{n_m}\left[\frac{\partial A_{m,ij}}{\partial (B_m)_{kl}} M_{o,ji} + A_{m,ij}\frac{\partial M_{o,ji}}{\partial (B_m)_{kl}}\right], \\ \frac{\partial M_{o,ij}}{\partial (B_m)_{kl}} &= \frac{S_{ij}^{(j)} - S_{ji}^{(j)}}{2(\omega_j - \omega_i)} - \frac{M_{o,ij}}{(\omega_j - \omega_i)}\left[\frac{\partial \omega_j}{\partial (B_m)_{kl}} - \frac{\partial \omega_i}{\partial (B_m)_{kl}}\right] \\ &= \frac{S_{ij}^{(j)} - S_{ji}^{(j)}}{2(\omega_j - \omega_i)} - \frac{M_{o,ij}\alpha_{kl}(\delta_{jk} - \delta_{ik})}{(\omega_j - \omega_i)}, \\ \frac{\partial M_c}{\partial (B_m)_{kl}} &= -\frac{1}{2}\left[S^{(j)} + \frac{\partial \Omega}{\partial (B_m)_{kl}} M_o + \Omega \frac{\partial M_o}{\partial (B_m)_{kl}}\right],\end{aligned}$$

$$\frac{\partial M_{o,ii}}{\partial (C_m)_{kl}} = -(A_{m,ii})^{-1} \sum_{j \neq i}^{n_m} \left[\frac{\partial A_{m,ij}}{\partial (C_m)_{kl}} M_{o,ji} + A_{m,ij} \frac{\partial M_{o,ji}}{\partial (C_m)_{kl}} \right],$$

$$\begin{aligned} \frac{\partial M_{o,ij}}{\partial (C_m)_{kl}} &= \frac{S_{ij}^{(j)} - S_{ji}^{(j)}}{2(\omega_j - \omega_i)} - \frac{M_{o,ij}}{(\omega_j - \omega_i)} \left[\frac{\partial \omega_j}{\partial (C_m)_{kl}} - \frac{\partial \omega_i}{\partial (C_m)_{kl}} \right] \\ &= \frac{S_{ij}^{(j)} - S_{ji}^{(j)}}{2(\omega_j - \omega_i)} - \frac{M_{o,ij} \beta_{kl} (\delta_{jl} - \delta_{il})}{(\omega_j - \omega_i)}, \end{aligned}$$

$$\frac{\partial M_c}{\partial (C_m)_{kl}} = -\frac{1}{2} \left[S^{(j)} + \frac{\partial \Omega}{\partial (C_m)_{kl}} M_o + \Omega \frac{\partial M_o}{\partial (C_m)_{kl}} \right],$$

where

$$S^{(j)} = 2(\tilde{P}_{12}^{T\,(j)} \tilde{Q}_{12} + \tilde{P}_{12}^{T} \tilde{Q}_{12}^{(j)} + \tilde{P}_{2}^{(j)} \tilde{Q}_{2} + \tilde{P}_{2} \tilde{Q}_{2}^{(j)}).$$

Formula in Appendix A in [41] are used in the above derivations.

Appendix B.

```
*
* This is the header file inc.h which defines the global variables
*
* Integers:
* ld, ld2 -- n + 1 , n + nm + 1
* n, nm, m, l -- dimensions of systems
* nn -- nm*m + nm*l, variables of the problem
* nt -- n + nm , dimension of the augmented system
* trace -- flag for writing output
* z -- ld * ld
*
* Double precision variables:
* a, b, c -- (A, B, C), the given system
* a0, b0, c0 -- (A0, B0, C0), the initial (A, B, C)
* am, bm, cm -- (Am, Bm, Cm)
* ami, bmi, cmi -- (Am, Bm, Cm) in input normal form
* v,r, bv, rc, bvbt, ctrc -- V, R, BV, RC, BVB', C' RC
* bmv, rcm, bvbtm, ctrcm -- BmV, RCm, BmVBm', Cm' RCm
* adot, bdot, cdot -- A - A0, B - B0, C - C0
* aal, bbl, ccl -- A0 + lambda(A-A0),B0 + lambda(B-B0),C0 + lambda(C-C0)
* hbm, hcm -- the homotopy
* f10, f20 -- the initial homotopy
* qt12, pt12 -- Q12, P12
* mo, mc, jam -- Lagrange multipliers Mo and Mc and S defined in (26).
* ome, delta -- Omega, and the unit matrix
* as, ua --  upper shur form of A(lambda), ua is the transformation
* at, uat --  upper shur form of A'(lambda), uat is the transformation
* ams, vam --  lower shur form of Am, vam is the transformation
* amts, vamt --  lower shur form of Am', vamt is the transformation
*
* as, ua, at, uat, ams, vam, amts, vamt --
*    are defined in RRHO called by RHOJAC at each step of curve tracking
*    and are saved for use in JACLAM, JACAM,JACBM, JACCM to avoid
*    computing the same decomposition again.
*
*
      integer ld,ld2,n,m,l,nm,nn,nt,trace,z
c ld = n + 1
      parameter(ld=8)
c ld2 = n + nm + 1 = nt + 1
      parameter(ld2=12)
      double precision a(ld,ld),b(ld,ld),c(ld,ld),
     &                 a0(ld,ld),b0(ld,ld),c0(ld,ld)
      double precision am(ld,ld),bm(ld,ld),cm(ld,ld),
     &                 ami(ld,ld),bmi(ld,ld),cmi(ld,ld)
      double precision v(ld,ld),bvbt(ld,ld),bv(ld,ld),
     &                 bmv(ld,ld),bvbtm(ld,ld)
      double precision r(ld,ld),ctrc(ld,ld),rc(ld,ld),
     &                 rcm(ld,ld), ctrcm(ld,ld)
      double precision adot(ld,ld),bdot(ld,ld),cdot(ld,ld),
     &                  aal(ld,ld),bbl(ld,ld),ccl(ld,ld),
     &                 qt12(ld,ld),pt12(ld,ld),delta(ld,ld)
      double precision hbm(ld,ld),hcm(ld,ld),mo(ld,ld),mc(ld,ld),
```

```
     &              jam(ld,ld),ome(ld,ld),f10(ld,ld),f20(ld,ld)
      double precision as(ld,ld),ats(ld,ld),ams(ld,ld),amts(ld,ld),
     &              ua(ld,ld),uat(ld,ld),vam(ld,ld),vamt(ld,ld)
      common/int/n,m,l,nm,nt,nn,trace,z
      common/sys/a,b,c,a0,b0,c0,am,bm,cm,ami,bmi,cmi,
     &          v,bvbt,bv,bmv,bvbtm,r,ctrc,rc,rcm,ctrcm,
     &          qt12,pt12,as,ats,ams,amts,ua,uat,vam,vamt,
     &          adot,bdot,cdot,delta,aal,bbl,ccl,
     &          f10,f20,hbm,hcm,mo,mc,jam,ome
*
*.....................................................................
* This is the main program.  It reads in data "in1.dat"
*   and calls track.  The output will be written to "out1.dat" if
*   the "trace" is set bigger than 1.
* The program implements the input normal form homotopy algorithm
*   to solve the H^2 optimal model order reduction problem.
*
* MAIN PROGRAM
      include 'inc.h'
*
      integer ipivot(20)
      parameter(lwa1 = 12000)
      parameter(lwa2 = 50000)
      real timing(2)
      double precision wa1(lwa1),wa2(lwa2),x(20)
*
      open(1,file='in1.dat')
      trace = 2
      open(trace,file='out1.dat')
*
      call dtime(timing)
      z = ld*ld
*
* Read input parameters
*
      read(1,*)n,nm,m,l
      nt = n+nm
      nn = nm*m +l*nm
      n1 = nn+1
      if (trace .gt. 0) then
        write(trace,100)n,nm,m,l
100     format(/'n=',i2,'   nm=',i2,'   m=',i2,'   l=',i2)
      endif
*
      call read2(ld,n,m,l,trace,a,b,c)
*
* Set V, R
*
      call setmat(ld,0.0d0,v,m,m)
      do 101 i = 1,m
101     v(i,i) = 1.d0
      call setmat(ld,0.d0,r,l,l)
      do 102 i = 1,l
102     r(i,i) = 1.0d0
*
* Perform tracking
```

```
*
      if (lwa1 .lt. 15*ld*ld) then
        print*, 'Error in memory, try to make wa1 bigger'
        stop
      endif
      if (lwa2 .lt. 11*(nn+1)+nn*(nn+2)+nn*(nn+1 3)/2) then
        print*, 'Error in memory, try to make wa2 bigger'
        stop
      endif
*
      call track(x,ipivot,wa1(1),wa1(1+z),wa1(1+2*z),wa1(1+3*z),
     *          wa1(1+4*z),wa1(1+5*z),wa2(1),wa2(n1+1),wa2(2*n1+1),
     *          wa2(3*n1+1),wa2(4*n1+1),wa2(5*n1+1),wa2(6*n1+1),
     *          wa2(7*n1+1),wa2(8*n1+1),wa2( 9*n1+1),wa2(10*n1+1),
     *          wa2(10*n1+nn*(nn+2)))
*
* Evaluate solution
*
      call evaluate(x(2),wa1(1),wa1(ld+1),wa1(2*ld+1),wa1(2*ld+z+1),
     &        wa1(2*ld+2*z+1),wa1(2*ld+3*z+1),wa1(2*ld+4*z+1),
     &        wa1(2*ld+5*z+1))
      call dtime(timing)
      write(2,201) timing(1),timing(2)
      write(6,201) timing(1),timing(2)
201   format(/'The CPU time',1x,f19.12,2x,'The system time ',f19.12)
*
      close(1)
      close(2)
*
      stop
      end
*
*.....................................................................
*
      subroutine track(x,ipivot,temp1,temp2,temp3,temp4,temp5,wa,
     &             alpha,aa,tz,w2,wp,yold,yp,ypold,z0,z1,qr,work)
*
* Subroutine TRACK1 solves the optimal projection equations for the
* reduced order model problem. It is based on the normal flow algorithm
* for dense Jacobian matrices. The HOMPACK subroutine FIXPNF is used
* for that purpose.
*
* Input parameters:
* arcre - relative error allowed of normal flow iterations
*        along the curve
* ansre - relative error of the answer at lambda = 1
*
* Output parameters:
* x - final point (lambda,Bm, Cm)
* iflag - flag returned by FIXPNF
*
* Working storage:
* ipivot(nn+1) - integer working array
* ip1 - array used for hompack to pass parameter (not used)
* yold,yp,ypold - working arrays of size nn+1
* alpha,tz,w2,wp,z0,z1 - working arrays of size nn+1
```

```
* qr(nn,nn+2) - working matrix
* work(nn*(nn+13)/2) - working array *
*
      include 'inc.h'
*
      integer iflag,ip(1),ipivot(nn+1),nfe
      double precision alpha(nn+1),ansae,ansre,arclen,arcae,arcre
      double precision qr(nn,nn+2),sspar(8),aa(nn),temp4(ld,ld)
      double precision tz(nn+1),w2(nn+1),work(nn*(nn+13)/2)
      double precision wp(nn+1),wa(25*z),x(nn+1),temp5(ld,ld)
      double precision temp1(ld,ld),temp2(ld,ld),temp3(ld,ld)
      double precision yold(nn+1),yp(nn+1),ypold(nn+1),z0(nn+1),z1(nn+1)
*
* Construct the starting point (Bm, Cm)
*
      call initp(wa(1),wa(z+1),wa(2*z+1),wa(3*z+1),wa(4*z+1),wa(5*z+1),
     &          wa(6*z+1),wa(7*z+1),wa(8*z+1))
*
*  The eigenvalues of A0 and Ami
*
      if (trace .ne. 0) write(trace,254)
254   format(/'eigenvalues of A0')
      call eig(ld,n,trace,a0,temp1,temp2,temp3,temp4,wa(1),wa(1+ld))
      if (trace .ne. 0) write(trace,255)
255   format(/'eigenvalues of Ami')
      call eig(ld,nm,trace,ami,temp1,temp2,temp3,temp4,wa(1),wa(1+ld))
*
* Transform the input normal form (Bm ,Cm) to array x.
*
      call xmat(ld,2,nm,m,bmi,x(2))
      call xmat(ld,2,1,nm,cmi,x(2+nm*m))
      x(1) = 0.d0
*
* Compute BV, BmV, RC, RCm, Bm V Bm^T, Cm^t R Cm
*
      call mulmat(ld,b0,v,bv,n,m,m,1)
      call mulmat(ld,bv,b0,bvbt,n,n,m,3)
      call mulmat(ld,bmi,v,bmv,nm,m,m,1)
      call mulmat(ld,r,c0,rc,l,n,l,1)
      call mulmat(ld,c0,rc,ctrc,n,n,l,2)
      call mulmat(ld,r,cmi,rcm,l,nm,l,1)
      call mulmat(ld,bmv,bmi,bvbtm,nm,nm,m,3)
      call mulmat(ld,cmi,rcm,ctrcm,nm,nm,l,2)
*
* Set aal,bbl,ccl be a0,b0,c0
*
      call scmat(ld,1.0d0,a0,aal,n,n)
      call scmat(ld,1.0d0,b0,bbl,n,m)
      call scmat(ld,1.0d0,c0,ccl,l,n)
*
* Compute delta,omega
*
      call setmat(ld,0.0d0,delta,nm,nm)
      call setmat(ld,0.d0,ome,nm,nm)
      do 201 i=1,nm
       delta(i,i)=1.0d0
```

```
      ome(i,i) = ctrcm(i,i)/bvbtm(i,i)
201   continue
*
* Compute Q12, P12
*
      call mulmat(ld,bv,bmi,temp2,n,nm,m,3)
      call scmat(ld,-1.0d0,temp2,temp1,n,nm)
      call scmat(ld,1.0d0,a0,temp2,n,n)
      call trans(ld,ami,temp3,nm,nm)
      call axpxb(temp2,temp4,n,ld,ld,temp3,temp5,nm,ld,ld,temp1,ld,
     &          1.0d-18,1.0d-18,ind)
      call scmat(ld,1.0d0,temp1,qt12,n,nm)
*
      call mulmat(ld,c0,rcm,temp2,n,nm,l,2)
      call scmat(ld,1.0d0,temp2,temp1,n,nm)
      call trans(ld,a0,temp2,n,n)
      call scmat(ld,1.0d0,ami,temp3,nm,nm)
      call axpxb(temp2,temp4,n,ld,ld,temp3,temp5,nm,ld,ld,temp1,ld,
     &          1.0d-18,1.0d-18,ind)
      call scmat(ld,1.0d0,temp1,pt12,n,nm)
*
*  Compute the average of the two just solved equations
*
      call mulmat(ld,a0,qt12,temp1,n,nm,n,1)
      call mulmat(ld,qt12,ami,temp2,n,nm,nm,3)
      call addmat(ld,temp1,temp2,temp3,n,nm,1)
      call mulmat(ld,bv,bmi,temp2,n,nm,m,3)
      call addmat(ld,temp3,temp2,temp1,n,nm,1)
*
      call mulmat(ld,a0,pt12,temp2,n,nm,n,2)
      call mulmat(ld,pt12,ami,temp3,n,nm,nm,1)
      call addmat(ld,temp2,temp3,temp4,n,nm,1)
      call mulmat(ld,rc,cmi,temp3,n,nm,l,2)
      call addmat(ld,temp4,temp3,temp2,n,nm,3)
*
      f10av  = 0.d0
      f20av = 0.d0
      do 611 i = 1,n
        do 611 j = 1,nm
          f10av = f10av + dabs(temp1(i,j))
          f20av = f20av + dabs(temp2(i,j))
611   continue
      f10av = f10av / dble(n*nm)
      f20av = f20av / dble(n*nm)
      write(2,271)f10av,f20av
      write(6,271)f10av,f20av
271   format(/'In track1 equation average =',d20.12)
*
* Compute S
*
      call mulmat(ld,pt12,qt12,temp1,nm,nm,n,2)
      call addmat(ld,temp1,ome,jam,nm,nm,1)
*
* Compute Mo
*
      do 301 j=1,nm
```

```
        do 301 i=j+1, nm
         mo(i,j) =  (jam(i,j)-jam(j,i)) / (ome(j,j)-ome(i,i))
         mo(j,i) = mo(i,j)
301   continue
     do 401 i = 1,nm
        mo(i,i) = 0.0d0
        do 402 j=1,i-1
402         mo(i,i) = mo(i,i) + ami(i,j)*mo(j,i)
        do 403 j=i+1,nm
403         mo(i,i) = mo(i,i) + ami(i,j)*mo(j,i)
        mo(i,i) = - mo(i,i) /(ami(i,i))
401   continue
*
* Compute Mc
*
      do 501 i=1,nm
         do 501 j=1,nm
          temp1(i,j) = ome(i,i) * mo(i,j)
501    continue
*
      call addmat(ld,temp1,jam,mc,nm,nm,4)
*
* Compute the homotopy map
*
* H_Bm
*
     call mulmat(ld,pt12,bv,temp1,nm,m,n,2)
     call addmat(ld,ome,mc,temp2,nm,nm,1)
     call mulmat(ld,temp2,bmv,temp3,nm,m,nm,1)
     call addmat(ld,temp1,temp3,f10,nm,m,1)
*
* H_Cm
*
     call addmat(ld,delta,mo,temp1,nm,nm,1)
     call mulmat(ld,rcm,temp1,temp2,l,nm,nm,1)
     call mulmat(ld,rc,qt12,temp3,l,nm,n,1)
     call addmat(ld,temp2,temp3,f20,l,nm,3)
*
* Compute average values of F1(0), F2(0)
*
     f10av  = 0.d0
     do 601 i = 1,nm
       do 601 j = 1,m
         f10av = f10av + dabs(f10(i,j))
601   continue
     f10av = f10av / dble(m*nm)
     write(2,251)f10av
     write(6,251)f10av
251   format(/'f10 average =',d9.2)
     f20av = 0.d0
     do 602 i = 1,l
       do 602 j = 1,nm
         f20av = f20av + dabs(f20(i,j))
602   continue
     f20av = f20av / dble(l*nm)
     write(2,253)f20av
```

```
      write(6,253)f20av
253   format('f20 average =',d9.2)
*
* Compute derivative w.r.t lambda
*
      call addmat(ld,a,a0,adot,n,n,3)
      call addmat(ld,b,b0,bdot,n,m,3)
      call addmat(ld,c,c0,cdot,l,n,3)
*
* Set parameters for FIXPNF
*
      arcre = 1.0d-5
      ansre = 1.0d-9
      arcae = arcre
      ansae = ansre
      do 701 i = 1,8
        sspar(i) = 0.0d0
701   continue
      iflag = -2
      ip(1) = 0
*
      if (trace .ne. 0) write(trace,200)
200   format(' ')
*
* Call tracking procedure
*
      do 801 i = 1,10
       call fixpnf(nn,x,iflag,arcre,arcae,ansre,ansae,trace,aa,nfe,
     &        arclen,yp,yold,ypold,qr,alpha,tz,ipivot,w2,wp,z0,z1,
     &        sspar,wa,ip)
        if (iflag .ne. 3) goto 901
801   continue
*
901   if (trace .ne. 0) write(6,206)iflag,nfe
      write(2,206)iflag,nfe
206   format(/'iflag =',i2,' nfe= ',i6)
*
      return
      end
*
*.....................................................................
*
      subroutine initp(wi,ww,sigma,u,ui,temp1,temp2,temp3,wa)
*
* initp is used to choose the initial point (Bm, Cm)_0
*
* Results: Bmi Cmi in input normal form (change in common block)
* Working storage:
* ww,wi,sigma,temp1,temp2,temp3  - working matrices
* wa - working arrays
*
      include 'inc.h'
*
      integer ind
      double precision eps,u(ld,ld),ui(ld,ld),f10av,f20av
      double precision temp1(ld,ld),temp2(ld,ld),temp3(ld,ld)
```

```
      double precision wi(ld,ld),ww(ld,ld),sigma(ld,ld),wa(15*z)
*
      eps = 1.d-26
*
* Compute B V B^T  and save it in bvbt.
*
      call mulmat(ld,b,v,temp1,n,m,m,1)
      call mulmat(ld,temp1,b,bvbt,n,n,m,3)
*
* Compute controllability Gramian Gc or Q
*
      call trans(ld,a,temp2,n,n)
      call scmat(ld,-1.d0,bvbt,temp1,n,n)
      call atxpxa(temp2,temp3,temp1,n,ld,ld,ld,eps,ind)
      call scmat(ld,1.d0,temp1,u,n,n)
*
* Compute C^T R C and save it in ctrc.
*
      call mulmat(ld,c,r,temp1,n,l,l,2)
      call mulmat(ld,temp1,c,ctrc,n,n,l,1)
*
* Compute observability Gramian Go or P
*
      call scmat(ld,1.d0,a,temp2,n,n)
      call scmat(ld,-1.d0,ctrc,temp1,n,n)
      call atxpxa(temp2,temp3,temp1,n,ld,ld,ld,eps,ind)
      call scmat(ld,1.d0,temp1,ui,n,n)
*
*  Compute the average of the two just solved equations
*
      call mulmat(ld,a,u,temp1,n,n,n,1)
      call mulmat(ld,u,a,temp2,n,n,n,3)
      call addmat(ld,temp1,temp2,temp3,n,n,1)
      call addmat(ld,temp3,bvbt,ww,n,n,1)
*
      call mulmat(ld,a,ui,temp1,n,n,n,2)
      call mulmat(ld,ui,a,temp2,n,n,n,1)
      call addmat(ld,temp1,temp2,temp3,n,n,1)
      call addmat(ld,temp3,ctrc,wi,n,n,1)
*
      f10av  = 0.d0
      f20av = 0.d0
      do 631 i = 1,n
        do 631 j = 1,n
          f10av = f10av + dabs(ww(i,j))
          f20av = f20av + dabs(wi(i,j))
631   continue
      f10av = f10av / dble(n*n)
      f20av = f20av / dble(n*n)
      write(2,281)f10av,f20av
      write(6,281)f10av,f20av
281   format(/'A Q + A Q^t + BVB^t average =',d27.15,
     &       /,'A^t P + P A + C^tRC average =',d27.15)
*
* Compute (W,U,Sigma) for given (Q,P)
*
```

```
      call cg(ld,n,1,u,ui,ww,wi,sigma,temp1,temp2,temp3,
     &         wa(1),wa(z+1),wa(2*z+1),wa(3*z+1),wa(4*z+1),
     &         wa(5*z+1),wa(6*z+1),wa(7*z+1))
*
* Compute the balanced form of (A,B,C)
*
      call mulmat(ld,wi,a,temp2,n,n,n,1)
      call mulmat(ld,temp2,ww,a,n,n,n,1)
      call mulmat(ld,wi,b,temp2,n,m,n,1)
      call mulmat(ld,c,ww,temp3,l,n,n,1)
      call scmat(ld,1.0d0,temp2,b,n,m)
      call scmat(ld,1.0d0,temp3,c,l,n)
*
* Compute the initial conditions (A0,B0,C0) and (Am, Bm, Cm)
*
      call setmat(ld,0.0d0,a0,n,n)
      do 101 j=1,nm
        do 101 i=1,nm
         a0(i,j) = a(i,j)
         am(i,j) = a0(i,j)
101   continue
*
      do 201 j=nm+1,n
        do 201 i=nm+1,n
         a0(i,j) = a(i,j)
201   continue
*
      call setmat(ld,0.0d0,b0,n,m)
      do 301 j=1,m
        do 301 i=1,nm
         b0(i,j) = b(i,j)
         bm(i,j) = b0(i,j)
301   continue
*
      call setmat(ld,0.0d0,c0,l,n)
      do 401 j=1,nm
        do 401 i=1,l
         c0(i,j) = c(i,j)
         cm(i,j) = c0(i,j)
401   continue
*
* Compute U =Sigma^0.5,  U^-1 =Sigma^-0.5
*
       call setmat(ld,0.0d0,u,nm,nm)
       call setmat(ld,0.0d0,ui,nm,nm)
       call setmat(ld,0.0d0,ome,nm,nm)
       do 402 i=1,nm
         u(i,i) = dsqrt(sigma(i,i))
         ui(i,i) = 1/u(i,i)
         ome(i,i) = sigma(i,i)*sigma(i,i)
402    continue
*
* Compute the input normal form of (Am,Bm,Cm)
*
      call mulmat(ld,ui,am,temp1,nm,nm,nm,1)
      call mulmat(ld,temp1,u,ami,nm,nm,nm,1)
```

```
      call mulmat(ld,ui,bm,bmi,nm,m,nm,1)
      call mulmat(ld,cm,u,cmi,l,nm,nm,1)
*
* Compute the accuracy of input normal form
*
      call mulmat(ld,bmi,v,temp1,nm,m,m,1)
      call mulmat(ld,temp1,bmi,temp2,nm,nm,m,3)
      call trans(ld,ami,temp1,nm,nm)
      call addmat(ld,ami,temp1,temp3,nm,nm,1)
      call addmat(ld,temp2,temp3,temp1,nm,nm,1)
*
      call mulmat(ld,cmi,r,temp2,nm,l,l,2)
      call mulmat(ld,temp2,cmi,temp3,nm,nm,l,1)
      do 501 i=1,nm
       do 501 j=1,nm
       temp2(i,j)=ami(j,i)*ome(j,j)+ome(i,i)*ami(i,j)+temp3(i,j)
501   continue
*
      f10av  = 0.d0
      f20av = 0.d0
      do 611 i = 1,nm
        do 611 j = 1,nm
          f10av = f10av + dabs(temp1(i,j))
          f20av = f20av + dabs(temp2(i,j))
611   continue
      f10av = f10av / dble(nm*nm)
      f20av = f20av / dble(nm*nm)
      write(2,271)f10av,f20av
      write(6,271)f10av,f20av
271   format(/'Input normal form equation average =',d20.10)
      return
      end
*
*.....................................................
*
      subroutine cg(ld,ndim,iflag,u,ui,ww,wi,sigma,temp1,temp2,
     &              temp3,t1,t2,t1i,t2i,vs,omega4,omega4i,s1)
*
* Compute a contragredient transformation (W,U,Sigma) of (Q,P).
*  and then compute the transformation  U, U inverse
*  to input normal form if iflag = 0.
*
* Input parameters:
* ndim - number of rows in W
* u,ui - (Q,P)
* iflag > 0 -- compute (W,U,Sigma), otherwise compute U, U inverse.
*
* Output parameters:
* ww,wi,sigma - transformation (W,U,Sigma), W = U inverse
* u,ui - transformation U for the input normal form ui=U inverse.
*
* Working storage:
* temp1,temp2,temp3,t1,t2,t1i,t2i,u,ui,vs,omega4,omega4i  -
*      - working matrices
* s1 - working array
*
```

```
      integer i,ind,j,iflag,ndim,ld
      double precision tmp,omega4i(ld,ld),omega4(ld,ld),s1(ld)
      double precision t1(ld,ld),t1i(ld,ld),t2(ld,ld),t2i(ld,ld)
      double precision temp1(ld,ld),temp2(ld,ld),temp3(ld,ld)
      double precision u(ld,ld),ui(ld,ld),vs(ld,ld)
      double precision ww(ld,ld),wi(ld,ld),sigma(ld,ld)
*
* Compute Q = u = V * diagD1 * V^t
*
      call rs(ld,ndim,u,s1,1,vs,temp1,temp2,ind)
*
* Compute T1 = V * diagD1^-1/2, T1^-1 = diagD1^1/2 * V^t
*
      call setmat(ld,0.d0,temp1,ndim,ndim)
      call setmat(ld,0.d0,temp2,ndim,ndim)
*
      nr=0
      do 301 i = 1,ndim
        if(s1(ndim-i+1).gt.0.0d0) then
        temp1(i,i) = 1.d0 / dsqrt(s1(ndim-i+1))
        temp2(i,i) = dsqrt(s1(ndim-i+1))
        else
        temp1(i,i) = 1.0d0
        temp2(i,i) = 1.0d0
        end if
 301    continue
*
      do 303 j = 1,ndim/2
        do 303 i = 1,ndim
          tmp = vs(i,j)
          vs(i,j) = vs(i,ndim-j+1)
          vs(i,ndim-j+1) = tmp
 303    continue
*
      call mulmat(ld,vs,temp1,t1,ndim,ndim,ndim,1)
      call mulmat(ld,temp2,vs,t1i,ndim,ndim,ndim,3)
*
* Compute T1^-1 * P * T1^-t = P1 and store in temp3,
*
      call mulmat(ld,t1i,ui,temp1,ndim,ndim,ndim,1)
      call mulmat(ld,temp1,t1i,temp3,ndim,ndim,ndim,3)
*
* Compute  P1 = T2 * D2 * T2^t = T2 * Omega* T2^t
*
      call rs(ld,ndim,temp3,s1,1,t2,temp1,temp2,ind)
*
      do 302 i = 1,ndim/2
        tmp = s1(ndim-i+1)
        s1(ndim-i+1) = s1(i)
        s1(i) = tmp
 302    continue
*
      do 304 j = 1,ndim/2
        do 304 i = 1,ndim
          tmp = t2(i,j)
          t2(i,j) = t2(i,ndim-j+1)
```

```
        t2(i,ndim-j+1) = tmp
 304   continue
*
* Compute T2^-1
*
      call trans(ld,t2,t2i,ndim,ndim)
*
* Compute U= T2^-1 * T1^-1, U^-1 = T1 * T2
*
      call mulmat(ld,t1i,t2i,u,ndim,ndim,ndim,4)
      call mulmat(ld,t2,t1,ui,ndim,ndim,ndim,4)
*
* Compute Sigma, Omega^1/4, Omega^-1/4
*
      call setmat(ld,0.d0,sigma,ndim,ndim)
      call setmat(ld,0.d0,omega4,ndim,ndim)
      call setmat(ld,0.d0,omega4i,ndim,ndim)
      do 311 i = 1,ndim
        sigma(i,i) = dsqrt(s1(i))
        omega4(i,i) = dsqrt(sigma(i,i))
        omega4i(i,i) = 1.d0/omega4(i,i)
311   continue
*
* Compute W = U^t * Omega^-1/4, W^-1 = Omega^1/4 * U^-t
*
      call mulmat(ld,u,omega4i,ww,ndim,ndim,ndim,1)
      call mulmat(ld,omega4,ui,wi,ndim,ndim,ndim,1)
*
      return
      end
*
*.....................................................
*
      subroutine rho(aa,lambda,x,y,p,ip)
*
* RHO evaluates the homotopy map rho(aa, lambda,x) at a given point
*
* The homotopy map is computed in RRHO and RHO simply
*  writes the homotopy hbm, hcm into array y.
*
* Output parameters:  y -- evaluated function
*
      include 'inc.h'
*
      integer ip(1)
      double precision aa(nn),lambda,p(25*z),y(nn+1),x(nn+1)
*
      call xmat(ld,2,nm,m,hbm,y)
      call xmat(ld,2,1,nm,hcm,y(1+m*nm))
*
      return
      end
*
*.....................................................
*
      subroutine rrho(lambda,x,temp1,temp2,temp3)
```

```
*
* RRHO evaluates the parameters needed to compute the homotopy
*    map hbm, hcm at a given point
*
* Input parameters: lambda,x - input point (lambda,x)
*
* Output parameters(through common block): hbm, hcm - homotopy map
*
      include 'inc.h'
*
      integer i,j
      double precision lambda,temp1(ld,ld),x(nn+1)
      double precision temp2(ld,ld),temp3(ld,ld)
*
* Transform variable x to (Bm, Cm)
*
      call xmat(ld,1,nm,m,bmi,x)
      call xmat(ld,1,l,nm,cmi,x(1+nm*m))
*
* Compute AAL = A0 + lambda *(A- A0)
*
      call scmat(ld,lambda,adot,temp1,n,n)
      call addmat(ld,a0,temp1,aal,n,n,1)
*
* Compute BBL = B0  + lambda *(B - B0)
*
      call scmat(ld,lambda,bdot,temp1,n,m)
      call addmat(ld,b0,temp1,bbl,n,m,1)
*
* Compute CCL = C0 + lambda *(C - C0)
*
      call scmat(ld,lambda,cdot,temp1,l,n)
      call addmat(ld,c0,temp1,ccl,l,n,1)
*
* Compute BV and BVB^T
*
      call mulmat(ld,bbl,v,bv,n,m,m,1)
      call mulmat(ld,bv,bbl,bvbt,n,n,m,3)
*
* Compute BmV and Bm V Bm^T
*
      call mulmat(ld,bmi,v,bmv,nm,m,m,1)
      call mulmat(ld,bmv,bmi,bvbtm,nm,nm,m,3)
*
* Compute RC and C^T RC
*
      call mulmat(ld,r,ccl,rc,l,n,l,1)
      call mulmat(ld,ccl,rc,ctrc,n,n,l,2)
*
* Compute RCm and Cm^T R Cm
*
      call mulmat(ld,r,cmi,rcm,l,nm,l,1)
      call mulmat(ld,cmi,rcm,ctrcm,nm,nm,l,2)
*
* Compute Omega and Ami
*
```

```
      call setmat(ld,0.d0,ome,nm,nm)
      do 301 i=1,nm
        ome(i,i) = ctrcm(i,i) / bvbtm(i,i)
301   continue
*
      do 401 i = 1,nm
         ami(i,i) = - 0.5d0*bvbtm(i,i)
        do 401 j=1,nm
         if(j.ne.i) then
         ami(i,j)=(ctrcm(i,j)-ome(j,j)*bvbtm(i,j))/(ome(j,j)-ome(i,i))
         end if
401   continue
*
* Compute Q12, P12 and save as,amts,ats,ams
*
      call mulmat(ld,bv,bmi,temp2,n,nm,m,3)
      call scmat(ld,-1.0d0,temp2,qt12,n,nm)
      call scmat(ld,1.0d0, aal, as,n,n)
      call trans(ld,ami,amts,nm,nm)
      call axpxb(as,ua,n,ld,ld,amts,vamt,nm,ld,ld,qt12,ld,
     &          1.0d-16,1.0d-16,ind)
*
      call mulmat(ld,ccl,rcm,pt12,n,nm,l,2)
      call trans(ld,aal,ats,n,n)
      call scmat(ld,1.0d0,ami,ams,nm,nm)
      call axpxb(ats,uat,n,ld,ld,ams,vam,nm,ld,ld,pt12,ld,
     &          1.0d-16,1.0d-16,ind)
*
* Compute S
*
      call mulmat(ld,pt12,qt12,temp1,nm,nm,n,2)
      call addmat(ld,temp1,ome,jam,nm,nm,1)
*
* Compute Mo
*
      do 501 j=1,nm
         do 501 i=j+1, nm
          mo(i,j) =  (jam(i,j)-jam(j,i)) / (ome(j,j)-ome(i,i))
          mo(j,i) = mo(i,j)
501   continue
      do 601 i = 1,nm
         mo(i,i) = 0.0d0
         do 602 j=1,i-1
            mo(i,i) = mo(i,i) + ami(i,j)*mo(j,i)
602      continue
         do 603 j=i+1,nm
            mo(i,i) = mo(i,i) + ami(i,j)*mo(j,i)
603      continue
         mo(i,i) = - mo(i,i) /(ami(i,i))
601   continue
*
* Compute Mc
*
       do 701 i=1,nm
          do 701 j=1,nm
           temp1(i,j) = ome(i,i) * mo(i,j)
```

```
701    continue
*
       call addmat(ld,temp1,jam,mc,nm,nm,4)
*
* Compute the homotopy map
*
* H_Bm
*
      call mulmat(ld,pt12,bv,temp1,nm,m,n,2)
      call addmat(ld,ome,mc,temp2,nm,nm,1)
      call mulmat(ld,temp2,bmv,temp3,nm,m,nm,1)
      call addmat(ld,temp1,temp3,hbm,nm,m,1)
*
* H_Cm
*
      call mulmat(ld,rcm,mo,temp1,l,nm,nm,1)
      call addmat(ld,rcm,temp1,temp2,l,nm,1)
      call mulmat(ld,rc,qt12,temp3,l,nm,n,1)
      call addmat(ld,temp2,temp3,hcm,l,nm,3)
*
      return
      end
*
*.....................................................
*
      subroutine rhojac(aa,lambda,x,y,kk,p,ip)
*
* RHOJAC evaluates kk-th column of Jacobian at given point.
*
* Input parameters:  lambda,x -- input point (lambda,x)
*                    kk -- column number
*
* Output parameters:  y - evaluated function
*
      include 'inc.h'
*
      integer der,i,j,kk,ip(1)
      double precision aa(nn),p(25*z),y(nn+1),lambda,x(nn+1)
*
      if (kk .eq. 1) der = 3
      if (kk .gt. 1 .and. kk .le. m*nm+1) der = 1
      if (kk .gt. m*nm+1) der = 2
*
* Determine position (i,j) in Bm, or Cm
*
      if (der .eq. 1) call compij(kk-1,nm,m,i,j)
      if (der .eq. 2) call compij(kk-m*nm-1,l,nm,i,j)
*
* Derivatives with respect to Bmij
*
      if (der .eq. 1) then
       call jacbm(i,j,y,p(1),p(z+1),p(2*z+1),p(3*z+1),p(4*z+1),
     &       p(5*z+1),p(6*z+1),p(7*z+1),p( 8*z+1),
     &       p(9*z+1),p(10*z+1))
      endif
*
```

```
* Derivatives with respect to Cmij
*
      if (der .eq. 2) then
       call jaccm(i,j,y,p(1),p(z+1),p(2*z+1),p(3*z+1),p(4*z+1),
     &       p(5*z+1),p(6*z+1),p(7*z+1),p(8*z+1),
     &       p(9*z+1),p(10*z+1))
      endif
*
* Derivatives with respect to lambda
*
      if (der .eq. 3) then
*
* Compute the parameters needed to the Jacobian it needs to be
* called once for each complete Jacobian evaluation
*
       call rrho(lambda,x,p(1),p(z+1),p(2*z+1))
*
       call jaclam(lambda,y,p(1),p(z+1),p(2*z+1),p(3*z+1),p(4*z+1),
     &       p(5*z+1),p(6*z+1),p(7*z+1),p(8*z+1),
     &       p(9*z+1),p(10*z+1))
      endif
*
      return
      end
*
*.....................................................................
*
      subroutine jaclam(lambda,y,temp1,temp2,temp3,temp4,temp5,
     &            temp6,qtd12,ptd12,mol,mcl,jaml)
*
* JACLAM evaluates 1st column of Jacobian (w.r.t Lambda) at given point.
*
* Input parameters: lambda
*
* Output parameters:  y -- evaluated function
*
* Working storage: mol, mcl, jaml -- derivatives of Mo, Mc, S
*                  qtd12, ptd12 -- derivatives of Q12, P12
*                  temp1 - temp6 -- temporary storage
*
      include 'inc.h'
*
      integer i,j
      double precision temp1(ld,ld),temp2(ld,ld),temp6(ld,ld)
      double precision temp3(ld,ld),temp4(ld,ld),temp5(ld,ld)
      double precision jaml(ld,ld),qtd12(ld,ld),ptd12(ld,ld)
      double precision mol(ld,ld),mcl(ld,ld),y(nn+1),lambda
*
* Compute derivatives of Q12 ,P12, S w.r.t. lambda
*
      call mulmat(ld,adot,qt12,temp1,n,nm,n,1)
      call mulmat(ld,bdot,bmv,temp3,n,nm,m,3)
      call addmat(ld,temp1,temp3,qtd12,n,nm,4)
      call axpxb(as,ua,n,ld,ld,amts,vamt,nm,ld,ld,qtd12,ld,
     &          -1.0d0,-1.0d0,ind)
*
```

```
      call mulmat(ld,adot,pt12,temp1,n,nm,n,2)
      call mulmat(ld,cdot,rcm,temp3,n,nm,l,2)
      call addmat(ld,temp3,temp1,ptd12,n,nm,3)
      call axpxb(ats,uat,n,ld,ld,ams,vam,nm,ld,ld,ptd12,ld,
     &          -1.0d0,-1.0d0,ind)
*
* Compute S
*
      call mulmat(ld,ptd12,qt12,temp1,nm,nm,n,2)
      call mulmat(ld,pt12,qtd12,temp2,nm,nm,n,2)
      call addmat(ld,temp1,temp2,jaml,nm,nm,1)
*
* Compute the derivatives of Mo, Mc and save them in mol, mcl
*
      do 401 j=1,nm
         do 401 i=j+1,nm
          mol(i,j) = ( jaml(i,j)-jaml(j,i))/(ome(j,j)-ome(i,i))
          mol(j,i) = mol(i,j)
401    continue
*
      do 501 i=1,nm
      mol(i,i) = 0.0d0
        do 502 j=i+1,nm
          mol(i,i) = mol(i,i)+ami(i,j)*mol(j,i)
502     continue
        do 503 j=1,i-1
          mol(i,i) = mol(i,i) +ami(i,j)*mol(j,i)
503     continue
        mol(i,i) = -mol(i,i)/ami(i,i)
501    continue
*
* Compute  derivative of Mc
*
       do 601 i = 1, nm
  do 601 j = 1,nm
           mcl(i,j) = -ome(i,i) * mol(i,j)-jaml(i,j)
601     continue
*
* Compute derivatives of H_Bm
*
       call mulmat(ld,ptd12,bv,temp1,nm,m,n,2)
       call mulmat(ld,bdot,v,temp2,n,m,m,1)
       call mulmat(ld,pt12,temp2,temp3,nm,m,n,2)
       call mulmat(ld,mcl,bmv,temp2,nm,m,nm,1)
  call addmat(ld,temp1,temp3,temp4,nm,m,1)
  call addmat(ld,temp4,temp2,temp1,nm,m,1)
*
       call xmat(ld,2,nm,m,temp1,y)
*
* Compute derivatives of H_Cm
*
       call mulmat(ld,r,cdot,temp1,l,n,l,1)
       call mulmat(ld,temp1,qt12,temp2,l,nm,n,1)
       call mulmat(ld,rc,qtd12,temp1,l,nm,n,1)
  call mulmat(ld,rcm,mol,temp3,l,nm,nm,1)
       call addmat(ld,temp1,temp2,temp4,l,nm,4)
```

```
       call addmat(ld,temp4,temp3,temp1,1,nm,1)
*
       call xmat(ld,2,1,nm,temp1,y(1+nm*m))
*
      return
      end
*
*.....................................................................
*
      subroutine jacbm(k,ll,y,temp1,temp2,temp3,temp4,temp5,temp6,
     &          qtd12,ptd12,mol,mcl,jaml)
*
* JACBM -- evaluates k-th column of Jacobian at given point
*          with respect to Bm. or w.r.t. Bm_k ll
*
* Input parameters: k, ll -- subscript of Bm
*
* Output parameters: y -- evaluated function (kk-th column of Jacobian)
*
* Working storage: mol, mcl, jaml -- derivatives of Mo, Mc, S
*                  qtd12, ptd12 -- derivatives of Q12, P12
*                  temp1 - temp6 -- temporary storage
*
      include 'inc.h'
*
      integer i,j,k,ll
      double precision alpha,y(nn+1)
      double precision temp1(ld,ld),temp2(ld,ld),temp3(ld,ld)
      double precision temp4(ld,ld),temp5(ld,ld),temp6(ld,ld)
      double precision qtd12(ld,ld),ptd12(ld,ld)
      double precision mol(ld,ld),mcl(ld,ld),jaml(ld,ld)
*
* Compute BV E^(ll,k)
*
       call setmat(ld,0.d0,temp1,n,nm)
       do 301 i=1,n
         temp1(i,k) = bv(i,ll)
301    continue
*
* Compute derivative of Am w.r.t. Bmkl and save it in temp6
*
* alpha_k ll
*
       alpha = - 2.0d0 * bmv(k,ll)* ome(k,k) / bvbtm(k,k)
*
       do 401 j=1,nm
         temp6(j,j) = - bmv(k,ll) * delta(k,j)
         do 401 i = 1,nm
          if(i.ne.j) then
            temp6(i,j)= -((delta(k,j) -delta(k,i)) * alpha * ami(i,j)
     &           +alpha * delta(k,j) *bvbtm(i,j) +ome(j,j) *delta(k,i)
     &           * bmv(j,ll) +ome(j,j) *bmv(i,ll) *delta(j,k) ) /
     &           (ome(j,j) -ome(i,i))
          end if
401    continue
*
```

```
* Compute derivative of Q12, P12
*
     call mulmat(ld,qt12,temp6,temp3,n,nm,nm,3)
     call addmat(ld,temp3,temp1,qtd12,n,nm,4)
     call axpxb(as,ua,n,ld,ld,amts,vamt,nm,ld,ld,qtd12,ld,
    &          -1.0d0,-1.0d0,ind)
*
     call mulmat(ld,pt12,temp6,temp2,n,nm,nm,1)
     call scmat(ld,-1.0d0,temp2,ptd12,n,nm)
     call axpxb(ats,uat,n,ld,ld,ams,vam,nm,ld,ld,ptd12,ld,
    &          -1.0d0,-1.0d0,ind)
*
* Compute S
*
     call mulmat(ld,ptd12,qt12,temp1,nm,nm,n,2)
     call mulmat(ld,pt12,qtd12,temp2,nm,nm,n,2)
     call addmat(ld,temp1,temp2,temp3,nm,nm,1)
     call setmat(ld,0.d0,temp5,nm,nm)
      temp5(k,k) = alpha
     call addmat(ld,temp3,temp5,jaml,nm,nm,1)
*
* Compute derivatives of Mo,Mc w.r.t. Bm_k ll
*
      do 501 j=1,nm
         do 501 i = j+1,nm
           mol(i,j)= ( jaml(i,j) -jaml(j,i)-mo(i,j)*alpha *
    &         (delta(j,k) -delta(i,k)) )/(ome(j,j) -ome(i,i))
           mol(j,i) = mol(i,j)
501    continue
*
      do 601 i=1,nm
         mol(i,i) = 0.0d0
         do 602 j=1,nm
           if(i.ne.j) then
           mol(i,i) = mol(i,i) + temp6(i,j)*mo(j,i) + ami(i,j)*mol(j,i)
           end if
602       continue
         mol(i,i)=(-temp6(i,i)*mo(i,i) -mol(i,i)) /ami(i,i)
601    continue
*
* Compute derivative of Mc w.r.t. Bm_k ll
*
       call mulmat(ld,temp5,mo,temp2,nm,nm,nm,1)
       call mulmat(ld,ome,mol,temp3,nm,nm,nm,1)
       call addmat(ld,temp2,temp3,temp4,nm,nm,1)
       call addmat(ld,temp4,jaml,mcl,nm,nm,4)
*
* Compute derivatives of H_Bm
*
       call setmat(ld,0.0d0,temp1,nm,m)
        do 801 j=1,m
         temp1(k,j) = v(ll,j)
 801     continue
       call addmat(ld,ome,mc,temp2,nm,nm,1)
  call mulmat(ld,temp2,temp1,temp3,nm,m,nm,1)
       call mulmat(ld,ptd12,bv,temp1,nm,m,n,2)
```

```
  call addmat(ld,temp5,mcl,temp4,nm,nm,1)
  call mulmat(ld,temp4,bmv,temp2,nm,m,nm,1)
       call addmat(ld,temp1,temp2,temp4,nm,m,1)
       call addmat(ld,temp4,temp3,temp1,nm,m,1)
       call xmat(ld,2,nm,m,temp1,y)
*
* Compute derivatives of H_Cm
*
       call mulmat(ld,rcm,mol,temp1,l,nm,nm,1)
  call mulmat(ld,rc,qtd12,temp2,l,nm,n,1)
  call addmat(ld,temp1,temp2,temp3,l,nm,3)
       call xmat(ld,2,l,nm,temp3,y(1+m*nm))
*
     return
     end
*
*.....................................................................
*
     subroutine jaccm(k,ll,y,temp1,temp2,temp3,temp4,temp5,temp6,
    &              qtd12,ptd12,mol,mcl,jaml)
*
* JACCM evaluates k-th column of Jacobian at given point w.r.t. Cm.
*
* Input parameters: k, ll -- subscripts of the element of Cm
*
* Output parameters: y -- evaluated function
*
* Working storage: mol, mcl, jaml -- derivatives of Mo, Mc, S
*                qtd12, ptd12 -- derivatives of Q12, P12
*                temp1 - temp6 -- temporary storage
*
     include 'inc.h'
*
     integer i,j,ll,k
     double precision qtd12(ld,ld),ptd12(ld,ld),beta
     double precision jaml(ld,ld),mol(ld,ld),mcl(ld,ld),y(nn+1)
     double precision temp1(ld,ld),temp2(ld,ld),temp3(ld,ld)
     double precision temp4(ld,ld),temp5(ld,ld),temp6(ld,ld)
*
* Compute derivative of Am w.r.t. Cm_k ll and save it in temp6
* beta_k ll
*
      beta = 2.0d0 * rcm(k,ll)/ bvbtm(ll,ll)
*
      do 401 j=1,nm
        temp6(j,j) = 0.0d0
        do 401 i = 1,nm
         if(i.ne.j) then
           temp6(i,j)= (delta(ll,i)*rcm(k,j)+delta(ll,j)
    &           *rcm(k,i) -beta * delta(ll,j) *bvbtm(i,j)
    &           -(delta(j,ll)- delta(i,ll))*beta*ami(i,j))/
    &           (ome(j,j) -ome(i,i))
         end if
401    continue
*
* Compute derivatives of Q12, P12
```

```
*
      call mulmat(ld,qt12,temp6,temp3,n,nm,nm,3)
      call scmat(ld,-1.0d0,temp3,qtd12,n,nm)
      call axpxb(as,ua,n,ld,ld,amts,vamt,nm,ld,ld,qtd12,ld,
     &          -1.0d0,-1.0d0,ind)
*
* Compute C^T R E^(k,ll) save it in temp1
*
       call setmat(ld,0.0d0,temp1,n,nm)
       do 201 j=1,n
         temp1(j,ll) = rc(k,j)
201    continue
*
      call mulmat(ld,pt12,temp6,temp3,n,nm,nm,1)
      call addmat(ld,temp3,temp1,ptd12,n,nm,2)
      call axpxb(ats,uat,n,ld,ld,ams,vam,nm,ld,ld,ptd12,ld,
     &          -1.0d0,-1.0d0,ind)
*
* Compute S
*
      call mulmat(ld,ptd12,qt12,temp1,nm,nm,n,2)
      call mulmat(ld,pt12,qtd12,temp2,nm,nm,n,2)
      call addmat(ld,temp1,temp2,temp3,nm,nm,1)
      call setmat(ld,0.d0,temp5,nm,nm)
       temp5(ll,ll) = beta
      call addmat(ld,temp3,temp5,jaml,nm,nm,1)
*
* Compute derivatives of Mo,Mc w.r.t. Cm_k ll
*
       do 501 j=1,nm
          do 501 i = j+1,nm
            mol(i,j)= ( (jaml(i,j) -jaml(j,i))-mo(i,j)*beta *
     &           (delta(j,ll) -delta(i,ll)))/(ome(j,j)-ome(i,i))
            mol(j,i)=mol(i,j)
501    continue
*
       do 601 i=1,nm
        mol(i,i) = 0.0d0
        do 602 j=1,nm
        if(i.ne.j)then
         mol(i,i)=mol(i,i)+temp6(i,j)*mo(j,i)+ami(i,j)*mol(j,i)
 end if
602     continue
       mol(i,i)=-mol(i,i) /ami(i,i)
601    continue
*
* Compute derivative of Mc w.r.t. Cm_k ll
*
       call mulmat(ld,temp5,mo,temp2,nm,nm,nm,1)
call mulmat(ld,ome,mol,temp3,nm,nm,nm,1)
       call addmat(ld,temp2,temp3,temp4,nm,nm,1)
       call addmat(ld,temp4,jaml,mcl,nm,nm,4)
*
* Compute derivative of H_Bm
*
     call mulmat(ld,ptd12,bv,temp1,nm,m,n,2)
```

```
      call addmat(ld,temp5,mcl,temp2,nm,nm,1)
      call mulmat(ld,temp2,bmv,temp3,nm,m,nm,1)
      call addmat(ld,temp1,temp3,temp4,nm,m,1)
      call xmat(ld,2,nm,m,temp4,y)
*
* Compute derivative of H_Cm
*
       call mulmat(ld,rcm,mol,temp1,l,nm,nm,1)
       call mulmat(ld,rc,qtd12,temp2,l,nm,n,1)
       call addmat(ld,temp2,temp1,temp3,l,nm,2)
       call addmat(ld,delta,mo,temp1,nm,nm,1)
        call setmat(ld,0.0d0,temp2,l,nm)
        do 701 j=1,nm
 701    temp2(k,j) = temp1(ll,j)
        call mulmat(ld,r,temp2,temp4,l,nm,l,1)
        call addmat(ld,temp3,temp4,temp1,l,nm,1)
        call xmat(ld,2,l,nm,temp1,y(1+m*nm))
*
       return
       end
*
*.....................................................................
*
      subroutine evaluate(x,e1,e2,mat2,al,temp1,temp2,temp3,wa)
*
* Evaluate the solution and the cost of the solution
*
      include 'inc.h'
*
      integer z2
      double precision e1(ld),e2(ld),mat2(ld,ld),al(ld,ld)
      double precision temp1(ld,ld),temp2(ld,ld),temp3(ld,ld)
      double precision wa(20*z),x(nn),f10av,f20av
*
* Transform x into Bm and Cm
*
      call xmat(ld,1,nm,m,bmi,x(1))
      call xmat(ld,1,l,nm,cmi,x(1+nm*m))
*
* Compute precision of the solution
*
* Compute BmV, BV, RC, RCm, BmVBm^t, Cm^tRCm
*
      call mulmat(ld,b,v,bv,n,m,m,1)
      call mulmat(ld,bmi,v,bmv,nm,m,m,1)
      call mulmat(ld,r,c,rc,l,n,l,1)
      call mulmat(ld,r,cmi,rcm,l,nm,l,1)
      call mulmat(ld,bmv,bmi,bvbtm,nm,nm,m,3)
      call mulmat(ld,cmi,rcm,ctrcm,nm,nm,l,2)
*
* Compute ome and Ami
*
      do 301 i = 1,nm
        ome(i,i) = ctrcm(i,i) / bvbtm(i,i)
301    continue
*
```

```
      do 401 i = 1,nm
         ami(i,i) = - 0.5d0*bvbtm(i,i)
        do 401 j=1,nm
         if(j.ne.i) then
          ami(i,j)=(ctrcm(i,j)-ome(j,j)*bvbtm(i,j))/(ome(j,j)-ome(i,i))
         end if
401    continue
*
* Compute Q P tilde 12
*
      call mulmat(ld,bv,bmi,temp2,n,nm,m,3)
      call scmat(ld,-1.0d0,temp2,qt12,n,nm)
      call scmat(ld,1.0d0,a,temp2,n,n)
      call trans(ld,ami,temp3,nm,nm)
      call axpxb(temp2,ua,n,ld,ld,temp3,vamt,nm,ld,ld,qt12,ld,
     &          1.0d-20,1.0d-20,ind)
*
      call mulmat(ld,c,rcm,pt12,n,nm,l,2)
      call trans(ld,a,temp2,n,n)
      call scmat(ld,1.0d0,ami,temp3,nm,nm)
      call axpxb(temp2,uat,n,ld,ld,temp3,vam,nm,ld,ld,pt12,ld,
     &          1.0d-20,1.0d-20,ind)
*
*  Compute the average of the two just solved equations
*
      call mulmat(ld,a,qt12,temp1,n,nm,n,1)
      call mulmat(ld,qt12,ami,temp2,n,nm,nm,3)
      call addmat(ld,temp1,temp2,temp3,n,nm,1)
      call mulmat(ld,bv,bmi,temp2,n,nm,m,3)
      call addmat(ld,temp3,temp2,temp1,n,nm,1)
*
      call mulmat(ld,a,pt12,temp2,n,nm,n,2)
      call mulmat(ld,pt12,ami,temp3,n,nm,nm,1)
      call addmat(ld,temp2,temp3,al,n,nm,1)
      call mulmat(ld,rc,cmi,temp3,n,nm,l,2)
      call addmat(ld,al,temp3,temp2,n,nm,3)
*
      f10av  = 0.d0
      do 621 i = 1,n
        do 621 j = 1,nm
          f10av = f10av + dabs(temp1(i,j))
621    continue
      f10av = f10av / dble(n*nm)
      write(2,271)f10av
      write(6,271)f10av
271    format(/'In eval equation average =',d20.12)
      f20av = 0.d0
      do 622 i = 1,n
        do 622 j = 1,nm
          f20av = f20av + dabs(temp2(i,j))
622    continue
      f20av = f20av / dble(n*nm)
      write(2,273)f20av
      write(6,273)f20av
273    format('In eval equation average =',d20.12)
*
```

```
* Check whether the solutions satisfy optimal projection eqs
*
      call mulmat(ld,qt12,qt12,mat2,n,n,nm,3)
      do 623 i=1,nm
         do 623 j=1,n
         temp1(i,j) = pt12(j,i)/ome(i,i)
623    continue
      call mulmat(ld,pt12,temp1,al,n,n,nm,1)
      call mulmat(ld,bv,b,bvbt,n,n,m,3)
      call mulmat(ld,c,rc,ctrc,n,n,l,2)
      call mulmat(ld,qt12,temp1,temp2,n,n,nm,1)
      call scmat(ld,-1.0d0,temp2,jam,n,n)
*
* 1st equation
*
      call mulmat(ld,a,mat2,temp1,n,n,n,1)
      call mulmat(ld,mat2,a,temp2,n,n,n,3)
      call addmat(ld,temp1,temp2,temp3,n,n,1)
      call addmat(ld,temp3,bvbt,temp2,n,n,1)
      call mulmat(ld,jam,temp2,temp1,n,n,n,1)
*
* 2nd equation
*
      call mulmat(ld,a,al,temp2,n,n,n,2)
      call mulmat(ld,al,a,temp3,n,n,n,1)
      call addmat(ld,temp3,temp2,mat2,n,n,1)
      call addmat(ld,mat2,ctrc,temp3,n,n,1)
      call mulmat(ld,temp3,jam,temp2,n,n,n,1)
     f10av  = 0.d0
     f20av  = 0.d0
     do 605 i = 1,n
       do 605 j = 1,n
         f10av = f10av + dabs(temp1(i,j))
         f20av = f20av + dabs(temp2(i,j))
605   continue
     f10av = f10av / dble(n*n)
     f20av = f20av / dble(n*n)
     write(2,241)f10av,f20av
     write(6,241)f10av,f20av
241   format(/'Optimal Proj. equation average =',d9.2,2x,d9.2)
*
* Compute S (jam)
*
     call mulmat(ld,pt12,qt12,temp1,nm,nm,n,2)
     call addmat(ld,temp1,ome,jam,nm,nm,1)
*
* Compute Mo
*
     do 302 j=1,nm
        do 302 i=j+1, nm
         mo(i,j) =  (jam(i,j)-jam(j,i)) / (ome(j,j)-ome(i,i))
         mo(j,i) = mo(i,j)
302   continue
     do 411 i = 1,nm
        mo(i,i) = 0.0d0
        do 412 j=1,i-1
```

```
412         mo(i,i) = mo(i,i) + ami(i,j)*mo(j,i)
        do 413 j=i+1,nm
413         mo(i,i) = mo(i,i) + ami(i,j)*mo(j,i)
        mo(i,i) = - mo(i,i) /(ami(i,i))
411   continue
*
* Compute Mc
*
      do 501 i=1,nm
         do 501 j=1,nm
          temp1(i,j) = ome(i,i) * mo(i,j)
501    continue
*
      call addmat(ld,temp1,jam,mc,nm,nm,4)
*
* Compute the homotopy map
*
* H_Bm
*
     call mulmat(ld,pt12,bv,temp1,nm,m,n,2)
     call addmat(ld,ome,mc,temp2,nm,nm,1)
     call mulmat(ld,temp2,bmv,temp3,nm,m,nm,1)
     call addmat(ld,temp1,temp3,f10,nm,m,1)
*
* H_Cm
*
     call addmat(ld,delta,mo,temp1,nm,nm,1)
     call mulmat(ld,rcm,temp1,temp2,l,nm,nm,1)
     call mulmat(ld,rc,qt12,temp3,l,nm,n,1)
     call addmat(ld,temp2,temp3,f20,l,nm,3)
*
* Compute average values of Hbm (f10) and Hcm (f20)
*
     f10av  = 0.d0
     do 601 i = 1,nm
       do 601 j = 1,m
         f10av = f10av + dabs(f10(i,j))
601   continue
     f10av = f10av / dble(m*nm)
     write(2,251)f10av
     write(6,251)f10av
251   format(/'f10 average =',d9.2)
*
     f20av = 0.d0
     do 602 i = 1,l
       do 602 j = 1,nm
         f20av = f20av + dabs(f20(i,j))
602   continue
     f20av = f20av / dble(l*nm)
     write(2,253)f20av
     write(6,253)f20av
253   format('f20 average =',d9.2)
*
* Compute the accuracy of input normal form
*
     call trans(ld,ami,temp1,nm,nm)
```

```
      call addmat(ld,ami,temp1,temp3,nm,nm,1)
      call addmat(ld,bvbtm,temp3,temp1,nm,nm,1)
*
      do 511 i=1,nm
       do 511 j=1,nm
       temp2(i,j)=ami(j,i)*ome(j,j)+ome(i,i)*ami(i,j)+ctrcm(i,j)
511    continue
*
      f10av  = 0.d0
      do 611 i = 1,nm
        do 611 j = 1,nm
          f10av = f10av + dabs(temp1(i,j))
611    continue
      f10av = f10av / dble(nm*nm)
      write(2,272)f10av
      write(6,272)f10av
272    format(/'The final Input normal form average =',d20.10)
*
      f20av = 0.d0
      do 612 i = 1,nm
        do 612 j = 1,nm
          f20av = f20av + dabs(temp2(i,j))
612    continue
      f20av = f20av / dble(nm*nm)
      write(2,274)f20av
      write(6,274)f20av
274    format('The final Input normal form average =',d20.10)
*
      if (trace .ne. 0) then
        write(trace,211)
211      format(/'x')
        call writev(x,nn)
      endif
*
* Print Am, Bm, Cm in input normal form
*
      if (trace .ne. 0) then
        write(trace,261)
261      format(/'The input normal form',/'Am')
        call writem(ld,ami,nm,nm)
        write(trace,262)
262      format(/'Bm')
        call writem(ld,bmi,nm,m)
        write(trace,263)
263      format(/'Cm')
        call writem(ld,cmi,l,nm)
      endif
*
* Print eigenvalues of A, Am
*
      if (trace .ne. 0) write(trace,254)
254    format(/'eigenvalues of A')
      call eig(ld,n,trace,a,temp1,temp2,temp3,mat2,e1,e2)
      if (trace .ne. 0) write(trace,255)
255    format(/'eigenvalues of Am')
      call eig(ld,nm,trace,ami,temp1,temp2,temp3,mat2,e1,e2)
```

```
*
* Compute BVB^t C^t RC
*
      call mulmat(ld,b,v,temp1,n,m,m,1)
      call mulmat(ld,temp1,b,bvbt,n,n,m,3)
      call mulmat(ld,c,r,temp1,n,l,l,2)
      call mulmat(ld,temp1,c,ctrc,n,n,l,1)
*
* Compute cost J
*
      z2 = ld2*ld2
      call cost(temp1,temp2,temp3,wa(1),wa(z2+1),wa(2*z2+1),
     &          wa(3*z2+1),wa(4*z2+1),wa(5*z2+1),wa(6*z2+1),
     &          wa(7*z2+1))
*
      return
      end
*
*.....................................................................
*
      subroutine cost(temp1,temp2,temp3,temp4,bt,ct,temp5,temp6,
     *                rt,vt,at)
*
* Compute cost J at the solution
*
      include 'inc.h'
*
      integer i,ind,j
      double precision bt(ld2,ld2),ct(ld2,ld2),eps,fn1,fn2
      double precision rt(ld2,ld2),vt(ld2,ld2),at(ld2,ld2)
      double precision temp1(ld,ld),temp2(ld,ld),temp3(ld,ld)
      double precision temp4(ld2,ld2),temp5(ld2,ld2),temp6(ld2,ld2)
*
      eps = 1.d-16
*
* Make At, Bt, Ct
*
      call setmat(ld2,0.d0,at,nt,nt)
      do 301 i = 1,n
        do 301 j = 1,n
          at(i,j) = a(i,j)
301   continue
      do 303 i = 1,nm
        do 303 j = 1,nm
          at(n+i,n+j) = ami(i,j)
303   continue
*
      call setmat(ld2,0.d0,bt,nt,m)
      do 311 j = 1,m
        do 312 i = 1,n
          bt(i,j) = b(i,j)
312     continue
        do 313 i = 1,nm
          bt(n+i,j) = bmi(i,j)
313     continue
311   continue
```

```
*
      call setmat(ld2,0.d0,ct,l,nt)
      do 321 i = 1,l
        do 322 j = 1,n
          ct(i,j) = c(i,j)
322     continue
        do 323 j = 1,nm
          ct(i,n+j) = -cmi(i,j)
323     continue
321   continue
*
* Make Vt, Rt
*
      do 331 i = 1,nt
        do 331 j = 1,m
          temp4(i,j) = 0.d0
          do 331 k = 1,m
            temp4(i,j) = temp4(i,j) + bt(i,k) * v(k,j)
331   continue
      call mulmat(ld2,temp4,bt,vt,nt,nt,m,3)
*
      do 336 i = 1,nt
        do 336 j = 1,l
          temp4(i,j) = 0.d0
          do 336 k = 1,l
            temp4(i,j) = temp4(i,j) + ct(k,i) * r(k,j)
336   continue
      call mulmat(ld2,temp4,ct,rt,nt,nt,l,1)
*
* Compute cost J and Sqrt (J / tr(C^t R C Gc))
*
      call trans(ld2,at,temp4,nt,nt)
      call scmat(ld2,-1.d0,vt,temp6,nt,nt)
      call atxpxa(temp4,temp5,temp6,nt,ld2,ld2,ld2,eps,ind)
      call mulmat(ld2,temp6,rt,temp4,nt,nt,nt,1)
      fn2 = 0.d0
      do 361 i = 1,nt
        fn2 = fn2 + temp4(i,i)
361   continue
*
      call trans(ld,a,temp1,n,n)
      call scmat(ld,-1.d0,bvbt,temp3,n,n)
      call atxpxa(temp1,temp2,temp3,n,ld,ld,ld,eps,ind)
      call mulmat(ld,ctrc,temp3,temp1,n,n,n,1)
      fn1 = 0.d0
      do 362 i = 1,n
        fn1 = fn1 + temp1(i,i)
362   continue
*
* Print J and  Sqrt (J / tr(C^t R C Gc))
*
        write(2,200)
200     format(' ')
        write(2,231) fn2
231     format('J          =',d23.16)
        write(2,232) fn1
```

```
232     format('ctrcQc      =',d23.16)
        fn2 = dsqrt(dabs(fn2 / fn1))
        write(2,233) fn2
233     format('normalized J =',f23.16)
        write(6,233) fn2
*
      return
      end
```

Subroutines mulmat(), addmat(), setmat(), scmat(), xmat(), eig(), read2(), trans(), writem() and writev() can be found in [41].

VITA.

Yuzhen Ge was born on November 27, 1961 in Hunan, China. She was admitted to Zhongshan university by passing the first nationwide college entrance exams after the culture revolution in March, 1978. she received B.S. in Physics in February of 1982, and then was admitted to the graduate program of Institute of Physics, Academia Sinica. She came to the department of mathematics of Virginia Polytechnic Institute and State University in September, 1986 and graduated with Ph.D. in mathematics in December, 1990. In May of 1993, she received M.S. degree in computer science and applications.