STOCHASTIC MODEL OF THE NASA/MSFC GROUND FACILITY FOR LARGE SPACE STRUCTURES WITH UNCERTAIN PARAMETERS – THE MAXIMUM ENTROPY APPROACH

MARSHALL GRANT
IN-61-CR
239803
64P.

Report Part II

by

Dr. Wei Shen Hsia
Department of Mathematics
The University of Alabama
Tuscaloosa, Alabama 35487
(205) 348 – 5153

NASA Grant Number: NAG8 – 081
Grant Period: 5–23–88 to 11–22–89

(NASA-CR-186009) STOCHASTIC MODEL OF THE NASA/MSFC GROUND FACILITY FOR LARGE SPACE STRUCTURES WITH UNCERTAIN PARAMETERS: THE MAXIMUM ENTROPY APPROACH, PART 2 Final Report, 23 May 1988 - 22 Nov. 1989 (Alabama G3/61

N90-16407

Unclas
0239803

TABLE OF CONTENTS

1. INTRODUCTION

The National Aeronautics and Space Administration and the Department of Defense are actively involved in the development of a validated technology data base in the areas of control/structures inter–action, deployment dynamics and system performance for Large Space Structures (LSS). In the Control System Division of the System Dynamics Laboratory of the NASA/MSFC, a Ground Facility (GF), in which the dynamics and control system concepts being considered for LSS applications can be verified, has been designed and built under Dr. Henry Waites' supervision [8]. The viability and versatility of this MSFC LSS ground test facility was recognized by the U.S. Air Force Wright Aeronautical Laboratory as a site for their Vibration Control of Space Structures (VCOSS) testing.

One of the important aspects of the GF is to verify the analytical model for the control system design. The procedure is to describe the control system mathematically as well as possible, then to perform tests on the control system, and finally to factor those results into the mathematical model.

However, development of a "correct" mathematical model of a system is still an art. In constructing large order structural models, various errors, such as modelling errors, parameter errors, improperly modeled uncertainties, and errors due to linearization of non–linear effect, create a great challenging task of determining "best" models for a dynamic system. It is recognized that it is conceivable that better performance will be anticipated when uncertainties are modeled through stochastic multiplicative and additive noise terms. Optimal control strategies generated under all possible parameter variations will definitely create more robust control systems, under controllability and observability conditions, than those generated by the usual approaches [15]. To avoid ad hoc assumptions regarding "a priori" statistics, Hyland [13,14,15] used the maximum entropy principle to determine a priori probability assignment induced from available data. A main advantage of maximum entropy approach is that it sacrifices as little near–nominal performance as possible while securing performance insensitivity over the likely range of modelling errors.

The second issue addressed in this report is the reduction of the order of a higher order control plant. Usually, the principle is looking for a quadratically optimal but fixed–order compensator for a higher order plant in order to simplify implementation. Amongst the methods available in the literature, we studied methods developed by Hyland [16] and Wilson [34] in this project report.

In this report, we first improved the computer program for the maximum entropy principle adopted in Hyland's MEOP method [14] developed in the previous report. The new program then was tested against the testing problems ran by A. Gruzen [9]. It resulted very close match. Therefore, it is safe to say the program is successful.

The second part of this report is centered at the theme of model reduction. Two methods were examined: Wilson's model reduction method [34] and Hyland's optimal projection (OP) method [14]. Design a computer program for Hyland's OP method was attempted. Due to the difficulty encountered at the stage where a special matrix factorization technique is needed in order to obtain the required projection matrix, we were only able to have the program successively up to finding the LQG solution but not beyond. Apparently, a more thorough and deeper study of the OP method is needed.

Numerical results along with computer programs which employed ORACLS are given in this report.

This report is based on the final results of a special project conducted by Wan–Sik Choi who was a graduate student in the Mathematics Department at the University of Alabama. The project was supervised by Drs. Wei Shen Hsia and Stavros Belbas.

2. MAXIMUM ENTROPY MODELLING

2.1. Maximum Entropy Method

Consider a linear system:

$$\dot{X} = AX + BU + \omega_1$$
$$Y = CX + \omega_2 \tag{1}$$

where

$$X \in R^n,\ U \in R^m,\ Y \in R^\ell,\ A \in R^{nxn},\ B \in R^{nxm},\ C \in R^{\ell xm},$$

and

$$SD(\omega_1, \omega_2) = (v_1, v_2).$$

We seek to determine a dynamic compensator

$$\dot{Z} = A_c Z + FY$$
$$U = -KZ \tag{2}$$

where $Z \in R^n$, $A_c \in R^{nxn}$, $F \in R^{nx\ell}$ and $K \in R^{mxn}$ that minimizes the Quadratic Cost Function:

$$J = \int_0^\infty (X^T R_1 X + U^T R_2 U)\, dt \tag{3}$$

where R_1 and R_2 are penalty matrices. The maximum entropy [26,27] (ME) design approach [11,12,13,14,15] is used to minimize J in the presence of parameter uncertainties.

2.2. Stratonovich Correction

The stochastic integral $\int_a^b \Phi(x(t), t)\, dx(t)$ can be defined in two ways.

Ito Integral:

$$\int_a^b \Phi(x(t),t)dx(t) = \underset{\Delta \to 0}{\text{l.i.m.}} \sum_{j=1}^{N-1} \Phi(x(t_j),t_j)[x(t_{j+1}) - x(t_j)]$$

Stratonovich Integral:

$$\int_a^b \Phi(x(t),t)dx(t) = \underset{\Delta \to 0}{\text{l.i.m.}} \sum_{j=1}^{N-1} \Phi\left[\frac{x(t_j) + x(t_{j+1})}{2}, t_j\right][x(t_{j+1}) - x(t_j)]$$

where $\Delta = \max(t_{j+1} - t_j)$.

To find the relationship between two integrals, consider

$$D_\Delta = \sum_{j=1}^{N-1} \left[\Phi\left[\frac{x(t_j) + x(t_{j+1})}{2}, t_j\right] - \Phi(x(t_j), t_j)\right][x(t_{j+1}) - x(t_j)]$$

$$= \frac{1}{2} \sum_{j=1}^{N-1} \frac{\partial \Phi}{\partial x} [\{(1-\Theta)x(t_j) + \Theta x(t_{j+1})\}, t_j] [x(t_{j+1}) - x(t_j)]^2 , 0 \leq \Theta \leq \frac{1}{2}$$

It was shown by Stratonovich that with probability 1

$$\lim_{\Delta \to 0} D_\Delta = \frac{1}{2} \int \frac{\partial \Phi}{\partial x}(x,t)\, b(x,t)dt.$$

Therefore,

$$\underbrace{\int_a^b \Phi(x(t),t)dx(t)}_{\text{Stratonovich}} = \underbrace{\int_a^{b_*} \Phi(x(t),t)dx(t)}_{\text{Ito}} + \underbrace{\frac{1}{2} \int_a^b \frac{\partial \Phi}{\partial x} [x(t)]b[x(t),t]dt}_{\text{correction}} \tag{4}$$

where * denotes the integral in the sense of Ito.

The relationship for the stochastic differential equations is as follows.

Ito D.E.: $dx_t = m[x_t,t]dt + \Gamma[x_t,t]dy_t$

Stratonovich D.E.: $dx_t = m[x_t,t]dt + \frac{1}{2}\Gamma[x_t,t]\frac{\partial\ \Gamma[x_t,t]}{\partial x_t}dt + \Gamma[x_t,t]dy_t$

$$= \left\{m[x_t,t] + \underbrace{\frac{1}{2}\ \Gamma[x_t,t]\ \frac{\partial\ \Gamma}{\partial\ x_t}}_{\text{correction}}\right\} dt + \Gamma[x_t,t]\ dy_t$$

Above result was shown in [30] by using (4) and also proved in [35].

2.3. Stochastic Modelling of Errors

In most instances, the errors are made in the modelling process and some parameters may vary. Therefore, the actual system would be represented by

$$A_{\text{actual}} = A + \sum_{i=1}^{p} \alpha_i(t)\ Ai \qquad (5)$$

where i: set of uncorrelated uncertainties

$\alpha(t)$: zero–mean, unit intensity multiplicative white noise

A_i; Parameter error distribution matrices

B_{actual} and C_{actual} take a similar form.

Substituting (5) into $\dot{X}(t) = AX(t)$ yields

$$\dot{X}(t) = (A + \sum_{i=1}^{p} \alpha_i(t)A_i)\ X(t) \quad ; \quad \text{O.D.E.}$$

$$\Longrightarrow$$

$$dx_t = (A\ dt + \sum_{i=1}^{p} d\alpha_{it}A_i)X_t \quad ; \quad \text{Ito S.D.E}$$

$$= AX_i\ dt + \sum_{i=1}^{p} d\alpha_{it}A_i\ X_t \qquad (6)$$

By comparing (6) with I_{t_0} D.E. and Stratonovich D.E. we obtain

$$dX_t = \left\{ \left[A + \frac{1}{2} \sum_{i=1}^{p} A_i^2 \right] dt + \sum_{i=1}^{p} d\alpha_{it} A_i \right\} X_t : \text{ Stratonovich D.E.}$$

$$\Longrightarrow \text{ Stratonovich correction for } \dot{X}(t) = Ax(t) \text{ is } \frac{1}{2} \sum_{i=1}^{p} A_i^2$$

B_s and C_s take similar form.

2.4. Necessary Conditions for Optimality [10]

Necessary conditions take the form of two Riccati equations and two Lyapunov equations, all coupled by the stochastic parameters.

$$0 = PA_s + A_s^T P + \sum_{i=1}^{p} A_i^T PA_i - P_s^T R_{2s}^{-1} P_s + R_1 + \sum_{i=1}^{p} (A_i - Q_s V_{2s}^{-1} C_i)^T \hat{P} (A_i - Q_s V_{2s}^{-1} C_i)$$

$$0 = A_s Q + QA_s + \sum_{i=1}^{p} A_i Q A_i^T - Q_s V_{2s}^{-1} Q_s^T + V_1 + \sum_{i=1}^{p} (A_i - B_i R_{2s}^{-1} P_s) \hat{Q} (A_i - B_i R_{2s}^{-1} P_s)^T$$

$$0 = \hat{P} A_{Q_s} + A_{Q_s}^T \hat{P} + P_s^T R_{2_s}^{-1} P_s$$

$$0 = A_{P_s} \hat{Q} + \hat{Q} A_{P_s}^T + Q_s V_{2_s}^{-1} Q_s^T$$

where $A_s = A + \frac{1}{2} \sum_{i=1}^{p} A_i^2$, $B_s = B + \frac{1}{2} \sum_{i=1}^{p} A_i B_i$, $C_s = C + \frac{1}{2} \sum_{i=1}^{p} C_i A_i$

$$R_{2s} = R_2 + \sum_{i=1}^{p} B_i^T (P + \hat{P}) B_i$$

$$V_{2s} = V_2 + \sum_{i=1}^{p} C_i (Q + \hat{Q}) C_i^T$$

$$P_s = B_s^T P + \sum_{i=1}^{p} B_i^T (P + \hat{P}) A_i$$

$$Q_s = Q C_s^T + \sum_{i=1}^{p} A_i (Q + \hat{Q}) C_i^T$$

$$A_{Qs} = A_s - Q_s V_{2s}^{-1} C_s$$

$$A_{ps} = A_s - B_s R_{2s}^{-1} P_s$$

The compensator matrices are,

$$A_c = A_s - Q_s V_{2s}^{-1} C_s - B_s R_{2s}^{-1} P_s + Q_s V_{2s}^{-1} D R_{2s}^{-1} P_s$$
$$F = Q_s V_{2s}^{-1}$$
$$K = R_{2s}^{-1} P_s$$

2.5. Algorithm

Compute F_p,F_q • generate a stabilizing gain matrix (F) for initializing the solution of Riccati eq.

Solve for LQG, P, Q • Solve Riccati eqs without having parameter uncertainties – uncoupled eqs.

Begin Interations with LQG P, Q

Solves P – Riccati

no P converges ? $\| P_i| - | P_{i-1}\| < \epsilon_p$? where $|\cdot|$ is a Euclidean Norm.

Solves Q–Riccati

no Q converges ? $\| Q_i| - | Q_{i-1}\| < \epsilon_q$?

Solves $\hat{P}$–Lyapunov

no $\hat{P}$ converges ? $\| \hat{P}_i| - | \hat{P}_{i-1}\| < \epsilon_{\hat{p}}$

Solves $\hat{Q}$–Lyapunov • No need to iterate $\hat{Q}$–Lyapunov because parameter doesn't include $\hat{Q}$

no $\hat{P}$,$\hat{Q}$ converge ? $\| \hat{P}_i| + | \hat{Q}_i| - \{| \hat{P}_{i-1}|\}| < \epsilon$?

Form A_c, F, k • Compensator matrices

2.6. Solution of Riccati equation and Lyapunov equation

As we have seen in the necessary condition of model reductions and Maximum Entropy Method, the necessary conditions consist of Lyapunov equations or coupled Riccati and Lyapunov equations.

Therefore solution of Riccati and Lyapunov is required for the design of control system. A lot of algorithm [8,18,24,28,31,32] were proposed in the past.

In this section, algorithms which empoloyed for this special project are briefly discussed.

Kleinman [19] proposed an algorithm which is based on the method of successive substitution to solve the algebraic Riccati equation.

Consider the linear time–invariant system.

$$\dot{X}(t) = AX(t) + BU(t) \quad X(0) = X_0$$

where [A,B] is completely controllable.

The cost to be minimized is

$$J(X_0;\, U(\cdot)) = \int_0^{\infty} [X'(t)\, C'\, CX(t) + U'(t)\, R\, U(t)]\, dt$$

where R is positive definite and [A,C] is completely observable. Necessary conditions for optimality are

$$U^*(X(t)) = -R^{-1}B'\, K\, X(t)$$

$$\text{and} \quad 0 = KA + A'K + C'C - KBR^{-1}B'K$$

where K is positive definite and

$$J(X_0;\, U^*(\cdot)) = \min_{U(\cdot)} J(X_0;\, U(\cdot)) = X'_0 K\, X.$$

Thus for arbitrary feedback law $U_L(X(t))$,

$$J(X_0;\, U_L(\cdot)) = X'_0 V_L\, X.$$

$$\Longrightarrow \qquad V_L = \int_0^{\omega} e^{(A-BL)'t}\, (C'C + L'RL) \cdot e^{(A-BL)t} dt$$

$\Longrightarrow$ V_L is finite if and only if $A - BL$ has eigenvalues with negative real parts.

$\Longrightarrow$ $0 = (A - BL)'V_L + V_L(A - BL) + C'C + L'RL.$

Kleinman's Theorem.

Let V_k, $k = 0,1,\cdots$, be the (unique) positive definite solution of the linear algebraic equation

$$0 = A_k' V_k + V_k A_k + C'C + L_k' R L_k$$

where, recursively,

$$L_k = R^{-1}B' V_{k-1}, \quad k = 1,2,\cdots$$

$$A_k = A - BL_k$$

and where L_0 is chosen such that $A_0 = A - BL_0$ has eigenvalues with negative real parts.

Then

1) $K \le V_{k+1} \le V_k \le \cdots \quad , \ k = 0,1,\cdots$

2.) $\lim\limits_{k \to \infty} V_k = K$

Note. In this project, stabilizing matrix L_0 is computed by CSTAB in ORACLS and Riccati equation is solved by RICNWT in ORACLS [1].

An algorithm for the solution of the matrix equation $AX + XB = C$ was proposed by Bartels and Stewart [6]. Above equation has a unique solution if and only if $\lambda_i^A + \lambda_J^B \neq 0$ ($i = 1,2,\cdots, m$; $j = 1,2,\cdots,n$) where λ_i^A and λ_j^B are eigenvalues of A and B respectively [2]. The method of solution is based on the reduction of A and B to the real schur form, i.e., block lower (upper) triangular form.

Let

$$AX + XB = C \tag{7}$$

and U, V be the orthogonal matrix.

Then

$$\begin{cases} B' = V^T B V \implies B = V B' V^T \\ B \longrightarrow \underset{\langle\text{Heusehalder's method}\rangle}{\text{upper Hessenberg form}} \longrightarrow \underset{\langle\text{QR algorithm}\rangle}{\text{upper real Schur form; } B'} \end{cases} \tag{8}$$

$$\begin{cases} A' = U^T A\, U \implies A = U\, A'\, U^T \\ \text{A}' \text{ (lower real Schur form) is obtained by reducing the} \\ \text{transparse of A to upper real Schur form and transposing back.} \end{cases} \tag{9}$$

$$C' = U^T\, C\, V \implies C = U\, C'\, V^T \tag{10}$$

Substituting (8), (9), (10) into (7) yields

$$U\, A'\, U^T X + X\, V\, B' V^T = U\, C' V^T$$

$$A'\, U^T X + U^T\, X\, V\, B'\, V^T = C'\, V^T$$

$$A'\, U^T X\, V + U^T\, X\, V\, B' = C'$$

$$A'\, X' + X'\, B' = C'$$

$$\begin{bmatrix} A'_{11} & & & 0 \\ A'_{z1} & A'_{zz} & & \\ \vdots & \vdots & \ddots & \\ A'_{p1} & A'_{p2} & \cdots & A'_{pp} \end{bmatrix} \begin{bmatrix} x'_{11} & \cdots & x'_{1q} \\ \vdots & & \vdots \\ \vdots & & \vdots \\ x'_{p1} & \cdots & x'_{pq} \end{bmatrix} + \begin{bmatrix} x'_{11} & \cdots & x'_{1q} \\ \ddots & & \\ D & \ddots & \\ x'_{p1} & \cdots & x'_{pq} \end{bmatrix} \begin{bmatrix} B'_{11} & B'_{12} & \cdots & B'_{1q} \\ & B'_{22} & \cdots & B'_{22q} \\ & & \ddots & \vdots \\ 0 & & & B'_{qq} \end{bmatrix}$$

$$= \begin{bmatrix} C'_{11} & \cdots & C'_{1q} \\ \vdots & & \vdots \\ \vdots & & \vdots \\ C'_{p1} & \cdots & C'_{pq} \end{bmatrix}$$

$$\implies A'_{kk} X'_{k\ell} + X'_{k\ell} B'_{\ell\ell} = C'_{k\ell} - \sum_{j=1}^{k-1} A'_{kj} X'_{j\ell} - \sum_{i=1}^{\ell-1} X'_{ki} B'_{i\ell}\,,\ k = 1,2,\cdots,p\,,\ k = 1,2,\cdots,q \tag{11}$$

Equation (11) can be solved successively for $X'_{k\ell}$. Let the right side of (11) be D. Since the block matrices A'_{kk} and $B'_{\ell\ell}$ are of order at most two, we are again required to solve the matrix equation of the form (7).

Writing (11) in matrix form gives

$$\underbrace{\begin{bmatrix} a'_{11} & a'_{12} \\ a'_{21} & a'_{22} \end{bmatrix}}_{A'_{kk}} \underbrace{\begin{bmatrix} x'_{11} & x'_{12} \\ x'_{21} & x'_{22} \end{bmatrix}}_{X'_{k\ell}} + \begin{bmatrix} x'_{11} & x'_{12} \\ x'_{21} & x'_{22} \end{bmatrix} \underbrace{\begin{bmatrix} b'_{11} & b'_{12} \\ b'_{21} & b'_{22} \end{bmatrix}}_{B'_{\ell\ell}} = \underbrace{\begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix}}_{\substack{\text{Right side} \\ \text{of (11)}}}$$

$$\Rightarrow \quad \begin{bmatrix} a'_{11} + b'_{11} & a'_{12} & b'_{21} & 0 \\ a'_{21} & a'_{22} + a'_{11} & 0 & b'_{21} \\ b'_{12} & 0 & a'_{11} + b'_{22} & a'_{12} \\ 0 & b'_{12} & a'_{21} & a'_{22} + b'_{22} \end{bmatrix} \begin{bmatrix} x'_{11} \\ x'_{21} \\ x'_{12} \\ x'_{22} \end{bmatrix} = \begin{bmatrix} d_{11} \\ d_{21} \\ d_{12} \\ d_{22} \end{bmatrix} \qquad (12)$$

$X'_{k\ell}$ is obtained from (12). Then the solution of (7) is given by $X = U X' V^T$.

Note. In this project, Lyapunov equation is solved by BARSTW in ORACLS [1].

2.7. Numerical Example for Maximum Entropy Method

The following system posed by Doyle [9] was solved by Gruzen [10]. In this project some problem is solved for comparison of numerical results.

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 + \Delta b \end{bmatrix} U + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \omega$$

$$Y = [1 \;\; 0] \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + V$$

$$R_1 = \Theta \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \; R_2 = 1$$

$$V_1 = \mu \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \; V_2 = 1$$

Θ, μ: parameters related with the gain margin

Parameter uncertainty distribution matrices:

$$A_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \; B_1 = \begin{bmatrix} 0 \\ \beta \end{bmatrix}, \; C_1 = [0, \;0]$$

Note: $\Theta = \mu = 60$, $0.93 \leq 1 + \Delta b \leq 1.01$

$0 \leq \beta \leq 0.2$, size 0.05 is used.

Necessary conditions for this example are

$$0 = PA_s + A_s^T P - PB_s^T R_{2s}^T B_s^T P + R_1$$
$$0 = A_s Q + QA_s^T - QC_s^T V_{2s}^i C_s Q + V_1 + (B_1 R_{2s}^{-1} P_s) \hat{Q} (B_1 R_{2s}^{-1} P_s)^T$$
$$0 = \hat{P} A_{Qs} + A_{Qs}^T + P_s^T R_{2s}^{-1} P_s$$
$$0 = A_{p_s} \hat{Q} + \hat{Q} A_{p_s}^T + Q_s V_{2s}^{-1} Q_s^T$$

where

$$A_s = A, \; B_s = B, \; C_s = C, \; R_{2s} = R_2 + B_1^T (P + \hat{P}) B_1,$$

$$V_{2s} = V_2, \; P_s = B_s^T P, \; Q_s = QC_s^T \;, \; A_{Qs} = A_s - Q_s V_{2s}^{-1} C_s,$$

$$A_{ps} = A_s - B_s R_{2s}^{-1} P_s \, .$$

The compensator matrices are,

$$A_c = A_s - Q_s V_{2s}^{-1} C_s - B_s R_{2s}^{-1} P_s$$

$$F = Q_s V_{2s}^{-1}$$

$$K = R_{2s}^{-1} P_s$$

Table 1. Numerical Results *Column II: Results for this project

β ⟨Disturbance in Matrix B_1⟩	Compensator Gains						Remark
	A_c		F		K^T		
	I	II	I	II	I	II	
0 (LQG)	$\begin{bmatrix} -9 & 1 \\ -20 & -9 \end{bmatrix}$	$\begin{bmatrix} -9 & 1 \\ -20 & -9 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	$\begin{bmatrix} 10 \\ 10 \end{bmatrix}$	same results From II refer PP. 46
.05	$\begin{bmatrix} -9.255 & 1 \\ -20.69 & -7.356 \end{bmatrix}$	$\begin{bmatrix} -9.276 & 1 \\ -22.2 & -8.671 \end{bmatrix}$	$\begin{bmatrix} 10.26 \\ 12.33 \end{bmatrix}$	$\begin{bmatrix} 10.27 \\ 12.52 \end{bmatrix}$	$\begin{bmatrix} 8.356 \\ 8.356 \end{bmatrix}$	$\begin{bmatrix} 9.68 \\ 9.68 \end{bmatrix}$	refer PP. 46
.10	$\begin{bmatrix} -9.662 & 1 \\ -23.36 & -6.178 \end{bmatrix}$	$\begin{bmatrix} -9.712 & 1 \\ -25 & -7.326 \end{bmatrix}$	$\begin{bmatrix} 10.66 \\ 16.18 \end{bmatrix}$	$\begin{bmatrix} 10.71 \\ 16.67 \end{bmatrix}$	$\begin{bmatrix} 7.178 \\ 7.178 \end{bmatrix}$	$\begin{bmatrix} 8.325 \\ 8.325 \end{bmatrix}$	refer PP. 47
.15	$\begin{bmatrix} -10.18 & 1 \\ -27.73 & -5.368 \end{bmatrix}$	$\begin{bmatrix} -10.18 & 1 \\ -27.72 & -5.37 \end{bmatrix}$	$\begin{bmatrix} 11.18 \\ 21.37 \end{bmatrix}$	$\begin{bmatrix} 11.18 \\ 21.34 \end{bmatrix}$	$\begin{bmatrix} 6.368 \\ 6.368 \end{bmatrix}$	$\begin{bmatrix} 6.371 \\ 6.371 \end{bmatrix}$	almost same results. refer PP. 48
.20	$\begin{bmatrix} -10.89 & 1 \\ -34.51 & -4.741 \end{bmatrix}$	$\begin{bmatrix} -10.74 & 1 \\ -31.81 & -3.63 \end{bmatrix}$	$\begin{bmatrix} 11.89 \\ 28.77 \end{bmatrix}$	$\begin{bmatrix} 11.74 \\ 27.18 \end{bmatrix}$	$\begin{bmatrix} 5.741 \\ 5.741 \end{bmatrix}$	$\begin{bmatrix} 4.626 \\ 4.626 \end{bmatrix}$	refer PP. 49

Note: 1) Column I is a numerical result obtained by A. Gruzen.
Column II is a numerical result obtained by this project.

2.8. Discussions on ME method

As shown in table I, matrix K decreases as β ⟨disturbance⟩ in matrix B_1 increases. This is because $K = R_{2s}^{-1} P_s$, $R_{2s} = R_2 + B_1^T (P + \hat{P})B_1$ and similarly for matrix F but F increases as β increases.

When $\beta = 0$ ⟨LQG case⟩, the two results (A.G. & N.R.) are exactly same. But for $\beta \neq 0$ best results obtained for $\beta = .15$. Differences in numerical results between A.G. & N.R. are possibly occurred from the value of Δb. (In this project $\Delta b = 0$ is used, but A. Gruzen doesn't show the value of Δb which he was used).

As a whole, the results are pretty close each other. Therefore, this indirectly verifies that "ME FORTRAN" provides correct answers. And it supports the fact that ORACLS is a good design package for designing controllers.

3. MODEL REDUCTION: WILSON'S METHOD [34]

3.1. Problem Statement

Given an nth – order system

$$\dot{X} = AX + BU \tag{13}$$

$$Y = HX, \tag{14}$$

find an rth – order reduced system

$$\dot{X}_r = A_r X_r + B_r U \tag{15}$$

$$Y_r = H_r X_r. \tag{16}$$

The input vector U(t) will be taken as a white noise, i.e.,

$$E(t)] = 0$$

$$E[U(t)U^T(s)] = N\delta(t-s).$$

The cost function to be minimized is

$$J = \lim_{t \to \infty} E[e^T(t)\, Q\, e(t)] \tag{17}$$

where e is the reduction error, $e = y - y_r$ and

Q is positive definite. Without loss of generality assume Q is m x m idenity matrix.

<u>Note</u>. where A, B, H are n x n, n x p, m x n matrices,

A_r, B_r, H_r are r x r, r x p, m x r matrices,

x, y are n x 1, m x 1 vectors,

x_r, y_r are r x 1, m x 1 vectors,

U is p x 1 vector.

3.2. Necessary conditions for optimum

$$A_r = \Theta_1 A \Theta_2 \tag{18}$$

$$B_r = \Theta_1 B \tag{19}$$

$$H_r = H \Theta_2 \tag{20}$$

where $\Theta_1 \triangleq -P_{22}^{-1} P_{12}^T$ and $\Theta_2 \triangleq R_{12} R_{22}^{-1}$.

$$\Theta_1 \Theta_2 = I_r \tag{21}$$

$$FR + RF^T + S = 0 \tag{22}$$

$$F^TP + PF + M = 0 \tag{23}$$

3.3. Derivation of Necessary Conditions

Equation (13) - (16) may be written as

$$\dot{Z} = FZ + GU \tag{24}$$

where $Z = \begin{bmatrix} X \\ X_r \end{bmatrix}$, $F = \begin{bmatrix} A & 0 \\ 0 & A_r \end{bmatrix}$, $G = \begin{bmatrix} B \\ B_r \end{bmatrix}$.

From (17)

$$J = \lim_{t \to \infty} E[e^T Q e]$$

$$= \lim_{t \to \infty} E[e^T e] \text{ since we assumed } Q = I_m$$

$$= \lim_{t \to \infty} E[(Y - Y_r)^T (Y - Y_r)]$$

$$= \lim_{t \to \infty} E[(HX - H_r X_r)^T (HX - H_r X_r)]$$

Now,

$$(HX - H_r X_r)^T (HX - H_r X_r)$$

$$= X^T H^T HX - X^T H^T H_r X_r - X_r^T H_r^T HX + X_r^T H_r^T H_r X_r$$

$$= X^T H^T HX - X_r^T H_r^T HX - X^T H^T H_r X_r + X_r^T H_r^T H_r X_r$$

$$= \begin{bmatrix} X^T H^T H - X_r^T H_r^T H & - X^T H^T H_r + X_r^T H_r^T H_r \end{bmatrix} \begin{bmatrix} X \\ X_r \end{bmatrix}$$

$$= \begin{bmatrix} X^T & X_r^T \end{bmatrix} \underbrace{\begin{bmatrix} H^T H & - H^T H_r \\ - H_r^T H & H_r^T H_r \end{bmatrix}}_{M} \begin{bmatrix} X \\ X_r \end{bmatrix}$$

$= Z^T M Z.$

Thus,

$$J = \lim_{t \to \infty} E[Z^T M Z]$$

$$= \text{trace (RM)} \tag{25}$$

where $R = \lim_{t \to \infty} E[Z(t)\, Z^T(t)]$

Let, $r(t) = E[Z(t)\, Z^T(t)]$.

Then, $\dot{r}(t) = E[\dot{Z}(t)\, Z^T(t) + Z(t)\, \dot{Z}^T(t)]$

$= E[\dot{Z}(t)\, Z^T(t)] + E[Z(t)\, \dot{Z}^T(t)]$.

Since $\dot{Z}^T = Z^T F^T + U^T G^T$,

$$\begin{aligned}
\dot{r}(t) &= E[(FZ + GU)Z^T] + E[Z(Z^T F^T + U^T G^T)] \\
&= FE[ZZ^T] + GE[UZ^T] + E[ZZ^T]F^T + E[ZU^T]G^T \\
&= F\, r(t) + r(t)\, F^T + GE[UZ^T] + E[ZU^T]G^T .
\end{aligned} \tag{26}$$

But,

$$Z(t) = \Phi(t,t_o)\, Z(t_o) + \int_{t_0}^{k} \Phi(t,\lambda)\, G(\lambda)\, U(\lambda)\, d\lambda$$

where $\Phi(t,t)$ is the state transition matrix.

Thus,

$$E[UZ^T] = \underbrace{E[U(t)\, Z^T(t_o)]}_{0,\ \text{uncorrelated}}\, \Phi^T(t,t_o) + \int_{t_0}^{t} E[U(t)\, U^T(\lambda)]\, G^T\, \Phi^T(t,\lambda)\, d\lambda$$

$$= \int_{t_0}^{t} N\, \delta\, (t - \lambda)\, G^T\, \Phi^T\, (t,\lambda)\, d\lambda \tag{27}$$

$$E[ZU^T] = \Phi(t,t_o)\, \underbrace{E[Z(t_o)\, U^T(t)]}_{0} + \int_{t_0}^{t} \Phi(t,\lambda)\, G(\lambda)\, E[U(\lambda)U^T(t)]\, d\lambda$$

$$= \int_{t_0}^{t} \Phi(t,\lambda)\, G(\lambda)\, N\, \delta\,(\lambda - t)\, d\lambda \tag{28}$$

Substituting (27) and (28) into (26) yields

$$\dot{r}(t) = Fr(t) + r(t)\, F^T + \int_{t_0}^{t} GN\delta\,(t-\lambda)\, G^T\Phi^T(t,\lambda)d\lambda + \int_{t_0}^{t} \Phi(t,\lambda)G(\lambda)\, N\delta\,(\lambda - t)\, G^T d\lambda$$

$$= Fr(t) + r(t)\, F^T + \frac{1}{2}\, G\, N\, G^T\Phi^T(t,t) + \frac{1}{2}\,\Phi(t,t)\, G\, N\, G^T$$

$$= Fr(t) + r(t)\, F^T + G\, N\, G^T\,.$$

Since $R = \lim_{t\to\infty} r(t)$, $\quad FR + RF^T + G\, N\, G^T = 0.$

Let $S = G\, N\, G^T = \begin{bmatrix} BNB^T & BNB_r^T \\ B_rNB^T & B_rNB_r^T \end{bmatrix}$.

Then,

$$FR + RF^T + S = 0 \tag{29}$$

To minimize (25) subject to (29) form the

Lagrangian

$$L = tr[\lambda RM] + (FR + RF^T + S)P.$$

$$\frac{\partial\ L}{\partial\ R} = 0 \Longrightarrow\ \lambda M + F^TP + PF = 0$$

Let $\lambda = 1$. Then

$$F^TP + PF + M = 0 \tag{30}$$

By comparing (30) with (29) we may write

$$J = \text{trace}\ (PS) \tag{31}$$

Let the symmetric matrices P and R be partitioned as

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix}, \quad R = \begin{bmatrix} R_{11} & R_{12} \\ R_{12}^T & R_{22} \end{bmatrix}.$$

Differentiating J with respect to any parameter β,

$$\frac{\partial J}{\partial \beta} = 2\,\mathrm{tr}\left[\frac{\partial F}{\partial \beta} RP\right] + \mathrm{tr}\left[\frac{\partial S}{\partial \beta} P\right] + \mathrm{tr}\left[\frac{\partial M}{\partial \beta} R\right]. \tag{32}$$

To find A_r, obtain derivative of J with respect to a_r using (32). Then

$$\frac{\partial J}{\partial a_r} = 2\,\mathrm{tr}\left[\frac{\partial F}{\partial a_r} RP\right] + \mathrm{tr}\left[\frac{\partial S}{\partial a_r} P\right] + \mathrm{tr}\left[\frac{\partial M}{\partial a_r} R\right]$$

$$= 2\,\mathrm{tr}\left[\begin{bmatrix} 0 & 0 \\ 0 & \dfrac{\partial A_r}{\partial a_r} \end{bmatrix} RP\right] \quad \text{where } R = \begin{bmatrix} R_{11} & R_{12} \\ R_{12}^T & R_{22} \end{bmatrix} \text{ and } P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix}$$

$$= 2\,\mathrm{tr}\left[\begin{bmatrix} 0 & 0 \\ \dfrac{\partial A_r}{\partial a_r} R_{12}^T & \dfrac{\partial A_r}{\partial a_r} R_{22} \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix}\right]$$

$$= 2\,\mathrm{tr}\left[\begin{bmatrix} 0 & 0 \\ \dfrac{\partial A_r}{\partial a_r}(R_{12}^T P_{11} + R_{22} P_{12}^T) & \dfrac{\partial A}{\partial a_r}(R_{12}^T P_{12} + R_{22} P_{22} \end{bmatrix}\right]$$

$$= 2\,\mathrm{tr}\left\{\frac{\partial A}{\partial a_r}(R_{12}^T P_{12} + R_{22} P_{22}\right\}$$

$$\frac{\partial J}{a_r} = 0 \Longrightarrow R_{12}^T P_{12} + R_{22} P_{22} = 0 \tag{33}$$

$$\Longrightarrow P_{12}^T R_{12} + P_{22} R_{22} = 0$$

$$\therefore P_{22}^{-1} P_{12}^T R_{12} + R_{22} = 0 \tag{34}$$

From (29)

$$\begin{bmatrix} A & 0 \\ 0 & A_r \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ R_{12}^T & R_{22} \end{bmatrix} + \begin{bmatrix} R_{11} & R_{12} \\ R_{12}^T & R_{22} \end{bmatrix} \begin{bmatrix} A^T & 0 \\ 0 & A_r^T \end{bmatrix} + \begin{bmatrix} BNB^T & BNB_r^T \\ B_r NB^T & B_r NB_r^T \end{bmatrix} = 0$$

$$\begin{bmatrix} AR_{11} & AR_{12} \\ A_r R_{12}^T & A_r R_{22} \end{bmatrix} + \begin{bmatrix} R_{11} A^T & R_{12} A_r^T \\ R_{12}^T A^T & R_{22} A_r^T \end{bmatrix} + \begin{bmatrix} BNB^T & BNB_r^T \\ B_r NB^T & B_r NB_r^T \end{bmatrix} = 0$$

$$\left.\begin{aligned} AR_{12} + R_{12} A_r^T + BNB_r^T = 0 \\ A_r R_{22} + R_{22} A_r^T + B_r NB_r^T = 0 \end{aligned}\right\} \tag{35}$$

But $B_r = - P_{22}^{-1} P_{12}^T B$.

Thus (35) becomes

$$AR_{12} + R_{12} A_r^T - BNB^T P_{12} P_{22}^{-T} = 0 \tag{36}$$

$$A_r R_{22} + R_{22} A_r^T + P_{22}^{-1} P_{12}^T BNB^T P_{12} P_{22}^{-T} = 0 \tag{37}$$

Now, $P_{22}^{-1} P_{12}^T$ (36) + (37) gives

$$P_{22}^{-1} P_{12}^T AR_{12} + A_r R_{22} + \underbrace{(P_{22}^{-1} P_{12}^T R_{12} + R_{22})}_{0, \text{ by } (34)} A_r^T = 0$$

$\Longrightarrow$

$A_r = - P_{22}^{-1} P_{12}^T A\, R_{12} R_{22}^{-1} \quad \longleftarrow \quad$ same as sq. (18)

To find B_r, $\frac{\partial J}{\partial b_r} = 0.$

$$\frac{\partial J}{\partial b_r} = 2\,\mathrm{tr}\left[\frac{\partial F}{\partial b_r} RP\right] + \mathrm{tr}\left[\frac{\partial S}{\partial b_r} P\right] + \mathrm{tr}\left[R\frac{\partial M}{\partial b_r}\right]$$

$$= \mathrm{tr}\left[P\frac{\partial S}{\partial b_r}\right] = \mathrm{tr}\left[P\frac{\partial}{\partial b_r}\begin{bmatrix} BNB^T & BNB_r^T \\ B_rNB^T & B_rNB_r^T \end{bmatrix}\right]$$

$$= \mathrm{tr}\left[\begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix}\begin{bmatrix} 0 & BN \\ BN & 2B_rN \end{bmatrix}\right] = \mathrm{tr}\begin{bmatrix} P_{12}BN & P_{11}BN + 2P_{12}B_rN \\ P_{22}BN & P_{12}^TBN + 2P_{22}B_rN \end{bmatrix}$$

$$= \mathrm{tr}\,(P_{12}BN + P_{12}^TBN + 2P_{22}B_rN) = \mathrm{tr}\,(P_{12}^TBN + P_{12}^TBN + 2P_{22}B_rN)$$

$$\Longrightarrow\ 2P_{22}B_rN = -2P_{12}^TBN$$

$$B_r = \underbrace{-P_{22}^{-1}P_{12}^T}_{\Theta_1}B = \Theta_1 B \longleftarrow \text{same as (19)}$$

To find H_r, $\frac{\partial J}{\partial h_r} = 0.$

$$\frac{\partial J}{\partial h_r} = \mathrm{tr}\left[\frac{\partial M}{\partial h_r} R\right] = \mathrm{tr}\left[\frac{\partial}{\partial h_r}\begin{bmatrix} H^TH & H^TH_r \\ -H_r^TH & H_r^TH_r \end{bmatrix} R\right]$$

$$= \mathrm{tr}\left[\begin{bmatrix} 0 & -H \\ -H & 2H_r \end{bmatrix}\begin{bmatrix} R_{11} & R_{12} \\ R_{12}^T & R_{22} \end{bmatrix}\right] = \mathrm{tr}\,(-HR_{12}^T - HR_{12} + 2H_rR_{22})$$

$$= \mathrm{tr}\,(-2HR_{12} + 2H_rR_{22})$$

$$\Longrightarrow H_r = \underbrace{HR_{12}R_{22}^{-1}}_{\Theta_2} = H\Theta_2 \longleftarrow \text{same as (20)}$$

From (2.21), $R_{12}^T P_{12} = -R_{22}P_{22}$.

$$\Longrightarrow \qquad P_{12}^T R_{12} = -P_{22}R_{22}.$$

Now,

$$\begin{aligned}\Theta_1\,\Theta_2 &= -P_{22}^{-1}P_{12}^T R_{12}R_{22}^{-1} \\ &= -P_{22}^{-1}(-P_{22}R_{22})R_{22}^{-1} \\ &= I_r \qquad \text{same as (21)}\end{aligned}$$

When the conditions on A_r, B_r and H_r {(18), (19), (20)} substituted into eqns. (29) and (30), a set of nonlinear equations in the unknown matrices Θ_1 and Θ_2 is obtained. Namely,

$$R_{22}\Theta_2^T A^T \Theta_1 + \Theta_1 A \Theta_2 R_{22} + H\Theta_2 N \Theta_2^T H^T = 0$$

$$P_{22}\Theta_1 A \Theta_2 + \Theta_2^T A^T \Theta_1^T P_{22} + \Theta_2^T H^T H \Theta_2 = 0.$$

An explicit solution for Θ_1 and Θ_2 is not apparently possible. Θ_1 and Θ_2 are nonunique, in the sense that the output of the reduced model is invariant under any nonsingular transformation T.

An algorithm to solve this optimum reduced order model problem was presented by Mishra and Wilson [22].

3.4. Algorithm [22]

Step 1: Choose the matrices Q and N

Step 2: Choose a value for the parameter Δ satisfying $0 < \Delta \leq 1$. Normally, without prior knowledge choose $\Delta = 1$.

Step 3: Make initial guesses for the matrices A_r and B_r, such that the pair (A_r, B_r) defines a completely controllable, strictly stable system.

Step 4: Solve the matrix equation $FT + RF^T + S = 0$

Step 5: Compute the matrix $\Theta_2 = R_{12}R_{22}^{-1}$

Step 6: Set $H_r = H\Theta_2$

Step 7: Solve the matrix equation $F^TP + PF + M = 0$

Step 8: Compute the matrix $\Theta_1 = -P_{22}^{-1}P_{12}^T$

Step 9: Set $B_r = \Theta_1 B$

Step 10: If B_r computed in Step 9 is not the same as B_r used in Step 4, then go to Step 4 using the B_r from Step 9. Otherwise, the B_r computed in Step 9 and the H_r computed in Step 6 are taken to be the optimum for the present A_r matrix. Step 9 and the H_r computed in Step 6 are taken to be the optimum for the present A_r matrix.

Step 11: Compute the error function J using the present A_r matrix and the optimum B_r and H_r defined in Step 10.

Step 12: Designate the present A_r matrix as A_r^{old} and the present value of the error function as J_0.

Step 13: Compute a new A_r.

$$A_r^{new} = \Delta\ \Theta_1 A \Theta_2 + (1 - \Delta)A_r^{old}$$

where Θ_1 and Θ_2 were used to compute the optimum B_r and H_r for A_r^{old}.

Step 14: If (A_r^{new}, B_r) is strictly stable controllable, then go to Step 15. Otherwise, reduce Δ and go to Step 13.

Step 15: For A_r^{new} and the optimum B_r for A_r^{old}, use Steps 4 to 10 until the optimum B_r and H_r are obtained for A_r^{new}.

Step 16: Compute J using A_r^{new}, B_r and H_r defined in Step 10. Designate the value of J as J_1.

Step 17: Test

(a) If $J_1 < J_0$: Go to Step 12

(b) If $J_1 > J_0$: Decrease Δ and go to Step 13

(c) If $J_1 = J_0$: If $\Theta_1\Theta_2 = I_r$ step. The triple (A_r^{new}, B_r, H_r) used to compute J_1 are the optimal reduced model. Otherwise decrease Δ and go to Step 13.

3.5. Derivatives of Cost Function.

$$J = \mathrm{tr}(RM) \tag{25}$$

$$FR + RF^T + S = 0 \tag{29}$$

$$J = \mathrm{tr}(PS) \tag{30}$$

$$F^TP + PF + M = 0 \tag{31}$$

$$\frac{\partial J}{\partial \beta} = \mathrm{tr}\left[\frac{\partial R}{\partial \beta} M\right] + \mathrm{tr}\left[R \frac{\partial M}{\partial \beta}\right], \text{ where } \beta \text{ is any parameter}$$

$$= -\,\mathrm{tr}\left[\frac{\partial R}{\partial \beta}(F^TP + PF)\right] + \mathrm{tr}\left[R\frac{\partial M}{\partial \beta}\right] \text{ since } M = -(F^TP + PF) \text{ from (31)}$$

$$= -2\,\mathrm{tr}\left[\frac{\partial R}{\partial \beta} PF\right] + \mathrm{tr}\left[R\frac{\partial M}{\partial \beta}\right]. \tag{38}$$

Differentiating (29) with respect to β,

$$\frac{\partial\ F}{\partial\ \beta} R + F \frac{\partial\ R}{\partial\ \beta} + \frac{\partial\ R}{\partial\ \beta} F^T + R \frac{\partial\ F^T}{\partial\ \beta} + \frac{\partial\ S}{\partial\ \beta} = 0 \tag{39}$$

Postmultiply (39) by P and taking the trace

$$\frac{\partial\ F}{\partial\ \beta} RP + F \frac{\partial\ R}{\partial\ \beta} P + \frac{\partial\ R}{\partial\ \beta} F^T P + R \frac{\partial\ F^T}{\partial\ \beta} P + \frac{\partial\ S}{\partial\ \beta} P = 0$$

$$\mathrm{tr} \left[\frac{\partial\ F}{\partial\ \beta} RP\right] + \underbrace{\mathrm{tr} \left[F \frac{\partial\ R}{\partial\ \beta} P\right]}_{\mathrm{tr} \left[\frac{\partial\ R}{\partial\ \beta} PF\right]} + \underbrace{\mathrm{tr} \left[\frac{\partial\ R}{\partial\ \beta} F^T P\right]}_{\mathrm{tr} \left[\frac{\partial\ R}{\partial\ \beta} PF\right]} + \underbrace{\mathrm{tr} \left[R \frac{\partial\ F^T}{\partial\ \beta} P\right]}_{\mathrm{tr} \left[\frac{\partial\ R}{\partial\ \beta} RP\right]} + \mathrm{tr} \left[\frac{\partial\ S}{\partial\ \beta} P\right] = 0$$

$$\text{So,} - 2\ \mathrm{tr} \left[\frac{\partial\ R}{\partial\ \beta} PF\right] = 2\ \mathrm{tr} \left[\frac{\partial\ F}{\partial\ \beta} RP\right] + \mathrm{tr} \left[\frac{\partial\ S}{\partial\ \beta} P\right] \tag{40}$$

Substituting (40) into (38),

$$\frac{\partial\ J}{\partial\ \beta} = 2\ \mathrm{tr} \left[\frac{\partial\ F}{\partial\ \beta} RP\right] + \mathrm{tr} \left[\frac{\partial\ S}{\partial\ \beta} P\right] + \mathrm{tr} \left[R \frac{\partial\ M}{\partial\ \beta}\right]$$

4. MODEL REDUCTION: HYLAND'S METHOD [16].

4.1. Problem Statement

Given the system

$$\dot{X} = AX + BU \tag{41}$$

$$Y = CX \tag{42}$$

find a reduced – order model

$$\dot{X}_r = A_r X_r + B_r U \tag{43}$$

$$Y_r = C_r X_r \tag{44}$$

which minimizes the model – reduction criterion

$$J(A_r,B_r,C_r) = \lim_{t\to\infty} E[(Y-Y_r)^T R(Y-Y_r)]. \tag{45}$$

The input U(t) is taken to be white noise with positive – definite intensity V.

<u>Note</u>. A, B, C: $n \times n$, $n \times m$, $\ell \times n$ matrices

A_r, B_r, C_r: $n_r \times n_r$, $n_r \times m$, $\ell \times n_r$ matrices

R, V: $\ell \times \ell$, $m \times m$ p.d. matrices

x, u, y, x_r, y_r: n, m, ℓ, n_r, ℓ dimensional vectors

$\rho(z)$: rank of matrix Z

Assumption: A, A_r stable.

4.2. Necessary Conditions for Optimum

$$A_r = \Gamma A G^T \tag{46}$$

$$B_r = \Gamma B \tag{47}$$

$$C_r = C G^T \tag{48}$$

$$\rho(\hat{Q}) = \rho(\hat{P}) = \rho(\hat{Q}\hat{P}) = N_r \tag{49}$$

$$0 = A\hat{Q} + \hat{Q}A^T + BVB^T - \gamma_\perp BVB^T \gamma_\perp^T \tag{50}$$

$$0 = A^T\hat{P} + \hat{P}A + C^TRC - \gamma_\perp^T C^T R C \gamma_\perp \tag{51}$$

where $G = Q_2^{-1} Q_{12}^T$, $\Gamma = -P_2^{-1} P_{12}^T$,

$\gamma = G^T \Gamma$, $\gamma_\perp = I_n - \gamma$.

$\Gamma G^T = I_{n_r}$

4.3. Derivation of Necessary Conditions

Introducing the augmented system

$$\dot{\tilde{X}} = \tilde{A}\,\tilde{X} + \tilde{B}U,$$

$$\tilde{Y} = \tilde{C}\,\tilde{X}$$

where

$$\tilde{X} = \begin{bmatrix} X \\ X_r \end{bmatrix}, \qquad \tilde{Y} = Y - Y_r$$

$$\tilde{A} = \begin{bmatrix} A & 0 \\ 0 & A_r \end{bmatrix}, \qquad \tilde{B} = \begin{bmatrix} B \\ B_r \end{bmatrix}, \qquad \tilde{C} = [C \quad -C_r] .$$

$$J(A_r, B_r, C_r) = \lim_{t \to \infty} E[(Y - Y_r)^T R (Y - Y_r)]$$

$$= \mathrm{tr}\ \tilde{Q}\tilde{R} \ \text{ where } \tilde{R} = \tilde{C}^T R \tilde{C} \text{ and } \tilde{Q} = \lim_{t \to \infty} E[\tilde{X}(t)\tilde{X}^T(t)]. \tag{52}$$

As shown in Wilson's Method (25) ~ (29) $\tilde{Q}$ is given by the unique solution of

$$0 = \tilde{A}\,\tilde{Q} + \tilde{Q}\,\tilde{A}^T + \tilde{V} \tag{53}$$

where $\tilde{V} = \tilde{B} V \tilde{B}^T$

To minimize (52) subject to (53), form the

Lagrangian $L(A_r, B_r, C_r, \tilde{Q}) = \mathrm{tr}[\lambda\tilde{Q}\,\tilde{R} + (\tilde{A}\,\tilde{Q} + \tilde{Q}\,\tilde{A}^T + \tilde{V})\tilde{P}]$

where $\lambda \geq 0$ and $\tilde{P} \in \mathbb{R}^{(n + n_r) \times (n + n_r)}$.

Expanding $L(A_r, B_r, C_r, \tilde{Q})$ gives

$$L = \mathrm{tr} \Big[\lambda(Q_1 C^T R C - Q_{12} C_r^T R C - Q_{12}^T C^T R C_r + Q_2 C_r^T R C_r)$$

$$+ A Q_1 P_1 + A Q_{12} P_{12}^T + A_r Q_{12}^T P_{12} + A_r Q_2 P_2$$

$$+ Q_1 A^T P_1 + Q_{12} A_r^T P_{12}^T + Q_{12}^T A^T P_{12} + Q_2 A_r^T P_2$$

$$+ B V B^T P_1 + B V B_r^T P_{12}^T + B_r V B^T P_{12} + B_r V B_r^T P_2 \Big].$$

And,

$$\tilde{Q} = \begin{bmatrix} Q_1 & Q_{12} \\ Q_{12}^T & Q_2 \end{bmatrix} , \ \tilde{P} = \begin{bmatrix} P_1 & P_{12} \\ P_{12}^T & P_2 \end{bmatrix} , \ \tilde{R} = \tilde{C}^T R \tilde{C} = \begin{bmatrix} C^T RC & -C^T RC_r \\ -C_r^T RC & C_r^T RC_r \end{bmatrix} ,$$

$$\tilde{V} = \tilde{B} V \tilde{B}^T = \begin{bmatrix} BVB^T & BVB_r^T \\ B_r VB^T & B_r VB_r^T \end{bmatrix} .$$

Now,

$$\frac{\partial L}{\partial \tilde{Q}} = 0.$$

$$\frac{\partial L}{\partial \tilde{Q}} = \begin{bmatrix} \dfrac{\partial L}{\partial Q_1} & \dfrac{\partial L}{\partial Q_{12}} \\ \dfrac{\partial L}{\partial Q_{12}^T} & \dfrac{\partial L}{\partial Q_2} \end{bmatrix}$$

$$= \begin{bmatrix} \lambda C^T RC + A^T P_1 + P_1 A & -\lambda C^T RC_r + A^T P_{12} + P_{12}^T A_r \\ -\lambda C_r^T RC + A_r^T P_{12}^T + P_{12}^T A & \lambda C_r^T RC_r + A_r^T P_2 + P_2 A_r \end{bmatrix}$$

$$= \lambda \begin{bmatrix} C^T RC & -C^T RC_r \\ -C_r^T RC & C_r^T RC_r \end{bmatrix} + \begin{bmatrix} A^T P_1 & -A^T P_{12} \\ A_r^T P_{12}^T & A_r^T P_2 \end{bmatrix} + \begin{bmatrix} P_1 A & P_{12} A_r \\ P_1^T A & P_2 A_r \end{bmatrix}$$

$$= \lambda\tilde{R} + \begin{bmatrix} A^T & 0 \\ 0 & A_r^T \end{bmatrix} \begin{bmatrix} P_1 & P_{12} \\ P_{12}^T & P_2 \end{bmatrix} + \begin{bmatrix} P_1 & P_{12} \\ P_{12}^T & P_2 \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & A_r \end{bmatrix}$$

$= \lambda\tilde{R} + \tilde{A}^T\tilde{P} + \tilde{P}\tilde{A}$

Thus, $\tilde{A}^T\tilde{P} + \tilde{P}\tilde{A} + \lambda\tilde{R} = 0$.

Without loss of generality, take $\lambda = 1$.

Then $\tilde{A}^T\tilde{P} + \tilde{P}\tilde{A} + \tilde{R} = 0$ (54)

$\frac{\partial L}{\partial A_r} = 0,$

$\frac{\partial L}{\partial A_r} = 2\, P_{12}^T Q_{12} + 2P_2Q_2$

Thus, $P_{12}^T Q_{12} + P_2Q_2 = 0 \implies Q_{12}^T P_{12} + Q_2P_2 = 0$ (55)

$\frac{\partial L}{\partial B_r} = 0.$

$\frac{\partial L}{\partial B_r} = P_{12}^T BV + P_{12}^T BV + 2P_2B_rV$

Thus, $2[P_{12}^T B + P_2B_r]V = 0$ (56)

$\frac{\partial L}{\partial C_r} = 0$,

$\frac{\partial L}{\partial C_r} = -\, RCQ_{12} - RCQ_{12} + 2RC_rQ_2$

Thus $2R[C_rQ_2 - CQ_{12}] = 0$ (57)

Define,

$$G = Q_2^{-1}Q_{12}^T \quad \text{and} \quad \Gamma = -P_2^{-1}P_{12}^T \ .$$

Then,

$$\Gamma G^T = -P_2^{-1}P_{12}^T Q_{12} Q_2^{-T} \ .$$

But from (55), $P_{12}^T Q_{12} = -P_2^T Q_2^T = -P_2 Q_2^T$.

Thus,

$$\Gamma G^T = -P_2^{-1}(-P_2 Q_2^T)Q_2^{-T} = I_{n_r}$$

From (56), $B_r = -P_2^{-1}P_{12}^T B = \Gamma B$

From (57), $C_r = CQ_{12}Q_2^{-1} = C(Q_2^{-T}Q_{12}^T)^T$, Q_2 is P.d.
$= C(Q_2^{-1}Q_{12}^T)^T = CG^T$.

Expanding (53) and (54) yields

$$0 = AQ_1 + Q_1A^T + BVB^T \tag{58}$$
$$0 = AQ_{12} + Q_{12}A_r^T + BVB_r^T \tag{59}$$
$$0 = A_rQ_2 + Q_2A_r^T + B_rVB_r^T \tag{60}$$
$$0 = A^TP_1 + P_1A + C^TRC \tag{61}$$
$$0 = A^TP_{12} + P_{12}A_r - C^TRC_r \tag{62}$$
$$0 = A_r^TP_2 + P_2A_r + C_r^TRC_r \tag{63}$$

Since A_r, B_r and C_r are independent of Q_1 and P_1, (58) and (61) can be ignored.

Define $\hat{Q} = Q_{12}Q_2^{-1}Q_{12}^T = Q_{12}G$ (64)

$$\hat{P} = P_{12}P_2^{-1}P_{12}^T = -P_{12}\Gamma. \tag{65}$$

Now $(64)\cdot\Gamma^T$ yields

$$\hat{Q}\Gamma^T = Q_{12}G\,\Gamma^T = Q_{12}(\Gamma G^T)^T = Q_{12}\,. \tag{66}$$

Similarily, from (65)

$$P_{12} = -\hat{P}G^T\,. \tag{67}$$

$$\Gamma\hat{Q}\Gamma^T = -P_2^{-1}P_{12}^TQ_{12}Q_2^{-1}Q_{12}^T(-P_{12}P_2^{-T})$$
$$= Q_2$$

Thus, $Q_2 = \Gamma\hat{Q}\Gamma^T$ (68)

Similarily, $P_2 = G\hat{P}G^T$ (69)

Substitute (47), (48), (66), ~ (69) into (59), (60), (62), (63)

$$0 = A\hat{Q}\Gamma^T + \hat{Q}\Gamma^TA_r^T + BVB^T\Gamma^T \tag{70}$$
$$0 = A_r\Gamma\hat{Q}\Gamma^T + \Gamma\hat{Q}\Gamma^TA_r^T + \Gamma BVB^T\Gamma^T \tag{71}$$
$$0 = A^T\hat{P}G^T + \hat{P}G^TA_r + C^TRCG^T \tag{72}$$
$$0 = A_r^TG\hat{P}G^T + G\hat{P}G^TA_r + GC^TRCG^T. \tag{73}$$

$(71) - \Gamma\cdot(70)$,

$$A_r\underbrace{\Gamma\hat{Q}\Gamma^T}_{Q_2} = \Gamma A\underbrace{\hat{Q}\Gamma^T}_{Q_{12}}$$

Thus, $A_r = \Gamma A Q_{12} Q_2^{-1} = \Gamma A G^T$

$$\gamma \hat{Q} = G^T \Gamma \hat{Q} = (-Q_{12}Q_2^{-1})(P_2^{-1}\underbrace{P_{12}^T)(Q_{12}}_{-P_2Q_2}Q_2^{-1}Q_{12}^T)$$

$$= Q_{12}Q_2^{-1}P_2^{-1}P_2Q_2Q_2^{-1}Q_{12}^T$$

$$= Q_{12}Q_2^{-1}Q_{12}^T = \hat{Q} \tag{74}$$

Similarily, $\hat{P}\gamma = \hat{P}$ (75)

Finally, $G^T \cdot (70)^T$ yields

$$\underbrace{G^T\Gamma}_{\gamma}\hat{Q}^T A^T + G^T \underbrace{A_r \Gamma}_{\Gamma A G^T} \hat{Q}^T + \underbrace{G^T \Gamma}_{\gamma} B V^T B^T = 0$$

$$\gamma \hat{Q} A^T + \underbrace{\underbrace{G^T \Gamma A G^T \Gamma \hat{Q}}_{\gamma A \gamma \hat{Q}}}_{} + \gamma B V\, B^T = 0 \quad , \quad \hat{Q} \text{ and V symmetric.}$$

$$\gamma A \underbrace{\gamma \hat{Q}}_{\hat{Q} \text{ by } (74)}$$

$$\gamma\, [A\hat{Q} + \hat{Q}A^T + BVB^T] = 0 \tag{76}$$

Similarily, $(72) \cdot \Gamma$ yields

$$[A^T\hat{P} + \hat{P}A + C^TRC]\,\gamma = 0 \tag{77}$$

$(76) + (76)^T + (76) \cdot \gamma$

$= \gamma A\hat{Q} + \gamma\hat{Q}A^T + \gamma BVB^T + \hat{Q}A^T\gamma^T + A\hat{Q}\gamma^T + BVB^T\gamma^T + \gamma A\hat{Q}\gamma + \gamma\hat{Q}A^T\gamma + \gamma BVB^T\gamma$

$= \hat{Q}A^T + A\hat{Q} + \gamma BVB^T + BVB^T\gamma^T + \gamma A\hat{Q}\gamma^T + \gamma\hat{Q}A^T\gamma^T + \gamma A\hat{Q}\gamma + \gamma\hat{Q}A^T\gamma + \gamma BVB^T\gamma$

$= A\hat{Q} + \hat{Q}A^T + \gamma BVB^T + BVB^T\gamma^T + \gamma(A\hat{Q} + \hat{Q}A^T)\gamma^T + \gamma(A\hat{Q} + \hat{Q}A^T)\gamma + \gamma BVB^T\gamma$

$= A\hat{Q} + \hat{Q}A^T + \gamma BVB^T + BVB^T\gamma^T - \gamma BVB^T\gamma^T$

$$= A\hat{Q} + \hat{Q}A^T + BVB^T - BVB^T + \gamma BVB^T I_n^T + I_n BVB^T \gamma^T - \gamma BVB^T \gamma^T$$

$$= A\hat{Q} + \hat{Q}A^T + BVB^T - (I_n BVB^T I_n^T - \gamma BVB^T I_n^T - I_n BVB^T \gamma^T + \gamma BVB^T \gamma^T)$$

$$= A\hat{Q} + \hat{Q}A + BVB^T - \gamma_\perp BVB^T \gamma_\perp$$

which is the same as (50)

Similarily,

$$(77) + (77)^T + \gamma^T_\cdot(77) = A^T\hat{P} + \hat{P}A + C^TRC - \gamma_\perp^T C^T RC\gamma_\perp$$

which is the same as (51).

A computer program has been designed (appendix 3) for this algorithm. Due to the difficulty of finding the projection matrix r through a matrix factorization process, the program only run successively up to obtaining an LQG solution. Apparently, more words and researchs need to be done in that area.

4.4. Algorithm ([17,7])

Step 1: Initialize $\gamma^{(0)} = I_n$.

Step 2: Solve for $\hat{Q}^{(K)}$, $\hat{P}^{(K)}$ from

$$0 = (A - \gamma^{(K)} A \gamma_\perp^{(K)})\, \hat{Q}^{(K)} + \hat{Q}^{(K)}(A - \gamma^{(K)} A \gamma_\perp^{(K)})^T + BVB^T$$

$$0 = (A - \gamma_\perp^{(K)} A \gamma^{(K)})^T\, \hat{P}^{(K)} + \hat{P}^{(K)}(A - \gamma_\perp^{(K)} A \gamma^{(K)}) + C^T RC$$

Step 3: Balance

$$\Phi^{(K)}\hat{Q}^{(K)}(\Phi^{(K)})^T = (\Phi^{(K)})^{-T}\, \hat{P}^{(K)}(\Phi^{(K)})^{-1} = \Sigma^{(K)},$$

$$\Sigma^{(K)} = \mathrm{diag}\,(\sigma_1^{(K)}, \dots, \sigma_n^{(K)}),\ \sigma_1^{(K)} \geq \sigma_2^{(K)} \geq \cdots \geq \sigma_n^{(K)} \geq 0$$

Step 4: If $K > 1$ check for convergence

$$e_k = \left[\frac{\mathrm{tr}(C^T RCW_c) - \mathrm{tr}\,(C^T RC\gamma^{(K)}\hat{Q}^{(K)}(\gamma^{(K)^T})}{\mathrm{tr}(C^T RCW_c)} \right]^{1/2}$$

If $|e_k - e_{k-1}| <$ tolerance then go to step 8), else continue.

Step 5: Select N_m eigenprojections

$$\Pi_{i_1}[\hat{\hat{Q}}^{(K)}\hat{\hat{P}}^{(K)}], \cdots, \Pi_{in_m}[\hat{\hat{Q}}^{(K)}\hat{\hat{P}}^{(K)}],$$

$$\Pi_i[\hat{\hat{Q}}^{(K)}\hat{\hat{P}}^{(K)}] \triangleq \Phi^{(K)}E_i(\Phi^{(K)})^{-1}.$$

Step 6: Update $\gamma^{(K+1)} = \sum_{r=1}^{N_m} \Pi_{i_r}[\hat{\hat{Q}}^{(K)}\hat{\hat{P}}^{(K)}]$

Step 7: Increment K and return to Step 2.

Step 8: Set $\hat{Q} = \gamma^{(\infty)}\hat{\hat{Q}}(\gamma^{(\infty)})^T$, $\hat{P} = (\gamma^{(\infty)})^T \hat{\hat{P}} \gamma^{(\infty)}$

4.5. Relationship between two methods

Wilson's Method	Hyland's Method
$\dot{X} = AX + BU$	$\dot{X} = AX + BU$
$Y = HX$	$Y = CX$
$\dot{X}_r = A_rX_r + B_rU$	$\dot{X}_r = A_rX_r + B_rU$
$Y_r = H_rX_r$	$Y_r = C_rX_r$
$J = \lim_{t\to\infty} E[(Y - Y_r)^T(Y - Y_r)]$	$J = \lim_{t\to\infty} E[(Y - Y_r)^TR(Y - Y_r)]$
$A_r = \Theta_1 A\, \Theta_2$	$A_r = \Gamma AG^T$
$B_r = \Theta_1 B$	$B_r = \Gamma B$
$H_r = H\Theta_2$	$C_r = CG^T$
$\Theta_1 = -P_{22}^{-1}$	$\Gamma = -P_2^{-1}P_{12}^T$
$\Theta_2 = R_{12}R_{22}^{-1}$	$G^T = Q_{12}Q_2^{-1}$
$\Theta_1\Theta_2 = I_r$	$\Gamma G^T = I_{n_r}$
	$\gamma = G^T\Gamma$
$FR + RF^T + S = 0$	$\tilde{A}\tilde{Q} + \tilde{Q}\tilde{A}^T + \tilde{V} = 0$

Wilson's Method	Hyland's Method
$F^TP + PF + M = 0$	$\tilde{A}^T\tilde{P} + \tilde{P}\tilde{A} + \tilde{R} = 0$
$S = \begin{bmatrix} BNB^T & BNB_r^T \\ B_rNB^T & B_rNB_r^T \end{bmatrix}$	$\tilde{V} = \begin{bmatrix} BVB^T & BVB_r^T \\ B_rVB^T & B_rVB_r^T \end{bmatrix}$
$M = \begin{bmatrix} H^TH & -H^TH_r \\ -H_r^TH & H_r^TH_r \end{bmatrix}$	$\tilde{R} = \begin{bmatrix} C^TRC & -C^TRC_r \\ -C_r^TRC & C_r^TRC_r \end{bmatrix}$
	$A\hat{Q} + \hat{Q}A^T + BVB^T - \gamma_\perp BVB^T\gamma_\perp^T = 0$
	$A^T\hat{P} + \hat{P}A + C^TRC - \gamma_\perp^T C^TRC\gamma_\perp = 0$
	where $\gamma_\perp = I_n - \gamma$
i) Θ_1 and Θ_2 depend upon the solutions of a pair of $(n + n_r) \times (n + n_r)$ Lyapunov equations [29, 30] whose coefficients and nonhomogeneous terms depend in aturn on A_r, B_r and H_r.	i) necessary to solve n x n Lyapunov equation [50, 51] which is independent of A_r, B_r, and C_r
	ii) Need eigenprojections to form $\gamma = \sum_{i=1}^{N_m} \Pi_i[\hat{Q}\ \hat{P}$
ii) Required to make initial guesses for A_r and B_r.	iii) Need (G, M, Γ) – factorization of $\hat{Q}\ \hat{P}$ to determine G and Γ.

REFERENCES

1. E.S. Armstrong, "ORACLS – A system for linear – quadratic – gaussian control law design," NASA Technical Paper 1106, 1978.

2. E.S. Armstrong, "A Design System for Linear Multivariate Control," Marcell Dekker, Inc., New York, 1980.

3. Karl J. Åström, "Introduction to stochastic control theory," Academic Press, 1970.

4. Michael Athans, "The matrix minimum principle," Information and Control 11, 592 – 606, 1968.

5. S. Barnett & R.G. Cameron, "Introduction to stochastic control theory," Academic Press, 1970.

6. R.H. Bartels and G.W. Stewart, "Algorithm 432 solution of the matrix equation AX + XB = C," Communications, ACM, 1972.

7. Dennis S. Bernstein and David C. Hyland, "Numerical solution of the optimal model reduction equations," AIAA Guidance and Control Conference, August 20 – 22, 1984/ Seattle, Washington.

8. John C. Doyle, "Guaranteed margins for LQG regulators," IEEE Trans. A.C., Vol. AC–23, No. 4 1978.

9. Alexander Gruzen, "Robust reduced – order control of flexible structures," Master Thesis, MIT March 1986.

10. D.C. Hyland, "Optimal Regulation of Structural Systems with Uncertain Parameters," MIT, Lincoln Laboratory, TR–551, Feb. 1981, DDC# AD–A099111/7.

11. D.C. Hyland, "Structural modelling and control design under incomplete parameter information: The maximum – entropy approach," NASA CP 2258, May, 1982.

12. D.C. Hyland, "Maximum Entropy Stochastic Approach to Control Design for Uncertain Structural Systems," American Control Conference, Arlington, VA. June, 1982.

13. D.C. Hyland, "Minimum Information Modelling of Structural Systems with Uncertain Parameters," Proceedings of the Workshop on Applications of Distributed System Theory to the Control of Large Space Structures, JPL, Pasadena, CA. July, 1982.

14. D.C. Hyland, "Application of the Maximum entropy/optimal projection control design approach for large space structures," Large Space Antenna Systems Technology – 1984 Conference, December 4 – 6, 1984.

15. David C. Hyland and Dennis S. Bernstein, "The optimal projection approach to model reduction and the relationship between the methods of Wilson and Moore," Proceedings of 23rd conference on decision and control, Las Vegas, NV, December 1984.

16. David C. Hyland and Dennis S. Bernstein, "The optimal projection equations for model reduction and the relationships among the methods of Wilson, Skelton, and Moore," IEEE Trans. Automat. Contr., Vol. AC – 30, No. 12, 1985.

17. Anthony Jameson, "Solution of the equation AX + XB = C by inversion of an M x M or N x N matrix," SIAM J. Appl. Math. Vol. 16, No. 5, 1968.

18. D.L. Kleinman, "On an iterative technique for Riccati equation computations," IEEE Trans., A.C., 1968.

19. Er – Chieh Ma, "A finite series solution of the matrix equation AX – XB = C," J. SIAM Appl. Math. Vol. 14, No. 3, 1966.

20. James L. Meisa and Andrew P. Sage, "An introduction to probability and stochastic processes," Prentice Hall, 1973.

21. P. Chr. Miller, "Solution of the matrix equations AX + XB = – Q and $S^T X + XS = -Q$," SIAM J. Appl. Math. Vol. 18. No. 3, 1970.

22. Richard K. Miller and Anthony N. Michael, "Ordinary differential equations," Academic Press, 1982.

23. R.N. Mishra and D.A. Wilson, "A new algorithm for optimal reduction of multivariable systems," INT. J. Control, 1980, Vol. 31, No. 3, 443 – 466.

24. Bruce C. Moore, "Principal Component analysis in linear systems: Controllability, observability, and model reduction," IEEE Trans. A.C., Vol. AC – 26, 1, 1981.

25. W. Murray Wanham, "Linear Multivariable control," Springer – Verlay, 1985.

26. Athanasias Papoulis, "Probability Random Variables, and Stochastic Processes," McGraw – Hill, 1984.

27. Fazlullah M. Reza, "An introduction to information theory," McGraw – Hill, 1961.

28. D. Rothschild and A. Jameson, "Comparison of four numerical algorithms for solving the Liapunov matrix equation," INT. J. Control, 1970, Vol. 11, No. 2, 181 – 198.

29. Andrew P. Sage and Chelsea C. White IV, "Optimum systems control 2nd edition,".

30. R.L. Stratonovich, "A new representation for stochastic integrals and equations," J. SIAM Control Vol. 4, No. 2 1966.

31. David R. Vaughn, "A negative exponential solution for the matrix Riccati equation," IEEE Trans. A.C. 1969.

32. David R. Vaughn, "A nonrecursine algebraic solution for the discrete Riccati equation," IEEE Trans. A.C. 1970.

33. H.B. Waites, S.M. Seltzer, and D.K. Tollison, "NASA/MSFC Ground Experiment for Large Space Structure Control Verification," NASA TM–86496, December, 1984.

34. Eugene Wang and Masha Zakai, "On the relation between ordinary and stochastic differential equations," Int. J. Engng Sci. Vol. 3, pp. 213 – 229, 1965.

35. D. A. Wilson, "Optimum solution of model – reduction problem," proc. IEE, Vol. 117, pp. 1161 – 1165, 1970.

APPENDIX 1

Numerical Results of M.E. Method

```
FORTVS ME ( GS OPT ( 2 )
VS FORTRAN COMPILER ENTERED.  22:45:04

**MAIN** END OF COMPILATION 1 ******

**SUB1** END OF COMPILATION 2 ******

**SUB5** END OF COMPILATION 3 ******

**SUB8** END OF COMPILATION 4 ******

**SUB9** END OF COMPILATION 5 ******

**SUB12** END OF COMPILATION 6 ******

**SUB13** END OF COMPILATION 7 ******
VS FORTRAN COMPILER EXITED.  22:45:07


GLOBAL TXTLIB VFORTLIB CMSLIB FORTUTIL
GLOBAL LOADLIB VFLODLIB
FILEDEF 5 DISK NME DATA
LOAD ME H ( START
EXECUTION BEGINS...

SPECIAL PROJECT : MAXIMUM ENTROPY ALGORITHM


       A  MATRIX        2 ROWS        2 COLUMNS
  1.0000000D+00   1.0000000D+00
  0.0000000D+00   1.0000000D+00

       B  MATRIX        2 ROWS        1 COLUMNS
  0.0000000D+00
  1.0000000D+00

       C  MATRIX        1 ROWS        2 COLUMNS
  1.0000000D+00   0.0000000D+00

       R  MATRIX        2 ROWS        2 COLUMNS
  1.0000000D+00   1.0000000D+00
  1.0000000D+00   1.0000000D+00

      R2  MATRIX        1 ROWS        1 COLUMNS
  1.0000000D+00

       V  MATRIX        2 ROWS        2 COLUMNS
  1.0000000D+00   1.0000000D+00
  1.0000000D+00   1.0000000D+00

      V2  MATRIX        1 ROWS        1 COLUMNS
  1.0000000D+00

      B1  MATRIX        2 ROWS        1 COLUMNS
  0.0000000D+00
  0.0000000D+00

*** MATRIX F FOR P-RICCATI ***
  8.0080020D+00   4.0020000D+00
```

```
*** MATRIX F FOR Q-RICCATI ***
 4.0020000D+00    8.0080020D+00

*** SOLUTION OF LQG P-RICCATI ***


PROGRAM TO SOLVE CONTINUOUS STEADY-STATE RICCATI EQUATION BY THE NEWTON ALGORITHM


     A     MATRIX        2 ROWS        2 COLUMNS
 1.0000000D+00    1.0000000D+00
 0.0000000D+00    1.0000000D+00

     B     MATRIX        2 ROWS        1 COLUMNS
 0.0000000D+00
 1.0000000D+00

     Q     MATRIX        2 ROWS        2 COLUMNS
 6.0000000D+01    6.0000000D+01
 6.0000000D+01    6.0000000D+01

H IS AN IDENTITY MATRIX


     R     MATRIX        1 ROWS        1 COLUMNS
 1.0000000D+00

INITIAL F MATRIX


     F     MATRIX        1 ROWS        2 COLUMNS
 8.0080020D+00    4.0020000D+00


FINAL VALUES OF P AND F AFTER    7 ITERATIONS TO CONVERGE


     P     MATRIX        2 ROWS        2 COLUMNS
 2.0000000D+01    1.0000000D+01
 1.0000000D+01    1.0000000D+01

     F     MATRIX        1 ROWS        2 COLUMNS
 1.0000000D+01    1.0000000D+01

*** SOLUTION OF LQG Q-RICCATI ***


PROGRAM TO SOLVE CONTINUOUS STEADY-STATE RICCATI EQUATION BY THE NEWTON ALGORITHM


     A     MATRIX        2 ROWS        2 COLUMNS
 1.0000000D+00    0.0000000D+00
 1.0000000D+00    1.0000000D+00

     B     MATRIX        2 ROWS        1 COLUMNS
 1.0000000D+00
 0.0000000D+00

     Q     MATRIX        2 ROWS        2 COLUMNS
 6.0000000D+01    6.0000000D+01
```

```
  6.0000000D+01   6.0000000D+01

H IS AN IDENTITY MATRIX


     R    MATRIX        1 ROWS       1 COLUMNS
  1.0000000D+00

INITIAL F MATRIX


     F    MATRIX        1 ROWS       2 COLUMNS
  4.0020000D+00   8.0080020D+00


FINAL VALUES OF P AND F AFTER    7 ITERATIONS TO CONVERGE


     P    MATRIX        2 ROWS       2 COLUMNS
  1.0000000D+01   1.0000000D+01
  1.0000000D+01   2.0000000D+01

     F    MATRIX        1 ROWS       2 COLUMNS
  1.0000000D+01   1.0000000D+01

 DIF. OF PQ-LYAPUNOV =  1397.87078471772827
 DIF. OF PQ-LYAPUNOV = 0.568434188608080149E-12

 *** SOLUTION OF ME ALGORITHM ***
 BETA= 0.000000000000000000E+00

 *** MATRIX AC ***
 -9.0000000D+00   1.0000000D+00
 -2.0000000D+01  -9.0000000D+00

 *** MATRIX F ***
  1.0000000D+01
  1.0000000D+01

 *** MATRIX K ***
  1.0000000D+01   1.0000000D+01

 DIF. OF PQ-LYAPUNOV =  1483.52550453469172
 DIF. OF PQ-LYAPUNOV =  31.1122528653733639
 DIF. OF PQ-LYAPUNOV =  4.03515303082116361
 DIF. OF PQ-LYAPUNOV = 0.141727321507062243
 DIF. OF PQ-LYAPUNOV = 0.671832436983663683E-01
 DIF. OF PQ-LYAPUNOV = 0.182016129660951265E-01
 DIF. OF PQ-LYAPUNOV = 0.233668764548156105E-02
 DIF. OF PQ-LYAPUNOV = 0.102304770734917838E-03

 *** SOLUTION OF ME ALGORITHM ***
 BETA= 0.500000007450580597E-01

 *** MATRIX AC ***
 -9.2759643D+00   1.0000000D+00
 -2.2199309D+01  -8.6775519D+00

 *** MATRIX F ***
  1.0275964D+01
```

```
 1.2521757D+01

*** MATRIX K ***
 9.6775519D+00   9.6775519D+00

DIF. OF PQ-LYAPUNOV =  1549.05227113589928
DIF. OF PQ-LYAPUNOV =  84.4184428246941252
DIF. OF PQ-LYAPUNOV =  26.0149127163574008
DIF. OF PQ-LYAPUNOV =  3.20086419084577756
DIF. OF PQ-LYAPUNOV =  2.04724222381747722
DIF. OF PQ-LYAPUNOV =  1.86151733248436813
DIF. OF PQ-LYAPUNOV = 0.878094194215236712
DIF. OF PQ-LYAPUNOV = 0.263785988925747006
DIF. OF PQ-LYAPUNOV = 0.262883900012980121E-01
DIF. OF PQ-LYAPUNOV = 0.268890374742341010E-01
DIF. OF PQ-LYAPUNOV = 0.214989882915119779E-01
DIF. OF PQ-LYAPUNOV = 0.970914569580827447E-02
DIF. OF PQ-LYAPUNOV = 0.267261225042147998E-02
DIF. OF PQ-LYAPUNOV = 0.239413246333697316E-03

*** SOLUTION OF ME ALGORITHM ***
BETA= 0.999999642372131348E-01

*** MATRIX AC ***
-9.7125436D+00   1.0000000D+00
-2.4992726D+01  -7.3259751D+00

*** MATRIX F ***
 1.0712544D+01
 1.6666751D+01

*** MATRIX K ***
 8.3259751D+00   8.3259751D+00

DIF. OF PQ-LYAPUNOV =  1665.28167831654781
DIF. OF PQ-LYAPUNOV =  159.062705845073140
DIF. OF PQ-LYAPUNOV =  71.1380666167275990
DIF. OF PQ-LYAPUNOV =  13.3069521641417623
DIF. OF PQ-LYAPUNOV =  11.0438398010626315
DIF. OF PQ-LYAPUNOV =  16.0436452531334908
DIF. OF PQ-LYAPUNOV =  13.0792646555584611
DIF. OF PQ-LYAPUNOV =  8.25789695673404367
DIF. OF PQ-LYAPUNOV =  4.14705313050279756
DIF. OF PQ-LYAPUNOV =  1.45431039864143941
DIF. OF PQ-LYAPUNOV = 0.465922398768725543E-01
DIF. OF PQ-LYAPUNOV = 0.485000639653435428
DIF. OF PQ-LYAPUNOV = 0.539609938257910926
DIF. OF PQ-LYAPUNOV = 0.402903454531099214
DIF. OF PQ-LYAPUNOV = 0.236057926695423248
DIF. OF PQ-LYAPUNOV = 0.106004689502810834
DIF. OF PQ-LYAPUNOV = 0.279536444650148042E-01
DIF. OF PQ-LYAPUNOV = 0.915740669563547272E-02
DIF. OF PQ-LYAPUNOV = 0.197266314012836119E-01
DIF. OF PQ-LYAPUNOV = 0.182912521599405409E-01
DIF. OF PQ-LYAPUNOV = 0.120420679774611017E-01
DIF. OF PQ-LYAPUNOV = 0.652838842739811298E-02
DIF. OF PQ-LYAPUNOV = 0.257274780449279206E-02
DIF. OF PQ-LYAPUNOV = 0.162227934140446450E-03

*** SOLUTION OF ME ALGORITHM ***
```

```
BETA= 0.149999976158142090

*** MATRIX AC ***
-1.0182880D+01    1.0000000D+00
-2.7716769D+01   -5.3712423D+00

*** MATRIX F ***
 1.1182880D+01
 2.1345526D+01

*** MATRIX K ***
 6.3712423D+00    6.3712423D+00

DIF. OF PQ-LYAPUNOV =  2043.32093212088944
DIF. OF PQ-LYAPUNOV =  422.622199510942664
DIF. OF PQ-LYAPUNOV =  321.264125429695071
DIF. OF PQ-LYAPUNOV =  192.278331629577451
DIF. OF PQ-LYAPUNOV =  79.8465890746290938
DIF. OF PQ-LYAPUNOV = 0.861929903368036321
DIF. OF PQ-LYAPUNOV =  45.4660100056273109
DIF. OF PQ-LYAPUNOV =  67.0896696260947465
DIF. OF PQ-LYAPUNOV =  72.6089619424420221
DIF. OF PQ-LYAPUNOV =  68.7181502289284936
DIF. OF PQ-LYAPUNOV =  59.9148116939483657
DIF. OF PQ-LYAPUNOV =  49.0218397391997769
DIF. OF PQ-LYAPUNOV =  37.7723155764265357
DIF. OF PQ-LYAPUNOV =  27.2175524990368558
DIF. OF PQ-LYAPUNOV =  17.9715890001505159
DIF. OF PQ-LYAPUNOV =  10.3474085776692846
DIF. OF PQ-LYAPUNOV =  4.43796081232255801
DIF. OF PQ-LYAPUNOV = 0.174540652494613369
DIF. OF PQ-LYAPUNOV =  2.62192679792053696
DIF. OF PQ-LYAPUNOV =  4.19708560874767045
DIF. OF PQ-LYAPUNOV =  4.82445353761380602
DIF. OF PQ-LYAPUNOV =  4.77252454138550775
DIF. OF PQ-LYAPUNOV =  4.28303268764602763
DIF. OF PQ-LYAPUNOV =  3.55564212377225886
DIF. OF PQ-LYAPUNOV =  2.74423996702182649
DIF. OF PQ-LYAPUNOV =  1.95656871106899644
DIF. OF PQ-LYAPUNOV =  1.26023106524792183
DIF. OF PQ-LYAPUNOV = 0.689740011255651098
DIF. OF PQ-LYAPUNOV = 0.255423312301900296
DIF. OF PQ-LYAPUNOV = 0.497768301354426512E-01
DIF. OF PQ-LYAPUNOV = 0.242425045327479438
DIF. OF PQ-LYAPUNOV = 0.343552133679565941
DIF. OF PQ-LYAPUNOV = 0.376151789676043791
DIF. OF PQ-LYAPUNOV = 0.361426173297275000
DIF. OF PQ-LYAPUNOV = 0.317537609715657254
DIF. OF PQ-LYAPUNOV = 0.258647684084621687
DIF. OF PQ-LYAPUNOV = 0.196205113078065096
DIF. OF PQ-LYAPUNOV = 0.137271073638146390
DIF. OF PQ-LYAPUNOV = 0.858135003701931964E-01
DIF. OF PQ-LYAPUNOV = 0.444365240942943274E-01
DIF. OF PQ-LYAPUNOV = 0.137696488600909106E-01
DIF. OF PQ-LYAPUNOV = 0.759230053478177069E-02
DIF. OF PQ-LYAPUNOV = 0.207745545176294399E-01
DIF. OF PQ-LYAPUNOV = 0.272035746581309468E-01
DIF. OF PQ-LYAPUNOV = 0.286712760499199248E-01
DIF. OF PQ-LYAPUNOV = 0.269569517134300440E-01
DIF. OF PQ-LYAPUNOV = 0.232341396629749397E-01
```

```
DIF. OF PQ-LYAPUNOV = 0.184511397725941606E-01
DIF. OF PQ-LYAPUNOV = 0.139755047704852586E-01
DIF. OF PQ-LYAPUNOV = 0.972890090514511030E-02
DIF. OF PQ-LYAPUNOV = 0.593533052074235457E-02
DIF. OF PQ-LYAPUNOV = 0.274063213629460734E-02
DIF. OF PQ-LYAPUNOV = 0.648672616364365240E-03

*** SOLUTION OF ME ALGORITHM ***
BETA= 0.199999988079071045

*** MATRIX AC ***
-1.0741235D+01   1.0000000D+00
-3.1813464D+01  -3.6263966D+00

*** MATRIX F ***
 1.1741235D+01
 2.7187067D+01

*** MATRIX K ***
 4.6263966D+00   4.6263966D+00
```

APPENDIX 2

Program for M.E. Method

```
C MAIN PROGRAM FOR THE MAXIMUM ENTROPY METHOD
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(10),B(10),C(10),R(10),R1(10),R2(10),V(10),
     &          V2(10),B1(10),V1(10),DUMMY(100),FP(10),IOP(3),
     &          AT(10),CT(10),FQ(10),H(10),P(10),Q(10),PB(10),
     &          QB(10),AS(10),BS(10),V2S(10),CS(10),BST(10),
     &          CST(10),AST(10),CQ(10),COF(10),COP(10),COP1(10),
     &          QS(10),AQS(10),COQ(10),COQ1(10),APS(10),AC1(10),
     &          AC(10),F(10),AK(10),UI(10),R2S(10),PS(10),
     &          AP(10),AQ(10)
      DIMENSION NA(2),NB(2),NC(2),NR(2),NR2(2),NV2(2),NB1(2),
     &          NV(2),NR1(2),NV1(2),NCT(2),NFP(2),NFQ(2),NH(2),
     &          NP(2),NQ(2),NAS(2),NV2S(2),NCS(2),NBST(2),
     &          NCST(2),NAST(2),NPS(2),NCOP(2),NAP(2),NAQ(2),
     &          NQS(2),NAQS(2),NCOF(2),NCQ(2),NAPS(2),NAC1(2),
     &          NAC(2),NF(2),NK(2)
      LOGICAL IDENT,DISC,FNULL,SYM
      DATA STOL/1.E-4/, ETOL/1.E-3/,EPSA/1.E-4/,EPSB/1.E-4/
      CALL RDTITL
C INPUT THE MATRICES FOR THE SYSTEM
      CALL READ(5,A,NA,B,NB,C,NC,R,NR,R2,NR2)
      CALL READ(3,V,NV,V2,NV2,B1,NB1)
      THETA=60.
      AMU=60.
      CALL SCALE(R,NR,R1,NR1,THETA)
      CALL SCALE(V,NV,V1,NV1,AMU)
C     WRITE(*,*) ' MATRIX R1'
C     CALL PRNT(R1,NR1,0,3)
C     WRITE(*,*) ' MATRIX V1'
C     CALL PRNT(V1,NV1,0,3)
C COMPUTE THE F MATRICES FOR P & Q - RICCATI EQUATION
      IOP(1)=0
      IOP(2)=1
      IOP(3)=0
      SCLE=1.
      CALL CSTAB(A,NA,B,NB,FP,NFP,IOP,SCLE,DUMMY)
      CALL TRANP(A,NA,AT,NA)
      CALL TRANP(C,NC,CT,NCT)
      CALL CSTAB(AT,NA,CT,NCT,FQ,NFQ,IOP,SCLE,DUMMY)
      WRITE(*,*) ' *** MATRIX F FOR P-RICCATI ***'
      CALL PRNT(FP,NFP,0,3)
      WRITE(*,*) ' *** MATRIX F FOR Q-RICCATI ***'
      CALL PRNT(FQ,NFQ,0,3)
C  SOLVE FOR INITIAL P & Q FROM LQG SOLUTION
      IOP(1)=1
      IOP(2)=0
      IOP(3)=0
      IDENT=.TRUE.
      DISC=.FALSE.
      FNULL=.FALSE.
      WRITE(*,*) ' *** SOLUTION OF LQG P-RICCATI ***'
      CALL RICNWT(A,NA,B,NB,H,NH,R1,NR1,R2,NR2,FP,NFP,P,NP,IOP,
     &            IDENT,DISC,FNULL,DUMMY)
      WRITE(*,*) ' *** SOLUTION OF LQG Q-RICCATI ***'
      CALL RICNWT(AT,NA,CT,NCT,H,NH,V1,NV1,V2,NV2,FQ,NFQ,Q,NQ,IOP,
     &            IDENT,DISC,FNULL,DUMMY)
C PREPARE THE REQUIRED MATRICES FOR ME ITERATIONS
      CALL NULL(PB,NA)
      CALL NULL(QB,NA)
      CALL EQUATE(A,NA,AS,NAS)
```

```
      CALL EQUATE(B,NB,BS,NBS)
      CALL EQUATE(V2,NV2,V2S,NV2S)
      CALL EQUATE(C,NC,CS,NCS)
      CALL TRANP(BS,NBS,BST,NBST)
      CALL TRANP(CS,NCS,CST,NCST)
      CALL TRANP(AS,NAS,AST,NAST)
      CALL UNITY(UI,NA)
      S=-1.
      DO 300 IK=1,5
         BETA=.05*(IK-1)
         B1(2)=BETA
C BEGIN ITERATIONS
      PQTEMP=0.
5     K=1
      PTNORM=0.
      QTNORM=0.
      PLTNOR=0.
      QLTNOR=0.
C COMPUTE COEFFICIENTS FOR P-RICCATI
      I=1
10    CALL SUB12(R2,NR2,B1,NB1,P,NP,PB,NA,R2S,NR2)
C SOLVE P-RICCATI
      IOP(1)=0
      IOP(2)=0
      IOP(3)=0
      IDENT=.TRUE.
      DISC=.FALSE.
      FNULL=.FALSE.
C     WRITE(*,*) ' *** SOLUTION OF P-RICCATI ***'
      CALL RICNWT(AS,NA,BS,NBS,H,NH,R1,NR1,R2S,NR2,FP,NFP,P,NP,IOP,
     &             IDENT,DISC,FNULL,DUMMY)
C TEST FOR CONVERGENCE OF P - RICCATI SOLUTION
      IOPT=2
      M1=NP(1)
      CALL NORMS(M1,M1,M1,P,IOPT,PNORM)
      DIF=DABS(PNORM-PTNORM)
C     WRITE(*,*) ' DIF. OF P-RICCATI = ', DIF
      IF(DIF.LE.STOL) THEN
        GO TO 20
      ELSE
        PTNORM=PNORM
        I=I+1
        IF(I.GE.500) GO TO 200
        GO TO 10
      END IF
C COMPUTES COEFFICIENT FOR Q - RICCATI EQUATION
20    J=1
25    CALL SUB12(R2,NR2,B1,NB1,P,NP,PB,NA,R2S,NR2)
      CALL MULT(BST,NBST,P,NP,PS,NPS)
      CALL SUB13(B1,NB1,R2S,NR2,PS,NPS,QB,NA,CQ,NCQ)
      CALL ADD(V1,NV1,CQ,NCQ,COF,NCOF)
C SOLVE FOR Q-RICCATI
C     WRITE(*,*) ' *** SOLUTION OF Q-RICCATI EQ.'
      CALL RICNWT(AST,NAST,CST,NCST,H,NH,COF,NCOF,V2S,NV2S,FQ,NFQ,
     &     Q,NQ,IOP,IDENT,DISC,FNULL,DUMMY)
C TEST FOR CONVERGENCE OF Q-RICCATI
      N1=NQ(1)
      CALL NORMS(N1,N1,N1,Q,IOPT,QNORM)
      DIF=DABS(QNORM-QTNORM)
C     WRITE(*,*) ' DIF. OF Q-RICCATI = ', DIF
```

```
      IF(DIF.LE.STOL)THEN
        GO TO 30
      ELSE
        QTNORM=QNORM
        J=J+1
        IF(J.GE.500) GO TO 200
        GO TO 25
      END IF
C COMPUTE COEFFICIENTS FOR P-LYAPUNOV EQUATION
30    I1=1
35    CALL SUB12(R2,NR2,B1,NB1,P,NP,PB,NA,R2S,NR2)
      ITYPE=1
      CALL SUB5(ITYPE,UI,NA,P,NP,BS,NBS,R2S,NR2,COP,NCOP)
C     WRITE(*,*) ' *** MATRIX C OF P-LYAPUNOV ***'
C     CALL PRNT(COP,NCOP,0,3)
      CALL SCALE(COP,NCOP,COP1,NCOP,S)
      CALL MULT(Q,NQ,CST,NCST,QS,NQS)
      CALL SUB8(AS,NAS,QS,NQS,V2S,NV2S,CS,NCS,AQS,NAQS)
C SOLVE P-LYAPUNOV EQUATION
      IOPL=0
      SYM=.TRUE.
C     WRITE(*,*) ' *** SOLUTION P-LYAPUNOV EQ. ***'
      CALL BARSTW(AQS,NAQS,AQ,NAQ,COP1,NCOP,IOPL,SYM,EPSA,EPSB,DUMMY)
      CALL EQUATE(COP1,NCOP,PB,NA)
C TEST FOR CONVERGENCE OF P-LYAPUNOV
      CALL NORMS(M1,M1,M1,PB,IOPT,PLNORM)
      DIF=DABS(PLTNOR-PLNORM)
C     WRITE(*,*) ' DIF. OF P-LYAPUNOV =',DIF
      IF(DIF.LE.STOL) THEN
        GO TO 40
      ELSE
        PLTNOR=PLNORM
        IF(I1.GE.500) GO TO 200
        GO TO 35
      END IF
C COMPUTE COEFFICIENTS FOR Q-LYAPUNOV EQUATION
C40   J1=1
C45   ITYPE=2
40    ITYPE=2
      CALL SUB5(ITYPE,UI,NA,Q,NQ,CS,NCS,V2S,NV2S,COQ,NCOQ)
C     WRITE(*,*) ' *** MATRIX C OF Q-LYAPUNOV ***'
C     CALL PRNT(COQ,NCOQ,0,3)
      CALL SCALE(COQ,NCOQ,COQ1,NCOQ,S)
      CALL SUB12(R2,NR2,B1,NB1,P,NP,PB,NA,R2S,NR2)
      CALL MULT(BST,NBST,P,NP,PS,NPS)
      CALL SUB8(AS,NAS,BS,NBS,R2S,NR2,PS,NPS,APS,NAPS)
C SOLVE Q-LYAPUNOV EQUATION
C     WRITE(*,*) ' *** SOLUTION OF Q-LYAPUNOV ***'
      CALL BARSTW(APS,NAPS,AP,NAP,COQ1,NCOQ,IOPL,SYM,EPSA,EPSB,DUMMY)
      CALL EQUATE(COQ1,NCOQ,QB,NA)
C TEST FOR CONVERGENCE OF Q-LYAPUNOV
      CALL NORMS(M1,M1,M1,QB,IOPT,QLNORM)
C     DIF=DABS(QLNORM-QLTNOR)
C     WRITE(*,*) ' DIF. OF Q-LYAPUNOV =',DIF
C     IF(DIF.LE.STOL) THEN
C       GO TO 50
C     ELSE
C       QLTNOR=QLNORM
C       J1=J1+1
C       IF(J1.GE.500) GO TO 200
```

```
C       GO TO 45
C     END IF
C TEST FOR CONVERGENCE OF ME SOLUTION
50    PQNORM=PLNORM+QLNORM
      DIF=DABS(PQTEMP-PQNORM)
      WRITE(*,*) ' DIF. OF PQ-LYAPUNOV =',DIF
      IF(DIF.LE.ETOL) THEN
         GO TO 60
      ELSE
         PQTEMP=PQNORM
         IF(K.GE.50) GO TO 200
         GO TO 10
      END IF
C COMPUTE COMPENSATER MATRICES
C COMPUTE AC
60    CALL SUB8(AS,NAS,QS,NQS,V2S,NV2S,CS,NCS,AC1,NAC1)
      CALL SUB12(R2,NR2,B1,NB1,P,NP,PB,NA,R2S,NR2)
      CALL SUB8(AC1,NAC1,BS,NBS,R2S,NR2,PS,NPS,AC,NAC)
      WRITE(*,*) '  '
      WRITE(*,*) ' *** SOLUTION OF ME ALGORITHM ***'
      WRITE(*,*) ' BETA=', BETA
      WRITE(*,*) ' *** MATRIX AC ***'
      CALL PRNT(AC,NAC,0,3)
C COMPUTE F
      ITYPE=2
      CALL SUB9(ITYPE,Q,NQ,CS,NCS,V2S,NV2S,F,NF)
      WRITE(*,*) ' *** MATRIX F ***'
      CALL PRNT(F,NF,0,3)
C COMPUTE K
      ITYPE=1
      CALL SUB9(ITYPE,R2S,NR2,BS,NBS,P,NP,AK,NK)
      WRITE(*,*) ' *** MATRIX K ***'
      CALL PRNT(AK,NK,0,3)
300   CONTINUE
200   STOP
      END
C ****** SUBROUTINE SUB1
      SUBROUTINE SUB1(B,NB,C,NC,D,ND,A,NA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(50),B(50),C(50),D(50),BC(50)
      DIMENSION NA(2),NB(2),NC(2),ND(2),NBC(2)
      CALL MULT(B,NB,C,NC,BC,NBC)
      CALL MULT(BC,NBC,D,ND,A,NA)
      RETURN
      END
C ****** SUBROUTINE SUB5
      SUBROUTINE SUB5(ITYPE,B,NB,C,NC,D,ND,E,NE,A,NA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(50),B(50),C(50),D(50),E(50),
     &          DT(50),F(50),FT(50),EI(50),BT(50)
      DIMENSION NA(2),NB(2),NC(2),ND(2),NE(2),NBT(2),
     &          NDT(2),NF(2),NFT(2)
      CALL TRANP(B,NB,BT,NBT)
      IF(ITYPE.EQ.1) CALL SUB1(BT,NBT,C,NC,D,ND,F,NF)
      IF(ITYPE.EQ.2) CALL SUB1(D,ND,C,NC,B,NB,F,NF)
      CALL TRANP(F,NF,FT,NFT)
      CALL UNITY(EI,NE)
      N=NE(1)
      NR=NE(2)
      CALL GAUSEL(N,N,E,NR,EI,IERR)
```

```
      IF(ITYPE.EQ.1) CALL SUB1(F,NF,EI,NE,FT,NFT,A,NA)
      IF(ITYPE.EQ.2) CALL SUB1(FT,NFT,EI,NE,F,NF,A,NA)
      RETURN
      END
C ***** SUBROUTINE SUB8
      SUBROUTINE SUB8(B,NB,C,NC,D,ND,E,NE,A,NA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION B(50),C(50),D(50),E(50),A(50),F(50)
      DIMENSION NB(2),NC(2),ND(2),NE(2),NA(2),NF(2)
      CALL UNITY(DI,ND)
      N=ND(1)
      NR=ND(2)
      CALL GAUSEL(N,N,D,NR,DI,IERR)
      CALL SUB1(C,NC,DI,ND,E,NE,F,NF)
      CALL SUBT(B,NB,F,NF,A,NA)
      RETURN
      END
C ****** SUBROUTINE SUB9
      SUBROUTINE SUB9(ITYPE,B,NB,C,NC,D,ND,A,NA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(50),B(50),C(50),D(50),BI(50),CI(50),DI(50),CT(50)
      DIMENSION NA(2),NB(2),NC(2),ND(2),NCT(2)
      IF(ITYPE.EQ.1) THEN
         CALL UNITY(BI,NB)
         N=NB(1)
         NR=NB(2)
         CALL GAUSEL(N,N,B,NR,BI,IERR)
      ELSE
         CALL UNITY(DI,ND)
         N=ND(1)
         NR=ND(2)
         CALL GAUSEL(N,N,D,NR,DI,IERR)
      END IF
      CALL TRANP(C,NC,CT,NCT)
      IF(ITYPE.EQ.1) CALL SUB1(BI,NB,CT,NCT,D,ND,A,NA)
      IF(ITYPE.EQ.2) CALL SUB1(B,NB,CT,NCT,DI,ND,A,NA)
      RETURN
      END
C ***** SUBROUTINE SUB12
      SUBROUTINE SUB12(A,NA,B,NB,C,NC,D,ND,E,NE)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(50),B(50),C(50),D(50),E(50),BT(50),CD(50),TEMP(50)
      DIMENSION NA(2),NB(2),NC(2),ND(2),NE(2),NBT(2),NCD(2)
      CALL TRANP(B,NB,BT,NBT)
      CALL ADD(C,NC,D,ND,CD,NCD)
      CALL SUB1(BT,NBT,CD,NCD,B,NB,TEMP,NA)
      CALL ADD(A,NA,TEMP,NA,E,NE)
      RETURN
      END
C**** SUBROUTINE SUB13
      SUBROUTINE SUB13(A,NA,B,NB,C,NC,D,ND,E,NE)
      DIMENSION A(50),B(50),C(50),D(50),E(50),BI(50),TEMP(50),TT(50)
      DIMENSION NA(2),NB(2),NC(2),ND(2),NE(2),NT(2),NTT(2)
      CALL UNITY(BI,NB)
      N=NB(1)
      NR=NB(2)
      CALL GAUSEL(N,N,B,NR,BI,IERR)
      CALL SUB1(A,NA,BI,NB,C,NC,TEMP,NT)
      CALL TRANP(TEMP,NT,TT,NTT)
      CALL SUB1(TEMP,NT,D,ND,TT,NTT,E,NE)
```

```
      RETURN
      END
```

APPENDIX 3

Program for Optimal Projection

```
C MAIN PROGRAM FOR THE OPTIMAL PROJECTION METHOD
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(49),B(14),C(21),R1(49),R2(4),V1(49),
     &          V2(9),P(49),Q(49),UI(49),TAUO(49),C1(49),
     &          IOP(3),F(49),C3(49),CT(21),C5(49),C6(49),
     &          C12(21),AQC(49),AQ(49),AQT(49),BX(49),
     &          C8(49),C9(49),C13(14),AP(49),APC(49),ER(50),
     &          EI(57),V(49),TAU(49),C11(49),C14(49),GA(49),
     &          G(49),GT(49),AC(49),FC(49),RKC(49),H(49),
     &          FP(49),FQ(49),DUMMY(500),AT(49),APT(49),
     &          WK(98),QP(49),CK(14),CF(21),CFC(49),R2N(49),
     &          BCK(49),ACF(49),CA(49),AP1(49),AQ1(49),VI(49)
      DIMENSION NA(2),NB(2),NC(2),NR2(2),NV2(2),NF(2),NCT(2),
     &          NBX(2),NC13(2),NGA(2),NG(2),NGT(2),NAC(2),
     &          NRKC(2),NH(2),NFP(2),NFQ(2),NR1(2),NV1(2),
     &          NP(2),NQ(2),NCK(2),NCF(2),NAP1(2),NAQ1(2),NC12(2)
      LOGICAL IDENT,DISC,FNULL,SYM
      DATA STOL/1.E-4/, ETOL/1.E-3/,EPSA/1.E-6/,EPSB/1.E-6/
      CALL RDTITL
C     WRITE(*,*) ' INPUT THE ORDER TO BE REDUCED '
C     READ(*,*) NCR
      NCR=4
C INPUT THE MATRICES FOR THE SYSTEM
      CALL READ(5,A,NA,B,NB,C,NC,R1,NR1,R2,NR2)
      CALL READ(2,V1,NV1,V2,NV2)
C     R2(2)=1.
C     R2(3)=2.
      WRITE(6,*) ' *** NORMAL R2'
      CALL NORMAL(R2,NR2,R2N,NR2)
      CALL PRNT(R2N,NR2,0,3)

C COMPUTE THE F MATRICES FOR P & Q - RICCATI EQUATION
      IOP(1)=0
      IOP(2)=1
      IOP(3)=0
      SCLE=1
      CALL CSTAB(A,NA,B,NB,FP,NFP,IOP,SCLE,DUMMY)
      WRITE(6,*) ' MATRIX F'
      CALL TRANP(A,NA,AT,NA)
      CALL TRANP(C,NC,CT,NCT)
      CALL CSTAB(AT,NA,CT,NCT,FQ,NFQ,IOP,SCLE,DUMMY)
C  SOLVE FOR INITIAL P & Q FROM LQG SOLUTION
      IOP(1)=0
      IOP(2)=0
      IOP(3)=0
      IDENT=.TRUE.
      DISC=.FALSE.
      FNULL=.FALSE.
      WRITE(6,*) ' RICCATI'
      CALL RICNWT(A,NA,B,NB,H,NH,R1,NR1,R2,NR2,FP,NFP,P,NP,IOP,
     &            IDENT,DISC,FNULL,DUMMY)
      WRITE(6,*) ' Q RICCATI'
      CALL RICNWT(AT,NA,CT,NCT,H,NH,V1,NV1,V2,NV2,FQ,NFQ,Q,NQ,IOP,
     &            IDENT,DISC,FNULL,DUMMY)
C COMPUTE THE COMPENSATOR MATRICES FOR LQG
      CALL SUB9(1,R2,NR2,B,NB,P,NP,CK,NCK)
      CALL SUB9(2,Q,NQ,C,NC,V2,NV2,CF,NCF)
      CALL MULT(CF,NCF,C,NC,CFC,NA)
      CALL MULT(B,NB,CK,NCK,BCK,NA)
      CALL SUBT(A,NA,CFC,NA,ACF,NA)
```

```
      CALL SUBT(ACF,NA,BCK,NA,CA,NA)
      WRITE(6,*) ' K MATRIX FOR COMPENSATOR'
      CALL PRNT(CK,NCK,0,3)
      WRITE(6,*) ' F MATRIX FOR COMPENSATOR'
      CALL PRNT(CF,NCF,0,3)
      WRITE(6,*) ' AC MATRIX FOR COMPENSATOR'
      CALL PRNT(CA,NA,0,3)
C COMPUTES MATRIX NORM P & Q SOLUTIONS
      M1=NA(1)
      N1=NA(2)
      IOPT=2
      WRITE(6,*) ' NOW A'
C     CALL NORMS(M1,M1,N1,P,IOPT,PNORM)
      WRITE(6,*) ' NOW B'
C     CALL NORMS(M1,M1,N1,Q,IOPT,QNORM)
      CALL UNITY(UI,NA)
      CALL NULL(TAUO,NA)
C BEGIN ITERATIONS FOR OPTIMAL PROJECTION ALGORITHM
      K=1
5     I=1
      PNORM=0.
C COMPUTES COEFFICIENT FOR P - RICCATI EQUATION
10    ITYPE=1
      WRITE(6,*) ' NOW C'
      CALL SUB5(ITYPE,TAUO,NA,P,NA,B,NB,R2,NR2,C1,NA)
      WRITE(6,*) ' NOW D'
      CALL ADD(R1,NR1,C1,NA,C1,NA)
      WRITE(*,*) ' NOW E'
C SOLVES FOR P - RICCATI EQUATION
      IOP(1)=0
      IOP(2)=0
      IOP(3)=0
      IDENT=.TRUE.
      DISC=.FALSE.
      FNULL=.FALSE.
      CALL RICNWT(A,NA,B,NB,H,NH,C1,NA,R2,NR2,FP,NFP,P,NP,IOP,
     &             IDENT,DISC,FNULL,DUMMY)
      WRITE(*,*) ' PASS P-RICCATI'
C TEST FOR CONVERGENCE OF P - RICCATI SOLUTION
      IOPT=2
      CALL NORMS(M1,M1,N1,P,IOPT,PTNORM)
      DIF=DABS(PNORM-PTNORM)
      WRITE(*,*) ' DIF=',DIF
      IF(DIF.LE.STOL) THEN
        GO TO 20
      ELSE
        PNORM=PTNORM
        I=I+1
        IF(I.GE.1000) GO TO 200
        GO TO 10
      END IF
20    J=1
      QNORM=0.
C COMPUTES COEFFICIENT FOR Q - RICCATI EQUATION
      WRITE(*,*) ' NOW ONE'
30    ITYPE=2
      CALL SUB5(ITYPE,TAUO,NA,Q,NA,C,NC,V2,NV2,C3,NA)
      CALL ADD(V1,NA,C3,NA,C3,NA)
C SOLVES FOR Q - RICCATI EQUATION
      WRITE(*,*) ' NOW Q'
```

```
      CALL RICNWT(AT,NA,CT,NCT,H,NH,C3,NA,V2,NV2,FQ,NFQ,Q,NQ,IOP,
     &     IDENT,DISC,FNULL,DUMMY)
C TEST FOR CONVERGENCE OF Q - RICCATI SOLUTION
      WRITE(*,*) ' NORMS'
      CALL NORMS(M1,M1,N1,Q,IOPT,QTNORM)
      DIF=DABS(QNORM-QTNORM)
      WRITE(*,*) ' DIFQ=',DIF
      IF(DIF.LE.STOL) THEN
         GO TO 40
      ELSE
         QNORM=QTNORM
         J=J+1
         IF(J.GE.1000) GO TO 200
      WRITE(*,*) ' GO TO 30'
         GO TO 30
      END IF
C COMPUTE COEFFICIENTS FOR P-LYAPUNOV EQUATION
40    ITYPE=1
      WRITE(*,*) ' NOW TWO'
      CALL SUB5(ITYPE,UI,NA,P,NA,B,NB,R2,NR2,C5,NA)
      WRITE(*,*) ' NOW 3'
      CALL SUB5(ITYPE,TAUO,NA,P,NA,B,NB,R2,NR2,C6,NA)
      WRITE(*,*) ' NOW 4'
      CALL SUBT(C6,NA,C5,NA,C6,NA)
      ITYPE=2
      CALL SUB9(ITYPE,Q,NA,C,NC,V2,NV2,C12,NC12)
      WRITE(*,*) ' NOW5'
      CALL MULT(C12,NC12,C,NC,AQC,NA)
      WRITE(*,*) ' NOW6'
      CALL SUBT(A,NA,AQC,NA,AQ,NA)
      WRITE(*,*) ' AQ BARSTW - P'
      CALL PRNT(AQ,NA,0,3)
C SOLVE FOR P - LYAPUNOV EQUATION
      IOPL=1
      SYM=.TRUE.
      CALL TRANP(AQ,NA,AQT,NA)
      WRITE(*,*) ' NOW7'
      CALL BARSTW(AQT,NA,AQ1,NAQ1,C6,NA,IOPL,SYM,EPSA,EPSB,DUMMY)
C COMPUTE COEFFICIENTS FOR Q - RICCATI EQUATION
      ITYPE=2
      WRITE(*,*) 'Q1'
      CALL SUB5(ITYPE,UI,NA,Q,NA,C,NC,V2,NV2,C8,NA)
      WRITE(*,*) ' Q2'
      CALL SUB5(ITYPE,TAUO,NA,Q,NA,C,NC,V2,NV2,C9,NA)
      WRITE(*,*) ' Q3'
      CALL SUBT(C9,NA,C8,NA,C9,NA)
      ITYPE=1
      CALL SUB9(ITYPE,R2,NR2,B,NB,P,NA,C13,NC13)
      CALL MULT(B,NB,C13,NC13,APC,NA)
      WRITE(*,*) ' Q4'
      CALL SUBT(A,NA,APC,NA,AP,NA)
      WRITE(*,*) ' AP BARSTW - Q'
      CALL PRNT(AP,NA,0,3)
C SOLVES FOR Q - LYAPUNOV EQUATION
      WRITE(*,*) ' WRITE'
      CALL TRANP(AP,NA,APT,NA)
      CALL BARSTW(AP,NA,AP1,NAP1,C9,NA,IOPL,SYM,EPSA,EPSB,DUMMY)
C TEST FOR CONVERGENCE OF P & Q - LYAPUNOV SOLUTIONS
      CALL MULT(C9,NA,C6,NA,QP,NA)
      WRITE(*,*) ' *** MATRIX QP ***'
```

```
      CALL PRNT(QP,NA,0,3)
C COMPUTE EIGENVALUES AND EIGENVECTORS OF MATRIX QP
      N=NA(1)
      ISV=N
      ILV=0
      CALL EIGEN(N,N,QP,ER,EI,ISV,ILV,V,WK,IERR)
      WRITE(*,*) ' ISV =',ISV
      WRITE(*,*) ' ILV =',ILV
      WRITE(*,*) ' IERR =',IERR
C CHECK IF EIGENVALUES ARE ARRANGED IN INCREASING OR DECREASIG ORDER
      CALL LNCNT(4)
      PRINT 650
650   FORMAT(//,' EIGENVALUES OF QP',//)
675   FORMAT(10X,2D16.8)
      CALL LNCNT(N)
      DO 700 I1=1,N
      PRINT 675,ER(I1),EI(I1)
700   CONTINUE
      WRITE(*,*) ' EIGENVECTOR OF QP WITH NAMDA INCREASING ORDER'
      CALL PRNT(V,NA,0,3)
      N=NA(1)
      NU=N-NCR
      ND=NU+1
      RA=ER(NU)/ER(ND)
      RATIO=DABS(RA)
      WRITE(*,*) ' RATIO=',RATIO
      IF(RATIO.LT.ETOL)THEN
         GO TO 50
      ELSE
         K=K+1
         IF(K.GE.500) GO TO 200
C FORM NEW TAU
C        CALL UNITY(VI,NA)
C        N=NA(1)
C        NR=NA(2)
C        CALL GAUSEL(N,N,V,NR,VI,IERR)
C        CALL FOMTAU(V,NA,NCR,TAU,NA)
         CALL CONTAU(NCR,VI,NA,TAU,NA)
         CALL SUBT(UI,NA,TAU,NA,TAUO,NA)
      WRITE(*,*) ' TAU'
      CALL PRNT(TAU,NA,0,3)
      WRITE(*,*) ' TAUO'
      CALL PRNT(TAUO,NA,0,3)
      WRITE(*,*) ' GO TO 5'
         GO TO 5
      END IF
50    CALL SUBT(AQ,NA,APC,NA,C11,NA)
      CALL SUB1(C12,NC,D,ND,C13,NC13,C14,NA)
      CALL ADD(C11,NA,C14,NA,C14,NA)
C FORM GAMMA AND G
C
C
C
C
C     CALL SUB1(GA,NGA,C14,NA,GT,NG,AC,NAC)
C PRINT AC
C     CALL MULT(GA,NGA,C12,NC,FC,NFC)
C PRINT FC
C     CALL MULT(C13,NC13,GT,NG,RKC,NRKC)
C PRINT KC
```

```
 200   STOP
       END
C ****** SUBROUTINE SUB1
       SUBROUTINE SUB1(B,NB,C,NC,D,ND,A,NA)
       IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION A(*),B(*),C(*),D(*),BC(49)
       DIMENSION NA(2),NB(2),NC(2),ND(2),NBC(2)
       CALL MULT(B,NB,C,NC,BC,NBC)
       CALL MULT(BC,NBC,D,ND,A,NA)
       RETURN
       END
C ****** SUBROUTINE SUB5
       SUBROUTINE SUB5(ITYPE,B,NB,C,NC,D,ND,E,NE,A,NA)
       IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION A(50),B(*),C(*),D(*),E(*),
     &           DT(50),F(50),FT(50),EI(50),BT(50)
       DIMENSION NA(2),NB(2),NC(2),ND(2),NE(2),NBT(2),
     &           NDT(2),NF(2),NFT(2)
          CALL TRANP(B,NB,BT,NBT)
       IF(ITYPE.EQ.1) CALL SUB1(BT,NBT,C,NC,D,ND,F,NF)
       IF(ITYPE.EQ.2) CALL SUB1(D,ND,C,NC,BT,NBT,F,NF)
       CALL TRANP(F,NF,FT,NFT)
       CALL UNITY(EI,NE)
       N=NE(1)
       NR=NE(2)
       CALL GAUSEL(N,N,E,NR,EI,IERR)
       IF(ITYPE.EQ.1) CALL SUB1(F,NF,EI,NE,FT,NFT,A,NA)
       IF(ITYPE.EQ.2) CALL SUB1(FT,NFT,EI,NE,F,NF,A,NA)
       RETURN
       END
C ****** SOUROUTINE SUB9
       SUBROUTINE SUB9(ITYPE,B,NB,C,NC,D,ND,A,NA)
       IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION A(50),B(50),C(50),D(50),BI(50),CI(50),DI(50),CT(50)
       DIMENSION NA(2),NB(2),NC(2),ND(2),NCT(2)
       IF(ITYPE.EQ.1) THEN
          CALL UNITY(BI,NB)
          N=NB(1)
          NR=NB(2)
          CALL GAUSEL(N,N,B,NR,BI,IERR)
       ELSE
          CALL UNITY(DI,ND)
          N=ND(1)
          NR=ND(2)
          CALL GAUSEL(N,N,D,NR,DI,IERR)
       END IF
       CALL TRANP(C,NC,CT,NCT)
       IF(ITYPE.EQ.1) CALL SUB1(BI,NB,CT,NCT,D,ND,A,NA)
       IF(ITYPE.EQ.2) CALL SUB1(B,NB,CT,NCT,DI,ND,A,NA)
       RETURN
       END
C ***** SUBROUTINE FOMTAU
       SUBROUTINE FOMTAU(V,NV,NCR,TAU,NA)
       IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION V(50),TAU(50),VI(50),SUM(50),VKT(50),UK(50),UV(50)
       DIMENSION NV(2),NA(2),NVKT(2),NUK(2)
       CALL UNITY(VI,NV)
       N=NV(1)
       NR=NV(2)
       CALL GAUSEL(N,N,V,NR,VI,IERR)
```

```
      CALL NULL(SUM,NV)
      DO 10 K=1,NCR
        CALL UKVKT(K,V,NV,VI,VKT,NVKT,UK,NUK)
        CALL MULT(UK,NUK,VKT,NVKT,UV,NV)
        CALL ADD(SUM,NV,UV,NV,SUM,NV)
 10   CONTINUE
      CALL EQUATE(SUM,NV,TAU,NA)
      RETURN
      END
C ****** SUBROUTINE UKVKT
      SUBROUTINE UKVKT(K,V,NV,VI,VKT,NVKT,UK,NUK)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION V(50),VI(50),VKT(50),UK(50)
      DIMENSION NV(2),NVKT(2),NUK(2)
      N=NV(1)
      L=1+(K-1)*N
      DO 10 I=1,N
        JV=K+(I-1)*N
        VKT(I)=V(JV)
        JU=L+(I-1)
        UK(I)=VI(JU)
 10   CONTINUE
      NVKT(1)=1
      NVKT(2)=N
      NUK(1)=N
      NUK(2)=1
      RETURN
      END
C ****** SUBROUTINE CONTAU
      SUBROUTINE CONTAU(NCR,PI,NA,TAU,NTAU)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION PI(49),TAU(49),PSI(49),EI(49),NA(2),NTAU(2),PN(49)
C CONSTRUCT PSI FROM PI
      CALL PSICON(PI,NA,PSI,NA)
      WRITE(*,*) ' EIGENVECTOR OF QP WITH NAMDA DECREASING ORDER'
      CALL PRNT(PSI,NA,0,3)
C CONSTRUCT MATRIX (INC,0)
      CALL NORMAL(PSI,NA,PN,NA)
      WRITE(*,*) ' NORMALIZED EIGENVECTOR'
      CALL PRNT(PN,NA,0,3)
      CALL NULL(EI,NA)
      N=NA(1)
      N1=N+1
      DO 10 I=1,NCR
         K=1+(I-1)*N1
      EI(K)=1
 10   CONTINUE
      WRITE(*,*) ' MATRIX (INC, 0)'
      CALL PRNT(EI,NA,0,3)
C COMPUTES TAU
      ITYPE=2
      CALL SUB9(ITYPE,PN,NA,EI,NA,PN,NA,TAU,NA)
      RETURN
      END
C ****** SUBROUTINE PSICON
      SUBROUTINE PSICON(PI,NA,PSI,NPSI)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION PI(49),PSI(49),NA(2),NPSI(2)
      N=NA(1)
      L=1
```

```
      DO 10 I=1,N
      DO 20 J=1,N
      K=N*(N-I)+J
      PSI(L)=PI(K)
      L=L+1
 20   CONTINUE
 10   CONTINUE
      RETURN
      END
C **** SUBROUTINE NORMAL
      SUBROUTINE NORMAL(A,NA,B,NB)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(49),B(49),C(7),NA(2),NB(2)
C COMPUTES EUCLIDIAN NORM OF EACH COLUMN
      N=NA(1)
      K=0
      DO 10 I=1,N
        SUM=0.
        DO 20 J=1,N
         J1=J+K
         TEMP=A(J1)*A(J1)
         SUM=SUM+TEMP
20    CONTINUE
         K=K+N
         C(I)=DSQRT(SUM)
10    CONTINUE
C NORMALIZE EACH COLUMN
      K=0
      DO 30 I=1,N
      DO 40 J=1,N
         J1=J+K
         B(J1)=A(J1)/C(I)
40    CONTINUE
         K=K+N
30    CONTINUE
      RETURN
      END
```