Imperial College

Department of
Computing and
Control

Implemantation and
Evaluation of an algorithm
for the minimal realisation
of transfer function mat -
rices in Jordan form.

MSc. Sep. '76

IMPERIAL COLLEGE
OF SCIENCE AND TECHNOLOGY
UNIVERSITY of LONDON

Department of Computing and Control
Control Systems section

Implementation and evaluation of an algorithm for the minimal realisation of transfer function matrices in Jordan form.

by
A. POULIEZOS

Report submitted in part fulfilment of the requirements for the M. Sc. degree in Engineering of the University of London, and the Diploma of Imperial College of Science and Techonology.

September 1976.

CONTENTS

Abstract

This paper presents, implements and evaluates a computational algorithm for minimally realising a given proper transfer function matrix in the Jordan canonical form, using e-value and e-vector technics. The observable realisation of the transfer function matrix is calculated first and its Jordan form is obtained, which is then reduced to a minimal realisation, by a method which utilises the special form of the Jordan realisation. It also compares this algorithm against Rosenbrock's.[8].

Introduction

The minimal realisation of a proper rational transfer function matrix G(s) into a state space form is one of the basic problems in linear system theory. It is a simpler problem of a class of general realisation problems as for example the calculation of a set of differential equations of some special form from which G(s) can arise, or the calculation of an RLC network that realises G(s) etc.

Many authors have tackled this problem in different ways such as the ones mentioned in [1] - [8]. Certain algorithms have also been programmed such as in [9] and [9a].

This paper hopes to present an algorithm which by utilising the Jordan canonical form of a matrix has only to search for independence through fewer rows than in algorithm not using the Jordan form as for example Rosenbrock's.[8] Further the denominator of G(s) does not have to be in factored form as in [7], but the irreducible realisation is not in general achieved in one step.

However it has much the same structure as the algorithms suggested in [5] and [7], but differ in the calculation of the Jordan form and the reduction prodedure.

Particular emphasis is given in comparing this algorithm with Rosenbrock's, which is currently in use at the Control Department od Imperial College and has not produced very satisfactory results in certain cases.

CHAPTER I

Some Algebraic Preliminaries

1.1 E-values and E-vectors

Let A be a matrix in C^{nxn}. Then a scalar $\lambda \epsilon C$ is called an e-value of A, if there exists a nonzero vector $\underline{x}$ in C^n, such that $A\underline{x} = \lambda \underline{x}$. Any nonzero vector $\underline{x}$ satisfying this equation is called an e-vector of A associated with the e-value λ.
The vector space which contains all the $\underline{x}$ which satisfy the equation $(A-\lambda I)\underline{x} = \underline{0}$ is defined as the eigenspace (or e-space) of $(A-\lambda I)$ (or kernel of $(A-\lambda I)$).
If A has distinct e-values then a complete linearly independent set of e-vectors can be found. If the e-values are not distinct a linearly independent set cannot be guaranteed but it is always possible to find a linearly independent set which would contain generalised e-vectors as well as proper ones.
A generalised e-vector of order k associated with an e-value λ of a matrix A satisfies the equations :

$$(A-\lambda I)^k\underline{x} = \underline{0}$$

$$(A-\lambda I)^{k-1}\underline{x} \neq \underline{0}$$

1.2 The Jordan Canonical form of a matrix

For any matrix A in C^{nxn} there exists a transformation matrix Q in C^{nxn}, such that

$$J = Q^{-1}AQ$$

and J is in the form :

$$J = \left[\begin{array}{ccc|ccc|c|ccc}
J_{m_1}(\lambda_1) & & & & & & & & & \\
& J_{m_2}(\lambda_1) & & & & & & & & \\
& & \ddots & & & & & & & \\
& & J_{m_s}(\lambda_1) & & & & & & & \\
\hline
& & & J_{n_1}(\lambda_2) & & & & & & \\
& & & & \ddots & & & & & \\
& & & & & J_{n_t}(\lambda_2) & & & & \\
\hline
& & & & & & \ddots & & & \\
\hline
& & & & & & & J_{t_1}(\lambda_q) & & \\
& & & & & & & & \ddots & \\
& & & & & & & & & J_{t_q}(\lambda_q)
\end{array}\right]$$

where each $J_k(\lambda)$ is a kxk Jordan block

$$J_k(\lambda) = \begin{bmatrix} \lambda & 1 & 0 & \dots & 0 & 0 \\ 0 & \lambda & 1 & \dots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \dots & \lambda & 1 \\ 0 & 0 & 0 & \dots & 0 & \lambda \end{bmatrix}$$

We say that J is the Jordan canonical form of the matrix A. The distinct e-values of A are $\lambda_1, \lambda_2, \ldots \lambda_q$ and the multiplicity of λ_1 is $m_1+m_2+\cdots+m_s$ and so on. The numbers $m_1, m_2, \ldots, t_q$ are determined by the number of linearly independent e-vectors per e-value and the dimension of the e-space.

The set $\left[(m_1, m_2, \ldots, m_s), (n_1, \ldots, n_t), \ldots, (t_1, \ldots, t_q)\right]$ is called the Segre characteristic of A and is enough to determine the Jordan canonical form of a matrix.

The Jordan form of the matrix is unique up to the ordering of the Jordan blocks.

1.3 Theorem 1

Let $\underline{u}$ and $\underline{v}$ be two generalised e-vectors of rank k and l respectively, associated with the same e-value λ of a matrix A. Define $\underline{u}^i = (A-\lambda I)^{k-i}\underline{u}$ for i= 1, 2, ..., k and $\underline{v}^j = (A-\lambda I)^{l-j}\underline{v}$ for j = 1, 2, ..., l. If the two vectors $\underline{u}^1$ and $\underline{v}^1$ are linearly independent, then the generalised e-vectors $\underline{u}^1, \underline{u}^2, \ldots, \underline{u}^k, \underline{v}^1, \underline{v}^2, \ldots, \underline{v}^l$ are linearly independent.

Proof : We first prove that the sets $(\underline{u}^1, \underline{u}^2, \ldots, \underline{u}^k)$, $(\underline{v}^1, \underline{v}^2, \ldots, \underline{v}^l)$ are composed of linearly independent vectors only. We have that

$$\begin{aligned} \underline{u}^k &= \underline{u} \\ \underline{u}^{k-1} &= (A-\lambda I)\underline{u} = (A-\lambda I)\underline{u}^k \\ \underline{u}^{k-2} &= (A-\lambda I)^2\underline{u} = (A-\lambda I)\underline{u}^{k-1} \\ &\vdots \\ \underline{u}^1 &= (A-\lambda I)^{k-1}\underline{u} = (A-\lambda I)\underline{u}^2 \end{aligned}$$

so that all the $\underline{u}^i$'s are generalised e-vectors of rank i, since

$$(A-\lambda I)^i\underline{u}^i = (A-\lambda I)^i (A-\lambda I)^{k-i}\underline{u} = (A-\lambda I)^k\underline{u} = \underline{0}$$

$$\text{and } (A-\lambda I)^{i-1}\underline{u}^i = (A-\lambda I)^{i-1} (A-\lambda I)^{k-i}\underline{u} = (A-\lambda I)^{k-1}\underline{u} \neq \underline{0}$$

Similarly for the $\underline{v}^j$'s.

Now assume that the $\underline{u}^i$'s are linearly dependent. Then there exist $\beta_1, \beta_2, \ldots, \beta_k$ not all zero, such that

$$\beta_1\underline{u}^1 + \beta_2\underline{u}^2 + \ldots + \beta_k\underline{u}^k = \underline{0}$$

$$\therefore \quad (A-\lambda I)^{k-j}(\beta_1\underline{u}^1 + \ldots + \beta_k\underline{u}^k) = \underline{0} \quad \text{for } j=1,\ldots,k$$

$$\therefore \quad \beta_1(A-\lambda I)^{k-j}\underline{u}^1+\beta_2(A-\lambda I)^{k-j}\underline{u}^2+\ldots+\beta_k(A-\lambda I)^{k-j}\underline{u}^k=\underline{0}$$

or $$\beta_1(A-\lambda I)^{2k-(1+j)}\underline{u}+\beta_2(A-\lambda I)^{2k-(2+j)}\underline{u}+\ldots+\beta_k(A-\lambda I)^{2k-(k+j)}\underline{u}=\underline{0}$$

Now for $j = 1$, $2k-(i+1)\geq k$ for $i\leq k-1$

Therefore we are left with ,

$$\beta_k(A-\lambda I)^{k-1}\underline{u}^k = \underline{0}$$

but $(A-\lambda I)^{k-1}\underline{u}^k \neq \underline{0}$, hence $\beta_k = 0$

For $j = 2$, $2k-(i+2)\geq k$ for $i\leq k-2$

Since $\beta_k=0$, we are left with

$$\beta_{k-1}(A-\lambda I)^{k-2}\underline{u}^{k-1} = \underline{0} \text{ , hence } \beta_{k-1} = \underline{0} .$$

Similarly we prove the same for the rest of the β_j's. Hence

$$\beta_1 = \beta_2 = \ldots = \beta_k = 0$$

and therefore the set $(\underline{u}^1,\underline{u}^2, \ldots ,\underline{u}^k)$ is a linearly independent set. Similarly for the set $(\underline{v}^1,\underline{v}^2, \ldots,\underline{v}^l)$.

We next show that the two sets are linearly independent (i.e. that the elements of the two sets taken together form a linearly independent set)by contradiction.

For suppose that there exists a dependence relation,

$$\sum_{i=1}^{k} \beta_i\underline{u}^i = \sum_{j=1}^{l} \beta'_j\underline{v}^j$$, where the β_i's and β'_j's are not all zero. Then

$$\sum_{i=1}^{k} \beta_i(A-\lambda I)\underline{u}^i = \sum_{j=1}^{l} \beta'_j(A-\lambda I)\underline{v}^j$$

Hence $$\sum_{i=1}^{k-1} \beta_{i+1}\underline{u}^i = \sum_{j=1}^{l-1} \beta'_{j+1}\underline{v}^j \text{ , since } (A-\lambda I)\underline{u}^1=(A-\lambda I)\underline{v}^1=\underline{0} .$$

Suppose $l>k$ and do the above operation k times. We get

$$\underline{0} = \sum_{j=1}^{l-k} \beta'_{j+k}\underline{v}^j \quad \therefore \quad \beta'_j = 0 \text{ for } j = k+1,\ldots,l$$

One step before

$$\beta_k \underline{u}^1 = \sum_{j=1}^{l-k+1} \beta'_{j+k-1} \underline{v}^j = \beta'_k \underline{v}^1$$

But $\underline{u}^1$, $\underline{v}^1$ are linearly independent by assumption. Therefore

$$\beta_k = \beta'_k = 0$$

Continuing backwards in this way we prove

$$\beta_1 = \beta_2 = \dots = \beta_k = \beta'_1 = \beta'_2 = \dots = \beta'_l = 0$$

This contradicts the initial assumption and thus proves the theorem.

1.4 Procedure for calculating the Jordan form representation of a matrix A.

We now give a procedure for calculating the transformation matrix Q, such that

$$J = Q^{-1}AQ$$

is the Jordan canonical form for A.

Step 1 Compute the e-values of A. Let $\lambda_1, \lambda_2, \dots, \lambda_m$ be the distinct e-values with multiplicities $n_1, n_2, \dots, n_m$ respectively .

Step 2 Compute n_1 linearly independent generalised e-vectors of A corresponding to λ_1 as follows: Compute $(A-\lambda_1 I)^i$ for i= 1, 2, ... until the rank of $(A-\lambda_1 I)^k$ is equal to the rank of $(A-\lambda_1 I)^{k+1}$. Find a generalised e-vector of rank k, say $\underline{u}$. Define $\underline{u}^i = (A-\lambda_1 I)^{k-i}\underline{u}$, for i=1,2,...,k. If $k = n_1$ proceed to step 3. If $k < n_1$, find another linearly independent generalised e-vector with the largest possible rank. Continue in this way until n_1 linearly independent generalised e-vectors are found. Note that if rank $(A-\lambda_1 I) = a$, then there are totally (n_1-a) chains of generalised e-vectors associated with λ_1.

Step 3 Repeat from step 2 for $\lambda_2, \lambda_3, \ldots, \lambda_m$.

Bearing in mind the previous theories, it is easy to show that this procedure yields the required transofmation matrix Q, which has as its columns the chains of generalised e-vectors associated with each e-value, i.e.

$$Q = \left[\underline{u}_1^1 \ldots \underline{u}_1^{k_1} \vdots \underline{v}_1^1 \ldots \underline{v}_1^{l_1} \vdots \ldots \vdots \underline{u}_2^1 \ldots \underline{u}_2^{k_2} \vdots \underline{v}_2^1 \ldots \underline{v}_2^{l_2} \vdots \ldots \vdots \underline{u}_m^1 \ldots \underline{u}_m^{k_m} \vdots \underline{v}_m^1 \ldots \underline{v}_m^{l_m} \vdots \right.$$

CHAPTER II

The Algorithm

2.1 Introductory Concepts

Let G(s) be a transfer function matrix of dimensions mxl in the following form :

$$G(s) = \begin{bmatrix} \frac{n_{11}(s)}{d_1(s)} & \frac{n_{12}(s)}{d_1(s)} & \cdots & \frac{n_{1l}(s)}{d_1(s)} \\ \frac{n_{21}(s)}{d_2(s)} & \frac{n_{22}(s)}{d_2(s)} & \cdots & \frac{n_{2l}(s)}{d_2(s)} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{n_{m1}(s)}{d_m(s)} & \frac{n_{m2}(s)}{d_m(s)} & \cdots & \frac{n_{ml}(s)}{d_m(s)} \end{bmatrix}$$

where the n_{ij}'s and d_i's are polynomials in s given by:

$$\left.\begin{aligned} n_{ij}(s) &= p_q^{ij}s^q + p_{q-1}^{ij}s^{q-1} + \cdots + p_o^{ij} \\ d_i(s) &= s^t + r_{t-1}^i s^{t-1} + \cdots + r_o^i \end{aligned}\right\} \quad \begin{aligned} i &= 1, \ldots, m \\ j &= 1, \ldots, l \end{aligned}$$

Then an observable realisation of G(s) is given by:

$$\bar{S} : \dot{x} = \bar{A}x + \bar{B}u$$
$$y = \bar{C}x$$

where the matrices $\bar{A}$, $\bar{B}$, $\bar{C}$ are given by :

$$\bar{A} = \begin{bmatrix} \bar{A}_1 & \cdot & \cdot & \cdot & 0 \\ 0 & \bar{A}_2 & \cdot & \cdot & 0 \\ \vdots & & & \ddots & \\ 0 & \cdot & \cdot & \cdot & \bar{A}_m \end{bmatrix}$$

and $$\bar{A}_i = \begin{bmatrix} 0 & 0 & \ldots & -r_o^i \\ 1 & 0 & \ldots & -r_1^i \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \ldots & 1 & -r_{t-1}^i \end{bmatrix} \qquad i = 1, \ldots, m$$

$$\bar{B} = \begin{bmatrix} \bar{B}_1 \\ \bar{B}_2 \\ \vdots \\ \bar{B}_m \end{bmatrix} \quad \text{and} \quad \bar{B}_i = \begin{bmatrix} p_o^{i1} & p_o^{i2} & \ldots & p_o^{il} \\ \cdots & \cdots & \cdots & \cdots \\ p_q^{i1} & p_q^{i2} & \ldots & p_q^{il} \end{bmatrix}$$

$$\bar{C} = \begin{bmatrix} \bar{C}_1 & \bar{C}_2 & \ldots & \bar{C}_m \end{bmatrix} \quad \text{and} \quad \bar{C}_i = \left[\begin{array}{c|c} & 0 \\ & \vdots \\ 0 & 1 \\ & \vdots \\ & 0 \end{array}\right] \text{ in ith position}$$

This realisation can now be transformed into a Jordan canonical form, using the method described in 1.3. Let this Jordan canonical observable realisation be given by :

$$S : \dot{x} = Ax + Bu$$
$$y = Cx$$

where
$$A = Q\bar{A}Q^{-1}$$
$$B = Q^{-1}\bar{B}$$
$$C = \bar{C}Q \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Let the distinct e-values of $\bar{A}$ be $\lambda_1, \lambda_2, \ldots, \lambda_s$. Then the matrices A, B, C have the form as shown in fig. 1.
(Note that if some of the λ_i's are complex these matrices will have complex elements).

2.1.2 Conditions for controllability and observability based on the Jordan canonical form.

For the system S of 2.1.1 the conditions for controllability are provided by the following theorem :

Theorem 2.1.3 The representation S is controllable if and only if for each i = 1,2, ..., s the set of f(i) l-dimensional row vectors

$$\underline{b}^{i1}_{k_{i1}\cdot}\ ,\ \underline{b}^{i2}_{k_{i2}\cdot},\ \ldots,\ \underline{b}^{if(i)}_{k_{if(i)}\cdot}\quad ,$$

is a linearly independent set.

Proof: The fact is used that S is controllable if and only if rows of $e^{At}B$ are linearly independent on $[0,\infty)$. [11]
Since A is in the Jordan canonical form, $e^{At}B$ may be written explicitly as,

$$e^{At}B = \begin{bmatrix} e^{A_{11}t}B_{11} \\ e^{A_{12}t}B_{12} \\ \vdots \\ e^{A_{sf(s)}t}B_{sf(s)} \end{bmatrix} \qquad \ldots\ldots\ldots\ldots\ldots\ldots(2)$$

since,

$$\underset{n\times n}{A} = \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_s \end{bmatrix} \qquad \underset{n\times l}{B} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_s \end{bmatrix}$$

$$\underset{m\times n}{C} = \begin{bmatrix} C_1 & C_2 & \cdots & C_s \end{bmatrix}$$

....

$$\underset{k_i\times k_i}{A_i} = \begin{bmatrix} A_{i1} & & & \\ & A_{i2} & & \\ & & \ddots & \\ & & & A_{if(i)} \end{bmatrix} \qquad \underset{k_i\times l}{B_i} = \begin{bmatrix} B_{i1} \\ B_{i2} \\ \vdots \\ B_{if(i)} \end{bmatrix}$$

$$\underset{m\times k_i}{C_i} = \begin{bmatrix} C_{i1} & C_{i2} & \cdots & C_{if(i)} \end{bmatrix}$$

....

$$\underset{k_{ij}\times k_{ij}}{A_{ij}} = \begin{bmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & \cdots & 0 \\ \vdots & & \ddots & & \\ 0 & & & & \lambda_i \end{bmatrix} \qquad \underset{k_{ij}\times l}{B_{ij}} = \begin{bmatrix} \underline{b}^{ij}_{1.} \\ \underline{b}^{ij}_{2.} \\ \vdots \\ \underline{b}^{ij}_{k_{ij}.} \end{bmatrix}$$

$$\underset{m\times k_{ij}}{C_{ij}} = \begin{bmatrix} \underline{c}^{ij}_{.1} & \underline{c}^{ij}_{.2} & \cdots & \underline{c}^{ij}_{.k_{ij}} \end{bmatrix}$$

<u>FIGURE 1</u>

$$e^{At} = \begin{bmatrix} e^{A_{11}t} & & & \\ & e^{A_{12}t} & & \\ & & \ddots & \\ & & & e^{A_{sf(s)}t} \end{bmatrix}$$

Necessity :Observe that the last row of $e^{A_{ij}t}B_{ij}$ is $\underline{b}^{ij}_{k_{ij}\cdot}\, e^{\lambda_i t}$. Hence if $\underline{b}^{i1}_{k_{i1}\cdot}$, $\underline{b}^{i2}_{k_{i2}\cdot}$, ... , $\underline{b}^{if(i)}_{k_{if(i)}\cdot}$ are linearly independent then the last row of $e^{A_{ij}t}B_{ij}$, $j = 1, 2, \ldots, f(i)$ is linearly independent.

Sufficiency : Having in mind that,

$$e^{A_{ij}t}B_{ij} = e^{\lambda_i t} \begin{bmatrix} 1 & t & \frac{1}{2}t^2 & \cdots & \frac{1}{(k_{ij}-1)!}t^{k_{ij}-1} \\ 0 & 1 & t & \cdots & \frac{1}{(k_{ij}-2)!}t^{k_{ij}-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \underline{b}^{ij}_{1\cdot} \\ \underline{b}^{ij}_{2\cdot} \\ \cdots \\ \underline{b}^{ij}_{k_{ij}\cdot} \end{bmatrix} \quad [12]$$

it is clear that if $\underline{b}^{ij}_{k_{ij}\cdot} \neq \underline{0}$, then all the k_{ij} rows in $e^{A_{ij}t}B_{ij}$ are linearly independent. By hypothesis $\underline{b}^{i1}_{k_{i1}\cdot}$, $\underline{b}^{i2}_{k_{i2}\cdot}$, ... , $\underline{b}^{if(i)}_{k_{if(i)}\cdot}$ is a linearly independent set ; hence all the k_i rows in $e^{A_i t}B_i$ are linearly independent.

Recall that if $\lambda_i \neq \lambda_j$, then the two functions $p_i(t)e^{\lambda_i t}$ and $p_j(t)e^{\lambda_j t}$ [where $p_i(.)$ and $p_j(.)$ are nonzero polynomials] are linearly independent over any nonempty interval. Consequently, any row (or any linear combination of rows)of $e^{A_i t}B_i$ is linearly independent of any row (or any linear combination of rows) of $e^{A_j t}B_j$ with $i \neq j$. Thus if the rows of $e^{At}B$ are linearly dependent, it is because there is a linear dependence relation that applies to rows lying exclusively in Jordan blocks associated with the same e-value. Hence it may be concluded that the hypothesis imply that all k rows are linearly independent over $[0,\infty)$.

Theorem 2.1.4 The represantation S is observable if and only if, for each i = 1, 2, ..., s the set of f(i) m-dimensional column vectors

$$\underline{c}^{i1}_{.1}, \underline{c}^{i2}_{.1}, \ldots, \underline{c}^{if(i)}_{.1}$$

is a linearly independent set.

Proof : This theorem can be proved using duality theorems.

Remarks : The above two theorems provide a nice algorithm for minimally realising a given system in Jordan form, since it is only necessary to realise each subsystem corresponding to each e-value separately.

2.2 The algorithm

Since it is only necessary to look at each subsystem separately, we will drop the extra indices from the matrices A_{ij}, B_{ij}, C_{ij} for ease of notation. Let us therefore look at a subsystem S_i corresponding to e-value λ_i, given by :

$$S_i : \dot{x} = Ax+Bu$$
$$y = Cx$$

where,

$$A = \text{diag}\,[A_1, \ldots, A_r]$$

$$\underset{k_j x k_j}{A_j} = \begin{bmatrix} \lambda_i & 1 & \ldots & 0 \\ 0 & \lambda_i & \ldots & 0 \\ \vdots & & \ddots & \\ & & \lambda_i & 1 \\ 0 & 0 & \ldots & \lambda_i \end{bmatrix}$$

$$B = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_r \end{bmatrix} \qquad C = [C_1 \; C_2 \; \ldots \; C_r]$$

B_j is $k_j x l$ $\qquad$ C_j is $m x k_j$

We will denote by $b^i_{j.}$ the rows of B_i and by $c^i_{.j}$ the columns of C_i, for $j = 1, \ldots, k_i$ and $i = 1, \ldots, r$.

Consider now two types of operation :

(i) Let $1 \leq i, j \leq r$, $k_j \geq k_i$ and β in $\mathbb{C}$. $(i \neq j)$

Let $B \longrightarrow T^{-1}B$ be the linear operation

$$b^i_{k.} \longrightarrow b^i_{k.} + \beta b^j_{(k_j-k_i+k)} \qquad k = 1, \ldots, k_i$$

i.e. the last k_i rows of B_j multiplied by β are added to B_i.

The matrix T which achieves this transformation has the following form :

$$T = \begin{bmatrix} I_{k_1} & & & & & & \\ & \ddots & & & & & \\ & & I_{k_j} & & & & 0 \\ & & & \ddots & & & \\ & & F & & I_{k_i} & & \\ & & & & & \ddots & \\ 0 & & & & & & I_{k_r} \end{bmatrix}$$

Where I_r is the identity matrix of size r x r, and

$$F = \begin{bmatrix} I_{k_j - k_i} & & & \\ & \beta & & \\ & & \beta & \\ & & & \ddots \\ & & & & \beta \end{bmatrix}$$

Therefore,

$$T = \begin{bmatrix} I_{r_1} & & \\ & F' & \\ & & I_{r_2} \end{bmatrix} \quad \therefore \quad T^{-1} = \begin{bmatrix} I_{r_1} & & \\ & (F')^{-1} & \\ & & I_{r_2} \end{bmatrix}$$

Now,

$$F' = \left[\begin{array}{c:c:c} I_{k_j} & & \\ \hdashline & I_{k_l} & \\ \hdashline F & & I_{k_j} \end{array}\right]$$

in general and hence

$$(F')^{-1} = \left[\begin{array}{c:c:c} I_{k_j} & & \\ \hdashline & I_{k_l} & \\ \hdashline -F & & I_{k_j} \end{array}\right]$$

Therefore T^{-1} has the same elements as T but with the elements of F reversed in sign.

Then $C \longrightarrow CT$ is the operation,

$$c^j_{\cdot}(k_j-k_i+k) \longrightarrow c^j_{\cdot}(k_j-k_i+k) - \beta c^i_{\cdot k} \qquad \text{for } k = 1, \ldots, k_i$$

i.e. C_i multiplied by β is subtracted from the last k_i columns of C_j.

Now A is in block Jordan form and $k_i \geq k_j$ so that

$$T^{-1}AT = A$$

Thus the similarity transformation T on the triple (A, B, C) produces another observable realisation of the system, which is still in the Jordan canonical form.

(ii) Suppose that the last row of B_l is zero for some $1 \leq l \leq r$

i.e. $b^l_{k_l \cdot} = \underline{0}$

Now $G(s) = \sum_{i=1}^{r} C_i(sI-A_i)^{-1}B_i$, since A is in block diagonal form,

$$= \sum_{\substack{i=1\\ i\neq l}}^{r} C_i(sI-A_i)^{-1}B_i + \sum_{j=1}^{k_l}\sum_{k=j}^{k_l} c^l_{.j}b^l_{k.}\frac{1}{(s-\lambda)^{k-j+1}} \quad \ldots\ldots\ldots(3)$$

since,

$$C_l(sI-A_l)B_l = \begin{bmatrix} c^l_{.1} & \ldots & c^l_{.k_l}\end{bmatrix} \begin{bmatrix} \frac{1}{s-\lambda} & \frac{1}{(s-\lambda)^2} & \ldots & \frac{1}{(s-\lambda)^{k_l}} \\ 0 & \frac{1}{s-\lambda} & \ldots & \frac{1}{(s-\lambda)^{k_l-1}} \\ \cdot & \cdot & \cdot\quad\cdot\quad\cdot & \cdot \\ 0 & \cdot & \cdot & \frac{1}{s-\lambda}\end{bmatrix} \begin{bmatrix} b^l_{1.} \\ b^l_{2.} \\ \cdot \\ b^l_{k_l.}\end{bmatrix}$$

(A) If $k_l > 1$ then from (3),

$$G(s) = \sum_{\substack{i=1\\ i\neq l}}^{r} C_i(sI-A_i)^{-1}B_i + C'_l(sI-A'_l)^{-1}B'_l$$

where A'_l is a $(k_l-1) \times (k_l-1)$ Jordan block which is obtained by deleting the last row and column from A_l and B'_l, C'_l are obtained by deleting the last row and column from B_l, C_l respectively. Thus the triple (A', B', C') constitutes a realisation in Jordan form of order one less than of the original system (A, B, C), where,

$$A' = \begin{bmatrix} A_1 & & & & & & \\ & \ddots & & & & & \\ & & A_{l-1} & & & & \\ & & & A'_l & & & \\ & & & & A_{l+1} & & \\ & & & & & \ddots & \\ & & & & & & A_r \end{bmatrix}$$

$$C' = \begin{bmatrix} C_1 & \ldots & C_{l-1} & C'_l & C_{l+1} & \ldots & C_r \end{bmatrix}$$

$$B' = \begin{bmatrix} B_1 \\ \vdots \\ B_{l-1} \\ B'_l \\ B_{l+1} \\ \vdots \\ B_r \end{bmatrix}$$

Since observability is determined by the first columns of the blocks of the output matrix (Theorem 2.1.4) i.e. $c^1_{.1}$, $c^2_{.1}$, ..., $c^r_{.1}$, it is obvious that (A', B', C')is an observable realisation , since the last column of block l is deleted.

(B) $k_l = 1$

In this case we can realise G(s) by the triple (A', B', C'), where ,

$$A' = \text{diag}\left[A_1, \ldots, A_{l-1}, A_{l+1}, \ldots, A_r\right]$$

$$B' = \begin{bmatrix} B_1 \\ \vdots \\ B_{l-1} \\ B_{l+1} \\ \vdots \\ B_r \end{bmatrix} \qquad C' = \left[C_1 \ \ldots \ C_{l-1} \ C_{l+1} \ \ldots \ C_r\right]$$

which is again observable , since any nonempty subset of a set of linearly independent vectors is linearly independent.

Using Theorems 2.1.3 and 2.1.4 and operations of type (i) and (ii) mentioned above we produce an algorithm for reducing the observable realisation (A, B, C) to a minimal realisation, i.e. a controllable and observable realisation. The algorithm consists of searching through the last rows of the blocks of B, removing linear dependence, until only a linearly independent set remains. This ensures controllability of the system. The necessary transformations for this removal of dependance are achieved by the operations of type (i) and (ii).

2.2.1 Computational algorithm

For a triple (A, B, C) as defined in 2.2 , the following steps are performed :

(i) Define variables V(1), ..., V(r) by setting V(i) = 0 for i = 1, ..., r

(ii) If V(i) = 0 for all i and no deletions have taken place in previous loop, STOP. Otherwise if V(i) = 0 for some i, go to (iii), or if not go to (i).

(iii) Find i such that V(i) = 0 and

$$k_i = \max_{1 \leq j \leq r} \left\{ k_j : V(j) = 0 \right\} .$$

(iv) If $b^i_{k_i \cdot} = 0$ (i.e. the last row of B_i) go to (vi).

(v) Set V(i) = 1

Find the first nonzero element in row $b^i_{k_i \cdot}$, say the j th element, denoted by $b^i_{k_i,j}$. With this notation for each h such that V(h) = 0, subtract

$$\frac{b^h_{k_h,j}}{b^i_{k_i,j}} \times b^i_{k_i-k_j+k\cdot} \text{ from } b^h_{k\cdot} \quad \text{for } k = 1, \ldots, k_h$$

and add

$$\frac{b^h_{k_h,j}}{b^i_{k_i,j}} \times c^h_{.k} \quad \text{to} \quad c_{.k_i-k_j+k} \quad \text{for } k = 1, \ldots, k_h$$

(This is an operation of type (i) described in 2.2)

Go to (ii).

(vi) Delete the last row from B_i (i.e. $b^i_{k_i}$), the last column from C_i (i.e. $c^i_{.k_i}$) and the last row and column from A_i.

Set $k_i = k_i - 1$

If $k_i = 0$, set $V(i) = 1$

Go to (ii).

The above algorithm is repeated for each subsystem corresponding to each distinct e-value.

The algorithm stops when the set of last rows of the blocks B_i, $i = 1, \ldots, r$ are linearly independent. This is ensured by the algorithm since it really searches for independence in the last rows using a Gauss reduction process. If arow becomes zero, this means that there is a linear dependence between this and the rest of the set of the last rows, therefore this row is deleted from the realisation.When a row is deleted in this way, it is necessary to repeat the process of searching, since the set of last rows has now changed.

(Observe that in order for the k_i rows to be linearly independent $k_i \leq 1$. Therefore if B^i is a column vector, $k_i = 1$ and $b^i_{k_i}$. $\neq 0$ for all i.

2.2.2 Systems with complex poles.

Since it is impossible to realise a complex number in the real world, it is necessary when dealing with complex poles to perform a similarity transformation, which would remove this difficulty by transforming the given system with complex poles into one with only real poles.

Let us suppose that λ_1 is acomplex pole and $(A, B, C)_1$ the minimally realised subsystem corresponding to it. Let G(s), the original transfer function matrix,be expanded into partial fractions,

$$G(s) = G_1(s) + G_2(s) + \dots + G_s(s)$$

Now if $\lambda_2 = \bar{\lambda}_1$, it is clear that $G_2(s) = \bar{G}_1(s)$ and therefore $(\bar{A}, \bar{B}, \bar{C})$ minimally realises the subsystem corresponding to λ_2.

Consider therefore the triple (A′, B′, C′), where,

$$\underset{2n \times 2n}{A'} = \begin{bmatrix} A & \\ & \bar{A} \end{bmatrix} \quad \underset{2n\times 1}{B'} = \begin{bmatrix} B \\ \bar{B} \end{bmatrix} \quad \underset{m\times 2n}{C'} = \begin{bmatrix} C & \bar{C} \end{bmatrix}$$

which minimally realises that part of the system which corresponds to λ_1 and $\lambda_2 (= \bar{\lambda}_1)$.

If $\underset{2n\times 2n}{Q} = \begin{bmatrix} iI & I \\ -iI & I \end{bmatrix}$ then $Q^{-1} = \begin{bmatrix} -\frac{1}{2}iI & \frac{1}{2}iI \\ \frac{1}{2}I & \frac{1}{2}I \end{bmatrix}$

where $i = \sqrt{-1}$,

then,

$$Q^{-1} A' Q = \begin{bmatrix} Re(A) & Im(A) \\ -Im(A) & Re(A) \end{bmatrix}$$

$$Q^{-1} B' = \begin{bmatrix} Im(B) \\ Re(B) \end{bmatrix} \qquad C'Q = -2 \times \begin{bmatrix} Im(C) & Re(C) \end{bmatrix}$$

where Im(.) is the imaginary part of a complex matrix and Re(.) its real part.

Thus, having found a minimal realisation for that part of the system corresponding to a complex λ, we can simply write down a realisation involving only real numbers, of the part of the system corresponding to both λ and $\bar{\lambda}$.

CHAPTER III

The Program

3.1 General

The theorems and algorithms of the preceding chapters were used to write a program that would calculate the minimal realisation of a given proper transfer function matrix on the computer.
The program was written in FORTRAN IV language and all test runs were done on line at the telex terminals of a CDC 6400 computer with 65000 60-bit words of memory, at Imperial college.

3.2 Sections of the program

The program is divided into two large main sections, one which deals with cases of complex e-values and one for real ones. This was thought necessary since it takes much less time to do operations with real numbers , and also less core. Therefore in these cases considerable savings would be made.
A number of library subroutines were used. These are :
From the NAG library :

CO2AEF for the calculation of the roots of real polynomials. This routine claims accuracy equal to single machine word length. In the tests however it was found that too much time required even for a 10^{-20} accuracy, so it was decided to put the accuracy at 10^{-15}.

FO1AAF for the inversion of a real matrix. Uses Crout's method.

FO3AHF and FO4AKF for the inversion of a complex matrix.

From the SSP library (IBM scientific library) :

MFGR for finding the null space of a matrix

and

CMFGR a modification of the above for complex matrices.

The subroutines MRE,GCOM,COMDEN,CANCEL,PNORM belong to the multivariable design package in use at the Control Dept. of Imperial college and were used to obtain the observable realisation of the transfer function matrix.

3.3 Description of main program and subroutines.

The rest of the subroutines and the main program are :

MINREA main program. Reads data, calls subroutine MRE to calculate initial observable realisation and finds e-values of A. Depending on whether the e-values are real or not, the appropriate routine is called.

subs. RREAL and CCOMPL (for real and complex case resp.) checks for distinct e-values , finds e-vectors (a linearly independent set) and resequences e-values and e-vectors, so that equal and/or conjugate e-values are in consecutive places.

subs. TRANS and CTRANS calculation of Jordan form of triple (A, B, C) and of minimal realisation.

subs. CONVERT and CCONVE deletion of rows and columns with updating of vecessary variables.

subs. SWRIT and CSWRIT printout of minimal realisation and in the case of complex e-values transformation to real triple.

subs. MATRM and CMATRM multiplication of matrices.

subs. RRANK and CRANK calculation of rank

subs. JORDAN and CJORDA calculation of an independent set of e-vectors for a set of e-values

subs. NULL and CNULL calculation of basis vectors of null space and dimension of it.

subs. INDEP and CINDEP determination of suitable e-vectors from the null space of a matrix for generation of generalised e-vectors .

3.4 Calculations

A basic difficulty in all the calculations was the fact that two numbers which should be equal, didn't appear equal because of rounding off and inaccuracy in the various routines. The problem arose therefore when to consider two numbers equal, or alternatively how small should the absolute value of their difference be, so that they should be considered equal.
For this reason, a number, EPS in the program, is input to it, so that two numbers a and b are equal if $|a-b| \leq$ EPS.

Given a transfer function matrix G(s) of order m x n, an observable realisation is obtained using the method of 2.1. The matrix A of this observable realisation is in block companion form, i.e.

$$A = \text{diag}\left[A_1, A_2, \ldots, A_m\right]$$

and

$$A_i = \begin{bmatrix} 0 & 0 & \ldots & 0 & -\beta_o^i \\ 1 & 0 & \ldots & 0 & -\beta_1^i \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \ldots & 1 & -\beta_{q_i}^i \end{bmatrix}$$

so that the e-values of A are the e-values of the A_i's. The e-values of the A_i's are in turn the roots of the polynomials,

$$s^{q_i} + \beta_{q_i}^i s^{q_i - 1} + \ldots + \beta_o^i = 0 \;, \; i = 1, \ldots, m$$

So it was thought better to calculate the roots of these polynomials separately, than to calculate the e-values of A, since routines for the former are in general more accurate, and this method was employed.
Having done this, a complete linearly independent set of e-vectors is calculated using the method described in 1.3.
With the triple now in the form of fig. 1, the algorithm, as described in 2.2.1,is applied and the minimal realisation obtained.

3.5 Input descriptions

The maximum dimensions for the observable realisation are:
A: 20x20, B: 20x10, C: 10x20. The maximum numerator order is 20, the maximum denominator order 20, and the maximum order of G: 10x10.
The input description is as follows (see examples):
After typing thr heading the program asks for the order of acceptable error by typing:

```
 TOLERANCE

  +             +
```

The required tolerance is then typed in below and between the stars in a real number format.
The program then asks:

```
 ORDER OF MATRIX - FORMAT 2I2
?
```

The order of a matrix n x m is typed in in the appropriate format.

```
 ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?
```

The oders are typed in in I2 format and all numbers are typed sequentially starting from the order of the G(1,1) element and continuing to G(1,2), G(1,3)... G(1,m), G(2,1)... G(2,m)... G(n,m).

```
 ORDER OF DENOMINATOR
?
```

The order of denominator in I2 format.

```
 COEFFICIENTS OF NUMERATOR POLYNOMIALS
 +        +        +        +        +        +        +
?
```

The coefficients of the numerator polynomials are typed in, each set of coefficients in one line and in the order described above. The coefficients are typed in ascending order, i.e. first the constant term and so on. Eight coefficients can be typed in a row, between the stars, continuing in the next line for polynomials of order higher than seven.

When all coefficients of numerators have been read in , the program asks:

```
 COEFFICIENTS OF DENOMINATOR
?
```

Same rules as above but leading coefficient must be unity.

Now,

```
   +++ INPUT OF DATA FINISHED +++
```

3.5.1 Output descriptions

The output stars with,

```
 OBSERVABLE REALISATION A B C
```

and lists the three (or four if D≠0) matrices A,B,C,(D). If the size of A is greater than 10x10 which is the maximum number of numbers per line, the printing is continued in the next line.

The e-values of A are then printed out as:

E-VALUES

(real part)	(imaginary part)	(real part)	(imaginary part)
-1.00000000000	0	-1.00000000000	1.00000000000
-1.00000000000	0	-1.00000000000	-1.00000000000

If e-values are real

++ E-VALUES REAL, MATRICES WITH REAL ELEMENTS +++

otherwise

++ E-VALUES COMPLEX, MATRICES WITH COMPLEX ELEMENTS++

and in this case real and imaginary parts of numbers are always printed next to each other.

In both cases the printout continues as:

DISTINCT E-VALUES 2 (say)

At this point the e-values have been resequenced so that equal e-values are next to each other and sets of complex conjugate e-values are next to each other, e.g. if e-values were 1,1-i,2,1,1+i,1-i,1+i the resequenced set would be 1,1,1-i,1-i,1+i,1+i,2.

Then the leading e-vectors for each e-value are printed out and the numbers in the statement

LEADING E-VECTORS FOR E-VALUE 1

correspond to the previous sequencing.

The transformation matrix Q and its inverse Q^{-1} (see 1.3) are then printed together with the transformed system in Jordan form.

The algorithm of 2.2.1 is now starting.

The transformed B matrices are printed after each application of the operations defined in 2.2.1.

The final printout is :

DIMENSION 4 (say)

the dimension of the final minimal realisation.

In the case of complex matrices the printout is :

++ MATRICES WITH REAL ELEMENTS AFTER TRANSFORMATION ++

meaning the transformation of 2.2.2.

The minimal realisation is then printed and the program ends.

3.5.2 Error messages

Error messages are printed by the program in various parts of the computation and the program is discontinued after any error. These are :

NUMERATOR ORDER HIGHER THAN DENOMINATOR

in the case of non-proper transfer function matrices.

STORAGE LIMIT EXCEEDED

if order of observable realisation greater than 20.

ALLCOEFFICIENTS ZERO

OBSERVABLE REALISATION NOT OBTAINED

\+ E-VALUES NOT FOUND +

FAILURE IN FO1AAF

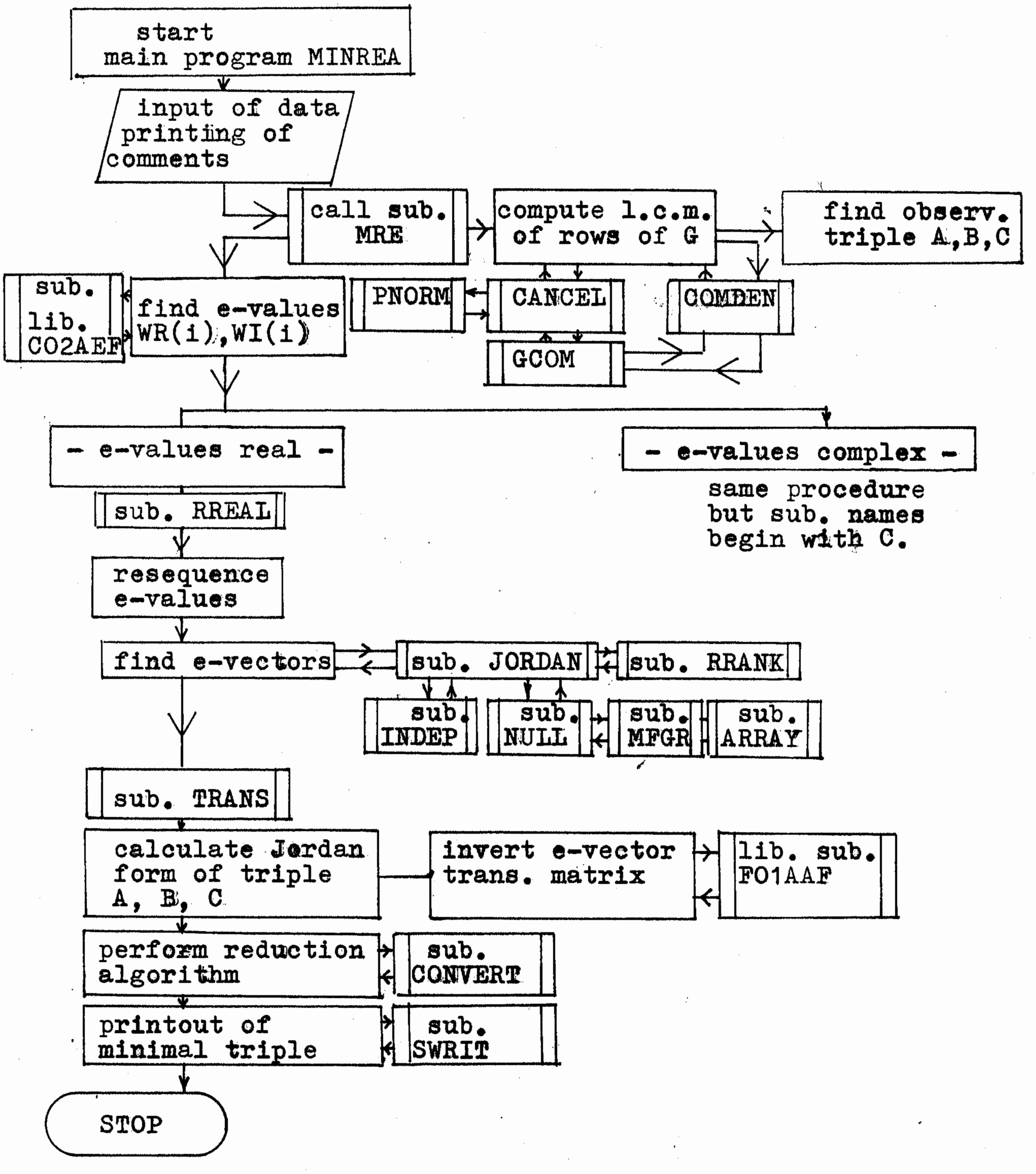
if transformation matrix Q singular.

WRONG CALCULATION OF RANK - PROCEDURE STOPPED

if e-vectors making up Q cannot be found.

All program listings can be found in Appendix I.

3.6 Subroutine Flowchart



CHAPTER IV

TESTS

4.1 Outline of tests

Ten examples were tested with both algorithms (Rosenbrock's and this) and accuracy and speed were compared. For the latter, extra runs had to be done, which printed only the final result.

The examples are given in the following pages, where it is shown :

1. The example and check of the result.
2. The results using Rosenbrock's algorithm.
3. The results using Jordan forms.
4. The results using Jordan forms but without intermediate printouts.

The accuracy shown (eg. TOLERANCE) is the maximum with which correct results were obtained.

TESTS 1-10

TEST 1

$$G(s) = \begin{bmatrix} \dfrac{1}{s+2} & \dfrac{2(2s+5)}{(s+2)(s+3)} \\[2ex] \dfrac{2}{s+3} & \dfrac{4s^2+22s+29}{(s+2)(s+3)^2} \end{bmatrix}$$

CHECK $\quad G(s) = C(sI-A)^{-1}B+D$

$$\begin{bmatrix} 0 & 2.5 & 0. & 0.333 \\ 1.25 & 0.25 & 0.111 & 0 \end{bmatrix} \begin{bmatrix} s+3 & 1 & & \\ & s+3 & & \\ & & s+3 & \\ & & & s+2 \end{bmatrix} \begin{bmatrix} 1.6 & 2.56 \\ 0 & 0.8 \\ 0 & 9. \\ 3 & 6 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & \dfrac{2.5}{s+3} & 0 & \dfrac{.333}{s+2} \\[2ex] \dfrac{1.25}{s+3} & \dfrac{1.25}{(s+3)^2} - \dfrac{.25}{s+3} & \dfrac{.111}{s+2} & 0 \end{bmatrix} \begin{bmatrix} 1.6 & 2.56 \\ 0 & .8 \\ 0 & 9. \\ 3 & 6 \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{3\times.333}{s+2} & \dfrac{2.5\times.8}{s+3} + \dfrac{.333\times6}{s+2} \\[2ex] \dfrac{1.25\times1.6}{s+3} & \dfrac{2.56\times1.25}{s+3} + \dfrac{1.25\times.8}{(s+3)^2} - \dfrac{.25\times.8}{s+3} + \dfrac{9\times.111}{s+2} \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{1}{s+2} & \dfrac{2}{s+3} + \dfrac{2}{s+2} \\[2ex] \dfrac{2}{s+3} & \dfrac{3.2}{s+3} + \dfrac{1}{(s+3)^2} - \dfrac{.2}{s+3} + \dfrac{1}{s+2} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{s+2} & \dfrac{4s+10}{(s+3)(s+2)} \\[2ex] \dfrac{2}{s+3} & \dfrac{3}{s+3} + \dfrac{1}{(s+3)^2} + \dfrac{1}{s+2} \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{1}{s+2} & \dfrac{2(2s+5)}{(s+3)(s+2)} \\[2ex] \dfrac{2}{s+3} & \dfrac{4s^2+22s+29}{(s+3)^2(s+2)} \end{bmatrix}$$

```
ANH,MI=14000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITHM
   *** START INPUT OF DATA ***
TOLERANCE
 *            *
?  .000000005
   ORDER OF MATRIX  -FORMAT 2I2
?  2 2

 ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


  ORDER OF DENOMINATOR
?  3


   COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *        *        *        *        *        *        *        *
?   9.          6.          1.
?   30.         22.         4.
?   12.         10.         2.
?   29.         22.         4.



 COEFFICIENTS OF DENOMINSTOR
?   18.       21.        8.        1.
    *** INPUT OF DATA FINISHED ***
   DIMENSION  4

  A MATRIX
  -1.000  -4.500  -1.759       0
   -.364  -1.182    .441   -.019
   2.069  -8.069  -5.335    .029
  -6.000   6.000   5.455  -2.483
  B MATRIX
   -.000       0
       0       0
  -1.139    .000
  12.000  29.000
  C MATRIX
       0   1.000    .576    .138
    .333   -.333  -.303    .138
TOP

 CP       4.048 SECS.

RUN COMPLETE.
```

```
*** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM

*** START INPUT OF DATA ***
TOLERANCE
 *              *
   .000005
   ORDER OF MATRIX  - FORMAT 2I2
?  2 2


  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


  ORDER OF DENUMINATOR
?  3


   COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *         *         *         *         *         *         *         *
?     9.        6.        1.
?     30.       22.       4.
?     12.       10.       2.
?     29.       22.       4.


 COEFFICIENTS OF DENOMINATOR
?    18.        21.       8.        1.
  *** INPUT OF DATA FINISHED ***

  OBSERVABLE REALISATION A  B  C
  A MATRIX

      0  -6.000        0        0        0

  1.000  -5.000        0        0        0

      0       0        0        0  -18.000

      0       0    1.000        0  -21.000

      0       0        0    1.000   -8.000
  B MATRIX

  3.000  10.000

  1.000   4.000

 12.000  29.000

 10.000  22.000

  2.000   4.000
  C MATRIX

      0   1.000        0        0        0

      0       0        0        0    1.000
E-VALUES
-3.00000000000                0  -2.00000000000          0
-3.00000023842                0  -2.99999976158          0
-2.00000000000                0
```

** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **

DISTINCT E-VALUES 2

LEADING E-VECTORS FOR E-VALUE 1
OF RANK 2

```
     0
     0
 1.000
     0
 -.250
```

LEADING E-VECTORS FOR E-VALUE 1
OF RANK 1

```
 1.000
  .500
     0
     0
     0
```

LEADING E-VECTORS FOR E-VALUE 2
OF RANK 1

```
     0   1.000
     0    .333
 1.000       0
  .667       0
  .111       0
```

TRANSFORMATION MATRIX FOR JORDAN FORM

0	0	1.000	0	1.000
0	0	.500	0	.333
7.500	1.000	0	1.000	0
6.250	0	0	.667	0
1.250	-.250	0	.111	0

INVERSE OF TRANSFORMATION MATRIX

0	0	-.960	2.080	-3.840
0	0	-.800	2.400	-7.200
-2.000	6.000	0	0	0
0	0	9.000	-18.000	36.000
3.000	-6.000	0	0	0

TRANSFORMED B MATRIX

1.500	2.560
.000	.800
-.000	4.000
-.000	9.000
3.000	6.000

TRANSFORMED C MATRIX

0	0	.500	0	.333
1.250	-.250	0	.111	0

TRANSFORMED A MATRIX IN JORDAN FORM

-3.000	1.000	0	.000	0
.000	-3.000	0	.000	0
0	0	-3.000	0	-.000
.000	.000	0	-2.000	0
0	0	.000	0	-2.000

```
TRANSFORMED B
 1.600    2.560
  .000     .800        ................last row 0, delete
 -.000     .000
 -.000    9.000
 3.000    6.000
TRANSFORMED B
 1.600    2.560
  .000     .800
     0    9.000
 3.000    6.000
 3.000    6.000


DIMENSION   4


A MATRIX
 -3.000    1.000  |   .000        0
   .000   -3.000  |   .000        0
   .000     .000  | -2.000   |    0
      0        0  |      0   | -2.000


B MATRIX
  1.600    2.560
   .000     .800
      0    9.000
  3.000    6.000


C MATRIX
      0    2.300  |     0  |  .333
  1.250    -.250  |  .111  | -.000
```

The dimension of the observable realisation has been reduced by 1. The last rows of the blocks corresponding to the same e-values , are easily seen to be independent.

/ 33

*** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM

*** START INPUT OF DATA ***
TOLERANCE

? .000005
ORDER OF MATRIX - FORMAT 2I2
? 2 2

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
? 2 2 2 2

ORDER OF DENOMINATOR
? 3

COEFFICIENTS OF NUMERATOR POLYNOMIALS

?	9.	6.	1.
?	30.	22.	4.
?	12.	10.	2.
?	39.	22.	4.

COEFFICIENTS OF DENOMINATOR

?	18.	21.	8.	1.

*** INPUT OF DATA FINISHED ***

DIMENSION 4

A MATRIX

-3.000	1.000	.000	0
.000	-3.000	.000	0
.000	.000	-2.000	0
0	0	0	-2.000

B MATRIX

1.500	2.560
.000	.500
0	4.000
3.000	5.000

C MATRIX

0	2.500	0	.333
1.250	-.250	.111	-.000

STOP
/STIME
STIME. 21.231 SEC.

TEST 2

$$G(s) = \frac{1}{s^3+2s^2-s-2}\begin{bmatrix} s^2+6 & s^2+s+4 \\ 2s^2-7s-2 & s^2-5s-2 \end{bmatrix}$$

CHECK $G(s) = C(sI-A)^{-1}B$

$$= \begin{bmatrix} .5 & -.5 & .33 \\ 1 & -.5 & -.33 \end{bmatrix}\begin{bmatrix} \frac{1}{s+2} & & \\ & \frac{1}{s+1} & \\ & & \frac{1}{s-1} \end{bmatrix}\begin{bmatrix} 6.6667 & 4 \\ 7 & 4 \\ 3.5 & 3 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{0.5}{s+2} & \frac{-0.5}{s+1} & \frac{.33}{s-1} \\ \frac{1}{s+2} & \frac{-0.5}{s+1} & \frac{-.33}{s-1} \end{bmatrix}\begin{bmatrix} 6.667 & 4 \\ 7 & 4 \\ 3.5 & 3 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{3.33}{s+2} - \frac{3.5}{s+1} + \frac{1.1666}{s-1} & \frac{2(s^2-1)-2(s+2)(s-1)+(s+2)(s+1)}{F(s)} \\ \frac{6.67}{s+2} + \frac{-3.5}{s+1} + \frac{-1.1666}{s-1} & \frac{4(s^2-1)-2(s+2)(s-1)-(s+2)(s+1)}{F(s)} \end{bmatrix}$$

WHERE $F(s) = (s+2)(s-1)(s+1)$

$$= \frac{1}{F(s)}\begin{bmatrix} 3.33(s^2-1)-3.5(s^2+s-2)+1.1666(s^2+3s+2) & s^2+s+4 \\ 6.67(s^2-1)-3.5(s^2+s+2)-1.1666(s^2+3s+2) & s^2-5s-2 \end{bmatrix}$$

$$= \frac{1}{F(s)}\begin{bmatrix} s^2+6 & s^2+s+4 \\ 2s^2-7s-2 & s^2-5s-2 \end{bmatrix}$$

```
ANH,MI=14000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITH
   *** START INPUT OF DATA ***
TOLERANCE
 *            *
  .000000000005
   ORDER OF MATRIX  -FORMAT 2I2
?  2 2

 ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


  ORDER OF DENOMINATOR
?  3


   COEFFICIENTS OF NUMERATOR POLYNOMIALS
   *          *          *          *          *          *          *          *
?     6.           0.         1.
?     4.           1.         1.
?    -2.          -7.         2.
?    -2.          -5.         1.


 COEFFICIENTS OF DENOMINATOR
?    -2.          -1.         2.          1.
    *** INPUT OF DATA FINISHED ***
   DIMENSION   3

  A MATRIX
  -2.000    .571   1.029
   -.000   -.200   -.960
    .000  -1.000    .200
  B MATRIX
       0       0
  -1.400       0
  -7.000  -5.000
  C MATRIX
    .500    .286   -.200
   1.000   -.429   -.200
TOP

 CP        4.176 SECS.

RUN COMPLETE.
```

```
*** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM

*** START INPUT OF DATA ***
TOLERANCE
 *              *
  .000000000005
   ORDER OF MATRIX  - FORMAT 2I2
?  2 2


  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


  ORDER OF DENOMINATOR
?  3


   COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *          *          *          *          *          *          *          *
?    6.          0.          1.
?    4.          1.          1.
?   -2.         -7.          2.
?   -2.         -5.          1.


 COEFFICIENTS OF DENOMINATOR
?    -2.         -1.          2.          1.
  *** INPUT OF DATA FINISHED ***

  OBSERVABLE REALISATION A  B  C
  A MATRIX

       0         0    2.000         0         0         0

   1.000         0    1.000         0         0         0

       0     1.000   -2.000         0         0         0

       0         0        0         0         0     2.000

       0         0        0     1.000         0     1.000

       0         0        0         0     1.000    -2.000
  B MATRIX

  6.000   4.000

      0   1.000

  1.000   1.000

 -2.000  -2.000

 -7.000  -5.000

  2.000   1.000
  C MATRIX

       0         0    1.000         0         0         0

       0         0        0         0         0     1.000
```

```
E-VALUES
-2.00000000000              0   1.00000000000        0
-1.00000000000              0  -2.00000000000        0
 1.00000000000              0  -1.00000000000        0
** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **
 DISTINCT E-VALUES   3

LEADING E-VECTORS FOR E-VALUE  1
   OF RANK   1
 -1.000        0
      0        0
  1.000        0
      0   -1.000
      0        0
      0    1.000

LEADING E-VECTORS FOR E-VALUE  2
   OF RANK   1
      0    1.000
      0    -.500
      0    -.500
  1.000        0
  -.500        0
  -.500        0

LEADING E-VECTORS FOR E-VALUE  3
   OF RANK   1
      0     .667
      0    1.000
      0     .333
   .667        0
  1.000        0
   .333        0

TRANSFORMATION MATRIX FOR JORDAN FORM
 -1.000        0        0    1.000        0     .667
      0        0        0    -.500        0    1.000
  1.000        0        0    -.500        0     .333
      0   -1.000    1.000        0     .667        0
      0        0    -.500        0    1.000        0
      0    1.000    -.500        0     .333        0

INVERSE OF TRANSFORMATION MATRIX
   .333    -.667    1.333        0        0        0
      0        0        0     .333    -.667    1.333
      0        0        0    1.000   -1.000    1.000
  1.000   -1.000    1.000        0        0        0
      0        0        0     .500     .500     .500
   .500     .500     .500        0        0        0
 TRANSFORMED B MATRIX
  3.333    2.000
  6.667    4.000
  7.000    4.000
  7.000    4.000
 -3.500   -3.000
  3.500    3.000
 TRANSFORMED C MATRIX
  1.000        0        0    -.500        0     .333
      0    1.000    -.500        0     .333        0
```

TRANSFORMED A MATRIX IN JORDAN FORM

-2.000	0	0	0	0	0
0	-2.000	0	0	0	0
0	0	-1.000	0	-.000	0
0	0	0	-1.000	0	-.000
0	0	0	0	1.000	0
0	0	0	0	0	1.000

TRANSFORMED B

0	0	........row corr. to first block of first e-value 0;delete.
6.667	4.000	
7.000	4.000	
7.000	4.000	
-3.500	-3.000	
3.500	3.000	

TRANSFORMED B

6.667	4.000	
0	0	........row corr. to first blok of second e-value 0 ; delete .
7.000	4.000	
-3.500	-3.000	
3.500	3.000	
3.500	3.000	

TRANSFORMED B

6.667	4.000	
7.000	4.000	
0	0	........row corr. to first block of third e-value 0 ; delete .
3.500	3.000	
3.500	3.000	
3.500	3.000	

DIMENSION 3

A MATRIX

-2.000	0	0
0	-1.000	-.000
0	0	1.000

B MATRIX

6.667	4.000
7.000	4.000
3.500	3.000

C MATRIX

.500	-.500	.333
1.000	-.500	-.333

TOP

The order of the realisation has been reduced by 3. The rows of B are non-zero and since the jordan form is diagonal, the realisation is minimal .

```
.GO
ILLEGAL COMMAND.

.GO
    *** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM **

 *** START INPUT OF DATA ***
 TOLERANCE
  *              *
?  .000000000005
   ORDER OF MATRIX  - FORMAT 2I2
?  2 2


   ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


   ORDER OF DENOMINATOR
?  3


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
 *  *        *        *        *        *        *        *        *
?     5.        0.       1.
?     4.        1.       1.
?    -2.       -7.       2.
?    -2.       -5.       1.


  COEFFICIENTS OF DENOMINATOR
?      -2.       -1.       2.       1.
 *** INPUT OF DATA FINISHED ***


 DIMENSION   3


 A MATRIX
  -2.000        0        0
       0   -1.000    -.000
       0        0    1.000


 B MATRIX
   6.667    4.000
   7.000    4.000
   3.500    3.000


 C MATRIX
    .500    -.500     .333
   1.000    -.500    -.333
STOP
.CTIME
CTIME.     22.372 SEC.
```

Test 3

$$G(s) = \frac{1}{s^4}\begin{bmatrix} s^3-s^2+1 & 1 & -s^3+s^2-2 \\ 1.5s+1 & s+1 & -1.5s-2 \\ s^3-9s^2-s^2+1 & -s^2+1 & s^3-s-2 \end{bmatrix}$$

This example is taken from a paper by S. P. Panda and C. T. Chen, in IEEE Transactions on Automatic Control, February 1969.

The two results agree as far as the order of the realisation is concerned, but the actual matrices are different due to the different methods employed. (Both realisations of course are in the Jordan canonical form, and the Jordan blocks are the same, but their ordering is different.)

```
ANH,MI=14000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITHM
   *** START INPUT OF DATA ***
TOLERANCE
 *                      *
?  .00000000005
   ORDER OF MATRIX  -FORMAT 2I2
?  3 3

 ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  3 0 3 1 1 1 3 2 3


  ORDER OF DENOMINATOR
?  4


   COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *         *         *         *         *         *         *         *
?     1.         0.        -1.         1.
?     1.
?    -2.         0.         1.        -1.
?     1.        1.5
?     1.        1.
?    -2.       -1.5
?     1.       -1.        09.        1.
?     1.        0.        -1.
?    -2.       -1.         0.          1.


  COEFFICIENTS OF DENOMINATOR
 ?     0.         0.         0.         0.         1.
     *** INPUT OF DATA FINISHED ***
    DIMENSION  8

   A MATRIX
   -1.080     .649     .302    -.216        0        0        0        0
   -1.000     .600     .780    -.450     .625        0        0        0
     .667    -.400    1.480    -.700     .417   -1.000        0     .000
        0        0     .500    -.250    -.042    -.400     .833        0
        0        0    -.600     .300     .050     .480     .200    1.200
        0        0    1.000    -.500    -.917   -1.300    -.750    -.750
        0        0        0        0    1.000     .600     .500     .500
        0        0        0        0        0        0        0        0
   B MATRIX
        0        0        0
    -.000        0        0
        0        0        0
        0        0        0
    -.000        0        0
    5.000        0        0
   -9.000   -1.000        0
    1.000    1.000   -2.000
   C MATRIX
        0        0        0        0        0    1.000     .500     .500
        0    1.000        0        0        0        0        0        0
        0        0        0    1.000    -.167    -.600    -.500    -.500
 TOP

  CP       4.232 SECS.

 RUN COMPLETE.
```

```
LGO
      *** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FOR

*** START INPUT OF DATA ***
TOLERANCE
 *                 *
  .000000000005
   ORDER OF MATRIX  - FORMAT 2I2
?  3 3


  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  3 0 3 1 1 1 3 3 3


  ORDER OF DENOMINATOR
?  4


  COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *          *          *          *          *          *          *          *
?   1.          0.          -1.          1.
?   1.
?  -2.          0.          1.          -1.
?   1.          1.5
?   1.          1.
?  -2.         -1.5
?   1.         -1.          -9.          1.
?   1.          0.          -1.
?  -2.         -1.          0.          1.


 COEFFICIENTS OF DENOMINATOR
?    0.          0.          0.          0.        1.
  *** INPUT OF DATA FINISHED ***
STOP
```

OBSERVABLE REALISATION A B C

A MATRIX

0	0	0	0	0	0	0	0	0	0	0
1.000	0	0	0	0	0	0	0	0	0	0
0	1.000	0	0	0	0	0	0	0	0	0
0	0	1.000	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1.000	0	0	0	0	0	0
0	0	0	0	0	1.000	0	0	0	0	0
0	0	0	0	0	0	1.000	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1.000	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1.000	0

B MATRIX

1.000	1.000	-2.000
0	0	0
-1.000	0	1.000
1.000	0	-1.000
1.000	1.000	-2.000
1.500	1.000	-1.500
0	0	0
0	0	0
1.000	1.000	-2.000
-1.000	0	-1.000
-9.000	-1.000	0
1.000	0	1.000

C MATRIX


```
      0         0         0    1.000         0         0         0         0         0
      0         0

      0         0         0        0         0         0         0     1.000        0
      0         0

      0         0         0        0         0         0         0         0         0
      0     1.000
E-VALUES
                0                  0                   0                   0
                0                  0                   0                   0
                0                  0                   0                   0
                0                  0                   0                   0
                0                  0                   0                   0
                0                  0                   0                   0
** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **
 DISTINCT E-VALUES   1

LEADING E-VECTORS FOR E-VALUE  1
  OF RANK   4
  1.000         0         0
      0         0         0
      0         0         0
      0         0         0
      0     1.000         0
      0         0         0
      0         0         0
      0         0         0
      0         0     1.000
      0         0         0
      0         0         0
      0         0         0

TRANSFORMATION MATRIX FOR JORDAN FORM
      0         0         0    1.000         0         0         0         0         0

      0         0
      0         0     1.000        0         0         0         0         0         0

      0         0
      0     1.000         0        0         0         0         0         0         0

      0         0
  1.000         0         0        0         0         0         0         0         0

      0         0
      0         0         0        0         0         0         0     1.000         0

      0         0
      0         0         0        0         0         0     1.000        0         0

      0         0
      0         0         0        0         0     1.000         0        0         0

      0         0
      0         0         0        0     1.000         0         0        0         0

      0         0
      0         0         0        0         0         0         0        0         0

      0     1.000
      0         0         0        0         0         0         0        0         0

  1.000         0
      0         0         0        0         0         0         0        0         0
```

```
INVERSE OF TRANSFORMATION MATRIX
     0        0        0   1.000      0        0        0        0        0

     0        0
     0        0    1.000      0        0        0        0        0        0

     0        0
     0    1.000       0       0        0        0        0        0        0

     0        0
 1.000        0       0       0        0        0        0        0        0

     0        0
     0        0       0       0        0        0        0   1.000        0

     0        0
      0        0       0       0        0        0   1.000        0        0

     0        0
     0        0       0       0        0   1.000        0        0        0

     0        0
     0        0       0       0   1.000        0        0        0        0

     0        0
     0        0       0       0        0        0        0        0        0

     0    1.000
     0        0       0       0        0        0        0        0        0

 1.000        0
      0        0       0       0        0        0        0        0

     0        0
     0        0       0       0        0        0        0        0    1.000

     0        0
 TRANSFORMED E MATRIX
  1.000       0  -1.000
 -1.000       0   1.000
      0       0       0
  1.000   1.000  -2.000
      0       0       0
      0       0       0
  1.500   1.000  -1.500
  1.000   1.000  -2.000
  1.000       0   1.000
 -9.000  -1.000       0
 -1.000       0  -1.000
  1.000   1.000  -2.000
 TRANSFORMED C MATRIX
  1.000       0       0       0       0       0       0       0       0

      0       0
      0       0       0       0   1.000       0       0       0       0

      0       0
      0       0       0       0       0       0       0       0   1.000

[illegible]  -.667  -.000
```

0 0

TRANSFORMED A MATRIX IN JORDAN FORM

0	1.000	0	0	0	0	0	0	0	0	0
0	0	1.000	0	0	0	0	0	0	0	0
0	0	0	1.000	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1.000	0	0	0	0	0
0	0	0	0	0	0	1.000	0	0	0	0
0	0	0	0	0	0	0	1.000	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1.000	0
0	0	0	0	0	0	0	0	0	0	1.000
0	0	0	0	0	0	0	0	0	0	0

TRANSFORMED B

0	0	-2.000
8.000	1.000	1.000
1.000	0	1.000
0	0	0
0	0	0
0	0	0
1.500	1.000	-1.500
1.000	1.000	-2.000
1.000	0	1.000
-9.000	-1.000	0
-1.000	0	-1.000
1.000	1.000	-2.000

```
TRANSFORMED B
      0           0   -2.000
  8.000       1.000    1.000
  1.000           0    1.000
      0           0        0 ....last row of first block 0 ; delete.
- - - - - - - - - - - - - - -
 -1.000           0   -1.000
  9.000       1.000        0
  2.500       1.000    -.500
      0           0        0 ....last row of second block 0;delete.
- - - - - - - - - - - - - - -
  1.000           0    1.000
 -9.000      -1.000        0
 -1.000           0   -1.000
  1.000       1.000   -2.000
TRANSFORMED B
   .400           0   -1.600
  4.400        .600    1.000
      0       -.400    1.200 *
- - - - - - - - - - - - - - -
 -1.000           0   -1.000
  9.000       1.000        0
  2.500       1.000    -.500 *
- - - - - - - - - - - - - - -
  1.000           0    1.000
 -9.000      -1.000        0
 -1.000           0   -1.000
  1.000       1.000   -2.000 *  last rows of blocks of B not linearly
- - - - - - - - - - - - - - -
  1.000       1.000   -2.000    independent;reduction possible.
  1.000       1.000   -2.000
TRANSFORMED B
   .400           0   -1.600
  4.400        .600    1.000
      0       -.400    1.200
- - - - - - - - - - - - - - -
 -1.000           0   -1.000
  9.000       1.000        0
  2.500       1.000    -.500
- - - - - - - - - - - - - - -
  1.000           0    1.000
 -9.000      -1.000        0
 -1.000           0   -1.000
  1.000       1.000   -2.000
- - - - - - - - - - - - - - -
  1.000       1.000   -2.000
  1.000       1.000   -2.000
TRANSFORMED B
   .400           0   -1.600
  4.400        .600    1.000
      0       -.400    1.200
- - - - - - - - - - - - - - -
 21.500       2.500   -1.000
 11.500       1.000    2.500
      0      -1.500    4.500
- - - - - - - - - - - - - - -
  1.000           0    1.000
 -9.000      -1.000        0
 -1.000           0   -1.000
  1.000       1.000   -2.000
- - - - - - - - - - - - - - -
  1.000       1.000   -2.000
  1.000       1.000   -2.000
TRANSFORMED B
 -5.333       -.667   -1.333
  1.333        .333     .333
      0       -.000     .000 ....row 0 ; delete.
- - - - - - - - - - - - - - -
 21.500       2.500   -1.000
 11.500       1.000    2.500
      0      -1.500    4.500
- - - - - - - - - - - - - - -
  1.000           0    1.000
 -9.000      -1.000        0
 -1.000           0   -1.000
  1.000       1.000   -2.000
- - - - - - - - - - - - - - -
  1.000       1.000   -2.000
  1.000       1.000   -2.000
TRANSFORMED B
```

```
TRANSFORMED B
-4.000    -.667    -.000
     0   -1.000    3.000
21.500    2.500   -1.000
11.500    1.000    2.500
     0   -1.500    4.500
 1.000        0    1.000
-9.000   -1.000        0
-1.000        0   -1.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000..last rows still not independent.
TRANSFORMED B
-4.000    -.667    -.000
     0   -1.000    3.000
21.500    2.500   -1.000
11.500    1.000    2.500
     0   -1.500    4.500
 1.000        0    1.000
-9.000   -1.000        0
-1.000        0   -1.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
TRANSFORMED B
-11.667  -1.333   -1.667
     0    -.000     .000 ......row 0 ; delete.
21.500    2.500   -1.000
11.500    1.000    2.500
     0   -1.500    4.500
 1.000        0    1.000
-9.000   -1.000        0
-1.000        0   -1.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
TRANSFORMED B
     0   10.333  -25.000
21.500    2.500   -1.000
11.500    1.000    2.500
     0   -1.500    4.500
 1.000        0    1.000
-9.000   -1.000        0
-1.000        0   -1.000
 1.000    1.000   -2.000xxxxxxrows now independent, dimension 8.
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
TRANSFORMED B
     0   10.333  -25.000
21.500    2.500   -1.000
11.500    1.000    2.500
     0   -1.500    4.500
 1.000        0    1.000
-9.000   -1.000        0
-1.000        0   -1.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
 1.000    1.000   -2.000
```

TRANSFORMED B

0	0	6.000
21.500	2.500	-1.000
11.500	1.000	2.500
0	-1.500	4.500
1.000	0	1.000
-9.000	-1.000	0
-1.000	0	-1.000
1.000	1.000	-2.000
1.000	1.000	-2.000
1.000	1.000	-2.000
1.000	1.000	-2.000
1.000	1.000	-2.000

DIMENSION 8

A MATRIX

0	0	0	0	0	0	0	0
0	0	1.000	0	0	0	0	0
0	0	0	1.000	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1.000	0	0
0	0	0	0	0	0	1.000	0
0	0	0	0	0	0	0	1.000
0	0	0	0	0	0	0	0

B MATRIX

0	0	6.000
21.500	2.500	-1.000
11.500	1.000	2.500
0	-1.500	4.500
1.000	0	1.000
-9.000	-1.000	0
-1.000	0	-1.000
1.000	1.000	-2.000

C MATRIX

1.000	.667	.667	-6.889	1.000	1.000	1.333	-11.667
0	1.000	0	0	1.000	2.500	0	0
0	0	0	0	1.000	0	0	0

STOP

```
*** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM

*** START INPUT OF DATA ***
TOLERANCE
  *               *
  .000000000005
  ORDER OF MATRIX  - FORMAT 2I2
  3 3

  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
  2 0 3 1 1 1 3 2 3

  ORDER OF DENOMINATOR
  4

   COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *        *        *        *        *        *        *        *
     1.        0.        -1.        1.
     1.
    -2.        0.         1.       -1.
     1.        1.5
     1.        1.
    -2.       -1.5
     1.       -1.        -3.        1.
     1.        0.        -1.
    -2.       -1.         0.        1.

 COEFFICIENTS OF DENOMINATOR
      0.         0.         0.         0.         1.
 *** INPUT OF DATA FINISHED ***

 DIMENSION  8

 A MATRIX
        0        0        0        0        0        0        0        0
        0        0    1.000        0        0        0        0        0
        0        0        0    1.000        0        0        0        0
        0        0        0        0        0        0        0        0
        0        0        0        0        0    1.000        0        0
        0        0        0        0        0        0    1.000        0
        0        0        0        0        0        0        0    1.000
        0        0        0        0        0        0        0        0

 B MATRIX
        0        0    6.000
   31.500    2.500   -1.000
   11.500    1.000    2.500
        0   -1.500    4.500
    1.000        0    1.000
   -9.000   -1.000        0
   -1.000        0   -1.000
    1.000    1.000   -2.000

 C MATRIX
    1.000     .667     .667   -6.333    1.000    1.000    1.333  -11.667
        0    1.000        0        0    1.000    2.500        0        0
        0        0        0        0    1.000        0        0        0
STOP
 STIME
 TIME.       26.097 SEC.
```

Test 4 $g(s)=\frac{.5s^2+2.5s+1}{s^3+6s^2+10s+8}$

```
RNH,MI=14000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITHM
    *** START INPUT OF DATA ***
 TOLERANCE
  *                 *
?  .000000000005
   ORDER OF MATRIX  -FORMAT 2I2
?  1 1

 ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2


   ORDER OF DENOMINATOR
?  3


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
     *         *         *         *         *         *         *         *
?      1.        2.5        0.5


  COEFFICIENTS OF DENOMINATOR
?      8.        10.        6.         1.
    *** INPUT OF DATA FINISHED ***
    DIMENSION   3

   A MATRIX
   -4.500     .313        0
   -4.000     .100     -.960
  -10.000    2.250    -1.600
   B MATRIX
        0
        0
    2.500
   C MATRIX
    1.000    -.125      .200
STOP

 CP          4.116 SECS.

RUN COMPLETE.
```

```
/L30
    ****TEST RUN FOR MINIMAL REALISATION USING THE JORDAN  CANONICAL F

  *** START INPUT OF DATA ***
 TOLERANCE
   *              *
?  .00000000005
   ORDER OF MATRIX  -FORMAT 2I2
?  1 1

  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2


   ORDER OF DENOMINATOR
?  3


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
   *       *       *       *       *       *       *       *
?      1.       2.5     .5


  COEFFICIENTS OF DENOMINATOR
?      3.       10.      5.       1.
    *** INPUT OF DATA FINISHED ***

   OBSERVABLE REALISATION A  B  C
   A MATRIX

      0         0  -3.000

   1.000        0  -10.000

      0    1.000   -6.000
   B MATRIX

   1.000

   2.500

    .500
   C MATRIX

      0         0   1.000
  E-VALUES
  -4.00000000000        0  -1.00000000000  -1.00000000000
  -1.00000000000   1.00000000000
  COMPLEX E-VALUES , MATRICES WITH COMPLEX ELEMENTS
   DISTINCT E-VALUES  3

  LEADING E-VECTORS FOR E-VALUE  1
    OF RANK  1
      1.000       0
      1.000       0
       .500       0

  LEADING E-VECTORS FOR E-VALUE  2
    OF RANK  1
      1.000       0
       .750    .500
       .125    .125
```

```
LEADING E-VECTORS FOR E-VALUE  3
  OF RANK  1
    1.000        0
     .750    -.500
     .125    -.125
TRANSFORMATION MATRIX FOR JORDAN FORM
   1.000        0   1.000       0   1.000       0
   1.000        0    .750    .500    .750   -.500
    .500        0    .125    .125    .125   -.125

 INVERSE OF TRANSFORMATION MATRIX
    .200     .000   -.300    .000   3.200       0
    .400     .300    .400  -1.200  -1.600    .300
    .400    -.300    .400   1.200  -1.600   -.300

 TRANSFORMED B MATRIX

  -.300    .000

   .600  -1.300

   .600   1.300

 TRANSFORMED C MATRIX

   .500        0    .125    .125    .125   -.125

  TRANSFORMED A MATRIX      IN JORDAN FORM

 -4.000    -.000 | -.000     .000    -.000    -.000
 ----------------+----------------+------------------
  -.000     .000 |-1.000   -1.000 |   .000     .000
                 +----------------+------------------
  -.000     .000    .000    -.000 |-1.000    1.000

DIMENSION  3
 *** MINIMAL REALISATION ***
*** MATRICES IN REAL FORM AFTER TRANSFORMATION ***

  A MATRIX
   -4.000    -.000    -.000
    -.000   -1.      -1.
    -.000    1.      -1.000
```

..... **note that A is not in Jordan form because of transformation.**

```
   B MATRIX
    -.300
   -1.300
     .600
   C MATRIX
     .500     .250     .250
STOP.

 JOB ACTIVE.

/CTIME
CTIME.      3.670 SEC.
```

Test 5

$$G(s)=\begin{bmatrix} \frac{1}{(s+1)^2}-\frac{1}{s+1} & \frac{1}{s+1} \\ \frac{1}{s}+\frac{1}{s+1} & \frac{1}{s+1} \end{bmatrix}$$

Check $G(s)=C(sI-A)^{-1}B$

$$=\begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 \end{bmatrix}\begin{bmatrix} \frac{1}{s+1} & \frac{1}{(s+1)^2} & & \\ & \frac{1}{s+1} & & \\ & & \frac{1}{s+1} & \\ & & & \frac{1}{s} \end{bmatrix}\begin{bmatrix} 0 & -1 \\ -1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$=\begin{bmatrix} -\frac{1}{s+1} & -\frac{1}{(s+1)^2}+\frac{1}{s+1} & 0 & 0 \\ 0 & -\frac{1}{s+1} & \frac{1}{s+1} & \frac{1}{s} \end{bmatrix}\begin{bmatrix} 0 & -1 \\ -1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$=\begin{bmatrix} \frac{1}{(s+1)^2}-\frac{1}{s+1} & \frac{1}{s+1} \\ \frac{1}{s+1}+\frac{1}{s} & \frac{1}{s+1} \end{bmatrix}$$

```
IDLE.

RNH,MI=14000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITH
    *** START INPUT OF DATA ***
 TOLERANCE
   *        *
?   .000000000005
   ORDER OF MATRIX   -FORMAT 2I2
?  2 2

  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


   ORDER OF DENOMINATOR
?  3


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
    *         *         *         *         *         *         *         *
?         0.          0.          -1.
?         0.          1.          1.
?         1.          3.          2.
?         0.          1.          1.


  COEFFICIENTS OF DENOMINATOR
?         0.          1.          3.          1.
   *** INPUT OF DATA FINISHED ***
   DIMENSION   4

  A MATRIX
   -.667   -.333       0       0
    .333   -.333   -.983       0
   1.000  -1.000  -1.000       0
       0   1.000   -.333  -1.000
  B MATRIX
       0       0
       0       0
  -3.000       0
   2.000   1.000
  C MATRIX
       0       0   1.000   1.000
       0       0       0   1.000
STOP

 CP       4.024 SECS.

RUN COMPLETE.
```

```
LSD
      *** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM
 *** START INPUT OF DATA ***
 TOLERANCE
   *              *
?   .000000000005
    ORDER OF MATRIX  - FORMAT 2I2
?   2 2

    ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?   2 2 2 2

    ORDER OF DENOMINATOR
?   3

     COEFFICIENTS OF NUMERATOR POLYNOMIALS
    *       *       *       *       *       *       *       *
?       0.        0.       -1.
?       0.        1.        1.
?       1.        3.        2.
?       0.        1.        1.

  COEFFICIENTS OF DENOMINATOR
?       0.        1.        2.        1.
 *** INPUT OF DATA FINISHED ***

   OBSERVABLE REALISATION A  B  C
   A MATRIX

       0   -1.000        0       0

   1.000   -2.000        0       0

       0        0        0       0

       0        0    1.000  -1.000
   B MATRIX

       0    1.000

  -1.000    1.000

   1.000        0

   2.000    1.000
   C MATRIX

       0    1.000        0       0

       0        0        0   1.000
  E-VALUES
  -1.00000000000                      0   -1.00000000000           0
  -1.00000000000                      0                0           0
 ** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **
  DISTINCT E-VALUES   2
```

LEADING E-VECTORS FOR E-VALUE 1
OF RANK 2

```
     0
 1.000
     0
     0
```

LEADING E-VECTORS FOR E-VALUE 1
OF RANK 1

```
     0
     0
     0
 1.000
```

LEADING E-VECTORS FOR E-VALUE 3
OF RANK 1

```
     0
     0
 1.000
 1.000
```

TRANSFORMATION MATRIX FOR JORDAN FORM

-1.000	0	0	0
-1.000	1.000	0	0
0	0	0	1.000
0	0	1.000	1.000

INVERSE OF TRANSFORMATION MATRIX

-1.000	0	0	0
-1.000	1.000	0	0
0	0	-1.000	1.000
0	0	1.000	0

TRANSFORMED B MATRIX

0	-1.000
-1.000	0
1.000	1.000
1.000	0

TRANSFORMED C MATRIX

-1.000	1.000	0	0
0	0	1.000	1.000

TRANSFORMED A MATRIX IN JORDAN FORM

-1.000	1.000	0	0
0	-1.000	0	0
0	0	-1.000	0
0	0	0	0

TRANSFORMED B

0	-1.000
-1.000	0
0	1.000
1.000	0

......last rows of blocks corr. to different e-values independent ; realisation minimal.

```
DIMENSION   1

A MATRIX
 -1.000    1.000        0        0
      0   -1.000        0        0
      0        0   -1.000        0
      0        0        0        0

B MATRIX
      0   -1.000
 -1.000        0
      0    1.000
  1.000        0

C MATRIX
 -1.000    1.000        0        0
      0   -1.000    1.000    1.000
STOP
/CTIME
CTIME.     17.390 SEC.
/
```

```
LGO
    *** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM ***

 *** START INPUT OF DATA ***
 TOLERANCE
  *         *
?  .000000000005
   ORDER OF MATRIX  - FORMAT 2I2
?  2 2


   ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


   ORDER OF DENOMINATOR
?  3


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *         *         *         *         *         *         *         *
?     0.        0.        -1.
?     0.        1.         1.
?     1.        3.         2.
?     0.        1.         1.


  COEFFICIENTS OF DENOMINATOR
?     0.        1.         2.         1.
  *** INPUT OF DATA FINISHED ***


  DIMENSION   4


  A MATRIX
   -1.000   1.000        0         0
        0  -1.000        0         0
        0       0   -1.000         0
        0       0        0         0


  B MATRIX
        0  -1.000
   -1.000       0
        0   1.000
    1.000       0


  C MATRIX
   -1.000   1.000        0         0
        0  -1.000    1.000     1.000
STOP
/ETIME
ETIME,     23.053 SEC.
```

Test 7

$$G(s)=\frac{1}{s^3}\begin{bmatrix} s^2+1 & 2s^2+s \\ s^2+3s & 2s^2 \end{bmatrix}$$

Check : $G(s) = C(sI-A)^{-1}B$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 1 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{s} & \frac{1}{s^2} & \frac{1}{s^3} & \\ & \frac{1}{s} & \frac{1}{s^2} & \\ & & \frac{1}{s} & \\ & & & \frac{1}{s} \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{s} & \frac{1}{s} & \frac{1}{s} & 0 \\ 0 & \frac{3}{s} & \frac{3}{s}+\frac{1}{s} & \frac{1}{s} \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 0 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} \frac{1}{s}+\frac{1}{s} & \frac{2}{s}+\frac{1}{s} \\ \frac{3}{s}+\frac{1}{s} & \frac{3}{s}+\frac{1}{s} \end{bmatrix}$$

$$= \frac{1}{s^3}\begin{bmatrix} s+1 & s(2s+1) \\ s(3+s) & 2s^2 \end{bmatrix}$$

```
RNH,MI=14000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORIT
    *** START INPUT OF DATA ***
 TOLERANCE
   *              *
?  .000000000005
   ORDER OF MATRIX   -FORMAT 2I2
?  2 2

  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2


   ORDER OF DENOMINATOR
?  3


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
   *       *       *       *       *       *       *       *
?      1.        0.        1.
?      0.        1.        2.
?      0.        3.        1.
?      0.        0.        2.


  COEFFICIENTS OF DENOMINATOR
?      0.        0.        0.        1.
    *** INPUT OF DATA FINISHED ***
    DIMENSION   4

   A MATRIX
         0         0     -.167        0
     1.000         0    -1.167      .500
         0         0         0        0
         0         0     1.000        0
   B MATRIX
         0         0
         0         0
     3.000         0
     1.000     2.000
   C MATRIX
         0     1.000         0     1.000
         0         0         0     1.000
STOP

 CP        4.026 SECS.

RUN COMPLETE.
```

```
LGO
    *** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM

 *** START INPUT OF DATA ***
 TOLERANCE
   *                *
?  .000000000005
   ORDER OF MATRIX   - FORMAT 2I2
?  2 2

   ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 2

   ORDER OF DENOMINATOR
?  3

    COEFFICIENTS OF NUMERATOR POLYNOMIALS
   *          *          *          *          *          *          *          *
?       1.         0.         1.
?       0.         1.         2.
?       0.         3.         1.
?       0.         0.         3.

  COEFFICIENTS OF DENOMINATOR
?       0.         0.         0.         1.
  *** INPUT OF DATA FINISHED ***

   OBSERVABLE REALISATION A  B  C
   A MATRIX

       0         0         0         0         0

   1.000         0         0         0         0

       0     1.000         0         0         0

       0         0         0         0         0

       0         0         0     1.000         0
   B MATRIX

   1.000         0

       0     1.000

   1.000     2.000

   3.000         0

   1.000     2.000
   C MATRIX

       0         0     1.000         0         0

       0         0         0         0     1.000
```

```
E-VALUES
              0                   0                   0                   0
              0                   0                   0                   0
              0                   0
** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **
 DISTINCT E-VALUES   1

LEADING E-VECTORS FOR E-VALUE   1
  OF RANK   3
  1.000
      0
      0
      0
      0

LEADING E-VECTORS FOR E-VALUE   1
  OF RANK   2
      0
      0
      0
  1.000
      0

TRANSFORMATION MATRIX FOR JORDAN FORM
      0         0     1.000         0         0
      0     1.000         0         0         0
  1.000         0         0         0         0
      0         0         0         0     1.000
      0         0         0     1.000         0

 INVERSE OF TRANSFORMATION MATRIX
      0         0     1.000         0         0
      0     1.000         0         0         0
  1.000         0         0         0         0
      0         0         0         0     1.000
      0         0         0     1.000         0
  TRANSFORMED B MATRIX
  1.000     2.000
      0     1.000
  1.000         0
  1.000     2.000
  3.000         0
  TRANSFORMED C MATRIX
  1.000         0         0         0         0
      0         0         0     1.000         0

  TRANSFORMED A MATRIX      IN JORDAN FORM

      0     1.000         0  |      0         0
                             |
      0         0     1.000  |      0         0
                             |
      0         0         0  |      0         0
  ---------------------------+-------------------
      0         0         0  |      0     1.000
                             |
      0         0         0  |      0         0
```

```
TRANSFORMED A
  1.000    2.000
      0    1.000
  1.000        0
  -----------------
  1.000   -1.000
      0        0
 TRANSFORMED B
  1.000    2.000
      0    1.000
  1.000        0
      0   -1.000
      0        0


DIMENSION   4


A MATRIX
      0    1.000        0  |      0
      0        0    1.000  |      0
      0        0        0  |      0
  -------------------------+--------
      0        0        0  |      0


B MATRIX
  1.000    2.000
      0    1.000
  1.000        0
  -----------------
      0   -1.000


C MATRIX
  1.000        0        0  |      0
      0    3.000    1.000  |  1.000
STOP
/ETIME
CTIME.      7.398 SEC.
```

```
USO
     *** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM ***

 *** START INPUT OF DATA ***
 TOLERANCE
  *              *
? .000000000005
  ORDER OF MATRIX  - FORMAT 2I2.
? 2 2


  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
? 2 2 2 2


  ORDER OF DENOMINATOR
? 3


   COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *       *       *       *       *       *       *       *
?    1.        0.        1.
?    0.        1.        2.
?    0.        2.        1.
?    0.        0.        2.


  COEFFICIENTS OF DENOMINATOR
?    0.        0.        0.        1.
 *** INPUT OF DATA FINISHED ***


  DIMENSION   4


  A MATRIX
          0     1.000         0         0
          0         0     1.000         0
          0         0         0         0
          0         0         0         0


  B MATRIX
    1.000     2.000
        0     1.000
    1.000         0
        0    -1.000


  C MATRIX
    1.000         0         0         0
        0     3.000     1.000     1.000
STOP
CITIME
CTIME.      29.083 SEC.
```

Test 8

$$G(s)=\begin{bmatrix} \frac{1.}{1+4s} & \frac{.7}{1+5s} & \frac{.3}{1+5s} & \frac{.2}{1+5s} \\ \frac{.6}{1+5s} & \frac{1.}{1+4s} & \frac{.4}{1+5s} & \frac{.35}{1+5s} \\ \frac{.35}{1+5s} & \frac{.4}{1+5s} & \frac{1.}{1+4s} & \frac{.6}{1+5s} \\ \frac{.2}{1+5s} & \frac{.3}{1+5s} & \frac{.7}{1+5s} & \frac{1.}{1+4s} \end{bmatrix}$$

$$= \frac{1}{s^2+0.45s+0.05} \begin{bmatrix} .05+.25s & .035+.14s & .015+.06s & .01+.04s \\ .03+.12s & .05+.25s & .02+.08s & .0175+.04s \\ .0175+.07s & .02+.08s & .05+.25s & .03+.12s \\ .01+.04s & .015+.06s & .035+.14s & .05+.25s \end{bmatrix}$$

This example is taken from H. H. Rosenbrock' s book "Computer aided control system design". The form of the matrix had to be changed , so that acommon denominator with leading coefficient 1, could multiply the matrix.

**** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITHM

*** START INPUT OF DATA ***

TOLERANCE

? .00000000005

ORDER OF MATRIX -FORMAT 2I2

? 4 4

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

? 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

ORDER OF DENOMINATOR

? 2

COEFFICIENTS OF NUMERATOR POLYNOMIALS

?	.05	.25
?	.035	.14
?	.015	.06
?	.01	.04
?	.03	.12
?	.05	.25
?	.02	.08
?	.0175	.07
?	.0175	.07
?	.02	.08
?	.05	.25
?	.03	.125
?	.01	.04
?	.015	.06
?	.035	.14
?	.05	.25

COEFFICIENTS OF DENOMINATOR

? .05 .45 1.

*** INPUT OF DATA FINISHED ***

DIMENSION 8

A MATRIX

-.200	-.000	.000	-.000	-.000	0	0	0
-.021	-.142	.029	.021	.003	-.002	.000	0
-.005	-.009	-.210	-.032	-.000	.000	.002	0
-.011	-.001	-.036	-.166	.001	.001	-.002	.001
1.000	-1.457	-.232	-.491	-.237	.029	.007	-.001
0	1.000	.572	.379	.028	-.261	.008	.009
0	0	-.500	1.352	.012	.014	-.285	.020
0	0	1.000	-.704	-.002	.005	.039	-.250

B MATRIX

0	0	0	0
.000	0	0	0
0	0	.000	0
0	0	0	0
.180	.000	0	0
.097	.222	0	0
.050	.050	.180	0
.040	.060	.140	.250

C MATRIX

0	0	0	0	1.000	.541	.209	.160
0	0	0	0	0	1.000	.227	.280
0	0	0	0	0	0	1.000	.500
0	0	0	0	0	0	0	1.000

STOP

ALGO

*** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM

*** START INPUT OF DATA ***
TOLERANCE

? .000000000005
ORDER OF MATRIX - FORMAT 2I2
? 4 4

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
? 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

ORDER OF DENOMINATOR
? 2

COEFFICIENTS OF NUMERATOR POLYNOMIALS

? .05 .25
? .035 .14
? .015 .06
? .01 .04
? .03 .12
? .05 .25
? .02 .08
? .0175 .07
? .0175 .07
? .02 .08
? .05 .25
? .03 .12
? .01 .04
? .015 .06
? .035 .14
? .05 .25

COEFFICIENTS OF DENOMINATOR
? .05 .45 1.
*** INPUT OF DATA FINISHED ***

OBSERVABLE REALISATION A B C
A MATRIX

0	-.050	0	0	0	0	0	0
1.000	-.450	0	0	0	0	0	0
0	0	0	-.050	0	0	0	0
0	0	1.000	-.450	0	0	0	0
0	0	0	0	0	-.050	0	0
0	0	0	0	1.000	-.450	0	0
0	0	0	0	0	0	0	-.050
0	0	0	0	0	0	1.000	-.450

B MATRIX

.050	.035	.015	.010
.250	.140	.060	.040
.030	.050	.020	.017
.120	.250	.080	.070
.017	.020	.050	.030
.070	.080	.250	.120
.010	.015	.035	.050
.040	.060	.140	.250

C MATRIX

0	1.000	0	0	0	0	0	0
0	0	0	1.000	0	0	0	0
0	0	0	0	0	1.000	0	0
0	0	0	0	0	0	0	1.000

E-VALUES

-.25000000000	0	-.20000000000	0
-.25000000000	0	-.20000000000	0
-.25000000000	0	-.20000000000	0
-.25000000000	0	-.20000000000	0

** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **

DISTINCT E-VALUES 2

LEADING E-VECTORS FOR E-VALUE 1
OF RANK 1

.200	0	0	0
1.000	0	0	0
0	0	.200	0
0	0	1.000	0
0	.200	0	0
0	1.000	0	0
0	0	0	.200
0	0	0	1.000

LEADING E-VECTORS FOR E-VALUE 2
OF RANK 1

.250	0	0	0
1.000	0	0	0
0	0	.250	0
0	0	1.000	0
0	.250	0	0
0	1.000	0	0
0	0	0	.250
0	0	0	1.000

TRANSFORMATION MATRIX FOR JORDAN FORM

.200	0	0	0	.250	0	0	0
1.000	0	0	0	1.000	0	0	0
0	0	.200	0	0	0	.250	0
0	0	1.000	0	0	0	1.000	0
0	.200	0	0	0	.250	0	0
0	1.000	0	0	0	1.000	0	0
0	0	0	.200	0	0	0	.250
0	0	0	1.000	0	0	0	1.000

INVERSE OF TRANSFORMATION MATRIX

-20.000	5.000	0	0	0	0	0	0
0	0	0	0	-20.000	5.000	0	0
0	0	-20.000	5.000	0	0	0	0
0	0	0	0	0	0	-20.000	5.000
20.000	-4.000	0	0	0	0	0	0
0	0	0	0	20.000	-4.000	0	0
0	0	20.000	-4.000	0	0	0	0
0	0	0	0	0	0	20.000	-4.000

TRANSFORMED B MATRIX

.250	.000	.000	.000
.000	.000	.250	.000
.000	.250	.000	.000
.000	.000	.000	.250
.000	.140	.060	.040
.070	.080	.000	.120
.120	.000	.080	.070
.040	.060	.140	.000

TRANSFORMED C MATRIX

1.000	0	0	0	1.000	0	0	0
0	0	1.000	0	0	0	1.000	0
0	1.000	0	0	0	1.000	0	0
0	0	0	1.000	0	0	0	1.000

TRANSFORMED A MATRIX IN JORDAN FORM

-.250	0	0	0	.000	0	0	0
0	-.250	0	0	0	.000	0	0
0	0	-.250	0	0	0	.000	0
0	0	0	-.250	0	0	0	.000
.000	0	0	0	-.200	0	0	0
0	.000	0	0	0	-.200	0	0
0	0	.000	0	0	0	-.200	0
0	0	0	.000	0	0	0	-.200

TRANSFORMED B

.250	.0	.000	0
.000	.0	.250	.000
.000	.25	.000	.000
.000	.0	.000	.250
.000	.14	.060	.040
.070	.08	.000	.120
.120	.00	.080	.070
.040	.06	.140	.000

.....start transformations for blocks of first e-value.

TRANSFORMED B

.250	.000	.000	0
.000	.000	.250	0
.000	.250	.000	.000
.000	.000	.000	.250
.000	.140	.060	.040
.070	.080	.000	.120
.120	.000	.080	.070
.040	.060	.140	.000

.250	.000	.000	0
.000	.000	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	.140	.060	.040
.070	.080	.000	.120
.120	.000	.080	.070
.040	.060	.140	.000

TRANSFORMED B

.250	0	.000	0
.000	.000	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	.140	.060	.040
.070	.080	.000	.120
.120	.000	.080	.070
.040	.060	.140	.000

TRANSFORMED B

.250	0	.000	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	.140	.060	.040
.070	.080	.000	.120
.120	.000	.080	.070
.040	.060	.140	.000

TRANSFORMED B

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	.140	.060	.040
.070	.080	.000	.120
.120	.000	.080	.070
.040	.060	.140	.000

TRANSFORMED B

.....start transformations for blocks of second e-value.

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
0	.140	.060	.040
.070	.080	.000	.120
.120	.000	.080	.070
.040	.060	.140	.000

TRANSFORMED B

.250	[illegible]	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
0	.140	.060	.040
0	−.025	−.245	.120
.120	.000	.080	.070
.040	.060	.140	.000

TRANSFORMED B

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
0	.140	.060	.040
0	−.025	−.245	.120
.000	−.180	−.340	.070
.040	.060	.140	.000

TRANSFORMED B

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	0	−.204	.094
0	−.025	−.245	.120

TRANSFORMED B

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	0	-.204	.094
-.000	0	-.198	.110
.000	-.180	-.340	.070
.040	.060	.140	.000

TRANSFORMED B

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	0	0	-.020
-.000	0	-.198	.110
.000	-.180	-.340	.070
.040	.060	.140	.000

DIMENSION 8

A MATRIX

-.250	0	0	0	.000	0	0	0
0	-.250	0	0	0	.000	0	0
0	0	-.250	0	0	0	.000	0
0	0	0	-.250	0	0	0	.000
.000	0	0	0	-.200	0	0	0
0	.000	0	0	0	-.200	0	0
0	0	.000	0	0	0	-.200	0
0	0	0	.000	0	0	0	-.200

B MATRIX

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	0	0	-.020
-.000	0	-.198	.110
.000	-.180	-.340	.070
.040	.060	.140	.000

C MATRIX

1.000	.000	.000	.000	1.000	1.034	-.778	.000
0	0	1.000	.000	0	0	1.000	3.000
0	1.000	.000	.000	0	1.000	.139	1.750
0	0	0	1.000	0	0	0	1.000

STOP

UMACP15 TIME-OUT. 15.10.33.
UMACP15 CP 13.684 SEC.

```
    *** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM

 *** START INPUT OF DATA ***
 TOLERANCE
?  .000000000005
   ORDER OF MATRIX  - FORMAT 3I2
?  4 4

   ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

   ORDER OF DENOMINATOR
?  2

    COEFFICIENTS OF NUMERATOR POLYNOMIALS
?   .05        .25
?   .035       .14
?   .015       .06
?   .01        .04
?   .03        .12
?   .05        .25
?   .02        .08
?   .0175      .07
?   .0175      .07
?   .02        .08
?   .05        .25
?   .03        .12
?   .01        .04
?   .015       .06
?   .035       .14
?   .05        .25

   COEFFICIENTS OF DENOMINATOR
?   .05        .45        1.
 *** INPUT OF DATA FINISHED ***
```

DIMENSION 8

A MATRIX

-.250	0	0	0	.000	0	0	0
0	-.250	0	0	0	.000	0	0
0	0	-.250	0	0	0	.000	0
0	0	0	-.250	0	0	0	.000
.000	0	0	0	-.200	0	0	0
0	.000	0	0	0	-.200	0	0
0	0	.000	0	0	0	-.200	0
0	0	0	.000	0	0	0	-.200

B MATRIX

.250	0	0	0
.000	0	.250	0
.000	.250	.000	0
.000	.000	.000	.250
.000	0	0	-.020
-.000	0	-.198	.110
.000	-.180	-.340	.070
.040	.080	.140	.000

C MATRIX

1.000	.000	.000	.000	1.000	1.064	-.778	.000
0	0	1.000	.000	0	0	1.000	3.000
0	1.000	.000	.000	0	1.000	.189	1.750
0	0	0	1.000	0	0	0	1.000

STOP

TIME

TIME. 31,630 SEC.

Test 9

$$G(s) = \begin{bmatrix} \frac{1}{(s+1)^2} & \frac{1}{(s+1)(s+2)} \\ \frac{1}{(s+1)(s+2)} & \frac{1}{(s+2)^2} \end{bmatrix}$$

Check : $G(s) = C(sI-A)^{-1}B$

$$= \begin{bmatrix} 0 & -3 & -0.5 & 0.5 \\ 3 & -1 & 0 & -0.5 \end{bmatrix} \begin{bmatrix} \frac{1}{s+2} & \frac{1}{(s+2)^2} & & \\ & \frac{1}{s+2} & & \\ & & \frac{1}{s+1} & \frac{1}{(s+1)^2} \\ & & & \frac{1}{s+1} \end{bmatrix} \begin{bmatrix} -0.33 & 0.44 \\ 0 & 0.33 \\ -2 & -2 \\ -2 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -\frac{3}{s+2} & -\frac{0.5}{s+1} & \frac{0.5}{(s+1)^2}+\frac{0.5}{s+1} \\ \frac{3}{s+3} & \frac{3}{(s+2)^2}-\frac{1}{s+2} & 0 & -\frac{0.5}{s+1} \end{bmatrix} \begin{bmatrix} -0.33 & 0.44 \\ 0 & 0.33 \\ -2 & -2 \\ -2 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{s+1}-\frac{1}{s+1}+\frac{1}{(s+1)^2} & -\frac{1}{s+2}+\frac{1}{s+1} \\ -\frac{1}{s+2}+\frac{1}{s+1} & \frac{1.33}{s+2}+\frac{1}{(s+2)^2}-\frac{1}{s+2} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{(s+1)^2} & \frac{1}{(s+1)(s+2)} \\ \frac{1}{(s+1)(s+2)} & \frac{s+3}{(s+2)^2} \end{bmatrix}$$

```
RNH,MI=10000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITHM
    *** START INPUT OF DATA ***
 TOLERANCE
  *          *
?  .000000005
  ORDER OF MATRIX  -FORMAT 2I2.
?  2 2

 ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 3


  ORDER OF DENOMINATOR
?  4


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *         *         *         *         *         *         *         *
?      4.         4.        1.
?      2.         3.        1.
?      2.         3.        1.
?      3.         7.        5.        1.


 COEFFICIENTS OF DENOMINATOR
?      4.        12.       13.        6.        1.
    *** INPUT OF DATA FINISHED ***
    DIMENSION  4

   A MATRIX
   -2.250   -.146   -.000      0
   -3.000  -2.036    .735    .429
     .000   -.500   -.464    .313
   -8.000  -2.667   1.857  -1.250
   B MATRIX
        0       0
        0       0
    1.750       0
    1.000   4.000
   C MATRIX
    1.000    .333       0       0
    1.000    .250   -.143    .250
STOP

 CP      4.156 SECS.

RUN COMPLETE.
```

LSD

*** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM *

```
*** START INPUT OF DATA ***
TOLERANCE
 *                      *
?  .00000005
   ORDER OF MATRIX  - FORMAT 2I2
?  2 2


   ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  2 2 2 3


   ORDER OF DENOMINATOR
?  4


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
 *          *          *          *          *          *          *          *
?     4.        4.        1.
?     2.        3.        1.
?     2.        3.        1.
?     3.        7.        5.        1.

  COEFFICIENTS OF DENOMINATOR
?   4.          12.       13.       6.        1.
  *** INPUT OF DATA FINISHED ***

    OBSERVABLE REALISATION A  B  C
    A MATRIX

        0         0    -2.000        0         0         0

    1.000         0    -5.000        0         0         0

        0     1.000    -4.000        0         0         0

        0         0         0        0         0    -4.000

        0         0         0    1.000         0    -8.000

        0         0         0        0     1.000    -5.000
    B MATRIX

    2.000     1.000

    1.000     1.000

        0         0

    2.000     3.000

    1.000     4.000

        0     1.000
    C MATRIX

        0         0     1.000        0         0         0

        0         0         0        0         0     1.000
```

```
E-VALUES
-2.00000000000                0    -.99999998073              0
-1.00000001927                0   -2.00000000000              0
-2.00000000000                0   -1.00000000000              0
** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **
 DISTINCT E-VALUES  2

LEADING E-VECTORS FOR E-VALUE  1
  OF RANK  2
       0
       0
       0
   1.000
       0
  -1.000

LEADING E-VECTORS FOR E-VALUE  1
  OF RANK  1
    .500
   1.000
    .500
       0
       0
       0

LEADING E-VECTORS FOR E-VALUE  2
  OF RANK  2
       0
   1.000
    .500
       0
       0
       0

LEADING E-VECTORS FOR E-VALUE  2
  OF RANK  1
       0
       0
       0
   1.000
   1.000
    .250

TRANSFORMATION MATRIX FOR JORDAN FORM
       0        0     .500   -1.000        0        0
       0        0    1.000   -1.500    1.000        0
       0        0     .500    -.500     .500        0
   6.000    1.000        0        0        0    1.000
   9.000        0        0        0        0    1.000
   3.000   -1.000        0        0        0     .250

 INVERSE OF TRANSFORMATION MATRIX
       0        0        0    -.444     .556    -.444
       0        0        0    -.333     .667   -1.333
   2.000   -4.000    8.000        0        0        0
    .000   -2.000    4.000        0        0        0
  -2.000    2.000   -2.000        0        0        0
       0        0        0    4.000   -4.000    4.000
  TRANSFORMED B MATRIX
   -.333     .444
   -.000     .333
    .000   -2.000
  -2.000   -2.000
  -2.000        0
   4.000     .000
```

TRANSFORMED C MATRIX

0	0	.500	-.500	.500	0
3.000	-1.000	0	0	0	.250

TRANSFORMED A MATRIX IN JORDAN FORM

-2.000	1.000	0	0	0	.000
.000	-2.000	0	0	0	0
0	0	-2.000	-.000	-.000	0
0	0	0	-1.000	1.000	0
0	0	.000	.000	-1.000	0
-.000	.000	0	0	0	-1.000

TRANSFORMED B

-.333	.444
-.000	.333
.000	0
-2.000	-2.000
-2.000	0
4.000	.000

TRANSFORMED B

-.333	.444
-.000	.333
-2.000	-2.000
-2.000	0
0	.000
4.000	.000

rows 0 ; delete !

DIMENSION 4

A MATRIX

-2.000	1.000	0	0
.000	-2.000	0	0
0	0	-1.000	1.000
0	0	.000	-1.000

B MATRIX

-.333	.444
-.000	.333
-2.000	-2.000
-2.000	0

C MATRIX

0	-3.000	-.500	.500
3.000	-1.000	0	-.500

STOP
CTIME
CTIME. 10.376 SEC.

Test 10

$$G(s) = \begin{bmatrix} \frac{1}{s+1} & \frac{s}{s-2} \\ 2 & 0 \\ \frac{2}{s-2} & 1 \end{bmatrix}$$

check : G(s) = C(sI−A)B + D

$$= \begin{bmatrix} 1 & 0 & -0.5 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{s-2} & & \\ & \frac{1}{s-2} & \\ & & \frac{1}{s+1} \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 2 & 0 \\ -2 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 2 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{s-2} & 0 & \frac{-0.5}{s+1} \\ 0 & 0 & 0 \\ 0 & \frac{1}{s-2} & 0 \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 2 & 0 \\ -2 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 2 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{s+1} & \frac{2}{s-2} \\ 0 & 0 \\ \frac{2}{s-2} & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 2 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{s+1} & \frac{s}{s-2} \\ 2 & 0 \\ \frac{2}{s-2} & 1 \end{bmatrix}$$

```
RUN,MI=140000

    **** TEST RUN FOR MINIMAL REALISATION USING ROSENBROCK S ALGORITHM
    *** START INPUT OF DATA ***
 TOLERANCE
   *           *
?  .000000000005
   ORDER OF MATRIX   -FORMAT 2I2
?  3 2

  ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?  1 2 2 0 1 2


   ORDER OF DENOMINATOR
?  2


    COEFFICIENTS OF NUMERATOR POLYNOMIALS
   *        *        *        *        *        *        *        *
?    -2.       1.
?     0.       1.        1.
?    -4.      -2.        2.
?     0.
?     2.       2.
?    -2.      -1.        2.


  COEFFICIENTS OF DENOMINATOR
?    -2.      -1.        2.
    *** INPUT OF DATA FINISHED ***
    DIMENSION   4

   A MATRIX
    1.333    -.315    1.467        0
    1.200    -.133     .240    1.200
    4.000    -.444     .800     .667
        0    4.000   -1.200    2.000
   B MATRIX
        0        0
        0        0
   -6.667        0
    4.000   12.000
   C MATRIX
    1.000     .556        0     .500
        0        0        0        0
        0    1.000    -.500     .500
STOP

 CP     4.201 SECS.

RUN COMPLETE.
```

```
*** TEST RUN FOR MINIMAL REALISATION USING JORDAN CANONICAL FORM *

*** START INPUT OF DATA ***
TOLERANCE
  *                      *
?   .000000000005
   ORDER OF MATRIX  - FORMAT 2I2
?   3 2


   ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
?   1 2 2 0 1 2


   ORDER OF DENOMINATOR
?   2


     COEFFICIENTS OF NUMERATOR POLYNOMIALS
  *        *        *        *        *        *        *        *
?    -2.         1.
?     0.         1.          1.
?    -4.        -2.          2.
?     0.
?     2.         2.
?    -2.        -1.          1.


  COEFFICIENTS OF DENOMINATOR
?    -2.        -1.          1.
  *** INPUT OF DATA FINISHED ***

   OBSERVABLE REALISATION A  B  C
   A MATRIX

       0    2.000       0

   1.000    1.000       0

       0        0   2.000
   B MATRIX

  -2.000    2.000

   1.000    2.000

   2.000        0
   C MATRIX

       0    1.000       0

       0        0       0

       0        0   1.000
   D MATRIX

       0    1.000

   2.000        0

       0    1.000
  E-VALUES
   2.00000000000                 0   -1.00000000000              0
   2.00000000000                 0
  ** E-VALUES REAL , MATRICES WITH REAL ELEMENTS **
   DISTINCT E-VALUES   2
```

```
LEADING E-VECTORS FOR E-VALUE  1
  OF RANK  1
  1.000        0
  1.000        0
      0    1.000

LEADING E-VECTORS FOR E-VALUE  2
  OF RANK  1
  1.000
  -.500
      0

TRANSFORMATION MATRIX FOR JORDAN FORM
  1.000        0    1.000
  1.000        0    -.500
      0    1.000        0

INVERSE OF TRANSFORMATION MATRIX
   .333     .667        0
      0        0    1.000
   .667    -.667        0
 TRANSFORMED B MATRIX
  -.000    2.000
  2.000        0
 -2.000        0
 TRANSFORMED C MATRIX
  1.000        0    -.500
      0        0        0
      0    1.000        0

 TRANSFORMED A MATRIX      IN JORDAN FORM

  2.000  |     0 |  -.000
 --------+-------+
      0  | 2.000 |      0
 ----------------+--------
  -.000        0 | -1.000
 TRANSFORMED B
      0    2.000
  2.000        0
 -2.000        0


 DIMENSION  3


 A MATRIX
  2.000        0    -.000
      0    2.000        0
  -.000        0   -1.000


 B MATRIX
      0    2.000
  2.000        0
 -2.000        0


 C MATRIX
  1.000    -.000    -.500
      0        0        0
      0    1.000        0
STOP
TSF(I=RAE3)
<CTIME
CTIME.     73.341 SEC.
```

4.3 Conclusions and possible improvements.

Though it was anticipated that Rosenbrock's algorithm would be worse (in accuracy terms and speed) than the algorithm presented in this paper, the results showed, unfortunately, that this is not the case, at least for the examples used. It was stated that the fact that Rosenbrock's algorithm searches for linear independence through, generally more columns than this algorithm, would make him worse. But in fact it is not this part of the algorithm which fails to be better but the first part, that is the calculation of the Jordan form and in particular of its e-values. E-value routines give accuracy of 6 decimal places at the worst, but cases were found where only 2 places of accuracy were obtained. This prompted the author to use a routine which calculates the roots of a polynomial for the reasons explained in 3.4. Unfortunately this change did not bring any spectacular changes, as expected, mainly because these routines seem to be very sensitive in the variation of parameters, whereas the e-values of matrices in block companion form are not so sensitive in the variation of matrix elements. However, polynomial roots routines must be favoured in high order matrices partitioned in small matrices of low order, and abetter program could incorporate a criterion test by which it would be decided, on the grounds of the size of A and the constituent matrices, which kind of routine it is better to employ.

This suggestion would not however better the CP time and in this respect Rosenbrock's algorithm favours considerably.

This is because of the considerable time required for the calculation of the Jordan form.

As far as the actual reduction process is concerned, the use of full pivoting in the transformation of the B and C matrices would better the accuracy, though such acase was not encountered in the tests (i.e. a case where this would be necessary) as mostly whole numbers were used.

Well, this algorithm has if course a major advantage and this is the fact that the minimal realisation is in the Jordan canonical form (unless of course complex e-values occur). This is a much easier form to work with and should therefore be preferred. The algorithm should also be preferred from the one suggested in [7], as it does not require the denominator in factored form.

A small advantage, as was pointed to me by a user of both methods, is that Jordan form gives "nice" numbers if the input has "nice" numbers whereas Rosenbrock's algorithm gives "nasty" decimal numbers. I do not know of what importance this can be but some people hate too many decimal points.

A remark was also made about the form of the input and in particular about the use of a common denominator, in contrast to the real situation where an experimentally obtained transfer function matrix would have every element in a rational polynomial form and the calculation of the common denominator (i.e. in this case multiplication of all denominators) would be a very tedious job. A better program should enable the user to choose the form which is most suited to his application.

Appendix I

```
      PROGRAM MINREA(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C
C
C        PROGRAMMER  :  A. POULIEZOS
C              DATE  :  SEPTEMBER 1976
C           MSC PROJECT FOR CONTROL DEPT.
C
C
C.....TO CALCULATE THE MINIMAL REALISATION OF A TRANSFER FUNCTION M
C.....USING THE JORDAN CANONICAL FORM
C ... THIS PROGRAM USES THE MRE SUBROUTINE FROM THE MOS PACKAGE TO
C.... INITIAL OBSERVABLE REALISATION
      INTEGER RDR,PRT,DSK,DTP
      REAL KEPL1,KEPL2,KEPL3,KEPL4
      REAL IMZ
      COMMON NRET,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JDIM(20),KI
      COMMON JSIGN,MULT(20),NR(20),WI(20),VR(20,20),VI(20,20),NDIAG(20)
      DIMENSION REZ(20),IMZ(20),COEF(20)
      DIMENSION ND(10,10)
      NRET=31
      RDR=5
      PTR=6
      WRITE (6,80)
   80 FORMAT(1H ,'    ****TEST RUN FOR MINIMAL REALISATION USING THE
     +CANONICAL FORM ****',//,'  *** START INPUT OF DATA ***')
C.... ..............................................................
C.....INPUT OF G(S) ................................................
C ....  ............................................................
C.... ORDER OF G(S),ORDER OF ACCEPTABLE ERROR
      WRITE(6,98)
   98 FORMAT(1H ,'TOLERANCE ',/,2X,'*',13X,'*')
      READ(5,97) EPS
   97 FORMAT(1X,F13.0)
      WRITE (6,99)
   99 FORMAT(1H ,'  ORDER OF MATRIX  -FORMAT 2I2 ')
      READ(5,100)N,M
  100 FORMAT(2I2)
C.... ORDER OF NUMERATOR POLYNOMIALS
      WRITE(6,101)
  101 FORMAT(1H ,/,2X,'ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW')
      READ(5,102)((NN(I,J),J=1,M),I=1,N)
  102 FORMAT(40I2)
C ... ORDER OF DENOMINATOR
      WRITE(6,103)
  103 FORMAT(1H ,//,3X,'ORDER OF DENOMINATOR')
      READ(5,104)NDEN
  104 FORMAT(I2)
C.... COEFFICIENTS OF NUMERATOR POLYNOMIALS
      WRITE (6,105)
  105 FORMAT(1H ,//,4X,'COEFFICIENTS OF NUMERATOR POLYNOMIALS')
      WRITE (6,107)
  107 FORMAT(1H ,1X,8('*',8X))
      DO 800 I=1,N
      DO 801 J=1,M
      IJ=NN(I,J)+1
      READ(5,106)(GN(I,J,K),K=1,IJ)
  801 CONTINUE
  800 CONTINUE
  106 FORMAT(8(1X,F8.0))
C.... COEFFICIENTS OF DENOMINATOR
      WRITE (6,108)
  108 FORMAT(1H ,//,2X,'COEFFICIENTS OF DENOMINATOR')
      READ(5,106)(CD(I),I=1,NDEN+1)
      WRITE(6,81)
   81 FORMAT(1H ,'    *** INPUT OF DATA FINISHED ***')
C.... ..............................................................
C.... INPUT OF DATA FINISHED
C....
C.....CALL MRE SUBROUTINE TO CALCULATE INITIAL OBSERVABLE REALISATI
      CALL MRE(ND)
      IF(NRET.EQ.1)STOP
      IFIN=NSUM
      WRITE (6,1007)
 1007 FORMAT(1H ,/,3X,'OBSERVABLE REALISATION A   B   C')
      WRITE (6,725)
  725 FORMAT(1H ,'  A MATRIX')
      DO 1001 I=1,IFIN
      WRITE (6,130)(A(I,J),J=1,IFIN)
 1001 CONTINUE
      WRITE (6,755)
  755 FORMAT(1H ,'  B MATRIX')
      DO 1002 I=1,IFIN
```

```
      WRITE(6,130)(B(I,J),J=1,M)
 1002 CONTINUE
      WRITE(6,784)
  784 FORMAT(1H ,'   C MATRIX')
      DO 1003 I=1,N
      WRITE(6,130)(C(I,J),J=1,IFIN)
 1003 CONTINUE
 1/3 / 1/ 3 CONTINUE
      DO 20 I=1,N
      DO 21 J=1,M
      IF(D(I,J).LE.EPS)GO TO 21
      GO TO 22
   21 CONTINUE
   20 CONTINUE
      GO TO 23
   22 WRITE(6,25)
   25 FORMAT(1H ,'   D MATRIX')
      DO 26 I=1,N
      WRITE(6,130)(D(I,J),J=1,M)
   26 CONTINUE
  130 FORMAT(1H ,/,10(1X,F7.3))
C.... FIND E-VALUES OF A MATRIX BY SUCCESIVELY CALLING LIB. SUB. C0
C.... TO CALCULATE THE ROOTS OF THE CHARACTERISTIC POLYNOMIALS OF T
C.... CANONICAL BLOCKS
   23 TOL=10.**(-15)
      IFIN=0
      DO 1 I=1,N
      IF(ND(I,1).EQ.0)GO TO 1
      IST=IFIN+1
      IFIN=IFIN+ND(I,1)
      IF(ND(I,1).GT.1)GO TO 8
      REZ(1)=A(IST,IFIN)
      IMZ(1)=0.
      GO TO 3
    8 II=ND(I,1)+1
      COEF(1)=1.
      IZERO=0
      DO 2 J=2,II
      COEF(J)=-A(IFIN-J+2,IFIN)
      IF(ABS(COEF(J)).LE.EPS)GO TO 2
      IZERO=1
    2 CONTINUE
      IF(IZERO)4,4,6
    4 DO 9 J=1,II-1
      REZ(J)=0.
    9 IMZ(J)=0.
      GO TO 3
    6 IFAIL=1
    5 CALL C02AEF(COEF,II,REZ,IMZ,TOL,IFAIL)
      IF(IFAIL-1)3,999,5
  999 WRITE(6,166)
  166 FORMAT(1H ,' * E-VALUES NOT FOUND *')
      STOP
    3 DO 7 J=IST,IFIN
      WR(J)=REZ(J-IST+1)
    7 WI(J)=IMZ(J-IST+1)
    1 CONTINUE
      IFIN=NSUM
      WRITE(6,1005)(WR(I),WI(I),I=1,IFIN)
 1005 FORMAT(1H ,' E-VALUES',/,4(1X,F15.11))
C.... ...........................................................
C.... CHECK FOR COMPLEX E-VALUES AND CALL APPROPRIATE ROUTINE.....
C.... ...........................................................
      DO 400 I=1,IFIN
      IF(ABS(WI(I)).LE.EPS)GO TO 400
      ICOMPL=1
  400 MULT(I)=1
      IF(ICOMPL)401,401,402
  401 CONTINUE
      GO TO 403
  402 CALL CCOMPL
  403 STOP
      END
C.... ...........................................................
C
C
C
C
C          SUBROUTINE CCOMPL
C          *****************
      SUBROUTINE CCOMPL
      INTEGER RDR,PRT,DSK,DTP
      REAL KEPL1,KEPL2,KEPL3,KEPL4,KEPV1,KEPV2,KEPV3,KEPV4
      COMPLEX AA,BB,CC
      COMMON NWIT,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
```

```
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JDIM(20),
      COMMON JSIGN,MULT(20),WR(20),WI(20),VR(20,20),VI(20,20),NDIAG(20
      DIMENSION AA(20,20),BB(20,10),CC(10,20)
      EQUIVALENCE (AA(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(1
      WRITE(6,1073)
 1073 FORMAT(1H ,' COMPLEX E-VALUES , MATRICES WITH COMPLEX ELEME
      IFIN=NSUM
      K=0
      K3=0
      IC=0
      DO 501 I=1,IFIN-1
      IF(MULT(I))501,501,402
  402 K=K+1
      K3=K3+1
      DO 601 J=I+1,IFIN
      IF(ABS(WR(I)-WR(J)).GT.EPS)GO TO 601
      IF(ABS(WI(I)-WI(J)).GT.EPS)GO TO 601
      MULT(I)=MULT(I)+1
      MULT(K+1)=0
      IC=1
      IF(J-I-1)666,666,610
  610 KEPL1=WR(K+1)
      KEPL2=WI(K+1)
      WR(K+1)=WR(J)
      WI(K+1)=WI(J)
      WR(J)=KEPL1
      WI(J)=KEPL2
  666 K=K+1
  601 CONTINUE
  501 CONTINUE
      IF(MULT(IFIN).GT.0)K3=K3+1
      WRITE(6,222)K3
  222 FORMAT(1H ,'  DISTINCT E-VALUES ',I2)
  543 FORMAT(1H ,' TRANSFORMATION MATRIX FOR JORDAN FORM ')
      DO 605 I=1,IFIN-1
      KO=I
      IF(MULT(I).EQ.0)GO TO 605
      IF(ABS(WI(I)).LE.EPS)GO TO 605
      DO 606 J=I+1,IFIN
      IF(MULT(J).EQ.0)GO TO 605
      KOO=J
      IF(ABS(WR(KO)-WR(KOO))-EPS)607,607,606
  607 IF(ABS(WI(KO)+WI(KOO))-EPS)608,608,606
  608 IF(J-I-MULT(I))606,606,609
  609 DO 611  I5=1,MULT(J)
      IE1=KO+I5-1+MULT(I)
      IE2=KOO+I5-1
      KEPM1=MULT(IE1)
      KEPL1=WR(IE1)
      KEPL2=WI(IE1)
      WR(IE1)=WR(IE2)
      WI(IE1)=WI(IE2)
      MULT(IE1)=MULT(IE2)
      DO 613 I6=IE1+1,IE2
      KEPM3=MULT(I6)
      KEPL3=WR(I6)
      KEPL4=WI(I6)
      WR(I6)=KEPL1
      WI(I6)=KEPL2
      MULT(I6)=KEPM1
      KEPM1=KEPM3
      KEPL1=KEPL3
  613 KEPL2=KEPL4
  611 CONTINUE
  606 CONTINUE
  605 CONTINUE
      CALL CJORDA(K3,IRET)
      IF(IRET)1006,1006,2000
 1006 WRITE(6,543)
  543 FORMAT(1H ,' TRANSFORMATION MATRIX FOR JORDAN FORM')
      DO 602 I=1,IFIN
      WRITE(6,500)(VR(I,J),VI(I,J),J=1,IFIN)
  500 FORMAT(1H ,10(1X,F7.3))
  602 CONTINUE
      CALL CTRANS(K3)
      IF(IRET.EQ.1)GO TO 2000
      CALL CSWRIT
 2000 RETURN
      END
C.......................................................................
C
C
C
```

```
C      SUBROUTINE CTRANS
C      ****************
      SUBROUTINE CTRANS(K3)
      INTEGER RDR,PRT,DSK,DTP
      COMPLEX AT,AA,BB,CC,TT,TTT,QUOT
      COMPLEX AT1,AT2
      COMMON NRIT,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JDIM(20),KI
      COMMON JSIGN,MULT(20),WR(20),WI(20),VR(20,20),VI(20,20),NDIAG(20)
      DIMENSION TTT(20,20),TT(20,20),IV(20)
      DIMENSION RIND(20),AT(20,20),AT1(20,10),AT2(10,20)
      DIMENSION AA(20,20),BB(20,10),CC(10,20)
      EQUIVALENCE (AA(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(1,1
      EQUIVALENCE (AT(1,1),AT1(1,1)),(AT(1,11),AT2(1,1))
C.....TRANSFORM B MATRIX
      IFIN=NSUM
      DO 222 I=1,IFIN
      DO 223 J=1,IFIN
      TT(I,J)=CMPLX(VR(I,J),VI(I,J))
  223 AT(I,J)=TT(I,J)
  222 CONTINUE
      DO 229 I=1,IFIN
      DO 225 JJ=1,M
  225 BB(I,JJ)=CMPLX(B(I,JJ),0.)
  229 CONTINUE
C.....INVERT TRANSFORMATION MATRIX
      IDIM1=20
      IFAIL=1
      CALL F03AHF(IFIN,AT,IDIM1,D1,D2,ID1,RIND,IFAIL)
      IF(IFAIL)398,398,999
  398 DO 397 I=1,IFIN
      DO 396 J=1,IFIN
      IF(I.EQ.J)GO TO 395
      TTT(I,J)=CMPLX(0.,0.)
      GO TO 396
  395 TTT(I,J)=CMPLX(1.,0.)
  396 CONTINUE
  397 CONTINUE
      CALL F04AKF(IFIN,IFIN,AT,IDIM1,RIND,TTT,IDIM1)
      WRITE (6,1024)
      GO TO 399
  999 WRITE(6,1020)
 1020 FORMAT(1H ,2X,'FAILURE IN F03AHF')
      IRET=1
      GO TO 51
  399 DO 1022 IL=1,IFIN
      WRITE(6,1023)(TTT(IL,JL),JL=1,IFIN)
 1022 CONTINUE
 1023 FORMAT(1H ,10(1X,F7.3))
 1024 FORMAT(1H ,/,2X,'INVERSE OF TRANSFORMATION MATRIX')
      IID1=20
      IID2=10
      CALL CMATRM(IID1,IID2,IFIN,IFIN,M,TTT,BB,AT1)
      WRITE(6,246)
  246 FORMAT(1H ,/,2X,'TRANSFORMED B MATRIX')
      DO 71 I=1,IFIN
      DO 73 J=1,M
   73 BB(I,J)=AT1(I,J)
      WRITE (6,1025)(BB(I,J),J=1,M)
   71 CONTINUE
 1025 FORMAT(1H ,/,10(1X,F7.3))
C.....TRANSFORM C
      DO 226 I=1,N
      DO 227 J=1,IFIN
  227 CC(I,J)=CMPLX(C(I,J),0.)
  226 CONTINUE
      CALL CMATRM(IID2,IID1,N,IFIN,IFIN,CC,TT,AT2)
      WRITE (6,254)
  254 FORMAT(1H ,/,2X,'TRANSFORMED C MATRIX')
      DO 72 I=1,N
      DO 74 J=1,IFIN
   74 CC(I,J)=AT2(I,J)
      WRITE(6,1025)(CC(I,J),J=1,IFIN)
   72 CONTINUE
  420 IID1=20
      IID2=20
      DO 200 I=1,IFIN
      DO 201 J=1,IFIN
  201 AA(I,J)=CMPLX(A(I,J),0.)
  200 CONTINUE
      CALL CMATRM(IID1,IID2,IFIN,IFIN,IFIN,TTT,AA,AT)
      CALL CMATRM(IID1,IID2,IFIN,IFIN,IFIN,AT,TT,AA)
      WRITE (6,1035)
```

```
      DO 830 I=1,IFIN
      WRITE(6,1333)(AA(I,J),J=1,IFIN)
 1335 FORMAT(1H ,/,'     TRANSFORMED A MATRIX     IN JORDAN FORM')
 1333 FORMAT(1H ,/,10(1X,F7.3))
  830 CONTINUE
      NDIM=IFIN
  615 IF(IC)416,416,432
C ... DISTINCT E-VALUES JORDAN FORM DIAGONAL
C.... IRRR IS NUMBER OF DISTINCT E-VALUES I.E. NUMBER OF SUBSYSTEM
C.....NDIAG IS NUMBER OF JORDAN BLOCKS PER SUBSYSTEM
C ... JDIM IS DIMENSION OF EACH JORDAN BLOCK
C.....KINDEX(I) IS POSITION , IN AA , OF BLOCK I .
  416 IRRR=IFIN
      NDIM=IFIN
      KBLOC=IFIN
      DO 250 IJ=1,KBLOC
      NDIAG(IJ)=1
  250 JDIM(IJ)=1
      GO TO 431
  432 IRRR=K3
  431 KKK=1
      KINDEX(1)=1
      DO 302 I=1,IRRR
      DO 303 I2=1,NDIAG(I)
      KINDEX(KKK+I2)=KINDEX(KKK+I2-1)+JDIM(KKK+I2-1)
  303 CONTINUE
      KKK=KKK+NDIAG(I)
  302 CONTINUE
      JS=1
      LSTART=1
      ISKIP=0
      MFORA=0
      DO 50 JJ=1,IRRR
      IF(ISKIP)184,184,183
C.... STEP 1,SET  V(I)= ,I=1,R
  556 IREP=0
  184 LFINI=LSTART+NDIAG(JJ)-1
      DO 55 J=LSTART,LFINI
   55 IV(J)=0
C.....STEPS 2 AND 3
C.... KIMAX IS MAXIMUM SIZE OF BLOCK FOR WHICH IV=0
   54 KIMAX=1
      KSTOP=0
      DO 56 J=LSTART ,LFINI
      IF(IV(J))56,57,56
   57 KSTOP=1
      IF(KIMAX-JDIM(J))58,58,56
   58 KIMAX=JDIM(J)
C.... JSIGN IS J ST. JDIM(J) IS MAX
      JSIGN =J
   56 CONTINUE
      IF(KSTOP)555,555,59
C.....STEP 4
   59 KTSIGN=KINDEX(JSIGN)+JDIM(JSIGN)-1
      DO 61 I=1,M
      IF(CABS(BB(KISIGN,I))-EPS)61,61,62
   61 CONTINUE
      GO TO 91
   62 IV(JSIGN)=1
      ISG=I
      ISG1=I+LSTART-1
      DO 63 I=LSTART,LFINI
      IF(IV(I))63,64,63
   64 INDB=KINDEX(I)+JDIM(I)-1
      INDBB=KINDEX(JSIGN)+JDIM(JSIGN)-1
      QUOT=BB(INDB,ISG)/BB(INDBB,ISG)
  821 FORMAT(1H ,10(1X,F7.3))
      DO 65 II=1,JDIM(I)
      DO 66 IB=1,M
      INI1=KINDEX(I)+II-1
      INI2=KINDEX(JSIGN)+JDIM(JSIGN)-JDIM(I)+II -1
      BB(INI1,IB)=BB(INI1,IB)-QUOT*BB(INI2,IB)
   66 CONTINUE
      DO 67 ICC=1,N
   67 CC(ICC,INI2)=CC(ICC,INI2)+QUOT*CC(ICC,INI1)
   65 CONTINUE
      WRITE(6,819)
  819 FORMAT(1H ,' * TRANSFORMED B MATRIX * ')
      DO 822 IN=1,IFIN
      WRITE(6,821)(BB(IN,J),J=1,M)
  822 CONTINUE
   63 CONTINUE
      GO TO 54
C.... IDEL IS INDEX FOR ROWS AND COLUMNS TO BE DELETED
   91 IF(JDIM(JSIGN).EQ.1)GO TO 69
```

```
      IREP=1
      GO TO 669
   69 IV(JSIGN)=1
      LFINI=LFINI-1
  669 CALL CCONVE(IV)
      IF(ABS(WI(JJ)).LE.EPS)GO TO 54
   57 JSIGN=JSIGN+MULT(JJ)-MFORA
      MFORA=MFORA+1
      CALL CCONVE(IV)
      ISKIP=1
      GO TO 54
  183 ISKIP=0
      MFORA=0
  555 IF(IREP)557,557,556
  557 LSTART=LFINI+1
   50 CONTINUE
   51 RETURN
      END
C.....................................................................
C
C
C
C
C           SUBROUTINE CSWRIT
C           ****************
      SUBROUTINE CSWRIT
      INTEGER RDR,PRT,DSK,DTP
      COMPLEX AA,BB,CC
      COMMON NWIT,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE ,N,M,NN(10,10),NDIN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AAA(2 ,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL
      DIMENSION AA(2 ,20),BB(20,10),CC(10,20)
      EQUIVALENCE (AA(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(1,
C.....................................................................
      WRITE(6,129)NDIM
  129 FORMAT(1H ,/,' DIMENSION ',I2)
      WRITE(6,379)
  379 FORMAT(1H ,'*** MINIMAL REALISATION ***',/,'*** MATRICES IN R
     +AFTER TRANSFORMATION ***',//,'  A MATRIX')
  130 FORMAT(1H ,10(1X,F7.3))
    5 KOUNT=0
      IFIN=NDIM
      DO 26 I=1,IFIN
      DO 27 J=1,IFIN
   27 A(I,J)=REAL(AA(I,J))
      DO 1 J1=1,M
    1 B(I,J1)=REAL(BB(I,J1))
      DO 2 J2=1,N
    2 C(J2,I)=REAL(CC(J2,I))
   26 CONTINUE
      I=1
   25 L=I
   15 IF(ABS(AIMAG(AA(I,I))+AIMAG(AA(L+1,L+1))).GT.EPS)GO TO 11
      IF(ABS(REAL(AA(I,I))-REAL(AA(L+1,L+1))).GT.EPS)GO TO 11
      L=L+1
      KOUNT=KOUNT+1
      IF((L+1).GT.NDIM)GO TO 11
      GO TO 15
   11 IF(KOUNT)10,10,13
   13 DO 14 J=I-KOUNT+1,I
      DO 30 JJ=I-KOUNT+1,I
      A(J,JJ)=REAL(AA(J,JJ))
   30 CONTINUE
      DO 20 I1=1,M
   20 B(J,I1)=AIMAG(BB(J,I1))
      DO 21 I2=1,N
   21 C(I2,J)=2.*AIMAG(CC(I2,J))
   14 CONTINUE
      DO 16 J=I+1,I+KOUNT
      DO 17 JJ=I+1,I+KOUNT
   17 A(J,JJ)=REAL(AA(J,JJ))
      DO 23 I1=1,M
   23 B(J,I1)=REAL(BB(J,I1))
      DO 24 I2=1,N
   24 C(I2,J)=2.*REAL(CC(I2,J))
   16 CONTINUE
      DO 18 J=I+1,I+KOUNT
      DO 19 JJ=I-KOUNT+1,I
      A(J,JJ)=-AIMAG(AA(J-KOUNT,JJ))
   19 A(JJ,J)=AIMAG(AA(JJ,J-KOUNT))
   18 CONTINUE
      I=I+KOUNT+1
      KOUNT=0
      GO TO 22
```

```
10 I=I+1
22 IF(I.LT.NDIM)GO TO 25
DO 40 I=1,IFIN
WRITE(6,100)(A(I,J),J=1,IFIN)
40 CONTINUE
WRITE(6,855)
855 FORMAT(1H ,'  B MATRIX')
DO 41 I=1,IFIN
WRITE(6,100)(B(I,J),J=1,M)
41 CONTINUE
WRITE(6,856)
856 FORMAT(1H ,'  C MATRIX')
DO 42 I=1,N
WRITE(6,100)(C(I,J),J=1,IFIN)
42 CONTINUE
200 RETURN
END
C..................................................................
C
C
C
C      SUBROUTINE CCONVE
C      *****************
SUBROUTINE CCONVE(ILS)
C.....TO DELETE ROWS AND COLUMNS FROM AA,BB,CC,
INTEGER RDR,PRT,DSK,DTP
COMPLEX AA,BB,CC
COMMON N,IT,RDR,PRT,DSK,DTP,LPD,PAR(4),EPS,NDUM(8)
COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JDIM(20),K
COMMON JSIGN,MULT(20)
DIMENSION ILS(20)
DIMENSION AA(20,20),BB(20,10),CC(10,20)
EQUIVALENCE (AA(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(1,
IST=KINDEX(JSIGN)+JDIM(JSIGN)-1
IF(IST.GE.NDIM)GO TO 11
DO 1 I=IST,NDIM-1
DO 12 J=1,NDIM
12 AA(I,J)=AA(I+1,J)
DO 2 J=1,M
BB(I,J)=BB(I+1,J)
2 CONTINUE
DO 3 IJ=1,N
3 CC(IJ,I)=CC(IJ,I+1)
1 CONTINUE
DO 13 J=IST,NDIM
DO 14 I=1,NDIM
14 AA(I,J)=AA(I,J+1)
13 CONTINUE
11 NDIM=NDIM-1
JDIM(JSIGN)=JDIM(JSIGN)-1
IL=0
IF(JDIM(JSIGN))7,7,8
7 DO 9 I=JSIGN,19
ILS(I)=ILS(I+1)
9 JDIM(I)=JDIM(I+1)
IL=1
8 DO 6 I=JSIGN+1,19
6 KINDEX(I)=KINDEX(I+IL)-1
RETURN
END
C..................................................................
C
C
C
C
C.....SUBROUTINE GCOM
C     ****************
SUBROUTINE GCOM(A,IA,GCD,IGCD,EPS)
C
C      SUBROUTINE TO CALCULATE THE GREATEST COMMON DIVISOR OF 2 PO
C      STORED IN ARRAY A.  THIS IS DONE BY SUCCESSIVELY SUBTRACTIN
C      MULTIPLES OF THE POLYNOMIAL OF LEAST DEGREE AT ANY ONE TIME
C      FROM THE OTHER UNTIL ONE NON-ZERO POLYNOMIAL REMAINS.  THIS
C      IS THE GCD OF THE SET AND IS MADE MONIC BEFORE EXIT.
C
DIMENSION A(2,21),GCD(21)
DIMENSION IA(2)
C
IS=0
IF(IA(1).EQ.0.OR.IA(2).EQ.0) GO TO 500
902 IF(ABS(A(1,1)).GT.EPS.AND.ABS(A(2,1)).GT.EPS) GO TO 500
IF(ABS(A(1,1)).LE.EPS.AND.ABS(A(2,1)).LE.EPS) GO TO 903
```

```
      IF(ABS(A(1,1)).GT.EPS) GO TO 904
      IA1=IA(1)+1
      DO 905 K=2,IA1
      K1=K-1
      A(1,K1)=A(1,K)
  905 CONTINUE
      A(1,IA1)=0.0
      IA(1)=IA(1)-1
      GO TO 50
  904 IA2=IA(2)+1
      DO 906 K=2,IA2
      K1=K-1
      A(2,K1)=A(2,K)
  906 CONTINUE
      A(2,IA2)=0.0
      IA(2)=IA(2)-1
      GO TO 50
  903 IS=IS+1
      IA1=IA(1)+1
      DO 900 K=2,IA1
      K1=K-1
      A(1,K1)=A(1,K)
  900 CONTINUE
      A(1,IA1)=0.0
      IA(1)=IA(1)-1
      IA2=IA(2)+1
      DO 901 K=2,IA2
      K1=K-1
      A(2,K1)=A(2,K)
  901 CONTINUE
      A(2,IA2)=0.0
      IA(2)=IA(2)-1
      GO TO 902
   50 LL=1
      IF(IA(2).LT.IA(1)) LL=2
      NN=3-LL
      IA1=IA(NN)+1
      IA2=IA(LL)+1
      IF(IA(LL).EQ.0.AND.ABS(A(LL,1)).LT.EPS) GO TO 100
      IF(IA(LL).EQ.0)GO TO 200
      IDEG=IA(NN)-IA(LL)
      RATIO=A(NN,IA1)/A(LL,IA2)
      DO 300 K=1,IA2
      KI=K+IDEG
      A(NN,KI)=A(NN,KI)-RATIO*A(LL,K)
      IF(ABS(A(NN,KI)).LT.EPS) A(NN,KI)=0.0
  300 CONTINUE
      DO 400 KK=1,IA1
      K=IA(NN)+2-KK
      IF(ABS(A(NN,K)).LT.EPS) GO TO 400
      IA(NN)=K-1
      GO TO 410
  400 CONTINUE
      IA(NN)=0
  410 CONTINUE
      GO TO 500
  100 DO 600 K=1,IA1
      GCD(K)=A(NN,K)/A(NN,IA1)
      A(NN,K)=0.0
  600 CONTINUE
      IGCD=IA(NN)
      GO TO 800
  200 DO 700 I=1,2
      IA3=IA(I)+1
      DO 700 K=1,IA3
  700 A(I,K)=0.0
      IGCD=0
  800 IF(IGCD.EQ.0) GCD(1)=1.0
      IF(IS.EQ.0) GO TO 801
      IA1=IGCD+1
      DO 802 K=1,IA1
      K1=K+IS
      GCD(K1)=GCD(K)
  802 CONTINUE
      DO 803 K=1,IS
  803 GCD(K)=0.0
      IGCD=IGCD+IS
  801 RETURN
      END
C     ..................................................................
C
C
C
C
C           SUBROUTINE PNORM
```

```
C                 ****************
      SUBROUTINE PPNORM(X,IDIMX,EPS)
C        PURPOSE
C           NORMALIZE COEFFICIENT VECTOR OF A POLYNOMIAL
C
C        USAGE
C           CALL PNORM(X,IDIMX,EPS)
C
C        DESCRIPTION OF PARAMETERS
C           X       - VECTOR OF ORIGINAL COEFFICIENTS, ORDERED FRO
C                     SMALLEST TO LARGEST POWER. IT REMAINS UNCHAN
C           IDIMX   - DIMENSION OF X.  IT IS REPLACED BY FINAL DIME
C           EPS     - TOLERANCE BELOW WHICH COEFFICIENT IS ELIMINA
C
C        REMARKS
C           IF ALL COEFFICIENTS ARE LESS THAN EPS, RESULT IS A ZE
C           POLYNOMIAL WITH IDIMX=0 BUT VECTOR X REMAINS INTACT
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           DIMENSION OF VECTOR X IS REDUCED BY ONE FOR EACH TRAI
C           COEFFICIENT WITH AN ABSOLUTE VALUE LESS THAN OR EQUAL
C
C     ..................................................................
C
      DIMENSION X(21)
C
    1 IF(IDIMX) 4,4,2
    2 IF(ABS(X(IDIMX))-EPS) 3,3,4
    3 IDIMX=IDIMX-1
      GO TO 1
    4 RETURN
      END
C.....................................................................
C
C
C
C
C.....  SUBROUTINE COMDEN
C       ****************
      SUBROUTINE COMDEN(A,IA,B,IB,M,N,LCM,ILCM,EPS)
C
C        SUBROUTINE TO FIND THE COMMON DENOMINATOR OF A RATIONAL
C        FUNCTION MATRIX.  THE NUMERATOR POLYNOMIALS A ARE RETURNED
C        THE COMMON DENOMINATOR LCM.
C        WHEN CALLED BY MRE,THE COMMON DENOMINATORD OF EACH ROW
C        ARE FOUND,AND STORED IN B(I,1),FOR EACH I
C
C        PROGRAMMER:        P.KATZBERG    7/73
C        MOD1:A.D.F.  12/11/74
C
C        SUBROUTINES CALLED: GCOM
C
C        NRET = 31 IFF CALLING PROGRAM IS MRE
C
      INTEGER RDR,PRT
      REAL LCM
      DIMENSION A(10,10,21),B(10,10,21),LCM(21),TEMP(21),POLY(2,21),G
      DIMENSION IA(10,10),IB(10,10),IPOLY(2)
      COMMON NRET,RDR,PRT
C
C
C        CALCULATE COMMON DENOMINATOR
C
C
      I=1
  701 IB1=IB(I,1)+1
C
C   SET POLY(1) = LCM = B(I,1)
      DO 100 K=1,IB1
      LCM(K)=B(I,1,K)
      POLY(1,K)=LCM(K)
  100 CONTINUE
C
      ILCM1 = IB(I,1)+1
      IPOLY(1) = IB(I,1)
      JST = 1
      IF(NRET.NE.31) GO TO 702
      IF(N.LT.2)GO TO 201
      JST = 2
C
C   RANGE OVER ALL COLUMNS
  702 DO 200 J=JST,N
```

```
      IF(NRET.EQ.31) GO TO 703
      IST=2
      IF(J.GT.1) IST=1
      IF(IST.GT.M) GO TO 200
C
C     RANGE OVER ALL ROWS
      DO 250 I=IST,M
C
C     SET POLY(2) = TEMP = B(I,J)
  703 IB1=IB(I,J)+1
      DO 210 K=1,IB1
      TEMP(K)=B(I,J,K)
      POLY(2,K)=TEMP(K)
  210 CONTINUE
C
C     CALCULATE GCD OF POLY(1) AND POLY(2)
      IPOLY(2)=IB(I,J)
      CALL GCDM(POLY,IPOLY,GCD,IGCD,EPS)
      IDEG1=IB1-IGCD
      IGCD1=IGCD+1
      IF(IGCD.EQ.0) GO TO 222
      IF((IDEG1+IGCD1).GT.22) GO TO 600
C
C     DIVIDE TEMP BY GCD,BY SUCCESSIVE SUBTRACTION
      L=IDEG1
  220 LL=L+IGCD
      RATIO=TEMP(LL)/GCD(IGCD1)
C
C     SUBTRACT C(L)*(S**L)*GCD FROM TEMP
      DO 221 K=1,IGCD1
      KL1=K+L-1
      TEMP(KL1)=TEMP(KL1)-RATIO*GCD(K)
  221 CONTINUE
C
C     STORE C(L) IN TEMP(LL)
      IF(ABS(RATIO).LT.EPS) RATIO= .0
      TEMP(LL)=RATIO
      L=L-1
      IF(L.GT.0) GO TO 220
C
C     SHIFT TO TOP OF ARRAY
      L=IGCD
      DO 223 K=1,IDEG1
      L=L+1
      TEMP(K)=TEMP(L)
  223 CONTINUE
C
C     SET POLY(1) = LCM*TEMP
  222 IF((ILCM1+IDEG1).GT.22) GO TO 600
      KLIM = ILCM1 + IDEG1 - 1
      DO 224 K=1,KLIM
  224 POLY(1,K) = .0
      DO 230 K=1,ILCM1
      DO 230 L=1,IDEG1
      KL1=K+L-1
      POLY(1,KL1)=POLY(1,KL1)+LCM(K)*TEMP(L)
  230 CONTINUE
C
C     SET LCM = POLY(1)
      IPOLY(1)=ILCM1+IDEG1-2
      IPOLY1=IPOLY(1)+1
      DO 240 K=1,IPOLY1
      IF(ABS(POLY(1,K)).LT.EPS) POLY(1,K)= .0
      LCM(K)=POLY(1,K)
  240 CONTINUE
C
      ILCM1=IPOLY1
      IF(ILCM1.GT.21) GO TO 600
      IF(NRET.EQ.31) GO TO 200
  250 CONTINUE
C     END OF ROW RANGE
C
  200 CONTINUE
  201 CONTINUE
C     END OF COLUMN RANGE.
C
C
C           CALCULATE NUMERATORS
C
C
C     RANGE OVER COLUMNS
      DO 450 J=1,N
      IF(NRET.EQ.31) GO TO 704
C
C     RANGE OVER ROWS
```

```
      DO 400 I=1,M
C     SET TEMP = LCM
  704 DO 410 K=1,ILCM1
      TEMP(K)=LCM(K)
  410 CONTINUE
C
      IB1=IB(I,J)+1
      IDEG1=ILCM1-IB(I,J)
      IF((IDEG1+IB1).GT.22) GO TO 600
      L=IDEG1
      IB2=IB1-1
C
C      DIVIDE TEMP BY B(I,J)
  420 LL=L+IB2
      RATIO=TEMP(LL)/B(I,J,IB1)
      DO 421 K=1,IB1
      KL1=K+L-1
      TEMP(KL1)=TEMP(KL1)-RATIO*B(I,J,K)
  421 CONTINUE
      IF(ABS(RATIO).LT.EPS) RATIO=0.0
      TEMP(LL)=RATIO
      L=L-1
      IF(L.GT.0) GO TO 420
C
C      SHIFT TO TOP OF ARRAY
      L=IB2
      DO 422 K=1,IDEG1
      L=L+1
      TEMP(K)=TEMP(L)
  422 CONTINUE
C
      IA1=IA(I,J)+1
      IF(IA1.EQ.1.AND.ABS(A(I,J,1)).LT.EPS) GO TO 460
      IF((IDEG1+IA1).GT.22) GO TO 600
C
      KLIM = IDEG1 + IA1 - 1
      DO 424 K=1,KLIM
  424 POLY(2,K) = 0.0
C
C      SET POLY(2) = TEMP * A(I,J)
      DO 430 K=1,IDEG1
      DO 430 L=1,IA1
      KL1=K+L-1
      POLY(2,KL1)=POLY(2,KL1)+TEMP(K)*A(I,J,L)
  430 CONTINUE
C
C      SET A(I,J) = POLY(2)
      IA(I,J)=IA(I,J)+IDEG1-1
      IA1=IA(I,J)+1
      DO 440 K=1,IA1
      IF(ABS(POLY(2,K)).LT.EPS) POLY(2,K)=0.0
      A(I,J,K)=POLY(2,K)
  440 CONTINUE
C
  460 IF(NRET.EQ.31) GO TO 450
  400 CONTINUE
C      END OF ROW RANGE
C
  450 CONTINUE
C      END OF COLUMN RANGE
C
C      SET B = 0
      DO 500 J=1,N
      IF(NRET.EQ.31) GO TO 705
      DO 510 I=1,M
  705 IB1=IB(I,J)+1
      DO 511 K=1,IB1
      B(I,J,K)=0.0
  511 CONTINUE
      IB(I,J)=0
      IF(NRET.EQ.31) GO TO 706
  510 CONTINUE
C
  706 ILCM = ILCM1 - 1
      IF(NRET.NE.31) GO TO 803
  520 IB(I,J)=ILCM1-1
      DO 707 K=1,ILCM1
  707 B(I,J,K)=LCM(K)
      IF(I.EQ.M) GO TO 803
      I = I+1
      GO TO 701
C
C           NORMALISE NUMERATOR POLYNOMIALS
C
  803 DO 800 I=1,M
```

```
      DO 800 J=1,N
      K=IA(I,J)+1
  802 IF(ABS(A(I,J,K)).GT.EPS.OR.K.EQ.1) GO TO 801
      K=K-1
      GO TO 802
  801 IA(I,J)=K-1
  800 CONTINUE
      NRET=2
      GO TO 610
  611 WRITE(6,620)
  620 FORMAT(23H STORAGE LIMIT EXCEEDED)
      NRET=1
  610 RETURN
      END
C.....................................................................
C
C
C
C.... SUBROUTINE CANCEL
C     ******************
      SUBROUTINE CANCEL (A,IA,B,IB,EPS)
C
C        SUBROUTINE TO CHECK FOR POLE-ZERO CANCELLATION AND IF FOUND
C        TO PERFORM THE CANCELLATION.
C
C        PROGRAMMER:         P.KATZBERG   7/73
C
C        SUBROUTINE CALLED: GCOM,PNORM
C
      INTEGER RDR,PRT
      DIMENSION A(21),B(21),POL(2,21),GCD(21),IPOL(2),P(21)
      COMMON NRET,RDR,PRT
C
      IF(IA.EQ.0.OR.IB.EQ.0) GO TO 12
      IA1=IA+1
      CALL PPNORM(A,IA1,EPS)
      IB1=IB+1
      CALL PPNORM(B,IB1,EPS)
      IF(IA1.GT.0.AND.IB1.GT.0) GO TO 2
      IF(IB1.NE.0) GO TO 13
      WRITE(6,100)
  100 FORMAT(22H ALL COEFFICIENTS ZERO)
      NRET=1
      GO TO 3
   13 IF(IA1.NE.0) GO TO 2
      IA=0
      A(1)=0.0
      IB=0
      B(1)=1.0
      GO TO 1
    2 DO 4 K=1,IA1
    4 POL(1,K)=A(K)
      DO 5 K=1,IB1
    5 POL(2,K)=B(K)
      IPOL(1)=IA1-1
      IPOL(2)=IB1-1
      CALL GCOM(POL,IPOL,GCD,IG,EPS)
      IF(IG.EQ.0) GO TO 1
      IG1=IG+1
      ID1=IA1-IG
      IA1=IG
      I=ID1
    7 II=I+IA1
      P(I)=A(II)/GCD(IG1)
      IF(ABS(P(I)).LT.EPS) P(I)=0.0
      DO 6 K=1,IA1
      J=K-1+I
      A(J)=A(J)-P(I)*GCD(K)
    6 CONTINUE
      I=I-1
      IF(I.GT.0) GO TO 7
      DO 8 K=1,ID1
    8 A(K)=P(K)
      IA=ID1-1
      ID1=IB1-IG
      IB1=IG
      I=ID1
   10 II=I+IB1
      P(I)=B(II)/GCD(IG1)
      IF(ABS(P(I)).LT.EPS) P(I)=0.0
      DO 9 K=1,IB1
      J=K-1+I
      B(J)=B(J)-P(I)*GCD(K)
    9 CONTINUE
      I=I-1
```

```
      IF(I.GT.0) GO TO 10
      DO 11 K=1,I01
   11 B(K)=P(K)
      IB=I01-1
      GO TO 1
   12 IF(IA.NE.1.OR.ABS(A(1)).GT.EPS) GO TO 1
      IB=0
      B(1)=1.0
    1 NRET=2
    3 RETURN
      END
C*****************************************************************************
C
C
C
C.... SUBROUTINE MRE
C     ***************
      SUBROUTINE MRE(ND)
C
C         SUBROUTINE TO COMPUTE THE MINIMAL REALISATION A,B,C,D OF A
C         TRANSFER FUNCTION MATRIX G, BY SETTING UP AN OBSERVABLE
C         REALISATION USING CANONICAL FORMS AND THEN FINDING THE
C         CONTROLLABLE PART USING ROSENBROCK'S ALGORITHM.
C
C         PROGRAMMER:      B.KATZBERG   8/73
C         MOD1:         A.D.FIELD    27/11/74
C
C         SUBROUTINES CALLED: IO,CANCEL,COMDEN,IOS
C
      INTEGER RDR,PRT,DSK,DTP
      DIMENSION ND(10,10),GD(10,10,21),WK1(21),WK2(21),FILE(2) ,IU(10)
CAB(20,30)
      COMMON NUNIT,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET
      COMMON NSUM
CEQUIVALENCE (AB,A)
      IRET=NRET
      ERR=EPS
C
C         COMPUTE THE L.C.M'S BY ROWS OF THE DENOMINATOR OF G
C
   41 DO 1 I=1,N
      DO 1 J=1,M
C
C   SET WK = ELEMENT (I,J) OF NUMERATOR
      IW1=NN(I,J)
      IW11=IW1+1
      DO 2 K=1,IW11
    2 WK1(K)=GN(I,J,K)
C
C   SET WK2 = DENOMINATOR
      IW2=NDEN
      IW21=IW2+1
      DO 3 K=1,IW21
    3 WK2(K)=CD(K)
      CALL CANCEL (WK1,IW1,WK2,IW2,EPS)
C
C   SET GN = REDUCED NUMERATOR FOR ELEMENT (I,J)
      IW11=IW1+1
      DO 4 K=1,IW11
    4 GN(I,J,K)=WK1(K)
      NN(I,J)=IW1
C
C   SET GD = REDUCED DENOMINATOR FOR ELEMENT (I,J)
      IW21=IW2+1
      DO 5 K=1,IW21
    5 GD(I,J,K)=WK2(K)
      ND(I,J)=IW2
    1 CONTINUE
      NRET=31
      CALL COMDEN(GN,NN,GD,ND,N,M,WK1,IW,EPS)
      IF(NRET.NE.1) GO TO 10
C
C          SET UP D MATRIX
C
      DO 35 I=1,N
      DO 35 J=1,M
   35 D(I,J)=0.0
C
      DO 50 I=1,N
      DO 50 J=1,M
      IF(NN(I,J).GT.ND(I,1)) GO TO 37
      IF(NN(I,J).NE.ND(I,1)) GO TO 50
      K1=NN(I,J)+1
```

```
      CON=GN(I,J,K1)
      IF(NN(I,J).GT.0) GO TO 55
C
      GN(I,J,1)=
      GO TO 56
C
C     SET  G  =  G-D
   55 K1 =K1-1
      DO 51  K=1,K1
   51 GN(I,J,K)=GN(I,J,K)-CON*GD(I,1,K)
      NN(I,J)=K1-1
C
   56 D(I,J)=CON
   50 CONTINUE
      GO TO 136
C
   37 WRITE(6,1001)
 1001 FORMAT(40H NUMERATOR ORDER HIGHER THAN DENOMINATOR/
     +ORDER)
      NRET = 1
      GO TO 10
C
C            SET UP AN INITIAL A MATRIX
C
  136 NSUM=0
C
C      CALCULATE THE DIMENSION OF THE A MATRIX
      DO 8  I=1,N
    8 NSUM=NSUM+ND(I,1)
      IF(NSUM.LE.20) GO TO 9
C
C      OVERFLOW
      WRITE(6,1000)
 1000 FORMAT(23H STORAGE LIMIT EXCEEDED)
      NRET=1
      GO TO 10
C
    9 DO 6  I=1,20
      DO 6  J=1,20
    6 A(I,J)=0.0
C
C      SET UP BLOCK IROW OF THE A MATRIX,FOR IROW = 1,N
      IFIN=0
      DO 7  IROW=1,N
      IF(ND(IROW,1).EQ 0) GO TO 7
      IST= IFIN+1
      IFIN=IFIN+ND(IROW,1)
      A(IST,IFIN)=-GD(IROW,1,1)
      I1=IST
      ID1=ND(IROW,1)
      IF(ID1.LT.2) GO TO 7
      DO 77  I=2,ID1
      I1=I1+1
      A(I1,I1-1)=1.0
      A(I1,IFIN)=-GD(IROW,1,I)
   77 CONTINUE
    7 CONTINUE
C
C            SET UP AN INITIAL B MATRIX
C
      DO 11  I=1,20
      DO 11  J=1,10
   11 B(I,J)= 0.0
C
      IFIN=0
      DO 12  IROW=1,N
      IF(ND(IROW,1).EQ.0) GO TO 12
      IST=IFIN+1
      IFIN=IFIN+ND(IROW,1)
      DO 13  J=1,M
      I1=NN(IROW,J)+IST
   38 K=0
      DO 13  I=IST,I1
      K=K+1
      B(I,J)=GN(IROW,J,K)
   13 CONTINUE
   12 CONTINUE
C
C            SET UP INITIAL C MATRIX
C
   15 DO 16  I=1,10
      DO 16  J=1,20
   16 C(I,J)=0.0
      J=0
      DO 17  I=1,N
```

```
      IF(ND(I,1).EQ.0) GO TO 17
      J=J+ND(I,1)
      C(I,J)=1.0
   17 CONTINUE
      GO TO 888
   10 WRITE(6,101)
  101 FORMAT(1H ,' OBSERVABLE REALISATION NOT OBTAINED')
  888 RETURN
      END
C.......................................................................
C
C
C
C
C          SUBROUTINE CRANK
C          ****************
      SUBROUTINE CRANK(ARANK,EPS,MS,M,IRAN)
C.....SUBROUTINE TO CALCULATE THE RANK OF AN MSXM MATRIX
C.....USES GAUSS ELIMIMATION WITH FULL PIVOTING
      COMPLEX ARANK,Q
      DIMENSION ARANK(20,20),IR(20),IC(20)
C.....PRESET ROW AND COLUMN INTERCHANGES
      DO 100 I=1,MS
  100 IR(I)=I
      DO 110 I=1,M
  110 IC(I)=I
      IRAN=M
      MM=M-1
C.....BEGIN ELIMINATION PROCEDURE
      DO 200 LS=1,MM
      AMAG=0.
C.....SEARCH FOR PIVOT
      DO 120 I=LS,MS
      DO 120 J=LS,M
      ADUM=CABS(ARANK(IR(I),IC(J)))
      IF(ADUM-AMAG)120,120,105
  105 IS=I
      JS=J
      AMAG=ADUM
  120 CONTINUE
C.....TEST FOR COMPLETION
      IF(AMAG-EPS)125,125,130
  125 IRAN=LS-1
      GO TO 300
  130 CONTINUE
C ....INTERCHANGE ROW AND COLUMN INDICES
      IT=IR(LS)
      IR(LS)=IR(IS)
      IR(IS)=IT
      IT=IC(LS)
      IC(LS)=IC(JS)
      IC(JS)=IT
C.....ELIMINATE IC(LS) COLUMN
      LSP=LS+1
      DO 150 I=LSP,MS
      Q=ARANK(IR(I),IC(LS))/ARANK(IR(LS),IC(LS))
      DO 150 J=LSP,M
      ARANK(IR(I),IC(J))=ARANK(IR(I),IC(J))-Q*ARANK(IR(LS),IC(J))
  150 CONTINUE
C.....PATCH UP RANK TEST
      IF(LS-MM)170,160,160
  160 AMAG=0.
      DO 162 I=LSP,MS
      ADUM=CABS(ARANK(IR(I),IC(M)))
      IF(ADUM-AMAG)162,162,161
  161 AMAG=ADUM
  162 CONTINUE
      IF(AMAG-EPS)165,165,170
  165 IRAN=M-1
  170 CONTINUE
  200 CONTINUE
  300 CONTINUE
      RETURN
      END
C.......................................................................
C
C
C
C
C          SUBROUTINE CJORDA
C          *****************
C ...      TO CALCULATE THE JORDAN FORM OF A MATRIX WITH MULTIPLE
C ...      E-VALUES BY GENERATING A SET OF GENERALISED E-VECTORS
C.....USES SSP SUBROUTINES ARRAY AND MFGR AND
C....     RRANK,INDEP
```

```
      SUBROUTINE CJORDA(K3,LSTOP)
      INTEGER RDR,PRT,DSK,DTP
      COMPLEX AP,EE,E,E4,U,UU,UR
      COMPLEX XE,BB,CC
      COMMON NRIT,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JDIM(20),K
      COMMON JSIGN,MULT(20),WR(20),WI(20),VR(20,20),VI(20,20),NDIAG(20)
      DIMENSION AP(5,20,20),U(20,1),UR(20,1),EE(20,20)
C....
      DIMENSION E4(20,20),E(20,20),UU(20,20)
      DIMENSION XE(20,20),BB(20,10),CC(10,20)
      EQUIVALENCE (XE(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(1,
      EQUIVALENCE (AP(1,1,1),GN(1,1,1)),(E4(1,1),B3(1,1)),(E(1,1),VI(1,
      EQUIVALENCE (UU(1,1),CD(1,1)),(EE(1,1),VR(1,1))
      IFIN=NSUM
      LSTOP=0
      IIDIM=1
      IVEC=1
      IRAN=0
      LLSUM=0
      KPOINT=1
      K1=0
      I1=1
C....BEGIN LOOP FOR EACH E-VALUE
      DO 1  I10=1,K3
      ITEST=0
      K1=K1+MULT(I1)
      K=1
      DO 1 I=1,IFIN
      DO 2 J=1,IFIN
      IF(I.EQ.J)GO TO 3
      AP(1,I,J)=CMPLX(A(I,J),0.)
      EE(I,J)=AP(1,I,J)
      GO TO 2
    3 AP(1,I,J)=CMPLX(A(I,J)-WR(K1),-WI(K1))
      EE(I,J)=AP(1,I,J)
    2 CONTINUE
    1 CONTINUE
      NDIAG(I10)=0
   22 DO 11 IJ1=1,IFIN
      DO 12 IJ2=1,IFIN
      E(IJ1,IJ2)=AP(K,IJ1,IJ2)
      E4(IJ1,IJ2)=E(IJ1,IJ2)
   12 CONTINUE
   11 CONTINUE
      IF(MULT(I1)-1)200,200,40
  200 K=2
      NDIAG(I10)=1
      GO TO 33
C....FIND RANK OF AP(K,I,J)
   40 KIRAN=IRAN
      CALL CRANK(E4,EPS,IFIN,IFIN,IRAN)
      DO 70 I=1,IFIN
      DO 71 J=1,IFIN
   71 E4(I,J)=E(I,J)
   70 CONTINUE
      IF(K-1)303,303,304
  303 NDIAG(I10)=IFIN-IRAN
      GO TO 6
  304 IF(KIRAN.EQ.IRAN)GO TO 33
      GO TO 6
   33 DO 80 I=1,IFIN
      DO 81 J=1,IFIN
   81 E4(I,J)=AP(K-1,I,J)
   80 CONTINUE
      CALL CNULL(E4,IFIN,IFIN,EPS,XE,NE)
      IF(NE.EQ.0)GO TO 23
      DO 166 I=1,IFIN
      IF(K.LE.2)GO TO 166
      DO 313 IJ1=1,IFIN
  313 E4(I,IJ1)=AP(K-2,I,IJ1)
  166 CONTINUE
      CALL CINDEP(K,E4,NE,KPOINT,IIDIM,UU)
      KW=K-1
      WRITE(6,111)I10,KW
  111 FORMAT(1H ,/,2X,'LEADING E-VECTORS FOR E-VALUE ',I2,/,3X,
     +' OF RANK ',I2)
      DO 315 I2=1,IFIN
      WRITE(6,1248)(XE(I2,J),J=1,NE)
  315 CONTINUE
      IF((K-1)*NE.LE.MULT(I1))GO TO 23
   99 WRITE(6,100)
 1248 FORMAT(1H ,2X,10(1X,F7.3))
```

```
  1.. FORMAT(1H ,/,'   WRONG CALCULATION OF RANK  PROCEDURE STOPPED
      LSTOP=1
      RETURN
   23 LS=1
   13 DO 14 I2=1,IFIN
   14 U(I2,1)=XE(I2,LS)
   41 IND=1
C.....BEGIN PROCEDURE FOR CALCULATIOON OF GENERALISED E-VECTOR CHA
C.....WITH LEADING E-VECTOR XE(I,LS)
      IF(MULT(I1).EQ.1)GO TO 18
   20 IF(IND.GE.K-1)GO TO 18
      DO 15 I2=1,IFIN
      DO 16 I3=1,IFIN
      E(I2,I3)=AP(K-IND-1,I2,I3)
   16 CONTINUE
   15 CONTINUE
      IID1=20
      IID2=1
      NN3=1
      CALL CMATRM(IID1,IID2,IFIN,IFIN,NN3,E,U,UR)
   34 DO 17 I2=1,IFIN
   17 UU(I2,IVEC)=UR(I2,1)
      IVEC=IVEC+1
      IND=IND+1
      IF(IND.EQ.K-1)GO TO 18
      GO TO 20
   18 DO 19 I2=1,IFIN
   19 UU(I2,IVEC)=U(I2,1)
      IVEC=IVEC+1
      JDIM(IIDIM)=K-1
      LLSUM=LLSUM+JDIM(IIDIM)
      IIDIM=IIDIM+1
      IF(NE.GT.1)GO TO 21
      GO TO 109
   21 LS=LS+1
      IF(LS.LE.NE)GO TO 13
C.....GENERATION OF CHAIN COMPLETED
  109 IF(LLSUM-MULT(I1))305,306,306
  306 LLSUM=0
      GO TO 9
  305 K=K-1
      IF(K.EQ.1)GO TO 99
C....   GO TO FIND ANOTHER INDEPENDENT CHAIN BUT OF SMALLER RANK
      GO TO 33
    6 K=K+1
      IF(K.GT.MULT(I1)+1)GO TO 99
      IID1=20
      IID2=20
      CALL CMATRM(IID1,IID2,IFIN,IFIN,IFIN,E,EC,E4)
      DO 31 IJ=1,IFIN
      DO 32 I6=1,IFIN
      AP(K,IJ,I6)=E4(IJ,I6)
   32 E(IJ,I6)=E4(IJ,I6)
   31 CONTINUE
      GO TO 40
    9 I1=I1+MULT(I1)
   10 KPOINT=KPOINT+NDIAG(I10)
      DO 202 I=1,IFIN
      DO 203 J=1,IFIN
      VR(I,J)=REAL(UU(I,J))
      VI(I,J)=AIMAG(UU(I,J))
  203 CONTINUE
  202 CONTINUE
      RETURN
      END
C.... .................................... ..............................
C
C
C
C
C         SUBROUTINE CNULL
C         ****************
      SUBROUTINE CNULL(E,M,N,TOL,XE,NE)
C.... TO DETERMINE BASIS VECTORS AND DIMENSION OF NULL SPACE
C.....CALLS SSP SUBROUTINES MFGR,ARRAY
      COMPLEX E,XE,EN
      DIMENSION E(20,20),XE(20,20),IROW(20),ICOL(20),EN(20)
      II1=2
      II2=20
      CALL CARRAY(II1,M,N,II2,II2,E,E)
      CALL CMFGR(E,M,N,TOL,IR,IROW,ICOL)
      NE=N-IR
      IF(NE.EQ.0)GO TO 5
      IF(IR)6,6,7
    6 DO 8 I=1,NE
```

```
      DO 9 J=1,M
      IF(I.EQ.J)GO TO 10
      XE(I,J)=CMPLX(0.,0.)
      GO TO 9
   10 XE(I,J)=CMPLX(1.,0.)
    9 CONTINUE
    8 CONTINUE
      GO TO 5
C.... RANK NOT M,DIMENSION OF NULL SPACE LESS THAN ORDER OF MATRIX
    7 II1=1
      CALL CARRAY(II1,M,N,II2,II2,E,E)
      DO 4 J=1,NE
      DO 2 K=1,M
    2 EN(K)=CMPLX(0.,0.)
      LI=ICOL(IR+J)
      EN(LI)=CMPLX(1.,0.)
      DO 3 L=1,IR
      LE=ICOL(L)
    3 EN(LE)=E(L,IR+J)
      DO 4 I=1,M
    4 XE(I,J)=EN(I)
    5 RETURN
      END
C.........................................................................
C
C
C
C
C        SUBROUTINE CARRAY
C        *****************
      SUBROUTINE CARRAY(MODE,I,J,N,M,S,D)
C.....CONVERT ARRAY FROM SINGLE TO DOUBLE DIMENSION OR V.V.
C.....FOR MODE=1 OR 2 RESP.
      COMPLEX S,D
      DIMENSION S(1),D(1)
      NI=N-I
C.....TEST TYPE OF CONVERSION
      IF(MODE-1)99,99,120
C.....CONVERT FROM SINGLE TO DOUBLE DIMENSION
   99 IJ=I*J+1
      NM=N*J+1
      DO 110 K=1,J
      NM=NM-NI
      DO 110 L=1,I
      IJ=IJ-1
      NM=NM-1
  110 D(NM)=S(IJ)
      GO TO 140
C.....CONVERT FROM DOUBLE TO SINGLE DIMENSION
  120 IJ=0
      NM=0
      DO 130 K=1,J
      DO 125 L=1,I
      IJ=IJ+1
      NM=NM+1
  125 S(IJ)=D(NM)
  130 NM=NM+NI
  140 RETURN
      END
C.........................................................................
C
C
C
C
C        SUBROUTINE CMFGR
C        ****************
C
      SUBROUTINE CMFGR(A,M,N,EPS,IRANK,IROW,ICOL)
C.....DETERMINATION OF THE FOLLOWING FOR A M X N MATRIX A
C.....1  RANK AND LINEARLY INDDEPENDANT ROWS AND COLUMNS
C.....2  FACTORISATION OF SUBMATRIX OF MAX. RANK
C.....3  NON-BASIC ROWS IN TERMS OF BASIC ONES
C.....4  BASIC VARIABLES IN TERMS OF FREE ONES
C.....   GAUSSIAN ELIMINATION WITH FULL PIVOTING
      COMPLEX A,HOLD,PIV,SAVE
      DIMENSION A(1),IROW(1),ICOL(1)
C.... INITIALIZE COLUMN INDEX VECTOR  SEARCH FIRST PIVOT ELEMENT
    4 IRANK=0
      PIV=CMPLX(0.,0.)
      JJ=0
      DO 6 J=1,N
      ICOL(J)=J
      DO 6 I=1,M
      JJ=JJ+1
      HOLD=A(JJ)
      IF(CABS(PIV)-CABS(HOLD))5,6,6
```

```
 5 PIV=HOLD
 IR=I
 IC=J
 6 CONTINUE
C.....INITIALIZE ROW INDEX VECTOR
 DO 7 I=1,M
 7 IROW(I)=I
C.....SET UP INTERNAL TOLERANCE
 TOL =CABS(EPS*PIV)
C.... INITILIZE ELIMINATION LOOP
 NM=N*M
 DO 19 NCOL=M,NM,M
C.....TEST FOR FEASIBILITY OF PIVOT ELEMENT
 8 IF(CABS(PIV)-TOL)20,20,9
C.....UPDATE RANK
 9 IRANK=IRANK+1
C.... INTERCHANGE ROWS IF NECESSARY
 JJ=IR-IRANK
 IF(JJ)12,12,10
 10 DO 11 J=IRANK,NM,M
 I=J+JJ
 SAVE=A(J)
 A(J)=A(I)
 11 A(I)=SAVE
C.... UPDATE ROW INDEX VECTOR
 JJ=IROW(IR)
 IROW(IR)=IROW(IRANK)
 IROW(IRANK)=JJ
C.... INTERCHANGE COLUMNS IF NECESSARY
 12 JJ=(IC-IRANK)*M
 IF(JJ)15,15,13
 13 KK=NCOL
 DO 14 J=1,M
 I=KK+JJ
 SAVE=A(KK)
 A(KK)=A(I)
 KK=KK-1
 14 A(I)=SAVE
C.....UPDATE COLUMN INDEX VECTOR
 JJ=ICOL(IC)
 ICOL(IC)=ICOL(IRANK)
 ICOL(IRANK)=JJ
 15 KK=IRANK+1
 MM=IRANK-M
 LL=NCOL+MM
 IF(MM)16,25,25
C.... TRANSFER CURRENT SUBMATRIX AND SEARCH FOR NEXT PIVOT
 16 JJ=LL
 SAVE=PIV
 PIV=CMPLX(0.,0.)
 DO 19 J=KK,M
 JJ=JJ+1
 HOLD=A(JJ)/SAVE
 A(JJ)=HOLD
 L=J-IRANK
C.....TEST FOR LAST COLUMN
 IF(IRANK-N)17,19,19
 17 II=JJ
 DO 19 I=KK,N
 II=II+M
 MM=II-L
 A(II)=A(II)-HOLD*A(MM)
 IF(CABS(A(II))-CABS(PIV))19,19,18
 18 PIV=A(II)
 IR=J
 IC=I
 19 CONTINUE
C.....SET UP MATRIX EXPRESSING ROW DEPENDANCIES
 20 IF(IRANK-1)3,25,21
 21 IR=LL
 DO 24 J=2,IRANK
 II=J-1
 IR=IR-M
 JJ=LL
 DO 23 I=KK,M
 HOLD=CMPLX(0.,0.)
 JJ=JJ+1
 MM=JJ
 IC=IR
 DO 22 L=1,II
 HOLD=HOLD+A(MM)*A(IC)
 IC=IC-1
 22 MM=MM-M
 23 A(MM)=A(MM)-HOLD
 24 CONTINUE
```

```
 25 IF(N-IRANK)3,3,26
C.....SET UP MATRIX EXPRESSING BASIC VARIABLES IN TERMS OF FREE
C.....PARAMETERS (HOMOGENEOUS SOLUTION)
 26 IR=LL
 KK=LL+M
 DO 30 J=1,IRANK
 DO 29 I=KK,NM,M
 JJ=IR
 LL=I
 HOLD=CMPLX(0.,0.)
 II=J
 27 II=II-1
 IF(II)29,29,28
 28 HOLD=HOLD-A(JJ)*A(LL)
 JJ=JJ-M
 LL=LL-1
 GO TO 27
 29 A(LL)=(HOLD-A(LL))/A(JJ)
 30 IR=IR-1
 3 RETURN
 END
C.....................................................................
C
C
C
C
C        SUBROUTINE CMATRM
C        ****************
 SUBROUTINE CMATRM(IID1,IID2,N1,N2,N3,A1,B1,C1)
C.... MULTIPLICATION OF TWO (COMPLEX) MATRICES
 COMPLEX A1,B1,C1,C
 DIMENSION A1(IID1,20),B1(20,IID2),C1(IID1,IID2)
 DO 1 I=1,N1
 DO 2 J=1,N3
 C=CMPLX(0.,0.)
 DO 3 JJ=1,N2
 3 C=C+A1(I,JJ)*B1(JJ,J)
 C1(I,J)=C
 2 CONTINUE
 1 CONTINUE
 RETURN
 END
C.....................................................................
C
C
C
C
C        SUBROUTINE CINDEP
C        ****************
C.... TO DETERMINE WHETHER VECTORS GENERATING THE NULL SPACE OF
C.... (A-LI)**K CAN BE GENERALISED E-VECTORS
 SUBROUTINE CINDEP(K,EE,NE,KPOINT,IIDIM,UU)
 INTEGER RDR,PRT,DSK,DTP
 COMPLEX UU,XE,BB,CC,E4,U,UR,EE
 COMMON NRET,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
 COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
 COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
 COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JDIM(20),
 COMMON JSIGN,MULT(20),WR(20),WI(20),VR(20,20),VI(20,20),NDIAG(2
 DIMENSION E4(20,20),U(20,1),UR(20,1),EE(20,20)
 DIMENSION UU(20,20)
 DIMENSION XE(20,20),BB(20,10),CC(10,20)
 EQUIVALENCE (XE(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(
 IFIN=NSUM
 IID1=20
 IID2=1
 IF(K.LE.2)GO TO 231
 LL=1
C.... CHECK IF (A-LI)**K-1  =  0
 DO 213 J=1,NE
 DO 214 I=1,IFIN
 214 U(I,1)=XE(I,J)
 CALL CMATRM(IID1,IID2,IFIN,IFIN,IID2,EE,U,UR)
 DO 215 I1=1,IFIN
 IF(CABS(UR(I1,1)).LE.EPS)GO TO 215
 GO TO 213
 215 CONTINUE
 KINDEX(LL)=J
 LL=LL+1
 213 CONTINUE
 IF(LL.EQ.1)GO TO 216
 IF(NE.EQ.1)GO TO 220
C.....DELETE UNSUITABLE E-VECTORS FROM E-VECTOR MATRIX XE
 DO 217 I=1,LL-1
 LX=KINDEX(I)-I+1
```

```
      IF(LX.EQ.NE)GO TO 222
      DO 218 I1=1,IFIN
      DO 219 I2=LX,NE-1
  219 XE(I1,I2)=XE(I1,I2+1)
  218 CONTINUE
  222 NE=NE-1
  217 CONTINUE
C.... PICK THE LEADING E-VECTORS ALREADY OBTAINED FROM UU
  216 IF(NE.LE.1)GO TO 210
  231 KJ=IIDIM-KPOINT
      IF(KJ)696,696,695
  695 KSTART=1
      IF(KPOINT.EQ.1)GO TO 698
      DO 699 I=1,KPOINT-1
  699 KSTART=KSTART+JDIM(I)
  698 KJJ=KSTART
      DO 701 I=1,KJ
      DO 700 J=1,IFIN
      E4(J,I)=UJ(J,KJJ)
  700 CONTINUE
      KJJ=KJJ+JDIM(I)
  701 CONTINUE
  696 I11=NE
      I10=1
      INE1=KJ+1
      INE2=1
  278 IF(K.GT.2)GO TO 228
      DO 262 I=1,IFIN
  262 U(I,1)=XE(I,INE2)
      GO TO 263
C.... GENERATE PRESENT LEADING E-VECTORS AND CHECK IF THEY FORM
C.....A L.I. SET WITH THE ALREADY OBTAINED E-VECTORS
  228 DO 241 I=1,IFIN
  241 UR(I,1)=XE(I,INE2)
      CALL CMATXM(IID1,IID2,IFIN,IFIN,IID2,EE,UR,U)
  263 II=INE1
      DO 244 I2=1,IFIN
  244 E4(I2,II)=U(I2,1)
      IF(II.LE.1)GO TO 224
      CALL CRANK(E4,EPS,IFIN,II,IRAN)
      IF(IRAN.EQ.II)GO TO 224
      IF(INE1.EQ.(NE+KJ))GO TO 225
      DO 226 I3=1,IFIN
      DO 227 I4=1,NE-1
  227 XE(I3,I4)=XE(I3,I4+1)
  226 CONTINUE
  225 NE=NE-1
      GO TO 255
  224 INE1=INE1+1
      INE2=INE2+1
  255 I10=I10+1
      IF(I10.LE.I11)GO TO 278
  210 RETURN
      END
C..................................................................
C
C
C
C
```

```
C.......................................................................
C
C
C
C
C      SUBROUTINE RREAL
C      ****************
      SUBROUTINE RREAL
      REAL KEEPL1
      INTEGER RDR,PRT,DSK,DTP
      COMMON NRET,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AA(20,20),BB(20,10),CC(10,20),NDIM,IC,ICOMPL,JDIM(20),KINDE
      COMMON JSIGN,MULT(20),WR(20),WI(20),VR(20,20),VI(20,20),NDIAG(20)
      K=0
      K3=0
      IC=0
C.....CHECK FOR DISTINCT E-VALUES
      IFIN=NSUM
      WRITE(6,1036)
 1036 FORMAT(1H ,1X,'** E-VALUES REAL , MATRICES WITH REAL ELEMENTS
      DO 501 I=1,IFIN-1
      IF(MULT(I))501,501,402
  402 K=K+1
      K3=K3+1
      DO 601 J=I+1,IFIN
      IF(ABS(WR(I)-WR(J)).GT.EPS)GO TO 601
      MULT(I)=MULT(I)+1
      MULT(K+1)=0
      IC=1
      IF(J-I-1)666,666,610
  610 KEEPL1=WR(K+1)
      KEEPL2=WI(K+1)
      WR(K+1)=WR(J)
      WR(J)=KEEPL1
  666 K=K+1
  601 CONTINUE
  501 CONTINUE
      IF(MULT(IFIN).GT.0)K3=K3+1
      WRITE(6,222)K3
 1004 CALL JORDAN(K3,LSTOP)
  222 FORMAT(1H ,'  DISTINCT E-VALUES ',I2)
      IF(LSTOP)1006,1006,2000
 1006 WRITE(6,543)
      DO 670 I=1,IFIN
      WRITE(6,500)(VR(I,J),J=1,IFIN)
  671 CONTINUE
  670 CONTINUE
  543 FORMAT(1H ,/,' TRANSFORNATION MATRIX FOR JORDAN FORM')
  500 FORMAT(1H ,10(1X,F7.3))
      CALL TRANS(K3)
      IF(IRET.EQ.1)GO TO 2000
      CALL SWRIT
 2000 RETURN
      END
C.......................................................................
C
C
C
C
C.....SUBROUTINE TRANS
C     ****************
      SUBROUTINE TRANS(K3)
      INTEGER RDR,PRT,DSK,DTP
      COMMON NRET,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AA(20,20),BB(20,10),CC(10,20),NDIM,IC,ICOMPL,JDIM(20),KINC
      COMMON JSIGN,MULT(20),WR(20),WI(20),TT(20,20),VI(20,20),NDIAG(20)
      DIMENSION TTT(20,20),IV(20)
      DIMENSION RIND(20),AT(20,20)
C.....TRANSFORM B MATRIX
      IFIN=NSUM
      DO 222 I=1,IFIN
      DO 223 J=1,IFIN
  223 AT(I,J)=TT(I,J)
  222 CONTINUE
C.....INVERT TRANSFORMATION MATRIX
      IDIM1=20
      IFAIL=1
      CALL F01AAF(AT,IDIM1,IFIN,TTT,IDIM1,RIND,IFAIL)
      WRITE(6,1024)
      IF(IFAIL)999,999,999
  999 WRITE(6,1020)
```

```
1020 FORMAT(1H ,'  FAILURE IN F01AAF')
IRET=1
GO TO 51
399 DO 1022 IL=1,IFIN
WRITE(6,1023)(TTT(IL,JL),JL=1,IFIN)
1022 CONTINUE
1023 FORMAT(1H ,10(1X,F7.3))
1024 FORMAT(1H ,/,2X,'INVERSE OF TRANSFORMATION MATRIX')
IID1=20
IID2=10
CALL MATRM(IID1,IID2,IFIN,IFIN,M,TTT,B,BB)
WRITE(6,1896)
1896 FORMAT(1H ,1X,' TRANSFORMED B MATRIX')
DO 71 I=1,IFIN
WRITE(6,1025)(BB(I,J1),J1=1,M)
71 CONTINUE
1025 FORMAT(1H ,10(1X,F7.3))
C.... TRANSFORM C
CALL MATRM(IID2,IID1,N,IFIN,IFIN,C,TT,CC)
WRITE(6,2016)
2016 FORMAT(1H ,1X,' TRANSFORMED C MATRIX')
DO 72 I=1,N
WRITE(6,1025)(CC(I,J1),J1=1,IFIN)
72 CONTINUE
420 IID1=20
IID2=20
CALL MATRM(IID1,IID2,IFIN,IFIN,IFIN,TTT,A,AT)
CALL MATRM(IID1,IID2,IFIN,IFIN,IFIN,AT,TT,AA)
WRITE(6,1035)
DO 830 I=1,IFIN
WRITE(6,1033)(AA(I,J),J=1,IFIN)
1035 FORMAT(1H ,/,'    TRANSFORMED A MATRIX     IN JORDAN FORM')
1033 FORMAT(1H ,/,10(1X,F7.3))
830 CONTINUE
NDIM=IFIN
615 IF(IC)416,416,43
C.... DISTINCT E-VALUES JORDAN FORM DIAGONAL
C..... IRRR IS NUMBER OF SUBSYSTEMS IN WHICH SYSTEM IS DIVIDED
416 IRRR=IFIN
NDIM=IFIN
KBLOC=IFIN
DO 250 IJ=1,KBLOC
NDIAG(IJ)=1
250 JDIM(IJ)=1
GO TO 431
43 IRRR=K3
431 KKK=1
KINDEX(1)=1
DO 302 I=1,IRRR
DO 303 I2=1,NDIAG(I)
KINDEX(KKK+I2)=KINDEX(KKK+I2-1)+JDIM(KKK+I2-1)
303 CONTINUE
KKK=KKK+NDIAG(I)
302 CONTINUE
LSTART=1
DO 50 JJ=1,IRRR
C.... STEP 1,SET V(I)=0,I=1,R
556 IREP=[illegible]
184 LFINI=LSTART+NDIAG(JJ)-1
DO 55 J=LSTART,LFINI
55 IV(J)=0
C.... STEPS 2 AND 3
C.....KIMAX IS MAXIMUM SIZE OF BLOCK FOR WHICH IV=0
54 KIMAX=1
KSTOP=0
DO 56 J=LSTART ,LFINI
IF(IV(J))56,57,56
57 KSTOP=1
IF(KIMAX-JDIM(J))58,58,56
58 KIMAX=JDIM(J)
C.....JSIGN IS J ST. JDIM(J) IS MAX
JSIGN =J
56 CONTINUE
IF(KSTOP)555,555,59
C.... STEP 4
59 KISIGN=KINDEX(JSIGN)+JDIM(JSIGN)-1
DO 61 I=1,M
IF(ABS(BB(KISIGN,I))-EPS)61,61,62
61 CONTINUE
GO TO 91
62 IV(JSIGN)=1
ISG=1
ISG1=I+LSTART-1
DO 63 I=LSTART,LFINI
IF(IV(I))63,64,63
```

```
      64 INDB=KINDEX(I)+JDIM(I)-1
      INDBB=KINDEX(JSIGN)+JDIM(JSIGN)-1
      QUOT=BB(INDB,ISG)/BB(INDBB,ISG)
      821 FORMAT(1H ,10(1X,F7.3))
      DO 65 II=1,JDIM(I)
      DO 66 IB=1,M
      INI1=KINDEX(I)+II-1
      INI2=KINDEX(JSIGN)+JDIM(JSIGN)-JDIM(I)+II-1
      BB(INI1,IB)=BB(INI1,IB)-QUOT*BB(INI2,IB)
      66 CONTINUE
      DO 67 ICC=1,N
      67 CC(ICC,INI2)=CC(ICC,INI2)+QUOT*CC(ICC,INI1)
      65 CONTINUE
      WRITE(6,2876)
      2876 FORMAT(1H ,1X,' TRANSFORMED B')
      DO 822 IN=1,IFIN
      WRITE(6,821)(BB(IN,J),J=1,M)
      822 CONTINUE
      63 CONTINUE
      GO TO 54
C.....IDEL IS INDEX FOR ROWS AND COLUMNS TO BE DELETED
      91 IF(JDIM(JSIGN).EQ.1)GO TO 69
      IREP=1
      GO TO 669
      69 IV(JSIGN)=1
      LFINI=LFINI-1
      669 CALL CONVERT(IV)
      GO TO 54
      183 ISKIP=0
      555 IF(IREP)557,557,556
      557 LSTART=LFINI+1
      50 CONTINUE
      51 RETURN
      END
C.....................................................................
C
C
C
C
C ... SUBROUTINE SWRIT
C     ****************
      SUBROUTINE SWRIT
      INTEGER RDR,PRT,DSK,DTP
      COMMON NRIT,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AA(20,20),BB(20,10),CC(10,20),NDIM,IC,ICOMPL
C.....................................................................
      WRITE(6,129)NDIM
      129 FORMAT(1H ,//,2X,'DIMENSION ',I2)
      WRITE(6,3355)
      3355 FORMAT(1H ,//,'  A MATRIX')
      4 DO 1 I=1,NDIM
      WRITE(6,100)(AA(I,J),J=1,NDIM)
      1 CONTINUE
      WRITE(6,3386)
      3386 FORMAT(1H ,//,'  B MATRIX')
      DO 2 I=1,NDIM
      WRITE(6,100)(BB(I,J),J=1,M)
      2 CONTINUE
      WRITE(6,3416)
      3416 FORMAT(1H ,//,'  C MATRIX')
      DO 3 I=1,N
      WRITE(6,100)(CC(I,J),J=1,NDIM)
      3 CONTINUE
      100 FORMAT(1H ,10(1X,F7.3))
      5 CONTINUE
C.....................................................................
      RETURN
      END
C.....................................................................
C
C
C
C
C.....SUBROUTINE CONVERT
C     *****************
      SUBROUTINE CONVERT(ILS)
C.....TO DELETE ROWS AND COLUMNS FROM AA,BB,CC,
      INTEGER RDR,PRT,DSK,DTP
      COMMON NRIT,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON AA(20,20),BB(20,10),CC(10,20),NDIM,IC,ICOMPL,JDIM(20),KIN
      COMMON JSIGN,MULT(20)
```

```
      DIMENSION ILS(20)
      IST=KINDEX(JSIGN)+JDIM(JSIGN)-1
      IF(IST.GE.NDIM)GO TO 11
      DO 1 I=IST,NDIM-1
      DO 12 J=1,NDIM
   12 AA(I,J)=AA(I+1,J)
      DO 2 J=1,N
      BB(I,J)=BB(I+1,J)
    2 CONTINUE
      DO 3 IJ=1,N
    3 CC(IJ,I)=CC(IJ,I+1)
    1 CONTINUE
      DO 13 J=IST,NDIM
      DO 14 I=1,NDIM
   14 AA(I,J)=AA(I,J+1)
   13 CONTINUE
   11 NDIM=NDIM-1
      JDIM(JSIGN)=JDIM(JSIGN)-1
      IL=0
      IF(JDIM(JSIGN))7,7,8
    7 DO 9 I=JSIGN,19
      ILS(I)=ILS(I+1)
    9 JDIM(I)=JDIM(I+1)
      IL=1
    8 DO 6 I=JSIGN+1,19
    6 KINDEX(I)=KINDEX(I+IL)-1
      RETURN
      END
C.......................................................................
C
C
C
C....       SUBROUTINE  MATRM
C           *****************
      SUBROUTINE MATRM(IID1,IID2,N1,N2,N3,A1,B1,C1)
C....       MULTIPLICATION OF TWO (COMPLEX) MATRICES
      DIMENSION A1(IID1,20),B1(20,IID2),C1(IID1,IID2)
      DO 1 I=1,N1
      DO 2 J=1,N3
      C=0.
      DO 3 JJ=1,N2
    3 C=C+A1(I,JJ)*B1(JJ,J)
      C1(I,J)=C
    2 CONTINUE
    1 CONTINUE
      RETURN
      END
C.......................................................................
C
C
C
```

```
      SUBROUTINE RRANK(ARANK,EPS,MS,M,IRAN)
C.... SUBROUTINE TO CALCULATE THE RANK OF AN MSXM MATRIX
C.....USES GAUSS ELIMINATION WITH FULL PIVOTING
      DIMENSION ARANK(20,20),IR(20),IC(20)
C.... PRESET ROW AND COLUMN INTERCHANGES
      DO 100 I=1,MS
  100 IR(I)=I
      DO 110 I=1,M
  110 IC(I)=I
      IRAN=M
      MM=M-1
C.....BEGIN ELIMINATION PROCEDURE
      DO 200 LS=1,MM
      AMAG=0.
C.....SEARCH FOR PIVOT
      DO 120 I=LS,MS
      DO 120 J=LS,M
      ADUM=ABS(ARANK(IR(I),IC(J)))
      IF(ADUM-AMAG)120,120,105
  105 IS=I
      JS=J
      AMAG=ADUM
  120 CONTINUE
C.....TEST FOR COMPLETION
      IF(AMAG-EPS)125,125,130
  125 IRAN=LS-1
      GO TO 300
  130 CONTINUE
C.... INTERCHANGE ROW AND COLUMN INDICES
      IT=IR(LS)
      IR(LS)=IR(IS)
      IR(IS)=IT
      IT=IC(LS)
      IC(LS)=IC(JS)
      IC(JS)=IT
C.... ELIMINATE IC(LS) COLUMN
      LSP=LS+1
      DO 150 I=LSP,MS
      Q=ARANK(IR(I),IC(LS))/ARANK(IR(LS),IC(LS))
      DO 150 J=LSP,M
      ARANK(IR(I),IC(J))=ARANK(IR(I),IC(J))-Q*ARANK(IR(LS),IC(J))
  150 CONTINUE
C.... PATCH UP RANK TEST
      IF(LS-MM)170,160,160
  160 AMAG=0.
      DO 162 I=LSP,MS
      ADUM=ABS(ARANK(IR(I),IC(M)))
      IF(ADUM-AMAG)162,162,161
  161 AMAG=ADUM
  162 CONTINUE
      IF(AMAG-EPS)165,165,170
  165 IRAN=M-1
  170 CONTINUE
  200 CONTINUE
  300 CONTINUE
      RETURN
      END
C..............................................................
C
C
C
C
C.....SUBROUTINE JORDAN
      *****************
C ...  TO CALCULATE THE JORDAN FORM OF A MATRIX WITH MULTIPLE
C ... E-VALUES BY GENERATING A SET OF GENERALISED E-VECTORS
C.....USES SSP SUBROUTINES ARRAY AND MFGR AND
C.... RRANK,INDEP
      SUBROUTINE JORDAN(K3,LSTOP)
      INTEGER RDR,PRT,DSK,DTP
      COMMON NRET,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON X(20,20),BB(20,10),CC(10,20),NDIM,IC,ICOMPL,JDIM(20),KTN
      COMMON JSIGN,MULT(20),WR(20),WI(20),UU(20,20),VI(20,20),NDIAG(20
      DIMENSION AP(5,20,20),U(20,1),UR(20,1),EF(20,20)
C....
      DIMENSION E4(20,20),E(20,20)
      EQUIVALENCE (GN(1,1,1),AP(1,1,1))
      EQUIVALENCE (BB(1,1),E4(1,1)),(VI(1,1),E(1,1))
      IFIN=NSUM
      LSTOP=0
      ITOM=1
      IVEC=1
      IRAN=0
```

```
      LLSUM=0
      KPOINT=1
      K1=0
      I1=1
C.....BEGIN LOOP FOR EACH E-VALUE
      DO 1 I1 =1,K3
      ITEST=0
      K1=K1+MULT(I1)
      K=1
      DO 1 I=1,IFIN
      DO 2 J=1,IFIN
      IF(I.EQ.J)GO TO 3
      AP(1,I,J)=A(I,J)
      E(I,J)=AP(1,I,J)
      GO TO 2
    3 AP(1,I,J)=A(I,J)-WR(K1)
      E(I,J)=AP(1,I,J)
    2 CONTINUE
    1 CONTINUE
      NDIAG(I10)=0
   22 DO 11 IJ1=1,IFIN
      DO 12 IJ2=1,IFIN
      E(IJ1,IJ2)=AP(K,IJ1,IJ2)
      E4(IJ1,IJ2)=E(IJ1,IJ2)
   12 CONTINUE
   11 CONTINUE
      IF(MULT(I1)-1)200,200,40
  200 K=2
      NDIAG(I10)=1
      GO TO 33
C.... FIND RANK OF AP(K,I,J)
   40 KIRAN=IRAN
      CALL RRANK(E4,EPS,IFIN,IFIN,IRAN)
      DO 70 I=1,IFIN
      DO 71 J=1,IFIN
   71 E4(I,J)=E(I,J)
   70 CONTINUE
      IF(K-1)303,303,304
  303 NDIAG(I10)=IFIN-IRAN
      GO TO 6
  304 IF(KIRAN.EQ.IRAN)GO TO 33
      GO TO 6
   33 DO 80 I=1,IFIN
      DO 81 J=1,IFIN
   81 E4(I,J)=AP(K-1,I,J)
   80 CONTINUE
      CALL NULL(E4,IFIN,IFIN,EPS,XE,NE)
      IF(NE.EQ.0)GO TO 23
      DO 166 I=1,IFIN
      IF(K.LE.2)GO TO 166
      DO 313 IJ1=1,IFIN
  313 E4(I,IJ1)=AP(K-2,I,IJ1)
  166 CONTINUE
      CALL INDEP(K,E4,NE,KPOINT,IIDIM)
      KW=K-1
      WRITE(6,347)I10,KW
  347 FORMAT(1H ,/,2X,'LEADING E-VECTORS FOR E-VALUE ',I2,/,3X,' OF
     +I2)
      DO 315 I2=1,IFIN
      WRITE(6,111)(XE(I2,J),J=1,NE)
  315 CONTINUE
      IF((K-1)*NE.LE.MULT(I1))GO TO 23
   99 WRITE(6,100)
  111 FORMAT(1H ,10(1X,F7.3))
  100 FORMAT(1H ,/,'    WRONG CALCULATION OF RANK  PROCEDURE STOPPE
      LSTOP=1
      RETURN
   23 LS=1
   13 DO 14 I2=1,IFIN
   14 U(I2,1)=XE(I2,LS)
   41 IND=1
C.... BEGIN PROCEDURE FOR CALCULATIOON OF GENERALISED E-VECTOR CH
C.... WITH LEADING E-VECTOR XE(I,LS)
      IF(MULT(I1).EQ.1)GO TO 18
   20 IF(IND.GE.K-1)GO TO 18
      DO 15 I2=1,IFIN
      DO 16 I3=1,IFIN
      E(I2,I3)=AP(K-IND-1,I2,I3)
   16 CONTINUE
   15 CONTINUE
      IID1=20
      IID2=1
      NN3=1
      CALL MATRM (IID1,IID2,IFIN,IFIN,NN3,E,U,UR)
   34 DO 17 I2=1,IFIN
```

```
17 UU(I2,IVEC)=UR(I2,1)
IVEC=IVEC+1
IND=IND+1
IF(IND.EQ.K-1)GO TO 18
GO TO 20
18 DO 19 I2=1,IFIN
19 UU(I2,IVEC)=U(I2,1)
IVEC=IVEC+1
JDIM(IIDIM)=K-1
LLSUM=LLSUM+JDIM(IIDIM)
IIDIM=IIDIM+1
IF(NE.GT.1)GO TO 21
GO TO 109
21 LS=LS+1
IF(LS.LE.NE)GO TO 13
C.....GENERATION OF CHAIN COMPLETED
109 IF(LLSUM-MULT(I1))305,306,306
306 LLSUM=0
GO TO 9
305 K=K-1
IF(K.EQ.1)GO TO 99
C....   GO TO FIND ANOTHER INDEPENDENT CHAIN BUT OF SMALLER RANK
GO TO 33
6 K=K+1
IF(K.GT.MULT(I1)+1)GO TO 99
IID1=20
IID2=20
CALL MATRM(IID1,IID2,IFIN,IFIN,IFIN,E,E2,E4)
DO 31 IJ=1,IFIN
DO 32 I6=1,IFIN
AP(K,IJ,I6)=E4(IJ,I6)
32 E(IJ,I6)=E4(IJ,I6)
31 CONTINUE
GO TO 40
9 I1=I1+MULT(I1)
10 KPOINT=KPOINT+NDIAG(I10)
RETURN
END
C.............................................................
C
C
C
C
C ....  SUBROUTINE NULL
C       **************
SUBROUTINE NULL(E,M,N,TOL,XE,NE)
C.....TO DETERMINE BASIS VECTORS AND DIMENSION OF NULL SPACE
C.... CALLS SSP SUBROUTINES MFGR,ARRAY
DIMENSION E(20,20),XE(20,20),IROW(20),ICOL(20),EN(20)
II1=2
II2=20
CALL ARRAY(II1,M,N,II2,II2,E,E)
CALL MFGR(E,M,N,TOL,IR,IROW,ICOL)
NE=N-IR
IF(NE.EQ.0)GO TO 5
IF(IR)6,6,7
6 DO 8 I=1,NE
DO 9 J=1,M
IF(I.EQ.J)GO TO 10
XE(I,J)=0
GO TO 9
10 XE(I,J)=1.
9 CONTINUE
8 CONTINUE
GO TO 5
C.... RANK NOT 0,DIMENSION OF NULL SPACE LESS THAN ORDER OF MATRI)
7 II1=1
CALL ARRAY(II1,M,N,II2,II2,E,E)
DO 4 J=1,NE
DO 2 K=1,M
2 EN(K)=0.
LI=ICOL(IR+J)
EN(LI)=1
DO 3 L=1,IR
LE=ICOL(L)
3 EN(LE)=E(L,IR+J)
DO 4 I=1,M
4 XE(I,J)=EN(I)
5 RETURN
END
C.............................................................
C
C
C
C....  SUBROUTINE ARRAY
```

```
C      ***************
      SUBROUTINE ARRAY(MODE,I,J,N,M,S,D)
C.....CONVERT ARRAY FROM SINGLE TO DOUBLE DIMENSION OR V.V.
C.... FOR MODE=1 OR 2 RESP.
      DIMENSION S(1),D(1)
      NI=N-I
C.....TEST TYPE OF CONVERSION
      IF(MODE-1)100,100,120
C.... CONVERT FROM SINGLE TO DOUBLE PRECISION
  100 IJ=I*J+1
      NM=N*J+1
      DO 110 K=1,J
      NM=NM-NI
      DO 110 L=1,I
      IJ=IJ-1
      NM=NM-1
  110 D(NM)=S(IJ)
      GO TO 140
C.....CONVERT FROM DOUBLE TO SINGLE DIMENSION
  120 IJ=0
      NM=0
      DO 130 K=1,J
      DO 125 L=1,I
      IJ=IJ+1
      NM=NM+1
  125 S(IJ)=D(NM)
  130 NM=NM+NI
  140 RETURN
      END
C ..............................................................................
C
C
C
C.....SUBROUTINE MFGR
C     **************
      SUBROUTINE MFGR(A,M,N,EPS,IRANK,IROW,ICOL)
C.....DETERMINATION OF THE FOLLOWING FOR A M X N MATRIX A
C.....1  RANK AND LINEARLY INDDEPENDANT ROWS AND COLUMNS
C.....2  FACTORISATION OF SUBMATRIX OF MAX. RANK
C.....3  NON-BASIC ROWS IN TERMS OF BASIC ONES
C.....4 BASIC VARIABLES IN TERMS OF FREE ONES
C.... GAUSSIAN ELIMINATION WITH FULL PIVOTING
      DIMENSION A(1),IROW(1),ICOL(1)
C.... INITIALIZE COLUMN INDEX VECTOR  SEARCH FIRST PIVOT ELEMENT
    4 IRANK=0
      PIV=0.
      JJ=0
      DO 6 J=1,N
      ICOL(J)=J
      DO 6 I=1,M
      JJ=JJ+1
      HOLD=A(JJ)
      IF(ABS(PIV)-ABS(HOLD))5,6,6
    5 PIV=HOLD
      IR=I
      IC=J
    6 CONTINUE
C.... INITIALIZE ROW INDEX VECTOR
      DO 7 I=1,M
    7 IROW(I)=I
C.....SET UP INTERNAL TOLERANCE
      TOL=ABS(EPS*PIV)
C.... INITIALIZE ELIMINATION LOOP
      NM=N*M
      DO 19 NCOL=M,NM,M
C.....TEST FOR FEASIBILITY OF PIVOT ELEMENT
    8 IF(ABS(PIV)-TOL)20,20,9
C.... UPDATE RANK
    9 IRANK=IRANK+1
C.... INTERCHANGE ROWS IF NECESSARY
      JJ=IR-IRANK
      IF(JJ)12,12,10
   10 DO 11 J=IRANK,NM,M
      I=J+JJ
      SAVE=A(J)
      A(J)=A(I)
   11 A(I)=SAVE
C.....UPDATE ROW INDEX VECTOR
      JJ=IROW(IR)
      IROW(IR)=IROW(IRANK)
      IROW(IRANK)=JJ
C.....INTERCHANGE COLUMNS IF NECESSARY
   12 JJ=(IC-IRANK)*M
      IF(JJ)15,15,13
   13 KK=NCOL
```

```
      DO 14 J=1,M
      I=KK+JJ
      SAVE=A(KK)
      A(KK)=A(I)
      KK=KK-1
   14 A(I)=SAVE
C ... UPDATE COLUMN INDEX VECTOR
      JJ=ICOL(IC)
      ICOL(IC)=ICOL(IRANK)
      ICOL(IRANK)=JJ
   15 KK=IRANK+1
      MM=IRANK-M
      LL=NCOL+MM
      IF(MM)16,25,25
C.... TRANSFER CURRENT SUBMATRIX AND SEARCH FOR NEXT PIVOT
   16 JJ=LL
      SAVE=PIV
      PIV=0.
      DO 19 J=KK,M
      JJ=JJ+1
      HOLD=A(JJ)/SAVE
      A(JJ)=HOLD
      L=J-IRANK
C.....TEST FOR LAST COLUMN
      IF(IRANK-N)17,19,19
   17 II=JJ
      DO 19 I=KK,N
      II=II+M
      MM=II-L
      A(II)=A(II)-HOLD*A(MM)
      IF(ABS(A(II))-ABS(PIV))19,19,18
   18 PIV=A(II)
      IR=J
      IC=I
   19 CONTINUE
C.....SET UP MATRIX EXPRESSING ROW DEPENDANCIES
   20 IF(IRANK-1)3,25,21
   21 IR=LL
      DO 24 J=2,IRANK
      II=J-1
      IR=IR-M
      JJ=LL
      DO 23 I=KK,M
      HOLD=0.
      JJ=JJ+1
      MM=JJ
      IC=IR
      DO 22 L=1,II
      HOLD=HOLD+A(MM)*A(IC)
      IC=IC-1
   22 MM=MM-M
   23 A(MM)=A(MM)-HOLD
   24 CONTINUE
   25 IF(N-IRANK)3,3,26
C.....SET UP MATRIX EXPRESSING BASIC VARIABLES IN TERMS OF FREE
C ... PARAMETERS (HOMOGENEOUS SOLUTION)
   26 IR=LL
      KK=LL+M
      DO 30 J=1,IRANK
      DO 29 I=KK,NM,M
      JJ=IR
      LL=I
      HOLD=0.
      II=J
   27 II=II-1
      IF(II)29,29,28
   28 HOLD=HOLD-A(JJ)*A(LL)
      JJ=JJ-M
      LL=LL-1
      GO TO 27
   29 A(LL)=(HOLD-A(LL))/A(JJ)
   30 IR=IR-1
    3 RETURN
      END
C...........................................................................
C
C
C
C.... SUBROUTINE INDEP
C     *****************
C..... TO DETERMINE WHETHER VECTORS GENERATING THE NULL SPACE OF
C ...  (A-LI)**K CAN BE GENERALISED E-VECTORS
      SUBROUTINE INDEP(K,EE,NE,KPOINT,IIDIM)
      INTEGER RDR,PRT,DSK,DTP
      COMMON NRET,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
```

```
      COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
      COMMON ITYPE,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
      COMMON XE(20,20),BB(20,10),CC(10,20),NDIM,IC,ICOMPL,JDIM(20),KIND
      COMMON JSIGN,MULT(20),WR(20),WI(20),UU(20,20),VI(20,20),NDIAG(20)
      DIMENSION E4(20,20),U(20,1),UR(20,1),EE(20,20)
      EQUIVALENCE (CC(1,1),E4(1,1))
      IFIN=NSUM
      IID1=20
      IID2=1
      IF(K.LE.2)GO TO 231
      LL=1
C.... CHECK IF (A-LI)**K-1 = 0
      DO 213 J=1,NE
      DO 214 I=1,IFIN
  214 U(I,1)=XE(I,J)
      CALL MATRM(IID1,IID2,IFIN,IFIN,IID2,EE,U,UR)
      DO 215 I1=1,IFIN
      IF(ABS(UR(I1,1)).LE.EPS)GO TO 215
      GO TO 213
  215 CONTINUE
      KINDEX(LL)=J
      LL=LL+1
  213 CONTINUE
      IF(LL.EQ.1)GO TO 216
      IF(NE.EQ.1)GO TO 220
C.....DELETE UNSUITABLE E-VECTORS FROM E-VECTOR MATRIX XE
      DO 217 I=1,LL-1
      LX=KINDEX(I)-I+1
      IF(LX.EQ.NE)GO TO 220
      DO 218 I1=1,IFIN
      DO 219 I2=LX,NE-1
  219 XE(I1,I2)=XE(I1,I2+1)
  218 CONTINUE
  220 NE=NE-1
  217 CONTINUE
C.... PICK THE LEADING E-VECTORS ALREADY OBTAINED FROM UU
  216 IF(NE.LE.1)GO TO 210
  231 KJ=IIDIM-KPOINT
      IF(KJ)696,696,695
  695 KSTART=1
      IF(KPOINT.EQ.1)GO TO 698
      DO 699 I=1,KPOINT-1
  699 KSTART=KSTART+JDIM(I)
  698 KJJ=KSTART
      DO 701 I=1,KJ
      DO 700 J=1,IFIN
      E4(J,I)=UU(J,KJJ)
  700 CONTINUE
      KJJ=KJJ+JDIM(I)
  701 CONTINUE
  696 I11=NE
      I10=1
      INE1=KJ+1
      INE2=1
  278 IF(K.GT.2)GO TO 228
      DO 262 I=1,IFIN
  262 U(I,1)=XE(I,INE2)
      GO TO 263
C.....GENERATE PRESENT LEADING E-VECTORS AND CHECK IF THEY FORM
C.... A L.I. SET WITH THE ALREADY OBTAINED E-VECTORS
  228 DO 241 I=1,IFIN
  241 UR(I,1)=XE(I,INE2)
      CALL MATRM(IID1,IID2,IFIN,IFIN,IID2,EE,UR,U)
  263 II=INE1
      DO 244 I2=1,IFIN
  244 E4(I2,II)=U(I2,1)
      IF(II.LE.1)GO TO 224
      CALL RRANK(E4,EPS,IFIN,II,IRAN)
      IF(IRAN.EQ.II)GO TO 224
      IF(INE1.EQ.(NE+KJ))GO TO 225
      DO 226 I3=1,IFIN
      DO 227 I4=1,NE-1
  227 XE(I3,I4)=XE(I3,I4+1)
  226 CONTINUE
  225 NE=NE-1
      GO TO 255
  224 INE1=INE1+1
      INE2=INE2+1
  255 I10=I10+1
      IF(I10.LE.I11)GO TO 278
  210 RETURN
      END
C.... .
```

REFERENCES

[1] M. Heymann and J. A. Thorpe, "Transfer equivalence of linear dynamical systems " , SIAM J. Control, vol. 8, pp. 19 - 40, 1970.

[2] R. E. Kalman, " Irreducible realisations and the degree of a rational matrix " , J. SIAM, vol. 13, pp. 520 - 544, June 1965.

[3] B. L. Ho and R. E. Kalman, " Effective construction of linear state variable models from input/output data ", Proc. 3rd Ann. Allerton Conf. on Circuit and System Theory, pp. 449 - 459.

[4] C. T. Chen and C. A. Desoer, " Controllability and observability of composite systems ", IEEE Trans. on Automatic Control, vol. AC-12, pp. 402-409, August 1959.

[5] Y. L. Kuo, " On the irreducible Jordan form realisation and the degree of a rational matrix ", IEEE Trans. on Circuit Theory, vol. CT-17, August 1970.

[6] Gueguen C. J. and E. Toumire, " Comments on Irreducible Jordan form realisation of a rational matrix ", IEEE Trans. on Automatic Control, vol. AC-15, 1970.

[7] Panda S. P. and C. T. Chen, "Irreducible Jordan form realisation of a rational matrix ", IEEE Trans. on Automatic Control, Feb. 69, pp. 66-69.

[8] H. H. Rosenbrock, " Computer aided control system design ", pp. 14-17.

Acknowledgements

The author wishes to thank his supervisor M. J. Denham, for his most helpful assistance through the writing of this paper and Mrs. H. Pouliezou - Mekali for patiently typing it.