



Imperial College

Department of
Computing and
Control

Implementation 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 Technology.

September 1976.

CONTENTS

Abstract	1
Introduction	1
CHAPTER I Algebraic Preliminaries	
1.1 E-values and e-vectors.....	4
1.2 Jordan form of a matrix	4
1.3 Generalised e-vectors	6
1.4 Procedure for calculating the Jordan form representation	8
CHAPTER II The Algorithm	
2.1 Introductory concepts	10
2.1.2 Observability and controllability conditions	12
2.2 The algorithm	16
2.2.1 Computational algorithm	21
2.2.2 Systems with complex poles	23
CHAPTER III The Program	
3.1 General	25
3.2 Sections of program	25
3.3 Description of main program and subroutines	26
3.4 Calculations	28
3.5 Input descriptions	29
3.5.1 Output descriptions	30
3.5.2 Error messages	32
3.6 Subroutine flowchart	33

CHAPTER IV Tests

4.1	Outline of tests	34
4.2	Test results	
	Tests 1 - 10	35
4.3	Conclusions and possible improvements	93
APPENDIX I	95
References	127
Acknowledgments	129

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 procedure.

Particular emphasis is given in comparing this algorithm with Rosenbrock's, which is currently in use at the Control Department of 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 $C^{n \times n}$. Then a scalar $\lambda \in C$ 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 $C^{n \times n}$ there exists a transformation matrix Q in $C^{n \times n}$, such that

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

and J is in the form :

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

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

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

We say that J is the Jordan canonical form of the matrix A .
 The distinct e-values of A are $\lambda_1, \lambda_2, \dots, \lambda_q$ and the multiplicity of λ_1 is $m_1 + m_2 + \dots + m_s$ and so on. The numbers m_1, m_2, \dots, t_q are determined by the number of linearly independent e-vectors per e-value and the dimension of the e-space.
 The set $\left[(m_1, m_2, \dots, m_s), (n_1, \dots, n_t), \dots, (t_1, \dots, t_q) \right]$ 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 $i = 1, 2, \dots, k$ and $\underline{v}^j = (A - \lambda I)^{l-j} \underline{v}$ for $j = 1, 2, \dots, l$. If the two vectors \underline{u}^1 and \underline{v}^1 are linearly independent, then the generalised e-vectors $\underline{u}^1, \underline{u}^2, \dots, \underline{u}^k, \underline{v}^1, \underline{v}^2, \dots, \underline{v}^l$ are linearly independent.

Proof : We first prove that the sets $(\underline{u}^1, \underline{u}^2, \dots, \underline{u}^k), (\underline{v}^1, \underline{v}^2, \dots, \underline{v}^l)$ 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 $\beta_1, \beta_2, \dots, \beta_k$ not all zero, such that

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

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

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

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

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

$$\beta_1 = \beta_2 = \dots = \beta_k = 0$$

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

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,

$$\sum_{i=1}^k \beta_i \underline{u}^i = \sum_{j=1}^l \beta'_j \underline{v}^j , \text{ where the } \beta_i \text{'s and } \beta'_j \text{'s are not}$$

all zero. Then

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

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

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

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

One step before

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

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

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

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 $i= 1, 2, \dots$ 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 $i=1,2,\dots,k$. 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 $\text{rank}(A - \lambda_1 I) = a$, then there are totally $(n_1 - a)$ chains of generalised e-vectors associated with λ_1 .

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

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

$$Q = \begin{bmatrix} \underline{u}_1^1 & \dots & \underline{u}_1^{k_1} & \underline{v}_1^1 & \dots & \underline{v}_1^{l_1} & \dots & \dots & \underline{u}_2^1 & \dots & \underline{u}_2^{k_2} & \underline{v}_2^1 & \dots & \underline{v}_2^{l_2} & \dots & \dots & \dots & \underline{u}_m^1 & \dots & \underline{u}_m^{k_m} & \underline{v}_m^1 & \dots & \underline{v}_m^{l_m} \end{bmatrix}$$

CHAPTER II

The Algorithm

2.1 Introductory Concepts

Let $G(s)$ be a transfer function matrix of dimensions $m \times l$ in the following form :

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

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

$$\left. \begin{aligned} n_{ij}(s) &= p_q^{ij}s^q + p_{q-1}^{ij}s^{q-1} + \dots + p_0^{ij} \\ d_i(s) &= s^t + r_{t-1}^i s^{t-1} + \dots + r_0^i \end{aligned} \right\} \begin{aligned} i &= 1, \dots, m \\ j &= 1, \dots, l \end{aligned}$$

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

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

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

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

and $\bar{A}_i = \begin{bmatrix} 0 & 0 & \dots & -r_{p_i}^i \\ 1 & 0 & \dots & -r_{p_i}^i \\ \dots & \dots & \dots & \dots \\ 0 & \dots & 1 & -r_{t-1}^i \end{bmatrix} \quad i = 1, \dots, m$

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

$$\bar{C} = \begin{bmatrix} \bar{C}_1 & \bar{C}_2 & \dots & \bar{C}_m \end{bmatrix} \quad \text{and} \quad \bar{C}_i = \begin{bmatrix} \vdots & 0 \\ \vdots & \vdots \\ 0 & \vdots \\ \vdots & 1 \\ \vdots & \vdots \\ \vdots & 0 \end{bmatrix} \quad \text{in } i\text{th position}$$

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

$$A = Q\bar{A}Q^{-1}$$

$$B = Q^{-1}\bar{B}$$

$$C = \bar{C}Q \quad \dots \dots \dots (1)$$

Let the distinct e-values of \bar{A} be $\lambda_1, \lambda_2, \dots, \lambda_s$. 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 $i = 1, 2, \dots, s$ the set of $f(i)$ 1-dimensional row vectors

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

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,

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

since,

$$A = \begin{matrix} \text{nxn} \\ \left[\begin{array}{cccc} A_1 & & & \\ & A_2 & & \\ & & \dots & \\ & & & A_s \end{array} \right] \end{matrix} \quad B = \begin{matrix} \text{nx1} \\ \left[\begin{array}{c} B_1 \\ B_2 \\ \vdots \\ B_s \end{array} \right] \end{matrix}$$

$$C = \begin{matrix} \text{mxn} \\ \left[C_1 \quad C_2 \quad \dots \quad C_s \right] \end{matrix}$$

.....

$$A_i = \begin{matrix} \text{k}_i \times \text{k}_i \\ \left[\begin{array}{cccc} A_{i1} & & & \\ & A_{i2} & & \\ & & \dots & \\ & & & A_{if(i)} \end{array} \right] \end{matrix} \quad B_i = \begin{matrix} \text{k}_i \times 1 \\ \left[\begin{array}{c} B_{i1} \\ B_{i2} \\ \vdots \\ B_{if(i)} \end{array} \right] \end{matrix}$$

$$C_i = \begin{matrix} \text{mxk}_i \\ \left[C_{i1} \quad C_{i2} \quad \dots \quad C_{if(i)} \right] \end{matrix}$$

.....

$$A_{ij} = \begin{matrix} \text{k}_{ij} \times \text{k}_{ij} \\ \left[\begin{array}{cccc} \lambda_i & 1 & 0 & \dots & 0 \\ 0 & \lambda_i & 1 & \dots & 0 \\ \vdots & & & \dots & \\ 0 & & & & \lambda_i \end{array} \right] \end{matrix} \quad B_{ij} = \begin{matrix} \text{k}_{ij} \times 1 \\ \left[\begin{array}{c} \underline{b}_{1.}^{ij} \\ \underline{b}_{2.}^{ij} \\ \vdots \\ \underline{b}_{\text{k}_{ij}.}^{ij} \end{array} \right] \end{matrix}$$

$$C_{ij} = \begin{matrix} \text{mxk}_{ij} \\ \left[\underline{c}_{.1}^{ij} \quad \underline{c}_{.2}^{ij} \quad \dots \quad \underline{c}_{.\text{k}_{ij}}^{ij} \right] \end{matrix}$$

FIGURE 1

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

Necessity : Observe that the last row of $e^{A_{ij}t} B_{ij}$ is $\frac{b_{ij}^{ij}}{k_{ij}^{ij}} e^{\lambda_i t}$. Hence if $\frac{b_{i1}^{i1}}{k_{i1}^{i1}}$, $\frac{b_{i2}^{i2}}{k_{i2}^{i2}}$, ..., $\frac{b_{if(i)}^{if(i)}}{k_{if(i)}^{if(i)}}$ are linearly independent

then the last row of $e^{A_{ij}t} B_{ij}$, $j = 1, 2, \dots, f(i)$ is linearly independent.

Sufficiency : Having in mind that,

$$e^{A_{ij}t} B_{ij} = e^{\lambda_i t} \begin{bmatrix} 1 & t & \frac{1}{2}t^2 & \dots & \frac{1}{(k_{ij}-1)!}t^{k_{ij}-1} \\ 0 & 1 & t & \dots & \frac{1}{(k_{ij}-2)!}t^{k_{ij}-2} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} \begin{bmatrix} \frac{b_{i1}^{ij}}{k_{i1}^{ij}} \\ \frac{b_{i2}^{ij}}{k_{i2}^{ij}} \\ \dots \\ \frac{b_{if(i)}^{ij}}{k_{if(i)}^{ij}} \end{bmatrix} \quad [12]$$

it is clear that if $\frac{b_{ij}^{ij}}{k_{ij}^{ij}} \neq 0$, then all the k_{ij} rows in $e^{A_{ij}t} B_{ij}$ are linearly independent. By hypothesis $\frac{b_{i1}^{i1}}{k_{i1}^{i1}}$, $\frac{b_{i2}^{i2}}{k_{i2}^{i2}}$, ..., $\frac{b_{if(i)}^{if(i)}}{k_{if(i)}^{if(i)}}$ 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 $p_i(\cdot)$ and $p_j(\cdot)$ 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 representantion S is observable if and only if, for each $i = 1, 2, \dots, s$ the set of $f(i)$ m-dimensional column vectors

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

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,

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

$$A_j = \begin{bmatrix} \lambda_i & 1 & \dots & 0 \\ 0 & \lambda_i & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_i \end{bmatrix}$$

$k_j \times k_j$

$$B = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_r \end{bmatrix} \quad C = [C_1 \ C_2 \ \dots \ C_r]$$

B_j is $k_j \times 1$ C_j is $m \times k_j$

We will denote by b_j^i the rows of B_i and by $c_{.j}^i$ the columns of C_i , for $j = 1, \dots, k_i$ and $i = 1, \dots, r$.

Consider now two types of operation :

(i) Let $1 \leq i, j \leq r$, $k_j \geq k_i$ and B in C , ($i \neq j$)

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

$$b_k^i \rightarrow b_k^i + B b_{(k_j - k_i + k)}^j \quad k = 1, \dots, k_i$$

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

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

Now,

$$F' = \begin{bmatrix} I_{k_j} & \vdots & \\ \cdots & I_{k_1} & \cdots \\ \cdots & \vdots & \cdots \\ F & \vdots & I_{k_j} \\ \vdots & \vdots & \vdots \end{bmatrix}$$

in general, and hence

$$(F')^{-1} = \begin{bmatrix} I_{k_j} & \vdots & \\ \cdots & I_{k_1} & \cdots \\ \cdots & \vdots & \cdots \\ -F & \vdots & I_{k_j} \\ \vdots & \vdots & \vdots \end{bmatrix}$$

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

Then $C \rightarrow CT$ is the operation,

$$c^j \cdot (k_j - k_i + k) \rightarrow c^j \cdot (k_j - k_i + k) - Bc^i \cdot k \quad \text{for } k = 1, \dots, k_i$$

i.e. C_i multiplied by B 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_1 is zero for some $1 \leq l \leq r$

$$\text{i.e. } b_{k_1}^l = 0$$

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

$$= \sum_{\substack{i=1 \\ i \neq 1}}^r C_i (sI - A_i)^{-1} B_i + \sum_{j=1}^{k_1} \sum_{k=j}^{k_1} c_{.j}^1 b_k^1 \cdot \frac{1}{(s-\lambda)^{k-j+1}} \dots (3)$$

since,

$$C_1 (sI - A_1) B_1 = \begin{bmatrix} c_{.1}^1 & \dots & c_{.k_1}^1 \end{bmatrix} \begin{bmatrix} \frac{1}{s-\lambda} & \frac{1}{(s-\lambda)^2} & \dots & \frac{1}{(s-\lambda)^{k_1}} \\ 0 & \frac{1}{s-\lambda} & \dots & \frac{1}{(s-\lambda)^{k_1-1}} \\ \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \frac{1}{s-\lambda} \end{bmatrix} \begin{bmatrix} b_1^1 \\ b_2^1 \\ \cdot \\ b_{k_1}^1 \end{bmatrix}$$

(A) If $k_1 > 1$ then from (3),

$$G(s) = \sum_{\substack{i=1 \\ i \neq 1}}^r C_i (sI - A_i)^{-1} B_i + C'_1 (sI - A'_1)^{-1} B'_1$$

where A'_1 is a $(k_1-1) \times (k_1-1)$ Jordan block which is obtained by deleting the last row and column from A_1 and B'_1 , C'_1 are obtained by deleting the last row and column from B_1 , C_1 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,

$$A' = \begin{bmatrix} A_1 & & & & \\ & \ddots & & & \\ & & A_{l-1} & & \\ & & & A'_1 & \\ & & & & A_{l+1} \\ & & & & & \ddots \\ & & & & & & A_r \end{bmatrix}$$

$$C' = \begin{bmatrix} C_1 & \dots & C_{l-1} & C'_1 & C_{l+1} & \dots & C_r \end{bmatrix}$$

$$B' = \begin{bmatrix} B_1 \\ \vdots \\ B_{l-1} \\ B'_1 \\ B_{l+1} \\ \vdots \\ B_r \end{bmatrix}$$

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

(B) $k_1 = 1$

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

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

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

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 dependence 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 $V(1), \dots, V(r)$ by setting $V(i) = 0$ for $i = 1, \dots, r$
- (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

$$k_i = 1 \leq \max_{j \leq r} \left\{ k_j : V(j) = 0 \right\} .$$
- (iv) If $b_{k_i}^i = 0$ (i.e. the last row of B_i) go to (vi).
- (v) Set $V(i) = 1$

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

$$\frac{b_{k_h, j}^h}{b_{k_i, j}^i} \times b_{k_i - k_j + k_h}^i \text{ from } b_{k_h}^h \text{ for } k = 1, \dots, k_h$$

and add

$$\frac{b_{k_h, j}^h}{b_{k_i, j}^i} \times c_{.k}^h \text{ to } c_{.k_i - k_j + k} \text{ for } k = 1, \dots, k_h$$

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

Go to (ii).

(vi) Delete the last row from B_i (i.e. $b_{k_i}^i$), the last column from C_i (i.e. $c_{.k_i}^i$) 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 , $i = 1, \dots, r$ 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 a row 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 $b_{k_i}^i \neq 0$ 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 a complex 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,

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

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,

$$A' = \begin{bmatrix} A & \\ & \bar{A} \end{bmatrix} \quad B' = \begin{bmatrix} B \\ \bar{B} \end{bmatrix} \quad C' = \begin{bmatrix} C & \bar{C} \end{bmatrix}$$

$2n \times 2n$ $2n \times 1$ $m \times 2n$

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

$$\text{If } Q = \begin{bmatrix} iI & I \\ -iI & I \end{bmatrix} \quad \text{then } Q^{-1} = \begin{bmatrix} -\frac{1}{2}iI & \frac{1}{2}iI \\ \frac{1}{2}I & \frac{1}{2}I \end{bmatrix}$$

$2n \times 2n$

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

then,

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

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

where $\text{Im}(\cdot)$ is the imaginary part of a complex matrix and $\text{Re}(\cdot)$ 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 necessary 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 $|a-b| \leq \text{EPS}$.

Given a transfer function matrix $G(s)$ of order $m \times n$, 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.

$$A = \text{diag} [A_1, A_2, \dots, A_m]$$

and

$$A_i = \begin{bmatrix} 0 & 0 & \dots & 0 & -\beta_0^i \\ 1 & 0 & \dots & 0 & -\beta_1^i \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \dots & 1 & -\beta_{q_i}^i \end{bmatrix}$$

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,

$$s^{q_i} + \beta_{q_i}^i s^{q_i-1} + \dots + \beta_0^i = 0, \quad i = 1, \dots, m$$

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 : 20×20 , B : 20×10 , C : 10×20 . The maximum numerator order is 20, the maximum denominator order 20, and the maximum order of G : 10×10 .

The input description is as follows (see examples):

After typing the 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 $n \times m$ is typed in in the appropriate format.

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

?

The orders 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 starts with,

OBSERVABLE REALISATION A B C

and lists the three (or four if $D \neq 0$) 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)
-1.000000000000	0	-1.000000000000	1.000000000000
-1.000000000000	0	-1.000000000000	-1.000000000000

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

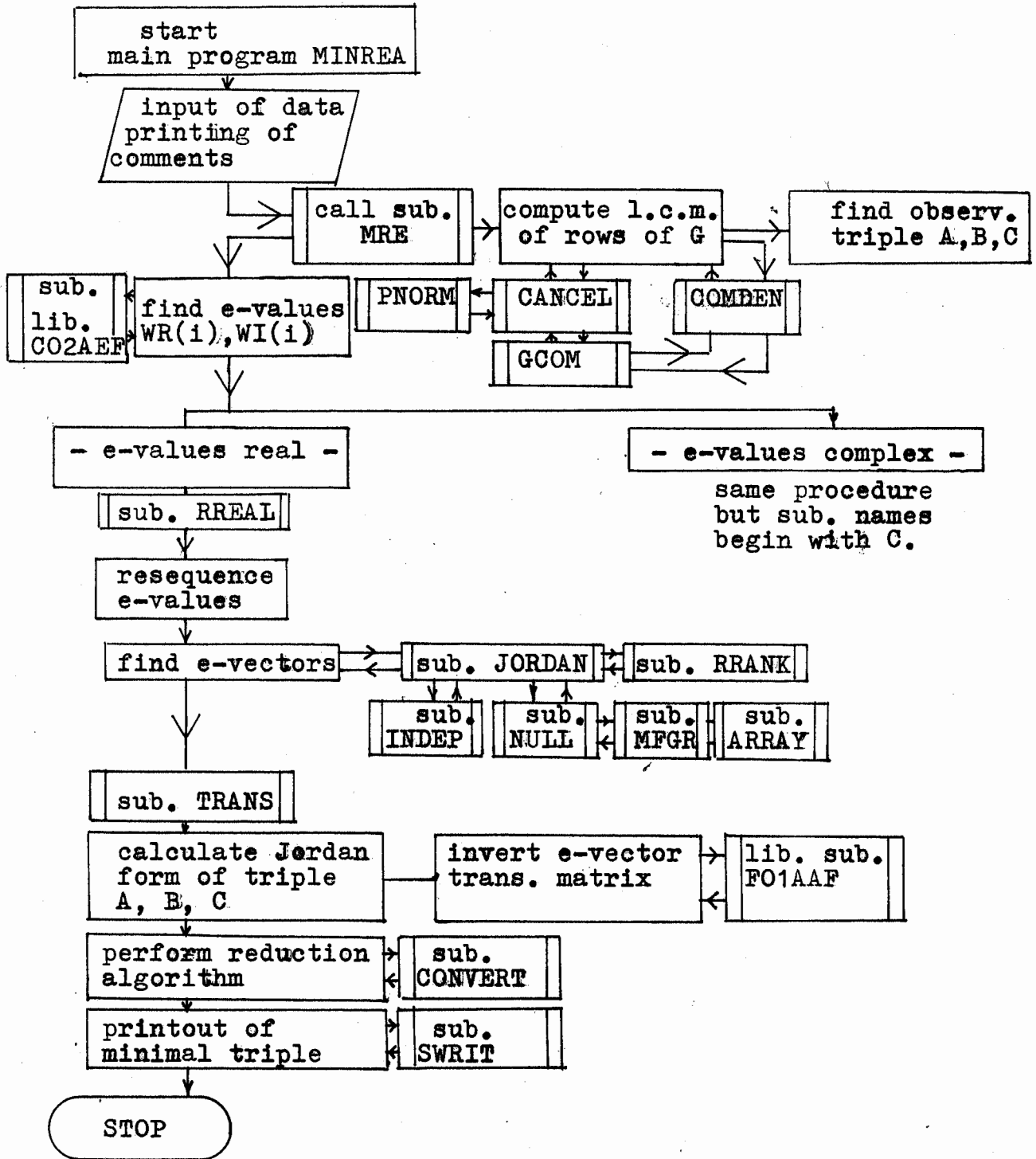
if transformation matrix Q singular.

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

$$G(s) = \begin{bmatrix} \frac{1}{s+2} & \frac{2(2s+5)}{(s+2)(s+3)} \\ \frac{2}{s+3} & \frac{4s+22s+29}{(s+2)(s+3)} \end{bmatrix}$$

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

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

$$= \begin{bmatrix} 0 & \frac{2.5}{s+2} & 0 & \frac{.333}{s+2} \\ \frac{1.25}{s+3} & \frac{1.25}{s+3} - \frac{.25}{s+3} & \frac{.111}{s+2} & 0 \end{bmatrix} \begin{bmatrix} 1.6 & 2.56 \\ 0 & .8 \\ 0 & 9. \\ 3 & 6 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{3 \times .633}{s+2} & \frac{2.5 \times .8}{s+3} + \frac{.333 \times 6}{s+2} \\ \frac{1.25 \times 1.6}{s+3} - \frac{2.56 \times 1.25}{s+3} + \frac{1.25 \times .8}{(s+3)} - \frac{.25 \times .8}{s+3} + \frac{9 \times .111}{s+2} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{s+2} & \frac{2}{s+3} + \frac{2}{s+2} \\ \frac{2}{s+3} & \frac{3.2}{s+3} + \frac{1}{(s+3)} - \frac{.2}{s+3} + \frac{1}{s+2} \end{bmatrix} = \begin{bmatrix} \frac{1}{s+2} & \frac{2(2s+5)}{(s+3)(s+2)} \\ \frac{2}{s+3} & \frac{3}{s+3} + \frac{1}{(s+3)} + \frac{1}{s+2} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{s+2} & \frac{2(2s+5)}{(s+3)(s+2)} \\ \frac{2}{s+3} & \frac{4s+22s+29}{(s+3)(s+2)} \end{bmatrix}$$

RNH:MI=14000

***** TEST RUN FOR MINIMAL REALISATION USING ROSENBRCK'S ALGORITHM
*** START INPUT OF DATA ***

TOLEANCE

? .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 DENOMINATOR

? 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	-5.336	.029
-6.000	6.000	5.455	-2.483

B MATRIX

-.000	0
0	0
-1.138	.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 212

? 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 DENOMINATOR

? 19.	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

-3.000000000000	0	-2.000000000000	0
-3.00000023842	0	-2.99999976158	0
-2.000000000000	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	-.950	2.050	-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
 1.500 2.560
 .000 .800

 - .000 9.000
 3.000 6.000

.....last row 0, delete

TRANSFORMED B
 1.500 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

1.500	2.560
.000	.800
0	9.000
3.000	6.000

C MATRIX

0	2.250	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.

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

*** START INPUT OF DATA ***
TOLERANCE

.000005

ORDER OF MATRIX - FORMAT 212

2 2

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

2 2 2 2

ORDER OF DENOMINATOR

3

COEFFICIENTS OF NUMERATOR POLYNOMIALS

9.	5.	1.
30.	20.	4.
12.	10.	2.
29.	22.	4.

COEFFICIENTS OF DENOMINATOR

19. 21. 3. 1.

*** INPUT OF DATA FINISHED ***

DIMENSION 4

A MATRIX

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

B MATRIX

1.500	2.500
.000	1.500
0	2.000
3.000	5.000

C MATRIX

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

STOP

<TIME

CTIME. 21.231 SEC.

TEST 2

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

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

$$= \begin{bmatrix} 0.5 & -0.5 & 0.33 \\ 1 & -0.5 & -0.33 \end{bmatrix} \begin{bmatrix} \frac{1}{s+2} \\ \frac{1}{s+1} \\ \frac{1}{s-1} \end{bmatrix} \begin{bmatrix} 6.6667 & 4 \\ 7 & 4 \\ 3.5 & 3 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{0.5}{s+2} & -\frac{0.5}{s+1} & \frac{0.33}{s-1} \\ \frac{1}{s+2} & -\frac{0.5}{s+1} & -\frac{0.33}{s-1} \end{bmatrix} \begin{bmatrix} 6.6667 & 4 \\ 7 & 4 \\ 3.5 & 3 \end{bmatrix}$$

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

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

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

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

RNA:MI=14000

**** TEST RUN FOR MINIMAL REALISATION USING ROSENBRACK'S ALGORITHM
*** START INPUT OF DATA ***

TOLERANCE

.000000000005

ORDER OF MATRIX -FORMAT 212

? 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
-1.000	-.200	-.950
.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 212

? 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

C MATRIX

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

E-VALUES

-2.000000000000	0	1.000000000000	0
-1.000000000000	0	-2.000000000000	0
1.000000000000	0	-1.000000000000	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
0	-2.000	0	0	0	0	0
0	0	-1.000	0	-1.000	0	0
0	0	0	-1.000	0	-1.000	0
0	0	0	0	1.000	0	0
0	0	0	0	0	1.000	0

TRANSFORMED B

0	0
5.667	4.000
7.000	4.000
7.000	4.000
-3.500	-3.000
3.500	3.000

.....row corr. to first block of first e-value 0; delete.

TRANSFORMED B

0	0
7.000	4.000
-3.500	-3.000
3.500	3.000
3.500	3.000

.....row corr. to first blok of second e-value 0 ; delete .

TRANSFORMED B

0	0
3.500	3.000
3.500	3.000

.....row corr. to first block of third e-value 0 ; delete .

DIMENSION 3

A MATRIX

-2.000	0	0
0	-1.000	-1.000
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 .

ILLEGAL COMMAND.

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

*** START INPUT OF DATA ***
TOLERANCE

.000000000005
ORDER OF MATRIX - FORMAT 212
2 2

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW
2 2 2 2

ORDER OF DENOMINATOR
3

COEFFICIENTS OF NUMERATOR POLYNOMIALS

3.	0.	1.
4.	1.	1.
-9.	-7.	2.
13.	-5.	1.

COEFFICIENTS OF DENOMINATOR

-2. 1. 2. 1.

*** INPUT OF DATA FINISHED ***

DIMENSION 3

A MATRIX

-2.000	0	0
0	-1.000	-4.000
0	0	1.000

B MATRIX

6.667	4.000
7.000	4.000
3.500	3.000

C MATRIX

.500	-.500	.999
1.000	-.500	-.999

STEP
%TIME
%TIME. 22.372 SEC.

Test 3

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

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.)

PNH:MI=14000

***** TEST RUN FOR MINIMAL REALISATION USING ROSENBRACK'S ALGORITHM

*** START INPUT OF DATA ***

TOLERANCE

? .000000000005

ORDER OF MATRIX -FORMAT 212

? 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.	0.	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	.648	.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	-.500	.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
-1.000	0	0
0	0	0
0	0	0
-1.000	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.

L60

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

*** START INPUT OF DATA ***

TOLERANCE

.000000000005

ORDER OF MATRIX - FORMAT 212

? 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.	-9.	1.
? 1.	0.	-1.	
? -2.	-1.	0.	1.

COEFFICIENTS OF DENOMINATOR

? 0. 0. 0. 0. 1.

*** INPUT OF DATA FINISHED ***

OBSERVABLE REALISATION A B C
A MATRIX

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	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	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	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	0	0

1.000
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

INVERSE OF TRANSFORMATION MATRIX

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

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

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
1.000	1.000	-2.000
1.000	1.000	-2.000

last rows of blocks of B not linearly independent;reduction possible.

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
21.500	2.500	-1.000

.....row 0 ; delete.

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
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

.....row 0 ; delete.

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

.....rows now independent, dimension 8.

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

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

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

*** START INPUT OF DATA ***
TOLERANCE

0.0000000000000005
ORDER OF MATRIX - FORMAT 212
3 3

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

ORDER OF DENOMINATOR
1

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 3

A MATRIX

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	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

B MATRIX

0	0	3.000
21.500	2.500	-1.000
11.500	1.000	2.500
0	-1.500	4.300
1.000	0	1.000
-2.000	-1.000	0
-1.000	0	-1.000
1.000	1.000	-2.000

C MATRIX

1.000	.657	.657	-5.989	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
TIME 25.097 SEC.

Test 4

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

TIME = 14000

*** TEST RUN FOR MINIMAL REALISATION USING ROSENBRACK'S ALGORITHM ***
*** START INPUT OF DATA ***

TOLERANCE

? .000000000005

ORDER OF MATRIX -FORMAT 212

? 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. 8. 1.

*** INPUT OF DATA FINISHED ***

DIMENSION 3

A MATRIX

-4.500 .313 0
-4.000 .417 -1.960
-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.

7L30

*****TEST RUN FOR MINIMAL REALISATION USING THE JORDAN CANONICAL F

*** START INPUT OF DATA ***

TOLERANCE

? .0000000000005

ORDER OF MATRIX -FORMAT 212

? 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 -5.000

B MATRIX

1.000
3.500
.500

C MATRIX

0 0 1.000

E-VALUES

-4.000000000000 0 -1.000000000000 -1.000000000000
-1.000000000000 1.000000000000

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 -1.500 -.300

TRANSFORMED B MATRIX

-.200 .000
.500 -1.300
.500 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 -1.000 -1.000
-.000 1.000 -1.000

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

B MATRIX

-.200
-1.300
.500

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

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

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

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

IDLE.

NUM, NI=14000

*** TEST RUN FOR MINIMAL REALISATION USING ROSENBRACK'S ALGORITHM ***
*** START INPUT OF DATA ***

TOLEANCE

? .000000000005
ORDER OF MATRIX -FORMAT 212
? 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. 3.
? 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.

LSO

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

*** START INPUT OF DATA ***
TOLERANCE

? .000000000005
ORDER OF MATRIX - FORMAT 212
? 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

B MATRIX
0 0 1.000 -1.000

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

-1.000000000000 0 -1.000000000000 0
-1.000000000000 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 2
 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 A 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 B 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 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

/CTIME

CTIME. 17.390 SEC.

1.33

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

*** START INPUT OF DATA ***

TOLEBRANCE

.000000000005

ORDER OF MATRIX - FORMAT A12

2 2

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

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

SIZE

ATTIME

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

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

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

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

PARAM=14000

**** TEST RUN FOR MINIMAL REALISATION USING ROSENBRACK'S ALGORITHM
*** START INPUT OF DATA ***

TOLERANCE

? .0000000000005

ORDER OF MATRIX -FORMAT 212

? 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	-1.157	0
1.000	0	-1.157	.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.

L60

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

*** START INPUT OF DATA ***

TOLERANCE

? .0000000000005

ORDER OF MATRIX - FORMAT 312

? 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

B MATRIX

1.000	0
0	1.000
1.000	2.000
3.000	0

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	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 A MATRIX

1.000 2.000
0 1.000
1.000 0
1.000 2.000
3.000 0

TRANSFORMED I 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 B

1.000	2.000
0	1.000
1.000	0
1.000	-1.000
0	0

TRANSFORMED E

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	0	1.000	0		0
0	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
0	3.000	1.000	1.000		1.000

STOP

ACTIME

CTIME. 7.358 SEC.

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

*** START INPUT OF DATA ***

TOLERANCE

.0000000000005

ORDER OF MATRIX - FORMAT 212

2 2

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

2 2 2 2

ORDER OF DENOMINATOR

2

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

ELAPSE

TIME. 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 a common denominator with leading coefficient 1, could multiply the matrix.

**** TEST RUN FOR MINIMAL REALISATION USING ROSENBRGK'S ALGORITHM

*** START INPUT OF DATA ***

TOLERANCE

*
? .0000000000005

ORDER OF MATRIX -FORMAT 212

? 4 4

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

? 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 3

A MATRIX

-.200	-.000	.000	-.000	-.000	0	0	0
-.021	-.142	.022	-.021	.003	-.002	.000	0
-.003	-.009	-.210	-.032	-.000	.000	.002	0
-.011	-.001	-.035	-.166	.001	.001	-.002	.001
1.000	-1.457	-.292	-.491	-.297	-.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

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

-.250000000000	0	-.200000000000	0
-.250000000000	0	-.200000000000	0
-.250000000000	0	-.200000000000	0
-.250000000000	0	-.200000000000	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	0	0	0	-20.000	5.000	0	0	0
0	0	-20.000	5.000	0	0	0	0	0	0
0	0	0	0	0	0	0	-20.000	5.000	0
20.000	-4.000	0	0	0	0	0	0	0	0
0	0	0	0	0	20.000	-4.000	0	0	0
0	0	20.000	-4.000	0	0	0	0	0	0
0	0	0	0	0	0	0	20.000	-4.000	0

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

-0.250	0	0	0	0.000	0	0	0
0	-0.250	0	0	0	0.000	0	0
0	0	-0.250	0	0	0	0.000	0
0	0	0	-0.250	0	0	0	0.000
0.000	0	0	0	-0.200	0	0	0
0	0.000	0	0	0	-0.200	0	0
0	0	0.000	0	0	0	-0.200	0
0	0	0	0.000	0	0	0	-0.200

TRANSFORMED E

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

TRANSFORMED F

.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

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

.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

.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

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

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
.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 B

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 13.694 SEC.

133

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

*** START INPUT OF DATA ***

TOLERANCE

.000000000005

ORDER OF MATRIX - FORMAT 312

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

1 1 1 1 1 1 1 1 1 1 1 1

ORDER OF DENOMINATOR

2

COEFFICIENTS OF NUMERATOR POLYNOMIALS

1	.05	.28
2	.035	.14
3	.015	.06
4	.01	.04
5	.03	.12
6	.05	.25
7	.02	.08
8	.0175	.07
9	.0175	.07
10	.02	.08
11	.05	.25
12	.03	.12
13	.01	.04
14	.015	.06
15	.035	.14
16	.05	.28

COEFFICIENTS OF DENOMINATOR

.05 .45 1.

*** INPUT OF DATA FINISHED ***

DIMENSION 3

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	-.193	.110
.000	-.190	-.340	.070
.040	.080	.140	.000

C MATRIX

1.000	.000	.000	.000	1.000	1.064	-.776	.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

STD-

CTIME

CTIME.

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=18000

**** TEST RUN FOR MINIMAL REALISATION USING ROSENBRACK'S ALGORITHM
*** START INPUT OF DATA ***

TOLEARNCE

? .000000005

ORDER OF MATRIX -FORMAT 212

? 2 2

ORDER OF NUMERATOR POLYNOMIALS ROW BY ROW

? 2 2 2 3

ORDER OF DENOMINATOR

? 4

COEFFICIENTS OF NUMERATOR POLYNOMIALS

? 4.	4.	1.		
? 0.	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	-1.146	-1.000	0
-3.000	-2.036	.735	.429
.000	-1.500	-1.464	.313
-3.000	-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	.667	-1.143	.250

STOP

CP 4.156 SECS.

RUN COMPLETE.

LSO

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

*** START INPUT OF DATA ***

TOLERANCE

? .00000005

ORDER OF MATRIX - FORMAT 212-

? 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. 5. 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

C MATRIX

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

E-VALUES

-2.000000000000	0	-.999999998073	0
-1.00000001927	0	-2.000000000000	0
-2.000000000000	0	-1.000000000000	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	3.000	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 E

-.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 : $G(s) = C(sI-A)B + D$

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

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

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

#JH MI#140000

*** TEST RUN FOR MINIMAL REALISATION USING ROSEN BROCK S ALGORITHM

*** START INPUT OF DATA ***

TOLERANCE

? .000000000005

ORDER OF MATRIX -FORMAT 212

? 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	.900	.567
0	4.000	-1.500	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 212
? 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

B MATRIX

-2.000 2.000
1.000 2.000

C MATRIX

0 1.000 0
0 0 0

D MATRIX

0 1.000
2.000 0

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

- 91 -

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

TOP (I=PAE3)

<CTIME

CTIME. 79.341 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 a better 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 a case 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 of 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

```

000110 PROGRAM MINREA(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
000111
000112
000113
000114
000115
000116
000117
000118
000119
000120
000121
000122
000123
000124
000125
000126
000127
000128
000129
000130
000131
000132
000133
000134
000135
000136
000137
000138
000139
000140
000141
000142
000143
000144
000145
000146
000147
000148
000149
000150
000151
000152
000153
000154
000155
000156
000157
000158
000159
000160
000161
000162
000163
000164
000165
000166
000167
000168
000169
000170
000171
000172
000173
000174
000175
000176
000177
000178
000179
000180
000181
000182
000183
000184
000185
000186
000187
000188
000189
000190
000191
000192
000193
000194
000195

```

PROGRAMMER : A. POULIEZOS
DATE : SEPTEMBER 1976
MSS PROJECT FOR CONTROL DEPT.

..... TO CALCULATE THE MINIMAL REALISATION OF A TRANSFER FUNCTION M
..... USING THE JORDAN CANONICAL FORM
..... THIS PROGRAM USES THE MRE SUBROUTINE FROM THE MSS PACKAGE TO
..... INITIAL OBSERVABLE REALISATION

```

INTEGER RDR,PRT,DSK,DTP
REAL KEPL1,KEPL2,KEPL3,KEPL4
REAL IMZ
COMMON N,IT,ND, PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
COMMON A(20,20),B(2,10),C(10,20),D(10,10),N2,NT
COMMON ITYPE,N,M,NN(2,1),NDEN,GN(1,1,21),CD(21),IRET,NSUM
COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,1COMPL,JOIM(20),KI
COMMON JSEGN,MULT(2),WR(2),WA(20),VA(2,20),VI(20,20),NOIAG(20)
DIMENSION REZ(20),IMZ(20),COEF(20)
DIMENSION ND(10,10)
NRRET=31
LDR=5
PTR=3
WRITE(6,87)
80 FORMAT(1H,' ****TEST RUN FOR MINIMAL REALISATION USING THE
CANONICAL FORM ****',//,' *** START INPUT OF DATA ***')
..... INPUT OF G(S) .....
..... ORDER OF G(S),ORDER OF ACCEPTABLE ERROR .....
WRITE(6,93)
95 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(5,101)N,M
100 FORMAT(2I2)
..... 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(4I2)
..... ORDER OF DENOMINATOR
WRITE(6,103)
103 FORMAT(1H,/,3X,'ORDER OF DENOMINATOR')
READ(5,104)NOEN
104 FORMAT(I2)
..... 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
J=NM(I,J)+1
READ(5,105)(GN(I,J,K),K=1,IJ)
801 CONTINUE
800 CONTINUE
106 FORMAT(8(1X,F8.0))
..... COEFFICIENTS OF DENOMINATOR
WRITE(6,108)
108 FORMAT(1H,/,2X,'COEFFICIENTS OF DENOMINATOR')
READ(5,105)(CD(I),I=1,NOEN+1)
WRITE(6,81)
81 FORMAT(1H,' *** INPUT OF DATA FINISHED ***')
..... INPUT OF DATA FINISHED .....
..... CALL MRE SUBROUTINE TO CALCULATE INITIAL OBSERVABLE REALISATI
CALL MRE(10)
FIN=NT-1)STOP
IFIN=NSUM
WRITE(6,107)
1007 FORMAT(1H,/,3X,'OBSERVABLE REALISATION A B C')
WRITE(6,725)
725 FORMAT(1H,' A MATRIX')
DO 1001 I=1,IFIN
WRITE(6,109)(A(I,J),J=1,FIN)
1001 CONTINUE
WRITE(6,735)
755 FORMAT(1H,' B MATRIX')
DO 1002 I=1,FIN

```

```

0096 WRITE (6,130) (B(I,J),J=1,M)
0097 CONTINUE
0098 WRITE (6,134)
0099 784 FORMAT(1H,' C MATRIX')
0100 DO 1003 I=1,N
0101 WRITE (6,130) (C(I,J),J=1,IFIN)
0102 CONTINUE
0103 CONTINUE
0104 DO 20 I=1,N
0105 DO 21 J=1,M
0106 F=(C(I,J)).LE.EPS)GO TO 21
0107 GO TO 22
0108 21 CONTINUE
0109 21 CONTINUE
0110 GO TO 23
0111 22 WRITE (6,25)
0112 23 FORMAT(1H,' D MATRIX')
0113 DO 25 I=1,N
0114 WRITE (6,130) (D(I,J),J=1,M)
0115 CONTINUE
0116 26 FORMAT(1H,'/,10(1X,F7.3))
0117 C.... FIND E-VALUES OF A MATRIX BY SUCCESSIVELY CALLING LIB. SUB. C
0118 C.... TO CALCULATE THE ROOTS OF THE CHARACTERISTIC POLYNOMIALS OF
0119 C.... CANONICAL BLOCKS
0120 23 TOL=1.E-15
0121 IFIN=0
0122 DO 1 I=1,N
0123 IF (ND(I,1).EQ.3)GO TO 1
0124 IST= IAN+1
0125 IFIN=IFIN+ND(I,1)
0126 IF (NU(I,1).GT.1)GO TO 8
0127 REZ(1)=A(IST,IFIN)
0128 IMZ(1)=0
0129 GO TO 3
0130 6 II=ND(I,1)+1
0131 COEF(1)=1
0132 IZERO=1
0133 DO 2 J=2,II
0134 COEF(J)=-A(IFIN-J+2,IFIN)
0135 IF (ABS(COEF(J)).LE.EPS)GO TO 2
0136 IZERO=1
0137 2 CONTINUE
0138 IF (IZERO)4,4,6
0139 4 DO 9 J=1,II-1
0140 REZ(J)=0
0141 9 IMZ(J)=1.
0142 GO TO 3
0143 5 IFAIL=1
0144 5 CALL C2ARF(COEF,II,REZ,IMZ,TOL,IFAIL)
0145 IF (IFAIL-1)3,999,5
0146 999 WRITE(6,166)
0147 166 FORMAT(1H,' * E-VALUES NOT FOUND *')
0148 STOP
0149 DO 7 J=IST,IFIN
0150 WR(J)=REZ(J-IST+1)
0151 7 WI(J)=IMZ(J-IST+1)
0152 1 CONTINUE
0153 IFIN=NSUM
0154 WRITE (6,1305) (WR(I),WI(I),I=1,IFIN)
0155 1305 FORMAT(1H,' E-VALUES',7,4(1X,F15.11))
0156 C....
0157 C.... CHECK FOR COMPLEX E-VALUES AND CALL APPROPRIATE ROUTINE.....
0158 C....
0159 DO 4 I=1,IFIN
0160 IF (ABS(W(I)).LE.EPS)GO TO 400
0161 ICOMPL=1
0162 400 MULT(I)=1
0163 IF (ICOMPL)401,401,402
0164 401 CONTINUE
0165 GO TO 403
0166 402 CALL CCOMPL
0167 403 STOP
0168 END
0169 C....
0170 C....
0171 C....
0172 C....
0173 C....
0174 C....
0175 C
SUBROUTINE CCOMPL
*****
SUBROUTINE CCOMPL
INTEGER RDR,PRT,DSK,DTP
REAL KEPL1,KEPL2,KEPL3,KEPL4,KEPV1,KEPV2,KEPV3,KEPV4
COMPLEX AA,BB,CC
COMMON N,IT,EDR,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 DIMENSION N, M, NN(10,1), NDEN, GN(1,1,21), CD(21), IRET, NSUM
COMMON AA(20,20), BB(20,10), CCC(10,20), NDIM, IC, COMPL, JDIM(20),
DIMENSION JS, GN, MULT(20), WR(20), WI(20), V(20,20), VI(20,20), NDIAG(20)
EQUIVALENCE(AA(1,1), AAA(1,1)), (BB(1,1), BB3(1,1)), (CC(1,1), CCC(1,1))
WRITE(6,1000)
F73 FORMAT(1H, ' COMPLEX E-VALUES , MATICES WITH COMPLEX ELEME
IFIN=NSUM
K=1
KX=K
IO=0
DO 501 I=1, IFIN-1
IF (MULT(I)) 501, 501, 402
KX=KX+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(KX+1)=0
IO=1
MULT(J)=MULT(J)+1
KX=KX+1
WR(KX)=WR(I)
WI(KX)=WI(I)
WR(KX+1)=WR(J)
WI(KX+1)=WI(J)
WR(J)=WR(KX)
WI(J)=WI(KX)
K=K+1
IF (IO) CONTINUE
IF (MULT(IFIN)).GT.C) K3=K3+1
WRITE(6,2000) K3
F73 FORMAT(1H, ' DISTINCT E-VALUES ', I2)
F73 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 606
KO=J
IF (ABS(WR(KO)) - WR(KOO)) - EPS) 607, 607, 606
IF (ABS(WI(KO)) + WI(KOO)) - LPS) 608, 608, 606
607 IF (J-I-MULT(I)) 606, 606, 609
608 DO 611 I5=1, MULT(J)
I6=KO+I5-1+MULT(I)
KX=KX+1
KXPM1=MULT(I5)-1
KXPL1=WR(I5)
KXPL2=WI(I5)
KXPL3=WR(I6)
KXPL4=WI(I6)
MULT(I5)=MULT(I5)+1
MULT(I6)=MULT(I6)+1
KXPM1=KX-1
KXPL1=KXPL1-KXPL3
KXPL2=KXPL2-KXPL4
DO 613 I6=I5+1, I6
KXPM1=KX-1
KXPL1=KXPL1-KXPL3
KXPL2=KXPL2-KXPL4
CONTINUE
CONTINUE
CONTINUE
CALL CJORDA(K3, IRET)
IF (IRET) 1006, 1006, 2000
1006 WRITE(6,543)
F73 FORMAT(1H, ' TRANSFORMATION MATRIX FOR JORDAN FORM ')
DO 602 I=1, IFIN
WRITE(6,544) (VR(I, J), VI(I, J), J=1, IFIN)
5000 FORMAT(1H, '10(1X, F7.3)')
602 CONTINUE
CALL CTRANS(K3)
IF (IRET) GO TO 2000
CALL CSWRITE
RETURN
END

```

```

SUBROUTINE CTRANS
** ** ** ** **
SUBROUTINE CTRANS(K3)
INTEGER RDR, PRT, DSK, DTP
COMPLEX AT, AA, BB, CC, TT, TTT, QUOT
COMMON AT1, AT2
COMMON NRJT, 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), NOIAG(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, M
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(.F.N, AT, DIM1, D1, D2, ID1, RIND, IFAIL)
IF(IFAIL)398,398,398,999
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 F04AXF(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 122 IL=1, IFIN
WRITE(6, 1023) (TTT(IL, JL), JL=1, IFIN)
122 CONTINUE
1023 FORMAT(1H, 10(1X, F7.3))
1024 FORMAT(1H, 7, 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, 7, 2X, 'TRANSFORMED B MATRIX')
DO 72 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, 7, 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, 7, 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
IID1=20
IID2=20
DO 228 I=1, IFIN
DO 221 J=1, IFIN
221 AA(I, J)=CMPLX(A(I, J), 0.)
228 CONTINUE
CALL CMATRM(IID1, IID2, IFIN, IFIN, IFIN, TTT, AA, AT)
CALL CMATRM(IID1, IID2, IFIN, IFIN, IFIN, AT, TT, AA)
WRITE(6, 1035)

```

```

00 WRITE (6,1733) (AA(I,J),J=1,IFIN)
0135 FORMAT(1H ,/,1 TRANSFORMED A MATRIX IN JORDAN FORM)
0136 FORMAT(1H ,/,1 (1X,F7.3))
0137 CONTINUE
0138 NDIM=IFIN
0139 IF(IG) 16,416,431
0140 . . . . . DISTINCT E-VALUES JORDAN FORM DIAGONAL
0141 . . . . . LFINI IS NUMBER OF DISTINCT E-VALUES I.E. NUMBER OF SUBSYSTEM
0142 . . . . . NOIAG IS NUMBER OF JORDAN BLOCKS PER SUBSYSTEM
0143 . . . . . JOIM IS DIMENSION OF EACH JORDAN BLOCK
0144 . . . . . KIND=X(I) IS POSITION, IN AA, OF BLOCK I
0145 . . . . . KFR=IFIN
0146 KFR=IFIN
0147 NOIM=IFIN
0148 KBLOC=IFIN
0149 DO 250 IJ=1,KBLOC
0150 NDIAG(IJ)=1
0151 JDIM(IJ)=1
0152 GO TO 431
0153 IRRR=<3
0154 KKK=1
0155 KND=X(1)=1
0156 DO 3 I=1,IRRR
0157 DO 3 I2=1,NDIAG(I)
0158 KIND=X(KKK+I2)=KINDEX(KKK+I2-1)+JDIM(KKK+I2-1)
0159 CONTINUE
0160 KKK=KKK+NDIAG(I)
0161 CONTINUE
0162 JS=1
0163 LSTART=1
0164 SKIP=0
0165 MFORA=0
0166 DO 5 JJ=1,IRRR
0167 IF(ISKIP) 184,184,183
0168 . . . . . STEP 1, SET V(I)=, I=1,R
0169 IREP=0
0170 LFINI=LSTART+NOIAG(JJ)-1
0171 DO 55 J=LSTART,LFINI
0172 V(J)=0
0173 . . . . . STEPS 2 AND 3
0174 . . . . . KIMAX IS MAXIMUM SIZE OF BLOCK FOR WHICH IV=0
0175 KI MAX=1
0176 KSTOP=0
0177 DO 55 J=LSTART,LFINI
0178 IF(IV(J)) 56,57,56
0179 KSTOP=1
0180 IF(KMAX-JDIM(J)) 58,58,56
0181 KIMAX=JDIM(J)
0182 . . . . . JSIGN IS J ST. JOIM(J) IS MAX
0183 JSIGN=J
0184 CONTINUE
0185 IF(KSTOP) 55,55,59
0186 . . . . . STEP 4
0187 KTSIGN=KINDEX(JSIGN)+JDIM(JSIGN)-1
0188 DO 61 I=1,M
0189 IF(CABS(BB(KTSIGN,I))-EPS) 61,61,62
0190 CONTINUE
0191 GO TO 91
0192 IV(JSIGN)=1
0193 ISG=
0194 ISG1=I+LSTART-1
0195 DO 63 I=LSTART,LFINI
0196 IF(IV(I)) 53,64,63
0197 INDBB=KINDEX(I)+JDIM(I)-1
0198 INDBB=KINDEX(JSIGN)+JDIM(JSIGN)-1
0199 QUOT=BB(INDBB,ISG)/BB(INDBB,ISG)
0200 821 FORMAT(1H ,10(1X,F7.3))
0201 DO 65 I=1,JDIM(I)
0202 DO 66 IB=1,M
0203 INI1=KINDEX(I)+II-1
0204 INI2=KINDEX(JSIGN)+JDIM(JSIGN)-JDIM(I)+1 -1
0205 BB(INI1,IB)=BB(INI1,IB)-QUOT*BB(INI2,IB)
0206 CONTINUE
0207 DO 67 ICC=1,N
0208 CC(ICC,INI2)=CC(ICC,INI1)+QUOT*CC(ICC,INI1)
0209 CONTINUE
0210 WRITE (6,819)
0211 819 FORMAT(1H , * TRANSFORMED B MATRIX *)
0212 DO 822 I=1,N
0213 WRITE (6,821) (BB(IN,J),J=1,M)
0214 822 CONTINUE
0215 CONTINUE
0216 GO TO 54
0217 . . . . . IDEL IS INDEX FOR ROWS AND COLUMNS TO BE DELETED
0218 91 IF(JDIM(JSIGN).EQ.1)GO TO 69

```



```

044139 IRTP=1
044140 GO TO 569
044141 JS=GN=1
044142 LFINI=LFINI-1
044143 569 CALL CCONV (IV)
044144 IF (ABS(WIJJ)).LE.EPS) GO TO 54
044145 557 JS=GN=JS.GN+MULT(JJ)-MFORA
044146 MFORA=MFORA+1
044147 CALL CCONV (IV)
044148 ISKIP=1
044149 GO TO 54
044150 563 ISKIP=0
044151 MFORA=0
044152 555 IF (IRTP) 557,557,556
044153 557 LSTART=LFINI+1
044154 550 CONTINUE
044155 551 RETURN
044156 END

```

SUBROUTINE CSWRIT

```

044157 *****
044158 SUBROUTINE CSWRIT
044159 INTEGER RDR,PRT,DSK,DTP
044160 COMPLEX AA,BB,CC
044161 COMMON N,N2,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
044162 COMMON A(21,20),B(20,10),C(10,20),D(10,10),N2,NT
044163 COMMON ITYPE,N,M,NN(10,10),NDIM,GN(10,10,21),CD(21),IRET,NSUM
044164 COMMON AAA(2,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL
044165 DIMENSION AA(2,20),BB(2,10),CC(10,20)
044166 EQUIVALENCE (AA(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(1,1))
044167 WRITE(6,129)NDIM
044168 129 FORMAT(1H,/, ' DIMENSION ',I2)
044169 WRITE(6,379)
044170 379 FORMAT(1H, '*** MINIMAL REALISATION ***',/, '*** MATRICES IN R
044171 AFTER TRANSFORMATION ***',/, ' A MATRIX')
044172 130 FORMAT(1H,10(1X,F7.3))
044173 KOUNT=0
044174 IFIN=NDIM
044175 DO 26 I=1,IFIN
044176 DO 27 J=1,IFIN
044177 27 A(I,J)=REAL(AA(I,J))
044178 DO 1 J1=1,M
044179 E(J,J1)=REAL(BB(I,J1))
044180 DO 2 J2=1,N
044181 C(J2,I)=REAL(CC(J2,I))
044182 26 CONTINUE
044183 I=1
044184 15 IF (ABS(AIMAG(AA(I,I))+AIMAG(AA(L+1,L+1))).GT.EPS) GO TO 11
044185 IF (ABS(REAL(AA(I,I))-REAL(AA(L+1,L+1))).GT.EPS) GO TO 11
044186 I=L+1
044187 KOUNT=KOUNT+1
044188 IF ((L+1) GT. NDIM) GO TO 11
044189 GO TO 15
044190 11 IF (KOUNT) 10,10,13
044191 DO 14 J=I-KOUNT+1,I
044192 DO 30 JJ=I-KOUNT+1,I
044193 A(J,JJ)=REAL(AA(J,JJ))
044194 30 CONTINUE
044195 DO 20 I1=1,M
044196 B(J,I1)=AIMAG(BB(J,I1))
044197 DO 21 I2=1,N
044198 C(I2,J)=2.*AIMAG(CC(I2,J))
044199 14 CONTINUE
044200 DO 16 J=I+1,I+KOUNT
044201 DO 17 JJ=I+1,I+KOUNT
044202 A(J,JJ)=REAL(AA(J,JJ))
044203 DO 23 I1=1,M
044204 B(J,I1)=REAL(BB(J,I1))
044205 DO 24 I2=1,N
044206 C(I2,J)=2.*REAL(CC(I2,J))
044207 16 CONTINUE
044208 DO 18 J=I+1,I+KOUNT
044209 DO 19 JJ=I-KOUNT+1,I
044210 A(J,JJ)=-AIMAG(AA(J-KOUNT,JJ))
044211 19 A(JJ,J)=AIMAG(AA(JJ,J-KOUNT))
044212 18 CONTINUE
044213 I=I+KOUNT+1
044214 KOUNT=0
044215 GO TO 22

```

```

11 I=I+1
12 IF (I.LT.NDIM) GO TO 25
13 DO 40 I=1,IFIN
14 WRITE (6,100) (A(I,J),J=1,IFIN)
15 CONTINUE
16 WRITE (6,101)
17 FORMAT (1H,' B MATRIX')
18 DO 41 I=1,N
19 WRITE (6,102) (B(I,J),J=1,M)
20 CONTINUE
21 WRITE (6,103)
22 FORMAT (1H,' C MATRIX')
23 DO 42 I=1,N
24 WRITE (6,104) (C(I,J),J=1,IFIN)
25 CONTINUE
26 RETURN
27 END

```

SUBROUTINE CCONVE

```

*****
SUBROUTINE CCONVE(ILS)
..... TO DELETE ROWS AND COLUMNS FROM AA,BB,CC,
INTEGER RDR,PRT,USK,DTP
COMPLEX AA,BB,CC
COMMON N,LD,PRT,USK,DTP,LP,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),CU(21),IRET,NSUM
COMMON AAA(2,2),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(1,1))
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=1,4
BB(I,J)=BB(I+1,J)
2 CONTINUE
DO 3 IJ=1,N
3 CC(IJ,)=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
IF (MULT(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

```

SUBROUTINE GCOM

```

*****
SUBROUTINE GCOM(A,IA,GCD,IGCD,EPS)
SUBROUTINE TO CALCULATE THE GREATEST COMMON DIVISOR OF 2 POLYNOMIALS
STORED IN ARRAY A. THIS IS DONE BY SUCCESSIVELY SUBTRACTING
MULTIPLES OF THE POLYNOMIAL OF LEAST DEGREE AT ANY ONE TIME
FROM THE OTHER UNTIL ONE NON-ZERO POLYNOMIAL REMAINS. THIS
IS THE GCD OF THE SET AND IS MADE MONIC BEFORE EXIT.
DIMENSION A(2,21),GCD(21)
DIMENSION IA(2)
IS=0
IF (IA(1).EQ.0.OR.IA(2).EQ.0) GO TO 500
92 IF (ABS(A(1,1)).GT.EPS.AND.ABS(A(2,1)).GT.EPS) GO TO 500
93 IF (ABS(A(1,1)).LE.EPS.AND.ABS(A(2,1)).LE.EPS) GO TO 903

```

```

6606 IF (ABS(S(A(1,1)) - GT.EPS) GO TO 904
6607 IA1=IA(1)+1
6608 DO 905 K=2, IA1
6609 K1=K-1
6610 A(1, K1)=A(1, K)
6611 905 CONTINUE
6612 A(1, IA1)=0.0
6613 IA(1)=IA(1)-1
6614 GO TO 501
6615 904 IA2=IA(2)+1
6616 DO 906 K=2, IA2
6617 K1=K-1
6618 A(2, K1)=A(2, K)
6619 906 CONTINUE
6620 A(2, IA2)=0.0
6621 IA(2)=IA(2)-1
6622 GO TO 501
6623 903 IS=IS+1
6624 IA1=IA(1)+1
6625 DO 907 K=2, IA1
6626 K1=K-1
6627 A(1, K1)=A(1, K)
6628 907 CONTINUE
6629 A(1, IA1)=0.0
6630 IA(1)=IA(1)-1
6631 IA2=IA(2)+1
6632 DO 908 K=2, IA2
6633 K1=K-1
6634 A(2, K1)=A(2, K)
6635 908 CONTINUE
6636 A(2, IA2)=0.0
6637 IA(2)=IA(2)-1
6638 GO TO 902
6639 501 LL=1
6640 IF (IA(2) .LT. IA(1)) LL=2
6641 NN=3-LL
6642 IA1=IA(NN)+1
6643 IA2=A(LL)+1
6644 IF (IA(LL) .EQ. 0. AND. ABS(A(LL,1)) .LT. EPS) GO TO 100
6645 IF (IA(LL) .EQ. 0) GO TO 200
6646 IDEG=IA(NN)-IA(LL)
6647 RATIO=A(NN, IA1)/A(LL, IA2)
6648 DO 300 K=1, IA2
6649 KI=K+IDEG
6650 A(NN, KI)=A(NN, K1)-RATIO*A(LL, K)
6651 IF (ABS(A(NN, KI)) .LT. EPS) A(NN, KI)=0.0
6652 300 CONTINUE
6653 DO 400 KK=1, IA1
6654 K=IA(NN)+2-KK
6655 IF (ABS(A(NN, K)) .LT. EPS) GO TO 400
6656 IA(NN)=K-1
6657 GO TO 410
6658 400 CONTINUE
6659 IA(NN)=0
6660 410 CONTINUE
6661 GO TO 500
6662 500 DO 600 K=1, IA1
6663 GCD(K)=A(NN, K)/A(NN, IA1)
6664 A(NN, K)=0.0
6665 600 CONTINUE
6666 IGCD=IA(NN)
6667 GO TO 800
6668 200 DO 700 I=1, 2
6669 IA3=IA(I)+1
6670 DO 700 K=1, IA3
6671 A(I, K)=0.0
6672 GCD=0
6673 800 IF (IGCD .EQ. 0) GCD(1)=1.0
6674 IF (ABS(EQ(1)) .GT. EPS) GO TO 801
6675 IA1=GCD+1
6676 DO 802 K=1, IA1
6677 K1=K+IS
6678 GCD(K1)=GCD(K)
6679 802 CONTINUE
6680 DO 803 K=1, IS
6681 GCD(K)=0
6682 IGCD=IGCD+IS
6683 803 RETURN
6684 END

```

```

0691 C *****
0692 C SUBROUTINE FNORM(X, IDIMX, EPS)
0693 C PJ-POSE
0694 C NORMALIZE COEFFICIENT VECTOR OF A POLYNOMIAL
0695 C
0696 C USAGE
0697 C CALL FNORM(X, IDIMX, EPS)
0698 C
0699 C DESCRIPTION OF PARAMETERS
0700 C X - VECTOR OF ORIGINAL COEFFICIENTS, ORDERED FROM
0701 C SMALLEST TO LARGEST POWER. IT REMAINS UNCHANGED
0702 C DIMX - DIMENSION OF X. IT IS REPLACED BY FINAL DIMENSION
0703 C EPS - TOLERANCE BELOW WHICH COEFFICIENT IS ELIMINATED
0704 C
0705 C REMARKS
0706 C IF ALL COEFFICIENTS ARE LESS THAN EPS, RESULT IS A ZERO
0707 C POLYNOMIAL WITH IDIMX=0 BUT VECTOR X REMAINS INTACT
0708 C
0709 C SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
0710 C NONE
0711 C
0712 C METHOD
0713 C DIMENSION OF VECTOR X IS REDUCED BY ONE FOR EACH TRAILING
0714 C COEFFICIENT WITH AN ABSOLUTE VALUE LESS THAN OR EQUAL
0715 C TO EPS
0716 C
0717 C .....
```

```

0718 C DIMENSION X(21)
0719 C
0720 C 1 IF (IDIMX) 4,4,2
0721 C 2 IF (ABS(X(IDIMX)) - EPS) 3,3,4
0722 C 3 IDIMX=IDIMX-1
0723 C GO TO 1
0724 C 4 RETURN
0725 C END
0726 C .....
```

```

0727 C
0728 C
0729 C
0730 C
0731 C SUBROUTINE COMDEN
0732 C *****
0733 C SUBROUTINE COMDEN(A, IA, B, IB, M, N, LCM, ILCM, EPS)
0734 C
0735 C SUBROUTINE TO FIND THE COMMON DENOMINATOR OF A RATIONAL
0736 C FUNCTION MATRIX. THE NUMERATOR POLYNOMIALS A ARE RETURNED
0737 C THE COMMON DENOMINATOR LCM.
0738 C WHEN CALLED BY MRE, THE COMMON DENOMINATOR OF EACH ROW
0739 C ARE FOUND, AND STORED IN B(I,1), FOR EACH I
0740 C
0741 C PROGRAMMER: P. KATZBERG 7/73
0742 C MOD1: A. D. F. 12/11/74
0743 C
0744 C SUBROUTINES CALLED: GCOM
0745 C
0746 C NRET = 31 IFF CALLING PROGRAM IS MRE
0747 C
```

```

0748 C INTEGER RDR, PRT
0749 C REAL LCM
0750 C DIMENSION A(10,10,21), B(10,10,21), LCM(21), TEMP(21), POLY(2,21), G
0751 C DIMENSION IA(10,10), IB(10,10), IPOLY(2)
0752 C COMMON NRET, RDR, PRT
0753 C
0754 C CALCULATE COMMON DENOMINATOR
0755 C
0756 C
0757 C I=1
0758 C 701 B1 = 0(I,1)+1
0759 C
0760 C SET POLY(1) = LCM = B(I,1)
0761 C DO 1 K=1, IB1
0762 C LCM(K) = B(I,1,K)
0763 C POLY(1,K) = LCM(K)
0764 C 100 CONTINUE
0765 C
0766 C ILCM1 = IB(I,1)+1
0767 C POLY(1) = B(I,1)
0768 C JST = 1
0769 C IF (NRET.NE.31) GO TO 702
0770 C IF (N.LT.2) GO TO 201
0771 C JST = 2
0772 C
0773 C
0774 C RANGE OVER ALL COLUMNS
0775 C 702 DO 200 J=JST, N
```

```

776. F(NRET.EQ.31) GO TO 773
777. IST=2
778. IF(J.GT.1) IST=1
779. IF(IST.GT.M) GO TO 200
780. C
781. C RANGE OVER ALL ROWS
782. DO 250 I=IST,M
783. C
784. C SET POLY(2) = TEMP = B(I,J)
785. 773 B1=B(I,J)+1
786. DO 210 K=1,IB1
787. TEMP(K)=B(I,J,K)
788. POLY(2,K)=TEMP(K)
789. 210 CONTINUE
790. C
791. C CALCULATE GCD OF POLY(1) AND POLY(2)
792. IPOLY(2)=IB(I,J)
793. CALL GCOM(POLY,1POLY,GCD,IGCD,EPS)
794. IDEG1=IB1-IGCD
795. IGCD1=IGCD+1
796. IF(IGCD.EQ.1) GO TO 222
797. IF((IDEG1+IGCD1).GT.22) GO TO 600
798. C
799. C DIVIDE TEMP BY GCD, BY SUCCESSIVE SUBTRACTION
800. L=IDEG1
801. 220 LL=L+IGCD
802. RATIO=TEMP(LL)/GCD(IGCD1)
803. C
804. C SUBTRACT C(L)*(S*L)*GCD FROM TEMP
805. DO 221 K=1,IGCD1
806. KL1=K+L-1
807. TEMP(KL1)=TEMP(KL1)-RATIO*GCD(K)
808. 221 CONTINUE
809. C
810. C STORE C(L) IN TEMP(LL)
811. IF(ABS(RATIO).LT.EPS) RATIO=.0
812. TEMP(LL)=RATIO
813. L=L-1
814. IF(L.GT.0) GO TO 220
815. C
816. C SHIFT TO TOP OF ARRAY
817. L=IGCD
818. DO 223 K=1,IDEG1
819. L=L+1
820. TEMP(K)=TEMP(L)
821. 223 CONTINUE
822. C
823. C SET POLY(1) = LCM+TEMP
824. 222 F((ILCM1+IDEG1).GT.22) GO TO 600
825. KLIM=ILCM1+IDEG1-1
826. DO 224 K=1,KLIM
827. 224 POLY(1,K)=.0
828. DO 230 K=1,ILCM1
829. DO 230 L=1,IDEG1
830. KL1=K+L-1
831. POLY(1,KL1)=POLY(1,KL1)+LCM(K)*TEMP(L)
832. 230 CONTINUE
833. C
834. C SET LCM = POLY(1)
835. IPOLY(1)=ILCM1+IDEG1-2
836. POLY1=IPOLY(1)+1
837. DO 240 K=1,IPOLY1
838. IF(ABS(POLY(1,K)).LT.EPS) POLY(1,K)=.0
839. LCM(K)=POLY(1,K)
840. 240 CONTINUE
841. C
842. C ILCM1=IPOLY1
843. IF(ILCM1.GT.21) GO TO 600
844. IF(NRET.EQ.31) GO TO 200
845. 250 CONTINUE
846. C
847. C END OF ROW RANGE
848. C
849. C 200 CONTINUE
850. 201 CONTINUE
851. C
852. C END OF COLUMN RANGE.
853. C
854. C CALCULATE NUMERATORS
855. C
856. C
857. C RANGE OVER COLUMNS
858. DO 450 J=1,N
859. IF(NRET.EQ.31) GO TO 784
860. C
861. C RANGE OVER ROWS

```

```

0886 DO 410 I=1,M
0887 C SET TEMP = LCM
0888 DO 410 K=1,ILCM1
0889 TEMP(K)=LCM(K)
0890 410 CONTINUE
0891 C
0892 IB1=IB(I,J)+1
0893 DEG1=ILCM1-IB(I,J)
0894 IF((1-DEG1+IB1).GT.22) GO TO 600
0895 L=IDEG1
0896 IB2=IB1-1
0897 C
0898 DIVIDE TEMP BY B(I,J)
0899 420 LL=L+IB2
0900 RATIO=TEMP(LL)/B(I,J,IB1)
0901 DO 421 K=1,IB1
0902 KL1=K+L-1
0903 TEMP(KL1)=TEMP(KL1)-RATIO*B(I,J,K)
0904 421 CONTINUE
0905 IF(ABS(RATIO).LT.EPS) RATIO=0.0
0906 TEMP(LL)=RATIO
0907 L=L-1
0908 IF(L.GT.0) GO TO 420
0909 C
0910 SHIFT TO TOP OF ARRAY
0911 L=IB2
0912 DO 422 K=1,IDEG1
0913 L=L+1
0914 TEMP(K)=TEMP(L)
0915 422 CONTINUE
0916 C
0917 IA1=IA(I,J)+1
0918 IF(IA1.EQ.1.AND.ABS(A(I,J,1)).LT.EPS) GO TO 460
0919 IF((1-DEG1+2*IA1).GT.22) GO TO 600
0920 C
0921 KLIM = IDEG1 + IA1 - 1
0922 DO 424 K=1,KLIM
0923 424 POLY(2,K) = 0.0
0924 C
0925 SET POLY(2) = TEMP * A(I,J)
0926 DO 430 K=1,IDEG1
0927 DO 430 L=1,IA1
0928 KL1=K+L-1
0929 POLY(2,KL1)=POLY(2,KL1)+TEMP(K)*A(I,J,L)
0930 430 CONTINUE
0931 C
0932 SET A(I,J) = POLY(2)
0933 IA(I,J)=IA(I,J)+IDEG1-1
0934 IA1=IA(I,J)+1
0935 DO 440 K=1,IA1
0936 IF(ABS(POLY(2,K)).LT.EPS) POLY(2,K)=0.0
0937 A(I,J,K)=POLY(2,K)
0938 440 CONTINUE
0939 C
0940 450 F(NR,LT,EQ,31) GO TO 450
0941 400 CONTINUE
0942 C END OF ROW RANGE
0943 C
0944 450 CONTINUE
0945 C END OF COLUMN RANGE
0946 C
0947 SET B = 0
0948 DO 500 J=1,N
0949 IF(NR,LT,EQ,31) GO TO 705
0950 DO 500 I=1,M
0951 705 IB1=IB(I,J)+1
0952 DO 510 K=1,IB1
0953 B(I,J,K)=0.0
0954 510 CONTINUE
0955 IB(I,J)=IB1
0956 IF(NR,LT,EQ,31) GO TO 706
0957 500 CONTINUE
0958 C
0959 706 ILCM = ILCM1 - 1
0960 IF(NR,LT,EQ,31) GO TO 803
0961 520 IB(I,J)=ILCM1-1
0962 DO 707 K=1,ILCM1
0963 707 B(I,J,K)=LCM(K)
0964 IF(I.EQ.M) GO TO 803
0965 I = I + 1
0966 GO TO 701
0967 C
0968 NORMALISE NUMERATOR POLYNOMIALS
0969 C
0970 803 DO 800 I=1,M

```

```

000000 DO 800 J=1,N
000001 K=A(I,J)+1
000002 CONTINUE IF(ABS(A(I,J,K)).GT.EPS.OR.K.EQ.1) GO TO 801
000003 K=K-1
000004 GO TO 802
000005 801 A(I,J)=K-1
000006 CONTINUE
000007 NPRINT=2
000008 GO TO 610
000009 600 WRITE(6,620)
000010 620 FORMAT(23H STORAGE LIMIT EXCEEDED)
000011 NPRINT=1
000012 610 RETURN
000013 END

```

.....

.... SUBROUTINE CANCEL

SUBROUTINE CANCEL (A,IA,B,IB,EPS)

SUBROUTINE TO CHECK FOR POLE-ZERO CANCELLATION AND IF FOUND
 TO PERFORM THE CANCELLATION.

PROGRAMMER: P.KATZBERG 7/73

SUBROUTINE CALLED: GCOM,PNORM

```

000014 INTEGER RDR,PRT
000015 DIMENSION A(21),B(21),POL(2,21),GCD(21),IPOL(2),P(21)
000016 COMMON NPRINT,RDR,PRT
000017 C
000018 IF(IA.EQ.1.OR.IB.EQ.0) GO TO 12
000019 IA1=IA+1
000020 CALL PPNORM(A,IA1,EPS)
000021 IB1=IB+1
000022 CALL PPNORM(B,IB1,EPS)
000023 IF(IA1.GT.0.AND.IB1.GT.0) GO TO 2
000024 IF(IB1.NE.0) GO TO 13
000025 WRITE(6,100)
000026 100 FORMAT(22H ALL COEFFICIENTS ZERO)
000027 NPRINT=1
000028 GO TO 3
000029 13 IF(IA1.NE.0) GO TO 2
000030 IA=0
000031 A(1)=0.0
000032 IB=0
000033 P(1)=1.0
000034 GO TO 1
000035 2 DO 4 K=1,IA1
000036 4 POL(1,K)=A(K)
000037 DO 5 K=1,IB1
000038 5 POL(2,K)=B(K)
000039 POL(1)=POL(1)-1
000040 IPOL(2)=IPOL(2)-1
000041 CALL GCOM(POL,IPOL,GCD,IG,EPS)
000042 IF(IG.EQ.0) GO TO 1
000043 IG1=IG+1
000044 IO1=IA1-IG
000045 IA1=IG
000046 I=IO1
000047 7 11=1+IA1
000048 P(I)=A(11)/GCD(IG1)
000049 IF(ABS(P(I)).LT.EPS) P(I)=0.0
000050 DO 6 K=1,IA1
000051 6 J=K-1+I
000052 A(J)=A(J)-P(I)*GCD(K)
000053 CONTINUE
000054 I=I-1
000055 IF(I.GT.0) GO TO 7
000056 DO 8 K=1,IO1
000057 8 A(K)=P(K)
000058 IA=IO1-1
000059 IO1=IB1-IG
000060 IB1=IG
000061 I=IO1
000062 10 11=I+IB1
000063 P(1)=B(11)/GCD(IG1)
000064 IF(ABS(P(1)).LT.EPS) P(1)=0.0
000065 DO 9 K=1,IB1
000066 9 J=K-1+I
000067 B(J)=B(J)-P(1)*GCD(K)
000068 CONTINUE
000069 I=I-1
000070 100000
000071 100000
000072 100000
000073 100000
000074 100000
000075 100000
000076 100000
000077 100000
000078 100000
000079 100000
000080 100000
000081 100000
000082 100000
000083 100000
000084 100000
000085 100000
000086 100000
000087 100000
000088 100000
000089 100000
000090 100000
000091 100000
000092 100000
000093 100000
000094 100000
000095 100000
000096 100000
000097 100000
000098 100000
000099 100000
000100 100000
000101 100000
000102 100000
000103 100000
000104 100000
000105 100000
000106 100000
000107 100000
000108 100000
000109 100000
000110 100000
000111 100000
000112 100000
000113 100000
000114 100000
000115 100000
000116 100000
000117 100000
000118 100000
000119 100000
000120 100000
000121 100000
000122 100000
000123 100000
000124 100000
000125 100000
000126 100000
000127 100000
000128 100000
000129 100000
000130 100000
000131 100000
000132 100000
000133 100000
000134 100000
000135 100000
000136 100000
000137 100000
000138 100000
000139 100000
000140 100000
000141 100000
000142 100000
000143 100000
000144 100000
000145 100000
000146 100000
000147 100000
000148 100000
000149 100000
000150 100000
000151 100000
000152 100000
000153 100000
000154 100000
000155 100000
000156 100000
000157 100000
000158 100000
000159 100000
000160 100000
000161 100000
000162 100000
000163 100000
000164 100000
000165 100000
000166 100000
000167 100000
000168 100000
000169 100000
000170 100000
000171 100000
000172 100000
000173 100000
000174 100000
000175 100000
000176 100000
000177 100000
000178 100000
000179 100000
000180 100000
000181 100000
000182 100000
000183 100000
000184 100000
000185 100000
000186 100000
000187 100000
000188 100000
000189 100000
000190 100000
000191 100000
000192 100000
000193 100000
000194 100000
000195 100000
000196 100000
000197 100000
000198 100000
000199 100000
000200 100000

```

```

10310 F(1,GT,0) GO TO 10
10320 DO 11 K=1, .01
10330 11 B(K)=P(K)
10340 1B=101-1
10350 GO TO 1
10360 2 F(LA VE...OR,ABS(A(1)),GT,EPS) GO TO 1
10370 1B=0
10380 B(1)=1.0
10390 1 NRET=2
10400 3 RETURN
10410 END
.....
10430 C
10440 C
10450 C
10460 C
10470 C
10480 C
10490 C
10500 C
10510 C
10520 C
10530 C
10540 C
10550 C
10560 C
10570 C
10580 C
10590 C
10600 C
10610 C
10620 C
10630 C
10640 C
10650 C
10660 C
10670 C
10680 C
10690 C
10700 C
10710 C
10720 C
10730 C
10740 C
10750 C
10760 C
10770 C
10780 C
10790 C
10800 C
10810 C
10820 C
10830 C
10840 C
10850 C
10860 C
10870 C
10880 C
10890 C
10900 C
10910 C
10920 C
10930 C
10940 C
10950 C
10960 C
10970 C
10980 C
10990 C
11000 C
11010 C
11020 C
11030 C
11040 C
11050 C
11060 C
11070 C
11080 C
11090 C
11100 C
11110 C
11120 C
11130 C
11140 C
11150 C

```

SUBROUTINE MRE

SUBROUTINE MRE(ND)
SUBROUTINE TO COMPUTE THE MINIMAL REALISATION A,B,C,D OF A
TRANSFER FUNCTION MATRIX G, BY SETTING UP AN OBSERVABLE
REALISATION USING CANONICAL FORMS AND THEN FINDING THE
CONTROLLABLE PART USING ROSLINDROCK'S ALGORITHM.
PROGRAMMER: P.KATZBERG 8/73
MOD1: A.D.FIELD 27/11/74
SUBROUTINES CALLED: IO,CANCEL,COMDEN,IOS
INTEGER ROR,PRT,DSK,DTP
DIMENSION ND(10,10),GD(10,10,21),WK1(21),WK2(21),FILE(2),IU(10)
CAE(20,30)
COMMON N,NT,PDR,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(10,1),NDEN,GN(1,10,21),CD(21),IRET
COMMON NSUM
EQUIVALENCE (AB,A)
IRET=NRET
ERR=EPS
COMPUTE THE L.C.M'S BY POWS OF THE DENOMINATOR OF G
41 DO 1 I=1,N
DO 1 J=1,M
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)
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)
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
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.EQ.1) GO TO 10
SET JPD MATRIX
DO 35 I=1,N
DO 35 J=1,M
35 D(I,J)=0.0
DO 50 I=1,N
DO 50 J=1,M
IF(NN(I,J).GT.ND(I,1)) GO TO 37
-F(NN(I,J).NE.ND(I,1)) GO TO 50
K1=NN(I,J)+1


```

11160 CON=GN(I,J,K1)
11170 IF(NN(I,J).GT.0) GO TO 55
11180 C
11190 GN(I,J,1)=
11200 GO TO 56
11210 C
11220 C SET G = G-D
11230 55 K1=K1-1
11240 DO 51 K=1,K1
11250 51 GN(I,J,K)=GN(I,J,K)-CON+GD(I,1,K)
11260 NN(I,J)=K1-1
11270 C
11280 56 D(I,J)=CON
11290 56 CONTINUE
11300 GO TO 138
11310 C
11320 37 WRITE(6,1001)
11330 1001 FORMAT(40H NUMERATOR ORDER HIGHER THAN DENOMINATOR/
11340 +ORDER)
11350 NRET = 1
11360 GO TO 10
11370 C
11380 C SET UP AN INITIAL A MATRIX
11390 C
11400 138 NSUM=0
11410 C
11420 C CALCULATE THE DIMENSION OF THE A MATRIX
11430 DO 8 I=1,N
11440 8 NSUM=NSUM+ND(I,1)
11450 IF(NSUM.LT.2) GO TO 9
11460 C
11470 C OVERFLOW
11480 WRITE(6,1000)
11490 1000 FORMAT(23H STORAGE LIMIT EXCEEDED)
11500 NRET=1
11510 GO TO 10
11520 C
11530 9 DO 6 I=1,20
11540 DO 6 J=1,20
11550 6 A(I,J)=0.0
11560 C
11570 C SET UP BLOCK IROW OF THE A MATRIX, FOR IROW = 1,N
11580 IFIN=0
11590 DO 7 IROW=1,N
11600 IF(ND(IROW,1).EQ.0) GO TO 7
11610 IST=IFIN+1
11620 IFIN=IFIN+ND(IROW,1)
11630 A(IST,IFIN)=-GD(IROW,1,1)
11640 I1=IST
11650 I2=ND(IROW,1)
11660 IF(I2.LT.2) GO TO 7
11670 DO 77 I=2,I2
11680 I1=I1+1
11690 A(I1,I1-1)=1.0
11700 A(I1,IFIN)=-GD(IROW,1,I)
11710 77 CONTINUE
11720 7 CONTINUE
11730 C
11740 C SET UP AN INITIAL B MATRIX
11750 C
11760 DO 11 I=1,20
11770 DO 11 J=1,20
11780 11 B(I,J)=0.0
11790 C
11800 IFIN=0
11810 DO 12 IROW=1,N
11820 IF(ND(IROW,1).EQ.0) GO TO 12
11830 IST=IFIN+1
11840 IFIN=IFIN+ND(IROW,1)
11850 DO 13 J=1,M
11860 13 NN(IROW,J)+IST
11870 38 K=0
11880 DO 13 I=IST,I1
11890 K=K+1
11900 B(I,J)=GN(IROW,J,K)
11910 13 CONTINUE
11920 12 CONTINUE
11930 C
11940 C SET UP INITIAL C MATRIX
11950 C
11960 15 DO 16 I=1,10
11970 DO 16 J=1,20
11980 16 C(I,J)=0.0
11990 J=0
12000 DO 17 I=1,N

```

```

1201 IF (ND(I,1),EQ,0) GO TO 17
1202 J=J+ND(I,1)
1203 C(I,J)=1
1204 17 CONTINUE
1205 GO TO 888
1206 18 WRITE(5,101)
1207 191 FORMAT(1H,'OBSERVABLE REALISATION NOT OBTAINED')
1208 888 RETURN
1209 END

```

.....

```

1210 C
1211 C
1212 C
1213 C
1214 C
1215 C
1216 C
1217 C
1218 C
1219 C
1220 C
1221 C
1222 C
1223 C
1224 C
1225 C
1226 C
1227 C
1228 C
1229 C
1230 C
1231 C
1232 C
1233 C
1234 C
1235 C
1236 C
1237 C
1238 C
1239 C
1240 C
1241 C
1242 C
1243 C
1244 C
1245 C
1246 C
1247 C
1248 C
1249 C
1250 C
1251 C
1252 C
1253 C
1254 C
1255 C
1256 C
1257 C
1258 C
1259 C
1260 C
1261 C
1262 C
1263 C
1264 C
1265 C
1266 C
1267 C
1268 C
1269 C
1270 C
1271 C
1272 C
1273 C
1274 C
1275 C
1276 C
1277 C
1278 C
1279 C
1280 C
1281 C
1282 C
1283 C
1284 C
1285 C

```

SUBROUTINE CRANK

```

SUBROUTINE CRANK(ARANK, EPS, MS, N, IRAN)
.....
SUBROUTINE TO CALCULATE THE RANK OF AN MSXM MATRIX
.....
USES GAUSS ELIMINATION WITH FULL PIVOTING
.....
COMPLEX ARANK, Q
.....
DIMENSION ARANK(20,20), IR(20), IC(20)
.....
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
.....
BEGIN ELIMINATION PROCEDURE
DO 200 LS=1, MM
200 AMAG=0
.....
SEARCH FOR PIVOT
DO 120 I=LS, MS
DO 121 J=LS, M
ADUM=CABS(ARANK(IR(I), IC(J)))
IF (ADUM-AMAG) 124, 120, 125
125 IS=I
JS=J
AMAG=ADUM
120 CONTINUE
.....
TEST FOR COMPLETION
IF (AMAG-EPS) 125, 125, 130
125 IRAN=LS-1
GO TO 300
130 CONTINUE
.....
INTERCHANGE ROW AND COLUMN INDICES
I=LS (LS)
IR(LS)=IR(IS)
IR(IS)=IT
IT=IC(LS)
IC(LS)=IC(JS)
IC(JS)=IT
.....
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
.....
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=I-1
170 CONTINUE
200 CONTINUE
300 CONTINUE
RETURN
END

```

.....

SUBROUTINE CJORDA

```

.....
TO CALCULATE THE JORDAN FORM OF A MATRIX WITH MULTIPLE
.....
E-VALUES BY GENERATING A SET OF GENERALISED E-VECTORS
.....
USES SSP SUBROUTINES ARRAY AND MFGR AND
.....
PRANK, INDEP

```

```

1120 SUBROUTINE CJORDA(K3,LSTOP)
1121 INTRINSIC COMPLEX,ABS,MIN,MAX,PRT,DSK,DTP
1122 COMPLEX X(2,20),Y(2,20),Z(2,20),U(2,20),UU,UR
1123 COMMON X(2,20),Y(2,20),Z(2,20),U(2,20),UU,UR
1124 COMMON PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)
1125 COMMON A(2,20),B(2,10),C(1,20),D(1,10),N2,NT
1126 COMMON IT,P,N,M,NN(10,10),NDEN,GN(10,10,21),CC(21),IRET,NSUM
1127 COMMON AAA(20,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JOIM(20),K
1128 COMMON JSIGN,MULT(20),WR(20),WI(20),VR(20,20),VI(20,20),NDIAG(20)
1129 DIMENSION ON AP(5,20,20),U(20,1),UR(20,1),E(2,2)
1130 DIMENSION EA(20,20),E(20,20),UU(20,20)
1131 DIMENSION XE(2,20),BB(20,10),CC(10,20)
1132 EQUIVALENCE (XE(1,1),AAA(1,1)),(BB(1,1),E(1,1)),(CC(1,1),CCC(1,1))
1133 EQUIVALENCE (AP(1,1,1),GN(1,1,1)),(E(1,1),BB(1,1)),(E(1,1),VI(1,1))
1134 EQUIVALENCE (UU(1,1),CC(1,1)),(E(1,1),VR(1,1))
1135 IFIN=NSUM
1136 LSTOP=0
1137 I=0
1138 IVEC=1
1139 IRAN=0
1140 LLSUM=3
1141 KPOINT=1
1142 KI=0
1143 I1=1
1144 C... BEGIN LOOP FOR EACH E-VALUE
1145 DO 1 I1=1,K3
1146 ITEST=0
1147 K1=KI+MULT(I1)
1148 KI=K1
1149 DO 2 I=1,FIN
1150 DO 3 J=1,IFIN
1151 IF(I.EQ.J)GO TO 3
1152 AP(1,I,J)=COMPLX(A(I,J),0.)
1153 E(1,J)=AP(1,I,J)
1154 GO TO 2
1155 3 AP(1,I,J)=COMPLX(A(I,J)-WR(K1),-WI(K1))
1156 E(1,J)=AP(1,I,J)
1157 2 CONTINUE
1158 1 CONTINUE
1159 NDIAG(I1)=0
1160 DO 11 IJ1=1,IFIN
1161 DO 12 IJ2=1,FIN
1162 E(IJ1,IJ2)=AP(K,IJ1,IJ2)
1163 E(1,IJ1,IJ2)=E(IJ1,IJ2)
1164 12 CONTINUE
1165 11 CONTINUE
1166 IF(MULT(I1)-1)200,200,40
1167 200 K=2
1168 NDIAG(I1)=1
1169 GO TO 33
1170 C... FIND RANK OF AP(K,I,J)
1171 40 KIRAN=IRAN
1172 CALL CRANK(F4,EPS,IFIN,IFIN,IRAN)
1173 DO 70 I=1,IFIN
1174 DO 71 J=1,FIN
1175 E4(I,J)=E(I,J)
1176 71 CONTINUE
1177 IF(K-1)307,303,304
1178 E NDIAG(I1)=IFIN-IRAN
1179 GO TO 6
1180 304 IF(KIRAN.EQ.IRAN)GO TO 33
1181 GO TO 6
1182 33 DO 80 I=1,IFIN
1183 DO 81 J=1,IFIN
1184 E4(I,J)=AP(K-1,I,J)
1185 81 CONTINUE
1186 CALL CNUC(F4,FIN,IFIN,EPS,XE,NE)
1187 IF(NE.EQ.0)GO TO 23
1188 DO 166 I=1,IFIN
1189 IF(K.LE.2)GO TO 166
1190 DO 313 IJ1=1,IFIN
1191 E4(I,IJ1)=AP(K-2,I,IJ1)
1192 166 CONTINUE
1193 CALL CINDP(K,F4,NE,KPOINT,IIDIM,UU)
1194 KW=K-1
1195 WRITE(6,111)I10,KW
1196 111 FORMAT(1H,/,2X,'LEADING E-VECTORS FOR E-VALUE ',I2,/,3X,
1197 ' OF RANK ',I2)
1198 DO 315 I2=1,IFIN
1199 WRITE(6,1243)(XE(I2,J),J=1,NE)
1200 315 CONTINUE
1201 IF((K-1).NE.MULT(I1))GO TO 23
1202 99 WRITE(3,1200)
1203 1248 FORMAT(1H,2X,10(1X,F7.3))

```



```

14556 00 9 J=1,4
14570 IF(I.EQ.0)GO TO 10
14580 XE(I,J)=CMPLX(0.,0.)
14590 GO TO 9
14600 10 X(I,J)=CMPLX(1.,0.)
14610 9 CONTINUE
14620 8 CONTINUE
14630 GO TO 5
14640 C..... RANK NOT 0, DIMENSION OF NULL SPACE LESS THAN ORDER OF MATRIX
14650 7 III1=1
14660 CALL CARRAY(III1,M,N,II2,II2,E,E)
14670 DO 4 J=1,NE
14680 DO 2 K=1,1
14690 EFN(K)=CMPLX(0.,0.)
14700 LI=ICOL(IR+J)
14710 EN(LI)=CMPLX(1.,0.)
14720 DO 3 L=1,IR
14730 LE=ICOL(L)
14740 3 EN(LE)=E(L,IR+J)
14750 DO 4 I=1,4
14760 4 XE(I,J)=EN(I)
14770 5 RETURN
14780 END

```

.....

```

14800 C
14810 C
14820 C
14830 C
14840 C SUBROUTINE CARRAY
14850 C *****
14860 SUBROUTINE CARRAY(MODE,I,J,N,M,S,D)
14870 C..... CONVERT ARRAY FROM SINGLE TO DOUBLE DIMENSION OR V.V.
14880 C..... FOR MODE=1 OR 2 RESP.
14890 COMPLEX S,D
14900 DIMENSION S(1),D(1)
14910 NI=N-1
14920 C..... TEST TYPE OF CONVERSION
14930 IF(MODE-1)99,99,120
14940 C..... CONVERT FROM SINGLE TO DOUBLE DIMENSION
14950 99 IJ=I*J+1
14960 NM=N-IJ+1
14970 DO 110 K=1,J
14980 NM=NM-NI
14990 DO 110 L=1,E
15000 IJ=IJ-1
15010 NM=NM-1
15020 110 U(NM)=S(IJ)
15030 GO TO 140
15040 C..... CONVERT FROM DOUBLE TO SINGLE DIMENSION
15050 120 IJ=0
15060 NM=0
15070 DO 125 K=1,J
15080 DO 125 L=1,E
15090 IJ=IJ+1
15100 NM=NM+1
15110 125 S(IJ)=D(NM)
15120 130 NM=NM+NI
15130 140 RETURN
15140 END

```

.....

```

15150 C
15160 C
15170 C
15180 C
15190 C SUBROUTINE CMFGR
15200 C *****
15210 SUBROUTINE CMFGR(A,M,N,EPS,IRANK,IROW,ICOL)
15220 C..... DETERMINATION OF THE FOLLOWING FOR A M X N MATRIX A
15230 C..... 1 RANK AND LINEARLY INDEPENDANT ROWS AND COLUMNS
15240 C..... 2 FACTORISATION OF SUBMATRIX OF MAX. RANK
15250 C..... 3 NON-BASIC ROWS IN TERMS OF BASIC ONES
15260 C..... 4 BASIC VARIABLES IN TERMS OF FREE ONES
15270 C..... GAUSSIAN ELIMINATION WITH FULL PIVOTING
15280 C..... COMPLEX A,HOLD,PIV,SAVE
15290 DIMENSION A(1),IROW(1),ICOL(1)
15300 C..... INITIALIZE COLUMN INDEX VECTOR SEARCH FIRST PIVOT ELEMENT
15310 1 IRANK=0
15320 PIV=CMPLX(0.,0.)
15330 JJ=0
15340 DO 6 J=1,1
15350 ICOL(J)=J
15360 DO 6 I=1,4
15370 JJ=JJ+1
15380 HOLD=A(JJ)
15390 IF(CABS(PIV)-CABS(HOLD))5,6,6

```

```

15541 5 PIV=HOLD
15542 IR=I
15543 IC=J
15544 6 CONTINUE
15545 C.....INITIALIZE ROW INDEX VECTOR
15546 DO 7 I=1,M
15547 7 IROW(I)=I
15548 C.....SET UP INTERNAL TOLERANCE
15549 TOL=CABS(EPS*PIV)
15550 C.....INITIALIZE ELIMINATION LOOP
15551 NM=N-M
15552 DO 19 NCOL=M,NM,M
15553 C.....TEST FOR FEASIBILITY OF PIVOT ELEMENT
15554 8 IF(CABS(PIV)-TOL)20,20,9
15555 C.....UPDATE RANK
15556 9 IRANK=IRANK+1
15557 C.....INTERCHANGE ROWS IF NECESSARY
15558 JJ=IR-IRANK
15559 IF(JJ)12,12,10
15560 DO 11 J=IRANK,NM,M
15561 I=J+JJ
15562 SAVE=A(I)
15563 A(I)=A(J)
15564 11 A(J)=SAVE
15565 C.....UPDATE ROW INDEX VECTOR
15566 JJ=IROW(IR)
15567 IROW(IR)=IROW(IRANK)
15568 IROW(IRANK)=JJ
15569 C.....INTERCHANGE COLUMNS IF NECESSARY
15570 12 JJ=(IC-IRANK)*M
15571 IF(JJ)15,15,13
15572 13 KK=NCOL
15573 DO 14 J=1,M
15574 I=KK+JJ
15575 SAVE=A(I)
15576 A(I)=A(J)
15577 14 A(J)=SAVE
15578 C.....UPDATE COLUMN INDEX VECTOR
15579 JJ=ICOL(IC)
15580 ICOL(IC)=ICOL(IRANK)
15581 ICOL(IRANK)=JJ
15582 15 KK=IRANK+1
15583 MM=IRANK-M
15584 LL=NCOL+MM
15585 F(MM)16,25,25
15586 C.....TRANSFER CURRENT SUBMATRIX AND SEARCH FOR NEXT PIVOT
15587 16 JJ=LL
15588 SAVE=PIV
15589 PIV=CMPLX(0.,0.)
15590 DO 19 J=KK,M
15591 JJ=JJ+1
15592 HOLD=A(JJ)/SAVE
15593 A(JJ)=HOLD
15594 I=J-IRANK
15595 C.....TEST FOR LAST COLUMN
15596 IF(IRANK-N)17,19,19
15597 17 II=JJ
15598 DO 19 J=KK,N
15599 II=II+M
15600 MM=II-L
15601 A(II)=A(II)-HOLD*A(MM)
15602 IF(CABS(A(II))-CABS(PIV))19,19,18
15603 18 PIV=A(II)
15604 IR=J
15605 IC=I
15606 19 CONTINUE
15607 C.....SET UP MATRIX EXPRESSING ROW DEPENDANCIES
15608 20 IF(IRANK-1)3,25,21
15609 21 IR=LL
15610 DO 24 J=2,IRANK
15611 I=J-1
15612 IR=IR-M
15613 JJ=LL
15614 DO 23 I=KK,M
15615 HOLD=CMPLX(0.,0.)
15616 JJ=JJ+1
15617 MM=JJ
15618 IC=IR
15619 DO 22 L=1,II
15620 HOLD=HOLD+A(MM)-A(IC)
15621 IC=IC-1
15622 22 MM=MM-M
15623 A(MM)=A(MM)-HOLD
15624 24 CONTINUE

```

```

16226 25 IF (N-IPANK) 3,3,26
16227 C.....SET UP MATRIX EXPRESSING BASIC VARIABLES IN TERMS OF FREE
16228 C.....PARAMETERS (HOMOGENEOUS SOLUTION)
16229 26 I=LL
16230 KK=LL+M
16231 DO 30 J=1,IPANK
16232 DO 29 I=KK,NM,M
16233 JJ=I
16234 LL=I
16235 HOLD=CMPLX(0.,0.)
16236 II=J
16237 N7=II-1
16238 IF (II) 29,29,28
16239 28 HOLD=HOLD-A(JJ)*A(LL)
16240 JJ=JJ-M
16241 LL=LL-1
16242 GO TO 27
16243 29 A(LL)=(HOLD-A(LL))/A(JJ)
16244 30 IR=IR-1
16245 31 IF (IR) 25,25
16246 END

```

SUBROUTINE CMATRM

```

16549 SUBROUTINE CMATRM(IID1,IID2,N1,N2,N3,A1,B1,C1)
16550 C.....MULTIPLICATION OF TWO (COMPLEX) MATRICES
16551 C.....COMPLEX A1,B1,C1,C
16552 DIMENSION A1(IID1,20),B1(20,IID2),C1(IID1,IID2)
16553 DO 1 I=1,N1
16554 DO 2 J=1,N3
16555 C=CMPLX(0.,0.)
16556 DO 3 JJ=1,N2
16557 3 C=C+A1(I,JJ)*B1(JJ,J)
16558 C1(I,J)=C
16559 2 CONTINUE
16560 1 CONTINUE
16561 RETURN
16562 END

```

SUBROUTINE CINDP

```

16740 C.....TO DETERMINE WHETHER VECTORS GENERATING THE NULL SPACE OF
16741 C.....(A-LI)+*K CAN BE GENERALISED E-VECTORS
16742 SUBROUTINE CINDP(K,NE,N,KPOINT,IID1,II)
16743 C.....INTEGER IJR,PRT,DSK,OTP
16744 C.....COMPLEX UJ,XE,BB,CC,EE,U,UR,EE
16745 C.....COMMON NRET,RDR,PRT,DSK,OTP,LPR,PAR(4),EPS,NDUM(8)
16746 C.....COMMON A(20,20),B(20,10),C(10,20),D(10,10),N2,NT
16747 C.....COMMON ITP,N,M,NN(10,10),NDEN,GN(10,10,21),CD(21),IRET,NSUM
16748 C.....COMMON AAA(2,20),BBB(20,10),CCC(10,20),NDIM,IC,ICOMPL,JOIM(20)
16749 C.....COMMON JSIGN,MULT(20),WR(20),WI(20),VR(2,20),VI(20,20),NDIAG(2)
16750 DIMENSION EE4(20,20),U(20,1),UR(20,1),EE(20,20)
16751 D=NDIM
16752 DIMENSION XE(20,20),BB(20,10),CC(10,20)
16753 EQUIVALENCE (XE(1,1),AAA(1,1)),(BB(1,1),BBB(1,1)),(CC(1,1),CCC(1,1))
16754 IFIN=NSUM
16755 IID1=20
16756 IID2=1
16757 IF(K,LE.2) GO TO 231
16758 LL=1
16759 C.....CHECK IF (A-LI)+*K-1 = 0
16760 DO 213 J=1,NE
16761 DO 214 I=1,IFIN
16762 214 U(I,1)=XE(I,J)
16763 CALL CMATRM(IID1,IID2,IFIN,IFIN,IID2,EE,U,UR)
16764 DO 215 I1=1,IFIN
16765 IF(CABS(UR(I1,1)).LE.EPS) GO TO 215
16766 GO TO 213
16767 215 CONTINUE
16768 KIND=X(LL)=J
16769 LL=LL+1
16770 213 CONTINUE
16771 IF(LL.EQ.1) GO TO 216
16772 IF(NE.EQ.1) GO TO 220
16773 C.....DELETE UNSUITABLE E-VECTORS FROM E-VECTOR MATRIX XE
16774 DO 217 I=LL-1,1
16775 LX=KINDEX(I)-I+1

```

```
17111 F(LX, EQ, VE) GO TO 221
17112 DO 218 I1=1, IFIN
17113 DO 219 I2=LX, NE-1
17114 XE(I1, I2)=XE(I1, I2+1)
17115 218 CONTINUE
17116 219 NE=NE-1
17117 217 CONTINUE
17118 C..... PICK THE LEADING E-VECTORS ALREADY OBTAINED FROM UU
17119 216 F(NE, LE, 1) GO TO 210
17120 231 KJ=IJDIM-KPOINT
17121 IF(KJ) 696, 696, 695
17122 695 KSTART=1
17123 F(KPOINT, EQ, 1) GO TO 698
17124 DO 699 I=1, KPOINT-1
17125 699 KSTART=KSTART+JDIM(I)
17126 698 KJJ=KSTART
17127 DO 701 I=1, KJ
17128 DO 702 J=1, IFIN
17129 E4(J, I)=UJ(J, KJJ)
17130 702 CONTINUE
17131 KJJ=KJJ+JDIM(I)
17132 701 CONTINUE
17133 696 I11=NE
17134 I10=1
17135 INE1=KJ+1
17136 INE2=1
17137 278 F(K, ST, 2) GO TO 228
17138 DO 262 I=1, IFIN
17139 262 U(I, 1)=XE(I, INE2)
17140 GO TO 263
17141 C..... GENERATE PRESENT LEADING E-VECTORS AND CHECK IF THEY FORM
17142 C..... A L.I. SET WITH THE ALREADY OBTAINED E-VECTORS
17143 228 DO 241 I=1, IFIN
17144 241 UR(I, 1)=XE(I, INE2)
17145 CALL CMAYM(I101, I102, IFIN, IFIN, I102, EE, UR, U)
17146 263 II=INE1
17147 DO 244 I2=1, IFIN
17148 244 E4(I2, II)=U(I2, 1)
17149 F(II, LE, 1) GO TO 224
17150 CALL CRANK(E4, EPS, IFIN, II, IRAN)
17151 IF(IRAN, EQ, II) GO TO 224
17152 IF(INE1, EQ, (NE+KJ)) GO TO 225
17153 DO 226 I3=1, IFIN
17154 DO 227 I4=1, NE-1
17155 227 XE(I3, I4)=XE(I3, I4+1)
17156 226 CONTINUE
17157 225 NE=NE-1
17158 GO TO 255
17159 224 INE1=INE1+1
17160 INE2=INE2+1
17161 255 I11=I11+1
17162 F(LE, LE, I11) GO TO 278
17163 210 RETURN
17164 END
17165 C.....
17166 C
17167 C
17168 C
17169 C
```



```

17700 C .....
17710 C
17720 C
17730 C
17740 C
17750 C SUBROUTINE RREAL
17760 C *****
17770 SUBROUTINE RREAL
17780 REAL KEEP, I
17790 INTEGER RDR, PRT, DSK, DTP
17800 COMMON NR, RT, RDR, PRT, DSK, DTP, LPR, PAR(4), EPS, NDUM(8)
17810 COMMON A(20,20), B(20,10), C(10,20), D(10,10), N2, NT
17820 COMMON I, TYPE, N, M, NN(10,10), NDEN, GN(10,10,21), CD(21), IRET, NSUM
17830 COMMON AA(20,20), BB(20,10), CC(10,20), NDIM, IC, ICOMPL, JDIM(20), KINDE
17840 COMMON JSIGN, MULT(20), WR(20), WI(20), VR(2,2), VI(20,20), NDIAG(20)
17850 K=0
17860 K3=0
17870 IC=0
17880 C..... CHECK FOR DISTINCT E-VALUES
17890 IFIN=NSUM
17900 WRITE(6,136)
17910 1336 FORMAT(1H,1X,'* E-VALUES REAL , MATRICES WITH REAL ELEMENTS'
17920 DO 501 I=1,IFIN-1
17930 IF(MULT(I))501,501,402
17940 402 K=K+1
17950 K3=K3+1
17960 DO 601 J=I+1,IFIN
17970 IF(ABS(WR(I)-WR(J)).GT.EPS)GO TO 601
17980 MULT(I)=MULT(I)+1
17990 MULT(K+1)=0
18000 IC=1
18010 IF(J-I-1)666,666,610
18020 610 KEEP1=WR(K+1)
18030 KEEP2=WI(K+1)
18040 WR(K+1)=WR(J)
18050 WR(J)=KEEP1
18060 666 K=K+1
18070 601 CONTINUE
18080 501 CONTINUE
18090 IF(MULT(IFIN).GT.0)K3=K3+1
18100 WRITE(6,222)K3
18110 1304 CALL JORDAN(K3,LSTOP)
18120 222 FORMAT(1H,1X,'DISTINCT E-VALUES ',I2)
18130 1304 LSTOP=6,10,6,2
18140 1306 WRITE(6,543)
18150 DO 670 I=1,IFIN
18160 WRITE(6,543)(VR(I,J),J=1,IFIN)
18170 670 CONTINUE
18180 670 CONTINUE
18190 543 FORMAT(1H,1X,' TRANSFORMATION MATRIX FOR JORDAN FORM')
18200 500 FORMAT(1H,10(1X,F7.3))
18210 CALL TRANS(K3)
18220 IF(IRET.EQ.1)GO TO 2000
18230 CALL SWIT
18240 2000 RETURN
18250 END
18260 C .....
18270 C
18280 C
18290 C
18300 C
18310 C..... SUBROUTINE TRANS
18320 C *****
18330 SUBROUTINE TRANS(K3)
18340 INTEGER RDR, PRT, DSK, DTP
18350 COMMON NR, RT, RDR, PRT, DSK, DTP, LPR, PAR(4), EPS, NDUM(8)
18360 COMMON A(20,20), B(20,10), C(10,20), D(10,10), N2, NT
18370 COMMON I, TYPE, N, M, NN(10,10), NDEN, GN(10,10,21), CD(21), IRET, NSUM
18380 COMMON AA(20,20), BB(20,10), CC(10,20), NDIM, IC, ICOMPL, JDIM(20), KINDE
18390 COMMON JSIGN, MULT(20), WR(20), WI(20), TT(2,20), VI(20,20), NDIAG(20)
18400 DIMENSION TTT(20,20), IV(20)
18410 DIMENSION RIND(20), AT(20,20)
18420 C..... TRANSFORM B MATRIX
18430 IFIN=NSUM
18440 DO 222 I=1,IFIN
18450 DO 223 J=1,IFIN
18460 223 AT(I,J)=TT(I,J)
18470 222 CONTINUE
18480 C..... INVERT TRANSFORMATION MATRIX
18490 IDIM1=20
18500 IFAIL=1
18510 CALL FC1AAF(AT, IDIM1, IFIN, TTT, IDIM1, RIND, IFAIL)
18520 WRITE(6,1324)
18530 IF(IFAIL)999,399,999
18540 999 WRITE(6,1020)

```

```

18550 1820 FORMAT(1H , ' FAILURE IN F01AAF')
18555 IRRR=1
18570 GO TO 51
18580 399 DO 1 22 IL=1,IFIN
18590 WRITE (6,1,23) (TTT(IL,JL),JL=1,IFIN)
18600 1022 CONTINUE
18610 1023 FORMAT(1H ,10(1X,F7.3))
18620 1024 FORMAT(1H ,/,2X,' INVERSE OF TRANSFORMATION MATRIX')
18630 IID1=20
18640 IID2=10
18650 CALL MATRM(IID1,IID2,IFIN,IFIN,M,TTT,B,BE)
18660 WRITE (6,1896)
18670 1896 FORMAT(1H ,1X,' TRANSFORMED B MATRIX')
18680 DO 7 1=1,IFIN
18690 WRITE (6,1,25) (BB(I,J1),J1=1,M)
18700 71 CONTINUE
18710 1025 FORMAT(1H ,10(1X,F7.3))
18720 C... TRANSFORM C
18730 CALL MATRM(IID2,IID1,N,IFIN,IFIN,C,TT,CC)
18740 WRITE (6,2,16)
18750 2,16 FORMAT(1H ,1X,' TRANSFORMED C MATRIX')
18760 DO 72 1=1,N
18770 WRITE (6,1,25) (CC(I,J1),J1=1,IFIN)
18780 72 CONTINUE
18790 IID1=20
18800 IID2=20
18810 CALL MATRM(IID1,IID2,IFIN,IFIN,IFIN,TTT,A,AT)
18820 CALL MATRM(IID1,IID2,IFIN,IFIN,IFIN,AT,TT,AA)
18830 WRITE (6,1,35)
18840 DO 83 1=1,IFIN
18850 WRITE (6,1,33) (AA(I,J),J=1,IFIN)
18860 1035 FORMAT(1H ,/,,' TRANSFORMED A MATRIX IN JORDAN FORM')
18870 833 FORMAT(1H ,/,10(1X,F7.3))
18880 837 CONTINUE
18890 NDIM=IFIN
18900 615 IF (IC) 416,416,431
18910 C... DISTINCT E-VALUES JORDAN FORM DIAGONAL
18920 C... RRRR IS NUMBER OF SUBSYSTEMS IN WHICH SYSTEM IS DIVIDED
18930 416 IRRR=IFIN
18940 NDIM=IFIN
18950 KBLOC=IFIN
18960 DO 250 1J=1,KBLOC
18970 NDIAG(IJ)=1
18980 250 JDIM(IJ)=1
18990 GO TO 431
19000 431 RRRR=<3
19010 431 KKK=1
19020 KINDEX(1)=1
19030 DO 302 1=1,IRRR
19040 DO 3 3 12=1,NDIAG(I)
19050 KINDEX(KKK+12)=KINDEX(KKK+12)+JDIM(KKK+12-1)
19060 303 CONTINUE
19070 KKK=KKK+NDIAG(I)
19080 312 CONTINUE
19090 LSTART=1
19100 DO 51 1JJ=1,IRRR
19110 C... STEP 1, SET V(I)=0, I=1, R
19120 556 IREP=1
19130 184 LFINI=LSTART+NDIAG(JJ)-1
19140 DO 55 1J=LSTART,LFINI
19150 55 IV(J)=0
19160 C... STEPS 2 AND 3
19170 C... KIMAX IS MAXIMUM SIZE OF BLOCK FOR WHICH IV=0
19180 54 KIMAX=1
19190 KSTOP=0
19200 DO 56 1J=LSTART,LFINI
19210 IF (IV(J)) 56,57,56
19220 57 KSTOP=1
19230 IF (KIMAX-JDIM(J)) 58,58,56
19240 56 KIMAX=JDIM(J)
19250 C... JSIGN IS J ST. JDIM(J) IS MAX
19260 JSIGN=J
19270 56 CONTINUE
19280 IF (KSTOP) 55,55,59
19290 C... STEP 4
19300 59 KISIGN=KINDEX(JSIGN)+JDIM(JSIGN)-1
19310 DO 61 1=1,M
19320 IF (ABS(BB(KISIGN,1))-EPS) 61,61,62
19330 61 CONTINUE
19340 GO TO 91
19350 62 IV(JSIGN)=1
19360 +SG=
19370 +SG1=1+LSTART-1
19380 DO 63 1=LSTART,LFINI
19390 IF (IV(1)) 53,64,63

```


5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

```

DIMENSION ILS(20)
IST=KINDEX(JSIGN)+JDIM(JSIGN)-1
IF(IST.GE.NDIM)GO TO 11
DO 11 I=IST,NDIM-1
DO 12 J=1,NDIM
  AB(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
  CC(IJ,I)=CC(IJ,I+1)
1 CONTINUE
DO 13 J=IST,NDIM
DO 14 I=1,NDIM
  AA(I,J)=AA(I,J+1)
13 CONTINUE
11 NDIM=NDIM-1
  JDIM(JSIGN)=JDIM(JSIGN)-1
  ILS=0
  IF(JDIM(JSIGN))7,7,8
  DO 9 I=JSIGN,19
  ILS(I)=ILS(I+1)
  JDIM(I)=JDIM(I+1)
  IL=1
  DO 6 I=JSIGN+1,19
  KINDEX(I)=KINDEX(I+1L)-1
RETURN
END

```

.....

```

SUBROUTINE MATRM
*****
SUBROUTINE MATRM(IID1,IID2,N1,N2,N3,A1,B1,C1)
.....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
  C=C+A1(I,JJ)*B1(JJ,J)
C1(I,J)=C
2 CONTINUE
1 CONTINUE
RETURN
END

```

.....

C

```

27150 SUBROUTINE ARANK(ARANK, EPS, MS, M, IRAN)
27160 C..... SUBROUTINE TO CALCULATE THE RANK OF AN MSXM MATRIX
27170 C..... USES GAUSS ELIMINATION WITH FULL PIVOTING
27180 DIMENSION ARANK(20,20), IR(20), IC(20)
27190 C..... PRESET ROW AND COLUMN INTERCHANGES
27200 DO 100 I=1, MS
27210 IR(I)=I
27220 DO 110 I=1, M
27230 IC(I)=I
27240 IRAN=M
27250 MM=M-1
27260 C..... BEGIN ELIMINATION PROCEDURE
27270 DO 200 LS=1, MM
27280 AMAG=0.
27290 C..... SEARCH FOR PIVOT
27300 DO 120 I=LS, MS
27310 DO 120 J=LS, M
27320 ADUM=ABS(ARANK(IR(I), IC(J)))
27330 IF(ADUM-AMAG)120,120,105
27340 105 LS=I
27350 JS=J
27360 AMAG=ADUM
27370 120 CONTINUE
27380 C..... TEST FOR COMPLETION
27390 IF(AMAG-EPS)125,125,130
27400 125 IRAN=LS-1
27410 GO TO 300
27420 130 CONTINUE
27430 C..... INTERCHANGE ROW AND COLUMN INDICES
27440 IT=IR(LS)
27450 IR(LS)=IR(JS)
27460 IR(JS)=IT
27470 IT=IC(LS)
27480 IC(LS)=IC(JS)
27490 IC(JS)=IT
27500 C..... ELIMINATE IC(LS) COLUMN
27510 LSP=LS+1
27520 DO 150 I=LSP, MS
27530 Q=ARANK(IR(I), IC(LS))/ARANK(IR(LS), IC(LS))
27540 DO 150 J=LSP, M
27550 ARANK(IR(I), IC(J))=ARANK(IR(I), IC(J))-Q*ARANK(IR(LS), IC(J))
27560 150 CONTINUE
27570 C..... PATCH UP RANK TEST
27580 IF(LS-MM)170,160,160
27590 160 AMAG=0.
27600 DO 162 I=LSP, MS
27610 ADUM=ABS(ARANK(IR(I), IC(M)))
27620 IF(ADUM-AMAG)162,162,161
27630 161 AMAG=ADUM
27640 162 CONTINUE
27650 IF(AMAG-EPS)165,165,170
27660 165 IRAN=M-1
27670 170 CONTINUE
27680 200 CONTINUE
27690 300 CONTINUE
27700 RETURN
27710 END
.....
27720 C.....
27730 C.....
27740 C.....
27750 C.....
27760 C..... SUBROUTINE JORDAN.
27770 C..... *****
27780 C..... TO CALCULATE THE JORDAN FORM OF A MATRIX WITH MULTIPLE
27790 C..... E-VALUES BY GENERATING A SET OF GENERALISED E-VECTORS
27800 C..... USES SSP SUBROUTINES ARRAY AND MFGR AND
27810 C..... RRANK, INDEP
27820 C..... SUBROUTINE JORDAN(K3, LSTOP)
27830 INTEGER J, PRT, DSK, JTP
27840 COMMON NRET, RDR, PRT, DSK, DTP, LPR, PAR(4), EPS, NDUM(8)
27850 COMMON A(20,20), B(20,10), C(10,20), D(10,10), N2, NT
27860 COMMON ITYPE, N, M, NN(10,1), NDEN, GN(10,10,21), CD(21), IRET, NSUM
27870 COMMON X(20,20), BB(20,1), CC(10,20), NDIM, IC, ICOMPL, JDIM(20), KIN
27880 COMMON JSIGN, MULT(20), WR(20), WI(20), UU(20,20), VI(20,20), NOIAG(20)
27890 DIMENSION AP(5,20,20), U(20,1), UR(20,1), EF(20,20)
27900 C.....
27910 DIMENSION E4(20,20), E(20,20)
27920 EQUIVALENCE (GN(1,1,1), AP(1,1,1))
27930 EQUIVALENCE (BB(1,1), E4(1,1)), (VI(1,1), E(1,1))
27940 IF IN=NSUM
27950 LSTOP=0
27960 ILO=M-1
27970 IVEC=1
27980 IRAN=0

```

```

27900 L1 SUM=0
27901 KPOINT=1
27902 K1=0
27903 I1=1
27904 C... BEGIN LOOP FOR EACH E-VALUE
27905 DO 1 I2=1, K3
27906 ITEST=0
27907 K1=K1 +MULT(I1)
27908 K=1
27909 DO 1 I=1, IFIN
27910 DO 2 J=1, IFIN
27911 IF (I EQ J) GO TO 3
27912 AP(1, I, J)=A(I, J)
27913 EP(1, J)=AP(1, I, J)
27914 GO TO 2
27915 3 AP(1, I, J)=A(I, J)-WR(K1)
27916 EP(I, J)=AP(1, I, J)
27917 2 CONTINUE
27918 1 CONTINUE
27919 NDIAG(I1)=0
27920 22 DO 11 IJ1=1, IFIN
27921 DO 12 IJ2=1, IFIN
27922 E(IJ1, IJ2)=AP(K, IJ1, IJ2)
27923 4(IJ1, IJ2)=E(IJ1, IJ2)
27924 12 CONTINUE
27925 11 CONTINUE
27926 IF (MULT(I1)-1)200, 200, 40
27927 200 K=2
27928 NDIAG(I1)=1
27929 GO TO 33
27930 C... FIND RANK OF AP(K, I, J)
27931 40 KIRAN=IRAN
27932 CALL RRANK(E4, EPS, IFIN, IFIN, IRAN)
27933 DO 70 I=1, IFIN
27934 DO 71 J=1, IFIN
27935 71 E4(I, J)=E(I, J)
27936 70 CONTINUE
27937 IF (K-1)303, 303, 304
27938 303 NDIAG(I1)=IFIN-IRAN
27939 GO TO 6
27940 304 IF (KIRAN EQ IRAN) GO TO 33
27941 GO TO 6
27942 DO 80 I=1, IFIN
27943 DO 81 J=1, IFIN
27944 81 E4(I, J)=AP(K-1, I, J)
27945 80 CONTINUE
27946 CALL NULL(E4, IFIN, IFIN, EPS, XI, NE)
27947 IF (NE EQ 0) GO TO 23
27948 DO 166 I=1, IFIN
27949 IF (K LE 2) GO TO 166
27950 DO 163 IJ1=1, IFIN
27951 163 4(I, IJ1)=AP(K-2, I, IJ1)
27952 166 CONTINUE
27953 CALL INDEX(K, E4, NE, KPOINT, IIDIM)
27954 KW=K-1
27955 WRITE(6, 347) I10, KW
27956 347 FORMAT(1H ,7, 2X, 'LEADING E-VECTORS FOR E-VALUE ', I2, ', 3X, ' OF
27957 (I2)
27958 DO 315 I2=1, IFIN
27959 WRITE(6, 111) (XE(I2, J), J=1, NE)
27960 315 CONTINUE
27961 IF ((K-1) NE LE, MULT(I1)) GO TO 23
27962 99 WRITE(3, 100)
27963 111 FORMAT(1H ,10(1X, F7.3))
27964 100 FORMAT(1H ,7, ' WRONG CALCULATION OF RANK. PROCEDURE STOPPED
27965 LSTOP=1
27966 RETURN
27967 23 LS=1
27968 DO 14 I2=1, IFIN
27969 14 U(I2, 1)=XE(I2, LS)
27970 11 IND=1
27971 C... BEGIN PROCEDURE FOR CALCULATION OF GENERALISED E-VECTOR CH/
27972 C... WITH LEADING E-VECTOR XE(I, LS)
27973 IF (MULT(I1) EQ 1) GO TO 18
27974 IF (IND, CF, K-1) GO TO 18
27975 DO 13 I2=1, IFIN
27976 DO 15 I3=1, IFIN
27977 15 4(I2, I3)=AP(K-IND-1, I2, I3)
27978 16 CONTINUE
27979 13 CONTINUE
27980 IID1=20
27981 IID2=1
27982 NN3=1
27983 CALL MATRY(IID1, IID2, IFIN, IFIN, NN3, E, U, UR)
27984 DO 17 I2=1, IFIN

```

```

17 UU(I2, IVEC)=UR(I2, 1)
+VEC=IVEC+1
IND=IND+1
IF(IND.EQ K-1)GO TO 18
GO TO 20
18 UU(I2, IVEC)=U(I2, 1)
IVEC=IVEC+1
JDIM(IIDIM)=K-1
LLSUM=LLSUM+JDIM(LLDIM)
IIDIM=IIDIM+1
IF(N.GT 1)GO TO 21
GO TO 109
21 LLS=LLS+1
IF(LLS.LE NE)GO TO 13
C... GENERATION OF CHAIN COMPLETED
109 IF(LLSUM-MULT(I1))305,306,306
306 LLSUM=
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=2N
O2=2N
CALL MATRY(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)
E(IJ, I6)=E4(IJ, I6)
31 CONTINUE
GO TO 40
9 I1=I1+MULT(I1)
10 KPOINT=KPOINT+NDIAG(I1)
RETURN
END

```

... SUBROUTINE NULL

```

SUBROUTINE NULL(E, M, N, TOL, XE, NE)
C... TO DETERMINE BASIS VECTORS AND DIMENSION OF NULL SPACE
C... CALLS SSP SUBROUTINES MFG, ARAY
DIMENSION L(20, 20), XE(20, 20), IR(20), ICOL(20), EN(20)
II1=2N
II2=2N
CALL ARAY(II1, M, N, II2, II2, E, E)
CALL MFG(L, M, N, TOL, IR, IR, 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 MATRIX
7 II1=1
CALL ARAY(II1, M, N, II2, II2, E, E)
DO 4 J=1, NE
DO 2 K=1, M
EN(K)=0
LI=ICOL(I3+J)
EN(LI)=1
GO 3 L=1, M
LE=ICOL(L)
EN(LE)=1(L, IR+J)
DO 4 I=1, M
XE(I, J)=EN(I)
5 RETURN
END

```

... SUBROUTINE ARRAY

```

C *****
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.
C DIMENSION S(1),D(1)
C N=N-1
C ..... TEST TYPE OF CONVERSION
C IF(MODE=1)100,100,120
C ..... CONVERT FROM SINGLE TO DOUBLE PRECISION
C J=J+1
C NM=N-J+1
C DO 110 K=1,J
C NM=NM-NI
C DO 110 L=1,I
C IU=IU-1
C NM=NM-1
C 110 D(NM)=S(IJ)
C GO TO 14
C ..... CONVERT FROM DOUBLE TO SINGLE DIMENSION
C 120 IU=0
C NM=0
C DO 120 K=1,J
C DO 125 L=1,I
C IU=IU+1
C NM=NM+1
C 125 S(IJ)=D(NM)
C 130 NM=NM+NI
C 140 RETURN
C END

```

.....

```

C ..... SUBROUTINE MFGR
C *****
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 INDEPENDANT 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
C DIMENSION A(1),IROW(1),ICOL(1)
C ..... INITIALIZE COLUMN INDEX VECTOR SEARCH FIRST PIVOT ELEMENT
C 4 IRANK=0
C 5 PIV=0.
C 6 JJ=1
C 7 DO 6 J=1,N
C 8 ICOL(J)=J
C 9 DO 6 I=1,I
C 10 JJ=JJ+1
C 11 HOLD=A(JJ)
C 12 IF(ABS(PIV)-ABS(HOLD))5,5,6
C 13 5 PIV=HOLD
C 14 I=I
C 15 IC=J
C 16 6 CONTINUE
C ..... INITIALIZE ROW INDEX VECTOR
C DO 7 I=1,I
C 7 IROW(I)=I
C ..... SET JP INTERNAL TOLERANCE
C 20 TOL=ABS(EPS*PIV)
C ..... INITIALIZE ELIMINATION LOOP
C NM=N-M
C DO 19 NCOL=M,NM,M
C ..... TEST FOR FEASIBILITY OF PIVOT ELEMENT
C 8 IF(ABS(PIV)-TOL)20,20,9
C ..... UPDATE RANK
C 9 IRANK=IRANK+1
C ..... INTERCHANGE ROWS IF NECESSARY
C JJ=IR-IRANK
C IF(JJ)12,2,10
C 10 DO 11 J=IRANK,NM,M
C I=J+JJ
C SAVE=A(J)
C A(J)=A(I)
C 11 A(I)=SAVE
C ..... UPDATE ROW INDEX VECTOR
C 10 JJ=IROW(IR)
C IROW(IR)=IROW(IRANK)
C IROW(IRANK)=JJ
C ..... INTERCHANGE COLUMNS IF NECESSARY
C 12 JJ=(IC-IRANK)+M
C IF(JJ)15,15,13
C 13 KK=NCOL-

```



```

445000 00 1 J=1,M
445010 I=KK+JJ
445020 SAVE=A(KK)
445030 T(KK)=A(I)
445040 KK=KK-1
445050 14 A(I)=SAVE
445060 C... UPDATE COLUMN INDEX VECTOR
445070 JJ=ICOL(I)
445080 ICOL(IC)=ICOL(IRANK)
445090 ICOL(IRANK)=JJ
445100 15 KK=IRANK+1
445110 MM=IRANK-4
445120 LL=NCOL+MM
445130 IF(MM)16,25
445140 C... TRANSFORM CURRENT SUBMAT IX AND SEARCH FOR NEXT PIVOT
445150 16 JJ=L
445160 SAVE=PIV
445170 PIV=A(JJ)
445180 DO 19 J=KK,M
445190 JJ=JJ+1
445200 HOLD=A(JJ)/SAVE
445210 A(JJ)=HOLD
445220 L=J-IRANK
445230 C... TEST FOR LAST COLUMN
445240 IF(IRANK-N)17,19,19
445250 17 II=JJ
445260 DO 19 I=KK,M
445270 MM=II+M
445280 MM=II-L
445290 A(II)=A(II)-HOLD*A(MM)
445300 IF(ABS(A(II))-ABS(PIV))19,19,18
445310 18 P.V=A(II)
445320 IR=J
445330 IC=I
445340 19 CONTINUE
445350 C... SET UP MATRIX EXPRESSING ROW DEPENDENCIES
445360 20 IF(IRANK-1)3,25,21
445370 21 IR=LL
445380 DO 24 J=2,IRANK
445390 IR=J-1
445400 IR=IR-M
445410 JJ=LL
445420 DO 23 I=KK,M
445430 HOLD=A(I)
445440 JJ=JJ+1
445450 MM=JJ
445460 IC=IR
445470 DO 22 L=1,II
445480 HOLD=HOLD+A(MM)*A(IC)
445490 IC=IC-1
445500 22 MM=MM-4
445510 23 A(MM)=A(MM)-HOLD
445520 24 CONTINUE
445530 25 IF(N-IRANK)3,3,26
445540 C... SET UP MATRIX EXPRESSING BASIC VARIABLES IN TERMS OF FREE
445550 C... PARAMETERS (HOMOGENEOUS SOLUTION)
445560 26 IR=LL
445570 KK=LL+M
445580 DO 31 J=1,IPANK
445590 DO 29 I=KK,NM,M
445600 JJ=I
445610 LL=I
445620 HOLD=0.
445630 II=J
445640 27 IF(II)29,39,28
445650 28 HOLD=HOLD-A(JJ)*A(LL)
445660 JJ=JJ-M
445670 LL=LL-1
445680 GO TO 27
445690 29 A(LL)=(HOLD-A(LL))/A(JJ)
445700 30 IR=IR-1
445710 31 RETURN
445720 END
445730 C.....
445740 C.....
445750 C.....
445760 C... SUBROUTINE INDEP
445770 *****
445780 C..... TO DETERMINE WHETHER VECTORS GENERATING THE NULL SPACE OF
445790 (A..I)**K CAN BE GENERALISED S-VECTORS
445800 SUBROUTINE INDEP(K,EP,NE,KPOINT,IIDIM)
445810 INTEGER K,R,PRT,DSK,DTP
445820 COMMON NRET,RDR,PRT,DSK,DTP,LPR,PAR(4),EPS,NDUM(8)

```

```

COMMON A(2,2),B(2,1),C(1,2),D(1,1),N2,NT
COMMON ITYPE,N,M,NN(10,1),NDEN,GN(10,1),CD(21),IRET,NSUM
COMMON XE(20,20),BB(20,10),CC(10,20),NDIM,IC,ICOMPL,JDIM(20),KIND
COMMON USIGN,MULT(2),WR(2),WI(2),UU(2,20),VI(20,20),NDIAG(20)
DIMENSION I4(20,20),U(20,1),UR(20,1),EE(2,20)
EQUIVALENCE (CC(1,1),E4(1,1))
IFIN=NSUM
IID1=20
IID2=1
IF(K.LL,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
KIND X(LL)=J
LL=LL+1
213 CONTINUE
IF(LL.EQ.1)GO TO 216
IF(N.EQ.1)GO TO 220
C..... DELETE UNSUITABLE E-VECTORS FROM E-VECTOR MATRIX XE
DO 217 I=1,LL-1
LX=K NDEX(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 700 I=1,KJ
DO 701 J=1,IFIN
E4(J,I)=UU(J,KJJ)
701 CONTINUE
KJJ=KJJ+JDIM(I)
700 CONTINUE
696 I11=N
I1=1
NE1=KJ+1
INE2=1
278 IF(K,3T,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,IFAN)
IF(I,AN.EQ.1)GO TO 224
IF(INE1,2J,(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 NE1=NE1+1
INE2=INE2+1
255 I10=I10+1
IF(I10,L,I11)GO TO 278
210 RETURN
NO
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 variables 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.