TR R176–94
October 1994

RASP-IDENT
Subspace Model Identification Programs

V. Sima & A. Varga

Institut für Informatik, Bukarest & DLR, Oberpfaffenhofen

Deutsche Forschungsanstalt für Luft- und Raumfahrt e. V.
DLR – Oberpfaffenhofen
Institut für Robotik & Systemdynamik
Entwurfsorientierte Regelungstechnik (Prof. G. Grübel)
D-82230 Weßling

CONTENTS

1. Introduction 1

2. The State Intersection Class of Subspace Model Identification 3

RPIMN4 - N4SID Deterministic-Stochastic Identification 4

3. The MOESP Class of Subspace Model Identification 13

RPIMOE - Ordinary MOESP Identification 16

RPIMPI - Ordinary MOESP Identification with Past Inputs 22

RPIMPO - Ordinary MOESP Identification with Past Inputs and Outputs 28

RPIMRS - Ordinary MOESP Identification with Reconstructed States and Past Inputs 34

RPIMKG - MOESP-SI Deterministic-Stochastic Identification 40

4. Auxiliary Tools for System Identification 49

IMMPID - Estimation of Markov Parameters from Input-Output Data 50

IMSHFT - I/O Data Shifting to Compensate Input Vector Dead-Times 57

IMICID - Estimation of the Initial Conditions 59

IMCLID - Closed-Loop Identification of Linear Systems 63

RPIMIU - Simulation of Linear Time-Invariant Discrete-Time Systems 68

Appendix 1. RASP-IDENT Driver Routines 73

Appendix 2. SLICOT Compatible Driver Routines 75

Appendix 3. SLICOT Compatible Computational and Auxiliary Routines 76

Appendix 4. Called RASP-MODRED, LAPACK and BLAS Routines 78

1. Introduction

Subspace Model Identification (SMI) is a recently developed promissing branch in system identification. One feature that makes this approach so appealing is that SMI formulates the identification of a linear time-invariant (LTI) system as a column space approximation problem followed by a set of linear least-squares problems. As a consequence, instead of using iterative optimisation schemes as are required in the classical parametric model identification approach documented e.g. in /1/, SMI schemes use basic numerical linear algebra tools, such as QR factorization and SVD. The attractiveness of SMI techniques is further increased by the limited number of parameters (essentially only one) the user has to select to determine the model structure and this without any restriction on model generality. For a further discussion of the features of SMI we refer to the specialized literature, such as /2/, /3/.

The RASP-IDENT package includes implementations of two families of SMI techniques to identify an LTI system. The first one is referred to as the State Intersection (SI) class of SMI techniques. The implemented SI method is described in /2/. The second one is referred to as the Multivariable Output Error State Space (MOESP) class of techniques. The implemented MOESP methods are described in /3/, /5/ and /6/.

In addition to the identification facilities, auxiliary routines are available that allow to detect dead-time and to shift appropriately the input-output signals, to estimate the initial state, to identify the system while operating in closed-loop, to simulate discrete-time LTI systems.

Literature:

/1/ Ljung L.
System Identification Toolbox.
The Math Works, Inc., 1992.

/2/ Van Overschee P. and De Moor B.
N4SID: Two subspace algorithms for the identification of combined deterministic-stochastic systems.
Automatica, Vol.30, No.1, pp.75-93, 1994.

/3/ Verhaegen M.
Identification of the deterministic part of MIMO state space models given in innovations form from input-output data.
Automatica, Vol.30, No.1, pp.61-74, 1994.

/4/ Moonen M., De Moor B., Vandenberghe L. and Vandewalle J.
On- and off-line identification of linear state-space models.
Int. J. Control, Vol. 49, pp. 219-232, 1989.

/5/ Verhaegen M. and Dewilde P.
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.
Int. J. Control, 56, pp. 1187-1210, 1992.

/6/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.

2. The State Intersection Class of Subspace Model Identification

The main feature of the State Intersection class of SMI techniques is the determination of the state sequence of the LTI system to be identified or of an observer to reconstruct its state sequence via the intersection of the row spaces of matrices constructed from "past" and "future" input-output data. The basic idea was first described in /1/. An extension of this idea led to the N4SID algorithm developed in /2/. This algorithm allows to identify LTI state space models in so-called innovation form. This form considers the additive perturbations acting on the output of the system to be modelled in the following way:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Kv_k \\ y_k &= Cx_k + Du_k + v_k \end{aligned}$$

where v_k is a zero-mean white noise sequence and u_k is deterministic input sequence (perfectly known to the user), that is statistically independent from the user. Both the basic and the extended variant of this class produce statistically consistent and efficient estimates when the stipulated assumptions hold.

The following subroutine is available for the SI class of SMI:

RPIMN4 computes, by using the basic N4SID algorithm, consistent and statistically efficient estimates of the matrices (A,B,C,D) of a state space model when v is generated by an innovation model.

/1/ Moonen M., De Moor B., Vandenberghe L. and Vandewalle J.
On- and off-line identification of linear state-space models.
Int. J. Control, Vol. 49, pp. 219-232, 1989.

/2/ Van Overschee P. and De Moor B.
N4SID: Two subspace algorithms for the identification of combined deterministic-stochastic systems.
Automatica, Vol.30, No.1, pp.75-93, 1994.

SUBROUTINE RPIMN4

N4SID Deterministic-Stochastic Identification

Procedure purpose:

This subroutine determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices and, optionally, the noise covariance matrices and the Kalman gain matrix using the "subspace state space system identification" (N4SID) algorithm. It is assumed that the input is a zero-mean white noise process.

The model structure is :

$$x(k+1) = Ax(k) + Bu(k) + w(k)$$
$$y(k) = Cx(k) + Du(k) + e(k)$$

where w(k) and e(k) are zero-mean white noise sequences, independent of the input u(k). Matrix D could be zero.

The Kalman predictor structure is :

$$\bar{x}(k+1) = A\bar{x}(k) + Bu(k) + K(y(k) - C\bar{x}(k) - Du(k))$$
$$\bar{y}(k) = C\bar{x}(k) + Du(k)$$

where K is the Kalman gain matrix stored in FK.
There is an option for sequential input-output data processing.

Usage:

```
CALL RPIMN4( U, NSMP, M, Y, L, NOBL, TOL, WITHD, WITHCV, WITHK, CONCT,
             LAST, CTRL, N, A, B, C, D, FK, R, IWORK, RWORK, LRWORK, *)
```

U : IN, DOUBLE (NSMP,M)
system input data sequence matrix U = [u_1 u_2 ... u_m]. Column j of U contains the NSMP values of the j-th input component for consecutive time increments.

NSMP : IN, INTEGER
number of rows of matrices U and Y (number of samples) (When sequential data processing is used, NSMP is the number of samples of the current data batch.)
NSMP >= 2*(M+L+1)*NOBL - 1, for non-sequential processing of the data;
NSMP >= 2*NOBL, for sequential processing of several data batches;
NSMP = 0, and LAST = .TRUE., is allowed if the upper

triangular factor of the QR factorization is already available in the array R, and only the system matrices are to be computed.

The total number of samples when calling with LAST = .TRUE. should be at least 2*(M+L+1)*NOBL - 1.

M : IN, INTEGER
dimension of system input vector.

Y : IN, DOUBLE (NSMP,L)
system output data sequence matrix Y = [y_1 y_2 ... y_l]. Column j of Y contains the NSMP values of the j-th output component for consecutive time increments.

L : IN, INTEGER
dimension of system output vector.

NOBL : IN, INTEGER
number of block rows in the processed input and output Hankel matrices. NOBL should be chosen larger than n, the estimated dimension of state vector.

TOL : IN, DOUBLE
absolute tolerance used for determining an estimate of the system order. If TOL > 0, the estimate is indicated by the number of singular values greater than or equal to TOL. (Singular values less than TOL are considered zero.) If TOL = 0, the internally computed default value TOL = NOBL*EPS*SV[1] is used, where EPS is the relative machine precision, and SV[1] is the maximal singular value. If TOL < 0, the estimate is indicated by the index of the singular value that has the largest logarithmic gap to its successor. (See Method for the definition of the singular values.)

WITHD : IN, LOGICAL
specifies whether or not a non-zero feedthrough matrix D to be included in the estimated state space representation:
WITHD = .TRUE. means a possibly non-zero D matrix should be estimated.
WITHD = .FALSE. means a zero D matrix is assumed.

WITHCV: IN, LOGICAL
specifies whether or not the covariance matrices are to be computed:
WITHCV = .TRUE. means the covariance matrices are computed.
WITHCV = .FALSE. means the covariance matrices should not be computed.
The covariance matrices are automatically computed if WITHK = .TRUE..

WITHK : IN, LOGICAL
specifies whether or not the Kalman gain matrix K is to be computed:
WITHK = .TRUE. means the Kalman gain matrix is computed.

```
          WITHK =  .FALSE. means the Kalman gain matrix is not
                           computed.
CONCT :   IN, LOGICAL
          specifies whether or not the successive data batches belong
          to a single experiment:
          CONCT = .TRUE.  means the current data batch is a
                          continuation of the previous data batch.
          CONCT = .FALSE. means there is no connection between the
                          current data batch and the previous ones.
LAST  :   IN, LOGICAL
          specifies whether or not sequential data processing is
          used, and (for sequential processing) whether or not the
          current data block is the last one, as follows:
          LAST =  .TRUE.  means non-sequential processing or the last
                          data block in sequential processing.
          LAST = .FALSE.  means not the last data block in sequential
                          processing. In this case the only calculation
                          performed is to update the triangular factor
                          of the QR factorization.
CTRL  :   IN, LOGICAL
          specifies whether or not the user's confirmation of the
          system order estimate is desired before the computation
          of system matrices, as follows:
          CTRL =  .TRUE. means user's confirmation is desired.
          CTRL = .FALSE. means no confirmation is necessary.
          If  CTRL =  .TRUE.,  a reverse communication routine,
          IMCORD,  is called, and, after inspecting the singular
          values and system order estimate,  n,  the user may
          accept  n  or set a new value.
          IMCORD  is not called by the routine if  CTRL = .FALSE..
          CTRL = .TRUE.  should be used in extreme cases only.
N     :   OUT, INTEGER
          estimated order of the system.
A     :   OUT, DOUBLE (NOBL-1,NOBL-1)
          the N*N estimated system state matrix A.
          (column dense)
B     :   OUT, DOUBLE (NOBL-1,M)
          the N*M estimated system input matrix B.
          (column dense)
C     :   OUT, DOUBLE (L,NOBL-1)
          the L*N estimated system output matrix C.
          (column dense)
D     :   OUT, DOUBLE (L,M)
          the L*M estimated system feedthrough matrix D if
          WITHD = .TRUE..
          D is not referenced if  WITHD = .FALSE..
          (column dense)
```

FK : OUT, DOUBLE (NOBL-1,L)
if WITHK = .TRUE., on normal return, or when the warning errors 1 or 3 are issued, the leading N*L part of this array contains the Kalman gain matrix K.
FK is not referenced when WITHK = .FALSE..
(column dense)

R : IN, OUT, DOUBLE (2*(M+L)*NOBL,2*(M+L)*NOBL)
On output, if LAST = .FALSE. then R contains the current upper triangular factor of the QR factorization used to compress the data before determining the singular values in the N4SID algorithm.
The content of R should be preserved between succesive calls of RPIMN4 with LAST = .FALSE..
On first input, the content of R is not meaningfull.
During the computations with LAST = .TRUE., R contains some relevant data for the N4SID algorithm.
On output with LAST = .TRUE., the content of R is destroyed; if WITHCV = .TRUE. or WITHK = .TRUE., on normal return, and in the absence of warning error 2, the first N+L rows and columns of R contain the estimated covariance matrix of the vector [w', e']';
if WITHK = .TRUE., on normal return, or when the warning errors 1 or 3 are issued, the rows N+L+1 to 2*N+L+1 and the columns 1 to N of R contain the estimation error covariance matrix X (the Riccati equation solution).

IWORK : OUT, INTEGER((M+L)*NOBL)
working array.

RWORK : OUT, DOUBLE (LRWORK)
working array.
On exit, if LAST = .TRUE., and normal return, then:
RWORK(1) contains the optimal suggested value of LRWORK;
RWORK(2) contains an estimate of the reciprocal condition number for the linear algebraic system solved for computing the system matrices (in the absence of warning error 2);
RWORK(3:L*NOBL+2) contain singular values used for determining the system order.
If WITHK = .TRUE., LAST = .TRUE., on normal return, or when the warning errors 1 or 3 are issued, then:
RWORK(L*NOBL+3) contains an estimate of the reciprocal condition number of the computed output covariance matrix Ry;
RWORK(L*NOBL+4) contains an estimate of the reciprocal condition number of the computed matrix A;
RWORK(L*NOBL+5) contains an estimate of the reciprocal condition number of the computed matrix U_11, used to obtain the Riccati equation solution X;
RWORK(L*NOBL+6) contains an estimate of the reciprocal condition number of the matrix Ry + C*X*C', used to

```
          obtain the Kalman gain matrix;
          RWORK(L*NOBL+7:L*NOBL+N+6)  contain the real parts of the
          optimal (Kalman predictor) poles;
          RWORK(L*NOBL+N+7:L*NOBL+2*N+6)  contain the imaginary
          parts of the optimal (Kalman predictor) poles.
          Part of this information is also available for warning
          errors 4 to 9, if logically possible.
          The first  (M+L)*(2*NOBL-1)  elements of  RWORK  should
          be preserved between succesive calls of the routine with
          LAST = .FALSE.,  and the final call with  LAST = .TRUE.,
          if successive data batches belong to a single experiment
          (CONCT = .TRUE.).
          On return with error number 4, RWORK(1) contains the
          minimum necessary value of LRWORK.
 LRWORK:  IN, INTEGER
          dimension of working array RWORK.
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN = max( 4*(NOBL+1)*(M+L)*NOBL, 5*(M+L)*NOBL,
                        2+L*NOBL +
                        max( 2*(NOBL-1) + max( (2*M+L)*NOBL+L,
                             (L*NOBL-L)*(NOBL-1) ), 4*(NOBL-1)+L,
                             4*(M*NOBL+NOBL-1),
                             4*(L*NOBL+M), 5*(L*NOBL+M)-4,
                             3 + max( 10*(NOBL-1), 3*L ),
                             5 + 2*(NOBL-1) + max( NOBL-1,3*L ) ) )
          For  NSMP = 0,  the first term of the first max expression
          could be omitted. For  WITHK = .FALSE.,  the last two terms
          of the second max expression could be omitted.
          This estimate is computed using the largest value for  n,
          namely  NOBL-1.
          For good performance,  LRWORK  should be larger.
 *      :  RETURN 1, target label in case of error (e.g. *1111)
```

File input/ output:

none

Method:

Let us denote m = M, l = L, t = NSPM, s = NOBL.

1) For non-sequential data processing, the t x 2(m+l)s matrix

$$\begin{bmatrix} U'_{1,2s,t} & Y'_{1,2s,t} \end{bmatrix}$$

is constructed, where $U_{1,2s,t}$ and $Y_{1,2s,t}$ are Hankel matrices defined in terms of the input and output data [1].

A QR factorization is used to compress the data, an oblique projection is computed in terms of some submatrices of the upper triangular factor R [2], and then a singular value decomposition (SVD) of the submatrix

R_22 := R(ms+1:(2m+1)s,ms+1:(2m+1)s)

of R reveals the order n of the system as the number of "non-zero" singular values. System matrices are finally computed using the first n right singular vectors of R_22, and other submatrices of R, solving a linear algebraic system in a total least squares sense [3]. The covariance matrices are computed by solving several least squares problems, constructed based on the residuals.

2) For sequential data processing, the QR decomposition is done sequentially, by updating the upper triangular factor R. When all data have been compressed, the system order and system matrices are computed as in the previous case.

The Kalman gain is obtained by solving a discrete-time algebraic matrix Riccati equation using the Schur vectors approach [4] for the dual of an optimal control problem.

Literature

/1/ Van Overschee, P., and De Moor, B.
N4SID: Two Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems.
Automatica, Vol.30, pp. 75-93, 1994.

/2/ Van Overschee, P.
Subspace Identification : Theory - Implementation - Applications.
Ph. D. Thesis, Katholieke Universiteit Leuven, 1995.

/3/ Van Huffel, S., and Vandewalle J.
The Total Least Squares Problem: Computational Aspects and Analysis.
SIAM, Philadelphia, 1991.

/4/ Laub A. J.
A Schur method for solving algebraic Riccati equations.
IEEE Trans. Automat. Control, AC-24, pp. 913-921, 1979.

Remarks:

- The NSMP argument may vary from a cycle to another in sequential data processing, but NOBL, M, and L should be kept constant. For efficiency, it is advisable to use NSMP as large as possible.
- When 100 cycles of sequential data processing are completed with LAST = .FALSE., a warning is issued, to prevent for an infinite loop.

Copyright:

Life cycle:
1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:
RASP, BLAS (1,2,3), LAPACK

Example:
Consider the discrete-time system

$$x(k+1) = Ax(k) + Bu(k) + Ew(k)$$
$$y(k) = Cx(k) + Du(k) + Fv(k)$$

with the following matrices:

$$A = \begin{pmatrix} 1.5 & -0.7 \\ 1.0 & 0.0 \end{pmatrix} \quad B = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad C = \begin{pmatrix} 1 & 0.5 \end{pmatrix} \quad D = 0$$

$$E = \begin{pmatrix} 0.0005 & 0.0 \\ 0.0 & 0.006 \end{pmatrix} \quad F = \begin{pmatrix} 0.0528 \end{pmatrix}$$

whose output response y(t) to normal (0,1) random input signals u(t), w(t), v(t) and zero initial state is determined by using the subroutine RPIMIU.
The following sequence of statements can be used to estimate the order and the matrices of the original system

```
      DATA   ISEED/ 1, 2, 3, 5 /
      NSMP   = 120
      NOBL   = 6
      M      = 1
      MW     = 2
      L      = 1
      LV     = 1
      TOL    = -1.0D0
      WITHD  = .FALSE.
      WITHK  = .TRUE.
      LAST   = .TRUE.
      CONCT  = .TRUE.
      CTRL   = .TRUE.
      LRWORK = 8100
C     Compute the output response for uniformly distributed
C     input sequence.
      X(1)   = 0.0D0
      X(2)   = 0.0D0
      CALL RPIMIU( NSMP, 2, M, L, MW, LV, A, B, E, C, D, F, X, U,
     *             W, V, ISEED, Y, RWORK, LRWORK, WITHD, 'G',
     *             'G', 'R', 'R', 'R', 3, *1111 )
```

```
C     Perform the system identification.
      CALL RPIMN4( U, NSMP, M, Y, L, NOBL, TOL, WITHD, WITHCV,
     *             WITHK, CONCT, LAST, CTRL, N, A, B, C, D, FK,
     *             R, IWORK, RWORK, LRWORK, *1111 )

 The order detection singular values are:

.87524260D+02  .42090540D+02  .32603442D+00  .22092251D+00  .19614963D+00
.16624421D+00

 We can take the system order n = 2.
 The estimated system matrices are:
                                 _                    _
 A = (  .8245 -.5418 )      B = ( 3.8116 )       C = ( .5093   .5822 )
     (  .2643  .6748 )          (-1.6200 )

 The estimated Kalman gain matrix K is:

 K = ( .1190 )
     ( .1635 )

 The estimated (inter-)covariance matrix (for [Ew; Fv]) is:

        ( .0001    .0002    .0003 )
        ( .0002    .0006    .0000 )
        ( .0003    .0000    .0029 )

 The estimated state covariance matrix X is:

        ( .0012   -.0001 )
        (-.0001    .0011 )
                                                 _
 The relative output error is: ||y(t)-y(t)|| / ||y(t)|| =  .41315D-02
```

Error Messages:

```
-1-
Invalid parameter value on entry.
-2-
Singular value decomposition failed.
-3-
A singular upper triangular matrix was found.
-4-
Not enough working storage. It should be at least //LENG//.
```

Warnings Messages:

```
-1-
More than 100 cycles in sequential data processing
-2-
All singular values are exactly zero (N = 0)
-3-
The least squares problem for covariances is rank-deficient
-4-
The computed output covariance matrix Ry is too small'
-5-
The covariance matrix Ry is numerically singular
-6-
The transformed matrix A is numerically singular
-7-
Schur form computation did not converged
-8-
Schur form ordering failed
-9-
The computed matrix U_11 is numerically singular
```

3. The MOESP Class of Subspace Model Identification

The main feature of this class of SMI techniques is the determination of the extended observability matrix of the deterministic part of the MOESP model. When this model is given in the following way:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + Du_k + v_k \end{aligned}$$

the extended observability matrix Γ_s is given as:

$$\Gamma_s = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{s-1} \end{bmatrix}$$

The basic idea was first described in /1,2/. The algorithm developed from this idea, which is indicated by the ordinary MOESP (OM) scheme, allows to identify in a statistically consistent and efficient way LTI systems that can be described by the MOESP model with $w_k = 0$ and v_k being zero-mean white noise independent from the input.

Later on, an extension was formulated based on the incorporation of instrumental variables based on past input quantities and/or reconstructed state variables that allow to consistently identify MOESP class of models for v_k being a zero-mean arbitrary stochastic disturbance independent from the input, and $w_k = 0$.
The price payed for this increased applicability is that the estimates are no longer efficient. This variant is indicated as the PI scheme (when past inputs are used as instrumental variables) and the RS scheme (when reconstructed state variables are used).

When the additive disturbance w_k in the MOESP model is generated by an innovation model of the form $w_k = Kv_k$ and v_k is a zero-mean white noise independent of the input, it is again possible to obtain both consistent and efficient estimates when, in addition to the past input, past output quantities are used as instrumental variables.
The scheme derived from this extension of ordinary MOESP scheme is indicated as the PO scheme.

For the deterministic-stochastic identification, it is possible to estimate the deterministic part by using the MOESP approach with the PO scheme, and the stochastic part, that is the noise covariance matrices, by using the Space-Intersection (SI) approach /4/. Further, the corresponding Kalman gain K can be computed to allow the design of state observers /3/.

The algorithms belonging to the MOESP class have the striking feature of being highly streamlined, in the sense that the sequence of

computations performed by these schemes is almost independent from the type of problems to be analyzed. This allows to modularize the implementations of the algorithms within this class to a great extend.

The following routines are available for MOESP class of SMI:

RPIMOE computes, by using the ordinary MOESP scheme, consistent and statistically efficient estimates of the matrices (A,B,C,D) of a state space model when $w = 0$ and v is zero-mean white noise independent from the input.

RPIMPI computes, by using the ordinary MOESP scheme extended with instrumental variables based on past input quantities, consistent estimates of the matrices (A,B,C,D) of a state space model when $w = 0$ and v is zero-mean but of arbitrary statistical color.

RPIMPO computes, by using the ordinary MOESP scheme extended with instrumental variables based on past input and past output quantities, consistent and statistically efficient estimates of the matrices (A,B,C,D) of a state space model when w is generated by an innovation model and v is zero-mean white noise independent from the input.

RPIMRS computes, by using the ordinary MOESP scheme extended with instrumental variables based on reconstructed state and/or past input quantities, consistent estimates of the matrices (A,B,C,D) of a state space model when $w = 0$ and v is zero-mean but of arbitrary statistical color.

RPIMKG in addition to the RPIMPO routine, also estimates by using the SI approach the noise covariance matrices and the Kalman gain to allow the design of a state observer as a state predictor.

Literature:

/1/ Verhaegen M. and Dewilde P.
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.
Int. J. Control, 56, pp. 1187-1210, 1992.

/2/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.

/3/ Verhaegen M.
Identification of the deterministic part of MIMO state space models given in innovations form from input-output data.
Automatica, Vol.30, No.1, pp.61-74, 1994.

/4/ Van Overschee P. and De Moor B.
N4SID: Two subspace algorithms for the identification of combined deterministic-stochastic systems.
Automatica, Vol.30, No.1, pp.75-93, 1994.

SUBROUTINE RPIMOE

Ordinary MOESP identification

Procedure purpose:

This subroutine estimates from given input and output data sequences, the order and the matrices of the linear time-invariant discrete-time state space model

$$x(k+1) = Ax(k) + Bu(k)$$

$$y(k) = Cx(k) + Du(k) + v(k)$$

by using the ordinary MOESP algorithm.

Usage:

```
CALL RPIMOE(U, NSMP, M, Y, L, NOBL, TOL, WITHD, CONCT, LAST,
            CTRL, N, A, B, C, D, R, IWORK, RWORK, LRWORK, *)
```

U : IN, DOUBLE (NSMP,M)
system input data sequence matrix U = [u_1 u_2 ... u_m].
Column j of U contains the NSMP values of the
j-th input component for consecutive time increments.

NSMP : IN, INTEGER
number of rows of matrices U and Y (number of samples)
(When sequential data processing is used, NSMP is the
number of samples of the current data batch.)
NSMP >= (M+L+1)*NOBL - 1, for non-sequential processing
of the data;
NSMP >= NOBL, for sequential processing of several data
batches;
NSMP = 0, and LAST = .TRUE., is allowed if the upper
triangular factor of the QR factorization is
already available in the array R, and only the
system matrices are to be computed.
The total number of samples when calling with LAST = .TRUE.
should be at least (M+L+1)*NOBL - 1.

M : IN, INTEGER
dimension of system input vector.

Y : IN, DOUBLE (NSMP,L)
system output data sequence matrix Y = [y_1 y_2 ... y_l].
Column j of Y contains the NSMP values of the
j-th output component for consecutive time increments.

L : IN, INTEGER
dimension of system output vector.

NOBL : IN, INTEGER
number of block rows in the processed input and output Hankel matrices. NOBL should be chosen larger than n, the estimated dimension of state vector.

TOL : IN, DOUBLE
absolute tolerance used for determining an estimate of the system order. If TOL > 0, the estimate is indicated by the number of singular values greater than or equal to TOL. (Singular values less than TOL are considered zero.) If TOL = 0, the internally computed default value TOL = NOBL*EPS*SV[1] is used, where EPS is the relative machine precision, and SV[1] is the maximal singular value. If TOL < 0, the estimate is indicated by the index of the singular value that has the largest logarithmic gap to its successor. (See Method for the definition of the singular values.)

WITHD : IN, LOGICAL
specifies whether or not a non-zero feedthrough matrix D to be included in the estimated state space representation:
WITHD = .TRUE. means a possibly non-zero D matrix should be estimated.
WITHD = .FALSE. means a zero D matrix is assumed.

CONCT : IN, LOGICAL
specifies whether or not the successive data batches belong to a single experiment:
CONCT = .TRUE. means the current data batch is a continuation of the previous data batch.
CONCT = .FALSE. means there is no connection between the current data batch and the previous ones.

LAST : IN, LOGICAL
specifies whether or not sequential data processing is used, and (for sequential processing) whether or not the current data block is the last one, as follows:
LAST = .TRUE. means non-sequential processing or the last data block in sequential processing.
LAST = .FALSE. means not the last data block in sequential processing. In this case the only calculation performed is to update the triangular factor of the QR factorization.

CTRL : IN, LOGICAL
specifies whether or not the user's confirmation of the system order estimate is desired before the computation of system matrices, as follows:
CTRL = .TRUE. means user's confirmation is desired.
CTRL = .FALSE. means no confirmation is necessary.
If CTRL = .TRUE., a reverse communication routine, IMCORD, is called, and, after inspecting the singular

values and system order estimate, n, the user may
accept n or set a new value.
IMCORD is not called by the routine if CTRL = .FALSE..
CTRL = .TRUE. should be used in extreme cases only.

N : OUT, INTEGER
estimated order of the system.

A : OUT, DOUBLE (NOBL-1,NOBL-1)
the N*N estimated system state matrix A.
(column dense)

B : OUT, DOUBLE (NOBL-1,M)
the N*M estimated system input matrix B.
(column dense)

C : OUT, DOUBLE (L,NOBL-1)
the L*N estimated system output matrix C.
(column dense)

D : OUT, DOUBLE (L,M)
the L*M estimated system feedthrough matrix D if
WITHD = .TRUE..
D is not referenced if WITHD = .FALSE..
(column dense)

R : IN, OUT, DOUBLE ((max(M,L)+L)*NOBL,(M+L)*NOBL)
On output, if LAST = .FALSE., then the leading
(M+L)*NOBL x (M+L)*NOBL part of R contains the current
upper triangular factor of the QR factorization used to
compress the data before determining the singular values
in the MOESP algorithm.
The content of R should be preserved between succesive
calls of RPIMOE with LAST = .FALSE..
On first input, the content of R is not meaningfull.

IWORK : OUT, INTEGER(M)
working array.

RWORK : OUT, DOUBLE (LRWORK)
working array.
RWORK(1) contains the optimal suggested value of LRWORK;
RWORK(2) contains an estimate of the reciprocal condition
number for the final linear algebraic system solved;
RWORK(3:2+L*NOBL) contain the singular values used for
determining the system order.
The first (M+L)*(NOBL-1) elements of RWORK should
be preserved between succesive calls of the routine with
LAST = .FALSE., and the final call with LAST = .TRUE.,
if successive data batches belong to a single experiment
(CONCT = .TRUE.).
On return with error number 4, RWORK(1) contains the
minimum necessary value of LRWORK.

LRWORK: IN, INTEGER
dimension of working array RWORK.

```
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN = max( (NOBL+2)*(M+L)*NOBL,
                        2+L*NOBL + max( ((2*NOBL-3)*L+2)*(NOBL-1),
                                        (L*NOBL-1)*M*NOBL,
                                        4*(L*NOBL+M),
                                        5*(L*NOBL+M)-4 ) )
          For  NSMP = 0,  the first term of the first max expression
          could be omitted.
          This estimate is computed using the largest and smallest
          values for  n, namely  NOBL-1  and 1, respectively
          (for positive and negative terms, respectively).
          In practice,  n  could be much smaller than  NOBL-1.
          For good performance,  LRWORK  should be larger.
*     :   RETURN 1, target label in case of error (e.g. *1111)
```

File input/ output:

none

Method:

Let us denote m = M, l = L, t = NSPM, s = NOBL.

For non-sequential data processing, the t x (m+l)s matrix

$$\begin{bmatrix} U'_{1,s,t} & Y'_{1,s,t} \end{bmatrix}$$

is constructed, where $U_{1,s,t}$ and $Y_{1,s,t}$ are Hankel matrices defined in terms of the input and output data /1/.
A QR factorization is used to compress the data, and then a singular value decomposition (SVD) of the principal submatrix R_22 := R(ms+1:(m+l)s,ms+1:(m+l)s) of the upper triangular factor R reveals the order n of the system as the number of "non-zero" singular values. System matrices are finally computed from the right singular vectors of R_22 and the submatrices R_11 and R_12 of R.

For sequential data processing, the QR decomposition is done sequentially, by updating the upper triangular factor R. When all data have been compressed, the system order and system matrices are computed as in the previous case.

Literature

/1/ Verhaegen M., and P. Dewilde
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.
Int. J. Control, 56, pp. 1187-1210, 1992.

/2/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.

Remarks:

- The NSMP argument may vary from a cycle to another in sequential data processing, but NOBL, M, and L should be kept constant. For efficiency, it is advisable to use NSMP as large as possible.
- When 100 cycles of sequential data processing are completed with LAST = .FALSE., a warning is issued, to prevent for an infinite loop.

Copyright:

Life cycle:

1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

Consider the discrete-time system (A,B,C,D) with the following matrices:

```
A = ( 1.5   -0.7 )     B = ( 1 )     C = ( 1   0.5 )     D =  0
    ( 1.0    0.0 )         ( 0 )
```

whose output response y(t) to random input signals u(t) and zero initial state is determined by using the subroutine RPIMIU. The following sequence of statements can be used to estimate the order and the matrices of the original system

```
      DATA   ISEED/ 1, 2, 3, 5 /
      NSMP   = 120
      NOBL   = 10
      M      = 1
      L      = 1
      TOL    = -1.0D0
      WITHD  = .FALSE.
      LAST   = .TRUE.
      CONCT  = .TRUE.
      CTRL   = .TRUE.
      LRWORK = 8100
C     Compute the output response for uniformly distributed
C     input sequence.
```

```
      X(1)   = 0.0D0
      X(2)   = 0.0D0
      CALL RPIMIU( NSMP, 2, M, L, 0, 0, A, B, E, C, D, F, X, U,
     *             W, V, ISEED, Y, RWORK, LRWORK, WITHD, 'G',
     *             'G', 'R', 'R', 'R', 1, *1111 )
C     Perform the system identification.
      CALL RPIMOE( U, NSMP, M, Y, L, NOBL, TOL, WITHD, CONCT, LAST,
     *             CTRL, N, A, B, C, D, R, IWORK, RWORK, LRWORK,
     *             *1111 )
```

The order detection singular values are:

3.2203D+01 1.4612D+01 0 0 0 0 0 0 0 0

The estimated system matrices are:

$$\bar{A} = \begin{pmatrix} .7831 & .5606 \\ -.2472 & .7169 \end{pmatrix} \quad \bar{B} = \begin{pmatrix} -3.5468 \\ -2.3707 \end{pmatrix} \quad \bar{C} = \begin{pmatrix} -.5749 & .4382 \end{pmatrix}$$

which coincide with the matrices obtained by applying a similarity transformation to the original system.

The relative output error is: $\|y(t)-\bar{y}(t)\| / \|y(t)\| = 0.11277D-14$

Error Messages:

-1-
Invalid parameter value on entry.
-2-
Singular value decomposition failed.
-3-
A singular upper triangular matrix was found.
-4-
Not enough working storage. It should be at least //LENG//.

Warnings Messages:

-1-
More than 100 cycles in sequential data processing
-2-
All singular values are exactly zero (N = 0)

SUBROUTINE RPIMPI

Ordinary MOESP identification with past inputs

Procedure purpose:

This subroutine estimates from given input and output data sequences, the order and the matrices of the linear time-invariant discrete-time state space model

$$x(k+1) = Ax(k) + Bu(k)$$

$$y(k) = Cx(k) + Du(k) + v(k)$$

by using the ordinary MOESP algorithm with past inputs.

Usage:

```
CALL RPIMPI(U, NSMP, M, Y, L, NOBL, TOL, WITHD, CONCT, LAST,
            CTRL, N, A, B, C, D, R, IWORK, RWORK, LRWORK, *)
```

U : IN, DOUBLE (NSMP,M)
system input data sequence matrix U = [u_1 u_2 ... u_m].
Column j of U contains the NSMP values of the
j-th input component for consecutive time increments.

NSMP : IN, INTEGER
number of rows of matrices U and Y (number of samples).
(When sequential data processing is used, NSMP is the
number of samples of the current data batch.)
NSMP >= (2*M+L+2)*NOBL - 1, for non-sequential processing
of the data;
NSMP >= 2*NOBL, for sequential processing of several data
batches;
NSMP = 0, and LAST = .TRUE., is allowed if the upper
triangular factor of the QR factorization is
already available in the array R, and only the
system matrices are to be computed.
The total number of samples when calling with LAST = .TRUE.
should be at least (2*M+L+2)*NOBL - 1.

M : IN, INTEGER
dimension of system input vector.

Y : IN, DOUBLE (NSMP,L)
system output data sequence matrix Y = [y_1 y_2 ... y_l].
Column j of Y contains the NSMP values of the
j-th output component for consecutive time increments.

L : IN, INTEGER
dimension of system output vector.

NOBL : IN, INTEGER
number of block rows in the processed input and output Hankel matrices. NOBL should be chosen larger than n, the estimated dimension of state vector.

TOL : IN, DOUBLE
absolute tolerance used for determining an estimate of the system order. If TOL > 0, the estimate is indicated by the number of singular values greater than or equal to TOL. (Singular values less than TOL are considered zero.) If TOL = 0, the internally computed default value TOL = NOBL*EPS*SV[1] is used, where EPS is the relative machine precision, and SV[1] is the maximal singular value. If TOL < 0, the estimate is indicated by the index of the singular value that has the largest logarithmic gap to its successor. (See Method for the definition of the singular values.)

WITHD : IN, LOGICAL
specifies whether or not a non-zero feedthrough matrix D to be included in the estimated state space representation:
WITHD = .TRUE. means a possibly non-zero D matrix should be estimated.
WITHD = .FALSE. means a zero D matrix is assumed.

CONCT : IN, LOGICAL
specifies whether or not the successive data batches belong to a single experiment:
CONCT = .TRUE. means the current data batch is a continuation of the previous data batch.
CONCT = .FALSE. means there is no connection between the current data batch and the previous ones.

LAST : IN, LOGICAL
specifies whether or not sequential data processing is used, and (for sequential processing) whether or not the current data block is the last one, as follows:
LAST = .TRUE. means non-sequential processing or the last data block in sequential processing.
LAST = .FALSE. means not the last data block in sequential processing. In this case the only calculation performed is to update the triangular factor of the QR factorization.

CTRL : IN, LOGICAL
specifies whether or not the user's confirmation of the system order estimate is desired before the computation of system matrices, as follows:
CTRL = .TRUE. means user's confirmation is desired.
CTRL = .FALSE. means no confirmation is necessary.
If CTRL = .TRUE., a reverse communication routine, IMCORD, is called, and, after inspecting the singular

values and system order estimate, n, the user may
accept n or set a new value.
IMCORD is not called by the routine if CTRL = .FALSE..
CTRL = .TRUE. should be used in extreme cases only.

N : OUT, INTEGER
estimated order of the system.

A : OUT, DOUBLE (NOBL-1,NOBL-1)
the N*N estimated system state matrix A.
(column dense)

B : OUT, DOUBLE (NOBL-1,M)
the N*M estimated system input matrix B.
(column dense)

C : OUT, DOUBLE (L,NOBL-1)
the L*N estimated system output matrix C.
(column dense)

D : OUT, DOUBLE (L,M)
the L*M estimated system feedthrough matrix D if
WITHD = .TRUE..
D is not referenced if WITHD = .FALSE..
(column dense)

R : IN, OUT, DOUBLE ((max(2*M,L)+L)*NOBL,(2*M+L)*NOBL)
On output, if LAST = .FALSE., then the leading
(2*M+L)*NOBL x (2*M+L)*NOBL part of R contains the current
upper triangular factor of the QR factorization used to
compress the data before determining the singular values in
the MOESP algorithm.
The content of R should be preserved between succesive
calls of RPIMPI with LAST = .FALSE..
On first input, the content of R is not meaningfull.

IWORK : OUT, INTEGER(M)
working array.

RWORK : OUT, DOUBLE (LRWORK)
working array.
RWORK(1) contains the optimal suggested value of LRWORK;
RWORK(2) contains an estimate of the reciprocal condition
number for the final linear algebraic system solved;
RWORK(3:2+min(M,L)*NOBL) contain the singular values used
for determining the system order.
The first M*(2*NOBL-1) + L*(NOBL-1) elements of RWORK
should be preserved between succesive calls of the routine
with LAST = .FALSE., and the final call with LAST = .TRUE.,
if successive data batches belong to a single experiment
(CONCT = .TRUE.).
On return with error number 4, RWORK(1) contains the
minimum necessary value of LRWORK.

LRWORK: IN, INTEGER
dimension of working array RWORK.

```
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN = max( 2*(NOBL+1)*(2*M+L)*NOBL, 2+min(M,L)*NOBL +
                        max( ((2*NOBL-3)*L+2)*(NOBL-1),
                             (L*NOBL-1)*M*NOBL, 4*(L*NOBL+M),
                             5*(L*NOBL+M)-4 ) )
          For  NSMP = 0,  the first term of the first max expression
          could be omitted.
          This estimate is computed using the largest and smallest
          values for  n, namely  NOBL-1  and 1, respectively
          (for positive and negative terms, respectively).
          In practice,  n  could be much smaller than  NOBL-1.
          For good performance,  LRWORK  should be larger.
*     :    RETURN 1, target label in case of error (e.g. *1111)
```

File input/ output:

none

Method:

Let us denote m = M, l = L, t = NSPM, s = NOBL.

For non-sequential data processing, the t x (2m+l)s matrix

$$\begin{bmatrix} Uf'_{s+1,s,t} & Up'_{1,s,t} & Yf'_{s+1,s,t} \end{bmatrix}$$

is constructed, where $Up_{1,s,t}$, $Uf_{s+1,s,t}$ and $Yf_{s+1,s,t}$ are Hankel matrices defined in terms of the input and output data /2/. A QR factorization is used to compress the data, and then a singular value decomposition (SVD) of the submatrix $R\_23 := R(ms+1:2ms, 2ms+1:(2m+1)s)$ of the upper triangular factor R reveals the order n of the system as the number of "non-zero" singular values. System matrices are finally computed from the right singular vectors of R_23 and the submatrices R_11 and R_13 of R.

For sequential data processing, the QR decomposition is done sequentially, by updating the upper triangular factor R. When all data have been compressed, the system order and system matrices are computed as in the previous case.

Literature

/1/ Verhaegen M., and P. Dewilde
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.
Int. J. Control, 56, pp. 1187-1210, 1992.

/2/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.

Remarks:

- The NSMP argument may vary from a cycle to another in sequential data processing, but NOBL, M, and L should be kept constant. For efficiency, it is advisable to use NSMP as large as possible.
- When 100 cycles of sequential data processing are completed with LAST = .FALSE., a warning is issued, to prevent for an infinite loop.

Copyright:

Life cycle:

1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

Consider the discrete-time system (A,B,C,D) with the following matrices:

```
A = ( 1.5   -0.7 )     B = ( 1 )     C = ( 1   0.5 )     D =  0
    ( 1.0    0.0 )         ( 0 )
```

whose output response y(t) to random input signals u(t) and zero initial state is determined by using the subroutine RPIMIU. The following sequence of statements can be used to estimate the order and the matrices of the original system

```
      DATA   ISEED/ 1, 2, 3, 5 /
      NSMP   = 120
      NOBL   = 8
      M      = 1
      L      = 1
      TOL    = -1.0D0
      WITHD  = .FALSE.
      LAST   = .TRUE.
      CONCT  = .TRUE.
      CTRL   = .TRUE.
      LRWORK = 8100
C     Compute the output response for uniformly distributed
C     input sequence.
```

```
      X(1)   = 0.0D0
      X(2)   = 0.0D0
      CALL RPIMIU( NSMP, 2, M, L, 0, 0, A, B, E, C, D, F, X, U,
     *             W, V, ISEED, Y, RWORK, LRWORK, WITHD, 'G',
     *             'G', 'R', 'R', 'R', 1, *1111 )
C     Perform the system identification.
      CALL RPIMPI( U, NSMP, M, Y, L, NOBL, TOL, WITHD, CONCT, LAST,
     *             CTRL, N, A, B, C, D, R, IWORK, RWORK, LRWORK,
     *             *1111 )
```

The order detection singular values are:

2.7731D+01 1.2852D+01 0 0 0 0 0 0

The estimated system matrices are:

$$\bar{A} = \begin{pmatrix} .7923 & -.5521 \\ .2523 & .7077 \end{pmatrix} \quad \bar{B} = \begin{pmatrix} 3.4759 \\ -2.2976 \end{pmatrix} \quad \bar{C} = \begin{pmatrix} .5794 & .4413 \end{pmatrix}$$

which coincide with the matrices obtained by applying a similarity transformation to the original system.

The relative output error is: $\|y(t)-\bar{y}(t)\| \,/\, \|y(t)\| = 0.80622D\text{-}15$

Error Messages:

-1-
Invalid parameter value on entry.
-2-
Singular value decomposition failed.
-3-
A singular upper triangular matrix was found.
-4-
Not enough working storage. It should be at least //LENG//.

Warnings Messages:

-1-
More than 100 cycles in sequential data processing
-2-
All singular values are exactly zero (N = 0)

SUBROUTINE RPIMPO

Ordinary MOESP identification with past inputs and outputs

Procedure purpose:

This subroutine estimates from given input and output data sequences, the order and the matrices of the linear time-invariant discrete-time state space model

$$x(k+1) = Ax(k) + Bu(k) + w(k)$$

$$y(k) = Cx(k) + Du(k) + v(k)$$

by using the ordinary MOESP algorithm with past inputs and outputs.

Usage:

```
CALL RPIMPO(U, NSMP, M, Y, L, NOBL, TOL, WITHD, CONCT, LAST,
            CTRL, N, A, B, C, D, R, IWORK, RWORK, LRWORK, *)
```

U : IN, DOUBLE (NSMP,M)
system input data sequence matrix U = [u_1 u_2 ... u_m]. Column j of U contains the NSMP values of the j-th input component for consecutive time increments.

NSMP : IN, INTEGER
number of rows of matrices U and Y (number of samples). (When sequential data processing is used, NSMP is the number of samples of the current data batch.)
NSMP >= 2*(M+L+1)*NOBL - 1, for non-sequential processing of the data;
NSMP >= 2*NOBL, for sequential processing of several data batches;
NSMP = 0, and LAST = .TRUE., is allowed if the upper triangular factor of the QR factorization is already available in the array R, and only the system matrices are to be computed.
The total number of samples when calling with LAST = .TRUE. should be at least 2*(M+L+1)*NOBL - 1.

M : IN, INTEGER
dimension of system input vector.

Y : IN, DOUBLE (NSMP,L)
system output data sequence matrix Y = [y_1 y_2 ... y_l]. Column j of Y contains the NSMP values of the j-th output component for consecutive time increments.

L : IN, INTEGER
dimension of system output vector.

NOBL : IN, INTEGER
number of block rows in the processed input and output Hankel matrices. NOBL should be chosen larger than n, the estimated dimension of state vector.

TOL : IN, DOUBLE
absolute tolerance used for determining an estimate of the system order. If TOL > 0, the estimate is indicated by the number of singular values greater than or equal to TOL. (Singular values less than TOL are considered zero.)
If TOL = 0, the internally computed default value TOL = NOBL*EPS*SV[1] is used, where EPS is the relative machine precision, and SV[1] is the maximal singular value.
If TOL < 0, the estimate is indicated by the index of the singular value that has the largest logarithmic gap to its successor. (See Method for the definition of the singular values.)

WITHD : IN, LOGICAL
specifies whether or not a non-zero feedthrough matrix D to be included in the estimated state space representation:
WITHD = .TRUE. means a possibly non-zero D matrix should be estimated.
WITHD = .FALSE. means a zero D matrix is assumed.

CONCT : IN, LOGICAL
specifies whether or not the successive data batches belong to a single experiment:
CONCT = .TRUE. means the current data batch is a continuation of the previous data batch.
CONCT = .FALSE. means there is no connection between the current data batch and the previous ones.

LAST : IN, LOGICAL
specifies whether or not sequential data processing is used, and (for sequential processing) whether or not the current data block is the last one, as follows:
LAST = .TRUE. means non-sequential processing or the last data block in sequential processing.
LAST = .FALSE. means not the last data block in sequential processing. In this case the only calculation performed is to update the triangular factor of the QR factorization.

CTRL : IN, LOGICAL
specifies whether or not the user's confirmation of the system order estimate is desired before the computation of system matrices, as follows:
CTRL = .TRUE. means user's confirmation is desired.
CTRL = .FALSE. means no confirmation is necessary.
If CTRL = .TRUE., a reverse communication routine, IMCORD, is called, and, after inspecting the singular

values and system order estimate, n, the user may
accept n or set a new value.
IMCORD is not called by the routine if CTRL = .FALSE..
CTRL = .TRUE. should be used in extreme cases only.

N : OUT, INTEGER
estimated order of the system.

A : OUT, DOUBLE (NOBL-1,NOBL-1)
the N*N estimated system state matrix A.
(column dense)

B : OUT, DOUBLE (NOBL-1,M)
the N*M estimated system input matrix B.
(column dense)

C : OUT, DOUBLE (L,NOBL-1)
the L*N estimated system output matrix C.
(column dense)

D : OUT, DOUBLE (L,M)
the L*M estimated system feedthrough matrix D if
WITHD = .TRUE..
D is not referenced if WITHD = .FALSE..
(column dense)

R : IN, OUT, DOUBLE (max(2*(M+L)*NOBL, 3*M*NOBL),2*(M+L)*NOBL)
On output, if LAST = .FALSE., then the leading
2*(M+L)*NOBL x 2*(M+L)*NOBL part of R contains the current
upper triangular factor of the QR factorization used to
compress the data before determining the singular values
in the MOESP algorithm.
The content of R should be preserved between succesive
calls of RPIMPO with LAST = .FALSE..
On first input, the content of R is not meaningfull.

IWORK : OUT, INTEGER(M)
working array.

RWORK : OUT, DOUBLE (LRWORK)
working array.
RWORK(1) contains the optimal suggested value of LRWORK;
RWORK(2) contains an estimate of the reciprocal condition
number for the final linear algebraic system solved;
RWORK(3:2+L*NOBL) contain the singular values used
for determining the system order.
The first (M+L)*(2*NOBL-1) elements of RWORK should be
preserved between succesive calls of the routine with
LAST = .FALSE., and the final call with LAST = .TRUE.,
if successive data batches belong to a single experiment
(CONCT = .TRUE.).
On return with error number 4, RWORK(1) contains the
minimum necessary value of LRWORK.

LRWORK: IN, INTEGER
dimension of working array RWORK.

```
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN = max( 4*(NOBL+1)*(M+L)*NOBL, 2+L*NOBL +
                        max( ((2*NOBL-3)*L+2)*(NOBL-1),
                             (L*NOBL-1)*M*NOBL, 4*(L*NOBL+M),
                             5*(L*NOBL+M)-4 ) )
          For  NSMP = 0,  the first term of the first max expression
          could be omitted.
          This estimate is computed using the largest and smallest
          values for  n, namely  NOBL-1  and 1, respectively
          (for positive and negative terms, respectively).
          In practice,  n  could be much smaller than  NOBL-1.
          For good performance,  LRWORK  should be larger.
*     :   RETURN 1, target label in case of error (e.g. *1111)
```

File input/ output:

none

Method:

Let us denote m = M, l = L, t = NSPM, s = NOBL.

For non-sequential data processing, the t x 2(m+l)s matrix

$$\begin{bmatrix} Uf'_{s+1,s,t} & Up'_{1,s,t} & Y'_{1,2s,t} \end{bmatrix}$$

is constructed, where $Up_{1,s,t}$, $Uf_{s+1,s,t}$ and $Y_{1,2s,t}$ are Hankel matrices defined in terms of the input and output data /3/. A QR factorization is used to compress the data, and then a singular value decomposition (SVD) of the submatrix $[\ R\_24'\ \ R\_34'\ ]' := R(ms+1:(2m+l)s,(2m+l)s+1:2(m+l)s)$ of the upper triangular factor R reveals the order n of the system as the number of "non-zero" singular values. System matrices are finally computed from the right singular vectors of $[\ R\_24'\ \ R\_34'\ ]'$ and the final, "compressed" submatrices R_11 and R_12 of R.

For sequential data processing, the QR decomposition is done sequentially, by updating the upper triangular factor R. When all data have been compressed, the system order and system matrices are computed as in the previous case.

Literature

/1/ Verhaegen M., and P. Dewilde
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.
Int. J. Control, 56, pp. 1187-1210, 1992.

/2/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.
/3/ Verhaegen M.
Identification of the deterministic part of MIMO state space models given in innovations form from input-output data.
Automatica, 30, pp. 61-74, 1994.

Remarks:

- The NSMP argument may vary from a cycle to another in sequential data processing, but NOBL, M, and L should be kept constant. For efficiency, it is advisable to use NSMP as large as possible.
- When 100 cycles of sequential data processing are completed with LAST = .FALSE., a warning is issued, to prevent for an infinite loop.

Copyright:

Life cycle:

1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

Consider the discrete-time system (A,B,C,D) with the following matrices:

```
A = ( 1.5   -0.7 )     B = ( 1 )     C = ( 2.5   -0.2 )     D  = ( 1 )
    ( 1.0    0.0 )         ( 0 )
```

whose output response y(t) to random input signals u(t) and zero initial state is determined by using the subroutine RPIMIU. The following sequence of statements can be used to estimate the order and the matrices of the original system

```
      DATA   ISEED/ 1, 2, 3, 5 /
      NSMP   = 80
      NOBL   = 6
      M      = 1
      L      = 1
      TOL    = -1.0D0
      WITHD  = .TRUE.
      CONCT  = .TRUE.
      LAST   = .TRUE.
```

```
      CTRL   = .TRUE.
      LRWORK = 8100
C     Compute the output response for uniformly distributed
C     input sequence.
      X(1)   = 0.0D0
      X(2)   = 0.0D0
      CALL RPIMIU( NSMP, 2, M, L, 0, 0, A, B, E, C, D, F, X, U,
     *             W, V, ISEED, Y, RWORK, LRWORK, WITHD, 'G',
     *             'G', 'R', 'R', 'R', 1, *1111 )
C     Perform the system identification.
      CALL RPIMPO( U, NSMP, M, Y, L, NOBL, TOL, WITHD, CONCT, LAST,
     *             CTRL, N, A, B, C, D, R, IWORK, RWORK, LRWORK,
     *             *1111 )
```

The order detection singular values are:

3.5290D+01 1.5322D+01 0 0 0 0

The estimated system matrices are:

$$\bar{A} = \begin{pmatrix} .7236 & .5580 \\ -.2476 & .7764 \end{pmatrix} \quad \bar{B} = \begin{pmatrix} 5.6737 \\ 3.3928 \end{pmatrix} \quad \bar{C} = \begin{pmatrix} .6706 & -.3845 \end{pmatrix}$$

$$\bar{D} = \begin{pmatrix} 1 \end{pmatrix}$$

which coincide with the matrices obtained by applying a similarity transformation to the original system.

The relative output error is: $||y(t)-\bar{y}(t)|| \; / \; ||y(t)|| = 0.89731D\text{-}15$

Error Messages:

-1-
Invalid parameter value on entry.
-2-
Singular value decomposition failed.
-3-
A singular upper triangular matrix was found.
-4-
Not enough working storage. It should be at least //LENG//.

Warnings Messages:

-1-
More than 100 cycles in sequential data processing
-2-
All singular values are exactly zero (N = 0)

SUBROUTINE RPIMRS

Ordinary MOESP Identification with Reconstructed States and Past Inputs

Procedure purpose:

This subroutine determines the order of a linear time-invariant discrete-time state space model and then estimates the quadruple of system matrices using the MOESP algorithm with reconstructed state variables as instrumental variables.

The model structure is :

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k) + Du(k) + e(k)$$

where e(k) is a zero-mean noise of arbitary color, independent of the noise-free input u(k). Matrix D could be zero.

Usage:

```
CALL RPIMRS( U, NSMP, M, Y, L, XEST, N, NOBL, TOL, WITHD, CONCT,
             LAST, CTRL, A, B, C, D, R, IWORK, RWORK, LRWORK, * )
```

U : IN, DOUBLE (NSMP,M)
system input data sequence matrix U = [u_1 u_2 ... u_m].
Column j of U contains the NSMP values of the
j-th input component for consecutive time increments.

NSMP : IN, INTEGER
number of rows of matrices U, Y and XEST (number of samples)
(When sequential data processing is used, NSMP is the
number of samples of the current data batch.)
NSMP >= (M+L+1)*NOBL + N - 1, for non-sequential processing
of the data;
NSMP >= NOBL, for sequential processing of several data batches;
NSMP = 0, and LAST = .TRUE., is allowed if the upper
triangular factor of the QR factorization is
already available in the array R, and only the
system matrices are to be computed.
The total number of samples when calling with LAST = .TRUE.
should be at least (M+L+1)*NOBL + N - 1.

M : IN, INTEGER
dimension of system input vector.

Y : IN, DOUBLE (NSMP,L)
system output data sequence matrix Y = [y_1 y_2 ... y_l].
Column j of Y contains the NSMP values of the
j-th output component for consecutive time increments.

L : IN, INTEGER
dimension of system output vector.

XEST : IN, DOUBLE (NSMP,N)
estimated state sequence matrix XEST = [x_1 x_2 ... x_n]. Column j of XEST contains the NSMP values of the j-th estimated state component for consecutive time increments.

N : IN, OUT, INTEGER
estimated order of the system.
On input, it is possible to use N = max(1, NOBL-1).

NOBL : IN, INTEGER
number of block rows in the processed input and output Hankel matrices. NOBL should be chosen larger than n, the estimated dimension of state vector.

TOL : IN, DOUBLE
absolute tolerance used for determining an estimate of the system order. If TOL > 0, the estimate is indicated by the number of singular values greater than or equal to TOL. (Singular values less than TOL are considered zero.)
If TOL = 0, the internally computed default value TOL = NOBL*EPS*SV[1] is used, where EPS is the relative machine precision, and SV[1] is the maximal singular value.
If TOL < 0, the estimate is indicated by the index of the singular value that has the largest logarithmic gap to its successor. (See Method for the definition of the singular values.)

WITHD : IN, LOGICAL
specifies whether or not a non-zero feedthrough matrix D to be included in the estimated state space representation:
WITHD = .TRUE. means a possibly non-zero D matrix should be estimated.
WITHD = .FALSE. means a zero D matrix is assumed.

CONCT : IN, LOGICAL
specifies whether or not the successive data batches belong to a single experiment:
CONCT = .TRUE. means the current data batch is a continuation of the previous data batch.
CONCT = .FALSE. means there is no connection between the current data batch and the previous ones.

LAST : IN, LOGICAL
specifies whether or not sequential data processing is used, and (for sequential processing) whether or not the current data block is the last one, as follows:
LAST = .TRUE. means non-sequential processing or the last data block in sequential processing.
LAST = .FALSE. means not the last data block in sequential processing. In this case the only calculation performed is to update the triangular factor

of the QR factorization.

CTRL : IN, LOGICAL
specifies whether or not the user's confirmation of the system order estimate is desired before the computation of system matrices, as follows:
CTRL = .TRUE. means user's confirmation is desired.
CTRL = .FALSE. means no confirmation is necessary.
If CTRL = .TRUE., a reverse communication routine, IMCORD, is called, and, after inspecting the singular values and system order estimate, n, the user may accept n or set a new value.
IMCORD is not called by the routine if CTRL = .FALSE..
CTRL = .TRUE. should be used in extreme cases only.

A : OUT, DOUBLE (NOBL-1,NOBL-1)
the N*N estimated system state matrix A.
(column dense)

B : OUT, DOUBLE (NOBL-1,M)
the N*M estimated system input matrix B.
(column dense)

C : OUT, DOUBLE (L,NOBL-1)
the L*N estimated system output matrix C.
(column dense)

D : OUT, DOUBLE (L,M)
the L*M estimated system feedthrough matrix D if WITHD = .TRUE.. D is not referenced if WITHD = .FALSE..
(column dense)

R : IN, OUT, DOUBLE (max(M*NOBL+N,L*NOBL)+L*NOBL,(M+L)*NOBL+N)
On output, if LAST = .FALSE., then the leading (M+L)*NOBL+N x (M+L)*NOBL+N part of R contains the current upper triangular factor of the QR factorization used to compress the data before determining the singular values in the MOESP algorithm.
The content of R should be preserved between succesive calls of RPIMRS with LAST = .FALSE..
On first input, the content of R is not meaningfull.

IWORK : OUT, INTEGER(M)
working array.

RWORK : OUT, DOUBLE (LRWORK)
working array.
RWORK(1) contains the optimal suggested value of LRWORK;
RWORK(2) contains an estimate of the reciprocal condition number for the final linear algebraic system solved;
RWORK(3:2+L*NOBL) contain the singular values used for determining the system order.
The first (M+N+L)*(NOBL-1) elements of RWORK should be preserved between succesive calls of the routine with LAST = .FALSE., and the final call with LAST = .TRUE.,

```
          if successive data batches belong to a single experiment
          (CONCT = .TRUE.).
          On return with error number 4, RWORK(1) contains the
          minimum necessary value of LRWORK.
 LRWORK:  IN, INTEGER
          dimension of working array RWORK.
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN = max( (NOBL+2)*((M+L)*NOBL+N),
                        2+L*NOBL + max( ((2*NOBL-3)*L+2)*N,
                                        (L*NOBL-N)*M*NOBL,
                                        4*(L*NOBL+M),5*(L*NOBL+M)-4 ))
          For  NSMP = 0,  the first term of the first max expression
          could be omitted.
          This estimate is computed using the given value for  n.
          For good performance,  LRWORK  should be larger.
 *      :  RETURN 1, target label in case of error (e.g. *1111)
```

File input/ output:

none

Method:

Let us denote m = M, l = L, t = NSPM, s = NOBL.
For non-sequential data processing, the t x (m+l)s+n matrix

$$\begin{bmatrix} U'_{1,s,t} & Xest' & Y'_{1,s,t} \end{bmatrix}$$

is constructed, where $U_{1,s,t}$ and $Y_{1,s,t}$ are Hankel matrices defined in terms of the input and output data /1/, /2/ and Xest is the estimated state trajectory.
A QR factorization is used to compress the data, and then a singular value decomposition (SVD) of the submatrix R_23 := R(ms+1:ms+n,ms+n+1:(m+l)s+n) of the upper triangular factor R reveals the order n of the system as the number of "non-zero" singular values. System matrices are finally computed from the right singular vectors of R_23 and the submatrices R_11 and R_13 of R.

For sequential data processing, the QR decomposition is done sequentially, by updating the upper triangular factor R. When all data have been compressed, the system order and system matrices are computed as in the previous case.

Literature

/1/ Verhaegen M., and P. Dewilde
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.

Int. J. Control, 56, pp. 1187-1210, 1992.
/2/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.

Remarks:

- The NSMP argument may vary from a cycle to another in sequential data processing, but NOBL, N, M, and L should be kept constant. For efficiency, it is advisable to use NSMP as large as possible.
- When 100 cycles of sequential data processing are completed with LAST = .FALSE., a warning is issued, to prevent for an infinite loop.

Copyright:

Life cycle:

1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

Consider the discrete-time system (A,B,C,D) with the following matrices:

$$A = \begin{pmatrix} 1.5 & -0.7 \\ 1.0 & 0.0 \end{pmatrix} \quad B = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad C = \begin{pmatrix} 1 & 0.5 \end{pmatrix} \quad D = 0$$

whose output response y(t) to random input signals u(t) and zero initial state is determined by using the subroutine RPIMIU. The following sequence of statements can be used to estimate the order and the matrices of the original system

```
      DATA   ISEED/ 1, 2, 3, 5 /
      NSMP   = 120
      NOBL   = 10
      N      = 2
      M      = 1
      L      = 1
      TOL    = -1.0D0
      WITHD  = .FALSE.
      LAST   = .TRUE.
      CONCT  = .TRUE.
      CTRL   = .TRUE.
      LRWORK = 8100
C     Compute the output response for uniformly distributed
```

```
C     input sequence.
      X(1)   = 0.0D0
      X(2)   = 0.0D0
      CALL RPIMIU( NSMP, N, M, L, 0, 0, A, B, E, C, D, F, X, U,
     *             W, V, ISEED, Y, RWORK, LRWORK, WITHD, 'G',
     *             'G', 'R', 'R', 'R', 1, *1111 )
C     Compute system state trajectory.
      CI   = 2 x 2 identity matrix
      CALL RPIMIU( NSMP, N, M, N, 0, 0, A, B, E, CI, D, F, X, U,
     *             W, V, ISEED, XEST, RWORK, LRWORK, .FALSE., 'G',
     *             'G', 'G', 'R', 'R', 1, *1111 )
C     Perform the system identification.
      CALL RPIMRS( U, NSMP, M, Y, L, XEST, N, NOBL, TOL, WITHD,
     *             CONCT, LAST, CTRL, A, B, C, D, R, IWORK,
     *             RWORK, LRWORK, *1111 )
```

The order detection singular values are:

3.2203D+01 1.4612D+01 0 0 0 0 0 0 0 0

The estimated system matrices are:

$$\bar{A} = \begin{pmatrix} .7831 & .5606 \\ -.2472 & .7169 \end{pmatrix} \quad \bar{B} = \begin{pmatrix} 3.5468 \\ 2.3707 \end{pmatrix} \quad \bar{C} = \begin{pmatrix} .5749 & -.4382 \end{pmatrix}$$

which coincide with the matrices obtained by applying a similarity transformation to the original system.

The relative output error is: $||y(t)-\bar{y}(t)|| \,/\, ||y(t)|| = 0.95410D-15$

Error Messages:

-1-
Invalid parameter value on entry.
-2-
Singular value decomposition failed.
-3-
A singular upper triangular matrix was found.
-4-
Not enough working storage. It should be at least //LENG//.

Warnings Messages:

-1-
More than 100 cycles in sequential data processing
-2-
All singular values are exactly zero (N = 0)

SUBROUTINE RPIMKG

MOESP-SI Deterministic-Stochastic Identification

Procedure purpose:

This subroutine determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices and, optionally, the noise covariance matrices and the Kalman gain matrix using the "ordinary MOESP algorithm with past inputs and outputs", combined with the "subspace state space system identification" algorithm, for covariance calculations. It is assumed that the input is a zero-mean white noise process.

The model structure is :

$$x(k+1) = Ax(k) + Bu(k) + w(k)$$
$$y(k) = Cx(k) + Du(k) + e(k)$$

where w(k) and e(k) are zero-mean white noise sequences, independent of the input u(k). Matrix D could be zero.

The Kalman predictor structure is :

$$\bar{x}(k+1) = A\bar{x}(k) + Bu(k) + K(y(k) - C\bar{x}(k) - Du(k))$$
$$\bar{y}(k) = C\bar{x}(k) + Du(k)$$

where K is the Kalman gain matrix corresponding to the estimated noise covariance matrices.
There is an option for sequential input-output data processing.

Usage:

```
CALL RPIMKG( U, NSMP, M, Y, L, NOBL, TOL, WITHD, WITHCV, WITHK, CONCT,
             LAST, CTRL, N, A, B, C, D, FK, R, IWORK, RWORK, LRWORK, *)
```

U : IN, DOUBLE (NSMP,M)
system input data sequence matrix U = [u_1 u_2 ... u_m].
Column j of U contains the NSMP values of the
j-th input component for consecutive time increments.

NSMP : IN, INTEGER
number of rows of matrices U and Y (number of samples)
(When sequential data processing is used, NSMP is the
number of samples of the current data batch.)
NSMP >= 2*(M+L+1)*NOBL - 1, for non-sequential processing
of the data;

NSMP >= 2*NOBL, for sequential processing of several data batches;
NSMP = 0, and LAST = .TRUE., is allowed if the upper triangular factor of the QR factorization is already available in the array R, and only the system matrices are to be computed.
The total number of samples when calling with LAST = .TRUE. should be at least 2*(M+L+1)*NOBL - 1.

M : IN, INTEGER
dimension of system input vector.

Y : IN, DOUBLE (NSMP,L)
system output data sequence matrix Y = [y_1 y_2 ... y_l]. Column j of Y contains the NSMP values of the j-th output component for consecutive time increments.

L : IN, INTEGER
dimension of system output vector.

NOBL : IN, INTEGER
number of block rows in the processed input and output Hankel matrices. NOBL should be chosen larger than n, the estimated dimension of state vector.

TOL : IN, DOUBLE
absolute tolerance used for determining an estimate of the system order. If TOL > 0, the estimate is indicated by the number of singular values greater than or equal to TOL. (Singular values less than TOL are considered zero.)
If TOL = 0, the internally computed default value TOL = NOBL*EPS*SV[1] is used, where EPS is the relative machine precision, and SV[1] is the maximal singular value.
If TOL < 0, the estimate is indicated by the index of the singular value that has the largest logarithmic gap to its successor. (See Method for the definition of the singular values.)

WITHD : IN, LOGICAL
specifies whether or not a non-zero feedthrough matrix D to be included in the estimated state space representation:
WITHD = .TRUE. means a possibly non-zero D matrix should be estimated.
WITHD = .FALSE. means a zero D matrix is assumed.

WITHCV: IN, LOGICAL
specifies whether or not the covariance matrices are to be computed:
WITHCV = .TRUE. means the covariance matrices are computed.
WITHCV = .FALSE. means the covariance matrices should not be computed.
The covariance matrices are automatically computed if WITHK = .TRUE..

```
WITHK :  IN, LOGICAL
         specifies whether or not the Kalman gain matrix  K  is to
         be computed:
         WITHK =  .TRUE.  means the Kalman gain matrix is computed.
         WITHK =  .FALSE. means the Kalman gain matrix is not
                          computed.
CONCT :  IN, LOGICAL
         specifies whether or not the successive data batches belong
         to a single experiment:
         CONCT = .TRUE.  means the current data batch is a
                         continuation of the previous data batch.
         CONCT = .FALSE. means there is no connection between the
                         current data batch and the previous ones.
LAST  :  IN, LOGICAL
         specifies whether or not sequential data processing is
         used, and (for sequential processing) whether or not the
         current data block is the last one, as follows:
         LAST =  .TRUE.  means non-sequential processing or the last
                         data block in sequential processing.
         LAST = .FALSE.  means not the last data block in sequential
                         processing. In this case the only calculation
                         performed is to update the triangular factor
                         of the QR factorization.
CTRL  :  IN, LOGICAL
         specifies whether or not the user's confirmation of the
         system order estimate is desired before the computation
         of system matrices, as follows:
         CTRL =  .TRUE. means user's confirmation is desired.
         CTRL = .FALSE. means no confirmation is necessary.
         If  CTRL =  .TRUE.,  a reverse communication routine,
         IMCORD,  is called, and, after inspecting the singular
         values and system order estimate,  n,  the user may
         accept  n  or set a new value.
         IMCORD  is not called by the routine if  CTRL = .FALSE..
         CTRL = .TRUE.  should be used in extreme cases only.
N     :  OUT, INTEGER
         estimated order of the system.
A     :  OUT, DOUBLE (NOBL-1,NOBL-1)
         the N*N estimated system state matrix A.
         (column dense)
B     :  OUT, DOUBLE (NOBL-1,M)
         the N*M estimated system input matrix B.
         (column dense)
C     :  OUT, DOUBLE (L,NOBL-1)
         the L*N estimated system output matrix C.
         (column dense)
D     :  OUT, DOUBLE (L,M)
```

```
          the L*M estimated system feedthrough matrix D if
          WITHD = .TRUE..
          D is not referenced if  WITHD = .FALSE..
          (column dense)
FK     :  OUT, DOUBLE (NOBL-1,L)
          if WITHK = .TRUE., on normal return, or when the
          warning errors 1 or 3 are issued, the leading N*L part
          of this array contains the Kalman gain matrix K.
          FK is not referenced when WITHK = .FALSE..
          (column dense)
R      :  IN, OUT, DOUBLE (max( 2*(M+L)*NOBL, 3*M*NOBL ),2*(M+L)*NOBL)
          On output, if  LAST = .FALSE., then the leading
          2*(M+L)*NOBL x 2*(M+L)*NOBL  part of  R  contains the
          current upper triangular factor of the QR factorization
          used to compress the data before determining the singular
          values in the  MOESP  algorithm.
          The content of  R  should be preserved between succesive
          calls of  RPIMKG  with  LAST = .FALSE..
          On first input, the content of R is not meaningfull.
          During the computations with  LAST = .TRUE.,  the leading
          M*NOBL * (M+L)*NOBL  part of  R  contains some relevant
          data for the  MOESP  algorithm (the concatenated
          compressed matrices R_11 = R_1C',  and  R_12 = R_2C').
          On output with  LAST = .TRUE.,  the content of  R  is
          destroyed; if  WITHCV = .TRUE.  or  WITHK = .TRUE.,
          on normal return, and in absence of the warning error 2,
          the first  N+L  rows and columns of  R  contain the estimated
          covariance matrix of the vector  [w', e']';
          if  WITHK = .TRUE.,  on normal return, or when the warning
          errors 1 or 3 are issued, the rows  N+L+1  to  2*N+L+1
          and the columns  1 to N  of  R  contain the estimation error
          covariance matrix  X  (the Riccati equation solution).
IWORK  :  OUT, INTEGER(K)
          working array.
          K = M, if WITHCV = .FALSE.  and  WITHK  = .FALSE.
          K = M*NOBL + NOBL-1, if  WITHCV = .TRUE., or WITHK = .TRUE..
RWORK  :  OUT, DOUBLE (LRWORK)
          working array.
          RWORK(1)  contains the optimal suggested value of LRWORK;
          RWORK(2)  contains an estimate of the reciprocal condition
          number for the final linear algebraic system solved for
          computing system matrices (in the absence of warning error 2);
          RWORK(3:L*NOBL+2)  contain singular values used for
          determining the system order;
          If  WITHK = .TRUE.,  LAST = .TRUE., on normal return,
          or when the warning errors 1 or 3 are issued, then:
          RWORK(L*NOBL+3)  contains an estimate of the reciprocal
```

```
          condition number of the computed output covariance matrix Ry;
          RWORK(L*NOBL+4)  contains an estimate of the reciprocal
          condition number of the computed matrix  A;
          RWORK(L*NOBL+5)  contains an estimate of the reciprocal
          condition number of the computed matrix  U_11, used to
          obtain the Riccati equation solution X;
          RWORK(L*NOBL+6)  contains an estimate of the reciprocal
          condition number of the matrix  Ry + C*X*C', used to
          obtain the Kalman gain matrix;
          RWORK(L*NOBL+7:L*NOBL+N+6)  contain the real parts of the
          optimal (Kalman predictor) poles;
          RWORK(L*NOBL+N+7:L*NOBL+2*N+6)  contain the imaginary
          parts of the optimal (Kalman predictor) poles.
          Part of this information is also available for warning
          errors 4 to 9, if logically possible.
          The first  (M+L)*(2*NOBL - 1)  elements of  RWORK  should
          be preserved between succesive calls of the routine with
          LAST = .FALSE.,  and the final call with  LAST = .TRUE.,
          if successive data batches belong to a single experiment
          (CONCT = .TRUE.).
          On return with error number 4, RWORK(1) contains the
          minimum necessary value of LRWORK.
 LRWORK:  IN, INTEGER
          dimension of working array RWORK.
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN = max( 4*(NOBL+1)*(M+L)*NOBL, 2+L*NOBL +
                        max( ((2*NOBL-3)*L+2)*(NOBL-1),
                             (L*NOBL-1)*M*NOBL, 4*(L*NOBL+M),
                             5*(L*NOBL+M)-4,
                             3 + max( 10*(NOBL-1), 3*L ),
                             5 + 2*(NOBL-1) + max( NOBL-1, 3*L ) ) )
          For NSMP = 0, the first term of the first max expression
          could be omitted. For  WITHK = .FALSE., the last two terms
          of the second max expression could be omitted.
          This estimate is computed using the largest and smallest
          values for  n, namely  NOBL-1  and 1, respectively (for
          positive and negative terms, respectively).
          For good performance,  LRWORK  should be larger.
 *      :  RETURN 1, target label in case of error (e.g. *1111)
```

File input/ output:

none

Method:

Let us denote m = M, l = L, t = NSPM, s = NOBL.

1) For non-sequential data processing, the t * 2(m+l)s matrix

$$\begin{bmatrix} Uf' & Up' & Y' \\ s+1,s,t & 1,s,t & 1,2s,t \end{bmatrix}$$

is constructed, where $Up_{1,s,t}$, $Uf_{s+1,s,t}$ and $Y_{1,2s,t}$ are Hankel matrices defined in terms of the input and output data [3]. A QR factorization is used to compress the data, and then a singular value decomposition (SVD) of the submatrix $[\ R\_24'\ \ R\_34'\ ]'$:= R(ms+1:(2m+1)s,(2m+1)s+1:2(m+1)s) of the upper triangular factor R reveals the order n of the system as the number of "non-zero" singular values. System matrices are finally computed using the right singular vectors of $[\ R\_24'\ \ R\_34'\ ]'$ and the final, "compressed" submatrices R_11 and R_12 of R.

2) For sequential data processing, the QR decomposition is done sequentially, by updating the upper triangular factor R. When all data have been compressed, the system order and system matrices are computed as in the previous case.

The covariance matrices are computed by solving several least squares problems, constructed based on the N4SID approach [4].

The Kalman gain is obtained by solving a discrete-time algebraic matrix Riccati equation using the Schur vectors approach [5] for the dual of an optimal control problem.

Literature

/1/ Verhaegen M., and P. Dewilde
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.
Int. J. Control, 56, pp. 1187-1210, 1992.

/2/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.

/3/ Verhaegen M.
Identification of the deterministic part of MIMO state space models given in innovations form from input-output data.
Automatica, Vol.30, No.1, pp.61-74, 1994.

/4/ Van Overschee, P., and De Moor, B.
N4SID: Two Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems.
Automatica, Vol.30, No.1, pp. 75-93, 1994.

/5/ Laub A. J.
A Schur method for solving algebraic Riccati equations.
IEEE Trans. Automat. Control, AC-24, pp. 913-921, 1979.

Remarks:

- The NSMP argument may vary from a cycle to another in sequential data processing, but NOBL, M, and L should be kept constant. For efficiency, it is advisable to use NSMP as large as possible.
- When 100 cycles of sequential data processing are completed with LAST = .FALSE., a warning is issued, to prevent for an infinite loop.

Copyright:

Life cycle:

1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

Consider the discrete-time system

$$x(k+1) = Ax(k) + Bu(k) + Ew(k)$$
$$y(k) = Cx(k) + Du(k) + Fv(k)$$

with the following matrices:

$$A = \begin{pmatrix} 1.5 & -0.7 \\ 1.0 & 0.0 \end{pmatrix} \quad B = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad C = \begin{pmatrix} 1 & 0.5 \end{pmatrix} \quad D = 0$$

$$E = \begin{pmatrix} 0.0005 & 0.0 \\ 0.0 & 0.006 \end{pmatrix} \quad F = \begin{pmatrix} 0.0528 \end{pmatrix}$$

whose output response y(t) to normal (0,1) random input signals u(t), w(t), v(t) and zero initial state is determined by using the subroutine RPIMIU.
The following sequence of statements can be used to estimate the order and the matrices of the original system

```
      DATA   ISEED/ 1, 2, 3, 5 /
      NSMP   = 120
      NOBL   = 6
      M      = 1
      MW     = 2
      L      = 1
```

```
      LV     = 1
      TOL    = -1.0D0
      WITHD  = .FALSE.
      WITHK  = .TRUE.
      LAST   = .TRUE.
      CONCT  = .TRUE.
      CTRL   = .TRUE.
      LRWORK = 8100
C     Compute the output response for uniformly distributed
C     input sequence.
      X(1)   = 0.0D0
      X(2)   = 0.0D0
      CALL RPIMIU( NSMP, 2, M, L, MW, LV, A, B, E, C, D, F, X, U,
     *             W, V, ISEED, Y, RWORK, LRWORK, WITHD, 'G',
     *             'G', 'R', 'R', 'R', 3, *1111 )
C     Perform the system identification.
      CALL RPIMKG( U, NSMP, M, Y, L, NOBL, TOL, WITHD, WITHCV,
     *             WITHK, CONCT, LAST, CTRL, N, A, B, C, D, FK,
     *             R, IWORK, RWORK, LRWORK, *1111 )
```

The order detection singular values are:

.87524260D+02 .42090540D+02 .32603442D+00 .22092251D+00 .19614963D+00
.16624421D+00

We can take the system order n = 2.
The estimated system matrices are:

$$\bar{A} = \begin{pmatrix} .8245 & .5418 \\ -.2643 & .6748 \end{pmatrix} \quad \bar{B} = \begin{pmatrix} 3.8186 \\ 1.6312 \end{pmatrix} \quad \bar{C} = \begin{pmatrix} .5093 & -.5822 \end{pmatrix}$$

The estimated Kalman gain matrix K is:

$$K = \begin{pmatrix} .1190 \\ -.1635 \end{pmatrix}$$

The estimated (inter-)covariance matrix (for [Ew; Fv]) is:

$$\begin{pmatrix} .0001 & -.0002 & .0003 \\ -.0002 & .0006 & .0000 \\ .0003 & .0000 & .0029 \end{pmatrix}$$

The estimated state covariance matrix X is:

$$\begin{pmatrix} .0012 & .0001 \\ .0001 & .0011 \end{pmatrix}$$

```
  The relative output error is: ||y(t)-ȳ(t)|| / ||y(t)|| =  .65877D-02
```

Error Messages:

```
-1-
Invalid parameter value on entry.
-2-
Singular value decomposition failed.
-3-
A singular upper triangular matrix was found.
-4-
Not enough working storage. It should be at least //LENG//.
```

Warnings Messages:

```
-1-
More than 100 cycles in sequential data processing
-2-
All singular values are exactly zero (N = 0)
-3-
The least squares problem for covariances is rank-deficient
-4-
The computed output covariance matrix Ry is too small
-5-
The covariance matrix Ry is numerically singular
-6-
The transformed matrix A is numerically singular
-7-
Schur form computation did not converged
-8-
Schur form ordering failed
-9-
The computed matrix U_11 is numerically singular
```

4. Auxiliary Tools for System Identification

To validate and extend the use of various subspace identification schemes, a number of auxiliary routines have been implemented:

IMMPID estimates a finite set of Markov parameters from input-output data (no constraints on input sequence to be white noise and the initial conditions to be zero).

IMSHFT shifts the input-output data sequences to compensate the dead-times in the components of the input vector.

IMICID estimates the initial state $x(0)$ of a linear time-invariant discrete-time system, given the matrix quadruple (A, B, C, D) and the input and output trajectories of the system.

IMCLID computes the parameters of a linear time-invariant (LTI) system operating in closed-loop with a LTI controller.

RPIMIU computes the output trajectory of a linear time-invariant discrete-time system, given the input trajectory and/or the initial state and output disturbances (or noise) trajectories.

SUBROUTINE IMMPID

Estimation of Markov Parameters from Input-Output Data

Procedure purpose:

This subroutine determines the order of a linear time-invariant discrete-time state space model and estimates a finite set of Markov parameters using the "ordinary MOESP algorithm with past inputs and outputs".

The model structure is :

$$x(k+1) = Ax(k) + Bu(k) + w(k)$$
$$y(k) = Cx(k) + Du(k) + e(k)$$

where w(k) and e(k) are zero-mean white noise sequences, independent of the noise-free input u(k).
Matrix D could be zero.

The set of Markov parameters is returned as

$$\begin{pmatrix} M_{11} & M_{12} & \dots & M_{1m} \\ M_{21} & M_{22} & \dots & M_{2m} \\ \vdots & \vdots & \dots & \vdots \\ M_{s1} & M_{s2} & \dots & M_{sm} \end{pmatrix}$$

where M_1:s,j are s x 1 matrices representing the impulse response of the system due to an impulse in the j-th input.

Usage:

```
CALL IMMPID( NOBL, NSMP, M, L, U, LDU, Y, LDY, N, PARM,
             LDPARM, R, LDR, IWORK, RWORK, LRWORK, TOL,
             CONCT, LAST, CTRL, IWARN, IERR )
```

NOBL : IN, INTEGER
number of block rows in the processed input and output Hankel matrices, s.
NOBL should be chosen larger than n, the estimated dimension of state vector.

NSMP : IN, INTEGER
number of rows of matrices U and Y (number of samples).
When sequential data processing is used, NSMP is the number of samples of the current data batch.
NSMP >= 2*(M+L+1)*NOBL - 1, for non-sequential processing;
NSMP >= 2*NOBL, for sequential processing;
NSMP = 0, and LAST = .TRUE., for computing the system matrices only, when the upper triangular factor of the QR factorization is already available in the array R.
The total number of samples when calling the routine with

LAST = .TRUE. should be at least 2*(M+L+1)*NOBL - 1.

M : IN, INTEGER
dimension of system input vector.

L : IN, INTEGER
dimension of system output vector.

U : IN, DOUBLE (LDU,M)
the NSMP*M input-data sequence matrix U,
U = [u_1 u_2 ... u_m]. Column j of U contains the NSMP values of the j-th input component for consecutive time increments.

LDU : IN, INTEGER.
leading dimension of the array U.
LDU >= NSMP.

Y : IN, DOUBLE (LDY,L)
the NSMP*L system output-data sequence matrix Y,
Y = [y_1 y_2 ... y_l]. Column j of Y contains the NSMP values of the j-th output component for consecutive time increments.

LDY : IN, INTEGER.
leading dimension of the array Y.
LDY >= NSMP.

N : OUT, INTEGER
the order of the system.

PARM : OUT, DOUBLE (LDPARM,L*M)
If IERR = 0 and IWARN <> 2, the leading NOBL x L*M part of this array contains the estimated Markov parameters. The first L columns represent the impulse response of the system due to an impulse in the first input; the next L columns represent the impulse response of the system due to an impulse in the second input and so on.

LDPARM: IN, INTEGER
leading dimension of the array PARM.
LDPARM >= NOBL.

R : OUT, DOUBLE (LDR,2*(M+L)*NOBL)
If LAST = .FALSE., the leading 2*(M+L)*NOBL x 2*(M+L)*NOBL part of this array contains the current upper triangular factor of the QR factorization used to compress the data before determining the singular values in the MOESP algorithm. The content of R should be preserved between succesive calls of the routine with LAST = .FALSE., and the final call with LAST = .TRUE..
During the computations with LAST = .TRUE., the leading M*NOBL x (M+L)*NOBL part of R contains some relevant data for the MOESP algorithm (the concatenated compressed matrices R_11 = R_1C', and R_12 = R_2C').

LDR : IN, INTEGER
leading dimension of the array R.

```
          LDR >= max( 2*(M+L)*NOBL, 3*M*NOBL ).
 IWORK :  OUT, INTEGER(M)
          working array.
 RWORK :  OUT, DOUBLE (LRWORK)
          On exit, if  LAST = .TRUE.,  and  IERR = 0:
          RWORK(1) contains the optimal suggested value of  LRWORK;
          RWORK(2) contains an estimate of the reciprocal condition
                   number for the linear algebraic system solved,
                   if IWARN <> 2;
          RWORK(3:2+L*NOBL) contain singular values used for
                            determining the system order.
          The first  (M+L)*(2*NOBL - 1)  elements of  RWORK  should
          be preserved between succesive calls of the routine with
          LAST = .FALSE.,  and the final call with  LAST = .TRUE.,
          if successive data batches belong to a single experiment
          (CONCT = .TRUE.).
          On exit, if  IERR = -1,
          RWORK(1) contains the minimal (estimated) value of  LRWORK.
LRWORK :  IN, INTEGER.
          dimension of working array RWORK.
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN = max( 4*(NOBL+1)*(M+L)*NOBL, 2+L*NOBL +
                        max( (L*NOBL-1)*M*NOBL, 4*(L*NOBL+M),
                             5*(L*NOBL+M)-4 ) )
          For  NSMP = 0,  the first term of the first max
          expression could be omitted.
          This is an overestimate computed using the largest and
          smallest values for  n, namely  NOBL-1  and 1, respectively
          (for positive and negative terms, respectively).
          For good performance,  LRWORK  should be large.
  TOL  :  IN, DOUBLE PRECISION.
          absolute tolerance used for determining an estimate of
          the system order. If  TOL >= 0,  the estimate is indicated
          by the index of the last singular value greater than or equal
          to  TOL.  (Singular values less than  TOL are considered as
          zero.) If TOL = 0,  an internally computed default value,
          TOL = NOBL*EPS*SV[1],  is used, where  EPS  is the relative
          machine precision, and SV[1] is the maximal singular value.
          If TOL < 0,  the estimate is indicated by the index of the
          singular value that has the largest logarithmic gap to its
          successor.
 CONCT :  IN, LOGICAL
          specifies whether or not the successive data batches in
          sequential data processing belong to a single experiment:
          CONCT = .TRUE.  the current data batch is a continuation
                          of the previous data batch.
          CONCT = .FALSE. there is no connection between the current
```

```
                        data batch and the previous ones.
LAST   :  IN, LOGICAL
          specifies whether or not sequential data processing is
          used, and (for sequential processing) whether or not the
          current data block is the last one:
          LAST =  .TRUE. non-sequential processing or the last
                         data block in sequential processing.
          LAST = .FALSE. not the last data block in sequential
                         processing. In this case the only calculation
                         performed is to update the triangular factor
                         of the QR factorization.
CTRL   :  IN, LOGICAL
          specifies whether or not the user's confirmation of the
          system order estimate is desired, before the computation
          of system matrices, as follows:
          CTRL =  .TRUE. means user's confirmation.
          CTRL = .FALSE. means no confirmation is necessary.
          If  CTRL =  .TRUE.,  a reverse communication routine, IMCORD,
          is called, and, after inspecting the singular values and
          system order estimate,  n,  the user may accept  n  or set a
          new value.
          IMCORD  is not called by the routine if CTRL = .FALSE..
          CTRL = .TRUE.  should be used in extreme cases only.
IWARN  :  OUT, INTEGER
          unless the routine generates a warning, IWARN contains 0
          on exit.
          =  1 : The number of 100 cycles in sequential data
                 processing has been exhausted without signaling
                 that the last block of data was got.

          =  2 : All singular values were exactly zero, hence  N = 0.
                 (Both input and output were identically zero.)
                 The Markov parameters could not be determined.
IERR   :  OUT, INTEGER
          unless the routine detects an error, IERR contains 0 on exit.
          =  1 : On entry, NOBL < 1 or M < 1 or L < 1 or
                 LDU  < NSMP or LDY < NSMP or NSMP < 0 or
                 NSMP = 0 (and LAST = .FALSE.) or
                 NSMP < 2*(M+L+1)*NOBL-1 (and NSMP > 0 and one data
                                          block and LAST = .TRUE.) or
                 NSMP < 2*NOBL (and LAST = .FALSE.) or
                 SUM(NSMP) < 2*(M+L+1)*NOBL-1 (and LAST = .TRUE.) or
                 LDR  < max(2*(M+L)*NOBL,3*M*NOBL) or
                 LDPARM < NOBL.
          = -1 : On entry, LRWORK < max( 4*(NOBL+1)*(M+L)*NOBL,
                                2+L*NOBL +
                                max( (L*NOBL-1)*M*NOBL, 4*(L*NOBL+M),
```

5*(L*NOBL+M)-4))
(For NSMP = 0, the first term of the first max expression could be omitted.)
The minimal value of LRWORK is returned in RWORK(1).
= 2 : The singular value decomposition (SVD) algorithm did not converged.
= 3 : A singular upper triangular matrix was found.

File input/ output:
none

Method:
Let us denote m = M, l = L, t = NSPM, s = NOBL.

For non-sequential data processing, the t x 2(m+l)s matrix

$$\begin{bmatrix} Uf'_{s+1,s,t} & Up'_{1,s,t} & Y'_{1,2s,t} \end{bmatrix}$$

is constructed, where $Up_{1,s,t}$, $Uf_{s+1,s,t}$ and $Y_{1,2s,t}$ are Hankel matrices defined in terms of the input and output data [3]. A QR factorization is used to compress the data, and then a singular value decomposition (SVD) of the submatrix [R_24' R_34']' := R(ms+1:(2m+l)s,(2m+l)s+1:2(m+l)s) of the upper triangular factor R reveals the order n of the system as the number of "non-zero" singular values. Markov parameters are finally computed using the right singular vectors of [R_24' R_34']' and the final, "compressed" submatrices R_11 and R_12 of R.

For sequential data processing, the QR decomposition is done sequentially, by updating the upper triangular factor R. When all data have been compressed, the system order and Markov parameters are computed as in the previous case.

Literature

/1/ Verhaegen M., and P. Dewilde
Subspace Model Identification. Part 1: The output-error state-space model identification class of algorithms.
Int. J. Control, 56, pp. 1187-1210, 1992.

/2/ Verhaegen M.
Subspace Model Identification. Part 3: Analysis of the ordinary output-error state-space model identification algorithm.
Int. J. Control, 58, pp. 555-586, 1993.

/3/ Verhaegen M.
Identification of the deterministic part of MIMO state space models given in innovations form from input-output data.
Automatica, Vol.30, No.1, pp.61-74, 1994.

Remarks:

- The NSMP argument may vary from a cycle to another in sequential data processing, but NOBL, M, and L should be kept constant. For efficiency, it is advisable to use NSMP as large as possible.
- When 100 cycles of sequential data processing are completed with LAST = .FALSE., a warning is issued, to prevent for an infinite loop.

Copyright:

Life cycle:

1994 December V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

Consider the original discrete-time system with the following matrices:

```
A = ( 1.5   -0.7 )     B = ( 1 )     C = ( 1   0.5 )
    ( 1.0    0.0 )         ( 0 )
```

whose output response y(t) to (0,1) random input signals u(t) and zero initial state is determined by using the subroutine RPIMIU. The following sequence of statements can be used to estimate the order and the Markov parameters of the original system:

```
      DATA   ISEED/ 1, 2, 3, 5 /
      NSMP   =   120
      NOBL   =   6
      M      = 1
      L      = 1
      TOL    =   -.10000D+01
      WITHD  =   .FALSE.
      CONCT  =   .TRUE.
      LAST   =   .TRUE.
      CTRL   =   .TRUE.
      LDR    =   MAX( 2*(M+L)*NOBL, 3*M*NOBL )
      LRWORK =   8100
```

```
C     Compute the output response for uniformly distributed
C     input sequence.
      X(1)   = 0.0D0
      X(2)   = 0.0D0
      CALL RPIMIU( NSMP, 2, M, L, 0, 0, A, B, E, C, D, F, X, U,
     *             W, V, ISEED, Y, RWORK, LRWORK, WITHD, 'G',
     *             'G', 'R', 'R', 'R', 1, *1111 )


C       Compute the system order and Markov parameters.
        CALL IMMPID( NOBL, NSMP, M, L, U, NSMP, Y, NSMP, N,
     *               PARM, NOBL, R, LDR, IWORK, RWORK, LRWORK,
     *               TOL, CONCT, LAST, CTRL, IWARN, IERR )
```

Singular values (in descending order) used to estimate the system order is:

```
2.9705D+01  1.3043D+01  0  0  0  0
```

The system order is n = 2. The estimated Markov parameters matrix is:

```
PARM = ( 0.000,  1.000, 2.000, 2.300, 2.050, 1.465 )
```

The maximum relative error (for each column of PARM) is 0.12120D-14.

SUBROUTINE IMSHFT

I/O Data Shifting to Compensate Input Vector Dead-Times

Procedure purpose:

This subroutine shifts the input and output data so that a linear time-invariant system with zero dead-time is obtained. It allows the different input components to have different delays. The initial model structure is :

$$x(k+1) = Ax(k) + \sum_{j=1}^{m} (B_j\, u_j(k-t_j)), \quad k \geq tmax + 1, \quad tmax := \max(t_j),$$

$$y(k) = Cx(k) + \sum_{j=1}^{m} (D_j\, u_j(k-t_j))$$

where x(k) is the n-dimensional state vector (at time k),
u(k) is the m-dimensional input vector,
y(k) is the l-dimensional output vector,
and A, B, C, and D are real matrices of appropriate dimensions.
The final model structure is :

$$\bar{x}(k+1) = A\bar{x}(k) + B\bar{u}(k), \quad k \geq 1,$$

$$\bar{y}(k) = C\bar{x}(k) + D\bar{u}(k),$$

where $\bar{k} = k - tmax$, $\bar{u}_j(1) = u_j(tmax - t_j + 1)$, $j = 1{:}m$,

$\bar{y}(1) = y(tmax + 1)$, $\bar{x}(1) = x(tmax + 1)$.

Usage:

```
CALL IMSHFT( LTM, NSMP, M, L, U, LDU, Y, LDY, IERR )
```

LTM : IN, INTEGER (M)
contains the dead-times of the system: LTM(j) is the dead-time corresponding to the j-th input component.

NSMP : IN, OUT, INTEGER
On input : number t of given input and output data values.
On output : number of data values that were shifted.
NSMP > 0.

M : IN, INTEGER
dimension of system input vector.

L : IN, INTEGER
dimension of system output vector.

U : IN, OUT, DOUBLE (NSMP,M)
On input : system input data sequence matrix

```
                    U = [u_1 u_2 ... u_m]. The j-th column of U
                    contains the NSMP values of the j-th input
                    component for consecutive time increments.
          On output : contains the shifted values of the input data
                    sequence.
 LDU    :  IN, INTEGER
          leading dimension of array U.
          LDU >= NSMP.
 Y      :  IN, OUT, DOUBLE (NSMP,L)
          On input  : system output data sequence matrix
                    Y = [y_1 y_2 ... y_l]. The j-th column of Y
                    contains the  NSMP  values of the j-th output
                    component for consecutive time increments.
          On output : contains the shifted values of the output data
                    sequence.
 LDY    :  IN, INTEGER
          leading dimension of array Y.
          LDY >= NSMP.
 IERR   :  OUT, INTEGER
          unless the routine detects an error, IERR contains 0 on exit.
          =  1 : On entry, NSMP < 1 or M < 1 or L < 1 or
                 LDU < NSMP or LDY < NSMP.
```

File input/ output:

none

Method:

The input and output sequences are shifted, taking into account the maximal dead-time of the system.

Copyright:

Life cycle:

1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

none

SUBROUTINE IMICID

Estimation of the Initial Conditions

Procedure purpose:

To estimate the initial state x(0) of a linear time-invariant (LTI) discrete-time system, given the matrix quadruple (A,B,C,D) and the input and output trajectories of the system.
The model structure is :

$$x(k+1) = Ax(k) + Bu(k), \qquad k >= 0,$$
$$y(k) = Cx(k) + Du(k),$$

where x(k) is the n-dimensional state vector (at time k),
u(k) is the m-dimensional input vector,
y(k) is the l-dimensional output vector,
and A, B, C and D are real matrices of appropriate dimensions.

Usage:

```
CALL IMICID( N, M, L, A, LDA, B, LDB, C, LDC, D, LDD,
             U, LDU, Y, LDY, XO, IWORK, RWORK, LRWORK,
             WITHD, WITHSV, IWARN, IERR )
```

N : IN, INTEGER
the number of system states.

M : IN, INTEGER
dimension of system input vector.

L : IN, INTEGER
dimension of system output vector.

A : IN, DOUBLE (LDA,N)
the N*N state matrix A.

LDA : IN, INTEGER.
leading dimension of the array A.
LDA >= N.

B : IN, DOUBLE (LDB,M)
the N*M input matrix B.

LDB : IN, INTEGER.
leading dimension of the array B.
LDB >= N.

C : IN, DOUBLE (LDC,N)
the L*N output matrix C.

LDC : IN, INTEGER.
leading dimension of the array C.
LDC >= L.

```
D      :  IN, DOUBLE (LDD,M)
          the L*M feedthrough matrix D.
LDD    :  IN, INTEGER.
          leading dimension of the array D.
          LDD >= L.
U      :  IN, DOUBLE (LDU,M)
          the N*M input-data sequence matrix  U,
          U = [u_1 u_2 ... u_m].  Column  j  of  U  contains the
          N  values of the j-th input component for consecutive
          time increments.
LDU    :  IN, INTEGER.
          leading dimension of the array U.
          LDU >= N.
Y      :  IN, DOUBLE (LDY,L)
          the N*L system output-data sequence matrix Y,
          Y = [y_1 y_2 ... y_l]. Column j of Y contains the N
          values of the j-th output component for consecutive time
          increments.
LDY    :  IN, INTEGER.
          leading dimension of the array Y.
          LDY >= N.
XO     :  OUT, DOUBLE (N)
          estimated initial state of the system  x(0).
IWORK  :  OUT, INTEGER(N)
          working array.
RWORK  :  OUT, DOUBLE (LRWORK)
          working array.
          On exit, if  IERR = 0,  then:
          RWORK(1)  contains the optimal suggested value of LRWORK;
          RWORK(2)  contains an estimate of the reciprocal
                    condition number of the triangular factor of
                    QR factorization.
                    (RWORK(2)  is also set if  IERR = 2.)
LRWORK:   IN, INTEGER.
          dimension of working array RWORK.
          LRWORK >= L*N*(N+1) + 2*N,  if  WITHSV = .FALSE.;
          LRWORK >= L*N*(N+1) + N + max( 4*N, 5*N - 4 ),
                                        if  WITHSV = .TRUE..
          For good performance,  LRWORK  should be larger.
 WITHD:   IN, LOGICAL.
          specifies whether or not a non-zero feedthrough matrix
          D  is to be included in the model, as follows:
          WITHD =  .TRUE., (a possibly non-zero matrix  D  should
                           be included);
          WITHD = .FALSE., (a zero matrix  D  is assumed).
WITHSV:   IN, LOGICAL
          specifies whether or not singular value decomposition is
```

```
          requested to solve the least squares problem, as follows:
          WITHSV =  .TRUE., (singular value decomposition is requested);
          WITHSV = .FALSE., (singular value decomposition is not
                             requested).
          The routine may override this option, if appropriate
          (see IWARN).
 IWARN :  OUT, INTEGER
          unless the routine generates a warning, IWARN contains 0
          on exit.
          = 1 : SVD was requested, but it is not needed and was not
                used.
          = 2 : SVD was not requested, but it is needed and was
                used.
 IERR  :  OUT, INTEGER
          unless the routine detects an error, IERR contains 0 on exit.
          =  1 : On entry, N < 1 or M < 1 or L < 1
                 or LDA < N or LDC < L or LDB < N or LDD < L
                 or LDU < N or LDY < N.
          = -1 : On entry, LRWORK < L*N*(N+1) + 2*N
                                    (and  WITHSV = .FALSE.) or
                           LRWORK < L*N*(N+1) + N
                                              + max( 4*N, 5*N - 4 )
                                    (and  WITHSV = .TRUE.).
                 The minimal value of LRWORK is returned in RWORK(1).
          =  2 : SVD was not requested, but the least squares
                 problem has not full column rank.
                 The upper triangular matrix of QR factorization was
                 found exactly or numerically singular, but there is
                 not enough workspace to perform the SVD.
          =  3 : The SVD algorithm did not converged.
```

File input/ output:

none

Method:

The output y0(k) of the system for zero initial state is computed for k = 0, 1,..., n-1 using the given model. Then the following least squares problem is solved for x(0)

$$\begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix} x(0) = \begin{pmatrix} y(0)-y0(0) \\ y(1)-y0(1) \\ \vdots \\ y(n-1)-y0(n-1) \end{pmatrix}$$

The QR decomposition of this matrix is computed. If its triangular factor is too ill conditioned, then singular value decomposition is used.

Literature

/1/ Verhaegen M., and A. Varga
Some Experience with the MOESP Class of Subspace Model
Identification Methods in Identifying the BO105 Helicopter.
Report TR R165-94, DLR Oberpfaffenhofen, Oberpfaffenhofen, 1994.

Copyright:

Life cycle:

1994 December V. Sima, Institut für Informatik, Bukarest: coded
1995 October A. Varga, DLR Oberpfaffenhofen,-Oberpfaffenhofen: revised

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

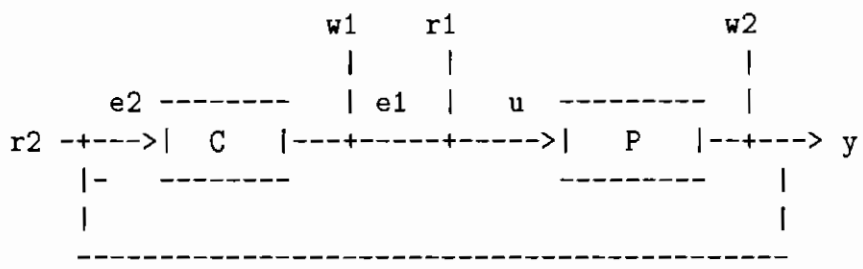
none

SUBROUTINE IMCLID

Closed-Loop Identification of Linear Systems

Procedure purpose:

This subroutine computes non-minimal state space models of a linear time-invariant plant P and/or controller C operating in the closed-loop configuration of the figure below, using a state space model estimate with matrices A, B, C and D, inputs r1 and/or r2 and outputs the signals e1 and/or e2, u and y.



where:

[r1,r2]	external excitations;
[w1,w2]	external disturbances;
[e1,e2,u,y]	internal recordable signals.

The given model of the closed-loop system is [A, B]
[C, D]

and its signals are:

	inputs	outputs	WITHP	WITHC
(*)	[r1,r2]	[e1,e2,u,y]	.TRUE.	.TRUE.
	[r1]	[e1,u,y] (or *)	.TRUE.	.FALSE.
	[r2]	[e2,u,y] (or *)	.FALSE.	.TRUE.

Usage:

```
CALL IMCLID( N, M, L, ML, LL, A, LDA, B, LDB, C, LDC, D, LDD,
             AP, LDAP, BP, LDBP, CP, LDCP, DP, LDDP, AC, LDAC,
             BC, LDBC, CC, LDCC, DC, LDDC, IWORK, RWORK, LRWORK,
             WITHP, WITHC, IERR )
```

N : IN, INTEGER
order of the system (closed-loop, plant, controller).

M : IN, INTEGER
number of plant inputs.

L : IN, INTEGER
number of plant outputs.

ML : IN, INTEGER
number of closed-loop system inputs.

```
          ML >= M          if  WITHP = .TRUE.;
          ML >= L          if  WITHC = .TRUE.;
          ML  = M + L      if  WITHP = .TRUE.  and  WITHC = .TRUE..
LL    :   IN, INTEGER
          number of closed-loop system outputs.
          LL >= 2*M+L      if  WITHP = .TRUE.;
          LL >= 2*L+M      if  WITHC = .TRUE.;
          LL  = 2*(M+L)    if  WITHP = .TRUE.  and  WITHC = .TRUE..
A     :   IN, DOUBLE (LDA,N)
          the closed-loop system state matrix.
LDA   :   IN, INTEGER
          leading dimension of the array A.
          LDA >= N.
B     :   IN, DOUBLE (LDB,ML)
          the closed-loop system input matrix.
LDB   :   IN, INTEGER
          leading dimension of the array B.
          LDB >= N.
C     :   IN, DOUBLE (LDC,N)
          the closed-loop system output matrix.
LDC   :   IN, INTEGER
          leading dimension of the array C.
          LDC >= LL.
D     :   IN, DOUBLE (LDD,ML)
          the matrix relating the input and output of the
          closed-loop system.
LDD   :   IN, INTEGER
          leading dimension of the array D.
          LDD >= LL.
AP    :   OUT, DOUBLE (LDAP,N)
          on exit, if  IERR = 0,  or  IERR = 3,  AP  contains
          the plant state matrix  Ap.
          AP  is not referenced if  WITHP = .FALSE..
LDAP  :   IN, INTEGER
          leading dimension of the array AP.
          LDAP >= N  if  WITHP = .TRUE.  and  LDAP >= 1, otherwise.
BP    :   OUT, DOUBLE (LDBP,M)
          on exit, if  IERR = 0,  or  IERR = 3,  BP  contains the
          plant input matrix  Bp.
          BP  is not referenced if  WITHP = .FALSE..
LDBP  :   IN, INTEGER
          leading dimension of the array BP.
          LDBP >= N  if  WITHP = .TRUE.  and  LDBP >= 1, otherwise.
CP    :   OUT, DOUBLE (LDCP,N)
          on exit, if  IERR = 0,  or  IERR = 3,  CP  contains the
          plant output matrix  Cp.
          CP  is not referenced if  WITHP = .FALSE..
```

LDCP : IN, INTEGER
leading dimension of the array CP.
LDCP >= L if WITHP = .TRUE. and LDCP >= 1, otherwise.

DP : OUT, DOUBLE (LDDP,M)
on exit, if IERR = 0, or IERR = 3, DP contains the matrix relating the input and output of the plant Dp.
DP is not referenced if WITHP = .FALSE..

LDDP : IN, INTEGER
leading dimension of the array DP.
LDDP >= L if WITHP = .TRUE. and LDDP >= 1, otherwise.

AC : OUT, DOUBLE (LDAC,N)
on exit, if IERR = 0, or IERR = 2, AC contains the controller state matrix Ac.
AC is not referenced if WITHC = .FALSE..

LDAC : IN, INTEGER
leading dimension of the array AC.
LDAC >= N if WITHC = .TRUE. and LDAC >= 1, otherwise.

BC : OUT, DOUBLE (LDBC,L)
on exit, if IERR = 0, or IERR = 2, BC contains the controller input matrix Bc.
BC is not referenced if WITHC = .FALSE..

LDBC : IN, INTEGER
leading dimension of the array BC.
LDBC >= N if WITHC = .TRUE. and LDBC >= 1, otherwise.

CC : OUT, DOUBLE (LDCC,N)
on exit, if IERR = 0, or IERR = 2, CC contains the controller output matrix Cc.
CC is not referenced if WITHC = .FALSE..

LDCC : IN, INTEGER
leading dimension of the array CC.
LDCC >= M if WITHC = .TRUE. and LDCC >= 1, otherwise.

DC : OUT, DOUBLE (LDDC,L)
on exit, if IERR = 0, or IERR = 2, DC contains the matrix relating the input and output of the controller Dc.
DC is not referenced if WITHC = .FALSE..

LDDC : IN, INTEGER
leading dimension of the array DC.
LDDC >= M if WITHC = .TRUE. and LDDC >= 1, otherwise.

IWORK : OUT, INTEGER(max(M,L))
working array.

RWORK : OUT, DOUBLE (LRWORK)
working array.
RWORK(1) contains an estimate of the reciprocal condition number of the matrix D_31 (if WITHP = .TRUE.);
RWORK(2) contains an estimate of the reciprocal condition number of the matrix D_22 (if WITHC = .TRUE.).
(See Method.)

```
LRWORK:  IN, INTEGER
         dimension of working array RWORK.
         The value of LRWORK must be at least LRWMIN, where
         LRWMIN = (N + M + 4)*M  if  WITHP = .TRUE.;
         LRWMIN = (N + L + 4)*L  if  WITHC = .TRUE.;
         LRWMIN = max( (N+M+4)*M, (N+L+4)*L )
                  if  WITHP = .TRUE.  and  WITHC = .TRUE..
WITHP :  IN, LOGICAL
         specifies whether or not the plant model is to be computed,
         as follows:
         WITHP =  .TRUE., (the plant model should be computed);
         WITHP = .FALSE., (the plant model should not be computed).
WITHC :  IN, LOGICAL
         specifies whether or not the controller model is to be
         computed, as follows:
         WITHC =  .TRUE., (the controller model should be
                           computed);
         WITHC = .FALSE., (the controller model should not be
                           computed).
IERR  : OUT, INTEGER
        Unless the routine detects an error, IERR contains 0 on exit.
        =  1 : On entry, N < 1 or M < 1 or L < 1 or
               ML < M (and WITHP = .TRUE.) or
               ML < L (and WITHC = .TRUE.) or
               ML <> M + L (and WITHP =.TRUE. and WITHC =.TRUE.) or
               LL < 2*M+L (and WITHP = .TRUE.) or
               LL < 2*L+M (and WITHC = .TRUE.) or
               LL <> 2*(M+L) (and WITHP = .TRUE. and WITHC =.TRUE.)
               or LDA  < N or LDB  < N or LDC  < LL or LDD  < LL or
               or LDAP < N or LDBP < N or LDCP < L or LDDP < L
                  (and  WITHP = .TRUE.)
               or LDAP < 1 or LDBP < 1 or LDCP < 1 or LDDP < 1
                  (and  WITHP = .FALSE.)
               or LDAC < N or LDBC < N or LDCC < M or LDDC < M or
                  (and  WITHC = .TRUE.)
               or LDAC < 1 or LDBC < 1 or LDCC < 1 or LDDC < 1
                  (and  WITHC = .FALSE.).
        = -1 : On entry,
               LRWORK < (N + M + 4)*M  if  WITHP = .TRUE.;
               LRWORK < (N + L + 4)*L  if  WITHC = .TRUE.;
               LRWORK < max((N+M+4)*M, (N+L+4)*L)
                        if  WITHP = .TRUE.  and  WITHC = .TRUE..
               The minimal value of  LRWORK  is returned in
               RWORK(1).
        =  2 : The matrix  D_31  is exactly singular.
               The plant matrices could not be computed.
        =  3 : The matrix  D_22  is exactly singular.
```

The controller matrices could not be computed.

File input/ output:
none

Method:
The computing formulas for the plant model are as follows:

$$Ap = A - B_1 D_{31}^{-1} C_3, \qquad Bp = B_1 D_{31}^{-1},$$

$$Cp = C_4 - D_{41} D_{31}^{-1} C_3, \qquad Dp = D_{41} D_{31}^{-1},$$

where B_1, C_3, C_4, D_31, and D_41 are defined in terms of A, B, C, D /1/. Similar relations hold for the controller model.

Literature

/1/ Verhaegen M.
Application of a Subspace Model Identification technique
to identify LTI systems operating in closed-loop.
Automatica, 29, pp. 1027-1040, 1993.

Copyright:

Life cycle:
1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:
RASP, BLAS (1,2,3), LAPACK

Example:
none

SUBROUTINE RPIMIU

Simulation of Linear Time-Invariant Discrete-Time Systems

Procedure purpose:

This subroutine computes the output trajectory of a linear time-invariant discrete-time system, given the input trajectory and/or the initial state and output disturbances (or noise) trajectories.

The model structure is :

$$x(k+1) = Ax(k) + Bu(k) + Ew(k), \quad k >= 1,$$
$$y(k) = Cx(k) + Du(k) + Fv(k),$$

where x(k) is the n-dimensional state vector (at time k),
u(k) is the m-dimensional input vector,
y(k) is the l-dimensional output vector,
w(k) is the mw-dimensional state disturbance vector,
v(k) is the lv-dimensional output disturbance vector,
and A, B, C, D, E, and F are real matrices of appropriate dimensions. The matrix D is taken as zero if WITHD = .FALSE..

Usage:

```
CALL RPIMIU( NSMP, N, M, L, MW, LV, A, B, E, C, D, F,
             X, U, W, V, ISEED, Y, RWORK, LRWORK, WITHD,
             UNITE, UNITF, UTRJ, WTRJ, VTRJ, IDIST, * )
```

NSMP : IN, INTEGER
number of time steps where y should be computed.
NSMP > 0.

N : IN, INTEGER
number of system states.

M : IN, INTEGER
dimension of system input vector.
M >= 0.

L : IN, INTEGER
dimension of system output vector.

MW : IN, INTEGER
number of state disturbances.
MW >= 0.

LV : IN, INTEGER
number of output disturbances.
LV >= 0.

A : IN, DOUBLE (N,N)
the N*N system state matrix A.
(column dense)

```
B     :  IN, DOUBLE (N,M)
         the N*M system input matrix B.
         B  is not referenced if M = 0.
         (column dense)
E     :  IN, DOUBLE (N,MW)
         the N*MW state disturbance matrix E.
         E is not referenced if MW = 0 or UNITE = 'I'.
         (column dense)
C     :  IN, DOUBLE (L,N)
         the L*N output matrix C.
         (column dense)
D     :  IN, DOUBLE (L,M)
         the L*M input-output matrix D if WITHD = .TRUE..
         D is not referenced if  M = 0 or WITHD = .FALSE..
         (column dense)
F     :  IN, DOUBLE (L,LV)
         the L*LV output disturbance matrix F.
         F is not referenced if  LV = 0  or  UNITF = 'I'.
X     :  IN, OUT, DOUBLE (N)
         On input  : the initial state of the system X,
                     X = [x_1 x_2 ... x_n].
         On output : the final state of the system.
                     The state trajectory is computed, but not stored.
U     :  IN, OUT, DOUBLE (NSMP,M)
         On input  : the NSMP*M input data sequence matrix  U,
                     U = [u_1 u_2 ... u_m].
                     If UTRJ = 'G', the j-th column of U must contain
                     the NSMP values of the j-th input component for
                     consecutive time increments.
                     If UTRJ = 'I', or UTRJ = 'S', only the first row
                     of U, containing the values for the impulse or
                     step signal applied to the system, must be given.
                     U should not be given if UTRJ = 'R'.
                     U is not referenced if  M = 0.
         On output : if  M > 0  and  UTRJ = 'R',  U  contains the random
                     input trajectory; otherwise  U  is unchanged.
         (column dense)
W     :  IN, OUT, DOUBLE (NSMP,MW)
         On input  : the NSMP*MW state disturbance (noise) data sequence
                     matrix W,  W = [w_1 w_2 ... w_mw].
                     If WTRJ = 'G', the j-th column of W contains the
                     NSMP values of the j-th state disturbance component
                     for consecutive time increments.
                     If WTRJ = 'I', or WTRJ = 'S', only the first row
                     of W, containing the values for the impulse or
                     step signal applied to the system, must be given.
                     W should not be given if WTRJ = 'R'.
```

```
                  W is not referenced if MW = 0.
          On output : if  MW > 0  and  WTRJ = 'R',  the first row of
                      W  contains the last randomly generated state
                      disturbance vector; otherwise  W  is unchanged.
          (column dense)
V       : IN, OUT, DOUBLE (NSMP,LV)
          On input  : the NSMP*LV output disturbance (noise) data
                      sequence matrix V,  V = [v_1 v_2 ... v_lv].
                      If VTRJ = 'G', the j-th column of V contains the
                      NSMP values of the j-th output disturbance
                      component for consecutive time increments.
                      If VTRJ = 'I', or VTRJ = 'S', only the first row
                      of V, containing the values for the impulse or
                      step signal applied to the system, must be given.
                      V should not be given if VTRJ = 'R'.
                      V is not referenced if LV = 0.
          On output : if  LV > 0  and  VTRJ = 'R',  the first row of
                      V  contains the last randomly generated output
                      disturbance vector; otherwise  V  is unchanged.
          (column dense)
ISEED   : IN, OUT, INTEGER(4)
          On input  : If either UTRJ = 'R', or WTRJ = 'R', or
                      VTRJ = 'R', ISEED contain the seed of the
                      random number generator; the array elements must
                      be between 0 and 4095, and ISEED(4) must be odd.
                      The random multidimensional deviates are computed
                      at each time step, in the order: V, U, and W.
                      ISEED is not referenced if neither UTRJ, nor
                      WTRJ, nor VTRJ is set to 'R'.
          On output : If either  UTRJ = 'R',  or  WTRJ = 'R',  or
                      VTRJ = 'R',  ISEED  contain the final updated
                      seed of the random number generator.
Y       : OUT, DOUBLE (NSMP,L)
          the NSMP*L system output data sequence matrix Y,
          Y = [y_1 y_2 ... y_l]. The j-th column of Y contains the NSMP
          values of the j-th output component for consecutive time
          increments.
          (column dense)
RWORK   : OUT, DOUBLE (LRWORK)
          working array.
LRWORK: IN, INTEGER
          dimension of working array RWORK.
          The value of LRWORK must be at least LRWMIN, where
          LRWMIN  = max(L,N) + M  if  UTRJ = 'R'.
          LRWMIN >= max(L,N), otherwise.
WITHD   : IN, LOGICAL
          specifies whether or not D is a non-zero matrix to be
```

```
          included in the model:
          WITHD = .TRUE.,  D is a non-zero matrix and is used;
          WITHD = .FALSE., D is a zero matrix.
 UNITE :  IN, CHARACTER*1
          specifies whether or not E is the identity matrix:
          UNITE = 'I',  E is the identity matrix of order n;
                        In this case MW should be equal to  N.
          UNITE = 'G',  E is a general matrix.
 UNITF :  IN, CHARACTER*1
          specifies whether or not F is the identity matrix:
          UNITF = 'I',  F is the identity matrix of order l;
                        In this case LV should be equal to  L.
          UNITF = 'G',  F is a general matrix.
 UTRJ  :  IN, CHARACTER*1
          specifies the desired time trajectory for the input U:
          UTRJ = 'G',   U is a given time trajectory;
          UTRJ = 'I',   U is an impulse signal applied on the first k;
          UTRJ = 'S',   U is a step signal;
          UTRJ = 'R',   U is a random m-dimensional deviate.
 WTRJ  :  IN, CHARACTER*1
          specifies the desired time trajectory for the state
          disturbance W:
          WTRJ = 'G',   W is a given time trajectory;
          WTRJ = 'I',   W is an impulse signal applied on the first k;
          WTRJ = 'S',   W is a step signal;
          WTRJ = 'R',   W is a random mw-dimensional deviate.
 VTRJ  :  IN, CHARACTER*1
          specifies the desired time trajectory for the output
          disturbance V:
          VTRJ = 'G',   V is a given time trajectory;
          VTRJ = 'I',   V is an impulse signal applied on the first k;
          VTRJ = 'S',   V is a step signal;
          VTRJ = 'R',   V is a random lv-dimensional deviate.
 IDIST :  IN, INTEGER.
          if either  UTRJ = 'R',  or  WTRJ = 'R',  or  VTRJ = 'R',
          IDIST  specifies the distribution of the random numbers:
          IDIST = 1,  uniform (0,1);
          IDIST = 2,  uniform (-1,1);
          IDIST = 3,  normal (0,1).
          Otherwise  IDIST  is not referenced.
 *     :  RETURN 1, target label in case of error (e.g. *1111)
```

File input/ output:

none

Method:

For the specified values of the input vector u, state disturbance

vector w and output disturbance vector v, the output of the system is computed step by step, using the given system model:

$$x(k+1) = Ax(k) + Bu(k) + Ew(k)$$
$$y(k) = Cx(k) + Du(k) + Fv(k).$$

For WITHD = .FALSE., the matrix D is taken as zero.

Copyright:

Life cycle:

1994 MARCH V. Sima, Institut für Informatik, Bukarest: coded

Libraries required:

RASP, BLAS (1,2,3), LAPACK

Example:

none

Error Messages:

-1-
Invalid parameter value on entry.

Appendix 1. RASP-IDENT Driver Routines

Tools for State Intersection Class of Subspace Model Identification

RPIMN4 computes, by using the basic N4SID algorithm, consistent and statistically efficient estimates of the matrices (A,B,C,D) of a state space model when the output is additively perturbed by a zero-mean white noise independent from the input, and the disturbance input is generated by an innovation model. Optionally, also calculates the noise covariances and the Kalman gain to allow the design of a state observer. RASP interface to the subroutine IMN4S.

Tools for MOESP Class of Subspace Model Identification

RPIMOE computes consistent and statistically efficient estimates of the matrices (A,B,C,D) of a state space model when the output is additively perturbed by a zero-mean white noise independent from the input. The ordinary MOESP scheme is used. RASP interface to the subroutine IMOESO.

RPIMPI computes consistent estimates of the matrices (A,B,C,D) of a state space model when the output is additively perturbed by a zero-mean but of arbitrary statistical color. The MOESP scheme extended with instrumental variables based on past input quantities is used. RASP interface to the subroutine IMOEPI.

RPIMPO computes consistent and statistically efficient estimates of the matrices (A,B,C,D) of a state space model when the output is additively perturbed by a zero-mean white noise independent from the input, and the disturbance input is generated by an innovation model. The MOESP scheme extended with instrumental variables based on past input and output quantities is used. RASP interface to the subroutine IMOEPO.

RPIMRS computes consistent estimates of the matrices (A,B,C,D) of a state space model when the output is additively perturbed by a zero-mean, but of arbitrary statistical color. The MOESP scheme extended with instrumental variables based on reconstructed state and/or past input quantities is used. RASP interface to the subroutine IMOERS.

RPIMKG in addition to the RPIMPO routine, also calculates the noise covariances and the corresponding Kalman gain to allow the design of a state observer. RASP interface to the subroutine IMOEKG.

Auxiliary Tools for System Identification

IMMPID estimates a finite set of Markov parameters from input-output data (no constraints on input sequence to be white noise and the initial conditions to be zero).

IMSHFT shifts the input-output data sequences to compensate the dead-times in the components of the input vector.

IMICID estimates the initial state $x(0)$ of a linear time-invariant discrete-time system, given the matrix quadruple (A, B, C, D) and the input and output trajectories of the system.

IMCLID computes the parameters of a linear time-invariant (LTI) system operating in closed-loop with a LTI controller.

RPIMIU computes the output trajectory of a linear time-invariant discrete-time system, given the input trajectory and/or the initial state and output disturbances (or noise). RASP-IDENT interface to the subroutine IMOUTP.

Appendix 2. SLICOT Compatible Driver Routines

IMCLID computes the parameters of a linear time-invariant system operating in closed-loop with a linear time-invariant controller.

IMMPID estimates a finite set of Markov parameters from input-output data (no constraints on input sequence to be white noise and the initial conditions to be zero).

IMN4S determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices, the noise covariance matrices and the Kalman gain matrix using the basic N4SID algorithm.

IMOEKG determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices, the noise covariance matrices and the Kalman gain matrix using the MOESP algorithm with past inputs and outputs (PO) combined with the N4SID approach.

IMOEPI determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices using the "ordinary MOESP algorithm with past inputs".

IMOEPO determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices using the "ordinary MOESP algorithm with past inputs and outputs".

IMOERS determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices using the MOESP algorithm with reconstructed state variables as instrumental variables.

IMOESO determines the order of a linear time-invariant discrete-time state space model and estimates the quadruple of system matrices using the "ordinary MOESP algorithm".

IMOUTP computes the output trajectory of a linear time-invariant discrete-time system, given the input trajectory and/or the initial state and output disturbances (or noise).

IMSHFT shifts the input and output data so that a linear time-invariant system with dead-time zero is obtained. It allows the different input quantities to have different delays.

Appendix 3. SLICOT Compatible Computational and Auxiliary Routines

DGELSR computes the rank-revealing QR factorization of a real general matrix A, which may be rank-deficient, and applies the left transformations to another matrix B.

DGELSZ uses the rank-revealing QR factorization of a real general matrix A, computed by DGELSR, to determine the minimum-norm solution to a real linear least squares problem: minimize || A * X - B ||.

DORQRC overwrites the real (n+m)*1 matrix $C = \begin{pmatrix} A \\ B \end{pmatrix}$, where A and B are real n*1 and m*1 matrices, respectively, with Q'*C, where Q is a real orthogonal matrix defined as the product of elementary Householder reflectors as returned by DTRQRC.

DORQRT overwrites the real matrix C with Q' * C, or Q * C, where Q is a real orthogonal matrix defined as the product of elementary Householder reflectors as returned by DTRQRC.

DTRQRC computes a QR factorization, C = Q * R, of the real (n+m)*n matrix $C = \begin{pmatrix} A \\ B \end{pmatrix}$, where A is an n*n upper triangular matrix and B is an m*n matrix.

DTRQRT computes a QR factorization of a real (m+n) by n matrix A (A = Q * R), whose trailing n by n submatrix is upper triangular.

DTRSVB returns part of the singular value decomposition of a real upper triangular matrix and optionally applies the left transformations to another matrix.

EXTEND constructs the Hamiltonian or symplectic matrix associated to the linear-quadratic optimization problem, used to solve a continuous- or discrete-time algebraic Riccati equation, respectively.

IMABCD estimates the quadruple of system matrices A, B, C, and D of a linear time-invariant state space model using the singular value decomposition information provided by other routines.

IMADN4 estimates the quadruple of system matrices A, B, C, and D, and, optionally, the noise covariance matrices of the linear time-invariant state space model using the N4SID approach.

IMBDN4 estimates the system matrices B and D of the linear time-invariant state space model for the N4SID approach.
(auxiliary routine called only by IMADN4 for efficient use of the memory space)

IMCORD asks for user's confirmation of the system order found by other subroutines. This reverse communication routine could be modified, but its interface must be preserved.

IMKPRE determines the Kalman gain (predictor) matrix of a linear time-invariant discrete-time state space model using the system and noise covariance matrices.

IMPARM estimates the Markov parameters of a linear time-invariant system using the singular value decomposition information provided by other routines.
(auxiliary routine called only by IMMPID)

IMTRQ computes the triangular QR factor of a structured matrix.
(auxiliary routine called only by IMABCD for efficient use of the memory space)

IMTRQN computes the triangular QR factor of a structured matrix.
(auxiliary routine called only by IMBDN4 for efficient use of the memory space)

OPTREG computes the state feedback gain matrix of continuous- or discrete-time optimal regulator, based on the solution of an algebraic Riccati equation.

RICSCH computes the solution matrix of continuous- or discrete-time algebraic Riccati equation, or an orthogonal basis for the stable or unstable invariant subspace, using the Schur vectors approach.
The Hamiltonian or symplectic matrix associated to the linear-quadratic optimization problem is assumed to be given.

Appendix 4. Called RASP-MODRED, LAPACK and BLAS Routines

RASP-MODRED Routines

DGEES1, DMNEG, DMTRA, DTRSVD, RCFI, RPARPR, SEOR1, SPLITB

Directly called LAPACK Routines

DBDSQR, DGEBRD, DGECON, DGEQPF, DGEQRF, DGESV, DGETRF, DGETRI, DGETRS, DORGBR, DORM2R, DORMBR, DORMQR, DPOCON, DPOTRF, DPOTRS, DTRCON, DTRTRI, DTRTRS, DTZRQF

DLABAD, DLACPY, DLAIC1, DLANGE, DLANSY, DLANTR, DLARF, DLARFG, DLARNV, DLASCL, DLASET, DLATZM

DLAMCH, ILAENV, LSAME

Directly called BLAS 1 Routines

DAXPY, DCOPY, DDOT, DNRM2, DSCAL, DSWAP

Directly called BLAS 2 Routines

DGEMV, DGER, DTRMV, XERBLA

Directly called BLAS 3 Routines

DGEMM, DSYMM, DSYRK, DTRSM

RASP Error Processing Routines

RPERRB, RPERRI, RPERRN, RPERRP, RPERRT, RPERRV, RPINIT, RPQSLE