International Finance Discussion Papers

Number 275

February 1986

A METHOD FOR SOLVING SYSTEMS OF FIRST ORDER LINEAR HOMOGENEOUS DIFFERENTIAL EQUATIONS WHEN THE ELEMENTS OF THE FORCING VECTOR ARE MODELLED AS STEP FUNCTIONS

by

Robert A. Johnson

Abstract

This paper presents a method for solving a system of first order linear differential equations with constant coefficients when the elements of the forcing vector are step functions. The analysis presented in the text has been programmed for use in the computer simulation of linear continous time rational expectations models using any combination of anticipated and unanticipated, permanent or temporary shocks. The program entitled "JAB" is available from the author upon request.

A Method for Solving Systems of First Order Linear Homogeneous Differential Equations When the Elements of the Forcing Vector are Modelled as Step Functions

by

Robert A. Johnson*

This paper presents analytical expressions for the solution of a system of first order linear differential equations with constant coefficients when the elements of the vector of exogenous forcing variables are restricted to being modelled as step functions. The solution of such a system can then be programmed quite easily for use on the computer. As a result, simulation of deterministic continuous time rational expectations models can be performed under a wide variety of circumstances including any combination of anticipated or unanticipated shocks, permanent or transitory shocks and future or present shocks.

The first section below develops the notation and presents the discussion of the basic solution of first order linear differential equation systems with constant coefficients. This section constitutes the background for the focus of this paper and is essentially a review of the material presented in Buiter (1984). Section three begins with the general solution form developed in the previous section and derives the specific solution when the elements of the forcing vector are restricted to step functions. A brief conclusion is followed by an appendix exhibiting the computer program which incorporates the analysis contained in the body of the paper.

---

* The author is grateful to seminar participants at the Board of Governors of the Federal Reserve and to William Buiter for helpful comments. I would also like to thank Ruby Brooks for her patience and skillful typing of this paper. This paper represents the views of the author and should not be interpreted as reflecting the views of the Board of Governors of the Federal Reserve System or other members of its staff.

SECTION II. Background

In this section we will develop the notation and the basic solution method that will be used below.

We begin with the linear first order differential system,

$$(1) \quad \begin{bmatrix} \dot{x}(t) \\ E\dot{y}(t) \end{bmatrix} = A \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} + Bz(t).$$

where:

$x(t)$	is an N_1 vector of predetermined variables
$y(t)$	is an $N-N_1$ vector of nonpredeterminted variables
$z(t)$	is a k vector of exogenous variables
A	is a constant matrix of dimension N*N
B	is a constant matrix of dimension N*K
E	denotes the expectations operator
"."	denotes the derivative with respect to time d/dt

I will assume that the A matrix can be diagonalized as follows.

$$(2) \quad A = V^{-1}\Lambda V \text{ or } VAV^{-1} = \Lambda .$$

where:

V	is the matrix of left eigenvectors of A of dimension N*N.
Λ	is a diagonal matrix whose elements are the eigenvalues of matrix A.

I will further assume that there are N_1 eigenvalues with nonpositive real components and $N-N_1$ eigenvalues with positive real components. Following the exposition of Buiter (1984) I will partition the A,B,V, V^{-1} and Λ matrices to conform to the dimensions of x(t) and y(t).

$$(3) \quad A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} ;$$

$$V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \quad V^{-1} = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \quad \Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} .$$

where A_{11}, V_{11}, and W_{11} are $N_1 * N_1$ matrices.

A_{12}, V_{12}, and W_{12} are $N_1 * (N-N_1)$ matrices.

A_{21}, V_{21} and W_{21} are $(N-N_1) * N_1$, matrices.

A_{22}, V_{22} and W_{22} are $(N-N_1) * (N-N_1)$ matrices.

Λ_1, is an $N_1 * N_1$ matrix containing the stable roots of A.

Λ_2 is an $(N-N_1) * (N-N_1)$ matrix containing the unstable roots of A.

B_1 is an $N_1 * K$ constant matrix.

B_2 is an $(N-N_1) * K$ constant matrix.

Let variables p and q be defined as follows.

(4) $\binom{p}{q} = V \binom{x}{y}$ or $\binom{x}{y} = V^{-1} \binom{p}{q}$.

p is an N_1 vector and q and $N-N_1$ vector.

Substituting expression (2) for A, and taking the expectation of (1) yields,

$$(5) \quad \begin{bmatrix} E_t \dot{x}(t) \\ E_t \dot{y}(t) \end{bmatrix} = V^{-1} \Lambda V \begin{bmatrix} E_t x(t) \\ E_t y(t) \end{bmatrix} + BE_t z(t).$$

Premultiplying both sides of equation (5) by V using the definitions presented in (4) and partitioning the Λ, V and B matrices as in (3) one obtains.

$$(6) \quad \begin{bmatrix} E_t \dot{p}(t) \\ E_t \dot{q}(t) \end{bmatrix} = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} \begin{bmatrix} E_t & P(t) \\ E_t & q(t) \end{bmatrix} + \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} E_t z(t).$$

Let

(7) $D = V_{21}B_1 + V_{22}B_2$.

Then

$$(8)\quad E_t \dot{q}(t) = \Lambda_2 E_t q(t) + DE_t z(t).$$

The forward looking solution for equations (8) is

$$(9)\quad E_t q(s) = K_2 e^{\Lambda_2 s} - \int_s^\infty e^{\Lambda_2 (s-\tau)} DE_t z(\tau) d\tau, \; s \geq t.$$

where K_2 is an $N-N_1$ vector of arbitrary constants. Provided that he expectation of $z(t)$ is bounded over the interval $[t, +\infty)$, the second term of equation (9) will exist. But the expression in (9) will not converge unless $K_2=0$. Assuming that the elements of K_2 are all set equal to zero (9) becomes.

$$(10)\quad E_t q(s) = - \int_s^\infty e^{\Lambda_2 (s-\tau)} * D^* E_t\, z(\tau) d\tau.$$

From the definition of $q(t)$ given in (4) above.

$$(11)\quad q(t) = V_{21} * x(t) + V_{22} * y(t).$$

or

$$(12)\quad y(t) = V_{22}^{-1} * q(t) - V_{22}^{-1} * V_{21} * x(t).$$

Evaluating (10) at s=t using the weak consistency assumption that $E_t q(t) = q(t)$ and substituting into (12) one obtains

$$(13)\quad y(t) = - V_{22}^{-1} * V_{21} * x(t) - V_{22}^{-1} \int_t^\infty e^{\Lambda_2 (t-\tau)} * D^* E_t z(\tau) d\tau.$$

which is equation (14) from Buiter (1984).

Recall that by definition (3),

$$V^*W = I.$$

Thus,

$$V_{21} W_{11} + V_{22} W_{21} = 0.$$

Premultiplying by V_{22}^{-1} and postmultiplying by W_{11}^{-1} yields,

(14) $W_{21} W_{11}^{-1} = -V_{22}^{-1} V_{21}$.

Equation (13) can be expressed as

(13') $y(t) = W_{21}*W_{11}^{-1}x(t) - V_{22}^{-1} \int_t^\infty e^{\Lambda_2(t-\tau)}*D*E_t\, z(\tau)d\tau.$

From equation (1)

$\dot{x}(t) = A_{11}*x(t) + A_{12}*y(t) + B_1 z(t).$

Substitute (13) in for y(t) and one obtains

(15) $\dot{x}(t)=(A_{11}-A_{12}V_{22}^{-1}V_{21})x(t) + B_1 z(t)$

$-A_{12}V_{22}^{-1} \int_t^\infty e^{\Lambda_2(t-\tau)}DE_t z(\tau)d\tau.$

The compound term which multiplies x(t) can be simplified as follows. From (2),

$A = W\Lambda V.$ (where $W=V^{-1}$)

$A_{11}=W_{11}\Lambda_1 V_{11} + W_{12}\Lambda_2 V_{21}.$

$A_{12}=W_{11}\Lambda_1 V_{12} + W_{12}\Lambda_2 V_{22}$

$A_{11}-A_{12}V_{22}^{-1}V_{21}=W_{11}\Lambda_1 V_{11}+W_{12}\Lambda_2 V_{21} - W_{11}\Lambda_1 V_{12}V_{22}^{-1}V_{21} - W_{12}\Lambda_2 V_{21}.$

(16) $= W_{11}\Lambda_1[V_{11} - V_{12}V_{22}^{-1}V_{21}]$

Recall again that

$$WV = I.$$

$$W_{11}V_{11}+W_{12}V_{21} = I.$$

$$V_{11}+W_{11}^{-1}W_{12}V_{21} = W_{11}^{-1}.$$

and $$W_{11}V_{12} + W_{12}V_{22} = 0.$$

Premultiplying by W_{11}^{-1} and post multiplying by V_{22}^{-1} yields,

$$W_{11}^{-1} W_{12} = -V_{12} V_{22}^{-1}.$$

Putting this back into (17) creates

$$V_{11} - V_{12} V_{22}^{-1} V_{21} = W_{11}^{-1}.$$

and (16) becomes

$$(18) \quad A_{11}-A_{12}V_{22}^{-1}V_{21} = W_{11}\Lambda_1W_{11}^{-1}.$$

As a result (15) can be reexpressed as

$$(19) \quad \mathring{x}(t)=W_{11}\Lambda_1W_{11}^{-1}x(t_o)+B_1z(t)-A_{12}V_{22}^{-1}\int_t^\infty e^{\Lambda_2(t-\tau)}DE_tz(\tau)d\tau.$$

Given an N_1 vector of initial conditions for each element of the vector $x(t)$ at time $t=t_o$ one can express the solution to (19) as

$$(20)\quad x(t) = W_{11}e^{\Lambda_1(t-t_o)}W_{11}^{-1}\bar{x}(t_o) + \int_{to}^{t} W_{11}e^{\Lambda_1(t-s)}W_{11}^{-1} B_1 z(s)ds$$

$$- \int_{to}^{t} W_{11}e^{\Lambda_1(t-s)}W_{11}^{-1}A_{12}V_{22}^{-1} * \int_{s}^{\infty} e^{\Lambda_2(s-\tau)}DE_s z(\tau)d\tau ds.$$

The expression $W_{11}^{-1}A_{12}V_{22}^{-1}$ is then transformed using (2).

$$A_{12} = W_{11}\Lambda_1 V_{12} + W_{12}\Lambda_2 V_{22}.$$

Premultiplication by W_{11}^{-1} and postmultiplication by V_{22}^{-1} yields,

$$(21)\quad W_{11}^{-1} A_{12}V_{22}^{-1} = \Lambda_1 V_{12}V_{22}^{-1} + W_{11}^{-1}W_{12}\Lambda_2.$$

and the solution for x(t) becomes

$$(20)'\quad x(t) = W_{11}e^{\Lambda_1(t-t_o)}W_{11}^{-1}\bar{x}(t_o) + \int_{to}^{t} W_{11}e^{\Lambda_1(t-s)}W_{11}^{-1}B_1 z(s)ds$$

$$- \int_{to}^{t} W_{11}e^{\Lambda_1(t-s)} * [\Lambda_1 V_{12}V_{22}^{-1} + W_{11}^{-1}W_{12}\Lambda_2] * * \int_{s}^{\infty} e^{\Lambda_2(s-\tau)}DE_s z(\tau)d\tau ds.$$

These two equations, (20) and (20') are the expressions developed by Buiter (1984).

The solution to x(t), the predetermined variable, reflects three influences. First, the initial condition $x(t_o)$, the relative influence of which diminishes as t increases. The second influence, which is expressed in the second term of (20) or (20') above, is that exerted by the elements of the forcing vector from the initial time, t_o until time t. Finally the third influence can be expressed as, what was believed about the future course of the forcing elements z(t) at each point in the past from t_o to t. Let us turn now to the evolution of the solution components when the elements of the forcing vector z are modelled as step functions.

Section III

We can now examine the components of the solution for $x(t)$ and $y(t)$. First the solution for $x(t)$ is developed and then the solution to $y(t)$ is analyzed.

To develop a solution that is analytically tractable and that can be implemented on computer I will limit the elements of the forcing vector to change only in steps.

$Z=Z_o$ for $t \leqq to$,

$Z=Z_1$ for $t_o < t \leqq t_1$,

$Z=Z_2$ for $t_1 < t \leqq t_2$,

$Z=Z_i$ for $t_{i-1} < t \leqq t_i$,

$Z=Z\infty$ for $t_n < t \leqq \infty$.

Where n= the number of columns in the Z matrix minus one.

The solution to $x(t)$ has three components. The first, representing the n_1 vector of initial conditions, can be evaluated without manipulation. The second term represents the influence of the forcing vector from the initial time t_o, to time to the present.

$$\int_{to}^{t} W_{11} e^{\Lambda_1 (t-s)} W_{11}^{-1} B_1 z(s) ds.$$

If the forcing vector is constant between sucessive points as assumed above then this integral becomes

$$(21) \sum_{i=1}^{J} -W_{11} \Lambda_1^{-1} [e^{\Lambda_1 (t-t_i)} - e^{\Lambda_1 (t-t_{i-1})}] z_i \; +$$

$$-W_{11} \Lambda_1^{-1} [1-e^{\Lambda_1 (t-t_J)}] z_{J+1}.$$

Where J is the number of transitions of the forcing vector z from time t_o to time t.

One then has to compute the third term of the solution,

$$\int_{to}^{t} W_{11} e^{\Lambda_1(t-s)} * M * \int_{s}^{\infty} e^{\Lambda_2(s-\tau)} * D\ E_s z(\tau) d\tau ds.$$

where $M = \Lambda_1 V_{12} V_{22}^{-1} + W_{11}^{-1} W_{12} \Lambda_2.$

First of all one analyses the inside integral,

$$(22) \quad f(s) = \int_{s}^{\infty} e^{\Lambda_2(s-\tau)} * DE_s z(\tau) d\tau.$$

The relationship between s and the transition times tt_i is crucial in the development of this expression. Let us call tt_s the first transition time after s. Thus our integral becomes.

$$f(s) = \int_{s}^{tts} e^{\Lambda_2(s-\tau)} * DE_s z(\tau) d\tau$$

$$+ \sum_{i=o}^{\infty} \int_{tts+i}^{tts+i+1} e^{\Lambda_2(s-\tau)} * DE_s\ z(\tau) d\tau.$$

When evaluated this becomes

$$(23) \quad f(s) = \Lambda_2^{-1} D\ Z_{tts} + \sum_{i=o}^{IMAX} e^{\Lambda_2(s-tt_{s+i})} (z_{tts+i+1} - z_{tts+i}).$$

where IMAX = number of transitions of the z vector after point s. Note that for s greater than the last transition time one obtains

$$f(s) = \int_S^\infty e^{\Lambda_2(s-\tau)} \cdot DE_s(Z(\tau))d\tau.$$

$$= \Lambda_2^{-1} Dz_\infty$$

Now we must evaluate the entire expression

$$(24) \quad \int_{to}^{t} W_{11} e^{\Lambda_1(t-s)} * M*f(s)ds.$$

The difficulty arises because the function f(s) is continuous across transition times but it is nondifferentiable at the transition times.

For instance at the first transition time, tt_1 for ε small

$$f(s)\Big|_{t+tt_{1-\varepsilon}} = \Lambda_2^{-1}DZ_1 + \Lambda_2^{-1}e^{\Lambda_2(s-tt_1)} * D(Z_2-Z_1)$$

$$+ \Lambda_2^{-1} e^{\Lambda_2(s-tt_2)} *D(Z_3-Z_2) + \ldots$$

$$f(s)\Big|_{t=tt_1+\varepsilon} = \Lambda_2^{-1}DZ_2 + \Lambda_2^{-1}e^{\Lambda_2(s-tt_2)} *D[Z_3-Z_2] + \ldots$$

As one takes the limit as $\varepsilon \to 0$, $f(s)_- = f(s)_+$.

The function f(s) is continuous. But differentiating the two expressions and evaluating as $\varepsilon \to 0$,

$$\frac{\partial f(s)}{\partial s}_- - \frac{\partial f(s)}{\partial s}_+ = D * (z_2-z_1)\Big|_{t=tt_1}.$$

Which is only zero (i.e. the expression is differentiable) if $z_2 = z_1$, in other words, if this is not a transition point.

Thus, the expression (24) must be broken up into a number of integrals.

$$(25) \quad \sum_{i=o}^{nt} \int_{t_i}^{t_{i+1}} W_{11} e^{\Lambda_1 (t-s)} * M * f(s)_i^{i+1} ds.$$

Where $f(s)_i^{i+1}$ is the appropriate function between t_i and t_{i+1},

$$t_{nt+1} = t.$$

and nt = the number of transitions of the z vector up to the current time t.

Consider a representative time t which is between tt_j and tt_{j+1}. One obtains,

$$\int_{to}^{t1} W_{11} e^{\Lambda_1 (t-s)} Mf(s)_o^1 ds + \int_{t1}^{t2} W_{11} \; e^{\Lambda_1 (t-s)} Mf(s)_1^2 ds$$

$$\ldots\ldots\ldots + \int_{t_j}^{t} W_{11} e^{\Lambda_1 (t-s)} f(s)_J^{J+1} ds.$$

Referring back to equation (23), we can recognize that a given point t between t_j and t_{j+1} all terms of the form,

$$\Lambda_2^{-1} De^{\Lambda_2 (s-t_{K+1})} * D[z_{K+2} - z_{K+1}],$$

for $J \leq K \leq \infty$ will be common to all subfunctions $f(s)_i^{i+1}$ between t_o and t. One can then recombine terms and evaluate the expression

$$(26) \quad \int_{to}^{t} W_{11} e^{\Lambda_1 (t-s)} * M * \Lambda_2^{-1} * e^{\Lambda_2 (s-t_{j+i+1})} D[z_{J+2+i} - z_{J+1+i}].$$

Thus

$$\int_{t_o}^{tt_K} W_{11} e^{\Lambda_1 (t-s)} * M * \Lambda_2^{-1} e^{\Lambda_2 (s-tk)} * D[z_{K+1} - z_K]$$

becomes

$$W_{11} \psi_K D[z_{K+1} - z_K],$$

where ψ_K is an $N_1 *(N-N_1)$ matrix whose element δ, σ equals,

$$\Psi_{i,\delta,\sigma} = \theta_{\delta,\sigma} * [e^{\Lambda_1^{\delta}(t-t_k)} - e^{\Lambda_1^{\delta}(t-t_o)} * e^{\Lambda_2^{\sigma}(t_o-t_k)}].$$

Now for every K from 1 to J there will be an expression of this type. Thus the contribution of these past terms can be expressed as

$$(28) \quad \sum_{K=1}^{J} W_{11} \psi_K D[z_{K+1} - z_K].$$

The final set of terms are derived from the elements of

$f(s)_i^{i=1}$ which are of the form $\Lambda_2^{-1} Dz_i$ and they evaluate to expressions of the form,

$$(29) \quad \sum_{i=o}^{J} - W_{11} [\Lambda_2]^{-1} * M * [e^{\Lambda_1 (t-t_i)} - e^{\Lambda_1 (t-t_i)} * Dz_{i+1}].$$

and finally between t_j and t one must evaluate the component of $f(s)_J^{J+1}$

of the form $\Lambda_2^{-1} Dz_{J+1}$ which integrates to become

$$(30) \quad W_{11} [\Lambda_2]^{-1} * [\Lambda_1]^{-1} * M * [1 - e^{\Lambda_1 (t-t_j)}] * D * z_{J+1}.$$

To solve this integral one forms a matrix θ whose element in row δ column σ is equal to

$$\theta_{\delta,\sigma} = [\Lambda_2^{\sigma} - \Lambda_1^{\delta}]^{-1} * M_{\delta,\sigma} * [\Lambda_2^{\sigma}]^{-1}$$

where $1 \leqq \delta \leqq N_1$,

and $1 \leqq \sigma \leqq N-N_1$.

The $N_1 * (N-N_1)$ matrix θ is then used to formulate another $N_1 * (N-N_1)$ matrix B_i, where the subscript i denotes the number of forcing element transitions beyond the first transition occuring after the current time t.

$$B_{i,\delta,\sigma} = \theta_{\delta,\sigma} * [e^{\Lambda_2^{\sigma}(t-tt_{J+i+1})} - e^{\Lambda_1^{\delta}(t-t_o)} * e^{\Lambda_2^{\sigma}(t_o-tt_{J+i+1})}].$$

The solution to (26) then becomes

$$(27) \quad \sum_{L=0}^{NZ-nt-1} W_{11} B_i D^* [z_{J+2+i} - z_{J+1+i}]$$

A similar technique is employed for all terms $f(s)_i^{i+1}$ from 0 to K

where K in turn runs from 0 to J, the number of the transition time preceding the current time t

For all $f(s)_i^{i+i}$, i=0,K the term

$$\Lambda_2^{-1} e_2^{\Lambda} {}^{(s-t_k)} D[z_{K+1} - z_K]$$

will be a common element.

The solution for x(t) in total is

$$(31)\quad x(t) = W_{11}e^{\Lambda_1(t-to)}W_{11}^{-1}x(t_o) - [(27) + (28) + (29) +$$

$$+ (30)] + (21).$$

One can then use x(t) in equation (13') and that leaves only the second term in that expression to evaluate

$$(13')\quad y(t) = W_{21}*W_{11}^{-1} * x(t)-V_{22}^{-1}* \int_t^\infty e^{\Lambda_2(t-\tau)}* D*E_t z(\tau)d\tau.$$

The second term reflects the expectation of future forcing vectors as of time t presuming once again that $E_t z(\tau) = z(\tau)$, (i.e. correct expectations), one can evaluate

$$\int_t^\infty e^{\Lambda_2(t-\tau)} * D*E_t z(\tau)d\tau.$$

At a time t between t_J and t_{J+1} one obtains,

$$(32)\quad \Lambda_2^{-1}*D*z_{J+1}+ \sum_{i=1}^{nt} \Lambda_2^{-1} *[e^{\Lambda_2(t-t_{J+i+1})}-e^{\Lambda_2(t-t_{J+i})}]*$$

$$D*[z_{J+i+1} - z_{J+i}]$$

Where nt is the number of transitions of the forcing vector between t and ∞ . Equation (32) and the first term of equation (13'), which can be computed using the results obtained in equation (31), are combined to solve for y(t).

Conclusion:

The analysis in the preceding sections can be readily implemented on the computer to permit one to conduct policy simulation experiments which illustrate the effects of any combination of anticipated or unanticipated, permanent or transitory changes in exogenous variables. The program called "JAB" written by Johnson, Austin and Buiter, which is the offspring of "Saddlepoint", has recently been completed and is available from the author. A copy of the program follows as an Appendix.

References

Austin, G. P., and Willem H. Buiter: "Saddlepoint" A Programme for Solving Continuous Time Linear Rational Expectations Models", University of Bristol Discussion Papaer No. 82/132, November, 1982.

Buiter, Willem H., "Saddlepoint Problems in Continuous Time Rational Expectations Models: A General Method and Some Macroeconomic Examples," Econometrica, Vol. 52 (May 84), pp. 665-80.

Appendix

Presented here is the program "JAB" which implements the results of the paper. This program is an outgrowth of "Saddlepoint"; see Austin and Buiter (1982).

```
C $SET LINEINFO
C $RESET FREE LIST
C $SET AUTOBIND
C $BIND=FROM *NAGF/=;
C FILE  19(KIND=DISK,TITLE='INFILE',FILETYPE=7)
C FILE  18(KIND=DISK,TITLE='OUTFILE',PROTECTION=SAVE)
C FILE  99(KIND=REMOTE,MYUSE=OUT)
C FILE  98(KIND=REMOTE,MYUSE=IN)
C
C       THIS IS THE 'JAB' PROGRAM WRITTEN BY G.P. AUSTIN, W. BUITER
C       AND ROBERT JOHNSON. ONE MUST HAVE THE N.A.G. LIBRARY TO BE ABLE
C       TO IMPLEMENT THIS PROGRAM.
C
C
        IMPLICIT REAL*8(A-H,O-Z)
        REAL*8 E1(24,24),E2(24,24),E3(24,24),E4(24,24),E5(24,24)
     1,E6(24,24),E7(24,24),E8(24,24),F1(24,24),F2(24,24),F3(24,24),F4(24
     1,24),B(24,24),Z(24,24),WKSPCE(24),F01DAF,X,Y,ZZ,T,CR,CI,XEND
         COMPLEX*16 C1(24,24),C2(24,24)
     1 ,C3(24,24),C4(24,24),C5(24,24),C6(24,24),C7(24,24),C8(
     124,24),C9(24,24),C10(24,24),C11(24,24),C12(24,24),C13(24
     1,24),C14(24,24),C15(24,24),C16(24,24),CC(24,24),CX
         INTEGER INTGER(24),N1,N2,N3,NZ,NSR,NUSR,NIC,IA
C        CHARACTER*128 INFILE,OUTFILE
C        PRINT, 'WHAT IS YOUR DATA FILE CALLED'
C        READ(98,*)INFILE
C        OPEN(19,FILE=INFILE,FORM='FORMATTED')
C        PRINT,'WHERE SHOULD I WRITE THE OUTPUT'
C        READ(98,*)OUTFILE
C        OPEN (18,FILE=OUTFILE,FORM='FORMATTED')
        READ(19,*)N1,N2,N3,NZ
        IA=MAX0(N1,N2,N3,NZ)
C
C       READ SIZE OF MODEL AND PASS LARGEST DIMENSION TO ALL SUBROUTINES
C
        CALL SUBS(E1,E2,E3,E4,E5,E6,E7,E8,F1,F2,F3,F4,B,Z,WKSPCE,C1,C2,C3,
     1C4,C5,C6,C7,
     1 C8,C9,C10,C11,C12,C13,C14,C15,C16,CC,CX,INTGER,N1,N2,N3,NZ,IA)
        RETURN
        END
        SUBROUTINE SUBS(E1,E2,E3,E4,E5,E6,E7,E8,F1,F2,F3,F4,B,Z,WKSPCE,C1,
     1C2,C3,C4,C5,C6,
     1 C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,CC,CX,INTGER,N1,N2,N3,NZ,IA)
        IMPLICIT REAL*8(A-H,O-Z)
        REAL*8 E1(IA,IA),E2(IA,IA),E3(IA,IA),E4(IA,IA),E5(IA,IA)
     1,E6(IA,IA),E7(IA,IA),
     1 E8(IA,IA),F1(IA,IA),F2(IA,IA),F3(IA,IA),F4(IA,IA),B(IA,IA),Z(IA,I
     1A),WKSPCE(IA)
        COMPLEX*16 C1(IA,IA),C2(IA,IA),C3(IA,IA),C4(IA,IA),C5(IA,IA),C6(
     1IA,IA),
     1 C7(IA,IA),C8(IA,IA),C9(IA,IA),C10(IA,IA),C11(IA,IA),
     1 C12(IA,IA),C13(IA,IA),C14(IA,IA),C15(IA,IA),C16(IA,IA),
     1 CC(IA,IA),CX
        REAL*8 BLANK /'      '/
        INTEGER IA,N1,N2,N3,NZ,NUSR,NIC,IFLAG,KSTR,INTGER
        DIMENSION INTGER(IA)
99995   FORMAT(1X,A6)
200     FORMAT(/'        SOLN. TO MATRIX A',/)
210     FORMAT(/'        SOLN. TO MATRIX B',/)
```

```
220    FORMAT(/'      SOLN. TO MATRIX C',/)
230    FORMAT(/'      SOLN. TO MATRIX D',/)
240    FORMAT(/'     REAL PART OF THE EIGENVALUES OF MATRIX A',/)
250    FORMAT(/'     IMAGINARY PARTS OF THE EIGENVALUES OF MATRIX A',/)
260    FORMAT('    REAL PARTS OF THE EIGENVECTORS OF MATRIX A (BY COLUMN)
      1'/)
270    FORMAT(/'      IMAGINARY PARTS OF THE EIGENVECTORS OF MATRIX A'/)
  280  FORMAT(/'       SOLN. TO XBAR',/)
290    FORMAT(/'      SOLN. TO XBARBAR',/)
300    FORMAT(/'      SOLN. TO YBAR',/)
310    FORMAT(/'      SOLN. TO YBARBAR',/)
320    FORMAT('    A PROGRAM FOR SOLVING FOR THE BEHAVIOUR OF LINEAR ',/'
      1    RATIONAL EXPECTATION MODELS ON THE STABLE MANIFOLD',//)
330    FORMAT('    DATA SUPPLIED ....E1,E2,E3,E4,E5,E6,E7,E8, EXOGENOUS Z
      1 VECTORS',///)
100    FORMAT(1X,16(F13.6))
199    FORMAT(1X,16(F10.6))
       DO 54 I=1,IA
       CC(I,I)= DCMPLX(1.0D0,0.0D0)
54     B(I,I)=1.0
C
C       SET ELEMENTS OF CC AND B FOR LATER INVERSION ROUTINES
C
       READ(19,*)((E1(I,J),J=1,N1),I=1,N1)
       READ(19,*)((E2(I,J),J=1,N1),I=1,N1)
       READ(19,*)((E3(I,J),J=1,N2),I=1,N1)
       READ(19,*)((E4(I,J),J=1,N3),I=1,N1)
       READ(19,*)((E5(I,J),J=1,N1),I=1,N2)
       READ(19,*)((E6(I,J),J=1,N1),I=1,N2)
       READ(19,*)((E7(I,J),J=1,N2),I=1,N2)
       READ(19,*)((E8(I,J),J=1,N3),I=1,N2)
       READ(19,*)((Z(I,J),J=1,NZ),I=1,N3)
       M=1
       IFAIL=0
C
C      SET ALL NAG ROUTINES TO TAKE IFAIL PARAMETER AS ZERO
C
       WRITE(18,320)
       WRITE(18,99995) BLANK
       WRITE(18,330)
       WRITE(18,99995) BLANK
       DO 81 I=1,N1
       WRITE(18,100)(E1(I,J),J=1,N1)
81     CONTINUE
       WRITE(18,99995) BLANK
       DO 82 I=1,N1
       WRITE(18,100)(E2(I,J),J=1,N1)
82     CONTINUE
       WRITE(18,99995) BLANK
       DO 83 I=1,N1
       WRITE(18,100)(E3(I,J),J=1,N2)
83     CONTINUE
       WRITE(18,99995) BLANK
       DO 84 I=1,N1
       WRITE(18,100)(E4(I,J),J=1,N3)
84     CONTINUE
       WRITE(18,99995) BLANK
       DO 85 I=1,N2
       WRITE(18,100)(E5(I,J),J=1,N1)
85     CONTINUE
```

```
      WRITE(18,99995) BLANK
      DO 86 I=1,N2
      WRITE(18,100)(E6(I,J),J=1,N1)
86    CONTINUE
      WRITE(18,99995) BLANK
      DO 87 I=1,N2
      WRITE(18,100)(E7(I,J),J=1,N2)
87    CONTINUE
      WRITE(18,99995) BLANK
      DO 88 I=1,N2
      WRITE(18,100)(E8(I,J),J=1,N3)
88    CONTINUE
      WRITE(18,99995) BLANK
      DO 89 I=1,N3
      WRITE(18,100)(Z(I,J),J=1,NZ)
89    CONTINUE
C
C     ECHO WHAT WE READ IN AS DATA
C
      WRITE(18,99995) BLANK
      CALL F04AEF(E7,IA,B,IA,N2,N2,F1,IA,WKSPCE,F2,IA,F3,IA,IFAIL)
      IOPT=2
      CALL F01CKF(E1,E3,F1,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      IOPT=1
      CALL F01CKF(F2,F1,E6,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CKF(E7,E3,E6,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CEF(E2,E2,E7,IA,IA,IFAIL)
      CALL F04AEF(E2,IA,B,IA,N1,N1,F3,IA,WKSPCE,F4,IA,E6,IA,IFAIL)
      CALL F01CKF(F4,E3,E5,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CEF(E1,E1,F4,IA,IA,IFAIL)
      CALL F01CKF(E6,E3,E8,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CEF(E4,E4,E6,IA,IA,IFAIL)
      IOPT=3
      CALL F01CKF(E1,F1,E8,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CKF(E1,F1,E5,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      IOPT=1
      CALL F01CKF(E6,F2,F3,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CKF(E7,E6,E4,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CEF(E8,E7,E8,IA,IA,IFAIL)
      CALL F01CKF(E2,E6,E1,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CEF(E7,E2,E5,IA,IA,IFAIL)
      CALL F01CKF(E5,F3,E1,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CKF(E6,F3,E4,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      DO 65 I=1,IA
      DO 65 J=1,IA
      E5(I,J)=-E5(I,J)
      E6(I,J)=-E6(I,J)
65    CONTINUE
      CALL F04AEF(E5,IA,B,IA,N1,N1,F1,IA,WKSPCE,F2,IA,F3,IA,IFAIL)
      CALL F01CKF(F2,F1,E6,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CKF(F3,E7,F2,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CEF(F4,E8,F3,IA,IA,IFAIL)
C
      CALL F01CKF(E1,F2,Z,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      CALL F01CKF(E2,F4,Z,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
C
      WRITE(18,99995) BLANK
      DO 636 I=1,IA
      DO 636 J=1,IA
      E1(I,J)=-E1(I,J)
```

```
 636   CONTINUE
       WRITE(18,99995) BLANK
C
C      MANIPULATE THE E MATRICES TO GIVE SOLN. TO A,B,C,D
       WRITE(18,99995) BLANK
       WRITE(99,501)
  501  FORMAT(/' SOLN TO XBAR')
       WRITE(99,100)(E1(J,1),J=1,N1)
       WRITE(18,99995) BLANK
       WRITE(99,502)
  502  FORMAT(/' SOLN TO  XBARBAR')
       WRITE(99,100)(E1(J,NZ),J=1,N1)
       WRITE(18,280)
       WRITE(18,100)(E1(J,1),J=1,N1)
       WRITE(18,99995) BLANK
       WRITE(18,290)
       WRITE(18,100)(E1(J,NZ),J=1,N1)
       WRITE(18,99995) BLANK
       WRITE(18,300)
       WRITE(18,100)(E2(J,1),J=1,N2)
       WRITE(18,99995) BLANK
       WRITE(18,310)
       WRITE(18,100)(E2(J,NZ),J=1,N2)
       WRITE(18,99995) BLANK
C
       DO 637 I=1,IA
       DO 637 J=1,IA
       E1(I,J)=-E1(I,J)
 637   CONTINUE
C      WRITE SOLN. TO XBAR,XBARBAR,YBAR,YBARBAR
C
       WRITE(18,200)
       DO 55 I=1,N1
       WRITE(18,100)(E5(I,J),J=1,N1)
 55    CONTINUE
       WRITE(18,99995) BLANK
       WRITE(18,210)
       DO 56 I=1,N1
       WRITE(18,100)(E6(I,J),J=1,N3)
 56    CONTINUE
       WRITE(18,99995) BLANK
       WRITE(18,220)
       DO 57 I=1,N2
       WRITE(18,100)(E7(I,J),J=1,N1)
 57    CONTINUE
       WRITE(18,99995) BLANK
       WRITE(18,230)
       DO 58 I=1,N2
       WRITE(18,100)(E8(I,J),J=1,N3)
 58    CONTINUE
C
C      WRITE A,B,C,D TO OUTPUT FILE.
C
       DO 10 I=1,N1
       DO 10 J=1,N1
 10    C12(I,J)=DCMPLX(E5(I,J),0.0D0)
       WRITE(18,99995) BLANK
       CALL F02AGF(E5,IA,N1,E2,E3,E4,IA,F1,IA,INTGER,IFAIL)
C
C      TO FIND THE E-VALUES AND VECTORS OF MATRIX A
```

```
C
11    ZZ=0.0
      N11=N1-1
      DO 12 I=1,N11
      IF(E2(I,1).LE.E2(I+1,1)) GO TO 12
      ZZ=ZZ+1.
      X=E2(I,1)
      Y=E3(I,1)
      E2(I,1)=E2(I+1,1)
      E3(I,1)=E3(I+1,1)
      E2(I+1,1)=X
      E3(I+1,1)=Y
      DO 13 J=1,N1
      X=E4(J,I)
      Y=F1(J,I)
      E4(J,I)=E4(J,I+1)
      F1(J,I)=F1(J,I+1)
      E4(J,I+1)=X
      F1(J,I+1)=Y
13    CONTINUE
12    CONTINUE
      IF (ZZ.GT.0)GO TO 11
C
C     SORT THE E-VALUES INTO ASCENDING ORDER AND ARRANGE THE
C     E-VECTORS CONFORMALLY.
C
      WRITE(18,99995) BLANK
      WRITE(18,240)
      WRITE(18,100)(E2(I,1),I=1,N1)
      WRITE(18,99995) BLANK
      WRITE(18,250)
      WRITE(18,100)(E3(I,1),I=1,N1)
      WRITE(18,99995) BLANK
      WRITE(18,260)
      DO 59 I=1,N1
59    WRITE(18,100)(E4(I,J),J=1,N1)
      WRITE(18,99995) BLANK
      WRITE(18,270)
      DO 888 I=1,N1
888   WRITE(18,100)(F1(I,J),J=1,N1)
C
C     WRITE THE E-VECTORS AND E-VALUES TO THE OUTPUT FILE.
C
      WRITE(18,99995) BLANK
      NSR=0
      DO 14 I=1,N1
      IF (E2(I,1).LE. 0)NSR=NSR+1
14    CONTINUE
C
C     FIND THE NUMBER OF STABLE ROOTS.
C
      NUSR=N1-NSR
      DO 15 I=1,N1
      DO 15 J=1,N1
      X=E4(I,J)
      Y=F1(I,J)
      C1(I,J)= DCMPLX(X,Y)
      C2(I,J)= DCMPLX(X,Y)
15    C3(I,J)= DCMPLX(X,Y)
      DO 16 I=1,NSR
```

```
16      C4(I,I)= DCMPLX(E2(I,1),E3(I,1))
        DO 17 I=1,NUSR
17      C5(I,I)= DCMPLX(E2(I+NSR,1),E3(I+NSR,1))
        CALL F04ADF(C1,IA,CC,IA,NSR,NSR,C6,IA,WKSPCE,IFAIL)
        CALL F04ADF(C2,IA,CC,IA,N1,N1,C1,IA,WKSPCE,IFAIL)
        CX= DCMPLX(0.0D0,0.0D0)
        L=1
        N11=NSR+1
        DO 19 I=N11,N1
        K=I-NSR
        DO 19 J=1,NSR
        CALL F01DDF(L,NSR,CX,I,J,C3,IA,C6,IA,N,CR,CI)
19      E3(K,J)=-CR
        DO 20 I=1,N1
        DO 20 J=1,N3
20      C7(I,J)= DCMPLX(E6(I,J),0.0D0)
        DO 21 I=1,NSR
        DO 21 J=1,N3
        CALL F01DDF(L,NSR,CX,I,J,C6,IA,C7,IA,N,CR,CI)
21      C8(I,J)= DCMPLX(-CR,-CI)
        DO 22 I=N11,N1
        K=I-NSR
        DO 22 J=1,N3
        CALL F01DDF(L,N1,CX,I,J,C1,IA,C7,IA,N,CR,CI)
22      C11(K,J)= DCMPLX(-CR,-CI)
        DO 23 I=N11,N1
        K=I-NSR
        DO 23 J=N11,N1
        N=J-NSR
23      C7(K,N)=C1(I,J)
        CALL F04ADF(C7,IA,CC,IA,NUSR,NUSR,C9,IA,WKSPCE,IFAIL)
        DO 24 I=1,NSR
        DO 24 J=N11,N1
        K=J-NSR
        CALL F01DDF(L,NSR,CX,I,J,C6,IA,C12,IA,N,CR,CI)
24      C1(I,K)= DCMPLX(-CR,-CI)
        DO 25 I=1,NSR
        DO 25 J=1,NUSR
        CALL F01DDF(L,NUSR,CX,I,J,C1,IA,C9,IA,N,CR,CI)
25      C7(I,J)= DCMPLX(-CR,-CI)
        DO 26 I=1,NSR
        DO 886 J=1,NSR
886     C1(I,J)= DCMPLX(0.0D0,0.0D0)
        X=DREAL(C4(I,I))
        Y=DIMAG(C4(I,I))
        ZZ=X*X+Y*Y
        C1(I,I)= DCMPLX((X/ZZ),-(Y/ZZ))
     26 CONTINUE
        DO 27 I=1,NUSR
        DO 887 J=1,NUSR
887     C2(I,J)=(0.,0.)
        X=DREAL(C5(I,I))
        Y=DIMAG(C5(I,I))
        ZZ=X*X+Y*Y
        C2(I,I)= DCMPLX((X/ZZ),-(Y/ZZ))
     27 CONTINUE
        DO 28 I=1,NSR
        DO 28 J=1,NUSR
        CALL F01DDF(L,NUSR,CX,I,J,C7,IA,C2,IA,N,CR,CI)
28      C10(I,J)= DCMPLX(-CR,-CI)
```

```
C
C      MANIPULATE THE E-VECTOR MATRICES AND THE E-VALUE MATRICES TO GIVE
C      THE COEFFICIENTS IN EQUATIONS 12B AND 13.
C
       DO 776 I=1,N3
       DO 776 J=1,NZ
776    CC(I,J)= DCMPLX(Z(I,J),0.0D0)
       X=0.0
       WRITE(99,399)NSR
   399 FORMAT(/' THE NUMBER OF STABLE ROOTS IS ',I2)
    50 WRITE(99,999)
   999 FORMAT(/' HOW MANY INITIAL CONDITIONS?')
       READ(98,*)NIC
       M=NSR-NIC
       IF(M .GE. 0) GO TO 29
       WRITE(99,998)
998    FORMAT(/' MORE INITIAL CONDITIONS THAN STABLE ROOTS')
       GO TO 50
    29 WRITE(99,99991)
99991  FORMAT(/'  ENTER -1 TO MAKE XBAR-BAR THE INITIAL CONDITIONS'/
     *' OR 0 TO USE XBAR, ELSE 1 TO INPUT YOUR OWN : = ')
       READ(98,*) Y
       IF(Y) 777,778,779
   777 DO 41 I=1,NIC
    41 E1(I,1)=E1(I,NZ)-E1(I,1)
       GO TO 53
   778 DO 781 I=1,NIC
   781 E1(I,1)=-E1(I,1)
       GO TO 53
   779 WRITE(6,99992)
99992  FORMAT(/'  WHAT ARE YOUR INITIAL CONDITIONS := ?'/)
       READ(98,*) (E1(I,1),I=1,NIC)
    53 IF(M .EQ. 0) GO TO 52
    51 DO 885 I=1,IA
       DO 885 J=1,IA
       E2(I,J)=0.0
       E6(I,J)=0.0
       F1(I,J)=0.0
       F2(I,J)=0.0
       F3(I,J)=0.0
885    F4(I,J)=0.0
       N11=NIC+1
       DO 884 I=N11,NSR
884    E1(I,1)=0.0
       WRITE(99,997)
997    FORMAT(/' WHAT ARE THE VALUES OF F1(I,J) (BY ROW)?')
       READ(98,*)((F1(I,J),J=1,M),I=1,M)
       WRITE(99,996)
996    FORMAT(/' WHAT ARE THE VALUES OF F2(I,J) (BY ROW)?')
       READ(98,*)((F2(I,J),J=1,NIC),I=1,M)
       WRITE(99,995)
995    FORMAT(/' WHAT ARE THE VALUES OF F3(I,J) (BY ROW)?')
       READ(98,*)((F3(I,J),J=1,NUSR),I=1,M)
       WRITE(99,994)
994    FORMAT(/' WHAT ARE THE VALUES OF F(I,1) ?')
       READ(98,*)(F4(I,1),I=1,M)
       WRITE(18,88881)
88881  FORMAT(' COEFFICIENT MATRIX F1')
       WRITE(18,100)((F1(I,J),I=1,M),J=1,M)
       WRITE(18,99995) BLANK
```

```
      WRITE(18,88882)
88882 FORMAT( ' COEFFICIENT MATRIX F2')
      WRITE(18,100)((F2(I,J),I=1,M),J=1,NIC)
      WRITE(18,99995) BLANK
      WRITE(18,88883)
88883 FORMAT(     ' COEFFICIENT MATRIX F3' )
      WRITE(18,100)((F3(I,J),I=1,M),J=1,NUSR)
      WRITE(18,99995) BLANK
      WRITE(18,88884)
88884 FORMAT(     ' R.H.S. VECTOR F' )
      WRITE(18,100)(F4(I,1),I=1,M)
      WRITE(18,99995) BLANK
      CALL F04AEF(F1,IA,B,IA,M,M,E2,IA,WKSPCE,E4,IA,E5,IA,IFAIL)
      CALL F01CKF(E6,F3,E3,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
      DO 30 I=1,M
      NIC1 = NIC+1
      DO 30 J=NIC1,NSR
      K=J-NIC
30    F1(I,K)=F01DAF(L,M,X,I,J,E2,IA,E6,IA,N)
      DO 31 I=1,M
31    F1(I,I)=F1(I,I)+1.0
      DO 33 I=1,M
      Y=0.0
      DO 32 J=1,NIC
   32 Y=Y+(F2(I,J)+E6(I,J))*E1(J,1)
33    F2(I,1)=Y
      CALL F04AEF(F1,IA,B,IA,M,M,E6,IA,WKSPCE,E4,IA,E5,IA,IFAIL)
      IOPT=2
      CALL F01CKF(E1,E6,E2,IA,IA,IA,WKSPCE,IA,IOPT,IFAIL)
52    E2(1,1)=0.0
C
C     SET T(0) TO BE ZERO
C
      WRITE(99,993)
993   FORMAT(/' WHAT ARE THE TRANSITION TIMES FOR THE Z(I)?')
      READ(98,*)(E2(I,1),I=2,NZ)
      WRITE(18,99993)  (E2(I,1),I=2,NZ)
99993 FORMAT(/'  TRANSITION TIMES FOR THE Z(I) GIVEN AS := '/
     * (10F13.6)/)
      E2(NZ+1,1)=1000.0
      IF(M.EQ.0)GOTO 780
      CALL A2(C2,C11,CC,E2,C5,C13,C14,C15,C16,E5,C9,CX,X,NZ,
     1 NUSR,N3,IA,L)
      DO 35 I=1,M
35    F3(I,1)=F01DAF(L,NUSR,X,I,L,F3,IA,E5,IA,N)
      DO 38 I=1,M
38    E1(I+NIC,1)=F01DAF(L,M,X,I,L,E6,IA,F3,IA,N)
500   DO 34 I=1,M
      E1(I+NIC,1)=E1(I+NIC,1)-F01DAF(L,M,X,I,L,E6,IA,F2,IA,N)
   34 E1(I+NIC,1)=E1(I+NIC,1)+F01DAF(L,M,X,I,L,E6,IA,F4,IA,N)
780   DO 42 I=1,NSR
      C7(I,1)= DCMPLX(0.0D0,0.0D0)
      DO 42 J=1,NSR
42     C7(I,1)= DCMPLX(E1(J,1),0.0D0)*C6(I,J)+C7(I,1)
      WRITE(99,989)
989   FORMAT(/' HOW MANY TIME PERIODS TO INTEGRATE OVER?')
      READ(98,*)NN
      WRITE(99,988)
988   FORMAT(/' HOW BIG A TIME PERIOD?')
      READ(98,*)XEND
```

```
      T=0.0
116   DO 44 I=1,NUSR
44    E1(I+NSR,1)=F01DAF(L,NSR,X,I,L,E3,IA,E1,IA,N)
      CALL A2(C2,C11,CC,E2,C5,C6,C13,C14,C15,E5,C9,CX,T,NZ,NUSR,N3,IA,L)
      DO 45 I=1,NUSR
      K=I+NSR
45    E1(K,1)=E1(K,1)-E5(I,1)
46    DO 47 I=1,N2
47    E1(I,2)=F01DAF(L,N1,X,I,L,E7,IA,E1,IA,N)
      IFLAG=0
      DO 48 I=1,NZ
      IF(T.LT.E2(I+1,1))GOTO 444
      GOTO 48
444   IF(IFLAG.EQ.0)KSTR=I
      IFLAG=1
48    CONTINUE
49    DO 110 J=1,N2
110   E1(J,2)=E1(J,2)+F01DAF(L,N3,X,J,KSTR,E8,IA,Z,IA,N)
111   WRITE(18,199)T,(E1(I,1),I=1,N1),(E1(I,2),I=1,N2)
      T=T+XEND
      NN=NN-1
      IF(NN.LT.0)GOTO117
      CALL A1(C4,T,E2,C6,L,NSR,IA)
      DO 112 I=1,NSR
      CALL F01DDF(L,NSR,CX,I,L,C6,IA,C7,IA,N,CR,CI)
      E4(I,1)=-CR
112   E4(I,2)=-CI
      CALL A4(C4,C5,CC,C10,C11,E2,C6,C13,C14,C15,C16,C1,E5,CX,
     1 T,NUSR,NSR,N3,NZ,IA,L)
      CALL A6(C4,E2,CX,T,C6,C13,C8,CC,C14,C15,C1,NSR,IA,N3,L)
      DO 113 I=1,NSR
113   C15(I,1)=DCMPLX(E4(I,1),E4(I,2))+C14(I,1)+DCMPLX(E5(I,1),E5(I,2))
119   DO 115 I=1,NSR
      CALL F01DDF(L,NSR,CX,I,L,C3,IA,C15,IA,N,CR,CI)
115   E1(I,1)=-CR
      GOTO 116
117   REWIND 18
      REWIND 19
      RETURN
      END
      SUBROUTINE A1(A,T,B,C,I,N,IA)
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 A(IA,IA),C(IA,IA)
      REAL*8 B(IA),X,Y,T,TT
      TT=T-B(I)
      DO 1 J=1,N
      DO 2 K=1,N
2     C(J,K)= DCMPLX(0.0D0,0.0D0)
      X=DEXP((DREAL(A(J,J)))*TT)
      Y=TT*DIMAG(A(J,J))
1     C(J,J)= DCMPLX((X*DCOS(Y)),(X*DSIN(Y)))
      RETURN
      END
      SUBROUTINE A2(A,B,C,D,F,G,H,P,Q,R,S,CX,T,NZ,NUSR,N3,IA,L)
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 A(IA,IA),B(IA,IA),C(IA,IA),F(IA,IA)
     1,G(IA,IA),
     1 H(IA,IA),P(IA,IA),Q(IA,IA),S(IA,IA),CX
      REAL*8 D(IA),R(IA,IA),T,CR,CI
      I=NZ
```

```
      CALL A5(B,C,CX,Q,NUSR,N3,I,IA,L)
      IF(D(I)-T)2,2,3
2     DO 1 J=1,NUSR
1     P(J,1)=Q(J,1)
      GOTO 14
3     CALL A1(F,T,D,H,I,NUSR,IA)
      DO 4 J=1,NUSR
      CALL F01DDF(L,NUSR,CX,J,L,H,IA,Q,IA,N,CR,CI)
4     P(J,1)= DCMPLX(-CR,-CI)
      I=I-1
10    CALL A5(B,C,CX,Q,NUSR,N3,I,IA,L)
      IF (D(I)-T)5,5,6
6     CALL A1(F,T,D,G,I,NUSR,IA)
      DO 7 J=1,NUSR
7     H(J,J)=G(J,J)-H(J,J)
      DO 8 J=1,NUSR
      CALL F01DDF(L,NUSR,CX,J,L,H,IA,Q,IA,N,CR,CI)
8     P(J,1)=P(J,1)- DCMPLX(CR,CI)
      DO 9 J=1,NUSR
9     H(J,J)=G(J,J)
      I=I-1
      GOTO 10
5     DO 11 J=1,NUSR
11    H(J,J)=DCMPLX(1.0D0,0.0D0)-H(J,J)
      DO 13 J=1,NUSR
      CALL F01DDF(L,NUSR,CX,J,L,H,IA,Q,IA,N,CR,CI)
13    P(J,1)=DCMPLX(-CR,-CI)+P(J,1)
14    DO 15 J=1,NUSR
      CALL F01DDF(L,NUSR,CX,J,L,A,IA,P,IA,N,CR,CI)
15    G(J,1)=DCMPLX(-CR,-CI)
      DO 16 J=1,NUSR
      CALL F01DDF(L,NUSR,CX,J,L,S,IA,G,IA,N,CR,CI)
16    R(J,1)=-CR
      RETURN
      END
      SUBROUTINE A3(A,B,C,D,E,F,G,H,I,T,NUSR,NSR,L,IA,KFLAG)
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 A(IA,IA),B(IA,IA),C(IA,IA),D(IA,IA),E(IA,IA),
     1 F(IA,IA),G(IA,IA)
      REAL*8 H(IA),X,XX,TT,T,Y,ZR,ZI
      TT=0.0
      IF(I-KFLAG)4,4,5
4     CALL A1(A,T,H,C,I,NSR,IA)
      GOTO 6
5     CALL A1(B,T,H,C,I,NUSR,IA)
6     CALL A1(A,T,H,D,L,NSR,IA)
      CALL A1(B,TT,H,E,I,NUSR,IA)
      IF(I-KFLAG)14,14,15
14    DO 1 J=1,NSR
      DO 1 N=1,NUSR
      XX=1.0
      X=DREAL(B(N,N)-A(J,J))
      Y=DIMAG(B(N,N)-A(J,J))
      CALL A02ACF(XX,TT,X,Y,ZR,ZI)
      X=DREAL(C(J,J))-(DREAL(D(J,J)))*(DREAL(E(N,N)))-(DIMAG(D(J,J)))*
     * (DIMAG(E(N,N)))
      Y=DIMAG(C(J,J))-(DREAL(D(J,J)))*(DIMAG(E(N,N)))-(DIMAG(D(J,J)))*
     * (DREAL(E(N,N)))
      XX=DREAL(G(J,N))*X-DIMAG(G(J,N))*Y
      Y=DIMAG(G(J,N))*X+DREAL(G(J,N))*Y
```

```
    1 F(J,N)=DCMPLX((XX*ZR-Y*ZI),(XX*ZI+Y*ZR))
      GOTO 23
 15   DO 11 J=1,NSR
      DO 11 N=1,NUSR
      XX=1.0
      X=DREAL(B(N,N)-A(J,J))
      Y=DIMAG(B(N,N)-A(J,J))
      CALL A02ACF(XX,TT,X,Y,ZR,ZI)
      X=DREAL(C(N,N))-(DREAL(D(J,J)))*(DREAL(E(N,N)))-(DIMAG(D(J,J)))*
     * (DIMAG(E(N,N)))
      Y=DIMAG(C(N,N))-(DREAL(D(J,J)))*(DIMAG(E(N,N)))-(DIMAG(D(J,J)))*
     * (DREAL(E(N,N)))
      XX=DREAL(G(J,N))*X-DIMAG(G(J,N))*Y
      Y=DIMAG(G(J,N))*X+DREAL(G(J,N))*Y
   11 F(J,N)=DCMPLX((XX*ZR-Y*ZI),(XX*ZI+Y*ZR))
   23 CONTINUE
      RETURN
      END
      SUBROUTINE A4(A,B,C,D,E,F,G,H,P,Q,R,S,U,CX,T,NUSR,NSR,N3,NZ,IA,L)
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 A(IA,IA),B(IA,IA),C(IA,IA),D(IA,IA),E(IA,IA),
     1 G(IA,IA),H(IA,IA),P(IA,IA),Q(IA,IA),R(IA,IA),S(IA,IA),CX
      REAL*8 F(IA),U(IA,IA),T,CR,CI
      DO 11 I=1,NSR
      U(I,1)=0.
 11   U(I,2)=0.
      KFLAG=1200
      DO 12 I=1,NZ
      IF(T.GE.F(I).AND.T.LT.F(I+1))KFLAG=I
 12   CONTINUE
      I=NZ
 10   CALL A3(A,B,P,Q,R,G,D,F,I,T,NUSR,NSR,L,IA,KFLAG)
      DO 3 J=1,NSR
      DO 3 K=1,N3
      CALL F01DDF(L,NUSR,CX,J,K,G,IA,E,IA,N,CR,CI)
 3    H(J,K)=DCMPLX(-CR,-CI)
      K=I-1
      DO 4 J=1,N3
 4    G(J,1)=DCMPLX(DREAL(C(J,K))-DREAL(C(J,I)),0.0D0)
      DO 5 J=1,NSR
      CALL F01DDF(L,N3,CX,J,L,H,IA,G,IA,N,CR,CI)
      U(J,1)=-CR+U(J,1)
 5    U(J,2)=-CI+U(J,2)
      I=I-1
      IF(I-2)2,10,10
 2    IF(KFLAG.EQ.1)GOTO 1
      NFLAG=KFLAG-1
      DO 113 IM=1,NFLAG
      JF=IM+1
      CALL A1(A,T,F,H,IM,NSR,IA)
      CALL A1(A,T,F,P,JF,NSR,IA)
      DO 112 J=1,NSR
      P(J,J)=DCMPLX(DREAL(P(J,J))-DREAL(H(J,J)),DIMAG(P(J,J))-DIMAG(H(J,
     CJ)))
 112  CONTINUE
      CALL A5(E,C,CX,G,NUSR,N3,IM,IA,L)
      DO 111 J=1,NSR
      DO 111 K=1,NSR
      CALL F01DDF(L,NSR,CX,J,K,S,IA,P,IA,N,CR,CI)
 111  Q(J,K)=DCMPLX(-CR,-CI)
```

```
      DO 115 J=1,NSR
      DO 115 K=1,NUSR
      CALL F01DDF(L,NSR,CX,J,K,Q,IA,D,IA,N,CR,CI)
115   P(J,K)=DCMPLX(-CR,-CI)
      DO 119 J=1,NSR
      CALL F01DDF(L,NUSR,CX,J,L,P,IA,G,IA,N,CR,CI)
      U(J,1)=-CR+U(J,1)
      U(J,2)=-CI+U(J,2)
119   CONTINUE
113   CONTINUE
1     CALL A5(E,C,CX,G,NUSR,N3,KFLAG,IA,L)
      CALL A1(A,T,F,P,KFLAG,NSR,IA)
      DO 6 J=1,NSR
6     P(J,J)=DCMPLX(1.0D0,0.0D0)-P(J,J)
      DO 7 J=1,NSR
      DO 7 K=1,NSR
      CALL F01DDF(L,NSR,CX,J,K,S,IA,P,IA,N,CR,CI)
7     Q(J,K)=DCMPLX(-CR,-CI)
      DO 8 J=1,NSR
      DO 8 K=1,NUSR
      CALL F01DDF(L,NSR,CX,J,K,Q,IA,D,IA,N,CR,CI)
8     P(J,K)=DCMPLX(-CR,-CI)
      DO 9 J=1,NSR
      CALL F01DDF(L,NUSR,CX,J,L,P,IA,G,IA,N,CR,CI)
      U(J,1)=-CR+U(J,1)
9     U(J,2)=-CI+U(J,2)
      RETURN
      END
      SUBROUTINE A5(A,B,CX,C,NUSR,N3,I,IA,L)
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 A(IA,IA),B(IA,IA),CX,C(IA,IA)
      REAL*8 CR,CI
      DO 1 J=1,NUSR
      CALL F01DDF(L,N3,CX,J,I,A,IA,B,IA,N,CR,CI)
1     C(J,1)=DCMPLX(-CR,-CI)
      RETURN
      END
      SUBROUTINE A6(A,B,CX,T,D,E,F,G,H,P,Q,NSR,IA,N3,L)
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 A(IA,IA),CX,D(IA,IA),E(IA,IA),F(IA,IA),
     1 G(IA,IA),H(IA,IA),P(IA,IA),Q(IA,IA)
      REAL*8 B(IA),T,CR,CI
      DO 12 J=1,NSR
12    P(J,1)=(0.,0.)
      I=2
      CALL A1(A,T,B,D,L,NSR,IA)
10    IF(T-B(I))1,1,2
2     CALL A1(A,T,B,E,I,NSR,IA)
      DO 3 J=1,NSR
3     D(J,J)=D(J,J)-E(J,J)
      GOTO 4
9     DO 5 J=1,NSR
5     D(J,J)=E(J,J)
      I=I+1
      GOTO 10
1     DO 6 J=1,NSR
6      D(J,J)=D(J,J)-DCMPLX(1.0D0,0.0D0)
4     DO 7 J=1,NSR
      DO 7 K=1,N3
      CALL F01DDF(L,NSR,CX,J,K,D,IA,F,IA,N,CR,CI)
```

```
7       H(J,K)=DCMPLX(-CR,-CI)
        K=I-1
        DO 8 J=1,NSR
        CALL F01DDF(L,N3,CX,J,K,H,IA,G,IA,N,CR,CI)
8       P(J,1)=P(J,1)+DCMPLX(-CR,-CI)
        IF(B(I).LT.T)GOTO 9
        DO 11 J=1,NSR
        CALL F01DDF(L,NSR,CX,J,L,Q,IA,P,IA,N,CR,CI)
11      H(J,1)=DCMPLX(-CR,-CI)
        RETURN
        END
```

International Finance Discussion Papers

IFDP NUMBER	TITLES	AUTHOR(s)
	1986	
276	Post-simulation Analysis of Monte Carlo Experiments: Interpreting Pesaran's (1974) Study of Non-nested Hypothesis Test Statistics	Neil R. Ericsson
275	A Method for Solving Systems of First Order Linear Homogeneous Differential Equations When the Elements of the Forcing Vector are Modelled as Step Functions	Robert A. Johnson
274	International Comparisons of Fiscal Policy: The OECD and the IMF Measures of Fiscal Impulse	Garry Schinasi
273	An Analysis of the Welfare Implications of Alternative Exchange Rate Regimes: An Intertemporal Model with an Application	Andrew Feltenstein David Lebow Anne Sibert
	1985	
272	Expected Fiscal Policy and the Recession of 1982	William H. Branson Arminio Fraga Robert A. Johnson
271	Elections and Macroeconomic Policy Cycles	Kenneth Rogoff Anne Sibert
270	Assertion Without Empirical Basis: An Econometric Appraisal of Monetary Trends in ... the United Kingdom by Milton Friedman and Anna J. Schwartz	David F. Hendry Neil R. Ericsson
269	Canadian Financial Markets: The Government's Proposal for Reform	Garry J. Schinasi
268	Was It Real? The Exchange Rate Interest Differential Relation, 1973-1984	Richard Meese Kenneth Rogoff
267	The U.K. Sector of the Federal Reserve's Multicountry Model: The Effects of Monetary and Fiscal Policies	Hali J. Edison
266	Optimal Currency Basket in a World of Generalized Floating: An Application to the Nordic Countries	Hali J. Edison Erling Vardal

---

Please address requests for copies to International Finance Discussion Papers, Division of International Finance, Stop 24, Board of Governors of the Federal Reserve Board, Washington, D.C. 20551.

International Finance Discussion Papers

IFDP NUMBER	TITLES	AUTHOR(s)
265	Money Demand in Open Economies: A Currency Substitution Model for Venezuela	Jaime Marquez
264	Comparing Costs of Note Issuance Facilities and Eurocredits	Rodney H. Mills
263	Some Implications of the President's Tax Proposals for U.S. Banks with Claims on Developing Countries	Allen B. Frankel
262	Monetary Stabilization Policy in an Open Economy	Marcus H. Miller
261	Anticipatory Capital Flows and the Behaviour of the Dollar	Arnold Kling
260	Simulating Exchange Rate Shocks in the MPS and MCM Models: An Evaluation	Arnold Kling
259	Trade Policy for the Multiple Product Declining Industry	Catherine Mann
258	Long Memory Models of Interest Rates, the Term Structure, and Variance Bounds Tests	Gary S. Shea
257	Currency Substitution and the New Divisia Monetary Aggregates: The U.S. Case	Jaime Marquez
256	The International Transmission of Oil Price Effects and OPEC's Pricing Policy	Jaime Marquez
255	U.S. Banks' Lending to Developing Countries: A Longer-Term View	Henry S. Terrell Rod Mills
254	Conditional Econometric Modelling: An Application to New House Prices in the United Kingdom	Neil R. Ericsson David F. Hendry
253	Low Pushing: Doctrine and Theory	William Darity, Jr.

1984
(partial listing)

252	Postwar Financial Policies in Taiwan, China	Robert F. Emery
251	Foreign Exchange Constraints and Growth Possibilities in LDCs	Jaime Marquez
250	The Determination of Front-end Fees on Syndicated Eurocurrency Credits	Rodney H. Mills Henry S. Terrell