UNIVERSIDAD NACIONAL DE INGENIERIA
FACULTAD DE INGENIERIA MECANICA Y ELECTRICA

"Solución de Sistemas con Parámetros que Varían con el Tiempo"

TESIS DE GRADO

PARA OPTAR EL TITULO DE INGENIERO MECANICO Y ELECTRICISTA

BACHILLER:
ERWIN ENRIQUE FETZER SENGE
PROMOCION 1964

LIMA - PERU
1968

A MI ESPOSA,

CHRIS

INDICE

I. INTRODUCCION

El propósito de esta tesis es encontrar una solución general para sistemas lineales que contienen parámetros que varían con el tiempo. No hay sistema en el cual sus parámetros no varían con el tiepmo. Además, hay sistemas que contienen parámetros que se hacen variar intencionalmente con el tiempo. Si los parámetros del sistema varían muy lentamente, es válido de analizar el sistema suponiendo parámetros constantes dentro de cierto intervalo de tiempo. Con el método desarrollado en esta tesis también pueden ser analizados algunos tipos de sistemas no lineales.

La solución en forma cerrada (es decir, una combinación finita de funciones elementales tales como polinomios, exponenciales, logaritmos, integrales indefinidas) de los sistemas de ecuaciones diferenciales con parámetros que varían con el tiempo es muy difícil y muchas veces imposible. Por lo tanto, se aplicarán para la solución métodos numéricos implementados por una computadora.

El método numérico usado está basada en la solución de sistemas de ecuaciones diferenciales para el problema del valor inicial. La solución numérica tendrá un valor pre-especificado de aproximación.

Para resolver el problema el sistema tiene que plantearse en forma de Variables de Estado $\dot{X}=AX$ (donde $\dot{X}$ y X son vectores columna y A es una matriz n x n) y las condiciones iniciales tienen que ser también especificadas.

II. REVISION DE LA LITERATURA

En la literatura no hay un método general para resolver sistemas que varían con el tiempo, pero hay bastante literatura relacionada con la solución de sistemas con parámetros constantes que es un caso especial de los sistemas que varían con el tiempo.

Mucho del material relacionado a las sistemas que varían con el tiempo se refieren a problemas especiales y métodos teóricos que en la práctica no se pueden aplicar fácilmente.

El problema principal reside como lo dice R. G. Brown (1) y S. C. Gupta (6), que no hay una transformada que se pueda aplicar satisfactoriamente para todos los casos de parámetros que varían con el tiempo.

L. A. Zadeh (20) introdujo la idea de la función del sistema H(s,t) pero la dificultad en el método es que uno intercambia una ecuación diferencial de enésimo órden por otra ecuación diferencial, y en muchos casos la ecuación auxiliar de H(s,t) será mucho mas difícil que la ecuación original. Claro está que si llegasemos a conocer de algún modo H(s,t), el comportamiento general del sistema puede ser fácilmente encontrado y es de gran valor.

El operador adjunto desarrollado por W. Kaplan (11) da valores satisfactorios cuando podemos obtener una solución de la ecuación adjunta.

La idea del matrizant desarrollado en P. M. DeRusso (4) y L. A. Pipes (14) presenta el problema en desarrollar la Matriz de Transición en una serie infinita que es difícil de evaluar.

En el área de sistemas de ecuaciones diferenciales, hay bastante información, la cual se puede encontrar en Earl A. Coddington (2), E. L. Ince, (9) y M. Tenenbaum (19). También existen en la literatura varias demostraciones de la existencia y unicidad de la solución para sistemas de ecuaciones con parámetros que varían con el tiempo.

También existen varios métodos de solución de ecuaciones diferenciales basados en los llamodos métodos de un sólo paso y métodos de varios pasos como se puede ver en R. L. Crane (3), P. Henrici (7,8) y A. Ralston (15).

III. ELEMENTOS BASICOS EN EL ANALISIS USANDO LAS VARIABLES DE ESTADO

La noción de estado es bien general. Cualquier variable de un sistema que define algún comportamiento del sistema sujeto a variaciones puede tomarse como variable de estado. Conociendo todas las variables de estado, se define el estado de sistema. En órden de aplicar este concepto de estado a un sistema dado, se debe tener presente dos puntos importantes. Primero, el número de variables de estado que representan el sistema debe de ser el mismo que el órden del sistema y, segundo, se debe tener asímismo algunas reglas sobre qué elementos del sistema deben ser representados por variables.

Es conveniente el clasificar las variables de estado que caracterizan o están asociados con el sistema en (1) variables de entrada, que representan el estímulo al sistema en estudio; (2) variables de salida, que representan la salida que se está investigando, y (3) variables de estado, que representan el comportamiento dinámico del sistema.

El conjunto de todos los valores posibles que el vector entrada puede asumir al tiempo t forman el espacio de entrade del sistema. De la misma forma, el conjunto de todos los valores posibles de salida al tiempo t forman el espacio de salida del sistema y el conjunto de valores que el vector de estado puede asumir al tiempo t forma el espacio de estado que puede asumir el sistema.

En cualquier instante t, el estado del sistema es una función del estado inicial y del vector entrada.

Un sistema lineal estacionario puede ser descrito por un conjunto de ecuaciones deferenciales de primer órden con coeficientes constantes, el cual puede ser descrito en forma matemática en

$$\frac{dX(t)}{dt} = AX(t) \qquad (3\text{-}1)$$

Donde X(t) es un vector columna para la variable de entrada y variables de estado, y A es una matriz con elementos constantes. Si las variables de entrada son tratadas como variables de estado del sistema total, el vector X puede ser considerado como vector de estado del sistema total.

Si se tiene X(0+) como condición inicial, para (3-1) se puede aplicar la transformación de Laplace

$$sX(s)-X(0+) = AX(s)$$

$$(sI-A)X(s) = X(0+)$$

$$X(s) = (sI=A)^{-1}X(0+)$$

Invirtiendo

$$X(t) = L^{-1} \quad (sI-A)^{-1} \quad X(0+)$$

$$\text{Si } \emptyset(t) = L^{-1} \quad (sI-A)^{-1}$$

$$\text{Entonces } X(t) = \emptyset(t)X(0=)$$

La función Ø(t) es referida como la matriz total de transición del sistema.

La solución de la ecuación diferencial (3-1) es

$$\exp(tA) = I+A+\frac{A^2t^2}{2!} \qquad + \ldots$$

Ya que A es constante

$$\frac{d}{dt} \quad (\exp(tA)) = A+A^2t+ \frac{A^3t^2}{2!} \qquad +$$

Sustituyendo en la ecuación (3-1)

$$A+A^2t+\frac{A^3t^2}{2!} \quad + \ldots = A(I+At+\frac{A^2t^2}{2!} \quad + \ldots)$$

la cual se satisface para coeficientes constantes.

$$X(t) = \exp(tA)X(0+)$$

Por lo tanto

$$\emptyset(t) = \exp(tA)$$

Para el caso en el cual los elementos de la matriz A no son constantes sino que varían con el tiempo, la ecuación diferencial es

$$\frac{dX(t)}{dt} = A(t)X(t) \tag{3-2}$$

La solución exp(tA) es verdadera en el caso de que la matriz A puede satisfacer la condición comutativa. Es decir, $A(t_1)\ A(t_2) = A(t_2)\ A(t_1)$ para todos t_1 y t_2. La solución de la ecuación (3-2) puede ser obtenida por el método conocido como el de Peano-Baker. El principio es bastante sencillo. Suponiendo que los valores que toma la columna X a t=0 es X=X . Integrando directamente (3-2) se obtiene

$$X(t) = X_o+ \quad \int_0^t A(\tau) \quad X(\tau)d\tau \tag{3-3}$$

Esta es una ecuación integral para determinar X. La variable τ es una variable aparente de integración.

Esta ecuación integral puede ser resuelta por un método iterativo sustituyendo repetidamente el valor obtenido de X en la integral de la ecuación (3-3).

Si se introduce el siguiente operador

$$Q = \int_{o}^{t} (\qquad) d\tau$$

la ecuación (3-3) puede ser escrita en la forma

$$X(t) = X_o + Q(A(\tau)X(\tau))$$

Sustituyendo repetidamente se obtiene

$$X(t) = (I + Q(A) + A(AQ(A)) + \ldots\ldots) \quad X_o$$

En la serie es la matriz unidad de órden N y la integral de A está dentro de los límites o y t. Para obtener Q(AQ(A)), A y Q(A) son multiplicados en el siguiente órden, A Q(A) y el producto es integrado entre los límites o y t. Los términos restantes son formados de la misma forma. Si los elementos de la matriz A se mantienen dentro del rango de o á t, se puede demostrar que la serie es convergente absoluta y uniforme. Se puede definir la matriz cuadrada G(A). La solución de la ecuación diferencial (3-2) estará dada por

$$X = G(A) \quad X_o$$

y $\frac{d}{dt}$ G(A) = A G(A) expresa la propiedad fundamental de G(A) o matrizant.

El problema que resulta de aplicar este método es que G(A) está compuesta de una serie infinita y a menos que converga rapidamente la computación se hace extensa.

La mejor manera de resolver el problema sería tratar con las ecuaciones diferenciales directamente y por métodos numéricos.

IV. FUNDAMENTO MATEMATICO (NUMERICO)

Considerando el problema del valor inicial $\dot{X} = AX$, ó $X' = f(t,X)$, $X(t_o) = S$. Digamos que la función $f = f(t,X)$ sea contínua en $a \leq t \leq b$, $-\infty \leq X < \infty$, y $f(t)$ satisfaga la condición de Lipschitz en X, es decir, que existe una constante L de tal manera que para cualquier X, Z y todos los $t \varepsilon [a,b]$, $|f(t,X)-f(t,Z)| \leq L|X-Z|$. Entonces para cualquier valor inicial de S, el problema de valor inicial tiene una solución única $X = X(t)$ para $t \varepsilon [a,b]$ como se puede ver en Henrici (8).

Cualquier algoritmo para resolver ecuaciones diferenciales que usa sólo t_n y X_n para la aproximación de X_{n+1} en t_{n+1} de la solución se llama método en un sólo paso. Si

$$X_{n+1} = X_n + h\emptyset(t_n, X_n; h)$$

$\emptyset$ es la función incremental y h es el paso. La función incremental para el método clásico de Runge-Kutta es

$$\emptyset\ (t,X;h) = 1/6\ \ (K_o+2K_1+2K_2+K_3)$$

Donde

$$K_o = f(t,X)$$
$$K_1 = f(t+1/2h,\ X+1/2hK_o)$$
$$K_2 = f(t+1/2h, X+1/2hK_1)$$
$$K_3 = f(t+h, X+hK_2)$$

La principal desventaja en este proceso es que cuatro substituciones se deben hacer en $f(t,x)$ por cada paso de integración o solución. Si la función $f(t,x)$ es complicada, se consume mucho tiempo en la computadora.

Para sobrepasar esta desventaja y ahorrar espacio en la memoria, el método de Runge-Kutta por Gill (5) será usado. En este método se define nuevas variables r y q, pero el proceso es directo, como se puede ver a continuación.

$$K_o = hg(t_o, X_o) \qquad r_1 = 1/2K_o + 1.q_o$$

$$q_o = 0 \text{ primera vez}$$

$$q_o = q_4 \text{ sucesivamente}$$

$$X_1 = X_o + r_1 \qquad q_1 = 0 + 3r_1 - 1/2K_o$$

$$K_1 = hf(t_o + 1/2h, X1) \qquad r_2 = (1-\sqrt{1/2})K_1 + (1-\sqrt{1/2})q_1$$

$$X_2 = X_1 + r_2 \qquad q_2 = q_1 + 3r_2 - (1-\sqrt{1/2})K_1$$

$$K_2 = hf(t_o + 1/2h, X_2) \qquad r_3 = (1+\sqrt{1/2})K_2 + (1+\sqrt{1/2})q_2$$

$$X_3 = X_2 + r_3 \qquad q_3 = q_2 + 3r_3 - (1+\sqrt{1/2})K_2$$

$$K_3 = hf(t_o + h, X_3) \qquad r_4 = 1/6K_3 + 1/3q_3$$

$$X_4 = X_3 + r_4 \qquad q_4 = q_3 + 3r_4 - 1/2K_3$$

El punto X_{n+1} en la solución es

$$X_4 \text{ a } t_{n+1} = t_n + h$$

Cada cantidad es calculada y almacenada en el registro que contenía el valor anterior y que ya no es útil. La solución posee en común con Runge-Kutta la ventaja que la variable independiente asume valores en los puntos extremos y en el punto medio de cada paso. También ambos métodos son de cuarto órden.

Para acelerar la solución, es conveniente usar métodos de múltiple paso. Este método es aplicado después que cuatro puntos de solución ya conocidos.

La solución de la ecuación $X' = f(t,X)$ puede ser aproximada por una ecuación de diferencias

$$X_{n+k} = \sum_{j=o}^{k-1} a_j X_{n+j} + h \sum_{j=o} b_j X_{n+j} \qquad (4-1)$$

Donde $t_{n+j} = t_{n+jh}$, $X_{n+j} = X(t_{n+j})$, $X'_{n+j} = f(t_{n+j}, X_{n+j})$ y a_j y b_u son constantes de tal manera que X_{n+k} es una buena aproximación a la solución. Si $b_k = 0$, la ecuación (4-1) provee en un método explícito para computar X_{n+k} cuando los valores de X_{n+j} para j=o, 1....K-1 son conocidos. La ecuación (4-1) con b_k=o es una fórmula de predicción.

Cuando $b_k \neq o$, la ecuación (4-1) provee una ecuación implícita para determinar X_{n+k} desde que X_{n+k} está en el término $X'_{n+k} = f(t_{n+k}, X_{n+k})$. Esta fórmula es de corrección.

Para resolver la ecuación $X' = f(t,X)$, se puede usar una fórmula de predicción o generar el valor X_{n+k} y sucesivamente o iterativamente se puede usar una fórmula de corrección.

Henrici (8, página 216) demuestra que si $f(t,X)$ es una función contínua y satisface la condición de Lipschitz con respecto a X, la ecuación (4-1) tiene una solución única y el proceso iterativo converge a una solución única independiente del valor de X_{n+k}, si h es lo suficientemente pequeña.

Las constantes usadas para las fórmulas de predicción y corrección están especificadas en R. L. Crane (3) y son los siguientes:

Para la fórmula de predicción:

$a_o = -0.69735280$

$a_1 = 2.0172069$

$a_2 = 1.8675052$

$a_3 = 1.5476511$

$b_o = -.7143200166666667$

$b_1 = 1.81861065$

$b_2 = -2.03168765$

$b_3 = 2.002747216666667$

Para la fórmula de corrección:

$a_3 = 1$

$b_1 = 0.04166666666666667$

$b_2 = -.2083333333333333$

$b_3 = .7916666666666667$

$b_4 = .375$

V. SOLUCION POR COMPUTADORA

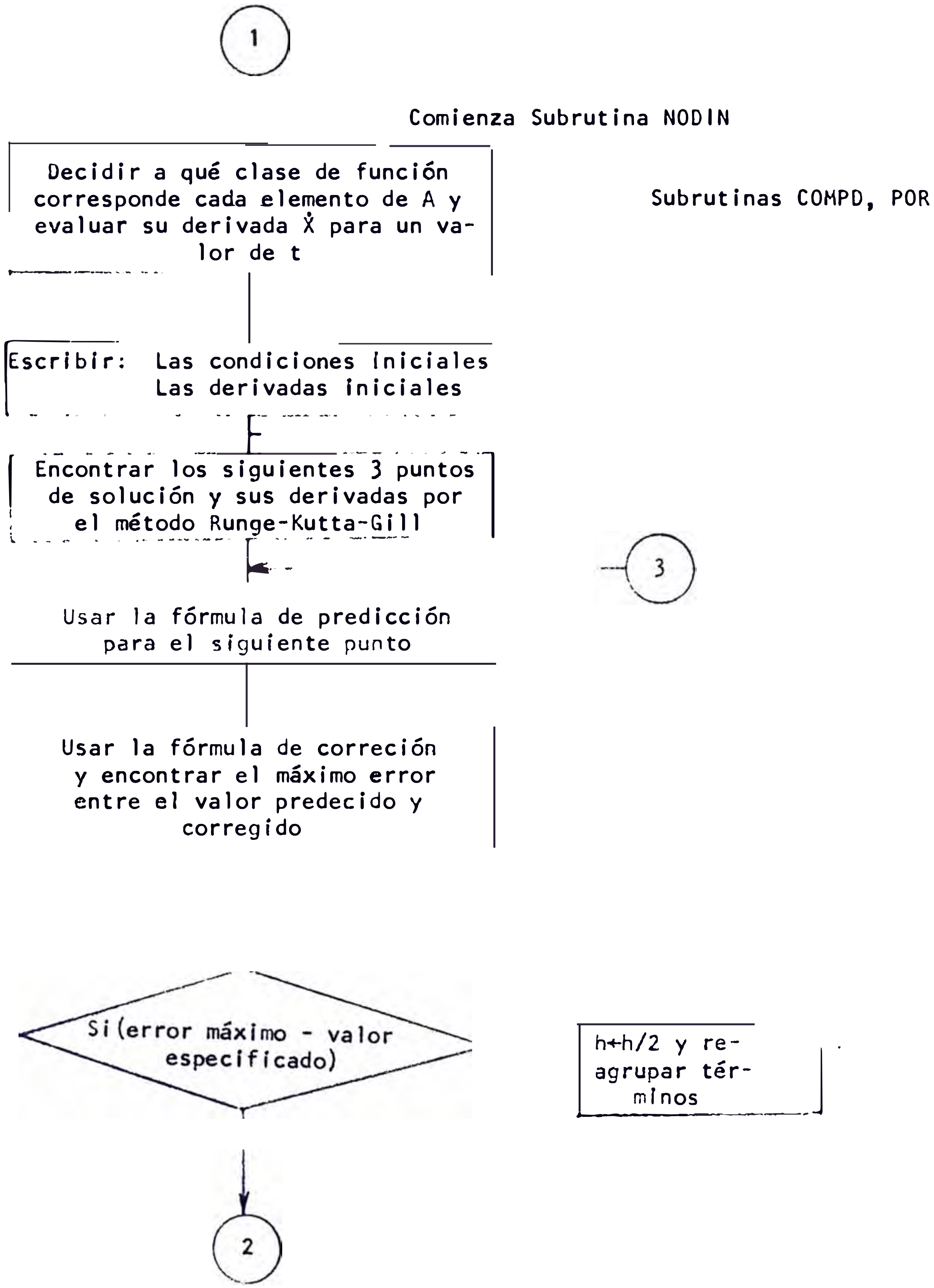
A. Descripción del Programa

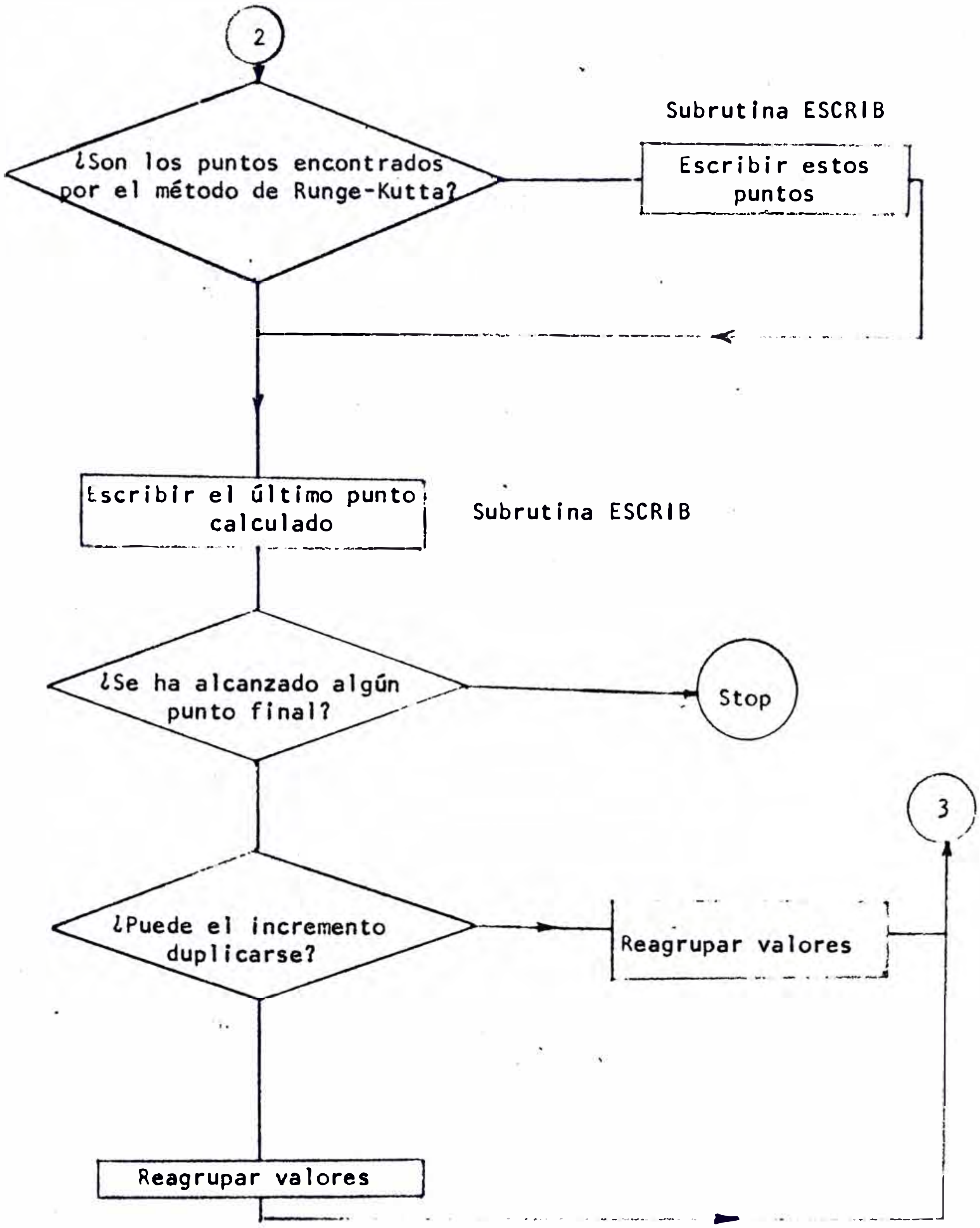
El programa se compone de un principal y cuatro subrutinas. El programa principal gobierna los datos iniciales y controla los puntos finales que deben ser alcanzados, especialmente cuando se trata de función de salto.

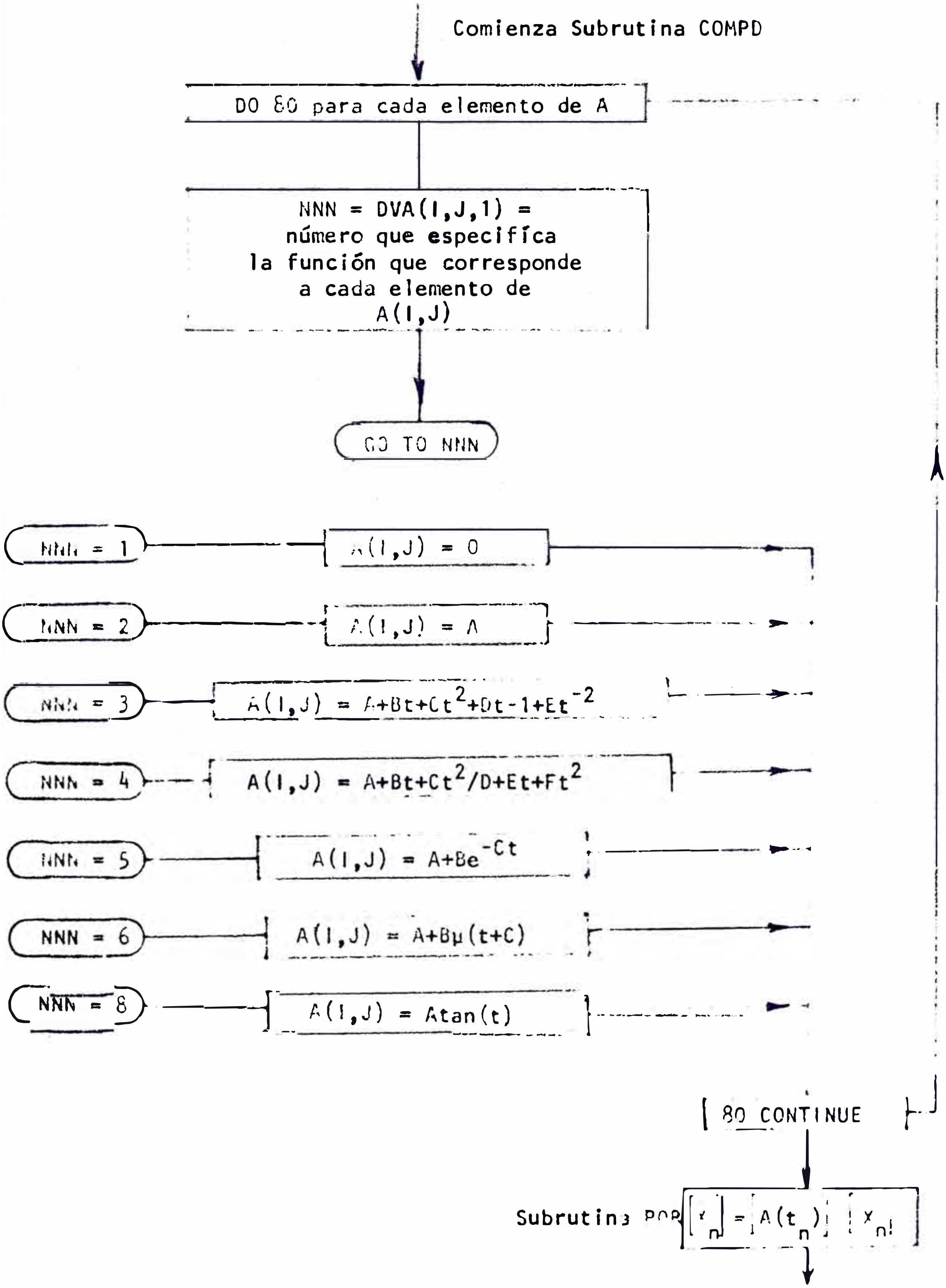
La subrutina COMPD decide a qué tipo de f(t) corresponde a cada elemento de la matriz A y evalue la matriz A(COE) para un tiempo específico t.

Subrutina POR evalua la multiplicación $X(t_n) = A(t_n)\ X(t_n)$. Subrutina ESCRIB escribe la respuesta que se busca y también comprueba si algún punto que debe ser alcanzado ha sido obtenido. Subrutina NODIN es el controlador principal en el programa. Controla en incremento de la variable independiente el método de Runge-Kutta las fórmulas de predicción y correción, así como el grado de aproximación que se quiere alcanzar.

En el siguiente diagrama de flujo se puede apreciar los pasos seguidos en el procedimiento.


1
Comienza Subrutina NODIN
Decidir a qué clase de función corresponde cada elemento de A y evaluar su derivada $\dot{X}$ para un valor de t
Subrutinas COMPD, POR
Escribir: Las condiciones iniciales
Las derivadas iniciales
Encontrar los siguientes 3 puntos de solución y sus derivadas por el método Runge-Kutta-Gill
3
Usar la fórmula de predicción para el siguiente punto
Usar la fórmula de correción y encontrar el máximo error entre el valor predecido y corregido
Si (error máximo - valor especificado)
h←h/2 y reagrupar términos
2


2
Subrutina ESCRIB
¿Son los puntos encontrados por el método de Runge-Kutta?
Escribir estos puntos
Escribir el último punto calculado
Subrutina ESCRIB
¿Se ha alcanzado algún punto final?
Stop
3
¿Puede el incremento duplicarse?
Reagrupar valores
Reagrupar valores



Comienza Subrutina COMPD
DO 80 para cada elemento de A
NNN = DVA(I,J,1) = número que especifíca la función que corresponde a cada elemento de A(I,J)
GO TO NNN
NNN = 1
A(I,J) = 0
NNN = 2
A(I,J) = A
NNN = 3
A(I,J) = A+Bt+Ct2+Dt-1+Et-2
NNN = 4
A(I,J) = A+Bt+Ct2/D+Et+Ft2
NNN = 5
A(I,J) = A+Be-Ct
NNN = 6
A(I,J) = A+Bμ(t+C)
NNN = 8
A(I,J) = Atan(t)
80 CONTINUE
Subrutina POR

VI. EJEMPLOS Y SOLUCIONES

Para probar el programa propuesto, se han realizado cuatro ejemplos. Cada uno de ellos representa un sistema de segundo órden sin entradas pero con condiciones iniciales especificadas.

En la salida obtenida en las figuras 1-4, la entrada está escrita de la misma forma que se ha leído (Apéndice B). El incremento inicial de la variable independiente también está mostrado. Bajo el título de "Numerical Solution" están las siguientes cuatro columnas:

Columna	<u>Explicación</u>
X	Tiempo
Numerical Approximation	Valor encontrado por técnicas númericas.
True Value	Valor encontrado por sustitución en la solución analítica.
Error	La diferencia entre los dos valores anteriores. Este muestra la aproximación.

Primer ejemplo:

$$X + \frac{t^2+4t+2}{t^2+3t+2}\dot{X} + \frac{t}{t^2+3t+2}X = 0 \qquad (6\text{-}1)$$

con condiciones iniciales $X = 4$, $\dot{X} = 12$ a $t = 0$. Si definimos las dos variables de estado

$$X_1 = X, \qquad X_2 = \dot{X}$$

la ecuación original (6-1) de segundo órden se puede reducir a dos ecuaciones simultáneas de primer órden,

$$\dot{X}_1 = X_2$$

$$\dot{X}_2 = \frac{t^2+4t+2}{t^2+3t+2} X_2 - \frac{t}{t^2+3t+2} X_1$$

o en forma matricial

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{-t}{t^2+3t+2} & \frac{-t^2+4t+2}{t^2+3t+2} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \qquad (6\text{-}2)$$

La solución de (6-1) en forma cerrada es

$$X = C_1 \exp(-t) + C_2 (t+2)^{-1}$$

y a t = 0, $X = 4$, $\dot{X} = 12$

$$C_1 = -28, \qquad C_2 = 64$$

Por lo tanto, la solución es

$$X = -28\exp(-t)+64(t+2)^{-1} \qquad (6\text{-}3)$$

Esta ecuación (6-3) nos sirve para comprobar el valor númerico obtenido por el programa y nos dará asímismo el grado de aproximación deseado. La solución de este primer ejemplo está en la Fig. 1.

INPUT DATA

```
0.0              0.50000000D 00 0.5000000D 01 0.0                2 400   5
4.000000012.0[illegible]00000
  1.00000   0.0         0.0         0.0         0.0         0.0         0.0
  2.00000   1.[illegible]00000   0.0         0.0         0.0         0.0         0.0
  4.00000   0.0        -1.00000     0.0         2.00000     3.00000     1.00000
  4.00000  -2.[illegible]0000  -4.00000    -1.00000     2.00000     3.00000     1.00000
```

NUMERICAL SOLUTION

H= 0.[illegible]99999999999900D 00

X	NUMERICAL APROX	TRUE VALUE	ERROR
0.0	0.[illegible]99999999999990D 01	0.39999999999999990D 01	0.0
	HALVED		
	HALVED		
	HALVED		
	HALVED		
0.0625	0.47[illegible]673744542407[illegible]0D 01	0.47267372715257030D 01	-0.17389837481118750D-06
0.1250	0.[illegible]7342019591000D 01	0.54077337864548580D 01	-0.41550424256797640D-06
0.1375	0.[illegible]32791[illegible]0239200 01	0.60443275480916460D 01	-0.37141074571422880D-06
0.2500	0.[illegible]019561[illegible]757600 01	0.66380225184451060D 01	0.29566775303635470D-05
0.3125	0.7[illegible]3254[illegible]2947OD 01	0.71904380651697020D 01	0.55166467554101900D-05

Fig. 1. Solución del primer ejemplo.

0.3750	0.77032613754272440D 01	0.77032686149054050D 01	0.72394781618356750D-05
0.4375	0.81782426834106440D 01	0.81782515164292800D 01	0.88330186365276360D-05
0.5000	0.86171321868896470D 01	0.86171415280462590D 01	0.93411566126633240D-05
0.5625	0.90216808319091780D 01	0.90216906636317140D 01	0.98317225365462920D-05
0.6250	0.93936223983764620D 01	0.93936323824206530D 01	0.99840441905030250D-05
0.6875	0.97346591949462870D 01	0.97346693051857460D 01	0.10110239458072080D-04
0.7500	0.100464523475952140D 02	0.10046463795978840D 02	0.10320026710530760D-04
0.8125	0.10330620765686030D 02	0.10330630873285290D 02	0.10107599280217960D-04
0.8750	0.10588723182678200D 02	0.10588733014219140D 02	0.98315409360338870D-05
0.9375	0.10822266578674310D 02	0.10822276495602810D 02	0.99169285050493250D-05
1.0000	0.11032699584960910D 02	0.11032708980532920D 02	0.93955720110727700D-05
1.0625	0.11221408843994120D 02	0.11221418111518180D 02	0.92675240424000530D-05
1.1250	0.11389721870422350D 02	0.11389730913966200D 02	0.90435438409031100D-05
DOUBLED			
1.2500	0.11670159339904760D 02	0.11670173380222360D 02	0.14040317584118590D-04
1.3750	0.11883440017700190D 02	0.11883454280430050D 02	-0.14262729866487670D-04
1.5000	0.12038056373596180D 02	0.12038069801558240D 02	0.13427962059608920D-04
1.6250	0.12141635894775370D 02	0.12141645508075660D 02	-0.96133002820408800D-05
1.7500	0.12200990676879870D 02	0.12200996250054200D 02	0.55731743202613850D-05
1.8750	0.12222188949584950D 02	0.12222189960600050D 02	-0.10110151080766600D-05
2.0000	0.12210615158081040D 02	0.12210612069374840D 02	-0.30887062081319500D-05
2.1250	0.12171034812927230D 02	0.12171028403683360D 02	-0.64092438800678030D-05
2.2500	0.12107654571533200D 02	0.12107645241679550D 02	-0.93298536403274550D-05
2.3750	0.12024177551269520D 02	0.12024165730672840D 02	-0.11820596680367480D-04
2.5000	0.11923856735229490D 02	0.11923842260753050D 02	-0.14474476436321360D-04
2.6250	0.11809540748596170D 02	0.11809524640878790D 02	-0.16107717394575570D-04

Fig. 1. (Continuación)

2.7500	0.11683721542358390D 02	0.11683704096738490D 02	-0.17445619894917970D-04
2.8750	0.11548571586608870D 02	0.11548553222099350D 02	-0.18364509524415550D-04
3.0000	0.11405981063842760D 02	0.11405962085699790D 02	-0.18978142964032310D-04
3.1250	0.11257590293884270D 02	0.11257570736593350D 02	-0.19557290904526050D-04
3.2500	0.11104818344116210D 02	0.11104798371187960D 02	-0.19972928236633410D-04
3.3750	0.10948889732360830D 02	0.10948869431459380D 02	-0.20300901442560340D-04
3.5000	0.10790857315063460D 02	0.10790836900538700D 02	-0.20414524758205460D-04
3.6250	0.10631623268127430D 02	0.10631603052359810D 02	-0.20215767617237620D-04
3.7500	0.10471958160400380D 02	0.10471937898640430D 02	-0.20261759950068710D-04
3.8750	0.10312515258789040D 02	0.10312495560812990D 02	-0.19697976059607190D-04
4.0000	0.10153848648071270D 02	0.10153828777782080D 02	-0.19870289179602070D-04
	DOUBLED		
4.2500	0.98406467437744130D 01	0.98406014505480160D 01	-0.45293226393017230D-04
4.5000	0.95351591110229480D 01	0.95351019430830570D 01	-0.57167939887747800D-04
4.7500	0.92392997741699200D 01	0.92392340157941020D 01	-0.65758375818125670D-04
5.0000	0.89542608261108380D 01	0.89541946268827470D 01	-0.66199228089924820D-04

Fig. 1. (Continuación)

Segundo ejemplo:

$$\ddot{X}-(1+4\exp(+t))\dot{X}+3\exp(2t)X = 0 \tag{6-4}$$

Con condiciones iniciales

$$X = 0, \qquad \dot{X} = e = 2.7182818 \text{ a } t = 0$$

Si definimos las variables de estado

$$X_1 = X, \qquad X_2 = \dot{X}$$

obtenemos

$$\begin{vmatrix} \dot{X}_1 \\ \dot{X}_2 \end{vmatrix} = \begin{bmatrix} 0 & 1 \\ -3\exp(2t) & 1+4\exp(t) \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{6-5}$$

La solución de (6-4) en forma cerrada es

$$X = C_1\exp(\exp(t))+C_2\exp(3\exp(t))$$

a $t = 0$, $\quad X_1 = 0$, $\quad X_2 = e$

Obtenemos

$$C_1 = -.5$$

$$C_2 = .5\exp(-2) = 0.067665$$

Por lo tanto, la solución es

$$X = -.5\exp(\exp(t))+.067665\ \exp(3\exp(t)) \tag{6-6}$$

La solución de este ejemplo está en la Fig. 2.

INPUT DATA

0.0	0.5000000D 00	0.3000000D 00	0.0	2	300	20
0.0	2.7182818					
1.00000	0.0	0.0	0.0	0.0	0.0	0.0
2.00000	1.00000	0.0	0.0	0.0	0.0	0.0
5.00000	0.0	-3.00000	-2.00000	0.0	0.0	0.0
5.00000	1.00000	4.00000	-1.00000	0.0	0.0	0.0

NUMERICAL SOLUTION

H= 0.14999999999999970D-01

X	NUMERICAL APROX	TRUE VALUE	ERROR
0.0	0.0	-0.31601551226945170D-05	-0.31601551226945170D-05
0.0150	0.42343866647570370D-01	0.42340567842315190D-01	-0.32980525518077900D-05
0.0300	0.87996628077235100D-01	0.87993168048575000D-01	-0.34600286600294770D-05
0.0450	0.13722962859901600D 00	0.13722600166058750D 00	-0.36269384284803840D-05
0.0600	0.19033867120742800D 00	0.19033484734837900D 00	-0.38238585489125370D-05
0.0750	0.24764633178710940D 00	0.24764231962920080D 00	-0.40121579085727860D-05
0.0900	0.30950474739074710D 00	0.30950054932474980D 00	-0.41980659972740120D-05
DOUBLED			
0.1200	0.44844812154769900D 00	0.44844355394847700D 00	-0.45675992219607030D-05
0.1500	0.61069792509078980D 00	0.61069273519201700D 00	-0.51898987727394520D-05
0.1800	0.80043322677612300D 00	0.80047727500733220D 00	-0.59517682908793980D-05
0.2100	0.10223831835937490D 01	0.10223812160200950D 01	-0.69675736540375490D-05
0.2400	0.12840433120727530D 01	0.12840356755520710D 01	-0.76365206804762640D-05
0.2700	0.15913677215576150D 01	0.15913597748387230D 01	-0.89467183897351920D-05
0.3000	0.19538669585181620D 01	0.19538562566108330D 01	-0.10702007330110040D-04

Fig. 2. Solución al segundo ejemplo.

Tercer ejemplo:

$$\ddot{X}-(2\tan(t))\dot{X}-10X = 0 \tag{6-7}$$

para $t \neq (2n-1)\pi/2$, $n = 1,2,3...$

con condiciones iniciales $X = 0$, $X = 3$ a $t = 0$.

Las variables de estado escojidos son

$$X_1 = X, \qquad X_2 = \dot{X}$$

Por lo tanto, obtenemos

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 10 & 2\tan(t) \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{6-8}$$

La solución de (6-7) es

$$X = .5(\mathrm{sec}t)\exp(3t)-.5(\mathrm{sec}t)\exp(-3t) \tag{6-9}$$

La solución numérica de este ejemplo está en la Fig. 3.

Cuarto ejemplo:

$$\ddot{X}+(-3+9\mu(t+.1))\dot{X}+(-4+13\mu(t+.1))X = 0 \tag{6-10}$$

donde $\mu(t-a)$ es una función de salto unitario en $t = a$.

A $t = 0$, $x = 0$, $\dot{X} = 5$

Las variables de estado son

$$X_1 = X, \quad X_2 = \dot{X}$$

entonces

INPUT DATA

0.0		0.5000000D 00	0.2000000D 00	0.0		2	400	10
0.0	3.0000000							
1.00000	0.0	0.0	0.0	0.0	0.0	0.0		
2.00000	1.00000	0.0	0.0	0.0	0.0	0.0		
2.00000	10.00000	0.0	0.0	0.0	0.0	0.0		
8.00000	2.00000	0.0	0.0	0.0	0.0	0.0		

NUMERICAL SOLUTION

H= 0.19999999999999970D-01

X	NUMERICAL APROX	TRUE VALUE	ERROR
0.0	0.0	0.0	0.0
0.0200	0.60048020595775830D-01	0.60048015683377180D-01	-0.49123986501850520D-08
0.0400	0.12038450307408150D 00	0.12038450219253420D 00	-0.88154723798528040D-09
0.0600	0.18129983428478980D 00	0.18129981763686120D 00	-0.16647928521851240D-07
0.0800	0.24308812618255610D 00	0.24308811180331390D 00	-0.14379242102569000D-07
0.1000	0.30604928731918330D 00	0.30604926499182320D 00	-0.22327360113538730D-07
0.1200	0.37049090862274170D 00	0.37049087906885970D 00	-0.29553881897292910D-07
	DOUBLED		
0.1600	0.50509810447692870D 00	0.50509696560235190D 00	-0.11388740767531690D-05
0.2000	0.64960443973541260D 00	0.64960238063680890D 00	-0.20590986035684520D-05

Fig. 3. Solución al tercer ejemplo.

$$\begin{vmatrix} \dot{X}_1 \\ \dot{X}_2 \end{vmatrix} = \begin{vmatrix} 0 & 1 \\ +4-13\mu(t+.1) & +3-9\mu(t+.1) \end{vmatrix} \begin{vmatrix} X_1 \\ X_2 \end{vmatrix} \qquad (6\text{-}11)$$

La solución de (6-10) es en dos partes. De $t = 0$ a $t = .1$ y de ..1 a infinito.

Los valores encontrados a $t = .1$ servirán como condiciones iniciales para la segunda parte. La solución de (6-10) para $0 \leq t \leq .1$ con condiciones iniciales a $t = 0$, $X = 0$, $\dot{X} = 5$, es $X = \exp(4t)-\exp(t)$, y de $t = .1$ a infinito, con condiciones iniciales a $t = .1$ de

$$X = .586987033838$$

$$\dot{X} = 6.872133255$$

la solución es

$$X = (-.372996286+11.65345777t)\exp(-3t)$$

La solución numérica de este ejemplo está en la Fig. 4.

INPUT DATA

0.0	0.5000000D 00	0.2000000D 00	0.0			2	400	10
0.0	5.0000000							
1.00000	0.0	0.0	0.0	0.0	0.0	0.0		
2.00000	1.00000	0.0	0.0	0.0	0.0	0.0		
6.00000	4.00000	-13.00000	0.10000	0.0	0.0	0.0		
6.00000	3.00000	-9.00000	0.10000	0.0	0.0	0.0		

NUMERICAL SOLUTION

H= 0.99999964237213070D-02

X	NUMERICAL APROX	TRUE VALUE	ERROR
0.0	0.0	0.0	0.0
0.0100	0.50760935610924180D-01	0.50760922013608470D-01	-0.13597315717817060D-07
0.0200	0.10308839010758670D 00	0.10308835636418470D 00	-0.33743402072516910D-07
0.0300	0.15705130931564780D 00	0.15705125923220200D 00	-0.50083445835014120D-07
0.0400	0.21272134780383790D 00	0.21272135094645490D 00	0.31376170550601040D-08
0.0500	0.27017319202423100D 00	0.27017322928861810D 00	0.37264387131585860D-07
0.0600	0.32948440313339230D 00	0.32948448741689610D 00	0.84283503731796820D-07
DOUBLED			
0.0800	0.45401114225387570D 00	0.45401123393899360D 00	0.91685117942219490D-07
0.1000	0.58698695898056030D 00	0.58698703383858270D 00	0.74858022447443730D-07
0.1000	0.58698695898056030D 00	0.58698669431402630D 00	-0.26466653392509510D-06
0.1200	0.71541015972616150D 00	0.71541005709487240D 00	-0.10263128799592440D-06
0.1400	0.82688513642642650D 00	0.82688514878665260D 00	0.12360226328556020D-07
0.1600	0.92295029576052910D 00	0.92295041886839750D 00	0.12310786841329670D-06
0.1800	0.10050220433501950D 01	0.10050226100459600D 01	0.56109577029062030D-06
0.2000	0.10744047164816970D 01	0.10744057161014250D 01	0.99960973143353610D-06

Fig. 4. Solución al cuarto ejemplo.

VII. CONCLUSIONES

En el método numérico propuesto, el incremento puede ser disminuído a cualquier cantidad deseada. La única limitación en el número de ecuaciones diferenciales simultáneas que se pueden resolver, depende solamente en la capacidad de la computadora y el único cambio que se debe hacer es en la sentencia "COMMON".

Si la solución va a infinito para algún tiempo t, el programa comenzará a acortar el incremento indefinidamente. Para resolver este problema en el programa, se puede insertar una prueba que comparará el valor del incremento con un valor mínimo especificado. Si el valor del incremento es menos a este valor especificado, la solución asumirá un valor grande y continuará. En el caso de funciones de salto como parámetros del sistema, el método usado en este programa es encontrar el tiempo más cercano que la función de salto ocurrirá y figar este tiempo como un punto que la solución debe alcanzar. Una vez alcanzado este tiempo pasa a un nuevo punto de salto al punto final de la solución.

El método permite el obtener una solución para todas las variables de estado sin el conocimiento de la Matriz de Transición, que muchas veces es difícil de obtener.

El método numérico propuesto también se puede aplicar para el caso de parámetros constantes. Pero en este caso la Matriz de Transición se requiere de bastante trabajo. Por ejemplo, en el método de Sylvester, primero tenemos que encontrar los valores de eigen de la matriz A, esto ya es un problema para sistemas grandes y ni se diga si los valores de eigen son bastante cercanos. En el método de $\emptyset = L^{-1}(SI-A)^{-1}$ la inversión de la matriz (SI-A) es también laboriosa ya que contiene términos algebraicos en S. El método de encontrar la Matriz de Transición por inspección de algún diagrama de variables de estado es también difícil.

El método numérico propuesto también admite algunos problemas no en el que debemos tener cuidado en no dividir por cero en ningún momento. punto en contra del método propuesto es que la solución se alcanza por incrementos de la variable independiente hasta un punto final, este punto final puede estar bastante alejado de las condiciones iniciales por lo cual tenemos bastante tiempo en su solución. En cambio, por el método de la Matriz de Transición, lo único que debemos hacer es sustituir los valores de t en multiplicar por las condiciones iniciales.

Uno podría desarrollar un método numérico para encontrar la Matriz de Transición después de tener la solución numérica, aunque esto no parece muy práctico ya que la Matriz de Transición es diferente para cada t.

VIII. BIBLIOGRAFIA

1. Brown, Robert G. and Nilsson, J. W. Introduction to Linear Systems Analysis. John Wiley and Sons, Inc., New York, New York. 1962.

2. Coddington, Earl A. Theory of Ordinary Differential Equations. McGraw-Hill Book Co., Inc., New York, New York. 1955.

3. Crane, Roger L. Stability and Local Accuracy of Numerical Methods for Ordinary Differential Equations. Unpublished Ph.D. Thesis. Biblioteca de Iowa State University, Ames, Iowa. 1962.

4. DeRusso, P. M. and Roy, R. J. State Variables for Engineers, John Wiley and Sons, Inc., New York, New York. 1965.

5. Gill, S. A Process for the Step-by-Step Integration of Differential Equations in an Automatic Digital Computing Machine. Cambridge Philosophical Society Proceedings 47:96. 1951.

6. Gupta, S. C. Transform and State Variable Methods in Linear Systems. John Wiley and Sons, Inc., New York, New York. 1966.

7. Henrici, P. Elements of Numerical Analysis. John Wiley and Sons, Inc., New York, New York. 1962.

8. Henrici, P. Discrete Variable Methods in Ordinary Differential Equations. John Wiley and Sons, Inc., New York, New York. 1962.

9. Ince, E. L. Ordinary Differential Equations. Dover Publications, New York, New York. 1956.

10. IRE International Convention Record. Symposium on Time-Varying Networks. Páginas 251-277. New York, New York. 1961.

11. Kaplan, W. Operational Methods for Linear Systems. Addison-Wesley, Reading, Massachusetts. 1962.

12 Ogata, K. State Space Analysis of Control Systems. Prentice-Hall, Inc., Englewood Cliffs, New Jersey. 1967.

13. Pipes, L. A. Four Methods for the Analysis of Time-Variable Circuits. IRE Transactions on Circuit Theory CT-2:4. Marzo, 1955.

14. Pipes, L. A. Matrix Methods for Engineering. Prentice-Hall, Inc., Englewood Cliffs, New Jersey. 1963.

15. Ralston, A. A First Course in Numerical Analysis. McGraw-Hill Book Co., Inc., New York, New York. 1965.

16. Schwarz, R. S. Linear Systems. McGraw-Hill Book Co., Inc., New York, New York. 1965.

17. Stubberud, Allen R. Analysis and Synthesis of Linear Time-Variable Systems. University of California Press, Berkeley, California. 1964.

18. Tou, Julius T. Modern Control Theory. McGraw-Hill Book Co., Inc., New York, New York. 1964.

19. Tenenbaum, M. and Pollard, H. Ordinary Differential Equations. Harper and Row Publishers, Inc., New York, New York, 1963.

20. Zadeh, L. A. and Desoer, Charles A. Linear System Theory. McGraw-Hill Book Co., Inc., New York, New York, 1963.

IX. APENDICE

NOTA

La persona que desea correr este programa debe copiar este apéndice, omitiendo las tarjetas marcadas *** en las columnas 73, 74 y 75. Estas tarjetas son utilizadas para verificar la solución de los ejemplos en el capítulo VI.

```
C      MAIN PROGRAM
       COMMON COE(8,8),VAR(14,50),A(4),B(4),C(4),H,BYHAL,ENDVA,FLAG,X,
      1CALC,ERROR,DVA(8,8,7),N,T,PHSIG,NODUB,ENDNO,NO
       DOUBLE PRECISION COE,VAR,A,B,C,H,BYHAL,ENDVA,FLAG,X,FIN,CALC,ERROR
       INTEGER ENDNO,T,PHSIG
       NO=0
    60 NO=NO+1                                                        ***
       WRITE(3,15)
    15 FORMAT('0',35X,'INPUT  DATA'//)
       READ(1,1)H,BYHAL,ENDVA,X,N,T,PHSIG,NODUB,ENDNO
       WRITE(3,1)H,BYHAL,ENDVA,X,N,T,PHSIG,NODUB,ENDNO
     1 FORMAT(4D14.7,I4,I2,2I1,I5)
       READ(1,50)(VAR(1,I),I=1,N)
       WRITE(3,50)(VAR(1,I),I=1,N)
    50 FORMAT(8F10.7)
       DO 55 I=1,N
       DO 55 J=1,N
       READ(1,81)(DVA(I,J,K),K=1,7)
    55 WRITE(3,81)(DVA(I,J,K),K=1,7)
    81 FORMAT(8F10.5)
       WRITE(3,16)
    16 FORMAT('0',35X,' NUMERICAL SOLUTION'//)
       FIN=ENDVA
    40 ENDVA=FIN
       DO 31 I=1,N
       DO 31 J=1,N
       IF(DVA(I,J,1)-6.)31,30,31
    30 IF(ENDVA-DVA(I,J,4))31,31,29
    29 ENDVA=DVA(I,J,4)
    31 CONTINUE
       CALL NODIN
       IF(FIN-ENDVA-0.00001)32,32,28
    28 DO 34 I=1,N
       DO 34 J=1,N
       IF(DVA(I,J,1)-6.)34,35,34
```

```
   35 IF(ENDVA-DVA(I,J,4))34,36,34
   36 DVA(I,J,2)=DVA(I,J,2)+DVA(I,J,3)
      DVA(I,J,4)=FIN
   34 CONTINUE
      NO=NO+1                                                    ***
      GO TO 40
   32 CONTINUE
      IF(NO-5)60,61,61                                           ***
   61 CONTINUE                                                   ***
      STOP
      END
```

```
      SUBROUTINE PCR
      COMMON COE(8,8),VAR(14,50),A(4),B(4),C(4),H,BYHAL,ENDVA,FLAG,X,
     1CALC,ERROR,DVA(8,8,7),N,T,PHSIG,NODUB,ENDNO,NO
      DOUBLE PRECISION COE,VAR,A,B,C,H,BYHAL,ENDVA,FLAG,X,FIN,CALC,ERROR
      INTEGER ENDNO,T,PHSIG
      DO 70 I=1,N
      VAR(8,I)=0.
      DO 70 J=1,N
   70 VAR(8,I)=COE(I,J)*VAR(1,J)+VAR(8,I)
      RETURN
      END
```

```
C  WRITE-OUT ROUTINE -- USER'S RESPONSIBILITY
      SUBROUTINE ESCRIB
      COMMON COE(8,8),VAR(14,50),A(4),B(4),C(4),H,BYHAL,ENDVA,FLAG,X,
     1CALC,ERROR,DVA(8,8,7),N,T,PHSIG,NODUB,ENDNO,NO
      DOUBLE PRECISION COE,VAR,A,B,C,H,BYHAL,ENDVA,FLAG,X,FIN,CALC,ERROR
      INTEGER ENDNO,T,PHSIG
      GO TO (1,2,3,4,5),NO                                              ***
    1 CALC=-28.0D0/DEXP(X)+64./(X+2.)                                   ***
      GO TO 6                                                           ***
    2 ERROR=DEXP(X)                                                     ***
      CALC =-.05*DEXP(ERROR)+0.0676675*DEXP(3.*ERROR)                   ***
      GO TO 6                                                           ***
    3 CALC=(DEXP(3.*X)-1./DEXP(3.*X))/(2.*DCOS(X))                      ***
      GO TO 6                                                           ***
    4 CALC=DEXP(4.*X)-1.0D0/DEXP(X)                                     ***
      GO TO 6                                                           ***
    5 CALC=(-.37296286+11.65345777*X)/DEXP(3.*X)                        ***
    6 ERROR=CALC-VAR(1,1)                                               ***
      WRITE(3,15)X,VAR(1,1),CALC,ERROR
   15 FORMAT(F10.4,3E25.17)
      IF(X-ENDVA+0.0001)10,11,11
   11 FLAG=-1.D0
   10 RETURN
      END
```

```
      SUBROUTINE CCMPD
      COMMON COE(8,8),VAR(14,50),A(4),B(4),C(4),H,BYHAL,ENDVA,FLAG,X,
     1CALC,ERROR,DVA(8,8,7),N,T,PHSIG,NODUB,ENDNO,NO
      DOUBLE PRECISION COE,VAR,A,B,C,H,BYHAL,ENDVA,FLAG,X,FIN,CALC,ERROR
      INTEGER ENDNO,T,PHSIG
      DO 80 I=1,N
      DO 80 J=1,N
      NNN=DVA(I,J,1)
      GO TO (1,2,3,4,5,6,7,8,9,10),NNN
    1 COE(I,J)=0.
      GO TO 80
    2 COE(I,J)=DVA(I,J,2)
      GO TO 80
    3 COE(I,J)=DVA(I,J,2)+DVA(I,J,3)*X+DVA(I,J,4)*X*X+DVA(I,J,5)/X+
     1DVA(I,J,6)/(X*X)
      GO TO 80
    4 COE(I,J)=(DVA(I,J,2)+DVA(I,J,3)*X+DVA(I,J,4)*X*X)/(DVA(I,J,5)+
     1DVA(I,J,6)*X+DVA(I,J,7)*X*X)
      GO TO 80
    5 COE(I,J)=DVA(I,J,2)+DVA(I,J,3)/DEXP(DVA(I,J,4)*X)
      GO TO 80
    6 COE(I,J)=DVA(I,J,2)
      GO TO 80
    7 GO TO 80
    8 COE(I,J)=DVA(I,J,2)*DSIN(X)/DCOS(X)
      GO TO 80
    9 GO TO 80
   10 GO TO 80
   80 CONTINUE
      CALL POR
      RETURN
      END
```

```
      SUBRCUTINE NCDIN
      COMMON COE(8,8),VAR(14,50),A(4),B(4),C(4),H,BYHAL,ENDVA,FLAG,X,
     1CALC,ERROR,DVA(8,8,7),N,T,PHSIG,NODUB,ENDNO,NO
      DOUBLE PRECISION COE,VAR,A,B,C,H,BYHAL,ENDVA,FLAG,X,FIN,CALC,ERROR
      INTEGER ENDNO,T,PHSIG
      NOD=0
      CHECK1 = 16.219658/(10**T)
      CHECK2 = CHECK1 / 200.D0
      A(1)=.5D0
      A(2)=.2928932188134525
      A(3)=1.707106781186547
      A(4)=.1666666666666667
      B(1)=1.D0
      B(2)=A(2)
      B(3)=A(3)
      B(4)=.3333333333333330
      C(1)=.5D0
      C(2)=A(2)
      C(3)=A(3)
      C(4)=.5D0
      FLAG=0.D0
      IF (ENDNO)503,504,503
  503 H=(ENDVA-X)/ENDNO
      WRITE(3,1) H
    1 FORMAT(5X,'H=',2X,E25.17)
      WRITE(3,17)
   17 FORMAT('0',5X,'X',6X,'NUMERICAL APROX ',16X,'TRUE VALUE',15X,'ERRO
     1R'//)
  504 CALL COMPD
      CALL ESCRIB
  505 CONTINUE
C   OBTAIN STARTING POINTS WITH RUNGE-KUTTA-GILL METHOD
  402 DO 402 I=1,N
  402 VAR(6,I)=0.D0
      J=4
```

```
  410 DO 408 I=1,N
      VAR(J+1,I)=VAR(1,I)
  408 VAR(J+8,I)=VAR(8,I)
      J=J-1
      IF(J) 420,420,403
  403 DO 407 K=1,4
      DO 404 I=1,N
      CK=H*VAR(8,I)
      R=(A(K)*CK)-(B(K)*VAR(6,I))
      VAR(1,I)=VAR(1,I)+R
  404 VAR(6,I)=VAR(6,I)+(3.*R)-(C(K)*CK)
      IF (K-1) 405,405,413
  413 IF (K-3)406,405,406
  405 X=X+(H/2.D0)
      CALL COMPD
      GO TO 407
  406 CALL PCR
  407 CONTINUE
      GO TO 410
  420 NOD=-1
      NSWHE=1
      IF (ENDNO)507,508,507
  507 ENDNO=ENDNO-3
  508 M=3
  509 FLAG= 0.D0
      X=X+H
C     PREDICT  Y-VALUE
      DO 450 I=1,N
  450 VAR(1,I)=(1.5476511 *VAR(2,I))-(1.8675052 *VAR(3,I))+(2.01720690*
     1VAR(4,I))-(.69735280000000 *VAR(5,I))+H*((2.002247216666667  *
     2VAR(9,I))-(2.03168765D0     *VAR(10,I))+(1.8186106 5D0        *
     3VAR(11,I))-(.7143200166666667 *VAR(12,I)))
      CALL COMPD
      PERR=0.D0
C     CORRECT Y-VALUE
```

```
      DO 462 I=1,N
      TEMP=VAR(2,I)+H*((.375*VAR(8,I))+(.791666666666667 *
     1 VAR(9,I))-(.208333333333333 *VAR(10,I))+
     2 (.041666666666666670*VAR(11,I)))
      IF (PHSIG)463,464,463
  463 TEMPA=DABS((TEMP-VAR(1,I))/TEMP)
      GO TO 465
  464 TEMPA=DABS(TEMP-VAR(1,I))
  465 VAR(1,I)=TEMP
      IF (PERR-TEMPA)461,462,462
  461 PERR=TEMPA
  462 CONTINUE
      CALL PCR
      IF (PERR-CHECK1) 517,517,535
C     NO HALVING NECESSARY
  517 NSWHF=0
      IF(MOD)550,518,518
  518 IF (ENDNO)519,520,519
  519 ENDNO=ENDNO-1
  520 CALL ESCRIB
      IF (FLAG)560,521,521
C     IS DOUBLING POSSIBLE
  521 IF (PERR- CHECK2 )525,525,522
  522 M=3
  528 J=13
  523 DO 524 I=1,N
  524 VAR(J+1,I)=VAR(J,I)
      J=J-1
      IF (J)509,509,523
C     DOUBLING
  525 IF(NODUB)522,526,522
  526 M=M-1
      IF (M)530,527,528
  527 IF (ENDNO)530,531,530
  530 MOD=ENDNO/2
```

```
      MOD= ENDNO-MOD*2
      IF(MOD)528,531,528
  531 WRITE(3,3)
    3 FORMAT('0',10X,'DOUBLED')
      DO 533 I=1,N
      VAR(2,I)=VAR(1,I)
      VAR(4,I)=VAR(5,I)
      VAR(5,I)=VAR(7,I)
      VAR(9,I)=VAR(8,I)
      VAR(11,I)=VAR(12,I)
  533 VAR(12,I)=VAR(14,I)
      H=2.D0*H
      IF (ENDNO)534,508,534
  534 ENDNO=ENDNO/2.
      GO TO 508
C     HALVING
  535 FLAG=DABS(BYHAL)
      WRITE(3,2)
    2 FORMAT('0',10X,'HALVED')
  548 IF (ENDNO)543,542,543
  543 ENDNO=2.*ENDNO
  542 IF (NSWHF)538,540,538
C     REPEATED HALVING
  538 DO 539 I=1,N
      VAR(1,I)=VAR(5,I)
  539 VAR(8,I)=VAR(12,I)
      X=X-(4.D0*H)
      IF (ENDNO)549,549,544
  544 ENDNO=ENDNO+6.
  549 H=H*DABS(BYHAL)
      GO TO 506
  540 DO 541 I=1,N
      VAR(1,I)=VAR(2,I)
```

```
  541 VAR(8,I)=VAR(9,I)
      X=X-H
      GO TO 549
  550 X=X-(3.D0*H)
      IF (ENDNO)551,552,551
  551 ENDNO=ENDNO+2.
  552 K=3
      DO 553 I=1,N
      VAR(6,I)=VAR(1,I)
  553 VAR(13,I)=VAR(8,I)
  557 DO 554 I=1,N
      VAR(1,I)=VAR(K+1,I)
  554 VAR(8,I)=VAR(K+8,I)
      CALL ESCRIB
      IF (FLAG)560,562,562
  560 RETURN
  562 X=X+H
      K=K-1
      IF (K)558,558,555
  555 IF (ENDNO)556,557,556
  556 ENDNO=ENDNO-1.
      GO TO 557
  558 DO 559 I=1,N
      VAR(1,I)=VAR(6,I)
  559 VAR(8,I)=VAR(13,I)
      NCD=0
      GO TO 518
      END
```

B. Formatos

La primera carta de entrada especifíca:

H	Incremento inicial de la variable independiente.

BYHAL	1 indica no partimiento ($h \leftarrow h/2$), .5 indica $h \leftarrow h/2$ ó .S si uno quiere reducir a Sh.

ENDVA	Valor final de la variable independiente.

X	Valor inicial de la variable independiente.

N	Número de ecuaciones del sistema.

T	El número de dígitos decimales de aproximación que se quiere (1 a 12 en "Double precision").

PHSIG	Comprobación de error: 0 para error absoluto y 1 para error relativo.

El error es aproximado por la diferencia entre el valor predecido (P_{n+1}) y entre el valor corregido (C_{n+1}). El error absoluto es $E = |P_{n+1} - C_{n+1}|$. El error relativo es $E = \left|\frac{P_{n+1} - C_{n+1}}{C_{n+1}}\right|$.

NODUB	Cualquier número excepto cero para suprimir el incremento de h a 2h.

ENDNO	Número de puntos en el intervalo de integración si un punto final debe ser alcanzado. En este caso, H no necesita ser especificado.

Descripción	Columnas de la Tarjeta	Formato
H	1-14	D14.7
BYHAL	15-28	D14.7
ENDVA	29-42	D14.7
X	43-56	D14.7
N	57-60	I4
T	61-62	I2
PHSIG	63	I1
NODUB	64	I1
ENDNO	65-69	I5

La segunda tarjeta de entrada especifica las condiciones iniciales del sistema de ecuaciones diferenciales y se almacena en la primera fila de la Matriz VAR. Cada una de las condiciones iniciales está en el Formato F10.7 y un máximo de ocho es permitido.

Los elementos de la Matriz A son datos, un elemento por tarjeta en el siguiente órden a_{11}, a_{12}, ..., a_{1n}, a_{21}, a_{22}, ..., a_{2n}, ..., a_{n1}, a_{n2}, ..., a_{nn}. Cada elemento (I,J) de la Matriz A tiene una tarjeta correspondiente que especifíca la función quiere decir las constantes que caracterizan esta función y están almacenados en DVA(I,J,K), K = 1,7.

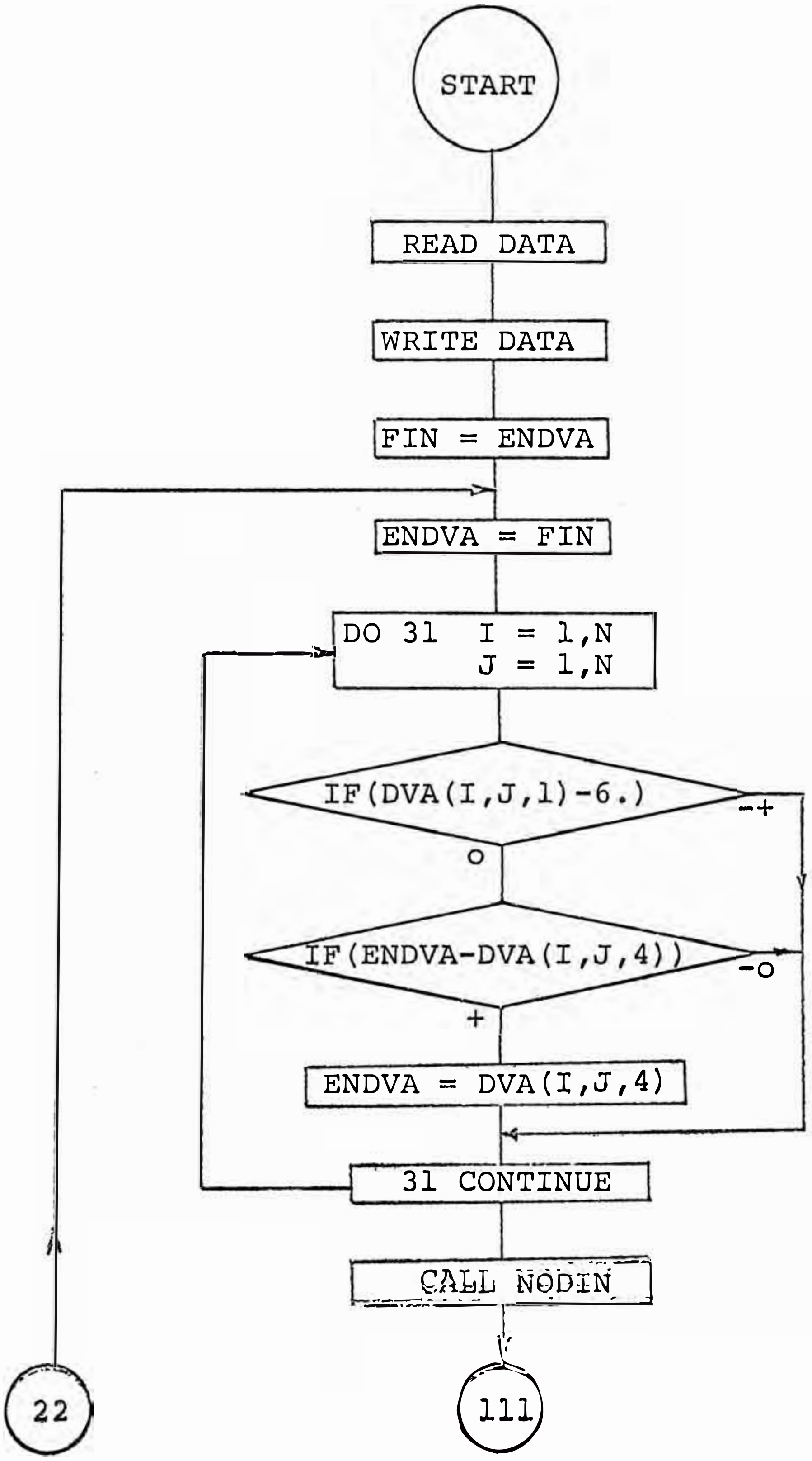
Para cada tarjeta, tenemos

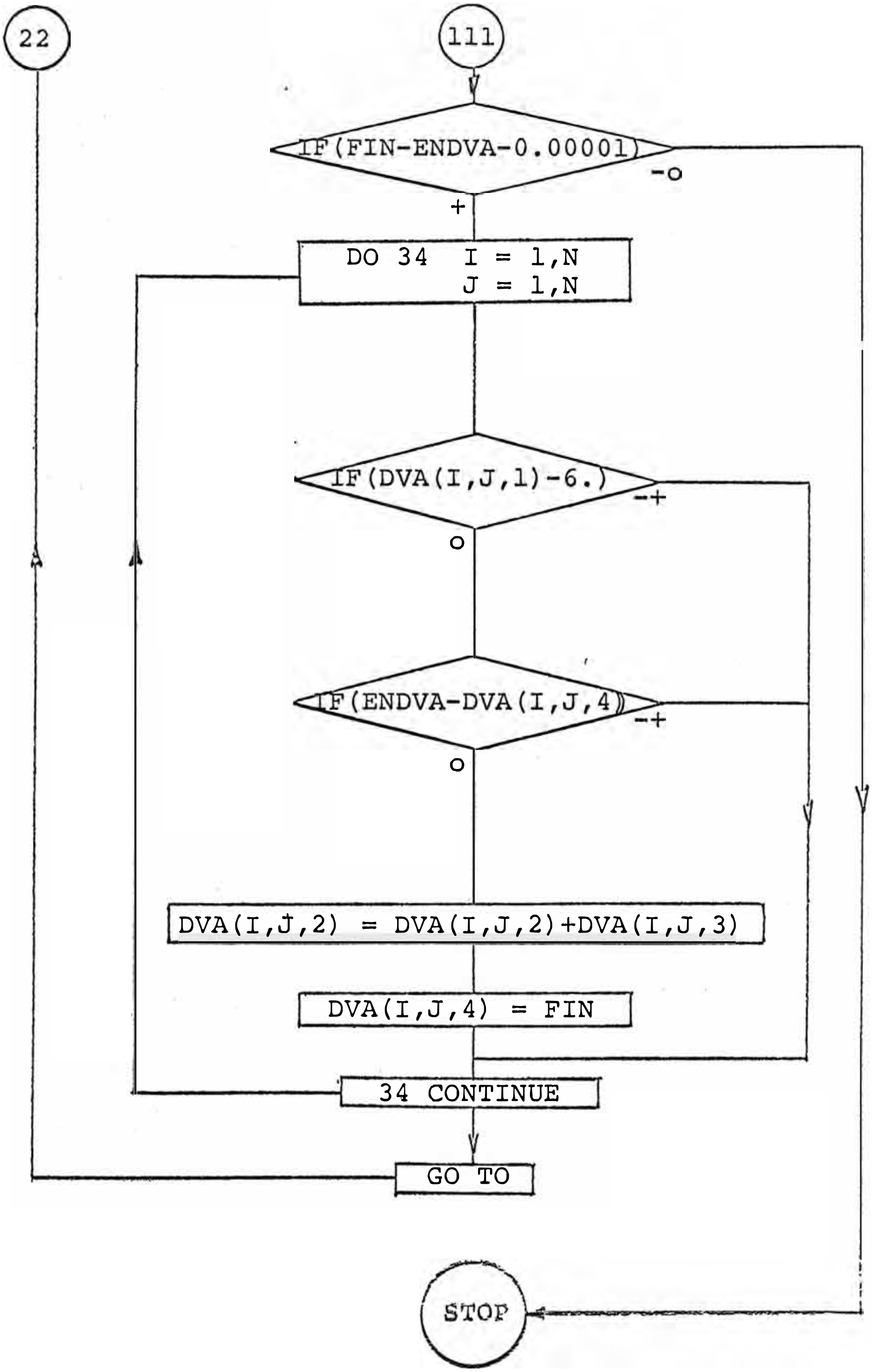
Descripción	Columnas de la Tarjeta	Formato	DVA(I,J,K)
Tipo de función	1-10	F10.5	DVA(I,J,1)
A	11-20	F10.5	DVA(I,J,2)
B	21-30	F10.5	DVA(I,J,3)
C	31-40	F10.5	DVA(I,J,4)
D	41-50	F10.5	DVA(I,J,5)
	51-60	F10.5	DVA(I,J,6)
F	61-70	F10.5	DVA(I,J,7)

Los siguientes son los tipos de funciones usados en el programa:

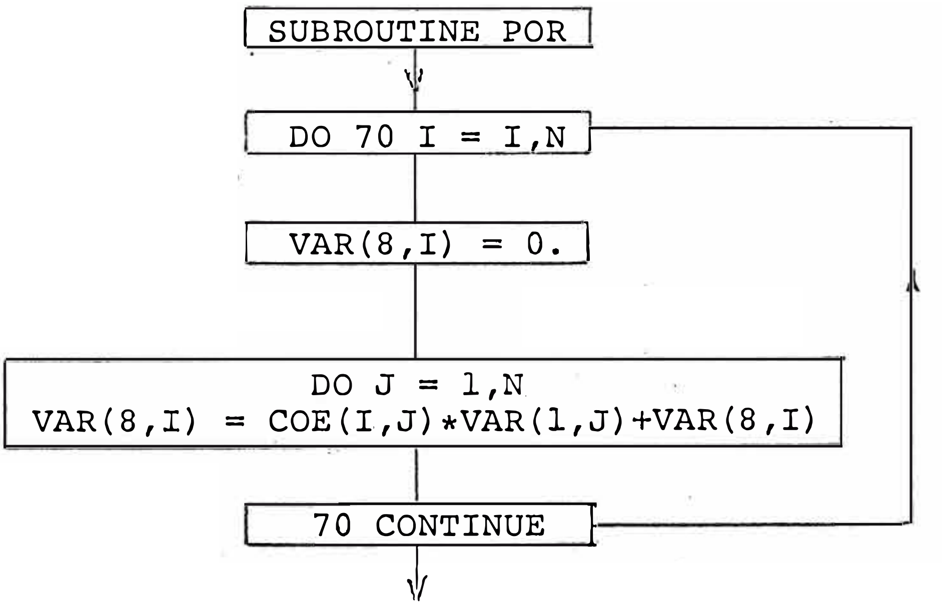
Tipo	Función
	$f(t) = 0$
2	$f(t) = A$
3	$f(t) = A+Bt+Ct^2+Dt^{-1}+Et^{-2}$
4	$f(t) = A+Bt+Ct^2/D+Et+Ft^2$
5	$f(t) = A+Be^{-Ct}$
6	$f(t) = A+B\mu(t+C)$
8	f(t) = A [illegible] (t)

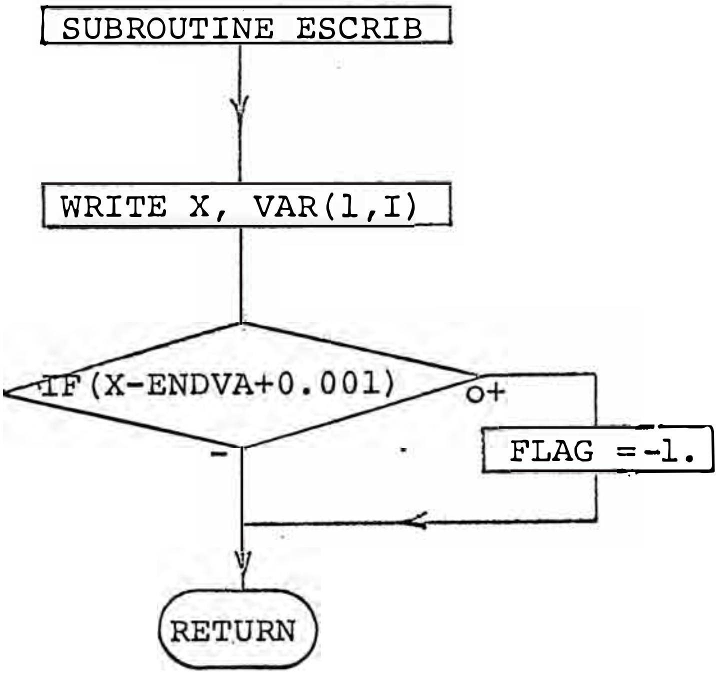
C. Diagrama de Flujo

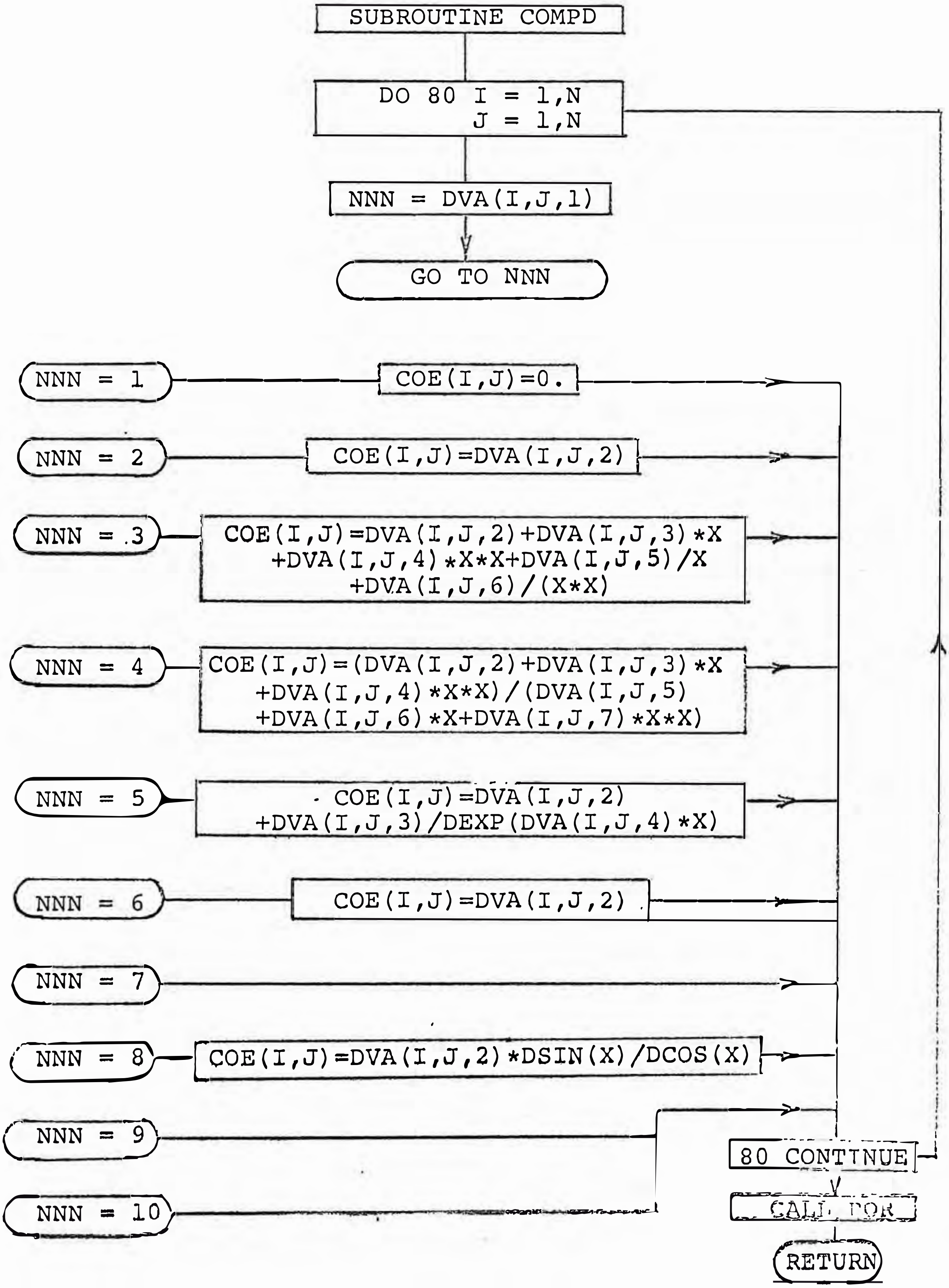


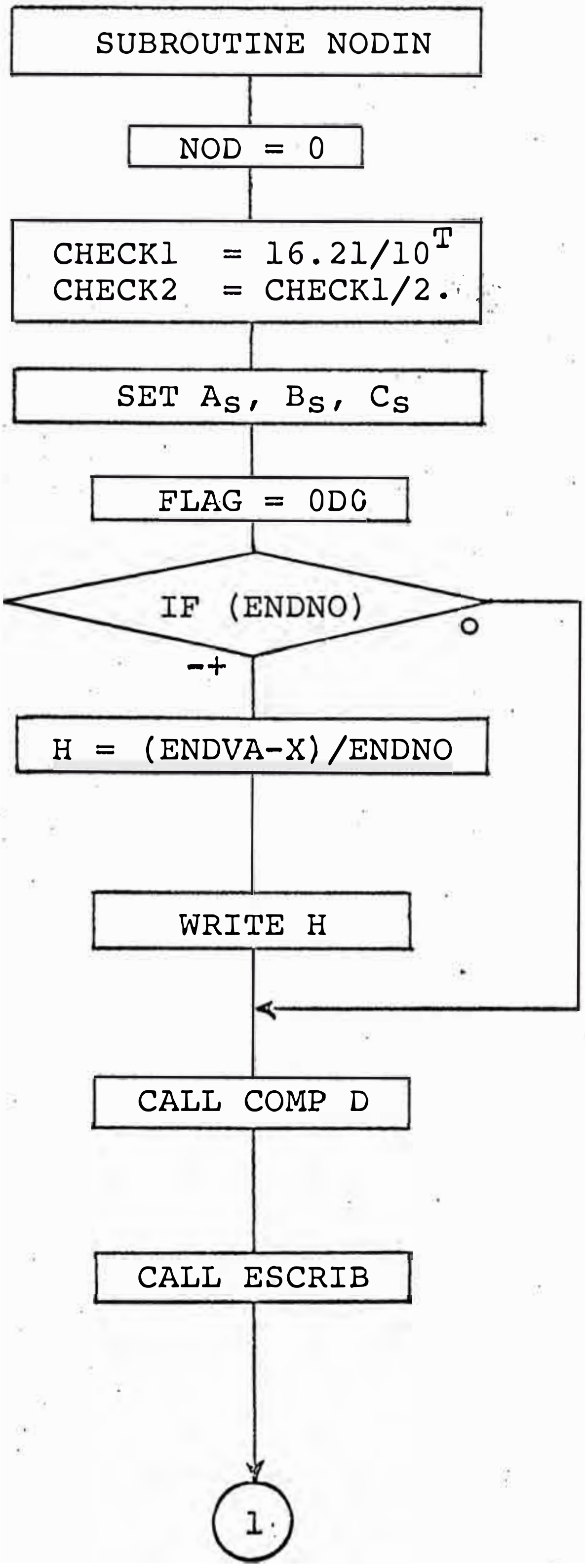


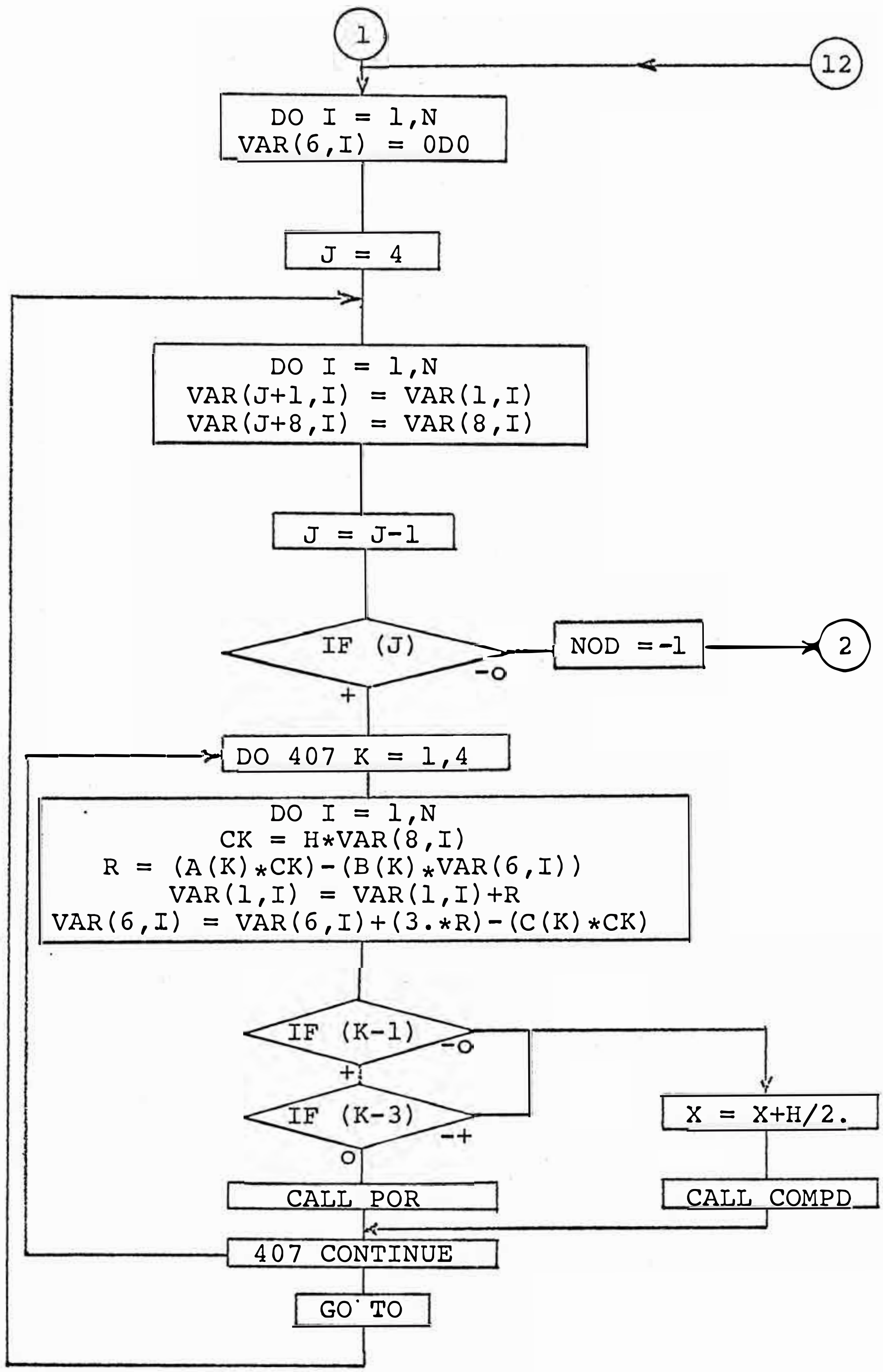
22
111
IF(FIN-ENDVA-0.00001)
-o
+
DO 34 I = 1,N
J = 1,N
IF(DVA(I,J,1)-6.)
-+
o
IF(ENDVA-DVA(I,J,4))
-+
o
DVA(I,J,2) = DVA(I,J,2)+DVA(I,J,3)
DVA(I,J,4) = FIN
34 CONTINUE
GO TO
STOP

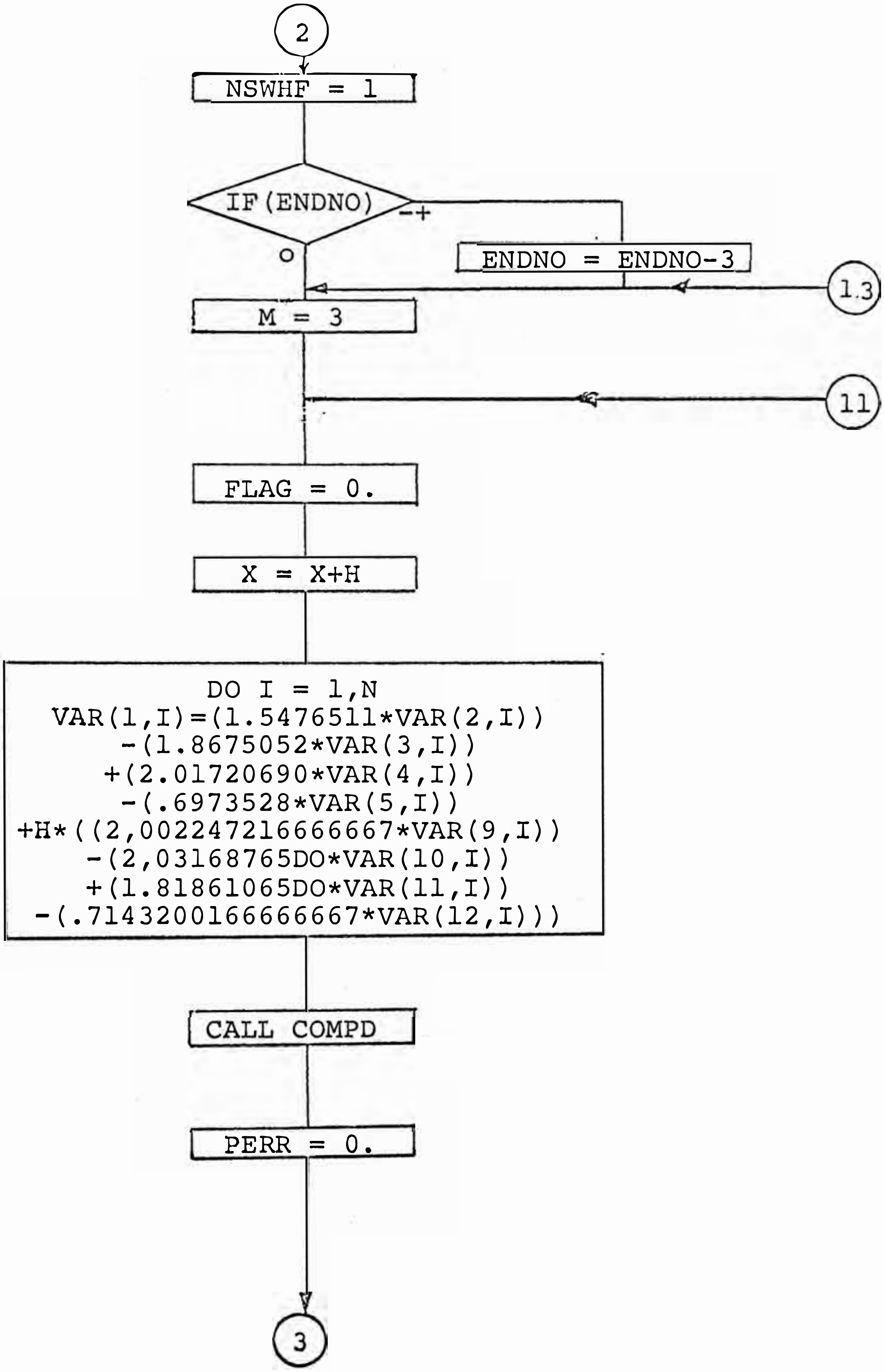

SUBROUTINE POR
DO 70 I = I,N
VAR(8,I) = 0.
DO J = 1,N
VAR(8,I) = COE(I,J)*VAR(1,J)+VAR(8,I)
70 CONTINUE

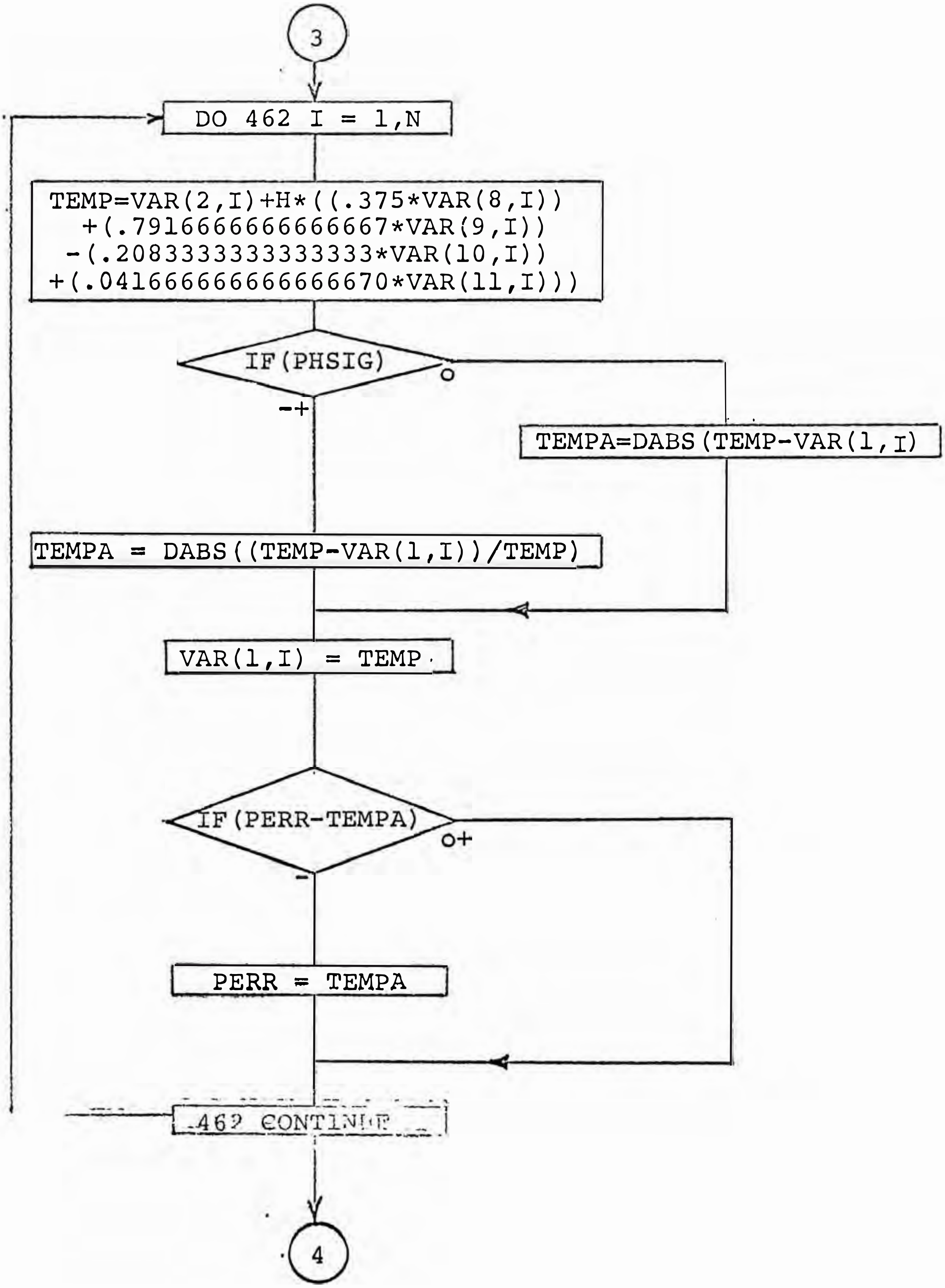

SUBROUTINE ESCRIB
WRITE X, VAR(1,I)
IF(X-ENDVA+0.001)
0+
-
FLAG =-1.
RETURN


SUBROUTINE COMPD
DO 80 I = 1,N
J = 1,N
NNN = DVA(I,J,1)
GO TO NNN
NNN = 1
COE(I,J)=0.
NNN = 2
COE(I,J)=DVA(I,J,2)
NNN = 3
COE(I,J)=DVA(I,J,2)+DVA(I,J,3)*X
+DVA(I,J,4)*X*X+DVA(I,J,5)/X
+DVA(I,J,6)/(X*X)
NNN = 4
COE(I,J)=(DVA(I,J,2)+DVA(I,J,3)*X
+DVA(I,J,4)*X*X)/(DVA(I,J,5)
+DVA(I,J,6)*X+DVA(I,J,7)*X*X)
NNN = 5
COE(I,J)=DVA(I,J,2)
+DVA(I,J,3)/DEXP(DVA(I,J,4)*X)
NNN = 6
COE(I,J)=DVA(I,J,2)
NNN = 7
NNN = 8
COE(I,J)=DVA(I,J,2)*DSIN(X)/DCOS(X)
NNN = 9
NNN = 10
80 CONTINUE
CALL FOR
RETURN

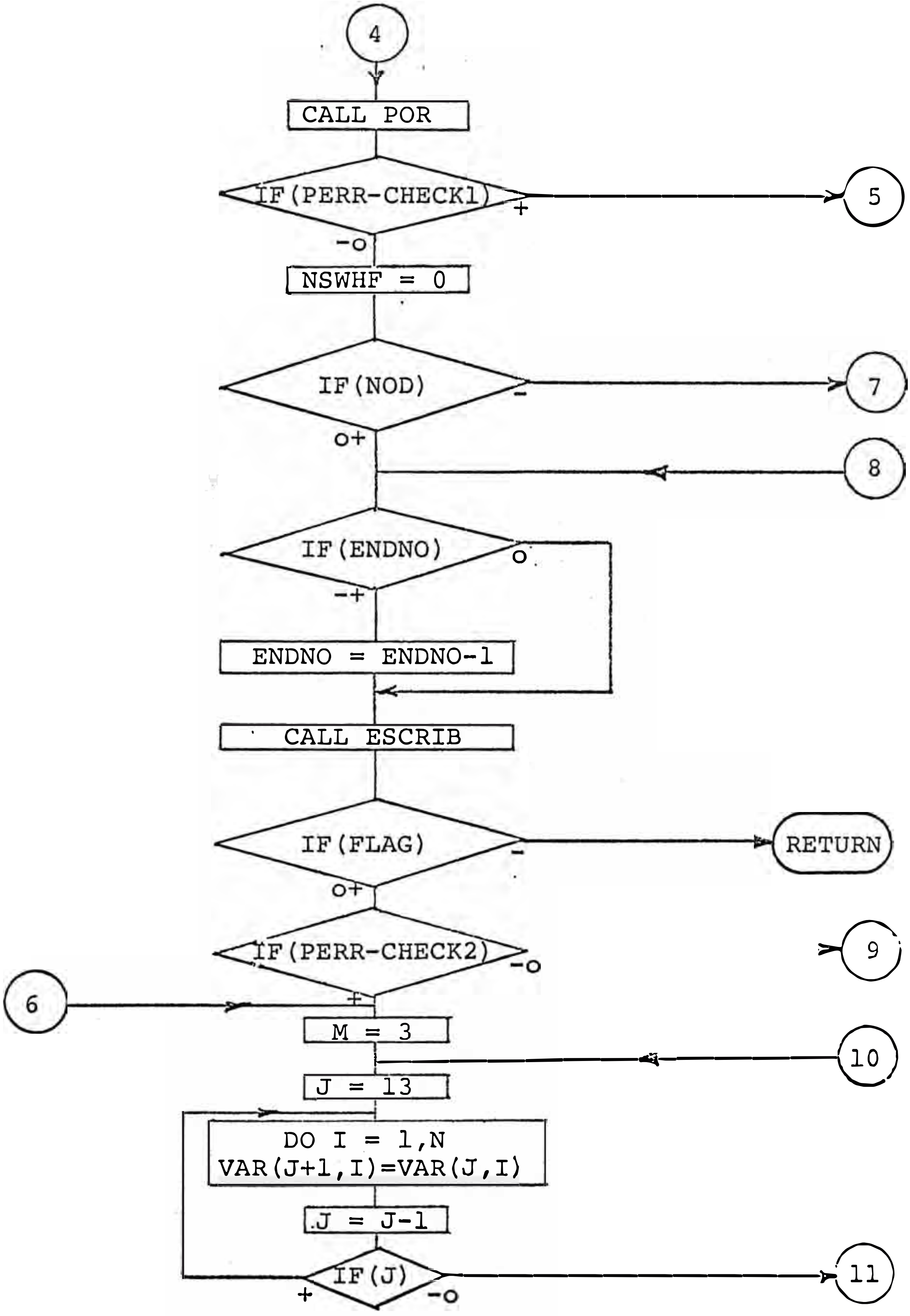

SUBROUTINE NODIN
NOD = 0
CHECK1 = 16.21/10T
CHECK2 = CHECK1/2.
SET As, Bs, Cs
FLAG = 0DC
IF (ENDNO)
0
-+
H = (ENDVA-X)/ENDNO
WRITE H
CALL COMP D
CALL ESCRIB
1

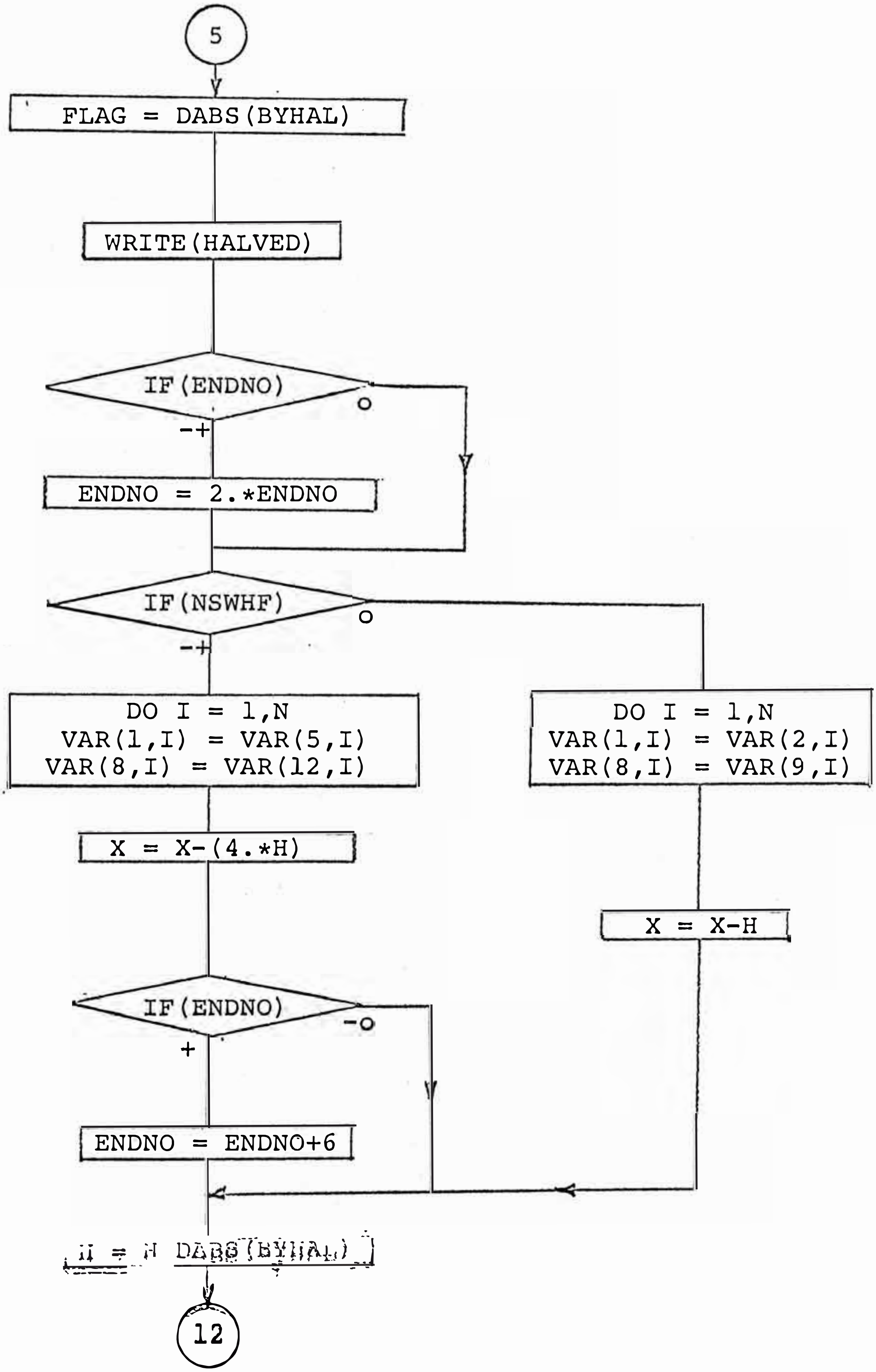

1
12
DO I = 1,N
VAR(6,I) = 0D0
J = 4
DO I = 1,N
VAR(J+1,I) = VAR(1,I)
VAR(J+8,I) = VAR(8,I)
J = J-1
IF (J)
-o
+
NOD =-1
2
DO 407 K = 1,4
DO I = 1,N
CK = H*VAR(8,I)
R = (A(K)*CK)-(B(K)*VAR(6,I))
VAR(1,I) = VAR(1,I)+R
VAR(6,I) = VAR(6,I)+(3.*R)-(C(K)*CK)
IF (K-1)
-o
+
IF (K-3)
-+
o
X = X+H/2.
CALL POR
CALL COMPD
407 CONTINUE
GO TO

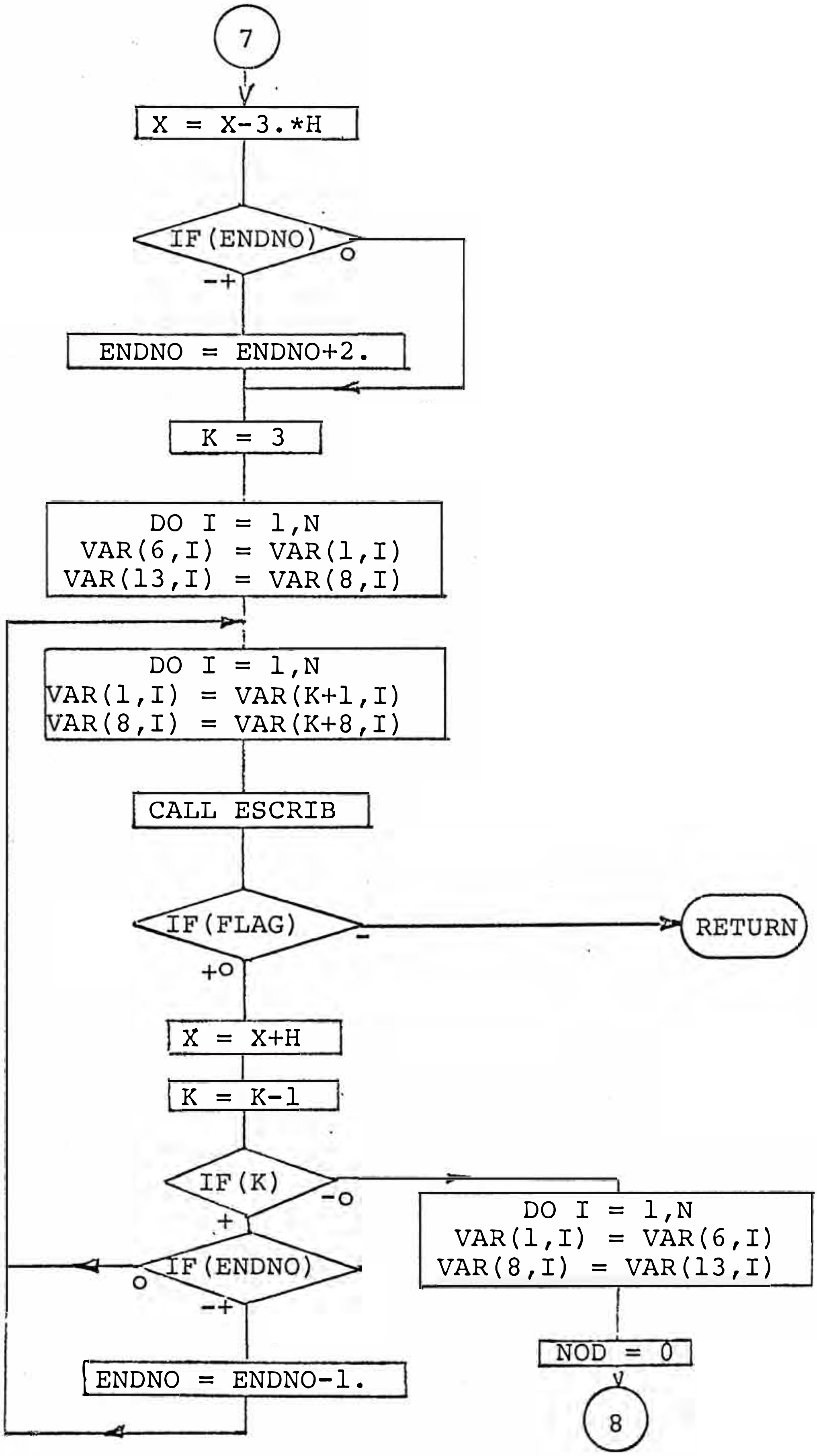

2
NSWHF = 1
IF(ENDNO)
−+
O
ENDNO = ENDNO-3
1.3
M = 3
11
FLAG = 0.
X = X+H
DO I = 1,N
VAR(1,I)=(1.5476511*VAR(2,I))
-(1.8675052*VAR(3,I))
+(2.01720690*VAR(4,I))
-(.6973528*VAR(5,I))
+H*((2,0022472166666667*VAR(9,I))
-(2,03168765DO*VAR(10,I))
+(1.81861065DO*VAR(11,I))
-(.7143200166666667*VAR(12,I)))
CALL COMPD
PERR = 0.
3

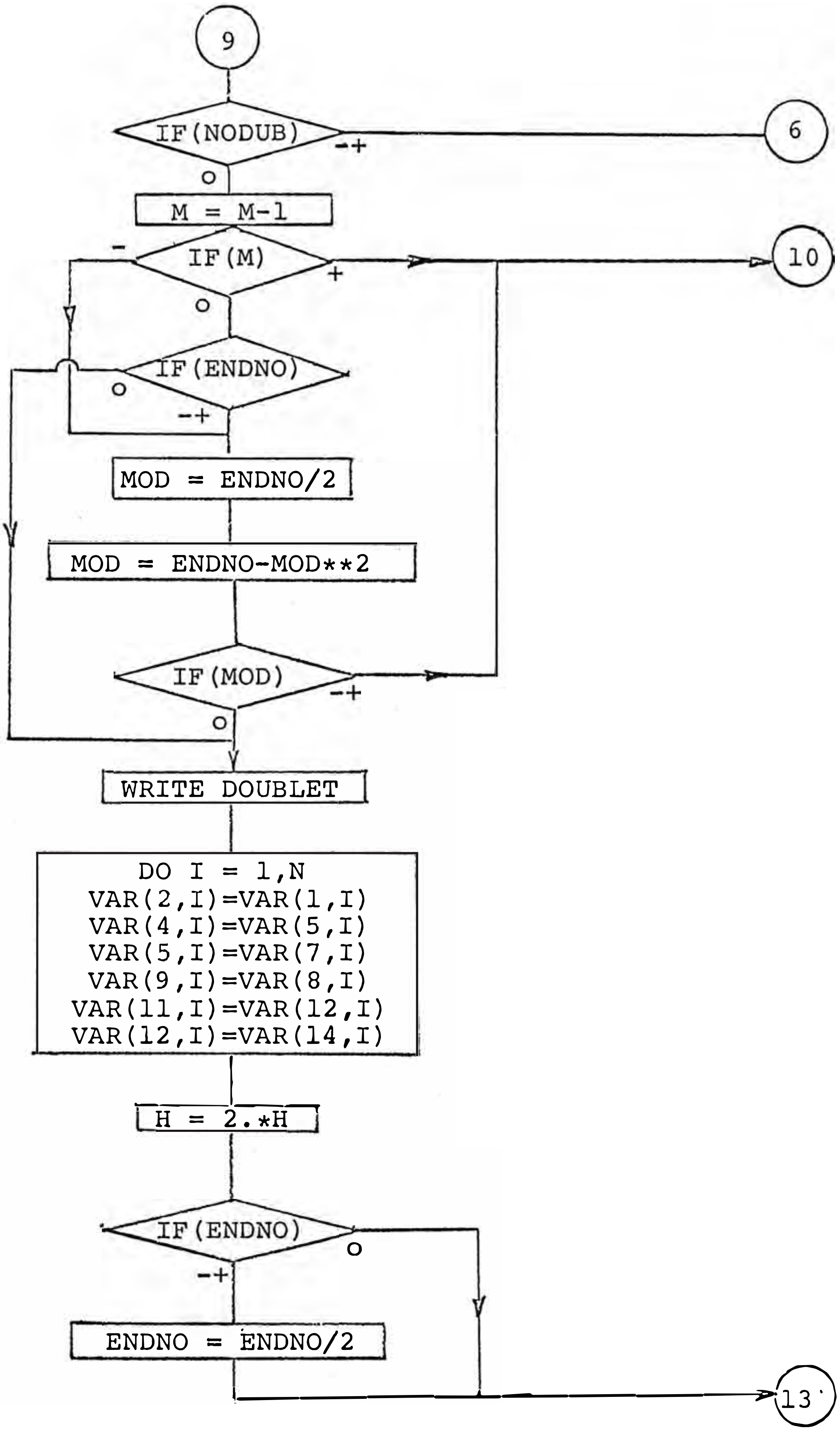


3
DO 462 I = 1,N
TEMP=VAR(2,I)+H*((.375*VAR(8,I))
+(.7916666666666667*VAR(9,I))
-(.2083333333333333*VAR(10,I))
+(.04166666666666670*VAR(11,I)))
IF(PHSIG)
0
-+
TEMPA=DABS(TEMP-VAR(1,I)
TEMPA = DABS((TEMP-VAR(1,I))/TEMP)
VAR(1,I) = TEMP
IF(PERR-TEMPA)
0+
-
PERR = TEMPA
462 CONTINUE
4


4
CALL POR
IF(PERR-CHECK1)
+
5
-o
NSWHF = 0
IF(NOD)
-
7
o+
8
IF(ENDNO)
o
-+
ENDNO = ENDNO-1
CALL ESCRIB
IF(FLAG)
-
RETURN
o+
IF(PERR-CHECK2)
-o
9
6
+
M = 3
10
J = 13
DO I = 1,N
VAR(J+1,I)=VAR(J,I)
J = J-1
IF(J)
+
-o
11


5
FLAG = DABS(BYHAL)
WRITE(HALVED)
IF(ENDNO)
-+
0
ENDNO = 2.*ENDNO
IF(NSWHF)
-+
0
DO I = 1,N
VAR(1,I) = VAR(5,I)
VAR(8,I) = VAR(12,I)
DO I = 1,N
VAR(1,I) = VAR(2,I)
VAR(8,I) = VAR(9,I)
X = X-(4.*H)
X = X-H
IF(ENDNO)
+
-0
ENDNO = ENDNO+6
H = H DABS(BYHAL)
12


7
X = X-3.*H
IF(ENDNO)
0
-+
ENDNO = ENDNO+2.
K = 3
DO I = 1,N
VAR(6,I) = VAR(1,I)
VAR(13,I) = VAR(8,I)
DO I = 1,N
VAR(1,I) = VAR(K+1,I)
VAR(8,I) = VAR(K+8,I)
CALL ESCRIB
IF(FLAG)
-
RETURN
+0
X = X+H
K = K-1
IF(K)
-0
+
DO I = 1,N
VAR(1,I) = VAR(6,I)
VAR(8,I) = VAR(13,I)
IF(ENDNO)
0
-+
ENDNO = ENDNO-1.
NOD = 0
8


9
IF(NODUB)
-+
6
0
M = M-1
-
IF(M)
+
10
0
IF(ENDNO)
0
-+
MOD = ENDNO/2
MOD = ENDNO-MOD**2
IF(MOD)
-+
0
WRITE DOUBLET
DO I = 1,N
VAR(2,I)=VAR(1,I)
VAR(4,I)=VAR(5,I)
VAR(5,I)=VAR(7,I)
VAR(9,I)=VAR(8,I)
VAR(11,I)=VAR(12,I)
VAR(12,I)=VAR(14,I)
H = 2.*H
IF(ENDNO)
0
-+
ENDNO = ENDNO/2
13