ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

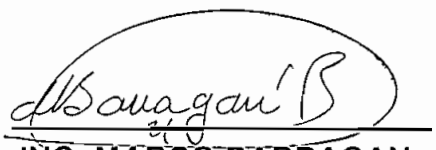
TESIS PREVIA A LA OBTENCION DEL TITULO DE INGENIERO EN LA ESPECIALIZACION DE ELECTRONICA Y CONTROL

ANALISIS COMPARATIVO DE METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA

JUAN FERNANDO DONOSO ARAUJO

QUITO 1992

CERTIFICO, que la presente tesis ha sido elaborada completamente por el Sr. Juan Fernando Donoso Araujo, bajo mi dirección.



ING. MARCO BARRAGAN

Agradezco a los ingenieros Marco Barragán y Rafael Fierro por su guía y ayuda para la realización del presente trabajo.

Este trabajo está dedicado a mis padres, familiares y amigos que apoyaron o ayudaron para su realización.

INDICE

INTRODUCCION

El análisis y diseño de sistemas de control de alto orden es, a menudo, difícil y costoso. Si se puede encontrar un modelo de bajo orden que se aproxime al sistema original, la simulación o el diseño del controlador para el modelo reducido, será más fácil que para el sistema original. Por ésta razón, los métodos de reducción de modelos han atraído especial interés en las áreas de control e instrumentación. La presente tesis está, por lo tanto, destinada a simplificar el diseño, análisis o trabajo matemático involucrado con cualquier sistema físico.

Dentro de la Teoría de Control, cualquier sistema real está representado por una ecuación matemática que relaciona las señales de entrada y de salida, a la que se denomina "función de transferencia" y que en ella están representadas todas las características que definen el comportamiento del sistema.

En muchos casos de sistemas reales, dicha función de transferencia, que no es más que una ecuación representada en el dominio de Laplace, tiene un elevado orden matemático, lo que trae como consecuencia que el análisis sobre la misma, para observar el comportamiento del sistema o realizar cualquier tipo de diseño, sea complicado y largo.

La finalidad de la presente tesis es, entonces, analizar varios métodos que permitan la posibilidad de reducir el orden matemático de la función de transferencia que define un cierto sistema, con la condición que, las características del mismo, representadas en su función de transferencia, no se alteren. En otras palabras, la nueva función de transferencia, equivalente, aproximada y de orden reducido a aquella original, debe ser capaz de reproducir la misma salida del sistema original cuando en él es aplicada una cierta señal de entrada.

Se han escogido varios métodos que cumplen este trabajo. Cada uno de ellos será analizado dependiendo de su campo de acción sobre cierto tipo de sistemas.

En el primer capítulo se analiza los métodos de reducción por expansión en fracciones continuas. Dichos métodos fueron propuestos por Cauer y basan su principio en la síntesis de redes con dos tipos de elementos (RL, RC o LC). Esta condición limita el uso de éstos métodos de reducción solamente a funciones de transferencia que representen los tipos de red antes mencionadas.

El capítulo segundo realiza un análisis de los métodos de reducción de modo dominante. Se llaman así porque dichos métodos basan su proceso de reducción en retener las características principales o dominantes del sistema y eliminar aquellas que no tienen mayor ingerencia en la salida. Se analizan tres métodos, los mismos que son:

- El método de Davison-Chidambara, cuyo proceso de basa en eliminar los polos más

alejados del eje *jw* y que no aportan mayormente a la salida del sistema. Sin embargo, dichos polos despreciados son usados para realizar una corrección en la ganancia del sistema reducido, cuando éste se encuentra en estado estable, de modo que sea la misma que la del sistema original.

- El método de Agregación, el mismo que mediante una matriz de "proyección" o matriz de "agregación" transforma las matrices asociadas al sistema original en otras, de menor orden, que conserven y reproduzcan las principales características del sistema original.

- El método de Realización Parcial, que define la llamada "matriz de Hankel", formada con muestras de la respuesta en el tiempo del sistema original, a una entrada impulso unitario, tomadas a un cierto intervalo de tiempo. Mediante transformaciones matemáticas, dicha matriz de Hankel es reducida a un orden menor y luego, usando un proceso de realización, se obtienen las matrices asociadas al sistema reducido buscado.

Los capítulos tres y cuatro analizan métodos de reducción para sistemas discretos.

En el capítulo tres, se estudian dos métodos, a saber:

- El método de Aproximaciones Optimas, el cual toma las muestras de la respuesta en el tiempo del sistema original, a una entrada impulso unitario, y mediante la resolución de un sistema de ecuaciones, encuentra los coeficientes de la función de transferencia del sistema reducido, en el plano z. Mediante un proceso matemático, es posible obtener las matrices asociadas al sistema reducido continuo a partir de aquellas del sistema discreto.

- El método de los Mínimos Cuadrados, que usando la respuesta en el tiempo del sistema original, a una entrada escalón unitario, determina los coeficientes de la función de transferencia del sistema reducido discreto, en varias iteraciones.

- En el capítulo cuarto, se usa el método denominado de los "Valores Singulares". Dichos valores son obtenidos a partir de una matriz de Hankel formada con las muestras de la respuesta en el tiempo, a una entrada impulso unitario, del sistema original. El orden del sistema reducido se obtendrá a partir del número de valores singulares calculados que sean distintos de cero. Mediante un proceso de realización se obtendrán las matrices asociadas al sistema reducido en el plano z, y usando los mismos procesos matemáticos utilizados en temas anteriores, se obtendrá la función de transferencia en el dominio de Laplace.

Una vez analizados los métodos mencionados, se exponen las conclusiones en las que se indican los resultados obtenidos en cada uno de ellos y la validez y confiabilidad de dichos métodos.

Finalmente se suscriben tres apéndices en los que se indican el listado completo del programa digital realizado en QuickBasic 4.0, el manual de uso del mismo y los conceptos de Parámetros de Markov y Momentos de tiempo de un cierto sistema, respectivamente, así como la bibliografía usada para el desarrollo de la tesis.

CAPITULO I

METODO DE REDUCCION POR EXPANSION EN FRACCIONES CONTINUAS

Los métodos de reducción de modelos presentados en este primer capítulo son aplicables a realización de redes con solamente dos clases de elementos. La función de transferencia que se obtiene de este tipo de redes posee ciertas características específicas que difieren de una función de transferencia cualquiera que describe un cierto sistema físico. Una de dichas características es que la diferencia de grado entre los polinomios del numerador y denominador de la función de transferencia no puede ser mayor a uno [1].

Existen cuatro realizaciones básicas de funciones de red con solamente dos tipos de elementos [2]. Dos de ellas son determinadas por expansión en fracciones parciales y fueron descubiertas por Foster; este tipo de realizaciones no caen dentro del campo de análisis del presente trabajo.

Las segundas dos formas de realización fueron descubiertas por Cauer y corresponden a una expansión en fracciones continuas de una cierta función de reactancia dada. Una primera expansión es alrededor de un punto en el infinito y puede designársela como la "Primera expansión de Cauer", y su correspondiente red en escalera como "Primera realización de Cauer". La segunda expansión es hecha alrededor del origen y, a menudo, es designada como "Segunda expansión de Cauer" y su correspondiente red en escalera como "Segunda forma de red de Cauer".

Es posible llamar "formas mixtas de realización" a los procesos que usan una combinación de expansión en fracciones continuas alrededor del origen y del infinito.

1.1 Primera Forma de Cauer.–

La Primera Forma de Cauer no se la considera como un método de reducción de modelos debido a que los resultados obtenidos al usarla para el efecto, en la mayoría de los casos, no guardan ninguna relación con aquel sistema original del que fueron obtenidos [3]. Solamente se la introduce aquí como una referencia del método usado a utilizar expansión en fracciones continuas para las formas generales de Cauer que se expondrán a continuación.

Sea G(s) una función de transferencia de n–ésimo orden de la forma:

$$G(s) = \frac{B_{20}s^{n-1} + B_{21}s^{n-2} + \ldots + B_{2,n-2}s + B_{2,n-1}}{B_{10}s^{n} + B_{11}s^{n-1} + \ldots + B_{1,n-1}s + B_{1n}} \qquad (1\text{-}1)$$

la expansión en fracciones continuas en la primera forma de Cauer es como se la representa a continuación:

$$G(s) = \cfrac{1}{H_1 s + \cfrac{1}{H_2 + \cfrac{1}{H_3 s + \cfrac{1}{\cdots}}}} \qquad (1\text{-}2)$$

Los coeficientes de la primera forma de Cauer pueden ser calculados desde el siguiente arreglo de Routh:

$$\begin{array}{lccccccc} & B_{10} & B_{11} & B_{12} & \cdots & B_{1,n-2} & B_{1,n-1} & B_{1n} \\ H_1 < & & & & & & & \\ & B_{20} & B_{21} & B_{22} & \cdots & B_{2,n-2} & B_{2,n-1} & \\ H_2 < & & & & & & & \\ & B_{30} & B_{31} & B_{32} & \cdots & B_{3,n-2} & B_{3,n-1} & \\ H_3 < & & & & & & & \\ & B_{40} & B_{41} & B_{42} & \cdots & B_{4,n-2} & & \\ H_4 < & & & & & & & \\ & B_{50} & B_{51} & B_{52} & \cdots & B_{5,n-2} & & \end{array} \qquad (1\text{-}3)$$

El algoritmo computacional que calcule dichos coeficientes vendrá dado por:

$$H(K) = \frac{B(K0)}{B(K+1,0)} \qquad (1\text{–}4)$$

$$B(I,J) = B(I\text{–}2,J+1) - H(I\text{–}2) * B(I\text{–}1,J+1) \qquad (1\text{–}5)$$

$$B(K+1,0) \neq 0$$

donde $K = 1, 2, 3, \ldots, 2m$

$I = 3, 4, 5, \ldots, 2m+2$

$J = 0, 1, \ldots, n\text{-}1$

n = orden del modelo original

m = orden del modelo reducido

Para obtener un modelo reducido de m–ésimo orden, los primeros 2m coeficientes deben ser calculados.

Para el caso de una función de transferencia de un modelo reducido de segundo orden se tiene que:

$$G^*(s) = \frac{C_{21}s + C_{20}}{C_{21}s^2 + C_{11}s + C_{10}} \qquad (1\text{–}6)$$

la expansión e inversión de la primera forma de Cauer puede ser obtenida como:

$$G^*(s) = [H_1 s + [H_2 + [H_3 s + [H_4]^{-1}]^{-1}]^{-1}]^{-1} \qquad (1\text{–}7)$$

o

$$G^*(s) = \frac{H_2H_3H_4s + (H_2+H_4)}{H_1H_2H_3H_4s^2 + (H_1H_2 + H_1H_4 + H_3H_4)s + 1} \qquad (1\text{–}8)$$

La relación entre los coeficientes C y H puede ser obtenida enseguida de las ecuaciones (1–6) y (1–8).

Las características del modelo original que son retenidas por el modelo reducido equivalente–aproximado, pueden ser observadas usando la ecuación (1–8).

Con frecuencia es necesario determinar el valor inicial y final de una función de respuesta f(t) [4], de un modo directo, a partir de la función de transformada F(s). Por medio de dichos valores es posible determinar el valor en estado estable que tendrá dicha función en el tiempo (si la función es de una señal estable) o las condiciones iniciales de la misma. Los teoremas del valor inicial y final permiten determinar los valores antes mencionados.

Si se aplica el teorema del valor final a las ecuaciones (1–1) y (1–8) teniendo como entrada a una función paso, se podrá determinar si el modelo reducido obtenido conserva o no el valor de la ganancia en estado estable del modelo original.

Al aplicarlo a la ecuación (1-1) se tiene:

$$\lim_{s \to 0} G(s) = \frac{B_{2,n-1}}{B_{1n}} \qquad (1\text{–}9)$$

Para la ecuación (1-8) se tiene:

$$\lim_{s \to 0} G^*(s) = H_2 + H_4 = \frac{B_{20}}{B_{30}} + \frac{B_{40}}{B_{50}} \qquad (1\text{–}10)$$

Aplicando el teorema del valor inicial, a las mismas ecuaciones, con una señal de entrada impulso unitario es posible obtener los valores de las condiciones iniciales de los mismos. A continuación se indican los resultados para las ecuaciones (1-1) y (1-8) respectivamente:

$$\lim_{s \to \infty} sG(s) = \frac{B_{20}}{B_{10}} \qquad (1\text{–}11)$$

$$\lim_{s \to \infty} sG^*(s) = \frac{1}{H_1} = \frac{B_{20}}{B_{10}} \qquad (1\text{–}12)$$

Comparando las ecuaciones (1–9) con (1–10) y (1–11) con (1–12) se puede concluir que el modelo reducido obtenido, al usar la expansión a fracciones continuas en la primera forma de Cauer, conserva las componentes rápidas del sistema, y no así, las componentes en estado estable. Esta es la razón primordial para no considerar a esta primera forma de Cauer como un método de reducción de modelos.

1.2 Segunda forma de Cauer.–

El modelo reducido obtenido a partir de la expansión en fracciones continuas usando la segunda forma de Cauer, tiene características diferentes de un modelo reducido obtenido usando la primera forma de Cauer, como se verá más adelante, al aplicar los teoremas de valor inicial y final, a los modelos original y reducido respectivamente.

Para obtener la segunda forma de Cauer, se puede arreglar la ecuación (1–1) de la siguiente forma:

$$G(s) = \frac{A_{20} + A_{21}s + \ldots + A_{2,n-2}s^{n-2} + A_{2,n-1}s^{n-1}}{A_{10} + A_{11}s + \ldots + A_{1,n-1}s^{n-1} + A_{1n}s^n} \qquad (1\text{-}13)$$

Cabe aclarar que los coeficientes representados en ésta última ecuación como A_{20}, $A_{21},\ldots,A_{2,n-1}$ son los mismos que los coeficientes $B_{2,n-1},\ldots,B_{21}$, B_{20}, respectivamente, representados en la ecuación (1–1). Exactamente lo mismo se puede decir de los coe-

ficientes A_{10}, A_{11},...,A_{1n} y B_{1n},...,B_{11}, B_{10} en las indicadas ecuaciones. Los cambios realizados solo se deben para diferenciar la notación entre los métodos que se analizan.

La expansión en fracciones continuas de la segunda forma de Cauer está dada por:

$$G(s) = \cfrac{1}{h_1 + \cfrac{1}{\cfrac{h_2}{s} + \cfrac{1}{h_3 + \cfrac{1}{\cfrac{h_4}{s} + \cfrac{1}{\ldots}}}}} \qquad (1\text{-}14)$$

De la misma manera como en la primera forma de Cauer, la variable h_K puede ser calculada desde el siguiente arreglo de Routh:

$$\begin{array}{lccccccc}
 & A_{10} & A_{11} & A_{12} & \ldots & A_{1,n-2} & A_{1,n-1} & A_{1n} \\
h_1 & & & & & & & \\
 & A_{20} & A_{21} & A_{22} & \ldots & A_{2,n-2} & A_{2,n-1} & \\
h_2 & & & & & & & \\
 & A_{30} & A_{31} & A_{32} & \ldots & A_{3,n-2} & A_{3,n-1} & \\
h_3 & & & & & & & \\
 & A_{40} & A_{41} & A_{42} & \ldots & A_{4,n-2} & & \\
h_4 & & & & & & & \\
 & A_{50} & A_{51} & A_{52} & \ldots & A_{5,n-2} & &
\end{array} \qquad (1\text{-}15)$$

El algoritmo que calcule los coeficientes del arreglo anterior estará dado por:

$$h(K) = \frac{A(K0)}{A(K+1,0)} \qquad (1\text{–}16)$$

$$A(I,J) = A(I\text{–}2,J+1) - h(I\text{–}2) * A(I\text{–}1,J+1) \qquad (1\text{–}17)$$

$$A(K+1,0) \neq 0$$

donde

K = 1, 2, 3,..., 2m

I = 3, 4, 5,..., 2m+1

J = 0, 1, 2,...,n-1

n = orden del modelo original

m = orden del modelo reducido

Para el caso de una función de transferencia de un modelo reducido de segundo orden se tiene:

$$G^*(s) = \frac{C_{21}s + C_{20}}{C_{21}s^2 + C_{11}s + C_{10}} \qquad (1\text{–}18)$$

la expansión e inversión de la segunda forma de Cauer puede ser obtenida como:

$$G^*(s) = \left[h_1 + \left[\frac{h_2}{s} + \left[h_3 + \left[\frac{h_4}{s} \right]^{-1} \right]^{-1} \right]^{-1} \right]^{-1} \qquad (1\text{-}19)$$

o

$$G^*(s) = \frac{(h_2 + h_4)s + h_2h_3h_4}{s^2 + (h_1h_2 + h_1h_4 + h_3h_4)s + h_1h_2h_3h_4} \qquad (1\text{–}20)$$

La relación entre los coeficientes C y h puede ser obtenida inmediatamente de las ecuaciones (1–18) y (1–20).

El modelo reducido de la ecuación (1–20) mantiene las componentes de estado estable del modelo original, lo que puede comprobarse aplicando el teorema del valor final a las ecuaciones (1–13) y (1–20):

$$\lim_{s \to 0} G(s) = \frac{A_{20}}{A_{10}} \qquad (1\text{–}21)$$

para la ecuación (1–13) y:

$$\lim_{s \to 0} G^*(s) = \frac{1}{h_1} = \frac{A_{20}}{A_{10}} \qquad (1\text{–}22)$$

para la ecuación (1–20).

Aplicando el teorema del valor inicial para una entrada impulso a las mismas ecuaciones, se tiene:

$$\lim_{s \to \infty} sG(s) = \frac{A_{2,n-1}}{A_{1n}} \qquad (1\text{–}23)$$

$$\lim_{s \to \infty} sG^*(s) = h_2 + h_4 = \frac{A_{20}}{A_{30}} + \frac{A_{40}}{A_{50}} \qquad (1\text{–}24)$$

Comparando las ecuaciones (1–21) con (1–22) y (1–23) con (1–24) se puede ver que el modelo reducido obtenido usando la segunda forma de Cauer, mantiene la ganancia de estado estable del modelo original a diferencia del modelo reducido obtenido mediante la primera forma de Cauer, que como ya se demostró, no conserva el valor de la ganancia en estado estable. Para los transitorios iniciales de la respuesta, se tiene exactamente lo contrario en ambos modelos, como ya se ha mencionado.

A continuación se muestra un ejemplo utilizando el proceso antes mencionado.

Sea un modelo de tercer orden dado por la siguiente función de transferencia:

$$G(s) = \frac{s^2+24s+140}{s^3+48s^2+144s+405}$$

De acuerdo a las ecuaciones (1-15) a (1-17) los coeficientes h pueden ser calculados con el al siguiente arreglo de Routh:

	405.0	144.0	48.0	1.0
2.8929				
	140.0	24.0	1.0	
1.8774				
	74.5714	45.1071	1.0	
-1.2289				
	-60.6839	-0.8774		
-1.3783				
	44.029	1.0		

El modelo reducido obtenido, según lo indicado por las ecuaciones (1-19) o (1-20) estará expresado por la siguiente función de transferencia"

$$G^*(s) = \frac{0.49912s+3.17933}{s^2+3.13758s+9.19849}$$

En el gráfico 1-1 se pueden observar las respuestas en el tiempo, a una señal escalón unitario, de los sistemas original y equivalente aproximado respectivamente.

1.3 Tercera forma de Cauer.–

La tercera forma de Cauer no es más que una combinación de las dos formas vistas anteriormente y por ello también se la denomina " Forma mixta de Cauer" [5]. Al obtener una combinación de las dos primeras formas de Cauer, lo que se consigue es que el modelo reducido obtenido mantenga tanto la ganancia en estado estable como las características de las componentes rápidas de la respuesta del modelo original. La expansión en fracciones continuas en esta tercera forma de Cauer vendrá dada por:

$$G(s) = \cfrac{1}{H_1 s + h_1 + \cfrac{1}{H_2 + \cfrac{h_2}{s} + \cfrac{1}{H_3 s + h_3 + \cfrac{1}{\cdots}}}} \qquad (1\text{-}25)$$

Los coeficientes H_k y h_k de esta tercera forma de Cauer pueden ser calculados desde el siguiente arreglo de Routh [3]:

$$\begin{array}{lccccccccl}
 & A_{10} & A_{11} & A_{12} & \cdots & A_{1,n-2} & A_{1,n-1} & A_{1n} & & \\
h_1 & & & & & & & & & H_1 \\
 & A_{20} & A_{21} & A_{22} & \cdots & A_{2,n-2} & A_{2,n-1} & & & \\
h_2 & & & & & & & & & H_2 \\
 & A_{30} & A_{31} & A_{32} & \cdots & A_{3,n-2} & & & & \\
h_3 & & & & & & & & & \\
 & A_{40} & A_{41} & A_{42} & \cdots & & & & & \\
h_4 & & & & & & & & & \\
 & A_{50} & A_{51} & A_{52} & \cdots & & & & &
\end{array} \qquad (1\text{-}26)$$

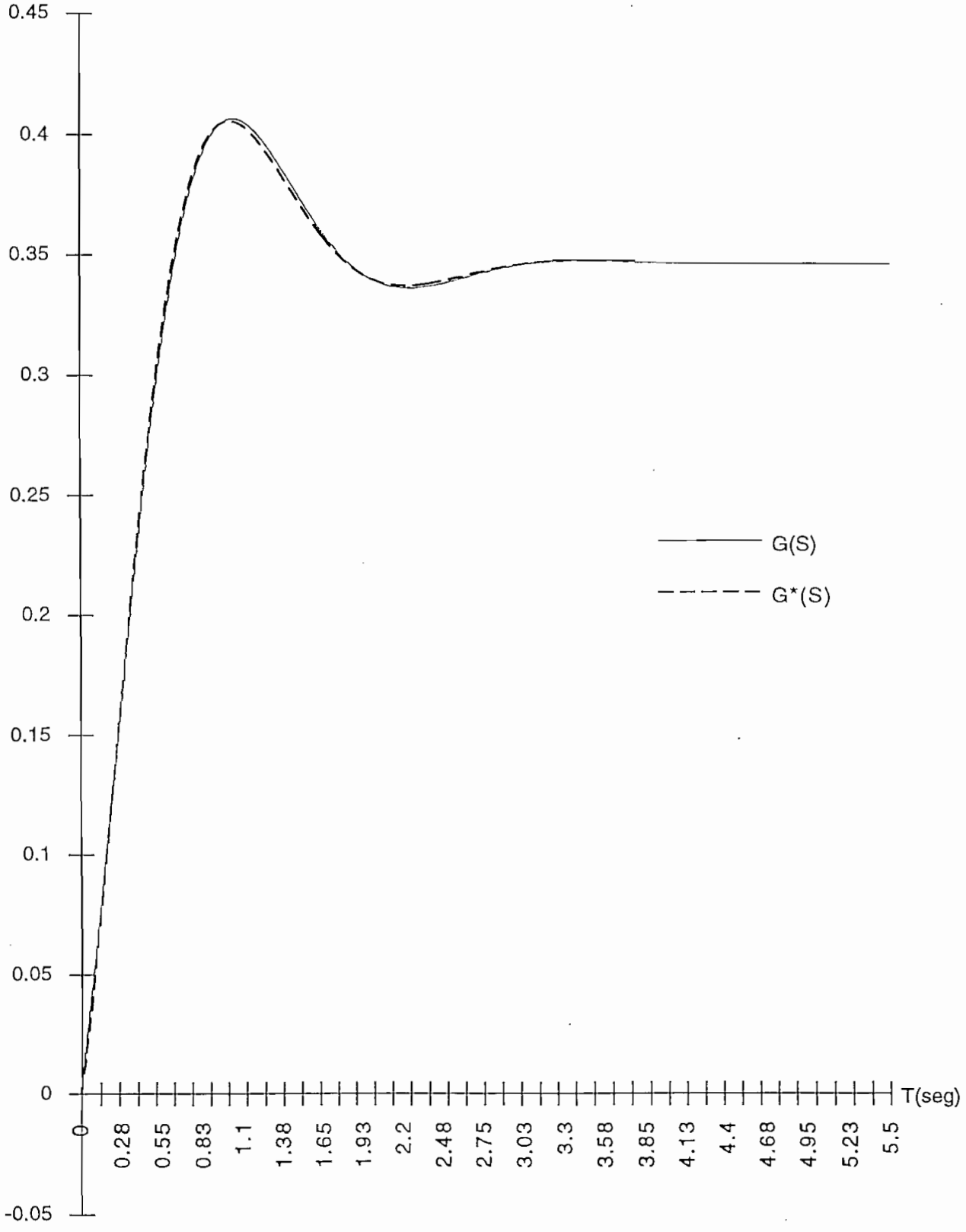

0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
0
0.28
0.55
0.83
1.1
1.38
1.65
1.93
2.2
2.48
2.75
3.03
3.3
3.58
3.85
4.13
4.4
4.68
4.95
5.23
5.5
T(seg)
G(S)
G*(S)

Gráfico 1-1

El algoritmo a implementarse puede ser expresado como:

$$h(K) = \frac{A(K0)}{A(K+1,0)} \tag{1–27}$$

$$H(K) = \frac{A(K,N+2-K)}{A(K+1,N+1-K)} \tag{1–28}$$

$$A(I,J) = A(I-2,J+1) - h(I-2) * A(I-1,J+1) - H(I-2) * A(I-1,J) \tag{1–29}$$

donde:

K= 1, 2, 3,..., m

I= 3, 4, 5,..., 2m+1

J= 0, 1, 2,..., n

n = orden del sistema original

m = orden del modelo reducido

Para obtener un modelo reducido de m–ésimo orden, m coeficientes h deben ser calculados.

Con los coeficientes requeridos por las fracciones parciales, la expansión en fracciones continuas de la ecuación (1–25) es convertida en una función de transferencia.

Si se considera el caso de segundo orden se tendrá:

$$G^*(s) = \frac{C_{21}s + C_{20}}{C_{21}s^2 + C_{11}s + C_{10}} \tag{1–30}$$

Dicha función de transferencia puede ser obtenida a partir de la expansión en fracciones continuas y respectiva conversión a función de transferencia siguientes:

$$G^*(s) = \frac{1}{H_1 s + h_1 + \dfrac{1}{H_2 + \dfrac{h_2}{s}}} \tag{1-31}$$

$$G^*(s) = \frac{H_2 s + h_2}{H_1 H_2 s^2 + (1 + h_1 H_2 + H_1 h_2)s + h_1 h_2} \tag{1–32}$$

Aplicando el teorema del valor final a las ecuaciones (1–13) y (1–32) con una entrada paso se tiene:

$$\lim_{s \to 0} G(s) = \frac{A_{20}}{A_{10}} \tag{1–33}$$

$$\lim_{s \to 0} G^*(s) = \frac{1}{h_1} = \frac{A_{20}}{A_{10}} \tag{1–34}$$

Con el teorema del valor inicial y una entrada impulso unitario:

$$\lim_{s \to \infty} sG(s) = \frac{A_{2,n-1}}{A_{1n}} \tag{1–35}$$

$$\lim_{s \to \infty} sG^*(s) = \frac{1}{H_1} = \frac{A_{2,n-1}}{A_{1n}} \tag{1–36}$$

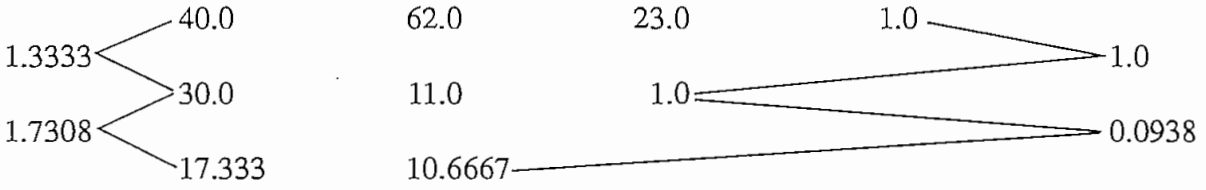
Comparando las ecuaciones (1–33) con (1–34) y (1–35) con (1–36), queda demostrado que, matemáticamente, el modelo reducido mantiene la ganancia en estado estable y las componentes rápidas de la respuesta del modelo original.

A continuación se presenta un ejemplo que explica el proceso:

Sea la función de transferencia de tercer orden dada por:

$$G(s) = \frac{s^2+11s+30}{s^3+23s^2+62s+40}$$

Los coeficientes y el arreglo de Routh para la obtención de la función de transferencia equivalente reducida estarán dados por:



La función de transferencia de orden reducido obtenida, de acuerdo a lo expresado en la ecuación (1-32) será:

$$G^*(s) = \frac{s+18.46154}{s^2+30.46154s+24.61538}$$

El gráfico 1-2 muestra las respuesta en el tiempo a una señal escalón unitario de los sistemas original y equivalente reducido.

1.4 Forma modificada de Cauer.–

Una modificación en la expansión en fracciones continuas de la tercera forma de Cauer es la base de este método de reducción de funciones de trasferencia. En la forma modificada de Cauer, la expansión en fracciones continuas viene dada en la siguiente expresión:

$$G(s) = \cfrac{1}{h_1 + \cfrac{s}{H_1 + \cfrac{1}{h_2 + \cfrac{s}{H_2 + \cfrac{1}{\cdots}}}}} \qquad (1\text{-}37)$$

Los coeficientes pueden ser calculados desde el arreglo de Routh mostrado a continuación:

$$\begin{array}{lccccccc}
 & A_{10} & A_{11} & A_{12} & \cdots & A_{1,n-2} & A_{1,n-1} & A_{1n} \\
h_1 & & & & & & & \\
 & A_{20} & A_{21} & A_{22} & \cdots & A_{2,n-2} & A_{2,n-1} & \\
 & & & & & & & H_1 \\
 & A_{30} & A_{31} & A_{32} & \cdots & A_{3,n-2} & A_{3,n-1} & \\
h_2 & & & & & & & \\
 & A_{40} & A_{41} & A_{42} & \cdots & A_{4,n-2} & & \\
 & & & & & & & H_2 \\
 & A_{50} & A_{51} & A_{52} & \cdots & A_{5,n-2} & &
\end{array} \qquad (1\text{-}38)$$

donde:

$$h_1 = \frac{A(1,0)}{A(2,0)}\ ;\ H_1 = \frac{A(2,n-1)}{A(3,n-1)}\ ;\ h_2 = \frac{A(3,0)}{A(4,0)}\ ;\ \text{etc.}$$

$$A(I,J) = A(I-2,J+1) - h((I-1)/2) * A(I-1,J+1) \qquad (1\text{–}39)$$

para i= 3, 5, 7,...
j= 0, 2,..., n-1

$$A(I,J) = A(I-2,J) - H((I-2)/2) * A(I-1,J)$$

para i = 4, 6, 8,...
j = 0, 2,...,n-1
n = orden del modelo original

Como en el caso anterior, para la obtención de un modelo reducido de m–ésimo orden, m coeficientes h deben ser calculados.

Una función de transferencia de segundo orden puede ser expandida de la siguiente manera:

$$G^*(s) = \cfrac{1}{h_1 + \cfrac{s}{H_1 + \cfrac{1}{h_2 + \cfrac{s}{H_2}}}} \qquad (1\text{-}40)$$

Resolviendo las fracciones continuas se tiene:

$$G^*(s) = \frac{H_1 s + (H_2 + H_1 H_2 h_2)}{s^2 + (h_1 H_1 + h_2 H_2)s + h_1(H_2 + H_1 H_2 h_2)} \qquad (1\text{–}41)$$

Con una entrada paso, el teorema del valor final del sistema reducido (ecuación 1–41), nos da el siguiente resultado:

$$\lim_{s \to 0} G^*(s) = \frac{1}{h_1} = \frac{A_{20}}{A_{10}} \qquad (1\text{–}42)$$

Y el teorema del valor inicial, para una entrada impulso, aplicado en la misma ecuación (1–41) nos da:

$$\lim_{s \to \infty} sG^*(s) = H_1 = \frac{A_{2,n-1}}{A_{3,n-1}} = \frac{A_{2,n-1}}{A_{1n}} \qquad (1\text{–}43)$$

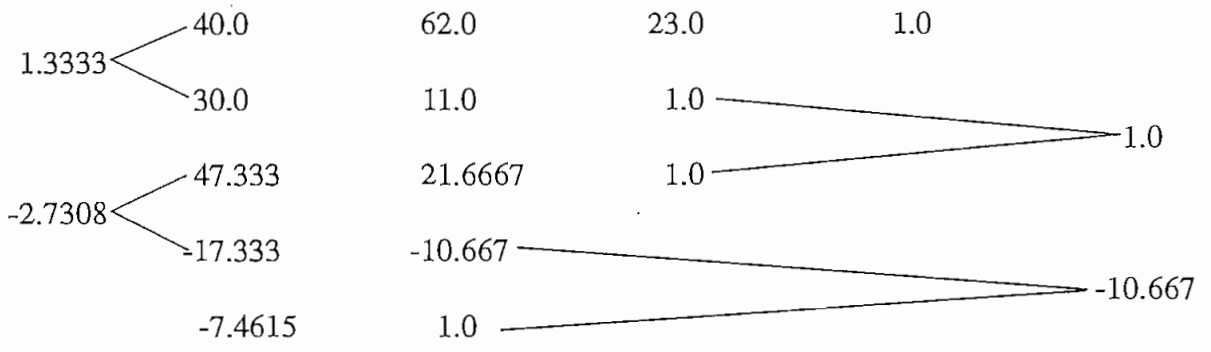
Como se puede observar, matemáticamente, la expansión en la forma modificada de Cauer también mantiene la ganancia en estado estable y las transiciones iniciales de la respuesta.

A continuación se expone un ejemplo explicativo del método enunciado.

Sea un sistema de tercer orden expresado por la siguiente función de transferencia:

$$G(s) = \frac{s^2+11s+30}{s^3+23s^2+62s+40}$$

Los coeficientes para el cálculo de la función de transferencia reducida se los puede obtener del siguiente arreglo de Routh de acuerdo con las ecuaciones (1-38) y (1-39):



De acuerdo con la ecuación (1-41) la función de transferencia reducida, de segundo orden, estará dada por:

$$G^*(s) = \frac{s+18.46154}{s^2+30.46154s+24.61538}$$

El gráfico 1-3 muestra las respuesta en el tiempo a una señal escalón unitario de los sistemas original y equivalente reducido.

1.5 Forma general de Cauer.–

Este método se basa en fijar momentos de tiempo y parámetros de Markov (ver Apéndice), y debe su nombre a que puede producir un modelo reducido en la primera forma de Cauer, en la segunda o en la forma modificada de Cauer de acuerdo a cómo se fijen los parámetros requeridos.

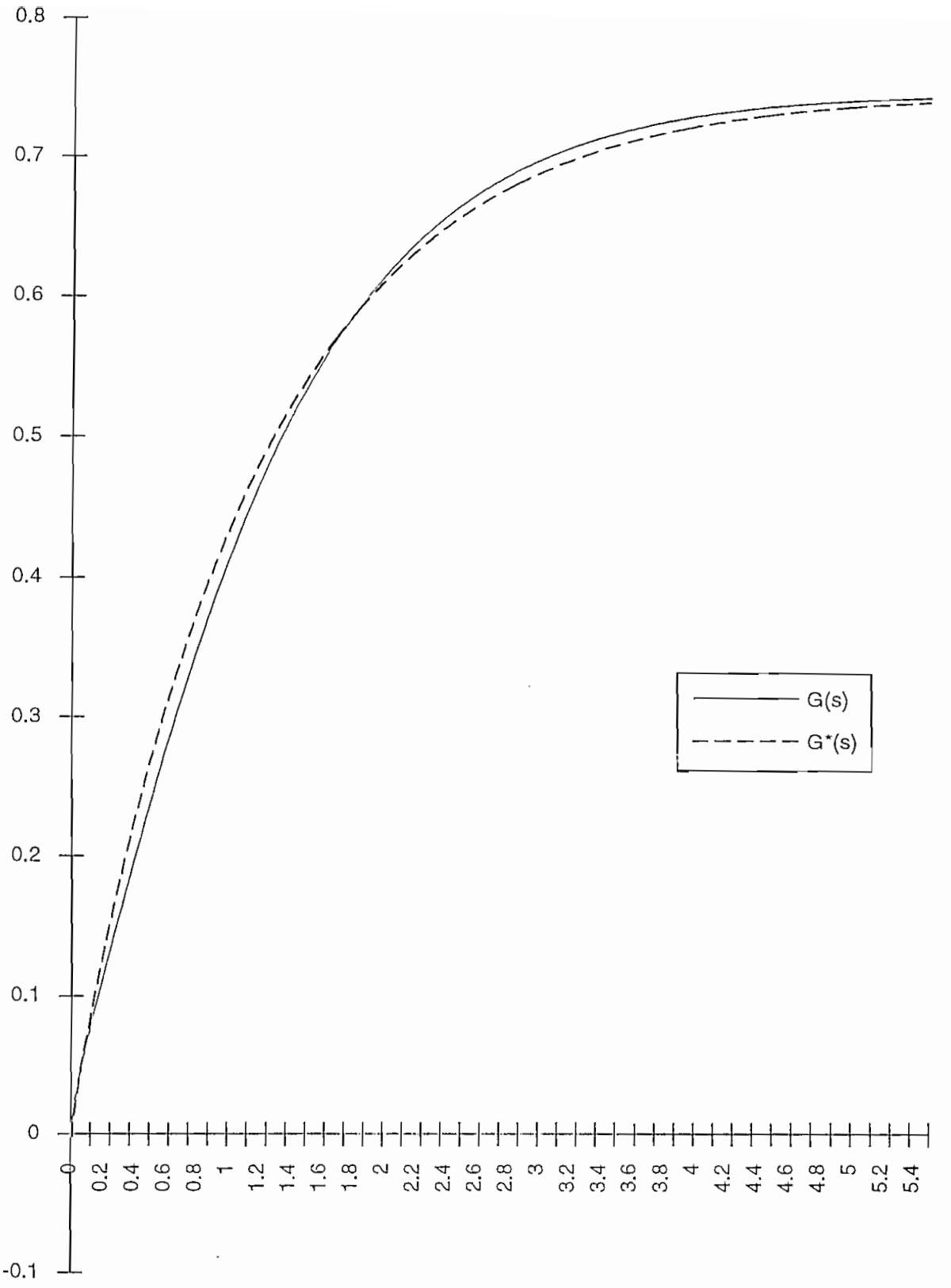

0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
4.2
4.4
4.6
4.8
5
5.2
5.4
G(s)
G*(s)

Gráfico 1-3

Sea m el orden del modelo reducido y t los momentos de tiempo retenidos o fijados. Los parámetros de Markov retenidos serán 2m–t. Las fracciones continuas de la forma general de Cauer están indicadas en la ecuacion (1–44). En el caso de t=0 la forma de la expansión será la misma que en la primera forma de Cauer. Para t=2m, obtenemos la segunda forma de Cauer. Para obtener la forma modificada de Cauer, necesitamos hacer t=m de modo que la forma retiene m momentos de tiempo y m parámetros de Markov.

$$G(s) = \cfrac{1}{h_1 + \cfrac{s}{h_2 + \cfrac{s}{\cdots \quad \cdots \quad h_t + \cfrac{1}{sH_1 + \cfrac{1}{H_2 + \cfrac{1}{H_3 s + \cfrac{1}{\cdots}}}}}}}} \qquad (1\text{-}44)$$

Los h_i (i=1,2,...,t) y H_i (i=1,2,...,2m–t) pueden ser obtenidos el arreglo de Routh indicado en (1-49).

Dichos coeficientes son calculados desde el siguiente algoritmo:

$$h(K) = \frac{A(K0)}{A(K+1,0)} \qquad (1\text{–}45)$$

$$A(I,J) = A(I\text{–}2,J+1) \; - \; h(I\text{–}2) * A(I\text{–}1,J+1) \qquad (1\text{–}46)$$

para $K = 1,2,...,t$

$I = 3,4,...,t+2$

$J = 0,...,n/2$

$$H(K) = \frac{B(K0)}{B(K+1,0)} \qquad (1\text{–}47)$$

$$B(I,J) = B(I\text{–}2,J+1) \; - \; H(I\text{–}2) * B(I\text{–}1,J+1) \qquad (1\text{–}48)$$

para K = 1,2,...,2m–t

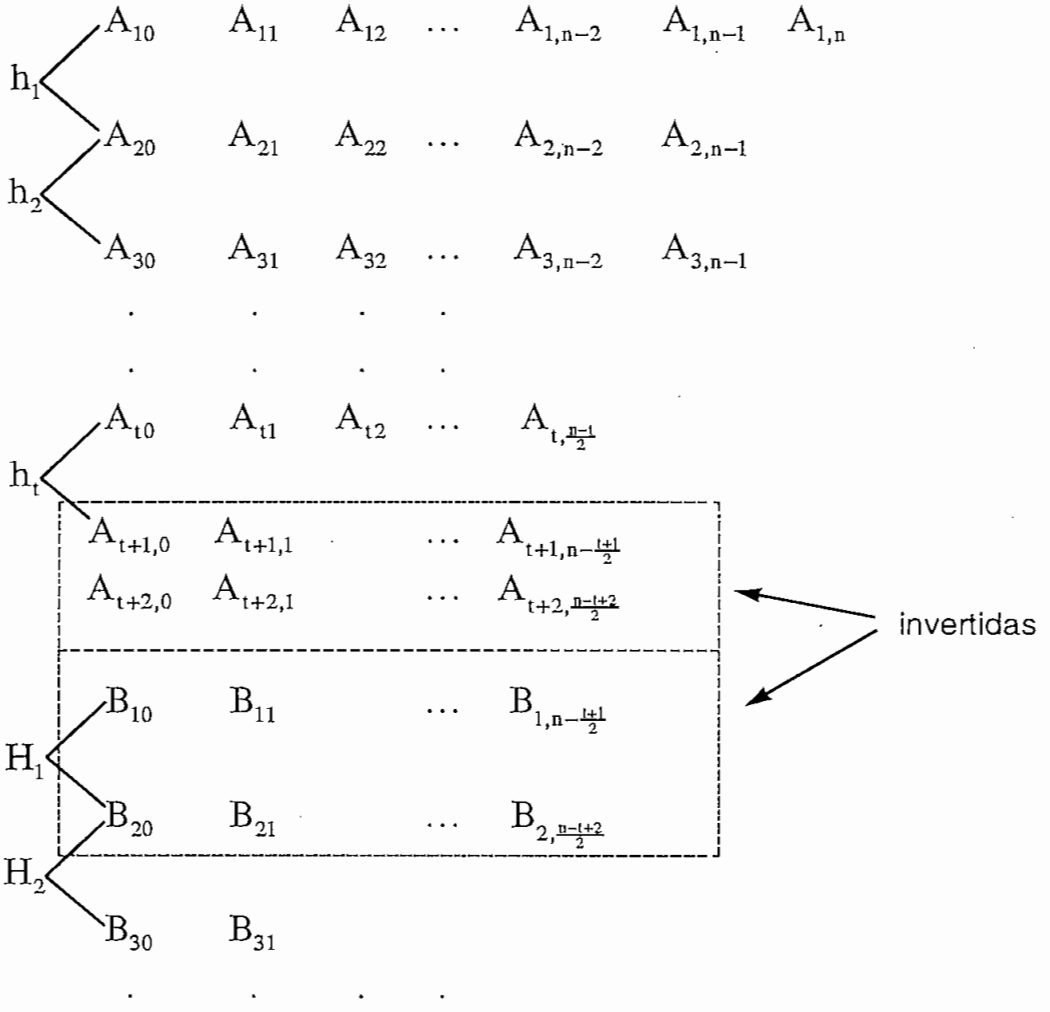
I = 3,4,...,2m–t+2

J = 0,...,n-i/2

n = orden del sistema original

m = orden del sistema reducido

t = momentos de tiempo fijados



(1-49)

De la ecuación (1–49) se puede observar que las dos primeras filas de la submatriz B_{ij} son idénticas, pero en sentido inverso a las dos últimas filas de la submatriz A_{ij}.

Si se retiene un momento de tiempo y tres parámetros de Markov, se obtendrá un sistema reducido de la siguiente forma:

$$G^*(s) = \cfrac{1}{h_1 + \cfrac{s}{H_1 + \cfrac{1}{sH_2 + \cfrac{1}{H_3}}}} \qquad (1\text{-}50)$$

Resolviendo la ecuación (1–50) se tiene:

$$G^*(s) = \frac{H_1H_2H_3s + (H_1 + H_3)}{H_2H_3s^2 + (H_1H_2H_3h_1 + 1)s + h_1(H_1 + H_3)} \qquad (1\text{–}51)$$

El teorema del valor final, aplicando una entrada paso al modelo reducido,$G^*(s)$, nos da como resultado:

$$\lim_{s \to 0} G^*(s) = \frac{1}{h_1} = \frac{A_{20}}{A_{10}} \qquad (1\text{–}52)$$

y el valor inicial para una entrada impulso es:

$$\lim_{s \to \infty} sG^*(s) = H_1 = \frac{B_{10}}{B_{20}} = \frac{A_{t+1,n-(t+1)/2}}{A_{t+2,(n-t+2)/2}} \qquad (1\text{–}53)$$

$$= \frac{A_{2,n-1}}{A_{3,n-1}} = \frac{A_{2,n-1}}{A_{1,n}}$$

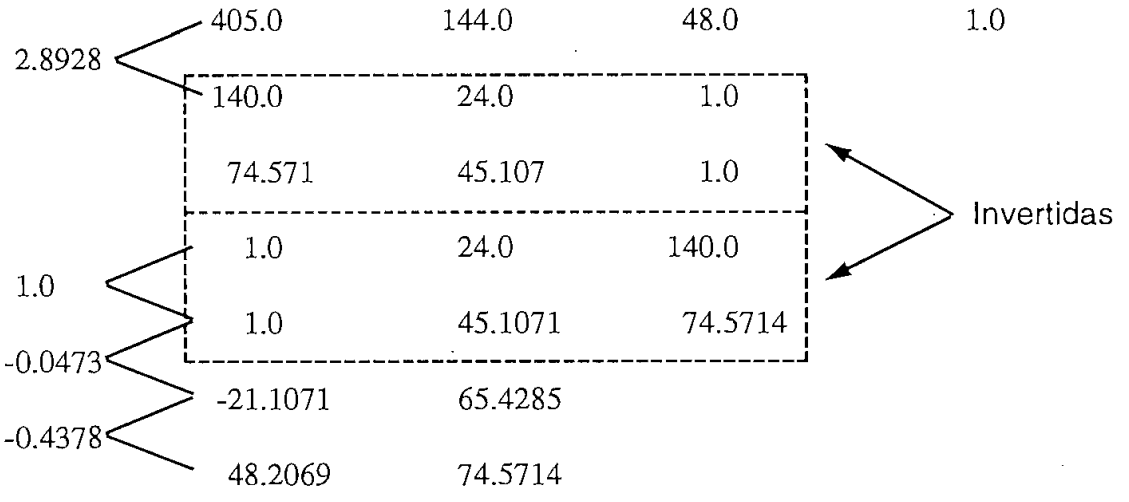
Si se compara las ecuaciones (1-52) y (1-53) con (1-33) y (1-34) se puede concluir que, fijando el primer momento de tiempo y los primeros tres parámetros de Markov, se mantiene la ganancia en estado estable y las transiciones iniciales de la respuesta.

Con el ejemplo siguiente se ilustra el proceso mencionado.

Sea un sistema de tercer orden, descrito por la siguiente función de transferencia:

$$G(s) = \frac{s^2+24s+140}{s^3+48s^2+144s+405}$$

Para la obtención de un sistema de segundo orden, reteniendo un sólo momento de tiempo, y de acuerdo con (1-49), el arreglo de Routh para el cálculo de los coeficientes vendrá dado por:



De acuerdo con la ecuación (1-51) la función de transferencia reducida, de segundo orden, estará dada por:

$$G^*(s) = \frac{s+27.09983}{s^2+51.09983s+78.39594}$$

El gráfico 1-4 muestra las respuesta en el tiempo a una señal de entrada escalón unitario de los sistemas original y equivalente reducido.

1.6 Nueva forma generalizada de expansión en fracciones continuas.–

Una vez analizado el subtema anterior se puede decir que el tiempo de establecimiento del modelo reducido es casi el mismo que el del sistema original, por lo que se conservaban las transiciones iniciales de la respuesta. Se puede concluir entonces que, si fijamos un mayor número de momentos de tiempo al hacer la reducción del sistema, el tiempo de establecimiento del sistema reducido se acercará más aún al del sistema original [6].

Sin embargo, la forma general de Cauer propuesta anteriormente falla al tratar de obtener el modelo reducido de un cierto sistema cuando se fijan tres momentos de tiempo y un parámetro de Markov [5]. Para ello se presenta aquí una nueva forma general de expansión en fracciones continuas, como lo muestra la ecuación (1–54). Este método es similar al anterior pero es efectivo al fijar tres momentos de tiempo y un parámetro de Markov.

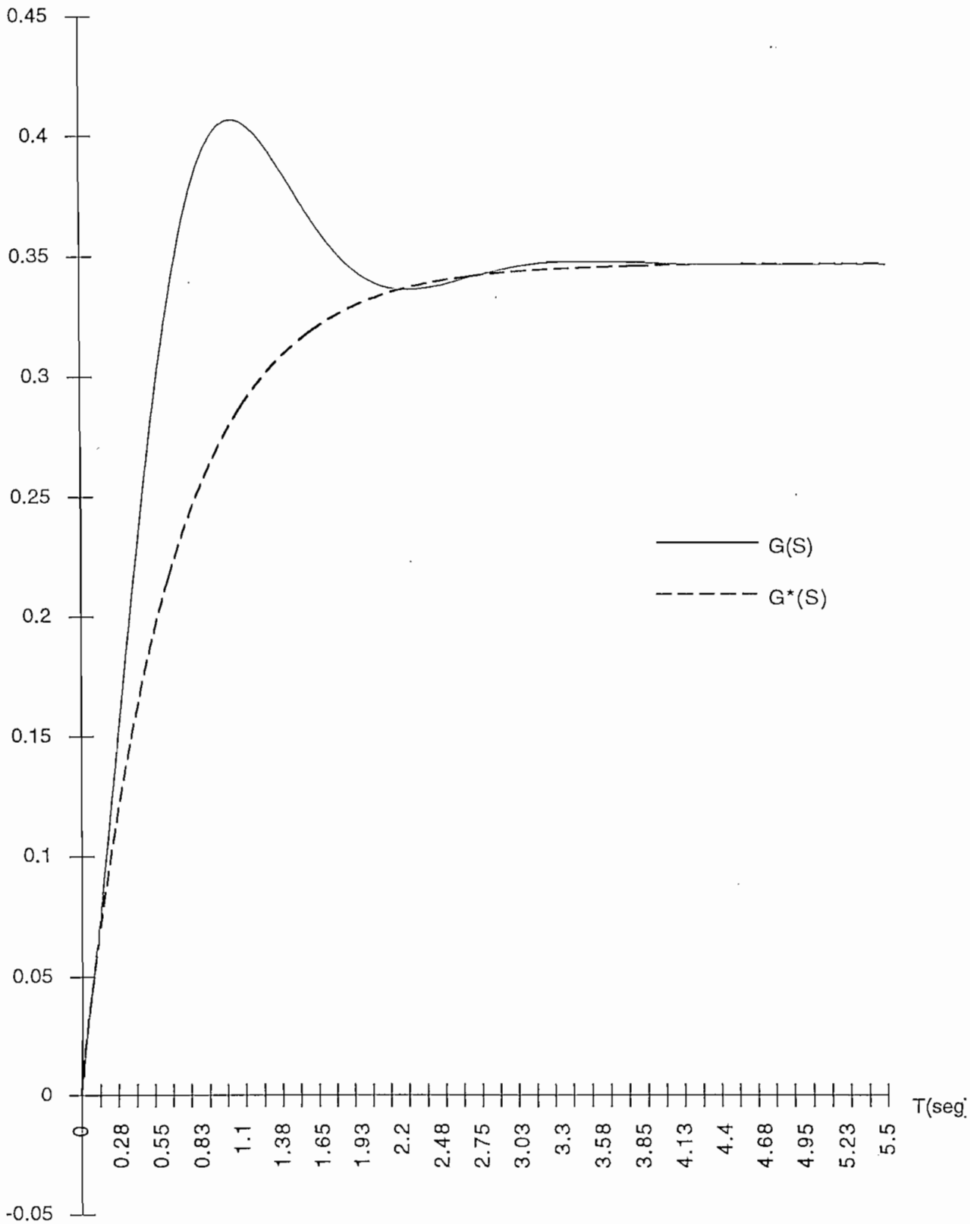

0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
G(S)
G*(S)
T(seg)
0
0.28
0.55
0.83
1.1
1.38
1.65
1.93
2.2
2.48
2.75
3.03
3.3
3.58
3.85
4.13
4.4
4.68
4.95
5.23
5.5

Gráfico 1-4

Sólo por notación, se considera que la función de transferencia que describe a un sistema cualquiera estará dada por:

$$G(s) = \frac{B_{20} + B_{21}s + \ldots + B_{2,n-2}s^{n-2} + B_{2,n-1}s^{n-1}}{B_{10} + B_{11}s + \ldots + B_{1,n-1}s^{n-1} + B_{1n}s^{n}} \qquad (1\text{-}54)$$

La forma de expansión en fracciones continuas será la siguiente:

$$G(s) = \cfrac{1}{sH_1 + \cfrac{1}{H_2 + \cfrac{1}{sH_3 + \cfrac{1}{\ldots \; H_w + \cfrac{1}{h_1 + \cfrac{s}{h_2 + \cfrac{s}{h_3 + \cfrac{s}{\ldots}}}}}}}} \qquad (1\text{-}55)$$

Sea m el orden del sistema reducido y w los parámetros de Markov retenidos; entonces, 2m–w serán los momentos de tiempo retenidos. Como en los métodos anteriores, los parámetros pueden ser obtenidos a partir de un arreglo de Routh, que para este modelo está indicado en la ecuación (1-60).

Los coeficientes definidos en el arreglo mencionado se calcularán mediante el siguiente algoritmo:

$$H(K) = \frac{B(K0)}{B(K+1,0)} \qquad (1\text{–}56)$$

$$B(I,J) = B(I\text{–}2,J+1) - H(I\text{–}2) * B(I\text{–}1,J+1) \qquad (1\text{–}57)$$

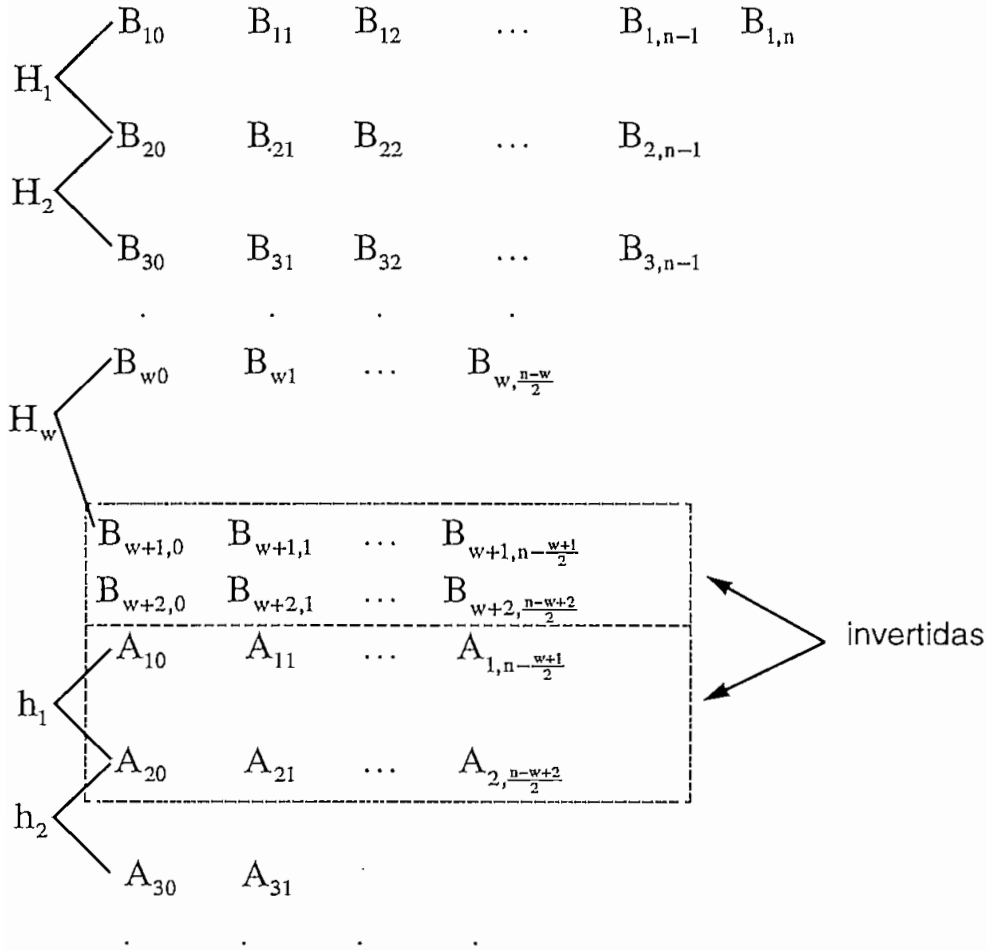
para K = 1, 2,..., w
I = 3, 4,..., w+2
J = 0, 1,...,n-i/2

$$h(K) = \frac{A(K0)}{A(K+1,0)} \qquad (1\text{–}58)$$

$$A(I,J) = A(I\text{–}2,J+1) - h(I\text{–}2) * A(I\text{–}1,J+1) \qquad (1\text{–}59)$$

para K = 1, 2,..., 2m–w
I = 3, 4,..., 2m–w+1
J = 0, 1,...,n-i/2

n = orden del sistema original
m = orden del sistema reducido
w = parámetros de Markov retenidos



(1-60)

En este caso también se cumplen que las dos primeras filas de la submatriz A_{ij} son idénticas, pero en orden inverso, a las dos últimas fila de la submatriz B_{ij}.

Cuando w=0 el método será equivalente a la segunda forma de Cauer, y si w=m significará que se está reteniendo m parámetros de Markov y m momentos de tiempo. Para calcular un sistema reducido de segundo orden reteniendo tres momentos de tiempo y un parámetro de Markov, se obtendrá:

$$G^*(s) = \cfrac{1}{sH_1 + \cfrac{1}{h_1 + \cfrac{s}{h_2 + \cfrac{s}{h_3}}}} \qquad (1\text{-}61)$$

Resolviendo la ecuación (1–61) se tiene:

$$G^*(s) = \frac{(h_1+h_3)s + h_1h_2h_3}{H_1(h_1+h_3)s^2 + (h_1h_2h_3H_1+1)s + h_2h_3} \qquad (1\text{–}62)$$

Aplicando el teorema del valor final, con una entrada escalón unitrario, al modelo reducido, tenemos:

$$\lim_{s \to 0} G^*(s) = h_1 = \frac{B_{20}}{B_{10}} \qquad (1\text{–}63)$$

y el valor inicial para una entrada impulso unitario es:

$$\lim_{s \to \infty} sG^*(s) = \frac{1}{H_1} = \frac{B_{2n-1}}{B_{1n-1}} \qquad (1\text{–}64)$$

Aplicando los mismos teoremas a la ecuación (1-54) con las mismas señales de entrada, respectivamente, se tiene:

$$\lim_{s \to 0} G(s) = \frac{B_{20}}{B_{10}} \qquad (1\text{–}65)$$

$$\lim_{s \to \infty} sG(s) = \frac{B_{2n-1}}{B_{1n-1}} \qquad (1\text{–}66)$$

Estas últimas ecuaciones demuestran que el sistema reducido conserva las mismas características del sistema original del que fue obtenido.

A continuación se desarrolla un ejemplo explicativo del método.

Sea un sistema de tercer orden dado por la siguiente función de transferencia:

$$G(s) = \frac{s^2+24s+140}{s^3+48s^2+144s+405}$$

Reteniendo un solo parámetro de Markov, el arreglo de Routh para el cálculo de los coeficientes vendrá dado por:

	1.0	48.0	144.0	405.0
1.0	1.0	24.0	140.0	
	24.0	4.0	405.0	
	140.0	24.0	1.0	
0.3457	405.0	4.0	24.0	
17.9067	22.6173	-7.2963	0	
0.168	134.6523	24.0	0	

invertidas

De acuerdo a la ecuación (1-62) la función de transferencia reducida vendrá dada por la siguiente expresión:

$$G^*(s) = \frac{s + 2.02418}{s^2 + 3.97104s + 5.8557}$$

El gráfico 5-1 muestra las respuesta en el tiempo, ante una entrada escalón unitario, de los sistemas original y equivalente reducido, respectivamente.

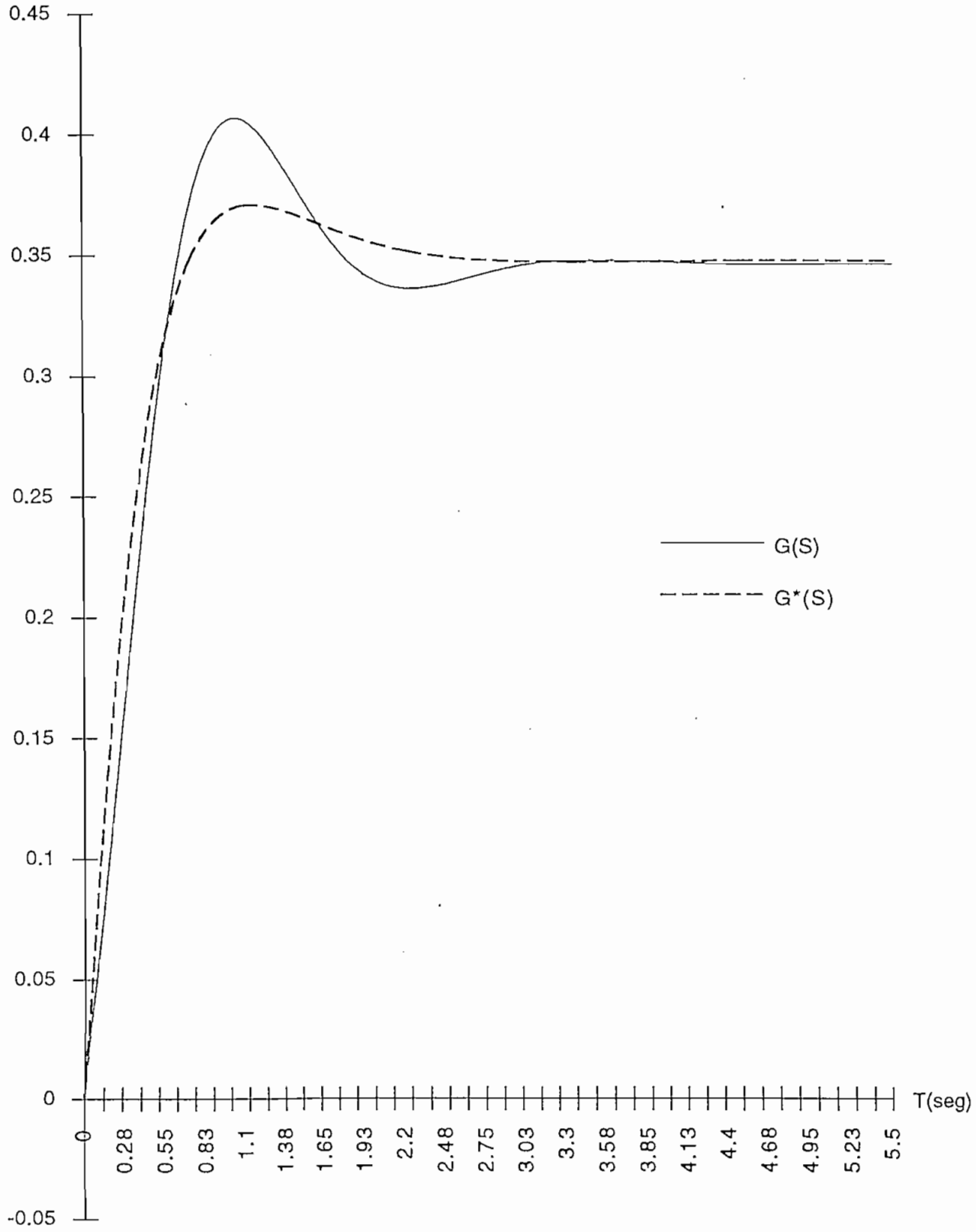

0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
G(S)
G*(S)
T(seg)
0
0.28
0.55
0.83
1.1
1.38
1.65
1.93
2.2
2.48
2.75
3.03
3.3
3.58
3.85
4.13
4.4
4.68
4.95
5.23
5.5

Gráfico 1-5

CAPITULO II

METODOS DE MODO DOMINANTE

En el presente capítulo se estudiarán tres métodos de obtención de modelos reducidos a partir de sistemas cuya ecuación de transferencia es de alto orden.

Básicamente, basan su principio de reducción en los siguientes parámetros, en una forma independiente cada uno de ellos:

- Obtención del sistema reducido a partir de los polos dominantes del sistema original (aquellos polos cercanos al eje jw), eliminando los polos menos significativos (aquellos alejados del eje jw con respecto a los polos dominantes y que no tienen mayor injerencia en la respuesta del sistema).

- Obtención del sistema reducido a través de una "matriz de proyección", la misma que reducirá el orden de las matrices asociadas al sistema original (descrito en forma de variables de estado), para de esta manera obtener un sistema de menor orden. Dicha matriz es escogida en una forma tal para conservar las principales características de respuesta del sistema original, en el sistema de orden reducido.

- Desarrollar una serie de potencias a partir de la función de transferencia del sistema original, arreglando sus términos en una matriz, y obtener el sistema de reducido eliminando ciertas filas y columnas de dicha matriz, de acuerdo al orden que se desee obtener.

2.1 Método de Davison-Chidambara.-

Davison desarrolla un método de reducción eliminando los polos menos significativos de un sistema dado. Según su método, conservando sólo aquellos polos dominantes (que se encuentran cerca del eje jw) puede obtenerse un modelo reducido con las principales características de respuesta del modelo original.

Chidambara, usando las ideas originales de Davison, propone utilizar los polos despreciados para "mejorar" la respuesta en estado estable, ya que de acuerdo al método de Davison, en el sistema reducido no se consigue que la ganancia en estado estable sea exactamente la misma que el sistema original [7], [8].

Dado el modelo original de un sistema lineal multivariable e invariante en el tiempo, al cual se lo puede representar por las siguientes ecuaciones de estado:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \qquad (2\text{-}1)$$
$$\mathbf{y} = C\mathbf{x}$$

donde A es una matriz n x n, **x** es un vector de estado n-dimensional, **u** es un vector de entrada p-dimensional, **y** es un vector de salida q-dimensional, y B y C son matrices constantes de dimensión apropiada, se requiere encontrar un modelo matemático, equivalente y aproximado y de orden reducido que esté dado por las siguientes ecuaciones de estado:

$$\dot{\mathbf{x}}^* = A^*\mathbf{x}^* + B^*\mathbf{u} \qquad (2\text{-}2)$$
$$\mathbf{y} = C\mathbf{x}$$

donde A^* una matriz m x m (m<n).

Para obtener el modelo reducido, los coeficientes de las matrices A^* y B^*, así como la relación entre **x** y $\mathbf{x}^*$ deben ser determinadas.

Supóngase que la ecuación característica asociada con la matriz A está dada por:

$$(\lambda - \lambda_1)^{j_1}(\lambda - \lambda_2)^{j_2} \ldots (\lambda - \lambda_k)^{j_k} = 0 \qquad (2\text{-}3)$$

donde $\lambda_1 < \lambda_2 < \ldots < \lambda_k$, y k_i es el orden de multiplicidad del valor propio λ_i, que cumple la relación:

$$\sum_{i=1}^{k} j_i = n \qquad (2\text{-}4)$$

De acuerdo al orden de multiplicidad de cada valor propio λ_i se pueden hallar k_i vectores propios linealmente independientes que arreglados en forma de una matriz no-singular Q, pueden usarse para transformar la matriz A en una matriz diagonal que tenga la forma canónica de Jordan. Sea J la matriz en la forma canónica de Jordan, expresada por la relación:

$$J = Q^{-1}AQ = \begin{bmatrix} j_1 & 0 & \ldots & 0 \\ 0 & j_2 & \ldots & 0 \\ . & . & . & . \\ 0 & 0 & \ldots & j_k \end{bmatrix} \qquad (2\text{-}5)$$

donde J_i es un bloque de Jordan k_i x k_i. Cada submatriz J_i tendrá como elemento de su diagonal principal a λ_i. Si λ_i es repetido, sobre la diagonal principal aparecerá una diagonal de 1 y los restantes elementos de la matriz serán cero.

Se define una matriz columna, o un vector **z,** obtendido a través de la matriz Q, cambiando el espacio de estado, que cumple la siguiente ecuación:

$$\mathbf{x} = Q\mathbf{z} \tag{2-6}$$

De las ecuaciones (2-5), (2-6) y (2-1) se puede obtener:

$$\dot{\mathbf{z}} = J\mathbf{z} + Q^{-1}B\mathbf{u} \tag{2-7}$$

Se puede particionar **z** en dos vectores $\mathbf{z_1}$ y $\mathbf{z_2}$ de orden mx1 y (n-m)x1 respectivamente, siendo m el orden del modelo reducido buscado. Como los valores propios de la matriz A han sido arreglados en orden ascendente, se puede seleccionar Q de modo que los valores propios dominantes de A estén presentes en J_1, J_2, ..., J_d, y los restantes en J_{d+1}, J_{d+2}, ..., J_k. Reescribiendo entonces la ecuación (2-7) se tiene:

$$\begin{bmatrix} \dot{\mathbf{z}}_1 \\ \dot{\mathbf{z}}_2 \end{bmatrix} = \begin{bmatrix} J_M & 0 \\ 0 & J_N \end{bmatrix} \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix} + \begin{bmatrix} B_M \\ B_N \end{bmatrix} \mathbf{u} \tag{2-8}$$

donde J_M es una matriz m x m que consiste de submatrices J_1, J_2, ..., J_d. J_N es una matriz (n-m) x (n-m) que consiste de submatrices J_{d+1}, J_{d+2}, ..., J_k. B_M son las primeras m filas de $Q^{-1}B$ y B_N son las restantes filas de $Q^{-1}B$. Los valores propios de la matriz J_M serán los valores propios dominantes, de modo que se asume que los valores propios de J_N tienen un pequeñísimo efecto sobre el sistema y pueden ser despreciados. Así, el sistema de orden reducido es obtenido ahora como:

$$\dot{\mathbf{z}}_1 = J_M\mathbf{z}_1 + B_M\mathbf{u} \tag{2-9}$$

Comparando las ecuaciones (2-9) y (2-2) se tiene las siguientes relaciones:

$$A^* = J_M \tag{2-10}$$

$$B^* = B_M \tag{2-11}$$

$$\mathbf{x}^* = \mathbf{z}_1 \tag{2-12}$$

La relación entre los vectores $\mathbf{x}$ y $\mathbf{x}^*$ o lo que es lo mismo, entre $\mathbf{x}$ y $\mathbf{z}_1$ puede ser determinada a partir de la ecuación (2-6) la cual puede ser reescrita de la siguiente manera:

$$\begin{bmatrix} \mathbf{x}_1 \\ \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} Q_{11} & Q_{12} \\ \\ Q_{21} & Q_{22} \end{bmatrix} \begin{bmatrix} \mathbf{z}_1 \\ \\ \mathbf{z}_2 \end{bmatrix} \tag{2-13}$$

$$x_1 = Q_{11}z_1 + Q_{12}z_2 \tag{2-14}$$
$$x_2 = Q_{21}z_1 + Q_{22}z_2 \tag{2-15}$$

Como se ha despreciado $\mathbf{z}_2$ y teniendo en cuenta que la salida está dada por la ecuación

$$\mathbf{y} = C\mathbf{x} \tag{2-16}$$

no es difícil deducir que se puede usar la siguiente ecuación para obtener el vector $\mathbf{x}$:

$$\mathbf{x} = \begin{bmatrix} Q_{11} \\ \\ Q_{21} \end{bmatrix} \mathbf{z}_1 \tag{2-17}$$

Chidambara demuestra [7] que se debe ser muy cuidadoso en despreciar ciertos términos de la ecuación (2-8)y usar las ecuaciones (2-16) y (2-17) para hallar la respuesta del sistema reducido. El indica que la parte correspondiente a $\mathbf{z}_2$ en la ecuación (2-8) no puede ser totalmente despreciada, como lo hace Davison en su método, ya que $\mathbf{z}_2$ a más de contener términos exponenciales con constantes de tiempo de alto orden, contiene términos que son las respuestas forzadas debidas a las entradas usadas para el sistema, y que tienen un efecto significativo en la ganancia de estado estable. Al despreciar totalmente $\mathbf{z}_2$ se pierden dichos términos y la ganancia del sistema reducido

nunca será la misma que aquella del sistema original. Todo esto puede demostrarse al resolver la ecuación de estado del sistema original.

La solución de la ecuación de estado del sistema original, expresada en la forma canónica de Jordan, como lo indica la ecuación (2-7) estará dada por [9]:

$$z(t) = e^{jt}z(0) + \int_0^t e^{j(t-t)}Q^{-1}Bu(t)dt \qquad (2\text{-}18)$$

Considerando que **z(0)** es cero, la solución a la ecuación (2-18) estará dada solamente por la resolución del integral. Si consideramos que $Q^{-1}B$ está dada por:

$$Q^{-1}B = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1p} \\ b_{21} & b_{22} & \dots & b_{2p} \\ . & . & . & . \\ . & . & . & . \\ b_{n1} & b_{n2} & \dots & b_{np} \end{bmatrix} \qquad (2\text{-}19)$$

y

$u(t) = 1$ para $t \geq 0$
$u(t) = 0$ para $t < 0$ (2-20)

entonces **z(t)** estará dada por:

$$z(t) = \int_0^t \begin{bmatrix} e^{\lambda_1(t-\tau)}(b_{11} + b_{12} + \bullet\bullet\bullet + b_{1p}) \\ e^{\lambda_2(t-\tau)}(b_{21} + b_{22} + \bullet\bullet\bullet + b_{2p}) \\ \vdots \qquad\qquad \vdots \\ e^{\lambda_n(t-\tau)}(b_{n1} + b_{n2} + \bullet\bullet\bullet + b_{np}) \end{bmatrix} d\tau \qquad (2\text{-}21)$$

Resolviendo se tiene:

$$z(t) = \begin{bmatrix} \frac{-1+e^{\lambda_1 t}}{\lambda_1}(b_{11} + b_{12} + \bullet\bullet\bullet + b_{1p}) \\ \frac{-1+e^{\lambda_2 t}}{\lambda_2}(b_{21} + b_{22} + \bullet\bullet\bullet + b_{2p}) \\ \bullet \qquad\qquad \bullet \\ \bullet \qquad\qquad \bullet \\ \bullet \qquad\qquad \bullet \\ \frac{-1+e^{\lambda_n t}}{\lambda_n}(b_{n1} + b_{n2} + \bullet\bullet\bullet b_{np}) \end{bmatrix} \qquad (2\text{-}22)$$

A través de las ecuaciones (2-6) y (2-16) se puede encontrar la función de la señal de salida del sistema. Los términos constantes de la ecuación (2-22) como:

$$\frac{-1}{\lambda_i}(b_{i1} + b_{i2} + \ldots + b_{in}) \quad i = 1,2,\ldots,p \qquad (2\text{-}23)$$

multiplicados por sendas constantes producto de las matrices de las ecuaciones (2-6) y (2-16) darán el valor de la ganancia de estado estable de la salida y(t).

Al eliminar $\mathbf{z_2}$ de la ecuación (2-8) se están eliminando términos constantes del tipo

$$K\,\frac{-1}{\lambda_i}(b_{i1} + b_{i2} + \ldots + b_{in}) \quad i = m,m+1,\ldots,p$$

donde K es una constante producto de las matrices Q y C de las ecuaciones (2-6) y (2-16) respectivamente, los cuales influyen directamente en el valor de la ganancia en estado estable del sistema. Estos son los términos que no pueden ser eliminados para que no exista diferencia entre la ganancia en estado estable del sistema original y aquella del sistema equivalente reducido. Los términos exponenciales eliminados a través de $\mathbf{z_2}$, por poseer constantes de tiempo altas, realmente no tienen mayor influencia en la respuesta del sistema.

De la ecuación (2-8) se observar que $\mathbf{z_2}$ está dada por:

$$\dot{\mathbf{z}}_2 = J_N \mathbf{z}_2 + B_N \mathbf{u} \qquad (2\text{-}25)$$

La solución complemetaria de la ecuación (2-25) contiene términos exponenciales cu-

yas constantes de tiempo tienen un efecto despreciable en la respuesta de $\mathbf{z_2}$ comparados con aquellos productos del integral. Por lo tanto, solo será considerado el integral en la solución de (2-25). Dicho integral, correspondiente a cualquier entrada, será dicha entrada multiplicada por una constante y generalmente retardada un cierto intervalo de tiempo. En el presente caso, como las constantes de tiempo del sistema representado por la ecuación (2-25) son muy pequeñas, el retardo de tiempo en el integral es pequeño y despreciable. Así, el integral correspondiente a la variable de estado z_k, la cual es un elemento de $\mathbf{z_2}$ está dado por:

$$z_k = \sum_{i=1}^{p} d_{ki} u_i \qquad (2\text{-}26)$$

donde p = número de entradas y d_{ki} son constantes. Si z_k no es afectada por ninguna entrada u_b, entonces, obviamente, $d_{kb}=0$.

Procediendo en forma similar para todos los estados de $\mathbf{z_2}$ se puede demostrar que:

$$\mathbf{z_2} = D\mathbf{u}$$

donde **D** es una matriz constante de orden (n-m)xp. Un método de determinar D es a través de la transformada de Laplace.

$$\mathbf{Z_2}(s) = G_2(s)U(s) \qquad (2\text{-}28)$$

siendo

$$D = G_2(0) \qquad (2\text{-}29)$$

y

$$G_2(s) = [sI - J_N]^{-1} B_N \qquad (2\text{-}30)$$

Así, el modelo de orden reducido obtenido por el método de Chidambara puede ser expresado como:

$$\dot{\mathbf{x}}^* = J_M \mathbf{x}^* + B_M \mathbf{u}$$

$$x = \begin{bmatrix} Q_{11} \\ Q_{21} \end{bmatrix} x^* + \begin{bmatrix} Q_{12} G_2(0) \\ Q_{22} G_2(0) \end{bmatrix} u \qquad (2\text{-}31)$$

$$\mathbf{y} = C\mathbf{x}$$

Si $G_2(0) = 0$ entonces el modelo (2-20) es idéntico al modelo presentado en (2-17).

Como puede verse, esta es la relación que utiliza Chidambara, usando los polos despreciados del sistema original, para mejorar la respuesta en estado estable del modelo reducido.

A continuación se presenta un ejemplo explicativo de todo el desarrollo teórico matemático mostrado:

Sea el sistema G(s), representado por la función de transferencia siguiente:

$$G(s) = \frac{(s+5)}{(s+1)(s+2)(s+20)} = \frac{(s+5)}{s^3 + 23s^2 + 62s + 40} \qquad (2\text{-}32)$$

El sistema expresado en variables de estado tendrá la forma:

$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \qquad y = C\mathbf{x}$

con:

$$A = \begin{bmatrix} 0 & 0 & -40 \\ 1 & 0 & -62 \\ 0 & 1 & -23 \end{bmatrix} \qquad B = \begin{bmatrix} 5 \\ 1 \\ 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \qquad (2\text{-}33)$$

La matriz Q, formada con los vectores propios del sistema, para efectuar el cambio del espacio de estado, estará dada por:

$$Q = \begin{bmatrix} 40 & 20 & 2 \\ 22 & 21 & 3 \\ 1 & 1 & 1 \end{bmatrix} \qquad (2\text{-}34)$$

El sistema expresado en términos de **z**, según la ecuación (2-7) estará dado por:

$$\dot{z} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -20 \end{bmatrix} z + \begin{bmatrix} 0.21053 \\ -0.16667 \\ -0.04386 \end{bmatrix} u \qquad (2\text{-}35)$$

$\mathbf{y} = C\mathbf{x}$

Si se toma la señal de entrada como un escalón unitario, y **z(0) = 0**, y resolvemos la ecuación de estado en términos de z, (aplicando la ecuación 2-18) se tiene:

$$z(t) = \begin{bmatrix} e^{-(t-\tau)} & 0 & 0 \\ 0 & e^{-2(t-\tau)} & 0 \\ 0 & 0 & e^{-20(t-\tau)} \end{bmatrix} \begin{bmatrix} 0.21053 \\ -0.16667 \\ -0.04386 \end{bmatrix} 1 \, d\tau \qquad (2\text{-}36)$$

Resolviendo se tiene:

$$z(t) = \begin{bmatrix} 0.211 - 0.211e^{-t} \\ -0.083 + 0.083e^{-2t} \\ -0.003 + 0.0022e^{-20t} \end{bmatrix} \qquad (2\text{-}37)$$

Aplicando las ecuaciones **x** = Q**z** e **y** = C**x**, la salida **y** estará dada por:

$$y(t)=0.211-0.211e^{-t}-0.083+0.083e^{-2t}-0.0022+0.0022e^{-20t} \qquad (2\text{-}38)$$

Términos que dan la ganancia en estado estable

Al eliminar la parte correspondiente a $\mathbf{z_2}$ de la partición indicada en la ecuación (2-8) se tiene:

$$z_1 = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix} z + \begin{bmatrix} 0.21053 \\ -0.16667 \end{bmatrix} u \qquad (2\text{-}39)$$

Resolviendo $\mathbf{z_1}$ para obtener el sistema reducido, se tiene:

$$z_1(t) = \begin{bmatrix} e^{-(t-\tau)} & 0 \\ 0 & e^{-2(t-\tau)} \end{bmatrix} \begin{bmatrix} 0.21053 \\ -0.16667 \end{bmatrix} 1 \, d\tau \qquad (2\text{-}40)$$

lo que da como resultado:

$$z_1(t) = \begin{bmatrix} 0.211 - 0.211e^{-t} \\ -0.083 + 0.083e^{-2t} \end{bmatrix} \tag{2-41}$$

Aplicando la ecuación (2-17) para obtener el vector **x** se tiene:

$$x = \begin{bmatrix} 40 & 20 \\ 22 & 21 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 0.211 - 0.211e^{-t} \\ -0.083 + 0.083e^{-2t} \end{bmatrix} \tag{2-42}$$

Realizando y = C**x**, la respuesta del sistema reducido será:

$$y = 0.211 - 0.211e^{-t} - 0.083 + 0.083e^{-2t} \tag{2-43}$$

Parámetros que dan la ganancia en estado estable

Comparando las ecuaciones (2-43) y (2-38) se puede observar que en la ecuación (2-43) falta un parámetro para llegar al mismo valor de ganancia de estado estable que el sistema original de la ecuación (2-38). La ausencia de ese parámetro se debe a la eliminación de $\mathbf{z_2}$.

Sin embargo, Chidambara, como ya se ha explicado, usa el sistema dado por $\mathbf{z_2}$ para hacer la corrección de la ganancia. De la ecuación (2-35) se tiene:

$$\mathbf{z_2} = -20\mathbf{z_2} - 0.04386\mathbf{u} \tag{2-44}$$

Aplicando la transformada de Laplace a la ecuación anterior, y resolviendo se tiene:

$$G_2 = \frac{z_2}{\mathbf{u}} = \frac{-0.4386}{s + 20} \tag{2-45}$$

Como D = $G_2(0)$, se tiene que D = -0.0022

Aplicando la ecuación (2-31) se tiene:

$$x = \begin{bmatrix} 40 & 20 \\ 22 & 21 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 0.211 - 0.211e^{-t} \\ -0.083 + 0.083e^{-2t} \end{bmatrix} + \begin{bmatrix} -0.0044 \\ -0.0066 \\ -0.0022 \end{bmatrix} \qquad (2\text{-}46)$$

Resolviendo y realizando y = C**x**, se tiene:

$$y = 0.211 - 0.211e^{-t} - 0.083 + 0.083e^{-2t} - 0.0022 \qquad (2\text{-}47)$$

Términos similares que en ecuación (2-38)

De la ecuación (2-43) se puede observar que se consigue el valor de la ganancia en estado estable del sistema original, tal como lo expresa la ecuación (2-38).

El gráfico 2-1 muestra las respuestas en el tiempo, indicadas en las ecuaciones (2-38), (2-43) y (2-47).

2.2 Criterio para la selección del orden de la función de transferencia equivalente.-

Por medio del método presentado a continuación, se puede tener un criterio de selección del orden del modelo reducido que se desea obtener, a partir del modelo original de alto orden [10], [11].

Considérese un sistema lineal, invariante en el tiempo representado por la ecuación de estado indicada en (2-1), con condiciones iniciales nulas. Dicha ecuación de estado puede ser reescrita en la siguiente forma particionada:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \mathbf{u} \qquad (2\text{-}48)$$

donde x_1 es de orden m, siendo m el número de estados a ser retenido, o el orden el sistema reducido equivalente que se desea obtener.

Considérese la siguiente transformación lineal:

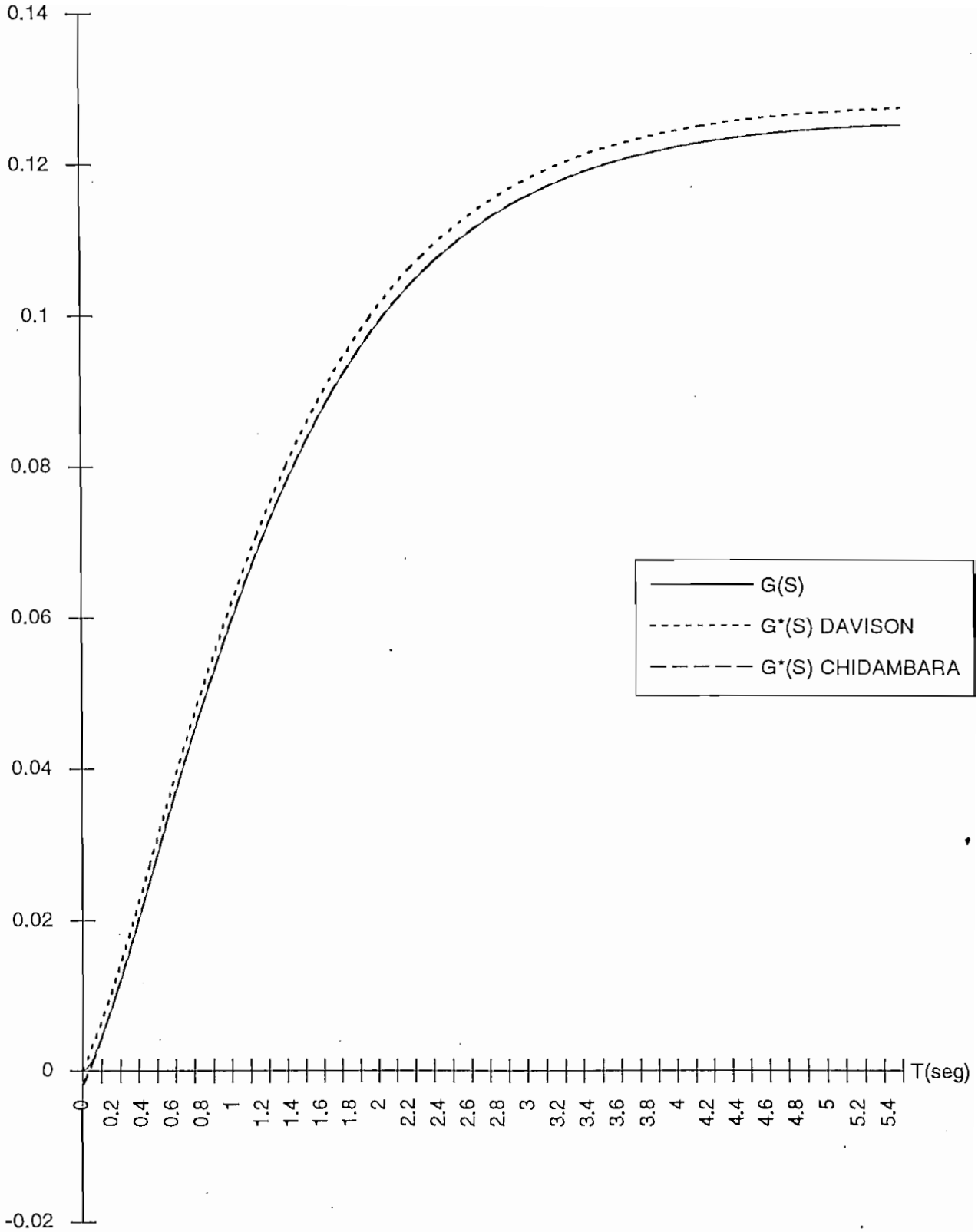

0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.02
T(seg)
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
4.2
4.4
4.6
4.8
5
5.2
5.4
G(S)
G*(S) DAVISON
G*(S) CHIDAMBARA

Gráfico 2-1

$$\mathbf{x} = P\mathbf{z} \tag{2-49}$$

donde P es la matriz modal de A. Entonces la forma normal de la ecuación de estado puede ser escrita como:

$$\dot{\mathbf{z}} = D\mathbf{z} + P^{-1}B\mathbf{u} \tag{2-50}$$

donde

$$D = P^{-1}AP$$

Usando las ecuaciones (2-48), (2-49) y (2-50), se puede particionar al vector $\mathbf{z}$ en $\mathbf{z}_1$, de orden m, y $\mathbf{z}_2$, de orden n-m, obteniéndose las siguientes ecuaciones:

$$\dot{\mathbf{z}}_1 = D_1\mathbf{z}_1 + R_1\mathbf{u} \tag{2-51}$$

$$\dot{\mathbf{z}}_2 = D_2\mathbf{z}_2 + R_2\mathbf{u} \tag{2-52}$$

donde,

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \tag{2-53}$$

$$P^{-1} = V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \tag{2-54}$$

$$D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \tag{2-55}$$

$$P^{-1}B = \begin{bmatrix} R_1 \\ R_2 \end{bmatrix} = \begin{bmatrix} V_{11}B_1 + V_{12}B_2 \\ V_{21}B_1 + V_{22}B_2 \end{bmatrix} \tag{2-56}$$

Con el propósito de simplificar la deducción del criterio, se asume que todos los valores propios de A son distintos. Entonces, D será una matriz diagonal. Sea D entonces:

$$D = \text{diag}\ [\lambda_1, \lambda_2, \lambda_3, ..., \lambda_m, \lambda_{m+1}, ..., \lambda_n] \tag{2-57}$$

donde λ_i son valores propios complejos y la parte real de λ_i es mayor o igual que la parte real de λ_{i+1} para i= 1,2,...n-1.

De (2-50) y (2-53) se puede obtener:

$$\mathbf{x_1} = P_{11}z_1 + P_{12}z_2 \tag{2-58}$$

Con el fin de que el sistema reducido sea una buena aproximación al modelo original, el orden m debe ser seleccionado de modo que el error producido al eliminar los valores propios $\lambda_{m+1}, ..., \lambda_n$, sea lo más pequeño posible.
Dicho error vendrá dado por la ecuación [11]:

$$E = P_{12}[z_2 + D_2^{-1}R_2u] \tag{2-59}$$

$$E(t) = P_{12}\left[\int_0^t e^{D_2(t-\tau)}R_2u(\tau)d\tau + D_2^{-1}R_2u(t)\right] \tag{2-60}$$

$$\|E(t)\| \le \|P_{12}\|\left[\int_0^t \|e^{D_2(t-\tau)}\|\ \|R_2\|\ \|u(\tau)\|d\tau + \|D_2^{-1}\|\ \|R_2\|\ \|u(t)\|\right] \tag{2-61}$$

$$\|P_{12}\| < \|P\|;\ \ \|R_2\| < \|V\|\ [\|B_1\| + \|B_2\|] \tag{2-62}$$

Sea u(t) una función escalón unitario. De las ecuaciones (2-60), (2-61) y (2-62) se tiene:

$$\begin{aligned}\|E(t)\| &< u_o\|P\|\ \|V\|\ [\|B_1\| + \|B_2\|]\left[\int_0^t \|e^{D_2(t-\tau)}\|d\tau + \|D_2^{-1}\|\right] \\ &< K\|D_2^{-1}\|\ [\|e^{D_2t} - I\| + 1]\end{aligned} \tag{2-63}$$

donde

$$K = u_o \|P\| \|V\| \left[\|B_1\| + \|B_2\|\right] \tag{2-64}$$

Ahora se pueden obtener los valores de $\|D_2^{-1}\|$ y $\|e^{D_2 t - I}\|$ para el caso general en el que D_2 tiene valores propios complejos, $\lambda_i = -\sigma_i - j\omega_i$, con $\lambda_i > 0$ y $\omega_i > 0$.

Se tiene que:

$$\left\|e^{D_2 t - I}\right\| = \left\|\mathrm{Diag}\left[e^{\lambda_{m+1} t} - 1,\ e^{\lambda_{m+2} t} - 1,\ \ldots,\ e^{\lambda_n t} - 1\right]\right\| \tag{2-65}$$

Pero

$$\left\|e^{\lambda_i t} - 1\right\| \leq 2 \tag{2-66}$$

Por lo tanto,

$$\left\|e^{D_2 t} - \mathbf{I}\right\| \leq 2\sqrt{n-m} \tag{2-67}$$

En forma similar,

$$\left\|D_2^{-1}\right\| = \left\|\sum_{i=m+1}^{n} \frac{\sigma_i \pm j\omega_i}{\sigma_i^2 + \omega_i^2}\right\| < \frac{\sqrt{n-m}}{\min\left|\lambda_i\right|} \tag{2-68}$$

donde $m+1 \leq i \leq n$. Por lo tanto

$$\|e(t)\| < K\bar{U}_m$$

Entonces, la función para el criterio de selección puede escribirse como [10]:

$$\overline{U}_m = \frac{\sqrt{n-m}}{\min|\lambda_i|}\left[2\sqrt{n-m}+1\right]; \quad m+1 \le i \le n \tag{2-69}$$

donde $\overline{U}_m$ puede ser vista como una función de error. Mientras más pequeño sea el valor de Um, mejor será la aproximación del modelo reducido.

2.3 Método de Agregación.-

El método de agregación fue propuesto por Aoki [12] para obtener un modelo reducido de un sistema dinámico con un vector de estado de alta dimensión. El modelo se deriva de la agregación del vector de estado del sistema original, en un vector de estado de menor dimensión. Este método se basa en determinar la "matriz de agregación", la cual da la relación entre las variables de estado del sistema original y del modelo reducido.

Sean dos sistemas dinámicos lineales, S_1 y S_2, donde la dimensión de S_1, n, es mucho más grande que la de S_2, m. Dichos sistemas dinámicos y continuos en el tiempo, pueden ser expresados por las siguientes ecuaciones de estado:

$$S_1 \qquad \dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$$
$$\mathbf{y} = C\mathbf{x} \tag{2-70}$$

$$S_2 \qquad \dot{\mathbf{x}}^* = A^*\mathbf{x}^* + B^*\mathbf{u}$$
$$\mathbf{y}^* = C^*\mathbf{x}^* \tag{2-71}$$

donde, **x** es n-dimensional, **x*** es m-dimensional, **y*** es la aproximación de **y**, **u** es p-dimensional y las matrices constantes A, B, C, A*, B*, C*, son de la dimensión adecuada para que cumplan con dichas ecuaciones.

Los vectores de estado de los dos sistemas, notados por **x** y **x*** respectivamente, pueden ser relacionados por la siguiente ecuación:

$$\mathbf{x}^* = K\mathbf{x} \tag{2-72}$$

donde K es una matriz constante de orden mxn. El sistema S_2 puede ser considerado como un modelo de S_1 ya que la estructura dinámica de S_2 es de tal manera que po-

drá reflejar las características más significativas de la dinámica de S_1. Por ejemplo, si S_1 es la descripción de un objeto físico o no físico de acuerdo a una cierta clasificación de variables, S_2 sería la descripción del mismo objeto usando una clasificación burda, no tan fina ni específica, de las mismas variables. De ahí su baja dimensión.

La matriz K, llamada "matriz de agregación" es considerada como una matriz de proyección del sistema de alto orden en uno de orden reducido. A* es considerada como la matriz agregada o como la agregación de la matriz A. El vector agregado **x*** satisface la ecuación dinámica (2-71) y de las ecuaciones (2-70) a (2-72) se pueden obtener las siguientes relaciones:

$$A^*K = KA \tag{2-73}$$

$$B^* = KB \tag{2-74}$$

$$C^* = CK' \tag{2-75}$$

Para el cálculo de las matriz C* se usa la matriz pseudo-inversa de K, ya que K, generalmente, no será una matriz cuadrada. Dicha matriz pseudo-inversa, K', vendrá dada por:

$$K' = K^t(KK^t)^{-1}$$

En el presente método, la matriz de agregación es obtenida a partir de la
del sistema original, aquella formada por los vectores propios y que e
cambiar el espacio de estado. Exactamente, dicha matriz de agregaci
mo las m primeras filas de la matriz modal normalizada del sistema
el orden del sistema reducido buscado.

Las matrices del sistema reducido se obtienen a partir de las
76).

A continuación se presenta un ejemplo explicativo del proceso

Sea el sistema descrito por la siguiente función de transferenci

$$G(s) = \frac{s+5}{s^3 + 23s^2 + 62s + 40}$$

Expresando en variables de estado, en la forma canónica obs
matrices que definen el sistema serán:

$$A=\begin{bmatrix}0 & 0 & -40\\ 1 & 0 & -62\\ 0 & 1 & -23\end{bmatrix} \qquad B=\begin{bmatrix}5\\ 1\\ 0\end{bmatrix} \qquad C=\begin{bmatrix}0 & 0 & 1\end{bmatrix} \tag{2-78}$$

La matriz formada con los vectores propios de sistema, para cambiar el espacio de estado, y que servirá para hallar la matriz de agregación, estará dada por:

$$Q=\begin{bmatrix}40 & 20 & 2\\ 22 & 21 & 3\\ 1 & 1 & 1\end{bmatrix} \tag{2-79}$$

Normalizando esta matriz se tendrá:

$$Qn=\begin{bmatrix}0.876 & 0.689 & 0.535\\ 0.482 & 0.724 & 0.802\\ 0.022 & 0.034 & 0.267\end{bmatrix} \tag{2-80}$$

La matriz de agregación estará dada por la matriz formada con las primeras m filas de la matriz Qn, siendo m el orden del sistema reducido. En este caso, m=2, por lo tanto, la matriz de agregación, Ag, será:

$$Ag=\begin{bmatrix}0.876 & 0.689 & 0.535\\ 0.482 & 0.724 & 0.802\end{bmatrix} \tag{2-81}$$

La matriz pseudo-inversa de la matriz de agregación, Aig, será (de acuerdo con la ecuación(2-79):

$$Aig=\begin{bmatrix}1.817 & -1.409\\ -0.040 & 0.556\\ -1.056 & 1.592\end{bmatrix} \tag{2-82}$$

De acuerdo con las ecuaciones (2-73), (2-74) y (2-75), las matrices A*, B* y C*, del sistema reducido estarán dadas por:

$$A^* = \begin{bmatrix} 96.365 & -144.083 \\ 88.512 & -132.066 \end{bmatrix} \quad (2\text{-}83)$$

$$B^* = \begin{bmatrix} 5.069 \\ 3.133 \end{bmatrix} \quad (2\text{-}84)$$

$$C^* = \begin{bmatrix} -1.056 & 1.592 \end{bmatrix} \quad (2\text{-}85)$$

La función de transferencia del sistema reducido, hallado a partir de las matrices A*, B* y C* será:

GRADO NUMERADOR: 1

COEFICIENTE GRADO 1 = -0.366377
COEFICIENTE GRADO 0 = 3.373397

GRADO DENOMINADOR: 2

COEFICIENTE GRADO 2 = 1.00000
COEFICIENTE GRADO 1 = 35.70089
COEFICIENTE GRADO 0 = 26.47556

$$G^*(s) = \frac{-0.366377s + 3.373397}{s^2 + 35.70089s + 26.47556} \quad (2\text{-}86)$$

El gráfico 2-2 compara las respuestas en el tiempo, a una señal escalón unitario de las funciones de transferencia indicadas pora las ecuaciones (2-77) y (2-86).

2.4 Método de Realización Parcial.-

Otro de los métodos de modo dominante es este que se ha denominado de Realización Parcial ya que, como se verá a continuación, es necesario emplear un proceso de realización, para obtener el sistema reducido, a partir de una matriz función de transferencia.

Básicamente, el método consiste en expandir al sistema original en una serie de po-

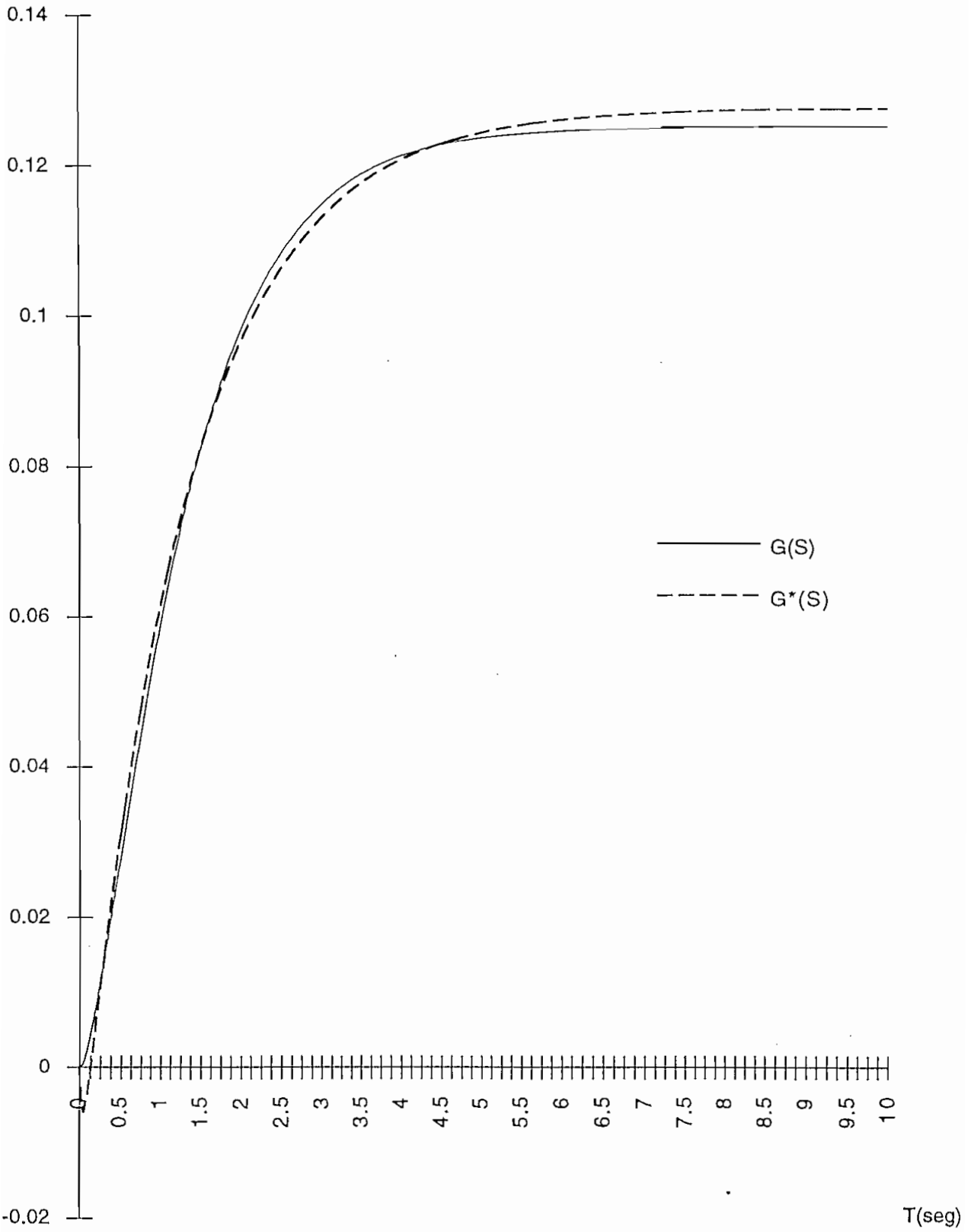

0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.02
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
6.5
7
7.5
8
8.5
9
9.5
10
T(seg)
G(S)
G*(S)

Gráfico 2-2

tencias. Con dichos coeficientes se construirá una matriz, definida como matriz de Hankel, y dependiendo del orden que se escoja para dicha matriz, se obtendrá el orden del sistema reducido. Mediante un proceso de realización se obtendrán las nuevas matrices asociadas a dicho sistema reducido. El escoger un orden arbitrario para la matriz de Hankel es equivalente a truncar la serie de potencias definida a partir del sistema original.

Al expandir el sistema original en una serie de potencias, definimos las llamadas matrices de Markov o parámetros de Markov (ver apéndice), J_i, asociadas a G(s), asumiendo que $G(\infty)$ es finito. Dichas matrices, como ya se mencionó, son los coeficientes de la expanción en matrices serie de potencias de G(s) en potencias de s^{-1}:

$$G(s) = J_{-1} + \frac{J_o}{s} + \frac{J_1}{s^2} + \frac{J_2}{s^3} + \ldots \tag{2-87}$$

La matriz de Hankel correspondiente estará definida por:

$$S_m = \begin{bmatrix} J_{-1} & J_0 & J_1 & \ldots & J_{m-1} \\ J_0 & J_1 & J_2 & \ldots & J_m \\ . & . & . & & . \\ . & . & . & & . \\ J_{m-1} & J_m & J_{m+1} & \ldots & J_{2m-2} \end{bmatrix} \tag{2-88}$$

donde m es el orden del sistema reducido a determinar.

Una vez definida la matriz de Hankel de un determinado orden, Rozsa y Sinha [13] proponen un algoritmo para la obtención de las matrices A^*B^* y C^*, asociadas al sistema reducido. Dicho algoritmo se basa en tranformar la matriz de Hankel en una matriz con forma normal Hermítica. Esta transformación puede ser ejecutada en r pasos, siendo r el rango de la matriz de Hankel.

Se define la forma normal Hermítica, cuando una matriz cuadrada satisface las siguientes condiciones:

a) Es una matriz triangular superior siendo los elementos de la diagonal principal 1 o 0.

b) Si un cierto elemento de la diagonal es 1, todos los demás elementos en la columna correspondiente son 0.

c) Si un cierto elemento de la diagonal es 0, todos los elementos en la fila correspondiente también son 0.

Ahora será descrito un proceso simple para transformar cualquier matriz cuadrada Q en una forma normal Hermítica. (Rozsa 1974).

Sea $q_{i_1 1}$ el primer elemento distinto de cero en la primera columna de Q. Entonces:

$$q_{i_1 1} = e_{i_1}^T Q e_1 \tag{2-89}$$

donde e_j representa un vector columna unidad con un 1 en la j-ésima fila y 0 en todas las demás y el exponente T representa transposición, y e_1 es un vector columna con un 1 en la primera fila y los demás elementos 0.

$$Q = \begin{bmatrix} 0 & . & . & . & . \\ 0 & . & . & . & . \\ & \cdots & & & \\ 0 & . & . & . & . \\ q_{i_1 1} & . & . & . & . \\ x & \cdots & & & \\ & \cdots & & & \\ x & \cdots & & & \end{bmatrix} \longleftarrow \text{fila } i_1 \tag{2-90}$$

Realizando la siguiente operación de sustracción de la matriz Q:

$$Q - \frac{Q e_1 e_{i_1}^T Q}{q_{i_1 1}} \tag{2-91}$$

obtenemos una matriz con ceros en la primera columna e i-ésima fila. Ahora, añadiendo al resultado anterior el producto:

$$\frac{e_{i_1} e_{i_1}^T Q}{q_{i_1 1}} \tag{2-92}$$

es decir:

$$Q - \frac{Qe_1 e_{i_1}^T Q}{q_{i_1 1}} + \frac{e_{i_1} e_{i_1}^T Q}{q_{i_1 1}}$$

se obtiene la matriz:

$$Q^{(2)} = Q - \frac{(Qe_1 - e_{i_1}) e_{i_1}^T Q}{e_{i_1}^T Q e_1} = T^{(1)} Q \qquad (2\text{-}93)$$

la cual tiene solamente un elemento distinto de cero en la primera columna y que precisamente es un 1 en la i_1-ésima fila.

Se puede notar que la matriz $T^{(1)}$ está definida por:

$$T^{(1)} = I - \frac{(Qe_1 - e_{i_1}) e_{i_1}^T}{q_{i_1 1}} \qquad (2\text{-}94)$$

Ahora, sin considerar la i_1-ésima fila de $Q^{(2)}$, se define j_2 como la siguiente columna con, al menos, un elemento distinto de cero, y $q_{i_2 j_2}^{(2)}$ como el primero de ellos, es decir:

$$q_{i_2 j_2}^{(2)} = e_{i_2}^T Q^{(2)} e_{j_2} \neq 0 \qquad (2\text{-}95)$$

Dicha estructura en $Q^{(2)}$ puede ser mostrada en la siguiente manera:

$$Q^{(2)} = \begin{bmatrix} 0 & .. & 0 & \cdots \\ 0 & .. & q_{i_2 j_2} & \cdots \\ & \cdots & & \\ & \cdots & & \\ 0 & .. & x & \\ 1 & .. & x & \\ & \cdots & & \\ 0 & .. & x & \end{bmatrix} \begin{array}{l} \\ \leftarrow \text{fila } i_2 \\ \\ \\ \\ \leftarrow \text{fila } i_1 \\ \\ \end{array}$$

$$\uparrow \qquad \uparrow$$

$$\text{col } 1 \qquad \text{col } j_2$$

Luego, la siguiente matriz en el algoritmo estará dada por:

$$Q^{(3)} = Q^{(2)} - \frac{(Q^{(2)}e_{j_2} - e_{i_2})e_{i_2}^T Q^{(2)}}{e_{i_2}^T Q^{(2)} e_{j_2}} \tag{2-96}$$

Así, en m pasos se consigue una matriz $Q^{(m+1)}$ que puede ser transformada en una forma normal Hermítica. De acuerdo con el algoritmo expuesto, se puede decir que $Q^{(m+1)}$ está dada por:

$$Q^{(m+1)} = T^{(m)}T^{(m-1)} \ldots T^{(1)}Q = TQ \tag{2-97}$$

donde

$$T^{(k)} = I - \frac{(Q^{(k)}e_{j_k} - e_{i_k})e_{i_k}^T}{e_{j_k}^T Q^{(k)} e_{j_k}} \quad \text{con} \quad e_{j_k}^T Q^{(k)} e_{j_k} \neq 0$$

De la ecuación (2-97) se puede decir que:

$$Q = T^{-1}Q^{(m+1)} \tag{2-98}$$

De acuerdo a la definición de la matriz de Hankel y parámetros o matrices de Markov (ver Apendice), esta puede ser descompuesta en dos matrices:

$$S_m = \begin{bmatrix} CA^{-1} \\ C \\ CA \\ . \\ . \\ CA^{m-1} \end{bmatrix} \begin{bmatrix} B & AB & A^2B & \ldots & A^{m-1}B \end{bmatrix} \tag{2-99}$$

$$S_m = V_i U_j \tag{2-100}$$

donde U_j es conocida como la matriz de controlabilidad y V_i es la matriz de observabilidad aumentada el término CA^{-1}. Esta última factorización de la matriz S no es única. Puede ser, por lo tanto, simplificada requiriendo que las r columnas de la matriz B sean las columnas de una matriz identidad I_n de orden n x n. Esto se justifica cuando el rango de B es igual a r, como lo será en el caso de todos los sistemas que tengan r entradas independientes.

Siguendo con el algoritmo para hallar las matrices asociadas al sistema reducido, utilizando la ecuación (2-98), y haciendo $Q = S_m$, se puede decir que:

$$S_m = T^{-1}_{(mxk)} S_{(kxm)} \tag{2-101}$$

donde $S_{(kxm)}$ es una submatriz de S_m con k filas y m columnas, resultante al omitir las filas de ceros que puedan generarse en S_m y $T^{-1}_{(mxk)}$ es obtenida al omitir las columnas correspondientes de T^{-1}.

Para explicar la forma de obtener las matrices A*, B* y C* que definen el sistema reducido, se va a exponer un ejemplo que indique las matrices originales de un cierto sistema, la matriz de Hankel generada, y la misma matriz de Hankel reducida a forma normal Hermítica. A partir de dicho ejemplo, va a ser más fácil darse cuenta, como trabaja el algoritmo para hallar las matrices asociadas al sistema reducido.

Sea la siguiente función de transferencia:

$$G(s) = \frac{s^2 + 10s + 24}{s^3 + 13s^2 + 32s + 20} \tag{2-102}$$

La matriz de Hankel obtenida a partir de dicho sistema, y que para el presente ejemplo será escogida de un orden tal que genere un sistema de tercer orden, es decir, el mismo que el original, estará dada por:

$$S_4 = \begin{bmatrix} -1.200 & 1.000 & -3.000 & 31.000 \\ 1.000 & -3.000 & 31.000 & -327.000 \\ -3.000 & 31.000 & -327.000 & 3319.000 \\ 31.000 & -327.000 & 3319.000 & 33303.000 \end{bmatrix} \tag{2-103}$$

Aplicando el algoritmo expuesto anteriormente, dicha matriz se la reduce a forma normal hermítica, obteniéndose:

$$S_h = \begin{bmatrix} 1.000 & 0.000 & 0.000 & -20.000 \\ 0.000 & 1.000 & 0.000 & -32.000 \\ 0.000 & 0.000 & 1.000 & -13.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \end{bmatrix} \qquad (2\text{-}104)$$

De acuerdo con la ecuación (2-101), se obtiene la matriz $S_{(3x4)}$ al eliminar la última fila (fila de ceros) de la matriz S_h. Por lo tanto, se tiene:

$$S_{(3x4)} = \begin{bmatrix} 1.000 & 0.000 & 0.000 & -20.000 \\ 0.000 & 1.000 & 0.000 & -32.000 \\ 0.000 & 0.000 & 1.000 & -13.000 \end{bmatrix} \qquad (2\text{-}105)$$

Igualando las ecuaciones (2-100) y (2-101), se tiene que

$$U_j = S_{(kxm)} \qquad (2\text{-}106)$$

Comparando la estructura de U_j con la de $S_{(kxm)}$ la matriz B* tendrá un número de columnas de la forma $e_{p_1}, e_{p_2}, \ldots, e_{p_m}$, cada uno siendo un vector unitario con un uno en la fila p_i y los demás elementos igual a cero.

Para el presente ejemplo, comparando la estructura de U_j con la matriz de la ecuación (2-105), la matriz B es obtenida inmediatamente:

$$S_{(3x4)} = \begin{bmatrix} 1.000 & 0.000 & 0.000 & -20.000 \\ 0.000 & 1.000 & 0.000 & -32.000 \\ 0.000 & 0.000 & 1.000 & -13.000 \end{bmatrix} \qquad (2\text{-}107)$$

$$U_j = \begin{bmatrix} \uparrow & \uparrow & \uparrow & \uparrow \\ B & AB & A^2B & A^3B \end{bmatrix}$$

$$B^* = \begin{bmatrix} 1.000 \\ 0.000 \\ 0.000 \end{bmatrix} \qquad (2\text{-}108)$$

Sean $s_q^{(m)}$ los vectores consistentes de un segundo bloque de q filas de la matriz S_m, siendo q el número de salidas del sistema original. De la ecuación (2-99), éste segundo bloque de q filas estará dado por:

$$C \left[B \;\; AB \;\; A^2B \ldots A^{m-1}B \right] = \left[s_1^{(m)} \; s_2^{(m)} \ldots s_m^{(m)} \right] \qquad (2\text{-}109)$$

La e-ésima columna de C*, notada por c_e estará dada por:

$$c_e = s_j^{(m)} \qquad (2\text{-}110)$$

siendo j el índice de la columna correspondiente a los vectores $e_{p_1}, e_{p_2}, \ldots, e_{p_m}$, generados en la matriz reducida a forma normal hermítica, S_h.

Para el ejemplo desarrollado, la matriz C* se hallará de la siguente manera:

$$S_4 = \begin{bmatrix} -1.200 & 1.000 & -3.000 & 31.000 \\ 1.000 & -3.000 & 31.000 & -327.000 \\ -3.000 & 31.000 & -327.000 & 3319.000 \\ 31.000 & -327.000 & 3319.000 & 33303.000 \end{bmatrix}$$

C* (columna 3)

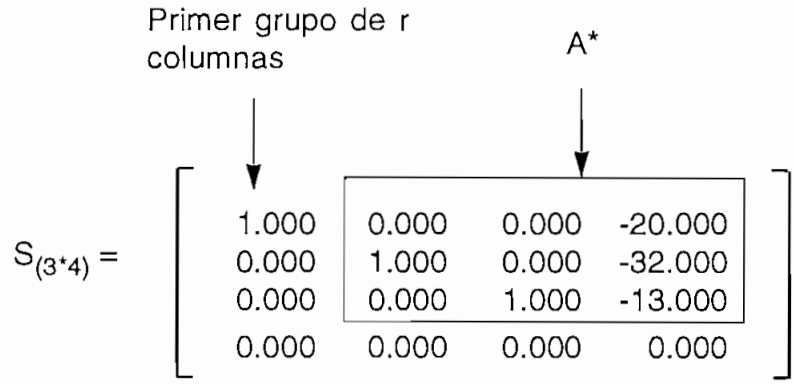
e_1 e_2 e_3

$$S_{(3x4)} = \begin{bmatrix} 1.000 & 0.000 & 0.000 & -20.000 \\ 0.000 & 1.000 & 0.000 & -32.000 \\ 0.000 & 0.000 & 1.000 & -13.000 \end{bmatrix}$$

Para obtener las columnas de A*, se divide las columnas de U_j en grupos de r columnas, siendo r para este caso, el número de entradas del sistema. Notando dichos grupos de columnas de U_j como u_q, q = 1, 2, ..., m, se puede ver claramente que multiplicando cada uno de esos grupos por A*, se obtiene el grupo siguiente. Por lo tanto, las columnas de A^* estarán dadas por:

$$a_k = u_{k+r} \qquad (2\text{-}111)$$

En el ejemplo propuesto, A* estará dada por:



Las matrices A, B y C generadas producen la siguiente función de transferencia:

$$G(s) = \frac{s^2 + 10s + 24}{s^3 + 13s^2 + 32s + 20}$$

Esta función de transferencia es la misma que la indicada en (2-102), comprobándose, como ya se había indicado, que el sistema generado es exacto al original.

En el ejemplo siguiente, se reducirá el mismo sistema a uno de segundo orden:

Sea la función de transferencia de un sistema de tercer orden dado por:

$$G(s) = \frac{s^2 + 10s + 24}{s^3 + 13s^2 + 32s + 20}$$

La matriz de Hankel generada vendrá dada por:

$$\begin{bmatrix} -1.200 & 1.000 & -3.000 \\ 1.000 & -3.000 & 31.000 \\ -3.000 & 31.000 & -327.000 \end{bmatrix}$$

La matriz de Hankel reducida tendrá la forma:

$$\begin{bmatrix} 1.000 & 0.000 & -8.462 \\ 0.000 & 1.000 & -13.154 \\ 0.000 & 0.000 & 55.385 \end{bmatrix}$$

La matriz de Hankel reducida a forma normal hermítica vendrá dada por:

$$\begin{bmatrix} 1.000 & 0.000 & -8.462 \\ 0.000 & 1.000 & -13.154 \\ 0.000 & 0.000 & 0.000 \end{bmatrix}$$

La función de tranferencia reducida estará dada por la siguiente función de transferencia:

$$G^*(s) = \frac{s + 10.1538}{s^2 + 13.1538s + 8.4615} \qquad (2\text{-}112)$$

El gráfico 2-3 muestra las respuestas en el tiempo ante una entrada escalón unitario de las funciones de tranferencia de los sistema original y reducido, dadas por las ecuaciones (2-102) y (2-112) respectivamente.

Cabe señalar que para este método, el orden del sistema reducido no puede ser dos veces menor que la diferencia entre los grados de los polinomios del denominador y del numerador del sistema original, debido a que la matriz de Hankel escogida para generar el sistema de orden reducido se configura con un rango menor al orden de la misma, lo que hace imposible la aplicación del algoritmo de realización, como podemos ver en el siguiente ejemplo, para el cual, el orden del sistema reducido no podría se menor a cuatro (grado denominador original = 5, grado numerador original = 0):

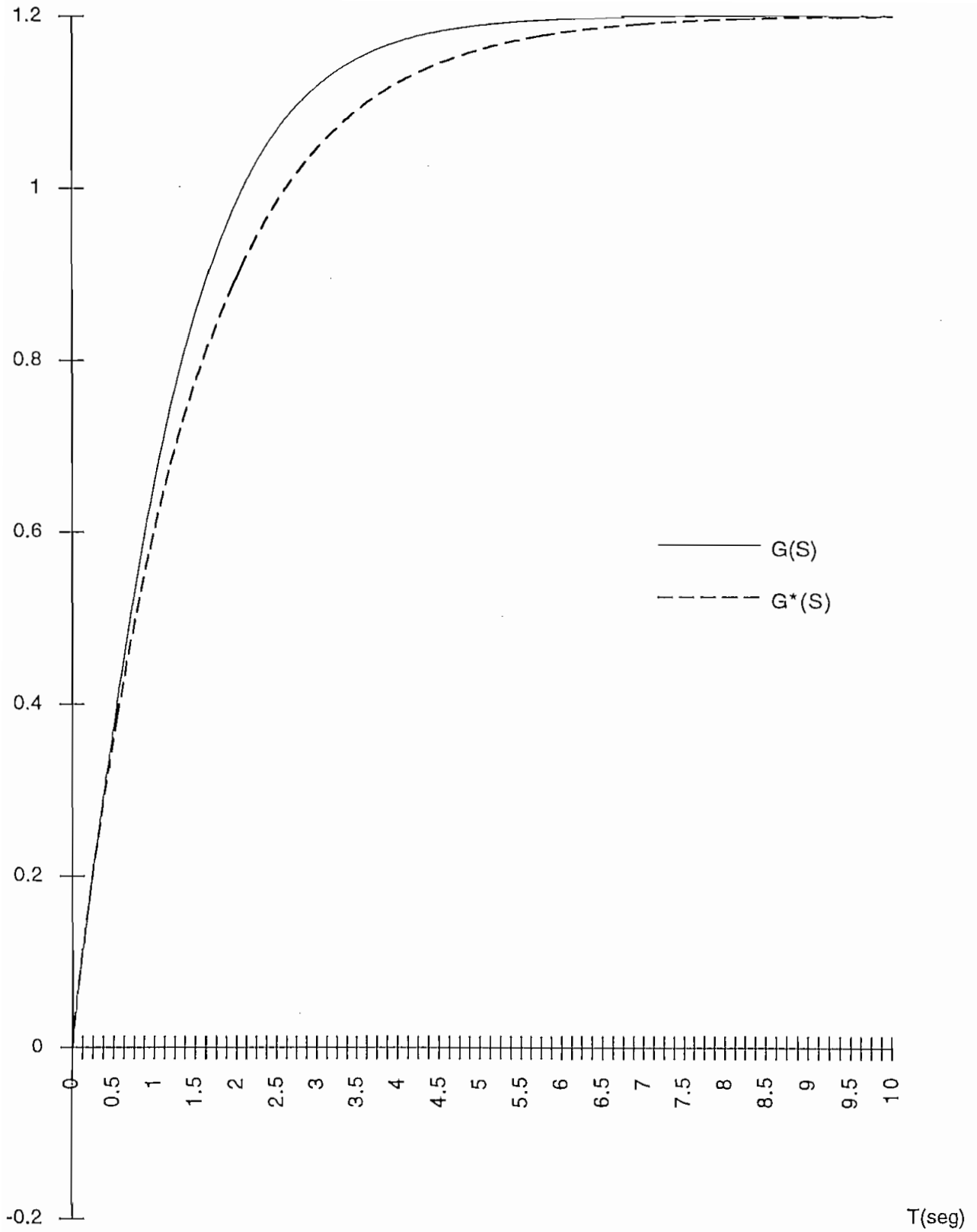

1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
6.5
7
7.5
8
8.5
9
9.5
10
T(seg)
G(S)
G*(S)

Gráfico 2-3

Sea la función de transferencia de quinto orden dada por la ecuación:

$$G(s) = \frac{5}{s^5+13.3s^4+57.92s^3+88.46s^2+22.68s+1.44} \qquad (2\text{-}113)$$

Se escoge un sistema de tercer orden como sistema reducido.

La matriz de Hankel generada por la ecuación (2-113) vendrá dada por

$$\begin{bmatrix} -3.472 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 5.000 \\ 0.000 & 0.000 & 5.000 & -66.500 \end{bmatrix}$$

Se generan columnas con ceros
lo que hace imposible aplicar
el algoritmo de solución.

CAPITULO III

METODOS DE APROXIMACIONES OPTIMAS

El presente capítulo estudia los modelos de reducción de sistemas discretos. Uno de los métodos de reducción a exponerse se basa en el método de los mínimos cuadrados, sobre el cual existen varios trabajos sobre su análisis. Se incluye también la determinación de la función de transferencia en el plano s a partir del modelo reducido obtendio en el plano z.

3.1.– Estimación del modelo reducido desde el muestreo de una respuesta impulso.–

Se asume que se conoce el muestreo de la respuesta a una entrada impulso unitario de un sistema de un cierto orden determinado. La función de transferencia del sistema en tiempo discreto, H(z), puede ser estimada fácilmente y en ésto se basan los métodos expuestos en el presente capítulo. Una vez determinado el sistema reducido en tiempo discreto, no es difícil determinar el sistema contínuo equivalente de un orden menor al sistema original.

3.1.1.– Estimación de la función de transferencia en tiempo discreto.–

Sea una función de transferencia en tiempo discreto, de orden n, H(z) [3]:

$$H(z) = \frac{d_0 + d_1 z^{-1} + d_2 z^{-2} + \ldots + d_n z^{-n}}{1 + c_1 z^{-1} + c_2 z^{-2} + \ldots + c_n z^{-n}} \qquad (3\text{-}1)$$

Sea w_i (i= 0,1,2,3,...) la respuesta impulso de un sistema a t = iT, donde T es el período de muestreo, es decir:

$$w_i = w(iT) \qquad (3\text{–}2)$$

Se puede decir, por lo tanto, que H(z) también tiene la forma:

$$H(z) = w_0 + w_1 z^{-1} + w_2 z^{-2} + \ldots \qquad (3\text{–}3)$$

Igualando las ecuaciones (3–1) y (3–3) se tiene:

$$w_o + w_1 z^{-1} + w_2 z^{-2} + \ldots = \frac{d_o + d_1 z^{-1} + \ldots + d_n z^{-n}}{1 + c_1 z^{-1} + \ldots + c_n z^{-n}}$$

$$(w_o + w_1 z^{-1} + w_2 z^{-2} + \ldots)(1 + c_1 z^{-1} + \ldots + c_n z^{-n}) = d_o + d_1 z^{-1} + \ldots + d_n z^{-n}$$

Realizando operaciones e igualando términos semejantes de acuerdo a las potencias de z se tiene:

$$d_0 = w_0$$
$$d_1 = w_1 + w_0 c_1$$
$$\ldots$$
$$d_n = \sum_{i=0}^{n} c_i w_{n-i}$$

Generalizando y resumiendo, se puede decir que:

$$d_j = \sum_{i=0}^{j} c_i w_{n-i} \qquad (j=0, 1, 2, \ldots, n) \tag{3-4}$$

$$w_{n+k} + \sum_{i=1}^{n} c_i w_{n+k-i} = 0 \qquad (k = 1, 2, \ldots) \tag{3-5}$$

Como la ecuación (3–5) no contiene coeficientes d_j, se puede considerar términos conteniendo $z^{-(n+1)}$ hasta z^{2n} para determinar los coeficientes c_i. Rearreglando la ecuación (3–5) en forma de sistema de ecuaciones se tiene:

$$\begin{bmatrix} w_1 & w_2 & \cdots & w_n \\ w_2 & w_3 & \cdots & w_{n+1} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ w_n & w_{n+1} & \cdots & w_{2n-1} \end{bmatrix} \begin{bmatrix} c_n \\ c_{n-1} \\ . \\ . \\ . \\ c_1 \end{bmatrix} = \begin{bmatrix} -w_{n+1} \\ -w_{n+2} \\ . \\ . \\ . \\ -w_{2n} \end{bmatrix} \qquad (3\text{-}6)$$

Una vez hallados los coeficientes c, se pueden hallar los d de la ecuación (3–4), esto es:

$$\begin{bmatrix} d_o \\ d_1 \\ d_2 \\ . \\ . \\ . \\ d_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ c_1 & 1 & 0 & \cdots & 0 & 0 \\ c_2 & c_1 & 1 & \cdots & 0 & 0 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ c_n & c_{n-1} & c_{n-2} & \cdots & c_1 & 1 \end{bmatrix} \begin{bmatrix} w_o \\ w_1 \\ w_2 \\ . \\ . \\ . \\ w_n \end{bmatrix} \qquad (3\text{-}7)$$

De las dos últimas ecuaciones, se determina el sistema reducido en el plano z. Cabe señalar, que dicho método obliga a muestrar la respuesta impulso del sistema original el doble de veces más uno, el orden de reducción del sistema requerido, es decir, si se necesita un sistema equivalente reducido de segundo orden, se tendrá que muestrear cinco señales de la respuesta del sistema original, en los instantes 0, T, 2T, 3T y 4T.

3.1.2.– Determinación de la función de transferencia del modelo reducido en el plano s.–

Dadas las representaciones en el espacio de estado de un sistema en tiempo discreto y su equivalente en tiempo contínuo, respectivamente, por ecuaciones:

$$\mathbf{x}(kT+T) = F\mathbf{x}(kT) + Gu(kT) \qquad (3\text{–}8)$$

$$y = H\mathbf{x}(kT)$$

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu(t) \qquad (3\text{–}9)$$

$$y = C\mathbf{x}(t)$$

es necesario determinar la relación existente entre las matrices que definen al sistema en tiempo continuo y tiempo discreto, para que una vez realizado el análisis de un sistema discreto, sea posible determinar su equivalente continuo [14].

Para determinar dicha relación, se usa la solución a la ecuación (3–9), la misma que vendrá dada por [9]:

$$x(t) = e^{At}x(0) = e^{At}\int_o^t e^{-A}Bu(\tau)d\tau \qquad (3\text{-}10)$$

Se supone que todos los componentes de **u(t)** serán constantes entre dos instantes de muestreo consecutivos, o **u(t) = u(KT)**, para el k–ésimo período de muestreo.

A partir de la ecuación (3–10) se pueden tener las siguientes ecuaciones [15]:

$$x(KT+T) = e^{A(KT+T)}x(0) + e^{A(KT+T)}\int_o^{KT+T} e^{-A\tau}Bu(\tau) \qquad (3\text{-}11)$$

$$x(KT) = e^{AKT}x(0) + e^{AKT}x(0) + e^{AKT}\int_o^{KT} e^{-A\tau}Bu(\tau)d\tau \qquad (3\text{-}12)$$

multiplicando la ecuación (3–12) por $e^{\mathbf{A}t}$ y restando de la ecuación (3–11) se obtiene:

$$x(KT+T) = e^{AT}x(KT) + e^{A(KT+T)}\int_{KT}^{KT+T} e^{-A\tau}Bu(\tau)d\tau \qquad (3\text{-}13)$$

$$x(KT+T) = e^{AT}x(KT) + e^{AT}\int_o^T e^{-At}Bu(KT)dt \qquad (3\text{-}14)$$

$$x(KT+T) = e^{AT}x(KT) + e\int_o^T e^{-A\alpha}Bu(\alpha)d\alpha \qquad (3\text{-}15)$$

donde $\alpha = T - t$.

Se define:

$$F = e^{At} \qquad (3\text{-}16)$$

$$G = \int_o^T e^{At}\, dt\, B \qquad (3\text{-}17)$$

A partir de las ecuaciones (3–16) y (3–17), la ecuación (3–15) toma la forma de la ecuación (3–8), y desde estas mismas ecuaciones, la relación matricial buscada, ya está dada.

De la ecuación (3–16) se obtiene entonces:

$$A = \frac{1}{T}\ln(F) \qquad (3\text{–}18)$$

Resolviendo la ecuación (3–17) se tiene:

$$G = \left[e^{At} - I\right]A^{-1}B \qquad (3\text{-}19)$$

de donde:

$$B = A\left[F - I_n\right]^{-1}G \qquad (3\text{-}20)$$

El problema de encontrar la matriz A a partir de la matriz F se basa generalmente en el método de truncación directa. Esto es, la función logaritmo de la matriz F, ln(F), con ciertas condiciones de convergencia, es expandida en cierto tipo de series de potencias infinitas. El método de truncamiemto directo es un método simple [14], sin embargo, el error de truncamiento depende del tipo de expansión en series usado y del número de términos tomados. La expansión en series dada por éste método es la siguiente:

$$A = \frac{1}{T}\ln(F) = \frac{2}{T}\left[R + \frac{1}{3}R^3 + \frac{1}{5}R^5 + \ldots + \frac{1}{n}R^n + \ldots\right]$$
$$\approx \frac{2}{T}R \approx \frac{2}{T}\left[R + \frac{1}{3}R^3\right] \approx \frac{2}{T}\left[R + \frac{1}{3}R^3 + \frac{1}{5}R^5\right] \approx \ldots \qquad (3\text{-}21)$$

donde

$$R = \left[F - I_n\right]\left[F + I_n\right]^{-1} \qquad (3\text{-}22)$$

La condición para la convergencia de la serie matricial dada en (3–21) es que:

$$\text{Re}\,\{\sigma(F)\} > 0,$$

donde $\sigma(F)$ es el espectro de los valores propios de F.

De (3–22) se observa que si Re $\{\sigma(F)\} > 0$, o que todos los valores propios de F se encuentran en la mitad derecha del plano complejo, entonces $|\sigma(R)| < 1$ y $|\sigma(R^2)| << 1$. Como resultado, los primeros términos de la serie de potencias infinita en (3-21) son los términos dominantes. La matriz A deseada puede ser obtenida tomando los primeros términos dominantes de la serie en (3–21).

Generalmente, los valores propios de la matriz F no están disponibles porque no han sido calculados y también, no todos los valores propios de F se encuentran siempre en la mitad derecha del plano complejo. Por lo tanto, el uso del método descrito anteriormemte a veces, no sería eficiente.
Sin embargo, existe un método computacional que usa la raiz q-ésima principal de una matriz no singular F (o $\sqrt[q]{F}$ para $q \geq 2$), junto con el método mostrado en (3–21), para el cálculo necesario de A. Dicho método, coloca todos los valores propios de la matriz $\sqrt[q]{F}$ en la mitad derecha del plano complejo para una rápida determinación de

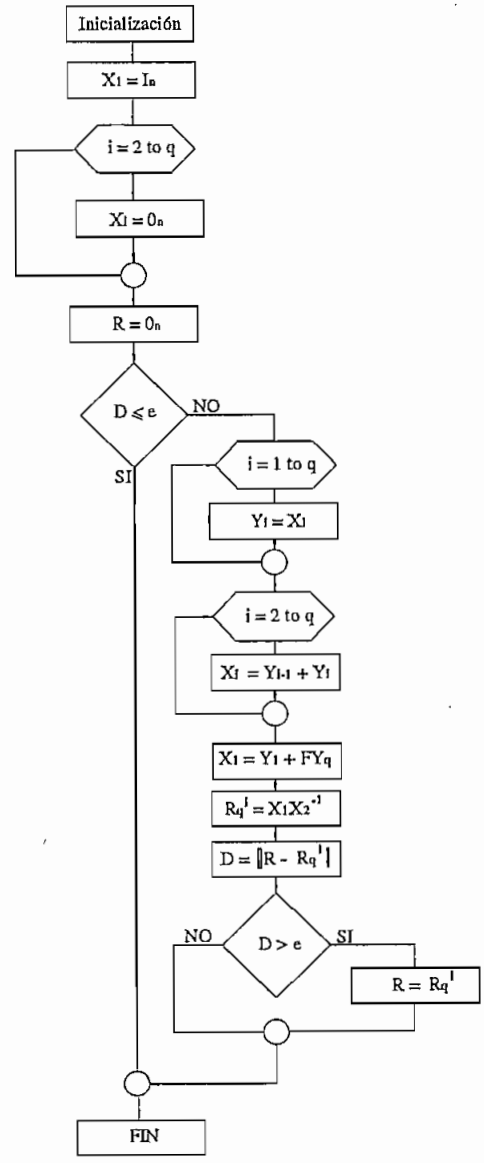
la matriz A a partir de la matriz F.

La q–ésima raiz principal de una matriz real o compleja puede ser definida como se indica a continuación [14]:

"Sea una matriz compleja F, donde F$\in \mathbf{C}^{nxn}$ con su espectro de valores propios $\sigma(F) = \{\lambda_i, i=1,2,..., n\}$ con $\lambda_i \neq 0$ y $\arg(\lambda_i) \neq \pi$. La q–ésima raiz principal de F se define como $\sqrt[q]{F} \in \mathbf{C_{nxn}}$ donde q es un entero positivo. Entonces, $\left(\sqrt[q]{F}\right)^q = F$ y $\arg(\sqrt[q]{\alpha_i}) \in (-\pi/q, \pi/q)$, para i= 1,2, ..., n; donde $\sqrt[q]{\alpha_i} \in \sigma(\sqrt[q]{F})$ es la q–ésima raiz principal de λ_i."

El algoritmo computacional para hallar $\sqrt[q]{F}$ está listado a continuación:

Dados : F – matriz compleja nxn con $|\sigma(F)| \neq 0$ y $\arg(\sigma(F)) \neq \pi$ y
e – error de tolerancia



R = q-ésima raiz principal de F

Con el presente algoritmo [14] y las ecuaciones mostradas en (3–18) y (3–20) es posible determinar el sistema continuo a partir de un cierto sistema discreto. A continuación se presenta un ejemplo en el que se obtiene un sistema por los métodos descritos anteriormente, el de reducción propiamente tal y el de obtencón del modelo continuo:

Sea la función de transferencia de un sistema de tercer orden dada por:

$$G(s) = \frac{(s+4)(s+6)}{(s+1)(s+2)(s+10)} = \frac{s^2+10s+24}{s^3+13s^2+32s+20}$$

Se escoje, como sistema reducido, un sistema de segundo orden, y un período de muestreo de T = 0.05

La función de transferencia en tiempo discreto (en el plano z) vendrá dada por :

$$G^*(z) = \frac{-0.07109z^2 - 0.077648z}{z^2 - 1.83985z + 0.86071}$$

La función de transferencia reducida será:

$$G^*(s) = \frac{0.007345s + 3.08998}{s^2 + 3.1011s + 9.26994}$$

El gráfico 3-1 muestra las respuestas de los sistemas original y reducido. Como puede observarse, el sistema obtenido no conserva los transitorios iniciales de la respuesta. Cabe señalar además que en el método de obtención del sistema continuo, a partir del modelo discreto, es necesario realizar un ajuste de la ganancia, para que ésta se con-

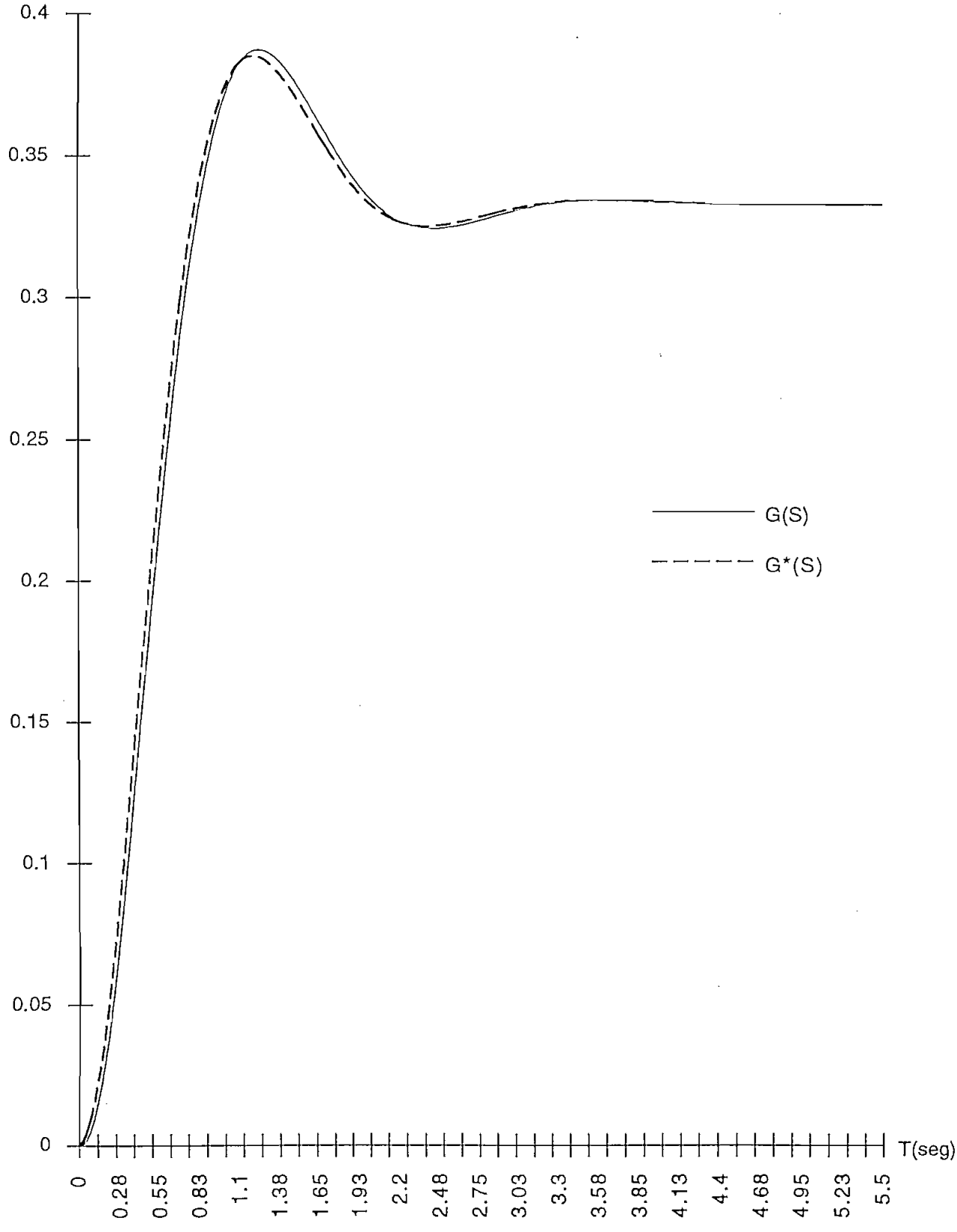

0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
G(S)
G*(S)
T(seg)
0
0.28
0.55
0.83
1.1
1.38
1.65
1.93
2.2
2.48
2.75
3.03
3.3
3.58
3.85
4.13
4.4
4.68
4.95
5.23
5.5

Gráfico 3-1

serve en estado estable en el modelo original y en el equivalente reducido.

3.2 Método de los Mínimos Cuadrados.–

Uno de los métodos más usados en la identificación de sistemas es éste, llamado de los mínimos cuadrados, y que se basa en determinar la función de transferencia de determinado orden de un cierto sistema, en base al muestreo de los datos de su señal de salida.

El método es aplicable en la reducción de sistemas ya que es posible obtener la respuesta en tiempo discreto de un sistema original dado, y en base a esos datos, tratar de obtener un nuevo sistema, de orden menor al original, mediante un proceso de identificación.

Existen trabajos específicos muy completos que versan sobre la identificación de sistemas usando el proceso de los mínimos cuadrados (por ejemplo, Tesis de Carlos Cordero, MINIMOS CUADRADOS GENERALIZADO, E.P.N. 1986). En la presente tesis, se desarrolla un método para la determinación del modelo discreto de un sistema reducido, utilizando los datos de la respuesta de un sistema de alto orden, tomados a un cierto intervalo de tiempo. El método se basa en la aplicación de la matriz pseudoinversa, la misma que deriva en el proceso de mínimos cuadrados. Un algoritmo iterativo se desarrollará para el proceso de identificación del sistema discreto y el sistema continuo se lo obtendrá usando el proceso de la q–ésima raiz, explicado en subtemas anteriores.

3.2.1 Matriz Pseudoinversa .–

Se puede considerar un modelo discreto, de una entrada y una salida, dado por la siguiente función de transferencia [16], [17]:

$$H(z)=\frac{X(z)}{U(z)}=\frac{d_o+d_1z^{-1}+d_2z^{-2}+\ldots+d_mz^{-m}}{1+c_1z^{-1}+c_2z^{-2}+\ldots+c_nz^{-n}} \qquad (3\text{-}23)$$

Dicho modelo puede expresarse en las siguientes ecuaciones:

$$X(z)[1+c_1z^{-1}+c_2z^{-2}+\ldots+c_nz^{-n}] = U(z)[\,d_0+d_1z^{-1}+d_2z^{-2}+\ldots+d_mz^{-m}\,] \qquad (3\text{–}24)$$

Expresando en ecuaciones de diferencias:

$$x(k) + c_1x(k-1) + c_2x(k-2) + \ldots + c_nx(k-n) = d_0u(k) + d_1u(k-1) + d_2u(k-2) + \ldots d_mx(k-m) \quad (3\text{–}25)$$

$$x(k) = d_0u(k) + d_1u(k-1) + \ldots + d_mx(k-m) - c_1x(k-1) - \ldots - c_nx(k-n)$$

$$x(k) = \sum_{i=0}^{m} d_i u(k) - \sum_{i=1}^{n} c_i x(k-i) \quad (3\text{-}26)$$

Siendo T el período de muestreo se tiene que x(k) = x(kT) y u(k) = u(kT). En éstas condiciones, el problema se reduce determinar los parámetros d_i y c_i.

Se define el vector θ como un arreglo formado por dichos parámetros:

$$\theta = [\, d_0\ d_1\ d_2 \ldots d_m\ c_1\ c_2 \ldots c_n\,]^T \quad (3\text{–}27)$$

siendo la dimensión de $\theta p = n+m+1$.

Si se arregla la ecuación (3–26) en forma de una multiplicación de matrices, se tiene:

$$x(k) = A_k\theta \quad (3\text{–}28)$$

donde $k \geq p$ y las matrices A_k y x(k) definidas como:

$$A_k = \begin{bmatrix} u_o & u_{-1} & \ldots & u_{-m} & -x_{-1} & -x_{-2} & \ldots & -x_{-n} \\ u_1 & u_o & \ldots & u_{1-m} & -x_o & -x_{-1} & \ldots & -x_{1-n} \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ u_K & u_{K-1} & \ldots & -u_{K-m} & -x_{K-1} & -x_{K-2} & \ldots & -x_{k-n} \end{bmatrix} \quad (3\text{-}29)$$

$$x(k) = [\, x_1\ x_2\ \ldots\ x_k\,] \quad (3\text{–}30)$$

La determinación del vector θ es directa, a través de los datos obtenidos en los instantes de muestreo y de la ecuación:

$$\theta = A_k^{+} x(k) \tag{3–31}$$

donde la matriz A_k^{+} es la matriz pseudo-inversa de A (ya que A, generalmente no es una matriz cuadrada) y que se la calcula mediante la expresión:

$$A_k^{+} = [A_k^{t} A_k]^{-1} A_k^{t} \tag{3–32}$$

Para que la matriz pseudo–inversa definida por la ecuación (2–32) exista y sea única, el rango de la matriz A_k debe ser igual al número de sus columnas para k >p. Esto se asegura cuando la señal de entrada u cumple con la condición:

$$u_i = 0 \text{ para } i < 0$$
$$u_i = 1 \text{ para } i \geq 0 \tag{3–33}$$

en otras palabras, la señal de entrada debe ser un escalón unitario.

Cumplida la condición anterior, se puede aplicar la ecuación (3–31) para la resolución del sistema. Sin embargo, un problema en hacerlo radica en la cantidad de datos que deben ser almacenados con el fin de minimizar la media–cuadrada del error sobre un intervalo suficientemente largo, especialmente cuando el período de muestreo es razonablemente alto. Esta dificultad se soluciona al aplicar el siguiente algoritmo recursivo propuesto originalmente por Albert y Sittler (1966) y modificado posteriormente por Wells (1967) [17].

Sea:

$$A_{K+1} = \begin{bmatrix} A_K \\ a_{K+1}^{T} \end{bmatrix} \tag{3-34}$$

donde

$$a_{k+1}^{T} = [u_{k+1} \; u_k \; \cdots \; u_{k-m+1} \; -x_k \; -x_{k-1} \; \cdots \; -x_{k-n+1}] \tag{3–35}$$

y sea:

$$x_{K+1} = \begin{bmatrix} x_K \\ x_{K+1} \end{bmatrix} \tag{3-36}$$

A partir de estas ecuaciones, el algoritmo recursivo está dado por (para k≥p):

$$\theta_{K+1} = \theta_K + \frac{P_K a_{K+1}\left(x_{K+1} - a_{K+1}{}^T \theta_K\right)}{1 + a_{K+1}{}^T P_K a_{K+1}} \tag{3-37}$$

donde

$$P_{K+1} = P_K - \frac{P_K a_{K+1}\left[P_K a_{K+1}\right]^T}{1 + a_{K+1}{}^T P_K a_{K+1}} \tag{3-38}$$

Debe notarse que el algoritmo no requiere el cálculo de matrices inversas. Sin embargo, para comenzarlo, una inversión de matrices es requerida, para el cálculo de θ_k y P_k para el caso cuando k=p, esto es, A_k es una matriz cuadrada:

$$\theta_p = A_p{}^{-1} x_p \tag{3–39}$$

$$P_p = [A_p{}^T A_p]^{-1} \tag{3–40}$$

Es posible evitar completamente la inversión de matrices si se comienza desde k=0, y se usa las siguinetes ecuaciones para k≤p:

$$\theta_{K+1} = \theta_K + \frac{Q_K a_{K+1}\left(x_{K+1} - a_{K+1}{}^T \theta_K\right)}{a_{K+1}{}^T Q_K a_{K+1}} \tag{3-41}$$

donde

$$Q_{K+1} = Q_K - \frac{Q_K a_{K+1}\left[Q_K a_{K+1}\right]^T}{a_{K+1}{}^T Q_K a_{K+1}} \tag{3-42}$$

y

$$P_{K+1} = P_K - \frac{\left[P_K a_{K+1}\right]\left[P_K a_{K+1}\right]^T + \left[Q_K a_{K+1}\right]\left[P_K a_{K+1}\right]^T}{a_{K+1}{}^T Q_K a_{K+1}} + \frac{Q_K a_{K+1}\left[Q_K a_{K+1}\right]^T\left[1 + a_{K+1}{}^T P_K a_{K+1}\right]}{\left(a_{K+1}{}^T Q_K a_{K+1}\right)^2} \tag{3-43}$$

con la condición inicial $\theta_0 = 0$, $Q_0 = I$ y $P_0 = 0$.

3.2.2 Determinación del modelo continuo equivalente.–

El método a usarse es el de la q–ésima raíz principal de la matriz F asociada al modelo discreto, cuando éste está descrito en el espacio de estado, por medio de la ecuación (3–8), y que ya fue analizado en la sección 3.1.2. Cabe resaltar que hay que realizar una corrección en la ganancia para que la salida en estado estable sea la misma, tanto en el modelo original como en el equivalente reducido.

A continuación se presenta un ejemplo que muestra las funciones de transferencia del modelo original y del equivalente reducido:

Función de transferencia original:

$$G(s) = \frac{135}{s^3 + 48s^2 + 144s + 405}$$

Usando un período de muestreo T = 0.05, la función de transferencia reducida está dada por:

$$G^*(s) = \frac{-0.1398417 + 3.0899798}{s^2 + 3.1011s + 9.26994}$$

La figura 3-2 muestra una comparación de las respuestas en el tiempo de las funciones de transferencia de los modelos original y equivalente reducido, usando como entrada una señal escalón unitario.

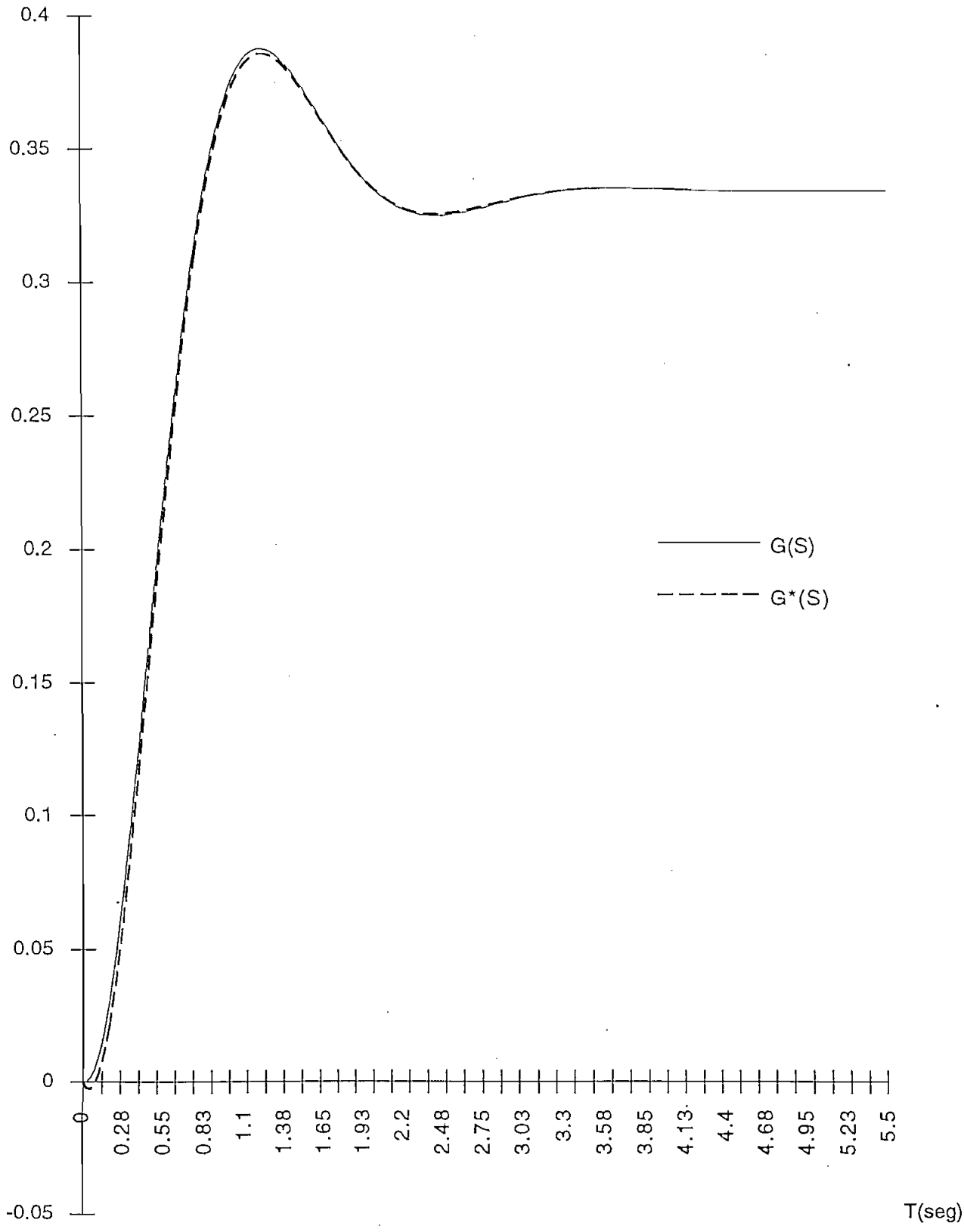

0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
0
0.28
0.55
0.83
1.1
1.38
1.65
1.93
2.2
2.48
2.75
3.03
3.3
3.58
3.85
4.13
4.4
4.68
4.95
5.23
5.5
G(S)
G*(S)
T(seg)

Gráfico 3-2

CAPITULO IV

METODO DE DESCOMPOSICION EN VALORES SINGULARES

Una nueva alternativa para la obtención de modelos reducidos, en sistemas discretos, a partir de sistemas de alto orden, es presentada en el presente capítulo.

El método se basa en el uso de los "valores singulares de una matriz" los mismos que se definen a continuación y en el uso de la matriz de Hankel, la misma que fue ya definida y utilizada en el método de la Realización Parcial, desarrollado en el capítulo II. Este método usa también un proceso de identificación para obtener el sistema reducido a partir de la matriz de Hankel, definida, en este caso, para un sistema discreto.

4.1 Valores Singulares.–

Se definen los valores singulares de una matriz hermitiana cualquiera A como lo siguiente [3]:

"Para una matriz hermitiana cualquiera A, de orden nxm, donde A^H significa la transpuesta hermitiana de A, las raíces cuadradas estrictamente positivas σ_i de los valores propios distintos de cero de A^HA (o su equivalente, AA^H), son llamadas valores singulares de A. Si n=m, los valores singulares serán iguales a los valores propios absolutos de A".

Con el ejemplo siguiente se ilustra explícitamente la definición anterior.

Sea una matriz hermitiana cualquiera A, de orden 3x3 dada por:

$$A = \begin{bmatrix} 3 & 0 & 2 \\ 0 & 2 & 4 \\ 2 & 4 & -8 \end{bmatrix} \qquad (4\text{-}1)$$

los valores propios de A estarán dados por:

$$\begin{aligned} \lambda_1 &= 2.6847 \\ \lambda_2 &= 4.0000 \\ \lambda_3 &= -9.6847 \end{aligned} \qquad (4\text{-}2)$$

La transpuesta hermitiana de A estará dada por:

$$A^H = \begin{bmatrix} 3 & 0 & 2 \\ 0 & 2 & 4 \\ 2 & 4 & -8 \end{bmatrix} \tag{4-3}$$

La matriz A^HA y sus respectivos valores propios estarán dados por:

$$A^HA = \begin{bmatrix} 13 & 8 & -10 \\ 8 & 20 & -24 \\ -10 & -24 & 84 \end{bmatrix} \tag{4-4}$$

$$\begin{aligned} \lambda_1 &= 16.0000 \\ \lambda_2 &= 96.7926 \\ \lambda_3 &= 7.2074 \end{aligned} \tag{4-5}$$

Los valores singulares (raíces cuadradas estrictamente positivas de los valores propios) de la matriz dada en (4-4) estarán dados por:

$$\begin{aligned} \sigma_1 &= 4.0000 \\ \sigma_2 &= 9.6847 \\ \sigma_3 &= 2.6845 \end{aligned} \tag{4-6}$$

Si se compara las ecuaciones (4-2) y (4-6) se comprueba que los valores singulares de A^HA son exactamente iguales al valor absoluto de los valores propios de A, tal como se indica en la definición expuesta al inicio del capítulo.

4.2 Parámetros de Markov en tiempo discreto y Matriz de Hankel .–

Un modelo en el espacio de estado, en tiempo discreto, con p entradas y q salidas está dado por las ecuaciones:

$$\begin{aligned} \mathbf{x}(k+1) &= F\mathbf{x}(k) + G\mathbf{u}(k) \\ \mathbf{y}(k) &= H\mathbf{x}(k) \end{aligned} \tag{4–7}$$

donde F es una matriz constante nxn, G es una matriz constante pxn, H es una matriz constante nxq, **x** es un vector de orden n, **u** es un vector de orden p e **y** es un vector de orden q. Los parámetros de Markov en tiempo discreto están definidos por las cantidades:

$$J(k) = HF^{k-1}G \quad (k = 1,2,...) \tag{4–8}$$

Como ya se vio en el capítulo 2, los parámetros de Markov constituyen los elementos de las matrices de Hankel, las cuales están definidas de la siguiente manera:

$$S_{uv}(i) = \begin{bmatrix} J_i & J_{i+1} & \cdots & J_v \\ J_{i+1} & J_{i+2} & \cdots & J_{v+1} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ J_u & J_{u+1} & \cdots & J_{u+v-i} \end{bmatrix} \tag{4-9}$$

Los parámetros de Markov en tiempo discreto están definidos por la respuesta impulso del sistema. Si el sistema dado es un modelo continuo representado por las matrices [A, B, C], entonces los parámetros de Markov en tiempo discreto pueden ser obtenidos usando un muestreo de la respuesta impulso:

$$J(k) = Ce^{AkT}B \quad (k = 1,2,...) \tag{4–10}$$

donde T es el período de muestreo.

4.3 Algoritmo de reducción del modelo.–

Como se anotó en el capítulo 2, en el método de Realización Parcial, el rango de la matriz de Hankel que se constituya, reflejará el orden del sistema correspondiente. Esta relación rango–orden puede ser usada, y de hecho lo es, en los problemas de realización.

Con el fin de abordar el problema de encontrar un sistema aproximado de bajo orden a partir de uno de orden superior, se necesita cierta información adicional que muestre la conexión de la matriz de Hankel, obtenida a partir del sistema original, con una matriz de orden inferior. Matemáticamente, el número de valores singulares distintos de cero corresponde al rango de la matriz. Por lo tanto, aplicando la técnica de descomposición en valores singulares, para encontrar una matriz de Hankel de rango menor a aquella obtenida desde el sistema de alto orden, de modo que al aplicar un proceso de

realización se pueda obtener las matrices asociadas al sistema reducido, es una aproximación razonable para dichos modelos.

Una acotación importante a este hecho es que debido a la manera en que está definida la matriz de Hankel a partir de los datos muestreados desde la respuesta impulso, dicha matriz siempre será una matriz hermitiana de orden vxv. Por lo tanto, sus valores singulares serán iguales a las raíces cuadradas positivas de sus valores propios distintos de cero.

Considérese un sistema de alto orden. La secuencia de la respuesta a una entrada impulso, estará dada por los parámetros J_i ($i = 1,2,...,2_{n+1}$) que pueden ser arreglados en una matriz de orden nxn.

Si el rango de la matriz de Hankel obtenida es r, siendo r< n, significa que los valores singulares $\sigma_{r+1}, ..., \sigma_{n+1}$ serán cero. Cuando éstos no son exactamente cero pero su valor es cercano a cero, se puede considerar que la matriz es aproximadamente de rango r.

Una vez definida la matriz de Hankel de rango r, mediante el proceso de realización indicado en el capítulo II en el método de Realización Parcial, es posible obtener las matrices F*, G* y H* asociadas al sistema reducido en el plano z. Usando el método desarrollado en la numeral 3.1.2 del capitulo III se obtendrán las matrices A*, B* y C* asociadas al modelo reducido en tiempo continuo.

A continuación se indica un ejemplo de los resultados obtenidos al utilizar el presente método.

Sea un sistema de tercer orden dado por la siguiente función de transferencia:

$$G(s) = \frac{135}{s^3 + 48s^2 + 144s + 405}$$

La matriz de Hankel obtenida a partir de los datos muestreados de la respuesta a una entrada impulso, con un período T = 0.05, estará dada por:

$$S = \begin{bmatrix} -0.0711 & -0.2084 & -0.3223 & -0.4136 \\ -0.2084 & -0.3223 & -0.4136 & -0.4836 \\ -0.3223 & -0.4136 & -0.4836 & -0.5337 \\ -0.4136 & -0.4836 & -0.5337 & -0.5657 \end{bmatrix}$$

Los valores singulares, que como ya se ha dicho, corresponden a las raíces cuadradas positivas de los valores propios estarán dados por:

$$\sigma_1 = 1.2753$$
$$\sigma_2 = 0.42638$$
$$\sigma_3 = 0$$
$$\sigma_4 = 0$$

Debido a que la matriz S es de rango 2 (solo 2 valores singulares son distintos de cero), el sistema reducido que se podrá obtener será de segundo orden.

Mediante el proceso de realización explicado en el capítulo 2 (Método de Realización Parcial), las matrices F, G y H asociadas al sistema reducido en el plano z, estarán dadas por:

$$F = \begin{bmatrix} 0 & -0.8607 \\ 1 & 1.8339 \end{bmatrix} \qquad G = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad H = \begin{bmatrix} -0.2084 & -0.3223 \end{bmatrix}$$

La función de transferencia del sistema reducido, en el plano z será:

$$G^*(z) = \frac{-0.2084z + 0.05988}{z^2 - 1.8339z + 0.8607}$$

Usando el proceso indicado en el numeral 3.1.2 del capítulo III, el sistema reducido en el plano s tendrá una función de transferencia dada por:

$$G^*(s) = \frac{0.2032s + 3.09}{s^2 + 3.1011s + 9.2699}$$

El gráfico 4-1 compara las respuestas en el tiempo, a una entrada escalón unitario, del sistema original y del reducido:

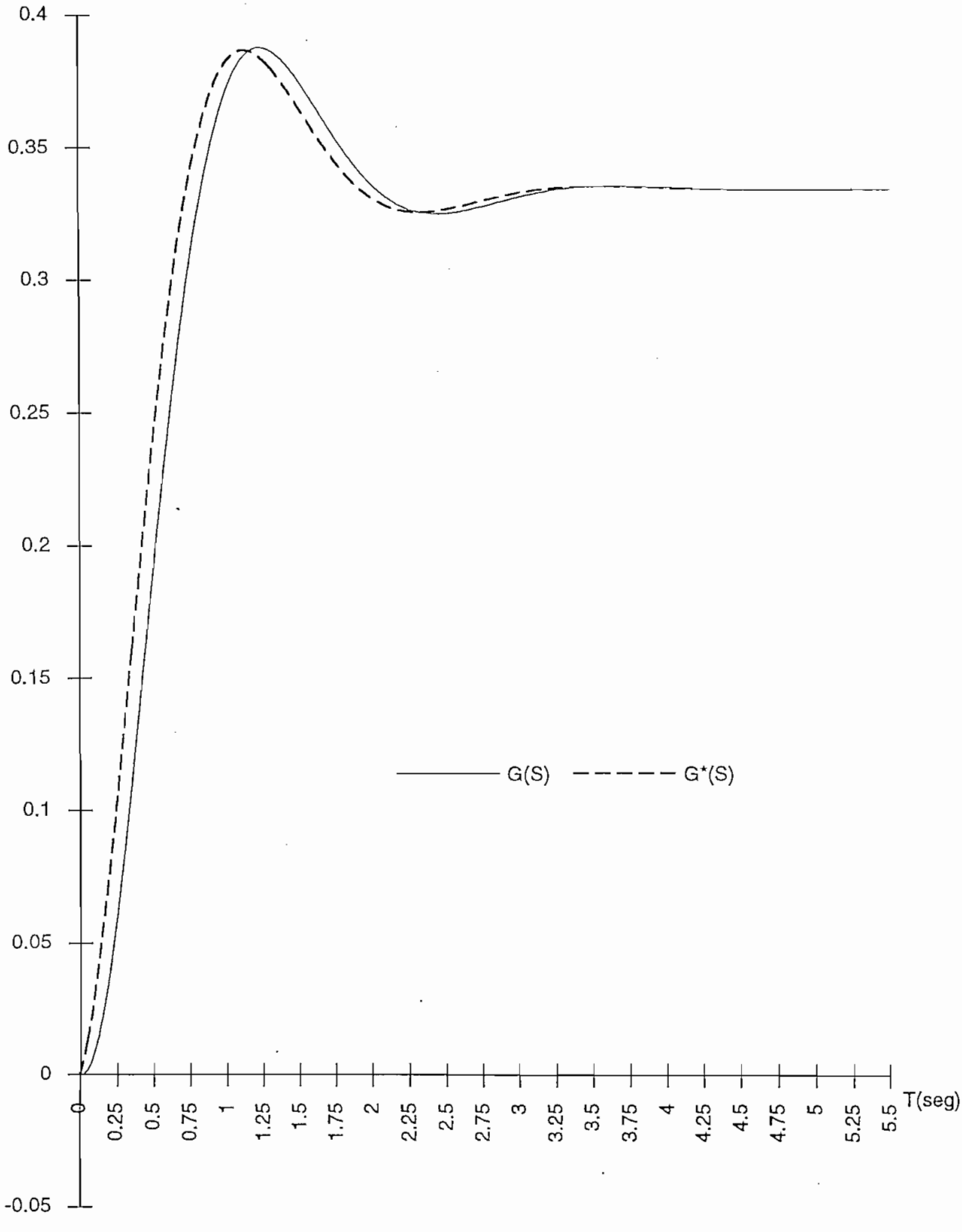

0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
G(S)
G*(S)
0
0.25
0.5
0.75
1
1.25
1.5
1.75
2
2.25
2.5
2.75
3
3.25
3.5
3.75
4
4.25
4.5
4.75
5
5.25
5.5
T(seg)

Gráfico 4-1

CONCLUSIONES

Luego de analizar los métodos de reducción expuestos en el presente trabajo , se puede exponer una conclusión global para todos ellos y conclusiones particulares de acuerdo con los resultados obtenidos de pruebas realizadas para cada caso.

Cabe indicar que todas las pruebas de respuesta en el tiempo, a una determinada señal de entrada escalón unitario, de las funciones de transferencia obtenidas a partir de cada uno de los métodos analizados, fueron realizadas en el programa CC.

Analizando particularmente los métodos expuestos se han llegado a determinar las conclusiones que a continuación se detallan:

Dentro de los métodos de reducción por expansión en fracciones continuas, la Segunda Forma de Cauer puede producir solamente una buena aproximación de la ganancia en estado estable del sistema original, con un cálculo matemático y de computador sencillo y rápido. Parámetros tales como el valor de máximo sobreimpulso, y tiempo de establecimiento de la respuesta del sistema original a una entrada escalón unitario, no se conservan en el sistema reducido.

La Tercera Forma de Cauer y la Forma Modificada de Cauer producen siempre la misma aproximación al sistema reducido buscado. Matemáticamente, la función de transferencia de orden reducido obtenida cumple con los teoremas del valor inicial y final [4] siendo ésta la finalidad de dichos métodos. Sin embargo, la ubicación de uno, o varios polos del sistema reducido obtenido, puede producirse en la mitad derecha del plano s, haciendo al sistema reducido inestable. Será necesario, por lo tanto, probar los resultados obtenidos por dichos métodos y de obtener un sistema reducido que no sea similar al sistema original, habrá que usar otro método de reducción.

Cuando el sistema reducido obtenido no presenta éste problema, la aproximación es bastante buena a la del sistema original, tal como se puede apreciar en los gráficos 1-2 y 1-3.

La Forma General de Cauer y la Nueva Forma General de Expansión en Fracciones Continuas pueden producir una buena aproximación del modelo original, dependiendo de los Parámetros de Markov o Momentos de Tiempo escogidos para cada caso. Como se explicó en el desarrollo de los temas mencionados, dependiendo de los parámetros fijados, se pueden obtener las mismas aproximaciones que producen la Primera Forma de Cauer, la Segunda o la Forma Modificada de Cauer. Esto quiere decir, que va a depender de los parámetros que se fijen para que el sistema reducido conserve o no, la ganancia en estado estable o los transitorios iniciales de la respuesta en el tiempo para una cierta señal de entrada. Por lo tanto, esto significa que habrá casos en que dichos métodos no produzcan una buena aproximación al sistema original.

Dentro de los métodos de modo dominante, el método de Davison-Chidambara y el

método de Realización Parcial producen una buena aproximación del modelo original.

Como ya se explicó en el capítulo correspondiente, el método de Davison-Chidambara usa los polos despreciados, que se encuentran lejos del eje jw, para ajustar la ganancia en estado estable del sistema reducido, de modo que sea exactamente la misma que aquella del sistema original.

En la aproximación obtenida usando el método de Agregación puede existir una ligera variación de la ganancia, en estado estable, entre los sistemas original y equivalente aproximado. Básicamente, esto se debe a que la llamada " matriz de agregación", usada por dicho método, rescata solo algunas características del sistema original, como se explicó en el desarrollo del tema, para producir un sistema de orden reducido. De los resultados obtenidos, no se recomienda usar este métodos de reducción.

En lo que se refiere al método de Realización Parcial, ya se indicó que produce una buena aproximación del sistema original, como lo demuestra el gráfico 2-3. Sin embargo, un inconveniente de usar este método es el de poder obtener solamente un cierto orden de sistema reducido, de acuerdo a la configuración del sistema original. Como ya se ha anotado en el desarrollo del tema correspondiente, el sistema reducido no podrá ser de un orden menor a dos veces la diferencia de grado existente entre los polinomios del denominador y numerador de la función de transferencia del sistema original. La razón se debe al proceso matemático usado por dicho método. Al tratar de obtener un sistema reducido menor al orden indicado, la matriz de Hankel generada para usarse en el proceso matemático de reducción del orden del sistema, se llena con columnas de ceros, haciendo que dicho proceso matemático falle al tratar de obtener las matrices asociadas al sistema reducido.

En los métodos de aproximaciones óptimas, el método de Aproximaciones Optimas como tal, no produce una buena aproximación al sistema original. Como se observó en el desarrollo de dicho método, éste se basa en un proceso matemático de resolución de un sistema de ecuaciones, para encontrar los coeficientes de los polinomios de la función de transferencia reducida. Los términos independientes de cada una de las ecuaciones generadas son las muestras de la respuesta en el tiempo, del sistema original, tomadas a un cierto período de muestreo, cuando se aplica una señal de entrada impulso unitario. La aproximación obtenida dependerá en buena parte, del período de muestreo escogido. Además, el número de datos muestreados solamente será igual a dos veces el orden del sistema reducido que se desee obtener. Esto es una gran limitación, ya que posiblemente, solo se alcance a muestrear una zona muy pequeña de la respuesta en el tiempo del sistema original, perdiéndose, de esa manera, muchas de las características de la misma.

El método de los Mínimos Cuadrados produce un sistema reducido con características muy similares de respuesta al sistema original bajo las mismas condiciones de entrada. Sin embargo, su algoritmo computacional es largo y lento.
Cabe anotar que, la estabilidad del sistema reducido obtenido así como su similitud con el sistema original, dependerá, también en éste caso, del período de muestreo es-

cogido. Es conveniente realizar varias aproximaciones, para la obtención del modelo reducido, con distintos períodos de muestreo, y analizar cuál de ellas es la que más se aproxima al sistema original.

El método de los valores singulares tampoco produce una buena aproximación del sistema original. A pesar de basar mucho su principio de funcionamiento en la retención de las características dominantes del sistema, como lo hacen los métodos de modo dominante, en especial, el de Realización Parcial, la aproximación obtenida dependerá nuevamente del período de muestreo utilizado y de la obtención del sistema reducido en el plano s usando el método de la q-ésima raíz principal.

Al igual que en los métodos de aproximaciones óptimas, es necesario obtener varios modelos reducidos utilizando distintos períodos de muestreo para cada caso, y escoger el que genere la mejor aproximación del sistema original.

Es necesario anotar también, como conclusiones particulares, todas las limitaciones que se presentaron o que se encontraron durante el desarrollo de la presente tesis:

- La bibliografía necesaria para realizar un análisis profundo y minucioso del tema escogido para la tesis es nula en las bibliotecas ecuatorianas. Fue necesario recurrir, principalmente, al Instituto de Investigaciones Tecnológicas de la Escuela Politécnica Nacional para poder desarrollar el tema.

- Ha sido imposible recopilar todos los artículos necesarios para del desarrollo de ciertos tema, tales como los métodos de expansión en fracciones continuas y el método de los Valores Singulares.

- La gran mayoría de los artículos recopilados no proporcionan una adecuada justificación matemática para el método que desarrollan. Esto ha significado el tener que realizar un esforzado análisis de cada uno de los métodos expuestos para poder desarrollar un justificativo matemático correcto.

- En algunos artículos analizados como fuente bibliográfica, se han encontrado fallas en el desarrollo y justificativo matemático de determinados temas, lo que ha dificultado aún más de desarrollo de la tesis.

- Los puntos expuestos anteriormente han significado que el tiempo de desarrollo de la tesis sea bastante extenso, razón por la cual se han omitido algunos análisis del comportamiento de los modelos reducidos obtenidos, de acuerdo a cada uno de los métodos expuestos.

- Las razones anteriores también han determinado que algunos de los algoritmos desarrollados para la resolución de los distintos métodos de reducción, no sean aplicables a casos generales de sistemas físicos. Un caso particular de los mismos se lo encuentra en el Método de Davison-Chidambara, en el que el programa digital no acepta funciones de transferencia que tengan polos complejos conjugados, cuando

el método de reducción es aplicable perfectamente a dicho caso.

Resumiendo lo expuesto anteriormente y como conclusiones generales sobre los métodos de reducción se puede decir que:

- Son una herramienta muy útil en el análisis y diseño de sistemas, puesto que simplifican el cálculo matemático y pueden producir una disminución en los costos de diseño e implementación de los mismos.

- En algunos casos, tales como los métodos de reducción usando la Segunda Forma de Cauer, la Tercera Forma de Cauer, la Forma Modificada de Cauer, el Método de Agregación y el Método de los Valores Singulares, la función de transferencia aproximada obtenida, no guarda las mismas características de respuesta que aquella del sistema original, cuando se aplican a los dos sistema, original y equivalente reducido, la misma señal de entrada.

- En ciertos casos también, como en la Tercera Forma de Cauer, la Forma Modificada de Cauer, el método de Aproximaciones Optimas, el método de los Valores Singulares, el modelo reducido obtenido puede ser inestable, así el modelo original sea estable, independientemente del método de reducción utilizado.

Las fallas que presentan los métodos en la obtención de un sistema reducido que no guarda un comportamiento similar o aproximado con aquel sistema original, pueden deberse principalmente a las siguientes razones:

- Algunos métodos solamente son aplicables a cierto tipo de sistemas reales que generan una cierta función de transferencia en particular. Un caso específico se tiene en los métodos de reducción por expansión en fracciones continuas, en las que los grados del numerador y denominador de la función de transferencia original, no pueden tener una diferencia mayor a uno. La razón para ello se basa en la configuración física que puede tener una red con dos tipos de elementos, RL, RC o LC [1].

- En los métodos de reducción que utilizan muestras de la respuesta en el tiempo del sistema original cuando se lo aplica una cierta señal de entrada, y dichas muestras son tomadas a un período determinado, tales como el Método de Agregación, el Método de los Mínimos Cuadrados y el Método de los Valores Singulares, cuenta mucho el haber escogido un adecuado período de muestreo para la obtención del sistema reducido. Dependiendo del valor de dicho período de muestreo, se pueden obtener funciones de transferencia de orden reducido, que generen una respuesta bastante similar a aquella generada por el sistema original, así como también sistemas reducidos cuya respuesta ni siquiera se asemeje a la del sistema original. Básicamente esto dependerá de la rapidez de respuesta del sistema original. Si el sistema original es rápido, es decir, el tiempo de establecimiento de su respuesta a una entrada escalón unitario es pequeño, un período de muestreo pequeño será capaz de recoger muestras tanto de las transiciones iniciales de la respuesta así como de su valor en estado estable. El sistema reducido obtenido podrá tener una buena

aproximación al sistema original. Por el contrario, un período de muestreo de alto valor, para ese mismo sistema rápido, solo recogerá las características de la respuesta cuando ésta haya alcanzado ya su estado estable. En éste caso, posiblemente, el sistema reducido obtenido no podrá generar una respuesta similar a la del sistema original, cuando se lo aplique la misma señal de entrada.

- Cabe anotar también que se realiza un proceso matemático de aproximaciones, para la obtención del sistema reducido en el dominio de Laplace, a partir de aquel obtenido, de acuerdo con el método de reducción usado, en el plano z. Dichas aproximaciones, dadas por el método de la q-ésima raíz principal [14], pueden también introducir errores ya que, básicamente, dicho método se basa en el truncamiento de la serie a la que se ha expandido la función ln (F), siendo F una de las matrices asociadas al sistema reducido en el plano z.

BIBLIOGRAFIA

1.- Van Valkenburg M.E., "Análisis de Redes" Ed. Limusa S.A., pp. 335 - 345. México 1977.

2.- Weinberg L , "Network Analysis and Synthesis", Mc.Graw Hill, International Student Edition, pp. 399, 1962.

3.- Lai I. S., "A Comparative Study of Model Reduction Methods", Master Thesis, Univ. of Tennessee, Knoxville, 1985.

4.- Van Valkenburg M.E., "Análisis de Redes", Ed. Limusa S.A., pp. 258-259. México, 1977.

5.- Shieh L.S. and Goldman M.J., "Continued Fraction Expansion and Inversion of the Cauer Third Form", IEEE Trans. Circuits Syst., Vol CAS-21, pp. 341-345. 1974.

6.- Eydgah Ali M. and Singh Harpreet., "A Modified Procedure for the Realization of the Transfer Function Matrix From a Mixture of Markov Parameters and Moments", IEEE Trans. Automatic Control, Vol AC-30, #3, pp. 299-301, 1985.

7.- Chidambra M.R., "On A Method for Simplifying Large Linear Dynamic Systems", IEEE Trans. Automatic Control, Vol AC-12, #1, pp. 119-120, 1967.

8.- Chidambra M.R., "Further Remark on Simplifying Large Linear Dynamic Systems", IEEE Trans. Automatic Control, Vol AC-12, #2, pp. 213-214, 1967.

9.- Ogata Katsuhiko, "Ingeniería de Control Moderna", Ed. Prentice-Hall International, pp. 744, 1974.

10.- Elrazaz Z. and Sinha N.K.,"On the Selection of the Dominant Poles of a System to be Retained in a Lower Order Model", IEEE Trans. Automatic Control, Vol AC-26, #6, pp. 1301, 1981.

11.- Mahapatra G.B., "A Further Note on Selecting a Lower Order System Using the Dominant Eigenvalue Concept", Vol AC-24, #1, pp. 135-136, 1979.

12.- Aoki M., "Control of Large Scale Dynamic System by Aggregation", IEEE Trans. Automatic Control, Vol AC-13, pp. 246-253, 1968.

13.- Rozsa P and Sinha N.K., "Efficient Algorithm for Irreducible Realization of a Rational Matrix", Int. J. Control, Vol 20, #5, pp. 739-751, 1974.

14.- Shieh Leang S, S.H. Jason, Tasai and Sui R Lian, "Determining Continuous Time State Equations from Discrete Time State Equations Via Principal qth. Method", IEEE Trans. Automatic Control, Vol AC-31, #5, pp. 454-457, 1986.

15.- Ogata Katsuhiko, "Ingeniería de Control Moderna", , Ed. Prentice- Hall International, pp. 762-763, 1974.

APENDICE A

Listado del Programa

```
'
'   Pantallas de presentacion
'
    CLEAR
    DEFINT I-N
    DEFDBL A-H, O-Z
    CLS
    LOCATE 2, 13
    COLOR 15, 0
    PRINT "Ú";
    PRINT STRING$(54, "Ä");
    PRINT "¿";
    LOCATE 3, 13
    PRINT "³          "; "ESCUELA POLITECNICA NACIONAL";
    PRINT "          ³"
    LOCATE 4, 13
    PRINT "³";
    PRINT STRING$(54, " "); "³"
    LOCATE 5, 13
    PRINT "³ "; "METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA"; " ³"
    LOCATE 6, 13
    PRINT "³";
    PRINT STRING$(54, " "); "³"
    LOCATE 7, 13
    PRINT "³          ";
    PRINT "Autor: Juan Fernando Donoso A."; "          ³"
    LOCATE 8, 13
    PRINT "À";
    PRINT STRING$(54, "Ä");
    PRINT "Ù"

    DIM A$(30)
    N = 5
    A$(1) = "Expansion en Fracciones Continuas"
    A$(2) = "Metodos de Modo Dominante"
    A$(3) = "Metodos de Aproximaciones Optimas"
    A$(4) = "Descomposicion en Valores Singulares"
    A$(5) = "Fin"

    mmax = 0
    FOR I = 1 TO N
    maux = LEN(A$(I))
    IF mmax < maux THEN
      mmax = maux
    ELSE
    END IF
    NEXT I
    IY = (80 - mmax - 6) / 2
    IX = 10

    LOCATE IX, IY
    cx = mmax + 4
    CY = N + 2
    COLOR 15, 1
    PRINT "Ú";
    PRINT STRING$(cx, "Ä");
    PRINT "¿"
    FOR I = 1 TO N
    LOCATE IX + I, IY
```

```
      PRINT "≥ ";
      PRINT A$(I);
      maux = cx - LEN(A$(I)) - 2
      PRINT STRING$(maux, " "); "≥"
      NEXT I
      LOCATE IX - 1 + CY, IY
      PRINT "¿";
      PRINT STRING$(cx, "ƒ");
      PRINT "Ÿ"
      IY = IY + 3

       LOCATE IX + N + 4, 30
       COLOR 12, 0
       PRINT "Seleccione un menu"
       LOCATE IX + N + 5, 23
       PRINT "y presione RETORNO para entrar"


      LOCATE IX + 1, IY
      COLOR 0, 15
      PRINT A$(1)
      ICONT = 0

LAZO1:   T$ = INKEY$

       IF T$ = CHR$(0) + CHR$(80) THEN
        ICONT = ICONT + 1
        IF ICONT < N THEN
           LOCATE IX + ICONT, IY
           COLOR 15, 1
           PRINT A$(ICONT)
           LOCATE IX + 1 + ICONT, IY
           COLOR 0, 15
           PRINT A$(ICONT + 1)
           GOTO LAZO1
        ELSE
           LOCATE IX + N, IY
           COLOR 15, 1
           PRINT A$(N)
           LOCATE IX + 1, IY
           COLOR 0, 15
           PRINT A$(1)
           ICONT = 0
           GOTO LAZO1
        END IF
       ELSEIF T$ = CHR$(0) + CHR$(72) THEN
        ICONT = ICONT - 1
        IF ICONT < 0 THEN
           LOCATE IX + 1, IY
           COLOR 15, 1
           PRINT A$(1)
           LOCATE IX + N, IY
           COLOR 0, 15
           PRINT A$(N)
           ICONT = N - 1
           GOTO LAZO1
        ELSE
           LOCATE IX + ICONT + 2, IY
           COLOR 15, 1
           PRINT A$(ICONT + 2)
           LOCATE IX + ICONT + 1, IY
           COLOR 0, 15
```

```
            PRINT A$(ICONT + 1)
            GOTO LAZO1
          END IF

        ELSEIF T$ = CHR$(13) THEN
            KSW = ICONT + 1
            COLOR 12, 0
            ON KSW GOTO UNO, DOS, TRES, CUATRO, CINCO
        ELSE
          GOTO LAZO1
        END IF
UNO:
      CLS
      LOCATE 13, 15
      PRINT "Cargando el modulo escogido.  Por favor espere..."
      COLOR 15, 0
      CHAIN "MENU1F"


DOS:
      CLS
      LOCATE 13, 15
      PRINT "Cargando el modulo escogido.  Por favor espere..."
      COLOR 15, 0
      CHAIN "MENU2F"

TRES:
      CLS
      LOCATE 13, 15
      PRINT "Cargando el modulo escogido.  Por favor espere..."
      COLOR 15, 0
      CHAIN "MENU3F"

CUATRO:
      CLS
      LOCATE 13, 15
      PRINT "Cargando el modulo escogido.  Por favor espere..."
      COLOR 15, 0
      CHAIN "MENU4F"

CINCO:
      CLS
      LOCATE 13, 25
      COLOR 15, 0
      PRINT "GRACIAS POR UTILIZAR EL PROGRAMA"

      END
```

```
'
'   Pantallas de presentacion
'

    CLEAR
    DEFINT I-N
    DEFDBL A-H, O-Z
    CLS
    LOCATE 2, 13
    COLOR 15, 0
    PRINT "ÿ";
    PRINT STRING$(54, "ƒ");
    PRINT "ø";
    LOCATE 3, 13
    PRINT "≥          "; "ESCUELA POLITECNICA NACIONAL";
    PRINT "          ≥"
    LOCATE 4, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 5, 13
    PRINT "≥ "; "METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA"; " ≥"
    LOCATE 6, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 7, 13
    PRINT "≥          ";
    PRINT "Autor: Juan Fernando Donoso A."; "          ≥"
    LOCATE 8, 13
    PRINT "¿";
    PRINT STRING$(54, "ƒ");
    PRINT "Ÿ"

    DIM A$(30)
    N = 7
    A$(1) = "Primera Forma de Cauer"
    A$(2) = "Segunda Forma de Cauer"
    A$(3) = "Tercera Forma de Cauer"
    A$(4) = "Forma Modificada de Cauer"
    A$(5) = "Forma General de Cauer"
    A$(6) = "Nueva Forma General de Cauer"
    A$(7) = "MENU ANTERIOR"

    mmax = 0
    FOR I = 1 TO N
    maux = LEN(A$(I))
    IF mmax < maux THEN
      mmax = maux
    ELSE
    END IF
    NEXT I
    IY = (80 - mmax - 6) / 2
    IX = 10

    LOCATE IX, IY
    cx = mmax + 4
    CY = N + 2
    COLOR 15, 1
    PRINT "ÿ";
    PRINT STRING$(cx, "ƒ");
    PRINT "ø"
    FOR I = 1 TO N
    LOCATE IX + I, IY
    PRINT "≥ ";
```

```
      PRINT A$(I);
      maux = cx - LEN(A$(I)) - 2
      PRINT STRING$(maux, " "); "≥"
      NEXT I
      LOCATE IX - 1 + CY, IY
      PRINT "¿";
      PRINT STRING$(cx, "ƒ");
      PRINT "Ÿ"
      IY = IY + 3

       LOCATE IX + N + 4, 30
       COLOR 12, 0
       PRINT "Seleccione un menu"
       LOCATE IX + N + 5, 23
       PRINT "y presione RETORNO para entrar"


      LOCATE IX + 1, IY
      COLOR 0, 15
      PRINT A$(1)
      ICONT = 0

LAZO1:   T$ = INKEY$

       IF T$ = CHR$(0) + CHR$(80) THEN
        ICONT = ICONT + 1
        IF ICONT < N THEN
           LOCATE IX + ICONT, IY
           COLOR 15, 1
           PRINT A$(ICONT)
           LOCATE IX + 1 + ICONT, IY
           COLOR 0, 15
           PRINT A$(ICONT + 1)
           GOTO LAZO1
        ELSE
           LOCATE IX + N, IY
           COLOR 15, 1
           PRINT A$(N)
           LOCATE IX + 1, IY
           COLOR 0, 15
           PRINT A$(1)
           ICONT = 0
           GOTO LAZO1
        END IF
       ELSEIF T$ = CHR$(0) + CHR$(72) THEN
        ICONT = ICONT - 1
        IF ICONT < 0 THEN
           LOCATE IX + 1, IY
           COLOR 15, 1
           PRINT A$(1)
           LOCATE IX + N, IY
           COLOR 0, 15
           PRINT A$(N)
           ICONT = N - 1
           GOTO LAZO1
        ELSE
           LOCATE IX + ICONT + 2, IY
           COLOR 15, 1
           PRINT A$(ICONT + 2)
           LOCATE IX + ICONT + 1, IY
           COLOR 0, 15
           PRINT A$(ICONT + 1)
```

```
        GOTO LAZO1
      END IF

    ELSEIF T$ = CHR$(13) THEN
        KSW = ICONT + 1
        COLOR 12, 0
        ON KSW GOTO UNO, DOS, TRES, CUATRO, CINCO, SEIS, SIETE
    ELSE
      GOTO LAZO1
    END IF
UNO:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "PFC1F"


DOS:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "SFC2F"

TRES:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "TFC3F"

CUATRO:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "FMC4F"

CINCO:
    CLS
    LOCATE 13, 25
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "FGC5F"

SEIS:
    CLS
    LOCATE 13, 25
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "NFGC6F"

SIETE:
    CLS
    LOCATE 13, 25
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "MENUPF"


  END
```

```
'
'   Pantallas de presentacion
'

    CLEAR
    DEFINT I-N
    DEFDBL A-H, O-Z
    CLS
    LOCATE 2, 13
    COLOR 15, 0
    PRINT "/";
    PRINT STRING$(54, "f");
    PRINT "ø";
    LOCATE 3, 13
    PRINT "≥          "; "ESCUELA POLITECNICA NACIONAL";
    PRINT "          ≥"
    LOCATE 4, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 5, 13
    PRINT "≥ "; "METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA"; " ≥"
    LOCATE 6, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 7, 13
    PRINT "≥          ";
    PRINT "Autor: Juan Fernando Donoso A."; "          ≥"
    LOCATE 8, 13
    PRINT "¿";
    PRINT STRING$(54, "f");
    PRINT "Ÿ"

    DIM A$(30)
    N = 4
    A$(1) = "Metodo de Davison-Chidambara"
    A$(2) = "Metodo de Agregacion"
    A$(3) = "Metodo de Realizacion Parcial"
    A$(4) = "MENU ANTERIOR"

    mmax = 0
    FOR I = 1 TO N
    maux = LEN(A$(I))
    IF mmax < maux THEN
      mmax = maux
    ELSE
    END IF
    NEXT I
    IY = (80 - mmax - 6) / 2
    IX = 10

    LOCATE IX, IY
    cx = mmax + 4
    CY = N + 2
    COLOR 15, 1
    PRINT "/";
    PRINT STRING$(cx, "f");
    PRINT "ø"
    FOR I = 1 TO N
    LOCATE IX + I, IY
    PRINT "≥ ";
    PRINT A$(I);
    maux = cx - LEN(A$(I)) - 2
    PRINT STRING$(maux, " "); "≥"
```

```
    NEXT I
    LOCATE IX - 1 + CY, IY
    PRINT "¿";
    PRINT STRING$(cx, "f");
    PRINT "Ÿ"
    IY = IY + 3

     LOCATE IX + N + 4, 30
     COLOR 12, 0
     PRINT "Seleccione un menu"
     LOCATE IX + N + 5, 23
     PRINT "y presione RETORNO para entrar"


    LOCATE IX + 1, IY
    COLOR 0, 15
    PRINT A$(1)
    ICONT = 0

LAZO1:  T$ = INKEY$

      IF T$ = CHR$(0) + CHR$(80) THEN
       ICONT = ICONT + 1
       IF ICONT < N THEN
          LOCATE IX + ICONT, IY
          COLOR 15, 1
          PRINT A$(ICONT)
          LOCATE IX + 1 + ICONT, IY
          COLOR 0, 15
          PRINT A$(ICONT + 1)
          GOTO LAZO1
       ELSE
          LOCATE IX + N, IY
          COLOR 15, 1
          PRINT A$(N)
          LOCATE IX + 1, IY
          COLOR 0, 15
          PRINT A$(1)
          ICONT = 0
          GOTO LAZO1
       END IF
      ELSEIF T$ = CHR$(0) + CHR$(72) THEN
       ICONT = ICONT - 1
       IF ICONT < 0 THEN
          LOCATE IX + 1, IY
          COLOR 15, 1
          PRINT A$(1)
          LOCATE IX + N, IY
          COLOR 0, 15
          PRINT A$(N)
          ICONT = N - 1
          GOTO LAZO1
       ELSE
          LOCATE IX + ICONT + 2, IY
          COLOR 15, 1
          PRINT A$(ICONT + 2)
          LOCATE IX + ICONT + 1, IY
          COLOR 0, 15
          PRINT A$(ICONT + 1)
          GOTO LAZO1
       END IF
```

```
      ELSEIF T$ = CHR$(13) THEN
          KSW = ICONT + 1
          COLOR 12, 0
          ON KSW GOTO UNO, DOS, TRES, CUATRO
      ELSE
        GOTO LAZO1
      END IF
UNO:
     CLS
     LOCATE 13, 15
     PRINT "Cargando el modulo escogido.  Por favor espere..."
     COLOR 15, 0
     CHAIN "D-CHF"


DOS:
     CLS
     LOCATE 13, 15
     PRINT "Cargando el modulo escogido.  Por favor espere..."
     COLOR 15, 0
     CHAIN "AGREG-NF"

TRES:
     CLS
     LOCATE 13, 15
     PRINT "Cargando el modulo escogido.  Por favor espere..."
     COLOR 15, 0
     CHAIN "REALPARF"

CUATRO:
     CLS
     LOCATE 13, 15
     PRINT "Cargando el modulo escogido.  Por favor espere..."
     COLOR 15, 0
     CHAIN "MENUPF"

     END
```

```
'
'   Pantallas de presentacion
'
    CLEAR
    DEFINT I-N
    DEFDBL A-H, O-Z
    CLS
    LOCATE 2, 13
    COLOR 15, 0
    PRINT "ɻ";
    PRINT STRING$(54, "ʃ");
    PRINT "ø";
    LOCATE 3, 13
    PRINT "≥          "; "ESCUELA POLITECNICA NACIONAL";
    PRINT "          ≥"
    LOCATE 4, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 5, 13
    PRINT "≥  "; "METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA"; "  ≥"
    LOCATE 6, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 7, 13
    PRINT "≥          ";
    PRINT "Autor: Juan Fernando Donoso A."; "          ≥"
    LOCATE 8, 13
    PRINT "¿";
    PRINT STRING$(54, "ʃ");
    PRINT "Ÿ"

    DIM A$(30)
    N = 3
    A$(1) = "Metodo de Aproximaciones Optimas"
    A$(2) = "Metodo de los Minimos Cuadrados"
    A$(3) = "MENU ANTERIOR"

    mmax = 0
    FOR I = 1 TO N
    maux = LEN(A$(I))
    IF mmax < maux THEN
      mmax = maux
    ELSE
    END IF
    NEXT I
    IY = (80 - mmax - 6) / 2
    IX = 10

    LOCATE IX, IY
    cx = mmax + 4
    CY = N + 2
    COLOR 15, 1
    PRINT "ɻ";
    PRINT STRING$(cx, "ʃ");
    PRINT "ø"
    FOR I = 1 TO N
    LOCATE IX + I, IY
    PRINT "≥ ";
    PRINT A$(I);
    maux = cx - LEN(A$(I)) - 2
    PRINT STRING$(maux, " "); "≥"
    NEXT I
```

```
      LOCATE IX - 1 + CY, IY
      PRINT "¿";
      PRINT STRING$(cx, "f");
      PRINT "Ÿ"
      IY = IY + 3

       LOCATE IX + N + 4, 30
       COLOR 12, 0
       PRINT "Seleccione un menu"
       LOCATE IX + N + 5, 23
       PRINT "y presione RETORNO para entrar"


      LOCATE IX + 1, IY
      COLOR 0, 15
      PRINT A$(1)
      ICONT = 0

LAZO1:  T$ = INKEY$

       IF T$ = CHR$(0) + CHR$(80) THEN
        ICONT = ICONT + 1
        IF ICONT < N THEN
           LOCATE IX + ICONT, IY
           COLOR 15, 1
           PRINT A$(ICONT)
           LOCATE IX + 1 + ICONT, IY
           COLOR 0, 15
           PRINT A$(ICONT + 1)
           GOTO LAZO1
        ELSE
           LOCATE IX + N, IY
           COLOR 15, 1
           PRINT A$(N)
           LOCATE IX + 1, IY
           COLOR 0, 15
           PRINT A$(1)
           ICONT = 0
           GOTO LAZO1
        END IF
       ELSEIF T$ = CHR$(0) + CHR$(72) THEN
        ICONT = ICONT - 1
        IF ICONT < 0 THEN
           LOCATE IX + 1, IY
           COLOR 15, 1
           PRINT A$(1)
           LOCATE IX + N, IY
           COLOR 0, 15
           PRINT A$(N)
           ICONT = N - 1
           GOTO LAZO1
        ELSE
           LOCATE IX + ICONT + 2, IY
           COLOR 15, 1
           PRINT A$(ICONT + 2)
           LOCATE IX + ICONT + 1, IY
           COLOR 0, 15
           PRINT A$(ICONT + 1)
           GOTO LAZO1
        END IF

       ELSEIF T$ = CHR$(13) THEN
```

```
        KSW = ICONT + 1
        COLOR 12, 0
        ON KSW GOTO UNO, DOS, TRES
      ELSE
        GOTO LAZO1
      END IF
UNO:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "AOF"


DOS:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "MCF"


TRES:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "MENUPF"

    END
```

```
'
'   Pantallas de presentacion
'
    CLEAR
    DEFINT I-N
    DEFDBL A-H, O-Z
    CLS
    LOCATE 2, 13
    COLOR 15, 0
    PRINT "ÿ";
    PRINT STRING$(54, "f");
    PRINT "ø";
    LOCATE 3, 13
    PRINT "≥          "; "ESCUELA POLITECNICA NACIONAL";
    PRINT "          ≥"
    LOCATE 4, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 5, 13
    PRINT "≥ "; "METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA"; " ≥"
    LOCATE 6, 13
    PRINT "≥";
    PRINT STRING$(54, " "); "≥"
    LOCATE 7, 13
    PRINT "≥          ";
    PRINT "Autor: Juan Fernando Donoso A."; "          ≥"
    LOCATE 8, 13
    PRINT "¿";
    PRINT STRING$(54, "f");
    PRINT "Ÿ"

    DIM A$(30)
    N = 2
    A$(1) = "Metodo de los Valores Singulares"
    A$(2) = "MENU ANTERIOR"

    mmax = 0
    FOR I = 1 TO N
    maux = LEN(A$(I))
    IF mmax < maux THEN
      mmax = maux
    ELSE
    END IF
    NEXT I
    IY = (80 - mmax - 6) / 2
    IX = 10

    LOCATE IX, IY
    cx = mmax + 4
    CY = N + 2
    COLOR 15, 1
    PRINT "ÿ";
    PRINT STRING$(cx, "f");
    PRINT "ø"
    FOR I = 1 TO N
    LOCATE IX + I, IY
    PRINT "≥ ";
    PRINT A$(I);
    maux = cx - LEN(A$(I)) - 2
    PRINT STRING$(maux, " "); "≥"
    NEXT I
    LOCATE IX - 1 + CY, IY
```

```
      PRINT "¿";
      PRINT STRING$(cx, "ƒ");
      PRINT "Ÿ"
      IY = IY + 3

       LOCATE IX + N + 4, 30
       COLOR 12, 0
       PRINT "Seleccione un menu"
       LOCATE IX + N + 5, 23
       PRINT "y presione RETORNO para entrar"


      LOCATE IX + 1, IY
      COLOR 0, 15
      PRINT A$(1)
      ICONT = 0

LAZO1:   T$ = INKEY$

       IF T$ = CHR$(0) + CHR$(80) THEN
        ICONT = ICONT + 1
        IF ICONT < N THEN
           LOCATE IX + ICONT, IY
           COLOR 15, 1
           PRINT A$(ICONT)
           LOCATE IX + 1 + ICONT, IY
           COLOR 0, 15
           PRINT A$(ICONT + 1)
           GOTO LAZO1
        ELSE
           LOCATE IX + N, IY
           COLOR 15, 1
           PRINT A$(N)
           LOCATE IX + 1, IY
           COLOR 0, 15
           PRINT A$(1)
           ICONT = 0
           GOTO LAZO1
        END IF
      ELSEIF T$ = CHR$(0) + CHR$(72) THEN
        ICONT = ICONT - 1
        IF ICONT < 0 THEN
           LOCATE IX + 1, IY
           COLOR 15, 1
           PRINT A$(1)
           LOCATE IX + N, IY
           COLOR 0, 15
           PRINT A$(N)
           ICONT = N - 1
           GOTO LAZO1
        ELSE
           LOCATE IX + ICONT + 2, IY
           COLOR 15, 1
           PRINT A$(ICONT + 2)
           LOCATE IX + ICONT + 1, IY
           COLOR 0, 15
           PRINT A$(ICONT + 1)
           GOTO LAZO1
        END IF

      ELSEIF T$ = CHR$(13) THEN
           KSW = ICONT + 1
```

```
        COLOR 12, 0
        ON KSW GOTO UNO, DOS
    ELSE
      GOTO LAZO1
    END IF
UNO:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "VSF"


DOS:
    CLS
    LOCATE 13, 15
    PRINT "Cargando el modulo escogido.  Por favor espere..."
    COLOR 15, 0
    CHAIN "MENUPF"


    END
```

```
DECLARE SUB FRAC (N#, NR#, DR#, G#(), Nu#(), D#(), N1#(), N2#(), N3#(), D1#(), D2#(), D3#())

'    PRIMERA FORMA DE CAUER
'    CALCULO DE COEFICIENTES H
'

1    CLEAR
     DEFINT I-M
     DEFDBL A-H, N-Z
     CLS
     PRINT "PRIMERA FORMA DE CAUER"
     PRINT STRING$(22, "=")
     PRINT : PRINT

9    PRINT : PRINT
     PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
     PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
     PRINT TAB(20); "IMPRESION PANTALLA (P)"
     PRINT : PRINT TAB(20); "Ingrese I, D o P";
     INPUT X$
     IMPR$ = "LPT1:"
     IF X$ = "I" OR X$ = "i" THEN
     GOTO 10
     ELSEIF X$ = "D" OR X$ = "d" THEN
     GOTO 11
     ELSEIF X$ = "P" OR X$ = "p" THEN
     GOTO 12
     ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
     GOTO 9
     END IF
12   IMPR$ = "SCRN:"
     GOTO 10
11   CLS
     PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
     INPUT AD$
     IMPR$ = AD$
10   OPEN IMPR$ FOR OUTPUT AS #1

15   CLS
     PRINT "INGRESO DE DATOS"
     PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ORIGINAL. (N>2) N =";
     INPUT N
     IF N <= 2 THEN 15
5    PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ";
     PRINT "DE TRANSFERENCIA REDUCIDA"
     PRINT "(M no puede ser mayor o igual que N ni menor o igual que 0); M = ";
     INPUT M
     IF M >= N OR M <= 0 THEN 5
     PRINT
6    PRINT "ESTA CORRECTO (S/N)";
     INPUT A$
     IF A$ = "N" OR A$ = "n" THEN 1
     IF A$ <> "S" AND A$ <> "s" THEN 6

     REDIM B(2 * N, N + 1), b(2 * M)
     FOR I = 1 TO 2 * N
     FOR J = 1 TO N + 1
     B(I, J) = 0
     NEXT J, I
7    CLS
```

```
    PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO N + 1
    PRINT "COEFICIENTE GRADO "; N - I + 1; " = ";
    INPUT B(1, I)
    NEXT I
    PRINT
    PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO N
    PRINT "COEFICIENTE GRADO "; N - I; " = ";
    INPUT B(2, I)
    NEXT I
    PRINT : PRINT
2   PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN 7
    IF P$ <> "S" AND P$ <> "s" THEN 2

    CLS
    PRINT #1, "FUNCION DE TRANSFERENCIA ORIGINAL"
    PRINT #1,
    PRINT #1, "GRADO NUMERADOR = "; N - 1
    PRINT #1,
    FOR I = 1 TO N
    PRINT #1, "COEFICIENTE DE GRADO "; N - I; " = "; B(2, I)
    NEXT I
    PRINT #1,
    PRINT #1, "GRADO DENOMINADOR = "; N
    PRINT #1,
    FOR I = 1 TO N + 1
    PRINT #1, "COEFICIENTE GRADO "; N - I + 1; " = "; B(1, I)
    NEXT I
    PRINT #1,
    PRINT "Presione cualquier tecla para continuar"
8   IF INKEY$ = "" THEN 8
    CLS
    K = 1
    I = 3
    WHILE K <= 2 * M
      IF B(K + 1, 1) = 0 THEN
        CLS
        PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
        PRINT #1, : PRINT #1, "LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
        GOTO FIN
      ELSE
      END IF
      b(K) = B(K, 1) / B(K + 1, 1)
      FOR J = 1 TO N
      B(I, J) = B(I - 2, J + 1) - h(I - 2) * B(I - 1, J + 1)
      NEXT J
      I = I + 1
      K = K + 1
    WEND

'
'   COMIENZA EL CALCULO DE FRACCIONES CONTINUAS
'

    NF1 = 2
    REDIM GF(4)
     GF(1) = 1
     GF(2) = 0
```

```
 GF(3) = 0
 GF(4) = 0
REDIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1), NF(M + 5), DF(M + 5), NN3(1), DD3(1)
 NN1(1) = h(2 * M - 1)
 NN1(2) = 0
 NN2(1) = 1
 DD1(1) = 1
 DD2(1) = h(2 * M)
KF = 2
KF1 = 2 * M - KF
WHILE KF <= 2 * M
    CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
    GF(3) = NNR - 1
    GF(1) = DDR - 1
    REDIM NN1(DDR), DD1(NNR)
    FOR I = 1 TO DDR
    NN1(I) = DF(I)
    DF(I) = 0
    NEXT I
    FOR I = 1 TO NNR
    DD1(I) = NF(I)
    NF(I) = 0
    NEXT I
    IF KF1 / 2 = INT(KF1 / 2) THEN
      GF(2) = 0
      GF(4) = 0
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = h(KF1)
      DD2(1) = 1
    ELSE
      GF(2) = 1
      GF(4) = 0
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = h(KF1)
      NN2(2) = 0
      DD2(1) = 1
    END IF
    KF = KF + 1
    KF1 = KF1 - 1
WEND
IF NN1(1) = DD1(1) THEN
  DIV = NN1(1)
  FOR I = 1 TO DDR
  NN1(I) = NN1(I) / DIV
  NEXT I
  FOR I = 1 TO NNR
  DD1(I) = DD1(I) / DIV
  NEXT I
ELSE
END IF

'
'   IMPRESION DE RESULTADOS
'

   CLS
   PRINT #1,
   PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
   PRINT #1,
   PRINT #1,
   NNR1 = NNR - 1
   DDR1 = DDR - 1
```

```
    PRINT #1, "GRADO NUMERADOR "; DDR1
    PRINT #1,
    FOR I = 1 TO DDR
    J = DDR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; NN1(I)
    NEXT I
    PRINT #1,
    PRINT #1, "GRADO DENOMINADOR "; NNR1
    PRINT #1,
    FOR I = 1 TO NNR
    J = NNR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; DD1(I)
    NEXT I
FIN:
    PRINT
    PRINT "Presione cualquier tecla para continuar ";
3   IF INKEY$ = "" THEN 3
    CHAIN "MENU1F"

    END
```

```
DECLARE SUB FRAC (N#, NR#, DR#, G#(), Nu#(), D#(), N1#(), N2#(), N3#(), D1#(), D2#(), D3#())

'    SEGUNDA FORMA DE CAUER
'    CALCULO DE COEFICIENTES h
'
1     CLEAR
     CLS
     DEFINT I-M
     DEFDBL A-H, N-Z
     PRINT "SEGUNDA FORMA DE CAUER"
     PRINT STRING$(22, "=")
     PRINT : PRINT

9     PRINT : PRINT
     PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
     PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
     PRINT TAB(20); "IMPRESION PANTALLA (P)"
     PRINT : PRINT TAB(20); "Ingrese I, D o P";
     INPUT X$
     IMPR$ = "LPT1:"
     IF X$ = "I" OR X$ = "i" THEN
     GOTO 10
     ELSEIF X$ = "D" OR X$ = "d" THEN
     GOTO 11
     ELSEIF X$ = "P" OR X$ = "p" THEN
     GOTO 12
     ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
     GOTO 9
     END IF
12    IMPR$ = "SCRN:"
     GOTO 10
11    CLS
     PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
     INPUT AD$
     IMPR$ = AD$
10    OPEN IMPR$ FOR OUTPUT AS #1

15    CLS
     PRINT "INGRESO DE DATOS"
     PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ORIGINAL. (N>2) N =";
     INPUT N
     IF N <= 2 THEN 15
5     PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ";
     PRINT "DE TRANSFERENCIA REDUCIDA"
     PRINT "(M no puede ser mayor o igual que N ni menor o igual que 0); M = ";
     INPUT M
     IF M >= N OR M <= 0 THEN 5
     PRINT
6     PRINT "ESTA CORRECTO (S/N)";
     INPUT A$
     IF A$ = "N" OR A$ = "n" THEN 1
     IF A$ <> "S" AND A$ <> "s" THEN 6

     DIM A(2 * N, N + 1), h(2 * M)
     FOR I = 1 TO 2 * N
     FOR J = 1 TO N + 1
     A(I, J) = 0
     NEXT J, I

7     CLS
```

```
PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MENOR"
PRINT
FOR I = 1 TO N + 1
PRINT "COEFICIENTE GRADO "; I - 1; " = ";
INPUT A(1, I)
NEXT I
PRINT
PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MENOR"
PRINT
FOR I = 1 TO N
PRINT "COEFICIENTE GRADO "; I - 1; " = ";
INPUT A(2, I)
NEXT I
PRINT : PRINT
2  PRINT "ESTA CORRECTO (S/N)";
INPUT P$
IF P$ = "N" OR P$ = "n" THEN 7
IF P$ <> "S" AND P$ <> "s" THEN 2

CLS
PRINT #1, "FUNCION DE TRANSFERENCIA ORIGINAL"
PRINT #1,
PRINT #1, "GRADO NUMERADOR = "; N - 1
PRINT #1,
FOR I = 1 TO N
PRINT #1, "COEFICIENTE DE GRADO "; N - I; " = "; A(2, N + 1 - I)
NEXT I
PRINT #1,
PRINT #1, "GRADO DENOMINADOR = "; N
PRINT #1,
FOR I = 1 TO N + 1
PRINT #1, "COEFICIENTE GRADO "; N - I + 1; " = "; A(1, N + 2 - I)
NEXT I
PRINT #1,
PRINT "Presione cualquier tecla para continuar"
8  IF INKEY$ = "" THEN 8

K = 1
I = 3
WHILE K <= 2 * M
  IF A(K + 1, 1) = 0 THEN
    CLS
    PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
    PRINT #1,
    PRINT #1, " LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
    GOTO FIN
  ELSE
  END IF
  h(K) = A(K, 1) / A(K + 1, 1)
  FOR J = 1 TO N
  A(I, J) = A(I - 2, J + 1) - h(I - 2) * A(I - 1, J + 1)
  NEXT J
  I = I + 1
  K = K + 1
WEND
'
'  COMIENZA EL CALCULO DE FRACCIONES CONTINUAS
'
NF1 = 2
DIM GF(4)
 GF(1) = 0
 GF(2) = 1
```

```
 GF(3) = 0
 GF(4) = 0
DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1), NF(M + 5), DF(M + 5), NN3(1), DD3(1)
 NN1(1) = h(2 * M - 1)

 NN2(1) = 1
 NN2(2) = 0
 DD1(1) = 1
 DD2(1) = h(2 * M)
KF = 2
KF1 = 2 * M - KF
WHILE KF <= 2 * M
    CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
    GF(3) = NNR - 1
    GF(1) = DDR - 1
    REDIM NN1(DDR), DD1(NNR)
    FOR I = 1 TO DDR
    NN1(I) = DF(I)
    DF(I) = 0
    NEXT I
    FOR I = 1 TO NNR
    DD1(I) = NF(I)
    NF(I) = 0
    NEXT I
    IF KF1 / 2 = INT(KF1 / 2) THEN
      GF(2) = 0
      GF(4) = 1
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = h(KF1)
      DD2(1) = 1
      DD2(2) = 0
    ELSE
      GF(2) = 0
      GF(4) = 0
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = h(KF1)

      DD2(1) = 1
    END IF
    KF = KF + 1
    KF1 = KF1 - 1
WEND
IF NN1(1) = DD1(1) THEN
  DIV = NN1(1)
  FOR I = 1 TO DDR
  NN1(I) = NN1(I) / DIV
  NEXT I
  FOR I = 1 TO NNR
  DD1(I) = DD1(I) / DIV
  NEXT I
ELSE
END IF


' IMPRESION DE RESULTADOS


CLS
PRINT #1,
PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
PRINT #1,
PRINT #1,
```

```
    NNR1 = NNR - 1
    DDR1 = DDR - 1
    PRINT #1, "GRADO NUMERADOR "; DDR1
    PRINT #1,
    FOR I = 1 TO DDR
    J = DDR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; NN1(I)
    NEXT I
    PRINT #1,
    PRINT #1, "GRADO DENOMINADOR "; NNR1
    PRINT #1,
    FOR I = 1 TO NNR
    J = NNR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; DD1(I)
    NEXT I
FIN:
    PRINT
    PRINT "Presione cualquier tecla para continuar ";
3   IF INKEY$ = "" THEN 3
    CHAIN "MENU1F"
    END
```

```
DECLARE SUB FRAC (N#, NR#, DR#, G#(), Nu#(), D#(), N1#(), N2#(), N3#(), D1#(), D2#(), D3#())
'    TERCERA FORMA DE CAUER
'    CALCULO DE COEFICIENTES H Y h
'
1     CLEAR
     CLS
     DEFINT I-M
     DEFDBL A-H, N-Z
     PRINT "TERCERA FORMA DE CAUER"
     PRINT STRING$(22, "=")
     PRINT : PRINT

9     PRINT : PRINT
     PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
     PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
     PRINT TAB(20); "IMPRESION PANTALLA (P)"
     PRINT : PRINT TAB(20); "Ingrese I, D o P";
     INPUT X$
     IMPR$ = "LPT1:"
     IF X$ = "I" OR X$ = "i" THEN
     GOTO 10
     ELSEIF X$ = "D" OR X$ = "d" THEN
     GOTO 11
     ELSEIF X$ = "P" OR X$ = "p" THEN
     GOTO 12
     ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
     GOTO 9
     END IF
12    IMPR$ = "SCRN:"
     GOTO 10
11    CLS
     PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
     INPUT AD$
     IMPR$ = AD$
10    OPEN IMPR$ FOR OUTPUT AS #1

15    CLS
     PRINT "INGRESO DE DATOS"
     PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ORIGINAL. (N>2) N =";
     INPUT N
     IF N <= 2 THEN 15
5     PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ";
     PRINT "DE TRANSFERENCIA REDUCIDA"
     PRINT "(M no puede ser mayor o igual que N ni menor o igual que 0); M = ";
     INPUT M
     IF M >= N OR M <= 0 THEN 5
     PRINT
6     PRINT "ESTA CORRECTO (S/N)";
     INPUT A$
     IF A$ = "N" OR A$ = "n" THEN 1
     IF A$ <> "S" AND A$ <> "s" THEN 6

     DIM A(M + 2, N + 1), H1(M), H2(M)
     FOR I = 1 TO M + 2
     FOR J = 1 TO N + 1
     A(I, J) = 0
     NEXT J, I

7     CLS
     PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MENOR"
```

```
    PRINT
    FOR I = 1 TO N + 1
    PRINT "COEFICIENTE GRADO "; I - 1; " = ";
    INPUT A(1, I)
    NEXT I
    PRINT
    PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MENOR"
    PRINT
    FOR I = 1 TO N
    PRINT "COEFICIENTE GRADO "; I - 1; " = ";
    INPUT A(2, I)
    NEXT I
    PRINT : PRINT
2   PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN 7
    IF P$ <> "S" AND P$ <> "s" THEN 2

    CLS
    PRINT #1, "FUNCION DE TRANSFERENCIA ORIGINAL"
    PRINT #1,
    PRINT #1, "GRADO NUMERADOR = "; N - 1
    PRINT #1,
    FOR I = 1 TO N
    PRINT #1, "COEFICIENTE DE GRADO "; N - I; " = "; A(2, N + 1 - I)
    NEXT I
    PRINT #1,
    PRINT #1, "GRADO DENOMINADOR = "; N
    PRINT #1,
    FOR I = 1 TO N + 1
    PRINT #1, "COEFICIENTE GRADO "; N - I + 1; " = "; A(1, N + 2 - I)
    NEXT I
    PRINT #1,
    PRINT "Presione cualquier tecla para continuar"
8   IF INKEY$ = "" THEN 8


    K = 1
    I = 3
    WHILE K <= M
      IF A(K + 1, 1) = 0 OR A(K + 1, N + 1 - K) = 0 THEN
        CLS
        PRINT #1, " NO HAY SOLUCION POR ESTE METODO"
        PRINT #1,
        PRINT #1, " LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
        GOTO FIN
      ELSE
      END IF
      H1(K) = A(K, 1) / A(K + 1, 1)
      H2(K) = A(K, N + 2 - K) / A(K + 1, N + 1 - K)
      FOR J = 1 TO N
      A(I, J) = A(I - 2, J + 1) - H1(I - 2) * A(I - 1, J + 1) - H2(I - 2) * A(I - 1, J)
      NEXT J
      I = I + 1
      K = K + 1
    WEND


'
'   COMIENZA EL CALCULO DE FRACCIONES CONTINUAS
'
    NF1 = 3
```

```
    DIM GF(6)
    IF M / 2 = INT(M / 2) THEN
     GF(1) = 0
     GF(2) = 0
     GF(3) = 0
     GF(4) = 0
     GF(5) = 1
     GF(6) = 0
     IX = 1
    ELSE
     GF(1) = 1
     GF(2) = 0
     GF(3) = 0
     GF(4) = 0
     GF(5) = 0
     GF(6) = 0
     IX = -1
    END IF
    DIM NN1(GF(1) + 1), NN2(GF(2) + 1), NN3(GF(3) + 1), DD1(GF(4) + 1), DD2(GF(5) + 1), DD3(GF(6) + 1), NF(M + 5),
DF(M + 5)
    IF IX = 1 THEN
     NN1(1) = H2(M)
     NN2(1) = H1(M)
     NN3(1) = 0
     DD1(1) = 1
     DD2(1) = 1
     DD2(2) = 0
     DD3(1) = 1
    ELSE
     NN1(1) = H2(M)
     NN1(2) = 0
     NN2(1) = H1(M)
     NN3(1) = 0
     DD1(1) = 1
     DD2(1) = 1
     DD3(1) = 1
    END IF
    KF = 1
    KF1 = M - KF
    WHILE KF <= M

       CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())

       GF(6) = NNR - 1
       GF(3) = DDR - 1
       REDIM NN3(DDR), DD3(NNR)
       FOR I = 1 TO DDR
       NN3(I) = DF(I)
       DF(I) = 0
       NEXT I
       FOR I = 1 TO NNR
       DD3(I) = NF(I)
       NF(I) = 0
       NEXT I
       IF IX = 1 THEN
         GF(1) = 1
         GF(5) = 0
         REDIM NN1(GF(1) + 1), DD2(GF(5) + 1)
         NN1(1) = H2(KF1)
         NN1(2) = 0
         NN2(1) = H1(KF1)
         DD2(1) = 1
```

```
          IX = -1 * IX
        ELSE
          GF(1) = 0
          GF(5) = 1
          REDIM NN1(GF(1) + 1), DD2(GF(5) + 1)
          NN1(1) = H2(KF1)
          NN2(1) = H1(KF1)
          DD2(1) = 1
          DD2(2) = 0
          IX = -1 * IX
        END IF
        KF = KF + 1
        KF1 = KF1 - 1
    WEND
    IF NN3(1) = DD3(1) THEN
      DIV = NN3(1)
      FOR I = 1 TO DDR
      NN3(I) = NN3(I) / DIV
      NEXT I
      FOR I = 1 TO NNR
      DD3(I) = DD3(I) / DIV
      NEXT I
    ELSE
    END IF

'
'    IMPRESION DE RESULTADOS
'

    CLS
    PRINT #1,
    PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
    PRINT #1,
    PRINT #1,
    NNR1 = NNR - 1
    DDR1 = DDR - 1
    PRINT #1, "GRADO NUMERADOR "; DDR1
    PRINT #1,
    FOR I = 1 TO DDR
    J = DDR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; NN3(I)
    NEXT I
    PRINT #1,
    PRINT #1, "GRADO DENOMINADOR "; NNR1
    PRINT #1,
    FOR I = 1 TO NNR
    J = NNR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; DD3(I)
    NEXT I
FIN:
    PRINT
    PRINT "Presione cualquier tecla para continuar ";
3   IF INKEY$ = "" THEN 3
    CHAIN "MENU1F"
    END
```

```
DECLARE SUB FRAC (N#, NR#, DR#, G#(), Nu#(), D#(), N1#(), N2#(), N3#(), D1#(), D2#(), D3#())

'    FORMA MODIFICADA DE CAUER
'    CALCULO DE COEFICIENTES H Y h
'
1     CLEAR
      CLS
      DEFINT I-M
      DEFDBL A-H, N-Z
      PRINT "FORMA MODIFICADA DE CAUER"
      PRINT STRING$(25, "=")
      PRINT : PRINT

9     PRINT : PRINT
      PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
      PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
      PRINT TAB(20); "IMPRESION PANTALLA (P)"
      PRINT : PRINT TAB(20); "Ingrese I, D o P";
      INPUT X$
      IMPR$ = "LPT1:"
      IF X$ = "I" OR X$ = "i" THEN
      GOTO 10
      ELSEIF X$ = "D" OR X$ = "d" THEN
      GOTO 11
      ELSEIF X$ = "P" OR X$ = "p" THEN
      GOTO 12
      ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
      GOTO 9
      END IF
12    IMPR$ = "SCRN:"
      GOTO 10
11    CLS
      PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
      INPUT AD$
      IMPR$ = AD$
10    OPEN IMPR$ FOR OUTPUT AS #1

25    CLS
      PRINT "INGRESO DE DATOS"
      PRINT
      PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ORIGINAL. (N>2) N =";
      INPUT N
      IF N <= 2 THEN 25
15    PRINT
      PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ";
      PRINT "DE TRANSFERENCIA REDUCIDA"
      PRINT "(M no puede ser mayor o igual que N ni menor o igual que 0); M = ";
      INPUT M
      IF M >= N OR M <= 0 THEN 15
      PRINT
16    PRINT "ESTA CORRECTO (S/N)";
      INPUT A$
      IF A$ = "N" OR A$ = "n" THEN 1
      IF A$ <> "S" AND A$ <> "s" THEN 16

      DIM A(2 * M + 2, N + 1), H1(M), H2(M)
      FOR I = 1 TO 2 * M + 2
      FOR J = 1 TO N + 1
      A(I, J) = 0
      NEXT J, I

17    CLS
```

```
      PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MENOR"
      PRINT
      FOR I = 1 TO N + 1
      PRINT "COEFICIENTE GRADO "; I - 1; " = ";
      INPUT A(1, I)
      NEXT I
      PRINT
      PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MENOR"
      PRINT
      FOR I = 1 TO N
      PRINT "COEFICIENTE GRADO "; I - 1; " = ";
      INPUT A(2, I)
      NEXT I
      PRINT : PRINT
2     PRINT "ESTA CORRECTO (S/N)";
      INPUT P$
      IF P$ = "N" OR P$ = "n" THEN 17
      IF P$ <> "S" AND P$ <> "s" THEN 2

      CLS
      PRINT #1, "FUNCION DE TRANSFERENCIA ORIGINAL"
      PRINT #1,
      PRINT #1, "GRADO NUMERADOR = "; N - 1
      PRINT #1,
      FOR I = 1 TO N
      PRINT #1, "COEFICIENTE DE GRADO "; N - I; " = "; A(2, N + 1 - I)
      NEXT I
      PRINT #1,
      PRINT #1, "GRADO DENOMINADOR = "; N
      PRINT #1,
      FOR I = 1 TO N + 1
      PRINT #1, "COEFICIENTE GRADO "; N - I + 1; " = "; A(1, N + 2 - I)
      NEXT I
      PRINT #1,
      PRINT "Presione cualquier tecla para continuar"
8     IF INKEY$ = "" THEN 8

      K = 1
      I1 = 3
      I2 = 4
      WHILE K <= M
        IF A(2 * (K - 1) + 2, 1) = 0 THEN
          CLS
          PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
          PRINT #1,
          PRINT #1, "LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
          GOTO FIN
        ELSE
        END IF
        H1(K) = A(2 * (K - 1) + 1, 1) / A(2 * (K - 1) + 2, 1)

        FOR J = 1 TO N
        A(I1, J) = A(I1 - 2, J + 1) - H1((I1 - 1) / 2) * A(I1 - 1, J + 1)
        NEXT J

        IF A(2 * (K - 1) + 3, N + 1 - K) = 0 THEN
          CLS
          PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
          PRINT #1,
          PRINT #1, "LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
          GOTO FIN
        ELSE
```

```
  END IF
  H2(K) = A(2 * (K - 1) + 2, N + 1 - K) / A(2 * (K - 1) + 3, N + 1 - K)

  FOR J = 1 TO N
  A(I2, J) = A(I2 - 2, J) - H2((I2 - 2) / 2) * A(I2 - 1, J)
  NEXT J

  I1 = I1 + 2
  I2 = I2 + 2
  K = K + 1
WEND


COMIENZA EL CALCULO DE FRACCIONES CONTINUAS

NF1 = 2
DIM GF(4)
 GF(1) = 1
 GF(2) = 0
 GF(3) = 0
 GF(4) = 0
DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1), NF(M + 5), DF(M + 5), NN3(1), DD3(1)
 NN1(1) = 1
 NN1(2) = 0
 NN2(1) = H1(M)
 DD1(1) = H2(M)
 DD2(1) = 1
KF = 1
KF1 = M - KF
KF2 = KF1
  DO

    CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())

    GF(3) = NNR - 1
    GF(1) = DDR - 1
    REDIM NN1(DDR), DD1(NNR)
    FOR I = 1 TO DDR
    NN1(I) = DF(I)
    DF(I) = 0
    NEXT I
    FOR I = 1 TO NNR
    DD1(I) = NF(I)
    NF(I) = 0
    NEXT I
    NN2(1) = H2(KF1)
    KF2 = KF2 - 1
      IF KF2 < 0 THEN 5

    CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())

    GF(3) = NNR - 1
    GF(1) = DDR
    REDIM NN1(DDR + 1), DD1(NNR)
    FOR I = 1 TO DDR
    NN1(I) = DF(I)
    DF(I) = 0
    NEXT I
    FOR I = 1 TO NNR
    DD1(I) = NF(I)
    NF(I) = 0
    NEXT I
```

```
        NN2(1) = H1(KF1)

        KF = KF + 1
        KF1 = KF1 - 1

6       LOOP WHILE KF <= M
        GOTO 7

5     KF = M + 1
      GOTO 6

7      IF NN1(1) = DD1(1) THEN
        DIV = NN1(1)
        FOR I = 1 TO DDR
        NN1(I) = NN1(I) / DIV
        NEXT I
        FOR I = 1 TO NNR
        DD1(I) = DD1(I) / DIV
        NEXT I
      ELSE
      END IF

'
'     IMPRESION DE RESULTADOS
'
      CLS
      PRINT #1,
      PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
      PRINT #1,
      PRINT #1,
      NNR1 = NNR - 1
      DDR1 = DDR - 1
      PRINT #1, "GRADO NUMERADOR "; DDR1
      PRINT #1,
      FOR I = 1 TO DDR
      J = DDR - I
      PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; NN1(I)
      NEXT I
      PRINT #1,
      PRINT #1, "GRADO DENOMINADOR "; NNR1
      PRINT #1,
      FOR I = 1 TO NNR
      J = NNR - I
      PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; DD1(I)
      NEXT I
FIN:
      PRINT
      PRINT "Presione cualquier tecla para continuar ";
3     IF INKEY$ = "" THEN 3
      CHAIN "MENU1F"
      END
```

```
DECLARE SUB FRAC (N#, NR#, DR#, G#(), Nu#(), D#(), N1#(), N2#(), N3#(), D1#(), D2#(), D3#())

'    FORMA GENERAL DE CAUER
'    CALCULO DE COEFICIENTES H Y h
'
1     CLEAR
     CLS
     DEFINT I-M
     DEFDBL A-H, N-Z
     PRINT "FORMA GENERAL DE CAUER"
     PRINT STRING$(22, "=")

9     PRINT : PRINT
     PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
     PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
     PRINT TAB(20); "IMPRESION PANTALLA (P)"
     PRINT : PRINT TAB(20); "Ingrese I, D o P";
     INPUT X$
     IMPR$ = "LPT1:"
     IF X$ = "I" OR X$ = "i" THEN
     GOTO 10
     ELSEIF X$ = "D" OR X$ = "d" THEN
     GOTO 11
     ELSEIF X$ = "P" OR X$ = "p" THEN
     GOTO 12
     ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
     GOTO 9
     END IF
12    IMPR$ = "SCRN:"
     GOTO 10
11    CLS
     PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
     INPUT AD$
     IMPR$ = AD$
10    OPEN IMPR$ FOR OUTPUT AS #1

25    CLS
     PRINT "INGRESO DE DATOS"
     PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ORIGINAL. (N>2) N =";
     INPUT N
     IF N <= 2 THEN 25
15    PRINT
     PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ";
     PRINT "DE TRANSFERENCIA REDUCIDA"
     PRINT "(M no puede ser mayor o igual que N ni menor o igual que 0); M = ";
     INPUT M
     IF M >= N OR M <= 0 THEN 15
     PRINT
4     PRINT "MOMENTOS DE TIEMPO A FIJAR  (MAXIMO "; 2 * M; "); T = ";
     INPUT T
     IF T > 2 * M OR T < 0 THEN 4
     IF T <> 3 THEN 5
     PRINT : PRINT "ESTE METODO FALLA AL FIJAR TRES MOMENTOS DE TIEMPO"
     PRINT
3     PRINT : PRINT "DESEA CONTINUAR (C) O FIJAR OTROS MOMENTOS DE TIEMPO (T)";
     INPUT Y$
     IF Y$ = "C" OR Y$ = "c" THEN 5
     IF Y$ = "T" OR Y$ = "t" THEN 4
     IF (Y$ <> "T" AND Y$ <> "t") OR (Y$ <> "C" AND Y$ <> "c") THEN 3
5     K1 = 2 * M - T
     PRINT
```

```
6    PRINT "ESTA CORRECTO (S/N)";
     INPUT A$
     IF A$ = "N" OR A$ = "n" THEN 1
     IF A$ <> "S" AND A$ <> "s" THEN 6

     DIM A(2 * M + 4, N + 1), H1(T), H2(2 * M - T)
     FOR I = 1 TO 2 * M + 4
     FOR J = 1 TO N + 1
     A(I, J) = 0
     NEXT J, I

7    CLS
     PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MENOR"
     PRINT
     FOR I = 1 TO N + 1
     PRINT "COEFICIENTE GRADO "; I - 1; " = ";
     INPUT A(1, I)
     NEXT I
     PRINT
     PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MENOR"
     PRINT
     FOR I = 1 TO N
     PRINT "COEFICIENTE GRADO "; I - 1; " = ";
     INPUT A(2, I)
     NEXT I
     PRINT : PRINT
2    PRINT "ESTA CORRECTO (S/N)";
     INPUT P$
     IF P$ = "N" OR P$ = "n" THEN 7
     IF P$ <> "S" AND P$ <> "s" THEN 2

     CLS
     PRINT #1, "FUNCION DE TRANSFERENCIA ORIGINAL"
     PRINT #1,
     PRINT #1, "GRADO NUMERADOR = "; N - 1
     PRINT #1,
     FOR I = 1 TO N
     PRINT #1, "COEFICIENTE DE GRADO "; N - I; " = "; A(2, N + 1 - I)
     NEXT I
     PRINT #1,
     PRINT #1, "GRADO DENOMINADOR = "; N
     PRINT #1,
     FOR I = 1 TO N + 1
     PRINT #1, "COEFICIENTE GRADO "; N - I + 1; " = "; A(1, N + 2 - I)
     NEXT I
     PRINT #1,
     PRINT "Presione cualquier tecla para continuar"
8    IF INKEY$ = "" THEN 8


     K = 1
     I = 3

     WHILE K <= T
       IF A(K + 1, 1) = 0 THEN
         CLS
         PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
         PRINT #1,
         PRINT #1, " LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
         GOTO FIN
       ELSE
       END IF
```

```
  H1(K) = A(K, 1) / A(K + 1, 1)

  FOR J = 1 TO N
  A(I, J) = A(I - 2, J + 1) - H1(I - 2) * A(I - 1, J + 1)

  NEXT J
  I = I + 1
  K = K + 1
WEND
AUX1 = T / 2
AUX2 = INT(AUX1)
IF AUX1 - AUX2 = 0 THEN
  FOR J = 1 TO N + 1 - T
  J1 = N + 2 - T - J
  A(I, J) = A(K, J1)
  NEXT J
  FOR J = 1 TO N - T
  J1 = N + 1 - T - J
  A(I + 1, J) = A(I - 1, J1)
  NEXT J
ELSE
  FOR J = 1 TO N + 1 - T
  J1 = N + 2 - T - J
  A(I, J) = A(K, J1)
  A(I + 1, J) = A(I - 1, J1)
  NEXT J
END IF
K = K + 2
I = K + 2
I1 = 1


WHILE I1 <= K1
  IF A(K + 1, 1) = 0 THEN
    CLS
    PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
    PRINT #1,
    PRINT #1, "LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
    GOTO FIN
  ELSE
  END IF

  H2(I1) = A(K, 1) / A(K + 1, 1)
  FOR J = 1 TO N
  A(I, J) = A(I - 2, J + 1) - H2(I1) * A(I - 1, J + 1)
  NEXT J

  K = K + 1
  I = I + 1
  I1 = I1 + 1
WEND


' COMIENZA EL CALCULO DE FRACCIONES CONTINUAS

NF1 = 2
DIM GF(4), NF(M + 5), DF(M + 5), NN3(1), DD3(1)
IF K1 = 1 THEN
  GF(1) = 0
  GF(2) = 1
  GF(3) = 0
  GF(4) = 0
```

```
DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1)
 NN1(1) = H1(T)
 NN2(1) = 1
 NN2(2) = 0
 DD1(1) = 1
 DD2(1) = H2(K1)
 KF = 2
ELSEIF K1 = 0 THEN
 GF(1) = 0
 GF(2) = 1
 GF(3) = 0
 GF(4) = 0
DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1)
 NN1(1) = H1(T - 1)
 NN2(1) = 1
 NN2(2) = 0
 DD1(1) = 1
 DD2(1) = H1(T)
 KF = 3
ELSE
 GF(1) = 1
 GF(2) = 0
 GF(3) = 0
 GF(4) = 0
DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1)
 NN1(1) = H2(K1 - 1)
 NN1(2) = 0
 NN2(1) = 1
 DD1(1) = 1
 DD2(1) = H2(K1)
 KF = 2
 KF1 = K1 - KF
 IF K1 / 2 = INT(K1 / 2) THEN
   WHILE KF <= K1
    CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
    GF(3) = NNR - 1
    GF(1) = DDR - 1
    REDIM NN1(DDR), DD1(NNR)
    FOR I = 1 TO DDR
    NN1(I) = DF(I)
    DF(I) = 0
    NEXT I
    FOR I = 1 TO NNR
    DD1(I) = NF(I)
    NF(I) = 0
    NEXT I
    IF KF1 / 2 = INT(KF1 / 2) THEN
      GF(2) = 0
      GF(4) = 0
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = H2(KF1)
      DD2(1) = 1
     ELSE
      GF(2) = 1
      GF(4) = 0
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = H2(KF1)
      NN2(2) = 0
      DD2(1) = 1
     END IF
    KF = KF + 1
    KF1 = KF1 - 1
```

```
    WEND
    NN2(1) = H1(T)
    KF = 2
  ELSE
    WHILE KF <= K1 - 1
     CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
     GF(3) = NNR - 1
     GF(1) = DDR - 1
     REDIM NN1(DDR), DD1(NNR)
     FOR I = 1 TO DDR
     NN1(I) = DF(I)
     DF(I) = 0
     NEXT I
     FOR I = 1 TO NNR
     DD1(I) = NF(I)
     NF(I) = 0
     NEXT I
     IF KF1 / 2 <> INT(KF1 / 2) THEN
       GF(2) = 0
       GF(4) = 0
       REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
       NN2(1) = H2(KF1)
       DD2(1) = 1
      ELSE
       GF(2) = 1
       GF(4) = 0
       REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
       NN2(1) = H2(KF1)
       NN2(2) = 0
       DD2(1) = 1
      END IF
     KF = KF + 1
     KF1 = KF1 - 1
    WEND
    KF = 1
  END IF
END IF

KF1 = T - KF
WHILE KF <= T
     CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
     GF(3) = NNR - 1
     GF(1) = DDR
     REDIM NN1(DDR + 1), DD1(NNR)
     FOR I = 1 TO DDR
     NN1(I) = DF(I)
     DF(I) = 0
     NEXT I
     FOR I = 1 TO NNR
     DD1(I) = NF(I)
     NF(I) = 0
     NEXT I
     GF(2) = 0
     GF(4) = 0
     REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
     NN2(1) = H1(KF1 + 1)
     DD2(1) = 1
     KF = KF + 1
     KF1 = KF1 - 1
WEND
IF T <> 0 THEN
NN2(1) = H1(1)
```

```
        CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
        GF(3) = NNR
        GF(1) = DDR
        REDIM NN1(DDR), DD1(NNR)
        FOR I = 1 TO DDR
        NN1(I) = DF(I)
        DF(I) = 0
        NEXT I
        FOR I = 1 TO NNR
        DD1(I) = NF(I)
        NF(I) = 0
        NEXT I
    ELSE
    END IF
    IF NN1(1) = DD1(1) THEN
      DIV = NN1(1)
      FOR I = 1 TO DDR
      NN1(I) = NN1(I) / DIV
      NEXT I
      FOR I = 1 TO NNR
      DD1(I) = DD1(I) / DIV
      NEXT I
    ELSE
    END IF


'
'    IMPRESION DE RESULTADOS
'
    CLS
    PRINT #1,
    PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
    PRINT #1,
    PRINT #1,
    NNR1 = NNR - 1
    DDR1 = DDR - 1
    PRINT #1, "GRADO NUMERADOR "; DDR1
    PRINT #1,
    FOR I = 1 TO DDR
    J = DDR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; NN1(I)
    NEXT I
    PRINT #1,
    PRINT #1, "GRADO DENOMINADOR "; NNR1
    PRINT #1,
    FOR I = 1 TO NNR
    J = NNR - I
    PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; DD1(I)
    NEXT I
FIN:
    PRINT
    PRINT "Presione cualquier tecla para continuar ";
13   IF INKEY$ = "" THEN 13
    CHAIN "MENU1F"
    END
```

```
DECLARE SUB FRAC (N#, NR#, DR#, G#(), Nu#(), D#(), N1#(), N2#(), N3#(), D1#(), D2#(), D3#())

'    NUEVA FORMA GENERAL DE CAUER
'    CALCULO DE COEFICIENTES H Y h
'
1     CLEAR
      CLS
      DEFINT I-M
      DEFDBL A-H, N-Z
      PRINT "NUEVA FORMA GENERAL DE CAUER"
      PRINT STRING$(28, "=")

9     PRINT : PRINT
      PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
      PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
      PRINT TAB(20); "IMPRESION PANTALLA (P)"
      PRINT : PRINT TAB(20); "Ingrese I, D o P";
      INPUT X$
      IMPR$ = "LPT1:"
      IF X$ = "T" OR X$ = "i" THEN
      GOTO 10
      ELSEIF X$ = "D" OR X$ = "d" THEN
      GOTO 11
      ELSEIF X$ = "P" OR X$ = "p" THEN
      GOTO 12
      ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "T" AND X$ <> "i") THEN
      GOTO 9
      END IF
12    IMPR$ = "SCRN:"
      GOTO 10
11    CLS
      PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
      INPUT AD$
      IMPR$ = AD$
10    OPEN IMPR$ FOR OUTPUT AS #1

25    CLS
      PRINT "INGRESO DE DATOS"
      PRINT
      PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ORIGINAL. (N>2) N =";
      INPUT N
      IF N <= 2 THEN 25
5     PRINT
      PRINT "INGRESE EL GRADO DEL DENOMINADOR DE LA FUNCION ";
      PRINT "DE TRANSFERENCIA REDUCIDA"
      PRINT "(M no puede ser mayor o igual que N ni menor o igual que 0); M = ";
      INPUT M
      IF M >= N OR M <= 0 THEN 5
      PRINT
4     PRINT "PARAMETROS DE MARKOV A RETENER (MAXIMO "; 2 * M; "); W = ";
      INPUT W
      IF W > 2 * M OR W < 0 THEN 4
      K1 = 2 * M - W
      PRINT
6     PRINT "ESTA CORRECTO (S/N)";
      INPUT A$
      IF A$ = "N" OR A$ = "n" THEN 1
      IF A$ <> "S" AND A$ <> "s" THEN 6

      DIM A(2 * M + 4, N + 1), H2(W), H1(2 * M - W)
      FOR I = 1 TO 2 * M + 4
      FOR J = 1 TO N + 1
```

```
    A(I, J) = 0
    NEXT J, I

7   CLS
    PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO N + 1
    PRINT "COEFICIENTE GRADO "; N - I + 1; " = ";
    INPUT A(1, I)
    NEXT I
    PRINT
    PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO N
    PRINT "COEFICIENTE GRADO "; N - I; " = ";
    INPUT A(2, I)
    NEXT I
    PRINT : PRINT
2   PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN 7
    IF P$ <> "S" AND P$ <> "s" THEN 2

    CLS
    PRINT #1, "FUNCION DE TRANSFERENCIA ORIGINAL"
    PRINT #1,
    PRINT #1, "GRADO NUMERADOR = "; N - 1
    PRINT #1,
    FOR I = 1 TO N
    PRINT #1, "COEFICIENTE DE GRADO "; N - I; " = "; A(2, I)
    NEXT I
    PRINT #1,
    PRINT #1, "GRADO DENOMINADOR = "; N
    PRINT #1,
    FOR I = 1 TO N + 1
    PRINT #1, "COEFICIENTE GRADO "; N - I + 1; " = "; A(1, I)
    NEXT I
    PRINT #1,
    PRINT "Presione cualquier tecla para continuar"
8   IF INKEY$ = "" THEN 8

    K = 1
    I = 3

    WHILE K <= W
      IF A(K + 1, 1) = 0 THEN
        CLS
        PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
        PRINT #1,
        PRINT #1, "LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
        GOTO FIN
      ELSE
      END IF

      H2(K) = A(K, 1) / A(K + 1, 1)

      FOR J = 1 TO N
      A(I, J) = A(I - 2, J + 1) - H2(I - 2) * A(I - 1, J + 1)

      NEXT J
      I = I + 1
      K = K + 1
```

```
WEND
AUX1 = W / 2
AUX2 = INT(AUX1)
IF AUX1 - AUX2 = 0 THEN
  FOR J = 1 TO N + 1 - W
  J1 = N + 2 - W - J
  A(I, J) = A(K, J1)
  NEXT J
  FOR J = 1 TO N - W
  J1 = N + 1 - W - J
  A(I + 1, J) = A(I - 1, J1)
  NEXT J
ELSE
  FOR J = 1 TO N + 1 - W
  J1 = N + 2 - W - J
  A(I, J) = A(K, J1)
  A(I + 1, J) = A(I - 1, J1)
  NEXT J
END IF
K = K + 2
I = K + 2
I1 = 1


WHILE I1 <= K1
  IF A(K + 1, 1) = 0 THEN
    CLS
    PRINT #1, "NO HAY SOLUCION POR ESTE METODO"
    PRINT #1,
    PRINT #1, "LA FUNCION DE TRANSFERENCIA NO CORRESPONDE A UNA RED DE CAUER"
    GOTO FIN
  ELSE
  END IF

  H1(I1) = A(K, 1) / A(K + 1, 1)
  FOR J = 1 TO N
  A(I, J) = A(I - 2, J + 1) - H1(I1) * A(I - 1, J + 1)
  NEXT J

  K = K + 1
  I = I + 1
  I1 = I1 + 1
WEND


' COMIENZA EL CALCULO DE FRACCIONES CONTINUAS

NF1 = 2
DIM GF(4), NF(M + 5), DF(M + 5), NN3(1), DD3(1)
IF K1 = 1 THEN
  GF(1) = 1
  GF(2) = 0
  GF(3) = 0
  GF(4) = 0
DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1)
  NN1(1) = H2(W)
  NN1(2) = 0
  NN2(1) = 1
  DD1(1) = 1
  DD2(1) = H1(K1)
  KF = 1
ELSEIF K1 = 0 THEN
```

```
  GF(1) = 1
  GF(2) = 0
  GF(3) = 0
  GF(4) = 0
 DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1)
  NN1(1) = H2(W - 1)
  NN1(2) = 0
  NN2(1) = 1
  DD1(1) = 1
  DD2(1) = H2(W)
  KF = 2
 ELSE
  GF(1) = 0
  GF(2) = 1
  GF(3) = 0
  GF(4) = 0
 DIM NN1(GF(1) + 1), NN2(GF(2) + 1), DD1(GF(3) + 1), DD2(GF(4) + 1)
  NN1(1) = H1(K1 - 1)
  NN2(1) = 1
  NN2(2) = 0
  DD1(1) = 1
  DD2(1) = H1(K1)
 IF K1 <> 2 THEN
   KF = 3
   KF1 = K1 - KF
   WHILE KF <= K1
     CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
     GF(3) = NNR - 1
     GF(1) = DDR
     REDIM NN1(DDR + 1), DD1(NNR)
     FOR I = 1 TO DDR
     NN1(I) = DF(I)
     DF(I) = 0
     NEXT I
     FOR I = 1 TO NNR
     DD1(I) = NF(I)
     NF(I) = 0
     NEXT I
     GF(2) = 0
     GF(4) = 0
     REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
     NN2(1) = H1(KF1 + 1)
     DD2(1) = 1
     KF = KF + 1
     KF1 = KF1 - 1
   WEND
 ELSE
 END IF
  CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
  GF(3) = NNR - 1
  GF(1) = DDR - 1
  REDIM NN1(DDR), DD1(NNR)
  FOR I = 1 TO DDR
  NN1(I) = DF(I)
  DF(I) = 0
  NEXT I
  FOR I = 1 TO NNR
  DD1(I) = NF(I)
  NF(I) = 0
  NEXT I
 IF W / 2 = INT(W / 2) THEN
   IF W = 0 THEN
```

```
    KF = 2
  ELSE
    GF(2) = 0
    GF(4) = 0
    REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
    NN2(1) = H2(W)
    DD2(1) = 1
    KF = 1
  END IF
 ELSE
  GF(2) = 1
  GF(4) = 0
  REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
  NN2(1) = H2(W)
  NN2(2) = 0
  DD2(1) = 1
  KF = 1
 END IF
END IF
KF1 = W - KF
WHILE KF <= W
    CALL FRAC(NF1, NNR, DDR, GF(), NF(), DF(), NN1(), NN2(), NN3(), DD1(), DD2(), DD3())
    GF(3) = NNR - 1
    GF(1) = DDR - 1
    REDIM NN1(DDR), DD1(NNR)
    FOR I = 1 TO DDR
    NN1(I) = DF(I)
    DF(I) = 0
    NEXT I
    FOR I = 1 TO NNR
    DD1(I) = NF(I)
    NF(I) = 0
    NEXT I
    IF KF1 / 2 = INT(KF1 / 2) THEN
      GF(2) = 0
      GF(4) = 0
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = H2(KF1)
      DD2(1) = 1
     ELSE
      GF(2) = 1
      GF(4) = 0
      REDIM NN2(GF(2) + 1), DD2(GF(4) + 1)
      NN2(1) = H2(KF1)
      NN2(2) = 0
      DD2(1) = 1
     END IF
    KF = KF + 1
    KF1 = KF1 - 1
   WEND
IF NN1(1) = DD1(1) THEN
  DIV = NN1(1)
  FOR I = 1 TO DDR
  NN1(I) = NN1(I) / DIV
  NEXT I
  FOR I = 1 TO NNR
  DD1(I) = DD1(I) / DIV
  NEXT I
ELSE
END IF
```

```
'
'    IMPRESION DE RESULTADOS
'
     CLS
     PRINT #1,
     PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
     PRINT #1,
     PRINT #1,
     NNR1 = NNR - 1
     DDR1 = DDR - 1
     PRINT #1, "GRADO NUMERADOR "; DDR1
     PRINT #1,
     FOR I = 1 TO DDR
     J = DDR - I
     PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; NN1(I)
     NEXT I
     PRINT #1,
     PRINT #1, "GRADO DENOMINADOR "; NNR1
     PRINT #1,
     FOR I = 1 TO NNR
     J = NNR - I
     PRINT #1, "COEFICIENTE GRADO "; J; " = "; USING "#####.#####"; DD1(I)
     NEXT I
FIN:
     PRINT
     PRINT "Presione cualquier tecla para continuar ";
3    IF INKEY$ = "" THEN 3
     CHAIN "MENU1F"
     END
```

```
DECLARE SUB MBIN (N1%, BE#(), PS#())
DECLARE SUB MINV (N%, AI#(), CI#(), F#)
DECLARE SUB MULTM (N%, N1%, M1%, VR#(), VM1#(), VM2#())
DECLARE SUB ROOT (NG%, A#(), R1#(), RI1#())
'
'    METODO DE DAVISON-CHIDAMBARA
'
'    B1.- Matriz con funcion de transferencia original
'    B.- Matriz del sistema
'    BR.- Matriz sistema reducido
'    AR.- Matriz sistema reducido
'    A.- Matriz del sistema en forma canonica observable
'    C.- Matriz del sistema
'    V.- Matriz modal del sistema original (con vectores propios)
'    VI.- Matriz inversa de la matriz modal
'    RJ.- Forma canonica de Jordan
'    B5, C5, P.- Matrices y arreglo para calculo de valores y vectores
'           propios
'    RD1.- Arreglo con parte real de valores propios
'    RDI1.- Arreglo con parte imaginaria de vectores propios
'    RJ1.- Matriz de ayuda para calcular la forma canonica de Jordan
'    DET.- Arreglo con denominador sistema reducido (dimensionado desde
'        subrutina MBIN)
'    ARI.- Matriz tridimensinal para calculo de Gr(s) (inversa de (sI-A))
'    ER.- Vector de ayuda para calculo de ARI
'    GRN.- Arreglo con numerador de la funcion de transferencia reducida
'        metodo de Davison
'    GN2.- Arreglo con numerador metodo de Chidambara
'    XR.- Arreglo con ganancias sistema despreciado (s=0). Se lo obtiene
'       al dividir valores despreciados de Br para valores despreciados
'       de la diagonal de RJ
'    XD.- Arreglo de ayuda calculo de GN2 (Suma de productos de valores
'       de XR con arreglo DET
'    D.- Vector de ayuda
'    N1.- Numero de filas de la matriz a multiplicar (usar antes de CALL MULTM)
'    M1.- Numero de columnas de la matriz a multiplicar (usar antes de CALL MULTM)
'    MR.- Orden del sistema despreciado en metodo de Davison (MR = N-NR)
'

1     CLEAR
      CLS
      DEFINT I-N
      DEFDBL A-H, O-Z
      PRINT "METODO DE DAVISON CHIDAMBARA"
      PRINT STRING$(28, "=")

9     PRINT : PRINT
      PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
      PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
      PRINT TAB(20); "IMPRESION PANTALLA (P)"
      PRINT : PRINT TAB(20); "Ingrese I, D o P";
      INPUT X$
      IMPR$ = "LPT1:"
      IF X$ = "T" OR X$ = "i" THEN
      GOTO 10
      ELSEIF X$ = "D" OR X$ = "d" THEN
      GOTO 11
      ELSEIF X$ = "P" OR X$ = "p" THEN
      GOTO 12
      ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "T" AND X$ <> "i") THEN
      GOTO 9
      END IF
```

```
12   IMPR$ = "SCRN:"
     GOTO 10
11   CLS
     PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
     INPUT AD$
     IMPR$ = AD$
10   OPEN IMPR$ FOR OUTPUT AS #1


     CLS
     PRINT "INGRESO DE DATOS DE LA FUNCION DE TRANSFERENCIA A REDUCIRSE"
     PRINT

     INPUT "GRADO DENOMINADOR; N = "; N
     IF N <= 0 THEN 1
     PRINT
     INPUT "GRADO NUMERADOR;(M no puede ser mayor o igual que N) M = "; M
     IF M >= N OR M < 0 THEN 1
     DIM B1(2, N + 1), A(N, N), V(N, N), B5(N, N), C5(N, N), P(N), RD1(N), RDI1(N)
     DIM RJ(N, N), VI(N, N), RJ1(N, N), B(N, 1), C(1, N)
     CLS
     PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
     PRINT
     FOR I = 1 TO N + 1
     PRINT "COEFICIENTE GRADO"; N + 1 - I; " = ";
     INPUT B1(1, I)
     NEXT I
     PRINT
     PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
     PRINT
     FOR I = 1 TO M + 1
     PRINT "COEFICIENTE GRADO"; M + 1 - I; " = ";
     INPUT B1(2, N - M + I)
     NEXT I
     PRINT : PRINT
2    PRINT "ESTA CORRECTO (S/N)";
     INPUT P$
     IF P$ = "N" OR P$ = "n" THEN 1
     IF P$ <> "S" AND P$ <> "s" THEN 2

     CLS
     PRINT #1, "FUNCION DE TRANSFERENCIA ORIGINAL"
     PRINT #1,
     PRINT #1, "GRADO NUMERADOR = "; M
     PRINT #1,
     FOR I = 1 TO M + 1
     PRINT #1, "COEFICIENTE DE GRADO "; M + 1 - I; " = "; B1(2, N - M + I)
     NEXT I
     PRINT #1,
     PRINT #1, "GRADO DENOMINADOR = "; N
     PRINT #1,
     FOR I = 1 TO N + 1
     PRINT #1, "COEFICIENTE GRADO "; N - I + 1; " = "; B1(1, I)
     NEXT I
     PRINT #1,
     PRINT "Presione cualquier tecla para continuar"
8    IF INKEY$ = "" THEN 8


'
'    Calculo matrices forma canonica obserbable
'    Matriz A
```

```
'
      FOR I = 1 TO N
      FOR J = 1 TO N
      A(I, J) = 0
      NEXT J, I
      FOR I = 2 TO N
      A(I, I - 1) = 1
      NEXT I
      FOR I = 1 TO N
      A(I, N) = -B1(1, N + 2 - I)
      NEXT I

'
'     Calculo matriz B
'
      FOR I = 1 TO N
      B(I, 1) = 0
      NEXT I
      FOR I = 1 TO M + 1
      B(I, 1) = B1(2, N + 2 - I) - B1(2, 1) * B1(1, N + 2 - I)
      NEXT I

'
'     Calculo matriz C
'
      FOR I = 1 TO N - 1
      C(1, I) = 0
      NEXT I
      C(1, N) = 1

'
'     Comienza calculo de valores propios
'

      FOR I = 1 TO N
      FOR J = 1 TO N
      B5(I, J) = A(I, J)
      NEXT J, I
      FOR K = 1 TO N - 1
      T = 0
      FOR I = 1 TO N
      T = T + B5(I, I)
      NEXT I
      P(K) = T / K
      FOR I = 1 TO N
      B5(I, I) = B5(I, I) - P(K)
      NEXT I
      FOR I = 1 TO N
      FOR J = 1 TO N
      C5(I, J) = B5(I, J)
      NEXT J, I
      FOR I = 1 TO N
      FOR J = 1 TO N
      B5(I, J) = 0
      FOR K1 = 1 TO N
      B5(I, J) = B5(I, J) + A(I, K1) * C5(K1, J)
      NEXT K1, J, I, K
      P(N) = B5(1, 1)
      IF ABS(P(N)) < 1E-08 THEN 1000
      FOR I = 1 TO N
      FOR J = 1 TO N
```

```
    C5(I, J) = C5(I, J) / P(N)
    NEXT J, I
1000  REM   ECUAPOL
    REM
    REM SOLUCION DE ECUACIONES POLINOMIALES
    REM
    REM CALCULO DE LAS RAICES DE LA ECUACION CARACTERISTICA
    REM DE LA MATRIZ A
    REM
    DIM P1(31)
    P1(1) = 1
    FOR J = 2 TO N + 1
    P1(J) = -P(J - 1)
    NEXT J

    CALL ROOT(N, P1(), RD1(), RDI1())

'
'   Ordenamiento de los valores propios de menor a mayor (en valor absoluto)
'
    I = 1
    WHILE I <= N - 1
      AMIN = ABS(RD1(I))
      J = I + 1
      IF RD1(I) = RD1(J) THEN
        I = I + 2
      ELSE
        WHILE J <= N
          IF ABS(RD1(J)) < AMIN THEN
            AMIN = RD1(I)
            AIMG = RDI1(I)
            RD1(I) = RD1(J)
            RDI1(I) = RDI1(J)
            RD1(J) = AMIN
            RDI1(J) = AIMG
            AMIN = ABS(RD1(I))
            J = J + 1
          ELSE
            J = J + 1
          END IF
        WEND
      I = I + 1
      END IF
    WEND
    IF RDI1(N) <> 0 THEN
      CLS
      PRINT #1, "ESTE PROGRAMA NO RESUELVE SISTEMAS CON VALORES PROPIOS"
      PRINT #1, "COMPLEJOS CONJUGADOS, Y ESTE SISTEMA LOS TIENE"
      PRINT #1,
      PRINT #1, "POR FAVOR INTRODUZCA UN SISTEMA CON POLOS REALES"
      GOTO FINAL
    ELSE
    END IF

'
'   Comienza el calculo de vectores propios
'
    K = 1
    WHILE K <= N

    CLS
    LOCATE 9, 16
```

```
PRINT "CALCULANDO VECTOR PROPIO No."; K; "POR FAVOR ESPERE"

  FOR I = 1 TO N
  FOR J = 1 TO N
  B5(I, J) = -A(I, J)
  NEXT J, I
  FOR I = 1 TO N
  B5(I, I) = B5(I, I) + RD1(K)
  NEXT I
  V(N, K) = 1
  FOR I = 1 TO N - 1
  TEMPO = 0
  FOR J = 1 TO I
  TEMPO = TEMPO - B5(N - I + 1, N - J + 1) * V(N - J + 1, K)
  NEXT J
  V(N - I, K) = TEMPO / B5(N - I + 1, N - I)
  NEXT I
  K = K + 1
WEND


'
'   Calculo de la matriz inversa de la matriz modal
'

CALL MINV(N, V(), VI(), F)
IF F = 1 THEN GOTO FINAL


'
'   Calculo de la forma canonica de Jordan
'

N1 = N: M1 = N
CALL MULTM(N, N1, M1, RJ1(), VI(), A())
CALL MULTM(N, N1, M1, RJ(), RJ1(), V())


'
'   Criterio de seleccion del orden del sistema
'

5   CLS
PRINT "DESEA INGRESAR EL ORDEN DEL MODELO REDUCIDO (Presione R)"
PRINT TAB(28); "O"
PRINT "DESEA ENTRAR AL CRITERIO DE SELECCION DEL ORDEN (Presione S)"
PRINT : PRINT "NOTA: El modelo reducido debe ser de orden menor o igual que el original"
PRINT : PRINT "(R/S)";
INPUT C$
IF C$ = "R" OR C$ = "r" THEN
  PRINT : PRINT "INGRESE ORDEN DEL SISTEMA REDUCIDO; (NR<="; N; "<=0) NR = ";
  INPUT NR
  IF NR > N OR NR <= 0 THEN GOTO 5
ELSEIF C$ <> "S" AND C$ <> "s" THEN
  GOTO 5
ELSE
    REDIM UM(N - 1), VM(N - 1)
    FOR I = 1 TO N - 1
    UM(I) = SQR(N - I) / ABS(RD1(I + 1)) * (2 * SQR(N - I) + 1)
    NEXT I
    FOR I = 2 TO N - 1
```

```
        VM(I) = UM(I - 1) / UM(I)
        NEXT I
        INDEX = 2
        VMAY = VM(2)
        I = 3
        WHILE I <= N - 1
          IF VMAY > VM(I) THEN
          ELSE
            VMAY = VM(I)
            INDEX = I
          END IF
          I = I + 1
        WEND
        NR = INDEX
    END IF
    PRINT
    PRINT #1,
    PRINT #1, "Grado del sistema reducido: "; NR
    PRINT #1,
    PRINT "Presione cualquier tecla para continuar"
67   IF INKEY$ = "" THEN 67

'
'   Comienza el Metodo de DAVISON
'

    DIM Ar(NR, NR), BR(NR, 1), BR1(N, 1), DET(NR + 1)
    DIM ARI(NR, NR, NR), ER(NR), GRN(NR)
'
'   Calculo de Ar
'
    FOR I = 1 TO NR
    FOR J = 1 TO NR
    Ar(I, J) = RJ(I, J)
    NEXT J, I

'
'   Calculo de Br
'
    N1 = N
    M1 = 1

    CALL MULTM(N, N1, M1, BR1(), VI(), B())

    FOR I = 1 TO NR
    BR(I, 1) = BR1(I, 1)
    NEXT I

'
'   Calculo de funcion de transferencia metodo de DAVISON
'
'   Calculo del denominador
'

    DIM D(NR)
    FOR I = 1 TO NR
    D(I) = -Ar(I, I)
    NEXT I

    CALL MBIN(NR, D(), DET())

    FOR I = 1 TO NR
```

```
  FOR II = 1 TO NR
  D(II) = 0
  NEXT II
J1 = 1
FOR J = 1 TO NR
IF J = I THEN
ELSE
  D(J1) = -Ar(J, J)
  J1 = J1 + 1
END IF
NEXT J

CALL MBIN(NR - 1, D(), ER())

FOR J = 1 TO NR
ER(J) = ER(J) * BR(I, 1)
ARI(I, I, J) = ER(J)
NEXT J

NEXT I



' Calculo del Numerador
'

FOR J = 1 TO NR
FOR I = 1 TO NR
GRN(J) = GRN(J) + ARI(I, I, J)
NEXT I, J

'
' Comienza metodo de CHIDAMBARA
'
' El denominador del metodo de Davison se conserva
' Solo se calcula el numerador en este metodo
'
MR = N - NR
DIM XR(MR), XD(NR + 1), GN2(NR + 1)
FOR I = 1 TO MR

XR(I) = -BR1(NR + I, 1) / RJ(NR + I, NR + I)
NEXT I

FOR I = 1 TO MR
FOR J = 1 TO NR + 1
XD(J) = XD(J) + DET(J) * XR(I)
NEXT J
NEXT I
FOR I = 1 TO NR
GN2(I + 1) = XD(I + 1) + GRN(I)
NEXT I
GN2(1) = XD(1)


CLS
PRINT #1, " FUNCION DE TRANSFERENCIA RESULTANTE - METODO DE DAVISON"
PRINT #1, : PRINT #1,
PRINT #1, "NUMERADOR GRADO: "; NR - 1

FOR I = 1 TO NR
PRINT #1, "Coef. grado"; NR - I; "= ";
```

```
    PRINT #1, USING "###.#####"; GRN(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "DENOMINADOR GRADO:"; NR
    FOR I = 1 TO NR + 1
    PRINT #1, "Coef. grado"; NR + 1 - I; "= ";
    PRINT #1, USING "###.#####"; DET(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT "Presione cualquier tecla para continuar y ver los resultados"
    PRINT "por el metodo de Chidambara"
77   IF INKEY$ = "" THEN 77


    CLS
    PRINT #1, " FUNCION DE TRANSFERENCIA RESULTANTE - METODO DE CHIDAMBARA"
    PRINT #1, : PRINT #1,
    PRINT #1, "NUMERADOR GRADO: "; NR

    FOR I = 1 TO NR + 1
    PRINT #1, "Coef. grado"; NR + 1 - I; "= ";
    PRINT #1, USING "#######.#####"; GN2(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "DENOMINADOR GRADO:"; NR
    FOR I = 1 TO NR + 1
    PRINT #1, "Coef. grado"; NR + 1 - I; "= ";
    PRINT #1, USING "#######.#####"; DET(I)
    NEXT I


FINAL:
    PRINT : PRINT : PRINT "Presione cualquier tecla para continuar"
87   IF INKEY$ = "" THEN 87
    CHAIN "MENU2F"
    END
```

```
DECLARE SUB DETER (ND%, AD#(), DETM#)
DECLARE SUB MPINV (N%, M%, A#(), AI#())
DECLARE SUB MBIN (N1%, BE#(), PS#())
DECLARE SUB ROOT (NG%, A#(), R1#(), RI1#())
DECLARE SUB MINV (N%, AI#(), CI#(), F1#)
DECLARE SUB MULTM (N%, N1%, M1%, VR#(), VM1#(), VM2#())
'
'    METODO DE AGREGACION
'
'    B1.- Matriz con funcion de transferencia original
'    B.- Matriz del sistema
'    G.- Matriz sistema reducido
'    F.- Matriz sistema reducido
'    H.- Matriz de salida del sistema reducido
'    A.- Matriz del sistema en forma canonica observable
'    C.- Matriz del sistema
'    AG.- Matriz de Agregacion (no es una matriz cuadrada)
'    AGI.- Matriz de agregacion inversa
'    V.- Matriz modal del sistema reducido (con vectores propios)
'    VM.- Matriz modal normalizada
'    B5, C5, P.- Matrices y arreglo para calculo de valores y vectores
'            propios
'    RD1.- Arreglo con parte real de valores propios
'    RDI1.- Arreglo con parte imaginaria de vectores propios
'    PDEN.- Polinomio con el denominador del sistema reducido
'    PNUM.- Polinomio con el numerador del sistema reducidi
'    SIA1,SIA2,SII2.- Matrices de ayuda para el calculo de la funcion
'              de transferencia del sistema reducido
'    N1.- Numero de filas de la matriz a multiplicar (usar antes de CALL MULTM)
'    M1.- Numero de columnas de la matriz a multiplicar (usar antes de CALL MULTM)

1    CLEAR
     CLS
     DEFINT I-N
     DEFDBL A-H, O-Z
     PRINT "METODO DE AGREGACION"
     PRINT STRING$(20, "=")

9    PRINT : PRINT
     PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
     PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
     PRINT TAB(20); "IMPRESION PANTALLA (P)"
     PRINT : PRINT TAB(20); "Ingrese I, D o P";
     INPUT X$
     IMPR$ = "LPT1:"
     IF X$ = "I" OR X$ = "i" THEN
     GOTO 10
     ELSEIF X$ = "D" OR X$ = "d" THEN
     GOTO 11
     ELSEIF X$ = "P" OR X$ = "p" THEN
     GOTO 12
     ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
     GOTO 9
     END IF
12   IMPR$ = "SCRN:"
     GOTO 10
11   CLS
     PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
     INPUT AD$
     IMPR$ = AD$
10   OPEN IMPR$ FOR OUTPUT AS #1
```

```
3    CLS
     INPUT "GRADO DENOMINADOR; (N >= 3) N = "; N
     IF N <= 2 THEN 3
     PRINT
     INPUT "GRADO NUMERADOR;(M no puede ser mayor o igual que N) M = "; M
     IF M >= N OR M < 0 THEN 3
     REDIM B1(2, N + 1), A(N, N), B5(N, N), C5(N, N), P(N), RD1(N), RDI1(N)
     REDIM B(N, 1), C(1, N)
     CLS
     PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
     PRINT
     FOR I = 1 TO N + 1
     PRINT "COEFICIENTE GRADO"; N + 1 - I; " = ";
     INPUT B1(1, I)
     NEXT I
     PRINT
     PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
     PRINT
     FOR I = 1 TO M + 1
     PRINT "COEFICIENTE GRADO"; M + 1 - I; " = ";
     INPUT B1(2, N - M + I)
     NEXT I
     PRINT : PRINT
2    PRINT "ESTA CORRECTO (S/N)";
     INPUT P$
     IF P$ = "N" OR P$ = "n" THEN 3
     IF P$ <> "S" AND P$ <> "s" THEN 2
'
'    Calculo matrices forma canonica obserbable
'    Matriz A
'
     FOR I = 1 TO N
     FOR J = 1 TO N
     A(I, J) = 0
     NEXT J, I
     FOR I = 2 TO N
     A(I, I - 1) = 1
     NEXT I
     FOR I = 1 TO N
     A(I, N) = -B1(1, N + 2 - I)
     NEXT I

'
'    Calculo matriz B
'
     FOR I = 1 TO N
     B(I, 1) = 0
     NEXT I
     FOR I = 1 TO M + 1
     B(I, 1) = B1(2, N + 2 - I) - B1(2, 1) * B1(1, N + 2 - I)
     NEXT I

'
'    Calculo matriz C
'
     FOR I = 1 TO N - 1
     C(1, I) = 0
     NEXT I
     C(1, N) = 1

COMIENZO:
```

```
'
'    Comienza calculo de valores propios
'

     FOR I = 1 TO N
     FOR J = 1 TO N
     B5(I, J) = A(I, J)
     NEXT J, I
     FOR K = 1 TO N - 1
     T = 0
     FOR I = 1 TO N
     T = T + B5(I, I)
     NEXT I
     P(K) = T / K
     FOR I = 1 TO N
     B5(I, I) = B5(I, I) - P(K)
     NEXT I
     FOR I = 1 TO N
     FOR J = 1 TO N
     C5(I, J) = B5(I, J)
     NEXT J, I
     FOR I = 1 TO N
     FOR J = 1 TO N
     B5(I, J) = 0
     FOR K1 = 1 TO N
     B5(I, J) = B5(I, J) + A(I, K1) * C5(K1, J)
     NEXT K1, J, I, K
     P(N) = B5(1, 1)
     IF ABS(P(N)) < 1E-08 THEN 1000
     FOR I = 1 TO N
     FOR J = 1 TO N
     C5(I, J) = C5(I, J) / P(N)
     NEXT J, I
1000 REM   ECUAPOL
     REM
     REM SOLUCION DE ECUACIONES POLINOMIALES
     REM
     REM CALCULO DE LAS RAICES DE LA ECUACION CARACTERISTICA
     REM DE LA MATRIZ A
     REM
     DIM P1(31)
     P1(1) = 1
     FOR J = 2 TO N + 1
     P1(J) = -P(J - 1)
     NEXT J

     CALL ROOT(N, P1(), RD1(), RDI1())

'
'    Ordenamiento de los valores propios de menor a mayor (en valor absoluto)
'

     I = 1
     WHILE I <= N - 1
       AMIN = ABS(RD1(I))
       J = I + 1
       IF RD1(I) = RD1(J) THEN
         I = I + 2
       ELSE
         WHILE J <= N
           IF ABS(RD1(J)) < AMIN THEN
```

```
            AMIN = RD1(I)
            AIMG = RDI1(I)
            RD1(I) = RD1(J)
            RDI1(I) = RDI1(J)
            RD1(J) = AMIN
            RDI1(J) = AIMG
            AMIN = ABS(RD1(I))
            J = J + 1
          ELSE
            J = J + 1
          END IF
        WEND
      I = I + 1
      END IF
    WEND
    IF RDI1(N) <> 0 THEN
      CLS
      PRINT "ESTE PROGRAMA NO RESUELVE SISTEMAS CON VALORES PROPIOS"
      PRINT "COMPLEJOS CONJUGADOS, Y ESTE SISTEMA LOS TIENE"
      PRINT
      PRINT "POR FAVOR INTRODUZCA UN SISTEMA CON POLOS REALES"
      GOTO FINAL
    ELSE
    END IF

'
'   Calculo de vectores propios
'
    DIM V(N, N)
    REDIM B5(N, N)

    K = 1
    WHILE K <= N
      CLS
      LOCATE 9, 16
      PRINT "CALCULANDO VECTOR PROPIO No."; K; "POR FAVOR ESPERE"

      FOR I = 1 TO N
      FOR J = 1 TO N
      B5(I, J) = -A(I, J)
      NEXT J, I
      FOR I = 1 TO N
      B5(I, I) = B5(I, I) + RD1(K)
      NEXT I
      V(N, K) = 1
      FOR I = 1 TO N - 1
      TEMPO = 0
      FOR J = 1 TO I
      TEMPO = TEMPO - B5(N - I + 1, N - J + 1) * V(N - J + 1, K)
      NEXT J
      V(N - I, K) = TEMPO / B5(N - I + 1, N - I)
      NEXT I
      K = K + 1
    WEND

'
'   Criterio de seleccion del orden del sistema
'

5   CLS
    PRINT "DESEA INGRESAR EL ORDEN DEL MODELO REDUCIDO (Presione R)"
    PRINT TAB(28); "O"
```

```
PRINT "DESEA ENTRAR AL CRITERIO DE SELECCION DEL ORDEN (Presione S)"
PRINT : PRINT "NOTA: El modelo reducido debe ser de orden menor o igual que el original"
PRINT : PRINT "(R/S)";
INPUT C$
IF C$ = "R" OR C$ = "r" THEN
  PRINT : PRINT "INGRESE ORDEN DEL SISTEMA REDUCIDO; (1<=NR<="; N; ") NR = ";
  INPUT NR
  IF NR >= N OR NR < 1 THEN GOTO 5
ELSEIF C$ <> "S" AND C$ <> "s" THEN
  GOTO 5
ELSE
    REDIM UM(N - 1), VM(N - 1)
    FOR I = 1 TO N - 1
    UM(I) = SQR(N - I) / ABS(RD1(I + 1)) * (2 * SQR(N - I) + 1)
    NEXT I
    FOR I = 2 TO N - 1
    VM(I) = UM(I - 1) / UM(I)
    NEXT I
    INDEX = 2
    VMAY = VM(2)
    I = 3
    WHILE I <= N - 1
      IF VMAY > VM(I) THEN
      ELSE
       VMAY = VM(I)
       INDEX = I
      END IF
      I = I + 1
    WEND
    NR = INDEX
END IF
PRINT
PRINT #1,
PRINT #1, "El orden del sistema reducido es: "; NR
PRINT #1,
IF NR > 2 THEN
   CLS
   PRINT "UN SISTEMA DE ORDEN "; NR; " NO ES UN SISTEMA REDUCIDO"
   PRINT
   PRINT "POR FAVOR, ESCOJA QUE EL SISTEMA REDUCIDO SEA DE ORDEN 2 O 1"
   GOTO 5
ELSEIF NR < 1 THEN
   CLS
   PRINT "UN SISTEMA DE ORDEN CERO (0) NO ES UN BUEN MODELO PARA SISTEMA REDUCIDO"
   PRINT
   PRINT "POR FAVOR, ESGOJA QUE EL SISTEMA REDUCIDO SEA DE ORDEN 2 O 1"
   GOTO 5
ELSE
END IF


PRINT "Presione cualquier tecla para continuar"
6  IF INKEY$ = "" THEN 6


'
'   Comienza el metodo de Agregacion
'

'
'   Calculo de la matriz de agregacion
```

```
'
      DIM AG(NR, N), VN(N, N)
'
'     Calculo de la matriz modal en forma normal
'
      FOR J = 1 TO N
      VMOD = 0
      VAUX = 0
      FOR I = 1 TO N
      VAUX = VAUX + V(I, J) ^ 2
      NEXT I
      VMOD = SQR(VAUX)
      FOR I = 1 TO N
      VN(I, J) = V(I, J) / VMOD
      NEXT I
      NEXT J


      FOR I = 1 TO NR
      FOR J = 1 TO N
      AG(I, J) = VN(I, J)
      NEXT J, I

'
'     Calculo de la matriz inversa de la matriz de agregacion
'
      DIM AGI(N, NR)
      N1 = NR
      M1 = N
      CALL MPINV(N1, M1, AG(), AGI())


'
'     Calculo de la matriz F
'
      DIM F(NR, NR), FAUX(NR, N)
      N1 = NR
      M1 = N
      CALL MULTM(N, N1, M1, FAUX(), AG(), A())
      N1 = NR
      M1 = NR
      CALL MULTM(N, N1, M1, F(), FAUX(), AGI())

105

'
'     Calculo de la matriz G
'
      DIM G(NR, 1)
      N1 = NR
      M1 = 1
      CALL MULTM(N, N1, M1, G(), AG(), B())

106
'
'     Calculo de la matriz H
'
      DIM H(1, NR)
      N1 = 1
      M1 = NR
      CALL MULTM(N, N1, M1, H(), C(), AGI())
```

```
'
'   Calculo de la funcion de transferencia reducida

    IF NR = 2 THEN

        DIM SIA1(NR, NR), SIA2(NR, NR), PDEN(NR + 1), PNUM(NR), DIAG(NR)
        FOR I = 1 TO NR
        SIA1(I, I) = 1
        NEXT I
        FOR I = 1 TO NR
        FOR J = 1 TO NR
        SIA2(I, J) = -F(I, J)
        NEXT J, I

        TEMPO = SIA2(2, 2)
        SIA2(2, 2) = SIA2(1, 1)
        SIA2(1, 1) = TEMPO
        SIA2(1, 2) = -SIA2(1, 2)
        SIA2(2, 1) = -SIA2(2, 1)


        FOR I = 1 TO NR
        DIAG(I) = SIA2(I, I)
        NEXT I

        CALL MBIN(NR, DIAG(), PDEN())
        PDEN(3) = PDEN(3) - SIA2(1, 2) * SIA2(2, 1)
        REDIM R1(NR, 1), R2(NR, 1), AUX(1, 1)

        N1 = NR
        M1 = 1
        CALL MULTM(NR, N1, M1, R1(), SIA1(), G())
        CALL MULTM(NR, N1, M1, R2(), SIA2(), G())
        N1 = 1
        M1 = 1
        CALL MULTM(NR, N1, M1, AUX(), H(), R1())
        PNUM(1) = AUX(1, 1)
        CALL MULTM(NR, N1, M1, AUX(), H(), R2())
        PNUM(2) = AUX(1, 1)

        IF XS = "T" OR XS = "D" THEN
        PRINT #1, : PRINT #1,
        ELSE
          CLS
        END IF

        PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
        PRINT #1,
        PRINT #1, "GRADO NUMERADOR: "; NR - 1
        FOR I = 1 TO NR
        PRINT #1, "COEFICIENTE GRADO"; NR - I; " = ";
        PRINT #1, USING "####.######"; PNUM(I)
        NEXT I
        PRINT #1, : PRINT #1,
        PRINT #1, "GRADO DENOMINADOR:"; NR
        FOR I = 1 TO NR + 1
        PRINT #1, "COEFICIENTE GRADO"; NR + 1 - I; " = ";
        PRINT #1, USING "####.#####"; PDEN(I)
        NEXT I
```

```
    ELSE

        IF F <= 0 THEN
          CLS
          PRINT "EL SISTEMA OBTENIDO ES INESTABLE"
          PRINT "PRUEBE REDUCIENDOLO A  UN SISTEMA DE SEGUNDO ORDEN"
          GOTO FINAL
        ELSE
        END IF
        A = F
        B = G

        IF X$ = "T" OR X$ = "D" THEN
        PRINT #1, : PRINT #1,
        ELSE
         CLS
        END IF
        PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
        PRINT #1,
        PRINT #1, "GRADO NUMERADOR: "; NR - 1
        PRINT #1, "COEFICIENTE GRADO"; NR - 1; " = ";
        PRINT #1, USING "#######.#####"; B
        PRINT #1, : PRINT #1,
        PRINT #1, "GRADO DENOMINADOR:"; NR
        PRINT #1, "COEFICIENTE GRADO 1 =  1.00000"
        PRINT #1, "COEFICIENTE GRADO 0 =";
        PRINT #1, USING "#######.#####"; -A

    END IF

FINAL:
    PRINT
    PRINT "Presione cualquier tecla para continuar"
101   IF INKEY$ = "" THEN 101
    CHAIN "MENU2F"
    END
```

```
DECLARE SUB MINV (N%, AI#(), CI#(), F1#)
DECLARE SUB MULTM (N%, N1%, M1%, Vr#(), VM1#(), VM2#())
'
'     Metodo de la realizacion parcial
'
'     A,B,C.-  Matrices del sistema
'     SA = CAUX * B
'     SA = C*A^-i*B (en general)
'     CAUX = C * AI
'     AUX1 = AUX * A (luego de multiplicar se hace AUX = AUX1)
'     AUX1 = A^i (en general)
'     AI = inversa de AUX1
'     S = Matriz de Hankel
'     NR1 = Orden real del sistema reducido (NR1 = NR y NR = NR+1)
'     Q.-  Matriz de Hankel sin modificar por la realizacion
'     EI.-  Vector unidad para hacer cero el i-esimo elemento
'     EIt.-  Vector EI transpuesto
'     E.-  Vector unidad de acuerdo a la columna que se este reduciendo
'     EAUX.-  Vector auxiliar ; EAUX = S*E
'     EAUX1.-  Vector auxiliar;  EAUX1 = S*E - EI
'     EAUX2.-  Vector auxiliar;  EAUX2 = EI*S
'     SAUX.-  Matriz auxiliar;  SAUX = EAUX1*EAUX2
'     IX.- Vector indice que nos indica que fila se redujo y no hay que
'        tomarla en cuenta en la proxima reduccion.
'     IX1.-  Vector IX ordenado de menor a mayor
'     SM.- Matriz de Hankel reducida y reordenada a forma Hermitica
'     IROW.-  Vector auxiliar para el calculo de SM
'     GDEN.-  Grado denominador
'     GNUM.-  Grado numerador
'     PDEN.-  Polinomio denominador
'     PNUM.-  Polinomio numerador
'
1     CLEAR
      CLS
      DEFINT I-N
      DEFDBL A-H, O-Z

      CLS
      PRINT "METODO DE REALIZACION PARCIAL"
      PRINT STRING$(29, "=")
      PRINT : PRINT

9     PRINT : PRINT
      PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
      PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
      PRINT TAB(20); "IMPRESION PANTALLA (P)"
      PRINT : PRINT TAB(20); "Ingrese I, D o P";
      INPUT X$
      IMPR$ = "LPT1:"
      IF X$ = "I" OR X$ = "i" THEN
      GOTO 10
      ELSEIF X$ = "D" OR X$ = "d" THEN
      GOTO 11
      ELSEIF X$ = "P" OR X$ = "p" THEN
      GOTO 12
      ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
      GOTO 9
      END IF
12    IMPR$ = "SCRN:"
      GOTO 10
11    CLS
      PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
```

```
    INPUT AD$
    IMPR$ = AD$
10   OPEN IMPR$ FOR OUTPUT AS #1

3    CLS
    PRINT "INGRESE LA FUNCION DE TRANSFERENCIA ORIGINAL"
    PRINT
    PRINT
    INPUT "GRADO DENOMINADOR; (N>1) N = "; N
    IF N <= 1 THEN 3
    PRINT
    INPUT "GRADO NUMERADOR;(M no puede ser mayor que N, ni menor que 0) M = "; M
    IF M > N OR M < 0 THEN 3
    REDIM A(N, N), B(N, 1), C(1, N), B1(2, N + 1)

    CLS
    PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO N + 1
    PRINT "COEFICIENTE GRADO"; N + 1 - I; " = ";
    INPUT B1(1, I)
    NEXT I
    PRINT
    PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO M + 1
    PRINT "COEFICIENTE GRADO"; M + 1 - I; " = ";
    INPUT B1(2, N - M + I)
    NEXT I
    PRINT : PRINT
2    PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN 3
    IF P$ <> "S" AND P$ <> "s" THEN 2

    CLS
    PRINT #1, " FUNCION DE TRANSFERENCIA ORIGINAL "
    PRINT #1, : PRINT #1,
    PRINT #1, "NUMERADOR GRADO: "; M

    FOR I = 1 TO M + 1
    PRINT #1, "Coef. grado"; M + 1 - I; "= ";
    PRINT #1, USING "#######.#####"; B1(2, N - M + I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "DENOMINADOR GRADO:"; N
    FOR I = 1 TO N + 1
    PRINT #1, "Coef. grado"; N + 1 - I; "= ";
    PRINT #1, USING "######.#####"; B1(1, I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT "Presione cualquier tecla para continuar"
17   IF INKEY$ = "" THEN 17

'
'    Calculo matrices forma canonica obserbable
'    Matriz A
'

    FOR I = 1 TO N
    FOR J = 1 TO N
    A(I, J) = 0
```

```
    NEXT J, I
    FOR I = 2 TO N
    A(I, I - 1) = 1
    NEXT I
    FOR I = 1 TO N
    A(I, N) = -B1(1, N + 2 - I)
    NEXT I

'
'    Calculo matriz B
'
    FOR I = 1 TO N
    B(I, 1) = 0
    NEXT I
    FOR I = 1 TO M + 1
    B(I, 1) = B1(2, N + 2 - I) - B1(2, 1) * B1(1, N + 2 - I)
    NEXT I

'
'    Calculo matriz C
'
    FOR I = 1 TO N - 1
    C(1, I) = 0
    NEXT I
    C(1, N) = 1

COMIENZO:


'    Empieza construccion de la matriz de Hankel
'
40   CLS
    PRINT "Ingrese el orden del sistema reducido"
    PRINT "(NR no puede ser mayor que"; N; " ni menor que"; N - M; "para este modelo)"
    PRINT " NR = ";
    INPUT NR
    IF NR > N OR NR < N - M THEN 40
    NR1 = NR

    IF X$ = "I" OR X$ = "D" THEN
     PRINT #1, : PRINT #1, "Orden del sistema reducido: "; NR
    ELSE

    END IF


    REDIM S(NR + 1, NR + 1), SA(1, 1), AI(N, N), CAUX(1, N)
    REDIM AUX(N, N), AUX1(N, N), Q(NR + 1, NR + 1)
    N1 = 1
    M1 = 1
    CALL MULTM(N, N1, M1, SA(), C(), B())
    S(1, 2) = SA(1, 1)
    S(2, 1) = SA(1, 1)
    CALL MINV(N, A(), AI(), F)
    IF F = 1 THEN GOTO FINAL
    N1 = 1: M1 = N
    CALL MULTM(N, N1, M1, CAUX(), C(), AI())
    N1 = 1: M1 = 1
    CALL MULTM(N, N1, M1, SA(), CAUX(), B())
    S(1, 1) = SA(1, 1)
```

```
FOR J = 1 TO N
AUX(J, J) = 1
NEXT J

I = 2
WHILE I <= 2 * NR
  N1 = N: M1 = N
  CALL MULTM(N, N1, M1, AUX1(), AUX(), A())

  FOR J = 1 TO N
  FOR K = 1 TO N
  AUX(J, K) = AUX1(J, K)
  NEXT K, J

  N1 = 1: M1 = N
  CALL MULTM(N, N1, M1, CAUX(), C(), AUX())
  N1 = 1: M1 = 1
  CALL MULTM(N, N1, M1, SA(), CAUX(), B())

  IF I < NR THEN
   FOR J = 1 TO I + 1
   S(I + 2 - J, J) = SA(1, 1)
   NEXT J
  ELSEIF I = NR THEN
   FOR J = 1 TO NR + 1
   S(NR - J + 2, J) = SA(1, 1)
   NEXT J
  ELSE
   FOR J = I - NR + 1 TO NR + 1
   S(I + 2 - J, J) = SA(1, 1)
   NEXT J
  END IF
  I = I + 1
WEND


FOR I = 1 TO NR + 1
FOR J = 1 TO NR + 1
Q(I, J) = S(I, J)
NEXT J, I

'
'   Comienza la realizacion
'

NR = NR + 1

REDIM EI(NR, 1), EIt(1, NR), E(NR, 1)
REDIM EAUX1(NR, 1), EAUX2(1, NR), SAUX(NR, NR)
REDIM IX(NR), IX1(NR)


I = 1
WHILE I <= NR - 1
  IX(I) = I
  J = 1
  WHILE J <= NR
    IF S(J, I) = 0 THEN
      IX(I) = J
      IF J = NR THEN
        CLS
        PRINT "LA MATRIZ DE HANKEL ES DE RANGO MENOR A"; NR
```

```
      PRINT "SE NECESITA UNA MATRIZ DE HANKEL DE RANGO"; NR + 1
      PRINT
      PRINT "ESCOJA OTRO SISTEMA"
      END
    ELSE
    END IF
  ELSE
    IF I = 1 THEN
      IX(I) = J
      IX1(I) = J
      J = NR + 1
    ELSE
      FOR K = 1 TO I - 2
      MIN = IX1(K)
      FOR K1 = K + 1 TO I - 1
      IF IX1(K1) > MIN THEN
      ELSE
        IX1(K) = IX1(K1)
        IX1(K1) = MIN
        MIN = IX1(K)
      END IF
      NEXT K1, K
      FOR K = 1 TO I - 1
      IF IX1(K) = J THEN
        J = J + 1
      ELSE
      END IF
      NEXT K
      IF S(J, I) = 0 THEN
      ELSE
        IX(I) = J
        IX1(I) = J
        J = NR + 1
      END IF
    END IF
  END IF
  J = J + 1
WEND
FOR J = 1 TO NR
EI(J, 1) = 0
EIt(1, J) = 0
E(J, 1) = 0
NEXT J
EI(IX(I), 1) = 1
EIt(1, IX(I)) = 1
E(I, 1) = 1

N1 = NR
M1 = 1
CALL MULTM(NR, N1, M1, EAUX1(), S(), E())

FOR J = 1 TO NR
EAUX1(J, 1) = EAUX1(J, 1) - EI(J, 1)
NEXT J

N1 = 1
M1 = NR
CALL MULTM(NR, N1, M1, EAUX2(), EIt(), S())

N1 = NR
M1 = NR
CALL MULTM(1, N1, M1, SAUX(), EAUX1(), EAUX2())
```

```
  DIV = S(IX(I), I)
  FOR J = 1 TO NR
  FOR K = 1 TO NR
  SAUX(J, K) = SAUX(J, K) / DIV
  NEXT K, J
  FOR J = 1 TO NR
  FOR K = 1 TO NR
  S(J, K) = S(J, K) - SAUX(J, K)
  NEXT K, J
  I = I + 1
WEND

'
'   Ordenamiento de la matriz de Hankel reducida
'   para la obtencion de la matriz F
'
REDIM SM(NR, NR), IROW(NR - 1)
I = 1
WHILE I <= NR - 1
 FOR J = 1 TO NR
 IF S(J, I) > .98 AND S(J, I) <= 1 THEN
  IROW(I) = J
 ELSE
 END IF
 NEXT J
 I = I + 1
WEND

FOR I = 1 TO NR - 1
FOR J = 1 TO NR
SM(I, J) = S(IROW(I), J)
NEXT J, I
'
'   Obtencion de la funcion de transferencia reducida
'


'
'   Calculo denominador
'
NR = NR - 1
DIM PDEM(NR + 1)
FOR I = 2 TO NR + 1
PDEM(I) = -SM(NR + 2 - I, NR + 1)
NEXT I
PDEM(1) = 1
'
'   Calculo numerador
'
REDIM FAUX(NR, NR)
FOR I = 1 TO NR
FOR J = 1 TO NR
FAUX(I, J) = 0
NEXT J, I
I = 1
WHILE I <= NR - 1
 FAUX(I, I) = 1
 J1 = 1
 FOR J = I + 1 TO NR
 FAUX(I, J) = -SM(NR + 1 - J1, NR + 1)
 J1 = J1 + 1
 NEXT J
```

```
    I = I + 1
   WEND
   FAUX(NR, NR) = 1

   IF M = 0 AND NR <> N THEN
    FOR I = 1 TO NR
    FOR J = 1 TO NR
    FAUX(I, J) = FAUX(I, J) * Q(2, I + 1)
    NEXT J, I
   ELSE
    FOR I = 1 TO NR
    FOR J = 1 TO NR
    FAUX(I, J) = FAUX(I, J) * Q(2, I)
    NEXT J, I
   END IF

   DIM PNUM(NR)
   FOR I = 1 TO NR
   FOR J = 1 TO NR
   PNUM(I) = PNUM(I) + FAUX(J, I)
   NEXT J, I
   GNUM = NR
   I = 1
   WHILE I <= NR
    IF PNUM(I) = 0 THEN
      GNUM = GNUM - 1
    ELSE
      I = NR + 1
    END IF
    I = I + 1
   WEND

   IF X$ = "I" OR X$ = "D" THEN
    PRINT #1, : PRINT #1,
   ELSE
    CLS
   END IF

   PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
   PRINT #1, : PRINT #1,
   PRINT #1, "GRADO NUMERADOR :"; GNUM
   PRINT #1,
   FOR I = NR - GNUM + 1 TO NR
   PRINT #1, "COEFICIENTE GRADO "; NR - I; " =";
   PRINT #1, USING "######.####"; PNUM(I)
   NEXT I
   PRINT #1, : PRINT #1,
   GDEM = NR
   PRINT #1, "GRADO DENOMINADOR :"; GDEM
   PRINT #1,
   FOR I = 1 TO NR + 1
   PRINT #1, "COEFICIENTE GRADO "; NR + 1 - I; " =";
   PRINT #1, USING "######.####"; PDEM(I)
   NEXT I

FINAL:
   PRINT
   PRINT
   PRINT "Presione cualquier tecla para continuar"
102  IF INKEY$ = "" THEN 102
   CHAIN "MENU2F"
   END
```

```
DECLARE SUB MBIN (N1%, BE#(), PS#())
DECLARE SUB MINV (N%, AI#(), CI#(), F1#)
DECLARE SUB QRAIZ (N%, T#, F#(), A#(), Z#)
DECLARE SUB MULTM (N%, N1%, M1%, Vr#(), VM1#(), VM2#())
DECLARE SUB ECUA (N%, A#(), B#(), X#())
DECLARE SUB RITD (N%, M%, NR%, T#, A#(), B#(), Q#(), QOR#(), QOI#(), QR11#(), QI11#(), QR12#(), QI12#(), W#())
DECLARE SUB EVALPC (N%, AB#(), Pr#, Pi#, Vr#, Vi#)
DECLARE SUB DIFP (N%, AB#(), ABD#())
DECLARE SUB ROOT (NG%, A#(), R1#(), RI1#())

'
'    Metodo de Aproximaciones Optimas
'
    CLEAR
    CLS
    DEFINT I-N
    DEFDBL A-H, O-Z
    PRINT "METODO DE APROXIMACIONES OPTIMAS"
    PRINT STRING$(32, "=")
    PRINT


9    PRINT : PRINT
    PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
    PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
    PRINT TAB(20); "IMPRESION PANTALLA (P)"
    PRINT : PRINT TAB(20); "Ingrese I, D o P";
    INPUT X$
    IMPR$ = "LPT1:"
    IF X$ = "I" OR X$ = "i" THEN
    GOTO 10
    ELSEIF X$ = "D" OR X$ = "d" THEN
    GOTO 11
    ELSEIF X$ = "P" OR X$ = "p" THEN
    GOTO 12
    ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
    GOTO 9
    END IF
12   IMPR$ = "SCRN:"
    GOTO 10
11   CLS
    PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
    INPUT AD$
    IMPR$ = AD$
10   OPEN IMPR$ FOR OUTPUT AS #1


3    CLS
    INPUT "GRADO DENOMINADOR; N = "; N
    IF N <= 0 THEN 3
    PRINT
    INPUT "GRADO NUMERADOR;(M no puede ser mayor que N) M = "; M
    IF M > N OR M < 0 THEN 3
    REDIM PN(M + 1), PD(N + 1)
    CLS
    PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO N + 1
    PRINT "COEFICIENTE GRADO"; N + 1 - I; " = ";
    INPUT PD(I)
    NEXT I
    PRINT
```

```
    PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO M + 1
    PRINT "COEFICIENTE GRADO"; M + 1 - I; " = ";
    INPUT PN(I)
    NEXT I
    PRINT : PRINT
2   PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN 3
    IF P$ <> "S" AND P$ <> "s" THEN 2

    CLS
    PRINT #1, " FUNCION DE TRANSFERENCIA ORIGINAL "
    PRINT #1, : PRINT #1,
    PRINT #1, "NUMERADOR GRADO: "; M

    FOR I = 1 TO M + 1
    PRINT #1, "Coef. grado"; M + 1 - I; "= ";
    PRINT #1, USING "#######.###"; PN(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "DENOMINADOR GRADO:"; N
    FOR I = 1 TO N + 1
    PRINT #1, "Coef. grado"; N + 1 - I; "= ";
    PRINT #1, USING "#######.###"; PD(I)
    NEXT I
    PRINT : PRINT
    PRINT "Presione cualquier tecla para continuar"
30  IF INKEY$ = "" THEN 30


COMIENZO:

    REDIM Q(N), QOR(N), QOI(N), QR11(N), QI11(N), QR12(N), QI12(N)
    GAIN = PN(M + 1) / PD(N + 1)


21  CLS
    PRINT "INGRESE EL GRADO DEL MODELO REDUCIDO (NR = 2 O NR = 1)";
    INPUT NR
    IF NR > 2 OR NR < 1 THEN 21
    IF NR >= N THEN
     CLS
     PRINT "EL SISTEMA QUE INGRESA ES UN SISTEMA REDUCIDO"
     PRINT "NO HACE FALTA APLICAR UN METODO DE REDUCCION"
     GOTO FINAL
    ELSE
    END IF
    PRINT
23  PRINT
    PRINT "INGRESE EL PERIODO DE MUESTREO"
    PRINT "El periodo de muestreo no puede ser 0 o menor que 0"
    INPUT "T = "; T
    IF T <= 0 THEN 23
    PRINT
20  PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN GOTO COMIENZO
    IF P$ <> "S" AND P$ <> "s" THEN 20
```

```
    IF X$ = "T" OR X$ = "D" THEN
     PRINT #1, : PRINT #1,
    ELSE
     CLS
    END IF

    PRINT #1, "PERIODO DE MUESTREO = "; T
    PRINT #1,
    PRINT
    PRINT "Presione cualquier tecla para continuar"
31   IF INKEY$ = "" THEN 31
    TT = T
    DIM W(2 * NR + 1)
    CALL RITD(N, M, NR, T, PN(), PD(), Q(), QOR(), QOI(), QR11(), QI11(), QR12(), QI12(), W())

'
'   Comienza obtencion del sistema reducido en el plano Z
'
    REDIM WM(NR, NR), C(NR), WS(NR), D(NR + 1, 1), CM(NR + 1, NR + 1), WL(NR + 1, 1)

    FOR I = 1 TO NR
    K = I + 1
    FOR J = 1 TO NR
    WM(I, J) = W(K)
    K = K + 1
    NEXT J, I

    FOR I = 1 TO NR
    WS(I) = -W(I + 1 + NR)
    NEXT I

    CALL ECUA(NR, WM(), WS(), C())

    FOR I = 1 TO NR + 1
    FOR J = 1 TO NR + 1
    CM(I, J) = 0
    NEXT J, I
    FOR I = 1 TO NR + 1
    CM(I, I) = 1
    NEXT I
    FOR I = 1 TO NR
    K = NR
    FOR J = I + 1 TO NR + 1
    CM(J, I) = C(K)
    K = K - 1
    NEXT J, I

    FOR I = 1 TO NR + 1
    WL(I, 1) = W(I)
    NEXT I

    N1 = NR + 1
    M1 = 1
    CALL MULTM(NR + 1, N1, M1, D(), CM(), WL())

    IF X$ = "T" OR X$ = "D" THEN
     PRINT #1, : PRINT #1,
    ELSE
     CLS
    END IF
    PRINT #1, "FUNCION DE TRANSFERENCIA EN TIEMPO DISCRETO"
    PRINT #1,
```

```
    PRINT #1, "GRADO NUMERADOR: "; NR
    FOR I = 1 TO NR + 1
    PRINT #1, "COEFICIENTE GRADO"; NR + 1 - I; " = ";
    PRINT #1, USING "####.######"; D(I, 1)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "GRADO DENOMINADOR:"; NR
    PRINT #1, "COEFICIENTE GRADO"; NR; " =   1.00000"
    FOR I = 1 TO NR
    PRINT #1, "COEFICIENTE GRADO"; NR - I; " = ";
    PRINT #1, USING "####.#####"; C(NR + 1 - I)
    NEXT I
    PRINT : PRINT
    PRINT "Presione cualquier tecla para continuar"
32  IF INKEY$ = "" THEN 32

'
'   Comienza el calculo del sistema continuo
'
    IF NR = 2 THEN

        DIM F(NR, NR), G(NR, 1), H(1, NR), AAUX(NR), BAUX(NR), HAUX(NR)

        FOR I = 1 TO NR
        FOR J = 1 TO NR
        F(I, J) = 0
        NEXT J, I
        F(2, 1) = 1
        FOR I = 1 TO NR
        F(I, 2) = -C(I)
        NEXT I

        FOR I = 1 TO NR
        G(I, 1) = D(NR + 1 - I, 1)
        NEXT I

        FOR I = 1 TO NR
        H(1, I) = 0
        NEXT I
        H(1, NR) = 1

        DIM A(NR, NR), B(NR, 1), FI(NR, NR), FII(NR, NR)

50       T = TT
        CALL QRAIZ(NR, T, F(), A(), Z)

    IF Z = 1 THEN
51    CLS
      PRINT "LA TOLERANCIA INGRESADA NO ES CONVENIENTE"
      PRINT
      PRINT "PUEDE INGRESAR UN EPSILON MAYOR O FINALIZAR EL PROGRAMA"
      PRINT
      PRINT "ESCOJA : OTRO EPSILON (E) O FINALIZAR (F)";
      INPUT C$
      IF C$ = "E" OR C$ = "e" THEN 50
      IF C$ <> "F" AND C$ <> "f" THEN 51
      GOTO FINAL
    ELSE
    END IF

        FOR I = 1 TO NR
        FOR J = 1 TO NR
```

```
FI(I, J) = F(I, J)
NEXT J, I
FOR I = 1 TO NR
FI(I, I) = FI(I, I) - 1
NEXT I

CALL MINV(NR, FI(), FII(), Z)
IF Z = 1 THEN GOTO FINAL
REDIM AUX(NR, NR)

N1 = NR
M1 = NR
CALL MULTM(NR, N1, M1, AUX(), A(), FII())
N1 = NR
M1 = 1
CALL MULTM(NR, N1, M1, B(), AUX(), G())

'
' Calculo de la funcion de transferencia en el plano s
'
DIM SIA1(NR, NR), SIA2(NR, NR), PDEN(NR + 1), PNUM(NR), DIAG(NR)

FOR I = 1 TO NR
SIA1(I, I) = 1
NEXT I
FOR I = 1 TO NR
FOR J = 1 TO NR
SIA2(I, J) = -A(I, J)
NEXT J, I

TEMPO = SIA2(2, 2)
SIA2(2, 2) = SIA2(1, 1)
SIA2(1, 1) = TEMPO
SIA2(1, 2) = -SIA2(1, 2)
SIA2(2, 1) = -SIA2(2, 1)

FOR I = 1 TO NR
DIAG(I) = SIA2(I, I)
NEXT I

CALL MBIN(NR, DIAG(), PDEN())
PDEN(3) = PDEN(3) - SIA2(1, 2) * SIA2(2, 1)
REDIM R1(NR, 1), R2(NR, 1), AUX(1, 1)

N1 = NR
M1 = 1
CALL MULTM(NR, N1, M1, R1(), SIA1(), B())
CALL MULTM(NR, N1, M1, R2(), SIA2(), B())
N1 = 1
M1 = 1
CALL MULTM(NR, N1, M1, AUX(), H(), R1())
PNUM(1) = AUX(1, 1)
CALL MULTM(NR, N1, M1, AUX(), H(), R2())
PNUM(2) = AUX(1, 1)

GAINR = GAIN * PDEN(NR + 1) / PNUM(NR)
FOR I = 1 TO NR
PNUM(I) = PNUM(I) * GAINR
NEXT I

IF X$ = "T" OR X$ = "D" THEN
  PRINT #1, : PRINT #1,
```

```
        ELSE
         CLS
        END IF

        PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
        PRINT #1,
        PRINT #1, "GRADO NUMERADOR: "; NR - 1
        FOR I = 1 TO NR
        PRINT #1, "COEFICIENTE GRADO"; NR - I; " = ";
        PRINT #1, USING "####.######"; PNUM(I)
        NEXT I
        PRINT #1, : PRINT #1,
        PRINT #1, "GRADO DENOMINADOR:"; NR
        FOR I = 1 TO NR + 1
        PRINT #1, "COEFICIENTE GRADO"; NR + 1 - I; " = ";
        PRINT #1, USING "####.#####"; PDEN(I)
        NEXT I

    ELSE

        F = -C(NR)
        G = D(NR + 1, 1)
        H = 1

        IF F <= 0 THEN
          CLS
          PRINT "EL SISTEMA OBTENIDO ES INESTABLE"
          PRINT "PRUEBE REDUCIENDOLO A  UN SISTEMA DE SEGUNDO ORDEN"
          GOTO FINAL
        ELSE
        END IF
        A = LOG(F) / TT
        B = A * G / (F - 1)

        GAINR = GAIN * (-A) / B
        B = GAINR * B

        IF X$ = "T" OR X$ = "D" THEN
        PRINT #1, : PRINT #1,
        ELSE
         CLS
        END IF
        PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
        PRINT #1,
        PRINT #1, "GRADO NUMERADOR: "; NR - 1
        PRINT #1, "COEFICIENTE GRADO"; NR - 1; " = ";
        PRINT #1, USING "#######.#####"; B
        PRINT #1, : PRINT #1,
        PRINT #1, "GRADO DENOMINADOR:"; NR
        PRINT #1, "COEFICIENTE GRADO 1 =  1.00000"
        PRINT #1, "COEFICIENTE GRADO 0 =";
        PRINT #1, USING "#######.#####"; -A

    END IF


FINAL:
    PRINT
    PRINT "Presione cualquier tecla para continuar"
101   IF INKEY$ = "" THEN 101
    CHAIN "MENU3F"
    END
```

```
DECLARE SUB DETER (ND%, AD#(), DETM#)
DECLARE SUB MBIN (N1%, BE#(), PS#())
DECLARE SUB QRAIZ (N%, T#, F#(), A#(), Z#)
DECLARE SUB MULTM (N%, N1%, M1%, VR#(), VM1#(), VM2#())
DECLARE SUB MINV (N%, AI#(), CI#(), F1#)
DECLARE SUB RPTD (N%, M%, T#, NTP%, A#(), B#(), G#())
'
'    METODO DE LOS MINIMOS CUADRADOS
'
     CLEAR
     CLS
     DEFINT I-N
     DEFDBL A-H, O-Z
     PRINT "METODO DE LOS MINIMOS CUADRADOS"
     PRINT STRING$(31, "=")

9    PRINT : PRINT
     PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
     PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
     PRINT TAB(20); "IMPRESION PANTALLA (P)"
     PRINT : PRINT TAB(20); "Ingrese I, D o P";
     INPUT X$
     IMPR$ = "LPT1:"
     IF X$ = "I" OR X$ = "i" THEN
     GOTO 10
     ELSEIF X$ = "D" OR X$ = "d" THEN
     GOTO 11
     ELSEIF X$ = "P" OR X$ = "p" THEN
     GOTO 12
     ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
     GOTO 9
     END IF
12   IMPR$ = "SCRN:"
     GOTO 10
11   CLS
     PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
     INPUT AD$
     IMPR$ = AD$
10   OPEN IMPR$ FOR OUTPUT AS #1



3    CLS
     INPUT "GRADO DENOMINADOR; N = "; NG
     IF NG <= 0 THEN 3
     PRINT
     INPUT "GRADO NUMERADOR;(M no puede ser mayor que N) M = "; MG
     IF MG > NG OR MG < 0 THEN 3
     REDIM PN(MG + 1), PD(NG + 1)
     CLS
     PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
     PRINT
     FOR I = 1 TO NG + 1
     PRINT "COEFICIENTE GRADO"; NG + 1 - I; " = ";
     INPUT PD(I)
     NEXT I
     PRINT
     PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
     PRINT
     FOR I = 1 TO MG + 1
     PRINT "COEFICIENTE GRADO"; MG + 1 - I; " = ";
     INPUT PN(I)
```

```
    NEXT I
    PRINT : PRINT
2    PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN 3
    IF P$ <> "S" AND P$ <> "s" THEN 2

    CLS
    PRINT #1, " FUNCION DE TRANSFERENCIA ORIGINAL "
    PRINT #1, : PRINT #1,
    PRINT #1, "NUMERADOR GRADO: "; MG

    FOR I = 1 TO MG + 1
    PRINT #1, "Coef. grado"; MG + 1 - I; "= ";
    PRINT #1, USING "#######.###"; PN(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "DENOMINADOR GRADO:"; NG
    FOR I = 1 TO NG + 1
    PRINT #1, "Coef. grado"; NG + 1 - I; "= ";
    PRINT #1, USING "#######.###"; PD(I)
    NEXT I
    PRINT : PRINT
    PRINT "Presione cualquier tecla para continuar"
7    IF INKEY$ = "" THEN 7

COMIENZO:

    CLS
21    PRINT "INGRESE EL GRADO DEL MODELO REDUCIDO (NR = 2 O NR = 1)";
    INPUT NR
    IF NR > 2 OR NR < 1 THEN 21
    MR = NR
    IF NR >= NG THEN
     CLS
     PRINT "EL SISTEMA QUE INGRESA ES UN SISTEMA REDUCIDO"
     PRINT "NO HACE FALTA APLICAR UN METODO DE REDUCCION"
     GOTO FINAL
    ELSE
    END IF
    PRINT
23    PRINT
    INPUT "INGRESE EL PERIODO DE MUESTREO: T= "; T
    TMAX = 500 * T
    IF TMAX <= NR + MR + 1 THEN
      TMAX = TMAX + NR + MR + 2
    ELSE
    END IF
    PRINT
    PRINT "INGRESE EL TIEMPO DE MUESTREO"
    PRINT "El tiempo maximo que se puede ingresar es: "; TMAX
    PRINT "y el tiempo minimo es: "; NR + MR + 1
    INPUT "TM = "; TM
    IF TM > TMAX OR TM <= NR + MR + 1 THEN GOTO 23
    PRINT
20    PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN GOTO COMIENZO
    IF P$ <> "S" AND P$ <> "s" THEN 20

    NTP = TM / T
    TT = T
```

```
N = NR
M = MR
P = NR + MR + 1


REDIM AK(NTP, P), TETA(P, 1), Y(NTP + 1), UO(NTP + 1)
REDIM AK1(P, 1), PK1(P, P), TK1(P, 1), AUX(P, P)
REDIM AP(P, P), TP(P, 1), YP(P, 1), PP(P, P), AUX1(P, 1), AUX1T(1, P)
REDIM API(P, P), APT(P, P), AUX2(1, 1), AKT(1, P), AUX3(1, 1), AUX4(P, P)
'
'   AK.- Matriz con entrada y respuesta a senial paso
'   NTP.- Numero total de puntos muestreados (NTP>>P)
'   TETA.- Vector a obtenerse
'   Y.-  Vector con todos los datos muestreados
'   U.- Vector con entrada paso
'   AK1.- k-esima fila de matriz AK (para k>p)
'   PK1.- Nueva iteracion de matriz P
'   TK1.- Nueva iteracion de vector teta
'   AP.- Matriz para el calculo de la primera matriz P (PP) (extraida de AK)
'   TP.- Primer calculo de TETA TP = Ap^(-1)*Yp
'   YP.- p primeros valores de la senial de salida Y
'   PP.- Primera aproximacion de P (PP = [Ap(t)*Ap]^(-1))
'   AUX1 = Pk*Ak+1
'   API .- Ap inversa
'   APT.- Ap transpuesta
'   AKT.- AK1 transpuesta
'   AUX1T = AUX1 transpuesta
'
'   Matrices para el calculo inicial de TETA y P
'
'   QA = Qk*Ak+1
'   QAT = QA transpuesta
'   Q0 = I
'   TETA0 = 0
'   P0 = 0

REDIM P0(P, P), Q0(P, P), QA(P, 1), QAT(1, P), QK(P, P)
REDIM AUX5(P, P), AUX6(1, 1), AUX7(1, 1), TETA0(P, 1)
REDIM AUX8(P, P), AUX9(1, 1), AUX10(P, P), PA(P, 1), PAT(1, P)


CALL RPTD(NG, MG, T, NTP, PN(), PD(), Y())

CLS
LOCATE 11, 15
PRINT "INICIALIZANDO EL SISTEMA"
LOCATE 12, 15
PRINT "Por favor espere..."



FOR I = 1 TO NTP + 1
UO(I) = 1
NEXT I

FOR I = 1 TO NTP
FOR J = 1 TO P
AK(I, J) = 0
NEXT J, I

FOR I = 1 TO P
YP(I, 1) = Y(I)
```

```
NEXT I


FOR I = 1 TO NTP
IF I <= M + 1 THEN
 FOR J = 1 TO I
 AK(I, J) = UO(I)
 NEXT J
ELSE
 FOR J = 1 TO M + 1
 AK(I, J) = UO(I)
 NEXT J
END IF
NEXT I

FOR I = 1 TO NTP - 1
FOR J = M + 2 TO P
AK(I + 1, J) = -Y(I - J + M + 2)
NEXT J, I

FOR I = 1 TO P
FOR J = 1 TO P
AP(I, J) = AK(I, J)
NEXT J, I

NP = P

'
' Comienza calculo de primera aproximacion de TETA
'

FOR I = 1 TO P
Q0(I, I) = 1
NEXT I

ICONT = 1
WHILE ICONT <= P
  FOR I = 1 TO P
   AK1(I, 1) = AK(ICONT, I)
   AKT(1, I) = AK(ICONT, I)
   NEXT I
   N1 = P
   M1 = 1
   CALL MULTM(NP, N1, M1, QA(), Q0(), AK1())

   FOR I = 1 TO P
   QAT(1, I) = QA(I, 1)
   NEXT I
   N1 = P
   M1 = P
   NAUX = 1
   CALL MULTM(NAUX, N1, M1, AUX5(), QA(), QAT())

   N1 = 1
   M1 = 1
   CALL MULTM(NP, N1, M1, AUX6(), AKT(), QA())

   FOR I = 1 TO P
   FOR J = 1 TO P
   QK(I, J) = Q0(I, J) - (AUX5(I, J) / AUX6(1, 1))
   NEXT J, I

   N1 = 1
```

```
    M1 = 1
    CALL MULTM(NP, N1, M1, AUX7(), AKT(), TETA0())

    CN1 = Y(ICONT) - AUX7(1, 1)
    FOR I = 1 TO P
    TP(I, 1) = TETA0(I, 1) + (QA(I, 1) * CN1 / AUX6(1, 1))
    NEXT I

    FOR I = 1 TO P
    FOR J = 1 TO P
    Q0(I, J) = QK(I, J)
    NEXT J, I
    FOR I = 1 TO P
    FOR J = 1 TO P
    QK(I, J) = 0
    NEXT J, I

    FOR I = 1 TO P
    TETA0(I, 1) = TP(I, 1)
    NEXT I
    FOR I = 1 TO P
    TP(I, 1) = 0
    NEXT I

    N1 = P
    M1 = 1
    CALL MULTM(NP, N1, M1, PA(), P0(), AK1())

    FOR I = 1 TO P
    PAT(1, I) = PA(I, 1)
    NEXT I
    N1 = P
    M1 = P
    CALL MULTM(NAUX, N1, M1, AUX10(), PA(), PAT())
    CALL MULTM(NAUX, N1, M1, AUX8(), QA(), PAT())
    N1 = 1
    M1 = 1
    CALL MULTM(NP, N1, M1, AUX9(), AKT(), PA())
    AUX9(1, 1) = AUX9(1, 1) + 1

    FOR I = 1 TO P
    FOR J = 1 TO P
    PP(I, J) = P0(I, J) - ((AUX10(I, J) + AUX8(I, J)) / AUX6(1, 1)) + (AUX5(I, J) * AUX9(1, 1) / (AUX6(1, 1) ^ 2))
    NEXT J, I

    FOR I = 1 TO P
    FOR J = 1 TO P
    P0(I, J) = PP(I, J)
    NEXT J, I

    ICONT = ICONT + 1

WEND


FOR I = 1 TO P
TK1(I, 1) = TETA0(I, 1)
NEXT I

CLS
LOCATE 11, 15
```

```
PRINT "CALCULANDO ITERACION"
LOCATE 12, 15
PRINT "DE UN TOTAL DE "; NTP; " ITERACIONES"
LOCATE 14, 15
PRINT "Por favor espere..."

ITERACIONES:

ICONT = P + 1
WHILE ICONT <= NTP
  FOR I = 1 TO P
  AK1(I, 1) = AK(ICONT, I)
  AKT(1, I) = AK(ICONT, I)
  NEXT I
  N1 = P
  M1 = 1
  CALL MULTM(NP, N1, M1, AUX1(), PP(), AK1())
  N1 = 1
  M1 = 1
  CALL MULTM(NP, N1, M1, AUX2(), AKT(), TK1())
  CNUM = Y(ICONT) - AUX2(1, 1)

  N1 = 1
  M1 = 1
  CALL MULTM(NP, N1, M1, AUX3(), AKT(), AUX1())
  CDEN = AUX3(1, 1) + 1

  FOR I = 1 TO P
  AUX1T(1, I) = AUX1(I, 1)
  NEXT I

  NC = 1
  N1 = P
  M1 = P
  CALL MULTM(NC, N1, M1, AUX4(), AUX1(), AUX1T())


  FOR I = 1 TO P
  AUX1(I, 1) = AUX1(I, 1) * CNUM / CDEN
  NEXT I

  FOR I = 1 TO P
  TETA(I, 1) = TK1(I, 1) + AUX1(I, 1)
  NEXT I

  FOR I = 1 TO P
  FOR J = 1 TO P
  AUX4(I, J) = AUX4(I, J) / CDEN
  NEXT J, I

  FOR I = 1 TO P
  FOR J = 1 TO P
  PK1(I, J) = PP(I, J) - AUX4(I, J)
  NEXT J, I

  LOCATE 11, 37
  PRINT ICONT

  FOR I = 1 TO P
  TK1(I, 1) = TETA(I, 1)
```

```
      NEXT I
      FOR I = 1 TO P
      FOR J = 1 TO P
      PP(I, J) = PK1(I, J)
      NEXT J, I

      ICONT = ICONT + 1
    WEND

    CLS
    LOCATE 12, 20
    PRINT "CALCULANDO FUNCION DE TRANSFERENCIA EN TIEMPO DISCRETO"
    LOCATE 13, 20
    PRINT "POR FAVOR ESPERE..."

    CLS
    PRINT "SISTEMA REDUCIDO EN TIEMPO DISCRETO"
    PRINT
    PRINT
    PRINT "Coeficientes del numerador"
    PRINT
    FOR I = 1 TO M + 1
    PRINT " Coef. grado"; M + 1 - I; " = ";
    PRINT USING "####.######"; TETA(I, 1)
    NEXT I
    PRINT
    PRINT
    PRINT "Coeficientes del denominador"
    PRINT
    PRINT " Coef. grado"; 2; "=    1.000000"
    FOR I = M + 2 TO M + N + 1
    PRINT " Coef. grado"; M + 3 - I; " = ";
    PRINT USING "####.######"; TETA(I, 1)
    NEXT I
    PRINT : PRINT
    PRINT "Presione cualquier tecla para continuar"
30  IF INKEY$ = "" THEN 30


'
'   Comienza el calculo del sistema continuo
'
    IF N = 2 THEN

        NR = N
        REDIM F(NR, NR), G(NR, 1), H(1, NR)
        REDIM A(NR, NR), B(NR, NR), AUX(NR, NR)
        FOR I = 1 TO NR
        FOR J = 1 TO NR
        F(I, J) = 0
        NEXT J, I
        F(2, 1) = 1
        FOR I = 1 TO NR
        F(I, 2) = -TETA(N + M + 2 - I, 1)
        NEXT I

        FOR I = 1 TO NR
        G(I, 1) = TETA(M + 2 - I, 1)
        NEXT I

50      CALL QRAIZ(N, TT, F(), A(), Z)
```

```
    IF Z = 1 THEN
51      CLS
      PRINT "LA TOLERANCIA INGRESADA NO ES CONVENIENTE"
      PRINT
      PRINT "PUEDE INGRESAR UN EPSILON MAYOR O FINALIZAR EL PROGRAMA"
      PRINT
      PRINT "ESCOJA : OTRO EPSILON (E) O FINALIZAR (F)";
      INPUT C$
      IF C$ = "E" OR C$ = "e" THEN 50
      IF C$ <> "F" AND C$ <> "f" THEN 51
      GOTO FINAL
    ELSE
    END IF

'
'       Calculo de la matriz B
'
        REDIM F1(N, N), F1I(N, N), AUX1(N, N)

        FOR I = 1 TO N
        FOR J = 1 TO N
        F1(I, J) = F(I, J)
        NEXT J, I
        FOR I = 1 TO N
        F1(I, I) = F1(I, I) - 1
        NEXT I
        N1 = N
        M1 = N
        CALL MINV(N, F1(), F1I(), Z)
        IF Z = 1 THEN GOTO FINAL
        CALL MULTM(N, N1, M1, AUX1(), A(), F1I())
        N1 = N
        M1 = 1
        CALL MULTM(N, N1, M1, B(), AUX1(), G())

        FOR I = 1 TO NR
        H(1, I) = 0
        NEXT I
        H(1, NR) = 1

'
'       Calculo de la funcion de transferencia en el plano s
'
        DIM SIA1(NR, NR), SIA2(NR, NR), PDEN(NR + 1), PNUM(NR), DIAG(NR)
        FOR I = 1 TO NR
        SIA1(I, I) = 1
        NEXT I
        FOR I = 1 TO NR
        FOR J = 1 TO NR
        SIA2(I, J) = -A(I, J)
        NEXT J, I

        TEMPO = SIA2(2, 2)
        SIA2(2, 2) = SIA2(1, 1)
        SIA2(1, 1) = TEMPO
        SIA2(1, 2) = -SIA2(1, 2)
        SIA2(2, 1) = -SIA2(2, 1)


        FOR I = 1 TO NR
        DIAG(I) = SIA2(I, I)
        NEXT I
```

```
    CALL MBIN(NR, DIAG(), PDEN())
    PDEN(3) = PDEN(3) - SIA2(1, 2) * SIA2(2, 1)
    REDIM R1(NR, 1), R2(NR, 1), AUX(1, 1)

    N1 = NR
    M1 = 1
    CALL MULTM(NR, N1, M1, R1(), SIA1(), B())
    CALL MULTM(NR, N1, M1, R2(), SIA2(), B())
    N1 = 1
    M1 = 1
    CALL MULTM(NR, N1, M1, AUX(), H(), R1())
    PNUM(1) = AUX(1, 1)
    CALL MULTM(NR, N1, M1, AUX(), H(), R2())
    PNUM(2) = AUX(1, 1)
   GAINR = GAIN * PDEN(NR + 1) / PNUM(NR)
    FOR I = 1 TO NR
    PNUM(I) = PNUM(I) * GAINR
    NEXT I


    IF X$ = "T" OR X$ = "D" THEN
    PRINT #1, : PRINT #1,
    ELSE
     CLS
    END IF

    PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
    PRINT #1,
    PRINT #1, "GRADO NUMERADOR: "; NR - 1
    FOR I = 1 TO NR
    PRINT #1, "COEFICIENTE GRADO"; NR - I; " = ";
    PRINT #1, USING "####.######"; PNUM(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "GRADO DENOMINADOR:"; NR
    FOR I = 1 TO NR + 1
    PRINT #1, "COEFICIENTE GRADO"; NR + 1 - I; " = ";
    PRINT #1, USING "####.#####"; PDEN(I)
    NEXT I

ELSE
    F = -TETA(N + M + 1, 1)
    G = TETA(M + 1, 1)
    H = 1

    IF F <= 0 THEN
      CLS
      PRINT "EL SISTEMA OBTENIDO ES INESTABLE"
      PRINT "PRUEBE REDUCIENDOLO A  UN SISTEMA DE SEGUNDO ORDEN"
      GOTO FINAL
    ELSE
    END IF
    A = LOG(F) / TT
    B = A * G / (F - 1)
   GAINR = GAIN * (-A) / B
    B = GAINR * B

    IF X$ = "T" OR X$ = "D" THEN
    PRINT #1, : PRINT #1,
    ELSE
     CLS
```

```
        END IF
        PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
        PRINT #1,
        PRINT #1, "GRADO NUMERADOR: "; NR - 1
        PRINT #1, "COEFICIENTE GRADO"; NR - 1; " = ";
        PRINT #1, USING "#######.#####"; B
        PRINT #1, : PRINT #1,
        PRINT #1, "GRADO DENOMINADOR:"; NR
        PRINT #1, "COEFICIENTE GRADO 1 =  1.00000"
        PRINT #1, "COEFICIENTE GRADO 0 =";
        PRINT #1, USING "#######.#####"; -A

    END IF

FINAL:

    PRINT
    PRINT
    PRINT "Presione cualquier tecla para continuar"
102   IF INKEY$ = "" THEN 102
    CHAIN "MENU3F"


    END
```

```
DECLARE SUB VALP (N%, SS#(), VPR#(), VPI#())
DECLARE SUB QRAIZ (N%, T#, F#(), A#(), Z#)
DECLARE SUB MBIN (N1%, BE#(), PS#())
DECLARE SUB MPINV (N%, M%, A1#(), AI1#())
DECLARE SUB MULTM (N%, N1%, M1%, VR#(), VM1#(), VM2#())
DECLARE SUB MINV (N%, AI#(), CI#(), F1#)
DECLARE SUB VECP (N%, SS!(), V#())
DECLARE SUB ECUA (N%, A#(), B#(), X#())
DECLARE SUB ROOT (NG%, A#(), R1#(), RI1#())
DECLARE SUB RITD (N%, M%, NR%, T#, A#(), B#(), Q#(), QOR#(), QOI#(), QR11#(), QI11#(), QR12#(), QI12#(), G#())
'
'    METODO DE VALORES SINGULARES
'
'    En esta version, la matriz de Hankel se la llena desde el
'    valor de la funcion de transferencia para T=0
'
    CLEAR
    CLS
    DEFINT I-N
    DEFDBL A-H, O-R, T-Z
    DEFSNG S
    PRINT "METODO DE LOS VALORES SINGULARES"
    PRINT STRING$(32, "=")
    PRINT


9    PRINT : PRINT
    PRINT TAB(20); "IMPRESION EN PAPEL? (I) "
    PRINT TAB(20); "IMPRESION EN ARCHIVO DE DISCO? (D)"
    PRINT TAB(20); "IMPRESION PANTALLA (P)"
    PRINT
    PRINT TAB(20); "Ingrese I, D o P";
    INPUT X$
    IMPR$ = "LPT1:"
    IF X$ = "I" OR X$ = "i" THEN
    GOTO 10
    ELSEIF X$ = "D" OR X$ = "d" THEN
    GOTO 11
    ELSEIF X$ = "P" OR X$ = "p" THEN
    GOTO 12
    ELSEIF (X$ <> "P" AND X$ <> "p") OR (X$ <> "D" AND X$ <> "d") OR (X$ <> "I" AND X$ <> "i") THEN
    GOTO 9
    END IF
12   IMPR$ = "SCRN:"
    GOTO 10
11   CLS
    PRINT "INGRESE EL NOMBRE DEL ARCHIVO A GENERAR:";
    INPUT AD$
    IMPR$ = AD$


10   OPEN IMPR$ FOR OUTPUT AS #1


3    CLS
    INPUT "GRADO DENOMINADOR; N = "; N
    IF N <= 0 THEN 3
    PRINT
    INPUT "GRADO NUMERADOR;(M no puede ser mayor que N) M = "; M
    IF M > N OR M < 0 THEN 3
    REDIM PN(M + 1), PD(N + 1)
    CLS
```

```
    PRINT " INGRESE EL DENOMINADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO N + 1
    PRINT "COEFICIENTE GRADO"; N + 1 - I; " = ";
    INPUT PD(I)
    NEXT I
    PRINT
    PRINT " INGRESE EL NUMERADOR ORIGINAL COMENZANDO POR GRADO MAYOR"
    PRINT
    FOR I = 1 TO M + 1
    PRINT "COEFICIENTE GRADO"; M + 1 - I; " = ";
    INPUT PN(I)
    NEXT I
    PRINT : PRINT
2   PRINT "ESTA CORRECTO (S/N)";
    INPUT P$
    IF P$ = "N" OR P$ = "n" THEN 3
    IF P$ <> "S" AND P$ <> "s" THEN 2

    CLS
    PRINT #1, " FUNCION DE TRANSFERENCIA ORIGINAL "
    PRINT #1, : PRINT #1,
    PRINT #1, "NUMERADOR GRADO: "; M

    FOR I = 1 TO M + 1
    PRINT #1, "Coef. grado"; M + 1 - I; "= ";
    PRINT #1, USING "######.###"; PN(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "DENOMINADOR GRADO:"; N
    FOR I = 1 TO N + 1
    PRINT #1, "Coef. grado"; N + 1 - I; "= ";
    PRINT #1, USING "######.###"; PD(I)
    NEXT I
    PRINT #1,
    PRINT #1,
    PRINT
    PRINT "Presione cualquier tecla para continuar"
24  IF INKEY$ = "" THEN 24


COMIENZO:

    REDIM Q(N), QOR(N), QOI(N), QR11(N), QI11(N), QR12(N), QI12(N)
    GAIN = PN(M + 1) / PD(N + 1)


20  IF X$ = "T" OR X$ = "D" THEN
     PRINT #1, : PRINT #1,
    ELSE
     CLS
    END IF

    PRINT "INGRESE EL PERIODO DE MUESTREO (T)"
    PRINT
    INPUT "T = "; T
    IF T <= 0 THEN 20
    PRINT
25  PRINT "ESTA CORRECTO (S/N)";
    INPUT Q$
    IF Q$ = "N" OR Q$ = "n" THEN 20
```

```
    IF Q$ <> "S" AND Q$ <> "s" THEN 25
    PRINT
    PRINT #1, "PERIODO DE MUESTREO = "; T
    PRINT #1,
    PRINT : PRINT
    PRINT "Presione cualquier tecla para continuar"
23   IF INKEY$ = "" THEN 23
    TT = T
    REDIM W(4 * N + 1)
    CALL RITD(N, M, 2 * N, T, PN(), PD(), Q(), QOR(), QOI(), QR11(), QI11(), QR12(), QI12(), W())

'
'   Llenado de matriz con valores de la respuesta impulso
'

    REDIM S(N + 1, N + 1)

    FOR I = 1 TO N + 1
    K = I
    FOR J = 1 TO N + 1
    S(I, J) = W(K)
    K = K + 1
    NEXT J, I
'
'   Z1.- Matriz diagonal con los valores propios de S
'   ZA.- Matriz diagonal con los valores singulares de S
'      (valores absolutos de valores propios)
'
    NR1 = N + 1
    REDIM Z1(NR1, NR1)
    REDIM ZA(NR1, NR1), VPPR(NR1), VPPI(NR1)

    FOR I = 1 TO NR1
    FOR J = 1 TO NR1
    Z1(I, J) = S(I, J)
    NEXT J, I

    CALL VALP(NR1, Z1(), VPPR(), VPPI())

    FOR I = 1 TO NR1
    ZA(I, I) = SQR(VPPR(I) ^ 2 + VPPI(I) ^ 2)
    NEXT I
'
'   ZR.- Matriz raiz cuadrada de ZA y reducida  al orden del nuevo sistema
'   VPPR.- Parte real de los valores propios de la matriz Z1
'   VPPI.- Parte imaginaria de los valores propios de la matriz Z1
'
    K = 0
    FOR I = 1 TO NR1
    IF ZA(I, I) >= (ZA(1, 1) / 100) THEN
      K = K + 1
    ELSE
    END IF
    NEXT I
    IF K < 1 THEN
      CLS
      PRINT "EL PERIODO DE MUESTREO ESCOGIDO NO ES CONVENIENTE"
      PRINT "PARA RESOLVER EL SISTEMA POR ESTE METODO"
      PRINT
      PRINT "PRUEBE ESCOGIENDO OTRO PERIODO DE MUETREO"
      PRINT
26     PRINT "Escoger otro periodo (P); Finalizar (F)";
```

```
      INPUT D$
        IF D$ = "P" OR D$ = "p" THEN 20
        IF D$ <> "F" AND D$ <> "f" THEN 26
        GOTO FINAL
      ELSE
      END IF
    IF K = N THEN
      CLS
      PRINT "El sistema no puede ser resuelto con ese periodo de muestreo"
      PRINT " ya que se obtuvo un sistema de orden similar al original"
      PRINT
      PRINT "Puede probar con otro periodo de muetreo"
      PRINT "o finalizar y usar otro metodo"
      PRINT
28     PRINT "ESGOJA: OTRO PERIODO (P) O FINALIZAR (F)";
      INPUT D2$
      IF D2$ = "P" OR D2$ = "p" THEN 20
      IF D2$ <> "F" AND D2$ <> "f" THEN 28
      GOTO FINAL
    ELSE
    END IF

    IF K > 2 THEN
      CLS
      PRINT "## PRECAUCION !!"
      PRINT
      PRINT "Este programa no es capaz de obtener la funcion de transferencia"
      PRINT "para un sistema mayor al de segundo orden."
      PRINT
      PRINT "Usted solamente podra obtener las matrices asociadas al"
      PRINT "sistema reducido"
      PRINT
27     PRINT "Desea continuar (C) o Finalizar (F)";
      INPUT D1$
      IF D1$ = "F" OR D1$ = "f" THEN GOTO FINAL
      IF D1$ <> "c" AND D1$ <> "C" THEN 27
    ELSE
    END IF


    CLS
    LOCATE 11, 25
    PRINT "ORDEN DEL SISTEMA REDUCIDO = "; K
    NR1 = K + 1
    LOCATE 25, 21
    PRINT "Presione cualquier tecla para continuar"
21   IF INKEY$ = "" THEN 21

    IF X$ = "T" OR X$ = "D" THEN
      PRINT #1,
      PRINT #1, "Orden del sistema reducido = "; K
      PRINT #1,
      CLS
    ELSE
    END IF


    REDIM ZR(NR1, NR1)

    FOR I = 1 TO NR1
    FOR J = 1 TO NR1
    ZR(I, J) = Z1(I, J)
```

```
NEXT J, I


'
'   Comienza la realizacion
'

NR = NR1

REDIM EI(NR, 1), EIt(1, NR), E(NR, 1), C(1, NR - 1)
REDIM EAUX1(NR, 1), EAUX2(1, NR), ZSAUX(NR, NR)
REDIM IX(NR), IX1(NR)

FOR I = 1 TO NR - 1
C(1, I) = ZR(2, I)
NEXT I

I = 1
WHILE I <= NR - 1
  IX(I) = I
  J = 1
  WHILE J <= NR
    IF ZR(J, I) = 0 THEN
      IX(I) = J
      IF J = NR THEN
        CLS
        PRINT "LA MATRIZ DE HANKEL ES DE RANGO MENOR A"; NR
        PRINT "SE NECESITA UNA MATRIZ DE HANKEL DE RANGO"; NR + 1
        PRINT
        PRINT "ESCOJA OTRO SISTEMA"
        END
      ELSE
      END IF
    ELSE
      IF I = 1 THEN
        IX(I) = J
        IX1(I) = J
        J = NR + 1
      ELSE
        FOR K = 1 TO I - 2
        MIN = IX1(K)
        FOR K1 = K + 1 TO I - 1
        IF IX1(K1) > MIN THEN
        ELSE
          IX1(K) = IX1(K1)
          IX1(K1) = MIN
          MIN = IX1(K)
        END IF
        NEXT K1, K
        FOR K = 1 TO I - 1
        IF IX1(K) = J THEN
          J = J + 1
        ELSE
        END IF
        NEXT K
        IF ZR(J, I) = 0 THEN
        ELSE
          IX(I) = J
          IX1(I) = J
          J = NR + 1
        END IF
      END IF
```

```
    END IF
    J = J + 1
  WEND
  FOR J = 1 TO NR
  EI(J, 1) = 0
  EIt(1, J) = 0
  E(J, 1) = 0
  NEXT J
  EI(IX(I), 1) = 1
  EIt(1, IX(I)) = 1
  E(I, 1) = 1

  N1 = NR
  M1 = 1
  CALL MULTM(NR, N1, M1, EAUX1(), ZR(), E())

  FOR J = 1 TO NR
  EAUX1(J, 1) = EAUX1(J, 1) - EI(J, 1)
  NEXT J

  N1 = 1
  M1 = NR
  CALL MULTM(NR, N1, M1, EAUX2(), EIt(), ZR())

  N1 = NR
  M1 = NR
  CALL MULTM(1, N1, M1, ZSAUX(), EAUX1(), EAUX2())

  DIV = ZR(IX(I), I)
  FOR J = 1 TO NR
  FOR K = 1 TO NR
  ZSAUX(J, K) = ZSAUX(J, K) / DIV
  NEXT K, J
  FOR J = 1 TO NR
  FOR K = 1 TO NR
  ZR(J, K) = ZR(J, K) - ZSAUX(J, K)
  NEXT K, J
  I = I + 1
WEND


'
'  Ordenamiento de la matriz de Hankel reducida
'  para la obtencion de la matriz F
'

REDIM ZSM(NR, NR), IROW(NR - 1)
I = 1
WHILE I <= NR - 1
 FOR J = 1 TO NR
 IF ZR(J, I) > .98 AND ZR(J, I) <= 1 THEN
  IROW(I) = J
 ELSE
 END IF
 NEXT J
 I = I + 1
WEND

FOR I = 1 TO NR - 1
FOR J = 1 TO NR
ZSM(I, J) = ZR(IROW(I), J)
NEXT J, I
```

```
    NR = NR - 1
    REDIM F(NR, NR), A(NR, NR), B(NR, 1)
    FOR I = 1 TO NR
    B(I, 1) = ZSM(I, 1)
    NEXT I
    FOR I = 1 TO NR
    FOR J = 1 TO NR
    F(I, J) = ZSM(I, J + 1)
    NEXT J, I

    IF NR > 1 THEN
64      T = TT
        CALL QRAIZ(NR, T, F(), A(), Z)
        IF Z = 1 THEN
65       CLS
         PRINT "LA TOLERANCIA INGRESADA NO ES CONVENIENTE"
         PRINT
         PRINT "PUEDE INGRESAR UN EPSILON MAYOR O FINALIZAR EL PROGRAMA"
         PRINT
         PRINT "ESCOJA : OTRO EPSILON (E) O FINALIZAR (F)";
         INPUT C$
         IF C$ = "E" OR C$ = "e" THEN 64
         IF C$ <> "F" AND C$ <> "f" THEN 65
         GOTO FINAL
        ELSE
        END IF
    ELSE
    END IF


'
'   Calculo de la funcion de transferencia en el plano s
'
    IF NR = 2 THEN
        REDIM TIA1(NR, NR), TIA2(NR, NR), PDEN(NR + 1), PNUM(NR), DIAG(NR)

        FOR I = 1 TO NR
        TIA1(I, I) = 1
        NEXT I
        FOR I = 1 TO NR
        FOR J = 1 TO NR
        TIA2(I, J) = -A(I, J)
        NEXT J, I

        TEMPO = TIA2(2, 2)
        TIA2(2, 2) = TIA2(1, 1)
        TIA2(1, 1) = TEMPO
        TIA2(1, 2) = -TIA2(1, 2)
        TIA2(2, 1) = -TIA2(2, 1)

        FOR I = 1 TO NR
        DIAG(I) = TIA2(I, I)
        NEXT I

        CALL MBIN(NR, DIAG(), PDEN())
        PDEN(3) = PDEN(3) - TIA2(1, 2) * TIA2(2, 1)
        REDIM R1(NR, 1), R2(NR, 1), AUX(1, 1)

        N1 = NR
        M1 = 1
```

```
    CALL MULTM(NR, N1, M1, R1(), TIA1(), B())
    CALL MULTM(NR, N1, M1, R2(), TIA2(), B())
    N1 = 1
    M1 = 1
    CALL MULTM(NR, N1, M1, AUX(), C(), R1())
    PNUM(1) = AUX(1, 1)
    CALL MULTM(NR, N1, M1, AUX(), C(), R2())
    PNUM(2) = AUX(1, 1)

    GAINR = GAIN * PDEN(NR + 1) / PNUM(NR)
    FOR I = 1 TO NR
    PNUM(I) = PNUM(I) * GAINR
    NEXT I

    IF X$ = "T" OR X$ = "D" THEN
     PRINT #1, : PRINT #1,
    ELSE
     CLS
    END IF

    PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
    PRINT #1,
    PRINT #1, "GRADO NUMERADOR: "; NR - 1
    FOR I = 1 TO NR
    PRINT #1, "COEFICIENTE GRADO"; NR - I; " = ";
    PRINT #1, USING "######.####"; PNUM(I)
    NEXT I
    PRINT #1, : PRINT #1,
    PRINT #1, "GRADO DENOMINADOR:"; NR
    FOR I = 1 TO NR + 1
    PRINT #1, "COEFICIENTE GRADO"; NR + 1 - I; " = ";
    PRINT #1, USING "######.####"; PDEN(I)
    NEXT I

ELSEIF NR = 1 THEN

    F1 = F(NR, NR)
    B1 = B(NR, 1)
    C1 = C(1, NR)

    IF F1 <= 0 THEN
      CLS
      PRINT "EL SISTEMA OBTENIDO ES INESTABLE"
      PRINT "PRUEBE REDUCIENDOLO A  UN SISTEMA DE SEGUNDO ORDEN"
      GOTO FINAL
    ELSE
    END IF
    A = LOG(F1) / TT

    RN = B1 * C1
    GAINR = GAIN * (-A) / RN
    RN = GAINR * RN

    IF X$ = "T" OR X$ = "D" THEN
    PRINT #1, : PRINT #1,
    ELSE
     CLS
    END IF
    PRINT #1, "FUNCION DE TRANSFERENCIA REDUCIDA"
    PRINT #1,
    PRINT #1, "GRADO NUMERADOR: "; NR - 1
    PRINT #1, "COEFICIENTE GRADO"; NR - 1; " = ";
```

```
        PRINT #1, USING "######.####"; RN
        PRINT #1, : PRINT #1,
        PRINT #1, "GRADO DENOMINADOR:"; NR
        PRINT #1, "COEFICIENTE GRADO 1 =  1.00000"
        PRINT #1, "COEFICIENTE GRADO 0 =";
        PRINT #1, USING "######.####"; -A

    ELSE

        IF X$ = "I" OR X$ = "D" THEN
         PRINT #1, : PRINT #1,
        ELSE
         CLS
        END IF

        PRINT #1, "IMPRESION DE LA MATRIZ A"
        N1 = NR1 - 1
        LOCATE 5, 1
        PRINT #1, "Ú"; STRING$(N1 * 13, " "); "¿"
        FOR J = 1 TO N1
        LOCATE 5 + J, 1
        PRINT #1, "³";
        FOR I = 1 TO N1
        PRINT #1, "";
        PRINT #1, USING "#######.####"; A(J, I);
        PRINT #1, " ";
        NEXT I
        PRINT #1, "³"
        NEXT J
        LOCATE N1 + 6, 1
        PRINT #1, "À"; STRING$(N1 * 13, " "); "Ù"
        PRINT #1,
        PRINT : PRINT
        PRINT "Presione cualquier tecla para continuar"
66      IF INKEY$ = "" THEN 66

        IF X$ = "I" OR X$ = "D" THEN
         PRINT #1, : PRINT #1,
        ELSE
         CLS
        END IF

        PRINT #1, "IMPRESION DE LA MATRIZ B"
        N1 = NR
        LOCATE 5, 1
        PRINT #1, "Ú"; STRING$(12, " "); "¿"
        FOR J = 1 TO N1
        LOCATE 5 + J, 1
        PRINT #1, "³";
        PRINT #1, USING "#######.####"; B(J, 1);
        PRINT #1, "³"
        NEXT J
        LOCATE N1 + 6, 1
        PRINT #1, "À"; STRING$(12, " "); "Ù"
        PRINT #1,
        PRINT #1,

        PRINT #1, "IMPRESION DE LA MATRIZ C"
        N1 = NR
        M1 = 1
        LOCATE N1 + 10, 1
        PRINT #1, "Ú"; STRING$(N1 * 13, " "); "¿"
```

```
        FOR J = 1 TO M1
        LOCATE N1 + 10 + J, 1
        PRINT #1, "≥";
        FOR I = 1 TO N1
        PRINT #1, "";
        PRINT #1, USING "#######.####"; C(J, I);
        PRINT #1, " ";
        NEXT I
        PRINT #1, "≥"
        NEXT J
        LOCATE N1 + M1 + 11, 1
        PRINT #1, "¿"; STRING$(N1 * 13, " "); "Ÿ"
        PRINT #1,
        PRINT : PRINT
        PRINT "Presione cualquier tecla para continuar"
55       IF INKEY$ = "" THEN 55

    END IF

FINAL:
    PRINT
    PRINT
    PRINT "Presione cualquier tecla para continuar"
101   IF INKEY$ = "" THEN 101
    CHAIN "MENU4F"


    END
```

```
DECLARE SUB VECP (N%, SS!(), V#())
DECLARE SUB MPINV (N%, M%, A#(), AI#())
DECLARE SUB FRAC (N#, NR#, DR#, G#(), Nu#(), D#(), N1#(), N2#(), N3#(), D1#(), D2#(), D3#())
DECLARE SUB DETER (ND%, AD#(), DET#)
DECLARE SUB EVALPC (N%, AB#(), Pr#, Pi#, Vr#, Vi#)
DECLARE SUB DIFP (N%, AB#(), ABD#())
DECLARE SUB MBIN (N1%, BE#(), PS#())
DECLARE SUB MINV (N%, AI#(), CI#(), F#)
DECLARE SUB MULTM (N%, N1%, M1%, Vr#(), VM1#(), VM2#())
DECLARE SUB ROOT (NG%, A#(), R1#(), RI1#())
DECLARE SUB ECUA (N%, A#(), B#(), X#())
DECLARE SUB RITD (N%, M%, NR%, T#, A#(), B#(), Q#(), QOR#(), QOI#(), QR11#(), QI11#(), QR12#(), QI12#(), W#())
DECLARE SUB QRAIZ (N%, T#, F#(), A#(), Z#)

'
'     Archivo de Subrutinas
'

DEFINT I-N
DEFDBL A-H, O-Z
SUB DETER (ND, AD(), DETM)
'
'     CALCULO DE DETERMINANTE
'     Usa proceso de pivotaje y reduccion de Gauss a matriz
'     triangular superior

     DETM = 1
'
'     Comienza reduccion y pivotaje
'
     I = 1
     WHILE I <= ND - 1
       F = I
       T = ABS(AD(I, I))
'
'     Buscamos elemento pivote
'
       FOR J = I TO ND
       IF ABS(AD(J, I)) > T THEN
         F = J
         T = ABS(AD(J, I))
       ELSE
       END IF
       NEXT J
       IF T = 0 THEN GOTO NOTA
'
'     Intercambio de filas en A
'
       IF F <> I THEN
         FOR J = I TO ND
         P = AD(I, J)
         Q = AD(F, J)
         AD(I, J) = Q
         AD(F, J) = P
         NEXT J
         DETM = -1 * DETM
       ELSE
       END IF

'
'     Inicio de reduccion
'
```

```
        FOR J = I + 1 TO ND
        FAC = -AD(J, I) / AD(I, I)

        FOR K = I TO ND
        AD(J, K) = AD(J, K) + AD(I, K) * FAC
        NEXT K
        NEXT J
        I = I + 1

    WEND


    IF AD(ND, ND) = 0 THEN GOTO NOTA
'
'   Evaluacion del determinante
'
    FOR I = 1 TO ND
    DETM = DETM * AD(I, I)
    NEXT I

    GOTO FIN

NOTA:
    DETM = 0

FIN:

END SUB

SUB DIFP (N, AB(), ABD())
    M = N - 1
    C = N
    I = 1
    WHILE I <= N
        ABD(I) = C * AB(I)
        C = C - 1
        I = I + 1
    WEND

END SUB

SUB ECUA (N, A(), B(), X())

'   Resolucion de Sistemas de ecuaciones
'   METODO DE GAUSS-JORDAN
'   Se usa pivotaje completo
'
'   N.- Orden del sistema
'   A.- Matriz con coeficientes de las ecuaciones
'   B.- Vector con terminos independientes
'   X.- Vector con resultados
'
'   Parametros entrada para subrutina:  (NO EXISTE ESTA SUBRUTINA)
'   N.-  Orden del sistema
'   A.-  Matriz con coeficientes de las ecuaciones
'   B.-  Vector con terminos independientes
'   X.-  Vector de  salida resultados
'   SUB ECUA(N, A(), B(), X())
'

'
'   Comienza reduccion y pivotaje
```

```
'
    I = 1
    WHILE I <= N - 1
      F = I
      T = ABS(A(I, I))
'
'     Buscamos elemento pivote
'
      FOR J = I TO N
      IF ABS(A(J, I)) > T THEN
        F = J
        T = ABS(A(J, I))
      ELSE
      END IF
      NEXT J
      IF T = 0 THEN GOTO FINAL
'
'     Intercambio de filas en A y B
'
      IF F <> I THEN
        FOR J = I TO N
        P = A(I, J)
        Q = A(F, J)
        A(I, J) = Q
        A(F, J) = P
        NEXT J
        P = B(I)
        Q = B(F)
        B(I) = Q
        B(F) = P
      ELSE
      END IF

'
'     Inicio de reduccion
'
      FOR J = I + 1 TO N
      FAC = -A(J, I) / A(I, I)

      FOR K = I TO N
      A(J, K) = A(J, K) + A(I, K) * FAC
      NEXT K
      B(J) = B(J) + B(I) * FAC
      NEXT J
      I = I + 1

    WEND


    IF A(N, N) = 0 THEN GOTO FINAL
'
'     Evaluacion de las incognitas
'
    X(N) = B(N) / A(N, N)
    I = N
    WHILE I >= 2
      S = 0
      FOR J = I TO N
      S = S + X(J) * A(I - 1, J)
      NEXT J
      X(I - 1) = (B(I - 1) - S) / A(I - 1, I - 1)
      I = I - 1
```

```
    WEND
    GOTO SALTO

FINAL:
    CLS
    PRINT "EL SISTEMA ES LINEALMENTE DEPENDIENTE Y NO TIENE SOLUCION"

SALTO:
END SUB

SUB EVALPC (N, AB(), Pr, Pi, Vr, Vi)
DIM RR(N + 1), RI(N + 1)
    II = 1
    N1 = N
    WHILE N1 >= 2
      I = 1: R1 = Pr: R2 = Pi
      WHILE I < N1
        R11 = R1 * Pr - R2 * Pi
        R22 = R1 * Pi + R2 * Pr
        R1 = R11
        R2 = R22
        I = I + 1
      WEND
      RR(II) = R1
      RI(II) = R2
      N1 = N1 - 1
      II = II + 1
    WEND
    RR(II) = Pr
    RI(II) = Pi
    TR = 0
    TI = 0
    FOR I = 1 TO N
    TR = TR + AB(I) * RR(I)
    TI = TI + AB(I) * RI(I)
    NEXT I
    Vr = TR + AB(N + 1)
    Vi = TI

END SUB

DEFDBL N
SUB FRAC (N, NR, DR, G(), Nu(), D(), N1(), N2(), N3(), D1(), D2(), D3())

    IF N = 3 THEN
        IND = 1
    ELSE
        IND = 0
    END IF
    DR = 1
    FOR I = N + 1 TO 2 * N
    DR = DR + G(I)
    NEXT I
    IF IND = 1 THEN
        DIM T(3), U(3)
        T(1) = G(1) + G(5) + G(6)
        U(1) = T(1)
        T(2) = G(2) + G(4) + G(6)
        U(2) = T(2)
        T(3) = G(3) + G(4) + G(5)
        U(3) = T(3)
        I = 1
```

```
    MAY = T(I)
    WHILE I <= 3
      FOR J = I + 1 TO 3
      IF T(I) > T(J) THEN
      ELSE
      MAY = T(J)
      T(J) = T(I)
      T(I) = MAY
      END IF
      NEXT J
      I = I + 1
    WEND
    NR = T(1) + 1
    I = 1: J = 1
    WHILE I <= G(4) + 1
      FOR K = 1 TO G(5) + 1
      FOR K1 = 1 TO G(6) + 1
      D(J) = D(J) + D1(I) * D2(K) * D3(K1)
      J = J + 1
      NEXT K1
      J = I + 1
      NEXT K
      I = I + 1
      J = I
    WEND
    II = 1
    H1 = II
    H2 = G(1) + 1
    H3 = G(5) + 1
    H4 = G(6) + 1
    GOSUB 100
    II = II + 1
    H1 = II
    H2 = G(2) + 1
    H3 = G(4) + 1
    H4 = G(6) + 1
    GOSUB 100
    II = II + 1
    H1 = II
    H2 = G(3) + 1
    H3 = G(4) + 1
    H4 = G(5) + 1
    GOSUB 100
ELSE
    A1 = G(1) + G(4)
    A2 = G(2) + G(3)
    IF A1 > A2 THEN
    NR = A1 + 1
    ELSE
    NR = A2 + 1
    END IF
    J = 1
    FOR I = 1 TO G(3) + 1
    FOR K = 1 TO G(4) + 1
    D(J) = D(J) + D1(I) * D2(K)
    J = J + 1
    NEXT K
    J = I + 1
    NEXT I
    J = NR - A1
    J1 = J
    FOR I = 1 TO G(1) + 1
```

```
        FOR K = 1 TO G(4) + 1
        Nu(J) = Nu(J) + N1(I) * D2(K)
        J = J + 1
        NEXT K
        J1 = J1 + 1
        J = J1
        NEXT I
        J = NR - A2
        J1 = J
        FOR I = 1 TO G(2) + 1
        FOR K = 1 TO G(3) + 1
        Nu(J) = Nu(J) + N2(I) * D1(K)
        J = J + 1
        NEXT K
        J1 = J1 + 1
        J = J1
        NEXT I
    END IF
    GOTO 200
'
'SUBRUTINA 100; CALCULO DEL NUMERADOR DE SUMA DE TRES FRACCIONES
'
100   J = T(1) - U(II) + 1
    I = 1
    I1 = J
    I2 = J
    WHILE I <= H2
     FOR K = 1 TO H3
     FOR K1 = 1 TO H4
     IF H1 = 1 THEN
      Nu(J) = Nu(J) + N1(I) * D2(K) * D3(K1)
     ELSEIF H1 = 2 THEN
      Nu(J) = Nu(J) + N2(I) * D1(K) * D3(K1)
     ELSE
      Nu(J) = Nu(J) + N3(I) * D1(K) * D2(K1)
     END IF
     J = J + 1
     NEXT K1
     I1 = I1 + 1
     J = I1
     NEXT K
     I = I + 1
     I2 = I2 + 1
     I1 = I2
     J = I2
    WEND
    RETURN


200   END SUB

DEFINT N
SUB MBIN (N1, BE(), PS())

'
'    Comienza calculo de polinomio
'
    DIM PS1(N1 + 1)

    IF N1 <= 1 THEN
      FOR I = 1 TO N1
      PS(I + 1) = BE(I)
```

```
      NEXT I
      PS(1) = 1
    ELSE

    PS(N1 + 1) = BE(N1) * BE(N1 - 1)
    PS1(N1 + 1) = PS(N1 + 1)
    PS(N1) = BE(N1) + BE(N1 - 1)
    PS1(N1) = PS(N1)
    PS(N1 - 1) = 1
    PS1(N1 - 1) = 1
    I = N1 - 2
    WHILE I >= 1
      FOR J = 1 TO N1
      PS(J) = PS1(J + 1)
      NEXT J
      PS(N1 + 1) = PS(N1 + 1) * BE(I)
      FOR J = I + 1 TO N1
      PS(J) = PS(J) + PS1(J) * BE(I)
      NEXT J
      FOR J = I TO N1 + 1
      PS1(J) = PS(J)
      NEXT J
      I = I - 1
    WEND
    END IF

END SUB

SUB MINV (N, AI(), CI(), F1)
'
'     AI.- ENTRADA
'     CI.- SALIDA
'

    DIM BI(N, N), Pi(N)
    F1 = 0
    FOR K1 = 1 TO N
    FOR K2 = 1 TO N
    BI(K1, K2) = AI(K1, K2)
    NEXT K2, K1

    FOR K = 1 TO N - 1
    T = 0
    FOR I = 1 TO N
    T = T + BI(I, I)
    NEXT I
    Pi(K) = T / K
    FOR I = 1 TO N
    BI(I, I) = BI(I, I) - Pi(K)
    NEXT I
    FOR I = 1 TO N
    FOR J = 1 TO N
    CI(I, J) = BI(I, J)
    NEXT J, I
    FOR I = 1 TO N
    FOR J = 1 TO N
    BI(I, J) = 0
    FOR K1 = 1 TO N
    BI(I, J) = BI(I, J) + AI(I, K1) * CI(K1, J)
    NEXT K1, J, I, K
    Pi(N) = BI(1, 1)
    IF ABS(Pi(N)) < 1E-08 THEN 3000
```

```
    FOR I = 1 TO N
    FOR J = 1 TO N
    CI(I, J) = CI(I, J) / Pi(N)
    NEXT J, I
    GOTO 101

3000  CLS
    BEEP
    PRINT " NO EXISTE INVERSA DE LA MATRIZ QUE SE PROCESA"
    PRINT
    PRINT " EL SISTEMA NO PUEDE SER RESUELTO POR ESTE METODO"
    PRINT
    PRINT " POR FAVOR, ESCOJA OTRO METODO DE REDUCCION"
    PRINT : PRINT : PRINT
    PRINT " Presione cualquier tecla para continuar"
102   IF INKEY$ = "" THEN 102
    F1 = 1
    FOR I = 1 TO N
    FOR J = 1 TO N
    CI(I, J) = 0
    NEXT J, I

101 '


END SUB

SUB MPINV (N, M, A1(), AI1())
'
'    Ingreso:
'    N, M.- Orden de la matriz : N filas x M columnas
'    A1.- Matriz de entrada
'    AI1.- Matriz de salida
'

    DIM AT1(M, N), AAT1(M, M), AATI1(M, M), AI(M, N)

    FOR I = 1 TO N
    FOR J = 1 TO M
    AT1(J, I) = A1(I, J)
    NEXT J, I

    IF N > M THEN

    N1 = M
    M1 = M
    CALL MULTM(N, N1, M1, AAT1(), AT1(), A1())
    CALL MINV(M, AAT1(), AATI1(), Z)
    IF Z = 1 THEN GOTO MENSAJE

    N1 = M
    M1 = N
    CALL MULTM(M, N1, M1, AI1(), AATI1(), AT1())

    ELSEIF M > N THEN

    REDIM AAT1(N, N), AATI1(N, N)
    N1 = N
    M1 = N
    CALL MULTM(M, N1, M1, AAT1(), A1(), AT1())
    CALL MINV(N, AAT1(), AATI1(), Z)
    IF Z = 1 THEN GOTO MENSAJE
```

```
    N1 = M
    M1 = N
    CALL MULTM(N, N1, M1, AI1(), AT1(), AATI1())

    ELSE

    CLS
    PRINT "UTILICE EL PROGRAMA DE MATRIZ INVERSA PARA HALLAR"
    PRINT "LA MATRIZ INVERSA"

    END IF

MENSAJE:
    CLS
    PRINT "NO ES POSIBLE HALLAR LA MATRIZ PSEUDO-INVERSA"


END SUB

SUB MULTM (N, N1, M1, Vr(), VM1(), VM2())

    FOR I = 1 TO N1
    FOR J = 1 TO M1
    TEMPO = 0
    FOR K = 1 TO N
    TEMPO = TEMPO + (VM1(I, K) * VM2(K, J))
    NEXT K
    Vr(I, J) = TEMPO
    NEXT J, I

END SUB

SUB QRAIZ (N, T, F(), A(), Z)

'
'    F.- Matriz de entrada en tiempo discreto
'    A.- Matriz de salida en tiempo continuo
'
10   CLS
    PRINT "Calculo del sistema continuo usando el metodo de la q-esima raiz"
    PRINT : PRINT
    INPUT "INGRESE ORDEN DE LA RAIZ (Q > 2): Q= "; Q
    INPUT "INGRESE LA TOLERANCIA (E < 1): E= "; E
    IF Q < 2 THEN 10
    IF E >= 1 THEN 10
    Z = 0

    DIM X(N, N, Q), Y(N, N, Q), RQ(N, N, Q), R(N, N)
'
'    Inicializacion
'
    FOR I = 1 TO N
    FOR J = 1 TO N
    X(I, J, 1) = 0
    NEXT J, I
    FOR I = 1 TO N
    X(I, I, 1) = 1
    NEXT I
    FOR K = 2 TO N
    FOR I = 1 TO N
    FOR J = 1 TO N
```

```
X(I, J, K) = 0
NEXT J, I, K
FOR I = 1 TO N
FOR J = 1 TO N
R(I, J) = 0
NEXT J, I

' Computo de la q-esima raiz principal

D = 1
DIM X1(N, N), X2(N, N), YQ(N, N), YF(N, N), XI(N, N), RE(N, N), X12(N, N)

ICONT = 1
CLS
PRINT "ESPERE.  CALCULANDO EL SISTEMA CONTINUO"

DO UNTIL D <= E

  FOR I = 1 TO Q
  FOR J = 1 TO N
  FOR K = 1 TO N
  Y(J, K, I) = X(J, K, I)
  NEXT K, J, I
  FOR I = 2 TO Q
  FOR J = 1 TO N
  FOR K = 1 TO N
  X(J, K, I) = Y(J, K, I - 1) + Y(J, K, I)
  NEXT K, J, I
  FOR I = 1 TO N
  FOR J = 1 TO N
  YQ(I, J) = Y(I, J, Q)
  NEXT J, I


  N1 = N
  M1 = N

  CALL MULTM(N, N1, M1, YF(), F(), YQ())

  FOR I = 1 TO N
  FOR J = 1 TO N
  X(I, J, 1) = Y(I, J, 1) + YF(I, J)
  NEXT J, I

  FOR I = 1 TO N
  FOR J = 1 TO N
  X1(I, J) = X(I, J, 1)
  X2(I, J) = X(I, J, 2)
  NEXT J, I

  CALL MINV(N, X2(), XI(), Z)
  IF Z = 1 THEN GOTO FINAL1

  CALL MULTM(N, N1, M1, X12(), X1(), XI())


  FOR I = 1 TO N
  FOR J = 1 TO N
  RE(I, J) = R(I, J) - X12(I, J)
  NEXT J, I
```

```
D = 0
FOR I = 1 TO N
FOR J = 1 TO N
D = D + ABS(RE(I, J))
NEXT J, I

IF ICONT = 1 THEN
  DANT = D
  IC = 1
ELSE
END IF
IF D > DANT THEN
   IC = IC + 1
   IF IC > 6 THEN
     Z = 1
     GOTO 1
   ELSE
     IC = 1
   END IF
ELSE
END IF

LOCATE 5, 5
PRINT "ERROR= ";
PRINT USING "###.##########"; D
LOCATE 5, 30
PRINT "E = "; E
LOCATE 5, 50
PRINT ICONT

IF D > E THEN
  FOR I = 1 TO N
  FOR J = 1 TO N
  R(I, J) = X12(I, J)
  NEXT J, I
ELSE
END IF
ICONT = ICONT + 1
LOOP


DIM RT(N, N), RAUX3(N, N), RAUX1(N, N), RAUX2(N, N)

FOR I = 1 TO N
FOR J = 1 TO N
RAUX1(I, J) = R(I, J)
RAUX2(I, J) = R(I, J)
NEXT J, I
FOR I = 1 TO N
RAUX1(I, I) = RAUX1(I, I) - 1
NEXT I
FOR I = 1 TO N
RAUX2(I, I) = RAUX2(I, I) + 1
NEXT I
CALL MINV(N, RAUX2(), RAUX3(), Z)
IF Z = 1 THEN GOTO FINAL1
CALL MULTM(N, N1, M1, RT(), RAUX1(), RAUX3())


FOR I = 1 TO N
FOR J = 1 TO N
RAUX1(I, J) = RT(I, J)
```

```
    RAUX2(I, J) = 0
    RAUX3(I, J) = 0
    A(I, J) = RT(I, J)
    NEXT J, I

11   CLS
    PRINT "INGRESE EL NUMERO DE APROXIMACIONES PARA EL CALCULO DE MATRIZ A"
    PRINT
    PRINT "Con"; Q - 1; " aproximaciones se tiene un buen resultado"
    PRINT
    PRINT "No. DE APROXIMACIONES = ";
    INPUT IAPROX
    IF IAPROX < (Q - 1) THEN 11

    K = 2
    WHILE K <= IAPROX
      CALL MULTM(N, N1, M1, RAUX2(), RT(), RAUX1())
      P1 = K / 2
      P2 = INT(P1)
      P3 = P1 - P2
      IF P3 = 0 THEN
      ELSE
        FOR I = 1 TO N
        FOR J = 1 TO N
        RAUX2(I, J) = RAUX2(I, J) / K
        NEXT J, I
        FOR I = 1 TO N
        FOR J = 1 TO N
        A(I, J) = A(I, J) + RAUX2(I, J)
        NEXT J, I
      END IF
      FOR I = 1 TO N
      FOR J = 1 TO N
      RAUX1(I, J) = RAUX2(I, J)
      NEXT J, I
      K = K + 1
    WEND

    FOR I = 1 TO N
    FOR J = 1 TO N
    A(I, J) = A(I, J) * 2 * Q / T
    NEXT J, I
    GOTO 1

FINAL1:
    BEEP
    PRINT "NO HAY MATRIZ INVERSA"

1
END SUB

SUB RITD (N, M, NR, T, A(), B(), Q(), QOR(), QOI(), QR11(), QI11(), QR12(), QI12(), G())

'   Comienza division de polinomios

    IF M >= N THEN
      P = M - N
      DIM FAC1(P + 1)
      I = 1
      WHILE M >= N
       FC = A(1) / B(1)
       FOR J = 1 TO N + 1
```

```
      A(J) = A(J) - B(J) * FC
      NEXT J
      FAC1(I) = FC
      REDIM AA(M + 1)
      FOR J = 1 TO M + 1
      AA(J) = A(J)
      NEXT J
      M = M - 1
      REDIM A(M + 1)
      FOR J = 1 TO M + 1
      A(J) = AA(J + 1)
      NEXT J
      I = I + 1
     WEND
    ELSE
    END IF
'   Termina division de polinomios

INICIO:
    REDIM RD1(N), RDI1(N), RN1(M), RNI1(M), RDR(N), RDI(N), RSR(N), RSI(N)
    REDIM BD(N), BDD(N - 1)
    REDIM AUX(N)

      CALL ROOT(N, B(), RD1(), RDI1())
      CALL DIFP(N, B(), BD())
      ND = N - 1
      CALL DIFP(ND, BD(), BDD())
      NDD = ND - 1
    I = 1
    K = 1
    SW = 0
    KA = 1
    KB = 1
    WHILE I <= N
      FOR II = KB TO KA
      IF I = AUX(II) THEN
        SW = 1
      ELSE
      END IF
      NEXT II
      IF SW = 1 THEN
        IF I < AUX(KB) THEN
        ELSE
          KB = KB + 1
        END IF
      ELSE
        J = I + 1
        WHILE J <= N
          IF RD1(I) = RD1(J) AND RDI1(I) = RDI1(J) THEN
            AUX(KA) = J
            KA = KA + 1
            J = N + 1
          ELSE
            IF J = N THEN
              IF (I - 1) = AUX(KA - 1) AND I > 1 THEN
                CLS
                PRINT : PRINT " 1  EXISTE UN POLO TRIPLE"
                PRINT : PRINT "EL SISTEMA NO PUEDE SER RESUELTO"
                PRINT : PRINT "POR FAVOR, ESCOJA UN MODELO REDUCIDO DE DIFERENTE ORDEN"
                GOTO 14
              ELSE
                RSR(K) = RD1(I)
```

```
            RSI(K) = RDI1(I)
            K = K + 1
          END IF
        ELSE
        END IF
        J = J + 1
      END IF
    WEND
  END IF
  I = I + 1
  SW = 0
WEND
IF RD1(N - 1) = RD1(N) AND RDI1(N - 1) = RDI1(N) THEN
  IF (I - 2) = AUX(KA - 1) THEN
    CLS
    PRINT : PRINT " 2  EXISTE UN POLO TRIPLE"
    PRINT : PRINT "EL SISTEMA NO PUEDE SER RESUELTO"
    PRINT : PRINT "POR FAVOR, ESCOJA UN MODELO REDUCIDO DE DIFERENTE ORDEN"
    GOTO 14
  ELSE
  END IF
ELSE
  RSR(K) = RD1(N)
  RSI(K) = RDI1(N)
  K = K + 1
END IF
FOR I = 1 TO KA - 1
RDR(I) = RD1(AUX(I))
RDI(I) = RDI1(AUX(I))
NEXT I
I = 1
IQ = 1
IOQ = 1

WHILE I <= K - 1
    Pr = RSR(I)
    Pi = RSI(I)

    CALL EVALPC(M, A(), Pr, Pi, Vr, Vi)
    V1R = Vr
    V1I = Vi

    CALL EVALPC(ND, BD(), Pr, Pi, Vr, Vi)
    V2R = Vr
    V2I = Vi
    IF RSI(I) = 0 THEN
     Q(IQ) = V1R / V2R
     IQ = IQ + 1
    ELSE
     QOR(IOQ) = (V1R * V2R + V1I * V2I) / (V2R ^ 2 + V2I ^ 2)
     QOI(IOQ) = (V1I * V2R - V1R * V2I) / (V2R ^ 2 + V2I ^ 2)
     IOQ = IOQ + 1
    END IF
   I = I + 1
WEND
I = 1
I11 = 1
I12 = 1
WHILE I <= KA - 1
     Pr = RDR(I)
     Pi = RDI(I)
     CALL EVALPC(M, A(), Pr, Pi, Vr, Vi)
```

```
    V1R = Vr * 2
    V1I = Vi * 2

    CALL EVALPC(NDD, BDD(), Pr, Pi, Vr, Vi)
    V2R = Vr
    V2I = Vi
    QR12(I12) = (V1R * V2R + V1I * V2I) / (V2R ^ 2 + V2I ^ 2)
    QI12(I12) = (V1I * V2R - V1R * V2I) / (V2R ^ 2 + V2I ^ 2)

    CALL ROOT(M, A(), RN1(), RNI1())
    ZR1 = 0: ZI1 = 0
    IF M = 0 THEN
      ZR1 = 0
      ZI1 = 0
    ELSE
      FOR JJ = 1 TO M
      Z1 = RDR(I) - RD1(JJ)
      Z2 = RDI(I) - RDI1(JJ)
      ZR1 = ZR1 + (Z1 / (Z1 ^ 2 + Z2 ^ 2))
      ZI1 = ZI1 - (Z2 / (Z1 ^ 2 + Z2 ^ 2))
      NEXT JJ
    END IF
    ZR2 = 0: ZI2 = 0
    IF K = 1 THEN
      ZR2 = 0
      ZI2 = 0
    ELSE
      FOR JJ = 1 TO K - 1
      Z1 = RDR(I) - RSR(JJ)
      Z2 = RDI(I) - RSI(JJ)
      ZR2 = ZR2 + (Z1 / (Z1 ^ 2 + Z2 ^ 2))
      ZI2 = ZI2 - (Z2 / (Z1 ^ 2 + Z2 ^ 2))
      NEXT JJ
    END IF
    QR11(I11) = QR12(I12) * (ZR1 - ZR2)
    QI11(I11) = QI12(I12) * (ZI1 - ZI2)
    I12 = I12 + 1
    I11 = I11 + 1
    IF RDI(I) = 0 THEN
      I = I + 1
    ELSE
      I = I + 2
    END IF
WEND

CLS
PRINT "LA FUNCION EN EL TIEMPO SERA:"
PRINT
PRINT
I = 1
WHILE I <= K - 1
  IF RSI(I) <> 0 THEN
    PRINT "2EXP("; RSR(I); ")t*("; QOR(I); "COS("; RSI(I); ")t - "; QOI(I); "SIN("; RSI(I); ")t)"
'     PRINT "2EXP(";
'     PRINT USING ".###"; RSR(I);
'     PRINT ")t*(";
'     PRINT USING ".###"; QOR(I);
'     PRINT "COS(";
'     PRINT USING ".###"; RSI(I);
'     PRINT ")t - ";
'     PRINT USING ".###"; QOI(I);
'     PRINT "SIN(";
```

```
'          PRINT USING ".###"; RSI(I);
'          PRINT ")t)"


         I = I + 2
       ELSE
         PRINT Q(I);
         PRINT "EXP(";
         PRINT RSR(I);
         PRINT ")t"
         I = I + 1
       END IF
     WEND
     I = 1
     WHILE I <= KA - 1
       IF RDI(I) = 0 THEN
         PRINT QR11(I); "EXP("; RDR(I); "t) + "; QR12(I); "t*EXT("; RDR(I); "t)"
'          PRINT USING ".###"; QR11(I);
'          PRINT "EXP(";
'          PRINT USING ".###"; RDR(I);
'          PRINT "t) + ";
'          PRINT USING ".###"; QR12(I);
'          PRINT "t*EXT(";
'          PRINT USING ".###"; RDR(I);
'          PRINT "t)"

         I = I + 1
       ELSE
         PRINT "2EXP("; RDR(I); "t)*("; QR11(I); "COS("; RDI(I); "t) + "; QI11(I); "SIN("; RDI(I); "t) + "; QR12(I);
"tCOS("; RDI(I); "t) + "; QI12(I); "tSIN("; RDI(I); "t))"
'          PRINT "2EXP(";
'          PRINT USING ".###"; RDR(I);
'          PRINT "t)*(";
'          PRINT USING ".###"; QR11(I);
'          PRINT "COS(";
'          PRINT USING ".###"; RDI(I);
'          PRINT "t) + ";
'          PRINT USING ".###"; QI11(I);
'          PRINT "SIN(";
'          PRINT USING ".###"; RDI(I);
'          PRINT "t) + ";
'          PRINT USING ".###"; QR12(I);
'          PRINT "tCOS(";
'          PRINT USING ".###"; RDI(I);
'          PRINT "t) + ";
'          PRINT USING ".###"; QI12(I);
'          PRINT "tSIN(";
'          PRINT USING ".###"; RDI(I);
'          PRINT "t))"

         I = I + 2
       END IF
     WEND
     PRINT : PRINT : PRINT
     PRINT "Presione cualquier tecla para continuar"
13     IF INKEY$ = "" THEN 13

'
'    Comienza obtencion de los valores discretos
'
```

```
      FOR I = 1 TO 2 * NR
      G(I) = 0
      NEXT I
      VT = T
      T = -T
      FOR JJ = 1 TO 2 * NR + 1
      T = T + VT
      I = 1
      WHILE I <= K - 1
        IF RSI(I) <> 0 THEN
          G(JJ) = G(JJ) + (2 * EXP(RSR(I) * T) * (QOR(I) * COS(RSI(I) * T) + QOI(I) * SIN(RSI(I) * T)))
          I = I + 2
        ELSE
          G(JJ) = G(JJ) + (Q(I) * EXP(RSR(I) * T))
          I = I + 1
        END IF
      WEND
      I = 1
      WHILE I <= KA - 1
        IF RDI(I) = 0 THEN
          G(JJ) = G(JJ) + (QR11(I) * EXP(RDR(I) * T) + QR12(I) * T * EXP(RDR(I) * T))
          I = I + 1
        ELSE
          G(JJ) = G(JJ) + (2 * EXP(RDR(I) * T) * (QR11(I) * COS(RDI(I) * T) + QI11(I) * SIN(RDI(I) * T) + QR12(I) * T *
COS(RDI(I) * T) + QI12(I) * T * SIN(RDI(I) * T)))
          I = I + 2
        END IF
      WEND
      NEXT JJ


14

END SUB

SUB ROOT (NG, A(), R1(), RI1())
    DIM U2(30), V2(30)
    DIM A1(31), C(31)
    CLS
    LOCATE 9, 22
    PRINT "CALCULANDO RAICES;  POR FAVOR ESPERE"
    GOSUB 1550
    REM
    REM***VERIFICACION EN U2 + jV2***
    REM
    FOR J = 1 TO NG
    U2(J) = A(1)
    V2(J) = 0
    FOR J1 = 1 TO NG
    T = U2(J) * R1(J) - V2(J) * RI1(J) + A(J1 + 1)
    V2(J) = V2(J) * R1(J) + U2(J) * RI1(J)
    U2(J) = T
    NEXT J1, J
    FOR J = 1 TO NG
    U2(J) = SQR(U2(J) ^ 2 + V2(J) ^ 2)
    NEXT J
    GOTO 20
1550  REM
    REM SUBRUTINA PARA SOLUCION DE ECUACIONES POLINOMIALES
    REM METODO: DESCENSO MAS PRONUNCIADO EN ESCALAMIENTO DE RAICES
    REM
    REM    NG - GRADO DEL POLINOMIO
```

```
    REM    A - VECTOR CON COEFICIENTES EN ORDEN DESCENDENTE DE
    REM      POTENCIAS
    REM
    REM    R1 - VECTOR CON PARTES REALES DE LAS RAICES
    REM    RI1 - VECTOR CON PARTES IMAGINARIAS DE LAS RAICES
    REM
    REM    N,A N0 SON ALTERADOS POR LA SUBRUTINA
    REM
    N0 = NG
    J9 = 0
    E = .0000000001#
    E1 = E ^ 2
    FOR J = 1 TO N0: R1(J) = 0: RI1(J) = 0: NEXT
    FOR J = 1 TO N0
    A1(J) = A(J + 1) / A(1)
    NEXT J
1790  IF N0 > 0 THEN 1810
    RETURN
1810  IF A1(N0) <> 0 THEN 1850
    J9 = J9 + 1
    N0 = N0 - 1
    GOTO 1790
1850  IF N0 <> 1 THEN 1890
    J9 = J9 + 1
    R1(J9) = -A1(N0)
    RETURN
1890  IF N0 <> 2 THEN 2210
    X = -A1(1) / 2
    J9 = J9 + 1
    T = X * X - A1(2)
    IF T < 0 THEN 1980
    T = SQR(T)
    R1(J9) = X + T
    R1(J9 + 1) = X - T
    RETURN
1980  R1(J9) = X
    R1(J9 + 1) = R1(J9)
    RI1(J9) = SQR(-T)
    RI1(J9 + 1) = -RI1(J9)
    RETURN
2030  REM
    REM SUB. EVALUACION DE F(Z) = U + jV
    REM
    U = 1
    V = 0
    FOR J = 1 TO N0
    T = U * X - V * Y + C(J)
    V = V * X + U * Y
    U = T
    NEXT J
    RETURN
2120  REM
    REM SUB. EVALUACION F'(Z) = U1+jV1
    REM
    U1 = N0
    V1 = 0
    FOR J = 1 TO N0 - 1
    T = U1 * X - V1 * Y + (N0 - J) * C(J)
    V1 = V1 * X + U1 * Y
    U1 = T
    NEXT J
    RETURN
```

```
2210  REM
      REM DESCENSO MAS PRONUNCIADO
      REM
      T = ABS(A1(N0))
      IF T = 1 THEN 2310
      H = T ^ (1 / N0)
      T = 1
      FOR J = 1 TO N0
      T = T * H
      C(J) = A1(J) / T
      NEXT J
      GOTO 2330
2310  FOR JJ = 1 TO N0: C(JJ) = A1(JJ): NEXT JJ
      H = 1
2330  X = .7#
      Y = .6#
      GOSUB 2030
      F1 = U * U + V * V
      IF F1 < E1 THEN 2570
2380  GOSUB 2120
      T = U1 * U1 + V1 * V1
      X1 = -(U * U1 + V * V1) / T
      Y1 = (U * V1 - V * U1) / T
2420  X = X + X1
      Y = Y + Y1
      GOSUB 2030
      F2 = U * U + V * V
      IF F2 < E1 THEN 2550
      IF F2 < F1 THEN 2530
      X = X - X1
      Y = Y - Y1
      X1 = .8# * X1
      Y1 = .8# * Y1
      GOTO 2420
2530  F1 = F2
      GOTO 2380
2550  IF ABS(X1) < E AND ABS(Y1) < E THEN 2570
      GOTO 2530
2570  X = X * H
      Y = Y * H
      IF ABS(Y) > E THEN 2610
      Y = 0
2610  J9 = J9 + 1
      R1(J9) = X
      RI1(J9) = Y
      IF Y = 0 THEN 2780
      J9 = J9 + 1
      R1(J9) = X
      RI1(J9) = -Y
      R = -2 * X
      S = X * X + Y * Y
      A1(1) = A1(1) - R
      A1(2) = A1(2) - R * A1(1) - S
      IF N0 < 5 THEN 2760
      FOR J = 3 TO N0 - 2
      A1(J) = A1(J) - R * A1(J - 1) - S * A1(J - 2)
      NEXT J
2760  N0 = N0 - 2
      GOTO 1850
2780  T = 1
      FOR J = 1 TO N0 - 1
      T = T * X + A1(J)
```

```
    A1(J) = T
    NEXT J
    N0 = N0 - 1
    GOTO 1850
20    REM

END SUB

SUB RPTD (N, M, T, NTP, A(), B(), G())
'
'    N.- Grado denominador
'    M.- Grado numerador
'    T.- Periodo de muestreo
'    NTP.-  Numero total de puntos muestreados
'    A().- Polinomio numerador
'    B().- Polinomio denominador
'    G().-  Vector con puntos muestreados

'    Comienza division de polinomios

    IF M >= N THEN
      P = M - N
      DIM FAC1(P + 1)
      I = 1
      WHILE M >= N
       FC = A(1) / B(1)
       FOR J = 1 TO N + 1
       A(J) = A(J) - B(J) * FC
       NEXT J
       FAC1(I) = FC
       REDIM AA(M + 1)
       FOR J = 1 TO M + 1
       AA(J) = A(J)
       NEXT J
       M = M - 1
       REDIM A(M + 1)
       FOR J = 1 TO M + 1
       A(J) = AA(J + 1)
       NEXT J
       I = I + 1
      WEND
    ELSE
    END IF

'    Termina division de polinomios


    N = N + 1
    DIM BT(N)
    FOR I = 1 TO N
    BT(I) = B(I)
    NEXT I
    REDIM B(N + 1)
    FOR I = 1 TO N
    B(I) = BT(I)
    NEXT I
    B(N + 1) = 0

INICIO1:
    REDIM RD1(N), RDI1(N), RN1(M), RNI1(M), RDR(N), RDI(N), RSR(N), RSI(N)
    REDIM BD(N), BDD(N - 1), Q(N), QOR(N), QOI(N), QR11(N), QI11(N), QR12(N), QI12(N)
    REDIM AUX(N)
```

```
  CALL ROOT(N, B(), RD1(), RDI1())
  CALL DIFP(N, B(), BD())
  ND = N - 1
  CALL DIFP(ND, BD(), BDD())
  NDD = ND - 1
I = 1
K = 1
SW = 0
KA = 1
KB = 1
WHILE I <= N
  FOR II = KB TO KA
  IF I = AUX(II) THEN
    SW = 1
  ELSE
  END IF
  NEXT II
  IF SW = 1 THEN
    IF I < AUX(KB) THEN
    ELSE
      KB = KB + 1
    END IF
  ELSE
    J = I + 1
    WHILE J <= N
      IF RD1(I) = RD1(J) AND RDI1(I) = RDI1(J) THEN
        AUX(KA) = J
        KA = KA + 1
        J = N + 1
      ELSE
        IF J = N THEN
          IF (I - 1) = AUX(KA - 1) AND I > 1 THEN
            CLS
            PRINT : PRINT " 1  EXISTE UN POLO TRIPLE"
            PRINT : PRINT "EL SISTEMA NO PUEDE SER RESUELTO"
            PRINT : PRINT "POR FAVOR, ESCOJA UN MODELO REDUCIDO DE DIFERENTE ORDEN"
            GOTO 15
          ELSE
            RSR(K) = RD1(I)
            RSI(K) = RDI1(I)
            K = K + 1
          END IF
        ELSE
        END IF
        J = J + 1
      END IF
    WEND
  END IF
  I = I + 1
  SW = 0
WEND
IF RD1(N - 1) = RD1(N) AND RDI1(N - 1) = RDI1(N) THEN
  IF (I - 2) = AUX(KA - 1) THEN
    CLS
    PRINT : PRINT " 2  EXISTE UN POLO TRIPLE"
    PRINT : PRINT "EL SISTEMA NO PUEDE SER RESUELTO"
    PRINT : PRINT "POR FAVOR, ESCOJA UN MODELO REDUCIDO DE DIFERENTE ORDEN"
    GOTO 15
  ELSE
  END IF
ELSE
```

```
  RSR(K) = RD1(N)
  RSI(K) = RDI1(N)
  K = K + 1
END IF
FOR I = 1 TO KA - 1
RDR(I) = RD1(AUX(I))
RDI(I) = RDI1(AUX(I))
NEXT I
I = 1
IQ = 1
IOQ = 1

WHILE I <= K - 1
    Pr = RSR(I)
    Pi = RSI(I)

    CALL EVALPC(M, A(), Pr, Pi, Vr, Vi)
    V1R = Vr
    V1I = Vi

    CALL EVALPC(ND, BD(), Pr, Pi, Vr, Vi)
    V2R = Vr
    V2I = Vi
    IF RSI(I) = 0 THEN
     Q(IQ) = V1R / V2R
     IQ = IQ + 1
    ELSE
     QOR(IOQ) = (V1R * V2R + V1I * V2I) / (V2R ^ 2 + V2I ^ 2)
     QOI(IOQ) = (V1I * V2R - V1R * V2I) / (V2R ^ 2 + V2I ^ 2)
     IOQ = IOQ + 1
    END IF
    I = I + 1
WEND
I = 1
I11 = 1
I12 = 1
WHILE I <= KA - 1
      Pr = RDR(I)
      Pi = RDI(I)
      CALL EVALPC(M, A(), Pr, Pi, Vr, Vi)
      V1R = Vr * 2
      V1I = Vi * 2

      CALL EVALPC(NDD, BDD(), Pr, Pi, Vr, Vi)
      V2R = Vr
      V2I = Vi
      QR12(I12) = (V1R * V2R + V1I * V2I) / (V2R ^ 2 + V2I ^ 2)
      QI12(I12) = (V1I * V2R - V1R * V2I) / (V2R ^ 2 + V2I ^ 2)

      CALL ROOT(M, A(), RN1(), RNI1())
      ZR1 = 0: ZI1 = 0
      IF M = 0 THEN
        ZR1 = 0
        ZI1 = 0
      ELSE
        FOR JJ = 1 TO M
        Z1 = RDR(I) - RD1(JJ)
        Z2 = RDI(I) - RDI1(JJ)
        ZR1 = ZR1 + (Z1 / (Z1 ^ 2 + Z2 ^ 2))
        ZI1 = ZI1 - (Z2 / (Z1 ^ 2 + Z2 ^ 2))
        NEXT JJ
      END IF
```

```
        ZR2 = 0: ZI2 = 0
        IF K = 1 THEN
          ZR2 = 0
          ZI2 = 0
        ELSE
          FOR JJ = 1 TO K - 1
          Z1 = RDR(I) - RSR(JJ)
          Z2 = RDI(I) - RSI(JJ)
          ZR2 = ZR2 + (Z1 / (Z1 ^ 2 + Z2 ^ 2))
          ZI2 = ZI2 - (Z2 / (Z1 ^ 2 + Z2 ^ 2))
          NEXT JJ
        END IF
        QR11(I11) = QR12(I12) * (ZR1 - ZR2)
        QI11(I11) = QI12(I12) * (ZI1 - ZI2)
        I12 = I12 + 1
        I11 = I11 + 1
        IF RDI(I) = 0 THEN
          I = I + 1
        ELSE
          I = I + 2
        END IF
    WEND

    CLS
    PRINT "LA FUNCION EN EL TIEMPO SERA:"
    PRINT
    PRINT
    I = 1
    WHILE I <= K - 1
      IF RSI(I) <> 0 THEN
        PRINT "2EXP("; RSR(I); ")t*("; QOR(I); "COS("; RSI(I); ")t - "; QOI(I); "SIN("; RSI(I); ")t)"
        I = I + 2
      ELSE
        PRINT Q(I); "EXP("; RSR(I); ")t"
        I = I + 1
      END IF
    WEND
    I = 1
    WHILE I <= KA - 1
      IF RDI(I) = 0 THEN
        PRINT QR11(I); "EXP("; RDR(I); "t) + "; QR12(I); "t*EXT("; RDR(I); "t)"
        I = I + 1
      ELSE
        PRINT "2EXP("; RDR(I); "t)*("; QR11(I); "COS("; RDI(I); "t) + "; QI11(I); "SIN("; RDI(I); "t) + "; QR12(I);
"tCOS("; RDI(I); "t) + "; QI12(I); "tSIN("; RDI(I); "t))"
        I = I + 2
      END IF
    WEND
    PRINT : PRINT : PRINT
    PRINT "Presione cualquier tecla para continuar"
16   IF INKEY$ = "" THEN 16

'
'   Comienza obtencion de los valores discretos
'

      CLS
      LOCATE 12, 12
      PRINT "ESPERE UN MOMENTO..."
      LOCATE 13, 12
      PRINT "CALCULANDO LOS "; NTP + 1; " VALORES DISCRETOS"
```

```
        FOR I = 1 TO NTP
        G(I) = 0
        NEXT I
        VT = T
        T = -T
        FOR JJ = 1 TO NTP + 1

        LOCATE 13, 60
        PRINT JJ

        T = T + VT
        I = 1
        WHILE I <= K - 1
          IF RSI(I) <> 0 THEN
            G(JJ) = G(JJ) + (2 * EXP(RSR(I) * T) * (QOR(I) * COS(RSI(I) * T) + QOI(I) * SIN(RSI(I) * T)))
            I = I + 2
          ELSE
            G(JJ) = G(JJ) + (Q(I) * EXP(RSR(I) * T))
            I = I + 1
          END IF
        WEND
        I = 1
        WHILE I <= KA - 1
          IF RDI(I) = 0 THEN
            G(JJ) = G(JJ) + (QR11(I) * EXP(RDR(I) * T) + QR12(I) * T * EXP(RDR(I) * T))
            I = I + 1
          ELSE
            G(JJ) = G(JJ) + (2 * EXP(RDR(I) * T) * (QR11(I) * COS(RDI(I) * T) + QI11(I) * SIN(RDI(I) * T) + QR12(I) * T *
COS(RDI(I) * T) + QI12(I) * T * SIN(RDI(I) * T)))
            I = I + 2
          END IF
        WEND
        NEXT JJ


15

END SUB

SUB VALP (N, SS(), VPR(), VPI())

'
'     Subrutina para hallar los valores propios de una matriz
'

      REDIM B(2, N + 1), B5(N, N), C5(N, N), P(N), VPR(N), VPI(N)

      FOR I = 1 TO N
      FOR J = 1 TO N
      B5(I, J) = SS(I, J)
      NEXT J, I
      FOR K = 1 TO N - 1
      T = 0
      FOR I = 1 TO N
      T = T + B5(I, I)
      NEXT I
      P(K) = T / K
      FOR I = 1 TO N
      B5(I, I) = B5(I, I) - P(K)
      NEXT I
      FOR I = 1 TO N
      FOR J = 1 TO N
```

```
    C5(I, J) = B5(I, J)
    NEXT J, I
    FOR I = 1 TO N
    FOR J = 1 TO N
    B5(I, J) = 0
    FOR K1 = 1 TO N
    B5(I, J) = B5(I, J) + SS(I, K1) * C5(K1, J)
    NEXT K1, J, I, K
    P(N) = B5(1, 1)
    IF ABS(P(N)) < 1E-08 THEN 1001
    FOR I = 1 TO N
    FOR J = 1 TO N
    C5(I, J) = C5(I, J) / P(N)
    NEXT J, I
1001  REM   ECUAPOL
    REM
    REM SOLUCION DE ECUACIONES POLINOMIALES
    REM
    REM CALCULO DE LAS RAICES DE LA ECUACION CARACTERISTICA
    REM DE LA MATRIZ A
    REM
    DIM P1(31)
    P1(1) = 1
    FOR J = 2 TO N + 1
    P1(J) = -P(J - 1)
    NEXT J

    CALL ROOT(N, P1(), VPR(), VPI())

'
'    Ordenamiento de los valores propios de acuedor al valor absoluto
'    de su parte real, de mayor a menor
'
    REDIM AUX(N), AUXI(N)
    FOR I = 1 TO N
    AUX(I) = VPR(I)
    AUXI(I) = VPI(I)
    NEXT I
    FOR I = 1 TO N - 1
    FOR J = I TO N
    AMAY = ABS(VPR(I))
    IF AMAY < ABS(VPR(J)) THEN
       VAUX = VPR(I)
       VAUI = VPI(I)
       VPR(I) = VPR(J)
       VPI(I) = VPI(J)
       VPR(J) = VAUX
       VPI(J) = VAUI
       AMAY = ABS(VPR(I))
    ELSE
    END IF
    NEXT J, I

END SUB

DEFSNG S
SUB VECP (N, SS(), V())

'
'    Subrutina para hallar los valores y vectores propios
'    A.- Matriz de ingreso
'    V.- Matriz con los vectores propios
```

```
'
      REDIM B(2, N + 1), B5(N, N), C5(N, N), P(N), RD1(N), RDI1(N)

      FOR I = 1 TO N
      FOR J = 1 TO N
      B5(I, J) = SS(I, J)
      NEXT J, I
      FOR K = 1 TO N - 1
      T = 0
      FOR I = 1 TO N
      T = T + B5(I, I)
      NEXT I
      P(K) = T / K
      FOR I = 1 TO N
      B5(I, I) = B5(I, I) - P(K)
      NEXT I
      FOR I = 1 TO N
      FOR J = 1 TO N
      C5(I, J) = B5(I, J)
      NEXT J, I
      FOR I = 1 TO N
      FOR J = 1 TO N
      B5(I, J) = 0
      FOR K1 = 1 TO N
      B5(I, J) = B5(I, J) + SS(I, K1) * C5(K1, J)
      NEXT K1, J, I, K
      P(N) = B5(1, 1)
      IF ABS(P(N)) < 1E-08 THEN 1000
      FOR I = 1 TO N
      FOR J = 1 TO N
      C5(I, J) = C5(I, J) / P(N)
      NEXT J, I
1000  REM   ECUAPOL
      REM
      REM SOLUCION DE ECUACIONES POLINOMIALES
      REM
      REM CALCULO DE LAS RAICES DE LA ECUACION CARACTERISTICA
      REM DE LA MATRIZ A
      REM
      DIM P1(31)
      P1(1) = 1
      FOR J = 2 TO N + 1
      P1(J) = -P(J - 1)
      NEXT J

      CALL ROOT(N, P1(), RD1(), RDI1())

'
'     Ordenamiento de los valores propios de acuedor al valor absoluto
'     de su parte real, de mayor a menor
'
      REDIM AUX(N), AUXI(N)
      FOR I = 1 TO N
      AUX(I) = RD1(I)
      AUXI(I) = RDI1(I)
      NEXT I
      FOR I = 1 TO N - 1
      FOR J = I TO N
      AMAY = ABS(RD1(I))
      IF AMAY < ABS(RD1(J)) THEN
         VAUX = RD1(I)
```

```
        VAUI = RDI1(I)
        RD1(I) = RD1(J)
        RDI1(I) = RDI1(J)
        RD1(J) = VAUX
        RDI1(J) = VAUI
        AMAY = ABS(RD1(I))
      ELSE
      END IF
      NEXT J, I

'
'     Comienza el calculo de vectores propios
'

      VAR1 = RND
      VAR2 = INT(VAR1 * N) + 1
      REDIM A1(N - 1, N - 1), X1(N - 1), X(N - 1)

      K = 1
      WHILE K <= N

      CLS
      LOCATE 9, 16
      PRINT "CALCULANDO VECTOR PROPIO No."; K; "POR FAVOR ESPERE"

        FOR I = 1 TO N - 1
        X(I) = 0
        NEXT I

        FOR I = 1 TO N
        FOR J = 1 TO N
        B5(I, J) = -SS(I, J)
        NEXT J, I
        FOR I = 1 TO N
        B5(I, I) = B5(I, I) + RD1(K)
        NEXT I
        I1 = 1
        FOR I = 1 TO N
        IF I = VAR2 THEN
        ELSE
          FOR J = 1 TO N - 1
          A1(I1, J) = B5(I, J)
          NEXT J
          X1(I1) = -B5(I, N)
          I1 = I1 + 1
        END IF
        NEXT I
        CALL ECUA(N - 1, A1(), X1(), X())
        FOR I = 1 TO N - 1
        V(I, K) = X(I)
        V(N, K) = 1
        NEXT I

        K = K + 1
      WEND

END SUB
```

APENDICE B

Manual del Programa

El programa está realizado en QuickBasic v. 4.0 y consta de 17 módulos ejecutables, los mismos que a continuación se detallan:

- *PFC1F*: Módulo de calcula un sistema reducido usando la Primera Forma de Cauer.
- *SFC2F:* Módulo para el desarrollo de la Segunda Forma de Cauer.
- *TFC3F:* Módulo para el cálculo de un sistema reducido a través de la Tercera Forma de Cauer.
- *FMC4F:* Módulo que desarrolla la Forma Modificada de Cauer.
- *FGC5F:* Módulo para el desarrollo de la Forma General de Cauer.
- *NFGC6F:* Módulo que desarrolla la Nueva Forma Generalizada de Cauer.
- *DCHF:* El presente módulo desarrolla de método de Davison-Chidambara.
- *AGREG-NF:* Módulo que desarrola el cálculo de un sistema reducido usando el método de Agregación.
- *REALPARF:* Módulo que calcula un sistema reducido usando el método de la Realización Parcial.
- *AOF:* Módulo que desarrolla el método de Aproximaciones Optimas.
- *MCF:* Módulo que desarrolla el método de los Mínimos Cuadrados.
- *VSF:* Módulo que calcula un sistema reducido usando el método de los Valores Singulares.
- *MENUPF:* Este módulo presenta un menú principal, en el que se indica la clasificación general de los métodos de reducción desarrollados en la presente tesis, guardando una relación directa con los distintos capítulos en los que se ha divido el trabajo, de acuerdo con la naturaleza misma de cada uno de ellos. Por medio de éste menú se puede ingresar, por ejemplo, a los métodos de modo dominante, para que una vez allí se escoja un método específico de reducción. El menú generado en éste módulo también da la opción de abandonar el programa. La figura B-1 indica la pantalla generada por éste modulo.
- *MENU1F:* Permite escojer cualquiera de los métodos de expansión en fracciones contínuas a través de las distintas formas de Cauer. La figura B-2 indica la pantalla generada por éste módulo.
- *MENU2F:* Permite ingresar a cualquier método de modo dominante, tales como el método de Davison-Chidambara, el método de Agregación y el método de Realización Parcial. La figura B-3 indica la pantalla generada por éste módulo.
- *MENU3F:* Este módulo da la opción de ingresar a los métodos de Aproximaciones Optimas, tales como el método de Aproximaciones Optimas propiamente

dicho y el método de los Mínimos Cuadrados. La figura B-4 muestra la pantalla de presentación del presente módulo.

- *MENU4F:* Mediante éste módulo, el usuario puede escojer el método de los Valores Singulares para determinar un sistema reducido a partir de un sistema de alto orden. A través de la figura B-5 se puede conocer la patalla de presentación generada por éste módulo.

Cabe indicar que se diseñó un módulo más, que contiene todas las subrutinas usadas por cada uno de los módulos nombrados anteriormente. Este módulo de subrutinas, no consta como un archivo ejecutable porque es cargado en la memoria directamente con cada uno de los módulos que usan sus subrutinas.

Para iniciar el programa es conveniente correr siempre el archivo **MENUPF** ya que desde el mismo se puede escojer cualquier método de reducción deseado. Se puede generar un archivo "batch" que ejecute dicho módulo para iniciar el programa.

El programa se lo puede correr directamente desde un diskette de 3.5 pulgadas, pero es preferible instalarlo en el disco duro. Si sólamente se dispone de unidades de 5.25 pulgadas como disketeras del computador, será totalmente necesario instalarlo en el disco duro de la máquina.

A continuación se presentan los gráficos con las pantallas generadas por los módulos de menús:

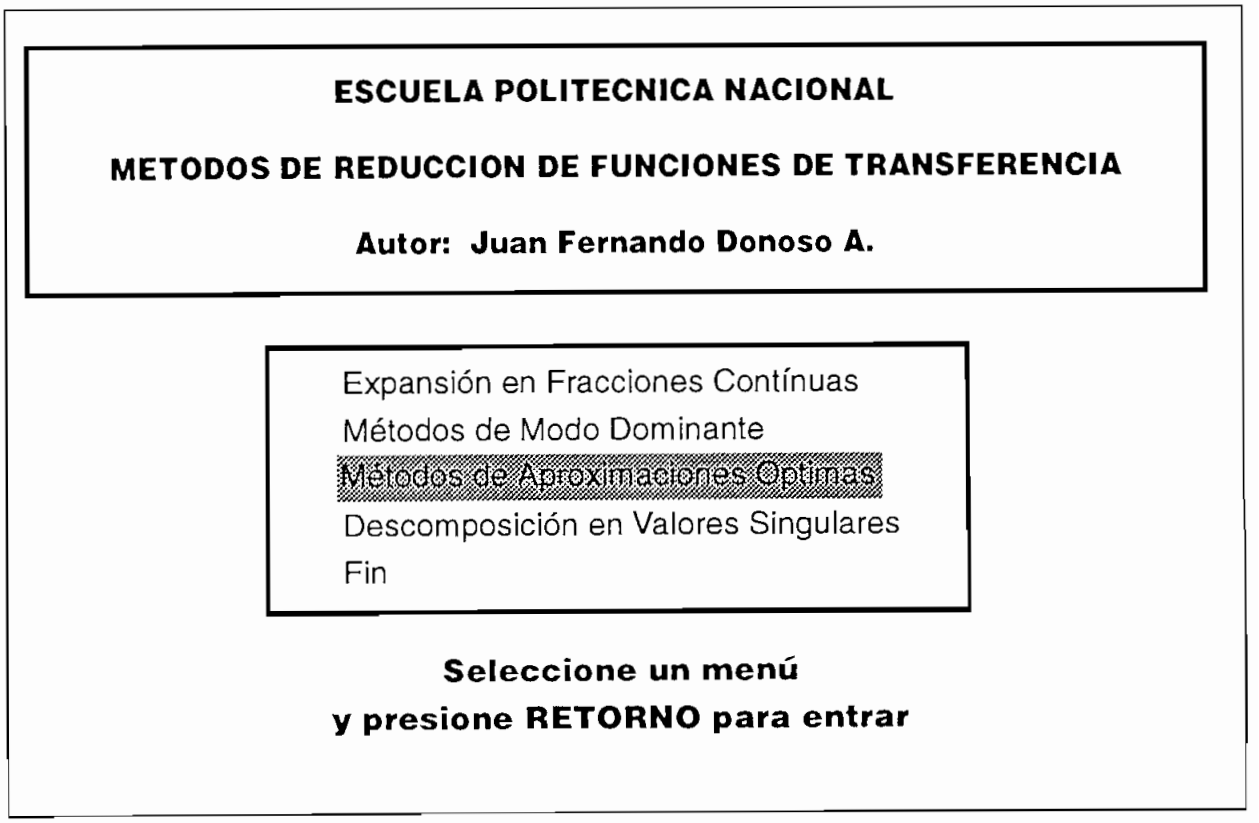


Gráfico B-1

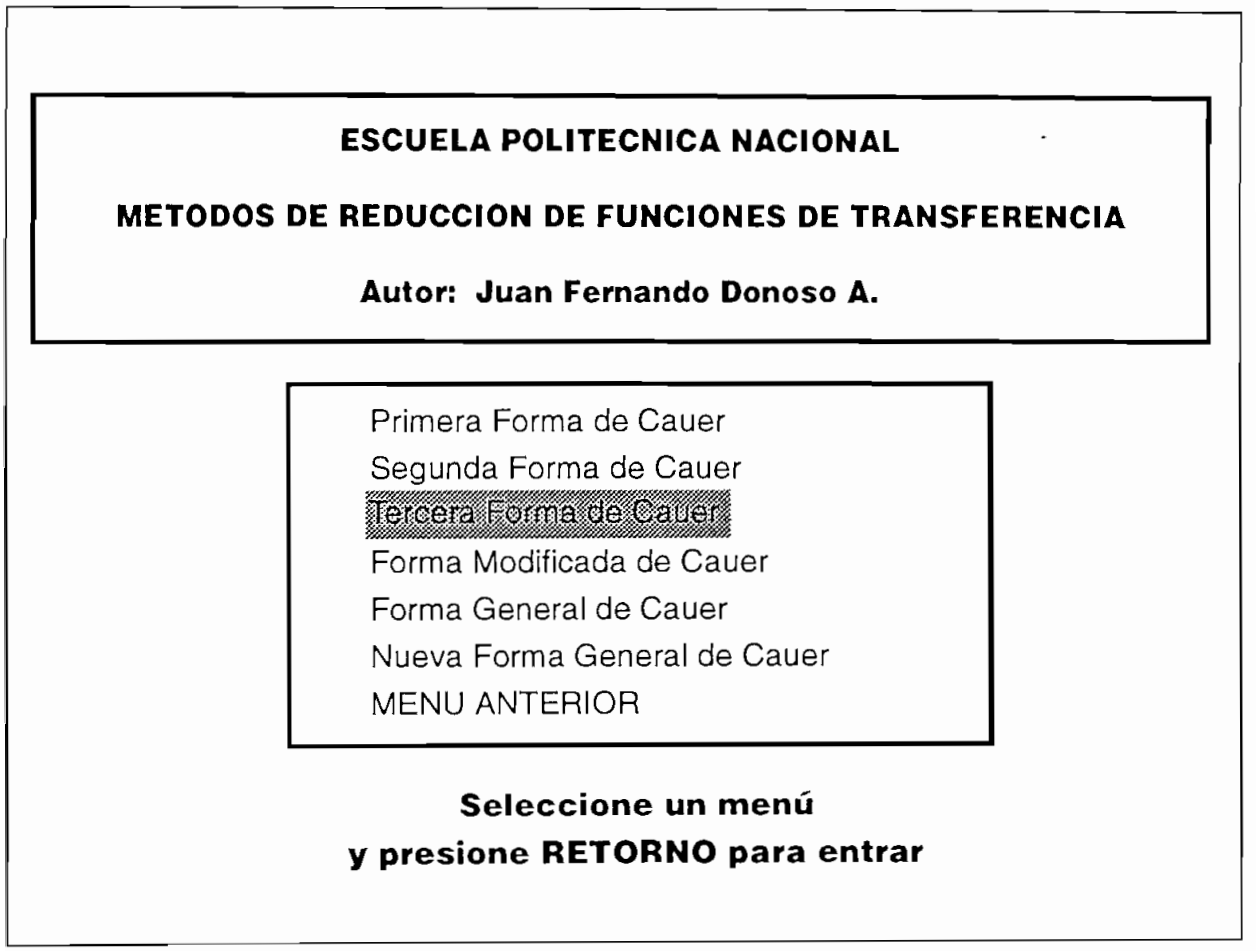

ESCUELA POLITECNICA NACIONAL
METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA
Autor: Juan Fernando Donoso A.
Primera Forma de Cauer
Segunda Forma de Cauer
Tercera Forma de Cauer
Forma Modificada de Cauer
Forma General de Cauer
Nueva Forma General de Cauer
MENU ANTERIOR
Seleccione un menú
y presione RETORNO para entrar

Gráfico B-2

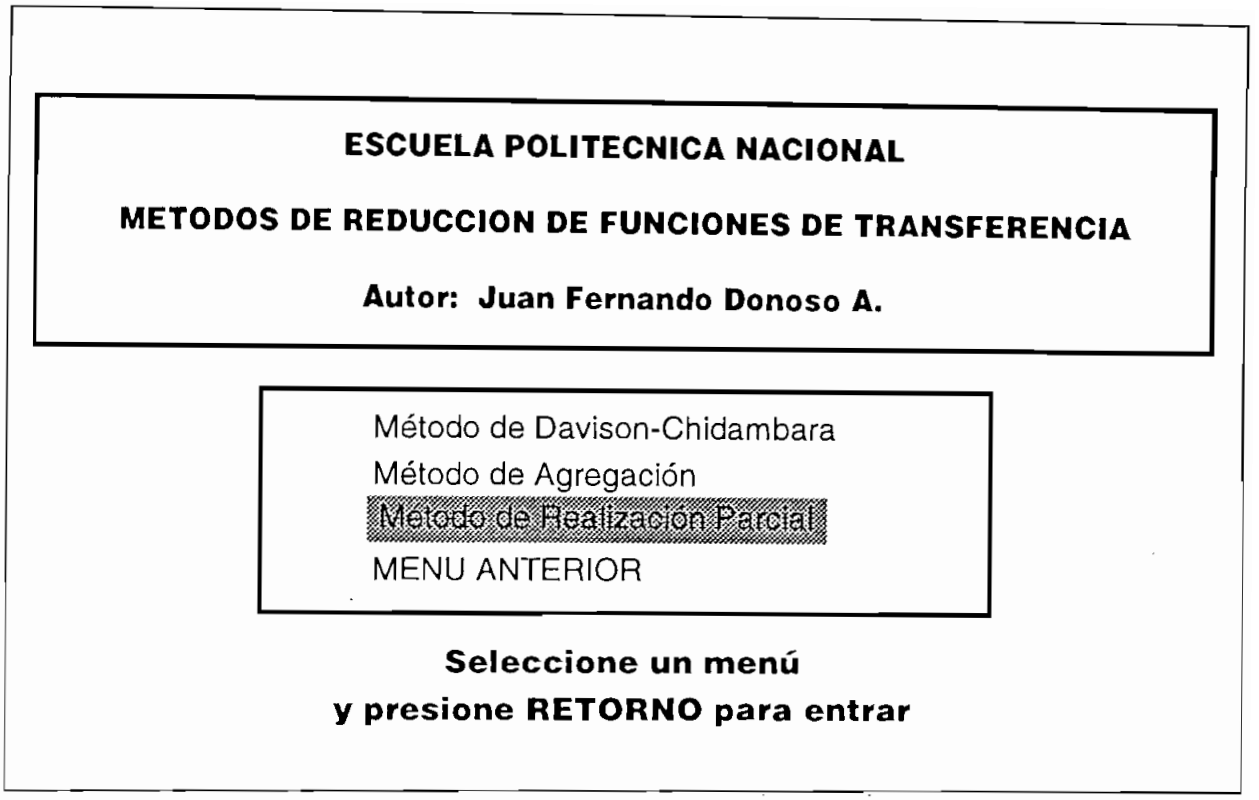


Gráfico B-3

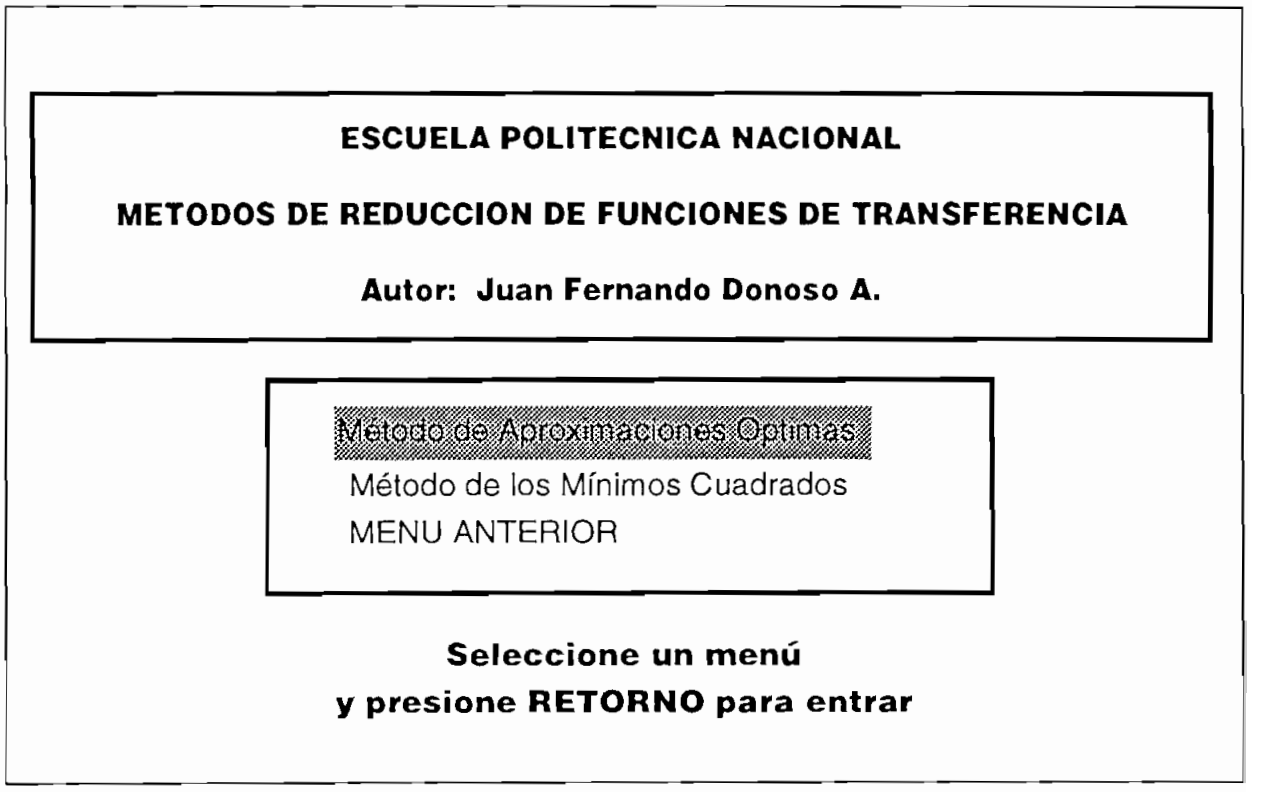


Gráfico B-4

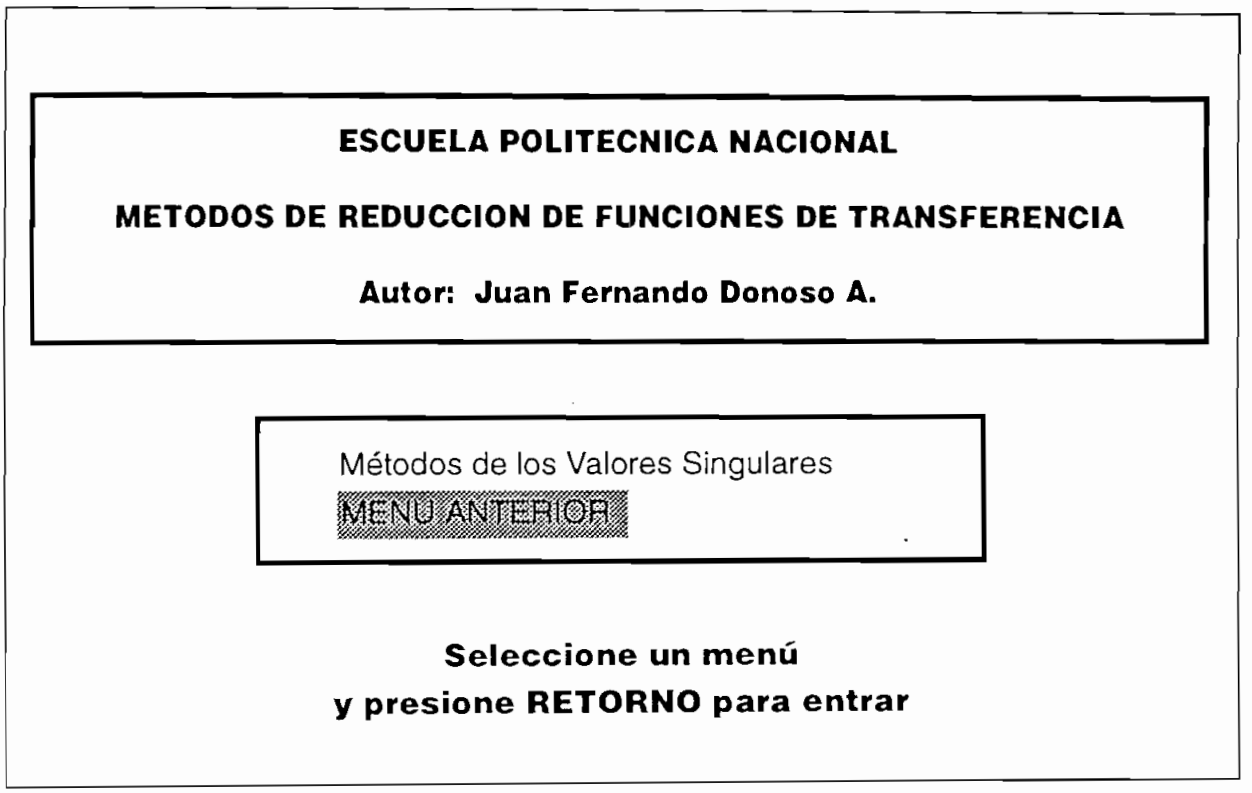

ESCUELA POLITECNICA NACIONAL
METODOS DE REDUCCION DE FUNCIONES DE TRANSFERENCIA
Autor: Juan Fernando Donoso A.
Métodos de los Valores Singulares
MENU ANTERIOR
Seleccione un menú
y presione RETORNO para entrar

Gráfico B-5

APENDICE C

Parámetros de Markov.

Si a una cierta función de transferencia G(s) se la expande en una serie de Laurent alrededor de s=0 se tiene:

$$G(s) = J_0 s^{-1} + J_1 s^{-2} + J_2 s^{-3} + \ldots \qquad \text{(C-1)}$$

o

$$G(s) = \sum_{i=0}^{\infty} J_i s^{-(i+1)} \qquad \text{(C-2)}$$

donde

$$J_i = CA^i B, \quad i = 0,1,2,\ldots \qquad \text{(C-3)}$$

se definen como "parámetros de Markov".

Momentos de Tiempo de una "respuesta impulso"

Es necesario indicar que "respuesta impulso" es la respuesta en el tiempo, de una cierta función de transferencia, a una entrada impulso unitario.

Si G(s) no tiene ningún polo en s=0, entonces su expansión en una serie de Taylor alrededor de s=0 puede ser obtenida como:

$$G(s) = -T_0 s^0 - T_1 s^1 - T_2 s^2 - \ldots \qquad \text{(C-4)}$$

o lo que es lo mismo:

$$G(s) = -\sum_{i=0}^{\infty} T_i s^i \qquad \text{(C-5)}$$

donde:

$$T_i = CA^{-(i+1)}B \qquad (i = 0,1,2,\ldots) \qquad \text{(C-6)}$$

se definen como "momentos de tiempo de una respuesta impulso" del sistema en mención.