T E S I S

MECANISMOS CON ELEMENTOS DE LONGITUD VARIABLE. SINTESIS OPTIMA DE GENERACION DE TRAYECTORIAS.

POR

JAVIER GARCIA-LOMAS JUNG

INGENIERO INDUSTRIAL POR LA E.T.S. DE I.I.
DE LA UNIVERSIDAD DE SEVILLA

PRESENTADA EN LA

ESCUELA TECNICA SUPERIOR DE INGENIEROS INDUSTRIALES
DE LA
UNIVERSIDAD DE SEVILLA

PARA LA OBTENCION DEL

GRADO DE DOCTOR INGENIERO INDUSTRIAL

ENERO, 1982

A Luis y Verena que lo hicieron posible.

Agradecimiento:

Con estas líneas, el autor, desea expresar su más sincero agradecimiento a todas las personas que mediante su incondicional apoyo y constante estímulo han hecho posible este trabajo.

En especial al Profesor D. Jaime Domínguez Abascal, bajo cuya dirección ha sido realizada esta Tesis, por su constante inquietud, estímulo y consejos durante el desarrollo de esta.

También agradece al Profesor D. Justo Nieto Nieto, quien motivó inicialmente al autor a trabajar en el campo de la Síntesis de Mecanismos, su ayuda y recomendaciones.

Igualmente agradece, la colaboración prestada por todos los componentes del Departamento de Mecánica de la E.T.S. de I. Industriales de Sevilla.

Sevilla, Enero 1.982

I N D I C E

B.- Aproximación con spline cúbicos
C.- Listado del programa OPRIS
D.- Listado del programa CIOPT
E.- Listado del programa ACUSC
F.- Listado del programa PERSP
G.- Listado del programa CHECU
H.- Listado de las aplicaciones

CAPITULO I.

INTRODUCCION Y ANTECEDENTES.

1.1 Introducción.
1.2 Propósito.
1.3 Antecedentes.
1.4 Estado actual.

1.1 INTRODUCCION.

La constante y rápida evolución de la tecnología actual demanda nuevas máquinas con un creciente número de exigencias. Estas exigencias irán desde una mayor rentabilidad y productividad a una mayor exactitud en las operaciones, pasando por una gran fiabilidad, un mantenimiento -- mínimo, etc.

También en muchos casos se desearán máquinas capaces de realizar funciones y operaciones no previstas - hasta ese momento como posibles de mecanizar. Esta situa-- ción determinará la concepción y diseño de nuevas máquinas, en la mayoría de los casos mas sofisticadas y complejas que las anteriores.

Tanto si lo que el diseñador pretende es mejorar una máquina ya existente como concebir y diseñar unas nuevas necesitará manejar, analizar, diseñar, etc, nuevos mecanismos mas versatiles y complejos. Por tanto los métodos de diseño de mecanismos habrán de evolucionar constantemente para adaptarse a las nuevas necesidades.

1.2 PROPOSITO.

Las investigaciones en el campo de la teoría de máquinas y mecanismos se dirigen en la actualidad hacia todos los frentes posibles. Ahora bien gran parte de los esfuerzos de los esfuerzos de los investigadores están di-

rigidos al desarrollo y concepción de robots y manipuladores industriales que constituyen elementos básicos en la creciente automatización de los procesos productivos.

Segun Tesar (1976) el futuro de los robots y manipuladores dependerá completamente de la obtención de una base eficaz de cálculo. De igual forma Seireg (1976) afirma que si se desean obtener nuevos diseños de robots y manipuladores, las investigaciones en este campo deberán estar orientadas al desarrollo de técnicas eficientes para la síntesis óptima de trayectorias y el control de mecanismos bajo diversos y cambiantes movimientos.

De acuerdo con esto el presente trabajo tiene como proposito fundamental, presentar la utilización de las técnicas de programación matemática (concretamente la de programación no lineal) tambien conocidas con el nombre de técnicas de optimización, en la sintesis de mecanismos capaces de generar una trayectoria, lo mas aproximada posible en todos sus puntos, a una trayectoria determinada. En esta síntesis óptima se considerará la posibilidad de que alguno de los elementos que forman el mecanismo, ya sea plano o espacial, tengan su longitud variable de acuerdo con una función del tiempo.

La síntesis será dimensional, es decir a partir de un mecanismo ya especificado se pretende determinar las coordenadas de los puntos fijos, el tamaño de los elementos de dimensión constante, los parámetros que definen las leyes de variación de longitud de los elementos de dimensión variable y demas valores necesarios para determinar completamente el mecanismo.

Como la mayoría de los algoritmos de progra-- mación no lineal son bastantes sensibles a la solución de partida o vector de diseño inicial, la sintesis óptima se realiza en dos etapas. En la primera se tratará de encon-- trar un conjunto de posibles soluciones o vectores de diseño a la sintesis propuesta. En esta primera etapa solo se tomarán algunos puntos de la trayectoria o curva de acopla dor deseada, en los cuales, habrán de coincidir las trayec torias generadas por las distintas soluciones que se bus-- can como vectores de diseño iniciales. En la mayoria de los casos esta etapa constituye una síntesis exacta con puntos de precisión.

La segunda etapa o fase, efectuará la búsqueda de la mejor solución a la síntesis propuesta utilizando -- como vectores de diseño iniciales los obtenidos por la fa- se anterior. Es decir, se busca el vector de estado que re- presenta al mecanismo que mejor ajusta en todos sus puntos la trayectoria deseada, constituyendo una sintesis aproxi- mada.

Las técnicas de programación no lineal se uti- lizarán por tanto en dos formas:

1- Búsqueda de posibles soluciones aproximadas a una síntesis propuesta.

2- Determinación de las solución aproximada -- que mejor satisface las condiciones de diseño a partir de un conjunto de soluciones aproximadas posibles.

1.3 ANTECEDENTES.

El área de la síntesis óptima de mecanismos se puede contemplar desde diversos aspectos. Uno de ellos es el desarrollo e implementación de los métods de optimización al diseño de mecanismos. El interés en este campo ha ido creciendo de una forma un tanto espectacular, y se puede decir que durante la última década es cuando se han hecho los grandes avances en esta materia.

Ahora bien la idea de optimización del diseño no es nueva pues se atribuyen a Da Vinci (1452-1519) y a Newton (1642-1727) los primeros trabajos de diseño mecá-- nicos desde el prisma de la optimización.

Uno de los pilares fundamentales en el desa-- rrollo de las técnicas modernas de optimización, fue el desarrollo del calculo diferencial. Su uso suponía una forma mas elegante de obtener el máximo o el mínimo de una fun-- ción diferenciable, no siendo necesario el cálculo repetitivo de la función.

Sin embargo lo que ha tenido una influencia decisiva en la optimización ha sido la aparición y desarrollo de los ordenadores digitales. El nacimiento de la optimi-- zación actual puede atribuirse al trabajo en programación lineal realizado por Dantzig (1949) en el que se incluía el desarrollo del método Simplex.

Aunque hay algunos intentos anteriores, la base de las modernas técnicas de síntesis se encuentra en los trabajos realizados por Freudenstein a finales de los años

50. Freudenstein y Sandor (1959), utilizando la teoría de los números complejos con un ordenador realizan la síntesis de mecanismos para la generación de una trayectoria.

Las técnicas de optimización, que imponen el uso de métodos numéricos, se desarrollaron inicialmente para resolver problemas de toma de decisión y para los problemas de los sistemas de control. Su primera aplicación en el área del diseño mecánico fué en aeronautica y en mecánica estructural, pero rapidamente se extendió al diseño de mecanismos.

La formulación general de la optimización de mecanismos se fundamenta, como es de sobra conocido, en la determinación del mínimo o máximo de una función, llamada función objetivo, de acuerdo con un grupo de restricciones ya sean de igualdad como de desigualdad. Hay que obtener el conjunto de valores de las variables que cumplan esa condición de mínimo.

La función objetivo representará la diferencia que hay entre el mecanismo obtenido y el deseado, en cuanto a alguna caracteristica definida, como puede ser la generación de una trayectoria, la generación de una función etc. Las restricciones están determinadas por las caracteristicas del diseño a obtener y pueden ser desde una limitación del tamaño de los elementos hasta una gama de valores aceptables de un angulo de transmisión. Las posibilidades de formulación de un caso de optimización de mecanismos son enormes, por cuanto, se pueden adoptar diversos criterios de optimización y distintas aplicaciones de los mecanismos.

En la mayoria de los casos, debido a la no linealidad y complejidad de los problemas del mecanismo no es posible obtener una solución de forma directa, siendo dificil de implementar la optimización analítica. En el intento de resolver esta dificultad se han desarrollado varios métodos de resolución. Sutherland (1977) formula un método mixto, exacto-aproximado para la síntesis de posición en mecanismos planos. Las soluciones se pueden obtener de forma directa o bien mediante métodos iterativos, dependiendo del orden de magnitud del problema. Bagci y Lee (1975) han desarrollado una técnica de superposición lineal en la que el error en el sistema de ecuaciones de lazo del mecanismo es minimizado mediante la partición de esas ecuaciones en ecuaciones de lazo de diadas, que son lineales.

Otra forma de resolver el problema es la utilizacion de una búsqueda aleatoria directa, de manera que genera un conjunto de mecanismos solución para el caso en estudio, y posteriormente se selecciona el mejor de entre estos. Roth, Sandor y Freudenstein (1962) utilizaron esta técnica para la obtención de un cuatro barras que satisfaciera especificaciones de trayectorias. Tomas (1968) plantea el tratamiento de la síntesis de mecanismos como un problema de programación no lineal, empleando para su resolución un método de búsqueda aleatoria al igual que Garrett y Hall (1968).

Aun cuando se pueden usar estas formas de tratamiento, la optimización de mecanismos en la actualidad esta enfocada al uso de métodos iterativos. Estos métodos imponen el uso de técnicas números de programación. Existen muchas de estas técnicas y aqui solo se comentarán al-

gunas de las mas usuales en las aplicaciones de mecanismos.

El método de Powell (1964) impone la búsqueda en direcciones conjugadas para encontrar el minimo local en cada iteración. Es quizás el método mas usado en los casos en que la función objetivo no es diferenciable. Suh y Radcliffe (1978) lo emplearon en la sintesis óptima combinandolo con el SUMT (Fiacco y McCormick, 1968).

El método del gradiente conjugado es un método de descenso similar al de Cauchy pero mucho mas eficiente. Este método fue desarrollado por Fletcher y Reeves --- (1964) y aunque presenta determinadas ventajas tiene el inconveniente de presentar problemas de convergencia si la - función objetivo es compleja de evaluar.

El método de Newton como tal no se suele utilizar en los problemas de optimización, dada las dificultades que presenta en su calculo el hessiano de la funcion. Rose y Sandor (1973) utilizan este método pues plantean la sintesis de generación de funciones con mecanismos de cuatro barras y error estructural óptimo de forma que obtienen un sistema de 10 ecuaciones diferenciales no lineales que resuelven aplicando dicho método.

Los llamados métodos quasi-Newton son conocidos tambien como métodos de Métrica Variable y son básicamente técnicas de gradiente que imponen la formulación de una matriz diferencial que aproxima el hessiano. El método fue propuesto en esencia por Davidon (1959) y posteriormente descrito y desarrollado por Fletcher y Powell (1963). Es uno de los métodos mas potentes y utilizados.

Como cuarto y último método de optimización sin restricciones se puede citar el de Newton-Raphson, que como es sabido es un método muy eficaz en cuanto a convergencia se refiere, pero presenta el inconveniente de necesitar la formulación del hessiano de la función objetivo en cada iteración. Han (1966) usó este método para la formulación de un método general de optimización de mecanismos.

Todos estos métodos de optimización sin restricciones, requieren la evaluación de las derivadas de la función objetivo, excepto el de Powell. Esto se puede soslayar aplicando las diferencias finitas pues generalmente dan buenos resultados.

Otra clase de métodos iterativos son los que realizan la optimización con restricciones, reduciendolos a casos de optimización sin restricciones. A estos se puede -- llegar, en determinados casos, realizando cambios de variables y transformaciones similares. Sin embargo en mecanismos esto suele ser demasiado complejo en la mayoría de las ocasiones. Lo usual será utilizar funciones de penalización para reducir el problema de optimización con restricciones a uno sin restricciones. Dentro del uso de las funciones de penalización es de destacar el método SUMT desarrollado por Fiacco y McCormick (1964), debido a lo mucho que se ha aplicado. Fox y Willmert (1967) lo usaron para optimizar un cuatro barras que genera una trayectoria, Gupta (1973) lo utiliza en mecanismos espaciales.

Además de estos métodos hay otro tercer tipo de métodos como son los métodos directos que permiten resolver la optimización con restricciones como tal, sin acudir a su transformación en sin restricciones. De entre

estos se pueden destacar el método de las direcciones factibles, formulado por Zoutendijk, el método del gradiente conjugado y las diversas extensiones del método Simplex.

La síntesis óptima de generación de trayectorias es una de las muchas formas de enfocar la optimización de mecanismos. Generalmente en este tipo de síntesis óptima la función objetivo planteada es tal que representa la diferencia que existe entre la trayectoria generada y la obtenida. Nolle (1974a, 1974b, 1975) en su excelente trabajo de recopilación, recoge la evolución que ha tenido la síntesis de curvas de acoplador hasta llegar a la síntesis optima, y a la síntesis espacial. En todos los casos considerados las barras o elementos que componen los mecanismos son de longitud constante en el ciclo de generación de la trayectoria.

1.4. ESTADO ACTUAL.

En la actualidad se continúa profundizando en el uso de las técnicas de optimización, en la síntesis de mecanismos. Así, cada día surgen más trabajos de síntesis de mecanismos que se apoyan para su realización en las técnicas de programación matemática mas o menos sofisticadas según sea el caso. De hecho, en muchas ocasiones, la optimización no solo se usa como búsqueda del óptimo, sino que se emplea como método de resolución, para la obtención de una solución aceptable, aunque no sea la mejor, cuando no es posible obtener soluciones de otra forma.

Como casos de aplicación de las técnicas de programación matemática a la síntesis de mecanismos, se puede citar entre otros, el trabajo de Rao (1979) en el que

realiza la síntesis de un cuatro barras para la generación de funciones, aplicando la programación geométrica. Garbarouk y Lebedev (1980) realizan la síntesis de una cadena espacial cerrada para la generación de una trayectoria, combinando la búsqueda con la secuencia de Fibonacci con la utilización del algoritmo de Rosenbrock. Con la introducción de la consideración de barras de dimensión variable, se abren unas grandes posibilidades en cuanto a obtener nuevos y mas versátiles mecanismos.

En cuanto a las tecnicas de optimizacion ,se encuentran en una constante evolución. Pappas(1980) ha desarrollado un método de busqueda directa combinando la rotacion de coordenadas con la busqueda en direcciones factibles. Otro método es el propuesto por Tanabe(1979),basado en el método de gradiente proyectado.

Como trabajos interesantes relacionados con el tema se pueden considerar el de Fox y Gupta(1973),que revisan las técnicas de optimización que se aplican al diseño de mecanismos, y el de Sutherland y Siddall(1974) que estudian la síntesis dimensional como un problema de optimización. Thompson(1975) revisa las técnicas analiticas de síntesis de generación de trayectorias en mecanismos planos. Una aplicacion de las funciones de penalización en el caso de mecanismos es el trabajo de Alizade,Novruzbekov y Sandor(1975).

Kramer y Sandor(1975) definen tolerancias en los puntos de precisión en la síntesis de generación de trayectorias.Para un mecanismo de Watt,Bagci(1980) resuelve la síntesis aplicando la partición lineal de las ecuaciones. Relacionados con el estudio de la trayectoria en si,son interesantes los trabajos de McCarthy-Roth(1980) y Kimbrell-Hunt(1980). Sucala(1980) desarrolla el estudio de las caractericticas cinemáticas de un mecanismo con acoplador de longitud variable.

CAPITULO II.

SINTESIS DE MECANISMOS PARA LA GENERACION DE TRAYECTORIAS CON BARRAS DE DIMENSION VARIABLE.

2.1.- INTRODUCCION.

Con el alto desarrollo experimentado en la tecnología actual, la demanda de máquinas más rápidas, seguras y precisas es una constante en la labor del técnico o equipo de diseño de máquinas.

Para satisfacer todas estas necesidades, las máquinas modernas suelen ser más complejas y sofisticadas, lo que supondrá que los mecanismos que las constituyen y forman parte de ellas, serán más precisos, fiables, versatiles y complicados, en suma unas mayores exigencias de funcionamiento. Mediante la síntesis dimensional de mecanismos se podrán obtener muchos de estos de forma que satisfagan las condiciones de diseño planteadas. De esta forma será posible determinar los parámetros y variables que definen al mecanismo que sea capaz de generar una trayectoria deseada, - es decir la síntesis de generación de trayectorias.

Clasicamente la síntesis de generación de trayectorias se realizaba utilizando métodos gráficos, mientras que la utilización de métodos analíticos y numéricos - no tenía prácticamente uso, debido a lo tediosos de su utilización y a su lentitud de resolución, cuando tal cosa era posible.

Con la aparición del ordenador y las calculadoras electrónicas, los métodos analíticos y su resolución numérica mediante programas de ordenador, se imponen de forma definitiva, debido fundamentalmente a su gran potencia y -- versatilidad en el tratamiento de muy diversos tipos de mecanismos, ya sean planos o espaciales. Los métodos gráfi--

cos están quedando un tanto en desuso por cuanto se ven superados en casi todo por los métodos analíticos. Además los métodos gráficos presentan muchas dificultades y son poco prácticos en el tratamiento de mecanismos espaciales.

Con la utilización de barras con dimensión variable, se pueden conseguir mecanismos mucho más versatiles dentro de una muy parecida configuración o estructura topológica, además y entre otras cosas, se podrán conseguir mejores aproximaciones en todos sus puntos entre una trayectoria deseada o prescrita previamente y la generada.

La síntesis con puntos de precisión, y barras de dimensión variable permite determinar que mecanismo cumple las características, en cuanto a la generación de una trayectoria deseada como condición de diseño, con una precisión aceptable. Cuando se desea una mayor aproximación se acude a la síntesis aproximada y a la síntesis óptima que proporciona una solución bastante mejorada del problema.

La síntesis con barras de dimensión variable tendrá gran aplicación en el campo del diseño y concepción de nuevos mecanismos, ya sean cadenas cinemáticas cerradas o bien abiertas, como es el caso de los amnipuladores por cuanto se puede llegar a establecer cual habrá de ser la ley de variación de la dimensión en la barra o barras elegidas, de forma que satisfagan las condiciones de diseño.

2.2. SINTESIS ANALITICA DE GENERACION DE TRAYECTORIAS.

En la síntesis dimensional para la generación de una trayectoria, se parte, del tipo de mecanismo o cade--

na cinemática conocida y se pretende determinar cuales serán los parámetros y dimensiones de los elementos que definen al mecanismo, de forma que se satisfaga en lo mejor posible, la condición de diseño de generación de la trayectoria deseada.

Para poder conocer los valores de los parámetros que definen el mecanismo que cumpla o genere la trayectoria deseada, será necesario, modelizarlo y más concretamente su movimiento, mediante ecuaciones que representen las distintas características del mecanismo, así como las exigencias en su comportamiento para que satisfaga la condición o condiciones de diseño.

Una vez que se formulan las ecuaciones que lo modelizan se podrán conocer los parámetros y dimensiones que definen fisicamente al mecanismo, resolviendo el sistema de ecuaciones planteado.

Ahora bien, para cada mecanismo se obtiene un sistema de ecuaciones específico a ese tipo de mecanismo y únicamente valido para él. Para otro habrá un cambio de topologia o estructura, así como de las relaciones de movimiento entre los distintos elementos, lo que impone que el sistema de ecuaciones que lo defina será distinto del de otro.

Los sistemas de ecuaciones que representan de forma analítica o que definen un mecanismo en la mayoría de los casos son no lineales. Esto supondrá una mayor dificultad de resolución, por cuanto será necesario acudir a métodos numéricos capaces de resolver dichos sistemas de

ecuaciones no lineales. Ahora bien, la bondad de las soluciones que se obtengan dependerán fundamentalmente del método numérico escogido, de las características de convergencia de este, y de la solución inicial o de partida, pues la mayoría de los métodos son de tipo iterativo.

El que cada mecanismo venga representado por un sistema de ecuaciones distinto del sistema que define a otro cualquiera, junto con la no linealidad de ese sistema de ecuaciones, impide una total generalización y automatización de su resolución. Como facilmente se comprende, la resolución mediante ordenador de cada caso, supondrá la realización de un programa distinto del que se realiza para otro caso cualquiera.

Este inconveniente se puede solucionar disponiendo de antemano de una biblioteca de programas de diversos tipos de mecanismos, de forma que puedan ser tratados por los programas de resolución, o bien, organizando los programas de ordenador de tal forma que la parte que define el tipo de mecanismo sea un subprograma que sea fácil de incorporar al programa principal de resolución.

En lo que sigue se adoptará el uso de la notación matricial aplicada a la cinemática (Gupta, 1973; Suh-Radcliffe, 1978; Angeles, 1978) por cuanto suponen una gran generalidad, versatilidad y potencia en su uso, posibilitando plantear los sistemas de ecuaciones que definen el mecanismo de una forma fácil y coherente.

El planteamiento de las ecuaciones que definen el mecanismo vendrá determinado fundamentalmente por el nú

mero de incognitas ó parámetros necesarios para definirlo. Así en el de la Figura 2-1 que representa un cuadrilátero - plano articulado, es necesario determinar los vectores de

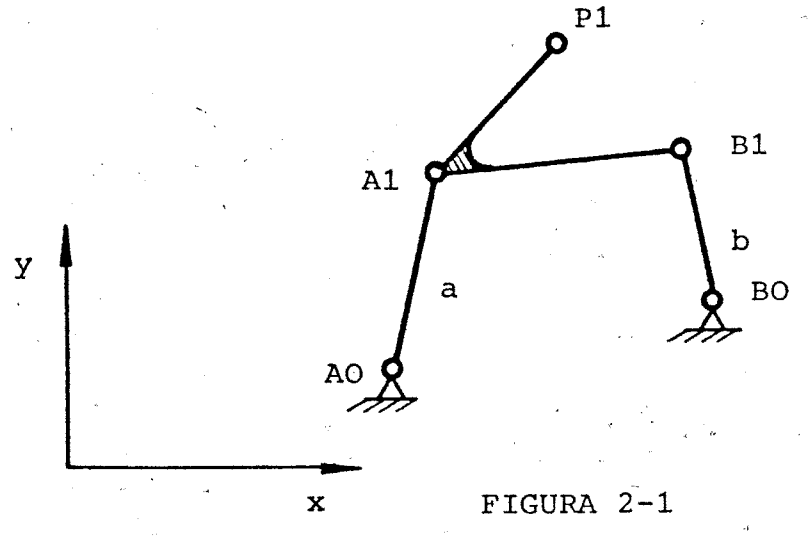


FIGURA 2-1

posición de los pares fijos $\underset{\sim}{A0}$, $\underset{\sim}{B0}$ y de los pares móviles $\underset{\sim}{A1}$, $\underset{\sim}{B1}$ de forma que el vector de posición del punto $\underset{\sim}{P1}$ del acoplador pase, en las sucesivas posiciones de la barra de entrada por los puntos deseados, de la trayectoria especificada.

Una forma posible de planteamiento del sistema de ecuaciones es exigir que las barras de entrada y salida tengan una longitud constante, lo cual expresado en forma matricial será:

$$(\underset{\sim}{AJ}-\underset{\sim}{A0})^T(\underset{\sim}{A1}-\underset{\sim}{A0})=(\underset{\sim}{A1}-\underset{\sim}{A0})^T(\underset{\sim}{A1}-\underset{\sim}{A0});\ j=2,3,\ \ldots\ n \qquad (2\text{-}1)$$

$$(\underset{\sim}{BJ}-\underset{\sim}{B0})^T(\underset{\sim}{BJ}-\underset{\sim}{B0})=(\underset{\sim}{B1}-\underset{\sim}{B0})^T(\underset{\sim}{B1}-\underset{\sim}{B0});\ j=2,3,\ \ldots\ n \qquad (2\text{-}2)$$

donde j indica las sucesivas posiciones del mecanismo, para que pase por los puntos deseados de la trayectoria a generar.

Teniendo en cuenta que:

$$\underset{\sim}{A0} = \begin{Bmatrix} a0x \\ a0y \end{Bmatrix} ; \qquad \underset{\sim}{A1} = \begin{Bmatrix} a1x \\ a1y \end{Bmatrix} ; \qquad \underset{\sim}{AJ} = \begin{Bmatrix} ajx \\ ajy \end{Bmatrix} \qquad (2\text{-}3)$$

la expresión (2-1) será:

$$(ajx - a0x,\ ajy - a0y) \cdot \begin{bmatrix} ajx - a0x \\ ajy - a0y \end{bmatrix} =$$

$$= (a1x - a0x,\ a1y - a0y) \cdot \begin{bmatrix} a1x - a0x \\ a1y - a0y \end{bmatrix} \qquad (2\text{-}4)$$

y que una vez efectuada las operaciones será de la forma:

$$(ajx - a0x)^2 + (ajy - a0y)^2 = (a1x - a0x)^2 + (a1y - a0)^2 \qquad (2\text{-}5)$$

expresión en la que se aprecia de forma más clara, la cons

tancia en la longitud de la barra de entrada en la evolución del movimiento del sistema.

De igual manera la (2-2) será:

$$(bjx - b0x)^2+(bjy - b0y)^2= (b1x - b0x)^2+(b1y - b0y)^2 \qquad (2\text{-}6)$$

Los vectores de posición de los pares moviles $\underset{\sim}{A}J$, $\underset{\sim}{B}J$ en las sucesivas posiciones se determinan de la forma

$$\underset{\sim}{A}J = [R(\theta)]\ (\underset{\sim}{A}1 - \underset{\sim}{A}0) + \underset{\sim}{A}0 \qquad (2\text{-}7)$$

$$\underset{\sim}{B}J = [R(\psi)]\ (\underset{\sim}{B}1 - \underset{\sim}{B}0) + \underset{\sim}{B}0 \qquad (2\text{-}8)$$

donde $[R(\theta)]$ y $[R(\psi)]$ son las matrices de rotación plana que vienen expresadas en forma genérica por :

$$[R(\theta)]=\begin{bmatrix}\cos\theta & -\operatorname{sen}\theta\\ \operatorname{sen}\theta & \cos\theta\end{bmatrix} \qquad (2\text{-}9)$$

donde θ es el valor del ángulo girado en el plano. También es posible expresar las (2-7) y (2-8) de una forma más compacta, haciendo uso de las matrices de desplazamiento plano, en la forma:

$$\underset{\sim}{A}J = |D1J|\ \underset{\sim}{A}1 \qquad (2\text{-}10)$$

$$\underset{\sim}{BJ} = [D1J] \cdot \underset{\sim}{B1} \qquad (2\text{-}11)$$

donde $[D1J]$ viene dada en forma general por:

$$[D1J] = \begin{bmatrix} [R(\theta)] & \underset{\sim}{Pj} - [R(\theta)] \cdot \underset{\sim}{P1} \\ 0,0 & 1 \end{bmatrix} \qquad (2\text{-}12)$$

que desarrollada queda:

$$[D1J] = \begin{bmatrix} \cos\theta 1j & -\text{sen}\theta 1j & (pjx - p1x \cdot \cos\theta 1j + p1y \cdot \text{sen}\theta 1j) \\ \text{sen}\theta 1j & \cos\theta 1j & (pjx - p1y \cdot \text{sen}\theta 1j - p1y \cdot \cos\theta 1j) \\ 0 & 0 & 1 \end{bmatrix} \qquad (2\text{-}13)$$

Para el uso de estas matrices de desplazamiento plano es necesario añadir una tercera fila para que sean coherentes las operaciones matriciales.

De acuerdo con las expresiones (2-1) y (2-2) se tiene el sistema de ecuaciones que definen el mecanismo de forma que genere una trayectoria deseada. Resolviendo el sistema de ecuaciones, para las incognitas que se tienen, se obtiene la solución al problema de diseño o síntesis dimensional del mecanismo para la generación de una trayectoria determinada.

2.3. SINTESIS CON PUNTOS DE PRECISION.

El sistema que se expresa en (2-1) y (2-2) - tendrá tantas ecuaciones como 2.(j-1) donde j es el - número de puntos por los que se desea que pase la trayectoria que genera el mecanismo. Para que el sistema de ecuaciones sea determinado, el número de puntos de precisión no puede ser uno cualquiera sino que esta especificado - por el número de incognitas o parámetros a determinar. - Esto impone que no se podrá exigir que la trayectoria que genera el mecanismo pase exactamente por más de un número determinado de estos.

Así en el caso del mecanismo de la Figura 2-1 y de acuerdo con el sistema que se expresa en (2-1) y (2-2), el número máximo de puntos de precisión que se puede fijar para que pase por ellos, es de nueve, que impone un sistema de 16 ecuaciones no lineales con 16 incognitas, como - son $\underset{\sim}{A}0(2)$, $\underset{\sim}{B}0(2)$, $\underset{\sim}{A}1(2)$, $\underset{\sim}{B}1(2)$ y $\theta j1(8)$ para $j=2,9$. Para otro ejemplo como puede ser el mecanismo de cuatro - -- barras espacial indicado por sus pares como RREE y planteando el sistema de forma análoga, el número máximo de puntos de precisión será de ocho lo que impone un sistema de 39 - ecuaciones con 39 incognitas.

Una vez obtenida la solución del sistema de -- ecuaciones no lineales planteado, se conocerá el mecanismo capaz de pasar de forma exacta por los puntos de precisión especificados. Es por ello que a la síntesis con punto de precisión también se la conoce como síntesis exacta.

Como es lógico, es posible realizar síntesis - exactas con menos puntos de precisión que los necesarios - para que el sistema esté determinado. Tan solo habrá que disminuir el número de parámetros o incognitas a determi-- nar, dandoles un valor específico conocido a las que se -- eliminan como incognitas.

Con la síntesis exacta o de puntos de preci- - sión tan solo se puede obtener un mecanismo que pase, de - forma exacta, por los puntos especificados, sin una posi-- ble actuación en el comportamiento o trayectoria generada entre esos puntos de precisión. Por ello, si se desea que la trayectoria generada sea lo más parecida a la especifi- cada en todos sus puntos, es necesario acudir, a otro tipo de síntesis, por cuanto con la síntesis exacta no es posi- ble de realizar en todos los puntos.

2.4. SINTESIS APROXIMADA DE GENERACION DE TRAYECTORIAS.

Cuando la solución obtenida mediante síntesis exacta no es una solución plenamente satisfactoria en to-- dos los puntos de la trayectoria a generar, se acude a la síntesis aproximada. En esta situación las condiciones -- que se exigen al mecanismo no se pueden satisfacer sin co- meter un cierto error, a pesar de lo cual, se obtienen so- luciones suficientemente satisfactorias, pues la curva de- seada y la obtenida difieren en un grado tan bajo que se - acepta como solución definitiva la así obtenida.

Los errores que se cometen en la síntesis aproximada van decreciendo de una forma progresiva, de acuerdo con el mejor desarrollo de los métodos numéricos de resolución, así como con el constante perfeccionamiento de los modernos ordenadores. Además y aunque se realice una síntesis exacta siempre habrá errores, que pueden ser debidos al propio método numérico de resolución del sistema de ecuaciones no lineales. También hay que tener en cuenta los errores propios del proceso tecnológico de fabricación y funcionamiento, como son los errores cometidos en la fabricación del mecanismo y sus elementos, y los debidos a desgastes y holguras durante el ciclo de vida útil del mecanismo. Según todo esto, el comportamiento del sistema es tan solo aproximado y por eso en la actualidad en muchos casos se consideran equivalentes las síntesis exactas y aproximadas en lo que a la solución obtenida se refiere.

El planteamiento de forma analítica de una síntesis aproximada puede de hecho, no diferir en mucho, del que se hace para una síntesis con puntos de precisión. Sin embargo hay que tener en cuenta que el sistema de ecuaciones no tiene porque ser determinado, así en la mayoría de los casos habrá muchas más ecuaciones que incognitas. Se intenta obtener por tanto, que combinación de los distintos valores de los parámetros de diseño, suministra una mejor aproximación de la curva deseada. Expresandolo de otra forma, se busca obtener el vector de diseño que da una aproximación satisfactoria de la trayectoria deseada. De todas maneras el planteamiento de una síntesis aproximada viene determinado fundamentalmente por el tipo de mecanismo, así como por el objetivo a cubrir.

2.5. SINTESIS OPTIMA.

La sintesis aproximada proporciona soluciones aceptables al problema del diseño, pero en muchas ocasiones presentan problemas de resolución de la formulación - analítica, además, la solución que proporcionan aun cuando satisface las condiciones de diseño no tiene porque - ser la mejor de entre todas las posibles que satisfacen - estas condiciones. Ahora bien si se utilizan las técni-- cas de programación matemática, también conocidas con el nombre de métodos de optimización [Fox, 1971], para obtener el mejor vector de diseño, se tiene una síntesis aproximada en la que el error que se comete es el mínimo posible dentro de las restricciones de diseño impuestas.

Debido al gran auge y desarrollo que están teniendo las técnicas matemáticas de optimización, junto -- con el perfeccionamiento de los ordenadores, el uso de estas en la síntesis aproximada es cada día mayor y su uso recibe, de una forma generica, el nombre de síntesis óptima.

El planteamiento de esta síntesis es basicamente idéntico a la formulación de cualquier problema de optimización, o sea, determinar el vector de diseño $\underset{\sim}{Z}$ (que es el vector formado por todos los parámetros o variables que definen el mecanismo del tipo elegido) de forma que haga

$$\text{Min} \quad F(\underset{\sim}{Z}) \qquad (2\text{-}14)$$

sujeta a las restricciones

$$l_k \leq G_k(Z) \leq h_k; \quad k = 1, 2 \ldots m \qquad (2\text{-}15)$$

donde $F(\underset{\sim}{Z})$ es la función objetivo ó función capaz de determinar la bondad del diseño. Esta función $F(\underset{\sim}{Z})$ determina cuando se obtiene una solución óptima del problema. Como función objetivo se puede tomar cualquier criterio capaz de evaluar la excelencia de un diseño, siempre que se pueda expresar de forma analítica. Ejemplo de funciones - objetivo puede ser, el peso, la resistencia, el costo, etc. También se toma en ocasiones como $F(\underset{\sim}{Z})$ una combinación ó mezcla de funciones objetivo.

En la síntesis de mecanismos para generación de trayectorias, la función objetivo será basicamente, aquella expresión que formule, la diferencia entre la trayectoria - deseada y la obtenida, la cual dependera en esencia del tipo de mecanismo y de su formulación analítica.

Las restricciones (2-15) vienen determinadas -- por las características del diseño a obtener, y estan formadas por un conjunto de igualdades y desigualdades, que expresan determinadas limitaciones y características propias del diseño que se desea conseguir. En síntesis de mecanismos se pueden tomar como restricciones, entre otras, las limitaciones en el tamaño de las barras, características determinadas de movimiento, relaciones de ángulos entre barras etc.

2.6. TECNICAS DE OPTIMIZACION.

En la formulación de la función objetivo y sus restricciones, puede suceder, que las expresiones resultantes sean funciones lineales de las variables de diseño, lo cual, permite obtener una solución de una forma eficiente, haciendo uso del método "SIMPLEX" y sus diversas extensiones (Dantzig, 1949). Sin embargo en síntesis de mecanismos las expresiones que se obtienen son en general no lineales, lo que ímplica el uso de las diversas técnicas de programación no lineal que sean capaces de resolver el problema.

Las técnicas de optimización son muchas y muy diversas según sea el problema a tratar. Estos métodos se clasifican de acuerdo con la posibilidad de que admitan o no el tratamiento de las restricciones. Así se pueden citar, los métodos de optimización sin restricciones, los de optimización con restricciones que se pueden reducir a sin restricciones y los métodos con restricciones propiamente dichos.

Otra característica que diferencia los distintos métodos numéricos de optimización, es el que el algoritmo necesite o no la evaluación de las derivadas de la función objetivo y las restricciones. En lo que sigue, se utilizaran unos y otros, bien de forma individual, bien en combinación, de forma que se obtenga la mejor solución al problema planteado.

2.7. MECANISMOS CON BARRAS DE DIMENSION VARIABLE.

Hasta ahora, en la síntesis dimensional, de un determinado tipo de mecanismo, de forma que un punto de él describa una trayectoria lo más aproximada a otra especificada de antemano, se viene considerando que la dimensión de las barras de este no cambian en la evolución del ciclo de movimiento. Esta combinación de que los elementos sean de dimensión constante es la usual, pero en determinados casos de síntesis no es posible obtener una solución satisfactoria con el tipo de mecanismo establecido, acudiendo a la síntesis óptima, lo que hace necesario orientar el sentido de la síntesis hacia otro tipo de sistema, es decir un cambio en su morfología. Sin embargo, sin cambiar en esencia, el tipo de mecanismo, es posible aumentar, de manera sustancial su versatilidad y posibilidades, si se considera que los elementos que lo forman, son o pueden ser, de dimensión variable en el tiempo, de acuerdo con una función determinada.

Esta suposición de barras de dimensión variable (B.D.V.) se basa en la posible realización física de mecanismos, cuyos elementos, sean cilíndros hidraúlicos o neumaticos, con distintos sistemas de regulación y control, de acuerdo con el caso en estudio y la ley de variación de la dimensión propuesta.

Como es lógico, la síntesis de mecanismos con B.D.V. será algo más compleja, por cuanto, no sólo se tienen que determinar, los parámetros necesarios para definir el mecanismo, como si sus barras fueran de dimensión cons-

tante (B.D.C.), si no que también se tienen que obtener, - los parámetros que definen la ley de variación de la longitud, en los elementos que son B.D.V.

De acuerdo con el número de B.D.V., así como, con las funciones del tiempo que definen la variación de - esa dimensión, se tiene un mayor número de incognitas a determinar, es decir el vector de diseño tiene mayor dimen-- sión, con lo que se podrán fijar un mayor número de puntos de precisión. De esta forma, se obtiene, un mecanismo ca-- paz de generar una trayectoria, que coincide en un mayor - número de puntos con la deseada. A pesar de este aumento en el número de puntos de coincidencia entre las curvas, - es necesario, en muchas ocasiones acudir a la síntesis óptima, a fin de obtener una solución adecuada al caso en estudio.

2.8. SINTESIS DE MECANISMOS CON B.D.V.

El planteamiento se hará de igual manera que en la síntesis con elementos de dimensión constante, con - algunas pequeñas particularidades. En este caso, no se -- puede imponer la invarianza en la longitud de un elemento como restricción de diseño, siendo necesario incluir en -- esa formulación la función del tiempo que representa la variación en la longitud del elemento.

Se utiliza como variable fundamental, el tiempo, con objeto de poder considerar de forma adecuada las - leyes de movimiento de las barras. Por tanto, todo irá referido al tiempo o instante en que se desea que suceda al-

go. Así, los puntos de la trayectoria a generar estarán - definidos además de por sus coordenadas cartesianas, por - el instante de tiempo en que el mecanismo habrá de pasar - por él. De esta forma se introduce directamente en la síntesis, que determinada zona de la trayectoria se recorra a una velocidad prefijada y otra parte de esa misma trayectoria a otra velocidad distinta.

Una vez planteado el sistema de ecuaciones -- que define al mecanismo, se resolverá de forma análoga a - los casos en que las barras eran todas de dimensión cons-- tante en el tiempo. La única diferencia es, que en este - caso hay un mayor número de incognitas a determinar y la - resolución del sistema se hará más tediosa y compleja. Si se desea tan sólo una primera aproximación entre las tra-- yectorias se acude a la síntesis con puntos de precisión,- pero ahora con más puntos al existir más incognitas a de-- terminar. Si se desea una buena aproximación entre ambas curvas es necesario usar la síntesis óptima que proporciona unos excelentes resultados.

Al considerar que alguna de sus barras sea de de dimensión variable, como por ejemplo la de salida en el mecanismo de cuatro barras plano, (Fig. 2-1) antes citado, y que se representa ahora tal como indica la Fig. 2-2, la expresión (2-2) vendrá expresada por

$$(\underset{\sim}{BJ} - \underset{\sim}{B0})^T(\underset{\sim}{BJ} - \underset{\sim}{B0}) = (\underset{\sim}{B1} - \underset{\sim}{B0})^T.(B1 - B0) + f_B(tj) \qquad (2\text{-}16)$$

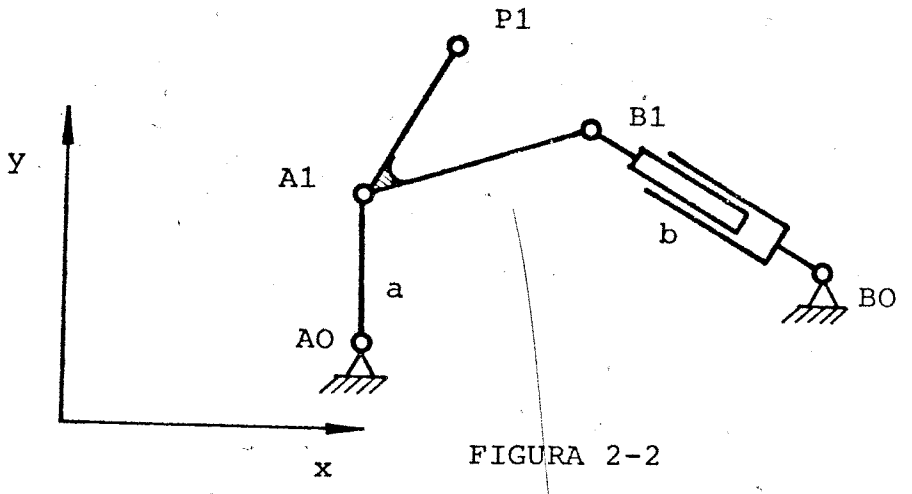


FIGURA 2-2

para j = 2,3,4 .... n, mientras que la (2-1) no cambia. - También hay que realizar determinadas modificaciones en -- las matrices de transformación, tanto planas como espaciales, de manera que las nuevas matrices contemplen la posibilidad de giro de un vector junto con la variación en su longitud.

2.8.1. MATRIZ ROTACION-EXTENSION.

Un vector de posición $\underset{\sim}{A1}$ se transforma en - otro $\underset{\sim}{AJ}$ mediante un giro θj y una variación en su longitud desde l_1 a l_j, lo que se puede expresar de la forma:

$$\underset{\sim}{AJ} = [RL(h,\theta j)] \ \underset{\sim}{A1} \qquad (2\text{-}17)$$

esta matriz de transformación $\left|RL(h,\theta_j)\right|$ puede recibir el nombre de matriz de rotación-extensión, y vendrá expresada por

$$\left[RL(h,\theta_j)\right] = \begin{bmatrix} h\cos\theta_j & -h\,\text{sen}\,\theta_j \\ h\,\text{sen}\,\theta_j & h\cos\theta_j \end{bmatrix} \qquad (2\text{-}18)$$

donde

$$h = \frac{l_j}{l_1} \qquad (2\text{-}19)$$

es la relación entre la longitud final (l_j) y la inicial (l_1). Considerando el caso espacial la expresión (2-18) para el giro θ alrededor de un eje cartesiano como puede ser el eje Z es

$$\left[RL(h,\theta,z)\right] = \begin{bmatrix} h\cos\theta & -h\,\text{sen}\theta & 0 \\ h\,\text{sen}\theta & h\cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (2\text{-}20)$$

Para un eje cualquiera $\underset{\sim}{u}^T = \{u_x, u_y, u_z\}$ esta matriz de rotación-extensión viene dada por

$$[RL(h,\theta,u)] = \begin{bmatrix} hC\theta+u_x^2V^*\theta & u_xu_yV^*\theta-u_zhS\theta & u_xu_zV^*\theta+hu_yS\theta \\ u_xu_yV^*\theta+hu_zS\theta & u_y^2V^*\theta+hC\theta & u_yu_zV^*\theta-hu_xS\theta \\ u_xu_zV^*\theta-hu_yS\theta & u_zu_yV^*\theta+hu_xS\theta & u_z^2V^*\theta+hC\theta \end{bmatrix} \qquad (2\text{-}21)$$

donde $C\theta = \cos\theta$; $S\theta = \text{sen}\ \theta$; y $V^*\theta = \frac{1}{h} - h\cos\theta$.

Utilizando estas matrices de rotación-extensión en aquellos elementos que tengan dimensión variable, el resto del planteamiento será idéntico al caso en que todas las barras - - sean de dimensión fija, dependiendo unicamente del caso en concreto a estudiar. En el Apendice A se ha desarrollado - la obtención de estas matrices de rotación-extensión.

CAPITULO III.

SINTESIS OPTIMA DE MECANISMOS PARA LA GENERACION DE TRAYECTORIAS CON BARRAS DE DIMENSION - VARIABLE.

3.1. INTRODUCCION.

La síntesis óptima permite la obtención de soluciones al problema de diseño suficientemente buenas. Por ello se determina su utilización para la obtención del mecanismo capaz de generar una trayectoria lo más aproximada posible a otra dada.

Sin embargo en la síntesis óptima, y más concretamente, en la utilización de la mayoría de los algoritmos de minimización, de la función definida como objetivo, es fundamental en cuanto a convergencia y obtención del óptimo, el valor del vector de diseño que se toma como solución inicial. Se puede decir que la mayoría de los algoritmos de optimización, son altamente sensibles a los valores de la primera aproximación de la solución, que se introducen como valores de partida, lo cual, condiciona, en la mayoría de las ocasiones, la obtención del óptimo a la determinación o conocimiento de una buena solución inicial.

Así y ante el condicionante fundamental, que es la determinación de una primera solución suficientemente buena, se aborda el problema de la síntesis óptima de mecanismos con barras de dimensión variable (B.D.V.) para la generación de trayectorias.

Al tratarse de una síntesis dimensional, primero se establece que tipo de mecanismo es el que se desea que genere la trayectoria pedida, también es necesario especificar con que tipo de ley de variación de la dimensión en cada barra se desea que se considere el sistema. Hechas

estas especificaciones, se conoce el número de parámetros - o incognitas a determinar, en suma, el vector de diseño que define al mecanismo.

Para la determinación de una primera solución, se acude a una síntesis con puntos de precisión, especifi-- cando tantos puntos de la trayectoria deseada como admita - esa síntesis, de acuerdo con el número de parámetros a de-- terminar. La elección de estos puntos de precisión se hace de acuerdo a un espaciado adecuado para la obtención de una mejor solución. Esta síntesis se resuelve mediante técni-- cas numéricas, de acuerdo con cada caso en particular.

Una vez que se obtiene una primera solución y de acuerdo con sus características, se intenta mejorar, de forma que sea lo más aproximada en cuanto a la generación - de los puntos deseados, pues hay que tener en cuenta, los - errores numéricos que se cometen, en la resolución de la - síntesis exacta o con puntos de precisión.

A partir de esta primera solución se realiza - la síntesis óptima de manera que se obtengan los parámetros que definen al mecanismo capaz de generar una trayectoria,- lo más aproximada posible en todos sus puntos a la deseada. Esta síntesis óptima se realiza de una forma ciclica aumen- tando progresivamente el número de puntos de comparación a tomar entre ambas trayectorias, en vez de hacerlo de una ma nera directa con el número total de puntos de comparación - deseados.

3.2. PLANTEAMIENTO DEL PROBLEMA.

En alguna de las fases de diseño de una máquina, aparece la necesidad de que un mecanismo que va a formar parte de esta, describa una trayectoria específica a fín de que cumpla una cadena de operaciones determinadas.- En muchas ocasiones este planteamiento es suficiente, o -- por decirlo de otra forma, no tiene más requisitos o exi-- gencias de diseño. Sin embargo hay muchos casos en los -- que no solo se desea que algún mecanismo describa una trayectoria sino que se especifica que tipo de estos habrá de ser y también pueden existir una serie de exigencias de diseño en forma de restricciones, como pueden ser, limitaciones en el tamaño de los elementos, determinadas características de movilidad, especificación de la posición o zona - de posición de los pares fijos etc.

Entre otras alternativas para resolver este - problema, está la síntesis de generación de trayectorias.- Con su uso se resuelve en muchas ocasiones el problema que se plantea. Sin embargo cuando existen restricciones en - el diseño, se especifica de una forma clara el tipo de mecanismo que se desea, y la trayectoria a generar es algo - compleja, puede suceder que no se obtenga un resultado aceptable. Una posible solución, sin cambiar de tipo de mecanismo, puede ser, considerar la posibilidad de que algunas de sus barras sean de dimensión variable, lo que introduce -- una gran versatilidad en las características del sistema y permite la determinación de una solución adecuada a los niveles de aproximación exigidos.

A la vista de esto urge la necesidad de estudiar la forma de tratar este tipo de síntesis. Entre las diversas posibilidades que presenta su formulación, en lo que sigue, este trabajo se centra en la síntesis óptima de generación de trayectorias con mecanismos de tipo especificado, con la posibilidad de que alguna de sus barras sea de dimensión variable, con unas leyes de variación determinadas y con la opción de utilizar o no, restricciones en la síntesis, ya sea en las variables de diseño, ya sea en alguna característica del sistema.

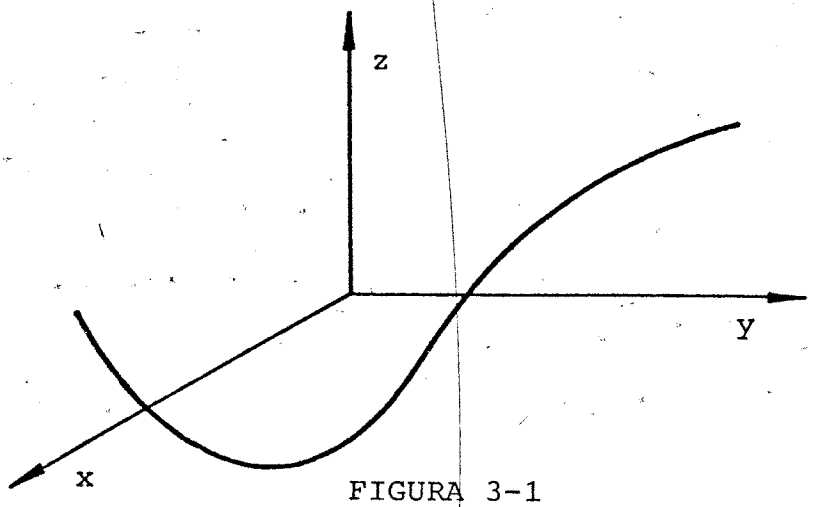


FIGURA 3-1

Se dispone de una trayectoria, plana o espacial (Figura 3-1) la cual se conoce bien mediante una expresión analítica f(t) que en forma paramétrica, donde t(tiempo) es el parámetro, puede expresarse:

$$x = f_1(t)$$
$$y = f_2(t) \qquad (3\text{-}1)$$
$$z = f_3(t)$$

o bien mediante un conjunto de puntos que de manera discreta definen la trayectoria, no disponiendose de una expresión analítica conocida que la represente. Hay que determinar que mecanismo, de un tipo fijado previamente, posee un punto que describe esa f(t) en los instantes de tiempo especificados y teniendo en cuenta que se habrán de cumplir en ocasiones determinadas restricciones de diseño y funcionamiento.

Para conocer el mecanismo es necesario determinar el vector de diseño $\underset{\sim}{Z}$ que lo define de forma dimensional. Para ello se formulan las ecuaciones que definen la trayectoria que genera un punto del mecanismo. Hay que tener en cuenta en esta formulación de la ecuaciones, que como los elementos que forman el sistema pueden ser B.D.V. se hará necesario utilizar las matrices de rotación-extensión, anteriormente mencionadas, que realizan esa consideración. Hay que fijar también cuales van a ser las B.D.V. y con que ley de variación de su dimensión en el tiempo se considera cada una.

Con estas ecuaciones, se plantea la función objetivo para realizar la síntesis óptima, de manera que el error que se cometa entre la trayectoria pedida y la generada, sea mínimo en todos los puntos de comparación entre ellas considerados y de forma que se cumplan las restricciones de diseño y funcionamiento estipuladas, viniendo todo ello expresado como: Determinar el vector de diseño $\underset{\sim}{Z}$ que hace

$$\text{Min } F(z) = \sum_{i=1}^{n} \left[(X_D - X_{OB})_i^2 + (Y_D - Y_{OB})_i^2 + (Z_D - Z_{OB})_i^2 \right] \qquad (3\text{-}2)$$

sujeta a las restricciones

$$l_i \leq G_i(Z) \leq h_i \qquad (3\text{-}3)$$

donde X_D, Y_D, Z_D, son las coordenadas de los puntos de la trayectoria deseada y X_{OB}, Y_{OB}, Z_{OB} lo son de la obtenida. - Para su resolución, será necesario conocer una primera solución, o mejor dicho, un vector de diseño $\underset{\sim}{Z}$ de partida, que sea suficientemente aceptable como solución, a fín de que los algoritmos de optimización obtengan la solución óptima, con una exactitud y convergencia adecuada.

3.3. APROXIMACION DE LA CURVA.

La trayectoria que se quiere generar, puede tener una expresión analítica conocida, que en forma parame-- trica puede ser:

$$\begin{aligned} X &= X(t) \\ Y &= Y(t) \\ Z &= Z(t) \end{aligned} \qquad (3\text{-}4)$$

Pero en muchas ocasiones, cuando no en la mayoría, no se conoce tal expresión analítica, si es que existe. Sólo se -- dispone de un conjunto de puntos que forman esa curva o trayectoria. Una expresión analítica aproximada de esa curva, definida de forma discreta, se obtiene interpolando funcio-

nés de polinomios de splines cúbicos (Ahlberg, 1967; De Boor, 1978) a través de los puntos de la curva, pues ajustan la - curva de una manera aceptable.

El disponer de la expresión analítica aproxima da con funciones de polinomios de spline cúbico, permite co nocer los valores que definen a un punto (X,Y,Z), en fun-- ción de un solo parámetro (t) sin que este punto tenga que ser uno de los conocidos o contenidos en el conjunto que de finía de forma discreta la trayectoria. Estos valores - -- (X,Y,Z) serán aproximados y no totalmente exactos, pero sí suficientemente válidos, como aproximación de los valores - reales por cuanto los errores que se cometen con la interpo lación de este tipo son del orden del 2 al 3% como máximo y dependiendo por supuesto del número de puntos considera-- dos, así como de la forma de la curva. Si se toman sufi- - cientes puntos, entre veinte y treinta, las aproximaciones obtenidas son excelentes, con errores menores del 0'5%. La expresión analítica de la curva dada por las funciones de - polinomios de spline cúbicos será:

$$
\begin{aligned}
X_i &= X_i(t) = B_{x1i} + B_{x2i}\cdot t + B_{x3i} t^2 + B_{x4i}\cdot t^3 \\
Y_i &= Y_i(t) = B_{y1i} + B_{y2i}\cdot t + B_{y3i} t^2 + B_{y4i}\, t^3 \\
Z_i &= Z_i(t) = B_{z1i} + B_{z2i}\cdot t + B_{z3i} t^2 + B_{z4i}\, t^3
\end{aligned}
\qquad (3\text{-}5)
$$

para $i=1, 2, \ldots. n$, donde n será el número de interva-- los o segmentos del polinomio de spline cúbico. Este valor de n es el número de puntos de paso ,menos uno.

Se determina el uso de las funciones de polinomios de spline cúbico, para la interpolación de la trayectoria deseada, debido fundamentalmente y entre otras cosas a la facilidad y flexibilidad en su tratamiento. Además estas funciones garantizan una continuidad hasta la segunda derivada, pasando siempre por los puntos dato o conocidos con lo cual y aunque el error que se cometa entre puntos conocidos es mínimo, también se dispone de puntos en los que el error es cero.

El desarrollo del planteamiento y resolución de los términos que componen la función de spline cúbico estan contemplados en el apendice B y su realización como programa de ordenador en el programa ACUSC (Aproximación de CUrvas con Splines Cúbicos) que más adelante se describe.

3.4.- LEYES DE VARIACION DE LA DIMENSION DE LAS BARRAS.

Como se considera que las barras del mecanismo a obtener pueden ser B.D.V., es preciso conocer cuales habrán de ser las leyes de variación de su dimensión, que tal y como se comentó en el capítulo anterior se consideran como una función del tiempo.

$$l_i = f_i(t) \quad ; \quad i = 1, 2, \ldots n \qquad (3\text{-}6)$$

donde n es el número de barras que tienen dimensión variable.

Será necesario por tanto conocer, la función $f_i(t)$ para cada B.D.V. Es fácil de comprender que la fun-

ción del tiempo que representa la variación de dimensión - de la longitud de la barra del mecanismo $f_i(t)$, puede ser de cualquier tipo. Sin embargo pensando en la posible realización física o tecnológica, mediante cilindros hidraúlicos o neumáticos, de las leyes de movimiento de las B.D.V. $f_i(t)$, se consideran cuatro tipos o clases de ellas. Dos de ellas muy particulares y simplés y las otras dos bastante generales aunque más complicadas. De todas formas siempre es posible, que según el criterio del diseñador se tomen otras leyes de movimiento a la vista del caso a resolver. Las cuatro leyes consideradas son:

3.4.1. LINEAL UNIFORME SIMETRICA (TIPO A)

Esta ley es la representada en la Figura 3-2, donde t_1 es el tiempo total del ciclo de movimiento, ó - intervalo en el que se produce la evolución del movimiento

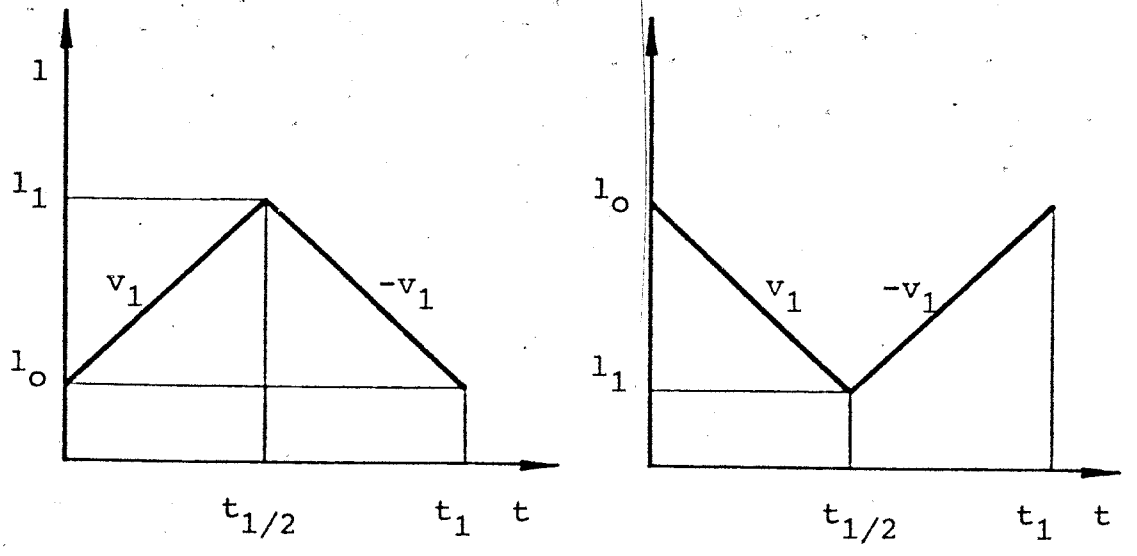


FIGURA 3-2

para la generación de la trayectoria deseada. La Figura - 3-2-a representa la variación de la longitud de la barra entre el instante 0 y t_1 que partiendo de un valor l_0 va aumentando a velocidad constante hasta alcanzar el valor l_1 en el instante $t_1/2$ (mitad del ciclo), a continuación decrece a velocidad constante (el valor absoluto - de esa velocidad es el mismo tanto en la subida como en la bajada) desde l_1 hasta l_0 en el instante t_1. La Figura 3-2-b, representa la misma ley de variación solo en este caso, la longitud l_0 inicial es mayor que la l_1 final y por tanto cambian los signos de la velocidad de crecimiento y decrecimiento.

La función analítica que la representa será:

$$\begin{aligned} f(t) &= l_0 + v.t && 0 \leq t \leq t_{1/2} \\ f(t) &= l_0 + \frac{v.t_1}{2} - v.t \quad ; && t_{1/2} < t \leq t_1 \end{aligned} \qquad (3\text{-}7)$$

Esta función es muy fácil y barata de materializar y simular, y además tan solo se necesita el conocimiento de una - componente más del vector de diseño que define el mecanismo, que será la velocidad v de aumento o disminución de la -- longitud de la barra en el tiempo de evolución del ciclo de movimiento del sistema.

Así, haciendo uso de esta ley, en las barras - que tengan dimensión variable, tan solo se introducen tantas incognitas más como B.D.V., y que en el caso de síntesis con puntos de precisión supone definir tantos puntos -- más, como B.D.V. se consideran.

3.4.2. LINEAL UNIFORME NO SIMETRICA (TIPO B)

En la figura 3-3, está indicada esta ley en la que la longitud de la barra varía desde l_0 a l_1 en el -- instante t_o, con una velocidad v_1 constante, a continuación

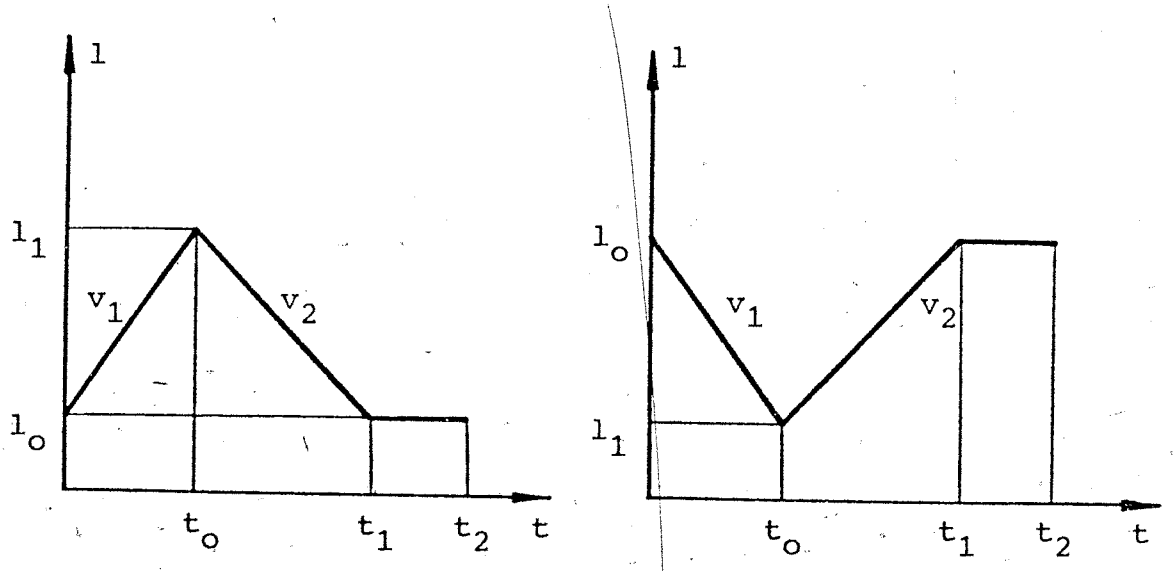


FIGURA 3-3

decrece desde l_1 a l_0 en el instante t_1 con una velocidad v_2. Posteriormente permanece con longitud l_0 constante - hasta el instante t_2 final del ciclo de movimiento. La - Figura 3-3-b representa la misma ley pero con los signos - de las velocidades cambiados. En forma analítica será:

$$
\begin{aligned}
f(t) &= l_0 + v_1 \cdot t && t \leq t_0 \\
f(t) &= l_0 + v_1 t_0 - v_2 (t - t_o) && t_o < t \leq t_1 \\
f(t) &= l_0 && t_1 < t \leq t_2
\end{aligned}
\qquad (3\text{-}8)
$$

Esta ley aunque bastante similar a la anterior, es más versatil, lo que determina la necesidad de conocer - ahora tres componentes más del vector de diseño, por cada - B.D.V. De acuerdo con (3-7) las nuevas incognitas son v_1,- v_2 y t_o por cada barra. En la síntesis con puntos de precisión supone tener que especificar tres puntos más por cada B.D.V. que tenga este tipo de ley.

Se puede considerar una pequeña variante de esta ley, que en realidad es una intermedia entre la ley tipo A y la tipo B. Esta variante, que se denomina ley tipo B-2, surge de considerar que el instante final de ciclo de - movimiento t_2 coincide con t_1. De esta forma una vez conocidos los valores de v_1 y t_o se puede determinar v_2. Por tanto con esta ley B-2 solo se introducen dos incognitas -- por B.D.V.

3.4.3. LINEAL A TRAMOS (TIPO C)

Con la misma filosofia que en las dos anteriores, se concibe una ley lineal más general, que las engloba, y de la que se puede decir que tiene tantos tramos lineales como el diseñador decida. La Figura 3-4 representa esta - ley de movimiento. Como indica dicha figura, la barra que parte con una longitud inicial l_0, va aumentando o disminuyendo su longitud de forma lineal en función de distintas - v, según en el tramo en que se encuentre un instante t determinado.

De forma analítica puede expresarse:

$$f(t) = l_0 + v_0(t_0 - t) \quad ; \quad t \leq t_o$$

$$f(t) = l_0 + v_o \cdot t_o + v_1(t_1 - t) \quad ; \quad t_o < t \leq t_1 \qquad (3\text{-}9)$$

$$f(t) = l_0 + v_0\, t_0 + v_1 t_1 + v_2(t_2 - t) \; ; \quad t_1 < t \leq t_2$$

.....

.....

que en forma genérica se puede escribir como:

$$f_i(t) = l_i = l_0 + \sum_{j=0}^{j=i-1} V_j t_j + V_i(t_i - t) \; ; \; t_{i-1} < t \leq t_i \qquad (3\text{-}10)$$

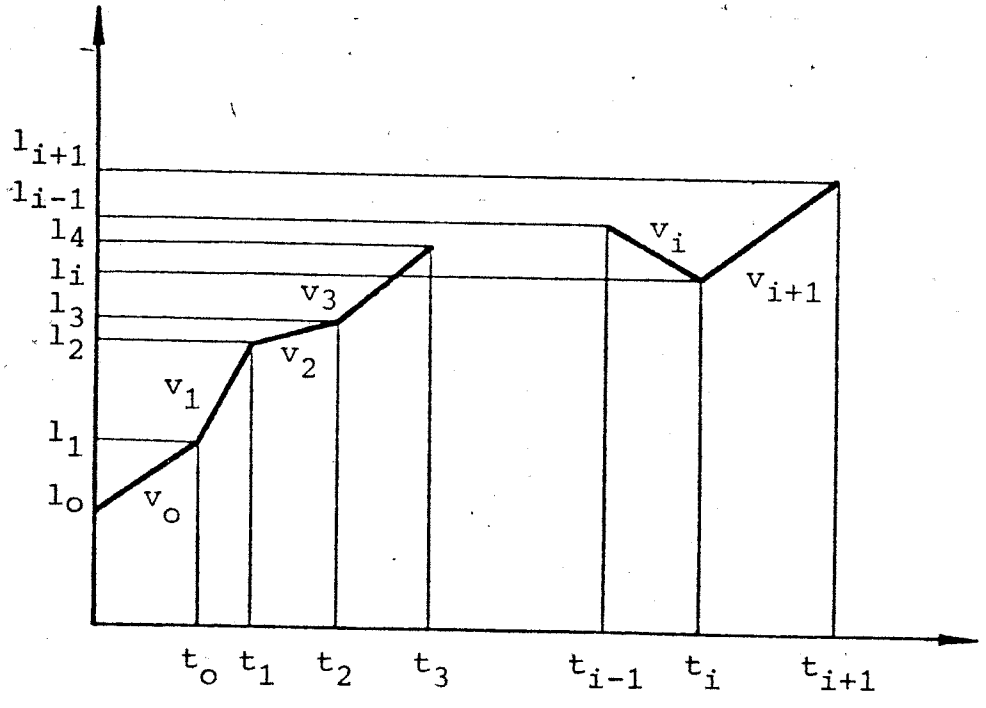


FIGURA 3-4

Esta ley es mucho más general que las anteriores y por tanto también más compleja de realizar, analizar y simular. Sin embargo, proporciona una gran versatilidad al mecanismo que tiene B.D.V. con esta ley. Es necesario para cada caso concreto, definir cual es el número de tramos a considerar, pues de este valor depende el de las componentes que hay que añadir al vector de diseño. Considerando el uso de una ley de este tipo en las B.D.V., el aumento en la dimensión del vector de diseño puede ser considerable, dependiendo de el número de tramos considerados para la ley de cada barra. Este aumento de la dimensión del vector de estado, aún cuando, le da al mecanismo una gran potencia y versatilidad en sus posibilidades, hace en muchas ocasiones que la obtención de la solución sea lenta, pues su resolución numérica viene determinada fundamentalmente por el nº de incognitas. Lo que supone que puede llegarse al caso de agotar la capacidad de la memoria central del ordenador que se utiliza, teniendo que trabajar en disco, resultando mucho más lenta la resolución.

3.4.4. LEY GENERAL (TIPO D)

La Figura 3-5 representa la ley general de variación de la longitud de una barra de un mecanismo. Esta ley al ser general viene determinada por alguna expresión f(t) que la define de acuerdo con el criterio de diseño.Con tal función f(t) general, de la que a priori no se conoce, ni su expresión ni su forma, no se puede establecer el nº de componentes que hay que añadir al vector de diseño para definir el mecanismo en un movimiento. Esto impide conocer el número de puntos de precisión a tomar, para realizar una síntesis exacta de mecanismos que tengan en sus B.D.V. leyes generales de este tipo.

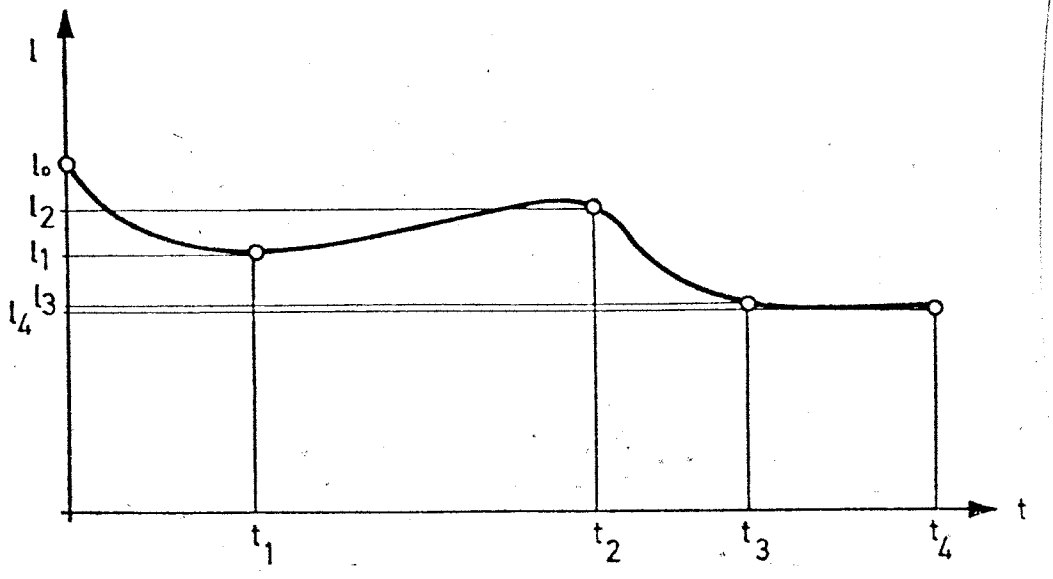


FIGURA 3-5

El no disponer de la expresión analítica que representa a f(t), constituye una seria dificultad en la realización de la síntesis, pues no se dispone de una forma cómoda de relacionar la variación de la dimensión en la barra, con los demás parámetros que definen el sistema. Para soslayar esta dificultad se acude a determinar una forma de tratamiento, basada en la obtención de una f*(t) que sea tan solo aproximada a la verdadera f(t) que es desconocida. Para obtener esta f*(t) se acude de nuevo a las funciones de splines cúbicos, ya utilizadas para aproximar la trayectoria. Su uso permite conocer una f*(t) bastante aproximada a f(t), si de alguna manera, se puede disponer de algunos puntos de f(t), pues no habrá que hacer nada más que interpolar por ellos las funciones de splines cúbicos. La f*(t) vendrá expresada de la forma:

$$f^*(t) = l_i = B_{1i} + B_{2i}(t-t_i) + B_{3i}(t-t_i)^2 + B_{4i}(t-t_i)^3 \quad (3\text{-}11)$$

para $t_i \leq t \leq t_{i+1}$ donde $i = 1, 2, \ldots n-1$

siendo n el número de puntos que se utilizan para la interpolación con lo que i indica el número del tramo en que se realiza esta.

Con la utilización de f*(t) es posible, en función de unos pocos puntos o incognitas, conocer y controlar la evolución de esta ley general de variación. Por tanto, solo habrá que añadir a las incognitas de definición del mecanismo (vector de diseño), tantas más, como dos veces el número de puntos entre los que se desea interpolar la función f*(t), menos uno, por cada una de las B.D.V. que tengan esa ley general. Así, y tomando como ejemplo la Figura 3-5, si esa es la ley que se pretende aproximar, se excogen cuatro puntos (lo que da 8 coordenadas) como son en esta ocasión

$$\begin{matrix} (t_1, l_1) \\ (t_2, l_2) \\ (t_3, l_3) \\ (t_4, l_4) \end{matrix} \qquad 3\text{-}12)$$

que junto con el punto inicial (t_o, l_o) definen los puntos de paso de esta función de spline cúbico. En este caso se añaden 8 incognitas a determinar por cada barra que tenga es

ta ley general, además de las especificas del tipo de mecanismo.

Para la resolución de la interpolación mediante funciones de spline cúbicos, además de los puntos de paso es necesario especificar dos condiciones de contorno, una por cada extremo de la curva total a interpolar, equivalentes a la pendiente de dicha curva en el punto inicial y final. Estas condiciones se pueden introducir como dos incognitas más por cada barra con ley general de variación, pero a la vista de los resultados y aproximaciones obtenidas se escoge como única condición de extremo, la denominada, extremo natural (Ahlberg, 1967), o lo que es lo mismo, que la segunda derivada es igual a cero.

Otra ventaja de la utilización de la interpolación con funciones de spline cúbico, es la de poder obtener su primera y segunda derivada, al ser funciones continuas con primera y segunda derivada continuas. Esto permite conocer fácilmente la velocidad y aceleración en la ley de variación de la dimensión del elemento.

La realización física de esta ley general es la más compleja, debido fundamentalmente a su generalidad, y por tanto también será la más costosa. Igualmente, la resolución numérica, para la determinación del vector de diseño, es más lenta que en los casos anteriores, si bien la versatilidad y sobre todo la precisión que se obtiene es muy elevada.

Estas cuatro leyes que hasta aquí se han definido estan basadas en su posible realización física con ci-

líndros hidráulicos o neumáticos de manera que:

- Ley A.- Es la que corresponde a un cilíndro con la misma velocidad de avance que de retroceso y que realiza una carrera de avance en la mitad del tiempo de ciclo y la de retroceso en la otra mitad.

- Ley B.- El cilíndro tiene distinta velocidad de avance que de retroceso. En el tiempo de ciclo solo realiza una carrera de avance y otra de retroceso.

- Ley C.- En este caso el cilíndro dispone de válvulas más complejas, teniendo una gama de velocidades de avance y retroceso, y realizando varias carreras de avance y retroceso en el tiempo de ciclo.

- Ley D.- El sistema tiene servoválvulas que permiten gobernar la variación de la longitud de avance y retroceso, de forma que se consigue una ley general propuesta por el diseñador.

De acuerdo con todo esto, el coste y las posibilidades de realización varían mucho entre usar una ley del tipo A a una del tipo D. Todo ello hay que tenerlo en cuenta a la hora de elegir o determinar que ley de variación van a tener las B.D.V. Hay casos en los que no se obtienen soluciones aceptables con la ley escogida y hay que acudir a otra más versatil aunque más costosa.

3.5. BUSQUEDA DE UNA PRIMERA SOLUCION.

Una vez conocida la trayectoria que se desea generar y el tipo de mecanismo, así como, las barras que son B.D.V. y se tienen escogidas las leyes de variación en esas B.D.V., se trata de obtener el vector de diseño o solución, que identifica al sistema capaz de generar esa trayectoria con el mínimo error, de acuerdo con las restricciones de diseño impuestas.

Es necesario, en principio, determinar una primera solución, que sea aceptable como solución de partida para la síntesis óptima. Para la determinación de esta primera solución se realiza una síntesis con puntos de precisión. Así, una vez definido el sistema de ecuaciones que representa al mecanismo, en su movimiento a lo largo del ciclo de generación de la trayectoria, se toman tantos puntos de está, como puntos de precisión sean necesarios, de acuerdo al número de componentes del vector de diseño.

Una vez planteada esta síntesis, se trata, en la mayoría de las ocasiones de resolver un sistema de ecuaciones algebraicas no lineales. Sin olvidar, en ningún caso, que pueden existir restricciones de diseño. A veces, y dependiendo del tipo de restricciones, es posible incluirlas, de alguna forma, en el sistema de ecuaciones de tal modo que su solución tiende a cumplirlas. Esto se consigue mediante manipulaciones algebraicas, como son los cambios de variables, sustituciones, transformaciones, etc. Cuando no es posible esta eliminación de las restricciones, la obtención de una solución a esta síntesis con puntos de preci

sión y restricciones es algo más compleja de obtener, y -- además se hace necesario acudir a la utilización de métodos de programación matemática. En este trabajo se consideran las dos posibles síntesis, una sin restricciones o si las - tiene se puede reducir a sin restricciones y la otra con es tas. Dependiendo de si es una u otra, se elige la forma de resolución.

3.5.1. SINTESIS CON PUNTOS DE PRECISION SIN RESTRICCIONES.

Si no hay restricciones, se trata de obtener la solución de un sistema de ecuaciones algebraícas no li-- neales. El método clásico de resolución es la utilización del algoritmo de Newton-Raphson (Paul, 1979) y aplicandolo se puede obtener la solución del sistema. Ahora bien, al - ser un método iterativo es necesario conocer un primer va-- lor del vector de diseño, para empezar el proceso iterativo. Puesto que no se conoce, en principio, cual puede se el valor de solución, ni de forma aproximada, no es fácil esta-- blecer un vector de diseño inicial para el método de N-R.

Para resolver esta dificultad, se genera una familia de vectores de diseño iniciales de forma aleatoria, es decir, se obtienen de manera sucesiva una serie de vecto- res en los que el valor de sus componentes es generado alea toriamente entre unos límites prefijados, de la forma

$$\underset{\sim}{Z}_i = \underset{\sim}{Z}_{min} + \underset{\sim}{\eta}_i (\underset{\sim}{Z}_{max} - \underset{\sim}{Z}_{min}) \qquad (3\text{-}13)$$

para $i = 1, 2, \ldots\ldots m$, donde m es el número de vecto-- res a generar y

$\underset{\sim}{\eta}_i$: vector de números aleatorios entre 0 y 1

$\underset{\sim}{Z}_i$: vector de diseño generado

$\underset{\sim}{Z}_{min}$: vector de límites inferiores de las componentes

$\underset{\sim}{Z}_{max}$: vector de límites superiores de las componentes.

Con este conjunto de vectores iniciales aleatorios se intenta resolver el sistema mediante el algoritmo de N-R. La mayoría de ellos o no producen soluciones aceptables o el método diverge a partir de ellos, pero hay otros que dan soluciones que se pueden considerar como válidas, entendiendo por tales, aquellas que queden dentro de un rango de error definido previamente. Es evidente, que cuanto mayores sean los intervalos entre los que se generaran los vectores aleatorios, más dificil será obtener una solución al sistema.

De acuerdo, con las posibilidades del mecanismo y las características del caso en estudio, se pueden establecer unos valores de $\underset{\sim}{Z}_{min}$ y $\underset{\sim}{Z}_{max}$ que representan los límites entre los que se desea que se encuentren los valores de las componentes del vector de diseño.

Al aplicar la familia de vectores aleatorios al algoritmo de N-R se obtiene un conjunto de soluciones de las que unas serán mejores que otras. Se establece una clasificación en cuanto a bondad de la solución, a continuación se ordenan de mejor a peor. La mejor solución se toma

rá como esa primera solución que se anda buscando. En ocasiones puede ser interesante realizar la síntesis óptima - con alguna otra de estas soluciones además de hacerlo con la mejor.

Sabido es, que no siempre se obtienen soluciones adecuadas con el método de N-R, pues o bien no converge o bien lo hace a soluciones que no son realizables - tecnológicamente. Una forma de mejorar su convergencia y estabilidad es hacer uso de la corrección retardada (Suh--Radcliffe, 1978). Sin embargo no siempre se obtienen soluciones adecuadas ni aún usando este tipo de corrección, lo cual hace necesario resolver esta síntesis mediante otros métodos.

Para la resolución del sistema de ecuaciones algebraicas no lineales que plantea esta síntesis, se acude a la formulación del problema como si uno de optimización - se tratara. Entonces lo que se busca es el vector de diseño $\underset{\sim}{Z}$ que hace mínima la norma del sistema de ecuaciones, - entendiendo por tal

$$\text{Norm} = \sum_{i=1}^{n} \left| f_i(\underset{\sim}{Z}) \right| \qquad (3\text{-}14)$$

donde n es el número de ecuaciones y $f_i(Z)$ es el valor del residuo en la ecuación i, cuando se toma por solución el vector $\underset{\sim}{Z}$. Luego, hay que determinar el vector $\underset{\sim}{Z}$ que hace

$$\text{Min} \quad F(\underset{\sim}{Z}) = \text{Min}\left(\sum_{i=1}^{n} \left| f_i(Z) \right| \right) \qquad (3\text{-}15)$$

sin restricciones de ningún tipo.

Planteandolo de esta forma, ya no es necesario para su solución, que sean estrictamente correspondientes, el número de puntos tomados de la curva, con la dimensión del vector a obtener. Cuando se llega a esta situación, ya se planteó y resolvió por N-R el sistema. Bueno será, por tanto, aprovechar esta formulación, que a la hora de resolverlo en el ordenador se traduce, en utilizar la misma subrutina de definición del sistema de ecuaciones, para esta forma de resolución. Se pueden tomar más puntos de la trayectoria, pero para obtener una primera solución es suficiente hacerlo con el mismo número de puntos. Además, si se aumenta el número de puntos hay que modificar los datos de entrada y sobre todo, que el tiempo de resolución será mayor cuanto más puntos se tomen.

De los muchos algoritmos de optimización sin restricciones que se conocen (Wolfe, 1978), se contempla la posibilidad de utilizar cuatro de ellos para la resolución del sistema de ecuaciones algebraícas no lineales. Dos de estos algoritmos, requieren la evaluación de la derivada de la función objetivo y son, el método del gradiente conjugado (GC), también conocido como método de Fletcher y Reeves y el método de la metrica variable (MV) y que se conoce como método de Davison-Fletcher-Powell (Fox, 1971). La elección de estos métodos se debe a que constituyen dos métodos, ya clásicos en la actualidad, suficientemente potentes y adecuados al problema a resolver. El método MV es esencialmente mejor que el GC y por tanto se utiliza en la mayoría de las ocasiones. Ahora bien, si el número de componentes

del vector de diseño es elevado, la utilización del GC presenta ventajas frente al MV, pues este último se vuelve - más lento y además puede presentar problemas de almacena-- miento. De esta forma, se utilizan uno u otro método, de forma alternativa, según sea el tamaño del problema.

La determinación de las derivadas de la función objetivo, necesarias para estos dos algoritmos al ser métodos de gradiente, se efectuán de forma numérica, mediante diferencias finitas, con objeto de ganar en generalidad ante el tratamiento de diversos casos, pero se admite la - posibilidad de suministrarselas directamente al algoritmo, en forma de subrutina, cuando tal caso se desee y sea posible calcularlas de forma analítica.

A veces, por las propias características de la función objetivo, no es posible determinar, de manera - adecuada, estas derivadas, y su determinación numérica no ofrece resultados satisfactorios tampoco. Para este caso se tiene prevista la utilización de otros dos algoritmos - de optimización, como son, el método de las direcciones - conjugadas (DC) (Powell, 1964) y el método de busqueda directa (BD) de Hooke y Jeeves (Rao, 1978), con el fin de - disponer también de dos alternativas en esta situación de no evaluación de las derivadas.

Con estos cuatro algoritmos de optimización, es suficiente para la resolución del sistema de ecuaciones algebraícas no lineales, en el caso de que NR no obtenga - una solución satisfactoria. No es necesario el uso de los cuatro, utilizandose de forma alternativa según sea el sistema. También se pueden usar para mejorar la primera solu

ción que suministra NR si, aún estando dentro del rango de - error pedido, se quiere refinar.

Con el uso de estos cinco métodos (NR, GC, - VM, DC, VD), a partir de un conjunto de vectores de diseño, generados aleatoriamente, se obtiene, una solución a la síntesis con puntos de precisión que se utiliza como vector - inicial en la síntesis óptima.

3.5.2. SINTESIS CON PUNTOS DE PRECISION Y RESTRICCIONES.

Cuando las restricciones al diseño no se pueden absorber en el sistema de ecuaciones, mediante las distintas transformaciones posibles, el sistema se resuelve directamente como si de un caso de optimización con restricciones se tratara. La función objetivo será la misma que - en el apartado anterior, para el caso en que NR no obtenía una solución adecuada (3-15). Además, hay que considerar que tiene unas restricciones que se pueden expresar como:

$$\begin{aligned} & G_j(\underset{\sim}{Z}) \leq 0 \; ; \quad j = 1, 2, \ldots m \\ & l_i \leq z_i \leq h_i ; \quad i = 1, 2, \ldots n \end{aligned} \qquad (3\text{-}16)$$

Para resolverlo se puede utilizar alguno de los métodos de - óptimización con restricciones existentes (Rao, 1979). Ahora bien, como la mayoría son bastante sensibles al valor del vector de partida, el método a utilizar tiene que ser tal -- que no presente este tipo de dificultad. Se obtienen buenas soluciones utilizando el método "Complex" (COM) (Box, 1965).

Este método, que es una extensión del método "Simplex", es un algoritmo de busqueda directa que parte de un conjunto de puntos generados aleatoriamente en el espacio R^n, y que cumplen las restricciones. Esta búsqueda la realiza dentro del espacio definido por las restricciones, y consigue una buena convergencia hacía el mínimo de la función contenido en dicho espacio.

El algoritmo que asegura una buena convergencia en el principio de su actuación a partir de los valores aleatorios generados, se vuelve lento en las proximidades del óptimo y presenta dificultades para obtener una solución muy ajustada si el vector de diseño tiene una dimensión elevada. Pero como la solución que se busca, es solo un valor cercano al óptimo, para entrar con él, en una síntesis óptima propiamente dicha, el resultado que proporciona este algoritmo, trás un número suficiente de iteraciones, es muy aceptable.

3.6. OPTIMIZACION Y CICLOS DE MEJORA DE LA OPTIMIZACION.

Tomando como vector de partida, el obtenido como solución de la sínteiss con puntos de precisión, se realiza la sínteis óptima propiamente dicha. La solución óptima será aquella que define un mecanismo capaz de generar una trayectoria especificada, con un error mínimo permitido en todos los puntos de esta. La función objetivo que se plantea para la realización de esta síntesis óptima, será una función, que cuando se haga mínima indicará que la trayectoria deseada y la generada son practicamente iguales.

Esta función tiene una expresión analítica que puede variar con cada tipo de mecanismo como se verá en el capítulo 5, - aunque de forma genérica se puede tomar como tal, la expresada por (3-2).

Con la función objetivo y las restricciones - de diseño, si las hay, está planteado el problema para la - realización de la síntesis óptima. Entre las dos trayectorias se definen tantos puntos de comparación, como intervalos de tiempo hay desde el principio al fín de la trayectoria, entendiendo por intervalos de tiempo, el incremento de t que se define al especificar la trayectoria deseada.

La síntesis se puede realizar, haciendo la -- comparación entre ambas trayectorias, en esos M puntos de - comparación y evaluando el error que se comete para que se haga mínimo. De acuerdo con los resultados obtenidos se puede decir, que es más eficiente, en tiempo de ordenador, así como en exactitud de la solución obtenida, realizar ciclos de optimización y mejora de la solución, aumentando progresivamente el número de puntos de comparación, que realizar-- lo de forma directa.

Así para la obtención de la solución final se procede como sigue:

- Una vez determinada una solución inicial - mediante la síntesis con puntos de precisión (n puntos), se utiliza como vector inicial en la síntesis óptima con n_1 -- puntos de comparación entre las trayectorias, siendo este - n_1 algo mayor que n.

- La solución que se obtiene con esos n_1 puntos de comparación se utiliza como solución inicial para una nueva síntesis óptima pero ahora con n_2 puntos de comparación ($n_2 > n_1$).

- Se continua realizando ciclos de este tipo, aumentando progresivamente el número de puntos de comparación, y tomando la solución de un ciclo como vector de -- partida del siguiente, bien hasta que se alcance el número de puntos de comparación deseados, ó bien se tenga una solución que sea satisfactoria como solución final.

Como la solución de partida para cada ciclo es ya de por si una buena aproximación de la solución final, cada uno de estos ciclos se efectúa de forma rápida, ya que convergen en pocas iteraciones. Las soluciones intermedias en la mayoría de los casos son suficientes como solución final y no es necesario realizar todos los ciclos hasta llegar a M puntos de comparación, ya que no se obtiene mejora apreciable.

La utilización de estos ciclos de optimización es fundamental para obtener en tiempos razonables, soluciones a la síntesis óptima de generación de trayectorias, con mecanismos que tengan B.D.V. Lo cual se fundamenta, en que las funciones objetivo que se formulan, son complicadas de evaluar y el número de puntos de comparación, está - relacionado directamente con el número de veces que se evalua la función objetivo.

Esta síntesis óptima se realiza haciendo uso de dos algoritmos de optimización con restricciones. Uno - de ellos es un método indirecto y necesita la evaluación de las derivadas de la función objetivo, en esencia es el método de la metrica variable modificado con funciones de penalización (MVP), con objeto de que pueda tratar las restricciones (Haarhoff, 1970). El otro algoritmo, no necesita -

la evaluación de las derivadas de la función objetivo y se basa en el algoritmo de Rosenbrock (RO) (Rosenbrock, 1960) con pequeñas modificaciones para que pueda tratar las restricciones. Aunque el MVP es suficientemente potente, da excelentes resultados, y será el que se use prácticamente siempre en las aplicaciones realizadas, el RO será útil en el caso en que se tengan problemas con las derivadas de la función objetivo.

Como resumen de todo lo dicho, sobre síntesis óptima de generación de trayectoria con mecanismos de B.D.V. se han realizado tres esquemas. El primero, Figura 3-6, representa un resumen general de la metodología seguida. En la Figura 3-7, se indica la forma de obtener la primera solución, es decir realizar la síntesis con puntos de precisión. Finalmente el tercer diagrama, Figura 3-8, esquematiza la forma de realizar la síntesis óptima propiamente dicha.

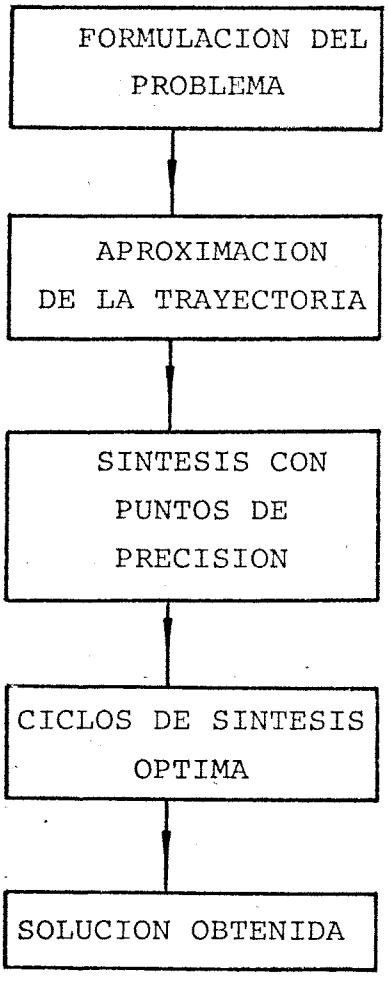

FORMULACION DEL PROBLEMA
APROXIMACION DE LA TRAYECTORIA
SINTESIS CON PUNTOS DE PRECISION
CICLOS DE SINTESIS OPTIMA
SOLUCION OBTENIDA

FIGURA 3-6

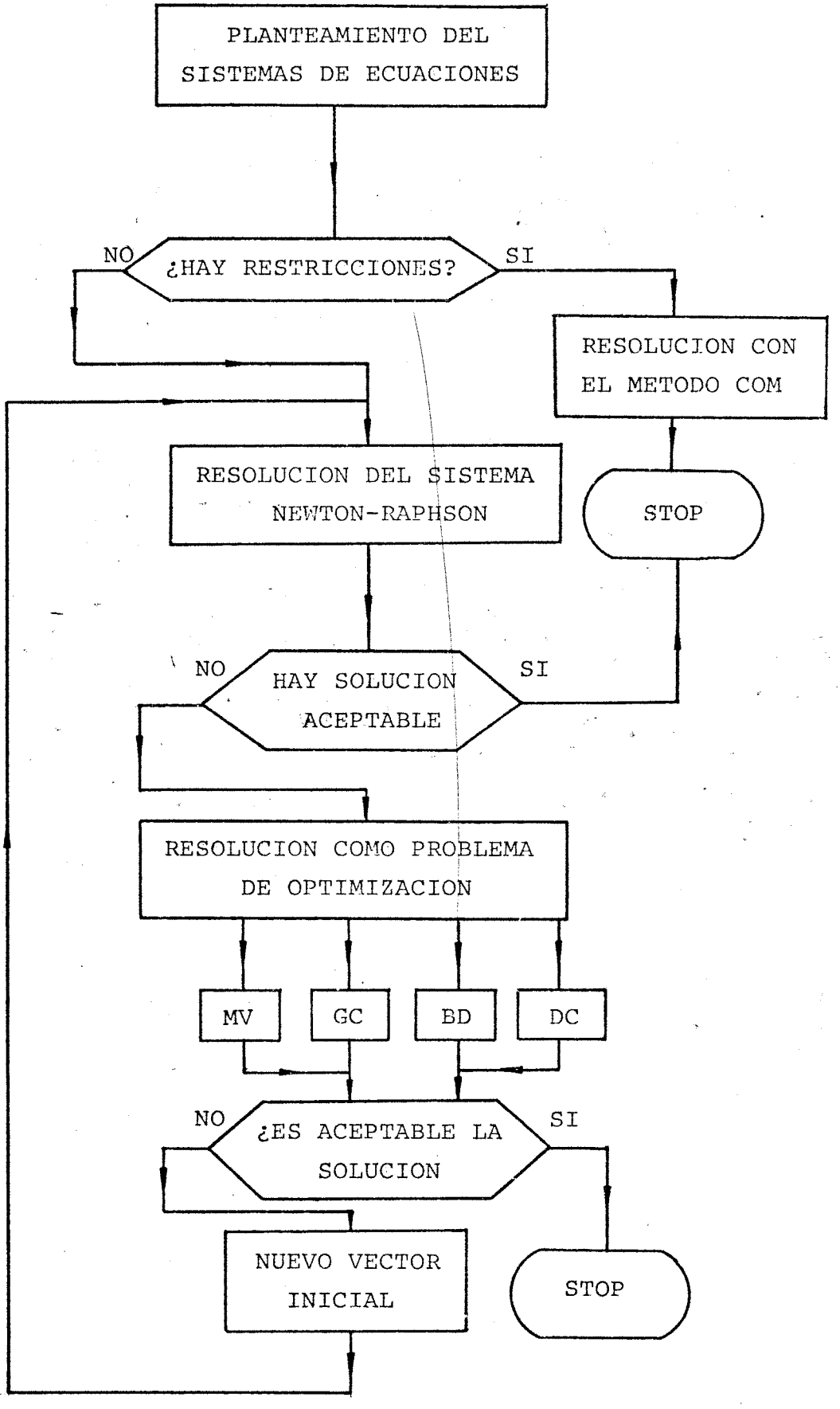

PLANTEAMIENTO DEL SISTEMAS DE ECUACIONES
NO
¿HAY RESTRICCIONES?
SI
RESOLUCION CON EL METODO COM
RESOLUCION DEL SISTEMA NEWTON-RAPHSON
STOP
NO
HAY SOLUCION ACEPTABLE
SI
RESOLUCION COMO PROBLEMA DE OPTIMIZACION
MV
GC
BD
DC
NO
¿ES ACEPTABLE LA SOLUCION
SI
NUEVO VECTOR INICIAL
STOP

FIGURA 3-7

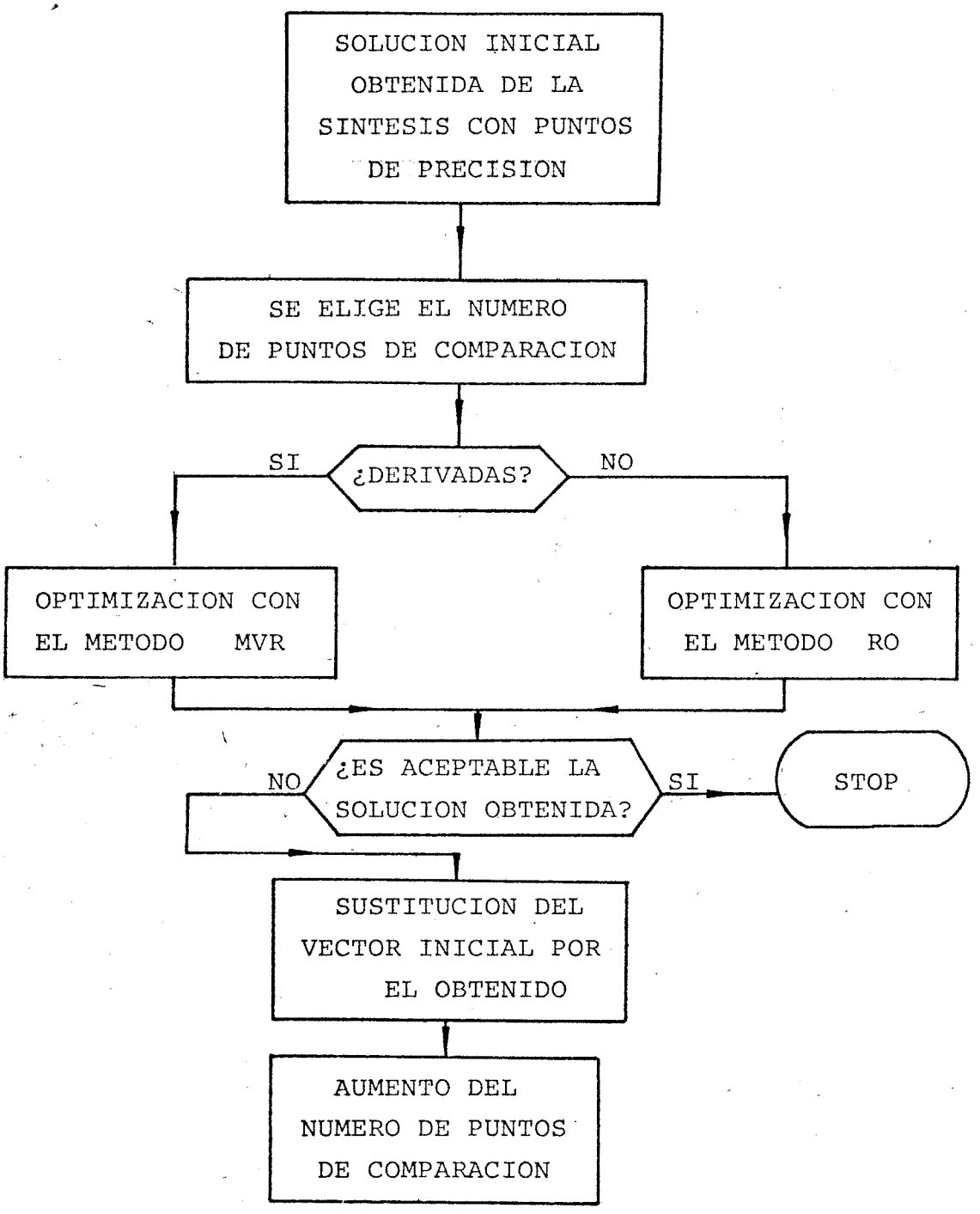

SOLUCION INICIAL OBTENIDA DE LA SINTESIS CON PUNTOS DE PRECISION
SE ELIGE EL NUMERO DE PUNTOS DE COMPARACION
SI
¿DERIVADAS?
NO
OPTIMIZACION CON EL METODO MVR
OPTIMIZACION CON EL METODO RO
NO
¿ES ACEPTABLE LA SOLUCION OBTENIDA?
SI
STOP
SUSTITUCION DEL VECTOR INICIAL POR EL OBTENIDO
AUMENTO DEL NUMERO DE PUNTOS DE COMPARACION

FIGURA 3-8

CAPITULO IV.

PROGRAMAS DE ORDENADOR UTILIZADOS.

4.1. INTRODUCCION.

Como se ha visto en el capítulo 3, la realización de la síntesis óptima de generación de trayectorias con mecanismos de B.L.V. se fundamenta en la utilización de determinados métodos números y algoritmos de optimización. Por eso, y para el desarrollo de lo descrito en el capítulo 3, se hace necesario la utilización de diversos programas de ordenador. Estos programas se han implementado en el ordenador HP-21MX, perteneciente a la Escuela Técnica Superior de Ingenieros Industriales de la Universidad de Sevilla.

La capacidad de memoria disponible en este ordenador es en la actualidad de 32K, de las que aproximadamente 28K son las que hay disponibles, quedando el resto para las subrutinas propias del sistema. Con esta memoria disponible, se hace necesario segmentar los dos programas fundamentales OPRIS (OPtimización pRImera Solución) y CIOPT (CIclos de OPTimización) pues, por su tamaño exigen una capacidad mucho más elevada. A pesar de esta segmentación, en la versión actual de los programas, el número máximo de variables de diseño a tratar es de 30, valor suficiente para las aplicaciones realizadas.

En el capítulo 2 se comentaba que es necesario un programa para cada tipo de mecanismo, debido a la no linealidad del sistema de ecuaciones que lo defina, así como a que la función objetivo varía de un caso a otro, además de la consideración o no de las restricciones. También se apuntaba, que una manera de resolverlo es organizando de

tal forma los programas, que solo sea necesario añadirles - una subrutina, en la que se define al sistema. De acuerdo con esto los programas están organizados de tal forma que - para estudiar distintos tipos de mecanismos, sólo es necesario cambiar una función FUNC en la que está contenida la formulación particular de cada caso. Cuando se tengan restricciones habrá que añadirlas tal y como se indica en los distintos segmentos.

Además de los dos programas ya citados y sobre los que descansa toda la síntesis óptima, se han desarrollado algunos programas auxiliares, necesarios para completar el proceso, como son el programa ACUSE, ya citado anteriormente, y los programas CHECU (CHEqueo de CUrvas) y - PERSP (Representación en PERSPectiva).

En lo que sigue se hace una descripción de - los programas utilizados y las distintas partes que los comparece. Han sido realizados en lenguaje FORTRAN, como es - usual en aplicaciones científicas del ordenador. En los -- Apendices se incluyen los listados de estos.

4.2. PROGRAMA PARA LA BUSQUEDA DE LA PRIMERA SOLUCION.

El programa OPRIS que realiza la busqueda - de la primera solución, consta de un programa principal, -- seis segmentos y 28 subrutinas. El programa se puede ejecutar secuencialmente tantas veces como se indique en la or-- den de ejecución, pudiendo por tanto resolver tantos casos de síntesis como se desee, siempre que sean sobre un mismo tipo de mecanismo. El cambio de tipo de mecanismo exige la

incorporación de una nueva función FUNC. Este programa está concebido de forma que se puede ejecutar cada segmento de manera independiente y tantas veces como se indique en los datos de entrada. Cada segmento ejecuta uno de los algoritmos citados en el capítulo 3.

4.2.1. PROGRAMA PRINCIPAL. OPRIS.

Como programa principal que es, controla todo el proceso de obtención de la primera solución a la síntesis óptima. En él están definidas todas las áreas COMMON necesarias tanto las específicas de cada segmento, como las comunes entre ellas. Realiza la lectura de los datos, tanto los propios del problema a resolver como las características de la ejecución. Tanto la lectura como la escritura las realizará por las unidades lógicas que se indican en la orden de ejecución del programa. Genera, de acuerdo con los valores límites leidos, la familia de vectores aleatorios iniciales. También existe la opción de realizar una lectura directa de estos vectores como datos, si así se desea.

Cuando no hay restricciones, llama al primer segmento OPTI1 (NR), que desarrolla el método Newton-Raphson. Según sean las soluciones que este obtenga y de acuerdo con un conjunto de parámetros de ejecución, leidos previamente, efectúa diversas opciones. Si el primer segmento no obtiene soluciones aceptables, llama a cualquiera de los segmentos OPTI2(MV), OPTI3(GC), OPTI4(BD) u OPTI5(DC) según se especifique. También existe la posibilidad de utilizar las soluciones obtenidas por NR como iniciales en - -

cualquiera de los otros métodos a fín de ajustarlas mejor. Otra opción posible es utilizando un mismo vector inicial, ejecutar todos los segmentos, desde el 1 al 5.

Para cada vector de la familia generada aleatoriamente se efectúa este ciclo de operaciones. Ciclos -- que se repiten tantas veces como sea necesario y de acuerdo a un número de vectores iniciales fijado. Si hay restric-- ciones se efectúa directamente la resolución con el sexto - segmento OPTI6(COM). Una vez obtenida la primera solución buscada, comprueba si es el último caso a resolver. De ser - así, finaliza la ejecución, si no vuelve al principio a leer los datos del caso siguiente a resolver. La Figura 4-1 representa un diagrama de flujo del programa principal.

El programa llama a las subrutinas RMPAR, OVLAY y LEER. Las dos primeras son propias del sistema, la primera se encarga de transmitir, hasta cinco parámetros, al programa dados en la orden de ejecución. La segunda gestiona la llamada a los segmentos de forma que devuelve el control de programa principal, después de la ejecución de cada segmento, lo que permite llamarlas tantas veces como se desee, así como no utilizar aquellos que no se deseen.

La tercera, LEER, es la que realiza toda la lectura de datos. Lee todos los parámetros de ejecución necesarios, y los propios de cada segmento. A continuación y tras escribir un encabezado conteniendo el título y las características de definición del problema, llama a la DATS, que se encarga de leer los datos referentes a los puntos de

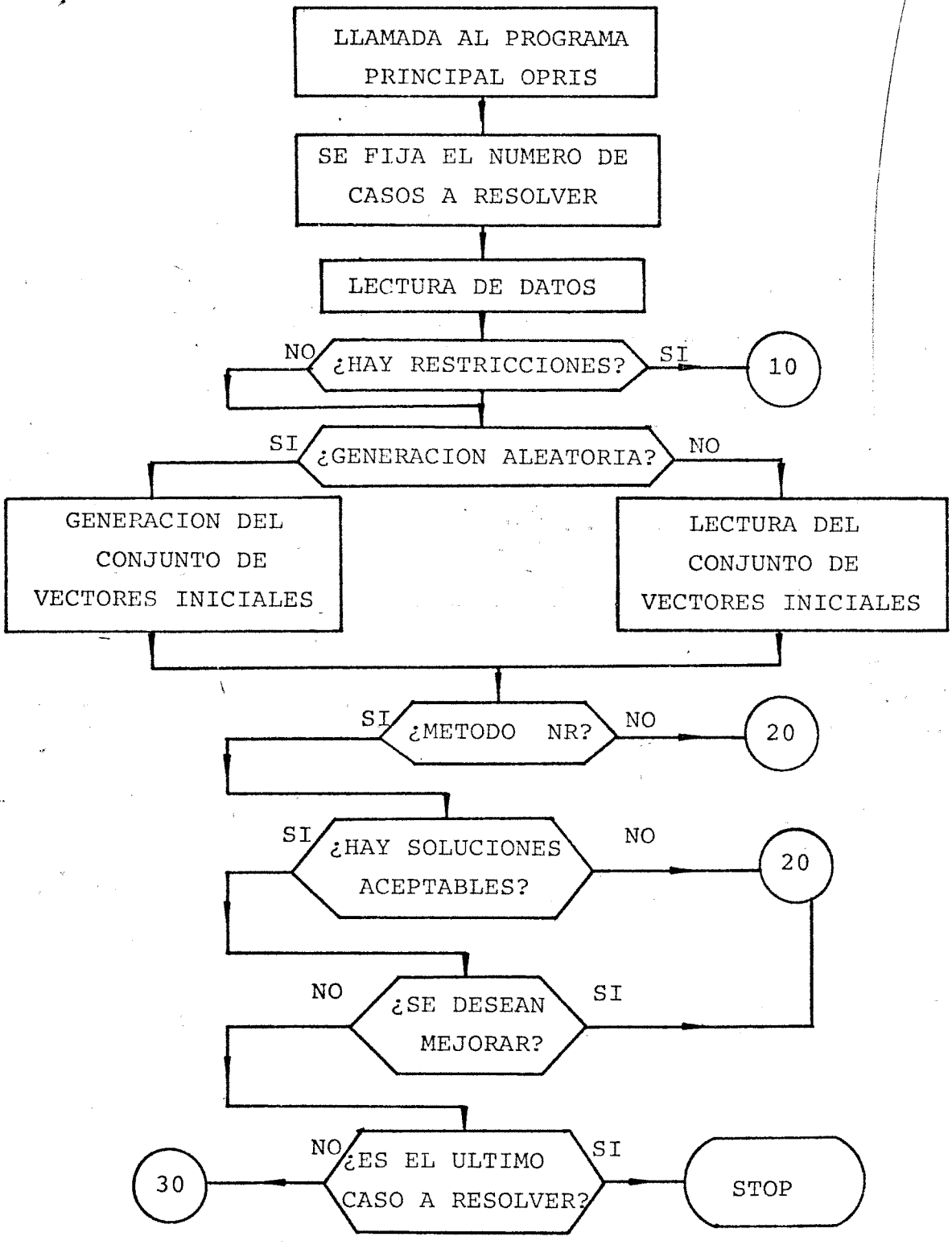

LLAMADA AL PROGRAMA PRINCIPAL OPRIS
SE FIJA EL NUMERO DE CASOS A RESOLVER
LECTURA DE DATOS
NO
¿HAY RESTRICCIONES?
SI
10
SI
¿GENERACION ALEATORIA?
NO
GENERACION DEL CONJUNTO DE VECTORES INICIALES
LECTURA DEL CONJUNTO DE VECTORES INICIALES
SI
¿METODO NR?
NO
20
SI
¿HAY SOLUCIONES ACEPTABLES?
NO
20
NO
¿SE DESEAN MEJORAR?
SI
NO
¿ES EL ULTIMO CASO A RESOLVER?
SI
30
STOP

FIGURA 4-1

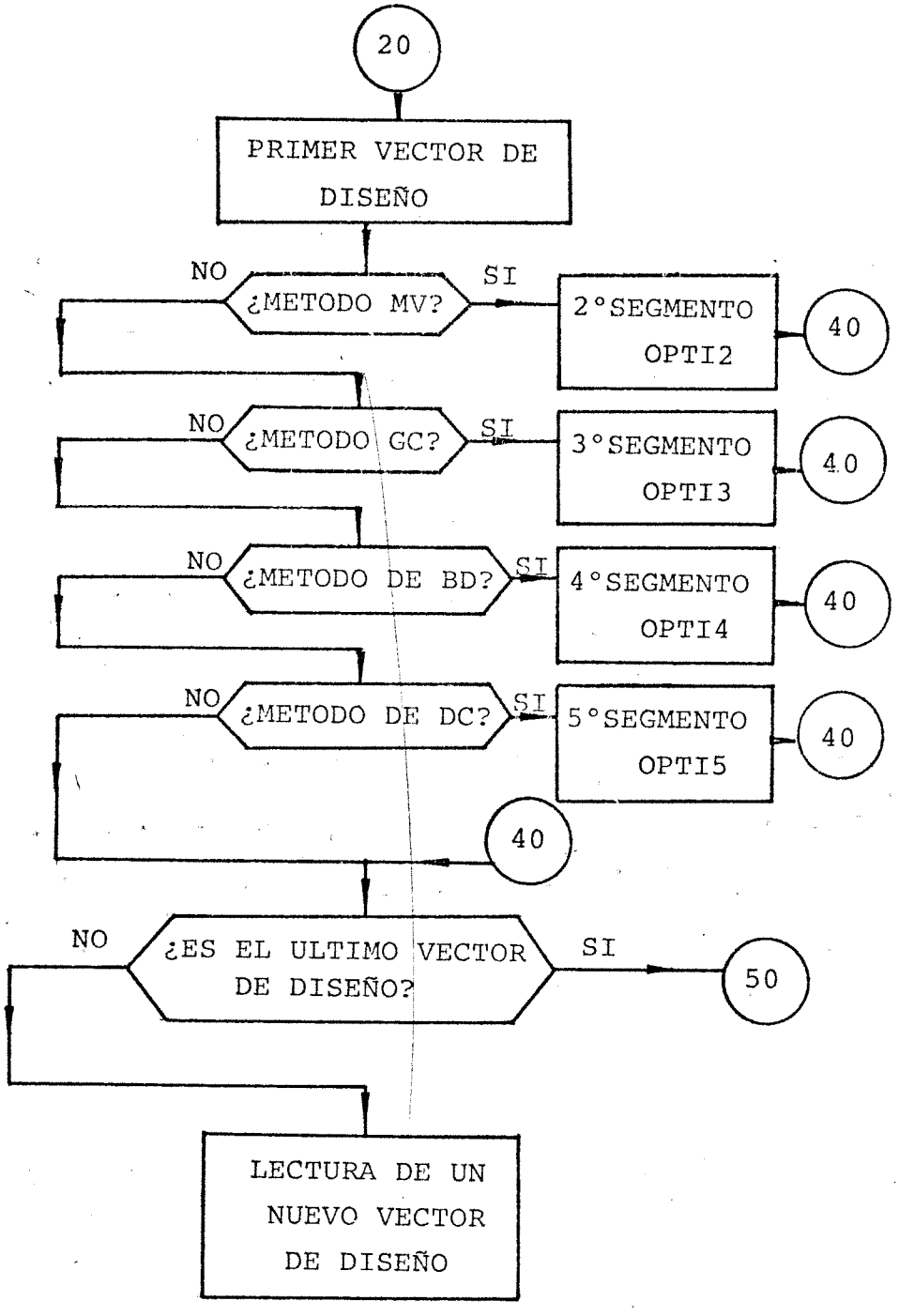


FIGURA 4-1 Cont.

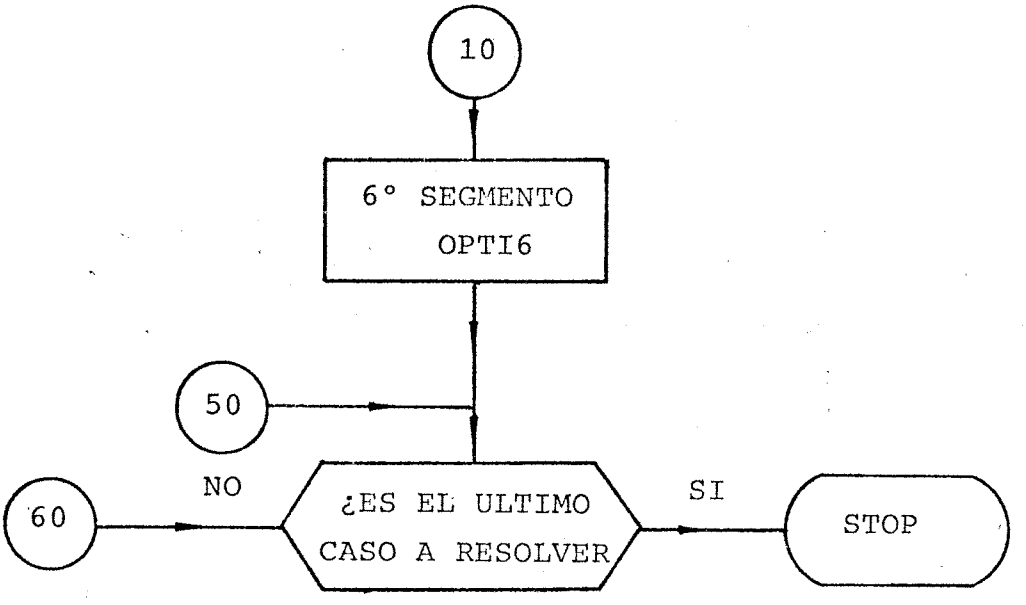


FIGURA 4-1 Cont.

precisión, de la trayectoria a generar, tomados. Con la -- llamada a la ALEAT, que genera números aleatorios entre 0 y 1, y la lectura de los límites superiores e inferiores,- realiza la generación de la familia de vectores iniciales aleatorios. Si no se desea generación aleatoria de los -- vectores, los lee directamente por la unidad de lectura. - En la DATS, también lee los parámetros relativos a las B.D. V., indicando el tipo de ley de variación elegida para ca- da barra. Esta última subrutina DATS, conviene que sea - específica en cada caso y por tanto se suministra al pro-- grama junto con la FUNC.

4.2.2. SEGMENTO OPTI1-

El primer segmento realiza la resolución del sistema de cuaciones algebraícos no lineales, mediante el - método de Newton-Raphson. El programa propiamente dicho, - no realiza otra cosa, que la evaluación de la memoria nece saria, para esta resolución. Almacena todos los conjuntos que solo se usan en este segmento en una sola área COMMON, en la que se irá volcando el almacenamiento específico - de cada segmento, que no hay que transmitir al programa - - principal. Se puede decir que el concepto de memoria diná- mica se aplica aquí a cada segmento en vez de hacerlo al -- programa principal, como es lo usual. Esta disposición per mite trabajar con cada segmento como si fuese un programa - independiente. El diagrama de flujo de este segmento se in dica en la Figura 4-2.

Con la llamada a la subrutina ALNER trans- fiere toda la ejecución a esta, ya que es aquí donde real--

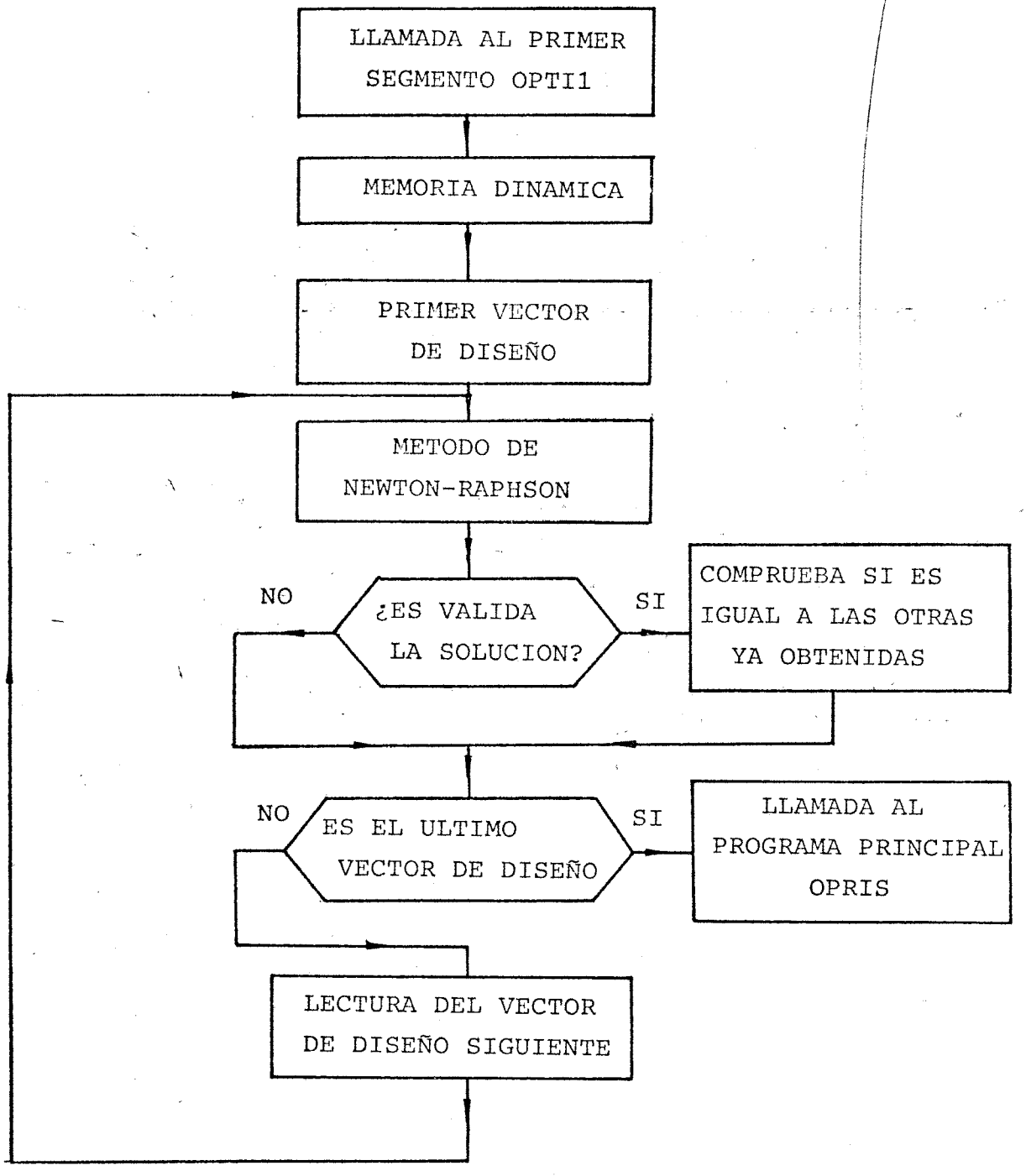

LLAMADA AL PRIMER SEGMENTO OPTI1
MEMORIA DINAMICA
PRIMER VECTOR DE DISEÑO
METODO DE NEWTON-RAPHSON
¿ES VALIDA LA SOLUCION?
NO
SI
COMPRUEBA SI ES IGUAL A LAS OTRAS YA OBTENIDAS
ES EL ULTIMO VECTOR DE DISEÑO
NO
SI
LLAMADA AL PROGRAMA PRINCIPAL OPRIS
LECTURA DEL VECTOR DE DISEÑO SIGUIENTE

FIGURA 4-2

mente se realiza la resolución del sistema de ecuaciones. - En la versión actual, puede resolver sistemas de hasta 30 - ecuaciones. Como se dispone de una familia de vectores iniciales, para cada uno de ellos se efectúa la resolución o - aplicación de NR, mediante la llamada a la subrutina ANERA. Tras la ejecución de esta subrutina, con el resultado obtenido se comprueba su convergencia y bondad. Esta solución se compara con las que ya se tengan determinadas a partir - de otros vectores iniciales para verificar que no es la misma. Si es igual a otra ya obtenida se desecha y no se considera como solución obtenida. Detectandose de esta manera cuantas soluciones distintas encuentra para el sistema de - ecuaciones propuesto a partir de la familia de vectores iniciales.

De acuerdo con los parámetros y datos leidos, admite diversas opciones de escritura, de forma que se pueden escribir todas las iteracciones paso a paso, ó solo los resultados finales de cada ciclo de resolución. También -- existe la posibilidad de que imprima tan solo las soluciones obtenidas en las que la norma del sistema es menor que un valor dado. Igualmente puede dar diversos mensajes sobre la convergencia, bien sea, que el número de iteraciones fijado es pequeño bien sea que se supera el límite de co- - rrección retardada. Al final, y además de las anteriores -- posibilidades de escritura, realiza un resumen del número de soluciones obtenidas para los ciclos de resolución previstos. Las distintas soluciones obtenidas las ordena de mejor, o más aproximada, a peor y las imprime.

La subrutina ANERA realiza la resolución - de un sistema de ecuaciones algebraícas no lineales, con --

corrección retardada, que como es sabido consiste en la -- aplicación progresiva de la corrección, cuando se detecta - que si se aplica directamente la norma del vector crece y no converge por tanto la solución. Llama a la subrutina - GRADI que se encarga de obtener las derivadas de las ecuaciones respecto de las distintas variables. Estas derivadas se obtienen de forma numérica mediante diferencias finitas. También realiza la evaluación del valor de cada -- ecuación para su vector de diseño determinado. Para el -- cálculo de la corrección en cada caso, hay que resolver un sistema de ecuaciones lineales, que realiza la subrutina - SYST. La resolución la efectúa mediante la reducción de - Gauss con pivotamiento total.

GRADI, es la subrutina, que llama a la función FUNC - en donde está expresado el sistema de ecuaciones algebraícas no lineal a resolver, para cada caso de síntesis. - Ya se comentó que esta función tiene que suministrarla el - usuario, para cada tipo de mecanismo a resolver. Como las barras del mecanismo pueden ser B.D.V. la función FUNC -- llamará a la subrutina LONG que evalúa, para cada tipo de ley de variación, la dimensión de la barra en el instante - deseado, de acuerdo con los parámetros de definición de esa ley. La LONG, y para el caso en el que la ley sea del tipo D (general), efectúa la llamada a la SPLIN, que realiza la interpolación de una función de polinomios de spline cúbi-- cos por los puntos que se le transmiten, devolviendo el valor de la dimensión de la barra para el instante pedido.

4.2.3. SEGMENTO OPTI2.

Este programa realiza la resolución del sistema de ecuaciones como si de un problema de optimización se tratara. Aplica el algoritmo de la Métrica Variable (MV). El segmento OPTI2 en si,tan sólo hace la evaluación de la memoria dinámica necesaria para esta optimización. Después llama a la subrutina SCFPS, devolviendo a continuación el control al programa principal OPRIS. La Figura 4-3, es el diagrama de flujo de este segundo segmento.

La subrutina SCFPS realiza la escritura del encabezado correspondiente al caso a resolver con este segmento, y de los resultados obtenidos, número de iteraciones que han sido necesarias, valor de la función objetivo y demás mensajes sobre la convergencia. Llama a FPSMV que es la que realmente realiza el algoritmo de optimización de la métrica variable (MV). En esta ejecución llama a la subrutina FUOBJ que es la que efectúa el cálculo de la función objetivo, en este caso la norma del sistema de ecuaciones,tal y como se comentó en el capítulo 3. FUOBJ llama a GRADI subrutina que ya se usó en el primer segmento OPTI1. El proceso de llamadas a función FUNC y subrutinas LONG y SPLIN es el mismo ya descrito anteriormente.

4.2.4. SEGMENTO OPTI3.

Realiza la determinación de la solución del sistema de ecuaciones no lineales mediante el algoritmo de optimización del Gradiente Conjugado (GC). Es esencialmen-

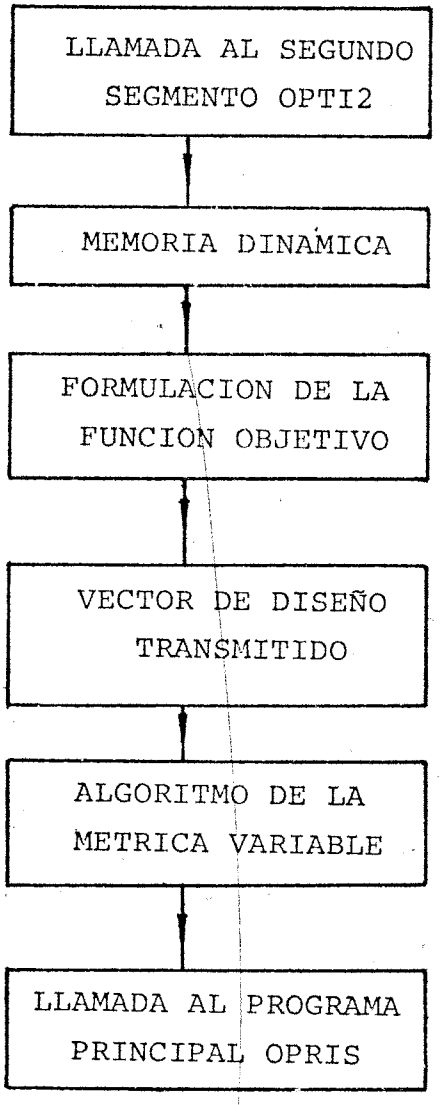

LLAMADA AL SEGUNDO SEGMENTO OPTI2
MEMORIA DINAMICA
FORMULACION DE LA FUNCION OBJETIVO
VECTOR DE DISEÑO TRANSMITIDO
ALGORITMO DE LA METRICA VARIABLE
LLAMADA AL PROGRAMA PRINCIPAL OPRIS

FIGURA 4-3

te, en cuanto a organización, igual al segmento anterior -- OPTI2. Así, el programa OPTI3 tan sólo se usa para la determinación de la memoria dinámica. Llama a SCFRS y devuelve el control al programa principal. El diagrama de - flujo de este segmento se indica en la Figura 4-4.

SCFRS efectúa la escritura del encabezado y datos de resolución junto con la solución obtenida y demás resultados, y llama a FRSGC.. Esta es la que ejecuta la optimización según el algoritmo GC, y utiliza la subrutina - FUOBJ para determinar la función objetivo y sus derivadas respecto de las distintas variables de diseño. Esta FUOBJ es la misma que se usa en el segmento OPTI2 anterior.

4.2.5. SEGMENTO OPTI4.

En este segmento se efectúa la resolución -- del sistema de ecuaciones no lineales, mediante la método - de optimización BD (Busqueda Directa) anteriormente citado. OPTI4 al igual que OPTI2 y OPTI3, tan sólo efectúa la - evaluación de la memoria dinámica de este proceso, y pasa - el control a SCBUS devolviendólo, al finalizar esta última subrutina, al programa principal OPRIS.

SCBUS realiza la escritura del encabezado y de los resultados que se obtienen, tanto parciales como finales, y llama a BUSDI, que efectúa la busqueda directa según el algoritmo desarrollado por Hooke-Reeves. Para eso hace necesario que BUSDI llame a la subrutina FUOB que es sumamente similar a la FUOBJ que se utiliza en los dos segmentos anteriores, pero sin realizar la evolución de las

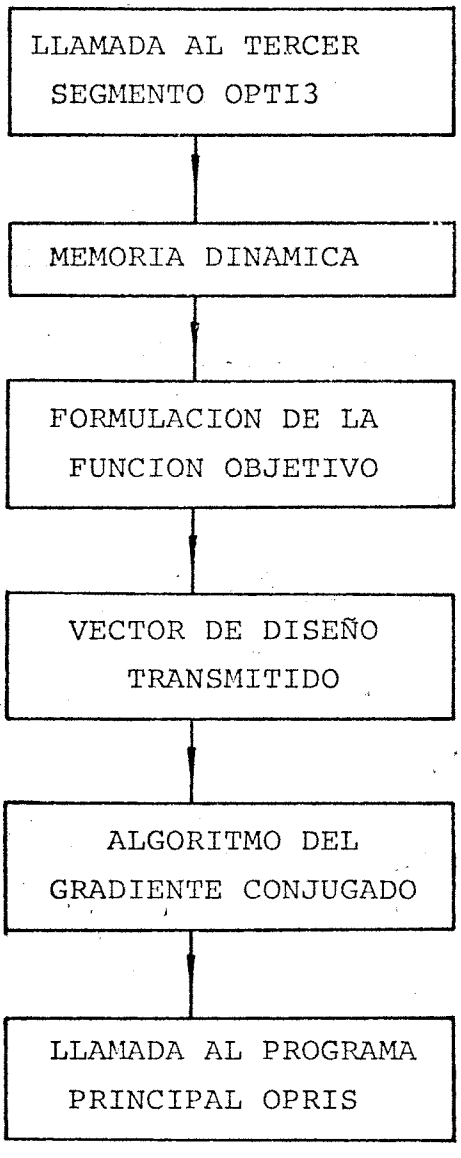

LLAMADA AL TERCER SEGMENTO OPTI3
MEMORIA DINAMICA
FORMULACION DE LA FUNCION OBJETIVO
VECTOR DE DISEÑO TRANSMITIDO
ALGORITMO DEL GRADIENTE CONJUGADO
LLAMADA AL PROGRAMA PRINCIPAL OPRIS

FIGURA 4-4

derivadas de la función objetivo, pues este algoritmo no - las necesita. FUOB usa la función FUNC de definición - del sistema de ecuaciones, que además usa las subrutinas - LONG y SPLIN para la utilización de las B.D.V. En la Figura 4-5 se esquematiza el diagrama de flujo de este segmento.

4.2.6. SEGMENTO OPTI5.

Este segmento se usa, si se desea efectuar la resolución del sistema de ecuaciones resultante de -- plantear la síntesis con puntos de precisión, mediante el algoritmo de optimización DC (Direcciones Conjugadas). -- OPTI5 dimensiona la memoria dinámica, llama a la subrutina de ejecución SCPOW, y después devuelve el control al - programa principal. La Figura 4-6 representa el diagrama de flujo de este segmento.

La subrutina SCPOW escribe el resumen de los resultados obtenidos y llama a POWDC que realiza la optimización según el algoritmo desarrrollado por Powell,- además escribe el encabezado relativo a la resolución con este algoritmo. En su ejecución POWDC utiliza la subrutina FUOB, que es la misma que efectúa la evaluación de - la función objetivo en el anterior segmento OPTI4, llamando igualmente a la función FUNC y demás. Este algoritmo tampoco necesita la evaluación de las derivadas de la función objetivo.

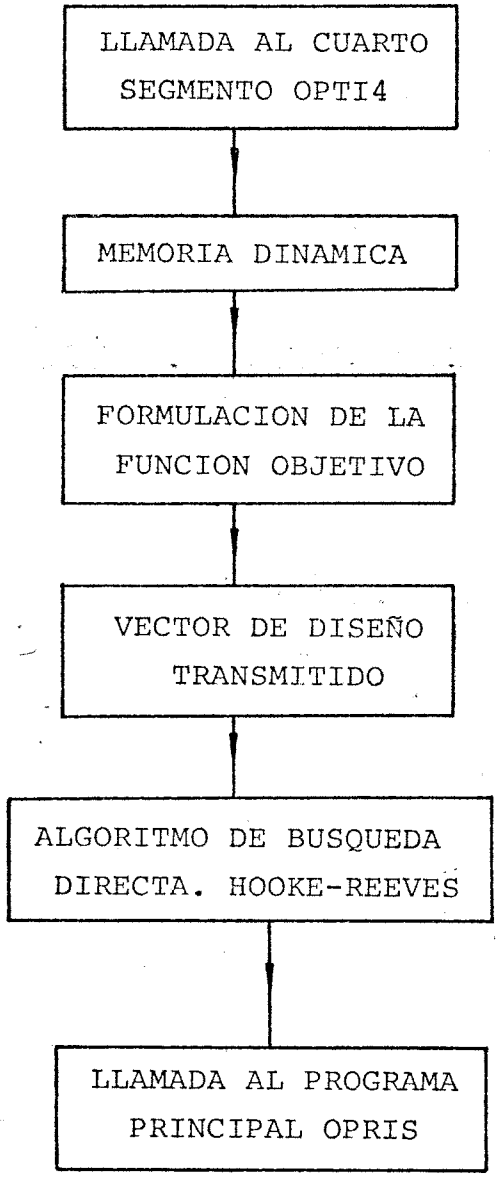

LLAMADA AL CUARTO SEGMENTO OPTI4
MEMORIA DINAMICA
FORMULACION DE LA FUNCION OBJETIVO
VECTOR DE DISEÑO TRANSMITIDO
ALGORITMO DE BUSQUEDA DIRECTA. HOOKE-REEVES
LLAMADA AL PROGRAMA PRINCIPAL OPRIS

FIGURA 4-5

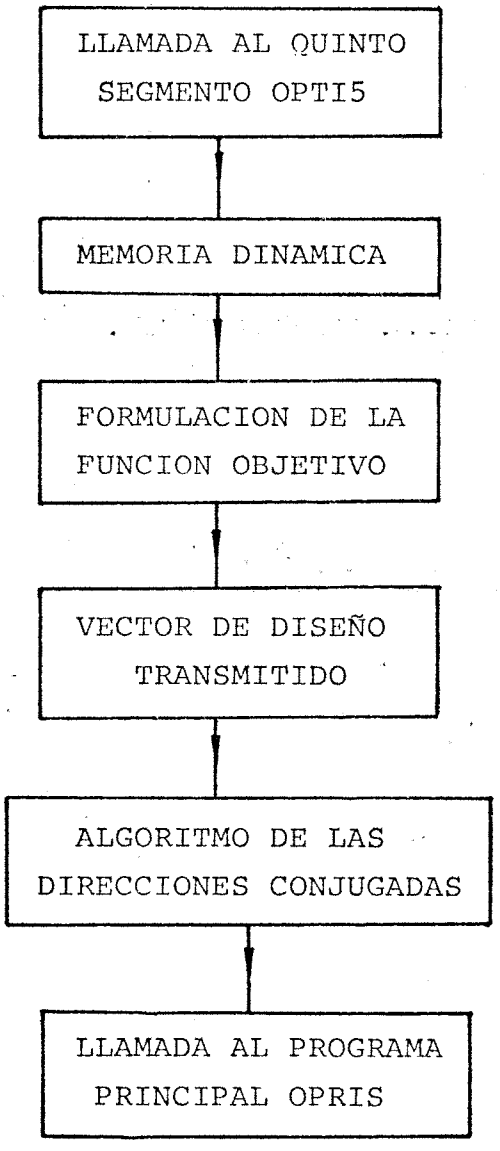

LLAMADA AL QUINTO SEGMENTO OPTI5
MEMORIA DINAMICA
FORMULACION DE LA FUNCION OBJETIVO
VECTOR DE DISEÑO TRANSMITIDO
ALGORITMO DE LAS DIRECCIONES CONJUGADAS
LLAMADA AL PROGRAMA PRINCIPAL OPRIS

FIGURA 4-6

4.2.7. SEGMENTO OPTI6.

Cuando hay restricciones al diseño, que no se pueden eliminar haciendo transformaciones y cambios de variables, se acude, al uso de este segmento, para la resolución del sistema de ecuaciones no lineales que plantea la síntesis con puntos de precisión. OPTI6 efectúa esta resolución mediante el algoritmo Complex de optimización con restricciones. El programa OPTI6, al igual que los segmentos anteriormente descritos, lo único que hace es la evaluación de la dimensión de la memoria dinámica necesaria, llamando a la subrutina SCCOM y devolviendo finalmente el control al programa principal OPRIS. El diagrama de flujo de la Figura 4-7 lo representa.

La subrutina SCCOM se encarga de la escritura del encabezado del problema como de los resultados. Efectúa la generación aleatoria del conjunto de puntos en el espacio R^n, ya que a este segmento no se le transmiten los vectores de diseño aleatorios. Necesita un vector de diseño inicial que cumpla las restricciones. Para esta generación aleatoria llama a la subrutina ALEAT que también usa el primer segmento OPTI1. Con la llamada a la subrutina ALCOM se transfiere a esta toda la ejecución del citado algoritmo. ALCOM utiliza en su ejecución varias subrutinas como es RESTR, con la que vienen expresadas las restricciones formuladas sobre las variables de diseño, Esta subrutina tiene que escribirse de acuerdo con estas restricciones y tiene que suministrarla el usuario para cada caso distinto a resolver. La subrutina COMRE verifica que los vectores que se van obteniendo en los sucesivos pasos cum--

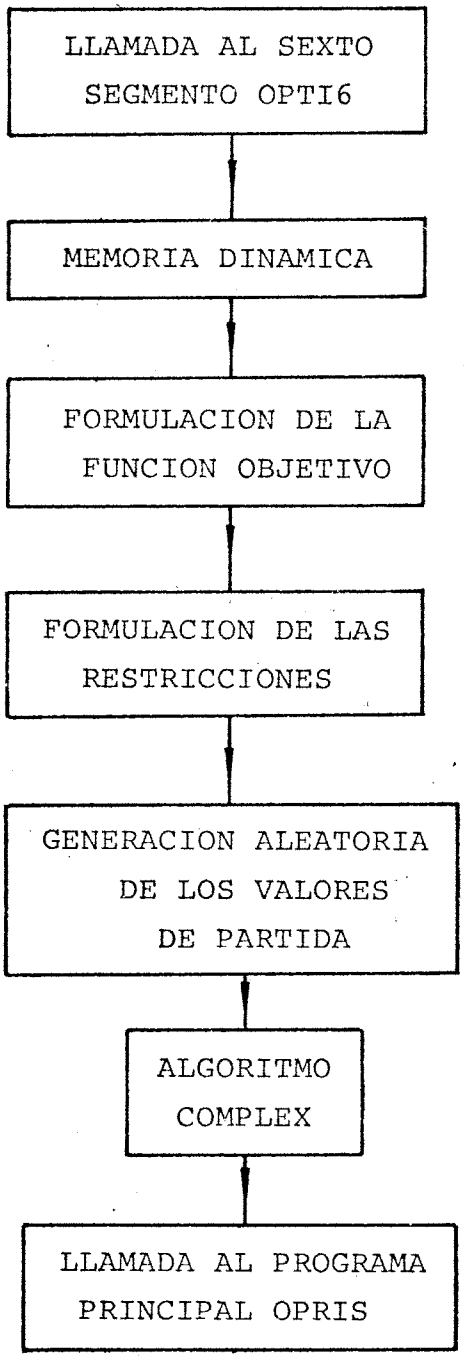

LLAMADA AL SEXTO SEGMENTO OPTI6
MEMORIA DINAMICA
FORMULACION DE LA FUNCION OBJETIVO
FORMULACION DE LAS RESTRICCIONES
GENERACION ALEATORIA DE LOS VALORES DE PARTIDA
ALGORITMO COMPLEX
LLAMADA AL PROGRAMA PRINCIPAL OPRIS

FIGURA 4-7

plen las restricciones impuestas y se mantengan dentro de ellas.

La subrutina FUNCI formula la función objetivo a minimizar llamando a FUNC, que usará a su vez de las subrutinas LONG y SPLIN. Este algoritmo tampoco necesita de la determinación de las derivadas de esa función objetivo.

4.2.8. ENTRADA DE DATOS PARA PROGRAMAS OPRIS.

A continuación se describe la entrada de datos necesaria para la ejecución de este programa. El formato de entrada de los datos salvo en la primera tarjeta, es formato libre. En la primera tarjeta el formato es - - 30A2. El título "Tarjeta" es puramente simbólico, pues los datos se le pueden suministrar mediante cualquier otro dispositivo de entrada.

Una tarjeta : Título

TITU : Título del ejemplo a resolver. Hasta 60 carácteres.

Una tarjeta : Parámetro de dimensión y tipo

NCVE : Dimensión del problema. Número de variables del vector de diseño.

MALEC : Párametro para indicar si se generan los - vectores de diseño de forma aleatoria, ó - se van a leer directamente
= 0 generación aleatoria
≠ 0 lee directamente estos vectores.

INCI : Número de vectores de diseño que se van a - utilizar como vectores iniciales.

NVE : Número total de vectores de diseño a gene-- rar aleatoriamente.

IX : Un número entero impar cualquiera necesario para iniciar la generación aleatoria.

NREST : Parámetro que define si hay o no restriccio nes.
= 0 no tiene restricciones
≠ 0 tiene restricciones.

Una tarjeta : Parámetros de ejecución

IMOD : Parámetro para indicar el modo de encadenar los distintos segmentos.
= -1 Todos los segmentos usan vectores de partida los generados aleatoriamente. Es el caso en que el NR no obtiene so lución.
= 0 Mejora de las soluciones obtenidas -- con NR.

IOP(5) : Conjunto entero de 5 variables para indicar que método o segmento se utilizará.
= 1 se usa ese método
= 0 no se usa ese método.

Una tarjeta : Datos específicos para Newton-Raphson (NR)

NITR : Número máximo de iteraciones que se admite en el método de Newton-Raphson.

TOLER : Tolerancia en el valor de la norma del sistema de ecuaciones.

DINCR : Valor del incremento finito necesario para el cálculo de las derivadas por diferencias finitas.

ANORMI : Valor de la norma mínima para considerar aceptable la solución obtenida.

ESCR : Indicador de formato de escritura
> 0 solo escribe los resultados finales
< 0 escribe todos los pasos intermedios.

Una tarjeta : Datos específicos para Métrica Variable(MV)

VMIFO : Valor mínimo estimado de la función objetivo.

ERRAB : Valor que representa el error absoluto en un movimiento.

NMITE : Número máximo de iteraciones a realizar

DINCI : Valor del incremento finito para las derivadas.

Una tarjeta : Datos específicos para Gradiente Conjugado (GC)

VMIF2 : Valor mínimo estimado de la función objetivo.

ERRA2 : Valor que representa el error absoluto en un movimiento.

NMIT2 : Número máximo de iteraciones que realiza.

DINC2 : Valor del incremento finito para las derivadas.

Una tarjeta : Datos específicos para Busqueda Directa(BD)

ITES : Párametro para indicar la forma de escritura deseada
=1 escribe todo paso a paso
=0 escribe solo resultados intermedios
=-1 solo escribe al final.

LIMITE : Número máximo de iteraciones a realizar.

VINPA : Valor inicial del paso.

VMIPA : Valor del paso mínimo.

Una tarjeta : Datos específicos para Direcciones Conjugadas (DC)

INESC : Indicador del tipo de escritura deseada
=0 solo escribe al final
=1 escribe paso a paso
=2 escribe resultados intermedios

NLITE : Número límite de iteraciones a realizar.

VMAPA : Valor máximo del paso.

NCVE Tarjetas : Continuación DC

VLCV(NCVE) : Conjunto de los valores límites de convergencia de cada variable. Un valor por tarjeta.

Una tarjeta : Datos específicos para Complex (COM). Solo si hay restricciones.

MS : Número de restricciones.

KSF : Número de puntos a generar. Se recomienda KSF = 2*NCVE+1.

IXX : Un número impar entero cualquiera, necesario para iniciar la generación aleatoria.

Una Tarjeta : Continuación COM. Solo si hay restricciones.

ITMAX : Número máximo de iteraciones deseadas.

ICS : Número de restricciones implícitas.

IPRJ : Control de escritura
=1 escribe paso a paso
=0 solo escribe los resultados finales

ALFHA : Factor de reflexión. Se recomienda 1.3.

Una tarjeta : Continuación COM. Solo si hay restricciones.

BETA : Parámetro de convergencia. Aproximadamente 10^{-4} por el valor estimado de la función objetivo.

GAMMA : Parámetro de convergencia. Se recomienda 5.

DELTA : Valor para la corrección de las restricciones explícitas que no se cumplen. 10^{-4} por el orden de magnitud de los componentes del vector de diseño.

NCVE Tarjetas : Continuación COM. Solo si hay restricciones.

X(1, NCVE) : Valores de los componentes del vector de diseño que tienen que cumplir las restricciones. Tantas tarjetas como NCVE.

A continuación se leerán los datos de especificacion del número de B, D, V,, de la ley de variación de cada una de éstas y de los puntos de precisión de la trayecto-

ria. Son los valores que leerá la subrutina DATS y que puesto que se incorpora al programa junto con la función FUNC, - tendrá los formatos de lectura, que para cada caso, se fijen por el usuario, de acuerdo con los datos necesarios en FUNC. Después se continúa la lectura de los datos generales.

NCVE Tarjetas : Valores de los límites inferior y superior de las variables de los vectores de diseño a generar aleatoriamente.

ZMIN(NCVE) : Valor mínimo variable NCVE

ZMAX(NCVE) : Valor máximo variable NCVE

NCVE x NVE Tarjetas : Valores de los componentes de los vectores de diseño iniciales, cuando se leen directamente. Solo si MALEC es distinto de cero.

ZMA(K,KK) : Valor de la variable K del vector KK.

4.3. PROGRAMA PARA LA OBTENCION DE LA SOLUCION FINAL.

El programa CIOPT que realiza los ciclos de optimización para la obtención de la solución a la sintesis óptima, está formado por un programa principal, dos segmentos y un total de 23 subprogramas. Se puede ejecutar tantas veces como se indique en la orden de ejecución, pudiendo por tanto resolver tantos casos de síntesis óptima como se deseen, siempre que sean sobre un mismo tipo de mecanismo. Para efectuar el cambio de tipo de mecanismo, función objetivo, hay que reemplazar la función FUNC que se esté usando, por -

la correspondiente a ese nuevo tipo. Además, como admite restricciones, habrá que modificar las subrutinas y funciones correspondientes a las restricciones.

Este programa está realizado de tal forma, que se puede ejecutar cada segmento de manera independiente, y tantas veces como sea necesario, para realizar los ciclos de optimización previstos. Cada uno de los segmentos ejecuta un algoritmo de los citados en el capitulo 3.

4.3.1. PROGRAMA PRINCIPAL CIOPT.

El programa principal CIOPT controla todo el proceso de obtención de la solución final, al problema de la síntesis óptima mediante los ciclos de optimización. En él se definen todas las áreas COMMON necesarias, tanto las específicas de cada segmento como las comunes entre ellos. Lee los datos necesarios para la ejecución del programa, así como para la resolución del problema en cuestión. La lectura y escritura se realizan por las unidades lógicas que se transmiten en la orden de ejecución del programa.

Trás la lectura de los datos y para el número de puntos de comparación efectúa la síntesis óptima. Para ello llama al primer segmento, si se desea resolver la optimización con el algoritmo MVP (Métrica Variable con funciones de Penalización para admitir restricciones), En caso contrario llama al segundo segmento que la efectúa mediante el algoritmo RO (algoritmo de Rosenbrockmodificado para admitir restricciones). Una vez obtenida una solución por cualquiera de los dos métodos y para el número de puntos de comparación especificado, si la solución es aceptable, finaliza la ejecución de este caso, si no lo es, aumenta el número de puntos de comparación y tomando el vector de diseño obtenido como -

solución inicial, realiza la optimización de nuevo. Realizará tantos ciclos como se indiquen hasta obtener una solución adecuada. Al finalizar cada caso comprueba si era el último a resolver, si lo es finaliza la ejecución y si no lo es lee los datos del siguiente y comienza de nuevo. La figura 4-8 representa el diágrama de flujo del programa CIOPT.

En su ejecución, el programa principal, llama a las subrutinas RMPAR, OVLAY y LEERD. Las dos primeras son propias del sistema y ya se comentó su utilización en el programa OPRIS antes descrito. La subrutina LEERD realiza toda la lectura de datos. Lee todos los parámetros de ejecución necesarios y los propios de cada segmento. A continuación realiza la lectura de los datos específicos de cada caso a estudiar mediante la DATS. Esta realiza la lectura de los datos referentes a la trayectoria y a las B.D.V. y sus leyes de variación. Es específica para cada caso a resolver y por ello se suministra al programa junto con la función FUNC. Después de la lectura de datos escribe el encabezado y los datos de resolución y devuelve el control al programa principal CIOPT.

4.3.2. SEGMENTO OPFI1.

El primer segmento realiza la optimización de acuerdo al algorimo MVP de la métrica variable en la que la función objetivo se modifica mediante funciones de penalización para tratar las restricciones. El uso del algoritmo de optimización se refina y mejora con el escalado de las variables, opción que este programa contempla. Después de la evaluación de la memoria dinámica necesaria para su ejecución, se realiza este escalado. A continuación determina los distintos valores de la función objetivo, restricciones y derivadas de éstas, respecto a las variables de diseño. Estas derivadas se --

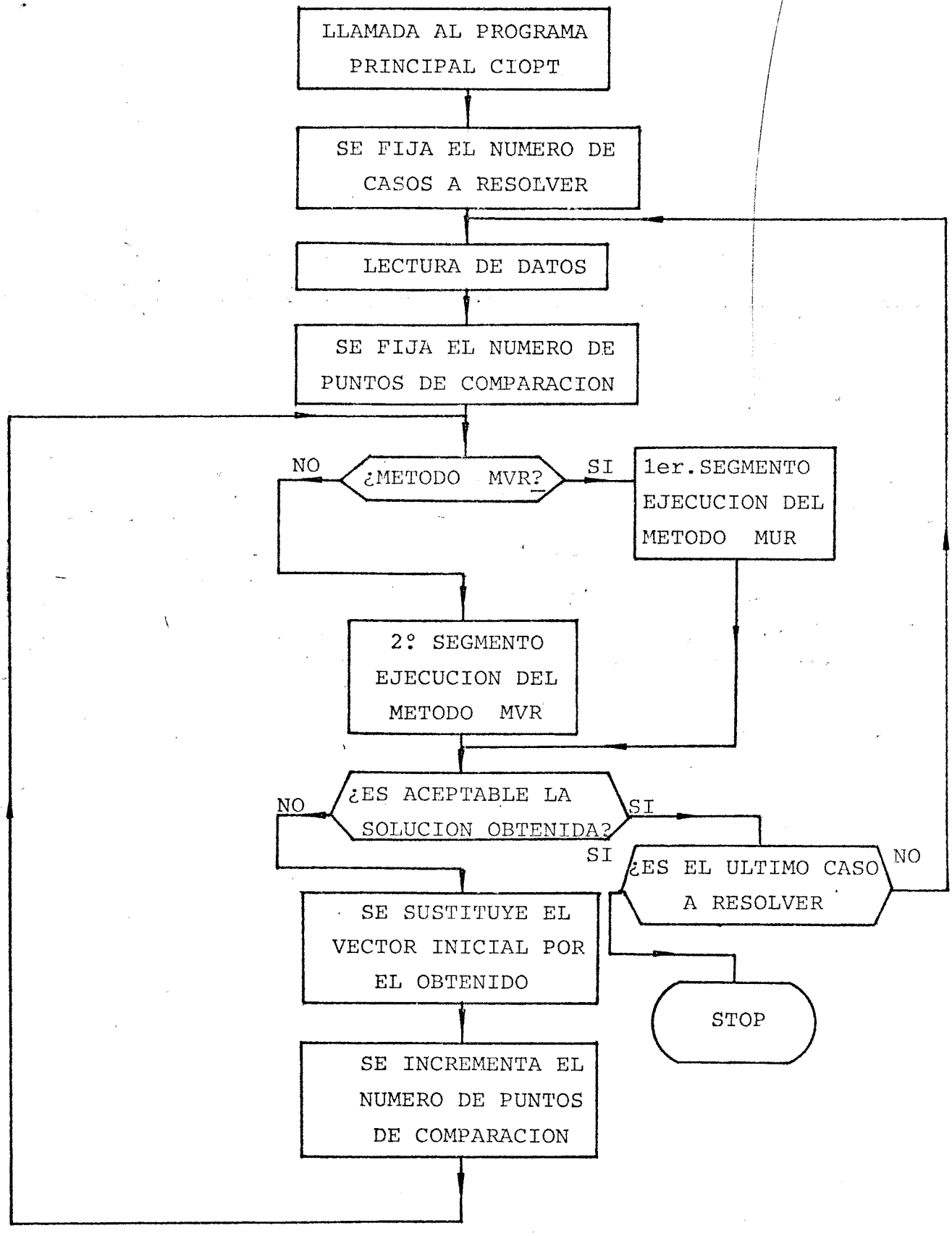

LLAMADA AL PROGRAMA PRINCIPAL CIOPT
SE FIJA EL NUMERO DE CASOS A RESOLVER
LECTURA DE DATOS
SE FIJA EL NUMERO DE PUNTOS DE COMPARACION
NO
¿METODO MVR?
SI
1er. SEGMENTO EJECUCION DEL METODO MUR
2º SEGMENTO EJECUCION DEL METODO MVR
NO
¿ES ACEPTABLE LA SOLUCION OBTENIDA?
SI
SI
¿ES EL ULTIMO CASO A RESOLVER
NO
SE SUSTITUYE EL VECTOR INICIAL POR EL OBTENIDO
STOP
SE INCREMENTA EL NUMERO DE PUNTOS DE COMPARACION

FIGURA 4-8

calculan mediante diferencias finitas. Se realiza la optimización propiamente dicha junto con la determinación de la -- función de penalización y sus parámetros de definición. La - función objetivo que define la síntesis óptima y las restricciones de diseño se evaluan en la subrutina FUOBJ que llama a la función FUNC. Una vez obtenida la solución devuelve el control al programa principal CIOPT para que aumente el número de puntos de comparación, si tal cosa es necesaria. En la Figura 4-9 se representa el diágrama de flujo de este programa.

4.3.3. SEGMENTO OPFIZ.

Es el segundo y último segmento del programa CIOPT. Aplica para la optimización el algorimo de rotación - de coordenadas propuesto por Rosenbrock, ligeramente modificado a fin de que admita las restricciones. El programa OPFI2 tan sólo hace la evaluación de la memoria dinámica necesaria y llama a la subrutina ALROS. Una vez obtenida la solución, devuelve el control al programa principal CIOPT. ALROS realiza la optimización propiamente dicha no usando derivadas, pués al algoritmo no le hacen falta. Además de la función - FUNC y las subrutinas LONG y SPLIN, es necesario que se le suministren las funciones CX, CH y CG para la evaluación de las restricciones específicas del problema a resolver. El - diágrama de flujo de este programa viene representado en la Figura 4-10.

4.3.4. ENTRADA DE DATOS PARA EL PROGRAMA CIOPT.

El formato de entrada de los datos es libre salvo en la primera tarjeta que es 30A2. A continuación se describe la entrada de datos necesaria para la ejecución del programa.

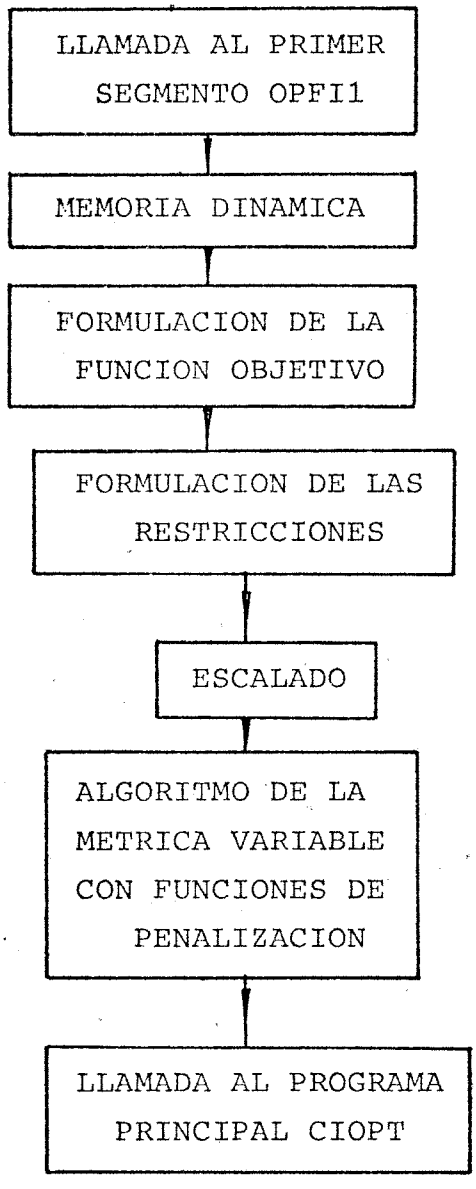

LLAMADA AL PRIMER SEGMENTO OPFI1
MEMORIA DINAMICA
FORMULACION DE LA FUNCION OBJETIVO
FORMULACION DE LAS RESTRICCIONES
ESCALADO
ALGORITMO DE LA METRICA VARIABLE CON FUNCIONES DE PENALIZACION
LLAMADA AL PROGRAMA PRINCIPAL CIOPT

FIGURA 4-9

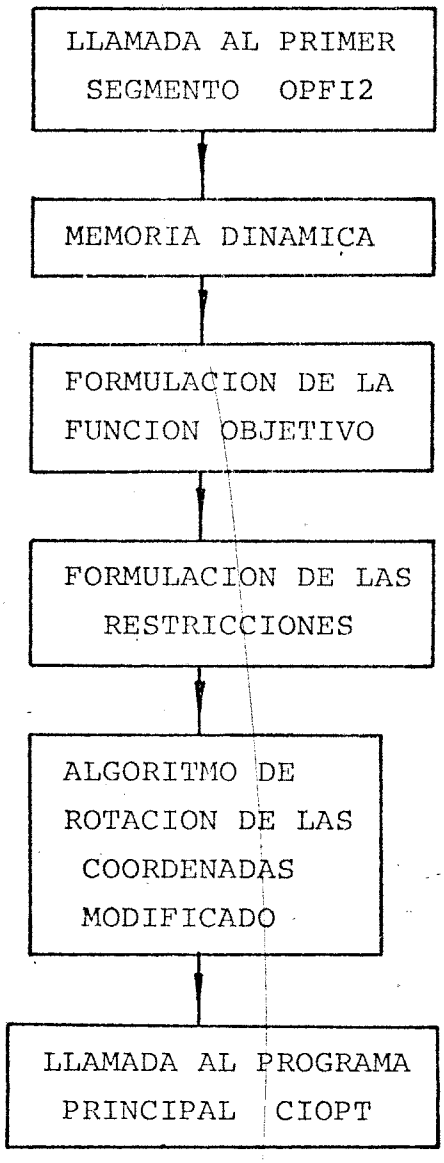

LLAMADA AL PRIMER SEGMENTO OPFI2
MEMORIA DINAMICA
FORMULACION DE LA FUNCION OBJETIVO
FORMULACION DE LAS RESTRICCIONES
ALGORITMO DE ROTACION DE LAS COORDENADAS MODIFICADO
LLAMADA AL PROGRAMA PRINCIPAL CIOPT

FIGURA 4-10

Una tarjeta : Título del caso a resolver

TITU : Conjunto alfanumérico de 60 caracteres.

Una tarjeta : Parámetros de dimensión

NCVE : Dimensión del problema. Número de variables del vector de diseño.

NVE : Número de ciclos de optimización a realizar.

NREST : Número de restricciones.

NTP : Número total de puntos de la curva.

Una tarjeta : Parámetros de ejecución.

IMOD : Parámetro para indicar el modo de encadenar los segmentos.
=1 los encadena
=0 no los encadena

IOP(2) : Conjunto entero de dos variables para indicar que método o segmento se utilizará.

Una tarjeta : Puntos de comparación

INP(NVE) : Conjunto de NVE valores indicando cada uno el número de puntos de comparación por ciclo. Se ordenan de menor a mayor.

NCVE tarjetas : Vector inicial

ZZ(I) : Valor de la variable I del vector inicial de diseño o primera solución.

Una tarjeta : Datos específicos MVP

NOP1 : Número de variables independientes más las de holgura.

MOP1 : Número de restricciones implicitas.

Una tarjeta : Continuación MVP

VEVE : Máximo valor del error absoluto en los valores escalados de la función objetivo y las restricciones.

VXES : Límite superior del escalado de las variables.

VXEM : Límite inferior del escalado de las variables.

CAXB : Constante auxiliar que se usa en la función objetivo modificada. Se recomienda un valor entre 30 y 100.

CDPI : Constante para la determinación de las direcciones en la primera iteración lineal. Se recomienda 10.

Una tarjeta : Continuación MVP

ERRA : Error absoluto en la posición de un punto.

ERRFO : Error máximo en la función objetivo modificada.

VMDM : Valor que representa la mayor distancia al mínimo.

ERRG : Valor que representa el error en los valores de las derivadas de la función objetivo y las restricciones.

Una tarjeta : Continuación MVP

NMITE : Número máximo de iteraciones a realizar.

IMPRE : Código de escritura
=0 solo escribe al final
≠0 escribe resultados intermedios.

INDER : Parámetro para indicar si la matriz H es inicializada a la unidad o lee una dada
=0 inicializa a la unidad
=1 se le suministra.

Una tarjeta : Datos específicos de RO

IIMPR : Código de escritura
=0 sólo escribe el resultado final
≠0 escribe resultados intermedios según ICPE.

ITPRO : Indice del tipo de problema
=+1 es de maximización
=-1 es de minimización.

NMTER : Número máximo de iteraciones.

ICPE : Indice de control de escritura para escribrir solo cada ICPE iteraciones.

ICTPA : Indice de control del tamaño del paso para cada rotación
=0 toma el tamaño inicial
=1 toma el tamaño de la última rotación.

NCVE tarjetas : Continuación RO

EPS(I) : Valor del tamaño inicial del paso en la variable I.

A continuación se leerán los datos de especificación del número de B.D.V., de la ley de variación de cada una de estas y los puntos de definición de la trayectoria. Son los valores que leera la subrutina DATS y que como se incorpora al programa junto con la función FUNC, tendrá los formatos de lectura, que para cada caso, se fijen por el usuario, de acuerdo a los datos necesarios en FUNC.

4.4. PROGRAMAS AUXILIARES.

Junto con los programas OPRIS y CIOPT se utilizan otros tres programas mucho más simples como auxi-

liares en la resolución. El programa ACUSC realiza la - interpolación de una función de polinomios de spline cúbicos a traves de una serie de puntos dados. Su uso es fundamental en la aproximación de la curva y una versión implicada de este programa constituye la subrutina SPLIN necesaria para la simulación de la ley general de variación en las B.D.V. El programa es simple y con los comentarios -- que se incluyen en el listado es fácil de seguir y de usar.

Los otros dos programas son áun más sim- - ples. CHECU tan solo hace la comparación entre la trayectoria deseada y la obtenida, los representa y determina el error que se comete. PERSP que es un programa para dibu-- jar en diversos tipos de perpectivas, realiza la representación del mecanismo en distintas posiciones, ya sea plano o espacial, si se le suministran las coordenadas de los pares y la conectividad entre estos. Sus listados también - se incluyen y son faciles de usar a la vista de ellos.

CAPITULO V.

APLICACIONES Y RESULTADOS.

5.1. INTRODUCCION.

Como comprobación de todo lo anteriormente descrito se han desarrollado diversas aplicaciones. En estas aplicaciones se han considerado distintos tipos de mecanismos, con diferentes números de barras de dimensiones variables y distintas leyes de variación de la dimensión.

Para cada una de las aplicaciones se formula el sistema de ecuaciones que define la sintesis de generación de una trayectoria con ese tipo de mecanismo. Se tiene que tener en cuenta, que barras son de dimensión variable y que tipo de ley de variación se considera.

A continuación se plantea la función objetivo para efectuar la resolucion del sistema y la sintesis óptima. Una vez realizada la programación, en una función FUNC, del sistema de ecuaciones y de la funcion objetivo, se incorporan a los respectivos programas, estando entonces en condiciones de realizar la aplicación número especifica.

En esta aplicación número se muestra de forma concisa los resultados obtenidos y se incluyen diversas representaciones gráficas de los sistemas obtenidos y las aproximaciones que en cada caso producen.

5.2. SINTESIS OPTIMA DE UN MECANISMO DE CUATRO BARRAS PLANO.

Con el objeto de comprobar la bondad de la metodología propuesta para la síntesis óptima de generaciones de trayectorias, se realiza la síntesis de un mecanismo de cuatro barras plano. Como se trata de un ejemplo de comprobación, la trayectoria a generar será la que genere un mecanismo de dimensiones conocidas a fin de evaluar el error que se comete no solo en la aproximación de la trayectoria sino tambien en los valores que definen al mecanismo. En esta primera aplicación se consideran todas las barras de longitud constante.

5.2.1. SISTEMA DE ECUACIONES.

La síntesis de generación de trayectorias con un mecanismo de cuatro barras, como el de la Figura 5-2-1, se puede expresar mediante el sistema de ecuaciones que resulta de imponer las condiciones de constancia en su longitud a las barras de entrada y de salida. Así se indicó, a modo de ejemplo, en el Capitulo 2. Sin embargo, formulando el

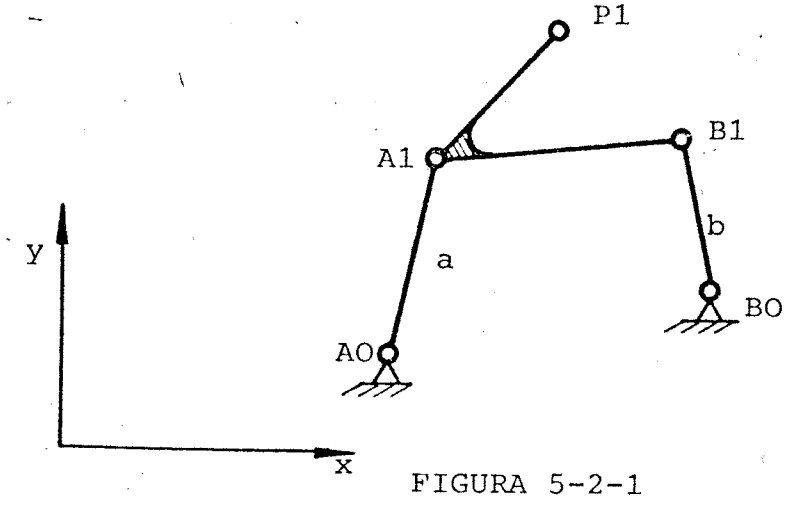


FIGURA 5-2-1

sistema de ecuaciones de otra forma, se pueden conseguir notables ventajas. Así, y de acuerdo con la figura 5-2-1 se puede expresar el sistema de ecuaciones de la forma:

$$f_i(\underset{\sim}{Z}) = 0; \quad i = 1, 2, 3, \ldots\ldots 8 \qquad (5\text{-}1)$$

donde $\underset{\sim}{Z}$ es el vector de diseño que será en este caso

$$\underset{\sim}{Z} = \left[a0x,\ a0y,\ b0x,\ b0y,\ a1x,\ a1y,\ b1x,\ b1y \right]^T \qquad (5\text{-}2)$$

La expresión (5-1) puede ponerse de la forma

$$f_i(\underset{\sim}{Z}) = f_{j1}(\underset{\sim}{Z}) + f_{j2}(\underset{\sim}{Z}) \qquad (5\text{-}3)$$

para $j = i + 1$ es decir, $j = 2, 3, \ldots\ldots 9$; siendo

$$f_{j1}(\underset{\sim}{Z}) = \left[(\underset{\sim}{PJ}-\underset{\sim}{AJ})^T(\underset{\sim}{PJ}-\underset{\sim}{AJ}) - (\underset{\sim}{P1}-\underset{\sim}{A1})^T(\underset{\sim}{P1}-\underset{\sim}{A1}) \right]^2 \qquad (5\text{-}4)$$

donde

$$\underset{\sim}{AJ} = \left[R(\theta 1j) \right] (\underset{\sim}{A1}-\underset{\sim}{A0}) + \underset{\sim}{A0} \qquad (5\text{-}5)$$

y representando PJ los vectores de posición de los puntos de la trayectoria a generar y siendo $\theta 1j$ el angulo girado por la barra de entrada desde la posición inicial, correspondiente al punto j de la trayectoria. Además

$$f_{j2}(\underset{\sim}{Z}) = \left[(\underset{\sim}{BJ}-\underset{\sim}{B0})^T(\underset{\sim}{BJ}-\underset{\sim}{B0}) - (\underset{\sim}{B1}-\underset{\sim}{B0})^T(\underset{\sim}{B1}-\underset{\sim}{B0}) \right]^2 \qquad (5\text{-}6)$$

donde

$$\underset{\sim}{BJ} = \left[R(\phi 1j)\right](\underset{\sim}{B1}-\underset{\sim}{P1})+\underset{\sim}{PJ} \qquad (5\text{-}7)$$

y donde $\phi 1j$ es el angulo girado por el acoplador desde la posición 1 a la posición j, y que puede calcularse de acuerdo a:

$$\phi\, 1j = \text{arctg}\left[\frac{(pjy - ajy)}{(pjx - ajx)}\right] - \text{arctg}\left[\frac{(p1y - a1y)}{(p1x - a1x)}\right] \qquad (5\text{-}8)$$

De acuerdo con esto, la expresión (5-1) se puede desarrollar en un sistema que como máximo tenga ocho ecuaciones y ocho incógnitas. Se puede, por tanto, hacer una síntesis de generación de trayectorias especificando nueve puntos de precisión y el ángulo de entrada correspondiente a cada uno de esos puntos de la trayectoria.

5.2.2. FUNCION OBJETIVO.

Para resolver esto mismo como un problema de optimización se puede plantear de la siguiente manera:

Determinar el vector $\underset{\sim}{Z}$ que hace

$$\text{Min} \quad F(\underset{\sim}{Z}) \tag{5-9}$$

donde

$$F(\underset{\sim}{Z}) = \sum_{i=1}^{i=8} f_i \tag{5-10}$$

y que desarrollada será:

$$F(\underset{\sim}{Z}) = \sum_{j=2}^{j=9} \left[(\underset{\sim}{PJ}-\underset{\sim}{AJ})^T(\underset{\sim}{PJ}-\underset{\sim}{AJ}) - (\underset{\sim}{P1}-\underset{\sim}{A1})^T(\underset{\sim}{P1}-\underset{\sim}{A1}) \right]^2 +$$

$$+ \sum_{j=2}^{j=9} \left[(\underset{\sim}{BJ}-\underset{\sim}{B0})^T(\underset{\sim}{BJ}-\underset{\sim}{B0}) - (\underset{\sim}{B1}-\underset{\sim}{B0})^T(\underset{\sim}{B1}-\underset{\sim}{B0}) \right]^2 \tag{5-11}$$

Cuando en vez de resolver el sistema de ocho ecuaciones con ocho incógnitas se trate de resolver la síntesis óptima, la función objetivo será la misma, pero el número de puntos j será el que se especifique en cada ciclo de optimización. - Realizando la programación de estas expresiones en una función FUNC e incorporándolas a los programas OPRIS y CIOPT -

se puede resolver la síntesis óptima.

5.2.3. APLICACION NUMERICA.

Se pretende realizar la síntesis óptima de generación de la trayectoria de la Figura 5-2-2 con un mecanismo de cuatro barras plano, mediante el planteamiento anteriormente descrito. La trayectoria de la Figura 5-2-2 se conoce en forma de 360 puntos, correspondientes cada uno de ellos a un ángulo girado, en la barra de entrada, de un grado. También se tiene como datos

$$a0x = 1.$$
$$b0x = 4.4641$$
$$p1x = -0.09163$$
$$p1y = 1.90934$$

quedando el vector de diseño expresado por:

$$\underset{\sim}{Z} = \left[a0y,\ b0y,\ a1x,\ a1y,\ b1x,\ b1y \right]^T$$

teniendo por tanto que determinar seis incógnitas. Para obtener una primera solución, se realiza la síntesis con puntos de precisión, tomando como tales los pertenecientes a la trayectoria para ángulos de entrada iguales a 50, 100,-150, 200, 250 y 300 grados y que respectivamente son:

(-0.06373, 2.872139)
(-1.36991, 2.78558)
(-1.76648, 1.92008)
(-1.5273, 0.85815)
(-0.79690, 0.23971)
(-0'12538, 0.51530)

Para la generación aleatoria de la familia de vectores de diseño inicial, los límites que se toman son:

$$
\begin{aligned}
0.0 &< a0y < 4.0 \\
0.0 &< b0y < 4.0 \\
-3.0 &< a1x < 5.0 \\
-3.0 &< a1y < 5.0 \\
-1.0 &< b1x < 7.0 \\
-1.0 &< b1y < 7.0
\end{aligned}
$$

Con estos datos y fijando como límite de error admisible en la norma un valor de 1.0, con el programa OPRIS se obtienen varias soluciones si se aplica la resolución con NR. La mejor de éstas es:

$$
\underset{\sim}{Z}_{NR} = \begin{Bmatrix} 0.60816 \\ 3.51746 \\ 1.90929 \\ 1.16347 \\ 2.65206 \\ 5.55299 \end{Bmatrix}
$$

correspondiente a una norma de 0.150194. Calculando la norma de los vectores de posición de los puntos de las trayectorias obtenida y deseada y comparándalos, resulta que el error relativo máximo es del orden del 20%.

A fin de estudiar el comportamiento de los otros métodos de resolución y de paso mejorar la solución obtenida, partiendo de ésta, se resuelve el sistema de ecuaciones con los métodos de optimización, obteniendo:

	VM	GC	BD	DC
norma	0.031914	0.04526	5.5×10^{-4}	0.1013
iteración	100	100	151	68

Aunque se indica el número de interacciones con que se obtiene este resultado, no es un índice comparativo entre los cuatro algoritmos, por las diferencias existentes en su realización.

La mejor solución es la correspondiente a BD y es:

$$\underset{\sim}{Z}_{BD} = \begin{Bmatrix} 1.01157 \\ 2.98403 \\ 1.85575 \\ 1.52484 \\ 2.69297 \\ 5.43112 \end{Bmatrix}$$

que corresponde a un error relativo máximo, en la norma de los vectores de posición de los puntos del 5%. Con este vector Z_{BD} como primera solución y el programa CIOPT se obtienen los siguientes valores en los ciclos de optimización

N°de Puntos	Norma	Iteraciones
9	5.77×10^{-4}	7
15	5.73×10^{-4}	10
20	7.50×10^{-4}	7
24	8.93×10^{-4}	4
30	1.11×10^{-3}	2
45	1.66×10^{-3}	2

Como puede apreciarse, el ciclo correspondiente a 45 puntos de comparación no impone ya progreso en la solución obtenida. El vector de diseño que define al mecanismo obtenido es

$$\underset{\sim}{Z}_{45} = \begin{Bmatrix} 0.99781 \\ 3.01133 \\ 1.86693 \\ 1.49724 \\ 2.69835 \\ 5.42131 \end{Bmatrix}$$

El error relativo que se comete, si se toma este vector como solución definitiva, es del orden del 0.025%

La trayectoria de la Figura 5-2-2 es generada por un mecanismo conocido. Comparando los verdaderos valores con los obtenidos

	Valor real	Valor obtenido
a0x	1.	1.
a0y	1.	0.99781
b0x	4.4641	4.4641
b0y	3.0	3.01133
a1x	1.86602	1.86693
a1y	1.5	1.49724
b1x	2.68471	2.68835
b1y	5.41532	5.42131

A la vista de lo cual, se puede decir que la solución obtenida, siguiendo la metodología propuesta y aplicando los ciclos de optimización, es del todo aceptable.

Si en vez de realizar los ciclos de optimización con incremento de puntos de comparación se realiza di--

rectamente con el número total de puntos de comparación deseados, que en este caso pueden ser 45, se obtiene como solución

$$Z_{45} = \begin{Bmatrix} 0.99642 \\ 3.0231 \\ 1.85731 \\ 1.50731 \\ 2.69864 \\ 5.42237 \end{Bmatrix}$$

con un error del 2% y tras realizar 47 iteraciones.

En la tabla 5-2-1 los valores intermedios -- del vector de diseño que se van obteniendo en algunos de los ciclos de optimización que se han realizado.

TABLA -5-2-1

Z_9	Z_{20}	Z_{30}
0.99142	0.99785	0.99781
3.01310	3.01333	3.01133
1.85829	1.86693	1.86693
1.50868	1.49728	1.49724
2.69921	2.69835	2.68925
5.42265	5.42130	5.42127

En la figura 5.2.3 se ha representado el mecanismo obtenido en cuatro posicioens distintas. También se adjuntan a continuación la representación gráfica de las -- trayectorias generadas por mecanismos definidos con los vectores de diseño obtenidos en los ciclos intermedios antes - citados.

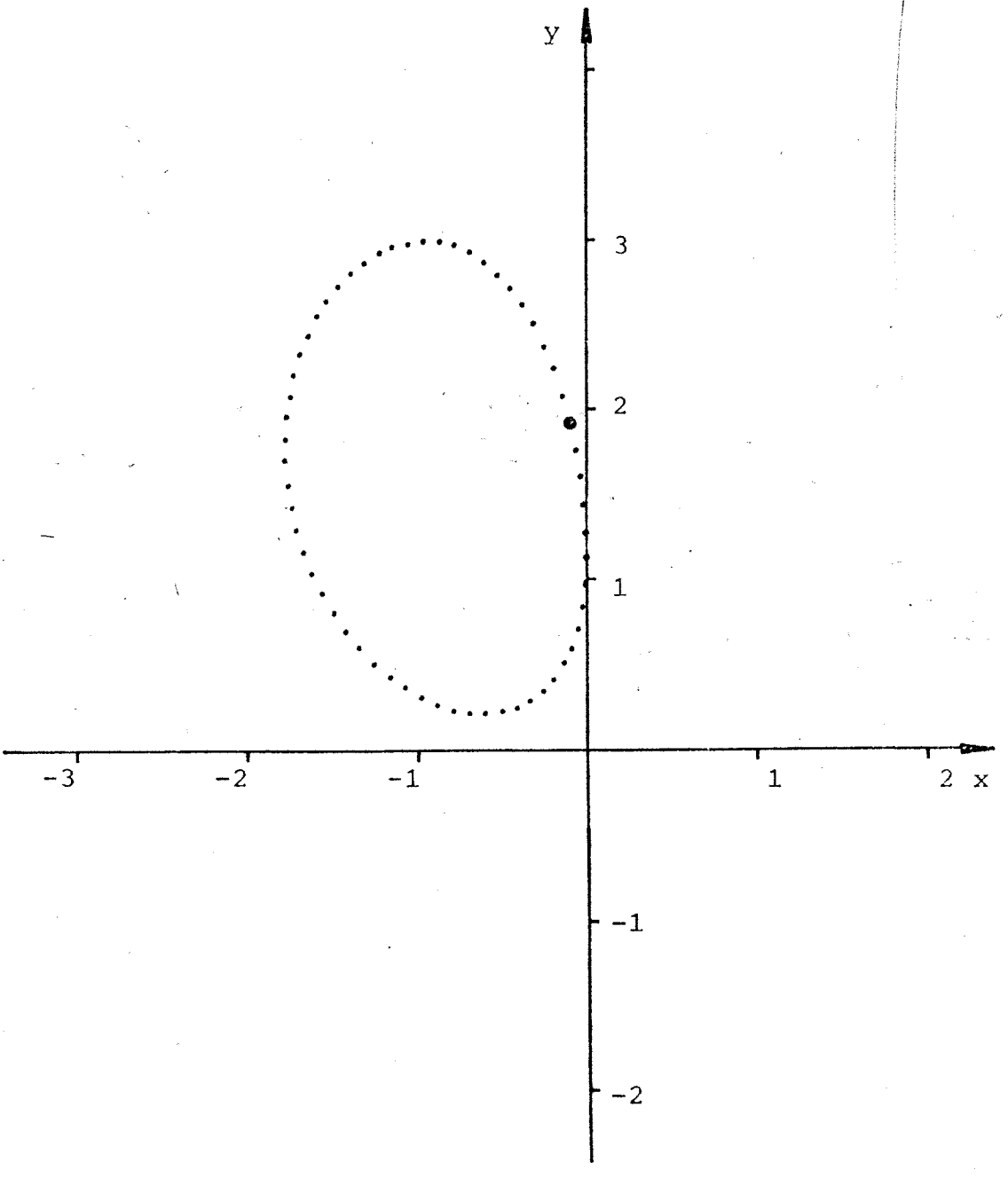


FIGURA 5-2-2

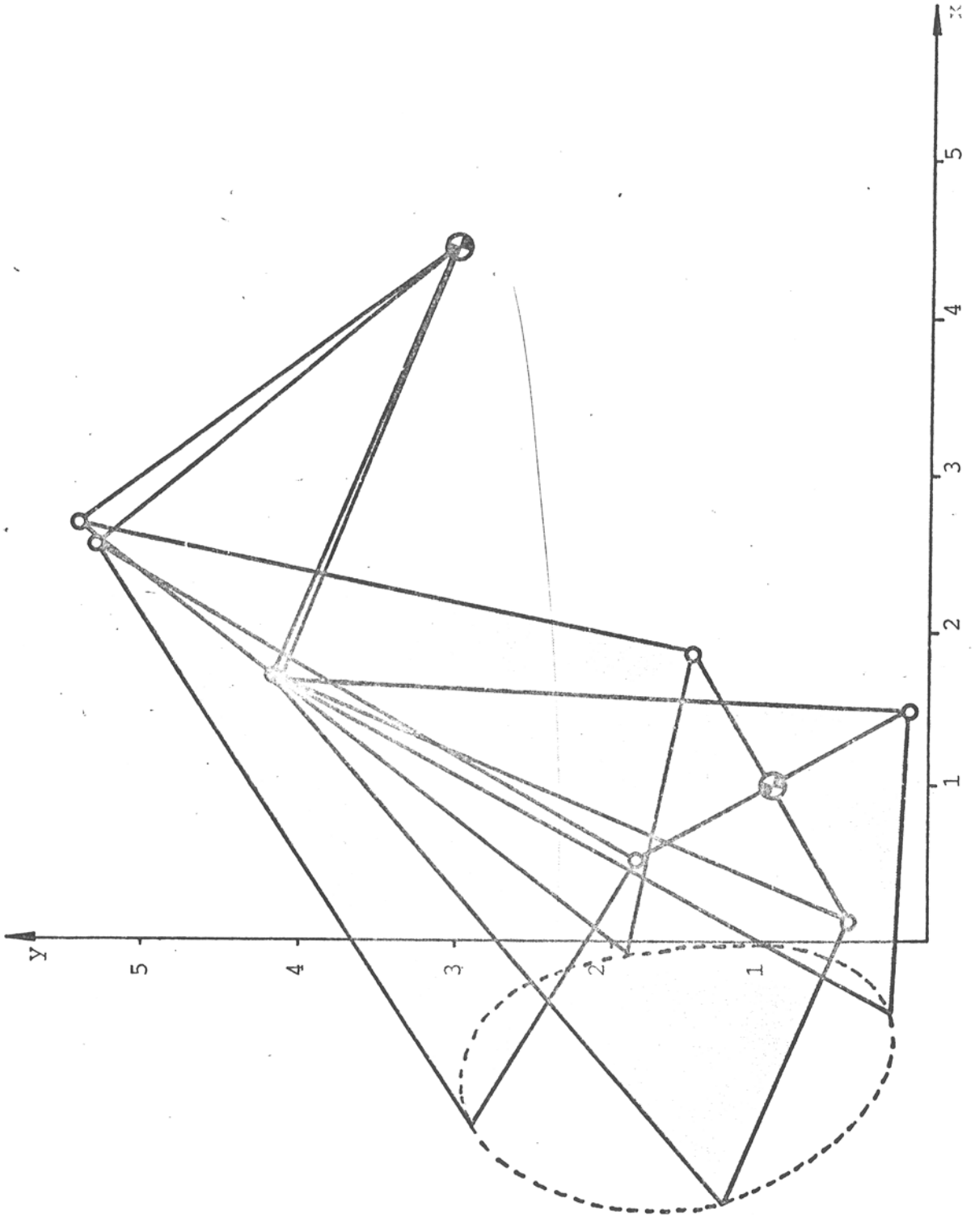


FIGURA 5-2-3

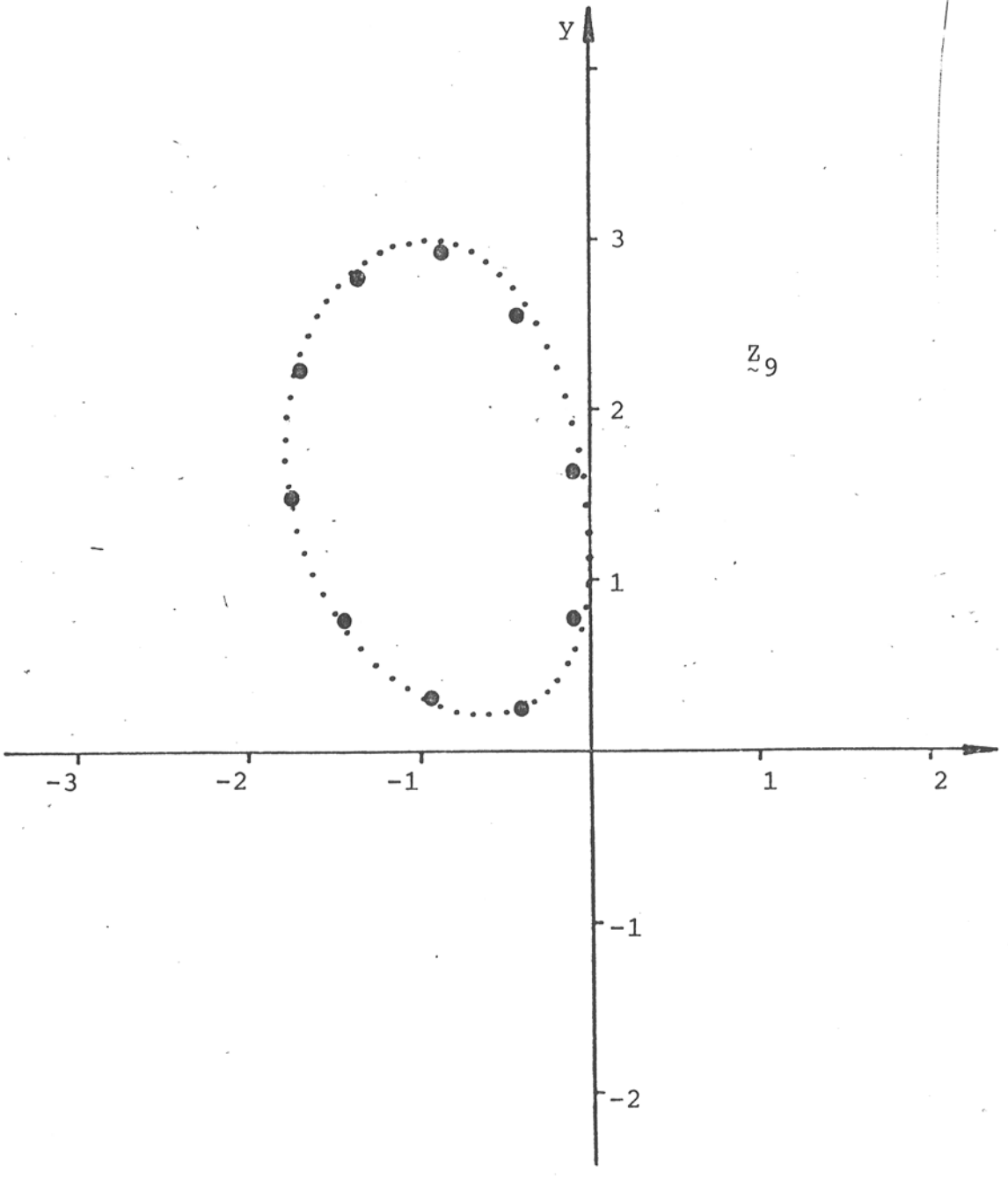


FIGURA 5-2-4

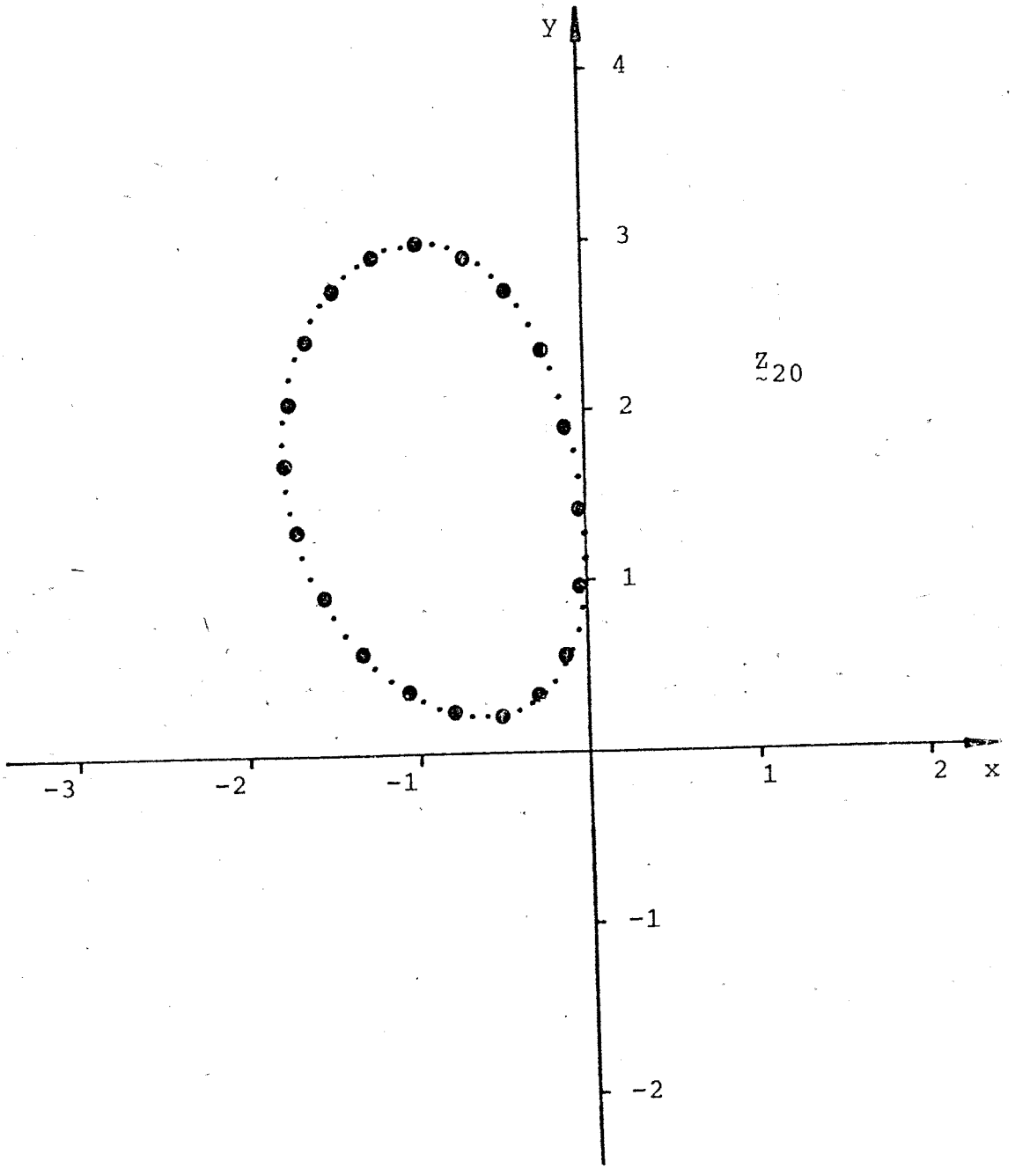


FIGURA 5-2-5

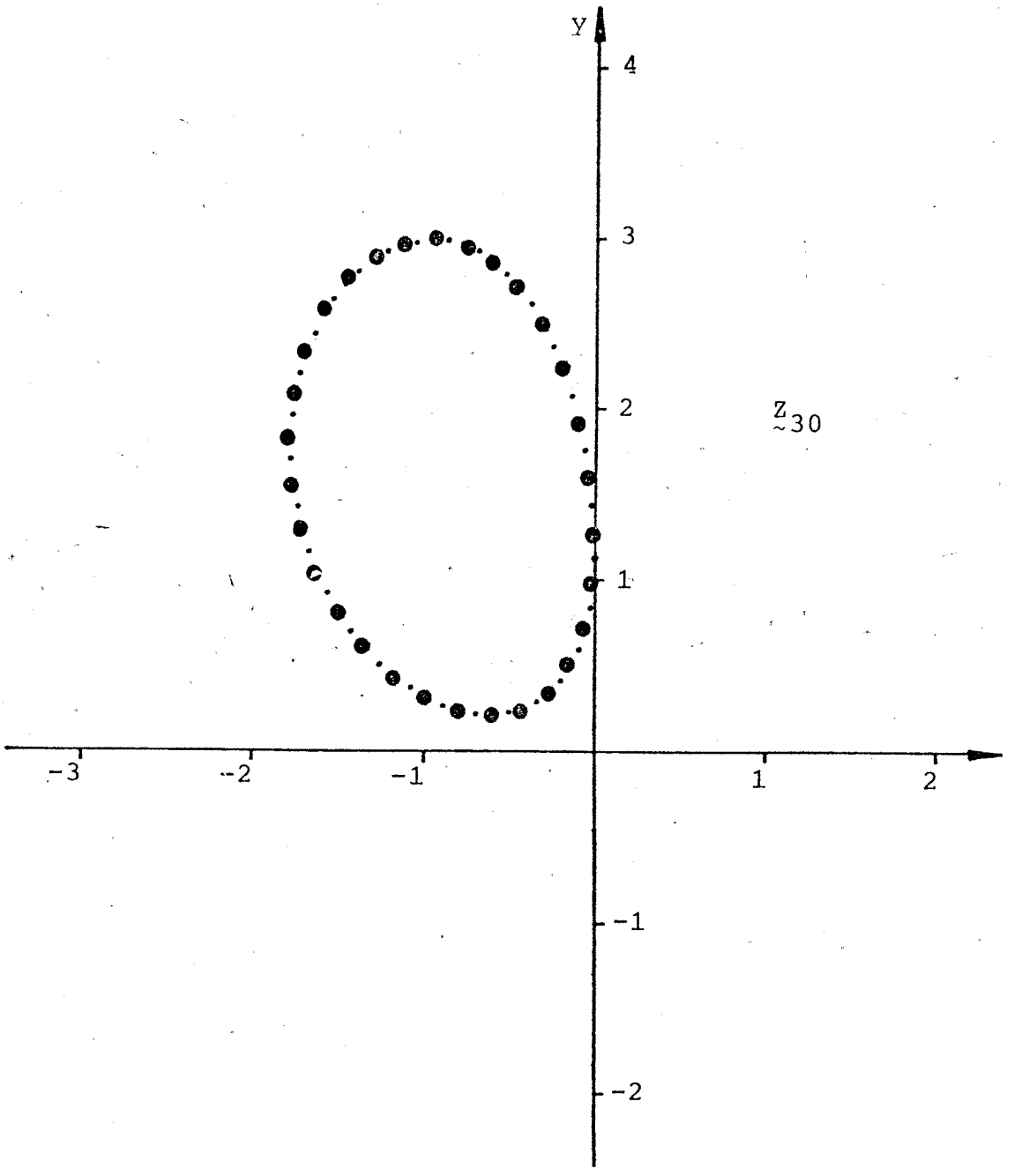


FIGURA 5-2-6

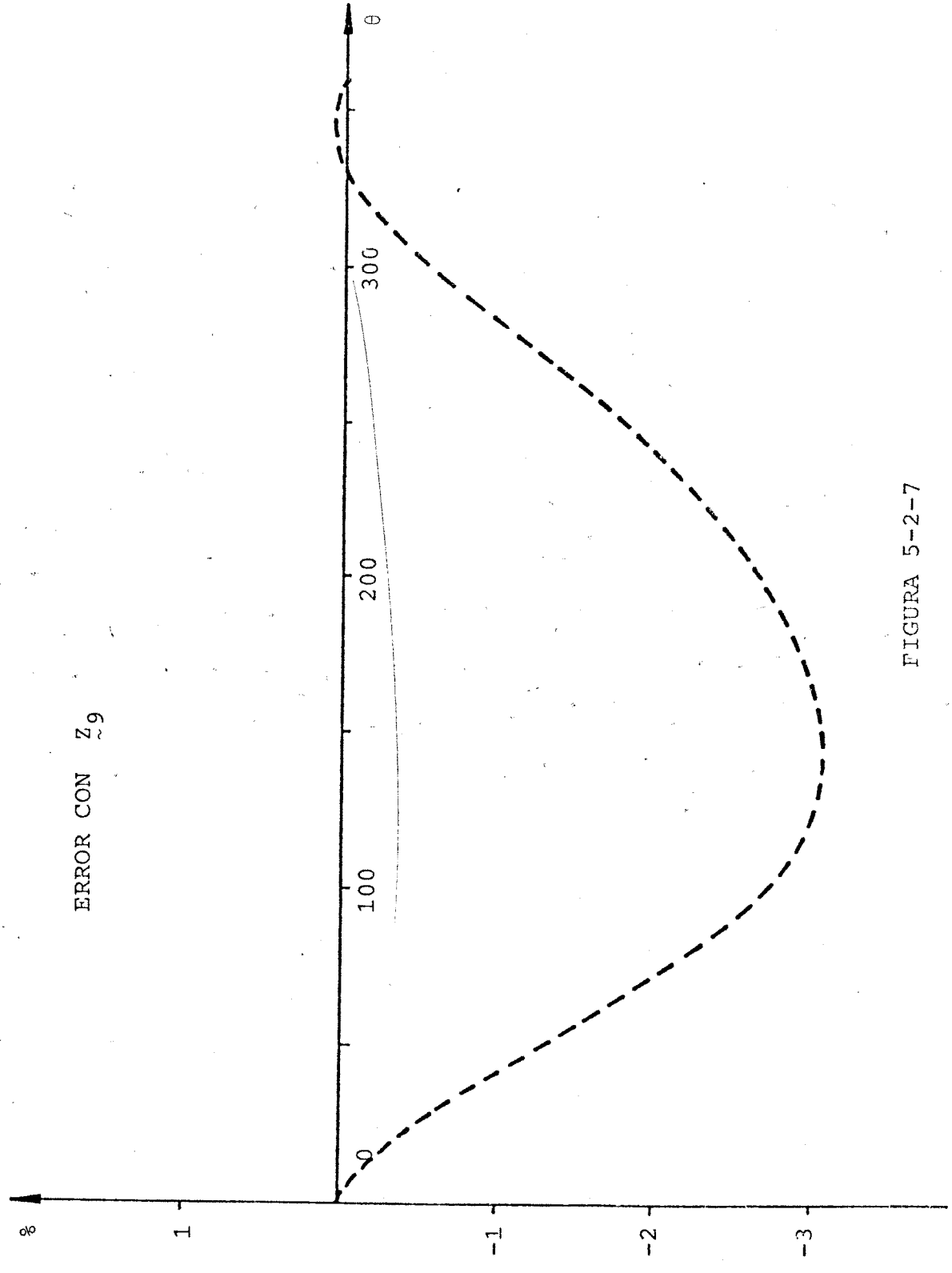


FIGURA 5-2-7

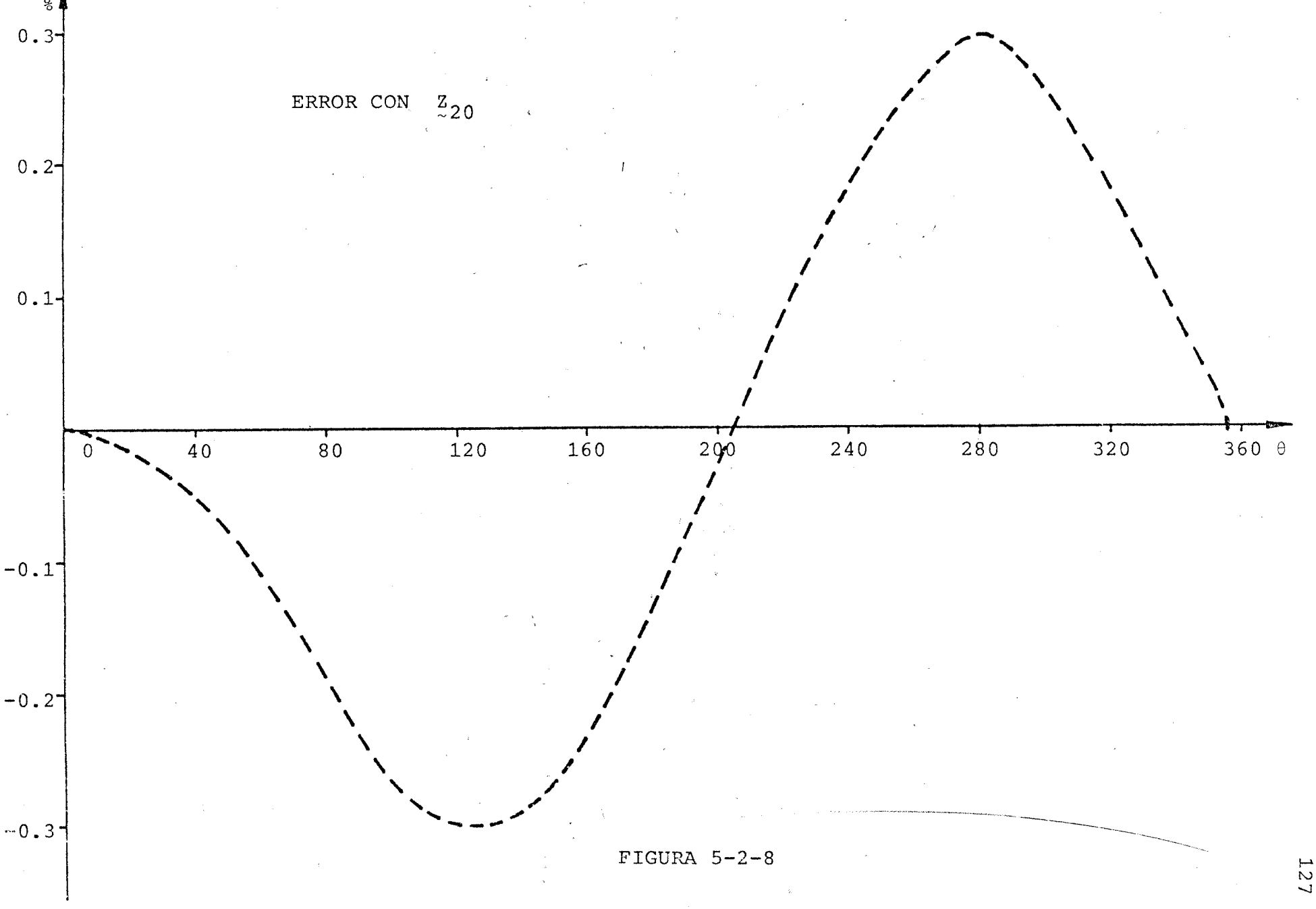

%
ERROR CON Z_{20}
0.3
0.2
0.1
-0.1
-0.2
-0.3
0
40
80
120
160
200
240
280
320
360
θ

FIGURA 5-2-8

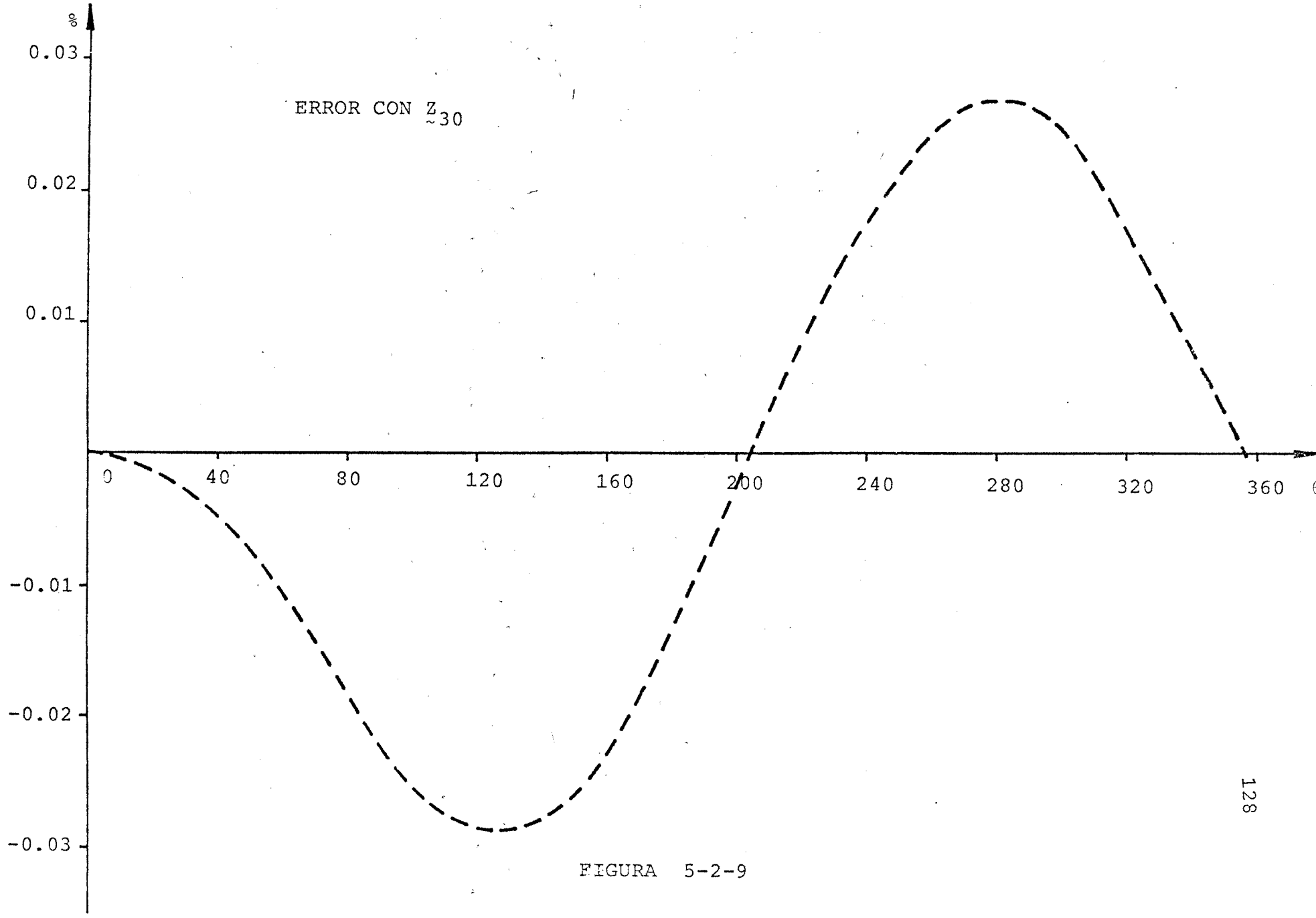

%
0.03
ERROR CON Z_{30}
0.02
0.01
0
40
80
120
160
200
240
280
320
360
-0.01
-0.02
-0.03

FIGURA 5-2-9

5.3. SINTESIS OPTIMA DE UN MECANISMO DE CUATRO BARRAS PLANO CON UNA B.D.V.

Entrando ya en la síntesis óptima de generación de trayectorias con mecanismos de B.D.V., se realiza la síntesis para un cuadrilátero articulado plano, en el que la barra de salida es de dimensión variable, y cuya ley de variación es del tipo B-2.

5.3.1. SISTEMA DE ECUACIONES.

Para el mecanismo de la Figura 5-3-1, se formula un sistema de ecuaciones, idéntico al de la aplicación anterior, pero con la particularidad, de que la expresión (5-6) cambia, para tener en cuenta, que la barra de salida es de dimensión variable. La (5-6) puede expresarse en este caso de la forma:

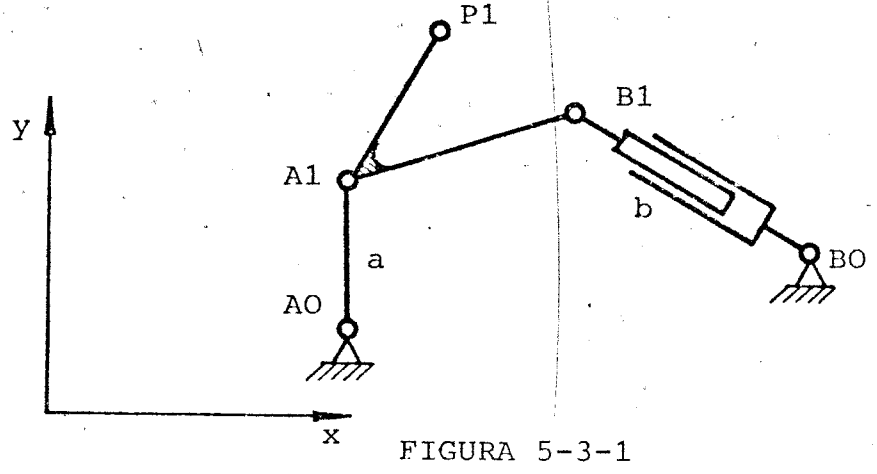


FIGURA 5-3-1

$$f_{j2}(\underset{\sim}{Z}) = \left[(\underset{\sim}{BJ} - \underset{\sim}{B0})^T (\underset{\sim}{BJ} - \underset{\sim}{B0}) - (l_j)^2 \right]^2 \qquad (5\text{-}12)$$

siendo l_j la longitud de la barra de salida en el instante j. Como se toma como ley de variación la B-2, resultará que

$$\begin{aligned} l_j &= l_0 + v_0\, t_j \quad ; \qquad t_j \leq t_o \\ l_j &= l_o + v_o . t_o - v_1 (t_j - t_o) \; ; \; t_o \leq t_j \leq t_1 \end{aligned} \qquad (5\text{-}13)$$

donde se cumple que

$$v_o . t_o = v_1 . (t_1 - t_o) \qquad (5\text{-}14)$$

luego

$$v_1 = \frac{v_o\, t_o}{(t_1 - t_o)} \qquad (5\text{-}15)$$

siendo t_1 el tiempo total de evolución del ciclo de generación de la trayectoria.

El sistema de ecuaciones es, por tanto, el indicado en (5-3) con el cambio mencionado en $f_{j2}(Z)$ por la representada en (5-12). La ley de variación escogida añade dos variables más al vector de diseño (t_o, v_o). Además hay que añadir una variable más, que es la velocidad angular de la barra de entrada, a fín de tener como variable fundamental el tiempo. En todas las aplicaciones se considera esta velocidad angular como constante en el tiem

po de generación de la trayectoria. El sistema será como - máximo de once ecuaciones con once incognitas.

$$f_i(\underset{\sim}{Z}) = 0 \quad ; \quad i = 1, 2, \ldots\ldots 11 \qquad (5\text{-}16)$$

donde

$$\underset{\sim}{Z} = \left\{ aox, aoy, box, boy, a1y, b1x, b1y, tb0, vb0, w \right\}^T \qquad (5\text{-}17)$$

Se puede hacer una síntesis de generación - de trayectorias especificando doce puntos de precisión, once más el inicial, y el instante de tiempo correspondiente a cada uno de ellos.

5.3.2. FUNCION OBJETIVO.

La función objetivo en este caso se formará de manera identica a como se hizo en el caso anterior, con - la variación ya apuntada en f_{j2} (5-12) y quedará de la forma:

$$F(\underset{\sim}{Z}) = \sum_{j=2}^{j=12} \left[(\underset{\sim}{PJ}-\underset{\sim}{AJ})^T(\underset{\sim}{PJ}-\underset{\sim}{AJ}) - (\underset{\sim}{P1}-\underset{\sim}{A1})^T(\underset{\sim}{P1}-\underset{\sim}{A1}) \right]^2 +$$
$$+ \sum_{j=2}^{j=12} \left[(\underset{\sim}{BJ}-\underset{\sim}{B0})^T(\underset{\sim}{BJ}-\underset{\sim}{B0}) - (l_j)^2 \right]^2 \qquad (5\text{-}18)$$

donde la l_j esta indicada en (5-13). Para los ciclos de - optimización el número de puntos j será el que se especifique por ciclo.

De las expresiones arriba indicadas se realiza su programación en lenguaje FORTRAN, constituyendo la función FUNC, en la que ya se hace una llamada a la subrutina LONG, para calcular la variación l_i de la dimensión de la barra de salida. Incorporando FUNC a los programas OPRIS y CIOPT se esta en condiciones de realizar la síntesis.

5.3.3. APLICACION NUMERICA.

Con el planteamiento ya indicado se pretende obtener el vector de diseño que define al mecanismo de cuatro barras plano, con la barra de salida como B.D.V y ley B-2, que genere la trayectoria de la Figura 5-3-2, con el mínimo error posible. La trayectoria es conocida en forma de 360 puntos, correspondientes cada uno de ellos, a un instante de tiempo determinado. Además

$$p1x = 0.1525$$
$$p1y = 2.5314$$
$$a0x = 1.0$$
$$b0x = 4.4641$$
$$w = 1.74532 \quad \text{rad/seg.}$$

el intervalo de tiempo entre puntos es de 0.01 seg. El vector de diseño quedará en la forma:

$$\underset{\sim}{Z} = \left\{a0y, b0y, a1x, a1y, b1x, b1y, tb0, vb0\right\}^T$$

Se realiza la síntesis con puntos de precisión tomando como tales los pertenecientes a la trayectoria para los ins-

tantes de tiempo 0.4, 0.8, 1.2, 1.6, 2.0, 2.4, 2.8, y 3.2 seg. y que respectivamente son:

(-0.48659, 2.74973)
(-1.28236, 2.42452)
(-1.86602, 1.50)
(-1.97798, 0,99143)
(-1.63721, 0,38314)
(-0.99925, 0.05477)
(-0.35674, 0.27716)
(0.0163 , 1.17991)

Para la generación aleatoria de la familia de vectores de diseño inicial, se toman como límites:

$$0.0 < a0x < 4.0$$
$$0.0 < b0y < 4.0$$
$$-3.0 < a1x < 5.0$$
$$-3.0 < a1y < 5.0$$
$$-1.0 < b1x < 7.0$$
$$-1.0 < b1y < 7.0$$
$$0.0 < tb0 \leq 3.6$$
$$0.0 < vb0 \leq 5.0$$

La solución obtenida con el programa OPRIS, que será la solución de partida para la síntesis óptima, es

$$\underset{\sim}{Z}_{IN} \begin{Bmatrix} 1.11249 \\ 3.24021 \\ 1.89059 \\ 1.5881243 \\ 3.27584 \\ 6.41146 \\ 1.20886 \\ 2.67861 \end{Bmatrix}$$

Si se toma como solución definitiva, el error relativo máximo, para la norma de los vectores de posición de los puntos de las trayectorias generada y deseada, que se comete en la generación de la trayectoria completa, es del orden del 23% en alguno de esos 360 puntos.

Realizando la síntesis óptima con el programa CIOPT y tomando esta $\underset{\sim}{Z}_{IN}$ como solución de partida se obtiene, entre otras, las soluciones indicadas en la tabla 5-3-1 correspondiente a 20, 30 y 45 puntos de comparación. Los errores que se cometen si se toma cada una de estas como solución final son respectivamente, 3%, 0.5% y 0.05 %.

TABLA 5-3-1

$\underset{\sim}{Z}_{20}$	$\underset{\sim}{Z}_{30}$	$\underset{\sim}{Z}_{45}$
0.97554	0.99084	1.00062
3.02278	2.98121	3.00182
1.86444	1.86437	1.86619
1.48355	1.49269	1.50058
4.23365	3.97258	3.92563
4.43365	4.81702	4.93401
1.20193	1.20027	1.19981
2.4569	2.48549	2.50120

A la vista de los resultados obtenidos se acepta como solución, la correspondiente a $\underset{\sim}{Z}_{45}$, con lo - que se puede decir que el mecanismo que genera la trayectoria deseada será un mecanismo de 4 barras plano definido por

aox = 1.0
aoy = 1.00062
box = 4.4641
boy = 3.00182
a1x = 1.86619
a1y = 1.50058
b1x = 3.92563
b1y = 4.93401
p1x = 0.1525
p1y = 2.5314

representado en la Figura 5-3-3 y en el que la barra de salida es de dimensión variable con una ley de variación - B-2, definida por los valores

$$tb0 = 1.19981 \quad \text{seg.}$$
$$vb0 = 2.5012 \quad \text{un/seg.}$$

Figura 5-3-4. La velocidad angular de la barra de entrada será constante e igual a

$$w = 1.74532 \quad \text{rad/seg.}$$

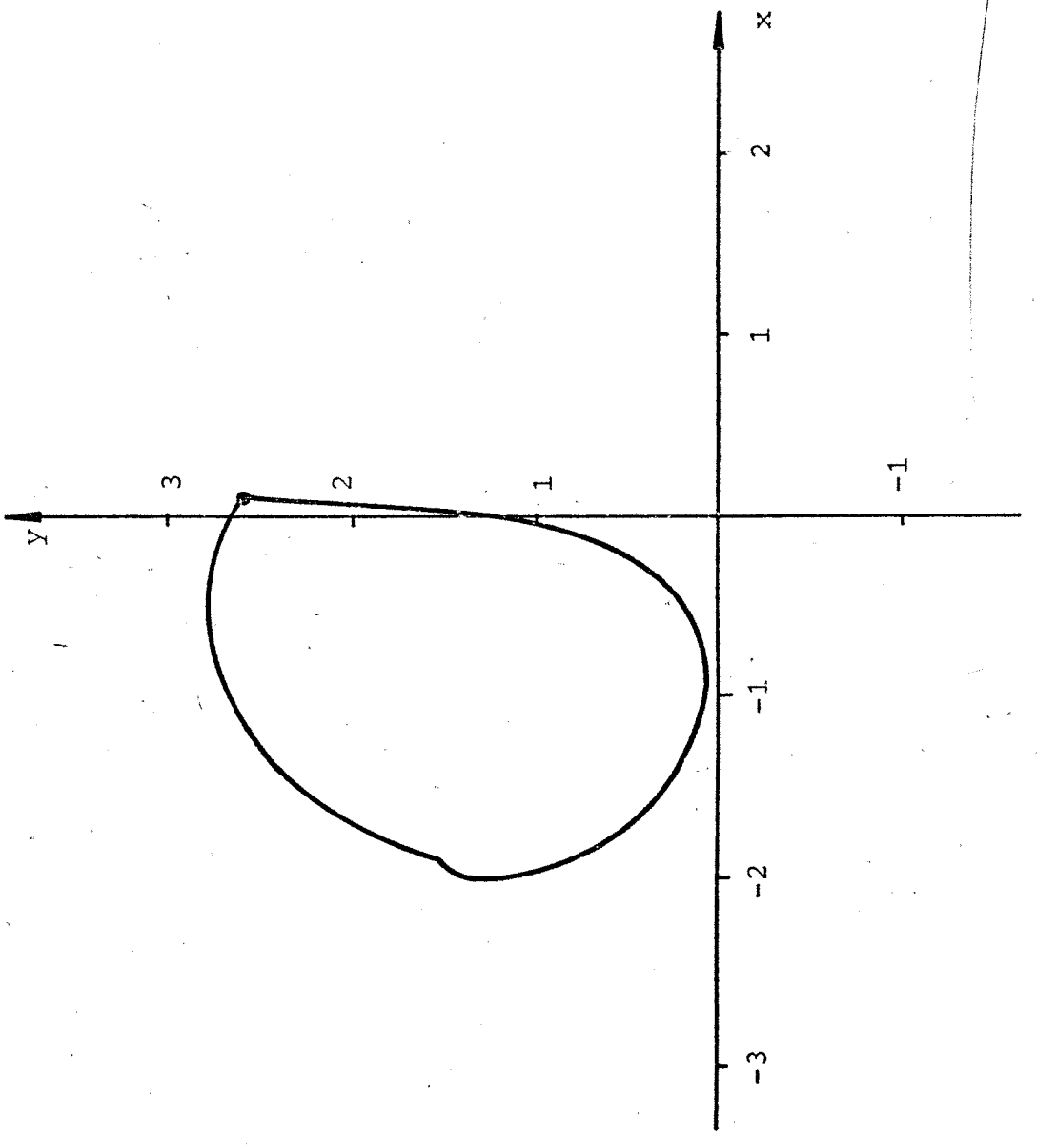


FIGURA 5-3-2

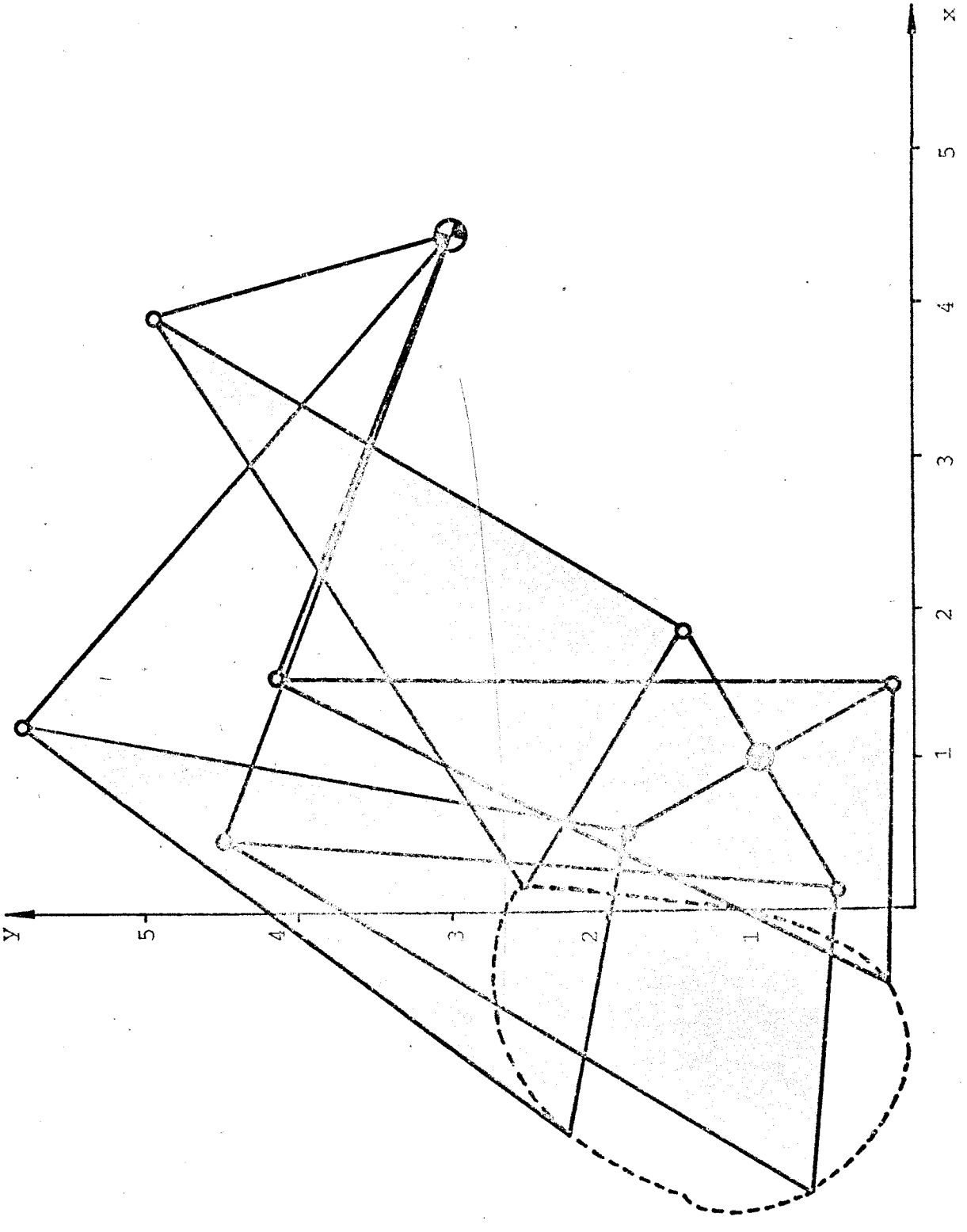


FIGURA 5-3-3

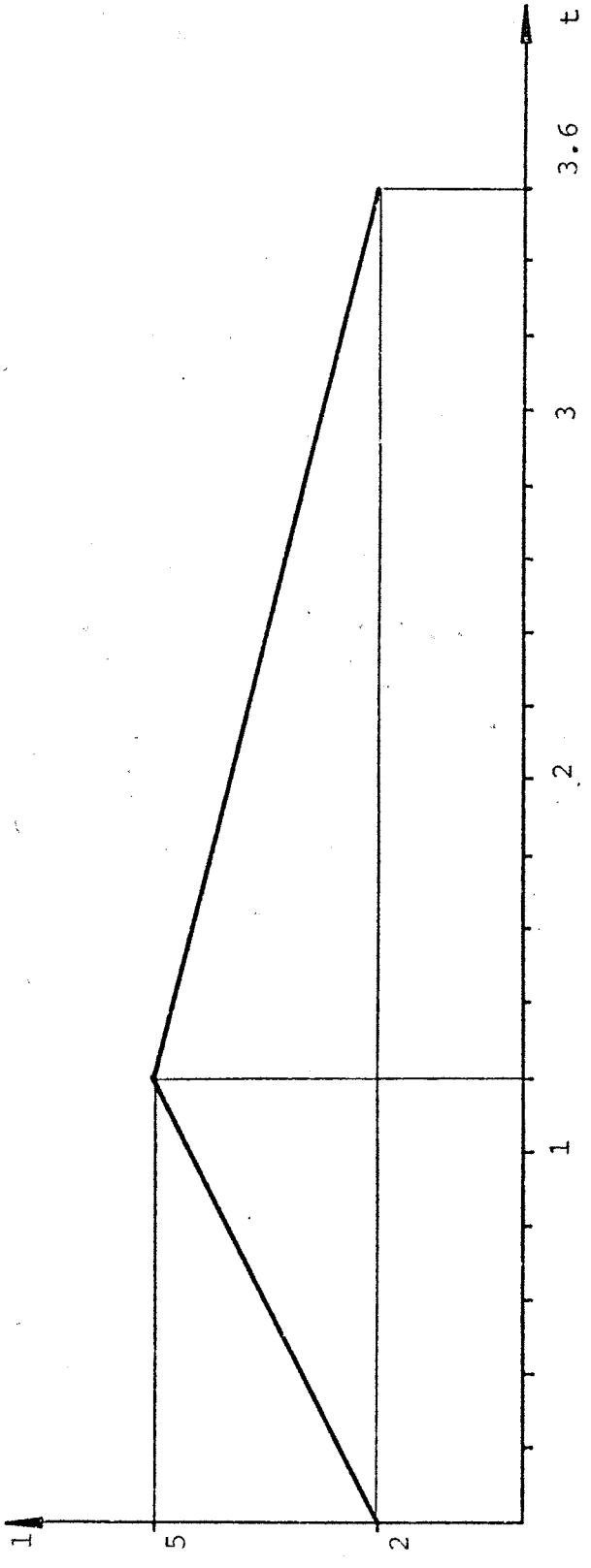


FIGURA 5-3-4

5.4. SINTESIS OPTIMA DE UN MECANISMO DE CUATRO BARRAS PLANO CON DOS B.D.V.

El mecanismo de cuatro barras plano tendrá ahora dos barras de longitud variable, que serán la de salida y el acoplador, con un mismo tipo de ley de variación de la dimensión como es la B-2.

5.4.1. SISTEMA DE ECUACIONES.

Para el sistema de la Figura 5-4-1, se formula

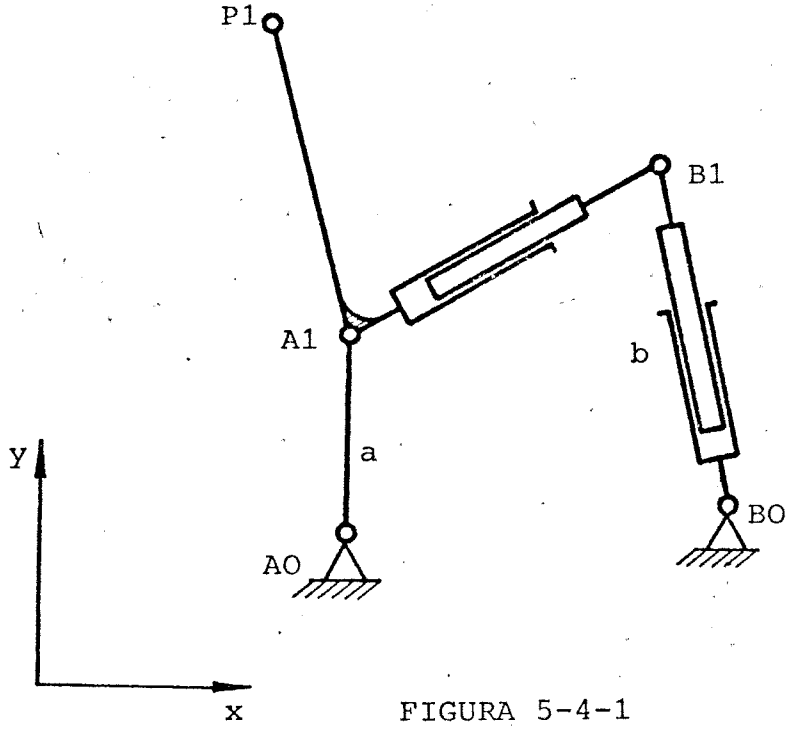


FIGURA 5-4-1

un sistema de ecuaciones muy similar al de los casos anteriores, sólo que ahora hay que tener en cuenta que el acoplador también es de dimensión variable, lo que supone que

la expresión (5-7) se transforme en:

$$\underset{\sim}{BJ} = \left[R(h_j, \phi 1j)\right] (\underset{\sim}{B1} - \underset{\sim}{A1}) + \underset{\sim}{AJ} \qquad (5\text{-}19)$$

donde $\left[R(h_j, \phi 1j)\right]$ es la matriz de rotación-extensión que representa el giro y la variación de longitud del acopla-- dor. El término h_j se obtiene a partir de la ley de variación, recordando que

$$h_j = \frac{l_j}{l_o} \qquad (5\text{-}20)$$

para la barra en cuestión. Como la ley es del tipo B-2 -- utilizando la expresión (5-13), se puede obtener este va-- lor para cada instante de tiempo j.

El sistema de ecuaciones será, por tanto - el mismo que el del caso anterior, pero teniendo en cuenta la expresión (5-19) para el cálculo de los sucesivos BJ.- Como es de trece ecuaciones con trece incognitas

$$f_i(\underset{\sim}{Z}) = 0 \qquad i = 1, 2, \ldots 13 \qquad (5\text{-}21)$$

donde

$$\underset{\sim}{Z} = \{a0x, a0y, b0x, b0y, a1x, a1y, b1x, b1y, tb0, vb0, tc0, vc0, w\}^T \qquad (5\text{-}22)$$

especificando como máximo catorce puntos de la trayectoria se puede realizar la síntesis con puntos de precisión.

5.4.2. FUNCION OBJETIVO.

Es la misma que la expresada en (5-18) - con la variante en el límite superior de j que es ahora 14 y que para calcular BJ hay que realizarlo de acuerdo a (5-19). Para la síntesis óptima el número de puntos j será el que se indique en cada ciclo.

Se realiza la programación de este sistema de ecuaciones y de la función objetivo, en la función FUNC que incorporada a los programas OPRIS y CIOPT - permite la resolución de la síntesis óptima. En esta ocasión la subrutina LONG será llamada dos veces en cada - evaluación del sistema de ecuaciones, ó de la función objetivo, a fín de obtener los valores de las longitudes de las dos B.D.V.

5.4.3. APLICACION NUMERICA.

Se pretende realizar la síntesis del mecanismo de cuatro barras plano, que con dos B.D.V. con ley tipo B-2, genere la trayectoria de la Figura 5-4-2, con el mínimo error posible. La trayectoria es conocida en - la misma forma que en el caso anterior y además

$$p1x = 1.04253$$
$$p1y = 3.32259$$
$$a0x = 1.0$$
$$b0x = 4.4641$$
$$w = 1.74532 \quad \text{rad/seg.}$$

con lo que el vector de diseño será:

$$\underset{\sim}{Z} = \left[a0y, b0y, a1x, a1y, b1x, b1y, tb0, vb0, tc0, vco\right]^T$$

Tomando como límites para la generación aleatoria de la familia de vectores de diseño los siguientes:

$$0.0 < a0x < 4.0$$
$$0.0 < b0x < 4.0$$
$$-3.0 < a1x < 5.0$$
$$-3.0 < a1y < 5.0$$
$$-1.0 < b1x < -7.0$$
$$-1.0 < b1y < -7.0$$
$$0.0 < tb0 < 3.6$$
$$0.0 < vb0 < 5.0$$
$$0.0 < tc0 < 3.6$$
$$0.0 < vc0 < 5.0$$

Realizando la síntesis con puntos de precisión el programa OPRIS obtiene como solución:

$$\underset{\sim}{Z}_{IN} = \begin{Bmatrix} 1.25283 \\ 3.15192 \\ 1.87638 \\ 1.73239 \\ 1.88005 \\ 5.83203 \\ 1.20299 \\ 2.54048 \\ 2.37012 \\ 0.46772 \end{Bmatrix}$$

que tomandola como solución inicial en el programa CIOPT se realiza la síntesis óptima, obteniendo los resultados - indicados en la tabla 5-4-1 correspondientes a 20, 36 y 60 puntos de comparación entre las trayectorias. Los - -- errores máximos que se cometen tomando cada una de estas - como solución final, son respectivamente del 4%, 1% y 0.3%.

TABLA 5-4-1

$\underset{\sim}{Z}_{20}$	$\underset{\sim}{Z}_{36}$	$\underset{\sim}{Z}_{60}$
0.77933	0.98154	0.98247
3.01458	2.93179	2.96311
1.86403	1.86418	1.86544
1.29716	1.48285	1.48362
1.79334	2.33233	2.33857
5.14149	4.75092	4.75153
1.19934	1.19917	1.19982
2.54157	2.5144	2.51302
2.41160	2.38946	2.39061
0.70972	0.71983	0.71016

De acuerdo con estos resultados se acepta - como solución la correspondiente a Z_{60}, y por tanto el mecanismo que genera la trayectoria deseada con un error mínimo, es el cuatro barras plano definido por

a0x = 1.0
a0y = 0.98247
b0x = 4.4641
b0y = 2.96311
a1x = 1.86544
a1y = 1.48362
b1x = 2.33857
b1y = 4.75153
p1x = 1.04253
p1y = 3.32259

representado en la Figura 5-4-3 y en el que, la barra de salida es B.D.V. con ley B-2 definida por los valores

t_o = 1.19982 seg.
v_o = 2.5130 un/seg.

que se representa en la figura 5-4-4, y el acoplador es también B.D.V. con ley B-2 definida por

t_o = 2.39061 seg.
v_o = 0.71016 un/seg.

que se representa en la figura 5-4-5. La velocidad angular de la barra de entrada será

$$w = 1.74532 \quad \text{rad/seg.}$$

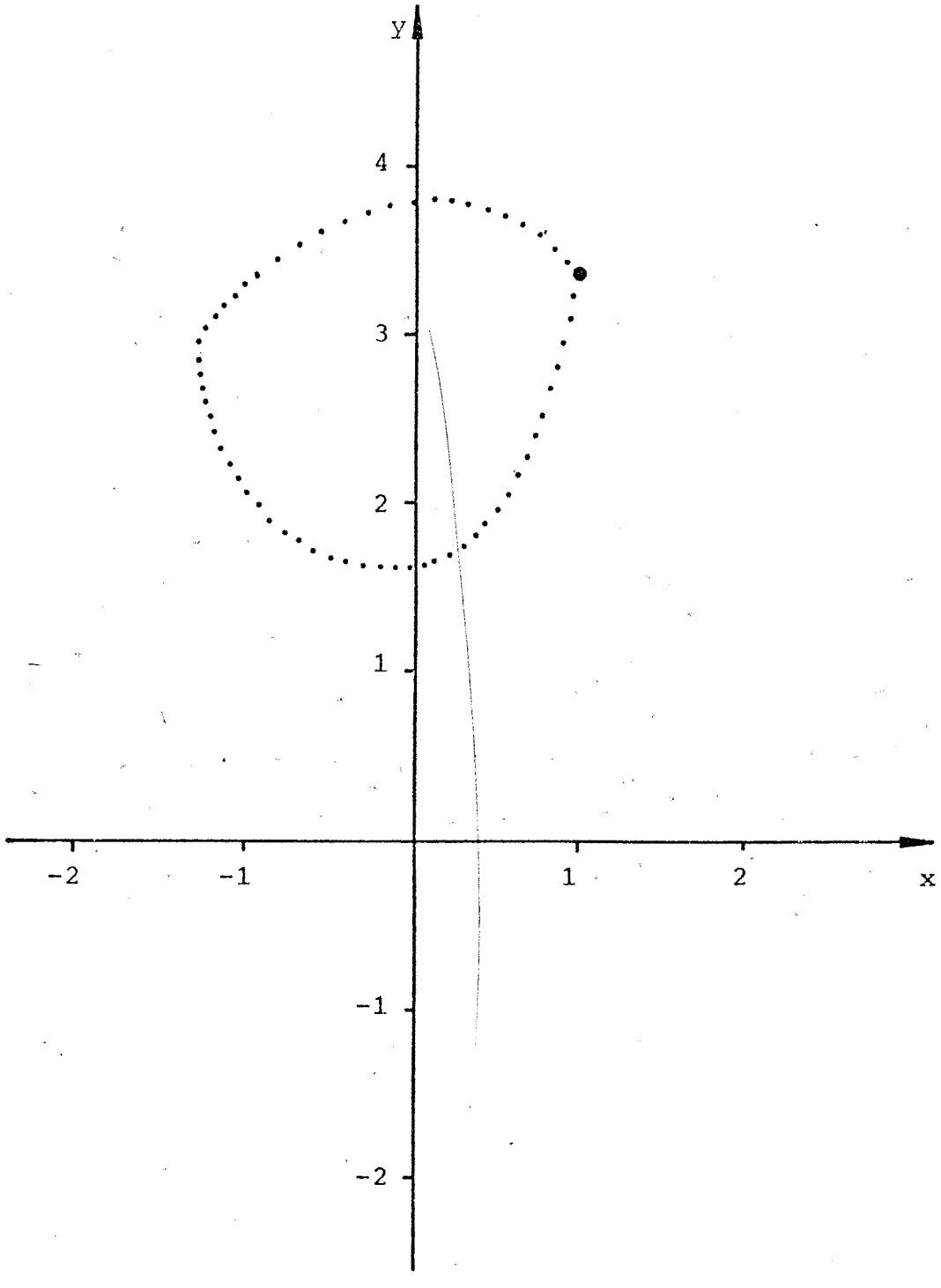


FIGURA 5-4-2

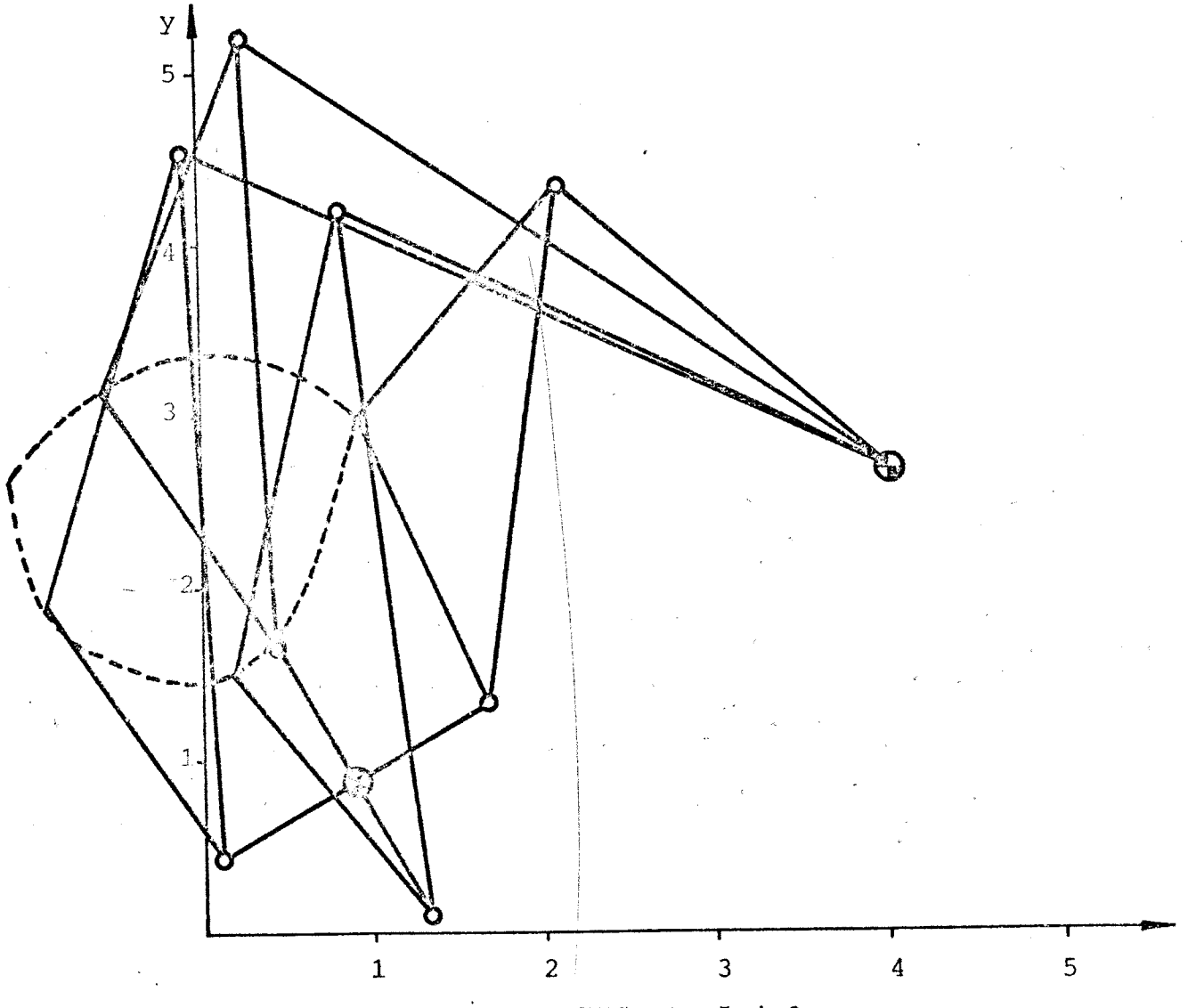


FIGURA 5-4-3

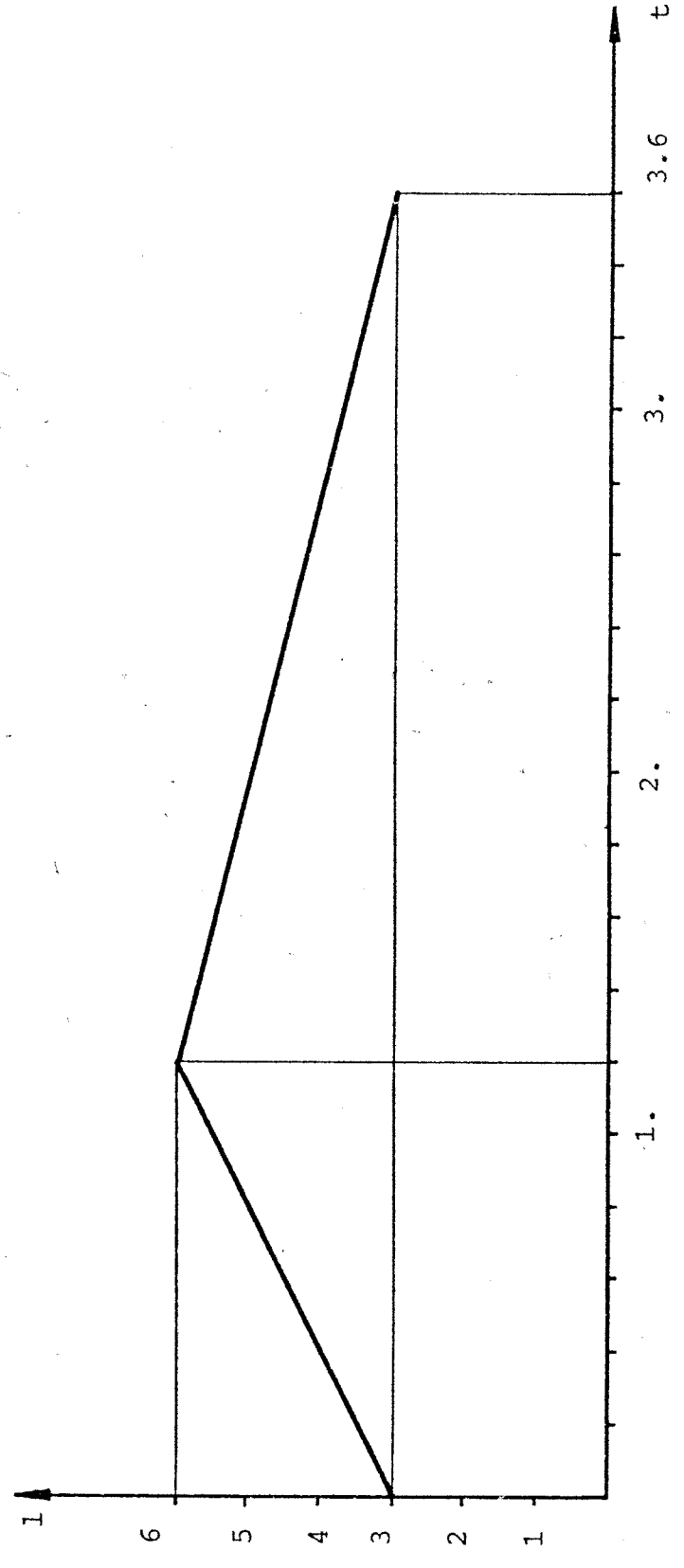


FIGURA 5-4-4

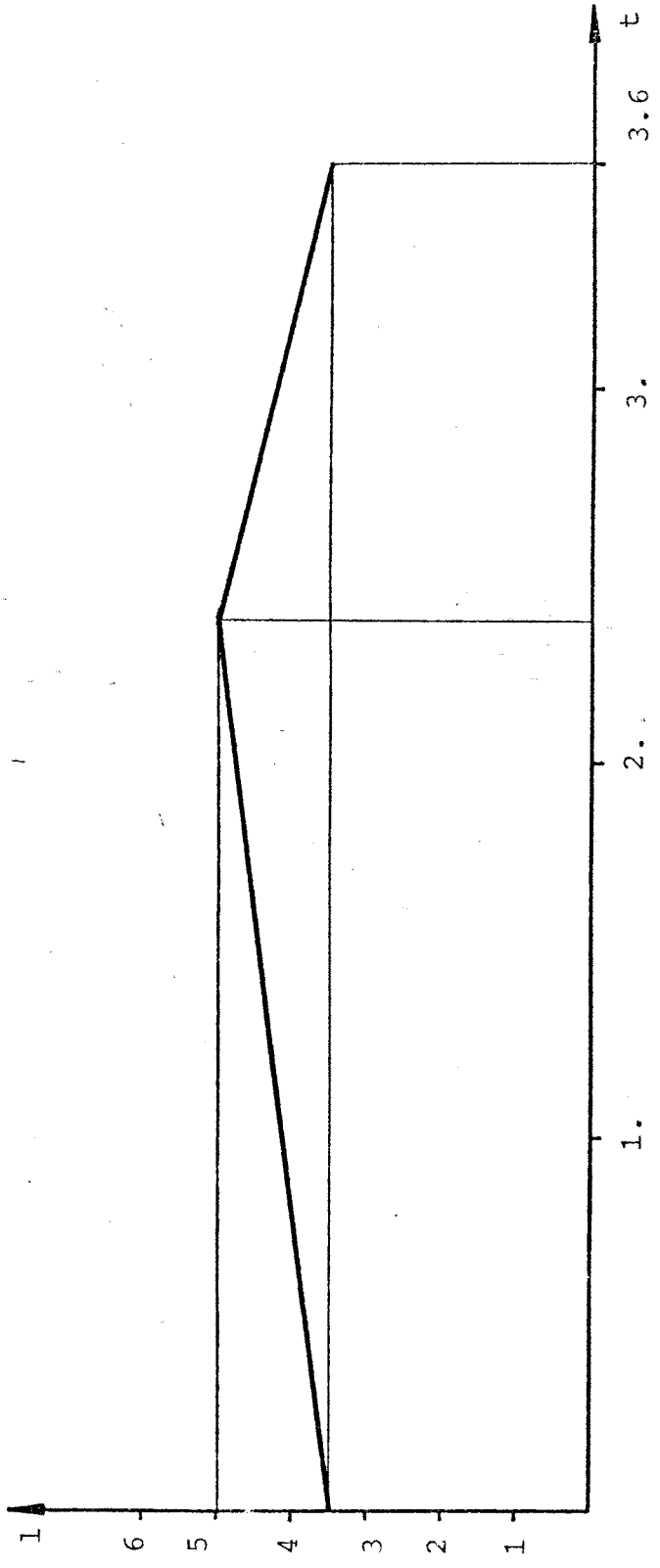


FIGURA 5-4-5

5.5. SINTESIS OPTIMA DE UNA CADENA CINEMATICA PLANA ABIERTA. DOS B.D.V. LEY C.

Cambiando de tipo de mecanismo, se realizará la síntesis óptima para la generación de una trayectoria con una cadena cinemática plana abierta muy simple, con B.D.V. y ley de variación tipo C.

5.5.1. SISTEMA DE ECUACIONES.

Se toma como mecanismo tipo la cadena cinemática plana abierta representada en la Figura 5-5-1,

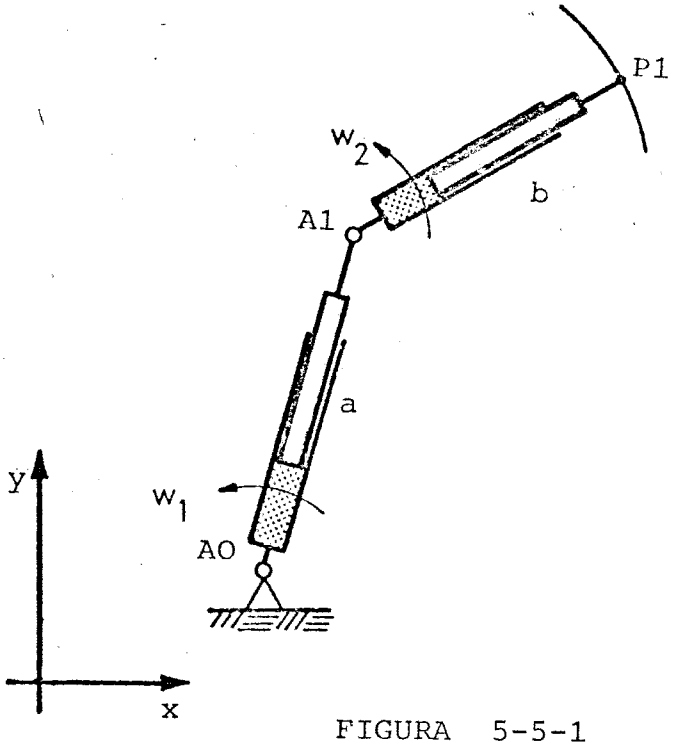


FIGURA 5-5-1

que está formada por dos barras, que son de dimensión va-- riable, unidas mediante un par de rotación, y cogida al -- bastidor o barra fija por otro par de rotación. La rota-- ción alrededor de esos pares de rotación se hace con velo- cidad angular constante. El sistema de ecuaciones será -- ahora

$$f_j(\underset{\sim}{Z}) = 0 \qquad j = 1, 2, 3, \ldots n \qquad (5\text{-}23)$$

para

$$f_j = (\underset{\sim}{QJ} - \underset{\sim}{PJ})^T(\underset{\sim}{QJ} - \underset{\sim}{PJ}) \qquad (5\text{-}24)$$

siendo $\underset{\sim}{QJ}$ el vector de posición de los puntos de la tra- yectoria deseada. $\underset{\sim}{PJ}$ lo será, pero en la trayectoria ob- tenida. n es el número de incognitas a determinar. El - vector de diseño $\underset{\sim}{Z}$ será en este caso:

$$\underset{\sim}{Z} = \left[a0x, a0y, a1x, a1y, w_1, w_2, ta1, va1 \ldots\right]^T \qquad (5\text{-}25)$$

La dimensión de este vector de diseño dependerá de las le- yes de variación en las B.D.V.. Aquí se impone que la ley en ambas es de tipo C, y para este tipo de ley hay que fi- jar el número de tramos que tienen. Para el cálculo de $\underset{\sim}{PJ}$ y $\underset{\sim}{AJ}$ hay que tener en cuenta que

$$\underset{\sim}{PJ} = \left[RL(h_j, \theta 1j)\right] (\underset{\sim}{P1} - \underset{\sim}{A1}) + \underset{\sim}{AJ} \qquad (5\text{-}26)$$

$$\underset{\sim}{AJ} = \left[RL(h_j, \alpha 1j)\right] (\underset{\sim}{A1} - \underset{\sim}{AO}) + \underset{\sim}{AO} \qquad (5\text{-}27)$$

donde

$$\alpha 1j = w_1 \, t_j \qquad (5\text{-}28)$$

$$\theta 1j = (w_1 + w_2) tj \qquad (5\text{-}29)$$

y h_j y h'_j se determinan de acuerdo con la ley C para cada barra.

Una vez fijado el número de tramos de la ley C de variación de cada barra, ya se conoce el número de incognitas a determinar y por lo tanto se conoce el número de ecuaciones. Se puede resolver una síntesis con puntos de precisión a partir del sistema que se plantea.

5.5.2. FUNCION OBJETIVO.

La función objetivo es en este caso

$$F(\underset{\sim}{Z}) = \sum_{j=1}^{j=n} (\underset{\sim}{QJ} - \underset{\sim}{PJ})^T (\underset{\sim}{QJ} - \underset{\sim}{PJ}) \qquad (5\text{-}30)$$

y que para el caso de síntesis óptima el valor de n en vez de ser el del número de incognitas o dimensión del vector del diseño, será el número de puntos de comparación entre trayectorias. Tanto el sistema de ecuaciones como la función objetivo se programan en la función FUNC y se añade a los programas OPRIS y CIOPT. La subrutina LONG, en este caso, será llamada dos veces, por la función FUNC, en cada evaluación del sistema de ecuaciones, función objetivo, para determinar las correspondientes longitudes de las B.D.V. con ley C de variación.

5.5.3. APLICACION NUMERICA.

Se desea obtener el vector de diseño que define un mecanismo del tipo aquí descrito de forma que genere una trayectoria lo más aproximada posible a la representada en la Figura 5-5-2. Tras algunos tanteos se fija, en 2 y 3, el número de tramos para la ley C de variación de la dimensión de las barras a y b. Se sabe que

$$p1x = -3.51104$$

$$p1y = 3.6103$$

y que el vector de diseño tendrá en esta ocasión la expresión:

$$\underset{\sim}{Z} = [a0x, a0y, a1x, a1y, w_1, w_2, t0a, v0a, t1a, v1a, tob, vob, t1b, v1b, t2b, v2b]^T$$

con lo que el número de incognitas o variables a obtener es de 16. Tomando ese número de puntos de precisión se realiza la síntesis con el programa OPRIS. Con el programa CIOPT, para la síntesis óptima, se obtiene como solución aceptable, el sistema de la Figura 5-5-1 definido por

$$a0x = -6.4730997$$
$$a0y = -1.0120$$
$$a1x = -6.8299$$
$$a1y = 1.947$$

$$p1x = -3.5104$$
$$p1y = 3.6103$$
$$w_1 = -0.78504 \text{ rad/seg.}$$
$$w_2 = 0.7849 \text{ rad/seg.}$$

y en el que la barra a es B.D.V. con ley C determinada por

$$t0a = 2.01$$
$$v0a = 0.00$$
$$t1a = 4.003$$
$$v1a = 1.1922$$

y que se representa en la Figura 5-5-3. Para la otra -- barra b la ley C viene definida por

$$t0b = 2.23$$
$$v0b = 0.9072$$
$$t1b = 2.81$$
$$v1b = 0.001$$
$$t2b = 4.02$$
$$v2b = 5.6199$$

representada en la Figura 5-5-4. El error máximo que se comete es del 0.8 %.

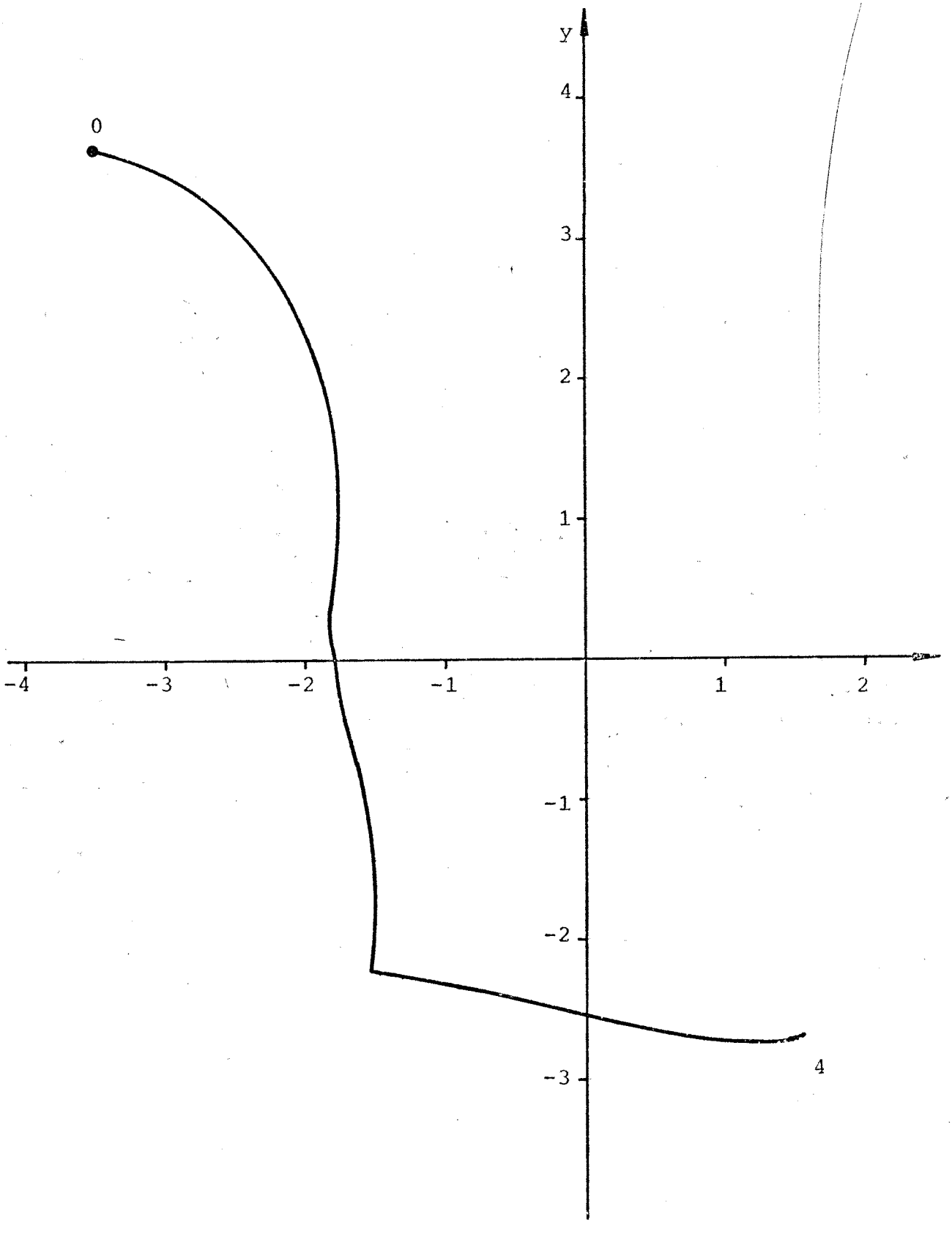


FIGURA 5-5-2

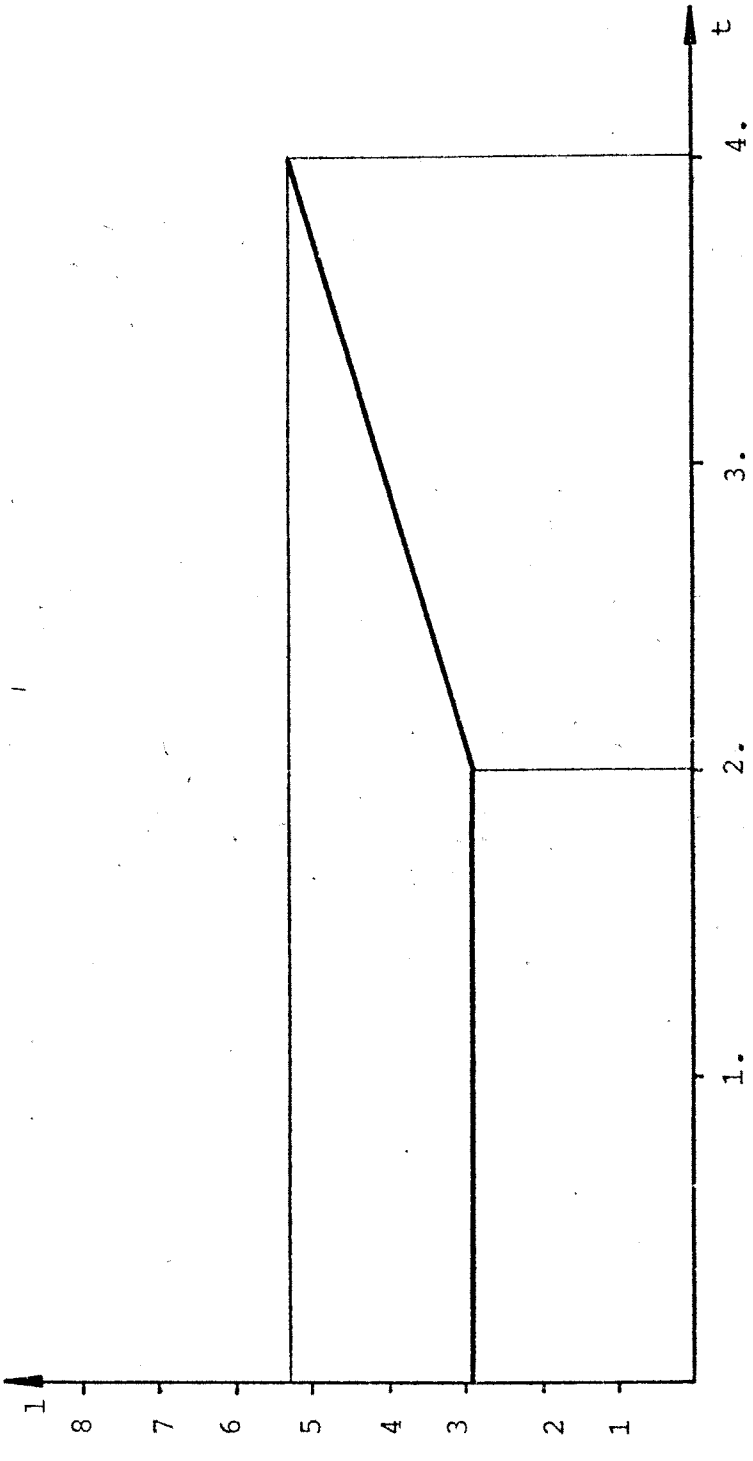


FIGURA 5-5-3

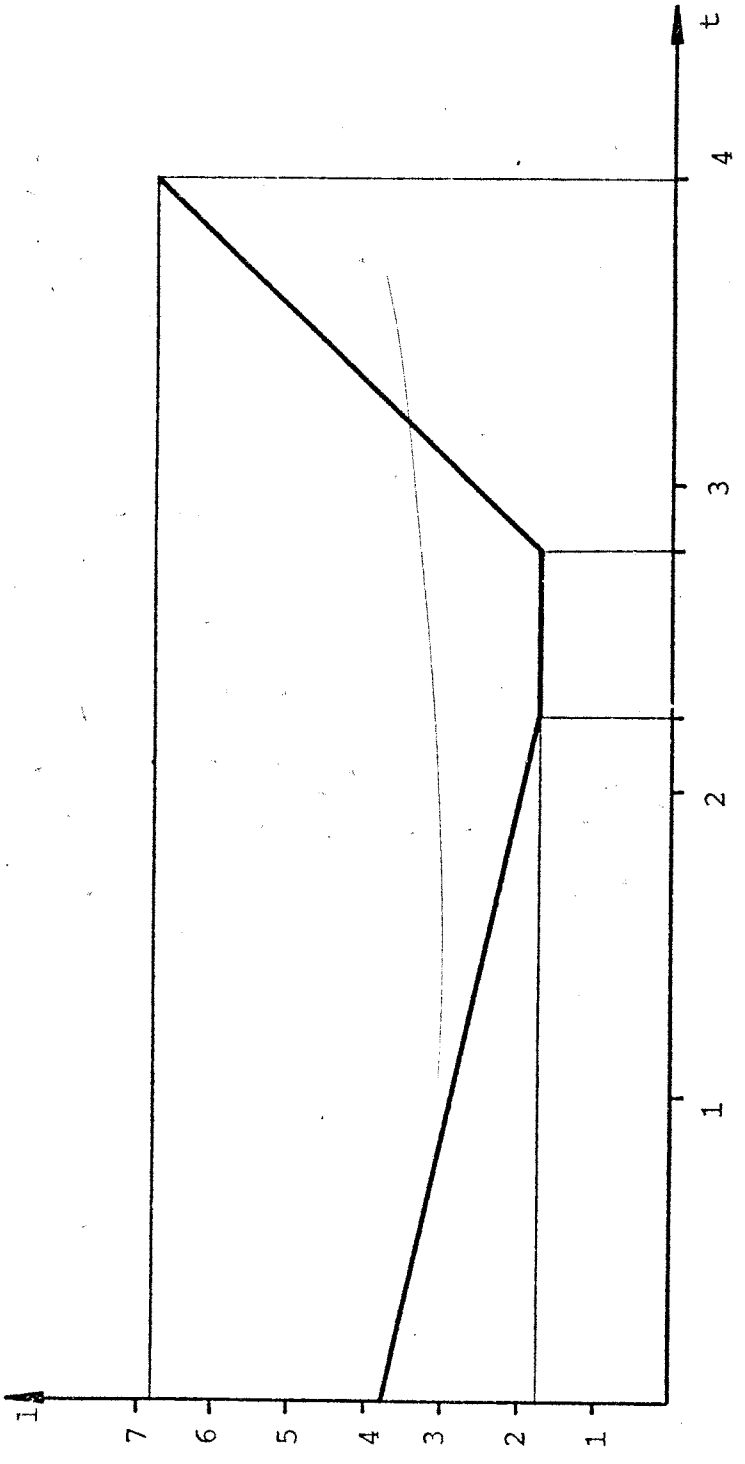


FIGURA 5-5-4

5.6. SINTESIS OPTIMA DE UNA CADENA CINEMATICA ESPACIAL - CON RESTRICCIONES.

Con objeto de comprobar la posibilidad de - aplicación a cadenas cinemáticas espaciales, y el tratamiento de las restricciones en el diseño, se realiza esta síntesis óptima. La cadena cinemática será la definida - en la Figura 5-6-1 formada por dos barras de longitud - constante y unidas entre sí por un par de rotación, ade-- más del que une a la primera de ellas a la barra fija, y situadas en el espacio como se puede ver en la citada Figura 5-6-1.

5.6.1. SISTEMA DE ECUACIONES.

Para la generación de una trayectoria con - el sistema de la Figura 5-6-1 se puede plantear que

$$f_j(\underset{\sim}{Z}) = 0 \qquad j = 1, 2, \ldots 12 \qquad (5\text{-}31)$$

para

$$\underset{\sim}{Z} = \left[a0x, a0y, aoz, a1x, a1y, a1z, u0x, u0y, u1x, u1y, w_1, w_2\right]^T \qquad (5\text{-}32)$$

y donde

$$f_j(\underset{\sim}{Z}) = (\underset{\sim}{QJ} - \underset{\sim}{PJ})^T(\underset{\sim}{QJ} - \underset{\sim}{PJ}) = 0 \qquad (5\text{-}33)$$

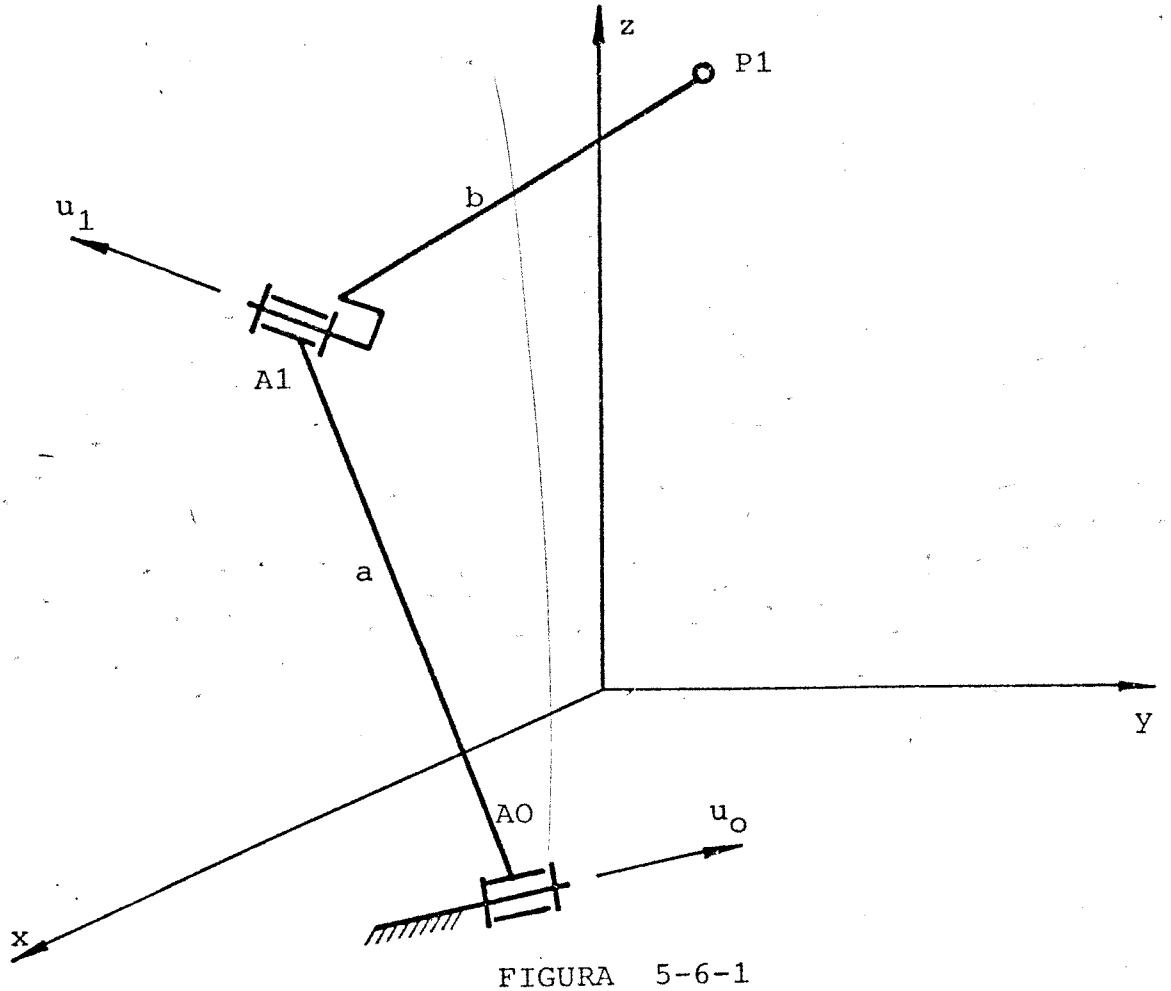


FIGURA 5-6-1

siendo $\underset{\sim}{QJ}$ el vector de posición de los puntos de la trayectoria deseada y $\underset{\sim}{PJ}$ lo es de los puntos de la trayectoria obtenida. Hay que tener en cuenta que

$$\underset{\sim}{PJ} = [R(\alpha 1j, uj)] \ (\underset{\sim}{PJ}^* - \underset{\sim}{AJ}) + \underset{\sim}{AJ} \qquad (5\text{-}34)$$

donde

$$\underset{\sim}{PJ}^* = [R(\theta 1j, u_o)] \ (\underset{\sim}{P1} - \underset{\sim}{AO}) + \underset{\sim}{AO} \qquad (5\text{-}35)$$

$$\underset{\sim}{AJ} = [R(\theta 1j, u_o)] \ (\underset{\sim}{A1} - \underset{\sim}{AO}) + \underset{\sim}{AO} \qquad (5\text{-}36)$$

y

$$\underset{\sim}{u_j} = [R(\theta 1j, u_o)] \ \underset{\sim}{u_1} \qquad (5\text{-}37)$$

donde u_o y u_1 son los vectores unitarios que definen la dirección del eje de rotación de los pares en el espacio.

El sistema que se puede obtener es como máximo de 12 ecuaciones con 12 incógnitas. Especificando ese número de puntos de la trayectoria, se realiza la síntesis con puntos de precisión. Junto con las coordenadas de cada punto, es necesario especificar, el instante de tiempo -- correspondiente a cada punto.

5.6.2. FUNCION OBJETIVO.

Para resolver el sistema de ecuaciones, me-- diante métodos de optimización, se puede enunciar éste de

de la forma:

Determinar el vector de diseño $\underset{\sim}{Z}$ que hace

$$\text{Min} \quad F(\underset{\sim}{Z}) \tag{5-38}$$

donde

$$F(\underset{\sim}{Z}) = \sum_{j=1}^{j=12} f_i \tag{5-39}$$

que desarrollando es

$$F(\underset{\sim}{Z}) = \sum_{j=1}^{j=12} (\underset{\sim}{QJ} - \underset{\sim}{PJ})^T (\underset{\sim}{QJ} - \underset{\sim}{PJ}) \tag{5-40}$$

Cuando se trata de la síntesis óptima, la función, es la misma, pero ahora, el número de puntos de comparación entre las trayectorias irá aumentando y por tanto j varia para cada ciclo de optimización. Programando la formulación del sistema y la función FUNC se puede realizar la síntesis óptima. En el caso de que se tengan restricciones, que no se puedan eliminar con transformaciones y cambios de variable, siempre se resolverá en forma de problema de optimización. La función objetivo será la misma, pero ahora hay que añadir a los programas la subrutina que define esas restricciones.

5.6.3. APLICACION NUMERICA.

Con una cadena cinemática como la de la Figura 5-6-1, se pretende realizar la generación de la trayec-

toria de la Figura 5-6-2, pero con las siguientes restricciones de diseño

$$7.0 < [(\underset{\sim}{A1} - \underset{\sim}{A0})^T(\underset{\sim}{A1} - \underset{\sim}{AO})]^{1/2} < 14.0$$

$$7.0 < [(\underset{\sim}{P1} - \underset{\sim}{A1})^T(\underset{\sim}{P1} - \underset{\sim}{A1})]^{1/2} < 14.0$$

además se conoce

$$\begin{aligned} p1x &= 10.267 \\ p1y &= 4.634 \\ p1z &= -5.723 \end{aligned}$$

y como el vector de diseño $\underset{\sim}{Z}$ es el indicado por (5-32) hay que determinar 12 variables. Realizando la síntesis con 12 puntos de precisión, pero resolviendo el sistema -- con el sexto segmento del programa OPRIS, ya que hay restricciones que cumplir.

La solución obtenida es:

$$\begin{aligned} a0x &= -1.10010 \\ a0y &= 2.43100 \\ a0z &= 0.95688 \\ a1x &= 4.97099 \\ a1y &= -3.21999 \\ a1z &= -7.65400 \\ u0x &= 0.623 \\ u0y &= 0.39969 \\ u1x &= 0.23559 \\ u1y &= 0.47229 \\ w_1 &= 5.9914 \\ w_2 &= -3.567 \end{aligned}$$

que si se toma como solución definitva para la generación de la trayectoria deseada, comete un error máximo del 12%, en cuanto a la norma de los vectores de posición de los - puntos de ambas trayectorias se refiere.

Tomando esta solución como de partida para - el programa CIOPT y realizando ciclos de optimización se obtiene como solución válida:

$$
\begin{aligned}
a0x &= -1.30119 \\
a0y &= \ \ 2.43100 \\
a0z &= \ \ 0.97289 \\
a1x &= \ \ 4.6020 \\
a1y &= -2.90999 \\
a1z &= -7.2340 \\
p1x &= 10.267 \\
p1y &= \ \ 4.-34 \\
p1z &= -5.723 \\
u0x &= \ \ 0.552 \\
u0y &= \ \ 0.458 \\
u1x &= \ \ 0.2278 \\
u1y &= \ \ 0.45120 \\
w_1 &= \ \ 5.8234 \\
w_2 &= -3.2090
\end{aligned}
$$

que se representa en la Figura 5-6-3, con la que se comete un error máximo del 0.2%, y cumple las restricciones im- - puestas.

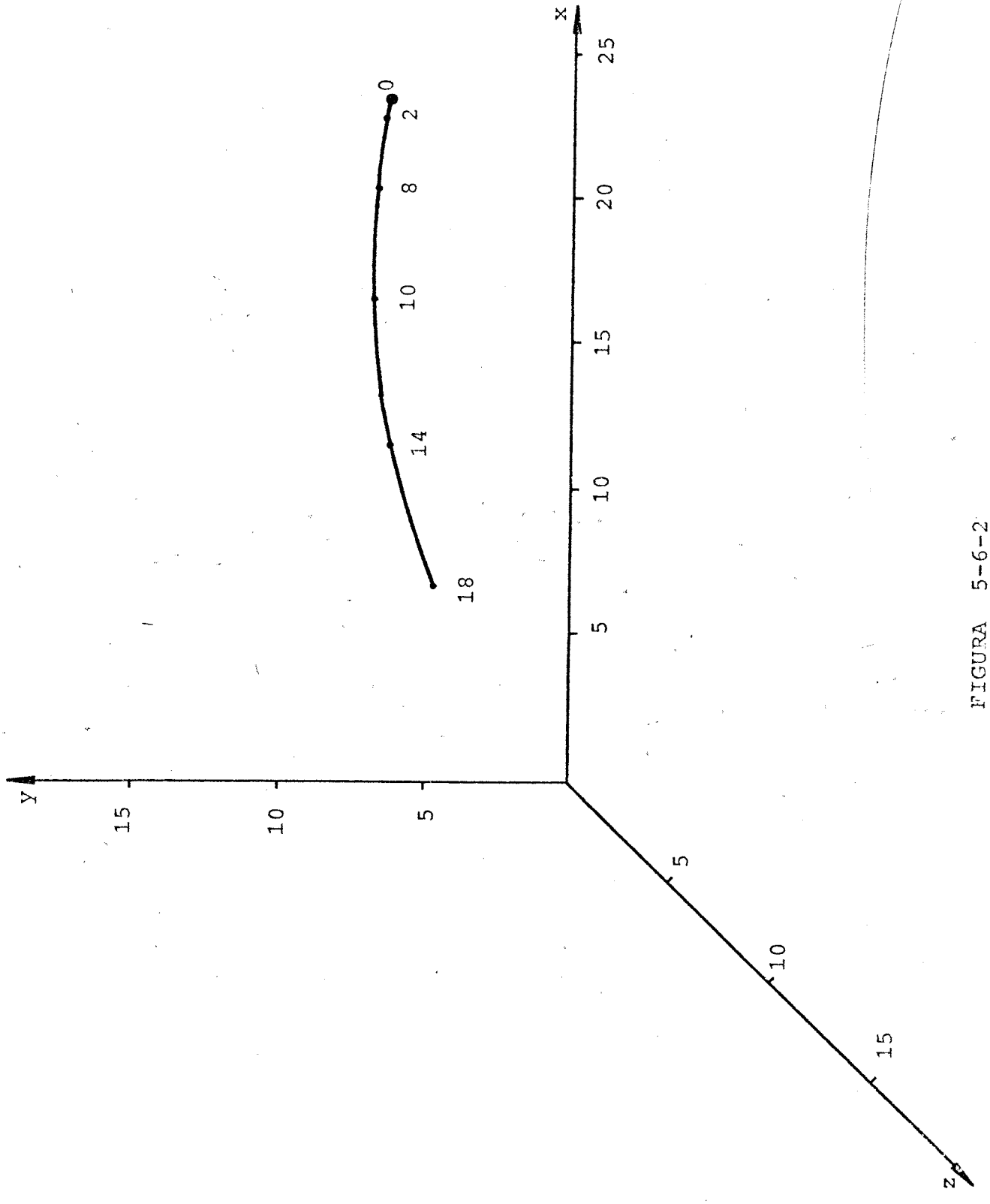


FIGURA 5-6-2

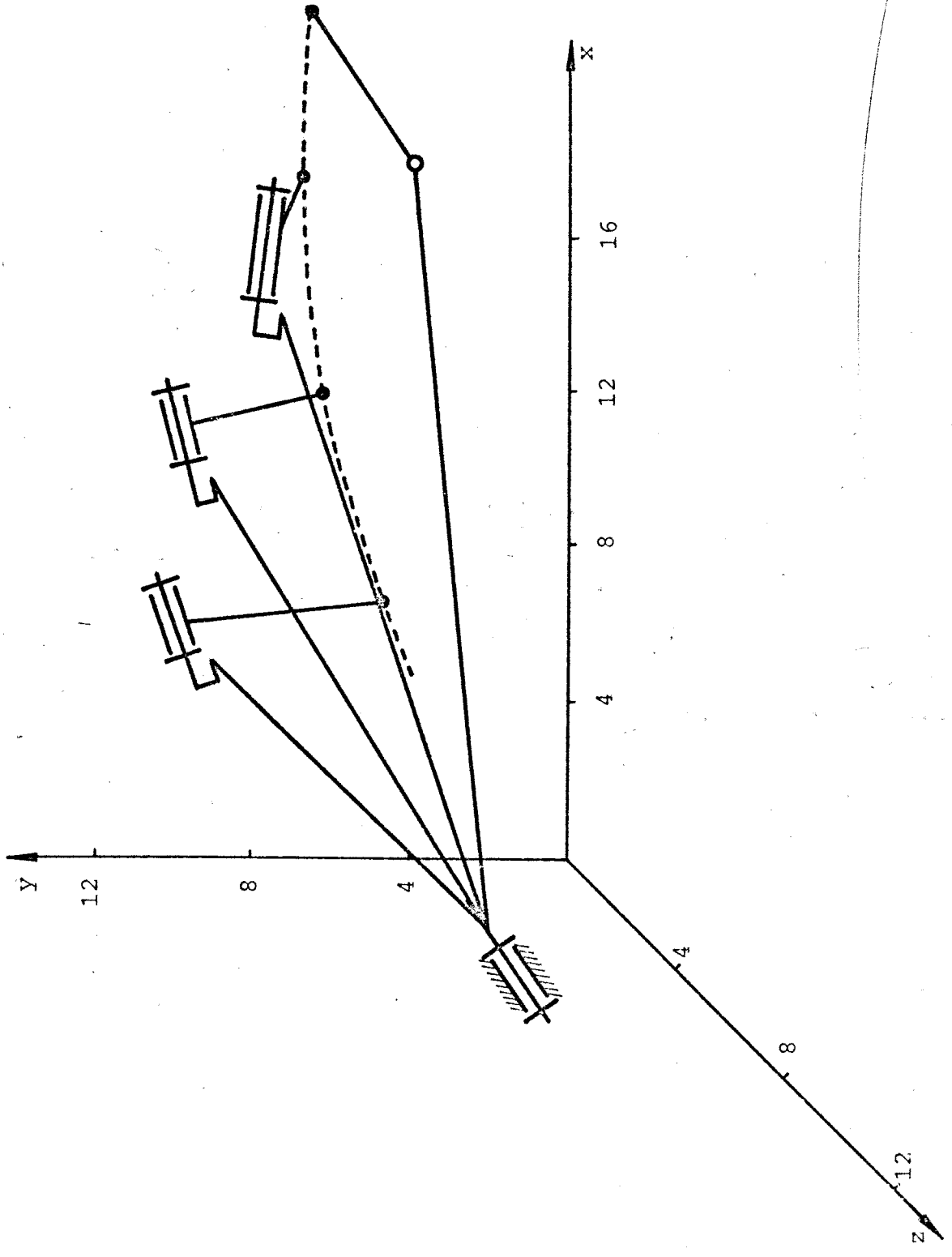


FIGURA 5-6-3

5.7. SINTESIS OPTIMA DE UNA CADENA CINEMATICA ESPACIAL CON B.D.V. LEY C.

Se realiza la síntesis óptima de generación de trayectorias, de una cadena cinemática abierta espacial a-- náloga a la de la Figura 5-6-1 pero en la que las barras -- que la forman son B.D.V. con ley C de variación en ambas.

5.7.1. SISTEMA DE ECUACIONES.

Para la realización de la síntesis de genera-- ción de trayectorias, con el sistema de la Figura 5-7-1, se plantea el mismo sistema de ecuaciones que en el caso ante- rior (5-3) y (5-33). Ahora bien, al ser las barras de - dimensión variable, las expresiones (5-34), (5-35) y -- (5-36) se transforman en :

$$\underset{\sim}{PJ} = \left[RL(h'_j, \alpha 1_j, u_j)\right] (\underset{\sim}{PJ}^* - \underset{\sim}{AJ}) + \underset{\sim}{AJ} \qquad (5\text{-}40)$$

$$\underset{\sim}{PJ}^* = \left[R(\theta 1_j, u0)\right] (P1 - A1) + \underset{\sim}{AJ} \qquad (5\text{-}41)$$

$$\underset{\sim}{AJ} = \left[RL(h_j, \theta 1_j, u0)\right] (\underset{\sim}{A1} - \underset{\sim}{A0}) + \underset{\sim}{A0} \qquad (5\text{-}42)$$

Los valores de h'_j y h_j se determinan de acuerdo con la ley C correspondiente a cada barra. El vector de diseño para -- este caso es:

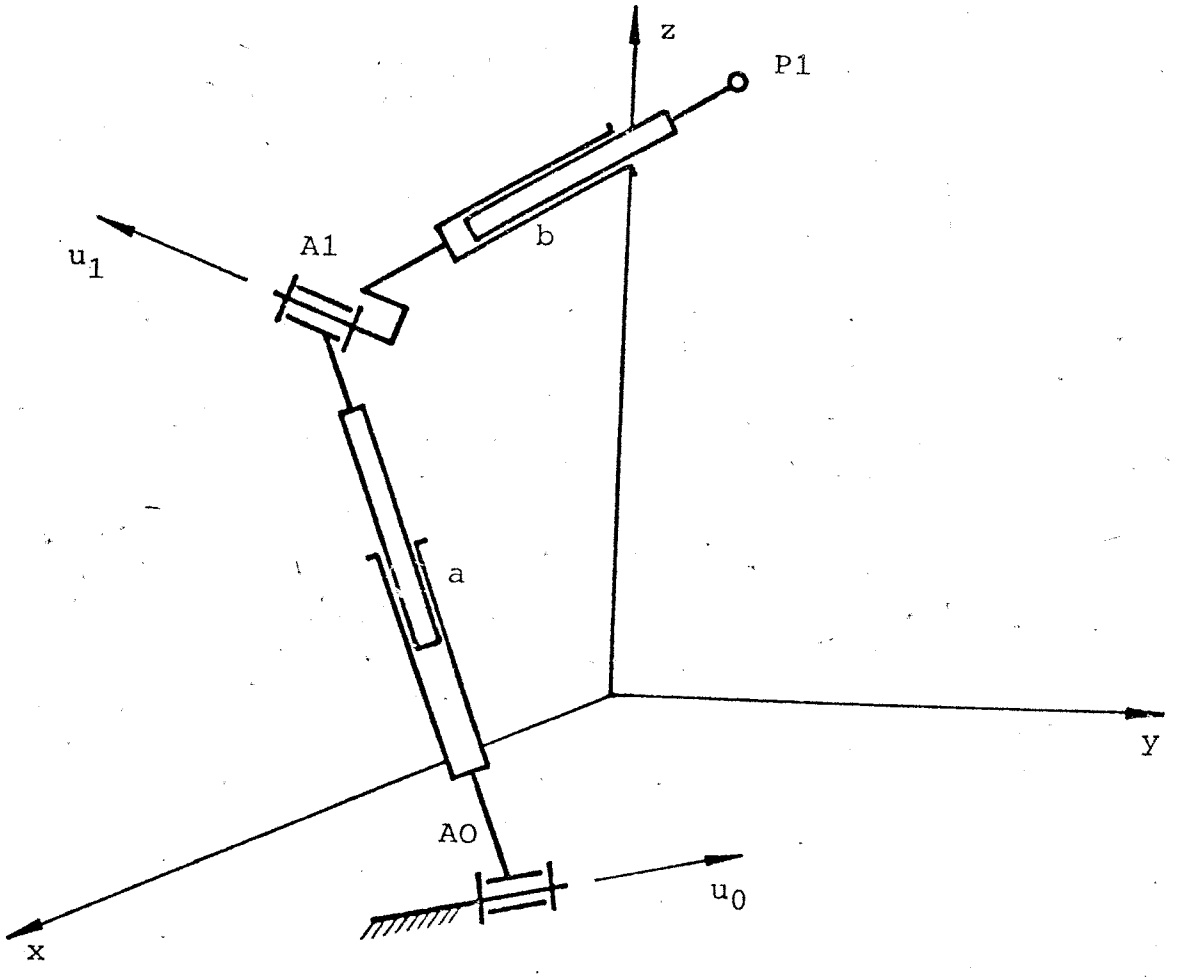


FIGURA 5-7-1

$$\underset{\sim}{Z} = \{a0x,a0y,a0z,a1x,a1y,a1z,u0x,u0y,u1x,u1y,w1,w2,$$
$$,ta0,va0... \}^T \qquad (5\text{-}43)$$

su dimensión estará determinada por el número de tramos -- que se fije para cada ley de variación. De acuerdo con la - dimensión del vector de diseño se especifican tantos puntos de precisión como sea necesario para realizar la síntesis.

5.7.2. FUNCION OBJETIVO.

La función objetivo será la misma que se ex-- presa en (5-40) es decir:

$$F(\underset{\sim}{Z}) = \sum_{j=1}^{j=m} (\underset{\sim}{QJ}-\underset{\sim}{PJ})^T(\underset{\sim}{QJ}-\underset{\sim}{PJ}) \qquad (5\text{-}44)$$

donde m es el número de puntos de precisión que se toman -- para realizar esta síntesis, y $\underset{\sim}{PJ}$ para este caso viene in-- dicado en la expresión (5-40). Para la síntesis óptima, el valor de m, que representa el número de puntos de com-- paración entre ambas trayectorias, irá cambiando de un ci- clo de optimización a otro, segun se indique.

Realizando la programación tanto del sistema - de ecuaciones como de la función objetivo, se obtiene la -- función FUNC necesaria para la realización de la síntesis óptima con los programas OPRIS y CIOPT.

5.7.3. APLICACION NUMERICA.

Usando una cadena cinemática como la de la figura 5-7-1 con ley C en las B.D.V., se desea generar la trayectoria especificada en la figura 5-7-2. Se sabe que:

p1x = 2.1221
p1y = 3.7696
p1z = 2.8741

y tomando 3 y 4, como el número de tramos a considerar en las leyes de variación de la dimensión de las B.D.V, se realiza la síntesis óptima obteniendo como solución aceptable, que esa trayectoria la puede aproximar bastante bien el sistema de la Figura 5-7-1 definido por:

a0x = 1.35719
a0y = -4.5598
a0z = 2.4567
a1x = 1.0526
a1y = 4.3401
a1z = -3.0103
p1x = 2.1221
p1y = 3.7696
p1z = 2.8741
u0x = 0.841
u0y = 0.511
u1x = 0.579
u1y = 0.6009
w1 = 8.963 rad/sg
w2 = 10.0132 rad/sg

y cuyas barras son de dimensión variable con la ley C de -- variación. Para la barra a esta ley está definida por:

ta0 = 1.7931
v0a = 0.7824
t1a = 3.887
v1a = 0.532
t2a = 5.989
v2a = -2.152

y que se representa en la Figura 5-7-3 . La ley C de la otra barra esta representada en la Figura 5-7-4 y se define con:

t0b = 1.013
v0b = 1.62
t1b = 3.5119
v1b = 2.20
t2b = 4.6122
v2b = 4.502
t3b = 6.0081
v3b = 0.003

En la Figura 5-7-5 se representa este sistema en cuatro posiciones distintas. El error que se comete en la aproxima-- ción de la trayectoria no supera el 1.5%.

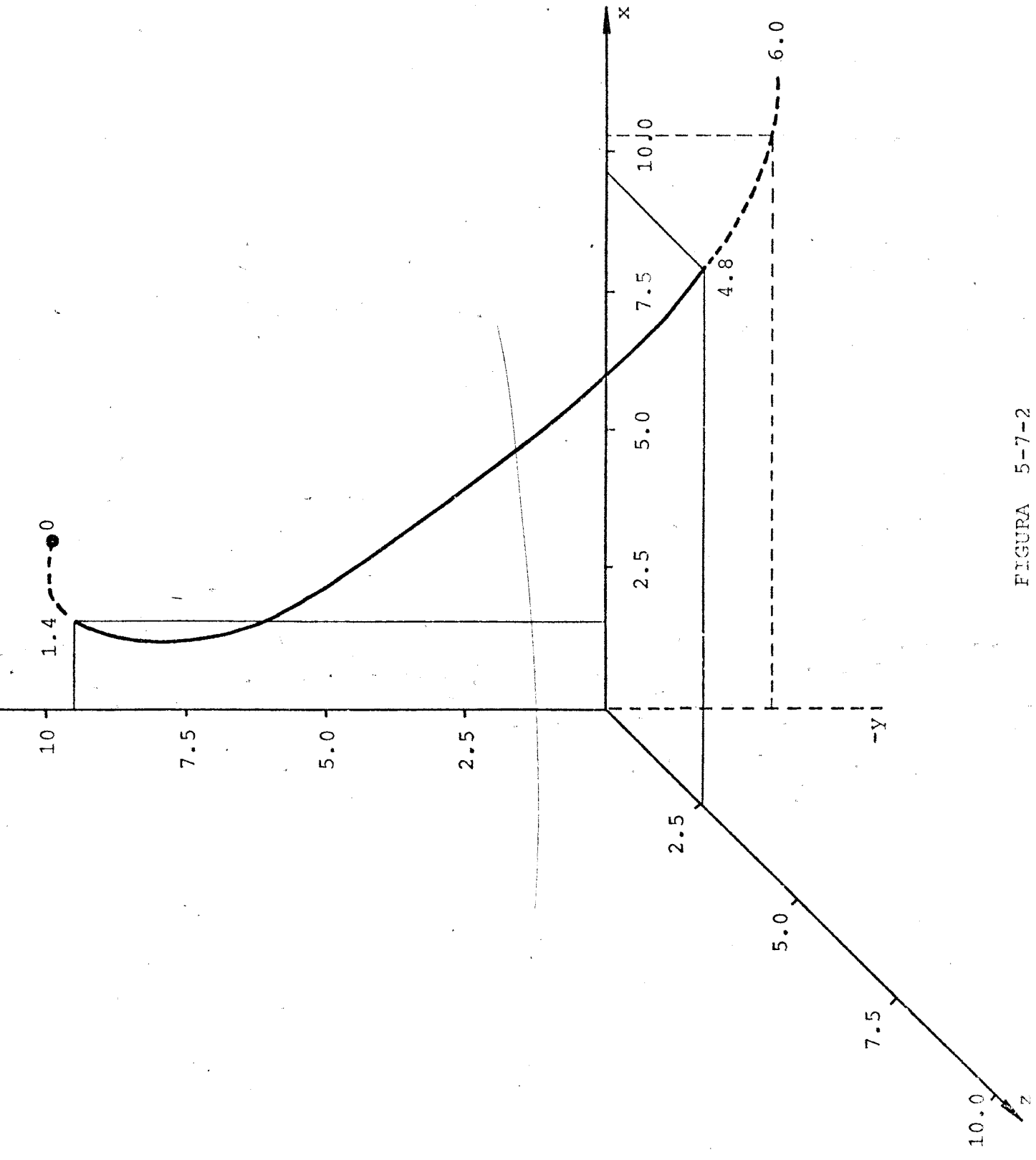


FIGURA 5-7-2

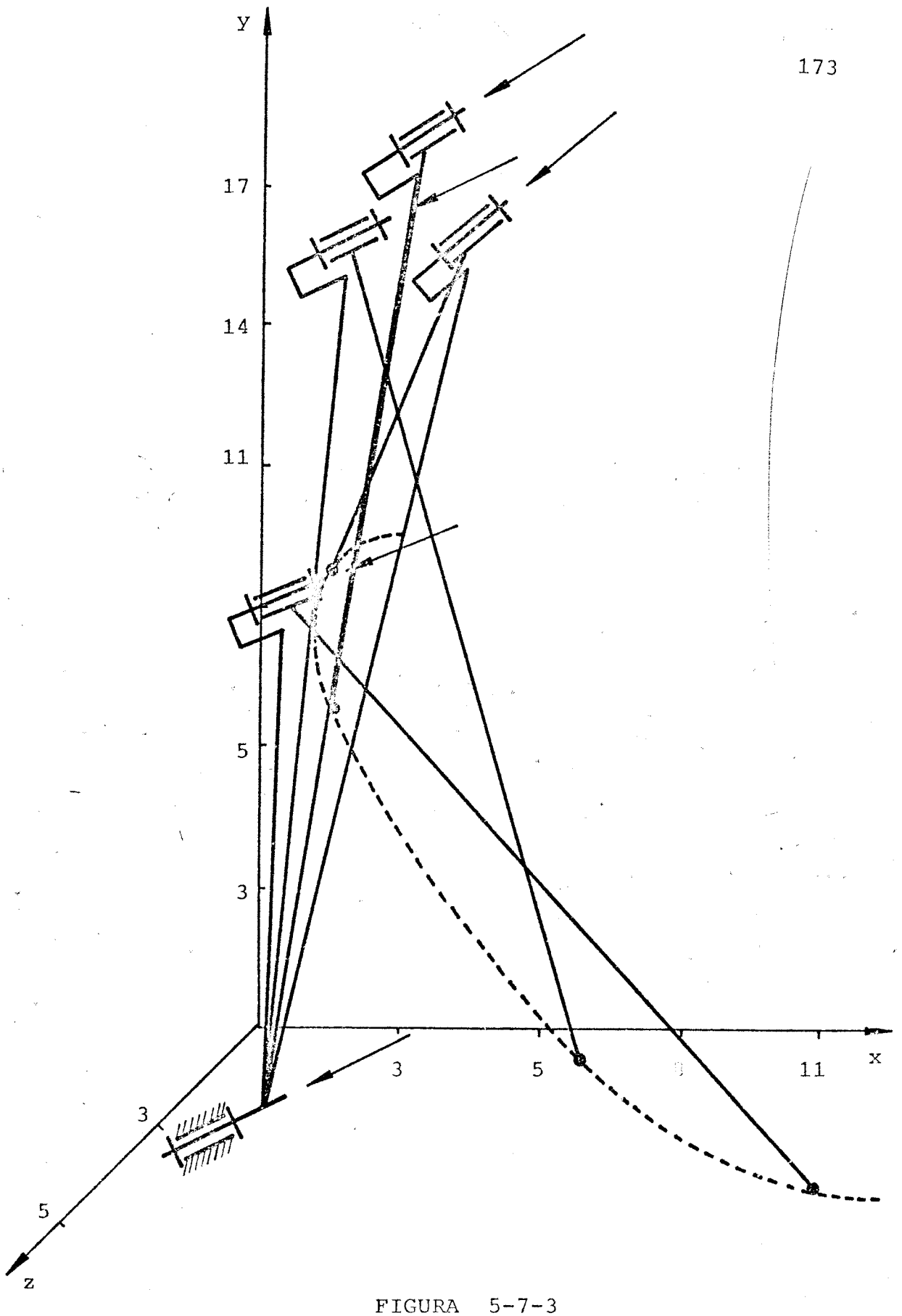


FIGURA 5-7-3

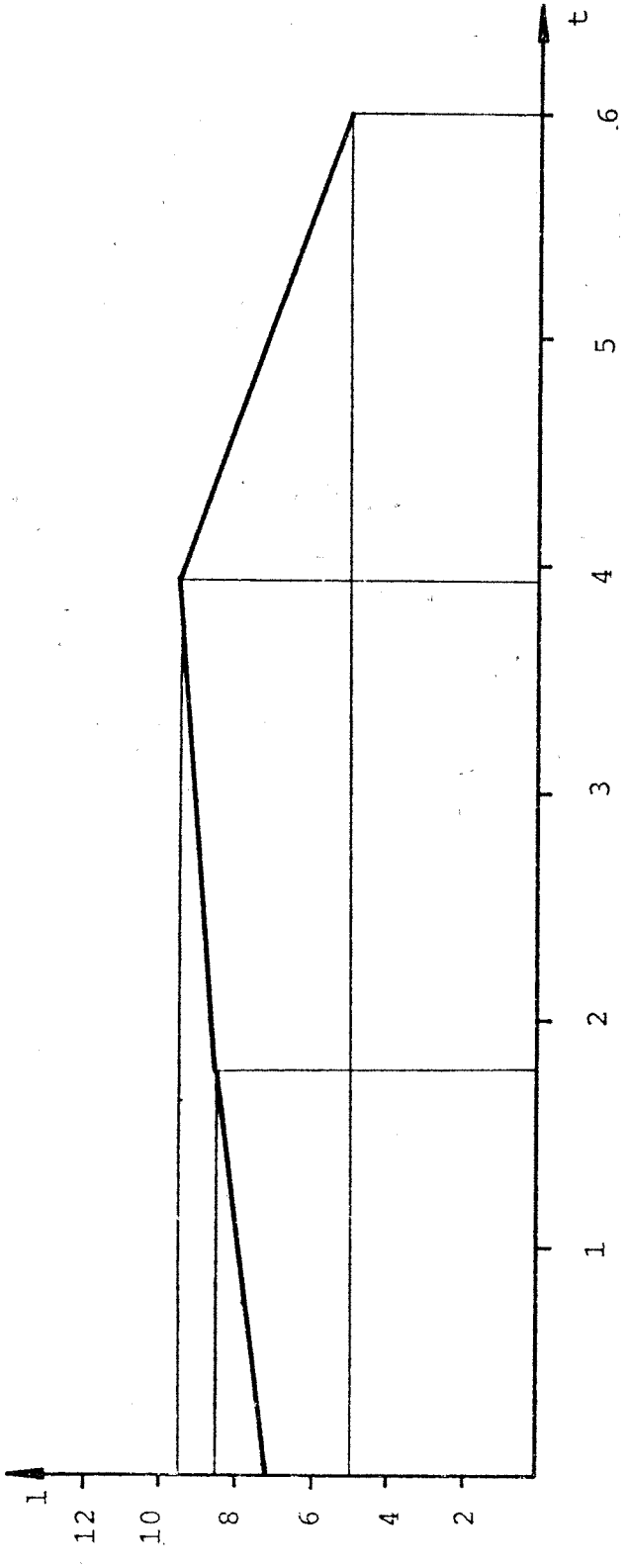


FIGURA 5-7-4

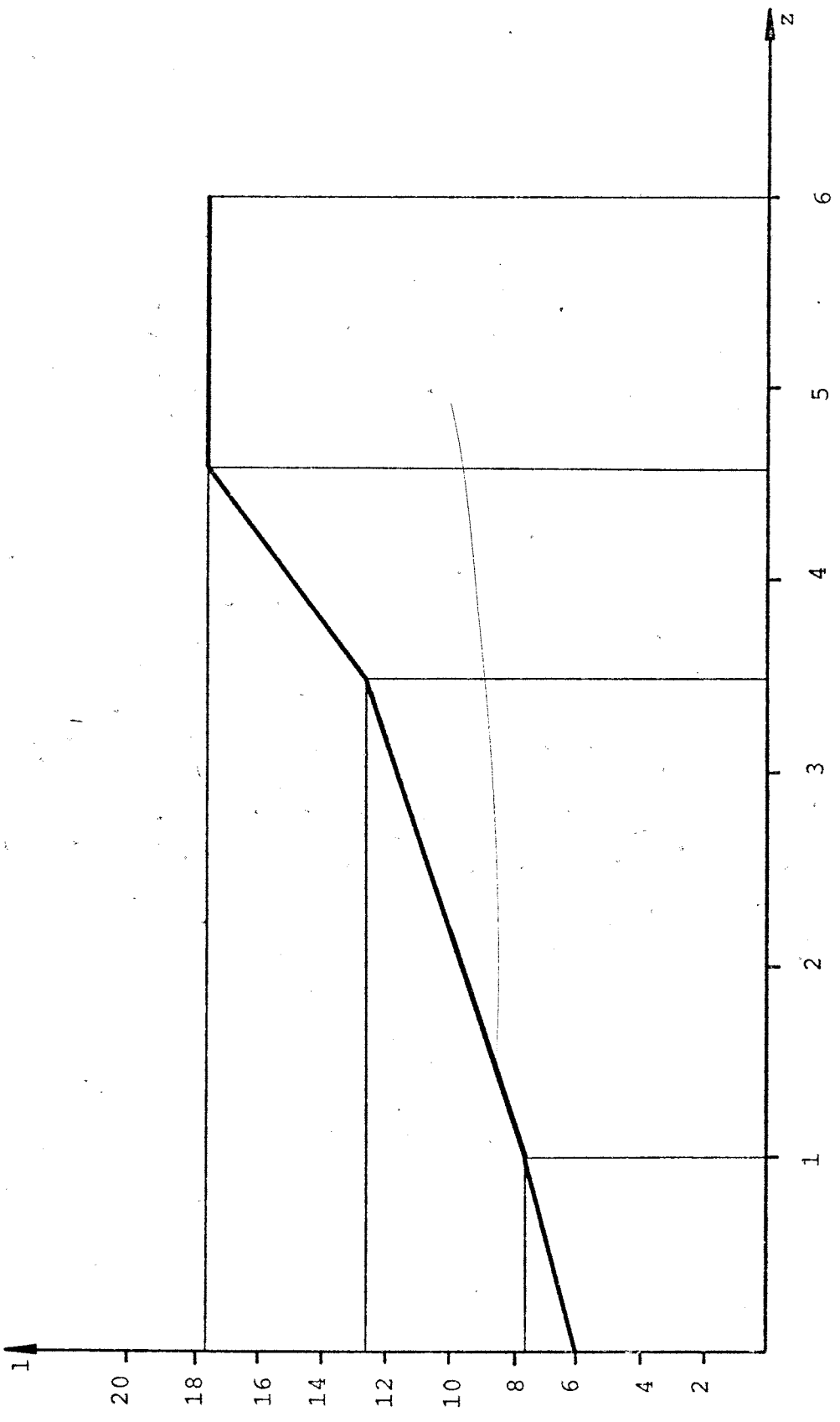


FIGURA 5-7-5

5.8. SINTESIS OPTIMA DE GENERACION DE UN CUADRADO.

Se realiza la síntesis de un mecanismo de cuatro barras plano con una B.D.V. de ley tipo D, que genera una trayectoria practicamente idéntica a un cuadrado. Como es lógico, si se trata de obtener tan solo un mecanismo capaz de generar un cuadrado no se acudirá a un cuatro barras como mecanismo base. Sin embargo de lo que realmente se trata es de estudiar las posibilidades de la síntesis de generación de trayectorias con mecanismos de D.D.V. El cuadrilátero articulado plano a utilizar para la síntesis es el de la Figura 5-8-1, es decir aquel en el que la barra de salida es B.D.V. La ley de variación para esa barra se considera de tipo D, también llamada ley general.

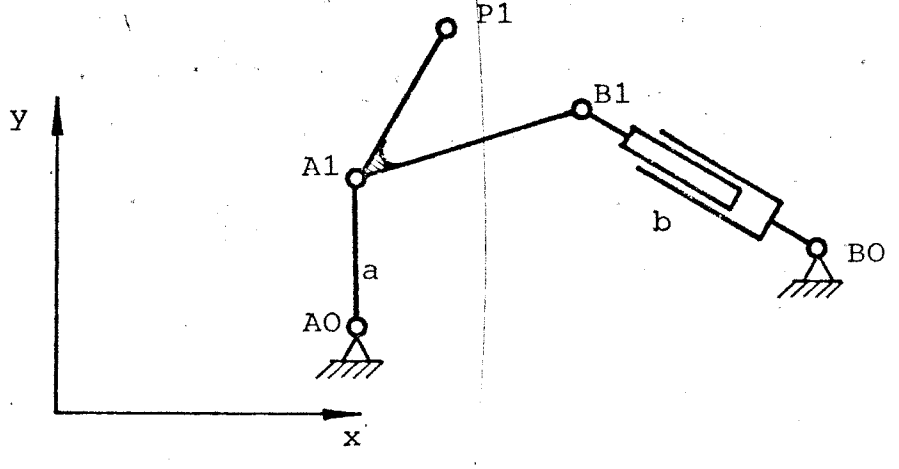


FIGURA 5-8-1

5.8.1. SISTEMA DE ECUACIONES.

El sistema de ecuaciones es el mismo que se -- plantea para la segunda aplicación con una única diferencia como es que en la expresión (5-12) el valor de la longi-- tud (l_j) de la B.D.V. en el instante j, viene ahora determinada por la interpolación, entre algunos puntos, de una función de spline cúbico, pues de esa forma se definiria el uso de la ley D ó general. De acuerdo con el número de puntos de interpolación que se deseen utilizar, para la aproximación de esta ley, asi será la dimensión del vector de diseño o número de variables a determianr.

5.8.2. FUNCION OBJETIVO.

La función objetivo es la misma que la del segundo caso estudiado, con la variante en cuanto a l_j antes indicada.

5.8.3. APLICACION NUMERICA.

De acuerdo con lo anterior se pretende generar el cuadrado de la Figura 5-8-2 con el mínimo error posible. Se sabe que:

p1x = 6.0
p1y = 6.0
w = 1.7453 rad/sg.

Realizando la síntesis óptima se obtiene como solución el mecanismo definido por:

a0x = 1.32369
a0y = 9.67666
b0x = 9.65243
b0y = 9.59440
a1x = 1.82308
a1y = 10.1770
b1x = 9.17794
b1y = 10.0970
p1x = 6.0
p1y = 6.0
w = 1.7453 rad/sg.

y en el que la barra de salida es B.D.V. con una ley de tipo D que se obtiene interpolando una función de polinomios de splines cúbicos a través de los puntos

(0.0,0.0)
(0.27,0.15481)
(1.26,1.04668)
(1.71,0.86754)
(2.16,0.43636)
(2.79,-0.04336)
(3.33, -0.37715)
(3.6,0.0)

En la Figura 5-8-3 se representa dicha ley a la que se sumó el valor de la longitud inicial (l_0) de esa barra. La Figura 5-8-4 es el mecanismo obtenido en varias posiciones, el error cometido es del orden del o.1%.

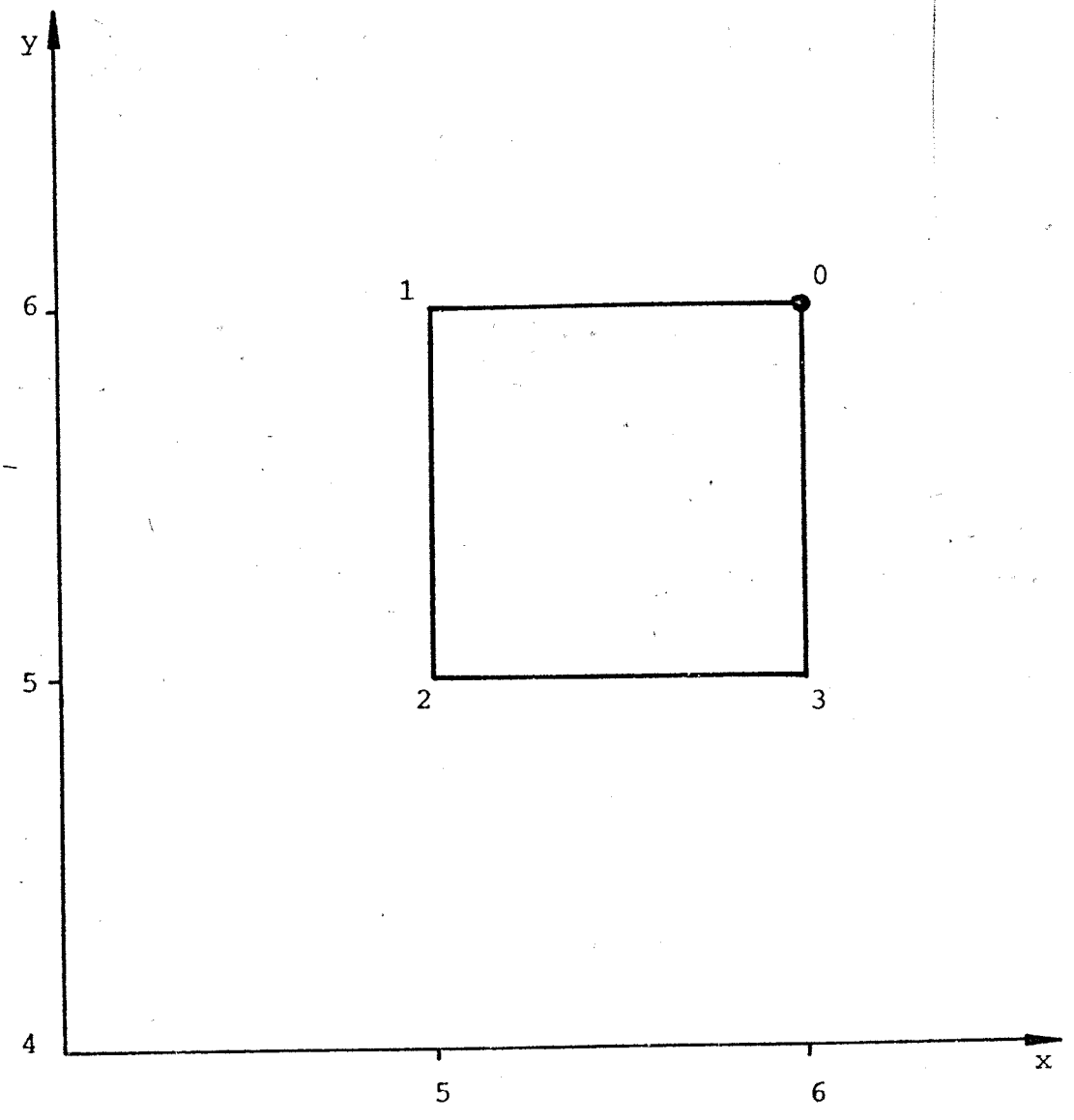


FIGURA 5-8-2

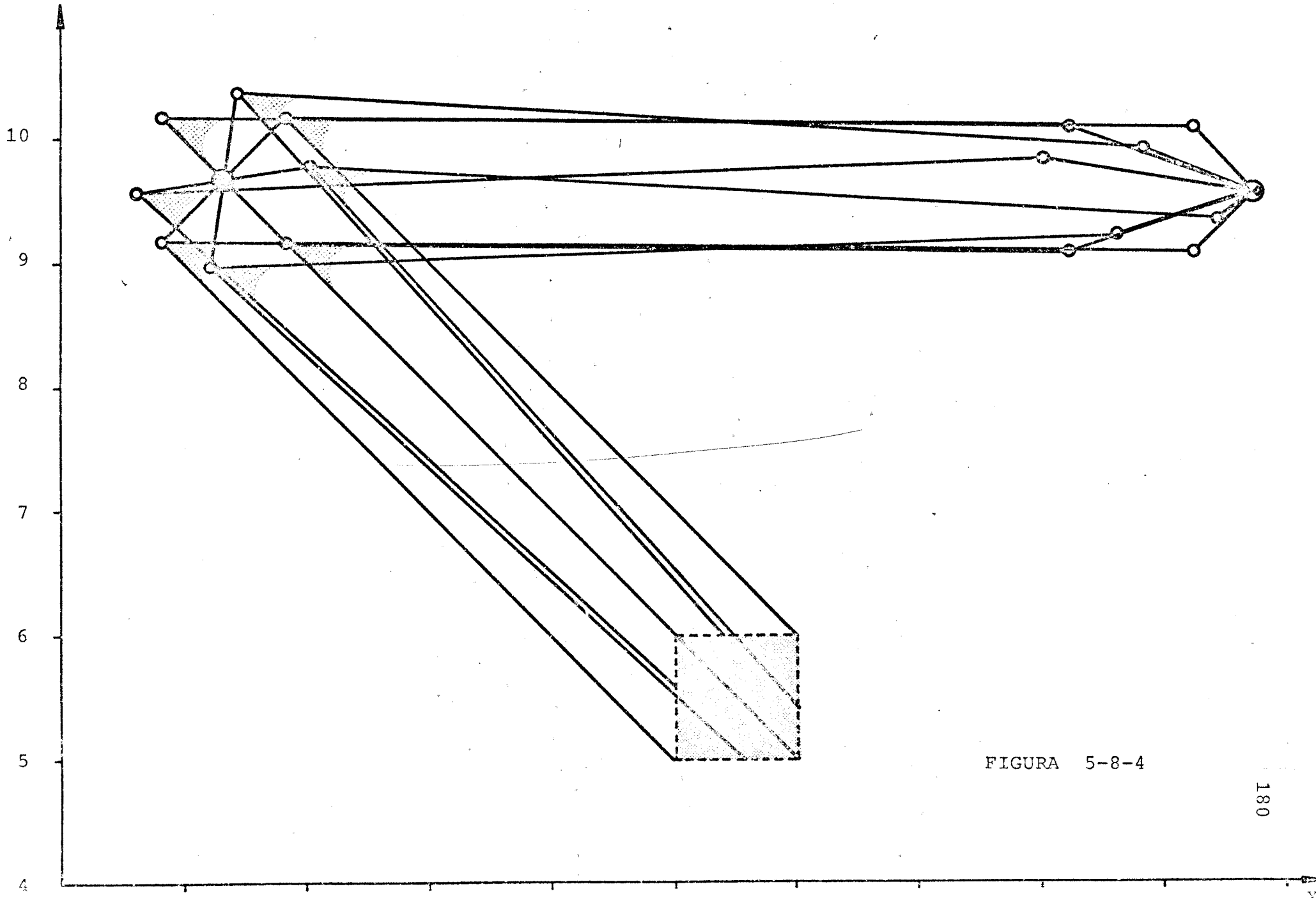


FIGURA 5-8-4

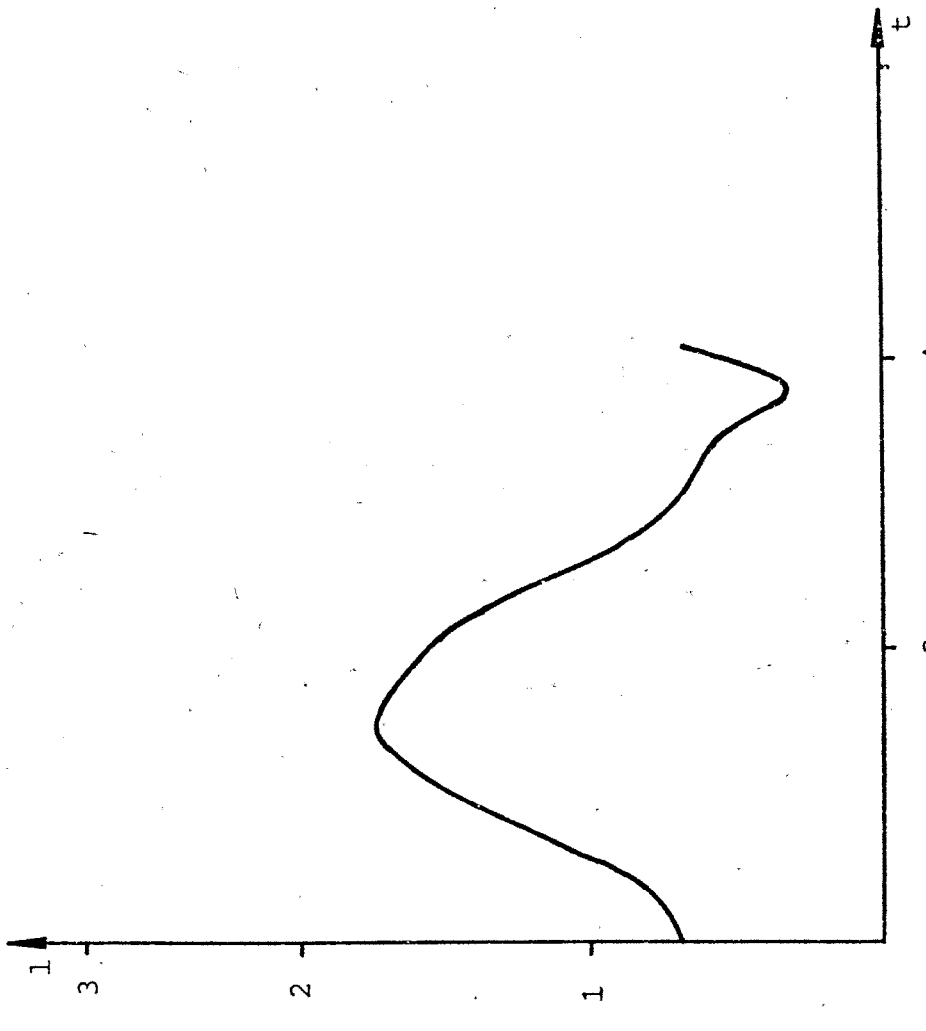


FIGURA 5-8-3

5.9. SINTESIS OPTIMA DE GENERACION DE UN CUADRADO. DOBLE MANIVELA.

En realidad se trata tan sólo de un caso derivado del resultado anterior. Asi, a la vista del mecanismo obtenido se piensa si será posible generar este cuadrado -- con un mecanismo como el de la figura 5-8-1 pero en el que tanto la barra de entrada como la de salida den vueltas completas.

El planteamiento será el mismo que en el caso anterior pero con la diferencia de que ahora para conseguir que sea un cuatro barras de doble manivela se añade una restricción al sistema de forma que el ángulo girado por la barra de salida sea tal que verifique:

$$\theta_{j-1} < \theta_j < 360^o \qquad j= 1,2....n$$

Realizando la síntesis óptima con esta restricción la solución obtenida es el mecanismo definido por:

a0x = 1.32309
a0y = 9.67666
b0x = -1.9927
b0y = 6.4965
a1x = 1.82308
a1y = 10.17700
b1x = -1.49539
b1y = 6.902131
p1x = 6.0
p1y = 6.0
w = 1.7453 rad/sg.

y la ley D de la barra de salida estará definida por la interpolación de una función de polinomios de splines cúbicos a traves de los puntos:

(0.0,0.0)
(0.54,0.160302)
(1.08,-0.11691)
(1.26,-0.10269)
(1.80,0.1229557)
(2.34,0.33448)
(2.70,1.3239)
(3.06,-0.09132)
(3.6,0.0)

En la Figura 5-9-1 se representa el mecanismo obtenido en sucesivas posiciones. El error cometido es del orden del 0.135%.

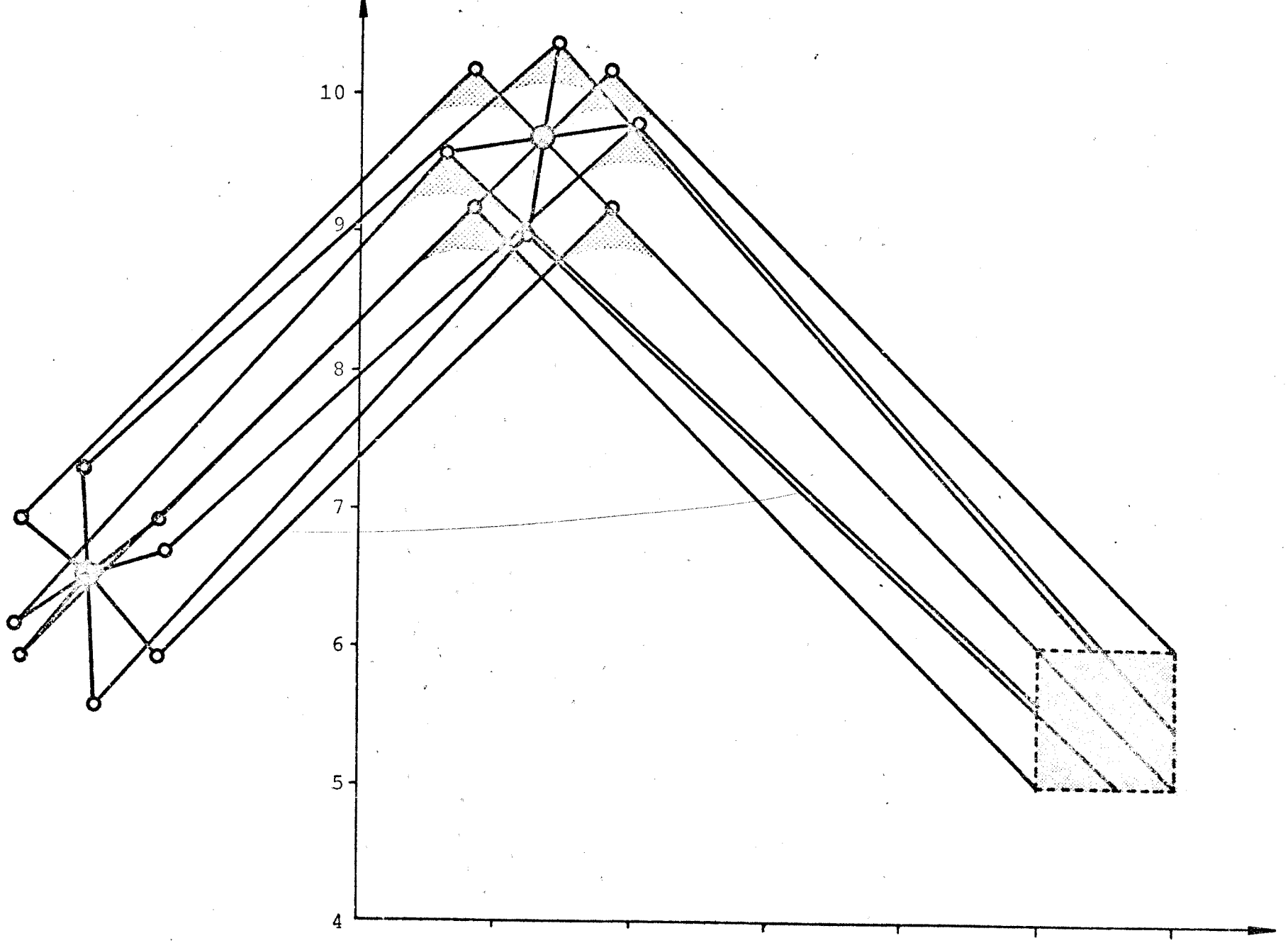


FIGURA 5-9-1

CAPITULO VI.

CONCLUSIONES Y DESARROLLO FUTURO.

6.1 Conclusiones.
6.2 Desarrollo futuro.

6.1. CONCLUSIONES.

De acuerdo con todo lo expuesto anteriormente y a la vista de las aplicaciones resueltas, se pueden concretar las realizaciones de este trabajo en dos aspectos generales como son, la utilización de mecanismos con barras de dimensión variable, y la síntesis óptima de generación de trayectorias aplicada a estas.

De entre los resultados obtenidos, en cuanto al uso de mecanismos de barras de dimensión variable se pueden destacar los siguientes:

1.- Se ha desarrollado el tratamiento de los mecanismos con B.D.V. modelizandolos a partir de una realidad física, como es, el imponer que algunas barras de este sistema son cilíndros hidráulicos o neumáticos.

2.- Se han formulado de acuerdo con esta modelización leyes de variación en la dimensión de estas barras, en función de las posibilidades de control que presentan estos tipos de cilíndros.

3.- Se ha obtenido una formulación para la aproximación de una ley general de variación de la dimensión de una barra, mediante la utilización de funciones de interpolación como son los polinomios de spline cúbicos. De esta forma con el control y conocimiento de tan solo unos pocos puntos se simula esa ley general.

4.- Se han desarrollado unas matrices de tranformación denominadas matrices de rotación-extensión, necesarias para la formulación de las ecuaciones de los mecanismos con B.D.V. Estas matrices permiten obtener las componentes de un vector que ha efectuado un giro y a la vez ha variado su longitud. Su uso facilita en gran medida el tratamiento de mecanismos planos o espaciales con B.D.V.

En lo que se refiere a la síntesis óptima de generación de trayectorias se pueden destacar los siguientes resultados:

1.- Se ha desarrollado una sistemática del cálculo de la solución de sistemas de ecuaciones algebraícas no lineales, a partir de una generación aleatoria de los valores iniciales. Lo que permite resolver la síntesis de generación de trayectorias con puntos de precisión con y sin restricciones de diseño.

2.- Se ha comprobado que la realización de lo que se ha llamado optimización cíclica ó ciclos de optimización, reporta notables ventajas, en cuanto a tiempo de ejecución y a precisión obtenida, en la síntesis óptima de generación de trayectorias.

3.- Como consecuencia de estos dos últimos apartados se ha formulado una metodología para reali

zar la síntesis óptima de generación de trayectorias, tomando como solución de partida para la optimización, la solución obtenida mediante la síntesis con puntos de precisión, y realizando a continuación ciclos de optimización con aumento de puntos de comparación, hasta obtener una solución aceptable.

4.- Se han desarrollado dos programas de ordenador que permiten realizar lo anteriormente expuesto para mecanismos planos o espaciales con barras de dimensión variable.

6.2. DESARROLLO FUTURO.

El desarrollo futuro de este trabajo puede contemplarse desde diferentes aspectos. Desde el punto de vista de la optimización surge la necesidad de estudiar y desarrollar nuevos algoritmos de optimización paramétrica que permitan obtener, de una forma más rápida y precisa soluciones a las síntesis óptimas de mecanismos.

En cuanto a los mecanismos con B.D.V. parece interesante que se realicen síntesis dinámicas, teniendo en cuenta no solo la variación en la longitud, sino también en la masa. De esta forma se podrá estudiar cual habrá de ser la distribución óptima de la masa, para que el mecanismo este perfectamente equilibrado ó produzca reacciones mínimas.

También parece interesante la realización de síntesis de generación de funciones y de guiado de biela - con mecanismos que tengan B.D.V.

REFERENCIAS.

- AHLBERG,J. H., NILSON,E. N. and WALSH,J. L. (1967)
The Theory of Splines and Their Applications
Academic Press, Inc.

- ALIZADE,R.I.,NOVRUZBEKOV,I.G.,SANDOR,G.N.(1975)
Optimization of four-bar funtion generating mechanisms using penalty functions with inequality and equelity constranints
Mechanism and Machine Theory,Vol 10,pp.327-336

- ANGELES,J. (1978)
Matrix Methods in Applied Kinematics
Division de estudios superiores. Facultad de Ingeniería
Universidad Autónoma de Mexico

- BAGCI,C. , and I.P.J. LEE (1974)
Optimum synthesis of plane mechanisms for the generation of paths and rigid-body positions via the linear superposition techinique
ASME paper No. 74-DET-10.

- BAGCI,C.(1980)
Optimum sythesis of multi-loop planar mechanisms via the linear partition of design equations.(Method of no misconceptions)
ASME. Paper No.80-DET-1

- BOX,M. J. (1965)
A new method of constrained optimization and a comparison with other methods
Computer Journal. Vol. 8 No. 1 pp. 42-52

- DANTZIG,G.B. (1949)
Programming of independent activities II
Mathematical Model, Econometrica, Vol 17 pp. 200-211

- DAVIDON, W. C. (1959)
Variable Metric Method for Minimization
Argone Nat. Lab. ANL-5990 Rev.

- DE BOOR, C. (1978)
A Practical Guide to Splines
Springer-Verlag

- FIACCO,A.V. and G.P. McCORMICK (1964)
Compunational algorithm for the sequential unconstrained minimization for nonlinear programming.
Management Science, Vol. 10, pp. 601-617

- FIACCO,A.V. and G.P. McCORMICK (1968)
Nonlinear programming: Sequential Unconstrained Minimization Techniques.
John Wiley and Sons.

- FLETCHER,R. and M.J.D. POWELL (1963)
A rapidly convergent descent method for minimization.
Computer Journal, Vol. 6, pp. 163-168

- FLETCHER,R., and C.M. REEVES (1964)
Function minimization by conjugate gradients.
Computer Journal, Vol. 7 , pp. 149-154

- FOX, R.L. (1971)
Optimization Method for Engineering Desing
Addison-Wesley Publishing Company.

- FOX, R.L., and K.D. WILLMERT. (1967)
Optimum desing of curve generating linkage with inequa--lity constraints.
Journal of Engineering for Industry, Trans. ASME, Series B Vol. 89, No. 1, pp. 144-152.

- FOX,R.L.,GUPTA,K.C.,(1973)
Optimization technology as applied to mechanism design
Journal of Engineering for Industry,Trans. ASME. Mayo, pp.657-663

- FREUDENSTEIN, F., and G.N. SANDOR (1959)
Synthesis of path generating mechanisms by a programmed digital computer.
Trans ASME, Series B, Vol. 81,No. 2,pp. 15-22

- GARBAROUK, V.V., and LEBEDEV P.A. (1980)
Synthesis of spatial automatic operator with the aid of electronic digital computers.
Mec. and Mac. Theory, Vol 15, pp. 9-17

- GARREYY, R.E., and A.S. HALL (1968)
Optimal synthesis of randomly generater linkages.
Journal of Engineering for Industry, Trans. ASME, Series B Vol. 90, No. 3,pp.475-480

- GUPTA V.K. (1973)
Computer-Aided synthesis of mechanisms using nonlinear programming
Trans. ASME. Paper No. 72-WA/DE-3.

- GUPTA V.K. (1973)
Kinematic analysis of plane and spatial mechanisms.
Trans. ASME, Paper No.72 - Mech - 1.

- HAARHOFF, P.C. BUYS J.D. (1970)
A new method for the optimization of a nonlinear function subjet to nonlinear constraints.
Computer Journal Vol. 13, pp. 178-184

- HAN, C.Y. (1966)
A general method for the optimum design of mechanisms.
Journal of Mechanisms Vol. 1 , No. 4, pp. 301-313

- KIMBRELL,J.E.,HUNT,K.H.(1980)
A classification of coupler-line envelopes from hinged four-bar linkages
ASME. Paper No.80-DET-42

- KRAMER,S.N.,SANDOR,G.N.(1975)
Selective precision synthesis.A general method of optimization for planar mechanisms
Journal of Engineering for Industry,Trans. ASME. Mayo, pp.689-701

- McCARTHY,J.M.,ROTH,B.(1980)
Instantaneous properties of trajectories generated by planar, spherical, and spatial rigid body motions
ASME. Paper No.80-DET-19

- NOLLE, H. (1974,a)
Linkage coupler curve synthesis: A historical review I. Developments apto 1875
Mec and Mach. Theory Vol. 9, pp. 147-168

- NOLLE, H. (1974,b)
Linkage coupler curve synthesis: A historical review II Developments apto 1875
Mec and Mach. Theory Vol. 9, pp. 325-348

- NOLLE, H. (1975)
Linkage coupler curve synthesis: A historical review III Spatial synthesis and optimization
Mech. and Mach. Theory, Vol 10, pp. 41-65

- PAPPAS, M. (1980)
An improved direct search numerical optimization procedure
Comp. Struct. Vol. 11, pp. 539-557

- PAUL, B. (1979)
Kinematics and Dynamics of Planar Machinery
Prentice-Hall Inc.

- POWELL, M.J.D. (1964)
An efficient method for finding the minimum of a function of several variables without calculating derivatives
Computer Journal Vol. 7 , No. 4 , pp. 303-307

- RAO, A.C. (1979)
Synthesis of 4-bar-function-generators using geometric programming
Mech. and Mach. Theory, Vol 14, pp. 141-149

- RAO, S.S. (1978)
Optimization Theory and Applications
Wiley Eastern Limited

- ROSE, R., and G.N. SANDOR (1973)
Direct analytical synthesis of four-bar function generator with optimal structural error.
Journal of Engineering for Industry,Trans ASME, Series B
Vol. 95, No. 2 .

- ROSENBROCK H.H. (1960)
An automatic method for finding the greats or least values of function.
Computer Journal, Vol. 3, pp. 175-184

- ROTH, B., G.N. SANDOR, and F.FREUDENSTEIN. (1962)
Synthesis of four-bar path generation mechanisms with optimum transmission characteristics
Transactions of the Seventh Conference on Mechanisms,Purdue University.

- SEIREG, A. (1976)
The role of mechanisms research in some areas of current and future needs.
Proceeding of N.S.F.. Workshop on new directions for Kinematics research.

- SUCALA,F.(1980)
Determination of kinematic characteristics of a mechanism with a variable leng connecting rod
ASME. Paper No.80-DET-35

- SUH, C.H., and C.W. RADCLIFFE (1978)
Kinematics and Mechanisms Design.
John Wiley and Sons.

- SUTHERLAND,G.H.,SIDDALL,J.N.(1974)
Dimensional synthesis of linkages by multifactor optimization
Mechanism and Machine Theory,Vol 9,pp. 81-95

- SUTHERLAND, G.H. (1976)
Mixed exact-approximate planar mechanism position synthesis.
Trans. ASME, ASME Paper No.76-DET-29

- TANABE,K.,(1979)
Differential geometric methods for solving nonlinear constrained optimization problems and a related system nonlinear equations: global analisis and implementation
Méthodes numériques dans les sciences de l'ingénieur
Dunod, pp.547-556

- TESAR (1976)
A personal view of the past, present and future of mechanism science.
Proceeding of N.S.F.. Workshop on new directions for Kinematics reseach.

- THOMPSON,B.S.(1975)
A survey of analytical path-synthesis techniques for plane mechanisms
Mechanism ans Machine Theory,vol 10,pp.197-205

- TOMAS, J. (1968)
The synthesis of mechanisms as nonlinear programming problem
Journal of Mechanisms, Vol.3,pp. 119-130

- WOLFE, M.A. (1978)
Numerical Methods for Unconstrained optimization.
Van Nostrand Reinhold Company.

APENDICE A

A.- MATRICES ROTACION-EXTENSION.

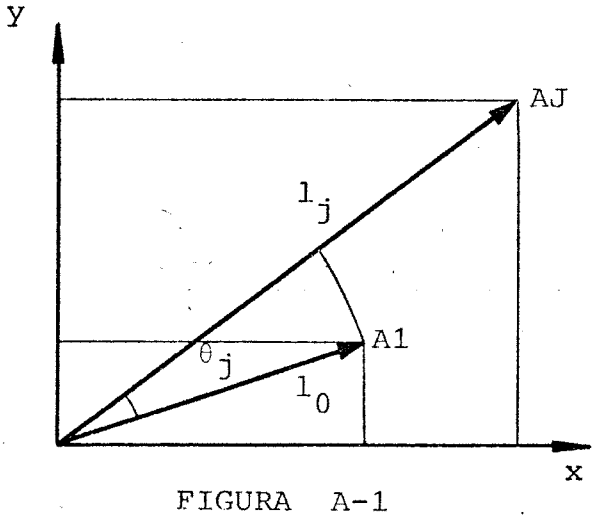


FIGURA A-1

El vector de posición A1 gira un ángulo θj y su longitud varía desde l_o l_j. Se trata de encontrar la matriz que representa esa transformación

$$\underset{\sim}{AJ} = |T| \underset{\sim}{A1} \qquad (A\text{-}1)$$

de acuerdo con la Figura A-1, las componentes del vector $\underset{\sim}{AJ}$ se pueden expresar en función de las componentes de A1 de la forma:

$$ajx = a1x \cos \theta j - a1y \operatorname{sen} \theta j + \frac{l_j - l_o}{l_o}(a1x \cos \theta j - a1y \operatorname{sen} \theta j) \qquad (A\text{-}2)$$

$$ajy = a1x \operatorname{sen} \theta j + a1y \cos \theta j + \frac{l_j - l_o}{l_o}(a1x \operatorname{sen} \theta j + a1y \cos \theta j) \qquad (A\text{-}3)$$

que operando serán:

$$ajx = (a1x \cos \theta j - a1y \operatorname{sen} \theta j)\left\{1 + \frac{l_j - l_o}{l_o}\right\} \qquad (A\text{-}4)$$

$$ajy = (a1x \text{ sen } \theta j + a1y \cos \theta j) \left[1 + \frac{l_j - l_o}{l_o}\right] \quad (A-5)$$

haciendo

$$h = \frac{l_j}{l_o} \quad (A-6)$$

se tendrá:

$$ajx = h(a1x \cos \theta j - a1y \text{ sen } \theta j) \quad (A-7)$$

$$ajy = h(a1x \text{ sen } \theta j + a1y \cos \theta j) \quad (A-8)$$

que expresandolo en forma matricial será:

$$AJ = \begin{vmatrix} h \cos \theta j & -h \text{ sen } \theta j \\ h \text{ sen } \theta j & h \cos \theta j \end{vmatrix} A1 \quad (A-9)$$

y que indica que la matriz de transformación T que se - buscaba viene representada por

$$|T| = |RL(h,\theta j)| = \begin{vmatrix} h \cos \theta j & -h \text{ sen } \theta j \\ h \text{ sen } \theta j & h \cos \theta j \end{vmatrix} \quad (A-10)$$

y que puede recibir el nombre de matriz de rotación extensión.

Ahora bien si se tiene en cuenta la tercera dimensión, se puede decir que en

$$\begin{Bmatrix} ajx \\ ajy \\ ajz \end{Bmatrix} = \begin{vmatrix} h\cos\theta j - h\,\text{sen}\,\theta j & 0 \\ \text{sen}\,\theta j & h\cos\theta j & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{Bmatrix} a1x \\ a1y \\ a1z \end{Bmatrix} \quad \text{(A-11)}$$

la matriz de transformación representa la matriz de rotación extensión en el espacio, siendo el eje de giro z uno de los ejes coordenados. De forma análoga se pueden expresar las matrices de rotación-extensión para los otros ejes coordenados; así para el eje y será:

$$\begin{Bmatrix} ajx \\ ajy \\ ajz \end{Bmatrix} = \begin{vmatrix} h\cos\beta & 0h\,\text{sen}\beta \\ 0 & 1 & 0 \\ -h\,\text{sen}\beta & oh\cos\beta \end{vmatrix} \begin{Bmatrix} a1x \\ a1y \\ a1z \end{Bmatrix} \quad \text{(A-12)}$$

y para el eje x

$$\begin{Bmatrix} ajx \\ ajy \\ ajz \end{Bmatrix} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & h\cos\gamma & -h\,\text{sen}\gamma \\ 0 & h\,\text{sen}\gamma & \cos\gamma \end{vmatrix} \begin{Bmatrix} a1x \\ a1y \\ a1z \end{Bmatrix} \quad \text{(A-13)}$$

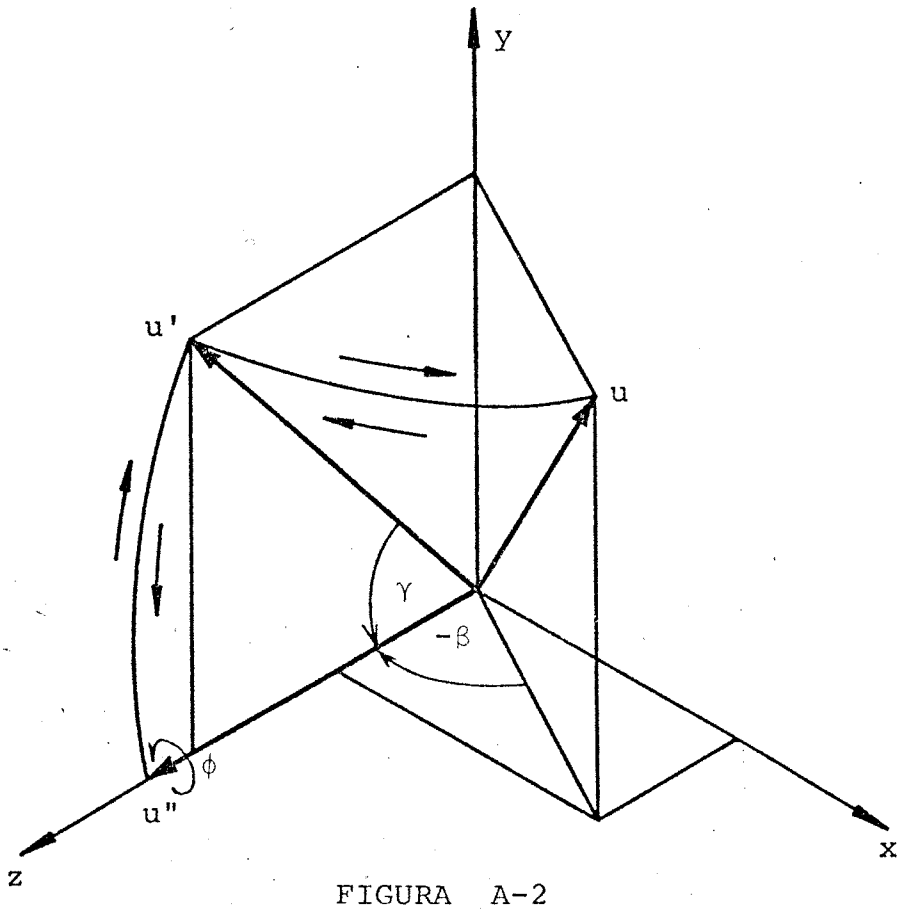


FIGURA A-2

Disponiendo de estas matrices de rotación extensión en el espacio, se puede obtener la posición final de un vector, a partir de la inicial, cuando éste ha efectuado una rotación en el espacio y ha sufrido además una variación en su longitud. Sin embargo lo usual y más comodo para trabajar en tres dimensiones es utilizar la matriz de rotación alrededor de un eje cualquiera en el espacio. Por tanto, también parece necesario, el que se formule la matriz de rotación extensión en el espacio alrededor de un eje cualquiera.

De acuerdo con la Figura A-2 una forma de determinar esa matriz, es aplicar al eje u las matrices de rotación alrededor de los ejes cartesianas con objeto de hacerlo coincidir con uno de ellos, entonces se aplica el giro deseado y luego se realizan las rotaciones inversas hasta llegar el eje a su posición real. Estas transformacio--

nes están indicadas en la Figura A-2 y en forma matricial se pueden expresar:

$$AJ = |R(\beta,y)|\ |R(-\gamma,x)|\ |RL(\phi,z)|\ |\ R(\gamma,x)|\ |\ R(-\beta,y)|A1 \qquad (A-14)$$

y como se cumple que

$$\text{sen}\gamma = u_y \qquad (A-15)$$

$$\cos\gamma = \sqrt{u_x^2 + u_y^2} \qquad (A-16)$$

$$\text{sen}\beta = \frac{u_x}{\sqrt{u_x^2 + u_y^2}} \qquad (A-17)$$

$$\cos\beta = \frac{u_z}{\sqrt{u_x^2 + u_y^2}} \qquad (A-18)$$

con lo que

$$\cos\gamma\ \text{sen}\beta = u_x \qquad (A-19)$$

$$\cos\gamma\ \cos\beta = u_z \qquad (A-20)$$

y operando en la expresión A-14 tenemos:

$$\underset{\sim}{AJ} = \begin{bmatrix} hC\theta+u_x^2V^*\theta & u_xu_yV^*\theta-u_zhS\theta & u_xu_zV^*\theta+hu_yS\theta \\ u_xu_yV^*\theta+hu_zS\theta & u_y^2V^*\theta+hC\theta & u_yu_zV^*\theta-hu_xS\theta \\ u_xu_zV^*\theta-hu_yS\theta & u_zu_yV^*\theta+hu_xS\theta & u_z^2V^*\theta+hC\theta \end{bmatrix} \underset{\sim}{A1}$$

(A-21)

donde $C\theta = \cos\theta$; $S\theta = \text{sen}\ \theta$; y $V^*\theta = \frac{1}{h} - h\cos\theta$.

que sera la matriz de rotación-extensión en el espacio alredededor de un eje $\underset{\sim}{u}$ y que se representara simbolicamente en la forma:

$$\left[RL\ (h,\theta\ ,u)\right] \qquad \text{(A-22)}$$

APENDICE B

B.- APROXIMACION CON SPLINE CUBICOS.

La expresión analítica de una curva plana o espacial determinada no siempre es conocida. En muchos casos solo se dispone de algunos puntos por los que habrá de pasar la curva, lo cual hace necesario la interpolación de una función de manera que ajuste la curva de una forma aceptable. Es por ello, que se tratará de obtener una función P(t), que pase por unos puntos determinados de la función o curva que se desea aproximar f(t), y que cumpla, en su caso, una serie de condiciones en esos puntos.

Como funciones de interpolación se usan con mucha frecuencia las funciones spline, debido fundamentalmente, a su facilidad de manejo así como al elevado grado de continuidad que proporcionan. El concepto matemático del spline deriva del hecho físico del spline. Una spline es una pieza larga y flexible de plastico, metal u otro material, usada por los delineantes para dibujar curvas que pasen por unos puntos determinados, obteniendo la curva de mínima energía, es decir la más suave.

Matemáticamente y de forma general un spline es un polinomio segmentado de grado K, con continuidad de grado K-1 en los puntos comunes a los segmentos. Así un spline cúbico tiene continuidad hasta la segunda derivada en los puntos comunes. El uso frecuente para aproximar curvas, de los polinomios segmentados con un bajo grado en los polinomios, viene motivado por las facilidades y reducciones en el cálculo así como la disminución de las inestabilidades inherentes o que se puedan producir con curvas de mayor or-

den. Sin embargo, hay que tener en cuenta que los polinomios de un grado bajo no pueden pasar por un número arbitrario de puntos, por lo que será necesario la segmentación del polinomio.

De acuerdo con todo esto una técnica muy común es el uso de series de splines cúbicos en los que cada segmento pasa por dos puntos. Además los splines cúbicos tienen la ventaja de ser las curvas espaciales de menor grado que permiten puntos de inflexión.

La ecuación para un segmento simple de un spline cúbico en términos de un parametro t viene dada por:

$$P(t) = \sum_{i=1}^{4} B_i \, t^{i-1} \; ; \quad t_1 \leq t \leq t_2 \qquad \text{(B-1)}$$

donde $P(t)$ es un vector cuyas tres componentes son: $x(t)$, $y(t)$, $z(t)$. A $P(t)$ se le puede considerar como el vector posición de un punto del spline. Las tres componentes $x(t)$, $y(t)$, $z(t)$, serán las coordenadas cartesianas del vector de posición. Los coeficientes B_i se determinan imponiendo cuatro condiciones de contorno para el segmento de spline.- La expresión anterior puede desarrollarse de la forma:

$$P(t) = B_1 + B_2 t + B_3 \, t^2 + B_4 \, t^3 \qquad \text{(B-2)}$$

de acuerdo con esto, para un par de puntos, definidos por un vector de posición P_1 y P_2, por las que habrá de pasar el segmento de spline, los vectores tangente en esos

puntos estarán indicados por P_1' y P_2', es decir las derivadas con respecto al parametro t de los vectores de posición. A lo largo del segmento de spline cúbico el parametro t varia entre t_1 y t_2. Para simplificación de los cálculos se le asigna a t_1 el valor cero.

Las condiciones de contorno necesarias para cada segmento de spline cúbico serán, los vectores de posición de los extremos y los vectores tangente en cada extremo. Para un segmento simple de spline cúbico entre P_1 y P_2 estas condiciones de contorno serán:

$$P(0) = P_1 \tag{B-3}$$

$$P(t_2) = P_2 \tag{B-4}$$

$$\left.\frac{dP}{dt}\right|_{t=0} = P_1' \tag{B-5}$$

$$\left.\frac{dP}{dt}\right|_{t=t_2} = P_2' \tag{B-6}$$

que desarrollando de acuerdo con las expresiones B-1 y B-2 será:

$$P(0) = B_1 = P_1 \tag{B-7}$$

$$\left.\frac{dP}{dt}\right|_{t=0} = \sum_{i=2}^{4} (i-1)\, B_i\, t^{i-2}\Big|_{t=0} = B_2 = P_1' \tag{B-8}$$

$$P(t_2) = \sum_{i=1}^{4} B_1 t^{i-1}\Big|_{t=t_2} = B_1+B_2t_2+B_3t_2^2+B_4t_2^3 = P_2 \qquad \text{(B-9)}$$

$$\frac{dP}{dt}\Big|_{t=t_2} = \sum_{i=1}^{4} (i-1)B_i t^{i-2}\Big|_{t=t_2} = B_2+2B_3t_2+3B_4t_2^2 = P_2' \qquad \text{(B-10)}$$

resumiendo será:

$$B_1 = P_1 \qquad \text{(B-11)}$$

$$B_2 = P_1' \qquad \text{(B-12)}$$

$$B_1 + B_2t_2+B_3t_2^2+B_4t_2^3 = P_2 \qquad \text{(B-13)}$$

$$B_2 + 2B_3t_2+ 3B_4t_2^2 = P_2' \qquad \text{(B-14)}$$

que resolviendo para B_3 y B_4 dará:

$$B_3 = \frac{3(P_2-P_1)}{t_2^2} - \frac{2P_1'}{t_2} - \frac{P_2'}{t_2} \qquad \text{(B-15)}$$

$$B_4 = \frac{2(P_1-P_2)}{t_2^3} + \frac{P_1'}{t_2^2} + \frac{P_2'}{t_2^2} \qquad \text{(B-16)}$$

Conocidos los valores de B_1, B_2, B_3 y B_4 estará definido el segmento de spline cúbico deseado. Susti

tuyendo en la ecuación (B-2) los valores de los B_i se obtiene:

$$P(t) = P_1 + P_1' t + \left[\frac{3(P_2 - P_1)}{t_2^2} - \frac{2P_1'}{t_2} - \frac{P_2'}{t_2} \right] t^2 + \left[\frac{2(P_1 - P_2)}{t_2^3} + \frac{P_1'}{t_2^2} + \frac{P_2'}{t_2^2} \right] t^3 \qquad \text{(B-17)}$$

en esta expresión puede verse, que la forma del segmento de spline cúbico depende de los vectores de posición de los extremos y de los vectores tangentes en dichos puntos. Además el valor de parámetro t en el extremo del segmento -- (t_2) aparece también en la expresión (B-17).

La ecuación anterior (B-17) que es para un segmento simple de spline cúbico puede generalizarse para dos segmentos consecutivos $P_k(t)$ y $P_{k+1}(t)$; $1 \leq k \leq n-2$, donde n es el número de puntos por los que habrá de pasar la curva, de la forma:

$$P_k(t) = P_k + P_k' t + \left[\frac{3(P_{k+1} - P_k)}{t_2^2} - \frac{2P_k'}{t_2} - \frac{P_{k+1}'}{t_2} \right] t^2 + \left[\frac{2(P_k - P_{k+1})}{t_2^3} + \frac{P_k'}{t_2^2} + \frac{P_{k+1}'}{t_2^2} \right] t^3 \qquad \text{(B-18)}$$

y

$$P_{k+1}(t) = P_{k+1} + P'_{k+1}t + \left[\frac{3(P_{k+2}-P_{k+1})}{t_3^2} - \frac{2P'_{k+1}}{t_3} - \frac{P'_{k+2}}{t_3} \right] t^2 +$$

$$+ \left[\frac{2(P_{k+1}-P_{k+2})}{t_3^3} + \frac{P'_{k+1}}{t_3^2} + \frac{P'_{k+2}}{t_3^2} \right] t^3 \quad \text{(B-19)}$$

En las expresiones anteriores se supone que la variación del parametro es $0 \leq t \leq t_2$ para el primer segmento y $0 \leq t \leq t_3$ para el segundo segmento.

Por ejemplo:si se desea una curva que pase por tres puntos se especificaran tres vectores de posición P_1, P_2, P_3 y los valores tangente en los extremos de la curva P'_1 y P'_3. Para asegurar la continuidad de segundo orden para el spline cúbico se impone la condición de que la curvatura sea constante en el punto interno. Esto supone que la segunda derivada $P''(t)$ debe ser continua en el punto interno.

De la ecuación (B-1) se puede poner

$$P''(t) = \sum_{i=1}^{4} (i-1)(i-2)B_i t^{i-3} \quad ; \quad t_1 \leq t \leq t_2 \quad \text{(B-20)}$$

Que particularizada para el valor de $t = t_2$ que es el del

extremo final del primer segmento del spline cúbico dará

$$P'' = 6\ B_4\ t_2 + 2\ B_3 \qquad \text{(B-21)}$$

y si se particulariza para $t = o$, que es el valor de t en el extremo inicial del segundo segmento del spline cúbico será

$$P'' = 2\ B_3 \qquad \text{(B-22)}$$

igualando las expresiones (B-21) y (B-22) y teniendo en cuenta los valores de B_3 y B_4 expresados por (B-15) y (B-16) se obtendrá

$$6\ t_2 \left[\frac{2(P_1-P_2)}{t_2^3} + \frac{P_1'}{t_2^2} + \frac{P_2'}{t_2^2} \right] + 2 \left[\frac{3(P_2-P_1)}{t_2^2} - \frac{2P_1'}{t_2} - \frac{P_2'}{t_2} \right] =$$

$$= 2 \left[\frac{3(P_3-P_2)}{t_3^2} - \frac{2P_2'}{t_3} - \frac{P_3'}{t_3} \right] \qquad \text{(B-23)}$$

Multiplicando por $t_2\ t_3$ y reordenando términos se tendrá:

$$t_3 P_1' + 2(t_3+t_2)P_2' + t_2 P_3' = \frac{3}{t_3\ t_2} \left[t_2^2\ (P_3-P_2) + t_3^2 (P_2-P_1) \right] \qquad \text{(B-24)}$$

ecuación en la que hay como única incognita P_2' , vector - tangente en el punto interior, que se podrá obtener facilmente.

Para n puntos de pasc los resultados se pueden generalizar a n-1 segmentos de spline cúbico con continuidad en los puntos de paso internos hasta la segunda derivada. La condición de continuidad de la segunda derivada de forma generalizada será:

$$t_{k+2}\,P_k' + 2(t_{k+2} + t_{k+1})P_{k+1}' + t_{k+1}\,P_{k+2}' =$$

$$= \frac{3}{t_{k+1}\,t_{k+2}}\left[t_{k+1}^2\,(P_{k+2} - P_{k+1}) + t_{k+2}^2\,(P_{k+1} - P_k)\right];$$

$$1 \leq k \leq n-2 \qquad \text{(B-25)}$$

Esta ecuación se aplicará sucesivamente a todos los segmentos. Escribiendo en forma matricial la (B-25) correspondiente a todos los segmentos del spline cúbico se tendrá

$$\begin{bmatrix} t_3 & 2(t_2+t_3) & t_2 & 0 & & \\ 0 & t_4 & 2(t_3+t_4) & t_3 & 0 & \\ & & t_5 & 2\,t_4+t_5 & t_4 & \\ & & & \cdot & \cdot & \cdot \\ & & & & \cdot & \cdot \end{bmatrix} \begin{Bmatrix} P_1' \\ P_2' \\ P_3' \\ \vdots \\ P_n' \end{Bmatrix} =$$

$$= \begin{bmatrix} \frac{3}{t_2\ t_3} \left[t_2^2\ (P_3-P_2) + t_3^2(P_2-P_1)\right] \\ \frac{3}{t_3\ t_4} \left[t_3^2\ (P_4-P_3) + t_4^2(P_3-P_2)\right] \\ \vdots \\ \frac{3}{t_{n-1}t_n} \left[t_{n-1}^2(P_n-P_{n-1}) + t_n^2(P_{n-1}-P_{n-2})\right] \end{bmatrix} \qquad \text{(B-26)}$$

La expresión anterior representa un sistema de n-2 ecuaciones con n incognitas que son los vectores tangente. Si los vectores tangente de los extremos P_1' y P_2' son conocidos el sistema de ecuaciones estará determinado.- Así para interpolar una curva habrá que especificar los vectores de posición de los puntos por los que tendrá que pasar y los vectores tangente en los puntos extremos. De esta forma el sistema de ecuaciones representado por la ecuación matricial (B-26) se usará para determinar los vectores tangente P_2', P_3' ..... P_{n-1}' de los puntos intermedios.

Una vez conocidos los vectores tangente tanto en los puntos intermedios o internos como en los extremos, se usaran para determinar los valores de los coeficientes B_i, de acuerdo con las ecuaciones (B-11) a (B-14) que generalizadas para un segmento cualquiera son:

$$B_1 = P_k \qquad \text{(B-27)}$$

$$B_2 = P_k' \qquad \text{(B-28)}$$

$$B_3 = \frac{3(P_{k+1}-P_k)}{t_{k+1}^2} - \frac{2P_k'}{t_{k+1}} - \frac{P_{k+1}'}{t_{k+1}} \qquad \text{(B-29)}$$

$$B_4 = \frac{2(P_k - P_{k+1})}{t_{k+1}^3} + \frac{P'_k}{t_{k+1}^2} + \frac{P'_{k+1}}{t_{k+1}^2} \qquad \text{(B-30)}$$

Finalmente se genera cada segmento del spline cúbico haciendo uso de la ecuación (B-1) con $0 \leq t \leq t_{max}$. Antes de generar la curva será necesario establecer el valor máximo de parametro t para cada tramo o segmento.

Para determinar el valor de t_{max} se pueden utilizar métodos relativamente complejos, que en la mayoría de los casos no compensan el esfuerzo adicional que suponen para la mejora en la aproximación que se obtiene. Una forma simple, que permite obtener el valor de t_{max} y conseguir unos buenos resultados, es hacer que el valor máximo del parametro (t_{max}) sea igual a la longitud de la cuerda entre los sucesivos puntos. Otra posibilidad es normalizar la variación de t tomando $t_{max} = 1.0$ para cada segmento de spline cúbico. Se obtienen mejores resultados con la primera forma indicada, que con la segunda.

La ecuación (B-26) se puede escribir de la forma:

$$\underset{\sim}{M}^{*} \cdot \underset{\sim}{P'} = \underset{\sim}{B} \qquad \text{(B-31)}$$

donde

$\underset{\sim}{M}^{*}$ es la matriz de n-2 x n

$\underset{\sim}{P'}$ " " " " n x 1

$\underset{\sim}{D}$ " " " " n-2 x 1

los términos no nulos de cada fila de la matriz M^* son:

$$M^*(J,J-1);\quad M^*(J,J)\quad y\quad M^*(J,J+1);2 \leq J \leq n-1$$

es decir que la ecuación (B-31) se puede poner de la forma

$$\begin{bmatrix} M^*(2,1) & M^*(2.2)..... & & \\ & M^*(3.2) & M^*(3.3).... & \\ & \vdots & \vdots & \\ & & M^*(n-1,n-1) & M^*(n-1,n) \end{bmatrix} \begin{bmatrix} P'(1) \\ P'(2) \\ \vdots \\ P'(n) \end{bmatrix} \begin{bmatrix} D(2) \\ D(3) \\ \vdots \\ D(n-1 \end{bmatrix} \qquad \text{(B-32)}$$

Será necesario que la matriz $\underset{\sim}{M}^*$ sea una matriz cuadrada $\underset{\sim}{M}$ de forma que se pueda obtener una única solución para los vectores tangente incognita. Esta matriz $\underset{\sim}{M}$ puede conseguirse si se especifican las condiciones de contorno del spline cúbico total en el extremo inicial y final, como se verá a continuación. En este caso la ecuación (B-31) se puede poner como:

$$\underset{\sim}{M} \cdot \underset{\sim}{P}' = \underset{\sim}{D} \qquad \text{(B-33)}$$

y por tanto los vectores tangentes internos vendran dados por:

$$\underset{\sim}{P}' = \underset{\sim}{M}^{-1} \cdot \underset{\sim}{D} \qquad \text{(B-34)}$$

donde $\underset{\sim}{M}^{-1}$ es la matriz inversa de M.

Existe muchas alternativas para especificar las condiciones de contorno de los extremos del spline cúbico total. La solución más directa será la que se obtiene especificando los valores de los vectores tangente de los extre -

mos P'_1 y P_n. Esta condición de contorno es conocida como de extremos fijos. La ecuación matricial para esa condición de contorno será:

$$\begin{bmatrix} 1 & & & \\ M(2,1) & M(2,2) & \cdots\cdots\cdots\cdots & \\ & M(3,2) & M(3.3) & \cdots\cdots\cdots \\ & \vdots & & \\ & & \cdots M(n-1,n-1) & M(n-1,n) \\ & & & 1 \end{bmatrix} \begin{bmatrix} P'(1) \\ P'(2) \\ \vdots \\ P'(n) \end{bmatrix} = \begin{bmatrix} D(1) \\ D(2) \\ D(3) \\ \vdots \\ D(n) \end{bmatrix} \qquad \text{(B-35)}$$

como puede verse esta ecuación indica que $P'_1 = D(1)$ y que $P'(n) = D(n)$. Para esta condición de contorno la matriz $\underset{\sim}{M}$ es una matriz tridiagonal lo cual facilita la resolución del sistema de ecuaciones (B-35).

Otra condición de contorno en el extremo puede ser:

$$\frac{d^2 P}{d^2 t} = 0 \qquad \text{(B-36)}$$

esta condición es conocida como de extremo natural. La expresión (B-36) de acuerdo con la (B-20) para el primer segmento y $t=0$ dará:

$$P'_1 + \frac{1}{2} P'_2 = \frac{3}{2} \frac{P_2 - P_1}{t_2} \qquad \text{(B-37)}$$

y para el último segmento y $t=t_n$ será:

$$2P'_{n-1} + 4P'_n = \frac{6}{t_n} (P_n - P_{n-1}) \qquad \text{(B-38)}$$

de la (B-37) se obtiene que los términos, no nulos, de la primera fila de la matriz $\underset{\sim}{M}$ son M(1,1)= 1.0 y M(1,2)= 0.5, así como de la (B-38) se obtiene que M(n,n-1) = 2.0. y M(n,n) = 4.0. En la matriz $\underset{\sim}{D}$ el término D(1) será igual al de la derecha de la (B-37) y el D(n) será igual al de (B-38). Esta condición de contorno también da una matriz M tridiagonal.

Cuando se pretende aproximar curvas cerradas o que se repiten ciclícamente, se usa la condición de contorno de extremo ciclico, que consiste en hacer que

$$P'_1 = P'_n \qquad \text{(B-39)}$$

$$P''_1 = P''_n \qquad \text{(B-40)}$$

es decir que la pendiente y la curvatura al principio y al final sean iguales.

La expresión (B-39) de acuerdo con las (B-14) y (B-27) a (B-30) puede ponerse de la forma:

$$P'_n - P'_{n-1} = 2\left[\frac{3(P_n - P_{n-1})}{t_n^2} - \frac{2P'_{n-1}}{t_n} - \frac{P'_n}{t_n}\right] t_n +$$

$$+ 3\left[\frac{2(P_{n-1} - P_n)}{t_n^3} + \frac{P'_{n-1}}{t_n^2} + \frac{P'_n}{t_n^2}\right] t_n \qquad \text{(B-41)}$$

de igual manera la (B-40) será:

$$2\left[\frac{3(P_2-P_1)}{t_2^2}-\frac{2P'_1}{t_2}-\frac{P'_2}{t_2}\right]=2\left[\frac{3(P_n-P_{n-1})}{t_n^2}-\frac{2P'_{n-1}}{t_n}-\frac{P'_n}{t_n}\right]+$$

$$+6\left[\frac{2(P_{n-1}-P_n)}{t_n^3}+\frac{P'_{n-1}}{t_n^2}+\frac{P'_n}{t_n^2}\right]t_n \qquad \text{(B-42)}$$

Estas dos ecuaciones (B-41) y (B-42) pueden relacionarse de forma que se obtenga una sola ecuación que combinada con la matriz $\underset{\sim}{M}$ dará una matriz de $(n-1)x(n-1)$. El orden de la matriz se reduce debido a que las condiciones de pendiente y curvatura impuestas no son independientes. Sólo hay que determinar n-1 vectores tangente.

Si se multiplica la ecuación (B-42) por t_n y se resta a la ecuación (B-41) se tendrá:

$$P'_1-P'_{n-1}-2\left[\frac{3(P_2-P_1)}{t_2^2}-\frac{2P'_1}{t_2}-\frac{P'_2}{t_2}\right]t_n=$$

$$=3\left[\frac{2(P_{n-1}-P_n)}{t_n^3}+\frac{P_{n-1}}{t_n^2}+\frac{P'_n}{t_n^2}\right]t_n^2-$$

$$6\left[\frac{2(P_{n-1}-P_n)}{t_n^3}+\frac{P'_{n-1}}{t_n^2}+\frac{P'_n}{t_n^2}\right]t_n^2 \qquad \text{(B-43)}$$

como $P'_1 = P'_n$ y reordenando términos

$$2\left(1+\frac{t_n}{t_2}\right)P'_1 + P'_2\frac{t_n}{t_2} + P'_{n-1} = 3(P_2-P_1)\frac{t_n}{t_2^2} - 3(P_{n-1}-P_n)\frac{1}{t_n} \qquad (B-44)$$

de acuerdo con esto la ecuación matricial será de la forma

$$\begin{bmatrix} 2\left(1+\frac{t_n}{t_2}\right) & \frac{t_n}{t_2} & 0 \ \cdots\cdots\cdots\cdots 1 \\ M(2,1) & M(2,2) & M(2,3) \\ & M(3,2) & M(3,3) \\ & & \\ & & M(n-1,n-1) \end{bmatrix} \begin{bmatrix} P'(1) \\ P'(2) \\ \vdots \\ P'(n-1) \end{bmatrix} = \begin{bmatrix} D(1) \\ D(2) \\ \vdots \\ D(n-1) \end{bmatrix} \qquad (B-45)$$

que se puede resolver con cualquier método de resolución - de sistemas de ecuaciones como puede ser el de eliminación Gauss.

Hay otra condición de contorno que es básicamente la anterior con la particularidad de tener signos cambiados es decir que las ecuaciones (B-39) y (B-40) sean de la forma:

$$P'_1 = - P'_n \qquad (B-46)$$

$$P''_1 = - P''_n \qquad (B-47)$$

esta condición de contorno es conocida como extremo anticí clico. Se desarrolla de forma análoga a lo expuesto para el extremo cíclico y se llega a un sistema similar que se resolverá de manera análoga.

Se podrán plantear muchas más condiciones de - contorno de extremo, de acuerdo a las restricciones y particularidades de cada caso.

A P E N D I C E C

```
PAGE 0001  FTN.  11:25 PM  FRI., 15  JAN., 1982


0001   FTN4
0002          BLOCK DATA OPTRZ
0003   C
0004   C      DEFINICION DE LOS COMMON NECESARIOS PARA EL PROGRAMA OPRIS
0005   C
0006          COMMON/COMU/LECT,IMPR,TITU(30),NCVE,ZZ(30),NVE,ICOMM
0007          COMMON/NEWT/TEMF(30,20),IKOUNT,TOLER,DISCR,ANORMI,ESCR,ZMA(30,20
0008          COMMON/OPMV/VMIFO,ERRAB,NMITE,DINC1
0009          COMMON/OPGC/VMIF2,ERRAB2,NMTR2,DINC2
0010          COMMON/OPBD/ITES,LMITE,VIMPA,VMIPA
0011          COMMON/OPDC/INESC,NLITE,VMAPA,VLCV(30)
0012          COMMON/OPCOM/NS,KSF,IXX,ITMAX,ICS,IPRT,ALFHA,BETA,GAMMA,DELTA
0013          COMMON/DATOS/DATOP(2,31),DATOT(30),IB,IC,ID,IEL,ISL(80),ZP(80),
0014         *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IFEN
0015          COMMON/LOGIC/COVER,LIM,NSRE,PRINT,NITP,ITNS,NCORE,SUMI,IMPME,IMPV
0016   C
0017   C      PARA AUMENTAR LA MEMORIA DINAMICA CAMBIAR LA TARJETA SIGUIENTE PO
0018   C      NIENDO LA DIMENSION DEL COMMON IW AL TAMA#O DESEADO.CAMBIAR TAM-
0019   C      BIEN EL DATA ICOMM AL MISMO VALOR.
0020   C
0021          COMMON//IW(10000)
0022          LOGICAL COVER,LIM,NSRE,PRINT,IMPME,IMPMA
0023          INTEGER GAMMA
0024          DATA ICOMM/10000/
0025          END

 FTN4 COMPILER) HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **

BLOCK COMMON COMU    SIZE =  00125

BLOCK COMMON NEWT    SIZE =  02409

BLOCK COMMON OPMV    SIZE =  00007

BLOCK COMMON OPGC    SIZE =  00007

BLOCK COMMON OPBD    SIZE =  00006

BLOCK COMMON OPDC    SIZE =  00064

BLOCK COMMON OPCOM   SIZE =  00013

BLOCK COMMON DATOS   SIZE =  00438

BLOCK COMMON LOGIC   SIZE =  00011
```

```
PAGE 0001  FTN.  11:10 PM  FRI.,  15  JAN.,  1982

0001   FTN4
0002         PROGRAM OPRIS
0003   C
0004   C     PROGRAMA PARA LA OBTENCION DE UNA PRIMERA SOLUCION PARA LA SINTE
0005   C     SIS OPTIMA DE MECANISMOS.REALIZA LA SINTESIS CON PUNTOS DE PRECI-
0006   C     SION RESOLVIENDO EL SISTEMA DE ECUACIONES MEDIANTE DIVERSOS METO-
0007   C     DOS.GENERA ALEATORIAMENTE LA FAMILIA DE VECTORES DE PARTIDA.SU C
0008   C     PACIDAD ACTUAL ESTA EN 30 VARIABLES O COMPONENTES DEL VECTOR DE
0009   C     DISEÑO.
0010   C
0011         COMMON/COMU/LECT,IMPR,TITU(30),NCVE,ZZ(30),NVE,ICOMM
0012         COMMON/NEWT/TEMP(30,20),ICONT,TOLER,DINOR,ANORMI,ESCR,ZMA(30,20)
0013         COMMON/OPMV/VMIFO,ERRAB,NMITE,DINC1
0014         COMMON/OPGC/VMIF2,ERRAB2,NMIT2,DINC2
0015         COMMON/OPBD/ITES,LMITE,VIMPA,VNIPA
0016         COMMON/OPDC/INESC,NLITE,VMAPA,VLCV(30)
0017         COMMON/OPCOM/NS,KSF,IXX,ITMAX,ICS,IPRI,ALFHA,BETA,GAMMA,DELTA
0018         COMMON/DATOS/DATOP(2,31),DATOT(30),IB,IC,ID,IEL,ISL(80),ZP(80),
0019        *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,ISEN
0020         COMMON/LOGIC/COVER,LIM,NSRE,PRINT,NITR,ITNS,NCORE,SUMI,IMPRE,IMPV
0021         COMMON//IW(1)
0022         LOGICAL COVER,LIM,NSRE,PRINT,IMPME,IMPMA
0023         INTEGER GAMMA
0024         DIMENSION IPAR(5),IOP(5),IP1(3),IP2(3),IP3(3),IP4(3),IP5(3),IP6(3
0025         DATA IP1/2HOP,2HTI,2H1 /,IP2/2HOP,2HTI,2H2 /,IP3/2HOP,2HTI,2H3 /,
0026        *IP4/2HOP,2HTI,2H4 /,IP5/2HOP,2HTI,2H5 /,IP6/2HOP,2HTI,2H6 /
0027   C
0028   C     RECOGIDA DE LOS PARAMETROS DE ENTRADA QUE DEFINEN LAS UNIDADES DE
0029   C     LECTURA Y ESCRITURA ASI COMO DEL NUMERO DE CASOS A RESOLVER
0030   C
0031         CALL RMPAR(IPAR)
0032         LECT=IPAR(1)
0033         IMPR=IPAR(2)
0034         LS=IPAR(3)
0035         LK=IPAR(4)
0036   C
0037   C     COMPRUEBA SI HAY ALGUN CASO MAS POR RESOLVER
0038   C
0039     100 IF(LK.EQ.LS)STOP
0040         LK=LK+1
0041         DO 25 JT=1,20
0042         DO 25 JTT=1,30
0043         ZMA(JTT,JT)=0.0
0044      25 TEMP(JTT,JT)=0.0
0045   C
0046   C     LECTURA DE LOS DATOS  Y PARAMETROS DE EJECUCION
0047   C
0048         CALL LEER(IOP,IMMOD,INCI,IX,NREST)
0049   C
0050   C     COMPRUEBA SI HAY RESTRICCIONES
0051   C
0052         IF(NREST.NE.0)GOTO 101
0053   C
0054   C     REALIZA LA EJECUCION DE LAS DIVERSAS OPCIONES DE RESOLUCION DE
0055   C     ACUERDO CON LOS PARAMETROS LEIDOS
```

```
PAGE 0002  OPRIS 11:10 PM  FRI., 15  JAN., 1982

0056   C
0057          IF(IOP(1).EQ.0)GOTO 1
0058   C
0059   C       LLAMADA AL PRIMER SEGMENTO.NEWTON-RAPHSON
0060   C
0061          CALL OVLAY(IP1)
0062          IF(ICONT.EQ.0)GOTO 2
0063       1  IF(INMOD)2,3,3
0064       3  IF(ICONT.EQ.0)GOTO 200
0065          INCI=MIN0(INCI,ICONT)
0066     200  IF(IOP(1).EQ.0)GOTO 2
0067          DO 5 IS=1,ICONT
0068          DO 5 ISN=1,NCVE
0069          ZMA(ISN,IS)=TEMP(ISN,IS)
0070        5 CONTINUE
0071        2 DO 10 I=1,INCI
0072          DO 20 IL=1,NCVE
0073          ZZ(IL)=ZMA(IL,I)
0074       20 CONTINUE
0075          IF(IOP(2).EQ.0)GOTO 6
0076   C
0077   C       LLAMADA AL SEGUNDO SEGMENTO.METRICA VARIABLE
0078   C
0079          CALL OVLAY(IP2)
0080        6 IF(INMOD.EQ.1)GOTO 7
0081          DO 30 IL=1,NCVE
0082          ZZ(IL)=ZMA(IL,I)
0083       30 CONTINUE
0084        7 IF(IOP(3).EQ.0)GOTO 8
0085   C
0086   C       LLAMADA AL TERCER SEGMENTO.GRADIENTE CONJUGADO
0087   C
0088          CALL OVLAY(IP3)
0089        8 IF(INMOD.EQ.1)GOTO 9
0090          DO 40 IL=1,NCVE
0091          ZZ(IL)=ZMA(IL,I)
0092       40 CONTINUE
0093        9 IF(IOP(4).EQ.0)GOTO 11
0094   C
0095   C       LLAMADA AL CUARTO SEGMENTO.BUSQUEDA DIRECTA
0096   C
0097          CALL OVLAY(IP4)
0098       11 IF(INMOD.EQ.1)GOTO 12
0099          DO 50 IL=1,NCVE
0100          ZZ(IL)=ZMA(IL,I)
0101       50 CONTINUE
0102       12 IF(IOP(5).EQ.0)GOTO 10
0103   C
0104   C       LLAMADA AL QUINTO SEGMENTO.DIRECCIONES CONJUGADAS
0105   C
0106          CALL OVLAY(IP5)
0107       10 CONTINUE
0108          GOTO 102
0109   C
0110   C     LLAMADA AL SEXTO SEGMENTO.COMPLEX
```

```
PAGE 0003  OPRIS 11:10 PM  FRI., 15  JAN., 1982


 1:1   C
 1:2    101  CALL OVLAY(IP6)
 1:3     102 GOTO 100
 1:4         END


 FTN4 COMPILER) HP9206)-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00405       COMMON = 00001
```

```
PAGE 0004  FTN.  11:10 PM  FRI., 15  JAN., 1982

      SUBROUTINE LEER(IOP,INMOD,INCI,IX,NREST)
C
C     LECTURA DE TODOS LOS DATOS RELATIVOS AL CASO A RESOLVER Y DE LOS
C     PARAMETROS DE EJECUCION
C
      COMMON/COMU/LECT,IMPR,TITU(30),NCVE,ZZ(30),NVE,ICOMM
      COMMON/NEWT/TEMP(30,20),ICONT,TOLER,DINCR,ANORMI,ESCR,ZMA(30,20)
      COMMON/OPMV/VNIF0,ERRAB,NMITE,DINC1
      COMMON/OPGC/VNIF2,ERRA2,NMIT2,DINC2
      COMMON/OPBD/ITES,LMITE,VINPA,VNIPA
      COMMON/OPDC/INESC,NLITE,VMAPA,VLCV(30)
      COMMON/60X/NS,KSF,IXX,ITMAX,ICS,IPRI,ALPHA,BETA,GAMMA,DELTA
      COMMON/DATOS/DATOP(2,31),DATOT(30),IB,IC,ID,IEL,ISL(80),ZP(80),
     *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IBEN
      COMMON/LOGIC/COVER,LIM,NSRE,PRINT,NITR,ITNS,NCORE,SUMI,IMPME,IMP
      COMMON/Z/IW(1)
      LOGICAL COVER,LIM,NSRE,PRINT,IMPME,IMPNA
      INTEGER GAMMA
      DIMENSION IOP(5),ZMIN(30),ZMAX(30),IHORA(15)
      READ(LECT,1)TITU
    1 FORMAT(30A2)
      READ(LECT,*)NCVE,MALEC,INCI,NVE,IX,NREST
      READ(LECT,*)INMOD,(IOP(NS),NS=1,5)
      READ(LECT,*)NITR,TOLER,DINCR,ANORMI,ESCR
      READ(LECT,*)VNIF0,ERRAB,NMITE,DINC1
      READ(LECT,*)VNIF2,ERRAB2,NMIT2,DINC2
      READ(LECT,*)ITES,LMITE,VINPA,VNIPA
      READ(LECT,*)INESC,NLITE,VMAPA
      READ(LECT,*)(VLCV(JJ),JJ=1,NCVE)
      CALL FTIME(IHORA)
C
C     ESCRITURA DEL ENCABEZADO Y DATOS DE EJECUCION DE PROBLEMA
C
      WRITE(IMPR,200)IHORA,NCVE,MALEC,INCI,NVE,IX,NREST,INMOD,IOP
  200 FORMAT(1H1,15/,1X,79(1H*)//,1X,5(1H*),6X," PROGRAMA OPRIS.(OPTIMI
     *ACION.PRIMERA SOLUCION) ",6X,5(1H*),/,1X,5(1H*),15X,15A2,15X,
     *5(1H*)//,1X,79(1H*)/,5X,62(1H*),//,5
     *X,"DIMENSION DEL V.E. = ",I5,//,5X,"SOLUCION DE PARTIDA = ",I5,//,5
     *,"V.E A ENSAYAR = ",I5,//,5X,"N. DE V.E. = ",I5,//,5X,
     *"V.I. ALEATORIO = ",I5,//,5X,"N.DE RESTRICCIONES = ",I5,//,5X,"MODO
     *DE OPERACION = ",I5,//,5X,"SEGMENTOS QUE USA : ",5I5,//,5X,62(1H*))
      IF(NREST.NE.0)GOTO 125
C
C     LECTURA DE LOS DATOS ESPECIFICOS DEL SISTEMA DE ECUACIONES A RE-
C     SOLVER Y DE LA TRAYECTORIA A GENERAR.
C
  105 CALL DATS(NCVE,LECT)
      IF(NREST.NE.0)GOTO 100
      IF(MALEC.NE.0)GOTO 4
C
C     LECTURA DE LOS LIMITES PARA LA GENERACION ALEATORIA
C
      DO 2 I=1,NCVE
      READ(LECT,*)ZMIN(I),ZMAX(I)
    2 CONTINUE
```

```
PAGE 0005  LEER  11:10 PM  FRI., 15  JAN., 1982


0170   C
0171   C      GENERACION ALEATORIA DE LA FAMILIA DE VECTORES DE PARTIDA
0172   C
0173           DO 3 K=1,NVE
0174           DO 3 KK=1,NCVE
0175           CALL ALEAT(IX,IY,RA)
0176           ZMA(KK,K)=ZMIN(KK)+(ZMAX(KK)-ZMIN(KK))*RA
0177           IX=IY
0178      3    CONTINUE
0179           GOTO 100
0180   C
0181   C      LECTURA DIRECTA DE LA FAMILIA DE VECTORES
0182   C
0183      4    DO 5 KK=1,NVE
0184           DO 5 K=1,NCVE
0185           READ(LECT,*)ZMA(K,KK)
0186      5    CONTINUE
0187           GOTO 100
0188   C
0189   C      LECTURA DE LOS DATOS CUANDO HAY RESTRICCIONES
0190   C
0191      125  READ(LECT,*)MS,KSF,IXX
0192           READ(LECT,*)ITMAX,ICS,IPRI,ALFHA
0193           READ(LECT,*)BETA,GAMMA,DELTA
0194           READ(LECT,*)(ZZ(NSF),NSF=1,NCVE)
0195           GOTO 105
0196      100  RETURN
0197           END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **    PROGRAM = 00733       COMMON = 00001
```

```
0001   FTN4
0002         PROGRAM OPTI1(5)
0003   C
0004   C     PRIMER SEGMENTO DEL PROGRAMA OPRIS.REALIZA LA RESOLUCION DE
0005   C     UN SISTEMA DE ECUACIONE ALGEBRAICAS NO LINEALES MEDIANTE EL
0006   C     METODO DE NEWTON-RAPHSON.
0007   C
0008         COMMON/COMU/LECT,IMPR,TITU(30),NCVE,ZZ(30),NVE,ICOMM
0009         COMMON/NEWT/TEMP(30,20),ICONTA,TOLER,DIMCR,ANORMI,ESCR,ZMAX(30,20
0010         COMMON/LOGIC/COVER,MLI,NSRE,PRINT,NITP,ITNS,NCORE,SUMI,IMPME,IMP
0011         COMMON//IW(1)
0012         LOGICAL COVER,MLI,NSRE,PRINT,IMPME,IMPH
0013         DO 10 I=1,ICOMM
0014      10 IW(I)=0
0015   C
0016   C     CALCULO DE LOS INDICES DE LA MEMORIA DINAMICA
0017   C
0018         N2=1+2*NCVE
0019         N3=N2+2*NCVE*NCVE
0020         N4=N3+2*NCVE
0021         N5=N4+2*NCVE
0022         N6=N5+2*NCVE
0023         N7=N6+2*NCVE
0024         N8=N7+2*NVE
0025         N9=N8+2*NVE
0026         N10=N9+2*NCVE
0027         N11=N10+2*NCVE
0028   C
0029   C     COMPRUEBA SI LA MEMORIA DISPONIBLE ES SUFICIENTE PARA EL CASO A
0030   C     RESOLVER
0031   C
0032         IF(N11.GT.ICOMM)GOTO 100
0033   C
0034   C     LLAMADA A LA SUBRUTINA DE EJCUCION
0035   C
0036         CALL ALNER(IW(1),IW(N2),IW(N3),IW(N4),IW(N5),IW(N6),IW(N7),IW(N8
0037        *IW(N9),IW(N10))
0038         GOTO 101
0039   C
0040   C     ESCRITURA DE MENSAJE PARA EL CUANDO NO HAY MEMORIA SUFICIENTE
0041   C
0042      100 WRITE(IMPR,15)N11
0043      15  FORMAT(3/,4X,'OPTI1 SE SUPERA LA CAPACIDAD ACTUAL N11 = ',I5)
0044      101 CONTINUE
0045   C
0046   C     DEVUELVE EL CONTROL AL PROGRAMA PRINCIPAL OPRIS
0047   C
0048         CALL OVRTN
0049         END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

```
PAGE 0003  FTN.  12:25 AM  SAT., 16  JAN., 1982

0050          SUBROUTINE ALNER(Z,A,B,C,D,ZI,VALNO,VALNA,TAMSI,TAMSO)
0051    C
0052    C     REALIZA LA ESCRITURA DE LOS DATOS PROPIOS DE ESTE SEGMENTO
0053    C     ADEMAS REALIZA LA RESOLUCION DEL SISTEMA DE ECUACIONES TAN-
0054    C     TAS VECES COMO SE LE INDIQUE PARA DISTINTOS VECTORES INICI-
0055    C     ALES.ESCRIBE TODOS LOS RESULTADOS OBTENIDOS EN DISTINTOS
0056    C     FORMAS SEGUN SE LE INDIQUE EN LOS DATOS DE EJECUCION
0057    C
0058          COMMON/COMU/LECT,IMPR,TITU(30),NCVE,ZZ(30),NVE,ICOMM
0059          COMMON/NEUT/TEMP(30,20),ICONTA,TOLER,DINCR,ANORMI,ESCR,ZMA(30,20
0060          COMMON/LOGIC/COVER,NLI,NSRE,PRINT,NITR,JINS,NCORE,SUMI,IMPME,IMP
0061          COMMON//IW(1)
0062          LOGICAL COVER,NLI,NSRE,PRINT,IMPME,IMPNA
0063          DIMENSION Z(1),A(1,1),B(1),C(1),D(1),ZI(1),VALNO(1),TAMSI(1),
0064         *VALNA(1),TAMSO(1)
0065    C
0066    C     ESCRITURA DEL ENCABEZADO Y DEMAS DATOS DE EJECUCION
0067    C
0068          WRITE(IMPR,1002)TITU
0069     1002 FORMAT(1H1,70(1H*),/,1X,5(1H*),30A2,5(1H*),/,1X,27(1H*),
0070         *" NEWTON-RAPHSON ",27(1H*),/,1X,70(1H*),/)
0071          PRINT=.FALSE.
0072          IMPME=.FALSE.
0073          IMPNA=.TRUE.
0074          ICONTA=1
0075          IF(NITR.LT.0)IMPME=.TRUE.
0076          IF(ANORMI.LT.0)IMPNA=.FALSE.
0077          IF(ESCR.LT.0)PRINT=.TRUE.
0078          WRITE(IMPR,1006)
0079     1006 FORMAT(/,5X,50(1H*),/)
0080          WRITE(IMPR,1003)NCVE,NVE,ANORMI,NITR,TOLER,DINCR
0081     1003 FORMAT(5X,"ECUACIONES = ",I5,/,5X,"CICLOS DE ITERAC. = ",I5,
0082         *,/,5X,"NORMA MINIMA = ",I5,/,5X,"ITERACIONES = ",I5,/,5X,
0083         *"TOLERANCIA = ",1PE15.7,/,5X,"DIF-FIN = ",1PE15.7)
0084          IF(NCVE.GT.30)GOTO 75
0085          NITR=IABS(NITR)
0086          ANORMI=ABS(ANORMI)
0087          WRITE(IMPR,1006)
0088          DO 99 I=1,NVE
0089          DO 12 K=1,NCVE
0090          Z(K)=ZMA(K,I)
0091          ZI(K)=Z(K)
0092       12 CONTINUE
0093    C
0094    C    LLAMADA A LA SUBRUTINA DEL ALGORITMO DE NEWTON-RAPHSON
0095    C
0096          CALL ANERA(NCVE,Z,A,B,C,D,TOLER,LEST,DINCR,ANORMI,IMPR)
0097          N=0
0098          IF(LEST.EQ.1)GOTO 655
0099          IF(COVER)GOTO 30
0100          IF(NLI)GOTO 55
0101          IF(NCORE.GE.12)GOTO 65
0102          GOTO 99
0103       30 DO 31 II=1,NCVE
0104       31 TEMP(II,ICONTA)=Z(II)
```

```
      IF(ICONTA.EQ.1)GOTO 40
      DO 33 J=1,ICONTA
      IF(J.EQ.1)GOTO 33
C
C     COMPARACION ENTRE LAS SOLUCIONES OBTENIDAS A FIN DE VER SI SON
C     LAS MISMAS O DISTINTAS DE FORMA QUE DETECTE CUANTAS SOLUCIONES
C     DISTINTAS AL MISMO SISTEMA ENCUENTRA
C
      DO 32 M=1,NCVE
      JJ=J-1
      VCOM=ABS(Z(M)-TEMP(M,JJ))
      IF(VCOM.LT.0.00001)N=N+1
      IF(N.EQ.NCVE)GOTO 50
   32 CONTINUE
   33 CONTINUE
   40 N=0
      DO 41 M=1,NCVE
      VCOM=ABS(Z(M)-ZI(M))
      IF(VCOM.LT.0.0001)N=N+1
      IF(N.EQ.NCVE)GOTO 61
   41 CONTINUE
      GOTO 43
C
C     ESCRITURA DE RESULTADOS Y DISTINTOS MENSAJES SOBRE LA CONVERGENCIA
C
   61 COVER=.FALSE.
      IF(IMPNA)GOTO 100
      WRITE(IMPR,62)I,(N,ZI(N),N=1,NCVE)
   62 FORMAT(/,5X,"CICLO = ",I3,/,5X,"NO HAY MEJORA A PARTIR DEL VECTOR
     *INICIAL",2/,(25X,"Z(",I2,")=",E14.7))
  100 CONTINUE
      GOTO 99
   43 CONTINUE
      IF(IMPNA)GOTO 111
      WRITE(IMPR,1025)
 1025 FORMAT(16X,40(1H+))
      WRITE(IMPR,45)ICONTA,ITNS,(N,ZI(N),N=1,NCVE)
   45 FORMAT(2/,16X,5(1H)),I3," SOLUCION ",I4," ITERACIONES "
     *,5(1H(),/,16X,"VECTOR INICIAL",2/,(25X,"Z(",I2,")=",E14.7))
  111 CONTINUE
      VALNO(ICONTA)=SUMI
      IF(IMPNA)GOTO 101
      WRITE(IMPR,46)SUMI,(N,Z(N),N=1,NCVE)
   46 FORMAT(/,16X,"VECTOR FINAL. NORMA =",E15.7,2/,
     *(25X,'Z(',I2,')=',E14.7))
      WRITE(IMPR,1026)
 1026 FORMAT(16X,40(1H#))
  101 CONTINUE
      ICONTA=ICONTA+1
   50 COVER=.FALSE.
      GO TO 99
   55 IF(IMPNA)GOTO 102
      WRITE(IMPR,60)ITNS,SUMI,(N,Z(N),N=1,NCVE)
  102 CONTINUE
      NLI=.FALSE.
```

```
0160      60 FORMAT(/,5X,'SE SUPERA EL NUMERO DE ITERACIONES',I3,/,16X,'VALOR
0161        *CTUAL DE Z. NORMA =',E15.7,2/,(25X,'Z(',I2,')=',E14.7))
0162         IF(INPNA)GOTO 103
0163         WRITE(INPR,161)(N,ZI(N),N=1,NCVE)
0164        161 FORMAT(/,5X,'EL VECTOR INICIAL FUE',2/,(25X,'Z(',I2,')=',E14.7))
0165        103 CONTINUE
0166         GOTO 99
0167         65 IF(INPNA)GOTO 99
0168         IF(INPNE)GOTO 655
0169         IF(SUMI.GE.ANORMI)GOTO 99
0170        655 IF(INPNA)GOTO 99
0171         WRITE(INPR,68)ITNS,SUMI,(N,Z(N),N=1,NCVE)
0172         68 FORMAT(/,5X,'SE SUPERA EL NUMERO DE C.RETARDADA EN LA ITERACION',
0173           1I3,/,16X,'VALOR ACTUAL DE Z.  NORMA =',E15.7,2/,
0174           *(25X,'Z(',I2,')=',E14.7))
0175         WRITE(INPR,161)(N,ZI(N),N=1,NCVE)
0176         99 CONTINUE
0177         ICONTA=ICONTA-1
0178         WRITE(INPR,70)ICONTA,NVE
0179         70 FORMAT(/,16X,5(1H>),I3,' SOLUCIONES PARA ',I3,' CICLOS',5(1H<))
0180         WRITE(INPR,1024)
0181       1024 FORMAT(16X,40(1H#))
0182          C
0183          C      ORDENA LAS SOLUCIONES OBTENIDS DE MEJOR A PEOR SEGUN LA NORMA
0184          C
0185         DO 1670 LRT=1,ICONTA
0186         VALNA(LRT)=VALNO(LRT)
0187         DO 1670 JNR=1,NCVE
0188         TANSI(JNR)=0.0
0189         TANSO(JNR)=0.0
0190       1670 CONTINUE
0191         NO1=1
0192         B01=0.0
0193        120  A01=1.E10
0194         DO 130 LTX=1,ICONTA
0195         IF(VALNA(LTX).GT.A01)GOTO 130
0196         IF(VALNA(LTX).LE.B01)GOTO 130
0197         A01=VALNA(LTX)
0198         J01=LTX
0199        130  CONTINUE
0200         VALNO(NO1)=VALNA(J01)
0201         B01=VALNA(J01)
0202         VALNA(J01)=VALNA(NO1)
0203         VALNA(NO1)=B01
0204         DO 1800 IPUNT=1,NCVE
0205         TANSI(IPUNT)=TEMP(IPUNT,J01)
0206         TANSO(IPUNT)=TEMP(IPUNT,NO1)
0207       1800 CONTINUE
0208         DO 140 IMC=1,NCVE
0209         TEMP(IMC,NO1)=TANSI(IMC)
0210         TEMP(IMC,J01)=TANSO(IMC)
0211        140  CONTINUE
0212         NO1=NO1+1
0213         IF(NO1.LE.ICONTA)GOTO 120
0214         WRITE(INPR,104)
```

```
 PAGE 0006  ALNER 12:25 AM  SAT., 16  JAN., 1982


0215          WRITE(IMPR,1024)
0216    C
0217    C     ESCRIBE EL RESUMEN FINAL DE LAS SOLUCIONES OBTENIDAS ORDENADAS
0218    C     DE MEJOR A PEOR
0219    C
0220          DO 105 ISL=1,ICONTA
0221          WRITE(IMPR,106)ISL,VALNO(ISL)
0222          DO 105 ISP=1,NCVE
0223          WRITE(IMPR,107)ISP,ISL,TEMP(ISP,ISL)
0224     105  CONTINUE
0225     104  FORMAT(16X,5(1H+)," RESUMEN FINAL ",15X,5(1H+))
0226     106  FORMAT(/,16X,"SOLUCION ",I2," NORMA = ",1PE15.7,/)
0227     107  FORMAT(23X,"Z(",I2,",",I2,")=",1PE15.7)
0228          GOTO 1
0229    C
0230    C     ESCRIBE UN MENSAJE CUANDO EL NUMERO DE ECUACIONES ES MAYOR DE 30
0231    C
0232       75 WRITE(IMPR,80)
0233       80 FORMAT(2/,5X,"EL NUMERO DE ECUACIONES ES MAYOR DE 30")
0234        1 RETURN
0235          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 01513       COMMON = 00001
```

```
0236          SUBROUTINE ANERA(NVA,Z,A,B,C,D,TOLER,LEST,DINCR,ANORMI,IMPR)
0237    C
0238    C     ALGORITMO DE NEWTON-RAPHSON PARA RESOLUCION DE SISTEMAS DE ECUA-
0239    C     CIONES NO LINEALES CON CORRECCION RETARDADA
0240    C
0241          COMMON/LOGIC/COVER,MLI,NSRE,PRINT,NITR,ITNS,NCORE,SUMI,IMPME,IMPR
0242          LOGICAL COVER,MLI,NSRE,PRINT,IMPME,IMPR
0243          DIMENSION Z(NVA),A(NVA,NVA),B(NVA),C(NVA),D(NVA)
0244          MLIIT=NITR
0245          LEST=0
0246          ITNS=0
0247          NCORE=0
0248          SUMI=0.0
0249    C
0250    C     CALCULO DEL VALOR DE LAS ECUACIONES
0251    C
0252          DO 100 I=1,NVA
0253          CALL GRADI(Z,I,0,NVA,VADE,DINCR)
0254          D(I)=-VADE
0255      100 SUMI=SUMI+ABS(D(I))
0256          IF(SUMI.LT.TOLER)GOTO 500
0257      200 DO 205 I=1,NVA
0258          B(I)=D(I)
0259    C
0260    C     CALCULO DEL GRADIENTE DE LAS ECUACIONES DERIVADAS PARCIALES
0261    C
0262          DO 205 J=1,NVA
0263          CALL GRADI(Z,I,J,NVA,VADE,DINCR)
0264      205 A(I,J)=VADE
0265          NN=NVA*NVA
0266    C
0267    C     CALCULO DE VECTOR CORRECCION.RESOLUCION DEL SISTEMA LINEAL
0268    C
0269          CALL SYST(NVA,NN,A,B,KS)
0270          IF(KS.EQ.1)GOTO 700
0271      210 ITNS=ITNS+1
0272      300 DO 305 I=1,NVA
0273      305 C(I)=Z(I)+B(I)
0274    C
0275    C     COMPROBACION DE LA DISMINUCION DE LA NORMA DE LA FUNCION VECTORIAL
0276    C     PARA EL VECTOR DE CORRECCION DETERMINADO
0277    C
0278          ANSRE=0.0
0279          DO 310 I=1,NVA
0280          CALL GRADI(C,I,0,NVA,VADE,DINCR)
0281          D(I)=-VADE
0282      310 ANSRE=ANSRE+ABS(D(I))
0283          IF(PRINT)GOTO 900
0284          IF(ANSRE.LE.TOLER)GOTO 316
0285      315 IF(ANSRE.GT.SUMI)GOTO 900
0286      316 CONTINUE
0287          NSRE=.TRUE.
0288          SUMI=ANSRE
0289          NCORE=0
0290          DO 400 I=1,NVA
```

```
0291      400 Z(I)=C(I)
0292    C
0293    C     COMPROBACION DE LA CONVERGENCIA
0294    C
0295          DO 500 I=1,NVA
0296          VCOM=ABS(B(I))
0297          IF(ABS(Z(I)).GE.1.0)VCOM=ABS(B(I)/Z(I))
0298          IF(VCOM.GT.1.0E-7)GOTO 600
0299      500 CONTINUE
0300          IF(SUMI.LT.ANSREI)COVER=.TRUE.
0301          RETURN
0302      600 IF(ITNS.LE.NLIIT)GOTO 200
0303          MLI=.TRUE.
0304          RETURN
0305    C
0306    C     LA SYST DA MATRIZ SINGULAR.SE ESCRIBEN LAS MATRICES QUE ENTRAN EN
0307    C     EL SISTEMA
0308    C
0309      700 DO 705 I=1,NVA
0310          B(I)=D(I)
0311          DO 705 J=1,NVA
0312          CALL GRADI(Z,I,J,NVA,VADE,DINCR)
0313      705 A(I,J)=VADE
0314          IF(IMPRA)GOTO 719
0315          WRITE(IMPR,10)ITNS
0316       10 FORMAT(/,5X,"MATRIZ SINGULAR EN LA ITERACION",I3,/,5X,"LAS MATRI
0317         *S B(I) Y A(I,J) SON:",/)
0318          IF(IMPME)GOTO 719
0319          GOTO 719
0320      718 DO 710 I=1,NVA
0321      710 WRITE(IMPR,20)I,B(I),(A(I,J),J=1,NVA)
0322       20 FORMAT(1X,I4,E15.7,5X,3E15.7/(25X,3E15.7))
0323      719 DO 720 I=1,NVA
0324          VAESC=0.0
0325          DO 715 J=1,NVA
0326          AIJ=ABS(A(I,J))
0327      715 IF(AIJ.GT.VAESC)VAESC=AIJ
0328          IF(VAESC.LT.1.0E-10)VAESC=1.0
0329          B(I)=D(I)/VAESC
0330          DO 720 J=1,NVA
0331      720 A(I,J)=A(I,J)/VAESC
0332          IF(IMPRA)GOTO 726
0333          IF(IMPME)GOTO 723
0334          GOTO 726
0335      723 WRITE(IMPR,30)
0336       30 FORMAT(5X,"ESTAN ESCALADAS A")
0337          DO 725 I=1,NVA
0338      725 WRITE(IMPR,20)I,B(I),(A(I,J),J=1,NVA)
0339      726 NN=NV4*NVA
0340          CALL SYST(NVA,NN,A,B,KS)
0341          IF(KS.NE.1)GOTO 210
0342          LEST=1
0343          RETURN
0344    C
0345    C     ESCRITURA DE LAS DISTINTAS VARIABLES EN CADA ITERACION SI ASI SE
```

```
PAGE 0009  ANEXA 12:25 AM  SAT., 16  JAN., 1982


0346   C    INDICAN PREVIAMENTE
0347   C
0348     900 WRITE(IMPR,40)ITNS
0349      40 FORMAT(/,2X,"ITERACION ",I3,2/,7X,"CORRECCION",10X,"VECTOR CORRE
0350        *DO",13X,"ECUACIONES",/)
0351          DO 845 I=1,NVA
0352          WRITE(IMPR,50)I,B(I),I,C(I),I,D(I)
0353      50 FORMAT(2X,"C(",I2,")=",E14.7,3X,"Z(",I2,")=",E14.7,3X,"F(",I2,")=
0354        *,E14.7)
0355     845 CONTINUE
0356          GOTO 315
0357   C
0358   C    CORRECCION RETARDADA CUANDO LA NORMA DE LA FUNCION DECRECE
0359   C
0360     900 NCORE=NCORE-1
0361          IF(NCORE.GT.12)RETURN
0362          NSRE=.FALSE.
0363          VRESC=5.0
0364     910 DO 915 I=1,NVA
0365     915 B(I)=B(I)/VRESC
0366          GO TO  300
0367          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00903      COMMON = 00000
```

```
0368          SUBROUTINE SYST(N,NN,A,B,KS)
0369    C
0370    C     RESOLUCION DE SISTEMAS DE ECUACIONES LINEALES MEDIANTE LA REDUCCIO
0371    C     DE GAUSS CON PIVOTAMIENTO TOTAL
0372    C
0373          DIMENSION A(NN),B(N)
0374    C
0375    C     PRIMERA SOLUCION
0376    C
0377          TOL=1.E-20
0378          KS=0
0379          JJ=-N
0380          DO 65 J=1,N
0381          JY=J+1
0382          JJ=JJ+N+1
0383          BIGA=0
0384          IT=JJ-J
0385          DO 30 I=J,N
0386    C
0387    C     BUSQUEDA DEL MAXIMO COEFICIENTE DE LA COLUMNA
0388    C
0389          IJ=IT+I
0390          IF(ABS(BIGA)-ABS(A(IJ)))20,30,30
0391       20   BIGA=A(IJ)
0392          IMAX=I
0393       30 CONTINUE
0394    C
0395    C     COMPRUEBA SI EL MAYOR VALOR ES MENOR QUE LA TOLERANCIA (MATRIZ
0396    C     SINGULAR)
0397    C
0398          IF(ABS(BIGA)-TOL)35,35,40
0399       35   KS=1
0400            RETURN
0401    C
0402    C     INTERCAMBIO DE FILAS SI ES NECESARIO
0403    C
0404       40 II=J+N*(J-2)
0405          IT=IMAX-J
0406          DO 50 K=J,N
0407          II=II+N
0408          I2=II+IT
0409          SAVE=A(II)
0410          A(II)=A(I2)
0411          A(I2)=SAVE
0412    C
0413    C     DIVIDE LA ECUACION POR EL COEFICIENTE DE MAYOR RANGO
0414    C
0415       50    A(II)=A(II)/BIGA
0416             SAVE=B(IMAX)
0417          B(IMAX)=B(J)
0418          B(J)=SAVE/BIGA
0419    C
0420    C     ENLIMINACION DE LA VARIABLE CORRESPONDIENTE
0421    C
0422          IF(J-N)55,70,55
```

```
PAGE 0011  SYST  12:25 AM  SAT., 16  JAN., 1982


0423      55 IOS=N*(J-1)
0424         DO 65 IX=JY,N
0425         IXJ=IOS+IX
0426         IT=J-IX
0427         DO 60 JX=JY,N
0428         IXJX=N*(JX-1)+IX
0429         JJX=IXJX+IT
0430      60 A(IXJX)=A(IXJX)-(A(IXJ)*A(JJX))
0431      65  B(IX)=B(IX)-(B(J)*A(IXJ))
0432   C
0433   C    ULTIMA SOLUCION
0434   C
0435      70  NY=N-1
0436         IT=N*N
0437         DO 80 J=1,NY
0438         IA=IT-J
0439         IB=N-J
0440         IC=N
0441         DO 80 K=1,J
0442         B(IB)=B(IB)-A(IA)*B(IC)
0443         IA=IA-N
0444      80 IC=IC-1
0445         RETURN
0446         END


 FTN4 (COMPILER) HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00427    COMMON = 00000
```

```
PAGE 0001  FTN.  12:50 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM OPTI2(5)
0003   C
0004   C     SEGUNDO SEGMENTO DEL PROGRAMA OPRIS.METRICA VARIABLE
0005   C
0006         COMMON/COMU/LECT,IMPR,TITU(30),NCVE,X(30),NVE,ICOMM
0007         COMMON//IU(1)
0008         DO 10 I=1,ICOMM
0009      10 IU(I)=0
0010   C
0011   C     DETERMINACION DE LOS INDICES DE LEA MEMORIA DINAMICA
0012   C
0013         M=NCVE*(NCVE+7)/2
0014         N2=1+2*NCVE
0015         N3=N2+2*M
0016   C
0017   C     COMPRUEBA SI HAY SUFICIENTE MEMORIA
0018   C
0019         IF(N3.GT.ICOMM)GOTO 100
0020   C
0021   C     LLAMADA A LA SUBRUTINA DE CONTROL DEL PROGRAMA
0022   C
0023         CALL SOFPS(NCVE,M,X,IU(1),IU(N2),IMPR,TITU)
0024         GO TO 101
0025   C
0026   C     ESCRIBE UN MENSAJE SI NO HAY MEMORIA SUFICIENTE
0027   C
0028     100 WRITE(IMPR,15)N3
0029      15 FORMAT(3/,4X,"IOPT2 SE SUPERA LA CAPACIDAD ACTUAL,N3 - ",I5)
0030   C
0031   C     DEVUELVE EL CONTROL AL PROGRAMA PRINCIPAL OPRIS
0032   C
0033     101 CALL OVRTN
0034         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)


 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00109       COMMON = 00001
```

```
0035          SUBROUTINE SCFPS(N,M,X,G,H,IMPR,TITU)
0036    C
0037    C     SUBRUTINA DE CONTROL DE LA EJECUCION DE LA METRICA VARIABLE
0038    C
0039          COMMON/OPMV/ERRAB,VMIFO,LIMIT,DINCR
0040          COMMON//IW(1)
0041          DIMENSION X(1),G(1),H(1),TITU(30)
0042    C
0043    C     LECTURA Y ESCRITURA DE DATOS Y ENCABEZADO
0044    C
0045          WRITE (IMPR,3)TITU
0046        3 FORMAT(1H1,/,1X,69(1H*),/,5(1H*),30A2,5(1H*),/,1X,26(1H*)," METR
0047         *A VARIABLE ",25(1H*),/,1X,69(1H*),2/)
0048          WRITE (IMPR,4)N,LIMIT,ERRAB,VMIFO,DINCR
0049        4 FORMAT(5X,62(1H*),/,5X,"DIMENSION DEL V E = ",I5,/,5X,"ITERACION
0050         * = ",I5,/,5X,"V.ERRABIMADO F. = ",1PE15.7,/,5X,"V.ERRAB.PASO = "
0051         *,7,/,5X,"DIF-FIN = ",1PE15.7,/,5X,62(1H*))
0052          WRITE (IMPR,5)
0053        5 FORMAT (/,25X,"VECTOR INICIAL",/)
0054          DO 100 J=1,N
0055          WRITE (IMPR,6) J,X(J)
0056        6 FORMAT(25X,"X(",I2,") = ",1PE15.7)
0057      100 CONTINUE
0058    C
0059    C     LLAMADA A LA SUBRUTINA QUE REALIZA EL ALGORITMO
0060    C
0061          CALL FPSMV(N,M,X,F,G,ERRAB,VMIFO,LIMIT,INDER,H,CONTA,DINCR)
0062    C
0063    C     ESCRITURA DE LOS RESULTADOS
0064    C
0065          WRITE (IMPR,7)
0066        7 FORMAT(/,25X,"RESULTADO FINAL",/)
0067          WRITE (IMPR,8)INDER,CONTA
0068        8 FORMAT(25X,"CONVERG = ",I3,/,25X,"N.ITERAC = ",I5)
0069          WRITE (IMPR,9)F
0070        9 FORMAT(25X,"FU.OBJ = ",1PE15.7)
0071          WRITE (IMPR,10)
0072       10 FORMAT(/,25X,"VECTOR FINAL ",/)
0073          DO 200 J=1,N
0074          WRITE (IMPR,6)J,X(J)
0075      200 CONTINUE
0076          RETURN
0077          END
```

```
0078          SUBROUTINE FPSMV(N,M,X,F,G,ERRAB,VNIFO,LIMIT,INDER,H,CONTA,DINCR)
0079    C
0080    C     SUBRUTINA QUE EJECUTA EL ALGORITMO
0081    C
0082          DIMENSION X(N),G(N),H(M)
0083    C
0084    C     CALCULO DEL VALOR DE LA FUNCION Y EL VECTOR GRADIENTE PARA
0085    C     EL ARGUMENTO INICIAL
0086    C
0087          CALL FUOBJ(N,X,F,G,DINCR)
0088    C
0089    C     PUESTA A CERO DEL CONTADOR DE ITERACIONES Y GENERACION
0090    C     DE LA MATRIZ UNIDAD
0091    C
0092          INDER=0
0093          CONTA=0
0094          N2=N+N
0095          N3=N2+N
0096          N31=N3+1
0097        1 K=N31
0098          DO 4 J=1,N
0099          H(K)=1.
0100          NJ=N-J
0101          IF(NJ)5,5,2
0102        2 DO 3 L=1,NJ
0103          KL=K+L
0104        3 H(KL)=0.
0105        4 K=KL+1
0106    C
0107    C     INICIO DEL CICLO DE ITERACION
0108    C
0109        5 CONTA=CONTA+1
0110    C
0111    C     GUARDA EL  VALOR DE LA FUNCION,VECTOR ARGUMENTO Y
0112    C     VECTOR GRADIENTE
0113    C
0114          ANTF=F
0115          DO 9 J=1,N
0116          K=N+J
0117          H(K)=G(J)
0118          K=K+N
0119          H(K)=X(J)
0120    C
0121    C     DETERMINA LA DIRECCION DEL VECTOR H
0122    C
0123          K=J+N3
0124          T=0.
0125          DO 8 L=1,N
0126          T=T-G(L)*H(K)
0127          IF(L-J)6,7,7
0128        6 K=K+N-L
0129          GOTO 8
0130        7 K=K+1
0131        8 CONTINUE
0132        9 H(J)=T
```

```
0133   C
0134   C      COMPRUEBA SI LA FUNCION DECRECE CONSECUTIVAMENTE A LO LARGO DE H
0135   C
0136          DY=0.
0137          HGRA=0.
0138          GRAN=0.
0139   C
0140   C      CALCULA DERIVADA DIRECCIONAL Y COMPRUEBA LOS VALORES PARA EL
0141   C      VECTOR DIRECCION H Y EL VECTOR GRADIENTE G
0142   C
0143          DO 10 J=1,N
0144          HGRA=HGRA+ABS(H(J))
0145          GRAN=GRAN+ABS(G(J))
0146       10 DY=DY+H(J)*G(J)
0147   C
0148   C      REPITE LA BUSQUEDA EN DIRECCION DE LA PENDIENTE DESCENDIENTE SI
0149   C      LA DERIVADA DIRECCIONAL APARECE POSITIVA O CERO
0150   C
0151          IF(DY)11,51,51
0152   C
0153   C      REPITE LA BUSQUEDA EN LA DIRECCION DE LA PENDIENTE DESCENDIENTE
0154   C      SI EL VECTOR DIRECCION H ES PEQUE'O COMPARADO CON EL VECTOR
0155   C      GRADIENTE G
0156   C
0157       11 IF(HGRA/GRAN-VNIFO)51,51,12
0158   C
0159   C      BUSCA EL MINIMO A LO LARGO DE LA DIRECCION H PARA
0160   C      DERIVADA DIRECCIONAL POSITIVA
0161   C
0162       12 FY=F
0163          FALA=2.*(ERRAB-F)/DY
0164          DBNAN=1.
0165   C
0166   C       USA EL VALOR ERRABINADO COMO VALOR DEL INTERVALO SOLO SI ES
0167   C      POSITIVO Y MENOR QUE 1. SI NO TOMA 1. COMO INTERVALO O
0168   C      INCREMENTO.
0169   C
0170          IF(FALA)15,15,13
0171       13 IF(FALA-DBNAN)14,15,15
0172       14 DBNAN=FALA
0173       15 FALA=0.
0174   C
0175   C      GUARDA LOS VALORES DE LA FUNCION Y DERIVADA PARA LOS ANTIGUOS
0176   C      ARGUMENTOS
0177   C
0178       16 FX=FY
0179          DX=DY
0180   C
0181   C      CAMBIO DE ARGUMENTO A LO LARGO DE H
0182   C
0183          DO 17 I=1,N
0184       17 X(I)=X(I)+DBNAN*H(I)
0185   C
0186   C      CALCULA EL VALOR DE LA FUNCION Y EL GRADIENTE PARA EL NUEVO
0187   C      ARGUMENTO
```

```
PAGE 0005  FPSMV 12:50 AM  SAT., 16  JAN., 1982


0188   C
0189         CALL FUOBJ(N,X,F,G,DINCR)
0190         FY=F
0191   C
0192   C      CALCULA LA DERIVADA DIRECCIONAL(DY) PARA EL NUEVO ARGUMENTO,
0193   C      TERMINA LA BUSQUEDA,SI DY ES POSITIVO.SI DY ES CERO HEMOS
0194   C      ENCONTRADO EL MINIMO
0195   C
0196         DY=0.
0197         DO 18 I=1,N
0198      18 DY=DY+G(I)*H(I)
0199         IF(DY)19,36,22
0200   C
0201   C      TERMINA LA BUSQUEDA TAMBIEN SI EL VALOR DE LA FUNCION INDICA QUE
0202   C      SE PASO EL MINIMO
0203   C
0204      19 IF(FY-FX)20,22,22
0205   C
0206   C      REFITE LA BUSQUEDA Y DOBLA EL INTERVALO PARA ADELANTAR LA
0207   C      BUSQUEDA
0208   C
0209      20 DBMAA=DBMAA-FALA
0210         FALA=DBMAA
0211   C
0212   C      FIN DEL CICLO DE BUSQUEDA
0213   C      TERMINA SI EL CAMBIO EN ARGUMENTOS OBTENIDOS ES MUY GRANDE
0214   C
0215         IF(HGRA*DBMAA-1.E10)16,16,21
0216   C
0217   C      LA TECNICA DE BUSQUEDA LINEAL INDICA QUE EL MINIMO NO EXISTE
0218   C
0219      21 INDER=2
0220         RETURN
0221   C
0222   C      INTERPOLACION CUBICA EN EL INTERVALO DEFINIDO PARA LA
0223   C      BUSQUEDA Y ADEMAS CALCULA EL ARGUMENTO DE X PARA EL QUE LA
0224   C      INTERPOLACION POLINOMIAL ES MINIMIZADA
0225   C
0226      22 T=0.
0227      23 IF(DBMAA)24,36,24
0228      24 Z=3.*(FX-FY)/DBMAA+DX+DY
0229         FALA=AMAX1(ABS(Z),ABS(DX),ABS(DY))
0230         DFALA=Z/FALA
0231         DFALA=DFALA*DFALA-DX/FALA*DY/FALA
0232         IF(DFALA)51,25,25
0233      25 W=FALA*SQRT(DFALA)
0234         FALA=(DY+W-Z)*DBMAA/(DY+2.*W-DX)
0235         DO 26 I=1,N
0236      26 X(I)=X(I)+(T-FALA)*H(I)
0237   C
0238   C      TERMINA,SI EL VALOR ACTUAL DE LA FUNCION EN X ES MENOR QUE LOS
0239   C      VALORES DE LA FUNCION EN EL INTERVALO FINAL,SI NO REDUCE EL
0240   C      INTERVALO ESCOGIENDO UN PUNTO FINAL IGUAL A X Y REPETE LA
0241   C      INTERPOLACION.EL PUNTO FINAL ES ESCOGIDO DEPENDIENDO DEL
0242   C      VALOR DE LA FUNCION Y SU GRADIENTE EN X
```

```
0243   C
0244           CALL FUOBJ (N,X,F,G,DINCR)
0245           IF(F-FX)27,27,29
0246        27 IF(F-FY)36,36,28
0247        28 DFALA=0.
0248           DO 29 I=1,N
0249        29 DFALA=DFALA-G(I)*H(I)
0250           IF(DFALA)30,33,33
0251        30 IF(F-FX)32,31,33
0252        31 IF(DX-DFALA)32,36,32
0253        32 FX=F
0254           DX=DFALA
0255           T=FALA
0256           DBMAA=FALA
0257           GOTO 23
0258        33 IF(FY-F)35,34,35
0259        34 IF(DY-DFALA)35,36,35
0260        35 FY=F
0261           DY=DFALA
0262           DBMAA=DBMAA-FALA
0263           GOTO 22
0264   C
0265   C       CALCULA LA DIFERENCIA DE LOS VECTORES ARGUMENTO Y GRADIENTE PARA
0266   C       DOS ITERACIONES CONSECUTIVAS
0267   C
0268        36 DO 37 J=1,N
0269           K=N+J
0270           H(K)=G(J)-H(K)
0271           K=N+K
0272        37 H(K)=X(J)-H(K)
0273   C
0274   C       TERMINA,SI LA FUNCION NO DECRECE DURANTE LA ULTIMA ITERACION
0275   C
0276           IF(ANTF-F+VMIFO)51,38,38
0277   C
0278   C       COMPRUEBA EL TAMA'O DE LA DIFERENCIA DEL VECTOR Y EL VECTOR
0279   C       DIRECCION SI AL MENOS SE HAN REALIZADO N ITERACIONES.
0280   C       TERMINA,SI AMBOS SON MENORES QUE VMIFO
0281   C
0282        38 INDER=0
0283           IF(CONTA-N)42,39,39
0284        39 T=0.
0285           Z=0.
0286           DO 40  J=1,N
0287           K=N+J
0288           W=H(K)
0289           K=K+N
0290           T=T+ABS(H(K))
0291        40 Z=Z+W*H(K)
0292           IF(HGBA-VMIFO)41,41,42
0293        41 IF(T-VMIFO)56,56,42
0294   C
0295   C       TERMINA,SI EL NUMERO DE ITERACIONES EXCEDE A LIMIT
0296   C
0297        42 IF(CONTA-LIMIT)43,50,50
```

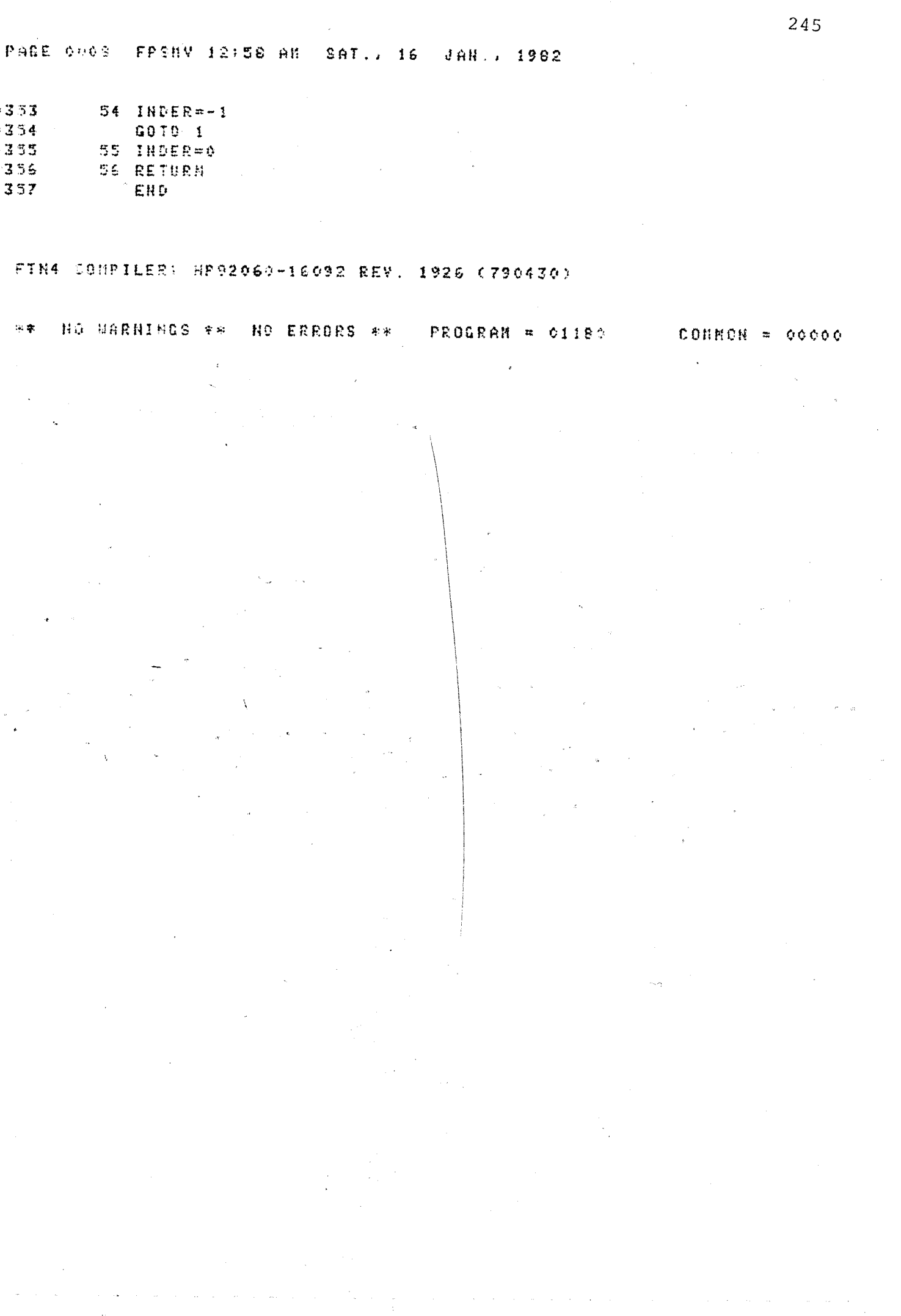
```
C
C       PREPARA LA ACTUALIZACION DE LA MATRIZ H
C
   43 FALA=0.
      DO 47 J=1,N
      K=J+NN
      W=0.
      DO 46 L=1,N
      KL=N+L
      W=W+H(KL)*H(K)
      IF(L-J)44,45,45
   44 K=K+N-L
      GOTO 46
   45 K=K+1
   46 CONTINUE
      K=N+J
      FALA=FALA+W*H(K)
   47 H(J)=W
C
C       REPITE LA BUSQUEDA EN LA DIRECCION DE PENDIENTE DESCENDIENTE SI
C       LOS RESULTADOS OBTENIDOS NO SON SATISFACTORIOS
C
      IF(Z*FALA)48,1,48
C
C       ACTUALIZA LA MATRIZ H
C
   48 K=NN1
      DO 49  L=1,N
      KL=N2+L
      DO 49  J=L,N
      NJ=N2+J
      H(K)=H(K)+H(KL)*H(NJ)/Z-H(L)*H(J)/FALA
   49 K=K+1
      GOTO 5
C
C       NO CONVERGE DESPUES DE LIMIT ITERACIONES
C
   50 INDER=1
      RETURN
C
C       RESTAURA LOS ANTIGUOS VALORES DE LA FUNCION Y ARGUMENTOS
C
   51 DO 52 J=1,N
      K=N2+J
   52 X(J)=H(K)
      CALL FUOBJ(N,X,F,G,DINCR)
C
C       REPITE EN LA DIRECCION DE PENDIENTE DESCENDIENTE SI LOS ERRORES
C       DE LA DERIVADA SON SUFICIENTEMENTE PEQUEÑOS
C
      IF(GRAN-VMIFO)55,55,53
C
C       COMPRUEBA LOS REPETIDOS FALLOS DE LA ITERACION
C
   53 IF(INDER)56,54,54
```

PAGE 0003 FPSNV 12:58 AM SAT., 16 JAN., 1982

```
0353      54 INDER=-1
0354         GOTO 1
0355      55 INDER=0
0356      56 RETURN
0357         END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 01180 COMMON = 00000



```
PAGE 0001  FTN.   1:22 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM OPTI3(5)
0003   C
0004   C     TERCER SEGMENTO DEL PROGRAMA OPRIS.GRADIENTE CONJUGADO
0005   C
0006         COMMON/COMU/LECT,IMPR,TITU(30),NCVE,X(30),NVE,ICOMM
0007         COMMON//IU(1)
0008   C
0009   C      LECTURA DE DATOS INICIALES PARA MEMORIA DINAMICA
0010   C
0011         DO 10 I=1,ICOMM
0012      10 IU(I)=0
0013   C
0014   C      DETERMINACION DE LOS INDICES DE LA MEMORIA DINAMICA
0015   C
0016         M=2*NCVE
0017         N2=1+2*NCVE
0018         N3=N2+2*M
0019   C
0020   C      COMPRUEBA SI HAY SUFICIENTE MEMORIA
0021   C
0022         IF(N3.GT.ICOMM)GOTO 100
0023   C
0024   C      LLAMADA A LA SUBRUTINA DE CONTROL DEL PROGRAMA
0025   C
0026         CALL SOFRS(NCVE,M,X,IU(1),IU(N2),IMPR,TITU)
0027         GO TO 101
0028   C
0029   C      ESCRIBE UN MENSAJE SI NO HAY SUFICIENTE MEMORIA
0030   C
0031     100 WRITE(IMPR,15)N3
0032      15 FORMAT(3/,4X,"OPTI3 SE SUPERA LA CAPACIDAD ACTUAL,N3 - ",I5)
0033   C
0034   C      DEVUELVE EL CONTROL AL PROGRAMA PRINCIPAL
0035   C
0036     101 CALL OVRTN
0037         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00102      COMMON = 00001
```

```
PAGE 0002  FTN.    1:22 AM  SAT., 16  JAN., 1982


0038          SUBROUTINE SCFRS(N,M,X,G,H,IMPR,TITU)
0039    C
0040    C     SUBRUTINA QUE CONTROLA LA EJECUCION DEL ALGORITMO
0041    C
0042          COMMON/OPGC/ERRA2,VMIF2,NMITE,DINCR
0043          COMMON//IW(1)
0044          DIMENSION X(1),G(1),H(1),TITU(30)
0045    C
0046    C     LECTURA Y ESCRITURA DE DATOS Y ENCABEZADO
0047    C
0048          WRITE (IMPR,3)TITU
0049        3 FORMAT(1H1,/,1X,69(1H*),/,5(1H*),30A2,5(1H*),/,1X,24(1H*)," GRADI
0050         *NTE CONJUGADO ",24(1H*),/,1X,69(1H*),2/)
0051          WRITE (IMPR,4)N,NMITE,ERRA2,VMIF2,DINCR
0052        4 FORMAT(5X,62(1H*),/,5X,"DIMENSION DEL V.E = ",I5,/,5X,"ITERACIONE
0053         * = ",I5,/,5X,"V.ESTIMADO F. = ",1PE15.7,/,5X,"V.EST.PASO = ",1PE1
0054         *.7,/,5X,"DIF-FIN = ",1PE15.7,/,5X,62(1H*))
0055          WRITE (IMPR,5)
0056        5 FORMAT (/,25X,"VECTOR INICIAL",/)
0057          DO 100 J=1,N
0058          WRITE (IMPR,6) J,X(J)
0059        6 FORMAT(25X,'X(',I2,') = ',1PE15.7)
0060      100 CONTINUE
0061    C
0062    C     LLAMADA A LA SUBRUTINA QUE REALIZA EL ALGORITMO
0063    C
0064          CALL PRGC(N,M,X,F,G,ERRA2,VMIF2,NMITE,INDER,H,CONTA,DINCR)
0065    C
0066    C     ESCRITURA DE LOS RESULTADOS
0067    C
0068          WRITE (IMPR,7)
0069        7 FORMAT(/,25X,"RESULTADO FINAL",/)
0070          WRITE (IMPR,8)INDER,CONTA
0071        8 FORMAT(25X,'CONVERG = ',I3,/,25X,"N.ITERAC = ",I5)
0072          WRITE (IMPR,9)F
0073        9 FORMAT(25X,'FU.OBJ = ',1PE15.7)
0074          WRITE (IMPR,10)
0075       10 FORMAT(/,25X,"VECTOR FINAL ",/)
0076          DO 200 J=1,N
0077          WRITE (IMPR,6)J,X(J)
0078      200 CONTINUE
0079          RETURN
0080          END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **    PROGRAM = 00367       COMMON = 00001
```

```
 PAGE 0003  FTN.   1:22 AM  SAT., 16  JAN., 1982


0001          SUBROUTINE FRSCG(N,M,X,F,G,ERRA2,VMIF2,XMITE,INDER,H,CONTA,DINCR)
0002          DIMENSION X(N),G(N),H(M)
0003    C
0004    C     CALCULO DEL VALOR DE LA FUNCION Y EL VECTOR GRADIENTE PARA
0005    C     EL ARGUMENTO INICIAL
0006    C
0007          CALL FUOBJ(N,X,F,G,DINCR)
0008          CONTA=0
0009          INDER=0
0010          N1=N+1
0011    C
0012    C     INICIO DEL CICLO DE ITERACION CADA N+1 ITERACCINES
0013    C
0014        1 DO 43 II=1,N1
0015    C
0016    C     GUARDA EL VALOR DE LA FUNCION
0017    C
0018          CONTA=CONTA+1
0019          ANTF=F
0020    C
0021    C     CALCULA EL CUADRADO DEL GRADIENTE Y TERMINA SI ES CERO
0022    C
0023          GRAN=0.
0024          DO 2 J=1,N
0025        2 GRAN=GRAN+G(J)*G(J)
0026          IF(GRAN)46,46,3
0027    C
0028    C     CADA VEZ QUE EL CICLO DE ITERACION SE EJECUTA EL PRIMER PASO
0029    C     SERA EN LA DIRECCION DE LA PENDIENTE DESCENDIENTE
0030    C
0031        3 IF(II-1)4,4,6
0032        4 DO 5 J=1,N
0033        5 H(J)=-G(J)
0034          GOTO 8
0035    C
0036    C     LOS VECTORES DIRECCION H SERAN LOS DETERMINADOS
0037    C     POR EL METODO DEL GRADIENTE CONJUGADO
0038    C
0039        6 DMBAA=GRAN/ANTG
0040          DO 7 J=1,N
0041        7 H(J)=DMBAA*H(J)-G(J)
0042    C
0043    C     CALCULA EL VALOR DEL VECTOR DIRECCION Y LA DERIVADA DIRECCIONAL
0044    C
0045        8 DY=0.
0046          HGRA=0.
0047          DO 9 J=1,N
0048          K=J+N
0049          H(K)=H(J)
0050          HGRA=HGRA+ABS(H(J))
0051        9 DY=DY+H(J)*G(J)
0052    C
0053    C     COMPRUEBA SI LA FUNCION DECRECE A LO LARGO DE H
0054    C
0055          IF(DY)10,42,42
```

```
0136       10 CVSR=1./HGRA
0137          FY=F
0138          ALFA=2.*(ERRA2-F)/DY
0139          DMBAA=CVSR
0140          IF(ALFA)13,13,11
0141       11 IF(ALFA-DMBAA)12,13,13
0142       12 DMBAA=ALFA
0143       13 ALFA=0.
0144    C
0145    C     GUARDA LOS VALORES DE LA FUNCION Y DERIVADA PARA LOS ANTIGUOS
0146    C     ARGUMENTOS
0147    C
0148       14 FX=FY
0149          DX=DY
0150    C
0151    C     CAMBIA EL ARGUMENTO A LO LARGO DE H
0152    C
0153          DO 15 I=1,N
0154       15 X(I)=X(I)+DMBAA*H(I)
0155    C
0156    C     CALCULO EL VALOR DE LA FUNCION Y EL GRADIENTE PARA EL NUEVO
0157    C     ARGUMENTO
0158    C
0159          CALL FUOBJ(N,X,F,G,DINCR)
0160          FY=F
0161    C
0162    C     CALCULO DE LA DERIVADA DIRECCIONAL DY PARA EL NUEVO ARGUMENTO
0163    C     TERMINA LA BUSQUEDA,SI DY ES POSITIVO.SI DY ES CERO HEMOS
0164    C     ENCONTRADO EL MINIMO
0165    C
0166          DY=0.
0167          DO 16 I=1,N
0168       16 DY=DY+G(I)*H(I)
0169          IF(DY)17,38,20
0170    C
0171    C     TERMINA LA BUSQUEDA TAMBIEN SI EL VALOR DE LA FUNCION INDICA
0172    C     QUE SE PASO EL MINIMO
0173    C
0174       17 IF(FY-FX)18,20,20
0175    C
0176    C     REPITE LA BUSQUEDA Y AUMENTA EL INTERVALO PARA ACELERAR LA
0177    C     BUSQUEDA
0178    C
0179       18 DMBAA=DMBAA+ALFA
0180          ALFA=DMBAA
0181          IF(HGRA+DMBAA-1.E10)14,14,19
0182    C
0183    C     LA TECNICA DE BUSQUEDA LINEAL INDICA QUE NO EXISTE MINIMO
0184    C
0185       19 INDEX=2
0186          RETURN
0187    C
0188    C     FIN DEL CICLO DE ITERACION
0189    C     INTERPOLACION CUBICA EN EL INTERVALO DEFINIDO A LO LARGO DE LA
0190    C     BUSQUEDA Y CALCULA EL ARGUMENTO DE X PARA QUE LA INTERPOLACION
```

```
0191   C      POLINOMIAL SEA MINIMIZADA
0192   C
0193       20 T=0.
0194       21 IF(DMBAA)22,39,22
0195       22 Z=3.*(FX-FY)/DMBAA+DX+DY
0196          ALFA=AMAX1(ABS(Z),ABS(DX),ABS(DY))
0197          DFALA=Z/ALFA
0198          DFALA=DFALA*DFALA-DX/ALFA*DY/ALFA
0199          IF(DFALA)23,27,27
0200   C
0201   C      RECUPERACION DE LOS ANTIGUOS VALORES DE LA FUNCION Y ARGUMENTOS
0202   C
0203       23 DO 24 J=1,N
0204          K=N+J
0205       24 X(J)=H(K)
0206          CALL FUOBJ(N,X,F,G,DINCR)
0207   C
0208   C      COMPRUEBA EL FALLO REPETIDO DE LA ITERACION
0209   C
0210       25 IF(INDER)47,26,47
0211       26 INDER=-1
0212          GOTO 1
0213       27 W=ALFA*SQRT(DFALA)
0214          ALFA=(DY+W-Z)*DMBAA/(DY+2.*W-DX)
0215          DO 28 I=1,N
0216       28 X(I)=X(I)+(T-ALFA)*H(I)
0217   C
0218   C      TERMINA SI EL VALOR ACTUAL DE LA FUNCION EN X ES MENOR QUE EL
0219   C      VALOR DE LA FUNCION EN EL INTERVALO FINAL.SI NO REDUCE EL
0220   C      INTERVALO ESCOGIENDO UN PUNTO FINAL IGUAL A X Y REPITE LA
0221   C      INTERPOLACION.EL PUNTO FINAL ESCOGIDO DEPENDE DEL VALOR DE
0222   C      LA FUNCION Y SU GRADIENTE EN X
0223   C
0224          CALL FUOBJ(N,X,F,G,DINCR)
0225          IF(F-FX)29,29,30
0226       29 IF(F-FY)38,38,30
0227   C
0228   C      CALCULO DE DERIVADA DIRECCIONAL
0229   C
0230       30 DFALA=0.
0231          DO 31 I=1,N
0232       31 DFALA=DFALA-G(I)*H(I)
0233          IF(DFALA)32,35,35
0234       32 IF(F-FX)34,33,35
0235       33 IF(DX-DFALA)34,38,34
0236       34 FX=F
0237          DX=DFALA
0238          T=ALFA
0239          DMBAA=ALFA
0240          GOTO 21
0241       35 IF(FY-F)37,36,37
0242       36 IF(DY-DFALA)37,38,37
0243       37 FY=F
0244          DY=DFALA
0245          DMBAA=DMBAA-ALFA
```

```
0246          GOTO 20
0247   C
0248   C      CALCULO DE LA DIFERENCIA DEL NUEVO Y ANTIGUO VECTOR ARGUMENTO
0249   C
0250       38 T=0.
0251          DO 39 J=1,N
0252          K=J+N
0253          H(K)=X(J)-H(K)
0254       39 T=T+ABS(H(K))
0255          IF(CONTA-N1)41,40,40
0256       40 IF(T-VMIF2)45,45,41
0257   C
0258   C      TERMINA SI LA FUNCION NO HA DISMINUIDO DURANTE LA ITERACION,
0259   C      GUARDA LA NORMA DEL GRADIENTE
0260   C
0261       41 IF(ANTF-F+VMIF2)19,25,42
0262       42 ANTG=GRAN
0263   C
0264   C      TERMINA SI EL NUMERO DE ITERACIONES EXCEDE A NMITE
0265   C
0266          IF(CONTA-NMITE)43,44,44
0267       43 INDER=0
0268   C
0269   C      FIN DEL CICLO DE ITERACION
0270   C
0271          GOTO 1
0272   C
0273   C      NO CONVERGE EN EL NUMERO DE ITERACIONES
0274   C
0275       44 INDER=1
0276          IF(GRAN-VMIF2)46,46,47
0277   C
0278   C      COMPRUEBA SI EL GRADIENTE ES SUFICIENTEMENTE PEQUENO
0279   C
0280       45 IF(GRAN-VMIF2)46,46,25
0281       46 INDER=0
0282       47 RETURN
0283          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00683 COMMON = 00000

```
PAGE 0001  FTN.   3:01 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM OPTI4(5)
0003   C
0004   C     CUARTO SEGMENTO DEL PROGRAMA OPRIS.BUSQUEDA DIRECTA
0005   C
0006         COMMON/COMU/LECT,IMPR,TITU(30),NCVE,X(30),MVE,ICOMM
0007         COMMON//IW(1)
0008         DO 10 I=1,ICOMM
0009      10 IW(I)=0
0010   C
0011   C     DETERMINACION DE LOS PARAMETROS DE LA MEMORIA DINAMICA
0012   C
0013         N2=1+2*NCVE
0014         N3=N2+2*NCVE
0015         N4=N3+2*NCVE
0016         N5=N4+2*NCVE
0017   C
0018   C     COMPRUEBA SI HAY MEMORIA SUFICIENTE
0019   C
0020         IF(N5.GT.ICOMM)GOTO 100
0021   C
0022   C     LLAMADA A LA SUBRUTINA QUE CONTROLA EL PROCESO
0023   C
0024         CALL SCBUS(NCVE,X,IW(1),IW(N2),IW(N3),IW(N4),IMPR,TITU)
0025         GOTO 101
0026   C
0027   C     ESCRIBE UN MENSAJE SI NO HAY SUFICIENTE MEMORIA
0028   C
0029     100 WRITE(IMPR,15)N5
0030      15 FORMAT(3/,4X,'OPTI4 SE SUPERA LA CAPACIDAD ACTUAL N5 = ',I5)
0031   C
0032   C     DEVUELVE EL CONTROL AL PROGRAMA PRINCIPAL
0033   C
0034     101 CALL OVRTN
0035         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00116      COMMON = 00001
```

```
0036          SUBROUTINE SCBUS(N,X,DS,FIT,YOP,ZPTE,IMPR,TITU)
0037    C
0038    C     SUBRUTINA QUE CONTOLA LA EJECUCION DEL ALGORITMO
0039    C
0040          COMMON/OPBD/ITES,LMITE,VINPA,VMIPA
0041          COMMON//IW(1)
0042          DIMENSION X(1),DS(1),FIT(1),YOP(1),ZPTE(1),TITU(30)
0043          WRITE(IMPR,5)TITU,N,LMITE,VINPA,VMIPA,ITES
0044          DO 10 IL=1,N
0045          DS(IL)=X(IL)
0046       10 CONTINUE
0047          CALL FUOB(DS,VAL,N)
0048          WRITE(IMPR,11)VAL
0049          DO 13 IS=1,N
0050          WRITE(IMPR,12)IS,DS(IS)
0051       13 CONTINUE
0052    C
0053    C     LLAMADA AL SUBRUTINA QUE REALIZA EL ALGORITMO
0054    C
0055          CALL SUSDI(X,OBJ,N,VINPA,VMIPA,LMITE,ITES,NITR,FIT,YOP,ZPTE,IMPR)
0056    C
0057    C     ESCRITURA DE LOS RESULTADOS
0058    C
0059          WRITE(IMPR,4)NITR,OBJ
0060          DO 7 JJ=1,N
0061          WRITE(IMPR,6)JJ,X(JJ)
0062       7  CONTINUE
0063       5  FORMAT(1H1,70(1H*),/,1X,5(1H*),30A2,5(1H*),/,1X,26(1H*)," BUSQUED
0064         * DIRECTA ",26(1H*),/,1X,70(1H*),//,5X,62(1H*),/,5X,"DIMENSION DEL
0065         *V.E = ",I5,/,5X,"ITERACIONES = ",I5,/,5X,"INT.INIC = ",
0066         *1PE15.7,/,5X,"INT.MIN = ",1PE15.7,/,5X,"C.IMPRE. = ",I4,/,5X,62(1
0067         **),/)
0068       4  FORMAT(2/,25X,21(1H*),/,25X,1H*," SOLUCION OBTENIDA ",1H*,/,25X,
0069         *21(1H*),/,25X,"ITERACIONES = ",I5,/,25X,"FU-OBJ = ",1PE15.7,/,25X
0070         *"VECTOR FINAL",/)
0071       6  FORMAT(25X,"Z(",I2,")=",1PE15.7)
0072      11  FORMAT(2/,25X,"VALORES INICIALES",/,25X,"FU-OBJ = ",1PE15.7,/,25X
0073         *"VECTOR INICIAL",/)
0074      12  FORMAT(25X,"ZI(",I2,")=",1PE15.7)
0075          RETURN
0076          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00411 COMMON = 00001

```
0077          SUBROUTINE SUSDI(TIS,ADF,N,VINPAS,DMINO,KTIMO,ITES,ITER,FIT,YOP,
0078         *ZPTE,IMPR)
0079    C
0080    C     SUBRUTINA QUE REALIZA EL ALGORITMO DE HOOKE-REEVES
0081    C
0082          DIMENSION TIS(1),FIT(1),YOP(1),ZPTE(1)
0083          DATA AALF/1.02/
0084          F(SSS)=SSS-ABS(SSS)*.0001*CORT
0085          VINPA=VINPAS
0086          DO 75 I=1,N
0087       75 ZPTE(I)=1.
0088          ITER=0
0089          CORT=1.
0090       10 CALL FUOB(TIS,ADF,N)
0091       90 TRAST=F(ADF)
0092      100 S=ADF
0093          MTERM=0
0094          DO 101 I=1,N
0095      101 FIT(I)=TIS(I)
0096          JLLA=1
0097          IF(ITES.LT.0)GOTO 15
0098          WRITE(IMPR,58)ITER
0099          WRITE(IMPR,20)(TIS(J),J=1,N)
0100          WRITE(IMPR,21)S,VINPA
0101          GOTO 15
0102       16 IF(S.LT.TRAST)GOTO 20
0103          VINPA=VINPA/2.
0104          IF(VINPA.GT.DMINO)GOTO 100
0105          IF(ITES.GE.0)WRITE(IMPR,74)
0106          IF(CORT.LT.5)GOTO 72
0107          IF(ITES.GE.0)WRITE(IMPR,77)
0108          VINPA=VINPAS
0109          CORT=0.
0110          GOTO 90
0111       20 ADF=S
0112          TRAST=F(ADF)
0113          ITER=ITER+1
0114          MTERM=MTERM+1
0115          IF(ITER.GT.KTIMO)GOTO 70
0116          IF(ITES.LT.0)GOTO 23
0117          WRITE(IMPR,58)ITER
0118          WRITE(IMPR,59)MTERM
0119          WRITE(IMPR,60)(FIT(I),I=1,N)
0120          WRITE(IMPR,61)ADF,VINPA
0121       23 DO 21 I=1,N
0122          YOP(I)=TIS(I)
0123          TIS(I)=FIT(I)
0124       21 FIT(I)=FIT(I)+AALF*(FIT(I)-YOP(I))
0125          CALL FUOB(FIT,FIA,N)
0126          S=FIA
0127          IF(ITES.NE.1)GOTO 22
0128          WRITE(IMPR,66)(FIT(I),I=1,N)
0129          WRITE(IMPR,61)FIA,VINPA
0130       22 JLLA=2
0131          GOTO 15
```

```
PAGE 0004  DUSDI  3:01 AM  SAT., 16  JAN., 1982

0132       30 IF(S.LT.TRAST)GOTO 20
0133          GOTO 100
0134       15 DO 19 K=1,N
0135          FITANT=FIT(K)
0136          PASO=FITANT*.05
0137          IF(PASO.EQ.0)PASO=.05
0138          PASO=SIGN(PASO*VINPA,ZPTE(K))
0139          FIT(K)=FITANT+PASO
0140          CALL FUOB(FIT,PIA,N)
0141          IF(ITES.EQ.1)WRITE(IMPR,62)JLLA,K,PIA,(FIT(L),L=1,N)
0142          IF(PIA.LT.S)GOTO 37
0143          ZPTE(K)=-ZPTE(K)
0144          FIT(K)=FITANT-PASO
0145          CALL FUOB(FIT,PIA,N)
0146          IF(ITES.EQ.1)WRITE(IMPR,62)JLLA,K,PIA,(FIT(L),L=1,N)
0147          IF(PIA.LT.S)GOTO 37
0148          FIT(K)=FITANT
0149          GOTO 19
0150       37 S=PIA
0151       19 CONTINUE
0152          GOTO(16,26) JLLA
0153       70 IF(ITES.GE.0)WRITE(IMPR,71)
0154       72 DO 73 I=1,N
0155       73 TIS(I)=FIT(I)
0156          IF(ITES.GE.0)WRITE(IMPR,67)ITER
0157          RETURN
0158       50 FORMAT(5X,' ITERACION ',I5,2/)
0159       60 FORMAT(5X,' PUNTO ',/,(10X,E15.7,5X,E15.7,5X,E15.7),/)
0160       66 FORMAT(5X,' TRAYEC ',/,(10X,E15.7,5X,E15.7,5X,E15.7),/)
0161       61 FORMAT(6X,'FUN-OBS',E15.7,6X,'INTERVALO',E15.7)
0162       62 FORMAT(1X,I2,3X,I2,2X,'FUN-OBJ',E15.7,/ 3X,'ENTRADA',/,(10X,E15.7,
0163         *5X,E15.7,5X,E15.7),/)
0164       67 FORMAT(5X,' NUMERO TOTAL DE ITERACIONES= ',I5)
0165       71 FORMAT(5X,' EL NUMERO DE ITERACIONES ES MAYOR QUE EL LIMITE ')
0166       74 FORMAT(5X,' BUSQUEDA TERMINADA.EL INTERVALO ES MENOR QUE EL DADO')
0167       77 FORMAT(5X,' INICIO CON EL PUNTO INICIAL E INTERVALO VINPA ')
0168          END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **    PROGRAM = 00031        COMMON = 00000
```

```
PAGE 0001  FTN.   2:06 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM OPTI5(5)
0003   C
0004   C     QUINTO SEGMENTO DEL PROGRAMA OPRIS.DIRECCIONES CONJUGADAS
0005   C
0006         COMMON/COMU/LECT,IMPR,TITU(30),NCVE,X(30),NVE,ICOMM
0007         COMMON//IW(1)
0008         DO 10 I=1,ICOMM
0009      10 IW(I)=0
0010   C
0011   C     DETERMINACION DE LOS INDICES DE LA MEMORIA DINAMICA
0012   C
0013         NW=NCVE*(NCVE+3)
0014         N2=I+2*NW
0015   C
0016   C     COMPRUEBA SI HAY SUFICIENTE MEMORIA
0017   C
0018         IF(N2.GT.ICOMM)GOTO 100
0019   C
0020   C     LLAMA A LA SUBRUTINA QUE CONTROLA LA EJECUCION
0021   C
0022         CALL SCPOW(X,IW(1),TITU,IMPR,NCVE,NW)
0023         GOTO 101
0024   C
0025   C     ESCRIBE UN MENSAJE SI NO HAY SUFICIENTE MEMORIA
0026   C
0027     100 WRITE(IMPR,15)N2
0028      15 FORMAT(3/,4X,"OPTI5 SE SUPERA LA CAPACIDAD ACTUAL N2 - ",I5)
0029   C
0030   C     DEVUELVE EL CONTROL AL PROGRAMA PRINCIPAL
0031   C
0032     101 CALL OVRTN
0033         END


 FTN4 COMPILER) HP92060-16092 REV. 1926 (790430)


 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00094       COMMON = 00001
```

```
0034          SUBROUTINE SCPOU(X,W,TITU,IMPR,N,NW)
0035    C
0036    C     SUBRUTINA QUE CONTROLA LA EJECUCION DE ALGORITMO
0037    C
0038          COMMON/OPDC/INESC,NLITE,VMAPA,VLCV(30)
0039          COMMON//IW(1)
0040          DIMENSION X(1), W(1),TITU(30)
0041    C
0042    C     LLAMADA A LA SUBRUTINA QUE REALIZA EL ALGORITMO
0043    C
0044          CALL POUDC(X,E,N,EF,VMAPA,INESC,NLITE,W,IMPR,NW,TITU,ITERC,
0045         *NFCC)
0046    C
0047    C     ESCRITURA DE LOS RESULTADOS
0048    C
0049          WRITE(IMPR,1)
0050        1 FORMAT(25X,21(1H#),/,28X,"RESULTADO FINAL",/,25X,21(1H#),/,
0051         *25X,"VECTOR FINAL",/)
0052          DO 100 J=1,N
0053          WRITE (IMPR,2)J,X(J)
0054        2 FORMAT(25X,"X(",I2,")=",1PE15.7)
0055      100 CONTINUE
0056          WRITE(IMPR,3)ITERC,EF,NFCC
0057        3 FORMAT(/,25X,"ITERACIONES =",I5,/,25X,"FUN-OBJ =",1PE16.7,/,25X,
0058         *"NUMERO DE C. DE F. =",I5)
0059          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00179 COMMON = 00001

```
0060          SUBROUTINE POUDC(X,VLCV,N,EF,VMAPA,INES,NLITE,W,IMPR,NL,TITU,
0061         *ITERC,NFCC)
0062          DIMENSION X(N), W(NW), VLCV(N),TITU(30)
0063    C
0064    C     ESCRITURA DEL ENCABEZADO Y DATOS ESPECIFICOS DEL ALGORITMO
0065    C
0066          WRITE (IMPR,1)TITU,N,NLITE,VMAPA
0067        1 FORMAT(1H1,/,1X,70(1H*),/,1X,4(1H*),34A2,2X,4(1H*),/,1X,23(1H*),
0068         *" DIRECCIONES CONJUGADAS ",23(1H*),/,1X,70(1H*),2/,5X,62(1H*),
0069         */,5X,"NUMERO DE VARIABLES = ",I5,/,5X,"ITERACIONES = ",6X,I5,/,
0070         *5X,"INTERVALO INICIAL = ",1PE16.7,/,5X,,2(1H*),//,
0071         *5X,"VECTOR INICIAL",11X,"APROX.DE LAS VARIABLES",/,)
0072          DO 125 JJK=1,N
0073          WRITE(IMPR,2)JJK,X(JJK),JJK,VLCV(JJK)
0074      125 CONTINUE
0075        2 FORMAT(5X,"X(",I2,")=",1PE16.7,3X,"VLCV(",I2,")=",1PE16.7)
0076          AGMDT=0.1*VMAPA
0077          BERO=0.05/VMAPA
0078          JJ=N*(N+1)
0079          JJJ=JJ+N
0080          K=N+1
0081          NFCC=1
0082          IND=1
0083          INN=1
0084          DO 4 I=1,N
0085          W(I)=VMAPA
0086          DO 4 J=1,N
0087          W(K)=0.
0088          IF(I-J)4,3,4
0089        3 W(K)=ABS (VLCV(I))
0090        4 K=K+1
0091          ITERC=1
0092          LRCIBA=2
0093          CALL FUOB(X,F,N)
0094          FMLAT=2 *ABS (F)
0095        5 LKONE=1
0096          FP=F
0097          SUM=0.
0098          IXP=JJ
0099          DO 6 I=1,N
0100          IXP=IXP+1
0101        6 W(IXP)=X(I)
0102          JNIBR=N+1
0103          KKENI=1
0104        7 PORE=W(KKENI)
0105          CPOT=PORE*BERO
0106          CIEAR=AMIN1 (AGMDT,0.1*PORE)
0107          CIEAR=AMAX1(CIEAR,20.*CPOT)
0108          TYURE=10.*CIEAR
0109          GO TO (70,70,71),LKONE
0110       70 DL=0.
0111          D=CIEAR
0112          FUATE=F
0113          IS=5
0114          FA=FUATE
```

```
0115          DA=DL
0116        8 DB=D-DL
0117          DL=D
0118       38 K=JHIDR
0119          DO 9 I=1,N
0120          X(I)=X(I)+DD*U(K)
0121        9 K=K+1
0122          CALL FUOB(X,F,N)
0123          NFCC=NFCC+1
0124          GO TO (10,11,12,13,14,96) ,IS
0125       14 IF(F-FA)15,16,24
0126       16 IF (ABS (D)-PURE) 17,17,18
0127       17 D=D+D
0128          GO TO 6
0129       18 URITE (IMPR,19)
0130       19 FORMAT(5X,"EL MAYOR VALOR NO ALTERA LA FUNCION")
0131          GO TO 20
0132       15 FB=F
0133          DB=D
0134          GO TO 21
0135       24 FB=FA
0136          DB=DA
0137          FA=F
0138          DA=D
0139       21 GO TO (83,23),LRGIDA
0140       23 D=DB+DB-DA
0141          IS=1
0142          GO TO 8
0143       83 D=0.5*(DA+DB-(FA-FB)/(DA-DB))
0144          IS=4
0145          IF((DA-D)*(D-DB))25,8,8
0146       25 IS=1
0147          IF(ABS (D-DB)-TYURE)8,8,26
0148       26 D=DB+SIGN (TYURE,DB-DA)
0149          IS=1
0150          TYURE=TYURE+TYURE
0151          AGMDT=AGMDT+AGMDT
0152          IF(AGMDT.GE 1.0E+37)AGMDT=1.0E+37
0153          IF(TYURE-PURE)8,8,27
0154       27 TYURE=PURE
0155          GO TO 8
0156       13 IF(F-FA)28,23,23
0157       28 FC=FB
0158          DC=DB
0159       29 FB=F
0160          DB=D
0161          GO TO 30
0162       12 IF(F-FB)28,28,31
0163       31 FA=F
0164          DA=D
0165          GO TO 30
0166       11 IF(F-FB)32,10,10
0167       32 FA=FB
0168          DA=DB
0169          GO TO 29
```

```
0170      71 DL=1.
0171         TYURE=5.
0172         FA=FP
0173         DA=-1.
0174         FB=FOANT
0175         DB=0.
0176         D=1.
0177      10 FC=F
0178         DC=D
0179      30 A=(DB-DC)*(FA-FC)
0180         B=(DC-DA)*(FB-FC)
0181         IF((A+B)*(DA-DC))33,33,34
0182      33 FA=FB
0183         DA=DB
0184         FB=FC
0185         DB=DC
0186         GO TO 20
0187     34  D=0.5*(A*(DB+DC)+B*(DA+DC))/(A+B)
0188         DI=DB
0189         FI=FB
0190         IF(FB-FC)44,44,43
0191     43  DI=DC
0192         FI=FC
0193     44  GOTO(86,86,85),LKONE
0194     85  LKONE=2
0195         GOTO 45
0196     86  IF(ABS(D-DI)-CPDT)41,41,93
0197     93  IF(ABS(D-DI)-0.03*ABS(D))41,41,45
0198     45  IF((DA-DC)*(DC-D))47,46,46
0199     46  FA=FB
0200         DA=DB
0201         FB=FC
0202         DB=DC
0203         GOTO 25
0204     47  IS=2
0205         IF ((DB-D)*(D-DC)) 48,9,9
0206      48 IS=3
0207         GO TO 9
0208      41 F=FI
0209         D=DI-DL
0210         DD=SQRT ((DC-DB)*(DC-DA)*(DA-DB)/(A+B))
0211         DO 49 I=1,N
0212         X(I)=X(I)+D*W(JNIDR)
0213         W(JNIDR)=DD*W(JNIDR)
0214      49 JNIDR=JNIDR+1
0215         W(KKENI)=W(KKENI)/DD
0216         KKENI=KKENI+1
0217         IF(INESC-1)51,50,51
0218      50 WRITE(IMPR,52)ITERC,NFCC,F
0219      52 FORMAT(/,5X,"ITERACION",I5,2X,"C.DE F.",I5,/,5X,"FUN-OBJ, = ",
0220        *1PE16.7)
0221         WRITE(IMPR,500)(I,X(I),I=1,N)
0222    500  FORMAT(5X,"X(",I2,")=",1PE16.7)
0223         GO TO(51,53),INESC
0224      51 GO TO (55,30),LKONE
```

```
0225      55 IF (FUNTE-F-SUM) 94,95,95
0226      95 SUM=FUNTE-F
0227         JIL=KKENI
0228      94 IF (JNIDR-JJ) 7,7,84
0229      84 GO TO (92,72),IND
0230      92 FUNNT=F
0231         IS=6
0232         IXF=JJ
0233         DO 59 I=1,N
0234         IXF=IXF+1
0235      59 W(IXF)=X(I)-U(IXF)
0236         DD=1.
0237         GO TO 58
0238      96 GO TO (112,97),IND
0239     112 IF(FP-F) 37,37,91
0240      91 D=2.*(FP+F-2.*FUANT)/(FP-F)**2
0241         IF (D*(FP-FUANT-SUM)**2-SUM) 87,37,37
0242      87 J=JIL+N+1
0243         IF (J-JJ) 60,60,61
0244      60 DO 62 J=JJ,JJ
0245         K=I-N
0246      62 W(K)=W(I)
0247         DO 97 I=JIL,N
0248      97 W(I-1)=W(I)
0249      61 JNIDR=JNIDR-N
0250         LKONE=3
0251         K=JNIDR
0252         IXF=JJ
0253         ABC=0.
0254         DO 67 I=1,N
0255         IXF=IXF+1
0256         W(K)=U(IXF)
0257         IF (ABC-ABS (W(K)/VLCV(I))) 66,67,67
0258      66 ABC=ABS (W(K)/VLCV(I))
0259      67 K=K+1
0260         AGNDT=1.
0261         W(N)=VKAPA/ABC
0262         KKENI=N
0263         GO TO 7
0264      37 IXF=JJ
0265         ABC=0.
0266         F=FUANT
0267         DO 99 I=1,N
0268         IXF=IXF+1
0269         X(I)=X(I)-U(IXF)
0270         IF(ABC*ABS (VLCV(I))-ABS (W(IXF))) 98,99,99
0271      98 ABC=ABS (W(IXF)/VLCV(I))
0272      99 CONTINUE
0273         GO TO 72
0274      38 ABC=ABC*(1.+DI)
0275         GO TO (72,106),IND
0276      72 IF(INSEC-2)53,50,50
0277      53 GO TO (109,88),IND
0278     109 IF (ABC - 0.1) 20,20,76
0279      76 IF(F-FP)35,70,70
```

```
0290      78 WRITE (IMPR,80)
0291      80 FORMAT(5X,*APROXIMACION LIMITADA POR ERROR EN F*)
0292         GO TO 20
0293      90 IND=1
0294      35 AGMDT=0.4*SQRT(ABS(FP-F))
0295         IF(AGMDT.GE.1.0E+37)AGMDT=1.0E+37
0296         LRGIDA=1
0297     100 ITERC=ITERC+1
0298         IF(ITERC-NLITE)5,5,81
0299      81 WRITE (IMPR,82) NLITE
0300      82 FORMAT(1X,*SE SUPERA EL NUMERO DE ITERACIONES = *,I5)
0301         IF(F-FMLAT)20,20,110
0302     110 F=FMLAT
0303         DO 111 I=1,N
0304         JJJ=JJJ+1
0305     111 X(I)=U(JJJ)
0306         GO TO 20
0307     101 JIL=1
0308         FP=FMLAT
0309         IF(F-FMLAT)105,78,104
0310     104 JIL=2
0311         FP=F
0312         F=FMLAT
0313     105 IXP=JJ
0314         DO 113 I=1,N
0315         IXP=IXP+1
0316         K=IXP+N
0317         GO TO (114,115),JIL
0318     114 W(IXP)=U(K)
0319         GO TO 113
0320     115 W(IXP)=X(I)
0321         X(I)=U(K)
0322     113 CONTINUE
0323         JIL=2
0324         GO TO 92
0325     106 IF(ABS-0.1) 20,20,107
0326      20 EF=F
0327         RETURN
0328     107 IND=1
0329         GO TO 35
0330         END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 02053 COMMON = 00000

```
PAGE 0001  FTN.   3:23 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM OPTI6(5)
0003   C
0004   C     SEXTO SEGMENTO DEL PROGRAMA OPRIS.COMPLEX.SOLO SE UTILIZA SI HAY
0005   C     RESTICCIONRES
0006   C
0007         COMMON/COMU/LECT,IMPR,TITU(30),N,ZZ(30),MVE,ICOMM
0008         COMMON/OPCOM/M,K,IX,ITMAX,IC,IPRINT,ALPHA,BETA,GAMMA,DELTA
0009         COMMON//IW(1)
0010         INTEGER GAMMA
0011         DO 10 I=1,ICOMM
0012      10 IW(I)=0
0013   C
0014   C     DETERMINACION DE LOS INDICES DE LA MEMORIA DINAMICA
0015   C
0016         N2=1+2*K*M
0017         N3=N2+2*K*N
0018         N4=N3+2*K
0019         N5=N4+2*M
0020         N6=N5+2*M
0021         N7=N6+2*N
0022   C
0023   C     COMPRUEBA SI HAY MEMORIA SUFICIENTE
0024   C
0025         IF(N7.GT.ICOMM)GOTO 100
0026   C
0027   C     LLAMADA A LA SUBRUTINA QUE CONTROLA EL PROGRAMA
0028   C
0029         CALL SCCOM(IW(1),IW(N2),IW(N3),IW(N4),IW(N5),IW(N6))
0030         GOTO 101
0031   C
0032   C     ESCRIBE UN MENSAJE SI NO HAY MEMORIA SUFICIENTE
0033   C
0034     100 WRITE(IMPR,2)N7
0035       2 FORMAT(3X,'OPTI6 SE SUPERA LA CAPACIDAD ACTUAL.N7= ',I5)
0036     101 CALL OVRTN
0037         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00131       COMMON = 00001
```

```
0038          SUBROUTINE SCCOM(X,R,F,G,H,XC)
0039          COMMON/CONU/LECT,IMPR,TITU(30),N,ZZ(30),NVE,ICONM
0040          COMMON/OPCON/N,K,IX,ITMAX,IC,IPRINT,ALPHA,BETA,GAMMA,DELTA
0041          COMMON//IW(1)
0042          INTEGER GAMMA
0043          DIMENSION X(1,1),R(1,1),F(1),G(1),H(1),XC(1)
0044    C
0045    C     LECTURA DEL VECTOR INICIAL QUE CUMPLER LAS RESTICCIONES
0046    C
0047          DO 101 J=1,N
0048          X(1,J)=ZZ(J)
0049      101 CONTINUE
0050    C
0051    C     GENERACION ALEATORIA DE LAS PUNTOS
0052    C
0053          DO 100  II=2,K
0054          DO 100  JJ=1,N
0055          CALL ALEAT(IX,IY,RATI)
0056          R(II,JJ)=RATI
0057          IX=IY
0058      100 CONTINUE
0059    C
0060    C     ESCRITURA DE ENCABEZADO Y PARAMETROS
0061    C
0062          WRITE(IMPR,10)TITU
0063       10 FORMAT(1H1,/,1X,69(1H*),/,5(1H*),30A2,5(1H*),/,1X,30(1H*),
0064         *" COMPLEX " 30(1H*),/,1X,69(1H*),2/)
0065          WRITE(IMPR,18)
0066       18 FORMAT(//,20X,"PARAMETROS")
0067          WRITE(IMPR,11)N,M,K,ITMAX,IC,ALPHA,BETA,GAMMA,DELTA
0068       11 FORMAT (//,10X,"N=",I2,3X,"M = ",I2,3X,"K= ",I2,2X,"ITMAX =",
0069         1I5,2X," IC =",I2,//,2X," ALPHA =",F5.3,3X," BETA =",E14.7,3X,
0070         *"GAMMA= ",I4,3X,"DELTA = ",E14.7)
0071          WRITE(IMPR,12)
0072       12 FORMAT (//,20X,"NUMEROS ALEATORIOS TOMADOS")
0073          DO 200  J=2,K
0074          WRITE(IMPR,13)(J,I,R(J,I),I=1,N)
0075       13 FORMAT(/,3(2X,"R(",I2,",",I2,") = ",E14.7))
0076      200 CONTINUE
0077    C
0078    C     EJECUCION DEL ALGORITHO
0079    C
0080          CALL ALCOM(N,M,K,ITMAX,ALPHA,BETA,GAMMA,DELTA,X,R,F,IT,IEV2,IMPR,
0081         1G,H,XC,IPRINT,NX,NY,NZ)
0082    C
0083    C     ESCRITURA DE LOS RESULTADOS
0084    C
0085          IF(IT-ITMAX)20,20,30
0086       20 WRITE(IMPR,14)IT,F(IEV2)
0087       14 FORMAT(1H1,10X,"SOLUCION OBTENIDA EN LA ITERACION",2X,I6,
0088         *///,10X," VALOR FINAL DE LA FUNCION = ",E14.7)
0089          WRITE(IMPR,15)
0090       15 FORMAT(//,10X,"  VALORES FINALES DE X")
0091          DO 300  J=1,N
0092          WRITE(IMPR,16)J,X(IEV2,J)
```

```
PAGE 0003  SCCOM  3:23 AM  SAT., 16  JAN., 1982


0093       16 FORMAT(/,18X,"X(",I2,") = ",E14.7)
0094      300 CONTINUE
0095          GOTO 999
0096       30 WRITE(IMPR,17)ITMAX
0097    C
0098    C     MENSAJE PARA CUANDO NO CONVERGE EN EL NUMERO DE ITERACIONES DADO
0099    C
0100       17 FORMAT(///,20X,"EL NUMERO DE ITERACIONES ES MAYOR DE ",I6,10X,
0101         1" PROGRAMA TERMINADO")
0102          GOTO 20
0103      999 RETURN
0104          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00547       COMMON = 00001
```

```
0105          SUBROUTINE ALCOM(N,M,K,ITMAX,ALPHA,BETA,GAMMA,DELTA,X,R,F,IT,
0106         1IEV2,INPR,G,H,XC,IPRINT,NX,NY,NZ),ALGORITMO COMPLEX
0107    C
0108    C     SUBRUTINA QUE EFECTUA EL ALGORITMO COMPLEX
0109    C
0110          DIMENSION X(K,M),R(K,N),F(K),G(M),H(M),XC(N)
0111          INTEGER GAMMA
0112          IT=1
0113          ICODI=0
0114          IF(M-N)20,20,10
0115       10 ICODI=1
0116       20 CONTINUE
0117          DO 40 II=2,K
0118          DO 30 J=1,N
0119       30 X(II,J)=0.0
0120       40 CONTINUE
0121    C
0122    C     DETERMINA LOS PUNTOS COMPLEJOS Y COMPRUEBA LAS RESTRICCIONES
0123    C
0124          DO 65 II=2,K
0125          DO 50  J=1,N
0126          I=II
0127          CALL RESTR (N,M,K,X,G,H,I)
0128          X(II,J)=G(J)+R(II,J)*(H(J)-G(J))
0129       50 CONTINUE
0130          K1=II
0131          CALL CONRE (N,M,K,X,G,H,I,ICODI,XC,DELTA,K1)
0132          IF(II-2)51,51,55
0133       51 IF(IPRINT)52,65,52
0134       52 WRITE (INPR,18)
0135       18 FORMAT (//,20X,"COORDENADAS COMPLEJAS INICIALES")
0136          IO=1
0137          WRITE (INPR,19)(IO,J,X(IO,J),J=1,N)
0138       19 FORMAT (/,3(2X,"X(",I2,",",I2,") = ",E14.7))
0139       55 IF(IPRINT)56,65,56
0140       56 WRITE(INPR,619)(II,J,X(II,J),J=1,N)
0141       65 CONTINUE
0142          K1=K
0143          DO 70 I=1,K
0144          CALL FUNCI (N,M,K,X,F,I,NX,NY,NZ)
0145       70 CONTINUE
0146          LCONT=1
0147          IA=0
0148    C
0149    C     DETERMINA EL PUNTO EN EL QUE LA FUNCION TIENE MENOR VALOR
0150    C
0151          IF(IPRINT)72,80,72
0152       72 WRITE (INPR,21)
0153       21 FORMAT(/,20X,"VALORES DE LA FUNCION")
0154          WRITE (INPR,22) (J,F(J),J=1,K)
0155       22 FORMAT (/,3(2X,"F(",I2,") = ",E14.7))
0156       80 IEV1=1
0157          DO 100 ICM=2,K
0158          IF(F(IEV1)-F(ICM))100,100,90
0159       90 IEV1=ICM
```

```
0160      100 CONTINUE
0161    C
0162    C       DETERMINA EL PUNTO EN EL QUE LA FUNCION TIENE EL MAYOR VALOR
0163    C
0164            IEV2=1
0165            DO 120 ICM=2,K
0166            IF(F(IEV2)-F(ICM))110,110,120
0167      110   IEV2=ICM
0168      120   CONTINUE
0169    C
0170    C       COMPRUEBA EL CRITERIO DE CONVERGENCIA
0171    C
0172            IF(F(IEV2)-(F(IEV1)+BETA))140,130,130
0173      130   LCONT=1
0174            GOTO 150
0175      140   LCONT=LCONT+1
0176            IF(LCONT-GAMMA)150,240,240
0177    C
0178    C       CAMBIA EL PUNTO EN EL QUE TINE LA FUNCION EL MENOR VALOR
0179    C
0180      150   CALL CENTR(N,M,K,IEV1,I,XC,X,K1)
0181            DO 160  JJ=1,N
0182      160   X(IEV1,JJ)=(1.0+ALPHA)*(XC(JJ))-ALPHA*(X(IEV1,JJ))
0183            I=IEV1
0184            CALL COMRE(N,M,K,X,G,H,I,ICODI,XC,DELTA,K1)
0185            CALL FUNCI (N,M,K,X,F,I,NX,NY,NZ)
0186    C
0187    C       INTRODUCE EL NUEVO PUNTO SI SE REPITE COMO EL MENOR VALOR DE LA
0188    C       FUNCION
0189    C
0190      170   IEV2=1
0191            DO 190 ICM=2,K
0192            IF(F(IEV2)-F(ICM))190,190,180
0193      180   IEV2=ICM
0194      190   CONTINUE
0195            IF(IEV2-IEV1)220,200,220
0196      200   DO 210  JJ=1,N
0197            X(IEV1,JJ)=(X(IEV1,JJ)+XC(JJ))/2.0
0198      210   CONTINUE
0199            I=IEV1
0200            CALL COMRE (N,M,K,X,G,H,I,ICODI,XC,DELTA,K1)
0201            CALL FUNCI (N,M,K,X,F,I,NX,NY,NZ)
0202            GOTO 170
0203      220   CONTINUE
0204            IF(IPRINT)230,228,230
0205      230   WRITE (IMPR,023)IT
0206      023   FORMAT(/,20X,"NUMERO DE LA ITERACION",I5)
0207            WRITE (IMPR,024)
0208      024   FORMAT(/,20X,"COORDENADAS DEL PUNTO CORREGIDO")
0209            WRITE(IMPR,019) (IEV1,JC,X(IEV1,JC),JC=1,N)
0210            WRITE (IMPR,021)
0211            WRITE (IMPR,022) (I,F(I),I=1,K)
0212            WRITE (IMPR,025)
0213      025   FORMAT(/,20X,"COORDENADAS DEL PUNTO MEDIO")
0214            WRITE (IMPR,026) (JC,XC(JC), JC=1,N)
```

```
PAGE 0006  ALCOH  3:23 AM  SAT., 16  JAN., 1982


0215      226 FORMAT(/,4(2X,"X(",I2,",C) = ",E14.7,))
0216    . 228 IT=IT+1
0217          IF(IT-ITMAX)80,80,240
0218      240 RETURN
0219          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00993       COMMON = 00000
```

```
PAGE 0007  FTN.   3:23 AM  SAT., 16  JAN., 1982


0220          SUBROUTINE COMREC(N,M,K,X,G,H,I,ICODI,XC,DELTA,K1)
0221          DIMENSION X(K,M),G(M),H(M),XC(N)
0222       10 KT=0
0223          CALL RESTR (N,M,K,X,G,H,I)
0224    C
0225    C     COMPRUEBA LAS RECTRICCIONES EXPLICITAS
0226    C
0227          DO 50 J=1,N
0228          IF(X(I,J)-G(J))20,20,30
0229       20 X(I,J)=G(J)+ DELTA
0230          GOTO 50
0231       30 IF(H(J)-X(I,J))40,40,50
0232       40 X(I,J)=H(J)-DELTA
0233       50 CONTINUE
0234          IF(ICODI)110,110,60
0235    C
0236    C     COMPRUEBA LAS RECTRICCIONES IMPLICITAS
0237    C
0238       60 NN=M+1
0239          DO 100  J=NN,M
0240          CALL RESTR (N,M,K,X,G,H,I)
0241          IF(X(I,J)-G(J))80,70,70
0242       70 IF(H(J)-X(I,J))80,100,100
0243       80 IEV1=I
0244          KT=1
0245          CALL CENTR(N,M,K,IEV1,I,XC,X,K1)
0246          DO 90  JJ=1,M
0247          X(I,JJ)=(X(I,JJ)+XC(JJ))/2.0
0248       90 CONTINUE
0249      100 CONTINUE
0250          IF(KT)110,110,10
0251      110 RETURN
0252          END


FTN4 COMPILER) HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00264       COMMON = 00000
```

```
PAGE 0003  FTN.   3:23 AM  SAT., 16  JAN., 1982

0253          SUBROUTINE CENTR(N,M,K,IEV1,I,XC,X,K1),CALCULA EL CENTRO
0254          DIMENSION X(K,M),XC(N)
0255          DO 20  J=1,N
0256          XC(J)=0.0
0257          DO 10  IL=1,K1
0258       10 XC(J)=XC(J)-X(IL,J)
0259          RK=K1
0260       20 XC(J)=(XC(J)-X(IEV1,J))/(RK-1.0)
0261          RETURN
0262          END


FTN4 COMPILER) HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00123       COMMON = 00000
```

```
PAGE 0001  FTN.   3:35 AM  SAT., 16  JAN., 1982


)0001   FTN4
)0002         SUBROUTINE ALEAT(IX,IY,YFL)
)0003   C
)0004   C    GENERACION ALEATORIA DE VALORES ENTRE 0 Y 1
)0005   C
)0006         IY=IX*899
)0007         IF(IY)5,6,6
00008       5 IY=IY+32767+1
00009       6 YFL=IY/32767.
00010         RETURN
00011         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00034      COMMON = 00000
```

```
PAGE 0002  FTN.   3:35 PM  SAT., 16  JAN., 1982


0012          SUBROUTINE GRADI(Z,II,J,NVA,PTDE,DINCR)
0013    C
0014    C     DETERMINACION DE LAS DERIVADAS PARCIALES DE LA I-ESIMA ECUACION
0015    C     SI J=0 DEVUELVE EL VALOR DE LA I-ESIMA ECUACION
0016    C
0017          DIMENSION Z(NVA)
0018          IF(J.NE.0)GOTO 5
0019          PTDE=FUNC(Z,II)
0020          RETURN
0021    C
0022    C     CALCULO DE LAS DERIVADAS PARCIALES DE FORMA NUMERICA
0023    C
0024        5 Y1=FUNC(Z,II)
0025          TEMP=Z(J)
0026          H=DINCR*ABS(TEMP)*10
0027          IF(H.LT.DINCR)H=DINCR
0028          Z(J)=TEMP+H
0029          Y2=FUNC(Z,II)
0030          Z(J)=TEMP
0031          PTDE=(Y2-Y1)/H
0032          RETURN
0033          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00113      COMMON = 00000
```

```
00034          SUBROUTINE FUOB(X,VAL,N)
00035    C
00036    C     FUNCION OBJETIVO
00037    C
00038          DIMENSION X(1)
00039          VAL=0.0
00040          DO 1 I=1,N
00041          VAL=VAL+FUNC(X,I)
00042      1   CONTINUE
00043          RETURN
00044          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00034 COMMON = 00000

```
PAGE 0004  FTN.   3:35 AM  SAT., 16  JAN., 1982


0045          SUBROUTINE FUOBJ(N,Z,VAL,GRAD,DINCR)
0046    C
0047    C     FUNCION OBJETIVO
0048    C
0049          DIMENSION Z(N),GRAD(N)
0050          VAL=0.0
0051          DO 50 II=1,N
0052       50 GRAD(II)=0.0
0053          DO 100 I=1,N
0054          CALL GRADI(Z,I,0,N,PTDE,DINCR)
0055      100 VAL=VAL+ABS(PTDE)
0056          DO 200 J=1,N
0057          DO 200 I=1,N
0058          CALL GRADI(Z,I,J,N,PTDE,DINCR)
0059      200 GRAD(J)=GRAD(J)+PTDE
0060          RETURN
0061          END


FTN4 COMPILER) HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00111      COMMON = 00000
```

```
PAGE 0005  FTN.   3:35 AM  SAT., 16  JAN., 1982


0062          SUBROUTINE FUNCION,M,K,X,F,I,NX,NY,NZ),FUNCION OBJETIVO
0063          DIMENSION  X(K,M),F(K)
0064    C
0065    C     FUNCION OBJETIVO
0066    C
0067          DO 100 I=NX,NY,NZ
0068          VALN=VALN+FUNC(X,I)
0069      100 CONTINUE
0070          RETURN
0071          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00043       COMMON = 00000
```

APENDICE D

```
PAGE 0001  FTN.   4:43 AM  SAT., 16  JAN., 1982


00001   FTN4
00002         BLOCK DATA OPTBD
00003         COMMON/COMUN/LECT,IMPR,TITU(30),ICOMN,NIVE,NVE,NREST,NP,NTP
00004         COMMON/OPMVP/IFUNC,NOP1,MOP1,FTR,VEVE,VXES,VXEM,CARD,CDFI,ERRA,
00005        *ERFFO,VMDM,ERRO,NMITE,INPRE,INDER,ISR,IST
00006         COMMON/OPROC/KOUNT,IINPR,ITPRO,NNTER,ICPE,ICTPA,YLED,EPS(50)
00007         COMMON/VECES/Z(50)
00008         COMMON/DATOS/DATOP(3,361),DATOA(360),IB,IC,ID,IEL,ISL(80),ZP(80),
00009        *ISD,ICC,IDD,IEEL,NVAL,IAN,ISN,ICN,IDN,IEEN
00010         COMMON//IW(9000)
00011         DATA ICOMM/9000/
00012         END


  FTN4 COMPILER> HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **

 BLOCK COMMON COMUN   SIZE =  00068

 BLOCK COMMON OPMVP   SIZE =  00028

 BLOCK COMMON OPROC   SIZE =  00100

 BLOCK COMMON VECES   SIZE =  00100

 BLOCK COMMON DATOS   SIZE =  03140
```

```
PAGE 0001  FTN.   4:46 AM  SAT., 16  JAN., 1982

0001   FTN4
0002         PROGRAM CIOPT
0003   C
0004   C     PROGRAMA PARA LA REALIZACION DE LOS CICLOS DE OPTIMIZACION.
0005   C     OPTIMIZACION CICLICA.
0006   C
0007         COMMON/COMUN/LECT,INPR,TITU(30),ICOMM,NCVE,NVE,NREST,NP,NTP
0008         COMMON/OPMVP/IFUAC,NOP1,MOP1,FTR,VEVE,VXES,VXEN,CMXP,CDF1,ERRA,
0009        *ERRFO,VMDM,ERPG,NMITE,INPRE,INDER,IER,IFT
0010         COMMON/OPROG/KOUNT,IINPR,ITPRO,NMTER,ICPE,ICTPA,YLED,EPS(50)
0011         COMMON/VECES/Z(50)
0012         COMMON/DATOS/DATOP(3,361),DATOA(360),ID,IC,ID,IEL,ISL(80),ZP(80),
0013        *IBS,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IFEN
0014         COMMON//IU(1)
0015         DIMENSION IPAR(5),IOP(3),IP1(3),IP2(3),INP(16),ZZ(50)
0016         DATA IP1/2HOP,2HF1,2H1 /,IP2/2HOP,2HF1,2H2 /
0017         CALL SMPAR(IPAR)
0018   C
0019   C     RECOGIDA DE LOS PARAMETROS QUE INDICAN LA UNIDAD DE LECTURA Y
0020   C     ESCRITURA.NUMERO DE CASO A RESOLVER
0021   C
0022         LECT=IPAR(1)
0023         INPR=IPAR(2)
0024         LS=IPAR(3)
0025         LK=IPAR(4)
0026   C
0027   C     COMPRUEBA SI QUEDA ALGUN CASO POR RESOLVER
0028   C
0029     100 IF(LK.EQ.LS)STOP
0030         LK=LK-1
0031   C
0032   C     LECTURA DE LOS DATOS
0033   C
0034         CALL LEERD(IMOD,IOP,INP,ZZ)
0035   C
0036   C     REALIZACION DE LOS CICLOS DE OPTIMIZACION
0037   C
0038         DO 5 KAN=1,NCVE
0039         Z(KAN)=ZZ(KAN)
0040       5 CONTINUE
0041         IF(IOP(1).EQ.0)GOTO 3
0042         DO 1 IKX=1,NVE
0043         IF(INP(IKX).EQ.0)GOTO 3
0044         NP=NTP/INP(IKX)
0045         CALL OVLAY(IP1)
0046       1 CONTINUE
0047       3 IF(IMOD.EQ.1)GOTO 4
0048         DO 6 KAN=1,NCVE
0049         Z(KAN)=ZZ(KAN)
0050       6 CONTINUE
0051       4 IF(IOP(2).EQ.0)GOTO 7
0052         DO 10 IKX=1 NVE
0053         IF(INP(IKX).EQ.0)GOTO 7
0054         NP=NTP/INP(IKX)
0055         CALL OVLAY(IP2)
```

```
PAGE 0002  CICPT  4:46 AM  SAT., 16  JAN., 1982


0056    10    CONTINUE
0057    7     GOTO 100
0058          END


 FTN4 COMPILER) HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00312       COMMON = 00001
```

```
PAGE 0003  FTN.   4:46 AM  SAT., 16  JAN., 1982


0058          SUBROUTINE LEERD(IMOD,IOP,INP,ZZ)
0060    C
0061    C       LECTURA DE LOS DATOS DE EJECUCION Y RESOLUCION
0062    C
0063          COMMON/COMUN/LECT,IMPR,TITU(30),ICOMN,MCVE,NVE,NREST,NP,NTP
0064          COMMON/OPNVP/IFUAC,NOP1,MOP1,FTR,VEVE,VXES,VXEM,CAXB,CDFI,ERRA,
0065         *ERRFO,VMDM,ERRG,NMITE,IMPRE,INDER,IER,IST
0066          COMMON/OPROC/KOUNT,IINPR,ITPRO,NMTER,ICPE,ICTPA,YLED,EPS(50)
0067          COMMON/VECES/Z(50)
0068          COMMON/DATOS/DATOP(3,361),DATON(360),IB,IC,ID,IEL,ISL(80),ZP(80),
0069         *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IPEN
0070          COMMON//IW(1)
0071          DIMENSION IOP(3),INP(16),ZZ(50),IHORA(15)
0072          READ(LECT,1)TITU
0073        1 FORMAT(30A2)
0074          READ(LECT,*)MCVE,NVE,NREST,NTP
0075          READ(LECT,*)IMOD,(IOP(NS),NS=1,3)
0076          READ(LECT,*)(INP(NS),NS=1,NVE)
0077          DO 2 NST=1,MCVE
0078          READ(LECT,*)ZZ(NST)
0079        2 CONTINUE
0080          READ(LECT,*)NOP1,MOP1
0081          READ(LECT,*)VEVE,VXES,VXEM,CAXB
0082          READ(LECT,*)ERRA,ERRFO,VMDM,ERRG
0083          READ(LECT,*)NMITE,IMPRE,INDER
0084          READ(LECT,*)IINPR,ITPRO,NMTER,ICPE,ICTPA,YLED
0085          DO 3 NST=1,MCVE
0086          READ(LECT,*)EPS(NST)
0087        3 CONTINUE
0088          CALL DATS(NTP,LECT)
0089          CALL FTIME(IHORA)
0090    C
0091    C       ESCRITURA DEL ENCABEZADO Y DATOS DE EJECUCION
0092    C
0093          WRITE(IMPR,1000)TITU,IHORA,MCVE,NVE,NREST,NTP,IMOD,
0094         *(IOP(NX),NX=1,3),(INP(NX),NX=1,NVE)
0095          WRITE(IMPR,1001)(NX,ZZ(NX),NX=1,MCVE)
0096     1000 FORMAT(1H1,//,1X,70(1H*),/,1X,1H*," OPTIMIZACION CON RESTRICCION
0097         *S. SEGUNDA SOLUCION ",19X,1H*,/,1X,1H*,5X,30A2,3X,1H*,/,1X,1H*,19X
0098         *1SA2,19X,1H*,/,1X,70(1H*),
0099         *,2/,5X,60(1H*),/,5X,"DIMENSION DEL V.E. =",I5,/,5X,"CICLOS DE ENS
0100         *AYO =",I5,/,5X,"N.DE RESTRICCIONES = ",I5,/,5X"NUMERO TOTAL DE PU
0101         *TOS =",I5,/,5X,"MODO DE OPERACION = ",I3,/,5X,"SEGMENTOS QUE USA
0102         * ",3I5,/,5X,"NUMERO DE PUNTOS POR CICLO : ",/,(34X,5I5))
0103     1001 FORMAT(10X,"VECTOR INICIAL",/,(5X,"X(",I2,")=",1PE15.7))
0104          WRITE(IMPR,1002)
0105     1002 FORMAT(5X,60(1H*),5X)
0106          RETURN
0107          END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)
```

```
PAGE 0001  FTN.   5:58 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM OPFI1(5),OPTIMIZACION CON RESTRICCIONES
0003   C
0004   C     PRIMER SEGMENTO DEL PROGRAMA CIOPT.METRICA VARIABLE CON RES-
0005   C     TRICCIONES
0006   C
0007         COMMON/COMUN/LECT,NG,TITU(30),ICOMM
0008         COMMON/OPMVF/IFUAC,NOP1,NOP1,FTR,VEVE,VXES,VXEM,B,R,ERRA,
0009        *ERRFO,VMDM,ERRG,IT,IMPRE,INDER,IDEER,IET
0010         COMMON/VECES/X(50)
0011         COMMON//IW(1)
0012   C
0013   C     DETERMINACION DE LOS INDICES DE LA MEMORIA DINAMICA
0014   C
0015         DO 10 I=1,ICOMM
0016    10   IW(I)=0
0017         N3=1+2*N
0018         N4=N3+2*M
0019         N5=N4+2*N*M
0020         N6=N5+2*(N*(N+1)/2)
0021         N7=N6+2*M
0022         N8=N7+2*N
0023         N9=N8+2*N*M
0024         N10=N9+2*N*M
0025         N11=N10+4*N
0026         N12=N11+2*N*M
0027         N13=N12+2*N
0028         N14=N13+2*M
0029         N15=N14+2*N
0030         N16=N15+2*N
0031         N17=N16+2*M
0032         N18=N17+2*M
0033         N19=N18+2*M
0034         N20=N19+2*N
0035         N21=N20+2*N
0036         N22=N21+2*M
0037         N23=N22+2*N
0038         N24=N23+4*N
0039         N25=N24+4*N
0040         N26=N25+2*M
0041         IF(N26.GT.ICOMM)GO TO 100
0042         CALL %CODI(X,N)
0043         IFUNC=0
0044   C
0045   C     ESCALADO
0046   C
0047         CALL ESCAL(X,IW(N6),IW(N9),IW(N12),IW(N13),IW(N15),
0048        1IW(N25),IW(N16),IW(N17),IW(N19),IW(N20),IW(N21),IW(N14),
0049        1IW(N18))
0050   C
0051   C     ESCRIBE EL ENCABEZADO DE ESTE SEGMENTO
0052   C
0053         WRITE(NO,20)TITU
0054      20 FORMAT(1H1,10X,30A2,//,4X,"RESULTADOS DEL CALCULO "//4X,
0055        1"VALORES INICIALES")
```

```
0056          IFUNC=1
0057    C
0058    C     APLICA EL ESCALADO
0059    C
0060          CALL %FEFG(1,X,F,IW(1),IW(N3),IW(N4),IW(N12),IW(N13),
0061         1IW(N14),IW(N15),IW(N10))
0062    C
0063    C     REALIZA LA OPTIMIZACION
0064    C
0065          IFUNC=0
0066          CALL MINRE(X,F,IW(1),IW(N3),IW(N4),IW(N5),IW(N6),IW(N7),
0067         1IW(N8),IW(N9),IW(N10),IW(N11),IW(N12),IW(N13),IW(N14),IW(N15),
0068         1IW(N16),IW(N22),IW(N23),IW(N24))
0069          WRITE (NO,30)IET,IDEER
0070       30 FORMAT (///4X,"VALORES FINALES PARA ",I5,"ITERACIONES",
0071         15X,"IDEER= ",I3)
0072          IFUNC=1
0073    C
0074    C     VUELVE A LOS VERDADEROS VALORES
0075    C
0076          CALL %FEFG(1,X,F,IW(1),IW(N3),IW(N4),IW(N12),IW(N13),
0077         1IW(N14),IW(N15),IW(N10))
0078          CALL ACONT(X,IW(N14))
0079          GOTO 101
0080    C
0081    C     ESCRIBE UN MENSAJE SI SE SUPERA LA CAPACIDAD ACTUAL
0082    C
0083      100 WRITE(NO,31)N26
0084       31 FORMAT(3/,4X,"SE SUPERA LA CAPACIDAD ACTUAL,N26= ",3X,I5)
0085    C
0086    C     DEVUELVE EL CONTROL AL PROGRAMA PRINCIPAL CIOPT
0087    C
0088      101 CALL OVRTN
0089          END
```

```
FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00545      COMMON = 00001
```

```
0090          SUBROUTINE ESCAL(X,GRADG1,GRADG2,XSC,PDI,TRINCR,G,GP,GM,GRADF1,
0091         1GRADF2,A2GD,XERT,GD),ESCALADO
0092          COMMON/COMUN/LECT,NG,TITU(30)
0093          COMMON/OPMVP/IFUAC,NOP1,NOP1,FTR,VEVE,VXES,VXEM,B,R,ERRA,
0094         *ERRFO,VNDM,ERRG,IT,IMPRE,INDER,IDEER,IE
0095          COMMON//IW(1)
0096          DIMENSION X(1),GRADG1(1),GRADG2(1),XSC(1),PDI(1),TRINCR(1),G(1),
0097         1GP(1),GM(1),GRADF1(1),GRADF2(1),A2GD(1),XERT(1),GD(1)
0098    C
0099    C     CALCULO DE LA PRIMERA Y SEGUNDA DERIV<0A
0100    C
0101          DEL1= SQRT(DEL)
0102          DEL2= DEL**(1./3.)
0103          FSC=1.0
0104          IF (1.GT.N)GO TO 5
0105          DO 10 I=1,N
0106       10 XSC(I)=1.0
0107        5 CONTINUE
0108          IF(N)40,40,20
0109       20 IF (1.GT.N)GO TO 15
0110          DO 30 J=1,N
0111       30 PDI(J)=1.0
0112       15 CONTINUE
0113       40 CALL AFEFG(1,X,F,GRADF1,G,GRADG1,XSC,PDI,XERT,TRINCR,GD)
0114          IF (1.GT.N)GO TO 25
0115          DO 70 I=1,N
0116          X(I)=X(I)+DEL2
0117          CALL AFEFG(1,X,FP,GRADF1,GP,GRADG1,XSC,PDI,XERT,TRINCR,GD)
0118          X(I)=X(I)-2.*DEL2
0119          CALL AFEFG(1,X,FM,GRADF1,GM,GRADG1,XSC,PDI,XERT,TRINCR,GD)
0120          X(I)=X(I)+DEL2
0121          GRADF1(I)=(FP-FM)/2./DEL2
0122          GRADF2(I)=(FP+FM-2.*F)/DEL2/DEL2
0123          IF(N)70,70,50
0124       50 K=1
0125          IF (1.GT.N)GO TO 35
0126          DO 60 J=1,N
0127          GRADG1(K)=(GP(J)-GM(J))/2./DEL2
0128          GRADG2(K)=(GP(J)+GM(J)-2.*G(J))/DEL2/DEL2
0129       60 K=K+N
0130       35 CONTINUE
0131       70 CONTINUE
0132       25 CONTINUE
0133    C
0134    C   CALCULO DE FSC Y PRIMERA APROXIMACION DE XSC
0135    C
0136          FSC=ABS(F)
0137          IF(N)160,160,80
0138       80 K=1
0139          IF (1.GT.N)GO TO 45
0140          DO 100 J=1,N
0141          S=0.0
0142          IF (1.GT.N)GO TO 55
0143          DO 90 I=1,N
0144          S=S+GRADG1(K)**2
```

```
0145          90 K=K+1
0146          55 CONTINUE
0147         100 ABGD(J)=SQRT(S)
0148          45 CONTINUE
0149             RN=10.*SQRT(FLOAT(N))
0150             IF(1.GT.N)GO TO 65
0151             DO 130 I=1,N
0152             FS1=ABS(GRADF1(I)/F)
0153             FS2=SQRT(ABS(GRADF2(I)/F))
0154             S=0.0
0155             K=I
0156             IF(1.GT.N)GO TO 75
0157             DO 120 J=1,N
0158             GRAT=ABS(GRADG1(K))/ABGD(J)*VXES*RN/VXEM
0159             IF(GRAT-1.)120,120,110
0160         110 T=ABS(GRADG2(K)/GRADG1(K))
0161             S=AMAX1(S,T)
0162         120 K=K+N
0163          75 CONTINUE
0164             S=AMAX1(FS1,FS2,S,1./VXES)
0165             S=AMIN1(S,1./VXEM)
0166         130 XSC(I)=1./S
0167          65 CONTINUE
0168   C
0169   C  CALCULO DE PGI
0170   C
0171             K=1
0172             IF(1.GT.N)GO TO 85
0173             DO 150 J=1,N
0174             S=0.0
0175             IF(1.GT.N)GO TO 95
0176             DO 140 I=1,N
0177             T=GRADG1(K)*XSC(I)
0178             S=S+T*T
0179         140 K=K+1
0180          95 CONTINUE
0181         150 PGI(J)=SQRT(S)
0182          85 CONTINUE
0183         160 IF(1.GT.N)GO TO 105
0184   C
0185   C  CALCULO DE LOS VALORES FINALES DE XSCY TRINCR
0186   C
0187             DO 200 I=1,N
0188             S=0.0
0189             IF(N)190,190,170
0190         170 K=I
0191             IF (1.GT.N)GO TO 115
0192             DO 180 J=1,N
0193             T=GRADG1(K)/PGI(J)
0194             S=S+T*T
0195         180 K=K+N
0196         115 CONTINUE
0197             S=2.*S*S
0198         190 S=S+GRADF2(I)/FSC
0199             S=AMAX1(S,1./VXES/VXES)
```

```
 PAGE 0003  ESCAL  5:59 AM  SAT., 16  JAN., 1982


02000          S=AMAX1(SQRT(1./S),XMEM)
02001          XSC(I)=S
02002          TRINCR(I)=XSC(I)*DEL1
02003      200 X(I)=X(I)/XSC(I)
02004      105 CONTINUE
02005          RETURN
02006          END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00253      COMMON = 00001
```

```
PAGE 0006  FTN.   5:58 AM  SAT., 16  JAN., 1982


0207          SUBROUTINE AFEFG(IF,X,F,GRADF,G,GRADG,XSC,POI,XERT,TRINCR,CD),APL
0208         1CACION DEL ESCALADO
0209          COMMON/COMUN/LECT,NO,TITU(30)
0210          COMMON/OPMVF/IFUAC,NOF1,NOF1,FTR,VEVE,VXES,VXEM,B,R,ERRA,
0211         *ERRFO,VMDM,ERRC,IT,IMPRE,INDER,IDEER,IET
0212          COMMON//IW(1)
0213          DIMENSION X(1),GRADF(1),G(1),GRADG(1),XSC(1),POI(1),XERT(1),
0214         1TRINCR(1),CD(1)
0215          IF(1.GT.N)GO TO 5
0216          DO 10 I=1,N
0217       10 XERT(I)=X(I)*XSC(I)
0218        5 CONTINUE
0219          IF(N)40,40,20
0220       20 CALL GRADI(IF,XERT,F,GRADF,G,GRADG,TRINCR,CD)
0221          IF(1.GT.N)GO TO 15
0222          DO 30 J=1,N
0223       30 G(J)=G(J)/POI(J)
0224       15 CONTINUE
0225          GO TO 50
0226       40 CALL GRADI(IF,XERT,F,GRADF,G,GRADG,TRINCR,CD)
0227       50 F=F/FSC
0228          IF(IF-1)100,100,60
0229       60 IF(1.GT.N)GO TO 25
0230          DO 90 I=1,N
0231          GRADF(I)=GRADF(I)/FSC*XSC(I)
0232          IF(N)90,90,70
0233       70 K=I
0234          IF(1.GT.N)GO TO 35
0235          DO 80 J=1,N
0236          GRADG(K)=GRADG(K)/POI(J)*XSC(I)
0237       80 K=K+N
0238       35 CONTINUE
0239       90 CONTINUE
0240       25 CONTINUE
0241      100 RETURN
0242          END


 FIN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00232      COMMON = 00001
```

```
PAGE 0007  FTN.   5:58 AM  SAT., 16  JAN., 1982


0243          SUBROUTINE GRADI(IF,X,F,GRADF,G,GRADG,TAINCR,GD),GRADIENTE
0244          COMMON/COMUN/LECT,NO,TITU(30)
0245          COMMON/OPMVP/IFUNC,NOP1,NOP1,FTR,VEVE,VXES,VXEN,B,R,ERRA,
0246         *ERRFD,VNDM,ERRG,IT,IMPRE,INDER,IDEER,IET
0247          COMMON//IW(1)
0248    C
0249          DIMENSION X(1),GRADF(1),G(1),GRADG(1),TAINCR(1),GD(1)
0250    C
0251    C   CALCULO DE LOS VALORES
0252    C
0253          IF (M)20,20,10
0254       10 CALL FUOBJ(X,F,G)
0255          GO TO 30
0256       20 CALL FUOBJ(X,F,G)
0257       30 IF (IF-1)40,40,50
0258       40 RETURN
0259    C
0260    C   CALCULO DEL GRADIENTE
0261    C
0262       50 IF(1.GT.M)GO TO 5
0263          DO 100 I=1,M
0264          DT=TAINCR(I)
0265          X(I)=X(I)+DT
0266          IF(M)90,80,60
0267       60 CALL FUOBJ(X,FD,GD)
0268          K=I
0269          IF(1.GT.M)GO TO 15
0270          DO 70 J=1,M
0271          GRADG(K)=(GD(J)-G(J))/DT
0272       70 K=K+M
0273       15 CONTINUE
0274          GO TO 90
0275       80 CALL FUOBJ(X,FD,G)
0276       90 X(I)=X(I)-DT
0277          GRADF(I)=(FD-F)/DT
0278      100 CONTINUE
0279        5 CONTINUE
0280          RETURN
0281          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **   NO WARNINGS **   NO ERRORS **    PROGRAM = 00171        COMMON = 00001
```

```
0232          SUBROUTINE MINPE(X,F,GRADF,G,GRADG,H,FLAM,S,GO,GN,W,A,XSC,PDI,
0233         1XERT,TRINCR,GD,HIS,F10,T10),MINIMIZACION
0234          COMMON/COMUN/LECT,NO,TITU(30)
0235          COMMON/OPMVF/IFUAC,NOP1,NOP1,FTR,VEVE,VXES,VXEM,B,R,ERRA,
0236         *ERRFO,VMDM,ERRG,IT,IMPRE,INDER,IDEBR,IE1
0237          COMMON/VIU(1)
0238    C
0239    C       INICIO DE LA MATRIZ H
0240    C
0241          DIMENSION X(1),GRADF(1),G(1),GRADG(1),H(1),FLAM(1),S(1),GO(1),
0242         1GN(1),W(1),A(1),XSC(1),PDI(1),XERT(1),TRINCR(1),GD(1),HIS(1),
0243         1F10(1),T10(1)
0244          IF (INDER-1)10,20,20
0245       10 CALL CADBU(H,1,N,S,GO,GN,ALFA,ERRA,ERRG,W)
0246    C
0247    C       ESCRITURA DE LAS CONDICIONES INICIALES
0248    C
0249       20 IP=IMPRE/10
0250          IR=IMPRE-10*IP
0251          IF (IP)40,40,30
0252       30 WRITE (NO,420)N,M,IT,IMPRE,INDER,B,R,ERRA,ERRFO,VMDM,ERRG,DEL
0253          WRITE (NO,430)(I,X(I),I=1,N)
0254    C
0255    C       FUNCIONES Y SUS GRADIENTES EN EL PUNTO INICIAL
0256    C
0257       40 CALL AFEFG(2,X,F,GRADF,G,GRADG,XSC,PDI,XERT,TRINCR,GD)
0258          IEI=0
0259          ICV=-1
0260          IEL=0
0261          INL=1
0262    C
0263    C       CALCULO DE LOS VALORES DE LANDA
0264    C
0265          IF (1.GT.M)GO TO 5
0266          DO 50 I=1,M
0267       50 FLAM(I)=0.0
0268        5 CONTINUE
0269       60 CALL LANDA(NO,N,M,GRADF,GRADG,G,S,FLAM,R,ERRG,W,A,ICV,HIS)
0270          IEL=IEL+1
0271    C
0272    C       DETERMINACION DE LA FUNCION AUXILIAR(PHI),SU GRADIENTE(GN),
0273    C       EL VALOR(GNA) DE GN Y LA SUMA(GS) DE S*(I)**2 DESDE I=1 A M
0274    C
0275          L=1
0276       70 PHI=F
0277          IF (1.GT.M)GO TO 15
0278          DO 80 I=1,M
0279       80 PHI=PHI+G(I)*(B*G(I)-FLAM(I))
0280       15 CONTINUE
0281          ICKCGT=L
0282          IF (ICKCGT.LE.1)GO TO 90
0283          IF (ICKCGT.EQ.2)GO TO 240
0284          GO TO (90,240),ICKCGT
0285       90 GNA = 0.
0286          IF (1.GT.N)GO TO 25
```

```
0337          DO 110 I=1,M
0338          T=GRADF(I)
0339          K=I
0340          IF(1.GT.N)GO TO 35
0341          DO 100 J=1,N
0342          T=T+GRADG(K)*(2.*B*G(J)-FLAM(J))
0343      100 K=K+M
0344       35 CONTINUE
0345          GN(I)=T
0346      110 GNA=GNA+T*T
0347       25 CONTINUE
0348          GNA=SQRT(GNA)
0349          GS=0.
0350          IF(1.GT.M)GO TO 45
0351          DO 120 I=1,M
0352      120 GS=GS+B*G(I)*G(I)
0353       45 CONTINUE
0354          ICKCGT=L
0355          IF(ICKCGT.LE.1)GO TO 130
0356          IF(ICKCGT.GE.2)GO TO 280
0357          GO TO (130,280),ICKCGT
0358    C
0359    C     ESCRITURA DE LOS LANDA DETERMINADOS
0360    C
0361      130 IF(IP)150,150,140
0362      140 WRITE (ND,450)IEL,IET,ICV
0363          WRITE (ND,430)(I,X(I),I=1,M)
0364          WRITE (ND,440)(FLAM(I),I=1,M)
0365          WRITE (ND,460)(G(I),I=1,M)
0366          WRITE (ND,470)PHI,F,GS,GNA
0367    C
0368    C     COMPROBAR SI SE OBTIENE LA CONVERGENCIA
0369    C
0370      150 IF(ICV-1)160,170,180
0371      160 IF(INL)180,170,180
0372      170 IDEER=0
0373          GO TO 380
0374    C
0375    C     ITERACION LINEAL
0376      180 L=2
0377          INL=1
0378          ITN=0
0379          IF (1.GT.N)GO TO 55
0380          DO 190 I=1,M
0381      190 GO(I)=GN(I)
0382       55 CONTINUE
0383      200 IF(IET-IT)220,210,210
0384      210 IDEER=1
0385          GO TO 390
0386      220 IF (1.GT.N)GO TO 65
0387          DO 230 I=1,M
0388      230 W(I)=X(I)
0389       65 CONTINUE
0390          IET=IET+1
0391          ITN=ITN+1
```

```
0392          INLP=INL
0393          INL=0
0394          ALFA=1.0
0395      240 CALL SUDIN(INL,N,NF,W,S,ALFA,X,PHI,ERRA,VMDW,ERRFG,F10,T10)
0396          IF(INL-1)270,260,250
0397      250 IDEER=2
0398          GO TO 380
0399      260 CALL AFEFG(1,X,F,GRADF,G,GRADG,XSC,POI,KERT,TRINCR,GD)
0400          GO TO 70
0401      270 CALL AFEFG(2,X,F,GRADF,G,GRADG,XSC,POI,KERT,TRINCR,GD)
0402          GO TO 90
0403    C
0404    C     CALCULO DEL INDICE INL PARA LA ITERACION LINEAL
0405    C
0406      280 INL=INL+2
0407          IF(ALFA)300,300,290
0408      290 IF(GNA-ERRG)300,300,310
0409      300 INL=0
0410    C
0411    C     ESCITURA DE LOS DATOS PARA LA ITERACION LINEAL
0412    C
0413      310 IF (IP+IR)340,340,320
0414      320 IF(NOD(IET,IR))340,330,340
0415      330 WRITE (NO,430)IET,INL,ALFA,NF
0416          WRITE (NO,420)(I,X(I),I=1,N)
0417          WRITE (NO,470)PHI,F,GS,GNA
0418    C
0419    C     CORRECION DE H Y CALCULO DE NUEVA S
0420    C
0421      340 CALL CADBU(H,3,N,S,GO,GH,ALFA,ERRA,ERRG,W)
0422    C
0423    C     LA CONVERGENCIA SE CUMPLE SI IMPL=0 Y INL=0 (ITN=2)
0424    C
0425          IF (INLP+INL)350,350,360
0426      350 IDEER=0
0427          GO TO 390
0428    C
0429    C     CALCULO DE NUEVOS VALORES DE LANDA PARA ITERACIONES POSTERIORES
0430    C
0431      360 IF (ITN-1)200,200,370
0432      370 IF (INL*(N-N+1-ITN))60,60,200
0433    C
0434    C     FIN DE LA MINIMIZACION
0435    C
0436      380 IF(IP)410,410,390
0437      390 WRITE(NO,490)IDEER
0438          K=1
0439          IF(1.GT.N) GO TO 75
0440          DO 400 I=1,N
0441          W(I)=H(K)
0442      400 K=K+N-1-I
0443       75 CONTINUE
0444          WRITE(NO,500)(W(I),I=1,N)
0445      410 RETURN
0446      420 FORMAT (1H1,'DATOS INICIALES PARA LA MINIMIZACION',//' N=',I3,3
```

```
PAGE 0011  MIERE  5:58 AM  SAT., 16  JAN., 1982


0447          1X,"N=",I2,3X,"IT=",I4,3X,"IMPRE=",I2,3X,"INDER=",I1/" B=",E10.4,
0448          15X,"R=",E10.3,5X,"ERRA=",E10.3,5X,"ERRFO=",E10.3/1X,"VRDR=",F10.3
0449          1,5X,"ERRG=",E10.3,5X," DEL = ",E10.3)
0450      430 FORMAT(/,3(2X,"X(",I2,")=",E14.7,2X))
0451      440 FORMAT(/,19X,"LANDA =",(2X,E14.7))
0452      450 FORMAT(3/" DETERMINACION DE LANDA",I4.2X,"DESPUES DE LA ITERACION
0453          1,I5,7X,"ICV=",I1)
0454      460 FORMAT(22X,"G =",(2X,E14.7))
0455      470 FORMAT(20X," PHI =",2X,E14.7,/,23X,"F =",2X,E14.7/19X,"SUMA DE B
0456          1**2 =",2X,E14.7,/," VALOR DEL GRADIENTE(PHI)=",2X,E14.7)
0457      480 FORMAT (/" ITERACION ",I5,7X,"INL=",I1,5X,"ALFR=",E10.4,6X,"N. D
0458          1,PUNTOS=",I3)
0459      490 FORMAT (3/" MINIMIZACION CON RECTRICCIONES TERMINADA",3X,"IDERR="
0460          1,I1)
0461      500 FORMAT (" LOS ELEMENTOS DE LA DIAGONAL DE H ",/,5(3X,E14.7))
0462          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **    PROGRAM = 01263       COMMON = 00001
```

```
0463          SUBROUTINE CADSUC(H,INDER,N,S,GO,GN,ALFA,ERRA,ERRG,V),CALCULO DE L
0464         1DIRECCIONES DE BUSQUEDA
0465          DIMENSION H(1),GO(1),GN(1),S(1),W(1),Y(1),H(1)
0466    C
0467    C     GENERACION DE LA MATRIZ IDENTIDAD
0468    C
0469          IF (INDER-2)10,60,90
0470       10 K=1
0471          IF (1.GT.N)GO TO 5
0472          DO 40 I=1,N
0473          H(K)=1.0
0474          NJ=N-I
0475          IF (NJ)50,50,20
0476       20 IF (1.GT.NJ)GO TO 15
0477          DO 30 J=1,NJ
0478          KJ=K+J
0479       30 H(KJ) =0.0
0480       15 CONTINUE
0481       40 K=KJ+1
0482        5 CONTINUE
0483       50 IF (INDER)60,60,80
0484    C
0485    C     DIRECION DE DESCENSO
0486    C
0487       60 IF (1.GT.N)GO TO 25
0488          DO 70 I=1,N
0489       70 S(I)=-GO(I)
0490       25 CONTINUE
0491       80 RETURN
0492    C
0493    C     CORRECCION DE LA MATRIZ H
0494    C
0495    C     CALCULO DEL CAMBIO DE GRADIENTE DE Y(CK)
0496    C
0497       90 IF (INDER-4)100,100,210
0498      100 IF (1.GT.N)GO TO 35
0499          DO 110 I=1,N
0500          T=GN(I)
0501          GN(I)=T-GO(I)
0502      110 GO(I)=T
0503    C
0504    C     CALCULO DE H,Y.(G(1) A G(N))1)
0505    C
0506       35 CONTINUE
0507          IF(1.GT.N)GO TO 45
0508          DO 150 I=1,N
0509          T=0.0
0510          K=I
0511          IF (1.GT.N) GO TO 55
0512          DO 140 J=1,N
0513          T=T+GN(J)*H(K)
0514          IF (J-I)120,130,130
0515      120 K=K+N-J
0516          GO TO 140
0517      130 K=K+1
```

```
0518      140 CONTINUE
0519       55 CONTINUE
0520      150 U(I)=I
0521       45 CONTINUE
0522   C
0523   C      CALCULO DEL CAMBIO DE POSICON DE Z(G(N+1) A G(2*N))(2*N)
0524   C
0525          K=N
0526          IF(1.GT.N)GO TO 65
0527          DO 160 I=1,N
0528          K=K+1
0529      160 U(I)=ALFA*S(I)
0530       65 CONTINUE
0531   C
0532   C      CALCULO DE Q=Y.H.Y, P=Y.Z  Y  R=SQRT(Z*Z)
0533   C
0534          R=0.0
0535          Q=0.0
0536          P=0.0
0537          K=N
0538          IF (1.GT.N)GO TO 75
0539          DO 170 I=1,N
0540          K=K+1
0541          R=R+U(K)*U(K)
0542          Q=Q+G(I)*U(I)
0543      170 P=P+G(I)*U(K)
0544       75 CONTINUE
0545          R=SQRT(R)
0546   C
0547   C      NO CAMBIA H SI P O R SON MUY PEQUENAS
0548   C
0549          IF (R-ERRA)210,210,180
0550      180 IF (P-R*ERRO)210,210,190
0551   C
0552   C      CALCULO DE UNA NUEVA H
0553   C
0554      190 K=1
0555          IF(1.GT.N)GO TO 85
0556          DO 200 I=1,N
0557          LI=N+I
0558          DO 200 J=1,N
0559          LJ=N+J
0560          H(K)=H(K)+U(LI)*U(LJ)/P-U(I)*U(J)/Q
0561      200 K=K+1
0562       85 CONTINUE
0563   C
0564   C      CALCULO DE NUEVA DIRECION
0565   C
0566      210 IF (INDER-4)220,270,220
0567      220 IF (1.GT.N)GOTO 85
0568          DO 260 I=1,N
0569          T=0.0
0570          K=I
0571          IF (1.GT.N)GOTO 105
0572          DO 250 J=1,N
```

```
PAGE 0014  CADBU  5:58 AM  SAT., 16  JAN., 1982


0573          T=T+GO(J)*H(K)
0574          IF (J-1)230,240,240
0575      230 K=K+H-J
0576          GO TO 250
0577      240 K=K+1
0578      250 CONTINUE
0579      105 CONTINUE
0580      260 S(I)=-T
0581       95 CONTINUE
0582      270 RETURN
0583          END


 FTN4, COMPILER) HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00613       COMMON = 00000
```

```
0594          SUBROUTINE BUDIN (INDER,ND,NPT,BEGIN,S,ALFA,END,FN,ERRA,VMDM,
0595         1ERRFO,F,T),BUSQUEDA DIMENSIONAL
0596          DIMENSION S(1),BEGIN(1),END(1),F(1),T(1)
0597          IF (INDER)40,40,10
0598       10 F(KK)=FN
0599          NPT=NPT+1
0600          ICKCGT=NGOTO
0601          IF (ICKCGT.LE.1)GO TO 60
0602          IF (ICKCGT.GE.6) GO TO 360
0603          GO TO (60,70,110,150,290,360),ICKCGT
0604       20 IF (1.GT.ND)GO TO 5
0605    C
0606    C        OBTENCION DE LOS VALORES DE LA FUNCION
0607    C
0608          DO 30 I=1,ND
0609       30 END(I)=BEGIN(I)+T(KK)*S(I)
0610        5 CONTINUE
0611          RETURN
0612    C
0613    C        CAMBIO EN ALFA CORRESPONDIENTE A ERRA Y VMDM
0614    C
0615       40 INDER=1
0616          NPT=0
0617          Z=0.0
0618          IF (1.GT.ND)GO TO 15
0619          DO 50 I=1,ND
0620       50 Z=Z+S(I)*S(I)
0621       15 CONTINUE
0622          Z=SQRT(Z)
0623          Z=AMAX1(Z,1.E-20)
0624          DEL=ERRA/Z
0625          DEF=VMDM/Z
0626    C
0627    C        OBTENCION DE TRES PUNTOS,L,M,N,CON L Y N EN LADOS OPUESTOS DE
0628    C        M, F(L) Y=F(N) NO MENORES QUE F(M),Y LAS DISTANCIAS(L A N)
0629    C        Y (M A N) EL MINIMO DE VMDM
0630    C        PUNTOS INICIALES Y RECOMENDADOS
0631    C
0632          T(1)=0.0
0633          KK=1
0634          NGOTO=1
0635          GO TO 20
0636       60 FKEEP=F(1)
0637          T(2)=ALFA
0638          KK=2
0639          NGOTO=2
0640          GO TO 20
0641    C
0642    C        PUNTOS M Y L
0643    C
0644       70 L=1
0645          M=2
0646          IF (F(1)-F(2))80,90,90
0647       80 M=1
0648          L=2
```

```
0638      90 IF (ABS(T(M)-T(L))-DEF)100,130,130
0640     100 T(L)=T(M)+SIGN(DEF,T(L)-T(M))
0641         KK=L
0642         NGOTO=3
0643         GO TO 20
0644     110 IF (F(L)-F(M))120,130,130
0645     120 I=L
0646         L=M
0647         M=I
0648   C
0649   C     PUNTO N(INTERVALO DE LONGITUD DOBLE CADA VEZ)
0650   C
0651     130 Z=1.0
0652         N=3
0653     140 T(N)=T(M)+Z*(T(M)-T(L))
0654         KK=N
0655         NGOTO=4
0656         GO TO 20
0657     150 IF (F(N)-F(M))160,180,180
0658     160 I=L
0659         L=M
0660         M=N
0661         N=I
0662         Z=2.0
0663         IF (ABS(T(N)-T(L))-1.E20)140,170,170
0664     170 INDEP=2
0665         RETURN
0666   C
0667   C     DISMINUCION DE LA DISTANCIA(L A N) A  MENOR QUE 4*VNDN,MANTENIENDO
0668   C     LA DISTANCIA (L A M) Y (M A N) EN EL MINIMO VNDN
0669   C
0670     180 NEW=4
0671         NBAD=0
0672     190 IF (ABS(T(N)-T(L))-ABS(T(M)-T(N)))210, 210,200
0673     200 I=L
0674         L=N
0675         N=I
0676   C
0677   C     ESTIMACION DE LA POSICION DEL PUNTO MINIMO PARA UN AJUSTE PARABO-
0678   C     LICO F=A+B*T+C*T**2
0679   C
0680     210 T1=T(L)-T(M)
0681         T2=T(N)-T(M)
0682         H1=ABS(F(L)-F(M))/T1
0683         H2=ABS(F(N)-F(M))/T2
0684         C=(H2-H1)/(T2-T1)
0685         B=(H1*T2-H2*T1)/(T2-T1)
0686         T(NEW)=T(M)-B/2./(C+SIGN(1.E-37,C))
0687   C
0688   C     FINALIZA EL CICLO CUANDO LA DISTANCIA (L A N) ES MENOR QUE 4*VND
0689   C
0690         IF (ABS(T(N)-T(L))-4.*DEF)350,350,220
0691     220 IF (NBAD-2)240,230,230
0692     230 T(NEW)=SQRT(T1*(T1-T2))
0693         T(NEW)=T(L)+SIGN(T(NEW),T2)
```

```
0694          NBAD=0
0695       240 IF (ABS(T(NEW)-T(M))-DEF) 250,260,260
0696       250 T(NEW)=T(M)-SIGN(DEF,T2)
0697     C
0698     C      SE HACE QUE NEW ESTE ENTRE M Y N
0699     C
0700       260 IF ((T(NEW)-T(M))*T2) 270,290,280
0701       270 I=L
0702           L=N
0703           N=I
0704       280 KK=NEW
0705           NGOTO =5
0706           GO TO 20
0707       290 Z=ABS(T(N)-T(L))
0708           IF (F(NEW)-F(M))300,310,310
0709       300 I=L
0710           L=M
0711           M=NEW
0712           NEW=I
0713           GO TO 320
0714       310 I=N
0715           N=NEW
0716           NEW=I
0717     C
0718     C      PRUEBA DE QUE LA DISTANCIA (L A N)  DECRECE AL MENOS EL 10%
0719     C
0720       320 IF (ABS(T(N)-T(L))/Z-0.9) 330,340,340
0721       330 NBAD=0
0722           GO TO 190
0723       340 NBAD=NBAD+1
0724           GO TO 190
0725     C
0726     C      OBTENCION DEL VALOR DE LA FUNCION EN EL MINIMO PUNTO ESTIMADO
0727     C
0728       350 KK=NEW
0729           NGOTO=6
0730           GO TO 20
0731       360 IF (F(NEW)-F(M))380,380,370
0732       370 NEW=M
0733       380 ALFA=T(NEW)
0734           FM=F(NEW)
0735           IF (1.GT.ND)GO TO 25
0736           DO 390 I=1,ND
0737       390 END(I)=BEGIN(I)+ALFA*S(I)
0738        25 CONTINUE
0739     C
0740     C      COMPROBACION QUE LA APROXIMACION ES SIGNIFICATIVA
0741     C
0742           IF (FKEEP-FM-ERRFO)400,410,410
0743       400 INDER=-2
0744           RETURN
0745       410 IF (ABS(ALFA)-DEL)400,420,420
0746       420 INDER=-1
0747           RETURN
0748           END
```

```
0749          SUBROUTINE LANDA(NO,N,M,F,GM,GV,SL,FLAM,R,ERRG,W,A,ICV,HIS),C.LA>
0750         1A
0751          DIMENSION F(1),GM(1),HIS(1),GV(1),SL(1),FLAM(1),A(1),W(1)
0752    C
0753    C
0754    C       CALCULO DE U(I)=F,G(I)
0755    C
0756          AS=0.)
0757          B=0.0
0758          Z=0.0
0759       10 K=1
0760          IF (1.GT.M)GO TO 5
0761          DO 30 I=1,M
0762          T=0.0
0763          IF (1.GT.N) GO TO 15
0764          DO 20 J=1,N
0765          T=T+F(J)*GM(K)
0766       20 K=K+1
0767       15 CONTINUE
0768       30 U(I)=T
0769        5 CONTINUE
0770    C
0771    C       CALCULO DE A(M*(I-1)+J)=G(I),G(J)
0772    C
0773          K=1-M
0774          IF (1.GT.M)GO TO 25
0775          DO 50 I=1,M
0776          K=K+M
0777          L=K
0778          DO 50 J=I,M
0779          T=0.0
0780          DO 40 IN=1,N
0781          T=T+GM(K)*GM(L)
0782          L=L+1
0783       40 K=K+1
0784       35 CONTINUE
0785          K=K-M
0786          LL=M*(I-1)+J
0787          A(LL)=T
0788          LL=M*(J-1)+I
0789       50 A(LL)=T
0790       25 CONTINUE
0791    C
0792    C       CALCULO DE RD Y B
0793    C
0794          RD =0.0
0795          K=1
0796          L=1
0797          IF(1.GT.M)GO TO 45
0798          DO 70 I=1,M
0799          SGI=U(I)
0800          IF(1.GT.M)GO TO 55
0801          DO 60 J=1,M
0802          SGI=SGI-FLAM(J)*A(K)
0803       60 K=K+1
```

```
0804      55 CONTINUE
0805         SGI=ABS(SGI)
0806         AGI=A(L)**0.5
0807         B=B+A(L)
0808         L=L+M+1
0809         T=AGI+ERRG
0810         T=SGI/T/ERRG
0811      70 RD=AMAX1(RD,T)
0812      45 CONTINUE
0813   C
0814   C       CALCULO DE SL(I) Y NUEVOS VALORES PARA FLAM(I)
0815   C
0816         K=1
0817         Q=0*Z
0818         IF (1.GT.M)GO TO 65
0819         DO 90 I=1,M
0820         A(K)=A(K)+Q
0821         SL(I)=-GV(I)
0822      80 K=K+M+1
0823      85 CONTINUE
0824         CALL RSIEL(NO,A,W,SL,M,IDEER)
0825         IF (IDEER)100,100,90
0826      90 Z=10.*Z+1.E-12
0827         GO TO 10
0828     100 D=0.0
0829         IF (1.GT.N)GO TO 75
0830         DO 110 I=1,M
0831         Z=W(I)-FLAM(I)
0832         FLAM(I)=W(I)
0833         W(I)=SL(I)
0834     110 D=D+Z*Z
0835      75 CONTINUE
0836         D=SQRT(D)
0837         IF(1.GT.N)GO TO 85
0838         DO 120 I=1,M
0839         K=I
0840         SL(I)=0.0
0841         DO 120 J=1,M
0842         SL(I)=SL(I)+W(J)*GM(K)
0843     120 K=K+M
0844      85 CONTINUE
0845   C
0846   C       MODIFICA SL(I) SI ES NECESARIO
0847   C
0848         T=0.0
0849         IF(1.GT.N)GO TO 95
0850         DO 140 I=1,M
0851         W(I)=0.0
0852         K=I
0853         IF(1.GT.M)GO TO 105
0854         DO 130 J=1,M
0855         W(I)=W(I)+GV(J)*GM(K)
0856     130 K=K+M
0857     105 CONTINUE
0858         Z=W(I)*SIGN(1.E-20,W(I))
```

```
0858          W(I)=Z
0860      140 T=T+Z*Z
0861       95 CONTINUE
0862          T=SQRT(T)
0863          SW=0.0
0864          IF(1.GT.N)GO TO 115
0865          DO 150 I=1,N
0866          W(I)=W(I)/T
0867      150 SW=SW+W(I)*SL(I)
0868      115 CONTINUE
0869          SW=SW+SIGN(1.E-20,SW)
0870          SP=0.0
0871          IF (1.GT.N)GO TO 125
0872          DO 160 I=1,N
0873          Z=SL(I)-SW*W(I)
0874          SL(I)=Z
0875      160 SP=SP+Z*Z
0876      125 CONTINUE
0877          SP=SQRT(SP)+1.E-20
0878          T=AMIN1(R*ABS(SW)/SP,1.)
0879          IF(1.GT.N)GO TO 135
0880          DO 170 I=1,N
0881      170 SL(I)=T*SL(I)+SW*W(I)
0882      135 CONTINUE
0883    C
0884    C       PARAMETROS DE CONVERGENCIA
0885    C
0886          IF (ICV)180,200,200
0887      180 I100=0
0888          DO 190 I=1,4
0889      190 HIS(I)=1.E20
0890      200 ICV=2
0891          T=HIS(1)
0892          DO 210 I=1,3
0893          T=T+HIS(I+1)
0894      210 HIS(I)=HIS(I+1)
0895          HIS(4)=0
0896          T=T/4.
0897          IF (RO-100.)230,230,220
0898      220 I100=0
0899          RETURN
0900      230 I100=I100+1
0901          IF(RO-1.)240,240,250
0902      240 ICV=0
0903          RETURN
0904      250 IF(I100-5)290,260,260
0905      260 IF(O-T)280,270,270
0906      270 ICV=1
0907      280 RETURN
0908          END
```

```
0909          SUBROUTINE RSIEL(NO,A,B1,B2,N,KS),RESOLUCION DEL SISTEMA
0910          DIMENSION A(1),B1(1),B2(1)
0911          TOL=1.E-20
0912          KS=0
0913          JJ=-N
0914          IF (1.GT.N)GO TO 5
0915          DO 90 J=1,N
0916          JY=J+1
0917          JJ=JJ+N+1
0918          BIGA=0.
0919          IT=JJ-J
0920          IF(J.GT.N)GO TO 15
0921          DO 20 I=J,N
0922    C
0923    C     BUSQUEDA DEL MAXIMO COEFICIENTE DE LA COLUMNA
0924    C
0925          IJ=IT+I
0926          IF (ABS(BIGA)-ABS(A(IJ)))10,20,20
0927       10 BIGA=A(IJ)
0928          IMAX=I
0929       20 CONTINUE
0930       15 CONTINUE
0931    C
0932    C     PRUEBA DE SI EL PIVOTE ES MENOR QUE LA TOLERANCIA(MATRIZ SINGULAR
0933    C
0934          IF (ABS(BIGA)-TOL) 30,30,40
0935       30 KS=1
0936          WRITE(NO,1)
0937        1 FORMAT(5X,"MATRIZ SINGULAR",)
0938          RETURN
0939    C
0940    C     INTERCAMBIO DE FILAS SI ES NECESARIO
0941    C
0942       40 I1=J+N*(J-2)
0943          IT=IMAX-J
0944          IF (J.GT.N)GO TO 25
0945          DO 50 K=J,N
0946          I1=I1+N
0947          I2=I1+IT
0948          SAVE=A(I1)
0949          A(I1)=A(I2)
0950          A(I2)=SAVE
0951    C
0952    C
0953       50 A(I1)=A(I1)/BIGA
0954       25 CONTINUE
0955          SAVE=B1(IMAX)
0956          B1(IMAX)=B1(J)
0957          B1(J)=SAVE/BIGA
0958          SAVE=B2(IMAX)
0959          B2(IMAX)=B2(J)
0960          B2(J)=SAVE/BIGA
0961    C
0962    C     ELIMINACION DE LA VARIABLE SIGUIENTE
0963    C
```

PAGE 0024 RSIEL 5:56 AM SAT., 16 JAN., 1982

```
0964          IF (J-N)60,90,60
0965       60 IQS=N*(J-1)
0966          DO 80 IX=JY,N
0967          IXJ=IQS+IX
0968          IT=J-IX
0969          IF (JY.GT.N)GO TO 35
0970          DO 70 JX=JY,N
0971          IXJX=N*(JX-1)+IX
0972          JJX=IT+JX
0973       70 A(IXJX)=A(IXJX)-(A(IXJ)*A(JJX))
0974       35 CONTINUE
0975          B1(IX)=B1(IX)-(B1(J)*A(IXJ))
0976       80 B2(IX)=B2(IX)-(B2(J)*A(IXJ))
0977        5 CONTINUE
0978  C
0979  C
0980       90 IF (N-1) 120,120,100
0981      100 NY=N-1
0982          IT=N*N
0983          IF (1.GT.NY)GO TO 45
0984          DO 110 J=1,NY
0985          IA=IT-J
0986          IB=N-J
0987          IC=N
0988          DO 110 K=1,J
0989          B1(IB)=B1(IB)-A(IA)*B1(IC)
0990          B2(IB)=B2(IB)-A(IA)*B2(IC)
0991          IA=IA-N
0992      110 IC=IC-1
0993       45 CONTINUE
0994  C
0995      120 RETURN
0996          END
```

FTN4 COMPILER: HP92060-16032 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00572 COMMON = 00000

```
PAGE 0001  FTN.   6:24 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM OPPI2(5)
0003   C
0004   C     SEGUNDO SEGMENTO DEL PROGRAMA CIOPT.ROTACION DE COORDENADAS
0005   C
0006         COMMON#/COMUN/NI,NO,TITU(30),ICOMM,P,NVE L,NP,NTP
0007         COMMON#/OPROG/KOUNT,CIMPR,M,MMTER,PR,ISTRA,YLED,E(50)
0008         COMMON#/VECES/X(50)
0009         COMMON//IW(1)
0010         INTEGER P,PR,R,C,CIMPR
0011         REAL LC
0012         LOGICAL CIMP
0013         CIMP=.FALSE.
0014         IF(CIMPR.EQ.0)CIMP=.TRUE.
0015         DO 10 I=1,ICOMM
0016    10   IW(I)=0
0017   C
0018   C     DETERMINACION DE LOS INDICES DE LA MEMORIA DINAMICA
0019   C
0020         N4=1+2*P*P
0021         N5=N4+2*P
0022         N6=N5+2*P
0023         N7=N6+2*L
0024         N8=N7+2*L
0025         N9=N8+2*L
0026         N10=N9+2*L
0027         N11=N10+2*L*L
0028         N12=N11+2*L*L
0029         N13=N12+2*L
0030         N15=N13+2*L*L
0031         N16=N15+2*P
0032         N17=N16+2*L
0033   C
0034   C     COMPRUEBA SI HAY MEMORIA SUFICIENTE
0035   C
0036         IF(N17.GT.ICOMM)GOTO 2
0037   C     LLAMADA A LA SUBRUTINA QUE REALIZA EL ALGORITMO
0038   C
0039         CALL ALROS(P,L,TITU,CIMP,NI,NO,X,IW(1),IW(N4),IW(N5),
0040        *IW(N6),IW(N7),IW(N8),IW(N9),IW(N10),IW(N11),IW(N12),IW(N13),
0041        *IW(N15),IW(N16))
0042         GOTO 3
0043   C
0044   C     ESCRIBE UN MENSAJE SI NO HAY MEMORIA SUFICIENTE
0045   C
0046    2    WRITE(NO,4)N17
0047    4    FORMAT(5X,' SE SUPERA LA CAPACIDAD ACTUAL N17= ',I6)
0048   C
0049   C     DEVUELE EL CONTROL AL PROGRAMA PRINCIPAL CIOPT
0050   C
0051    3    CALL OVRTN
0052         END
```

PAGE 0003 FTN. 6:24 AM SAT., 16 JAN., 1982

```
0053          SUBROUTINE ALROS(P,L,TITU,CIMP,NI,NC,X,S,SA,D,G,H,
0054         *AL,PH,A,B,DX,VV,EANT,VM)
0055          COMMON/OPROC/KOUNT,CIMPR,N,NMTER,PR,ICTPA,YLED,E(50)
0056          COMMON//IW(1)
0057          DIMENSION X(1),V(1,1), SA(1), D(1), G(1), H(1), AL(1),
0058         1 PH(1), A(1,1), B(1,1), DX(1), VV(1,1), EANT(1), VM(1),
0059         2TITU(30)
0060          INTEGER P,PR,R,C,CIMPR
0061          REAL LC
0062          LOGICAL CIMP
0063          WRITE(NO,13)TITU
0064       13 FORMAT (1H1 69(1H*),/,1X,1H*,14X," ALGORITMO DE ROSENBROCK ",28X,
0065         *1H*,/,1X,1H*,3X,30A2,4X,1H*,/,1X,69(1H*),2/)
0066       30 WRITE(NO,2)N,P,L,ICTPA,NMTER,PR,CIMPR
0067        2  FORMAT(5X,"TIPO DE PROBLEMA =",I3,"SI +1 MAX SI -1 MIN",/,5X,"NUME
0068         *0 DE VARIABLES INDEPENDIENTES =",I3,/,5X,"NUMERO TOTAL DE RESTRIC
0069         *IONES =",I3 /,5X,"CONTROL DE T. DEL INCREMENTO  =",I3,/,5X,"NUMER
0070         * MAXIMO DE ITERACIONES =",I5,/,5X,"INTERVALO DE IMPRESION = ",I5,
0071         *5X,"CODIGO DE IMPRESION = ",I2,2/,10X,"VECTOR INICIAL ",/)
0072          WRITE(NO,1009)(LS,X(LS),LS=1,P)
0073      1009 FORMAT(10X,"X(",I2,")=",1PE15.7)
0074          WRITE(NO,1002)
0075          WRITE(NO,1001)(LS,E(LS),LS=1,P)
0076      1002 FORMAT(1X,2/,10X,"VECTOR INCR-INICIAL",/)
0077      1001 FORMAT(10X,"E(",I2,")=",1PE15.7)
0078          INDIC=PR-1
0079          ICIC=0
0080          LW%=0
0081          KILI=0
0082          KOUNT = 0
0083          TERM =0.0
0084          F1 = 0.0
0085          N=L
0086          DO 40  K=1,L
0087       40 AL(K) = (CH(X,N,K)-CG(X,N,K))*.0001
0088          DO 60  I=1,P
0089          DO 60  J=1,P
0090          V(I,J) = 0.0
0091          IF (I-J) 60,61,60
0092       61 V(I,J) = 1.0
0093       60 CONTINUE
0094          DO 65 KK = 1,P
0095          EANT(KK) = E(KK)
0096       65 CONTINUE
0097     1000 DO 70  J=1,P
0098          IF (ICTPA.EQ.0)  E(J) = EANT(J)
0099          SA(J) = 2.0
0100       70 D(J) = 0.0
0101          FNEJ  = F1
0102       80 I = 1
0103          IF (KILI.EQ.0) GO TO 120
0104       90 DO 110  K=1,P
0105      110 X(K) = X(K) + E(I)*V(I,K)
0106          DO 50  K=1,L
0107       50 H(K) = F0
```

```
0108      120 F1 = F(X,N)
0109          F1 = M*F1
0110          IF (LUX.EQ.0) F0 = F1
0111          LUX = 1
0112          IF (ABS(FME-F1)-YLED) 122,122,125
0113      122 TERM = 1.0
0114          GO TO 450
0115      125 CONTINUE
0116          J = 1
0117      130 XC = CX(K,N,J)
0118          LC = CC(K,N,J)
0119          UC = CW(K,N,J)
0120          IF(XC.LE.LC) GO TO 420
0121          IF (XC.GE.UC) GO TO 420
0122          IF (F1.LT.F0) GO TO 420
0123          IF (XC.LT.LC+AL(J))  GO TO 140
0124          IF (XC.GT.UC-AL(J))  GO TO 140
0125          H(J) = F0
0126          GO TO 210
0127      140 CONTINUE
0128          BW = AL(J)
0129          IF (XC.LE.LC.OR.UC.LE.XC) GO TO 150
0130          IF (LC.LT.XC.AND.XC.LT.LC+BW) GO TO 160
0131          IF (UC-BW.LT.XC.AND.XC.LT.UC) GO TO 170
0132          PW (J) = 1.0
0133          GO TO 210
0134      150 PW(J) = 0.0
0135          GO TO 190
0136      160 PW = (LC+BW-XC)/BW
0137          GO TO 180
0138      170 PW = (XC-UC+BW)/BW
0139      180 PW(J) = 1.0-3.0*PW+4.0*PW*PW-2.0+PW*PW*PW
0140      190 F1 = H(J)+(F1-H(J))*PW(J)
0141      210 CONTINUE
0142          IF (J.EQ.L) GO TO 220
0143          J = J+ 1
0144          GO TO 130
0145      220 KILI = 1
0146          IF (F1.LT.F0) GO TO 420
0147          D(I) = D(I) + E(I)
0148          E(I) = 3.0 * E(I)
0149          F0 = F1
0150          IF (SA(I).GE.1.5)  SA(I) = 1.0
0151      230 DO 240 JJ=1,P
0152          IF (SA(JJ).GE.0.5) GO TO 440
0153      240 CONTINUE
0154          DO 250 R=1,P
0155          DO 250 C=1,P
0156      250 VV(C,R) = 0.0
0157          DO 260 R=1,P
0158          KR = R
0159          DO 260 C=1,P
0160          DO 265 K=KR,P
0161      265 VV(R,C) = D(K) * V(K,C) + VV(R,C)
0162      260 D(R,C) = VV(R,C)
```

```
0183          BMAG = 0.0
0184          DO 290 C=1,P
0185          BMAG = BMAG + B(1,C)*B(1,C)
0186      290 CONTINUE
0187          BMAG = SQRT(BMAG)
0188          BX(1) = BMAG
0189          DO 310 C=1,P
0190      310 V(1,C) = B(1,C)/BMAG
0191          DO 390 R=2,P
0192          IR = R-1
0193          DO 390 C=1,P
0194          ADICH = 0.0
0195          DO 320 KK=1,IR
0196          ADICV= 0.0
0197          DO 330 KJ=1,P
0198      330 ADICV = ADICV + VV(R,KJ)*V(KK,KJ)
0199      320 ADICH = ADICV*V(KK,C) + ADICH
0200      390 B(R,C) = VV(R,C) - ADICH
0201          DO 340 R=2,P
0202          BBMAG = 0.0
0203          DO 350 K=1,P
0204      350 BBMAG = BBMAG + B(R,K)*B(R,K)
0205          BBMAG = SQRT(BBMAG)
0206          DO 340 C=1,P
0207      340 V(R,C) = B(R,C)/BBMAG
0208          ICIC = ICIC+1
0209          INDIC = INDIC+1
0210          IF (INDIC.EQ.PR) GO TO 450
0211          GO TO 1000
0212      420 IF (KILI.EQ.0) GO TO 450
0213          DO 430 IX=1,P
0214      430 X(IX) = X(IX)-E(I)*V(I,IX)
0215          E(I) = -0.5*E(I)
0216          IF (SA(I).LT.1.5) SA(I)=0.0
0217          GO TO 230
0218      440 CONTINUE
0219          IF (I.EQ.P) GO TO 80
0220          I=I+1
0221          GO TO 30
0222      450 IF(CIMP)GOTO 451
0223          WRITE (NO,3)
0224        3 FORMAT(1X,/,2X,"ITERACION",6X,"FUNCION",12X,"PROGRESO",9X,
0225         1"PROGRESO LATERAL")
0226          WRITE (NO,4) ICIC, FO, BMAG, BBMAG
0227        4 FORMAT(2X,I5,3E20.8)
0228          WRITE (NO,14) KOUNT
0229       14 FORMAT (/,2X,"NUMERO DE VECES QUE SE CALCULA LA FUNCION = ",I8)
0230          WRITE (NO,5)
0231        5 FORMAT(/,10X,"VECTOR DE ESTADO",/)
0232          WRITE (NO,6) (JM, X(JM), JM=1,P)
0233        6 FORMAT (10X,"X(",I2,")=",1PE15.7)
0234      451 INDIC = 0
0235          IF (KILI.EQ.0) GO TO 470
0236          IF (TERM.EQ.1.0) GO TO 480
0237          IF (ICIC.GE.MAXTER) GO TO 480
```

PAGE 0006 ALRDS 6:24 AM SAT., 16 JAN., 1982

```
0218          GO TO 1000
0219      470 WRITE (ND,7)
0220        7 FORMAT (3/,2X,"EL PUNTO INICIAL DEBE CUMPLIR LAS RESTRICCIONES.CO
0221       1PRUEBE SI ES CIERTO ")
0222      480 CONTINUE
0223      490 IF(CIMP)GOTO 491
0224          WRITE (ND,8)
0225        8 FORMAT (2/,2X,"MATRI/ DEL VECTOR DIRECCION FINAL ",/)
0226          DO 500 J=1,P
0227      500 WRITE (ND,9)  (J, I, V(J,I), I=1,P)
0228        9 FORMAT(10X,"V(",I2,",",I2,")=",1PE15.7)
0229          WRITE (ND,11)
0230       11 FORMAT(2/,2X,"VALOR FINAL DEL INTERVALO",/)
0231          WRITE (ND,12)  (J, E(J), J=1,P)
0232       12 FORMAT (10X,"S(",I2,")=",1PE15.7)
0233      491 WRITE(ND,492)
0234      492 FORMAT(1H1,15X,21(1H#),/,18X,"RESULTADO FINAL",/,15X,21(1H*),2/)
0235          WRITE(ND,493)F0,ICIC
0236      493 FORMAT(10X,"FUNCION OBJETIVO = ",1PE13.7,/,10X,"ITERACIONES = ",
0237         *I5,2/)
0238          WRITE(ND,494)
0239      494 FORMAT(10X,"VECTOR FINAL",/)
0240          WRITE(ND,6)(JM,X(JM),JM=1,P)
0241          DO 1570 I=1,P
0242          E(I)=SANT(I)
0243     1570 CONTINUE
0244          RETURN
0245          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 02085 COMMON = 00001

```
PAGE 0001  FTN.   6:30 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         SUBROUTINE FUOBJ(X,FF,G)
0003         COMMON/COMUN/LECT,IMPR,TITU(30),ICOMN,NOVE,NVE,NREST,NP,NTP
0004         COMMON/OPMVP/IFUNC,N,M
0005         DIMENSION X(1),G(1)
0006         VAL=0.0
0007         DO 100 I=NP,NTP,NP
0008         FUN=FUNC(X,I)
0009         VAL=VAL+FUN
0010    100  CONTINUE
0011         FF=VAL
0012         IF(IFUNC)40,40,60
0013    60   NTCC=NTP/NP
0014         WRITE(IMPR,70)NTCC,NTP,FF
0015         WRITE(IMPR,50)(I,X(I),I=1,N)
0016    70   FORMAT(4X,"NP =",I5,2X,"NTP =",I5,2X,"FU-OBJ =",1PE15.7,/,5X,
0017        *"VECTOR DE ESTADO",/)
0018    50   FORMAT(10X,"X(",I2,")=",1PE15.7)
0019    40   RETURN
0020         END


 FTN4 (HP) HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00150       COMMON = 00000
```

```
 PAGE 0002  FTN.   6:30 AM  SAT., 16  JAN., 1982

0021          SUBROUTINE ACODI(Z,N)
0022          DIMENSION Z(50)
0023          RETURN
0024          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00003        COMMON = 00000
```

```
PAGE 0003  FTN.   6:30 AM  SAT., 16  JAN., 1982


0025          SUBROUTINE ACONT(X,XRAW)
0026          COMMON/COMUN/LECT,IMPR,TITU(30)
0027          COMMON/SPRVP/IFUNC,N,M
0028          COMMON//IW(1)
0029          DIMENSION X(1),XRAW(1)
0030          DO 125 I=1,M
0031          X(I)=XRAW(I)
0032      125 CONTINUE
0033          RETURN
0034          END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00033       COMMON = 00001
```

```
PAGE 0004  FTN.   5:30 AM  SAT., 16  JAN., 1982


0035          FUNCTION F(X,N)
0036          COMMON/COMUN/NI,NO,TITU(30),ICOMN,NCVEP,NVE,L,NP,NTP
0037          COMMON/OPROC/KOUNT
0038          DIMENSION X(1)
0039          VAL=0.0
0040          DO 100 I=NP,NTP,NP
0041          VAL=VAL+FUNC(X,I)
0042      100 CONTINUE
0043          F=VAL
0044          KOUNT = KOUNT + 1
0045          RETURN
0046          END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00047       COMMON = 00000
```

PAGE 0005 FTN. 6:30 AM SAT., 16 JAN., 1982

```
0047          FUNCTION CX(X,N,K)
0048          DIMENSION X(1)
0049          CX = X(K)
0050          RETURN
0051          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00020 COMMON = 00000

```
PAGE 0006  FTN.   6:30 AM  SAT., 16  JAN., 1982


0052          FUNCTION CG(X,N,K)
0053          DIMENSION X(1)
0054          CG = 0.0
0055          RETURN
0056          END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00017       COMMON = 00000
```

```
 PAGE 0007  FTN.   5:30 AM  SAT., 16  JAN., 1982


0057          FUNCTION CH(X,N,K)
0058          DIMENSION X(1)
0059          CH=20.00
0060          RETURN
0061          END


  FTN4 (COMPILER) HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00017       COMMON = 00000
```

APENDICE E

PAGE 0001 FTN. 7:03 AM SAT., 16 JAN., 1982

```
0001   FTN4
0002         PROGRAM ACUSC
0003   C*************************************************************
0004   C**** PROGRAMA PARA APROXIMAR UNA FUNCION CON SPLINE CUBICOS *
0005   C*************************************************************
0006   C
0007   C
0008   C      ENTRADA DE DATOS:
0009   C   ************************
0010   C
0011   C      TITU= TITULO DEL PROBLEMA.FORMATO 30A2
0012   C      I=DIMENSION.DOS O TRES DIMENSIONES
0013   C      N=NUMERO DE PUNTOS DE PASO
0014   C      C1=TIPO DEL EXTREMO INICIAL
0015   C      C2=TIPO DEL EXTREMO FINAL
0016   C      IPINT=INDICE PARA LA GENERACION DE TANTOS PUNTOS COMO SE INDI-
0017   C            QUE ENCADA INTERVALO
0018   C      INPES=INDICE PARA TOMAR LA CUERDA O NO
0019   C      Z(NS)=CONJUNTO DE VALORES QUE INDICA EL NUMERO DE PUNTOS QUE
0020   C      SE GENERAN EN CADA INTERVALO
0021   C      P(I,J)=PUNTOS DE PASO
0022   C      U( , )=TANGENTE ENEL EXTREMO
0023   C      L(LN)=CONJUNTO DE VALORES DE L SI ES QUE LOS TIENE QUE LEER
0024   C*************************************************************
0025         INTEGER C1,C2,Z(17)
0026         REAL M(18,4),L(17),MN(17,17)
0027         DIMENSION P(3,18),B(3,18),U(3,18),C(3,360),IP(5),F(4,3),R(3)
0028        *,V(18,18),S(18,36),CC(17),W(17),TITU(30),RP(3),CP(3,360)
0029         CALL EMPAR(IP)
0030         IO=IP(1)
0031         IS=IP(2)
0032         IN=IP(3)
0033         IK=IP(4)
0034         IM=IP(5)
0035    1024 IF(IK.EQ.IN)STOP
0036         IK=IK+1
0037         READ(IO,1029)TITU
0038    1029 FORMAT(30A2)
0039         READ(IO,*)I,N,C1,C2,IPINT,INPES
0040         NN=N-1
0041         IF(IPINT.LT.0)GOTO 1576
0042         READ(IO,*)(Z(NS),NS=1,NN)
0043    1576 CONTINUE
0044         IF(I.EQ.3)GOTO 1000
0045         DO  126 JS=1,N
0046         READ(IO,*)P(1,JS),P(2,JS)
0047     126 CONTINUE
0048         IF(C1.NE.2)GOTO 1001
0049         READ(IO,*)U(1,1),U(2,1)
0050         IF(C2.NE.2)GOTO 1001
0051         READ(IO,*)U(1,N),U(2,N)
0052         GOTO 1001
0053    1000 DO 5 J=1,N
0054         READ(IO,*)P(1,J),P(2,J),P(3,J)
0055       5 CONTINUE
```

```
PAGE 0002  ACUSC  7103 AM  SAT., 16  JAN., 1982


0056          IF(C1.NE.2)GOTO 1001
0057          READ(IO,*)U(1,1),U(2,1),U(3,1)
0058          IF(C2.NE.2)GOTO 1001
0059          READ(IO,*)U(1,N),U(2,N),U(3,N)
0060     1001 IF(INPES.EQ.0)GOTO 1002
0061          NX=N-1
0062          DO 1003 LN=1,NX
0063          READ(IO,*)L(LN)
0064     1003 CONTINUE
0065     1002 CONTINUE
0066          NI=N
0067          DO 1575 LK=1,NN
0068          IF(IPINT.NE.0)GOTO 1575
0069          LKK=LK+1
0070          Z(LK)=P(1,LKK)-P(1,LK)-1
0071     1575 NI=NI+Z(LK)
0072          N2=NN*2
0073          WRITE(IS,50)TITU
0074       50 FORMAT(1H1,70(1H*),/,1X,5(1H*),30A2,5(1H*),/,1X,70(1H*),//)
0075          WRITE(IS,57)I,N,C1,C2,(Z(NX),NX=1,NN)
0076       57 FORMAT(5X,61(1H*),/,5X,
0077         *"DIMENSION=",I2,/,5X,"NUMERO DE PUNTOS=",I3,/,5X,
0078         *"TIPO EXTREMO INICIAL=",I2,/,5X,"TIPO EXTREMO FINAL=",I2,/,
0079         *5X,"NUMERO DE PUNTOS INTERMEDIOS=",17I4,/5X,61(1H*),//,5X,
0080         *"PUNTO",20X,"X",18X,"Y",18X,"Z",2/)
0081          DO 6 J=1,N
0082          WRITE(IS,59)J,P(1,J),P(2,J),P(3,J)
0083        6 CONTINUE
0084       59 FORMAT(6X,I2,10X,3(5X,1PE14.6))
0085          CALL SPLIN(I,N,P,C1,C2,H,B,L,Z,U,C,NN,NI,F,R,V,W,S,CC,N2,
0086         1NN,INPES,RP,CP)
0087          WRITE(IS,100)
0088      100 FORMAT(1H1,5X,"SPLINE CUBICO.RESULTADOS",3/,5X,"PUNTO",20X,"X",
0089         *18X,"Y",18X,"Z",10X,"XP",10X,"YP",10X,"ZP",2/)
0090          DO 7 K=1,NI
0091          DYX=CP(2,K)/CP(1,K)
0092          DZX=CP(3,K)/CP(1,K)
0093          WRITE(IS,101)K,C(1,K),C(2,K),C(3,K),DYX,DZX,CP(1,K)
0094        7 CONTINUE
0095      101 FORMAT(3X,I3,5X,6(2X,1PE14.6))
0096          GOTO 1024
0097          END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **    PROGRAM = 08125       COMMON = 00000
```

```
PAGE 0005  FTN.   7:03 AM  SAT., 16  JAN., 1982


0000          SUBROUTINE SPLIN(I,N,P,C1,C2,M,B,L,Z,U,C,NN,NI,F,R,V,W,S,CC,N2,
0000         1NN,INPES,RP,CP)
0100          INTEGER C1,C2,Z(NN)
0101          REAL M(N,4),L(NN),NN(NN,NP)
0102          DIMENSION P(3,N),B(3,N),U(3,N),C(3,NI),F(4,3),R(3),RP(3),CP(3,NI)
0102         *V(NN,NP),W(NN),S(NN,N2),CC(NN)
0104          DO 1575 LK=1,NN
0105     1575 Z(LK)=Z(LK)+1
0106          IF(C1.GT.2)GOTO 560
0107          CALL EXTDI(I,N,P,M,B,L,Z,U,C1,C2,NN,INPES)
0108          CALL ELIGA(I,N,P,M,B,L,U,NN)
0109          GOTO 590
0110      560 CALL EXTCI(I,N,P,NN,B,L,U,C1,Z,NN,NI,S,N2,CC,W,V,INPES)
0111      590 CALL CENCU(I,N,P,L,Z,U,C,NN,NI,F,R,RP,CP)
0112          RETURN
0113          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00131       COMMON = 00000
```

```
PAGE 0007  FTN.   7103 AM  SAT., 16  JAN., 1982


0114   C
0115         SUBROUTINE EXTDI(I,N,P,M,B,L,Z,U,C1,C2,NN,INPES)
0116         INTEGER C1,C2,Z(NN)
0117         REAL M(N,4),L(NN)
0118         DIMENSION P(3,N),B(3,N),U(3,N)
0119         IF(C1.EQ.1)GOTO 660
0120         M(1,2)=1.
0121         M(1,3)=0.
0122         DO 660 K=1,I
0123         B(K,1)=U(K,1)
0124    660  CONTINUE
0125         GOTO 700
0126    680  M(1,2)=1.
0127         M(1,3)=0.5
0128    700  DO 770 J=1,NN
0129         IF(I.EQ.3)GOTO 760
0130         JJ=J+1
0131         IF(INPES.NE.0)GOTO 761
0132         L(J)=SQRT((P(1,JJ)-P(1,J))**2+(P(2,JJ)-P(2,J))**2)
0133    761  GOTO 770
0134    760  IF(INPES.NE.0)GOTO 770
0135         L(J)=SQRT((P(1,J+1)-P(1,J))**2+(P(2,J+1)-P(2,J))**2+(P(3,J+1)-P(3
0136        +,J))**2)
0137    770  CONTINUE
0138         IF(C1.EQ.2)GOTO 820
0139         DO 810 K=1,I
0140         B(K,1)=(3/(2*L(1)))*(P(K,2)-P(K,1))
0141    810  CONTINUE
0142    820  IF(C2.EQ.1)GOTO 890
0143         M(N,1)=0.
0144         M(N,2)=1.
0145         DO 870 K=1,I
0146         B(K,N)=U(K,N)
0147    870  CONTINUE
0148         GOTO 940
0149    890  M(N,1)=2.
0150         M(N,2)=4.
0151         DO 930 K=1,I
0152         B(K,N)=(6/L(NN))*(P(K,N)-P(K,NN))
0153    930  CONTINUE
0154    940  RETURN
0155         END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

**   NO WARNINGS **  NO ERRORS **    PROGRAM = 00541       COMMON = 00000
```

```
 PAGE 0005  FTN.   7:03 AM  SAT., 16  JAN., 1982


0156   C
0157         SUBROUTINE FLIGA(I,H,P,M,B,L,U,NN)
0158         REAL M(N,4),L(NN)
0159         DIMENSION P(3,N),B(3,N),U(3,N)
0160         DO 1040 J=2,NN
0161         M(J,1)=L(J)
0162         M(J,2)=2*(L(J)+L(J-1))
0163         M(J,3)=L(J-1)
0164         DO 1040 K=1,I
0165         B(K,J)=3*(L(J-1)**2*(P(K,J+1)-P(K,J))+L(J)**2*(P(K,J)-P(K,J-1)))
0166         B(K,J)=B(K,J)/(L(J)*L(J-1))
0167  1040   CONTINUE
0168         DO 1180 II=2,N
0169         IF(M(II,1).EQ.0.)GOTO 1180
0170         D=M(II-1,2)/M(II,1)
0171         DO 1121 K=1,3
0172  1121   M(II,K)=M(II,K)*D-M(II-1,K+1)
0173         DO 1120 K=1,I
0174  1120   B(K,II)=B(K,II)*D-B(K,II-1)
0175         Q=M(II,2)
0176         DO 1181 K=1,3
0177  1181   M(II,K)=M(II,K)/Q
0178         DO 1180 K=1,I
0179  1180   B(K,II)=B(K,II)/Q
0180         DO 1220 K=1,I
0181         DO 1220 J=0,NN
0182         NA=N-J
0183         U(K,NA)=(B(K,NA)-M(NA,3)*U(K,NA+1))/M(NA,2)
0184  1220   CONTINUE
0185         RETURN
0186         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00553      COMMON = 00000
```

```
 PAGE 0011  FTN.   7:03 AM  SAT., 16  JAN., 1982


0187   C
0188         SUBROUTINE GENCU(I,N,P,L,Z,U,C,NN,NI,F,R,RP,CP)
0189         INTEGER Z(NN)
0190         REAL L(NN)
0191         DIMENSION P(3,N),U(3,N),C(3,NI),F(4,3),R(3),RP(3),CP(3,NI)
0192         II=1
0193         DO 1500 J=1,NN
0194         DO 1330 K=1,I
0195         F(1,K)=P(K,J)
0196         F(2,K)=U(K,J)
0197         F(3,K)=(3/L(J)**2)*(P(K,J+1)-P(K,J))-(1/L(J))*(U(K,J+1)+2*U(K,J))
0198         F(4,K)=((-2)/L(J)**3)*(P(K,J+1)-P(K,J))+(1/L(J)**2)*(U(K,J+1)+
0199        +U(K,J))
0200    1330 CONTINUE
0201         T=0.
0202    1220 IF(J.EQ.1)GOTO 1380
0203         IF(T)1380,1490,1380
0204    1380 DO 1400 K=1,I
0205         R(K)=F(1,K)-F(2,K)*T+F(3,K)*(T**2)+F(4,K)*(T**3)
0206         RP(K)=F(2,K)+2*F(3,K)*T+3*F(4,K)*(T**2)
0207    1400 CONTINUE
0208         IF(I.EQ.3)GOTO 1450
0209         C(1,II)=R(1)
0210         C(2,II)=R(2)
0211         CP(1,II)=RP(1)
0212         CP(2,II)=RP(2)
0213         GOTO 1480
0214    1450 C(1,II)=R(1)
0215         C(2,II)=R(2)
0216         C(3,II)=R(3)
0217         CP(1,II)=RP(1)
0218         CP(2,II)=RP(2)
0219         CP(3,II)=RP(3)
0220    1480 II=II+1
0221    1490 A2=Z(J)+1.0
0222         T=L(J)/A2+T
0223         TDL=T-L(J)
0224         TD=L(J)/A2*0.1
0225         IF(TDL-TD)2220,2220,1500
0226    1500 CONTINUE
0227         RETURN
0228         END


 FTN4 COMPILER) HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00183       COMMON = 00000
```

```
PAGE 0013  FTN.   7:03 AM  SAT., 16  JAN., 1982


0229  C
0230        SUBROUTINE EXTCI(I,N,P,MM,D,L,U,C1,Z,NN,NI,S,N2,C,W,V)
0231        INTEGER C1,Z(MM),N2
0232        REAL MM(MM,MM),L(MM)
0233        DIMENSION P(3,N),B(3,N),U(3,N),C(MM),W(MM),V(MM,MM),S(MM,N2)
0234        DO 1 J=1,MM
0235        C(J)=0.
0236        W(J)=0.
0237        DO 1 JJ=1,MM
0238        MM(J,JJ)=0.
0239        V(J,JJ)=0.
0240     1  CONTINUE
0241        DO 1630 J=1,NN
0242        IF(I.EQ.3)GOTO 1620
0243        IF(INPES.NE.0)GOTO 1621
0244        L(J)=SQRT((P(1,J+1)-P(1,J))**2+(P(2,J+1)-P(2,J))**2)
0245     1621 GOTO 1630
0246     1620 IF(INPES.NE.0)GOTO 1630
0247        L(J)=SQRT((P(1,J+1)-P(1,J))**2+(P(2,J+1)-P(2,J))**2+(P(3,J+1)-P(
0248       *3,J))**2)
0249     1630 CONTINUE
0250        S3=L(NN)/L(1)
0251        IF(C1.EQ.4)GOTO 1730
0252        MM(1,1)=2+2*S3
0253        MM(1,2)=S3
0254        MM(1,NN)=1
0255        DO 1710 K=1,I
0256        B(K,1)=(3/L(1))*(S3*(P(K,2)-P(K,1))+(1/S3)*(P(K,N)-P(K,NN)))
0257     1710 CONTINUE
0258        GOTO 1800
0259     1730 MM(1,1)=2+2*S3
0260        MM(1,2)=S3
0261        MM(1,NN)=-1
0262        DO 1750 K=1,I
0263        B(K,1)=(3/L(1))*(S3*(P(K,2)-P(K,1))-(1/S3)*(P(K,N)-P(K,NN)))
0264     1750 CONTINUE
0265     1800 DO 1850 J=2,NN
0266        MM(J,J-1)=L(J)
0267        MM(J,J)=2*(L(J)+L(J-1))
0268        DO 1850 K=1,I
0269        B(K,J)=3*(L(J-1)**2*(P(K,J+1)-P(K,J))+L(J)**2*(P(K,J)-P(K,J-1)))
0270        B(K,J)=B(K,J)/(L(J)*L(J-1))
0271     1850 CONTINUE
0272        CALL  INMAF(NN,N2,MM,V,S)
0273        DO 1950 K=1,I
0274        DO 1950 J=1,NN
0275        C(J)=B(K,J)
0276     1950 CONTINUE
0277        CALL  MULM(NN,V,C,W)
0278        DO 1960 J=1,NN
0279        U(K,J)=W(J)
0280     1960 CONTINUE
0281        IF(C1.EQ.4)GOTO 2040
0282        DO 2020 K=1,I
0283        U(K,N)=U(K,1)
```

```
PAGE 0014  ENTCI  7103 HP  SAT., 16  JAN., 1982

0284     2020 CONTINUE
0285          GOTO 2070
0286     2040 DO 2060 K=1,I
0287          U(K,H)=-U(K,1)
0288     2060 CONTINUE
0289     2070 RETURN
0290          END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790630)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00947        COMMON = 00000
```

PAGE 0016 FTN. 7:03 AM SAT., 16 JAN., 1982

```
0291   C
0292          SUBROUTINE INMAT(N,N2,AM,V,S)
0293          INTEGER N,N2
0294          DIMENSION AM(N,N),V(N,N),S(N,N2)
0295          DO 1 I=1,N
0296          DO 1 J=1,N
0297          S(I,J)=AM(I,J)
0298      1   CONTINUE
0299          I=1.
0300          NX=N+1
0301          NY=2*N
0302          DO 80 J=NX,NY
0303          S(I,J)=1.
0304          I=I+1
0305       80 CONTINUE
0306          L=1
0307          K=2
0308      110 XM=S(L,L)
0309          DO 140 J=L,NY
0310          S(L,J)=S(L,J)/XM
0311      140 CONTINUE
0312          DO 190 I=K,N
0313          X=S(I,L)
0314          DO 190 J=L,NY
0315          S(I,J)=S(I,J)-S(L,J)*X
0316      190 CONTINUE
0317          L=L+1
0318          K=K+1
0319          IF(L-N)110,110,230
0320      230 L=N
0321      235 LZ=L-1
0322          DO 290 K=1,LZ
0323          I=L-K
0324          Y=S(I,L)
0325          DO 290 J=L,NY
0326          S(I,J)=S(I,J)-S(L,J)*Y
0327      290 CONTINUE
0328          L=L-1
0329          IF(L-1)320,320,235
0330      320 DO 2 I=1,N
0331          DO 2 J=NX,NY
0332          L=NX-N
0333          V(I,L)=S(I,J)
0334        2 CONTINUE
0335          RETURN
0336          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00353 COMMON = 00000

```
PAGE 0018  FTN.   7:03 AM  SAT., 16  JAN., 1982


0337   C
0338         SUBROUTINE MULMCN,V,C,W)
0339         DIMENSION V(N,N),C(N),W(N)
0340         DO 10 I=1,N
0341         W(I)=0.
0342         DO 10 J=1,N
0343         W(I)=V(I,J)*C(J)+W(I)
0344      10 CONTINUE
0345         RETURN
0346         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00081       COMMON = 00000
```

APENDICE F

```
PAGE 0001  FTN.   6:43 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         PROGRAM PERSP
0003   C
0004   C    PROGRAMA PARA DIBUJAR EN PERSPECTIVA
0005   C
0006         REAL P(400,3),P2(400,2)
0007         INTEGER LB(400,2),IP(5),NT(400)
0008         DIMENSION TITU(30),ANG(3),NL(3),IL(3)
0009         DATA IL/2H X,2H Y,2H Z/
0010         CALL RKPAR(IP)
0011         IN=IP(1)
0012         IO=IP(2)
0013         TA=IP(3)*0.001
0014         LK=IP(4)
0015         LS=IP(5)
0016         READ(IN,2)TITU
0017         WRITE(IO,3)TITU
0018         READ(IN,*)N,NB,NIN,NFI,INCR
0019         WRITE(IO,4)N,NB,NIN,NFI,INCR
0020       2 FORMAT(30A2)
0021       3 FORMAT(5X,30A2,2/)
0022       4 FORMAT(5X,"NUMERO DE NUDOS=",I3,5X,"NUMERO DE BARRAS=",I3,1/,
0023        *5X,"N.INICIAL=",I3,5X,"N.FINAL=",I3,5X,"INTERVALO=",I3,2/)
0024   C
0025   C    LECTURA DE LOS DATOS DEL DIBUJO
0026   C
0027         CALL LEER(N,NB,IN,IO,P,LB,NT)
0028   C
0029   C    LLAMADA A LAS DISTINTAS SUBRUTINAS SEGUN SEA EL DIBUJO DESEADO
0030   C
0031         GO TO(101,105,106,102,103,104)LK
0032     101 CALL PLAN1(N,P,P2,IL,ANG,NL,TA)
0033         GOTO 195
0034     105 CALL PLAN2(N,P,P2,IL,ANG,NL,TA)
0035         GOTO 195
0036     106 CALL PLAN3(N,P,P2,IL,ANG,NL,TA)
0037         GOTO 195
0038     102 CALL CASA(N,P,P2,IL,ANG,NL,TA)
0039         GOTO 195
0040     103 CALL PISO(N,P,P2,IL,ANG,NL,TA)
0041         GOTO 195
0042     104 CALL PISOB(N,P,P2,IL,ANG,NL,TA)
0043   C
0044   C    REPRESENTACION EN EL PLOTER DEL DIBUJO
0045   C
0046     195 CALL REPRE(N,NB,LB,P2,TA)
0047         IF(LS)10,10,20
0048   C
0049   C    NUMERACION DE LOS PUNTOS SI ASI SE DESEA
0050   C
0051      20 CALL SINNU(N,P2,TA,NT,NIN,NFI,INCR)
0052      10 STOP
0053         END
```

```
PAGE 0003  FTN.   6:43 AM  SAT., 16  JAN., 1982


0054   C.
0055         SUBROUTINE PLAN1(N,P,P2,IL,ANG,NL,TA)
0056   C
0057   C    DIBUJA FIGURAS PLANAS EN EJES X=Y
0058   C
0059         DIMENSION P(N,3),P2(N,2),IL(3),ANG(3),NL(3)
0060         DO 100 I=1,N
0061         P2(I,1)=P(I,1)
0062         P2(I,2)=P(I,2)
0063     100 CONTINUE
0064         ANG(1)=0.0
0065         ANG(2)=90.0
0066         NL(1)=-2
0067         NL(2)=2
0068         CALL EJES(N,P,1,2,1,TA,ANG,IL,NL)
0069         RETURN
0070         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00114       COMMON = 00000
```

```
PAGE 0004  FTN.   6:43 AM  SAT., 16  JAN., 1982


0071   C
0072         SUBROUTINE PLAN2(N,P,P2,IL,ANG,NL,TA)
0073   C
0074   C    DIBUJA FIGURAS PLANAS EN EJES Y=Z
0075   C
0076         DIMENSION P(N,3),P2(N,2),IL(3),ANG(3),NL(3)
0077         DO 100 I=1,N
0078         P2(I,1)=P(I,2)
0079         P2(I,2)=P(I,3)
0080     100 CONTINUE
0081         ANG(2)=0.0
0082         ANG(3)=90.0
0083         NL(2)=-2
0084         NL(3)=2
0085         CALL EJES(N,P,2,3,1,TA,ANG,IL,NL)
0086         RETURN
0087         END


FTN4 COMPILER: HP92060-16032 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00115      COMMON = 00000
```

PAGE 0005 FTN. 6:43 AM SAT., 16 JAN., 1982

```
0088   C
0089         SUBROUTINE PLAN3(N,P,P2,IL,ANG,NL,TA)
0090   C
0091   C    DIBUJA FIGURAS PLANAS EN EJES -X=Z
0092   C
0093         DIMENSION P(N,3),P2(N,2),IL(3),ANG(3),NL(3)
0094         DO 100 I=1,N
0095         P2(I,1)=(-1.)*P(I,1)
0096         P2(I,2)=P(I,3)
0097     100 CONTINUE
0098         ANG(1)=-180.0
0099         ANG(3)=90.0
0100         NL(1)=2
0101         NL(3)=-2
0102         CALL EJES(N,P,1,3,2,TA,ANG,IL,NL)
0103         RETURN
0104         END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00138 COMMON = 00000

```
 PAGE 0006  FTN.   6:43 AM  SAT., 16  JAN., 1982


0105   C
0106   C
0107         SUBROUTINE CABA(N,P,P2,IL,ANG,NL,TA)
0108   C
0109   C    DIBUJA EN PERSPECTIVA CABALLERA EN X=Y=Z
0110   C
0111         DIMENSION P(N,3),P2(N,2),IL(3),ANG(3),NL(3)
0112         DO 100 I=1,N
0113         P(I,3)=P(I,3)*0.707107
0114         P2(I,1)=P(I,1)-P(I,3)/1.414213
0115         P2(I,2)=P2(I,1)-P(I,1)+P(I,2)
0116     100 CONTINUE
0117         ANG(1)=0.0
0118         ANG(2)=90.0
0119         ANG(3)=225.0
0120         NL(1)=-2
0121         NL(2)=2
0122         NL(3)=-2
0123         CALL EJES(N,P,1,3,1,TA,ANG,IL,NL)
0124         RETURN
0125         END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00193       COMMON = 00000
```

```
PAGE 0007  FTN.   6:43 AM  SAT., 16  JAN., 1982


0126   C
0127         SUBROUTINE PISO(N,P,P2,IL,ANG,NL,TA)
0128   C
0129   C    DIBUJA EN PERSPECTIVA ISOMETRICA EN X=Y=Z
0130   C
0131         DIMENSION P(N,3),P2(N,2),IL(3),ANG(3),NL(3)
0132         DO 100 I=1,N
0133         P2(I,1)=(P(I,1)-P(I,3))*0.8660254
0134         P2(I,2)=P(I,2)-(P(I,1)+P(I,3))*0.5
0135     100 CONTINUE
0136         ANG(1)=330.0
0137         ANG(2)=90.0
0138         ANG(3)=-150.0
0139         NL(1)=-2
0140         NL(2)=2
0141         NL(3)=2
0142         CALL EJES(N,P,1,3,1,TA,ANG,IL,NL)
0143         RETURN
0144         END


  FTN4 COMPILER> HP9206>-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00177      COMMON = 00000
```

```
PAGE 0009  FTN.   6:43 AM  SAT., 16  JAN., 1982


0145   C
0146         SUBROUTINE PISOB(N,P,P2,IL,ANG,NL,TA)
0147   C
0148   C    DIBUJA EN PERSPECTIVA EN Y=Z=X
0149   C
0150         DIMENSION P(N,3),P2(N,2),IL(3),ANG(3),NL(3)
0151         DO 100 I=1,N
0152         P2(I,1)=(P(I,2)-P(I,1))*0.8660254
0153         P2(I,2)=P(I,3)-(P(I,2)+P(I,1))*0.5
0154     100 CONTINUE
0155         ANG(1)=-150.0
0156         ANG(2)=330.0
0157         ANG(3)=90.0
0158         NL(1)=2
0159         NL(2)=-2
0160         NL(3)=2
0161         CALL EJES(N,P,1,3,1,TA,ANG,IL,NL)
0162         RETURN
0163         END


  FTN4 (COMPILER) HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00177       COMMON = 00000
```

PAGE 0009 FTN. 6:43 AM SAT., 16 JAN., 1982

```
0164   C
0165         SUBROUTINE EJES(N,P,N1,N2,N3,TA,ANG,IL,NL)
0166         DIMENSION P(N,3),ANG(3),IL(3),NL(3)
0167         CALL PLTLU(10)
0168         CALL FACT(1.)
0169         CALL PLOT(0.0,0.0,-3)
0170         ALMAX=0.0
0171         DO 20 I2=1,3
0172         DO 20 I3=1,N
0173      20 ALMAX=AMAX1(ALMAX,P(I3,I2))
0174         DO 30 I4=N1,N2,N3
0175         AL=ALMAX*TA
0176         DEL=ALMAX/AL
0177         CALL AXIS(0.,0.,IL(I4),NL(I4),AL,ANG(I4),0.,DEL)
0178      30 CONTINUE
0179         CALL PLOT(0.0,0.0,-3)
0180         RETURN
0181         END
```

FTN4 (COMPILER) HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00136 COMMON = 00000

```
PAGE 0010  FTN.   6:43 AM  SAT., 16  JAN., 1982


0182   C
0183         SUBROUTINE REPRE(N,NB,LB,P2,IA)
0184   C
0185   C    REPRESENTA EN EL PLOTER EL DIBUJO TRANSFORMADO
0186   C
0187         DIMENSION LB(NB,2),P2(N,2)
0188         CALL PLTLU(10)
0189         CALL FACT(IA)
0190         DO 100 I=1,NB
0191         IA=LB(I,1)
0192         IB=LB(I,2)
0193         CALL PLOT(P2(IA,1),P2(IA,2),3)
0194         CALL PLOT(P2(IB,1),P2(IB,2),2)
0195     100 CONTINUE
0196         CALL PLOT(0.,0.,-3)
0197         RETURN
0198         END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **    PROGRAM = 00107       COMMON = 00000
```

```
 PAGE 0001  FTN.   6:43 AM  SAT., 16  JAN., 1982


0199   C
0200   C
0201         SUBROUTINE SINNU(N,P2,TA,NT,NIN,NFI,INCR)
0202   C
0203   C    NUMERA LOS PUNTOS O NUDOS
0204   C
0205         DIMENSION P2(N,2),NT(N)
0206         CALL PLTLU(10)
0207         CALL FACT(1.)
0208         DO 100 I=NIN,NFI,INCR
0209         IF(NT(I).EQ.0)GOTO 100
0210         NT(I)=NT(I)-1
0211         B=I
0212         AX=P2(I,1)*TA
0213         BY=P2(I,2)*TA
0214         CALL SYMB(AX,BY,.08,NT(I),0.,-1)
0215         AX=AX+0.07
0216         BY=BY+0.07
0217         CALL NUMB(AX,BY,.10,B,0.,-1)
0218         AX=AX-0.07
0219         BY=BY-0.07
0220    100  CONTINUE
0221         RETURN
0222         END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00150      COMMON = 00000
```

PAGE 0012 FTN. 6:43 AM SAT., 16 JAN., 1982

```
0233   C
0234   C
0235         SUBROUTINE LEER(N,NB,IN,IO,P,LB,NT)
0236   C
0237   C    LECTURA DE LOS DATOS DE LOS PUNTOS Y LAS CONECTIVIDADES
0238   C
0239         DIMENSION P(N,3),LB(NB,2),NT(N)
0240         WRITE(IO,7)
0241         DO 10 I=1,N
0242         READ(IN,*)L,P(L,1),P(L,2),P(L,3),NT(L)
0243         WRITE(IO,6)L,P(L,1),P(L,2),P(L,3),NT(L)
0244      10 CONTINUE
0245         WRITE(IO,8)
0246         DO 20 II=1,NB
0247         READ(IN,*)L,LB(L,1),LB(L,2)
0248         WRITE(IO,11)L,LB(L,1),LB(L,2)
0249      20 CONTINUE
0250       7 FORMAT(10X,"COORDENADAS DE LOS NUDOS")
0251       6 FORMAT(12X,I3,3(5X,F10.4),10X,I3)
0252       8 FORMAT(10X,"CONECTIVIDAD ENTRE NUDOS")
0253      11 FORMAT(12X,3(I3,10X))
0254         RETURN
0255         END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00251 COMMON = 00000

APENDICE G

```
PAGE 0001  FTN.   7:24 AM  SAT., 16  JAN., 1982


0001  FTN4
0002        PROGRAM CHECUC),CHEQUEO DE CURVAS
0003        REAL IL
0004        DIMENSION IL(363),XD(363),YD(363),XOD(363),YOD(363),IX(3),IY(4),
0005       *ITU(6),ITU2(7),IP(5),TITU(30),ELOR(363),ELO(363),ALOR(363),
0006       *BLO(363),ERNOR(363),AAA(363),BBB(363)
0007        DATA IX/2HAB,2HCI,2HSA/
0008        DATA IY/2HOR,2HDE,2HNA,2HDA/
0009        DATA ITU/2HCU,2HRV,2HA ,2HOB,2HJE,2HTO/
0010        DATA ITU2/2HCU,2HRV,2HA ,2HOB,2HTE,2HNI,2HDA/
0011        CALL RNPAR(IP)
0012        IN=IP(1)
0013        NO=IP(2)
0014        TH=IP(3)*.1
0015        LS=IP(4)
0016        LK=IP(5)
0017        READ(IN,1000)TITU
0018   1000 FORMAT(30A2)
0019        READ(IN,*)N,N1,N2,N3,N4,N5
0020        DO 10 I=1,N
0021        READ(IN,*)IL(I),XOD(I),YOD(I),XD(I),YD(I),ELOR(I),ELO(I),
0022       *AAA(I),BBB(I)
0023     10 CONTINUE
0024        DI=0.0
0025        FO=0.
0026        WRITE(NO,1)N,N1,N2,N3
0027        WRITE(NO,1000)TITU
0028        WRITE(NO,2)
0029        DO 20 I=N1,N2,N3
0030        D=SQRT((XD(I)-XOD(I))**2+(YD(I)-YOD(I))**2)
0031        DI=AMAX1(D,DI)
0032        EX=(XD(I)-XOD(I))/XD(I)*100
0033        EY=(YD(I)-YOD(I))/YD(I)*100
0034        FO=FO+D
0035        WRITE(NO,100)I,XD(I),XOD(I),YD(I),YOD(I),D,EX,EY
0036     20 CONTINUE
0037        WRITE(NO,150)FO,DI
0038    150 FORMAT(10X,"SUMA DE D.=",E14.7,3X,"MAX.D.=",E14.7)
0039        DO 3575 LL=N1,N2,N3
0040        ERO=(ELOR(LL)-ELO(LL))/ELOR(LL)*100
0041        ERNOR(LL)=(AAA(LL)-BBB(LL))/AAA(LL)*100
0042        WRITE(NO,2000)IL(LL),ELOR(LL),ELO(LL),ERO,ERNOR(LL)
0043   2000 FORMAT(2X,5(E11.5,3X))
0044   3575 CONTINUE
0045        IZ=I+2
0046        IF(LK-1)48,49,49
0047     49 CALL GRAF(XD,YD,IZ,IX,3,IY,4,ITU,6,9.,9.,TA,LS,LK)
0048        IF(LK-2)48,50,48
0049     50 CALL GRAF(XOD,YOD,IZ,IX,3,IY,4,ITU2,7,9.,9.,TA,N4,N5)
0050     48 PAUSE 001
0051        CALL GRAF(IL,ERNOR,IZ,IX,3,IY,4,ITU,6,9.,9.,TA,LS,LK)
0052        PAUSE 002
0053        CALL GRAF(IL,ELOR,IZ,IX,3,IY,4,ITU,6,9.,9.,TA,LS,LK)
0054        CALL GRAF(IL,ELO,IZ,IX,3,IY,4,ITU2,7,9.,9.,TA,LS,LK)
0055        PAUSE 003
```

```
0056          CALL GRAF(IL,BLOR,IZ,IX,3,IY,4,ITU,6,9.,9.,TA,LS,LK)
0057          CALL GRAF(IL,BLO,IZ,IX,3,IY,4,ITU2,7,9.,9.,TA,LS,LK)
0058          STOP
0059      100 FORMAT(5X,I3,7(3X,E14.7))
0060        1 FORMAT(1H1,10X," CURVA DEL ACOPLADOR DE UN CUATRO BARRAS ",2/,
0061         #15X,"N=",I3,2X,"ANGULO INICIAL=",I5,2X,"ANGULO FINAL=",I5,2X,
0062         #"INTERVALO ANGULAR=",I5,3/)
0063        2 FORMAT(5X,"CHEQUEO DE TRAYECTORIAS",2/,3X,"I",6X,"XD",8X,"XOD",
0064         #6X,"YD",8X,"YOD",7X,"D",7X,"EX",7X,"EY",2/)
0065          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 09202 COMMON = 00000

```
PAGE 0003  FTN.   7:24 AM  SAT., 16  JAN., 1982


0066          SUBROUTINE GRAF(ABCI,ORDE1,L,IX,LX,IY,LY,ITU,IT,PX,PY,TA,LS,LK)
0067   C    SUBRUTINA PARA DIBUJAR  UNA SOLA GRAFICA .EL PROGRAMA ESCALA LAS
0068   C    DIMENSIONES DEL DIBUJO DE ACUERDO CON LOS VALORES QUE LES SON TRANS
0069   C    MITIDOS
0070          DIMENSION ABCI(L),ORDE1(L),IX(LX),IY(LY),ITU(IT)
0071          CALL PLTLU(10)
0072          CALL FACT(TA)
0073          LTK=L-2
0074          LXX=2*LX
0075          LYY=2*LY
0076          ITT=2*IT
0077          CALL SCALE(ABCI(1),PX,LTK,1)
0078       10 CALL AXIS(0.,0.,IX,-LXX,PX,0.,ABCI(LTK+1),ABCI(LTK+2))
0079          CALL SCALE(ORDE1(1),PY,LTK,1)
0080          CALL AXIS(0.,0.,IY,LYY,PY,90.,ORDE1(LTK+1),ORDE1(LTK+2))
0081       20 CALL LINE(ABCI,ORDE1,LTK,1,LS,LK)
0082          CALL SYMB(0.,PY,0.1,ITU,0.,ITT)
0083          PYY=PY+2
0084          CALL PLOT(0.,PYY,-3)
0085          RETURN
0086          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **    PROGRAM = 00163      COMMON = 00000
```

APENDICE H

```
 PAGE 0001  FTN.   4:03 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         FUNCTION FUNC(Z,J)
0003   C
0004   C    FUNCION OBJETIVO Y SISTEMA DE ECUACIONES PARA LA SINTESIS DE
0005   C    GENERACION DE TRAYECTORIAS CON UN MECANISMO DE CUATRO BARRAS
0006   C    QUE PUEDEN SER DE LONGITUD FIJA O VARIABLE
0007   C
0008         COMMON/COMU/LECT,INPE,TITU(30),NCVE,ZI(30),NVE,ICOMM
0009         COMMON/DATOS/P(2,31),TIEMP(30),IB,IC,ID,IEL,ISL(80),ZP(80),
0010        *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0011         DIMENSION Z(30),ZII(60),IS(80)
0012         ISLL=0
0013         DO 1 LN=1,NVAL
0014         ISLL=ISLL+ISL(LN)
0015         IF(ISL(LN).EQ.0)ZII(LN)=ZP(LN)
0016         IF(ISL(LN).EQ.1)ZII(LN)=Z(ISLL)
0017       1 CONTINUE
0018   C
0019   C      CALCULO DE LAS LONGITUDES INICIALES A PARTIR DE LOS VALORES
0020   C      DEL VECTOR DE DISEÑO TRANSMITIDO
0021   C
0022         BLO=SQRT((ZII(7)-ZII(3))**2+(ZII(8)-ZII(4))**2)
0023         CLO=SQRT((ZII(7)-ZII(5))**2+(ZII(8)-ZII(6))**2)
0024         DLO=SQRT((ZII(3)-ZII(1))**2+(ZII(4)-ZII(2))**2)
0025         ELO=SQRT((P(1,1)-ZII(5))**2+(P(2,1)-ZII(6))**2)
0026   C
0027   C      EVALUACION DE LOS DATOS LEIDOS.SE DETERMINA QUE BARRAS SON
0028   C      DE LONGITUD VARIABLE Y CON QUE LEY.SE CALCULA LA NUEVA LONGITUD
0029   C
0030         I=J
0031         ISLL=IAN
0032         DO 2 ISP=1,NVAL
0033       2 IS(ISP)=ISP
0034         IF(IB.NE.1)GOTO 10
0035         DO 11 ISP=1,NVAL
0036      11 IS(ISP)=IS(ISP)+ISLL
0037         CALL LONG(TIEMP(I),BLO,IBB,IBN,ZII,IS)
0038         ISLL=IBN
0039      10 IF(IC.NE.1)GOTO 20
0040         DO 21 ISP=1,NVAL
0041      21 IS(ISP)=IS(ISP)+ISLL
0042         CALL LONG(TIEMP(I),CLO,ICC,ICN,ZII,IS)
0043         ISLL=ICN
0044      20 IF(ID.NE.1)GOTO 30
0045         DO 31 ISP=1,NVAL
0046      31 IS(ISP)=IS(ISP)+ISLL
0047         CALL LONG(TIEMP(I),DLO,IDD,IDN,ZII,IS)
0048         ISLL=IDN
0049      30 IF(IEL.NE.1)GOTO 40
0050         DO 41 ISP=1,NVAL
0051      41 IS(ISP)=IS(ISP)+ISLL
0052         CALL LONG(TIEMP(I),ELO,IEEL,IEEN,ZII,IS)
0053         ISLL=IEEN
0054      40 CONTINUE
0055   C
```

```
PAGE 0002  FUNC    4:03 AM  SAT., 16  JAN., 1982

0096   C       DETERMINACION DE LA I-ESIMA ECUACION DEL SISTEMA
0097   C
0098           A0X=ZII(1)
0099           A0Y=ZII(2)
0100           A1X=ZII(5)
0101           A1Y=ZII(6)
0102           SSX=(ZII(4)-A0Y)
0103           SSY=ZII(3)-A0X
0104           SSZ=(ZII(8)-ZII(6))
0105           SZS=ZII(7)-ZII(5)
0106           THETA=ATAN2(SSX,SSY)
0107           DELTA=ATAN2(SSZ,SZS)
0108           B0X=COS(THETA)*BLO+A0X
0109           B0Y=SIN(THETA)*BLO+A0Y
0110           B1X=COS(DELTA)*CLO+A1X
0111           B1Y=SIN(DELTA)*CLO+A1Y
0112           IL=I+1
0113           CON=ATAN(1.)/45.
0114           THE=TIEMP(I)*CON
0115           C=COS(THE)
0116           S=SIN(THE)
0117           AX=A1X-A0X
0118           AY=A1Y-A0Y
0119           AJX=C*AX-S*AY+A0X
0120           AJY=S*AX+C*AY+A0Y
0121           COC=(P(2,IL)-AJY)
0122           COI=P(1,IL)-AJX
0123           ANG=ATAN2(COC,COI)
0124           COC1=(P(2,1)-A1Y)
0125           COCO=P(1,1)-A1X
0126           AGNU=ATAN2(COC1,COCO)
0127           ALFA=ANG-AGNU
0128           CA=COS(ALFA)
0129           SA=SIN(ALFA)
0130           BX=B1X-A1X
0131           BY=B1Y-A1Y
0132           BJX=CA*BX-SA*BY+AJX
0133           BJY=SA*BX+CA*BY+AJY
0134           F1=((P(1,IL)-AJX)**2+(P(2,IL)-AJY)**2-(BLO**2))**2
0135           F2=((BJX-B0X)**2+(BJY-B0Y)**2-(BLO**2))**2
0136   C
0137   C       VALOR DE LA I-ESIMA ECUACION
0138   C
0139           FUNC=F1+F2
0140           RETURN
0141           END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

**   NO WARNINGS **   NO ERRORS **    PROGRAM = 01023       COMMON = 00000
```

```
PAGE 0003  FTN.   4103 AM  SAT., 16  JAN., 1982

0102          SUBROUTINE DATS(NEQ,LECT)
0103    C
0104    C     LECTURA DE DATOS ESPECIFICOS AL CASO PROPUESTO
0105    C
0106          COMMON/DATOS/DATOP(2,31),DATOA(30),IB,IC,ID,IEL,ISL(80),ZP(80),
0107         *IBB,ICC,IDD,IELL,NVAL,IAN,IBN,ICN,IDN,IEFN
0108          READ(LECT,*)NVAL,IAN,IBN,ICN,IDN,IEFN
0109          READ(LECT,*)IB,IBB,IC,ICC,ID,IDD,IEL,IELL
0110          DO 2 N=1,NVAL
0111          READ(LECT,*)ISL(N),ZP(N)
0112        2 CONTINUE
0113    C
0114    C     LECTURA DE LOS DATOS DE LA TRAYECTORIA
0115    C
0116          READ(LECT,*)ANGIN,DATOP(1,1),DATOP(2,1)
0117          DO 1  I=1,NEQ
0118          NN=I+1
0119          READ(LECT,*)DATOA(I),DATOP(1,NN),DATOP(2,NN)
0120          DATOA(I)=DATOA(I)-ANGIN
0121        1 CONTINUE
0122          RETURN
0123          END


FTN4 COMPILER> HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00165      COMMON = 00000
```

```
PAGE 0001  FTN.   4109 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         FUNCTION FUNC(Z,J)
0003   C
0004   C    FUNCION OBJETIVO DE DETERMINACION DE UNA  CADENA PLANA ABIERTA
0005   C    CON BARRAS DE LONGITUD VARIABLE CON CUALQUIER TIPO DE LEY
0006   C    DE VARIACION EN EL TIEMPO
0007   C
0008         COMMON/COMU/LECT,IMPR,TIEMP(30),NCVE,Z2(30),NVE,ICONM
0009         COMMON/DATOS/P(2,31),TIEMP(30),IB,IC,ID,IEL,ISL(80),ZP(80),
0010        *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0011         DIMENSION Z(30),ZII(80),IS(80)
0012         ISLL=0
0013         DO 1 LN=1,NVAL
0014         ISLL=ISLL+ISL(LN)
0015         IF(ISL(LN).EQ.0)ZII(LN)=ZP(LN)
0016         IF(ISL(LN).EQ.1)ZII(LN)=Z(ISLL)
0017       1 CONTINUE
0018         BLOI=SQRT((ZII(4)-ZII(2))**2+(ZII(3)-ZII(1))**2)
0019         CLOI=SQRT((ZII(6)-ZII(4))**2+(ZII(5)-ZII(3))**2)
0020         BLO=BLOI
0021         CLO=CLOI
0022         THEI=ZII(7)
0023         ALFAI=ZII(8)
0024         I=J
0025         ISLL=IAN
0026         DO 2 ISF=1,NVAL
0027       2 IS(ISF)=ISF
0028         IF(IB.NE.1)GOTO 10
0029         DO 11 ISF=1,NVAL
0030      11 IS(ISF)=IS(ISF)+ISLL
0031         CALL LONG(TIEMP(I),BLO,IBB,IBN,ZII,IS)
0032         ISLL=IBN
0033      10 IF(IC.NE.1)GOTO 20
0034         DO 21 ISF=1,NVAL
0035      21 IS(ISF)=IS(ISF)+ISLL
0036         CALL LONG(TIEMP(I),CLO,ICC,ICN,ZII,IS)
0037         ISLL=ICN
0038      20 IF(ID.NE.1)GOTO 30
0039         DO 31 ISF=1,NVAL
0040      31 IS(ISF)=IS(ISF)+ISLL
0041         CALL LONG(TIEMP(I),THEI,IDD,IDN,ZII,IS)
0042         ISLL=IDN
0043      30 IF(IEL.NE.1)GOTO 40
0044         DO 41 ISF=1,NVAL
0045      41 IS(ISF)=IS(ISF)+ISLL
0046         CALL LONG(TIEMP(I),ALFAI,IEEL,IEEN,ZII,IS)
0047         ISLL=IEEN
0048      40 CONTINUE
0049         A0X=ZII(1)
0050         A0Y=ZII(2)
0051         COALB=(ZII(3)-A0X)/BLOI
0052         COBEB=(ZII(4)-A0Y)/BLOI
0053         A1X=COALB*BLO+A0X
0054         A1Y=COBEB*BLO+A0Y
0055         COALC=(ZII(5)-ZII(3))/CLOI
```

```
PAGE 0002  FUNC    4:09 AM  SAT.,  16  JAN ,  1982

0056          COSEC=(ZII(2)-ZII(4))/CLOI
0057          BIX=COALD*CLO+AIX
0058          BIY=COSEC*CLO+AIY
0059          CON=ATAN(1.)/45.
0060          CICH=CON*360.0
0061          CONN=TIENP(I)*CON
0062          THETA=THEI*CONN
0063          THETAD=ABS(THETA)
0064          ALFA=ALFAI*CONN+THETA
0065          ALFAD=ABS(ALFA)
0066          IF(THETAD.GT.CION)THETA=AMOD(THETA,CICH)
0067          IF(ALFAD.GT.CION)ALFA=AMOD(ALFA,CION)
0068          ZII(7)=THETA/CONN
0069          ZII(8)=(ALFA-THETA)/CONN
0070          CT=COS(THETA)
0071          ST=SIN(THETA)
0072          CA=COS(ALFA)
0073          SA=SIN(ALFA)
0074          AX=A1X-A0X
0075          AY=A1Y-A0Y
0076          AJX=CT*AX-ST*AY+A0X
0077          AJY=ST*AX+CT*AY+A0Y
0078          BX=B1X-A1X
0079          BY=B1Y-A1Y
0080          BJX=CA*BX-SA*BY+AJX
0081          BJY=SA*BX+CA*BY+AJY
0082          F1=(P(1,I)-BJX)**2+(P(2,I)-BJY)**2
0083          F11=SQRT(F1)
0084          FUNC=F11
0085          RETURN
0086          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **    PROGRAM = 01054        COMMON = 00000
```

```
PAGE 0003  FTN.   4:09 PM  SAT., 16  JAN., 1982


0007          SUBROUTINE DATS(NEO,LECT)
0008          COMMON/DATOS/DATOP(2,31),DATOA(30),IB,IC,ID,IEL,ISL(80),ZP(80),
0009         *IBB,ICC,IDD,IELL,NVAL,IAN,IBN,ICN,IDN,IEEN
0090          READ(LECT,*)NVAL,IAN,IBN,ICN,IDN,IEEN
0091          READ(LECT,*)IB,IBB,IC,ICC,ID,IDD,IEL,IELL
0092          DO 2 N=1,NVAL
0093          READ(LECT,*)ISL(N),ZP(N)
0094        2 CONTINUE
0095          DO 1  I=1,NEO
0096          READ(LECT,*)DATOA(I),DATOP(1,I),DATOP(2,I)
0097        1   CONTINUE
0098          RETURN
0099          END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)


**   NO WARNINGS **  NO ERRORS **    PROGRAM = 00119       COMMON = 00000
```

```
PAGE 0001  FTN.   6:39 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         FUNCTION FUNC(Z,J)
0003   C
0004   C    FUNCION OBJETIVO DE DETERMINACION DE UNA  CADENA ESPACIAL ABIERTA
0005   C    CON BARRAS DE LONGITUD VARIABLE CON CUALQUIER TIPO DE LEY
0006   C    DE VARIACION EN EL TIEMPO
0007   C
0008         COMMON/COMU/LECT,IMPR,TITU(30),NCVE,ZZ(30),NVE,ICOMM
0009         COMMON/DATOS/P(3,31),TIEMP(30),IB,IC,ID IEL,ISL(80),ZP(80),
0010        *IBB,ICC,IDD IEEL,NVAL,IAN,IBN,ICN,IDN IEEN
0011         DIMENSION Z(30),ZII(80),IS(80),U0(3),U1(3),A0(3),AJ(3),EJ(3),
0012        *AI(3),BI(3) RE(3,3),DE(4,4),UU1(3),BU(3),BJI(3)
0013         ISLL=0
0014         DO 1 LN=1,NVAL
0015         ISLL=ISLL+ISL(LN)
0016         IF(ISL(LN).EQ.0)ZII(LN)=ZP(LN)
0017         IF(ISL(LN).EQ.1)ZII(LN)=Z(ISLL)
0018       1 CONTINUE
0019         BLOI=SQRT((ZII(1)-ZII(4))**2+(ZII(2)-ZII(5))**2+(ZII(3)-ZII(6))
0020        ***2)
0021         CLOI=SQRT((ZII(4)-ZII(7))**2+(ZII(5)-ZII(8))**2+(ZII(6)-ZII(9))
0022        ***2)
0023         BLO=BLOI
0024         CLO=CLOI
0025         DO 100 KK=10,13
0026         IF(ZII(KK).LE.1.0)GOTO 100
0027         ZII(KK)=ZII(KK)-IFIX(ZII(KK))
0028    100  CONTINUE
0029         U0X2=ZII(10)**2
0030         U0Y2=ZII(11)**2
0031         U1X2=ZII(12)**2
0032         U1Y2=ZII(13)**2
0033         U0S=U0X2+U0Y2
0034         U1S=U1X2+U1Y2
0035         IF(U0S.LE.1.)GOTO 200
0036         U0(1)=ZII(10)/(U0S**0.5)
0037         U0(2)=ZII(11)/(U0S**0.5)
0038         GOTO 300
0039    200  U0(1)=ZII(10)
0040         U0(2)=ZII(11)
0041    300  U0(3)=1-U0(1)**2-U0(2)**2
0042         IF(U1S.LE.1.)GOTO 400
0043         U1(1)=ZII(12)/(U1S**.5)
0044         U1(2)=ZII(13)/(U1S**.5)
0045         GOTO 500
0046    400  U1(1)=ZII(12)
0047         U1(2)=ZII(13)
0048    500  U1(3)=1-U1(1)**2-U1(2)**2
0049         I=J
0050         ISLL=IAN
0051         DO 2 ISP=1,NVAL
0052       2 IS(ISP)=ISP
0053         IF(IB.NE.1)GOTO 10
0054         DO 11 ISP=1,NVAL
0055      11 IS(ISP)=IS(ISP)+ISLL
```

```
0056          CALL LONG(TIEMP(I),BLO,IBB,IBN,ZII,IS)
0057          ISLL=IBN
0058       10 IF(IC.NE.1)GOTO 20
0059          DO 21 ISF=1,NVAL
0060       21 IS(ISF)=IS(ISF)+ISLL
0061          CALL LONG(TIEMP(I),CLO,ICC,ICN,ZII,IS)
0062          ISLL=ICN
0063       20 CONTINUE
0064          CDAA=(ZII(4)-ZII(1))/BLOI
0065          CDAB=(ZII(5)-ZII(2))/BLOI
0066          CDAC=(ZII(6)-ZII(3))/BLOI
0067          A0(1)=ZII(1)
0068          A0(2)=ZII(2)
0069          A0(3)=ZII(3)
0070          A1(1)=CDAA*BLO+ZII(1)
0071          A1(2)=CDAB*BLO+ZII(2)
0072          A1(3)=CDAC*BLO+ZII(3)
0073          CDBA=(ZII(7)-ZII(4))/CLOI
0074          CDBB=(ZII(8)-ZII(5))/CLOI
0075          CDBC=(ZII(9)-ZII(6))/CLOI
0076          B1(1)=CDBA*CLO+A1(1)
0077          B1(2)=CDBB*CLO+A1(2)
0078          B1(3)=CDBC*CLO+A1(3)
0079          CON=ATAN(1.)/45.
0080          CONN=TIEMP(I)*CON
0081          THETA=CONN*ZII(14)
0082          ALFA=CONN*ZII(15)
0083          CALL REJE(U,THETA,RE,DE,A1,A0)
0084          DO 600 LLK=1,3
0085          UU1(LLK)=0.0
0086          DO 599 LKK=1,3
0087      599 UU1(LLK)=RE(LLK,LKK)*U1(LKK)+UU1(LLK)
0088      600 U1(LLK)=UU1(LLK)
0089          DO 601 LLS=1,3
0090      601 AJ(LLS)=DE(LLS,1)*A1(1)+DE(LLS,2)*A1(2)+DE(LLS,3)*A1(3)+DE(LLS,4)
0091          CALL REJE(U,THETA,RE,DE,AJ,A1)
0092          DO 602 LIT=1,3
0093      602 BJI(LIT)=DE(LIT,1)*B1(1)+DE(LIT,2)*B1(2)+DE(LIT,3)*B1(3)+DE(LIT,4
0094          CALL REJE(U,ALFA,RE)
0095          DO 604 ILK=1,3
0096          BU(ILK)=0.0
0097          DO 605 ILF=1,3
0098      605 BU(ILK)=RE(ILK,ILF)*BJI(ILF)+BU(ILK)
0099      604 BJ(ILK)=BU(ILK)
0100          F1=(P(1,I)-BJ(1))**2+(P(2,I)-BJ(2))**2+(P(3,I)-BJ(3))**2
0101          F11=SQRT(F1)
0102          FUNC=F11
0103          RETURN
0104          END
```

```
PAGE 0004  FTN.   6:39 AM  SAT., 16  JAN., 1982


0105          SUBROUTINE REJE(U,PHI,RE)
0106          DIMENSION U(3),RE(3,3)
0107          C=COS(PHI)
0108          S=SIN(PHI)
0109          V=1.-C
0110          RE(1,1)=U(1)*U(1)*V+C
0111          RE(1,2)=U(1)*U(2)*V-U(3)*S
0112          RE(1,3)=U(1)*U(3)*V+U(2)*S
0113          RE(2,1)=U(1)*U(2)*V+U(3)*S
0114          RE(2,2)=U(2)*U(2)*V+C
0115          RE(2,3)=U(2)*U(3)*V-U(1)*S
0116          RE(3,1)=U(1)*U(3)*V-U(2)*S
0117          RE(3,2)=U(2)*U(3)*V+U(1)*S
0118          RE(3,3)=U(3)*U(3)*V+C
0119          RETURN
0120          END


 FTN4 COMPILER> HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **    PROGRAM = 00352       COMMON = 00000
```

```
 PAGE 0005  FTN.   6:29 AM  SAT., 16  JAN., 1982


0121          SUBROUTINE REJE(U,PHI,RE,DE,PJ,P1)
0122          DIMENSION U(3) RE(3,3),DE(4,4),PJ(3),P1(3)
0123          CALL REJE(U,PHI,RE)
0124          DO 10 I=1,3
0125          DO 10 K=1,3
0126       10 DE(I,K)=RE(I,K)
0127          DE(4,4)=1.0
0128          DO 20 I=1,3
0129          DE(4,I)=0.0
0130       20 DE(I,4)=PJ(I)-DE(I,1)*P1(1)-DE(I,2)*P1(2)-DE(I,3)*P1(3)
0131          RETURN
0132          END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00193       COMMON = 00000
```

```
PAGE 0V06  FTN.   6:39 AM  SAT., 16  JAN., 1982


0133          SUBROUTINE DATS(NEO,LECT)
0134          COMMON/DATOI/DATOP(3,31),DATOA(30),ID,IC,ID,IEL,ISL(80),ZP(80),
0135         *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0136          READ(LECT,*)NVAL,IAN,IBN,ICN,IDN,IEEN
0137          READ(LECT,*)IB,IBB,IC,ICC,ID,IDD,IEL,IEEL
0138          DO 2 N=1,NVAL
0139          READ(LECT,*)ISL(N),ZP(N)
0140        2 CONTINUE
0141          DO 1  I=1,NEO
0142          READ(LECT,*)DATOA(I),DATOP(1,I),DATOP(2,I),DATOP(3,I)
0143        1  CONTINUE
0144          RETURN
0145          END


FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

**  NO WARNINGS **  NO ERRORS **   PROGRAM = 00136       COMMON = 00000
```

```
0124          SUBROUTINE LONG(TI,DIO,IAA,INN,ZII,IS)
0125    C
0126    C     SUBRUTINA QUE REALIZA LA VARIACION DE LA LONGITUD
0127    C     DE LAS BARRAS SEGUN LA LEY QUE SE LE INDIQUE
0128    C
0129          COMMON/COMU/LEC1,IMPR,TITU(30),NCVE,ZZ(30),NVE,ICOMM
0130          DIMENSION P(2,20),IS(1),ZII(1)
0131          GOTO(1,2,3,3),IAA
0132    C
0133    C     LEY SIMETRICA TIPO A
0134    C
0135        1 ZII(IS(1))=ABS(ZII(IS(1)))
0136          IF(TI.LE.180.0)NS=1
0137          IF(TI.GT.180.0)NS=2
0138          GOTO(10,11),NS
0139       10 DIO=DIO+ZII(IS(1))*TI
0140          GOTO 15
0141       11 DIO=DIO+ZII(IS(1))*180.0-ZII(IS(1))*(TI-180.0)
0142       15 GOTO 125
0143    C
0144    C     LEY TIPO B Y B-2
0145    C
0146        2 ZII(IS(1))=ABS(ZII(IS(1)))
0147          ZII(IS(2))=ABS(ZII(IS(2)))
0148          ZII(IS(3))=ABS(ZII(IS(3)))
0149          IF(TI.LE.ZII(IS(1)))NS=1
0150          IF(TI.GT.ZII(IS(1)))NS=2
0151          GOTO(20,21),NS
0152       20 DIO=DIO+ZII(IS(2))*TI
0153          GOTO 25
0154       21 DIO=DIO+ZII(IS(2))*ZII(IS(1))-ZII(IS(3))*(TI-ZII(IS(1)))
0155       25 GOTO 125
0156    C
0157    C     LEYES TIPO C Y TIPO D
0158    C
0159        3 ITLJ=0
0160          YD=0.0
0161          INN1=INN/2
0162          DO 67 ILJK=1,INNT
0163          DO 67 ILLK=1,2
0164          P(ILLK,ILJK)=0.0
0165          ITLJ=ITLJ+1
0166          IF(ILLK.EQ.1.AND.ILJK.NE.1)GOTO 60
0167          P(ILLK,ILJK)=ZII(IS(ITLJ))
0168          GOTO 67
0169       60 ITTE=ILJK-1
0170          P(1,ILJK)=P(1,ITTE)+ABS(ZII(IS(ITLJ)))
0171       67 CONTINUE
0172          DIOO=DIO
0173          NS=0
0174          IF(IAA.EQ.4)GOTO 37
0175          IF(TI.LE.P(1,1))NS=1
0176          ILA=INNT-1
0177          DO 53 LL=1,ILA
0178          LA=LL+1
```

```
0179          IF(TI.GT.P(1,LL).AND.TI.LE.P(1,LA))NS=L
0180       53 CONTINUE
0181          IF(TI.GT.P(1,LA))NS=LA+1
0182          IF(NS.NE.1)GOTO 32
0183          DIO=DIO+T1*P(2,1)
0184          GOTO 126
0185       32 NSS=NS-1
0186          VANT=0.0
0187          DO 54 LAL=1 NSS
0188          LALL=LAL-1
0189          IF(LAL.NE.1)VANT=P(1,LALL)
0190          DIO=DIO+(P(1,LAL)-VANT)*P(2,LAL)
0191       54 CONTINUE
0192          IF(NS.EQ.LA+1)GOTO 127
0193          DIO=DIO+(TI-P(1,NSS))*P(2,NS)
0194          GOTO 126
0195      127 DIO=DIO+P(1,NS)*P(2,NS)
0196      126 IF(DIO.LE.0)DIO=0.0
0197          RETURN
0198       37 IF(TI.GT.P(1,INNT).OR.TI.LT.P(1,1))GOTO 124
0199          CALL SPLIN(INNT,P,TI,YD)
0200      124 DIO=DIO+YD
0201          IF(DIO.LE.0)DIO=0.0
0202      125 CONTINUE
0203          RETURN
0204          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00613 COMMON = 00000

```
0205          SUBROUTINE SPLIN(N,P,PD,YD)
0206    C
0207    C     INTERPOLACION CON SPLINE CUBICOS
0208    C
0209          REAL M(20,4),L(19)
0210          DIMENSION P(2,20),B(20),F(4),U(20)
0211          NN=N-1
0212          DO 1575 LK=1,NN
0213          LKK=LK+1
0214          L(LK)=P(1,LKK)-P(1,LK)
0215     1575 CONTINUE
0216          M(1,2)=1.
0217          M(1,3)=0.5
0218          M(N,1)=2.
0219          M(N,2)=4.
0220          B(1)=(3/(2*L(1)))*(P(2,2)-P(2,1))
0221          B(N)=(6/L(NN))*(P(2,N)-P(2,NN))
0222          DO 1040 J=2,NN
0223          M(J,1)=L(J)
0224          M(J,2)=2*(L(J)+L(J-1))
0225          M(J,3)=L(J-1)
0226          B(J)=3*(L(J-1)**2*(P(2,J+1)-P(2,J))+L(J)**2*(P(2,J)-P(2,J-1)))
0227          B(J)=B(J)/(L(J)*L(J-1))
0228    1040  CONTINUE
0229          DO 1180 II=2,N
0230          IF(M(II,1).EQ.0.)GOTO 1180
0231          D=M(II-1,2)/M(II,1)
0232          DO 1120 K=1,3
0233     1120 M(II,K)=M(II,K)*D-M(II-1,K+1)
0234          B(II)=B(II)*D-B(II-1)
0235          D=M(II,2)
0236          DO 1181 K=1,3
0237     1181 M(II,K)=M(II,K)/D
0238     1180 B(II)=B(II)/D
0239          DO 1220 J=0,NN
0240          NA=N-J
0241          U(NA)=(B(NA)-M(NA,3)*U(NA+1))/M(NA,2)
0242    1220  CONTINUE
0243          AT=0.0
0244          DO 1500 J=1,NN
0245          LIM=J-1
0246          AT=AT+L(LIM)
0247          ATT=AT+L(J)
0248          IF(PD.GT.ATT.OR.PD.LT.AT)GOTO 1500
0249          F(1)=P(2,J)
0250          F(2)=U(J)
0251          F(3)=3/L(J)**2)*(P(2,J+1)-P(2,J))-(1/L(J))*(U(J+1)+2*U(J))
0252          F(4)=((-2)/L(J)**3)*(P(2,J+1)-P(2,J))+(1/L(J)**2)*(U(J+1)+
0253         +U(J))
0254          T=PD-AT
0255          YD=F(1)+F(2)*T+F(3)*(T**2)+F(4)*(T**3)
0256     1500 CONTINUE
0257          RETURN
0258          END
```

```
PAGE 0001  FTN.   6:34 AM  SAT., 16  JAN., 1982

0001   FTN4
0002         FUNCTION FUNC(Z,J)
0003   C
0004   C    FUNCION OBJETIVO DEL MECANISMO DE 4 BARRAS CON LONGITUD FIJA
0005   C    O VARIABLE CON UNA LEY CUALQUIERA
0006   C
0007         COMMON/CONUN/LECT,IMPR,TITU(30),ICOMN,NOVE,NVE,NREST,NP,NTP
0008         COMMON/DATOS/P(2,361),TIEMP(360),IB,IC,ID,IEL,ISL(80),ZP(80),
0009        *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0010         DIMENSION Z(50),ZII(30),IS(80)
0011         ISLL=0
0012         DO 1 LN=1,NVAL
0013         ISLL=ISLL+ISL(LN)
0014         IF(ISL(LN).EQ.0)ZII(LN)=ZP(LN)
0015         IF(ISL(LN).EQ.1)ZII(LN)=Z(ISLL)
0016       1 CONTINUE
0017         BLO=SQRT((ZII(7)-ZII(3))**2+(ZII(8)-ZII(4))**2)
0018         CLO=SQRT((ZII(7)-ZII(5))**2+(ZII(8)-ZII(6))**2)
0019         DLO=SQRT((ZII(3)-ZII(1))**2+(ZII(4)-ZII(2))**2)
0020         ELO=SQRT((P(1,1)-ZII(5))**2+(P(2,1)-ZII(6))**2)
0021         I=J
0022         ISLL=IAN
0023         DO 2 ISF=1,NVAL
0024       2 IS(ISF)=ISF
0025         IF(IB.NE.1)GOTO 10
0026         DO 11 ISF=1,NVAL
0027      11 IS(ISF)=IS(ISF)+ISLL
0028         CALL LONG(TIEMP(I),BLO,IBB,IBN,ZII,IS)
0029         ISLL=IBN
0030      10 IF(IC.NE.1)GOTO 20
0031         DO 21 ISF=1,NVAL
0032      21 IS(ISF)=IS(ISF)+ISLL
0033         CALL LONG(TIEMP(I),CLO,ICC,ICN,ZII,IS)
0034         ISLL=ICN
0035      20 IF(ID.NE.1)GOTO 30
0036         DO 31 ISF=1,NVAL
0037      31 IS(ISF)=IS(ISF)+ISLL
0038         CALL LONG(TIEMP(I),DLO,IDD,IDN,ZII,IS)
0039         ISLL=IDN
0040      30 IF(IEL.NE.1)GOTO 40
0041         DO 41 ISF=1,NVAL
0042      41 IS(ISF)=IS(ISF)+ISLL
0043         CALL LONG(TIEMP(I),ELO,IEEL,IEEN,ZII,IS)
0044         ISLL=IEEN
0045      40 CONTINUE
0046         A0X=ZII(1)
0047         A0Y=ZII(2)
0048         AIX=ZII(5)
0049         AIY=ZII(6)
0050         SSX=(ZII(4)-A0Y)
0051         SSY=ZII(3)-A0X
0052         SSZ=(ZII(8)-ZII(6))
0053         SZS=ZII(7)-ZII(5)
0054         THETA=ATAN2(SSX,SSY)
0055         DELTA=ATAN2(SSZ,SZS)
```

```
PAGE 0002  FUNC   6134 AK  SAT., 16  JAN., 1982


0036          B0X=COS(THETA)*DLO+A0X
0037          B0Y=SIN(THETA)*DLO+A0Y
0038          B1X=COS(DELTA)*CLO+A1X
0039          B1Y=SIN(DELTA)*CLO+A1Y
0040          TEST=ABS((B0X-A0X)/A0X)*100
0041          TEST1=ABS((B0Y-A0Y)/A0Y)*100
0042          IF(TEST.GT.50)GOTO 200
0043          IF(TEST1.GT.50)GOTO 200
0044          FUNC=1.E+15
0045          GOTO 250
0046      200 CONTINUE
0047          IL=I+1
0048          CON=ATAN(1.)/45.
0049          THE=TEMP(I)*CON
0050          C=COS(THE)
0051          S=SIN(THE)
0052          AX=A1X-A0X
0053          AY=A1Y-A0Y
0054          AJX=C*AX-S*AY+A0X
0055          AJY=S*AX+C*AY+A0Y
0056          COC=(P(2,IL)-AJY)
0057          COI=P(1,IL)-AJX
0058          ANG=ATAN2(COC,COI)
0059          COC1=(P(2,1)-A1Y)
0060          COCO=P(1,1)-A1X
0061          AGNU=ATAN2(COC1,COCO)
0062          ALFA=ANG-AGNU
0063          CA=COS(ALFA)
0064          SA=SIN(ALFA)
0065          BX=B1X-H1X
0066          BY=B1Y-H1Y
0067          BJX=CA*BX-SA*BY+AJX
0068          BJY=SA*BX+CA*BY+AJY
0069          F1=((P(1,IL)-AJX)**2+(P(2,IL)-AJY)**2-(ELO**2))**2
0070          F2=((BJX-B0X)**2+(BJY-B0Y)**2-(BLO**2))**2
0071          FUNC=F1+F2
0072      250 RETURN
0073          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 01273      COMMON = 00000
```

PAGE 0003 FTN. 6134 AM SAT., 16 JAN., 1982

```
0004          SUBROUTINE DATS(NEG,LECT)
0005          COMMON/DATOS/DATOP(2,361),DATOA(360),IB,IC,ID,IEL,ISL(80),ZP(80),
0006         *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0007          READ(LECT,*)NVAL,IAN,IBN,ICN,IDN,IEEN
0008          READ(LECT,*)IB,IBB,IC,ICC,ID,IDD,IEL,IEEL
0009          DO 2 N=1,NVAL
0010          READ(LECT,*)ISL(N),ZP(N)
0011        2 CONTINUE
0012          READ(LECT,*)ANGIN,DATOP(1,1),DATOP(2,1)
0013          DO 1 I=1,NEG
0014          NN=I+1
0015          READ(LECT,*)DATOA(I),DATOP(1,NN),DATOP(2,NN)
0016          DATOA(I)=DATOA(I)-ANGIN
0017        1 CONTINUE
0018          RETURN
0019          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00165 COMMON = 00080

```
PAGE 0001  FTN.   7:14 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         FUNCTION FUNC(Z,J)
0003   C
0004   C    FUNCION OBJETIVO DE DETERMINACION DE UNA  CADENA PLANA ABIERTA
0005   C    CON BARRAS DE LONGITUD VARIABLE CON CUALQUIER TIPO DE LEY
0006   C    DE VARIACION EN EL TIEMPO
0007   C
0008         COMMON/COMUN/LECT,IMPR,TITU(30),ICOMM,NOVE,NVE,NRES,NP,NTP
0009         COMMON/DATOS/P(2,361),TIEMP(360),IB,IC,ID,IEL,ISL(80),ZP(80),
0010        *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0011         DIMENSION Z(50),ZII(80),IS(80)
0012         ISLL=0
0013         DO 1 LN=1,NVAL
0014         ISLL=ISLL+ISL(LN)
0015         IF(ISL(LN).EQ.0)ZII(LN)=ZP(LN)
0016         IF(ISL(LN).EQ.1)ZII(LN)=Z(ISLL)
0017       1 CONTINUE
0018         BLOI=SQRT((ZII(4)-ZII(2))**2+(ZII(3)-ZII(1))**2)
0019         CLOI=SQRT((ZII(6)-ZII(4))**2+(ZII(5)-ZII(3))**2)
0020         BLO=BLOI
0021         CLO=CLOI
0022         THEI=ZII(7)
0023         ALFAI=ZII(8)
0024         I=J
0025         ISLL=IAN
0026         DO 2 ISP=1,NVAL
0027       2 IS(ISP)=ISP
0028         IF(IB.NE.1)GOTO 10
0029         DO 11 ISP=1,NVAL
0030      11 IS(ISP)=IS(ISP)+ISLL
0031         CALL LONG(TIEMP(I),BLO,IBB,IBN,ZII,IS)
0032         ISLL=IBN
0033      10 IF(IC.NE.1)GOTO 20
0034         DO 21 ISP=1,NVAL
0035      21 IS(ISP)=IS(ISP)+ISLL
0036         CALL LONG(TIEMP(I),CLO,ICC,ICN,ZII,IS)
0037         ISLL=ICN
0038      20 IF(ID.NE.1)GOTO 30
0039         DO 31 ISP=1,NVAL
0040      31 IS(ISP)=IS(ISP)+ISLL
0041         CALL LONG(TIEMP(I),THEI,IDD,IDN,ZII,IS)
0042         ISLL=IDN
0043      30 IF(IEL.NE.1)GOTO 40
0044         DO 41 ISP=1,NVAL
0045      41 IS(ISP)=IS(ISP)+ISLL
0046         CALL LONG(TIEMP(I),ALFAI,IEEL,IEEN,ZII,IS)
0047         ISLL=IEEN
0048      40 CONTINUE
0049         AOX=ZII(1)
0050         AOY=ZII(2)
0051         COALB=(ZII(3)-AOX)/BLOI
0052         COBEB=(ZII(4)-AOY)/BLOI
0053         A1X=COALB*BLO+AOX
0054         A1Y=COBEB*BLO+AOY
0055         COALC=(ZII(5)-ZII(3))/CLOI
```

```
PAGE 0002  FUNC    7:14 AM  SAT., 16  JAN., 1982


0056          COSEC=(ZII(6)-ZII(4))/CLOI
0057          B1X=COALC*CLO+A1X
0058          B1Y=COBEC*CLO+A1Y
0059          CON=ATAN(1.)/45.
0060          CION=CON*360.0
0061          CONN=TENP(I)*CON
0062          THETA=THEI*CONN
0063          THETAB=ABS(THETA)
0064          ALFA=ALFAI*CONN+THETA
0065          ALFAB=ABS(ALFA)
0066          IF(THETAB.GT.CION)THETA=AMOD(THETA,CION)
0067          IF(ALFAB.GT.CION)ALFA=AMOD(ALFA,CION)
0068          ZII(7)=THETA/CONN
0069          ZII(8)=(ALFA-THETA)/CONN
0070          CT=COS(THETA)
0071          ST=SIN(THETA)
0072          CA=COS(ALFA)
0073          SA=SIN(ALFA)
0074          AX=A1X-A0X
0075          AY=A1Y-A0Y
0076          AJX=CT*AX-ST*AY+A0X
0077          AJY=ST*AX+CT*AY+A0Y
0078          BX=B1X-A1X
0079          BY=B1Y-A1Y
0080          BJX=CA*BX-SA*BY+AJX
0081          BJY=SA*BX+CA*BY+AJY
0082          F1=(P(1,I)-BJX)**2+(P(2,I)-BJY)**2
0083          F11=SQRT(F1)
0084          FUNC=F11
0085          RETURN
0086          END


 FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM = 01051       COMMON = 00000
```

```
 PAGE 0003  FTN.   7:14 AM  SAT., 16  JAN., 1982


0087          SUBROUTINE DATS(NEQ,LECT)
0088          COMMON/DATOS/DATOP(2,361),DATOA(360),IB,IC,ID,IEL,ISL(80),ZP(80),
0089         *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0090          READ(LECT,*)NVAL,IAN,IBN,ICN,IDN,IEEN
0091          READ(LECT,*)IB,IBB,IC,ICC,ID,IDD,IEL,IEEL
0092          DO 2 N=1,NVAL
0093          READ(LECT,*)ISL(N),ZP(N)
0094        2 CONTINUE
0095          DO 1 I=1,NEQ
0096          READ(LECT,*)DATOA(I),DATOP(1,I),DATOP(2,I)
0097        1 CONTINUE
0098          RETURN
0099          END


  FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

  **  NO WARNINGS **  NO ERRORS **   PROGRAM = 00119       COMMON = 00000
```

```
PAGE 0001  FTN.   7:17 AM  SAT., 16  JAN., 1982


0001   FTN4
0002         FUNCTION FUNC(Z,J)
0003   C
0004   C    FUNCION OBJETIVO DE DETERMINACION DE UNA  CADENA ESPACIAL ABIERTA
0005   C    CON BARRAS DE LONGITUD VARIABLE CON CUALQUIER TIPO DE LEY
0006   C    DE VARIACION EN EL TIEMPO
0007   C
0008         COMMON/COMUN/LECT,INPR,TITU(30),ICONN,NOVE,NVE,NREST,NP,NTP
0009         COMMON/DATOS/P(3,361),TIEMP(360),ID,IC,JD,IEL,ISL(80),ZP(80),
0010        *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IREN
0011         DIMENSION Z(80),ZII(80),IS(80),U0(3),U1(3),A0(3),AJ(3),BJ(3),
0012        *A1(3),B1(3),RE(3,3),DE(4,4),UU1(3),BU(3),BJI(3)
0013         ISLL=0
0014         DO 1 LN=1,NVAL
0015         ISLL=ISLL+ISL(LN)
0016         IF(ISL(LN).EQ.0)ZII(LN)=ZP(LN)
0017         IF(ISL(LN).EQ.1)ZII(LN)=Z(ISLL)
0018       1 CONTINUE
0019         BLOI=SQRT((ZII(1)-ZII(4))**2+(ZII(2)-ZII(5))**2+(ZII(3)-ZII(6))
0020        ***2)
0021         CLOI=SQRT((ZII(4)-ZII(7))**2+(ZII(5)-ZII(8))**2+(ZII(6)-ZII(9))
0022        ***2)
0023         BLO=BLOI
0024         CLO=CLOI
0025         DO 100 KK=10,13
0026         IF(ZII(KK).LE.1.0)GOTO 100
0027         ZII(KK)=ZII(KK)-IFIX(ZII(KK))
0028     100 CONTINUE
0029         U0X2=ZII(10)**2
0030         U0Y2=ZII(11)**2
0031         U1X2=ZII(12)**2
0032         U1Y2=ZII(13)**2
0033         U0S=U0X2+U0Y2
0034         U1S=U1X2+U1Y2
0035         IF(U0S.LE.1.)GOTO 200
0036         U0(1)=ZII(10)/(U0S**0.5)
0037         U0(2)=ZII(11)/(U0S**0.5)
0038         GOTO 300
0039     200 U0(1)=ZII(10)
0040         U0(2)=ZII(11)
0041     300 U0(3)=1-U0(1)**2-U0(2)**2
0042         IF(U1S.LE.1.)GOTO 400
0043         U1(1)=ZII(12)/(U1S**.5)
0044         U1(2)=ZII(13)/(U1S**.5)
0045         GOTO 500
0046     400 U1(1)=ZII(12)
0047         U1(2)=ZII(13)
0048     500 U1(3)=1-U1(1)**2-U1(2)**2
0049         I=J
0050         ISLL=IAN
0051         DO 2 ISP=1,NVAL
0052       2 IS(ISP)=ISP
0053         IF(IB.NE.1)GOTO 10
0054         DO 11 ISP=1,NVAL
0055      11 IS(ISP)=IS(ISP)+ISLL
```

```
0056          CALL LONG(TIEMP(I),DLO,IBB,IBN,ZII,IS)
0057          ISLL=IBN
0058       10 IF(IC.NE.1)GOTO 20
0059          DO 21 ISP=1,NVAL
0060       21 IS(ISP)=IS(ISP)+ISLL
0061          CALL LONG(TIEMP(I),CLO,ICC,ICN,ZII,IS)
0062          ISLL=ICN
0063       20 CONTINUE
0064          CDAA=(ZII(4)-ZII(1))/DLOI
0065          CDAB=(ZII(5)-ZII(2))/DLOI
0066          CDAC=(ZII(6)-ZII(3))/DLOI
0067          A0(1)=ZII(1)
0068          A0(2)=ZII(2)
0069          A0(3)=ZII(3)
0070          A1(1)=CDAA*DLO+ZII(1)
0071          A1(2)=CDAB*DLO+ZII(2)
0072          A1(3)=CDAC*DLO+ZII(3)
0073          CDBA=(ZII(7)-ZII(4))/CLOI
0074          CDBB=(ZII(8)-ZII(5))/CLOI
0075          CDBC=(ZII(9)-ZII(6))/CLOI
0076          B1(1)=CDBA*CLO+A1(1)
0077          B1(2)=CDBB*CLO+A1(2)
0078          B1(3)=CDBC*CLO+A1(3)
0079          CON=ATAN(1.)/45.
0080          CONN=TIEMP(I)*CON
0081          THETA=CONN*ZII(14)
0082          ALFA=CONN*ZII(15)
0083          CALL DEJE(U0,THETA,RE,DE,A1,A0)
0084          DO 600 LLK=1,3
0085          UU1(LLK)=0.0
0086          DO 500 LKK=1,3
0087       500 UU1(LLK)=RE(LLK,LKK)*U1(LKK)+UU1(LLK)
0088       600 U1(LLL)=UU1(LLK)
0089          DO 601 LLS=1,3
0090       601 AJ(LLS)=DE(LLS,1)*A1(1)+DE(LLS,2)*A1(2)+DE(LLS,3)*A1(3)+DE(LLS,4)
0091          CALL DEJE(U0,THETA,RE,DE,AJ,A1)
0092          DO 602 LIT=1,3
0093       602 BJI(LIT)=DE(LIT,1)*B1(1)+DE(LIT,2)*B1(2)+DE(LIT,3)*B1(3)+DE(LIT,4
0094          CALL REJE(U1,ALFA,RE)
0095          DO 604 ILK=1,3
0096          BU(ILK)=0.0
0097          DO 605 ILF=1,3
0098       605 BU(ILK)=RE(ILK,ILF)*BJI(ILF)+BU(ILK)
0099       604 BJ(ILK)=BU(ILK)
0100          F1=(P(1,I)-BJ(1))**2+(P(2,I)-BJ(2))**2+(P(3,I)-BJ(3))**2
0101          F11=SQRT(F1)
0102          FUNC=F11
0103          RETURN
0104          END
```

PAGE 0004 FTN. 7:17 AM SAT., 16 JAN., 1982

```
0105          SUBROUTINE REJE(U,PHI,RE)
0106          DIMENSION U(3),RE(3,3)
0107          C=COS(PHI)
0108          S=SIN(PHI)
0109          V=1.-C
0110          RE(1,1)=U(1)*U(1)*V+C
0111          RE(1,2)=U(1)*U(2)*V-U(3)*S
0112          RE(1,3)=U(1)*U(3)*V+U(2)*S
0113          RE(2,1)=U(1)*U(2)*V+U(3)*S
0114          RE(2,2)=U(2)*U(2)*V+C
0115          RE(2,3)=U(2)*U(3)*V-U(1)*S
0116          RE(3,1)=U(1)*U(3)*V-U(2)*S
0117          RE(3,2)=U(2)*U(3)*V+U(1)*S
0118          RE(3,3)=U(3)*U(3)*V+C
0119          RETURN
0120          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00353 COMMON = 00000

PAGE 0003 FTN. 7:17 AM SAT., 16 JAN., 1982

```
0131          SUBROUTINE DEJE(U,PHI,RE,DE,PJ,P1)
0132          DIMENSION U(3),RE(3,3),DE(4,4),PJ(3),P1(3)
0133          CALL AEJE(U,PHI,RE)
0134          DO 10 I=1,3
0135          DO 10 K=1,3
0136       10 DE(I,K)=RE(I,K)
0137          DE(4,4)=1.0
0138          DO 20 I=1,3
0139          DE(4,I)=0.0
0140       20 DE(I,4)=PJ(I)-DE(I,1)*P1(1)-DE(I,2)*P1(2)-DE(I,3)*P1(3)
0141          RETURN
0142          END
```

FTN4 (COMPILER) HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00193 COMMON = 00000

PAGE 0006 FTN. 7:17 AM SAT., 16 JAN., 1982

```
0133          SUBROUTINE DATS(NEB,LECT)
0134          COMMON/DATOS/DATOP(2,361),DATOA(360),IB,IC,ID,IEL,ISL(80),ZP(80)
0135         *IBB,ICC,IDD,IEEL,NVAL,IAN,IBN,ICN,IDN,IEEN
0136          READ(LECT,*)NVAL,IAN,IBN,ICN,IDN,IEEN
0137          READ(LECT,*)IB,IBB,IC,ICC,ID,IDD,IEL,IEEL
0138          DO 2 N=1,NVAL
0139          READ(LECT,*)ISL(N),ZP(N)
0140        2 CONTINUE
0141          DO 1 I=1,NEB
0142          READ(LECT,*)DATOA(I),DATOP(1,I),DATOP(2,I),DATOP(2,1)
0143        1 CONTINUE
0144          RETURN
0145          END
```

FTN4 COMPILER: HP92060-16092 REV. 1926 (790430)

** NO WARNINGS ** NO ERRORS ** PROGRAM = 00130 COMMON = 00000