TESIS DOCTORAL

APLICACION DE LAS TECNOLOGIAS CAD AL DISEÑO OPTIMO DE ESTRUCTURAS MEDIANTE PROGRAMACION MATEMATICA NO LINEAL

presentada en la

UNIVERSIDAD POLITECNICA DE VALENCIA

Por

PEDRO PABLO COMPANY CALLEJA

para la obtención del

GRADO DE DOCTOR INGENIERO INDUSTRIAL

Director de tesis

Dr. PASCUAL MARTI MONTRULL

Valencia, Enero 1989.

TESIS DOCTORAL

APLICACION DE LAS TECNOLOGIAS CAD AL DISEÑO OPTIMO DE ESTRUCTURAS MEDIANTE PROGRAMACION MATEMATICA NO LINEAL

Por PEDRO PABLO COMPANY CALLEJA

Director Dr. PASCUAL MARTI MONTRULL

TRIBUNAL CALIFICADOR

Presidente: Dr.

Vocales:

Dr.

Dr.

Dr.

Dr.

A Melia y Jose Antonio.

AGRADECIMIENTOS.

En primer lugar, deseo expresar mi gratitud al profesor Pascual Martí, cuya dirección y apoyo constantes han hecho posible este trabajo. El profesor Martí me introdujo en el campo del diseño óptimo de estructuras, y con rigor y paciencia ha orientado y dirigido mi trabajo de investigación en este campo. Considero un privilegio el haber podido trabajar bajo su tutela durante estos últimos años.

Este trabajo se apoya en un software en cuyo desarrollo han intervenido diferentes personas, cuyos nombres aparecen en las referencias, pero quiero destacar los buenos momentos compartidos con Mario Sanchis, y la colaboración con Francisco Mas.

La cuidada delineación de las figuras 'no informáticas' es debida a Antonio Fontes; y Vicent Carmona y Magín Lapuerta han prestado una inestimable colaboración en la edición final.

Por último, quiero dar las gracias a todos mis compañeros y amigos, que han soportado con estoica resignación los relatos de mis tribulaciones. Sin su muy oportuna oposición, este trabajo no hubiera llegado a su fin.

INDICE

INDICE DE FIGURAS

NOTACION

A lo largo del texto se utiliza una notación tan estandarizada como es posible. En esta sección vamos a describir las características mas destacadas de dicha notación, y a incluir una lista con el significado de todos los símbolos utilizados.

Los vectores y las matrices se representan por letras en negrita. Generalmente se respeta el criterio de utilizar letras mayúsculas para las matrices y minúsculas para los vectores. No obstante, a veces se representa un vector con una letra mayúscula, para indicar que dicho vector está referido a un sistema de coordenadas global.

Un componente de un vector se representa por la misma letra (sin subrayar). seguida del correspondiente subíndice para indicar qué componente es. Análogamente, un componente de una matriz se indica con la misma letra (sin negrita), seguida de los subíndices que indican el número de fila y columna del componente.

La derivada parcial $\delta y/\delta x$ representa la relación del cambio en la variable y, inducido por el cambio unitario de la variable x. El símbolo $\delta y/\delta \underline{x}$, indica el vector en el que cada componente es la derivada parcial de y respecto a cada elemento del vector. Es decir:

$$x = (x1,x2,...,xn) \Rightarrow \delta y/\delta x = (\delta y/\delta x1, \delta y/\delta x2, ..., \delta y/\delta xn)$$

En el caso de que y sea un vector, denotaremos:

$$\delta y/\delta x = \begin{bmatrix} \delta y1/\delta x1 & \delta y1/\delta x2 & \dots & \delta y1/\delta xn \\ \delta y2/\delta x1 & \delta y2/\delta x2 & \dots & \delta y2/\delta xn \\ \vdots & \vdots & & \vdots \\ \delta ym/\delta x1 & \delta ym/\delta x2 & \dots & \delta ym/\delta xn \end{bmatrix}$$

En algún momento se emplea la notación reducida para las derivadas (prima, doble prima, etc.).

Se emplean subíndices para particularizar una componente del vector o la matriz considerados. Los supraíndices indican un caso particular del símbolo a que acompañan (p.e. * indica una magnitud virtual).

LISTA DE SIMBOLOS

a	distancia.
A	área de la sección recta de un elemento.
A_{rt}	Area de los rigidizadores transversales:
A_{rl}	Area del rigidizador longitudinal.
B	matriz que relaciona las deformaciones con los desplazamientos.
B_j	coeficientes de aproximación de curvas.
b	ancho de las alas.
C, C_1, C_2	coeficientes de las hipérbolas a que se aproximan los desarrollos en serie de Taylor aproximadas.
C_{zP}	coordenada Z, del punto P, respecto al sistema del punto de vista.
CPV	coordenada X ó Y, del punto P, respecto al sistema del punto de vista.
CPC	coordenada X ó Y de la proyección del punto P, respecto al sistema del plano del cuadro.
C_{alma}	costo de los materiales del alma (con preparación).
C_{alas}	costo de los materiales de las alas (con preparación).
C_{rt}	costo de los materiales de los rigidizadores transversales (con preparación).
C_{rl}	costo de los materiales del rigidizador longitudinal (con preparación).
C_{sa}	costo de soldadura entre el alma y las alas.
D	matriz de propiedades elásticas.
d	desplazamiento.
d_k	dirección de búsqueda para la k-esima iteración.

d_1	componente de máximo desdenso de la dirección de búsqueda.
d_2	componente de máximo acercamiento a la región factible de la dirección de búsqueda.
DPC	distancia desde el PM hasta el PC.
e_a	espesor del alma.
e_b	espesor de las alas.
E	módulo de elasticidad. También factor de corrección (o escala).
f	función objetivo. También la flecha.
f_p	función objetivo penalizada.
F	fuerza. También una función de una variable.
f_m	esfuerzos sobre el m-esimo elemento.
g_j	cada una de las md funciones de las variables que expresan las restricciones de desigualdad.
$G_i(g_i)$	una función de las restricciones.
h_j	cada una de las mi funciones de las variables que expresan las restricciones de igualdad.
h_a	altura del alma.
h_{r1}	altura de colocación del rigidizador longitudinal.
H	canto.
H	matriz de transformación de desplazamientos.
I	momento de inercia.
J_{ak}	conjunto formado por todas las restricciones de igualdad más las restricciones activas en la iteración k.
J_{ik}	todas las restricciones no pertenecientes a Jak.
K	matriz de rigidez de la estructura.
K_m	matriz de rigidez del m-esimo elemento.
l	longitud.
L	función lagrangiana. También una longitud.

M	momento flector.
m	número de restricciones.
m_i	número de restricciones de igualdad.
m_d	número de restricciones de desigualdad.
n	número de variables.
N_e	número de elementos.
N_{rt}	número de rigidizadores transversales:
P	carga puntual.
PC	plano del cuadro.
PV	punto de vista.
PM	punto de mira.
q	carga distribuida. También una polinomio.
Q	esfuerzo cortante.
r	factor de penalización.
R_f	región factible.
R	vector de cargas.
s_{rt}	separación de los rigidizadores transversales.
T	matriz de transformación.
U	vector de desplazamientos.
u	vector de multiplicadores de lagrange.
W	matriz definida positiva.
x	vector n-dimensional que contiene a las variables de diseño.
x_I, x_S	límites inferiores y superiores de las variables.

α	longitud de búsqueda.También un ángulo.
β	factor de relajación. También un ángulo.
γ	peso específico.
∇	operador nabla (gradiente). También un incremento finito.
∇^2	operador laplaciano (hesiano).
ϵ	constante positiva muy pequeña.
$\boldsymbol{\epsilon}$	el vector de deformaciones.
λ	seudodesplazamientos (variables complementarias).
μ	variables asociadas a los multiplicadores de Lagrange.
ζ	partición de intervalo de búsqueda.
η	partición de intervalo de búsqueda.
ρ	parámetros de penalización.
σ	tensión normal.
τ	tensión tangencial.
$\boldsymbol{\tau}$	vector de tensiones.
ω	coeficiente de pandeo.

CAPITULO 1

ANTECEDENTES, ESTADO ACTUAL Y TENDENCIAS.

1.1 INTRODUCCION.

El fin perseguido por cualquier tipo de diseño es el de encontrar una solución a un problema planteado, cumpliendo una serie de requerimientos de diferente naturaleza. Uno de los objetivos implícitos de cualquier diseño es que la solución encontrada sea la mejor de todas las posibles, pero el problema solo se dice de diseño 'óptimo' cuando la búsqueda de la mejor solución se convierte en el objetivo prioritario.

El problema de diseño es de por sí muy amplio, puesto que tanto el objetivo del diseño como los condicionamientos del mismo pueden ser muy variados. Los condicionamientos a considerar pueden ser tan dispares como los criterios estéticos y los requerimientos funcionales. Cuando, además, se pretende optimizar, el problema se vuelve tan complejo que es imposible tratarlo globalmente. Entre otras razones, por la falta de objetividad de algunos de los aspectos a optimizar. Entonces, las únicas alternativas consisten en reducir la cantidad de factores a considerar (hasta enfrentar un problema que sea abordable), o descomponer el problema (tratando por separado aspectos que sean lo mas independientes posible).

En el campo de la técnica, los aspectos estéticos y similares han tenido tradicionalmente una importancia relativa frente a los condicionamientos puramente funcionales. Por ello la descomposición necesaria para abordar el problema resulta casi "natural" en este campo. Así, centrandonos en los aspectos puramente "técnicos" del diseño, el concepto de mejor queda reducido a la búsqueda del diseño mas barato de los que cumplan todos los condicionamientos. No obstante, cuando los métodos disponibles no son suficientes para encontrar el mejor diseño, se suele considerar como mejor a aquel diseño que optimice el, o los, condicionamientos mas criticos, olvidandose del costo.

Una aspiración aspiración general en cualquier tipo de diseño es la de desarrollar métodos, susceptibles de automatización, que permitan encontrar el mejor de los diseños posibles. El desarrollo de toda la tecnología asociada a la informática ha permitido potenciar este aspecto del diseño. El campo del diseño de estructuras fué uno de los primeros en los que se introdujo el empleo de ordenadores. La parte del diseño más susceptible de ser automatizada (el análisis de la respuesta de la estructura), se convirtió en el objetivo de numerosos desarrollos ya en los años 50. Por otra parte, el reto constante en la industria aérea de reducir el peso manteniendo las características resistentes necesarias para el funcionamiento correcto de las estructuras, fué el principal motor de los primeros desarrollos en el campo de la optimización de estructuras. El posterior aumento de las prestaciones de los ordenadores, unido a la disminución de su costo, hizo posible, y rentable, aplicar las técnicas de optimización de estructuras ya conocidas, en el campo más general de la ingeniería civil.

Durante la pasada década, se desarrollaron muchos programas de diseño optimizado. Sin embargo, estos programas no han alcanzado la difusión que cabría esperar. Las razones de la falta de popularidad del diseño optimizado derivan de la propia complejidad del problema. El utilizar técnicas muy complejas, y aún no bién establecidas, ha llevado a desarrollos muy sofisticados, pero difíciles de usar. Por ello, para que la optimización de estructuras sea aceptada y ampliamente usada, es necesario disponer de programas de calidad y bien documentados, pero sobre todo fáciles de usar. Es en este aspecto, de facilidad de uso, donde las técnologías C.A.D. (Diseño Asistido por Computador) deben aportar todos sus recursos. En especial, se debe dotar al diseño optimizado de toda la capacidad de comunicación de los gráficos, y todos los recursos de interacción con el usuario, que han sido desarrollados en el campo del diseño asistido.

A continuación vamos a resumir la evolución del diseño óptimo de estructuras. El resumen no pretende ser exaustivo; por el contrario, solo se destacan los aspectos que permiten situar el problema en los términos en que se va a considerar en el presente trabajo.

Al final del capítulo se detallan los antecedentes inmediatos, así como una primera visión del alcance de este trabajo.

1.2 ANTECEDENTES.

Hasta el final de la década de los 50 la optimización de estructuras estaba reducida a la resolución de problemas muy concretos por medio de técnicas particulares. Así, se diseñaban componentes estructurales (columnas, paneles rigidizados,...) de forma que se alcanzaran simultaneamente varios modos de fallo. Estos métodos daban una solución, no siempre óptima, pero bastante buena. Su grán inconveniente radicaba en que su éxito dependía totalmente de la elección intuitiva de que condiciones de las que debía cumplir el diseño, iban a ser críticas (cumplirse sin holgura) en el óptimo.

En la búsqueda de métodos de diseño en los que no fuera necesario recurrir en cada caso a la experiencia del diseñador, se abrieron dos caminos: a) los criterios de optimalidad (CO), y b) la programación matemática (PM). El primer camino se basa en la idea de que conociendo el comportamiento de la estructura en el óptimo, se pueden extraer condiciones generales que permitan hallar ese óptimo para cada caso particular. De este planteamiento surgieron aproximaciones como la FSD (Fully Stressed Design), que parte de la suposición de que en el óptimo cada elemento está sometido a la máxima tensión en, al menos, uno de los estados de carga. A partir de este criterio, se establece una sencilla regla de rediseño de cada elemento, que permite encontrar el óptimo (o una buena aproximación) en unas pocas iteraciones. Los trabajos de Berke, Gellatly, Venkayya y Khot en los años 60 sentaron las bases de los posteriores desarrollos en este campo.

Por otra parte, la aplicación de métodos matemáticos para la optimización de estructuras, empezó en los años 50. Las primeras aplicaciones se basaban en la programación lineal , que había alcanzado un notable desarrollo como técnica de optimización de procesos. Se aplicó esta técnica al diseño plástico de estructuras planas de barras sometidas a un único estado de cargas. Heyman (1951), Foulkes (1954), Prager (1956), y Livesley (1956) fueron algunos de los investigadores más destacados en este campo.

Casi desde el primer momento, se intentó enfocar el problema de una forma más apropiada a sus características. Así, en 1955 Klein /40/ reconoció la importancia fundamental de las restricciones de desigualdad en la formulación general del problema de diseño óptimo. Así como que el problema más general era un problema no lineal. En 1958, Pearson /60/ trató el diseño óptimo de estructuras de barras sometidas a múltiples estados de carga. En su desarrollo, introdujo

tres ideas de grán importancia: (1) un tratamiento integrado del análisis y la optimización; (2) conversión del problema con restricciones de desigualdad a uno o más problemas equivalentes sin restricciones, y (3) reducción de la dimensión del problema por medio de cambios de variables.

Partiendo de todas estas consideraciones, en la década de los 60 se sentaron las bases de la optimización de estructuras tal como se entiende actualmente. En 1960, Schmit /68/ introdujo la idea, e indicó la viabilidad, de acoplar el análisis por elementos finitos con la programación matemática no lineal. La segunda idea clave de esta década fué la de considerar el diseño de estructuras como un problema multinivel. Lo cual llevó a distinguir entre el diseño global de la estructura y el diseño de componentes. Para el diseño global se desarrollaron, durante esta década, dos grandes sistemas que combinaban los elementos finitos con la programación matemática. El primero lo realizaron Gallagher y Gellatly para Bell Aerosystems; el segundo fué realizado en la Boeing por Karnes y Tocher. Los intentos de diseño de componentes se caracterizaron por: (1) considerar relativamente pocas variables; (3) creciente cantidad de modos complejos de fallo, y (3) funciones objetivos distintas (y más complejas) que el peso.

Cuando estos primeros sistemas empezaron a ser operativos, se hicieron patentes sus carencias. En primer lugar se enfrentó el problema de la gran cantidad de tiempo que requería la resolución de problemas prácticos (incluso los de pequeño tamaño), por medio de los grandes sistemas resultantes de acoplar los elementos finitos con la programación matemática. Un primer paso para superar este problema consistió en aumentar la eficiencia de los programas de elementos finitos, adaptandolos a las particularidades del problema de optimización. Básicamente se enfrentó el problema de realizar gran cantidad de análisis muy similares. Se desarrollaron ideas como la de distinguir entre partes repetibles y no repetibles en el análisis (Bhatia,/8/), o la de realizar análisis aproximados (Melosh y Luik, /53/).

Por otra parte, los métodos matemáticos empleados seguian siendo incapaces de resolver eficientemente la optimización. Por ello, se siguió investigando en busca de mejores métodos matemáticos. Pero también se profundizó en la comprensión de los problemas particulares de la aplicación de estos métodos al diseño de estructuras. Se puso

así de manifiesto que la ineficiencia residía en tres características del problema a tratar: (1) gran cantidad de variables de diseño; (2) gran cantidad de restricciones de desigualdad, y (3) que muchas restricciones son función, indirectamente, de las variables. La distinción entre el modelo de análisis y el de diseño permitió agrupar variables de analisis en una sola variable de diseño, reduciendo así el número de estas últimas. La inclusión en los algoritmos de programación matemática de la capacidad de determinar las restricciones activas (con la consiguiente eliminación temporal de las demás) redujo el número de restricciones que debían ser consideradas a un tiempo. Para solucionar el tercer problema se recurrió a buscar formas de definir explícitamente las restricciones. Para ello, se recurrió a aproximaciones del tipo de los desarrollos en serie.

También se intentó resolver la dualidad entre los criterios de optimalidad y la programación matemática, así como las diferentes soluciones que se fueron proponiendo dentro de cada una de estas técnicas. Se pretendía encontrar el mejor método para la resolución de todos los problemas de diseño. En esta vía deben destacarse los trabajos de Fleury /22/.

Entre tanto los múltiples problemas asociados con el uso del ordenador en el diseño de estructuras, fueron abordados independientemente,dando lugar a varias vias de desarrollo paralelas. Uno de estos problemas era el de gestionar la información que se maneja en un problema de diseño. Se determinó la inviabilidad de utilizar las bases de datos desarrolladas en el campo de la gestión comercial, dadas las claras particularidades que presenta la información a tratar (principalmente por ser muy variada). Los trabajos de Felippa /18/, establecieron la necesidad de un tratamiento específico de esta información. Posteriormente, se definieron modelos apropiados que permitieron unos primeros desarrollos operativos; como el realizado por Rajan y Bhatti /64/.

También el análisis de sensibilidades (que proporciona derivadas de diseño necesarias para la mayoría de los algoritmos de optimización), recibió una atención creciente. De la tendencia original a utilizar métodos particulares embebidos en el propio código de optimización, se pasó, a finales de los 70, a intentar una formulación general y eficiente de este problema. Uno de los métodos para el análisis de sensibilidades, para respuesta lineal bajo cargas estáticas y dinámicas, fué descrito por Arora /1/, ya en 1979. En esta obrá, Arora apuntaba como segunda aplicación del análisis de sensibilidades el

aportar una información que permita que el diseñador haga cambios sabiendo el efecto aproximado que van a tener. Es decir, que la información sobre sensibilidades muestre al diseñador la influencia sobre el diseño de todos los cambios posibles; reduciendo, por tanto, el proceso de prueba y error habitual en la búsqueda de mejoras en el diseño no óptimo.

Otro de los problemas que se empezaron a tratar era el de facilitar el uso de los programas desarrollados. La gran cantidad de datos que necesita un programa de análisis por elementos finitos para definir el modelo de cálculo era la causa de muchas horas de trabajo y múltiples fallos. Para paliar este problema, se desarrollaron programas 'preprocesadores' que generaban la información del modelo, a partir de una información básica dada por el usuario.

Independientemente de todos estos trabajos, al final de los años 60, había florecido un aspecto de la informática que iba a tener una importante aplicación en el diseño de estructuras: los gráficos por ordenador. La utilización de los gráficos por ordenador en los años 50, se limitó a caras y primitivas aplicaciones militares. En la década de los 60, se vivió el despegue en la utilización de los gráficos. Ya en 1963, el programa SKETCHPAD, desarrollado por Sutherland en el Massachusetts Institute of Technology como trabajo de tesis doctoral, incluia ideas fundamentales y utilizaba técnicas que aún están vigentes. Supuso el nacimiento del concepto de interacción gráfica, utilizando periféricos apropiados a tal fin. Así, este trabajo sirvió como punto de partida a las investigaciones sobre periféricos adaptados al uso interactivo de gráficos (DAC-1 de General Motors, GRAPHIC 1 de Bell Telephones, etc), y sobre software gráfico básico (trazado de líneas, recortado, cálculo de líneas ocultas, etc).

Al principio de los años 70, coincidió el desarrollo de preprocesadores para los programas de elementos finitos, con el abaratamiento de periféricos relacionados con las capacidades gráficas de los ordenadores (pantallas, trazadores, etc). Y, puesto que el software básico estaba ya disponible, se inició una rápida incorporación de representaciones gráficas de dicho preproceso. El "Integrated Civil Engineering System" (ICES), sirve como referencia significativa de los resultados que se consiguieron con estos sistemas.

Del enfoque puramente estructural que tenía inicialmente el diseño de estructuras se pasó a uno más matemático, conforme se trabajaba sobre

la optimización de dicho diseño. Pero, paulatinamente, fueron mezclandose el aspecto matemático del diseño óptimo con una serie de aspectos más "informáticos". Los desarrollos que consideraban los diferentes aspectos del diseño, empezaron a ser denominados conjuntamente como "Diseño Asistido por Ordenador" (generalizandose las siglas inglesas CAD). El hecho de que el CAD incluyese una parte gráfica cada vez más potente, unido al desarrollo paralelo de programas de dibujo asistido por ordenador, llevó a que el significado de CAD llegase paulatinamente a ser sinónimo de "dibujo asistido por ordenador". Otros aspectos importantes del uso del ordenador en diseño, empezaron a englobarse bajo las siglas CAE ("Computer Aided Engineering").

1.3 ESTADO ACTUAL.

El método de los elementos finitos se ha decantado como la solución más apropiada para la fase de análisis del problema de diseño. Los programas utilizados en la década de los 70 (NASTRAN, STRUDL, SAP, etc) han evolucionado, o han sido sustituidos por programas más modernos. Los programas más usados actualmente , han sido depurados para obtener un máximo de eficiencia, tanto en velocidad de cálculo como en capacidad de memoria. La práctica totalidad de los programas disponibles comercialmente incluyen sofisticados pre y postprocesadores interactivos y gráficos que permiten definir el modelo y estudiar la respuesta del mismo. En /16/ se recoge una clasificación de los programas disponibles comercialmente, en función de sus principales características.

En cuanto al proceso de optimización, ya ha sido asumido por la comunidad científica que ninguna de las aproximaciones disponibles para la fase de optimización permite resolver satisfactoriamente todos los problemas que plantea el diseño de estructuras /2/. Por ello, la solución adoptada es la de desarrollar sistemas de diseño que incluyan diferentes posibilidades de resolución de la optimización; de forma que el usuario pueda en cada caso seleccionar la más apropiada. Así, en la actualidad, es normal emplear varios algoritmos conjuntamente, pasando de uno a otro en función de la evolución del proceso. H. Hörnlein /35/, muestra en una tabla resumen las principales características de la mayoría de los sistemas usados en la actualidad. La principal conclusión que puede extraerse de dicha tabla es que estos sistemas pueden dividirse en dos campos: (1) los desarrollados por grandes empresas (generalmente industrias aeroespaciales), y (2)

los desarrollados en universidades. De esta división se deduce que el diseño óptimo está bién asentado en industrias de alta tecnología, y que se mantiene el interés por la investigación que permitirá nuevos desarrollos en este campo. Pero es evidente que el diseño óptimo no ha alcanzado la difusión del análisis por elementos finitos.

Por su parte, las siglas CAD están adquiriendo nuevamente un sentido más amplio; pasando a comprender todos los aspectos que facilitan el uso del ordenador en el diseño de estructuras /17/. Utilizando la definición de Encarnaçao y Schlechtendahl /16/, ' el CAD es la disciplina que provee los necesarios "saber-como" en hardware y software, en análisis de sistemas y metodología en ingeniería, para especificar, diseñar, implementar, introducir y usar sistemas para diseño basados en el ordenador'. En este sentido, una de las técnicas CAD que esta recibiendo mayor atención es la del desarrollo y aplicación de bases de datos apropiadas para diseño. Sreekanta /69/ describe los conceptos, y desarrolla la metodología, para el diseño de bases de datos para diseño óptimo. También describe el sistema de manejo de datos MIDAS, que es representativo del estado del arte en la materia. Se ha reemplazado la organización intuitiva de los datos de análisis y diseño por otra basada en las características de la información a tratar y las particularidades de los procesos que sigue dicha información. De forma que estos sistemas sofisticados de manejo de datos puedan solucionar problemas de organización de los procesos de optimización haciendo su implementación más directa y transparente.

Por otra parte, los aspectos básicos del problema están ya perfectamente resueltos. Las librerias de programas que empezaron a aparecer a comienzos de los 70 (p.e. IMSL, HARWELL, etc) han sido depuradas y ampliadas. Por tanto, los programas de aplicación que se desarrollan en la actualidad a partir de estas librerias resultan muy eficientes, al tiempo que robustos. Esfuerzos semejantes se han llevado a cabo en el tratamiento gráfico por ordenador. Durante los últimos quince años se han desarrollado numerosas librerias (p.e. PLOT, HALO, TEKMAR, etc), orientadas a diferentes tipos de ordenadores y periféricos. Un notable esfuerzo de estandarización llevado a cabo por la Organización Internacional para la Estandarización (ISO), derivó en la propuesta del Graphical Kernel System (GKS) como estandar /34/. Aunque, si bién el estado actual hace preveer que el uso de GKS se generalizará a corto plazo, hasta el momento la dispersión sigue siendo alta. En consecuencia la compatibilidad y robustez de los

desarrollos en el campo de la representación gráfica es aún muy limitada.

1.4 TENDENCIAS.

Todos los aspectos del desarrollo de sistemas de diseño de estructuras, se pueden englobar en tres componentes: a) la librería de programas de aplicación; b) la base de datos, y c) la comunicación interactiva con el usuario. Y hemos visto que se han obtenido soluciones bastante eficientes para todos ellos por separado. Por lo tanto, el siguiente paso parece ser el de desarrollar un sistema de diseño que englobe el proceso completo. El objetivo de este hipotético sistema debe ser el de resolver problemas prácticos de propósito general.

Hasta la fecha se han desarrollado sistemas de optimización siguiendo estas pautas. No obstante, estos sistemas no han conseguido la aceptación que se pretendía. La causa de este rechazo no parece ser única, y el problema se puede resumir en la pregunta planteada en la mesa redonda del NATO Advanced Study Institute on Computer Aided Optimal Design, cuyas actas se recogen en /58/. Dicha pregunta fué: "¿ Por que tenemos problemas para que los diseñadores acepten, y usen habitualmente, los métodos de optimización ?".

En respuesta a esta pregunta, Arora /58/ plantea la necesidad de aumentar la fiabilidad de los algoritmos de optimización. Las aplicaciones "académicas" (o, cuanto menos muy especializadas) que se han dado a los algoritmos desarrollados hasta la fecha, han llevado a primar en exceso la eficiencia frente a cualquier otra consideración. Pero, para que la optimización pueda ser empleada en diseños prácticos, lo que se debe primar es la robustez. Además, como no se dispone de ningún algoritmo general, capaz de resolver todo tipo de problemas, se recurre a utilizar diferentes tipos de algoritmos; cambiando de uno a otro a medida que el diseño progresa, o en función del tipo de diseño. Esto aumenta la necesidad previa de hacer más robustos los algoritmos. En definitiva se tiende a establecer una biblioteca de aplicaciones que incluya varios algoritmos de optimización que puedan ser usados alternativamente, en función de las características del problema o del progeso de su resolución.

Otra línea de actuación para eliminar el rechazo del diseño óptimo es la apuntada por Haug /58/: se debe pasar de concebir el diseño óptimo

como una tarea automática ("black-box"), a tratarlo como un proceso interactivo que exija la intervención del usuario. Una de las finalidades de este cambio de concepción el la de permitir al diseñador con experiencia (que se muestra escéptico ante el diseño optimo) seguir la evolución del proceso; de forma que al conocer y controlar el proceso de optimización se familiarice con él. Como ayuda a esta interacción, Haug plantea la conveniencia de la incorporación de los gráficos. Fleury /58/ va más allá, insistiendo en la importancia de encontrar formas innovadoras de representación que permitan, primero familiarizarse, y despues controlar el proceso de diseño.

Por otra parte, el diseño interactivo implica la necesidad de interrumpir el proceso en ciertos momentos y modificar el flujo de la ejecución, lo cual aumenta la necesidad de centralizar la información. Por ello, el futuro inmediato de la aplicación de las bases de datos es el de perfeccionar los modelos en los que se basan. Además, para que el diseño óptimo llegue a un gran número de usuarios, se deben conseguir implementaciones para mini y microordenadores. Para este fin, las bases de datos deben ser capaces de usar eficientemente todos los recursos de estos ordenadores /69/.

Otra vía, apuntada por Fleury /23/, es la de desarrollar modelos orientados al diseño, en lugar de los modelos orientados al análisis que se están empleando. Para ello, propone la utilización de los conceptos de modelos geométricos empleados en CAD.

Todo lo dicho se puede resumir diciendo que se tiende a definir un modelo ajustado al proceso de diseño (empleando conceptos habituales en CAD), que debe ser diseñado, de forma interactiva (con gran participación de representaciones gráficas), por un sistema definido en torno a una base de datos apropiada al problema. El sistema debe ser robusto (para lo cual debe generarse a partir de librerias estandarizadas), y capaz de resolver problemas prácticos de propósito general (flexible). Por último (pero no menos importante), el sistema debe ser fácil de usar.

1.5 ANTECEDENTES Y FINALIDAD DE ESTE TRABAJO.

Los antecedentes de este trabajo pueden situarse en el desarrollo de un programa de análisis por elementos finitos (ADEF) llevado a cabo por M. Sanchis y el propio autor, como trabajos final de carrera, bajo la dirección de P. Martí /49/. Posteriormente, y siguiendo la línea de investigación del profesor Martí, se implementó una versión simplificada de un programa de análisis matricial (PAM) para utilizarlo con un algoritmo de programación matemática, en una primera versión de sistema de diseño (DISSENY). Se pretendía utilizar un programa sencillo (pequeño y con pocos requerimientos de memoria), y rápido, para adquirir la experiencia necesaria en el funcionamiento del algoritmo de optimización. También se modificó el programa ADEF, adaptandolo a una versión más estandarizada del lenguaje de programación (FORTRAN), y depurando la gestión de datos /46/. Este fué un paso previo para una segunda revisión y actualización del programa ADEF, que fué adaptado para su funcionamiento en ordenadores personales (ADEF/88).

Paralelamente, F. Mas implementó un algoritmo de optimización FSD (Stress-ratio) que conectó a una versión simplificada del programa (ADEF/ART) ,como trabajo final de carrera bajo la dirección de P. Martí /51/.

Se contaba, por tanto, con un programa de análisis por el método de los elementos finitos, y con un algoritmo de optimización por programación matemática no lineal. Ambos eran operativos por separado, y el objetivo era unirlos para obtener un sistema de diseño óptimo. Puesto que el sistema debía ser de propósito general, se buscaban los objetivos de robustez, flexibilidad y facilidad de uso. Para lograr estos objetivos, se recurrió a la aplicación de las tecnologías CAD.

Debe destacarse que existía una experiencia previa, acumulada en los trabajos anteriores. Así, ya se ha comentado que la gestión de datos había sido potenciada en la versión ADEF/88 del programa de análisis. Además, este programa contaba con unas incipientes posibilidades de interacción. El otro aspecto de las tecnologías CAD a considerar (la representación gráfica), había sido abordado en el preprocesador de la primera versión del programa ADEF /49/. No obstante, aquella primera implementación se llevó a cabo en una instalación totalmente obsoleta, por lo que la experiencia de utilización de hardware gráfico era desechable (aún representando mucho trabajo, debido a las grandes dependencias de máquina que hay en este campo).

En definitiva, se contaba con un sistema de diseño implementado en un ordenador potente, pero útil solo para ejecuciones automáticas. Además, el sistema solo era utilizable por un usuario experto; ya que parte del software era específico para cada problema, y era difícil de modificar. Por tanto, la implementación del sistema interactivo debía considerar los siguientes aspectos:

- hacer robusto el algoritmo de optimización;
- conectarlo con ADEF/88, en lugar de PAM;
- hacerlo interactivo el sistema;
- mantener los cálculos en un ordenador potente, llevandose la interacción a un ordenador personal, y
- hacer un uso intensivo de gráficos para que el sistema fuera de fácil uso.

Para hacer posible esta solución, se marcaron dos objetivos al presente trabajo:

- adaptar y ampliar la gestión de datos y el control interactivo, y
- estudiar y desarrollar nuevas formas de representación gráfica del proceso de diseño.

Con la primera tarea se pretendía definir la configuración del sistema desarrollado para que permitiese un alto nivel de interacción con el usuario, y pudiese estar distribuido de forma que las partes que implican cálculos puedan realizarse en ordenadores potentes, mientras que la interacción con el usuario se lleve a cabo en ordenadores personales con mejores capacidades gráficas (y con mayor disponibilidad).

La segunda tarea implica el estudio de formas de representación que permitan al usuario conocer, en todo momento, como se esta desarrollando el diseño. Básicamente, se pretendía representar la evolución del proceso (historia de todos los valores que influyen en el diseño), y el estado del mismo (representación del espacio de diseño). Para ello, una vez decidida la forma de representación más apropiada, se pretendía sacar el mayor rendimiento posible a toda la información disponible. Se trataba de obtener una representación lo

más cercana posible a la realidad, pero sin realizar cálculos para obtener información adicional a la requerida por el propio proceso.

Finalmente, la implementación del sistema y su utilización en la resolución de casos prácticos, debería hacer patentes las carencias del mismo y sugerir posteriores desarrollos.

CAPITULO 2

EL PROBLEMA DEL DISEÑO OPTIMO DE ESTRUCTURAS

2.1 INTRODUCCION.

Diseño es una palabra que no tiene un significado único hoy en día. A modo de ejemplo, podemos citar a Jones /37/, que recoge diferentes definiciones de lo que es diseño, y concluye que parece que hay "tantas clases de procesos de diseño como escritores hay sobre ello". Posteriormente, intenta extraer las características generales de los "aparentemente" diferentes tipos de diseño para llegar a una definición única. No obstante, entre las definiciones citadas por Jones, podemos tomar una que se adapta perfectamente a lo que vamos a entender por diseño en este contexto. Dicha definición, debida a Fielden, es:

"El diseño técnico es la utilización de principios científicos, información técnica e imaginación en la definición de una estructura mecánica, máquina o sistema que realice funciones específicas con el máximo de economía y eficiencia".

El primer aspecto destacable de esta definición es el de plantear la coexistencia del método analítico con el uso de la imaginación, para llegar a la solución buscada. El segundo aspecto importante es el de considerar el diseño como la búsqueda de la mejor de las soluciones posibles.

Desde este enfoque podemos ver al método tradicional de diseño como un método básicamente imaginativo. Dicho método consistia en utilizar la experiencia previa del diseñador, o hacer experimentos cuando no existía dicha experiencia. Se diseñaba en función de la experiencia previa, y se comprobaba experimentalmente la utilidad del diseño elegido. La experimentación podía, en algunos casos, poner de manifiesto ciertas mejoras menores. Con la aparición del ordenador se pudo sustituir la parte experimental por el análisis numérico. De esta forma, el estudio del comportamiento de módelos matemáticos permitía conocer la respuesta del diseño. Dado que el análisis numérico es

menos costoso que la experimentación, se podían (utilizando la experiencia, o haciendo tanteos) realizar ciertas modificaciones, que eran analizadas por separado y comparadas posteriormente para elegir la mejor. Se llegó así a un método de prueba y error en el que el diseñador definía un diseño inicial cuyo comportamiento era analizado numéricamente. De los resultados de dicho análisis se podían deducir (por experiencia o intuición) los cambios a realizar para mejorar dicho diseño. El esquema es el mostrado en la figura (2.1). El inconveniente de esta forma de diseño es que la elección de las modificaciones depende totalmente de la experiencia del diseñador. Por lo tanto la solución alcanzada es buena, pero raramente la mejor.

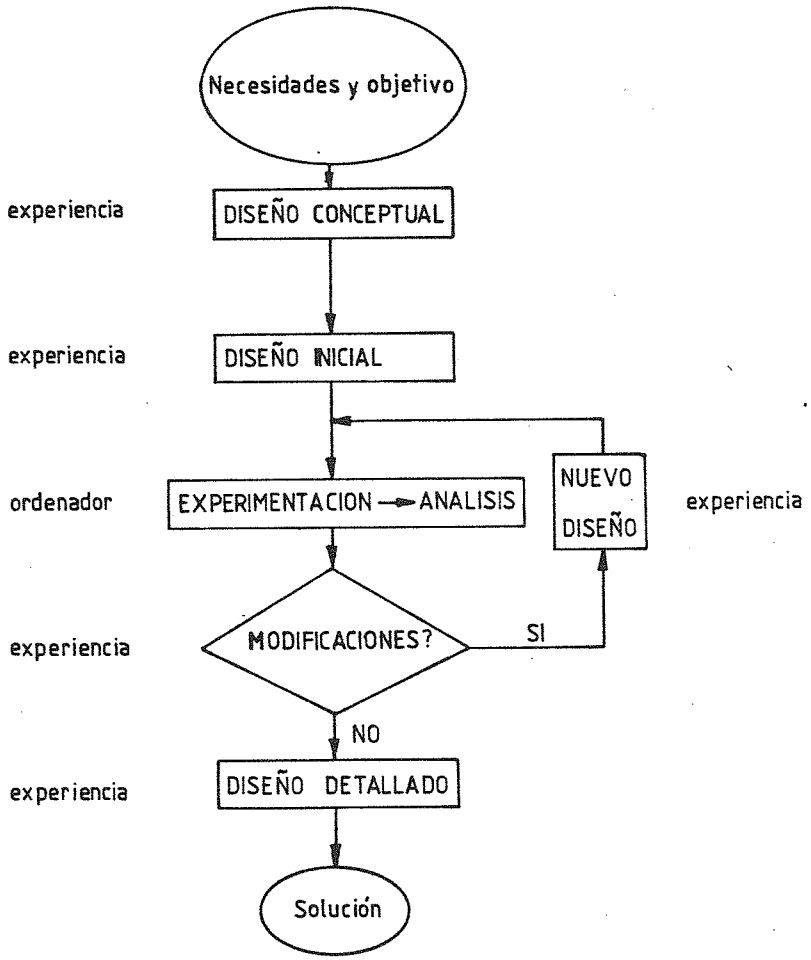

Figura 2.1. Diseño por prueba y error.

Para optimizar el resultado (que la solución obtenida sea la mejor posible) caben dos alternativas: a) comparar todas las soluciones posibles, y b) buscar criterios generales que permitan encontrar la solución óptima en cada caso; bién por una formulación general, o bién a traves de un número finito de aproximaciones sucesivas. Inicialmente, se intentó la primera alternativa, comparando únicamante aquellas soluciones que la experiencia indicaba como más susceptibles de ser óptimas. Dada la falta de generalidad de este método, se intentó formular el problema de forma que fuese apto para resolverse automáticamente. Esto llevó a una reconfiguración del proceso de diseño como la mostrada en la figura (2.2).

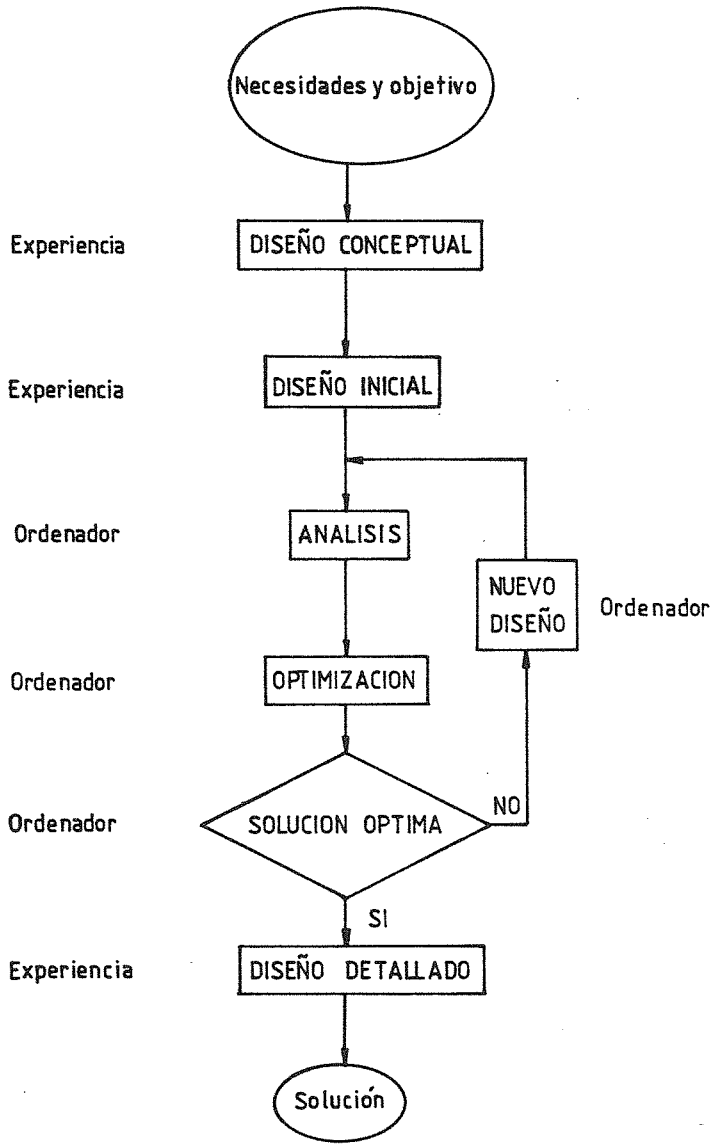

Figura 2.2. Diseño optimizado.

En algún sentido, todo proceso de diseño siempre puede considerarse como un proceso de optimización. Aunque ni las necesidades ni el objetivo del diseño incluyan explícitamente la búsqueda de la mejor de las soluciones, esta estará incluida en otros aspectos tales como facilidad y economía de producción, uso, mantenimiento, etc. Sin embargo, todavia se considera a la optimización como una parte del proceso más general de diseño. La razón de esto radica básicamente en la incapacidad que las técnicas actuales de optimización tienen para abarcar el proceso completo. Esto es debido a que para poder encontrar automáticamente una solución óptima, el objetivo perseguido y todos los condicionamientos que este debe cumplir, deben ser susceptibles de explicitarse formalmente. Los aspectos más complejo implicados en el diseño no se pueden formular aún explícitamente, y por otra parte, los aspectos que si se pueden formular pueden llegar a ser tantos que hagan que el problema definido resulte inmanejable por las técnicas disponibles.

Por ello, un paso previo a la tarea de diseño es el de definir una idealización del objeto a diseñar, a fin de obtener un modelo que incluya un número finito de los aspectos más importantes del diseño y sea más sencillo de manejar que el objeto real. Lo habitual es reservar, para incluir en el modelo de optimización, aquellos aspectos en los que la experiencia tiene una menor importancia, y que ,por contra, implican un grán esfuerzo de cálculo. De ahí que en la figura (2.2) puede verse como tanto el diseño conceptual como el diseño detallado (que son aún partes del proceso de diseño en las que interviene fundamentalmente la experiencia y el juicio del diseñador) no se incluyen normalmente en el proceso automático de optimización.

Una vez hecha la idealización del objeto, la forma habitual de definir los aspectos a incluir en el diseño consiste en especificar formalmente lo que se puede modificar y como se puede modificar. Además, para definir la finalidad del diseño, se especifican las características que debe cumplir la solución para ser la mejor. Para ello, se recurre generalmente a definir el problema agrupando sus características bajo tres aspectos: a) las variables del diseño; b) las restricciones, y c) la función objetivo. A continuación vamos a definir y describir cada uno de estos aspectos del proceso de diseño; si bién, en las obras de Fox /25/ y Gallagher /26/ se pueden encontrar explicaciones más detalladas.

Por otra parte, estos conceptos deben usarse con diferentes métodos para resolver problemas de distinta naturaleza. Por ello, se incluye una descripción de las particularidades mas destacadas de cada tipo de aplicación, de forma semejante a la que puede encontrarse en las obras de Kirsch /39/ y Haug y Arora /31/. Se describen tres tipos de aplicaciones implementados actualmente en el sistema DISSENY: a) Estructuras espaciales de nudos articulados; b) estructuras espaciales de nudos rígidos, y c) elementos estructurales.

2.2 VARIABLES.

Para realizar un diseño se debe asignar un valor a todas las magnitudes que intervienen en el mismo. En principio, modificando cualquiera de estas magnitudes se obtiene un diseño diferente. Por lo tanto, para encontrar la solución óptima, se deberían considerar todas las posibles combinaciones de valores de todas las magnitudes que diesen soluciones factibles. Lo que ocurre es que, generalmente, los problemas tienen una serie de condiciones impuestas externamente que obligan a fijar algunas de estas magnitudes. Por ello, las magnitudes pueden dividirse en:

- parámetros del problema, y

- variables del problema.

Así, por ejemplo, en la figura (2.3). se muestra que si queremos diseñar una viga empotrada para que sujete un objeto, la solución óptima la obtendremos eligiendo las características apropiadas de dicha viga (E,A,I,L). Pero no podremos elegir ni el peso que podrá soportar (P), ni la posición en la que actuará dicho peso (a).

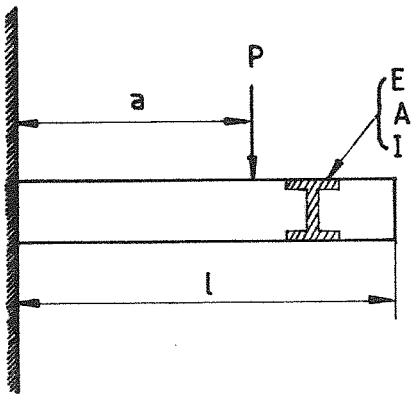

Figura 2.3. Parametros y variables de un problema.

Además de los condicionamientos externos, la capacidad del método de búsqueda de la solución también limita la posibilidad de hacer variables o parámetros ciertas características del problema. En el ejemplo anterior, se podrían estudiar topologías alternativas a la dada (cercha plana, estructura a base de cables, etc), siempre que el método de resolución fuese capaz de tratar como variables tales características. En el caso más general, se distinguen cuatro tipos de variables (en función de la complejidad creciente que plantea su optimización):

- de propiedades de la sección transversal de los elementos;

- de geometría;

- de topología, y

- de material.

Actualmente, no existen técnicas de optimización que puedan considerar eficientemente los cuatro tipos de variables. Por ello, lo habitual es considerar como parámetros al material, e incluso a la topología.

Por otra parte, algunas de las características del problema admiten infinitas soluciones, mientras que otras solo aceptan un conjunto limitado. Por ello, es habitual distinguir dos tipos de variables:

- continuas, y

- discretas.

Volviendo al ejemplo anterior, la longitud de la barra puede tomar el valor que se estime más oportuno (es un caso claro de variable continua), mientras que la sección de la misma convendría limitarla a ser alguno de los valores de los perfiles disponibles comercialmente (variable discreta). No obstante, los métodos de optimización disponibles actualmente trabajan, en su mayoría, solo con variables continuas. Por lo que el uso de variables discretas está limitado a algunos métodos aún no muy establecidos. En la tesis doctoral de Wu /61/, puede conocerse el estado del arte sobre el tema.

2.3 RESTRICCIONES.

Restricción es toda condición que el diseño buscado deba cumplir obligatoriamente para ser aceptable. La forma habitual de expresar

estas condiciones es como funciones de las variables, a las que se les exige tomar algún valor.

Las restricciones sobre un problema de diseño pueden tomar varias formas, y es conveniente hacer distinciones entre los diferentes tipos posibles porque generalmente pueden tener tratamientos específicos más efectivos que cualquier tratamiento general. Así, una primera distinción entre restricciones es la dada en función de aquello que condicionan:

- las variables del diseño, o

- la respuesta de la estructura.

Las primeras son las que imponen condiciones a las características que tendrá la solución. Las segundas imponen las condiciones sobre el comportamiento que debe tener dicha solución. Esta diferencia se puede expresar de otra forma, ligeramente distinta, planteando que algunas restricciones actuan directamente imponiendo condiciones a una variable (o un grupo de variables), mientras que otras restricciones imponen condiciones sobre magnitudes que dependen a su vez de las variables. De esta forma, la distinción es entre restricciones explícitas e implícitas. La distinción entre ambos tipos de variables es importante porque las explícitas tienen un tratamiento más fácil en la mayoría de métodos de diseño.

Una segunda distinción de las restricciones proviene de diferenciar el tipo de condición que plantean:

- restricciones de igualdad, y

- restricciones de desigualdad.

Las primeras son aquellas en las que la condición se plantea de forma que la restricción debe tomar un valor en concreto. Por el contrario, las segundas únicamente dividen el conjunto de los posibles valores de las variables en dos partes: a) los que no cumplen la condición, y b) los que la cumplen. Esta distinción viene propiciada por el tratamiento diferente que se da a cada tipo de restricción. En pura teoría, las restricciones de igualdad deberían permitir reducir el número de variables, sustituyendo estas por el valor constante que se dedujese de la condición de igualdad. En la práctica, la dificultad de las operaciones algebráicas implicadas en tal proceso hace que no puedan eliminarse. A pesar de ello, las restricciones de igualdad son

más fáciles de tratar. Además, la mayoría de las restricciones de igualdad corresponden al sistema de ecuaciones de equilibrio, y suelen considerarse por separado (utilizando un método de análisis del comportramiento de la estructura). No obstante, la mayoría de las condiciones que se deben imponer en diseño de estructuras no pueden expresarse de esta forma. Por ello, en muchos métodos no se prevee el tratamiento de restricciones de igualdad. Siendo frecuente convertir dichas restricciones de igualdad en parejas de restricciones de desigualdad y trabajar únicamente con estas últimas.

2.4 FUNCION OBJETIVO.

La finalidad perseguida en el diseño debe ser explícita para que este pueda automatizarse. Por lo tanto, se debe definir una relación entre todas las variables que intervienen en el diseño, de forma que se pueda plantear como objetivo de la optimización el encontrar un conjunto de valores de las mismas que haga que la relación elegida cumpla unas condiciones impuestas.

La forma más sencilla de establecer una relación entre todas las variables de diseño es definir una función escalar de las mismas, y la condición habitual que se le exige a dicha función es que para la solución óptima tome un valor mínimo. Una función de este tipo es lo que se conoce como 'función objetivo' del problema de optimización.

Esta forma de definir las funciones objetivo viene en gran medida heredada del hecho de que las primeras aproximaciones a soluciones óptimas se hicieron en el campo aeronáutico. En este tipo de aplicaciones el condicionamiento esencial es el del peso de la estructura. Por ello, lo habitual era plantear como función objetivo el peso de la estructura (en función de las variables de diseño), y perseguir la minimización de dicho peso. Este planteamiento se ha mantenido porque se adapta bién a la formulación matemática del proceso y es susceptible de generalizarse sin más que definir la función objetivo apropiada para considerar los aspectos que interese optimizar.

De hecho, en la actualidad es habitual emplear funciones objetivo que consideren otros aspectos. Una de las funciones objetivo más usuales es el coste total de una estructura. Dicho tipo de función objetivo se define por medio de unos coeficientes de ponderación que suelen ser costos unitarios de materiales o procesos de construcción.

2.5 OPTIMIZACION DE ESTRUCTURAS DE NUDOS ARTICULADOS.

Hemos visto que en el caso mas general pueden intervenir cuatro tipos de variables. No obstante, actualmente no existen técnicas de optimización que puedan considerar, eficientemente, los cuatro tipos de variables en la optimización de estructuras de nudos articulados, por lo que es habitual considerar fijos el material y la topología de la estructura, empleando únicamente variables de propiedades y de geometría.

El problema de diseño óptimo con geometría variable presenta mayores dificultades que el de propiedades variables, debido a la diferente naturaleza de ambos tipos de variables. Para superar estos problemas se han propuesto diferentes métodos: Vanderplaats y Moses /76/ realizan la optimización en dos espacios de diseño separados, empleando una modificación del algoritmo de Zoutendijk's de direcciones posibles; Pedersen /77/ trata todas las variables simultaneamente, resolviendo el problema no lineal mediante una secuencia de pasos lineales; Kanji y Schmit /78/ resuelven este mismo problema con el método de los multiplicadores. En el sistema DISSENY se emplea un método de programación cuadrática sucesiva para resolver el problema considerando todas las variables simultaneamente. Para el buen funcionamiento del método se emplean técnicas de agrupamiento de variables y restricciones que exponemos a continuación.

2.5.1 Reducción de las variables de diseño.

Generalmente, en el diseño de estructuras de nudos articulados, se toman como variables las áreas de las secciones rectas de los elementos (dichos elementos son prismáticos de sección constante, y están sometidos a esfuerzos axiales únicamente). También es habitual tomar como variables las inversas de las áreas, para mejorar el funcionamiento de ciertos métodos de optimización. En cualquier caso, se llega a un problema en el que se tienen tantas variables como elementos. De ahí que, para las estructuras prácticas (del tipo de las grandes mallas espaciales) el número de variables suele ser muy grande; por ello es conveniente agruparlas para reducir el tamaño del problema. La forma de realizar estos agrupamientos en el sistema DISSENY está tratada en /45,47/.

Por otra parte, el agrupamiento de variables es una forma cómoda de introducir condiciones prácticas de diseño, tales como igualdad de

perfiles, simetrías, etc. Además, en los programas de elementos finitos, es práctica corriente el definir grupos de elementos con las mismas propiedades. Respecto al agrupamiento de elementos, en el caso de las variables de propiedades de los elementos, es posible expresar las áreas de los mismos en función de un número menor de áreas independientes. Este agrupamiento se puede formular matricialmente como:

$$A^E = A^{OE} + T^{EG} A^G \qquad (2.1)$$

Siendo:

A^E el vector que contiene las áreas de los elementos (habiendo n elementos).

A^G el vector que contiene las áreas independientes (será m-dimensional, siendo $m \leq n$).

T^{EG} una matriz de nxm, que relaciona el área de cada elemento con el vector de áreas independientes.

A^{OE} un vector n-dimensional de coeficientes constantes, que ponderan el efecto de cada componente.

De modo análogo, las variables de geometría correspondientes a coordenadas de nudos, pueden ponerse en función de un número menor de coordenadas independientes:

$$C^N = C^{ON} + T^{NG} C^G \qquad (2.2)$$

En ambos casos, el agrupamiento habitual en los programas de elementos finitos, da lugar al caso particular en el que el vector de coeficientes constantes se hace cero, y las filas de la matriz T tienen todos los valores igual a cero excepto uno, que toma el valor unidad.

Los agrupamientos anteriores son suficientes para incluir la mayoría de condiciones prácticas de diseño y de simetría. Sin embargo los valores de las variables de propiedades y de geometría suelen ser muy diferentes. Esto produce un mal condicionamiento del problema y relaciones de convergencia pobres en muchos algoritmos de optimización. Para el buén funcionamiento de dichos algoritmos se requiere que todas las variables de optimización tengan el mismo orden de magnitud. Por tanto, es conveniente establecer unas relaciones

entre las variables de diseño anteriores y las variables de optimización para igualar los órdenes de magnitud. Así pues, se establecen las relaciones:

$$A^G = A^{OG} + T^{GVE}\, X^E \qquad (2.3)$$

$$C^G = C^{OG} + T^{GVN}\, X^N \qquad (2.4)$$

Siendo X^E las variables de optimización asociadas con las áreas de los elementos, y X^N las asociadas con las coordenadas de los nudos.

2.5.2. Agrupamiento de las restricciones de diseño.

La distinción entre las variables asociadas al diseño y las variables de la respuesta de la estructura es útil. Porque, si la respuesta de la estructura puede calcularse fácilmente para un valor dado de las variables asociadas al diseño, las ecuaciones de eqilibrio de la estructura pueden tratarse por separado; eliminandose de las restricciones del problema de optimización. En el caso de las estructuras de barras de nudos articulados resulta sencillo obtener la respuesta de la estructura por medio de un análisis de tipo matricial o por elementos finitos, por lo que las restricciones que habrá que considerar serán las de: desplazamiento; tensión; pandeo, y las debidas a los condicionamientos prácticos de diseño.

Como restricciones de desplazamiento se imponen límites a la traslación de los nudos:

$$d_i^{min} \leq d_i(x) \leq d_i^{max} \quad i = 1,2,\ldots,N^o \text{ de nudos} \qquad (2.5)$$

Cada una de estas condiciones pueden expresarse mediante las condiciones siguientes:

$$g_{i1}(x) = 1 - d_i(x) \,/\, d_i^{max} \qquad (2.6)$$

$$g_{i2}(x) = -1 + d_i(x) \,/\, d_i^{min} \qquad (2.7)$$

Esta forma de plantear las restricciones resulta muy conveniente, ya que con ella todas las restricciones toman valores cercanos a la unidad, lo que mejora el funcionamiento de la mayoría de algoritmos de optimización. Estas dos restricciones pueden considerarse como una sola restricción $g_i(x)$, que tomará el valor g_{i1} cuando d_i sea mayor que d_i^{max}, y g_{i2} cuando sea menor que d_i^{min}.

La tensión normal en los elementos se obtiene como cociente entre el esfuerzo a que está sometido el elemento y el área de su sección recta:

$$\sigma_j(x) = N_j(x) \;/\; A_k \qquad (2.8)$$

En donde N_j es el esfuerzo en la barra j, y A_k es el área del grupo al que pertenece el j-esimo elemento. A esta tensión se le imponen límites de la forma:

$$\sigma_j^{min} \leq \sigma_j(x) \leq \sigma_j^{max} \quad j= 1,2,\ldots,N^o \text{ de elementos} \qquad (2.9)$$

Como en el caso anterior, cada una de estas condiciones puede expresarse mediante las restricciones:

$$g_{j1}(x) = 1 - \sigma_j(x) \;/\; \sigma_j^{max} \qquad (2.10)$$

$$g_{j2}(x) = -1 + \sigma_j(x) \;/\; \sigma_j^{min} \qquad (2.11)$$

que también pueden considerarse como una sola restricción g_j, que tomará el valor g_{j1} cuando $\sigma_j(x)$ sea mayor que σ_j^{max} y el valor g_{j2} cuando sea menor que σ_j^{min}.

En el caso de elementos comprimidos, si existe posibilidad de pandeo, las restricciones anteriores pasarán a ser de pandeo. Cuando se emplee la fórmula de Euler para obtener la tensión crítica, la condición de pandeo de un elemento j será:

$$-\sigma_j(x) \leq \sigma_j^{crit}(x) \quad j= 1,2,\ldots,N^o \text{ de elementos} \qquad (2.12)$$

Cuando se emplee el método Omega, la condición será la misma, adoptando como tensión crítica de pandeo:

$$\sigma_j^{crit}(x) = \sigma_j^{max} \;/\; \omega(x) \qquad (2.13)$$

Así pues, la restricción correspondiente al pandeo de un elemento j puede ponerse como:

$$g_j(x) = (\sigma_j^{crit}(x) - \sigma_j(x)) \;/\; \sigma_j^{max} \qquad (2.14)$$

Habiendose introducido σ_j^{max} en el denominador con el objeto de normalizar la restricción respecto a un valor constante.

Debe notarse que el cálculo del coeficiente ω de la expresión (2.13) requiere conocer la inercia de la sección. Por ello, el número de variables del diseño aumentará en el caso de considerar el pandeo. Para evitarlo, se puede poner la inercia como una variable dependiente

del área de la sección. El método para hacer esto se expone a continuación, en el caso de estructuras de barras de nudos rígidos.

Una posibilidad de reducir el tamaño del problema de optimización es el agrupamiento de restricciones, consistente en considerar como única restricción de tensión la correspondiente al elemento más cargado de cada grupo de propiedades. Este agrupamiento de restricciones puede dar problemas en la iteraciones iniciales, debido a que los cambios de las variables suelen ser grandes y pueden variar los elementos mas cargados de cada grupo. El cambio del elemento más cargado de un grupo produce una discontinuidad en la derivada de la restricción, con los problemas de estabilidad y convergencia que esto lleva asociados. En las cercanias del óptimo, como los cambios suelen ser pequeños, no suelen presentarse estos problemas y el agrupamiento produce resultados muy satisfactorios.

Las restricciones prácticas de diseño corresponden a aquellos condicionamientos del diseño no incluidos en el modelo de análisis, tales como los valores máximos y mínimos de las variables de diseño (áreas o coordenadas), igualdad de perfiles, simetrías, etc. Grán parte de estos condicionamientos es posible cumplirlos a través del agrupamiento de variables descrito anteriormente, con lo cual no es necesario considerarlos como restricciones, y se reduce el tamaño del problema. En aquellos casos en que sea necesario considerarlas, pueden ponerse como restricciones de igualdad o de desigualdad. Sin embargo, cuando estas condiciones dan lugar a restricciones constantes o lineales es conveniente formularlas por separado, ya que muchos algoritmos de optimización pueden tratar este tipo de restricciones de una forma más eficiente que las no lineales.

2.6 OPTIMIZACION DE ESTRUCTURAS DE NUDOS RIGIDOS.

En los elementos barra de nudos rígidos sometidos a un estado tensional basado en la flexión, cada sección recta debe describirse por mas de una variable. Esta característica diferencia a las estructuras de barras de nudos rígidos de las de barras de nudos articulados, en las que este problema solo aparecía cuando considerabamos el pandeo en el diseño (en este caso, además del área de la sección recta de un elemento, necesitabamos conocer el momento de inercia de dicha sección).

En las estructuras pequeñas, es posible optimizar simultaneamente todas estas variables, pero conforme la estructura va siendo mayor, el número de variables crece tan rápidamente que hace inabordables los problemas. Por ello, para poder optimizar estructuras de tamaño real debemos recurrir a soluciones del tipo de agrupar estas variables.

2.6.1. Agrupamiento de las variables de diseño.

Hemos visto que uno de los métodos más adecuados para reducir el número de variables de optimización es el agrupamiento de elementos, expresando las propiedades de todos ellos en función de un número menor de propiedades. También hemos visto que esto no reduce el número de variables cuando tenemos que considerar diferentes características de un mismo elemento. Este caso es el habitual cuando se diseñan estructuras de barras de nudos rígidos. Por lo tanto debemos recurrir a otra forma de reducción del número de variables.

Una solución posible consiste en considerar solo una variable por elemento en la optimización de la estructura, seleccionando las otras variables por suboptimización de cada uno de los elementos por separado. En principio, este procedimiento unicamente traslada el problema: se simplifica la optimización de la estructura, pero a costa de añadir optimizaciones de los elementos. No obstante, se han desarrollado métodos de suboptimización que se aprovechan de las características del problema para reducir el esfuerzo de cálculo requerido, llegandose a obtener soluciones operativas.

Otra solución es la apuntada por Haug y Arora /31/; en ella plantean la posibilidad de referir las variables de propiedades de un elemento a un subconjunto de dichas propiedades. La reducción del número de variables de propiedades de una sección transversal se basa en las relaciones que presentan los perfiles correspondientes a una misma gama. La forma operativa consiste en adoptar una de las características de la sección transversal como independiente, y relacionarla con el resto de propiedades mediante expresiones del tipo:

$$D = a\, V^b \qquad (2.15)$$

Siendo D la variable dependiente y V la independiente, y obteniendose los valores de a y b mediante ajustes (p.e. por mínimos cuadrados) de la gama de perfiles a emplear.

En el caso de estructuras de barras de nudos articulados, lo más conveniente es adoptar el área como variable independiente, y obtener el resto a partir de ajustes. Para las estructuras de barras de nudos rígidos, Haug y Arora proponen tomar como variables independientes las inercias, quedando las áreas, los módulos resistentes y los radios de giro como variables dependientes.

2.6.2. Restricciones de diseño.

En cuanto a las restricciones de diseño para las estructuras de barras de nudos rígidos caben las mismas consideraciones hechas para las de nudos articulados. Las restricciones de desplazamiento se generalizan para considerar los giros además de las traslaciones. En cuanto a las restricciones de tensión, la única, e importante diferencia estriba en que el estado tensional de los elementos es más complejo. Los elementos barra de nudos rígidos están, en general, sometidos a tensiones derivadas de la flexión y la torsión, además de las de tracción-compresión. Por ello la expresión de la tensión obtenida en (2.8) no es aplicable en este caso.

La tensión que se debe emplear en las expresiones (2.9) a (2.14) es una 'tensión de comparación' que se obtendrá considerando todos los esfuerzos que actuan sobre el elemento. Utilizando las condiciones de agotamiento de la norma MV-103 obtendremos la tensión de comparación como:

$$\sigma = \sqrt{1/2 \; (\; (\sigma_I-\sigma_{II})^2 + (\sigma_{II}-\sigma_{III})^2 + (\sigma_{III}-\sigma_I)^2 \;)} \qquad (2.16)$$

Siendo σ_I, σ_{II} y σ_{III} las tensiones principales que definen al estado tensional. Esta expresión se suele expresar respecto a un sistema de coordenadas locales cualesquiera como:

$$\sigma = \sqrt{1/2 \; ((\sigma_x-\sigma_y)^2 + (\sigma_y-\sigma_z)^2 + (\sigma_z-\sigma_x)^2 + 6 \; (\tau_{xy}^2 + \tau_{xz}^2 + \tau_{yz}^2))} \qquad (2.17)$$

que queda convertida en:

$$\sigma = \sqrt{\sigma_x^2 + \sigma_y^2 - \sigma_x\sigma_y + 3 \; \tau_{xy}^2} \qquad (2.18)$$

cuando el estado tensional es plano.

2.7 OPTIMIZACION DE ELEMENTOS ESTRUCTURALES.

Hemos comentado anteriormente que la optimización de una estructura incluyendo mas de una variable de diseño por elemento, lleva a problemas inviables, aún considerando estructuras de pequeño tamaño. También hemos descrito formas de condensación de dichas variables que permiten resolver problemas prácticos. No obstante, cuando el problema no puede resolverse con elementos estandarizados (perfiles laminados, etc.), las técnicas de reducción de variables descritas no tienen validez. Entonces no hay mas remedio que considerar diferentes variables para un mismo elemento. También se ha apuntado que una posible solución en este caso es la de hacer una suboptimización de cada elemento.

Para la optimización de elementos estructurales por separado se necesita conocer las solicitaciones que actuan sobre el elemento, y las condiciones de contorno del mismo. Estas pueden venir fijadas arbitrariamente (p.e. considerando unos valores máximos), o pueden ser el resultado de un proceso de optimización de la estructura global. En cualquier caso, una vez determinadas las solicitaciones y las condiciones de contorno, es normalmente fácil realizar el cálculo de esfuerzos sin ayuda de complejos programas de análisis.

El mayor problema en la formulación de la optimización de elementos proviene de la imprecisión de la definición de las características y limitaciones que las normas imponen para el diseño de los elementos. Las dificultades provienen de las discontinuidades en las expresiones de dichas normas. Estas discontinuidades derivan de que dichas normas fueron desarrolladas para una comprobación manual en lugar de una comprobación automática (de ahí el abundante empleo de tablas en lugar de expresiones matemáticas), y se consideraban solamente las posibilidades mas habituales.

Una vez salvadas las dificultades de formulación del problema, este queda convertido en un problema con un número relativamente pequeño de variables (no suele pasar de 10), y unas restricciones que pueden tener expresiones muy complejas, pero que son generalmente explícitas. Por ello, el problema puede resolverse perfectamente con cualquiera de los algoritmos de optimización existentes.

CAPITULO 3

PLANTEAMIENTO Y RESOLUCION DEL PROBLEMA DE OPTIMIZACION

3.1. INTRODUCCION.

En este capítulo se trata de la formulación y resolución del problema de optimización descrito en el capítulo precedente. El capítulo incluye una somera descripción de los aspectos más significativos de los diferentes métodos de resolución desarrollados a traves del tiempo. La finalidad de describir esta evolución es la de introducir, de la forma más natural posible, la situación actual como base para describir las características del método de optimización empleado en el presente trabajo. Se pretende, así mismo, que al dar una visión más amplia, las conclusiones que se deriven de este trabajo sean más fácilmente trasladables a sistemas similares basados en otros métodos de optimización.

Las descripciones de los diferentes métodos se han tomado de distintas fuentes; aunque la información básica de todos ellos se puede encontrar en las referencias /20,25,27 y 70/. Los métodos no son descritos con rigor matemático. Por el contrario, se da una descripción intuitiva de los mismos, basada en consideraciones geométricas, pues lo que se persigue es una introducción a los aspectos gráficos de los mismos.

La primera parte del capítulo está dedicada al planteamiento del problema, introduciendose la formulación matemática del mismo y las condiciones que se cumplen en el óptimo. En la segunda parte del capítulo se describen los métodos de resolución. La última parte está dedicada a una descripción más detallada del método de programación cuadrática sucesiva, que es el empleado en este trabajo.

3.2. FORMULACION DEL PROBLEMA DE OPTIMIZACION.

3.2.1 Formulación básica.

La formulación algebraica más comunmente empleada para expresar el problema general de optimización es la de considerar que se trata de:

encontrar un conjunto de variables de diseño x
tales que

minimicen $f(x)$

$$\text{sujetas a } h_j(x) = 0 \qquad j= 1,\ldots,m_i$$

$$g_k(x) \geq 0 \qquad k= 1,\ldots,m_d \qquad (3.1)$$

$$x_i^I \leq x_i \leq x_i^S \qquad i= 1,\ldots,n$$

Siendo:

x un vector n-dimensional que contiene a las variables de diseño;

f la función objetivo;

h_j cada una de las m_i funciones de las variables que expresan las restricciones de igualdad;

g_j cada una de las m_d funciones de las variables que expresan las restricciones de desigualdad;

x_i^I los límites inferiores de las variables, y

x_i^S los límites superiores de las variables.

Como ya se ha comentado, las restricciones de igualdad se han formulado por separado porque aparecen como tales en ciertos tipos de optimización. No obstante estas restricciones pueden convertirse en parejas de restricciones de desigualdad. También se ha comentado que la distinción que se ha hecho entre restricciones de desigualdad en general y las restricciones de las variables, se debe a que estas últimas son un caso muy particular de restricciones de desigualdad que en muchos métodos pueden tratarse eficientemente por separado. Por tanto, expresando estos dos tipos de restricciones como resticciones generales de desigualdad, podemos obtener una formulación resumida del problema (3.1) de la forma:

encontrar un conjunto de variables de diseño x
tales que

minimicen $f(x)$ (3.2)

sujetas a $g_k(x) \geq 0$ $k= 1,...,m$

En donde m será el número de restricciones de desigualdad (m_d) aumentado en las restricciones de desigualdad que sustituyen al resto de restricciones (m_i y n).

Se debe notar que tampoco es única la forma de considerar las desigualdades. En función del método de optimización empleado puede ser más conveniente expresar las restricciones como positivas o como negativas. Sin embargo cualquier formulación puede convertirse en la otra sin más que cambiar el signo de las restricciones.

Cuando la función objetivo y todas las restricciones son funciones lineales de las variables, se dice que el problema (3.2) es 'lineal'. En cualquier otro caso se dice que el problema es 'no lineal'. Esta distinción es importante porque existen métodos de resolución del problema que solamente sirven para problemas lineales.

Otro caso particular del problema (3.2) es cuando no existen restricciones. El problema sin restricciones es un caso muy importante, no solo porque existen técnicas muy efectivas de tratarlo, sino porque muchas técnicas para resolver el problema general se basan en la solución de problemas equivalentes sin restricciones.

3.2.2 Condiciones de optimalidad.

Cualquier conjunto de condiciones que deban cumplirse en el óptimo se denominan condiciones de optimalidad. Se han propuesto diferentes conjuntos de condiciones, la mayoría de los cuales están estudiados en la bibliografía que plantea una aproximación matemáticamente más rigurosa sobre el tema (p.e. en Bazaraa /7/). La importancia de identificar las condiciones de optimalidad se deriva del hecho de que la mayoría de métodos de búsqueda de soluciones óptimas son iterativos, y por tanto se necesitan criterios para saber cuando dichos métodos han alcanzado la solución buscada.

La condición necesaria y suficiente para que un punto sea la solución de (3.2) es que dicho punto pertenezca a la región factible y que la función objetivo tenga valor mínimo en él. Es decir:

$$x^* \text{ mínimo } \langle=\rangle \begin{cases} x^* \in R_f \\ \exists\, \epsilon > 0 \;/\; f(x) \geq f(x^*) \;\; \forall x \in B_{x^*} \end{cases} \tag{3.3}$$

Siendo:

$$R_f = (x \;/\; g_k(x) \geq 0 \quad k = 1,\ldots,m)$$

$$B_{x^*} = (x \;/\; |x^*-x| \leq \epsilon)$$

En general, el punto será un mínimo local de la función. Solo en el caso de que además la función f y la región factible R_f sean convexas /7/, el mínimo será global. En este caso x^* será la solución al problema (3.2). Habitualmente no se cumplen estas condiciones, por lo que el mínimo solo será local. No obstante, muchos métodos de búsqueda del óptimo llegan al mínimo global a partir de los mínimos locales. Por ello, las condiciones (3.3) son válidas como criterio de optimalidad.

Sin embargo, el problema estriba en obtener unas condiciones de optimalidad susceptibles de implementarse en los algoritmos de optimización, de forma que sirvan para detectar que el mínimo ha sido encontrado. Por tanto, es lógico esperar que estas condiciones utilicen la información habitualmente manipulada por los algoritmos de optimización: valores de f, g_k, y sus derivadas en el punto de diseño. Es decir, que se busca un conjunto de condiciones de optimalidad que: a) sean necesarias y suficientes; b) utilicen únicamente información de las funciones y sus derivadas en el punto de diseño, y c) sean implementables. No obstante, se sabe que, en ausencia de convexidad, no existe dicho conjunto de condiciones. Por lo tanto, se usan condiciones necesarias (e implementables), que a efectos prácticos puedan considerarse suficientes.

Belegundu /10/, resume las condiciones de optimalidad más comunmente empleadas. La formulación general de las más extendidas de dichas condiciones es la que se conoce como 'condiciones de Kuhn-Tucker'. Dichas condiciones se deducen de la relación establecida por Kuhn y Tucker /41/ de que el punto óptimo del problema (3.2) es también un punto estacionario para la función lagrangiana. Dicha función lagrangiana se define, para el problema (3.2), como:

$$L(x,u) = f(x) - \sum_{j=1}^{m} u_j \, g_j(x) \tag{3.4}$$

En donde los parámetros u son los denominados multiplicadores de lagrange. También son conocidos como las variables duales debido a que en el óptimo, el lagrangiano debe ser mínimo respecto a x y máximo respecto a u. La condición de estacionareidad lleva a las siguientes condiciones:

$$\begin{aligned} &\nabla L(x,u) = 0 \\ &g_j(x) \geq 0 \qquad j= 1,2,\ldots,m \\ &u_j \geq 0 \qquad j= 1,2,\ldots,m \\ &g_j(x)\, u_j = 0 \qquad j= 1,2,\ldots,m \end{aligned} \tag{3.5}$$

Estas condiciones necesarias, son también suficientes si se puede demostrar que la función f es convexa y las funciones g_j son cóncavas (en /10/ se demuestra que basta con que las restricciones sean cóncavas y el lagrangiano convexo). Esto es debido a que para que la región factible sea convexa debe cumplirse que las restricciones sean cóncavas, puesto que entonces se cumple:

$$\left.\begin{array}{l} g_k \text{ cóncava} \\ \forall\, \alpha \in R \end{array}\right\} \Rightarrow R_{fK} = (x \;/\; g_k(x) \geq \alpha) \text{ convexo} \tag{3.6}$$

y, además:

$$R_{f1},\ R_{f2},\ldots,R_{fm} \text{ convexos} \Rightarrow R_f = R_{f1} \cap R_{f2} \cap \ldots \cap R_{fm} \text{ convexo} \tag{3.7}$$

3.2.3 Interpretación geométrica.

Una interpretación geométrica del problema que se desprende de su definición (3.2), es la de considerar que consiste en encontrar un punto (en un espacio n-dimensional) para el cual el valor de una función (n-dimensional) sea el mínimo posible, cumpliendo ciertas condiciones. En el caso más sencillo de una sola variable, la función objetivo es una curva, para la que hay que buscar el valor mínimo (fig. 3.1) dentro de un segmento del eje que representa los posibles valores de la variable.

En la figura puede verse como las restricciones dividen en dos partes a los posibles valores de la variable. El punto que marca la división es el de valor de la variable que convierte a la restricción en una igualdad. Queda entonces una parte de valores 'factibles' de la variable, y otra parte de valores no factibles.

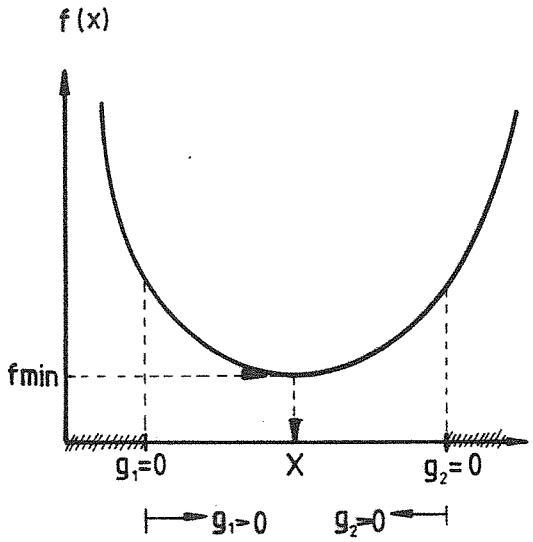

Figura 3.1. Representación geométrica del proceso de optimización para una sola variable.

En el caso más general de n variables, el espacio n-dimensional formado por todas ellas se denomina 'espacio de diseño'. Entonces, las restricciones son superficies n-dimensionales que en conjunto definen un volumen contenido en el espacio de diseño denominado 'región factible'. En la figura (3.2) se han representado la región factible en un caso con dos variables.

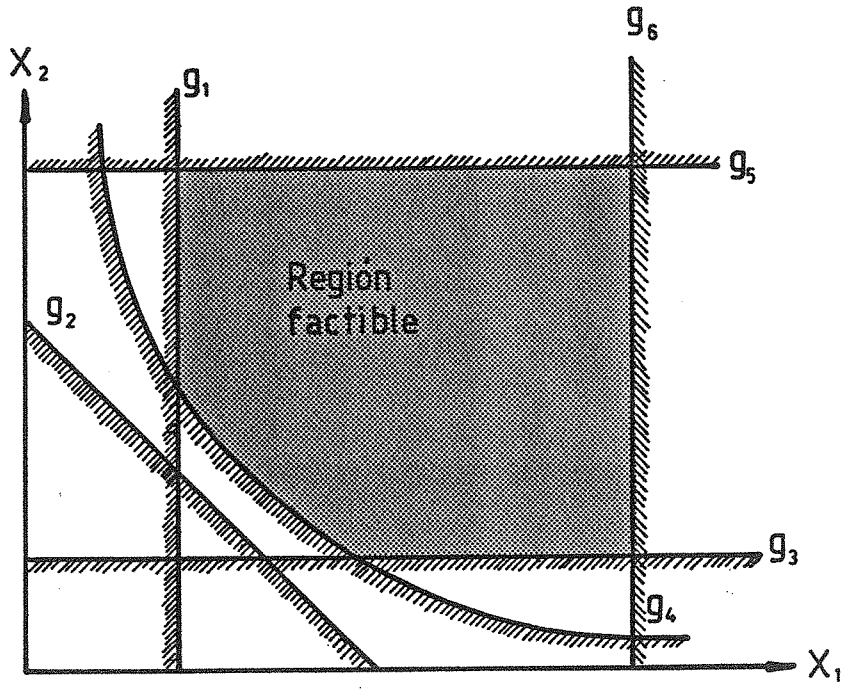

Figura 3.2. Región factible.

Los puntos de la región factible se denominan puntos 'interiores'; mientras que el resto se denominan 'exteriores'. Cuando un punto es exterior a una restricción se dice de esta que está 'violada'. En el caso particular de que el punto esté sobre la superficie de valores de las variables que convierten a una restricción en igualdad, se dice de la restricción que está 'activa'. Los puntos definidos por la intersección de dos o más superficies límites de las restricciones se denominan 'vértices'. En esta figura puede observarse como la condición impuesta por la restricción 2 es más suave que la debida al resto de restricciones. Es decir, que la restricción 2 no forma parte de la frontera de la región factible. Intuitivamente es fácil comprender que este tipo de restricciones no intervienen en la determinación de la solución óptima. Por tanto, cuando exista una restricción de este tipo, puede ser eliminada. De esta forma se reduce la complicación del problema, sin que esto suponga ningún tipo de simplificación.

En la figura (3.3) se ha dibujado el espacio de diseño para un problema con dos variables y tres restricciones. En el se observa que las restricciones definen dos vértices (V_1 y V_2). El primero de ellos es el punto de la región factible en donde la función objetivo toma el mínimo valor posible. Por tanto V_1 es el punto óptimo de diseño.

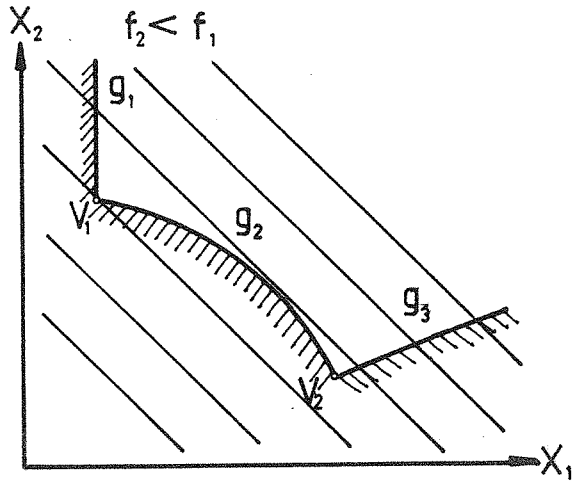

Figura 3.3. Mínimos locales.

Este ejemplo ilustra, además, uno de los principales problemas, ya comentado, de muchos de los métodos actuales de optimización: el de los mínimos locales. En efecto, el vértice V_2 es un punto del espacio de diseño tal que ningún punto de su entorno conduce a un diseño mejor, por lo tanto el punto es un mínimo. Sin embargo, el punto V_1 es una solución mejor. Por lo tanto este es el mínimo absoluto. Debido al proceso de búsqueda de la solución, muchos métodos son incapaces de encontrar siempre el mínimo absoluto. Obteniendo un mínimo relativo, que dependerá del punto tomado como diseño inicial.

La interpretación geomética de las condiciones de optimalidad es la de que una condición necesaria para que un punto sea un mínimo es que no haya en su entorno ningún otro punto (factible) con un valor menor de la función objetivo (fig. 3.4).

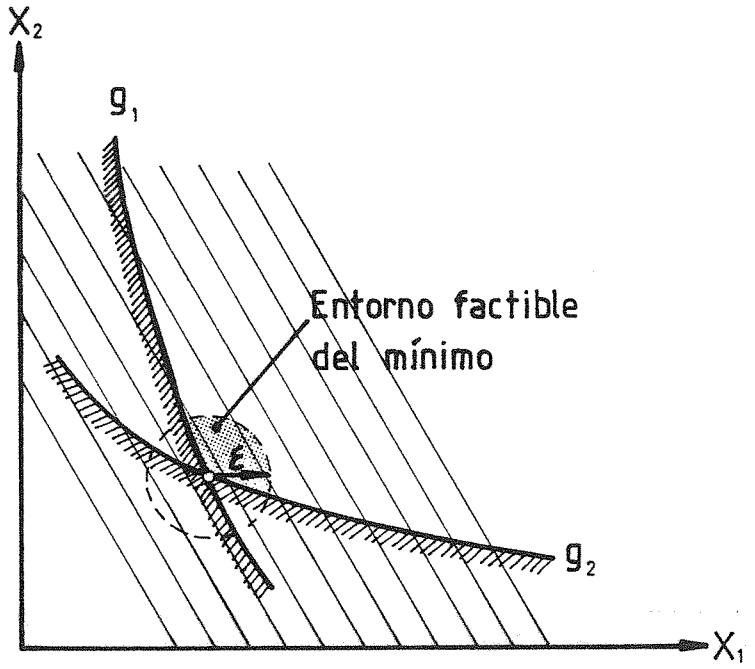

Figura 3.4. Condición de optimalidad.

Las condiciones de optimalidad de Kuhn-Tucker tienen una interpretación geométrica sencilla que puede observarse planteando la condición de que el gradiente del lagrangiano debe ser cero en el óptimo:

$$\nabla L(x,u) = 0 \qquad (3.8)$$

que se puede poner como:

$$\nabla f(x) = \sum_{k=1}^{m} u_k \nabla g_k(x) \qquad (3.9)$$

De la expresión (3.9) se observa que los negativos de los vectores gradiente de las restricciones activas (para las no activas será $u_k=0$), definen un "cono". De tal forma que el negativo del gradiente de la función objetivo debe ser interior a dicho cono para que el punto sea un mínimo, tal como se observa en la figura (3.5).

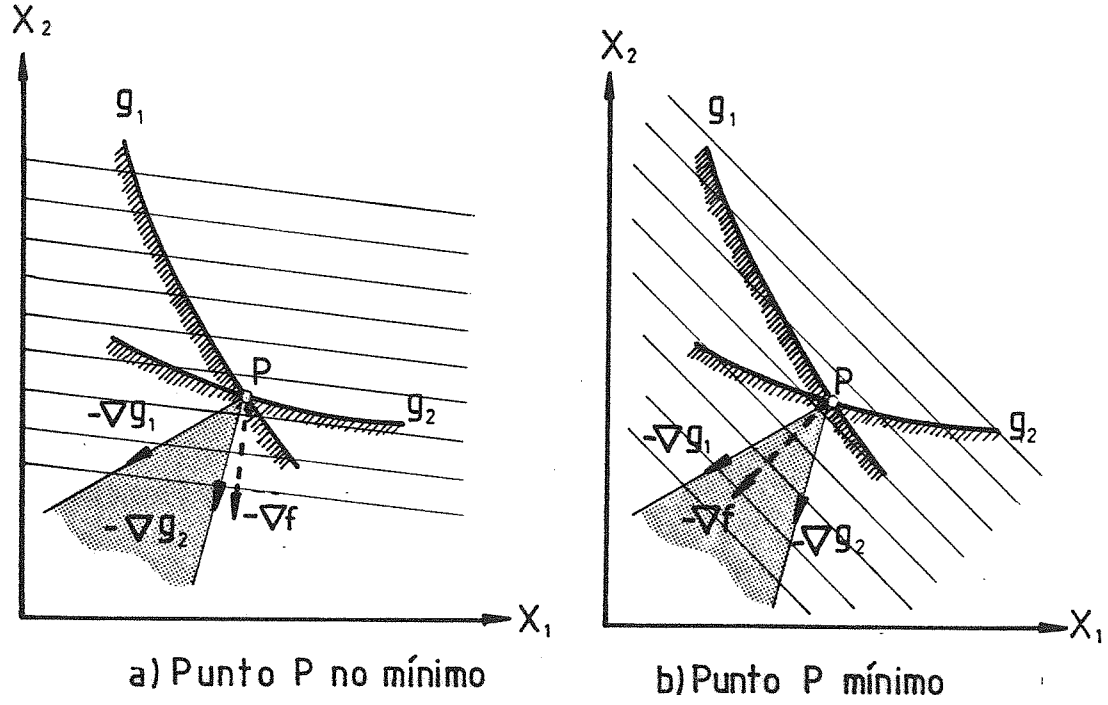

Figura 3.5. Condiciones de Kuhn-Tucker.

3.3. RESOLUCION DEL PROBLEMA DE OPTIMIZACION.

Esencialmente, se distinguen dos aproximaciones para la resolución del problema de optimización formulado en (3.2): a) los criterios de optimalidad, y b) la programación matemática. La primera aproximación es la formada por los métodos llamados 'indirectos'. Esto es debido a que lo que se busca en dichos métodos es obtener un diseño que satisfaga un criterio especificado, que, a su vez, implique el cumplimiento del objetivo buscado. El criterio puede ser intuitivo o deducido matemáticamente, a partir de las características particulares del problema a tratar. La segunda aproximación es más general; en

lugar de basarse en aspectos 'fisicos' del problema, trata de llegar a una formulación matemática del mismo, intentando que dicha formulación matemática sea general y susceptible de implementación.

La coexistencia de ambas tendencias se debe a que los resultados obtenidos con los criterios de optimalidad dependen del problema y no pueden ser generalizados; por lo cual se requiere un método para cada tipo de problema, y un mínimo de experiencia para elegir el método apropiado. Por el contrario, los métodos desarrollados hasta la fecha a partir de la formulación matemática son capaces de resolver problemas generales, pero resultan menos eficientes que los equivalentes basados en criterios de optimalidad para cada caso en concreto.

En la actualidad, autores como Fleury /22/ han llegado a comprobar que ambas aproximaciones se basan en los mismos principios. No obstante, el problema considerado para plantear esta unificación es el de diseño de propiedades de la sección de los elementos, por lo que el problema general de diseño aún está lejos de alcanzar la unificación. De ahí que en la descripción de los métodos existentes que vamos a hacer a continuación mantendremos la división entre ambas aproximaciones.

3.3.1. Criterios de optimalidad.

Los criterios de optimalidad son formas de solucionar problemas concretos de optimización, dirigiendo la solución por medio de la aplicación de criterios que se saben (o creen) apropiados para el problema tratado. Algunos de los criterios de optimalidad tienen un claro sentido físico; tal es el caso del diseño F.S.D. ("Fully Stressed Design"), que es aquel en el cual cada elemento de la estructura soporta una tensión límite bajo, al menos, uno de los estados de carga especificados. El F.S.D. es uno de los conceptos tradicionales en el diseño óptimo de estructuras, pero mantiene su interés por su caracter intuitivo y por ser un procedimiento rápido y fácil de implementar en comparación con los basados en programación matemática.

A continuación vamos a exponer brevemente las características y bases analíticas del F.S.D., porque el método puede tener utilidad como parte de un proceso de diseño basado en métodos de programación matemática. Una referencia más detallada del método se puede encontrar en la bibliografía sobre el tema, especialmente en los trabajos de

Gellatly y Berke /27/. También se pueden encontrar en /45,47 y 51/ las aplicaciones de este método en el sistema DISSENY.

3.3.1.1 Características del F.S.D.

El F.S.D. es un diseño a resistencia, de propiedades de los elementos. Inicialmente, el método no considera otro tipo de restricciones, si bién se puede generalizar para tratar restricciones de desplazamiento. La característica más destacada en F.S.D. es la ausencia de función objetivo. Por tanto, no existe una cantidad a minimizar o maximizar. No se puede asegurar que un algoritmo para calcular F.S.D. converja al diseño de mínimo peso. Esto se debe a que esta meta no se puede explicitar al no haber función objetivo.

Hace mucho tiempo que Mitchell demostró que para una estructura de barras de nudos articulados sometida a un único estado de cargas, la solución de mínima relación peso/resistencia es la dada por un diseño F.S.D /56/. Sin embargo, esta condición no es generalizable para más estados de carga. Esto se debe a que un diseño F.S.D. no es único. Dado que cada estructura estáticamente determinada puede ser proporcionada para alcanzar un F.S.D., si se da valor inicial nulo a cada elemento de una estructura estáticamente indeterminada, cada forma subsidiaria de la misma que sea estáticamente determinada da lugar a un F.S.D. alternativo /5/. Por tanto, un F.S.D. puede conducir a una solución optima en ciertos casos; pero también puede llevar a una solución más pesada que el óptimo, o incluso puede no encontrarse solución.

En general, la experiencia ha demostrado que, salvo en los casos de estructuras con comportamientos extraños, el F.S.D. da un diseño a tensión óptimo o muy cercano /27/. La posibilidad de que la solución obtenida no sea la óptima queda compensada por la facilidad de implementación del método y la rapidez con que converge; aunque el número de iteraciones requeridas para alcanzar la solución está controlado por características del problema que no están muy bién determinadas. No obstante, si se sabe que, en general, el número de iteraciones no esta ligado al número de variables, siendo esta la característica que hace más apropiado este método de diseño para problemas con grán cantidad de elementos.

Otra característica interesante del método es que algunas de sus aproximaciones dan una solución cercana al óptimo en la primera o

segunda iteración independientemente del punto de partida; por lo cual es un buen método para obtener rápidamente un diseño de partida apropiado para arrancar otros métodos más sofisticados.

3.3.1.2 Bases analíticas

No hay un único método de cálculo de un F.S.D. La aproximación más simple consiste en un procedimiento de análisis iterativos en el cual el resultado de una iteración se usa para escalar los elementos para que trabajen a tensión máxima. Los elementos escalados se emplean en la siguiente iteración del análisis. Este proceso se puede expresar en forma algebraica:

$$x_i^{k+1} = x_i^k \, (\sigma_i^k / \sigma_i) \qquad (3.10)$$

siendo

x_i^{k+1} la variable del i-esimo elemento en la iteración k+1;

x_i^k la variable del i-esimo elemento en la iteración k;

σ_i^k la tensión del i-esimo elemento en la iteración k, y

σ_i la tensión de cálculo del i-esimo elemento.

Esta aproximación es la conocida como 'método Stress-Ratio', o aproximación de orden cero (debido a que no utiliza información de derivadas). El proceso iterativo se mantiene hasta que la variación de las áreas entre dos iteraciones consecutivas sea tan pequeña como interese.

Aceptando que un aumento en el tamaño del elemento hace que absorba más carga, y que un decremento hace que quede más descargado, se puede acelerar el proceso de convergencia de este método por medio de un proceso de 'sobrerelajación'. Para ello se plantea:

$$x_i^{k+1} = x_i^k \, (\sigma_i^k / \sigma_i)^{\beta} \qquad (3.11)$$

en donde β es el factor de relajación, mayor que la unidad, que debe ser ajustado (un valor habitual es 1.2).

El método Stress-Ratio solo es estrictamente riguroso cuando la estructura es isostática; porque entonces, la tensión de un elemento no se vé influenciada por la de los demás. En este caso, la solución

óptima se alcanza en la primera iteración. En cualquier otro caso, para utilizar una aproximación más depurada, se debe tener en cuenta la influencia de todos los elementos sobre la tensión de cada uno de ellos.

La tensión de un elemento aislado sometido a tracción/compresión se puede expresar como:

$$\sigma_i = F_i \; / \; x_i \qquad (3.12)$$

siendo

F_i la fuerza a que está sometido el elemento, y

x_i el área de la sección recta de dicho elemento.

Tomando como variables de diseño las áreas de los elementos, y derivando esta expresión respecto a ellas, se obtiene:

$$\delta\sigma_i/\delta x_j = -F_i/x_i^2 + 1/x_i \; \delta F_i/\delta x_i \qquad (3.13)$$

y sustituyendo el valor de F_i de (3.3):

$$\delta\sigma_i/\delta x_j = -\sigma_i/x_i + 1/x_i \; \delta F_i/\delta x_i \qquad (3.14)$$

Un cambio finito de la tensión de uno de los elementos en una estructura con n elementos, se puede expresar como:

$$\sigma_i - \sigma_i^o = \sum_{j=1}^{n} \delta\sigma_i/\delta x_j \; (x_j - x_j^o) \qquad (3.15)$$

Sustituyendo (3.14) en (3.15), para un cambio desde el k-esimo punto de diseño hasta el siguiente (k+1), se obtiene:

$$\sigma_i^{k+1} - \sigma_i^k = -(\sigma/x)_i^k \; (x_i^{k+1}-x_i^k) + (1/x)_i^k \sum_{j=1}^{n} (\delta F_i/\delta x_j)^k \; (x_j^{k+1}-x_j^k) \qquad (3.16)$$

Ecuación que únicamente es válida cuando los dos puntos de diseño (k y k+1) están en un entorno lo suficientemente pequeño.

Para eliminar una incógnita, vamos a sustituir el valor de la tensión en el punto k+1 por el valor de cálculo de dicha tensión. Con ello, si el diseño inicial es malo, en las primeras iteraciones σ_i^k no estará en el entorno de σ_i, por lo que el método divergerá.

Haciendo el cambio, la ecuación (3.16) se puede expresar en forma matricial como:

$$\bar{\sigma} - \sigma^k = \left[G(\sigma) + H(F) \right] (x^{k+1} - x^k) \qquad (3.17)$$

siendo

$$G(\sigma) = \left[\sigma_i / x_i \right] \qquad i = 1,2,\ldots,n \text{ (matriz diagonal)} \qquad (3.18)$$

$$H(F) = \left[\delta F_i / \delta x_i \right]^k \qquad i = 1,2,\ldots,n \qquad (3.19)$$

La evaluación de la ecuación (3.17) en el punto de diseño x^k nos permite obtener una estimación de un punto x^{k+1} que mejora el diseño, según el criterio F.S.D.

No obstante, la evaluación de la matriz (3.19) es difícil, por lo que es preferible plantear la diferenciación directa de las tensiones respecto a las variables de diseño. Así, la ecuación (3.15) se puede expresar en forma matricial como:

$$\bar{\sigma} - \sigma^k = H(\sigma) (x^{k+1} - x^k) \qquad (3.20)$$

siendo

$$H(\sigma) = \left[\delta \sigma_i / \delta x_i \right]^k \qquad i = 1,2,\ldots,n \qquad (3.21)$$

La expresión (3.20), la podemos poner, finalmente, como:

$$x^{k+1} = x^k + H(\sigma)^{-1} (\bar{\sigma} - \sigma^k) \qquad (3.22)$$

Los términos de la matriz $H(\sigma)$ no son más fáciles de evaluar que los de la matriz $H(F)$; pero, a cambio, son los únicos coeficientes a invertir en la expresión (3.22).

Las expresiones (3.22) y (3.21) constituyen la aplicación del método Mewton, o aproximación de primer orden (puesto que utiliza información de las primeras derivadas).

El método Newton converge más rápidamente que el Stress-Ratio. Por el contrario, el Stress-Ratio no tiene los problemas de divergencia del Mewton, cuando el punto inicial es malo. Por ello, es bastante

habitual emplear un método híbrido en el que las primeras iteraciones se hacen por Stress-Ratio y luego se llega a la convergencia final por Newton.

3.3.2 Programación matemática.

La programación matemática obtiene la solución al problema de diseño aplicando métodos numéricos de minimización (o maximización) de funciones objetivo sujetas a restricciones. Como los métodos propuestos hasta la fecha no son capaces de resolver eficientemente un problema como el descrito (cuando todos los condicionamientos son críticos), la solución habitual es la de emplear varios métodos, que se adapten mejor a diferentes tipos de problemas, y elegir en cada caso el más apropiado. Las características del problema de optimización de estructuras son las que condicionan el método numérico más apropiado para obtener la solución. Estas características son:

- Se trata de un problema n-dimensional (pudiendo ser n muy grande);
- La función objetivo no es lineal generalmente;
- Las restricciones no son lineales generalmente, y
- Las variables no pueden tomar cualquier valor (existen límites de las variables).

La mayoría de los métodos numéricos para resolver el problema de diseño óptimo de estructuras, se pueden definir por el mismo planteamiento general de buscar la solución por aproximaciones sucesivas (iterativamente). Cada solución se obtiene a partir de la anterior de la forma:

$$x^{k+1} = x^k + \alpha^k \, d^k \qquad (3.23)$$

es decir, que el punto de diseño (vector de variables x) para la iteración k+1 se obtiene a partir del punto de diseño anterior, moviendose según una 'dirección de búsqueda' (d^k), durante una cierta 'longitud' α^k.

Vamos a estudiar primeramente las diferentes estrategias que hay para decidir la longitud a recorrer. Esta estrategia se conoce como 'búsqueda unidimensional', porque coincide con la búsqueda de la solución óptima para un problema con una sola variable. A continuación

trataremos el problema de determinar la dirección de búsqueda en el caso n-dimensional. Haremos inicialmente la simplificación de considerar que no hay restricciones, para modificar luego las estrategias desarrolladas de forma que las tengan en cuenta.

3.3.2.1 Búsqueda unidimensional.

Pretendemos hallar el mínimo de una función de una sola variable en un intervalo de variación de dicha variable. Es decir, el valor que debe tomar la variable para que la función tome el mínimo valor posible. La dificultad del problema estriba en que no se conoce la expresión analítica de la función; o si se conoce, no es generalizable la deducción analítica del mínimo a partir de ella.

Los métodos utilizados para encontrar el mínimo se dividen en dos grandes grupos:

- los que calculan el valor de la función en diferentes puntos para encontrar el mínimo, y

- los que ajustan la función a una función de tipo conocido, y obtiene analíticamente el mínimo de esta función /21/.

En general, todos los métodos hacen la suposición de que la función es unimodal (tiene un solo mínimo). Cuando la función no cumple esta condición, se debe repetir el proceso elegido para varios puntos iniciales; de forma que se obtengan varios mínimos locales. El menor de estos mínimos se toma como mínimo global. El proceso está representado en la figura (3.6).

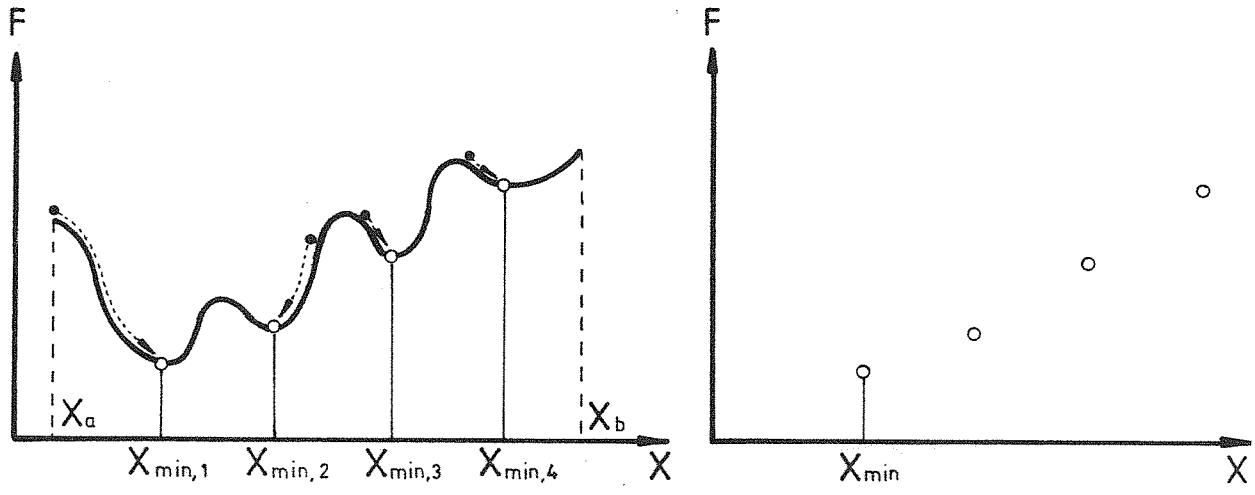

Figura 3.6. Búsqueda del mínimo de una función.

Entre los métodos de búsqueda que calculan valores de la función para encontrar el mínimo se pueden distinguir varios con orden creciente de complejidad y de eficiencia. Evidentemente, una búsqueda exaustiva es muy costosa, por lo que la solución habitual es la de aislar el mínimo en un intervalo e ir decreciendo dicho intervalo tanto como sea necesario para alcanzar la precisión requerida. Un posible criterio para decidir que parte es la peor se puede deducir de la suposición de que la función es una curva unimodal en el intervalo considerado.La eficiencia de estos métodos depende totalmente del criterio seguido para seleccionar los puntos de muestreo. Un método representativo de este tipo es el conocido como 'Golden Search'. En esta aproximación se calculan inicialmente dos puntos desigualmente espaciados (fig. 3.7). A partir de ellos se determina el subintervalo en el que está el mínimo. El proceso se repite pero calculando únicamente un punto cada vez, y aprovechando el punto no desechado en el paso anterior.

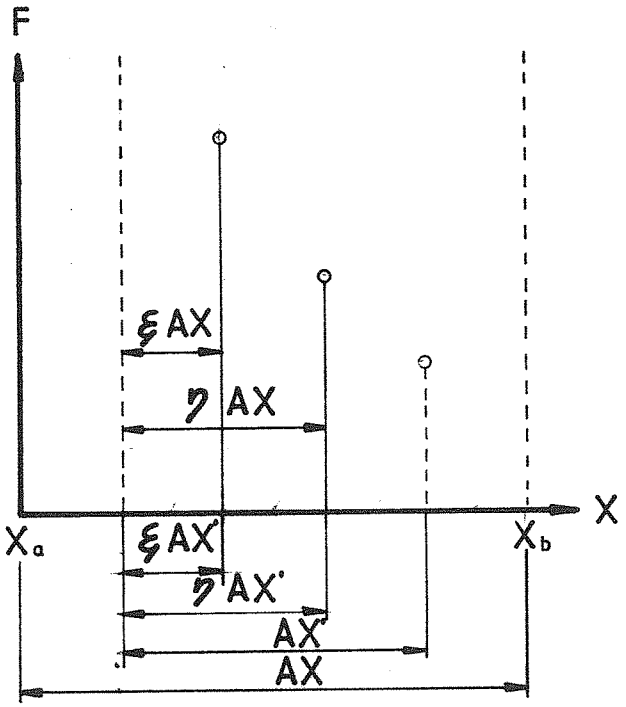

Figura 3.7. Golden Search.

En general, para hacer ajustes que nos permitan encontrar el mínimo empleando únicamente la información de valores de la función, debemos conocer tantos puntos (mas uno) de la misma como el orden del que sea

la función. De esta forma podemos ajustar la curva a un polinomio. En el caso de una curva cuadrática, será:

$$q(x) = a_0 + a_1 x + a_2 x^2 \qquad (3.24)$$

planteando las condiciones:

$$q(x_i) = F(x_i) \qquad i= 1,2,3 \qquad (3.25)$$

Una vez conocida q(x), basta con igualar a cero su primera derivada para obtener la estimación del mínimo.

Si se emplean polinomios de menor grado que la curva real, la estimación será solo aproximada, por lo que habrá que establecer un ciclo iterativo con los sucesivos mínimos estimados.

Si se dispone de información de primeras derivadas, se puede emplear conjuntamente con un método de busqueda para decidir la parte a eliminar, en función de la inclinación de las tangentes a la curva en los puntos considerados (fig. 3.8).

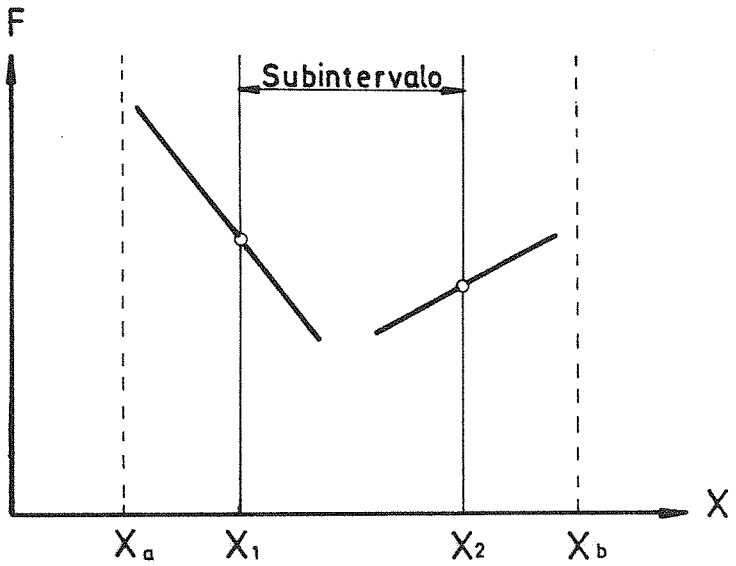

Figura 3.8. Estimación del subintervalo en que está el mínimo a partir de las tangentes.

Una forma más precisa de emplear la información de derivadas es por medio de un desarrollo en serie truncado en las derivadas de las que se tiene información. Para ello, debemos disponer de, al menos, las dos primeras derivadas. Entonces:

$$q(x) = F(x_1) + F'(x_1)\,(x-x_1) + 1/2\,F''(x_1)\,(x-x_1)^2 \qquad (3.26)$$

siendo:

$F(x_1)$ valor de la función en un punto x_1

$F'(x_1)$ valor de la primera derivada de la función en un punto x_1

$F''(x_1)$ valor de la segunda derivada de la función en un punto x_1

De aquí se puede estimar analíticamente un mínimo (x_2) de la función $q(x)$:

$$0 = q'(x_2) = F'(x_1) + F''(x_1)\ (x_2-x_1) \qquad (3.27)$$

de donde:

$$x_2 = x_1 - F'(x_1)/F''(x_1) \qquad (3.28)$$

Como, en general, la función en estudio dependerá de más términos del desarrollo, el mínimo encontrado no será una buena aproximación. Por ello, se toma este mínimo como nuevo punto de referencia y se repite el proceso iterativamente. Este método, denominado Mewton-Raphson, está representado en la figura (3.9).

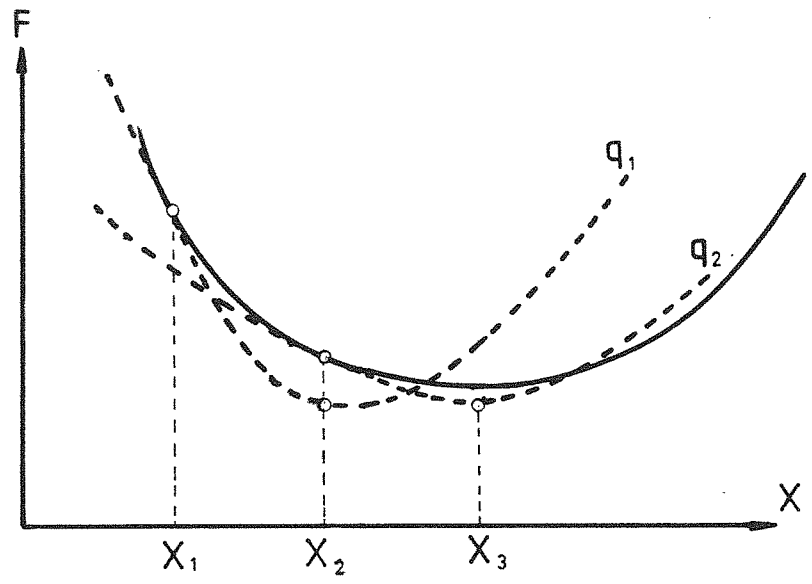

Figura 3.9. Ajuste por Newton-Raphson.

Las primeras derivadas ya es habitual calcularlas como parte de la información necesaria para el diseño óptimo. El gran inconveniente de

este método es la necesidad de conocer las derivadas de segundo orden. Dado que no están disponibles generalmente, y deben calcularse derivando las primeras derivadas.

Por último, existe una nueva tendencia en la búsqueda unidimensional que consiste en buscar una solución que simplemente mejore a la anterior, sin exigir que sea la que minimice dicha solución. El planteamiento de este tipo de búsquedas es el de simplificar los pasos intermedios (aunque no sean los mejores), a fin de agilizar el proceso global de convergencia (aumentar la eficiencia global disminuyendo las eficiencias parciales). El método de Armijo es representativo de este tipo de búsquedas.

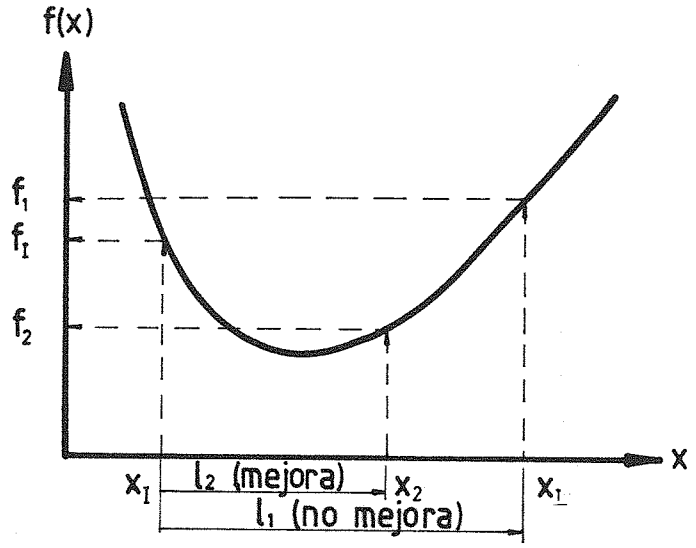

Figura 3.10. Busqueda de un diseño mejor.

3.3.2.2 Dirección de búsqueda.

Encontrar una dirección de búsqueda consiste en elegir una dirección en el espacio de diseño; de forma que partiendo del punto de diseño actual, según esa dirección, se puede hacer una búsqueda unidimensional que conduzca a un nuevo punto de diseño. A continuación vamos a describir varios métodos para seleccionar la dirección de búsqueda, distinguiendolos en función del tipo de información que usan para encontrar la solución (Vanderplaats /70/).

Todos los métodos que buscan la dirección de búsqueda utilizando únicamente información del valor de la función objetivo se conocen genéricamente como métodos de orden cero. El más sencillo de estos métodos se denomina 'busqueda aleatoria', y consiste en elegir una

dirección arbitraria tal que el valor de la función sea menor que el valor en el punto de partida. Un método más eficiente se puede derivar del anterior sin más que imponer algún criterio objetivo para determinar la dirección de búsqueda. La forma más elemental para hacer esto consiste en no cambiar de dirección mientras no nos conduzca a un punto peor que el último. La siguiente mejora consiste en utilizar una dirección que en la mayoría de los casos nos lleve a un punto mejor. Se puede definir el algoritmo de búsqueda en cuatro pasos:

1- Tomar una dirección arbitraria.

2- Hacer una búsqueda unidireccional para hallar el mínimo en la dirección elegida.

3- Tomar una 'dirección conjugada' de la anterior.

4- Volver al paso 2 hasta que se encuentre el óptimo.

La definición de lo que se entiende por 'dirección conjugada' da lugar a diferentes métodos. Así, la alternativa más sencilla consiste en variar cada vez una sola variable (fig. 3.11).

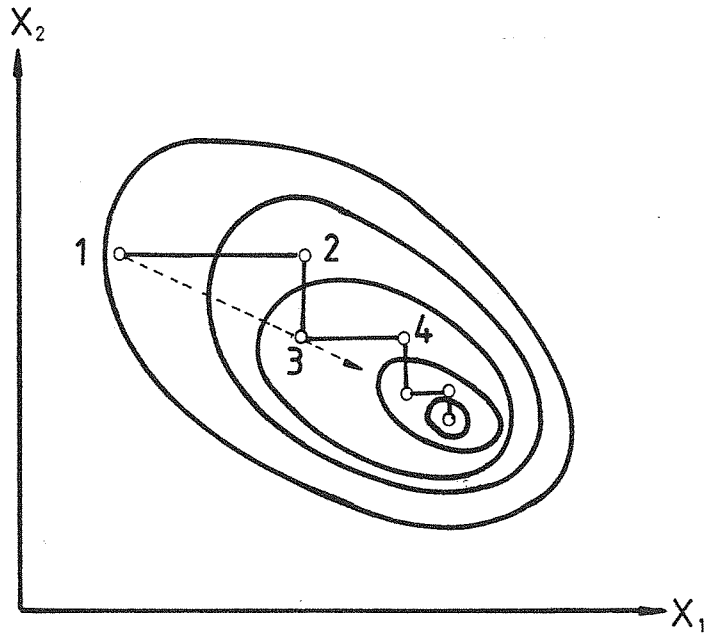

Figura 3.11. "Univariate method" para dos variables.

Observando la figura (3.11) se puede ver que las parejas de puntos alternos (1-3, 2-4. etc) definen direcciones que pasan muy cerca del óptimo. Esta observación intuitiva da lugar a un método mejorado del

anterior: el 'Pattern Move'. En este método, despues de haber dado un paso en dirección de cada uno de los ejes, podemos dar el siguiente paso en la dirección resultante de componer todas las anteriores. Sin embargo, la generalización de este método para más de dos variables no es correcta, y el método solo es útil hasta un número limitado de variables (5 ó 6).

El método de Powell es una ampliación de la idea anterior. La diferencia estriba en que cada dirección conjugada sustituye a una de las direcciones coordenadas originales (fig. 3.12).

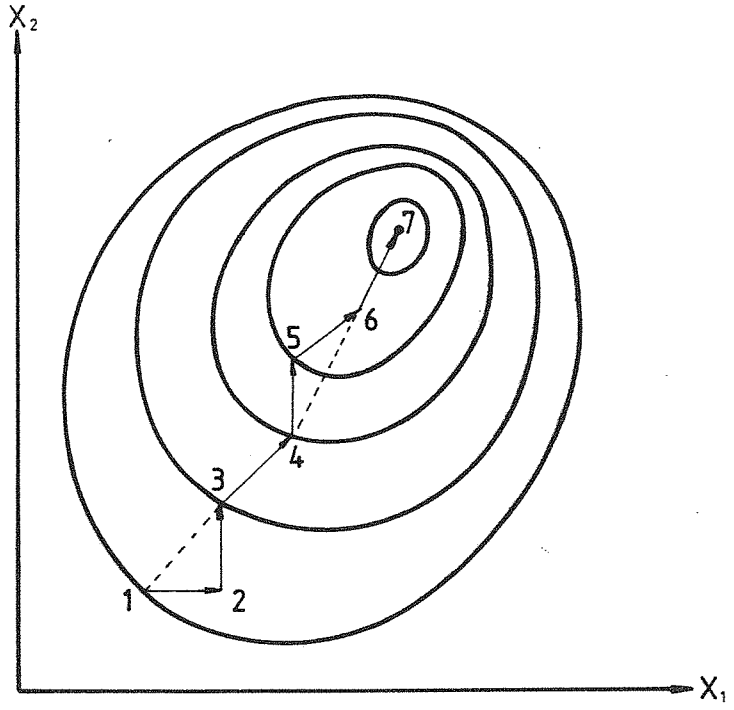

Figura 3.12. Método Powell de direcciones conjugadas.

Un segundo tipo de métodos son los que utilizan información de primeras derivadas. Genéricamente se conocen como metodos de primer orden. El más elemental es el 'Steepest Descent', que se basa en que en cualquier punto, la dirección de máximo descenso es la opuesta al gradiente de la función en dicho punto. Tal como se puede ver en la figura (3.13), el método es bueno para funciones "redondas", pero es muy lento para funciones excéntricas.

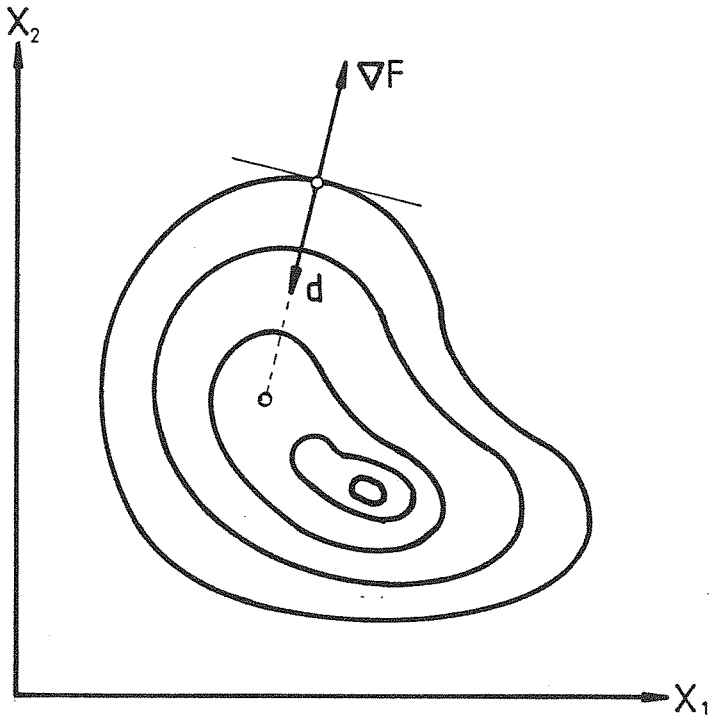

Figura 3.13. Metodo Steepest Descent.

Para funciones no redondas, es apropiado un método del tipo del de direcciones conjugadas, pero que aproveche la información de derivadas. Tal es el método de 'Gradiente conjugado' de Fletcher y Reeves.

El tercer tipo de métodos es el formado por aquellos que utilizan información de segundas derivadas. Estos métodos son los llamados métodos de segundo orden. Entre los métodos de segundo orden destaca el método Newton, que consiste en hacer un desarrollo en serie de la función objetivo truncado en las segundas derivadas. Derivando dicha expresión e igualando a cero, se obtiene el mínimo. Si la función objetivo es cuadrática, el mínimo se obtiene en la primera evaluación. En caso contrario se debe establecer un proceso iterativo. Una variante de este método se conoce como método quasi-Newton. Dicha variante consiste en obtener la matriz de segundas derivadas por un cálculo aproximado a partir de las primeras derivadas.

3.3.2.3 Convergencia.

Tal como se ha visto en (3.14), el proceso de diseño es iterativo. Por lo tanto, se debe preveer un mecanismo que detecte que se ha encontrado la solución optima y finalice el proceso. En teoría, el proceso se debe repetir tantas iteraciones como haga falta hasta llegar a la solución óptima. En la práctica, el proceso se da por acabado cuando se obtiene una solución que no se puede mejorar, o cuando se hace excesivamente largo. El mecanismo más elemental de parada es el que controla el máximo número de iteraciones.

Para evitar que el proceso entre en una etapa de mejoras muy pequeñas (cuyo costo de cálculo no compense la mejora obtenida), se suele definir una tolerancia alrededor de la solución óptima. De esta forma se considera buena cualquier solución que difiera de la optima menos que la tolerancia. Puesto que la solución óptima es desconocida en general, lo que se suele hacer es definir la tolerancia como un valor mínimo en el que debe diferenciarse el nuevo diseño respecto al actual para no dar por terminado el proceso. Es decir, se supone que en las proximidades del óptimo, el proceso iterativo obtiene diseños que cada vez difieren menos entre ellos Por tanto, el criterio de convergencia más simple es:

$$(x^{k+1} - x^{k}) < \epsilon \ / \ \epsilon > 0 \qquad (3.29)$$

En la figura (3.14) se observa como las diferencias de valor de la función objetivo y de las variables no tienen porqué ser del mismo orden de magnitud. De forma que, en las proximidades del óptimo, puede ocurrir que la función objetivo varie muy poco en un rango de variación de las variables muy grande.

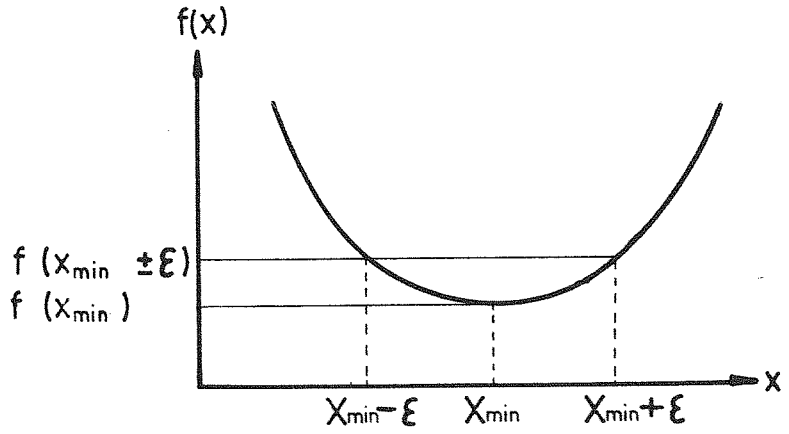

Figura 3.14. Variación de punto de diseño y función objetivo.

De la figura se desprende que no es rentable utilizar la modificación de las variables como criterio de convergencia (puesto que un gran esfuerzo para mejorar la estimación de las variables en el óptimo puede llevar a una mejora muy pequeña en la función objetivo) /3/. En consecuencia, un criterio de convergencia basado en la variación de la función objetivo conducirá en general a soluciones aceptables en menos iteraciones:

$$(f(x^{k+1}) - f(x^k)) < \epsilon \quad / \quad \epsilon > 0 \qquad (3.30)$$

Otros razonamientos semejantes dan lugar a diferentes criterios de convergencia, que generalmente aprovechan la información disponible sobre gradientes, derivadas, etc.

Por último, indicar que en la mayoría de los algoritmos existentes, el criterio de parada no es único. Por el contrario, se utilizan criterios multiples adaptados a las características del método y de la implementación, a fin de preveer todos los posibles malfuncionamientos.

3.3.2.4 Problema con restricciones.

En el caso más general, el problema de optimización es un problema de minimizar una función objetivo sujeto a restricciones. El efecto de las restricciones sobre el problema de optimización se puede ver en la figura (3.15). Un segundo efecto de las restricciones que debe tenerse en cuenta, es que pueden hacer aparecer varios mínimos relativos (incluso en un problema caracterizado por una función objetivo unimodal).

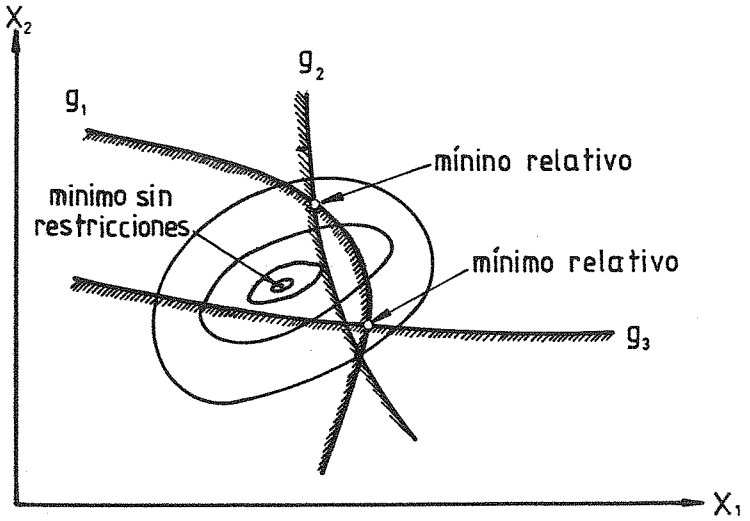

Figura 3.15. Efecto de las restricciones sobre el mínimo y mínimos relativos.

Para resolver el problema con restricciones hay dos estrategias diferentes:

- utilizar un método de resolución sin restricciones (al que se añaden indirectamente las condiciones que implican las restricciones), y

- utilizar métodos que abordan el problema completo.

Los primeros se denominan genéricamente 'metodos indirectos'. La forma más extendida de incluir las restricciones en un método que no las considera consiste en modificar el método para que tenga en cuenta las restricciones. Para ello, se puede modificar la función objetivo de forma que incluya el efecto de las restricciones. Esta idea da lugar a los llamados 'métodos de penalización' (Fiacco y McCormick /19/ introdujeron la denominación SUMT, o 'técnica de minimización secuencial sin restricciones'). Para estos métodos, la expresión de la función objetivo queda convertida en:

$$f_p(x,r) = f(x) + r \sum_{i=1}^{m} Gi(g_i) \qquad (3.31)$$

Siendo:

f_p la función objetivo penalizada;

f la función objetivo original;

r el 'peso' de la penalización, y

$G_i(g_i)$ una función de las restricciones.

La elección de las funciones G_i, da lugar a diferentes métodos de penalización, entre los que podemos distinguir dos variantes:

- métodos de penalización interior (o de barrera), y

- métodos de penalización exterior.

La diferencia entre ambos estriba en que en los primeros se llega a la solución a traves de soluciones intermedias factibles, mientras que en los segundos se llega desde soluciones intermedias no factibles. La ventaja de los primeros es que cualquier solución intermedia es buena (aunque no la mejor), mientras que en los segundos solo es buena la

última. A cambio, los primeros necesitan un punto de partida factible. Unas definiciones típicas de G_i en ambos casos son:

$$G_i = \begin{cases} g_i & \text{si } g_i > 0 \\ 0 & \text{si } g_i \leq 0 \end{cases} \qquad (3.32)$$

para métodos de penalización exterior, y

$$G_i = 1/g_i \qquad (3.33)$$

para métodos de penalización interior.

La representación gráfica de estas penalizaciones (Fox, /25/) es la mostrada en las figuras (3.16 y 3.17). En ellas se puede ver como la función objetivo se modifica en las cercanías de las restricciones, adaptandose progresivamente al contorno de las mismas, conforme va aumentandose el peso de su penalización.

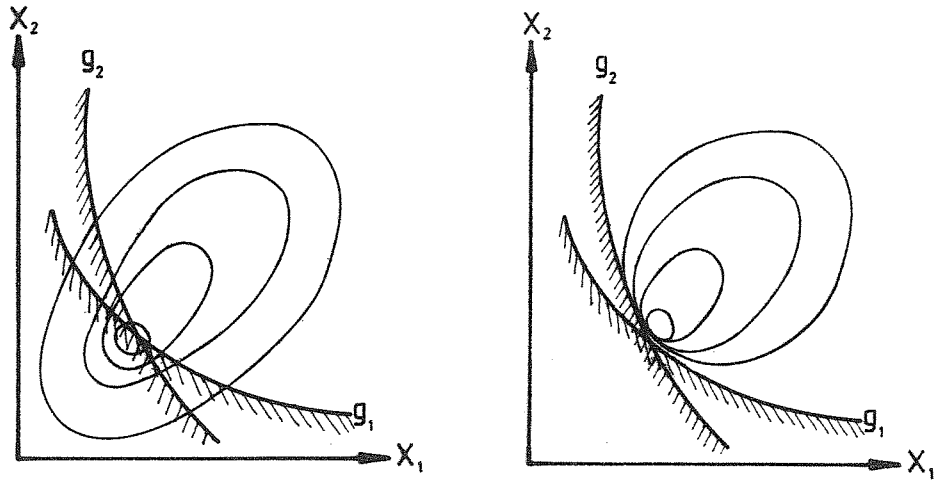

Figura 3.16. Penalización exterior.

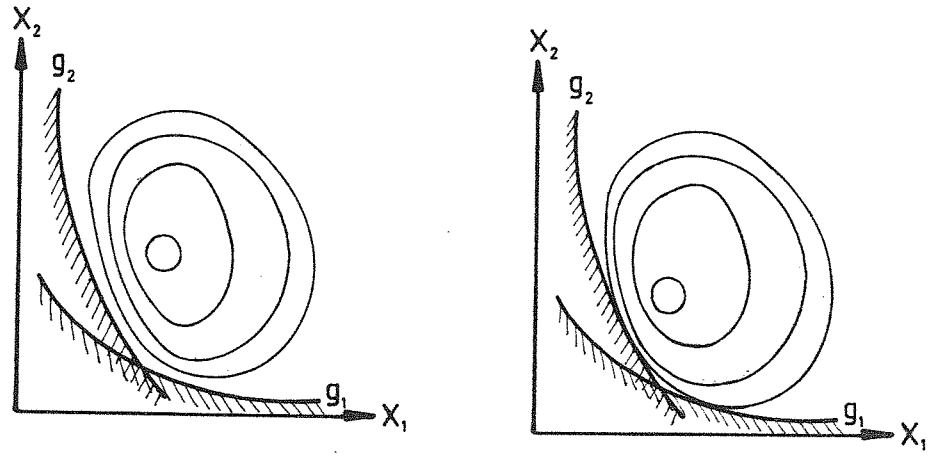

Figura 3.17. Penalización interior.

El segundo tipo de métodos son los denominados métodos directos. El más representativo de los métodos directos es el denominado 'método de direcciones posibles'. La estrategia de este método es básicamente una modificación del Steepest-Descent. Se trata de encontrar una dirección de descenso, para lo cual se utiliza la dirección del gradiente. Es decir, que la dirección d debe cumplir:

$$d^T \nabla f \leq 0 \qquad (3.34)$$

Pero, en este caso, además se debe cumplir que la dirección no lleve inmediatamente fuera de la región factible. Es decir:

$$d^T \nabla g_j \geq 0 \qquad j= 1,\ldots,m \qquad (3.35)$$

siendo m el número de restricciones activas.

Las direcciones que cumplen las dos condiciones impuestas (3.34 y 3.35) definen un cono de direcciones posibles, tal como se vé en la figura (3.18).

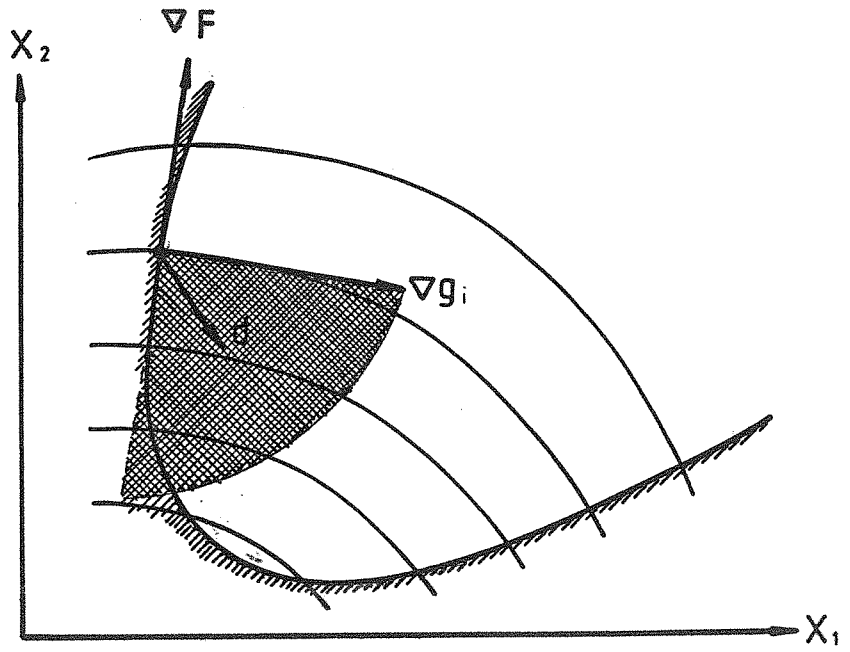

Figura 3.18. Direcciones posibles.

Para seleccionar la mejor dirección se puede formular el problema como un problema de programación lineal /74/, para el cual la dirección de máximo descenso de la función objetivo es la solución de:

maximizar β

$$\text{sujeta a} \quad d^T \nabla f + \beta \leq 0 \tag{3.36}$$

$$d^T \nabla g_i + \theta_j \beta \geq 0 \qquad j = 1,\ldots,m$$

y estando d acotada.

En donde θ son parametros que se ajustan para "primar" la dirección de máximo descenso (de la función objetivo) frente a la de máximo alejamiento (de la restricción), o viceversa. La elección depende de las características del problema. En la figura (3.19) se ve un ejemplo en el que primar la dirección de máximo descenso lleva a un camino muy escalonado.

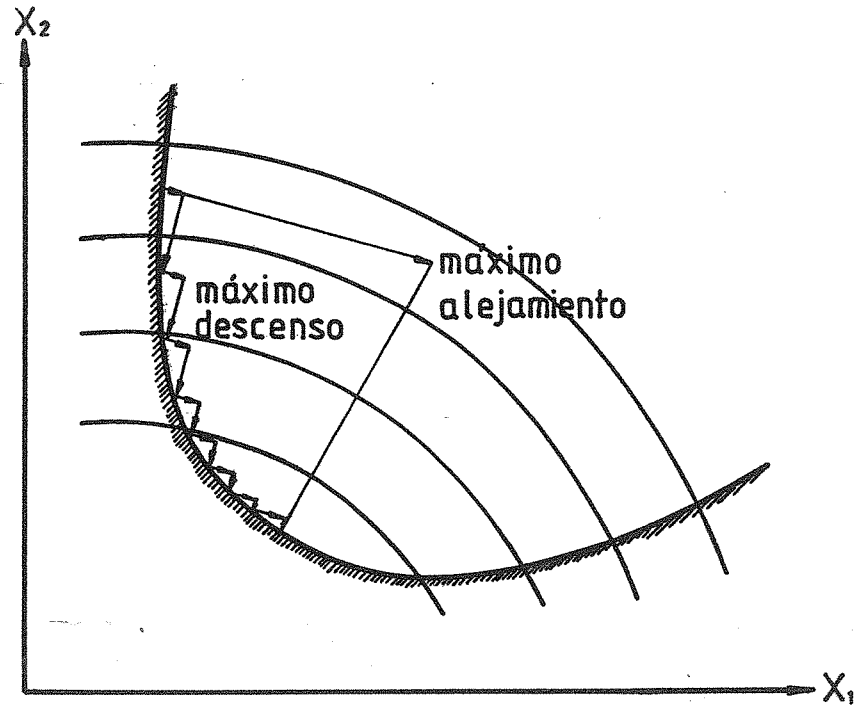

Figura 3.19. Maximo descenso frente a máximo alejamiento.

Una vez determinada la dirección de búsqueda, se debe aplicar alguno de los métodos de búsqueda unidimensional comentados, para encontrar la longitud de avance. Debe hacerse notar que según la longitud calculada puede hacerse activa o violada alguna restricción, por lo que hay que añadir un control para que esto no ocurra.

3.4 PROGRAMACION CUADRATICA SUCESIVA

3.4.1 Introducción.

La programación cuadrática sucesiva (RQP de aquí en adelante) es un método directo de programación matemática con restricciones; cuyas características generales han sido descritas en el epígrafe 3.3.4.2. En esencia, el método consiste en reducir el problema a un subproblema cuadrático para cada iteración del proceso de búsqueda del óptimo.

La idea básica de la RQP fué desarrollada por Wilson en 1963 /71/. En 1977, Han /30/ desarrolló, según esta técnica, un algoritmo que fué implementado por Powell /62/ con algunas modificaciones. Independientemente, Psenichny /63/ publicó en 1970 otro algoritmo RQP, en el que incluyó una estrategia de 'conjunto activo' de restricciones.

En la actualidad, los métodos RQP están ganando gran aceptación debido a dos características de los mismos:

- Se ha probado que convergen globalmente (desde cualquier punto arbitrario de partida);

- tienen relación lineal (o incuso superior) de convervencia.

A continuación vamos a hacer una descripción general del método RQP, que ampliaremos y particularizaremos despues con una descripción del algoritmo de Schittkowski /67/, dado que es el empleado en el sistema DISSENY. En /10/ se puede encontrar una descripción comparada de los algoritmos de Han y Psenichny

3.4.2 Descripción general.

Se pretende resolver el problema general de optimización no lineal:

$$\text{minimizar} \quad f(x)$$

$$\text{sujeto a} \quad g_j(x) \geq 0 \qquad j = 1,2,\ldots,m \qquad (3.37)$$

Para ello se va a emplear un método iterativo que nos permita encontrar un nuevo punto de diseño x^{k+1}, que mejore el punto actual x^k. La expresión que nos dará el nuevo punto de diseño será:

$$x^{k+1} = x^k + \alpha^k \, d^k \qquad (3.38)$$

siendo d^k un vector 'dirección', y α^k una 'longitud a recorrer', desde el punto de diseño actual al punto mejorado. Las formas generales de obtener α^k han sido discutidas en 3.3.1, por lo que vamos a centrarnos en el cálculo de d^k.

Podemos decir que encontrar el nuevo punto de diseño es encontrar un cambio en el diseño actual, tal que minimice el valor de la función objetivo y cumpla las restricciones. Es decir, que el problema se puede reformular como:

$$\begin{aligned} &\text{minimizar} \quad f(x^k+d^k) \\ &\text{sujeto a} \quad g_j(x^k+d^k) \geq 0 \qquad j= 1,2,\ldots,m \end{aligned} \tag{3.39}$$

La idea básica de todos los métodos RQP para calcular d^k consiste en convertir el problema en un subproblema cuadrático que se resuelve por las técnicas conocidas para dicho tipo de problemas. Hay diferentes aproximaciones para hacer la conversión del problema, pero todas ellas se basan en sustituir las funciones f y g_i por sus desarrollos en serie alrededor del punto de diseño x^k.

En todas las aproximaciones se hace uso de la función Lagrangiano:

$$L(x,u) = f(x) - \sum_{j=1}^{m} u_j \, g_j(x) \tag{3.40}$$

En donde los u_j son los multiplicadores de Lagrange.

El que todas las aproximaciones utilicen el lagrangiano, se debe a que una vez hecha la aproximación correspondiente para convertir en cuadrático el problema se exige a esta el cumplimiento de las condiciones de optimalidad de Kuhn-Tucker:

$$\begin{aligned} &\nabla L(x,u) = 0 \\ &g_j(x) \geq 0 \qquad j=1,2,\ldots,m \\ &u_j \geq 0 \qquad j=1,2,\ldots,m \\ &g_j(x)\, u_j = 0 \qquad j=1,2,\ldots,m \end{aligned} \tag{3.41}$$

La forma más directa de convertir el problema en cuadrático es hacer una aproximación cuadrática de la función objetivo. Hacer también una aproximación cuadrática de las restricciones conduce a un problema para el que se han propuesto algunas técnicas iterativas, pero que no es suficientemente conocido /10/. Por ello, lo habitual es hacer una

aproximación lineal de las restricciones. Con estas aproximaciones, el problema se puede reformular como:

$$\text{minimizar} \quad 1/2\ d^{kT}\ \nabla^2 f(x^k)^T\ d^k + \nabla f(x^k)^T\ d^k$$

$$\text{sujeto a} \quad \nabla g_j(x^k)^T\ d^k + g_j(x^k) \geq 0 \quad j= 1,2,\ldots,m \qquad (3.42)$$

Se añade una condición para asegurar que la solución este acotada:

$$1/2\ d^{kT}\ W^k\ d^k \leq e^2 \qquad (3.43)$$

En donde W^k es una matriz que debe ser definida positiva. Por tanto, basta tomar la matriz identidad. No obstante, en /10/ se muestra como incluyendo en dicha matriz información de curvatura de las funciones se aumenta la relación de convergencia.

El problema así formulado consiste en encontrar una dirección d^k tal que conduzca a la región factible (ninguna restricción violada), con el menor valor posible para la función objetivo, y a una distancia no superior a e. No obstante, si e es lo suficientemente pequeño, o el k-esimo punto de diseño está suficientemente lejos de la región factible, será imposible encontrar una solución que cumpla las condiciones (3.42) y (3.43) al mismo tiempo.

Las condiciones de optimalidad del punto elegido, quedan más patentes planteando las condiciones de Kuhn-Tucker:

$$\nabla f(x^k) - \sum_{j=1}^{m} u'_j\ \nabla g_j(x^k)^T + (\nabla^2 L(x^k,u') + \mu' I)\ d^k = 0$$

$$u'_j\ (\nabla g_j(x^k)^T\ d^k + g_j(x^k)) = 0 \quad j= 1,2,\ldots,m$$

$$\mu'(1/2\ d^{kT}\ W^k\ d^k - e^2) = 0 \qquad (3.44)$$

$$u'_j \geq 0 \quad j= 1,2,\ldots,m$$

$$\mu' \geq 0$$

En estas condiciones, es importante observar que la expresión del lagrangiano se ha obtenido haciendo una aproximación cuadrática tanto de la función objetivo como de las restricciones.

De estas condiciones se deduce que tomando $\mu'>0$ se asegura indirectamente que la región factible se puede alcanzar independientemente de lo lejano que esté el punto de diseño x^k. Por lo tanto, imponiendo esta condición en (3.44), se obtiene:

$$\nabla f(x^k) - \sum_{j=1}^{m} u'_j \nabla g_j(x^k)^T + (\nabla^2 L(x^k,u') + \mu' I)\, d^k = 0$$

$$u'_j\, (\nabla g_j(x^k)^T\, d^k + g_j(x^k)) = 0 \quad j= 1,2,\ldots,m \tag{3.45}$$

$$u_j \geq 0 \quad j= 1,2,\ldots,m$$

problema que tiene siempre solución. Afortunadamente, no es necesario resolver el problema así planteado. Haciendo el cambio:

$$u_j = u'_j/\mu' \tag{3.46}$$

$$\mu = 1/\mu' \tag{3.47}$$

podemos reescribir (3.45) como:

$$\mu\, \nabla f(x^k) - \sum_{j=1}^{m} u_j \nabla g_j(x^k)^T + \mu\, (\nabla^2 L(x^k,u) + \mu' I)\, d^k = 0$$

$$u_j\, (\nabla g_j(x^k)^T\, d^k + g_j(x^k)) = 0 \quad j= 1,2,\ldots,m \tag{3.48}$$

$$u_j \geq 0 \quad j= 1,2,\ldots,m$$

Planteando ahora el problema cuadrático:

$$\begin{aligned} &\text{minimizar} \quad 1/2\; d^{kT}\, W^k\, d^k + \mu\; f(x^k)^T\, d^k \\ &\text{sujeto a} \quad g_j(x^k)^T\, d^k + g_j(x^k) \geq 0 \quad j= 1,2,\ldots,m \end{aligned} \tag{3.49}$$

se observa que las condiciones de Kuhn-Tucker para él, son la mismas de (3.48), sin más que tomar:

$$W^k = \nabla^2 L(x^k,u) + \mu' I \tag{3.50}$$

$$\mu = 1/\mu' = 1 \tag{3.51}$$

De la expresión (3.50) se puede comentar que para un μ' suficientemente grande, W^k es definida positiva. Por tanto, W^k puede verse como una aproximación definida positiva al hesiano de la función lagrangiano. Esta matriz, conteniendo información de curvaturas de las funciones, es la que hace que el método tenga una relación de convergencia superlineal. No obstante, dado el esfuerzo de cálculo que requiere conocer la matriz W^k, es habitual recurrir a estimaciones de la misma del tipo Quasi-Newton. Por lo tanto, la relación de convergencia suele ser menor que la máxima posible; aunque sigue siendo igual o mayor que la unidad mientras siga siendo definida positiva.

El significado de la expresión (3.51) se puede ver claramente en /10/, en donde se llega a una expresión:

$$\mathbf{d}^k = \mu\, \mathbf{d}_1 + \mathbf{d}_2 \tag{3.52}$$

en donde $\mathbf{d}_1$ es la componente de máximo descenso (mayor reducción de la función objetivo); mientras que $\mathbf{d}_2$ es la componente de máximo acercamiento a la región factible. Por tanto, tomar μ=1 significa dar el mismo peso a la minimización de la función objetivo que al cumplimiento de las restricciones.

De todo lo anterior se deduce que podemos resolver el problema (3.49) para encontrar el valor $\mathbf{d}^k$ buscado en (3.39). No obstante, es difícil encontrar la solución debido a que:

$$\nabla^2 L(\mathbf{x}^k,\mathbf{u}) = \nabla^2 f(\mathbf{x}^k)^T - \sum_{j=1}^{m} u_j\, \nabla^2 g_j(\mathbf{x}^k)^T \tag{3.53}$$

es una función que depende de $\mathbf{u}$, que es el vector de los multiplicadores de Lagrange en el óptimo. Dado que el valor de los multiplicadores en el óptimo no es conocido a priori, para obtener la solución se recurre a estimaciones de los mismos para las diferentes iteraciones:

$$\nabla^2 L(\mathbf{x}^k,\mathbf{u}) = \nabla^2 f(\mathbf{x}^k)^T - \sum_{j=1}^{m} u^k{}_j\, \nabla^2 g_j(\mathbf{x}^k)^T \tag{3.54}$$

Estas estimaciones de los multiplicadores se pueden realizar de varias formas; una de las más habituales es:

$$\mathbf{u}^k = -N^+ \nabla f(\mathbf{x}^k) \tag{3.55}$$

siendo:

$$N^+ = (N^T N)^{-1} N^T \tag{3.56}$$

la seudoinversa de la matriz N formada por m columnas que contienen a los vectores $g_j(\mathbf{x}^k)^T$. Esta aproximación será tanto más real conforme el proceso iterativo se vaya acercando al óptimo.

3.4.3 Algoritmo de Schittkowski.

3.4.3.1 Descripción.

El algoritmo propuesto por Schittkowski /67/ es un RQP con aproximación cuadrática de la función Lagrangiana y aproximación lineal de las restricciones. El algoritmo plantea la búsqueda iterativa del punto óptimo de diseño por medio de la expresión (3.38). Para ello establece una estrategia de cálculo diferente para la dirección de búsqueda y para la longitud.

Para la dirección de búsqueda el algoritmo plantea la resolución del problema formulado en (3.49). Si bién establece dos diferencias importantes respecto a dicha formulación:

- la consideración de restricciones de igualdad, y

- la consideración del subconjunto de restricciones activas.

Considerando las restricciones de igualdad independientemente, se llega a una formulación más general. Esto permite un tratamiento numérico más eficiente de cada tipo de restricciones. La formulación del problema queda convertida en:

$$\begin{aligned} &\text{minimizar} \quad 1/2\ d^{kT}\ W^k\ d^k + \nabla f(x^k)^T\ d^k \\ &\text{sujeto a} \quad \nabla g_j(x^k)^T\ d^k + g_j(x^k) = 0 \quad j = 1,2,\ldots,m_i \\ &\qquad\qquad \nabla g_j(x^k)^T\ d^k + g_j(x^k) \geq 0 \quad j = m_i+1,\ldots,m \end{aligned} \tag{3.57}$$

La segunda consideración pretende aumentar la eficiencia del algoritmo eliminando cálculos innecesarios de gradientes de las restricciones no activas. La formulación (3.57) se replantea como:

$$\begin{aligned} &\text{minimizar} \quad 1/2\ d^{kT}\ W^k\ d^k + \nabla f(x^k)^T\ d^k \\ &\text{sujeto a} \quad \nabla g_j(x^k)^T\ d^k + g_j(x^k) = 0 \quad j \in J_a{}^k \\ &\qquad\qquad \nabla g_j(x^k)^T\ d^k + g_j(x^k) \geq 0 \quad j \in J_i{}^k \end{aligned} \tag{3.58}$$

siendo:

$J_a{}^k$ el conjunto formado por todas las restricciones de igualdad más las restricciones activas en la iteración k.

$J_i{}^k$ todas las restricciones no pertenecientes a $J_a{}^k$.

Una restricción es activa si su valor es negativo o si el correspondiente multiplicador de Lagrange es mayor que cero. Para

evitar situaciones inestables provocadas por restricciones no activas pero muy cercanas a serlo ($g_j(x^k) > 0$; $g_j(x^k) \to 0$), se trabaja con restricciones ϵ-activas. Así ,se define el conjunto $J_a{}^k$ como el formado por todas las restricciones de igualdad más las restricciones de desigualdad que cumplen $g_j(x^k) \leq \epsilon$ para la iteración k.

Para evitar que el problema pueda ser inconsistente, introducimos esta nueva condición por medio de una nueva variable δ. De forma que el problema (3.58) queda convertido en:

$$\begin{aligned} &\text{minimizar} \quad 1/2\ d^{kT}\ W^k\ d^k + \nabla f(x^k)^T\ d^k + 1/2\ \rho^k\ (\delta^k)^2 \\ &\text{sujeto a} \quad \nabla g_j(x^k)^T\ d^k + (1-\delta)\ g_j(x^k) = 0 \quad j \in J_a{}^k \\ &\qquad\qquad \nabla g_j(x^k)^T\ d^k + g_j(x^k) \geq 0 \quad j \in J_i{}^k \end{aligned} \tag{3.59}$$

en donde a la variable δ se le exige que $0 \leq \delta \leq 1$, y ρ es el parámetro de penalización de dicha variable.

Una vez resuelto el problema (3.59), se tiene una dirección de búsqueda d^k y una estimación (p.e. 3.46) de los multiplicadores de Lagrange. Entonces, para actualizar las variables y los multiplicadores se debe plantear:

$$x^{k+1} = x^k + \alpha^k\ d^k \qquad v^{k+1} = v^k + \alpha^k\ (u^k - v^k) \tag{3.60}$$

Para encontrar la longitud de búsqueda α^k, Han y Powell minimizaban una función de búsqueda dada por:

$$\Phi(x,r) = f(x) + \sum_{j=1}^{mi} r_j\ |g_j(x)| + \sum_{j=mi+1}^{m} r_j\ |\min(0,g_j(x))| \tag{3.61}$$

en donde $r = (r_1,\ldots,r_m)^T$ son los parámetros de penalización por violación de las diferentes restricciones.

Para soslayar los problemas que da la utilización de esta función (pérdida de la relación de convergencia superlineal, etc), se ha sustituido por una función lagrangiana diferenciable:

$$\begin{aligned} \Phi_r(x,v) = f(x) &+ \sum_{j \in Ja} (v_j\ g_j(x) - 1/2\ r_j\ g_j(x)^2) \\ &- 1/2 \sum_{j \in Ji} v_j{}^2/r_j \end{aligned} \tag{3.62}$$

A partir de (3.62) podemos definir la función a minimizar con respecto a α para obtener la longitud α^k:

$$\phi(\alpha) = \Phi_r \left[\begin{bmatrix} x^k \\ v^k \end{bmatrix} + \alpha \begin{bmatrix} d^k \\ (u^k - v^k) \end{bmatrix} \right] \qquad (3.63)$$

Debe, no obstante señalarse que la elección de los parámetros de penalización debe cumplir ciertas condiciones /67/ para evitar comportamientos anómalos del algoritmo.

3.4.3.2 Algoritmo.

Los principales pasos del algoritmo pueden describirse de la forma:

0) Inicio (k=0). Se elige cualquier punto de partida (x^0, v^0, W^0, r^0), se evaluan los valores de las funciones y sus gradientes. Luego se determina el subconjunto activo $J_a{}^k$.

1) Se resuelve el subproblema cuadrático (3.59). La solución obtenida es d^k, δ^k, y los multiplicadores u^k.

2) Se determinan los nuevos parámetros de penalización r^{k+1}, ρ^{k+1}.

3) Se realiza una búsqueda unidimensional por (3.63) para obtener α^k.

4) Se actualizan las variables y los multiplicadores.

5) Se evaluan los nuevos valores de las funciones y sus gradientes y se estima el nuevo valor del Hesiano.

6) Se comprueba el criterio de convergencia y se finaliza; o bién comienza un nuevo ciclo (k= k+1) volviendo al paso 1.

En cuanto a la resolución del subproblema cuadrático, puede emplearse la subrutina de Gill, Murray Saunders y Wright /28/, u otra basada en la publicada por Lawson y Hanson /44/ (que es la empleada en el sistema DISSENY).

Por lo que respecta al criterio de convergencia, se utiliza uno combinado que tiene en cuenta tanto el valor del lagrangiano:

$$\|L(x^k,u)\|^2 \leq \epsilon \qquad (3.64)$$

como la información de derivadas de las funciones:

$$d^{kT} W^k d^k \leq \epsilon \qquad (3.65)$$

Por último, indicar que en /67/ se encuentra una descripción más detallada del algoritmo, que incluye ciertos parámetros de ajuste necesarios para evitar comportamientos anómalos.

CAPITULO 4

EL PROBLEMA DE ANALISIS

4.1 INTRODUCCION.

El análisis de una estructura es la determinación del comportamiento de la idealización de la misma, bajo las acciones exteriores a que está sometida. La formulación de este problema se lleva a cabo por diferentes procedimientos; siendo el de los elementos finitos uno de los que poseen mejores características numéricas, conservando al mismo tiempo una formulación más general. Este es el método que se ha empleado en la implementación del sistema DISSENY.

Puesto que el comportamiento de las estructuras depende de muchos factores, el primer paso consiste en identificar cuales de ellos tienen una influencia más drástica. De esta forma, se podrán establecer diferentes tipos de análisis en función de la consideración que se de a tales factores. Con ello se pretende abordar los problemas con formulaciones más sencillas que las resultantes de considerar el problema más general. Es decir, se establecen como principios fundamentales de cada tipo de análisis las condiciones que se imponen a los factores que pueden modificar tanto las condiciones del problema, que invaliden las suposiciones hechas para poder resolverlo. En nuestro caso, vamos a ceñirnos al caso de análisis estático elástico-lineal, en el que se considera que se cumplen las siguientes condiciones:

- Las cargas a que está sometida la estructura no varian con el tiempo, y su aplicación es suficientemente lenta para que su efecto no varie con el tiempo (problema estático).

- El orden de aplicación de las cargas es indiferente.

- El material se comporta de forma elástica.

- Las deformaciones producidas son suficientemente pequeñas como para considerar despreciable el cambio de geometría que provocan.

Las técnicas de análisis que se emplean en la actualidad están muy depuradas, y son cómodas de utilizar , pero no están adaptadas a su uso dentro de un proceso de diseño interactivo. Esto se debe a que:

- un proceso de diseño requiere generalmente muchos análisis;

- los programas de análisis no generan habitualmente toda la información necesaria para un proceso de diseño.

- no hay adaptación del modelo de diseño al de análisis, y

Si bién los programas de análisis disponibles actualmente están muy depurados, y resultan muy eficientes, al incluirlos como parte de un proceso de diseño aparecen ciertas peculiaridades que deben ser tenidas en cuenta para que el proceso global resulte más eficiente. Estas técnicas se basan fundamentalmente en el hecho de que el análisis se repite mas de una vez. Por tanto, lo que se busca es la eficiencia del "reanálisis".

Por otra parte, el método de los elementos finitos es una poderosa herramienta para evaluar diseños, pero no es útil para identificar posibles formas de modificar el diseño para evitar problemas o mejorar una solución. El método se usa para "encontrar" malos diseños, pero no ayuda a mejorarlos. Utilizando métodos que se apoyan en la formulación del método de los elementos finitos, se puede generar información de sensibilidades de diseño que permitan al diseñador mejorar los diseños de forma sistemática.

Una de las formas más habituales de diseño implica la toma de decisiones, por parte del diseñador, basadas en la experiencia y la intuición. Esta forma convencional de diseñar puede ser mejorada sustancialmente si se provee al diseñador de información de sensibilidades. Estas sensibilidades le muestran la influencia de los posibles cambios en el diseño, sin recurrir a la prueba y el error.

La segunda razón del creciente interés que despierta el cálculo de sensibilidades descansa en el hecho de que la gran mayoría de los métodos de optimización que se utilizan en la actualidad se basan en el empleo de sensibilidades. Desde este punto de vista, se puede definir el propósito del cálculo de sensibilidades como el de evaluar

las derivadas que se necesitan para resolver los problemas de optimización.

El tercer problema es el de la idealización del modelo, que ha sido abordado por diferentes autores, llegandose a la conclusión de que el modelo que se emplea para describir una estructura que se va a analizar por elementos finitos no es el apropiado para describir el modelo de diseño de dicha estructura. Entre las soluciones apuntadas, la que parece más prometedora es la de Fleury /23/, que propugna el uso de los conceptos de modelado geométrico aplicados en CAD.

En este capítulo vamos a revisar brevemente el problema "clasico" de análisis por el método de los elementos finitos, y, a continuación, vamos a introducir los aspectos que el análisis debe considerar para incluirse eficientemente en el problema más amplio de diseño. Referencias más amplias sobre el método de análisis se pueden encontrar en la obra de Zienkiewicz /76/, y en la propia implementación del procesador ADEF de análisis del sistema DISSENY /51/. El aspecto concerniente a la idealización , ó modelización, se considera en el capítulo 6, cuando se describe la organización de la información.

4.2 EL PROBLEMA DE ANALISIS

4.2.1 Idealización de la estructura.

El proceso de idealización de la estructura consiste en la representación de la estructura real, infinitamente compleja, por un modelo matemático con un grado finito de complejidad. El proceso de idealización es necesario porque la mente humana, la capacidad analítica de los ordenadores y el tiempo disponible para hallar la solución de los problemas planteados, son limitados.

El proceso de idealización se aplica tanto a la geometría como a las cargas; al comportamiento y a la función de la estructura. Incluye la identificación y cuantificación de aquellos aspectos de la estructura que van a ser tenidos en cuenta en el proceso de análisis. Parte del proceso de idealización es llevado a cabo por los legisladores. Así, las normas prescriben las cargas de diseño, los factores de carga, las flechas admisibles, etc, comenzando de esta manera el proceso de idealización. El resto del proceso de idealización debe ser llevado a cabo por el diseñador como fase previa al análisis.

Cuando se puede obtener un modelo, que describa con suficiente precisión al objeto real, utilizando un número finito de componentes bién definidos,el problema se denomina 'discreto'. En caso contrario, el problema se dice 'continuo'.

Los problemas discretos pueden resolverse generalmente sin dificultad, aún cuando el número de elementos sea muy elevado. Es el caso de las estructuras de elementos prismáticos de directriz recta (barras). Por el contrario, como la capacidad de los ordenadores es finita, los problemas continuos solo pueden resolverse de forma exacta mediante manipulaciones matemáticas (usando ecuaciones diferenciales o expresiones equivalentes con un número infinito de elementos implicados). Además, las técnicas matemáticas disponibles se limitan a casos extremadamente simplificados. Por tanto, la única alternativa práctica disponible consiste en convertir el problema en otro equivalente y discreto. Puesto que en general no es posible obtener un problema exactamente equivalente, se recurre a aproximaciones; de las cuales cabe esperar que se acerquen, tanto como se quiera, a la solución correcta al ir aumentando el número y calidad de los elementos discretos implicados.

El acercamiento más intuitivo al problema de la discretización se obtiene creando una analogía entre elementos discretos reales y porciones finitas de un dominio continuo. Así, en la década de los años cuarenta se demostró que se pueden obtener soluciones razonablemente buenas de un problema continuo, sustituyendo pequeñas porciones del continuo por una distribución de barras elásticas simples. Más tarde, se demostró que se pueden sustituir las propiedades del continuo de un modo más directo y no menos intuitivo, suponiendo que las pequeñas porciones del mismo, o 'elementos', se comporten de una cierta forma simplificada.

Este planteamiento dió lugar al nacimiento del concepto de 'elemento finito'. En él, el continuo de una estructura se idealiza por un número limitado de elementos, interconectados en un conjunto dado de puntos.

Los puntos de conexión de los elementos son llamados 'puntos nodales', o 'nudos'. Estos nudos están situados en el espacio tridimensional, y referidos a un sistema de coordenadas. En general, tienen libertad de traslación y rotación, que se restringe en ciertos casos para incluir las condiciones de contorno de la estructura. Los elementos finitos

son el medio de especificar las propiedades locales de la estructura: la geometría y las características del material.

A partir de los nudos y los elementos quedan definidas las características de la estructura. A continuación se deben definir las acciones a que va a estar sometida. Para ello se idealizan las cargas reales que deberá soportar por medio de cargas puntuales aplicadas en los nudos, y (en ciertos tipos de elementos), como cargas distribuidas sobre ellos.

4.2.2 El Método de los Elementos Finitos.

4.2.2.1 Introducción.

El análisis del comportamiento del tipo de estructura en estudio, suele referirse a la deformación de la misma, y a los esfuerzos que soporta, que es la información que el diseñador necesita para conocer el efecto de las acciones sobre la misma. En ciertos casos solo se necesita conocer una de las dos medidas de la respuesta. En función de cuales sean las incognitas del problema, el método apropiado para resolverlo varía. Esta dualidad es la que ha dado lugar a la aparición de dos métodos básicos de resolución del problema de análisis:

- el método de los desplazamientos, y

- el método de las fuerzas.

Mientras en el primer caso las incógnitas son los desplazamientos de los nudos de la estructura, en el segundo las incognitas son los esfuerzos a que están sometidos los elementos. En ambos casos se deben cumplir tres condiciones:

- que la estructura esté en equilibrio;

- que haya compatibilidad de los desplazamientos, y

- que se cumplan las leyes de comportamiento de los elementos.

Además, la estructura deberá cumplir unas condiciones de contorno que ligan su comportamiento global respecto a su entorno (interacción estructura-entorno).

Los dos métodos propuestos únicamente se diferencian en el orden en que exigen que se cumplan estas condiciones (para lo cual emplean

diferentes incognitas). Pero, en definitiva, de ambas formas se puede llegar a obtener todos los resultados.

Si bién el método de las fuerzas es ventajoso para algún tipo de problemas, su dependencia del problema le hace menos apto para ser implementado en ordenador. Por ello, el método de los desplazamientos es el más ampliamente usado. En la obra de Kardestuncer /39/ se puede encontrar un estudio de ambos métodos.

Como ya se ha comentado al hablar de la idealización, el método de los elementos finitos se basa en idealizar el continuo de una estructura por un número limitado de elementos, interconectados en un conjunto dado de nudos. Este método basa su atractivo en la relativa facilidad para establecer las ecuaciones que gobiernan el comportamiento del sistema y las buenas propiedades numéricas de las matrices del sistema, tanto como en su generalidad respecto a los tipos de estructuras que puede analizar.

El proceso de análisis por el método de los elementos finitos se puede resumir en los siguientes pasos /76/:

a) El continuo se divide, mediante líneas o superficies imaginarias, en un número limitado de 'elementos finitos'.

b) Se conectan entre sí los elementos en un número limitado de nudos, situados en sus contornos.
Los desplazamientos de estos nudos serán las incognitas fundamentales del problema.

c) Se toma un conjunto de funciones que definan (de manera única) el campo de desplazamientos dentro de cada elemento, en función de los desplazamientos de los nudos a que está conectado dicho elemento.

d) Estas funciones de desplazamiento definen entonces (de manera única) el estado de deformación dentro del elemento, en función de los desplazamientos de los nudos.
Estas deformaciones, junto con las propiedades constitutivas del material, definen el estado de tensiones en todo el elemento, y, por consiguiente, también en sus contornos.

e) Se aplica el teorema de los desplazamientos virtuales para establecer las ecuaciones de equilibrio (cálculo de las matrices de rigidez de los elementos).

Se calcula la matriz de rigidez global ensamblando las matrices de rigidez de los elementos.

f) Se calcula el vector de cargas puntuales aplicadas sobre los nudos de la idealización, a partir de las cargas distribuidas sobre el continuo.

g) Se imponen las condiciones de contorno y se resuelve el sistema de ecuaciones de equilibrio sobre los nudos; obteniendose los desplazamientos de los mismos.

h) Se evaluan todo tipo de esfuerzos.

Debe notarse que la solución obtenida es únicamente una aproximación. La razón es que hemos hecho una serie de aproximaciones. En primer lugar, no siempre es fácil asegurar que las funciones de desplazamientos escogidas satisfagan las condiciones de continuidad de los desplazamientos entre elementos adyacentes. Por consiguiente, esta condición de compatibilidad, puede no cumplirse en el contorno de los elementos (aunque es evidente que dentro de cada elemento sí se cumplirá, a causa de la unicidad de los desplazamientos implicada en el hecho de que los mismos estén representados por funciones continuas). En segundo lugar, al concentrar las cargas equivalentes en los nudos, las condiciones de equilibrio solo se cumplirán para el conjunto continuo. No se equilibra cada elemento por separado.

4.2.2.2 Formulación general del método.

Para obtener las ecuaciones de equilibrio del cuerpo sometido a estudio, se le aplica el teorema de los desplazamientos virtuales. Para ello, consideramos el cuerpo sometido a fuerzas de volumen F^v, fuerzas de superficie F^s y fuerzas concentradas F^i. Entonces planteamos:

$$\int_V \epsilon^{*T} \tau \, dV = \int_V U^{*T} F^V \, dV + \int_S U^{s*T} F^s \, dS + \sum_i U^{i*T} F^i \qquad (4.1)$$

Siendo :

ϵ^* el vector de deformaciones virtuales.

τ el vector de tensiones.

U^* el vector de desplazamientos virtuales.

U^{s*} el vector de desplazamientos virtuales para los puntos de la superficie.

U^{i*} el vector de desplazamientos virtuales para los puntos en que se aplican cargas puntuales.

Las integrales están extendidas sobre el volumen y la superficie del cuerpo, respectivamente. El sumatorio está extendido a todos los puntos donde hay cargas puntuales.

La expresión (4.1) podemos ponerla como suma de integrales sobre el volumen y áreas de todos los elementos finitos:

$$\sum_m \int_{V_m} \epsilon_m^{*T} \tau_m \, dV_m = \sum_m \int_{V_m} U_m^{*T} F_m^V \, dV_m + \sum_m \int_{S_m} U_m^{s*T} F_m^s \, dS_m + \sum_i U^{i*T} F^i \qquad (4.2)$$

La ecuación (4.2) expresa la condición de equilibrio del cuerpo considerado como un ensamblaje de elementos conectados en los puntos nodales. Para poder aplicar esta condición necesitamos modificarla de forma que en lugar de considerar los desplazamientos y las deformaciones de un punto arbitrario de cualquier elemento (m), considere únicamente los desplazamientos de los puntos nodales, que van a ser nuestras incógnitas. Para ello debemos encontrar una matriz de transformación de desplazamientos H, tal que:

$$U_m = H^m U \qquad (4.3)$$

Siendo U el vector de desplazamientos de todos los nudos del ensamblaje.

Análogamente, debemos encontrar una matriz B que relacione las deformaciones con los desplazamientos:

$$\epsilon_m = B^m U \qquad (4.4)$$

Las expresiones de las matrices H y B se obtienen por la formulación particular para cada tipo de elemento. Esta forma de definir los desplazamientos y deformaciones del elemento en función del vector de desplazamientos totales del cuerpo facilita el proceso de ensamblaje

de la matriz de rigidez de la estructura a partir de las matrices de los elementos, y se conoce como 'método directo de las rigideces'.

Por otra parte, las tensiones de un elemento están relacionadas con las deformaciones del mismo por medio de la matriz de propiedades elásticas D:

$$\tau_m = D_m \epsilon_m \tag{4.5}$$

Por lo tanto, sustituyendo las expresiones (4.3), (4.4) y (4.5) en la expresión (4.2), se obtiene:

$$U^{*T} \sum_m \int_{V_m} B_m^T D_m B_m \, dV_m \; U = U^{*T} \sum_m \int_{V_m} H_m^T F_m^V \, dV_m +$$

$$U^{*T} \sum_m \int_{S_m} H_m^{sT} F_m^s \, dS_m + U^{*T} F \tag{4.6}$$

Donde las matrices de interpolación de los desplazamientos de la superficie H_m^s se obtiene a partir de las matrices de interpolación de desplazamientos H_m, sustituyendo las coordenadas de la superficie del elemento; F es un vector de fuerzas aplicadas en los puntos nodales de la estructura.

Para obtener los desplazamientos de los puntos nodales imponemos desplazamientos virtuales unitarios en todas las componentes de los desplazamientos de la ecuación (4.6). De esta forma, llegamos a las ecuaciones de equilibrio de la estructura correspondientes a los desplazamientos de los puntos nodales:

$$K U = R \tag{4.7}$$

La matriz de rigidez del ensamblaje de elementos K, se obtiene de la forma:

$$K = \sum_m K_m = \sum_m \int_{V_m} B_m^T D_m B_m \, dV_m \tag{4.8}$$

Y el vector de cargas R (que incluye el efecto de las fuerzas de volumen, superficie y puntuales) toma el valor:

$$R = R_V + R_S + R_P = \sum_m \int_{V_m} H_m^T F_m^V \, dV_m + \sum_m \int_{S_m} H_{sm}^T F_m^s \, dS_m + F \tag{4.9}$$

4.2.2.3 Transformación de coordenadas.

Con el objeto de facilitar la formulación, las matrices de rigidez de los elementos se obtienen en sistemas de coordenadas locales para cada elemento, que en general son distintos del sistema de coordenadas global utilizado para la estructura. Así pues, antes de que podamos formar el sistema de ecuaciones de equilibrio, debemos transformar las rigideces de los elementos, y las cargas, a un sistema de coordenadas común: el sistema global de coordenadas.

Si adoptamos la notación de emplear letras minúsculas para los valores en cada sistema local, y letras mayúsculas para estos mismos valores en el global, podemos escribir la relación entre ambos sistemas como:

$$x = T_p \, X \tag{4.10}$$

Esta ecuación (4.10) corresponde a una transformación que incluye las traslaciones y giros que relacionan ambos sistemas de coordenadas.

Así, la relación entre los desplazamientos de los nudos de un elemento, y el vector de desplazamientos de toda la estructura vendrá dada de la forma:

$$u = T \; U = \begin{bmatrix} T_p & & \\ & \text{el resto de } T_p \text{ submatrices} & \\ & \text{son nulas} & \\ & \cdot & \\ & \cdot & \\ & & T_p \end{bmatrix} U \tag{4.11}$$

Donde el número de matrices T_p es igual al número de nudos del elemento. En el caso de elementos en los que aparezcan giros como grados de libertad, el número de matrices T_p será igual al doble del número de nudos.

4.2.2.4 Condiciones de contorno.

En el análisis de una estructura debemos imponer las condiciones que la ligan con su entorno. Estas condiciones de contorno pueden ser 'esenciales' (giros y desplazamientos), o 'naturales' (fuerzas y momentos). Empleando el método de los desplazamientos, las condiciones de contorno naturales se tienen en cuenta al evaluar el vector de fuerzas exteriores aplicado a los nudos.

Para considerar las condiciones esenciales, pongamos el sistema de ecuaciones de equilibrio (4.7) como:

$$\begin{bmatrix} K_{aa} & K_{ab} \\ K_{ba} & K_{bb} \end{bmatrix} \quad \begin{bmatrix} U_a \\ U_b \end{bmatrix} = \begin{bmatrix} R_a \\ R_b \end{bmatrix} \tag{4.12}$$

Es decir, que el sistema se ha reordenado de forma que U_a son los desplazamientos incógnita, y U_b son los valores conocidos en los apoyos.

Resolviendo (4.12) para U_a tenemos:

$$K_{aa}\ U_a = R_a - K_{ab}\ U_b \tag{4.13}$$

En la ecuación (4.13), para obtener U_a solo son necesarias las matrices de rigidez del ensamblaje correspondientes a los grados de libertad incógnita. En cuanto al vector de cargas R_a, debe modificarse para incluir el efecto de los desplazamientos conocidos cuyo valor sea distinto de cero ($- K_{ab}\ U_b$).

4.2.2.5 Implementación del Método de los Elementos Finitos.

La implementación del método de los elementos finitos se suele dividir en tres partes: a) entrada de datos y preproceso; b) resolución del sistema de ecuaciones, y c) salida de resultados. No obstante, cuando el análisis forma parte de un proceso global de diseño, los procesos de entrada y salida de datos pasan a estar embebidos en la parte más general de comunicación con el usuario; quedando la entrada y salida de información de análisis convertida en accesos a la base de datos correspondiente. Por ello, el problema queda reducido a una primera fase de preproceso y una segunda fase de resolución del sistema de ecuaciones.

La parte de preproceso consiste en tomar secuencialmente la información de cada uno de los elementos y generar la matriz de rigidez del mismo. A continuación se procesa (también secuencialmente) la información de los estados de cargas, para obtener los vectores de cargas correspondientes.

La segunda parte consiste en ensamblar el sistema de ecuaciones planteado en (4.7) (a partir de las matrices de los elementos y los vectores de cargas), y resolverlo numéricamente. Para esta resolución

numérica se puede emplear cualquiera de los métodos conocidos de resolución de sistemas lineales de ecuaciones. No obstante, lo habitual es emplear algún método que aproveche las particularidades del sistema de ecuaciones a tratar: matriz positiva, simétrica, y en banda. Es decir, que lo más ventajoso es triangularizar la matriz de rigidez (aprovechando sus características de simetría, etc.), y realizar después un proceso de sustitución regresiva /6,39/.

Uno de los métodos de triangularización más empleados es el método de eliminación de Gauss; en el que se factoriza la matriz obteniendose la descomposición:

$$K = L^T D L \tag{4.14}$$

Paralelamente, el vector de cargas R sufre un proceso de reducción de la forma:

$$L^T V = R \tag{4.15}$$

Una vez obtenido el vector de cargas reducido V, se calculan los desplazamientos resolviendo por sustitución regresiva el sistema:

$$V = D L U \tag{4.16}$$

Cabe notar que si bién el proceso de reducción puede llevarse a cabo al mismo tiempo que se triangulariza, dado que normalmente se analiza la estructura sometiendola a más de un estado de cargas, se recurre a realizar el proceso de reducción junto con el proceso de sustitución regresiva, en el momento de calcular los desplazamientos para cada estado de cargas.

4.3 EL ANALISIS DE ESTRUCTURAS DENTRO DE UN PROCESO DE DISEÑO

4.3.1 Introducción.

La mayoría de los procesos de optimización son iterativos. Por lo tanto se necesita analizar la estructura despues de cada modificación para actualizar la respuesta, en función de las modificaciones hechas al modelo. Este proceso supone realizar muchos análisis para completar el diseño. Por ello, la forma más rentable de aumentar la eficiencia del proceso de diseño, consiste en estudiar posibles formas de simplificar el proceso de análisis, a fin de disminuir el tiempo de cálculo requerido para realizar cada reanálisis. Se han propuesto diferentes formas de simplificar el análisis, y ninguna de ellas es la

mejor en todos los casos. Todas las aproximaciones existentes se pueden agrupar en tres tipos según Kirsch /40/:

- métodos directos;

- Métodos iterativos, y

- métodos aproximados.

Los dos primeros tipos calculan la solución de (4.7) aprovechando la información que se mantiene constante desde una solución previa; el tercer tipo de métodos está formado por aquellos que buscan la nueva solución por medio de estimaciones a partir de la solución conocida. Este tercer tipo de métodos utilizan un planteamiento totalmente independiente del proceso normal de resolución de (4.7), empleando únicamente la solución de dicho sistema, por lo que resultan ser métodos muy especializados y poco aptos para ser implementados como modificaciones a un flujo de análisis ya existente. Los otros dos tipos de métodos vamos a describirlos a continuación.

4.3.2 Reanálisis por métodos exactos.

Cuando las modificaciones hechas en el modelo afectan a pocos elementos, la forma más rentable de realizar un reanálisis consiste en identificar todas las partes del proceso que no se han modificado, a fin de repetir únicamente los cálculos que han variado /25,39,40/.

Para ello, la ecuación (4.7) la podemos expresar de la forma:

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \end{bmatrix} \qquad (4.17)$$

Es decir, que el sistema se ha reordenado de forma que las ecuaciones que contienen información de los elementos que cambian se han agrupado al principio. De esta forma, la submatriz K_{22} no se modifica desde el diseño anterior al actual

El sistema (4.17) lo podemos descomponer en:

$$K_{11}\, U_1 - K_{12}\, U_2 = R_1 \qquad (4.18)$$

$$K_{21}\, U_1 - K_{22}\, U_2 = R_2 \qquad (4.19)$$

La ecuación (4.19) la podemos reescribir como:

$$U_2 = K_{22}^{-1} (R_2 - K_{21} U_1) \qquad (4.20)$$

Esta ecuación se puede sustituir en (4.18), obteniendose:

$$K_{11} U_1 - K_{12} (K_{22}^{-1} (R_2 - K_{21} U_1)) = R_1 \qquad (4.21)$$

de donde:

$$U_1 = (K_{11} - K_{12} K_{21})^{-1} (R_1 - K_{12} K_{22}^{-1} R_2) \qquad (4.22)$$

En la ecuación (4.22), se observa que una vez realizado el cálculo inicial, K_{22}^{-1} es conocida y constante, por lo que los siguientes análisis solo necesitan invertir una matriz $(K_{11} - K_{12} K_{21})$ que es de orden igual al número de elementos que se modifican.

4.3.3 Reanálisis por métodos aproximados.

En el caso en que entre dos diseños consecutivos haya grán cantidad de elementos que se modifiquen, pero que las modificaciones que sufran sean pequeñas, la distinción entre partes repetibles y no repetibles no es efectiva. Por el contrario, resulta más eficiente recalcular completamente. Ahora bién, como las modificaciones son pequeñas, en lugar de hacer un cálculo riguroso del sistema (4.7), puede ser conveniente hacer un cálculo aproximado modificando la solución anterior.

Además, se debe tener en cuenta que el hecho de que el resultado obtenido en los diseños intermedios no sea exacto, no influye en el comportamiento de muchos algoritmos de optimización. Por otra parte, el cálculo aproximado se puede repetir, de forma iterativa, obteniendo una solución tan próxima a la exacta como se quiera; siempre que el procedimiento aproximado que se elija sea converjente.

Entre los métodos iterativos más empleados está el conocido como 'metodo de Jacobi' /25/. Este método converge a la solución con una relación que depende de la proximidad entre esta y el punto inicial. Por ello es apropiado cuando las diferencias entre dos análisis consecutivos son pequeñas. El método se basa en descomponer la matriz de rigidez de la forma:

$$K = K_b + K_d \qquad (4.23)$$

en donde K_b es una matriz que únicamente difiere de K en que la diagonal principal ha sido reemplazada por ceros, y K_d es una matriz diagonal con los mismos valores que los de la diagonal de K. Entonces, la ecuación (4.7) puede reescribirse como:

$$(K_b + K_d)\ U = R \qquad (4.24)$$

de la cual obtenemos:

$$U = K_d^{-1}\ (R - K_b\ U) \qquad (4.25)$$

A partir de la expresión (4.25), podemos establecer un proceso iterativo para calcular U, de la forma:

$$U_{k+1} = K_d^{-1}\ (R - K_b\ U_k) \qquad (4.26)$$

Teniendo la ventaja de que la matriz a invertir K_d, es una matriz diagonal; por lo que los cálculos son mucho más sencillos.

4.4 ANALISIS DE SENSIBILIDADES

4.4.1 Introducción.

En un sentido amplio, podemos decir que el análisis de sensibilidades para el diseño de estructuras trata de la relación entre las variables de diseño y las variables de estado (o respuesta) de la estructura. La dependencia de las variables que miden la respuesta de la estructura (desplazamientos, tensiones, etc) con las variables de diseño (áreas de la sección recta, etc) está implícita en las ecuaciones de estado. En este sentido, es de destacar que en el año 1983 apareció una primera versión de cálculo de sensibilidades incluida en uno de los programas de cálculo por elementos finitos más ampliamente usados: el NASTRAN. Es decir, que se empieza a considerar el cálculo de sensibilidades como una parte más del análisis, en lugar de como unos cálculos previos para un proceso de optimización.

Para el cálculo de sensibilidades coexisten dos aproximaciones totalmente diferentes:

- los métodos numéricos basados en diferencias finitas, y

- los métodos analíticos basados en la formulación matemática.

En la mayor parte de las aplicaciones, los métodos analíticos son mucho más eficientes. Sin embargo, debido al esfuerzo requerido para

implementarlos (básicamente debido a su falta de generalidad), coexiste con los menos eficientes (pero más generales y sencillos) métodos numéricos. Fleury apunta que a largo plazo, es previsible el predominio de los métodos analíticos /23/. A continuación vamos a estudiar con más detalle ambas aproximaciones.

4.4.2 Derivadas por diferencias finitas.

El cálculo de derivadas por diferencias finitas consiste básicamente en analizar la estructura para un valor dado de las variables de diseño, y repetir el análisis para un nuevo conjunto de variables de diseño que resultan de perturbar ligeramente las variables iniciales. La diferencia entre los valores de las variables de estado obtenidos en ambos casos dan la medida de la sensibilidad de las mismas frente a los cambios en las variables de diseño /25,29/.

La formulación básica de este método parte de la conocida expresión de la diferenciación:

$$\delta\phi/\delta X_j = \frac{\phi(X^j) - \phi(X)}{\Delta X_j} \qquad (4.27)$$

En donde ϕ es la función que queremos derivar; X es el vector de las n variables de diseño; ΔX_j es un incremento finito de la j-esima variable, y $X^{jT} = (X_1, X_2, \ldots\ldots, X_j+\Delta X_j, \ldots\ldots, X_n)$. Por tanto, el método requiere calcular el diseño perturbado por un incremento finito. Esta forma de cálculo es la que se conoce como 'diferencias finitas hacia adelante'.

El error que se comete utilizando este método puede ser relativamente grande. Dicho error se denomina 'error de truncamiento' porque se puede medir comparando la expresión (4.27), con el valor de $\delta\phi/\delta X_j$ que se obtiene despejando de la expresión del desarrollo en serie de Taylor para la función ϕ /25/. Para reducir dicho error, se debe disminuir la variación de las variables X_j. Cuando, pese a todo, no se consigue suficiente precisión, la solución consiste en modificar la expresión (4.27), obteniendo las denominadas 'diferencias finitas centrales':

$$\delta\phi/\delta X_j = \frac{\phi(X^{j+}) - \phi(X^{j-})}{\Delta X_j} \qquad (4.28)$$

En donde $X^{j+}=X^j$, y $X^{j-T} = (X_1, X_2, \ldots\ldots, X_j-\Delta X_j, \ldots\ldots, X_n)$.

Comparando el resultado obtenido en (4.28), con el correspondiente al del desarrollo en serie, se concluye que esta formulación es más exacta que el de las diferencias finitas hacia adelante. A cambio, en (4.28) se puede observar que se necesitan dos evaluaciones de la función. es decir, que el proceso descrito arriba se tiene que repetir también para X_j^- (mientras que en el caso anterior empleabamos el valor sin perturbar X).

En cualquier caso, la principal ventaja de este método consiste en que se puede acomodar fácilmente al proceso de análisis por elementos finitos. En líneas generales, basta con añadir en el cálculo por elementos finitos unos estados de carga ficticios (llamados 'seudocargas'). Por contra, el principal inconveniente de este método es el elevado número de cálculos que requiere; especialmente cuando crece el número de variables de diseño.

Una mejora de este método es un método semianalítico que puede resumirse como:

a) Se modifica ligeramente cada una de las variables de diseño sucesivamente; regenerando un nuevo modelo de la estructura que se adapte a dichas modificaciones.

b) Se calculan las matrices de rigidez de cada uno de los elementos (tantas veces como variables de diseño haya).

c) se obtienen las matrices globales ensamblando las matrices 'perturbadas' de los elementos.

d) Se calculan los vectores de 'seudocargas'. Estos vectores se obtienen como la diferencia entre el vector que resulta de multiplicar los desplazamientos originales por la matriz de rigidez original, menos el que resulta de multiplicar los desplazamientos originales por la matriz de rigidez perturbada.

e) Se obtienen las derivadas respecto a los desplazamientos resolviendo la estructura para estas seudocargas.

f) Se obtiene las derivadas respecto a las tensiones calculando valores aproximados de las mismas a partir de la resolución de la estructura perturbada, y usando los desplazamientos obtenidos para esta solución.

4.4.3 Derivadas analíticas.

En muchos problemas de optimización se pretende minimizar una función del peso de la estuctura, sometida a restriciones de desplazamiento, tensión y valores admisibles de las variables. En este caso, cualquiera de estas funciones se puede expresar de la forma:

$$\phi = \phi(X.U) \tag{4.29}$$

Es decir, como funciones de las variables de diseño (X), y de la respuesta de la estructura (U).

Si consideramos el sistema de ecuaciones de equilibrio después de aplicar las condiciones de contorno (4.7), lo podemos expresar en la forma resumida:

$$K(X)\ U = R(X) \tag{4.30}$$

Es decir, que tanto la matriz de rigidez global, como los vectores de cargas son funciones de las variables de diseño /32/. Por lo tanto, el problema que consideramos es el caso en el que la respuesta de la estructura es lineal respecto a las variables de estado, una vez que se han fijado las variables de diseño. Sin embargo, como las variables de diseño aparecen en los coeficientes de los operadores lineales, las ecuaciones de estado no son funciones lineales respecto a dichas variables de diseño. Por ello, el reto matemático consiste en tratar el problema no lineal del análisis de sensibilidades de diseño, con métodos que utilicen las ventajas matemáticas de las propiedades lineales (para un diseño fijo) de las variables de estado.

Para llegar a una formulación que nos permita resolver el problema, nos vamos a apoyar en el hecho de que la matriz de rigidez es definida positiva (y por tanto no singular). Y vamos a asumir que todas las componentes de K(X), y R(X) son continuamente diferenciables respecto a las variables de diseño. Entonces, usando la regla de la cadena, podemos derivar la expresión (4.29) respecto a una cualquiera de las variables de diseño:

$$d\phi/dX_j = \delta\phi/\delta X_j + \delta\phi/\delta U\ dU/dX_j \tag{4.31}$$

Y también podemos derivar la ecuación (4.30):

$$K(X)\ dU/dX_j = -\delta K(X)/\delta X_j\ U + \delta R(X)/\delta X_j \tag{4.32}$$

Puesto que la matriz K es no singular, podemos despejar en la ecuación (4.32):

$$dU/dX_j = K^{-1}(X) \left[\delta R(X)/\delta X_j - \delta K(X)/\delta X_j \; U\right] \qquad (4.33)$$

Sustituyendo la expresión (4.33), en la (4.31), se obtiene:

$$d\phi/dX_j = \delta\phi/\delta X_j + \delta\phi/\delta U \; K^{-1}(X)\left[\delta R(X)/\delta X_j - \delta K(X)/\delta X_j \; U\right] \qquad (4.34)$$

La utilidad práctica de la ecuación (4.34) es nula, dado que es inviable obtener la inversa de la matriz de rigidez. Por tanto, quedan dos alternativas:

- resolver numéricamente la ecuación (4.32) (obteniendo dU/dX_j),y sustituirla en (4.31), y
- definir unas variables complementarias que modifiquen el problema.

Las variables complementarias del segundo método se definen como:

$$\lambda = \left[\delta\phi/\delta U \; K^{-1}(X)\right]^T = K^{-1}(X) \; (\delta\phi/\delta U)^T \qquad (4.35)$$

Multiplicando por K(X) los dos miembros de la ecuación (4.35) se obtiene:

$$K(X) \; \lambda = (\delta\phi/\delta U)^T \qquad (4.36)$$

Del sistema (4.36) se puede calcular λ sin necesidad de invertir la matriz, y el resultado se sustituye en la ecuación:

$$d\phi/dX_j = \delta\phi/\delta X_j + \lambda \left[\delta R(X)/\delta X_j - \delta K(X)/\delta X_j \; U\right] \qquad (4.37)$$

que resulta de sustituir (4.35) en (4.34).

De la expresión (4.36) puede observarse que el método de las variables adjuntas es en realidad un método de seudocargas, tal como los utilizados para las derivadas por diferencias finitas.

Ambos métodos requieren la resolución de un sistema de ecuaciones. Sin embargo, en función de las características del problema alguno de los dos es más apropiado que el otro. El primero de los dos métodos es el que se conoce como 'diferenciación directa', y es apropiado cuando el número de variables sea inferior al número de funciones a derivar. Por el contrario, el segundo método, conocido como 'método de las variables adjuntas', es apropiado cuando el número de variables es superior al de funciones.

Es importante destacar que la matriz K(X) ya ha sido factorizada para resolver la ecuación (4.31), para obtener los desplazamientos. Por lo tanto, la resolución de (4.33) para encontrar dU/dX_j; o la resolución de (4.37) para encontrar , es mucho más rápida

CAPITULO 5

REPRESENTACION GRAFICA

5.1 INTRODUCCION

La representación gráfica es la forma más efectiva para que el ser humano asimile cualquier tipo de información. Las características de la mente humana hacen que seamos capaces de percibir y procesar muy rápidamente información muy variada cuando se nos presenta gráficamente. Esto es especialmente cierto en la comunicación de tipo técnico, debido a que los gráficos permiten transmitir información de relaciones y magnitudes de forma exacta y resumida. De hecho, los dibujos son fundamentales, como medios de visualización y comunicación, en casi todo tipo de trabajos técnicos. Los dibujos ayudan a obtener resultados de mejor calidad y menor costo, al aumentar la eficiencia de las comunicaciones entre las personas implicadas en desarrollos técnicos.

Para que la representación gráfica sirva como medio de comunicación se debe establecer una "semántica" gráfica. Es decir, que se deben definir unas reglas que conduzcan a que la representación gráfica que se realice sea la mejor forma de transmitir la información. El criterio de que es lo mejor se establece en base a dos objetivos que debe alcanzar la representación: a) ser de fácil asimilación para la mente humana, y b) tener una interpretación unívoca.

Estas dos características nos llevan a plantear el estudio de la representación gráfica desde dos puntos de vista: por una parte debemos conocer las reglas generales de la percepción de la mente humana; por otra parte se debe estudiar que forma de representación es la más efectiva para cada tipo de información. El comienzo de este capítulo está dedicado a introducir estos dos aspectos del estudio. El propósito perseguido es el de resaltar únicamente aquellos aspectos que se han utilizado en el desarrollo de este trabajo. La segunda parte del capítulo desarrolla los aspectos de la representación que conviene que el usuario del sistema conozca mínimamente; pues este

conocimiento le puede permitir utilizar el sistema de forma más sofisticada.

5.2 REGLAS DE LA REPRESENTACION GRAFICA.

La psicología de la percepción ha intentado medir la habilidad del ser humano para discriminar formas y colores a fin de establecer una "gramática" de la representación gráfica. En nuestro caso, el objetivo perseguido es establecer una serie de reglas que debe cumplir una representación para tener una interpretación sencilla y única. Por lo que debemos utilizar esta gramática gráfica de forma que se consiga la máxima claridad de interpretación de la información, con el mínimo "ruido" posible. Para conseguir este tipo de representación vamos, por una parte, a diferenciar la representación en un conjunto de "variables visuales", y por otra parte vamos a determinar de que forma podemos visualizar la (o las) relaciones entre las unidades de información.

Las variables visuales de una representación se pueden definir como partes de la representación que pueden variar para transmitir información diferente. Desde nuestro punto de vista, las podemos diferenciar según el criterio de conocer la influencia que cada una tiene en la percepción del conjunto. Así, se puede determinar la forma de emplear estas variables para conseguir el efecto deseado. Bertin /11/, habla de ocho 'variables visuales':

-las dos dimensiones del papel;

-tamaño y valor;

-forma;

-orientación;

-color, y

-textura.

De esta división, las variables de dimensión, tamaño y valor son las que mejor se adaptan para generar la representación gráfica de la información. Por el contrario, el resto solo son apropiadas para separar las unidades de información contenidas en la representación.

El segundo paso consiste en establecer una relación entre la información a transmitir y las variables visuales. Conseguir que la relación sea unívoca es una tarea relativamente sencilla. Por el contrario, conseguir que el resultado tenga una interpretación fácil resulta bastante más complejo. A pesar de los estudios realizados, se trata de un área en donde la habilidad sigue siendo un factor importante para obtener resultados efectivos. No obstante se pueden lograr representaciones satisfactorias ateniendose a las formas de representación que la práctica ha decantado como las más apropiadas para cada tipo de información.

Un aspecto de las representaciones gráficas que cabe resaltar es la importancia de no cometer "errores gramaticales" /60/. Dos son los tipos de errores que se pueden cometer. En primer lugar, la excesiva ornamentación o los pequeños errores de trazado (faltas de alineamiento, saltos, etc.) distraen sobremanera la atención del observador (fig. 5.1). También hay que evitar efectos inesperados en la interpretación de los gráficos, debidos a asociaciones de las imagenes representadas con imagenes utilizadas en otras aplicaciones con distinto significado (fig. 5.2).

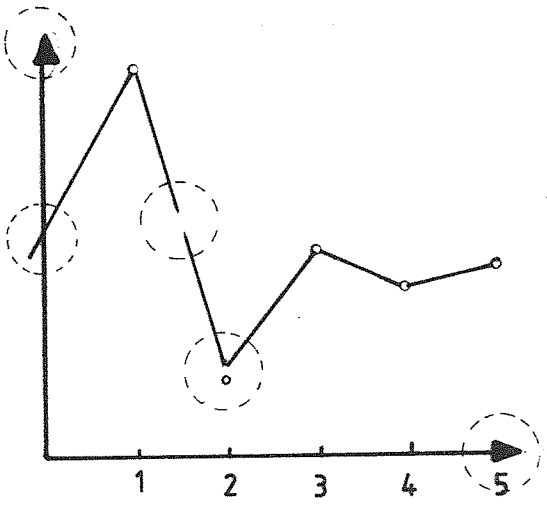

Figura 5.1. Elementos que distraen la atención.

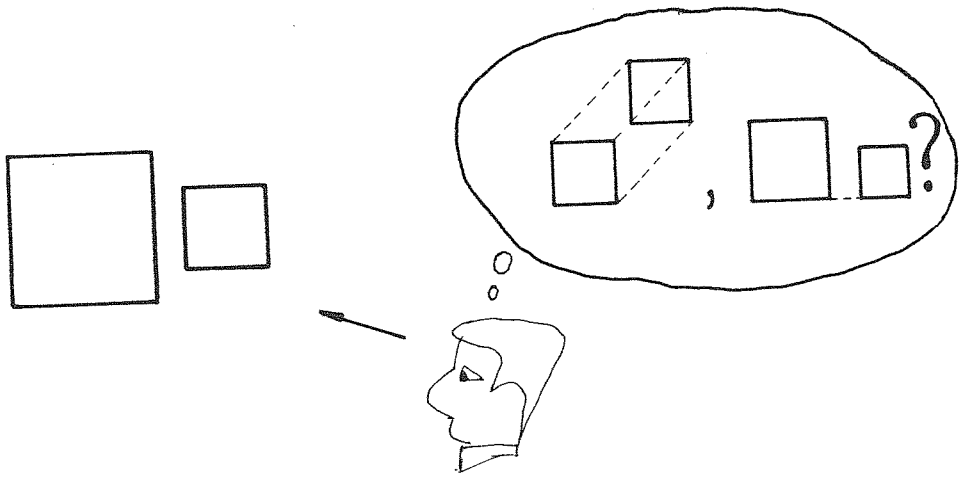

Figura 5.2. Interpretación confusa de un símbolo.

Como consecuencia de esto, un aspecto de la presentación gráfica en un ordenador que debe estudiarse con detalle, es el de la divisón de la pantalla. En palabras de Newman y Sproull /60/, la escasez de espacio en la pantalla se exacerba a menudo por la necesidad de acomodar menús, mensajes y otros objetos sobre la misma. Por ello, se debe encontrar el sitio apropiado para cada tipo de información que debe coexistir en la pantalla. Esto se debe hacer en función de su importancia (zona central para lo más importante, y laterales para la información complementaria), y de sus características (necesidad de un hueco cuadrado, alargado, etc).

5.3 FORMAS DE REPRESENTACION.

Una primera distinción entre formas de representación nace de la naturaleza de la información a representar. Así, se distingue entre:

- representación de un objeto para transmitir sus características (aspecto, forma, dimensiones, etc.), y

- representación de dos, o más, informaciones relacionadas para hacer patente su relación (de semblanza, de orden o de proporcionalidad).

La representación de objetos tridimensionales sobre las dos dimensiones del papel (o la pantalla) se lleva a cabo por medio de un

proceso en dos fases: la idealización del objeto, y la proyección de esta idealización. Las diferentes formas de proyección definidas se adaptan a distintas necesidades de transmisión de información. Así, informaciones cualitativas de objeto se transmiten por medio de proyecciones semejantes a la visión humana; mientras que cuando se pretende transmitir información de dimensiones del objeto se recurre a sistemas en donde resulta cómodo medir.

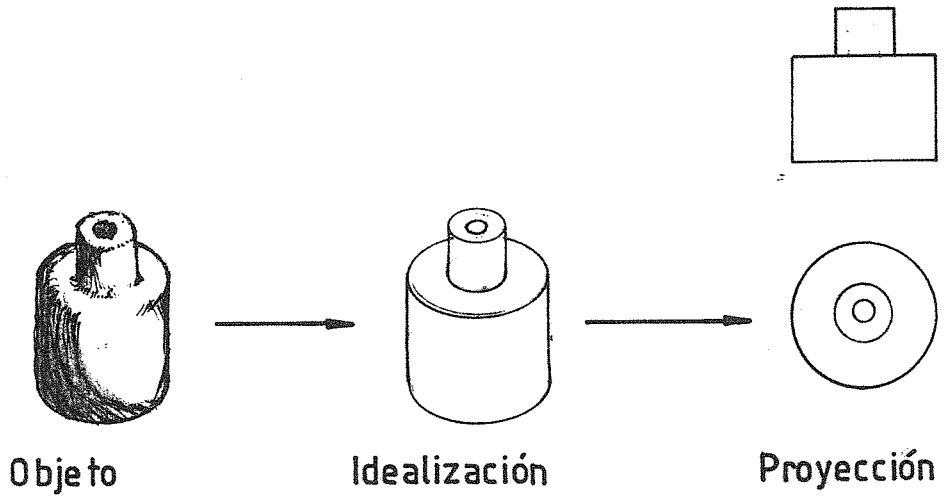

Figura 5.3. Representación de objetos: proyecciones.

Para representar relaciones entre conjuntos de datos se utilizan los diagramas. Estos son representaciónes gráficas de dos o más conjuntos de información, y de las relaciones que se establecen entre ellos. Hay dos formas básicas de representar estas relaciones:

- representar los elementos de cada conjunto por medio de puntos, y su relación por medio de líneas.

- representar los elementos por líneas y su relación por puntos.

Las dos formas de representación se pueden ver en la figura (5.4). En dicha figura puede observarse que la primera es una gráfica para "leer", mientras que la segunda es una gráfica para "ver". Es decir, la información contenida en la primera gráfica debe extraerse, en tanto que la de la segunda se observa al primer golpe de vista. Por otra parte, la segunda gráfica seguiría siendo igualmente simple aunque aumentasen los elementos de los conjuntos. Mientras que la primera se iría haciendo más confusa.

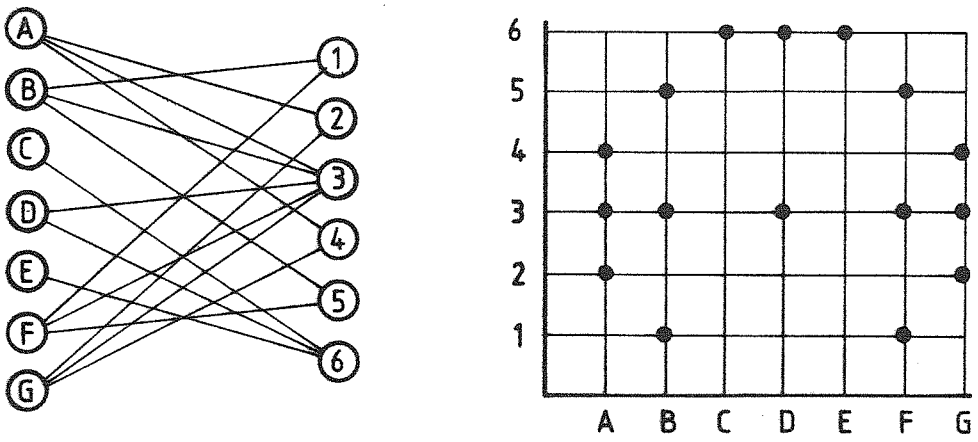

Figura 5.4. Representación de relaciones: Diagramas.

Por ello, la tabla de doble entrada es, en general, la mejor forma de representar relaciones entre conjuntos de información. Si bién, en algunos casos especiales se han desarrollados formas de representación mejor adaptadas al problema a tratar (p.e. en el caso de los mapas, donde la relación es de tipo distancia, basta copiar dicha relación sobre el papel a la escala apropiada).

Una ampliación habitual de los diagramas es la de unir los puntos por medio de líneas, para añadir la información de evolución o secuencia que suele acompañar a cierto tipo de información.

5.4 REPRESENTACION DE OBJETOS.

Para representar objetos reales (tridimensionales) en dos dimensiones, deben seguirse dos pasos /9/:

- Idealizar el objeto, y

- Proyectar esta idealización en dos dimensiones.

5.4.1. Idealización.

El proceso de idealización es la búsqueda de un compromiso entre el grado de similitud requerido entre el objeto y su modelo por una parte, y el grado de dificultad de manipulación del modelo inherente a la cantidad de información que contenga. En definitiva, se trata de diferenciar entre lo que pueden considerarse "detalles" del objeto, y lo que son sus aspectos fundamentales. Para eso, se debe decidir que información del modelo es básica, y cual prescindible, en función de la aplicación a que se destine.

Una forma de encontrar el modelo apropiado puede ser la de intentar definir el objeto con modelos de complejidad creciente (partiendo del modelo más sencillo), hasta llegar a tener definidos todos los aspectos que interesen. La forma más sencilla de definir el modelo que idealiza un cuerpo tridimensional es la de simular su contorno por medio de un entramado de segmentos (como las lineas de una fina malla que se apoyase sobre el objeto). Se obtiene así un modelo que puede manejarse con comodidad, en la conversión de tres a dos dimensiones. La sencillez de este modelo lo hace apto para representar de forma aproximada, pero muy rápida, cualquier objeto. Por otra parte, el modelo se adapta perfectamente a cierto tipo de objetos (tal es el caso de las estructuras de barras). Así, este modelo, que se conoce como modelo alámbrico, queda definido por:

-Un conjunto de puntos (a los que llamaremos nudos), situados en un espacio tridimensional por medio de sus coordenadas respecto a un sistema dado;

-Un conjunto de segmentos definidos por los dos nudos que conectan.

Definiendo un tercer conjunto de entes (polígonos, planos y opacos, dados por los sucesivos nudos que forman sus vértices), obtendremos un 'modelo de fronteras'. Este modelo es una primera aproximación para representar objetos sólidos tridimensionales, y se adapta perfectamente a objetos con contornos planos (p.e. los elementos de cualquier modelo de análisis por el método de los elementos finitos). Además es un modelo suficientemente fácil de calcular /24/.

5.4.2 Proyección.

La forma de llevar a cabo la proyección no es única; lo que da lugar a diferentes sistemas de representación. Las alternativas son:

-Utilizar uno o más planos de proyección.

-Utilizar un haz de proyección de rayos paralelos (perpendiculares o no al plano de proyección), o de rayos concentricos.

De todas la posibles combinaciones, la que da un resultado más parecido a la visión humana el la de emplear un solo plano de proyección, sobre el que incide un haz de rayos concéntricos. Esta forma de proyección se conoce como 'perspectiva cónica'. Se denomina perspectiva por utilizar un solo plano de proyección, y cónica por serlo el haz de proyección. Por tanto, se proyecta un objeto sobre un plano, llamado 'plano del cuadro' (PC), por medio de un haz de rayos concéntricos en un punto, llamado 'punto de vista' (PV). El PC se situa entre el objeto y el PV, de forma que resulte perpendicular a un punto del espacio que llamamos 'punto de mira'(PM), y que junto con el punto de vista define el eje del cono formado por el haz. Un esquema de esta forma de proyección puede verse en la figura (5.5).

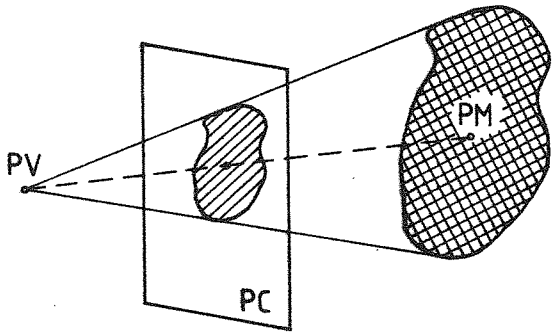

Figura 5.5. Perspectiva cónica.

Una ventaja adicional de este sistema de proyección es que basta llevarse el punto de vista a un punto impropio para obtener una proyección cilíndrica (haz de rayos paralelos). Y por tanto, girando la linea PV-PM en ángulos rectos, se puede obtener cualquiera de las

seis vistas en diédrico (vistas habituales en las representaciones técnicas).

Puesto que lo que se representa es una porción del plano del cuadro, se deben definir más parámetros para poder generar la proyección. Tal como puede verse en la figura (5.6), se definen los 'semiángulos de visión' como los ángulos que forman los rayos verticales que inciden en los extremos superior e inferior de la parte vista del PC. Se define, así mismo, la relación altura/anchura del rectangulo visto del PC. De esta forma se define una 'pirámide de visión', así como el tamaño aparente de la proyección (la relación entre el tamaño de la proyección y el del PC).

Definiendo, por último, la distancia desde el PM hasta el PC (DPC) se delimita el 'volumen observado'.

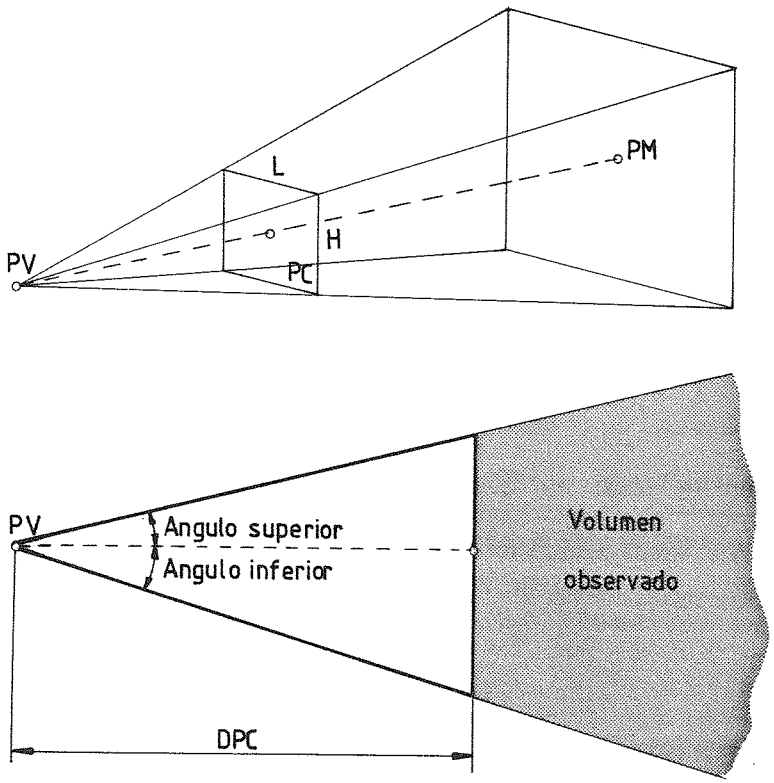

Figura 5.6. Parametros de representación en prespectiva cónica.

5.4.3. Implementación de la proyección.

Las coordenadas de los puntos del modelo están, generalmente, dadas en un sistema "comodo" para definir el modelo, o para situarlo en un entorno físico apropiado. A este sistema lo llamaremos 'sistema implícito'.

Para que el cálculo de la perspectiva resulte sencillo, es conveniente utilizar un sistema de coordenadas (cartesiano, ortogonal y destrógiro) con origen en PM y eje Z en dirección y sentido PM-PV. A este sistema lo llamaremos 'sistema del punto de vista'. Para definir unívocamente este sistema, aún se puede plantear una condición más. Así, impondremos que el segundo eje sea perpendicular a alguno de los ejes del sistema implícito. De esta forma, podremos conseguir que la perspectiva no pierda la relación arriba/abajo del objeto representado.

Definimos también un tercer sistema de coordenadas (en este caso bidimensional), de forma que el origen esté en el punto de intersección del PC con la línea PV-PM, y que sus ejes X e Y sean paralelos a los respectivos ejes del sistema del punto de vista. A este sistema lo llamaremos 'sistema del plano del cuadro'.En la figura (5.7). pueden verse los tres sistemas definidos.

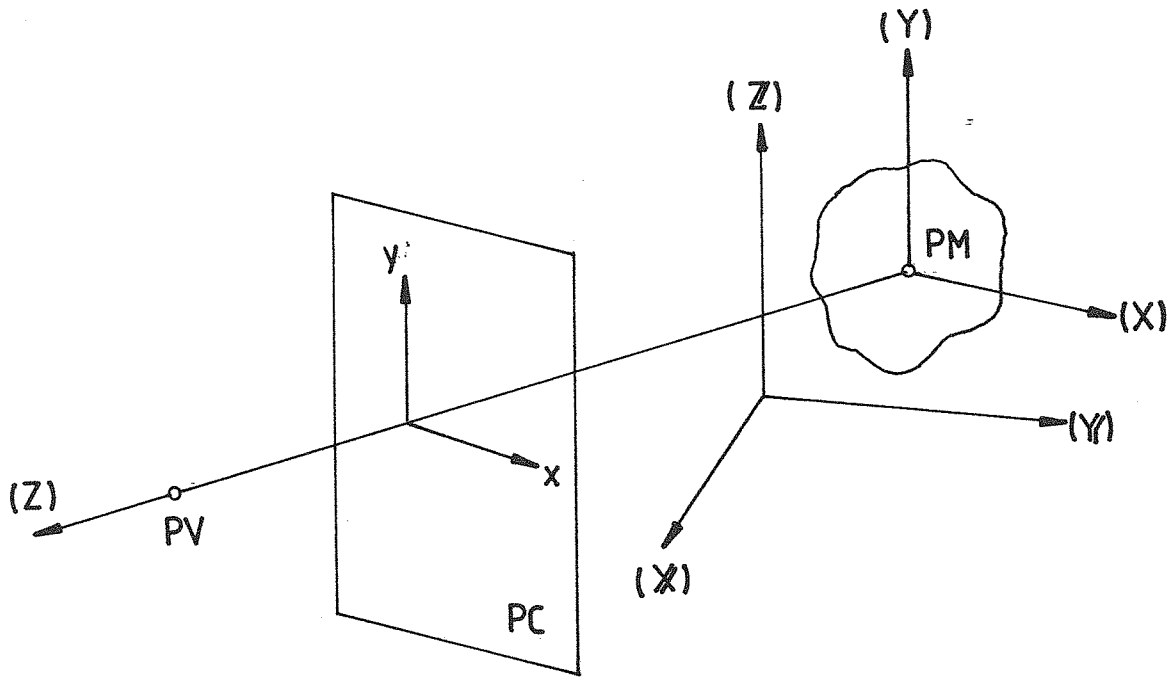

Figura 5.7. Sistemas de coordenadas.

Por la forma de definir los sistemas de coordenadas, una vez tenemos el modelo referido al sistema del punto de vista (haciendo un simple cambio de coordenadas), obtener su proyección sobre el plano del cuadro se limita a aplicar a las coordenadas X e Y de cada punto la regla de proporcionalidad que se observa en la figura (5.8).

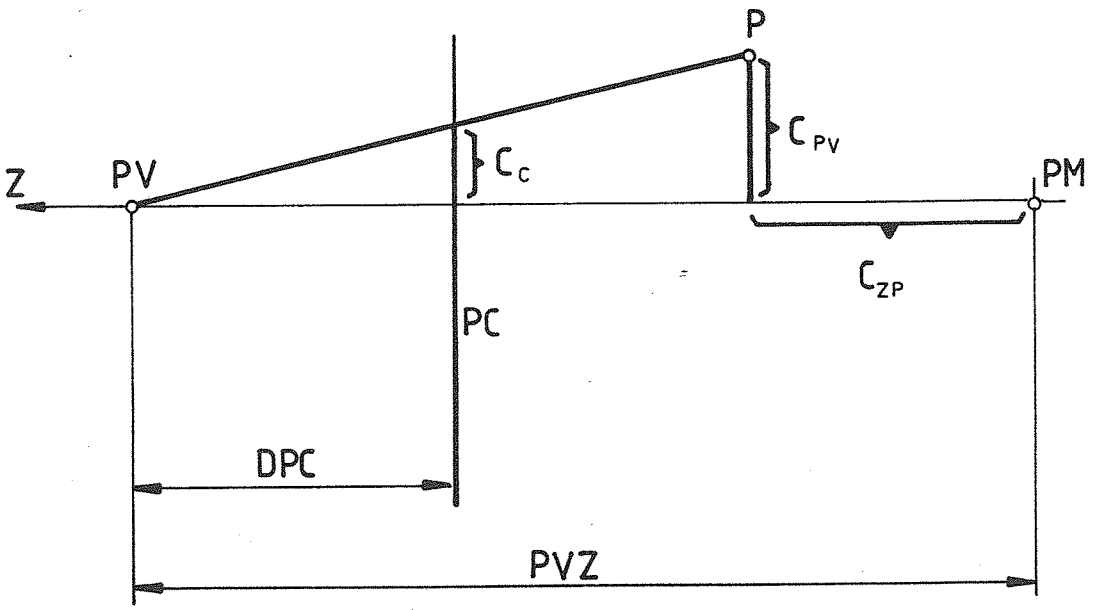

Figura 5.8. Paso del sistema del punto de vista al sistema del plano del cuadro.

En donde:

C_{ZP} es la coordenada Z, del punto P, respecto al sistema del punto de vista.

C_{PV} es la coordenada X ó Y, del punto P, respecto al sistema del punto de vista.

C_{PC} es la coordenada X ó Y de la proyección del punto P, respecto al sistema del plano del cuadro.

Para determinar que puntos quedan fuera del volumen observado se puede dejar simplemente que la rutina de recortado ("clipping") los elimine. Hay, sin embargo, una excepción: los puntos que quedan delante del plano del cuadro. Estos puntos deben eliminarse previamente porque el

recortado actua sobre la proyección bidimensional, no sobre el modelo tridimensional. Para eliminarlos, basta comprobar:

$$C_{zP} \leq PVZ - DPC \qquad (5.1)$$

Para obtener las líneas proyectadas, basta conectar con líneas rectas las proyecciones de los puntos que conectan.

En el caso de que los dos puntos queden fuera del volumen de trabajo, la línea no se trazará. En el caso de que la línea quede en parte dentro del volumen de trabajo, se puede recurrir de nuevo a la rutina de recortado; salvo en el caso de que la línea atraviese el plano del cuadro. En este caso, calcularemos el punto de corte de la línea con el plano del cuadro, y lo utilizaremos como extremo del segmento visible, en lugar del punto que queda fuera del volumen observado (fig. 5.9). El cálculo puede resumirse definiendo $C_{xy}(K)$ como la coordenada X ó Y de la proyección del punto K, respecto al sistema del plano del cuadro. Entonces, de la figura 5.9 puede deducirse:

$$C_{xy}(II) = C_{xy}(J) + L_J \qquad (5.2)$$

siendo:

$$L_J = D_J \, L_{IJ} \, / \, (D_I + D_J) \qquad (5.3)$$

Y, a su vez:

$$D_J = PVZ - DPC - C_{zJ} \qquad (5.4)$$

En donde C_{zJ} es la coordenada Z, del punto J, respecto al sistema del punto de vista.

Si llamamos punto I al que está delante del PC, y J al que está detrás. D_J será siempre positivo; igual que D_I+D_J. Por tanto, el signo de L_J depende del valor de L_{IJ}, que viene dado por la expresión:

$$L_{IJ} = C_{xy}(I) - C_{xy}(J) \qquad (5.5)$$

Por último:

$$D_I + D_J = C_{zI} - C_{zJ} \qquad (5.6)$$

Y, sustituyendo (5.3) a (5.6) en (5.2), obtenemos:

$$C_{xy}(II) = C_{xy}(J) + \frac{(PVZ - DPC - C_{zJ}) \, (C_{xy}(I) - C_{xy}(J))}{(C_{zI} - C_{zJ})} \qquad (5.7)$$

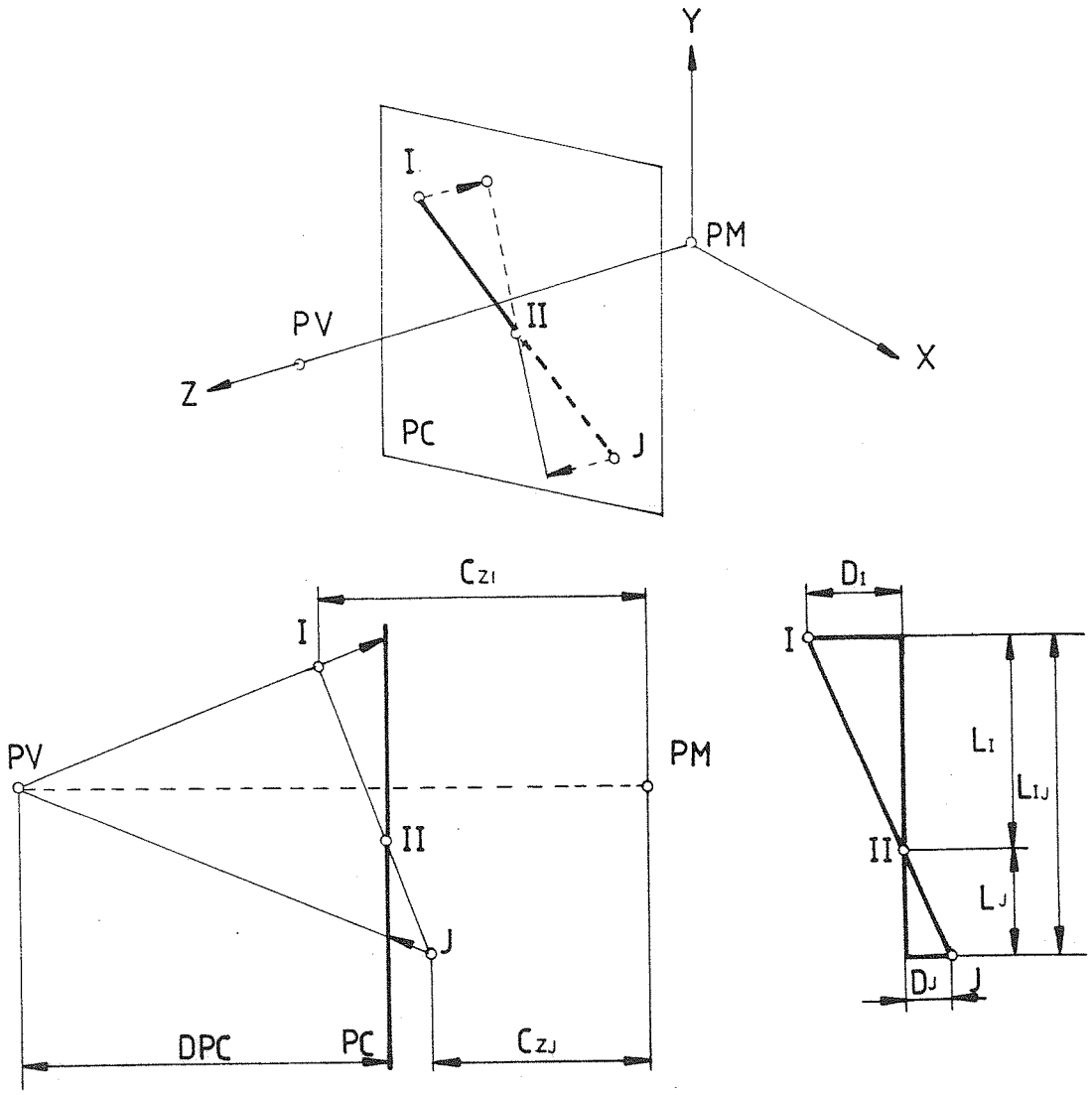

Figura 5.9. Determinación del segmento visto.

5.5 REPRESENTACION DE RELACIONES.

5.5.1 Introducción.

Como ya hemos comentado, la representación de relaciones intenta visualizar de forma gráfica las relaciones existentes entre varios conjuntos de información. Por lo tanto, se debe encontrar la forma más apropiada para representar, por una parte los conjuntos de información, y por otra las relaciones que los ligan. Tal como se ha dicho en el apartado 5.3, existen representaciones particulares que se adaptan muy bién a cada tipo de relaciones. No obstante, la más general de estas representaciones es el diagrama. También se ha indicado que el tipo de diagrama más apropiado para representar las

relaciones consiste en representar los conjuntos de información por medio de líneas y sus relaciones por medio de puntos. Es decir, que se define un sistema de coordenadas sobre un espacio de tantas dimensiones como conjuntos de información se tengan más uno. Sobre este sistema de coordenadas se representa cada uno de los conjuntos de información, dejando una de las coordenadas para representar el valor de la relación. De esta forma, cada punto de dicho espacio representa una posible relación. Si puede definirse algún tipo de orden para los conjuntos de información y para las posibles relaciones, el espacio sobre el que se define el sistema de coordenadas, se puede hacer uso de los espacios métricos definidos en la geometría analítica. Esta forma de representación se adapta particularmente bién al caso de representación de funciones.

A continuación vamos a describir las diferentes formas que puede adoptar un diagrama. La descripción se particularizará al caso de las funciones dado que las relaciones a representar en un proceso de diseño pueden expresarse siempre en dicha forma. También se va a estudiar el caso en que se pretenden representar diferentes relaciones sobre los mismos conjuntos de información. Pues es un caso de especial interés en el proceso de diseño, en el que se pretenden comparar diferentes comportamientos para tener una idea global de la evolución del proceso. Después se introducirán los aspectos generales de la implementación de este tipo de representaciones. Dado que la representación de relaciones se debe llevar a cabo en dos dimensiones (pantalla o papel), y por comodidad en la descripción, esta se hará para el caso particular en el que solo haya un conjunto de información (funciones de una variable). Por lo que al final se extenderá la explicación al caso de dos o más conjuntos de información.

5.5.2 Diagramas.

El diagrama habitual para representar información no ordenada es el de barras (fig. 5.10); mientras que si la información está ordenada, se recurre a los diagramas de polilinea (fig. 5.11). De esta forma, se añade la información de evolución o secuencia que implica el orden /16/.

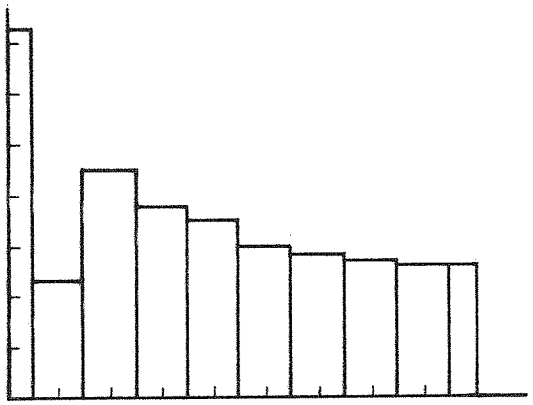

Figura 5.10. Histograma.

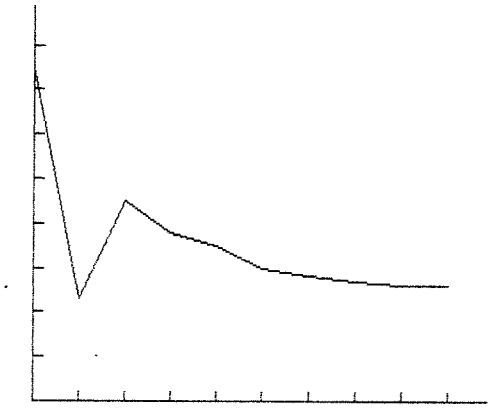

Figura 5.11. Diagrama de polilínea.

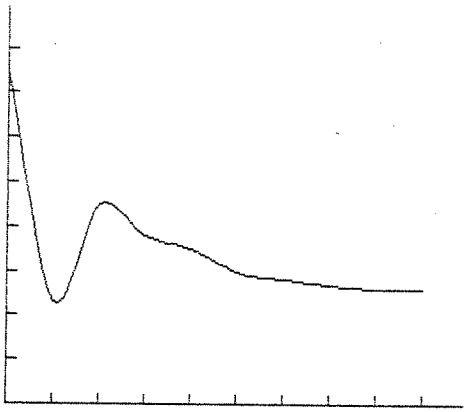

Figura 5.12. Diagrama de curva suave.

Por tanto, este tipo de diagramas son apropiados para representar funciones de una variable. Los diagramas de columnas son apropiados para representar funciones discontinuas. Por el contrario, los diagramas de línea continua resultan más apropiados para representar funciones continuas.

Para las funciones continuas, generalmente, no se conocen más que un conjunto de valores a intervalos (normalmente regulares) de la variable, la línea que representa la evolución suele ser la polilínea que resulta de unir con tramos rectos cada dos puntos consecutivos. Por motivos estéticos, se puede recurrir a una línea curva ajustada a los puntos conocidos (p.e. un spline) a fin de "suavizar" la representación (fig. 5.12). Pero, como la interpolación lineal se obtiene más rápidamente y da una información más ajustada a la conocida, es la empleada habitualmente para transmitir información técnica.

5.5.3 Representación de varias relaciones.

Para que la representación de las magnitudes que intervienen en el diseño sea utilizable para reconducir este, es fundamental poder comparar variaciones de diferentes magnitudes. Por tanto, debemos recurrir a representaciones de varias funciones sobre el mismo diagrama. En las figuras (5.13) y (5.14) pueden verse ejemplos de los dos tipos de representaciones.

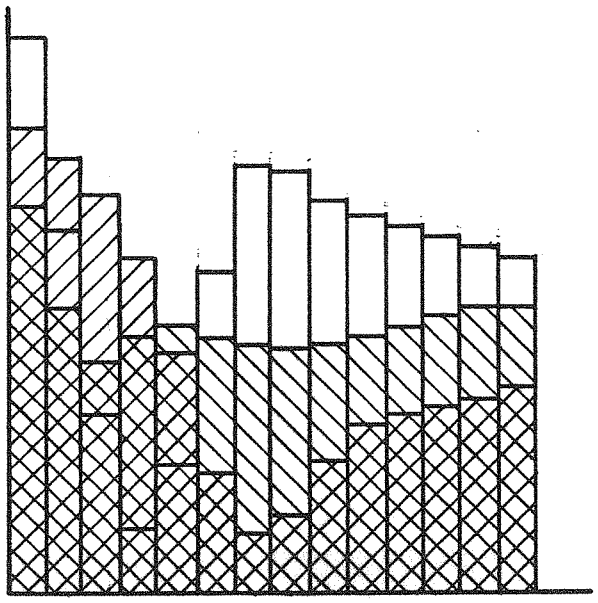

Figura 5.13. Histogramas superpuestos

La representación superpuesta de varios histogramas puede llegar a resultar confusa. Por el contrario, para representar varias lineas continuas superpuestas basta recurrir a diferentes tipos de líneas. Esta es otra de las razones que hacen preferible, para este problema, la representación por lineas continuas frente a los histogramas.

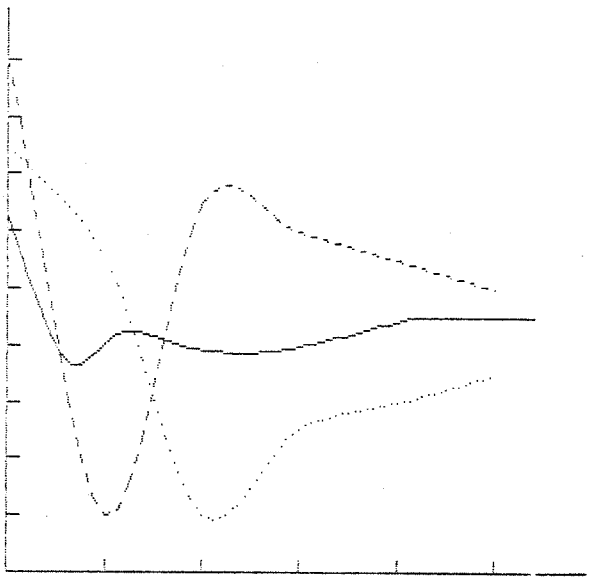

Figura 5.14. Lineas continuas superpuestas

El mayor inconveniente de esta representación es que obliga a que todas las funciones varien en el mismo rango. En caso contrario, se puede recurrir a afectar a las funciones a representar de una escala, o a representar los diferentes ejes. En la figura (5.15) puede verse que la representación de diferentes ejes es una alternativa válida cuando se trata de dos, o a lo sumo tres. La solución de 'escalar' las magnitudes a representar para que encajen en el rango de variación del eje elegido previamente, tiene el inconveniente de que las escalas deben elegirse cuidadosamente para evitar complicar excesivamente la lectura del diagrama. Por el contrario, tiene la ventaja de que se pueden representar, de forma legible, hasta seis o siete funciones al mismo tiempo.

Una ventaja adicional de la opción de escalar, es que permite introducir de forma cómoda el escalado que sufren las variables de diseño en ciertos algoritmos, a fin de mejorar el funcionamiento de los mismos.

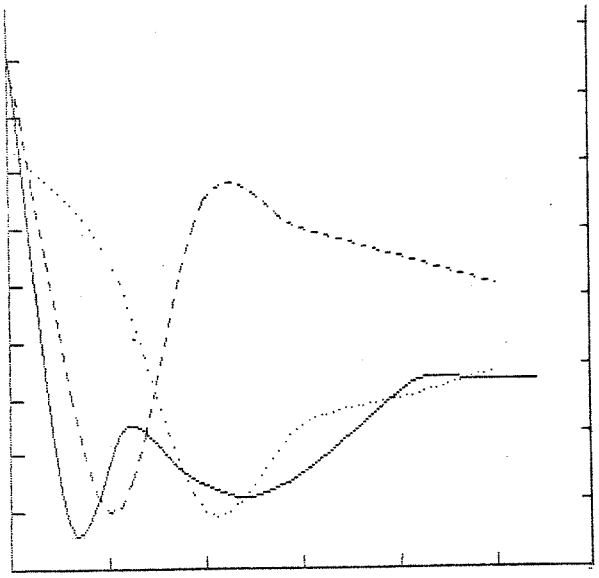

Figura 5.15. Representación superpuesta de varias funciones con diferentes ejes

Una forma alternativa de representar varias funciones sobre un mismo diagrama es la representación seudoperspectiva, del tipo que puede verse en la figura (5.16).

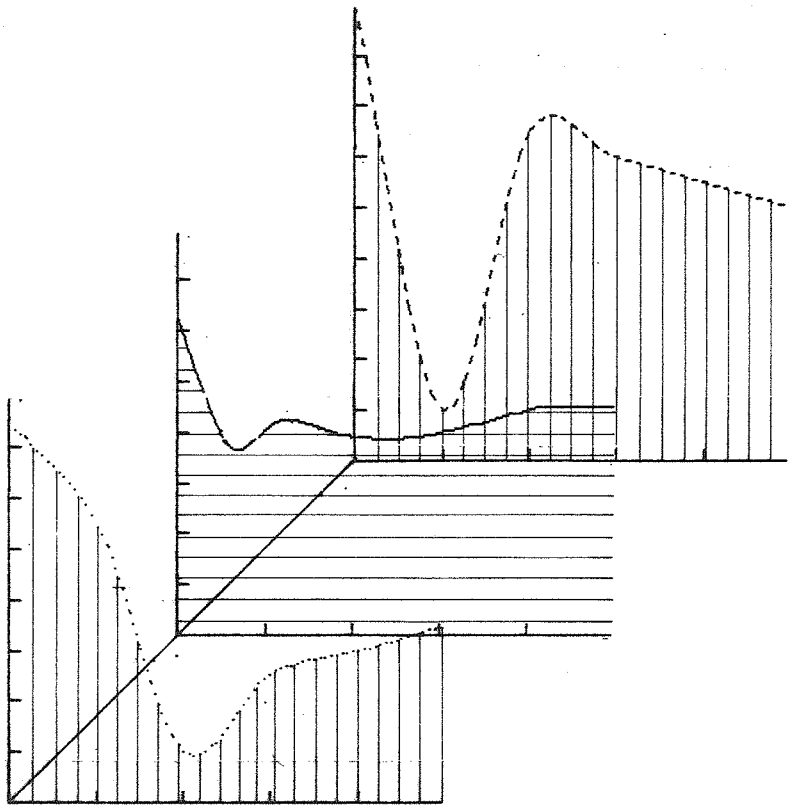

Figura 5.16. Representación de varias funciones en seudoperspectiva

La ventaja de esta representación es su mayor claridad; pero su inconveniente más grave (además de que su obtención es complicada) es que no permite comparaciones cuantitativas entre las diferentes funciones representadas.

5.5.4 Implementación de la representación de funciones.

La representación de funciones se reduce a representar pares de valores: a) valor de la variable X, y b) valor de la función F(X). Los valores máximos y mínimos que pueden tomar la variable y la función determinan los 'rangos' de la representación. No obstante, puede ser útil representar solo parte del rango de una función (p.e. para poder compararla con otra). Por ello, la implementación debe incluir la posibilidad de definir el rango que se quiere representar; tanto para valores de la variable, como para valores de la función.

Por otra parte, el rango a representar, debe encajar dentro de la ventana de la pantalla destinada a tal fin. Para lo cual se debe calcular un factor de corrección de los valores de la variable, y otro para los valores de la función (fig. 5.17).

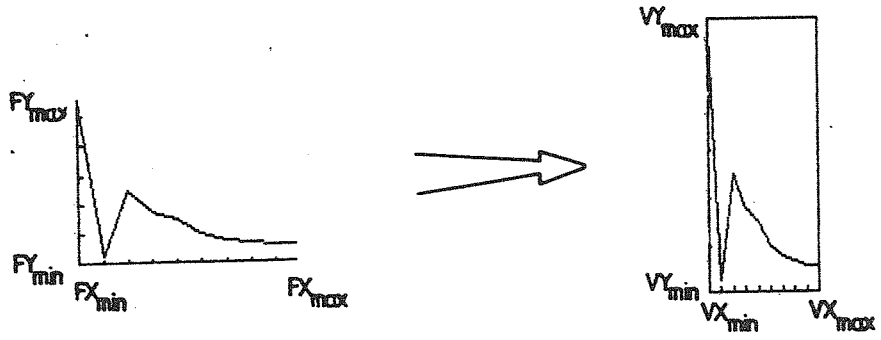

Figura 5.17. Rango de la función y ventana de pantalla.

El factor de corrección (o escala) necesario para la variable será de la forma:

$$E = (VX_{max} - VX_{min}) / (FX_{max} - FX_{min}) \qquad (5.8)$$

De esta forma, cualquier valor de la variable FX, podrá trasladarse a la representación como:

$$VX = (FX - FX_{min})\ E + VX_{min} \qquad (5.9)$$

Análogamente, para la función se tendrá:

$$E = (VY_{max} - VY_{min}) \ / \ (FY_{max} - FY_{min}) \qquad (5.10)$$

y

$$VY = (FY - FY_{min})\ E + VY_{min} \qquad (5.11)$$

Una vez obtenida la representación de la función, se debe tener en cuenta que toda gráfica debe mostrar que cosas está relacionando, ademas de mostrar cual es la relación que tienen. Por ello es de capital importancia incluir en la implementación los mecanismos necesarios para identificar la función que se esta representando (rotulos indicando la función y la variable), así como el rango de la representación (escalas en los ejes de la representación).

5.5.5 Representación de funciones de más de una variable.

En el caso de funciones de dos variables, se toma un espacio de tres dimensiones. Sobre los dos primeros ejes se representan las variables, y sobre el tercer eje los valores de la función. Esto da lugar a que los puntos de dicho espacio que cumplen la función definan una superficie tridimensional.

Para representar este tipo de funciones se han propuesto diferentes métodos /16/. El más habitual, sin embargo, sigue siendo la representación en perspectiva. Esta representación de la función consiste en aplicar las mismas técnicas descritas anteriormente para representación de objetos tridimensionales, a la superficie de la función. En la figura (5.18) puede verse un ejemplo de dicho tipo de representación.

No obstante, esta forma de representación resulta poco práctica cuando hay que representar más de una función. El problema es semejante al caso de funciones de una sola variable. La representación se vuelve rápidamente confusa. Además, la información cuantitativa para comparar las funciones queda enmascarada por el efecto de la perspectiva.

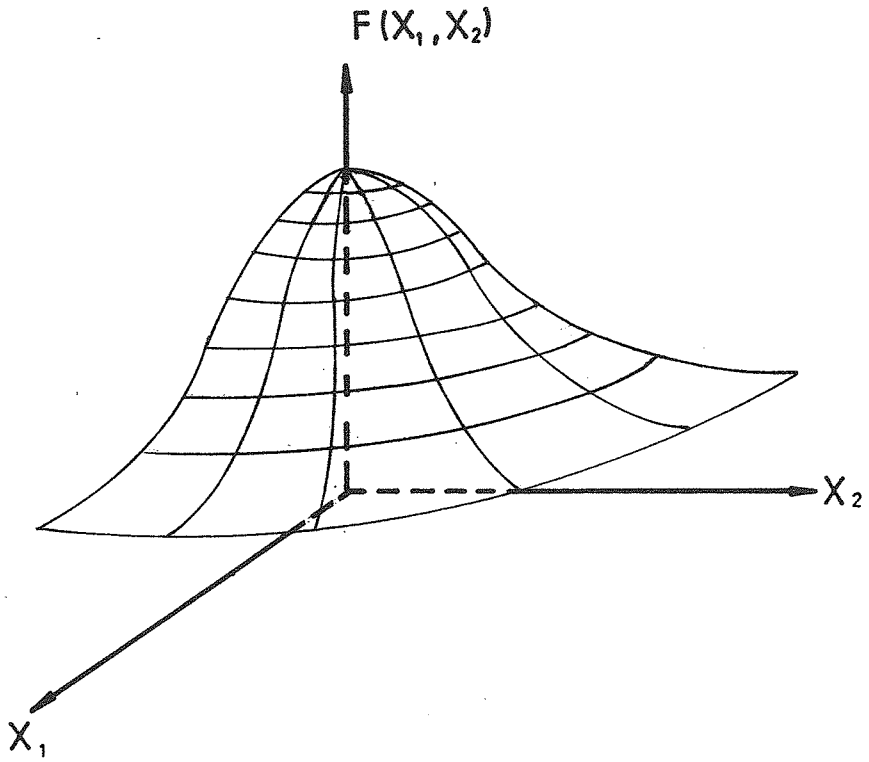

Figura 5.18. Seudoperspectiva de una función de dos variables.

Para evitar estos problemas, se puede recurrir a una representación menos intuitiva, pero más simple. Es la representación por líneas de contorno (o curvas de nivel). Tal como se vé en la figura (5.19), esta representación consiste en dibujar sobre el plano coordenado formado por los ejes de las dos variables, las proyecciones de las curvas de valor constante de la función (curvas que resultan de la intersección de la superficie de la función con planos de valor constante de la función).

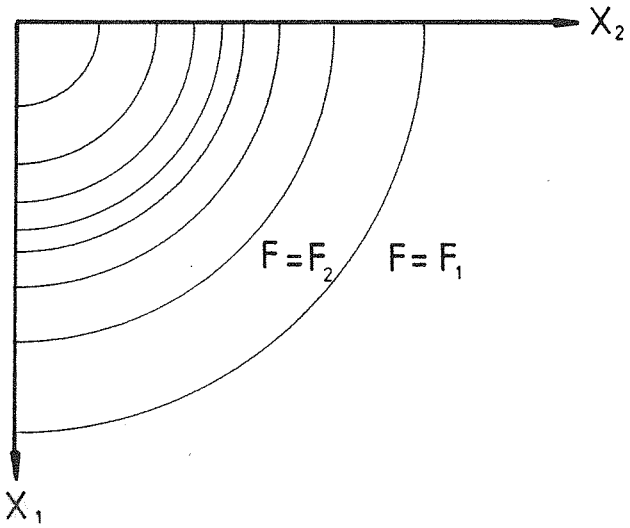

Figura 5.19. Función de dos variables por curvas de nivel.

Para representación de espacios de más de dos dimensiones, no cabe otra alternativa que la representación de diferentes subespacios de dos o tres dimensiones. Los subespacios de dos dimensiones se puede representar de las dos formas que acabamos de describir. Por el contrario, para representar subespacios de tres dimensiones se debe recurrir a una generalización de las curvas de nivel: se representan las superficies de tres dimensiones que resultan de "cortar" a la función por hiperplanos de valor constante de la misma. En este caso, la representación vuelve a ser una seudoperspectiva, por lo que sigue teniendo los problemas de falta de claridad e imprecisión. Por ello , lo habitual es la representación de subespacios de dos variables.

CAPITULO 6

CONFIGURACION DE UN SISTEMA INTERACTIVO Y GRAFICO DE DISEÑO OPTIMO DE ESTRUCTURAS

6.1. INTRODUCCION

El desarrollo de un sistema de programación es una tarea muy compleja, que debe considerarse desde varios puntos de vista. Además, se deben tener en cuenta los condicionamientos particulares del tipo de problema a resolver /3/. En general, se deben tener presentes dos aspectos en un primer nivel de descomposición del problema:

- organización del código, y

- gestión de la información.

Para la organización del código, debe tenerse presente que actualmente, un sistema de programación contiene decenas de miles de instrucciones. Por ello, el desarrollo de uno de estos sistemas está fuera del alcance de una sola persona. Esto conlleva una necesidad de comunicación y coordinación entre todas las personas implicadas en el desarrollo. Además, la modificación del software obtenido es difícil, porque nadie llega a tener un conocimiento suficientemente detallado de todo el sistema.

En cuanto a la información, la característica determinante en un problema de diseño es su variedad. Esta variedad, unida a que en ciertas partes del proceso también hay gran cantidad, determina la necesidad de desarrollar una gestión de la información específica para este tipo de problema. Por estas características, no resulta efectivo emplear bases de datos de propósito general en los sistemas de programación de diseño optimo de estructuras. Esto se debe a que estas bases de datos fueron desarrolladas para su aplicación en problemas de distinta naturaleza: grandes cantidades de información muy semejante entre sí.

Además de estos aspectos generales, debe destacarse que la parte más impredecible de un programa interactivo es la interacción con el usuario. Es decir, la parte del programa que determina la forma de comunicarse el usuario con el ordenador. También debe tenerse en cuenta que este aspecto del sistema influye, de forma absoluta, en la organización del mismo. Por lo cual, este es un tercer aspecto importante en el tipo de problema que nos ocupa.

A continuación se estudian con más detalle cada uno de estos tres aspectos, y su influencia decisiva en la organización adoptada para el sistema. En la segunda parte del capítulo se describe la forma en la que se han abordado estos problemas en el sistema de diseño DISSENY.

6.2 ORGANIZACION DE UN SISTEMA DE PROGRAMACION.

6.2.1 Organización del código.

Tal como hemos comentado en la introducción, los códigos de los programas actuales son demasiado extensos para que puedan ser creados y mantenidos por una sola persona. Además, el desarrollo y modificación del software resulta cada vez mas dificil y caro. Por ello, necesitamos marcar un conjunto de objetivos que permitan la creación de software en equipo y trasciendan los efectos de cualquier cambio. Ross, Goodenough y Irvine (citados por Booch, /12/) describen cuatro objetivos. Estos són:

- Adaptabilidad/modificabilidad.
  Su significado básico es el de permitir "cambios controlados". Así, para conseguir un nuevo diseño se pueden alteran ciertas partes, sin que el resto se vea afectado.

- Eficiencia.
  Obtener eficiencia implica que el sistema utilice todos los recursos disponibles de forma óptima. La división clásica de los recursos es la que distingue entre dos tipos: tiempo y memoria.

- Fiabilidad.
  Se trata de prevenir fallos de concepción, diseño y construcción, y también de detectar y anular los fallos en ejecución.

- Comprensibilidad.
  Se debe perseguir una estructura clara para cualquier sistema de software. Obtenerla es esencial para que el resultado cumpla las tres condiciones anteriores.

Analizando con más detalle los cuatro objetivos propuestos, observamos que pueden llegar a ser contrapuestos. Por ejemplo, utilizar recursos sofisticados para incrementar la eficacia va en detrimento de la modificabilidad (e incluso de la fiabilidad). Por otra parte, vemos que estos objetivos tienen el doble peligro de que constriñen demasiado ("la fiabilidad debe costruirse desde el principio, no puede añadirse al final"), y nos pueden llevar a un callejón de "microeficiencia" (mejorar pequeños detalles carentes de importancia global). En definitiva, estos objetivos deben considerarse como metas deseables pero utópicas. Lo ideal es encontrar el punto de compromiso entre un costo y unos resultados aceptables.

Para conseguir los objetivos propuestos, debemos plantear una serie de condiciones. Es decir, que debemos exigirle al software unos atributos que induzcan la consecución de dichos objetivos. Estos requisitos o atributos son:

- Abstracción y ocultamiento.
  El software debe ocultar los detalles carentes de importancia. Se trata de hacer inaccesibles los detalles que no afectan al resto del sistema: conforme descomponemos el problema en partes, los detalles de como se soluciona cada parte deben quedar ocultos al resto del sistema.
  Cuando ocultamos detalles (como la organización de datos de un fichero), evitamos que la estrategia global se base en ellos, produciendo una falta de adaptabilidad.

- Modularidad.
  Descomposición de la solución en tareas lo más independientes y completas que sea posible. Se aumenta así, principalmente, la fiabilidad.

- Uniformidad.
  Se trata únicamente de utilizar una notación consistente y homogenea, evitando toda diferencia innecesaria. Esto redunda en un aumento de la comprensibilidad.

- Verificabilidad.
  Se debe intentar que tanto el sistema total como cada una de sus partes (módulos) puedan ser comprobados independientemente del resto.

Tal como ya indicabamos con los objetivos, los requisitos deben buscarse aún sabiendo que son, generalmente, inalcanzables e incompatibles.

La programación estructurada propone reglas para programar, que conducen a los objetivos descritos respetando los requisitos. Las reglas de programación se plantean a tres niveles:

- a alto nivel, las reglas de fragmentación del problema en tareas;
- a medio nivel, las reglas de utilización de las 'estructuras de control', y
- a bajo nivel, las reglas de utilización del lenguaje de programación.

Evidentemente, no existe una separación clara entre los tres niveles, puesto que todos tienen influencia sobre el resto. No obstante, esta separación ayuda a plantear la forma de actuación.

La forma habitual de fragmentar un programa en tareas, es la que se denomina 'modulación'. Se trata de descomponer el problema en niveles jerárquicos y considerar dentro de cada nivel, un módulo independiente para cada una de las tareas requeridas a tal nivel. El problema, así descompuesto, se puede esquematizar por medio de un 'diagrama de árbol' como el mostrado en la figura (6.1).

La característica que debe cumplir cada módulo es la de realizar una única tarea del problema. Además debe realizar dicha tarea de forma totalmente independiente del resto de módulos. Según el orden en el que se diseñen los módulos, se distingen dos variantes, que reciben diferentes denominaciones:

- diseñar desde el primer nivel hasta el último es "TOP-DOWN", "descomposición sucesiva", "refinamiento sucesivo", etc.
- Diseñar desde el último nivel hasta el primero es "BOTTOM-UP", "composición sucesiva", etc.

Y lo habitual es no aplicar ninguna de las dos en su forma pura. Por el contrario, se suelen utilizar conjuntamente. Así, se potencia la creatividad "desbloqueando" al diseñador cuando llega a un "punto muerto".

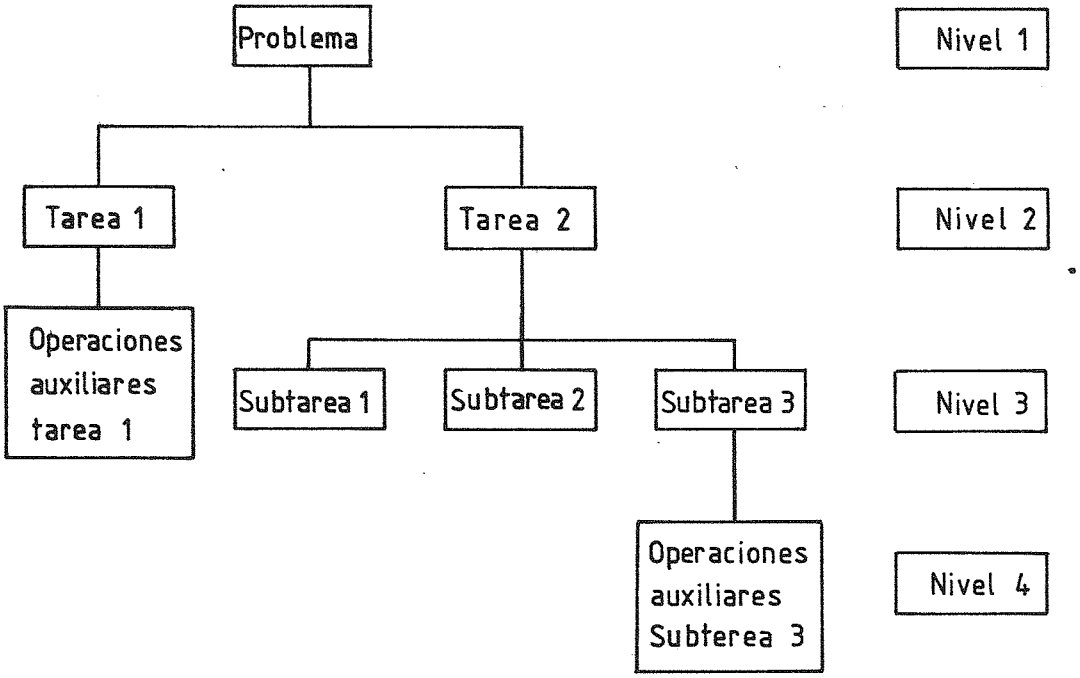

Figura 6.1. Diagrama de árbol

En cualquier caso, lo que se pretende es llegar a aislar partes del problema que puedan ser manejables por una sola persona. En ese momento, se emplean las técnicas propuestas en el nivel medio para detallar los algoritmos de instrucciones. Estas técnicas consisten básicamente en emplear únicamente ciertas estructuras de control. Las estructuras que deben utilizarse dependerán del lenguaje a emplear. Para FORTRAN, las estructuras apropiadas son:

- secuencia;
- decisión, y
- bucle.

Para obligar a utilizar correctamente estas estructuras (e impedir el uso de cualquier otra estructura), es muy útil el empleo de un tipo particular de diagramas de flujo a la hora de plantear la forma de realizar la tarea. Los diagramas apropiados son los que se conocen

como diagramas 'Nassi-Schneiderman', que fueron intoducidos por Holzer en el campo del análisis de estructuras /33/.

Las reglas de programación que deben seguirse a bajo nivel son las que indican que instrucciones deben o no emplearse para conseguir un código portable y eficiente. Evidentemente, este aspecto depende completamente del lenguaje de programación empleado, y además de la máquina en la que se realice la implementación, por lo que no se pueden dar reglas generales sobre el mismo. Por ello, sobre este aspecto, existe poca información. Por lo tanto, la forma habitual de determinar las instrucciones apropiadas consiste en comparar manuales de diferentes máquinas, o recurrir a la experiencia del propio programador.

6.2.2 Gestión de la información.

Durante la última década, se han llevado a cabo diferentes desarrollos de bases de datos específicas para el problema de diseño /4,67,68/. No obstante, el problema sigue sin estar resuelto. Los resultados alcanzados carecen de la generalidad y la eficiencia necesarias para su empleo en diferentes sistemas de programación. Es decir, que los desarrollos existentes están orientados a una aplicación particular y son dificilmente utilizables en otro sistema. Además, en su mayoría, están adaptados para su uso en ordenadores de gran capacidad. Mientras que la potencia y capacidad de los ordenadores personales existentes hace que la tendencia actual sea la de utilizarlos para ciertas partes del proceso de diseño (básicamente para la parte más interactiva). Debido a esta tendencia, la necesidad de un tratamiento independiente y generalizado de la información, se está viendo aumentada con la de trasmitir la información entre máquinas diferentes (con el condicionamiento añadido de la escasa capacidad de almacenamiento de los ordenadores pequeños). La solución, apuntada por Sreekanta /71/, consiste en desarrollar implementaciones para ordenadores pequeños de las bases de datos específicas para este problema.

Para definir una base de datos se debe determinar el tipo de información que se va a manipular, y que operaciones se deben realizar con dicha información. Esta parte del desarrollo de la base de datos se conoce como 'esquema conceptual'. Una vez definido dicho esquema conceptual se requiere una fase de implementación del mismo. Los detalles de como realizar dicha implementación se conocen como 'esquema físico' de la base de datos.

6.2.2.1 Esquema conceptual.

Para manipular la información es conveniente agruparla en función de ciertos tipos de semejanza. Los diferentes niveles en que se puede agrupar la información, reciben denominaciones habituales en la terminología de bases de datos. Así, Rajan y Bhatti definen los siguientes conceptos /66/:

- Dominio; que es el conjunto de la información que se considera.
- Entidad; que es cualquier parte del dominio que existe y es distinguible (unidad de información).
- Conjunto de entidades; formado por entidades que guardan algún tipo de similitud.
- Pieza de información; formada por conjuntos de entidades que describen un aspecto de la información.

El primer paso de la definición del esquema conceptual, consiste entonces en definir los 'atributos', o cualidades que debe reunir una entidad para pertenecer a un conjunto de entidades. Es decir, que si una entidad describe al conjunto de entidades al que pertenece, es un atributo de dicho conjunto. En definitiva, se trata de definir una forma de organizar la información.

El esquema que se elija para organizar esta información debe tender, por una parte a hacer eficientes todos los tipos de manipulaciones previstas, y por otra parte a evitar la dependencia mutua y minimizar redundancias en el almacenamiento de la información. Como es difícil que con una primera definición de la forma de agrupar la información y las manipulaciones permitidas se consiga un esquema que elimine las dependencias y sea eficiente, el segundo paso del proceso de definición del esquema conceptual consiste en un proceso de 'normalización' de la información /66,71/. En este proceso se redefinen los conjuntos de entidades y los atributos hasta lograr minimizar las redundancias.

La forma más habitual de evitar las redundancias consiste en almacenar cada entidad en un solo conjunto, y utilizar 'llaves' con las que se pueda hacer referencia al lugar en el que está la información. Se puede decir que una llave es un atributo que únicamente define a una

entidad del conjunto. Un ejemplo de utilización de llave es el de asociar cada tipo de material empleado en la estructura con un número de orden. La información del material se almacena de forma que se pueda recuperar invocando dicho número, y en la información de cada elemento únicamente se especifica el número que identifica al material del que está compuesto.

Al definir la forma en que se va a agrupar la información, se deben tener en cuenta las necesidades básicas de manipulación de la misma. En un programa de análisis de estructuras, las necesidades de manipulación de información se pueden resumir en:

- crear;
- leer, y
- borrar entidades.

En el campo más amplio del diseño, estas necesidades se incrementan con las de:

- actualizar, e
- insertar entidades.

Por lo tanto, el esquema propuesto debe tener en cuenta estas necesidades de manipulación, a fin de evitar incompatibilidades que las dificulten (o imposibiliten).

Se debe definir también un lenguaje que permita al usuario utilizar la base de datos ("query language" /66/). Este lenguaje de interacción del usuario con la base de datos debe ser la única forma en la que el usuario pueda manipular los datos. Por ello debe incluir todas las formas de manipulación que sean necesarias. Además, este lenguaje debe "ocultar" al usuario todo lo relacionado con la forma en que la tarea requerida sea llevada a cabo. De esto se deduce que la definición de este lenguaje influye tanto en el esquema conceptual como en el físico. Por ello, el lenguaje se define inicialmente en función de la que se prevee como forma más cómoda de uso (se intenta que sea un "lenguaje natural"), y se redefine y completa conforme se termina la especificación conceptual y comienza la implementación.

Por último, en la definición del esquema se debe distinguir entre lo que se prevee que serán usos habituales, y aquellos más esporádicos, a

fin de poder realizar una implementación que haga más eficientes a los primeros (aún a costa de hacer ineficientes los segundos).

Las diferentes formas de organizar las unidades de información resultantes de considerar todos los aspectos expuestos, dan lugar a tres tipos de relaciones diferentes. Así se distingue entre organización de datos:

- jerárquica;
- relacional, y
- red.

En el esquema jerárquico las unidades de información están clasificadas por niveles, que guardan una dependencia con sus respectivos niveles superiores (fig. 6.2). En el esquema relacional se permite la conexión entre unidades de información al mismo nivel (fig. 6.3). Por último, en el esquema de red, la conexión es al mismo nivel y en todos los sentidos (fig. 6.4).

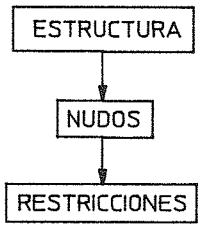

Figura 6.2. Esquema jerárquico.

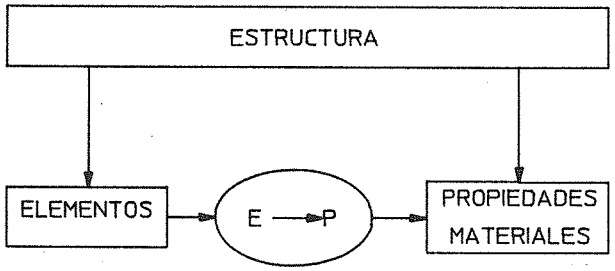

Figura 6.3. Esquema relacional.

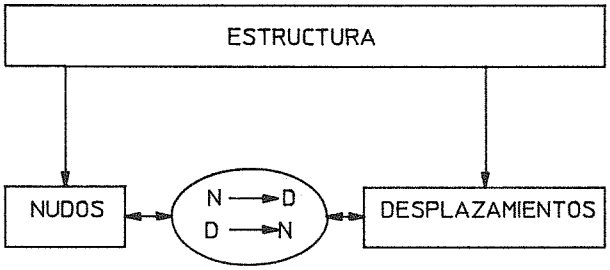

Figura 6.4. Esquema de red.

Ninguno de los esquemas se puede usar exclusivamente. Hay información que encaja mejor con cada uno de ellos. No obstante, la mayor parte de la información implicada en el proceso de diseño, se adapta bién al esquema relacional. Esto es debido a que evita las dependencias y redundancias del esquema jerárquico, y se adapta a las relaciones unidireccionales (que son la mayoría).

6.2.2.2 Esquema físico.

Para conseguir el esquema físico de la estructura de datos, se debe implementar el esquema conceptual. Si se usan tipos de datos que encajen al máximo con el esquema conceptual, la codificación se aproximará mucho a una especificación en lenguaje natural. Desafortunadamente, el lenguaje empleado mayoritariamente hasta la actualidad en la comunidad científica (FORTRAN) no soporta más que una mínima cantidad de tipos básicos: variables enteras, reales, doble precisión, caracter y lógicas. Los tipos estructurados son aún más limitados: solo se dispone de matrices. Otros tipos de datos básicos (escalares, punteros, etc) y de datos estructurados (pilas, colas, listas, etc) no están definidos. Por ello, llegamos a la alternativa de implementar la base de datos utilizando únicamente los tipos disponibles o 'emular' otros tipos que se adapten mejor al problema.

Para emular tipos de datos no soportados por el lenguaje, se recurre a gestionar la información almacenada en algún lugar de la memoria. En función del lugar elegido, existen tres opciones con nivel creciente de complejidad y generalidad (según Kunz y Hopkins /43/). La primera opción consiste en utilizar como almacenamiento los tipos de datos soportados por el lenguaje, y gestionar el acceso a ellos de forma que

se comporten como si fuesen los tipos emulados. El inconveniente de esta aproximación es que la cantidad de memoria utilizable debe fijarse de antemano (por ello se conoce como estructura estática de datos).

El siguiente paso consiste en almacenar la información en una única zona de memoria, que es fija durante la ejecución pero fácilmente modificable entre dos ejecuciones. Es decir, se utiliza uno de los tipos disponibles en el lenguaje (p.e. una matriz) para almacenar la información, que gestiona el emulador de forma que se comporte externamente como otro tipo de datos El tamaño total de la memoria disponible sigue siendo fijo. Si bién, ahora es fácilmente modificable entre ejecuciones. Además la memoria disponible se puede repartir libremente entre toda la información a almacenar (la estructura se denomina seudodinámica).

La aproximación más general consiste en almacenar la información sobre una zona de la memoria directamente controlada por la base de datos (sin utilizar como intermediarios ni al lenguaje de programación, ni al sistema operativo). Así puede disponerse de la cantidad de memoria del ordenador necesaria en cada momento, y la gestión de la información es muy eficiente. A cambio, el costo de desarrollo de esta estructura es muy alto. Por ello solo es aconsejable cuando el tamaño de los problemas a tratar sea muy variable y el ordenador a utilizar tenga mucha capacidad de memoria que deba ser compartida por otros usuarios. En cualquier otro caso, la estructura seudodinámica es el mejor compromiso entre costo de desarrollo y versatilidad de uso. Se obtiene con ella un código portable, que permite tipos dinámicos (definibles y modificables durante la ejecución) y área total de memoria seudodinámica (modificable entre ejecuciones diferentes).

Debido al trabajo que comporta el desarrollo de una base de datos, se han empleado también soluciones intermedias. Una de ellas consiste en definir un esquema conceptual de la organización de los datos que se ajuste a las necesidades del problema a tratar. Este esquema se implementa directamente en el código, en lugar de desarrollar completamente una base de datos que se encargue de manipular los datos según dicho esquema. Es decir, que se define un esquema conceptual de la base de datos, pero luego no se desarrolla el esquema físico de la misma; por el contrario, se implementa directamente sobre el código el esquema conceptual.

La solución que se obtiene resulta poco legible (luego, difícil de modificar); ineficiente (porque no aprovecha todos los recursos del ordenador), y no portable (porque utiliza detalles de implementación dependientes de máquina). Para mantener un alto nivel de efectividad y conseguir, al mismo tiempo, un código portable, se debe recurrir al empleo de las técnicas de abstracción, para emular los tipos de datos que se adaptan a la información que hay que manipular.

6.2.3 Interacción

La dificultad para definir la interacción deriva básicamente del hecho, corroborado por la experiencia, de que el usuario suele reaccionar de forma totalmente inesperada frente a un programa que permita el diálogo. La forma de actuar de cada usuario frente al programa depende en gran medida de su experiencia previa con otros programas y ordenadores. Los usuarios que no han utilizado anteriormente ningún ordenador esperan poder comunicarse con este de la misma forma como lo harían con un colega (aunque carecen de la soltura necesaria para comenzar el diálogo). Por el contrario, los usuarios que han utilizado otros programas tienen tendencia a emplear esta experiencia previa (aunque provenga del uso de programas orientados a problemas totalmente dispares). Esto lleva a que el programa no responda a las acciones del usuario; o a que lo haga de forma diferente a la esperada por este. Como consecuencia, pequeños malentendidos pueden ocasionar respuestas desastrosas, que desalentarán a los pocos usuarios que no hayan abandonado previamente al enfrentarse con un difícil aprendizaje. De aquí se deriva que la estrategia a seguir consiste en: a) determinar las necesidades del usuario, y b) desarrollar una interacción, fácil de usar, que cubra estas necesidades.

Por tanto, el primer paso debe ser la determinación de las necesidades del futuro usuario. Este estudio resulta generalmente difícil; bién porque las necesidades sean muy variadas; bien porque se carezca de la experiencia necesaria para determinarlas a priori. En cualquier caso, se puede hacer una primera aproximación, que se completará en la fase de pruebas. En este estudio se deben establecer los requerimientos funcionales derivados de las necesidades de los potenciales usuarios. El estudio puede llevarlo a cabo el mismo equipo que va a desarrollar el programa (cuando cuente con experiencia propia suficiente en la utilización de programas semejantes), o puede encargarse a un equipo de especialistas no relacionados con el desarrollo.

Para desarrollar una interacción con el usuario que haga que el programa sea fácil de usar, debemos enfocarla desde los tres aspectos propuestos por Newman y Sproull /60/:

-Modelo del usuario;

-Lenguaje de comandos, y

-Realimentación.

Vamos a estudiar con más detalle cada uno de estos aspectos.

6.2.3.1 Modelo del usuario

Entendemos como tal a la idea que el usuario tiene de la información que está manipulando y de los procesos que puede sufrir dicha información.

Sin este modelo, el usuario únicamente puede ejecutar secuencias de instrucciones (a modo de receta); sin anticipar los resultados de estas instrucciones, y sin poder establecer alternativas cuando se produzca cualquier tipo de fallo. El efecto más habitual de esta situación es el de que el programa se use únicamente en su forma más simple. Para evitar esto, en la mayoría de los casos, basta con proveer al usuario de un modelo conceptual muy simple.

Una forma cómoda de representar el modelo es la de un conjunto de objetos a los que el usuario puede aplicar un conjunto de acciones. El usuario debe saber que tipo de objetos puede manejar, y que control tiene sobre ellos. El modelo debe estar basado en unas cuantas ideas generales que permitan conocer:

- El comportamiento general del programa (si ejecuta inmediatamente las ordenes o pide confirmación, que ordenes se pueden usar siempre, que pasa si una orden se aborta, etc.).

- La forma de realizar las tareas encomendadas (sin entrar en descripciones técnicas).

En el desarrollo del modelo del usuario se deben tener en cuenta dos características del mismo:

- El modelo determinará las estrategias que el usuario considere apropiadas para trabajar con el programa.

- Debe emplear conceptos que resulten "familiares" al tipo de usuarios al que va destinado (aunque un modelo que difiera de la realidad de forma consistente es mejor que uno que solo difiera en algunas ocasiones inesperadas).

6.2.3.2 El lenguaje de comandos

Denominamos así al conjunto de ordenes que el usuario puede ejercer sobre el programa. Los lenguajes de comandos se emplean extensamente en la utilización de todo tipo de ordenadores. Podemos distinguir tres tipos de lenguajes de comandos entre los más empleados:

-Diálogo;

-Comandos de teclado, y

-Menus gráficos.

En el diálogo, el ordenador comienza planteando una serie de opciones y pide al usuario una elección. A continuación, se establece una cadena de preguntas y acciones, en la que, generalmente, cada pregunta depende de la respuesta dada a la anterior. De esta forma el usuario es guiado en todo momento; lo cual resulta muy cómodo en la fase de aprendizaje de un programa.

Conforme el usuario va adquiriendo experiencia, el dialogo llega a frenarle, con lo que aparece la necesidad de dar ordenes que contengan toda la información que anteriormente se le daba al ordenador como respuesta a un grupo de preguntas. Aparecen así los comandos de teclado. En estos el usuario debe aprender un conjunto de comandos (de sintaxis reducida), que, complementados por ciertos parámetros, permiten transmitir de forma muy resumida ordenes bastante complejas. Evidentemente, los comandos de teclado solo están justificados cuando la utilización del programa sea tan alta que haga rentable al usuario el tiempo de aprendizaje. Un caso particular de comandos de teclado es el que emplea teclas de función con una misión preprogramada. La función asignada a cada tecla se muestra habitualmente en la pantalla.

La alternativa más utilizada en los programas interactivos es la de menús gráficos. Su popularidad se basa en que muestran constantemente

todas las posibilidades de actuación del usuario (evitando indirectamente cualquier intento de utilizar acciones no permitidas en un momento dado). La opción habitual es la de un menú de comandos que se complementa con un diálogo para establecer los parámetros necesarios. Si bién, se emplea también un segundo menú de parámetros. Una segunda razón para utilizar menús interactivos es lo bién que se adaptan a entradas a través de periféricos, tales como ratones o tabletas digitalizadoras.

6.2.3.3 Realimentación

Por realimentación entendemos cualquier tipo de señal que recibe el usuario para confirmar que sus ordenes se cumplen.

La forma más sencilla de realimentación consiste en mostrar los resultados de la orden en la pantalla. Evidentemente, esto solo es válido cuando el tiempo de respuesta es muy corto. En caso contrario, el usuario comienza a impacientarse al no saber que esta pasando. Por ello el proposito más general de la realimentación se puede resumir en:

- Informar al usuario de que su orden ha sido aceptada;

- Indicar las acciones que se están ejecutando, y

- Mostrar el resultado.

Un caso particularmente importante de realimentación es la información al usuario de cualquier posible anomalia; indicando posibles causas y alternativas.

Una característica básica que debe poseer cualquier realimentación es la rapidez. El usuario queda totalmente desconcertado cuando el ordenador parece no responder a sus ordenes, y reacciona dando nuevas ordenes (que se acumulan, produciendo efectos incontrolados), o abandonando el programa.

6.3 IMPLEMENTACION DEL SISTEMA DE DISEÑO OPTIMO DE ESTRUCTURAS DISSENY

6.3.1 Características de la instalación.

El sistema DISSENY comenzó a ser implementado en un ordenador de tamaño medio: un UNIVAC 1100/70. El sistema operativo empleado en dicho ordenador es el EXEC-8, y la implementación comenzó en lenguaje FORTRAN V (una versión para Univac del FORTAN IV estandar).

Posteriormente, se pasó a emplear el nuevo estandar del lenguaje: el FORTRAN/77, manteniendose la implementación en el mismo ordenador. Del lenguaje de programación empezaron a eliminarse las dependencias de máquina, a fin de conseguir una versión compatible para diferentes compiladores y/o ordenadores /44/.

El tercer paso consistió en el traslado de ciertas partes del sistema a un ordenador personal (NCR PC8). Concretamente, se trasladó una versión reducida del procesador de análisis, y el procesador de control e interacción. Adaptandose, este último, para su uso bajo el sistema operativo DOS 3.0. El lenguaje de implementación siguió siendo el FORTAN/77 (subset), en la versión 3.3 de Microsoft /56/.

En este ordenador personal comenzó el desarrollo de una nueva versión del procesador de tratamiento gráfico; cuya primera versión estaba adaptada para su uso a traves de una terminal gráfica (Tektronix 4025), conectada al UNIVAC 1100/70. En la primera versión se utilizó como soporte para el programa la librería gráfica PLOT-10. En la nueva versión se ha utilizado la librería gráfica HALO /54/.

La versión utilizada actualmente, está implementada en un ordenador personal de la última generación (IBM PS/80). Esta versión aún funciona bajo sistema operativo DOS, pero esta prevista su adaptación a los sistemas operativo Operating Sistem, y UNIX.

La parte de cálculos más costosos del proceso, se sigue realizando en el UNIVAC 1100/70, y la información entre los dos ordenadores, se transmite por medio de los ficheros de datos. Estos ficheros están controlados por los respectivos sistemas operativos, pero son perfectamente compatibles.

6.3.2 Características del sistema.

El sistema DISSENY está organizado de forma modular a varios niveles (jerarquizado). En la figura (6.5) puede verse la organización a mayor nivel. En ella se observa que el sistema está dividido en 'procesadores' encargados de realizar las diferentes tareas que componen el diseño /50/.

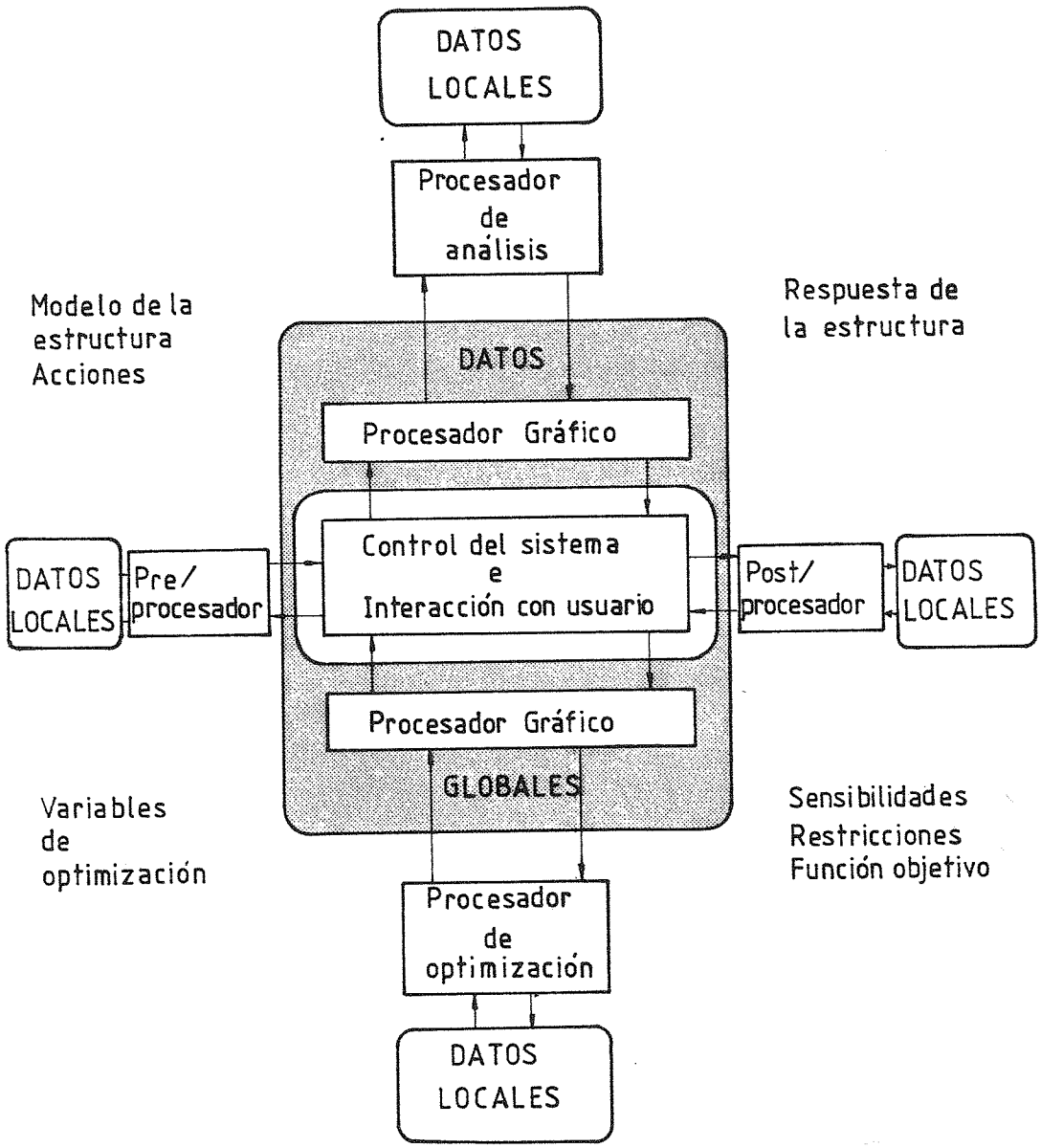

Figura 6.5. Organización del sistema DISSENY

Dentro de cada procesador, la organización vuelve a ser modular, descomponiendose la tarea del procesador en operaciones claramente diferenciadas encapsuladas en 'módulos' independientes. Por último,

cada uno de los módulos puede estar dividido en 'submódulos', cuando la operación que debe realizar es suficientemente larga o compleja. Esta organización modular de más bajo nivel está también relacionada con las características de la gestión de datos que vamos a ver a continuación.

El procesador de control e interacción es el encargado de mantener una relación interactiva con el usuario, y controlar el flujo de ejecución e información del sistema. La organización de este procesador puede verse en la figura (6.6).

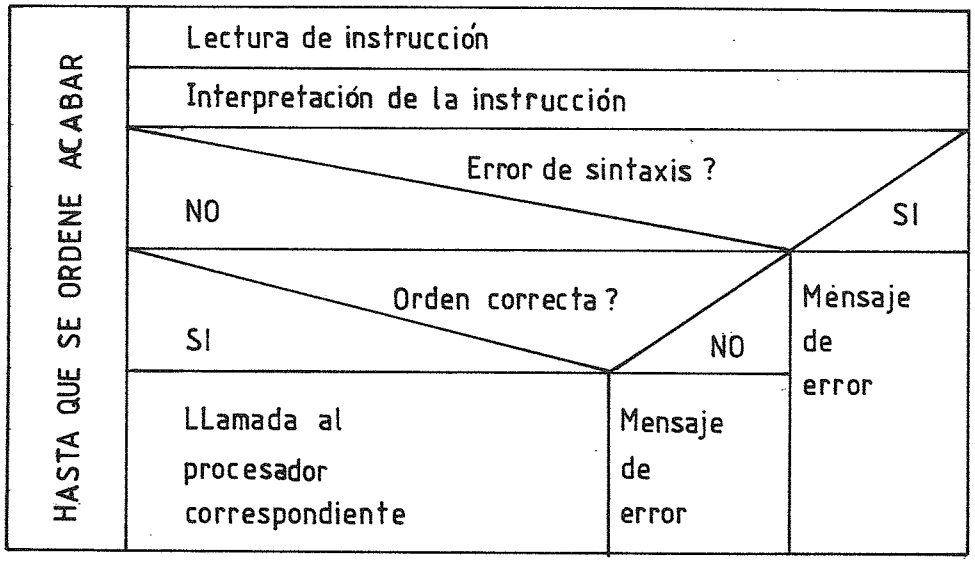

Figura 6.6. Organización del procesador de control e interacción.

El procesador de análisis se basa en el método de los desplazamientos aplicado a elementos finitos. Se ensambla la matriz de rigidez a partir de las matrices de los elementos, y se resuelve el sistema de equilibrio por el método de eliminación de Gauss, con descomposición en bloques para resolver sistemas grandes con poca memoria (fig. 6.7).

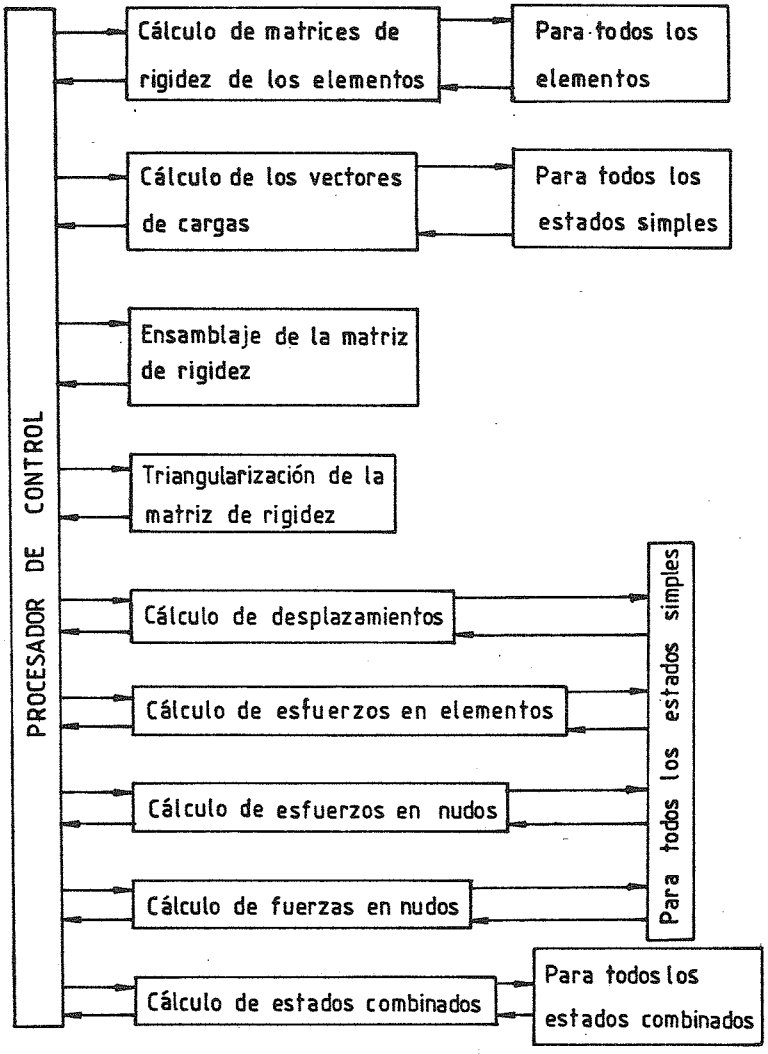

Figura 6.7. Organización del procesador de análisis.

El procesador de optimización se basa en un algoritmo de programación cuadrática sucesiva. Este algoritmo resuelve el problema de función objetivo y restricciones no lineales de forma sucesiva; mediante aproximaciones cuadráticas de la función objetivo, y aproximaciones lineales de las restricciones. El algoritmo utiliza una estrategia de conjunto activo de restricciones (fig 6.8).

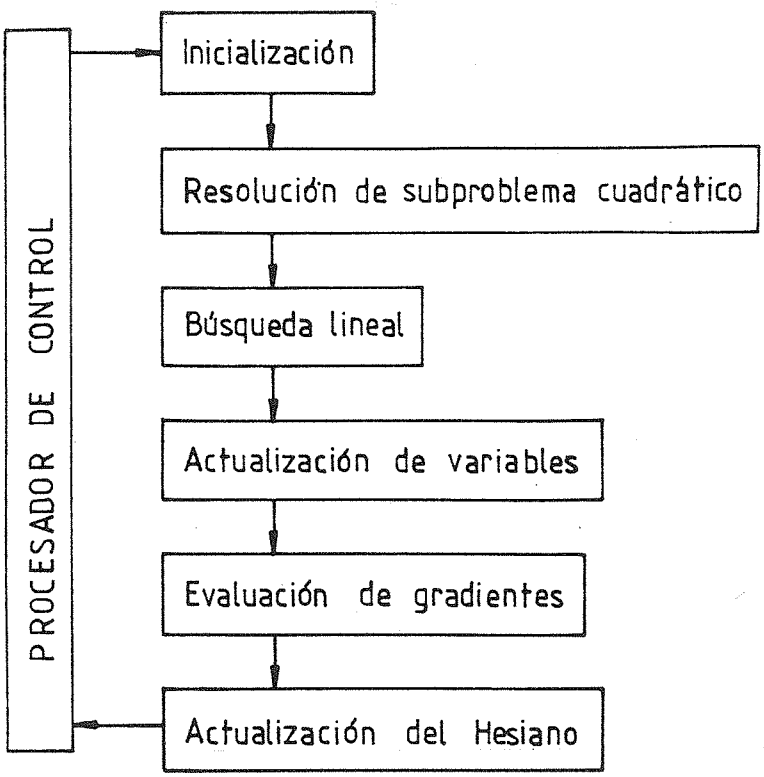

Figura 6.8. Organización del procesador de optimización.

La función del preprocesador es el cálculo o actualización de las variables de diseño a partir de las variables de optimización. El paso contrario es llevado a cabo por el postprocesador, que se encarga también de realizar el cálculo de sensibilidades.

Tal como se ve en la figura (6.5), la comunicación entre los diferentes procesadores se realiza a traves de una gestión centralizada de la información global. Cada procesador toma la información que necesita, y actualiza información, generada por él, que sea de utilidad al resto del sistema.

Como no toda la información utilizada por un procesador es común al resto del sistema, cada uno controla una información local en donde almacena la información para su uso exclusivo.

A continuación describiremos las características más destacadas de la gestión de la información, y del procesador de interacción (el resto

de los procesadores realizan tareas que ya han sido descritas en los capítulos precedentes).

6.3.3 Esquema conceptual de la información.

Para realizar la gestión de datos se ha realizado la primera parte del esquema conceptual de una base de datos. Se han definido las entidades, los conjuntos y los atributos del dominio de la información considerada. Esto se ha hecho según el criterio de minimizar redundancias y facilitar las manipulaciones habituales en el proceso de diseño. El esquema resultante es, básicamente, de tipo relacional; si bién hay cierto tipo de información que sigue un esquema jerárquico /14/.

Para establecer el esquema se ha partido de una primera distinción entre el modelo de la estructura y las acciones aplicadas. El segundo nivel de distinción es el habitual en análisis: a) la información de la estructura se descompone en la correspondiente a nudos y la de elementos, y b) la información de cargas se separa por estados (o hipótesis). Esta separación viene condicionada tanto por la idealización necesaria para realizar el análisis, como por las características del propio proceso de análisis. Por ello resulta cómoda desde todos los puntos de vista. Recurriendo a un tercer nivel, la información se divide en conjuntos de entidades perfectamente coherentes, y se llega al esquema habitual en análisis de estructuras (fig. 6.9).

Este esquema contiene, sin embargo, ciertas redundancias. La mayor parte de ellas son debidas a la forma de relacionar la información concerniente a los elementos. En primer lugar, las formas de los elementos hacen referencia a coordenadas que ya estaban definidas en los nudos. Por lo tanto, lo lógico es definir una numeración de los nudos, para utilizarla como llaves de las coordenadas. Así, la forma de los elementos se define por medio de sus conexiones con los nudos (de los cuales se dá únicamente la numeración). Por otra parte, tanto las características del material como las de la sección están asociadas con cada elemento. Lo cual impide, por ejemplo, definir un material sin que esté asociado a ningún elemento. Esto no deja de ser un inconveniente menor en un proceso de análisis, que únicamente obliga a mantener una información generalmente redundante. Sin embargo, de cara al proceso de optimización, se convierte en un inconveniente grave. Por ello, se recurre a normalizar esta

información definiendo tipos de materiales y de propiedades, que se asocian a los elementos por medio de llaves.

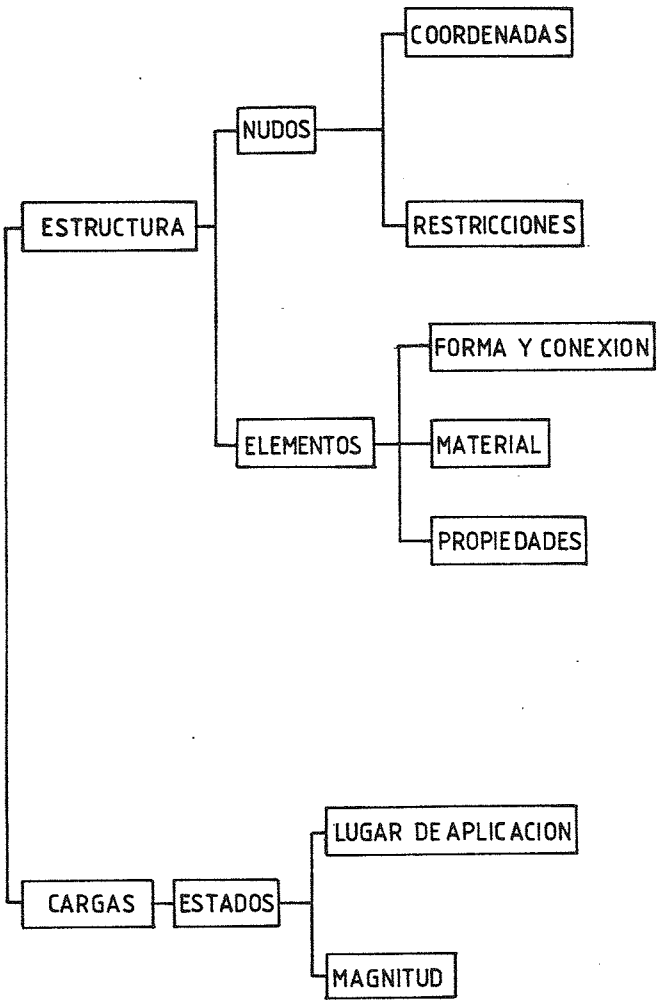

Figura 6.9. Conjuntos de información para análisis.

El segundo inconveniente del esquema propuesto en la figura 6.9 parte del hecho de que en análisis por elementos finitos es habitual considerar ensamblajes de elementos de diferentes tipos. Dado que estos elementos reciben un tratamiento análogo, pero con ciertas diferencias de formulación, conviene agrupar los elementos de un mismo tipo a fin de facilitar el proceso de cálculo. Por ello, es cómodo definir el concepto de grupos de elementos (fig. 6.10).

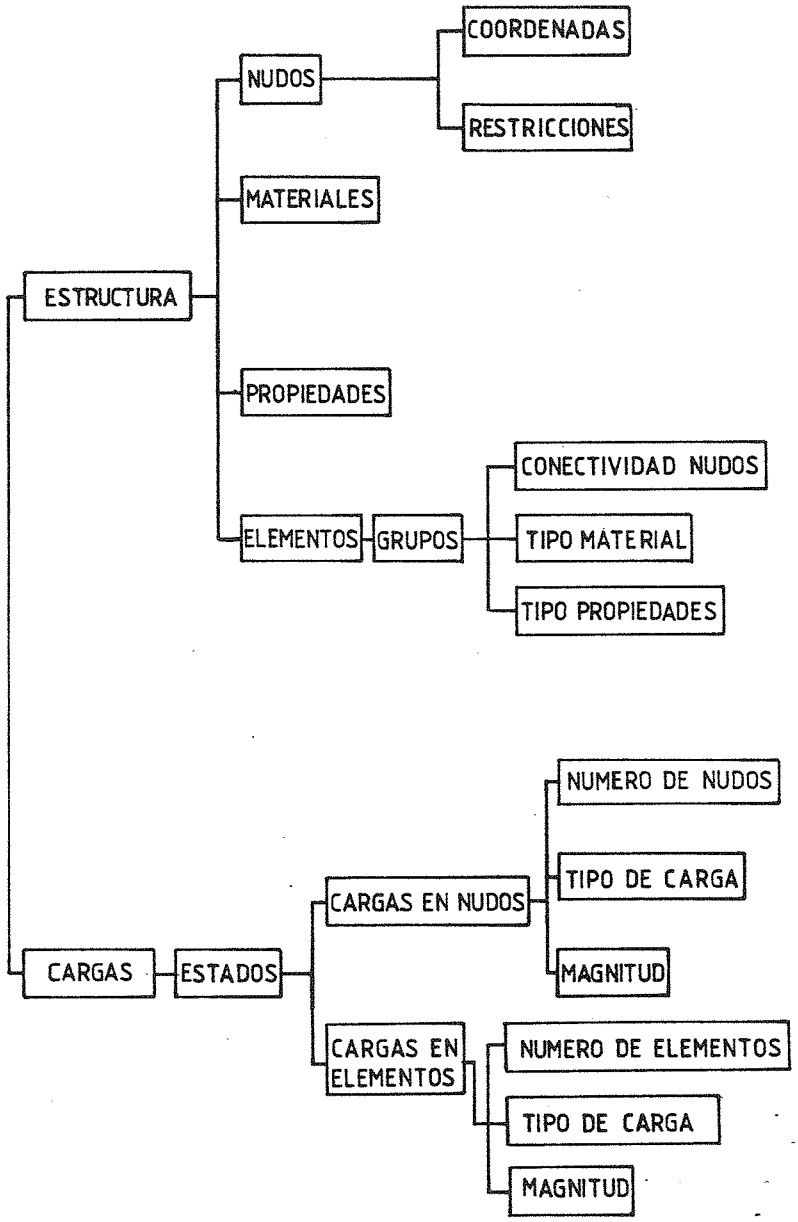

Figura 6.10. Información no redundante para análisis.

Por otra parte, cuando generalizamos el proceso, pasando del análisis al diseño, nos aparece un nuevo tipo de información a considerar: las variables de diseño. Con el esquema propuesto en la figura (6.9), la inclusión de las variables de diseño implica la modificación/actualización de mucha información en cada ciclo del proceso de optimización. Además, para poder tratar problemas prácticos se debe reducir el número de varibles de diseño. Por ello, la introducción de grupos de materiales, propiedades y elementos, facilita también el agrupamiento de las variables de diseño, (fig. 6.10).

Además, con el esquema propuesto en la figura (6.10), nos acercamos a lo que Fleury denomina 'modelo orientado a diseño' /23/. En este esquema, la información del modelo de análisis debería quedar totalmente supeditada a la información de diseño (llegando hipoteticamente a ser definida de forma automática a partir de ella) (fig. 6.11).

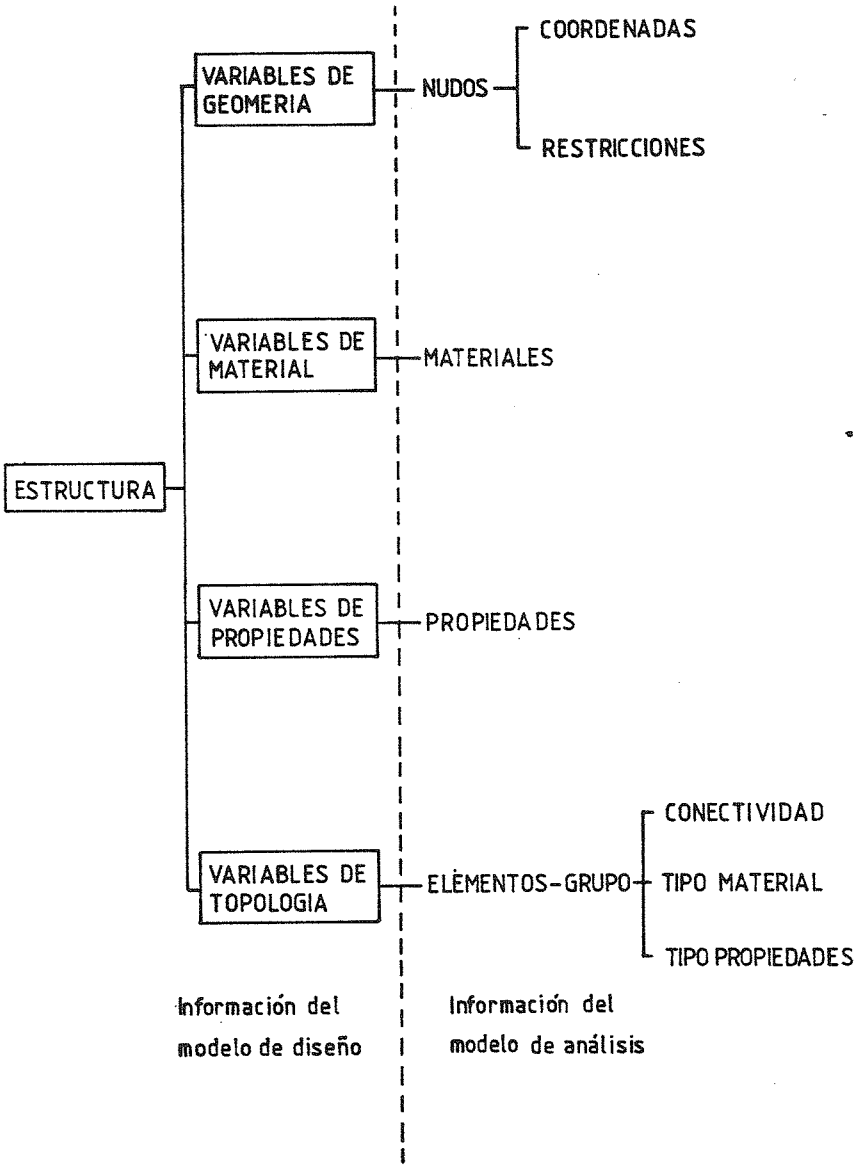

Figura 6.11. Información de diseño.

Indicar, por último, que el esquema conceptual propuesto se ha implementado directamente sobre el sistema (debiendo hacerse la gestión sobre el soporte físico de la información). Es decir, que

aunque se ha contemplado la introducción de un lenguaje de interacción (que realice las tareas necesarias de manipulación de información de forma transparente al resto del sistema) no se ha llegado a definir. Esto se debe a que, si bién la grán mayoría de las manipulaciones están perfectamente definidas (las que corresponden al proceso de análisis básicamente), hay algunas que no se han llegado a determinar completamente. La razón estriba en que debido a la introducción de la interactividad en el proceso de diseño, el flujo más elemental de un proceso automático se complica; haciendo muy fácil la aparición de anomalías por mal uso de la gestión de la información. Con el uso del sistema se deberá adquirir la experiencia que permita definir las manipulaciones necesarias, evitando estas posibles anomalias, con lo que se podrá definir el lenguaje de manipulación apropiado.

6.3.4 Esquema físico de la información.

Para implementar la gestión de datos se ha recurido a emular tipos de datos a partir de la metodología de tipos abstractos de Isner /36/. La emulación la lleva a cabo un bloque de subrutinas que gestiona la información que está físicamente almacenada en un único vector de gran tamaño, según el esquema empleado por Wilson y Hoit /75/. Variando la dimensión de este vector se consigue el caracter seudodinámico del área total de memoria. Esto permite su rápida adaptación a diferentes máquinas y/o problemas.

En la actualidad el tipo de datos emulado por los gestores es la 'matriz dinámica'. Esta es similar a la matriz definida en FORTRAN, pero con posibilidad de modificarla en tiempo de ejecución. Estas matrices dinámicas tienen tres dimensiones, y pueden contener datos de tipo entero, real, doble precisión, caracter y lógico. Las operaciones que se permiten sobre este tipo de datos se realizan a traves de llamadas a subrutinas con los argumentos necesarios.

El conjunto de subrutinas que realizan todas las operaciones está encapsulado en un solo paquete /15/. No obstante, no resulta viable realizar las operaciones de lectura y escritura a traves del gestor. Es más práctico "asignar" estas matrices dinámicas a matrices estandar de FORTRAN, haciendo una equivalencia de argumentos entre subrutinas y trabajando directamente sobre ellas en la forma habitual. Esta solución tiene el inconveniente de romper el encapsulamiento del tipo de datos, permitiendo accesos ilegales al mismo. La solución de

compromiso consiste en reglamentar la forma de utilización del tipo de datos, recomendando no usar otras alternativas.

La primera regla a seguir es la de descomponer el programa en tareas simples (la modularidad ya comentada). A continuación se debe descomponer cada tarea en una fase de gestión y otra de realización de la tarea propiamente dicha (ver fig. 6.12). Este planteamiento tiene la ventaja adicional de que combinando de forma apropiada las operaciones sobre matrices dinámicas, se pueden emular otros tipos de datos (p. e. las listas) de forma relativamente sencilla. Además, la emulación queda "segregada" del resto del programa; lo que hace que no se pierda la legibilidad, y que estos tipos emulados puedan ser fácilmente reemplazados por tipos abstractos cuando se compruebe que su utilización es tan frecuente que hace rentable su desarrollo.

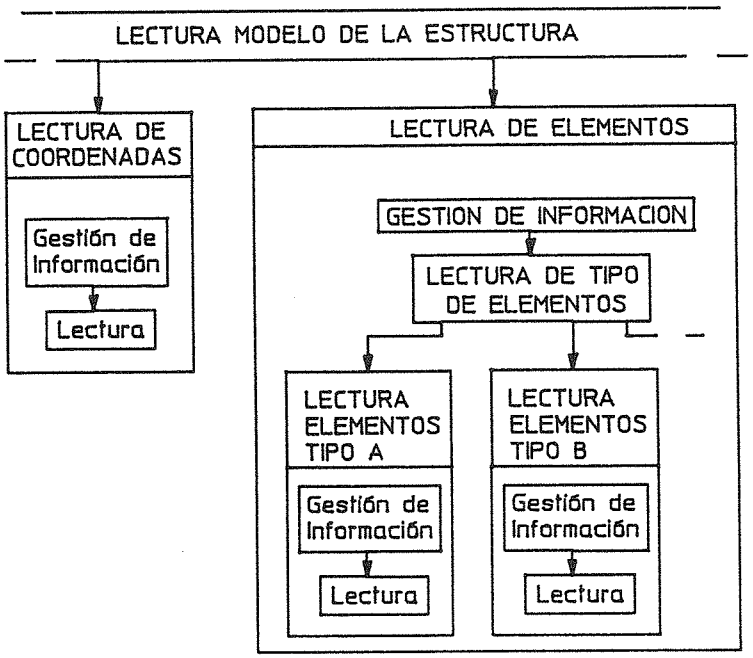

Figura. 6.12. Descomposición jerárquica y gestión de información

Las subrutinas que se encargan de las operaciones con matrices dinámicas son llamadas por medio de las instrucciones:

```
CALL CREA   ('NOMB',TIPO,NF,NC,NP,
             INOMB,*999)
CALL BORRA  ('NOMB'*999)
CALL LIMPIA ('NOMB',*999)
CALL BUSCA  ('NOMB',INOMB,*999)
CALL EXISTE ('NOMB',EXNOMB,*999)
```

```
CALL AMPLIA ('NOMB',IF,IC,IP,*999)
CALL REDUCE ('NOMB',IF,IC,IP,*999)
CALL TAMANO ('NOMB',NF,NC,NP,TIPO,
            LONGI,*999)
CALL HUECO  (TIPO,LONGI,INDICE,*999)
CALL NOMBRE ('NOM ',1,NOM1,*999)
CALL CAMBIA ('NOM1','NOM2',*999)
```

Siendo 'NOMB' el identificador de la matriz; NF,NC,NP sus dimensiones, y INOMB el puntero que se utiliza para la equivalencia con una matriz estandar. EXNOMB es una variable lógica que toma el valor .TRUE. cuando la matriz está definida y .FALSE. en otro caso. IF,IC,IP son los incrementos (o decrementos) de las dimensiones. LONGI es el total de memoria utilizada por la matriz en el almacenamiento interno. NOM1 es una variable caracter formada a partir de una raiz alfabética y un número (es muy útil para operar con matrices dinámicas dentro de bucles).

Se han implementado también operaciones auxiliares que ayudan a la detección de errores por mala utilización de operaciones y/o por utilización de operaciones ilegales:

```
CALL VUELCA ('NOMB',*999)
CALL TRAZAM ('SN',*999)
```

Dado que la memoria disponible en un ordenador personal es muy escasa en relación a las necesidades de un problema de diseño óptimo, y dado que se busca independizar las diferentes tareas que componen el proceso, no es viable mantener toda la información en la memoria principal. Aparece, por tanto, la necesidad de utilizar almacenamiento auxiliar (discos, cintas, etc). Así, se permite además que varios usuarios compartan la misma información, o que un solo usuario realice parte de un proceso y lo reanude posteriormente. En el sistema DISSENY se dispone de un gestor encargado de realizar estas tareas /13/. Este gestor permite:

- transvasar matrices dinámicas desde la memoria principal hasta ficheros del Sistema Operativo (S.O.) y viceversa. Las operaciones son:

```
CALL GUARDA ('NOMB',FORMA,*999)
CALL COPIAF ('NOMB',FORMA,*999)
CALL REPONE ('NOMB',INOMB,*999)
CALL COPIAM ('NOMB',INOMB,*999)
```

siendo FORMA un caracter que indica el tipo de formato con que se almacena.

- Conectar ficheros secuenciales de trabajo con ficheros del S.O. Las operaciones son:

```
CALL CREAF ('NOMB',FORMA,NUMU,*999)
CALL ABRE ('NOMB',FORMA,NUMU,*999)
CALL CIERRA ('NOMB',*999)
```

(siendo NUMU el número de la unidad lógica con la que se conecta el fichero para realizar las operaciones READ y WRITE)

El gestor dispone, así mismo, de operaciones auxiliares que facilitan las tareas de depuración. También dispone de operaciones que interaccionan con el S.O. De esta forma, el gestor utiliza los recursos del S.O. en lugar de gestionar directamente la memoria. La eficiencia de esta solución dependerá de la del propio S.O. En el caso del DOS esta eficiencia es muy baja, por lo que resultaría rentable el costo de implementación de una gestión directa.

6.3.5 Procesador de control e interacción.

Dado que el sistema tiene partes que funcionan de modo secuencial (y, generalmente, automático), y otras muy interactivas, la estrategia de comunicación entre el usuario y el ordenador no puede ser única. Por ello, se ha optado por un lenguaje de comandos basado en menús jerarquizados, que se complementa con diálogos en las partes más interactivas.

Por último, en algunas de las formas de uso que la experiencia ha mostrado como las más habituales, se ha empezado a introducir ciertos comandos de teclado, que permiten mayor rapidez a los usuarios habituales.

La estructura jerárquica de los menús en la parte más secuencial del sistema (análisis y optimización) está claramente justificada para permitir un funcionamiento normal rápido (permitiendo, al mismo tiempo, usos más complejos). Puesto que, además, el nivel de los posibles usuarios no va a ser uniforme, la estructura jerarquizada se ha organizado de forma que el sistema pueda tener un uso básico fácil (pero que también pueda accederse a opciones de uso más sofisticadas).

Para distinguir los diferentes niveles de los menús, y permitir el movimiento entre ellos, se ha establecido una jerarquía basada en un caracter identificador que antecede a cualquier instrucción. Los niveles definidos son (de mayor a menor rango):

* para identificar órdenes;
> para identificar comandos;
/ para identificar opciones;
_ para identificar subopciones.

Se ha definido también un identificador complementario (?) que permite indagar en cualquier momento sobre el menú activo.

Este esquema de menús tiene el grave inconveniente de que gran parte de las posibilidades del programa estan "ocultas" al usuario; que tiende a perder la visión global del funcionamiento del programa. No obstante, se ha adoptado esta forma de trabajo por resultar la alternativa más viable durante la etapa de desarrollo de un programa (dados los continuos cambios y ampliaciones de diferentes partes del mismo).

Para simplificar las entradas del usuario, se ha desarrollado un grupo de subrutinas que potencian las posibilidades de lectura del FORTRAN/77. De forma que el usuario pueda introducir la información cómodamente (formatos en campo libre, identificadores alfanuméricos, comentarios, etc). Internamente, estas entradas son "traducidas" a un formato unificado para que las utilice el programa /52/.

Como complemento a esta utilidad, se ha desarrollado otra que permite definir conjuntos de instrucciones ('macros'). Estos conjuntos de ordenes se pueden definir interactivamente. En este caso se pueden ejecutar secuencialmente al acabar de definirlos, o almacenarlos para ejecutarlos cada vez que sean llamados. También pueden definirse externamente al programa, y ser llamados posteriormente por este.

Las particularidades de la parte gráfica de la interacción, que realiza el procesador GRAFIC, se estudian detalladamente en el capitulo 7.

La realimentación del sistema se realiza por medio de:

a) escritura de las instrucciones recibidas ('ecos');

b) escritura de mensajes indicando las operaciones en curso, y

c) detección de errores y anomalias, con escritura de mensajes orientativos sobre las posibles causas.

El sistema imprime mensajes, a modo de "acuse de recibo", para indicar que una instrucción ha sido recibida. El mensaje puede ser un simple eco de la instrucción (p.e. durante la lectura de datos), o una indicación de la tarea que ha comenzado. Cuando la tarea en curso es larga, el sistema imprime un mensaje indicando que la está realizando, seguido por otro que indica cuando está conlcuida. Si la tarea es excesivamente larga (p.e. algunas resoluciones de sistemas de ecuaciones) se imprimen mensajes intermedios para indicar como va progresando.

Cuando se detecta cualquier anomalía se imprime un mensaje indicando la causa. Se ha distinguido dos tipos de anomalias, en función de que su gravedad impida seguir la ejecución (se denominan mensajes de ERROR) o que solo se vea comprometida la tarea en curso (son mensajes de ATENCION). En el primer caso la ejecución se aborta despues de imprimir el mensaje de error y salvar toda la información posible. En el segundo caso se finaliza la tarea en curso intentando que se modifiquen lo mínimo posible la información afectada y el flujo del programa, y se permite que el usuario siga la ejecución.

Para que la interacción sea efectiva, la detección de errores debe estar muy depurada, de forma que se detecte tempranamente cualquier intento de uso ilegal y se minimicen sus efectos. De todas formas es imposible predecir todos los posibles usos incorrectos, por lo que no se puede asegurar el funcionamiento correcto del sistema despues de algunos tipos de fallos. Para paliar cualquier posible secuela oculta de un fallo (y para evitarla en un futuro) se ha incluido en el sistema un menú de operaciones sobre la información que permite a un usuario experto trabajar directamente sobre cualquier unidad de información (viendo y modificando su contenido).

El modelo del usuario se basa en el modelo de análisis por elementos finitos, que está firmemente establecido y resulta familiar a cualquier posible usuario de este sistema /51/. El modelo se amplia con el de diseño que, aunque no es tan conocido, está suficientemente contrastado.

Resaltar, por último, que a fin de facilitar el uso del sistema sin hacerlo muy rígido, en la mayor parte de los dialogos el sistema indica un valor por defecto (valor que tomará si no se le introduce

otro) para los parámetros para los que pide un valor. La puesta a punto de los valores por defecto es una tarea costosa, pero su utilización permite que usuarios poco entrenados saquen gran rendimiento al sistema.

CAPITULO 7

REPRESENTACION GRAFICA DEL DISEÑO OPTIMO DE ESTRUCTURAS

7.1 INTRODUCCION

En un sistema de diseño automático, el flujo de la ejecución no es modificable. Por lo tanto, el usuario no precisa conocer como está evolucionando el proceso. En este caso, las necesidades de representación gráfica se limitan a considerar la entrada de datos y la salida de resultados, a fin de proveer un mecanismo cualitativo que ayude a detectar fallos en la entrada, y los resultados críticos en la salida. Este aspecto de la representación gráfica es el más extendido, y su implementación en el sistema DISSENY se considera en el siguiente apartado de este capítulo. Por el contrario, en un proceso interactivo de diseño, el usuario debe reconducir el proceso en función de la forma en que este progrese. Para tomar las decisiones de como reconducir el proceso, el usuario debe disponer de toda la información del mismo.

Una de las características generales de los ordenadores es el efecto multiplicador que ejercen sobre la información que manipulan. En el caso que nos concierne, el ordenador genera abundante información sobre el comportamiento de la estructura en estudio; pero genera también mucha información sobre la evolución del propio estudio. Toda esta información (mayoritariamente numérica) es poco aprovechable cuando se presenta en forma de tablas numéricas.

De lo anterior se deduce que se debe buscar una forma de presentar al usuario la información del proceso, que resulte fácil de interpretar y tenga capacidad para presentarla en una gran variedad de formas distintas (a fin de hacer patentes todas las variaciones del proceso). Estas dos características son las que hacen que la presentación gráfica de esta información sea la mejor solución. Esta representación gráfica afecta a información muy variada, y de un tipo que no resulta habitual a la mayoría de los posibles usuarios. Por ello, se requiere el empleo de formas de visualización innovadoras que permitan expresar

todas las características del proceso /23/. El tercer apartado del capitulo se dedica a describir la propuesta de formas de visualización del proceso de diseño adoptada en el sistema DISSENY.

Antes de entrar en la descripción de como se han realizado las representaciones vamos a comentar dos aspectos, menores, pero que afectan mucho a la implementación de las mismas:

- la interacción gráfica, y

- la distribución del área de trabajo.

7.1.1 Interacción gráfica.

En el capítulo 6 se ha definido el lenguaje de comandos, de tipo menú, que se ha adoptado para el sistema DISSENY. En la parte gráfica del sistema, es habitual que el menú aparezca en la pantalla, y que pueda seleccionarse la acción requerida "señalandola". Es decir, que el usuario utilice cualquiera de los periféricos gráficos habituales en lugar del teclado.

El periférico mas habitual es el ratón; que, además, se adapta muy bién a la tarea de seleccionar puntos de la pantalla. Por ello, en el sistema DISSENY se ha previsto la implementación de las funciones habituales del ratón. La primera de ellas es la de seleccionar opciones de un menú. Para lo cual se ha dispuesto de forma visible en la pantalla el menú disponible. La segunda es la de detectar la selección de puntos de la pantalla en los que se vá a llevar a cabo alguna acción (escribir un texto, inquirir sobre la parte de la estructura que está representada en dicho punto, etc). No obstante, dado que la parte gráfica del sistema está aún en desarrollo, se ha preferido no añadir la complicación adicional de gestión del ratón; dejando su implementación para cuando el esquema del sistema esté completamente establecido.

7.1.2 Area de trabajo.

El área habitual de trabajo interactivo es la pantalla del ordenador. Como ya hemos comentado en el capítulo 5, la división de la pantalla para albergar toda la información útil en cada momento, tiene gran importancia debido a la escasez de la misma. La división adoptada se muestra en la figura (7.1).

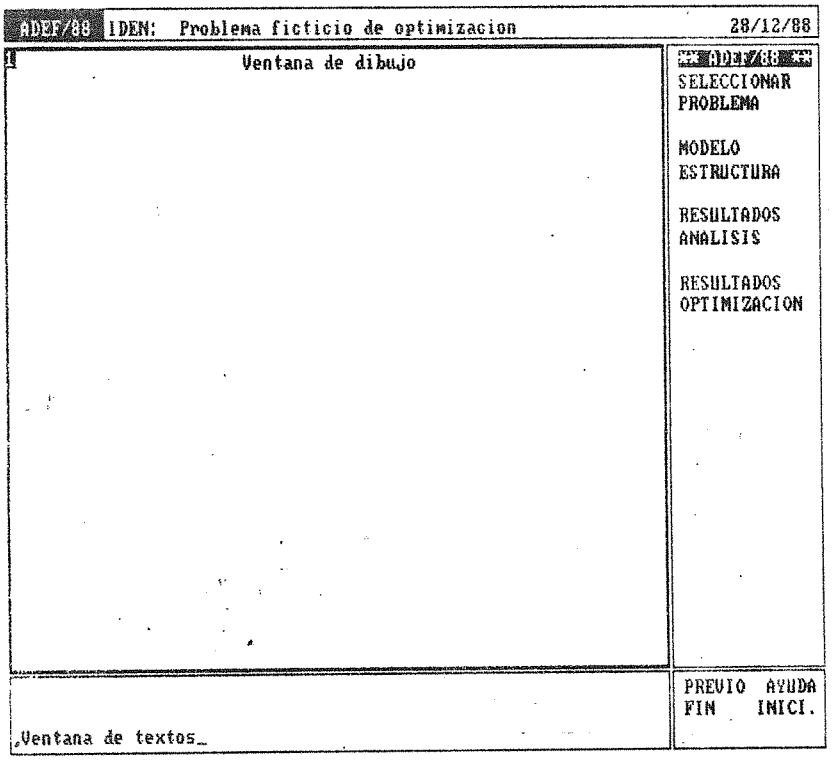

Figura 7.1. División de la pantalla.

En esta figura se observa que la pantalla se ha dividido en cinco zonas:

- La ventana de identificación; que permite conocer en cada momento el problema que se está tratando. Aporta además información complementaria (fecha, etc).

- La ventana de menú; que presenta el menú de todas las operaciones utilizables por el usuario en cada momento.

- La ventana de ayuda; que muestra las operaciones generales de control de la ejecución, disponibles en todo momento.

- La ventana de textos; que muestra el diálogo entre el ordenador y el usuario (preguntas, avisos, etc).

- La ventana de dibujo; que es la zona de la pantalla en la que se llevan a cabo las representaciones.

La posibilidad de cambiar la posición relativa de las ventanas no se ha incluido por haberse contrastado que la disposición adoptada es la más extendida en los programas similares existentes. Sin embargo, todas las ventanas, salvo las de dibujo son opcionales; la opción de incluirlas o no se especifica en un fichero de configuración que puede ser modificado en cualquier momento. También son parámetros variables (contenidos en el fichero de configuración) las dimensiones de todas las ventanas.

El sistema utiliza la ventana gráfica para representar, en función de las órdenes y los parámetros introducidos por el usuario. Dada la naturaleza del problema a tratar, resulta conveniente que el usuario pueda fragmentar esta ventana en varias subventanas, sobre las que pueda representar información diferente para contrastarla. Por tanto, se ha incluido la posibilidad de que el usuario redefina interactivamente la subdivisión de la ventana de dibujo (fig. 7.2).

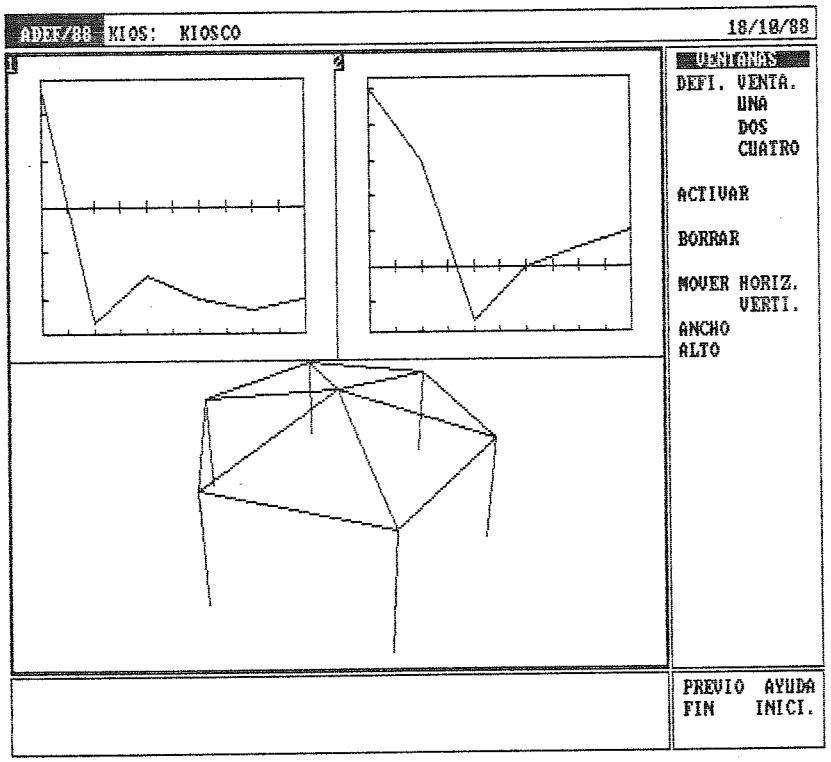

Figura 7.2. Ventanas de dibujo.

La principal finalidad de la ventana de textos es la de 'confinar' la parte alfanumérica de la ejecución (para que no se superponga a la parte gráfica, ni necesite de constantes vaivenes entre un modo gráfico y otro alfanumérico de la pantalla). Además, conforme el usuario va adquiriendo experiencia sobre el uso del sistema, resulta normal que disminuya el área de esta ventana, para ganar área de dibujo /16/. Por ello, se ha diseñado esta ventana de forma que presente tantas líneas como quepan (comenzando desde la última hacia atrás), y manteniendo en memoria las líneas anteriores; de forma que el usuario pueda realizar un "scroll" (mover la parte vista del texto para inspeccionar las últimas líneas escritas).

7.2 REPRESENTACION DEL MODELO DE LA ESTRUCTURA Y DE LOS RESULTADOS DEL ANALISIS

Se pretende representar gráficamente el modelo de cálculo de las estructuras a analizar. Esta representación sirve como complemento al preprocesador de análisis, permitiendo comprobar la bondad de los datos suministrados para la codificación de dicho modelo. Teniendo en cuenta la forma en que se ha idealizado la estructura (capítulo 4), las necesidades de representación se pueden dividir en:

- nudos;

- elementos, e

- información complementaria (tipo de material, tensión, etc).

Por otra parte, para que esta representación ayude a encontrar fallos en la definición del modelo, se precisa que cada uno de sus componentes se pueda representar opcionalmente, en función de una selección realizada por el usuario. Es decir, que se debe poder elegir entre representar o no cada uno de los componentes. E incluso, se debe tener la opción de representar cada componente (o grupo de componentes) distinguiendolo del resto por medio de tipos de línea, colores, etc.

También se pretende representar los resultados del proceso de análisis. Con la generalización del proceso de diseño, los resultados del análisis deben acabar convirtiendose en información interna del proceso. No obstante, dado que el proceso de diseño aún no está suficientemente extendido, todavía resulta conveniente que el usuario disponga de información gráfica sobre estos resultados. De esta forma

se puede hacer una evaluación cualitativa de la bondad del diseño elegido. Además parte de los resultados del análisis siguen siendo útiles como resultados del diseño (por ejemplo la deformada).

Las características que debe tener el procesador de representación para los resultados de análisis son muy similares a las de la representación de los datos: posibilidad de seleccionar la parte a representar; posibilidad de diferenciar distintas partes de la representación, etc.

Para llevar a cabo estas representaciones hay dos alternativas básicas:

- utilizar un paquete CAD disponible comercialmente; o

- desarrollar un programa de representación específico.

La primera alternativa tiene la ventaja de que se puede aprovechar toda la capacidad gráfica, e interactiva, de los sistemas disponibles en el mercado. El inconveniente estriba en la necesidad de desarrollar una interfase que prepare la información que se quiere representar para que pueda ser asimilada por el sistema CAD. Interfase que será bastante compleja si el sistema elegido no puede trabajar directamente con modelos tridimensionales. Por el contrario, desarrollar un programa específico, tiene la clara ventaja de que este se adaptará perfectamente a las necesidades de representación del sistema; y se debe tener presente que se pretende representar información de diferentes características, tal como se verá mas adelante. A cambio, habrá que implementar algunas opciones que son habituales en cualquier sistema CAD. Por lo tanto, se trata de comparar el costo de desarrollo de un programa específico frente a la generalidad y elevado número de opciones de un programa comercial.

La solución adoptada ha sido la de desarrollar un procesador de representación específico para el problema a tratar, porque se ha considerado mas importante que se adapte a las necesidades del sistema, que el que disponga de mas opciones. No obstante, el desarrollo se ha hecho a partir de una librería gráfica (HALO /54/), con lo que disponer de las opciones elementales mas habituales ha sido inmediato. Además, de esta forma se ha adquirido la experiencia suficiente para afrontar la representación del proceso de diseño.

El procesador gráfico se ha desarrollado de forma que permita la representación en perspectiva cónica de los modelos de las estructuras en estudio (capítulo 5). Tal como se ve en la figura (7.3), el usuario puede modificar todos los parámetros de la perspectiva (pudiendo conseguir, como ya se ha comentado en el capítulo 5, una representación en diédrico). En la figura se incluye el menú de opciones disponibles para controlar la forma de la representación. En la ventana de texto puede verse la información de los parámetros que controlan esta representación.

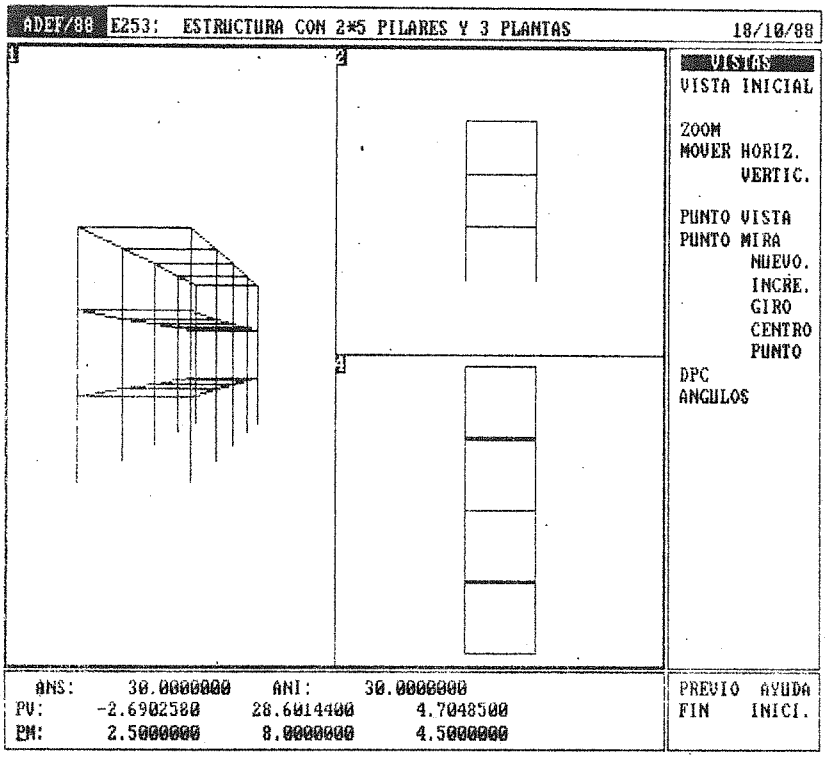

Figura 7.3. Formas de representación.

Para que se puedan comprobar los datos del problema, se dispone de diferentes posibilidades de seleccionar partes de la estructura (elementos del mismo material, con las mismas propiedades, etc). También se ha incluido (aunque en la versión actual no estan implementadas) las representaciones de las restricciones y las cargas definidas. En la figura (7.4) se puede ver un ejemplo en el que se han representado los elementos con tipos de líneas diferentes en función del tipo de propiedades de su sección recta. En dicha figura puede

verse el menú de opciones que se han contemplado para seleccionar las representaciones.

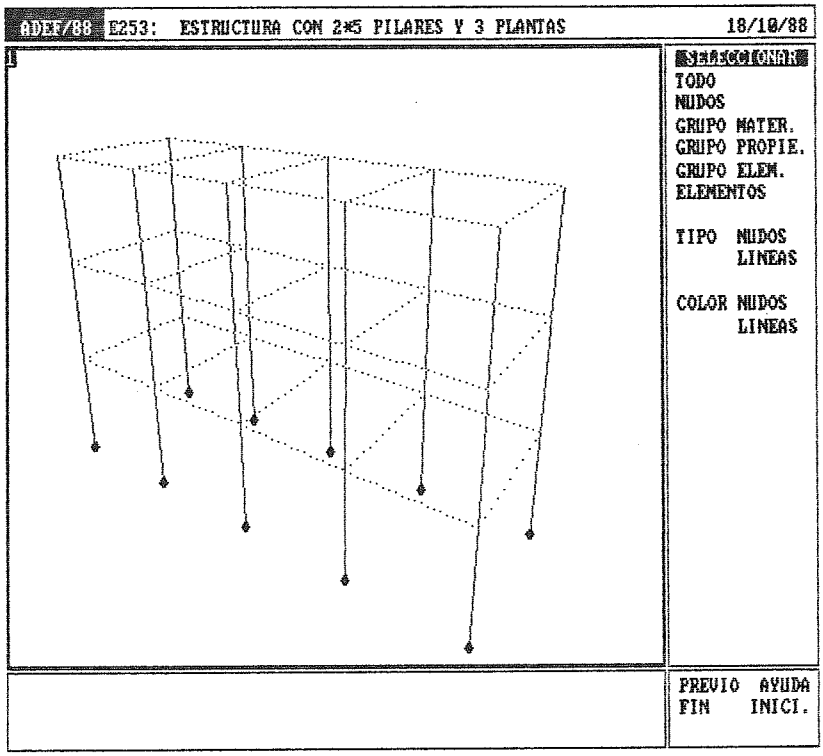

Figura 7.4. Modelo de una estructura.

En cuanto a la representación de los resultados del análisis, se puede representar la deformada de la estructura (fig. 7.5). Dicha deformada es la obtenida modificando la posición de los nudos en función de su desplazamiento. Un factor de amplificación permite escalar los desplazamientos, a fin de contrastar mejor las deformaciones. No se ha incluido el cálculo de las deformaciones internas de los elementos por ser muy costoso en tiempo de cálculo y no ser significativo en descomposiciones por elementos finitos (únicamente es habitual para elementos barra de nudos rígidos).

Está prevista la implementación de la representación del resto de los resultados del análisis. Así, por ejemplo, se representarán los esfuerzos en los elementos por medio de una representación del modelo de la estructura en el que cada elemento esté diferenciado (por color o tipo de línea) en función del tipo de esfuerzo (p.e. tracción/compresión) o del valor de dicho esfuerzo.

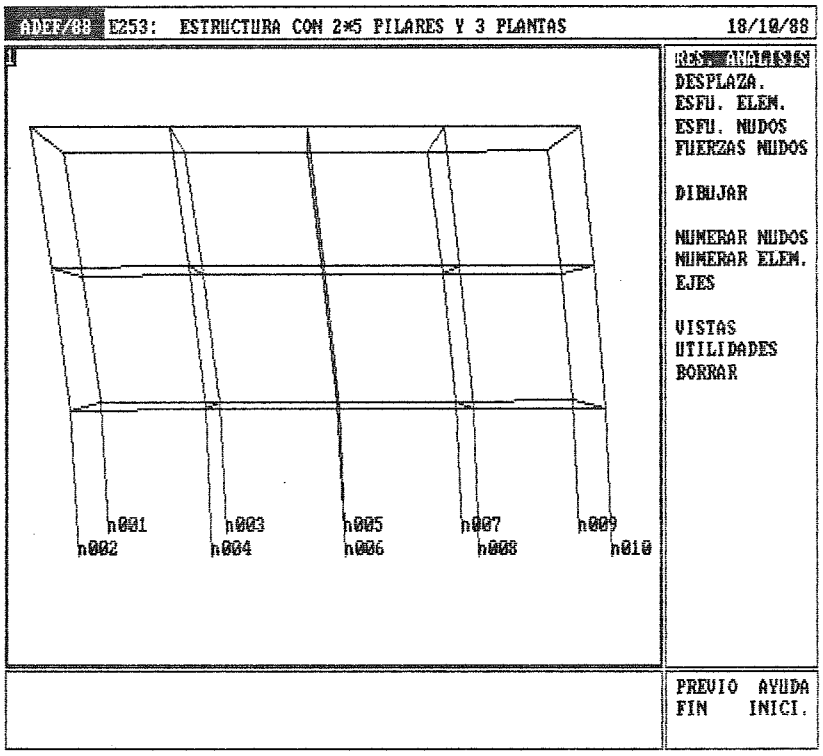

Figura 7.5. Resultados del análisis.

En todos los casos, hay una serie de informaciones complementarias que se pueden añadir a las representaciones. Estas informaciones son:

- los ejes de referencia para las coordenadas de los nudos;
- la numeración de los nudos que se soliciten, y
- la numeración de los elementos que se soliciten.

7.3 REPRESENTACION GRAFICA DEL PROCESO DE DISEÑO

7.3.1. Introducción.

Dado que el proceso de diseño es iterativo, para que un diseñador experimentado pueda evaluar su progreso precisa conocer tanto el estado actual como la evolución seguida desde el principio. Es decir, que la representación del proceso de diseño se puede dividir en:

- representación de la 'historia' del proceso, y

- representación del estado actual del diseño.

La forma mas sencilla de conocer la historia del proceso consiste en representar la evolución de las magnitudes implicadas en él. La evolución se refiere a los sucesivos pasos del proceso, que se denominan habitualmente 'iteraciones'. En el siguiente epígrafe vamos a considerar las magnitudes del proceso que resulta conveniente representar de esta forma; así como algunos detalles significativos de la forma de implementar esta representación.

Una forma menos convencional de representar la historia del proceso, es plantear relaciones cruzadas entre la evolución de diferentes magnitudes del mismo. Algunas sugerencias sobre la forma de emplear este tipo de representación se incluyen despues de describir las evoluciones.

La representación del estado del diseño es la parte que requiere mas esfuerzo innovador. Esto es debido, por una parte a la complejidad de la información a representar, y por otra parte a lo escasa que resulta la información disponible respecto a la que sería deseable para que el usuario dispusiese de un panorama completo del diseño. La última parte del capítulo es la dedicada a explicar con detalle toda la experiencia adquirida en la implementación de este tipo de representación en el sistema DISSENY.

7.3.2. Evoluciones del proceso de diseño

La representación de la variación (en las sucesivas iteraciones del proceso de diseño) de las diferentes cantidades implicadas permite conocer la evolución de dicho proceso. Tal como se ha visto en el capítulo 5, la forma mas apropiada de representar estas evoluciones es por medio de diagramas. En la figura (7.6) se muestran tres ejemplos

de las diferentes opciones que permite el sistema DISSENY para representar este tipo de funciones: a) diagramas de polilínea continua; b) diagramas de curva suave, y c) diagrama de puntos.

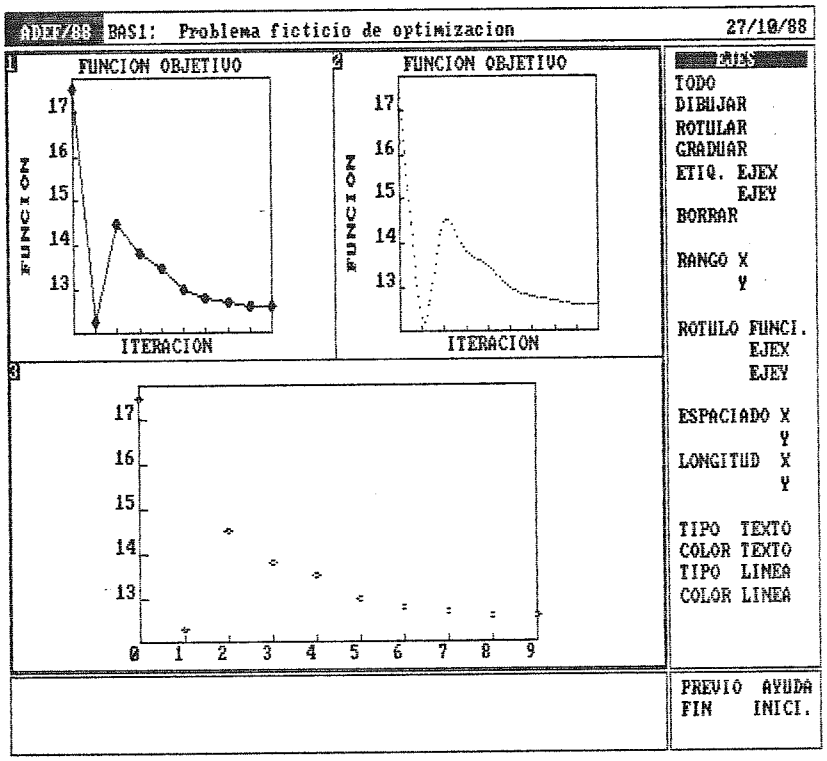

Figura 7.6. Diferentes representaciones de una función.

La variación del proceso de diseño es una evolución histórica, aunque no depende del tiempo sino de una variable discreta: las iteraciones. Es decir, que la relación que se pretende representar es la de cada información del proceso en función las iteraciones del mismo. En la representación en diagramas, habitualmente se representa en ordenadas la magnitud que varía con el tiempo (en este caso las iteraciones), y en abcisas el valor de la propia magnitud.

Tal como se ha indicado en el capítulo 5, es fundamental en las representaciones de diagramas y funciones, especificar el rango de la representación; así como controlar la identificación de la gráfica representada. En el menú de la figura (7.6) se muestran los comandos disponibles para controlar esta representación.

Las magnitudes del proceso de diseño que deben representarse son:

- Las variables de diseño; que permiten conocer como va modificandose el diseño.

- La función objetivo; para conocer la mejora que implica el diseño (puede usarla el diseñador como criterio cualitativo de parada).

- Las restricciónes; especialmente las que son activas, para seguir su influencia en el proceso.

Estas magnitudes puede ayudar al usuario experto a conocer como ha evolucionado el proceso, y a intentar preveer cual va a ser la evolución inmediata. De forma que le permita adelantarse al proceso y sugerir cambios que lo aceleren. Para que esta decisión no sea tan intuitiva, conviene que disponga también de la posibilidad de representar la evolución de las sensibilidades de estas mismas magnitudes (se puede decir que así conoce tanto la variación de la magnitud como la 'velocidad' de dicha variación).

En la figura (7.7) se puede ver un ejemplo de las diferentes evoluciones que se pueden representar en el sistema DISSENY.

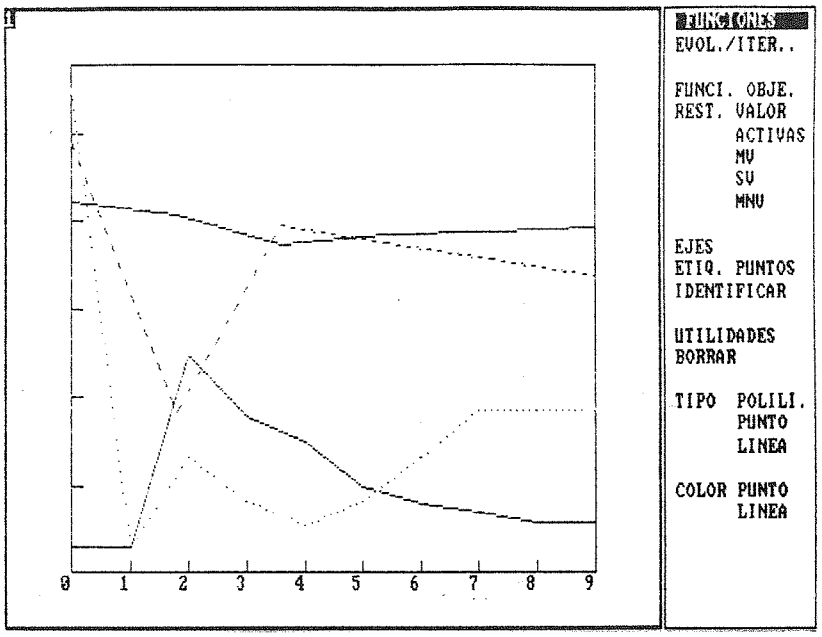

Figura 7.7. Evoluciones de diferentes magnitudes.

En la figura (7.7) puede observarse el escalado que han sufrido las funciones representadas (0.25 para la función objetivo y 5.0 para las restricciones). Tal como se ha indicado en el capítulo 5, este escalado es una forma de conseguir representar funciones diferentes sobre un mismo rango de variación. Además, de las alternativas posibles (diferentes ejes, etc) es la más conveniente, porque permite introducir en la representación el escalado que sufren estas magnitudes en el proceso de normalización que se emplea en algunos algoritmos de optimización.

7.3.3 Relaciones entre magnitudes del proceso de diseño.

Ya hemos comentado que una representación alternativa a la de superponer las evoluciones de dos magnitudes a lo largo de las iteraciones, consiste en representar una de estas magnitudes en ordenadas y la otra en abcisas. Es decir, que sobre este sistema de ejes se pueden representar las parejas de valores que toma cada una de las magnitudes en las diferentes iteraciones del proceso de diseño. Los puntos así obtenidos se pueden unir por una polilinea, ordenados por iteraciones. Un ejemplo de representación de este tipo puede verse en la figura (7.8).

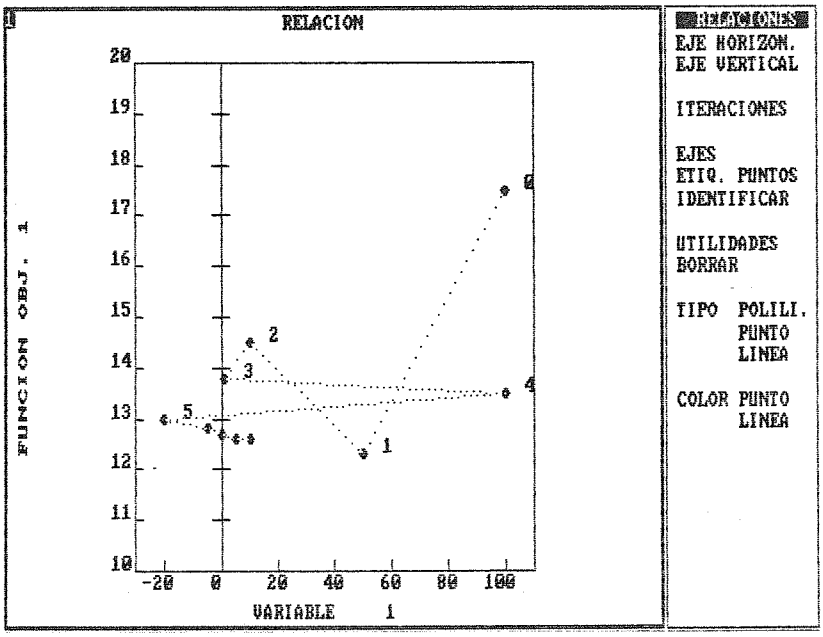

Figura 7.8. Relación entre la función objetivo y una variable de diseño

Esta representación alternativa puede dar lugar a una gráfica de la que no se pueda extraer información. Esto puede deberse tanto a que la información sea demasiado compleja o poco adaptada a este tipo de representación, como a que el usuario tenga poca experiencia en el empleo de las mismas. Así, el ejemplo de la figura es un caso de representación confusa de la que difícilmente se puede estraer alguna consecuencia. No obstante, Bertin /11/ recoge un ejemplo en el que comparando algunas de estas gráficas "confusas" se llega a obtener una información (de semejanza, secuencias, cadencias, etc) que permite agruparlas. En definitiva, consideramos que este tipo de representación permitirá (con mas experiencia sobre ellas) poner de manifiesto relaciones que en la anterior representación por iteraciones (mas convencional) no quedaban patentes.

7.3.4 Espacio de Diseño.

Por representación del espacio de diseño se entiende la representación de la función objetivo y las restricciones, en función de las variables de diseño. La idea, lanzada por Fleury, de representar el espacio de diseño /23/, es interesante no solo como auxilio en la enseñanza del proceso de optimización, sino como un medio de acelerarlo. Observando la función objetivo y las restricciones se pueden detectar mínimos locales. También se pueden representar sobre el espacio de diseño los puntos de diseño obtenidos por el algoritmo de optimización en las sucesivas iteraciones; lo que puede permitir reconducir el proceso de forma que se se acelere la convergencia, acercando el punto de diseño al punto óptimo. Se trata en definitiva de una interpretación gráfica del proceso de diseño (capítulo 2), que permita al usuario conocer cualitativamente el estado del diseño.

Puesto que el problema de diseño suele considerar un número elevado de variables (n), se trata de representar funciones n-dimensionales. Una forma de representar funciones n-dimensionales consiste en definir una nueva coordenada (n+1) sobre la que se definen los valores de la función. Se obtiene así una hipresuperficie n+1-dimensional que representa el valor que toma la función para cualquier posible valor de las variables (capítulo 5).

En la figura (7.9) puede verse la representación del espacio de diseño para el problema:

$$\text{minimizar } f(x)= x_1^2+x_2^2 \qquad (7.1)$$
$$\text{sujeto a } g(x)= x_1+x_2-b \geq 0$$

En este caso se ha representado sobre un sistema de tres dimensiones (dos variables mas la función objetivo). Las restricciones son del tipo $g(x)\geq 0$ (capítulo 2), y por tanto, se suele representar la función $g(x)=0$ que marca el límite de la zona válida de diseño.

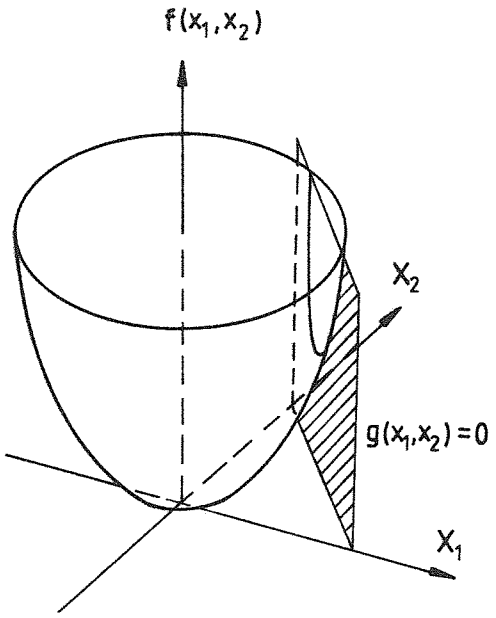

Figura 7.9. Espacio de diseño (tridimensional) en proyección

La forma habitual de representar el espacio de diseño, consiste en sustituir la dimensión correspondiente a la función objetivo por curvas de valor constante ($f=k_1$, $f=k_2$,...) sobre el subespacio formado por las variables (fig 7.10).

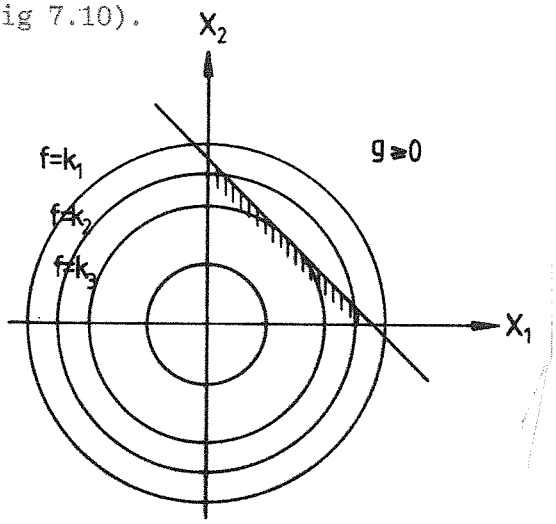

Figura 7.10. Espacio de diseño (bidimensional) con función objetivo dada por curvas de valor constante

Sin embargo, la representación sobre pantalla, debe ser forzosamente bidimensional. Por lo que podremos representar directamente espacios de dos dimensiones; o indirectamente (con proyecciones seudoperspectivas) espacios de tres dimensiones. En los problemas de orden superior únicamente podremos representar subespacios de diseño. Así, se mantienen constantes todas las variables menos dos (o tres), y se representan sobre un sistema de ejes cartesianos las curvas que resultan de "cortar" las hipersuperficies de las funciones a representar por el hiperplano donde todas las variables toman un valor fijo salvo las que forman el subespacio.

Por otra parte, no se tiene información suficiente para representar directamente estas funciones, de las que generalmente no se dispone de expresión analítica. Recordemos que en el capítulo 3 se ha planteado un problema semejante con la búsqueda unidimensional. De las soluciones que alli se han apuntado, la de calcular valores de la función no parece conveniente en este caso, porque se pretende que el procesador de representación gráfica tenga la máxima independencia posible. Además, la representación gráfica solo pretende que el usuario disponga de una apreciación cualitativa. Por ello, la alternativa de aproximar las funciones a representar es la mas apropiada. Se consigue una representación bastante parecida a la función real, poco costosa en cálculos, e independiente del resto del sistema.

A continuación vamos a describir los aspectos mas significativos de las soluciones adoptadas en el sistema DISSENY para estos dos problemas de la representación de espacios de diseño.

7.3.4.1. Subespacios de diseño

El problema de diseño depende de un número de variables (n) generalmente mayor que dos. Por lo tanto, para representar el espacio de diseño, se debe recurrir a la representación de un subespacio formado por dos de sus variables (por ejemplo x_1 y x_2), manteniendo constantes el resto (x_3,...,x_n).

Debe notarse que en función del valor constante que se asigne al resto de variables, el subespacio obtenido será diferente. En la figura (7.11) se ha representado, sobre un espacio tridimensional, la

superficie formada por los puntos en donde una función dada $f(x_1,x_2,x_3)$ toma un valor constante k.

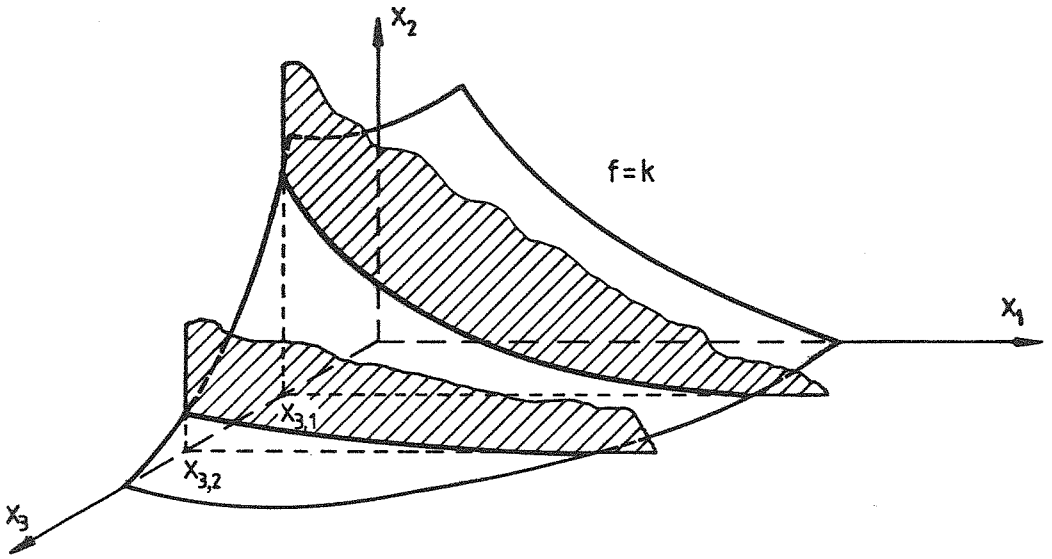

Figura 7.11. Superficie de valor constante de una función de tres variables

En la figura (7.12) puede verse que la curva f= k para el subespacio (x_1,x_2) depende del valor constante que se asigne a la variable x_3

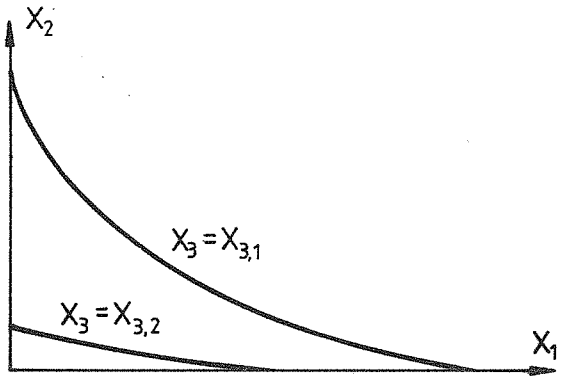

Figura 7.12. Curvas de valor constante de una función de tres variables para diferentes valores de una variable

Por lo tanto, para que la representación sea coherente, todas las curvas deberán estar referidas al mismo plano, definido por

$$x_j = x_j^i \qquad j = 3,\ldots,n \qquad (7.2)$$

siendo x_j^i valores arbitrarios en principio; si bien será conveniente (para mayor exactitud de las aproximaciones a realizar) tomar como valores los del punto de diseño.

7.3.4.2. Aproximación de funciones

A partir de la información de una función, disponible para la mayoría de los algoritmos de optimización (su valor y el de sus derivadas parciales, respecto a todas las variables, para un punto), pretendemos obtener una ecuación que nos de ,de forma aproximada, la curva que representa los puntos de un subespacio de diseño en donde dicha función toma un cierto valor.

Para resolver este problema recurrimos a aproximaciones basadas en derivadas de primer orden.

Por lo tanto, el problema lo podemos formular como:

Dado un punto del espacio de diseño (n-dimensional)
$x_{o1}, x_{o2}, \ldots, x_{on}$,

y conocidos:

f_o	valor que toma una función en ese punto, (puede ser la función objetivo o una restricción);
$\delta f/\delta x_j\|_o$ $j= 1, \ldots, n$	derivadas parciales de la función en ese punto.

(7.3)

Determinar la expresión de la curva de todos los puntos del subespacio de diseño (x_1, x_2) que cumplen

$x_j = x_j^i$ $\quad j= 3, \ldots, n$
$f = k$

El desarrollo en serie de Taylor ofrece aproximaciones de bastante calidad para este problema. Si bién, se debe llegar a un compromiso entre el costo de los cálculos requeridos para construir la aproximación y el tamaño del dominio sobre el cual dicha aproximación es válida. Así,la aproximación más simple para este problema consiste en utilizar un desarrollo en serie de Taylor truncado en las primeras derivadas

$$f = f_o + \sum_{j=1}^{n} \delta f/\delta x_j \Big|_o (x_j - x_{oj}) \qquad (7.4)$$

A partir de esta expresión se obtiene una aproximación lineal de la función para las variables x_1 y x_2

$$k - f_o - \sum_{j=3}^{n} \delta f/\delta x_j \Big|_o (x_j{}^i - x_{oj}) = \begin{matrix} \delta f/\delta x_1 \Big|_o (x_1 - x_{oj}) + \\ \delta f/\delta x_2 \Big|_o (x_2 - x_{oj}) \end{matrix} \qquad (7.5)$$

Sin embargo, muchas de las restricciones de un espacio de diseño, e incluso la función objetivo, suelen ser no lineales respecto a las variables de diseño. En este caso, para obtener una aproximación mas exacta se deberían emplear mas términos del desarrollo de Taylor; lo cual ya hemos comentado que requiere cálculos costosos de derivadas de orden superior (derivadas que no suelen calcularse en la mayoría de algoritmos de optimización). La otra alternativa es hacer un cambio de variables que linealicen la función. Es decir, definir unas variables

$$y_i = y_i(x) \qquad (7.6)$$

respecto a las cuales la función f sea lineal.

La expresión mas sencilla para estas variables es

$$y_i = 1/x_i \qquad (7.7)$$

con la cual obtenemos la aproximación inversa

$$f = f_o + \sum_{j=1}^{n} \delta f/\delta x_j \Big|_o (x_j - x_{oj})\ x_{oj}/x_j \qquad (7.8)$$

Un tercer tipo de aproximación, híbrida de las dos anteriores y más conservadora que ellas se puede deducir restando la aproximación directa de la recíproca:

$$f_D - f_R = \sum_{j=1}^{n} \delta f/\delta x_j \Big|_o (x_j - x_{oj})^2\ 1/x_j \qquad (7.9)$$

En esta expresión se observa que la influencia de cada sumando depende del signo de cada variable (x_j) y de cada derivada ($\delta f/\delta x_j|_o$). De forma que si el producto de la variable por la derivada es positivo contribuye a que el sumando haga más positiva a la aproximación directa que a la inversa y viceversa. Como las restricciones se expresan de la forma $f \geq 0$ y la función objetivo es un valor a minimizar, la aproximación mas conservadora de cualquiera de estas funciones es la que la hace mas negativa. Así, se puede expresar:

$$f = f_o + \sum_{j=1}^{n} \delta f/\delta x_j|_o \ (x_j - x_{oj})\ B_j \qquad (7.10)$$

siendo

$$B_j = 1 \qquad \text{si } x_j\ \delta f/\delta x_j|_o \leq 0$$

$$B_j = x_{oj}/x_j \qquad \text{si } x_j\ \delta f/\delta x_j|_o > 0$$

En definitiva, tenemos:

$$f = k = f_o + \sum_{j=1}^{n} \delta f/\delta x_j|_o \ (x_j - x_{oj})\ B_j + \qquad (7.11)$$
$$\delta f/\delta x_1|_o \ (x_1 - x_{o1})\ B_1 +$$
$$\delta f/\delta x_2|_o \ (x_2 - x_{o2})\ B_2$$

siendo

$B_j = 1$ si la aproximación es híbrida y $x_j\ \delta f/\delta x_j|_o \leq 0$, o si la aproximación es directa;

$B_j = x_{oj}/x_j$ si si la aproximación es híbrida y $x_j\ \delta f/\delta x_j|_o > 0$, o si la aproximación es inversa.

Definiendo:

$$A = k - f_o - \sum_{j=3}^{n} \delta f/\delta x_j|_o \ (x_j - x_{oj})\ B_j \qquad (7.12)$$

resulta

$$A = \delta f/\delta x_1|_o \ (x_1 - x_{o1})\ B_1 + \delta f/\delta x_2|_o \ (x_2 - x_{o2})\ B_2 \qquad (7.13)$$

que es la curva de intersección de la función f=k con el hiperplano (de dimensión n-2) definido por

$$x_j = x_j^i \qquad j = 3, \ldots, n \qquad (7.14)$$

El conocimiento del sentido físico de las restricciones puede ayudar en la elección de alguna de estas tres aproximaciones (o de cualquier otra que se defina), para conseguir aproximaciones de la mayor calidad posible a partir de la información disponible. La primera aproximación de las definidas (la directa) se adapta bién a funciones objetivos lineales y a restricciones que dependen directamente de las variables (p.e. es el caso de las restricciones de tensión para estructuras de barras de nudos articulados, cuando las variables son las áreas de los elementos). La aproximación inversa da buenos resultados cuando las funciones dependen de las inversas de las variables de diseño (las restricciones de desplazamiento para el caso anterior). La aproximación híbrida define un espacio de diseño convexo (muy conservador) por lo que resulta conveniente como primera aproximación, aunque puede llevar a regiones factibles mucho mas restringidas que las reales.

7.3.4.3. Algoritmos de evaluación y representación

Dando valores en la expresión (7.13) podemos calcular un número finito de puntos que unidos por una linea continua que pase por ellos nos permite representar la función f en el subespacio de diseño. Para ello deberemos discutir previamente la ecuación, a fin de determinar los puntos que se deben calcular, para obtener una representación sin errores con el mínimo número posible de puntos calculados.

Para el cálculo de estas aproximaciones se debe tener en cuenta que en el caso particular de que

$$\delta f/\delta x_1|_o = 0 \qquad \delta f/\delta x_2|_o = 0 \tag{7.15}$$

la función es independiente de las dos variables, en las proximidades del punto considerado. Por tanto, no se puede utilizar una aproximación que considere únicamente las primeras derivadas.

En el caso de aproximación directa, la expresión (7.13) queda convertida en:

$$A= \delta f/\delta x_1|_o \ (x_1-x_{o1}) + \delta f/\delta x_2|_o \ (x_2-x_{o2}) \tag{7.16}$$

que se puede poner como:

$$(x_2-(x_{o2}+A))/(x_1-(x_{o1}+A))= -\delta f/\delta x_1|_o/\delta f/\delta x_2|_o \qquad (7.17)$$

es decir, que se trata de una recta que pasa por el punto $((x_{o1}+A),(x_{o2}+A))$ y tiene pendiente $-\delta f/\delta x_1|_o/\delta f/\delta x_2|_o$

Notese que A se puede interpretar como una "traslación" del punto (x_{o1},x_{o2}), que depende de: a) la diferencia entre el valor conocido de la función y el buscado, y b) la diferencia entre el plano de referencia conocido y el buscado.

Para representar esta recta sobre unos ejes hay que tener presente que el rango de variación de las dos variables define una "ventana" del subespacio:

$$\begin{aligned} x_{1min} \leq x_1 \leq x_{1max} \\ x_{2min} \leq x_2 \leq x_{2max} \end{aligned} \qquad (7.18)$$

Por tanto, se pueden calcular los dos puntos de intersección de dicha recta con la ventana y unirlos.

Se pueden dar varias alternativas según la situación de la línea. el problema es básicamente el tratado por el algoritmo de recortado de un segmento de Cohen-Sutherland. Con la ventaja de que se trata de una línea infinita en lugar de un segmento de la misma. En este caso pueden ocurrir dos cosas:

a) que la línea atraviese la ventana;

b) que no la atraviese;

Para decidir en que caso estamos bastará comprobar si la línea corta al marco en algún punto.

El algoritmo a utilizar puede ser:

. Sustituir la coordenada x_{1min} en la ecuación de la línea y comprobar si la x_2 resultante cumple

$$x_{2min} \leq x_2 \leq x_{2max} \qquad (7.19)$$

. Repetir el proceso para x_{1max}, x_{2min} y x_{2max}

El resultado puede ser:

- No se cumple la condición en ninguno de los cuatro lados. Significa que la línea no atraviesa la ventana.

- Se cumple en dos lados. Los puntos obtenidos en esos lados serán los extremos del segmento visible.

- Otro resultado. Hay un error (por ejemplo porque la ecuación no corresponde a una recta).

En el caso de aproximación inversa, la expresión (7.13) queda convertida en

$$A= \delta f/\delta x_1\big|_o\ (x_1-x_{o1})\ x_{o1}/x_1 + \delta f/\delta x_2\big|_o\ (x_2-x_{o2})\ x_{o2}/x_2 \qquad (7.20)$$

Es decir

$$A= \delta f/\delta x_1\big|_o\ (x_{o1}-x_{o1}^2/x_1) + \delta f/\delta x_2\big|_o\ (x_{o2}-x_{o2}^2/x_2) \qquad (7.21)$$

y definiendo

$$C= A - x_{o1}\ \delta f/\delta x_1\big|_o - x_{o2}\ \delta f/\delta x_2\big|_o \qquad (7.22)$$

$$C_1= - x_{o1}^2\ \delta f/\delta x_1\big|_o \qquad (7.23)$$

$$C_2= - x_{o2}^2\ \delta f/\delta x_2\big|_o \qquad (7.24)$$

se tiene

$$C= C_1/X_1 + C_2/X_2 \qquad (7.25)$$

Ecuación de una hipérbola equilátera cuyas asíntotas son:

$$X_1 = C_1/C \qquad (7.26)$$

$$X_2 = C_2/C \qquad (7.27)$$

En el caso particular en que C= 0, se convierte en una recta que pasa por el origen con pendiente $-C_2/C_1$.

En la figura (7.13) pueden verse las cuatro formas que puede adoptar la curva, en función de los signos de los coeficientes C, C_1, C_2:

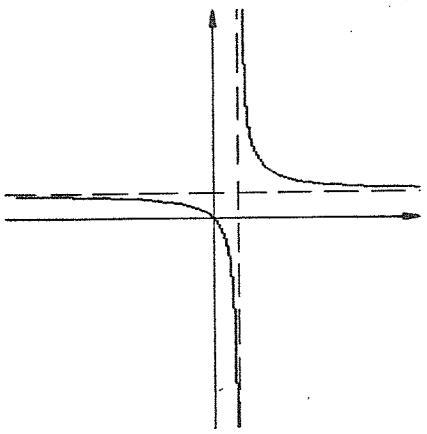

Figura 7.13.a
$C,C_1,C_2 > 0$
$C,C_1,C_2 < 0$

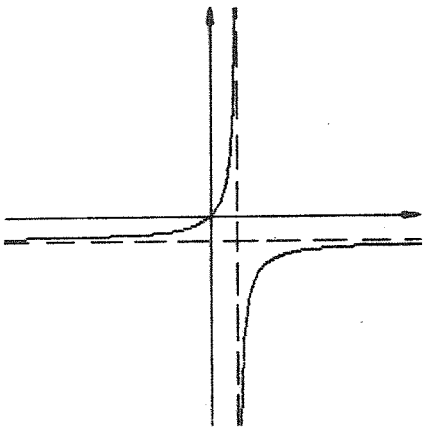

Figura 7.13.b
$C,C_1 > 0$, $C_2 < 0$
$C,C_1 < 0$, $C_2 > 0$

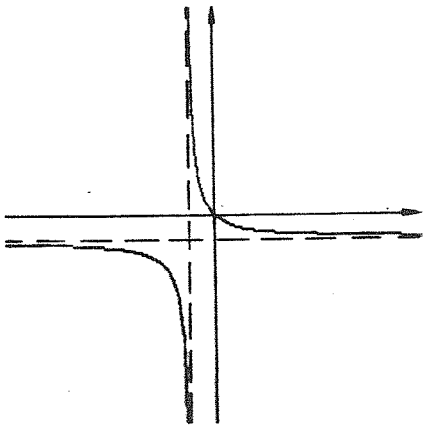

Figura 7.13.c
$C_1,C_2 > 0$, $C < 0$
$C_1,C_2 < 0$, $C < 0$

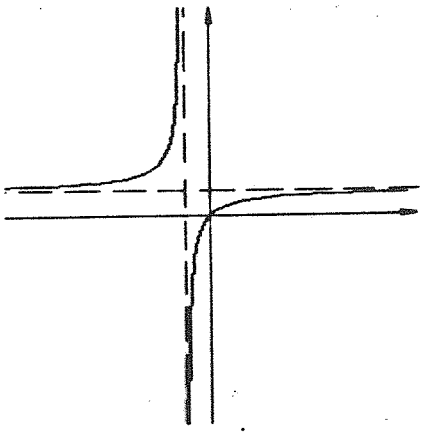

Figura 7.13.d
$C,C_2 > 0$, $C_1 < 0$
$C,C_2 < 0$, $C_1 > 0$

Figura 7.13. Formas de la aproximación inversa en función de los coeficientes C,C_1,C_2

En el caso de aproximación híbrida, la expresión (7.13) queda en su forma mas general

$$A= \delta f/\delta x_1|_o\ (x_1-x_{o1})\ B_1 + \delta f/\delta x_2|_o\ (x_2-x_{o2})\ B_2 \qquad (7.28)$$

En función de los valores de B_1 y B_2, se pueden presentar cuatro casos:

- B1= 1 , B2= 1
  Coincide con la aproximación directa.

- B1= x_{o1}/x_1 , B2= x_{o2}/x_2
  Coincide con la aproximación inversa.

- B1= 1 , B2= x_{o2}/x_2
  En este caso se puede expresar

$$x_1 = K_1 + K_2/x_2 \tag{7.29}$$

siendo

$$K_1 = x_{o1} + A \,/\, (\delta f/\delta x_1|_o) - (x_{o2}\ \delta f/\delta x_2|_o) \,/\, (\delta f/\delta x_1|_o) \tag{7.30}$$

$$K_2 = x_{o2}^2\ (\delta f/\delta x_2|_o) \,/\, (\delta f/\delta x_1|_o) \tag{7.31}$$

Hipérbola equilátera con asíntotas en

$$x_1 = K_1 \tag{7.32}$$

$$x_2 = 0 \tag{7.33}$$

- B1= x_{o1}/x_1 , B2= 1
  En este caso se puede expresar

$$x_1 = K_1 + K_2/x_2 \tag{7.34}$$

siendo

$$K_1 = x_{o2} + A \,/\, (\delta f/\delta x_2|_o) - (x_{o1}\ \delta f/\delta x_1|_o) \,/\, (\delta f/\delta x_2|_o) \tag{7.35}$$

$$K_2 = x_{o1}^2\ (\delta f/\delta x_1|_o) \,/\, (\delta f/\delta x_2|_o) \tag{7.36}$$

Hipérbola equilátera con asíntotas en

$$x_1 = 0 \tag{7.37}$$

$$x_2 = K_1 \tag{7.38}$$

Para calcular estas curvas se puede realizar un barrido según una de las dos variables (se dan valores a una de las dos variables, según un incremento determinado, obteniendo los correspondientes valores de la otra variable que hacen que se cumpla la ecuación). Se obtiene un conjunto de puntos que, conectados ordenadamente, nos dan la representación de la función.

El algoritmo puede ser:

- decidir sobre que variable se hace el barrido.
  Para ello se calcula la pendiente de la tangente a la curva en el punto conocido (punto de diseño) y se toma como eje de barrido aquel que sea mas paralelo a dicha tangente.

- Hacer un primer medio barrido, decrementanto la variable elegida hasta salirse de la ventana dada por:

$$\begin{aligned} x_{1min} \leq x_1 \leq x_{1max} \\ x_{2min} \leq x_2 \leq x_{2max} \end{aligned} \qquad (7.39)$$

- Hacer un segundo medio barrido, incrementanto la variable elegida hasta salirse de la ventana.

El tomar como punto de partida el punto de diseño, y tomar como variable sobre la que se hace el barrido la mas paralela a la tangente, nos asegura que el primer punto calculado (punto central) pertenece a la rama correcta de la hipérbola (fig. 7.14).

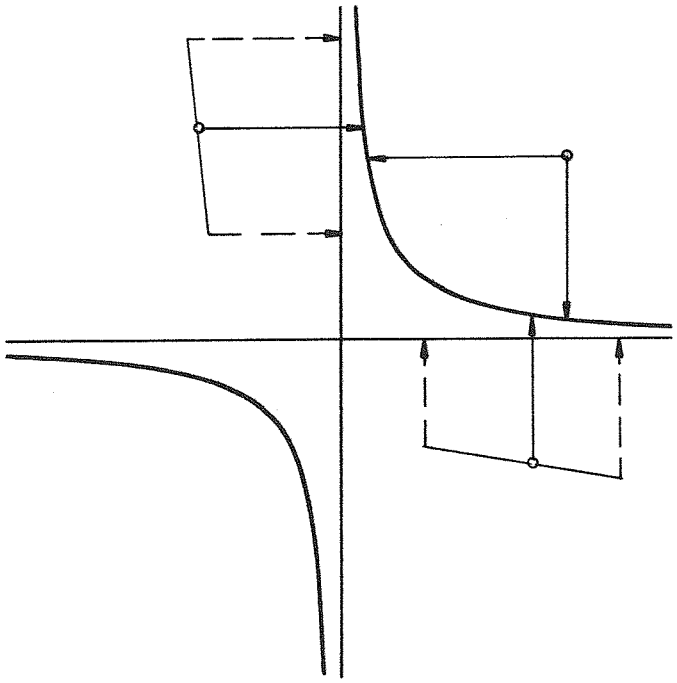

Figura 7.14 Selección de la variable sobre la que se hace el barrido.

Para comprobar que el segundo punto calculado pertenece a la misma rama que el primero, se debe calcular un punto intermedio y comparar el signo de los ángulos que forman los segmentos que unen dichos

puntos con el punto central respecto a la recta tangente son del mismo signo (fig. 7.15). El método se generaliza para los siguientes puntos: para comprobar que siguen perteneciendo a la misma rama se compara el ángulo del último segmento respecto al penúltimo con el anterior. Es decir, del mismo signo que el del primer segmento respecto a la tangente.

Un cambio de signo significa que el último punto calculado pertenece a la otra rama. Para evitarlo, se repite el cálculo del último punto con un incremento/decremento de la variable mas pequeño, hasta obtener un punto auxiliar de la misma rama.

Dado que las curvas tienen dos tramos claramente diferenciados (cada uno sensiblemente paralelo a uno de los ejes), conviene que el barrido se haga sobre la variable mas sensible, en cada uno de los tramos.

Para ello hay que preveer un cambio de la variable sobre la que se hace el barrido. Puesto que las curvas son suaves y sin puntos de inflexión, el criterio puede ser simplemente que el incremento de la variable de barrido pase a ser menor que el incremento de la variable calculada, entre dos puntos consecutivos.

Con este cambio de barrido, al llegar a la zona donde se puede saltar a la otra rama, el punto auxiliar provocará el cambio de barrido, por lo que no habrá que seguir reduciendo el paso de barrido para los siguientes puntos.

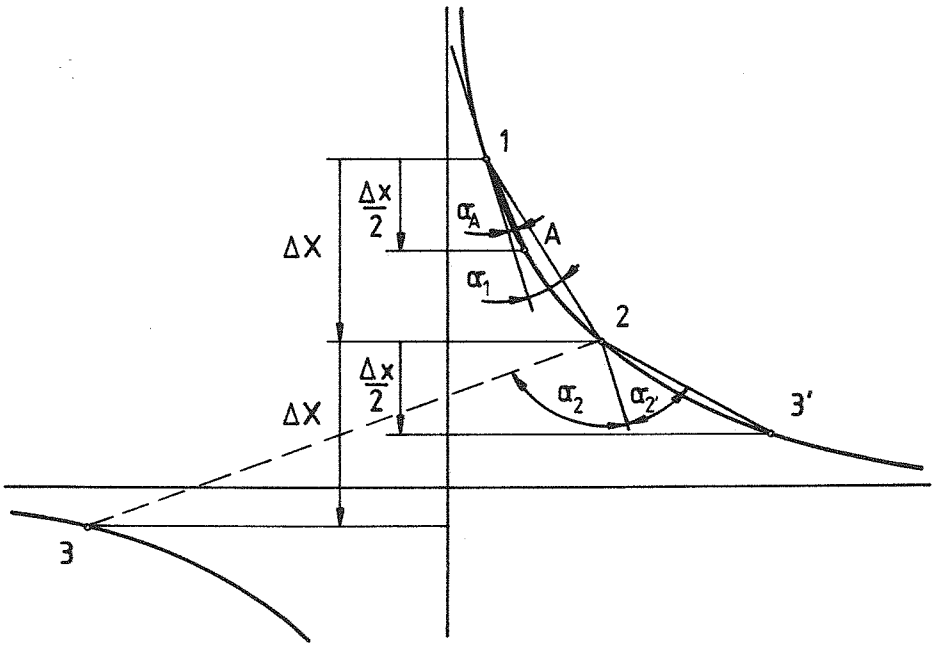

Figura 7.15. Seguimiento de una rama de la curva.

CAPITULO 8

EJEMPLOS DE APLICACION DEL SISTEMA DISSENY

8.1 INTRODUCCION

En este capítulo se van a estudiar varios ejemplos representativos de diferentes tipos de problemas. Los ejemplos se ha resuelto utilizando el sistema de diseño óptimo de estructuras DISSENY, descrito en los capítulos precedentes, y se han seleccionado con el doble criterio de que permitan evaluar las posibilidades del sistema e ilustren las capacidades gráficas del mismo.

El primer ejemplo que se va a considerar es puramente académico, y se incluye con la finalidad principal de comprobar la bondad de la implementación realizada, puesto que es un ejemplo típico en optimización, y sus resultados (tanto numéricos como gráficos) pueden encontrarse en la mayor parte de la bibliografia sobre el tema.

Los otros dos ejemplos son representativos de dos tipos diferentes de problemas con características muy diferentes. El primero de ellos es un caso de optimización de elemento estructuras complejo. En el segundo se considera una estructura de barras de nudos articulados, con todas las características que acompañan a una estructura de elevado número de elementos.

Los ejemplos muestran que el sistema es capaz de optimizar estructuras muy variadas de forma eficiente, y con un apoyo gráfico muy completo.

8.2 ESTRUCTURA DE TRES BARRAS.

8.2.1 Descripción del problema. Criterios de diseño.

Como primer ejemplo, vamos a considerar un problema clásico en la bibliografía sobre optimización. Se trata de la estructura de tres barras de nudos articulados que puede verse en la figura (8.1).

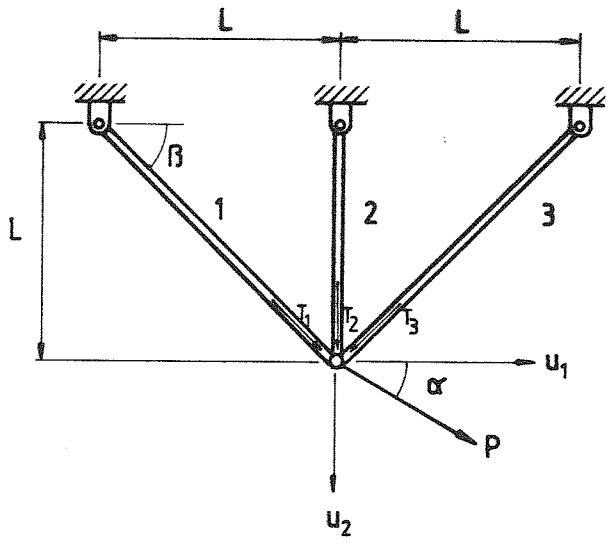

Figura 8.1. Estructura de tres barras

El propósito del diseño es el de encontrar las áreas de las secciones transversales de las tres barras (A_1,A_2,A_3) que hagan la estructura lo más ligera posible; cumpliendo ciertas restricciones.

Los datos que se van a considerar son:

Altura (L)	10.0	in.
Modulo de Elasticidad (E)	1.0E+7	psi
Peso Específico (γ)	0.1	lb/in^3

El modelo se va a diseñar sometido a 3 estados de carga:

Estado 1	P= 40000. lb	α= 45°
Estado 2	P= 30000. lb	α= 90°
Estado 3	P= 20000. lb	α= 135°

Se han realizado dos diseños, en el primero se han impuesto únicamente restricciones de tensión en las tres barras:

Tensión máxima en barra 1 (σ_{1m}) 5000.0 psi

Tensión máxima en barra 2 (σ_{2m}) 20000.0 psi

Tensión máxima en barra 3 (σ_{3m}) 5000.0 psi

En el segundo se han mantenido las restricciones de tensión, y se han añadido restricciones de desplazamiento para el nudo común:

Desp. máximo según eje horizontal (u_{1m}) 0.005 in

Desp. máximo según eje vertical (u_{2m}) 0.004 in

Por último, se ha impuesto una restricción de igualdad:

$$A_1 = A_3$$

para obtener un diseño simétrico.

8.2.2 Formulación del problema.

Puesto que se pretende obtener la estructura más ligera posible, se ha tomado como función objetivo el peso de la estructura. Así, la expresión de dicha función queda como:

$$F = \gamma \, (L \sqrt{2} \, A_1 + L \, A_2 + L \sqrt{2} \, A_3) \qquad (8.1)$$

Todas las restricciones se han normalizado a valor 1. Así, las restricciones de tensión se han expresado de la forma:

$$g_i(A) = 1 - \sigma_i/\sigma_{im} \qquad (8.2)$$

Análogamente, las restricciones de desplazamiento se han expresado de la forma:

$$g_j(a) = 1 - u_j/u_{jm} \qquad (8.3)$$

Las expresiones de σ_i y u_j, en función de los parámetros del problema y de las variables de diseño, se han obtenido planteando las ecuaciones de equilibrio para el nudo 1:

$$P = B^T \, T \qquad (8.4)$$

que en este caso se convierte en:

$$\begin{bmatrix} P\cos\alpha \\ P\,\mathrm{sen}\alpha \end{bmatrix} = \begin{bmatrix} \cos\beta & 0 & -\cos\beta \\ \mathrm{sen}\beta & 1 & \mathrm{sen}\beta \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix} \tag{8.5}$$

en donde T es el vector de fuerzas internas sobre las barras. Estas fuerzas internas, las podemos expresar en función de los desplazamientos de la forma:

$$T = D\ B\ u \tag{8.6}$$

que es este caso queda como:

$$\begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix} = \begin{bmatrix} A_1E/l_1 & 0 & 0 \\ 0 & A_2E/l_2 & 0 \\ 0 & 0 & A_3E/l_3 \end{bmatrix} \begin{bmatrix} \cos\beta & \mathrm{sen}\beta \\ 0 & 1 \\ -\cos\beta & \mathrm{sen}\beta \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{8.7}$$

De las expresiones (8.4) y (8.6) obtenemos finalmente:

$$P = B^T\ D\ B\ u = K\ u \tag{8.8}$$

Por lo tanto, la matriz de rigidez queda de la forma:

$$K = \sqrt{2}E/4L \begin{bmatrix} (A_1+A_3) & (A_1-A_3) \\ (A_1-A_3) & (A_1+2\sqrt{2A_2}+A_3) \end{bmatrix} \tag{8.9}$$

De donde podemos obtener las expresiones de los desplazamientos:

$$u_1 = \frac{\sqrt{2}L}{E} \frac{-s_2\,(A_1-A_3) + s_1\,(A_1+2\sqrt{2A_2}+A_3)}{\sqrt{2}A_1A_2 + 2A_1A_3 + \sqrt{2}A_2A_3} \tag{8.10}$$

$$u_2 = \frac{\sqrt{2}L}{E} \frac{s_2\,(A_1+A_3) - s_1\,(A_1-A_3)}{\sqrt{2}A_1A_2 + 2A_1A_3 + \sqrt{2}A_2A_3} \tag{8.11}$$

siendo

α_1 la componente horizontal de la carga (P cosα)
α_2 la componente vertical de la carga (P senα)

A partir de los desplazamientos, podemos obtener las expresiones de las tensiones:

$$\sigma_1 = E(u_1+u_2)/2L \qquad (8.12)$$

$$\sigma_2 = E\ u_2/L \qquad (8.13)$$

$$\sigma_3 = E(u_2-u_1)/2L \qquad (8.14)$$

Estas expresiones se han implementado directamente, eliminando la necesidad de acudir al procesador de análisis para evaluar los desplazamientos y tensiones.

8.2.3 Resultados del estudio.

8.2.3.1 Evolución.

La evolución de la función objetivo puede verse en la figura (8.2). En ella se observa como el diseño inicial es más ligero que el final. Esto es debido a que el diseño inicial no cumple las condiciones impuestas. La aproximación a la solución optima es muy rápida; observandose que ya en la tercera iteración la solución es muy buena. También se aprecia como el diseño es más ligero con restricciones de tensión (línea de puntos), que cuando se incluyen las restricciones de desplazamiento (línea continua). Es decir, que estas últimas son más "restrictivas".

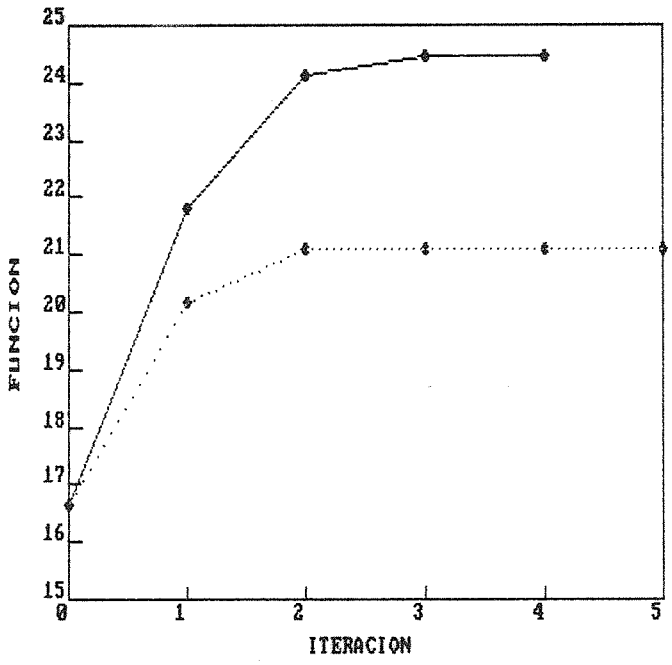

Figura 8.2. Evolución de la función objetivo.

En la figura (8.3) puede verse la evolución de las cinco restricciones para los dos primeros estados de carga. Se han representado únicamente las restricciones para el segundo caso, puesto que para el primero serían las mismas restricciones de tensión (basta con no considerar las de desplazamiento).

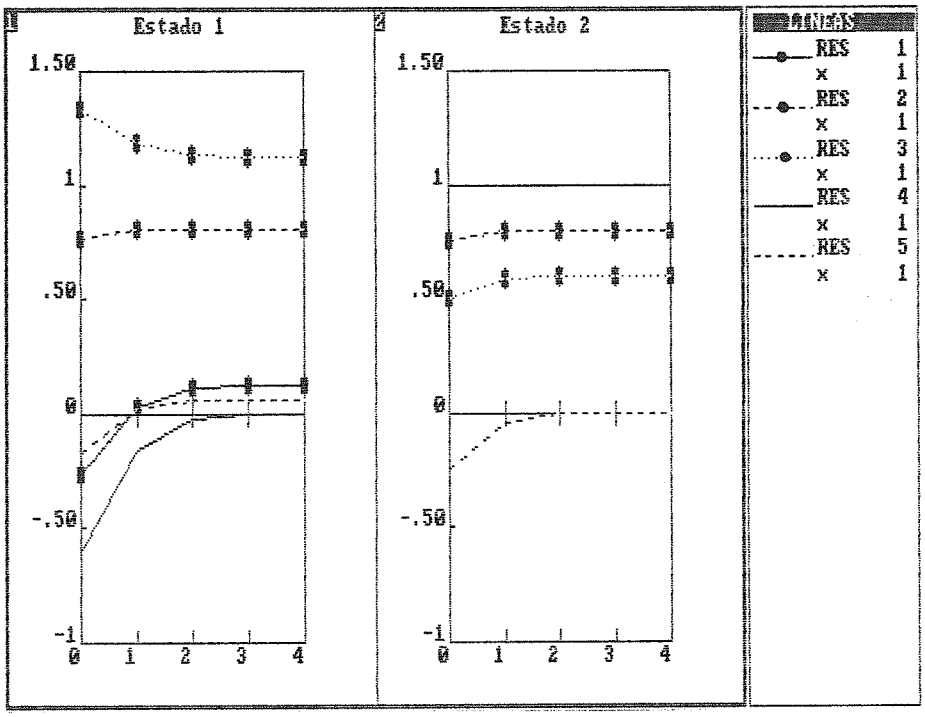

Figura 8.3. Evolución de las restricciones.

No se han representado las restricciones del tercer estado porque es semejante al 1 (dado que las cargas están situadas simétricamente), pero con restricciones menos críticas (dado que la carga es menor). La simetría se observa también el la coincidencia de las restricciones de tensión de las barras 1 y 3 en el estado 2 (carga vertical).

Considerando únicamente las restricciones de tensión (líneas con puntos gruesos), se aprecia que la única restricción crítica es la tensión de la barra 1 para el primer estado de carga (fig. 8.3, línea continua con puntos gruesos). Mientras que considerando todas las restricciones se vé como las de desplazamiento son más críticas que las de tensión. Entre las restricciones de desplazamiento, son críticas la de desplazamiento horizontal para el estado 1 (fig. 8.3, línea continua); y la de desplazamiento vertical para el estado 2 (fig. 8.3, línea de trazos).

Por último, en la figura (8.4), puede verse la relación entre las variables y la función objetivo. En la parte derecha de la gráfica aparece la relación de la función objetivo con la variable 1; tanto en el caso de restricciones de tensión (línea de puntos), como en el de las de desplazamiento (línea continua). Se observa que la evolución de la función objetivo es practicamente lineal con la de la variable en los dos casos. También puede verse la relación de la variable 2 con la función objetivo (líneas de la izquierda), que es sensiblemente lineal en el caso de restricciones de tensión, pero no lo es para las restricciones de desplazamiento.

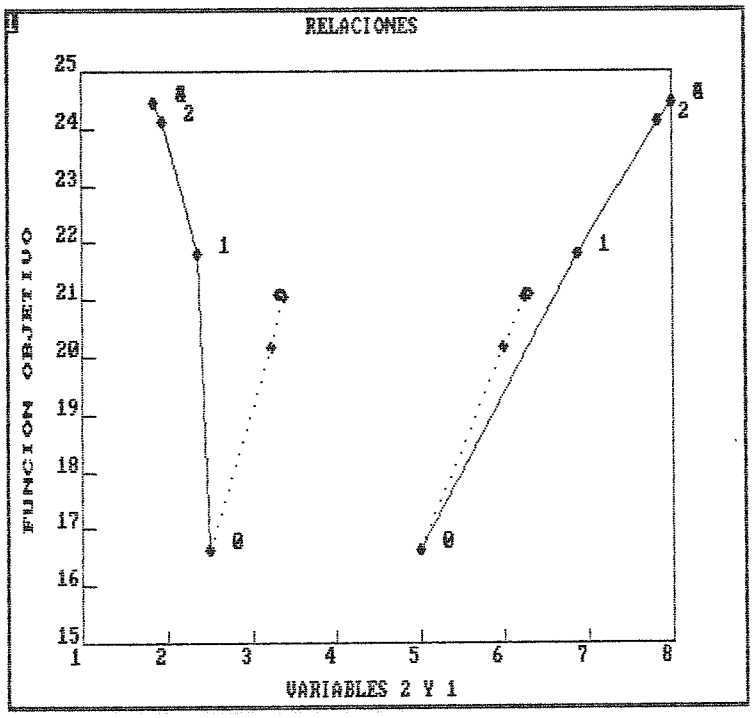

Figura 8.4. Relación entre las variables y la función objetivo.

8.2.3.2 Espacio de diseño

Por la condición de simetría, el problema tiene, realmente, dos variables (A_1 y A_2), respecto a las que se dan los resultados del estudio. Tanto la función objetivo del problema (8.1), como las restricciones activas (8.10 a 8.14), han sido calculadas independientemente para obtener sus curvas "reales". Estas curvas se han obtenido dando valores a una de las variables, y calculando el valor que debe tomar la otra variable para que la función tome un valor buscado (por barrido). En la figura (8.5) se representa la función objetivo y la restricción de tensión para la barra 1 en el primer estado de carga (que es la activa). La aproximación directa de

la función objetivo, e inversa de la restrición de tensión, dada por el programa se muestra en la figura (8.6). En ella puede observarse que la función objetivo coincide exáctamente con la real, y la restricción resulta más conservadora que la real.

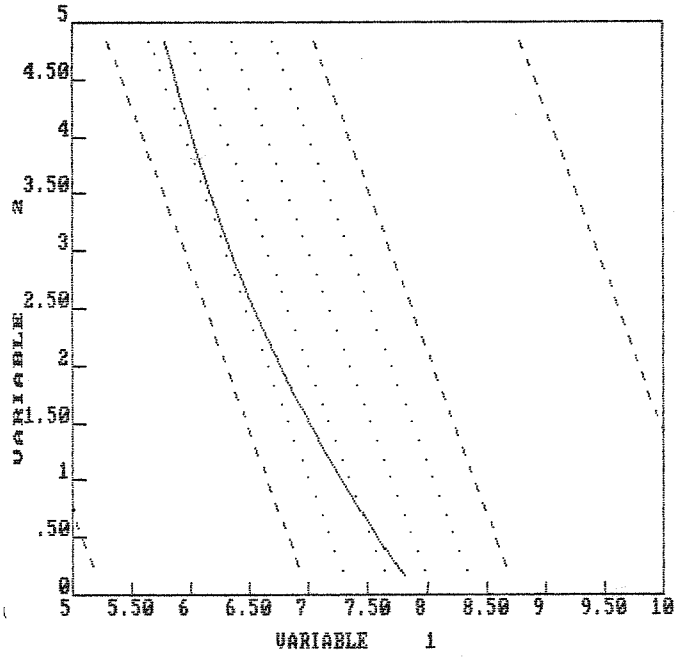

Figura 8.5. Espacio de diseño real,con restricción de tensión.

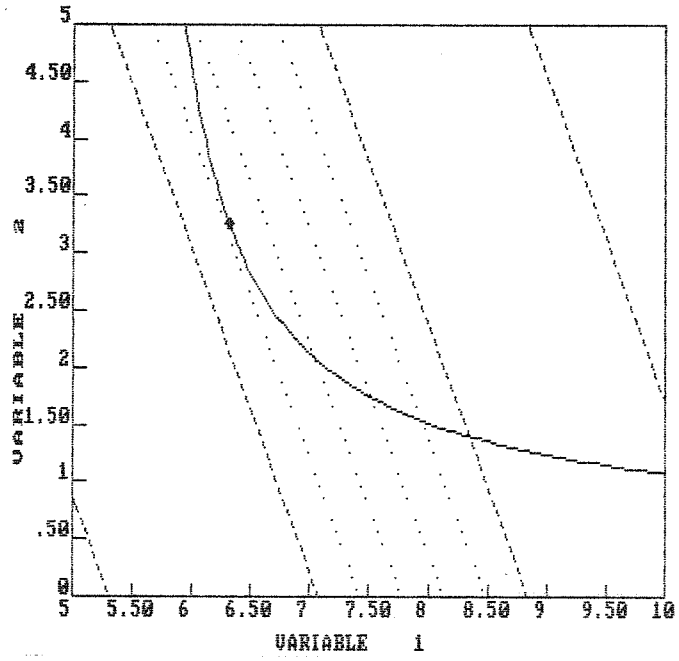

Figura 8.6. Espacio de diseño calculado, con restricción de tensión por aproximación inversa.

En la figura (8.7) se representa la función objetivo y las restricciones de desplazamiento; según el eje horizontal, para el estado 1 (línea vertical), y según el eje vertical para el estado 2. La estimación dada por el programa puede verse en la figura (8.8).

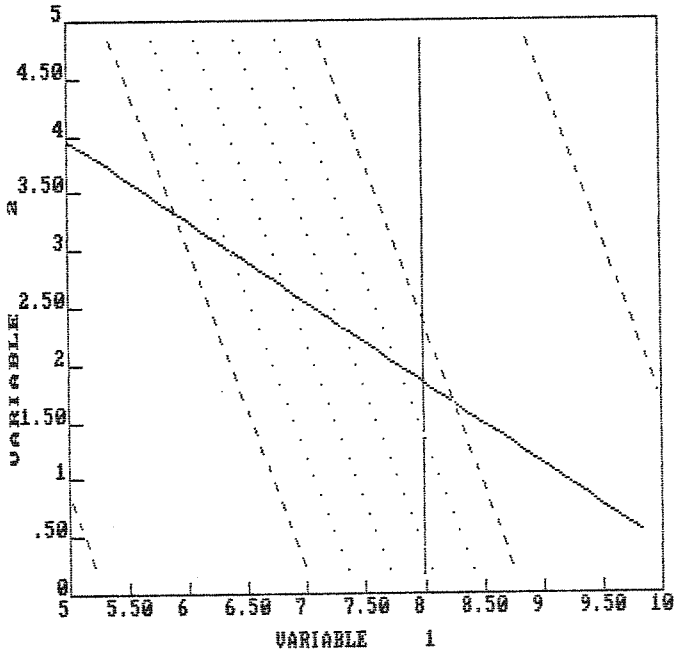

Figura 8.7. Espacio de diseño real, con restricciones de desplazamiento.

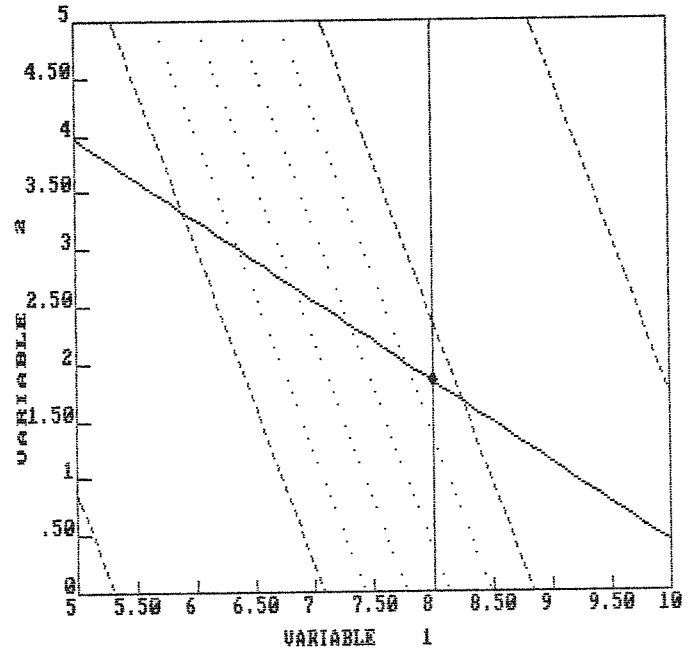

Figura 8.8. Espacio de diseño calculado, con restricciones de desplazamiento por aproximación directa.

Debe destacarse que derivando la expresión (8.10) se obtiene:

$$\frac{\delta u_1}{\delta A_2} = \frac{2L}{E} \frac{-s_1 (A_1-A_3)^2 + s_2 (A_1^2-A_3^2)}{(\sqrt{2}A_1A_2 + 2A_1A_3 + \sqrt{2}A_2A_3)^2} \qquad (8.15)$$

y, dada la condición de simetría impuesta, se cumple que:

$$\frac{\delta u_1}{\delta A_2} = 0 \qquad (8.16)$$

Es decir, que las restricciones de desplazamiento horizontal son independientes de A_2.

Sustituyendo en la expresión (8.11) la igualdad $A_1 = A_3$, se tiene:

$$u_2 = \frac{\sqrt{2}Ls_2}{E} \frac{1}{A_1 + \sqrt{2}A_2} \qquad (8.17)$$

Es decir, que las restricciones de desplazamiento vertical son lineales respecto a las variables A_1 y A_2.

Las figuras anteriores corresponden todas ellas a la iteración final (iteración 4). Si dibujamos el subespacio de diseño con la información obtenida en la primera iteración (iteración 0), obtenemos los resultados mostrados en las figuras (8.9) y (8.10). En dichas figuras se ha incluido la evolución del punto de diseño (puntos gruesos unidos por líneas de puntos), comenzando por el punto inicial: A_1=5, A_2=2.5.

En las dos figuras puede observarse como el punto de diseño calculado por el algoritmo de optimización coincide con la aproximación directa de las mismas. De forma que el punto de diseño obtenido en la iteración 1 es el que se podría haber estimado observando estas representaciones.

Hay que notar que en el caso de la restricción de tensión (fig. 8.9) la aproximación directa nos permite conocer el valor de la función objetivo en el punto de diseño; pero falta alguna condición complementaria para elegir dicho punto, dado que la restricción es paralela a las lineas de valor constante de la función objetivo. En la misma figura (8.9) se ha superpuesto la aproximación inversa. Esta aproximación si que nos permite elegir un punto de diseño, que además en este caso resulta mucho más cercano al óptimo que el dado por la estimación directa. La aproximación inversa de las restricciones de

desplazamiento se muestra en la figura (8.10). Como en el caso anterior, resulta una aproximación más conservadora, que, en el caso de la restricción 4 (desplazamiento horizontal para el estado 1) vuelve a coincidir con el óptimo.

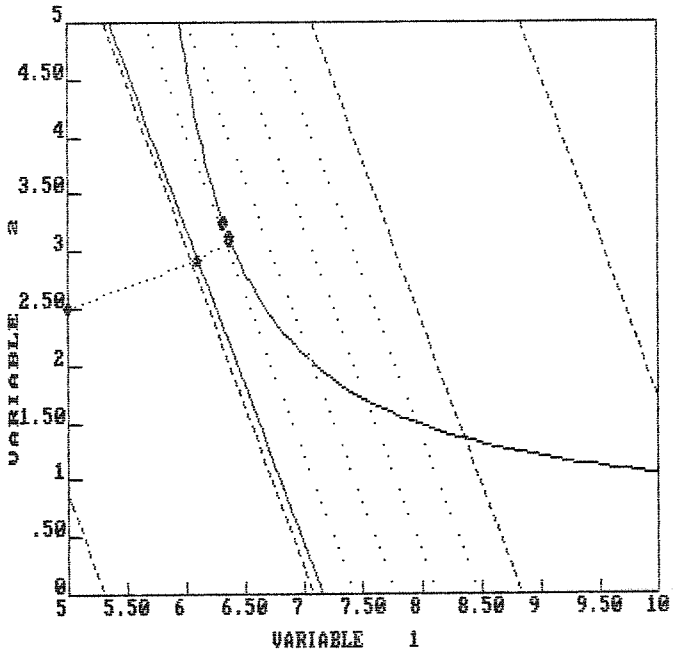

Figura 8.9. Espacio de diseño calculado, (para iteración inicial) con restricción de tensión por aproximación directa e inversa.

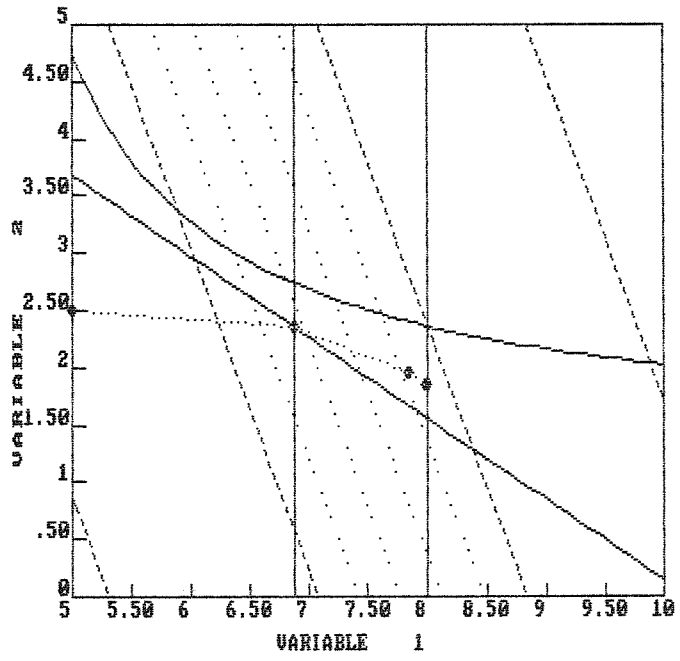

Figura 8.10. Espacio de diseño calculado, (para iteración inicial) con restricciones de desplazamiento por aproximación directa e inversa.

8.3 VIGA ARMADA.

8.3.1 Descripción del problema. Criterios de diseño.

Como segundo ejemplo, vamos a considerar un problema de optimización de un elemento estructural complejo. Se trata de diseñar una viga metálica armada. El problema está tratado en profundidad en la referencia /46/, por lo que vamos a limitarnos a resumir el planteamiento antes de describir el proceso de diseño. Además, el problema se ha resuelto siguiendo básicamente la normativa vigente (Norma MV-103-1972 sobre Cálculo de las Estructuras de Acero Laminado en Edificación), por lo que también se puede consultar dicha normativa en lo referente a criterios de diseño. La viga que se pretende optimizar está esquematizada en la figura (8.11).

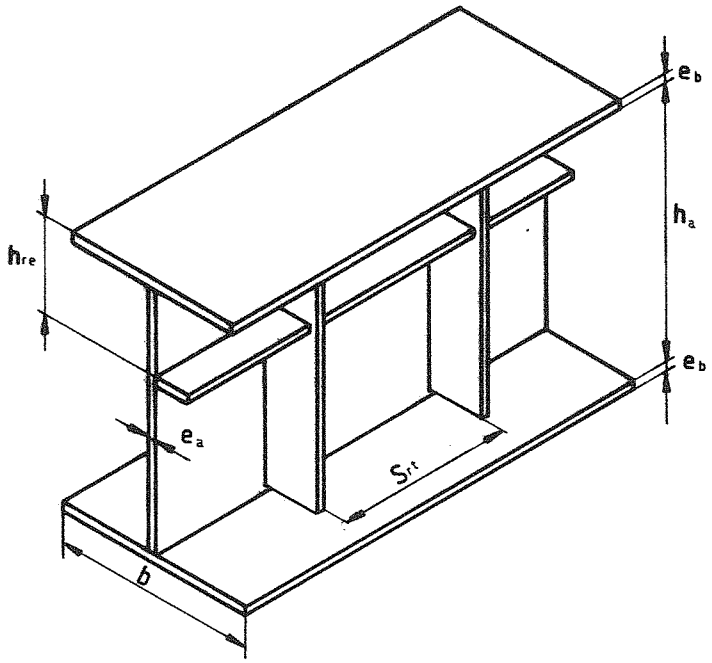

Figura 8.11. Viga armada.

Según esta figura, las variables de diseño consideradas son:

1- altura del alma (h_a)

2- espesor del alma (e_a)

3- ancho de las alas (b)

4- espesor de las alas (e_b)

5- separación de los rigidizadores transversales (s_{rt})

6- altura de colocación del rigidizador longitudinal (h_{r1})

El propósito del diseño es el de definir los valores de todas estas variables que hagan la estructura lo más barata posible, teniendo en cuenta los costos de cada componente y los de montaje, y cumpliendo ciertas restricciones impuestas por la normativa.

En la figura (8.12) pueden verse las condiciones de contorno y las acciones que se van a considerar. Los datos son:

Luz (L)	30.0 m.
Peso específico del material (γ)	7820.0 Kg/m^3
Modulo de Elasticidad (E)	2100000.0 Kp/cm^2

El modelo se va a diseñar sometido a un estado de carga, definido por:

Carga uniforme (q)	1500.0 Kp/m.
Cargas puntuales (P)	15000.0 Kp
Posición de las cargas puntuales	
desde el extremo inicial (a)	10.0 m
desde el extremo final (b)	10.0 m
Coeficiente de mayoración de carga uniforme	1.33
Coeficiente de mayoración de cargas puntuales	1.50

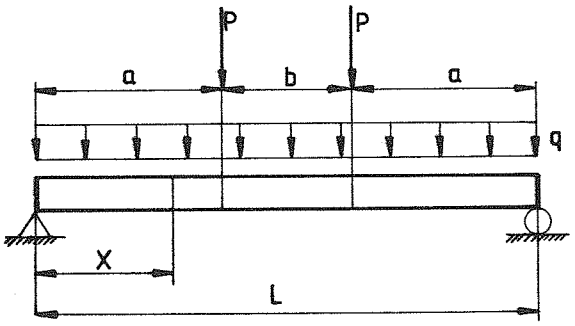

Figura 8.12. Cargas y restricciones de la viga.

Otros parámetros fijos del problema son:

Distancia entre arriostramientos laterales (s_a)	5.0 m.
Tensión máxima admisible (σ_u)	2600.0 Kp/cm^2

Las condiciones que impone la normativa se han tratado como restricciones de:

1- resistencia.

2- desplazamiento.

3- pandeo lateral.

4- abolladura del ala comprimida y del alma.

El resto de restricciones corresponden a limitaciones constructivas de dimensiones máximas y mínimas.

8.3.2 Formulación del problema.

Puesto que se pretende obtener la estructura más barata posible, se ha tomado como función objetivo el costo de la misma. Para definir este costo, se han utilizado unos coeficientes de ponderación que permiten evaluar el costo de los materiales a partir de su peso (costo unitario). Además se han añadido unos coeficientes que permiten incluir el costo de manipulación (soldadura, etc). Así, la expresión de dicha función queda como:

$$F = \gamma \, (C_{alma} \, L \, h_a \, e_a + C_{alas} \, 2 \, L \, b \, e_b + C_{rt} \, N_{rt} \, h_a \, A_{rt} + C_{rl} \, L \, A_{rl}) \qquad (8.18)$$

Siendo:

C_{alma} el costo de los materiales del alma (con preparación).

C_{alas} el costo de los materiales de las alas (con preparación).

C_{rt} el costo de los materiales de los rigidizadores transversales (con preparación).

C_{rl} el costo de los materiales del rigidizador longitudinal (con preparación).

C_{sa} el costo de soldadura entre el alma y las alas.

N_{rt} Número de rigidizadores transversales:

$$N_{rt} = L / S_{rt} + 1 \qquad (8.19)$$

A_{rt} Area de los rigidizadores transversales:

$$A_{rt} = (h_a / 47.78)^2 \qquad (8.20)$$

A_{rl} Area del rigidizador longitudinal:

$$A_{rl} = \frac{\sqrt{45\ h_a\ e_a^3\ (2.4\ (s_{rt}/h_a)^2 - 0.13)}}{15} \qquad (8.21)$$

Todas las restricciones se han normalizado a valor 1, habiendose considerado las siguientes:

1- Tensión normal:

$$g_1 = 1 - \sigma^* / \sigma_u \qquad (8.22)$$

siendo σ^* la tensión normal
(en la sección más desfavorable)

2- Tensión tangencial:

$$g_2 = 1 - \tau^* / \tau_{max} \qquad (8.23)$$

siendo τ^* la tensión tangencial
(en la sección más desfavorable)

3- Desplazamiento:

$$g_3 = 1 - f / f_{adm} \quad (fadm = L/1000) \qquad (8.24)$$

siendo f la flecha.

4- Pandeo lateral de la viga:

$$g_4 = b / b_{min} - 1 \quad (b_{min} = s_a \sqrt{12}\ /4Q) \qquad (8.25)$$

5- Pandeo local del ala comprimida:

$$g_5 = \frac{e_b\ 15\ \sqrt{2400/\sigma^*}}{b/2} - 1 \qquad (8.26)$$

Para calcular estas restricciones se utilizan los valores de momento flector (M) y esfuerzo cortante (Q), que se obtienen calculandolos en varias secciones de la viga y eligiendo la mas desfavorable.

Se han definido otras seis restricciones que consideran la abolladura del alma. Para ellas, se ha seguido la indicación de la norma de considerar independientes los distintos rectángulos comprendidos entre las dos alas, los rigidizadores transversales, y el rigidizador longitudinal.

Se calculan las tensiones de comparación (σ_{co}), y de comparación ideal (σ_{coi}), y se introducen las restricciones como:

$$g_i = (\sigma_{coi} - \sigma_{co}) / \sigma_u \qquad i= 6,...,11 \qquad (8.27)$$

Las restricciones 6,8 y 10 consideran los rectángulos inferiores (por debajo del rigidizador longitudinal), y las otras tres los superiores en diferentes puntos de la viga.

8.3.3 Resultados del estudio.

8.3.3.1 Evolución.

La evolución de los parámetros del diseño se muestra en las figuras (8.13) a (8.17). En la primera se vé la evolución de la función objetivo. Esta, parte de un valor factible alto que disminuye lentamente en las dos primeras iteraciones. En la tercera iteración hay un cambio brusco que lleva al diseño prácticamente a la solución óptima.

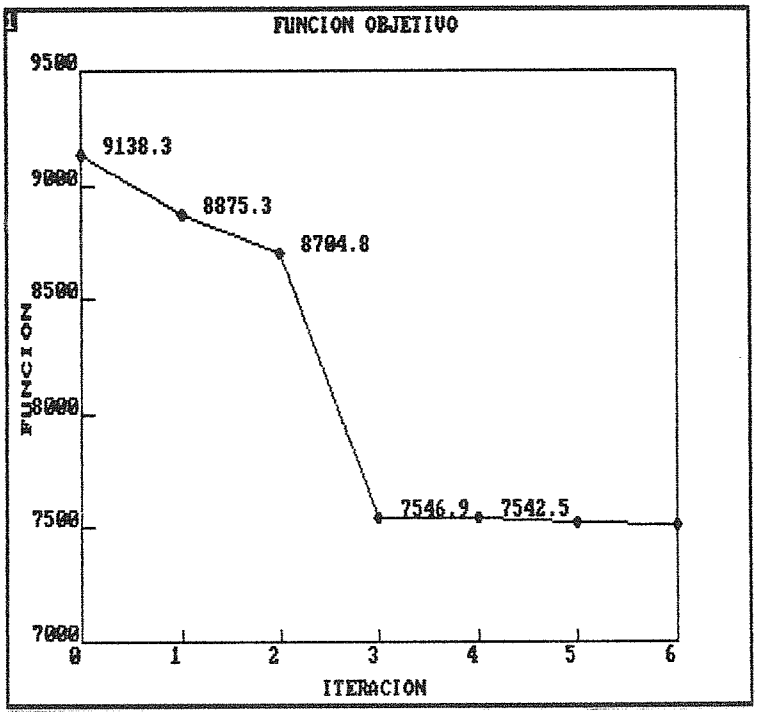

Figura 8.13. Evolución de la función objetivo.

La evolución de las variables viene dada en la figura (8.14). En ella se aprecia como las variables tampoco sufren apenas modificaciones en

las dos primeras iteraciones; estabilizandose en valores cercanos al óptimo a partir de la tercera iteración (salvo la variable 5).

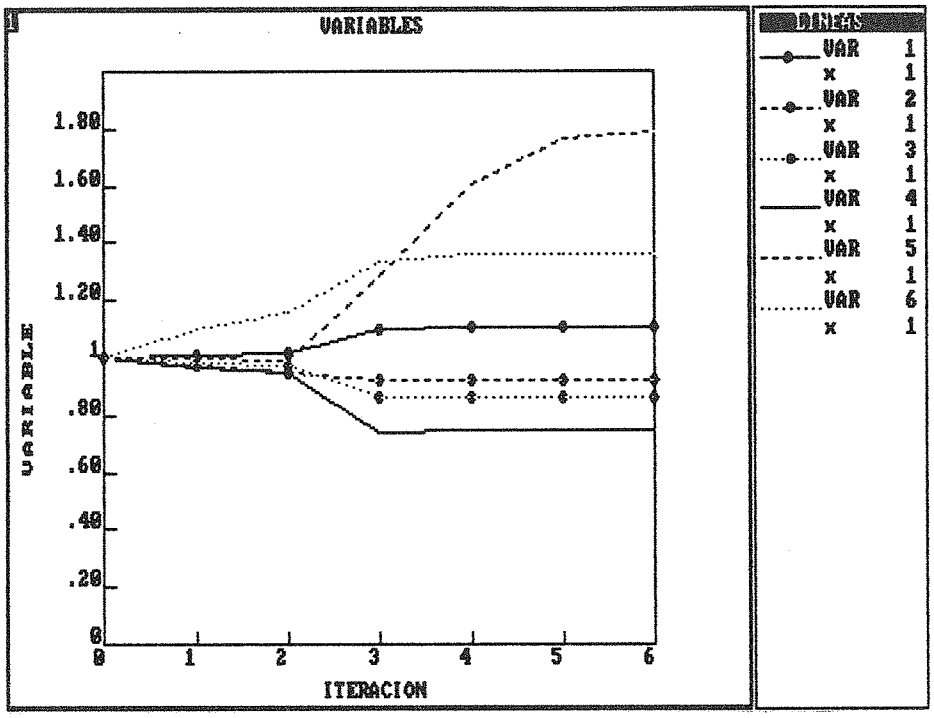

Figura 8.14. Evolución de las variables.

En las figuras (8.15) y (8.16) se han representado las evoluciones de todas las restricciones. En (8.15) se incluyen las cinco primeras restricciones. Y se observa que la restricción 3 es la más crítica de todas ellas. En la figura (8.16) se han representado las demás variables. Debe destacarse que la restricción 8 es equivalente a la 6, y la 9 a la 7, debido a que el ejemplo se ha resuelto para el caso particular de que el rigidizador longitudinal se extienda a lo largo de toda la viga. En este ejemplo se observa como la restricción 11 es la más critica de las que afectan al pandeo del alma.

Por último, en la figura (8.17) se incluyen, como ejemplo de evolución de las sensibilidades, las evoluciones de las derivadas de la restricción 4 (pandeo lateral de la viga) respecto a las seis variables. Se ha escogido esta restricción porque depende únicamente de la variable 3 (ancho de las alas), tal como se desprende de su formulación (8.23); dependencia que puede observarse en la figura.

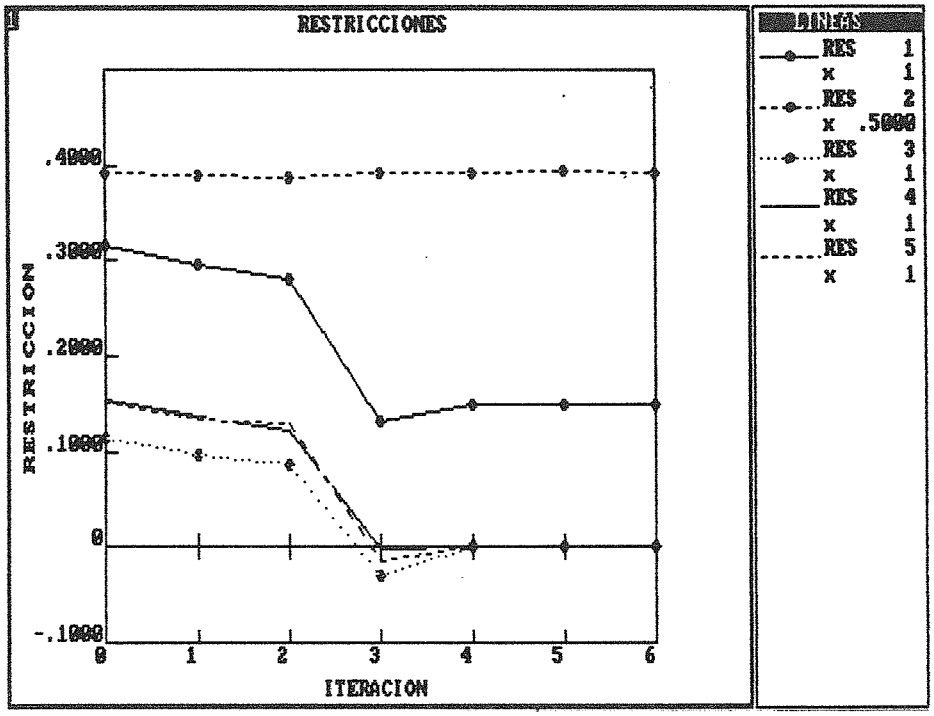

Figura 8.15. Evolución de las restricciones 1 a 5.

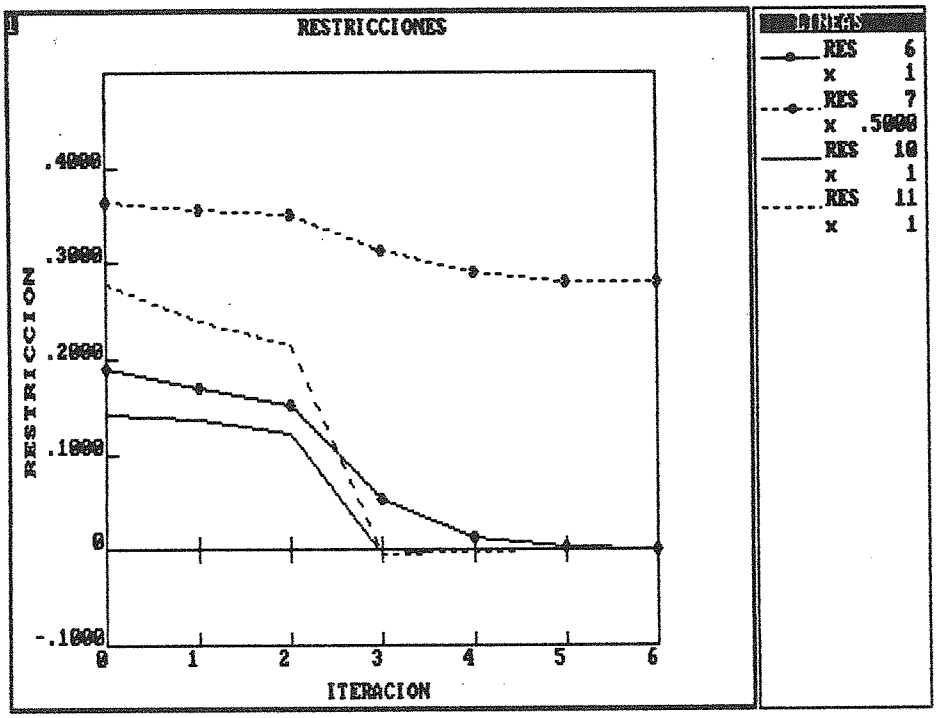

Figura 8.16. Evolución de las restricciones 6 a 11.

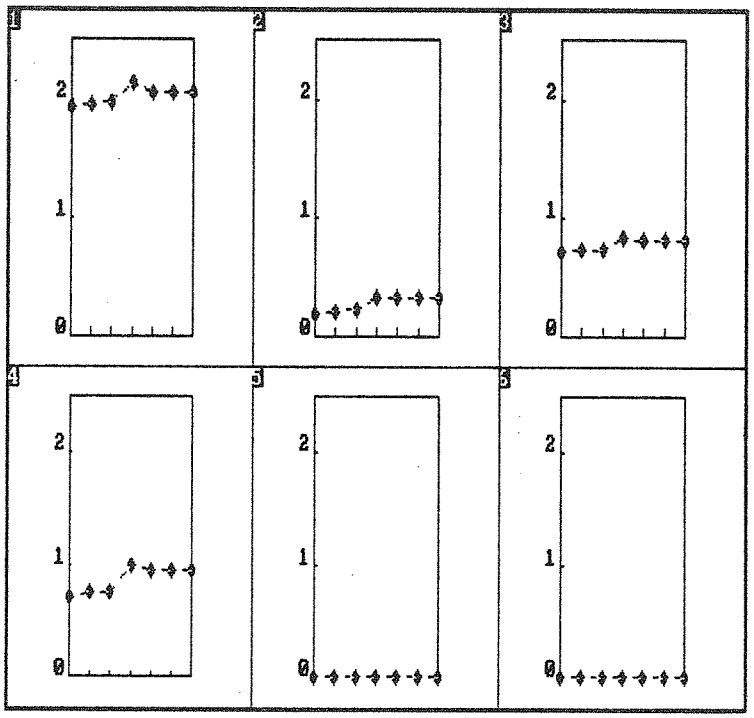

Figura 8.17. Evolución de las sensibilidades de la restricción 3.

8.3.3.2 Espacio de diseño

En este ejemplo el espacio de diseño tiene seis dimensiones, por lo que no es posible representarlo completo. Las representaciones se pueden obtener para subespacios formados combinando cualquier pareja de variables. La agrupación de variables elegida ha sido:

- subespacio de las variables 1 y 2 (dimensiones del alma);

- subespacio de las variables 3 y 4 (dimensiones de las alas), y

- subespacio de las variables 5 y 6 (situación de rigidizadores).

Análogamente al ejemplo anterior, se han calculado las curvas "reales" de la función objetivo y las restricciones, para poder contrastar los resultados dados por el programa. No obstante, estas curvas reales se han calculado únicamente para el primer subespacio.

En las figuras (8.18) y (8.19) se ha representado la función objetivo (para los valores de 5000, 10000 y 15000; y para valores intermedios de mil en mil) con los tres tipos de aproximación de que dispone actualmente el programa: lineal, inversa e híbrida (coincide en este caso con la inversa). Estas representaciones se pueden comparar con la función real representada en la figura (8.20). Debe notarse que el brusco cambio de curvatura que sufre la función real para valores de la variable 1 cercanos a cero, solo se ha calculado para valor 5000 por simplicidad. De la comparación de estas figuras se desprende que, si bién la función objetivo no es lineal , en la zona de diseño, esta resulta la mejor aproximación.

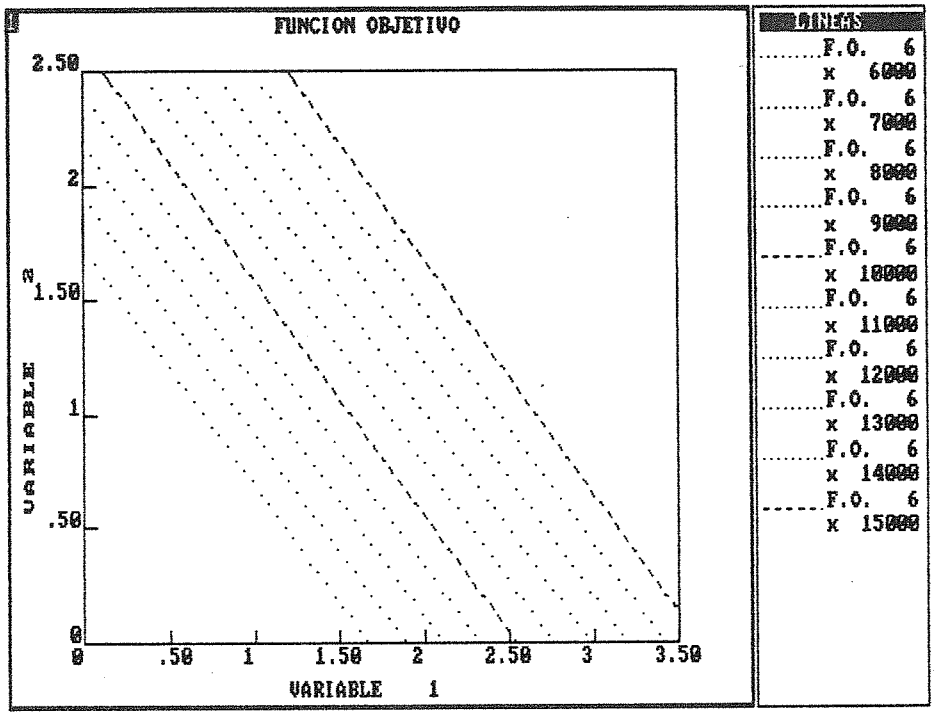

Figura 8.18. Función objetivo lineal
para el subespacio de las variables 1 y 2.

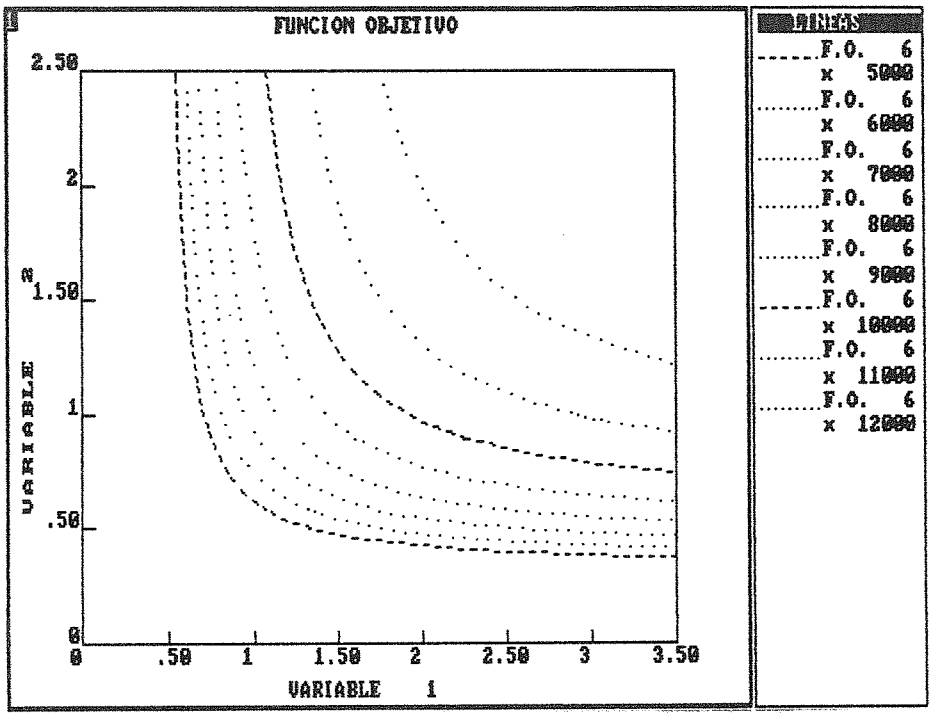

Figura 8.19. Función objetivo inversa y híbrida
para el subespacio de las variables 1 y 2.

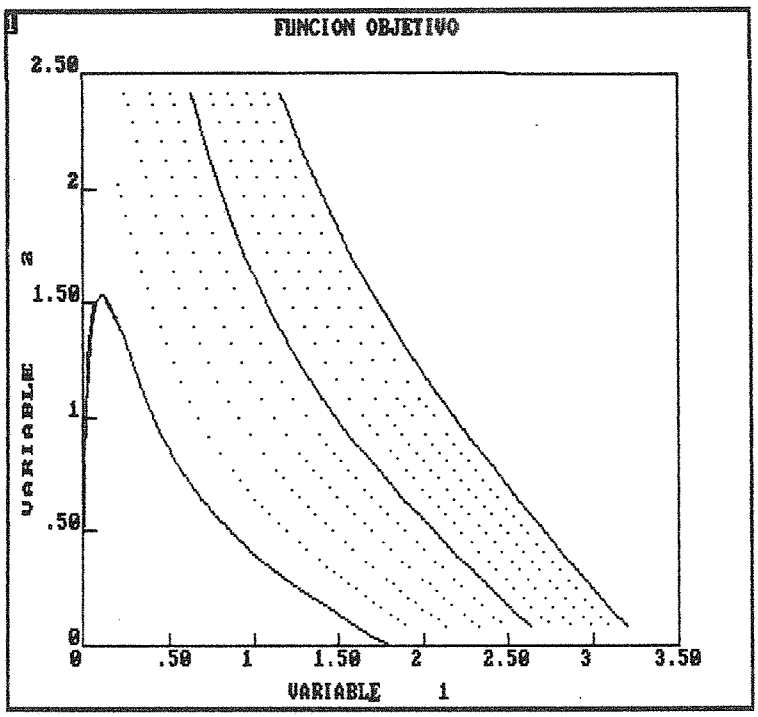

Figura 8.20. Función objetivo real
para el subespacio de las variables 1 y 2.

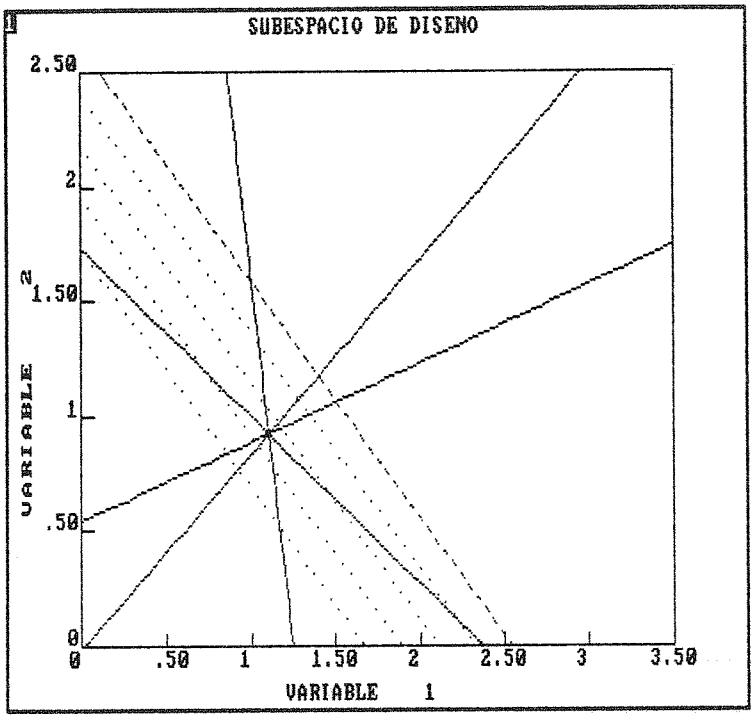

Figura 8.21. Restricciones activas y función objetivo con aproximación directa.

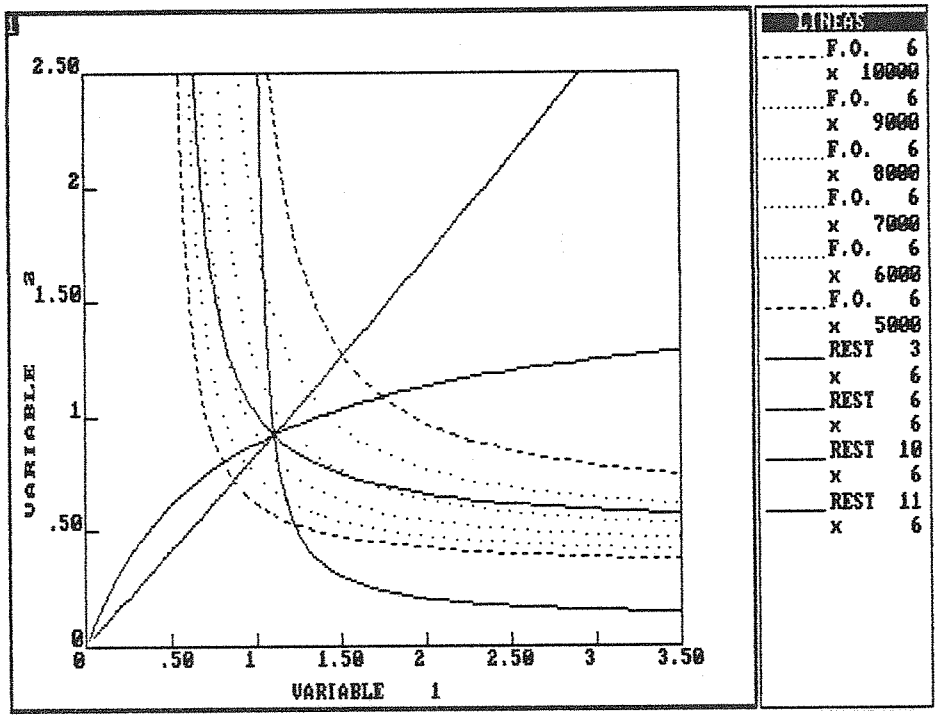

Figura 8.22. Restricciones activas y función objetivo con aproximación inversa.

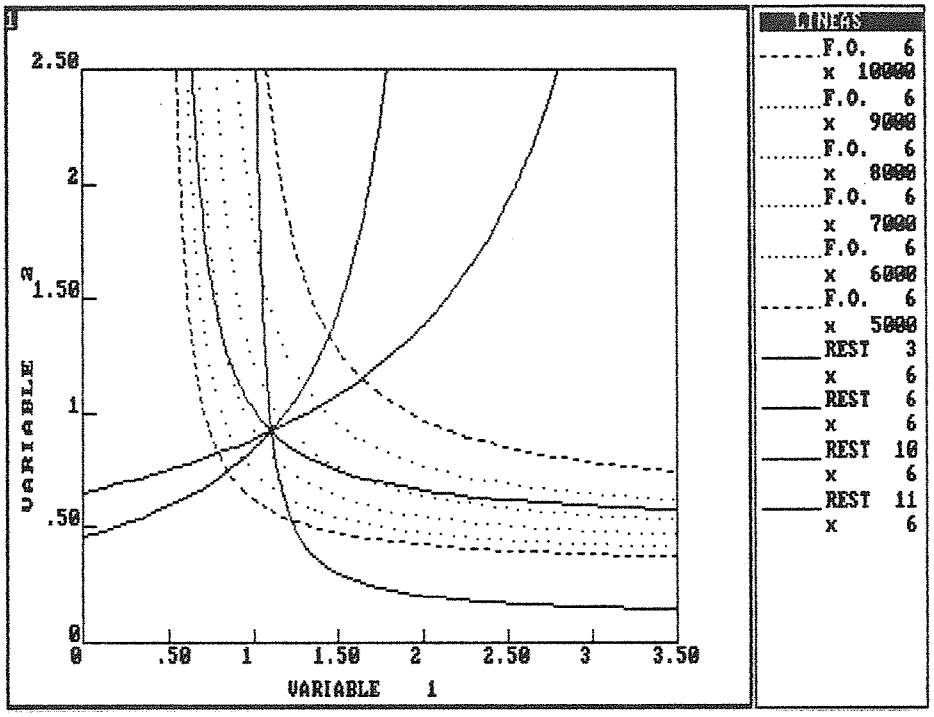

Figura 8.23. Restricciones activas y función objetivo con aproximación híbrida.

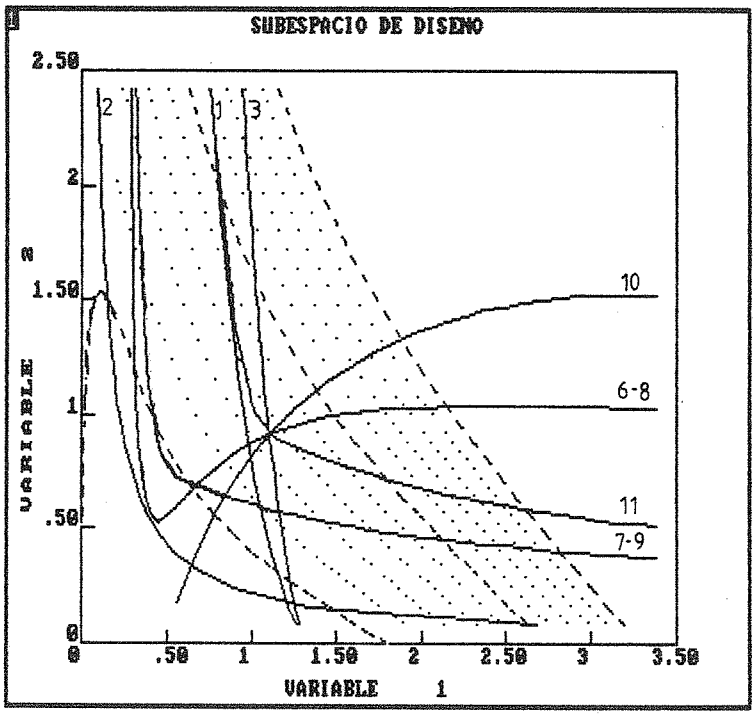

Figura 8.24. Espacio real de diseño.

En las figuras (8.21), (8.22) y (8.23) se han representado las restricciones activas; con aproximaciones: directa, inversa e híbrida, respectivamente. Las restricciones activas en el óptimo son (como puede verse en las figuras (8.15) y (8.16)) las 3, 4, 5, 6, 10 y 11. No obstante, la restricción 4 es independiente de las variables 1 y 2, por lo que no aparece en dichas figuras. La restricción 5 tiene derivadas nulas respecto a las dos variables del subespacio, por lo que no puede hallarse su aproximación. Las aproximaciones obtenidas se pueden contrastar con la figura (8.24), en la que se incluye el espacio real de diseño, con todas las restricciones.

Por último, en las figuras (8.25), (8.26) y (8.27) se muestran los subespacios de diseño iniciales. Es decir, los subespacios calculados con la información inicial del programa. Además, se incluye la evolución del diseño desde ese diseño inicial (punto 0). En las tres figuras se observa también que en las dos primeras iteraciones el diseño apenas se modifica, llegando en la tercera iteración a las cercanias del óptimo. Debe tenerse presente que las restricciones dibujadas son las únicas que están en las cercanias del punto de diseño inicial, y que su aproximación es muy burda (pudiendo diferir bastante de la aproximación basada en la información de iteraciones posteriores).

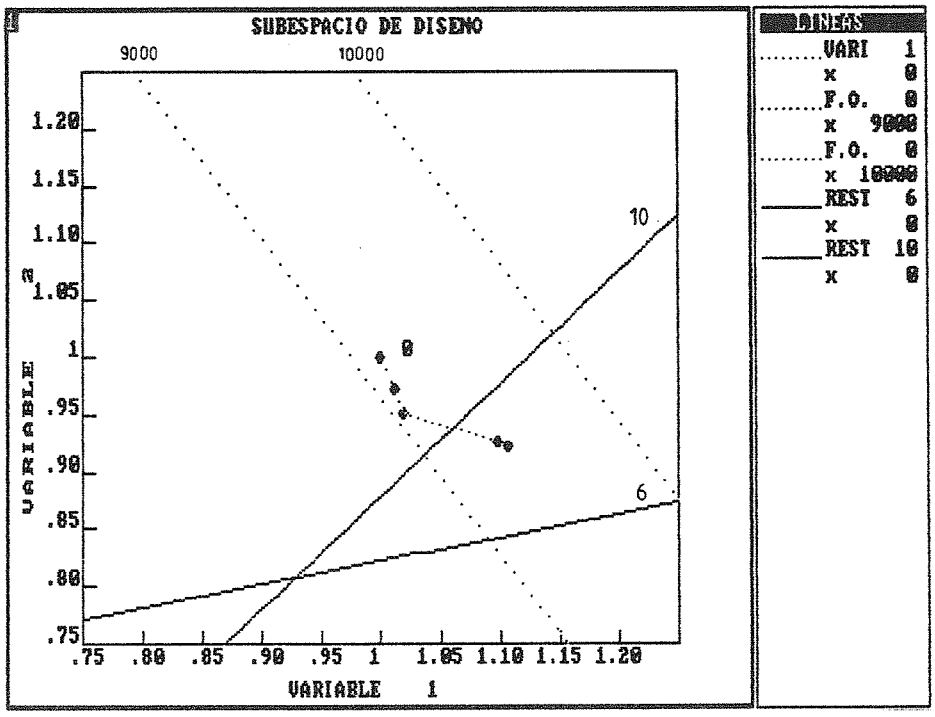

Figura 8.25. Subespacio 1-2 de diseño inicial.

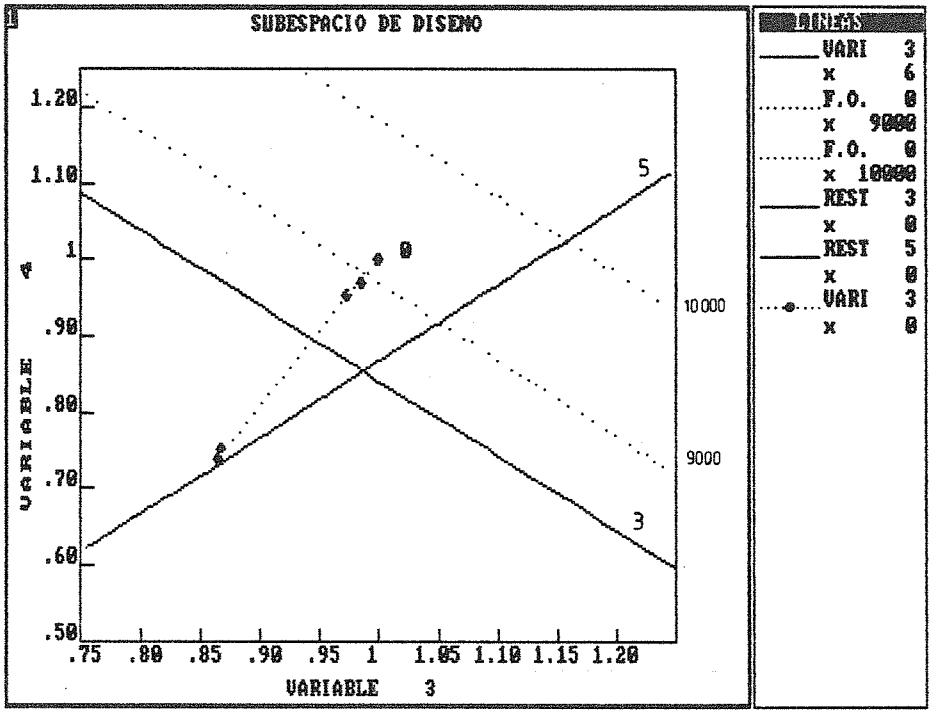

Figura 8.26. Subespacio 3-4 de diseño inicial.

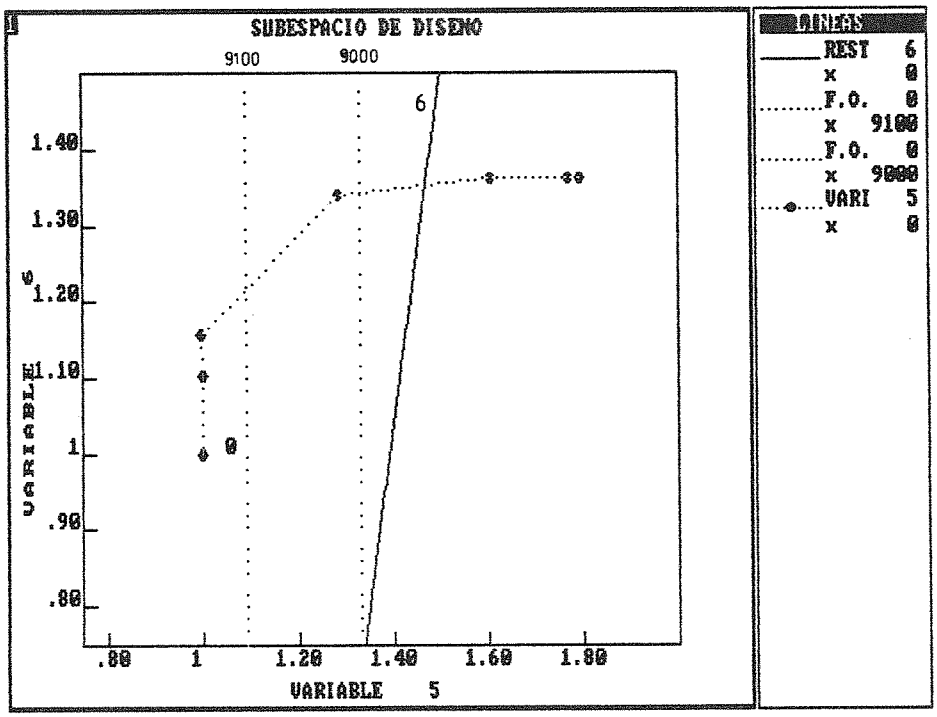

Figura 8.27. Subespacio 5-6 de diseño inicial.

8.4 MALLA ESPACIAL.

8.4.1 Descripción del problema. Criterios de diseño.

Como último ejemplo, vamos a considerar un problema de optimización de una malla espacial de barras de nudos articulados de una capa y siete módulos. El problema está tratado en las referencias /47,49/. La malla está representada en la figura (8.28).

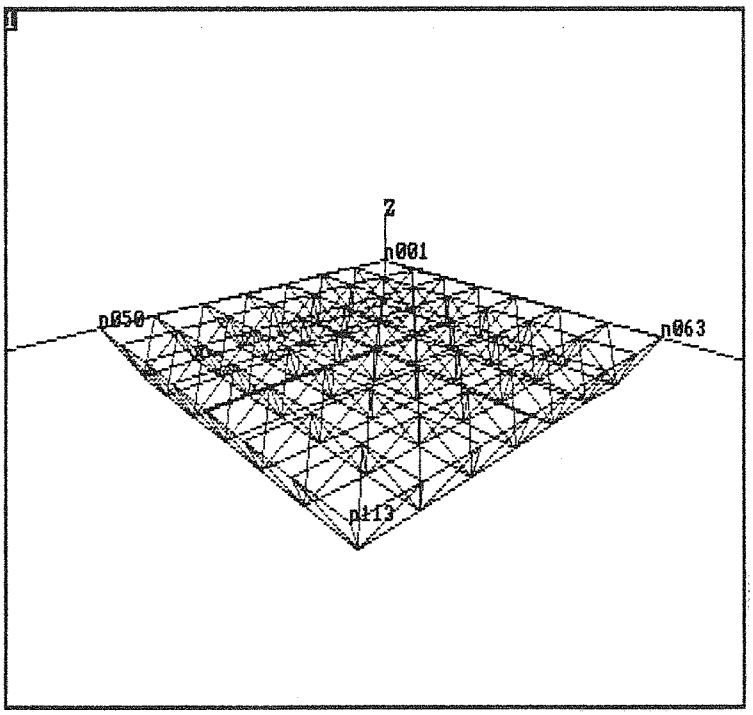

Figura 8.28. Malla espacial.

Los datos son:

Luz (L)	30.0 m.
Canto (H)	3.0 m.
Peso específico del material (γ)	7820.0 Kg/m^3
Modulo de Elasticidad (E)	2100000.0 Kp/cm^2
Tensión máxima admisible (σ_u)	2600.0 Kp/cm^2

La malla está apoyada en las cuatro esquinas; con movimiento totalmente restringido en el nudo 1; restricción en Y y Z para el nudo 50, y restricción en Z para los nudos 63 y 113. La malla se ha definido con cinco grupos de elementos, siendo iguales las propiedades

de todos los elementos de cada grupo. Los cinco grupos son (tal como se vé en las figuras (8.29), (8.30) y (8.31)):

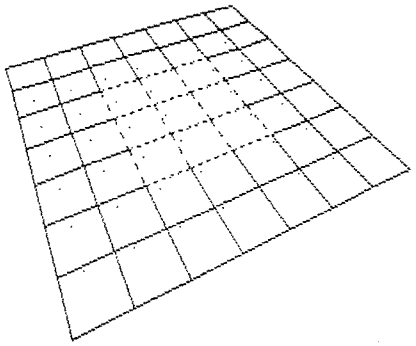

Figura 8.29. Grupos 1 y 2 de elementos.

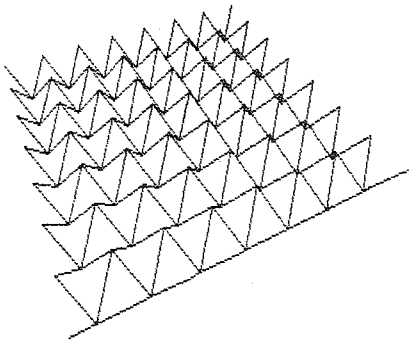

Figura 8.30. Grupo 3 de elementos.

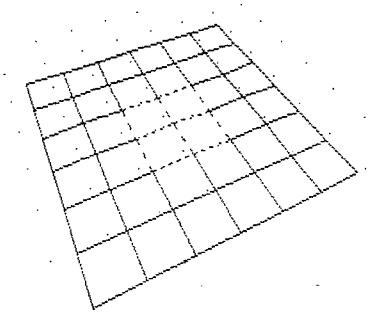

Figura 8.31. Grupos 4 y 5 de elementos.

1. barras de la parte exterior de la capa superior;

2. barras de la parte interior de la capa superior;

3. barras diagonales;

4. barras de la parte exterior de la capa inferior, y

5. barras de la parte interior de la capa inferior.

El propósito del diseño es el de definir las áreas de las secciones rectas de las barras que hagan la estructura lo más ligera posible. Por tanto se toman como variables de diseño las áreas de los cinco grupos de elementos.

El modelo se va a diseñar sometido a dos estados de carga:

1. Sobrecarga uniforme ponderada 150.0 Kp/m^2.

2. Sobrecarga uniforme ponderada (succión) -66.5 Kp/m^2.

Las condiciones que imponen al diseño son de resistencia y de pandeo.

8.4.2 Formulación del problema.

Puesto que se pretende obtener la estructura más ligera posible, se ha tomado como función objetivo el peso de la misma. Así, la expresión de dicha función queda como:

$$F = \gamma \sum_{i=1}^{Ne} L_i \, A_i \qquad (8.28)$$

Siendo:

N_e el número de elementos.

L_i la longitud del elemento i.

A_i es área de la sección recta del elemento i.

Todas las restricciones se han normalizado a valor 1, habiendose considerado una restricción de tensión/pandeo (la mas desfavorable) por cada grupo de elementos (para lo cual se elige el elemento más cargado en cada caso), para cada estado de cargas.

8.4.3 Resultados del estudio.

8.4.3.1 Resultados del análisis.

La deformada de la estructura, bajo los dos estados de carga, se muestra en las figuras (8.32) y (8.33). Estas deformadas se han representado amplificadas por un factor de 20 y 30, respectivamente, para hacer más visibles los efectos de la carga.

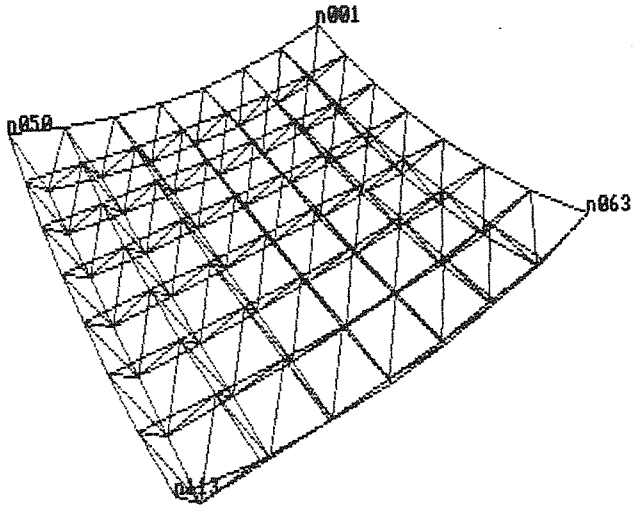

Figura 8.32. Deformada del primer estado de cargas.

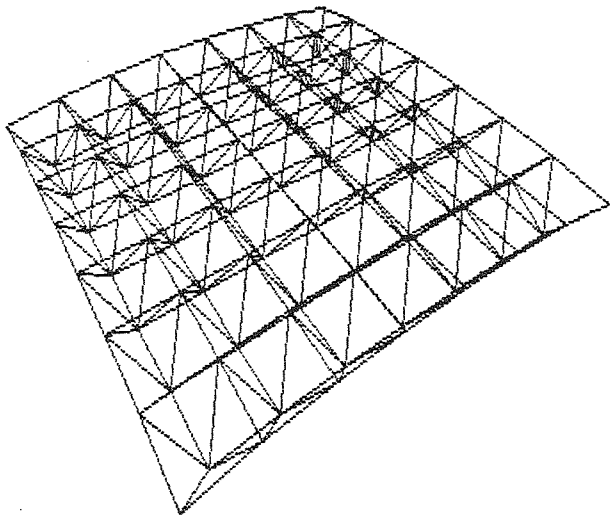

Figura 8.33. Deformada del segundo estado de cargas.

8.4.3.2 Evolución.

En la figura (8.34) se muestra la evolución de la función objetivo y de las variables. En dicha figura puede verse como la función objetivo no sufre variación apreciable a lo largo del proceso de diseño. No obstante, el diseño inicial sí que se modifica, tal como pone de manifiesto la variación de las variables. Es decir, que el diseño final es tan caro como el inicial, pero se adapta mejor a las condiciones impuestas.

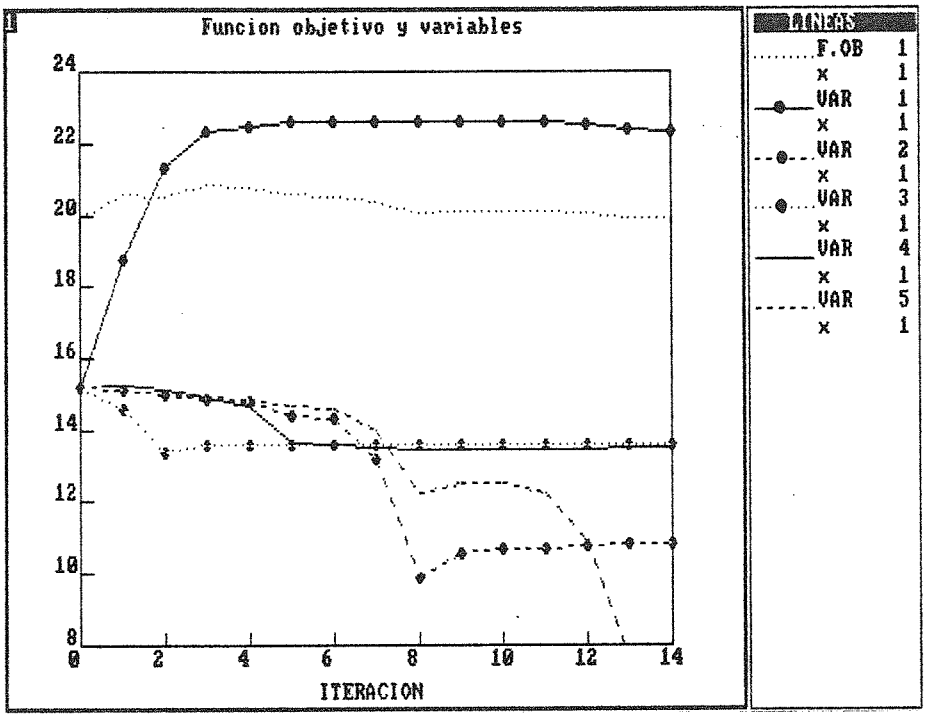

Figura 8.34. Evolución de la función objetivo y las variables.

En las figuras (8.35) y (8.36) se incluyen las evoluciones de las restricciones. En dichas figuras puede observarse que las restricciones asociadas al segundo estado de carga no son críticas, y que las restricciones críticas son las asociadas con la limitación de tensión/pandeo de las barras de los grupos 1 a 4 para el primer estado de cargas.

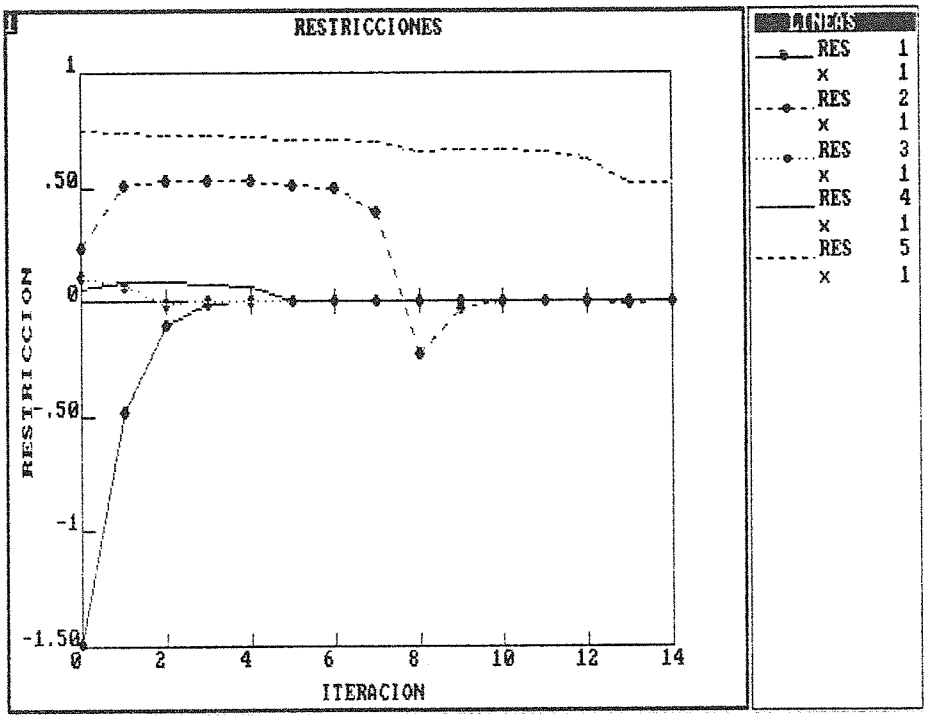

Figura 8.35. Restricciones para el primer estado de cargas.

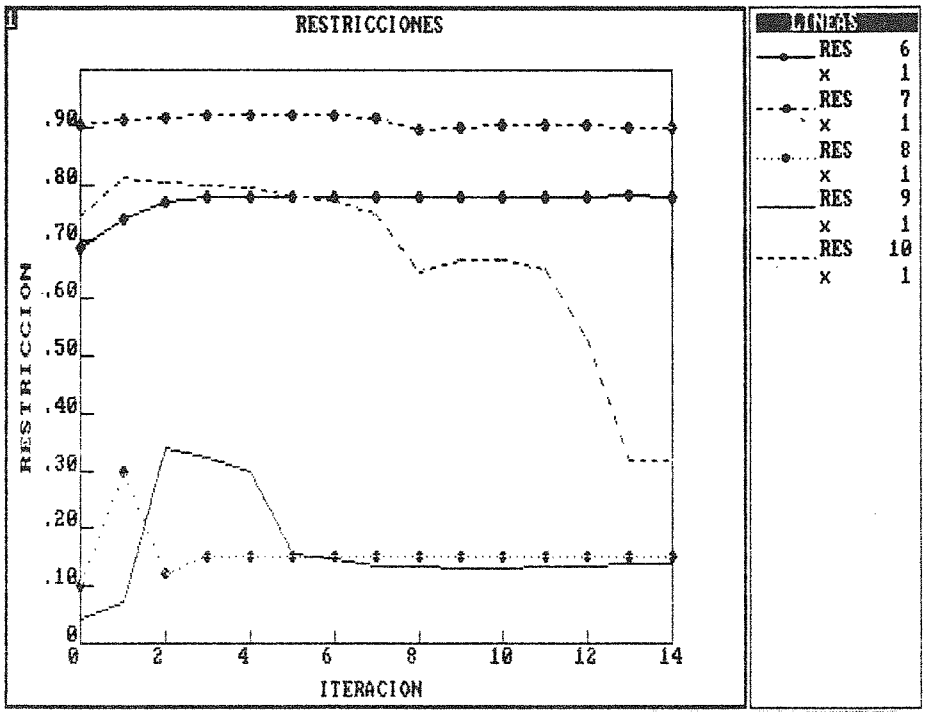

Figura 8.36. Restricciones para el segundo estado de cargas.

8.4.3.3 Espacio de diseño.

El espacio de diseño de este ejemplo se muestra en las figuras (8.37) a (8.40), por medio de los subespacios obtenidos tomando como referencia común la variable 1.

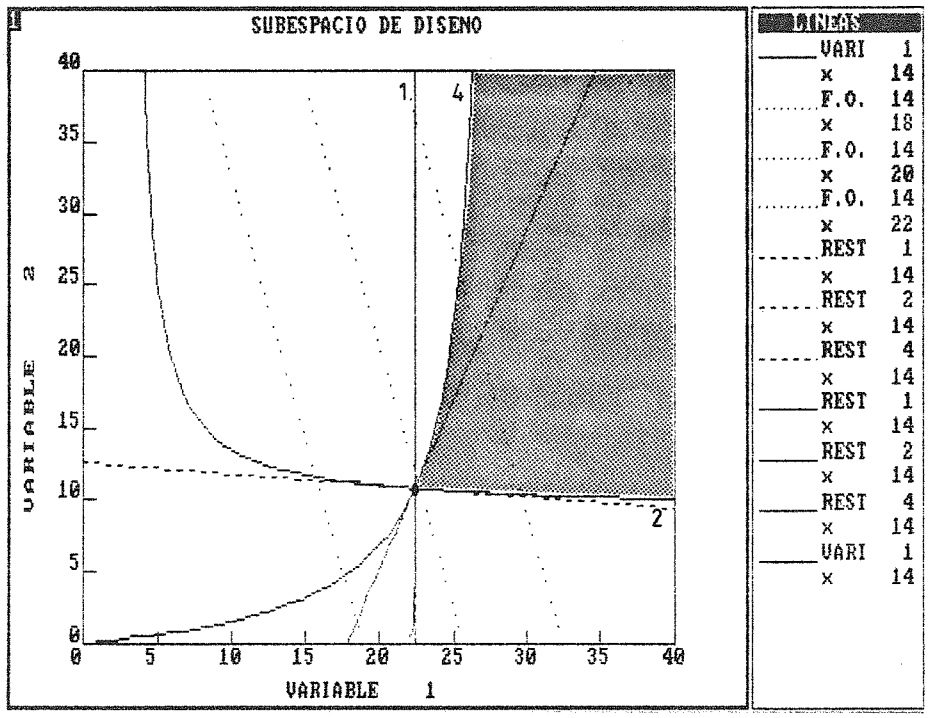

Figura 8.37. Subespacio de diseño 1-2.

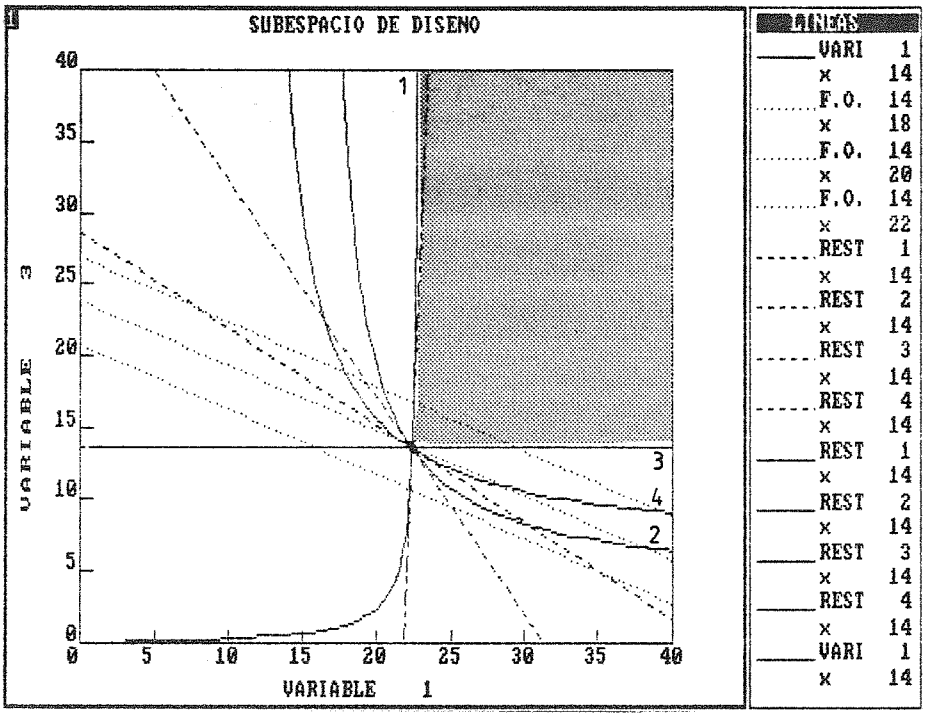

Figura 8.38. Subespacio de diseño 1-3.

En las figuras se observa como cada una de las restricciones activas depende principalmente de la variable asociada al área del grupo de elementos que restringe. El caso más destacado es el de la restricción 3, que únicamente aparece activa en el subespacio que incluye la variable 3.

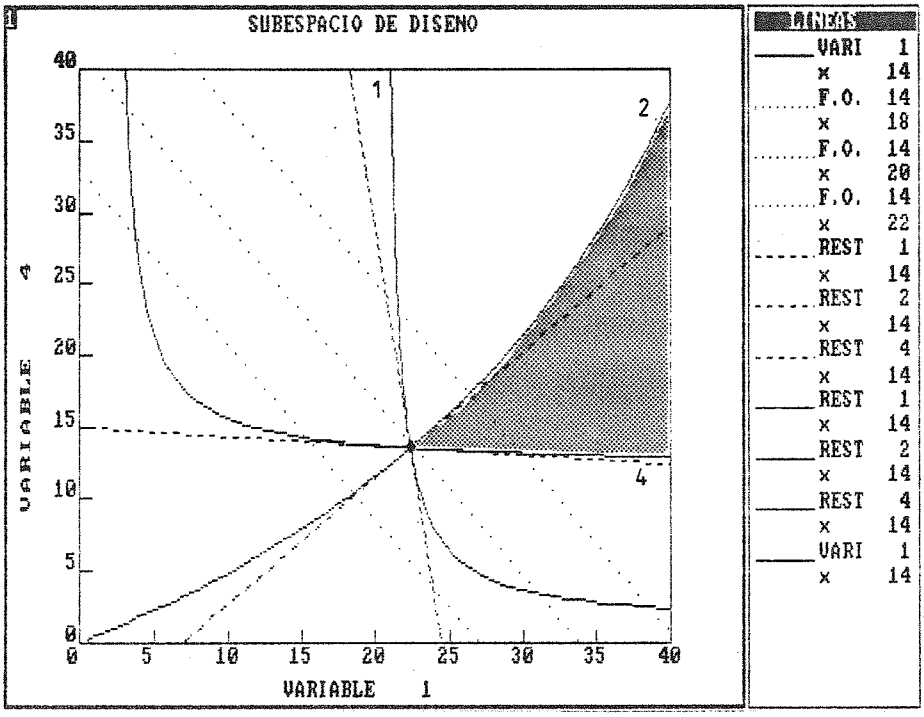

Figura 8.39. Subespacio de diseño 1-4.

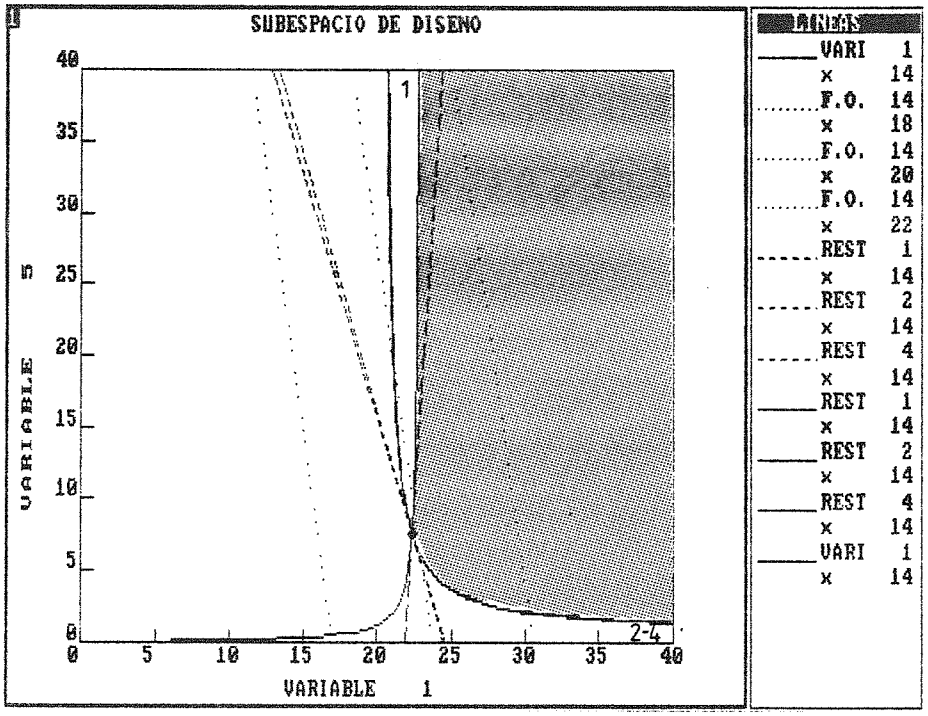

Figura 8.40. Subespacio de diseño 1-5.

CAPITULO 9

CONCLUSIONES Y SUGERENCIAS PARA TRABAJOS POSTERIORES

9.1 INTRODUCCION.

En este trabajo se ha analizado el problema de diseño óptimo de estructuras mediante programación matemática no lineal, estudiando la forma de aplicar las tecnologías de diseño asistido por ordenador (CAD) para conseguir un sistema de diseño interactivo que sea de propósito general y fácil de usar.

Se ha visto que la solución para que los sistemas de diseño automático evolucionen a los de diseño interactivo pasa por la utilización de las tecnologías desarrolladas en el campo del diseño asistido por ordenador (entendido en su sentido más amplio). Dos de las áreas que se consideran del mayor interés son las técnicas de gestión de la información, y las técnicas de interacción (considerando la representación gráfica como parte fundamental de la interacción). Además, como paso previo al desarrollo de cualquier sistema de programación, se debe determinar la metodología de programación que se va a seguir.

A continuación se resumen las conclusiones sobre cada uno de estos aspectos, derivadas del estudio y desarrollo de la aplicación de las técnicas CAD al sistema de diseño óptimo de estructuras DISSENY. Se incluye también una discusión sobre ciertos aspectos del problema, y se exponen algunas sugerencias para futuras investigaciones y desarrollos.

9.2 CONCLUSIONES.

El diseño interactivo requiere la posibilidad de ejecutar tareas independientemente (incluso en ordenadores diferentes) para poder encadenar los flujos particulares requeridos por los usuarios. Para lograr esta separación de tareas, la metodología de la programación

que se ha considerado más apropiada es la de descomposición en módulos, debido a que este método conlleva un cierto tipo de organización final del sistema apropiado a las características del problema interactivo.

Por lo que respecta a la gestión de la información, hemos visto que por la variedad y cantidad de la misma, la mejor solución es desarrollar una base de datos adaptada al diseño óptimo de estructuras. Además, la posibilidad de flujos de ejecución diversos (o incluso repartidos en diferentes ordenadores) requiere una gestión de la información centralizada y robusta. No obstante, dado que el desarrollo de una base de datos es costoso, y que en la actualidad no se dispone comercialmente de bases de datos específicas para el problema de diseño óptimo, se ha adoptado una alternativa más sencilla consistente en desarrollar el esquema conceptual de la base de datos, y recurrir posteriormente a una implementación del esquema físico directamente sobre el código del sistema. La implementación de esta gestión de la información se ha mostrado muy apropiada para su uso en ordenadores personales.

La interacción se ha considerado desde los tres aspectos de: realimentación, lenguaje de comandos y modelo de usuario. Para la realimentación se ha utilizado la escritura de ecos de instrucciones, mensajes indicando operaciones en curso, y detección de errores con escritura de mensajes orientativos. En cuanto al lenguaje de comandos, este se ha basado en una estructura de menús jerarquizados por considerarse la mejor solución para un problema que tiene partes con diferentes niveles de interacción. Además, se pretende que el sistema pueda ser utilizado por usuarios con diferente nivel de conocimiento sobre el diseño óptimo de estructuras, por lo cual se ha buscado con los menús jerarquizados la posibilidad de un uso básico fácil, sin coartar usos más sofisticados. Por último, se ha visto que el modelo de usuario, basado en el modelo de análisis, debe evolucionar hacia un modelo de diseño en el que el de análisis quede embebido y sea generado por el sistema de forma transaprente al usuario. Es decir que el usuario debe poder redefinir el problema, sin entrar en detalles que afectan a la modelización de la estructura requerida para poder analizarla.

Para que la interacción sea posible, además de los tres aspectos ya comentados, es fundamental que el usuario disponga de toda la información sobre el proceso que debe controlar. La forma más efectiva

de informar al usuario se consigue haciendo un uso intensivo de representaciones gráficas. En este sentido, se ha considerado el problema descompuesto en la representación de la estructura a diseñar y la representación del propio proceso de diseño.

Para representar la estructura a diseñar, se ha recurrido a la perspectiva cónica, por ser el tipo de representación más semejante a la visión humana. Se ha tenido en cuenta que el objetivo de estas representaciones es el de mostrar la estructura ya definida, o (en todo caso) ayudar a definirla; pero, al contrario que en los programas de dibujo asistido, la facilidad para que el usuario dibuje con el ordenador no es fundamental. También se ha optado por la representación en cónico porque, además de ser fácil de interpretar, se basa en una formulación tan general que permite conseguir otros tipos de representaciones modificando ciertos parámetros.

Se han discutido las dos alternativas de adaptar un programa de dibujo asistido, o desarrollar uno específico; llegandose a la conclusión de que la mejor solución es desarrollar un programa específico. Los tiempos requeridos en ambos casos se han estimado semejantes, valorandose más la capacidad de modificación del programa específico (por ser totalmente accesible) de cara a investigación sobre formas alternativas de representación. También se ha tenido en cuenta que el programa específico puede ser menos voluminoso que los programas comerciales (que disponen de muchas opciones carentes de utilidad en este caso).

La representación del proceso de diseño se ha considerado descompuesta en dos partes: la evolución del proceso, y el estado actual del mismo. Para la representación de la evolución se ha recurrido a diagramas de línea continua, que admiten la información (bastante compleja) habitual en el proceso de diseño (superposiciones, escalados, etc.). De esta forma se pueden representar todas ls magnitudes implicadas en el proceso. También se ha incluido una forma alternativa de representar estas magnitudes: la representación cruzada de relaciones entre parejas de magnitudes.

Por último, se ha abordado la representación del estado del diseño por medio del espacio de diseño. Dada la limitación de dos dimensiones de la pantalla del ordenador, se ha recurrido a representar subespacios de dos dimensiones, con la función objetivo y las restricciones representadas por medio de las curvas de valor constante de las mismas. Puesto que se dispone de poca información de las funciones a

representar (generalmente el valor de las mismas y el de sus primeras derivadas, para el punto de diseño) se ha recurrido a aproximaciones para estimar la forma de las mismas. Las aproximaciones empleadas se han basado en desarrollos en serie de Taylor (truncados en la primera derivada). Estos desarrollos se han modificado para aceptar aproximaciones alternativas que se adapten mejor a las particularidades de los tipos de funciones a representar. Por ello, se han contemplado las aproximaciones:

- directa (desarrollo en serie de Taylor truncado en las primeras derivadas).

- inversa (semejante a la anterior pero basada en las inversas de las variables).

- híbrida (en la que cada sumando del desarrollo considera bien la variable o bien su inversa, de forma que se obtenga una estimación mas conservadora del espacio de diseño).

9.3 DISCUSION Y SUGERENCIAS PARA TRABAJOS POSTERIORES.

Se ha establecido un modelo de uso de las tecnologías CAD para facilitar la comprensión y uso de los sistemas de diseño óptimo de estructuras. La validez del modelo se ha contrastado por medio de varios ejemplos representativos.

Se ha puesto de manifiesto que para que un sistema de diseño pueda ser usado con comodidad por usuarios no avezados el nivel de interacción debe ser muy completo. En este sentido, se debe completar la capacidad de interacción añadiendo las ayudas mas habituales (entrada por ratón, tableta digitalizadora, etc).

Por otra parte, aunque se han utilizado ciertas formas de representación que son habituales en otros campos, y por lo tanto perfectamente conocidas, también se han propuesto representaciones mas innovadoras, sobre las que un uso mas exaustivo debe conducir a extraer de ellas toda la información que suministran (y que por falta de experiencia puede quedar oculta).

Los resultados obtenidos en la representación del espacio de diseño son francamente esperanzadores. Por lo que se propone como línea de investigación a seguir la búsqueda de formas alternativas de utilizar la información disponible para conseguir aproximaciones del espacio de

diseño más ajustadas al espacio real. Para ello, el camino que parece más prometedor es el de usar la información acumulada de todas las iteraciones previas del proceso de diseño. El objetivo de que las representaciones no requieran costosos cálculos adicionales para complementar la información necesaria para el algoritmo de optimización, junto con la diversidad de esta información en diferentes algoritmos, debe conducir a la necesidad de que estas formas de representación sean flexibles para conseguir representaciones en función de diferente información de partida.

REFERENCIAS

1- Arora J.S., y Haug E.J.
Methods of Design Sensitivity Analysis in Structural Optimization.
AIAA Journal, 17. pp. 823-832, 1979.

2- Atrek E.; Gallagher R.H.; Ragsdell K.M., y Zienkiewicz O.C.: editores.
Mew Directions in Optimum Structural Design.
Ed. John Wiley & Sons, New York, 1984.

3- Balling R.J.
DELIGHT.STRUCT: I. A Computer-Aided Design Environement for Structural Engineering.
M.S. Thesis. University of California, Berkeley, 1982.

4- Balling R.J.; Pister K.S., y Polak E.
DELIGHT.STRUCT: An Optimization-Based Computer-Aided Design Environment for Structural Engineering.
Computer Methods in Applied Mechanics and Engineering. Vol. 38, pp. 237-251, 1983.

5- Barta J.
On the Minimum Weight of Certain Redundant Structures.
Acta Tech. Acad. Sci. Hungar. 18, pp. 67-76, 1957.

6- Bathe K.J.,y Wilson E.L.
Numerical Methods in Finite Elements Analysis.
Ed. Prentice Hall, Englewood Cliffs, New Jersey, 1976.

7- Bazaraa M.S., y Shetty C.M.
Nonlinear Programming: Theory and Algorithms.
Ed. John Wiley & Sons, New York, 1979.

8- Bhatia K.G.
Rapid Iterative Reanalysis for Automated Design.
NASA TN 6534, 1971.

9- Beckers P.
CAD Technology in Optimal Designs.
En: Mota Soares C.: editor.
Computer Aided Optimal Design: Structural and Mechanical Systems.
Ed. Springer-Verlag, Berlin-Heidelberg. pp 805-829, 1987.

10- Belegundu A.D.
A Study of Mathematical Programming Methods for Structural Optimization.
M.S. Thesis. University of Iowa, 1982.

11- Bertin J.
La Graphique el le Traitment Graphique de L'Information.
Ed. Flammarion, Paris, 1977.

12- Booch G.
Software Engineering with ADA
Ed Benjamin/Cummings Publishing, Merlo Park (California), 1983

13- Company P., y Martí P.
Memoria secundaria en FORTRAN-77.
Memoria interna de trabajo, U.P. de Valencia, Julio 1986.

14- Company P., y Martí P.
Organización y Manejo de Información en el Sistema de Diseño Optimo de Estructuras DISSENY.
Anales de Ingeniería Mecánica, No. 5(2), pp. 33-38, 1987.

15- Company P.; Sanchis M., y Martí P.
Memoria Dinámica en FORTRAN-77.
Memoria interna de trabajo, U.P. de Valencia, Enero 1986.

16- Encarnacao J.,y Schlechtendahl E.G..
Computer-Aided Design.
Ed. Springer-Verlag, Berlin-Heidelberg, 1979.

17- Esping B.J.D.,y Holm D.
A CAD Aproach to Structural Optimization.
En: Mota Soares C.: editor.
Computer Aided Optimal Design: Structural and Mechanical Systems.
Ed. Springer-Verlag, Berlin-Heidelberg. pp 987-1001, 1987.

18- Felippa C.A.
Database Management in Scientific Computing-I General Description.
Computer & Structures, Vol. 10 (1), pp. 53-61, 1979.

19- Fiacco A., y McCormick G.P.
Nonlinear Programming: Sequential Unconstrained Minimization Techniques.
Ed. John Wiley & Sons, New York, 1968.

20- Fletcher R.
Practical Methods of Optimization.
Ed. John Wiley & Sons, New York, 1981.

21- Fleury C.
Introduction to Mathematical Programming Methods.
En: Morris A.J.: editor.
Foundations of Structural Optimization: A Unified Approach.
Ed. John Wiley & Sons, New York, 1982.

22- Fleury C.
Reconciliation of Mathematical Programming and Optimality Criteria Methods.
En: Morris A.J.: editor.
Foundations of Structural Optimization: A Unified Approach.
Ed. John Wiley & Sons, New York, 1982.

23- Fleury C.
Computer Aided Optimal Design of Elastic Structures.
En: Mota Soares C.: editor.
Computer Aided Optimal Design: Structural and Mechanical Systems.
Ed. Springer-Verlag, Berlin-Heidelberg 1987. pp 832-900

24- Foley J.D.,y Van Dam A.
Fundamentals of Interactive Computer Graphics.
Ed Addison Wesley, 1982 (reprinted, July 1984)

25- Fox R.L.
Optimization Methods for Engineering Design.
Ed. Addison Wesley, 1971.

26- Gallagher R.H.
Terminology and Basic Concepts.
En: Gallagher R.H., y Zienkiewicz O.Z.: editores.
Optimum Structural Design: Theory and Applications.
Ed. John Wiley & Sons, New York, 1973.

27- Gellatly R.A.,y Berke L.
Optimality-Criterion-Based Algorithms.
En: Gallagher R.H., y Zienkiewicz O.Z.: editores.
Optimum Structural Design: Theory and Applications.
Ed. John Wiley & Sons, New York, 1973.

28- Gill P.; Murray W.; Saunders M.,y Wright M.
User's Guide for SOL/QPSOL: A Fortran Package for Quadratic Programming.
Report SOL 82-7, Systems Optimization Laboratory, Standford University, Standfors, 1982.

29- Gill P.; Murray W.,y Wright M.
Practical Optimization.
Ed. Academic Press, Orlando (Florida), 1981.

30- Han S.P.
A Globally Convergent Method for Nonlinear Programming.
J. Optimization Theory Appl. 22, pp. 297-309, 1977.

31- Haug E.J.,y Arora J.S.
Applied Optimal Design.
Ed. John Wiley & Sons, New York, 1979.

32- Haug E.J.; Kyung K. C.,y Komkov V.
Design Sensitivity Analyisis of Structural Systems.
Ed. Academic Press, Orlando (Florida), 1986.

33- Holzer S.M.
Computer Analyisis of Structures. Matrix Structural Analysis. Structured Programming.
Ed. Elsevier, New York, 1985.

34- Hopgood F.R.A.; Duce D.A.; Gallop J.R., y Sutcliffe D.C.
Introduction to the Graphical Kernel System (GKS).
Ed. Academic Press, Orlando (Florida), 1984.

35- Hörnlein H.R.E.M.
Take-off in Optimum Structural Design.
En: Mota Soares C.: editor.
Computer Aided Optimal Design: Structural and Mechanical Systems.
Ed. Springer-Verlag, Berlin-Heidelberg. pp 901-919, 1987.

36- Isner J.F.
A FORTRAN Programming Methodology Based on Data Abstraction.
Communications of the ACM, Vol. 25 (10), pp. 686-697, 1982.

37- Jones Ch.
Métodos de Diseño.
Ed. Gustavo Gili, Barcelona, 1978.

38- Kanji I., y Schmit L.A.
Configuration Optimization of Trusses.
Journal of the Structural Division, ASCE, Vol. 107, No.ST05, pp. 745-756, May, 1981.

39- Kardestuncer H.
Introducción al Análisis Estructural con Matrices.
Ed. Mc Graw-Hill de Mexico S.A., México, 1975.

40- Kirsch U.
Optimum structural Design: Concepts, Methods and Applications.
Ed. Mc Graw-Hill, New York, 1981.

41- Klein B.
Direct Use of External Principles in Solving Certain Optimization Problems Involving Inequalities.
J. ORSA, 3, pp 168-175, 1, 1955.

42- Kuhn H.W., y Tucker A.W.
Nonlinear Programming.
Proc. 2nd Berkeley Symp. on Mathematical Statistics and Probability, Univ. California Press, pp. 481-492, 1951.

43- Kunz D.L.,y Hopkins, A.S.
Structured Data in Structural Analysis Software
Computer & Structures, Vol. 26 (6), pp. 965-978, 1986.

44- Larmouth J.
FORTRAN-77 Portability.
Software Practice and Experience. Vol 11, pp. 1071-1117, 1981.

45- Lawson C.L., y Hanson R.J.
Solving Least Squares Problems.
Ed. Prentice Hall, Englewood Cliffs, New Jersey, 1974

46- Martí P.
Optimización de vigas armadas mediante programación matemática no lineal.
Memoria interna de trabajo, Valencia, Mayo-Junio 1988.

47- Martí P.,y Company P.
Optimización de Mallas Espaciales de Nudos Articulados.
Anales de Ingeniería Mecánica, No. 5(2), pp. 147-152, 1987.

48- Martí P.; Company P.,y Sanchis M.
DISSENY. Un sistema interactivo para el diseño de estructuras basado en técnicas de optimización.
Anales de Ingeniería Mecánica, No. 3(1), pp. 285-290, 1985.

49- Martí P.; Company P.,y Mas F.
Diseño Optimo de Estructuras Espaciales de Nudos Articulados con Geometría Variable.
Anales de Ingeniería Mecánica, No. 4(2), pp. 213-218, 1986.

50- Martí P.; Company P.,y Sanchis M.
Acoplamiento elementos finitos-técnicas de optimización en el sistema DISSENY.
II Simposium sobre Aplicaciones del Método de los Elementos Finitos en Ingeniería, Barcelona, pp. A-133 a A-147. 1986.

51- Martí P.; Sanchis M.,y Company P.
Programa ADEF. Analisis de Estructuras por Elementos Finitos. Manual de Usuario.
Ed. Serv. Public. U.P.Valencia, Valencia, 1985.

52- Martí P.; Sanchis M.,y Company P.
Lectura Libre en FORTRAN-77.
Memoria interna de trabajo, U.P. de Valencia, Enero 1986.

53- Mas F.
Diseño, Implementación y Prueba de un Programa de Análisis y Diseño de Estructuras Metálicas de Nudos Articulados.
P.F.Carrera, U.P. Valencia, Valencia, 1987

54- Media Cibernetics, Inc.
HALO. Graphics Kernel and Device Drivers. Version 2.26. Funcional Description Manual. 1986.

55- Melosh R.J., y Luik R.
Approximate Multiple Configuration Analysis and Allocation for Least Weight Structural Design.
AFFDL-TR-67-29, 1967.

56- Microsoft Corporation.
Microsoft FORTRAN Compiler. User's Guide & Reference Manual. Version 3.30, 1985.

57- Mitchell A.G.M.
The Limits of Economy of Material in Framed Structures.
Phil. Mag. (Series 6),8, pp. 589-597, 1904.

58- Morris A.J.
Basic Mathematical Concepts.
En: Morris A.J.: editor.
Foundations of Structural Optimization: A Unified Approach.
Ed. John Wiley & Sons, New York, 1982.

59- Mota Soares C.: editor.
Computer Aided Optimal Design: Structural and Mechanical Systems.
Ed. Springer-Verlag, Berlin-Heidelberg 1987.

60- Newman W.M.,y Sproull R.F.
Principles of Interactive Computer Graphics.
Ed. McGraw-Hill, 1979.

61- Pearson C.W.
Structural Design by High Speed Computing Machines.
Proceedings of the First Conference on Electronic Computation, ASCE, New York, pp. 417-436, 1958.

62- Pedersen P.
Optimum Joint Position for Space Trusses.
Journal of the Structural Division, ASCE,No.ST12 Proc. Paper 10199, pp. 2459-2475, December, 1973.

63- Peiming Wu.
Structural Optimization with Nonlinear and Discrete Programming.
M.S. Thesis. University of Wisconsin-Madison, 1986.

64- Powell M.J.D.
A Fast Algorithm for Nonlinear Constrained Optimization Calculations.
Lecture Notes in Mathematics, Vol. 630. Springer Verlag, 1978.

65- Pshenichny B.N.
Algorithms for the General Problem of Mathematical Programming.
Kibernetica, 5,pp. 120-125, 1970.

66- Rajan S.D.,y Bhatti, M.A.
On Designing a Database Management System for a Computer-Aided Engineering Software System.
Computer & Structures Vol. 21(5), pp. 1047-1057, 1985.

67- Rajan S.D.,y Bhatti, M.A.
SADDLE: A Computer-Aided Structural Analysis and Dynamic Design Language. Part II. Database Management System.
Computers & Structures Vol. 22(2), pp. 205-212, 1986.

68- Sabourin R.;Lowther D.A., y Webb P.
GBASE. A Generalised Database Management System for CAD.
Computer-Aided Engineering Journal, pp. 207-212, October 1986.

69- Schittkowski K.
On The Convergence of a Sequential Quadratic Programming Method with an Augmented Lagrangian Line Search Function.
Math. Operationsforsch u Statist. Ser. Optimization, 14(2), pp 197-216, 1983.

70- Schmit L.A.
Structural Design by Systematic Synthesis.
Proceedings of the Second Conference on Electronic Computation, ASCE, Pittsburgh, pp. 105-122, 1960.

71- Sreekanta-Murthy T.
Computer-Aided Structural Design Optimization Using a Database Management system.
M.S. Thesis. University of Iowa, 1985.

72- Vanderplaats G.N.
Numerical Optimization Techniques for Engineering Design: With Applications.
Ed. Mac Graw-Hill, 1984.

73- Vanderplaats G.N., y Moses F.
Autommated Design of Trusses for Optimum Geometry.
Journal of the Structural Division, ASCE,No.ST3 Proc. Paper 8795, pp. 671-690, March, 1972.

74- Wilson R.B.
A Simplicial Algorithm for Concave Programming.
Ph. D. Thesis, Graduate School of Business Administration, Hardward University, Boston, 1963.

75- Wilson E.L.,y Hoit M.I.
A Computer Adaptative Languaje for the Development of Structural Analysis Programs.
Computer & Structures, Vol. 19 (3), pp. 321-338, 1984.

76- Zienkiewicz O.C.
El Método de los Elementos Finitos.
Ed. Reverté S.A., Barcelona, 1980.

77- Zoutendijk G.
Methods of Feasible Directions.
Ed. Elsevier, New York, 1960.