UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

DIVISION DE ESTUDIOS DE POSGRADO
FACULTAD DE INGENIERÍA

ANÁLISIS DE ESFUERZOS EN ESTRUCTURAS DE ACERO UTILIZANDO EL MÉTODO DEL ELEMENTO FINITO BAJO EL ENFOQUE ORIENTADO A OBJETOS CON UTILERÍAS DE DIFFPACK Y LENGUAJE C++ (CASO ISOPARAMÉTRICO EN 2D).

T E S I S

PARA OBTENER EL GRADO DE

MAESTRO EN INGENIERIA

P R E S E N T A :

ESCOBAR MORENO LIZ. ROCÍO E.

ASESOR : M. EN I. FRANCISCA IRENE SOLER ANGUIANO

CIUDAD UNIVERSITARIA, MÉXICO, D.F. MARZO DE 2002

Dedicatoria :

Dedico esta tesis especialmente a mi hija Valeria, que da a mi vida la luz y fuerza necesarias para seguir adelante y que día con día me demuestra lo maravillosa que es la vida y la bondad de Dios. Muchas gracias por existir hijita.

Agradecimientos:

Mis más sincero agradecimiento a la M. en I. Francis Soler y al M. en I. Augusto Villarreal.

Doy gracias también al Dr. Jorge Carrera Bolaños, al Dr. Vladimir Tchijov y a la Dra. Ana María Vázquez por todo el tiempo y empeño que dedicaron a esta tesis y por sus comentarios que enriquecieron esta tesis.

Deseo hacer especial mención al ingeniero Sergio F. Beltrán, gracias por todo su apoyo.

Por otra parte, agradezco a todas las instituciones que me apoyaron: Intituto de Ingeniería, Fundación Telmex y especialmente DGAPA.

Sinceramente, gracias por todo y por tanto.

MARZO DE 2002

ÍNDICE

INTRODUCCIÓN

INTRODUCCIÓN

Actualmente se emplea para modelación de elementos en el medio continuo una técnica basada en métodos numéricos avanzados, llamada del elemento finito. Dicha técnica es compleja aún para los ingenieros y personal especializado, debido a su dificultad matemática y a la profundidad en conocimientos de computación que requiere (estructuras de datos, bases de datos, graficación por computadora, etc.), además de la instrucción vasta del tema en particular relacionada con problema que se pretende resolver.

Existen en el extranjero y aquí en el país (en instituciones de investigación, tales como la UNAM y el IPN) paquetes de elemento finito, los cuales, como se mencionó previamente, requieren que el personal que los emplee tenga cuando menos estudios de maestría en matemáticas, física, ingeniería o ciencias, aunque sería recomendable que tuviese estudios de doctorado.

Por otra parte, el avance en el desarrollo de software ha acusado un excelente adelanto al desarrollarse la tecnología orientada a objetos (TOO), la cual posee algunas ventajas importantes (que posteriormente serán descritas) con respecto a la programación tradicional.

Los programas de elementos finitos actuales están diseñados siguiendo el enfoque algorítmico, lo cual conlleva a que el experto se enfrente con el desarrollo de cientos de rutinas, subrutinas, corrutinas, etc., las cuales convierten a los programas en verdaderas murallas para el experto, y además consumen enormes cantidades de memoria, mucho espacio en disco duro y por consiguiente un gran tiempo de proceso.

El objetivo de este trabajo de tesis es demostrar que la TOO es una buena alternativa para la solución de problemas en el continuo, ya que al aplicar sus paradigmas permite crear los objetos adecuados para discretizar dicho continuo y manipular las rutinas matemáticas en una forma más "comprensible" para el experto empleando objetos, los cuales se agrupan en forma

totalmente lógica para formar clases y manejar con fluidez la herencia, el polimorfismo, etc., que son términos propios de la TOO.

La idea de realizar esta labor surgió a raíz de que la autora ha colaborado en el diseño de algunos módulos para la solución por computadora de problemas aplicados al medio continuo, en particular para el análisis de esfuerzos en placas de acero para la industria petrolera, empleando la programación tradicional. Después de analizar exhaustivamente los beneficios de la TOO aplicada a este tipo de simulaciones matemáticas, se llegó a la conclusión que no sólo es efectiva para las aplicaciones administrativas sino también dentro del campo científico.

Para lograr estos objetivos se juzgaron durante el análisis dos alternativas: la primera fue utilizar la TOO, mediante alguno de los lenguajes de programación orientados a objetos, tales como el C++, SmallTalk, Eiffel, Pascal orientado a objetos, CLOS, etc., y la segunda fue utilizar el paquete noruego Diffpack, el cual está hecho en C++ y cuenta con bibliotecas especiales para la aplicación de modelos matemáticos.

Siendo objetivos de este trabajo el eliminar en lo posible la complejidad en las etapas de análisis, diseño, codificación, implantación y mantenimiento para el diseñador, así como auxiliar al usuario final, que si bien es experto en el campo de aplicación, no lo es tanto en computación ni en matemáticas, se optó por la segunda alternativa, ya que se mostraría que cumple satisfactoriamente su cometido.

Objetivo general

Diseñar e implantar un software para resolver problemas inherentes al análisis de esfuerzos a través del método del elemento finito con enfoque orientado a objetos, para así mostrar que la TOO no sólo produce excelentes resultados en problemas de índole administrativo, sino que también puede emplearse en problemas científicos y obtener resultados óptimos.

Objetivos particulares

* Desarrollar una metodología que permita solucionar problemas de análisis de esfuerzos en estructuras, de una manera sencilla y comprensible.

* Proporcionar un marco teórico adecuado para comprender la tecnología orientada a objetos y el método del elemento finito.

* Reducir los costos de diseño y operación.

* Dar a conocer las bondades del software Diffpack y el lenguaje de programación C++.

Así, para el desarrollo de este trabajo de tesis se ha planteado la siguiente secuencia agrupada en capítulos, para que el lector vaya introduciéndose en forma sencilla a la solución de los objetivos planteados.

En el capítulo 1 se presenta un panorama general de los que es la tecnología orientada a objetos, tratando estos temas desde el análisis hasta la programación orientada a objetos.

En el capítulo 2 se proporciona una introducción a lo que es el método del elemento finito (MEF), presentando sus ecuaciones más importantes, tanto del comportamiento de los elementos individuales como de los ensamblados.

Como el MEF es un conjunto de métodos para solucionar problemas en el continuo y ésto es modelación matemática, es casi imposible soslayar las matemáticas, pues ello haría que se perdiera la objetividad con la que se pretende desarrollar este capítulo.

En el capítulo 3 se trata de dos casos característicos del MEF que son el estudio de elementos isoparamétricos en 2 dimensiones y el estudio de elementos axisimétricos, sin que ello quiera decir que son los únicos que existen. Lo que sucede es que para la aplicación que aquí se presenta éstos son los elementos más representativos.

Finalmente, en el capítulo 4 se hace la introducción al paquete Diffpack y su aplicación al elemento finito, y en el apéndice A se presenta un caso práctico que se desarrolló con bastante éxito para la industria petrolera, cumpliendo así con los objetivos planteados en este trabajo.

El apéndice B contiene las tablas que se utilizan en la aplicación.

Cabe agregar que la fuente de donde se obtuvo información del problema es el Departamento de Investigación y Desarrollo de la Fábrica de Implementos Petroleros y la Empresa Kalsi Eng. Inc. de Houston, Texas.

CAPÍTULO 1. CONCEPTOS BÁSICOS DE LA TECNOLOGÍA ORIENTADA A OBJETOS

CAPÍTULO 1. CONCEPTOS BÁSICOS DE LA TECNOLOGÍA ORIENTADA A OBJETOS

La orientación a objetos tiene más de treinta años. Sus raíces pueden encontrarse en Noruega a finales de la década de los sesenta cuando Kristen Nygaard y Ole-Johan Dahl desarrollaron el lenguaje Simula67; éste introdujo por primera vez los conceptos de clases, corrutinas y subclases, términos muy parecidos a los que hoy se emplean en la orientación a objetos. A lo largo de todos estos años, la tecnología orientada a objetos ha seguido evolucionando y se ha consolidado; ahora comprende el análisis, el diseño y la programación orientada a objetos. Enseguida se proporciona una descripción breve de la TOO.

1.1) Panorama general

Actualmente existen muchas formas de descomponer un problema complejo, pero las más sobresalientes son por algoritmos y por objetos. El enfoque algorítmico se basa en el ordenamiento de los eventos, en tanto que la orientación a objetos enfatiza que los objetos pueden causar una acción o ser sujetos sobre los cuales estas operaciones actúan. La orientación a objetos conduce a sistemas más pequeños que pueden reutilizar mecanismos comunes, lo que trae como consecuencia, en algunos casos, un ahorro importante de expresiones. Los sistemas orientados a objetos son más flexibles al cambio y son más capaces de evolucionar a través del tiempo porque su diseño está basado en formas intermedias estables (su bloque constructor es el

módulo, el cual representa una colección lógica de clases y objetos en lugar de subprogramas; además no existen o hay muy pocos datos globales). Hasta aquí se ha hablado de objetos y clases sin proporcionar su definición, por lo tanto a continuación se explica lo que son. Un **objeto** es algo que tiene fronteras bien definidas; además posee un estado, un comportamiento y una identidad. El estado se refiere a las propiedades del objeto y a los valores de esas propiedades; dado que un objeto tiene estado, ocupa una cierta cantidad de espacio en el mundo físico o en la memoria de la computadora. El comportamiento o conducta indica como actúa y reacciona en términos de cambios de estado y del paso de mensajes; de acuerdo a su comportamiento los objetos pueden ser pasivos o activos. Un *objeto activo* es autónomo, lo cual significa que exhibe un comportamiento sin que sea operado por otro objeto; en tanto, un *objeto pasivo* únicamente puede cambiar de estado cuando actúan sobre él.

La identidad es la propiedad de un objeto que lo hace distinguible de otros objetos.

Una **clase** representa solamente una abstracción, es un conjunto de objetos que comparten una estructura y comportamiento comunes. Un solo objeto es simplemente una instancia de una clase. La implantación de una clase básicamente consiste de la implantación de todas las operaciones definidas en la interfaz de la clase. Aparte de los tipos de clases ya mencionados previamente, existen otros y son: la clase contenedora, que es una clase cuyas instancias son colecciones de otros objetos, debe denotar colecciones homogéneas; y la metaclase, que es una clase cuyas instancias son clases.

La tecnología orientada a objetos está fundamentada en el modelo del objeto. Éste trabaja sobre los principios de abstracción, encapsulación, modularidad y jerarquía. Otros elementos menores son el tipado, la concurrencia y la persistencia.

La ***abstracción*** denota las características esenciales que distinguen a un objeto de otros tipos de objetos y proporciona fronteras conceptuales bien definidas. Se enfoca en la vista externa de un objeto, así que sirve para separar a la implantación de la conducta esencial de un objeto. Ejemplo de abstracción: en una granja hidropónica, las plantas crecen en una solución nutriente, sin tierra, grava u otros sólidos. Aquí se deben controlar diversos factores como la temperatura, humedad, luz, pH y concentración de nutrientes.

Una de las abstracciones clave en este problema es definir el tipo de sensor que se utilizará; después de elegirlo, hay que decidir que medirá, en qué parte de la granja se localizará, qué operaciones podrá ejecutar un cliente sobre él, etc.

La ***encapsulación*** es el proceso de esconder todos los detalles de implantación de un objeto. Es complementaria de la abstracción. Ejemplo de encapsulación: retomando al sensor de la granja hidropónica, la encapsulación consistiría en ocultar los componentes que lo integran tales como relevadores, resistencias, capacitores, etc.

La ***modularidad*** consiste en dividir un programa en módulos, los cuales pueden ser compilados en forma separada, pero tienen conexiones con otros módulos. El objetivo general de la descomposición en módulos es la reducción del costo del software, al permitir que los módulos sean diseñados y revisados independientemente.

Ejemplo: supóngase que se desea realizar un programa que efectúe la suma de dos matrices, uno de los módulos se encargaría de leer y validar el orden de las

matrices para realizar la suma, otro llevaría a cabo el llenado de cada una de las matrices y un tercero para sumar las matrices.

La ***jerarquía*** consiste en ordenar las abstracciones. La herencia es la jerarquía más importante y es un elemento esencial de los sistemas orientados a objetos, define una relación entre clases o si una clase comparte la estructura o conducta definida en una o varias clases. La herencia representa una jerarquía de abstracciones en la que una subclase hereda de una o más superclases. Casi siempre una subclase aumenta o redefine la estructura y conducta existentes de su superclase (la superclase representa abstracciones generales y la subclase, especializaciones). Cuando una clase hereda sólo de una clase se dice que la herencia es simple; y cuando hereda de dos o más, se habla de herencia múltiple, se debe ser cuidadoso con ésta porque puede ocasionar colisiones o repetición de herencia.

Debido a la herencia, la interfaz de una clase necesita comprender tres partes: privada (private), que es donde se declaran los miembros que sólo son visibles a la clase; protegida (protected), que sirve para declarar a los miembros que serán visibles a la clase y sus subclases; y pública (public), que es visible a todos los clientes. La clase más generalizada en una estructura de clases se llama clase base y la que no tiene instancias (objetos) se denomina clase abstracta.

El ***tipado*** es el refuerzo de la clase de un objeto, asegura que objetos de diferentes tipos no puedan ser intercambiados o que si son intercambiados, sea de forma restringida; hay tipado fuerte y estático. En el estático, llamado ligadura estática o temprana, los tipos de todas las variables y expresiones son establecidos en el tiempo de compilación; mientras que en el fuerte, que también se conoce como ligadura dinámica o tardía, son conocidos hasta el tiempo de ejecución.

Cuando interactúan la herencia y la ligadura dinámica nace el *polimorfismo* (en los lenguajes de programación orientados a objetos se llama sobrecarga), el cual significa que un solo mensaje puede ser interpretado de diferentes formas por diversos objetos relacionados por superclases comunes. Tal vez, el polimorfismo sea la característica más importante de la orientación a objetos.

La ***concurrencia*** permite que diferentes objetos actúen al mismo tiempo, además distingue a un objeto activo de uno pasivo.

Finalmente, la ***persistencia*** es la propiedad que un objeto posee para existir más allá del tiempo y espacio de su creación.

1.2) Análisis orientado a objetos

El análisis, el diseño y la programación orientada a objetos están íntimamente relacionados; básicamente, los resultados obtenidos del análisis sirven para iniciar un diseño y los resultados de este último ayudan a construir un sistema usando programación orientada a objetos. Por este motivo, en los siguientes apartados se provee una descripción de cada uno de ellos.

El *análisis estructurado* fue desarrollado en los sesenta e introdujo un método definitivo y más manejable para el análisis de sistemas. La formalización de éste se basa en el hecho de que los subprogramas pueden servir como mecanismos de abstracción para construir sistemas grandes y complejos. El proceso más antiguo del análisis estructurado es la descomposición funcional, la cual considera a un sistema como un conjunto de áreas funcionales que se pueden dividir en procesos; los

procesos se descomponen en pasos o procedimientos comprensibles para las personas involucradas en el diseño del sistema tanto usuarios como desarrolladores (figura 1.1).

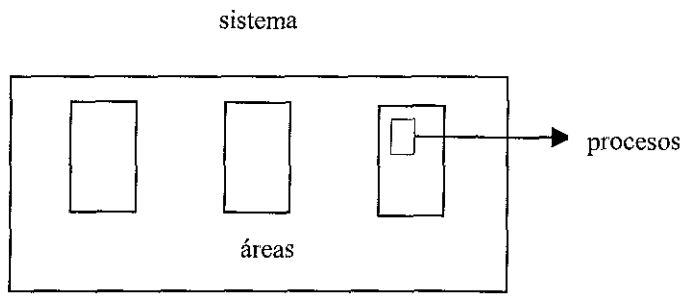


FIGURA 1.1 DESCOMPOSICIÓN FUNCIONAL

La descomposición ha alcanzado gran popularidad con autores como Yourdon, Constantine, DeMarco, etc. Al existir normas para su notación y software disponible, el método estructurado cuenta con muchos adeptos; sin embargo, tiene algunos inconvenientes, por ejemplo obliga a los programadores a concentrarse en operaciones en lugar de en las estructuras de datos, los diseños generalmente se traducen en más código y menos datos y el método de diseño no está bien ligado.

En tanto, el *modelado de información* que es el precursor más próximo del análisis de sistemas orientado a objetos, genera un diagrama entidad-relación, el cual se desarrolla listando los atributos, clasificándolos en categorías de entidades y agregando las relaciones existentes entre ellos. El modelo se perfecciona continuamente hasta que satisface los requerimientos establecidos; su desventaja es que no contempla la encapsulación de datos y no permite la herencia ni el paso de mensajes.

El ***análisis orientado a objetos*** *(AOO)* realza la definición de las características y el comportamiento de un sistema de objetos, éste se traduce en un mínimo código derivado de los datos. No enfatiza en las transformaciones de entradas en salidas, sino en los objetos; se trata de agrupar métodos cuando éstos funcionan sobre una misma abstracción de datos. El análisis debe centrarse en la identificación de objetos y la definición de clases, la organización jerarquizada de clases, la reutilización de clases y la construcción de marcos estructurales de aplicación a partir de librerías de clases. De lo anterior se desprende que la dificultad del AOO radica en encontrar el conjunto correcto de objetos; una buena selección de éstos asegurará la reutilización, promoverá la extensibilidad y ayudará a garantizar el mejoramiento en la calidad y productividad intrínsecas en el paradigma de orientación a objetos (OO). Por el contrario, una mala elección de clases de objetos tiene un profundo impacto negativo sobre el éxito de un proyecto.

El elegir los objetos adecuados representa una gran problemática, por este motivo se han propuesto dos métodos para llevar a cabo este proceso: las *tres vistas de modelación* (3VM) y el *análisis de la información basado en lingüística* (LIA). En ***3VM*** se ha encontrado que tres herramientas del análisis de sistemas tradicionales son particularmente útiles porque capturan el proceso, los datos y el control; éstas son: diagramas de flujo de datos, diagramas entidad-relación y diagramas de transición de estados.

Los *diagramas de flujo de datos* se pueden emplear porque establecen las fronteras del sistema y manifiestan una descomposición funcional del sistema propuesto en unidades primitivas.

En cuanto a los *diagramas entidad-relación*, se puede decir que éstos son los precursores del AOO; aquí, las entidades sugieren objetos y los atributos indican los datos que deben ser almacenados por los objetos. Las relaciones entre entidades proponen la creación de objetos asociativos.

Respecto a los *diagramas de transición* de estados, se sabe que ayudan a reconocer cada evento u ocurrencia del sistema y las propiedades que mantienen la información de estado.

El ***LIA*** auxilia en la identificación de componentes de objetos y sus respectivas relaciones. Se auxilia del análisis de frecuencia de frase (*PFA*) y del análisis matricial (*MA*); ambos requieren del estudio de una base de recursos (documentos importantes, modelos, software, opiniones de personas, etc.). De estos datos algunos pueden resultar irrelevantes y otros serán componentes de un modelo de análisis o diseño orientado a objetos.

3VM y LIA son predecesores del análisis orientado a objetos, porque conducen al modelo inicial de AOO; posteriormente, el modelo generado debe ser validado de acuerdo a los requerimientos del usuario.

En breve, el análisis orientado a objetos es un método de análisis que examina los requerimientos desde la perspectiva de clases y objetos encontrados en el vocabulario del dominio del problema.

1.3) Diseño orientado a objetos

El propósito del diseño es construir un sistema que:

* Satisfaga una especificación funcional dada.

* Se adecué a las limitaciones del medio.

* Encuentre los requerimientos implícitos o explícitos sobre la ejecución y fuente de uso.

* Satisfaga los criterios implícitos o explícitos de diseño.

* Satisfaga las restricciones del proceso de diseño como costo o herramientas disponibles.

El ***diseño orientado a objetos*** (DOO) es el método que nos conduce a una descomposición orientada a objetos. Al aplicar el diseño orientado a objetos se crea un software que es flexible al cambio y se tiene la garantía que está escrito con ahorro de expresiones. Se alcanza un nivel más alto de confianza en la corrección del software mediante una separación inteligente de su espacio de estados; además se reduce el riesgo de construir sistemas de software complejos. Es un desarrollo evolutivo y no revolucionario porque no rompe con los avances del pasado, al contrario, los asimila y produce un mejor método.

En DOO, el reconocer la similitud entre las cosas permite obtener las abstracciones y mecanismos clave que conducen a un diseño más simple. La identificación de las clases y los objetos es la parte más difícil del DOO porque involucra descubrimiento e inventiva; la clasificación ayuda a identificar generalizaciones, especializaciones y jerarquías agregadas entre clases; también sirve para la toma de decisiones acerca de la modularización.

Un desarrollador debe considerar lo siguiente en un DOO:

* Qué clases existen y cómo están relacionadas.

* Qué mecanismos son empleados para regular la colaboración entre objetos.

* Dónde debe ser declarada cada clase y objeto.

* En qué procesador debe colocarse cada proceso y cómo deben programarse procesos múltiples.

Para responder ésto se debe recurrir a los diagramas de clase, de objetos, modulares y de proceso. Estos cuatro diagramas forman la notación básica de un DOO, los dos primeros son parte de la vista lógica de un sistema porque describen la existencia y significado de las abstracciones clave que conforman el diseño; los últimos dos, son parte de la estructura física del sistema porque son usados para describir los componentes de software y hardware de una implantación. En DOO, la semántica dinámica de un diseño es expresada en los diagramas de transición de estados y de tiempo.

Un ***diagrama de clase*** muestra la existencia de clases y sus relaciones en el diseño lógico de un sistema. Los elementos más importantes en una estructura de clases son las clases, las relaciones entre clases y las utilerías de clase. El icono que se emplea para denotar una clase es el siguiente:

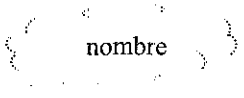
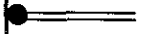


FIGURA 1.2 REPRESENTACIÓN DE UNA CLASE

Esta figura representa una abstracción con algunas fronteras bien definidas, las líneas discontinuas indican que generalmente los clientes únicamente operan sobre las instancias de una clase, no sobre la clase misma. El dibujo debe contener el nombre de la clase, el cual es único.

Las relaciones entre clases se indican con doble línea, las relaciones más comunes se especifican en la siguiente tabla:

Imagen	Uso
	Para interfaz
	Para implantación
	Instancias compatibles
	Nuevas instancias
	Herencia compatible
	Nueva herencia
	Metaclase
	Indefinida

El icono para una utilería de clase es distinguible porque está sombreado, tal como se aprecia en la figura 1.3:

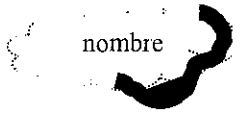


FIGURA 1.3 UTÍLERÍA DE CLASE

La utilería de clase representa un subprograma o una colección de subprogramas, requiere un nombre, el cual es escrito dentro de la figura.

Para hacer más completo un diagrama de clase debe realizarse su plantilla (template), ésta captura todos los aspectos importantes de una clase y se muestra a continuación:

Nombre	Identificador
Documentación	Texto
Visibilidad	Exportada/importada/privada
Cardinalidad	0/1/n
Jerarquía Superclase Metaclase	 Lista de nombres de clases Nombre de clase
Parámetros genéricos	Lista de parámetros
Interfaz/Implantación Usos Campos Operaciones	 Lista de nombres de clases Lista de declaraciones de campos Lista de declaraciones de operaciones
Máquina de estados finitos	Diagrama de transición de estados
Concurrencia	Secuencial/lotes/activa
Espacio de complejidad	Texto
Persistencia	Estática/dinámica

La plantilla para una utilería de clase es como la siguiente:

Nombre	Identificador
Documentación	Texto
Visibilidad	Exportada/importada/privada
Parámetros genéricos	Lista de parámetros
Interfaz/Implantación	
Usos	Lista de nombres de clases
Campos	Lista de declaraciones de campos
Operaciones	Lista de declaraciones de operaciones

Los ***diagramas de transición de estados*** muestran el espacio de estados de una clase, los eventos que causan el cambio de un estado a otro y las acciones que resultan de este cambio. Un estado se representa mediante un círculo, el nombre del estado (debe ser único) se escribe dentro del círculo. El estado de inicio se identifica rápidamente porque las líneas dobles no están rellenas, lo contrario es para un estado final (ver figura 1.4).

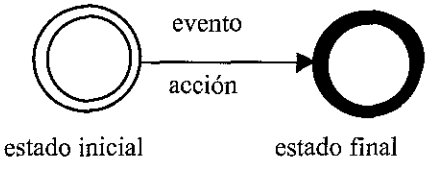


FIGURA 1.4 DIAGRAMA DE ESTADOS

Cada línea debe ser etiquetada con el nombre de al menos un evento que ocasiona la transición y debe escribírsele el nombre de la acción resultante. Los nombres de las acciones, resultados y acciones no necesariamente son únicos dentro de un diagrama de transición de estados, porque el mismo evento puede provocar

transiciones a diferentes estados y la misma acción puede resultar de diferentes transiciones.

Los ***diagramas de objetos*** se emplean para señalar la existencia de objetos y sus relaciones en el diseño lógico del sistema, su propósito es ilustrar la semántica de los mecanismos clave del diseño lógico. Se utilizan para capturar la semántica dinámica de las operaciones y la máquina de estados finitos. Cada objeto indica algunas instancias de una clase. El esquema para un objeto se observa en la figura 1.5.

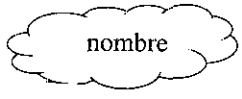


FIGURA 1.5 ESQUEMA DE UN OBJETO

El nombre del objeto debe escribirse dentro del icono, éste no es necesariamente único. Las propiedades (concurrencia, persistencia, etc.) deben aparecer en la esquina inferior izquierda del objeto. Una relación entre dos objetos significa simplemente que pueden mandarse mensajes el uno al otro. Éstos también tienen su plantilla, la cual documenta la clase del objeto, qué operaciones pueden ejecutar sobre él los clientes, la persistencia, tal como se aprecia en la siguiente tabla:

Nombre	Identificador
Documentación	Texto
Clase	Nombre de la clase
Persistencia	Persistente/estático/dinámico

Los ***diagramas de tiempo*** son gráficas que colocan al tiempo en el eje X y los objetos en el eje Y, el tiempo debe ser expresado en unidades absolutas o relativas e incrementarse a la derecha (ver figura 1.6).

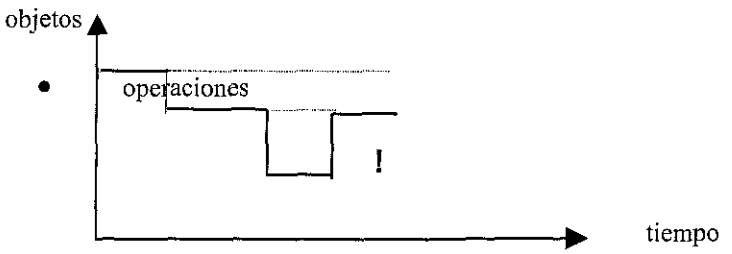


FIGURA 1.6 DIAGRAMA DE TIEMPO

Las líneas discontinuas indican el anidamiento dinámico de los mensajes. El • señala cuando es creado el objeto y el !, cuando es destruido.

Los ***diagramas modulares*** son usados para exhibir la localización de las clases y los objetos dentro de los módulos en el diseño físico del sistema, sus elementos más importantes son los módulos y la visibilidad de módulos. Los módulos tienen un nombre, se escribe en la parte superior del icono y es único. Un módulo sin sombreado representa un módulo privado a su subsistema; uno sombreado, que es exportado a un subsistema; y uno subrayado, que es importado de otro subsistema. En la figura 1.7 se observa el diagrama para un módulo.

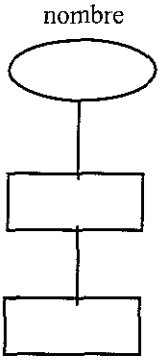


FIGURA 1.7 DIAGRAMA MODULAR

La visibilidad entre módulos es indicada a través de líneas directas. La documentación más importante asociada con un módulo debe contener una lista que incluya clases, utilerías de clases, objetos y otras declaraciones.

Los ***diagramas de proceso*** sirven para visualizar y razonar acerca del problema de asignación de procesos a procesadores en el diseño físico del sistema, es útil cuando la instalación involucra varios dispositivos u objetos activos. La figura 1.8 muestra un diagrama de proceso.

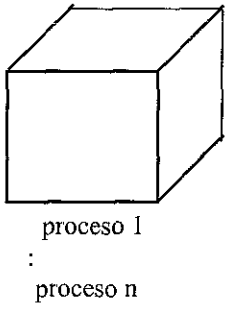


FIGURA 1.8 DIAGRAMA DE PROCESO

1.4) Programación orientada a objetos

En la primera y en el inicio de la segunda generación de los lenguajes de programación, los bloques constructores físicos básicos de todas las aplicaciones eran los subprogramas (o párrafos). Las aplicaciones escritas en estos lenguajes exhibían una estructura física relativamente plana, la cual consistía sólo de datos globales y subprogramas; ésto se aprecia en la figura 1.9.

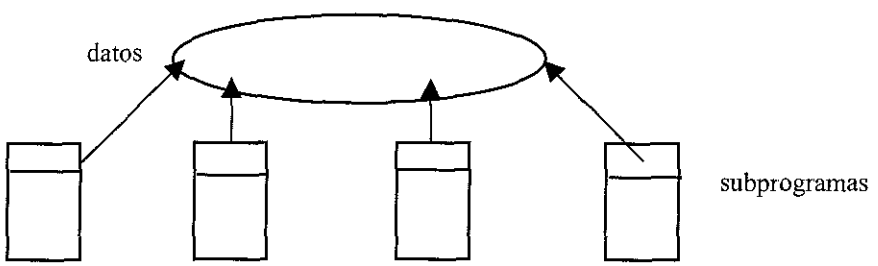


FIGURA 1.9 TOPOLOGÍA DE LA PRIMERA GENERACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN

Un error en cualquier parte del programa tenía efectos devastadores en todo el resto del programa, porque las estructuras globales de datos estaban expuestas a todos los subprogramas. Cuando se hacían modificaciones a lo largo del sistema, era muy difícil mantener la integridad del diseño original.

En la segunda y al principio de la tercera generación de los lenguajes de programación, los subprogramas fueron apreciados como un medio para abstraer funciones del programa. Los lenguajes soportaban una variedad de mecanismos de paso de parámetros, se establecieron los fundamentos de la programación estructurada (se manifestó en el anidamiento de subprogramas y el desarrollo de teorías de estructuras de control para el alcance y visibilidad de declaraciones) y emergieron métodos de diseño estructurado que ofrecían guías a los diseñadores para

construir grandes sistemas utilizando subprogramas como bloques constructores básicos (figura 1.10).

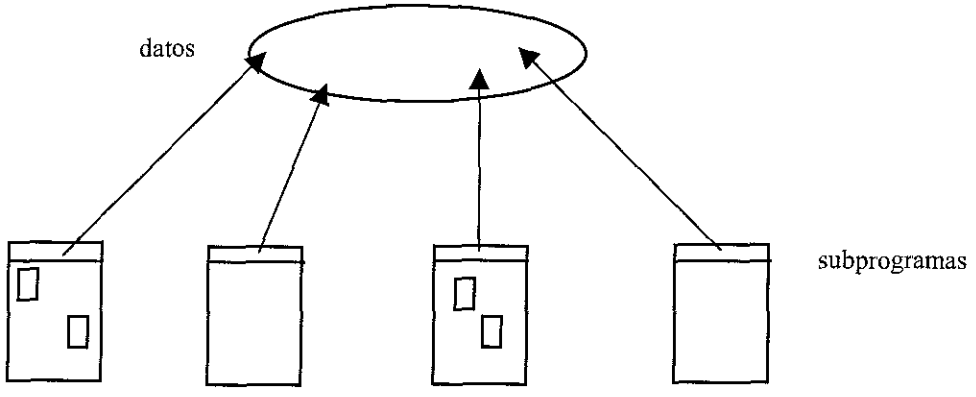


FIGURA 1.10 TOPOLOGÍA DE LA SEGUNDA GENERACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN

En la tercera generación de los lenguajes de programación, la mayoría de los lenguajes soportaban la estructura modular y tenían unas cuantas reglas acerca de la semántica requerida para mantener la consistencia entre interfaces modulares, tal como se aprecia en la figura 1.11. Desafortunadamente, debido a que éstos poseían baja abstracción de datos y fuerte tipado, los errores eran detectados únicamente durante la ejecución del programa.

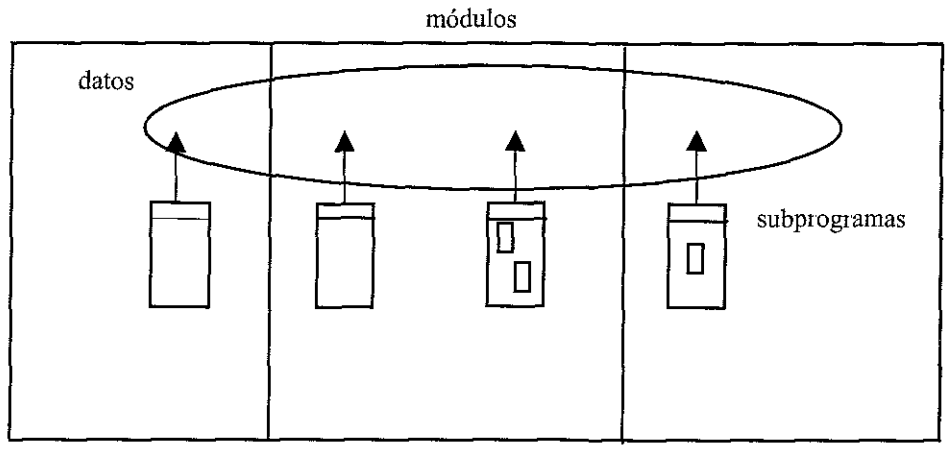


FIGURA 1.11 TOPOLOGÍA DE LA TERCERA GENERACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN

En la cuarta generación nacen los lenguajes de programación orientados a objetos, entre los más populares se encuentran Smalltalk, Eiffel, Pascal orientado a objetos, CLOS y C++.

En la programación orientada a objetos (POO), el programar ya no significa solamente escribir líneas de código sino desarrollar modelos empleando clases; gracias a ella, los programas tienen menos líneas de código, menos sentencias de bifurcación y módulos más comprensibles porque reflejan la relación unívoca entre el modelo del objeto y el conceptual. Además, proveen una sintaxis para guiar el paradigma y soportar los mecanismos de herencia y ligadura dinámica que permiten volver a utilizar las clases y las bibliotecas de clases.

Otra característica fundamental de los lenguajes orientados a objetos es la herencia; sin ella, la adición de un nuevo tipo de objeto requeriría escribir completamente procedimientos para operaciones comunes. También permiten la creación de una jerarquía de objetos con nombres de métodos comunes para operaciones que son conceptualmente similares, pero están realizadas de manera

diferente para cada una de las clases de la jerarquía. Como consecuencia, cuando diferentes objetos reciban el mismo mensaje, cada uno actuará de manera distinta (polimorfismo).

En los lenguajes orientados a objetos, la ligadura asocia un mensaje con el método para efectuar dicho mensaje, que puede ser en el proceso de compilación y enlazado o en la ejecución.

Las dos herramientas más destacadas para los lenguajes orientados a objetos son los examinadores (browser) y los depuradores simbólicos. Un examinador de código fuente suministra una visión del código de la aplicación y de la librería de clases; básicamente es una ayuda de navegación porque permite ver la estructura general jerárquica de la aplicación y moverse entre clases y métodos.

Un depurador simbólico lleva a cabo el seguimiento de las llamadas a los métodos, ayuda a fijar puntos de ruptura dentro de los métodos o en la invocación de un método y está capacitado para examinar y alterar los valores de las variables modelo.

La POO es la construcción de subclases, con ello se logra modelar los conceptos más generales del problema; los casos especiales se manejan por medio de subclases más específicas. Las subclases perfeccionan a las clases existentes agregando y/o modificando variables modelo y métodos.

Cabe aclarar que los lenguajes y técnicas de programación orientadas a objetos no eliminan la necesidad de mantenimiento de sistemas en funcionamiento, mas pueden disminuir la severidad y el gasto del proceso de mantenimiento porque el programador invertirá menos tiempo en comprender el comportamiento de cada una

de las clases y sólo examinará los mensajes entre objetos para identificar de donde provino el problema.

A pesar de todas las virtudes que se le han atribuido a este tipo de programación, se siguen presentando algunos problemas. La causa de éstos, es que los programadores aún sin entenderla bien se han atrevido a utilizarla en ambientes de producción. El hecho de que se escriba un programa que emplea herencia, polimorfismo, plantillas, constructores y destructores no garantiza un programa maravilloso OO revestido con todos los beneficios de tal.

Para comprender la POO, el programador debe empezar por ver el código como objetos y las funciones que operan sobre él. Aprender POO y aplicarla a algo no familiar hará que la transición de programar estructuradamente a hacerlo con objetos sea más difícil. Sin embargo, con el entendimiento adecuado y dando tiempo, el cambio a las técnicas OO resultará un paso natural.

CAPÍTULO 2. MÉTODO DEL ELEMENTO FINITO (MEF)

CAPÍTULO 2. MÉTODO DEL ELEMENTO FINITO (MEF)

En la mayoría de los problemas de ingeniería no es posible encontrar soluciones matemáticas analíticas. Una solución analítica es una expresión matemática que proporciona los valores deseados de una cantidad desconocida en una localidad de un cuerpo y como consecuencia es válida para un número infinito de localidades del cuerpo.

Para problemas que involucran propiedades de materiales y condiciones de frontera complejas, los recursos del ingeniero son el empleo de los métodos numéricos; el más común es el esquema general de diferencias finitas. El modelo familiar de diferencias finitas ofrece una buena aproximación a las ecuaciones gobernantes de un problema. Con éste se pueden tratar problemas complicados, pero cuando se encuentran geometrías irregulares o especificaciones inusuales de condiciones de frontera, este método es bastante difícil de utilizar. Es entonces cuando otra técnica numérica más reciente entra en acción: el Método del Elemento Finito, el cual es descrito a continuación; cabe aclarar que las ecuaciones aquí presentadas se obtuvieron de diversas fuentes, principalmente de los autores Zienkiewicz y Abel y después fueron adaptadas para el problema particular aquí estudiado.

2.1) Generalidades

El Método del Elemento Finito (MEF) es un procedimiento numérico para obtener soluciones a muchos de los problemas encontrados en análisis en ingeniería; tiene dos subdivisiones primarias: la primera utiliza elementos discretos para obtener los desplazamientos en las uniones y las fuerzas en los miembros de un marco

estructural, en tanto que la segunda utiliza elementos continuos para obtener soluciones aproximadas a problemas de mecánica de sólidos, mecánica de fluidos y transferencia de calor (entre otros). La formulación que usa los elementos discretos se conoce como análisis matricial de estructuras y sus resultados son idénticos a los provistos por el análisis clásico de marcos estructurales. El segundo enfoque es el verdadero MEF, en éste se obtienen valores aproximados de los parámetros deseados en puntos específicos llamados nodos. Un programa de computadora de MEF de uso general debe ser capaz de resolver ambos tipos de problemas.

El MEF combina varios conceptos matemáticos para producir un sistema de ecuaciones lineales o no lineales, generalmente el número de ecuaciones es muy grande, pueden ser veinte o veinte mil o más, por lo tanto se requiere una computadora para solucionarlas, es decir, el método tiene poco valor práctico si se carece de una computadora.

Es imposible documentar el origen exacto del MEF ya que sus conceptos básicos se han estado utilizando desde hace más de ciento cincuenta años. El método como lo conocemos actualmente, es la extensión y desarrollo de varios artículos publicados en los años cincuenta, los cuales abarcaron el análisis matricial de estructuras a cuerpos en el continuo (es el cuerpo físico, estructura o sólido a ser analizado). Las exploraciones hechas en los sesenta colocaron al método sobre bases matemáticas firmes, estimularon el desarrollo de programas de computadora de propósito múltiple que implantaron el método y las áreas de aplicación principales fueron el diseño de aviones, misiles, cápsulas espaciales y algunas otras similares.

Cabe aclarar que el MEF no está restringido al estudio de la mecánica, como técnica de análisis sirve para examinar un amplio rango de estructuras en ingeniería y componentes, desde el cuerpo humano hasta las alas de un aeroplano.

Aunque el origen del método es vago, sus ventajas son claras: es fácilmente aplicable a objetos de forma irregular compuestos de varios materiales y con condiciones de frontera mixtas. Es aplicable a problemas de estado estable y dependientes del tiempo, así como problemas que involucran propiedades no lineales del material.

El MEF es la base computacional de muchos programas de CAD, el aumento en la utilización de programas de CAD hace imperativo que el ingeniero tenga un buen conocimiento de cómo trabaja el MEF.

Los pasos a seguir en el proceso de análisis de MEF son:

a) *Discretización del continuo*: El primer paso es dividir el continuo o región en elementos. Una amplia variedad de formas de elementos pueden utilizarse y teniendo cuidado se pueden emplear diferentes tipos de elementos en la misma región de solución. Pese a que se han hecho esfuerzos para automatizar el proceso de subdivisión, éste es a juicio del ingeniero; él debe decidir qué número, tamaño y arreglo de elementos finitos será ocupado.

b) *Selección de las funciones de interpolación*: El siguiente paso es asignar nodos a cada elemento y elegir el tipo de función de interpolación que representará la variación de la variable de campo sobre este elemento. A menudo se seleccionan polinomios como funciones de interpolación para la variable de campo, ya que son más fáciles de integrar y diferenciar. El grado del polinomio elegido depende del número de nodos asignados al elemento, la naturaleza y número de incógnitas en

cada nodo, así como de ciertos requerimientos de continuidad impuestos a los nodos a lo largo de los elementos de frontera.

c) Encontrar las propiedades de los elementos: El paso siguiente es establecer las ecuaciones matriciales expresando las propiedades de los elementos individuales. Básicamente hay cuatro enfoques diferentes para determinar las propiedades de los elementos. El primero es llamado el *enfoque directo*, porque su origen es el método directo de rigidez del análisis estructural. Éste sugiere la necesidad de usar el álgebra matricial para tratar las ecuaciones del elemento finito, además es el más fácil de entender cuando se trabaja con el MEF por primera vez. Pero su bondad principal radica en que es más fácil de programar en una computadora.

El segundo es el *enfoque variacional*, él es más dinámico y avanzado porque toma ventaja del cálculo de variaciones. El conocimiento del enfoque variacional es necesario para trabajar más allá del nivel introductorio y extender el MEF a una amplia variedad de problemas sofisticados de ingeniería.

El tercero es el *enfoque de pesos residuales*, el cual comienza con las ecuaciones gobernantes del problema y procede sin considerar las proposiciones variacionales; tiene la ventaja de poder extender el MEF a problemas donde el cálculo de funciones no esté disponible.

El cuarto enfoque se apoya en el *balance de la energía térmica y/o mecánica* de un sistema. No requiere proposiciones variacionales, por lo tanto incrementa considerablemente el rango de posibles aplicaciones del MEF.

d) *Ensamblar las propiedades de los elementos para obtener las ecuaciones del sistema*: Para encontrar las propiedades del sistema total modelado

por la red de elementos, se deben ensamblar todas las propiedades de los elementos; es decir, se deben combinar las ecuaciones matriciales que expresan el comportamiento de los elementos y formar las ecuaciones matriciales que manifiestan el comportamiento de la región solución total o sistema. Las ecuaciones matriciales del sistema tienen la misma forma que las ecuaciones para un elemento individual, excepto que contienen muchos más términos porque incluyen a todos los nodos.

Aquí hay que tomar en cuenta las condiciones de frontera del problema con el fin de que las ecuaciones del sistema estén listas para su solución.

e) Resolver las ecuaciones del sistema: El proceso de ensamble del paso anterior genera un conjunto de ecuaciones algebraicas simultáneas, éstas deben resolverse para obtener los valores nodales desconocidos de la variable de campo. Si las ecuaciones son lineales se puede utilizar cualquier técnica gaussiana de solución y si son no lineales, su solución es más difícil y deben utilizarse métodos alternativos.

f) Cálculos adicionales: Algunas veces se necesita emplear la solución de las ecuaciones del sistema para calcular otros parámetros importantes.

Dependiendo de la naturaleza del problema a resolver, las aplicaciones del MEF se pueden clasificar en tres categorías. En la primera están todos los problemas conocidos como *problemas de equilibrio* o *problemas independientes del tiempo*; la mayoría de las aplicaciones del MEF caen dentro de esta categoría.

Para la solución de problemas del área de mecánica de sólidos, se necesita encontrar la distribución de desplazamientos, esfuerzos o temperaturas en una carga térmica o mecánica dada; en tanto, en mecánica de fluidos, se debe calcular la presión, velocidad, temperatura y algunas veces la concentración de distribuciones bajo condiciones de estado estable.

El segundo grupo corresponde a los llamados *problemas de eigenvalores* de mecánica sólida o de fluidos. Estos son problemas de estado estable cuya solución requiere la determinación de frecuencias naturales y modos de vibración de sólidos y fluidos. Aquí es posible estudiar la interacción de lagos y presas o el comportamiento de combustibles líquidos en tanques flexibles.

Otra clase de problemas de eigenvalores incluye la estabilidad de estructuras y la estabilidad de flujos laminares.

En el tercer género se hallan los *problemas de propagación* o *dependientes del tiempo* de mecánica del medio continuo. Esta categoría está compuesta de los problemas que resultan cuando la dimensión tiempo es adicionada a los problemas de las dos categorías anteriores.

2.2) Ecuaciones gobernantes del MEF

La solución de problemas en mecánica de sólidos, así como placas y estructuras se puede tratar de varias formas. El enfoque clásico es formular la ecuación diferencial gobernante y obtener la solución analítica. Ésto no funciona para muchos problemas debido a la dificultad para describir matemáticamente la geometría estructural y/o las condiciones de frontera. Una alternativa popular al enfoque clásico es un procedimiento numérico basado en el principio que establece que los desplazamientos en la posición de equilibrio ocurren de tal forma que la energía potencial de un sistema estable es un valor mínimo.

Un término que contribuye a la energía potencial es la energía de deformación, ésta es la energía almacenada durante el proceso de deformación. La energía de deformación es una integral de volumen que comprende los productos de los componentes de esfuerzo y deformación. Por ejemplo, la energía de deformación en un miembro con fuerza axial es

$$A = \int_v \frac{\sigma_{xx}\varepsilon_{xx}}{2} dv$$

Lo importante de todo ésto, es que el análisis de los desplazamientos en mecánica estructural y de sólidos combina la integral de energía de deformación en un proceso de minimización. Computacionalmente, el análisis de una placa o armadura es muy similar al enfoque variacional o de Galerkin.

El MEF es un procedimiento numérico para resolver problemas físicos gobernados por una ecuación diferencial o un teorema de energía. Tiene dos características que lo distinguen de otros métodos numéricos: 1) emplea una formulación integral para generar un sistema de ecuaciones algebraicas y 2) usa funciones suavizadas continuas para la aproximación de las cantidades desconocidas.

La segunda de estas características es lo que distingue al MEF de los otros procedimientos numéricos que emplean una formulación integral. El MEF utiliza una función continua, pero una función con sólo bastante continuidad en las derivadas para permitir que las integrales sean evaluadas. Para una formulación integral no se requiere continuidad en la primera derivada. La integral puede evaluarse cuando la primera derivada recae en el medio continuo. Una ecuación compuesta de varios segmentos lineales puede emplearse como la ecuación de aproximación.

Un modelo de elemento finito para el problema de deflexión de una viga se podría parecer al de la figura 2.1.

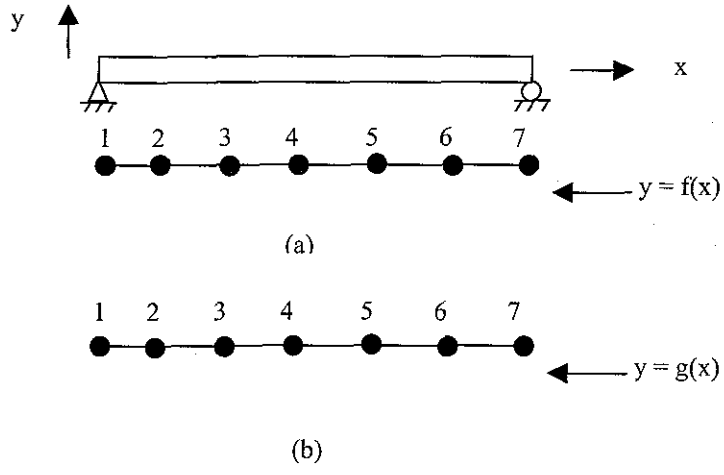


FIGURA 2.1 PROBLEMA DE DEFLEXIÓN DE UNA VIGA

Consistiría de varios segmentos lineales definidos en términos de los valores nodales, como se ve en la parte a. El intervalo entre cada nodo sería considerado un elemento y la deflexión se aproxima por segmentos de líneas rectas. Una malla alternativa consistiría de tres elementos cada uno definido por tres puntos nodales, como se ve en b. En este caso se define una ecuación cuadrática sobre cada conjunto de tres puntos. En cualquier caso, las ecuaciones y = f(x) o y = g(x) no tendrían una primera derivada continua entre algún par de elementos adyacentes.

Las funciones sin términos de primera derivada continua, también se pueden utilizar con el método de Galerkin. El término en segunda derivada d^2y/dx^2 se modifica empleando integración por partes.

Ahora se discutirá la división de una región unidimensional en elementos lineales y se desarrollará la ecuación del elemento. Después, esta ecuación se

generalizará de tal forma que una ecuación en el medio continuo sea obtenida para

$$D\frac{d^2\phi}{dx^2} + Q = 0 \qquad (1)$$

toda la región. El elemento lineal se utiliza para conseguir una solución aproximada a

Este elemento, también se emplea para calcular el desplazamiento en un sistema de miembros con fuerzas axiales.

La región unidimensional es un segmento de línea y la subdivisión en subregiones o elementos es muy sencilla; el segmento de línea se divide en pequeños fragmentos utilizando nodos, los cuales normalmente se numeran consecutivamente de izquierda a derecha del elemento y los números de los elementos se encierran entre paréntesis para distinguirlos de los nodos, tal como se muestra en la figura 2.2

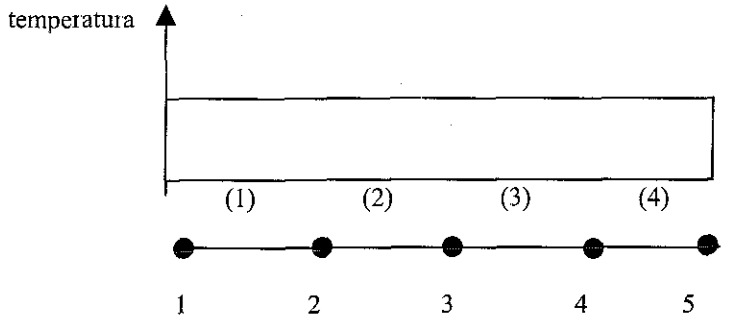


FIGURA 2.2 NUMERACIÓN DE LOS NODOS Y DE LOS ELEMENTOS

Hay algunas reglas para la colocación de los nodos cuando se obtiene una solución aproximada a una ecuación:

1. Colocar los nodos lo más cercano posible, en las regiones donde los parámetros desconocidos cambian rápidamente y también donde las incógnitas son relativamente constantes.

2. Colocar un nodo donde exista un cambio en el valor de los coeficientes de D y Q en la ecuación 1.

3. Colocar un nodo donde se necesita un valor numérico para ϕ en la expresión 1.

La regla 1 requiere que el usuario tenga algún conocimiento de cómo varían los parámetros desconocidos, precisamente es aquí donde se observa la experiencia del ingeniero; la segunda es necesaria porque es indispensable evaluar las integrales que incluyen los parámetros D y Q en 1, las integrales son más fáciles de evaluar si los coeficientes no experimentan un cambio brusco dentro del intervalo de integración.

El elemento lineal unidimensional es un segmento de línea de longitud L y dos nodos, uno en cada extremo (ver figura 2.3).

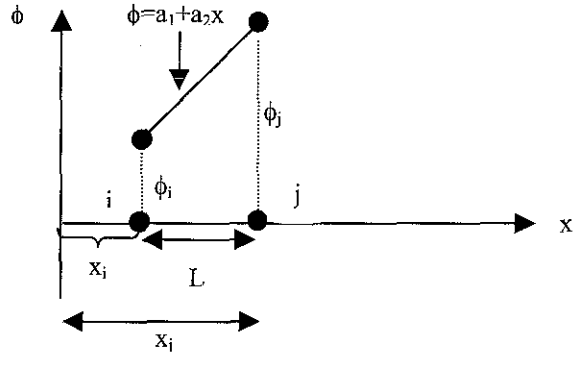


FIGURA 2.3 ELEMENTO LINEAL UNIDIMENSIONAL

Los nodos se denotan por i y j, y los valores nodales por ϕ_i yϕ_j. El origen del sistema coordenado está a la izquierda del nodo i. El parámetro ϕ varía linealmente entre los nodos y la ecuación de ϕ es

$$\phi = a_1 + a_2 x \qquad (2)$$

Los coeficientes a_1 y a_2 se determinan utilizando las condiciones nodales:

$$\phi = \phi_i \text{ en } x = x_i \text{ y } \phi = \phi_j \text{ en } x = x_j \qquad (3)$$

Para desarrollar el par de ecuaciones

$$\phi_i = a_1 + a_2 x_i \quad y \quad \phi_j = a_1 + a_2 x_j \qquad (4)$$

de donde se obtiene

$$a_1 = \frac{\phi_i x_j - \phi_j x_i}{x_j - x_i} \quad y \quad a_2 = \frac{\phi_j - \phi_i}{x_j - x_i} \qquad (5)$$

Sustituyendo 5 en 2 y reacomodando se obtiene

$$\phi = \left(\frac{x_j - x}{L}\right)\phi_i + \left(\frac{x - x_i}{L}\right)\phi_j \qquad (6)$$

donde $x_j - x_i$ fue reemplazado por L.

La ecuación 6 está en la forma normal de elemento finito. Los valores nodales son multiplicados por funciones lineales de x, denominadas funciones de forma o funciones de interpolación. Éstas se denotan por N con un subíndice para indicar el nodo con el cual está asociada la función de forma específica. Las funciones de forma en 6 se denotan por N_i y N_j con

$$N_i = \frac{x_j - x}{L} \quad y \quad N_j = \frac{x - x_i}{L} \qquad (7)$$

La ecuación 6 puede reescribirse como

$$\phi = N_i\phi_i + N_j\phi_j \qquad (8)$$

y también como

$$\phi = [N]\{\phi\} \qquad (9)$$

donde $[N] = [N_i\ N_j]$ es un vector renglón de funciones de forma y $\{\phi\} = \begin{Bmatrix} \phi_i \\ \phi_j \end{Bmatrix}$ es un vector columna que contiene a los valores nodales del elemento.

Cada función de forma tiene un valor de uno en su propio nodo y cero en el otro nodo y además las dos funciones de forma suman uno. Generalmente las funciones de forma son polinomios del mismo tipo que la ecuación de interpolación original. La ecuación 2 es una ecuación lineal y las funciones de forma son lineales. Si la ecuación de interpolación hubiera sido un modelo cuántico definido por tres nodos, las funciones de forma resultantes hubieran sido ecuaciones cuánticas. Otra característica es que las derivadas de la función de forma con respecto a x suman cero. Las funciones de forma se muestran en la figura 2.4.

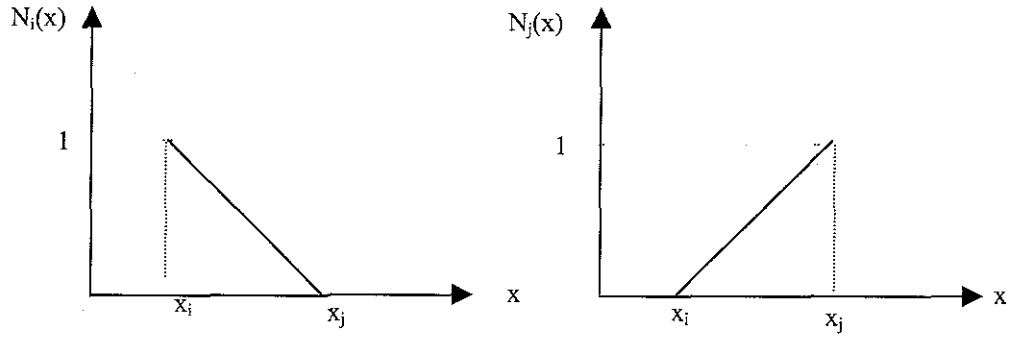


FIGURA 2.4 FUNCIONES DE FORMA

Se ha utilizado un elemento lineal unidimensional para aproximar la distribución de temperatura en un alambre. La solución indica que las temperaturas en los nodos i y j son 120° C y 90° C respectivamente. Determinar la temperatura en un punto a 4 cm del origen y el gradiente de temperatura dentro del elemento. Los nodos i y j están a 1.5 y 6 cm del origen.

La temperatura ϕ dentro del elemento está dada por (6), al sustituir los valores correspondientes se obtiene

$$\phi = \left(\frac{6-4}{4.5}\right)120 + \left(\frac{4-1.5}{4.5}\right)90 = 103.3°C$$

El gradiente de temperatura es la derivada de la expresión 6 y sustituyendo los valores nodales se produce

$$\frac{d\phi}{dx} = \frac{\phi_j - \phi_i}{L} \qquad (10)$$

$$\frac{d\phi}{dx}\left(\frac{90-120}{4.5}\right) = -6.67°C/cm$$

Una ecuación para una pieza continua para una región unidimensional se puede construir conectando varias ecuaciones lineales con las mismas propiedades que las desarrolladas anteriormente. Cada una de estas ecuaciones se puede escribir como

$$\phi^{(e)} = N_i^{(e)}\phi_i + N_j^{(e)}\phi_j \qquad (11)$$

donde

$$N_i^{(e)} = \frac{x_j - x}{x_j - x_i} \quad y \quad N_j^{(e)} = \frac{x - x_i}{x_j - x_i} \qquad (12)$$

El superíndice (e) indica un elemento, todo lo que se necesita para completar el proceso es colocar los valores correspondientes de i, j y e para cada elemento. Los valores de i y j para una e correspondiente se obtienen de la malla. El nodo i es el que está a la izquierda del elemento.

La función de forma anterior se puede utilizar para resolver problemas en mecánica de sólidos o en ecuaciones diferenciales. Ahora se desarrollará una solución aproximada para la ecuación diferencial unidimensional

$$D\frac{d^2\phi}{dx^2} + Q = 0 \qquad (1)$$

con condiciones de frontera

$$\phi(0) = \phi_0 \quad y \quad \phi(H) = \phi_H \qquad (2)$$

Dos tipos de problemas físicos se pueden solucionar con (1): la deflexión de vigas simplemente apoyadas cuando se conoce el diagrama de momento flexionante y

el flujo de calor a través de una pared compuesta cuando se conocen las temperaturas de la superficie.

Las ecuaciones del elemento finito se obtienen empleando la formulación de Galerkin. La evaluación de la integral residual conduce a una ecuación nodal que se aplica de forma recursiva para generar un sistema lineal de ecuaciones. Se puede generar un sistema de ecuaciones lineales al evaluar la integral de pesos residuales

$$-\int_0^H w(x)(D\frac{d^2\phi}{dx^2}+Q)dx=0 \qquad (3)$$

El signo negativo sólo es para que los resultados se escriban en una forma más conveniente.

La formulación de Galerkin del método de pesos residuales requiere que las funciones de peso se construyan empleando las funciones de forma N_i y N_j. Dichas funciones se definen como sigue: "las funciones de peso para el s-ésimo nodo w_s, consisten de las funciones de forma asociadas con el s-ésimo nodo". La función de peso para una malla lineal de 3 nodos (figura 2.5) está constituida por las funciones de forma para tres nodos.

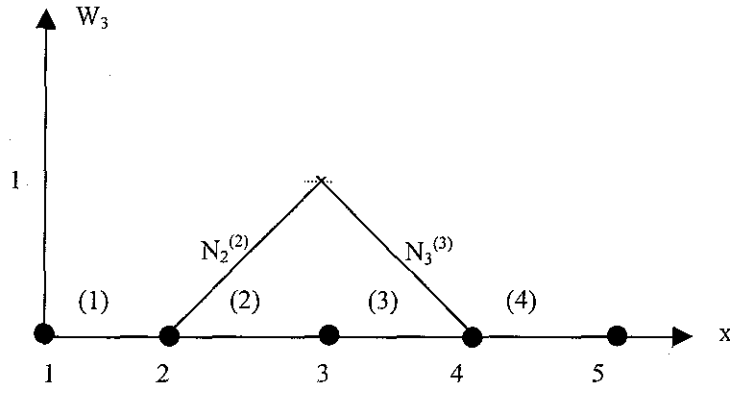


FIGURA 2.5 MALLA LINEAL DE TRES NODOS

$$w_3(x) = \begin{cases} N_3^{(2)} & x_2 \le x \le x_3 \\ N_3^{(3)} & x_3 \le x \le x_4 \end{cases} \qquad (4)$$

y en general

$$w_s(e) = \begin{cases} N_s^{(e)} & x_r \le x \le x_s \\ N_s^{(e+1)} & x_s \le x \le x_t \end{cases} \qquad (5)$$

A esta función a veces se le llama función sombrero por la forma que adopta; así, la función de peso para cada nodo consiste tanto de N_i, N_j o una combinación de los dos. Una vez que se han evaluado las funciones de peso, lo siguiente es estimar la integral residual 3; recurriendo a la secuencia de nodos r, s y t resulta que es

$$R_s = R_s^{(e)} + R_s^{(e+1)} = -\int_{x_r}^{x_s} \left[N_s\left(D\frac{d^2\phi}{dx^2}\right) + Q\right]^{(e)} dx - \int_{x_s}^{x_t} \left[N_s\left(D\frac{d2\phi}{dx2}\right) + Q\right]^{(e+1)} dx = 0 \qquad (6)$$

debido a que $w_s = 0$ para $x < x_r$ y $x > x_t$. La integral es en dos partes porque $w_s^{(x)}$ está definida por dos ecuaciones separadas dentro del intervalo $x_r \le x \le x_t$. Los términos $R_s^{(e)}$ y $R_s^{(e+1)}$ representan las contribuciones de los elementos (e) y (e+1) a la ecuación residual para el nodo.

Sin embargo, existe un problema en cada una de las integrales de 6 y es que la solución aproximada no tiene continuidad en la primera derivada $d\phi/dx$; además la integral de $d^2\phi/dx^2$ no está definida. Ésto puede resolverse cambiando $d^2\phi/dx^2$ en un nuevo término, por ejemplo considerando la primera integral en 6 y notando que

$$\frac{d}{dx}(N_s \frac{d\phi}{dx}) = N_s \frac{d^2\phi}{dx^2} + \frac{dN_s}{dx}\frac{d\phi}{dx} \qquad (7)$$

entonces

$$N_s \frac{d^2\phi}{dx^2} = \frac{d}{dx}\left(\frac{N_s d\phi}{dx}\right) - \frac{dN_s}{dx}\frac{d\phi}{dx} \qquad (8)$$

y sustituyendo en las integrales, queda

$$-\int_{x_r}^{x_s}\left[N_s(D\frac{d^2\phi}{dx^2})\right]^{(e)}dx = -(DN_s\frac{d\phi}{dx})^{(e)}\Big|_{x_r}^{x_s} + \int_{x_r}^{x_s}\left[D\frac{dN_s}{dx}\frac{d\phi}{dx}\right]^{(e)}dx \qquad (9)$$

Operaciones similares aplicadas al primer término de la segunda integral en la expresión 6 producen

$$-\int_{x_s}^{x_t}\left[N_s(D\frac{d^2\phi}{dx^2})\right]^{(e+1)}dx = -(DN_s\frac{d\phi}{dx})^{(e+1)}\Big|_{x_s}^{x_s} + \int_{x_s}^{x_t}\left[D\frac{dN_s}{dx}\frac{d\phi}{dx}\right]^{(e+1)}dx \qquad (10)$$

Los primeros términos, tanto en 9 como en 10, se simplifican debido a que las funciones de forma son cero o uno en los nodos respectivos. La ecuación residual completa es

$$R_s = R_s^{(e)} + R_s^{(e+1)} = -\int_0^{xt} \left[w_s \left(D \frac{d^2\phi}{dx^2} + Q \right) \right] dx = -\left(D \frac{d\phi}{dx} \right)^{(e)} \Big|_{x = x_s} + \int_{x_r}^{xs} \left[D \frac{dN_s}{dx} \frac{d\phi}{dx} - N_s Q \right]^{(e)} dx$$

$$+ \quad \left(D \frac{d\phi}{dx} \right)^{(e+1)} \Big|_{x = x_s} + \int_{xs}^{xt} \left[D \frac{dN_s}{dx} \frac{d\phi}{dx} - N_s Q \right]^{(e+1)} dx = 0 \qquad (11)$$

El par de términos evaluados en x = x_s establecen que debe existir un requerimiento entre elementos: el residual no puede ser cero hasta que la diferencia entre las dos cantidades sea cero.

Al estimar las integrales en 11, se obtiene la ecuación residual para un nodo interior. Comenzando con el elemento (e): $\phi^{(e)} = N_r\phi_r + N_s\phi_s$ y ahora

$$\phi^{(e)} = \left(\frac{x_s - x}{L} \right) \phi_r + \left(\frac{x - x_r}{L} \right) \phi_s \qquad (12)$$

Así

$$N_s^{(e)} = \frac{x - x_r}{L}, \quad \frac{dN_s^{(e)}}{dx} = \frac{1}{L} \qquad (13) \quad y$$

$$\frac{d\phi^{(e)}}{dx} = \frac{1}{L}(-\phi_r + \phi_s) \qquad (14)$$

Sustituyendo los términos apropiados y evaluando las integrales, se llega a

$$\int_{x_r}^{xs} \left[D \frac{dN_s}{dx} \frac{d\phi}{dx} \right] dx = \frac{D}{L}(-\phi_r + \phi_s) \qquad (15) \quad y \quad \int_{x_r}^{xs} [QN_s] dx = \frac{QL}{2} \qquad (16)$$

Al combinar las expresiones 15 y 16 con la contribución entre elementos para el elemento (e), resulta

$$R_s^{(e)} = -\left(D \frac{d\phi}{dx} \right)^{(e)} \Big|_{x = x_s} + \frac{D}{L}(-\phi_r + \phi_s) - \frac{QL}{2} \qquad (17)$$

Procediendo con la segunda integral de (12), $\phi^{(e+1)} = N_s\phi_s + N_t\phi_t$, se tiene

$$\phi^{(e+1)} = \left(\frac{x_t - x}{L}\right)\phi_s + \left(\frac{x - x_s}{L}\right)\phi_t \qquad (18)$$

Así que

$$N_s^{(e+1)} = \frac{x_t - x}{L}, \quad \frac{dN_s^{(e+1)}}{dx} = -\frac{1}{L} \qquad (19) \quad y \quad \frac{d\phi^{(e+1)}}{dx} = \frac{1}{L}(-\phi_s + \phi_t) \qquad (20)$$

Al valuar las integrales se obtiene

$$\int_{x_s}^{x_t} D\,\frac{dN_s}{dx}\frac{d\phi}{dx}\,dx = -\frac{D}{L}(\phi_s - \phi_t) \qquad (21), \quad \int_{x_s}^{x_t} QN_s\,dx = \frac{QL}{2} \qquad (22)$$

$$y \quad R_s^{(e+1)} = D\frac{d\phi}{dx}\bigg|_{x = x_s} + \frac{D}{L}(\phi_s - \phi_t) - \frac{QL}{2} \qquad (23)$$

Al conjuntar (17) y (23) se obtiene la ecuación residual para el nodo s

$$R_s = \left(D\frac{d\phi}{dx}\right)^{(e+1)}\bigg|_{x = x_s} - D\left(\frac{d\phi}{dx}\right)^{(e)}\bigg|_{x = x_s} - \left(\frac{D}{L}\right)^{(e)}\phi_{rs} + \left[\left(\frac{D}{L}\right)^{(e)} + \left(\frac{D}{L}\right)^{(e+1)}\right]\phi_s$$
$$-\left(\frac{D}{L}\right)^{(e+1)}\phi_t - \left(\frac{QL}{2}\right)^{(e)} - \left(\frac{QL}{2}\right)^{(e+1)} \qquad (24)$$

El procedimiento usual de solución, es generar el sistema de ecuaciones sin los términos entre elementos; una vez que la ecuación ha sido resuelta se puede calcular

$$\left(D\frac{d\phi}{dx}\right)^{(e+1)}\bigg|_{x = x_s} - D\left(\frac{d\phi}{dx}\right)^{(e)}\bigg|_{x = x_s} \qquad (25)$$

Teóricamente, el valor de la expresión 25 se puede utilizar para evaluar la calidad de la malla e indicar en donde ésta ha sido refinada; no obstante, todavía no se ha establecido una forma práctica de implantar esta idea.

Existe información importante en la expresión 25: si $D^{(e)} = D^{(e+1)}$ entonces el requerimiento entre elementos se reduce a

$$\left(\frac{d\phi}{dx}\right)^{(e+1)}\Bigg|_{X = X_s} = \left(\frac{d\phi}{dx}\right)^{(e)}\Bigg|_{X = X_s} \qquad (26)$$

Debe existir continuidad en las pendientes antes que el residual se haga cero; esta continuidad nunca puede alcanzarse, a menos que la solución sea una línea recta.

El valor del enunciado 27 se hace pequeño a medida que la malla se refina, no obstante, nunca es cero para todos los nodos.

El requerimiento entre elementos (26) se puede ver como un término con error similar al asociado con las aproximaciones por diferencias finitas (el error no se incorpora al sistema de ecuaciones). Sin embargo, es un residuo constante en que la solución es sólo aproximada.

Eliminando los requerimientos entre elementos de 24, suponiendo una numeración secuencial de elementos y nodos y escribiendo todas las cantidades en términos de s se obtiene la ecuación nodal residual siguiente:

$$R_s = -\frac{D^{(s-1)}}{L}\phi_{s-1} + \left[\left(\frac{D}{L}\right)^{(s-1)} + \left(\frac{D}{L}\right)^{(s-1)}\right]\phi_s - \left(\frac{D}{L}\right)^{(s)}\phi_{rs+1} - \left(\frac{QL}{2}\right)^{(s-1)} - \left(\frac{QL}{2}\right)^{(s)} = 0 \qquad (27)$$

2.3) MEF en el elemento individual

Es evidente que la teoría del MEF puede ser dividida en dos fases. La primera consiste en el estudio del *elemento individual* y la segunda es el *ensamble de los elementos* representando al cuerpo entero. Enseguida se explica lo concerniente al elemento individual.

La primera decisión que el ingeniero debe tomar es seleccionar la figura o configuración del elemento básico que se empleará en el análisis. Esta selección depende de la geometría del cuerpo o estructura y del número de coordenadas independientes que son imprescindibles para describir al problema. Existen diversos tipos de elementos, los elementos más comunes son:

a) *Elementos unidimensionales*: Cuando la geometría, propiedades del material y las variables dependientes pueden ser expresadas en términos de un único espacio coordenado independiente se emplea el elemento unidimensional. Un elemento unidimensional puede ser representado por una línea recta cuyos extremos son los puntos nodales.

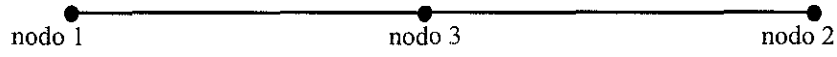


FIGURA 2.6 ELEMENTO UNIDIMENSIONAL

Los puntos 1 y 2 se denominan *nodos externos* porque representan puntos de conexión a los elementos adyacentes. El nodo 3 se conoce como *nodo interno*, y algunas veces es necesario porque no existen conexiones a otros elementos.

b) *Elementos bidimensionales*: Entre éstos están la deformación plana, esfuerzo plano y vigas. El elemento más simple para problemas bidimensionales es el triángulo (figura 2.7).

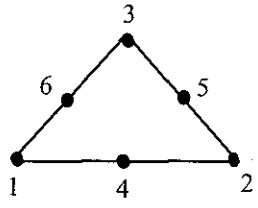


FIGURA 2.7 ELEMENTO TRIANGULAR

Existen dos arquetipos posibles de nodos externos para los elementos triangulares; los nodos de las esquinas indicados por 1, 2 y 3 son llamados *nodos externos primarios*.

Cuando hay nodos adicionales en los lados de los elementos (nodos 4, 5 y 6), se hace referencia a ellos como *nodos externos secundarios*.

Esta distinción es necesaria porque los nodos secundarios tienen menos desplazamientos de interés que los nodos de la esquina.

Otros tipos comunes de elementos bidimensionales son los cuadriláteros y rectángulos, éstos se aprecian en las figuras 2.8 y 2.9.

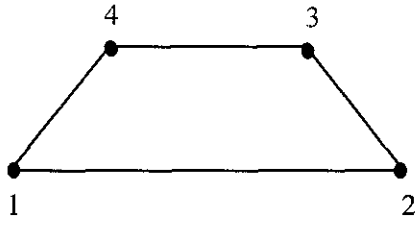


FIGURA 2.8 ELEMENTO CUADRILÁTERO

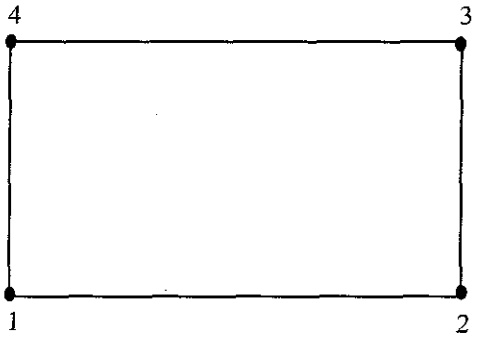
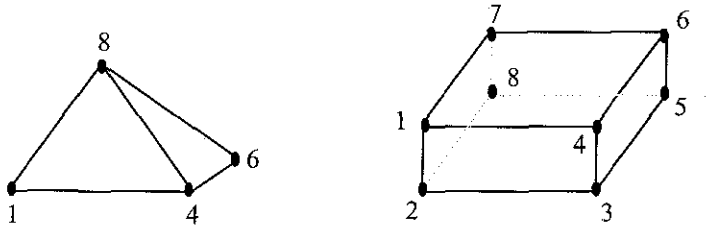


FIGURA 2.9 ELEMENTO RECTANGULAR

c) *Elementos tridimensionales*: Los elementos tridimensionales con ocho nodos externos primarios son de la forma general de un hexaedro o prisma rectangular; en la figura 2.10 se aprecian unos elementos tridimensionales.



TETRAEDRO PRISMA RECTANGULAR

FIGURA 2.10 ELEMENTOS TRIDIMENSIONALES

La solución en cada elemento debe ser casi igual al valor real, para lograrlo deben cumplirse tres condiciones:

a) Los modelos de desplazamiento deben ser continuos dentro de los elementos y los desplazamientos deben ser compatibles entre los modelos adyacentes. La primera parte de este requisito se cumple al escoger los modelos de polinomios, los cuales son inherentemente continuos. La segunda parte implica que los elementos

adyacentes se deben deformar sin causar aberturas, traslapes o discontinuidades entre los elementos.

b) Los modelos de desplazamiento deben incluir los desplazamientos del cuerpo rígido de un elemento.

c) Los modelos de desplazamiento deben contener las constantes de estado de deformación del elemento. Deben existir combinaciones de valores de las coordenadas generalizadas que causen que todos los puntos en el elemento experimenten la misma deformación.

Los desplazamientos en los nodos, rotaciones y/o deformaciones necesarias para especificar completamente la deformación del elemento finito son los *grados de libertad del elemento*. El número mínimo de grados de libertad necesarios para un elemento dado es determinado por la totalidad de restricciones de convergencia, isotropía geométrica y la necesidad de una adecuada representación de los términos de la energía potencial funcional.

Considerando estas restricciones, se puede decir que el análisis en un solo elemento finito está compuesto por los siguientes pasos:

i) Seleccionar la configuración del elemento básico.

ii) Escoger el modelo de desplazamiento y adoptar los desplazamientos en los nodos convenientes como las amplitudes del modelo.

iii) Usando las relaciones apropiadas de esfuerzo y deformación, escribir los elementos de esfuerzo y deformación en términos de los desplazamientos nodales.

iv) Sobre la base del método directo de rigidez o el principio variacional, formular la rigidez de los elementos y las cargas.

v) Si es necesario, condensar para eliminar grados de libertad internos.

2.4) MEF y el ensamble de elementos

El proceso de subdivisión es esencialmente la construcción de la malla. Se tiene que decidir el número, forma, tamaño y configuración de los elementos, de tal forma que el cuerpo sea lo mejor posible simulado. El objetivo general de tal discretización es dividir el cuerpo en elementos suficientemente pequeños para que el modelo más simple de desplazamiento pueda proporcionar la solución más real. Al mismo tiempo, se debe recordar que entre más fina sea la malla mayor será el esfuerzo computacional.

La malla más fácil de construir es la regular, ésta es una subdivisión igual de tamaño y forma para todos los elementos. Para encontrar una solución aproximada útil es necesario un refinamiento de las subdivisiones en las regiones donde se esperan las concentraciones de esfuerzo. Tres condiciones deben satisfacerse para el proceso del refinamiento de la malla. Estos requisitos son:

a) Cada punto en el cuerpo puede ser incluido dentro un pequeño elemento en cualquier etapa del refinamiento de la malla.

b) Todas las mallas previas deben ser contenidas en la malla más fina.

c) La misma forma y orden del modelo de desplazamiento debe ser retenido para la red refinada como para la malla previa.

Habitualmente a la subdivisión que cumple los dos primeros requisitos, se le llama red reducible.

Si elementos rectos son empleados, las fronteras curvadas son aproximadas por medio de pequeñas piezas lineales en los lados adyacentes de la frontera.

El proceso de construir las ecuaciones algebraicas para el ensamble de las ecuaciones de los elementos individuales es una rutina. La compatibilidad nodal es usada como base para este proceso. Este simple requisito establece que todos los elementos adyacentes para un nodo particular deben tener los mismos desplazamientos. La imposición de compatibilidad nodal representa la construcción del ensamble por la unión rígida de las piezas o elementos a ciertos puntos de unión.

El método directo de rigidez es utilizado casi universalmente para ensamblar las ecuaciones algebraicas en las aplicaciones de elementos finitos. Su popularidad reside en la economía de almacenamiento del método y su facilidad de codificación. Las etapas del método directo de rigidez son:

a) Calcular la matriz de rigidez de los elementos [k] y la de cargas [Q] para cada elemento. Generalmente éstas son almacenadas en un arreglo de tamaño n x n y uno de n x 1, respectivamente.

b) Transformar las cargas y rigideces de los elementos a coordenadas globales.

c) Si el elemento tiene grados de libertad internos, llevar a cabo el procedimiento de condensación. La matriz de multiplicadores [c], pivotes [d] y el vector condensado de cargas [f] se almacenan para la recuperación posterior de los desplazamientos internos por ecuación.

d) Utilizar un arreglo que relacione los índices nodales globales y locales para el elemento, añadir las submatrices de la rigidez de elementos [k] a la localidades apropiadas de la matriz de rigidez total [K]. Además, agregar los subvectores de las cargas de los elementos [Q] a la posición apropiada del vector total de cargas [R].

e) Regresar al paso a y repetir el proceso hasta que todos los elementos hayan sido procesados.

Como el almacenamiento es una situación crítica, se debe emplear un esquema de partición. El ancho de banda del sistema de ecuaciones final depende del tamaño de la matriz de rigidez de los elementos individuales y del sistema de notación para los nodos; si se minimiza el ancho de banda, se reduce el tiempo de solución y los requisitos de almacenamiento para la matriz de rigidez total. Hay dos pasos que hay que tomar en cuenta para esta minimización: primero, si es posible hay que evitar el uso de muchos nodos externos secundarios. Ésto puede hacerse al elegir derivadas del desplazamiento como grados de libertad adicionales en los nodos externos primarios. Segundo, se debe ejecutar una subdivisión cuidadosa y adoptar el sistema de numeración adecuado para los nodos. Así que, un aspecto esencial del proceso de discretización es la designación del sistema de numeración para los nodos y los elementos. La figura 2.11 muestra la forma más común para numerar una malla bidimensional.

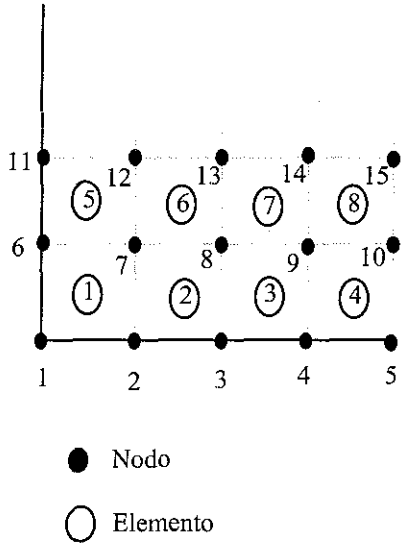


FIGURA 2.11 NUMERACIÓN DE LA MALLA

Los nodos son numerados consecutivamente de izquierda a derecha y de abajo hacia arriba, de acuerdo al sistema de la mano derecha. Una numeración similar es empleada para los elementos. Los datos adicionales deben ser proporcionados manualmente.

Si los números de los nodos sirven como base para numerar los desplazamientos en los nodos, entonces el ancho de banda de la matriz de rigidez total depende de la diferencia más grande entre dos números de nodo externos cualesquiera para un solo elemento. Dejando a D ser la máxima diferencia que ocurre en todos los elementos del montaje, el ancho de la semibanda B está dado por B = (D + 1) f. Aquí f es el número de grados de libertad en cada nodo.

La implantación en computadora del MEF se hace paralela a la solución por computadora de marcos estructurales. El análisis matricial de estructuras hace énfasis en los elementos. El sistema de ecuaciones se constituye calculando la contribución de los elementos y colocando sus valores en las posiciones correctas en el sistema final de ecuaciones. Este conjunto final de ecuaciones se forma después de que todos los elementos se han considerado; aquí se emplea notación matricial, una matriz de rigidez y un vector fuerza de los elementos para aplicarlos a las ecuaciones diferenciales analizadas previamente.

Es necesario recordar tres cosas cuando se desarrollan matrices para los elementos: 1) las ecuaciones residuales siempre se arreglan en sucesión numérica, 2) los valores nodales se arreglan secuencialmente dentro de una ecuación y 3) se desarrolla una ecuación para cada nodo. Una vez que todas las ecuaciones se han desarrollado, se incorporan las condiciones de frontera.

Primero se define un vector columna {R}, cuyos componentes representan ecuaciones residuales. El vector es

$$\{R\} = \begin{Bmatrix} R_1 \\ R_2 \\ \vdots \\ R_{p-1} \\ R_p \end{Bmatrix} \qquad (1)$$

donde R_B es la ecuación residual para el nodo B, y así $R_B^{(e)}$ es la contribución del elemento (e) a la ecuación residual para el nodo B.

La matriz de rigidez [k] siempre es simétrica y positiva definida para problemas estructurales y ecuaciones diferenciales gobernantes que son autoadjuntas. La diagonal siempre tiene valores positivos y relativamente grandes comparados con los otros elementos.

Las ecuaciones del MEF normalmente se resuelven por métodos de eliminación gaussiana y ésto es debido a que el sistema de ecuaciones no siempre es diagonalmente dominante, lo que significa que k_{ii} puede ser menor que la suma de los coeficientes no diagonales del renglón i. Sin embargo, en casos normales ésto se puede hacer y es importante porque los coeficientes diferentes de cero necesitan almacenarse en la computadora. Además, es esencial recurrir a ciertas estructuras de datos que permitan almacenar solamente el triángulo superior de la matriz.

La matriz global [K] es bandada; una matriz bandada tiene como característica que todos los coeficientes diferentes de cero se localizan relativamente cerca de la diagonal principal y todos los coeficientes más allá del ancho de banda son cero, se permite que haya ceros dentro del ancho de banda.

Una de las cantidades importantes que se desea obtener para análisis y diseño de un procedimiento de elemento finito son los esfuerzos y/o las deformaciones. Las relaciones de desplazamiento, esfuerzo y deformación dependen de la especialización del comportamiento del material, el cual debe ser propiamente aplicado.

Si [ε] es el vector de los componentes de deformación relevante en un punto arbitrario dentro del elemento finito, entonces se recurre a las ecuaciones de deformación-desplazamiento y al modelo de desplazamiento para escribir:

$$[\varepsilon] = [B_\alpha][\alpha]$$

$$[\varepsilon] = [B][q]$$

Donde $[B_\alpha]$ es la matriz de deformación-desplazamiento para las coordenadas generales; $[\alpha]$, el vector de coordenadas generalizadas; [B], la matriz de deformación-desplazamiento para los modelos de interpolación; y [q], el vector de desplazamientos nodales.

Es importante notar que [ε] y $[B_\alpha]$ o [B] son funciones de las coordenadas del espacio independiente. Sin embargo, si $[B_\alpha]$ y [B] están en el mismo sistema de coordenadas pueden relacionarse por:

$$[B] = [B_\alpha][A^{-1}]$$

Aquí, $[A^{-1}]$ representa las transformaciones de los desplazamientos relacionando $[\alpha]$ con [q].

Si $[\sigma]$ es el vector de esfuerzos correspondiente a las deformaciones $[\varepsilon]$, se emplea una matriz apropiada de las ecuaciones de esfuerzo-deformación para escribir los esfuerzos en los elementos como sigue

$$[\sigma] = [C][B_\alpha][\alpha]$$

O bien como

$$[\sigma] = [C][B][q]$$

donde [C] es la matriz esfuerzo-deformación.

Ahora, si [k] es la matriz de rigidez de los elementos y [Q] el vector de cargas en los elementos, entonces la relación de equilibrio entre estas dos variables y el vector de desplazamientos nodales [q] es expresada como un conjunto de ecuaciones algebraicas lineales simultáneas de la forma:

$$[k][q] = [Q]$$

La rigidez relaciona los desplazamientos en los puntos nodales con las fuerzas aplicadas en los puntos nodales. Los elementos de la matriz de rigidez son los coeficientes de influencia. La rigidez de una estructura es un coeficiente de influencia

que suministra la fuerza en un punto sobre una estructura asociada con un desplazamiento unitario en un mismo (o diferente) punto.

La matriz de rigidez para un elemento depende de: el modelo de desplazamiento, la geometría del elemento y las propiedades locales del material o sus relaciones constitutivas.

CAPÍTULO 3. CASOS DEL MEF

CAPÍTULO 3. CASOS DEL MEF

Una de las mayores ventajas del MEF es la facilidad para generalizar a problemas bidimensionales compuestos de varios y diferentes materiales y con fronteras irregulares. La discusión de estos elementos se inicia considerando los elementos triangulares lineales y rectangulares bilineales, debido a que éstos son los más empleados en los casos isoparamétrico y axisimétrico (que se explicarán en esta sección).

3.1) Caso isoparamétrico en 2D

La formulación de los modelos de desplazamiento y los cálculos de los elementos de rigidez es simplificada y generalizada por el concepto conocido como elemento isoparamétrico. Las principales bases de éste son el sistema de coordenadas natural y los modelos de desplazamiento e interpolación.

Todas las soluciones del MEF requieren la evaluación de integrales, algunas son fáciles de resolver, pero otras son difíciles, muchas son imposibles de evaluar analíticamente y es cuando hay que emplear técnicas numéricas.

Las dificultades asociadas con la evaluación de una integral se pueden disminuir cambiando las variables de integración; para ello, hay que escribir la integral en un nuevo sistema de coordenadas. Las funciones de forma lineal para un elemento en el cual el origen del sistema de coordenadas está a la izquierda del nodo i son

$$N_i(x) = \frac{x_j - x}{L} \quad y \quad N_j(x) = \frac{x - x_i}{L} \qquad (1)$$

La desventaja de estas funciones de forma se nota cuando se evalúan integrales que involucran productos de las funciones de forma, tales como

$$\int_{x_i}^{x_j} N_i(x) N_j(x) dx \quad o \quad \int_{x_i}^{x_j} N_i^{\,2}(x) \qquad (2)$$

Lo mismo sucede en problemas de campo y de mecánica de sólidos. La integración en 2 se puede simplificar desarrollando nuevas funciones de forma circunscritas a un sistema de coordenadas cuyo origen esté localizado en el elemento, éste se denomina sistema de coordenadas local. Los sistemas de coordenadas locales más comunes para elementos unidimensionales tienen el origen en el nodo i o en el centro del elemento (figura 3.1).

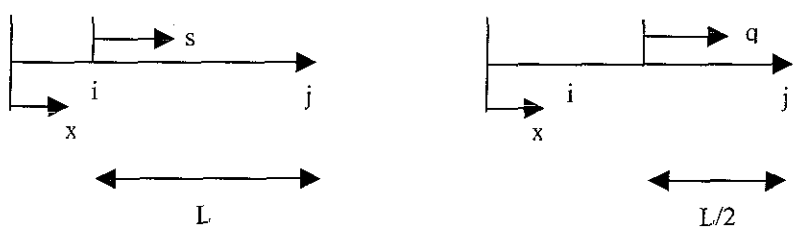


FIGURA 3.1 SISTEMA DE COORDENADAS LOCAL

Las funciones de forma para un sistema de coordenadas local con origen en i se obtienen de 1 reemplazando a x por $x = x_i + s$, lo que conlleva a

$$N_i(s) = \frac{x_j - x}{L} = \frac{x_j - (x_i + s)}{L} = 1 - \frac{s}{L} \qquad (3) \quad y$$

$$N_j(s) = \frac{x - x_i}{L} = \frac{x_i + s - x_i}{L} = \frac{s}{L} \qquad (4)$$

Cada función de forma vale uno en su propio nodo y cero en los otros nodos; la función de forma para un sistema de coordenadas localizado en el centro del elemento se obtiene de la expresión 1 y es $x = x_i + (L/2) + q$. Las funciones de forma relativas a este origen son:

$$N_i(q) = \frac{1}{2} - \frac{q}{L} \quad y \quad N_j(q) = \frac{1}{2} + \frac{q}{L} \qquad (5)$$

Las funciones de forma 3, 4 y 5 son de utilidad únicamente si se efectúa un cambio en las variables de integración. El cambio de variable en la integral produce

$$\int_a^b f(x)dx = \int_{p_1}^{p_2} f(g(p))\left[\frac{d(g(p))}{dp}\right]dp \qquad (6)$$

donde p es la variable de la nueva coordenada y g(p) es la ecuación que relaciona a x con p, es decir, $x = g(p)$.

La forma de interpretar estas ecuaciones con relación a los sistemas de coordenadas de las figuras anteriores es: para la coordenada s, donde $x = x_i + s$,

$$\int_{x_i}^{x_j} f(x)dx = \int_{s_1}^{s_2} \frac{d(x_i + s)}{ds} ds = \int_0^L h(s)ds \qquad (7)$$

donde h(s) es f(x) en términos de s. Los límites de integración se obtienen sustituyendo x_i por $x = x_i + s$ y resolviendo para s.

Para la coordenada q, donde $x = x_i + L/2 + q$

$$\int_{x_i}^{x_j} f(x)dx = \int_{q_1}^{q_2} r(q) \frac{d(x_i + \frac{1}{2} + q)}{dq} dq = \int_{-L/2}^{L/2} r(q)dq \qquad (8)$$

donde r(q) es f(x) en términos de q.

La utilidad de las expresiones 7 y 8 se aprecia cuando se calculan integrales como

$$\int_{x_i}^{x_j} N_i^{\,2} dx$$

Aplicando las variables como coordenadas se obtiene

$$\int_{x_i}^{x_j} N_i^{\,2}(x)dx = \int_0^L N_i^{\,2}(s)ds = \int_0^L (1-\tfrac{s}{L})^2 ds = \frac{L}{3}$$

Utilizando la coordenada q se llega a

$$\int_{x_i}^{x_j} N_i^{\,2}(x)dx = \int_{-L/2}^{L/2} N_i^{\,2}(q)dq = \int_{-L/2}^{L/2} (\tfrac{1}{2}-\tfrac{q}{L})^2 ds = \frac{L}{3}$$

El sistema de coordenadas locales s y q se puede convertir a sistemas de coordenadas naturales. Un sistema de coordenadas natural es un sistema local que permite la especificación de un punto dentro del elemento por un número adimensional cuya magnitud absoluta nunca excede la unidad.

Comenzando con la coordenada q, en la figura anterior y formando la relación

$$\frac{q}{L/2} = \frac{2q}{L} = \xi$$

La coordenada ξ varía de -1 a 1 (figura 3.2a).

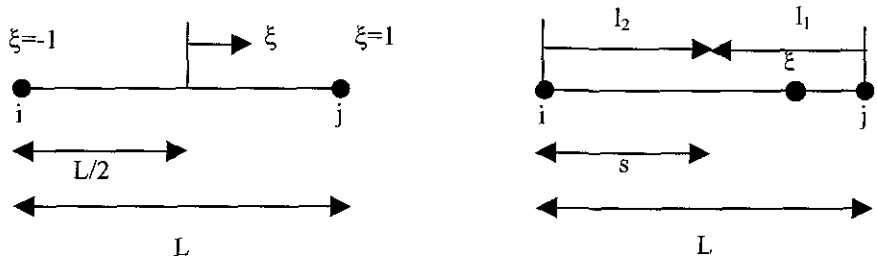


FIGURA 3.2 SISTEMA DE COORDENADAS NATURAL

Las nuevas funciones de forma son

$$N_i(\xi) = \frac{1}{2}(1-\xi) \quad y \quad N_j(\xi) = \frac{1}{2}(1+\xi) \qquad (1)$$

El cambio de variables en la integración produce

$$\int_{-L/2}^{L/2} r(q)dq = \int_{\xi_1}^{\xi_2} g(\xi)\frac{d(\xi L/2)}{d\xi}d\xi = \frac{L}{2}\int_{-1}^{1} g(\xi)d\xi \qquad (2)$$

donde g(ξ) es r(q) en términos de ξ.

La ventaja que se consigue con estas variables son los límites de integración de -1 a 1. Un método numérico apropiado para la solución de este tipo de esquemas es el Gauss-Legendre, el cual maneja puntos de muestreo y coeficientes de peso difundidos en el intervalo de -1 a 1.

Otro sistema de coordenadas naturales interesante consiste de un par de relaciones de longitud (figura 3.2 b). Si s es la distancia desde i, entonces l_1 y l_2 se definen como las razones

$$l_1 = \frac{L-s}{L} \quad y \quad l_2 = \frac{s}{L} \qquad (3)$$

Este par de coordenadas no son independientes porque

$$l_1 + l_2 = 1 \qquad (4)$$

La dificultad de estas coordenadas radica en la estimación de integrales del tipo

$$\int_0^L N_i{}^a(s) N_j{}^b(s) ds \qquad (5)$$

que involucran al producto de funciones de forma.

Para hacer el cambio de variable se aplican las relaciones siguientes

$$N_i(s) = l_1, \quad N_j(s) = l_2, \quad s = L l_2 \quad y \quad \frac{ds}{dl_2} = L$$

que originan

$$\int_0^L N_i{}^{(a)}(s) N_j{}^{(b)}(s) ds = \int_0^1 l_1{}^{(a)} l_2{}^{(b)} L dl_2 \qquad (1)$$

La integral del lado derecho de 1 puede transformarse en

$$L \int_0^1 (1-l_2)^a l_2{}^b dl_2 \qquad (2)$$

La integral en 2 tiene la misma forma que

$$\int_0^1 t^{z-1}(1-t)^{w-1} dt = \frac{\Gamma(z)\Gamma(w)}{\Gamma(z+w)} \qquad (3)$$

donde $\Gamma(n+1) = n!$, y así se llega a

$$L\int_0^1 l_1^{\,a} l_2^{\,b}\, dl_2 = L\frac{\Gamma(a+1)\Gamma(b+1)}{\Gamma(a+b+1+1)} = L\frac{a!\,b!}{(a+b+1)!} \qquad (4)$$

Esta expresión es muy útil porque establece que una integral complicada puede calcularse empleando una ecuación que involucra sólo la longitud del elemento y las potencias inherentes al producto.

Los sistemas de coordenadas, funciones de forma y límites de integración para el elemento lineal unidimensional se resumen en la siguiente tabla.

Tipo de sistema	Coordenada variable	Función de forma	Límites de integración
Global	X	$N_i=(x_j-x)/L$ $N_j=(x-x_i)/L$	x_i, x_j
Local	S	$N_i=1-s/L$	0, L
Local	Q	$N_i=(1/2-q/L)$ $N_j=(1/2+q/L)$	-L/2, L/2
Natural	ξ	$N_i=1/2(1-\xi)$ $N_j=1/2(1+\xi)$	-1, 1
Natural	l_2	$N_i=l_1$ $N_j=l_2$	0, 1

Como se dijo previamente, una de las mayores ventajas del MEF es la facilidad para generalizar a problemas bidimensionales compuestos de varios y diferentes materiales y con fronteras irregulares, para ello se utilizan elementos triangulares y rectangulares.

El elemento triangular tiene lados rectos y un nodo en cada esquina y se utiliza para modelar fronteras irregulares. La división de una región en elementos triangulares se hace más fácil dividiendo el continuo en subregiones cuadriláteras y triangulares; cada una de estas subregiones se subdivide en triángulos. Una subregión triangular es más fácilmente escindida en elementos especificando el mismo número de nodos a lo largo de cada lado y conectando los nodos apropiados por líneas rectas y colocando los nodos en los puntos de intersección.

La región triangular de la figura 3.3 se ha dividido en nueve elementos colocando cuatro nodos por lado.

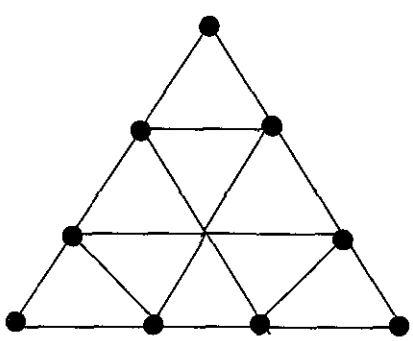


FIGURA 3.3 REGIÓN TRIANGULAR

No hay razón para que los nodos estén igualmente espaciados por lado. Al variar la separación se cambia el tamaño del elemento.

Hay $(n-1)^2$ elementos triangulares en una región triangular, donde n es el número de nodos por lado. Si la región triangular tiene lados curvos, los elementos de la frontera modelan la curvatura empleando elementos con lados rectos (ver figura 3.4)

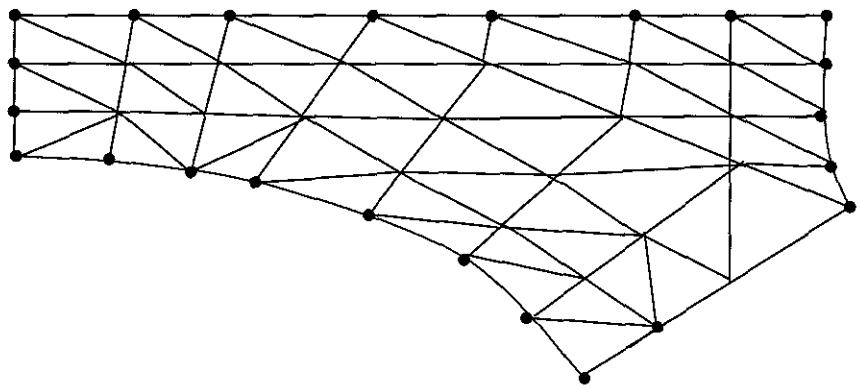


FIGURA 3.4 REGIÓN CON LADOS CURVOS

Una región cuadrilátera se divide fácilmente en elementos triangulares conectando los nodos de lados opuestos por líneas rectas. Los nodos interiores están localizados en los puntos de intersección (figura 3.5).

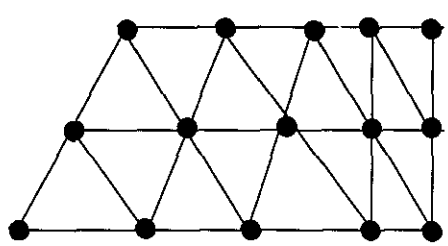


FIGURA 3.5 REGIÓN CUADRILATERAL

Los cuadriláteros interiores se dividen en elementos triangulares insertando la diagonal más corta. La división que recurre a la diagonal más corta es preferible debido a que los elementos cercanos a una forma equilátera producen resultados más

exactos y precisos que los otros. El número de nodos a lo largo de lados adyacentes de una subregión cuadrilátera no tiene que ser el mismo, pero el número de nodos en lados opuestos debe ser igual, a menos que la malla se refine. El espaciado entre nodos frontera puede ser variado para producir elementos de tamaños diferentes. Hay 2(n - 1)(m - 1) elementos triangulares en un cuadrilátero, donde n y m son el número de nodos en un par de lados adyacentes.

Los nodos sobre la frontera entre subregiones deben ser idénticos en número y deben tener la misma posición relativa, ésto es necesario para asegurar la continuidad de ϕ a través de un elemento de frontera. Una malla regular no es necesaria debido a que generalmente hay regiones en las cuales las variables nodales son relativamente constantes. La habilidad para modificar el tamaño del elemento es una ventaja importante del elemento triangular. La forma más fácil de hacer una transición en el tamaño de elementos es emplear una región cuadrilateral que tiene un número desigual de nodos en dos lados opuestos. Una buena combinación es colocar dos nodos en un lado por cada tres nodos del lado opuesto (figura 3.6).

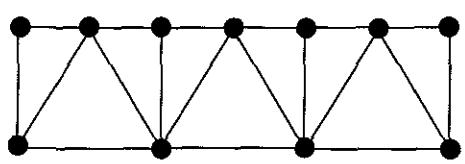


FIGURA 3.6

La asignación del número de nodos sería una operación trivial si la numeración no influyera en el ancho de banda del sistema de ecuaciones. Dos esquemas de numeración diferentes para un conjunto de nodos se ven en la figura 3.7 a y 3.7 b.

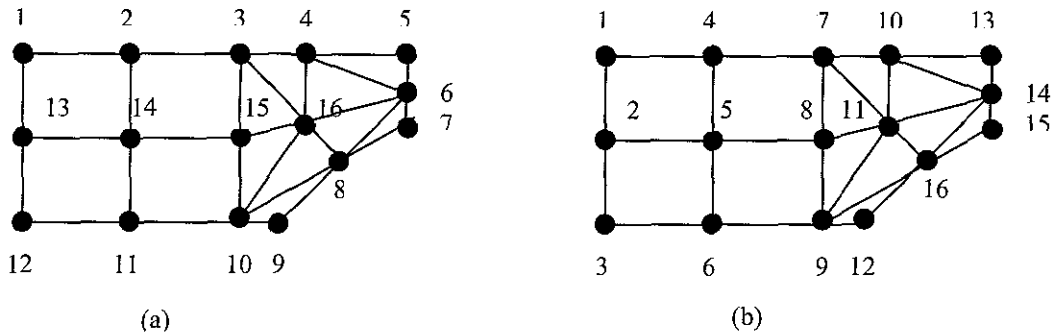


FIGURA 3.7 ESQUEMAS DE NUMERACIÓN PARA LOS NODOS

Los valores del ancho de banda son catorce y seis, respectivamente. El elemento triangular mostrado en la figura 3.8 tiene lados rectos y tres nodos, uno en cada esquina.

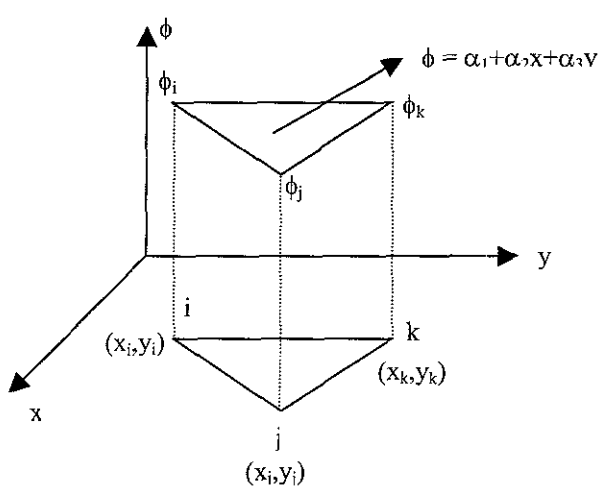


FIGURA 3.8

Es necesario que el etiquetado de los nodos sea consistente; aquí se están marcando en sentido contrario a las manecillas del reloj. El nodo i puede ser

cualquiera. Los valores nodales son ϕ_i, ϕ_j y ϕ_k, mientras que las coordenadas nodales son (x_i, y_i), (x_j, y_j) y (x_k, y_k). El polinomio de interpolación es:

$$\phi = \alpha_1 + \alpha_2 x + \alpha_3 y \qquad (1)$$

y con las condiciones nodales:

$$\begin{aligned} \phi &= \phi_i \quad en \quad x = x_i, y = y_i \\ \phi &= \phi_j \quad en \quad x = x_j, y = y_j \\ \phi &= \phi_k \quad en \quad x = x_k, y = y_k \end{aligned}$$

La solución de estas condiciones en 1 produce el sistema de ecuaciones:

$$\begin{aligned} \phi_i &= \alpha_1 + \alpha_2 x_i + \alpha_3 y_i \\ \phi_j &= \alpha_1 + \alpha_2 x_j + \alpha_3 y_j \\ \phi_k &= \alpha_1 + \alpha_2 x_k + \alpha_3 y_k \end{aligned} \qquad (2)$$

de donde se obtiene

$$\begin{aligned} \alpha_1 &= \frac{1}{2A}\left[(x_j y_k - x_k y_j)\phi_i + (x_k y_i - x_i y_k)\phi_j + (x_i y_j - x_j y_i)\phi_k\right] \\ \alpha_2 &= \frac{1}{2A}\left[(y_j - y_k)\phi_i + (y_k - y_i)\phi_j + (y_i - y_j)\phi_k\right] \\ \alpha_3 &= \frac{1}{2A}\left[(x_k - x_j)\phi_1 + (x_i - x_k)\phi_j + (x_j - x_i)\phi_k\right] \end{aligned}$$

donde el determinante

$$\begin{vmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_k & y_k \end{vmatrix} = 2A \qquad (3)$$

y A = área del triángulo.

Sustituyendo para α_1, α_2 y α_3 en la expresión 1 y arreglándola se produce una ecuación para ϕ en términos de tres funciones de forma y ϕ_i, ϕ_j y ϕ_k que es

$$\phi = N_i\phi_i + N_j\phi_j + N_k\phi_k \qquad (4)$$

donde

$$N_i = \frac{1}{2A}(a_i + b_i x + c_i y) \qquad (5)$$

$$N_j = \frac{1}{2A}(a_j + b_j x + c_j y) \qquad (6)$$

$$N_k = \frac{1}{2A}(a_k + b_k x + c_k y) \qquad (7)$$

y

$$a_i = x_j y_k - x_k y_j \qquad b_i = y_j - y_k \qquad c_i = x_k - x_j$$
$$a_j = x_k y_i - x_i y_k \qquad b_j = y_k - y_i \qquad y \qquad c_j = x_i - x_k$$
$$a_k = x_i y_j - x_j y_i \qquad b_k = y_i - y_j \qquad c_k = x_j - x_i$$

El escalar ϕ está relacionado a los valores nodales por un conjunto de funciones de forma que son lineales en X y Y, lo cual significa que los gradientes $\partial\phi/\partial x$ y $\partial\phi/\partial y$ son constantes dentro del elemento. Un gradiente constante dentro de algún elemento significa que muchos elementos pequeños deben emplearse para aproximar en forma precisa y exacta un cambio rápido en ϕ.

Una función de forma cambia linealmente a lo largo de los lados entre sus nodos y los otros dos nodos, ésto es, N_i varía linealmente a lo largo de los lados ij e ik;

una función de forma es cero a lo largo del lado opuesto a su nodo, ésto es, N_i es cero a lo largo del lado jk. Así que ϕ cambia linealmente a lo largo de los tres lados.

También se sabe que cualquier línea de ϕ es una línea recta que cruza dos lados del elemento.

El elemento rectangular bilineal tiene longitud 2b y altura 2a; los nodos se etiquetan como i, j, k y m, siendo i el nodo de la esquina inferior izquierda, ésto se muestra en la figura 3.9.

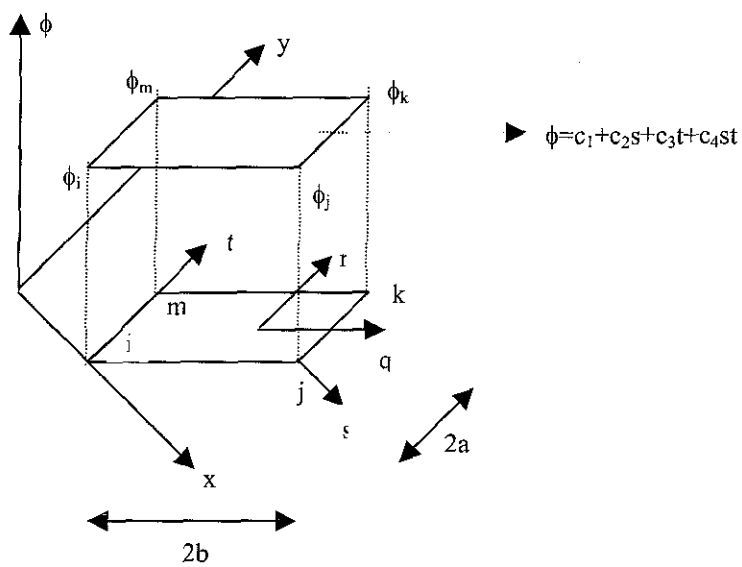


FIGURA 3.9 ELEMENTO RECTANGULAR

La ecuación de interpolación se escribe en términos de las coordenadas locales s y t. Existen al menos tres elecciones con

$$\phi = c_1 + c_2 s + c_3 t + c_4 st \qquad (1)$$

Esta ecuación es útil porque ϕ es lineal en s a lo largo de cualquier línea de la constante t y es lineal en t a lo largo de cualquier línea de la constante s. Debido a eso

se dice que el elemento es bilineal. La ecuación 1 está escrita en función a un sistema de coordenadas local cuyo origen es el nodo i, ya que las funciones de forma son más fáciles de evaluar en este marco de referencia. Otro sistema de coordenadas popular es el rq que tiene su origen en el centro del elemento.

Los coeficientes c_1, c_2, c_3 y c_4 se obtienen empleando los valores nodales de ϕ y las coordenadas nodales st para generar cuatro ecuaciones que son

$$\begin{aligned}
\phi_i &= c_1 \\
\phi_j &= c_1 + 2tc_2 \\
\phi_k &= c_1 + (2b)c_2 + 2ac_3 + 4abc_4 \qquad (2) \\
\phi_m &= c_1 + 2ac_3
\end{aligned}$$

Las cuales al resolverlas generan

$$\begin{aligned}
c_1 &= \phi_i \\
c_2 &= \frac{1}{2b}(\phi_j - \phi_i) \\
c_3 &= \frac{1}{2a}(\phi_m - \phi_i) \qquad (3) \\
c_4 &= \frac{1}{4ab}(\phi_i - \phi_j + \phi_k - \phi_m)
\end{aligned}$$

Reemplazando la expresión 3 en la 1 y acomodándola, se obtiene

$$\phi = N_i\phi_2 + N_j\phi_j + N_k\phi_k + N_m\phi_m \qquad (4)$$

donde

$$N_i = (1-\tfrac{5}{2b})(1-\tfrac{t}{2a}), \quad N_j = \frac{s}{2b}(1-\tfrac{t}{2a}), \quad N_k = \frac{st}{4ab}, \quad N_m = \frac{t}{2a}(1-\tfrac{3}{2b}) \qquad (5)$$

Cada función de forma varía linealmente a lo largo de sus elementos entre su nodo y los dos adyacentes. Así que N_i cambia linealmente a lo largo de los lados ij e im.

También cada función de forma es cero en los lados que no tocan a su nodo, ésto es N_i es cero en los lados jk y km. La transformación lineal de ϕ en un lado del elemento rectangular y un lado del elemento triangular significa que estos dos elementos son compatibles y se pueden utilizar adyacentes uno con otro.

Las ecuaciones de transformación entre los sistemas de coordenadas st y qr son:

$$s = b + q \quad y \quad t = a + r \qquad (6)$$

Al sustituir la expresión 6 en la 5 se obtienen las funciones de forma en términos de q y r.

$$N_i = \frac{1}{4}(1-\tfrac{q}{b})(1-\tfrac{r}{a}), \quad N_j = \frac{1}{4}(1+\tfrac{q}{b})(1-\tfrac{r}{a}),$$
$$N_k = \frac{1}{4}(1+\tfrac{q}{b})(1+\tfrac{r}{a}), \quad N_m = \frac{1}{4}(1-\tfrac{q}{b})(1+\tfrac{r}{a}) \qquad (7)$$

Estas funciones de forma son benéficas porque conducen a un sistema de coordenadas natural que permite al rectángulo deformarse en un cuadrilátero general.

La línea de contorno en un elemento rectangular, generalmente es curva; la intersección de esta línea con los lados se obtiene por interpolación. La forma más

fácil de obtener un tercer punto es ajustar s o t a cero en las ecuaciones de las funciones de forma y resolver la ecuación 4 para los otros valores coordenados.

Los sistemas de coordenadas naturales se definen para elementos bidimensionales y tienen las mismas ventajas que para los unidimensionales y son más convenientes para integración analítica y numérica.

La figura 3.10 muestra un elemento rectangular en el sistema de coordenadas natural, se localiza en el centro del elemento y las coordenadas son relaciones de longitud

$$\xi = \frac{q}{b} \quad y \quad \eta = \frac{r}{a} \qquad (12)$$

donde q y r son coordenadas locales.

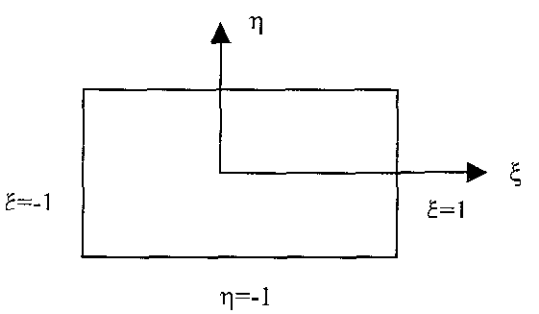


FIGURA 3.10 ELEMENTO RECTANGULAR EN SISTEMA DE COORDENADAS NATURAL

Las funciones de forma se convierten muy fácilmente al sistema de coordenadas natural

$$N_i = \frac{1}{4}(1-\xi)(1-\eta), \quad N_j = \frac{1}{4}(1+\xi)(1-\eta),$$
$$N_k = \frac{1}{4}(1+\xi)(1+\eta), \quad N_m = \frac{1}{4}(1-\xi)(1+\eta) \qquad (13)$$

El rango para ξ y η es $-1 \leq \xi \leq 1$ y $-1 \leq \eta \leq 1$.

Un sistema de coordenadas natural para el elemento triangular se obtiene definiendo las relaciones de longitud L_1, L_2 y L_3 según se aprecia en la figura 3.11a.

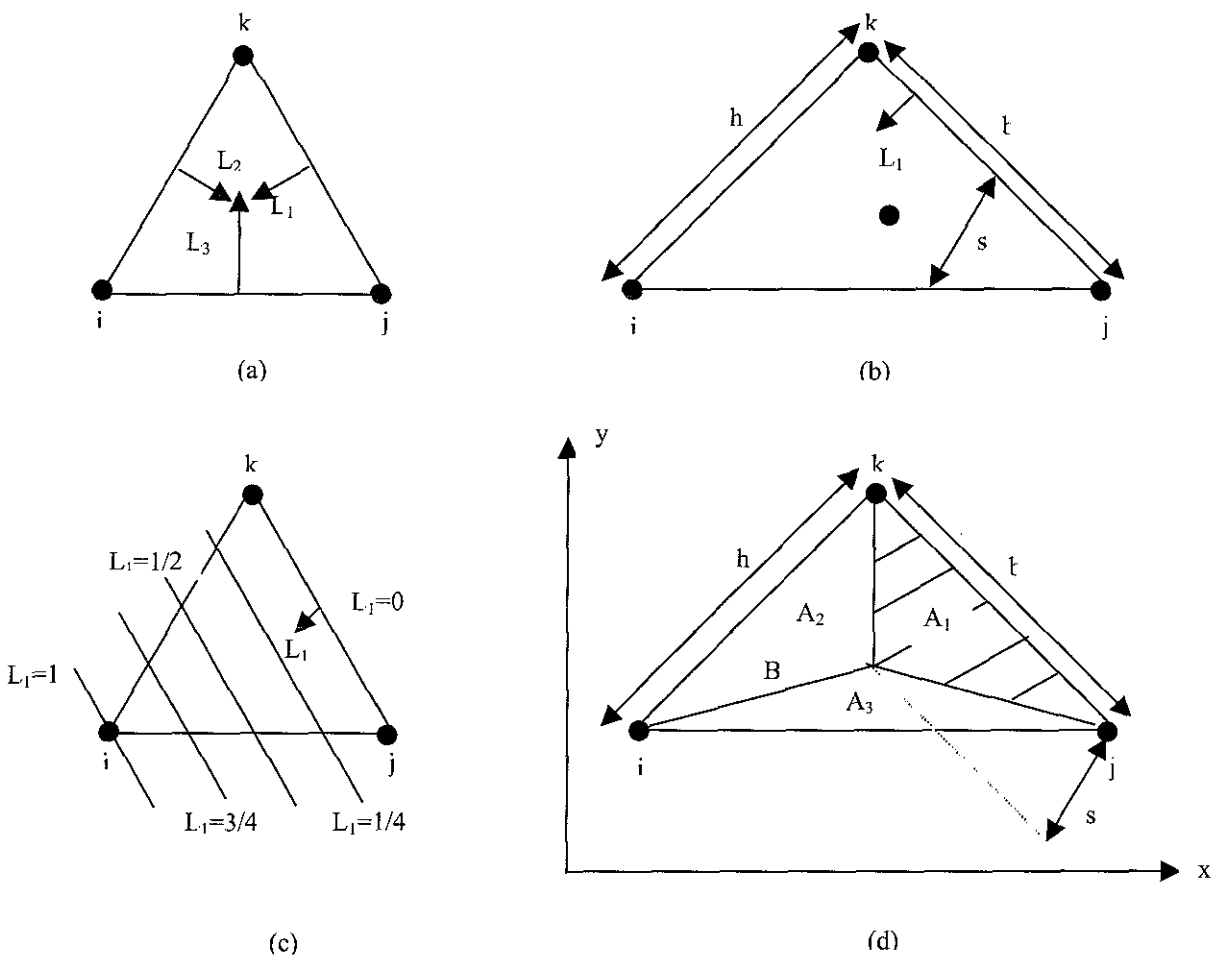


FIGURA 3.11 SISTEMA DE COORDENADAS NATURAL PARA UN ELEMENTO TRIANGULAR

Cada coordenada es la razón de una distancia perpendicular de un lado s a la altura h de ese mismo lado (figura 3.11b). Cada coordenada es una razón de longitud que varía entre 0 y 1. Las líneas de constante L_1 se ven en la figura 3.11c; cada una de estas líneas es paralela al lado desde el cual se mide L_1.

Las coordenadas L_1, L_2 y L_3 se llaman coordenadas de área porque sus valores dan la razón del área de una región subtriangular al área del triángulo completo.

Considerando el punto B de la figura 3.11d, el área de todo el triángulo es A=bh/2 y el área del triángulo sombreado es

$$A_1 = \frac{bs}{2} \qquad (1)$$

Así que la razón es

$$\frac{A_1}{A_2} = \frac{s}{h} = L_1 \qquad (2)$$

De igual forma se encuentran L_2 y L_3:

$$L_2 = \frac{A_2}{A} \quad y \quad L_3 = \frac{A_3}{A} \qquad (3)$$

y ya que $A = A_1 + A_2 + A_3$, entonces

$$L_1 + L_2 + L_3 = 1 \qquad (4)$$

Ésto es debido a que las tres coordenadas no son independientes, para encontrar un punto se utilizan dos de las coordenadas.

La ecuación 2 se puede emplear de otra forma: multiplicando el numerador y el

$$L_1 = \frac{2A_1}{2A} \qquad (5)$$

denominador por 2 y se obtiene

En forma de determinante sería

$$2A_1 = \begin{vmatrix} 1 & x & y \\ 1 & x_j & y_j \\ 1 & x_k & y_k \end{vmatrix}$$

o lo que es lo mismo

$$2A_1 = (x_j y_k - x_k y_j) + (y_j - y_k)x + (x_k - x_j)y \qquad (6)$$

donde x y y son las coordenadas de B. Sustituyendo la expresión 6 en la 5 se llega a

$$L_1 = \frac{1}{2A}\left[(x_j - y_k - x_k y_j) + (y_j - y_k)x + (x_k - x_j)y\right] \qquad (7)$$

y así

$$L_1 = N_i \qquad (8)$$

Un análisis similar para L_2 y L_3 produce

$$L_2 = N_j \quad y \quad L_3 = N_k \qquad (9)$$

Las coordenadas de área para el elemento triangular son idénticas a las funciones de forma de tal manera que los dos conjuntos se pueden intercambiar. La ventaja de utilizar el sistema de coordenadas de área es la existencia de una ecuación integral que simplifica la evaluación de las integrales de área, ésta es

$$\int_A L_1^{\,a} L_2^{\,b} L_3^{\,c}\, dA = \frac{a!\,b!\,c!}{(a+b+c+2)!}\,2A \qquad (10)$$

Cuando se incorporan las condiciones de frontera o las cargas de superficie en el análisis del MEF se requiere evaluar una integral a lo largo de la orilla del elemento; estas integrales se evalúan fácilmente una vez que se sabe como se comportan en la orilla las coordenadas de área. Considerando el punto B en el lado ij de la figura 3.12

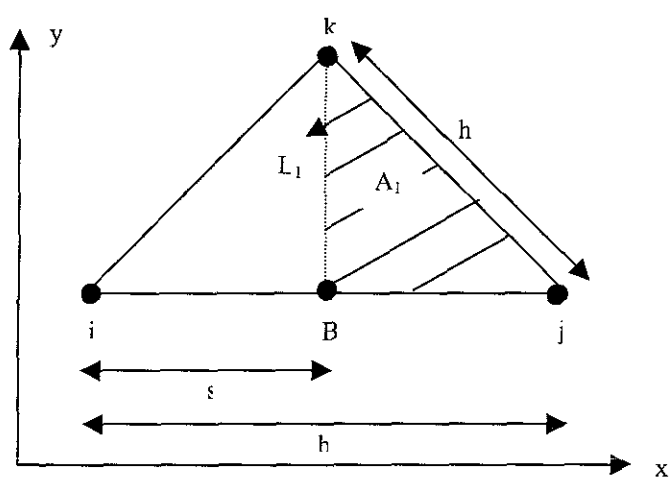


FIGURA 3.12

La coordenada L_3 es cero y L_1 es la razón del área sombreada al área total. La coordenada variable s es paralela a ij y se mide desde el nodo i; si la coordenada B es s y su longitud es b, entonces

$$L_1 = \frac{2A_1}{2A} = \frac{2h(b-s)/2}{2bh/2} = \frac{b-s}{b} = 1 - \frac{s}{b} \qquad (11)$$

La coordenada de área L_2 es

$$L_2 = \frac{s}{b}$$

Las coordenadas de área L_1 y L_2 se reducen a las funciones de forma unidimensional $N_i(s)$ y $N_j(s)$ definidas anteriormente. Empleando las coordenadas naturales unidimensional l_1 y l_2, las coordenadas de área son

$$L_1 = l_1 \quad y \quad L_2 = l_2 \quad \text{lado} \quad i-j \qquad (12)$$

Las relaciones para los otros dos lados son

$$L_2 = l_1 \quad y \quad L_3 = l_2 \quad \text{lado} \quad j-k \qquad (13)$$

$$L_3 = l_1 \quad y \quad L_1 = l_2 \quad \text{lado} \quad k-i \qquad (14)$$

La importancia de las relaciones en las expresiones 12, 13 y 14 radica en que cualquier integral sobre la orilla de un elemento triangular se puede remplazar por una línea integral escrita en términos de s o l_2, ésto es

$$\int_{\Gamma} f(L_1, L_2, L_3)d\Gamma = \int_0^L g(s)ds = L\int_0^1 \eta(l_2)dl_2 \qquad (15)$$

La frontera de un elemento bidimensional se denota por Γ.

La función de aproximación $\phi(x,y)$ consiste de un conjunto de ecuaciones continuas suavizadas, definidas cada una sobre un elemento simple. La necesidad de integrar estas funciones genera un requerimiento de continuidad entre elementos. La integral

$$\int_0^H \frac{d^n\phi}{dx^n}dx$$

se define únicamente si ϕ tiene continuidad de orden (n-1). Ésto asegura que sólo *existen discontinuidades (saltos) en la misma derivada; es decir, que la primera* derivada de la función de aproximación debe ser continua entre elementos si la integral contiene términos en segunda derivada, n = 2. Además ϕ debe ser continua entre elementos, pero sus derivadas no tienen que ser continuas. En el elemento viga se requiere continuidad en la derivada.

Como ya se dijo, el elemento isoparamétrico se basa en el sistema de coordenadas natural y los modelos de desplazamiento e interpolación. Además de cumplir estos dos requisitos, la geometría y desplazamientos de los elementos son definidos en términos de los mismos parámetros. Este concepto ayuda a formular elementos de cualquier orden con modelos de desplazamiento isotrópicos, por este motivo satisfacen los requisitos de totalidad y compatibilidad.

Si las funciones de forma en coordenadas naturales llenan la continuidad de la geometría y desplazamientos dentro del elemento y entre elementos adyacentes, se

puede mostrar que la condición de compatibilidad es satisfecha en las coordenadas globales. Se observa que los desplazamientos individuales en cualquier parte del elemento dependen sólo de los desplazamientos en los nodos.

Más aún, si el modelo de interpolación produce desplazamientos del cuerpo rígido en el sistema de coordenadas naturales local, las condiciones de desplazamiento y las constantes de estado de deformación se cumplen en las coordenadas globales.

La concepción del elemento isoparamétrico es una herramienta poderosa generalizada para construir elementos completos y conformantes de cualquier orden.

3.2) Caso Axisimétrico

Hay un grupo de problemas de campo tridimensionales que pueden resolverse usando elementos bidimensionales. Estos problemas poseen simetría alrededor de un eje de rotación y se conocen como problemas axisimétricos. Las condiciones de frontera, así como la geometría deben ser independientes de la dirección circunferencial.

La ecuación de campo en un sistema de coordenadas cilíndrico (r, θ, z) es

$$D_r \frac{\partial^2 \phi}{dr^2} + \frac{D_r}{r} \frac{\partial \phi}{dr} + \frac{D_\theta}{r^2} \frac{\partial^2 \phi}{d\theta^2} + D_z \frac{\partial^2 \phi}{dz^2} + Q = 0 \qquad (1)$$

Como los problemas axisimétricos son independientes de θ, la expresión 1 se reduce a:

$$D_r \frac{\partial^2 \phi}{dr^2} + \frac{D_r}{r} \frac{\partial \phi}{dr} + D_z \frac{\partial^2 \phi}{dz^2} + Q = 0 \qquad (2)$$

la cual también se puede escribir como

$$\frac{1}{r}\left[D_r \frac{\partial}{dr}\left(r \frac{\partial \phi}{\partial r}\right)\right] + D_z \frac{\partial^2 \phi}{dz^2} + Q = 0 \qquad (3)$$

suponiendo que D_r es una constante.

Las condiciones de frontera asociadas con la expresión 3:

$$\phi(\Gamma) = \text{valores especificados} \qquad (4)$$

sobre una parte de la frontera (llamada Γ_1) y

$$D_r \frac{\partial \phi}{dr} \cos\theta + D_z \frac{\partial \phi}{dz} \operatorname{sen}\theta = M\phi_b + s \qquad (5)$$

sobre el resto de la frontera (llamada Γ_2).

El elemento axisimétrico se obtiene rotando un elemento bidimensional alrededor del eje Z para obtener un toro. En la figura 3.13 se ilustra ésto con un elemento triangular.

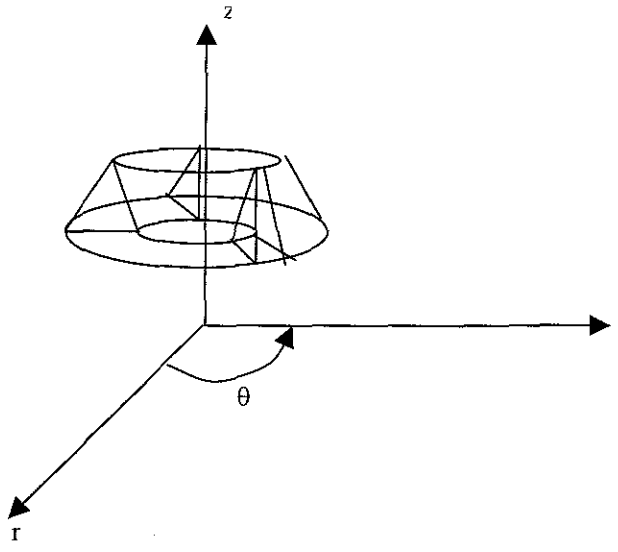


FIGURA 3.13

Un elemento triangular simple en el plano r – z se muestra en la figura 3.14; este elemento es idéntico a los vistos anteriormente sólo que las coordenadas variables son r y z en lugar de x y y.

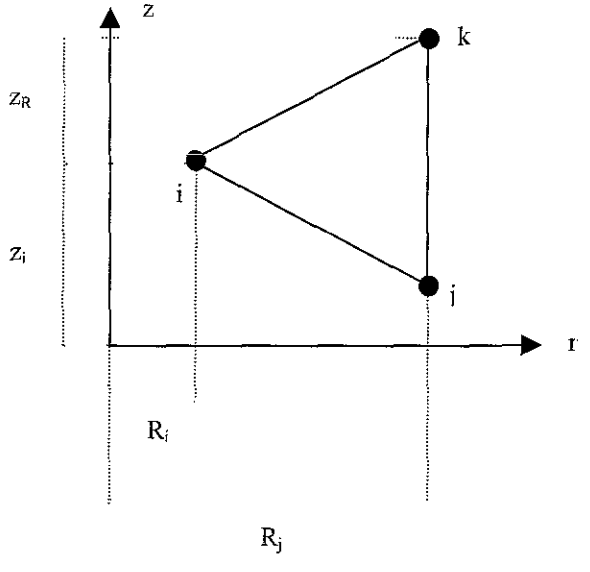


FIGURA 3.14 ELEMENTO TRIANGULAR EN EL PLANO R - Z

La variable ϕ y las funciones de forma triangular en el nuevo sistema de coordenadas son:

$$\phi = N_i\phi_i + N_j\phi_j + N_k\phi_k \qquad (1)$$

donde

$$N_i = \frac{1}{2A}(a_i + b_i r + c_i z)$$
$$N_j = \frac{1}{2A}(a_j + b_j r + c_j z) \qquad (2)$$
$$N_k = \frac{1}{2A}(a_k + b_k r + c_k z)$$

con

$$a_i = R_j Z_k - R_k Z_k, \quad b_i = Z_j - Z_k, \quad c_i = R_k - R_j$$
$$a_j = R_k Z_i - R_i Z_k, \quad b_j = Z_k - Z_i, \quad c_j = R_i - R_k$$
$$a_k = R_i Z_j - R_j Z_i, \quad b_k = Z_i - Z_j, \quad c_k = R_j - R_i$$

La figura 3.15 muestra un elemento rectangular simple en el plano r – z.

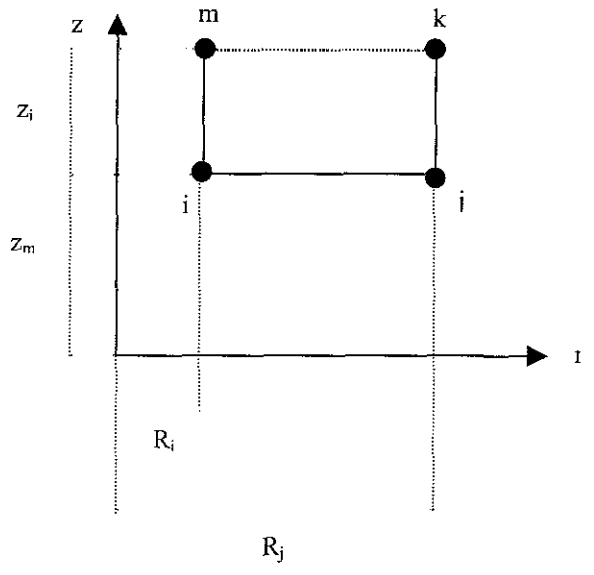


FIGURA 3.15 ELEMENTO RECTANGULAR EN EL PLANO R-Z

Las funciones de forma para este elemento son:

$$N_i = \frac{1}{4ab}(R_j - r)(Z_m - z)$$
$$N_j = \frac{1}{4ab}(r - R_i)(Z_m - z) \qquad (3)$$
$$N_k = \frac{1}{4ab}(r - R_i)(z - Z_i)$$
$$N_m = \frac{1}{4ab}(R_j - r)(z - Z_i)$$

La integral de peso residual para un problema axisimétrico es:

$$\{R^e\} = -\int_v [N]^t \left(\frac{D_r}{r}\frac{\partial}{\partial r}\left(r\frac{\partial \phi}{\partial r}\right) + D_z\frac{\partial^2 \phi}{\partial z^2} + Q\right)dv \qquad (1)$$

Los términos a derivar deben transformarse en formas de orden inferior empleando la regla del producto para diferenciación y el teorema de Gauss; ésta conduce a:

$$\frac{\partial}{\partial z}([N]^T \tfrac{\partial \phi}{\partial z}) = [N]^T \tfrac{\partial^2 \phi}{\partial z^2} + \frac{\partial [N]^T}{\partial z}\frac{\partial \phi}{\partial z} \qquad (2)$$

y reacomodando

$$([N]^T \tfrac{\partial^2 \phi}{\partial z^2}) = \frac{\partial}{\partial z}([N]^T \tfrac{\partial \phi}{\partial z}) - \frac{\partial [N]^T}{\partial z}\frac{\partial \phi}{\partial z} \qquad (3)$$

La forma general de R(e) es

$$\{R^{(e)}\} = \{I^{(e)}\} + \{k^{(e)}\}\{\phi^{(e)}\} - \{f_Q{}^{(e)}\} \qquad (4)$$

en donde

$$\{I^{(e)}\} = \int_\Gamma [N]^T (D_r \tfrac{\partial \phi}{\partial r}\cos\theta + D_z \tfrac{\partial \phi}{\partial z}\operatorname{sen}\theta) d\Gamma \qquad (5)$$

$$\{f_Q{}^{(e)}\} = \int_v Q[N]^T dv \qquad (6)$$

$$\{k^{(e)}\} = \int_v (D_r \tfrac{\partial [N]^T}{\partial r}\tfrac{\partial [N]}{\partial r} + D_z \tfrac{\partial [N]^T}{\partial z}\tfrac{\partial [N]}{\partial z}) dv \qquad (7)$$

El volumen de un área que da vueltas alrededor del eje z es V= $2\pi\bar{r}$A, donde $\bar{r}$ es la distancia radial al centroide del área; la cual para un elemento triangular es

$$\bar{r} = \frac{R_i + R_j + R_k}{3} \qquad (8)$$

o

$$r = N_i R_i + N_j R_j + N_k R_k = L_1 R_i + L_2 R_j + L_3 R_k \qquad (9)$$

Al hacer las manipulaciones matemáticas correspondientes se llega a

$$\left\{f_Q^{(e)}\right\} = \frac{2\pi QA}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}\begin{Bmatrix} R_i \\ R_j \\ R_k \end{Bmatrix} \qquad (10)$$

Una Q uniforme dentro del elemento no se distribuye uniformemente entre los nodos como ocurre con el elemento bidimensional. Cada nodo recibe una cantidad relacionada con su distancia radial desde el origen.

CAPÍTULO 4. APLICACIÓN

CAPÍTULO 4. APLICACIÓN

Diffpack es un software hecho en Noruega que sigue los preceptos de la orientación a objetos. Además es una herramienta vigorosa para desarrollar software numérico para resolver ecuaciones diferenciales parciales.

A continuación se proporciona más información de este software.

4.1) Generalidades

Diffpack es una biblioteca que contiene clases escritas en el lenguaje de programación C++. Las clases pueden utilizarse en programas desarrollados por el usuario (investigadores, estudiantes, programadores, etc.) en los campos de análisis numérico, matemáticas, física, mecánica, ingeniería, etc. Algunas de las herramientas con que cuenta Diffpack son: vectores, matrices, arreglos multidimensionales, cadenas de caracteres, entrada y salida estándar simple y mejorada, un sistema simple de menúes para obtener datos de entrada, administración de los archivos de resultados (curvas y campos), visualización de parejas ordenadas, representación de sistemas lineales, representación de sistemas de matrices grandes dispersas que se utilizan en el método del elemento finito, un gran número de método iterativos para sistemas de matrices dispersas, diferentes mallas y campos, campos escalares y vectoriales sobre rejillas del elemento finito y diferencias finitas, una colección de elementos finitos, varios algoritmos de elementos finitos, diversas estructuras de datos, distribuciones de probabilidad, generadores de números aleatorios, métodos de solución para

ecuaciones diferenciales estocásticas ordinarias, soporte para campos aleatorios y ejemplos numéricos en simuladores para problemas como las ecuaciones de Navier en elasticidad lineal, las ecuaciones de Navier-Stokes para flujo de fluidos laminares, la transferencia-difusión de calor o contaminación, etc. Su funcionalidad incluye elaboración de mallas adaptativas, métodos para rejilla múltiple y computación paralela; estas características se proporcionan en cajas de herramientas separadas.

El sistema de menús de Diffpack permite la selección en tiempo real de todas las entidades de aplicación, desde parámetros numéricos simples hasta cantidades abstractas como los tipos de elementos, formatos de matrices, etc.

Diffpack provee además herramientas de preprocesamiento y posprocesamiento para soportar rejillas, mapeos transfinitos y técnicas de superelementos, así como interfaces para Geompack y Triangle. El preprocesamiento es después extendido por la caja de herramientas Datafilter del mismo Diffpack. El posprocesamiento soporta herramientas como MatLab, Gnuplot, IRIS Explorer, AVS y Vtk.

La documentación completa de proyectos conteniendo la descripción del problema, código fuente, resultados numéricos, imágenes y películas puede ser generada en formato ASCII, LaTeX o HTML.

Diffpack se encuentra en versiones para Unix, Linux, Solaris y Win32. La versión con la cual se trabajó, es para ambiente Unix. Cuando se trabaja bajo ambiente Unix se debe emplear el csh o tcsh. También los programas de Diffpack deben estar localizados en directorios especiales, por lo tanto es necesario crear un directorio para almacenar los programas que se generen, ésto se hace con

Mkdir nombre_del_directorio

Todos los comandos de Unix trabajan de forma normal en los directorios de Diffpack.

La M es importante, ya que Diffpack hace diferencia entre mayúsculas y minúsculas. La instrucción anterior es un comando especial para crear los directorios de Diffpack. Se trata de una especialización del comando mkdir de Unix. Después es necesario cambiarse al directorio creado, mediante cd nombre_del_directorio, y se verifica su contenido tecleando ls. La computadora debe listar

Makefile

Éste es un archivo hecho por el comando Mkdir, el cual es muy importante para poder compilar los programas de Diffpack.

Emplear Diffpack significa que se debe hacer un programa en C++ para acceder a las características de Diffpack. La idea fundamental de Diffpack es tener una colección de macros o funciones disponibles para utilizarse en programas escritos en C++. Estas macros harán más sencillo el desarrollo de software numérico para resolver ecuaciones diferenciales.

Cuando se programe, el nombre del archivo no es importante, pero sí lo es la extensión, ésta debe ser .C.

Para compilar un programa se teclea

Make

Este comando muestra los pasos de la compilación y el proceso de ligado. El resultado es un archivo ejecutable con el nombre app. Para ejecutar el programa se escribe app.

Diffpack, también incluye compilación optimizada, la cual se logra con

Make MODE=opt.

Por otra parte, Diffpack permite ahorrar memoria y evitar operaciones matemáticas innecesarias cuando un problema genera una matriz tridiagonal (es importante ahorrar memoria y tiempo de ejecución cuando se habla de cómputo); se le indica a Diffpack que la matriz es tridiagonal con la instrucción

MatTri A(n)

Un objeto de MatTri asume que los índices de la matriz son –1, 0 y 1, donde –1 denota la subdiagonal, 0 la diagonal principal y 1 la superdiagonal.

En Diffpack hay herramientas para administrar gráficas de curvas generadas por problemas específicos, la visualización es necesaria porque facilita la interpretación de los resultados numéricos.

La ventaja de Diffpack y C++ es que permiten la reutilización, es decir únicamente se implantan los detalles del problema en particular, lo cual disminuye el esfuerzo de codificación y aumenta la confiabilidad del código ya que los cálculos se efectúan en módulos generales que ya han sido probados en otros contextos.

4.2) Diffpack y el elemento finito

El método del elemento finito es una herramienta general para resolver ecuaciones diferenciales parciales, entre otras aplicaciones. La herramienta principal de Diffpack para desarrollar software de elemento finito es la clase FEM. Esta clase contiene datos y versiones por defecto de algoritmos que son empleados frecuentemente en programas de elementos finitos. Para crear una aplicación, el

usuario debe suministrar una clase dependiente del problema, la cual debe ser derivada de FEM.

Algunos elementos 2D y sus funciones base correspondientes con que cuenta Diffpack son:

ElmTensorProd1: Elementos de producto tensor multidimensional, donde las funciones base subyacentes unidimensionales son lineales (el número uno indica que son de primer orden). El nombre debe ser proporcionado por el número de dimensiones para especificar completamente al elemento.

ElmTensorProd2: Igual que el anterior, sólo que las funciones base inferiores unidimensionales son cuadráticas.

ElmTensorProd: Idéntico a los elementos previos, mas el orden de la función base subyacente unidimensional es un parámetro requerido por el objeto (en adición al número de dimensiones espaciales).

ElmB2n1D: Elemento de forma cuadrada con dos nodos en 1D.

ElmB4n2D: Elemento cuadrilátero bilineal en 2D, ampliamente utilizado.

ElmB9n2D: Elemento cuadrado con nueve nodos en 2D.

ElmT3n2D: Elemento de forma triangular, con tres nodos en 2D.

Otros elementos en 2d son ElmB8n2D y ElmT6n2D.

Algunos elementos en 1D y 3D son

ElmB2n1D: Elemento estándar lineal en 1D con dos nodos.

ElmB3n1D:Elemento cuadrático en 1D con tres nodos.

ElmB8n3D: Elemento estándar trilinear.

ElmB20n3D: Polinomio de segundo orden sobre 3D con veinte nodos.

ElmTensorProd* puede ser aplicado en 1D y 3D o cualquier número de dimensiones: Polinomio de segundo orden completo en 3D.

Diffpack puede manipular geometrías más complicadas a través del preprocesador. Un preprocesador es utilizado para generar los datos de entrada geométricos requeridos por los métodos de elemento finito. El dominio es dividido por el preprocesador en una malla de elementos simples y pequeños, generalmente triángulos o cuadriláteros para 2D y tetraedros o hexaedros para 3D. Los preprocesadores en Diffpack son implantados con una interfaz común para los usuarios y para los programadores utilizando técnicas de programación orientada a objetos. El propósito de un preprocesador es dividir una geometría dada en elementos finitos, ello es, para calcular la estructura de datos asociada con una rejilla de elementos finitos en la geometría. La salida es habitualmente un objeto rejilla cuando la entrada es alguna información sobre la geometría del dominio y la partición del dominio en elementos finitos.

Las rejillas unidimensionales son descritas convenientemente por medio de un arreglo de coordenadas unidimensional. En 2D y 3D se requieren estructuras de datos más sofisticadas. Es normal en problemas de elemento finito contar con algunos arreglos para describir algunas características de la malla, tales como

* Las coordenadas de los puntos nodales
* Los números de nodos globales de los nodos en cada elemento
* Un número de materiales asociado con cada elemento
* El tipo de cada elemento finito (lineal, cuadrático, trilineal, etc)
* Los nodos que están sujetos a ciertas condiciones de frontera.

Para representar esta información se cuenta con la clase GridFE. El objeto grid siempre es generado por un preprocesador y después es ligado a algunos campos de elementos finitos. El empleo futuro está oculto al programador.

Puede ser favorable marcar algunos nodos en la rejilla, donde las condiciones de frontera serán aplicadas; es claro que la clase GridFE soporta la marcación de nodos: un nodo puede ser señalado llamando únicamente al indicador de frontera; con el empleo de la clase Boundary se puede definir un conjunto de indicadores de frontera. Cada indicador posee un nombre y puede estar encendido o apagado en cada nodo. Generalmente los programas de aplicación tienen una convención que relaciona a los indicadores de frontera con las condiciones de frontera. Es necesario recordar que la selección de indicadores de frontera está en parte gobernada por las reglas de programación de la aplicación y en parte por el usuario. Los nombres de los indicadores de frontera son elegidos libremente por el usuario.

La marcación de las partes de frontera por indicadores de frontera es usualmente ejecutada en un preprocesador. El preprocesador más simple que incluye Diffpack es PreproBox, el cual asigna automáticamente a los indicadores de frontera; otros preprocesadores requieren que el usuario especifique información del indicador de frontera en un archivo. El nombre de un indicador puede obtenerse usando la función getBoIndName. Las funciones BoSide y BoNode retornan un valor verdadero si un lado en un elemento o en un nodo tiene el indicador de frontera encendido. La función BoNodes regresa un valor verdadero si algún nodo en un elemento está marcado con un indicador de frontera dado.

El dominio puede ser particionado en subdominios, los cuales pueden corresponder a diferentes medios físicos.

Si todos los elementos son isoparamétricos, la geometría del elemento es suficiente para determinar del tipo de elemento finito.

Diffpack contiene una jerarquía de clases para los métodos preprocesadores, con la clase Prepro como la clase base. El preprocesador más simple es el preprocesador box, representado por la clase PreProBox. Esta clase puede crear una malla en forma de caja en un número arbitrario de dimensiones. Los elementos finitos en la malla son de tipo multilineal o multicuadrático. La familia de elementos multilineales consisten de elementos lineales en 1D, cuadriláteros bilineales en 2D y cajas trilineales en 3D. La clase PreProBox puede para 2D y 3D generar mallas consistentes de triángulos o tetraedros. Cada elemento multilineal o multicuadrático es posteriormente dividido en triángulos o tetraedros.

Los preprocesadores de Diffpack requieren dos tipos de información, una que refleje la geometría del dominio y otra, la partición del dominio en elementos finitos. La geometría es representada por un objeto en la clase Geometry hierarchy donde la clase Partition hierarchy representa la partición de geometrías. En una clase preprocesadora hay un objeto Geometry y uno Partition.

Las geometrías más complicadas pueden ser discretizadas empleando el preprocesador superelemento, representado por la clase PreproSupElSet. Este preprocesador primeramente divide el dominio en unos cuantos elementos grandes, llamados superelementos y después lleva a cabo la discretización de cada superelemento y calcula la malla de todos los elementos finitos. La división de la geometría en superelementos es realizada manualmente por el usuario y el resto del proceso es automático. El preprocesador puede generar, para 2D y 3D, mallas consistentes de elementos bilineales, trilineales, bicuadráticos, tricuadráticos,

cuadriláteros con ocho y veinte nodos. Mallas de triángulos para 2D y tetraedros para 3D pueden ser generados. La principal ventaja de este preprocesador es que el usuario tiene control total sobre el proceso de partición y que varios tipos de elementos pueden ser aplicados.

La generación de la malla o triangulación empleando algoritmos geométricos eficientes ha sido estudiada por muchos investigadores durante varios años, y algún software ha sido desarrollado para la triangulación, entre ellos se encuentra GEOMPACK, el cual es un software matemático escrito en Fortran 77 estándar; GEOM significa Generation Of two-and three dimensional meshes (Generación de mallas bi y tridimensionales); este paquete fue desarrollado por Barry Joe[2]. El paquete contiene subrutinas para construir triangulaciones de Delaunay bidimensionales y tridimensionales, descomposición de una región poligonal general en polígonos simples o convexos, construcción del polígono visible de un polígono simple desde un punto y otras aplicaciones geométricas simples. Cabe recordar que GEOMPACK solamente puede generar mallas consistentes en triángulos en 2D y tetraedros en 3D.

Para acceder a las utilerías de GEOMPACK se usa una subclase denominada PreproGeomPack en la jerarquía Prepro. Básicamente el empleo de GEOMPACK en Diffpack consiste en preparar los archivos de datos de entrada para la geometría y la partición del dominio, y Diffpack únicamente traducirá esta información a un archivo con un formato que GEOMPACK pueda leer. Después el programa GEOMPACK será ejecutado por la clase PreproGeomPack, la salida proporcionada por GEOMPACK será leída, interpretada y hecha un objeto GridFE.

Diffpack permite también incorporar preprocesadores comerciales o externos a su sistema, para ello se declara una clase preprocesadora que se comunique con los

preprocesadores externos. Habitualmente ésto es efectuado al iniciar el preprocesador por el shell de comandos de Unix y leyendo el archivo de resultados conteniendo la malla de elementos finitos. Muchos preprocesadores comerciales están enfocados a problemas de ingeniería estructural y producen un archivo de salida conteniendo la rejilla y datos físicos. La función del preprocesador generateMesh debe leer este archivo, inicializar el objeto GridFE y posiblemente cargar datos físicos en un objeto de una clase particular construida para soportar los datos físicos del preprocesador. En Diffpack, los preprocesadores significan construir solamente la geometría de la malla, es decir, no están involucrados con datos de entrada físicos. Este principio es motivado por la idea de generalización de los problemas.

La convención de Diffpack para numerar los nodos y los lados de los elementos finitos en la jerarquía ElmDef se muestra en las figuras 4.1 a 4.12

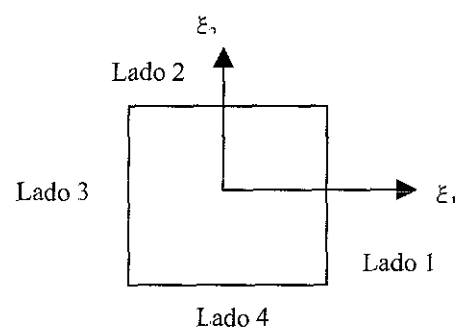


FIGURA 4.1 NUMERACIÓN DE LOS LADOS DE ELEMENTOS BIDIMENSIONALES

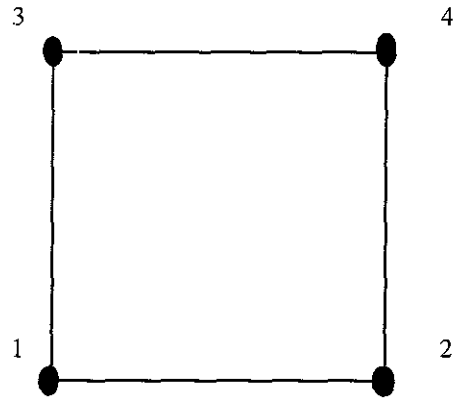


FIGURA 4.2 ElmB4n2D

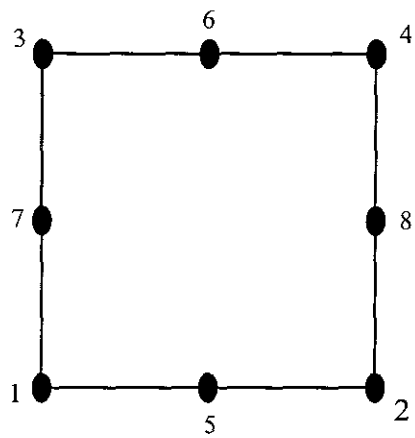


FIGURA 4.3 ElmB8n2D

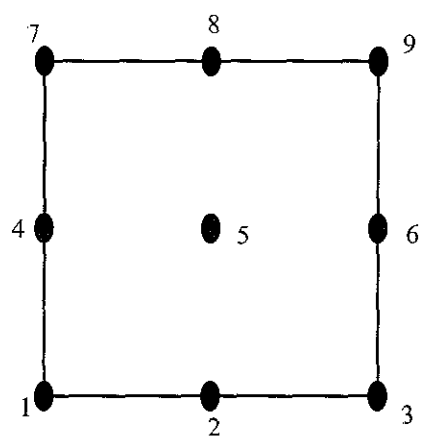


FIGURA 4.4 ElmB9n2D

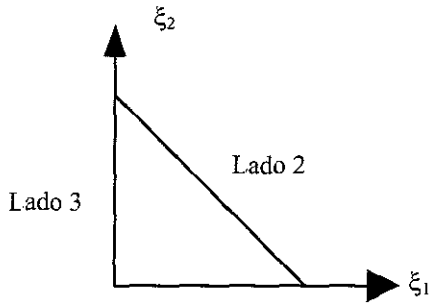


FIGURA 4.5 NUMERACIÓN DE TRIÁNGULOS BIDIMENSIONALES

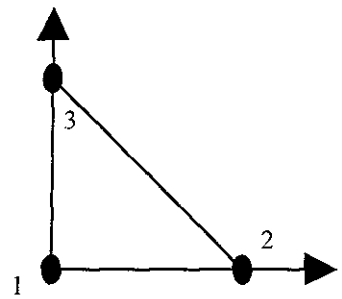


FIGURA 4.6 ElmT3n2D

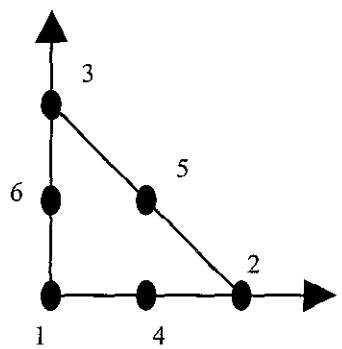
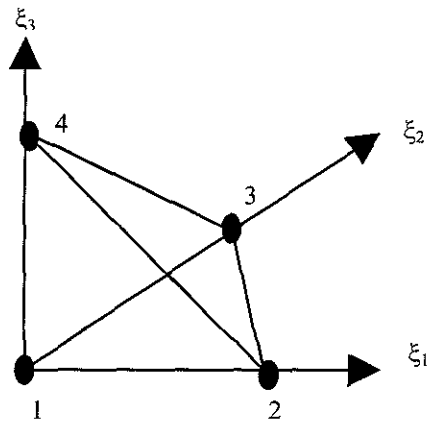
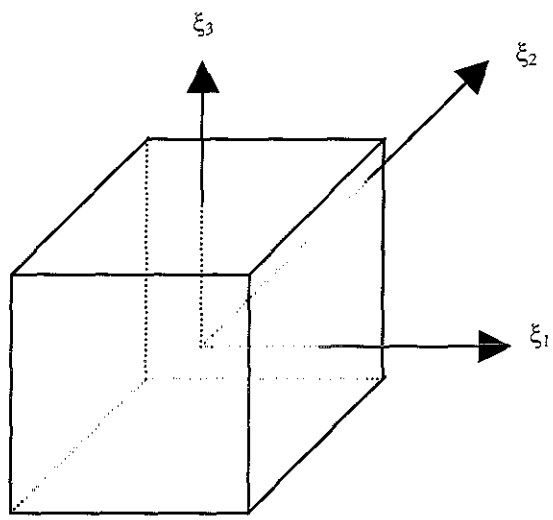


FIGURA 4.7 ElmT6n2D



Lado 1 = $\xi_1\xi_3$ – plano
Lado 2 = $(1-\xi_1-\xi_2-\xi_3)$ – plano
Lado 3 = $\xi_2\xi_3$ – plano
Lado 4 = $\xi_1\xi_2$ – plano

FIGURA 4.8 ElmT4n3D



Lado i < 4: (ξ_i = 1) - plano
Lado i > 3: (ξ_i = -1) - plano

FIGURA 4.9 NUMERACIÓN PARA ELEMENTOS TRIDIMENSIONALES

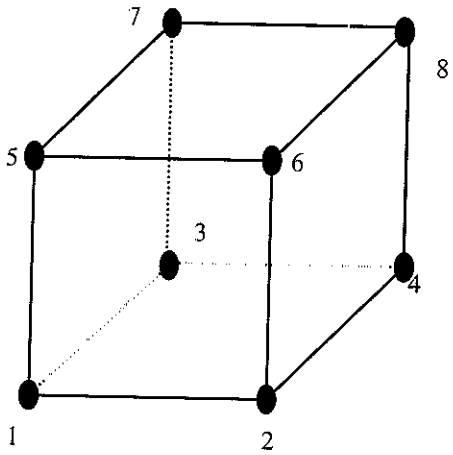


FIGURA 4.10 ElmB8n3D

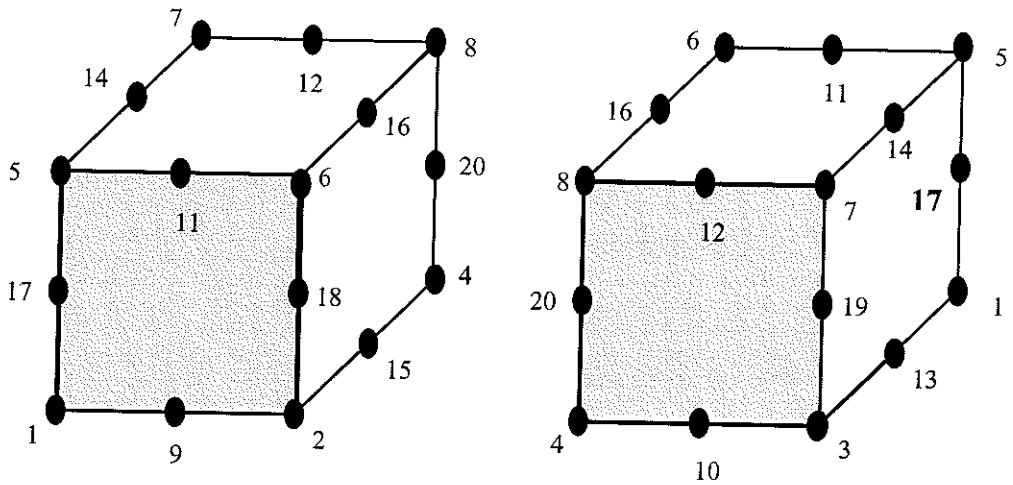


FIGURA 4.11 ElmB20n3D

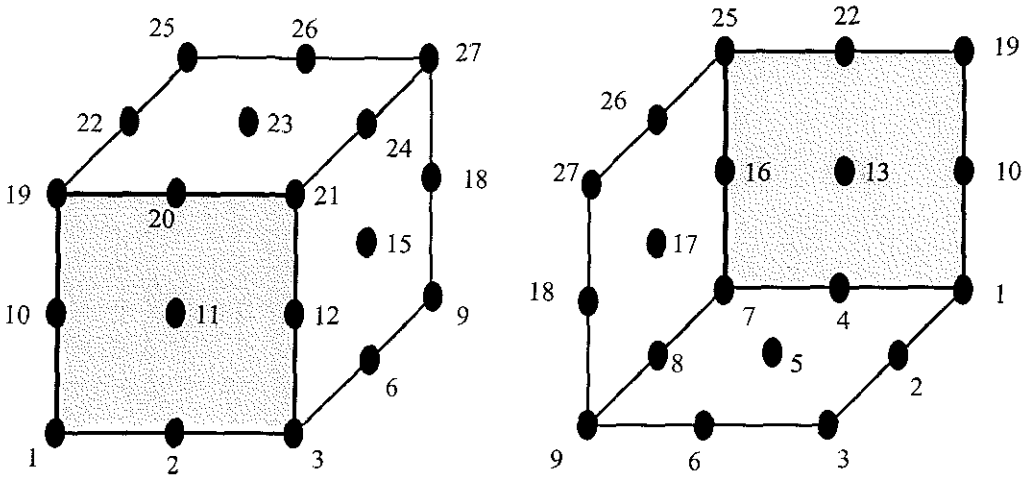


FIGURA 4.12 ElmB27n3D

La numeración de la malla, se realiza como se indica en la figura 4.13.

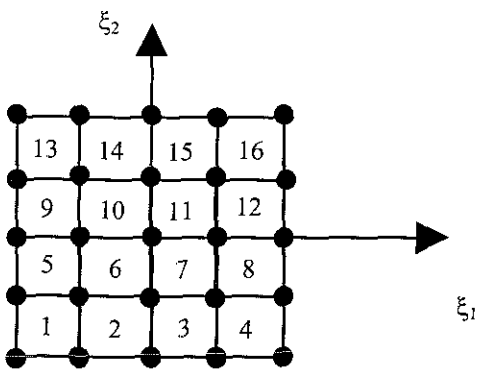


FIGURA 4.13 NUMERACIÓN DE LA MALLA EN DIFFPACK

Y la numeración de los nodos que integran a un elemento, se muestra en la figura 4.14

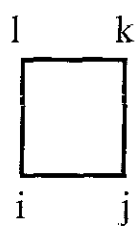


FIGURA 4.14 NODOS QUE CONSTITUYEN A UN ELEMENTO

Además se debe cumplir lo siguiente :

$$j = i + 1$$

$$l = k - 1$$

$$l - i = k - j = B$$

Donde B es el ancho de la semibanda (calculado por la función DegfreeFE) de la matriz simétrica de coeficientes.

4.3) Aplicación

El programa está limitado al análisis de cuerpos isotrópicos, con comportamiento lineal, elásticos, deformación plana o esfuerzo plano. Sólo se puede acomodar un caso de carga para cada problema. Los elementos usados son cuadriláteros y / o triángulos de deformación constante.

Para capturar los datos se deben respetar las siguientes restricciones :

a) Los datos deben introducirse en el orden en que se indica en el programa.

b) Las unidades empleadas deben ser consistentes.

Debido a la naturaleza del algoritmo generador de la malla de la estructura, no es necesario introducir manualmente todos los nodos. Basta con diseñar la numeración de los nodos de acuerdo a la convención establecida previamente. Con ésto, sólo se indica el número de dimensiones, el tipo de elemento a emplear, el número de elementos en X y en Y, el dominio de la aplicación y el número de

materiales asociados con cada elemento, todo lo anterior se efectúa gracias a la clase GidFE. Por ejemplo si se proporciona

"d=2, [0,1]x[0,1]

d=2, elm_tp= ElmB4n2D div= [5,5] grading= [1,1]

hará una malla en 2D en el dominio [0,1] con (5+1)x(5+1) nodos y elementos bilineales. El comando grading=[1,1] indica que la malla es uniforme.

Otro tipo de entrada sería

"d= 2, subdomains=2

subdomain=1, [0,0.5]x[0,0.5] elm_tp= ElmB4n2D div= [3,3] grading= [1,1]

subdomain=2, [0.5,1]x[0.5,1] elm_tp= ElmT3n2D div= [3,3] grading= [1,1]

la cual construirá una malla en 2D que contendrá dos tipos de materiales. El primer material estará en el dominio [0,0.5]x[0,0.5] con (3+1)x(3+1) nodos y elementos bilineales. El segundo material abarcará el dominio [0.5,1]x[0.5,1] con (3+1)x(3+1) nodos y elementos triangulares. El comando grading=[1,1] indica que la malla es uniforme en ambos tipos de materiales.

Es conveniente marcar algunos nodos en la rejilla donde se aplicarán las condiciones de frontera; esto se lleva a cabo cuando hay cargas externas. Un nodo puede ser marcado con el boundary indicator (indicador de fronteras) de la clase GridFE.

Una vez que se han obtenido los resultados se procede a interpretarlos. Para placas de acero y dependiendo del tipo de que se trate, existen valores dados en las normas (**API** , **ANSI**, **ISO** , etc.) como máximos tanto para los desplazamientos como para los esfuerzos.

La primer pantalla de salida muestra para cada nodo de la malla, los desplazamientos En la pantalla de salida 2 se presenta para cada elemento en su centroide, los esfuerzos correspondientes tanto en la dirección X como en Y; posteriormente se exhiben los números de nodos en donde ocurren los máximos desplazamientos en ambas direcciones.

Es necesario examinar minuciosamente los resultados, ya que si detectan desplazamientos en algunos nodos, que excedan los valores prescritos en las normas, entonces quizá ya no valga la pena examinar los siguientes resultados, ya que un desplazamiento mayor que el máximo tolerado, ocasionará riesgos para el buen funcionamiento de la estructura de la que forma parte la placa.

Si todos los desplazamientos en la pantalla de salida 1 están por debajo de los valores especificados en las normas, se procede a examinar la pantalla 2. Aquí se comparan los valores de esfuerzo constante e intensidades de esfuerzo con las especificaciones del material que se está tratando. Cada material tiene sus propios valores máximos de esfuerzo cortantes e intensidades permitidos. Si existen valores que sobrepasen los indicados en la especificación, entonces se sabrá con certeza en que región sufrirá colapso la placa estudiada.

Para el acero, el desplazamiento máximo aceptable (en números redondos) es aproximadamente $\frac{1}{32}$", no importando el espesor de la placa.

Para esfuerzos, aquí no se puede mostrar algún ejemplo ya que los valores dependen de las dimensiones de la placa en cuestión.

Si los resultados obtenidos en la primera corrida no son satisfactorios (si está permitido por las condiciones técnicas, administrativas y económicas de la empresa), es necesario efectuar corridas posteriores modificando los valores de las variables

exógenas y/o las de estado. Se deben realizar tantas corridas como número de combinaciones discretas se hayan calculado, hasta lograr un rango de resultados que ofrezcan las condiciones óptimas globales.

Si los resultados son satisfactorios en la primera corrida, es necesario llevar a cabo de todas maneras un número tal de corridas igual al número de combinaciones estimadas con antelación, para lograr el rango de resultados necesarios para llegar a tener una buena toma de decisiones.

Cuando ya se tiene el rango de resultados, se procede a graficar para todos los casos, cargas-esfuerzos, cargas-deformaciones, indicando los valores de espesores y dimensiones generales para estas gráficas. La decisión tomada a partir de las gráficas son para aprobar el diseño de la placa; sin embargo, habrá que verificar que este diseño (por condiciones de carga, propiedades del material y dimensiones) esté acorde con los procesos de manufactura, producción, adquisición de materiales, recursos económicos, utilidades, procesos de control de calidad, etc.

Si alguna de estas condiciones se contrapone al diseño obtenido por MEF se hace necesario volver a diseñar y a realizar otra vez el número de corridas necesarias, hasta que todos los requisitos de la empresa estén conciliados; es decir, se trabaja bajo el enfoque sistémico.

En el apéndice A se presenta un caso práctico; el código de la rutina principal para esta aplicación, así como otros ejemplos desarrollados en Diffpack y los resultados obtenidos, se encuentran en el disco flexible adjunto.

CONCLUSIONES

CONCLUSIONES

1. Al término de este trabajo de investigación, se concluye que los objetivos planteados se cumplieron satisfactoriamente, puesto que la TOO condujo a resultados óptimos en el área aquí estudiada: el análisis de esfuerzos en estructuras de acero (placas).
2. La TOO permite la reutilización de mecanismos, lo que conduce a un ahorro considerable en expresiones (codificación). Ello significa que el sistema se simplifica, es más flexible al cambio y es capaz de evolucionar a través del tiempo, puesto que su diseño está basado en clases y objetos.
3. El punto anterior contribuyó a que la metodología para solucionar problemas de análisis de esfuerzos fuese sencilla y comprensible, y a que el mantenimiento del sistema sea menos complejo.
4. El software empleado para construir el módulo fueron los lenguajes de programación C++ y Diffpack, ambos orientados a objetos. El Diffpack proveyó las utilerías necesarias para trabajar con elementos finitos y, con C++ se construyó y codificó la clase derivada para la aplicación tratada. También con dicho lenguaje se construyeron las interfaces necesarias.
5. Lo anterior permitió una disminución importante en el esfuerzo de codificación y un ahorro en memoria, aunque este módulo se ejecutó en una supercomputadora, si se dispone de una licencia para uso de Diffpack, éste puede ser empleado en computadoras pequeñas.
6. El tiempo de proceso fue mucho menor (máximo de cinco horas) que el reportado por los métodos tradicionales. Estos cálculos eran anteriormente llevados a cabo por procedimientos de resistencia de materiales y de diseño mecánico, los cuales dada su naturaleza y la dificultad involucrada, consumían hasta seis semanas de labor y la participación de profesionistas de diferentes especialidades.

7. El módulo desarrollado en este trabajo disminuye la probabilidad de cometer errores de cálculo en la aplicación de técnicas, etc. (errores humanos), y los errores que pueden presentarse con esta metodología serían de redondeo y en la captura de datos.
8. El logro principal de esta investigación, es la confiabilidad y exactitud del procedimiento. La exactitud en su empleo fue del 99%, y la confiabilidad del 98.5%, ya que al realizar corridas paralelas con módulos similares de Ansis, Cosmos y Nastran (los paquetes comerciales más empleados), se obtuvieron los mismos resultados.
9. La reducción en el tiempo de cálculo permite suponer un decremento en los costos correspondientes.

APÉNDICE A. CASO PRÁCTICO

A.1 CASO PRÁCTICO

Se desea estudiar deformaciones y esfuerzos a que estaría sujeta una serie de placas rectangulares, que deberán contener presión hidrostática interna; el universo a analizar está comprendido según las siguientes condiciones (ver figura A.1, la cual muestra la vista superior del cuerpo de la válvula, para significado de variables y forma geométrica).

caso	A	B	T_1	T_2	A/B	T_1/T_2
1	18	7	3.7	1.25	2.57	2.96
2	18	7	2.65	1.25	2.57	2.12
3	18	7	2.13	1.25	2.57	1.704
4	18	5	3.7	1.25	3.6	2.96
5	18	5	2.65	1.25	3.6	2.12
6	18	5	2.13	1.25	3.6	1.704
7	18	3	3.7	1.25	6.0	2.96
8	18	3	2.65	1.25	6.0	2.12
9	18	3	2.13	1.25	6.0	1.704

En todos los nueve casos anteriores, la presión hidrostática interna se considerará como 300 lb/pulg^2 manométricas.

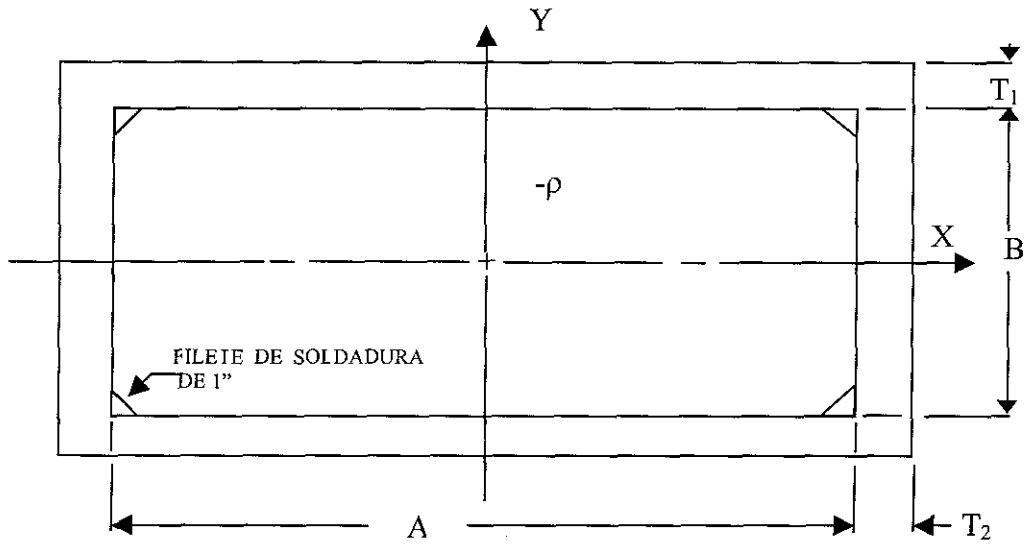


FIGURA A.1

Para solucionar este problema, se emplearon elementos isoparamétricos, 2D, rectangulares y triangulares. Para el análisis es suficiente usar una cuarta parte de la sección.

La discretización del continuo se aprecia en la figura A.2., que exhibe una cuarta parte del continuo.

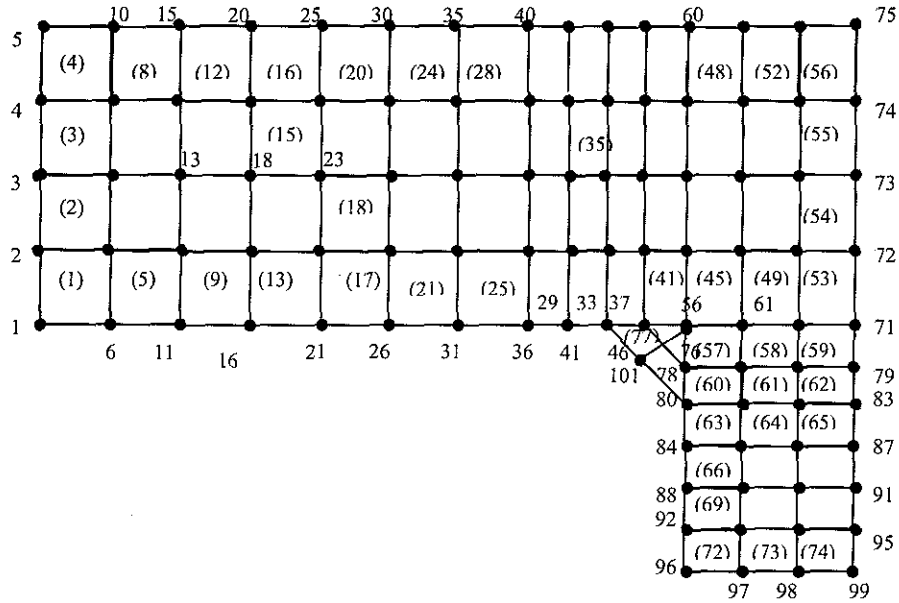


FIGURA A.2 DISCRETIZACIÓN E IDENTIFICACIÓN DE ELEMENTOS Y NODOS

Nótese que la numeración de elementos está entre paréntesis.

Los resultados obtenidos se muestran en las figuras siguientes y en la tabla ulterior.

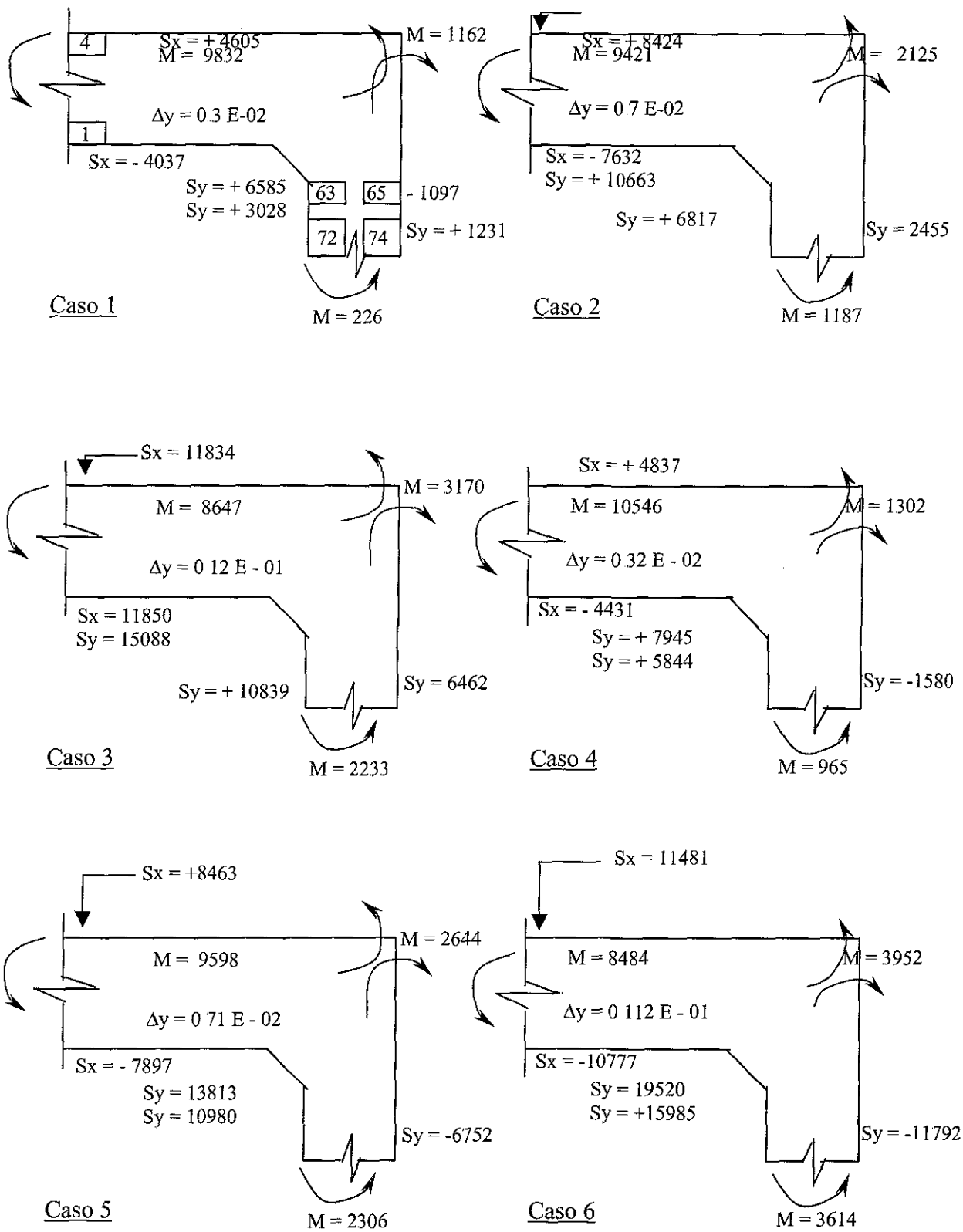


FIGURA A.3 RESUMEN DE LA INFORMACIÓN OBTENIDA

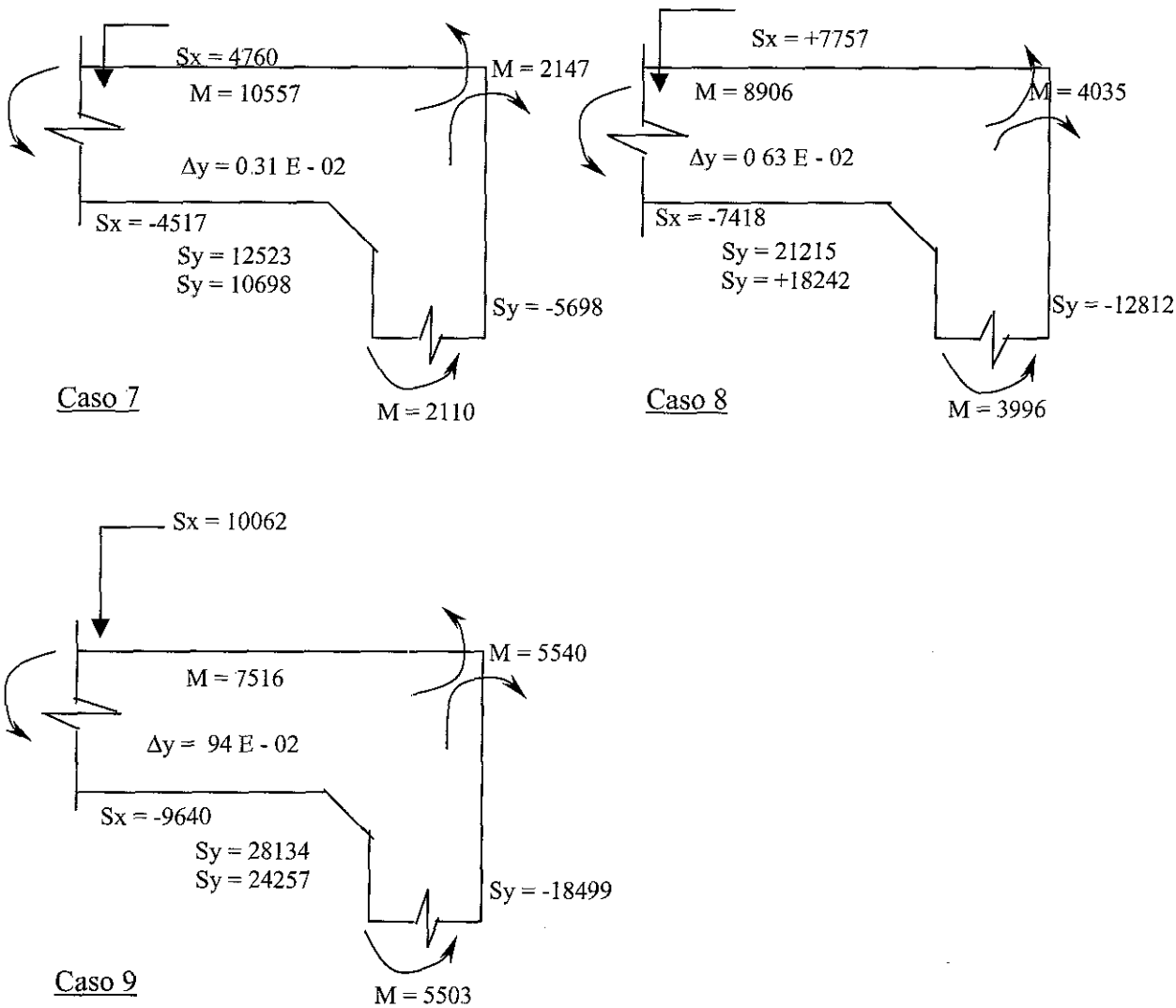


FIGURA A.3 RESUMEN DE LA INFORMACIÓN OBTENIDA (CONTINUACIÓN)

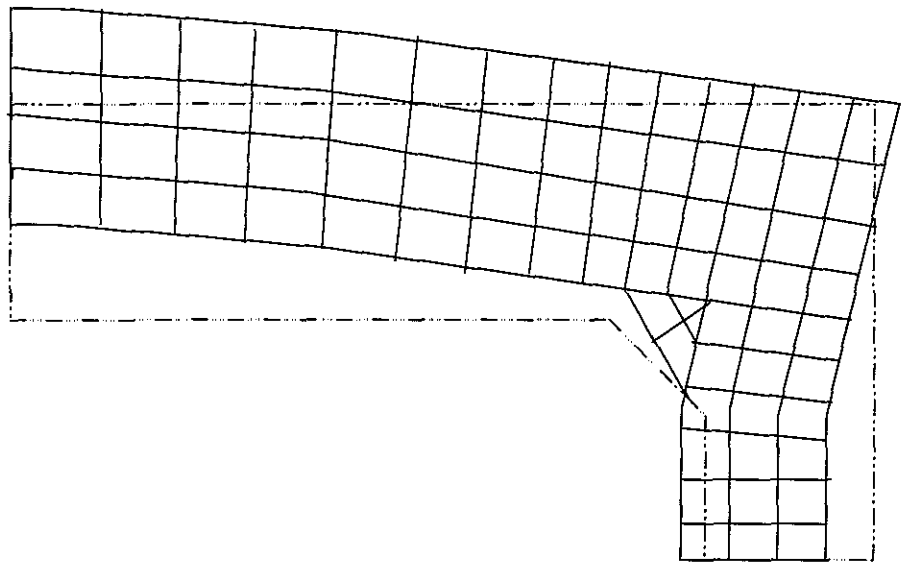


FIGURA A.4 DEFORMACIONES

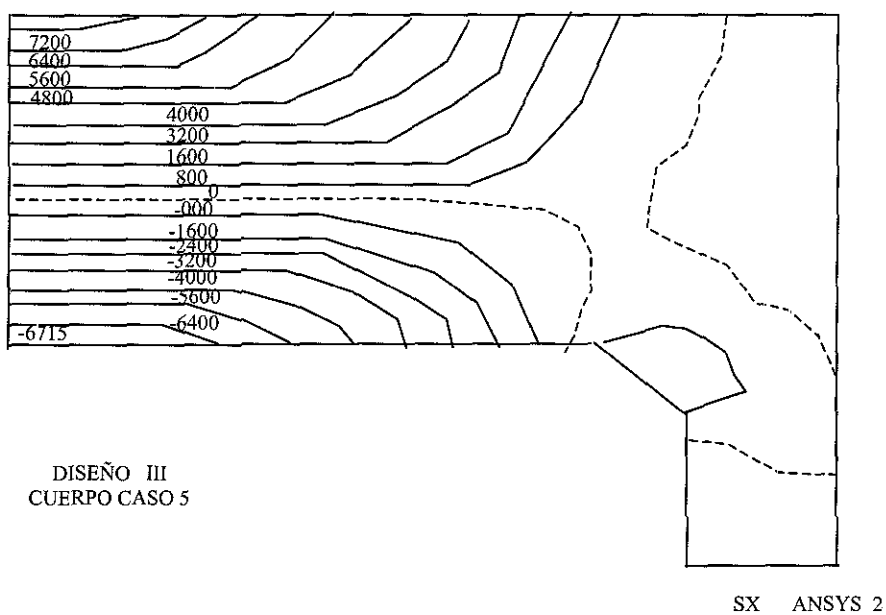


FIGURA A.5 ESFUERZOS S/G EJE X

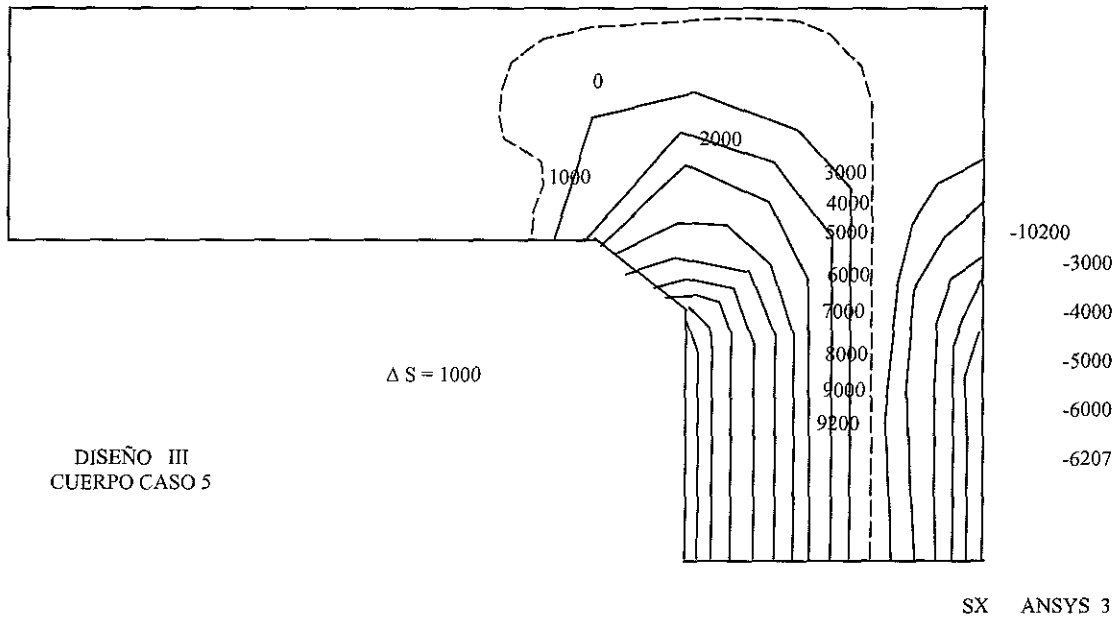


FIGURA A.6 ESFUERZOS S/G EJE Y

Caso N°	*Momento de Esquina S / G E. F.*	*Valor de* K_1
1	1162	0 6302
2	2125	0 7920
3	3170	0.8847
4	1302	0 8476
5	2644	0.9888
6	3952	1.0438
7	2147	1.2599
8	4035	1.2348
9	5540	1.2163

Graficando los nueve pasos, según los parámetros A/B, T_1/T_2 y K, se obtiene la figura A.7.

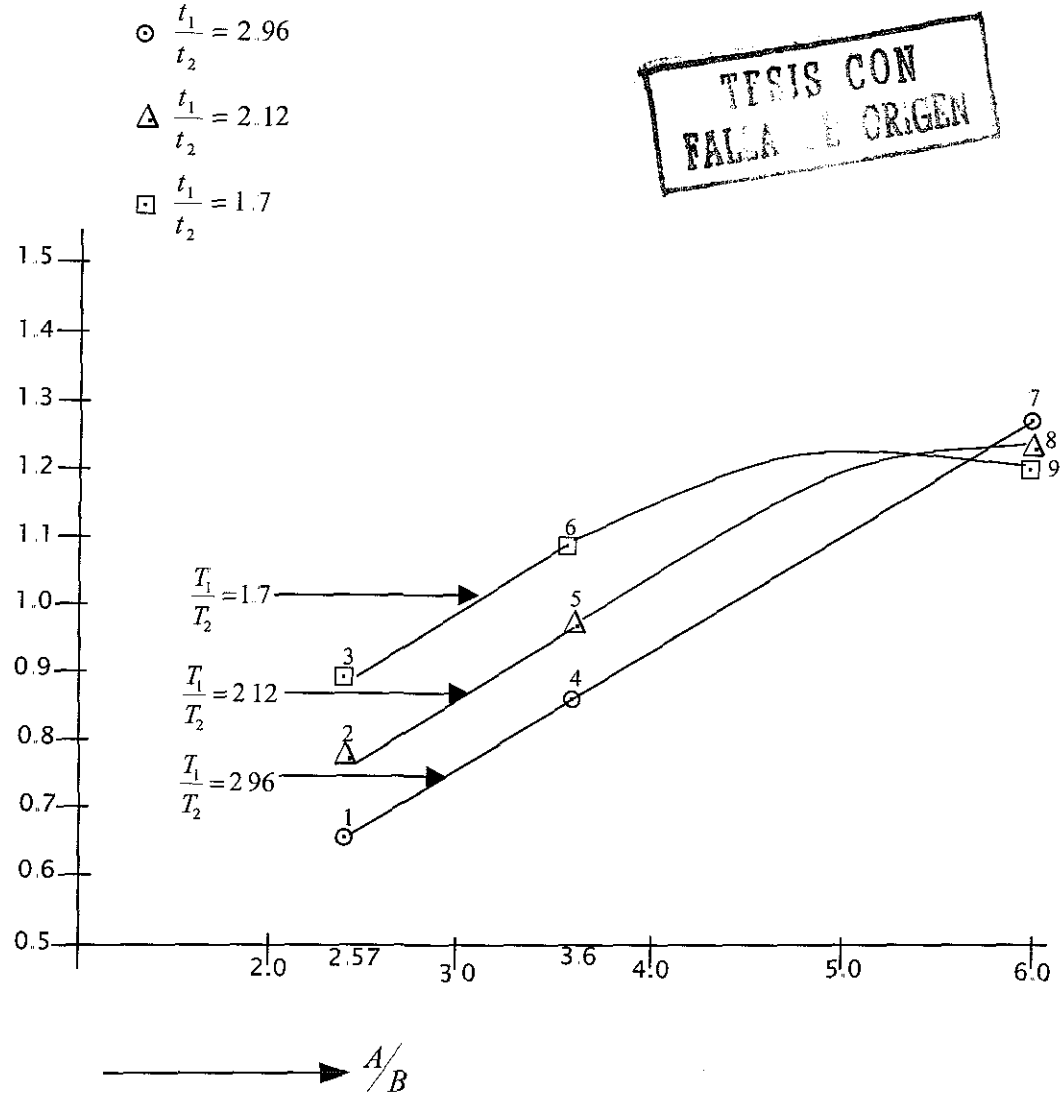


FIGURA A.7 FRONTERAS DEL UNIVERSO ESTUDIADO

En la siguiente tabla se exhibe el grado de exactitud logrado, comparado con los valores que reportó el MEF.

Caso Nº	Momento de Esquina			Esfuerzos en Elementos (63)		
	Por AEF	Por Cálculo corregido	Exactitud	Por AEF	Por Cálculo corregido	Exactitud
1	1162	1162	1.000	6585	6557	0.996
2	2125	2123	0.999	10663	10626	0.997
3	3170	3160	0.997	15088	15008	0.995
4	1302	1297	0.996	7945	7926	0.998
5	2644	2642	0.999	13813	13786	0.998
6	3952	3933	0.995	19520	19508	0.999
7	2147	2145	0.999	12523	12476	0.996
8	4035	4024	0.997	21215	21144	0.996
9	5540	5523	0.997	28134	28041	0.997

Nota: Este ejemplo de aplicación industrial es una de las bases para el diseño III de los cuerpos de válvulas de línea de FIP S.A. de C.V.

APÉNDICE B. TABLAS MÁS IMPORTANTES

MATERIAL	MÓDULO DE ELASTICIDAD, E		MÓDULO DE RIGIDEZ, G	
	Mpsi	GPa	Mpsi	GPa
Abeto Douglas	1.6	11.0	0.6	4.5
Acero al carbono	30.0	207.0	11.5	79.3
Acero inoxidable	27.6	190.0	10.6	73.1
Acero níquel	30.0	207.0	11.5	79.3
Aluminio (todas las aleaciones)	10.3	71.0	3.80	26.2
Bronce fosforado	16.1	111.0	6.0	41.4
Cobre	17.2	119.0	6.49	44.7
Cobre al berilio	18.0	124.0	7.0	48.3
Hierro fundido gris	14.5	100.0	6.0	41.4
Inconel	31.0	214.0	11.0	75.8
Latón	15.4	106.0	5.82	40.1
Magnesio	6.5	44.8	2.4	16.5
Molibdeno	48.0	331.0	17.0	117.0
Monel	26.0	179.0	9.5	65.5
Plata níquel	18.5	127.0	7.0	48.3
Plomo	5.3	36.5	1.9	13.1
Vidrio	6.7	46.2	2.7	18.6

TABLA B.1 MÓDULO DE ELASTICIDAD Y RIGIDEZ PARA DIVERSOS MATERIALES

MATERIAL	RELACIÓN DE POISSON
Abeto Douglas	0.330
Acero al carbono	0.292
Acero inoxidable	0.305
Acero níquel	0.291
Aluminio (todas las aleaciones)	0.334
Bronce fosforado	0.349
Cobre	0.326
Cobre al berilio	0.285
Hierro fundido gris	0.211
Inconel	0.290
Latón	0.324
Magnesio	0.350
Molibdeno	0.307
Monel	0.320
Plata níquel	0.322
Plomo	0.425
Vidrio	0.245

TABLA B.2 RELACIÓN DE POISSON PARA DIFERENTES MATERIALES

BIBLIOGRAFÍA

BIBLIOGRAFÍA

ARGYNS, J. H.

Computer Aided Structural Analysis

Edit. ASKA

2a. ed.; Dinamarca, 1990

297 pág.

BARAN

Finite Element Method on Microcomputer

Edit. Prentice Hall

Estados Unidos de América, 1990

435 pág.

BOOCH, Grady

Object oriented design with application

Edit. Benjamin/Cummings Publishing Company Inc.

Estados Unidos de América, 1991

499 pág.

CARNAHAN, B. Luther y Wilkes J.

Applied Numerical Methods

Edit. John Wiley & sons

7a. ed.; Estados Unidos de América,1994

876 pág.

CLOUGH, R. W.

The Finite Element Method in Plane Stress Analysis

Edit. ASCE

5a. ed.; Estados Unidos de América, 1995

283 pág.

COAD, Peter and Edward Yourdan

Object oriented análisis

Edit. Yourdon Press, PTR Prentice may Building

Estados Unidos de América, 1997

545 pág.

COLNET, Napoli, et. al.

Lenguajes orintados a objetos

Edit. Academic Press

Estados Unidos de América, 1996

503 pág.

DAHLQUIST G.

Numerical Methods

Edit. Prentice Hall

4a. ed.; Estados Unidos de América, 1993

790 pág.

DESAI, Chandrakaut y Abel John

Introduction to the Finite Element Method

Edit. Van Nostrand Reinhold Company

9a. ed.; Suecia, 1995

578 pág.

ELLIS, MARGARET A., Bjarne Stroustrup

The annotated C++, Reference manual

Edit. Addison Wesley

Estados Unidos de América, 1992

579 pág.

FELIPPA, C. A.

The Finite Element Method in Solid Mechanics

Edit. American Mathematical Society

8a. ed.; Inglaterra, 1994

319 pág.

FOX, L.

Computing Methods for Scientists and Engineers

Edit. Oxford University Press

2a. ed.; Inglaterra, 1992

401 pág.

GALLAGHER, R. H.

Finite Element Analysis : Fundamentals

Edit. Prentice Hall

4a. ed.; Estados Unidos de América,1991

432 pág.

GERE, J. M.

Matrix Algebra for Engineers

Edit. Van Nostrand Reinhold Company

5a. ed.; Suecia, 1994

373 pág.

HILDEBRAND, F. B.

Methods of Applied Mathematics

Edit. Englewood Cliffs

Estados Unidos de América, 1989

499 pág.

IRONS, B. M.

Numerical Integration Applied to Finite Element Method

Edit. AIAAJ

Alemania, 1993

520 pág.

JENSEN, H. G. and Parks G.

Efficient Solutions for Linear Matrix Equations

Edit. ASCE

2a. ed.; Estados Unidos de América, 1996

188 pág.

KATRIB, Miguel

Programación orientada a objetos en C++

Edit. Infosys

México, 1997

496 pág.

MARCAL, P. U.

Finite Element Analysis Theory and Practice

Edit. University of Alabama

2a. ed.; Estados Unidos de América, 1990

434 pág.

MARTIN, H. C.

Plane Elasticity Problems and the Direct Stiffness Method

Edit. Trend Eng.

5a. ed.; Estados Unidos de América, 1995

316 pág.

MASE, George

Continuum Mechanics

Edit. Schaum's outline series

3a. ed.; Estados Unidos de América, 1989

1091 pág.

MC LEAN y Nelson

Engineering Mechanics

Edit. *Schaum publishing Co.*

2a. ed.; Estados de América, 1988

405 pág.

ODEN, J. T.

Finite Elements of Non - Linear Continua

Edit. Mc Graw Hill

Estados Unidos de américa, 1992

382 pág.

POPOV, E. P.

Introduction to Mechanics of Solids

Edit. Englewood Cliffs

3a. ed.; Estados Unidos de América, 1995

389 pág.

RUMBAUGH,, J.

Object oriented modelling and design

Edit. Prentice Hall/Englewood Cliffs

Estados Unidos de América, 1997

503 pág.

SEELY y Ensign

Mecánica Analítica para Ingenieros

Edit. U.T.E.H.A.

12a. ed.; México, 1993

458 pág.

SHIGLEY, Joseph y Mitchell, Larry

Manual de Diseño Mecánico Vol. I

Edit. Mc Graw Hill

4a. ed.; España, 1990

230 pág.

SHIGLEY, Joseph y Mitchell, Larry

Manual de Diseño Mecánico Vol. II

Edit. Mc Graw Hill

4a. ed.; España, 1990

230 pág.

SHIGLEY, Joseph y Mitchell, Larry

Manual de Diseño Mecánico Vol. III

Edit. Mc Graw Hill

4a. ed.; España, 1990

230 pág.

SHIGLEY, Joseph y Mitchell, Larry

Manual de Diseño Mecánico Vol. IV

Edit. Mc Graw Hill

4a. ed.; España, 1990

230 pág.

SPIEGEL, M.

Vector Analysis

Edit. Schaum publishing Co.

5a. ed.; Estados Unidos de América, 1991

867 pág.

TARG, I. C.

Structural Analysis by the Matrix Displacement Method

Edit. English Electric Aviation Rept

Estados Unidos de américa, 1988

179 pág.

TUMA y Munshi

Análisis Estructural Avanzado

Edit. Mc Graw Hill

5a. ed.; México, 1992

795 pág.

ZIENKIEWICZ, O.

The finite element method in engineering science

Edit. Mc Graw Hill

4a. ed.; Estados Unidos de América, 1994

650 pág.

REFERENCIAS

ARGILA, Carl. Finding and Keeping Good Objects. American programmer, Estados Unidos de América 1994. p.p 36-43

COOK, Steve and John Daniels. Object-Oriented Methods and the Great Object Myth. SIS, Estados Unidos de América 1994. p.p 13-18

DLUGOSZ, John. The Dark Side of OOP. American programmer, Estados Unidos de América 1994. p.p. 12-17

LANGTAGEN, H.P. et.al. Getting Started with Diffpack. The Diffpack Report Series, Sintef, Noruega enero de 1999. p.p.1-20

LANGTAGEN, H.P. et.al. Getting started with finite element programming in Diffpack. The Diffpack Report Series, Sintef, Noruega octubre de 2000. p.p.1-69

LANGTAGEN, H.P. et.al. Finite Element Preprocessors in Diffpack. The Diffpack Report Series, Sintef, Noruega octubre de 2000. p.p.1-57

LANGTAGEN, H.P. et.al.Details of Finite Element Programming in Diffpack . The Diffpack Report Series, Sintef, Noruega octubre de 2000. p.p.1-34

PARTRIDGE, Chris. Modelling the real world: Are classes abstrations or objects?. Rev.-Eng. Consulting. Estados Unidos de América 1994. p.p. 39-45