UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

INSTITUTO DE GEOFÍSICA

"Aplicaciones del Cómputo en Paralelo a la Modelación de Sistemas Terrestres"

T E S I S

QUE PARA OBTENER EL GRADO DE MAESTRO EN CIENCIAS DE LA TIERRA (MODELACIÓN DE SISTEMAS TERRESTRES)

P R E S E N T A:

MAT. ANTONIO CARRILLO LEDESMA

DIRECTOR DE TESIS:

DR. ISMAEL HERRERA REVILLA

2006

Dedico el presente trabajo con todo cariño a:

- Mi madre *Alfonsina* por haber hecho posible esto, después de pensar que yo era incorregible.
- Mi esposa *Josefina* por todo su apoyo y tiempo cedido para realizar la presente.
- Mi hijo *José Antonio* por mostrarme lo que realmente es la vida y ceder tanto de su tiempo de juegos conmigo para poder materializar este proyecto.
- Toda mi *familia y amigos* presentes y a los ya ausentes por mostrarme el camino en vida y así poder lograr este trabajo.

Agradecimientos:

Quiero agradecer a todas las personas que hicieron posible que el presente trabajo llegará a su fin:

- Al director de tesis:
  Ismael Herrera Revilla
  por todo el apoyo, tiempo y
  sus invaluables enseñanzas
- A los sinodales:
  Robert Yates
  Martín Díaz Viera
  Fabián García Nocetti
  Arón Jazcilevich Diamant
  por sus enseñanzas y comentarios
- A los investigadores:
  Alejandra Arciniega Ceballos
  Jorge Luís Ortega Arjona
  por todo el apoyo proporcionado

Índice

1. Introducción

1.1. Antecedentes

La necesidad de entender su entorno y anticiparse a los acontecimientos tiene raíces muy profundas en el ser humano. Desde la prehistoria, el hombre trató de predecir a la naturaleza, pues de ella dependía su supervivencia, para lo cual inicialmente nuestros antepasados utilizaron a la brujería, así como el pensamiento mágico y el religioso. Sin embargo, el medio más efectivo para predecir el comportamiento de la naturaleza es el método científico (o su antecesor el método empírico) y es por eso que este anhelo humano ancestral, a través de la historia, ha sido motor de la ciencia.

La maduración y el progreso de la predicción científica es, sin duda, el resultado del avance general de la ciencia, pero además ha habido elementos catalizadores esenciales sin los cuales esto no hubiera sido posible. La predicción científica, además de ser científica, es matemática y computacional. En la actualidad, cuando deseamos predecir el comportamiento de un sistema, los conocimientos científicos y tecnológicos se integran en modelos matemáticos los cuales se convierten en programas de cómputo que son ejecutados por las computadoras tanto secuenciales como paralelas (entenderemos por una arquitectura paralela a un conjunto de procesadores interconectados capaces de cooperar en la solución de un problema).

Aunque el sólo hecho de poseer la capacidad de predicción científica nos llena de satisfacción y orgullo, sin embargo, aún más trascendente es el hecho de que ella también es la base de una gran parte del extraordinario progreso material que la humanidad ha experimentado en épocas recientes. En efecto, nuestra facultad para predecir es una herramienta muy poderosa de la ingeniería, de la tecnología y de la ciencia misma, la cual, entre otras muchas cosas, nos ha permitido ampliar la disponibilidad de los recursos naturales y utilizarlos con mayor eficiencia.

Para resolver, por ejemplo, algún problema del área de ciencias e ingeniería (el movimiento de un fluido libre o de un fluido en un medio poroso) lo primero que debemos de hacer es formular un modelo matemático del problema a tratar. Esto se logra a través de los fundamentos de la física macroscópica los cuales son proporcionados por la 'teoría de los medios continuos' [13]. Con base en ella se introduce una formulación clara, general y sencilla de los modelos matemáticos de los sistemas continuos.

La formulación es tan sencilla y tan general, que los modelos básicos de sistemas tan complicados y diversos como la atmósfera, los océanos, los yacimientos petroleros, o los geotérmicos, se derivan por medio de la aplicación repetida de una sola ecuación diferencial: 'la ecuación diferencial de balance' [8].

Dicha formulación también es muy clara, pues en el modelo general no hay ninguna ambigüedad; en particular, todas las variables y parámetros que intervienen en él, están definidos de manera unívoca. En realidad, este modelo general de los sistemas continuos constituye una realización extraordinaria de los paradigmas del pensamiento matemático. El descubrimiento del hecho de que

los modelos matemáticos de los sistemas continuos, independientemente de su naturaleza y propiedades intrínsecas, pueden formularse por medio de balances, cuya idea básica no difiere mucho de los balances de la contabilidad financiera, fue el resultado de un largo proceso de perfeccionamiento en el que concurrieron una multitud de mentes brillantes.

Los modelos matemáticos de los sistemas continuos son ecuaciones diferenciales, las cuales son parciales (con valores iniciales y condiciones de frontera) para casi todos los sistemas de mayor interés en la ciencia y la ingeniería, o sistemas de tales ecuaciones. Salvo para los problemas más sencillos, no es posible obtener por métodos analíticos las soluciones de tales ecuaciones, que son las que permiten predecir el comportamiento de los sistemas.

La capacidad para formular los modelos matemáticos de sistemas complicados y de gran diversidad, es sin duda una contribución fundamental para el avance de la ciencia y sus aplicaciones, tal contribución quedaría incompleta y, debido a ello, sería poco fecunda, si no se hubiera desarrollado simultáneamente su complemento esencial: los métodos matemáticos y la computación electrónica. En cambio, la diversidad y complejidad de problemas que pueden ser tratados con métodos numéricos y computacionales es impresionante.

Los modelos de los sistemas continuos -es decir, sistemas físicos macroscópicos tales como los yacimientos petroleros, la atmósfera, los campos electromagnéticos, los océanos, los metalúrgicos, el aparato circulatorio de los seres humanos, la corteza terrestre, lo suelos y las cimentaciones, muchos sistemas ambientales, y muchos otros cuya enumeración ocuparía un espacio enormecontienen un gran número de grados libertad.

Por ello, la solución numérica por los esquemas tradicionales tipo diferencias finitas y elemento finito generan una discretización del problema, la cual es usada para generar sistemas de ecuaciones algebraicos [8]. Estos sistemas algebraicos en general son de gran tamaño para problemas reales, al ser estos algoritmos secuenciales su implantación suele hacerse en equipos secuenciales y por ello no es posible resolver muchos problemas que involucren el uso de una gran cantidad de memoria.

Actualmente para tratar de subsanar la limitante de procesar sólo en equipos secuenciales, se usan equipos paralelos para soportar algoritmos secuenciales mediante directivas de compilación, haciendo ineficiente su implantación en dichos equipos.

La computación en paralelo es una técnica que nos permite distribuir una gran carga computacional entre muchos procesadores. Y es bien sabido que una de las mayores dificultades del procesamiento en paralelo es la coordinación de las actividades de los diferentes procesadores y el intercambio de información entre los mismos [21]. Al usar métodos de descomposición de dominio conjuntamente con el cómputo en paralelo (supercomputadoras, clusters o grids) nos permite atacar una gran variedad de problemas que sin estas técnicas sería imposible hacerlo de manera eficiente.

1.2. Métodos de Descomposición de Dominio

Los métodos de descomposición de dominio introducen desde la formulación matemática del problema una separación natural de las tareas a realizar por el método y simplifican considerablemente la transmisión de información entre los subdominios [2]. En ellos, los sistemas físicos representados por su modelo son descompuestos en varios subdominios.

Los métodos de descomposición de dominio permiten tratar los problemas de tamaño considerable, empleando algoritmos paralelos en computadoras secuenciales y/o paralelas. Esto es posible ya que cualquier método de descomposición de dominio se basa en la suposición de que dado un dominio computacional Ω, este se puede particionar en subdominios $\Omega_i, i = 1, 2, ..., E$ entre los cuales puede o no existir traslape [1] y [2]. Entonces, el problema es reformulado en términos de cada subdominio (empleando algún método del discretización) obteniendo una familia de subproblemas de tamaño reducido independientes en principio entre si, y que están acoplados a través de la solución en la interfaz de los subdominios que es desconocida.

De esta manera, podemos clasificar de forma burda a los métodos de descomposición de dominio, como aquellos en que: existe traslape entre los subdominios y en los que no existe traslape. A la primera clase pertenece el método de Schwarz (en el cual el tamaño del traslape es importante en la convergencia del método) y a los de la segunda clase pertenecen los métodos del tipo subestructuración (en el cual los subdominios sólo tienen en común los nodos de la frontera interior).

Estos métodos son un paradigma natural usado por la comunidad de modeladores. Los sistemas físicos son descompuestos en dos o más subdominios contiguos basados en consideraciones fenomenológicas. Estas descomposiciones basadas en dominios físicos son reflejadas en la ingeniería de software del código correspondiente.

Así, mediante los métodos de descomposición de dominio, la programación orientada a objetos (que nos permite dividir en niveles la semántica de los sistemas complejos tratando así con las partes, más manejables que el todo, permitiendo su extensión y un mantenimiento más sencillo) y esquemas de paralelización que usan el paso de mensajes, es posible construir aplicaciones que coadyuven a la solución de problemas concomitantes en ciencia e ingeniería.

Esta metodología permite utilizar todas las capacidades del cómputo en paralelo (grids de decenas de clusters, cada uno con miles de procesadores interconectados por red con un creciente poder de cómputo medible en Peta Flops), así como el uso de una amplia memoria (ya sea distribuida y/o compartida del orden de Tera Bytes), permitiendo atacar una gran variedad de problemas que sin estas técnicas es imposible hacerlo de manera flexible y eficiente.

Pero hay que notar que existe una amplia gama de problemas que nos interesan resolver que superan las capacidades de cómputo actuales, ya sea por el tiempo requerido para su solución, por el consumo excesivo de memoria o ambos.

La lista de los métodos de descomposición de dominio y el tipo de problemas

que pueden ser atacados por estos, es grande y está en constante evolución [2], ya que se trata de encontrar un equilibrio entre la complejidad del método (aunada a la propia complejidad del modelo), la eficiencia en el consumo de los recursos computacionales y la precisión esperada en la solución encontrada por los diversos métodos y las arquitecturas paralelas en la que se implante.

1.3. Objetivos de la Tesis

Uno de los grandes retos del área de cómputo científico es poder analizar a priori una serie de consideraciones dictadas por factores externos al problema de interés que repercuten directamente en la forma de solucionar el problema, estas consideraciones influirán de manera decisiva en la implementación computacional de la solución numérica. Algunas de estas consideraciones son:

- Número de Procesadores Disponibles
- Tamaño y Tipo de Partición del Dominio
- Tiempo de Ejecución Predeterminado

Siendo común que ellas interactúan entre si, de forma tal que normalmente el encargado de la implementación computacional de la solución numérica tiene además de las complicaciones técnicas propias de la solución, el conciliarlas con dichas consideraciones.

Esto deja al implementador de la solución numérica con pocos grados de libertad para hacer de la aplicación computacional una herramienta eficiente y flexible que cumpla con los lineamientos establecidos a priori y permita también que esta sea adaptable a futuros cambios de especificaciones -algo común en ciencia e ingeniería-.

Para tratar de tener una idea clara de cómo afectan a la aplicación computacional estos factores, a continuación detallamos algunas consideraciones acerca de cada uno de ellos.

Número de Procesadores Disponibles El número de procesadores disponibles es una barrera en constante evolución pero las principales limitantes son económicas y tecnológicas, es común ahora contar con miles de procesadores interactuando de manera conjunta en grids, pero todo ese poder de cómputo creciente es aun insuficiente para las necesidades del cómputo científico.

Por otro lado la gran mayoría de los proyectos científicos y tecnológicos no cuentan con recursos computacionales grandes, ya que este es un recurso costoso y por ende muy limitado y en la mayoría de los casos no es el adecuado a las necesidades propias del problema. Así, para poder atacar el problema de forma eficiente es necesario adaptarse al equipo de cómputo disponible y tratar de conocer de antemano los factores que mermaran el rendimiento de la implementación computacional para buscar alternativas que mejoren la eficiencia de la implementación.

Tamaño y Tipo de Partición del Dominio Normalmente cuando se plantea un problema de sistemas continuos la determinación del dominio y tipo de malla a usar en la solución del mismo esta sujeta a restricciones fenomenológicas, por ello la malla deberá de adaptarse de la mejor forma posible para tratar de capturar los rasgos esenciales del fenómeno estudiado. Pero a la hora de la implementación computacional es común hacer adecuaciones a esta, para que pueda ser soportada por el equipo de cómputo y su ejecución sea en un tiempo razonable. Esto puede ser un problema sobre todos al ser implementada la solución usando cómputo paralelo, ya que comúnmente una elección de una malla no homogénea ocasionará problemas de mal balanceo de carga de trabajo entre los procesadores utilizados en la implementación computacional.

Tiempo de Ejecución Predeterminado Otro factor determinante es el tiempo de solución esperado de la implementación computacional del problema, esto es crítico en problemas de control en tiempo real, comunes en la ciencia e ingeniería actual. Por ello en estos casos es permisible el aumento en el número y capacidades del equipo de cómputo necesario en la implementación con el fin de lograr un tiempo de ejecución por debajo del máximo permisible dictado por las especificaciones propias del problema.

Todos estos factores influirán en la versión final implementada en la solución computacional de un problema particular, debiendo ser todas ellas sopesadas antes de tomar una decisión en cuanto a las especificaciones que el programa de cómputo deberá satisfacer.

Así, el objetivo de este trabajo es desarrollar una metodología en la cual se implemente de forma paralela el método numérico de descomposición de dominio de subestructuración precondicionado, explicando detalladamente los fundamentos matemáticos de la modelación de fenómenos de sistemas continuos por medio de ecuaciones diferenciales parciales, los fundamentos del método de descomposición de dominio al aplicarse a ecuaciones diferenciales parciales elípticas y las ventajas sobre otros métodos de solución numérica.

Mostraremos como construir el modelo computacional, esto no sólo nos ayudará a demostrar que es factible la construcción del propio modelo computacional a partir del modelo matemático y numérico para la solución de problemas reales. Además, mostrará los alcances y limitaciones en el consumo de recursos computacionales y nos permitirá la evaluación de algunas de las variantes de los métodos numéricos con los que es posible implementar el modelo computacional y haremos algo de análisis de rendimiento sin llegar a ser exhaustivo esté.

También se muestra el diseño, análisis y programación de la aplicación computacional del método de descomposición de dominio en forma serial como paralela, basada en el paradigma de programación orientado a objetos y como este paradigma ofrece una forma robusta y flexible para la representación de entidades abstractas y que permiten una modelación más versátil así como una forma más eficiente para la organización y reutilización de código.

1.3.1. Objetivos Generales

- Mostrar las bases de una metodología que se utiliza para aplicar el cómputo en paralelo a la modelación matemática y computacional de sistemas continuos, de forma flexible, escalable y eficiente.
- Mostrar los alcances y limitaciones de la metodología usando como herramientas de evaluación a los métodos de elemento finito secuencial, método de subestructuración secuencial y método de subestructuración paralelo.
- Mostrar los diversos esquemas de optimización de los recursos computacionales aplicables a un problema especifico.

Para facilitar la comprensión de las ideas básicas, se ha tomado la ecuación de Poisson. Es un ejemplo sencillo, pero gobierna los modelos de muchos sistemas de la ingeniería y de la ciencia, entre ellos el flujo de agua subterránea a través de un acuífero isotrópico, homogéneo bajo condiciones de equilibrio y es muy usada en múltiples ramas de la física. Por ejemplo, gobierna la ecuación de la conducción de calor en un sólido bajo condiciones de equilibrio.

1.3.2. Objetivos Particulares

Como objetivos particulares de este trabajo tenemos:

- Mostrar cómo aplicar la metodología para manejar problemas de gran tamaño (descomposición de malla fina).
- Mostrar cómo descomponer un dominio en un conjunto de subdominios que den una partición en la que el tiempo de cálculo sea mínimo para una configuración de hardware dada.
- Mostrar cuales son las posibles optimizaciones aplicables a una configuración de hardware dada.
- Mostrar que es posible trabajar problemas con una malla muy fina en un equipo paralelo pequeño.

Para poder cumplir con los objetivos, primeramente empezaremos describiendo las bases de los sistemas continuos y sus modelos, para conocer algunas propiedades de las ecuaciones diferenciales parciales que son generadas en la elaboración del modelo matemático. Después veremos la forma de pasar del modelo matemático al modelo numérico (el cual no es muy cercano al modelo computacional) introduciendo los fundamentos del método Galerkin y la derivación a partir de este del método de elemento finito, para posteriormente describir dos métodos de descomposición de dominio (Schwarz y subestructuración).

Mostraremos como construir el modelo computacional que dependerá fuertemente de la arquitectura de cómputo disponible y del lenguaje de programación

usado. Esto nos ayudará a demostrar que es factible la construcción del propio modelo computacional a partir del modelo matemático y numérico para la solución de problemas reales.

Además conoceremos los alcances y limitaciones en el consumo de recursos computacionales y nos permitirá la evaluación de algunas de las variantes de los métodos numéricos con los que es posible implementar el modelo computacional y haremos algo de análisis de rendimiento sin llegar a ser exhaustivo este.

Finalmente se exploran los alcances y limitaciones de cada uno de los métodos implementados (elemento finito secuencial, descomposición de dominio secuencial y paralelo) y se muestra como es posible optimizar los recursos computacionales con que se cuente.

1.4. Infraestructura Usada

El modelo computacional generado, está contenido en un programa de cómputo bajo el paradigma de orientación a objetos, programado en el lenguaje C++ en su forma secuencial y en su forma paralela en C++ en conjunto con la interfaz de paso de mensajes (MPI) bajo el esquema de maestro-esclavo. Una versión de estos programas está disponible en la página Web http://www.mmc.igeofcu.unam.mx/, bajo el rubro *FEM y DDM*.

Para desarrollar estos códigos, se realizo una jerarquía de clases para cada uno de los distintos componentes del sistema de elemento finito como de descomposición de dominio (rehusando todo el código de elemento finito en este), permitiendo usarlos tanto en forma secuencial como paralela.

Las pruebas de rendimiento de los distintos programas se realizaron en equipos secuenciales y paralelos (clusters) que están montados en el Instituto de Geofísica de la UNAM, en las pruebas de análisis de rendimiento se usaron para la parte secuencial el equipo:

- Computadora Pentium IV HT a 2.8 GHtz con 1 GB de RAM corriendo bajo el sistema operativo Linux Debian Stable con el compilador g++ de GNU.

Para la parte paralela se usaron los siguientes equipos:

- Cluster homogéneo de 10 nodos duales Xeon a 2.8 GHtz con 1 GB de RAM por nodo, unidos mediante una red Ethernet de 1 Gb, corriendo bajo el sistema operativo Linux Debian Stable con el compilador mpiCC de MPI de GNU. A cargo de la Dra. Alejandra Arciniega Ceballos del departamento de Geomagnetismo y Exploración.
- Cluster heterogéneo con el nodo maestro Pentium IV HT a 3.4 GHtz con 1 GB de RAM y 7 nodos esclavos Pentium IV HT a 2.8 GHtz con 0.5 GB de RAM por nodo, unidos mediante una red Ethernet de 100 Mb, corriendo bajo el sistema operativo Linux Debian Stable con el compilador mpiCC de MPI de GNU. A cargo del Dr. Ismael Herrera Revilla del departamento de Recursos Naturales.

También se realizaron algunas pruebas de rendimiento en otro cluster heterogéneo de 64 nodos (donde algunos nodos son duales y otros hasta con 4 procesadores) Intel Xeon de 64 bits interconectados mediante una red de baja latencia a 1 Gb, pero los resultados no son comparables con respecto a los clusters anteriores debido a las diferencias entre las arquitecturas.

Hay que notar que, el paradigma de programación orientada a objetos sacrifica algo de eficiencia computacional por requerir mayor manejo de recursos computacionales al momento de la ejecución. Pero en contraste, permite mayor flexibilidad a la hora adaptar los códigos a nuevas especificaciones. Adicionalmente, disminuye notoriamente el tiempo invertido en el mantenimiento y búsqueda de errores dentro del código. Esto tiene especial interés cuando se piensa en la cantidad de meses invertidos en la programación comparado con los segundos consumidos en la ejecución del mismo.

2. Sistemas Continuos y sus Modelos

Los fundamentos de la física macroscópica los proporciona la 'teoría de los medios continuos'. En este capítulo, con base en ella se introduce una formulación clara, general y sencilla de los modelos matemáticos de los sistemas continuos. Esta formulación es tan sencilla y tan general, que los modelos básicos de sistemas tan complicados y diversos como la atmósfera, los océanos, los yacimientos petroleros, o los geotérmicos, se derivan por medio de la aplicación repetida de una sola ecuación diferencial: 'la ecuación diferencial de balance'.

Dicha formulación también es muy clara, pues en el modelo general no hay ninguna ambigüedad; en particular, todas las variables y parámetros que intervienen en él, están definidos de manera unívoca. En realidad, este modelo general de los sistemas continuos constituye una realización extraordinaria de los paradigmas del pensamiento matemático. El descubrimiento del hecho de que los modelos matemáticos de los sistemas continuos, independientemente de su naturaleza y propiedades intrínsecas, pueden formularse por medio de balances, cuya idea básica no difiere mucho de los balances de la contabilidad financiera, fue el resultado de un largo proceso de perfeccionamiento en el que concurrieron una multitud de mentes brillantes.

2.1. Los Modelos

Un modelo de un sistema es un sustituto de cuyo comportamiento es posible derivar el correspondiente al sistema original. Los modelos matemáticos, en la actualidad, son los utilizados con mayor frecuencia y también los más versátiles. En las aplicaciones específicas están constituidos por programas de cómputo cuya aplicación y adaptación a cambios de las propiedades de los sistemas es relativamente fácil. También, sus bases y las metodologías que utilizan son de gran generalidad, por lo que es posible construirlos para situaciones y sistemas muy diversos.

Los modelos matemáticos son entes en los que se integran los conocimientos científicos y tecnológicos, con los que se construyen programas de cómputo que se implementan con medios computacionales. En la actualidad, la simulación numérica permite estudiar sistemas complejos y fenómenos naturales que sería muy costoso, peligroso o incluso imposible de estudiar por experimentación directa. En esta perspectiva la significación de los modelos matemáticos en ciencias e ingeniería es clara, porqué la modelación matemática constituye el método más efectivo de predecir el comportamiento de los diversos sistemas de interés. En nuestro país, ellos son usados ampliamente en la industria petrolera, en las ciencias y la ingeniería del agua y en muchas otras.

2.1.1. Física Microscópica y Física Macroscópica

La materia, cuando se le observa en el ámbito ultramicroscópico, está formada por moléculas y átomos. Estos a su vez, por partículas aún más pequeñas como los protones, neutrones y electrones. La predicción del comportamiento de

estas partículas es el objeto de estudio de la mecánica cuántica y la física nuclear. Sin embargo, cuando deseamos predecir el comportamiento de sistemas tan grandes como la atmósfera o un yacimiento petrolero, los cuales están formados por un número extraordinariamente grande de moléculas y átomos, su estudio resulta inaccesible con esos métodos y en cambio el enfoque macroscópico es apropiado.

Por eso en lo que sigue distinguiremos dos enfoques para el estudio de la materia y su movimiento. El primero -el de las moléculas, los átomos y las partículas elementales- es el enfoque microscópico y el segundo es el enfoque macroscópico. Al estudio de la materia con el enfoque macroscópico, se le llama física macroscópica y sus bases teóricas las proporciona la mecánica de los medios continuos.

Cuando se estudia la materia con este último enfoque, se considera que los cuerpos llenan el espacio que ocupan, es decir que no tienen huecos, que es la forma en que los vemos sin el auxilio de un microscopio. Por ejemplo, el agua llena todo el espacio del recipiente donde está contenida. Este enfoque macroscópico está presente en la física clásica. La ciencia ha avanzado y ahora sabemos que la materia está llena de huecos, que nuestros sentidos no perciben y que la energía también está cuantizada. A pesar de que estos dos enfoques para el análisis de los sistemas físicos, el microscópico y el macroscópico, parecen a primera vista conceptualmente contradictorios, ambos son compatibles, y complementarios, y es posible establecer la relación entre ellos utilizando a la mecánica estadística.

2.2. Cinemática de los Modelos de Sistemas Continuos

En la teoría de los sistemas continuos, los cuerpos llenan todo el espacio que ocupan. Y en cada punto del espacio físico hay una y solamente una partícula. Así, definimos como sistema continuo a un conjunto de partículas. Aún más, dicho conjunto es un subconjunto del espacio Euclidiano tridimensional. Un cuerpo es un subconjunto de partículas que en cualquier instante dado ocupa un dominio, en el sentido matemático, del espacio físico; es decir, del espacio Euclidiano tridimensional. Denotaremos por $B(t)$ a la región ocupada por el cuerpo $\mathcal{B}$, en el tiempo t, donde t puede ser cualquier número real.

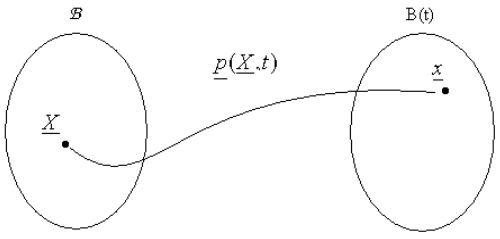

Figura 1: Representación del movimiento de partículas de un cuerpo $\mathcal{B}$, para un tiempo dado.

Frecuentemente, sin embargo, nuestro interés de estudio se limitará a un intervalo finito de tiempo. Dado un cuerpo $\mathcal{B}$, todo subdominio $\tilde{\mathcal{B}} \subset \mathcal{B}$, constituye a su vez otro cuerpo; en tal caso, se dice que $\tilde{\mathcal{B}} \subset \mathcal{B}$ es un subcuerpo de $\mathcal{B}$. De acuerdo con lo mencionado antes, una hipótesis básica de la teoría de los sistemas continuos es que en cualquier tiempo $t \in (-\infty, \infty)$ y en cada punto $x \in \mathcal{B}$ de la región ocupada por el cuerpo, hay una y sólo una partícula del cuerpo. Como en nuestra revisión se incluye no solamente la estática (es decir, los cuerpos en reposo), sino también la dinámica (es decir, los cuerpos en movimiento), un primer problema de la cinemática de los sistemas continuos consiste en establecer un procedimiento para identificar a las partículas cuando están en movimiento en el espacio físico.

Sea $\underline{X} \in \mathcal{B}$, una partícula y $p(\underline{X}, t)$ el vector de la posición que ocupa, en el espacio físico, dicha partícula en el instante t. Una forma, pero no la única, de identificar la partícula $\underline{X}$ es asociándole la posición que ocupa en un instante determinado. Tomaremos en particular el tiempo $t = 0$, en tal caso $\underline{p}(\underline{X}, 0) \equiv \underline{X}$.

A las coordenadas del vector $\underline{X} \equiv (x_1, x_2, x_3)$, se les llama las coordenadas materiales de la partícula. En este caso, las coordenadas materiales de una partícula son las coordenadas del punto del espacio físico que ocupaba la partícula en el tiempo inicial, $t = 0$. Desde luego, el tiempo inicial puede ser cualquier otro, si así se desea. Sea $\mathcal{B}$ el dominio ocupado por un cuerpo en el tiempo inicial, entonces $\underline{X} \in \mathcal{B}$ si y solamente si la partícula $\underline{X}$ es del cuerpo. Es decir, $\mathcal{B}$ caracteriza al cuerpo. Sin embargo, debido al movimiento, la región ocupada por el mismo cambia con el tiempo y será denotada por $\mathcal{B}(t)$.

Formalmente, para cualquier $t \in (-\infty, \infty)$, $\mathcal{B}(t)$ se define por

$$\mathcal{B}(t) \equiv \left\{ \underline{x} \in \mathbb{R}^3 \mid \exists \underline{X} \in \mathcal{B} \quad tal \quad que \quad \underline{x} = p(\underline{X}, t) \right\} \tag{1}$$

el vector posición $\underline{p}(\underline{X}, t)$ es función del vector tridimensional $\underline{X}$ y del tiempo. Si fijamos el tiempo t, $\underline{p}(\underline{X}, t)$ define una transformación del espacio Euclidiano $\mathbb{R}^3$ en si mismo y la Ec. (1) es equivalente a $\mathcal{B}(t) = \underline{p}(\mathcal{B}, t)$. Una notación utilizada para representar esta familia de funciones es $\underline{p}(\cdot, t)$. De acuerdo a la hipótesis de los sistemas continuos: En cualquier tiempo $t \in (-\infty, \infty)$ y en cada punto $\underline{x} \in \mathcal{B}$ de la región ocupada por el cuerpo hay una y sólo una partícula del cuerpo $\mathcal{B}$ para cada t fijo. Es decir, $\underline{p}(\cdot, t)$ es una función biunívoca, por lo que existe la función inversa $\underline{p}^{-1}(\cdot, t)$.

Si se fija la partícula $\underline{X}$ en la función $\underline{p}(\underline{X}, t)$ y se varía el tiempo t, se obtiene su trayectoria. Esto permite obtener la velocidad de cualquier partícula, la cual es un concepto central en la descripción del movimiento. Ella se define como la derivada con respecto al tiempo de la posición cuando la partícula se mantiene fija. Es decir, es la derivada parcial con respecto al tiempo de la función de posición $\underline{p}(\underline{X}, t)$. Por lo mismo, la velocidad como función de las coordenadas materiales de las partículas, está dada por

$$\underline{\mathrm{V}}(\underline{X}, t) \equiv \frac{\partial \underline{p}}{\partial t}(\underline{X}, t). \tag{2}$$

2.2.1. Propiedades Intensivas y sus Representaciones

En lo que sigue consideraremos funciones definidas para cada tiempo, en cada una de las partículas de un sistema continuo. A tales funciones se les llama 'propiedades intensivas'. Las propiedades intensivas pueden ser funciones escalares o funciones vectoriales. Por ejemplo, la velocidad, definida por la Ec. (2), es una función vectorial que depende de la partícula $\underline{X}$ y del tiempo t.

Una propiedad intensiva con valores vectoriales es equivalente a tres escalares, correspondientes a cada una de sus tres componentes. Hay dos formas de representar a las propiedades intensivas: la representación Euleriana y la representación Lagrangiana. Los nombres son en honor a los matemáticos Leonard Euler (1707-1783) y Joseph Louis Lagrange (1736-1813), respectivamente. Frecuentemente, el punto de vista Lagrangiano es utilizado en el estudio de los sólidos, mientras que el Euleriano se usa más en el estudio de los fluidos.

Considere una propiedad intensiva escalar, la cual en el tiempo t toma el valor $\phi(\underline{X}, t)$ en la partícula $\underline{X}$. Entonces, de esta manera se define una función $\phi : \mathcal{B} \to \mathbb{R}^1$, para cada $t \in (-\infty, \infty)$ a la que se denomina representación Lagrangiana de la propiedad intensiva considerada. Ahora, sea $\psi(\underline{x}, t)$ el valor que toma esa propiedad en la partícula que ocupa la posición $\underline{x}$, en el tiempo t. En este caso, para cada $t \in (-\infty, \infty)$ se define una función $\psi : \mathcal{B}(t) \to \mathbb{R}^1$ a la cual se denomina representación Euleriana de la función considerada. Estas dos representaciones de una misma propiedad están relacionadas por la siguiente identidad

$$\phi(\underline{X}, t) \equiv \psi(\underline{p}(\underline{X}, t), t). \tag{3}$$

Nótese que, aunque ambas representaciones satisfacen la Ec. (3), las funciones $\phi(\underline{X}, t)$ y $\psi(\underline{x}, t)$ no son idénticas. Sus argumentos $\underline{X}$ y $\underline{x}$ son vectores tridimensionales (es decir, puntos de $\mathbb{R}^3$); sin embargo, si tomamos $\underline{X} = \underline{x}$, en general

$$\phi(\underline{X}, t) \neq \psi(\underline{X}, t). \tag{4}$$

La expresión de la velocidad de una partícula dada por la Ec. (2), define a su representación Lagrangiana, por lo que utilizando la Ec. (3) es claro que

$$\frac{\partial p}{\partial t}(\underline{X}, t) = \underline{\mathrm{V}}(\underline{X}, t) \equiv \underline{\mathrm{v}}(\underline{p}(\underline{X}, t), t) \tag{5}$$

donde $\underline{\mathrm{v}}(\underline{x}, t)$ es la representación Euleriana de la velocidad. Por lo mismo

$$\underline{\mathrm{v}}(\underline{x}, t) \equiv \underline{\mathrm{V}}(\underline{p}^{-1}(\underline{x}, t), t). \tag{6}$$

Esta ecuación tiene la interpretación de que la velocidad en el punto $\underline{x}$ del espacio físico, es igual a la velocidad de la partícula que pasa por dicho punto en el instante t. La Ec. (6) es un caso particular de la relación

$$\psi(\underline{x}, t) \equiv \phi(\underline{p}^{-1}(\underline{x}, t), t)$$

de validez general, la cual es otra forma de expresar la relación de la Ec. (3) que existe entre las dos representaciones de una misma propiedad intensiva.

La derivada parcial con respecto al tiempo de la representación Lagrangiana $\phi(\underline{X},t)$ de una propiedad intensiva, de acuerdo a la definición de la derivada parcial de una función, es la tasa de cambio con respecto al tiempo que ocurre en una partícula fija. Es decir, si nos montamos en una partícula y medimos a la propiedad intensiva y luego los valores así obtenidos los derivamos con respecto al tiempo, el resultado final es $\frac{\partial\phi(\underline{X},t)}{\partial t}$. En cambio, si $\psi(\underline{x},t)$ es la representación Euleriana de esa misma propiedad, entonces $\frac{\partial\psi(\underline{x},t)}{\partial t}$ es simplemente la tasa de cambio con respecto al tiempo que ocurre en un punto fijo en el espacio. Tiene interés evaluar la tasa de cambio con respecto al tiempo que ocurre en una partícula fija, cuando se usa la representación Euleriana. Derivando con respecto al tiempo a la identidad de la Ec. (3) y la regla de la cadena, se obtiene

$$\frac{\partial\phi(\underline{X},t)}{\partial t}=\frac{\partial\psi}{\partial t}(\underline{p}(\underline{X},t),t)+\sum_{i=1}^{3}\frac{\partial\psi}{\partial x_i}(\underline{p}(\underline{X},t),t)\frac{\partial p_i}{\partial t}(\underline{X},t). \tag{7}$$

Se acostumbra definir el símbolo $\frac{D\psi}{Dt}$ por

$$\frac{D\psi}{Dt}=\frac{\partial\psi}{\partial t}+\sum_{i=1}^{3}v_i\frac{\partial\psi}{\partial x_i} \tag{8}$$

o, más brevemente,

$$\frac{D\psi}{Dt}=\frac{\partial\psi}{\partial t}+\underline{\mathrm{v}}\cdot\nabla\psi \tag{9}$$

utilizando esta notación, se puede escribir

$$\frac{\partial\phi(\underline{X},t)}{\partial t}=\frac{D\psi}{Dt}(\underline{p}(\underline{X},t)\equiv\left(\frac{\partial\psi}{\partial t}+\underline{\mathrm{v}}\cdot\nabla\psi\right)(\underline{p}(\underline{X},t),t). \tag{10}$$

Por ejemplo, la aceleración de una partícula se define como la derivada de la velocidad cuando se mantiene a la partícula fija. Aplicando la Ec. (9) se tiene

$$\frac{D\underline{\mathrm{v}}}{Dt}=\frac{\partial\underline{\mathrm{v}}}{\partial t}+\underline{\mathrm{v}}\cdot\nabla\underline{\mathrm{v}} \tag{11}$$

una expresión más transparente se obtiene aplicando la Ec. (9) a cada una de las componentes de la velocidad. Así, se obtiene

$$\frac{D\mathrm{v}_i}{Dt}=\frac{\partial\mathrm{v}_i}{\partial t}+\underline{\mathrm{v}}\cdot\nabla\mathrm{v}_i. \tag{12}$$

Desde luego, la aceleración, en representación Lagrangiana es simplemente

$$\frac{\partial}{\partial t}\underline{\mathrm{V}}(\underline{X},t)=\frac{\partial^2}{\partial t^2}\underline{p}(\underline{X},t). \tag{13}$$

2.2.2. Propiedades Extensivas

En la sección anterior se consideraron funciones definidas en las partículas de un cuerpo, más precisamente, funciones que hacen corresponder a cada partícula y cada tiempo un número real, o un vector del espacio Euclidiano tridimensional $\mathbb{R}^3$. En ésta, en cambio, empezaremos por considerar funciones que a cada cuerpo $\mathcal{B}$ de un sistema continuo, y a cada tiempo t le asocia un número real o un vector de $\mathbb{R}^3$. A una función de este tipo $\mathbb{E}(\mathcal{B}, t)$ se le llama 'propiedad extensiva' cuando esta dada por una integral

$$\mathbb{E}(\mathcal{B}, t) \equiv \int_{\mathcal{B}(t)} \psi(\underline{x}, t) d\underline{\mathrm{x}}. \tag{14}$$

Observe que, en tal caso, el integrando define una función $\psi(\underline{x}, t)$ y por lo mismo, una propiedad intensiva. En particular, la función $\psi(\underline{x}, t)$ es la representación Euleriana de esa propiedad intensiva. Además, la Ec. (14) establece una correspondencia biunívoca entre las propiedades extensivas y las intensivas, porqué dada la representación Eulereana $\psi(\underline{x}, t)$ de cualquier propiedad intensiva, su integral sobre el dominio ocupado por cualquier cuerpo, define una propiedad extensiva. Finalmente, la notación empleada en la Ec. (14) es muy explicita, pues ahí se ha escrito $\mathbb{E}(\mathcal{B}, t)$ para enfatizar que el valor de la propiedad extensiva corresponde al cuerpo $\mathcal{B}$. Sin embargo, en lo que sucesivo, se simplificara la notación omitiendo el símbolo $\mathcal{B}$ es decir, se escribirá $\mathbb{E}(t)$ en vez de $\mathbb{E}(\mathcal{B}, t)$.

Hay diferentes formas de definir a las propiedades intensivas. Como aquí lo hemos hecho, es por unidad de volumen. Sin embargo, es frecuente que se le defina por unidad de masa véase [15]. Es fácil ver que la propiedad intensiva por unidad de volumen es igual a la propiedad intensiva por unidad de masa multiplicada por la densidad de masa (es decir, masa por unidad de volumen), por lo que es fácil pasar de un concepto al otro, utilizando la densidad de masa.

Sin embargo, una ventaja de utilizar a las propiedades intensivas por unidad de volumen, en lugar de las propiedades intensivas por unidad de masa, es que la correspondencia entre las propiedades extensivas y las intensivas es más directa: dada una propiedad extensiva, la propiedad intensiva que le corresponde es la función que aparece como integrando, cuando aquélla se expresa como una integral de volumen. Además, del cálculo se sabe que

$$\psi(\underline{x}, t) \equiv \lim_{Vol \to 0} \frac{\mathbb{E}(t)}{Vol} = \lim_{Vol \to 0} \frac{\int_{\mathcal{B}(t)} \psi(\underline{\xi}, t) d\underline{\xi}}{Vol}. \tag{15}$$

La Ec. (15) proporciona un procedimiento efectivo para determinar las propiedades extensivas experimentalmente: se mide la propiedad extensiva en un volumen pequeño del sistema continuo de que se trate, se le divide entre le volumen y el cociente que se obtiene es una buena aproximación de la propiedad intensiva.

El uso que haremos del concepto de propiedad extensiva es, desde luego, lógicamente consistente. En particular, cualquier propiedad que satisface las condiciones de la definición de propiedad extensiva establecidas antes es, por

ese hecho, una propiedad extensiva. Sin embargo, no todas las propiedades extensivas que se pueden obtener de esta manera son de interés en la mecánica de los medios continuos. Una razón básica por la que ellas son importantes es porqué el modelo general de los sistemas continuos se formula en términos de ecuaciones de balance de propiedades extensivas, como se verá más adelante.

2.2.3. Balance de Propiedades Extensivas e Intensivas

Los modelos matemáticos de los sistemas continuos están constituidos por balances de propiedades extensivas. Por ejemplo, los modelos de transporte de solutos (los contaminantes transportados por corrientes superficiales o subterráneas, son un caso particular de estos procesos de transporte) se construyen haciendo el balance de la masa de soluto que hay en cualquier dominio del espacio físico. Aquí, el término balance se usa, esencialmente, en un sentido contable. En la contabilidad que se realiza para fines financieros o fiscales, la diferencia de las entradas menos las salidas nos da el aumento, o cambio, de capital. En forma similar, en la mecánica de los medios continuos se realiza, en cada cuerpo del sistema continuo, un balance de las propiedades extensivas en que se basa el modelo.

Ecuación de Balance Global Para realizar tales balances es necesario, en primer lugar, identificar las causas por las que las propiedades extensivas pueden cambiar. Tomemos como ejemplo de propiedad extensiva a las existencias de maíz que hay en el país. La primera pregunta es: ¿qué causas pueden motivar su variación, o cambio, de esas existencias?. Un análisis sencillo nos muestra que dicha variación puede ser debida a que se produzca o se consuma. También a que se importe o se exporte por los límites del país (fronteras o litorales). Y con esto se agotan las causas posibles; es decir, esta lista es exhaustiva. Producción y consumo son términos similares, pero sus efectos tienen signos opuestos, que fácilmente se engloban en uno solo de esos conceptos. De hecho, si convenimos en que la producción puede ser negativa, entonces el consumo es una producción negativa.

Una vez adoptada esta convención, ya no es necesario ocuparnos separadamente del consumo. En forma similar, la exportación es una importación negativa. Entonces, el incremento en las existencias $\Delta\mathbb{E}$ en un período Δt queda dado por la ecuación

$$\Delta\mathbb{E} = \mathbb{P} + \mathbb{I} \tag{16}$$

donde a la producción y a la importación, ambas con signo, se les ha representado por $\mathbb{P}$ y $\mathbb{I}$ respectivamente.

Similarmente, en la mecánica de los medios continuos, la lista exhaustiva de las causas por las que una propiedad extensiva de cualquier cuerpo puede cambiar, contiene solamente dos motivos:

i) Por producción en el interior del cuerpo; y

ii) Por importación (es decir, transporte) a través de la frontera.

Esto conduce a la siguiente ecuación de "balance global", de gran generalidad, para las propiedades extensivas

$$\frac{d\mathbb{E}}{dt}(t) = \int_{\mathcal{B}(t)} g(\underline{x},t)d\underline{x} + \int_{\partial\mathcal{B}(t)} q(\underline{x},t)d\underline{x} + \int_{\Sigma(t)} g_\Sigma(\underline{x},t)d\underline{x}. \tag{17}$$

Donde $g(\underline{x},t)$ es la generación en el interior del cuerpo, con signo, de la propiedad extensiva correspondiente, por unidad de volumen, por unidad de tiempo. Además, en la Ec. (17) se ha tomado en cuenta la posibilidad de que haya producción concentrada en la superficie $\Sigma(t)$, la cual está dada en esa ecuación por la última integral, donde $g_\Sigma(\underline{x},t)$ es la producción por unidad de área. Por otra parte $q(\underline{x},t)$ es lo que se importa o transporta hacia el interior del cuerpo a través de la frontera del cuerpo $\partial\mathcal{B}(t)$, en otras palabras, es el flujo de la propiedad extensiva a través de la frontera del cuerpo, por unidad de área, por unidad de tiempo. Puede demostrarse, con base en hipótesis válidas en condiciones muy generales, que para cada tiempo t existe un campo vectorial $\tau(\underline{x},t)$ tal que

$$q(\underline{x},t) \equiv \tau(\underline{x},t)\cdot\underline{n}(\underline{x},t) \tag{18}$$

donde $\underline{n}(\underline{x},t)$ es normal exterior a $\partial\mathcal{B}(t)$. En vista de esta relación, la Ec. (17) de balance se puede escribir como

$$\frac{d\mathbb{E}}{dt}(t) = \int_{\mathcal{B}(t)} g(\underline{x},t)d\underline{x} + \int_{\partial\mathcal{B}(t)} \tau(\underline{x},t)\cdot\underline{n}(\underline{x},t)d\underline{x} + \int_{\Sigma(t)} g_\Sigma(\underline{x},t)d\underline{x}. \tag{19}$$

La relación (19) se le conoce con el nombre de "ecuación general de balance global" y es la ecuación básica de los balances de los sistemas continuos. A la función $g(\underline{x},t)$ se le denomina el generación interna y al campo vectorial $\tau(\underline{x},t)$ el campo de flujo.

Condiciones de Balance Local Los modelos de los sistemas continuos están constituidos por las ecuaciones de balance correspondientes a una colección de propiedades extensivas. Así, a cada sistema continuo le corresponde una familia de propiedades extensivas, tal que, el modelo matemático del sistema está constituido por las condiciones de balance de cada una de las propiedades extensivas de dicha familia.

Sin embargo, las propiedades extensivas mismas no se utilizan directamente en la formulación del modelo, en su lugar se usan las propiedades intensivas asociadas a cada una de ellas. Esto es posible porqué las ecuaciones de balance global son equivalentes a las llamadas condiciones de balance local, las cuales se expresan en términos de las propiedades intensivas correspondientes. Las condiciones de balance local son de dos clases: 'las ecuaciones diferenciales de balance local' y 'las condiciones de salto'.

Las primeras son ecuaciones diferenciales parciales, que se deben satisfacer en cada punto del espacio ocupado por el sistema continuo, y las segundas son ecuaciones algebraicas que las discontinuidades deben satisfacer donde ocurren; es decir, en cada punto de Σ. Cabe mencionar que las ecuaciones diferenciales

de balance local son de uso mucho más amplio que las condiciones de salto, pues estas últimas solamente se aplican cuando y donde hay discontinuidades, mientras que las primeras en todo punto del espacio ocupado por el sistema continuo.

Una vez establecidas las ecuaciones diferenciales y de salto del balance local, e incorporada la información científica y tecnológica necesaria para completar el modelo (la cual por cierto se introduce a través de las llamadas 'ecuaciones constitutivas'), el problema matemático de desarrollar el modelo y derivar sus predicciones se transforma en uno correspondiente a la teoría de la ecuaciones diferenciales, generalmente parciales, y sus métodos numéricos.

Las Ecuaciones de Balance Local En lo que sigue se supone que las propiedades intensivas pueden tener discontinuidades, de salto exclusivamente, a través de la superficie $\Sigma(t)$. Se entiende por 'discontinuidad de salto', una en que el límite por ambos lados de $\Sigma(t)$ existe, pero son diferentes.

Se utilizará en lo que sigue los resultados matemáticos que se dan a continuación, ver [8].

Teorema 1 *Para cada $t > 0$, sea $\mathcal{B}(t) \subset \mathbb{R}^3$ el dominio ocupado por un cuerpo. Suponga que la 'propiedad intensiva' $\psi(\underline{x}, t)$ es de clase C^1, excepto a través de la superficie $\Sigma(t)$. Además, sean las funciones $\underline{v}(\underline{x}, t)$ y $\underline{v}_\Sigma(\underline{x}, t)$ esta última definida para $\underline{x} \in \Sigma(t)$ solamente, las velocidades de las partículas y la de $\Sigma(t)$, respectivamente. Entonces*

$$\frac{d}{dt}\int_{\mathcal{B}(t)} \psi d\underline{x} \equiv \int_{\mathcal{B}(t)} \left\{ \frac{\partial \psi}{\partial t} + \nabla \cdot (\underline{v}\psi) \right\} d\underline{x} + \int_\Sigma \left[(\underline{v} - \underline{v}_\Sigma)\, \psi \right] \cdot \underline{n} d\underline{x}. \tag{20}$$

Teorema 2 *Considere un sistema continuo, entonces, la 'ecuación de balance global' (19) se satisface para todo cuerpo del sistema continuo si y solamente si se cumplen las condiciones siguientes:*

i) La ecuación diferencial

$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\underline{v}\psi) = \nabla \cdot \underline{\tau} + g \tag{21}$$

vale en todo punto $\underline{x} \in \mathbb{R}^3$, de la región ocupada por el sistema.

ii) La ecuación

$$\left[\psi\, (\underline{v} - \underline{v}_\Sigma) - \underline{\tau} \right] \cdot \underline{n} = g_\Sigma \tag{22}$$

vale en todo punto $\underline{x} \in \Sigma$.

A las ecuaciones (21) y (22), se les llama 'ecuación diferencial de balance local' y 'condición de salto', respectivamente.

Desde luego, el caso más general que se estudiará se refiere a situaciones dinámicas; es decir, aquéllas en que las propiedades intensivas cambian con el tiempo. Sin embargo, los estados estacionarios de los sistemas continuos son de sumo interés. Por estado estacionario se entiende uno en que las propiedades intensivas son independientes del tiempo. En los estados estacionarios, además,

las superficies de discontinuidad $\Sigma(t)$ se mantienen fijas (no se mueven). En este caso $\frac{\partial \psi}{\partial t} = 0$ y $\underline{v}_\Sigma = 0$. Por lo mismo, para los estados estacionarios, la ecuación de balance local y la condición de salto se reducen a

$$\nabla \cdot (\underline{v}\psi) = \nabla \cdot \underline{\tau} + g \tag{23}$$

que vale en todo punto $\underline{x} \in \mathbb{R}^3$ y

$$[\psi \underline{v} - \underline{\tau}] \cdot \underline{n} = g_\Sigma \tag{24}$$

que se satisface en todo punto de la discontinuidad $\Sigma(t)$ respectivamente.

2.3. Ejemplos de Modelos

Una de las aplicaciones más sencillas de las condiciones de balance local es para formular restricciones en el movimiento. Aquí ilustramos este tipo de aplicaciones formulando condiciones que se deben cumplir localmente cuando un fluido es incompresible. La afirmación de que un fluido es incompresible significa que todo cuerpo conserva el volumen de fluido en su movimiento. Entonces, se consideraran dos casos: el de un 'fluido libre' y el de un 'fluido en un medio poroso'. En el primer caso, el fluido llena completamente el espacio físico que ocupa el cuerpo, por lo que el volumen del fluido es igual al volumen del dominio que ocupa el cuerpo, así

$$V_f(t) = \int_{\mathcal{B}(t)} d\underline{x} \tag{25}$$

aquí, $V_f(t)$ es el volumen del fluido y $\mathcal{B}(t)$ es el dominio del espacio físico (es decir, de $\mathbb{R}^3$) ocupado por el cuerpo. Observe que una forma más explicita de esta ecuación es

$$V_f(t) = \int_{\mathcal{B}(t)} 1 d\underline{x} \tag{26}$$

porqué en la integral que aparece en la Ec. (25) el integrando es la función idénticamente 1. Comparando esta ecuación con la Ec. (14), vemos que el volumen del fluido es una propiedad extensiva y que la propiedad intensiva que le corresponde es $\psi = 1$.

Además, la hipótesis de incompresibilidad implica

$$\frac{dV_f}{dt}(t) = 0 \tag{27}$$

esta es el balance global de la Ec. (19), con $g = g_\Sigma = 0$ y $\tau = 0$, el cual a su vez es equivalente a las Ecs. (21) y (22). Tomando en cuenta además que $\psi = 1$, la Ec. (21) se reduce a

$$\nabla \cdot \underline{v} = 0. \tag{28}$$

Esta es la bien conocida condición de incompresibilidad para un fluido libre Además, aplicando la Ec. (22) donde haya discontinuidades, se obtiene $[\underline{v}] \cdot \underline{n} = 0$. Esto implica que si un fluido libre es incompresible, la velocidad de sus partículas es necesariamente continua.

El caso en que el fluido se encuentra en un 'medio poroso', es bastante diferente. Un medio poroso es un material sólido que tiene huecos distribuidos en toda su extensión, cuando los poros están llenos de un fluido, se dice que el medio poroso esta 'saturado'. Esta situación es la de mayor interés en la práctica y es también la más estudiada. En muchos de los casos que ocurren en las aplicaciones el fluido es agua o petróleo. A la fracción del volumen del sistema, constituido por la 'matriz sólida' y los huecos, se le llama 'porosidad' y se le representara por ϕ, así

$$\phi(x,t) = \lim_{V \to 0} \frac{\text{Volumen de huecos}}{\text{Volumen total}} \tag{29}$$

aquí hemos escrito $\phi(x,t)$ para enfatizar que la porosidad generalmente es función tanto de la posición como del tiempo. Las variaciones con la posición pueden ser debidas, por ejemplo, a heterogeneidad del medio y los cambios con el tiempo a su elasticidad; es decir, los cambios de presión del fluido originan esfuerzos en los poros que los dilatan o los encogen.

Cuando el medio esta saturado, el volumen del fluido V_f es igual al volumen de los huecos del dominio del espacio físico que ocupa, así

$$V_f(t) = \int_{\mathcal{B}(t)} \phi(x,t) d\underline{x}. \tag{30}$$

En vista de esta ecuación, la propiedad intensiva asociada al volumen de fluido es la porosidad $\phi(x,t)$ por lo que la condición de incomprensibilidad del fluido contenido en un medio poroso, esta dada por la ecuación diferencial

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\underline{v}\phi) = 0. \tag{31}$$

Que la divergencia de la velocidad sea igual a cero en la Ec. (28) como condición para que un fluido en su movimiento libre conserve su volumen, es ampliamente conocida. Sin embargo, este no es el caso de la Ec. (31), como condición para la conservación del volumen de los cuerpos de fluido contenidos en un medio poroso. Finalmente, debe observarse que cualquier fluido incompresible satisface la Ec. (28) cuando se mueve en el espacio libre y la Ec. (31) cuando se mueve en un medio poroso.

Cuando un fluido efectúa un movimiento en el que conserva su volumen, al movimiento se le llama 'isocorico'. Es oportuno mencionar que si bien cierto que cuando un fluido tiene la propiedad de ser incompresible, todos sus movimientos son isocoricos, lo inverso no es cierto: un fluido compresible en ocasiones puede efectuar movimientos isocoricos.

Por otra parte, cuando un fluido conserva su volumen en su movimiento satisface las condiciones de salto de Ec. (22), las cuales para este caso son

$$[\phi(\underline{v} - \underline{v}_\Sigma)] \cdot \underline{n} = 0. \tag{32}$$

En aplicaciones a geohidrología y a ingeniería petrolera, las discontinuidades de la porosidad están asociadas a cambios en los estratos geológicos y por esta

razón están fijas en el espacio; así, $\underline{v}_\Sigma = 0$ y la Ec. (32) se reduce a

$$[\phi \underline{v}] \cdot \underline{n} = 0 \tag{33}$$

o, de otra manera

$$\phi_+ v_{n_+} = \phi_- v_{n_-}. \tag{34}$$

Aquí, la componente normal de la velocidad es $v_n \equiv \underline{v} \cdot \underline{n}$ y los subíndices más y menos se utilizan para denotar los límites por los lado más y menos de Σ, respectivamente. Al producto de la porosidad por la velocidad se le conoce con el nombre de velocidad de Darcy $\underline{U}$, es decir

$$\underline{U} = \phi \underline{v} \tag{35}$$

utilizándola, las Ecs. (33) y (34) obtenemos

$$[\underline{U}] \cdot \underline{n} = 0 \quad \text{y} \quad \underline{U}_{n_+} = \underline{U}_{n_-} \tag{36}$$

es decir, 1.

La Ec. (34) es ampliamente utilizada en el estudio del agua subterránea (geohidrología). Ahí, es frecuente que la porosidad ϕ sea discontinua en la superficie de contacto entre dos estratos geológicos diferentes, pues generalmente los valores que toma esta propiedad dependen de cada estrato. En tal caso, $\phi_+ \neq \phi_-$ por lo que $v_{n_+} \neq v_{n_-}$ necesariamente.

Para más detalles de la forma y del desarrollo de algunos modelos usados en ciencias de la tierra, véase [8], [15], [23] y [13].

3. Ecuaciones Diferenciales Parciales

Cada una de las ecuaciones de balance da lugar a una ecuación diferencial parcial u ordinaria (en el caso en que el modelo depende de una sola variable independiente), la cual se complementa con las condiciones de salto, en el caso de los modelos discontinuos. Por lo mismo, los modelos de los sistemas continuos están constituidos por sistemas de ecuaciones diferenciales cuyo número es igual al número de propiedades intensivas que intervienen en la formulación del modelo básico.

Los sistemas de ecuaciones diferenciales se clasifican en elípticas, hiperbólicas y parabólicas. Es necesario aclarar que esta clasificación no es exhaustiva; es decir, existen sistemas de ecuaciones diferenciales que no pertenecen a ninguna de estas categorías. Sin embargo, casi todos los modelos de sistemas continuos, en particular los que han recibido mayor atención hasta ahora, si están incluidos en alguna de estas categorías.

3.1. Clasificación

Es importante clasificar a las ecuaciones diferenciales parciales y a los sistemas de tales ecuaciones, porqué muchas de sus propiedades son comunes a cada una de sus clases. Así, su clasificación es un instrumento para alcanzar el objetivo de unidad conceptual. La forma más general de abordar la clasificación de tales ecuaciones, es estudiando la clasificación de sistemas de ecuaciones. Sin embargo, aquí solamente abordaremos el caso de una ecuación diferencia de segundo orden, pero utilizando un método de análisis que es adecuado para extenderse a sistemas de ecuaciones.

La forma general de un operador diferencial cuasi-lineal de segundo orden definido en $\Omega \subset \mathbb{R}^2$ es

$$\mathcal{L}u \equiv a(x,y)\frac{\partial^2 u}{\partial x^2} + b(x,y)\frac{\partial^2 u}{\partial x \partial y} + c(x,y)\frac{\partial^2 u}{\partial y^2} = F(x,y,u,u_x,u_y) \tag{37}$$

para una función u de variables independientes x e y. Nos restringiremos al caso en que a, b y c son funciones sólo de x e y y no funciones de u.

Para la clasificación de las ecuaciones de segundo orden consideraremos una simplificación de la ecuación anterior en donde $F(x,y,u,u_x,u_y) = 0$ y los coeficientes a, b y c son funciones constantes, es decir

$$a\frac{\partial^2 u}{\partial x^2} + b\frac{\partial^2 u}{\partial x \partial y} + c\frac{\partial^2 u}{\partial y^2} = 0 \tag{38}$$

en la cual, examinaremos los diferentes tipos de solución que se pueden obtener para diferentes elecciones de a, b y c. Entonces iniciando con una solución de la forma

$$u(x,y) = f(mx + y) \tag{39}$$

para una función f de clase C^2 y para una constante m, que deben ser determinadas según los requerimientos de la Ec. (38). Usando un apostrofe para

denotar la derivada de f con respecto de su argumento, las requeridas derivadas parciales de segundo orden de la Ec. (38) son

$$\frac{\partial^2 u}{\partial x^2} = m^2 f'', \quad \frac{\partial^2 u}{\partial x \partial y} = m f'', \quad \frac{\partial^2 u}{\partial y^2} = f'' \tag{40}$$

sustituyendo la ecuación anterior en la Ec. (38) obtenemos

$$\left(am^2 + bm + c\right) f'' = 0 \tag{41}$$

de la cual podemos concluir que $f'' = 0$ ó $am^2 + bm + c = 0$ ó ambas. En el caso de que $f'' = 0$ obtenemos la solución $f = f_0 + mx + y$, la cual es una función lineal de x e y y es expresada en términos de dos constantes arbitrarias, f_0 y m. En el otro caso obtenemos

$$am^2 + bm + c = 0 \tag{42}$$

resolviendo esta ecuación cuadrática para m obtenemos las dos soluciones

$$m_1 = \frac{\left(-b + \sqrt[2]{b^2 - 4ac}\right)}{2a}, \quad m_2 = \frac{\left(-b - \sqrt[2]{b^2 - 4ac}\right)}{2a} \tag{43}$$

de donde es evidente la importancia de los coeficientes de la Ec. (38), ya que el signo del discriminante $\left(b^2 - 4ac\right)$ es crucial para determinar el número y tipo de soluciones de la Ec. (42). Así, tenemos tres casos a considerar:

Caso I. $\left(b^2 - 4ac\right) > 0$, **Ecuación Hiperbólica**.

La Ec. (42) tiene dos soluciones reales distintas, m_1 y m_2. Así cualquier función de cualquiera de los dos argumentos $m_1 x + y$ ó $m_2 x + y$ resuelven a la Ec. (38). Por lo tanto la solución general de la Ec. (38) es

$$u(x, y) = \mathcal{F}\left(m_1 x + y\right) + \mathcal{G}(m_2 x + y) \tag{44}$$

donde $\mathcal{F}$ y $\mathcal{G}$ son cualquier función de clase C^2. Un ejemplo de este tipo de ecuaciones es la ecuación de onda, cuya ecuación canónica es

$$\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial t^2} = 0. \tag{45}$$

Caso II. $\left(b^2 - 4ac\right) = 0$, **Ecuación Parabólica**.

Asumiendo que $b \neq 0$ y $a \neq 0$ (lo cual implica que $c \neq 0$). Entonces se tiene una sola raíz degenerada de la Ec. (42) con el valor de $m_1 = \frac{-b}{2a}$ que resuelve a la Ec. (38). Por lo tanto la solución general de la Ec. (38) es

$$u(x, y) = \mathcal{F}\left(m_1 x + y\right) + y\mathcal{G}(m_1 x + y) \tag{46}$$

donde $\mathcal{F}$ y $\mathcal{G}$ son cualquier función de clase C^2. Si $b = 0$ y $a = 0$, entonces la solución general es

$$u(x, y) = \mathcal{F}\left(x\right) + y\mathcal{G}(x) \tag{47}$$

la cual es análoga si $b = 0$ y $c = 0$. Un ejemplo de este tipo de ecuaciones es la ecuación de difusión o calor, cuya ecuación canónica es

$$\frac{\partial^2 u}{\partial x^2} - \frac{\partial u}{\partial t} = 0. \tag{48}$$

Caso III. $\left(b^2 - 4ac\right) < 0$, **Ecuación Elíptica**.

La Ec. (42) tiene dos soluciones complejas m_1 y m_2 las cuales satisfacen que m_2 es el conjugado complejo de m_1, es decir, $m_2 = m_1^*$. La solución general puede ser escrita en la forma

$$u(x, y) = \mathcal{F}\left(m_1 x + y\right) + \mathcal{G}(m_2 x + y) \tag{49}$$

donde $\mathcal{F}$ y $\mathcal{G}$ son cualquier función de clase C^2. Un ejemplo de este tipo de ecuaciones es la ecuación de Laplace, cuya ecuación canónica es

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0. \tag{50}$$

Consideremos ahora el caso de un operador diferencial lineal de segundo orden definido en $\Omega \subset \mathbb{R}^n$ cuya forma general es

$$\mathcal{L}u = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=1}^{n} b_i \frac{\partial u}{\partial x_i} + cu \tag{51}$$

y consideremos también la ecuación homogénea asociada a este operador

$$\mathcal{L}u = 0 \tag{52}$$

además, sea $\underline{x} \in \Omega$ un punto del espacio Euclidiano y $V(x)$ una vecindad de ese punto. Sea una función u definida en $V(\underline{x})$ con la propiedad de que exista una variedad Σ de dimensión $n-1$ cerrada y orientada, tal que la función u satisface la Ec. (52) en $V(\underline{x})\backslash\Sigma$. Se supone además que existe un vector unitario $\underline{n}$ que apunta en la dirección positiva (único) está definido en Σ. Además, la función u y sus derivadas de primer orden son continuas a través de Σ, mientras que los límites de las segundas derivadas de u existen por ambos lados de Σ. Sea $\underline{x} \in \Sigma$ tal que

$$\left[\frac{\partial^2 u}{\partial x_i \partial x_j}(\underline{x})\right] \neq 0 \tag{53}$$

para alguna pareja $i, j = 1, ..., n$. Entonces decimos que la función u es una solución débil de esta ecuación en $\underline{x}$.

Teorema 3 *Una condición necesaria para que existan soluciones débiles de la ecuación homogénea (52) en un punto $\underline{x} \in \Sigma$ es que*

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} n_i n_j = 0. \tag{54}$$

Así, si definimos a la matriz $\underline{\underline{A}} = (a_{ij})$ y observamos que

$$\underline{n} \cdot \underline{\underline{A}} \cdot \underline{n} = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} n_i n_j \tag{55}$$

entonces podemos decir que:

I) Cuando todos los eigenvalores de la matriz $\underline{\underline{A}}$ son distintos de cero y además del mismo signo, entonces se dice que el operador es **Elíptico**.

II) Cuando todos los eigenvalores de la matriz $\underline{\underline{A}}$ son distintos de cero y además $n-1$ de ellos tienen el mismo signo, entonces se dice que el operador es **Hiperbólico**.

III) Cuando uno y sólo uno de los eigenvalores de la matriz $\underline{\underline{A}}$ es igual a cero, entonces se dice que el operador es **Parabólico**.

Para el caso en que $n = 2$, esta forma de clasificación coincide con la dada anteriormente.

3.1.1. Condiciones Iniciales y de Frontera

Dado un problema concreto de ecuaciones en derivadas parciales sobre un dominio Ω, si la solución existe, esta no es única ya que generalmente este tiene un número infinito de soluciones. Para que el problema tenga una y sólo una solución es necesario imponer condiciones auxiliares apropiadas y estas son las condiciones iniciales y condiciones de frontera.

En esta sección sólo se enuncian de manera general las condiciones iniciales y de frontera que son esenciales para definir un problema de ecuaciones diferenciales:

A) **Condiciones Iniciales**

Las condiciones iniciales expresan el valor de la función al tiempo inicial $t = 0$ (t puede ser fijada en cualquier valor)

$$u(\underline{x}, \underline{y}, 0) = \gamma(\underline{x}, \underline{y}). \tag{56}$$

B) **Condiciones de Frontera**

Las condiciones de frontera especifican los valores que la función $u(\underline{x}, \underline{y}, t)$ o $\nabla u(\underline{x}, \underline{y}, t)$ tomarán en la frontera $\partial\Omega$, siendo de tres tipos posibles:

1) **Condiciones tipo Dirichlet**

Especifica los valores que la función $u(\underline{x}, \underline{y}, t)$ toma en la frontera $\partial\Omega$

$$u(\underline{x}, \underline{y}, t) = \gamma(\underline{x}, \underline{y}). \tag{57}$$

2) **Condiciones tipo Neumann**

Aquí se conoce el valor de la derivada de la función $u(\underline{x}, \underline{y}, t)$ con respecto a la normal $\underline{n}$ a lo largo de la frontera $\partial\Omega$

$$\nabla u(\underline{x}, \underline{y}, t) \cdot \underline{n} = \gamma(\underline{x}, \underline{y}). \tag{58}$$

3) **Condiciones tipo Robin**

Está condición es una combinación de las dos anteriores

$$\alpha(\underline{x}, \underline{y})u(\underline{x}, \underline{y}, t) + \beta(\underline{x}, \underline{y})\nabla u(\underline{x}, \underline{y}, t) \cdot \underline{n} = g_{\partial}(\underline{x}, \underline{y}) \tag{59}$$

$\forall\ \underline{x},\ \underline{y} \in \partial\Omega$.

En un problema dado se debe prescribir las condiciones iniciales al problema y debe de existir alguno de los tipos de condiciones de frontera o combinación de ellas en $\partial\Omega$.

3.1.2. Modelos Completos

Los modelos de los sistemas continuos están constituidos por:

- Una colección de propiedades intensivas o lo que es lo mismo, extensivas.
- El conjunto de ecuaciones de balance local correspondientes (diferenciales y de salto).
- Suficientes relaciones que liguen a las propiedades intensivas entre sí y que definan a g, $\underline{\tau}$ y $\underline{v}$ en términos de estas, las cuales se conocen como leyes constitutivas.

Una vez que se han planteado las ecuaciones que gobiernan al problema, las condiciones iniciales, de frontera y mencionado los procesos que intervienen de manera directa en el fenómeno estudiado, necesitamos que nuestro modelo sea *completo.* Decimos que el modelo de un sistema es *completo* si define un problema *bien planteado.* Un problema de valores iniciales y condiciones de frontera es *bien planteado* si cumple que:

i) Existe una y sólo una solución y,

ii) La solución depende de manera continua de las condiciones iniciales y de frontera del problema.

Es decir, un modelo completo es aquél en el cual se incorporan condiciones iniciales y de frontera que definen conjuntamente con las ecuaciones diferenciales un problema bien planteado.

A las ecuaciones diferenciales definidas en $\Omega \subset \mathbb{R}^n$

$$\begin{aligned} \Delta u &= 0 \\ \frac{\partial^2 u}{\partial t^2} - \Delta u &= 0 \\ \frac{\partial u}{\partial t} - \Delta u &= 0 \end{aligned} \tag{60}$$

se les conoce con los nombres de ecuación de Laplace, ecuación de onda y ecuación del calor, respectivamente. Cuando se considera la primera de estas ecuaciones, se entiende que u es una función del vector $x \equiv (x_1, ..., x_n)$, mientras que cuando se considera cualquiera de las otras dos, u es una función del vector $x \equiv (x_1, ..., x_n, t)$. Así, en estos últimos casos el número de variables independientes es $n + 1$ y los conceptos relativos a la clasificación y las demás nociones discutidas con anterioridad deben aplicarse haciendo la sustitución $n \to n + 1$ e identificando $x_{n+1} = t$.

Ecuación de Laplace Para la ecuación de Laplace consideraremos condiciones del tipo Robin. En particular, condiciones de Dirichlet y condiciones de Neumann. Sin embargo, en este último caso, la solución no es única pues cualquier función constante satisface la ecuación de Laplace y también $\frac{\partial u}{\partial \underline{n}} = g_\partial$ con $g_\partial = 0$.

Ecuación de Onda Un problema general importante consiste en obtener la solución de la ecuación de onda, en el dominio del espacio-tiempo $\Omega \times [0, t]$, que satisface para cada $t \in (0, t]$ una condición de frontera de Robin en $\partial\Omega$ y las condiciones iniciales

$$u(\underline{x}, 0) = u_0(\underline{x}) \quad \text{y} \quad \frac{\partial u}{\partial t}(\underline{x}, 0) = v_0(\underline{x}), \quad \forall \underline{x} \in \Omega \tag{61}$$

aquí $u_0(\underline{x})$ y $v_0(\underline{x})$ son dos funciones prescritas. El hecho de que para la ecuación de onda se prescriban los valores iniciales, de la función y su derivada con respecto al tiempo, es reminiscente de que en la mecánica de partículas se necesitan las posiciones y las velocidades iniciales para determinar el movimiento de un sistema de partículas.

Ecuación de Calor También para la ecuación del calor un problema general importante consiste en obtener la solución de la ecuación de onda, en el dominio del espacio-tiempo $\Omega \times [0, t]$, que satisface para cada $t \in (0, t]$ una condición de frontera de Robin en y ciertas condiciones iniciales. Sin embargo, en este caso en ellas sólo se prescribe a la función

$$u(\underline{x}, 0) = u_0(\underline{x}), \quad \forall \underline{x} \in \Omega. \tag{62}$$

3.2. Análisis Funcional y Problemas Variacionales

Restringiéndonos a problemas elípticos, una cuestión central en la teoría de problemas elípticos con valores en la frontera se relaciona con las condiciones bajo las cuales uno puede esperar que el problema tenga solución y esta es única, así como conocer la regularidad de la solución.

Todas estas preguntas se pueden responder, pero se requiere utilizar herramientas de análisis funcional y de problemas variacionales con valor en la frontera. En esta sección tratamos de dar un panorama de esta teoría mostrando los resultados que la sustentan y las condiciones bajo las cuales un problema elíptico tiene solución y esta es única, además de conocer la regularidad de la solución, para mayor referencia de estos resultados ver [12], [18] y [3].

3.2.1. Espacios de Sobolev

En esta subsección detallaremos algunos resultados de los espacios de Sobolev sobre el conjunto de números reales, en estos espacios son sobre los cuales trabajaremos tanto para plantear el problema elíptico como para encontrar la solución al problema. Primeramente definiremos lo que entendemos por un espacio L^2.

Definición 4 *Una función medible $u(\underline{x})$ definida sobre $\Omega \subset \mathbb{R}^n$ se dice que pertenece al espacio $L^2(\Omega)$ si*

$$\int_{\Omega} |u(\underline{x})|^2 \, d\underline{x} < \infty \tag{63}$$

es decir, es integrable.

La definición de los espacios medibles, espacios L^p, distribuciones y derivadas de distribuciones están dados en el apéndice, estos resultados son la base para poder definir a los espacios de Sobolev.

Para poder expresar de forma compacta derivadas parciales de orden m o menor, usaremos la definición siguiente.

Definición 5 *Sea $\mathbb{Z}_+^n$ el conjunto de todas las n-duplas de enteros no negativos, un miembro de $\mathbb{Z}_+^n$ se denota usualmente por α ó β (por ejemplo $\alpha = (\alpha_1, \alpha_2, ..., \alpha_n)$). Denotaremos por $|\alpha|$ la suma $|\alpha| = \alpha_1 + \alpha_2 + ... + \alpha_n$ y por $D^\alpha u$ la derivada parcial*

$$D^\alpha u = \frac{\partial^{|\alpha|} u}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} ... \partial x_n^{\alpha_n}} \tag{64}$$

así, si $|\alpha| = m$, entonces $D^\alpha u$ denota la m-ésima derivada parcial de u.

Entonces haciendo uso de la definición anterior, definimos lo que entendemos por un espacio de Sobolev.

Definición 6 *El espacio de Sobolev de orden m, denotado por $H^m(\Omega)$, es definido como el espacio que consiste de todas las funciones en $L^2(\Omega)$ que satisfacen*

$$H^m(\Omega) = \left\{ u \mid D^\alpha u \in L^2(\Omega) \quad \forall \alpha \text{ tal que } |\alpha| \leq m \right\}. \tag{65}$$

El producto escalar $\langle \cdot, \cdot \rangle$ *de dos elementos* u *y* $v \in H^m(\Omega)$ *esta dado por*

$$\langle u, v \rangle_{H^m} = \int_\Omega \sum_{|\alpha| \leq m} \left(D^\alpha u\right) \left(D^\alpha v\right) d\underline{x} \text{ para } u, v \in H^m\left(\Omega\right). \tag{66}$$

Nota: Es común que el espacio $L^2(\Omega)$ sea denotado por $H^0(\Omega)$.

Un espacio completo con producto interior es llamado un espacio de Hilbert, un espacio normado y completo es llamado espacio de Banach. Y como todo producto interior define una norma, entonces todo espacio de Hilbert es un espacio de Banach.

Definición 7 *La norma* $\|\cdot\|_{H^m}$ *inducida a partir del producto interior* $\langle \cdot, \cdot \rangle_{H^m}$ *queda definida por*

$$\|u\|_{H^m}^2 = \langle u, u \rangle_{H^m} = \int_\Omega \sum_{|\alpha| \leq m} \left(D^\alpha u\right)^2 d\underline{x}. \tag{67}$$

Ahora, con norma $\|\cdot\|_{H^m}$, el espacio $H^m(\Omega)$ es un espacio de Hilbert, esto queda plasmado en el siguiente resultado.

Teorema 8 *El espacio* $H^m(\Omega)$ *con la norma* $\|\cdot\|_{H^m}$ *es un espacio de Hilbert.*

Ya que algunas de las propiedades de los espacios de Sobolev sólo son validas cuando la frontera del dominio es suficientemente suave. Para describir al conjunto donde los espacios de Sobolev están definidos, es común pedirle algunas propiedades y así definimos lo siguiente.

Definición 9 *Una función* f *definida sobre un conjunto* $\Gamma \subset \mathbb{R}^n$ *es llamada Lipschitz continua si existe una constante* $L > 0$ *tal que*

$$|f(x) - f(y)| \leq L\,|x - y| \quad \forall x, y \in \Gamma. \tag{68}$$

Notemos que una función Lipschitz continua es uniformemente continua.

Definición 10 *Entenderemos por un dominio al conjunto* $\Omega \subset \mathbb{R}^n$ *que sea abierto y conexo.*

Sea $\Omega \subset \mathbb{R}^n$ $(n \geq 2)$ un dominio con frontera $\partial\Omega$, sea $x_0 \in \partial\Omega$ y construyamos la bola abierta con centro en x_0 y radio ε, i.e. $B(x_0, \varepsilon)$, entonces definiremos el sistema coordenado $(\xi_1, ..., \xi_n)$ tal que el segmento $\partial\Omega \cap B(x_0, \varepsilon)$ pueda expresarse como una función

$$\xi_n = f(\xi_1, ..., \xi_{n-1}) \tag{69}$$

entonces definimos.

Definición 11 *La frontera* $\partial\Omega$ *del dominio* Ω *es llamada de Lipschitz si* f *definida como en la Ec. (69) es una función Lipschitz continua.*

El siguiente teorema resume las propiedades más importantes de los espacios de Sobolev $H^m(\Omega)$.

Teorema 12 *Sea $H^m(\Omega)$ el espacio de Sobolev de orden m y sea $\Omega \subset \mathbb{R}^n$ un dominio acotado con frontera Lipschitz. Entonces*

i) $H^r(\Omega) \subset H^m(\Omega)$ si $r \geq m$

ii) $H^m(\Omega)$ es un espacio de Hilbert con respecto a la norma $\|\cdot\|_{H^m}$

iii) $H^m(\Omega)$ es la cerradura con respecto a la norma $\|\cdot\|_{H^m}$ del espacio $C^\infty(\overline{\Omega})$.

De la parte *iii)* del teorema anterior, se puede hacer una importante interpretación: Para toda $u \in H^m(\Omega)$ es siempre posible encontrar una función infinitamente diferenciable f, tal que este arbitrariamente cerca de u en el sentido que

$$\|u - f\|_{H^m} < \varepsilon$$

para algún $\varepsilon > 0$ dado.

Cuando $m = 0$, se deduce la propiedad $H^0(\Omega) = L^2(\Omega)$ a partir del teorema anterior.

Corolario 13 *El espacio $L^2(\Omega)$ es la cerradura, con respecto a la norma L^2, del espacio $C^\infty(\overline{\Omega})$.*

Otra propiedad, se tiene al considerar a cualquier miembro de $u \in H^m(\Omega)$, este puede ser identificado con una función en $C^m(\overline{\Omega})$, después de que posiblemente sean cambiados algunos valores sobre un conjunto de medida cero, esto queda plasmado en los dos siguientes resultados.

Teorema 14 *Sean X y Y dos espacios de Banach, con $X \subset Y$. Sea $i : X \to Y$ tal que $i(u) = u$. Si el espacio X tiene definida la norma $\|\cdot\|_X$ y el espacio Y tiene definida la norma $\|\cdot\|_Y$, decimos que X está inmersa continuamente en Y si*

$$\|i(u)\|_Y = \|u\|_Y \leq K \|u\|_X$$

para alguna constante $K > 0$.

Teorema 15 *(Inmersión de Sobolev)*

Sea $\Omega \subset \mathbb{R}^n$ un dominio acotado con frontera $\partial\Omega$ de Lipschitz. Si $(m - k) > n/2$, entonces toda función en $H^m(\Omega)$ pertenece a $C^k(\overline{\Omega})$, es decir, hay un miembro que pertenece a $C^k(\overline{\Omega})$. Además, la inmersión

$$H^m(\Omega) \subset C^k(\overline{\Omega}) \tag{70}$$

es continua.

Traza de una Función en $H^m(\Omega)$. Una parte fundamental en los problemas con valores en la frontera definidos sobre el dominio Ω, es definir de forma única los valores que tomará la función sobre la frontera $\partial\Omega$, en este apartado veremos bajo que condiciones es posible tener definidos de forma única los valores en la frontera $\partial\Omega$ tal que podamos definir un operador $tr(\cdot)$ continuo que actué en $\overline{\Omega}$ tal que $tr(u) = u_{|\partial\Omega}$.

El siguiente lema nos dice que el operador $tr(\cdot)$ es continuo de $C^1\left(\overline{\Omega}\right)$ a $C(\partial\Omega)$.

Lema 16 *Sea Ω un dominio con frontera $\partial\Omega$ de Lipschitz. La estimación*

$$\|tr(u)\|_{L^2(\partial\Omega)} \leq C\|u\|_{H^1(\Omega)}$$

se satisface para toda función $u \in C^1\left(\overline{\Omega}\right)$, para alguna constante $C > 0$.

Ahora, para el caso $tr(\cdot) : H^1(\Omega) \to L^2(\partial\Omega)$, se tiene el siguiente teorema.

Teorema 17 *Sea Ω un dominio acotado en $\mathbb{R}^n$ con frontera $\partial\Omega$ de Lipschitz. Entonces:*

i) Existe un único operador lineal acotado $tr(\cdot) : H^1(\Omega) \to L^2(\partial\Omega)$, tal que

$$\|tr(u)\|_{L^2(\partial\Omega)} \leq C\|u\|_{H^1(\Omega)},$$

con la propiedad que si $u \in C^1\left(\overline{\Omega}\right)$, entonces $tr(u) = u|_{\partial\Omega}$.

ii) El rango de $tr(\cdot)$ es denso en $L^2(\partial\Omega)$.

Definición de los Espacios $H_0^m(\Omega)$. Los espacio $H_0^m(\Omega)$ surgen comúnmente al trabajar con problemas con valor en la frontera y serán aquellos espacios que se nulifiquen en la frontera del dominio, es decir:

Definición 18 *Sea una función u definida sobre un dominio $\Omega \subset \mathbb{R}^n$ con frontera $\partial\Omega$, definimos*

a) $H_0^1(\Omega) = \left\{u \in H^1(\Omega) : u = 0 \text{ sobre } \partial\Omega\right\}$

b) $H_0^2(\Omega) = \left\{u \in H^2(\Omega) : u = \partial_1 u = 0 \text{ sobre } \partial\Omega\right\}$

c) $H_0^m(\Omega) = \{u \in H^m(\Omega) : u = \partial_1 u = ... = \partial_{m-1} u = 0 \text{ sobre } \partial\Omega\}$.

Las propiedades básicas de estos espacios están contenidas en el siguiente resultado.

Teorema 19 *Sea Ω un dominio acotado en $\mathbb{R}^n$ con frontera $\partial\Omega$ suficientemente suave y sea $H_0^m(\Omega)$ la cerradura de $C_0^\infty(\Omega)$ en la norma $\|\cdot\|_{H^m}$, entonces*

a) $H_0^m(\Omega)$ es la cerradura de $C_0^\infty(\Omega)$ en la norma $\|\cdot\|_{H^m}$;

b) $H_0^m(\Omega) \subset H^m(\Omega)$;

c) Si $u \in H^m(\Omega)$ pertenece a $H_0^m(\Omega)$, entonces

$$D^\alpha u = 0, \text{ sobre } \partial\Omega, |\alpha| \leq m-1.$$

3.2.2. Formulas de Green y Problemas Adjuntos

Una cuestión central en la teoría de problemas elípticos con valores en la frontera se relaciona con las condiciones bajo las cuales uno puede esperar una única solución a problemas de la forma

$$\begin{array}{rcl} \mathcal{L}u & = & f_\Omega \quad \text{en } \Omega \subset \mathbb{R}^n \\ & & \left.\begin{array}{c} B_o u = g_0 \\ B_1 u = g_1 \\ \vdots \\ B_{m-1} u = g_{m-1} \end{array}\right\} \quad \text{en } \partial\Omega \end{array} \tag{71}$$

donde $\mathcal{L}$ es un operador elíptico de orden $2m$, de forma

$$\mathcal{L}u = \sum_{|\alpha|\leq m} (-1)^{|\alpha|} D^\alpha \left(\sum_{|\beta|\leq m} a_{\alpha\beta}(\underline{x}) D^\beta u \right), \quad \underline{x} \in \Omega \subset \mathbb{R}^n \tag{72}$$

donde los coeficientes $a_{\alpha\beta}$ son funciones de $\underline{x}$ suaves y satisfacen las condiciones para que la ecuación sea elíptica, el conjunto $B_0, B_1, ..., B_{m-1}$ de operadores de frontera son de la forma

$$B_j u = \sum_{|\alpha|\leq q_j} b_\alpha^{(j)} D^\alpha u = g_j \tag{73}$$

y constituyen un conjunto de condiciones de frontera que cubren a $\mathcal{L}$. Los coeficientes $b_\alpha^{(j)}$ son asumidos como funciones suaves.

En el caso de problemas de segundo orden la Ec. (73) puede expresarse como una sola condición de frontera

$$Bu = \sum_{j=1}^{n} b_j \frac{\partial u}{\partial x_j} + cu = g \quad \text{en } \partial\Omega. \tag{74}$$

Antes de poder ver las condiciones bajo las cuales se garantice la existencia y unicidad es necesario introducir el concepto de formula de Green asociada con el operador $\mathcal{L}^*$, para ello definimos:

Definición 20 *Con el operador dado como en la Ec. (72), denotaremos por $\mathcal{L}^*$ al operador definido por*

$$\mathcal{L}^* u = \sum_{|\alpha|\leq m} (-1)^{|\alpha|} D^\alpha \left(\sum_{|\beta|\leq m} a_{\beta\alpha}(\underline{x}) D^\beta u \right) \tag{75}$$

y nos referiremos a $\mathcal{L}^*$ como el adjunto formal del operador $\mathcal{L}$.

La importancia del adjunto formal es que si aplicamos el teorema de Green (65) a la integral

$$\int_\Omega v\mathcal{L}u d\underline{x} \tag{76}$$

obtenemos

$$\int_{\Omega} v\mathcal{L}u d\underline{x} = \int_{\Omega} u\mathcal{L}^* v d\underline{x} + \int_{\partial\Omega} F(u,v) d\underline{s} \tag{77}$$

en la cual $F(u,v)$ representa términos de frontera que se nulifican al aplicar el teorema ya que la función $v \in H_0^1(\Omega)$. Si $\mathcal{L} = \mathcal{L}^*$; i.e. $a_{\alpha\beta} = a_{\beta\alpha}$ el operador es llamado de manera formal el auto-adjunto.

En el caso de problemas de segundo orden, dos sucesivas aplicaciones del teorema de Green (65) y obtenemos, para i y j fijos

$$\begin{aligned} -\int_{\Omega} v\frac{\partial}{\partial x_i}\left(a_{ij}\frac{\partial u}{\partial x_j}\right) d\underline{x} &= -\int_{\partial\Omega} v a_{ij}\frac{\partial u}{\partial x_j} n_i d\underline{s} + \int_{\Omega} a_{ij}\frac{\partial u}{\partial x_j}\frac{\partial v}{\partial x_i} d\underline{x} \\ &= -\int_{\partial\Omega}\left[v a_{ij}\frac{\partial u}{\partial x_j} n_i - u a_{ij}\frac{\partial v}{\partial x_i} n_j\right] d\underline{s} \\ &\quad -\int_{\Omega} u\frac{\partial}{\partial x_j}\left(a_{ij}\frac{\partial v}{\partial x_i}\right) d\underline{x}. \end{aligned} \tag{78}$$

Pero sumando sobre i y j, obtenemos de la Ec. (77)

$$\mathcal{L}^* v = -\sum_{i,j=1}^{n} \frac{\partial}{\partial x_i}\left(a_{ji}(\underline{x})\frac{\partial v}{\partial x_j}\right) \tag{79}$$

y

$$F(u,v) = -\sum_{i,j=1}^{n} a_{ij}\left(v\frac{\partial u}{\partial x_j} n_i - u\frac{\partial v}{\partial x_i} n_j\right) \tag{80}$$

tal que $\mathcal{L}$ es formalmente el auto-ajunto si $a_{ji} = a_{ij}$.

En lo que resta de la sección, daremos los pasos necesarios para poder conocer bajo que condiciones el problema elíptico con valores en la frontera de la Ec. (71) con $s \geq 2m$ tiene solución y esta es única. Para ello, necesitamos adoptar el lenguaje de la teoría de operadores lineales, algunos resultados clave de algebra lineal están detallados en el apéndice.

Primeramente denotemos $N(B_j)$ al espacio nulo del operador de frontera $B_j : H^s(\Omega) \to L^2(\Omega)$, entonces

$$N(B_j) = \{u \in H^s(\Omega) \mid B_j u = 0 \text{ en } \partial\Omega\} \tag{81}$$

para $j = 0, 1, 2, ..., m-1$.

Adicionalmente definimos al dominio del operador $\mathcal{L}$, como el espacio

$$\begin{aligned} D(\mathcal{L}) &= H^s(\Omega) \cap N(B_0) \cap ... \cap N(B_{m-1}) \\ &= \{u \in H^s(\Omega) \mid B_j u = 0 \text{ en } \partial\Omega, j = 0, 1, .., m-1\}. \end{aligned} \tag{82}$$

Entonces el problema elíptico con valores en la frontera de la Ec. (71) con $s \geq 2m$, puede reescribirse como, dado $\mathcal{L} : D(\mathcal{L}) \to H^{s-2m}(\Omega)$, hallar u que satisfaga

$$\mathcal{L}u = f_{\Omega} \quad \text{en } \Omega. \tag{83}$$

Lo primero que hay que determinar es el conjunto de funciones f_Ω en $H^{s-2m}(\Omega)$ para las cuales la ecuación anterior se satisface, i.e. debemos identificar el rango $R(\mathcal{L})$ del operador $\mathcal{L}$. Pero como nos interesa conocer bajo que condiciones la solución u es única, entonces podemos definir el núcleo $N(\mathcal{L})$ del operador $\mathcal{L}$ como sigue

$$\begin{aligned} N(\mathcal{L}) &= \{u \in D(\mathcal{L}) \mid \mathcal{L}u = 0\} \\ &= \{u \in H^s(\Omega) \mid \mathcal{L}u = 0 \quad \text{en } \Omega, B_j u = 0 \text{ en } \partial\Omega, j = 0, 1, .., m-1\}. \end{aligned} \tag{84}$$

Si el $N(\mathcal{L}) \neq \{0\}$, entonces no hay una única solución, ya que si u_0 es una solución, entonces $u_0 + w$ también es solución para cualquier $w \in N(\mathcal{L})$, ya que

$$\mathcal{L}(u_0 + w) = \mathcal{L}u_0 + \mathcal{L}w = \mathcal{L}u_0 = f_\Omega. \tag{85}$$

Así, los elementos del núcleo $N(\mathcal{L})$ de $\mathcal{L}$ deberán ser excluidos del dominio $D(\mathcal{L})$ del operador $\mathcal{L}$, para poder asegurar la unicidad de la solución u.

Si ahora, introducimos el complemento ortogonal $N(\mathcal{L})^\perp$ del núcleo $N(\mathcal{L})$ del operador $\mathcal{L}$ con respecto al producto interior L^p, definiéndolo como

$$N(\mathcal{L})^\perp = \{v \in D(\mathcal{L}) \mid (v, w) = 0 \ \forall w \in N(\mathcal{L})\}. \tag{86}$$

De esta forma tenemos que

$$D(\mathcal{L}) = N(\mathcal{L}) \oplus N(\mathcal{L})^\perp \tag{87}$$

i.e. para toda $u \in D(\mathcal{L})$, u se escribe como $u = v + w$ donde $v \in N(\mathcal{L})^\perp$ y $w \in N(\mathcal{L})$. Además $N(\mathcal{L}) \cap N(\mathcal{L})^\perp = \{0\}$.

De forma similar, podemos definir los espacios anteriores para el problema adjunto

$$\begin{aligned} \mathcal{L}^* u &= f_\Omega \quad \text{en } \Omega \subset \mathbb{R}^n \\ &\left.\begin{array}{c} B_o^* u = g_0 \\ B_1^* u = g_1 \\ \vdots \\ B_{m-1}^* u = g_{m-1} \end{array}\right\} \quad \text{en } \partial\Omega \end{aligned} \tag{88}$$

y definimos

$$\begin{aligned} D(\mathcal{L}^*) &= H^s(\Omega) \cap N(B_0^*) \cap ... \cap N(B_{m-1}^*) \\ &= \left\{u \in H^s(\Omega) \mid B_j^* u = 0 \text{ en } \partial\Omega, j = 0, 1, .., m-1\right\}. \end{aligned} \tag{89}$$

Entonces el problema elíptico con valores en la frontera de la Ec. (71) con $s \geq 2m$, puede reescribirse como, dado $\mathcal{L}^* : D(\mathcal{L}^*) \to H^{s-2m}(\Omega)$, hallar u que satisfaga

$$\mathcal{L}^* u = f_\Omega \quad \text{en } \Omega. \tag{90}$$

Definiendo para el operador $\mathcal{L}^*$

$$\begin{aligned} N(\mathcal{L}^*) &= \{u \in D(\mathcal{L}^*) \mid \mathcal{L}^* u = 0\} \\ &= \left\{u \in H^s(\Omega) \mid \mathcal{L}^* u = 0 \quad \text{en } \Omega, B_j^* u = 0 \text{ en } \partial\Omega, j = 0, 1, .., m-1\right\}. \end{aligned} \tag{91}$$

y

$$N(\mathcal{L}^*)^{\perp} = \{v \in D(\mathcal{L}^*) \mid (v, w)_{H^0} = 0 \ \forall w \in N(\mathcal{L}^*)\}.$$

Así, con estas definiciones, es posible ver una cuestión fundamental, esta es, conocer bajo que condiciones el problema elíptico con valores en la frontera de la Ec. (71) con $s \geq 2m$ tiene solución y esta es única, esto queda resuelto en el siguiente teorema cuya demostración puede verse en [3] y [12].

Teorema 21 *Considerando el problema elíptico con valores en la frontera de la Ec. (71) con $s \geq 2m$ definido sobre un dominio Ω acotado con frontera $\partial\Omega$ suave. Entonces*

i) Existe al menos una solución si y sólo si $f \in N(\mathcal{L}^*)^{\perp}$, esto es, si

$$(f, v)_{H^o(\Omega)} = 0 \quad \forall v \in N(\mathcal{L}^*). \tag{92}$$

ii) Asumiendo que la solución u existe, esta es única si $u \in N(\mathcal{L})^{\perp}$, esto es, si

$$(u, w)_{H^0(\Omega)} = 0 \quad \forall w \in N(\mathcal{L}). \tag{93}$$

iii) Si existe una única solución, entonces existe una única constante $C > 0$, independiente de u, tal que

$$\|u\|_{H^s} \leq C \|f\|_{H^{s-2m}}. \tag{94}$$

3.2.3. Problemas Variacionales con Valor en la Frontera

Restringiéndonos ahora en problemas elípticos de orden 2 (problemas de orden mayor pueden ser tratados de forma similar), reescribiremos este en su forma variacional, ya que en esta forma se facilita su tratamiento por los métodos numéricos de ecuaciones diferenciales parciales y veremos algunos resultados clave como es la existencia y unicidad de la solución de este tipo de problemas, para mayores detalles, ver [3] y [12].

Si el operador $\mathcal{L}$ está definido por

$$\mathcal{L}u = -\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu \tag{95}$$

con $\underline{\underline{a}}$ una matriz positiva definida, simétrica y $c \geq 0$, el problema queda escrito como

$$\begin{aligned} -\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu &= f_\Omega \quad \text{en } \Omega \\ u &= g \quad \text{en } \partial\Omega. \end{aligned} \tag{96}$$

Si multiplicamos a la ecuación $-\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu = f_\Omega$ por $v \in V = H_0^1(\Omega)$, obtenemos

$$-v\left(\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu\right) = v f_\Omega \tag{97}$$

aplicando el teorema de Green (65) obtenemos la Ec. (78), que podemos reescribir como

$$\int_\Omega \left(\nabla v \cdot \underline{\underline{a}} \cdot \nabla u + cuv\right) d\underline{x} = \int_\Omega v f_\Omega d\underline{x}. \tag{98}$$

Definiendo el operador bilineal

$$a(u,v) = \int_\Omega \left(\nabla v \cdot \underline{\underline{a}} \cdot \nabla u + cuv\right) d\underline{x} \tag{99}$$

y la funcional lineal

$$l(v) = \langle f, v\rangle = \int_\Omega v f_\Omega d\underline{x} \tag{100}$$

podemos reescribir el problema dado por la Ec. (71) de orden 2, haciendo uso de la forma bilineal $a(\cdot,\cdot)$ y la funcional lineal $l(\cdot)$. Entonces entenderemos en el presente contexto un problema variacional con valores de frontera (VBVP) por uno de la forma: hallar una función u que pertenezca a un espacio de Hilbert $V = H_0^1(\Omega)$ y que satisfaga la ecuación

$$a(u,v) = \langle f, v\rangle \tag{101}$$

para toda función $v \in V$.

Definición 22 *Una forma bilineal $a(\cdot,\cdot)$ es V-elíptica si existe una constante $\alpha > 0$ tal que*

$$a(v,v) \geq \alpha \|v\|_V^2 \quad \forall v \in V \tag{102}$$

donde $\|\cdot\|_V$ es la norma asociada al espacio V.

Esto significa que una forma $V-$ elíptica es una que siempre es no negativa y toma el valor de 0 sólo en el caso de que $v = 0$, i.e. es positiva definida.

Notemos que el problema (96) definido en $V = H_0^1(\Omega)$ reescrito como el problema (101) genera una forma bilineal V-elíptica cuyo producto interior sobre V es simétrico y positivo definido ya que

$$a(v,v) \geq \alpha \|v\|_V^2 > 0, \quad \forall v \in V, v \neq 0 \tag{103}$$

reescribiéndose el problema (101), en el cual debemos encontrar $u \in V$ tal que

$$a(u,v) = \langle f, v\rangle - a(u_0, v) \tag{104}$$

donde $u_0 = g$ en $\partial\Omega$, para toda $v \in V$.

Entonces, la cuestión fundamental, es conocer bajo que condiciones el problema anterior tiene solución y esta es única, el teorema de Lax-Milgram nos da las condiciones bajo las cuales el problema (96) reescrito como el problema (101) tiene solución y esta es única, esto queda plasmado en el siguiente resultado.

Teorema 23 *(Lax-Milgram)*

Sea V un espacio de Hilbert y sea $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ una forma bilineal continua V-elíptica sobre V. Además, sea $l(\cdot) : V \to \mathbb{R}$ una funcional lineal continua sobre V. Entonces

i) El VBVP de encontrar $u \in V$ que satisfaga

$$a(u,v) = \langle f, v\rangle, \forall v \in V \tag{105}$$

tiene una y sólo una solución;

ii) La solución depende continuamente de los datos, en el sentido de que

$$\|u\|_V \leq \frac{1}{\alpha}\|l\|_V \tag{106}$$

y α es la constante de la definición de V-elíptica.

Más específicamente, considerando ahora V un subespacio cerrado de $H^m(\Omega)$ las condiciones para la existencia, unicidad y la dependencia continua de los datos queda de manifiesto en el siguiente resultado.

Teorema 24 *Sea V un subespacio cerrado de $H^m(\Omega)$, sea $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ una forma bilineal continua V-elíptica sobre V y sea $l(\cdot) : V \to \mathbb{R}$ una funcional lineal continua sobre V. Sea P un subespacio cerrado de V tal que*

$$a(u+p, v+\overline{p}) = a(u,v) \quad \forall u, v \in V \text{ y } p, \overline{p} \in P. \tag{107}$$

También denotando por Q el subespacio de V consistente de las funciones ortogonales a P en la norma L^2; tal que

$$Q = \left\{ v \in V \mid \int_\Omega upd\underline{x} = 0 \quad \forall p \in P \right\}, \tag{108}$$

y asumiendo que $a(\cdot,\cdot)$ es Q-elíptica: existe una constante $\alpha > 0$ tal que

$$a(q,q) \geq \alpha \|q\|_Q^2 \quad \text{para } q \in Q, \tag{109}$$

la norma sobre Q será la misma que sobre V. Entonces

i) Existe una única solución al problema de encontrar $u \in Q$ tal que

$$a(u,v) = \langle f, v\rangle, \quad \forall v \in V \tag{110}$$

si y sólo si las condiciones de compatibilidad

$$\langle f, q\rangle = 0 \quad \text{para } p \in P \tag{111}$$

se satisface.

ii) La solución u satisface

$$\|u\|_Q \leq \alpha^{-1}\|l\|_{Q^*} \tag{112}$$

(dependencia continua de los datos).

Otro aspecto importante es la regularidad de la solución, si la solución u al VBVP de orden $2m$ con $f \in H^{s-2m}(\Omega)$ donde $s \geq 2m$, entonces u pertenecerá a $H^s(\Omega)$ y esto queda de manifiesto en el siguiente resultado.

Teorema 25 *Sea $\Omega \subset \mathbb{R}^n$ un dominio suave y sea $u \in V$ la solución al VBVP*

$$a(u, v) = \langle f, v \rangle, \quad v \in V \tag{113}$$

donde $V \subset H^m(\Omega)$. Si $f \in H^{s-2m}(\Omega)$ con $s \geq 2m$, entonces $u \in H^s(\Omega)$ y la estimación

$$\|u\|_{H^s} \leq C \|f\|_{H^{s-2m}} \tag{114}$$

se satisface.

4. El Método Galerkin y el Método de Elemento Finito

En el presente capítulo se prestará atención a varios aspectos necesarios para encontrar la solución aproximada de problemas con valor en la frontera (VBVP). Ya que en general encontrar la solución a problemas con geometría diversa es difícil y en algunos casos imposible usando métodos analíticos. Por ello se explorará el método Galerkin para obtener la solución aproximada al problema de ecuaciones diferenciales parciales. El método de elemento finito está basado en la formulación del método Galerkin, tomando como punto de partida la formulación variacional.

Pero hay que notar que existen una gama amplia de métodos para la solución de VBVP, algunos derivados del método Galerkin y otros que utilizan otro enfoque a los vistos en el presente trabajo, como los desarrollados en [16] conocidos como métodos de Trefftz-Herrera.

En este capítulo se considera el VBVP de la forma

$$\begin{aligned} \mathcal{L}u &= f_\Omega \quad \text{en } \Omega \\ u &= g \quad \text{en } \partial\Omega \end{aligned} \tag{115}$$

donde

$$\mathcal{L}u = -\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu \tag{116}$$

con $\underline{\underline{a}}$ una matriz positiva definida, simétrica y $c \geq 0$, como un caso particular del operador elíptico definido por la Ec. (71) de orden 2, con $\Omega \subset R^2$ un dominio poligonal, es decir, Ω es un conjunto abierto acotado y conexo tal que $\bar{\Omega}$ es la unión de un número finito de polígonos.

La sencillez del operador $\mathcal{L}$ nos permite facilitar la comprensión de muchas de las ideas básicas que se expondrán a continuación, pero tengamos en mente que esta es una ecuación que gobierna los modelos de muchos sistemas de la ciencia y la ingeniería, por ello es muy importante su solución. Y que al considerar al operador definido así, nos permite garantizar que la forma bilineal asociada será simétrica y definida positiva, lo cual como será visto posteriormente permite usar una serie de herramientas que sacan ventaja del operador, como es el uso de el método de gradiente conjugado usado para encontrar la solución del sistema algebraico de ecuaciones generado al buscar la solución aproximada al VBVP.

4.1. Método Galerkin

Si multiplicamos a la ecuación $-\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu = f_\Omega$ por $v \in V = H_0^1(\Omega)$, obtenemos

$$-v\left(\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu\right) = v f_\Omega \tag{117}$$

aplicando el teorema de Green (65) obtenemos la Ec. (78), que podemos reescribir como

$$\int_\Omega \left(\nabla v \cdot \underline{\underline{a}} \cdot \nabla u + cuv\right) d\underline{x} = \int_\Omega v f_\Omega d\underline{x}. \tag{118}$$

Definiendo el operador bilineal

$$a(u,v) = \int_{\Omega} \left(\nabla v \cdot \underline{\underline{a}} \cdot \nabla u + cuv \right) d\underline{x} \tag{119}$$

y la funcional lineal

$$l(v) = \langle f, v \rangle = \int_{\Omega} v f_{\Omega} d\underline{x} \tag{120}$$

podemos reescribir el problema dado por la Ec. (115) de orden 2 en forma variacional, haciendo uso de la forma bilineal $a(\cdot,\cdot)$ y la funcional lineal $l(\cdot)$. La idea básica detrás del método Galerkin es, considerando el VBVP, encontrar $u \in V = H_0^1(\Omega)$ que satisfaga

$$a(u,v) = \langle f, v \rangle \quad \forall v \in V \tag{121}$$

donde V es un subespacio de un espacio de Hilbert H (por conveniencia nos restringiremos a espacios definidos sobre los números reales). El problema al tratar de resolver la Ec. (121) está en el hecho de que el espacio V es de dimensión infinita, por lo que resulta que en general no es posible encontrar el conjunto solución.

En lugar de tener el problema en el espacio V, se supone que se tienen funciones linealmente independientes $\phi_1, \phi_2, ..., \phi_N$ en V y definimos el espacio V^h a partir del subespacio dimensionalmente finito de V generado por las funciones ϕ_i, es decir,

$$V^h = Generado\{\phi_i\}_{i=1}^N, \quad V^h \subset V. \tag{122}$$

El índice h es un parámetro que estará entre 0 y 1, cuya magnitud da alguna indicación de cuan cerca V^h esta de V, h se relaciona con la dimensión de V^h. Y como el número N de las funciones base se escoge de manera que sea grande y haga que h sea pequeño. En el límite, cuando $N \to \infty$, $h \to 0$, se puede elegir $\{\phi_i\}$ tal que V^h se aproxime a V de la forma que se detallará posteriormente.

Después de definir el espacio V^h, es posible trabajar con V^h en lugar de V y encontrar una función u_h que satisfaga

$$a(u_h, v_h) = \langle f, v_h \rangle \quad \forall v_h \in V^h. \tag{123}$$

Esta es la esencia del método Galerkin, notemos que u_h y v_h son sólo combinaciones lineales de las funciones base de V^h, tales que

$$u_h = \sum_{i=1}^{N} c_i \phi_i \quad \text{y} \quad v_h = \sum_{j=1}^{N} d_j \phi_j \tag{124}$$

donde v_h es arbitraria, como los coeficientes de d_j y sin perdida de generalidad podemos hacer $v_h = \phi_j$. Así, para encontrar la solución u_h sustituimos las Ecs. (124) en la Ec. (123) y usando el hecho que $a(\cdot,\cdot)$ es una forma bilineal y $l(\cdot)$ es una funcional lineal se obtiene la ecuación

$$\sum_{i=1}^{N} a\left(\phi_i, \phi_j\right) c_i = \left\langle f, \phi_j \right\rangle \tag{125}$$

o más concisamente, como

$$\sum_{i=1}^{N} K_{ij} c_i - F_j = 0 \tag{126}$$

en la cual

$$K_{ij} = a\left(\phi_i, \phi_j\right) \quad \text{y} \quad F_j = \left\langle f, \phi_j \right\rangle \tag{127}$$

notemos que tanto K_{ij} y F_j pueden ser evaluados, ya que ϕ_i, $a\left(\cdot,\cdot\right)$ y $l\left(\cdot\right)$ son conocidas.

Entonces el problema se reduce a resolver el sistema de ecuaciones lineales

$$\sum_{i=1}^{N} K_{ij} c_i - F_j, \quad j = 1, 2, ...N \tag{128}$$

o más compactamente

$$\underline{\underline{\mathbb{K}}} \underline{u} = \underline{F} \tag{129}$$

en la cual $\underline{\underline{\mathbb{K}}}$ y $\underline{F}$ son la matriz y el vector cuyas entradas son K_{ij} y F_j. Una vez que el sistema es resuelto, la solución aproximada u_h es encontrada.

Notemos que el problema (115) definido en $V^h = H_0^1\left(\Omega\right)$ reescrito como el problema (121) genera una forma bilineal V^h-elíptica cuyo producto interior sobre V^h es simétrico y positivo definido ya que

$$a\left(v_h, v_h\right) \geq \alpha \left\|v_h\right\|_{V^h}^2 > 0, \quad \forall v_h \in V^h, v_h \neq 0 \tag{130}$$

reescribiéndose el problema (123) como el problema aproximado en el cual debemos encontrar $u_h \in V^h \subset V$ tal que

$$a\left(u_h, v_h\right) = \left\langle f, v_h \right\rangle - a\left(u_0, v_h\right) \tag{131}$$

donde $u_0 = g = 0$ en $\partial\Omega$, para toda $v_h \in V^h$, es decir

$$\int_\Omega \left(\nabla v_h \cdot \underline{\underline{a}} \cdot \nabla u_h + c u_h v_h\right) dxdy = \int_\Omega f_\Omega v_h dxdy \tag{132}$$

para todo $v_h \in V^h$.

Entonces, el problema (115) al aplicarle el método Galerkin obtenemos (118), el cual podemos reescribirlo como (132). Aplicando el teorema de Lax-Milgram (23) a este caso particular, tenemos que este tiene solución única y esta depende continuamente de los datos.

El siguiente resultado nos da una condición suficiente para que la aproximación u_h del método Galerkin converja a la solución u del problema dado por la Ec. (121), para más detalle véase [12] y [3].

Teorema 26 *Sea V un subespacio cerrado de un espacio de Hilbert, y sea la forma bilineal $a\left(\cdot,\cdot\right) : V^h \times V^h \to \mathbb{R}$ continua V-elíptica y sea $l\left(\cdot\right)$ una funcional lineal acotada. Entonces existe una constante C, independiente de h, tal que*

$$\left\|u - u_h\right\|_V \leq C \inf_{v_h \in V^h} \left\|u - v_h\right\|_V \tag{133}$$

donde u es solución de (121) y u_h es solución de (131), consecuentemente, una condición suficiente para que la aproximación u_h del método Galerkin converge a la solución u del problema dado por la Ec. (121) es que exista una familia $\{V^h\}$ de subespacios con la propiedad de que

$$\inf_{v_h \in V^h} \|u - v_h\|_V \to 0 \quad cuando \quad h \to 0. \tag{134}$$

La derivación de algunos otros métodos tomando como base al método Galerkin pueden consultarse en [3], [15], [25] y [18].

4.2. Método de Elemento Finito

El método de elementos finitos provee una manera sistemática y simple de generar las funciones base en un dominio con geometría Ω poligonal. Lo que hace al método de elemento finito especialmente atractivo sobre otros métodos, es el hecho de que las funciones base son polinomios definidos por pedazos (elementos Ω_i) que son no cero sólo en una pequeña parte de Ω, proporcionando a la vez una gran ventaja computacional al método ya que las matrices generadas resultan bandadas ahorrando memoria al implantarlas en una computadora.

Así, partiendo del problema aproximado (132), se elegirá una familia de espacios $V^h (h \in (0,1))$ definido por el procedimiento de elementos finitos (descritos en las subsecciones siguientes en el caso de interpoladores lineales, para otros tipos de interpoladores, ver [15]), teniendo la propiedad de que V^h se aproxima a V cuando h se aproxima a cero en un sentido apropiado, esto es, por supuesto una propiedad indispensable para la convergencia del método Galerkin.

Mallado del dominio Para comenzar con el método, dividimos el dominio $\Omega \subset \mathbb{R}^2$ en E subdominios o elementos Ω_e llamados elementos finitos, tal que

$$\overline{\Omega} = \bigcup_{e=1}^{E} \overline{\Omega}_e$$

donde:

- Cada Ω_e es un polígono (rectángulo o triángulo) con interior no vacío ($\mathring{\Omega}_e \neq \emptyset$).
- $\Omega_i \cap \Omega_j = \emptyset$ para cada $i \neq j$.
- El diámetro $Diam(\Omega_e) \leq h$ para cada $e = 1, 2, ..., E$.
- Los vértices de cada Ω_e son llamados nodos, teniendo N de ellos.

Funciones Base A continuación describiremos la manera de construir las funciones base usada por el método de elemento finito. En este procedimiento debemos tener en cuenta que las funciones base están definidas en un subespacio de $V = H^1(\Omega)$ para problemas de segundo orden que satisfacen las condiciones de frontera.

Las funciones base deberán satisfacer las siguientes propiedades:

i) Las funciones base ϕ_i son acotadas y continuas, i.e $\phi_i \in C(\Omega_e)$.

ii) Existen ℓ funciones base por cada nodo del polígono Ω_e, y cada función ϕ_i es no cero solo en los elementos contiguos conectados por el nodo i.

iii) $\phi_i = 1$ en cada i nodo del polígono Ω_e y cero en los otros nodos.

iv) La restricción ϕ_i a Ω_e es un polinomio, i.e. $\phi_i \in \mathbb{P}_k[\Omega_e]$ para alguna $k \geq 1$ donde $\mathbb{P}_k[\Omega_e]$ es el espacio de polinomios de grado a lo más k sobre Ω_e.

Decimos que $\phi_i \in \mathbb{P}_k[\Omega_e]$ es una base de funciones y por su construcción es evidente que estas pertenecen a $H^1(\Omega)$. Al conjunto formado por todas las funciones base definidas para todo Ω_e de Ω será el espacio $\mathbb{P}^h[k]$ de funciones base, i.e.

$$\mathbb{P}^h[k] = \bigcup_{e=1}^{E} \mathbb{P}_k[\Omega_e]$$

estas formarán las funciones base globales.

Solución aproximada Para encontrar la solución aproximada elegimos el espacio $\mathbb{P}^h[k]$ de funciones base, como el espacio de funciones lineales ϕ_i definidas por pedazos de grado menor o igual a k (en nuestro caso $k = 1$), entonces el espacio a trabajar es

$$V^h = Generado\left\{\phi_i \in \mathbb{P}^h[k] \;\mid \phi_i(x) = 0 \text{ en } \partial\Omega\right\}. \tag{135}$$

La solución aproximada de la Ec. (132) al problema dado por la Ec. (115) queda en términos de

$$\int_\Omega \left(\nabla\phi_i \cdot \underline{\underline{a}} \cdot \nabla\phi_j - c\phi_i\phi_j\right) dxdy = \int_\Omega f_\Omega \phi_j dxdy \tag{136}$$

si definimos el operador bilineal

$$K_{ij} \equiv a\left(\phi_i, \phi_j\right) = \int_\Omega \left(\nabla\phi_i \cdot \underline{\underline{a}} \cdot \nabla\phi_j - c\phi_i\phi_j\right) dxdy \tag{137}$$

y la funcional lineal

$$F_j \equiv \left\langle f, \phi_j \right\rangle = \int_\Omega f_\Omega \phi_j dxdy \tag{138}$$

entonces la matriz $\underline{\underline{K}} \equiv [K_{ij}]$, los vectores $\underline{u} \equiv (u_1, ..., u_N)$ y $\underline{F} \equiv (F_1, ..., F_N)$ definen el sistema lineal (que es positivo definido)

$$\underline{\underline{K}}\,\underline{u} = \underline{F} \tag{139}$$

donde $\underline{u}$ será el vector solución a la Ec. (139) cuyos valores serán la solución al problema dado por la Ec. (132) que es la solución aproximada a la Ec. (115) en los nodos interiores de Ω.

Un Caso más General Sea el operador elíptico (caso simétrico) en el dominio Ω, y el operador definido por

$$\begin{aligned} \mathcal{L}u &= f_\Omega \quad \text{en } \Omega\backslash\Sigma \\ u &= g \quad \text{en } \partial\Omega \\ [u]_\Sigma &= J_0 \\ [a_n \cdot \nabla u]_\Sigma &= J_1 \end{aligned} \tag{140}$$

donde

$$\mathcal{L}u = -\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu \tag{141}$$

conjuntamente con una partición $\prod = \{\Omega_1, ..., \Omega_E\}$ de Ω. Multiplicando por la función w obtenemos

$$w\mathcal{L}u = -w\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cwu = wf_\Omega \tag{142}$$

entonces si $w(x)$ es tal que $[w] = 0$ (es decir w es continua) y definimos

$$a(u,w) = \sum_{i=1}^{E} \int_{\Omega_i} \left(\nabla u \cdot \underline{\underline{a}} \cdot \nabla w + cwu\right) d\underline{x} \tag{143}$$

tal que $a(u,w)$ define un producto interior sobre

$$H^1(\Omega) = H^1(\Omega_1) \oplus H^1(\Omega_2) \oplus .... \oplus H^1(\Omega_E).$$

Entonces, reescribimos la Ec. (142) como

$$\begin{aligned} a(u,w) &= \int_\Omega wf d\underline{x} + \sum_{i=1}^{E} \int_{\partial\Omega} wa_n \cdot \nabla u d\underline{s} \\ &= \int_\Omega wf_\Omega d\underline{x} + \int_{\partial\Omega} wa_n \cdot \nabla u d\underline{s} - \int_\Sigma w\left[a_n \cdot \nabla u\right]_\Sigma d\underline{s}. \end{aligned} \tag{144}$$

Sea $u_0(x)$ una función que satisface las condiciones de frontera y J_0 una función que satisface las condiciones de salto, tal que

i) $u_0(x) = g(x)$ en $\partial\Omega$
ii) $[u_0(x)]_\Sigma = J_0$

y sea $u(x) = u_0(x) + v(x)$. Entonces $u(x)$ satisface la Ec. (143) si y sólo si $v(x)$ satisface

$$a(u,w) = \int_\Omega wf_\Omega d\underline{x} - \langle u_0, w\rangle - \int_\Sigma J_1 w d\underline{s} \tag{145}$$

para toda w tal que $w(x) = 0$ en $\partial\Omega$. Sea $\{\phi_i\}$ una base de un subespacio de dimensión finita V^h definido como

$$V^h = \left\{\phi_i \mid \phi_i \in C^1(\Omega_i), \forall i, \phi_i = 0 \text{ en } \partial\Omega \text{ y } \phi_i \in C^0(\Omega)\right\}. \tag{146}$$

La solución por elementos finitos de (145) se obtiene al resolver el sistema lineal

$$\underline{\underline{K}}\underline{u} = \underline{F} \tag{147}$$

donde

$$K_{ij} = a\left(\phi_i, \phi_j\right) \tag{148}$$

y

$$F_j = \int_\Omega \phi_j f_\Omega d\underline{x} - a\left(u_0, \phi_j\right) - \int_\Sigma J_1 \phi_j d\underline{s} \tag{149}$$

esta solución será la solución en los nodos interiores de Ω.

4.2.1. Discretización Usando Rectángulos

La discretización en dos dimensiones usando el método de elementos finitos puede hacerse mediante rectángulos o triángulos, cada una de ellas ofrece ventajas y desventajas que tienen que ser sopesadas al momento de la implementación del método.

La mayor ventaja de la discretización por medio de rectángulos es que su implementación es más simple, pero discretizar dominios poligonales no siempre es posible hacerlo correctamente con rectángulos, pero si con triángulos.

Aquí para ejemplificar, sólo la haremos usando rectángulos, pero es muy similar el caso con triángulos, para mayor información ver [15].

Para resolver la Ec. (115), dividimos $\Omega \subset \mathbb{R}^2$ en N_x nodos horizontales por N_y nodos verticales, teniendo $E = (N_x - 1)(N_y - 1)$ subdominios o elementos rectangulares Ω_e tales que $\Omega = \cup_{e=1}^{E} \Omega_e$ y $\Omega_i \cap \Omega_j \neq \varnothing$ si son adyacentes y $N = N_x N_y$ nodos.

Donde las funciones lineales definidas por pedazos en Ω_e serán en nuestro caso polinomios de orden uno en cada variable separadamente y cuya restricción de ϕ_i a Ω_e es $\phi_i^{(e)}$, entonces se tiene que para la integral del lado izquierdo de la Ec. (136),

$$\int_\Omega \left(\nabla\phi_i \cdot \underline{\underline{a}} \cdot \nabla\phi_j + c\phi_i\phi_j\right) dxdy = \int_\Omega f_\Omega \phi_j dxdy \tag{150}$$

queda expresada como

$$\begin{aligned} K_{ij} &= \int_\Omega \left(\nabla\phi_i \cdot \underline{\underline{a}} \cdot \nabla\phi_j + c\phi_i\phi_j\right) dxdy \qquad (151)\\ &= \sum_{e=1}^{E} \int_{\Omega_e} \left(\nabla\phi_i^{(e)} \cdot \underline{\underline{a}} \cdot \nabla\phi_j^{(e)} + c\phi_i^{(e)}\phi_j^{(e)}\right) dxdy \\ &= \sum_{e=1}^{E} \int_{\Omega_e} \left(\left[\frac{\partial\phi_i^{(e)}}{\partial x} \underline{\underline{a}} \frac{\partial\phi_j^{(e)}}{\partial x} + \frac{\partial\phi_i^{(e)}}{\partial y} \underline{\underline{a}} \frac{\partial\phi_j^{(e)}}{\partial y} \right] + c\phi_i^{(e)}\phi_j^{(e)} \right) dxdy \end{aligned}$$

y para la del lado derecho, queda expresada como

$$\begin{aligned} F_j &= \int_{\Omega} f_{\Omega}\phi_j dxdy \\ &= \sum_{e=1}^{E} \int_{\Omega_e} f_{\Omega}\phi_j^{(e)} dxdy. \end{aligned} \tag{152}$$

Para cada Ω_e de Ω, la submatriz de integrales

$$K_{ij} = \int_{\Omega_e} \left(\left[\frac{\partial \phi_i^{(e)}}{\partial x} \underline{\underline{a}} \frac{\partial \phi_j^{(e)}}{\partial x} + \frac{\partial \phi_i^{(e)}}{\partial y} \underline{\underline{a}} \frac{\partial \phi_j^{(e)}}{\partial y} \right] + c\phi_i^{(e)}\phi_j^{(e)} \right) dxdy \tag{153}$$

tiene la estructura

$$\begin{bmatrix} K_{1,1}^{(e)} & K_{1,2}^{(e)} & K_{1,3}^{(e)} & K_{1,4}^{(e)} \\ K_{2,1}^{(e)} & K_{2,2}^{(e)} & K_{2,3}^{(e)} & K_{2,4}^{(e)} \\ K_{3,1}^{(e)} & K_{3,2}^{(e)} & K_{3,3}^{(e)} & K_{3,4}^{(e)} \\ K_{4,1}^{(e)} & K_{4,2}^{(e)} & K_{4,3}^{(e)} & K_{4,4}^{(e)} \end{bmatrix}$$

la cual deberá ser ensamblada en la matriz global que corresponda a la numeración de nodos locales del elemento Ω_e con respecto a la numeración global de los elementos en Ω.

De manera parecida, para cada Ω_e de Ω se genera el vector de integrales

$$F_j = \int_{\Omega_e} f_{\Omega}\phi_j^{(e)} dxdy \tag{154}$$

con la estructura

$$\begin{bmatrix} F_1^{(e)} \\ F_2^{(e)} \\ F_3^{(e)} \\ F_4^{(e)} \end{bmatrix}$$

el cual también deberá ser ensamblado en el vector global que corresponda a la numeración de nodos locales al elemento Ω_e con respecto a la numeración global de los elementos de Ω.

Montando los $K_{ij}^{(e)}$ en la matriz $\underline{\underline{\mathbb{K}}}$ y los $F_j^{(e)}$ en el vector $\underline{\mathbb{F}}$ según la numeración de nodos global, se genera el sistema $\underline{\underline{\mathbb{K}}}\underline{u_h}=\underline{\mathbb{F}}$ donde $\underline{u_h}$ será el vector cuyos valores serán la solución aproximada a la Ec. (115) en los nodos interiores de Ω. La matriz $\underline{\underline{\mathbb{K}}}$ generada de esta forma, tiene una propiedad muy importante, es bandada y el ancho de banda es de 9 elementos, esto es muy útil al momento de soportar la matriz en memoria.

Para resolver numéricamente en cada Ω_e las integrales

$$\int_{\Omega_e} \left(\left[\frac{\partial \phi_i^{(e)}}{\partial x} \underline{\underline{a}} \frac{\partial \phi_j^{(e)}}{\partial x} + \frac{\partial \phi_i^{(e)}}{\partial y} \underline{\underline{a}} \frac{\partial \phi_j^{(e)}}{\partial y} \right] + c\phi_i^{(e)}\phi_j^{(e)} \right) dxdy \tag{155}$$

y

$$\int_{\Omega_e} f_\Omega \phi_j^{(e)} dxdy \tag{156}$$

y con la finalidad de simplificar los cálculos computacionales se considera a un elemento de referencia $\hat{\Omega}$ en los ejes coordenados (ε, η) cuyos vértices están el $(0,0),(1,0),(1,1)$ y $(0,1)$, en el cual mediante una función afín será proyectado cualquier elemento rectangular Ω_e (cuyos vértices $(x_1^{(e)}, y_1^{(e)}), (x_2^{(e)}, y_2^{(e)}), (x_3^{(e)}, y_3^{(e)})$ y $(x_4^{(e)}, y_4^{(e)})$ están tomados en sentido contrario al movimiento de las manecillas del reloj) por la transformación $f(\varepsilon, \eta) = \underline{\underline{T}}(\varepsilon, \eta) + \underline{b}$, quedando dicha transformación como

$$\begin{aligned} x &= \frac{x_2^{(e)} - x_1^{(e)}}{2}\varepsilon + \frac{x_1^{(e)} + x_2^{(e)}}{2} \\ y &= \frac{y_4^{(e)} - y_1^{(e)}}{2}\eta + \frac{y_1^{(e)} + y_4^{(e)}}{2} \end{aligned} \tag{157}$$

en la cual la matriz $\underline{\underline{T}}$ está dada por

$$\underline{\underline{T}} = \begin{pmatrix} \frac{x_2^{(e)} - x_1^{(e)}}{2} & \frac{y_2^{(e)} - y_1^{(e)}}{2} \\ \frac{x_4^{(e)} - x_1^{(e)}}{2} & \frac{y_4^{(e)} - y_1^{(e)}}{2} \end{pmatrix} \tag{158}$$

y el vector $\underline{b}$ es la posición del vector centroide del rectángulo Ω_e, también se tiene que la transformación inversa es

$$\begin{aligned} \varepsilon &= \frac{x - \frac{x_1^{(e)} + x_2^{(e)}}{2}}{\frac{x_2^{(e)} - x_1^{(e)}}{2}} \\ \eta &= \frac{y - \frac{y_1^{(e)} + y_4^{(e)}}{2}}{\frac{y_4^{(e)} - y_1^{(e)}}{2}}. \end{aligned} \tag{159}$$

Entonces las $\phi_i^{(e)}$ quedan definidas en términos de $\hat{\phi}_i$ como

$$\begin{aligned} \hat{\phi}_1(\varepsilon, \eta) &= \frac{1}{4}(1-\varepsilon)(1-\eta) \\ \hat{\phi}_2(\varepsilon, \eta) &= \frac{1}{4}(1+\varepsilon)(1-\eta) \\ \hat{\phi}_3(\varepsilon, \eta) &= \frac{1}{4}(1+\varepsilon)(1+\eta) \\ \hat{\phi}_4(\varepsilon, \eta) &= \frac{1}{4}(1-\varepsilon)(1+\eta) \end{aligned} \tag{160}$$

y las funciones $\phi_i^{(e)}$ son obtenidas por el conjunto $\phi_i^{(e)}(x,y) = \hat{\phi}_i(\varepsilon, \eta)$ con (x,y)

y (ε,η) relacionadas por la Ec. (157), generándose las integrales

$$\begin{aligned} K_{ij}^{(e)} &= \int_{\Omega_e} \left(\left[\frac{\partial \phi_i^{(e)}}{\partial x} \underline{a} \underline{=} \frac{\partial \phi_j^{(e)}}{\partial x} + \frac{\partial \phi_i^{(e)}}{\partial y} \underline{a} \underline{=} \frac{\partial \phi_j^{(e)}}{\partial y} \right] + c\phi_i^{(e)} \phi_j^{(e)} \right) dxdy \\ &= \int_{\hat{\Omega}} \left(\left[\left(\frac{\partial \hat{\phi}_i}{\partial \varepsilon} \underline{a} \underline{=} \frac{\partial \varepsilon}{\partial x} + \frac{\partial \hat{\phi}_i}{\partial \eta} \underline{a} \underline{=} \frac{\partial \eta}{\partial x} \right) \left(\frac{\partial \hat{\phi}_j}{\partial \varepsilon} \underline{a} \underline{=} \frac{\partial \varepsilon}{\partial x} + \frac{\partial \hat{\phi}_j}{\partial \eta} \underline{a} \underline{=} \frac{\partial \eta}{\partial x} \right) + \right. \right. \\ &\quad \left. \left. \left(\frac{\partial \hat{\phi}_i}{\partial \varepsilon} \underline{a} \underline{=} \frac{\partial \varepsilon}{\partial y} + \frac{\partial \hat{\phi}_i}{\partial \eta} \underline{a} \underline{=} \frac{\partial \eta}{\partial y} \right) \left(\frac{\partial \hat{\phi}_j}{\partial \varepsilon} \underline{a} \underline{=} \frac{\partial \varepsilon}{\partial y} + \frac{\partial \hat{\phi}_j}{\partial \eta} \underline{a} \underline{=} \frac{\partial \eta}{\partial y} \right) \right] + c\hat{\phi}_i \hat{\phi}_j \right) |J| \, d\varepsilon d\eta \end{aligned} \tag{161}$$

donde el índice i y j varia de 1 a 4. En está última usamos la regla de la cadena y $dxdy = |J| \, d\varepsilon d\eta$ para el cambio de variable en las integrales, aquí $|J| = \det T$, donde T está dado como en la Ec. (158). Para resolver $\int_{\Omega_e} f_\Omega \phi_j^{(e)} dxdy$ en cada Ω_e se genera las integrales

$$\begin{aligned} F_j^{(e)} &= \int_{\Omega_e} f_\Omega \phi_j^{(e)} dxdy \\ &= \int_{\hat{\Omega}} f_\Omega \hat{\phi}_j \, |J| \, d\varepsilon d\eta \end{aligned} \tag{162}$$

donde el índice i y j varia de 1 a 4.

Para realizar el cálculo numérico de las integrales en el rectángulo de referencia $\hat{\Omega} = [-1, 1] \times [-1, 1]$, debemos conocer $\frac{\partial \phi i}{\partial \varepsilon}$, $\frac{\partial \phi i}{\partial \eta}$, $\frac{\partial \varepsilon}{\partial x}$, $\frac{\partial \varepsilon}{\partial y}$, $\frac{\partial \eta}{\partial x}$ y $\frac{\partial \eta}{\partial y}$, entonces realizando las operaciones necesarias a la Ec. (160) obtenemos

$$\begin{array}{ll} \frac{\partial \phi_1}{\partial \varepsilon} = -\frac{1}{4}(1-\eta) & \frac{\partial \phi_1}{\partial \eta} = -\frac{1}{4}(1-\varepsilon) \\ \frac{\partial \phi_2}{\partial \varepsilon} = \frac{1}{4}(1-\eta) & \frac{\partial \phi_2}{\partial \eta} = -\frac{1}{4}(1+\varepsilon) \\ \frac{\partial \phi_3}{\partial \varepsilon} = \frac{1}{4}(1+\eta) & \frac{\partial \phi_3}{\partial \eta} = \frac{1}{4}(1+\varepsilon) \\ \frac{\partial \phi_4}{\partial \varepsilon} = -\frac{1}{4}(1+\eta) & \frac{\partial \phi_4}{\partial \eta} = \frac{1}{4}(1-\varepsilon) \end{array} \tag{163}$$

y también

$$\begin{array}{ll} \frac{\partial \varepsilon}{\partial x} = \begin{pmatrix} \frac{y_4^{(e)} - y_1^{(e)}}{2 \det T} \end{pmatrix} & \frac{\partial \varepsilon}{\partial y} = \begin{pmatrix} \frac{x_4^{(e)} - x_1^{(e)}}{2 \det T} \end{pmatrix} \\ \frac{\partial \eta}{\partial x} = \begin{pmatrix} \frac{y_2^{(e)} - y_1^{(e)}}{2 \det T} \end{pmatrix} & \frac{\partial \eta}{\partial y} = \begin{pmatrix} \frac{x_2^{(e)} - x_1^{(e)}}{2 \det T} \end{pmatrix} \end{array} \tag{164}$$

las cuales deberán de ser sustituidas en cada $\underline{\underline{K_{ij}^{(e)}}}$ y $\underline{F_j^{(e)}}$ para calcular las integrales en el elemento Ω_e. Estas integrales se harán en el programa usando cuadratura Gaussiana, permitiendo reducir el número de cálculos al mínimo pero manteniendo el balance entre precisión y número bajo de operaciones necesarias para realizar las integraciones.

Suponiendo que Ω fue dividido en E elementos, estos elementos generan N nodos en total, de los cuales N_d son nodos desconocidos y N_c son nodos

conocidos con valor γ_j, entonces el algoritmo de ensamble de la matriz $\underline{\underline{K}}$ y el vector $\underline{F}$ se puede esquematizar como:

$K_{i,j} = (\phi_i, \phi_j) \quad \forall i = 1, 2, ..., E, j = 1, 2, ..., E$

$F_j = (f_\Omega, \phi_j) \quad \forall j = 1, 2, ..., E$

$\forall j = 1, 2, ..., N_d$:

$$b_j = b_j - \gamma_i K_{i,j} \quad \forall i = 1, 2, ..., E$$

Así, se construye una matriz global en la cual están representados los nodos conocidos y los desconocidos, tomando sólo los nodos desconocidos de la matriz $\underline{\underline{K}}$ formaremos una matriz $\underline{\underline{A}}$, haciendo lo mismo al vector $\underline{F}$ formamos el vector $\underline{b}$, entonces la solución al problema será la resolución del sistema de ecuaciones lineales $\underline{\underline{A}}\underline{x} = \underline{b}$, este sistema puede resolverse usando por ejemplo el método de gradiente conjugado. El vector $\underline{x}$ contendrá la solución buscada en los nodos desconocidos N_d.

5. Solución de Grandes Sistemas de Ecuaciones

En el capítulo anterior se discutió como proceder para transformar un problema de ecuaciones diferenciales parciales con valores en la frontera en un sistema algebraico de ecuaciones y así poder hallar la solución resolviendo el sistema de ecuaciones lineales que se pueden expresar en la forma matricial siguiente

$$\underline{\underline{A}}\underline{u} = \underline{b} \tag{165}$$

donde la matriz $\underline{\underline{A}}$ es bandada (muchos elementos son nulos) y en problemas reales tiene grandes dimensiones.

La elección del método específico para resolver el sistema de ecuaciones depende de las propiedades particulares de la matriz $\underline{\underline{A}}$, en las siguientes secciones examinaremos varios métodos y sus implicaciones en cuanto al costo computacional de la resolución del sistema de ecuaciones. En términos generales, si el problema de ecuaciones diferenciales parciales con valores en la frontera en dos dimensiones se discretiza usando una malla de $n \times m$ nodos, el sistema algebraico de ecuaciones asociado es del orden de $(n \times m)^2$, pero en general la matriz $\underline{\underline{A}}$ resultante para el tipo de problemas de interés en el presente trabajo es bandada y definida positiva, por ello es posible hacer uso de estas características para solucionar el sistema algebraico de ecuaciones de forma óptima.

Los métodos de resolución del sistema algebraico de ecuaciones $\underline{\underline{A}}\underline{u} = \underline{b}$ se clasifican en dos grandes grupos: los métodos directos y los métodos iterativos.

En los métodos directos la solución $\underline{u}$ se obtiene en un número fijo de pasos y sólo están sujetos a los errores de redondeo. En los métodos iterativos, se realizan iteraciones para aproximarse a la solución $\underline{u}$ aprovechando las características propias de la matriz $\underline{\underline{A}}$, tratando de usar un menor número de pasos que en un método directo.

Los métodos iterativos rara vez se usan para resolver sistemas lineales de dimensión pequeña (el concepto de dimensión pequeña es muy relativo), ya que el tiempo necesario para conseguir una exactitud satisfactoria rebasa el que requieren los métodos directos. Sin embargo, en el caso de sistemas grandes con un alto porcentaje de elementos cero, son eficientes tanto en el almacenamiento en la computadora como en el tiempo que se invierte en su solución. Por ésta razón al resolver éstos sistemas algebraicos de ecuaciones es preferible aplicar los métodos iterativos tales como son: Jacobi, Gauss-Seidel, sobrerrelajación sucesiva (SOR), etc. Para más información de éstos y otros métodos, así como pruebas en la velocidad de convergencia y precisión, pueden consultarse en [15], [5], [7], [9], [17], [24], [25] y [19].

Cabe hacer mención de que la mayoría del tiempo de cómputo necesario para resolver el problema de ecuaciones diferenciales parciales (EDP), es consumido en la solución del sistema algebraico de ecuaciones asociado a la discretización, por ello es determinante elegir aquel método numérico que minimice el tiempo invertido en este proceso.

5.1. Métodos Directos

En estos métodos, la solución $\underline{u}$ se obtiene en un número fijo de pasos y sólo están sujetos a los errores de redondeo. Entre los métodos más importantes podemos encontrar: Eliminación Gausiana, descomposición LU, eliminación bandada y descomposición de Cholesky.

Los métodos antes mencionados, se colocaron en orden descendente en cuanto al consumo de recursos computacionales y ascendente en cuanto al aumento en su eficiencia; describiéndose a continuación:

Eliminación Gausiana Tal vez es el método más utilizado para encontrar la solución usando métodos directos. Este algoritmo sin embargo no es eficiente, ya que en general, un sistema de N ecuaciones requiere para su almacenaje en memoria de N^2 entradas para la matriz $\underline{\underline{A}}$, pero cerca de $N^3/3 + O(N^2)$ multiplicaciones y $N^3/3 + O(N^2)$ adiciones para encontrar la solución siendo muy costoso computacionalmente.

La eliminación Gausiana se basa en la aplicación de operaciones elementales a renglones o columnas de tal forma que es posible obtener matrices equivalentes.

Escribiendo el sistema de N ecuaciones lineales con N incógnitas como

$$\sum_{j=1}^{N} a_{ij}^{(0)} x_j = a_{i,n+1}^{(0)}, \quad i = 1, 2, ..., N \tag{166}$$

y si $a_{11}^{(0)} \neq 0$ y los pivotes $a_{ii}^{(i-1)}, i = 2, 3, ..., N$ de las demás filas, que se obtienen en el curso de los cálculos, son distintos de cero, entonces, el sistema lineal anterior se reduce a la forma triangular superior (eliminación hacia adelante)

$$x_i + \sum_{j=i+1}^{N} a_{ij}^{(i)} x_j = a_{i,n+1}^{(i)}, \quad i = 1, 2, ..., N \tag{167}$$

donde

$$\begin{aligned} k &= 1, 2, ..., N; \{j = k+1, ..., N\{ \\ a_{kj}^{(k)} &= \frac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}; \\ i &= k+1, ..., N+1\{ \\ a_{ij}^{(k)} &= a_{ij}^{(k-1)} - a_{kj}^{(k)} a_{ik}^{(k-1)}\}\}\} \end{aligned}$$

y las incógnitas se calculan por sustitución hacia atrás, usando las fórmulas

$$\begin{aligned} x_N &= a_{N,N+1}^{(N)}; \\ i &= N-1, N-2, ..., 1 \\ x_i &= a_{i,N+1}^{(i)} - \sum_{j=i+1}^{N} a_{ij}^{(i)} x_j. \end{aligned} \tag{168}$$

En algunos casos nos interesa conocer $\underline{\underline{A}}^{-1}$, por ello si la eliminación se aplica a la matriz aumentada $\underline{\underline{A}} \mid \underline{\underline{I}}$ entonces la matriz $\underline{\underline{A}}$ de la matriz aumentada se convertirá en la matriz $\underline{\underline{I}}$ y la matriz $\underline{\underline{I}}$ de la matriz aumentada será $\underline{\underline{A}}^{-1}$. Así, el sistema $\underline{\underline{A}}\underline{u} = \underline{b}$ se transformará en $\underline{u} = \underline{\underline{A}}^{-1}\underline{b}$ obteniendo la solución de $\underline{u}$.

Descomposición LU Sea $\underline{\underline{U}}$ una matriz triangular superior obtenida de $\underline{\underline{A}}$ por eliminación bandada. Entonces $\underline{\underline{U}} = \underline{\underline{L}}^{-1}\underline{\underline{A}}$, donde $\underline{\underline{L}}$ es una matriz triangular inferior con unos en la diagonal. Las entradas de $\underline{\underline{L}}^{-1}$ pueden obtenerse de los coeficientes m_{ij} definidos en el método anterior y pueden ser almacenados estrictamente en las entradas de la diagonal inferior de $\underline{\underline{A}}$ ya que estas ya fueron eliminadas. Esto proporciona una factorización $\underline{\underline{LU}}$ de $\underline{\underline{A}}$ en la misma matriz $\underline{\underline{A}}$ ahorrando espacio de memoria.

El problema original $\underline{\underline{A}}\underline{u} = \underline{b}$ se escribe como $\underline{\underline{LU}}\underline{u} = \underline{b}$ y se reduce a la solución sucesiva de los sistemas lineales triangulares

$$\underline{\underline{L}}\underline{y} = \underline{b} \quad \text{y} \quad \underline{\underline{U}}\underline{u} = \underline{y}. \tag{169}$$

La descomposición $\underline{\underline{LU}}$ requiere también $N^3/3$ operaciones aritméticas para la matriz llena, pero sólo Nb^2 operaciones aritméticas para la matriz con un ancho de banda de b siendo esto más económico computacionalmente.

Nótese que para una matriz no singular $\underline{\underline{A}}$, la eliminación de Gausiana (sin redondear filas y columnas) es equivalente a la factorización LU.

Eliminación Bandada Cuando se usa la ordenación natural de los nodos, la matriz $\underline{\underline{A}}$ que se genera es bandada, por ello se puede ahorrar considerable espacio de almacenamiento en ella. Este algoritmo consiste en triangular a la matriz $\underline{\underline{A}}$ por eliminación hacia adelante operando sólo sobre las entradas dentro de la banda central no cero. Así el renglón j es multiplicado por $m_{ij} = a_{ij}/a_{jj}$ y el resultado es restado al renglón i para $i = j+1, j+2, ...$

El resultado es una matriz triangular superior $\underline{\underline{U}}$ que tiene ceros abajo de la diagonal en cada columna. Así, es posible resolver el sistema resultante al sustituir en forma inversa las incógnitas.

Descomposición de Cholesky Cuando la matriz es simétrica y definida positiva, se obtiene la descomposición $\underline{\underline{LU}}$ de la matriz $\underline{\underline{A}}$, así $\underline{\underline{A}} = \underline{\underline{LDU}} = \underline{\underline{LDL}}^T$ donde $\underline{\underline{D}} = diag(\underline{\underline{U}})$ es la diagonal con entradas positivas. La mayor ventaja de esta descomposición es que, en el caso en que es aplicable, el costo de cómputo es sustancialmente reducido, ya que requiere de $N^3/6$ multiplicaciones y $N^3/6$ adiciones.

5.2. Métodos Iterativos

En estos métodos se realizan iteraciones para aproximarse a la solución $\underline{u}$ aprovechando las características propias de la matriz $\underline{\underline{A}}$, tratando de usar un

menor número de pasos que en un método directo, para más información de estos y otros métodos ver [15] y [24].

Un método iterativo en el cual se resuelve el sistema lineal

$$\underline{\underline{A}}\underline{u} = \underline{b} \tag{170}$$

comienza con una aproximación inicial $\underline{u}^0$ a la solución $\underline{u}$ y genera una sucesión de vectores $\left\{u^k\right\}_{k=1}^{\infty}$ que converge a $\underline{u}$. Los métodos iterativos traen consigo un proceso que convierte el sistema $\underline{\underline{A}}\underline{u} = \underline{b}$ en otro equivalente de la forma $\underline{u} = \underline{\underline{T}}\underline{u} + \underline{c}$ para alguna matriz fija $\underline{\underline{T}}$ y un vector $\underline{c}$. Luego de seleccionar el vector inicial $\underline{u}^0$ la sucesión de los vectores de la solución aproximada se genera calculando

$$\underline{u}^k = \underline{\underline{T}}\underline{u}^{k-1} + \underline{c} \quad \forall k = 1, 2, 3, ... \tag{171}$$

La convergencia a la solución la garantiza el siguiente teorema cuya solución puede verse en [25].

Teorema 27 *Si* $\left\|\underline{\underline{T}}\right\| < 1$*, entonces el sistema lineal* $\underline{u} = \underline{\underline{T}}\underline{u} + \underline{c}$ *tiene una solución única* $\underline{u}^*$ *y las iteraciones* $\underline{u}^k$ *definidas por la fórmula* $\underline{u}^k = \underline{\underline{T}}\underline{u}^{k-1} + \underline{c} \quad \forall k = 1, 2, 3, ...$ *convergen hacia la solución exacta* $\underline{u}^*$ *para cualquier aproximación lineal* $\underline{u}^0$.

Notemos que mientras menor sea la norma de la matriz $\underline{\underline{T}}$, más rápida es la convergencia, en el caso cuando $\left\|\underline{\underline{T}}\right\|$ es menor que uno, pero cercano a uno, la convergencia es muy lenta y el número de iteraciones necesario para disminuir el error depende significativamente del error inicial. En este caso, es deseable proponer al vector inicial $\underline{u}^0$ de forma tal que se mínimo el error inicial. Sin embargo, la elección de dicho vector no tiene importancia si la $\left\|\underline{\underline{T}}\right\|$ es pequeña ya que la convergencia es rápida.

Como es conocido, la velocidad de convergencia de los métodos iterativos dependen de las propiedades espectrales de la matriz de coeficientes del sistema de ecuaciones, cuando el operador diferencial $\mathcal{L}$ de la ecuación del problema a resolver es auto-adjunto se obtiene una matriz simétrica y positivo definida y el número de condicionamiento de la matriz $\underline{\underline{A}}$, es por definición

$$cond(\underline{\underline{A}}) = \frac{\lambda_{\text{máx}}}{\lambda_{\text{mín}}} \geq 1 \tag{172}$$

donde $\lambda_{\text{máx}}$ y $\lambda_{\text{mín}}$ es el máximo y mínimo de los eigenvalores de la matriz $\underline{\underline{A}}$. Si el número de condicionamiento es cercano a 1 los métodos numéricos al solucionar el problema convergerá en pocas iteraciones, en caso contrario se requerirán muchas iteraciones. Frecuentemente al usar el método de elemento finito se tiene una velocidad de convergencia de $O\left(\frac{1}{h^2}\right)$ y en el caso de métodos de descomposición de dominio se tiene una velocidad de convergencia de $O\left(\frac{1}{h}\right)$ en el mejor de los casos, donde h es la máxima distancia de separación entre nodos continuos de la partición, es decir, que poseen una pobre velocidad de convergencia cuando $h \to 0$, para más detalles ver [2].

Entre los métodos más usados para el tipo de problemas tratados en el presente trabajo podemos encontrar: Jacobi, Gauss-Seidel, Richardson, relajación sucesiva, gradiente conjugado, gradiente conjugado precondicionado.

Los métodos antes mencionados se colocaron en orden descendente en cuanto al consumo de recursos computacionales y ascendente en cuanto al aumento en la eficiencia en su desempeño, describiéndose a continuación:

Jacobi Si todos los elementos de la diagonal principal de la matriz $\underline{\underline{A}}$ son diferentes de cero $a_{ii} \neq 0$ para $i = 1, 2, ...n$. Podemos dividir la $i-$ésima ecuación del sistema lineal (170) por a_{ii} para $i = 1, 2, ...n$, y después trasladamos todas las incógnitas, excepto x_i, a la derecha, se obtiene el sistema equivalente

$$\underline{u} = \underline{\underline{B}}\underline{u} + \underline{d} \tag{173}$$

donde

$$d_i = \frac{b_i}{a_{ii}} \quad \text{y} \quad B = \{b_{ij}\} = \begin{cases} -\frac{a_{ij}}{a_{ii}} & \text{si } j \neq i \\ 0 & \text{si } j = i \end{cases} .$$

Las iteraciones del método de Jacobi están definidas por la fórmula

$$x_i = \sum_{j=1}^{n} b_{ij} x_j^{(k-1)} + d_i \tag{174}$$

donde $x_i^{(0)}$ son arbitrarias $(i = 1, 2, ...n; k = 1, 2, ...)$.

También el método de Jacobi se puede expresar en términos de matrices. Supongamos por un momento que la matriz $\underline{\underline{A}}$ tiene la diagonal unitaria, esto es $diag(\underline{\underline{A}}) = \underline{\underline{I}}$. Si descomponemos $\underline{\underline{A}} = \underline{\underline{I}} - \underline{\underline{B}}$, entonces el sistema dado por la Ecs. (170) se puede reescribir como

$$\left(\underline{\underline{I}} - \underline{\underline{B}}\right) \underline{u} = \underline{b}. \tag{175}$$

Para la primera iteración asumimos que $\underline{k}$=$\underline{b}$; entonces la última ecuación se escribe como $\underline{u} = \underline{\underline{B}}\underline{u}+\underline{k}$. Tomando una aproximación inicial $\underline{u}^0$, podemos obtener una mejor aproximación remplazando $\underline{u}$ por la más resiente aproximación de $\underline{u}^m$. Esta es la idea que subyace en el método Jacobi. El proceso iterativo queda como

$$\underline{u}^{m+1} = \underline{\underline{B}}\underline{u}^m + \underline{k}. \tag{176}$$

La aplicación del método a la ecuación de la forma $\underline{\underline{A}}\underline{u} = \underline{b}$, con la matriz $\underline{\underline{A}}$ no cero en los elementos diagonales, se obtiene multiplicando la Ec. (170) por $D^{-1} = \left[diag(\underline{\underline{A}})\right]^{-1}$ obteniendo

$$\underline{\underline{B}} = \underline{\underline{I}} - \underline{\underline{D}}^{-1}\underline{\underline{A}}, \quad \underline{k} = \underline{\underline{D}}^{-1}\underline{b}. \tag{177}$$

Gauss-Seidel Este método es una modificación del método Jacobi, en el cual una vez obtenido algún valor de $\underline{u}^{m+1}$, este es usado para obtener el resto de los valores utilizando los valores más actualizados de $\underline{u}^{m+1}$. Así, la Ec. (176) puede ser escrita como

$$u_i^{m+1} = \sum_{j<i} b_{ij} u_j^{m+1} + \sum_{j>i} b_{ij} u_j^m + k_i. \tag{178}$$

Notemos que el método Gauss-Seidel requiere el mismo número de operaciones aritméticas por iteración que el método de Jacobi. Este método se escribe en forma matricial como

$$\underline{u}^{m+1} = \underline{\underline{E}}\underline{u}^{m+1} + \underline{\underline{F}}\underline{u}^m + \underline{k} \tag{179}$$

donde $\underline{\underline{E}}$ y $\underline{\underline{F}}$ son las matrices triangular superior e inferior respectivamente. Este método mejora la convergencia con respecto al método de Jacobi en un factor aproximado de 2.

Richardson Escribiendo el método de Jacobi como

$$\underline{u}^{m+1} - \underline{u}^m = \underline{b} - \underline{\underline{A}}\underline{u}^m \tag{180}$$

entonces el método Richardson se genera al incorporar la estrategia de sobrerrelajación de la forma siguiente

$$\underline{u}^{m+1} = \underline{u}^m + \omega \left(\underline{b} - \underline{\underline{A}}\underline{u}^m \right). \tag{181}$$

El método de Richardson se define como

$$\underline{u}^{m+1} = \left(\underline{\underline{I}} - \omega \underline{\underline{A}} \right) \underline{u}^m + \omega \underline{b} \tag{182}$$

en la práctica encontrar el valor de ω puede resultar muy costoso computacionalmente y las diversas estrategias para encontrar ω dependen de las características propias del problema, pero este método con un valor ω óptimo resulta mejor que el método de Gauss-Seidel.

Relajación Sucesiva Partiendo del método de Gauss-Seidel y sobrerrelajando este esquema, obtenemos

$$u_i^{m+1} = (1-\omega)\, u_i^m + \omega \left[\sum_{j=1}^{i-1} b_{ij} u_j^{m+1} + \sum_{j=i+1}^{N} b_{ij} u_j^m + k_i \right] \tag{183}$$

y cuando la matriz $\underline{\underline{A}}$ es simétrica con entradas en la diagonal positivas, éste método converge si y sólo si $\underline{\underline{A}}$ es definida positiva y $\omega \in (0, 2)$. En la práctica encontrar el valor de ω puede resultar muy costoso computacionalmente y las diversas estrategias para encontrar ω dependen de las características propias del problema.

Gradiente Conjugado El método del gradiente conjugado ha recibido mucha atención en su uso al resolver ecuaciones diferenciales parciales y ha sido ampliamente utilizado en años recientes por la notoria eficiencia al reducir considerablemente en número de iteraciones necesarias para resolver el sistema algebraico de ecuaciones. Aunque los pioneros de este método fueron Hestenes y Stiefel (1952), el interés actual arranca a partir de que Reid (1971) lo planteara como un método iterativo, que es la forma en que se le usa con mayor frecuencia en la actualidad, esta versión está basada en el desarrollo hecho en [9].

La idea básica en que descansa el método del gradiente conjugado consiste en construir una base de vectores ortogonales y utilizarla para realizar la búsqueda de la solución en forma más eficiente. Tal forma de proceder generalmente no sería aconsejable porqué la construcción de una base ortogonal utilizando el procedimiento de Gramm-Schmidt requiere, al seleccionar cada nuevo elemento de la base, asegurar su ortogonalidad con respecto a cada uno de los vectores construidos previamente. La gran ventaja del método de gradiente conjugado radica en que cuando se utiliza este procedimiento, basta con asegurar la ortogonalidad de un nuevo miembro con respecto al último que se ha construido, para que automáticamente esta condición se cumpla con respecto a todos los anteriores.

En el algoritmo de gradiente conjugado (CGM), se toman como datos de entrada al sistema

$$\underline{\underline{A}}\underline{u} = \underline{b} \tag{184}$$

el vector de búsqueda inicial $\underline{u}^0$ y se calcula $\underline{r}^0 = \underline{b} - \underline{\underline{A}}\underline{u}^0$, $\underline{p}^0 = \underline{r}^0$, quedando el método esquemáticamente como:

$$\begin{aligned} \beta^{k+1} &= \frac{\underline{\underline{A}}\underline{p}^k \cdot \underline{r}^k}{\underline{\underline{A}}\underline{p}^k \cdot \underline{p}^k} \\ \underline{p}^{k+1} &= \underline{r}^k - \beta^{k+1}\underline{p}^k \\ \alpha^{k+1} &= \frac{\underline{r}^k \cdot \underline{r}^k}{\underline{\underline{A}}\underline{p}^{k+1} \cdot \underline{p}^{k+1}} \end{aligned} \tag{185}$$

$$\begin{aligned} \underline{u}^{k+1} &= \underline{u}^k + \alpha^{k+1}\underline{p}^{k+1} \\ \underline{r}^{k+1} &= \underline{r}^k - \alpha^{k+1}\underline{\underline{A}}\underline{p}^{k+1}. \end{aligned}$$

Si denotamos $\{\lambda_i, V_i\}_{i=1}^N$ como las eigensoluciones de $\underline{\underline{A}}$, i.e. $\underline{\underline{A}}V_i = \lambda_i V_i$, $i = 1, 2, ..., N$. Ya que la matriz $\underline{\underline{A}}$ es simétrica, los eigenvalores son reales y podemos ordenarlos por $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_N$. Definimos el número de condición por $Cond(\underline{\underline{A}}) = \lambda_N/\lambda_1$ y la norma de la energía asociada a $\underline{\underline{A}}$ por $\|\underline{u}\|^2_{\underline{\underline{A}}} = \underline{u} \cdot \underline{\underline{A}}\underline{u}$ entonces

$$\left\|\underline{u} - \underline{u}^k\right\|_{\underline{\underline{A}}} \leq \left\|\underline{u} - \underline{u}^0\right\|_{\underline{\underline{A}}} \left[\frac{1 - \sqrt{Cond(\underline{\underline{A}})}}{1 + \sqrt{Cond(\underline{\underline{A}})}}\right]^{2k}. \tag{186}$$

El siguiente teorema nos da idea del espectro de convergencia del sistema $\underline{\underline{A}}\underline{u} = \underline{b}$ para el método de gradiente conjugado.

Teorema 28 *Sea* $\kappa = cond(\underline{\underline{A}}) = \frac{\lambda_{máx}}{\lambda_{mín}} \geq 1$*, entonces el método de gradiente conjugado satisface la* $\underline{\underline{A}}-$*norma del error dado por*

$$\frac{\|e^n\|}{\|e^0\|} \leq \frac{2}{\left[\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^n + \left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^{-n}\right]} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^n \tag{187}$$

donde $\underline{e}^m = \underline{u} - \underline{u}^m$ *del sistema* $\underline{\underline{A}}\underline{u} = \underline{b}$.

Notemos que para κ grande se tiene que

$$\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \simeq 1 - \frac{2}{\sqrt{\kappa}} \tag{188}$$

tal que

$$\|\underline{e}^n\|_{\underline{\underline{A}}} \simeq \|\underline{e}^0\|_{\underline{\underline{A}}} \exp\left(-2\frac{n}{\sqrt{\kappa}}\right) \tag{189}$$

de lo anterior podemos esperar un espectro de convergencia del orden de $O(\sqrt{\kappa})$ iteraciones, para mayor referencia ver [25].

5.3. Precondicionadores

Una vía que permite mejorar la eficiencia de los métodos iterativos consiste en transformar al sistema de ecuaciones en otro equivalente, en el sentido de que posea la misma solución del sistema original pero que a su vez tenga mejores condiciones espectrales. Esta transformación se conoce como precondicionamiento y consiste en aplicar al sistema de ecuaciones una matriz conocida como precondicionador encargada de realizar el mejoramiento del número de condicionamiento.

Una amplia clase de precondicionadores han sido propuestos basados en las características algebraicas de la matriz del sistema de ecuaciones, mientras que por otro lado también existen precondicionadores desarrollados a partir de las características propias del problema que lo origina, un estudio más completo puede encontrarse en [2] y [17].

¿Qué es un Precondicionador? De una manera formal podemos decir que un precondicionador consiste en construir una matriz $\underline{\underline{C}}$, la cuál es una aproximación en algún sentido de la matriz $\underline{\underline{A}}$ del sistema $\underline{\underline{A}}\underline{u} = \underline{b}$, de manera tal que si multiplicamos ambos miembros del sistema de ecuaciones original por $\underline{\underline{C}}^{-1}$ obtenemos el siguiente sistema

$$\underline{\underline{C}}^{-1}\underline{\underline{A}}\underline{u} = \underline{\underline{C}}^{-1}\underline{b} \tag{190}$$

donde el número de condicionamiento de la matriz del sistema transformado $\underline{\underline{C}}^{-1}\underline{\underline{A}}$ debe ser menor que el del sistema original, es decir

$$Cond(\underline{\underline{C}}^{-1}\underline{\underline{A}}) < Cond(\underline{\underline{A}}), \tag{191}$$

dicho de otra forma un precondicionador es una inversa aproximada de la matriz original

$$\underline{\underline{C}}^{-1} \simeq \underline{\underline{A}}^{-1} \tag{192}$$

que en el caso ideal $\underline{\underline{C}}^{-1} = \underline{\underline{A}}^{-1}$ el sistema convergería en una sola iteración, pero el coste computacional del cálculo de $\underline{\underline{A}}^{-1}$ equivaldría a resolver el sistema por un método directo. Se sugiere que $\underline{\underline{C}}$ sea una matriz lo más próxima a $\underline{\underline{A}}$ sin que su determinación suponga un coste computacional elevado.

Dependiendo de la forma de platear el producto de $\underline{\underline{C}}^{-1}$ por la matriz del sistema obtendremos distintas formas de precondicionamiento, estas son:

$\underline{\underline{C}}^{-1}\underline{\underline{A}}\underline{x} = \underline{\underline{C}}^{-1}\underline{b}$	Precondicionamiento por la izquierda
$\underline{\underline{A}}\underline{\underline{C}}^{-1}\underline{\underline{C}}\underline{x} = \underline{b}$	Precondicionamiento por la derecha
$\underline{\underline{C}}_1^{-1}\underline{\underline{A}}\underline{\underline{C}}_2^{-1}\underline{\underline{C}}_2\underline{x} = \underline{\underline{C}}_1^{-1}\underline{b}$	Precondicionamiento por ambos lados si $\underline{\underline{C}}$ puede factorizarse como $\underline{\underline{C}} = \underline{\underline{C}}_1\underline{\underline{C}}_2$.

El uso de un precondicionador en un método iterativo provoca que se incurra en un costo de cómputo extra debido a que inicialmente se construye y luego se debe aplicar en cada iteración. Teniéndose que encontrar un balance entre el costo de construcción y aplicación del precondicionador versus la ganancia en velocidad en convergencia del método.

Ciertos precondicionadores necesitan poca o ninguna fase de construcción, mientras que otros pueden requerir de un trabajo substancial en esta etapa. Por otra parte la mayoría de los precondicionadores requieren en su aplicación un monto de trabajo proporcional al número de variables; esto implica que se multiplica el trabajo por iteración en un factor constante.

De manera resumida un buen precondicionador debe reunir las siguientes características:

i) Al aplicar un precondicionador $\underline{\underline{C}}$ al sistema original de ecuaciones $\underline{\underline{A}}\underline{u} = \underline{b}$, se debe reducir el número de iteraciones necesarias para que la solución aproximada tenga la convergencia a la solución exacta con una exactitud ε prefijada.

ii) La matriz $\underline{\underline{C}}$ debe ser fácil de calcular, es decir, el costo computacional de la construcción del precondicionador debe ser pequeño comparado con el costo total de resolver el sistema de ecuaciones $\underline{\underline{A}}\underline{u} = \underline{b}$.

iii) El sistema $\underline{\underline{C}}\underline{z} = \underline{r}$ debe ser fácil de resolver. Esto debe interpretarse de dos maneras:

a) El monto de operaciones por iteración debido a la aplicación del precondicionador $\underline{\underline{C}}$ debe ser pequeño o del mismo orden que las

que se requerirían sin precondicionamiento. Esto es importante si se trabaja en máquinas secuenciales.

b) El tiempo requerido por iteración debido a la aplicación del precondicionador debe ser pequeño.

En computadoras paralelas es importante que la aplicación del precondicionador sea paralelizable, lo cual eleva su eficiencia, pero debe de existir un balance entre la eficacia de un precondicionador en el sentido clásico y su eficiencia en paralelo ya que la mayoría de los precondicionadores tradicionales tienen un componente secuencial grande.

El método de gradiente conjugado por si mismo no permite el uso de precondicionadores, pero con una pequeña modificación en el producto interior usado en el método, da origen al método de gradiente conjugado precondicionado que a continuación detallaremos.

5.3.1. Gradiente Conjugado Precondicionado

Cuando la matriz $\underline{\underline{A}}$ es simétrica y definida positiva se puede escribir como

$$\lambda_1 \leq \frac{\underline{u}\underline{\underline{A}} \cdot \underline{u}}{\underline{u} \cdot \underline{u}} \leq \lambda_n \tag{193}$$

y tomando la matriz $\underline{\underline{C}}^{-1}$ como un precondicionador de $\underline{\underline{A}}$ con la condición de que

$$\lambda_1 \leq \frac{\underline{u}\underline{\underline{C}}^{-1}\underline{\underline{A}} \cdot \underline{u}}{\underline{u} \cdot \underline{u}} \leq \lambda_n \tag{194}$$

entonces la Ec. (184) se pude escribir como

$$\underline{\underline{C}}^{-1}\underline{\underline{A}}\underline{u} = \underline{\underline{C}}^{-1}b \tag{195}$$

donde $\underline{\underline{C}}^{-1}\underline{\underline{A}}$ es también simétrica y definida positiva en el producto interior $\langle \underline{u}, \underline{v} \rangle = \underline{u} \cdot \underline{\underline{C}}\underline{v}$, por que

$$\begin{aligned} \left\langle \underline{u}, \underline{\underline{C}}^{-1}\underline{\underline{A}}\underline{v} \right\rangle &= \underline{u} \cdot \underline{\underline{C}} \left(\underline{\underline{C}}^{-1}\underline{\underline{A}}\underline{v} \right) \\ &= \underline{u} \cdot \underline{\underline{A}}\underline{v} \end{aligned} \tag{196}$$

que por hipótesis es simétrica y definida positiva en ese producto interior.

La elección del producto interior $\langle \cdot, \cdot \rangle$ quedará definido como

$$\langle \underline{u}, \underline{v} \rangle = \underline{u} \cdot \underline{\underline{C}}^{-1}\underline{\underline{A}}\underline{v} \tag{197}$$

por ello las Ecs. (185[1]) y (185[3]), se convierten en

$$\alpha^{k+1} = \frac{\underline{r}^k \cdot \underline{r}^k}{\underline{p}^{k+1} \cdot \underline{\underline{C}}^{-1}\underline{p}^{k+1}} \tag{198}$$

y

$$\beta^{k+1} = \frac{\underline{p}^k \cdot \underline{\underline{C}}^{-1}\underline{r}^k}{\underline{p}^k \cdot \underline{\underline{A}}\underline{p}^k} \tag{199}$$

generando el método de gradiente conjugado precondicionado con precondicionador $\underline{\underline{C}}^{-1}$. Es necesario hacer notar que los métodos gradiente conjugado y gradiente conjugado precondicionado sólo difieren en la elección del producto interior.

Para el método de gradiente conjugado precondicionado, los datos de entrada son un vector de búsqueda inicial $\underline{u}^0$ y el precondicionador $\underline{\underline{C}}^{-1}$. Calculándose $\underline{r}^0 = \underline{b} - \underline{\underline{A}}\underline{u}^0$, $\underline{p} = \underline{\underline{C}}^{-1}\underline{r}^0$, quedando el método esquemáticamente como:

$$\begin{aligned}
\beta^{k+1} &= \frac{\underline{p}^k \cdot \underline{\underline{C}}^{-1}\underline{r}^k}{\underline{p}^k \cdot \underline{\underline{A}}\underline{p}^k} \\
\underline{p}^{k+1} &= \underline{r}^k - \beta^{k+1}\underline{p}^k \\
\alpha^{k+1} &= \frac{\underline{r}^k \cdot \underline{r}^k}{\underline{p}^{k+1} \cdot \underline{\underline{C}}^{-1}\underline{p}^{k+1}} \\
\underline{u}^{k+1} &= \underline{u}^k + \alpha^{k+1}\underline{p}^{k+1} \\
\underline{r}^{k+1} &= \underline{\underline{C}}^{-1}\underline{r}^k - \alpha^{k+1}\underline{\underline{A}}\underline{p}^{k+1}.
\end{aligned} \tag{200}$$

Algoritmo Computacional del Método Dado el sistema $\underline{\underline{A}}\underline{u} = \underline{b}$, con la matriz $\underline{\underline{A}}$ simétrica y definida positiva de dimensión $n \times n$. La entrada al método será una elección de $\underline{u}^0$ como condición inicial, $\varepsilon > 0$ como la tolerancia del método, N como el número máximo de iteraciones y la matriz de precondicionamiento $\underline{\underline{C}}^{-1}$ de dimensión $n \times n$, el algoritmo del método de gradiente conjugado precondicionado queda como:

$\underline{r} = \underline{b} - \underline{\underline{A}}\underline{u}^0$
$\underline{w} = \underline{\underline{C}}^{-1}\underline{r}$
$\underline{v} = (\underline{\underline{C}}^{-1})^T\underline{w}$
$\alpha = \sum_{j=1}^{N} w_j^2$
$k = 1$
Mientras que $k \leq N$

Si $\|\underline{v}\|_\infty < \varepsilon$ Salir
$\underline{u} = \underline{\underline{A}}\underline{v}$
$t = \frac{\alpha}{\sum_{j=1}^{N} v_j u_j}$
$\underline{x} = \underline{x} + t\underline{v}$
$\underline{r} = \underline{r} - t\underline{u}$
$\underline{w} = \underline{\underline{C}}^{-1}\underline{r}$
$\beta = \sum_{j=1}^{N} w_j^2$
Si $\|\underline{r}\|_\infty < \varepsilon$ Salir

$s = \frac{\beta}{\alpha}$

$\underline{v} = \left(\underline{\underline{C}}^{-1}\right)^T \underline{w} + s\underline{v}$

$\alpha = \beta$

$k = k + 1$

La salida del método será la solución aproximada $\underline{x} = (x_1, ..., x_N)$ y el residual $\underline{r} = (r_1, ..., r_N)$.

En el caso del método sin precondicionamiento, $\underline{\underline{C}}^{-1}$ es la matriz identidad, que para propósitos de optimización sólo es necesario hacer la asignación de vectores correspondiente en lugar del producto de la matriz por el vector. En el caso de que la matriz $\underline{\underline{A}}$ no sea simétrica, el método de gradiente conjugado puede extenderse para soportarlas, para más información sobre pruebas de convergencia, resultados numéricos entre los distintos métodos de solución del sistema algebraico $\underline{\underline{A}}\underline{u} = \underline{b}$ generada por la discretización de un problema elíptico y como extender estos para matrices no simétricas ver [9] y [7].

Clasificación de los Precondicionadores En general se pueden clasificar en dos grandes grupos según su manera de construcción: los algebraicos o a posteriori y los a priori o directamente relacionados con el problema continuo que lo origina.

5.3.2. Precondicionador a Posteriori

Los precondicionadores algebraicos o a posteriori son los más generales, ya que sólo dependen de la estructura algebraica de la matriz $\underline{\underline{A}}$, esto quiere decir que no tienen en cuenta los detalles del proceso usado para construir el sistema de ecuaciones lineales $\underline{\underline{A}}\underline{u} = \underline{b}$. Entre estos podemos citar los métodos de precondicionamiento del tipo Jacobi, SSOR, factorización incompleta, inversa aproximada, diagonal óptimo y polinomial.

Precondicionador Jacobi El método precondicionador Jacobi es el precondicionador más simple que existe y consiste en tomar en calidad de precondicionador a los elementos de la diagonal de $\underline{\underline{A}}$

$$C_{ij} = \begin{cases} A_{ij} \quad si \quad i = j \\ \\ 0 \quad si \quad i \neq j. \end{cases} \tag{201}$$

Debido a que las operaciones de división son usualmente más costosas en tiempo de cómputo, en la práctica se almacenan los recíprocos de la diagonal de $\underline{\underline{A}}$.

Ventajas: No necesita trabajo para su construcción y puede mejorar la convergencia.

Desventajas: En problemas con número de condicionamiento muy grande, no es notoria la mejoría en el número de iteraciones.

Precondicionador SSOR Si la matriz original es simétrica, se puede descomponer como en el método de sobrerrelajamiento sucesivo simétrico (SSOR) de la siguiente manera

$$\underline{\underline{A}} = \underline{\underline{D}} + \underline{\underline{L}} + \underline{\underline{L}}^T \tag{202}$$

donde $\underline{\underline{D}}$ es la matriz de la diagonal principal y $\underline{\underline{L}}$ es la matriz triangular inferior.

La matriz en el método SSOR se define como

$$\underline{\underline{C}}(\omega) = \frac{1}{2-w}\left(\frac{1}{\omega}\underline{\underline{D}} + \underline{\underline{L}}\right)\left(\frac{1}{\omega}\underline{\underline{D}}\right)^{-1}\left(\frac{1}{\omega}\underline{\underline{D}} + \underline{\underline{L}}\right)^T \tag{203}$$

en la práctica la información espectral necesaria para hallar el valor óptimo de ω es demasiado costoso para ser calculado.

> Ventajas: No necesita trabajo para su construcción, puede mejorar la convergencia significativamente.
>
> Desventajas: Su paralelización depende fuertemente del ordenamiento de las variables.

Precondicionador de Factorización Incompleta Existen una amplia clase de precondicionadores basados en factorizaciones incompletas. La idea consiste en que durante el proceso de factorización se ignoran ciertos elementos diferentes de cero correspondientes a posiciones de la matriz original que son nulos. La matriz precondicionadora se expresa como $\underline{\underline{C}} = \underline{\underline{LU}}$, donde $\underline{\underline{L}}$ es la matriz triangular inferior y $\underline{\underline{U}}$ la superior. La eficacia del método depende de cuán buena sea la aproximación de $\underline{\underline{C}}^{-1}$ con respecto a $\underline{\underline{A}}^{-1}$.

El tipo más común de factorización incompleta se basa en seleccionar un subconjunto S de las posiciones de los elementos de la matriz y durante el proceso de factorización considerar a cualquier posición fuera de éste igual a cero. Usualmente se toma como S al conjunto de todas las posiciones (i, j) para las que $A_{ij} \neq 0$. Este tipo de factorización es conocido como factorización incompleta LU de nivel cero, ILU(0).

El proceso de factorización incompleta puede ser descrito formalmente como sigue:

Para cada k, si $i, j > k$:

$$S_{ij} = \begin{cases} A_{ij} - A_{ij}A_{ij}^{-1}A_{kj} & \text{Si} \quad (i,j) \in S \\ \\ A_{ij} & \text{Si} \quad (i,j) \notin S. \end{cases} \tag{204}$$

Una variante de la idea básica de las factorizaciones incompletas lo constituye la factorización incompleta modificada que consiste en que si el producto

$$A_{ij} - A_{ij}A_{ij}^{-1}A_{kj} \neq 0 \tag{205}$$

y el llenado no está permitido en la posición (i, j), en lugar de simplemente descartarlo, esta cantidad se le substrae al elemento de la diagonal A_{ij}. Matemáticamente esto corresponde a forzar a la matriz precondicionadora a tener la misma suma por filas que la matriz original. Esta variante resulta de interés puesto

que se ha probado que para ciertos casos la aplicación de la factorización incompleta modificada combinada con pequeñas perturbaciones hace que el número de condicionamiento espectral del sistema precondicionado sea de un orden inferior.

Ventaja: Puede mejorar el condicionamiento y la convergencia significativamente.

Desventaja: El proceso de factorización es costoso y difícil de paralelizar en general.

Precondicionador de Inversa Aproximada El uso del precondicionador de inversas aproximada se ha convertido en una buena alternativa para los precondicionadores implícitos debido a su naturaleza paralelizable. Aquí se construye una matriz inversa aproximada usando el producto escalar de Frobenius.

Sea $\mathcal{S} \subset C_n$, el subespacio de las matrices $\underline{\underline{C}}$ donde se busca una inversa aproximada explícita con un patrón de dispersión desconocido. La formulación del problema esta dada como: Encontrar $\underline{\underline{C}}_0 \in \mathcal{S}$ tal que

$$\underline{\underline{C}}_0 = \arg mín_{\underline{\underline{C}} \in \mathcal{S}} \left\| \underline{\underline{AC}} - \underline{\underline{I}} \right\|. \tag{206}$$

Además, esta matriz inicial $\underline{\underline{C}}_0$ puede ser una inversa aproximada de $\underline{\underline{A}}$ en un sentido estricto, es decir,

$$\left\| \underline{\underline{AC}}_0 - \underline{\underline{I}} \right\| = \varepsilon < 1. \tag{207}$$

Existen dos razones para esto, primero, la ecuación (207) permite asegurar que $\underline{\underline{C}}_0$ no es singular (lema de Banach), y segundo, esta será la base para construir un algoritmo explícito para mejorar $\underline{\underline{C}}_0$ y resolver la ecuación $\underline{\underline{A}}\underline{u} = \underline{b}$.

La construcción de $\underline{\underline{C}}_0$ se realiza en paralelo, independizando el cálculo de cada columna. El algoritmo permite comenzar desde cualquier entrada de la columna k, se acepta comúnmente el uso de la diagonal como primera aproximación. Sea r_k el residuo correspondiente a la columna k-ésima, es decir

$$r_k = \underline{\underline{AC}}_k - \underline{e}_k \tag{208}$$

y sea $\mathcal{I}_k$ el conjunto de índices de las entradas no nulas en r_k, es decir, $\mathcal{I}_k = \{i = \{1, 2, ..., n\} \mid r_{ik} \neq 0\}$. Si $\mathcal{L}_k = \{l = \{1, 2, ..., n\} \mid C_{lk} \neq 0\}$, entonces la nueva entrada se busca en el conjunto $\mathcal{J}_k = \{j \in \mathcal{L}_k^c \mid A_{ij} \neq 0, \forall i \in \mathcal{I}_k\}$. En realidad las únicas entradas consideradas en $\underline{\underline{C}}_k$ son aquellas que afectan las entradas no nulas de r_k. En lo que sigue, asumimos que $\mathcal{L}_k \cup \{j\} = \left\{i_1^k, i_2^k, ..., i_{p_k}^k\right\}$ es no vacío, siendo p_k el número actual de entradas no nulas de $\underline{\underline{C}}_k$ y que $i_{p_k}^k = j$, para todo $j \in \mathcal{J}_k$. Para cada j, calculamos

$$\left\| \underline{\underline{AC}}_k - \underline{e}_k \right\|_2^2 = 1 - \sum_{l=1}^{p_k} \frac{\left[\det\left(\underline{\underline{D}}_l^k\right)\right]^2}{\det\left(\underline{\underline{G}}_{l-2}^k\right) \det\left(\underline{\underline{G}}_l^k\right)} \tag{209}$$

donde, para todo k, $\det\left(\underline{\underline{G}}_0^k\right) = 1$ y $\underline{\underline{G}}_l^k$ es la matriz de Gram de las columnas $i_1^k, i_2^k, ..., i_{p_k}^k$ de la matriz $\underline{\underline{A}}$ con respecto al producto escalar Euclideano; $\underline{\underline{D}}_l^k$ es la matriz que resulta de remplazar la última fila de la matriz $\underline{\underline{G}}_l^k$ por $a_{ki_1^k}, a_{ki_2^k}, , ..., a_{ki_l^k}$, con $1 \leq l \leq p_k$. Se selecciona el índice j_k que minimiza el valor de $\left\|\underline{\underline{A}}\underline{C}_k - \underline{e}_k\right\|_2$.

Esta estrategia define el nuevo índice seleccionado j_k atendiendo solamente al conjunto $\mathcal{L}_k$, lo que nos lleva a un nuevo óptimo donde se actualizan todas las entradas correspondientes a los índices de $\mathcal{L}_k$. Esto mejora el criterio de (206) donde el nuevo índice se selecciona manteniendo las entradas correspondientes a los índices de $\mathcal{L}_k$. Así $\underline{C}_k$ se busca en el conjunto

$$\mathcal{S}_k = \left\{\underline{C}_k \in \mathbb{R}^n \mid C_{ik} = 0, \forall i \in \mathcal{L}_k \cup \{j_k\}\right\},$$

$$\underline{m}_k = \sum_{l=1}^{p_k} \frac{\det\left(\underline{\underline{D}}_l^k\right)}{\det\left(\underline{\underline{G}}_{l-2}^k\right)\det\left(\underline{\underline{G}}_l^k\right)}\underline{\tilde{m}}_l \tag{210}$$

donde $\underline{\tilde{C}}_l$ es el vector con entradas no nulas i_h^k $(1 \leq h \leq l)$. Cada una de ellas se obtiene evaluado el determinante correspondiente que resulta de remplazar la última fila del $\det\left(\underline{\underline{G}}_l^k\right)$ por e_h^t, con $1 \leq l \leq p_k$.

Evidentemente, los cálculos de $\left\|\underline{\underline{A}}\underline{C}_k - \underline{e}_k\right\|_2^2$ y de $\underline{C}_k$ pueden actualizarse añadiendo la contribución de la última entrada $j \in \mathcal{J}_k$ a la suma previa de 1 a $p_k - 1$. En la práctica, $\det\left(\underline{\underline{G}}_l^k\right)$ se calcula usando la descomposición de Cholesky puesto que $\underline{\underline{G}}_l^k$ es una matriz simétrica y definida positiva. Esto sólo involucra la factorización de la última fila y columna si aprovechamos la descomposición de $\underline{\underline{G}}_{l-1}^k$. Por otra parte, $\det\left(\underline{\underline{D}}_l^k\right) / \det\left(\underline{\underline{G}}_l^k\right)$ es el valor de la última incógnita del sistema $\underline{\underline{G}}_l^k \underline{d}_l = \left(a_{ki_1^k}, a_{ki_2^k}, , ..., a_{ki_l^k}\right)^T$ necesitándose solamente una sustitución por descenso. Finalmente, para obtener $\underline{\tilde{C}}_l$ debe resolverse el sistema $\underline{\underline{G}}_l^k \underline{v}_l = \underline{e}_l$, con $\underline{\tilde{C}}_{i_1^k l} = v_{hl}, (1 \leq h \leq l)$.

Ventaja: Puede mejorar el condicionamiento y la convergencia significativamente y es fácilmente paralelizable.

Desventaja: El proceso construcción es algo laborioso.

5.3.3. Precondicionador a Priori

Los precondicionadores a priori son más particulares y dependen para su construcción del conocimiento del proceso de discretización de la ecuación diferencial parcial, dicho de otro modo dependen más del proceso de construcción de la matriz $\underline{\underline{A}}$ que de la estructura de la misma.

Estos precondicionadores usualmente requieren de más trabajo que los del tipo algebraico discutidos anteriormente, sin embargo permiten el desarrollo de métodos de solución especializados más rápidos que los primeros.

Veremos algunos de los métodos más usados relacionados con la solución de ecuaciones diferenciales parciales en general y luego nos concentraremos en el caso de los métodos relacionados directamente con descomposición de dominio.

En estos casos el precondicionador $\underline{\underline{C}}$ no necesariamente toma la forma simple de una matriz, sino que debe ser visto como un operador en general. De aquí que $\underline{\underline{C}}$ podría representar al operador correspondiente a una versión simplificada del problema con valores en la frontera que deseamos resolver.

Por ejemplo se podría emplear en calidad de precondicionador al operador original del problema con coeficientes variables tomado con coeficientes constantes. En el caso del operador de Laplace se podría tomar como precondicionador a su discretización en diferencias finitas centrales.

Por lo general estos métodos alcanzan una mayor eficiencia y una convergencia óptima, es decir, para ese problema en particular el precondicionador encontrado será el mejor precondicionador existente, llegando a disminuir el número de iteraciones hasta en un orden de magnitud. Donde muchos de ellos pueden ser paralelizados de forma efectiva.

El Uso de la Parte Simétrica como Precondicionador La aplicación del método del gradiente conjugado en sistemas no auto-adjuntos requiere del almacenamiento de los vectores previamente calculados. Si se usa como precondicionador la parte simétrica

$$(\underline{\underline{A}} + \underline{\underline{A}}^T)/2 \tag{211}$$

de la matriz de coeficientes $\underline{\underline{A}}$, entonces no se requiere de éste almacenamiento extra en algunos casos, resolver el sistema de la parte simétrica de la matriz $\underline{\underline{A}}$ puede resultar más complicado que resolver el sistema completo.

El Uso de Métodos Directos Rápidos como Precondicionadores En muchas aplicaciones la matriz de coeficientes $\underline{\underline{A}}$ es simétrica y positivo definida, debido a que proviene de un operador diferencial auto-adjunto y acotado. Esto implica que se cumple la siguiente relación para cualquier matriz $\underline{\underline{B}}$ obtenida de una ecuación diferencial similar

$$c_1 \leq \frac{\underline{x}^T \underline{\underline{A}} \underline{x}}{\underline{x}^T \underline{\underline{B}} \underline{x}} \leq c_2 \quad \forall \underline{x} \tag{212}$$

donde c_1 y c_2 no dependen del tamaño de la matriz. La importancia de esta propiedad es que del uso de $\underline{\underline{B}}$ como precondicionador resulta un método iterativo cuyo número de iteraciones no depende del tamaño de la matriz.

La elección más común para construir el precondicionador $\underline{\underline{B}}$ es a partir de la ecuación diferencial parcial separable. El sistema resultante con la matriz $\underline{\underline{B}}$ puede ser resuelto usando uno de los métodos directos de solución rápida, como pueden ser por ejemplo los basados en la transformada rápida de Fourier.

Como una ilustración simple del presente caso obtenemos que cualquier operador elíptico puede ser precondicionado con el operador de Poisson.

Construcción de Precondicionadores para Problemas Elípticos Empleando DDM Existen una amplia gama de este tipo de precondicionadores, pero son específicos al método de descomposición de dominio usado, para el método de subestructuración, los más importantes se derivan de la matriz de rigidez y por el método de proyecciones, el primero se detalla en la sección (6.2.1) y el segundo, conjuntamente con otros precondicionadores pueden ser consultados en [11], [5], [4] y [2].

La gran ventaja de este tipo de precondicionadores es que pueden ser óptimos, es decir, para ese problema en particular el precondicionador encontrado será el mejor precondicionador existente, llegando a disminuir el número de iteraciones hasta en un orden de magnitud.

6. Métodos de Descomposición de Dominio (DDM)

La solución numérica por los esquemas tradicionales de discretización tipo elemento finito y diferencias finitas generan una discretización del problema, la cual es usada para generar un sistema de ecuaciones algebraicos $\underline{\underline{A}}\underline{u} = \underline{b}$. Este sistema algebraico en general es de gran tamaño para problemas reales, al ser estos algoritmos secuenciales su implantación suele hacerse en equipos secuenciales y por ello no es posible resolver muchos problemas que involucren el uso de una gran cantidad de memoria, actualmente para tratar de subsanar dicha limitante, se usa equipo paralelo para soportar algoritmos secuenciales, haciendo ineficiente su implantación en dichos equipos.

Los métodos de descomposición de dominio son un paradigma natural usado por la comunidad de modeladores. Los sistemas físicos son descompuestos en dos o más subdominios contiguos basados en consideraciones fenomenológicas. Estas descomposiciones basadas en dominios físicos son reflejadas en la ingeniería de software del código correspondiente.

Los métodos de descomposición de dominio permiten tratar los problemas de tamaño considerable, empleando algoritmos paralelos en computadoras secuenciales y/o paralelas. Esto es posible ya que cualquier método de descomposición de dominio se basa en la suposición de que dado un dominio computacional Ω, este se puede particionar en subdominios $\Omega_i, i = 1, 2, ..., E$ entre los cuales puede o no existir traslape. Entonces el problema es reformulado en términos de cada subdominio (empleando algún método del tipo elemento finito) obteniendo una familia de subproblemas de tamaño reducido independientes en principio entre si, que están acoplados a través de la solución en la interfaz de los subdominios que es desconocida.

De esta manera, podemos clasificar de manera burda a los métodos de descomposición de dominio, como aquellos en que: existe traslape entre los subdominios y en los que no existe traslape. A la primera clase pertenece el método de Schwarz (en el cual el tamaño del traslape es importante en la convergencia del método) y a los de la segunda clase pertenecen los métodos del tipo subestructuración (en el cual los subdominios sólo tienen en común los nodos de la frontera interior).

La computación en paralelo es una técnica que nos permite distribuir una gran carga computacional entre muchos procesadores. Y es bien sabido que una de las mayores dificultades del procesamiento en paralelo es la coordinación de las actividades de los diferentes procesadores y el intercambio de información entre los mismos [21] mediante el paso de mensajes.

Así, mediante los métodos de descomposición de dominio, la programación orientada a objetos y esquemas de paralelización que usan el paso de mensajes, es posible construir aplicaciones que coadyuven a la solución de problemas concomitantes en ciencia e ingeniería, ya que permiten utilizar todas las capacidades del cómputo en paralelo (supercomputadoras, clusters o grids), de esta forma es posible atacar una gran variedad de problemas que sin estas técnicas es imposible hacerlo de manera flexible y eficiente.

Pero hay que notar que existen una amplia gama de problemas que nos

interesan resolver que superan las capacidades de cómputo actuales, ya sea por el tiempo requerido para su solución, por el consumo excesivo de memoria o ambos.

La lista de los métodos de descomposición de dominio y el tipo de problemas que pueden ser atacados por estos, es grande y está en constante evolución, ya que se trata de encontrar un equilibrio entre la complejidad del método (aunada a la propia complejidad del modelo), la eficiencia en el consumo de los recursos computacionales y la precisión esperada en la solución encontrada por los diversos métodos y las arquitecturas paralelas en la que se implante.

A continuación describiremos algunos de estos métodos generales. En este capítulo se considerarán problemas con valor en la frontera (VBVP) de la forma

$$\begin{aligned} \mathcal{L}u &= f \quad \text{en } \Omega \\ u &= g \quad \text{en } \partial\Omega \end{aligned} \tag{213}$$

donde

$$\mathcal{L}u = -\nabla \cdot \underline{\underline{a}} \cdot \nabla u + cu \tag{214}$$

como un caso particular del operador elíptico definido por la Ec. (71) de orden dos.

6.1. Método de Schwarz

El método fue desarrollado por Hermann Amandus Schwarz en 1869 (no como un método de descomposición de dominio), ya que en esos tiempos los matemáticos podían resolver problemas con geometrías sencillas de manera analítica, pero no tenían una idea clara de como poder resolver problemas que involucraran el traslape de esas geometrías sencillas. Como se conocía la solución para las geometrías sencillas por separado, la idea de Schwarz fue usar estas para conocer la solución en la geometría resultante al tener traslape, para más detalle ver [1].

Para describir el método, consideremos primero un dominio Ω que está formado de dos subdominios Ω_1 y Ω_2 traslapados, es decir $\Omega_1 \cap \Omega_2 \neq \varnothing$, entonces $\Omega = \Omega_1 \cup \Omega_2$ y denotemos a $\Sigma_1 = \partial\Omega_1 \cap \Omega_2$, $\Sigma_2 = \partial\Omega_2 \cap \Omega_1$ y $\Omega_{1,2} = \Omega_1 \cap \Omega_2$, como se muestra en la figura para dos dominios distintos:

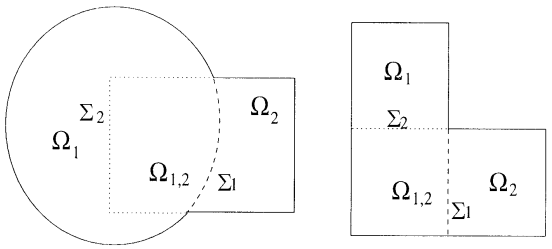

Figura 2: Dominio Ω subdividido en dos subdominios Ω_1 y Ω_2.

La forma original del método iterativo de Schwarz conocido como métodos alternantes de Schwarz, consiste en resolver sucesivamente los siguientes problemas.

Sea u^o una función de inicialización definida en Ω, que se nulifica en $\partial\Omega$, además hacemos $u_1^0 = u^0_{|\Omega_1}$ y $u_2^0 = u^0_{|\Omega_2}$. Para $k \geq 0$ definimos dos sucesiones u_1^{k+1} y u_2^{k+1} para resolver respectivamente

$$\begin{cases} \mathcal{L}u_1^{k+1} = f \quad \text{en } \Omega_1 \\ u_1^{k+1} = u_2^k \quad \text{en } \Sigma_1 \\ u_1^{k+1} = 0 \quad \text{en } \partial\Omega_1 \cap \partial\Omega \end{cases} \tag{215}$$

y

$$\begin{cases} \mathcal{L}u_2^{k+1} = f \quad \text{en } \Omega_2 \\ u_2^{k+1} = u_1^{k+1} \quad \text{en } \Sigma_2 \\ u_2^{k+1} = 0 \quad \text{en } \partial\Omega_2 \cap \partial\Omega \end{cases} \tag{216}$$

resolviendo los problemas secuencialmente en cada subdominio (por ejemplo con el método de elemento finito). Este método se conoce como Schwarz multiplicativo.

El método alternante de Schwarz dado por las Ecs. (215) y (216) converge a la solución u de (213) si suponemos alguna suavidad en los subdominios Ω_1 y Ω_2, ya que existen constantes C_1 y $C_2 \in (0,1)$ tal que para todo $k \geq 0$ se tiene

$$\begin{aligned} \left\|u_{|\Omega_1} - u_1^{k+1}\right\|_{L^\infty(\Omega_1)} &\leq C_1^k C_2^k \left\|u - u^0\right\|_{L^\infty(\Sigma_1)} \\ \left\|u_{|\Omega_2} - u_2^{k+1}\right\|_{L^\infty(\Omega_2)} &\leq C_1^{k+1} C_2^k \left\|u - u^0\right\|_{L^\infty(\Sigma_2)} \end{aligned} \tag{217}$$

las constantes C_1 y C_2 de reducción de error deben de estar bastante cerca de 1 si la región de traslape $\Omega_{1,2}$ es delgada, la prueba de esta estimación puede encontrarse en [12].

Por otro lado, teniendo el conjunto $u_1^0 = u^0_{|\Omega_1}$ y $u_2^0 = u^0_{|\Omega_2}$, podemos generar dos pasos independientes uno de otro

$$\begin{cases} \mathcal{L}u_1^{k+1} = f \quad \text{en } \Omega_1 \\ u_1^{k+1} = u_2^k \quad \text{en } \Sigma_1 \\ u_1^{k+1} = 0 \quad \text{en } \partial\Omega_1 \cap \partial\Omega \end{cases} \tag{218}$$

y

$$\begin{cases} \mathcal{L}u_2^{k+1} = f \quad \text{en } \Omega_2 \\ u_2^{k+1} = u_1^k \quad \text{en } \Sigma_2 \\ u_2^{k+1} = 0 \quad \text{en } \partial\Omega_2 \cap \partial\Omega \end{cases} \tag{219}$$

resolviendo los problemas en paralelo de cada subdominio (por ejemplo con el método de elemento finito). Este método se conoce como Schwarz aditivo.

La convergencia de este método en general requiere de algunas hipótesis adicionales, pero si converge, el número de iteraciones necesarias para converger será del doble que el método Schwarz multiplicativo.

La generalización del método de Schwarz en el caso en que Ω es particionada en $E > 2$ subdominios traslapados puede describirse como sigue:

Descomponiendo el dominio Ω en E subdominios Ω_e con traslape como por ejemplo, la descomposición siguiente

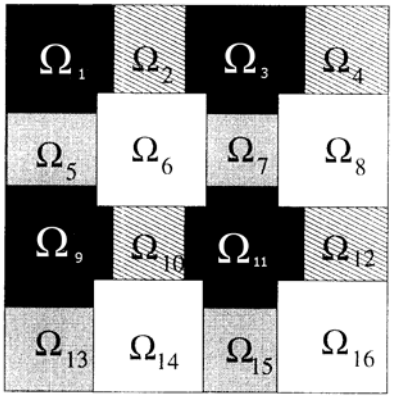

Figura 3: Descomposición de Ω en múltiples subdominios con traslape para el método de Schwarz.

Entonces para resolver el problema por el método de Schwarz, primeramente es necesario definir los siguientes subespacios del espacio de Sobolev $H^1(\Omega)$, en ellos estarán definidas las funciones usadas en el desarrollo del método:

$$\begin{aligned} V_i &= \left\{ v_i \in H^1(\Omega_i) \mid v_{|\partial\Omega \cap \partial\Omega_i} = 0 \right\} \\ V_i^0 &= H_0^1(\Omega_i) \\ V_i^* &= \left\{ v \in H_0^1(\Omega) \mid v = 0 \text{ en } \Omega \backslash \bar{\Omega}_i \right\}. \end{aligned}$$

Denotaremos por I al operador identidad, por $J_i, i = 1, ...E$ la inmersión de V_i^* sobre V (i.e. $J_i v = v$ para toda $v \in V_i^*$), y por $J_i^T : H_0^1(\Omega_i) \to V_i^*$ al transpuesto del operador J_i definido por

$$\left\langle J_i^T F, v \right\rangle = \left\langle F, J_i v \right\rangle, \quad \forall F \in V^{'}, v \in V_i^*. \tag{220}$$

y definimos

$$P_i = J_i P_i^* : H_0^1(\Omega_i) \to H_0^1(\Omega_i).$$

Sea $\mathcal{L}_i : V_i^0 \to \left(V_i^0\right)^{'}$ la restricción del operador $\mathcal{L}$ al subespacio V_i^0, definido como

$$\left\langle \mathcal{L}_i w_i, v_i \right\rangle = a\left(w_i, v_i\right) \text{ para toda } w_i, v_i \in V_i^0$$

y como un operador de extensión $\rho_i^T : V_i^0 \to V_i^*$ definido como

$$\rho_i^T v_i = \tilde{v}_i \text{ para toda } v_i \in V_i^0 \tag{221}$$

y el transpuesto del operador restricción $\rho_i : (V_i^*)' \to (V_i^0)'$ como

$$\langle \rho_i G, v_i \rangle = \left\langle G, \rho_i^T v_i \right\rangle, \text{para toda } G \in (V_i^*)', v_i \in V_i^0. \tag{222}$$

De lo anterior se tiene que

$$\mathcal{L}_i = \rho_i J_i^T \mathcal{L} J_i \rho_i^T$$

y

$$P_i = J_i P_i^* : V \to V \text{ para } i = 1, ..., E. \tag{223}$$

Entonces el método Multiplicativo de Schwarz queda como

$$u^{k+\frac{i}{E}} = (I - P_i) u^{k+\frac{i-1}{E}} + J_i \rho_i^T \mathcal{L}_i^{-1} \rho_i J_i^T f \tag{224}$$

para $i = 1, 2, ..., E$ y

$$u^{k+1} = \left(I - \sum_{i=1}^{E} P_i \right) u^k + \sum_{i=1}^{M} J_i \rho_i^T \mathcal{L}_i^{-1} \rho_i J_i^T f \tag{225}$$

y la correspondiente ecuación de error como

$$u - u^{k+1} = (I - P_m) ... (I - P_1) (u - u^k). \tag{226}$$

El el método Aditivo de Schwarz queda como

$$u - u^{k+1} = \left(I - \sum_{i=1}^{E} P_i \right) \left(u - u^k \right) \tag{227}$$

y la correspondiente ecuación de error como

$$u - u^{k+1} = (I - P_m) ... (I - P_1) (u - u^k). \tag{228}$$

Observaciones:

- La precisión del método depende fuertemente del número de iteraciones realizadas en el proceso iterativo y converge a la precisión usada en la solución de cada subdominio en el mejor de los casos.
- El método aditivo de Schwarz es secuencial, en el caso del método multiplicativo de Schwarz es paralelizable pero tiene una parte serial importante en el algoritmo y su convergencia no es la óptima en esta formulación, pero existen variantes del método que permiten remediar esto, para más detalles ver [2], [4] y [5].
- Hay que notar que por cada subdominio (supóngase n) y en cada iteración (supóngase I) se resuelve un problema para cada Ω_i, esto significa que si se usa el método de elemento finito para resolver el problema local donde se usen en promedio r iteraciones para resolver el sistema lineal (no tomando en cuenta el costo invertido en generar las matrices), el total de iteraciones necesarias para resolver el problema en el dominio Ω será $r * n * I$, resultando muy costoso computacionalmente con respecto a otros métodos de descomposición de dominio.

6.2. Método de Subestructuración

Consideremos el problema dado por la Ec. (213) en el dominio Ω, el cual es subdividido en E subdominios Ω_i, $i = 1, 2, ..., E$ sin traslape, es decir

$$\Omega_i \cap \Omega_j = \varnothing \quad \forall i \neq j \quad \text{y} \quad \overline{\Omega} = \bigcup_{i=1}^{E} \overline{\Omega}_i, \tag{229}$$

y al conjunto

$$\Sigma = \bigcup_{i=1}^{E} \Sigma_i, \quad \text{si } \Sigma_i = \partial\Omega_i \backslash \partial\Omega \tag{230}$$

lo llamaremos la frontera interior de los subdominios, un ejemplo se muestra en la figura:

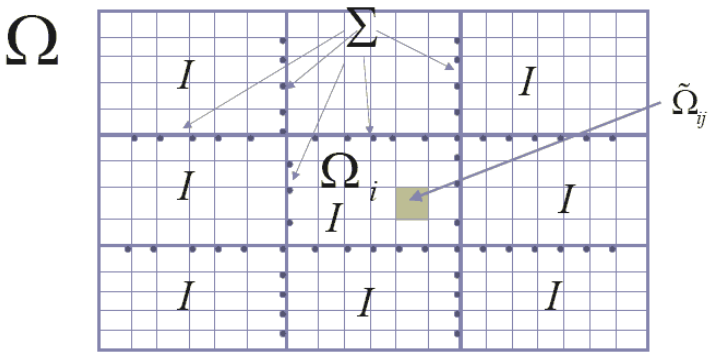

Figura 4: Dominio Ω descompuesto en subdominios Ω_i, con $i = 1, 2, ..., 9$.

Sin perdida de generalidad tomemos $g = 0$ en $\partial\Omega$, notemos que siempre es posible poner el problema de la Ec. (213) como uno con condiciones de frontera Dirichlet que se nulifiquen mediante la adecuada manipulación del término del lado derecho de la ecuación.

Primeramente sea $D \subset H_0^1(\Omega)$ un espacio lineal de funciones de dimensión finita N, en el cual esté definido un producto interior denotado para cada $u, v \in D$ por

$$u \cdot v = \langle u, v \rangle \tag{231}$$

además sean $\tilde{D}_I, \tilde{D}_\Sigma$ y $\bar{D}_\Sigma$ subespacios lineales de D con la propiedad de que $\tilde{D}_\Sigma$ y $\bar{D}_\Sigma$ cada uno por separado sean los complementos algebraicos con respecto a D de $\tilde{D}_I$. Más explícitamente

$$D = \tilde{D}_I + \tilde{D}_\Sigma \quad \text{y} \quad \tilde{D}_I \cap \tilde{D}_\Sigma = \{0\} \tag{232}$$

y

$$D = \tilde{D}_I + \bar{D}_\Sigma \quad \text{y} \quad \tilde{D}_I \cap \bar{D}_\Sigma = \{0\}. \tag{233}$$

Adicionalmente $\bar{D}_\Sigma$ es ortogonal a $\tilde{D}_I$, i.e. $\bar{D}_\Sigma$ es el complemento ortogonal de $\tilde{D}_I$, como se muestra en la figura:

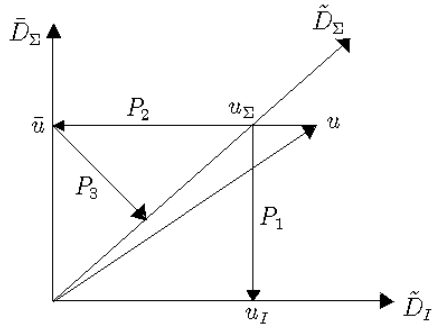

Figura 5: Esquematización de los subespacios $\tilde{D}_I, \tilde{D}_\Sigma$ y $\bar{D}_\Sigma$ del espacio D.

Sean

$$\xi_I = \left\{ w_I^i \mid i = 1, 2, ..., N_I \right\}, \quad \xi_\Sigma = \{ w_\Sigma^\alpha \mid \alpha = 1, 2, ..., N_\Sigma \} \tag{234}$$

bases linealmente independientes de $\tilde{D}_I$ y $\tilde{D}_\Sigma$ respectivamente, tales que

$$\xi = \xi_I \cup \xi_\Sigma \tag{235}$$

sea una base del espacio D, donde $N = N_I + N_\Sigma$ y sin perdida de generalidad podemos suponer que ξ es una base ordenada, es decir, primero están los vectores linealmente independientes de ξ_I y después los de ξ_Σ. Denotaremos al dual algebraico de D por D^*, el cual será el espacio lineal de funciones definidas en D.

Definimos también los operadores proyección P_1, P_2 y P_3 de $\tilde{D}_I, \bar{D}_\Sigma$ y $\tilde{D}_\Sigma$ respectivamente, notemos que podemos definir una biyección entre los subespacios $\bar{D}_\Sigma$ y $\tilde{D}_\Sigma$ por medio de las proyecciones $P_2 : \tilde{D}_\Sigma \to \bar{D}_\Sigma$ y $P_3 : \bar{D}_\Sigma \to \tilde{D}_\Sigma$, al tener la misma cardinalidad y ser complementos algebraicos con respecto a D de $\tilde{D}_I$.

Como trabajaremos con espacios de dimensión finita, cuando $\xi_I \subset \tilde{D}_I$ y $\xi_\Sigma \subset \tilde{D}_\Sigma$ se mantengan fijos, se puede asociar con cada $u \in D$ un único vector $\underline{u} \equiv (u_I, u_\Sigma)$ donde $u_I = \left(u_{I_1}, ..., u_{I_{N_I}} \right)$ y $u_\Sigma = \left(u_{\Sigma_1}, ..., u_{\Sigma_{N_\Sigma}} \right)$, entonces $\underline{u} = (u_1, ..., u_N) \in \mathbb{R}^N$ tal que

$$u = \sum_{i=1}^{N} u_i w^i \tag{236}$$

donde $w^i \in \xi$ y $u_i \in \mathbb{R}$ para $i = 1, ..., N$. En especial cuando $u_\Sigma \in \tilde{D}_\Sigma$ y $u_I \in \tilde{D}_I$

podemos asociar un único vector $\underline{u_\Sigma} \in \mathbb{R}^{N_\Sigma}$ y $\underline{u_I} \in \mathbb{R}^{N_I}$ respectivamente tal que

$$\underline{u_\Sigma} = \sum_{\alpha=1}^{N} u_\alpha w_\Sigma^\alpha \quad \text{y} \quad \underline{u_I} = \sum_{i=1}^{N} u_i w_I^i. \tag{237}$$

Las funciones de $D \to \mathbb{R}^N$ definidas de está manera son una biyeción a la cual nos referiremos como la *inmersión natural* de $\mathbb{R}^N$ en D. Adicionalmente, las imágenes bajo la inmersión natural de $\tilde{D}_I$ y $\tilde{D}_\Sigma$ son isomorfos a $\mathbb{R}^{N_I}$ y $\mathbb{R}^{N_\Sigma}$ respectivamente, sin perdida de generalidad podemos suponer que la base $(u_1, ..., u_N) \in \mathbb{R}^N$ es una base ordenada, es decir, suponemos que primero aparecen los vectores de $(v_1, ..., v_{N_I}) \in \mathbb{R}^{N_I}$ y después los vectores $(w_1, ..., w_{N_\Sigma}) \in \mathbb{R}^{N_\Sigma}$, entonces la base para $\mathbb{R}^N$ será $(v_1, ..., v_{N_I}, w_1, ..., w_{N_\Sigma})$.

El producto interior Euclidiano en $\mathbb{R}^N$, para cada par $\underline{u} \in \mathbb{R}^N$ y $\underline{v} \in \mathbb{R}^N$, denotado por $\underline{u} \cdot \underline{v}$, será definido por

$$\underline{u} \cdot \underline{v} \equiv \sum_{i=1}^{N} u_i v_i, \tag{238}$$

el cual no necesariamente coincide con el producto interior del espacio D definido en la Ec. (231).

Si adicionalmente suponemos que existe una familia ortogonal $\tilde{D}_{I_i}(\Omega_i)$ de subespacios linealmente independientes del subespacio de $\tilde{D}_I(\Omega)$, con $i = 1, ..., E$, es decir $\left\{\tilde{D}_{I_1}(\Omega_1), ..., \tilde{D}_{I_E}(\Omega_E)\right\}$ tales que

$$\tilde{D}_I = \sum_{i=1}^{E} \tilde{D}_{I_i} \tag{239}$$

y denotamos por $P_{1_i}, i = 1, ..., E$, al operador proyección sobre $\tilde{D}_{I_i}$, entonces

$$P_1 = \sum_{i=1}^{E} P_{1_i} \tag{240}$$

además, para cada $j = 1, ..., E$, sea

$$\xi_{I_j} \equiv \left\{w_I^j \mid i = 1, ..., E\right\} \tag{241}$$

tal que ξ_{I_j} es una base de $\tilde{D}_{I_j}$ (las funciones w_I^j pueden ser las ϕ_i usadas en el método de elemento finito o cualquier otro tipo de funciones base). En adición, asumimos que

$$\xi_I \equiv \bigcup_{j=1}^{E} \xi_{I_j} \equiv \xi_{I_1} \cup \xi_{I_2} \cup ... \cup \xi_{I_E} \tag{242}$$

y que el orden del conjunto $\xi_I \equiv \left\{ w_I^i \mid i = 1, ..., N_I \right\}$ y es la dada por la Ec. (242). Obsérvese la siguiente implicación lógica

$$\text{Si } w_{I_\delta}^i, w_{I_{\delta'}}^{i'} \in \xi_I \quad \text{y} \quad \delta' \neq \delta \Longrightarrow \left\langle w_{I_\delta}^i, w_{I_{\delta'}}^{i'} \right\rangle = 0. \tag{243}$$

Entonces definiendo para toda $\delta = 1, ..., E$, la matriz de $N_\delta \times N_\delta$

$$\underline{\underline{A}}_\delta^{II} \equiv \left[\left\langle w_\delta^i, w_\delta^j \right\rangle \right] \tag{244}$$

que sólo esta definida en cada subespacio (subdominio Ω_δ). Entonces, la matriz virtual $\underline{\underline{A}}^{II}$ es dada por la matriz diagonal de la forma

$$\underline{\underline{A}}^{II} \equiv \begin{bmatrix} \underline{\underline{A}}_1^{II} & & & \\ & \underline{\underline{A}}_2^{II} & & \\ & & \ddots & \\ & & & \underline{\underline{A}}_E^{II} \end{bmatrix} \tag{245}$$

donde el resto de la matriz fuera de la diagonal en bloques es cero.

De forma similar definimos

$$\underline{\underline{A}}_\delta^{I\Sigma} \equiv \left[\left\langle w_I^i, w_\Sigma^\alpha \right\rangle \right], \quad \underline{\underline{A}}_\delta^{\Sigma I} \equiv \left[\left\langle w_\Sigma^\alpha, w_I^i \right\rangle \right] \tag{246}$$

y

$$\underline{\underline{A}}_\delta^{\Sigma\Sigma} \equiv \left[\langle w_\Sigma^\alpha, w_\Sigma^\alpha \rangle \right] \tag{247}$$

para toda $\delta = 1, ..., E$, obsérvese que $\underline{\underline{A}}_\delta^{I\Sigma} = \left(\underline{\underline{A}}_\delta^{\Sigma I} \right)^T$. Entonces las matrices virtuales $\underline{\underline{A}}^{\Sigma I}, \underline{\underline{A}}^{\Sigma I}$ y $\underline{\underline{A}}^{\Sigma\Sigma}$ quedarán definidas como

$$\underline{\underline{A}}^{I\Sigma} \equiv \begin{bmatrix} \underline{\underline{A}}_1^{I\Sigma} \\ \underline{\underline{A}}_2^{I\Sigma} \\ \vdots \\ \underline{\underline{A}}_E^{I\Sigma} \end{bmatrix} \tag{248}$$

$$\underline{\underline{A}}^{\Sigma I} \equiv \begin{bmatrix} \underline{\underline{A}}_1^{\Sigma I} & \underline{\underline{A}}_2^{\Sigma I} & \cdots & \underline{\underline{A}}_E^{\Sigma I} \end{bmatrix} \tag{249}$$

y

$$\underline{\underline{A}}^{\Sigma\Sigma} \equiv \left[\sum_{i=1}^{E} \underline{\underline{A}}_i^{\Sigma\Sigma} \right] \tag{250}$$

donde $\left[\sum_{i=1}^{E} \underline{\underline{A}}_i^{\Sigma\Sigma} \right]$ es construida sumando las $\underline{\underline{A}}_i^{\Sigma\Sigma}$ según el orden de los nodos globales versus los nodos locales.

También consideremos al vector $\underline{u} \equiv (u_1, ..., u_E)$ el cual puede ser escrito como $\underline{u} = (\underline{u_I}, \underline{u_\Sigma})$ donde $u_I = (u_1, ..., _{N_I})$ y $u_\Sigma = (u_1, ..., _{N_\Sigma})$.

Así, el sistema virtual

$$\begin{aligned} \underline{\underline{A}}^{II}\underline{u_I} + \underline{\underline{A}}^{I\Sigma}\underline{u_\Sigma} &= \underline{b_I} \\ \underline{\underline{A}}^{\Sigma I}\underline{u_I} + \underline{\underline{A}}^{\Sigma\Sigma}\underline{u_\Sigma} &= \underline{b_\Sigma} \end{aligned} \tag{251}$$

quedando expresado como

$$\begin{bmatrix} \underline{\underline{A}}_1^{II} & & \\ & \ddots & \\ & & \underline{\underline{A}}_E^{II} \end{bmatrix} \begin{bmatrix} \underline{u_I}_1 \\ \vdots \\ \underline{u_I}_E \end{bmatrix} + \begin{bmatrix} \underline{\underline{A}}_1^{I\Sigma} \\ \vdots \\ \underline{\underline{A}}_E^{I\Sigma} \end{bmatrix} \begin{bmatrix} \underline{u_\Sigma}_1 \\ \vdots \\ \underline{u_\Sigma}_E \end{bmatrix} = \begin{bmatrix} \underline{b_I}_1 \\ \vdots \\ \underline{b_I}_E \end{bmatrix}$$

$$\begin{bmatrix} \underline{\underline{A}}_1^{\Sigma I} & \cdots & \underline{\underline{A}}_E^{\Sigma I} \end{bmatrix} \begin{bmatrix} \underline{u_I}_1 \\ \vdots \\ \underline{u_I}_E \end{bmatrix} + \begin{bmatrix} \underline{\underline{A}}^{\Sigma\Sigma} \end{bmatrix} \begin{bmatrix} \underline{u_\Sigma}_1 \\ \vdots \\ \underline{u_\Sigma}_E \end{bmatrix} = \begin{bmatrix} \underline{b_\Sigma}_1 \\ \vdots \\ \underline{b_\Sigma}_E \end{bmatrix}$$

o más compactamente como $\underline{\underline{A}}\underline{u} = \underline{b}$.

Si del sistema anterior eliminamos $\underline{u_I}$ nos queda

$$\left(\underline{\underline{A}}^{\Sigma\Sigma} - \underline{\underline{A}}^{\Sigma I}\left(\underline{\underline{A}}^{II}\right)^{-1}\underline{\underline{A}}^{I\Sigma}\right)\underline{u_\Sigma} = \underline{b_\Sigma} - \underline{\underline{A}}^{\Sigma I}\left(\underline{\underline{A}}^{II}\right)^{-1}\underline{b_I} \tag{252}$$

a la matriz

$$\underline{\underline{S}} = \underline{\underline{A}}^{\Sigma\Sigma} - \underline{\underline{A}}^{\Sigma I}\left(\underline{\underline{A}}^{II}\right)^{-1}\underline{\underline{A}}^{I\Sigma} \tag{253}$$

se le llama el complemento de Schur global.

En nuestro caso, tenemos definidas las matrices $\underline{\underline{A}}_i^{\Sigma\Sigma}, \underline{\underline{A}}_i^{\Sigma I}, \underline{\underline{A}}_i^{I\Sigma}$ y $\underline{\underline{A}}_i^{II}$ de manera local, por ello definimos el complemento de Schur local como

$$\underline{\underline{S}}_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I}\left(\underline{\underline{A}}_i^{II}\right)^{-1}\underline{\underline{A}}_i^{I\Sigma} \tag{254}$$

adicionalmente definimos

$$\underline{b}_i = \underline{\underline{A}}_i^{\Sigma I}\left(\underline{\underline{A}}_i^{II}\right)^{-1}\underline{b_I}_i \tag{255}$$

en cada subespacio $i = 1, 2, ..., E$. Notemos que las matrices $\underline{\underline{A}}_i^{\Sigma\Sigma}, \underline{\underline{A}}_i^{\Sigma I}, \underline{\underline{A}}_i^{I\Sigma}$ y $\underline{\underline{A}}_i^{II}$ son matrices bandadas y las matrices $\left(\underline{\underline{A}}_i^{II}\right)^{-1}$ y $\underline{\underline{S}}_i$ son matrices densas.

El sistema dado por la Ec. (252) lo escribimos como

$$\underline{\underline{S}}\,\underline{u_\Sigma} = \underline{b} \tag{256}$$

y queda definido de manera virtual a partir de

$$\left[\sum_{i=1}^{E}\underline{\underline{S}}_i\right]\underline{u_\Sigma} = \left[\sum_{i=1}^{E}\underline{b}_i\right] \tag{257}$$

donde $\left[\sum_{i=1}^{E} \underline{\underline{S}}_i\right]$ y $\left[\sum_{i=1}^{E} \underline{b}_i\right]$ podrían ser construida sumando las S_i y b_i respectivamente según el orden de los nodos globales versus los nodos locales.

El sistema lineal virtual obtenido de esta forma (256) se resuelve eficientemente usando el método de gradiente conjugado visto en la sección (5.2), para ello no es necesario construir la matriz $\underline{\underline{S}}$ con las contribuciones de cada S_i correspondientes al subdominio i, lo que hacemos es pasar a cada subdominio el vector $\underline{u_\Sigma}^i$ correspondiente a la i-ésima iteración del método de gradiente conjugado para que en cada subdominio se evalué $\underline{\tilde{u}_\Sigma}^i = \underline{\underline{S}}_i \underline{u_\Sigma}^i$ localmente y con el resultado se forma el vector $\underline{\tilde{u}_\Sigma} = \sum_{i=1}^{E} \underline{\tilde{u}_\Sigma}^i$ y se continué con los demás pasos del método. Esto es ideal para una implementación en paralelo del método de gradiente conjugado.

Una vez resuelto el sistema de la Ec. (257) en el que hemos encontrado la solución para los nodos de la frontera interior $\underline{u_\Sigma}$, entonces debemos resolver localmente los $\underline{u_I}_i$ correspondientes a los nodos interiores para cada subespacio Ω_i, para esto empleamos

$$\underline{u_I}_i = \left(\underline{\underline{A}}_i^{II}\right)^{-1} \left(\underline{b_I}_i - \underline{\underline{A}}_i^{\Sigma I} \underline{u_\Sigma}_i\right)$$

para cada $i = 1, 2, ..., E$, quedando así resuelto el problema $\underline{\underline{A}}\underline{u} = \underline{b}$ tanto en los nodos interiores $\underline{u_I}_i$ como en los de la frontera interior $\underline{u_\Sigma}_i$ correspondientes a cada subespacio Ω_i.

Para más detalles de la forma como se presento este método ver [10] y para ver otras formulaciones ver [5], [4] y [2].

Observaciones:

- La precisión del método es la misma que la usada por los interpoladores en la descomposición de los subdominios.
- El método es paralelizable permitiendo que cada subdominio sea manipulado por un procesador y además es posible usar en cada subdominio para el manejo de las matrices generadas diversos esquemas de paralelización para aumentar el rendimiento de los procesadores.
- Hay que notar que por cada subdominio (supóngase E) se resuelven sólo los nodos de la frontera interior k, esto significa que en promedio se usan menos de k iteraciones en el método de gradiente conjugado para resolver el sistema lineal asociado (estas iteraciones son en cantidad menor que las realizadas por el método de Schwarz para un sólo subdominio, si Ω_i es tomado aproximadamente del mismo tamaño en ambos métodos). Y como la solución de los nodos interiores no conllevan iteraciones adicionales, este método resulta más económico computacionalmente que el método de Schwarz. Está economía se hace aún más patente cuando se usa un buen precondicionador, logrando bajar hasta en un orden de magnitud el número de iteraciones del caso no precondicionado.

6.2.1. Precondicionador Derivado de la Matriz de Rigidez

En esta sección detallaremos la construcción del precondicionador derivado de la matriz de rigidez para problemas elípticos usando en el método de subestructuración. Para mayor información de estos y otros precondicionadores ver [11], [5], [4] y [2].

Para el caso en que Ω es subdividido en $E = 2$ subdominios $\bar{\Omega} = \bar{\Omega}_1 \cup \bar{\Omega}_2$, la matriz $\underline{\underline{A}}$ queda expresada como

$$\underline{\underline{A}} = \begin{pmatrix} \underline{\underline{A}}_1^{II} & 0 & \underline{\underline{A}}_1^{I\Sigma} \\ 0 & \underline{\underline{A}}_2^{II} & \underline{\underline{A}}_2^{I\Sigma} \\ \underline{\underline{A}}_1^{\Sigma I} & \underline{\underline{A}}_2^{\Sigma I} & \underline{\underline{A}}_1^{\Sigma\Sigma} + \underline{\underline{A}}_2^{\Sigma\Sigma} \end{pmatrix} \tag{258}$$

y puede expresarse de forma factorizada como

$$\underline{\underline{A}} = \underline{\underline{L}}\underline{\underline{D}}\underline{\underline{L}}^T \tag{259}$$

donde, denotando por $\underline{\underline{I}}_1, \underline{\underline{I}}_2$ y $\underline{\underline{I}}_\Sigma$ a las matrices identidad de dimensión N_1, N_2 y N_Σ respectivamente, y a

$$\underline{\underline{L}} = \begin{pmatrix} \underline{\underline{I}}_1 & 0 & 0 \\ 0 & \underline{\underline{I}}_2 & 0 \\ \underline{\underline{A}}_1^{\Sigma I}\left(\underline{\underline{A}}_1^{II}\right)^{-1} & \underline{\underline{A}}_2^{\Sigma I}\left(\underline{\underline{A}}_2^{II}\right)^{-1} & \underline{\underline{I}}_\Sigma \end{pmatrix}, \tag{260}$$

$$\underline{\underline{D}} = \begin{pmatrix} \underline{\underline{A}}_1^{II} & 0 & 0 \\ 0 & \underline{\underline{A}}_2^{II} & 0 \\ 0 & 0 & \underline{\underline{S}}_1 + \underline{\underline{S}}_2 \end{pmatrix} \tag{261}$$

y

$$S_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I}\left(\underline{\underline{A}}_i^{II}\right)^{-1}\underline{\underline{A}}_i^{I\Sigma}, \text{ con } i = 1, 2 \tag{262}$$

entonces, equivalentemente, obtendremos la siguiente descomposición por bloques $\underline{\underline{L}}\underline{\underline{U}}$ de la matriz $\underline{\underline{A}} = \underline{\underline{L}}\underline{\underline{U}}$ donde

$$\underline{\underline{U}} = \underline{\underline{D}}\underline{\underline{L}}^T = \begin{pmatrix} \underline{\underline{A}}_1^{II} & 0 & \underline{\underline{A}}_1^{I\Sigma} \\ 0 & \underline{\underline{A}}_2^{II} & \underline{\underline{A}}_2^{I\Sigma} \\ 0 & 0 & \underline{\underline{S}}_1 + \underline{\underline{S}}_2 \end{pmatrix}. \tag{263}$$

Asumiendo que el precondicionador $\underline{\underline{P}}_h$ conveniente esta disponible para la matriz $\underline{\underline{S}} = \underline{\underline{S}}_1 + \underline{\underline{S}}_2$, entonces podemos construir el siguiente precondicionador $\underline{\underline{Q}}_h$ de $\underline{\underline{A}}$:

$$\underline{\underline{Q}}_h = \underline{\underline{L}}\tilde{\underline{\underline{U}}} \tag{264}$$

donde $\underline{\underline{L}}$ es dada como en (260) y $\tilde{\underline{\underline{U}}}$ es obtenida de $\underline{\underline{U}}$ de (263) por la aproximación de $\underline{\underline{S}}$ con $\underline{\underline{P}}_h$; es decir

$$\tilde{\underline{\underline{U}}} = \begin{pmatrix} \underline{\underline{A}}_1^{II} & 0 & \underline{\underline{A}}_1^{I\Sigma} \\ 0 & \underline{\underline{A}}_2^{II} & \underline{\underline{A}}_2^{I\Sigma} \\ 0 & 0 & \underline{\underline{P}}_h \end{pmatrix}. \tag{265}$$

Notemos que los bloques de $\underline{\underline{Q}}_h$ coinciden con los de $\underline{\underline{A}}$, excepto para el bloque $(3,3)$, el cual es

$$\left(\underline{\underline{Q}}_h\right)_{3,3} = \underline{\underline{A}}_1^{\Sigma I}\left(\underline{\underline{A}}_1^{II}\right)^{-1}\underline{\underline{A}}_1^{I\Sigma} + \underline{\underline{A}}_2^{\Sigma I}\left(\underline{\underline{A}}_2^{II}\right)^{-1}\underline{\underline{A}}_2^{I\Sigma} + \underline{\underline{P}}_h \tag{266}$$

pero tomando como precondicionador $\underline{\underline{S}}_2$ obtenemos

$$\left(\underline{\underline{Q}}_h\right)_{3,3} = \underline{\underline{A}}_1^{\Sigma I}\left(\underline{\underline{A}}_1^{II}\right)^{-1}\underline{\underline{A}}_1^{I\Sigma} + \underline{\underline{A}}_2^{\Sigma\Sigma}. \tag{267}$$

Sea λ un eigenvalor de $\underline{\underline{Q}}_h\underline{\underline{A}}$ y $\underline{w} \in \mathbb{R}^{N_\Sigma}$ su correspondiente eigenvector, escribimos $\underline{w} = (\underline{w}_1, \underline{w}_2, \underline{w}_\Sigma)$, por obvias razones de notación, tenemos que

$$\underline{\underline{A}}\underline{w} = \lambda \underline{\underline{Q}}_h \underline{w} \tag{268}$$

donde

$$\begin{cases} (1-\lambda)\left(\underline{\underline{A}}_1^{II}\underline{w}_1 + \underline{\underline{A}}_1^{II}\underline{w}_\Sigma\right) = 0 \\ (1-\lambda)\left(\underline{\underline{A}}_2^{II}\underline{w}_2 + \underline{\underline{A}}_2^{II}\underline{w}_\Sigma\right) = 0 \\ (1-\lambda)\left(\underline{\underline{A}}_1^{\Sigma I}\underline{w}_1 + \underline{\underline{A}}_2^{\Sigma I}\underline{w}_2\right) + \underline{\underline{A}}^{\Sigma\Sigma}\underline{w}_\Sigma \\ = \lambda\left(\underline{\underline{P}}_h\underline{w}_\Sigma + \underline{\underline{A}}_1^{\Sigma I}\left(\underline{\underline{A}}_1^{II}\right)^{-1}\underline{\underline{A}}_1^{I\Sigma}\underline{w}_\Sigma + \underline{\underline{A}}_2^{\Sigma I}\left(\underline{\underline{A}}_2^{II}\right)^{-1}\underline{\underline{A}}_2^{I\Sigma}\underline{w}_\Sigma\right). \end{cases} \tag{269}$$

Reescribiendo la última ecuación como

$$(1-\lambda)\left(\underline{\underline{A}}_1^{\Sigma I}\underline{w}_1 + \underline{\underline{A}}_2^{\Sigma I}\underline{w}_2 + \underline{\underline{A}}_1^{\Sigma I}\left(\underline{\underline{A}}_1^{II}\right)^{-1}\underline{\underline{A}}_1^{I\Sigma}\underline{w}_\Sigma + \right. \tag{270}$$

$$\left.\underline{\underline{A}}_2^{\Sigma I}\left(\underline{\underline{A}}_2^{II}\right)^{-1}\underline{\underline{A}}_2^{I\Sigma}\underline{w}_\Sigma\right) + \underline{\underline{S}}\underline{w}_\Sigma = \lambda\underline{\underline{P}}_h\underline{w}_\Sigma.$$

y si $\lambda \neq 1$, tenemos

$$\begin{cases} \underline{\underline{A}}_1^{II}\underline{w}_1 + \underline{\underline{A}}_1^{I\Sigma}\underline{w}_\Sigma = 0 \\ \underline{\underline{A}}_2^{II}\underline{w}_2 + \underline{\underline{A}}_2^{I\Sigma}\underline{w}_\Sigma = 0 \end{cases} \tag{271}$$

por lo tanto, $\underline{w}_\Sigma \neq 0$ y $\underline{\underline{S}}\underline{w}_\Sigma = \lambda\underline{\underline{P}}_h\underline{w}_\Sigma$.

De lo anterior podemos concluir que la matriz $\left(\underline{\underline{Q}}_h\right)^{-1}\underline{\underline{A}}$ tiene los mismos eigenvalores que $\left(\underline{\underline{P}}_h\right)^{-1}$, además de el eigenvalor 1, el cual es también un eigenvalor de $\left(\underline{\underline{P}}_h\right)^{-1}\underline{\underline{S}}$ siempre que el correspondiente eigenvector $\underline{w}$ satisfasca que $\underline{w}_\Sigma \neq 0$.

Si asumimos que $\underline{\underline{P}}_h$ es espectralmente equivalente a $\underline{\underline{S}}$, es decir, existen dos constantes K_1 y K_2 independientes de h, tal que

$$K_1\left[\underline{\underline{P}}_h\underline{\eta}, \underline{\eta}\right] \leq \left[\underline{\underline{S}}\underline{\eta}, \underline{\eta}\right] \leq K_2\left[\underline{\underline{P}}_h\underline{\eta}, \underline{\eta}\right] \tag{272}$$

para toda $\eta \in \mathbb{R}^{N_\Sigma}$. Entonces se deriva de la caracterización de los eigenvalores de $\left(\underline{\underline{Q}}_h\right)^{-1} \underline{\underline{A}}$ que satisface

$$\kappa = \left(\left(\underline{\underline{Q}}_h\right)^{-1} \underline{\underline{A}} \right) \leq \frac{\tilde{K}_2}{\tilde{K}_1} \tag{273}$$

donde

$$\tilde{K}_1 = \text{mín}\left\{1, K_1\right\}, \tilde{K}_2 = \text{mín}\left\{1, K_2\right\}$$

por lo tanto podemos concluir que $\underline{\underline{Q}}_h$ es espectralmente equivalente a $\underline{\underline{A}}$.

En conclusión, el precondicionador $\underline{\underline{Q}}_h$ hereda todas las buenas propiedades que tiene $\underline{\underline{P}}_h$ en términos de buen paralelismo y equivalencia espectral.

En el caso de una partición con $E > 2$ subdominios se puede proceder de manera similar. En este caso Ω será particionado en E subdominios Ω_i que no se traslapen, de diámetro h_i, con Σ como frontera interior, es decir

$$\Sigma = \bigcup_{i=1}^{E} \Sigma_i, \quad \text{si } \Sigma_i = \partial\Omega_i \backslash \partial\Omega$$

y sea $\alpha = \bigcup_{i=1}^{E} N_i$ que denote los índices correspondientes a los nodos internos.

Entonces, para usar la misma notación podemos escribir el problema algebraico $\underline{\underline{A}}\underline{u} = \underline{b}$ en bloques como sigue

$$\begin{pmatrix} \underline{\underline{A}}_\alpha^{II} & \underline{\underline{A}}_\alpha^{I\Sigma} \\ \underline{\underline{A}}_\alpha^{\Sigma I} & \underline{\underline{A}}_\alpha^{\Sigma\Sigma} \end{pmatrix} \begin{pmatrix} \underline{u}_\alpha \\ \underline{u}_\Sigma \end{pmatrix} = \begin{pmatrix} \underline{b}_\alpha \\ \underline{b}_\Sigma \end{pmatrix} \tag{274}$$

donde $\underline{\underline{A}}_\alpha^{\Sigma I} = \left(\underline{\underline{A}}_\alpha^{I\Sigma}\right)^T$.

Ya que los nodos interiores de cada subdominio permanecen desacoplados de los nodos interiores de los demás subdominios, obtenemos

$$\underline{\underline{A}}^{II} \equiv \begin{bmatrix} \underline{\underline{A}}_1^{II} & 0 & \cdots & 0 \\ 0 & \underline{\underline{A}}_2^{II} & 0 & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & 0 & \underline{\underline{A}}_E^{II} \end{bmatrix} \tag{275}$$

el bloque i de $\underline{\underline{A}}_i^{II}$ es la submatriz principal de la matriz local de rigidez que esta asociada al problema del subdominio Ω_i.

En efecto, la matriz

$$\underline{\underline{A}}_i^{II} \equiv \begin{pmatrix} \underline{\underline{A}}_i^{II} & \underline{\underline{A}}_i^{I\Sigma} \\ \underline{\underline{A}}_i^{\Sigma I} & \underline{\underline{A}}_i^{\Sigma\Sigma} \end{pmatrix} \tag{276}$$

es la matriz de elemento finito asociado con el problema de Poisson en Ω_i con frontera Σ_i de Neumann.

Similarmente, la matriz

$$\underline{\underline{D}}_i \equiv \begin{pmatrix} \underline{\underline{A}}_i^{II} & \underline{\underline{A}}_i^{I\Sigma} \\ 0 & \underline{\underline{I}}_\Sigma \end{pmatrix} \tag{277}$$

donde $\underline{\underline{I}}_\Sigma$ es la matriz identidad de orden N_Σ que como en el caso anterior es asociada con el problema de Poisson en Ω_i con frontera Σ_i de Dirichlet.

La matriz del complemento de Schur $\underline{\underline{S}}$ asociada con las variables de la frontera interior u_Σ de (274) es

$$\underline{\underline{S}} = \underline{\underline{A}}^{\Sigma\Sigma} - \underline{\underline{A}}^{\Sigma I} \left(\underline{\underline{A}}^{II}\right)^{-1} \underline{\underline{A}}^{I\Sigma} \tag{278}$$

pero $\underline{\underline{S}} = \sum_{i=1}^{E} \underline{\underline{S}}_i$ donde

$$\underline{\underline{S}}_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I} \left(\underline{\underline{A}}_i^{II}\right)^{-1} \underline{\underline{A}}_i^{I\Sigma}. \tag{279}$$

En cuanto al número de condicionamiento de $\underline{\underline{S}}$ satisface

$$\kappa = \left(\underline{\underline{S}}\right) \leq C \frac{H_{\text{máx}}}{h H_{\text{mín}}^2} \tag{280}$$

donde $H_{\text{mín}}$ y $H_{\text{máx}}$ denotan respectivamente el mínimo y máximo de los diámetros de los subdominios.

Continuando con la aproximación directa usada en dos subdominios, obtenemos que la matriz de rigidez $\underline{\underline{A}}$ puede ser factorizada como sigue

$$\underline{\underline{A}} = \underline{\underline{L}}\,\underline{\underline{D}}\,\underline{\underline{L}}^T \tag{281}$$

con

$$\underline{\underline{L}} = \begin{pmatrix} \underline{\underline{I}}_{\Omega\setminus\Sigma} & 0 \\ \underline{\underline{A}}_\alpha^{\Sigma I} \left(\underline{\underline{A}}_\alpha^{II}\right)^{-1} & \underline{\underline{I}}_\Sigma \end{pmatrix}, \tag{282}$$

$$\underline{\underline{D}} = \begin{pmatrix} \underline{\underline{A}}_\alpha^{II} & 0 \\ 0 & \underline{\underline{S}} \end{pmatrix} \tag{283}$$

o

$$\underline{\underline{A}} = \underline{\underline{L}}\,\underline{\underline{U}} \tag{284}$$

con

$$\underline{\underline{U}} = \begin{pmatrix} \underline{\underline{A}}_\alpha^{II} & \underline{\underline{A}}_\alpha^{I\Sigma} \\ 0 & \underline{\underline{S}} \end{pmatrix}. \tag{285}$$

$\underline{\underline{P}}_h$ será un precondicionador para el complemento de Schur $\underline{\underline{S}}$, entonces la matriz

$$\underline{\underline{Q}}_h = \underline{\underline{L}}\,\underline{\underline{\tilde{U}}} \tag{286}$$

con

$$\underline{\underline{\tilde{U}}} = \begin{pmatrix} \underline{\underline{A}}_{\alpha}^{II} & \underline{\underline{A}}_{\alpha}^{I\Sigma} \\ 0 & \underline{\underline{P}}_{h} \end{pmatrix} \tag{287}$$

es un precondicionador para la matriz de rigidez $\underline{\underline{A}}$. Los eigenvalores de

$$\left(\underline{\underline{Q}}_{h}\right)^{-1} \underline{\underline{A}} \tag{288}$$

son los mismos que para

$$\left(\underline{\underline{P}}_{h}\right)^{-1} \underline{\underline{S}} \tag{289}$$

además del eigenvalor 1.

7. El Cómputo en Paralelo

Los sistemas de cómputo con procesamiento en paralelo surgen de la necesidad de resolver problemas complejos en un tiempo razonable, utilizando las ventajas de memoria, velocidad de los procesadores, formas de interconexión de estos y distribución de la tarea, a los que en su conjunto denominamos arquitectura en paralelo. Entenderemos por una arquitectura en paralelo a un conjunto de procesadores interconectados capaces de cooperar en la solución de un problema.

Así, para resolver un problema en particular, se usa una o combinación de múltiples arquitecturas (topologías), ya que cada una ofrece ventajas y desventajas que tienen que ser sopesadas antes de implementar la solución del problema en una arquitectura en particular. También es necesario conocer los problemas a los que se enfrenta un desarrollador de programas que se desean correr en paralelo, como son: el partir eficientemente un problema en múltiples tareas y como distribuir estas según la arquitectura en particular con que se trabaje.

7.1. Arquitecturas de Software y Hardware

En esta sección se explican en detalle las dos clasificaciones de computadoras más conocidas en la actualidad. La primera clasificación, es la clasificación clásica de Flynn en dónde se tienen en cuenta sistemas con uno o varios procesadores, la segunda clasificación es moderna en la que sólo tienen en cuenta los sistemas con más de un procesador.

El objetivo de esta sección es presentar de una forma clara los tipos de clasificación que existen en la actualidad desde el punto de vista de distintos autores, así como cuáles son las ventajas e inconvenientes que cada uno ostenta, ya que es común que al resolver un problema particular se usen una o más arquitecturas de hardware interconectadas generalmente por red.

7.1.1. Clasificación de Flynn

Clasificación clásica de arquitecturas de computadoras que hace alusión a sistemas con uno o varios procesadores, Flynn la publicó por primera vez en 1966 y por segunda vez en 1970.

Esta taxonomía se basa en el flujo que siguen los datos dentro de la máquina y de las instrucciones sobre esos datos. Se define como flujo de instrucciones al conjunto de instrucciones secuenciales que son ejecutadas por un único procesador y como flujo de datos al flujo secuencial de datos requeridos por el flujo de instrucciones.

Con estas consideraciones, Flynn clasifica los sistemas en cuatro categorías:

Single Instruction stream, Single Data stream (SISD) Los sistemas de este tipo se caracterizan por tener un único flujo de instrucciones sobre un único flujo de datos, es decir, se ejecuta una instrucción detrás de otra. Este es el concepto de arquitectura serie de Von Neumann donde, en cualquier

momento, sólo se ejecuta una única instrucción, un ejemplo de estos sistemas son las máquinas secuenciales convencionales.

Single Instruction stream, Multiple Data stream (SIMD) Estos sistemas tienen un único flujo de instrucciones que operan sobre múltiples flujos de datos. Ejemplos de estos sistemas los tenemos en las máquinas vectoriales con hardware escalar y vectorial.

El procesamiento es síncrono, la ejecución de las instrucciones sigue siendo secuencial como en el caso anterior, todos los elementos realizan una misma instrucción pero sobre una gran cantidad de datos. Por este motivo existirá concurrencia de operación, es decir, esta clasificación es el origen de la máquina paralela.

El funcionamiento de este tipo de sistemas es el siguiente. La unidad de control manda una misma instrucción a todas las unidades de proceso (ALUs). Las unidades de proceso operan sobre datos diferentes pero con la misma instrucción recibida.

Existen dos alternativas distintas que aparecen después de realizarse esta clasificación:

- Arquitectura Vectorial con segmentación, una CPU única particionada en unidades funcionales independientes trabajando sobre flujos de datos concretos.
- Arquitectura Matricial (matriz de procesadores), varias ALUs idénticas a las que el procesador da instrucciones, asigna una única instrucción pero trabajando sobre diferentes partes del programa.

Multiple Instruction stream, Single Data stream (MISD) Sistemas con múltiples instrucciones que operan sobre un único flujo de datos. Este tipo de sistemas no ha tenido implementación hasta hace poco tiempo. Los sistemas MISD se contemplan de dos maneras distintas:

- Varias instrucciones operando simultáneamente sobre un único dato.
- Varias instrucciones operando sobre un dato que se va convirtiendo en un resultado que será la entrada para la siguiente etapa. Se trabaja de forma segmentada, todas las unidades de proceso pueden trabajar de forma concurrente.

Multiple Instruction stream, Multiple Data stream (MIMD) Sistemas con un flujo de múltiples instrucciones que operan sobre múltiples datos. Estos sistemas empezaron a utilizarse antes de la década de los 80s. Son sistemas con memoria compartida que permiten ejecutar varios procesos simultáneamente (sistema multiprocesador).

Cuando las unidades de proceso reciben datos de una memoria no compartida estos sistemas reciben el nombre de MULTIPLE SISD (MSISD). En

arquitecturas con varias unidades de control (MISD Y MIMD), existe otro nivel superior con una unidad de control que se encarga de controlar todas las unidades de control del sistema (ejemplo de estos sistemas son las máquinas paralelas actuales).

7.1.2. Categorías de Computadoras Paralelas

Clasificación moderna que hace alusión única y exclusivamente a los sistemas que tienen más de un procesador (i.e máquinas paralelas). Existen dos tipos de sistemas teniendo en cuenta su acoplamiento:

- Los sistemas fuertemente acoplados son aquellos en los que los procesadores dependen unos de otros.
- Los sistemas débilmente acoplados son aquellos en los que existe poca interacción entre los diferentes procesadores que forman el sistema.

Atendiendo a esta y a otras características, la clasificación moderna divide a los sistemas en dos tipos: Sistemas multiprocesador (fuertemente acoplados) y sistemas multicomputadoras (débilmente acoplados).

Multiprocesadores o Equipo Paralelo de Memoria Compartida Un multiprocesador puede verse como una computadora paralela compuesta por varios procesadores interconectados que comparten un mismo sistema de memoria.

Los sistemas multiprocesadores son arquitecturas MIMD con memoria compartida. Tienen un único espacio de direcciones para todos los procesadores y los mecanismos de comunicación se basan en el paso de mensajes desde el punto de vista del programador.

Dado que los multiprocesadores comparten diferentes módulos de memoria, pudiendo acceder a un mismo módulo varios procesadores, a los multiprocesadores también se les llama sistemas de memoria compartida.

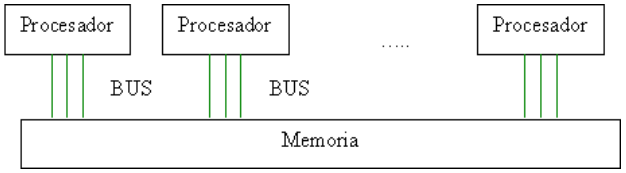

Figura 6: Arquitectura de una computadora paralela con memoria compartida

Para hacer uso de la memoria compartida por más de un procesador, se requiere hacer uso de técnicas de semáforos que mantienen la integridad de la memoria; esta arquitectura no puede crecer mucho en el número de procesadores interconectados por la saturación rápida del bus o del medio de interconexión.

Dependiendo de la forma en que los procesadores comparten la memoria, se clasifican en sistemas multiprocesador UMA, NUMA, COMA y Pipeline.

Uniform Memory Access (UMA) Sistema multiprocesador con acceso uniforme a memoria. La memoria física es uniformemente compartida por todos los procesadores, esto quiere decir que todos los procesadores tienen el mismo tiempo de acceso a todas las palabras de la memoria. Cada procesador tiene su propia caché privada y también se comparten los periféricos.

Los multiprocesadores son sistemas fuertemente acoplados (tightly-coupled), dado el alto grado de compartición de los recursos (hardware o software) y el alto nivel de interacción entre procesadores, lo que hace que un procesador dependa de lo que hace otro.

El sistema de interconexión debe ser rápido y puede ser de uno de los siguientes tipos: bus común, red crossbar y red multietapa. Este modelo es conveniente para aplicaciones de propósito general y de tiempo compartido por varios usuarios, existen dos categorías de sistemas UMA.

- Sistema Simétrico

  Cuando todos los procesadores tienen el mismo tiempo de acceso a todos los componentes del sistema (incluidos los periféricos), reciben el nombre de sistemas multiprocesador simétrico. Los procesadores tienen el mismo dominio (prioridad) sobre los periféricos y cada procesador tiene la misma capacidad para procesar.

- Sistema Asimétrico

  Los sistemas multiprocesador asimétrico, son sistemas con procesadores maestros y procesadores esclavos, en donde sólo los primeros pueden ejecutar aplicaciones y dónde en tiempo de acceso para diferentes procesadores no es el mismo. Los procesadores esclavos (attached) ejecutan código usuario bajo la supervisión del maestro, por lo tanto cuando una aplicación es ejecutada en un procesador maestro dispondrá de una cierta prioridad.

Non Uniform Memory Access (NUMA) Un sistema multiprocesador NUMA es un sistema de memoria compartida donde el tiempo de acceso varía según donde se encuentre localizado el acceso.

El acceso a memoria, por tanto, no es uniforme para diferentes procesadores, existen memorias locales asociadas a cada procesador y estos pueden acceder a datos de su memoria local de una manera más rápida que a las memorias de otros procesadores, debido a que primero debe aceptarse dicho acceso por el procesador del que depende el módulo de memoria local.

Todas las memorias locales conforman la memoria global compartida y físicamente distribuida y accesible por todos los procesadores.

Cache Only Memory Access (COMA) Los sistemas COMA son un caso especial de los sistemas NUMA. Este tipo de sistemas no ha tenido mucha trascendencia, al igual que los sistemas SIMD.

Las memorias distribuidas son memorias cachés, por este motivo es un sistema muy restringido en cuanto a la capacidad de memoria global. No hay jerarquía de memoria en cada módulo procesador. Todas las cachés forman un mismo espacio global de direcciones. El acceso a las cachés remotas se realiza a través de los directorios distribuidos de las cachés.

Dependiendo de la red de interconexión utilizada, se pueden utilizar jerarquías en los directorios para ayudar a la localización de copias de bloques de caché.

Procesador Vectorial Pipeline En la actualidad es común encontrar en un solo procesador los denominados Pipeline o Procesador Vectorial Pipeline del tipo MISD. En estos procesadores los vectores fluyen a través de las unidades aritméticas Pipeline.

Las unidades constan de una cascada de etapas de procesamiento compuestas de circuitos que efectúan operaciones aritméticas o lógicas sobre el flujo de datos que pasan a través de ellas, las etapas están separadas por registros de alta velocidad usados para guardar resultados intermedios. Así la información que fluye entre las etapas adyacentes está bajo el control de un reloj que se aplica a todos los registros simultáneamente.

Multicomputadoras o Equipo Paralelo de Memoria Distribuida Los sistemas multicomputadoras se pueden ver como una computadora paralela en el cual cada procesador tiene su propia memoria local. En estos sistemas la memoria se encuentra distribuida y no compartida como en los sistemas multiprocesador. Los procesadores se comunican a través de paso de mensajes, ya que éstos sólo tienen acceso directo a su memoria local y no a las memorias del resto de los procesadores.

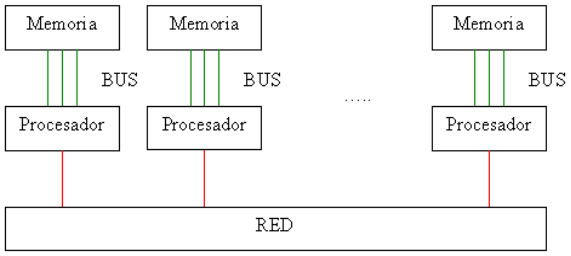

Figura 7: Arquitectura de una computadora paralela con memoria distribuida

La transferencia de los datos se realiza a través de la red de interconexión que conecta un subconjunto de procesadores con otro subconjunto. La transferencia de unos procesadores a otros se realiza por múltiples transferencias entre procesadores conectados dependiendo del establecimiento de dicha red.

Dado que la memoria está distribuida entre los diferentes elementos de proceso, estos sistemas reciben el nombre de distribuidos. Por otra parte, estos sistemas son débilmente acoplados, ya que los módulos funcionan de forma casi independiente unos de otros. Este tipo de memoria distribuida es de acceso lento por ser peticiones a través de la red, pero es una forma muy efectiva de tener acceso a un gran volumen de memoria.

Equipo Paralelo de Memoria Compartida-Distribuida La tendencia actual en las máquinas paralelas es de aprovechar las facilidades de programación que ofrecen los ambientes de memoria compartida y la escalabilidad de las ambientes de memoria distribuida. En este modelo se conectan entre si módulos de multiprocesadores, pero se mantiene la visión global de la memoria a pesar de que es distribuida.

Clusters El desarrollo de sistemas operativos y compiladores del dominio público (Linux y software GNU), estándares para el pase de mensajes (MPI), conexión universal a periféricos (PCI), etc. han hecho posible tomar ventaja de los económicos recursos computacionales de producción masiva (CPU, discos, redes).

La principal desventaja que presenta a los proveedores de multicomputadoras es que deben satisfacer una amplia gama de usuarios, es decir, deben ser generales. Esto aumenta los costos de diseños y producción de equipos, así como los costos de desarrollo de software que va con ellos: sistema operativo, compiladores y aplicaciones. Todos estos costos deben ser añadidos cuando se hace una venta. Por supuesto alguien que sólo necesita procesadores y un mecanismo de pase de mensajes no debería pagar por todos estos añadidos que nunca usará. Estos usuarios son los que están impulsando el uso de clusters principalmente de computadoras personales (PC), cuya arquitectura se muestra a continuación:

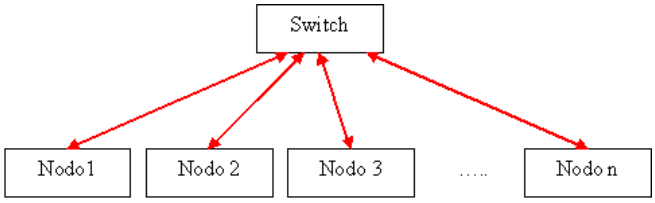

Figura 8: Arquitectura de un cluster

Los cluster se pueden clasificar en dos tipos según sus características físicas:

- Cluster homogéneo si todos los procesadores y/o nodos participantes en el equipo paralelo son iguales en capacidad de cómputo (en la cual es permitido variar la cantidad de memoria o disco duro en cada procesador).

- Cluster heterogéneo es aquel en que al menos uno de los procesadores y/o nodos participantes en el equipo paralelo son de distinta capacidad de cómputo.

Los cluster pueden formarse de diversos equipos; los más comunes son los de computadoras personales, pero es creciente el uso de computadoras multiprocesador de más de un procesador de memoria compartida interconectados por red con los demás nodos del mismo tipo, incluso el uso de computadoras multiprocesador de procesadores vectoriales Pipeline. Los cluster armados con la configuración anterior tienen grandes ventajas para procesamiento paralelo:

• La reciente explosión en redes implica que la mayoría de los componentes necesarios para construir un cluster son vendidos en altos volúmenes y por lo tanto son económicos. Ahorros adicionales se pueden obtener debido a que sólo se necesitará una tarjeta de vídeo, un monitor y un teclado por cluster. El mercado de los multiprocesadores es más reducido y más costoso.

• Remplazar un componente defectuoso en un cluster es relativamente trivial comparado con hacerlo en un multiprocesador, permitiendo una mayor disponibilidad de clusters cuidadosamente diseñados.

Desventajas del uso de clusters de computadoras personales para procesamiento paralelo:

• Con raras excepciones, los equipos de redes generales producidos masivamente no están diseñados para procesamiento paralelo y típicamente su latencia es alta y los anchos de banda pequeños comparados con multiprocesadores. Dado que los clusters explotan tecnología que sea económica, los enlaces en el sistema no son veloces implicando que la comunicación entre componentes debe pasar por un proceso de protocolos de negociación lentos, incrementando seriamente la latencia. En muchos y en el mejor de los casos (debido a costos) se recurre a una red tipo Fast Ethernet restringimiento la escalabilidad del cluster.

• Hay poco soporte de software para manejar un cluster como un sistema integrado.

• Los procesadores no son tan eficientes como los procesadores usados en los multiprocesadores para manejar múltiples usuarios y/o procesos. Esto hace que el rendimiento de los clusters se degrade con relativamente pocos usuarios y/o procesos.

• Muchas aplicaciones importantes disponibles en multiprocesadores y optimizadas para ciertas arquitecturas, no lo están en clusters.

Sin lugar a duda los clusters presentan una alternativa importante para varios problemas particulares, no sólo por su economía, si no también por que

pueden ser diseñados y ajustados para ciertas aplicaciones. Las aplicaciones que pueden sacar provecho de clusters son en donde el grado de comunicación entre procesos es de bajo a medio.

`Tipos de Cluster`

Básicamente existen tres tipo de clusters, cada uno de ellos ofrece ventajas y desventajas, el tipo más adecuado para el cómputo científico es del de alto-rendimiento, pero existen aplicaciones científicas que pueden usar más de un tipo al mismo tiempo.

- Alta-disponibilidad (Fail-over o High-Availability): este tipo de cluster esta diseñado para mantener uno o varios servicios disponibles incluso a costa de rendimiento, ya que su función principal es que el servicio jamás tenga interrupciones como por ejemplo un servicio de bases de datos.
- Alto-rendimiento (HPC o High Performance Computing): este tipo de cluster está diseñado para obtener el máximo rendimiento de la aplicación utilizada incluso a costa de la disponibilidad del sistema, es decir el cluster puede sufrir caídas, este tipo de configuración está orientada a procesos que requieran mucha capacidad de cálculo.
- Balanceo de Carga (Load-balancing): este tipo de cluster esta diseñado para balancear la carga de trabajo entre varios servidores, lo que permite tener, por ejemplo, un servicio de cálculo intensivo multiusuarios que detecte tiempos muertos del proceso de un usuario para ejecutar en dichos tiempos procesos de otros usuarios.

Grids Son cúmulos (grupo de clusters) de arquitecturas en paralelo interconectados por red, los cuales distribuyen tareas entre los clusters que lo forman, estos pueden ser homogéneos o heterogéneos en cuanto a los nodos componentes del cúmulo. Este tipo de arquitecturas trata de distribuir cargas de trabajo acorde a las características internas de cada cluster y las necesidades propias de cada problema, esto se hace a dos niveles, una en la parte de programación conjuntamente con el balance de cargas y otra en la parte de hardware que tiene que ver con las características de cada arquitectura que conforman al cúmulo.

7.2. Métricas de Desempeño

Las métricas de desempeño del procesamiento de alguna tarea en paralelo es un factor importante para medir la eficiencia y consumo de recursos al resolver una tarea con un número determinado de procesadores y recursos relacionados de la interconexión de éstos.

Entre las métricas para medir desempeño en las cuales como premisa se mantiene fijo el tamaño del problema, destacan las siguientes: Factor de aceleración, eficiencia y fracción serial. Cada una de ellas mide algo en particular

y sólo la combinación de estas dan un panorama general del desempeño del procesamiento en paralelo de un problema en particular en una arquitectura determinada al ser comparada con otras.

Factor de Aceleración (o Speed-Up) Se define como el cociente del tiempo que se tarda en completar el cómputo de la tarea usando un sólo procesador entre el tiempo que necesita para realizarlo en p procesadores trabajando en paralelo

$$s = \frac{T(1)}{T(p)} \tag{290}$$

en ambos casos se asume que se usará el mejor algoritmo tanto para un solo procesador como para p procesadores.

Esta métrica en el caso ideal debería de aumentar de forma lineal al aumento del número de procesadores.

Eficiencia Se define como el cociente del tiempo que se tarda en completar el cómputo de la tarea usando un solo procesador entre el número de procesadores multiplicado por el tiempo que necesita para realizarlo en p procesadores trabajando en paralelo

$$e = \frac{T(1)}{pT(p)} = \frac{s}{p}. \tag{291}$$

Este valor será cercano a la unidad cuando el hardware se esté usando de manera eficiente, en caso contrario el hardware será desaprovechado.

Fracción serial Se define como el cociente del tiempo que se tarda en completar el cómputo de la parte secuencial de una tarea entre el tiempo que se tarda el completar el cómputo de la tarea usando un solo procesador

$$f = \frac{T_s}{T(1)} \tag{292}$$

pero usando la ley de Amdahl

$$T(p) = T_s + \frac{T_p}{p}$$

y rescribiéndola en términos de factor de aceleración, obtenemos la forma operativa del cálculo de la fracción serial que adquiere la forma siguiente

$$f = \frac{\frac{1}{s} - \frac{1}{p}}{1 - \frac{1}{p}}. \tag{293}$$

Esta métrica permite ver las inconsistencias en el balance de cargas, ya que su valor debiera de tender a cero en el caso ideal, por ello un incremento en el valor de f es un aviso de granularidad fina con la correspondiente sobrecarga en los procesos de comunicación.

7.3. Cómputo Paralelo para Sistemas Continuos

Como se mostró en los capítulos anteriores, la solución de los sistemas continuos usando ecuaciones diferenciales parciales genera un alto consumo de memoria e involucran un amplio tiempo de procesamiento; por ello nos interesa trabajar en computadoras que nos puedan satisfacer estas demandas.

Actualmente, en muchos centros de cómputo es una práctica común usar directivas de compilación en equipos paralelos sobre programas escritos de forma secuencial, con la esperanza que sean puestos por el compilador como programas paralelos. Esto en la gran mayoría de los casos genera códigos poco eficientes, pese a que corren en equipos paralelos y pueden usar toda la memoria compartida de dichos equipos, el algoritmo ejecutado continua siendo secuencial en la gran mayoría del código.

Si la arquitectura paralela donde se implemente el programa es UMA de acceso simétrico, los datos serán accesados a una velocidad de memoria constante. En caso contrario, al acceder a un conjunto de datos es común que una parte de estos sean locales a un procesador (con un acceso del orden de nano segundos), pero el resto de los datos deberán de ser accesados mediante red (con acceso del orden de mili segundos), siendo esto muy costoso en tiempo de procesamiento.

Por ello, si usamos métodos de descomposición de dominio es posible hacer que el sistema algebraico asociado pueda distribuirse en la memoria local de múltiples computadoras y que para encontrar la solución al problema se requiera poca comunicación entre los procesadores.

Por lo anterior, si se cuenta con computadoras con memoria compartida o que tengan interconexión por bus, salvo en casos particulares no será posible explotar éstas características eficientemente. Pero en la medida en que se adecuen los programas para usar bibliotecas y compiladores acordes a las características del equipo disponible (algunos de ellos sólo existen de manera comercial) la eficiencia aumentará de manera importante.

La alternativa más adecuada (en costo y flexibilidad), es trabajar con computadoras de escritorio interconectadas por red que pueden usarse de manera cooperativa para resolver nuestro problema. Los gastos en la interconexión de los equipos son mínimos (sólo el switch y una tarjeta de red por equipo y cables para su conexión). Por ello los clusters y los grids son en principio una buena opción para la resolución de este tipo de problemas.

Esquema de Paralelización Maestro-Esclavo La implementación de los métodos de descomposición de dominio que se trabajarán será mediante el esquema Maestro-Esclavo (Farmer) en el lenguaje de programación C++ bajo la interfaz de paso de mensajes MPI trabajando en un cluster Linux Debian.

Donde tomando en cuenta la implementación en estrella del cluster, el modelo de paralelismo de MPI y las necesidades propias de comunicación del programa, el nodo maestro tendrá comunicación sólo con cada nodo esclavo y no existirá comunicación entre los nodos esclavos, esto reducirá las comunicaciones y optimizará el paso de mensajes.

El esquema de paralelización maestro-esclavo, permite sincronizar por parte

del nodo maestro las tareas que se realizan en paralelo usando varios nodos esclavos, éste modelo puede ser explotado de manera eficiente si existe poca comunicación entre el maestro y el esclavo y los tiempos consumidos en realizar las tareas asignadas son mayores que los períodos involucrados en las comunicaciones para la asignación de dichas tareas. De esta manera se garantiza que la mayoría de los procesadores estarán trabajando de manera continua y existirán pocos tiempos muertos.

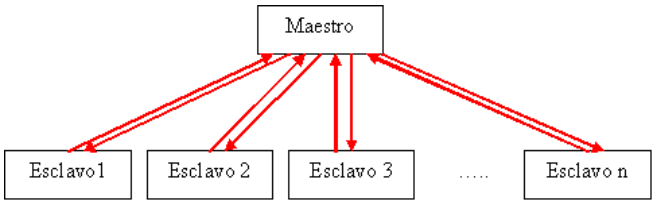

Figura 9: Esquema del maestro-esclavo

Un factor limitante en este esquema es que el nodo maestro deberá de atender todas las peticiones hechas por cada uno de los nodos esclavos, esto toma especial relevancia cuando todos o casi todos los nodos esclavos compiten por ser atendidos por el nodo maestro.

Se recomienda implementar este esquema en un cluster heterogéneo en donde el nodo maestro sea más poderoso computacionalmente que los nodos esclavos. Si a éste esquema se le agrega una red de alta velocidad y de baja latencia, se le permitirá operar al cluster en las mejores condiciones posibles, pero este esquema se verá degradado al aumentar el número de nodos esclavos inexorablemente.

Pero hay que ser cuidadosos en cuanto al número de nodos esclavos que se usan en la implementación en tiempo de ejecución versus el rendimiento general del sistema al aumentar estos, algunas observaciones posibles son:

- El esquema maestro-esclavo programado en C++ y usando MPI lanza P procesos (uno para el nodo maestro y $P-1$ para los nodos esclavos), estos en principio corren en un solo procesador pero pueden ser lanzados en múltiples procesadores usando una directiva de ejecución, de esta manera es posible que en una sola maquina se programe, depure y sea puesto a punto el código usando mallas pequeñas (del orden de cientos de nodos) y cuando este listo puede mandarse a producción en un cluster.

- El esquema maestro-esclavo no es eficiente si sólo se usan dos procesadores (uno para el nodo maestro y otro para el nodo esclavo), ya que el nodo maestro en general no realiza los cálculos pesados y su principal función será la de distribuir tareas; los cálculos serán delegados al nodo esclavo. En el caso que nos interesa implementar, el método de descomposición de dominio adolece de este problema.

Paso de Mensajes Usando MPI Para poder intercomunicar al nodo maestro con cada uno de los nodos esclavos se usa la interfaz de paso de mensajes (MPI), una biblioteca de comunicación para procesamiento en paralelo. MPI ha sido desarrollado como un estándar para el paso de mensajes y operaciones relacionadas.

Este enfoque es adoptado por usuarios e implementadores de bibliotecas, en la cual se proveen a los programas de procesamiento en paralelo de portabilidad y herramientas necesarias para desarrollar aplicaciones que puedan usar el cómputo paralelo de alto desempeño.

El modelo de paso de mensajes posibilita a un conjunto de procesos que tienen solo memoria local la comunicación con otros procesos (usando Bus o red) mediante el envío y recepción de mensajes. Por definición el paso de mensajes posibilita transferir datos de la memoria local de un proceso a la memoria local de cualquier otro proceso que lo requiera.

En el modelo de paso de mensajes para equipos paralelos, los procesos se ejecutan en paralelo, teniendo direcciones de memoria separada para cada proceso, la comunicación ocurre cuando una porción de la dirección de memoria de un proceso es copiada mediante el envío de un mensaje dentro de otro proceso en la memoria local mediante la recepción del mismo.

Las operaciones de envío y recepción de mensajes es cooperativa y ocurre sólo cuando el primer proceso ejecuta una operación de envío y el segundo proceso ejecuta una operación de recepción, los argumentos base de estas funciones son:

- Para el que envía, la dirección de los datos a transmitir y el proceso destino al cual los datos se enviarán.
- Para el que recibe, debe de tener la dirección de memoria donde se pondrán los datos recibidos, junto con la dirección del proceso del que los envío.

Es decir:
Send(dir, lg, td, dest, etiq, com)

$\{dir, lg, td\}$ describe cuántas ocurrencias *lg* de elementos del tipo de dato *td* se transmitirán empezando en la dirección de memoria *dir*.

$\{des, etiq, com\}$ describe el identificador *etq* de destino *des* asociado con la comunicación *com*.

Recv(dir, mlg, td, fuent, etiq, com, st)

$\{dir, lg, td\}$ describe cuántas ocurrencias *lg* de elementos del tipo de dato *td* se transmitirán empezando en la dirección de memoria *dir*.

$\{fuent, etiq, com, est\}$ describe el identificador *etq* de la fuente *fuent* asociado con la comunicación *com* y el estado *st*.

El conjunto básico de directivas (en nuestro caso sólo se usan estas) en C++ de MPI son:

MPI::Init	Inicializa al MPI
MPI::COMM_WORLD.Get_size	Busca el número de procesos existentes
MPI::COMM_WORLD.Get_rank	Busca el identificador del proceso
MPI::COMM_WORLD.Send	Envía un mensaje
MPI::COMM_WORLD.Recv	Recibe un mensaje
MPI::Finalize	Termina al MPI

Estructura del Programa Maestro-Esclavo La estructura del programa se realizo para que el nodo maestro mande trabajos de manera síncrona a los nodos esclavos. Cuando los nodos esclavos terminan la tarea asignada, avisan al nodo maestro para que se le asigne otra tarea (estas tareas son acordes a la etapa correspondiente del método de descomposición de dominio ejecutándose en un instante dado). En la medida de lo posible se trata de mandar paquetes de datos a cada nodo esclavo y que estos regresen también paquetes al nodo maestro, a manera de reducir las comunicaciones al mínimo y tratar de mantener siempre ocupados a los nodos esclavos para evitar los tiempos muertos, logrando con ello una granularidad gruesa, ideal para trabajar con clusters.

La estructura básica del programa bajo el esquema maestro-esclavo codificada en C++ y usando MPI es:

```
main(int argc, char *argv[])
{
    MPI::Init(argc,argv);
    ME_id = MPI::COMM_WORLD.Get_rank();
    MP_np = MPI::COMM_WORLD.Get_size();
    if (ME_id == 0) {
        // Operaciones del Maestro
    } else {
        // Operaciones del esclavo con identificador ME_id
    }
    MPI::Finalize();
}
```

En este único programa se deberá de codificar todas las tareas necesarias para el nodo maestro y cada uno de los nodos esclavos, así como las formas de intercomunicación entre ellos usando como distintivo de los distintos procesos a la variable *ME_id*. Para más detalles de esta forma de programación y otras funciones de MPI ver [21] y [6].

Los factores limitantes para el esquema maestro-esclavo pueden ser de dos tipos, los inherentes al propio esquema maestro-esclavo y al método de descomposición de dominio:

- El esquema de paralelización maestro-esclavo presupone contar con un nodo maestro lo suficientemente poderoso para atender simultáneamente las tareas síncronas del método de descomposición de dominio, ya que este distribuye tareas acorde al número de subdominios, estas si son balanceadas ocasionaran que todos los procesadores esclavos terminen al mismo tiempo y el nodo maestro tendrá que atender múltiples comunicaciones simultáneamente, degradando su rendimiento al aumentar el número de subdominios.
- Al ser síncrono el método de descomposición de dominio, si un nodo esclavo acaba la tarea asignada y avisa al nodo maestro, este no podrá asignarle otra tarea hasta que todos los nodos esclavos concluyan la suya.

Para los factores limitantes inherente al propio esquema maestro-esclavo, es posible implementar algunas operaciones del nodo maestro en paralelo, ya sea usando equipos multiprocesador o en más de un nodo distintos a los nodos esclavos.

Para la parte inherente al método de descomposición de dominio, la parte medular la da el balanceo de cargas. Es decir que cada nodo esclavo tenga una carga de trabajo igual al resto de los nodos. Este balanceo de cargas puede no ser homogéneo por dos razones:

- Al tener P procesadores en el equipo paralelo, la descomposición del dominio no sea la adecuada.
- Si se tiene una descomposición particular, esta se implemente en un número de procesadores inadecuado.

Cualquiera de las dos razones generarán desbalanceo de la carga en los nodos esclavos, ocasionando una perdida de eficiencia en el procesamiento de un problema bajo una descomposición particular en una configuración del equipo paralelo especifica, es por esto que en algunos casos al aumentar el número de procesadores que resuelvan la tarea no se aprecia una disminución del de tiempo de procesamiento.

El número de procesadores P que se usen para resolver un dominio Ω y tener buen balance de cargas puede ser conocido si aplicamos el siguiente procedimiento: Si el dominio Ω se descompone en $n \times m$ subdominios (la partición gruesa), entonces se generarán $s = n * m$ subdominios Ω_i, en este caso, se tiene un buen balanceo de cargas si $(P-1) \mid s$. La partición fina se obtiene al descomponer a cada subdominio Ω_i en $p \times q$ subdominios.

Como ejemplo, supongamos que deseamos resolver el dominio Ω usando 81×81 nodos ($nodos = n*p+1$ y $nodos = m*q+1$), de manera inmediata nos surgen las siguientes preguntas: ¿cuales son las posibles descomposiciones validas? y ¿en cuantos procesadores se pueden resolver cada descomposición?. Para este ejemplo, sin hacer la tabla exhaustiva obtenemos:

Partición	Subdominios	Procesadores
1x2 y 80x40	2	2,3
1x4 y 80x20	5	2,5
1x5 y 80x16	6	2,6
2x1 y 40x80	2	2,3
2x2 y 40x40	4	2,3,5
2x4 y 40x20	8	2,3,5,9
2x5 y 40x16	10	2,3,6,11
2x8 y 40x10	16	2,3,5,9,17
4x1 y 20x80	4	2,3,5
4x2 y 20x40	8	2,3,5,9
4x4 y 20x20	16	2,3,5,9,17
4x5 y 20x16	20	2,3,5,6,11,21
5x1 y 16x80	5	2,6
5x2 y 16x40	10	2,3,6,11
5x4 y 16x20	20	2,3,5,6,11,21
5x5 y 16x16	25	2,6,26

De esta tabla es posible seleccionar (para este ejemplo en particular), las descomposiciones que se adecuen a las necesidades particulares del equipo con que se cuente. Sin embargo hay que tomar en cuenta siempre el número de nodos por subdominio de la partición fina, ya que un número de nodos muy grande puede que exceda la cantidad de memoria que tiene el nodo esclavo y un número pequeño estaría infrautilizando el poder computacional de los nodos esclavos. De las particiones seleccionadas se pueden hacer corridas de prueba para evaluar su rendimiento, hasta encontrar la que menor tiempo de ejecución consuma, maximizando así la eficiencia del equipo paralelo.

Programación Paralela en Multihilos En una computadora, sea secuencial o paralela, para aprovechar las capacidades crecientes del procesador, el sistema operativo divide su tiempo de procesamiento entre los distintos procesos, de forma tal que para poder ejecutar a un proceso, el kernel les asigna a cada proceso una prioridad y con ello una fracción del tiempo total de procesamiento, de forma tal que se pueda atender a todos y cada uno de los procesos de manera eficiente.

En particular, en la programación en paralelo usando MPI, cada proceso (que eventualmente puede estar en distinto procesador) se lanza como una copia del programa con datos privados y un identificador del proceso único, de tal forma que cada proceso sólo puede compartir datos con otro proceso mediante paso de mensajes.

Esta forma de lanzar procesos por cada tarea que se desee hacer en paralelo es costosa, por llevar cada una de ellas todo una gama de subprocesos para poderle asignar recursos por parte del sistema operativo. Una forma más eficiente de hacerlo es que un proceso pueda generar bloques de subprocesos que puedan ser ejecutados como parte del proceso (como subtareas), así en el tiempo asignado

se pueden atender a más de un subproceso de manera más eficiente, esto es conocido como programación multihilos.

Los hilos realizarán las distintas tareas necesarias en un proceso. Para hacer que los procesos funcionen de esta manera, se utilizan distintas técnicas que le indican kernel cuales son las partes del proceso que pueden ejecutarse simultáneamente y el procesador asignará una fracción de tiempo exclusivo al hilo del tiempo total asignado al proceso.

Los datos pertenecientes al proceso pasan a ser compartidos por los subprocesos lanzados en cada hilo y mediante una técnica de semáforos el kernel mantiene la integridad de estos. Esta técnica de programación puede ser muy eficiente si no se abusa de este recurso, permitiendo un nivel más de paralelización en cada procesador. Esta forma de paralelización no es exclusiva de equipos multiprocesadores o multicomputadoras, ya que pueden ser implementados a nivel de sistema operativo.

8. Implementación Computacional Secuencial y Paralela de DDM

A partir de los modelos matemáticos (capítulo 2 y 3) y los modelos numéricos (capítulos 4, 5, y 6), en este capítulo se describe el modelo computacional contenido en un programa de cómputo orientado a objetos en el lenguaje de programación C++ en su forma secuencial y en su forma paralela en C++ usando la interfaz de paso de mensajes (MPI) bajo el esquema maestro-esclavo (capítulo 7).

Esto no sólo nos ayudará a demostrar que es factible la construcción del propio modelo computacional a partir del modelo matemático y numérico para la solución de problemas reales. Además, se mostrará los alcances y limitaciones en el consumo de los recursos computacionales, evaluando algunas de las variantes de los métodos numéricos con los que es posible implementar el modelo computacional y haremos el análisis de rendimiento sin llegar a ser exhaustivo esté.

También exploraremos los alcances y limitaciones de cada uno de los métodos implementados (FEM, DDM secuencial y paralelo) y como es posible optimizar los recursos computacionales con los que se cuente.

Primeramente hay que destacar que el paradigma de programación orientada a objetos es un método de implementación de programas, organizados como colecciones cooperativas de objetos. Cada objeto representa una instancia de alguna clase y cada clase es miembro de una jerarquía de clases unidas mediante relaciones de herencia, contención, agregación o uso.

Esto nos permite dividir en niveles la semántica de los sistemas complejos tratando así con las partes, que son más manejables que el todo, permitiendo su extensión y un mantenimiento más sencillo. Así, mediante la herencia, contención, agregación o usó nos permite generar clases especializadas que manejan eficientemente la complejidad del problema. La programación orientada a objetos organiza un programa entorno a sus datos (atributos) y a un conjunto de interfases bien definidas para manipular estos datos (métodos dentro de clases reusables) esto en oposición a los demás paradigmas de programación.

El paradigma de programación orientada a objetos sin embargo sacrifica algo de eficiencia computacional por requerir mayor manejo de recursos computacionales al momento de la ejecución. Pero en contraste, permite mayor flexibilidad al adaptar los códigos a nuevas especificaciones. Adicionalmente, disminuye notoriamente el tiempo invertido en el mantenimiento y búsqueda de errores dentro del código. Esto tiene especial interés cuando se piensa en la cantidad de meses invertidos en la programación comparado con los segundos consumidos en la ejecución del mismo.

Para empezar con la implementación computacional, primeramente definiremos el problema a trabajar. Este, pese a su sencillez, no pierde generalidad permitiendo que el modelo mostrado sea usado en muchos sistemas de la ingeniería y la ciencia.

8.1. El Operador de Laplace y la Ecuación de Poisson

Consideramos como modelo matemático el problema de valor en la frontera (BVP) asociado con el operador de Laplace en dos dimensiones, el cual en general es usualmente referido como la ecuación de Poisson, con condiciones de frontera Dirichlet, definido en Ω como:

$$\begin{aligned} -\nabla^2 u &= f_\Omega \text{ en } \Omega \\ u &= g_{\partial\Omega} \text{ en } \partial\Omega. \end{aligned} \tag{294}$$

Se toma está ecuación para facilitar la comprensión de las ideas básicas. Es un ejemplo muy sencillo, pero gobierna los modelos de muchos sistemas de la ingeniería y de la ciencia, entre ellos el flujo de agua subterránea a través de un acuífero isotrópico, homogéneo bajo condiciones de equilibrio y es muy usada en múltiples ramas de la física. Por ejemplo, gobierna la ecuación de la conducción de calor en un sólido bajo condiciones de equilibrio.

En particular consideramos el problema con Ω definido en:

$$\Omega = [-1, 1] \times [0, 1] \tag{295}$$

donde

$$f_\Omega = 2n^2\pi^2 \sin(n\pi x) * \sin(n\pi y) \quad \text{y} \quad g_{\partial\Omega} = 0 \tag{296}$$

cuya solución es

$$u(x, y) = \sin(n\pi x) * \sin(n\pi y). \tag{297}$$

Para las pruebas de rendimiento en las cuales se evalúa el desempeño de los programas realizados se usa $n = 10$, pero es posible hacerlo con $n \in \mathbb{N}$ grande. Por ejemplo para $n = 4$, la solución es $u(x, y) = \sin(4\pi x) * \sin(4\pi y)$, cuya gráfica se muestra a continuación:

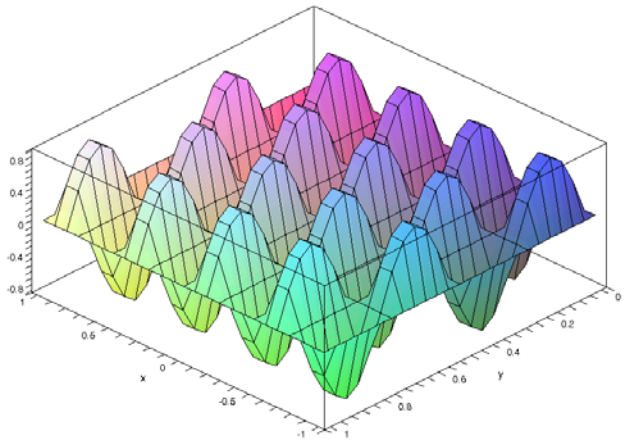

Figura 10: Solución a la ecuación de Poisson para n=4.

Hay que hacer notar que al implementar la solución numérica por el método del elemento finito y el método de subestructuración secuencial en un procesador, un factor limitante para su operación es la cantidad de memoria disponible en la computadora, ya que el sistema algebraico de ecuaciones asociado a esté problema crece muy rápido (del orden de n^2), donde n es el número de nodos en la partición. Es por ello que la elección de un buen manejador de matrices será determinante en la eficiencia alcanzada por las distintas implementaciones de los programas.

Actualmente existen múltiples bibliotecas que permiten manipular operaciones de matrices tanto en forma secuencial como en paralelo (hilos y Pipeline) para implementarlas tanto en procesadores con memoria compartida como distribuida. Pero no están presentes en todas las arquitecturas y sistemas operativos. Por ello en este trabajo se implementaron todas las operaciones necesarias usando clases que fueron desarrolladas sin usar ninguna biblioteca externa a las proporcionadas por el compilador de C++ de GNU, permitiendo la operación de los programas desarrollados en múltiples sistemas operativos del tipo Linux, Unix y Windows.

En cuanto a las pruebas de rendimiento que mostraremos en las siguientes secciones se usaron para la parte secuencial el equipo:

- Computadora Pentium IV HT a 2.8 GHtz con 1 GB de RAM corriendo bajo el sistema operativo Linux Debian Stable con el compilador g++ de GNU.

Para la parte paralela se usaron los equipos siguientes:

- Cluster homogéneo de 10 nodos duales Xeon a 2.8 GHtz con 1 GB de RAM por nodo, unidos mediante una red Ethernet de 1 Gb, corriendo bajo el sistema operativo Linux Debian Stable con el compilador mpiCC de MPI de GNU.
- Cluster heterogéneo con el nodo maestro Pentium IV HT a 3.4 GHtz con 1 GB de RAM y 7 nodos esclavos Pentium IV HT a 2.8 GHtz con 0.5 GB de RAM por nodo, unidos mediante una red Ethernet de 100 Mb, corriendo bajo el sistema operativo Linux Debian Stable con el compilador mpiCC de MPI de GNU.

A estos equipos nos referiremos en lo sucesivo como equipo secuencial, cluster homogéneo y cluster heterogéneo respectivamente.

El tiempo dado en los resultados de las distintas pruebas de rendimiento de los programas y mostrado en todas las tablas y gráficas fue tomado como un promedio entre por lo menos 5 corridas, redondeado el resultado a la unidad siguiente. En todos los cálculos de los métodos numéricos usados para resolver el sistema lineal algebraico asociado se usó una tolerancia mínima de 1×10^{-10}.

Ahora, veremos la implementación del método de elemento finito secuencial para después continuar con el método de descomposición de dominio tanto secuencial como paralelo y poder analizar en cada caso los requerimientos de cómputo, necesarios para correr eficientemente un problema en particular.

8.2. Método del Elemento Finito Secuencial

A partir de la formulación del método de elemento finito visto en la sección (4.2), la implementación computacional que se desarrolló tiene la jerarquía de clases siguiente:

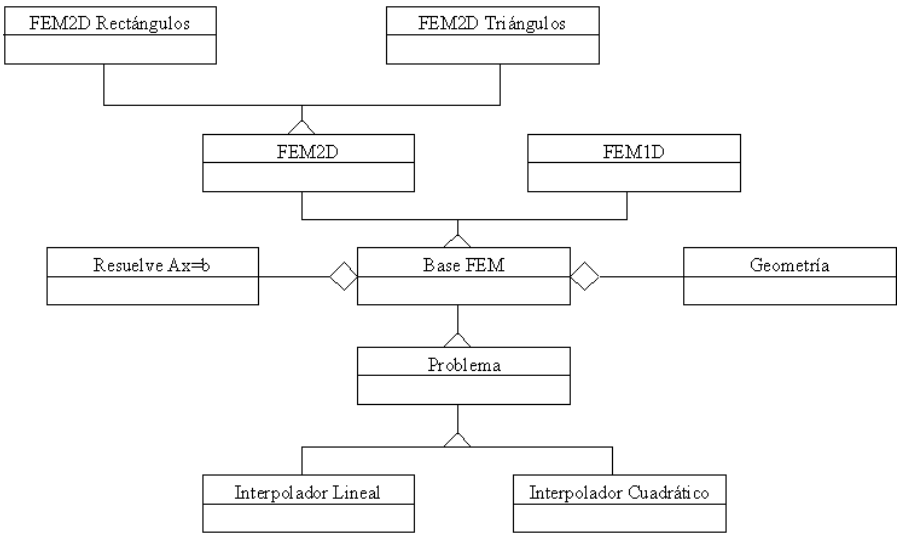

Figura 11: Jerarquía de clases para el método de elemento finito

Donde las clases participantes en *FEM2D Rectángulos* son:

La clase *Interpolador Lineal* define los interpoladores lineales usados por el método de elemento finito.

La clase *Problema* define el problema a tratar, es decir, la ecuación diferencial parcial, valores de frontera y dominio.

La clase *Base FEM* ayuda a definir los nodos al usar la clase *Geometría* y mantiene las matrices generadas por el método y a partir de la clase *Resuelve Ax=B* se dispone de diversas formas de resolver el sistema lineal asociado al método.

La clase *FEM2D* controla lo necesario para poder hacer uso de la geometría en 2D y conocer los nodos interiores y de frontera, con ellos poder montar la matriz de rigidez y ensamblar la solución.

La clase *FEM2D Rectángulos* permite calcular la matriz de rigidez para generar el sistema algebraico de ecuaciones asociado al método.

Notemos que esta misma jerarquía permite trabajar problemas en una y dos dimensiones, en el caso de dos dimensiones podemos discretizar usando

rectángulos o triángulos, así como usar varias opciones para resolver el sistema lineal algebraico asociado a la solución de EDP.

Como ya se menciono, el método de elemento finito es un algoritmo secuencial, por ello se implementa para que use un solo procesador y un factor limitante para su operación es la cantidad de memoria disponible en la computadora, por ejemplo:

Resolver la Ec. (294) con una partición rectangular de 81×81 nodos, genera 6400 elementos rectangulares con 6561 nodos en total, donde 6241 son desconocidos; así el sistema algebraico de ecuaciones asociado a este problema es de dimensión 6241×6241.

Usando el equipo secuencial, primeramente evaluaremos el desempeño del método de elemento finito con los distintos métodos para resolver el sistema algebraico de ecuaciones, encontrando los siguientes resultados:

Método Iterativo	Iteraciones	Tiempo Total
Jacobi	14115	9511 seg.
Gauss-Seidel	7359	4962 seg.
Gradiente Conjugado	91	150 seg.

Como se observa el uso del método de gradiente conjugado es por mucho la mejor elección. En principio, podríamos quedarnos solamente con el método de gradiente conjugado sin hacer uso de precondicionadores por los buenos rendimientos encontrados hasta aquí, pero si se desea resolver un problema con un gran número de nodos, es conocido el aumento de eficiencia al hacer uso de precondicionadores.

Ahora, si tomamos ingenuamente el método de elemento finito conjuntamente con el método de gradiente conjugado con precondicionadores a posteriori (los más sencillos de construir) para resolver el sistema algebraico de ecuaciones, encontraremos los siguientes resultados:

Precondicionador	Iteraciones	Tiempo Total
Jacobi	89	150 seg.
SSOR	85	149 seg.
Factorización Incompleta	79	149 seg.

Como es notorio el uso del método de gradiente conjugado precondicionado con precondicionadores a posteriori no ofrece una ventaja significativa que compense el esfuerzo computacional invertido al crear y usar un precondicionador en los cálculos por el mal condicionamiento del sistema algebraico. Existen también precondicionadores a priori para el método de elemento finito, pero no es costeable en rendimiento su implementación.

Finalmente, para el método de elemento finito las posibles mejoras de eficiencia para disminuir el tiempo de ejecución pueden ser:

- Al momento de compilar los códigos usar directivas de optimización (ofrece mejoras de rendimiento en ejecución de 30 % aproximadamente en las pruebas realizadas).

• Usar la biblioteca Lapack++ de licencia GNU que optimiza las operaciones en el manejo de los elementos de la matriz usando punteros y hacen uso de matrices bandadas (obteniéndose una mejora del rendimiento de 15 % aproximadamente en las pruebas realizadas).

• Combinando las opciones anteriores se obtiene una mejora sustancial de rendimiento en la ejecución (de 45 % aproximadamente en las pruebas realizadas).

Adicionalmente si se cuenta con un equipo con más de un procesador con memoria compartida es posible usar bibliotecas para la manipulación de matrices y vectores que paralelizan o usan Pipeline como una forma de mejorar el rendimiento del programa. Este tipo de mejoras cuando es posible usarlas disminuyen sustancialmente el tiempo de ejecución, ya que en gran medida el consumo total de CPU está en la manipulación de matrices, pero esto no hace paralelo al método de elemento finito.

8.3. Método de Subestructuración Secuencial

A partir de la formulación del método de subestructuración visto en la sección (6.2) se generan las matrices locales $\underline{\underline{A}}_i^{II}, \underline{\underline{A}}_i^{I\Sigma}, \underline{\underline{A}}_i^{\Sigma I}$ y $\underline{\underline{A}}_i^{\Sigma\Sigma}$ y con ellas se construyen $\underline{\underline{S}}_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I}\left(\underline{\underline{A}}_i^{II}\right)^{-1}\underline{\underline{A}}_i^{I\Sigma}$ y $\underline{b}_i = \underline{\underline{A}}_i^{\Sigma I}\left(\underline{\underline{A}}_i^{II}\right)^{-1}\underline{b}_{I_i}$ que son problemas locales a cada subdominio Ω_i, con $i = 1, 2, ..., E$. Generando de manera virtual el sistema lineal $\underline{\underline{S}}\,\underline{u_\Sigma} = \underline{b}$ a partir de

$$\left[\sum_{i=1}^{E} \underline{\underline{S}}_i\right] \underline{u_\Sigma} = \left[\sum_{i=1}^{E} \underline{b}_i\right] \tag{298}$$

donde $\underline{\underline{S}} = \left[\sum_{i=1}^{E} \underline{\underline{S}}_i\right]$ y $\underline{b} = \left[\sum_{i=1}^{E} \underline{b}_i\right]$ podría ser construida sumando las $\underline{\underline{S}}_i$ y $\underline{b}_i$ respectivamente según el orden de los nodos globales versus los nodos locales a cada subdominio.

El sistema lineal virtual resultante

$$\underline{\underline{S}}\,\underline{u_\Sigma} = \underline{b}$$

es resuelto usando el método de gradiente conjugado visto en la sección (5.2), para ello no es necesario construir la matriz $\underline{\underline{S}}$ con las contribuciones de cada S_i correspondientes al subdominio i. Lo que hacemos es pasar a cada subdominio el vector $\underline{u_\Sigma}^i$ correspondiente a la i-ésima iteración del método de gradiente conjugado para que en cada subdominio se evalué $\underline{\tilde{u}_\Sigma}^i = \underline{\underline{S}}_i \underline{u_\Sigma}^i$ localmente y con el resultado se forma el vector $\underline{\tilde{u}_\Sigma} = \sum_{i=1}^{E} \underline{\tilde{u}_\Sigma}^i$ y se continué con los demás pasos del método.

La implementación computacional que se desarrolló tiene una jerarquía de clases en la cual se agregan las clases *FEM2D Rectángulos* y *Geometría*, además de heredar a la clase *Problema*. De esta forma se rehusó todo el código desarrollado para *FEM2D Rectángulos*, la jerarquía queda como:

La clase *DDM2D* realiza la partición gruesa del dominio mediante la clase *Geometría* y controla la partición de cada subdominio mediante un objeto de la clase de *FEM2D Rectángulos* generando la partición fina del dominio. La resolución de los nodos de la frontera interior se hace mediante el método de gradiente conjugado, necesaria para resolver los nodos internos de cada subdominio.

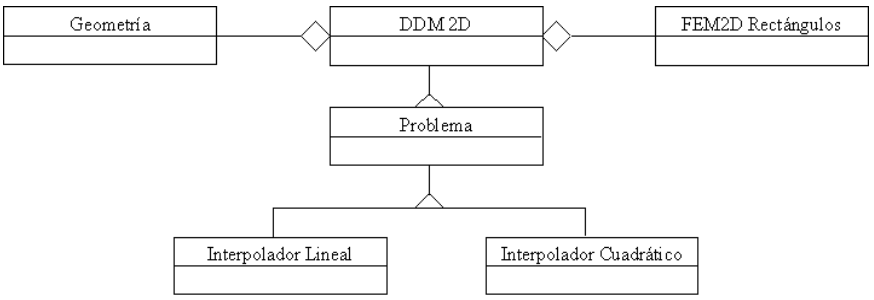

Figura 12: Jerarquía de clases para el método de subestructuración secuencial

Así, el dominio Ω es descompuesto en una descomposición gruesa de $n \times m$ subdominios y cada subdominio Ω_i se parte en $p \times q$ subdominios, generando la participación fina del dominio como se muestra en la figura:

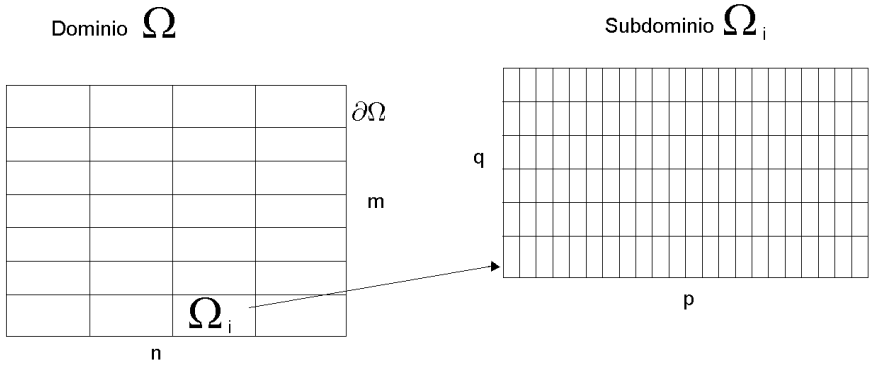

Figura 13: Descomposición del dominio Ω en $E = n \times m$ subdominios y cada subdominio Ω_i en $p \times q$ subdominios

El método de descomposición de dominio se implementó realizando las siguientes tareas:

A) La clase *DDM2D* genera la descomposición gruesa del dominio mediante la agregación de un objeto de la clase *Geometría* (supongamos particionado en $n \times m$ subdominios, generando $s = n * m$ subdominios Ω_i, $i = 1, 2, ..., E$).

B) Con esa geometría se construyen los objetos de *FEM2D Rectángulos* (uno por cada subdominio Ω_i), donde cada subdominio es particionado (supongamos en $p \times q$ subdominios) y regresando las coordenadas de los nodos de frontera del subdominio correspondiente a la clase DDM2D.

C) Con estas coordenadas, la clase *DDM2D* conoce a los nodos de la frontera interior (son estos los que resuelve el método de descomposición de dominio). Las coordenadas de los nodos de la frontera interior se dan a conocer a los objetos *FEM2D Rectángulos*, transmitiendo sólo aquellos que están en su subdominio.

D) Después de conocer los nodos de la frontera interior, cada objeto *FEM2D Rectángulos* calcula las matrices $\underline{\underline{A}}_i^{\Sigma\Sigma}, \underline{\underline{A}}_i^{\Sigma I}, \underline{\underline{A}}_i^{I\Sigma}$ y $\underline{\underline{A}}_i^{II}$ necesarias para construir el complemento de Schur local $\underline{\underline{S}}_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I} \left(\underline{\underline{A}}_i^{II}\right)^{-1} \underline{\underline{A}}_i^{I\Sigma}$ sin realizar comunicación alguna. Al terminar de calcular las matrices se avisa a la clase *DDM2D* de la finalización de los cálculos.

E) Mediante la comunicación de vectores del tamaño del número de nodos de la frontera interior entre la clase *DDM2D* y los objetos *FEM2D Rectángulos*, se prepara todo lo necesario para empezar el método de gradiente conjugado y resolver el sistema lineal virtual $\left[\sum_{i=1}^{E} \underline{\underline{S}}_i\right] \underline{u_\Sigma} = \left[\sum_{i=1}^{E} \underline{b}_i\right]$.

F) Para usar el método de gradiente conjugado, se transmite un vector del tamaño del número de nodos de la frontera interior para que en cada objeto se realicen las operaciones pertinentes y resolver así el sistema algebraico asociado, esta comunicación se realiza de ida y vuelta entre la clase *DDM2D* y los objetos *FEM2D Rectángulos* tantas veces como iteraciones haga el método. Resolviendo con esto los nodos de la frontera interior $\underline{u_{\Sigma_i}}$.

G) Al término de las iteraciones se pasa la solución $\underline{u_{\Sigma_i}}$ de los nodos de la frontera interior que pertenecen a cada subdominio dentro de cada objeto *FEM2D Rectángulos* para que se resuelvan los nodos interiores $\underline{u_{I_i}} = \left(\underline{\underline{A}}_i^{II}\right)^{-1} \left(\underline{b_{I_i}} - \underline{\underline{A}}_i^{\Sigma I} \underline{u_{\Sigma_i}}\right)$, sin realizar comunicación alguna en el proceso, al concluir se avisa a la clase *DDM2D* de ello.

I) La clase *DDM2D* mediante un último mensaje avisa que se concluya el programa, terminado así el esquema maestro-esclavo secuencial.

Por ejemplo, para resolver la Ec. (294), usando 81×81 nodos (igual al ejemplo de *FEM2D Rectángulos* secuencial), podemos tomar alguna de las siguientes descomposiciones:

Descomposición	Nodos Interiores	Subdominios	Elementos Subdominio	Total Nodos Subdominio	Nodos Desconocidos Subdominio
2x2 y 40x40	6084	4	1600	1681	1521
4x4 y 20x20	5776	16	400	441	361
8x8 y 10x10	5184	64	100	121	81
16x16 y 5x5	4096	256	25	36	16

Cada una de las descomposiciones genera un problema distinto. Usando el equipo secuencial y evaluando el desempeño del método de subestructuración secuencial se obtuvieron los siguientes resultados:

Partición	Nodos Frontera Interior	Iteraciones	Tiempo Total
2x2 y 40x40	157	66	1040 seg.
4x4 y 20x20	465	92	68 seg.
8x8 y 10x10	1057	128	23 seg.
16x16 y 5x5	2145	169	87 seg.

Nótese que aún en un solo procesador es posible encontrar una descomposición que desminuya los tiempos de ejecución (la descomposición de 8x8 y 10x10 concluye en 23 seg. versus los 150 seg. en el caso de *FEM2D Rectángulos)*, ello es debido a que al descomponer el dominio en múltiples subdominios, la complejidad del problema es también disminuida y esto se ve reflejado en la disminución del tiempo de ejecución.

Notemos también que en la última descomposición, en lugar de disminuir el tiempo de ejecución este aumenta, esto se debe a que se construyen muchos objetos *FEM2D Rectángulos* (256 en este caso), con los cuales hay que hacer comunicación resultando muy costoso computacionalmente.

Finalmente las posibles mejoras de eficiencia para el método de subestructuración secuencial para disminuir el tiempo de ejecución son las mismas que en el caso del método de elemento finito pero además se tienen que:

- Encontrar la descomposición pertinente entre las posibles descomposiciones que consuma el menor tiempo de cálculo.

Adicionalmente si se cuenta con un equipo con más de un procesador con memoria compartida es posible usar bibliotecas para la manipulación de matrices y vectores que paralelizan o usan Pipeline como una forma de mejorar el rendimiento del programa. Este tipo de mejoras cuando es posible usarlas disminuyen sustancialmente el tiempo de ejecución, ya que en gran medida el consumo total de CPU está en la manipulación de matrices, pero esto no hace paralelo al método de subestructuración secuencial.

8.4. Método de Subestructuración en Paralelo

La computación en paralelo es una técnica que nos permite distribuir una gran carga computacional entre muchos procesadores. Y es bien sabido que una de las mayores dificultades del procesamiento en paralelo es la coordinación de las actividades de los diferentes procesadores y el intercambio de información entre los mismos.

Para hacer una adecuada coordinación de actividades entre los diferentes procesadores, el programa que soporta el método de subestructuración paralelo, usa la misma jerarquía de clases que el método de subestructuración secuencial. Este se desarrolló para usar el esquema maestro-esclavo, de forma tal que el nodo maestro mediante la agregación de un objeto de la clase de *Geometría* genere la descomposición gruesa del dominio y los nodos esclavos creen un conjunto de objetos *FEM2D Rectángulos* para que en estos objetos se genere la participación fina y mediante el paso de mensajes (vía MPI) puedan comunicarse los nodos esclavos con el nodo maestro, realizando las siguientes tareas:

A) El nodo maestro genera la descomposición gruesa del dominio (supongamos particionado en $n \times m$ subdominios) mediante la agregación de un objeto de la clase *Geometría*, esta geometría es pasada a los nodos esclavos.

B) Con esa geometría se construyen los objetos *FEM2D Rectángulos* (uno por cada subdominio), donde cada subdominio es particionado (supongamos en $p \times q$ subdominios). Cada objeto de *FEM2D Rectángulos* genera la geometría solicitada, regresando las coordenadas de los nodos de frontera del subdominio correspondiente al nodo maestro.

C) Con estas coordenadas, el nodo maestro conoce a los nodos de la frontera interior (son estos los que resuelve el método de descomposición de dominio). Las coordenadas de los nodos de la frontera interior se dan a conocer a los objetos *FEM2D Rectángulos* en los nodos esclavos, transmitiendo sólo aquellos que están en su subdominio.

D) Después de conocer los nodos de la frontera interior, cada objeto *FEM2D Rectángulos* calcula las matrices $\underline{\underline{A}}_i^{\Sigma\Sigma}, \underline{\underline{A}}_i^{\Sigma I}, \underline{\underline{A}}_i^{I\Sigma}$ y $\underline{\underline{A}}_i^{II}$ necesarias para construir el complemento de Schur local $\underline{\underline{S}}_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I}\left(\underline{\underline{A}}_i^{II}\right)^{-1}\underline{\underline{A}}_i^{I\Sigma}$ sin realizar comunicación alguna. Al terminar de calcular las matrices se avisa al nodo maestro de la finalización de los cálculos.

E) Mediante la comunicación de vectores del tamaño del número de nodos de la frontera interior entre el nodo maestro y los objetos *FEM2D Rectángulos*, se prepara todo lo necesario para empezar el método de gradiente conjugado y resolver el sistema lineal virtual $\left[\sum_{i=1}^{E} \underline{\underline{S}}_i\right] \underline{u_\Sigma} = \left[\sum_{i=1}^{E} \underline{b}_i\right]$.

F) Para usar el método de gradiente conjugado, se transmite un vector del tamaño del número de nodos de la frontera interior para que en cada objeto se realicen las operaciones pertinentes y resolver así el sistema algebraico asociado, esta comunicación se realiza de ida y vuelta entre el nodo maestro y los objetos *FEM2D Rectángulos* tantas veces como iteraciones haga el método. Resolviendo con esto los nodos de la frontera interior $\underline{u_{\Sigma_i}}$.

G) Al término de las iteraciones se pasa la solución $\underline{u_{\Sigma_i}}$ de los nodos de la frontera interior que pertenecen a cada subdominio dentro de cada objeto *FEM2D Rectángulos* para que se resuelvan los nodos interiores $\underline{u_{I_i}} = \left(\underline{\underline{A}}_i^{II}\right)^{-1}\left(\underline{b_{I_i}} - \underline{\underline{A}}_i^{\Sigma I}\,\underline{u_{\Sigma_i}}\right)$, sin realizar comunicación alguna en el proceso, al concluir se avisa al nodo maestro de ello.

I) El nodo maestro mediante un último mensaje avisa que se concluya el programa, terminado así el esquema maestro-esclavo.

Del algoritmo descrito anteriormente hay que destacar la sincronía entre el nodo maestro y los objetos *FEM2D Rectángulos* contenidos en los nodos esclavos, esto es patente en las actividades realizadas en los incisos A, B y C, estas consumen una parte no significativa del tiempo de cálculo.

Una parte importante del tiempo de cálculo es consumida en la generación de las matrices locales descritas en el inciso D que se realizan de forma independiente en cada nodo esclavo, esta es muy sensible a la discretización particular del dominio usado en el problema.

Los incisos E y F del algoritmo consumen la mayor parte del tiempo total del ejecución al resolver el sistema lineal que dará la solución a los nodos de la frontera interior. La resolución de los nodos interiores planteada en el inciso G consume muy poco tiempo de ejecución, ya que sólo se realiza una serie de cálculos locales previa transmisión del vector que contiene la solución a los nodos de la frontera interior.

Este algoritmo es altamente paralelizable ya que los nodos esclavos están la mayor parte del tiempo ocupados y la fracción serial del algoritmo esta principalmente en las actividades que realiza el nodo maestro, estas nunca podrán ser eliminadas del todo pero consumirán menos tiempo del algoritmo conforme se haga más fina la malla en la descomposición del dominio.

Para resolver la Ec. (294), usando 81×81 subdominios (igual al ejemplo de *FEM2D Rectángulos* secuencial), en la cual se toma una partición rectangular gruesa de 4×4 subdominios y cada subdominio se descompone en 20×20 subdominios.

Usando para los cálculos en un procesador el equipo secuencial y para la parte paralela el cluster homogéneo resolviendo por el método de gradiente conjugado sin precondicionador, la solución se encontró en 92 iteraciones obteniendo los siguientes valores:

Procesadores	Tiempo	Factor de Aceleración	Eficiencia	Fracción Serial
1	68 seg.			
2	73 seg.	0.93150	0.46575	1.14705
3	38 seg.	1.78947	0.59649	0.33823
4	29 seg.	2.34482	0.58620	0.23529
5	22 seg.	3.09090	0.61818	0.15441
6	22 seg.	3.09090	0.51515	0.18823
7	17 seg.	4.00000	0.57142	0.12500
8	17 seg.	4.00000	0.50000	0.14285
9	12 seg.	5.66666	0.62962	0.07352
10	12 seg.	5.66666	0.56666	0.08496
11	12 seg.	5.66666	0.51515	0.09411
12	11 seg.	6.18181	0.51515	0.08556
13	11 seg.	6.18181	0.47552	0.09191
14	11 seg.	6.18181	0.44155	0.09728
15	11 seg.	6.18181	0.41212	0.10189
16	10 seg.	6.80000	0.42500	0.09019
17	6 seg	11.33333	0.66666	0.03125

Estos resultados pueden ser apreciados mejor de manera gráfica como se muestra a continuación:

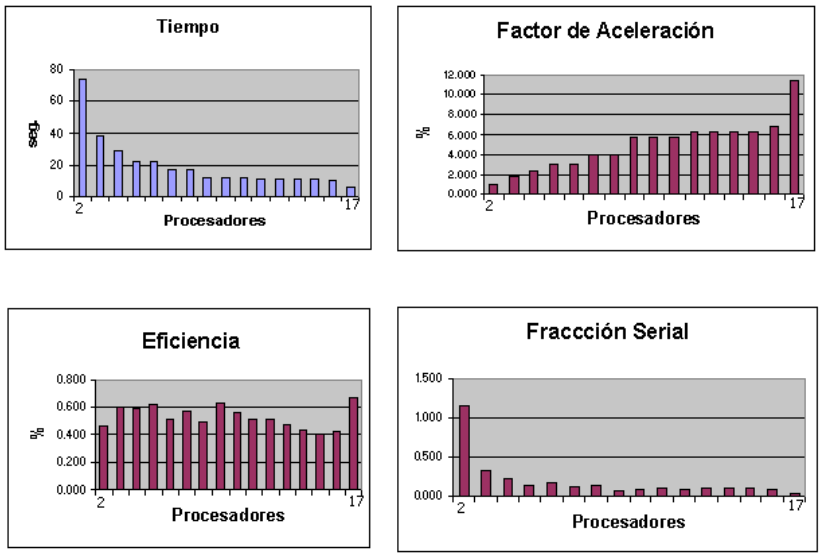

Figura 14: Métricas de desempeño de 2 a 17 procesadores

Primeramente notemos que en el caso de 2 procesadores se obtiene un tiempo de ejecución superior al de un procesador, ello es debido a que el esquema

maestro esclavo en dos procesadores no hace paralelización real de la tarea, pues el nodo maestro sólo hace control de comunicaciones y el nodo esclavo hace todo el trabajo de cálculo (como si fuera proceso serial), resultando en un aumento de tiempo de cómputo por las comunicaciones de red.

Segundo, existe mal balanceo de cargas. La descomposición adecuada del dominio para tener un buen balanceo de cargas se logra cuando se descompone en $n \times m$ nodos en la partición gruesa, generándose $n * m$ subdominios y si se trabaja con P procesadores (1 para el nodo maestro y $P - 1$ para los nodos esclavos), entonces el balance de cargas adecuado será cuando $(P-1) \mid (n*m)$.

En nuestro caso se logra un buen balanceo de cargas cuando se tienen $2, 3, 5, 9, 17$ procesadores, cuyas métricas de desempeño se muestran a continuación:

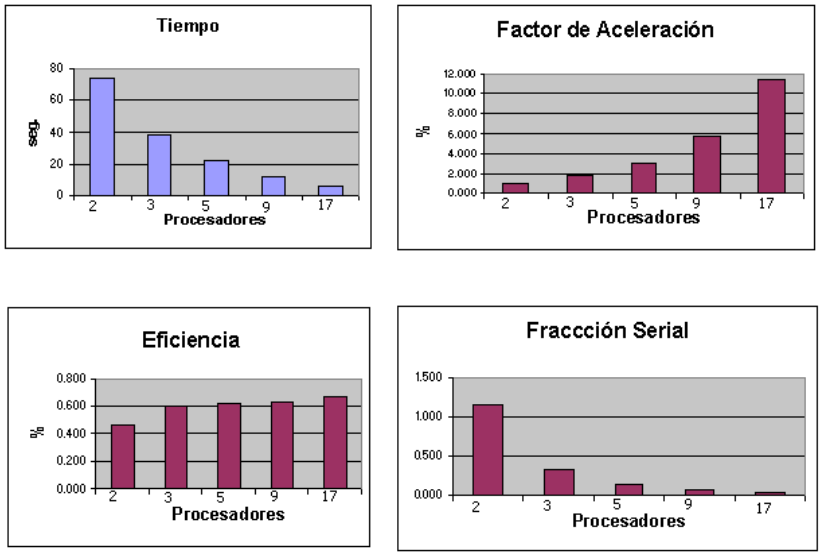

Figura 15: Métricas de desempeño mostrando sólo cuando las cargas están bien balanceadas (2, 3, 5, 9 y 17 procesadores).

En cuanto a las métricas de desempeño, obtenemos que el factor de aceleración en el caso ideal debería de aumentar de forma lineal al aumento del número de procesadores, que en nuestro caso no es lineal pero cumple bien este hecho si están balanceadas las cargas de trabajo.

El valor de la eficiencia deberá ser cercano a uno cuando el hardware es usado de manera eficiente, como es en nuestro caso cuando se tiene un procesador por cada subdominio.

Y en la fracción serial su valor debiera de tender a cero en el caso ideal, siendo este nuestro caso si están balanceadas las cargas de trabajo, de aquí se puede concluir que la granularidad del problema es gruesa, es decir, no existe

una sobrecarga en los procesos de comunicación siendo el cluster una buena herramienta de trabajo para este tipo de problemas.

Finalmente las posibles mejoras de eficiencia para el método de subestructuración en paralelo para disminuir el tiempo de ejecución pueden ser:

- Balanceo de cargas de trabajo homogéneo.
- Al compilar los códigos usar directivas de optimización.
- Usar bibliotecas que optimizan las operaciones en el manejo de los elementos de la matriz usando punteros en las matrices densas o bandadas.
- El cálculo de las matrices que participan en el complemento de Schur pueden ser obtenidas en paralelo.

8.5. Método de Subestructuración en Paralelo Precondicionado

En este método a diferencia del método de subestructuración paralelo, se agrega un precondicionador al método de gradiente conjugado precondicionado visto en la sección (5.3.1) a la hora de resolver el sistema algebraico asociado al método de descomposición de dominio.

En este caso por el mal condicionamiento de la matriz, los precondicionadores a posteriori no ofrecen una ventaja real a la hora de solucionar el sistema lineal algebraico. Es por ello que usaremos los precondicionadores a priori vistos en la sección (6.2.1). Estos son más particulares y su construcción depende del proceso que origina el sistema lineal algebraico.

Existe una amplia gama de este tipo de precondicionadores, pero son específicos al método de descomposición de dominio usado. Para el método de subestructuración usaremos el derivado de la matriz de rigidez, este no es el precondicionador óptimo para este problema, pero para fines demostrativos nos vasta.

La implementación de los métodos a priori, requieren de más trabajo tanto en la face se construcción como en la parte de su aplicación, la gran ventaja de este tipo de precondicionadores es que pueden ser óptimos, es decir, para ese problema en particular el precondicionador encontrado será el mejor precondicionador existente, llegando a disminuir el número de iteraciones hasta en un orden de magnitud.

Por ejemplo, al resolver la Ec. (294) usando 81×81 nodos en la cual se toma una partición rectangular gruesa de 4×4 subdominios y cada subdominio se descompone en 20×20 subdominios.

Usando para el cálculo en un procesador el equipo secuencial y para la parte paralela el cluster homogéneo resolviendo por el método de gradiente conjugado con precondicionador la solución se encontró en 47 iteraciones (una mejora en promedio cercana al 50 % con respecto a no usar precondicionador 92 iteraciones) obteniendo los siguientes valores:

Procesadores	Tiempo	Factor de Aceleración	Eficiencia	Fracción Serial
1	68 seg.			
2	72 seg.	0.94444	0.47222	1.11764
3	37 seg.	1.83783	0.61261	0.31617
4	28 seg.	2.42857	0.60714	0.21568
5	21 seg.	3.23809	0.64761	0.13602
6	21 seg.	3.23809	0.53968	0.17058
7	16 seg.	4.25000	0.60714	0.10784
8	16 seg.	4.25000	0.53125	0.12605
9	11 seg.	6.18181	0.68686	0.05698
10	11 seg.	6.18181	0.61818	0.06862
11	11 seg.	6.18181	0.56198	0.07794
12	10 seg.	6.80000	0.56666	0.06951
13	10 seg.	6.80000	0.52307	0.07598
14	10 seg.	6.80000	0.48571	0.08144
15	10 seg.	6.80000	0.45333	0.08613
16	9 seg.	7.55555	0.47222	0.07450
17	5 seg.	13.60000	0.80000	0.01562

Estos resultados pueden ser apreciados mejor de manera gráfica como se muestra a continuación:

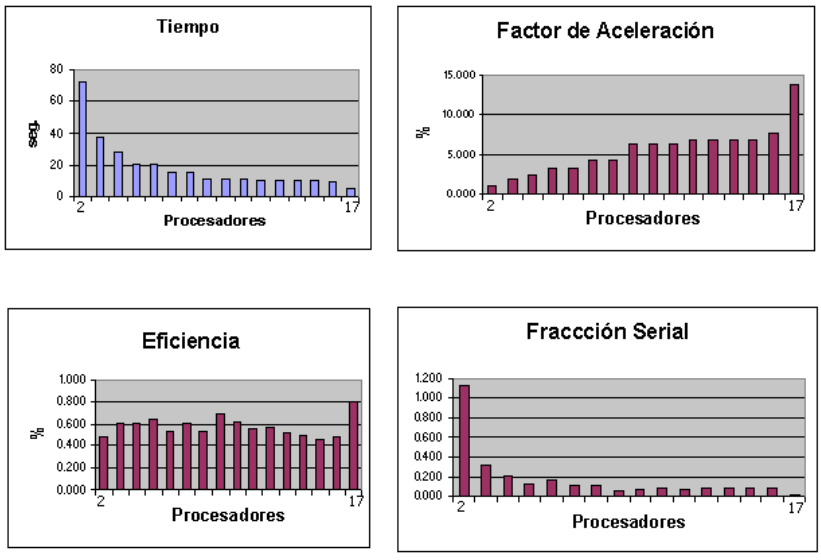

Figura 16: Métricas de desempeño de 2 a 17 procesadores

De las métricas de desempeño, se observa que el factor de aceleración, en el caso ideal debería de aumentar de forma lineal al aumentar el número de

procesadores, que en nuestro caso no es lineal pero cumple bien este hecho si está balanceada las cargas de trabajo.

En la eficiencia su valor deberá ser cercano a uno cuando el hardware es usado de manera eficiente, como es en nuestro caso cuando se tiene un procesador por cada subdominio. Y en la fracción serial su valor debiera de tender a cero en el caso ideal, siendo este nuestro caso si están balanceadas las cargas de trabajo.

En este ejemplo, como en el caso sin precondicionador el mal balanceo de cargas está presente y es cualitativamente igual, para este ejemplo se logra un buen balanceo de cargas cuando se tienen $2, 3, 5, 9, 17$ procesadores, cuyas métricas de desempeño se muestran a continuación:

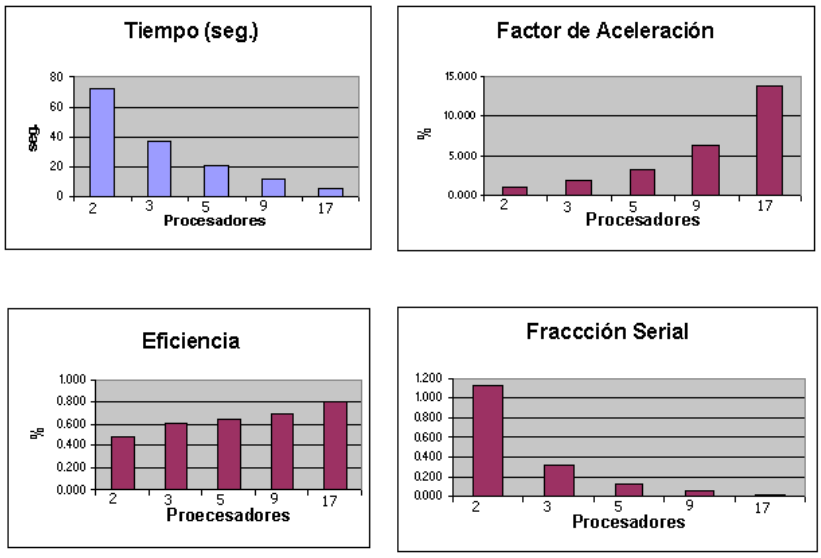

Figura 17: Métricas de desempeño mostrando sólo cuando las cargas están bien balanceadas (2, 3, 5, 9 y 17 procesadores).

Las mismas mejoras de eficiencia que para el método de subestructuración en paralelo son aplicables a este método, adicionalmente:

- Usar el mejor precondicionador a priori disponible para el problema en particular, ya que esto disminuye sustancialmente el número de iteraciones (hasta en un orden de magnitud cuando el precondicionador es óptimo).

9. Análisis de Rendimiento y Conclusiones

Uno de los grandes retos del área de cómputo científico es poder analizar a priori una serie consideraciones dictadas por factores externos al problema de interés que repercuten directamente en la forma de solucionar el problema, estas consideraciones influirán de manera decisiva en la implementación computacional de la solución numérica. Algunas de estas consideraciones son:

- Número de Procesadores Disponibles
- Tamaño y Tipo de Partición del Dominio
- Tiempo de Ejecución Predeterminado

Siendo común que ellas interactúan entre si, de forma tal que normalmente el encargado de la implementación computacional de la solución numérica tiene además de las complicaciones técnicas propias de la solución, el conciliarlas con dichas consideraciones.

Esto deja al implementador de la solución numérica con pocos grados de libertad para hacer de la aplicación computacional una herramienta eficiente y flexible que cumpla con los lineamientos establecidos a priori y permita también que esta sea adaptable a futuros cambios de especificaciones -algo común en ciencia e ingeniería-.

En este capítulo haremos el análisis de los factores que merman el rendimiento de la aplicación y veremos algunas formas de evitarlo, haremos una detallada descripción de las comunicaciones entre el nodo principal y los nodos esclavos, la afectación en el rendimiento al aumentar el número de subdominios en la descomposición y detallaremos algunas consideraciones generales para aumentar el rendimiento computacional. También daremos las conclusiones generales a este trabajo y veremos las diversas ramificaciones que se pueden hacer en trabajos futuros.

9.1. Análisis de Comunicaciones

Para hacer un análisis de las comunicaciones entre el nodo principal y los nodos esclavos en el método de subestructuración es necesario conocer qué se trasmite y su tamaño, es por ello detallaremos en la medida de lo posible las comunicaciones existentes (hay que hacer mención que entre los nodos esclavos no hay comunicación alguna).

Tomando la descripción del algoritmo detallado en el método de subestructuración en paralelo visto en la sección (8.4), suponiendo una partición del dominio Ω en $n \times m$ y $p \times q$ subdominios, las comunicaciones correspondientes a cada inciso son:

A) El nodo maestro transmite 4 coordenadas (en dos dimensiones) correspondientes a la delimitación del subdominio.

B) $2 * p * q$ coordenadas transmite cada objeto *FEM2D Rectángulos* al nodo maestro.

C) A lo más $n * m * 2 * p * q$ coordenadas son las de los nodos de la frontera interior, y sólo aquellas correspondientes a cada subdominio son trasmitidas por el nodo maestro a los objetos *FEM2D Rectángulos* siendo estas a lo más $2 * p * q$ coordenadas.

D) Sólo se envía un aviso de la conclusión del cálculo de las matrices.

E) A lo más $2 * p * q$ coordenadas son transmitidas a los objetos *FEM2D Rectángulos* desde el nodo maestro y los nodos esclavos transmiten al nodo maestro esa misma cantidad información.

F) A lo más $2 * p * q$ coordenadas son transmitidas a los objetos *FEM2D Rectángulos* en los nodos esclavos y estos retornan un número igual al nodo maestro por iteración del método de gradiente conjugado.

G) A lo más $2 * p * q$ valores de la solución de la frontera interior son transmitidas a los objetos *FEM2D Rectángulos* desde el nodo maestro y cada objeto transmite un único aviso de terminación.

I) El nodo maestro manda un aviso a cada objeto *FEM2D Rectángulos* para concluir con el esquema.

La transmisión se realiza mediante paso de arreglos de enteros y números de punto flotante que varían de longitud pero siempre son cantidades pequeñas de estos y se transmiten en forma de bloque, por ello las comunicaciones son eficientes.

9.2. Afectación del Rendimiento al Aumentar el Número de Subdominios en la Descomposición

Una parte fundamental a considerar es la afectación del rendimiento al aumentar el número de subdominios en descomposición, ya que el complemento de Schur $\underline{\underline{S}}_i = \underline{\underline{A}}_i^{\Sigma\Sigma} - \underline{\underline{A}}_i^{\Sigma I} \left(\underline{\underline{A}}_i^{II}\right)^{-1} \underline{\underline{A}}_i^{I\Sigma}$ involucra el generar las matrices $\underline{\underline{A}}_i^{II}, \underline{\underline{A}}_i^{\Sigma\Sigma}, \underline{\underline{A}}_i^{\Sigma I}, \underline{\underline{A}}_i^{I\Sigma}$ y calcular de alguna forma $\left(\underline{\underline{A}}_i^{II}\right)^{-1}$.

Si el número de nodos interiores en el subdominio es grande entonces obtener la matriz anterior será muy costoso computacionalmente, como se ha mostrado en el transcurso de las últimas secciones del capítulo anterior.

Al aumentar el número de subdominios en una descomposición particular, se garantiza que las matrices a generar y calcular sean cada vez más pequeñas y fáciles de manejar.

Pero hay un límite al aumento del número de subdominio en cuanto a la eficiencia de ejecución, este cuello de botella es generado por el esquema maestro-esclavo y es reflejado por un aumento del tiempo de ejecución al aumentar el número de subdominios en una configuración de hardware particular.

Esto se debe a que en el esquema maestro-esclavo, el nodo maestro deberá de atender todas las peticiones hechas por cada uno de los nodos esclavos, esto

toma especial relevancia cuando todos o casi todos los nodos esclavos compiten por ser atendidos por el nodo maestro.

Por ello se recomienda implementar este esquema en un cluster heterogéneo en donde el nodo maestro sea más poderoso computacionalmente que los nodos esclavos. Si a éste esquema se le agrega una red de alta velocidad y de baja latencia, se le permitirá operar al cluster en las mejores condiciones posibles, pero este esquema se verá degradado al aumentar el número de nodos esclavos inexorablemente. Por ello hay que ser cuidadosos en cuanto al número de nodos esclavos que se usan en la implementación en tiempo de ejecución versus el rendimiento general del sistema al aumentar estos.

9.3. Descomposición Óptima para un Equipo Paralelo Dado.

Otra cosa por considerar es que normalmente se tiene a disposición un número fijo de procesadores, con los cuales hay que trabajar, así que es necesario encontrar la descomposición adecuada para esta cantidad de procesadores. No es posible hacer un análisis exhaustivo, pero mediante pruebas podemos determinar cual es la mejor descomposición en base al tiempo de ejecución.

Para el análisis, consideremos pruebas con $3, 4, 5$ y 6 procesadores y veremos cual es la descomposición más adecuada para esta cantidad de procesadores tomando como referencia el resolver la Ec. (294), usando 81×81 nodos.

Usando para estos cálculos el cluster homogéneo, al resolver por el método de gradiente conjugado sin precondicionador para cada descomposición se obtuvieron los siguientes resultados:

Partición	Tiempo en 3 Procesadores	Tiempo en 4 Procesadores	Tiempo en 5 Procesadores	Tiempo en 6 Procesadores
2×2 y 40×40	533 seg.	533 seg.	267 seg.	–
4×4 y 20×20	41 seg.	31 seg.	24 seg.	24 seg.
8×8 y 10×10	40 seg.	42 seg.	41 seg.	42 seg.
16×16 y 5×5	184 seg.	188 seg.	172 seg.	150 seg.

De estas pruebas se observa que el mal balanceo de cargas es reflejado en los tiempos de ejecución, ya que no obstante de contar con más procesadores no hay una disminución del tiempo de ejecución.

Ahora para las mismas descomposiciones, usando el cluster heterogéneo para cada descomposición se obtuvieron los siguientes resultados:

Partición	Tiempo en 3 Procesadores	Tiempo en 4 Procesadores	Tiempo en 5 Procesadores	Tiempo en 6 Procesadores
2×2 y 40×40	558 seg.	533 seg.	255 seg.	–
4×4 y 20×20	36 seg.	25 seg.	21 seg.	17 seg.
8×8 y 10×10	24 seg.	24 seg.	24 seg.	24 seg.
16×16 y 5×5	125 seg.	133 seg.	140 seg.	144 seg.

Primeramente hay que destacar que los nodos esclavos de ambos clusters son comparables en poder de cómputo, pero aquí lo que hace la diferencia es que el nodo maestro del segundo ejemplo tiene mayor rendimiento. Es por ello que al disminuir la fracción serial del problema y atender mejor las comunicaciones que se generan en el esquema maestro-esclavo con todos los objetos *FEM2D Rectángulos* creados en cada uno de los nodos esclavos mejora sustancialmente el tiempo de ejecución.

En ambas pruebas el mal balanceo de cargas es un factor determinante del rendimiento, sin embargo el usó de un cluster en el que el nodo maestro sea más poderoso computacionalmente, hará que se tenga una mejora sustancial en el rendimiento.

Para evitar el mal balanceo de cargas se debe de asignar a cada nodo esclavo una cantidad de subdominios igual. La asignación mínima del número de nodos por subdominio queda sujeta a la velocidad de los procesadores involucrados para disminuir en lo posible los tiempos muertos, obteniendo así el máximo rendimiento.

La asignación máxima del número de nodos por subdominio a cada nodo esclavo, estará en función de la memoria que consuman las matrices que contienen cada uno de los objetos de *FEM2D Rectángulos*. La administración de ellos y las comunicaciones no son un factor limitante y por ello se pueden despreciar.

Descomposición Fina del Dominio Supongamos ahora que deseamos resolver el problema de una descomposición fina del dominio Ω en 8192×8192 nodos, este tipo de problemas surgen cotidianamente en la resolución de sistemas reales y las opciones para implantarlo en un equipo paralelo son viables, existen y son actualmente usadas. Aquí las opciones de partición del dominio son muchas y variadas, y la variante seleccionada dependerán fuertemente de las características del equipo de cómputo paralelo del que se disponga, es decir, si suponemos que una descomposición de 100×100 nodos en un subdominio consume 1 GB de RAM y el consumo de memoria crece linealmente con el número de nodos, entonces algunas posibles descomposiciones son:

Procesadores	Descomposición	Nodos Subdominio	RAM Mínimo
5	2×2 y 4096×4096	4096×4096	≈40.0 GB
257	16×16 y 512×512	512×512	≈5.0 GB
1025	32×32 y 256×256	256×256	≈2.5 GB
4097	64×64 y 128×128	128×128	≈1.2 GB

Notemos que para las primeras particiones, el consumo de RAM es excesivo y en las últimas particiones la cantidad de procesadores en paralelo necesarios es grande (pero ya de uso común en nuestros días). Como en general, contar con equipos paralelos de ese tamaño es en extremo difícil, ¿es posible resolver este tipo de problemas con una cantidad de procesadores menor al número sugerido y donde cada uno de ellos tiene una memoria muy por debajo de lo sugerido?, la respuesta es si.

Primero, notemos que al considerar una descomposición del tipo 64×64 y 128×128 requerimos aproximadamente 1.2 GB de RAM por nodo, si suponemos que sólo tenemos unos cuantos procesadores con memoria limitada (digamos 2 GB), entonces no es posible tener en memoria de manera conjunta a las matrices generadas por el método.

Una de las grandes ventajas de los métodos de descomposición de domino es que los subdominios son en principio independientes entre si y que sólo están acoplados a través de la solución en la interfaz de los subdominios que es desconocida.

Como sólo requerimos tener en memoria la información de la frontera interior, es posible bajar a disco duro todas las matrices y datos complementarios (que consumen el 99 % de la memoria del objeto *FEM2D Rectángulos*) generados por cada subdominio que no se requieran en ese instante para la operación del esquema maestro-esclavo.

Recuperando del disco duro solamente los datos del subdominio a usarse en ese momento (ya que el proceso realizado por el nodo maestro es secuencial) y manteniéndolos en memoria por el tiempo mínimo necesario. Así, es posible resolver un problema de una descomposición fina, usando una cantidad de procesadores fija y con una cantidad de memoria muy limitada por procesador.

En un caso extremo, la implementación para resolver un dominio Ω descompuesto en un número de nodos grande es posible implementarla usando sólo dos procesos en un procesador, uno para el proceso maestro y otro para el proceso esclavo, en donde el proceso esclavo construiría las matrices necesarias por cada subdominio y las guardaría en disco duro, recuperándolas conforme el proceso del nodo maestro lo requiera. Nótese que la descomposición del domino Ω estará sujeta a que cada subdominio Ω_i sea soportado en memoria conjuntamente con los procesos maestro y esclavo.

De esta forma es posible resolver un problema de gran envergadura usando recursos computacionales muy limitados, sacrificando velocidad de procesamiento en aras de poder resolver el problema. Está es una de las grandes ventajas de los métodos de descomposición de dominio con respecto a los otros métodos de discretización tipo diferencias finitas y elemento finito.

El ejemplo anterior nos da una buena idea de las limitantes que existen en la resolución de problemas con dominios que tienen una descomposición fina y nos pone de manifiesto las características mínimas necesarias del equipo paralelo para soportar dicha implantación.

9.4. Consideraciones para Aumentar el Rendimiento

Algunas consideraciones generales para aumentar el rendimiento son:

a) Balanceo de cargas de trabajo homogéneo, si se descompone en $n \times m$ subdominios en la partición gruesa se y si se trabaja con P procesadores, entonces el balance de cargas adecuado será cuando $(P-1) \mid (n*m)$.), ya que de no hacerlo el rendimiento se ve degradado notoriamente.

b) Usar el mejor precondicionador a priori disponible para el problema en particular, ya que esto disminuye sustancialmente el número de iteraciones (hasta en un orden de magnitud cuando el precondicionador es óptimo).

c) Usar la biblioteca Lapack++ de licencia GNU que optimiza las operaciones en el manejo de los elementos de la matriz usando punteros en donde se usan matrices densas y bandadas.

d) Si se cuenta con un equipo en que cada nodo del cluster tenga más de un procesador, usar bibliotecas (PLAPACK por ejemplo) que permitan paralelizar mediante el uso de procesos con memoria compartida, Pipeline o hilos, las operaciones que involucren a vectores y matrices; como una forma de mejorar el rendimiento del programa.

e) Siempre usar al momento de compilar los códigos, directivas de optimización (estas ofrecen mejoras de rendimiento en la ejecución de 30 % aproximadamente en las pruebas realizadas), pero existen algunos compiladores con optimizaciones especificas para ciertos procesadores (Intel compiler para 32 y 64 bits) que pueden mejorar aun más este rendimiento (más de 50 %).

Todas estas mejoras pueden ser mayores si se usa un nodo maestro del mayor poder computacional posible aunado a una red en el cluster de de 1 Gb o mayor y de ser posible de baja latencia, si bien las comunicaciones son pocas, estas pueden generar un cuello de botella sustancial.

Por otro lado, hay que hacer notar que es posible hacer usó de múltiples etapas de paralelización que pueden realizarse al momento de implantar el método de descomposición de dominio, estas se pueden implementar conforme se necesite eficiencia adicional, pero implica una cuidadosa planeación al momento de hacer el análisis y diseño de la aplicación y una inversión cuantiosa en tiempo para implantarse en su totalidad, estas etapas se describen a continuación:

- El propio método de descomposición de dominio ofrece un tipo particular y eficiente de paralelización al permitir dividir el dominio en múltiples subdominios independientes entre si, interconectados sólo por la frontera interior.
- A nivel subdominio otra paralelización es posible, específicamente en el llenado de las matrices involucradas en el método de descomposición de dominio, ya que varias de ellas son independientes.
- A nivel de los cálculos, entre matrices y vectores involucrados en el método también se pueden paralelizar de manera muy eficiente.
- A nivel del compilador es posible generar el ejecutable usando esquemas de paralelización automático y opciones para eficientizar la ejecución.

Por lo anterior es posible usar una serie de estrategias que permitan realizar estas etapas de paralelización de manera cooperativa y aumentar la eficiencia

en una factor muy significativo, pero esto implica una programación particular para cada una de las etapas y poder distribuir las tareas paralelas de cada etapa en uno o más procesadores distintos a los de las otras etapas.

Notemos finalmente que si se toma el programa de elemento finito y se paraleliza usando sólo directivas de compilación, el aumento del rendimiento es notorio pero este se merma rápidamente al aumentar del número de nodos (esto es debido al aumento en las comunicaciones para mantener y acceder a la matriz del sistema algebraico asociado al método). Pero es aun más notorio cuando el método de descomposición de dominio serial usando las mismas directivas de compilación se paraleliza (sin existir merma al aumentar el número de nodos siempre y cuando las matrices generadas estén el la memoria local del procesador).

Esto se debe a que en el método de elemento finito la matriz estará distribuida por todos los nodos usando memoria distribuida, esto es muy costoso en tiempo de cómputo ya que su manipulación requiere de múltiples comunicaciones entre los procesadores, en cambio el el método de descomposición de dominio ya están distribuidas las matrices en los nodos y las operaciones sólo involucran transmisión de un vector entre ellos, minimizando las comunicaciones entre procesadores.

Pero aún estos rendimientos son pobres con respecto a los obtenidos al usar el método de descomposición de dominio paralelizado conjuntamente con bibliotecas para manejo de matrices densas y dispersas en equipos con nodos que cuenten con más de un procesador, en donde mediante el uso de memoria compartida se pueden usar el resto de los procesadores dentro del nodo para efectuar en paralelo las operaciones en donde estén involucradas las matrices.

9.5. Conclusiones Generales

A lo largo del presente trabajo se ha mostrado que al aplicar métodos de descomposición de dominio conjuntamente con métodos de paralelización es posible resolver una gama más amplia de problemas de ciencias e ingeniería que mediante las técnicas tradicionales del tipo diferencias finitas y elemento finito.

La resolución del sistema algebraico asociado es más eficiente cuando se hace uso de precondicionadores a priori conjuntamente con el método de gradiente conjugado precondicionado al implantar la solución por el método de descomposición de dominio.

Y haciendo uso del análisis de rendimiento, es posible encontrar la manera de balancear las cargas de trabajo que son generadas por las múltiples discretizaciones que pueden obtenerse para la resolución de un problema particular, minimizando en la medida de lo posible el tiempo de ejecución y adaptándolo a la arquitectura paralela disponible, esto es especialmente útil cuando el sistema a trabajar es de tamaño considerable.

Adicionalmente se vieron los alcances y limitaciones de esta metodología, permitiendo tener cotas tanto para conocer las diversas descomposiciones que es posible generar para un número de procesadores fijo, como para conocer el número de procesadores necesarios en la resolución de un problema particular.

También se vio una forma de usar los métodos de descomposición de dominio en casos extremos en donde una partición muy fina, genera un problema de gran envergadura y como resolver este usando recursos computacionales muy limitados, sacrificando velocidad de procesamiento en aras de poder resolver el problema.

Así, podemos afirmar de manera categórica que conjuntando los métodos de descomposición de dominio, la programación orientada a objetos y esquemas de paralelización que usan el paso de mensajes, es posible construir aplicaciones que coadyuven a la solución de problemas en dos o más dimensiones concomitantes en ciencia e ingeniería, los cuales pueden ser de tamaño considerable.

Las aplicaciones desarrolladas bajo este paradigma serán eficientes, flexibles y escalables; a la vez que son abiertas a nuevas tecnologías y desarrollos computacionales y al ser implantados en clusters, permiten una codificación ordenada y robusta, dando con ello una alta eficiencia en la adaptación del código a nuevos requerimientos, como en la ejecución del mismo.

De forma tal que esta metodología permite tener a disposición de quien lo requiera, una gama de herramientas flexibles y escalables para coadyuvar de forma eficiente y adaptable a la solución de problemas en medios continuos de forma sistemática.

9.6. Trabajo Futuro

Pese a que en este trabajo sólo se mostraron dos métodos de descomposición de dominio, es posible adaptar la metodología usada en este trabajo a muchos otros métodos de descomposición de dominio (como Trefftz-Herrera, FETI, Multigrid, entre otros) donde cada uno de ellos acepta interpoladores de distintos grados. Permitiendo de esta manera tener un grupo de herramientas que pueden ser usadas en múltiples problemas escogiendo la que ofrezca mayores ventajas computacionales, pero las ventajas y desventajas deberán de ser sopesadas al implementarse en un problema particular.

Además, a cada método de descomposición de dominio se pueden construir diversos precondicionadores a priori, con el objetivo de obtener un balance entre la complejidad de los diversos precondicionadores (aunado al del método de descomposición de dominio) y el aumento del rendimiento, tomando en cuenta que el precondicionador óptimo para un problema particular puede involucrar mucho trabajo de programación.

Finalmente, estos métodos no están constreñidos a problemas elípticos, pueden adaptarse a problemas parabólicos e hiperbólicos, tanto lineales como no lineales, permitiendo así atacar una gran gama de problemas en medios continuos, para más detalles ver [15] y [2].

10. Apéndice

En este apéndice se darán algunas definiciones que se usan a lo largo del presente trabajo, así como se detallan algunos resultados generales de algebra lineal y análisis funcional (en espacios reales) que se anuncian sin demostración pero se indica en cada caso la bibliografía correspondiente donde se encuentran estas y el desarrollo en detalle de cada resultado.

10.1. Nociones de Algebra Lineal

A continuación detallaremos algunos resultados de álgebra lineal, las demostraciones de los siguientes resultados puede ser consultada en [20].

Definición 29 *Sea V un espacio vectorial y sea $f(\cdot) : V \to \mathbb{R}$, f es llamada funcional lineal si satisface la condición*

$$f(\alpha v + \beta w) = \alpha f(v) + \beta f(w) \quad \forall v, w \in V \quad y \quad \alpha, \beta \in \mathbb{R}. \tag{299}$$

Definición 30 *Si V es un espacio vectorial, entonces el conjunto V^* de todas las funcionales lineales definidas sobre V es un espacio vectorial llamado espacio dual de V.*

Teorema 31 *Si $\{v_1, ..., v_n\}$ es una base para el espacio vectorial V, entonces existe una única base $\{v_1^*, ..., v_n^*\}$ del espacio vectorial dual V^* llamado la base dual de $\{v_1, ..., v_n\}$ con la propiedad de que $V_i^* = \delta_{ij}$. Por lo tanto V es isomorfo a V^*.*

Definición 32 *Sea $D \subset V$ un subconjunto del espacio vectorial V. El nulo de D es el conjunto $N(D)$ de todas las funcionales en V^* tal que se nulifican en todo el subconjunto D, es decir*

$$N(D) = \{f \in V^* \mid f(v) = 0 \quad \forall v \in D\}. \tag{300}$$

Teorema 33 *Sea V un espacio vectorial y V^* el espacio dual de V, entonces*

a) $N(D)$ es un subespacio de V^*

b) Si $M \subset V$ es un subespacio de dimensión m, V tiene dimensión n, entonces $N(M)$ tiene dimensión $n - m$ en V^*.

Corolario 34 *Si $V = L \oplus M$ (suma directa) entonces $V^* = N(L) \oplus N(M)$.*

Teorema 35 *Sean V y W espacios lineales, si $T(\cdot) : V \to W$ es lineal, entonces el adjunto T^* de T es un operador lineal $T^* : W^* \to V^*$ definido por*

$$T^*(w^*)(u) = w^*(Tu). \tag{301}$$

Teorema 36 *Si H es un espacio completo con producto interior, entonces $H^* = H$.*

Definición 37 *Si V es un espacio vectorial con producto interior y $T(\cdot): V \to V$ es una transformación lineal, entonces existe una transformación asociada a T llamada la transformación auto-adjunta T^* definida como*

$$\langle Tu, v\rangle = \langle u, T^* v\rangle . \tag{302}$$

Definición 38 *Sea V un espacio vectorial sobre los reales. Se dice que una función $\tau(\cdot,\cdot): V \times V \to \mathbb{R}$ es una forma bilineal sobre V, si para toda $x, y, z \in V$ y $\alpha, \beta \in \mathbb{R}$ se tiene*

$$\begin{aligned} \tau(\alpha x + \beta y, z) &= \alpha\tau(x, z) + \beta\tau(y, z) \\ \tau(x, \alpha y + \beta z) &= \alpha\tau(x, y) + \beta\tau(x, z) . \end{aligned} \tag{303}$$

Definición 39 *Si $\tau(\cdot,\cdot)$ es una forma bilineal sobre V, entonces la función $q_\tau(\cdot): V \to \mathbb{R}$ definida por*

$$q_\tau(x) = \tau(x, x) \quad \forall x \in V \tag{304}$$

se le llama la forma cuadrática asociada a τ.

Notemos que para una forma cuadrática $q_\tau(\cdot)$ se tiene que $q_\tau(\alpha x) = |\alpha|^2 q_\tau(x)$ $\forall x \in V$ y $\alpha \in \mathbb{R}$.

10.2. σ-Algebra y Espacios Medibles

A continuación detallaremos algunos resultados conjuntos de espacios σ-algebra, conjuntos de medida cero y funciones medibles, las demostraciones de los siguientes resultados puede ser consultada en [22] y [3].

Definición 40 *Una σ-algebra sobre un conjunto Ω es una familia ξ de subconjuntos de Ω que satisface*

- $\varnothing \in \xi$
- *Si* $\psi_n \in \xi$ *entonces* $\bigcup_{n=1}^{\infty} \psi_n \in \xi$
- *Si* $\psi \in \xi$ *entonces* $\psi^c \in \xi$.

Definición 41 *Si Ω es un espacio topológico, la familia de Borel es el conjunto σ-algebra más pequeño que contiene a los abiertos del conjunto Ω.*

Definición 42 *Una medida μ sobre Ω es una función no negativa real valuada cuyo dominio es una σ-algebra ξ sobre Ω que satisface*

- $\mu(\varnothing) = 0$ *y*
- *Si* $\{\psi_n\}$ *es una sucesión de conjuntos ajenos de* ξ *entonces*

$$\mu\left(\bigcup_{n=1}^{\infty} \psi_n\right) = \sum_{n=1}^{\infty} \mu(\psi_n) . \tag{305}$$

Teorema 43 *Existe una función de medida μ sobre el conjunto de Borel de $\mathbb{R}$ llamada la medida de Lebesge que satisface $\mu([a,b]) = b-a$.*

Definición 44 *Una función $f : \Omega \to \mathbb{R}$ es llamada medible si $f^{-1}(U)$ es un conjunto medible para todo abierto U de Ω.*

Definición 45 *Sea $E \subset \Omega$ un conjunto, se dice que el conjunto E tiene medida cero si $\mu(E) = 0$.*

Teorema 46 *Si α es una medida sobre el espacio X y β es una medida sobre el espacio Y, podemos definir una medida μ sobre $X \times Y$ con la propiedad de que $\mu(A \times B) = \alpha(A)\beta(B)$ para todo conjunto medible $A \in X$ y $B \in Y$.*

Teorema 47 *(Fubini)*
Si $f(x,y)$ es medible en $X \times Y$ entonces

$$\int_{X\times Y} f(x,y)\, d\mu = \int_X \int_Y f(x,y)\, d\beta d\alpha = \int_Y \int_X f(x,y)\, d\alpha d\beta \tag{306}$$

en el sentido de que cualquiera de las integrales existe y son iguales.

Teorema 48 *Una función f es integrable en el sentido de Riemann en Ω si y sólo si el conjunto de puntos donde $f(\underline{x})$ es no continua tiene medida cero.*

Observación 49 *Sean f y g dos funciones definidas en Ω, decimos que f y g son iguales salvo en un conjunto de medida cero si $f(x) \neq g(x)$ sólo en un conjunto de medida cero.*

Definición 50 *Una propiedad P se dice que se satisface en casi todos lados, si existe un conjunto E con $\mu(E) = 0$ tal que la propiedad se satisface en todo punto de E^c.*

10.3. Espacios L^p

Las definiciones y material adicional puede ser consultada en [12], [18] y [3].

Definición 51 *Una función medible $f(\cdot)$ (en el sentido de Lebesgue) es llamada integrable sobre un conjunto medible $\Omega \subset \mathbb{R}^n$ si*

$$\int_\Omega |f|\, d\underline{x} < \infty. \tag{307}$$

Definición 52 *Sea p un número real con $p \geq 1$. Una función $u(\cdot)$ definida sobre $\Omega \subset \mathbb{R}^n$ se dice que pertenece al espacio $L^p(\Omega)$ si*

$$\int_\Omega |u(\underline{x})|^p\, d\underline{x} \tag{308}$$

es integrable.

Al espacio $L^2(\Omega)$ se le llama cuadrado integrable.

Definición 53 *La norma $L^2(\Omega)$ se define como*

$$\|u\|_{L^2(\Omega)} = \left(\int_\Omega |u(\underline{x})|^2 \, d\underline{x} \right)^{\frac{1}{2}} < \infty \tag{309}$$

y el producto interior en la norma $L^2(\Omega)$ como

$$\langle u, v \rangle_{L^2(\Omega)} = \int_\Omega u(\underline{x}) v(\underline{x}) d\underline{x}. \tag{310}$$

Definición 54 *Si $p \to \infty$, entonces definimos al espacio $L^\infty(\Omega)$ como el espacio de todas las funciones medibles sobre $\Omega \subset \mathbb{R}^n$ que sean acotadas en casi todo Ω (excepto posiblemente sobre un conjunto de medida cero), es decir,*

$$L^\infty(\Omega) = \{u \mid |u(x)| \leq k\} \tag{311}$$

definida en casi todo Ω, para algún $k \in \mathbb{R}$.

10.4. Distribuciones

La teoría de distribuciones es la base para definir a los espacios de Sobolev, ya que permiten definir las derivadas parciales de funciones no continuas, pero esta es coincidente con las derivadas parciales clásica si las funciones son continuas, para mayor referencia de estos resultados ver [12], [18] y [3]

Definición 55 *Sea $\Omega \subset \mathbb{R}^n$ un dominio, al conjunto de todas las funciones continuas definidas en Ω se denotarán por $C^0(\Omega)$, o simplemente $C(\Omega)$.*

Definición 56 *Sea u una función definida sobre un dominio Ω la cual es no cero solo en los puntos pertenecientes a un subconjunto propio $K \subset \Omega$. Sea $\overline{K}$ la clausura de K. Entonces $\overline{K}$ es llamado el soporte de u. Decimos que u tiene soporte compacto sobre Ω si su soporte $\overline{K}$ es compacto. Al conjunto de funciones continuas con soporte compacto se denota por $C_0(\Omega)$.*

Definición 57 *Sea $C^m(\Omega)$ el conjunto de todas las funciones $D^\alpha u$ tales que sean funciones continuas con $|\alpha| = m$. Y $C^\infty(\Omega)$ como el espacio de funciones en el cual todas las derivadas existan y sean continuas en Ω.*

Definición 58 *El espacio $\mathcal{D}(\Omega)$ será el subconjunto de funciones infinitamente diferenciales con soporte compacto, algunas veces se denota también como $C_0^\infty(\Omega)$.*

Definición 59 *Sea $\{\phi_n\}$ una sucesión de funciones en $\mathcal{D}(\Omega)$. Entonces la sucesión es llamada convergente a $\phi \in \mathcal{D}(\Omega)$ si*

- *Existe un conjunto compacto K en Ω que contiene el soporte de todas las ϕ_n; y*

- *La sucesión* $\{D^{\alpha}\phi_n\}$ *converge uniformemente en* k *a* $D^{\alpha}\phi$ *para todo* α.

Decimos que la función $f(\cdot)$ definida en $\mathcal{D}(\Omega)$ es continua sobre $\mathcal{D}(\Omega)$, si para cada sucesión convergente $\{\phi_n\}$ en $\mathcal{D}(\Omega)$, con límite ϕ,

$$\langle f, \phi_n \rangle_{L^2(\Omega)} \to \langle f, \phi \rangle_{L^2(\Omega)} \tag{312}$$

cuando $n \to \infty$.

Definición 60 *Una distribución sobre un dominio* $\Omega \subset \mathbb{R}^n$ *es toda funcional lineal continua sobre* $\mathcal{D}(\Omega)$.

Definición 61 *El espacio de distribuciones es el espacio de funcionales continuas (dual) de* $\mathcal{D}(\Omega)$ *denotado como* $\mathcal{D}^*(\Omega)$.

Definición 62 *Un función* $f(\cdot)$ *es llamada localmente integrable, si para todo subconjunto compacto* $K \subset \Omega$ *se tiene*

$$\int_K |f(x)|\, dx < \infty. \tag{313}$$

Ejemplo de una distribución es cualquier función $f(\cdot)$ localmente integrable en Ω. La distribución F asociada a f se puede definir de manera natural como $F : \mathcal{D}(\Omega) \to \mathbb{R}$ como

$$\langle F, \phi \rangle = \int_\Omega f\phi dx \tag{314}$$

con $\phi \in \mathcal{D}(\Omega)$.

Si el soporte de ϕ es $K \subset \Omega$, entonces

$$|\langle F, \phi \rangle| = \left| \int_\Omega f\phi dx \right| = \left| \int_K f\phi dx \right| \leq \sup_{x \in K} |\phi| \int_\Omega |f(x)|\, dx \tag{315}$$

la integral es finita y $\langle F, \phi \rangle$ tiene sentido. Bajo estas circunstancias F es llamada una distribución generada por f.

Definición 63 *Si una distribución es generada por funciones localmente integrables es llamada una distribución regular. Si una distribución no es generada por una función localmente integrable, es llamada distribución singular.*

Es posible definir de manera natural en producto de una función y una distribución. Específicamente, si $\Omega \subset \mathbb{R}^n$, u perteneciente a $C^{\infty}(\Omega)$, y si $f(\cdot)$ es una distribución sobre Ω, entonces la distribución satisface

$$\langle (uf), \phi \rangle = \langle f, u\phi \rangle \tag{316}$$

para toda $\phi \in \mathcal{D}(\Omega)$. Notemos que la anterior ecuación es una generalización de la identidad

$$\int_\Omega [u(x) f(x)] \phi(x)\, dx = \int_\Omega f(x) [u(x) \phi(x)]\, dx \tag{317}$$

la cual se satisface si f es localmente integrable.

Derivadas de Distribuciones

Teorema 64 *La versión clásica del teorema de Green es dada por la identidad*

$$\int_{\Omega} u\frac{\partial v}{\partial x_i}d\underline{x} = \int_{\partial\Omega} uvn_i d\underline{s} - \int_{\Omega} v\frac{\partial u}{\partial x_i}d\underline{x} \tag{318}$$

que se satisface para todas las funciones u, v en $C^1(\overline{\Omega})$, donde n_i es la i-ésima componente de la derivada normal del vector v en la frontera $\partial\Omega$ de un dominio Ω.

Una versión de la Ec. (318) en una dimensión se obtiene usando la formula de integración por partes, quedando como

$$\int_a^b uv'dx = [uv]\,|_a^b - \int_a^b vu'dx, \quad u, v \in C^1[a,b] \tag{319}$$

como un caso particular de la Ec. (318).

Este resultado es fácilmente generalizable a un resultado usando derivadas parciales de orden m de funciones $u, v \in C^m(\overline{\Omega})$ pero remplazamos u por $D^\alpha u$ en la Ec. (318) y con $|\alpha| = m$, entonces se puede mostrar que:

Teorema 65 *Otra versión del teorema de Green es dado por*

$$\int_{\Omega} (D^\alpha u)\, v d\underline{x} = (-1)^{|\alpha|}\int_{\Omega} uD^\alpha v d\underline{x} + \int_{\partial\Omega} h(u,v)d\underline{s} \tag{320}$$

donde $h(u,v)$ es una expresión que contiene la suma de productos de derivadas de u y v de orden menor que m.

Ahora remplazando v en la Ec. (320) por ϕ perteneciente a $\mathcal{D}(\Omega)$ y como $\phi = 0$ en la frontera $\partial\Omega$ tenemos

$$\int_{\Omega} (D^\alpha u)\,\phi d\underline{x} = (-1)^{|\alpha|}\int_{\Omega} uD^\alpha \phi d\underline{x} \tag{321}$$

ya que u es m-veces continuamente diferenciable, esta genera una distribución denotada por u, tal que

$$\langle u, \phi\rangle = \int_{\Omega} u\phi d\underline{x} \tag{322}$$

o, como $D^\alpha\phi$ también pertenece a $\mathcal{D}(\Omega)$, entonces

$$\langle u, D^\alpha\phi\rangle = \int_{\Omega} uD^\alpha\phi d\underline{x} \tag{323}$$

además, $D^\alpha u$ es continua, así que es posible generar una distribución denotada por $D^\alpha u$ satisfaciendo

$$\langle D^\alpha u, \phi\rangle = \int_{\Omega} (D^\alpha u)\,\phi d\underline{x} \tag{324}$$

entonces la Ec. (321) puede reescribirse como

$$\langle D^\alpha u, \phi\rangle = (-1)^{|\alpha|}\langle u, D^\alpha\phi\rangle, \quad \forall\phi \in D(\Omega). \tag{325}$$

Definición 66 *La derivada de cualquier distribución $f(\cdot)$ se define como: La $\alpha-$ésima distribucional o derivada parcial generalizada de una distribución f es definida por una distribución denotada por $D^{\alpha}f$, que satisface*

$$\langle D^{\alpha}f, \phi\rangle = (-1)^{|\alpha|}\langle f, D^{\alpha}\phi\rangle, \quad \forall\phi \in \mathcal{D}(\Omega).$$

Nótese que si f pertenece a $C^{m}\left(\bar{\Omega}\right)$, entonces la derivada generalizada coincide con la derivada parcial $\alpha-$ésima para $|\alpha| \leq m$.

Derivadas Débiles Supóngase que una función $u(\cdot)$ es localmente integrable que genere una distribución, también denotada por u, que satisface

$$\langle u, \phi\rangle = \int_{\Omega} u\phi dx \tag{326}$$

para toda $\phi \in \mathcal{D}(\Omega)$.

Además la distribución u posee derivada distribucional de todos los ordenes, en particular la derivada $D^{\alpha}u$ es definida por

$$\langle D^{\alpha}u, \phi\rangle = (-1)^{|\alpha|}\langle u, D^{\alpha}\phi\rangle, \quad \forall\phi \in D(\Omega). \tag{327}$$

por supuesto $D^{\alpha}u$ puede o no ser una distribución regular. Si es una distribución regular, entonces es generada por una función localmente integrable tal que

$$\langle D^{\alpha}u, \phi\rangle = \int_{\Omega} D^{\alpha}u(x)\,\phi(x)\,d\underline{x} \tag{328}$$

y se sigue que la función u y $D^{\alpha}u$ están relacionadas por

$$\int_{\Omega} D^{\alpha}u(x)\,\phi(x)\,d\underline{x} = (-1)^{|m|}\int_{\Omega} u(x)\,D^{\alpha}\phi(x)\,d\underline{x} \tag{329}$$

para $|\alpha| \leq m$.

Definición 67 *Llamamos a la función (o más precisamente, a la equivalencia de clases de funciones) $D^{\alpha}u$ obtenida en la Ec. (329), la $\alpha-$ésima derivada débil de la función u.*

Notemos que si u pertenece a $C^{m}\left(\bar{\Omega}\right)$, entonces la derivada $D^{\alpha}u$ coincide con la derivada clásica para $|\alpha| \leq m$.

11. Bibliografía

Referencias

[1] A. Quarteroni, A. Valli; *Domain Decomposition Methods for Partial Differential Equations*. Clarendon Press Oxford 1999.

[2] A. Toselli, O. Widlund; *Domain Decomposition Methods - Algorithms and Theory*. Springer, 2005.

[3] B. D. Reddy; *Introductory Functional Analysis - With Applications to Boundary Value Problems and Finite Elements*. Springer 1991.

[4] B. F. Smith, P. E. Bjørstad, W. D. Gropp; *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.

[5] B. I. Wohlmuth; *Discretization Methods and Iterative Solvers Based on Domain Decomposition*. Springer, 2003.

[6] I. Foster; *Designing and Building Parallel Programs*. Addison-Wesley Inc., Argonne National Laboratory, and the NSF, 2004.

[7] G. Herrera; Análisis de Alternativas al Método de Gradiente Conjugado para Matrices no Simétricas. Tesis de Licenciatura, Facultad de Ciencias, UNAM, 1989.

[8] I. Herrera, M. Díaz; *Modelación Matemática de Sistemas Terrestres* (Notas de Curso en Preparación). Instituto de Geofísica, (UNAM).

[9] I. Herrera;*Un Análisis del Método de Gradiente Conjugado*. Comunicaciones Técnicas del Instituto de Geofísica, UNAM; Serie Investigación, No. 7, 1988.

[10] I. Herrera; *Método de Subestructuración* (Notas de Curso en Preparación). Instituto de Geofísica, (UNAM).

[11] J. II. Bramble, J. E. Pasciak and A. II Schatz. *The Construction of Preconditioners for Elliptic Problems by Substructuring*. I. Math. Comput., 47, 103-134,1986.

[12] J. L. Lions & E. Magenes; *Non-Homogeneous Bonduary Value Problems and Applications Vol. I*, Springer-Verlag Berlin Heidelber New York 1972.

[13] K. Hutter & K. Jöhnk; *Continuum Methods of Physical Modeling.* Springer-Verlag Berlin Heidelber New York 2004.

[14] L. F. Pavarino, A. Toselli; *Recent Developments in Domain Decomposition Methods*. Springer, 2003.

[15] M.B. Allen III, I. Herrera & G. F. Pinder; *Numerical Modeling in Science And Engineering.* John Wiley & Sons, Inc . 1988.

[16] M. Diaz; *Desarrollo del Método de Colocación Trefftz-Herrera Aplicación a Problemas de Transporte en las Geociencias.* Tesis Doctoral, Instituto de Geofísica, UNAM, 2001.

[17] M. Diaz, I. Herrera; *Desarrollo de Precondicionadores para los Procedimientos de Descomposición de Dominio.* Unidad Teórica C, Posgrado de Ciencias de la Tierra, 22 pags, 1997.

[18] P.G. Ciarlet, J. L. Lions; *Handbook of Numerical Analysis, Vol. II.* North-Holland, 1991.

[19] R. L. Burden y J. D. Faires; *Análisis Numérico.* Math Learning, 7 ed. 2004.

[20] S. Friedberg, A. Insel, and L. Spence; *Linear Algebra*, 4th Edition, Prentice Hall, Inc. 2003.

[21] W. Gropp, E. Lusk, A. Skjellem, *Using MPI, Portable Parallel Programming Whit the Message Passing Interface.* Scientific and Engineering Computation Series, 2ed, 1999.

[22] W. Rudin; *Principles of Mathematical Analysis.* McGraw-Hill International Editions, 1976.

[23] X. O. Olivella, C. A. de Sacribar; *Mecánica de Medios Continuos para Ingenieros.* Ediciones UPC, 2000.

[24] Y. Saad; *Iterative Methods for Sparse Linear Systems.* SIAM, 2 ed. 2000.

[25] Y. Skiba; *Métodos y Esquemas Numéricos, un Análisis Computacional.* UNAM, 2005.