Simulación Numérica de Flujo de Fluidos en 2D utilizando el Método de Elementos Finitos Mínimo Cuadrado

Sergio Velásquez Zeballos

Valencia, 10 de Noviembre de 2004

UNIVERSIDAD DE CARABOBO
Facultad de Ciencias y Tecnología
Departamento de Computación

Simulación Numérica de Flujo de Fluidos en 2D utilizando el Método de Elementos Finitos Mínimo Cuadrado

Autor: Sergio Velásquez Zeballos
Tutor: Carlos Cadenas

Trabajo Especial de Grado presentado ante la Universidad de Carabobo como credencial de mérito para optar al título de Licenciado en Computación.

http://eter.sourceforge.net/svelasqu/thesis
svelasqu@uc.edu.ve

Valencia, 10 de Noviembre de 2004

Resumen

La mejora de los métodos de aproximación discreta de ecuaciones diferenciales a través de los años ha incrementado la posibilidad de realizar herramientas computacionales que produzcan muy eficientemente resultados óptimos en campos como la simulación de flujo de fluidos. En este trabajo se estudiará el método de elementos finitos mínimos cuadrados (LSFEM) aplicándolo a las ecuaciones de Navier-Stokes y compresibilidad artificial para simular flujo de fluidos a través de cuerpos poligonales en 2D. La implementación de este método se basará en una nueva formulación que involucra la incorporación de las condiciones de frontera en la formas variacionales involucradas en el mismo.

♡♡♡ *...a Áurea, el amor de mi vida, mi todo...* ♡♡♡

—:: ĵìýî í ĵɒ̄ ɋɒ ȋp̄ ì íɣ ::—

Sergio Velásquez Zeballos

Agradecimiento

Agradezco con falsa humildad a la Vida, a la Muerte y a todos los que las disfrutan o les huyen $\tilde{\smile}$.

Sergio Velásquez Zeballos

Índice general

Índice de algoritmos y cuadros

Índice de figuras

Introducción

La creación y crecimiento de los métodos numéricos ha hallado su apoyo en diversas aplicaciones modernas que varían desde tareas cotidianas o procesos naturales comunes hasta el estudio de tecnología de punta. Uno de esos métodos es el de Elementos Finitos Mínimos Cuadrados, que se presenta como una buena opción para resolver ecuaciones diferenciales aplicadas a cualquier área. Este método es relativamente antiguo, sin embargo es desde hace poco que su uso se ha extendido y su efectividad ha crecido conjunto a la atención que llama a los investigadores modernos. Ahora nuevas técnicas se implementan para resolver problemas reduciendo al máximo el tiempo y los costos.

El área de flujo de fluidos es una de las aplicaciones más estudiadas e importantes en la investigación científica y tecnológica de éstos métodos numéricos. Esto se debe a la amplia gama de importantes campos en los que es muy útil tener un conocimiento apropiado de la mecánica de fluidos: En biomecánica el flujo de sangre y fluido cerebral son de particular interés; en meteorología e ingeniería oceánica se necesita entender el movimiento del aire y las corrientes oceánicas; los ingenieros químicos deben saber diseñar los diferentes equipos de procesamiento químico; los ingenieros aeronáuticos utilizan este conocimiento para incrementar al máximo la fuerza de elevación, reducir al mínimo el retardo de aeronaves y para diseñar motores de reacción; los ingenieros mecánicos diseñan bombas, turbinas, motores de combustión interna, compresores de aire, equipos de aire acondicionado, para el control de contaminación y plantas eléctricas con base en el conocimiento apropiado de la mecánica de fluidos; los ingenieros civiles también utilizan los resultados obtenidos en el estudio de mecánica de fluidos para comprender el transporte de sedimentos y la erosión en ríos, la contaminación del aire y agua, y así diseñar sistemas

de tuberías, plantas de tratamiento de aguas negras, canales de irrigación, sistemas de control de inundaciones, presas y estadios deportivos cubiertos. En refinerías petrolíferas también se encuentra extremadamente importante el poder prever la actuación del fluido en diversos casos, al igual que en empresas públicas o privadas de alto peso, como la *NASA*, en donde la creación de nueva tecnología depende de conocimiento en fluidos aplicado.

Surge entonces la simulación computacional de la dinámica de fluidos (*CFD*, por sus siglas en inglés) como una herramienta para la evaluación e investigación de situaciones como estas sin necesidad de realizarlas, lo que ha llevado a muchos investigadores dedicarse a resolver los problemas de orden numérico para cumplir con estos objetivos, logrando así reducir costos en tiempo, dinero y esfuerzo.

En éste trabajo se intentarán aproximar los problemas del flujo de fluidos utilizando una nueva formulación del Método de Elementos Finitos Mínimos Cuadrados.

En el primer capítulo (1) se dará a conocer el problema a resolver, los objetivos que se persiguen, la justificación y los antecedentes en las áreas involucradas.

En el capítulo 2 se introducirán algunos conceptos básicos de la dinámica de fluidos computacional, las ecuaciones diferenciales y los fundamentos teóricos que se aplican a ellas. A continuación sigue una descripción de los métodos a manera de introducción, así como cada tema que sea de interés para conocer el proceso que ocurre al resolverse un problema de este calibre y que se implementará para ello.

En el capítulo 3 se explicará la metodología a utilizar para realizar el trabajo y se transcribirá el seguimiento al proceso a transcur durante su transcurso, como el desarrollo de las ecuaciones a través de los métodos propuestos, los artefactos producidos durante el diseño y una vista previa de la estructura final de las herramientas que son el objetivo principal. Para esto se presentará la implementación de las funciones necesarias, explicando las estructuras de datos a crear y los prototipos de las principales funciones.

En el capítulo 4 se muestran los resultados de las pruebas finales realizadas, así como también las conclusiones, recomendaciones y posibles trabajos futuros.

En el apéndice A estará el código fuente de las funciones implementadas.

Como un aspecto de notación, todas las variables escalares y vectores o matrices, involucrados se expresan en letras itálicas (ie: x, u, y) y letra negrilla (ie: $\mathbf{u}$, $\mathbf{f}$, $\mathbf{A}$) respectivamente, mientras que para aquellas variables o instrucciones involucradas en la implementación computacional se escribirán con letra tipo (i.e: `ii`, `TDMatrix`)

CAPÍTULO 1

PLANTEAMIENTO DEL PROBLEMA

La necesidad del hombre por formalizar matemáticamente las leyes de la naturaleza, lo ha llevado a considerar temas de difícil comprensión, pero de gran utilidad en el desarrollo de la tecnología. Al estudiarse importantes fenómenos, visibles en la vida cotidiana pero de compleja estructura, se han originado numerosas técnicas que modelan y aproximan su comportamiento.

Entre estos fenómenos, el flujo de fluidos, más concretamente, es objeto de intensos estudios por científicos alrededor del mundo, no sólo por la amplitud que el tema ofrece, sino por sus numerosas aplicaciones en campos tecnológicos y empresariales.

1.1 Formulación del Problema

El planteamiento exige simular el flujo de fluidos sobre sólidos poligonales 2D utilizando el método de Elementos Finitos Mínimos Cuadrados aplicado a un sistema de ecuaciones diferenciales conocidas como ecuaciones de Navier-Stokes para el fluido no estacionario e incompresible.

Sean: Re: la constante de Reynolds
p: la presión del fluido en el punto $(x, y) \in \Omega \subset \mathbb{R}^2$
t: el tiempo en $(0, Tmax)$
$\mathbf{v}$: el vector velocidad

y la constante de Reynolds definida como:

$$Re = \frac{UL}{\nu},$$

donde L es una longitud de referencia, U es una velocidad de referencia y ν la viscosidad cinemática, el sistema de ecuaciones de Navier-Stokes se representa como:

$$\frac{D\mathbf{v}}{\partial t} - \frac{1}{Re}\Delta\mathbf{v} + \nabla p = \mathbf{0} \quad en \ \ \Omega \tag{1.1.1}$$

En donde:

$$\begin{aligned}
\mathbf{v} &= \begin{pmatrix} v_x \\ v_y \end{pmatrix} \\
\nabla p &= \begin{pmatrix} \frac{\partial p}{\partial x} \\ \frac{\partial p}{\partial y} \end{pmatrix} \\
\tfrac{D\mathbf{v}}{\partial t} &= \tfrac{\partial \mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\,\mathbf{v} \\
\Delta\mathbf{v} &= \nabla(\nabla\cdot\mathbf{v}) - \nabla\times\nabla\times\mathbf{v} = \begin{pmatrix} \frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} \\ \frac{\partial^2 v_y}{\partial x^2} + \frac{\partial^2 v_y}{\partial y^2} \end{pmatrix} \\
(\mathbf{v}\cdot\nabla)\,\mathbf{v} &= \left((v_x v_y)\begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{pmatrix}\right)\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \begin{pmatrix} v_x\frac{\partial v_x}{\partial x} + v_y\frac{\partial v_x}{\partial y} \\ v_x\frac{\partial v_y}{\partial x} + v_y\frac{\partial v_y}{\partial y} \end{pmatrix}
\end{aligned} \tag{1.1.2}$$

Al escribir algebraicamente el sistema, se tiene:

$$\begin{aligned}
&\frac{\partial v_x}{\partial t} + v_x\frac{\partial v_x}{\partial x} + v_y\frac{\partial v_x}{\partial y} - \frac{1}{Re}\left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2}\right) + \frac{\partial p}{\partial x} = 0 \quad en \quad \Omega \\
&\frac{\partial v_y}{\partial t} + v_x\frac{\partial v_y}{\partial x} + v_y\frac{\partial v_y}{\partial y} - \frac{1}{Re}\left(\frac{\partial^2 v_y}{\partial x^2} + \frac{\partial^2 v_y}{\partial y^2}\right) + \frac{\partial p}{\partial y} = 0 \quad en \quad \Omega,
\end{aligned} \tag{1.1.3}$$

teniendo así 2 ecuaciones (1.1.3) y 3 incógnitas, por ello es necesario involucrar otra ecuación (ecuación de continuidad) (1.1.4) que permita completar el sistema y al agregarle las condiciones de frontera, que serán (1.1.6) , se tiene un problema bien planteado y con solución única, en la simulación de flujo de fluido viscoso.

Ecuación de continuidad:

$$\nabla \cdot \mathbf{v} = 0 \tag{1.1.4}$$

es decir,

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0 \tag{1.1.5}$$

Sobre la Frontera Γ se establecen las siguientes condiciones:

$$\begin{aligned} \mathbf{n} \cdot \mathbf{v} &= 0 \\ p &= p_0 \end{aligned} \tag{1.1.6}$$

aquí $\mathbf{n}$ representa el vector normal a la frontera $[n_x, n_y]$ y p_0 es un valor constante de presión. O lo que es lo mismo

$$\begin{aligned} n_x v_x + n_y v_y &= 0 \\ p &= p_0 \end{aligned} \tag{1.1.7}$$

1.2 Antecedentes

En la década de los 30 aparecen y se desarrollan los métodos de diferencias finitas (MDF). El método de elementos finitos nace en 1943 cuando Courant [1] utiliza una combinación lineal de funciones en un artículo en la American Mathematical Society, pero no es hasta los 50 en que se le comienza a denominar *Elementos Finitos*.

En 1979 Bramble y Schatz [3, 4] desarrollan el método de elementos finitos del tipo mínimos cuadrados, concernientes a la solución de problemas elípticos con valores en la frontera.

En 1998, B. Jiang [17] demuestra cómo utilizar el Método de Elementos Finitos Mínimos Cuadrados basado en la formulación velocidad-presión-vorticidad de primer orden, con el fin de resolver varios problemas relacionados con flujo viscoso incompresible, entre ellos una adaptación rigurosa de la ecuaciones de Navier-Stokes y sus condiciones de frontera adicionales.

En 1999, J. Stam [19] propone por primera vez un modelo incondicionalmente estable para producir flujos de fluidos complejos que permite la interacción de los mismos con otros elementos en un escenario de simulación en 2D y 3D. Esto lo logra a través del modelo matemático de Navier-Stokes y un uso eficiente de técnicas propias de la graficación por computadora.

Posteriormente J. Stam [22] presenta la implementación de un *solver* para fluidos, consistente con las ecuaciones de flujo de fluidos que producen campos de velocidad que contienen estructuras rotacionales compresibles y que reaccionan dinámicamente a fuerzas ejercidas por el usuario. Este

solver especializado en fluidos que cubren el espacio, permite tomar ventaja de la Transformada de Fourier, lo que simplifica considerablemente muchos aspectos del mismo y puede ser usado como primitiva de movimientos básicos de diferentes aplicaciones en computación gráfica.

Más recientemente, De Cecchis [23] desarrolla un conjunto de rutinas para resolver ecuaciones diferenciales elípticas de segundo orden en dos dimenciones, mientras que Manteuffel et al. [25], abordan el problema de ecuaciones diferenciales parciales lineales hiperbólicas. Asimismo, Starke [27] utiliza LSEFM en la formulación tensión-desplazamiento del problema de elasticidad. Finalmente, Cadenas [31] efectúa un trabajo acerca del problema de dispersión de ondas en una dimensión, comparando LSEFM con otros métodos numéricos para resolver ecuaciones diferenciales, con base en funciones para aproximación globales lineales a trozos.

Para el año 2003 Fagúndez y Medina [30] desarrollan un software para la Simulación Numérica de Flujo de Fluido Viscoso Incompresible en 2D, utilizando para ello la librería UCSparseLib y OpenGL.

Ya en 2004 Cadenas y Villamizar [32] y [33] presentan una nueva formulación del método de elementos finitos mínimos cuadrados donde se involucran las condiciones de frontera en la formulación variacional.

En este mismo año, Rojas [34] realiza la implementación del método LSEFM utilizando una base de polinomios de interpolación de Lagrange de grado superior en una dimensión, así como también, base de polinomios de Hermite de grado tres y cinco.

Estos tres últimos trabajos serán extremadamente importantes para el entendimiento de la metodología y aplicación de este método al problema de la simulación de flujo de fluidos, se debe destacar que la implementación de esta nueva técnica sólo se ha hecho en 1D.

1.3 Justificación

El estudio de la dinámica de fluidos comprende una amplia rama de análisis numérico en la que se analiza la interacción de numerosas y complejas fuerzas físicas que afectan a una sustancia en movimiento. Este campo ha sido vastamente explorado y mejorado por matemáticos e ingenieros en todo el mundo, quienes han creado teorías y modelos que representan las leyes que obedecen los fluidos al transcurrir el tiempo. Estos modelos han sido altamente aplicados a varios problemas actuales en diversos lugares como universidades, refinerías petroleras, juegos y películas de cine, entre otros.

En esta oportunidad se implementará el método de elementos finitos mínimos cuadrados para aproximar eficientemente las ecuaciones diferenciales correspondientes a la simulación del flujo de fluidos en 2D en interacción con un cuerpo regular.

Este paquete de simulación no sólo es de gran interés para los investigadores en el área, sino también para empresas que deseen observar resultados inmediatos en situaciones no existentes y que conllevarían a gastos excesivos.

1.4 Objetivos

Objetivo General

Desarrollar un conjunto de herramientas computacionales que permitan simular el flujo de fluidos en 2D mediante el uso de elementos finitos mínimos cuadrados.

Objetivos Específicos

1. Estudiar la dinámica de fluidos, la teoría básica de los métodos de elementos finitos, el método de elementos finitos mínimos cuadrados, la metodología de diseño de software, los métodos numéricos para resolver un sistema de ecuaciones lineales esparcidas y los lenguajes necesarios, para obtener los conocimientos básicos para la resolución del problema por medio de una detallada revisión bibliográfica.

2. Diseñar un conjunto de herramientas computacionales que simulen numéricamente el flujo y su interacción con un cuerpo poligonal utilizando los conocimientos ya mencionados.

3. Implementar el diseño elaborado, utilizando el lenguaje de programación *C* y las librerías necesarias, entre las cuales estará openGL para el manejo de gráficos, y UCSparceLib para las operaciones matriciales.

4. Probar el software resultante utilizando la visualización gráfica para observar la dinámica del fluido simulado y estudiar su comportamiento, así como, por medio de comparaciones con resultados presentados en la bibliografía.

1.5 Enfoque Metodológico

Con el fin de cumplir con los objetivos planteados se ha trazado un proceso iterativo de desarrollo con etapas simples y convencionalmente delimitadas pero extremadamente flexibles:

1. Análisis: Se estudiará exhaustivamente la teoría sobre las ecuaciones diferenciales, su aproximación a través del método de elementos finitos mínimos cuadrados y su aplicación a la dinámica de fluidos.
2. Diseño: Se elaborará un diseño de toda la estructura del producto que se creará y todos los artefactos necesarios para la previa comprensión de éste.
3. Implementación: Basado en los pasos anteriores se plasmarán en lenguaje *C* las herramientas computacionales mencionadas como objetivo del trabajo. Se realizará una retroalimentación iterativa que posiblemente conllevará a la mejora del diseño del software.
4. Prueba: Una vez realizada la implementación se probará el producto, lo que abrirá la posibilidad de retornar a los pasos 2 ó 3 para la mejora incremental del software.

CAPÍTULO 2

MARCO TEÓRICO

En este capítulo se estudiarán los diferentes aspectos que comprenderán este trabajo, como lo son algunos conceptos básicos de la dinámica de fluidos y el basamento de las ecuaciones que los rigen, así como la formulación de los métodos utilizados para aproximar estas ecuaciones por medio de ecuaciones en diferencias y obtener su solución numérica.

2.1 Conceptos Básicos

2.1.1 Mecánica de Fluidos

Mecánica de fluidos, según James [20], es la parte de la física que se ocupa de la acción de los fluidos en reposo o en movimiento, así como de las aplicaciones y mecanismos de ingeniería que utilizan fluidos. La mecánica de fluidos es fundamental en campos tan diversos como la aeronáutica, la ingeniería química, civil e industrial, la meteorología, las construcciones navales y la oceanografía.

La mecánica de fluidos puede subdividirse en dos campos principales: la estática de fluidos, o hidrostática, que se ocupa de los fluidos en reposo, y la dinámica de fluidos, que trata de los fluidos en movimiento.

Los principios básicos del movimiento de los fluidos se desarrollaron lentamente a través de los siglos XVI al XIX como resultado del trabajo de muchos científicos como Da Vinci, Galileo, Torricelli, Pascal, Bernoulli, Euler, Navier, Stokes, Kelvin, Reynolds y otros que hicieron interesantes aportes teóricos a lo que se denomina hidrodinámica. También en el campo de hidráulica experimental hicieron importantes contribuciones Chezy, Ventura, Hagen, Manning, Pouseuille, Darcy, Froude y otros, fundamentalmente durante el siglo XIX.

La Mecánica de Fluidos moderna aparece a principios del siglo XX como un esfuerzo para unir estas dos tendencias: experimental y científica. Generalmente se reconoce como fundador de la mecánica de fluidos al alemán L. Prandtl (1875-1953). Esta es una ciencia relativamente joven a la cual aún hoy se están haciendo importantes contribuciones.

La referencia que da el autor Vernard J.K [5] acerca de los antecedentes de la mecánica de fluidos como un estudio científico datan según sus investigaciones de la antigua Grecia en el año 420 a.C. hechos por Tales de Mileto y Anaxímenes; que después continuarían los romanos y se siguiera continuando el estudio hasta el siglo XVII.

Continuidad

Un fluido es una sustancia que se deforma continuamente al ser sometida a un esfuerzo cortante (esfuerzo tangencial) no importa cuan pequeño sea.

Todos los fluidos están compuestos de moléculas que se encuentran en movimiento constante. Sin embargo, en la mayor parte de las aplicaciones de ingeniería, nos interesa más conocer el efecto global o promedio (es decir, macroscópico) de las numerosas moléculas que forman el fluido. Son estos efectos macroscópicos los que realmente podemos percibir y medir.

Por lo anterior, consideraremos que el fluido está idealmente compuesto de una sustancia infinitamente divisible (es decir, como un continuo) y no nos preocuparemos por el comportamiento de las moléculas individuales.

El concepto de un continuo es la base de la mecánica de fluidos clásica. La hipótesis de un continuo resulta válida para estudiar el comportamiento de los fluidos en condiciones normales. Sin embargo, dicha hipótesis deja de ser válida cuando la trayectoria media libre de las moléculas (aproximadamente 6,3 x 10-5 mm o bien 2.5 x 10-6 pulg para aire en condiciones normales de presión y temperatura) resulta del mismo orden de magnitud que la longitud significativa más pequeña, característica del problema en cuestión.

Una de las consecuencias de la hipótesis del continuo es que cada una de las propiedades de un fluido se supone que tenga un valor definido en cada punto del espacio. De esta manera, propiedades como la densidad, temperatura, velocidad, etc., pueden considerarse como funciones continuas de la posición y del tiempo.

Euler fue el primero en reconocer que las leyes dinámicas para los fluidos sólo pueden expresarse de forma relativamente sencilla si se supone que el fluido es incompresible e ideal, es decir, si se pueden despreciar los efectos del rozamiento y la viscosidad. Sin embargo, como esto nunca es así en el caso de los fluidos reales en movimiento, los resultados de dicho análisis sólo pueden servir como estimación para flujos en los que los efectos de la viscosidad son pequeños.

La ecuación de continuidad o conservación de masa es una herramienta muy útil para el análisis de fluidos que fluyen a través de tubos o ductos con diámetro variable. En estos casos, la velocidad del flujo cambia debido a que el área transversal varía de una sección del ducto a otra.

Propiedades del fluido

De acuerdo con Muller [12] y Williams [16], el fluido tiene las siguientes propiedades principales:

- Densidad Es la cantidad de masa por unidad de volumen. Afecta de manera decisiva en cualquier fuerza aplicada sobre o desde el fluido.
- Peso específico

  Es el peso por unidad de volumen.
- Viscosidad

  Propiedad de un fluido que tiende a oponerse a su flujo cuando se le aplica una fuerza. Los fluidos de alta viscosidad presentan una cierta resistencia a fluir; los fluidos de baja viscosidad fluyen con facilidad. La fuerza con la que una capa de fluido en movimiento arrastra consigo a las capas adyacentes de fluido determina su viscosidad, que se mide con un recipiente (viscosímetro) que tiene un orificio de tamaño conocido en el fondo. La velocidad con la que el fluido sale por el orificio es una medida de su viscosidad. La viscosidad de un fluido disminuye con la reducción de densidad que tiene lugar al aumentar la temperatura. En un fluido menos denso hay menos moléculas por unidad de volumen que puedan transferir impulso desde la capa en movimiento hasta la capa estacionaria. Esto, a su vez, afecta a la velocidad de las distintas capas. El momento se transfiere con más dificultad entre las capas, y la viscosidad disminuye. En algunos líquidos, el aumento de la velocidad molecular compensa la reducción de la densidad.

- Compresibilidad.

  La compresibilidad representa la relación entre los cambios de volumen y los cambios de presión a que está sometido un fluido. Las variaciones de volumen pueden relacionarse directamente con variaciones de la masa específica si la cantidad de masa permanece constante. En general se sabe que en los fluidos la masa específica depende tanto de la presión como de la temperatura de acuerdo a la ecuación de estado.

- Tensión superficial

  Las moléculas de un líquido ejercen fuerzas pequeñas de atracción, unas sobre las otras. Aún cuando en general las moléculas son eléctricamente neutras, con frecuencia existe una ligera asimetría de carga que da origen a fuerzas de atracción entre ellas (llamadas fuerzas de van der Waals). Dentro de un líquido, en el que cada molécula está completamente rodeada de otras moléculas, la fuerza neta es cero. A pesar de ello, para las moléculas de la superficie del líquido, no existen fuerzas de atracción que actúen de arriba de la superficie hacia el interior del líquido. Como resultado, las moléculas de la capa superficial experimentan fuerzas netas debido a las moléculas vecinas, que están justo debajo de la superficie. Este impulso hacia abajo sobre las moléculas de la superficie causa que el líquido se contraiga y resista ser estirado o roto, propiedad que se llama tensión superficial. El efecto neto de la tensión superficial es hacer que el área de la superficie de un líquido sea tan pequeña como sea posible. Esto es, un volumen dado de líquido tiende a adoptar la forma que tiene el área superficial menor. Cuantitativamente se define como la fuerza por unidad de longitud que actúa a lo largo de una línea cuando se estira la superficie.

Clasificación

- Flujos compresibles e incompresibles

  Uno de los principios básicos del flujo compresible es que la densidad de un gas cambia cuando el gas se ve sometido a grandes cambios de velocidad y presión. Al mismo tiempo, su temperatura también cambia, lo que lleva a problemas de análisis más complejos. Los fluidos incompresibles cumplen el llamado teorema de Bernoulli, que afirma que la energía mecánica total de un flujo incompresible y no viscoso (sin rozamiento) es constante a lo largo de una línea de corriente. Las líneas de corriente son líneas de flujo imaginarias que siempre son paralelas a la dirección del flujo en cada punto, y en el caso de flujo uniforme coinciden con la trayectoria de las partículas individuales de fluido. El teorema de Bernoulli implica una relación entre los efectos de la presión, la velocidad y la gravedad, e indica que la velocidad aumenta cuando la presión disminuye.

- Flujos viscosos e invíscidos

  Un flujo puede ser clasificado de una manera general como flujo viscoso o flujo invíscido. Un flujo invíscido es aquel en el que los efectos viscosos no influyen significativamente en el flujo y por lo tanto son ignorados. En un flujo viscoso los efectos de viscosidad son importantes y no pueden ser ignorados. Para modelar un flujo invíscido analíticamente, simplemente la viscosidad se hace cero; esto obviamente hace que todos los efectos viscosos sean cero. Con base en la experiencia, se encuentra que la clase principal de flujos que pueden ser modelados como flujos invíscidos, son los flujos externos, es decir, los flujos que existen en el exterior de un cuerpo, tales como el flujo alrededor de una superficie aerodinámica o una superficie hidrodinámica. Los flujos viscosos incluyen la amplia clase de flujos internos, tales como flujos en tubos y conductos y en canales abiertos. En flujos como esos los efectos viscosos provocan pérdidas sustanciales y responden a las inmensas cantidades de energía que deben ser utilizadas para transportar petróleo y gas por oleoductos. La condición no deslizante que produce una velocidad cero en la pared, y los esfuerzos cortantes resultantes, conducen directamente a estas pérdidas.

  Los primeros experimentos cuidadosamente documentados del rozamiento en flujos de baja velocidad a través de tuberías fueron realizados independientemente por Poiseuille y por Hagen. El primer intento de incluir los efectos de la viscosidad en las ecuaciones matemáticas se debió a Navier e, independientemente, a Stokes, quien perfeccionó las ecuaciones básicas para los fluidos viscosos incompresibles. Actualmente se las conoce como ecuaciones de Navier-Stokes, y son tan complejas que sólo se pueden aplicar a flujos sencillos. Uno de ellos es el de un fluido real que circula a través de una tubería recta.

  El teorema de Bernoulli no se puede aplicar aquí, porque parte de la energía mecánica total se disipa como consecuencia del rozamiento viscoso, lo que provoca una caída de presión a lo largo de la tubería. Las ecuaciones sugieren que, dados una tubería y un fluido determinados, esta caída de presión debería ser proporcional a la velocidad de flujo. Los experimentos demostraron que esto sólo era cierto para velocidades bajas; para velocidades mayores, la caída de presión era más bien proporcional al cuadrado de la velocidad.

  Este problema se resolvió cuando Reynolds demostró la existencia de dos tipos de flujo viscoso en tuberías. A velocidades bajas, las partículas del fluido siguen las líneas de corriente (flujo laminar), y los resultados experimentales coinciden con las predicciones analíticas. A velocidades más elevadas, surgen fluctuaciones en la velocidad del flujo, o remolinos (flujo turbulento), en una forma que ni siquiera en la actualidad se puede predecir completamente.

  Reynolds también determinó que la transición del flujo laminar al turbulento era función de un único parámetro, que desde entonces se conoce como número de Reynolds. Si el número de Reynolds (que carece de dimensiones y es el producto de la velocidad, la densidad del fluido

y el diámetro de la tubería dividido entre la viscosidad del fluido) es menor de 2.000, el flujo a través de la tubería es siempre laminar; cuando los valores son mayores a 3000 el flujo es turbulento. El concepto de número de Reynolds es esencial para gran parte de la moderna mecánica de fluidos.

- Flujo laminar y turbulento

  En un flujo laminar el fluido fluye sin mezclado significativo de sus partículas próximas entre sí. El flujo es ordenado y predecible, el movimiento se produce en capas o láminas y las soluciones matemáticas son factibles. Es este flujo las partículas se mueven en trayectorias independientes de las partículas de capas adyacentes.

  En un flujo turbulento los movimientos del fluido varían irregularmente de tal suerte que las cantidades tales como velocidad y presión muestran una variación aleatoria con el tiempo y las coordenadas espaciales. Las cantidades físicas con frecuencia se describen mediante promedios estadísticos. En este sentido, un flujo turbulento ”continuo” puede ser definido como un flujo en el que las cantidades físicas promedio dependen del tiempo.

  Los flujos turbulentos no se pueden evaluar exclusivamente a partir de las predicciones calculadas, y su análisis depende de una combinación de datos experimentales y modelos matemáticos; gran parte de la investigación moderna en mecánica de fluidos está dedicada a una mejor formulación de la turbulencia. Puede observarse la transición del flujo laminar al turbulento y la complejidad del flujo turbulento cuando el humo de un cigarrillo asciende en aire muy tranquilo. Al principio, sube con un movimiento laminar a lo largo de líneas de corriente, pero al cabo de cierta distancia se hace inestable y se forma un sistema de remolinos entrelazados.

- Flujos de la capa límite

  Los flujos pueden separarse en dos regiones principales. La región próxima a la superficie está formada por una delgada capa límite donde se concentran los efectos viscosos y en la que puede simplificarse mucho el modelo matemático. Fuera de esta capa límite, se pueden despreciar los efectos de la viscosidad, y pueden emplearse las ecuaciones matemáticas más sencillas para flujos no viscosos.

  La teoría de la capa límite ha hecho posible gran parte del desarrollo de las alas de los aviones modernos y del diseño de turbinas de gas y compresores.

Movimientos de fluido

Los movimientos de fluido se manifiestan de diferentes maneras. Algunos pueden ser descritos con facilidad, en tanto que otros requieren de un conocimiento completo de las leyes de física.

En aplicaciones de ingeniería es importante describir los movimientos de fluidos tan simplemente como puedan ser justificados. Esto en general depende de la precisión requerida y de las suposiciones simplificadoras de las ecuaciones de movimiento, las cuales son muy difíciles de resolver. Algunas suposiciones comunes utilizadas para simplificar una situación de flujo tienen que ver con las propiedades del fluido. Por ejemplo, en ciertas condiciones, la viscosidad puede afectar el flujo de manera significativa; en otras, los efectos viscosos pueden ser omitidos, con lo que se simplifican en gran medida las ecuaciones sin que se alteren significativamente las predicciones. La compresibilidad de un gas en movimiento deberá ser tomada en cuenta si las velocidades son muy altas, mas los efectos de compresibilidad no tienen que ser tomados en cuenta para predecir las fuerzas del viento que actúan en edificios o para predecir cualquier otra cantidad física que sea un efecto directo del viento. Después de estudiar los movimientos de fluidos, las suposiciones apropiadas utilizadas deberán ser más que obvias. El análisis de problemas de flujo de fluidos complejo a menudo se simplifica mediante la visualización de patrones de flujo, los que permiten desarrollar un mayor entendimiento intuitivo y ayudan a formular el problema matemático. En este trabajo se considera la simulación de los llamados fluidos Newtonianos, los cuales comprenden la gran mayoría, entre ellos el agua, aire y aceite. La principal característica de este tipo de fluidos es que el esfuerzo cortante es proporcional al gradiente de velocidad.

Más información sobre fluidos se puede encontrar en el trabajo de Iván Torres http://www.monografias.com/trabajos12/mecflui/mecflui.shtml.

2.1.2 Métodos de Compresibilidad Artificial

Como lo mencionan Medina y Fagúndez [30], el área de los flujos compresibles es una de gran importancia en la mecánica de fluidos por sus aplicaciones en aerodinámica y diseño de turbinas, lo que ha llevado al desarrollo de métodos para su solución numérica. Surge entonces la posibilidad de adaptar tales métodos a la solución de flujos incompresibles. La versión compresible contiene la derivada en el tiempo de la densidad. En el caso de los flujos incompresibles la densidad es constante, lo cual elimina la opción de colocar tal derivada. Esto plantea que la derivada en el tiempo de la presión es una clara elección para estos casos, lo cual significa que realmente no se resuelven las ecuaciones de incompresibilidad. Como resultado, se ha cuestionado mucho la idea de utilizar métodos de compresibilidad artificial en problemas de flujo incompresible; sin embargo, ha sido ampliamente probado. Esta propuesta fue hecha por primera vez por Chorin [2] en 1967, planteandola entre otras versiones posibles basadas en los métodos compresibles. En resumen, la propuesta esencial es agregar la derivada en el tiempo de la presión a la ecuación de continuidad:

$$\frac{1}{\beta}\frac{\partial p}{\partial t} + \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0, \tag{2.1.1}$$

donde β es un parámetro de compresibilidad artificial cuyo valor es clave en el desempeño de este método. Obviamente, mientras mayor sea el valor de β, mayor será la "incompresibilidad" en las ecuaciones considerando el caso de densidad constante.

El factor crucial en la convergencia de este método, es la escogencia del parámetro β, debido a que el valor óptimo es dependiente del problema. Aún así, algunos autores han sugerido procedimientos automáticos para tal selección, ya que valores demasiado grandes requieren de un correcto campo de velocidad para poder satisfacer la ecuación de continuidad incompresible. También es posible determinar el menor valor aceptable para β a través de la propagacion de velocidad de las ondas de presión. Obviamente, $1/(\beta\Delta t)$ debe ser pequeño comparado con los coeficientes de los otros términos de la ecuación, lo cual es necesario si se desea una rápida convergencia.

2.1.3 Ecuaciones Diferenciales y Condiciones de Frontera

Con el fin de analizar un sistema de ingeniería, un modelo matemático es desarrollado para describirlo. Algunas expresiones matemáticas nacen para ello y la mayoría consiste en ecuaciones diferenciales sometidas a ciertas condiciones. En general, un sistema de ecuaciones diferenciales puede escribirse como

$$\mathbf{A}\mathbf{u} = \mathbf{f} \quad \text{en } \Omega, \tag{2.1.2}$$

donde $\mathbf{A}$ es un operador diferencial, $\mathbf{u}$ y $\mathbf{f}$ son vectores de funciones suficientemente diferenciables para satisfacer las restricciones del operador $\mathbf{A}$, y $\mathbf{u}$ se conoce como *vector solución* o *vector de variables dependientes*. Por lo que $\mathbf{A}$ puede verse como una función que depende de $\mathbf{u}$.

Un sistema de ecuaciones diferenciales puede siempre ser transladado a un sistema de primer orden para facilitar los cálculos. Un sistema como éste en cualquier dimensión puede escribirse de esta forma:

$$\sum_{k=1}^{nd}\sum_{j=1}^{m} a_{ij,k}\frac{\partial u_j}{\partial x_k} + \sum_{j=1}^{m} a_{ij}u_j = f_i \quad i = 1, 2, \ldots, m. \tag{2.1.3}$$

o matricialmente:

$$\mathbf{A}\mathbf{u} \equiv \sum_{k=1}^{n} \mathbf{A}_k\frac{\partial \mathbf{u}}{\partial x_k} + \mathbf{A}_0\mathbf{u} = \mathbf{f} \tag{2.1.4}$$

donde $\mathbf{u}^T = (u_1, u_2, \ldots, u_m)$ es un vector de m funciones desconocidas que dependen de $x = (x_1, x_2, ..., x_n)$, $\mathbf{A}_k = (a_{ij,k})$ y $\mathbf{A}_0 = (a_{ij})$ son matrices $m \times m$, y $\mathbf{f}^T = (f_1, f_2, \ldots, f_m)$.

El sistema (2.1.3) se dice quasi-lineal si $a_{ij,k}$ y a_{ij} son funciones que dependen de x y $\mathbf{u}$, casi-lineal si $a_{ij,k}$ dependen sólo de x y a_{ij} son funciones como en sistemas quasi-lineales, y lineal, si ambos $a_{ij,k}$ y a_{ij} dependen sólo de x.

Un sistema se puede definir lineal si y solo si se cumple que

$$\mathbf{A}(\alpha\mathbf{u} + \beta\mathbf{v}) = \alpha\mathbf{A}(\mathbf{u}) + \beta\mathbf{A}(\mathbf{v}). \tag{2.1.5}$$

para todo escalar α, β, y vectores de funciones $\mathbf{u}, \mathbf{v}$. Si $\mathbf{f} \equiv \mathbf{0}$ en la ecuación (2.1.2) , se dice que el sistema es *homogéneo*, en caso contrario, *no homogéneo*.

Un dominio es una colección de puntos en el espacio con la propiedad de que si p es un punto del dominio, entonces todos los puntos suficientemente cercanos a p, pertenecen al dominio. Si dos puntos cualesquiera del dominio pueden ser unidos por una recta, cuyos puntos también pertenezcan al dominio, se conoce como dominio *convexo y simplemente conexo*. En la ecuación (2.1.2) se conoce como Ω al *dominio* del sistema de ecuaciones diferenciales, y para este contexto siempre se definirá real.

Cuando un dominio tiene una *frontera*, se denomina dominio *acotado*. La frontera se define como un conjunto de puntos tales que en su vecindad, existen puntos que pueden pertenecer o no al dominio (ver figura 2.1).

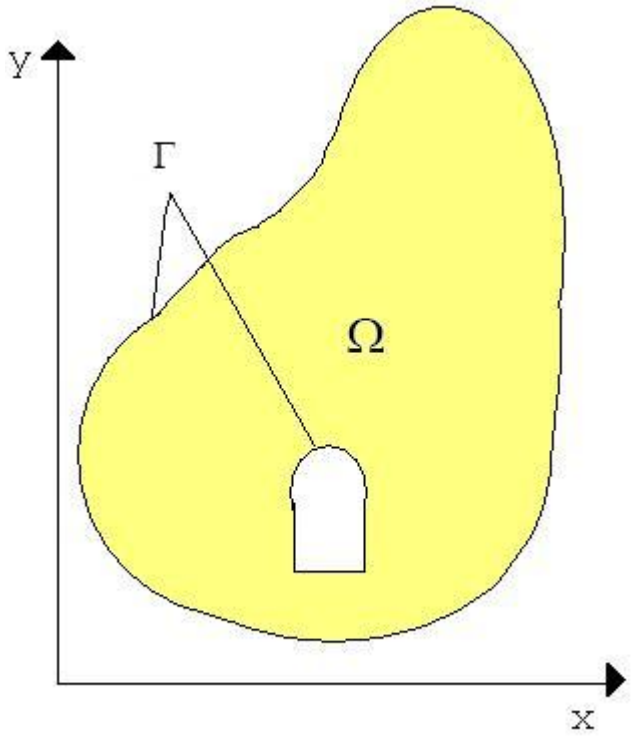


Figura 2.1: Dominio bidimensional acotado

Para garantizar la unicidad de la solución del sistema de ecuaciones diferenciales, es necesario imponer algunas condiciones sobre la variable dependiente. Cuando estas restricciones se aplican a los valores de la función solución y/o a sus derivadas en los puntos de la frontera, se conoce como un *problema de valor de frontera*. Las condiciones de frontera se denotan como

$$\mathbf{Bu} = \mathbf{g} \quad \text{sobre } \Gamma, \tag{2.1.6}$$

donde B es un operador diferencial, u la variable dependiente y g es una función definida en la frontera Γ. Dependiendo de como se formulan las condiciones de frontera se pueden tener diferentes tipos. Cuando se condiciona únicamente el valor de la variable dependiente, se conoce como condición de *Dirichlet*, y tiene la forma

$$\mathbf{hu} = \mathbf{g} \quad \text{sobre } \Gamma. \tag{2.1.7}$$

donde $\mathbf{h}$ es un operador lineal definido en Γ.

Cuando la condición prescribe el valor de la derivada de la variable dependiente, se conoce como condición de *Newmann*,

$$\mathbf{a}\frac{\partial \mathbf{u}}{\partial n} = \mathbf{g} \quad \text{sobre } \Gamma, \tag{2.1.8}$$

donde n es el vector normal unitario externo a la frontera y $\mathbf{a}$ un operador definido en Γ.

La condición donde se involucra tanto la variable dependiente como su derivada, se conoce como condición de *Robin*, *Newmann generalizada* o *mixta*, y es de la forma

$$\mathbf{a}\frac{\partial \mathbf{u}}{\partial n} + \mathbf{hu} = \mathbf{g} \quad \text{sobre } \Gamma \tag{2.1.9}$$

Si en cualquiera de los casos, $\mathbf{g} \equiv \mathbf{0}$, se denomina condición *homogénea*, en otro caso se denomina *no homogénea*. Para un mismo problema de frontera, dado por una ecuación diferencial de segundo orden, se pueden tener particiones de Γ que estén determinadas por diferentes condiciones, por ejemplo para $\Gamma_1 \cup \Gamma_2 \cup \Gamma_3 = \Gamma \quad \Gamma_i \cap \Gamma_j = \emptyset \quad i \neq j \quad i, j = 1, 2, 3$:

$$\begin{aligned} \mathbf{h_1}\,\mathbf{u} &= \mathbf{g_1} \quad \text{sobre } \Gamma_1 \\ \mathbf{a_1}\,\frac{\partial \mathbf{u}}{\partial n} &= \mathbf{g_2} \quad \text{sobre } \Gamma_2 \\ \mathbf{a_2}\,\frac{\partial \mathbf{u}}{\partial n} + \mathbf{h_2}\,\mathbf{u} &= \mathbf{g_3} \quad \text{sobre } \Gamma_3 \end{aligned} \tag{2.1.10}$$

2.1.4 Operadores diferenciales básicos

Durante el transcurso del presente trabajo se emplearán operadores utilizados frecuentemente por la comunidad científica.

Gradiente (grad):

$$\nabla a = \begin{pmatrix} \frac{\partial a}{\partial x} \\ \frac{\partial a}{\partial y} \end{pmatrix} \tag{2.1.11}$$

Divergencia (div):

$$\nabla \cdot \mathbf{A} = \frac{\partial a_1}{\partial x} + \frac{\partial a_2}{\partial x} \tag{2.1.12}$$

Rotor (curl):

$$\nabla \times \mathbf{A} = \begin{vmatrix} i & j & k \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & 0 \\ a1 & a2 & 0 \end{vmatrix} = \left(\frac{\partial a_2}{\partial} x - \frac{\partial a_1}{\partial y} \right) \widehat{k} \tag{2.1.13}$$

Laplaciano:

$$\Delta = \nabla \cdot \nabla \Rightarrow \Delta a = \frac{\partial^2 a}{\partial x^2} + \frac{\partial^2 a}{\partial y^2} \tag{2.1.14}$$

Estas expresiones son representadas en 2D en donde a es un escalar y **A** un vector igual a $(a_1, a_2)^T$. Además, para su posterior utilización, se define la propiedad:

$$\nabla \times \nabla \times \mathbf{A} = \nabla (\nabla \cdot \mathbf{A}) - \Delta \mathbf{A} \tag{2.1.15}$$

y, por lo tanto, si $\nabla \cdot \mathbf{A} = 0$ resulta:

$$\nabla \times \nabla \times \mathbf{A} = -\Delta \mathbf{A} \tag{2.1.16}$$

Ahora, considérese la función $F(x, u, u')$ definida en un dominio Ω. Para un valor fijo arbitrario de la variable independiente x, F depende solo de u y u'. El cambio αv en u, con α constante y v una función, es llamado *variación* de u, denotado como δu:

$$\delta u = \alpha v. \tag{2.1.17}$$

El operador δ es llamado *símbolo variacional*. δu representa la variación admisible de la función u, en un punto fijado de la variable independiente x. Si el punto es especificado en la frontera, la variación de u debe ser cero. Por lo que δu debe satisfacer las condiciones de Dirichlet homogéneas. Hay que tener presente que δu es un cambio virtual de u. La *primera variación*

de F en u es definida como:

$$\delta F = \frac{dF}{du}\,\delta u + \frac{dF}{du'}\,\delta u' \tag{2.1.18}$$

Es de hacer notar la similitud entre la primera variación (2.1.18) y el diferencial total de F

$$dF = \frac{\partial F}{\partial x}\,dx + \frac{\partial F}{\partial u}\,du + \frac{\partial F}{\partial u'}\,du' \tag{2.1.19}$$

y ya que x es fija durante la variación de u a $u + \delta u$, el $dx = 0$. Por lo que se muestra la analogía entre δF y dF. Por tanto δ es como un operador diferencial con respecto a la variable dependiente.

Además, el operador variacional se puede intercambiar con los operadores diferenciales e integrales:

$$\frac{d}{dx}(\delta u) = \frac{d}{dx}(\alpha v) = \alpha\frac{dv}{dx} = \alpha v' = \delta u' = \delta\left(\frac{du}{dx}\right) \tag{2.1.20}$$

$$\delta\int_a^b u(x)\,dx = \int_a^b \delta u(x)\,dx \tag{2.1.21}$$

2.1.5 Discretización del tiempo

Los fenómenos físicos modelados por las ecuaciones diferenciales suelen estar relacionados con movimiento o variaciones de datos a través del tiempo. Para proceder a la simulación numérica de estos fenómenos se han desarrollado técnicas en las que se maneja el dominio temporal. La mejor técnica conocida es la del método Dominio Temporal en Diferencias Finitas o FDTD (por Finite-Difference Time-Domain). También se han diseñado otros métodos como el de Momentos (MoM por Method of Moments) que resuelve ecuaciones integrales, y otros que resuelve ecuaciones de Maxwell directamente. Sin embargo estos métodos tienen varias desventajas, como la necesidad de resolver una ecuación matricial en cada paso de tiempo.

En esta sección se explicarán algunos esquemas para discretizar el dominio del tiempo. Para esto, se dividirán los ejes temporales uniformemente en una cantidad de intervalos de tiempo. Como resultado, la variable t puede ser escrita como $t = n\Delta t$ donde Δt representa un paso de tiempo. Por brevedad, se denota $u(n\Delta t)$ como u^n. El objetivo es rescribir la expresión

$$[R]\frac{\partial\{u\}}{\partial t} = [S]\{u\} + \{f\} \tag{2.1.22}$$

Diferencia hacia adelante

Una función $u(t+\Delta t)$ se expande es una serie de Taylor sobre t como

$$u(t+\Delta t) = u(t) + \frac{\partial u}{\partial t}\Delta t + \frac{\partial^2 u}{\partial t^2}\frac{(\Delta t)^2}{2} + O\left[(\Delta t)^3\right] \tag{2.1.23}$$

donde $O\left[(\Delta t)^3\right]$ denota la suma de los términos restantes conteniendo $(\Delta t)^p$ con $p \geq 3$. A partir de la expresión (2.1.23) obtenemos

$$\frac{\partial u}{\partial t} = \frac{u(t+\Delta t) - u(t)}{\Delta t} + O\left[\Delta t\right] \approx \frac{u(t+\Delta t) - u(t)}{\Delta t} \tag{2.1.24}$$

Esta aproximación es la representación de la diferencia hacia adelante y su grado de aproximación es de primer orden ya que el error truncado contiene $(\Delta t)^p$ con $p \geq 1$. Usando la notación $u(t) = u(n\Delta t) = u^n$ se rescribe (2.1.24) como

$$\frac{\partial u}{\partial t} \approx \frac{u^{n+1} - u^n}{\Delta t}. \tag{2.1.25}$$

Si aplicamos la diferencia hacia adelante a la expresión (2.1.22) , se obtiene:

$$[R]\frac{(u^{n+1} - u^n)}{\Delta t} = [S]u^n + \{f\}^n \tag{2.1.26}$$

Diferencia hacia atrás

Una función $u(t+\Delta t)$ se expande es una serie de Taylor sobre t como

$$u(t-\Delta t) = u(t) - \frac{\partial u}{\partial t}\Delta t + \frac{\partial^2 u}{\partial t^2}\frac{(\Delta t)^2}{2} + O\left[(\Delta t)^3\right] \tag{2.1.27}$$

de la cual obtenemos

$$\frac{\partial u}{\partial t} = \frac{u(t) - u(t-\Delta t)}{\Delta t} + O\left[\Delta t\right] \approx \frac{u(t) - u(t-\Delta t)}{\Delta t} \tag{2.1.28}$$

o lo que es lo mismo

$$\frac{\partial u}{\partial t} \approx \frac{u^n - u^{n-1}}{\Delta t}. \tag{2.1.29}$$

Aplicando a nuestra expresión (2.1.22) obtenemos

$$[R]\frac{(u^n - u^{n-1})}{\Delta t} = [S]u^n + \{f\}^n \tag{2.1.30}$$

Diferencia central

Si sustraemos la aproximación (2.1.23) de (2.1.27) obtenemos

$$\frac{\partial u}{\partial t} = \frac{u(t+\Delta t) - u(t-\Delta t)}{2\Delta t} + O\left[(\Delta t)^3\right] \tag{2.1.31}$$

de lo que resulta la aproximación de segundo orden llamada de diferencia central.

$$\frac{\partial u}{\partial t} \approx \frac{u(t+\Delta t) - u(t-\Delta t)}{2\Delta t} = \frac{u^{n+1} - u^{n-1}}{2\Delta t} \tag{2.1.32}$$

La expresión (2.1.22) se rescribe como:

$$[R]\frac{(u^{n+1} - u^{n-1})}{2\Delta t} = [S]u^n + \{f\}^n \tag{2.1.33}$$

Método theta

El método theta generaliza los métodos anteriores a través de un parámetro constante θ.

Se expande la función $u(t+\Delta t)$ como una suma de Taylor:

$$u^{n+1} = u^n + \frac{\partial u}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 u}{\partial t^2}(\Delta t)^2 + O\left[(\Delta t)^3\right] \tag{2.1.34}$$

Multiplicando la expresión por $1-\theta$ tenemos:

$$(1-\theta)u^{n+1} = (1-\theta)u^n + (1-\theta)\frac{\partial u}{\partial t}\Delta t + \frac{1-\theta}{2}\frac{\partial^2 u}{\partial t^2}(\Delta t)^2 + O\left[(\Delta t)^3\right] \tag{2.1.35}$$

Lo cual se puede escribir como

$$u^{n+1} = \theta u^{n+1} + (1-\theta)u^n + O\left[\Delta t\right] \tag{2.1.36}$$

siendo esta expresión en general de primer orden, aunque para valores $\theta = 1$ y $\theta = 1 - \Delta t$ se reduce a una expresión de segundo orden.

Teniendo esto, la expresión (2.1.22) se escribe como:

$$[R]\frac{u^{n+1} - u^n}{\Delta t} = \theta[S]u^{n+1} + (1-\theta)[S]^n + \{f\}^{n+1} \tag{2.1.37}$$

Así, para $\theta = 1/2$ la aproximación corresponde a (2.1.32) y es de segundo orden mientras que con $\theta = 0$ y $\theta = 1$ la aproximación se asemeja a (2.1.25) y a (2.1.26) respectivamente y es de primer orden. Con el valor $\theta = \frac{2}{3}$ concuerda con un enfoque llamado de $Galerkin$. Se ha probado que con valores de θ entre este rango $[0, 1]$ el proceso será incondicionalmente estable.

2.1.6 Linealización de términos

Frecuentemente en la práctica se encuentran ecuaciones que no están condicionadas para ser procesadas por métodos de aproximación numérica. Estas ecuaciones, después de pasar por ciertos tipos de procesamiento como el de la discretización temporal 2.1.5, debido a que contienen productos cuyos factores son ambos incógnitas, necesitan ser linealizados a través de procedimientos de aproximación.

En la siguiente explicación se tienen funciones conocidas que dependen del tiempo t y de cierta variable x, $u(t) = u^n, v(t) = v^n$ cuyo valor en el siguiente paso de tiempo $u(t + \Delta t) = u^{n+1}, v(t + \Delta t) = v^{n+1}$ es desconocido.

Se intenta linealizar el siguiente término:

$$u^{n+1}\frac{\partial v^{n+1}}{\partial x} \tag{2.1.38}$$

para no tener términos incógnitas en el mismo factor.

Se aplican las sumas de Taylor:

$$u^{n+1} = u^n + \frac{\partial u^n}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 u^n}{\partial t^2}(\Delta t)^2 + O\left[(\Delta t)^3\right] \tag{2.1.39}$$

$$\frac{\partial v^{n+1}}{\partial x} = \frac{\partial v^n}{\partial x} + \frac{\partial^2 v}{\partial t \partial x}\Delta t + \frac{1}{2}\frac{\partial^3 v}{\partial t^2 \partial x}(\Delta t)^2 + O\left[(\Delta t)^3\right] \tag{2.1.40}$$

despejando se tiene

$$u^{n+1} - u^n = \frac{\partial u^n}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 u^n}{\partial t^2}(\Delta t)^2 + O\left[(\Delta t)^3\right] \quad (2.1.41)$$

$$\frac{\partial v^{n+1}}{\partial x} - \frac{\partial v^n}{\partial x} = \frac{\partial^2 v}{\partial t \partial x}\Delta t + \frac{1}{2}\frac{\partial^3 v}{\partial t^2 \partial x}(\Delta t)^2 + O\left[(\Delta t)^3\right] \quad (2.1.42)$$

Multiplicando (2.1.41) y (2.1.42) tenemos:

$$\left(u^{n+1} - u^n\right)\left(\frac{\partial v^{n+1}}{\partial x} - \frac{\partial v^n}{\partial x}\right) = \left(\frac{\partial u^n}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 u^n}{\partial t^2}(\Delta t)^2 + O\left[(\Delta t)^3\right]\right)\left(\frac{\partial v^2}{\partial t \partial x}\Delta t + \frac{1}{2}\frac{\partial^3 v}{\partial t^2 \partial x}(\Delta t)^2 + O\left[(\Delta t)^3\right]\right) \quad (2.1.43)$$

Considerando $\Delta t \to 0$ queda:

$$\left(u^{n+1} - u^n\right)\left(\frac{\partial v^{n+1}}{\partial x} - \frac{\partial v^n}{\partial x}\right) \approx 0 \quad (2.1.44)$$

$$u^{n+1}\left(\frac{\partial v^{n+1}}{\partial x} - \frac{\partial v^n}{\partial x}\right) \approx u^n\left(\frac{\partial v^{n+1}}{\partial x} - \frac{\partial v^n}{\partial x}\right) \quad (2.1.45)$$

Así, se demuestra:

$$u^{n+1}\frac{\partial v^{n+1}}{\partial x} \approx u^n\frac{\partial v^{n+1}}{\partial x} + u^{n+1}\frac{\partial v^n}{\partial x} - u^n\frac{\partial v^n}{\partial x} \quad (2.1.46)$$

donde esta última expresión es de segundo orden $O\left[(\Delta t)^2\right]$.

2.2 Elementos Finitos

Los métodos de elementos finitos han llegado a ser una importante y práctica herramienta de análisis numérico. Se han hallado aplicaciones en casi todas las áreas de ingeniería y matemática aplicada. La literatura sobre estos métodos son extensas y crecen día a día. Como cita un famoso investigador, ”Quizás ninguna otra familia de métodos de aproximación ha tenido un mayor impacto en la teoría y aplicación de los métodos numéricos durante el siglo veinte” (Jiang [17]).

El método de elementos finitos o FEM (por Finite Elements Method) no opera directamente en las ecuaciones diferenciales sino que son colocadas en forma variacional. La solución aparece en la

integral de una cantidad sobre un dominio. Esta integral sobre un dominio arbitrario puede ser dividida en la suma de otras integrales sobre subdominios llamados elementos finitos.

Según personalidades como Zienkiewicz [6] y Oden [10], las más importantes características de FEM son:

- **Geometría Arbitraria**. El método de elementos finitos es esencialmente independiente de la geometría del dominio. Puede ser aplicado a dominios de forma compleja y condiciones de frontera bastante arbitrarias.
- **Mallado sin estructura**. Los elementos finitos pueden ser colocados en cualquier parte del dominio físico. En la práctica frecuentemente se modifica el diseño original para satisfacer los diferentes requerimientos. Los analistas de FEM pueden añadir o borrar elementos sin cambiar la estructura global de los datos. Si se resuelven las ecuaciones lineales iterativamente, la numeración de nodos y elementos puede ser hecha aleatoriamente sin sacrificar eficiencia.
- **Flexibilidad**. La claridad de la estructura y versatilidad de FEM puede hacer posible construir aplicaciones de propósito general.
- **Base matemática**. Por el extensivo trabajo en la teoría matemática durante las últimas dos décadas, FEM disfruta de una rica y sólida base matemática. Esto le da confiabilidad y hace posible analizar un problema matemáticamente y estimar el grado de aproximación de las soluciones.

En esta sección se dará una breve introducción orientada principalmente al método de elementos finitos del tipo mínimos cuadrados que será el utilizado en el trabajo.

2.2.1 Discretización del Dominio

El método de elementos finitos (FEM) provee, de manera general y sistemática, una técnica para construir las funciones de forma o de aproximación ϕ_j con la finalidad de hallar una solución aproximada $\widetilde{\mathbf{u}}$ al vector variable dependiente $\mathbf{u}$ que satisfaga el sistema de ecuaciones diferenciales gobernante de algún problema particular, en donde la misma puede ser un sistema de ecuaciones diferenciales ordinarias o parciales, i.e., que las variables dependientes pertenecientes al vector $\mathbf{u}$ sean funciones de varias variables. La idea central del método de elementos finitos consiste en definir las funciones de aproximación como funciones a trozos sobre un conjunto de subregiones del dominio Ω. En este sentido, la idea general subyacente del método de elementos finitos —que

lo distingue de otros métodos para resolver ecuaciones diferenciales — consiste en la división del dominio Ω del problema dado, en un conjunto de subdominios más simples —desde un punto de vista geométrico — denominados elementos finitos.

Así pues, se considera el dominio Ω en un espacio de dimensión nd un subespacio de $\mathbb{R}^{nd}$ el cual será dividido en E subdominios Ω_e o $elemento\ finito$ no necesariamente iguales pero preferentemente de formas geométricas fácilmente integrables. Cada elemento finito está integrado por una cantidad n_e de vértices llamados $nodos$ sobre el borde y el interior del elemento. Este proceso se llama *discretización del dominio* (ver fig 2.2). La colección completa de E elementos finitos se denomina *mallado de elementos finitos*. La elección del tipo de elemento, el número de elementos y su densidad dependen de la geometría del dominio y el grado de precisión deseado. Para espacios muy irregulares o mallados muy poco detallados se puede generar un error en la discretización. El dominio Ω queda pues expresado como la aproximación de la unión generalizada de todos los elementos finitos, es decir

$$\Omega \approx \bigcup_{e=1}^{E} \Omega_e$$

Cualquier forma geométrica que permita determinar una solución aproximada, o que suministre un conjunto de relaciones algebraicas suficientemente amplio entre distintos valores de la solución —en puntos específicos del subdominio llamados nodos — califica como un elemento finito.

En dos o más dimensiones hay más de una forma geométrica simple que puede ser usada como elemento finito. Como se verá después, las funciones de interpolación no sólo dependen del número de nodos en el elemento sino también de su forma. La forma del elemento debe ser tal que su geometría es definida únicamente por un conjunto de puntos, que sirven como nodos del elemento en el desarrollo de las funciones de interpolación.

La representación de una región dada por un conjunto de elementos (discretización o generación de mallado) es un paso importante en el análisis de los elementos finitos. La escogencia del tipo de elemento, su número y densidad dependen de la geometría del dominio, del problema a ser analizado y del grado de aproximación deseado. Por supuesto, no hay fórmulas específicas para obtener esta información. Sin embargo, ciertas reglas generales pueden ser declaradas:

1. Seleccionar elementos que caractericen las ecuaciones gobernantes del problema.
2. El número, forma y tipo de los elementos debe ser tal que la geometría del dominio queda representada lo más exactamente posible.
3. La densidad de los elementos debe ser tal que las regiones de los grandes gradientes de la solución estén adecuadamente modeladas.

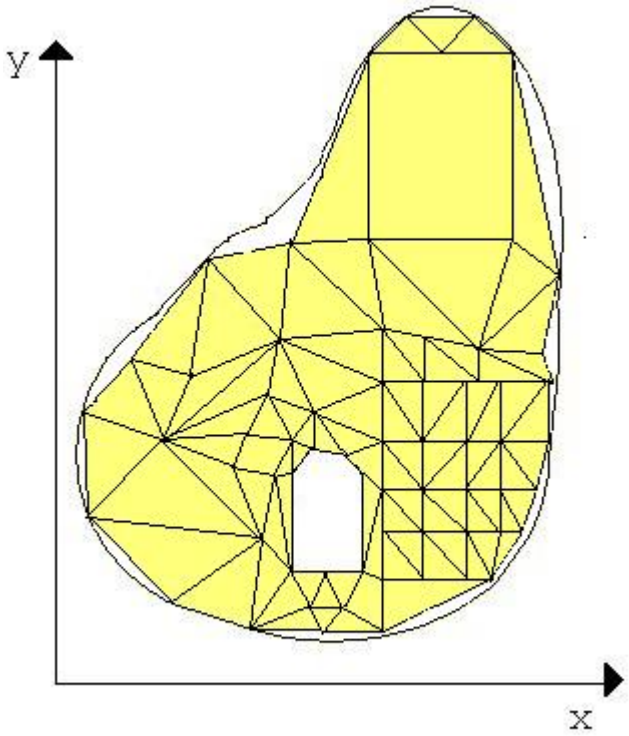


Figura 2.2: Discretización de un dominio bidimensional

4. El refinamiento del mallado debe variar gradualmente desde regiones de alta densidad a regiones de baja densidad. Si se usan elementos de transición, éstos deben ser usados lejos de las regiones críticas (grandes gradientes).

En este trabajo se desarrollará el problema en un dominio en dos dimensiones, que será discretizado en un mallado regular que compondrá elementos triangulares de tres nodos. La forma triangular es la más simple y comúnmente usada para dominios bidimensionales, por lo que se explicará más detalladamente durante esta sección sobre su aplicación y configuración, más específicamente, aquellos que generan polinomios de primer grado.

2.2.2 Construcción de las funciones de forma

El objetivo principal del método de elementos finitos consiste en determinar una colección de funciones desconocidas denominadas variables dependientes, que satisfacen un conjunto dado de ecuaciones diferenciales definidas sobre cierto número de elementos finitos que aproximan un dominio o región, sujetas, además a una serie de condiciones en la frontera de dicho dominio. Estas

funciones incógnitas son representadas a través de una combinación lineal de entre unas funciones de interpolación y los valores nodales o coeficientes cuya solución se halla a través de un sistema de ecuaciones lineales.

Sobre cada elemento finito el conjunto de funciones de interpolación definidas representa lo que se conoce como el espacio de $funciones\ de\ forma$ que generalmente consisten en una colección de polinomios algebraicos de grado específico obtenidos a partir de la teoría de interpolación. Cada función de forma depende no sólo de la geometría de un elemento sino también de la cantidad de nodos que hay en él y su posición. Estos factores inciden en el método de interpolación a aplicar y por lo tanto en el grado del polinomio que será la función de forma.

Sea sobre un dominio Ω de N puntos un elemento e compuesto por n_e nodos, se escogerán para ese elemento n_e funciones de forma ϕ_j que cumplan:

$$\phi_j(x_k) = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases} \tag{2.2.1}$$

donde x_k es un nodo que compone al elemento mencionado y $j, k = 1, 2, \ldots, n_e$.

Así, dado un sistema de ecuaciones diferenciales $\mathbf{Au} = \mathbf{f}$ definida en la expresión (2.1.4) sobre un dominio Ω y sujeta a la condición de frontera $\mathbf{Bu} = \mathbf{g}$ sobre Γ el método consiste en conseguir una aproximación $\widetilde{\mathbf{u}}_i$

$$\widetilde{\mathbf{u}}^e(x) = \sum_{j=1}^{n_e} \phi_j^e(x)\mathbf{c}_j^e \tag{2.2.2}$$

en donde ϕ_j son las llamadas funciones de forma pertenecientes a un espacio generado y c_j es el vector de coeficientes incógnitas a ser determinados igual a

$$\mathbf{c}_j = \begin{bmatrix} c_{1j} \\ c_{2j} \\ \vdots \\ c_{mj} \end{bmatrix} \tag{2.2.3}$$

y c_{ij} el el valor nodal en el nodo j para la incógnita i.

Las funciones de forma sólo pueden reproducir exactamente variaciones polinómicas de grado igual o inferior al del polinomio completo de mayor grado contenido en dichas funciones.

La aproximación $\widetilde{\mathbf{u}}_i^e(x)$ en un elemento finito Ω_e debe satisfacer las siguientes condiciones para que la solución aproximada pueda converger:

1. $\widetilde{\mathbf{u}}_i^e(x)$ debe ser diferenciable, como se requiere en la forma débil del problema (ver sección 2.2.3).

2. Los polinomios usados para representar $\widetilde{\mathbf{u}}_i^e(x)$ deben ser completos, es decir, todos los términos, comenzando con el término independiente hasta el de orden más alto usado en el polinomio, deben ser incluidos en $\widetilde{\mathbf{u}}_i^e(x)$.

3. Todos los términos en el polinomio deben ser linealmente independientes.

En 2D, un polinomio de grado ng, puede escribirse como:

$$f(x,y) = \sum_{i=1}^{p} \alpha_i x^j y^k, \quad j+k \leq ng, \quad j,k = 0,\ldots,ng \tag{2.2.4}$$

donde el número de términos del polinomio es

$$p = \frac{(ng+1)(ng+2)}{2} \tag{2.2.5}$$

Las funciones de forma que contienen términos de polinomios incompletos generan valores nodales que no contribuyen, en términos del orden, en aumentar la precisión de la aproximación del elemento. Por esto, se expondrán sólo los elementos polinómicos triangulares, que son aquellos que, según su orden, generan polinomios de aproximación completos.

2.2.3 Forma integral o débil del sistema de ecuaciones diferenciales

El método de elementos finitos está basado en la formulación integral de las ecuaciones. Con el fin de garantizar que el número de ecuaciones e incógnitas sea el mismo se emplea la formulación $integral\ pesada$ o $ponderada$ del error del sistema de ecuaciones diferenciales igualada a cero.

Dado un sistema de ecuaciones escrita como

$$\mathbf{Au} = \mathbf{f} \quad \text{en } \Omega \tag{2.2.6}$$

bajo las condiciones de frontera en

$$\mathbf{Bu} = \mathbf{g} \quad \text{en } \Gamma \tag{2.2.7}$$

se definen los residuos por cada expresión

$$\begin{aligned} \mathcal{R}_\Omega(\mathbf{u}) &= \mathbf{A}\mathbf{u} - \mathbf{f} \\ \mathcal{R}_\Gamma(\mathbf{u}) &= \mathbf{B}\mathbf{u} - \mathbf{g}. \end{aligned} \tag{2.2.8}$$

En el caso de ser $\mathbf{u}$ una solución exacta de (2.2.6) los residuales se anulan. Pero por ser FEM un método de aproximación, se considera una solución aproximada $\mathbf{u} \cong \widetilde{\mathbf{u}}$ descrita como (2.2.2) se tiene también $\mathcal{R}_\Omega(\widetilde{\mathbf{u}}) \cong 0$ y $\mathcal{R}_\Gamma(\widetilde{\mathbf{u}}) \cong 0$, por lo que el valor de los residuos indica el error que se comete en el cumplimiento de la ecuación diferencial al escoger la solución aproximada $\widetilde{\mathbf{u}}$.

Se toma un conjunto de funciones de peso $\mathbf{\Psi}, \mathbf{Z}$, definidas según el método a utilizar para integrar su producto con los residuales, igualarlo a cero y obtener lo que se denomina la expresión *integral pesada* o *residual ponderada*

$$(\mathcal{R}_\Omega(\widetilde{\mathbf{u}}), \mathbf{\Psi})_\Omega + \langle \mathcal{R}_\Gamma(\widetilde{\mathbf{u}}), \mathbf{Z} \rangle_\Gamma = 0 \tag{2.2.9}$$

es decir,

$$(\mathbf{A}\mathbf{u} - \mathbf{f}, \mathbf{\Psi})_\Omega + \langle \mathbf{B}\mathbf{u} - \mathbf{g}, \mathbf{Z} \rangle_\Gamma = 0 \tag{2.2.10}$$

donde los productos internos se definen como:

$$(\mathcal{R}_\Omega(\widetilde{\mathbf{u}}), \mathbf{\Psi})_\Omega = \int_\Omega \mathbf{\Psi}^T \mathcal{R}_\Omega(\widetilde{\mathbf{u}}) d\Omega \tag{2.2.11}$$

$$\langle \mathcal{R}_\Gamma(\widetilde{\mathbf{u}}), \mathbf{Z} \rangle_\Gamma = \oint_\Gamma \mathbf{Z}^T \mathcal{R}_\Gamma(\widetilde{\mathbf{u}}) d\Gamma \tag{2.2.12}$$

Se requiere que las funciones de peso sean no nulas, ya que así la expresión (2.2.9) se cumple cuando se anulan los residuales, por lo tanto el sistema de ecuaciones (2.2.6) se satisface.

Si la diferenciación se distribuye entre la solución aproximada $\widetilde{\mathbf{u}}$ y las funciones de peso $\mathbf{\Psi}, \mathbf{Z}$, la forma integral resultante requerirá condiciones de continuidad menos fuertes sobre ψ_j , por lo que esta forma integral se conoce como forma débil. Como se verá, la formulación débil tiene las características deseables: primero, requiere condiciones de continuidad más débiles de la variable dependiente, y frecuentemente resultan sistemas de ecuaciones simétricos. Segundo, las condiciones naturales de frontera del problema son incluidas en la forma débil, por lo que la solución aproximada $\widetilde{\mathbf{u}}$ sólo necesita satisfacer las condiciones esenciales de frontera.

2.2.4 Ensamblaje de los elementos

La subdivisión del dominio es la base fundamental del método de elementos finitos. A través de esta técnica la tarea se hace muchísimo más simple. No sólo se crea una independencia absoluta de la forma del dominio, sino que cada subproblema originado posee una extrema facilidad de ser resuelto.

La solución aproximada del problema consiste en unir las propiedades de los elementos de una manera significativa. Esto es llamado *ensamblaje* de los elementos. Está basado en la idea de que el dominio total es aproximado a la suma de los elementos individuales.

Así pues, la ecuación (2.2.10) , que representa un funcional operado sobre todo el dominio Ω se escribe como la suma de su aplicación sobre cada elemento Ω_e

$$\sum_{e=1}^{E}\left\{\int_{\Omega_e}\mathbf{\Psi}^T(\mathbf{Au})d\Omega_e-\int_{\Omega_e}\mathbf{\Psi}^T\mathbf{f}d\Omega_e+\oint_{\Gamma_e}\mathbf{Z}^T(\mathbf{Bu})d\Gamma_e-\oint_{\Gamma_e}\mathbf{Z}^T\mathbf{g}d\Gamma_e\right\}=0. \tag{2.2.13}$$

Aplicando la aproximación (2.2.2) a esta última ecuación se reduce en extremo el procesamiento de nodos, ya que las operaciones realizadas por elemento sólo son necesarias en nodos pertenecientes a éste, mientras que en los demás se tienen asegurados valores nulos. Una vez conocida esta peculiaridad, la suma de las integrales se realiza de manera controlada únicamente para los nodos internos de cada elemento.

$$\sum_{e=1}^{E}\sum_{j=1}^{N}\left\{\int_{\Omega_e}(\mathbf{\Psi}_i)^T\mathbf{A}\phi_j d\Omega_e+\oint_{\Gamma_e}(\mathbf{Z}_i)^T\mathbf{B}\phi_j d\Gamma_e\right\}\mathbf{c_j}=\sum_{e=1}^{E}\left\{\int_{\Omega_e}(\mathbf{\Psi}_i)^T\mathbf{f}d\Omega_e+\oint_{\Gamma_e}(\mathbf{Z}_i)^T\mathbf{g}d\Gamma_e\right\} \tag{2.2.14}$$

Esta última expresión representa una parte generalizada de un sistema de ecuaciones global, es decir, se aplica para cada nodo i. Esto da como resultado un sistema de N expresiones. Pero los operadores internos a cada expresión comprenden subsistemas correspondientes a las m variables en el sistema de ecuaciones diferenciales. Por eso, se han de escoger m funciones de peso para cada variable k por cada nodo i. De este modo se generan $N \times m$ expresiones algebraicas que generan un sistema de ecuaciones lineales bien determinado.

$$\sum_{e=1}^{E}\sum_{j=1}^{N}\left\{\int_{\Omega_e}(\mathbf{\Psi}_{ik})^T\mathbf{A}\phi_j d\Omega_e+\oint_{\Gamma_e}(\mathbf{Z}_{ik})^T\mathbf{B}\phi_j d\Gamma_e\right\}\mathbf{c}_j=\sum_{e=1}^{E}\int_{\Omega_e}(\mathbf{\Psi}_{ik})^T\mathbf{f}d\Omega_e+\oint_{\Gamma_e}(\mathbf{Z}_{ik})^T\mathbf{g}d\Gamma_e \tag{2.2.15}$$

El sistema de ecuaciones lineales se escribe como:

$$\mathbf{K}\mathbf{c} = \mathbf{Q} \tag{2.2.16}$$

donde

$$\begin{aligned} \mathbf{K} &= \sum_{e=1}^{E} \mathbf{K}^e \\ \mathbf{Q} &= \sum_{e=1}^{E} \mathbf{q}^e \end{aligned} \tag{2.2.17}$$

y $\mathbf{c}$ es el vector incógnita de valores nodales.

Para conseguir este sistema se rescribe la expresión (2.2.14) de una forma computacionalmente práctica usando la notación $\mathbf{K}_{ikjl}$ como el elemento en la fila $(i-1)m+k$ y columna $(j-1)m+l$ de la matriz $\mathbf{K}$ y $\mathbf{q}_{ik}$ como el elemento en la posición $(i-1)m+k$ del vector $\mathbf{q}$ y $i = 1, \ldots, N; k = 1, \ldots, m$:

$$\sum_{j=1}^{N}\sum_{l=1}^{m} \mathbf{K}^e_{ikjl} c_{jl} = \mathbf{q}^e_{ik} \tag{2.2.18}$$

en donde

$$\mathbf{K}^e_{ikjl} = \int_{\Omega_e} (\mathbf{\Psi}_{ik})^T \mathbf{A}_l \phi_j d\Omega_e + \oint_{\Gamma_e} (\mathbf{Z}_{ik})^T \mathbf{B}_l \phi_j d\Gamma_e \tag{2.2.19}$$

$$\mathbf{q}^e_{ik} = \int_{\Omega_e} (\mathbf{\Psi}_{ik})^T \mathbf{f}_i d\Omega_e + \oint_{\Gamma_e} (\mathbf{Z}_{ik})^T \mathbf{g}_i d_{\Gamma_e} \tag{2.2.20}$$

Aquí $\mathbf{A}_l$, y similarmente $\mathbf{B}_l$, acorde con su definición (2.1.3) representa un operador sobre un vector incógnita $\mathbf{u}$ tal que:

$$\mathbf{A}_l \mathbf{u} = \sum_{k=1}^{nd} a_{il,k} \frac{\partial u_l}{\partial x_k} + a_{il} u_l \quad i = 1, 2, \ldots, m. \tag{2.2.21}$$

Después de sustituir las funciones de peso $\mathbf{\Psi}_{ik}, \mathbf{Z}_{ik}$ la simplificación de la expresión (2.2.18) llega a ser un proceso de llenado de una matriz elemental, compuesta por una suma de factores en los que se involucran los coeficientes de los operadores $\mathbf{A}, \mathbf{B}$ y las integrales de las funciones de forma simples o derivadas con respecto a cada eje según el número de dimensiones representadas en el problema.

Es decir, para cada integral en (2.2.19) y (2.2.21) existe una representación tal se puede escribir como una suma de productos entre un coeficiente y una integral definida según las funciones de prueba y la forma del elemento.

La solución exacta de las integrales de los elementos de la matriz $\mathbf{K}^e$ y $\mathbf{f}^e$ en (2.2.19) y (2.2.20) no siempre puede darse fácilmente, por lo que deben utilizarse técnicas de integración numéricas; comúnmente, se utiliza la cuadratura de Gauss - Legendre. Sin embargo, cuando los coeficientes de $\mathbf{A}, \mathbf{B}$ y $\mathbf{f}, \mathbf{g}$ son constantes en el elemento, es posible evaluar las integrales exactamente. Las integrales de frontera en q^e de (2.2.20) pueden ser evaluadas si se conoce a $\mathbf{g}$. Para aquellos elementos internos, esto es que ninguno de sus lados coincida con la frontera del problema, la contribución de la integral de frontera se cancela con una contribución igual del lado que comparte con el elemento adyacente en la malla.

2.2.5 Métodos de Elementos Finitos

Como fue dislumbrado anteriormente, existen varios métodos utilizados conjuntamente con el de elementos finitos para aproximar $\widetilde{\mathbf{u}}$ que difieren unos de otros en términos de la selección de las funciones de peso $\mathbf{\Psi}$ y $\mathbf{Z}$, así como de las funciones de prueba o aproximación ϕ_j . Algunos de estos métodos se describen a continuación.

- **Método de colocación por puntos**. Este método consiste el la selección de N puntos arbitrarios x_j sobre el dominio Ω, así como la selección de la función Delta de Dirac, i.e., $\delta(x - x_j)$ para $j = 1, 2, \ldots, N$, como funciones de peso. El resultado es un sistema de N ecuaciones lineales con N incógnitas representadas por las cantidades desconocidas c_j , cuya solución, al ser sustituida en la expresión (2.2.2) , permite obtener una aproximación de la variable dependiente u que satisfaga la ecuación diferencial.

- **Método de colocación por subdominios**. Este método consiste en la división del dominio Ω, en N subdominios Ω_j para $j = 1, 2, \ldots, N$, seleccionando las funciones de peso según la siguiente relación

$$\mathbf{\Psi}(\mathbf{x}) = \mathbf{Z}(\mathbf{x}) = \begin{cases} 1 & \text{si } x \in \Omega_j \\ 0 & \text{en cualquier otro caso} \end{cases}$$

- **Método de Galerkin**. Este método consiste en seleccionar las funciones de peso $\mathbf{\Psi}$ y $\mathbf{Z}$ para que sean iguales a las funciones de prueba, i.e., $\Psi_i(x) = Z_i(x) \equiv \psi_j$. Es práctica común seleccionar las funciones de prueba de manera tal que satisfagan las condiciones de frontera de la ecuación diferencial, con lo que el producto interno en (2.2.12) se anula, i.e., se hace cero.

- **Método de mínimos cuadrados**. Este método se basa fundamentalmente en la definición de una integral de la forma

$$I = (\mathcal{R}_\Omega, \mathcal{R}_\Omega)_\Omega + \langle \mathcal{R}_\Gamma, \mathcal{R}_\Gamma \rangle_\Gamma \tag{2.2.22}$$

en donde los productos internos siguen la misma definición que en (2.2.11) . De esta manera, el problema de resolver (2.2.9) se reduce al problema de minimizar la expresión formulada en (2.2.22) en términos de la condición necesaria $\delta I = 0$, con lo que se obtiene un sistema de ecuaciones lineales dado por

$$\frac{\partial}{\partial \mathbf{c}_j} \left[(\mathcal{R}_\Omega, \mathcal{R}_\Omega)_\Omega + \langle \mathcal{R}_\Gamma, \mathcal{R}_\Gamma \rangle_\Gamma \right] = 0 \text{ para } j = 1, 2, \ldots, N. \tag{2.2.23}$$

Debe recordarse el hecho de que los residuos $\mathcal{R}_\Omega$ y $\mathcal{R}_\Gamma$ dependen de la aproximación $\widehat{\mathbf{u}}$ y que ésta, a su vez, depende de las cantidades desconocidas $\mathbf{c}_j$ para $j = 1, 2, \ldots, N$.

En la siguiente sección se profundizará acerca de este último método, que hasta la fecha ha demostrado producir excelentes resultados.

2.3 Método Elementos Finitos Mínimos Cuadrados

Durante las últimas tres décadas la investigación en elementos finitos se ha dedicado a la mejora del método Galerkin. Su generalidad y versatilidad le dio un fuerte ímpetu para su utilización en campos como la mecánica de fluidos. Sin embargo, en la práctica, las soluciones provistas por el método Galerkin, en problemas de transporte dominado por convección, son corrompidas por oscilaciones inexactas que sólo pueden ser resueltas a través de refinamientos severos en el mallado. También se comporta pobremente para problemas de flujo compresible de alta velocidad y de hondas de agua, que están gobernados por ecuaciones hiperbólicas de primer orden no lineales, cuya solución puede ser discontinua. Pero donde más dificultades trae el método es en problemas elípticos de segundo orden, en los que la situación se complica en varios aspectos y hace de éste método decepcionante.

El método Mínimos Cuadrados o LSFEM (por Least Squares Finite Elements Method), por su parte, ha sido considerado por algunos investigadores como el mejor método para sistemas no auto-adjuntos, como de mecánica de fluidos y electromagnética. A pesar de simpleza ofrece méritos significantes.

- **Universalidad**. LSFEM ha unificado la formulación para la solución numérica de cualquier tipo de ecuaciones diferenciales parciales. No importa si son ecuaciones elípticas,

parabólicas, hiperbólicas o mixtas, mientras las ecuaciones tengan una solución única, este método siempre puede determinar una buena solución aproximada. Por eso, LSFEM puede simular problemas de dinámica de fluidos en cualquier régimen, desde subsónico hasta transónico, supersónico e hipersónico.

- **Eficiencia**. En muchas áreas de la ingeniería y ciencia aplicada las ecuaciones diferenciales parciales gobernantes originales obtenidas a partir de las leyes físicas son o pueden ser transformadas a unas de primer orden. LSFEM maneja de forma natural los operadores diferenciales de primer orden, ya que siempre produce matrices simétricas y definidas positivas que son resueltas eficientemente a través de métodos iterativos.

- **Robustez**. Con LSFEM no se realizan todos los artificios necesarios en los otros métodos, como disipación artificial, precondicionamiento de operadores o fraccionamiento de operadores. En problemas de convección, por ejemplo, LSFEM puede capturar discontinuidades en los elementos automáticamente.

- **Optimalidad**. En muchos casos ha sido probado rigurosamente que la solución de LSFEM es la mejor aproximación, es decir, que el error de la solución tiene el mismo orden que el error de la interpolación.

- **Código de múltiple uso**. LSFEM es una formulación muy general. Puede ser programado sistemáticamente en un único algoritmo para resolver diferentes ecuaciones en varios campos de la ciencia. Esto reduce el tiempo, costo y los errores de programación en el desarrollo del código.

2.3.1 Forma integral o débil

Para la solución del sistema de ecuaciones (2.2.6) bajo las condiciones de frontera (2.2.7) se procede a definir unas funciones de peso $\mathbf{\Psi}$ con $\mathbf{\Psi}_i \in \text{Gen}\{\phi_1, \phi_2, ..., \phi_n\}$. Para un sistema de ecuaciones de múltiples incógnitas en un dominio real la generación de funciones de peso suele realizarse así:

$$\Psi_{ik} = \begin{cases} \phi_i & \text{si} \quad i = k \\ 0 & \text{si} \quad i \neq k \end{cases} \tag{2.3.1}$$

De acuerdo al método de elementos finitos mínimos cuadrados, explicado por Rojas [34], resolver la ecuación (2.2.6) sujeta a (2.2.7) equivale a minimizar el funcional cuadrático

$$I(\mathbf{u}) = \frac{1}{2}\left\{\|\mathbf{Au} - \mathbf{f}\|_\Omega^2 + \|\mathbf{Bu} - \mathbf{g}\|_\Gamma^2\right\} \tag{2.3.2}$$

para lo cual es necesario que

$$\lim_{t\to 0}\frac{d\mathbf{I}\left(\mathbf{u}+t\mathbf{\Psi}\right)}{dt}=0 \tag{2.3.3}$$

Se desarrolla la expresión $I\left(\mathbf{u}+t\mathbf{\Psi}\right)$:

$$\begin{aligned}
I\left(\mathbf{u}+t\mathbf{\Psi}\right)=&\frac{1}{2}\{\|\mathbf{A}\left(\mathbf{u}+t\mathbf{\Psi}\right)-\mathbf{f}\|_{\Omega}^{2}+\|\mathbf{B}\left(\mathbf{u}+t\mathbf{\Psi}\right)-\mathbf{g}\|_{\Gamma}^{2}\}\\
=&\frac{1}{2}\{\left(\mathbf{A}\left(\mathbf{u}+\mathbf{t\Psi}\right)-\mathbf{f},\mathbf{A}\left(\mathbf{u}+\mathbf{t\Psi}\right)-\mathbf{f}\right)_{\Omega}\\
&+\left\langle\mathbf{B}\left(\mathbf{u}+t\mathbf{\Psi}\right)-g,\mathbf{B}\left(u+t\mathbf{\Psi}\right)-g\right\rangle_{\Gamma}\}\\
=&\frac{1}{2}\{\left(\mathbf{Au},\mathbf{Au}\right)_{\Omega}+t\left(\mathbf{Au},\mathbf{A\Psi}\right)_{\Omega}+t\left(\mathbf{A\Psi},\mathbf{Au}\right)_{\Omega}\\
&+t^{2}\left(\mathbf{A\Psi},\mathbf{A\Psi}\right)_{\Omega}-\left(\mathbf{Au},\mathbf{f}\right)_{\Omega}-t\left(\mathbf{A\Psi},\mathbf{f}\right)_{\Omega}\\
&-\left(\mathbf{f},\mathbf{Au}\right)_{\Omega}-t\left(\mathbf{f},\mathbf{A\Psi}\right)_{\Omega}-\left(\mathbf{f},\mathbf{f}\right)_{\Omega}\\
&+\left\langle\mathbf{Bu},\mathbf{Bu}\right\rangle_{\Gamma}+t\left\langle\mathbf{Bu},\mathbf{B\Psi}\right\rangle_{\Gamma}+t\left\langle\mathbf{B\Psi},\mathbf{Bu}\right\rangle_{\Gamma}\\
&+t^{2}\left\langle\mathbf{B\Psi},\mathbf{B\Psi}\right\rangle_{\Gamma}-\left\langle\mathbf{Bu},\mathbf{g}\right\rangle_{\Gamma}-t\left\langle\mathbf{B\Psi},\mathbf{g}\right\rangle_{\Gamma}\\
&-\left\langle\mathbf{g},\mathbf{Bu}\right\rangle_{\Gamma}-t\left\langle\mathbf{g},\mathbf{B\Psi}\right\rangle_{\Gamma}-\left\langle\mathbf{g},\mathbf{g}\right\rangle_{\Gamma}
\end{aligned} \tag{2.3.4}$$

se deriva con respecto a t para obtener

$$\begin{aligned}
\frac{d\mathbf{I}\left(\mathbf{u}+t\mathbf{\Psi}\right)}{dt}=&\left(\mathbf{Au},\mathbf{A\Psi}\right)_{\Omega}+\left(\mathbf{A\Psi},\mathbf{Au}\right)_{\Omega}+2t\left(\mathbf{A\Psi},\mathbf{A\Psi}\right)_{\Omega}\\
&-\left(\mathbf{A\Psi},\mathbf{f}\right)_{\Omega}-\left(\mathbf{f},\mathbf{A\Psi}\right)_{\Omega}+\left\langle\mathbf{Bu},\mathbf{B\Psi}\right\rangle_{\Gamma}\\
&+\left\langle\mathbf{B\Psi},\mathbf{Bu}\right\rangle_{\Gamma}+2t\left\langle\mathbf{B\Psi},\mathbf{B\Psi}\right\rangle_{\Gamma}-\left\langle\mathbf{B\Psi},\mathbf{g}\right\rangle_{\Gamma}-\left\langle\mathbf{g},\mathbf{B\Psi}\right\rangle_{\Gamma}
\end{aligned} \tag{2.3.5}$$

tomando el limite cuanto t tiende a cero se tiene

$$\begin{aligned}
\lim_{t\to 0}\frac{d\mathbf{I}\left(\mathbf{u}+t\mathbf{\Psi}\right)}{dt}=&\left(\mathbf{Au},\mathbf{A\Psi}\right)_{\Omega}+\left(\mathbf{A\Psi},\mathbf{Au}\right)_{\Omega}-\left(\mathbf{A\Psi},\mathbf{f}\right)_{\Omega}-\left(\mathbf{f},\mathbf{A\Psi}\right)_{\Omega}\\
&+\left\langle\mathbf{Bu},\mathbf{B\Psi}\right\rangle_{\Gamma}+\left\langle\mathbf{B\Psi},\mathbf{Bu}\right\rangle_{\Gamma}+\left\langle\mathbf{B\Psi},\mathbf{g}\right\rangle_{\Gamma}-\left\langle\mathbf{g},\mathbf{B\Psi}\right\rangle_{\Gamma}
\end{aligned} \tag{2.3.6}$$

Aplicando las propiedades del producto interno se escribe

$$\left(\mathbf{Au}-\mathbf{f},\mathbf{A\Psi}\right)_{\Omega}+\left(\mathbf{A\Psi},\mathbf{Au}-\mathbf{f}\right)_{\Omega}+\left\langle\mathbf{Bu}-\mathbf{g},\mathbf{B\Psi}\right\rangle_{\Gamma}+\left\langle\mathbf{B\Psi},\mathbf{Bu}-\mathbf{g}\right\rangle_{\Gamma}=0 \tag{2.3.7}$$

Por simetría del producto interno para valores reales

$$2\left(\mathbf{Au}-\mathbf{f},\mathbf{A\Psi}\right)_{\Omega}+2\left\langle\mathbf{Bu}-\mathbf{g},\mathbf{B\Psi}\right\rangle_{\Gamma}=0 \tag{2.3.8}$$

y finalmente

$$\left(\mathbf{Au}-\mathbf{f},\mathbf{A\Psi}\right)_{\Omega}+\left\langle\mathbf{Bu}-\mathbf{g},\mathbf{B\Psi}\right\rangle_{\Gamma}=0 \tag{2.3.9}$$

La última expresión ((2.3.9)) se denomina $forma\ variacional$ o $forma\ débil$ del sistema de

ecuaciones diferenciales. Mediante el desarrollo algebraico de la aproximación se obtiene el sistema de ecuaciones lineales mencionado, cuya solución permite determinar los valores de las cantidades desconocidas c_{ij} para i = 1,2, ..., m y j = 1,2, ..., N. Los valores c_{ij} se determinan de manera tal que la solución aproximada $\widetilde{\mathbf{u}}$ satisfaga la forma débil del sistema de ecuaciones diferenciales gobernante o que minimice el funcional cuadrático asociado con el sistema en estudio.

Como $(\mathbf{Au}-\mathbf{f},\mathbf{A\Psi})_\Omega = (\mathbf{Au},\mathbf{A\Psi})_\Omega - (\mathbf{f},\mathbf{A\Psi})_\Omega$ y por (2.2.11) se desarrollan

$$(\mathbf{Au},\mathbf{A\Psi})_\Omega = \int_\Omega (\mathbf{A\Psi})^T \mathbf{Au}d\Omega \tag{2.3.10}$$

$$(\mathbf{f},\mathbf{A\Psi})_\Omega = \int_\Omega (\mathbf{A\Psi})^T \mathbf{f}d\Omega \tag{2.3.11}$$

Como también $\langle \mathbf{Bu}-\mathbf{g},\mathbf{B\Psi}\rangle_\Gamma = \langle \mathbf{Bu},\mathbf{B\Psi}\rangle_\Gamma - \langle \mathbf{g},\mathbf{A\Psi}\rangle_\Gamma$ se definen similarmente por (2.2.12)

$$\langle \mathbf{Bu},\mathbf{B\Psi}\rangle_\Gamma = \oint_\Gamma (\mathbf{B\Psi})^T \mathbf{Bu}d\Gamma \tag{2.3.12}$$

$$\langle \mathbf{g},\mathbf{B\Psi}\rangle_\Gamma = \oint_\Omega (\mathbf{B\Psi})^T \mathbf{g}d\Gamma \tag{2.3.13}$$

Así la ecuación (2.3.9) queda escrita como:

$$\int_\Omega (\mathbf{A\Psi})^T \mathbf{Au}d\Omega + \oint_\Gamma (\mathbf{B\Psi})^T \mathbf{Bu}d\Gamma = \int_\Omega (\mathbf{A\Psi})^T \mathbf{f}d\Omega + \oint_\Omega (\mathbf{B\Psi})^T \mathbf{g}d\Gamma \tag{2.3.14}$$

2.3.2 Ensamblaje de los elementos en LSFEM

El objetivo general del método es obtener, a partir de las ecuaciones diferenciales en (2.2.6) condicionado a (2.2.7) un sistema de ecuaciones lineales (2.2.16) con (2.2.17) . Para lograr esto a través de LSFEM generalizamos la ecuación (2.3.14) a cada elemento Ω_e, y se aproxima $\mathbf{u}$ según (2.2.2) .

$$\sum_{e=1}^{E}\sum_{j=1}^{N}\left\{\int_{\Omega_e}(\mathbf{A\Psi}_i)^T\mathbf{A}\phi_j d\Omega_e + \oint_{\Gamma_e}(\mathbf{B\Psi}_i)^T\mathbf{B}\phi_j d\Gamma_e\right\}\mathbf{c}_j = \sum_{e=1}^{E}\left\{\int_{\Omega_e}(\mathbf{A\Psi}_i)^T\mathbf{f}d\Omega_e + \oint_{\Gamma_e}(\mathbf{B\Psi}_i)^T\mathbf{g}d\Gamma_e\right\} \tag{2.3.15}$$

Y al aplicar (2.3.1) en (2.3.15) tenemos para $i = 1,\ldots,N; k = 1,\ldots,m$:

$$\sum_{e=1}^{E}\sum_{j=1}^{N}\left\{\int_{\Omega_e}\left(\mathbf{A}_k\phi_i\right)^T\mathbf{A}\phi_j d\Omega_e+\oint_{\Gamma_e}\left(\mathbf{B}_k\phi_i\right)^T\mathbf{B}\phi_j d\Gamma_e\right\}\mathbf{c}_j=\sum_{e=1}^{E}\left\{\int_{\Omega_e}\left(\mathbf{A}_k\phi_i\right)^T\mathbf{f}d\Omega_e+\oint_{\Gamma_e}\left(\mathbf{B}_k\phi_i\right)^T\mathbf{g}d\Gamma_e\right\} \tag{2.3.16}$$

Así, llegamos a la expresión (2.2.18) en donde:

$$\mathbf{K}^e_{ikjl} = \int_{\Omega_e}\left(\mathbf{A}_k\phi_i\right)^T\mathbf{A}_l\phi_j d\Omega_e+\oint_{\Gamma_e}\left(\mathbf{B}_k\phi_i\right)^T\mathbf{B}_l\phi_j d\Gamma_e \tag{2.3.17}$$

$$\mathbf{q}^e_{ik} = \int_{\Omega_e}\left(\mathbf{A}_k\phi_i\right)^T\mathbf{f}_i d\Omega_e+\oint_{\Gamma_e}\left(\mathbf{B}_k\phi_i\right)^T\mathbf{g}_i d\Gamma_e \tag{2.3.18}$$

2.4 Elementos Triangulares

En esta sección se examinará una de las formas particulares de elementos más comunes que se usa en dos dimensiones.

2.4.1 Triangulación de Delaunay-Voronoï

Tradicionalmente para los métodos de elementos finitos se utiliza un mallado que optimice la aproximación de la interpolación a través del cumplimiento de ciertas condiciones geométricas requeridas para ello.

La triangulación de *Delaunay* provee estas condiciones, además de un poderoso método de generación de mallado no estructurado, que por su flexibilidad y características es apto para un trabajo como el presente.

La generación de mallado basado en la triangulación de *Delaunay* usa un criterio particularmente simple para conectar nodos para formar elementos que no se intersectan. Esta construcción geométrica ha sido conocida por varios años, pero sólo recientemente ha sido usada para simulación computacional de fluidos.

En general, la triangulación de *Delaunay* es, de entre todas las triangulaciones posibles, aquella tal que el menor ángulo definido sea máximo, es decir que los triángulos sean lo más equiláteros

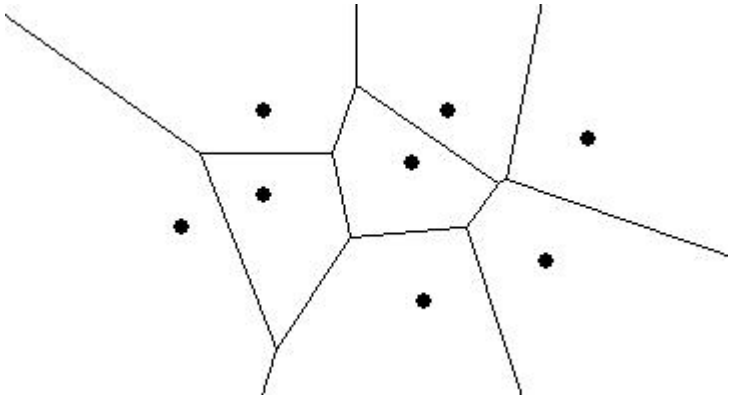


Figura 2.3: Diagrama de *Voronoï*

posibles. Dicha Triangulación puede ser resuelta mediante el diagrama de *Voronoï*, ya que la Triangulación de *Delaunay* es el dual de éste.

Decimos que un Diagrama de *Voronoï* de una nube de puntos, se realiza basándose fundamentalmente en la proximidad. Así, supuesto dado un conjunto finito de puntos $S = s_1, \ldots, s_n$, con un número de elementos no menor que dos. Si a cada s_j le asociamos aquellos puntos del plano que están más cerca de él que de cualquier otro de los p_i, todo punto del plano queda asociado a algún p_i, formándose conjuntos que recubren a estos puntos que forman la nube original. De esta manera existirán puntos que disten lo mismo de dos elementos de S y que formarán la frontera de cada región.

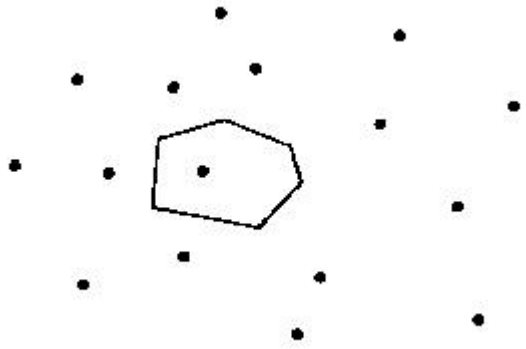


Figura 2.4: Polígono de *Voronoï*

Los conjuntos resultantes forman una repartición del plano, en el sentido de que son exhaustivos (todo punto del plano pertenece a alguno de ellos) y mutuamente excluyentes salvo en su frontera. A esta partición del plano en forma de mosaico a la que se la llama Diagrama de *Voronoï* (fig 2.3). A cada una de las regiones resultantes se las llamará regiones de *Voronoï* o polígonos de *Voronoï* (ver figura 2.4).

Los bordes del diagrama de *Voronoï* están formados por los bisectores perpendiculaes de las línas que conectan los vecinos de los nodos, y cuyo vértice es el circuncentro de un triángulo formado por tres nodos. Esto determina una triangulación única conocida como la triangulación de *Delaunay* (fig 2.5).

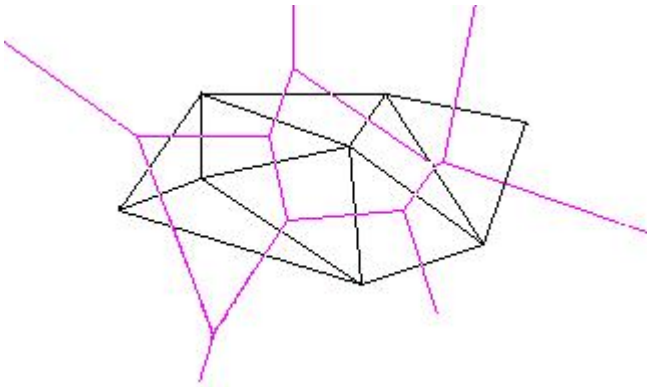


Figura 2.5: Triangulación de *Delaunay*

Según la página en español de *Voronoï*(http://ma1.eii.us.es/miembros/almar/voronoi/teoria.html), la triangulación de *Delaunay* y su dual de *Voronoï* cumplen con las siguientes propiedades:

- Cada vértice del diagrama de *Voronoï* es la intersección común a tres bordes del diagrama.
- Para cada vértice v del diagrama de *Voronoï* de S, la circunferencia $C(v)$ no contiene a ningún otro punto de S. Es decir, la circunferencia formada por cada triángulo no contiene más nodos que los tres que la conforman (ver figura 2.6).

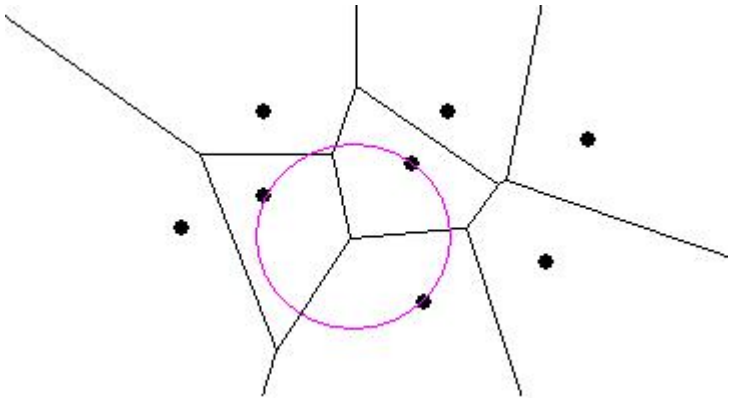


Figura 2.6: Proximidad de puntos

- Cada vecino más cercano de p_i de S define un borde del polígono de *Voronoï* $V(i)$.

- El polígono $V(i)$ es ilimitado si y solo si p_i es un punto de la frontera de la envolvente convexa del conjunto S.
- Un diagrama de *Voronoï* de N puntos tiene como máximo $N-5$ vértices y $3N-6$ bordes.

2.4.2 Funciones de forma lineales

La generación de los polinomios de primer grado en los elementos triangulares puede ser deducida a partir de las condiciones que deben cumplir las funciones de forma y de las propiedades en el plano de esta simple figura geométrica. La construcción de los polinomios triangulares, así como algunas de las integrales escritas en la siguiente sección, es extraída de los trabajos de Reddy [13].

Considérese la aproximación lineal

$$c_i^e(x,y) = s_1 + s_2 x + s_3 y \tag{2.4.1}$$

donde s_1, s_2, s_3 son los coeficientes de los polinomios a utilizar y c es un componente arbitrario del vector $\mathbf{c}$ utilizado para aproximar el componente arbitrario correspondiente u del vector $\mathbf{u}$.

El conjunto $\{1, x, y\}$ es linealmente independiente y completo. La expresión (2.4.1) debe satisfacer las condiciones

$$u_i^e(x_i^e, y_i^e) = c_i^e, \quad i = 1,2,3 \tag{2.4.2}$$

donde (x_i^e, y_i^e) son las coordenadas de los tres vértices del triángulo Ω^e. Se pueden determinar las tres constantes s_i en (2.4.1) en términos de los c_i^e de (2.4.2) haciendo

$$\begin{aligned} c_1 &= s_1 + s_2 x_1 + s_3 y_1 \\ c_2 &= s_1 + s_2 x_2 + s_3 y_2 \\ c_3 &= s_1 + s_2 x_3 + s_3 y_3 \end{aligned} \tag{2.4.3}$$

donde la etiqueta e se omite por simplicidad. En forma matricial, de (2.4.3) se tiene que

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} \tag{2.4.4}$$

La solución de (2.4.4) para s_i requiere invertir la matriz de coeficientes. La inversa no existe si dos filas o columnas son linealmente dependientes. Esto sucede solo cuando los tres nodos se encuentran sobre una misma recta. Así, en teoría, mientras los vértices sean distintos unos de otros y no estén sobre una misma recta, la matriz de coeficientes será invertible. Sin embargo, para efectos computacionales, si dos vértices están muy cercanos, o los tres nodos están casi sobre la misma línea, la matriz podría no ser numéricamente invertible, por lo que se deben evitar elementos con estas características.

Al invertir la matriz de coeficientes en (2.4.4) se tiene que

$$\mathbf{A}^{-1} = \frac{1}{2A_e}\begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix}, \quad 2A_e = \alpha_1 + \alpha_2 + \alpha_3 \tag{2.4.5}$$

y resolviendo para s_i en términos de c_i

$$\mathbf{s} = \mathbf{A}^{-1}\mathbf{c} \tag{2.4.6}$$

obteniéndose que

$$\begin{aligned} s_1 &= \frac{1}{2A_e}(\alpha_1 c_1 + \alpha_2 c_2 + \alpha_3 c_3) \\ s_2 &= \frac{1}{2A_e}(\beta_1 c_1 + \beta_2 c_2 + \beta_3 c_3) \\ s_3 &= \frac{1}{2A_e}(\gamma_1 c_1 + \gamma_2 c_2 + \gamma_3 c_3) \end{aligned} \tag{2.4.7}$$

donde A_e es el área del triángulo, o bien $2A_e$ es el determinante de $\mathbf{A}$, y α_i, β_i, γ_i son las constantes geométricas

$$\begin{aligned} \alpha_i &= x_j y_k - x_k y_j \\ \beta_i &= y_j - y_k \\ \gamma_i &= x_k - x_j \end{aligned} \tag{2.4.8}$$

con $i \neq j \neq k$, donde i, j y k permutan ciclicamente los valores 1, 2 y 3.

Sustituyendo para s_i con $i = 1,\ 2,\ 3$ de (2.4.7) en (2.4.1) , se tiene

$$\begin{aligned} u^e(x,y) &= \frac{1}{2A_e}[(\alpha_1 u_1 + \alpha_2 u_2 + \alpha_3 u_3) + (\beta_1 u_1 + \beta_2 u_2 + \beta_3 u_3)x \\ &\quad + (\gamma_1 u_1 + \gamma_2 u_2 + \gamma_3 u_3)y] \\ &= \sum_{i=1}^{3} u_i^e \phi_i^e(x,y) \end{aligned} \tag{2.4.9}$$

donde ϕ_i^e son las funciones de interpolación para el elemento triangular lineal, con

$$\phi_i^e = \frac{1}{2A_e}(\alpha_i + \beta_i x + \gamma_i y) \tag{2.4.10}$$

La expresión (2.4.9) determina el plano que pasa por c_1, c_2 y c_3. En la figura 2.7 (creada por MatLab 6.5 http://www.mathworks.com) puede verse la forma de las ϕ_i^e.

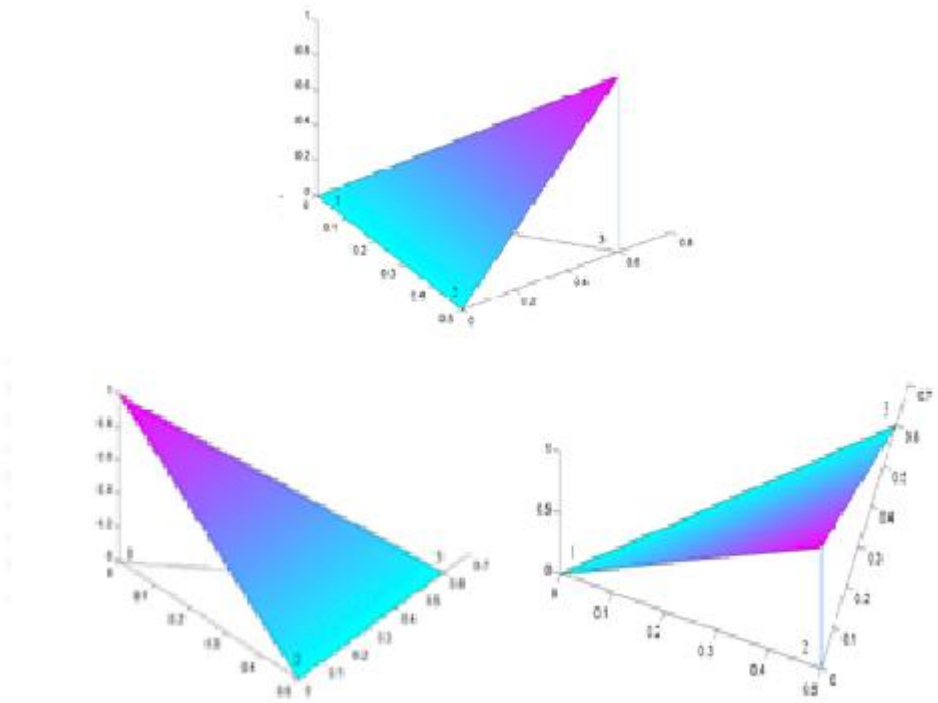


Figura 2.7: Funciones de forma de un elemento triangular lineal.

2.4.3 Integración de las funciones de forma lineales

La evaluación de las integrales hecha para los elementos triangulares en un dominio bidimensional tienen la poco común característica de ser exacta.

Las integrales que se buscarán se escriben como:

$$S_{ij}^{\alpha\beta} = \int_{\Omega^e} \phi_{i,\alpha}^e \phi_{j,\beta}^e d\Omega^e \tag{2.4.11}$$

donde

$$\begin{aligned} \alpha, \beta &= 0, 1, \ldots, nd \\ \phi_{i,\alpha}^e &= \frac{\partial \phi_i^e}{\partial x_\alpha} \\ \phi_{i,0}^e &= \phi_i^e \end{aligned}$$

En un elemento bidimensional triangular lineal, se puede usar la siguiente fórmula para evaluar las integrales. Sea

$$I_{mn} \equiv \int_{\triangle} x^m y^n dx dy \tag{2.4.12}$$

se conocen las siguientes integrales exactas:

$$\begin{aligned} I_{00} =& A_e \\ I_{10} =& A_e \widehat{x} \\ I_{01} =& A_e \widehat{y} \\ I_{11} =& \frac{A_e}{12} \left(\sum_{i=1}^{3} x_i y_i + 9 \widehat{x}\widehat{y} \right) \\ I_{20} =& \frac{A_e}{12} \left(\sum_{i=1}^{3} x_i^2 + 9 \widehat{x}^2 \right) \\ I_{02} =& \frac{A_e}{12} \left(\sum_{i=1}^{3} y_i^2 + 9 \widehat{y}^2 \right) \end{aligned} \tag{2.4.13}$$

donde A_e es el área del triángulo expresado en (2.4.5) y $\alpha_i, \beta_i, \gamma_i$ se escriben como (2.4.8) y

$$\begin{aligned} \widehat{x} &= \frac{1}{3}\sum_{i=1}^{3} x_i \\ \widehat{y} &= \frac{1}{3}\sum_{i=1}^{3} y_i \end{aligned} \tag{2.4.14}$$

Utilizando las funciones de interpolación (2.4.10) en (2.4.11) y teniendo en cuenta que

$$\frac{\partial \phi_i}{\partial x} = \frac{\beta_i}{2A_e}, \frac{\partial \phi_i}{\partial y} = \frac{\gamma_i}{2A_e} \tag{2.4.15}$$

y la identidad $\alpha_i + \beta_i\widehat{x} + \gamma_i\widehat{y} = \frac{2}{3}A$ (originada por (2.4.10) y (2.4.13)) se obtiene

$$\begin{aligned} S_{ij}^{00} =& \frac{1}{4A_e}\Big\{[\alpha_i\alpha_j + (\alpha_i\beta_j + \alpha_j\beta_i)\widehat{x} + (\alpha_i\gamma_j + \alpha_j\gamma_i)\widehat{y}] \\ &+ \frac{1}{A_e}[I_{20}\beta_i\beta_j + I_{11}(\gamma_i\beta_j + \gamma_j\beta_i) + I_{02}\gamma_i\gamma_j]\Big\} \\ S_{ij}^{01} =& \frac{\beta_j}{6} \\ S_{ij}^{02} =& \frac{\gamma_j}{6} \\ S_{ij}^{10} =& \frac{\beta_i}{6} \\ S_{ij}^{11} =& \frac{1}{4A_e}\beta_i\beta_j \\ S_{ij}^{12} =& \frac{1}{4A_e}\beta_i\gamma_j \\ S_{ij}^{20} =& \frac{\gamma_i}{6} \\ S_{ij}^{21} =& \frac{1}{4A_e}\beta_j\gamma_i \\ S_{ij}^{22} =& \frac{1}{4A_e}\gamma_i\gamma_j \end{aligned} \tag{2.4.16}$$

Para la evaluación de (2.2.20) se predefinen

$$\int_{\Omega_e} f\phi_i d\Omega_e = \frac{1}{3}fA_e \tag{2.4.17}$$

$$\int_{\Omega_e} f\frac{\partial \phi_i}{\partial x} d\Omega_e = f\frac{\beta_i}{2} \tag{2.4.18}$$

$$\int_{\Omega_e} f\frac{\partial \phi_i}{\partial y} d\Omega_e = f\frac{\gamma_i}{2} \tag{2.4.19}$$

2.4.4 Integración en la frontera para funciones lineales

Considérese ahora la evaluación de las integrales de frontera Γ_e, cuya deducción involucra también las funciones de formas creadas en (2.4.10) . No es necesario calcular las integrales cuando una porción no coincide con la frontera Γ en el dominio total Ω ya que la parte Γ_e en el interior del dominio Ω se anula con la contribución de la misma porción del elemento adyacente, en lo que se conoce como equilibrio de flujo interno, como condición de ensamblaje de las contribuciones elementales.

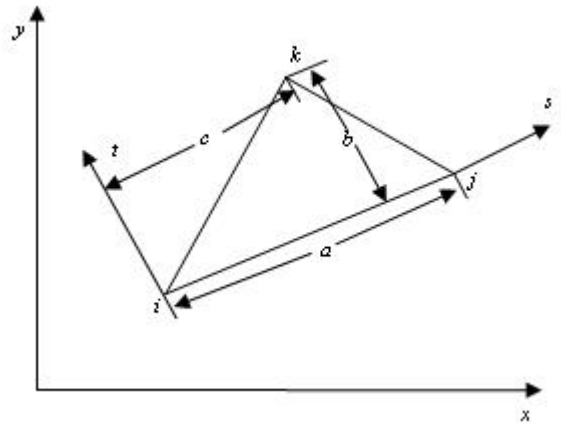


Figura 2.8: Elemento triangular en (x, y) y (s, t)

La frontera Γ_e de los elementos bidimensionales lineales es un conjunto de elementos unidimensionales lineales. Por lo tanto, la evaluación de las integrales de fronteras en dos dimensiones conlleva a evaluar integrales sobre líneas. Trabajando como siempre sobre elementos triangulares lineales como ilustración, se escoge un sistema de coordenadas (s, t) (ver figura 2.8) cuyo origen se localiza en el nodo i y cuyo eje s sea paralelo al lado perteneciente a la frontera Γ y que conecta los nodos i y j.

La transformación vectorial del sistema de coordenadas se rige bajo las siguientes ecuaciones:

$$\begin{aligned} x(s,t) = &\quad x_i + (x_j - x_i)\frac{s}{a} + \left[\left(\frac{c}{a} - i\right) x_i - \frac{c}{a}x_j + x_k\right]\frac{t}{b} \\ y(s,t) = &\quad y_i + (y_j - x_i)\frac{s}{a} + \left[\left(\frac{c}{a} - i\right) y_i - \frac{c}{a}y_j + y_k\right]\frac{t}{b} \end{aligned} \tag{2.4.20}$$

donde a es la distancia entre los nodos i y j, y (c, b) es la posición en el espacio del nodo k sobre el sistema de coordenadas (s, t).

Al evaluar estas ecuaciones sobre el lado que conecta los nodos i y j se hace $t = 0$ siendo ahora

las ecuaciones de traslación

$$\begin{aligned} x(s) &= x_i + (x_j - x_i)\frac{s}{a} \\ y(s) &= y_i + (y_j - y_i)\frac{s}{a}. \end{aligned} \tag{2.4.21}$$

Esto nos sirve para expresar las funciones de peso $\phi_i(x, y)$:

$$\phi_i(s) \equiv \phi_i(s, 0) = \phi_i(x(s, 0), y(s, 0)) \tag{2.4.22}$$

aplicando los polinomios interpoladores triangulares lineales (2.4.10) obtenemos

$$\phi_i = 1 - \frac{s}{a}, \quad \phi_j = \frac{s}{a}, \qquad \phi_k = 0 \tag{2.4.23}$$

donde los índices de los nodos i, j y k permutan en orden natural.

Notamos que ϕ_i y ϕ_j son precisamente las funciones de interpolación unidimensionales asociadas con la línea del elemento que conecta los nodos i y j.

La evaluación de la integral de una función υ_e sobre la frontera Γ_e en un elemento puede generalizarse como

$$Q_e = \oint_{\Gamma_e} \upsilon_e d\Gamma_e \tag{2.4.24}$$

e involucra la integración de funciones apropiadas de interpolación unidimensionales en la frontera, es decir

$$Q_e = \int_{i-j} \upsilon_e(s)ds + \int_{j-k} \upsilon_e(s)ds + \int_{k-i} \upsilon_e(s)ds = Q_i^e + Q_j^e + Q_k^e \tag{2.4.25}$$

donde $\int_{i-j}$ denota la integral sobre la línea que conecta el nodo i con el nodo j.

En la práctica se necesitarán las integrales predefinidas de las combinaciones de los productos de las funciones de peso y sus derivadas con respecto a los ejes x y y. Para esto deducimos varias expresiones.

Primeramente se recuerda la definición de la derivada sobre una función de recta:

$$\frac{\partial \phi(s)}{\partial s} = \frac{\phi(s_j) - \phi(s_i)}{s_j - s_i} \tag{2.4.26}$$

Las derivadas $\frac{\partial \phi}{\partial x}$ y $\frac{\partial \phi}{\partial y}$ se hallan aplicando (2.4.21) sobre (2.4.26) :

$$\frac{\partial \phi(s)}{\partial x} = \frac{\phi(s_j) - \phi(s_i)}{x_i + (x_j + x_i)\frac{s_j}{a} - x_i - (x_j - x_i)\frac{s_i}{a}} = \frac{\phi_(s_j) - \phi_i(s_i)}{x_j - x_i} \tag{2.4.27}$$

similarmente

$$\frac{\partial\phi(s)}{\partial y} = \frac{\phi(s_j) - \phi(s_i)}{y_j - y_i} \tag{2.4.28}$$

Aplicando (2.4.23) sobre (2.4.27) y (2.4.28) y sabiendo (2.4.8) se tiene:

$$\begin{array}{lll} \dfrac{\partial\phi_i(s)}{\partial x} = -\dfrac{1}{\gamma_k} & \dfrac{\partial\phi_j(s)}{\partial x} = \dfrac{1}{\gamma_k} & \dfrac{\partial\phi_k(s)}{\partial x} = 0 \\ \dfrac{\partial\phi_i(s)}{\partial y} = \dfrac{1}{\beta_k} & \dfrac{\partial\phi_j(s)}{\partial y} = -\dfrac{1}{\beta_k} & \dfrac{\partial\phi_k(s)}{\partial y} = 0 \end{array} \tag{2.4.29}$$

Considerando esto definimos, al igual que en (2.4.11) :

$$S_{ij}^{\alpha\beta} = \int \phi_{i,\alpha}^{e}\phi_{j,\beta}^{e} ds \tag{2.4.30}$$

Como $\phi_k = 0$ resulta que cualquier integral que involucre este término se anula, por lo cual no se incluirán en la tabla de integrales predefinidas, a continuación:

$$\begin{array}{llll} S_{ii}^{00} = S_{jj}^{00} = \dfrac{a}{3} & S_{ij}^{00} = S_{ji}^{00} = \dfrac{a}{6} \\ S_{ii}^{01} = S_{ji}^{01} = -\dfrac{a}{3\gamma_k} & S_{ij}^{01} = S_{jj}^{01} = \dfrac{a}{3\gamma_k} \\ S_{ii}^{02} = S_{ji}^{02} = \dfrac{a}{3\beta_k} & S_{ij}^{02} = S_{jj}^{02} = -\dfrac{a}{3\beta_k} \\ S_{ii}^{10} = S_{ij}^{10} = -\dfrac{a}{3\gamma_k} & S_{ji}^{10} = S_{jj}^{10} = \dfrac{a}{3\gamma_k} \\ S_{ii}^{11} = S_{jj}^{11} = \dfrac{a}{\gamma_k^2} & S_{ij}^{11} = S_{ji}^{11} = -\dfrac{a}{\gamma_k^2} \\ S_{ii}^{12} = S_{jj}^{12} = -\dfrac{a}{\beta_k\gamma_k} & S_{ij}^{12} = S_{ji}^{12} = \dfrac{a}{\beta_k\gamma_k} \\ S_{ii}^{20} = S_{ij}^{20} = \dfrac{a}{3\beta_k} & S_{ji}^{20} = S_{jj}^{20} = -\dfrac{a}{3\beta_k} \\ S_{ii}^{21} = S_{jj}^{21} = -\dfrac{a}{\beta_k\gamma_k} & S_{ij}^{21} = S_{ji}^{21i} = \dfrac{a}{\beta_k\gamma_k} \\ S_{ii}^{22} = S_{jj}^{22} = \dfrac{a}{\beta_k^2} & S_{ij}^{22} = S_{ji}^{22} = -\dfrac{a}{\beta_k^2} \end{array} \tag{2.4.31}$$

La integral calculada para evaluar la frontera en (2.2.20) es también simple

$$\int_{i-j} g\phi_i ds = \int_{i-j} g\phi_j ds = \frac{a}{2}g \tag{2.4.32}$$

2.4.5 Funciones de forma de mayor grado

Para desarrollar elementos triangulares de mayor orden, como el cuadrático, es necesario introducir las *coordenadas de área*, *baricéntricas* o *trilineales*.

Si se une un punto interior P de un triángulo de área A con los tres vértices (figura 2.9) se obtienen tres subáreas A_1, A_2 y A_3 tales que $A_1 + A_2 + A_3 = A$. Las coordenadas de área se definen como

$$L_1 = \frac{A_1}{A}, \quad L_2 = \frac{A_2}{A}, \quad L_3 = \frac{A_3}{A} \tag{2.4.33}$$

cumpliéndose que

$$L_1 + L_2 + L_3 = 1 \tag{2.4.34}$$

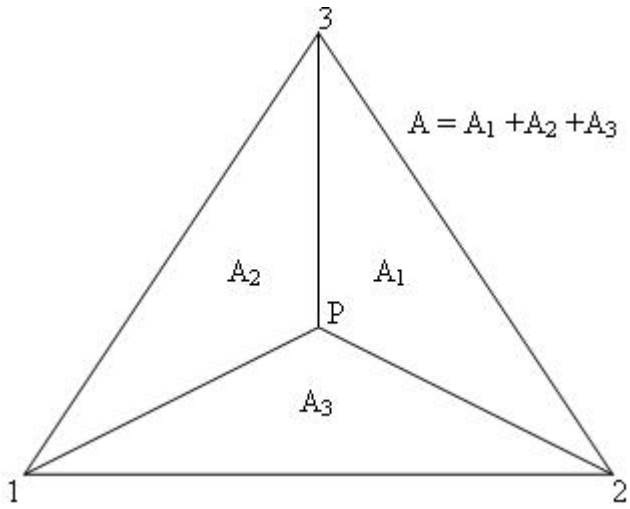


Figura 2.9: División de un triángulo para obtener las coordenadas de área.

La posición del punto P puede definirse por dos de dichas coordenadas. El centro de gravedad del triángulo tiene como coordenadas de área $L_1 = L_2 = L_3 = 1/3$. Las coordenadas de área tienen la propiedad que para cualquier nodo p_j, se tiene que

$$L_i(p_j) = \delta_{ij} \tag{2.4.35}$$

cumpliendo la restricción de las funciones de forma. Además, si se define una interpolación lineal del campo de desplazamiento de las variables independientes; es decir, las coordenadas del plano donde se define el dominio; como:

$$\begin{aligned} x &= L_1 x_1 + L_2 x_2 + L_3 x_3 \\ y &= L_1 y_1 + L_2 y_2 + L_3 y_3 \end{aligned} \tag{2.4.36}$$

se puede demostrar que para el elemento triangular lineal

$$L_i = \frac{1}{2A_e}(\alpha_i + \beta_i x + \gamma_i y) = \phi_i^e, \quad i = 1, 2, 3 \tag{2.4.37}$$

Por lo que las coordenadas de área pueden utilizarse para definir una interpolación paramétrica del elemento (no de la función solución).

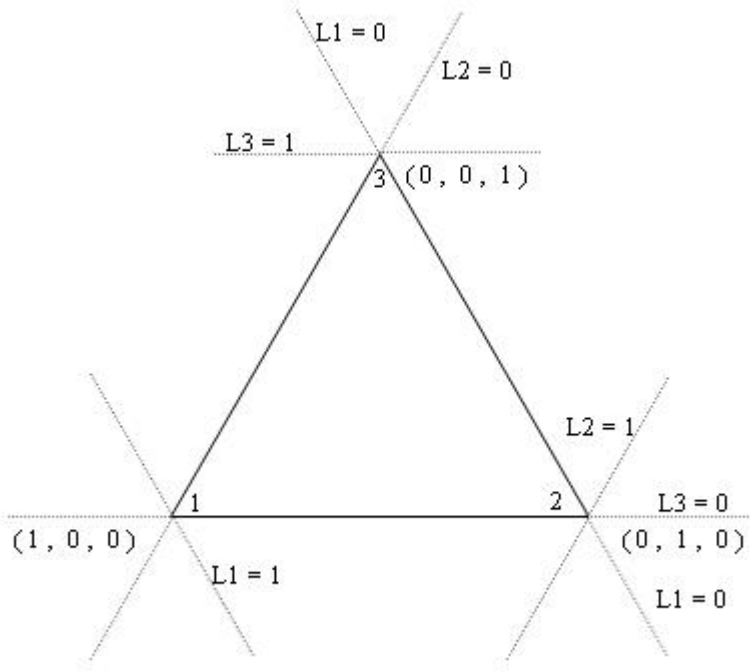


Figura 2.10: Elemento triangular lineal, valores nodales de la coordenadas L_i y los valores de las ternas (I, J, K).

Las funciones de forma de los elementos triangulares que contienen polinomios completos de grado M pueden obtenerse en función de las coordenadas de área por el procedimiento siguiente: la distribución de los nodos en el triángulo es el mismo que los términos del triángulo de Pascal para generar el polinomio completo de grado deseado. Se numeran los vértices del triángulo con la secuencia 1, 2, 3 en dirección contraria a las agujas del reloj, de manera de identificar las coordenadas de área correspondientes L_1, L_2 y L_3. A cada nodo i se le asigna una terna (I, J, K) tal que $I + J + K = M$ y varían en 1 con respecto a la coordenada de área asociada, L_1 para I, L_2 para J y L_3 para K. Lo que quiere decir que para el nodo 1, se tiene $(M, 0, 0)$; en el nodo 2, $(0, M, 0)$ y en el nodo 3, $(0, 0, M)$. Los valores de I, J y K coinciden con los exponentes con que van afectadas cada una de las coordenadas de área L_1, L_2 y L_3 en la expresión de la función de forma del nodo i, y ésta viene dada por:

$$\varphi_i^e = l_i^I(L_1) l_i^J(L_2) l_i^K(L_3) \tag{2.4.38}$$

donde $l_i^I(L_1)$ es el polinomio de Lagrange de grado I en L_1, que toma el valor unidad en el nodo i y cero en el resto, es decir

$$l_i^I(L_1) = \prod_{\substack{j=1 \\ j\neq i}}^{I+1} \frac{(L_1 - L_1^j)}{(L_1^i - L_1^j)} \tag{2.4.39}$$

donde L_1^i es el valor de la coordenada L_1 en el nodo i. Se tienen expresiones similares para $l_i^J(L_2)$ y $l_i^K(L_3)$.

Para el caso del elemento triangular lineal $M = 1$, la posición de cada nodo, las ternas (I, J, K) asignadas y sus coordenadas de área pueden verse en la figura 2.10.

Para el nodo 1 se tiene la terna $(1, 0, 0)$, por lo que de (2.4.38) y (2.4.39) se tiene que

$$\varphi_1^e = l_1^1(L_1) = L_1 \tag{2.4.40}$$

de igual forma se obtienen $\varphi_2^e = L_2$ y $\varphi_3^e = L_3$, resultado ya conocido.

Para las funciones de forma del elemento cuadrático de 6 nodos, $M = 2$, se muestran las distribuciones de los nodos, el valor de las coordenadas de área y las ternas asignadas en la figura 2.11.

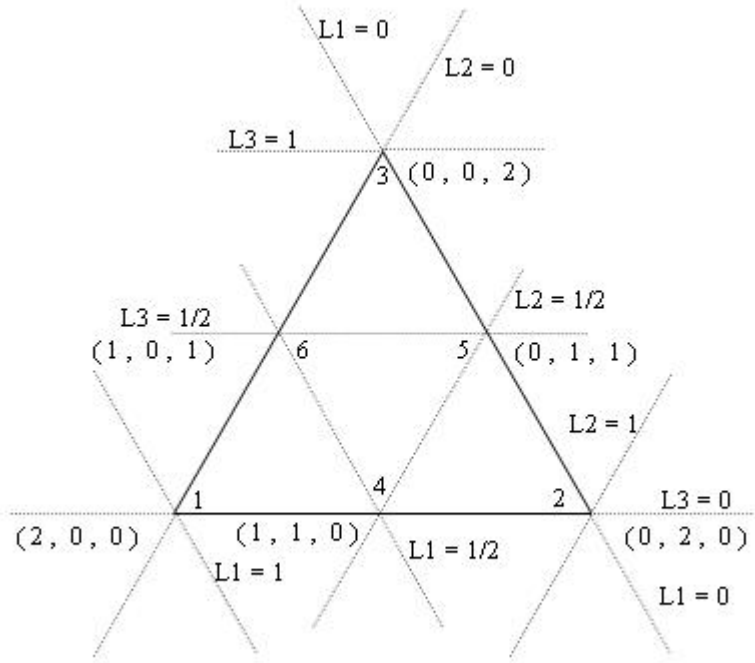


Figura 2.11: Elemento triangular cuadrático, valores nodales de la coordenadas L_i y los valores de las ternas (I, J, K).

Para el nodo 1, se tiene la terna $(2, 0, 0)$ y las coordenadas de área son: $L_1 = 1$, $L_2 = L_3 = 0$;

entonces de (2.4.38) y (2.4.39) se deduce

$$\varphi_1^e = l_1^2(L_1) = \frac{(L_1 - \frac{1}{2})L_1}{(1 - \frac{1}{2})1} = (2L_1 - 1)L_1 \tag{2.4.41}$$

En el nodo 4, la terna asignada es $(1, 1, 0)$ y $L_1 = L_2 = 1/2$, $L_3 = 0$, por tanto se tiene que

$$\varphi_4^e = l_4^1(L_1)l_4^1(L_2) = \frac{L_1}{\frac{1}{2}}\frac{L_2}{\frac{1}{2}} = 4L_1L_2 \tag{2.4.42}$$

Siguiendo el mismo procedimiento se obtienen las funciones de forma para todos los nodos.

$$\begin{array}{lll} \varphi_1^e = (2L_1 - 1)L_1 & \varphi_2^e = (2L_2 - 1)L_2 & \varphi_3^e = (2L_3 - 1)L_3 \\ \varphi_4^e = 4L_1L_2 & \varphi_5^e = 4L_2L_3 & \varphi_6^e = 4L_3L_1 \end{array} \tag{2.4.43}$$

En la figura 2.12 se muestran las funciones φ_1^e y φ_6^e.

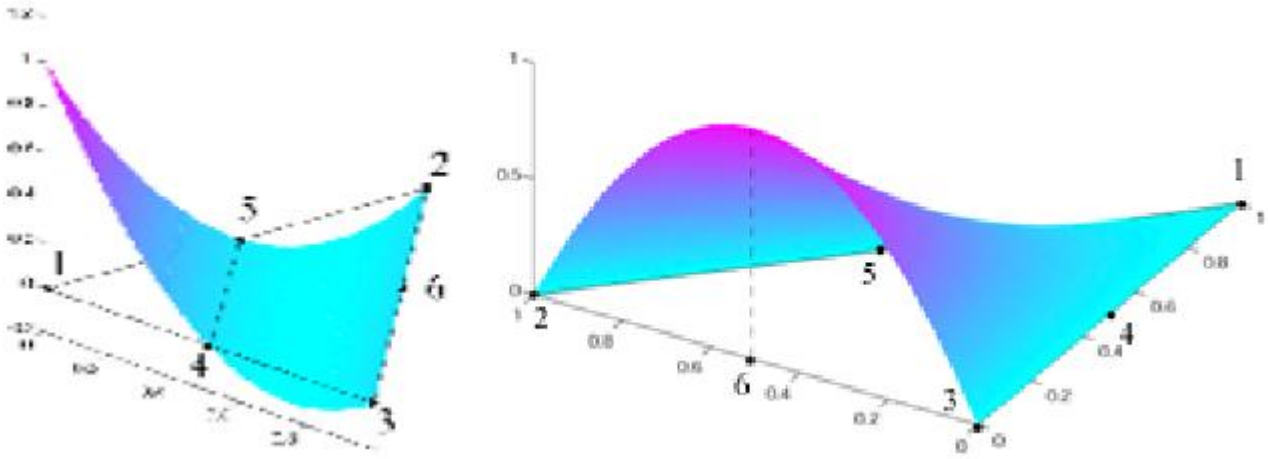


Figura 2.12: Funciones φ_1^e y φ_6^e.

2.4.6 Integración para funciones de grado mayor

Es importante señalar que las coordenadas de área no solo facilitan la construcción de las funciones de interpolación para elementos de orden superior, sino también facilitan el cálculo de las integrales de área y frontera de las funciones L_i. De [13] y [6] se tienen las expresiones

$$\begin{aligned} \int_a^b L_1^m L_2^n \, ds &= \frac{m!\, n!}{(m+n+1)!}(b-a) \\ \iint_A L_1^m L_2^n L_3^p \, dA &= \frac{m!\, n!\, p!}{(m+n+p+2)!} 2A \end{aligned} \tag{2.4.44}$$

donde m, n y p son enteros positivos, A es el área del dominio de integración, y $m!$ es el factorial de m. Por supuesto, se deben transformar las integrales de las coordenadas x, y a las coordenadas L_i, utilizando la expresión (2.4.36) .

En caso general, las derivadas de ϕ_i con respecto a las coordenadas globales pueden ser calculadas a partir de

$$\begin{aligned} \frac{\partial \phi_i}{\partial x} &= \frac{\partial \phi_i}{\partial L_1}\frac{\partial L_1}{\partial x} + \frac{\partial \phi_i}{\partial L_2}\frac{\partial L_2}{\partial x} \\ \frac{\partial \phi_i}{\partial y} &= \frac{\partial \phi_i}{\partial L_1}\frac{\partial L_1}{\partial y} + \frac{\partial \phi_i}{\partial L_2}\frac{\partial L_2}{\partial y} \end{aligned} \tag{2.4.45}$$

o sea

$$\begin{bmatrix} \frac{\partial \phi_i}{\partial x} \\ \frac{\partial \phi_i}{\partial y} \end{bmatrix} = [\mathcal{J}]^{-1} \begin{bmatrix} \frac{\partial \phi_i}{\partial L_1} \\ \frac{\partial \phi_i}{\partial L_2} \end{bmatrix}, \quad [\mathcal{J}] = \begin{bmatrix} \frac{\partial x}{\partial L_1} & \frac{\partial y}{\partial L_1} \\ \frac{\partial x}{\partial L_2} & \frac{\partial y}{\partial L_2} \end{bmatrix} \tag{2.4.46}$$

Nótese que sólo L_1 y L_2 han sido tratadas como linealmente independientes, ya que $L_3 = 1 - L_1 - L_2$.

A la matriz $[\mathcal{J}]$ se le llama la matriz Jacobiana o simplemente el Jacobiano. Para un elemento triangular recto la matriz jacobiana es fácil de calcular:

$$[\mathcal{J}] = \begin{bmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{bmatrix} = \begin{bmatrix} \gamma_3 & -\beta_3 \\ -\gamma_2 & \beta_2 \end{bmatrix} \tag{2.4.47}$$

donde β_i y γ_i son las constantes definidas en 2.4.8. La inversa del Jacobiano está dada por

$$[\mathcal{J}] = \frac{1}{\mathcal{J}} \begin{bmatrix} \beta_2 & \beta_3 \\ \gamma_2 & \gamma_3 \end{bmatrix}, \quad \mathcal{J} = \beta_2\gamma_3 - \gamma_2\beta_3 = 2A \tag{2.4.48}$$

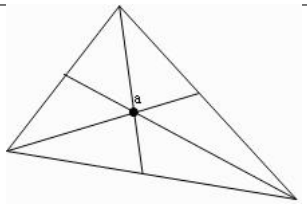
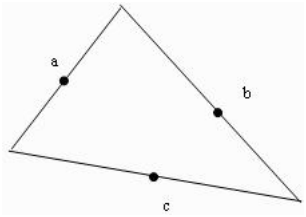
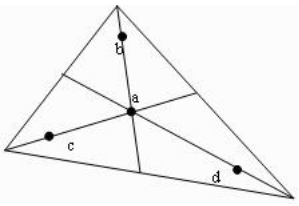
Sabiendo esto, la cuadratura de Gauss puede ser implementada fácilmente para la integración. Después de la transformación, las integrales en Ω_e, tienen la forma:

$$\int_{\Omega_e} G(x,y)d\Omega_e = \int_{\widehat{\Omega}_e} \widehat{G}(L_1, L_2, L_3)dL_1dL_2 \tag{2.4.49}$$

que puede ser aproximada por la fórmula de la cuadratura

$$\int_{\widehat{\Omega}_e} \widehat{G}(L_1, L_2, L_3) dL_1 dL_2 \approx \frac{1}{2} \sum_{I=1}^{Ng} W_I \widehat{G}(S_i) \tag{2.4.50}$$

donde W_I y S_I denotan los pesos y los puntos de integración de las reglas de cuadratura y N_g es el número de puntos de integración. La tabla 2.1 muestra la localización de los puntos de integración y los posos para las relas de cuadratura de uno, tres y cuatro puntos para los elementos triangulares.

Número de puntos de integración	**Grado del polinomio y orden del residual**	**Localización de los puntos de integración**					
		L_1	L_2	L_3	W		**Geometric locations**
1	1 $O(h^2)$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	1	a	
3	2 $O(h^3)$	$\frac{1}{2}$ $\frac{1}{2}$ 0	0 $\frac{1}{2}$ $\frac{1}{2}$	$\frac{1}{2}$ 0 $\frac{1}{2}$	$\frac{1}{3}$ $\frac{1}{3}$ $\frac{1}{3}$	a b c	
4	3 $O(h^4)$	$\frac{1}{3}$ 0.6 0.2 0.2	$\frac{1}{3}$ 0.2 0.6 0.2	$\frac{1}{3}$ 0.2 0.2 0.6	$-\frac{27}{48}$ $\frac{25}{48}$ $\frac{25}{48}$ $\frac{25}{48}$	a b c d	

Cuadro 2.1: Puntos de cuadratura y pesos

2.5 Sistemas de ecuaciones lineales

El objetivo de esta sección es examinar los aspectos numéricos que se presentan al resolver sistemas de ecuaciones lineales de la forma:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n = & b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n = & b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \ldots + a_{3n}x_n = & b_3 \\ \vdots & \vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nn}x_n = & b_n \end{cases} \tag{2.5.1}$$

Se trata de un sistema de n ecuaciones con n incógnitas, $x_1, x_2, \ldots, x_n$. Los elementos a_{ij} y b_i son números reales fijados. El sistema de ecuaciones (2.5.1) se puede escribir empleando una muy útil representación matricial, como:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \ldots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \ldots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \ldots & a_{3n} \\ \vdots & & & & \\ a_{n1} & a_{n2} & a_{n3} & \ldots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix} \tag{2.5.2}$$

Entonces podemos denotar estas matrices por $\mathbf{A}$, $\mathbf{x}$ y $\mathbf{b}$ de forma que la ecuación se reduce simplemente a:

$$\mathbf{Ax} = \mathbf{b} \tag{2.5.3}$$

Según Kincaid [14], los métodos de resolución de sistemas de ecuaciones se pueden dividir en dos grandes grupos:

- **Los Métodos exactos o directos** o algoritmos finitos que permiten obtener la solución del sistema de manera directa.
- **Los Métodos aproximados o iterativos** que utilizan algoritmos iterativos y que calculan la solución del sistema por aproximaciones sucesivas.

Al contrario de lo que pueda parecer, en muchas ocasiones los métodos aproximados permiten obtener un grado de exactitud superior al que se puede obtener empleando los denominados métodos exactos, debido fundamentalmente a los errores de truncamiento que se producen en el proceso.

De entre los métodos exactos se pueden mencionar el método de Gauss y una modificación de éste denominado método de Gauss-Jordan. Sin embargo, nos centraremos en el estudio de los métodos iterativos GMRES (Residual Mínimo Generalizado), ConjGrad (Gradiente Conjugado), BiCGstab (Gradiente Biconjugado Estabilizado) y BiCG (Gradiente BiConjugado).

2.5.1 Métodos iterativos

El método de Gauss y sus variantes se conocen con el nombre de métodos $directos$: se ejecutan a través de un número finito de pasos y dan lugar a una solución que sería exacta si no fuese por los errores de redondeo.

Por contra, un método $indirecto$ da lugar a una sucesión de vectores que idealmente converge a la solución. El cálculo se detiene cuando se cuenta con una solución aproximada con cierto grado de precisión especificado de antemano o después de cierto número de iteraciones. Los métodos indirectos son casi siempre iterativos: para obtener la sucesión mencionada se utiliza repetidamente un proceso sencillo.

En general, en todos los procesos iterativos para resolver el sistema $\mathbf{Ax} = \mathbf{b}$ se recurre a una cierta matriz $\mathbf{Q}$, llamada matriz *descomposición*, escogida de tal forma que el problema original adopte la forma equivalente:

$$\mathbf{Qx} = (\mathbf{Q} - \mathbf{A})\mathbf{x} + \mathbf{b} \tag{2.5.4}$$

La ecuación (2.5.4) sugiere un proceso iterativo que se concreta al escribir:

$$\mathbf{Q}\mathbf{x}^{(k)} = (\mathbf{Q} - \mathbf{A})\,\mathbf{x}^{(k-1)} + \mathbf{b} \quad (k \geq 1) \tag{2.5.5}$$

El vector inicial $\mathbf{x}^{(0)}$ puede ser arbitrario, aunque si se dispone de un buen candidato como solución éste es el que se debe emplear. La aproximación inicial que se adopta, a no ser que se disponga de una mejor, es la idénticamente nula $x_1 = x_2 = \ldots = x_n = 0$. A partir de la ecuación (2.5.5) se puede calcular una sucesión de vectores $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$. Nuestro objetivo es escoger una matriz **Q** de manera que:

- se pueda calcular fácilmente la sucesión $\left[\mathbf{x}^{(k)}\right]$.
- la sucesión $\left[\mathbf{x}^{(k)}\right]$ converja rápidamente a la solución.

Como en todo método iterativo, deberemos especificar un criterio de convergencia δ y un número máximo de iteraciones M, para asegurar que el proceso se detiene si no se alcanza la convergencia. En este caso, puesto que $\mathbf{x}$ es un vector, emplearemos dos criterios de convergencia que se deberán satisfacer simultáneamente:

1. El módulo del vector diferencia, $\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|$, partido por el módulo del vector $\mathbf{x}$, $\left\|\mathbf{x}^{(k)}\right\|$ deberá ser menor que la convergencia deseada:

$$\frac{\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|}{\left\|\mathbf{x}^{(k)}\right\|} \leq \delta \tag{2.5.6}$$

2. La diferencia relativa del mayor elemento en valor absoluto del vector $\mathbf{x}^{(k)}$, $\mathbf{x}_m = Max\{x_i\}$, deberá ser diez veces menor que δ:

$$ABS\left(\frac{\mathbf{x}_m^{(k)} - \mathbf{x}_m^{(k-1)}}{\mathbf{x}_m^{(k)}}\right) \leq \frac{\delta}{10} \tag{2.5.7}$$

2.5.2 Método de Residual Mínimo Generalizado

El método *Residual Mínimo Generalizado* o GMRES (*Generalized Minimal Residual*) es una extensión del llamado MINRES o Método Mínimo Generalizado, el cual funciona sólo para sistemas simétricos. Al igual que MINRES, genera una secuencia de vectores ortogonales, pero en ausencia de la simetría no puede hacerse con cortas recurrencias, sino que tienen que ser retenidos vectores previamente calculados en la secuencia ortogonal. Por esta razón, son usadas versiones $reiniciadas$ del método. En el método Gradiente Conjugado, los residuales forman una base ortogonal para el espacio $span\left\{\mathbf{r}^{(0)}, \mathbf{A}\mathbf{r}^{(0)}, \mathbf{A}^2\mathbf{r}^{(0)}, \ldots\right\}$. En GMRES, esta base es formada explícitamente:

$w^{(k)} = Av^{(k)}$
para $i = 1, \ldots, k$
    $w^{(k)} = w^{(k)} - \left(w^{(k)}, v^{(i)}\right) v^{(k)}$
fin
$v^{(k+1)} = w^{(k)} / \left\|w^{(k)}\right\|$

Esto es muy parecido a una ortogonalización Gram-Schmidt. Aplicado a la secuencia de Krylov $\left\{A^i r^{(0)}\right\}$ esta ortogonalización es llamada *método Arnoldi*. Los coeficientes de los productos internos $(w^{(k)}, v^{(i)})$ y $\left\|w^{(k)}\right\|$ son almacenados en una matriz superior Hessenberg. La iteración GMRES es construida como

$$x^{(k)} = x^{(0)} + y_1 v^{(1)} + \ldots + y_k v^{(k)}, \tag{2.5.8}$$

donde los coeficientes y_h han sido escogidos para minimizar la norma residual $\left\|b - Ax^{(k)}\right\|$. El algoritmo GMRES tiene la propiedad de que ésta norma residual puede ser calculada sin que

la iteración haya sido formada. Por lo tanto, la cara acción de formar la iteración puede ser postpuesta hasta que la norma residual sea juzgada suficientemente pequeña. El pseudocódigo para el algoritmo reinicializado $GMRES^{(m)}$ con precondicionante M es dado en el cuadro 2.2.

$x^{(0)}$ es un valor inicial
para $j = 1, 2, \ldots$
    resolver r desde $Mr = b - Ax^{(0)}$
    $v^{(1)} = r / \|r\|_2$
    $a = \|r\|_{2e1}$
    para $i = 1, 2, \ldots, m$
        resolver w desde $Mw = Av^{(i)}$
        para $k = 1, \ldots, i$
            $h_{k,i} = (w, v^{(k)})$
            $w = w - h_{k,i}v^{(k)}$
        finpara
        $h_{i+1,i} = \|w\|_2$
        $v^{(i+1)} = w/h_{i+1,i}$
        aplicar $J_1, \ldots, J_{i-1}$ en $(h_{1,i}, \ldots, h_{i+1,i})$
        construir J_i actuando sobre ith y el $i(i+1)^o$ componente
        de $h_{.,i}$, tal que el $(i+1)^o$ componente de $J_i h_{.,i}$ es $a = J_i a$
        si $a(i+1)$ es lo suficientemente pequeño
            ACTUALIZAR(x, i)
            salir
        finsi
    finpara
    ACTUALIZAR$(\widetilde{x}, m)$
    revisar convergencia; continuar si es necesario
fin

Cuadro 2.2: El Método Precondicionado $GMRES^{(m)}$

2.5.3 Método de Gradiente Conjugado

El método *Gradiente Conjugado* es muy efectivo para sistemas definidos simétricos positivos. Es el más antiguo y mejor conocido método no-estacionario discutido aquí. El método procede generando sucesivas aproximaciones a la solución y sus correspondientes residuales, y busca direcciones usadas para actualizar las aproximaciones y los residuales. Aunque la longitud de estas secuencias puede crecer, sólo un pequeño número de vectores necesita ocupar la memoria. En cada iteración dos productos internos son realizados para calcular actualizaciones de escalares que son definidos para que las secuencias satisfagan ciertas condiciones de ortogonalidad. En un sistema lineal definido positivo simétrico estas condiciones implican que la distancia a la verdadera

solución es minimizada en alguna norma.

El vector $\mathbf{x}^{(k)}$ es actualizado en cada iteración por un múltiplo (α_i) del vector de dirección de búsqueda $p^{(k)}$:

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_i p^{(k)}. \tag{2.5.9}$$

De manera correspondiente los residuales $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ son actualizados como

$$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha \mathbf{q}^{(k)} \tag{2.5.10}$$

donde $\mathbf{q}^{(k)} = \mathbf{A}p^{(i)}$.

La escogencia de $\alpha = \alpha_i = \mathbf{r}^{(k-1)^T}\mathbf{r}^{(k-1)}/p^{(k)^T}\mathbf{A}p^{(k)}$ minimiza $\mathbf{r}^{(k)^T}\mathbf{A}^{-1}\mathbf{r}^{(k)}$ sobre cualquier posible valor de α en la ecuación (2.5.10) .

Las direcciones de búsqueda son actualizados usando los residuales

$$p^{(k)} = \mathbf{r}^{(k)} + \beta_{i-1} p^{(k-1)} \tag{2.5.11}$$

donde la escogencia de $\beta_i = \mathbf{r}^{(k)^T}\mathbf{r}^{(k)}/\mathbf{r}^{(k-1)^T}\mathbf{r}^{(k-1)}$ asegura que $p^{(k)}$ y $\mathbf{A}p^{(k-1)}$ o equivalentemente, $\mathbf{r}^{(k)}$ y $\mathbf{r}^{(k-1)}$ sean ortogonales. De hecho, se puede demostrar que esta escogencia de β_i hace que $p^{(k)}$ y $\mathbf{r}^{(i)}$ sean ortogonales para cualquier previo $\mathbf{A}p^{(j)}$ y $\mathbf{r}^{(j)}$ respectivamente.

El pseudocódigo del Método Gradiente Conjugado Precondicionado se muestra en el cuadro 2.3. Usa como precondicionante a M para $M = I$ se obtiene la versión no precondicionada del algoritmo Gradiente Conjugado. En ese caso el algoritmo puede ser simplificado saltándose la línea *solve* y remplazando $z^{(k-1)} por \mathbf{r}^{(k-1)}$ y $z^{(0)}$ por $\mathbf{r}^{(0)}$.

2.5.4 Método de Gradiente BiConjugado

El método Gradiente Conjugado no funciona en sistemas no simétricos porque los vectores residuales no pueden ser creados ortogonales con las recurrencias cortas. El método GMRES retiene la ortogonalidad de los residuales usando recurrencias largas al costo de una mayor demanda de almacenamiento. El método Gradiente BiConjugado aprovecha otra ventaja, reemplazando la secuencia ortogonal de residuales por dos secuencias mutuamente ortogonales, al precio de no proveer más una disminución.

Las relaciones de actualización para residuales en el método Gradiente Conjugado son incrementados en el método Gradiente BiConjugado por relaciones que son similares pero basadas

calcular $r^{(0)} = b - Ax^{(0)}$ para un valor inicial $x^{(0)}$
para $i = 1, 2, ...$
resolver $Mz^{(k-1)} = r^{(k-1)}$
$p_{k-1} = r^{(k-1)^T} z^{(k-1)}$
si $k = 1$
$p^{(1)} = z^{(0)}$
sino
$\beta_{k-1} = p_{k-1}/p_{k-2}$
$p^{(k)} = z^{(k-1)} + \beta_{k-1} p^{(k-1)}$
finsi
$q^{(k)} = Ap^{(k)}$
$a_k = p_{k-1}/p^{(k)^T} q^{(k)}$
$x^{(i)} = x^{(k-1)} + a_i p^{(k)}$
$r^{(k)} = r^{(k-1)} - a_k q^{(k)}$
revisar convergencia; continuar si es necesario
fin

Cuadro 2.3: El Método Gradiente Conjugado Precondicionado

en $\mathbf{A}^T$ y no en **A**. Así pues se actualizan dos secuencias de residuales.

$$r^{(k)} = r^{(k-1)} - \alpha_k \mathbf{A} p^{(k)}, \quad \widetilde{r}^{(k)} = \widetilde{r}^{(k-1)} - \alpha_k \mathbf{A}^T \widetilde{p}^{(k)},$$

y dos secuencias de direcciones de búsqueda

$$p^{(k)} = r^{(k-1)} + \beta_{k-1} \mathbf{A} p^{(k-1)}, \quad \widetilde{p}^{(k)} = \widetilde{r}^{(k-1)} + \beta_{k-1} \mathbf{A} \widetilde{p}^{(k-1)},$$

donde se escoge

$$\alpha_k = \frac{\widetilde{r}^{(k-1)^T} r^{(k-1)}}{\widetilde{p}^{(k)^T} \mathbf{A} p^{(k)}}, \quad \beta_k = \frac{\widetilde{r}^{(k)^T} r^{(k-1)}}{\widetilde{r}^{(k-1)^T} \mathbf{A} r^{(k-1)}},$$

asegurando las relaciones de bi-ortogonalidad

$$\widetilde{r}^{(k)^T} r^{(j)} = \widetilde{p}^{(k)^T} \mathbf{A} p^{(j)} = 0 \quad i \neq j$$

El pseudocódigo del método BiCG con precondicionante **M** es dado en la tabla 2.2

2.5.5 Método de Gradiente Conjugado Cuadrado

En el Método BiCG, el vector residual $r^{(k)}$ puede ser considerado como el producto de $r^{(0)}$ y un i^o grado polinomial en **A**, que es

$$r^{(k)} = P_k(\mathbf{A}) r^{(0)}$$

calcular $r^{(0)} = b - Ax^{(0)}$ para un valor inicial $x^{(0)}$
escoger $\widetilde{r}^{(0)}$ (por ejemplo, $\widetilde{r}^{(0)} = r^{(0)}$).
para $i = 1, 2, ...$
resolver $Mz^{(k-1)} = r^{(k-1)}$
resolver $M\widetilde{z}^{(k-1)} = \widetilde{r}^{(k-1)}$
$p_{k-1} = r^{(k-1)^T}\widetilde{r}^{(k-1)}$
si $k = 1$
$p^{(1)} = z^{(0)}$
$\widetilde{p}^{(1)} = \widetilde{z}^{(0)}$
sino
$\beta_{k-1} = p_{k-1}/p_{k-2}$
$p^{(k)} = z^{(k-1)} + \beta_{k-1}p^{(k-1)}$
$p^{(k)} = \widetilde{z}^{(k-1)} + \beta_{k-1}\widetilde{p}^{(k-1)}$
finsi
$q^{(k)} = Ap^{(k)}$
$\widetilde{q}^{(k)} = A^T\widetilde{p}^{(k)}$
$a_k = p_{k-1}/p^{(k)^T}q^{(k)}$
$x^{(i)} = x^{(k-1)} + a_i p^{(k)}$
$r^{(k)} = r^{(k-1)} - a_k q^{(k)}$
$\widetilde{r}^{(k)} = r^{(k-1)} - a_k\widetilde{q}^{(k)}$
revisar convergencia; continuar si es necesario
fin

Cuadro 2.4: El Método Gradiente BiConjugado Precondicionado

El mismo polinomio satisface $\widetilde{r}^{(k)} = P_k(\mathbf{A}^T)\widetilde{r}^{(0)}$ tal que

$$p_k = \left(\widetilde{r}^{(k)}, r^{(k)}\right) = \left(P_k(\mathbf{A}^T)\widetilde{r}^{(0)}, P_k(\mathbf{A})r^{(0)}\right) = \left(\widetilde{r}^{(0)}, P_k^2(\mathbf{A})r^{(0)}\right)$$

Esto sugiere que si $P_k(\mathbf{A})$ reduce $r^{(0)}$ al menor vector $r^{(k)}$, entonces puede ser ventajoso aplicar este operador "contracción" dos veces, y calcular $P_k^2(\mathbf{A})r^{(0)}$. La última ecuación muestra que los coeficientes de la iteración pueden aún ser redescubiertos a partir de estos vectores, y vienen a ser fácil de encontrar las aproximaciones correspondientes para x. El método CGS está centrado en esta aproximación.

2.5.6 Método de Gradiente BiConjugado Estabilizado

El método Gradiente BiConjugado Estabilizado, explicado en [35], fue desarrollado para resolver sistemas lineales no simétricos mientras se evitan los frecuentes patrones irregulares de convergencia de los otros métodos. En vez de calcular la secuencia de CGS $i \longrightarrow P_k^2(\mathbf{A})r^{(0)}$,

calcular $r^{(0)} = b - Ax^{(0)}$ para un valor inicial $x^{(0)}$
escoger $\widetilde{r}^{(0)}$ (por ejemplo, $\widetilde{r}^{(0)} = r^{(0)}$).
para $i = 1, 2, ...$
$p_{k-1} = \widetilde{r}^{(k-1)^T} r^{(k-1)}$
si $p_{k-1} = 0$ **error**
si $k = 1$
$u^{(1)} = r^{(0)}$
$p^{(1)} = u^{(1)}$
sino
$\beta_{k-1} = p_{k-1}/p_{k-2}$
$u^{(k)} = r^{(k-1)} + \beta_{k-1} q^{(k-1)}$
$p^{(k)} = u^{(k)} + \beta_{k-1}(q^{(k-1)} + \beta_{k-1} p^{(k-1)})$
finsi
resolver $M\beta = p^{(k)}$
$\theta = A\beta$
$\alpha_k = p_{k-1}/\widetilde{r}^T\theta$
$q^{(k)} = u^{(k)} - \alpha_k\theta$
resolver $M\theta = u^{(k)} + q^{(k)}$
$x^{(i)} = x^{(k-1)} + \alpha_{ip}{}^{(k)}$
$\widetilde{q} = A\theta$
$r^{(k)} = r^{(k-1)} - \alpha_k\widetilde{q}$
revisar convergencia; continuar si es necesario
fin

Cuadro 2.5: El Método Gradiente Cuadrado Precondicionado

Bi-CGSTAB calcula $i \rightarrow Q_k(\mathbf{A})P_k(\mathbf{A})r^{(0)}$ donde Q_k es un plinomio de i^o grado que describe una actualización más profundamente descendiente.

calcular $r^{(0)} = b - Ax^{(0)}$ para un valor inicial $x^{(0)}$
escoger $\widetilde{r}^{(0)}$ (por ejemplo, $\widetilde{r}^{(0)} = r^{(0)}$).
para $i = 1, 2, ...$
$p_{k-1} = \widetilde{r}^{(k-1)^T} r^{(k-1)}$
si $p_{k-1} = 0$ **error**
si $k = 1$
$p^{(1)} = r^{(k-1)}$
sino
$\beta_{k-1} = (p_{k-1}/p_{k-2})(\alpha_{k-1}/w_{k-1})$
$p^{(k)} = r^{(k-1)} + \beta_{k-1}(p^{(k-1)} - w_{k-1}v^{(k-1)}$
finsi
resolver $M\beta = p^{(k)}$
$v^{(k)} = A\beta$
$\alpha_k = p_{k-1}/\widetilde{r}^T v^{(k)}$
$a = r^{(k-1)} - \alpha_{kv}{}^{(k)}$
si revisar norma de a es suficientemente pequeña
$x^{(k)} = x^{(k-1)} + \alpha_{k\beta}$
stop
finsi
resolver $M\theta = a$
$t = A\theta$
$w_k = t^T a/t^T t$
$x^{(k)} = x^{(k-1)} + \alpha_k\beta + w_k{}^{\theta}$
$r^{(k)} = a - w_k t$
revisar convergencia; continuar si es necesario
para continuar es necesario que $w_k \neq 0$
fin

Cuadro 2.6: El Método Gradiente Cuadrado Precondicionado

CAPÍTULO 3

DESARROLLO

Como se mencionó en 1.5 el desarrollo del software en este trabajo se guía por un enfoque metodológico incremental rápido, ligero y eficiente.

El proceso iterativo que encamina el desarrollo del software facilita el flujo del trabajo de una manera organizada, coherente y eficaz. A pesar de no apegarse a los lineamientos de ninguna metodología actualmente conocida, ha basado algunas fases en otros enfoques de desarrollo como el Ciclo de Vida y XP. Su filosofía se basa principalmente en la existencia de una sencillez tal que permita que florezca el desarrollo del sistema sin dificultar el avance del trabajo con documentaciones o técnicas rígidas innecesarias que no corresponden con un proceso real de trabajo fluido. Ya que el proceso a realizar no prioritiza el desarrollo en equipo, la comercialización del producto ni su interacción con el usuario, simplifica al máximo las tareas, optando únicamente por las que colaboren con el objetivo.

La practicidad que exige una mente racional crea la necesidad de adaptar una metodología al software en vez de forzar el desarrollo a un proceso que ha sido preestablecido sin pensar en cada caso específico. El proceso se puede describir en 4 fases o etapas principales, no para catalogar tareas que en realidad tienen aspectos que las hacen inseparables sino más bien con propósito descriptivo. Estos pasos son:

- Análisis.

  Se identifica el problema y los conocimientos involucrados a adquirir, y se formalizan de manera unívoca los requerimientos. Esta fase observa el estudio exhaustivo de la base teórica y práctica a nivel matemático, algorítmico y de desarrollo en programación. Este estudio global y deliberado se realiza a fondo al principio del proceso. Sin embargo, el posterior incremento de conocimiento obtenido en los pasos posteriores puede producir cambios suaves en la percepción inicial del problema, que son considerados agregados de las fases respectivas. Además, se aplican los estudios de forma práctica desarrollando el problema con un propósito didáctico dirigido a un mayor dominio del tema y del problema en sí.

- Diseño.

  La etapa de diseño se centra principalmente en la creación de la idea abstracta de lo que se convierte en un producto tangible. Una vez conocido el procedimiento con el que se ataca el problema se forma la estructura que ha de tener el software al ser implementado. Haciendo uso debido de herramientas modernas como el lenguaje UML queda claro para el desarrollador el cuerpo funcional del programa, que éste no crea sino que capta a partir del problema. Orientado a diferentes aspectos como el lenguaje de programación, el diseño propone una arquitectura general para el software, enumera sus componentes y los algoritmos básicos a utilizar. Todo esto según una perspectiva modular, cuyas innumerables ventajas se obvian, y evitando el paradigma orientado a objetos, que tampoco se discutirá.

- Implementación.

  El código puede ser implementado libremente en conjunto con el diseño mismo. Esto puede traer ciertos riesgos de generación de errores, por eso muy pocas metodologías apoyan esta flexibilidad, pero con una estrategia adecuada de programación este peligro puede ser evitado y puede acelerarse así el desarrollo de manera considerable. Esta estrategia en la codificación también sirve para que la repetición de iteraciones sobre las etapas no produzca parches que debiliten la estructura del software, sino que más bien la fortalezcan y mejoren. El código contiene la documentación necesaria para que se pueda tomar libremente por un grupo de trabajo que lo mejore sin esfuerzo.

- Pruebas.

  Al tratarse éste de un software de aproximación numérica, las pruebas pueden no ser tan exigentes. La detección de errores se facilita cuando se trata de métodos numéricos tan delicados, pues cualquier defecto en los datos o su procesamiento concluyen en su colapso total. También, por ser el objetivo principal la visualización de los datos producidos, al observarse ésta de manera obvia y satisfactoria la posible existencia de errores resulta

sin importancia y por lo tanto es ignorada. Al detectarse fallos se producen iteraciones a los pasos 2 o 3. Sin embargo, la flexibilidad del proceso puede fusionar este cuarto paso al anterior o incluso a los dos anteriores, permitiéndose depurar el programa como mejor convenga. Sin embargo, el proceso sólo finaliza con la aprobación completa de las pruebas finales.

A continuación se describirá el proceso realizado así como la documentación producida por su seguimiento metódico.

3.1 Análisis

El planteamiento del problema es formulado en la primera parte 1.1, mientras que en capítulo 2 se presenta la materia en estudio con relativo detalle.

Las ecuaciones referenciadas anteriormente (1.1.3) conjuntamente con el resto del sistema deben ser desarrolladas a través del método LSFEM para buscar un estado óptimo en que se puedan aplicar algorítmicamente.

3.1.1 Desarrollo de las ecuaciones de Navier-Stokes

El problema presenta el sistema de ecuaciones llamadas de Navier-Stokes definido en el dominio Ω (3.1.1) que modelan el movimiento del fluido a través de un conducto poligonal. Este sistema de ecuaciones diferenciales, para ser procesado a través de un programa de computación, debe ser representado como un operador matricial (2.1.4) que sea resuelto en cada instante de tiempo en el que se ejecute la simulación. A continuación, las ecuaciones de Navier-Stokes adimensionales:

$$\begin{aligned} \frac{D\mathbf{v}}{\partial t} - \frac{1}{Re}\Delta\mathbf{v} + \nabla p &= \mathbf{0} \\ \nabla \cdot \mathbf{v} &= \mathbf{0} \end{aligned} \tag{3.1.1}$$

descritas en (1.1.2) .

Al escribir algebraicamente en dos dimensiones el sistema, se tiene:

$$\begin{aligned} \frac{\partial v_x}{\partial t} + v_x\frac{\partial v_x}{\partial x} + v_y\frac{\partial v_x}{\partial y} - \frac{1}{Re}\left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2}\right) + \frac{\partial p}{\partial x} &= 0 \\ \frac{\partial v_y}{\partial t} + v_x\frac{\partial v_y}{\partial x} + v_y\frac{\partial v_y}{\partial y} - \frac{1}{Re}\left(\frac{\partial^2 v_y}{\partial x^2} + \frac{\partial^2 v_y}{\partial y^2}\right) + \frac{\partial p}{\partial y} &= 0 \\ \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} &= 0 \end{aligned} \tag{3.1.2}$$

Según [17], LSFEM queda optimizado al procesar sistemas de primer orden. Para escribir el sistema de ecuaciones anteriores (3.1.2) como uno de primer orden se realiza el cambio de variable:

$$\mathbf{w} = \nabla \times \mathbf{v}, \tag{3.1.3}$$

es decir,

$$\mathbf{w} = \left(\frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y}\right)\widehat{k} = w\widehat{k}. \tag{3.1.4}$$

El cambio de la variable queda justificado al aplicar la propiedad (2.1.16) sobre

$$\Delta\mathbf{v} = \nabla\left(\nabla \cdot \mathbf{v}\right) - \nabla \times \nabla \times \mathbf{v} \tag{3.1.5}$$

resultando que

$$\Delta\mathbf{v} = -\nabla \times \nabla \times \mathbf{v} = -\nabla \times \mathbf{w} \tag{3.1.6}$$

donde

$$\nabla \times \mathbf{w} = \frac{\partial w}{\partial y}\widehat{i} - \frac{\partial w}{\partial x}\widehat{j} \tag{3.1.7}$$

Aplicando el cambio de variable (3.1.3) al sistema inicial (3.1.1) y agregando la ecuación de continuidad (1.1.4) tenemos el sistema:

$$\begin{aligned} \frac{D\mathbf{v}}{\partial t} + \frac{1}{Re}\nabla \times w + \nabla p &= \mathbf{0} \\ \mathbf{w} - \nabla \times \mathbf{v} &= 0 \\ \nabla \cdot \mathbf{v} &= 0 \end{aligned} \tag{3.1.8}$$

Es decir:

$$\begin{aligned}
\frac{\partial v_x}{\partial t} + v_x\frac{\partial v_x}{\partial x} + v_y\frac{\partial v_x}{\partial y} + \frac{1}{Re}\frac{\partial w}{\partial y} + \frac{\partial p}{\partial x} &= 0 \\
\frac{\partial v_y}{\partial t} + v_x\frac{\partial v_y}{\partial x} + v_y\frac{\partial v_y}{\partial y} - \frac{1}{Re}\frac{\partial w}{\partial x} + \frac{\partial p}{\partial y} &= 0 \\
w + \frac{\partial v_x}{\partial y} - \frac{\partial v_y}{\partial x} &= 0 \\
\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} &= 0
\end{aligned} \tag{3.1.9}$$

Se tiene aquí el modelado de un sistema a través del tiempo cuya continuidad es difícilmente representable computacionalmente. La discretización del tiempo se puede realizar a través de diferentes métodos. En este caso se utilizará el método theta 2.1.5. Sea cada paso de tiempo $\Delta t = t^{n+1} - t^n$ y $\mathbf{v}^n$ el valor de **v** para el paso actual $t = t^n$, la solución $(\mathbf{v}, p, \mathbf{w})^{n+1}$ para el paso siguiente está determinada por el sistema:

$$\begin{aligned}
\frac{\mathbf{v}^{n+1} - \mathbf{v}^n}{\Delta t} + \theta\left(\left(\mathbf{v}^{n+1}\cdot\nabla\right)\mathbf{v}^{n+1} + \nabla p^{n+1} + \frac{1}{Re}\nabla\times\mathbf{w}^{n+1}\right) + \\
(1-\theta)\left(\left(\mathbf{v}^{n}\cdot\nabla\right)\mathbf{v}^{n} + \nabla p^{n} + \frac{1}{Re}\nabla\times\mathbf{w}^{n}\right) = \mathbf{0} \\
\mathbf{w}^{n+1} - \nabla\times\mathbf{v}^{n+1} = 0 \\
\nabla\cdot\mathbf{v}^{n+1} = 0
\end{aligned} \tag{3.1.10}$$

Trasladando la formulación al problema en dos dimensiones tenemos:

$$\begin{aligned}
\frac{v_x^{n+1} - v_x^n}{\Delta t} + \theta\left(v_x^{n+1}\frac{\partial v_x^{n+1}}{\partial x} + v_y^{n+1}\frac{\partial v_x^{n+1}}{\partial y} + \frac{\partial p^{n+1}}{\partial x} + \frac{1}{Re}\frac{\partial w^{n+1}}{\partial y}\right) \\
+ (1-\theta)\left(v_x^{n}\frac{\partial v_x^{n}}{\partial x} + v_y^{n}\frac{\partial v_x^{n}}{\partial y} + \frac{\partial p^{n}}{\partial x} + \frac{1}{Re}\frac{\partial w^{n}}{\partial y}\right) = 0 \\
\frac{v_y^{n+1} - v_y^n}{\Delta t} + \theta\left(v_x^{n+1}\frac{\partial v_y^{n+1}}{\partial x} + v_y^{n+1}\frac{\partial v_y^{n+1}}{\partial y} + \frac{\partial p^{n+1}}{\partial y} - \frac{1}{Re}\frac{\partial w^{n+1}}{\partial x}\right) \\
+ (1-\theta)\left(v_x^{n}\frac{\partial v_y^{n}}{\partial x} + v_y^{n}\frac{\partial v_y^{n}}{\partial y} + \frac{\partial p^{n}}{\partial y} - \frac{1}{Re}\frac{\partial w^{n}}{\partial x}\right) = 0 \\
w^{n+1} + \frac{\partial v_x^{n+1}}{\partial y} - \frac{\partial v_y^{n+1}}{\partial x} = 0 \\
\frac{\partial v_x^{n+1}}{\partial x} + \frac{\partial v_y^{n+1}}{\partial y} = 0
\end{aligned} \tag{3.1.11}$$

Este sistema se adapta más a la expresión buscada, con la excepción de que presenta términos cuyos factores son ambos valores desconocidos. Es necesario linealizar la ecuación para que cada término corresponda a una localización única del operador matricial. Para ello se aproxima como se demuestra en 2.1.6, por ejemplo

$$v_x^{n+1}\frac{\partial v_y^{n+1}}{\partial x} \approx v_x^n\frac{\partial v_y^{n+1}}{\partial x} + v_x^{n+1}\frac{\partial v_y^n}{\partial x} - v_x^n\frac{\partial v_y^n}{\partial x} \tag{3.1.12}$$

El sistema linealizado resultante es:

$$\begin{aligned}
&\frac{v_x^{n+1}}{\Delta t} - \frac{v_x^n}{\Delta t} + \theta\left(v_x^n\frac{\partial v_x^{n+1}}{\partial x} + v_x^{n+1}\frac{\partial v_x^n}{\partial x} - v_x^n\frac{\partial v_x^n}{\partial x}\right) + \theta\left(v_y^n\frac{\partial v_x^{n+1}}{\partial y} + v_y^{n+1}\frac{\partial v_x^n}{\partial y} - v_y^n\frac{\partial v_x^n}{\partial y}\right) + \\
&\qquad \theta\left(\frac{\partial p^{n+1}}{\partial x} + \frac{1}{Re}\frac{\partial w^{n+1}}{\partial y}\right) + (1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x} + v_x^n\frac{\partial v_x^n}{\partial y} + \frac{\partial p^n}{\partial x} + \frac{1}{Re}\frac{\partial w^n}{\partial y}\right) = 0 \\
&\frac{v_y^{n+1}}{\Delta t} - \frac{v_y^n}{\Delta t} + \theta\left(v_x^n\frac{\partial v_y^{n+1}}{\partial x} + v_x^{n+1}\frac{\partial v_y^n}{\partial x} - v_x^n\frac{\partial v_y^n}{\partial x}\right) + \theta\left(v_y^n\frac{\partial v_y^{n+1}}{\partial y} + v_y^{n+1}\frac{\partial v_y^n}{\partial y} - v_y^n\frac{\partial v_y^n}{\partial y}\right) + \\
&\qquad \theta\left(\frac{\partial p^{n+1}}{\partial y} - \frac{1}{Re}\frac{\partial w^{n+1}}{\partial x}\right) + (1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x} + v_x^n\frac{\partial v_y^n}{\partial y} + \frac{\partial p^n}{\partial y} - \frac{1}{Re}\frac{\partial w^n}{\partial x}\right) = 0 \\
&\qquad\qquad w^{n+1} + \frac{\partial v_x^{n+1}}{\partial y} - \frac{\partial v_y^{n+1}}{\partial x} = 0 \\
&\qquad\qquad \frac{\partial v_x^{n+1}}{\partial x} + \frac{\partial v_y^{n+1}}{\partial y} = 0
\end{aligned} \tag{3.1.13}$$

De esta forma se pueden despejar los valores conocidos para facilitar el paso siguiente.

$$\begin{aligned}
&\frac{v_x^{n+1}}{\Delta t} + \theta\left(v_x^n\frac{\partial v_x^{n+1}}{\partial x} + v_x^{n+1}\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^{n+1}}{\partial y} + v_y^{n+1}\frac{\partial v_x^n}{\partial y} + \frac{\partial p^{n+1}}{\partial x} + \frac{1}{Re}\frac{\partial w^{n+1}}{\partial y}\right) = \\
&\qquad \theta\left(v_x^n\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^n}{\partial y}\right) + \frac{v_x^n}{\Delta t} - (1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^n}{\partial y} + \frac{\partial p^n}{\partial x} + \frac{1}{Re}\frac{\partial w^n}{\partial y}\right) \\
&\frac{v_y^{n+1}}{\Delta t} + \theta\left(v_x^n\frac{\partial v_y^{n+1}}{\partial x} + v_x^{n+1}\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^{n+1}}{\partial y} + v_y^{n+1}\frac{\partial v_y^n}{\partial y} + \frac{\partial p^{n+1}}{\partial y} - \frac{1}{Re}\frac{\partial w^{n+1}}{\partial x}\right) = \\
&\qquad \theta\left(v_x^n\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^n}{\partial y}\right) + \frac{v_y^n}{\Delta t} - (1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^n}{\partial y} + \frac{\partial p^n}{\partial y} - \frac{1}{Re}\frac{\partial w^n}{\partial x}\right) \\
&\qquad\qquad w^{n+1} + \frac{\partial v_x^{n+1}}{\partial y} - \frac{\partial v_y^{n+1}}{\partial x} = 0 \\
&\qquad\qquad \frac{\partial v_x^{n+1}}{\partial x} + \frac{\partial v_y^{n+1}}{\partial y} = 0
\end{aligned} \tag{3.1.14}$$

Así pues, la ecuación diferencial se puede representar como:

$$\mathbf{A}\mathbf{u} = \mathbf{f} \quad \text{en } \Omega \tag{3.1.15}$$

donde

$$\mathbf{u} = \begin{bmatrix} v_x \\ v_y \\ p \\ w \end{bmatrix}, \tag{3.1.16}$$

A es un operador diferencial definido como

$$\mathbf{A} = \mathbf{A}_0 + \mathbf{A}_1 \frac{\partial}{\partial x} + \mathbf{A}_2 \frac{\partial}{\partial y} \tag{3.1.17}$$

con

$$\begin{aligned} \mathbf{A}_0 &= \begin{bmatrix} \frac{1}{\Delta t} + \theta \frac{\partial v_x^n}{\partial x} & \theta \frac{\partial v_x^n}{\partial y} & 0 & 0 \\ \theta \frac{\partial v_y^n}{\partial x} & \frac{1}{\Delta t} + \theta \frac{\partial v_y^n}{\partial y} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ \mathbf{A}_1 &= \begin{bmatrix} \theta v_x^n & 0 & \theta & 0 \\ 0 & \theta v_x^n & 0 & -\theta \frac{1}{Re} \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \\ \mathbf{A}_2 &= \begin{bmatrix} \theta v_y^n & 0 & 0 & \theta \frac{1}{Re} \\ 0 & \theta v_y^n & \theta & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \end{aligned} \tag{3.1.18}$$

y

$$\mathbf{f} = \begin{bmatrix} \frac{v_x^n}{\Delta t} - (1-\theta)(v_x^n \frac{\partial v_x^n}{\partial x} + v_y^n \frac{\partial v_x^n}{\partial y} + \frac{\partial p^n}{\partial x} + \frac{1}{Re}\frac{\partial w^n}{\partial y}) + \theta(v_x^n \frac{\partial v_x^n}{\partial x} + v_y^n \frac{\partial v_x^n}{\partial y}) \\ \frac{v_y^n}{\Delta t} - (1-\theta)(v_x^n \frac{\partial v_y^n}{\partial x} + v_y^n \frac{\partial v_y^n}{\partial y} + \frac{\partial p^n}{\partial y} - \frac{1}{Re}\frac{\partial w^n}{\partial x}) + \theta(v_x^n \frac{\partial v_y^n}{\partial x} + v_y^n \frac{\partial v_y^n}{\partial y}) \\ 0 \\ 0 \end{bmatrix} \tag{3.1.19}$$

Las condiciones de frontera se representan en dos ecuaciones

$$\begin{cases} \mathbf{n} \cdot \mathbf{u} = 0 \\ p = p_0 \end{cases} \quad \text{en } \Gamma \tag{3.1.20}$$

aquí **n** representa el vector normal a la frontera $[n_x, n_y]$ y p_0 es un valor constante de presión. Este caso es mucho más simple que el anterior, ya que se puede representar directamente a un operador matricial.

Se define un operador $\mathbf{B}$ y un vector $\mathbf{g}$ que cumplen (2.1.6) tal que:

$$\mathbf{Bu} = \begin{bmatrix} n_x & n_y & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} v_x \\ v_y \\ p \\ w \end{bmatrix} \tag{3.1.21}$$

y

$$\mathbf{g} = \begin{bmatrix} 0 \\ p_0 \\ 0 \\ 0 \end{bmatrix} \tag{3.1.22}$$

3.1.2 Desarrollo de las ecuaciones siguiendo el método LSFEM

A continuación la expresión (3.1.17) se procesa siguiendo el método LSFEM. El método comienza a partir de la forma débil del sistema de ecuaciones diferenciales (2.3.14) que se escribe aplicando (3.1.17) con (3.1.18) , (3.1.19) y (3.1.21) , (3.1.22) . Dada la longitud de las ecuaciones, cada término en (2.3.14) se desarrolla por separado.

Se desarrolla la expresión (2.3.10) :

$$\begin{aligned}
(\mathbf{Au}, \mathbf{A\Psi}_i) &= \int_\Omega \left\{ \left(\mathbf{A}_0 + \mathbf{A}_1 \tfrac{\partial}{\partial x} + \mathbf{A}_2 \tfrac{\partial}{\partial y} \right) \mathbf{\Psi} \right\}^T \left\{ \left(\mathbf{A}_0 + \mathbf{A}_1 \tfrac{\partial}{\partial x} + \mathbf{A}_2 \tfrac{\partial}{\partial y} \right) \mathbf{u} \right\} d\Omega \\
&= \int_\Omega \Bigg\{ \left(\tfrac{\Psi_1}{\Delta t} + \theta \Psi_1 \tfrac{\partial v_x^n}{\partial x} + \theta \Psi_2 \tfrac{\partial v_x^n}{\partial y} + \theta v_x^n \tfrac{\partial \Psi_1}{\partial x} + \theta \tfrac{\partial \Psi_3}{\partial x} + \theta v_y^n \tfrac{\partial \Psi_1}{\partial y} + \theta \tfrac{1}{Re} \tfrac{\partial \Psi_4}{\partial y} \right) \\
&\left(\tfrac{v_x^{n+1}}{\Delta t} + \theta v_x^{n+1} \tfrac{\partial v_x^n}{\partial x} + \theta v_y^{n+1} \tfrac{\partial v_x^n}{\partial y} + \theta v_x^n \tfrac{\partial v_x^{n+1}}{\partial x} + \theta \tfrac{\partial p^{n+1}}{\partial x} + \theta v_y^n \tfrac{\partial v_x^{n+1}}{\partial y} + \theta \tfrac{1}{Re} \tfrac{\partial w^{n+1}}{\partial y} \right) + \\
&\left(\theta \Psi_1 \tfrac{\partial v_y^n}{\partial x} + \tfrac{\Psi_2}{\Delta t} + \theta \Psi_2 \tfrac{\partial v_y^n}{\partial y} + \theta v_x^n \tfrac{\partial \Psi_2}{\partial x} - \theta \tfrac{1}{Re} \tfrac{\partial \Psi_4}{\partial x} + \theta v_y^n \tfrac{\partial \Psi_2}{\partial y} + \theta \tfrac{\partial \Psi_3}{\partial x} \right) \\
&\left(\theta v_x^{n+1} \tfrac{\partial v_y^n}{\partial x} + \tfrac{v_y^{n+1}}{\Delta t} + \theta v_y^{n+1} \tfrac{\partial v_y^n}{\partial y} + \theta v_x^n \tfrac{\partial v_y^{n+1}}{\partial x} - \theta \tfrac{1}{Re} \tfrac{\partial w^{n+1}}{\partial x} + \theta v_y^n \tfrac{\partial v_y^{n+1}}{\partial y} + \theta \tfrac{\partial p^{n+1}}{\partial x} \right) + \\
&\left(\tfrac{\partial \Psi_1}{\partial x} + \tfrac{\partial \Psi_2}{\partial y} \right) \left(\tfrac{\partial v_x^{n+1}}{\partial x} + \tfrac{\partial v_y^{n+1}}{\partial y} \right) + \left(\Psi_4 - \tfrac{\partial \Psi_2}{\partial x} + \tfrac{\partial \Psi_1}{\partial y} \right) \left(w^{n+1} - \tfrac{\partial v_y^{n+1}}{\partial x} + \tfrac{\partial v_x^{n+1}}{\partial y} \right) \Bigg\} d\Omega
\end{aligned} \tag{3.1.23}$$

Similarmente la expresión (2.3.11) :

$$\begin{aligned}(\mathbf{f},\mathbf{A}\boldsymbol{\Psi}_i) &= \int_\Omega \left(\left(\mathbf{A}_0+\mathbf{A}_1\tfrac{\partial}{\partial x}+\mathbf{A}_2\tfrac{\partial}{\partial y}\right)\mathbf{v}\right)^T \mathbf{f}\,d\Omega \\ &= \int_\Omega\Bigg\{\left(\tfrac{\Psi_1}{\Delta t}+\theta\Psi_1\tfrac{\partial v_x^n}{\partial x}+\theta\Psi_2\tfrac{\partial v_x^n}{\partial y}+\theta v_x^n\tfrac{\partial\Psi_1}{\partial x}+\theta\tfrac{\partial\Psi_3}{\partial x}+\theta v_y^n\tfrac{\partial\Psi_1}{\partial y}+\theta\tfrac{1}{Re}\tfrac{\partial\Psi_4}{\partial y}\right)\\ &\left(\tfrac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\tfrac{\partial v_x^n}{\partial x}+v_y^n\tfrac{\partial v_x^n}{\partial y}+\theta\tfrac{\partial p^n}{\partial x}+\tfrac{1}{Re}\tfrac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\tfrac{\partial v_x^n}{\partial x}+v_y^n\tfrac{\partial v_x^n}{\partial y}\right)\right)+\\ &\left(\theta\Psi_1\tfrac{\partial v_y^n}{\partial x}+\tfrac{\Psi_2}{\Delta t}+\theta\Psi_2\tfrac{\partial v_y^n}{\partial y}+\theta v_x^n\tfrac{\partial\Psi_2}{\partial x}-\theta\tfrac{1}{Re}\tfrac{\partial\Psi_4}{\partial x}+\theta v_y^n\tfrac{\partial\Psi_2}{\partial y}+\theta\tfrac{\partial\Psi_3}{\partial x}\right)\\ &\left(\tfrac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\tfrac{\partial v_y^n}{\partial x}+v_y^n\tfrac{\partial v_y^n}{\partial y}+\theta\tfrac{\partial p^n}{\partial y}+\tfrac{1}{Re}\tfrac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\tfrac{\partial v_y^n}{\partial x}+v_y^n\tfrac{\partial v_y^n}{\partial y}\right)\right)\Bigg\}d\Omega\end{aligned} \tag{3.1.24}$$

Para la frontera es más simple, las expresiones (2.3.12) , (2.3.13) se resumen como:

$$\langle\mathbf{Bu},\mathbf{B}\boldsymbol{\Psi}\rangle_\Gamma = \oint_\Gamma\left\{(n_x\Psi_1+n_y\Psi_2)(n_xv_x+n_yv_y)+\Psi_3p\right\}d\Gamma \tag{3.1.25}$$

$$\langle\mathbf{g},\mathbf{B}\boldsymbol{\Psi}\rangle_\Gamma = \oint_\Gamma \Psi_3p_0d\Gamma \tag{3.1.26}$$

La aplicación de (2.3.1) sobre (3.1.23) , (3.1.24) , (3.1.25) , (3.1.26) para LSFEM utilizando valores reales crea en este caso 4 familias de ecuaciones que contribuirán para la resolución de las 4 funciones incógnitas buscadas. La primera familia se obtiene efectuando la sustitución

$$\boldsymbol{\Psi}_i=\begin{bmatrix}\Psi_{1i}\\ \Psi_{2i}\\ \Psi_{3i}\\ \Psi_{4i}\end{bmatrix}=\phi_i\begin{bmatrix}1\\0\\0\\0\end{bmatrix} \tag{3.1.27}$$

, donde $i\in\{1,2,....,N\}$ para cada nodo del dominio, y se escribe

$$\begin{aligned}(\mathbf{Au},\mathbf{A}\boldsymbol{\Psi}_i) &= \int_\Omega\Bigg\{\left(\tfrac{\phi_i}{\Delta t}+\theta\phi_i\tfrac{\partial v_x^n}{\partial x}+\theta v_x^n\tfrac{\partial\phi_i}{\partial x}+\theta v_y^n\tfrac{\partial\phi_i}{\partial y}\right)\\ &\left(\tfrac{v_x^{n+1}}{\Delta t}+\theta v_x^{n+1}\tfrac{\partial v_x^n}{\partial x}+\theta v_y^{n+1}\tfrac{\partial v_x^n}{\partial y}+\theta v_x^n\tfrac{\partial v_x^{n+1}}{\partial x}+\theta\tfrac{\partial p^{n+1}}{\partial x}+\theta v_y^n\tfrac{\partial v_x^{n+1}}{\partial y}+\theta\tfrac{1}{Re}\tfrac{\partial w^{n+1}}{\partial y}\right)\\ &+\left(\theta\phi_i\tfrac{\partial v_y^n}{\partial x}\right)\left(\theta v_x^{n+1}\tfrac{\partial v_y^n}{\partial x}+\tfrac{v_y^{n+1}}{\Delta t}+\theta v_y^{n+1}\tfrac{\partial v_y^n}{\partial y}+\theta v_x^n\tfrac{\partial v_y^{n+1}}{\partial x}-\theta\tfrac{1}{Re}\tfrac{\partial w^{n+1}}{\partial x}+\theta v_y^n\tfrac{\partial v_y^{n+1}}{\partial y}+\theta\tfrac{\partial p^{n+1}}{\partial x}\right)\\ &+\tfrac{\partial\phi_i}{\partial y}\left(w^{n+1}-\tfrac{\partial v_y^{n+1}}{\partial x}+\tfrac{\partial v_x^{n+1}}{\partial y}\right)+\tfrac{\partial\phi_i}{\partial x}\left(\tfrac{\partial v_x^{n+1}}{\partial x}+\tfrac{\partial v_y^{n+1}}{\partial y}\right)\Bigg\}d\Omega\end{aligned} \tag{3.1.28}$$

$$\begin{aligned}(\mathbf{f},\mathbf{A}\boldsymbol{\Psi}_i) &= \int_\Omega\Bigg\{\left(\tfrac{\phi_i}{\Delta t}+\theta\phi_i\tfrac{\partial v_x^n}{\partial x}+\theta v_x^n\tfrac{\partial\phi_i}{\partial x}+\theta v_y^n\tfrac{\partial\phi_i}{\partial y}\right)\\ &\left(\tfrac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\tfrac{\partial v_x^n}{\partial x}+v_y^n\tfrac{\partial v_x^n}{\partial y}+\theta\tfrac{\partial p^n}{\partial x}+\tfrac{1}{Re}\tfrac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\tfrac{\partial v_x^n}{\partial x}+v_y^n\tfrac{\partial v_x^n}{\partial y}\right)\right)\\ &+\left(\theta\phi_i\tfrac{\partial v_y^n}{\partial x}\right)\left(\tfrac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\tfrac{\partial v_y^n}{\partial x}+v_y^n\tfrac{\partial v_y^n}{\partial y}+\theta\tfrac{\partial p^n}{\partial y}+\tfrac{1}{Re}\tfrac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\tfrac{\partial v_y^n}{\partial x}+v_y^n\tfrac{\partial v_y^n}{\partial y}\right)\right)\Bigg\}d\Omega\end{aligned} \tag{3.1.29}$$

$$\langle \mathbf{Bu}, \mathbf{B\Psi} \rangle_\Gamma = \oint_\Gamma n_x \phi_i \left(n_x v_x + n_y v_y \right) d\Gamma \tag{3.1.30}$$

$$\langle \mathbf{g}, \mathbf{B\Psi} \rangle_\Gamma = 0 \tag{3.1.31}$$

Para

$$\mathbf{\Psi}_i = \begin{bmatrix} \Psi_{1i} \\ \Psi_{2i} \\ \Psi_{3i} \\ \Psi_{4i} \end{bmatrix} = \phi_i \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.1.32}$$

tenemos la segunda familia de ecuaciones

$$\begin{gathered} (\mathbf{Au}, \mathbf{A\Psi}_i) = \int_\Omega \Bigg\{ \theta \phi_i \frac{\partial v_x^n}{\partial y} \left(\frac{v_x^{n+1}}{\Delta t} + \theta v_x^{n+1} \frac{\partial v_x^n}{\partial x} + \theta v_y^{n+1} \frac{\partial v_x^n}{\partial y} + \theta v_x^n \frac{\partial v_x^{n+1}}{\partial x} + \right. \\ \left. \theta \frac{\partial p^{n+1}}{\partial x} + \theta v_y^n \frac{\partial v_x^{n+1}}{\partial y} + \theta \frac{1}{Re} \frac{\partial w^{n+1}}{\partial y} \right) + \left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_2}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \\ \left(\theta v_x^{n+1} \frac{\partial v_y^n}{\partial x} + \frac{v_y^{n+1}}{\Delta t} + \theta v_y^{n+1} \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial v_y^{n+1}}{\partial x} - \theta \frac{1}{Re} \frac{\partial w^{n+1}}{\partial x} + \theta v_y^n \frac{\partial v_y^{n+1}}{\partial y} + \theta \frac{\partial p^{n+1}}{\partial x} \right) \\ + \frac{\partial \phi_i}{\partial x} \left(w^{n+1} - \frac{\partial v_y^{n+1}}{\partial x} + \frac{\partial v_x^{n+1}}{\partial y} \right) + \frac{\partial \phi_i}{\partial y} \left(\frac{\partial v_x^{n+1}}{\partial x} + \frac{\partial v_y^{n+1}}{\partial y} \right) \Bigg\} d\Omega \end{gathered} \tag{3.1.33}$$

$$\begin{gathered} (\mathbf{f}, \mathbf{A\Psi}_i) = \int_\Omega \Bigg\{ \theta \phi_i \frac{\partial v_x^n}{\partial y} \left(\frac{v_x^n}{\Delta t} - (1-\theta) \left(v_x^n \frac{\partial v_x^n}{\partial x} + v_y^n \frac{\partial v_x^n}{\partial y} + \theta \frac{\partial p^n}{\partial x} + \frac{1}{Re} \frac{\partial w^n}{\partial y} \right) + \right. \\ \left. \theta \left(v_x^n \frac{\partial v_x^n}{\partial x} + v_y^n \frac{\partial v_x^n}{\partial y} \right) \right) + \left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \\ \left(\frac{v_y^n}{\Delta t} - (1-\theta) \left(v_x^n \frac{\partial v_y^n}{\partial x} + v_y^n \frac{\partial v_y^n}{\partial y} + \theta \frac{\partial p^n}{\partial y} + \frac{1}{Re} \frac{\partial w^n}{\partial x} \right) + \theta \left(v_x^n \frac{\partial v_y^n}{\partial x} + v_y^n \frac{\partial v_y^n}{\partial y} \right) \right) \Bigg\} d\Omega \end{gathered} \tag{3.1.34}$$

$$\langle \mathbf{Bu}, \mathbf{B\Psi} \rangle_\Gamma = \oint_\Gamma n_y \phi_i \left(n_x v_x + n_y v_y \right) d\Gamma \tag{3.1.35}$$

$$\langle \mathbf{g}, \mathbf{B\Psi} \rangle_\Gamma = 0 \tag{3.1.36}$$

De manera similar, con

$$\mathbf{\Psi}_i = \begin{bmatrix} \Psi_{1i} \\ \Psi_{2i} \\ \Psi_{3i} \\ \Psi_{4i} \end{bmatrix} = \phi_i \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.1.37}$$

se tiene la tercera familia de ecuaciones expresada como

$$\begin{gathered} (\mathbf{Au}, \mathbf{A\Psi}_i) = \int_\Omega \Bigg\{ \theta \frac{\partial \phi_i}{\partial x} \left(\frac{v_x^{n+1}}{\Delta t} + \theta v_x^{n+1} \frac{\partial v_x^n}{\partial x} + \theta v_y^{n+1} \frac{\partial v_x^n}{\partial y} + \theta v_x^n \frac{\partial v_x^{n+1}}{\partial x} \right. \\ + \theta \frac{\partial p^{n+1}}{\partial x} + \theta v_y^n \frac{\partial v_x^{n+1}}{\partial y} + \theta \frac{1}{Re} \frac{\partial w^{n+1}}{\partial y} + \theta v_x^{n+1} \frac{\partial v_y^n}{\partial x} + \frac{v_y^{n+1}}{\Delta t} \\ \left. + \theta v_y^{n+1} \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial v_y^{n+1}}{\partial x} - \theta \frac{1}{Re} \frac{\partial w^{n+1}}{\partial x} + \theta v_y^n \frac{\partial v_y^{n+1}}{\partial y} + \theta \frac{\partial p^{n+1}}{\partial x} \right) \Bigg\} d\Omega \end{gathered} \tag{3.1.38}$$

$$(\mathbf{f},\mathbf{A}\boldsymbol{\Psi}_i)=\int_\Omega\Big\{\theta\frac{\partial\phi_i}{\partial x}\Big(\frac{v_x^n}{\Delta t}-(1-\theta)\Big(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\Big)+\theta\Big(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\Big)$$
$$\frac{v_y^n}{\Delta t}-(1-\theta)\Big(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\Big)+\theta\Big(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\Big)\Big)\Big\}d\Omega \tag{3.1.39}$$

$$\langle\mathbf{Bu},\mathbf{B}\boldsymbol{\Psi}\rangle_\Gamma=\oint_\Gamma\phi_i p d\Gamma \tag{3.1.40}$$

$$\langle\mathbf{g},\mathbf{B}\boldsymbol{\Psi}\rangle_\Gamma=\oint_\Gamma\phi_i p_0 d\Gamma \tag{3.1.41}$$

y, por último con la sustitución

$$\boldsymbol{\Psi}_i=\begin{bmatrix}\Psi_{1i}\\ \Psi_{2i}\\ \Psi_{3i}\\ \Psi_{4i}\end{bmatrix}=\phi_i\begin{bmatrix}0\\0\\0\\1\end{bmatrix} \tag{3.1.42}$$

obtenemos la cuarta familia de ecuaciones

$$(\mathbf{Au},\mathbf{A}\boldsymbol{\Psi}_i)=\int_\Omega\Big\{\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial y}\Big(\frac{v_x^{n+1}}{\Delta t}+\theta v_x^{n+1}\frac{\partial v_x^n}{\partial x}+\theta v_y^{n+1}\frac{\partial v_x^n}{\partial y}+\theta v_x^n\frac{\partial v_x^{n+1}}{\partial x}$$
$$+\theta\frac{\partial p^{n+1}}{\partial x}+\theta v_y^n\frac{\partial v_x^{n+1}}{\partial y}+\theta\frac{1}{Re}\frac{\partial w^{n+1}}{\partial y}-\theta v_x^{n+1}\frac{\partial v_y^n}{\partial x}\Big)-\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial x}\Big(\frac{v_y^{n+1}}{\Delta t}-\theta v_y^{n+1}\frac{\partial v_y^n}{\partial y} \tag{3.1.43}$$
$$-\theta v_x^n\frac{\partial v_y^{n+1}}{\partial x}+\theta\frac{1}{Re}\frac{\partial w^{n+1}}{\partial x}-\theta v_y^n\frac{\partial v_y^{n+1}}{\partial y}-\theta\frac{\partial p^{n+1}}{\partial x}\Big)\Big\}d\Omega$$

$$(\mathbf{f},\mathbf{A}\boldsymbol{\Psi}_i)=\int_\Omega\Big\{\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial y}\Big(\frac{v_x^n}{\Delta t}-(1-\theta)\Big(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\Big)$$
$$+\theta\Big(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\Big)\Big)-\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial x}\Big(\frac{v_y^n}{\Delta t}-(1-\theta)\Big(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\Big) \tag{3.1.44}$$
$$+\theta\Big(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\Big)\Big)\Big\}d\Omega$$

$$\langle\mathbf{Bu},\mathbf{B}\boldsymbol{\Psi}\rangle_\Gamma=0 \tag{3.1.45}$$

$$\langle\mathbf{g},\mathbf{B}\boldsymbol{\Psi}\rangle_\Gamma=0 \tag{3.1.46}$$

Mientras que los términos (3.1.29) , (3.1.34) , (3.1.39) , (3.1.44) , (3.1.31) , (3.1.36) ,(3.1.41) ,(3.1.46) permanecen igual, se dan las aproximaciones de $\widetilde{\mathbf{u}}$, originadas por (2.2.2) :

$$\widetilde{v}_x=\sum_{j=0}^{n}c_{1j}\phi_j,\ \widetilde{v}_y=\sum_{j=0}^{n}c_{2j}\phi_j,\ \widetilde{p}=\sum_{j=0}^{n}c_{3j}\phi_j,\ \widetilde{w}=\sum_{j=0}^{n}c_{4j}\phi_j$$

definidas como combinaciones lineales de las funciones de base $\{\phi_0,\phi_1,...\phi_n\}$, que serán

sustituidas en las expresiones (3.1.28) ,(3.1.33) ,(3.1.38) ,(3.1.43)

$$\left(\mathbf{A}\widetilde{\mathbf{u}},\mathbf{A}\widetilde{\mathbf{\Psi}}_i\right)=\int_\Omega\Bigg\{\left(\frac{\phi_i}{\Delta t}+\theta\phi_i\frac{\partial v_x^n}{\partial x}+\theta v_x^n\frac{\partial\phi_i}{\partial x}+\theta v_y^n\frac{\partial\phi_i}{\partial y}\right)$$
$$\sum_{j=0}^n\left(\frac{c_{1j}\phi_j}{\Delta t}+\theta c_{1j}\phi_j\frac{\partial v_x^n}{\partial x}+\theta c_{2j}\phi_j\frac{\partial v_x^n}{\partial y}+\theta v_x^n c_{1j}\frac{\partial\phi_j}{\partial x}+\theta c_{3j}\frac{\partial\phi_j}{\partial x}+\theta v_y^n c_{1j}\frac{\partial\phi_j}{\partial y}+\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial y}\right)$$
$$+\left(\theta\phi_i\frac{\partial v_y^n}{\partial x}\right)\sum_{j=0}^n\Bigg(\theta c_{1j}\phi_j\frac{\partial v_y^n}{\partial x}+c_{2j}\frac{\phi_j}{\Delta t}+\theta c_{2j}\phi_j\frac{\partial v_y^n}{\partial y}+\theta v_x^n c_{2j}\frac{\partial\phi_j}{\partial x}-\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial x}+\theta v_y^n c_{2j}\frac{\partial\phi_j}{\partial y}$$
$$+\theta c_{3j}\frac{\partial\phi_j}{\partial x}\Bigg)+\frac{\partial\phi_i}{\partial y}\sum_{j=0}^n\left(c_{4j}\phi_j-c_{2j}\frac{\partial\phi_j}{\partial x}+c_{1j}\frac{\partial\phi_j}{\partial y}\right)+\frac{\partial\phi_i}{\partial x}\sum_{j=0}^n\left(c_{1j}\frac{\partial\phi_j}{\partial x}+c_{2j}\frac{\partial\phi_j}{\partial y}\right)\Bigg\}d\Omega \tag{3.1.47}$$

$$\left(\mathbf{A}\widetilde{\mathbf{u}},\mathbf{A}\widetilde{\mathbf{\Psi}}_i\right)=\int_\Omega\Bigg\{\theta\phi_i\frac{\partial v_x^n}{\partial y}\sum_{j=0}^n\Bigg(c_{1j}\frac{\phi_j}{\Delta t}+\theta c_{1j}\phi_j\frac{\partial v_x^n}{\partial x}+\theta c_{2j}\phi_j\frac{\partial v_x^n}{\partial y}+\theta v_x^n c_{1j}\frac{\partial\phi_j}{\partial x}$$
$$+\theta c_{3j}\frac{\partial\phi_j}{\partial x}+\theta v_y^n c_{1j}\frac{\partial\phi_j}{\partial y}+\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial y}\Bigg)+\left(\frac{\phi_i}{\Delta t}+\theta\phi_i\frac{\partial v_y^n}{\partial y}+\theta v_x^n\frac{\partial\phi_i}{\partial x}+\theta v_y^n\frac{\partial\phi_i}{\partial y}\right)$$
$$\sum_{j=0}^n\left(\theta c_{1j}\phi_j\frac{\partial v_y^n}{\partial x}+\frac{c_{2j}\phi_j}{\Delta t}+\theta c_{2j}\phi_j\frac{\partial v_y^n}{\partial y}+\theta v_x^n c_{2j}\frac{\partial\phi_j}{\partial x}-\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial x}+\theta v_y^n c_{2j}\frac{\partial\phi_j}{\partial y}+\theta c_{3j}\frac{\partial\phi_j}{\partial x}\right)$$
$$+\frac{\partial\phi_i}{\partial x}\sum_{j=0}^n\left(c_{4j}\phi_j-c_{2j}\frac{\partial\phi_j}{\partial x}+c_{1j}\frac{\partial\phi_j}{\partial y}\right)+\frac{\partial\phi_i}{\partial y}\sum_{j=0}^n\left(c_{1j}\frac{\partial\phi_j}{\partial x}+c_{2j}\frac{\partial\phi_j}{\partial y}\right)\Bigg\}d\Omega \tag{3.1.48}$$

$$\left(\mathbf{A}\widetilde{\mathbf{u}},\mathbf{A}\widetilde{\mathbf{\Psi}}_i\right)=\int_\Omega\Bigg\{\theta\frac{\partial\phi_i}{\partial x}\sum_{j=0}^n\Bigg(c_{1j}\frac{\phi_j}{\Delta t}+\theta c_{1j}\phi_j\frac{\partial v_x^n}{\partial x}+\theta c_{2j}\phi_j\frac{\partial v_x^n}{\partial y}+\theta v_x^n c_{1j}\frac{\partial\phi_j}{\partial x}+$$
$$\theta c_{3j}\frac{\partial\phi_j}{\partial x}+\theta v_y^n c_{1j}\frac{\partial\phi_j}{\partial y}+\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial y}+\theta c_{1j}\phi_j\frac{\partial v_y^n}{\partial x}+c_{2j}\frac{\phi_j}{\Delta t}+$$
$$\theta c_{2j}\phi_j\frac{\partial v_y^n}{\partial y}+\theta v_x^n c_{2j}\frac{\partial\phi_j}{\partial x}-\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial x}+\theta v_y^n c_{2j}\frac{\partial\phi_j}{\partial y}+\theta c_{3j}\frac{\partial\phi_j}{\partial x}\Bigg)\Bigg\}d\Omega \tag{3.1.49}$$

$$\left(\mathbf{A}\widetilde{\mathbf{u}},\mathbf{A}\widetilde{\mathbf{\Psi}}_i\right)=\int_\Omega\Bigg\{\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial y}\sum_{j=0}^n\Bigg(\frac{c_{1j}\phi_j}{\Delta t}+\theta c_{1j}\phi_j\frac{\partial v_x^n}{\partial x}+$$
$$\theta c_{2j}\phi_j\frac{\partial v_x^n}{\partial y}+\theta v_x^n c_{2j}\frac{\partial\phi_j}{\partial x}+\theta c_{3j}\frac{\partial\phi_j}{\partial x}+\ \theta v_y^n c_{1j}\frac{\partial\phi_j}{\partial y}+\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial y}\Bigg)+$$
$$\left(-\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial x}\right)\sum_{j=0}^n\Bigg(\theta c_{1j}\phi_j\frac{\partial v_y^n}{\partial x}+c_{2j}\frac{\phi_j}{\Delta t}+\theta c_{2j}\phi_j\frac{\partial v_y^n}{\partial y}+\theta v_x^n c_{2j}\frac{\partial\phi_j}{\partial x}-$$
$$\theta\frac{1}{Re}c_{4j}\frac{\partial\phi_j}{\partial x}+\theta v_y^n c_{2j}\frac{\partial\phi_j}{\partial y}+\theta c_{3j}\frac{\partial\phi_j}{\partial x}\Bigg)\Bigg\}d\Omega \tag{3.1.50}$$

así como en (3.1.30) ,(3.1.35) ,(3.1.40) ,(3.1.45) :

$$\langle \mathbf{Bu}, \mathbf{B\Psi} \rangle_\Gamma = \oint_\Gamma n_x \phi_i \sum_{j=0}^{n} \left(n_x c_{1j} \phi_j + n_y c_{2j} \phi_j \right) d\Gamma \tag{3.1.51}$$

$$\langle \mathbf{Bu}, \mathbf{B\Psi} \rangle_\Gamma = \oint_\Gamma n_y \phi_i \sum_{j=0}^{n} \left(n_x c_{1j} \phi_j + n_y c_{2j} \phi_j \right) d\Gamma \tag{3.1.52}$$

$$\langle \mathbf{Bu}, \mathbf{B\Psi} \rangle_\Gamma = \oint_\Gamma \phi_i \sum_{j=0}^{n} \left(c_{3j} \phi_j \right) d\Gamma \tag{3.1.53}$$

$$\langle \mathbf{Bu}, \mathbf{B\Psi} \rangle_\Gamma = 0 \tag{3.1.54}$$

Factorizando, y generalizando las expresiones para cada elemento, se llega al fin a la expresión (2.2.18) en donde:

$$\begin{aligned}
K^e_{i1j1} &= \int_{\Omega_e} \left(\left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_x^n}{\partial x} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \left(\frac{\phi_j}{\Delta t} + \theta \phi_j \frac{\partial v_x^n}{\partial x} + \theta v_x^n \frac{\partial \phi_j}{\partial x} + \theta v_y^n \frac{\partial \phi_j}{\partial y} \right) + \right. \\
&\left. \phi_i \phi_j \left(\theta \frac{\partial v_y^n}{\partial x} \right)^2 + \frac{\partial \phi_i}{\partial y} \frac{\partial \phi_j}{\partial y} + \frac{\partial \phi_i}{\partial x} \frac{\partial \phi_j}{\partial x} \right) d\Omega_e + \oint_{\Gamma_e} n_x^2 \phi_i \phi_j d\Gamma_e \\
K^e_{i1j2} &= \int_{\Omega_e} \left(\left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_x^n}{\partial x} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \left(\theta \phi_j \frac{\partial v_x^n}{\partial y} \right) + \right. \\
&\left. \theta \phi_i \frac{\partial v_y^n}{\partial x} \left(\frac{\phi_j}{\Delta t} + \theta \phi_j \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_j}{\partial x} + \theta v_y^n \frac{\partial \phi_j}{\partial y} \right) - \frac{\partial \phi_i}{\partial y} \frac{\partial \phi_j}{\partial x} + \frac{\partial \phi_i}{\partial x} \frac{\partial \phi_j}{\partial y} \right) d\Omega_e + \oint_{\Gamma_e} n_x n_y \phi_i \phi_j d\Gamma_e \\
K^e_{i1j3} &= \int_{\Omega_e} \left(\left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_x^n}{\partial x} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \left(\theta \frac{\partial \phi_j}{\partial x} \right) + (\theta)^2 \phi_i \frac{\partial v_y^n}{\partial x} \frac{\partial \phi_j}{\partial x} \right) d\Omega_e \\
K^e_{i1j4} &= \int_{\Omega_e} \left(\left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_x^n}{\partial x} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \left(\theta \frac{1}{Re} \frac{\partial \phi_j}{\partial y} \right) - (\theta)^2 \phi_i \frac{1}{Re} \frac{\partial v_y^n}{\partial x} \frac{\partial \phi_j}{\partial x} + \phi_j \frac{\partial \phi_i}{\partial y} \right) d\Omega_e
\end{aligned}$$

$$\begin{aligned}
K^e_{i2j1} &= \int_{\Omega_e} \left(\theta \phi_i \frac{\partial v_x^n}{\partial y} \left(\frac{\phi_j}{\Delta t} + \theta \phi_j \frac{\partial v_x^n}{\partial x} + \theta v_x^n \frac{\partial \phi_j}{\partial x} + \theta v_y^n \frac{\partial \phi_j}{\partial y} \right) \right. \\
&\left. + \left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \left(\theta \phi_j \frac{\partial v_y^n}{\partial x} \right) + \frac{\partial \phi_i}{\partial x} \frac{\partial \phi_j}{\partial y} + \frac{\partial \phi_i}{\partial y} \frac{\partial \phi_j}{\partial x} \right) d\Omega_e \\
&+ \oint_{\Gamma_e} n_x n_y \phi_i \phi_j d\Gamma_e \\
K^e_{i2j2} &= \int_{\Omega_e} \left(\phi_i \phi_j \left(\theta \frac{\partial v_x^n}{\partial y} \right)^2 + \left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \right. \\
&\left. \left(\frac{\phi_j}{\Delta t} + \theta \phi_j \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_j}{\partial x} + \theta v_y^n \frac{\partial \phi_j}{\partial y} \right) - \frac{\partial \phi_i}{\partial x} \frac{\partial \phi_j}{\partial x} + \frac{\partial \phi_i}{\partial y} \frac{\partial \phi_j}{\partial y} \right) d\Omega_e + \oint_{\Gamma_e} n_x^2 \phi_i \phi_j d\Gamma_e \\
K^e_{i2j3} &= \int_{\Omega_e} \left(\theta^2 \phi_i \frac{\partial v_x^n}{\partial y} \frac{\partial \phi_j}{\partial x} + \left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_i}{\partial x} + \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \left(\theta \frac{\partial \phi_j}{\partial x} \right) \right) d\Omega_e \\
K^e_{i2j4} &= \int_{\Omega_e} \left(\theta^2 \phi_i \frac{1}{Re} \frac{\partial v_x^n}{\partial y} \frac{\partial \phi_j}{\partial y} + \left(\frac{\phi_i}{\Delta t} + \theta \phi_i \frac{\partial v_y^n}{\partial y} + \theta v_x^n \frac{\partial \phi_i}{\partial x} - \theta v_y^n \frac{\partial \phi_i}{\partial y} \right) \left(\theta \frac{1}{Re} \frac{\partial \phi_j}{\partial x} \right) + \phi_j \frac{\partial \phi_i}{\partial x} \right) d\Omega_e
\end{aligned}$$

$$
\begin{aligned}
K^e_{i3j1} &= \int_{\Omega_e}\left(\theta\frac{\partial\phi_i}{\partial x}\left(\frac{\phi_j}{\Delta t}+\theta\phi_j\frac{\partial v_x^n}{\partial x}+\theta v_x^n\frac{\partial\phi_j}{\partial x}+\theta v_y^n\frac{\partial\phi_j}{\partial y}+\theta\phi_j\frac{\partial v_y^n}{\partial x}\right)\right)d\Omega_e\\
K^e_{i3j2} &= \int_{\Omega_e}\left(\theta\frac{\partial\phi_i}{\partial x}\left(\theta\phi_j\frac{\partial v_x^n}{\partial y}+\frac{\phi_j}{\Delta t}+\theta\phi_j\frac{\partial v_y^n}{\partial y}+\theta v_x^n\frac{\partial\phi_j}{\partial x}+\theta v_y^n\frac{\partial\phi_j}{\partial y}\right)\right)d\Omega_e\\
K^e_{i3j3} &= \int_{\Omega_e}\left(2\theta^2\frac{\partial\phi_i}{\partial x}\frac{\partial\phi_j}{\partial x}\right)d\Omega_e+\oint_{\Gamma_e}\phi_i\phi_j d\Gamma_e\\
K^e_{i3j4} &= \int_{\Omega_e}\left(\theta^2\frac{\partial\phi_i}{\partial x}\left(\frac{1}{Re}\frac{\partial\phi_j}{\partial y}-\frac{1}{Re}\frac{\partial\phi_j}{\partial x}\right)\right)d\Omega_e
\end{aligned}
$$

$$
\begin{aligned}
K^e_{41j1} &= \int_{\Omega_e}\left(\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial y}\left(\frac{\phi_j}{\Delta t}+\theta\phi_j\frac{\partial v_x^n}{\partial x}+\theta v_x^n\frac{\partial\phi_j}{\partial x}+\theta v_y^n\frac{\partial\phi_j}{\partial y}\right)-\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial x}\left(\theta\phi_j\frac{\partial v_y^n}{\partial x}\right)\right)d\Omega_e\\
K^e_{i4j2} &= \int_{\Omega_e}\left(\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial y}\left(\theta\phi_j\frac{\partial v_x^n}{\partial y}+\theta v_x^n\frac{\partial\phi_j}{\partial x}\right)-\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial x}\left(\frac{\phi_j}{\Delta t}+\theta\phi_j\frac{\partial v_y^n}{\partial y}\theta v_x^n\frac{\partial\phi_j}{\partial x}+\theta v_y^n\frac{\partial\phi_j}{\partial y}\right)\right)d\Omega_e\\
K^e_{i4j3} &= \int_{\Omega_e}\left(\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial y}-\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial x}\right)\left(\theta\frac{\partial\phi_j}{\partial x}\right)d\Omega_e\\
K^e_{i4j4} &= \int_{\Omega_e}\left(\theta\frac{1}{Re}\right)^2\left(\frac{\partial\phi_i}{\partial y}\frac{\partial\phi_j}{\partial y}+\frac{\partial\phi_i}{\partial x}\frac{\partial\phi_j}{\partial x}\right)d\Omega_e
\end{aligned}
$$

y además

$$
\begin{aligned}
q_{i1} &= \int_{\Omega_e}\left\{\left(\frac{\phi_i}{\Delta t}+\theta\phi_i\frac{\partial v_x^n}{\partial x}+\theta v_x^n\frac{\partial\phi_i}{\partial x}+\theta v_y^n\frac{\partial\phi_i}{\partial y}\right)\right.\\
&\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)\\
&\left.+\left(\theta\phi_i\frac{\partial v_y^n}{\partial x}\right)\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\right\}d\Omega_e\\
q_{i2} &= \int_{\Omega_e}\left\{\theta\phi_i\frac{\partial v_x^n}{\partial y}\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\right.\right.\\
&\left.\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)+\left(\frac{\phi_i}{\Delta t}+\theta\phi_i\frac{\partial v_y^n}{\partial y}+\theta v_x^n\frac{\partial\phi_i}{\partial x}+\theta v_y^n\frac{\partial\phi_i}{\partial y}\right)\\
&\left.\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\right\}d\Omega_e\\
q_{i3} &= \int_{\Omega_e}\left\{\theta\frac{\partial\phi_i}{\partial x}\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)\right.\\
&\left.\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right\}d\Omega_e+\oint_{\Gamma_e}p_0\phi_i d\Gamma_e\\
q_{i4} &= \int_{\Omega_e}\left\{\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial y}\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)\right.\right.\\
&\left.+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)-\theta\frac{1}{Re}\frac{\partial\phi_i}{\partial x}\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)\right.\\
&\left.\left.+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\right\}d\Omega_e
\end{aligned}
$$

La solución de las integrales queda expresada según la notación en (2.4.11) .

$$
\begin{aligned}
K^e_{i1j1} &= \left(\left(\frac{1}{\Delta t}+\theta\frac{\partial v_x^n}{\partial x}\right)^2+\left(\theta\frac{\partial v_y^n}{\partial x}\right)^2\right)S_{ij}^{00}+\left(\left(\frac{1}{\Delta t}+\theta\frac{\partial v_x^n}{\partial x}\right)\theta v_x^n\right)\left(S_{ij}^{01}+S_{ij}^{10}\right)+\\
&\left(\left(\frac{1}{\Delta t}+\theta\frac{\partial v_x^n}{\partial x}\right)\theta v_y^n\right)\left(S_{ij}^{02}+S_{ij}^{20}\right)++\left((\theta v_x^n)^2+1\right)S_{ij}^{11}+\left(\theta^2 v_x^n v_y^n\right)\left(S_{ij}^{12}+S_{ij}^{21}\right)+\\
&\left(\left(\theta v_y^n\right)^2+1\right)S_{ij}^{22}+n_x^2 S^{00}_{\Gamma^e ij}\\
K^e_{i1j2} &= \left(\left(\frac{1}{\Delta t}+\theta\frac{\partial v_x^n}{\partial x}\right)\theta\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial v_y^n}{\partial x}\left(\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)\right)S_{ij}^{00}+\theta^2\frac{\partial v_y^n}{\partial x}v_x^n S_{ij}^{01}+\theta^2 v_y^n\frac{\partial v_y^n}{\partial x}S_{ij}^{02}\\
&\theta^2 v_x^n\frac{\partial v_x^n}{\partial y}S_{ij}^{10}-S_{ij}^{11}+\theta^2 v_y^n\frac{\partial v_x^n}{\partial y}S_{ij}^{20}+S_ij^{22}+n_x n_y S^{00}_{\Gamma^e ij}\\
K^e_{i1j3} &= \left(\frac{\theta}{\Delta t}+\theta^2\frac{\partial v_x^n}{\partial x}+\theta^2\frac{\partial v_y^n}{\partial x}\right)S_{ij}^{01}+\theta^2 v_x^n S_{ij}^{11}+\theta^2 v_y^n S_{ij}^{21}\\
K^e_{i1j4} &= -\frac{\theta^2}{Re}\frac{\partial v_y^n}{\partial x}S_{ij}^{01}+\frac{\theta}{Re}\left(\frac{1}{\Delta t}+\theta\frac{\partial v_x^n}{\partial x}\right)S_{ij}^{02}+\theta^2 v_x^n\frac{1}{Re}S_{ij}^{12}+S_{20}+\frac{\theta^2}{Re}v_y^n S_{22}
\end{aligned}
$$

$$
\begin{aligned}
K^e_{i2j1} &= \theta\left(\frac{\partial v_x^n}{\partial y}\left(\frac{1}{\Delta t}+\theta\frac{\partial v_x^n}{\partial x}\right)+\frac{\partial v_y^n}{\partial x}\left(\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)\right)S_{ij}^{00}+\theta^2 v_x^n\frac{\partial v_x^n}{\partial y}S_{ij}^{01}+\theta^2 v_y^n\frac{\partial v_x^n}{\partial y}S_{ij}^{02}+\\
&S_{ij}^{11}+\theta^2 v_y^n\frac{\partial v_y^n}{\partial x}S_{ij}^{21}+S_{ij}^{22}+n_x n_y S^{00}_{\Gamma^e ij}\\
K^e_{i2j2} &= \left(\left(\theta\frac{\partial v_x^n}{\partial y}\right)^2+\left(\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)^2\right)S_{ij}^{00}+\theta v_x^n\left(\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)\left(S_{ij}^{01}+S_{ij}^{10}\right)+\\
&\theta v_y^n\left(\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)\left(S_{ij}^{02}+S_{ij}^{20}\right)\\
&\left((\theta v_x^n)^2-1\right)S_{ij}^{11}+\left(\theta^2 v_x^n v_y^n\right)\left(S_{ij}^{12}+S_{ij}^{21}\right)+\left(\left(\theta v_y^n\right)^2+1\right)S_{ij}^{22}\\
K^e_{i2j3} &= \theta\left(\theta\frac{\partial v_x^n}{\partial y}+\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)S_{ij}^{01}+\theta^2 v_x^n S_{ij}^{11}+\theta^2 v_y^n S_{ij}^{21}\\
K^e_{i2j4} &= \left(\theta\frac{1}{Re}\left(\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)+1\right)S_{ij}^{01}+\theta^2\frac{1}{Re}\frac{\partial v_x^n}{\partial y}S_{ij}^{02}+\theta^2\frac{1}{Re}v_x^n S_{11}-\theta^2\frac{1}{Re}v_y^n S_{21}
\end{aligned}
$$

$$
\begin{aligned}
K^e_{i3j1} &= \left(\frac{\theta}{\Delta t} + \theta^2\frac{\partial v_x^n}{\partial x} + \theta^2\frac{\partial v_y^n}{\partial x}\right) S_{ij}^{00} + \theta^2 v_x^n S_{ij}^{01} + \theta^2 v_y^n S_{ij}^{02} \\
K^e_{i3j2} &= \left(\theta^2\frac{\partial v_x^n}{\partial y} + \theta^2\frac{\partial v_y^n}{\partial y} + \frac{1}{\Delta t}\right) S_{ij}^{10} + \theta^2 v_x^n S_{ij}^{11} + \theta^2 v_y^n S_{ij}^{12} \\
K^e_{i3j3} &= 2\theta^2 S_{ij}^{11} + S_{\Gamma^e ij}^{00} \\
K^e_{i3j4} &= \frac{\theta^2}{Re}\left(S_{ij}^{12} - S_{ij}^{11}\right)
\end{aligned}
$$

$$
\begin{aligned}
K^e_{i4j1} &= -\frac{\theta^2}{Re}\frac{\partial v_y^n}{\partial x} S_{ij}^{11} + \theta\frac{1}{Re}\left(\frac{1}{\Delta t} + \theta\frac{\partial v_x^n}{\partial x}\right) S_{ij}^{20} + \theta^2\frac{1}{Re} v_x^n S_{ij}^{21} + \theta^2\frac{1}{Re} v_y^n S_{ij}^{22} \\
K^e_{i4j2} &= -\frac{\theta}{Re}\left(\frac{1}{\Delta t} + \theta\frac{\partial v_y^n}{\partial y}\right) S_{ij}^{10} - \frac{\theta^2}{Re} v_x^n S_{ij}^{11} - \frac{\theta^2}{Re} v_y^n S_{ij}^{12} + \frac{\theta^2}{Re}\frac{\partial v_x^n}{\partial y} S_{ij}^{20} + \frac{\theta^2}{Re} v_x^n S_{ij}^{21} \\
K^e_{i4j3} &= \frac{\theta^2}{Re}\left(S_{ij}^{21} - S_{ij}^{11}\right) \\
K^e_{i4j4} &= \left(\frac{\theta}{Re}\right)^2 \left(S_{ij}^{11} + S_{ij}^{22}\right)
\end{aligned}
$$

El lado derecho quedaría:

$$
\begin{aligned}
q_{i1} &= \left(\frac{1}{\Delta t}+\theta\frac{\partial v_x^n}{\partial x}\right)\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)+\\
&\left(\theta\frac{\partial v_y^n}{\partial x}\right)\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\int_{\Omega^e}\phi_i^e d\Omega^e\\
&+\theta v_x^n\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)\int_{\Omega^e}\frac{\partial\phi_i}{\partial x}d\Omega^e+\\
&\theta v_y^n\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)\int_{\Omega_e}\frac{\partial\phi_i}{\partial y}d\Omega_e
\end{aligned}
$$

$$
\begin{aligned}
q_{i2} &= \left(\theta\frac{\partial v_x^n}{\partial y}\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\right.\right.\\
&\left.\left.\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)+\left(\frac{1}{\Delta t}+\theta\frac{\partial v_y^n}{\partial y}\right)\right.\\
&\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\int_{\Omega_e}\phi_i d\Omega_e+\\
&+(\theta v_x^n)\\
&\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\int_{\Omega_e}\frac{\partial\phi_i}{\partial x}d\Omega_e+\\
&+(\theta v_y^n)\\
&\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\int_{\Omega_e}\frac{\partial\phi_i}{\partial y}d\Omega_e+
\end{aligned}
$$

$$
\begin{aligned}
q_{i3} &= \theta\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right.\\
&\left.\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\int_{\Omega_e}\frac{\partial\phi_i}{\partial x}d\Omega_e\\
&+p_0\oint_{\Gamma_e}\phi_i d\Gamma_e
\end{aligned}
$$

$$
\begin{aligned}
q_{i4} &= -\theta\frac{1}{Re}\left(\frac{v_y^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}+\theta\frac{\partial p^n}{\partial y}+\frac{1}{Re}\frac{\partial w^n}{\partial x}\right)\right.\\
&\left.+\theta\left(v_x^n\frac{\partial v_y^n}{\partial x}+v_y^n\frac{\partial v_y^n}{\partial y}\right)\right)\int_{\Omega_e}\frac{\partial\phi_i}{\partial x}d\Omega_e+\\
&\theta\frac{1}{Re}\left(\frac{v_x^n}{\Delta t}-(1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}+\theta\frac{\partial p^n}{\partial x}+\frac{1}{Re}\frac{\partial w^n}{\partial y}\right)\right.\\
&\left.+\theta\left(v_x^n\frac{\partial v_x^n}{\partial x}+v_y^n\frac{\partial v_x^n}{\partial y}\right)\right)\int_{\Omega_e}\frac{\partial\phi_i}{\partial y}d\Omega_e
\end{aligned}
$$

3.1.3 Desarrollo de las ecuaciones de fluido incompresible

Con fines didácticos y de trabajo futuro se desarrollan a continuación las ecuaciones de Navier-Stokes en las que se incluye el parámetro de compresibilidad artificial, que se comenta en la sección 2.1.2.

$$\begin{aligned} \frac{D\mathbf{v}}{\partial t} - \frac{1}{Re}\Delta\mathbf{v} + \nabla p &= \mathbf{0} \\ \frac{1}{\beta}\frac{\partial p}{\partial t} + \nabla\cdot\mathbf{v} &= \mathbf{0} \end{aligned} \tag{3.1.55}$$

Haciendo el mismo cambio de variable que en (3.1.3) se tiene:

$$\begin{aligned} \frac{D\mathbf{v}}{\partial t} + \frac{1}{Re}\nabla\times w + \nabla p &= \mathbf{0} \\ \frac{1}{\beta}\frac{\partial p}{\partial t} + \nabla\cdot\mathbf{v} &= 0 \\ \mathbf{w} - \nabla\times\mathbf{v} &= 0 \end{aligned} \tag{3.1.56}$$

La discretización se realiza de igual forma que en la sección anterior, utilizando el método θ. En esta ocasión quedan involucradas dos ecuaciones.

$$\begin{aligned} \frac{\mathbf{v}^{n+1}-\mathbf{v}^n}{\Delta t} + \theta\left(\left(\mathbf{v}^{n+1}\cdot\nabla\right)\mathbf{v}^{n+1} + \nabla p^{n+1} + \frac{1}{Re}\nabla\times\mathbf{w}^{n+1}\right) + \\ (1-\theta)\left(\left(\mathbf{v}^{n}\cdot\nabla\right)\mathbf{v}^{n} + \nabla p^{n} + \frac{1}{Re}\nabla\times\mathbf{w}^{n}\right) = \mathbf{0} \\ \frac{1}{\beta}\frac{p^{n+1}-p^n}{\Delta t} + \theta\left(\nabla\cdot\mathbf{v}^n\right) + (1-\theta)\left(\nabla\cdot\mathbf{v}^{n+1}\right) = 0 \\ \mathbf{w}^{n+1} - \nabla\times\mathbf{v}^{n+1} = 0 \end{aligned} \tag{3.1.57}$$

Trasladando la formulación al problema en dos dimensiones, y linealizando a través del método de ya visto tenemos:

$$
\begin{aligned}
\frac{v_x^{n+1}}{\Delta t} + \theta\left(v_x^n\frac{\partial v_x^{n+1}}{\partial x} + v_x^{n+1}\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^{n+1}}{\partial y} + v_y^{n+1}\frac{\partial v_x^n}{\partial y} + \frac{\partial p^{n+1}}{\partial x} + \frac{1}{Re}\frac{\partial w^{n+1}}{\partial y}\right) &= \\
\theta\left(v_x^n\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^n}{\partial y}\right) + \frac{v_x^n}{\Delta t} - (1-\theta)\left(v_x^n\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^n}{\partial y} + \frac{\partial p^n}{\partial x} + \frac{1}{Re}\frac{\partial w^n}{\partial y}\right) &\\
\frac{v_y^{n+1}}{\Delta t} + \theta\left(v_x^n\frac{\partial v_y^{n+1}}{\partial x} + v_x^{n+1}\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^{n+1}}{\partial y} + v_y^{n+1}\frac{\partial v_y^n}{\partial y} + \frac{\partial p^{n+1}}{\partial y} - \frac{1}{Re}\frac{\partial w^{n+1}}{\partial x}\right) &= \\
\theta\left(v_x^n\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^n}{\partial y}\right) + \frac{v_y^n}{\Delta t} - (1-\theta)\left(v_x^n\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^n}{\partial y} + \frac{\partial p^n}{\partial y} - \frac{1}{Re}\frac{\partial w^n}{\partial x}\right) &\\
\frac{1}{\beta}\frac{p^{n+1}}{\Delta t} + (1-\theta)(\frac{\partial v_x^{n+1}}{\partial x} + \frac{\partial v_y^{n+1}}{\partial y}) &= \\
\frac{1}{\beta}\frac{p^n}{\Delta t} - \theta(\frac{\partial v_x^n}{\partial x} + \frac{\partial v_y^n}{\partial y}) &\\
w^{n+1} + \frac{\partial v_x^{n+1}}{\partial y} - \frac{\partial v_y^{n+1}}{\partial x} &= 0
\end{aligned}
\tag{3.1.58}
$$

Así pues, el operador **A** está representado por las matrices

$$
\begin{aligned}
\mathbf{A}_0 &= \begin{bmatrix} \frac{1}{\Delta t} + \theta\frac{\partial v_x^n}{\partial x} & \theta\frac{\partial v_x^n}{\partial y} & 0 & 0 \\ \theta\frac{\partial v_y^n}{\partial x} & \frac{1}{\Delta t} + \theta\frac{\partial v_y^n}{\partial y} & 0 & 0 \\ 0 & 0 & \frac{1}{\beta\Delta t} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
\mathbf{A}_1 &= \begin{bmatrix} \theta v_x^n & 0 & \theta & 0 \\ 0 & \theta v_x^n & 0 & -\theta\frac{1}{Re} \\ 1-\theta & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \\
\mathbf{A}_2 &= \begin{bmatrix} \theta v_y^n & 0 & 0 & \theta\frac{1}{Re} \\ 0 & \theta v_y^n & \theta & 0 \\ 0 & 1-\theta & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{3.1.59}
$$

y

$$
\mathbf{f} = \begin{bmatrix} \frac{v_x^n}{\Delta t} - (1-\theta)(v_x^n\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^n}{\partial y} + \frac{\partial p^n}{\partial x} + \frac{1}{Re}\frac{\partial w^n}{\partial y}) + \theta(v_x^n\frac{\partial v_x^n}{\partial x} + v_y^n\frac{\partial v_x^n}{\partial y}) \\ \frac{v_y^n}{\Delta t} - (1-\theta)(v_x^n\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^n}{\partial y} + \frac{\partial p^n}{\partial y} - \frac{1}{Re}\frac{\partial w^n}{\partial x}) + \theta(v_x^n\frac{\partial v_y^n}{\partial x} + v_y^n\frac{\partial v_y^n}{\partial y}) \\ \frac{1}{\beta}\frac{p^n}{\Delta t} - \theta\left(\frac{\partial v_x^n}{\partial x} + \frac{\partial v_y^n}{\partial y}\right) \\ 0 \end{bmatrix}
\tag{3.1.60}
$$

3.2 Diseño

El análisis efectuado trae al conocimiento el proceso que ha de realizarse para resolver el problema planteado. Una vez que se ha estudiado este proceso se diseña un conjunto de algoritmos que puedan ser escritos para llevarlo a cabo.

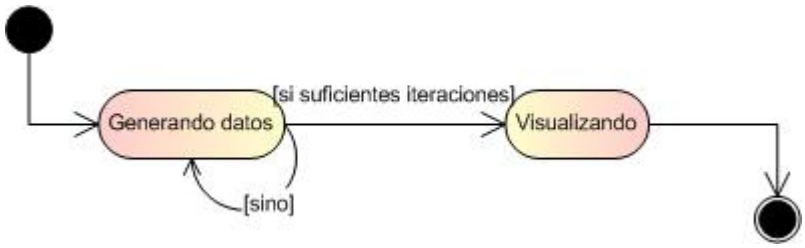


Figura 3.1: Diagrama de estado general del software

El cumplimiento de los objetivos formulados en este trabajo conlleva a la elaboración de un software consistente en dos fases (ilustradas en la figura 3.1) relacionadas pero diferentes en funcionamiento, estructura y composición:

- Por una parte se realiza el procesamiento matemático de las ecuaciones que conforman el problema a través de los métodos numéricos exigidos por éste y se generan los datos que conforman su solución.
- Por otra parte, una vez obtenidos los datos que integran la solución numérica del problema, se combinan visualmente de modo que se obtenga una simulación gráfica comprensible para cualquier usuario.

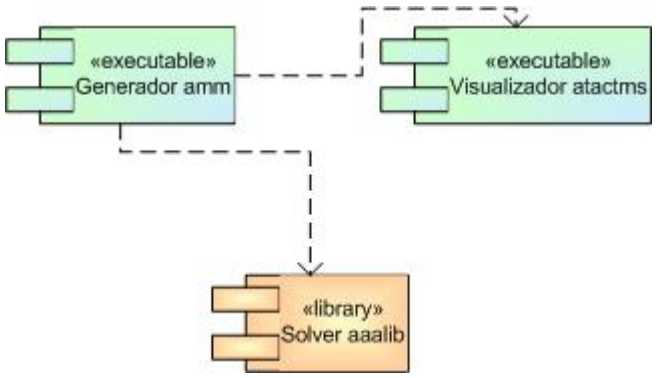


Figura 3.2: Diagrama de componentes general del software

El primer módulo, bautizado *amm*, utiliza una librería de aplicación general para la aproximación numérica de ecuaciones diferenciales y la emplea en resolver las ecuaciones de Navier-Stokes

(1.1.1) en dominios predeterminados iterando una cierta cantidad de veces y registrando las soluciones para su posterior visualización. La librería, llamada *aaalib*, toma estos datos de entrada y genera las soluciones parciales. Así se toman estos datos y se exportan para su postprocesamiento, además de reciclarlos en cada iteración hasta que se considere necesario.

El visualizador, que se le llamó *aTACTMS*, toma estos datos generados, y considerando el dominio del problema, grafica diferentes aspectos de la simulación, como la velocidad vectorial, la presión o una ambientación visual del comportamiento del fluido.

Cada uno de estos componentes, graficados en 3.2 (por Microsft Visio Professional 2002), se explica a continuación con más detalle:

3.2.1 Generador de datos

El generador de datos o *amm*, cuyo diagrama de actividad se grafica en 3.3 , se especializa únicamente en la simulación de flujo de fluidos en dominios poligonales en dos dimensiones.

Las definiciones de los datos de entrada son específicos para este funcionamiento. Por lo tanto los operadores diferenciales a resolver pueden ser programados tal como se obtienen en (3.1.18) , (3.1.19) , (3.1.21) , (3.1.22) . Sin embargo, además de los valores iniciales que éstos incluyen, pueden personalizarse parámetros como la constante θ, Δt, o el número de Reynolds, que hacen variar considerablemente los resultados de las ecuaciones y/o la velocidad en que convergen.

La definición del dominio es totalmente configurable. A través de una amigable y sencilla adición y eliminación de polígonos y líneas se puede construir o aproximar cualquier dominio que se desee, adaptando como se desee los valores nodales iniciales o fijos en zonas determinadas de éste como podría ser alguna de las fronteras. *amm* permite, además, la modificación deliberada de la discretización automática de nodos que genera, logrando así que el usuario obtenga un mayor control sobre el problema a resolver.

El programa utiliza la librería *aaalib* para resolver el problema recién definido. La solución la registra en un formato tal que pueda ser observado por editores de texto o aplicaciones como MS Excel y sobre todo por el visualizador gráfico. Las soluciones las utiliza también como soluciones iniciales en el siguiente paso de la simulación a través de un número de iteraciones configurable.

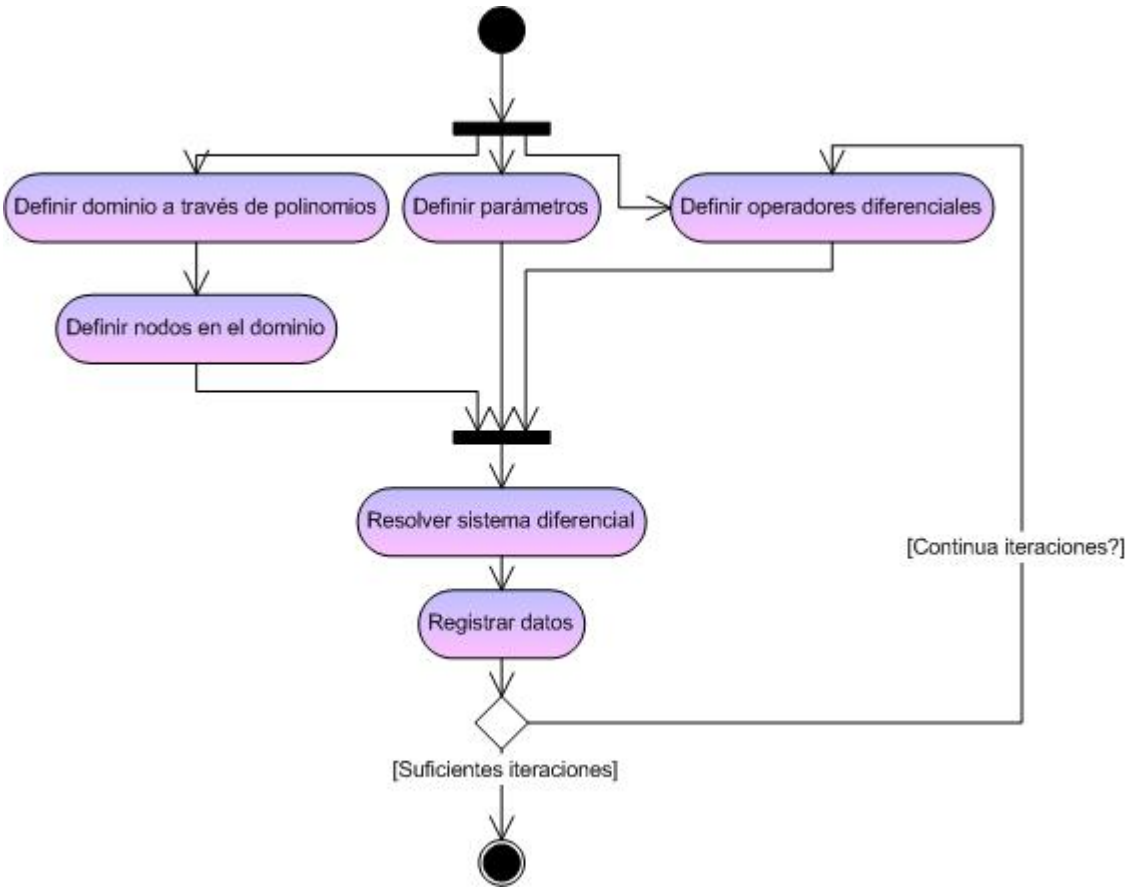


Figura 3.3: Diagrama de actividad del generador de datos

3.2.2 Solver

El Solver, o librería de resolución de ecuaciones diferenciales, es un módulo independiente que resuelve sistemas de ecuaciones diferenciales reales. Por el momento, y para cumplir con las exigencias del planteamiento, utiliza únicamente el método de elementos finitos mínimos cuadrados. Su diseño se muestra en la figura 3.4 y consta de las siguientes etapas:

- **Captura de datos**

  Hay varios datos que definen la entrada del sistema, comenzando por la dimensión del dominio, el dominio en sí, su frontera y las ecuaciones que condicionan la frontera, sin olvidar las matrices que actúan como operadores diferenciales y definan el sistema de ecuaciones a resolver.

- **Discretización del dominio**

  Ya definido el dominio se divide en cierta cantidad de elementos discretos N de forma personalizable.

- **Construcción de las funciones de forma**

  Con el fin de interpolar aproximaciones a la solución se crean funciones que cumplan con ciertas condiciones en cada nodo de cada elemento.

- **Construcción de las matrices elementales**

  Por cada elemento, y siguiendo el método LSFEM se crean las matrices K_e.

- **Ensamblaje de los elementos**

  Siempre según el método, se ensamblan los elementos para obtener un sistema de ecuaciones lineales resoluble matemáticamente.

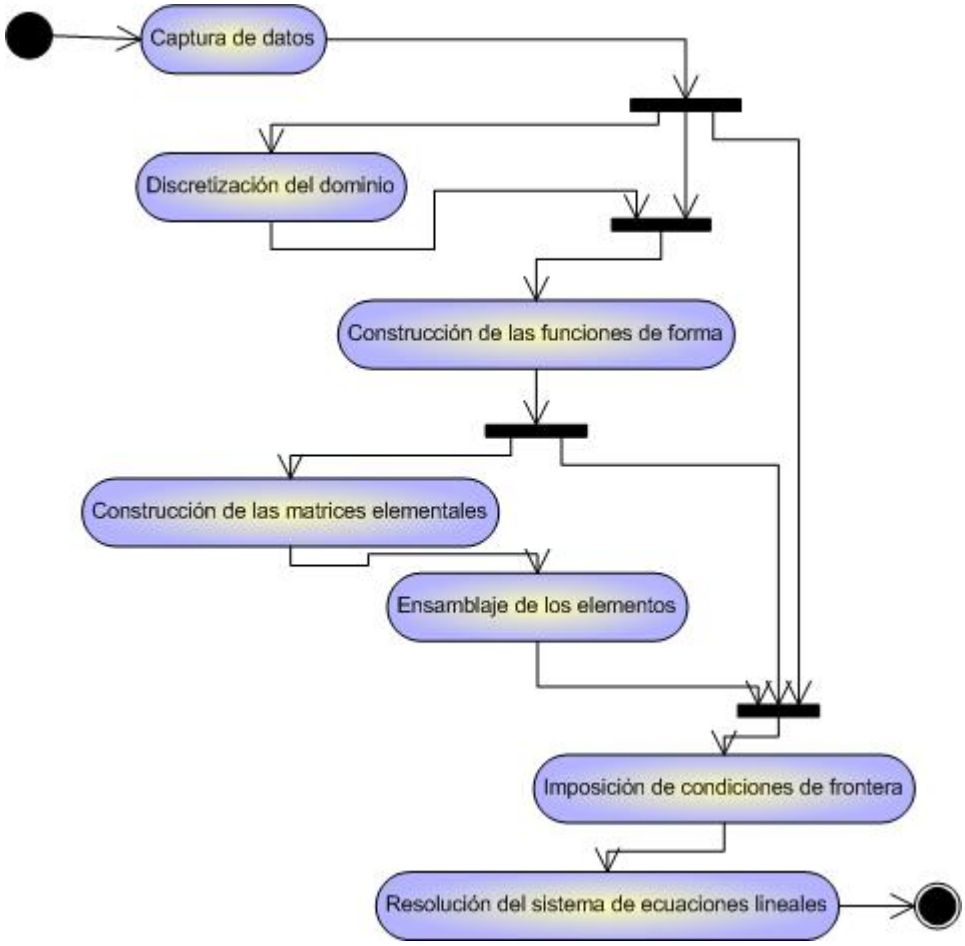


Figura 3.4: Diagrama de actividad del solver

- **Imposición de las condiciones de frontera**

  Implementando los avances del Prof. Cadenas [31] se modifican las ecuaciones lineales para que se cumplan las condiciones de frontera establecidas.

- **Resolución del sistema de ecuaciones lineales**

  Con alguno de los innumerables procedimientos existentes se diagonaliza la matriz resultante de los pasos anteriores.

3.2.3 Visualizador

El objetivo de la creación de un módulo de post-procesamiento para la visualización de los cálculos numéricos, es brindar una herramienta al usuario para comprender los resultados de una forma natural. Es por ello que se ha conseguido desarrollar con librerías estándares, una versión de dicho post-procesador para diferentes plataformas computacionales. La interfaz de entrada es nuevamente a través de archivos de datos, los cuales son generados automáticamente por las capas anteriores. Este tipo de entrada ofrece al usuario la posibilidad de procesar los cálculos de la simulación en computadores remotos de mayores prestaciones, obtener los resultados en archivos de texto ASCII y visualizar de manera local los resultados, independientemente del sistema operativo que posea.

El usuario puede elegir cualquiera de las tres modalidades de visualización, según su objeto de estudio. Si se quiere analizar las velocidades vectoriales del fluido en cada nodo del dominio se puede hacer a través de una interfaz sencilla. Para observar las variaciones de la presión sobre todo el dominio, se ofrece una gama de colores que indican los valores relativos de presión de una manera natural al ojo humano. También se puede visualizar el discurrir del fluido en una simulación de arrastre de partículas cromáticas.

3.3 Implementación

El paquete completo se implementa en lenguaje *C*, ya que cuenta con todas las ventajas necesarias en este trabajo, entre las cuales se encuentran la portabilidad, la amplia disponibilidad de reuso y la facilidad con que se desarrollan tanto algoritmos de procesamiento como poderosas interfaces gráficas.

3.3.1 La Librería para Resolución de Ecuaciones Diferenciales

La construcción de la librería *aaalib* es realizada manteniendo una amplia perspectiva que contempla desde los lenguajes y sistemas que soporta hasta la utilización que se le da. Al estar

totalmente escrita en lenguaje *ANSI C*, el único obstáculo para su portabilidad vendría a ser la librería de resolución lineal, que sin embargo puede ser reemplazada con extrema facilidad o configurarse a gusto.

Radiante de juventud, la librería muestra una simplicidad que no ha sido contaminada por códigos ajenos ya que está basada únicamente en fundamentos teóricos matemáticos. Aún así mantiene una interfaz intuitiva y muy manejable y transparente. Su potencialidad radica en su paradigma estructurado que facilita su remodelación y ampliación hasta niveles altísimos, según evolucione con respecto al tiempo o así lo vaya haciendo las diferentes bases matemáticas implementadas o que puedan implementarse. Por ejemplo, para el momento de la presentación de este trabajo la librería procesa dominios bidimensionales, los subdivide en elementos triangulares que interpola en polinomios lineales, realiza el procesamiento con el Método de Elementos Finitos Mínimos Cuadrados y opera la matriz lineal con el método del gradiente conjugado estabilizado. Sin embargo, la evolución puede darle poder sobre dominios en 3D, interpolaciones de mayor grado, o puede adherirle métodos diferentes como pueden ser los miméticos sobre las ecuaciones diferenciales o métodos mejorados tipo Krylov sobre las ecuaciones lineales.

La manera de usarse sigue varios pasos que corresponden con los realizados por los métodos de análisis numéricos que se utilizan, como lo son los de Elementos Finitos.

Primeramente se define un dominio, a través de la adición o substracción de objetos geométricos como polinomios, líneas o círculos. De esta forma se puede construir, o, al menos, aproximar cualquier dominio que se desee y con la granularidad que se especifique en cualquier zona del dominio.

Transparente a la creación automática del mallado y la interpolación de cada función elemental, el usuario indica los operadores diferenciales y sus condiciones de frontera pertenecientes al problema planteado. Por el momento no se soportan condiciones de frontera tipo Robin ni mixtas.

Una vez definido el problema la obtención de la solución es tan rápida como el procesamiento numérico de los datos. Algunas cuestiones como la linealización de las ecuaciones diferenciales o las iteraciones que involucre un problema corresponden al programa que utilice la librería, que sólo se encarga de procesar el operador en forma de matrices.

Estructura de datos

En las etapas de diseño del software y análisis de las bibliotecas a utilizar, fue considerada la creación de unas estructuras que proporcionarían una base sólida en el desarrollo de los modulos

propuestos en los objetivos, así como en creaciones futuras. La principal se llamó AAA y almacena toda la información referente al dominio, a los elementos y los operadores diferenciales involucrados, se define en 3.5 .

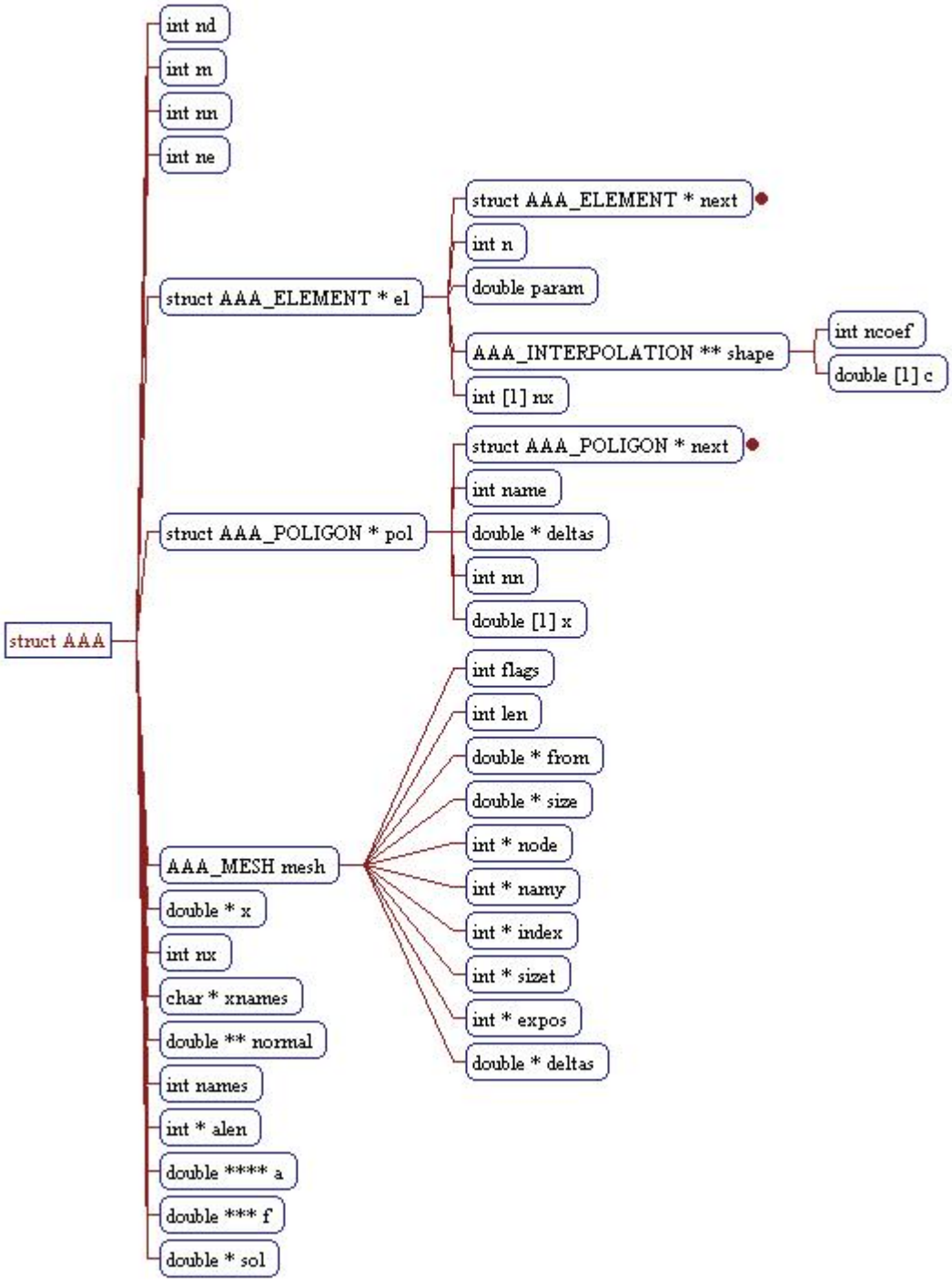


Figura 3.5: Estructura para ecuaciones diferenciales

Prototipos de las Funciones Principales

El Solver *aaalib* se desarrolla como una librería estructurada compuesta por una gama de funciones API que permiten realizar las funcionalidades diseñadas creando y manipulando los objetos necesarios para el procesamiento de los métodos.

Las funciones de la librería están clasificadas en varias categorías:

- Utilidad. Necesarias para ciertas actividades como la reserva de memoria o comparacíón de los datos.
- Dominio. El usuario, en este caso, el programa que utiliza la librería, indica con precisión el dominio del problema, ya sea añadiendo y eliminando polígonos, insertando elementos individualmente, o señalando la lista de nodo que representan el dominio discretizado.
- Mallado. Dado el dominio se especifican los detalles relativos a la estructura interna en la que se guarda el mallado.
- Datos. El usuario introduce los datos de entrada que corresponde a los operadores diferenciales para cada nodo en el dominio.
- Iterpolación. Se crean las funciones de interpolación de la forma y grado deseados por el usuario.
- Procesamiento. Las funciones de procesamiento realizan el trabajo necesario para la resolución de los sistemas de ecuaciones y producen la salida deseada que representa los valores nodales que aproximan la solución del problema.
- Salida.

Vale la pena hacer notar que aquí se enumeran sólo algunas funciones de todas las posibles utilizables.

Las principales funciones por cada categoría son:

Utilidad

- *aaa_new*. Crea la estructura principal que será usada durante todo el proceso como un ente que contiene los datos del problema y por lo tanto es un identificador de éste y debe ser incluido en todas las funciones posteriores.

- *aaa_del*. Libera de la memoria la estructura reservada y elimina el identificador creado.
- *aaa_malloc*. Reserva memoria con una característica especial que previene fallos de hardware o software.
- *aaa_free*. Libera la memoria del sistema que fue reservada con *aaa_malloc*.
- *aaa_new_operator*. Reserva la memoria para un operador diferencial aplicado a un nodo específico.
- *aaa_time*. Obtiene un número dependiente del tiempo en milisegundos. Utilizado para medir la eficacia de los algoritmos.

Dominio

- *aaa_domain_addpoligon*. Agrega el contenido de un polígono compuesto un conjunto infinito de puntos que pertenecerán al dominio.
- *aaa_domain_delpoligon*. Resta del dominio un conjunto infinito de puntos pertenecientes a un polígono especificado.
- *aaa_domain_addnodes*. Indica con exactitud qué nodos desean usarse para la solución del problema. Es opcional, ya que éstos pueden crearse automáticamente al agregar polígonos.

Mallado

- *aaa_mesh_generate*. Dado un dominio delimitado, crea un mallado regular para su discretización elemental.
- *aaa_mesh_coord*. Dado el índice de un nodo devuelve la coordenada en el eje indicado de éste nodo.
- *aaa_mesh_generate_normal*. Genera las normales a las fronteras existentes en cada nodo perteneciente a una de ellas.

Datos

- *aaa_set_data_a*. Por cada nodo se indican los operadores diferenciales que representan las ecuaciones diferenciales.
- *aaa_set_funcion_a*. El parámetro es una función que, dado un nodo devuelve los valores del operador. Útil cuando se tienen datos contínuos.
- *aaa_set_data_f*. Por cada nodo se señal el vector $\mathbf{f}$ incluido en las ecuaciones diferenciales.
- *aaa_set_data_s*. Necesario para generar una solución inicial, esta función fuerza a que ésta tenga un valor dado.
- *aaa_sol*. Una vez generada la solución del sistema es obtenida a través de esta llamada.
- *aaa_dsol*. Utiliza un método de diferencias finitas para calcular la derivada de la función solución en cada nodo especificado.

Interpolación

- aaa_interpolate_2d_triangle_linear. Construye las funciones de forma o de interpolación lineales por cada elemento triangular.
- aaa_interpolate_integral_2d_triangle_linear. Devuelve la integral de las funciones de forma y sus productos en cada nodo utilizando las predefinidas para los elementos triangulares lineales.

Procesamiento

- aaa_process_lsfem. Realiza el trabajo que corresponde en sí al método de elementos finitos mínimos cuadrados. Ensambla las matrices elementales, construye el sistema de ecuaciones lineales y lo resuelve utilizando la librería *UCSparseLib*, devolviendo los valores nodales de la función solución.

PseudoAlgoritmo del Procesamiento

Con el fin de ilustrar más de cerca el trabajo realizado se transcribe el código en alto nivel del algoritmo que implementa el método de elementos finitos mínimos cuadrados.

```
para cada elemento e
    para cada subdominio name
        si e en name
            para cada nodo i en e
                para cada nodo j en e
                    para k desde 0 hasta m
                        para l desde 0 hasta m
                            para cada matriz h1 en operador A[name]
                                para cada matriz h2 en operador A[name]
                                    erre = integral_interpolacion(name, e, i, j, h1, h2)
                                    suma = 0
                                    para cada fila t en A[name]
                                        suma = suma + A[name]^i_h1[k, t] × A[name]^j_h2[l, t]
                                    fpara
                                    suma = suma × erre
                                    K_ikjl = K_ikjl + suma
                                fpara // h2
                            fpara // h1
                        fpara // l
                    fpara // k
                fpara // j
                para k desde 0 hasta m
                    para cada matriz h1 en operador A[name]
                        erre = integral_interpolacion(name, e, i, h1)
                        suma = 0
                        para cada fila t en A[name]
                            suma = suma + A[name]^i_h1[k, t] × f[name]^i[t]
                        fpara
                        suma = suma × erre
                        q_ik = q_ik + suma
                    fpara // h1
                fpara // k
            fpara // i
        fsi
    fpara // name
fpara // e
```

Cuadro 3.1: Pseudo-código del solver

3.3.2 El Generador de Datos

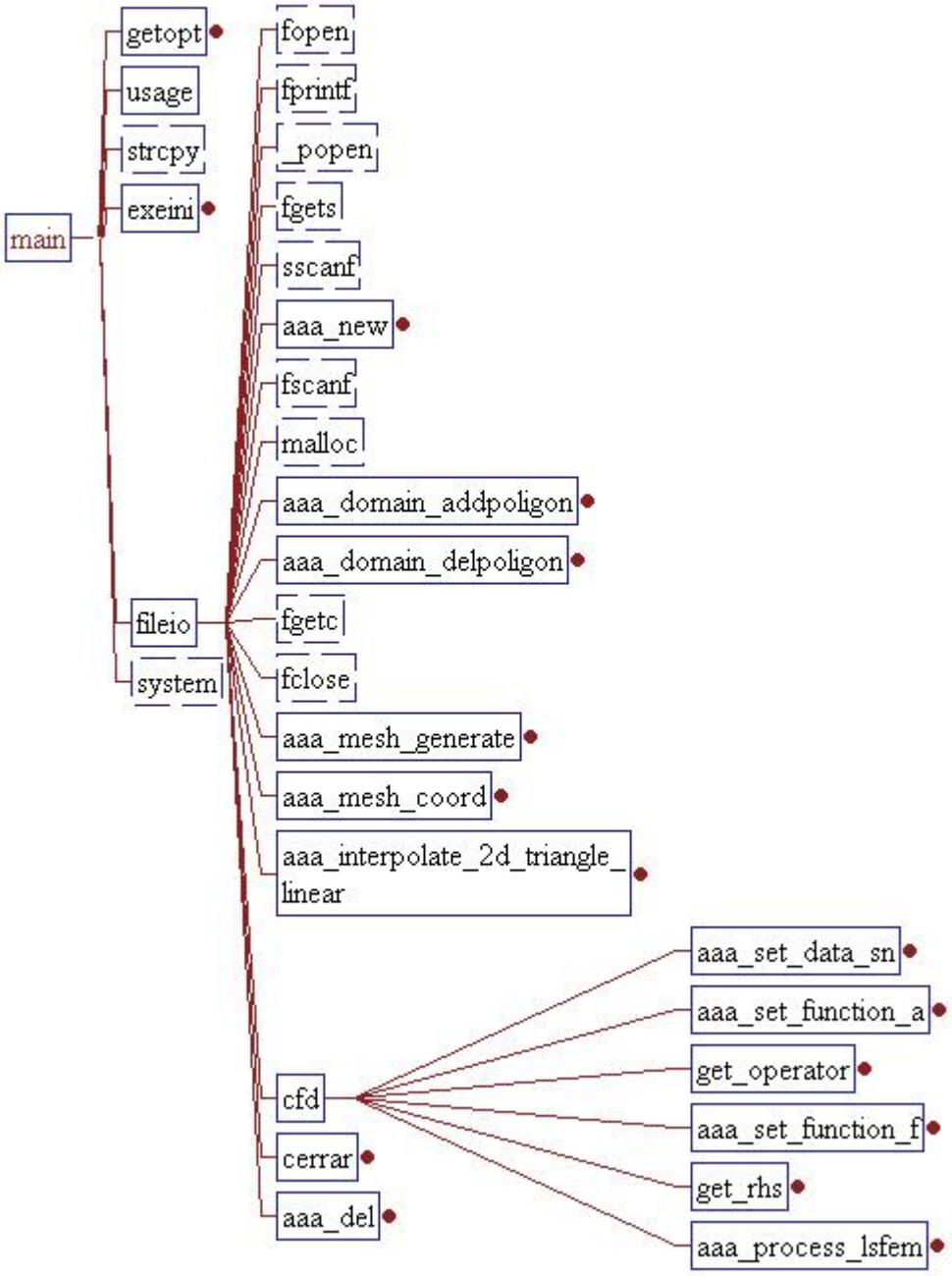


Figura 3.6: Estructura interna del generador de datos

Consiste en un módulo ejecutable que se auxilia con la librería *aaalib* para la resolución del problema planteado en 1.1. La codificación de las ecuaciones de *Navier-Stokes* estudiadas en 3.1.1 y 3.1.2 se realizan en este módulo.

Con total independencia de interfaz gráfica alguna, el módulo *amm* utiliza el mecanismo de *Input-*

Output procesando archivos de entrada para producir archivos de salida.

El archivo de entrada tiene un formato de texto sencillo y comentable, apto para la fácil comprensión humana. En él se define por sobre todo el dominio del problema, así como su frontera. También se pueden configurar algunos parámetros como el número de iteraciones y la discretización del tiempo, al igual que otros relativos a las ecuaciones de *Navier-Stokes* como el número de *Reynolds* o el parámetro de discretización temporal θ.

El formato de la salida está no sólo dirigido a una fácil lectura por el usuario humano o por cualquier programa o *script* sino específicamente para su procesamiento por el visualizador gráfico contenido en este paquete.

La estructura de funciones del programa, en 3.6 está graficado, al igual que otras figuras incluidas en este trabajo, por el paquete comercial de herramientas *Understand for C++* (http://www.scitools.com/).

3.3.3 El Visualizador

El módulo ejecutable encargado de la visualización, llamado *aTACTMS*, también hace énfasis en su portabilidad, aunque ésta es menor debido a su carácter gráfico. Éste módulo, a niveles de programación, no tiene relación alguna con los otros. Su trabajo es leer un archivo de datos y graficarlos, sin objetar sobre el origen de estos datos. Por esto puede ser utilizado como un visualizador general para cualquier generador que implemente ecuaciones que modelan fluidos en cualquier método numérico y sobre cualquier dominio bidimensional.

El programa está escrito por completo en *C* y basado en *OpenGL*. La librería de *OpenGL* se comenta en 3.3.5.

Para aumentar la portabilidad se escribieron varias subrutinas que actúan como interfaz de ventanas. Para los sistemas *Windows* se facilitan las llamadas *API* o funciones nativas de *Windows*. Esto hace que en éste ambiente no sea necesaria una librería adicional para el sistema de ventanas gráficas. Si el sistema es *linux* o similar probablemente se tenga la librería *GLX* que es una extensión de *OpenGL* al ambiente *X*. Las máquinas que contengan *SDL*, que además es portable, también podrán abrir la ventana principal. En último caso se necesitará *GLUT*, existente igualmente para cualquier plataforma. Ésta personalizable estructura se observa en la figura 3.7 .

aTACTMS lee de un archivo predefinido o de la entrada estándar un conjunto de puntos considerados como la composición del dominio. Con fines de acelerar el proceso, también se

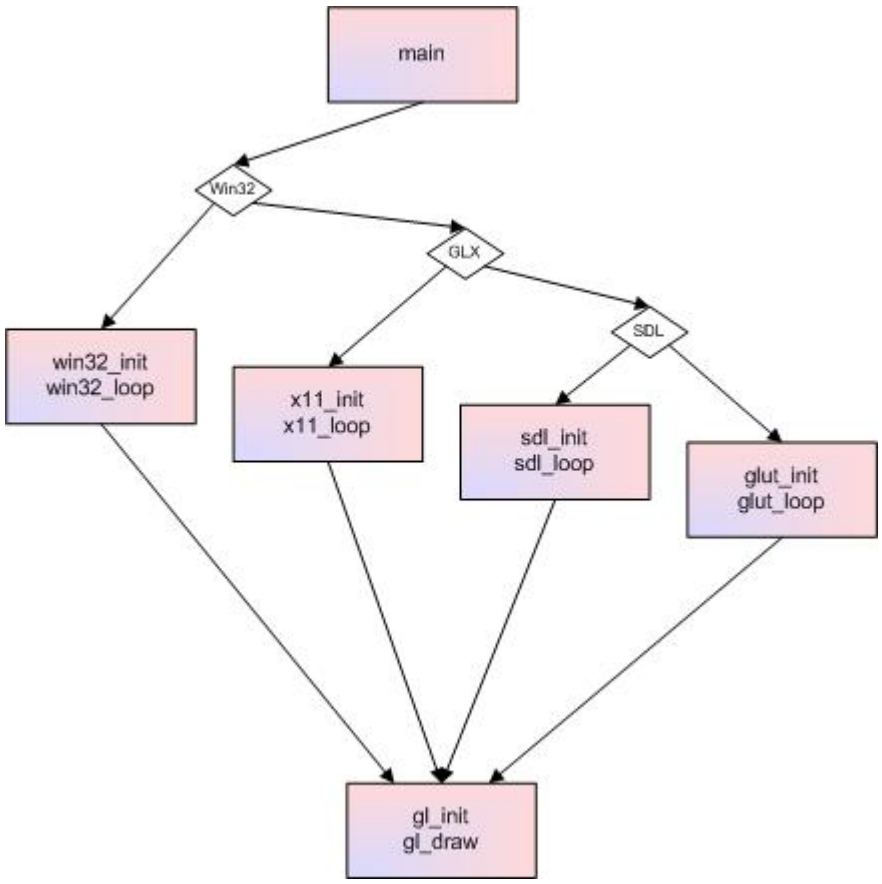


Figura 3.7: Opciones de sistemas de ventanas del visualizador

leen las celdas que componen el mallado que se formó al discretizar este dominio, es decir, los elementos finitos. Al implementar generadores de mallado a partir de puntos, esto no será necesario, ya que sólo se utiliza con fines gráficos. Finalmente, en texto raso o comprimido en formato *.zip*, se obtiene una serie de iteraciones que corresponden a cada paso de tiempo que transcurre mientras se mueve el fluido. En cada iteración se leen los valores nodales que indican la presión y el vector velocidad del fluido en cada nodo del dominio.

Internamente, *aTACTMS* posee un temporizador que regula la velocidad de actualización según un parámetro de configuración Δt. Según el estado en que se encuentre el visualizador operado por el usuario, cada submódulo en el programa decide si redibujar la escena y qué se va a dibujar (ver 3.8).

El visualizador puede verse como un autómata en el que los estados que lo componen son operados por el usuario con un sólo *click* del *mouse* o el pulso del teclado. El visualizador tiene controles sencillos e intuitivos que permiten al usuario suspender el movimiento, detenerlo por completo o reiniciarlo, además de regular la velocidad de actualización de las imágenes, o el parámetro Δt. Existen tres modalidades para la observación del fluido, que pueden activarse

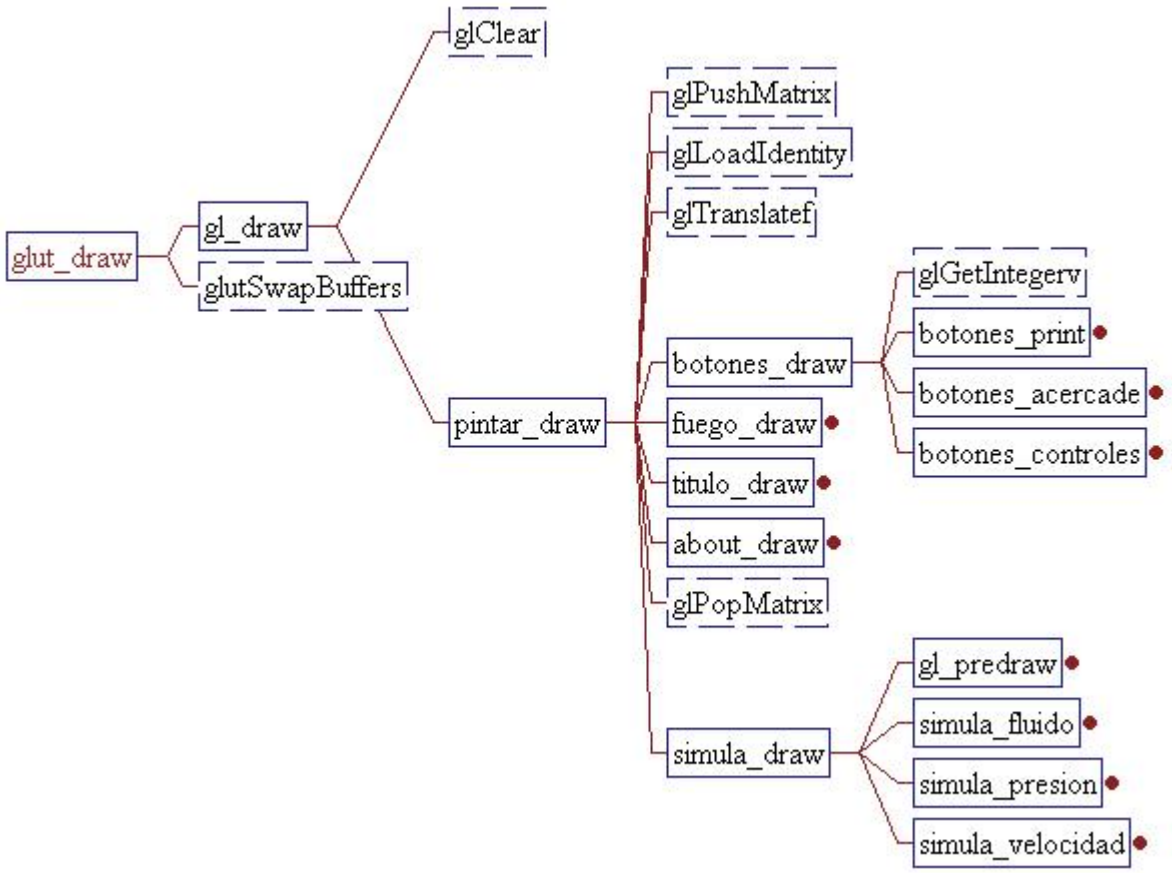


Figura 3.8: Acciones al dibujar una escena en el visualizador

y desactivarse con facilidad: *fluido*, que estudia el arrastre de partículas en el medio simulado; *presión*, que grafica una gama de colores representando los valores de la presión desde un mínimo a un máximo y que puede variar en cada nodo; y *velocidad*, que indica con pequeñas flechas también graduadas cromáticamente la velocidad vectorial que alcanza el fluido en cada nodo. Con cierta granularidad se puede apreciar la belleza del movimiento del fluido en cualquier modalidad.

3.3.4 Uso de la librería UCSparseLib

UCSparseLib es una biblioteca portable para la resolución serial y paralela de sistemas lineales densos y esparcidos. Los sistemas lineales considerados son de la forma

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{3.3.1}$$

, donde $\mathbf{A}\mathbf{x}$ es una matrix $n \times n$ densa o esparcida de gran tamaño suministrada por el usuario, y puede ser regular o irregular en su estructura. $\mathbf{b}$ es vector del lado derecho. Las investigaciones sobre técnicas de matrices esparcidas han llegado a incrementar su complejidad, y promete

continuar acentuándose debido a la necesidad creciente de diseñar algoritmos eficientes para modernas supercomputadoras. Aún cuando existe gran número de paquetes y herramientas para resolver computacionalemente matrices densas de pequeño tamaño, hay también la necesidad de herramientas similares o librerías de propósito general para el trabajo con matrices esparcidas. Una colección de unos cuantos programas básicos para realizar tareas comunes y elementales, puede ser muy útil y reducir el tiempo dedicado a implementar y probar algoritmos para matrices esparcidas. Los paquetes *Linpack* y *Eispack* desarrollados en los años 70, han sido de gran ayuda en varias áreas de la computación científica, ahorrando grandes cantidades de esfuerzo y tiempo. Por otro lado, es muy frecuente que los investigadores de la computación de matrices esparcidas, codifiquen sus propias subrutinas para los modos de almacenamiento de la matriz o de su reordenamiento acorde con ciertas permutaciones; una de las razones pudiera ser la ausencia de estándares. Por ello, para la misma estructura de datos básica, puede estar en uso un gran número de variaciones.

UCSparseLib es una biblioteca de propósito general debido a que puede manipular matrices generales no simétricas y con patrones de estructura irregulares. Los patrones de las posiciones diferentes de cero no necesariamente deben ser simétricos. Desarrollada en ANSI C, *UCSparseLib* resuelve sistemas lineales esparcidos y densos. Su estructura general se puede ver en la imágen 3.9 . Entre las funcionalidades incluidas, y que han sido de utilidad en el desarrollo de este trabajo, están:

- Rutinas para leer y escribir matrices utilizando un formato simple. También es posible transformar una matriz a un formato postscript.
- Operaciones básicas aplicadas a vectores, multiplicación de matrices, solvers triangulares y reordenamiento de matrices.
- Solvers Iterativos como Gauss-Seidel, Jacobi, Gradiente Conjugado, GM-RES(m), BiCGstab y BiCG.
- Rutinas de utilidades adicionales como timers, tiempos de CPU y manejo de memoria.

Estructura de datos para matrices esparcidas

Una de las mayores dificultades encontradas en el cálculo de matrices esparcidas, es la variedad de tipos de matrices resultantes en aplicaciones prácticas. El próposito de un esquema de almacenamiento debe ser entónces, ganar eficiencia en términos de utilización de memoria y en operaciones aritméticas. Como resultado, se han propuesto muchas maneras diferentes de

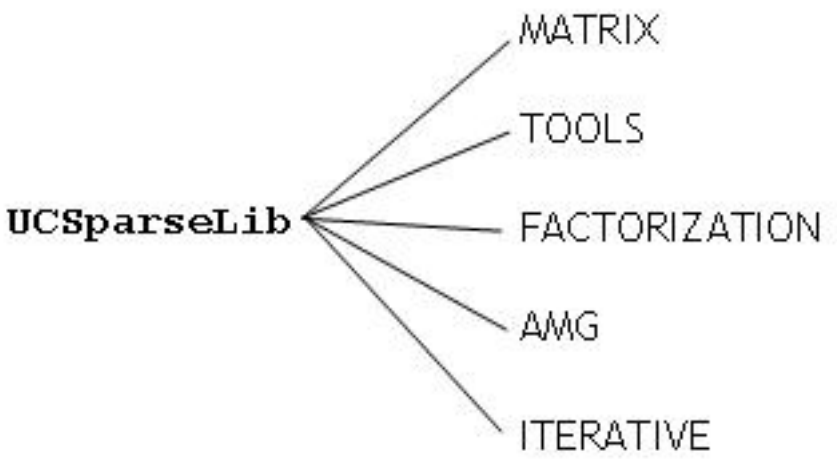


Figura 3.9: Organizacion General de UCSparseLib

almacenar matrices esparcidas tomando ventajas de la estrucutura de las matrices o del problema específico del cual surgen. Uno de los esquemas más comunes de almacenamiento es el Compressed Sparse Row (CSR). En este esquema, todas las entradas no cero son almacenadas fila por fila en un arreglo de reales unidimensional, junto a un arreglo entero que contenga los índices de sus columnas y arreglos de apuntadores a cada una de tales filas. El orden de los elementos dentro de cada fila no es importante, lo cual simplifica el esquema de almacenamiento. En UCSparseLib una matriz es un objeto llamado TDMatrix, el cual es creado utilizando la siguiente rutina:

```
TDMatrixCreate( TDMatris *M, TMatFormat format, int rows, int cols
);
```

donde M es una matriz, y debe ser declarada como $TDMatrix\ M$. El formato format es el modo de almacenamiento, el cual puede ser *MATRIX_DENSE_R, MATRIX_DENSE_C, MATRIX_SDENSE_R, MATRIX_SDENSE_C, MATRIX_DIAG, MATRIX_CSR, MATRIX_CSC, MATRIX_SCS_R, MATRIX_SCS_C*. $rows$ y $cols$ son las dimensiones de la matriz.

El objeto `TDMatrix` es definido en 3.10 .

```
typedef struct DMatrix_t *TDMatrix;
```

Un componente importante del objeto $TDMatrix$ es la estructura $TDSparseVec$.

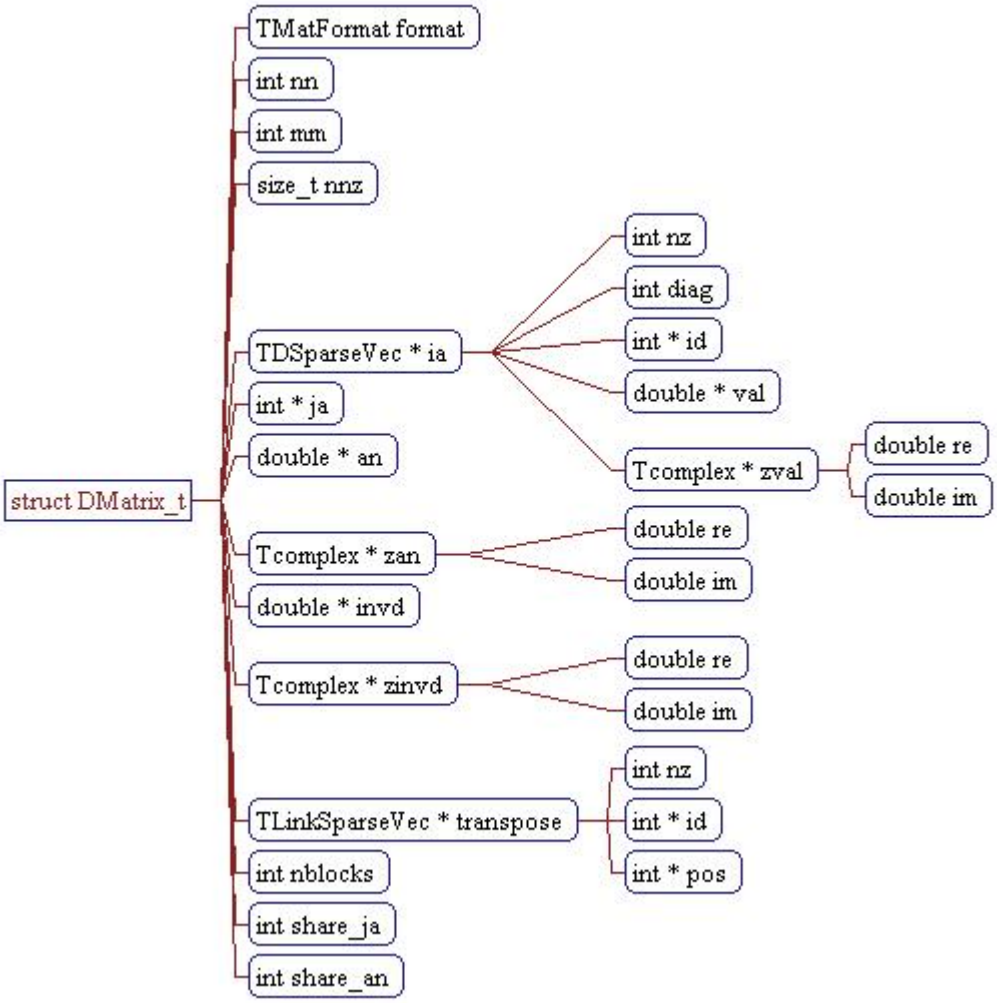


Figura 3.10: Estructura de la matriz esparcida

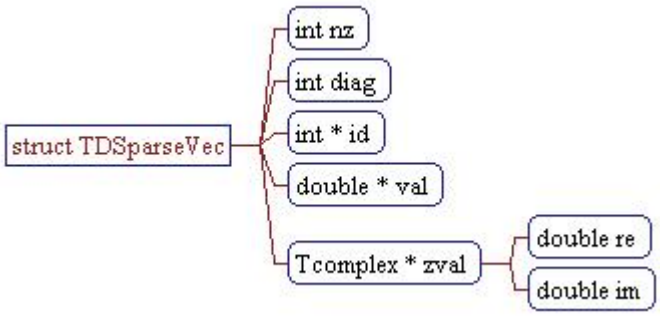


Figura 3.11: Estructura del vector esparcido

Esta estructura representa un vector esparcido y es 3.11

Para accesar la fila i de la matriz $TDMatrix\ M$, se utiliza la siguiente macro:

```
For_TDMatrix_Row( M, i, row, mode ){
.
.
.
}
```

donde, row es un $TDSparseVec$ y mode puede ser: $ACCESS\_READ$, $ACCESS\_WRITE$ Y $ACCESS\_RW$. Existen tambien macros para accesar columnas o fila/columnas si el usuario no conoce el formato de la matriz. Las macros son: $For\_TDMatrix\_Col$ y $For\_TDMatrix\_RC$, respectivamente.

Se puede conocer más sobre ésta librería en la página de su autor http://portal.facyt.uc.edu.ve/˜ glarra/

3.3.5 OpenGL como Herramienta de Graficación

OpenGL es una interfaz de software para hardware gráfico. Esta interfaz consiste en aproximadamente 150 comandos diferentes para especificar los objetos y operaciones necesarias para producir aplicaciones gráficas interactivas.

OpenGL está diseñado como una línea de procesos, y es una interfaz independiente de hardware que puede ser implementada en muchas plataformas diferentes.

Con el fin de satisfacer estas cualidades, no posee comandos para la creación de ventanas u obtener entradas del usuario, por lo que se debe hacer uso de algún controlador del sistema de ventanas propio del hardware en particular. De igual manera, *OpenGL* no provee comandos de alto nivel para la creación de objetos complejos; en su lugar, sólo define un pequeño conjunto de primitivas geométricas tales como puntos, líneas y polígonos con los cuales es posible generar cualquier modelo deseado. Es por ello que se hace necesaria la construcción de todos los elementos que conforman los gráficos deseados para este trabajo, como una capa de abstracción de software que tiene a *OpenGL* y otras librerías auxiliares como base.

Debido a que es posible realizar muchas cosas con el sistema grafico de OpenGL, un programa puede llegar a ser complicado. Sin embargo, la estructura básica puede ser realmente sencilla: inicializar ciertas variables de estado que controlan la forma en la cual *OpenGL* dibuja y los objetos a ser dibujados. Entre la terminología de la computación grafica, la palabra Render implica el proceso en el cual un computador crea imágenes a partir de modelos. Estos modelos u objetos son construidos con primitivas geométricas (puntos, líneas y polígonos) que son especificadas por sus vértices.

El render final de la imagen consiste en píxeles dibujados en la pantalla; un píxel es el elemento visible mas pequeño que el hardware grafico puede colocar en la pantalla. La información sobre los píxeles (como por ejemplo, el color que deben tener), es organizada en la memoria como mapas de bits. Un mapa de bit es un área de memoria que almacena un bit de información para cada píxel en la pantalla; un bit podría indicar cuan rojo se supone que debe ser un píxel en particular, por ejemplo.

La especificación del *API* de *OpenGL* se puede descargar de la dirección en Internet http://www.opengl.org/documentation/specs/version2.0/glspec20.pdf.

Como ya se ha mencionado, *OpenGL* provee un conjunto de comandos primitivos pero poderoso, y todos tipo de dibujado de mas alto nivel debe ser hecho en términos de tales comandos. Además, todos los programas hechos en *OpenGL* deben hacer uso de un mecanismo base de algún sistema específico de manejo de ventanas. Existe un número de bibliotecas que permiten simplificar algunas de tales tareas de programación, como:

- *OpenGL Utility Library (GLU)*, la cual contiene varias rutinas que hacen uso de los comandos de bajo nivel de *OpenGL* para llevar a cabo tareas como configuracion de las matrices que especifican la orientacion y proyeccion de las vistas, y dibujado de superficies. Esta biblioteca es distribuida como parte de todas las implementaciones de *OpenGL*. (http://www.opengl.org/documentation/specs/glu/glu1_3.pdf).

- *OpenGL Utility Toolkit (GLUT)* es un conjunto de herramientas de ventanas independiente del sistema, que simplifica con pocas rutinas la necesidad de crear el entorno de dibujado requerido por *OpenGL*. La mayoría de las variables de estado se encuentran inicializadas por defecto en valores que facilitan el desarrollo en programas simples. Las rutinas igualmente requieren pocos parámetros. Por estas razones de simplicidad, *GLUT* no hace uso de manejadores, apuntadores o estructuras de datos nativas a algun sistema en particular; incluso provee de su propio conjunto (limitado) de fuentes.(http://www.opengl.org/documentation/specs/glut/spec3/spec3.html)

- *OpenGL X (GLX)* es la extensión de *OpenGL* para *X11* que es el ambiente de ventanas que viene de forma predeterminada en casi todos los sistemas *unix*. La interfaz de programación con este ambiente es un conjunto de primitivas básicas que conforman la base para cualquier interfaz gráfica sobre estos sistemas, incluyendo el mismo *OpenGL*. Son funciones únicamente para el manejo de ventanas y de dibujo puntual, y no sirven lo que se denomina *Widgets* por lo que para la creación de herramientas como botones o cuadros de texto se necesitan otras librerías como *Lesstif*. Sin embargo, en éste trabajo se incluirá únicamente la librería nativa *X11* además de *GLX* (http://www.opengl.org/documentation/specs/glx/glx1.3.pdf).

OpenGL puede accederse libremente a través de su página http://www.opengl.org.

3.3.6 Otras librerías utilizadas

- *Windows 32 bits Aplication Programming Interface (Win32API)* es el estilo nativo de programación bajo windows. Es más complicado que *Glut* pero también más rápido y funcional. Pueden crearse ventanas de todo tipo y tener un mayor manejo de ellas, además de una gran variedad de utilidades que involucran la estructura interna de *Windows*. Su casi nula portabilidad lo hace un mal candidato para distribuciones, aunque es muy utilizado gracias a la aceptación mundial de este sistema operativo (http://msdn.microsoft.com).

- *Simple DirectMedia Layer (SDL)* es una librería independiente de la plataforma diseñada para facilitar la escritura de software multimedia, como juegos y emuladores. Provee un acceso a bajo nivel al audio, teclado, ratón, *joystick*, a marcos de memoria en 2D y a hardware en 3D vía *OpenGL*. Tiene licencia *GNU LGPL* y soporte para cualquier sistema y lenguaje (http://www.libsdl.org).

- *zlib* es una librería de propósito general para compresión de datos. Es compatible con los formatos *.zip* clásicos y con los archivos *gzip*. Es una librería portable a cualquier sistema y arquitectura además de tenera una licencia libre, por lo que es utilizada en casi todos los paquetes y distribuciones alrededor del mundo. Posee una API fácil de utilizar y muy manejable, además de las utilidades *minizip* y *miniunz* para un acceso inmediato a los archivos *.zip* (http://www.gzip.org/zlib).

- *iiview* es un programa abierto para *X* que despliega imágenes en un directorio en miniatura o en tamaño normal. Soporta varios formatos gráficos. Algunos extractos de su código fueron utilizados y modificados para cargar archivos *.bmp* y *.jpeg* sin la necesidad de una librería adicional como *glaux* o *sdl_image* (http://sourceforge.net/projects/iiview).

- *voronoi* es un programa en *C* creado por Steve J. Fortune para demostrar su creación de un algoritmo de barrido que resuelve el problema de los diagramas de *Voronoï* como se introduce en la sección 2.4.1. Su código fue adaptado y formulado como librería a través de una interfaz simple. Con él se logró crear un generador de mallado más flexible y con características tales como el refinamiento y la adaptación de mallados irregulares. Este código se puede descargar de http://www.csit.fsu.edu/~ burkardt/cpp_src/sweep2/sweep2.html.

3.4 Pruebas

La realización de las pruebas y la depuración de errores conlleva a veces incluso más tiempo y esfuerzo que la construcción de las herramientas en sí. La ”fase” de pruebas ayudó a encontrar errores en sectores arbitrarios del proceso, desde la formalización de las ecuaciones planteadas hasta los detalles en programación que siempre aparecen.

Las fórmulas teóricas desarrolladas por el autor del trabajo fueron perfeccionadas, mencionando entre ellas las integrales (2.4.16) y (2.4.31) . Los módulos *aalib* (3.3.1), *amm* (3.3.2) fueron depurados exhaustivamente. El programa de visualización *aTACTMS* fue perfeccionado para la observación de los resultados arrojados por éstos módulos. La librería *UCSparseLib* (3.3.4) presentó fallos utilizando el método *GMRES* (2.5.2) pero funcionó muy bien a través del método BiCGStab (2.5.6).

La mayor dificultad para la correcta consecución de los resultados fue la búsqueda de un dominio bien planteado en el que la simulación se mantenga estable. Para ello se basaron pruebas en el trabajo de *Jiang*[17].

Las pruebas iniciales presenciaron una inestabilidad producida primero por errores en la implementación y luego por una implicación de parámetros sin un previo estudio de estabilidad que correspondería a un trabajo de doctorado. Sin embargo se mejoró en sobremanera la estabilidad de los datos con algunas modificaciones.

El carácter iterativo del desarrollo permitió no sólo unas mejoras materializadas sino el proyecto de perfeccionar y acrecentar el software a proporciones gigantescas.

CAPÍTULO 4

RESULTADOS Y CONCLUSIONES

Las pruebas para las simulaciones fueron evaluadas en base al realismo que debian presentar los resultados de acuerdo a las condiciones proporcionadas. Todos los tests de tiempo se realizaron en una *Pentium Intel IV* de 1.8GHz y 256 MB de memoria, con un *performance* de *FPU/iSSE2* 418/2343 MFLOPS, bajo ambiente *Microsoft (R) Windows 2000 (5.2195)* con 378MB de swapping y *Linux Mandrake 9.1* con 256MB de memoria swapping.

4.1 Benchmark

Con el fin de probar la velocidad del método, aunque con la imposibilidad de compararlo con otros en condiciones semejantes, se ejecutó la simulación procediendo a una medición de tiempo de procesamiento. Los datos variantes fueron el número de nodos y el sistema operativo.

Número de nodos	LSFEM seg.		UCLib seg.		Total seg.		Memoria KB.
	win2k	linux	win2k	linux	win2k	linux	
12	0	0.001	0.01	0.06	0.01	0.061	844
121	0.03	0.0855	0.1616	0.2519	0.1916	0.3374	6.048
1250	0.34	0.5497	15.499	17.6487	0.839	18.1984	16.036
12544	3.495	5.164	166.74	192.235	170.235	197.399	100.080

Cuadro 4.1: Benchmark de tiempo de procesamiento

Estos valores de tiempo computacional, sin embargo, dependen no sólo del tipo de computador, sino de las condiciones, complejidad y magnitud de la situación deseada. Debido a que los resultados son procesados independientemente a la visualización, no existe una relacióon entre el tiempo de cómputo y la fluidez en que son representados los cuadros de animación.

4.2 Resultados Numéricos

Es importante mencionar que las pruebas representativas mostradas en este trabajo, constituyen apenas unos pocos casos planteados de entre una inmensa cantidad y variedad que pueden ser consultados en la bibliografía o en reportes de experimentos en laboratorios. La intensión de ello es verificar el comportamiento de las ecuaciones en función de los métodos numéricos utilizados, y no hacer una demostración de las verdaderas capacidades de las herramientas que han sido creadas; corresponde a los investigadores en las áreas particulares de la ciencia o de ingeniería, establecer condiciones y escenarios de utilidad práctica.

4.2.1 Prueba 1

Se probó el programa simulando una tubería rectangular sin obstáculos a través de la cual pasa un fluido incompresible entrando por un extremo y saliendo por el otro.

El dominio bidimensional es representado como un simple rectángulo con ($\Omega : -6 \leq x \leq 20, -6 \leq y \leq 6$) discretizado en un mallado regular con $\Delta x = 0.5$ y $\Delta y = 0.5$ como en la figura 4.1 . Este mallado produjo 1325 nodos y 2496 elementos triangulares.

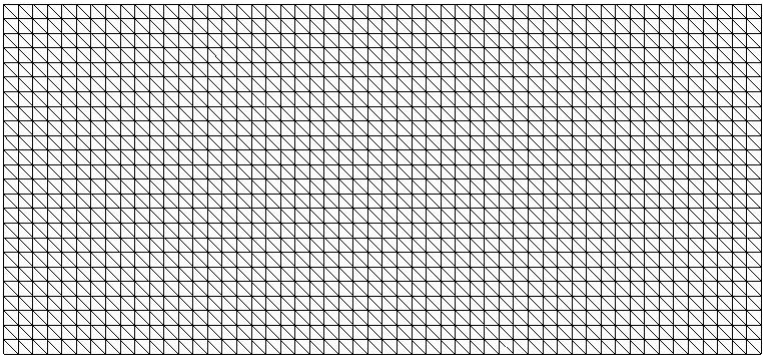


Figura 4.1: Dominio discretizado utilizado

Los valores iniciales establecidos fueron $v_x = 1$, $v_y = 0$, $p = 0$ y $w = 0$. Se establecieron las condiciones de frontera $v_x = 1, v_y = 0, p = 0$ sobre las paredes superior e inferior y la

frontera de influjo (izquierda). Sobre la frontera derecha se estableció únicamente $p = 0$. Se utilizó $\Delta t = 10^{-3}, \theta = 1$ (Discretización temporal de Euler, ver 2.1.5, 2.1.5) y un número de Reynolds $Re = 200$.

Para estos parámetros se logró gran estabilidad en el fluido, que permaneció prácticamente invariable con velocidades muy cercanas a las iniciales (ver imágen 4.2). La presión sin embargo, formó zonas de pequeñas variaciones en el centro vertical de la tubería (figura 4.3).

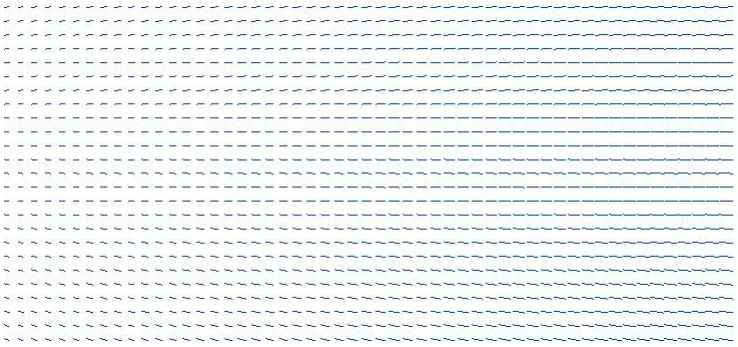


Figura 4.2: Velocidad vectorial en prueba 1

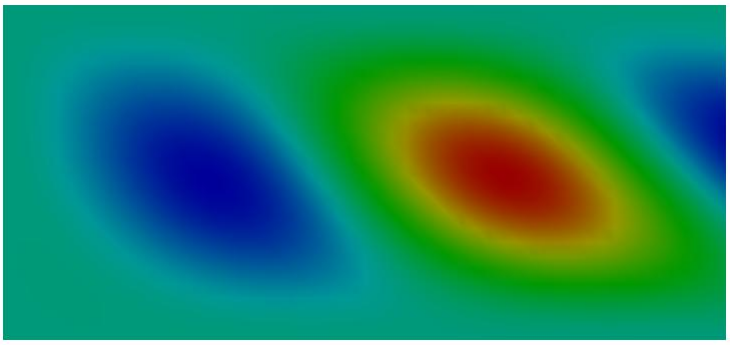


Figura 4.3: Presión presentada en prueba 1

4.2.2 Prueba 2

En esta prueba el fluido simulado llevaba una velocidad variable a través de la vertical en forma senoidal. La velocidad por el centro de la tubería era $v_x = 1$ que disminuía suavemente mientras más cerca de las paredes, en las que la velocidad era nula.

Se utilizó el mismo dominio que en la prueba anterior, discretizado de la misma manera.

La condición de frontera en la pared izquierda mantenía una velocidad similar a la inicial en el dominio Ω, mientras que en la pared superior e inferior se estableció $v_x = 0, v_y = 0, p = 0$. Al igual

que en la prueba anterior se utilizó $\Delta t = 10^{-3}, \theta = 1$ y $Re = 200$. Sin embargo, se estableció un parámetro de incompresibilidad $\beta = 10^3$.

Los resultados fueron parecidos a los ya mostrados, pues permanecieron muy cercanos a los iniciales (ver figura 4.4 y 4.5).

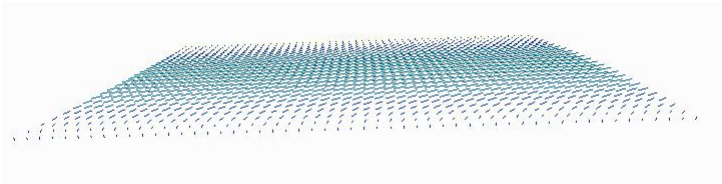


Figura 4.4: Perspectiva de la velocidad vectorial en prueba 2

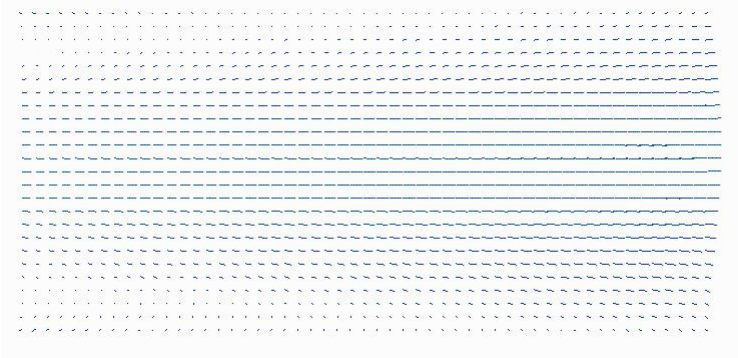


Figura 4.5: Velocidad vectorial mantenida en prueba 2

4.3 Conclusiones y Recomendaciones

- Los métodos implementados convergen para parámetros bien establecidos.
- Los resultados obtenidos en las simulaciones, en función de las condiciones borde de prueba, presentan una respuesta semejante a la que podría ocurrir en la realidad.
- En todos los casos la tolerancia del error en el cálculo de la solución de los sistemas lineales por el método BicGSTAB fue de 10^{-3}, exactitud generalmente alcanzada en las primeras iteraciones y por lo que no se considera que sea necesario cambiar el método a menos que se necesite mayor precisión.
- Se realizaron contribuciones a las formulaciones teóricas de los métodos así como una libería de uso general para resolución de sistemas de ecuaciones diferenciales de primer grado.

- El software desarrollado tiene como propósito ser una herramienta para los investigadores en diferentes ramas de la ciencia, donde aplique y sea importante la consideración de los aspectos de la dinámica de fluidos. A pesar de la alta complejidad de las ecuaciones y de las discretizaciones obtenidas en el desarrollo de esta investigación, se superaron las espectativas en los resultados numéricos y por ende visuales.

4.4 Trabajos Futuros

- Incrementar las capacidades del mallado.
- Incrementar los grados y las formas de las funciones de interpolación.
- Generalizar la librería de resolución de ecuaciones diferenciales para cualquier dimensión y grado.
- Implementer mayor cantidad de métodos a la librería.
- Perfeccionar el visualizador de fluidos.
- Estudiar el comportamiento de los fluidos en diferentes dominios y situaciones.
- Paralelizar las rutinas de simulación o de diagonalización de matrices esparcidas para la resolución de ecuaciones lineales.
- Probar más profundamente la librería, en ecuaciones como las de termodinámica o de electrostática.
- Realizar un estudio de estabilidad para los métodos utilizados.

APÉNDICE A

Codigo Fuente de la Implementación

Bibliografía

[1] R. Courant (1943). *Variational methods for the solution of problems of equilibrium and vibrations*, Bulletin of the American Mathematical Society.

[2] A. J. Chorin (1967). *A numerical method for solving incompressible viscous flow problems*. J. Comput. Phys.

[3] J. H. Bramble y A.H Schatz (1970). *Rayleigh-Ritz-Galerkin methods for Dirichlet's poblem using subspaces without boundary conditions*, Comm. Pure Appl. Math.

[4] J. H. Bramble y A. H. Schatz (1971). *Least squares for 2nth order elliptic boundary-value problems*, Math. Comp.

[5] J. K. Vernard, R. L. Street (1973). *Elementos de Dinámica de Fluidos*.

[6] Zienkiewicz, O. C. (1975).*Finite Elements in Fluids*, Vol. 1 ed by R.H. Gallagher et al.

[7] J. E. Akin (1982). *Application and Implementation of Finite Element Methods*. Academic Press, London.

[8] Bo-Nan Jiang, Lous Povinelly (1990). *Least-squares finite leement method for fluid dynamics*. Elsevier Science Publishers.

[9] Chin Lung Chang, Bo-Nan Jiang (1990). *An error analysis of least-squares finite element method of velocity-pressure-vorticity formulation for stokes problem*. Elsevier Science Publishers.

[10] Oden J.T. (1991) *Handbook of Numerical Analysis*, vol II, ed by P.G. Ciarlet, Amsterdam

[11] Bo-Nan Jiang, Louis Povinelli (1993). *Optimal least-squares finite element method for elliptic problems*. Elsevier Science Publishers.

[12] John Muller (1993). *La Mecánica de Fluidos*. Ed CECSA.

[13] Reddy, N.A (1993). *An Introduction to the Finite Element Method*. 2ºed. McGraw Hill NY.

[14] David Kincaid, Ward Cheney. (1994). *Análisis Numérico*. Delaware.

[15] H. K. Versteeg, W. Malalasekera (1995). *An Introduction to Computational Fluid Dynamics the Finite Volume Methods*. Prentice Hall.

[16] Gareth Williams (1996). *Fundamentos Básicos de Mecánica de Fluidos*, ed Mc Graw Hill. 3 ed.

[17] B. N. Jiang (1998). *The Least-Squares Finite Element Method: Theory and Applications in Computacional Fluid Dynamics and Electromagnetics*, Scientific Computation, NY.

[18] Joe Thompson, Bharat Soni y Nigel Weatherill (1999) *Handbook of Grid Generation*.

[19] J. Stam (1999). *Stable Fluids*, SIGGRAPH 99 Conference Proceedings, annual Conference Series.

[20] Fay A. James (1999). *Mecánica de Fluidos*, Editorial CECSA.

[21] Young W. Kwon, Hyo Choong Bong (2000). *The Finite Element Method using Matlab*. 2ed, CRC Florida.

[22] J. Stam (2001). *A Simple Fluid Solver based on the FFT*, Journal of Graphics Tools.

[23] D. De Cecchis (2001). *Una librería para resolver ecuaciones diferenciales parciales elípticas de 2º orden en 2D por el método de elementos finitos mínimos cuadrados*. Universidad de Carabobo.

[24] V. Thomee (2001). *From finite differences to finite elements. A short history of numerical analysis of partial differencial equations*, Journal of Computational and Applied Mathematics.

[25] T. Manteuffel, Hans De Sterck, Steve McCormick y Luke Olson (2002). *Least-squares finite element methods for linear hyperbolic PDEs*, Department of Applied Mathematics, University of Colorado at Boulder.

[26] M. Potter, D. Wiggert (2002). *Mecánica de Fluidos*. Thompson. 3ra edición

[27] G. Starke (2002). *Least-squares finite element methods for the stress-displacement formulation of elasticity*, Institud f'ur Angewandte Mathematik, Universit"at Hannover.

[28] Jianming Jin (2002). *The Finite Element Method in Electromagnetics*. Sec Edition. NY.

[29] Farid Moussaoui (2002) *A unified approach for inviscid compressible and nearly incompressible flow by least-squares finite element method*. IMACS. Elsevier Science.

[30] M. Fagúndez y J. Medina (2003). *Simulación Numérica de Flujo de Fluido Viscoso Incompresible en 2D*. Universidad de Carabobo.

[31] C. E. Cadenas R (2003). *Formulación y aplicacición del método de elementos finitos mínimos cuadrados a un problema de dispersión de onda y comparación con otros métodos numéricos*. Universidad de Carabobo.

[32] C. E. Cadenas y V. Villamizar (2003). *Application of Least Squares Finite Element Method to Acoustic Scattering and Comparision with Other Numeral Techniques*, NACoM-2003 Extended Abstracts.

[33] C. E. Cadenas y V. Villamizar (2004). *Comparision of Least Square FEM, Mixed Galerkin FEM on an implicit FDM Applied to Acoustic Scattering*, Appl. Num Anal. COmp. Math **1** No 1, 128-138, 2004.

[34] J Rojas (2004). *Un Conjunto de Herramientas de Software para resolver ecuaciones diferenciales ordinarias de segundo orden en 1D*.

[35] Brígida Molina, Marcos Raydán (2004). *Métodos Iterativos Tipo Krylov para Sistemas Lineales*. Mérida.