INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA
UNIDAD PROFESIONAL TICOMÁN

DESARROLLO DE PAQUETERÍA DE SOFTWARE OPEN SOURCE EN LENGUAJE PYTHON PARA LA GENERACIÓN DE MALLAS

TESIS
Que para obtener el grado de

INGENIERO EN AERONÁUTICA

PRESENTA

MARCO ANTONIO CARDOSO MORENO

DIRECTOR DE TESIS: M. EN C. RAFAEL MEDINA NOGUERÓN

Ciudad de México

Dedicado a Floresauria Fulgencia, amor de mi vida!

Agradecimientos

Resúmen

Indice

Indice de Figuras

Indice de Tablas

Nomenclatura

Introducción

En general, para analizar y diseñar sistemas en los cuales interviene el flujo de fluidos se cuenta con dos herramientas: la experimentación y el cálculo. La experimentación implica la construcción de modelos que serán probados en túneles de viento u otras instalaciones. En el caso del cálculo, se puede realizar de manera analítica o mediante el uso de métodos númericos, a esta última técnica se le da el nombre de dinamica de fluidos computacional (CFD, por sus siglas en inglés). El uso de la DFC (dinámica de fluidos computacional) se ha popularizado gracias al desarrollo de las capaciades de las computadoras, ya que las simulaciones presentan ciertas ventajas frente al enfoque exprimental en términos de velocidad, seguridad y en la mayoría de casos, de costo. Gracias a esto la DFC es ampliamente usada en la actualidad en sectores de la industria como el aeroespacial, petrolero, entre otros. En el ámbito de la investigación tambien se tiene en la DFC una herramienta importante, pues permite analizar fenomenos complejos que pueden resultar dificiles de reproducir en experimentos.

Se realiza el presente trabajo con la finalidad tanto de generar en los estudiantes un interés por la dinámica de fluidos computacional (CFD), como de fomentar el desarrollo de este campo mediante la implementación de códigos. Por otro lado, se busca fomentar en el lector el uso y desarrollo de software libre.

En el capítulo 1....
**********añadir descripción breve de los capítulos**********

Objetivo

Desarrollar códigos que sean capaces de generar mallas mediante diferentes métodos. Dichos códigos se implementarán en lenguaje Python 3, mediante la IDE (entorno de desarrollo integrado, por sus siglas en inglés) Spyder, la cual fue creada pensando específicamente en la implementación de códigos enfocados a aspectos científicos y numéricos, que además presenta una interfaz gráfica similar a la de Matlab u Octave.

El programa debe ser capaz de generar mallas mediantes diferentes métodos, en este trabajo se presta principal atención a las mallas generadas por métodos algebráicos, así como por métodos basados en la solución de sistemas de ecuaciones diferenciales parciales, más específico, en EDP elípticas. Se debe generar la malla para cualquier superficie o forma deseada, en este trabajo se hace la generación de mallas alrededor de perfiles alares, por lo que se debe desarrollar un módulo que genere la nube de puntos que describe un perfil NACA de la serie 4, pudiendo ser ajustable la densidad de puntos de la misma. También se da la opción de importar la nube de puntos de cualquier otro perfil, con la restricción de que la densidad de la misma, depende de los datos de entrada que recibe el código.

Una vez generado el mallado del dominio, se creará un archivo que contenga toda la información de la misma, el cual podrá ser importado a cualquier software *"solver"*, para poder llevar a cabo el análisis de flujo deseado.

Motivación

Existe en la actualidad, una vasta cantidad de software enfocado a la dinámica de fluidos computacional, tanto software privativo (o de licencia) como software libre. Para el caso del software privativo, el usuario paga por la licencia de uso del programa con lo cual puede usarlo, pero no tiene forma de saber la forma en la que el programa funciona ni la implementación del mismo. En cuanto al software libre, el usuario tiene la posibilidad de acceder al código fuente con el cual corre el programa, para analizarlo, comprenderlo e incluso de ser necesario, modificarlo ya sea para mejorar una característica ya implementada, o bien, para agregar una nueva característica al programa. Tanto el uso de software privativo, como la poca implementación de códigos y documentación accesible en México, han conducido a un punto en el que la comunidad académica en el país se limite únicamente a usar los programas sin un entendimiento claro de su funcionamiento, excluyéndose a sí misma del desarrollo de software.

Se desarrolla entonces, este trabajo con la intención de proporcionar información sobre la implementación de códigos de generación de mallas, para que el lector pueda a su vez, continuar contribuyendo al desarrollo de software libre para la dinámica de fluidos computacionala nivel nacional.

Los códigos aquí presentados quedan abiertos para que cualquier persona, ya sea alumno, profesor, investigador o simplemente entusiasta de la dinámica de fluidos computacional, pueda revisarlos, utilizarlos en beneficio propio, modificarlos, e incluso construir software nuevo a partir de éste.

Capítulo 1

Estado del Arte

1.1. Historia de la dinámica de fluidos computacional

La DFC tiene sus orígenes en el desarrollo de dos métodos númericos que son las herramientas base de esta disciplina, el método de diferencias finitas y el método de elementos finitos. En 1910, Richardson presentó en la "Royal Society of London" un artículo con una solución mediante el método de diferencias finitas para un análisis de la distribución de esfuerzos de una presa de mampostería. Por otro lado, el primer trabajo mediante el método de elementos finitos, fue publicado en 1956 en el Aeronautical Science Journal por Turner, Clough, Martin y Topp, el cual trata aplicaciones del método para el análisis de esfuerzos en aeronaves. [1]

Durante la segunda guerra mundial, así como los años siguientes a la misma, el profesor John Von Neumann desarrolló un método para determinar la estabilidad numérica para la resolución de problemas dependientes del tiempo. Su trabajo fue publicado por O'brien en 1950. El método de Von Neumann es ampliamente aplicado hoy en día para determinar la estabilidad númerica. [2]

En la década de los años 60s se dan los primeros esfuerzos en el desarrollo de técnicas para la generación de mallas. [3]

Al inicio de la década de 1970 se da un auge en la DFC gracias al incremento de las capacidades de procesamiento de las computadoras, incluso hoy en día el avance de la DFC va de la mano con el desarrollo de las computadoras. [4] Es en ésta década cuando se desarrollan, gracias a un grupo del "Imperial College" de Londres, algoritmos para flujos incompresibles de baja velocidad, así como el algoritmo SIMPLE (Semi-Implicit Method for Pressure-Linked Equations), lo cual sirvió como base para el desarrollo de esquemas de solución para las ecuaciones de Navier-Stokes para flujo incompresible. [2]

Para la década de 1980 se comienza a dar solución a las ecuaciones de Euler tanto en dos dimensiones como en tres dimensiones. Para mediados de esta decada, gracias de nuevo, al incremento en la capcidad de cálculo de los ordenadores, se comienza a hacer análisis de flujos viscosos, los cuales son descritos por las ecuaciones de Navier-Stokes. [4]

A finales de la década de 1980, comenzó una nueva etapa el el desarrollo de técnicas para la generación de mallas. Dicha etapa se caracteriza por la implementación de códigos de generación

de mallas comprehensivas, multi propósito, tridimensionales. [3]

1.2. La dinámica de fluidos actualmente

La DFC tiene hoy en día, un rol tan importante, que puede considerarse una tercera rama de la dinámica de fluidos, siendo las dos primeras las ramas experimental y la teoría pura. [5]

Actualmetne, la DFC se aplica en diversos sectores, como el aeroespacial, diseño de turbomaquinaria, carros y navíos. Además de campos como la meteorología, oceanografía, astrofísica e incluso arquitectura. Algunas técnicas desarrolladas para la DFC ahora se usan también en la solución de las ecuaciones de Maxwell, por lo que demuestra la creciente importancia que tiene la DFC como herramienta en ingeniería. [4]

Uno de los ámbitos en los que la DFC ha tenido mayor impacto es en el diseño de aeronaves, gracias al rápido decremento en los costos de computación, ocasionado por una mejora en las capacidades de cálculo y tecnológicas de los ordenares, así como de el fácil acceso a los mismos, aunado a un incremento en el costo de experimentos en túneles de viento. Esto ha dado como resultado que el cálculo de las características aerodinámicas de una aeronave resulte más barato mediante DFC que mediante la experimentación en túnel de viento. De modo que en la industria el diseño preliminar de aeronaves se realiza mediante cálculos en computadoras, mientras que los detalles del diseño final se analizan experimentalmente. [5]

En la actualidad, la DFC es capaz de analizar flujos laminares sin mayor complicación, sin embargo, aún no es capaz de analizar flujos turbulentos, los cuales se presentan en la mayoría de casos prácticos, por lo que se debe de recurrir a modelos de turbulencia, los cuales dan buenos resultados, pese a que ningún modelo es universal, y se debe prestar atención al modelo que se desea emplear. [6]

1.3. La generación de mallas actualmente

En la actualidad se ha desarrollado un gran número de métodos avanzados para la generación de mallas, entre los que destacan los métodos algebráicos, mediante la solución de ecuaciones diferenciales parciales elípticas, hiperbólicas, parabólicas, mallas variables, etc. La creación y mejora de todas estas técnicas nos ha llevado a un punto en el que se pueden realizar análisis en dominios de geometrías complejas.

Debido al desarrollo exitoso de esta área, el campo de generación numérica de mallas puede ser considerado una nueva disciplina matemática, con su propia metodología, tecnología y su propio enfoque.

La generación numérica de mallas, al ser una herramienta de amplia utilización tanto en la industria de la ingeniería, como en campos de computación científica, es ahora reconocida como una asignatura en algunas universidades. [3]

1.4. Software disponible para DFC

OpenFOAM

OpenFOAM (Open Source Field Operation and Manipulation) es un entorno de trabajo, para el desarrollo de ejecutables de aplicaciones que usa las funciones contenidas en aproximadamente 100 librerías escritas en C++. Contiene solvers específicos para cada tipo de problema en mecánica de fluidos, así como de mecánica del medio continuo. [7]

OpenFOAM se distribuye bajo la licencia GPL (General Public License), la cual garantiza a los usuarios finales la libertad de usar, estudiar, compartir y modificar el código fuente del software, lo cual es basicamente, la filosofía del software libre.

XFLR5

XFLR5 es una herramienta desarrollada para el análisis de perfiles alares, alas y aviones volando a bajos Números de Reynolds. Fue creado con dos propósitos princiales, dar una interfaz amigable al programa "XFoil", y pasar el código fuente original de FORTRAN a lenguaje C / C++, es decir que los agoritmos con los que se desarrolló Xfoil son exactamente los mismos en XFLR5. [8] Este programa no ha sido desarrollado con fines profesionales, sino para uso meramente personal y se distribuye bajo la licencias GPL.

XFoil

Es un programa interactivo para el diseño y análisis de perfiles operando en regímenes subsónicos. Dadas las coordenadas de la nube de puntos que describen la forma de un perfil, un número de Reynolds y el número de Mach, Xfoil puede calcular la distribución de presión a lo largo del perfil, y con esto, las fuerzas de levantamiento y arrastre. Consiste en una colección de funciones como:

- Análisis Viscoso de perfiles alares existentes.
- Diseño y rediseño de perfiles mediante la modificacion de la distribución de velocidad superficial
- Rediseño de perfiles mediante la modificación de sus parámetros geométricos.

[9]

Xfoil es desarrollado por el MIT (Massachusetts Institute of Technology), escrito en lenguaje FORTRAN y distribuido bajo la licencia GPL.

XFlow

Es un software privativo desarrollado por la empresa Dassault Systèmes, el cual ofrece soluciones mediante el método de Lattice-Boltzman basado en partículas. El usuario puede analizar casos de aerodinámica transitoria, flujos multifase complejos, acústica aérea, geometrías en movimiento e interacciones fluido-estructura. Destaca por sus capacidades de renderizado. [10]

ANSYS Fluent

Ansys Fluent es un software privativo creado por la empresa ANSYS, Inc. dedicado a la simulación de DFC, implementado para dar soluciones mediante el método de volúmenes finitos.

SU2

SU2 es una colección de software *"open source"* desarrollado en los lenguajes Python y C++ para el análisis de ecuaciones diferenciales parciales mediante métodos numéricos, del actual estado del arte de la DFC. Enfocado principalmente a su aplicación en la industria aeronáutica, así como la industria automotriz, naval y de energías renovables, entre otras. [11]

Capítulo 2

Discretización Espacial

Las soluciones analíticas modeladas mediante ecuaciones diferenciales parciales de las ecuaciones que describen el flujo de fluidos dan resultados "continuos" a lo largo del dominio que se está estudiando, por otro lado, las soluciones numéricas dan resultados en puntos "discretos" del dominio llamados nodos, éstos, unidos mediante líneas, generan la malla del dominio en el que se está trabajando el problema. Para el caso de un análisis en dos dimensiones, la malla se forma por triangulos o cuadriláteros, mientras que en un caso de tridimensional la malla es conformada por tetrahédros o hexaédros.

En las aplicaciones de DFC se busca que la distribución de los nodos sea uniforme, dado que esto simplifica los algoritmos de solución a implementar, y que se traduce en un ahorro de tiempo en la ejecución de los programas, además de un uso considerablemente menor de memoria.

Existen básicamente dos tipos de mallas:

- Estructuradas: los nodos de la malla están alineados entre sí y son identificados mediante ínidices i, j, k. Una malla estructurada bidimiensional está formada por cuadriláteros, mientras que una malla tridimensional se forma por hexahédros. Figura 3.2(a).

- No estructuradas: los nodos no tienen un orden particular, por lo que no se pueden identificar mediante índices, por lo que se debe llevar a cabo una forma diferente de identificación. Figura 3.2(b)

2.1. Método de Diferencias Finitas

El método de diferencias finitas fue de los primero métodos numéricos que se emplearon para la resolución de ecuaciones diferenciales parciales, y ha sido hasta la fecha, uno de los más utilizados en aplicaciones de DFC. Este método está basado en las series de expansión de Taylor.

"El uso del método de diferencias finitas en la DFC tiene como propósito el reemplazar las derivadas parciales que aparecen en las ecuaciones que gobiernan el flujo, por coeficientes diferenciales que dan como resultado un sistema de ecuaciones algebráicas, el cual se resuelve para obtener las propiedades del campo de flujo en puntos discretos, es decir, en los nodos de la malla." [5]

2.1.1. Desarrollo del método

Supongamos que se tiene el valor de una propiedad de flujo de $f_{i,j}$ en un punto (i, j) y se pretende calcular el valor de dicha propiedad en un punto $(i+1, j)$ (Figura 2.1) es decir, se quiere conocer el valor de $f_{i+1,j}$, dicho valor se puede expresar mediante una serie de expansión de Taylor, quedando:

$$f_{i+1,j} = f_{i,j} + \left(\frac{\partial f}{\partial x}\right)_{i,j} \Delta x + \left(\frac{\partial^2 f}{\partial x^2}\right)_{i,j} \frac{(\Delta x)^2}{2\,!} + \left(\frac{\partial^3 f}{\partial x^3}\right)_{i,j} \frac{(\Delta x)^3}{3\,!} + \cdots + \left(\frac{\partial^n f}{\partial x^n}\right)_{i,j} \frac{(\Delta x)^n}{n\,!} \quad (2.1)$$

Matemáticamente, esta serie es una solución exacta bajo una de dos condiciones:

- La serie está formada por un número infinito de términos, haciendo que la serie converja
- El valor del intervalo entre puntos tiende a cero ($\Delta x \to 0$)

Para un análisis computacional resulta impractico el trabajar con un número infinito de términos, por lo que se trunca la ecuación y se busca refinar la malla, haciendo el intervalo Δx lo más pequeño posible, incrementando de esta manera la exactitud de la solución.

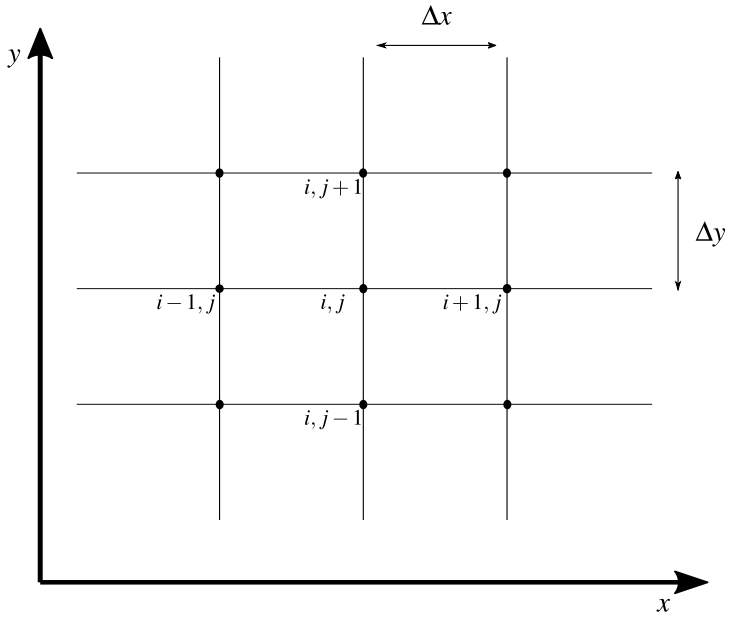


Figura 2.1: Discretización espacial (malla) por el método de diferencias finitas [5]

Todos los términos de la serie truncados se agrupan en un sólo término conocido como error de truncamiento y es representado por la simbología $O()$. Volviendo a la ecuación (2.1) y resolviendo para $\left(\frac{\partial f}{\partial x}\right)_{i,j}$ tenemos:

$$\left(\frac{\partial f}{\partial x}\right)_{i,j} = \frac{f_{i+1,j} - f_{i,j}}{\Delta x} - \left(\frac{\partial^2 f}{\partial x^2}\right)_{i,j} \frac{\Delta x}{2} - \left(\frac{\partial^3 f}{\partial x^3}\right)_{i,j} \frac{(\Delta x)^2}{6} - \cdots \quad (2.2)$$

o bien:

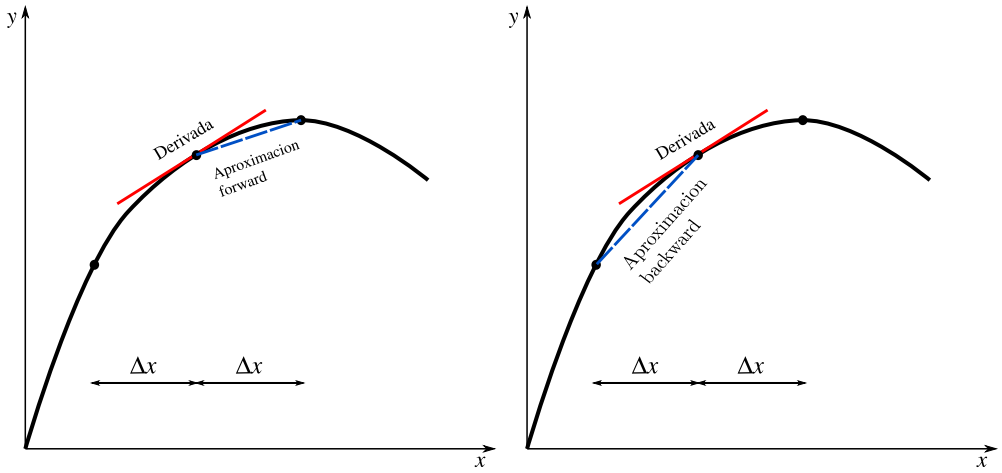
$$\left(\frac{\partial f}{\partial x}\right)_{i,j} = \frac{f_{i+1,j} - f_{i,j}}{\Delta x} + O(\Delta x)^2$$

que para simplificar, queda:

$$\left(\frac{\partial f}{\partial x}\right)_{i,j} = \frac{f_{i+1,j} - f_{i,j}}{\Delta x} \tag{2.3}$$

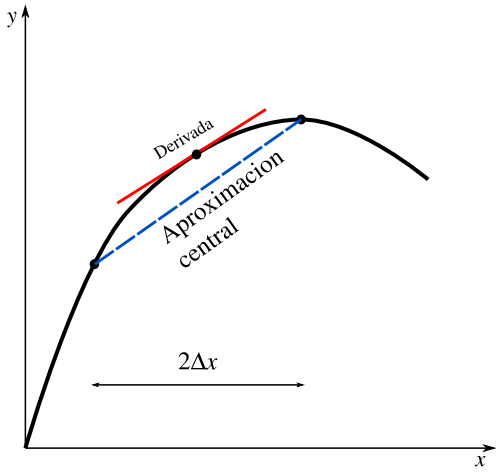
El símbolo $O(\Delta x)$ representa el error de truncamiento de la serie de expansión, y para este caso particular se dice que tiene una precisión de primer orden, ya que se están despreciando los términos de orden superior.

La ecuación 2.3 se le conoce como una aproximación tipo "forward" de primer orden y representa el valor aproximado del valor de la derivada parcial $\left(\frac{\partial f}{\partial x}\right)_{i,j}$ calculada mediante la relación lineal del punto (i, j) con el punto $(i + 1, j)$. (Figura 2.2(a))



(a) Aproximación forward

(b) Aproximación backward



(c) Aproximación central

Figura 2.2: Esquemas de las aproximaciones por el método de diferencias finitas. [12]

Para obtener una aproximación tipo "backward" se debe generar una serie de Taylor para el punto $(i-1,j)$ expandiendo a partir del punto (i,j)

$$f_{i-1,j} = f_{i,j} + \left(\frac{\partial f}{\partial x}\right)_{i,j} \Delta x + \left(\frac{\partial^2 f}{\partial x^2}\right)_{i,j} \frac{(-\Delta x)^2}{2\,!} + \left(\frac{\partial^3 f}{\partial x^3}\right)_{i,j} \frac{(-\Delta x)^3}{3\,!} + \cdots + \left(\frac{\partial^n f}{\partial x^n}\right)_{i,j} \frac{(-\Delta x)^n}{n\,!} \quad (2.4)$$

Una vez generada la serie, se sigue el mismo procedimiento empleado en la derivación de la aproximación "forward" para obtener:

$$\left(\frac{\partial f}{\partial x}\right)_{i,j} = \frac{f_{i,j} - f_{i-1,j}}{\Delta x} \quad (2.5)$$

que es la ecuación de una aproximación "backward". (Figura 2.2(b))

Para diferentes aplicaciones de DFC, no basta con tener aproximaciones con precisión de primer orden, por lo cual se opta por generar una ecuación que tenga una precisión de segundo orden, a esta se le conoce como "aproximación central de segundo orden". La deducción de ésta, parte de la resta de la ecuación 2.1 y la ecuación 2.4, que da como resultado:

$$f_{i+1,j} - f_{i-1,j} = 2\left(\frac{\partial f}{\partial x}\right)_{i,j} \Delta x + 2\left(\frac{\partial^3 f}{\partial x^3}\right)_{i,j} \frac{(\Delta x)^3}{3\,!} + \cdots + 2\left(\frac{\partial^n f}{\partial x^n}\right)_{i,j} \frac{(\Delta x)^n}{n\,!} \quad (2.6)$$

que se puede simplificar a:

$$\left(\frac{\partial f}{\partial x}\right)_{i,j} = \frac{f_{i+1,j} - f_{i-1,j}}{2\Delta x} \quad (2.7)$$

se observa en la ecuación 2.7 que para obtener un coeficiente en el punto (i,j) se está utilizando información proveniente de los nodos $(i-1,j)$ e $(i+1,j)$, adyacentes a dicho punto. (Figura 2.2(c))

Para el análisis de flujos viscosos, además de los coeficientes que sustituyen a las derivadas parciales de primer orden se necesitan desarrollar aproximaciones para las derivadas parciales de segundo orden, dado que, en las ecuaciones que describen el flujo de fluidos viscosos, las ecuaciones de Navier-Stokes, los términos de mayor orden son derivadas parciales de segundo orden.

Si se suman las ecuaciones 2.1 y 2.4, se obtiene un coeficiente para $\frac{\partial^2 f}{\partial x^2}$, como se muestra a continuación:

$$f_{i+1,j} + f_{i-1,j} = 2f_{i,j} + \left(\frac{\partial^2 f}{\partial x^2}\right)_{i,j} \frac{(\Delta x)^2}{2\,!} + \left(\frac{\partial^4 f}{\partial x^4}\right)_{i,j} \frac{(\Delta x)^4}{4\,!} + \cdots + \left(\frac{\partial^n f}{\partial x^n}\right)_{i,j} \frac{(\Delta x)^n}{n\,!}$$

despejando la segunda derivada parcial y despreciando el error de truncamiento:

$$\left(\frac{\partial^2 f}{\partial x^2}\right) = \frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{(\Delta x)^2} \quad (2.8)$$

a la ecuación 2.8 se le conoce como aproximación central para la derivada de segundo orden.

Ahora se desea obtener una aproximación por diferencias finitas para una derivada parcial mixta, llámese $\frac{\partial^2 f}{\partial x \partial y}$, donde f es una función dependiente de la posición de la partícula de fluido a lo largo de dos ejes, x y y. Dado que:

$$\left(\frac{\partial^2 f}{\partial x \partial y}\right) = \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x}\right) \tag{2.9}$$

podemos desarrollar una aproximacion por diferencias "central" de la derivada parcial de y en función de x, es decir:

$$\left(\frac{\partial^2 f}{\partial x \partial y}\right)_{i,j} = \frac{\left(\frac{\partial f}{\partial y}\right)_{i+1,j} - \left(\frac{\partial f}{\partial y}\right)_{i-1,j}}{2\Delta x} \tag{2.10}$$

y desarrollando una diferencia central para la derivada parcial de f con respecto a y tenemos:

$$\left(\frac{\partial^2 f}{\partial x \partial y}\right)_{i,j} = \frac{1}{2\Delta x}\left[\frac{f_{i+1,j+1} - f_{i+1,j-1}}{2\Delta y} - \frac{f_{i-1,j+1} - f_{i-1,j-1}}{2\Delta y}\right] \tag{2.11}$$

$$\left(\frac{\partial^2 f}{\partial x \partial y}\right)_{i,j} = \frac{1}{4\Delta x \Delta y}\left(f_{i+1,j+1} + f_{i-1,j-1} - f_{i+1,j-1} - f_{i-1,j+1}\right) \tag{2.12}$$

La misma metodología se aplica para obtener las aproximacinoes por el método de diferencia finitas de una función respecto a un eje cualquiera, llámese eje y, ξ o η.

La mayor ventaja que presenta el método de diferencias finitas es su simplicidad de implementación, aunque presenta una limitante, el método únicamente es aplicable a mallas estructuradas, además de tampoco poderse aplicar a cuerpos de geometría curvilíneas, por lo que es necesario hcer una transformación de la malla, de un dominio físico a un dominio lógico, como se analiza en el capítulo 3.

Es este último punto, la piedra angular en la importancia del desarrollo de métodos de generación de mallas, ya que se busca trabajar con mallas en las cuales, en sus sistema de coordenadas computacional, pueda ser aplciado el método de diferencias finitas para la solución de problemas.

Capítulo 3

Mallas

Las mallas proveen soporte matemático para llevar a cabo una solución numérica de las ecuaciones que gobiernan el campo de análisis como medio continuo. La solución numérica se obtiene superponiendo la malla sobre el medio continuo, discretizando las ecuaciones respecto a la malla y por último, aplicando un algoritmo numérico de solución a la discretización de las ecuaciones. Este proceso resulta ser una evaluación de la solución en los nodos de la malla.

La mecánica de fluidos trabaja con ecuaciones no lineales, las cuales describen los flujos de fluidos, dichas ecuaciones en la mayoría de los casos prácticos de análisis no pueden ser resueltas de una manera analítica, por lo que se ha optado por la solución de dichas ecuaciones mediante métodos de aproximaciones entre los cuales encontramos los métodos de expansión y perturbación, método de diferencias finitas, método de volúmenes finitos e incluso el método de elementos finitos. En general los de mayor aplicación práctica son los tres últimos, pero para poder usarlos es necesario discretizar el campo de análisis mediante el uso de una malla. [13]

La utilización del método de diferencias finitas es directa si el problema se puede expresar en un sistema de coordenadas cartesianas, aunque el mismo puede aplicarse a sistemas de coordenadas polares, cilindricas o esféricas, dando como resultado una malla rectangular estructurada, con espaciamiento uniforme entre los nodos en la dirección de los ejes del sistema. La mayoría de casos prácticos de interés en la DFC, tratan con geometrías complejas, lo que hace que sea muy dificil, si no imposible, generar una malla en alguno de los sitemas de cordenadas antes mencionados, que ajuste en su frontera interna de manera exacta a la forma de la geometría a analizar. Esto presenta una limitante que se debe atender, se debe llevar a cabo una transofrmación de sistemas de coordenadas, llevando el dominio físico del problema a un dominio computacional. (Figura 3.1) El dominio computacional es una abstración matemática, mientras que el dominio físico es el dominio continuo para el cual se desea una solución numérica.

Una malla es un conjunto de puntos destribuidos a lo largo de un campo dentro del cuál se llevaran a cabo los cálculos para obtener la solución de ecuaciones diferenciales parciales. Existen dos tipos principales de mallas: estructuradas y no estructuradas. (Figura 3.2) Las mallas estructuradas se forman mediante la intersección de coordenadas curvilíneas, formando celdas cuadriláteras en dos dimensiones, o hexahedros en tres dimensiones. En contraste, las mallas no estructuradas no tienen relación alguna con las direcciones coordenadas del sistema, éstas consisten generalmente de triángulos y tetahédros en dos y tres dimendiones respectivamente, aunque en realidad se puede hacer uso de calquier forma geométrica. Para este trabajo se considerará el desarrollo de mallas

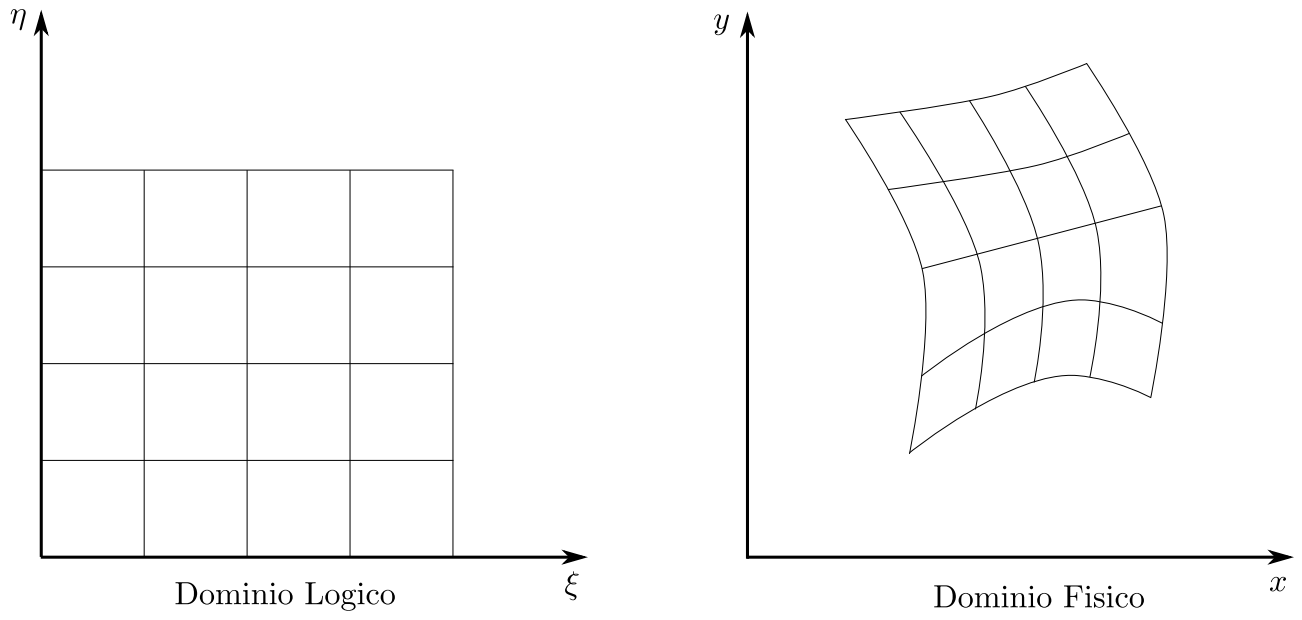
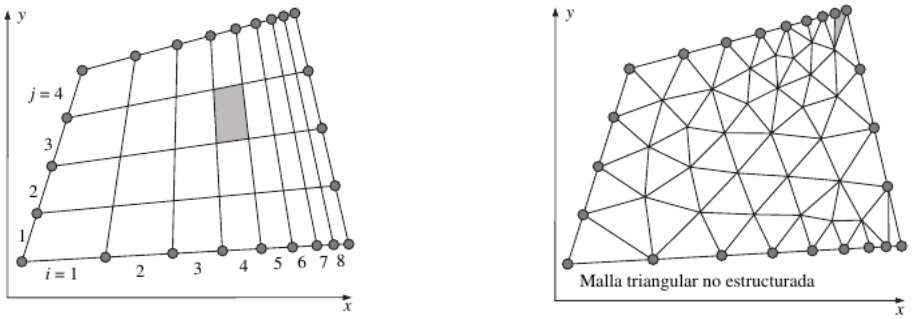


Figura 3.1: Dominios lógico (izquierda) y físico (derecha). [14]

estructuradas principalente, que en principio tendrán una forma curvilinea ajustada a la forma del perfil alar con el que se esté trabajando como frontera interna del dominio.



(a) Malla Estructurada (b) Malla no Estructurada

Figura 3.2: Mallas estructuradas y no estructuradas. [6]

La malla debe de generarse bajos ciertas restricciones las cuales suelen ser difíciles de satisfacer por completo. En la actualidad el tiempo que toma generar una malla llega a ser mayor en órdenes de magnitud que el tiempo requerido para la construcción y análisis de la solución del flujo sobre la malla. En especial ahora que hay mayor disponibilidad de software para la solución de flujos. [13]

3.1. Tipos de Mallas Estructuradas

Existen tres tipos base de mallas conocidos como mallas C, H u O respectivamente. El nombre que dichos tipos de mallado reciben se debe a que su estructura asemeja a dichas letras (en mayúsculas) desde una vista de planta.

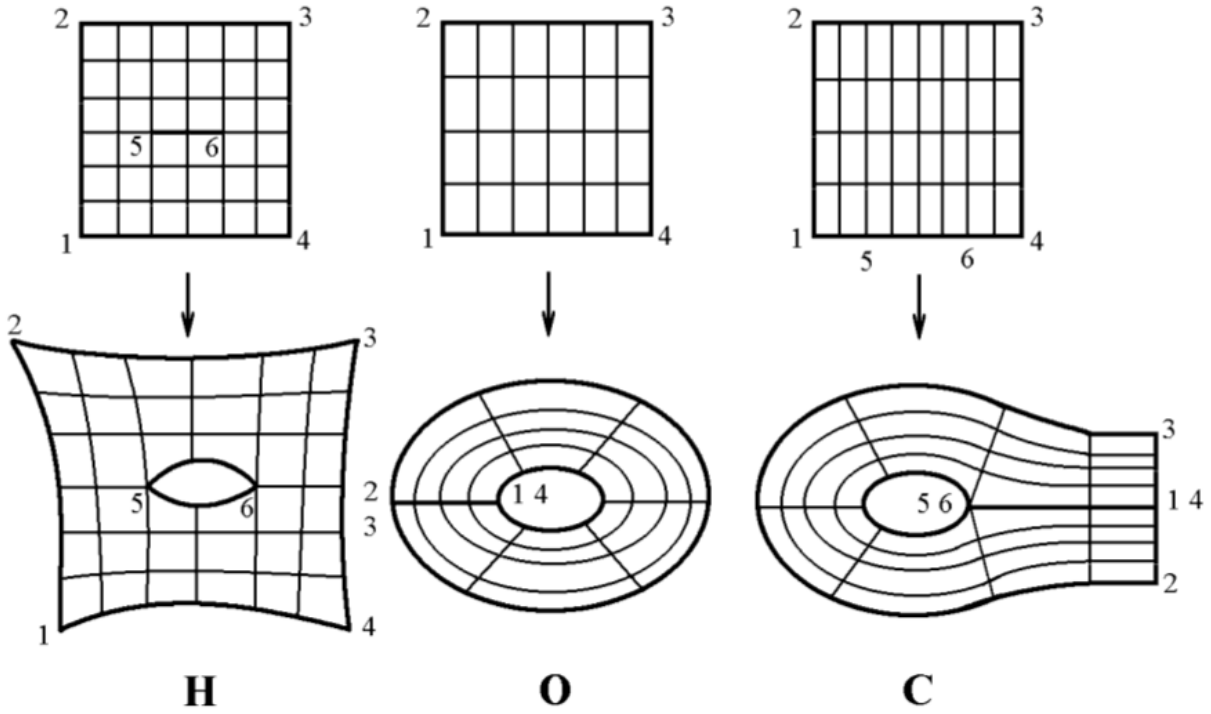


Figura 3.3: Tipología de malla y sus respectivos espacios computacionales.
H(izquierda), O(centro), C(derecha). [15]

3.1.1. Mallas tipo C

Las mallas tipos C están compuestas por una sección semicircular y otra rectangular en la frontera exterior, mientras que la frontera interior es de la misma forma del objeto que se analiza.

Suelen utilizarse para el análisis de flujos alrededor de perfiles alares, pues ofrecen como principal ventaja un buen análisis del flujo en la zona de deflexión de la estela, en especial si se compara contra las mallas tipo O. [16]

La transformación de esta malla de un espacio físico a un espacio computacional da como resultado una malla conformada por rectangulos. Durante la transformación se debe identificar un segmento, tambien conocido como corte, ya que este proceso implica conceptualemte, una separación en dicho corte, para posteriormente deformar el espacio físico llevándolo así a convertirse en una malla estructurada uniforme. En la figura 3.4 se observa que en el dominio físico, el segmento (ab) representa la sección del mallado donde se realiza el corte y que tambien se le llama $(a'b')$, y que en el dominio computacional este segmento representa dos segmentos diferentes.

3.1.2. Mallas tipo O

Este tipo de mallas son principalmente de sección circular, de ahí su nombre. Suelen ser utilizadas para el análisis del flujo alrededor de algunos componentes de aeronaves, como puede ser el fuselaje o las góndolas, entre otros. [15] Tambien pueden llegar a utilizarse en el análisis de perfiles alares, pero presentan la dificultad de ofrecer una baja calidad de mallado en el borde de salida. [4] [16]

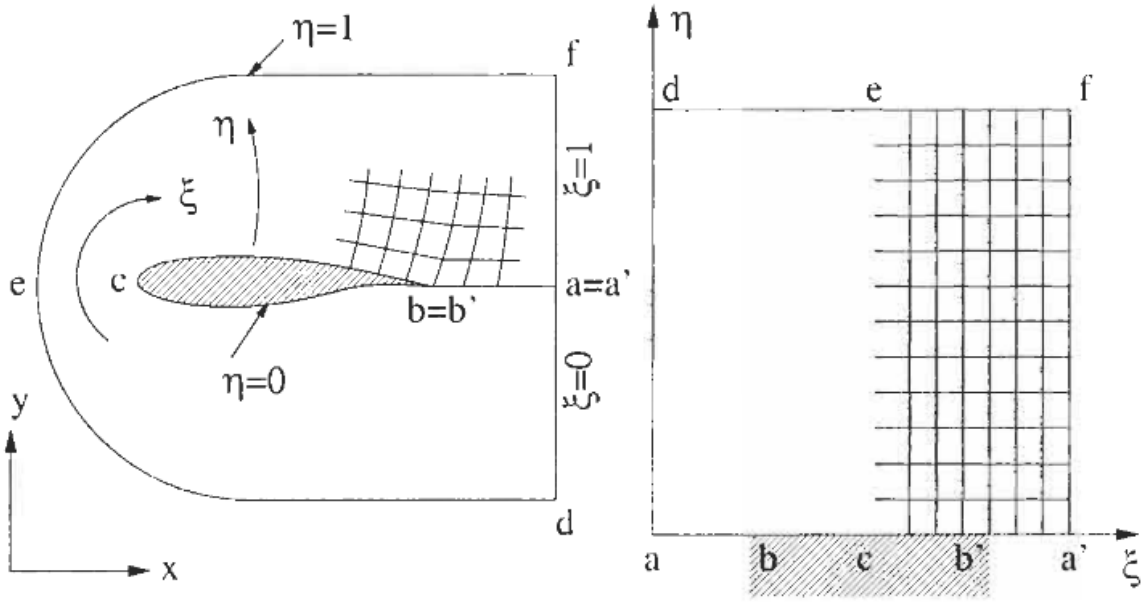


Figura 3.4: Malla tipo C y su transformación del espacio físico (izquierda) al espacio lógico (derecha). [4]

En este caso, al igual que sucede con las mallas tipo C, el resultado en el dominio lógico es un cuadrado.El proceso de transformación de este tipo de malla se puede conceptualizar como el desdoble del espacio físico a partir de un corte, en la figura 3.5 se representa como (ac) o como $(a'c')$, para después deformarlo hasta alcanzar la forma cuadrada esperada.

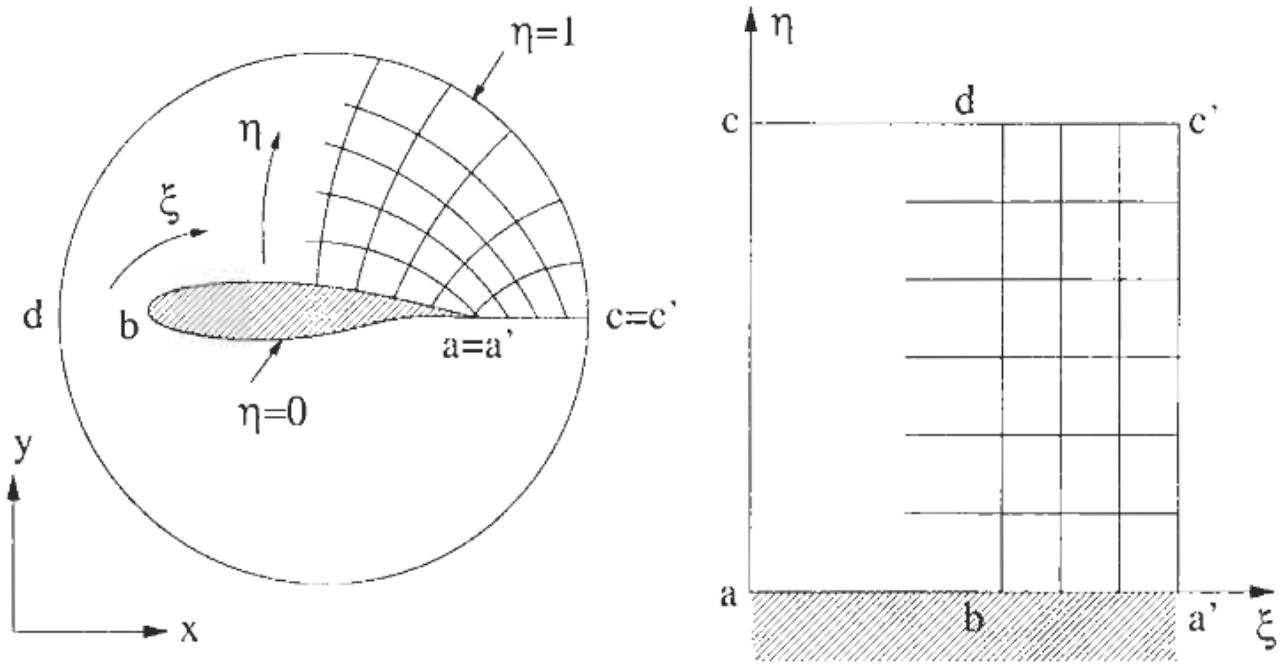


Figura 3.5: Malla tipo O y su transformación del espacio físico (izquierda) al espacio lógico (derecha). [4]

3.1.3. Mallas tipo H

Este tipo de mallas son utilizadas principalmente en los análisis de tubomaquinaria, específicamente en la zona de los alabes. [4] [16] Tambien suelen utilizarse en el análisis de alas de aeronaves. [15]

La transformación en este tipo de malla, tambien da como resultado un dominio computacional cuadrado, que puede tener o no un corte al interior el cual corresponde a la frontera interior del dominio físico. (Figura 3.3(H)) La frontera externa del dominio computacional corresponde a la frontera externa del dominio físico.

La figura 3.6 muestra el uso de una malla tipo H para el análisis del flujo a través de los álabes de una turbina, y su transformación del dominio físico al computacional. En este caso no existe un cuerpo en el centro del mallado, por lo que no hay corte alguno en el dominio lógico.

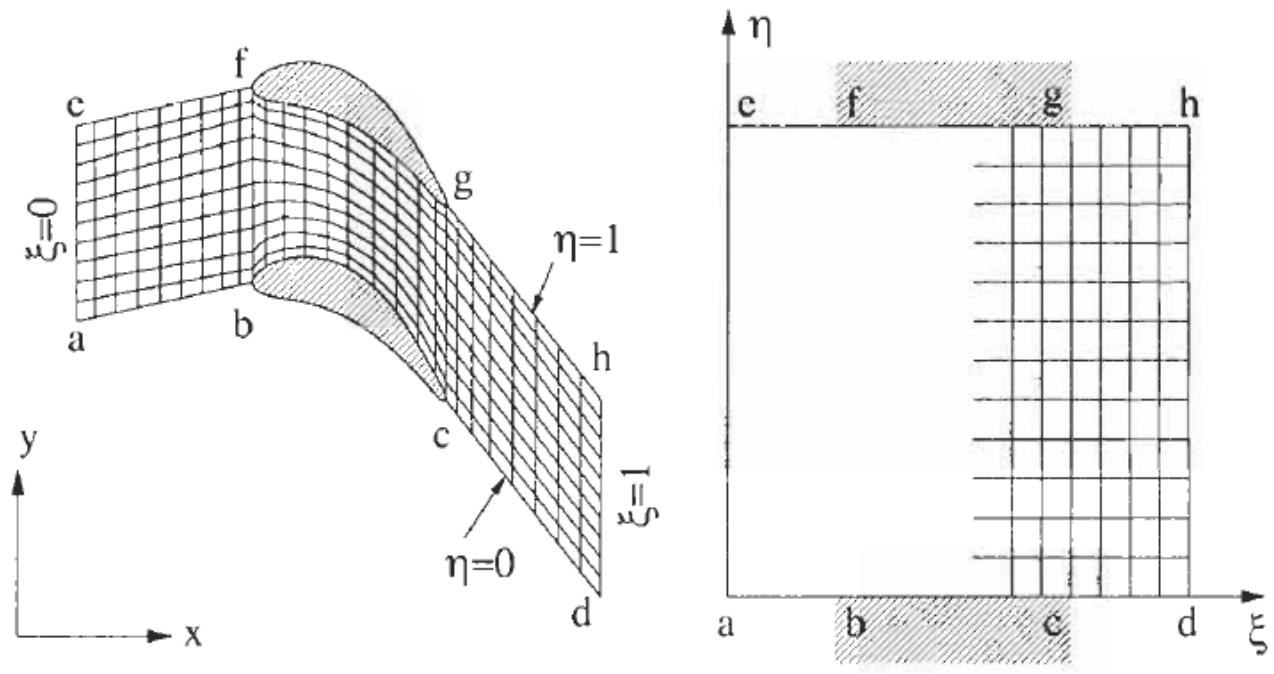


Figura 3.6: Malla tipo H y su transformación del espacio físico (izquierda) al espacio lógico (derecha). [4]

En el caso de que se quiera ocupar una malla H con un cuerpo a analizar en el interior del dominio, como el caso representado en la figura 3.3(H) se puede realizar un mallado uniforme, quizás mediante un método de interpolación algeráica, sobre el dominio físico quedando así coincidente con el dominio computacional. Los puntos 5 y 6 requieren de un trato especial al momento de llevar a cabo la solución, un ejemplo de esto puede ser que todos los puntos del mallado que caen dentro del objeto de análisis no sean utilizados durante la solución.

3.2. Mallas Adaptativas

A pesar de que existen varios modelos algebráicos y diferenciales para la generación de mallas, aún hay problemas para los cuales estos métodos no generen una malla adecuada, ya sea que la malla se aleje mucho de la ortogonalidad en ciertos puntos, que esté coprimida, e incluso no se llegue a la convergencia. Desde la década de 1960, época en la que la generación de mallas comienza a ser estudiada con mayor seriedad, se han analizado enfoques variables para solucionar los problemas presentados en los enfoques diferenciales.

3.3. Consideraciones preliminares para la generación de mallas

La densidad de la malla es el primer aspecto que se debe considerar, se debe generar una malla solo sufcientemente densa para que la aproximación numérica sea precisa, sin embargo, se debe tener cuidado en no hacer la malla demasiado densa, tanto que resulte impractico llevar a cabo un análisis y llegar a una eventual solución.

Otro importante aspecto a tener en cuenta durante este proceso, es la eficiencia computacional. Por un lado el uso de memoria podría ser tan grande que resulte impractico llevar a cabo este proceso, por lo que una organización adecuada de la información es de vital importancia. Por otro lado, la generación de mallas debe llevarse a cabo mediante la implementación de códigos lo menos complejos posible.

3.4. Técnicas de generación del mallado

La generación de mallas *per se*, es un proceso libre durante su desarrollo, es decir, no se lleva a cabo mediante alguna fórmula o procedimiento estricto, por lo tanto, cualquier desarrollo se puede considerar apto para este propósito siempre y cuando dé como resultado una mallado adecuado para el análisis que se pretende realizar. Sin embargo, si hay ciertas consideraciones a tomar en cuenta, tales como la capacidad de manejar problemas con múltiples variables, las cuales podrían variar en varios órdenes de magnitud. Del mismo modo, se debe considerar la posibilidad de presentar factores de compresión en ciertas áreas de la malla, en contraste con una malla uniforme.

Podemos señalar principalmente dos tipos de técnicas para la generación de mallado, de entre el amplio número de técnicas existentes al día de hoy, las cuales son:

- Métodos algebráicos: es de todos, el método más sencillo de implementar, una de sus ventajas es la rapidez con la que se genera la malla. Se utiliza un método de interpolación para determinar los nodos del mallado, partiendo de las coordenadas de la frontera externa e interna. Y se realiza la transformación del dominio físico al lógico mediante una ecuación algebráica.
- Metódos mediante ecuaciones diferenciales parciales (EDP): se resuelve una EDP para obtener la distribución de los nodos, la resolución de las mismas se lleva a cabo mediante aproximaciones por el método de diferencias finitas. La distribución de la malla puede ser uniforme o concentrada dependiendo del tipo de ecuación que se resuelva.

Capítulo 4

Generación de mallas mediante métodos algebráicos

Como ya se mencionó previamente, los métodos más eficientes para la generación de mallas son los métodos algebráicos. Éstos métodos basados en interpolación son de gran utilidad en la industria, debido a su sencilla implementación, además de la capacidad de control sobre la densidad de la malla respecto a un nodo dentro del dominio. Un aspecto en contra de la generación de mallas mediante estos métodos es que no tienen la capacidad de suavizar las curvas, si no que tienden a mantener la forma de las fronteras, por lo que si se llegara a presentar algún tipo de discontinuidad en la frontera interna (superficie de análisis), esta seguiría presente los nodos internos de la malla. Es por esto que estos métodos son utilizados de manera general, como una primera aproximación, la cual sirve como condición inicial en la generación de mallas mediante la solución de sistemas de ecuaciones diferenciales parciales. [17]

La idea fundamental sobre la cual se desarrollan todos los métodos algebráicos es el uso de funciones de interpolación matemática para interpolar entre algunos puntos conocidos o previamente asignados (fronteras) para así obtener la distribución de puntos que se ubican entre los ya mencionados. El método de interpolación puede variar dependiendo del método algebráico que se esté empleando, pero la idea base permanece. [18]

Los métodos algebráicos relacionan un dominio computacional, descrito por un cuadrado en 2D y un cubo en 3D, con un dominio físico de geometría arbitraria.

4.1. Interpolación unidireccional

La interpolación unidireccional se lleva a cabo interpolando una línea a la vez, en una dirección del espacio computacional, ya sea ξ o η. Es decir, para interpolar en la dirrección ξ (desarrollo que se presenta en este trabajo), se deben seleccionar puntos en las frontera tanto interna como externa, que correspondan a la misma línea $\eta = constante$ y a lo largo de la cual se hará la interpolación, haciendo la variación en la coordenada ξ, interpolando así, línea por línea hasta terminarse de generar la malla. El mismo procedimiento se puede llevar a cabo interpolando en la dirección de η.

4.1.1. Interpolación Polinomial

En este método se ocupa el polinomio de Lagrange como ecuación de interpolación para generar la malla.

$$L_i(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)(x_i-x_1)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)} \qquad i = 0, 1, \ldots, n \tag{4.1}$$

El polinomio resultante será de grado n, donde el numerador omite el término $(x - x_i)$

El caso más sencillo con el que se puede trabajar el polinomio de Lagrange es generando un polinomio de grado 1, que resulta ser una línea recta que pasa a través de dos puntos (x_0, y_0) y (x_1, y_1), los cuales pertenecen a la frontera interna y externa respectivamente. Para este caso, se tienen los siguientes términos:

$$L_0 = \frac{(x-x_1)}{(x_0-x_1)} \qquad\qquad L_1 = \frac{(x-x_0)}{(x_1-x_0)}$$

dando como resultado la ecuación que describe a una recta

$$y = y_0\frac{(x-x_1)}{(x_0-x_1)} + y_1\frac{(x-x_0)}{(x_1-x_0)} = y_0(1-\xi) + y_1\xi \tag{4.2}$$

de donde se sabe que

$$\xi = \frac{x-x_0}{x_1-x_0} \tag{4.3}$$

y se observa que $\xi = 0, 1$ cuando $x = x_0, x_1$ respectivamente.

Despejando x de la ecuación 4.3 se obtiene:

$$x = (x_1-x_0)\xi - x_0 = x_0(1-\xi) + x_1\xi \tag{4.4}$$

por lo que la ecuación constitutiva de éste método, para una interpolación en dirección del eje ξ del espacio computacional puede escribirse de la siguiente manera:

$$r(\xi, \eta_j) = (1-\xi_i)r(0, \eta_j) + \xi_i r(1, \eta_j) \tag{4.5}$$

Se puede hacer uso de la generación de un polinomio de mayor grado, para lo cual se deben proporcionar más puntos previamente definidos por los cuales tiene que pasar la curva descrita del polinomio. Para generar una ecuación de grado n se deben proporcionar como datos de inicio, al menos $n+1$, es decir, si se quisiera hacer la interpolación mediante un polinomio de grado 2, se debe proporcionar la información de al menos 3 puntos. Siempre cabe la posibilidad de que los puntos pertenezcan a una linea recta, en cuyo caso los términos de mayor orden se verán eliminados.

Las figuras 4.1 y 4.2 presentan una malla y su vista de detalle generada mediante este método alrededor de un perfil aerodinámico.

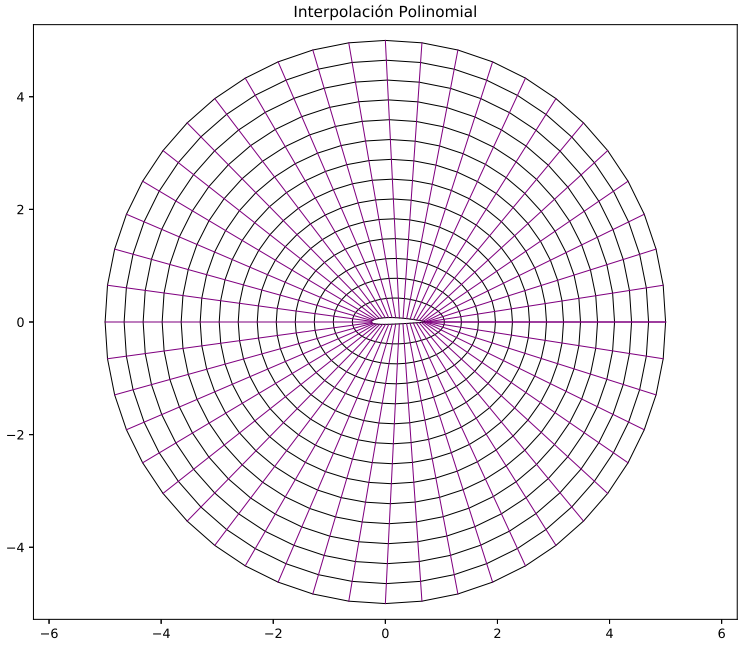


Figura 4.1: Malla tipo O generada mediante el método de interpolación polinomial alrededor de un perfil NACA 2412. Densidad de malla 49X15

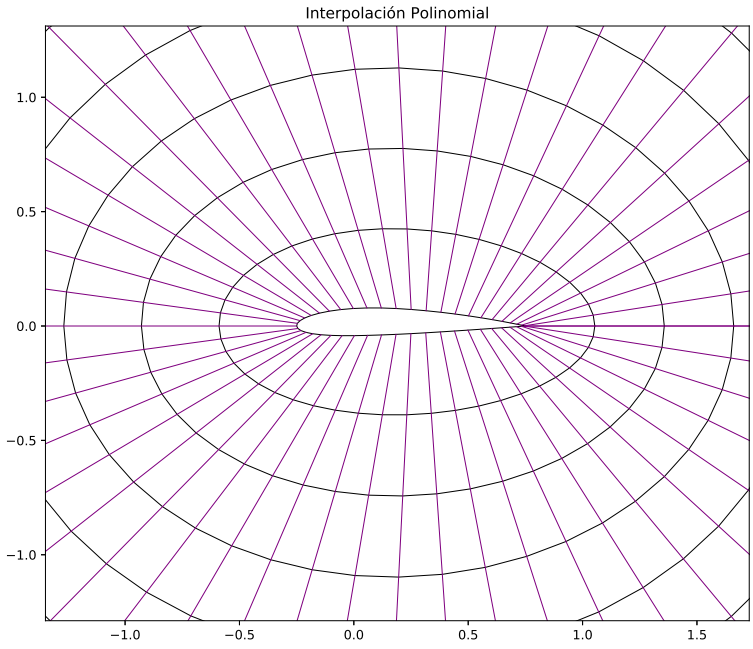


Figura 4.2: Vista de acercamiento a la malla generada de la figura 4.1

4.1.2. Interpolación mediante polinomios de Hermite

La interpolación mediante el polinomio de Lagrange se realiza mediante el uso de valores conocidos de ciertos puntos, sin embargo, es posible generar una malla partiendo tanto de valores conocidos de una función como de los valores de su primer derivada para ciertos puntos dados. El polinomio que describe este método se escribe como:

$$p(x) = \sum_{i=0}^{n} y_i H_i(x) + \sum_{i=0}^{n} y_i \prime \widetilde{H}_i(x) \tag{4.6}$$

Los siguientes polinomios definen los polinomios de Hermite $H_i(x), \widetilde{H}_i(x)$ en función de polinomios de Lagrange:

$$H_i(x) = \left\{1 - 2L\prime_i(x_i)(x - x_i)\right\}\left[L_i\right]^2 \tag{4.7}$$

$$\widetilde{H}_i(x) = (x - x_i)\left[L_i\right]^2 \tag{4.8}$$

Se suele usar polinomios de Hermite cubicos, para los cuales se requiere el uso de polinomios de Lagrange de primer grado, es decir polinomios lineales. La ecuación constitutiva de la interpolación en dirección del eje ξ, por polinomios cúbicos de Hermite queda expresada de la siguiente manera:

$$r(\xi) = \sum_{i=0}^{n} r_i H_i(\xi) + \sum_{i=0}^{n} r\prime_i \widetilde{H}_i(\xi) \tag{4.9}$$

que a su vez, desarrollando las operaciones matemáticas, se expresa como:

$$\begin{aligned} r(\xi, \eta_j) =& r(0, \eta_j)(2\xi^3 - 3\xi^2 + 1) + r(1, \eta_j)(3\xi^2 - 2\xi^3) \\ &+ r\prime(0, \eta_j)(\xi^3 - 2\xi^2 + \xi) + r\prime(1, \eta_j)(\xi^3 - \xi^2) \end{aligned} \tag{4.10}$$

4.2. Interpolación multidireccional

La interpolación multidireccional consiste en la obtención de una ecuación algebráica que permita la interpolación simultánea en 2 o más direcciones. Para este caso (2D), se generan ecuaciones que permitan la interpolación simultánea, en el espacio computaciones, en ambas direcciones ξ y η.

4.2.1. Interpolación Transfinita - TFI

El método de TFI es un método de interpolación multivariable. Cuando se aplica el TFI para la generación de mallas, se restringe la malla para que coincida con la fronteras especificadas. Este método consiste en la suma de las interpolaciones de una sola variable para cada una de las coordenadas computacionales. Existen diversos tipos de interpolación a lo largo de una coordena, por lo que existen práctimanete un número ilimitado de variantes de TFI creadas a partir de la combinación de diversas interpolaciones de una sola variable. [13]

Una de las principales ventajas del método de interpolación transfinita es que asegura una concordancia en las fronters en ambas direcciones. Su escencia es la de la interpolación en cada una de las coordenadas computacionales, formando así los productos tensores de las interpolaciones, y finalmente, se lleva a cabo una suma "booleana". Las interpolaciones unidireccionales son una combinación lineal de datos proporcionados por el usuarios del dominio físico para valores dados de las coordenadas del dominio lógico.

A partir de la existencia de una transformación $r = r(\xi, \eta)$ ($x = x(\xi, \eta), y = y(\xi, \eta)$ la cual hace un mapeo de un cuadrado de dimensiones $0 < \xi < 1, 0 < \eta < 1$ en el dominio computacional, con la región del dominio físico, delimitada por una frontera externa y una interna, que se pretende analizar. De igual modo, se puede escribir otra transformación P_ξ, la cual recibe el nombre de proyector, la cual relaciona puntos en el espacio lógico con puntos en el espacio físico, definida como:

$$P_\xi(\xi, \eta) = (1 - \xi)r(0, \eta) + \xi r(1, \eta) \tag{4.11}$$

Como ya se explicó en la sección 4.1.1, esta tranformación hace una transformación con la cual las fronteras en la dirección de ξ son mapeadas, mientras que las fronteras en dirección de η son reemplazadas por lineas rectas. De manera análoga se puede definir el proyector:

$$P_\eta(\xi, \eta) = (1 - \eta)r(\xi, 0) + \eta r(\xi, 1) \tag{4.12}$$

con el cual se preservan las fronteras en dirección η y se reemplazan las fronteras en ξ por líneas rectas.

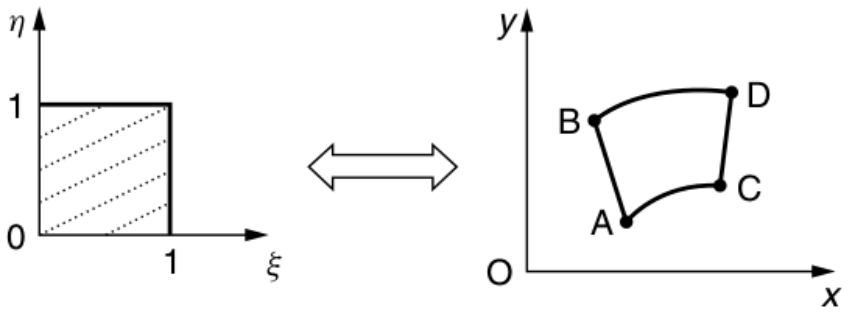


Figura 4.3: Transormación de una malla a través del proyector P_η [17]

La figura 4.3 representa gráficamente la transformación de la malla del dominio lógico al dominio computacional mediante la aplicación del proyector P_η. Se observa como las fronteras conformadas por los segmentos AC y BD se conservan durante las transformación, mientras que las fronteras AB y CD son reemplazadas por líneas rectas.

A partir de estas ecuaciones, se uede definir un mapeo compuesto $P_\xi P_\eta$, de tal manera que:

$$\begin{aligned} P_\xi(P_\eta(\xi, \eta)) &= P_\xi((1 - \eta)r(\xi, 0) + \eta r(\xi, 1)) \\ &= (1 - \xi)\left[(1 - \eta)r(0, 0) + \eta r(0, 1)\right] + \xi\left[(1 - \eta)r(1, 0) + \eta r(1, 1)\right] \\ &= (1 - \xi)(1 - \eta)r(0, 0) + (1 - \xi)\eta r(0, 1) + \xi(1 - \eta)r(1, 0) + \xi\eta r(1, 1) \end{aligned} \tag{4.13}$$

Con esta transformación compuesta, se logra preservar los 4 vértices en los que se interseccionan las fronteras, sin embargo, las cuatro fronteras son reemplazadas por líneas rectas. Es decir, las líneas $\xi = constante$ y $\eta = constante$ en el dominio computacional son transformadas como líneas rectas en el dominio físico, como se muestra en la figura 4.4.

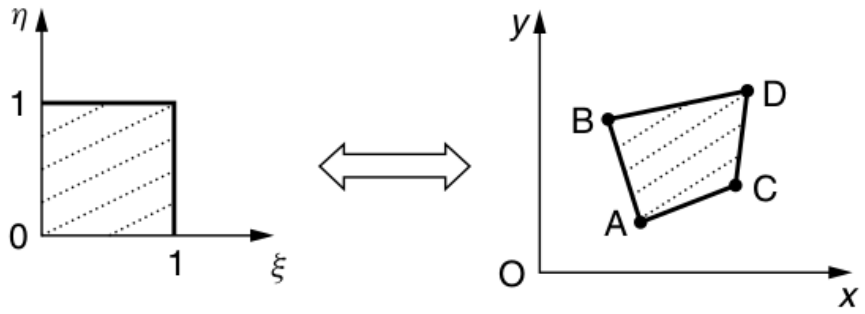


Figura 4.4: Transormación de una malla a través del proyector compuesto $P_\xi P_\eta$ [17]

Si consideramos los diferentes mapeos de un lado, bajo cualquiera de los proyectores previamente desarrollados, por ejemplo el lado $\eta = 0$, bajo el proyector P_ξ es mapeado con una línea recta AC, si por otro lado se lleva a cabo la transformación medainte el proyector P_η el mapeo resultante es uno a uno, es decir que la recta $\eta = 0$ se mapea con la curva de la frontera AC. Por último, transformando con el proyector compuesto $P_\xi P_\eta$ tambien se hace un mapeo con la línea recta AC.

Desarrollando la misma lógica y consideraciones a las 4 fronteras de la frontera, se concluye que el mapeo compuesto $(P_\xi + P_\eta - P_\xi P_\eta)$ lleva a cabo una transformación con un mapeo uno a uno de las fronteras del cuadrado de longitud uitaria (dominio computacional) con la curvas de frontera del dominio físico. A este mapeo se le conoce como la suma booleana de las transformaciones P_ξ y P_η, y se escribe $P_\xi \oplus P_\eta$, por lo tanto:

$$P_\xi \oplus P_\eta = P_\xi + P_\eta - P_\xi P_\eta \tag{4.14}$$

con lo que la fórmula final queda:

$$\begin{aligned}(P_\xi \oplus P_\eta)(\xi,\eta) =& P_\xi(\xi,\eta) + P_\eta(\xi,\eta) - P_\xi P_\eta(\xi,\eta) \\ =& (1-\xi)r(0,\eta) + \xi r(1,\eta) + (1-\eta)r(\xi,0) + \eta r(\xi,1) \\ & - (1-\xi)(1-\eta)r(0,0) - (1-\xi)\eta r(0,1) - (1-\eta)\xi r(1,0) \\ & - \xi\eta r(1,1)\end{aligned} \tag{4.15}$$

Esta ecuación es la base del métro de TFI para dos dimensiones. La malla se genera tomando valores discretos ξ_i, η_j para los ejes ξ y η.

Capítulo 5

Generación de mallas mediante EDP

La generación de mallas mediante estos métodos se ha esparcido gracias a a la versatilidad que poseen y la relativa facilidad con la que pueden ser aplicados. La idea sobre la cual se desarrolla el método, es la de obtener la solución númerica de una ecuación diferencial parcial, esta solución representa las coordenadas de la malla, la cual debe coincidir en su frontera interna, con la forma geométrica del cuerpo que se pretende analizar. [18]

Existen tres esquemas predominantes de generación de mallas mediante la solución de sistemas de ecuaciones diferenciales parciales:

- Ecuaciones Elípticas
- Ecuaciones Hiperbólicas
- Ecuaciones Parabólicas

5.1. Generación de mallas mediante EDP elípticas

Para la solución de un sistema de EDP elípticas se necesita cumplir la condición de tener definidas las condiciones de frontera a lo largo de todos los puntos pertenecientes a la misma, es decir, que se tengan condiciones de frontera de Dirichlet. La solución se obtiene mediante diferentes métodos numéricos iterativos, entre los que destacan el método de Jacobi, el método de Gauss-Seidel y métodos de sobre relajación.

La idea es que se calculen las coordenadas (ξ, η) y que la solución del sistema de ecuaciones genere las correspondientes coordenadas (x, y) del dominio físico, con un mapeo uno a uno, es decir, que exista una relación adecuada entre el dominio computacional y el lógico.

Considerando la ecuación de Laplace:

$$\xi_{xx} + \xi_{yy} = 0 \tag{5.1a}$$

$$\eta_{xx} + \eta_{yy} = 0 \tag{5.1b}$$

donde como ya se ha mencionado, ξ y η representan las coordenadas del dominio computacional y

además, son variables dependientes de x y de y. Si se quiere trabajar con la forma inversa, es decir, que ahora x y y sean variables dependientes de ξ y η, la ecuación de Laplace se expresa como:

$$\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} = 0 \tag{5.2a}$$

$$\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} = 0 \tag{5.2b}$$

donde:

$$\alpha = x_\eta^2 + y_\eta^2$$
$$\beta = x_\xi x_\eta + y_\xi y_\eta$$
$$\gamma = x_\xi^2 + y_\xi^2$$

La distribución de los nodos, obtenidos mediante la solución de la ecuación de Laplace (ecuación 5.1), tiende ser uniforme debido al efecto suavizante de dicha ecuación. Para poder manipular esta distribución, es necesario agregar funciones de forzado a la ecuación 5.1, dando como resultado la ecuación de Poisson:

$$\xi_{xx} + \xi_{yy} = P(\xi, \eta) \tag{5.3a}$$

$$\eta_{xx} + \eta_{yy} = Q(\xi, \eta) \tag{5.3b}$$

y transformando la ecuación, haciendo x y y las variables dependientes:

$$\alpha x_{\xi\xi} + \beta x_{\xi\eta} + \gamma x_{\eta\eta} = -I^2[P(\xi, \eta)x_\xi + Q(\xi, \eta)x_\eta] \tag{5.4a}$$

$$\alpha y_{\xi\xi} + \beta y_{\xi\eta} + \gamma y_{\eta\eta} = -I^2[P(\xi, \eta)y_\xi + Q(\xi, \eta)y_\eta] \tag{5.4b}$$

Las funciones P y Q se seleccionan dependiendo de la necesidad específica del problema que se analiza. Dichas necesidades pueden ser, por ejemplo, el agrupamiento de nodos alrededor de un punto especifico, o quizás sea la búsqueda de un sistema ortogonal en la superficie.

El uso de las funciones P y Q para manipular la densidad de puntos alrededor de ciertas zonas se logra atrayendo las líneas vecinas a una línea seleccionada. La misma funciona para los nodos, haciendo la atracción de los nodos colindantes hacie el nodo seleccionado. O bien, se puede llevar a cabo una combinación de ambos efectos.

Las ecuaciones para P y Q que llevan a cabo este proceso son:

$$\begin{aligned} P(\xi, \eta) = &- \sum_{m=1}^{M} a_m \frac{\xi - \xi_m}{|\xi - \xi_m|} \exp(-c_m|\xi - \xi_m|) \\ &- \sum_{n=1}^{N} b_n \frac{\xi - \xi_n}{|\xi - \xi_n|} \exp\{-d_n \left[(\xi - \xi_n)^2 + (\eta - \eta_n)^2\right]^{\frac{1}{2}}\} \end{aligned} \tag{5.5}$$

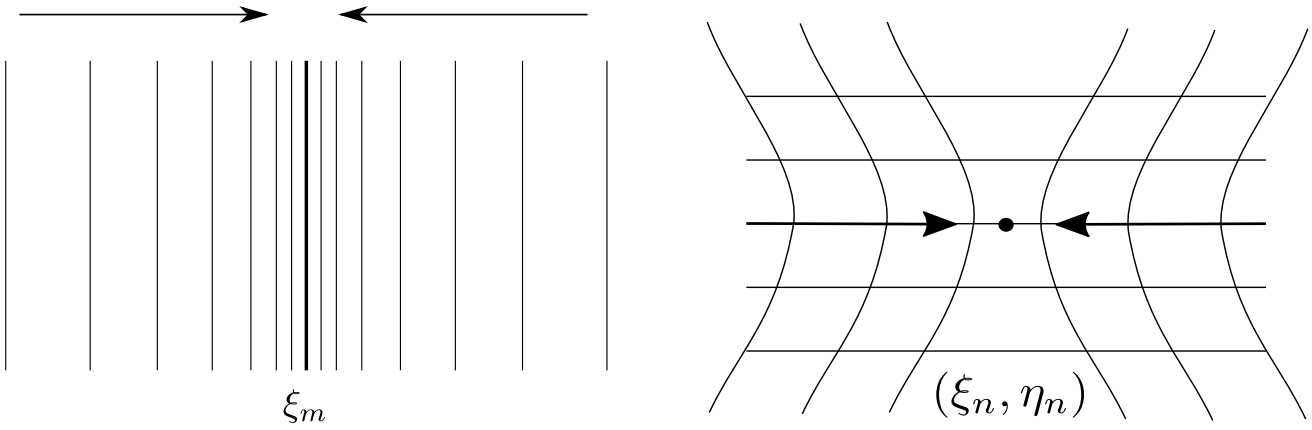
$$
\begin{aligned}
Q(\xi,\eta) = &- \sum_{m=1}^{M} a_m \frac{\eta-\eta_m}{|\eta-\eta_m|} \exp(-c_m|\eta-\eta_m|) \\
&- \sum_{n=1}^{N} b_n \frac{\eta-\eta_n}{|\eta-\eta_n|} \exp\{-d_n \left[(\xi-\xi_n)^2 + (\eta-\eta_n)^2\right]^{\frac{1}{2}}\}
\end{aligned}
\tag{5.6}
$$

estas ecuaciones fueron propuestas en 1974 por Thompson, Thames y Mastin [19], y son ampliamente referidas en diversos textos tanto de dinámica de fluidos computacional como de generación numérica de mallas.

En las ecuaciones 5.5 y 5.6 el valor de M es el número de líneas existentes en la malla, tanto líneas coordenadas $\xi = \xi_m$ como líneas $\eta = \eta_m$ y N representa el número de nodos ($\xi = \xi_n, \eta = \eta_n, 0 \leq \xi_n, \eta_n \leq 1$) a los que la malla se verá atraída. Los factores a_m y b_n son factores de amplificación, mientras que los factores c_m y d_n son factores de decaimiento,estos cuatro parámetros son datos de entrada para el código, asignados por el usuario, y los cuatro deben ser valores positivos.

El primer término en la ecuación para $P(\xi,\eta)$ tiene como efecto la atracción de las líneas ξ (líneas a lo largo de las cuales la coordenada en ξ permanece constante) hacia la línea $\xi = \xi_m$ en el dominio físico con una amplitud a_m, mientras que le segundo término atrae las líneas ξ hacia un punto determinado con una amplitud b_n. Estos parámetros tienen el mismo efecto en la ecuación de $Q(\xi,\eta)$, pero la atracción se ve reflejada en las líneas donde se mantiene constante la coordenada η.

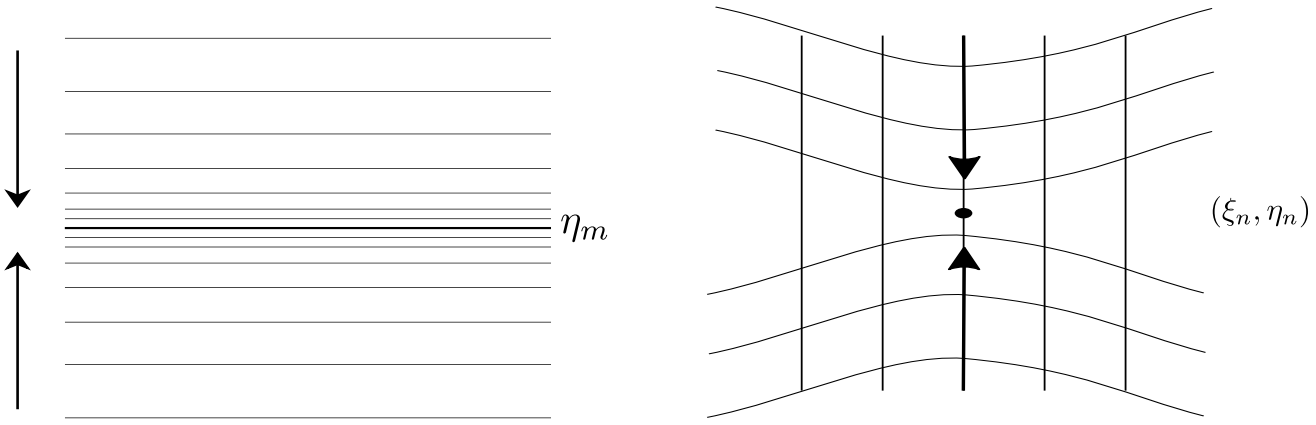
Las funciones $(\xi-\xi_m)/|\xi-\xi_m|$ y $(\eta-\eta_m)/|\eta-\eta_m|$ son funciones que solo pueden dar como resultado valores ± 1 y su propósito dentro de la fórmula es el garantizar que la atracción se dé, en caso de las lineas ξ y η por ambos lados de las mismas, y en todos los nodos vecinos para el caso de la atracción hacia un punto (ξ_n, η_n). Asignar un valor negativo a los factores de amplitud da como resultado un efecto contrario, es decir, se crea un efecto de repulsión.



(a) Efecto de atracción a la línea $\xi = \xi_m$

(b) Efecto de atracción al punto (ξ_n, η_n)

Figura 5.1: Efecto de atracción en el eje ξ por la función $P(\xi,\eta)$



(a) Efecto de atracción a la línea $\eta = \eta_m$

(b) Efecto de atracción al punto (ξ_n, η_n)

Figura 5.2: Efecto de atracción en el eje η por la función $Q(\xi, \eta)$

5.1.1. Algoritmos de solución

Existen en general, dos métodos de solción de sistemas de ecuaciones algebráicas lineales simultáneas: métodos directos y métodos iterativos.

Dentro de los métodos directos podemos encontrar los métodos de Cramer y de Gauss. La principal desventaja que presentan estos metodos es el gran número de opraciones aritméticas necesarias para obtener una solución del sistema. Desde el punto de vista computacional otras desventajas que presentan son el uso de memoria y cierta dificultad para ser programados. Los métodos iterativos son simples y fáciles de programar, por lo que se presentan como una mejor herramienta para llevar a cabo la solcuión del sistema. La idea de estos métodos, tal y como su nombre lo indica, es obtener la solución mediante un proceso iterativo, por lo general se asume una primera solución y con dichos valores propuestos se calculan nuevos valores de las incógnitas, con base en lo nuevos valores calculados se obtienen nuevos valores. Este proceso se repite hasta satisfacer el criterio de convergencia establecido.

5.1.2. Métodos iterativos: Jacobi, Gauss-Seidel, SOR

Para resolver una ecuación matricial de la forma $Au = d$ para el vector u de dimensiones $n \times 1$ suelen utilizarse métodos iterativos. Se pueden escribir las ecuaciones para cada término del vector u (asumiendo que ningún elemento de la diagonal principal de la matriz es igual a 0) de la siguiente manera

$$u_1 = \frac{1}{a_{11}} \left[d_1 - (a_{12}u_2 + a_{13}u_3 + \cdots + a_{1n}u_n) \right]$$

$$u_2 = \frac{1}{a_{22}} \left[d_2 - (a_{21}u_1 + a_{23}u_3 + \cdots + a_{2n}u_n) \right]$$

$$\cdots$$

$$u_n = \frac{1}{a_{nn}} \left[d_n - \left(a_{n1}u_1 + a_{n2}u_2 + \cdots + a_{n(n-1)}u_{n-1} \right) \right]$$

A partir de estas ecuaciones se derivan varios esquemas iterativos, dada una solución inicial $u_i^{(1)}, i = 1, 2, \ldots, n$. El primero de ellos es el método de Jacobi, en éste, la variable dependiente se calcula usando los datos previamente obtenidos, quedando sus ecuaciones cómo

$$u_1^{k+1} = \frac{1}{a_{11}} \left[d_1 - \left(a_{12}u_2^k + a_{13}u_3^k + \cdots + a_{1n}u_n^k \right) \right]$$

$$u_2^{k+1} = \frac{1}{a_{22}} \left[d_2 - \left(a_{21}u_1^k + a_{23}u_3^k + \cdots + a_{2n}u_n^k \right) \right]$$

$$\cdots$$

$$u_n^{k+1} = \frac{1}{a_{nn}} \left[d_n - \left(a_{n1}u_1^k + a_{n2}u_2^k + \cdots + a_{n(n-1)}u_{n-1}^k \right) \right] \tag{5.9}$$

La ecuación 5.9 es la ecuación constitutiva del método iterativo de Jacobi, en ella k representa los valores previamente calculados, es decir, obtenidos de la iteración anterior o de la solución inicial, según sea el caso. El superíndice $k + 1$ indica el valor obtenido en la iteración actual.

El siguiente método que puede desarrollarse es el método de Gauss-Seidel, el cual utiliza los valores calculados en la iteración actual para calcular los siguientes valores de la variable dependiente. Sus ecuaciones se escriben como

$$u_1^{k+1} = \frac{1}{a_{11}} \left[d_1 - \left(a_{12}u_2^k + a_{13}u_3^k + \cdots + a_{1n}u_n^k \right) \right]$$

$$u_2^{k+1} = \frac{1}{a_{22}} \left[d_2 - \left(a_{21}u_1^{k+1} + a_{23}u_3^k + \cdots + a_{2n}u_n^k \right) \right]$$

$$\cdots$$

$$u_n^{k+1} = \frac{1}{a_{nn}} \left[d_n - \left(a_{n1}u_1^{k+1} + a_{n2}u_2^{k+1} + \cdots + a_{n(n-1)}u_{n-1}^{k+1} \right) \right] \tag{5.11}$$

en este método se calcula un nuevo valor u_1^{k+1} con los valores de la iteración anterior, dicho valor luego es ocupado para el cálculo de u_2^{k+1}, ambos términos son utilizados para obtener u_3^{k+1} y el proceso continúa de la misma manera hasta llegar a la última ecuación. Es decir, el método de Gauss-Seidel implica el uso inmediato de los valores u_i^{k+1} tan pronto como están disponibles lo que da como resultado un uso menor de memoria en el ordenador, además de satisfacer el criterio de convergencia en menos iteraciones comparado con el método de Jacobi. La ecuación 5.11 puede escribirse de la siguiente manera

$$u_i^{k+1} = \frac{1}{a_{ii}} \left[d_i - \sum_{j=1}^{i-1} a_{ij}u_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}u_j^k \right], \quad i = 1, 2, \ldots, n \tag{5.12}$$

Si durante el proceso de solución se percibe una tendencia en los valores calculados se puede utilizar la dirección de cambio para extrapolar para la siguiente iteración y por lo tanto, acelerar el proceso de solución. A este procedimiento se le conoce como método SOR (Succesive Over-Relaxation o Sobre-relajación sucesiva). Este método introduce un parámetro extra ω conocido

como parámetro de aceleración, el cual puede acelerar la convergencia. Dicho método está representado por

$$\begin{aligned} u_i^{k+1} =& u_i^k + \frac{\omega}{a_{ii}} \left[d_i - \sum_{j=1}^{i-1} a_{ij} u_j^{k+1} - a_{ii} u_i^k - \sum_{j=i+1}^{n} a_{ij} u_j^k \right] \\ =& \frac{\omega}{a_{ii}} \left[d_i - \sum_{j=1}^{i-1} a_{ij} u_j^{k+1} - \sum_{j=i+1}^{n} a_{ij} u_j^k \right] + (1-\omega)\, u_i^k, \quad i = 1, 2, \ldots, n \end{aligned} \tag{5.13}$$

Cuando $\omega = 1$ el método SOR se convierte en el método de Gauss-Seidel. Si $1 < \omega < 2$ se está trabajando con sobre-relajación, mientras que al utilizar un valor $0 < \omega < 1$ se lleva a cabo una bajo-relajación.

5.2. Generación de mallas mediante EDP hiperbólicas

La generación de mallas mediante EDP elípticas puede ser demasiado costosa en términos de tiempo de cómputo, así como en el uso de memoria en el ordenador. Esto se debe a que dicho esquema intenta hacer coincidir un sistema de coordenadas curvilíneo a cuatro curvas de frontera, seis en mallas tridimensionales. De aquí surge un nuevo enfoque, en el cual solo se requiere iniciar la solución con una sola frontera e ir avanzando hacia afuera en el dominio físico usando EDP hiperbólicas. [17]

Con este propósito se toma en cuenta el trabajo publicado por Steger y Chausse [20] para la generación de mallas bidimensionales ortogonales y con control del área de celda. Se inicia el proceso con una frontera única en la que se considera que $\eta = 0$. Se propone tambien un sistema de ecuaciones de primer orden para las coordenadas x, y como funciones de ξ, η:

$$g_{12} = g_1 \cdot g_2 = x_\xi x_\eta + y_\xi y_\eta = 0 \tag{5.14a}$$

$$|g_1 \times g_2| = x_\xi y_\eta - x_\eta y_\xi = F \tag{5.14b}$$

La primera ecuación dota a la malla de ortogonalidad, mientras que la segunda tiene el término F como una medida del área de la celda (si el producto $\delta\xi\delta\eta$ es el mismo para cada celda). Se puede trabajar utilizando F como una función de ξ, η, lo que permitiría incrementar la densidad de la malla cerca de la frontera donde $\eta = 0$ haciendo que F tenga un valor pequeño en dicha zona. De las ecuaciones 5.14:

$$x_\eta = -\frac{F}{x_\xi^2 + y_\xi^2} y_\xi \tag{5.15a}$$

$$y_\eta = \frac{F}{x_\xi^2 + y_\xi^2} x_\xi \tag{5.15b}$$

Antes de proceder a la solución del sistema de las ecuaciones 5.14 se deben tomar en cuenta algunas consideraciones. En primera instancia debe tenerse en cuenta que el sistema es un sistema no lineal por lo que un proceso de linearización debe llevarse a cabo, una segunda consideración es que al ser un sistema de ecuaciones hiperbólicas se debe llevar a cabo el proceso de solución mediante un proceso de marcha, en este trabajo la marcha se realizará en la dirección del eje η. Tercera, para un sistema de ecuaciones hiperbólicas se debe especificar una condicion inicial así como un condición de frontera y por último, con el objetivo de evitar la presencia de oscilaciones, se puede agregar un término de"damping" al lado derecho de las ecuaciones.

Para el proceso de linearización, se utilizará el esquema iterativo de Newton, el cual especifica que un termino no lineal es aproximado mediante

$$AB = A^{k+1}B^k + B^{k+1}A^k - A^kB^k \tag{5.16}$$

donde el superíndice k representa el estado previo conocido. A partir de este punto el superíndice $k+1$, el cual representa el nivel actual a obtener, será eliminado de las ecuaciones, y se entiende que todos los términos sin superíndice pertenecen al nivel actual $k+1$. Por lo tanto, el sistema de ecuaciones hiperbólicas es expresado en su forma lineal como:

$$x_\xi x_\eta^k + x_\xi^k x_\eta - x_\xi^k x_\eta^k + y_\xi y_\eta^k + y_\xi^k y_\eta - y_\xi^k y_\eta^k = 0 \tag{5.17a}$$

$$x_\xi y_\eta^k + x_\xi^k y_\eta - x_\xi^k y_\eta^k - x_\eta y_\xi^k - x_\eta^k y_\xi + x_\eta^k y_\xi^k = F \tag{5.17b}$$

Aplicando el concepto de las ecuaciones 5.14 a las ecuaciones 5.17 podemos simplificar las últimas, quedando como

$$x_\xi x_\eta^k + x_\xi^k x_\eta + y_\xi y_\eta^k + y_\xi^k y_\eta = 0 \tag{5.18a}$$

$$x_\xi y_\eta^k + x_\xi^k y_\eta - x_\eta y_\xi^k - x_\eta^k y_\xi = F + F^k \tag{5.18b}$$

Este sistema de ecuaciones puede escribirse de forma compacta como

$$[A]\, R_\xi + [B]\, R_\eta = H \tag{5.19}$$

donde:

$$R = \begin{bmatrix} x \\ \\ y \end{bmatrix} \qquad A = \begin{bmatrix} x_\eta^k & y_\eta^k \\ \\ y_\eta^k & -x_\eta^k \end{bmatrix} \qquad B = \begin{bmatrix} x_\xi^k & y_\xi^k \\ \\ -y_\xi^k & x_\xi^k \end{bmatrix} \qquad H = \begin{bmatrix} 0 \\ \\ F + F^k \end{bmatrix}$$

Por definición, el sistema de ecuaciones descrito por la ecuación 5.19 es hiperbólico los eigenvalores de $[B]^{-1}[A]$ son reales. Se observa que

$$[B]^{-1} = \frac{1}{DN} \begin{bmatrix} x_\xi^k & -y_\xi^k \\ \\ y_\xi^k & x_\xi^k \end{bmatrix}$$

por lo tanto

$$[C] = [B]^{-1}[A] = \begin{bmatrix} x_\xi^k x_\eta^k - y_\xi^k y_\eta^k & x_\xi^k y_\eta^k + x_\eta^k y_\xi^k \\ x_\xi^k y_\eta^k + x_\eta^k y_\xi^k & -\left(x_\xi^k x_\eta^k - y_\xi^k y_\eta^k\right) \end{bmatrix}$$

donde

$$DN = \left(x_\xi^k\right)^2 + \left(y_\xi^k\right)^2$$

Los eigenvalores de $[C]$ son

$$\lambda_{1,2} = \pm \left[\frac{\left(x_\eta^k\right)^2 + \left(y_\eta^k\right)^2}{DN} \right]$$

los cuales cumplen con la condición de ser siempre reales (para un sistema hiperbólico) siempre que

$$DN = \left(x_\xi^k\right)^2 + \left(y_\xi^k\right)^2 \neq 0$$

Para obtener un sistema de ecuaciones algebráico, debemos aplicar el método de diferencias finitas. Para la sustitución de las derivadas parciales respecto al eje ξ se utiliza una aproximación central de segundo orden y para el caso de las derivadas parciales respecto al eje η se hará uso de aproximaciones de primer orden tipo "backwards", resultando la ecuación en

$$[A]\frac{R_{i+1,j} - R_{i-1,j}}{2\Delta\xi} + [B]\frac{R_{i,j} - R_{i,j-1}}{\Delta\eta} = [H]_{i,j} \tag{5.20}$$

que al ser multiplicada por la matriz $[B]^{-1}$ y reacomodada queda cómo

$$\frac{R_{i,j} - R_{i,j-1}}{\Delta\eta} + [B]^{-1}[A]\frac{R_{i+1,j} - R_{i-1,j}}{2\Delta\xi} = [B]^{-1} H_{i,j} \tag{5.21}$$

en donde las matrices A y B son evaluadas en la línea $(j-1)$. Esta ecuación se reacomoda y se le asignan nuevos términos, con el propósito de compactar a la misma, dando como resultado

$$[AA]\, R_{i-1,j} + [BB]\, R_{i,j} + [CC]\, R_{i+1,j} = [DD]_{i,j} \tag{5.22}$$

donde

$$[AA] = -\frac{1}{2\Delta\xi}[C]_{i,j-1}$$

$$[BB] = \frac{1}{\Delta\eta}[I]$$

$$[CC] = \frac{1}{2\Delta\xi}[C]_{i,j-1}$$

$$[DD] = [B]^{-1}_{i,j-1} H_{i,j} + \frac{R_{i,j-1}}{\Delta\eta}$$

una vez que se ha desarrollado la ecuación 5.22 para todas las instancias existentes a lo largo de i para un mismo nivel j, se obtiene un sistema de bloque tridiagonal

$$\begin{bmatrix} [BB]_2 & [CC]_2 & & & & & \\ [AA]_3 & [BB]_3 & [CC]_3 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & [AA]_{m-2} & [BB]_{m-2} & [CC]_{m-2} \\ & & & & [AA]_{m-1} & [BB]_{m-1} \end{bmatrix} \begin{bmatrix} R_2 \\ R_3 \\ \vdots \\ \vdots \\ R_{m-2} \\ R_{m-1} \end{bmatrix} = \begin{bmatrix} [DD]_2 \\ [DD]_3 \\ \vdots \\ \vdots \\ [DD]_{m-2} \\ [DD]_{m-1} \end{bmatrix} \tag{5.24}$$

los términos $[DD]_2$ y $[DD]_{m-1}$ son afectados por las condiciones de frontera de la siguiente manera

$$[DD]_2 = [DD]_2 - [AA]_2 R_1$$
$$[DD]_{m-1} = [DD]_{m-1} - [CC]_{m-1} R_m$$

Este sistema de ecuaciones se resuelve avanzando en dirección del eje η, siempre y cuando la distribución de punto en la superficie y en las fronteras sea proporcionada.

Los puntos en las fronteras deben ser libres de flotar, es decir, deben calcularse al final de cada iteración. Esto se logra aplicando la condición de ortogonalidad a las fronteras $i = 1$ e $i = m$ para actualizar dichos puntos.

5.2.1. Algoritmo de solución

En el presente proyecto se lleva a cabo la solución del sistema obtenido en la sección anterior mediante el método propuesto por Hoffman [21].

Un sistema de ecuaciones diferenciales parciales es aproximado mediante un sistema de bloque tridiagonal cuando en dicho sistema se involucran tres puntos de la malla en cada nivel. El sistema resultante puede expresarse en su forma general como:

$$S\Delta Q = R \tag{5.26}$$

donde ΔQ y R con vectores con m componentes, y a su vez, S representa el bloque tridiagonal

$$S = \begin{bmatrix} B_2 & C_2 & & & & \\ A_3 & B_2 & C_3 & & & \\ & A_4 & B_4 & C_4 & & \\ & & \ddots & \ddots & \ddots & \\ & & & A_{m-2} & B_{m-2} & C_{m-2} \\ & & & & A_{m-1} & B_{m-1} \end{bmatrix}$$

donde A_i, B_i y C_i son matrices de orden m.

Para poder continuar con la obtención de un esquema de solución, habrá que considerar la siguiente factorización

$$S = LU = \begin{bmatrix} \alpha_2 & & & & & \\ A_3 & \alpha_3 & & & & \\ & A_4 & \alpha_4 & & & \\ & & \ddots & \ddots & & \\ & & & A_{m-2} & \alpha_{m-2} & \\ & & & & A_{m-1} & \alpha_{m-1} \end{bmatrix} \begin{bmatrix} I & \beta_2 & & & & \\ & I & \beta_3 & & & \\ & & I & \beta_4 & & \\ & & & \ddots & \ddots & \\ & & & & I & \beta_{m-2} \\ & & & & & I \end{bmatrix} \tag{5.27}$$

donde I es la matriz identidad de orden m y las matrices cuadradas α_i y β_i se determinan mediante

$$\alpha_2 = B_2 \qquad y \qquad \beta_2 = B_2^{-1} C_2 \tag{5.28}$$

$$\alpha_i = B_i - A_i\beta_{i-1} \qquad i = 3, 4, \ldots, m-1 \tag{5.29}$$

y

$$\beta_i = \alpha_i^{-1} C_i \qquad i = 3, 4, \ldots, m-2 \tag{5.30}$$

Ahora, el sistema dado por la ecuación 5.26 es equivalente a

$$LY = R \tag{5.31}$$

donde

$$Y = U\Delta Q \tag{5.32}$$

La ecuación 5.31 al desarrollarse queda de la siguiente manera

$$\begin{bmatrix} \alpha_2 & & & & \\ A_3 & \alpha_3 & & & \\ & A_4 & \alpha_4 & & \\ & & \ddots & \ddots & \\ & & & A_{m-1} & \alpha_{m-1} \end{bmatrix} \begin{bmatrix} Y_2 \\ Y_3 \\ Y_4 \\ \vdots \\ Y_{m-1} \end{bmatrix} = \begin{bmatrix} R_2 \\ R_3 \\ R_4 \\ \vdots \\ R_{m-1} \end{bmatrix} \tag{5.33}$$

donde

$$Y_2 = \alpha_2^{-1} R_2 \tag{5.34}$$

y

$$Y_i = \alpha_i^{-1} \left(R_i - A_i Y_{i-1} \right) \qquad i = 3, 4, \ldots, m-1 \tag{5.35}$$

La ecuación 5.32 se desarrolla como

$$\begin{bmatrix} I & \beta_2 & & & & \\ & I & \beta_3 & & & \\ & & I & \beta_4 & & \\ & & & \ddots & \ddots & \\ & & & & I & \beta_{m-2} \\ & & & & & I \end{bmatrix} \begin{bmatrix} \Delta Q_2 \\ \Delta Q_3 \\ \Delta Q_4 \\ \vdots \\ \Delta Q_{m-2} \\ \Delta Q_{m-1} \end{bmatrix} = \begin{bmatrix} Y_2 \\ Y_3 \\ Y_4 \\ \vdots \\ Y_{m-2} \\ Y_{m-1} \end{bmatrix} \tag{5.36}$$

en donde

$$\Delta Q_{m-1} = Y_{m-1} \tag{5.37}$$

y

$$\Delta Q_i = Y_i - \beta_i \Delta Q_{i+1} \qquad i = m-1, m-2, \ldots, 3, 2 \tag{5.38}$$

5.3. Generación de mallas mediante EDP parabólicas

A pesar de que el esquema elíptico (ecuaciones 5.2), desarrollado por Thompson, et. al. [19] ha sido el esquema de mayor utilización en la dinámica de fluidos computacional, se han desarrollado nuevos métodos, con la intención de obtener esquemas más rápidos y menos costosos en términos de capacidad de cómputo. La intención de generar mallas mediante la solución de ecuaciones hiperbólicas, desarrollado por Steger y Chausse [20] representa un intento de alcanzar dicho objetivo, entre sus ventajas podemos encontrar el hecho de que la solución se logra gracias a un procedimiento de marcha a comparación del esquema por ecuaciones de tipo elípticas en el cual la solución se obtiene a través de un proceso iterativo, por lo que el tiempo de ejecución en el esquema hiperbólico es del mismo orden que el de una iteración en el esquema elíptico. Sin embargo, el generar mallas mediante ecuaciones hiperbólicas trae consigo ciertas desventajas entre ellas está la propagación de eventuales discontinuidades que pueda haber en la frontera interna conforme la solución marcha de dentro hacia afuera, además en la mayoría de casos es necesario agrerar términos de viscosidad artificial ya que las soluciones tienden a ser inestables y por último, y debido a las características matemáticas de las ecuaciones diferenciales parciales hiperbólicas, no se es posible definir una frontera externa.

En este trabajo se presenta el esquema desarrollado por Nakamura [22], el cual también fue retomado por Siladic [23] y que propone un método de generación de mallas mediante la solución de ecuaciones diferenciales parciales parabólicas. Las mayores ventajas de uso de ecuaciones parabólicas son: la solución de éstas es a través de problemas con condiciones iniciales, por lo que al igual que las ecuaciones hiperbólicas presentan un algoritmo de solución de marcha, lo que significa un bajo tiempo de cómputo, una ventaja adicional es que las ecuaciones parabólicas presentan mucho del comportamiento matemático de las ecuaciones elípticas, particularmente el hecho de poseer un efecto de difusión lo cual resulta en que cualquier discontinuidad que se presentase en la frontera interna será suavizada conforme el algoritmo de solución marcha y por último, hay la capacidad de satisfacer las condiciones de frontera para la frontera externa.

Además de lo explicado arriba, la importancia de desarrollar un algoritmo con el uso de ecuaciones parabólicas radica en dos principales aspectos, primero, el tiempo de ejecución computacional requerido para lograr una solución es una pequeña fracción del tiempo demandado por un algoritmo de ecuaciones elípticas y segundo, el uso de memoria que se necesita también es sustancialmente menor que aquel requerido por un esquema de generación elíptico.

Las ecuacuones utilizadas son las siguientes

$$a(\xi,\eta)x_\eta = b(\xi,\eta)x_{\xi\xi} + c(\xi,\eta)V_x(\xi,\eta) + d(\xi,\eta) \tag{5.39a}$$

$$a(\xi,\eta)y_\eta = b(\xi,\eta)y_{\xi\xi} + c(\xi,\eta)V_y(\xi,\eta) + d(\xi,\eta) \tag{5.39b}$$

donde a, b y c pueden ser constantes o alguna función de (ξ,η), (x,y) representan las coordenadas en el dominio físico mientras que (ξ,η) representan las coordenadas en el plano computacional, por último V_x y V_y son términos fuente. En este caso, la frontera interna representa una condición inicial, mientras que la frontera externa representa una restricción sobre el comportamiento de la solución en la dirección j. Las ecuaciones 5.39 son discretizadas utilizando diferencias finitas, particularmente aproximaciones tipo "backwards" en la coordenada η y aproximaciones de tipo central en la coordenada ξ. Dadas unas condiciones inciales en x y y para $\eta = 0$, en este caso la

distribución de la nube de puntos que definen la frontera interna, se obtiene un sistema tridiagonal, el cual debe ser resuelto para cada incremento contemplado en η.

Capítulo 6

Desarrollo Práctico

Los códigos de este trabajo fueron implementados en lenguaje Python 3, utilizando el paradigma de programación orientada a objetos, en éste los datos son trabajados como objetos con atributos y métodos pertenecientes a una clase, y que pueden ser heredados y aplicados a objetos o instancias de alguna subclase. Un objeto es la representación en un programa de un concepto y contiene toda la información necesaria para abstraerlo, como son los datos que describen sus atributos y operaciones que pueden realizarse dobre los mismos. En otras palabras un objetos es la relación que existe entre un conjunto de variables y métodos.

Se dice que todos las instancias (objetos) de un mismo tipo, son pertenecientes a una misma clase. Una de las características más importantes de la programación orientada a objetos es la herencia, la cual permite la definición de nuevas clases a partir de una "superclase" ya existente, las cuales se conocen como "subclases". Una subclase hereda todo el comportamiento de su superclase, pero se puede introducir además características propias.

En el presente trabajo se implementaron dos superclases: "airfoil" y "mesh", cada una con sus respectivas subclases. La clase airfoil se desarrolló para la creación de perfiles alares a partir de una archivo de texto que contenga la nube de puntos del perfil deseado, como dato de entrada se requiere el nombre del archivo texto y la dimensión en metros de la cuerda. Esta clase tiene una subclase "NACA4" la cual genera perfiles de la serie de 4 dígitos de perfiles aerodinámicos de la NACA. La creación del perfil se hace a partir de la ecuación constitutiva de la serie. Como datos de entrada se requieren los valores de m (combudara máxima), p (posición de la combadura máxima), t (espesor máximo) y c (cuerda). La subclase NACA4 tiene dos métodos para la creación del perfil, el primero de ellos, el método "create_linear" crea la nube de puntos con una distribución equidistante de los puntos en dirección del eje x. El segundo método, llamado "create_sin"genera a partir de una función senoidal una nube de punto que permite que haya mayor densidad de nodos tanto en el borde de ataque como en el borde de salida del perfil.

Bibliografía

[1] TJ Chung. *Computational fluid dynamics*. Cambridge university press, segunda edition, 2010.

[2] Richard H Pletcher, John C Tannehill, and Dale Anderson. *Computational fluid mechanics and heat transfer*. CRC Press, tercera edition, 2012.

[3] Vladimir D Liseikin. *Grid generation methods*, volume 1. Springer, 1999.

[4] Jiri Blazek. *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, tercera edition, 2015.

[5] John David Anderson and J Wendt. *Computational fluid dynamics*, volume 206. Springer, tercera edition, 1995.

[6] Yunus A Cengel and John M Cimbala. *Mecánica de fluidos: Fundamentos y Aplicaciones*. McGraw Hill Higher Education, segunda edition, 2012.

[7] CFD Direct. `https://cfd.direct/openfoam/user-guide/introduction/`, 2017.

[8] XFLR5. `www.xflr5.com/`, 2017.

[9] Massachusetts Institute of Technology. `http://web.mit.edu/drela/Public/web/xfoil/`, 2001.

[10] XFLOW. `https://www.3ds.com/products-services/simulia/products/xflow/`, 2018.

[11] SU2. `https://su2code.github.io/`, 2018.

[12] Steven C. Chapra and Raymond P. Canale. *Numerical Methods for Engineers*. McGraw-Hill, New York, sixth edition, 2010.

[13] Joe F Thompson, Bharat K Soni, and Nigel P Weatherill. Handbook of grid generation. 1999.

[14] Joe F Thompson, Zahir UA Warsi, and C Wayne Mastin. *Numerical grid generation: foundations and applications*, volume 45. North-holland Amsterdam, 1985.

[15] Vladimir D Liseikin. *Grid generation methods*, volume 1. Springer, 1999.

[16] Abhishek Khare, Ashish Singh, and Kishor Nokam. Best practices in grid generation for cfd applications using hypermesh. *Computational Research Laboratories*, page 2, 2009.

[17] M Farrashkhalvat and JP Miles. *Basic Structured Grid Generation: With an introduction to unstructured grid generation*. Elsevier, 2003.

[18] Mato F Siladic. Numerical grid generation and potential airfoil analysis and design. Technical report, AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH, 1988.

[19] Joe F Thompson, Frank C Thames, and C Wayne Mastin. Automatic numerical generation of body-fitted curvilinear coordinate system for field containing any number of arbitrary two-dimensional bodies. *Journal of computational physics*, 15(3):299–319, 1974.

[20] Joseph L Steger and Denny S Chaussee. Generation of body-fitted coordinates using hyperbolic partial differential equations. *SIAM Journal on Scientific and Statistical Computing*, 1(4):431–437, 1980.

[21] Klaus A Hoffmann and Steve T Chiang. Computational fluid dynamics volume 1. *Engineering Education System, Wichita, Kan, USA*, 2000.

[22] S Nakamura. Marching grid generation using parabolic partial differential equations. Technical report, OHIO STATE UNIV COLUMBUS DEPT OF MECHANICAL ENGINEERING, 1982.

[23] Mato F Siladic. Numerical grid generation and potential airfoil analysis and design. Technical report, AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH, 1988.