Alejandro Gronskis

Departamento de Ingeniería Mecánica de la Facultad de Ingeniería de la Universidad de Buenos Aires

Unidad de Investigación de la Facultad de Ingeniería de la Universidad de la Marina Mercante

Alternativas de mallado en un código para el estudio de escurrimientos

1. Interés del desarrollo de un codigo de calculo "casero"

Hoy en día, códigos de cálculo como Fluent, Star-CD y CFX entre otros, pueden responder a practicamente todas las situaciones consideradas dentro de la mecánica de fluidos debido a su potencia en cuanto a la gestión de geometrías complejas, a su versatilidad y al número importante de modelos implementados. Sin embargo existen sectores en los cuales estos códigos comerciales resultan menos adecuados que ciertos códigos "caseros"'. Los códigos comerciales son, en el caso más favorable, de orden dos en tiempo y espacio. Así, para todos los cálculos que necesitan un alto grado de precisión estos se demuestran poco adecuados. En el caso, por ejemplo, en el que se desee simular flujo turbulento, la mayoría de estos códigos utiliza modelos RANS o en el mejor caso LES, lo cual puede no ser apto si se desean conservar ciertas características del flujo en estudio. Estos códigos de cálculo comerciales pueden ser ejecutados en máquinas de arquitectura paralela, donde generalmente muestran una performance pobre en términos de CPU mientras que los códigos "caseros" particularmente bien optimizados pueden ser netamente de mayor performance.

Para situaciones de cálculo más específicas, los códigos comerciales no responden tampoco de la forma esperada. Aunque a menudo se ofrezca la posibilidad de modificar o implementar nuevos modelos por medio de subrutinas, no siempre es posible modificar el código lo suficientemente en profundidad para que este satisfaga nuestras necesidades.

Estas cuestiones hacen que ciertas personas prefieran desarrollar su propia herramienta de cálculo y así adaptarla conforme a la manera más óptima a sus necesidades específicas. La perspectiva de tener el control completo de los recursos del código es por lo tanto una de las mayores razones que nos motivan a desarrollar nuestra propia herramienta. Como última observación (no menor), el elevado costo anual de una licencia de software comercial justifíca aún más nuestra iniciativa de desarrollar nuestro propio código.

2. Opciones de mallado

Las ecuaciones de conservación pueden ser escritas de diferentas formas, dependiendo del sistema coordenado y los vectores base utilizados. Por ejemplo, se pueden seleccionar sistemas coordenados cartesianos, cilíndricos, esféricos u ortogonales curvilíneos, los cuales pueden estar fijos o en movimiento.

Además se deben seleccionar las bases sobre las cuales los vectores y tensores van a ser definidos (fijos o variables, covariante o contravariante, etc.). Dependiendo de esta elección, el vector velocidad y el tensor de tensiones pueden ser expresados en términos de, por ejemplo, componentes cartesianos.

Para poder llevar a cabo una simulación para un caso de interés se debe primero elegir el modelo matemático a utilizar, esto es, el conjunto de ecuaciones diferenciales (o integro-diferenciales) en derivadas parciales y las condiciones de borde. Se debe elegir un modelo apropiado para el fenómeno que se pretende simular (flujo incompresible, invicido, tubulento, etc.). Luego de haber seleccionado

el método, se debe elegir un método de discretización adecuado, o sea, un método de aproximación de las ecuaciones diferenciales mediante un sistema de ecuaciones albegráicas para las variables de algún conjunto discreto de localizaciones en tiempo y espacio (método de diferencias finitas, volúmenes finitos, espectrales, etc.).

2.1. *Grilla numérica*

Las localizaciones discretas sobre las cuales las variables van a ser calculadas están definidas por la grilla numérica, siendo esta esencialmente una representación discreta del dominio geométrico en el cual el problema va a ser resuelto. Esto divide el dominio solución en un número finito de subdominios (elementos, volúmenes de control, etc).

Algunos tipos posibles de grilla para el mallado son:

Grilla estructurada o regular: Las grillas estructuradas o regulares consisten de familias de líneas de grilla con la propiedad de que los miembros de una familia no se cruzan entre ellas y cruzan a cada miembro de las otras familas solo una vez.

Esto permite que las líneas de un dado conjunto (*set*) puedan ser enumeradas de forma consecutiva. La posición de cualquier punto de grilla (o volumen de control) dentro del dominio es indentificado de forma única por un set de dos (en 2D) o de tres (en 3D) índices, por ejemplo (*i,j,k*).

Esta es la estructura de grilla más simple dado que es lógicamente equivalente a la grilla cartesiana. Cada punto tiene cuatro puntos vecinos en dos dimensiones y seis puntos vecinos en tres dimensiones. Uno de los índices de cada punto vecino *P* difiere en *+-1* del índice correspondiente a *P*. Un ejemplo de grilla estructurada ortogonal regular se muestra en la **Figura 1**. Su estructura simplifica la programación y la matriz del sistema de ecuaciones algebraico tiene una estructura regular, la cual puede ser aprovechada para desarrollar una solución técnica. Debido a esto existe una gran cantidad de resolvedores (sistemas de resolución) aplicables solo a grillas estructuradas. La desventaja de las grillas estructuradas es que pueden ser utilizadas solo para dominios geométricamente simples. La otra desventaja es que puede ser difícil controlar la distribución de puntos de la grilla dado que la concentración de puntos en una región, por razones de precisión, produce un espaciamiento pequeño innecesario en otras partes del dominio solución resultando esto en un gasto de recursos. Este inconveniente se hace más notorio aún en los problemas en tres dimensiones. Este tipo de grilla es utilizado generalmente para cuerpos con bordes pronunciados. La simulación de geometrias en el dominio tiene también el inconveniente de que para bordes curvos el mallado grueso produce una representación del cuerpo poco adecuada en la superficie, dado que dicha superficie se forma a partir de cuadrados o rectángulos correspondientes a las localizaciones discretas.

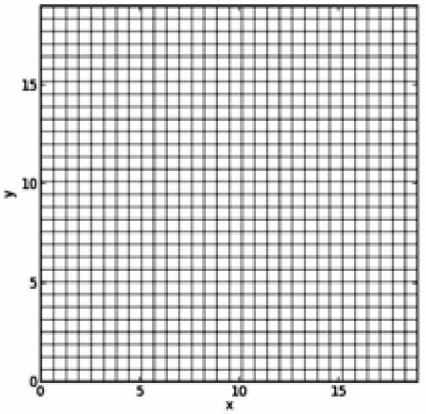

Figura 1. Grilla regular estructurada ortogonal.

Grilla estructurada en bloques: En una grilla estructurada en bloques hay dos o más niveles de subdivisión del dominio solución, estos son por ejemplo, el nivel basto o grosero (*coarse level*) y el nivel fino (dentro de cada bloque). En el nivel basto o grosero los bloques son segmentos relativamente grandes del dominio, su estructura puede ser irregular y pueden o no superponerse. En el nivel fino se define una grilla estructurada. Se requiere un tratamiento especial en las interfaces de los bloques. En la **Figura 2** se muestra un ejemplo de una grilla estructurada de tres bloques utilizada para el cálculo de flujo alrededor de un cilíndro. En este caso los puntos de las interfaces son coincidentes y no existe superposición. En la **Figura 3** se muestra una grilla en bloques estructurada donde los puntos de las interfaces no son coincidentes y tampoco existe superposición. Esta grilla es utilizada para calcular el flujo alrededor de un perfil alar. La programación en este caso es más dificultosa comparada con los casos mencionados anteriormente. Se pueden aplicar resolvedores para grillas estructuradas para cada bloque y así, con estas grillas, se pueden tratar dominios para flujos complejos. Es posible realizar un refinamiento local por bloque, lo cual es una ventaja destacable. Las grillas estructuradas en bloques con superposición son llamadas a veces *grillas composite*. Un ejemplo de tales grillas es el de la **Figura 4**. En la región de superposición las condiciones de borde para un bloque se obtienen interpolando la solución desde el otro bloque superpuesto. La desventaja de estas grillas es que la conservación no se logra fácilmente en los bordes de los bloques. Otra ventaja de este enfoque es que los dominios complejos se pueden tratar de manera más sencilla que para otros y ser utilizados para cuerpos en movimiento; por ejemplo, un bloque está unido al cuerpo y lo representa, moviendose con este mientras que una grilla estacionaria cubre los alrededores.

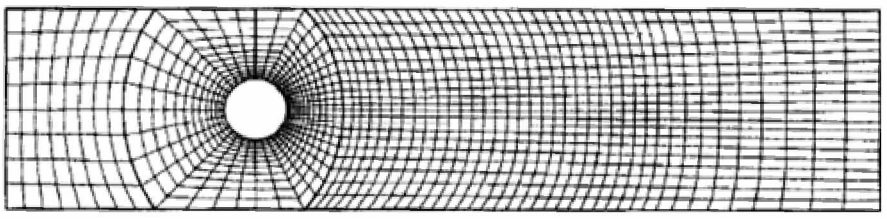

Figura 2. Grilla en bloques con puntos de interfaz coincidentes y sin superposición.

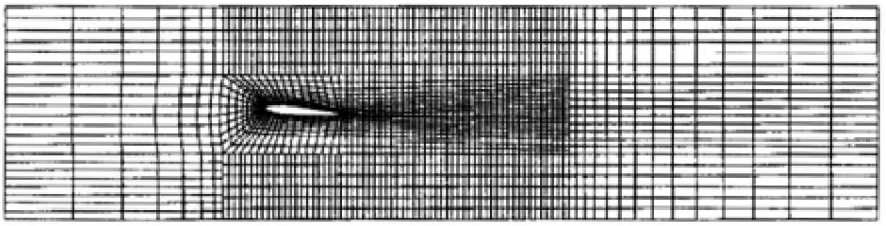

Figura 3. Grilla en bloques con puntos de interfaz no coincidentes y sin superposición.

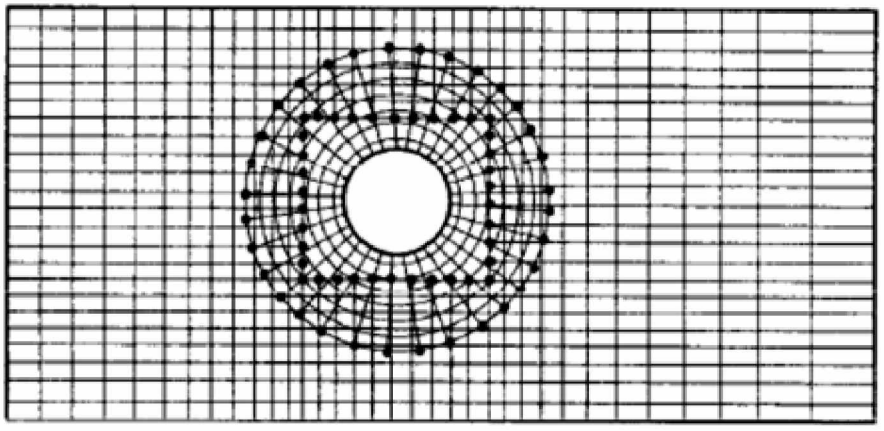

Figura 4. Grilla en bloques con puntos de interfaz no coincidentes y con superposición.

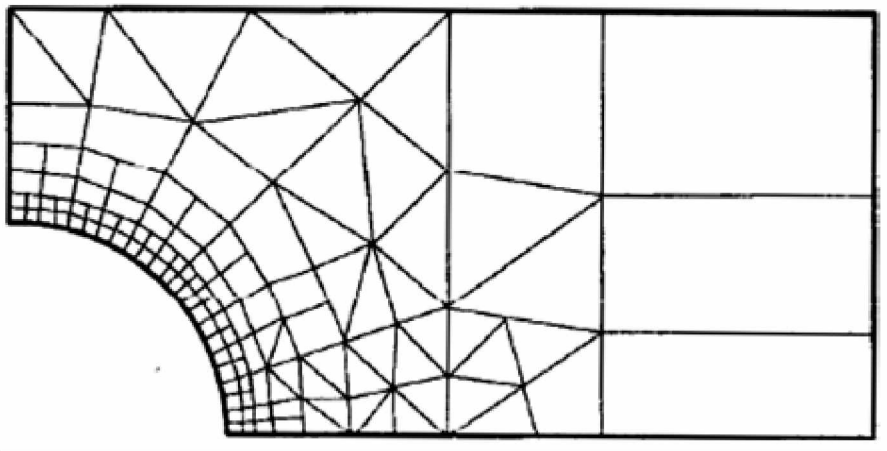

Figura 5. Grilla no estructurada.

Grillas no estructuradas: Para geometrías muy complejas, el tipo más flexible de grilla es el que se puede adaptar a una condición arbitraria de dominio para hallar la solución. En principio, tales grillas podrían ser utilizadas con cualquier esquema de discretización pero se adaptan mejor a enfoques de elementos o volúmenes finitos. Los elementos o volúmenes de control pueden ser de cualquier forma dado que no existe restricción alguna al número de elementos o nodos vecinos. En la práctica, las grillas más utilizadas son las formadas a partir de triángulos o cuadriláteros para el caso de dos dimensiones, y tetraedros o hexaedros para el caso de tres dimensiones. Tales grillas pueden ser generadas automáticamente con algoritmos ya existentes. La grilla se puede hacer ortogonal, controlar así fácilmente el radio de aspecto, y refinar localmente, con lo cual la flexibilidad es una ventaja destacable. Como desventaja se tiene la irregularidad de la estructura de datos. Las localizaciones de los nodos y sus conexiones vecinas necesitan ser especificadas explícitamente. La matriz del sistema de ecuaciones algebraico no presenta una estructura regular diagonal. El ancho de banda debe entonces ser reducido reordenando los puntos. Los resolvedores para los sistemas de ecuaciones algebraicos son usualmente más lentos que aquellos utilizados para grillas regulares. Las grillas no estructuradas son a menudo utilizadas con métodos de elementos finitos y volúmenes finitos con lo que, sumado a una relación de aspecto grande, la precisión en el cálculo de las derivadas se verá disminuida en comparación con el método de diferencias finitas. Los códigos computacionales para grillas no estructuradas son más flexibles dado que no necesitan ser cambiados cuando la grilla es localmente refinada o cuando se utilizan elementos

o volúmenes de control de formas distintas. Sin embargo, la generación de la grilla y el preprocesamiento son usualmente dificultosos e involucran un alto costo computacional para el caso de un cuerpo en movimiento. Un ejemplo de grilla no estructurada se muestra en la **Figura 5**.

3. Métodos de aproximación

Siguendo la elección de tipo de grilla se deben seleccionar las aproximaciones que serán utilizadas en el proceso de discretización. En un método de diferencias finitas se deben seleccionar las aproximaciones para las derivadas en los puntos de la grilla. En el método de volúmenes finitos se deben seleccionar lo métodos de aproximación de las integrales de superficie y volumen. En un método de elementos finitos se deben elegir las funciones de forma (elementos) y las funciones de peso. En los métodos espectrales la aproximación de la función hace uso de una serie truncada de funciones base ortogonales; por ejemplo, las series de Fourier son utilizadas para problemas periódicos, mientras que para problemas de valores en la frontera se utilizan los polinomios de Chebyshev o Legendre como funciones base.

Existen muchas posibilidades de elección; la misma influencia la precisión de la aproximación. Esto además afecta la dificultad de desarrollar el método de solución, su programación, su depuración, y la velocidad del código. Aproximaciones más precisas requieren el uso de más nodos y en algunos casos matrices más densas. El incremento de los requerimientos de memoria puede precisar del uso de grillas gruesas, afectando esto a la ventaja de la alta precisión. Por estas razones se debe hacer un compromiso entre simplicidad, facilidad de implementación, precisión y eficiencia computacional.

4. Eficiencia computacional

La misma aproximación puede ser muy precisa en una parte del flujo pero imprecisa en otra. El espaciamiento uniforme (tanto espacial como temporal) es raramente óptimo, dado que el flujo puede variar de forma muy pronunciada localmente tanto en el espacio como en el tiempo. Donde los cambios en las variables son pequeños, los errores serán tambien pequeños. Por lo tanto, con el mismo numero de elementos discretos y las mismas aproximaciones, los errores en los resultados pueden diferir en un orden de magnitud o más. Dado que el esfuerzo computacional es proporcional al número de elementos discretos, su propia distribución y tamaño son esenciales para la eficiencia computacional (o sea, el costo de alcanzar la precisión deseada).

5. El código a utilizar: *Incompact3d*

Si se busca alta precisión y rapidez de cálculo, entonces se emplean métodos espectrales. Estos métodos, que operan sobre las ecuaciones transformadas al espacio espectral, se limitan sin embargo a flujos de interés académico. En cambio, si se busca estudiar flujos que involucren geometrías complejas en detrimento de la precisión de los resultados y el costo computacional, entonces se recurre a algunos de los métodos mencionados anteriormente, que hacen uso de una malla no estructurada la cual se ajusta perfectamente al contorno de la geometría considerada. Estos métodos, que emplean elementos finitos y/o vol úmenes finitos, son muy difundidos en el ámbito industrial siendo ampliamente utilizados en códigos comerciales; sin embargo, como ya se dijo, su utilización presenta tres inconvenientes: el tiempo de cálculo para generar la malla, la complejidad del código a desarrollar y la distorsión de la malla la cual podría tener un efecto desastroso sobre la precisión del cálculo. Además, como se explicó antes, estos códigos utilizan modelos de turbulencia (RANS o LES) y no realizan DNS. El código de cálculo adoptado en el presente trabajo, ***Incompact3d***, se ubica en un lugar intermedio entre las dos aproximaciones descriptas anteriormente. Este código se basa en esquemas numéricos de alta precisión sobre una malla cartesiana. El empleo de tal malla no es incompatible con el estudio de flujos que involucren geometrías complejas, gracias al método de forzado (*immersed boundary method*), el cual emplea nuestro código para modelizar el obstáculo. La intención es poder competir en términos de precisión con los métodos espectrales y al mismo tiempo ser capaz de tratar geometrías complejas en forma simple y con un costo de cálculo lo menor posible.

Tradicionalmente, para la descripción de geometrias complejas, se realizan las discretizaciones de las ecuaciones gobernantes sobre mallas curvilíneas, estructuradas o no estructuradas, ajustadas al cuerpo (*body-fitted curvilinear grids*), haciendo que las fronteras geométricas del cuerpo inmerso coincidan con aquellas del dominio computacional. Un abordaje alternativo a este problema, que utiliza los esquemas cartesianos es la utilización del Método de Fronteras Sumergidas (*immerserd boundary method*). En este método, la presencia de cuerpos inmersos en el dominio debe ser considerada de otra forma. En el método de fronteras sumergidas se especifíca un término de fuerza de campo de modo de simular la presencia de una frontera inmersa en el escurrimiento sin alterar la malla cartesiana. La ventaja de este método es que cuerpos de formas completamentes arbitrarias pueder ser adicionados sin una reconstrucción de la malla.

El código *Incompact3d* permite la resolución de las ecuaciones de

Navier-Stokes para el caso de un fluido incompresible, siendo posibles dos tipos de aproximaciones numéricas para el cálculo de flujos: DNS o LES. Durante los últimos años este código ha sido validado para diferentes tipos de escurrimientos: chorro circular (Lardeau-tesis), control de chorros (Lardeau et al., 2002), capas de mezcla (Lardeau et al. 2001), flujos alrededor de un cilindro (Lamballais et al, 2002), (Silvestrini, 2003), (Parnaudeau et al., 2008), (Ribeiro and Vitola, 2006).

Incompact3d se vale de la discretización sobre malla cartesiana utilizando el método de diferencias finitas. El código tiene la capacidad de poder realizar una densificación de la grilla hacia el centro o hacia los costados del dominio en la dirección transversal al flujo. Si tenemos en cuenta la definición antes dada para la eficiencia computacional, en la cual haciamos incapié en que hay ciertas regiones donde el flujo varía más bruscamente que en otras, y por tanto es donde se requiere una malla más fina, entonces se puede presuponer que si realizamos la densificación en la zona donde se producirá dicha variación pronunciada podremos así tratar estos problemas utilizando una menor cantidad de puntos de grilla cartesiana para estudiar un problema que requeriria más puntos para su correcto tratamiento. De esta manera nuestro objetivo al densificar, es en parte aumentar la eficiencia computacional (aunque por supuesto, habrá que analizar también las cuestiones de memoria y tiempo de cálculo al realizar dicha densificación).

En un estudio proximo futuro nos propondremos realizar la densificación de la grilla en la otra dirección para un flujo bidimensional (o sea, la dirección del escurrimiento) y luego procederemos a realizar un estudio del desempeño del código utilizando la implementación ya disponible (ver Laizet, 2009) y a continuación nuestra implementación, efectuando además comparaciones con el caso de la grilla no densificada.

Entonces, como resultado del estudio mencionado tendremos un código que será capaz de realizar la densificación en la dirección transversal al escurrimiento, en la dirección del escurrimiento y en ambas direcciones al mismo tiempo.

6. Conclusiones

Hemos realizado una breve descripción de las grillas más comunmente utilizadas para discretizar el dominio a fin de llevar a cabo las simulaciones en un código CFD; asimismo describimos los métodos a menudo utilizados para estas estructuras de grilla, y podemos concluir que:

- Es de interés desarrollar un código "casero" acorde a nuestras necesidades para ciertos casos específicos.
- Es posible disponer de un código intermedio entre los códigos espectrales y de diferencias y/o volúmenes finitos considerando precisión y flexibilidad.
- Es de interés procurar llevar a cabo una densificación de la grilla para estudiar ciertos escurrimientos que presenten una gran variación local, en aras de aumentar así la eficiencia del código disminuyendo el esfuerzo computacional y la memoria de cálculo requerida.

Referencias

Laizet S. Developpement d'un code de calcul combinant des schemas de haute precision avec une methode de frontiere immergee pour la simulation des mouvements turbillonnaires en aval d'un bord de fuite. PhD thesis, Universite de Poitiers, 2005.

Laizet S. & Lamballais E. High-order compact schemes for incompressible flows: A simple and efficient method with quasi-spectral accuracy. J. Comp. Phys. 228: 5989-6015, 2009.

Laizet S. & Li N. Incompact3d: a powerful tool to tackle turbulence problems with up to 10^5 computational cores. Int. J. of Numerical Methods in Fluids 67 (11): 1735-1757, 2011.

Incompact3d. Web site de Incompact3d https://code.google.com/p/incompact3d/ y http://www3.imperial.ac.uk/tmfc/people/sylvainlaizet/incompact3d, Imperial College London.

Parnaudeau P. 2004 Etude numerique d'un ecoulement cisaille turbulent complexe a basse vitesse: Application a la protection rapprochee. PhD thesis, Universite de Poitiers.

Lardeau S. 2001 Simulation numerique directe du controle d'ecoulements cisailles libres par injection de fluide. PhD thesis, Universite de Poitiers.

Lardeau S., Lamballais E. and Bonnet J. P. 2002 Direct numerical simulation of a jet controlled by fluid injection. J. Turbulence 3, 002.

Lardeau S., Lamballais E. and Bonnet J. P. 2001 Control of a mixing layer flow by fluid injection: direct numerical simulation results. 2nd international Symposium on Turbulence and Shear Flow Phenomena.

Lamballais E. and Silvestrini J. 2002 Direct numerical simulation of interactions between a mixing layer and a wake around a cylinder. J. Turbul. 3, 028.

Silvestrini J. and Lamballais E. 2003 Direct numerical simulation of oblique vortex shedding from a cylinder in shear flow. Proc. 3rd International Symposium on Turbulence and Shear Flow Phenomena (Sendai, Japan).

Vitola M. 2006 Influencia de um contorno plano sobre o desprendimento de vortices ao redor de um cilindro circular. PhD thesis, Instituto de Pesquisas Hidraulicas - IPH/UFRGS, Brazil.