UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Simulación mediante DNS de un canal turbulento rectangular aplicando diferencias finitas compactas.

Trabajo Fin de Máster

Máster Universitario en Ingeniería Aeronáutica

AUTOR/A: Yebra Gómez, Aitor

Tutor/a: Hoyas Calvo, Sergio

CURSO ACADÉMICO: 2021/2022

Escuela Técnica Superior de Ingeniería del Diseño

IUMPA
Institut Universitari de Matemàtica Pura i Aplicada

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Simulación mediante DNS de un canal rectangular aplicando diferencias finitas compactas

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Trabajo Final de Máster

Máster Universitario en Ingeniería Aeronáutica

Curso 2021/2022

Autor: Aitor Yebra Gómez ayebgom@etsid.upv.es

Tutor: Sergio Hoyas Calvo serhocal@gmail.com

Valencia, Julio de 2022

Resumen

En este trabajo se ha extendido el código de canales turbulentos LISO para permitir la simulación flujos turbulentos en tuberías rectangulares (rectangular ducts). Este problema es de gran interés práctico, ya que una gran cantidad de sistemas de acondicionamiento de aire usan este tipo de tubería.

La implementación se ha basado en el uso de diferencias finitas compactas (DFC) de dimensión dos. Las DFC se inventaron, precisamente, para la simulación de flujos turbulentos de interés aeronáutico. Para realizar la implementación ha sido necesario analizar cada una de las 10000 líneas que componen el código LISO, teniendo implementada una versión en paralelo que ha sido probada en miles de procesadores.

En este TFM se presentan las estrategias de modelización, las técnicas de cálculo paralelo así como las mejoras implementadas tanto en la física del código como en su tratamiento de la memoria.

Palabras clave: DNS, turbulencia, CFD, LISO, conductos cuadrados.

Resum

En aquest treball s'ha estés el codi de canals turbulents LISO per a permetre la simulació fluxos turbulents en canonades rectangulars (rectangular ducts). Aquest problema és de gran interés pràctic, ja que una gran quantitat de sistemes de condicionament d'aire usen aquest tipus de canonada.

La implementació s'ha basat en l'ús de diferències finites compactes (DFC) de dimensió dues. Les DFC es van inventar, precisament, per a la simulació de fluxos turbulents d'interés aeronàutic. Per a realitzar la implementació ha sigut necessari analitzar cadascuna de les 10000 línies que componen el codi LISO, tenint implementada una versió en paral·lel que ha sigut provada en milers de processadors.

En aquest TFM es presenten les estratègies de modelització, les tècniques de càlcul paral·lel així com les millores implementades tant en la física del codi com en el seu tractament de la memòria.

Paraules clau: DNS, turbulència, CFD, LISO, conductes quadrats.

Abstract

In this project the LISO turbulent channel code has been extended to allow the simulation of turbulent flows in rectangular ducts. This problem is of great practical interest, since a large number of air conditioning systems use this type of pipe.

The implementation has been based on the use of compact finite differences (CFD) of dimension two. CFDs were invented precisely for the simulation of turbulent flows of aeronautical interest. In order to carry out the implementation it has been necessary to analyse each of the 10000 lines that make up the LISO code, having implemented a parallel version that has been tested on thousands of processors.

This TFM presents the modelling strategies, the parallel calculation techniques as well as the improvements implemented both in the physics of the code and in its memory processing.

Keywords: DNS, turbulence, CFD, LISO, square ducts.

MEMORIA

Índice

Índice de figuras

Índice de tablas

1. Introducción

En el presente trabajo se ha extendido el código de canales turbulentos LISO, para permitir la simulación de flujos turbulentos en tuberías rectangulares, siendo este un problema de gran interés práctico. Para ello se han empleado tanto conceptos de fluido dinámica como de supercomputación y computación paralela, apoyándose en trabajos previos como el realizado por Jesús Amo-Navarro [1] o el realizado por Federico Lluesma Rodríguez [2].

A continuación se hará una introducción a la física del problema, así como la evolución histórica de la mecánica de fluidos computacional (*CFD*).

1.1. Física de los fluidos

En primer lugar, es necesario dar la definición de fluido, de forma que enmarquemos correctamente el campo de estudio.

Un fluido es cualquier líquido o gas o, en general, cualquier material que no puede sostener una fuerza tangencial, o de cizallamiento, cuando está en reposo y que experimenta un cambio continuo de forma cuando se somete a dicha tensión. Este cambio continuo e irrecuperable de la posición de una parte del material con respecto a otra cuando se somete a un esfuerzo de cizallamiento constituye el flujo, una propiedad característica de los fluidos. Dentro de los posibles estados de la materia, son tres los que se identifican con esta definición de fluido: líquido, gas y plasma.

A partir de esto, la mecánica de fluidos se define como la rama de la física encargada del estudio del comportamiento de los fluidos, tanto del estudio de su movimiento como de las fuerzas aplicadas al mismo. Puede dividirse en estática de fluidos, el estudio de los fluidos en reposo, y dinámica de fluidos, el estudio del efecto de las fuerzas en el movimiento de los fluidos. Cuando el fluido se encuentra en movimiento se denomina flujo, siendo este de gran interés para la mecánica de fluidos.

Los flujos se pueden clasificar en dos grandes grupos, siendo estos laminares o turbulentos

- **Flujo laminar:** El flujo laminar se caracteriza porque las partículas del

fluido siguen trayectorias suaves en capas, con cada capa moviéndose suavemente respecto a las capas adyacentes con poca o ninguna mezcla a bajas velocidades. El movimiento de las partículas del fluido es muy ordenado y las partículas cercanas a una superficie sólida se mueven en líneas rectas paralelas a esa superficie. Existen ciertas condiciones bajo las cuales el flujo laminar podría transicionar a flujo turbulento, perdiendo las características anteriormente citadas.

- **Flujo turbulento:** En el flujo turbulento, el fluido experimenta fluctuaciones irregulares o mezcla. La velocidad del fluido en un punto experimenta continuamente cambios tanto de magnitud como de dirección.

La mayoría de los fluidos que observamos en el día a día entran dentro de la clasificación de flujos turbulentos, es por ello que este tipo de flujos ha sido de gran interés históricamente, siendo campos de estudio para grandes investigadores como Da Vinci o Isaac Newton, llegando finalmente a las ecuaciones desarrolladas por ingeniero y físico francés Claude-Louis Navier y al físico y matemático anglo irlandés George Gabriel Stokes, denominadas ecuaciones de Navier-Stokes.

Las ecuaciones de Navier-Sokes son ecuaciones en derivadas parciales (PDE) que describen y modelizan el comportamiento de los fluidos viscosos. Obtenidas a partir de las ecuaciones para fluidos no viscosos desarrolladas por Leonhard Euler en el siglo XVIII. Matemáticamente, las ecuaciones de Navier-Stokes representan la conservación de masa y momento para fluidos Newtonianos, no siendo posible presentar una solución analítica cerrada a las mismas, por lo que es necesario someterlas a distintas simplificaciones, dependiendo estas del caso a estudiar, que permitan realizar una aproximación analítica.

El siguiente gran avance en la mecánica de fluidos fue el presentado a finales del siglo XIX por el científico británico Osborne Reynolds, el cual realizó distintos estudios sobre el comportamiento de un flujo en el interior de un canal y el paso del mismo de régimen laminar a turbulento. A raíz de estos estudios se desarrolló el denominado como *número de Reynolds*(1,1), número adimensional que permite estudiar y parametrizar los cambios de régimen del flujo. Este parámetro se define como la relación entre las fuerzas inerciales y las fuerzas viscosas presentes en el fluido.

$$Re = \frac{\rho u L}{\mu} \tag{1,1}$$

donde ρ es la densidad del fluido, u es la velocidad, L es una longitud carac-

terística y μ es la viscosidad cinemática.

Finalmente, los últimos grandes avances en la mecánica de fluidos fueron los presentados a mediados del siglo XX por el matemático Andréi Kolmogórov, el cual define la turbulencia como un fenómeno multiescala donde los torbellinos (o *eddies*) de mayor tamaño son inestables, llegando a romperse eventualmente y formando torbellinos de menor tamaño, a los cuales se les proporciona la energía cinética del torbellino mayor. Esto ocurre de forma sucesiva hasta llegar a un punto en el cual la disipación viscosa es capaz de disipar la energía cinética de los torbellinos más pequeños.

La turbulencia, como ya se comentó anteriormente, está presente en la gran mayoría de los procesos que tienen lugar en el mundo cuando se trabaja con fluidos. Ya sea en el caso del transporte de fluidos como pueden ser canales de agua, gasoductos u oleoductos, o en el caso de aviones, barcos, coches o cualquier medio de transporte en general la turbulencia juega un papel primario, siendo la causante de gran parte de las pérdidas de energía originadas. Estas pérdidas se estiman en torno a un 5 % de la energía total consumida por la humanidad, usada simplemente para vencer la resistencia que impone la turbulencia [3] [4]. Es por ello que el correcto entendimiento de la turbulencia es necesario si se pretenden realizar mejoras hacia la eficiencia energética.

Es necesario mencionar que no todos los efectos de la turbulencia son negativos. En el caso del equilibrio térmico o de mezcla de flujos la turbulencia es una importancia mayúscula, siendo ella la causante de que sea posible realizar procesos de mezcla como los presentados en los motores de combustión. En el caso de que trabajase únicamente con flujo laminar el tiempo necesario para realizar estos procesos ascendería de tal manera que sería implanteable su realización. Es por esta dualidad por lo que es especialmente interesante entender el flujo turbulento, pudiendo eliminarlo o disminuirlo cuando sea necesario y aumentarlo o genéralo cuando cuando así se precise.

Los métodos clásicos usados para realizar los estudios de la mecánica de fluidos son los métodos teóricos y experimentales, debido a que la resolución de las ecuaciones de Navier-Stokes no puede realizarse de forma analítica. Estos métodos presentan como desventaja que para el caso del estudio teórico es necesario trabajar con un caso sencillo, donde se puedan realizar distintas simplificaciones de las ecuaciones para cada caso. Si el problema presenta una mayor complejidad es

necesario trabajar con ensayos en túneles de viento o en banco.

Con el aumento de la capacidad de computación del siglo XX llegó la mecánica de fluidos computacional, en un primer lugar como herramienta para combinar y complementar con la rama experimental, hasta llegar al punto de avance en la tecnología que ha permitido que la mecánica de fluidos computacional sea una de las opciones principales de estudio, resolviendo de forma numérica las ecuaciones en derivadas parciales de Navier-Stokes.

Trabajando con la mecánica de fluidos computacional es posible evitar muchos de los problemas e inconvenientes presentes en el estudio experimental, como pueden ser la gran dificultad a la hora de imponer unas condiciones de contorno precisas y constantes, requiriendo de una gran complejidad técnica, o la dificultad de realizar medidas sobre el flujo sin perturbarlo, ya que es necesario colocar los instrumentos de medida en el mismo, causando una discrepancia entre los valores que se miden en el flujo y lo que presentaría el mismo si no estuviesen los instrumentos de medida.

La mecánica de fluidos computacional solventa esto de forma directa mediante los cálculos numéricos a partir de las condiciones de contorno deseadas sin más gastos que los de computación, permitiendo esto el estudio de una gran variedad de flujos de forma precisa y asequible.

Sin embargo, existen motivos negativos al uso de la mecánica de fluidos computacional como es la gran cantidad de datos y variables con las que se debe trabajar, debido principalmente al realizar un estudio de un proceso con tantos grados de libertad, causando esto la necesidad del uso de supercomputadores con capacidades superiores a las actuales.

En el estudio de la mecánica de fluidos, las escalas características presentan un tamaño muy pequeño, causando que los grados de libertad de los flujos turbulentos sean proporcionales al número de elementos en los que se caracteriza el dominio y al paso temporal, causando que los grados de libertad necesarios para el estudio de la mayoría de flujos sea demasiado elevado.

El crecimiento del número de grados de libertad está también relacionado con el número de Reynolds según los estudios de Kolmogorov, el cual asegura que los

grados de libertad son directamente proporcionales a $Re^{9/4}$. Este es el principal problema con el que se encuentra la mecánica de fluidos computacional, ya que es el causante del gran aumento de la cantidad de memoria necesaria para realizar los cálculos, siendo necesarias varias décadas para poder llegar a utilizar estos métodos para el cálculo de los parámetros de un avión en vuelo, según las predicciones de la ley de Moore.

Con esto en cuenta y mientras se consigue alcanzar un punto en el que las capacidades de computación permitan realizar estos cálculos, se han tomado dos vías distintas para avanzar en la mecánica de fluidos computacional. La primera es la optimización de los códigos usados para la resolución de ecuaciones en derivadas parciales y, en segundo lugar, la búsqueda de metodologías para modelar la turbulencia en sus distintas escalas sin la necesidad de resolverlas.

Estas estrategias permiten optimizar los códigos de resolución de las ecuaciones de Navier-Stokes, sin embargo siguen enfrentándose al problema del gran aumento de los grados de libertad del flujo a altos números de Reynolds. Esto ha causado que los modelos de trabajo que se tomen sean tales que, si bien no representan de forma exacta la física presente en el problema, permitan obtener datos de interés del flujo bajo ciertas condiciones.

Tras esta introducción donde se ha explicado el fenómeno de la turbulencia como parte de la mecánica de fluidos y se ha tratado la mecánica de fluidos computacional como la tercera rama de estudio de la mecánica de fluidos, cabe destacar que, si bien nos hemos referido siempre a esta como una resolución numérica directa (DNS), la realidad es que, debido a la gran cantidad de recursos computacionales y de capacidad de cálculo necesaria, esta está restringida normalmente al campo académico y de investigación, usándose en las industrias modelos de turbulencia que aportan una información del flujo más limitada y que requiere de comprobaciones adicionales.

1.2. Objeto de estudio

El objetivo del trabajo que se presenta a continuación es la implementación y programación de un código eficiente en Fourier para la simulación DNS de un flujo turbulento incompresible, el cual circula por un canal de sección rectangular.

Para este trabajo, se parte de un código similar, el cual presenta una simulación

DNS de un flujo confinado por dos paredes enfrentadas. En este caso se aplica un esquema de FD 1D en la dirección de las paredes y transformadas de Fourier en las otras dos direcciones. A mayores de este código se parte de una función capaz de aplicar CFD en 2D, la cual será necesaria para aplicar este método en el caso de estudio del canal, ya que este presenta paredes enfrentadas en dos de sus tres direcciones.

Para nuestro caso de estudio, el aumento de grados de libertad vinculado al aumento del número de Reynolds corresponde en un aumento de los nodos del esquema de diferencias finitas compactas. Esto conlleva un aumento de la carga computacional de los procesadores y del volumen de memoria a ocupar, por lo que la optimización del código en estos apartados es de interés primario.

Los principales puntos de este trabajo son:

- Adaptación del código previo para su utilización en 2D.
- Adaptación e implementación del código para el cálculo de CFD en 2D.
- Estudio de las simetrías del problema para la reducción computacional.
- Paralelización del código.
- Estudio y validación de los resultados.
- Lanzamiento del código en supercomputador, simulando problemas superiores al estado del arte actual, pasando de números de Reynolds de fricción de 2000 a números de Reynolds de fricción de 4000.

2. Estado del arte

A continuación se explicará el estado actual de la mecánica de fluidos computacional, sus principales métodos junto con sus características, sirviendo esto de orientación para temas tratados posteriormente en este proyecto.

En primer lugar, se considera de interés explicar el concepto de *cascada de energía*.

2.1. Cascada de energía

El concepto de cascada de energía fue propuesto por primera vez por el matemático Andréi Nikoláyevich Kolmogórov en 1941 para resolver la paradoja de Hagen y Darcy la cual estipulaba que para un régimen de flujo suficientemente turbulento las pérdidas de presión generadas son independientes del número de Reynolds.

Esto entra en conflicto con diversas relaciones que se pueden obtener de la ecuación de la cantidad de movimiento, las cuales relacionan la variación temporal de la energía cinética con la disipación viscosa de energía. Esto indicaba que a medida que las fuerzas inerciales dominaban sobre las fuerzas viscosas, la disipación viscosa tendería a desaparecer y la energía cinética a conservarse. Este fue el hecho que demostraron como falso Hagen y Darcy en sus experimentos.

Figura 2.1: Representación de la naturaleza multiescala de la turbulencia

Ante esta paradoja, Andréi N. Kolmogorov introdujo el concepto de cascada de energía, la cual se basa en la naturaleza multiescala de la turbulencia. De esta manera, en primer lugar se forman torbellinos de un tamaño característico L, similar al problema de estudio, al introducirse la energía en el flujo. En estas primeras escalas de tamaño característico L la disipación presente es muy baja, sin embargo, estos torbellinos presentan una alta inestabilidad, por lo que tienden a romperse en torbellinos de menor tamaño. Este proceso se repite de forma periódica hasta que los torbellinos alcanzan un tamaño característico l en el cual la presencia de las fuerzas viscosas supera a las inerciales, el Reynolds local presenta un valor de $Re \approx 1$ y la disipación viscosa toma un valor importante en el flujo.

La escala de Kolmogorov (η) se define como la escala en el punto en el que los procesos de inestabilidad de los torbellinos son superados por la disipación viscosa, siendo esta la menor escala de estudio posible, ya que a partir de este punto se los torbellinos son destruidos por la viscosidad.

A partir de las dos escalas definidas, la de Kolmogorov y la escala integral, es posible dividir las estructuras turbulentas:

- Grande estructuras turbulentas, de dimensión $l \approx L$. En esta estructura se introduce energía al flujo.
- Rango inercial, de dimensión $L \gg l \gg \eta$. La inestabilidad domina sobre la

disipación viscosa, por lo que los torbellinos se dividen en otros de menor tamaño.

- Rango disipativo, de dimensión $l \ll \eta$. La disipación viscosa actúa más rápido que la inestabilidad, por lo que los torbellinos tienden a disiparse en calor en lugar de dividirse en otros más pequeños.

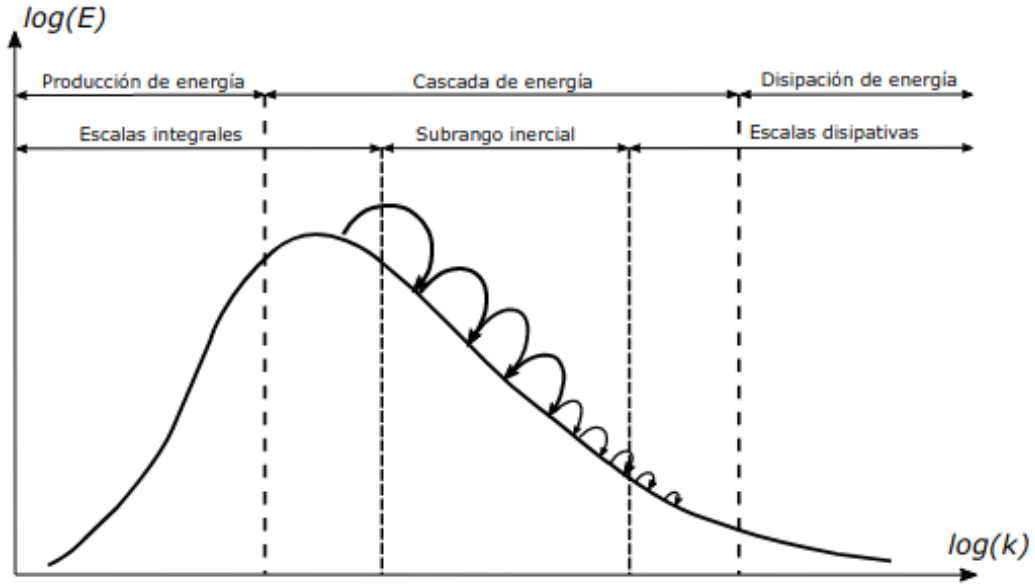

Figura 2.2: Cascada de energía de Kolmogorov.

En esta división la energía cinética del flujo está asociada a los torbellinos de mayor tamaño, mientras que los de menor tamaño experimentan las fuerzas de disipación.

2.2. Modelado de la turbulencia

A continuación se definen los distintos modelos de turbulencia más comunes: DNS, LES y RANS. El uso de estos modelados de la turbulencia se basa en la gran complejidad presente en los torbellinos, por lo que su resolución directa no se considera factible

2.2.1. Direct Numerical Simulation (DNS)

La simulación numérica directa o DNS consiste en la resolución numérica de las ecuaciones de Navier-Stokes, no siendo por tanto una modelización de la turbulencia en sí. Esto implica la resolución de todas las escalas de la turbulencia, dando los resultados más similares a los obtenidos en la realidad en caso de disponer de instrumentos de medida suficientemente precisos y que no causen una perturbación del flujo. Esto, si bien es su mayor ventaja, conlleva una complejidad en los cálculos excesivamente grande para la gran mayoría de flujos de interés para la industria, haciendo que este tipo de simulaciones sean usadas principalmente en el campo de la investigación, con dominios de estudio pequeños y Reynolds moderados.

2.2.2. Large Eddy Simulation (LES)

Las simulaciones LES surgieron como alternativa a las simulaciones DNS, debido a su gran coste computacional. Las simulaciones LES se basan en el principio de que gran parte de la energía presente en la turbulencia se encuentra en las escalas de mayor tamaño, siendo estas las que varían en los distintos problemas de estudio. Teniendo esto en cuenta, se realiza un filtrado de las ecuaciones de forma que solamente se resuelvan las escalas mayores de la turbulencia, modelando el efecto de la disipación viscosa para las escalas menores ya que estas presentan un carácter más universal entre los distintos problemas de estudio. Gracias a esto, es posible disminuir los grados de libertad presentes en el problema, al realizar una discretización de mayor grosor

2.2.3. Reynolds Averaged Navier Stokes (RANS)

Si bien las simulaciones LES suponen un ahorro de los recursos en comparación con las simulaciones DNS, en la mayoría de los casos estas siguen sin ser utilizadas en la industria, ya que suponen una carga computacional y un volumen de datos fuera de los intereses de esta. Esto es debido a que, en líneas generales, el interés de la industria se centra en obtener los parámetros globales del flujo, como las fuerzas medias o el grado de mezcla, no presentando tanto interés por las estructuras turbulentas. Es por esto, que en la mayoría de los casos el tipo de simulaciones usadas son las de tipo RANS. Este tipo de simulaciones se basan en promediar el Reynolds de las ecuaciones de Navier-Stokes, descomponiendo las variables de flujo en su valor medio y fluctuante, obteniendo un nuevo sistema de ecuaciones con un término adicional desconocido, el cual es necesario modelar para el cierre de las ecuaciones.

Los modelos de cierre de las ecuaciones se dividen en modelos de cierre de primer y segundo orden.
A diferencia del LES, en el caso del RANS tanto las escalas de turbulencia pequeñas como las grandes son modeladas, reduciendo en gran medida la carga computacional necesaria hasta unos valores asequibles. Como parte negativa, es necesario validar los resultados obtenidos por este tipo de simulaciones con resultados experimentales o simulaciones DNS, de forma que se pueda ajustar el modelo.

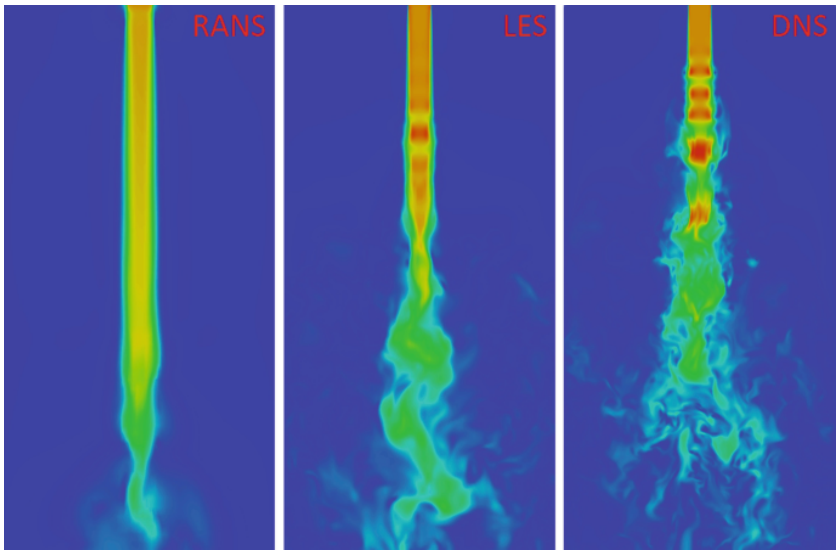

Figura 2.3: Comparativa de los distintos modelos de turbulencia

2.3. Técnicas de discretización espacial

A mayores de el modelo seleccionado a la hora de afrontar las derivadas parciales, es necesario realizar un proceso de discretización espacial del dominio de estudio de forma que las relaciones que se presentan a partir de las ecuaciones en derivadas parciales puedan solucionarse mediante operaciones sencillas tales como suma, resta, multiplicación y división.

Los esquemas numéricos más utilizados para conseguir esta simplificación son:

- **Diferencias finitas (FD):** Esquema basado en las series truncadas de Taylor, de forma que se usan estas para aproximar las derivadas espaciales. Es uno de los métodos de discretización más simples por lo que fue uno de los

primeros implementados. Su problema radica en la dificultad para la discretización de dominios complejos y su poca flexibilidad.

- **Elementos finitos (FEM):** En este método se presenta la solución como un vector de carácter nodal, presentando un valor nulo en todo el dominio a excepción de en los nodos que representan.

- **Volúmenes finitos (FVM):** Este método consiste en la representación de las ecuaciones del flujo de forma integral. De esta manera, las integrales de volumen en una ecuación diferencial parcial que contienen un término de divergencia se convierten en integrales de superficie, evaluándose estos como flujos en las superficies de cada volumen finito, imponiéndose el cumplimiento de las ecuaciones en forma integral. Uno de los mayores fuertes de este método es la facilidad para adaptarse a dominios complejos, por lo que la mayoría de simulaciones RANS optan por este esquema. Sin embargo, el FVM lleva asociado a su vez un bajo grado de precisión, haciendo necesaria una discretización más precisa para mantener el mismo error que otros esquemas.

- **Métodos espectrales:** De forma similar al FEM, este método representa la solución como una base de funciones, siendo en este caso estas funciones trigonométricas y obtenidas mediante su desarrollo en Fourier. Esto permite aproximar las derivadas de una forma exacta.

- **Método de Galerkin Discontinuo (DGM):** Este método combina características del marco de elementos finitos y de volumen finito y se han aplicado con éxito a problemas hiperbólicos, elípticos, parabólicos y de forma mixta que surgen de una amplia gama de aplicaciones.

- **Diferencias finitas compactas (CFD):** Desarrolladas por el científico S.K. Lele en un artículo de 1991 [5] presentan una gran similitud con el método tradicional de diferencias finitas. La diferencia con el FD reside en la utilización de la variable en ciertos nodos para la obtención de la derivada de la variable en los mismos. Esto permite obtener un método de mayor precisión que el FD para un mismo número de nodos.

Una vez contextualizadas las distintas técnicas de discretización espacial se procede a definir el objeto de estudio.

3. Simulación DNS en canales rectangulares

El dominio de estudio para este trabajo consiste en un flujo confinado en un canal rectangular, como ya se comentó. Este tipo de dominio, si bien presenta las características de simplicidad y simetría para poder aplicar simulación DNS, sigue siendo un flujo de gran interés para la industria, existiendo numerosos casos donde este tipo de flujo se nos presenta, como pueden ser sistemas de admisión en distintos tipos de motores, sistemas de ventilación, ... El principal interés presente en el estudio de este tipo de estructuras reside en el análisis de la turbulencia anisotrópica que se presenta.

Debido a la naturaleza compleja y caótica de la turbulencia esta suele ser estudiada mediante el análisis estadístico de la misma. Para ello se generan distintos campos aleatorios del tipo $U(x,t)$, los cuales se pueden dividir en tres tipos:

- **Campo estadísticamente homogéneo:** Consiste en un campo estadístico el cual sus estadísticas son constantes frente a los desplazamientos del sistema de referencia, tanto desplazamientos de posición como rotaciones y reflexiones.
- **Campo estadísticamente isótropo:** Los campos estadísticamente isótropos son aquellos campos cuyas estadísticas permanecen constantes frente a rotaciones y reflexiones del sistema de referencia.
- **Campo anisótropo:** Los campos anisótropos son aquellos en los que las estadísticas del campo no se mantienen constantes al modificar el sistema de referencia.

En nuestro caso de estudio, debido a la presencia de los dos pares de paredes enfrentadas que restringen nuestro dominio, el campo es de tipo anisótropo. Sin embargo, debido a la presencia de flujo desarrollado en la dirección longitudinal, se puede considerar un campo estadísticamente homogéneo en esa dirección.

El interés por este tipo de flujos se remonta al año 1972, cuando Orszag y Patterson [6] realizaron una simulación mediante métodos espectrales y condiciones de contorno en un dominio de flujo en tres dimensiones. Se simuló un caso de turbulencia isótropa con un Reynolds basado en la microescala de Taylor (Re_λ) de 35 y una descomposición del espacio físico de $3 \cdot 10^4$ puntos.

La microescala de Taylor consiste en una escala intermedia entre la escala de los grandes torbellinos y las pequeñas escalas. Esta es de gran interés en la turbu-

lencia isótropa ya que el número de Reynolds basado en la microescala de Taylor (Re_λ) es el de mayor interés cuando el flujo se encuentra desarrollado.

Posteriormente, en 1987, Kim, Moin y Moser [7] realizaron la primera simulación de flujo confinado por dos paredes. En su caso la simulación se llevó a cabo en un flujo confinado entre dos paneles enfrentadas, permitiendo así al flujo desarrollarse en las otras dos direcciones del dominio. Esta simulación se realizó con un Reynolds de fricción $Re_\tau = 180$ y una descomposición del espacio físico de $4 \cdot 10^6$ puntos.

En el caso del estudio realizado por Kim, Moin y Moser se recurrió al Reynolds de fricción (Re_τ) ya que este es el de mayor interés para el estudio de la turbulencia anisótropa, como es la presentada en la turbulencia de pared. Las relaciones entre los distintos números de Reynolds se pueden expresar como:

$$Re_L \approx \frac{3}{20} Re_\lambda^2 \tag{3,2}$$

$$Re_\tau \approx \frac{1}{20} Re_\lambda^2 \tag{3,3}$$

Actualmente, debido al aumento exponencial de los recursos y la capacidad de computación se han podido llevar a cabo grandes avances en este campo, llegando a estudios como los de Hoyas y Jiménez en 2003 [8], trabajando con un $Re_\tau = 2003$, el de Y. Yamamoto y Y. Tsuji en 2018 [9], con un $Re_\tau = 8000$ o, más recientemente, el estudio de Sergio Hoyas y Francisco Alcántara [10], publicado en enero de 2022 para un $Re_\tau = 10000$. Estos números presentan un gran avance respecto a los presentados en 1987, pero siguen estando fuera del rango de interés de un avión en vuelo, con valores de $Re_\tau \approx 40000$ o de fenómenos más complejos presentes en la atmósfera, con un $Re_\tau 10^6$.

3.1. Dominio de estudio

Como se comentó previamente, el dominio de estudio del problema consta de un canal rectangular con paredes enfrentadas en dos direcciones y flujo libre en la restante. Esto obliga a utilizar técnicas de discretización espacial compatibles con condiciones de contorno no periódicas, como son las asumidas en los métodos espectrales.

En el caso de la dirección principal del flujo, este puede considerarse completamente desarrollado y por tanto periódico. Con esta premisa se pueden aplicar métodos espectrales, dado que la turbulencia es homogénea, reduciendo así el coste computacional en gran medida, ya que estos métodos son muy baratos computacionalmente hablando.

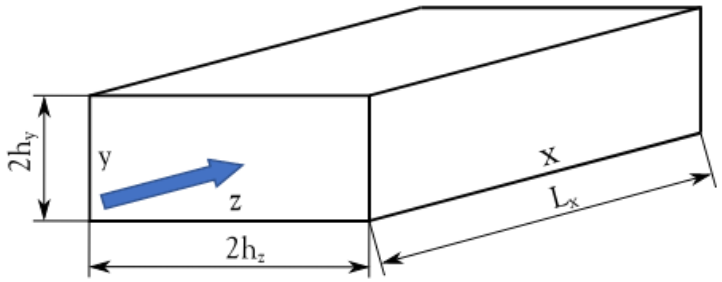

Figura 3.4: Representación del dominio tomado para el problema a estudiar.

Las dimensiones del canal son $2h_y$ de altura,$2h_z$ de anchura y L_x de longitud. La longitud L_x se considera periódica al utilizar un método espectral como el Fourier, por lo que a efectos prácticos es como si esta longitud se extendiera de forma infinita en tramos de L_x. Debido a esto la longitud L_x debe ser lo bastante grande como para que el flujo se llegue a desarrollar completamente. A mayores de esto, es importante considerar que para el caso particular de un canal cuadrado, donde $h_y = h_z$ se aumentan considerablemente las simetrías presentes en el problema, reduciendo en gran medida la carga de cálculo.

3.2. Discretización espacial

Las ecuaciones a resolver son ecuaciones en derivadas parciales del tipo:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathcal{L}(\mathbf{u}) = 0 \tag{3,4}$$

donde $\mathbf{u}$ es la magnitud de interés y $\mathcal{L}$ es el operador que contiene las derivadas del problema.

Como ya se comentó, el esquema de discretización espacial usado para este problema es mixto,aplicándose CFD en las direcciones donde se encuentran las

pareces (y, z) y un método espectral, en este caso la transformada rápida de Fourier (FFT) en la dirección periódica (x).

3.2.1. Métodos espectrales (Transformadas de Fourier)

Los métodos espectrales consisten en un desarrollo de la función a estudiar $(\mathbf{u}(x))$ mediante una base infinita de funciones ortogonales (ϕ_k). En nuestro caso se usa el desarrollo en series de Fourier, de tal manera que,

$$\mathbf{u}(x) = \sum \hat{u}_k \phi_k \tag{3,5}$$

$$\phi_k = e^{ikx} \tag{3,6}$$

donde $\hat{u}_k$ son los coeficientes de la serie.

Aplicando este método con el suficiente refinamiento se puede alcanzar la precisión espectral, la cual se da cuando el coeficiente k-ésimo de la serie decae a mayor velocidad que cualquier potencia inversa de k.

A partir de la estructura presentada en las ecuaciones (3,5) y (3,6), es necesario obtener el valor de los coeficientes de Fourier $(\hat{u}_k)$. Para esto será necesario aplicar el desarrollo discreto de Fourier, ya que estos coeficientes deben obtenerse de forma aproximada, siendo imposible su obtención de forma exacta.

Para la aplicación de este método se parte de un dominio (en nuestro caso se toma uno de tamaño 0-2π) discretizado con N nodos, de forma que:

$$x_j = \frac{2\pi j}{N}, \qquad j = 0, ..., N-1 \tag{3,7}$$

Los coeficientes de la serie discreta de Fourier se pueden expresar como:

$$\hat{u}_k = \frac{1}{N} \sum_{j=0}^{N-1} u(x) e^{-ikx_j} \tag{3,8}$$

Para deshacer el cambio y volver al espacio físico se hace mediante:

$$u(x_j) = \sum_{k=N/2}^{N/2-1} \hat{u}_k e^{ikx_j} \tag{3,9}$$

Al realizar los desarrollos de esta manera nos aseguramos trabajar con precisión espectral tanto para integrales como para derivadas, siendo además computacionalmente muy baratas ya que al trabajar con series de Fourier estas son únicamente multiplicaciones y divisiones del número de onda. Además de esto, las condiciones de contorno son impuestas de manera implícita, considerándose periódicas por ser este un desarrollo periódico.

La derivada del coeficiente de Fourier es:

$$\hat{u}'_k = ik\hat{u}_k \tag{3,10}$$

3.2.2. Diferencias Finitas Compactas (CFD)

A pesar de las claras ventajas de los métodos espectrales comentadas en el apartado anterior, con un coste computacional realmente bajo y gran sencillez para el cálculo de derivadas e integrales, no es posible aplicar este mismo método en las direcciones y e z, ya que no presentan condiciones de contorno periódicas, como si lo hace la dirección x.

Las diferencias finitas compactas o CFD permiten obtener unos resultados de alta precisión con un coste computacional bajo. Esto, combinado con la posibilidad de imponer distintas condiciones de contorno hace que sea la mejor opción como método para aplicar en las direcciones y y z.

Este método nació como una mejora directa del método de las diferencias finitas clásico, el cual consiste en la aproximación de la derivadas espaciales mediante series truncadas de Taylor, convirtiendo los operadores matemáticos en una expresión algebraica en función de **u**.

La diferencia existente entre las diferencias finitas y las diferencias finitas compactas reside en la existencia de una relación entre el valor de la derivada en varios puntos con el valor que toma la variable de estudio en los mismos. Esto se expresa

como:

$$\sum_{j=-n,n} \alpha_j u^k(x_i) = \sum_{j=-m,m} \beta_j u(x_i) \tag{3,11}$$

donde m y n son la semilongitud del *stencil* de cálculo, el cual determina el orden de truncamiento de la serie de Taylor y $u^k(x_i)$ es la derivada de orden k evaluada en x_i.

De esta manera la diferencia existente entre diferencias finitas normales y diferencias finitas compactas se puede representar como:

- Diferencias finitas

$$\mathbf{u}^k = B\mathbf{u} \tag{3,12}$$

- Diferencias finitas compactas

$$\mathbf{u}^k = B\mathbf{u} \tag{3,13}$$

Donde A y B son matrices dispersas que contienen los coeficientes α y β y u y u^k son la magnitud a estudiar y su derivada en forma vectorial, siendo $u_i = u(x_i)$.

$$u = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \\ u_{N+1} \end{pmatrix}; \tag{3,14}$$

$$u^k = \begin{pmatrix} u_1^{(k)} \\ u_2^{(k)} \\ \vdots \\ u_N^{(k)} \\ u_{N+1}^{(k)} \end{pmatrix}; \tag{3,15}$$

Uno de los principales problemas presentes en este método es la imposibilidad de usarlo en 2D, ya que implica un error en los puntos de las esquinas. La metodología que se ha empleado para solventar esto es realizar dos discretizaciones 1D

en ambas direcciones y y z. Esto permite definir distintas matrices, tanto para las derivadas simples como dobles, las cuales tienen la forma

$$\begin{aligned} A_y\mathbf{u}_y = B_y\mathbf{u}, \quad A_{yy}\mathbf{u}_{yy} = B_{yy}\mathbf{u} \\ A_z\mathbf{u}_z = B_z\mathbf{u}, \quad A_{zz}\mathbf{u}_{zz} = B_{zz}\mathbf{u} \end{aligned} \tag{3,16}$$

Donde $\mathbf{u}$ es la organización en forma de matriz de los nodos $u_{i,j} = u(y_i, z_i)$ para $i = 1, \ldots, n_y$, $j = 1, \ldots, n_z$ con la forma

$$\mathbf{u} = [u_{i,j}] = \begin{bmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,j} & \cdots & u_{1,n_z} \\ u_{2,1} & \vdots & \cdots & u_{2,j} & \cdots & u_{2,n_z} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ u_{i,1} & u_{i,2} & \cdots & u_{i,j} & \cdots & u_{i,n_z} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ u_{n_y,1} & u_{n_y,2} & \cdots & u_{n_y,j} & \cdots & u_{n_y,n_z} \end{bmatrix} \tag{3,17}$$

Para el caso de 1D, se premultiplica por la matriz A perteneciente a la dirección en la cual se quiera derivar. De esta manera la única incógnita a resolver es $\mathbf{u}$.

En el caso de 2D, esto se complica ya que es necesario realizar la transformación en ambas direcciones espaciales. Para ello se realiza una premultiplicación por la matriz correspondiente a la dirección y en nuestro caso (A_y) y una postmultiplicación por la matriz correspondiente a la dirección z (A_z). Esto da lugar las expresiones

$$A_y\mathbf{u}_yA_z = B_y\mathbf{u}A_z, \quad A_y\mathbf{u}_yA_z = A_y\mathbf{u}B_z \tag{3,18}$$

$$A_{yy}\mathbf{u}_{yy}A_{zz} = B_{yy}\mathbf{u}A_{zz}, \quad A_{yy}\mathbf{u}_{zz}A_{zz} = A_{yy}\mathbf{u}B_{zz} \tag{3,19}$$

De esta manera la variable a estudiar es premultiplicada y postmultiplicada por matrices A y B, lo que hace que su resolución sea más difícil e implique un

mayor gasto computacional.

El método que se ha utilizado para solventar esto consiste en la definición de unas matrices G de la forma

$$B_y \mathbf{u} A_z = G_1 \mathbf{u}^c, \quad B_{yy} \mathbf{u} A_{zz} = G_1 2 \mathbf{u}^c \tag{3,20}$$

$$A_y \mathbf{u} B_z = G_2 \mathbf{u}^c, \quad A_{yy} \mathbf{u} B_{zz} = G_2 2 \mathbf{u}^c \tag{3,21}$$

$$A_y \mathbf{u} A_z = G_3 \mathbf{u}^c, \quad A_{yy} \mathbf{u} A_{zz} = G_3 2 \mathbf{u}^c \tag{3,22}$$

Siendo $\mathbf{u}^c$ el vector columna que surge de reorganizar los nodos de $u_{i,j}$. Este tiene la forma

$$\mathbf{u}^c = \begin{bmatrix} u_{1,1} \\ \vdots \\ u_{n_y,1} \\ u_{1,2} \\ \vdots \\ u_{n_y,2} \\ \vdots \\ u_{n_y,n_z} \end{bmatrix}. \tag{3,23}$$

Para ejemplificar esta aplicación del CFD, se va a recurrir a la resolución de la ecuación de Poisson bidimensional,

$$\nabla^2 u = f \tag{3,24}$$

La cual puede expresarse como,

$$\left(\frac{\partial}{\partial y^2} + \frac{\partial}{\partial z^2} \right) u(y,z) = f(y,z) \tag{3,25}$$

siendo y y z las dos direcciones cartesianas de trabajo en el caso bidimensional.

Utilizando la expresión matricial vista en las ecuaciones (3,18) y (3,19) se puede expresar la ecuación anterior como,

$$\mathbf{u}_{yy} + \mathbf{u}_{zz} = \mathbf{F} \tag{3,26}$$

Aplicando el método de CFD explicado anteriormente se premultiplica y postmultiplica por las matrices A_{yy} y A_{zz} y se hacen los cambios pertinentes para llegar a las igualdades,

$$A_{yy}\mathbf{u}_{yy}A_{zz} + A_{yy}\mathbf{u}_{zz}A_{zz} = A_{yy}\mathbf{F}A_{zz} \tag{3,27}$$

$$B_{yy}\mathbf{u}A_{zz} + A_{yy}\mathbf{u}B_{zz} = \tilde{\mathbf{F}} \tag{3,28}$$

$$(G_{12} + G_{22})\mathbf{u}^c = \tilde{\mathbf{F}}^c \tag{3,29}$$

A partir de este punto se imponen las condiciones de contorno necesarias para la resolución del problema, en nuestro caso, condiciones de contorno tanto de Dirichlet como de Neumann.

3.2.3. Condiciones de contorno

En nuestro caso de estudio, las condiciones de contorno a aplicar son las condiciones de contorno de Dirichlet y de Neumann para la resolución de las EDP. En un primer lugar se abordarán las condiciones de contorno de Dirichlet ya que estas presentan una mayor simplicidad, dejando las de Neumann para un apartado posterior.

Al aplicar las condiciones de contorno de Dirichlet se están fijando los valores en la frontera del dominio de la variable incógnita. Para esto se define el dominio interior como $\mathbf{u}^{\Omega}$ y el dominio interior como $\mathbf{u}^{\partial}\Omega$ de forma que,

$$\mathbf{u} = \mathbf{u}^{\Omega} + \mathbf{u}^{\partial}\Omega \tag{3,30}$$

$$\mathbf{u}^{\partial\Omega} = \begin{bmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,n_z-1} & u_{1,n_z} \\ u_{2,1} & 0 & \cdots & 0 & u_{2,n_z} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ u_{n_y-1,1} & 0 & \cdots & 0 & u_{n_y-1,n_z} \\ u_{n_y,1} & u_{n_y,2} & \cdots & u_{n_y,n_z-1} & u_{n_y-1,n_z} \end{bmatrix} \tag{3,31}$$

$$\mathbf{u}^{\Omega} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & u_{2,2} & \cdots & u_{2,n_z-1} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & u_{n_y-1,2} & \cdots & u_{n_y-1,n_z-1} & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}. \tag{3,32}$$

Los valores que deben tomar los coeficientes de $\mathbf{u}^{\partial\Omega} =$ vienen dados por las condiciones de contorno tomadas, por lo que este es un valor conocido en nuestras ecuaciones, lo que deja como incógnita a resolver el valor de la variable de estudio en su dominio interior ($\mathbf{u}^{\Omega}$). Para resolver este problema se usa la siguiente simplificación,

$$G = G_{12} + G_{22}, \tag{3,33}$$

de forma que

$$G\mathbf{u} = \tilde{\mathbf{F}}^{c}, \tag{3,34}$$

$$G\left(\mathbf{u}^{\Omega c} + \mathbf{u}^{\partial\Omega c}\right) = \tilde{\mathbf{F}}^{c}, \tag{3,35}$$

$$G\mathbf{u}^{\Omega c} = \tilde{\mathbf{F}}^{c} - G\mathbf{u}^{\partial\Omega c} = \mathbf{f}^{c} \tag{3,36}$$

Tomando únicamente lo que nos interesa de estas ecuaciones, que son los dominios interiores, y descartando las condiciones de contorno obtenemos la igualdad

$$\bar{G}\mathbf{u}^{\bar{\Omega}c} = \bar{\mathbf{f}}^{c}. \tag{3,37}$$

De esta manera se pueden obtener los valores de la variable de estudio $\mathbf{u}$ en los nodos del dominio interior, los cuales al sumarse junto con las condiciones de contorno permiten obtener el valor completo de la variable de estudio en todos los nodos.

3.3. Discretización temporal

Una vez planteado todo el proceso de discretización espacial como se ha visto en el apartado (3.2), con el cual se puede resolver el operador diferencial $\mathcal{L}$, se procede a realizar el estudio de la resolución de la discretización temporal del problema. A la hora de abordar esto, es posible utilizar distintos métodos explícitos con diferentes ventajas cada uno. En nuestro caso se ha decidido implementar el método de discretización temporal basado en un esquema *Runge-Kutta* de 3 subpasos temporales.

El método de *Runge-Kutta* para un orden s para la integración de una variable $\mathbf{u}$ es de la forma,

$$\mathbf{u}^{n+1} = \mathbf{u}^{n} + \Delta t \sum_{i=0}^{s} b_i k_i, \tag{3,38}$$

donde s es el número de subpasos temporales del método y k_i son los términos de aproximación intermedios evaluados de manera local.

$$k_i = f\left(u^n + \Delta t \sum_{i=0}^{s} a_{ij} k_i, t_n + c_i \Delta t\right), \quad i = 1, \ldots, s \tag{3,39}$$

donde a_{ij}, b_i y c_i son los coeficientes del esquema, dependientes de la regla de cuadratura que se haya tomado. Los valores asociados a estas variables son los que determinarán la naturaleza del método, pudiendo ser éste explícito o implícito.

En el caso de los sistemas de tipo explícito se tomará la información exclusivamente de instantes anteriores al de estudio para calcular los valores tomados por

la variable. Para los esquemas implícitos se toma la información tanto de instantes previos como del propio instante de estudio para calcular el valor de la variable.

Si bien los sistemas implícitos presentan diversos puntos de interés, como la capacidad de utilizar pasos temporales (Δt) de mayor tamaño, permitiendo una discretización más gruesa, y por tanto de menor coste computacional,sin presentar inestabilidades, estos sistemas también presentan una importante limitación, la necesidad de un comportamiento lineal de las variables de estudio. En caso de no presentar este comportamiento, los sistemas implícitos no son capaces de afrontar el problema de forma correcta, ya que únicamente pueden usarse los instantes de tiempo anteriores para la resolución de las variables en el siguiente paso temporal, introduciendo esto un error en las mismas al usar métodos implícitos.

Dado que nuestras variables de estudio no presentan comportamiento lineal en el tiempo es necesario que el esquema que se use para afrontar el problema sea de tipo explícito o mixto.

A continuación se muestra la discretización tomada para el operador algebraico usado en nuestro problema, dividiendo este en su parte lineal (términos derivados de la presión y viscosidad) y en su parte no lineal (término convectivo).

El operador algebraico puede expresarse como:

$$\frac{\partial u}{\partial t} = L(\mathbf{u}) + N(\mathbf{u}) \tag{3,40}$$

Siendo n la agrupación de los términos lineales del operador $\frac{\partial u}{\partial t}$ y N la agrupación de los términos no lineales.

La implementación de este esquema es que permite la posibilidad de usar un método mixto de discretización, mezclando métodos implícitos para la resolución de la parte lineal y métodos explícitos para la parte no lineal. En nuestro caso se ha tomado un método implícito de tercer orden y un método explícito de segundo orden, de forma que para realizar un paso temporal completo es necesario realizar 3 subpasos.

Los subpasos a realizar siguen el siguiente esquema:

$$\mathbf{u}_1^n = \mathbf{u}^n + \Delta t \left[L(\alpha_1 \mathbf{u}^n + \beta_1 \mathbf{u}_1^n) + \gamma_1 N^n \right], \tag{3,41}$$
$$\mathbf{u}_2^n = \mathbf{u}_1^n + \Delta t \left[L(\alpha_2 \mathbf{u}_1^n + \beta_2 \mathbf{u}_2^n) + \gamma_2 N_1^n + \zeta_1 N^n \right], \tag{3,42}$$
$$\mathbf{u}^{n+1} = \mathbf{u}_2^n + \Delta t \left[L(\alpha_3 \mathbf{u}_2^n + \beta_3 \mathbf{u}_3^n) + \gamma_3 N_2^n + \zeta_2 N_1^n \right], \tag{3,43}$$

Los coeficientes usados en las ecuaciones 2.41 , 2.42 , 2.43 son obtenidos de forma experimental para el orden deseado. Los valores óptimos para el esquema utilizado en nuestro caso fueron obtenidos por [11] en 1991 y presentan los valores:

$$\begin{aligned} &\gamma_1 = \frac{8}{15}, \ \ \gamma_2 = \frac{5}{12}, \ \ \gamma_3 = \frac{3}{4}, \ \ \zeta_1 = -\frac{17}{60}, \ \ \zeta_2 = -\frac{5}{12}, \\ &\alpha_1 = \frac{29}{96}, \ \ \alpha_2 = -\frac{3}{40}, \ \ \alpha_3 = \frac{1}{6}, \\ &\beta_1 = \frac{37}{160}, \ \ \beta_2 = \frac{5}{24}, \ \ \beta_3 = \frac{1}{6}. \end{aligned} \tag{3,44}$$

3.3.1. Paso temporal (Δt)

Tras especificar la discretización temporal a utilizar en el apartado anterior es interesante realizar un análisis del incremento temporal (Δt) máximo que es posible utilizar en nuestro caso. Para esto es importante establecer previamente dos límites al mismo, de forma que lo acortemos:

- El paso temporal a utilizar (Δt) debe ser tal que no permita a ninguna partícula del flujo avanzar la suficiente distancia en ese (Δt) como para sobrepasar el volumen de control característico, ya que esto introduciría un error en el cálculo de las variables del flujo en cada iteración, causando una gran discrepancia entre los resultados obtenidos y los deseados.
- Es importante tener en cuenta que un paso temporal (Δt) demasiado refinado podría causar un aumento excesivamente grande del volumen de datos a evaluar para un periodo de tiempo simulado, lo que podría derivar en una carga computacional que se escape de nuestras capacidades o de nuestro interés.

Es importante no superar estos límites del paso temporal.

Para parametrizar esto, se desarrolló en 1928 la condición de *Courant-Friedrichs-Lewy (CFL)*, la cual permite relacionar la discretización espacial empleada con la temporal. La condición de Courant (CFL) se expresa como:

$$CFL = \pi \frac{u\Delta t}{\Delta x} \tag{3,45}$$

Siendo u la velocidad del flujo, Δx es el tamaño de la malla en la dirección del flujo y Δt es el paso temporal. Es importante indicar que:

- Para **esquemas explícitos** es condición necesaria que el $CFL \leq 1$ para su estabilidad.
- Para **esquemas implícitos** puede existir convergencia y estabilidad aunque $CFL \geq 1$.

Ya que tanto la velocidad del flujo como el tamaño de los elementos de la malla no son obligatoriamente constantes dentro del dominio de estudio, tampoco lo es el CFL, lo que puede provocar que, si bien para el caso de los valores medios del flujo puede ser estable, es posible que localmente en zonas específicas se presente un CFL inestable, lo que causaría problemas en la simulación.

Para solventar esto se pueden adoptar diversas medidas:

- **Establecer un CFL constante:** De esta manera se adapta el paso temporal a cada paso para que en todo el dominio de estudio se cumpla que el CFL sea menor al de diseño. Para llevar a cabo esto es necesario localizar el punto del dominio de condiciones más adversas, allí donde $\frac{u}{\Delta x}$ presente su valor máximo, y calcular el paso temporal en ese punto, cumpliéndose que:

$$\Delta t = \frac{CFL'}{\pi} \left(\frac{\Delta x}{u} \right)_{min} \tag{3,46}$$

- **Establecer un Δt constante:** la opción que se ha tomado en este proyecto es establecer un Δt fijo, variando el CFD, lo que simplifica en gran medida los cálculos a realizar y permite disminuir la carga computacional. Para esto se toma usa la siguiente igualdad con la cual se puede calcular el CFL a lo

largo del dominio para mantener ese Δt.

$$CFL = \pi \frac{u\Delta t}{\Delta x} \tag{3,47}$$

3.4. Flujos confinados

A continuación se explicarán y desarrollarán las ecuaciones de *Navier-Stokes* para flujos confinados e incompresibles, las cuales son las que gobiernan nuestro dominio de estudio.

Estas ecuaciones de N-S parten de las ecuaciones de la conservación de la masa y de la conservación de la cantidad de movimiento, y presentan la siguiente forma:

$$\nabla \cdot \vec{v} = 0 \tag{3,48}$$

$$\frac{\partial \vec{v}}{\partial t} + \vec{v} \cdot \nabla \vec{v} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \vec{v} \tag{3,49}$$

Estas ecuaciones junto con su resolución están recogidas en el artículo de Kim, Moin y Moser de 1987 [7]. En este documento se transforman las ecuaciones (3,48) y (3,49) en otras dos ecuaciones distintas a las cuales tienen como objetivo resolver las incógnitas de la vorticidad en la dirección normal a la pared (ω_y) y el operador laplaciano de la velocidad normal a la pared ($\nabla^2 v$).

Para esto se utilizará el concepto físico de *helicidad*, el cual se define como

$$\vec{H} = \vec{v} \times \vec{\omega}. \tag{3,50}$$

$\vec{H}$ es el vector helicidad de componentes (H_1, H_2, H_3), $\vec{v}$ es el vector velocidad de componentes (v_1, v_2, v_3) y $\vec{\omega}$ es el vector vorticidad de componentes $(\omega_1, \omega_2, \omega_3)$.

Una vez definida esta nueva variable se puede redefinir el término convectivo de la ecuación (3,49) $\vec{v} \cdot \nabla \vec{v}$ como:

$$\vec{v} \cdot \nabla \vec{v} = \frac{1}{2} \nabla (\vec{v} \cdot \vec{v}) - \vec{v} \times \vec{\omega} = \frac{1}{2} \nabla (\vec{v} \cdot \vec{v}) - \vec{H} \tag{3,51}$$

Tras esto se adimensionaliza con la longitud D característica del problema y con la velocidad de fricción (u_τ), pudiendo expresar la ecuación (3,49) en función del Reynolds de fricción ($Re_\tau = \frac{u_\tau D}{\mu}$)

$$\frac{\partial \vec{v}}{\partial t} + \frac{1}{2}\nabla(\vec{v}\cdot\vec{v}) - \vec{H} = -\frac{1}{\rho}\nabla p + \frac{1}{Re_\tau}\nabla^2\vec{v} \tag{3,52}$$

Siguiendo los procedimientos indicados en [7], junto con sus transformaciones, se puede obtener las ecuaciones de ω_y y ϕ, siendo $\phi = \nabla^2 v$.

$$\frac{\partial \phi}{\partial t} = h_v + \frac{1}{Re_\tau}\nabla^2\phi \tag{3,53}$$

$$\frac{\partial \omega_y}{\partial t} = h_g + \frac{1}{Re_\tau}\nabla^2\omega_y \tag{3,54}$$

donde h_v y h_g son la agrupación de los términos lineales y no lineales, siendo estos:

$$h_v = -\frac{\partial}{\partial y}\left(\frac{\partial H_1}{\partial x} + \frac{\partial H_3}{\partial z}\right) + \left(\frac{\partial^2 H_2}{\partial x^2} + \frac{\partial^2 H_2}{\partial z^2}\right) \tag{3,55}$$

$$h_g = \frac{\partial H_1}{\partial z} - \frac{\partial H_3}{\partial x} \tag{3,56}$$

Donde H_1, H_2 y H_3 son las componentes de la helicidad, las cuales se obtienen de la definición de la misma, siendo estas:

$$\begin{aligned} H_1 &= v\omega_z - w\omega_y \\ H_2 &= w\omega_x - u\omega_z \\ H_3 &= u\omega_y - v\omega_x \end{aligned} \tag{3,57}$$

Una vez obtenidas estas igualdades, se añaden las expresiones de las tres componentes de velocidad y vorticidad:

$$\phi = \nabla^2 v \tag{3,58}$$

$$\frac{\partial u}{\partial x} + \frac{\partial w}{\partial z} = -\frac{\partial v}{\partial y} \tag{3,59}$$

$$\frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} = \omega_y \tag{3,60}$$

$$\frac{\partial \omega_x}{\partial x} + \frac{\partial \omega_z}{\partial z} = -\frac{\partial \omega_y}{\partial y} \tag{3,61}$$

$$\frac{\partial \omega_z}{\partial x} - \frac{\partial \omega_x}{\partial z} = \phi \tag{3,62}$$

Al juntar las ecuaciones mencionadas anteriormente, es posible eliminar el término correspondiente a la presión, ya que esta se calculará posteriormente, no siendo necesario su cálculo de forma explícita. Con esto se consigue la dependencia de únicamente dos variables, por lo que el coste computacional se reducirá en gran medida. Para esto en primer lugar se calcularán la velocidad y vorticidad dentro del dominio de Fourier, calcular la helicidad en el dominio físico, transformar la coordenada x al dominio de Fourier, calcular la parte derecha (RHS) de las ecuaciones de transporte, resolver los sistemas de ecuaciones, divididos en parte real e imaginaria y avanzar en el tiempo.

4. Esquema numérico

Las ecuaciones que rigen en este problema son:

$$\frac{\partial \phi}{\partial t} = h_v + \frac{1}{Re_\tau}\nabla^2 \phi \tag{4,63}$$

$$\nabla^2 v = \phi \tag{4,64}$$

$$\frac{\partial \omega}{\partial t} = h_g + \frac{1}{Re_\tau}\nabla^2 \omega_y \tag{4,65}$$

Se aplicará el esquema de resolución a las ecuaciones (4,66) y (4,67), sirviendo esto como ejemplificación para la resolución de la ecuación (4,65), a la cual se le aplicará un método idéntico para resolverla.

A continuación se pasan las ecuaciones (4,66) y (4,67) del dominio real al dominio de Fourier,

$$\partial_t \hat{\phi}_k(y,z) = \hat{h}_v^k + \frac{1}{Re_\tau}\left(-k^2 + \nabla^2\right)\hat{\phi}_k(y,z) \tag{4,66}$$

$$\left(-k^2 + \nabla^2\right)\hat{v}_k(y,z) = \hat{\phi}_k(y,z). \tag{4,67}$$

Siendo k el modo de la transformada de Fourier, $\hat{\phi}_k$ el modo k de la transformada de Fourier del laplaciano de la velocidad normal a la pared, $\hat{h}_v^k$ el modo k de la transformada de Fourier de los términos no lineales y $\hat{v}_k$ el modo k de la transformada de Fourier de la velocidad lineal normal a la pared.

Una vez se han pasado las ecuaciones al dominio de Fourier, lo que se presenta es un sistema de n_x ecuaciones, siendo n_x el tamaño de la transformada, es decir, el número de planos en los que se ha decidió dividir la dirección periódica del flujo (x).

Este problema que se nos presenta con n_x planos, si bien se puede resolver de forma serial, haciendo que los procesadores trabajen en serie, se ha considerado de

mayor interés la implementación de la resolución en paralelo. Esto permite a los procesadores con los que se vaya a simular dividirse la carga de trabajo, de forma que la cantidad de planos (y, z) que existan en el dominio se repartan de forma equitativa entre los distintos procesadores, resolviéndose todos ellos de forma independiente y comunicadores finalmente con el *máster*, de forma que éste obtenga los resultados finales del problema. Todo éste procedimiento se definirá con mayor detalle en el apartado (5).

Tras pasar al dominio de Fourier se aplica el esquema de integración temporal como se indicó en el apartado (3.3).

$$\hat{\phi}^{n+1}(y,z) - \alpha\nabla^2\hat{\phi}^{n+1}(y,z) = F^n(y,z), \tag{4,68}$$

$$\nabla^2\hat{v}^{n+1}(y,z) = \hat{\phi}^{n+1}(y,z), \tag{4,69}$$

siendo α una constante generada por diversos términos como el número de Reynolds, el número de onda, el paso temporal y constantes de la discretización espacial y $F^n(y,z)$ la conjunción de los términos no lineales.

Para mayor simplicidad en la notación se elimina el acento circunflejo de las variables de Fourier:

$$\phi^{n+1}(y,z) - \alpha\nabla^2\phi^{n+1}((y,z) = F^n(y,z), \tag{4,70}$$

$$\nabla^2 v^{n+1}(y,z) = \phi^{n+1}(y,z), \tag{4,71}$$

A continuación se añaden las condiciones de contorno para la velocidad lineal normal a la pared, las cuales consisten en condiciones de contorno mixtas, siendo estas una mezcla de Dirichlet y Neumann. Esto recae en el problema comentado anteriormente en el apartado (3.2.3), ya que la aplicación de las condiciones de frontera de Neumann sobre la variable v engloban cierta complejidad.

El sistema de ecuaciones resultante una vez se han introducido las condiciones de frontera tiene la forma:

$$\begin{cases} \phi(y,z) - \alpha\nabla^2(y,z) = F(y,z), \\ \nabla^2 v(y,z) = \phi(y,z), \\ v = 0 \text{ en } \partial\Omega, \\ \partial_n v = 0 \text{ en } \partial\Omega, \end{cases} \tag{4,72}$$

A continuación se plantea el método propuesto por Kim, Moin y Moser en [7] para conseguir un valor nulo de la derivada normal de la velocidad en la pared ($\partial_n v = 0$ en $\partial\Omega$):

$$v = v_p + c_1 v_1 + c_2 v_2 + \ldots + c_{2n_y+2n_z} v_{2n_y+2n_z} \tag{4,73}$$

descomponiendo la variable de la velocidad lineal normal a la pared (v) en $2n_y + 2n_z + 1$ variables, siendo estas una solución de tipo particular del sistema de ecuaciones (v_p) y $2n_y + 2n_z$ soluciones homogéneas (v_h) donde h es el número de la solución homogénea.

Los sistemas de ecuaciones para las soluciones particulares y homogéneas se pueden expresar como:

- **Solución particular (v_p):** El sistema de ecuaciones para la solución particular tiene la forma:

$$\begin{cases} \phi_p(y,z) - \alpha\nabla^2\phi_p(y,z) = f(y,z), \\ \nabla^2 v_p(y,z) = \phi_p(y,z), \\ v_p = 0 \text{ en } \partial\Omega, \\ \phi_p = 0 \text{ en } \partial\Omega, \end{cases} \tag{4,74}$$

  donde v_p y ϕ_p son las soluciones partículas, que satisface las condiciones de Dirichlet.

- **Solución particular (v_h):** El sistema de ecuaciones para calcular cualquier solución homogénea tiene la forma:

$$\begin{cases} \phi_h(y,z) - \alpha\nabla^2\phi_h(y,z) = 0, \\ \nabla^2 v_h(y,z) = \phi_h(y,z), \\ v_h = 0 \text{ en } \partial\Omega, \\ \phi_h = 0 \text{ en } \partial\Omega \backslash \{X_h\}, \\ \phi_h(X_h) = 1, \end{cases} \tag{4,75}$$

  donde v_h y ϕ_h son las condiciones homogéneas que satisfacen las condiciones de Neumann y X_h es cualquier punto de la frontera.

De esta manera, las condiciones de contorno impuestas implican que $\partial_n v = 0$, por lo que el número de ecuaciones homogéneas a resolver se ve disminuido en 8 directamente (4 *esquinas* $\times$ 2 *direcciones cartesianas*), teniendo que resolver finalmente un total de $2n_y + 2n_z - 8$ ecuaciones homogéneas.

Tomando derivadas normales en la ecuación (4,73) :

$$\partial_n v(X_h) = \partial_n v_p(X_h) + c_1\partial_n v_1(X_h) + \ldots + c_{2n_y+2n_z-8}\partial_n v_{2n_y+2n_z-8}(X_h) \tag{4,76}$$

Aplicando las condiciones de frontera de Dirichlet y despejando las constantes c_h se tiene:

$$\begin{bmatrix} \partial_n v_1(x_1) & \cdots & \partial_n v_{N_{max}}(x_1) \\ \vdots & \vdots & \vdots \\ \partial_n v_1(x_h) & \cdots & \partial_n v_{N_{max}}(x_h) \\ \vdots & \vdots & \vdots \\ \partial_n v_1(x_{N_{max}}) & \cdots & \partial_n v_{N_{max}}(x_{N_{max}}) \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_h \\ \vdots \\ c_{N_{max}} \end{bmatrix} = \begin{bmatrix} -\partial_n v_p(x_1) \\ \vdots \\ -\partial_n v_p(x_h) \\ \vdots \\ -\partial_n v_p(x_{N_{max}}) \end{bmatrix} \tag{4,77}$$

donde $N_{max} = 2n_y + 2n_z - 8$.

4.1. Aplicación del CFD

A continuación se plantea el método de resolución para obtener v_p y v_h. Aplicando el CFD, se calcula ϕ_p haciendo la premultiplicación y postmultiplicación de A_{yy} y A_{zz}:

$$\begin{aligned} A_{yy}\mathbf{\Phi}A_{zz} - \alpha A_{yy}\mathbf{\Phi}_{yy}A_{zz} - \alpha A_{yy}\mathbf{\Phi}_{zz}A_{zz} &= A_{yy}\boldsymbol{F}A_{zz} \\ A_{yy}\mathbf{\Phi}A_{zz} - \alpha B_{yy}\mathbf{\Phi}A_{zz} - \alpha A_{yy}\mathbf{\Phi}B_{zz} &= \tilde{\boldsymbol{F}} \\ G_{32}\mathbf{\Phi}^c - \alpha G_{12}\mathbf{\Phi}^c - \alpha G_{22}\mathbf{\Phi}^c &= \tilde{\boldsymbol{F}}^c \\ (G_{32} - \alpha G_{12} - \alpha G_{22})\mathbf{\Phi}^c &= \tilde{\boldsymbol{F}}^c \end{aligned} \tag{4,78}$$

Donde se ha seguido el procedimiento explicado en (3.2.2).

Dividiendo las matrices en condiciones de contorno y la zona interior del dominio se obtiene

$$\begin{aligned} G\mathbf{\Phi}^{\Omega c} = \tilde{\boldsymbol{F}}^c - G\mathbf{\Phi}^{\partial\Omega c} &= \mathbf{f}^c \\ \overline{G}\,\overline{\mathbf{\Phi}^{\Omega c}} &= \overline{\mathbf{f}^c} \end{aligned} \tag{4,79}$$

Donde G es la matriz conjunta de $(G_{32} - \alpha G_{12} - \alpha G_{22})$. Con esto es posible determinar el valor de ϕ_p para su dominio interior, ya que el exterior viene dado por las condiciones de contorno. Teniendo este valor se puede calcular de forma similar v_p resolviendo la igualdad $\nabla^2 v_p(y,z) = \phi_p(y,z)$ vista en la ecuación (4), obteniendo así finalmente v_p.

En el caso de v_h y ϕ_h, al no presentar RHS tenemos la igualdad:

$$
\begin{aligned}
A_{yy}\mathbf{\Phi}A_{zz} - \alpha A_{yy}\mathbf{\Phi}_{yy}A_{zz} - \alpha A_{yy}\mathbf{\Phi}_{zz}A_{zz} = 0 \\
A_{yy}\mathbf{\Phi}A_{zz} - \alpha B_{yy}\mathbf{\Phi}A_{zz} - \alpha A_{yy}\mathbf{\Phi}B_{zz} = 0 \\
G_{32}\mathbf{\Phi}^c - \alpha G_{12}\mathbf{\Phi}^c - \alpha G_{22}\mathbf{\Phi}^c = 0 \\
(G_{32} - \alpha G_{12} - \alpha G_{22})\mathbf{\Phi}^c = 0
\end{aligned}
\tag{4,80}
$$

Que aplicando el mismo procedimiento que en el caso de la ϕ_p

$$
\begin{aligned}
G\mathbf{\Phi}^{\Omega c} = -G\mathbf{\Phi}^{\partial\Omega c} = \mathbf{f}^c \\
\overline{G}\overline{\mathbf{\Phi}^{\Omega c}} = \overline{\mathbf{f}^c}
\end{aligned}
\tag{4,81}
$$

Obteniendo de esta manera todas las soluciones homogéneas v_p y ϕ_p. Con esto solo sería necesario comprobar la última condición de frontera, relacionada con la derivada normal en el contorno, como puede verse en (4,72).

Para esto es necesario calcular la derivada normal a la pared de la variable v, siendo $v = v_p + c_1 v_1 + c_2 v_2 + \ldots + c_{2n_y+2n_z} v_{2n_y+2n_z}$, y siendo la definición de derivada normal:

$$
\partial_n v(y,z) = \nabla v(y,z) \cdot \vec{n} \tag{4,82}
$$

Siendo $v(y,z)$ el campo a estudiar y $\vec{n}$ el vector normal unitario a la superficie a estudiar.

Para el cálculo de la derivada normal se aplicará nuevamente *CFD*, calculando en primer lugar las derivadas en y y en z.

$$
\begin{aligned}
G_3\partial_y\mathbf{v}^c = G_1\mathbf{v}^c \\
G_3\partial_z\mathbf{v}^c = G_2\mathbf{v}^c
\end{aligned}
\tag{4,83}
$$

Imponiendo finalmente la condición de frontera de $\partial_n v = 0$ en $\partial\Omega$.

Como puede observarse, la resolución de las ecuaciones del problema implica la resolución de un sistema de ecuaciones homogéneas en el cual se mantiene estable el lado izquierdo de la igualdad, variando el lado derecho, de forma que las matrices del sistema se mantienen constantes. Esto hace que puedan precalcularse antes de entrar a los bucles temporales, ahorrando una gran cantidad de memoria y de carga computacional, y almacenándolas en un módulo para su uso posterior.

Para la resolución de estos complejos sistemas de ecuaciones se implantó en el código el método iterativo denominado PGMRES, el cual consiste en un método GMRES precondicionado, desarrollado por Yousef Saad en el paquete SPARSEKIT2 [12]. Se han usado los métodos de precondicionado ILU(0) (*Incomplete LU Factorization*) y MILU(0) (*Modified Incomplete LU Factorization*), los cuales son los que mejor se adaptan a las características de nuestro problema, tomándose el método ILU(0) para casos donde el condicionamiento inicial de la matriz a precondicionar sea bueno, ya que en este método sobresale por su simplicidad y su velocidad, y tomándose MILU(0) para matrices cuyo condicionamiento inicial sea más complejos y requieran de un precondicionador más completo.

Finalmente, se van a explicar las simetrías tomadas en el dominio del problema, ya que este, al ser rectangular, presenta cierto número de simetrías las cuales permiten reducir en gran medida la carga de cálculos que se deben realizar al calcular las variables homogéneas.

A continuación se explicarán las simetrías tomadas para el caso del dominio rectangular y para el caso específico del dominio cuadrado, ya que este constituye un caso particular de dominio rectangular con más simetrías. Para esto se han diseñado dos imágenes explicativas de las simetrías de estos distintos dominios.

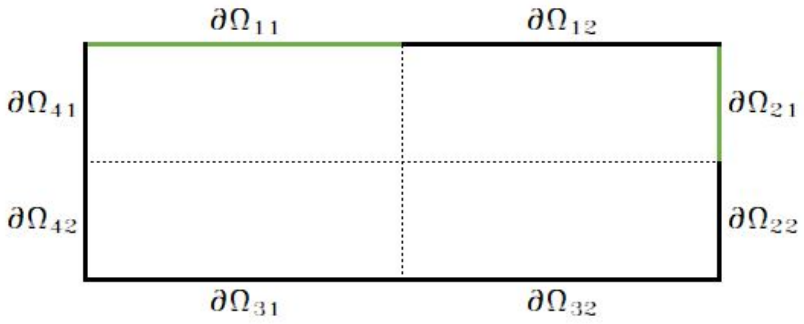

Figura 4.5: Representación de las simetrías existentes en un dominio de estudio rectangular

Como puede verse, la imagen (4.5) representa las simetrías existentes en un dominio de trabajo de tipo rectangular, siendo los nodos de los subcontornos $\partial\Omega_{12}$ y $\partial\Omega_{21}$ los que se irán haciendo no nulos debidos a las condiciones de frontera y siendo el resto de $\partial\Omega_{ij}$ los subcontornos de dominio exterior $\partial\Omega$. Aplicando las simetrías marcadas por las lineas discontinuas se pueden obtener los valores de las soluciones homogéneas de los contornos $\partial\Omega_{22}$, $\partial\Omega_{22}$, $\partial\Omega_{31}$, $\partial\Omega_{32}$, $\partial\Omega_{41}$ y $\partial\Omega_{42}$. Junto con los valores de las variables homogéneas es posible obtener los valores de las derivadas normales, de forma que se obtengan los valores completos de la matriz del sistema (4,77) y pudiendo resolverse la misma.

Para el caso del dominio de estudio cuadrado, esto es similar, solo que se le añaden simetrías diagonales, las cuales simplifican en mayor medida la cantidad de ecuaciones a resolver, como puede verse en la imagen (4.6)

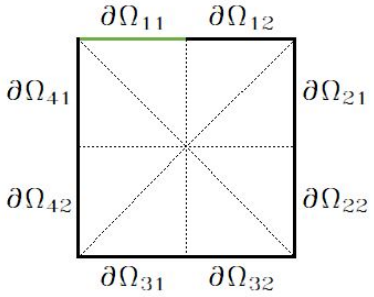

Figura 4.6: Representación de las simetrías existentes en un dominio de estudio cuadrado

Donde se reducen los dominios calculados de $\partial\Omega_{12}$ y $\partial\Omega_{21}$ a únicamente el subcontorno $\partial\Omega_{12}$.

A continuación se definen las operaciones necesarias para realizar estas simetrías, tanto para el dominio cuadrado como para el dominio rectangular.

4.1.1. Dominio rectangular

Como ya se comentó previamente, las soluciones de las variables homogéneas que se calcularan en este dominio son las de los subcontornos $\partial\Omega_{12}$ y $\partial\Omega_{21}$, correspondientes a $k \in \mathbb{Z} \cap [1, (N_z-1)/2]$ y $k \in \mathbb{Z} \cap [N_z - 1, N_z - 1 + (N_y - 1)/2]$. Por lo que las simetrías se pueden expresar como

$$\partial\Omega_{12} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} J((\partial_y u_k)\partial\Omega_1) \\ J((\partial_y u_k)\partial\Omega_3) \\ -(\partial_z u_k)\partial\Omega_4 \\ -(\partial_z u_k)\partial\Omega_2 \end{bmatrix}; \quad \partial\Omega_{22} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} -J((\partial_z u_k)\partial\Omega_4) \\ -J((\partial_z u_k)\partial\Omega_2) \\ -J((\partial_y u_k)\partial\Omega_3) \\ -J((\partial_y u_k)\partial\Omega_1) \end{bmatrix}$$

$$\partial\Omega_{31} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_2 \\ (\partial_z u_j)\partial\Omega_4 \end{bmatrix} = \begin{bmatrix} -(\partial_y u_k)\partial\Omega_3 \\ -(\partial_y u_k)\partial\Omega_1 \\ J((\partial_z u_k)\partial\Omega_2) \\ J((\partial_z u_k)\partial\Omega_4) \end{bmatrix}; \quad \partial\Omega_{32} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} -J((\partial_y u_k)\partial\Omega_3) \\ -J((\partial_y u_k)\partial\Omega_1) \\ -J((\partial_z u_k)\partial\Omega_2) \\ -J((\partial_z u_k)\partial\Omega_4) \end{bmatrix}$$

$$\partial\Omega_{41} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} J((\partial_y u_k)\partial\Omega_1) \\ J((\partial_y u_k)\partial\Omega_3) \\ -(\partial_z u_k)\partial\Omega_2 \\ -(\partial_z u_k)\partial\Omega_4 \end{bmatrix}; \quad \partial\Omega_{42} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} -J((\partial_y u_k)\partial\Omega_3) \\ -J((\partial_y u_k)\partial\Omega_1) \\ -J((\partial_z u_k)\partial\Omega_2) \\ -J((\partial_z u_k)\partial\Omega_4) \end{bmatrix}$$

Siendo k los dominios arriba descritos, usándose $k \in \mathbb{Z} \cap [1, (N_z - 1)/2]$ para los contornos $\partial\Omega_1$ y $\partial\Omega_3$ y $k \in \mathbb{Z} \cap [N_z - 1, N_z - 1 + (N_y - 1)/2]$ para los contornos $\partial\Omega_2$ y $\partial\Omega_4$, siendo j el domino del $\partial\Omega_{ij}$ a estudiar y siendo J la matriz cuadrada que equivale a $J(\mathbf{u}) = J\mathbf{u}$, la cual tiene la forma:

$$J = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}. \tag{4,84}$$

4.1.2. Dominio cuadrado

En el caso del dominio de tipo cuadrado presentamos únicamente el intervalo $k \in \mathbb{Z} \cap [1, (N_z - 1)/2]$, ya que solamente trabajaremos con la simetría de este dominio.

Las simetrías presentes en este caso son:

$$\partial\Omega_{12} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} J((\partial_y u_k)\partial\Omega_1) \\ J((\partial_y u_k)\partial\Omega_3) \\ -(\partial_z u_k)\partial\Omega_2 \\ -(\partial_z u_k)\partial\Omega_4 \end{bmatrix}$$

$$\partial\Omega_{21} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} J((\partial_z u_k)\partial\Omega_4) \\ J((\partial_z u_k)\partial\Omega_2) \\ -(\partial_y u_k)\partial\Omega_1 \\ -(\partial_y u_k)\partial\Omega_3 \end{bmatrix}; \quad \partial\Omega_{22} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} -J((\partial_y u_k)\partial\Omega_3) \\ -J((\partial_y u_k)\partial\Omega_1) \\ -J((\partial_z u_k)\partial\Omega_2) \\ -J((\partial_z u_k)\partial\Omega_4) \end{bmatrix}$$

$$\partial\Omega_{31} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_2 \\ (\partial_z u_j)\partial\Omega_4 \end{bmatrix} = \begin{bmatrix} -(\partial_y u_k)\partial\Omega_3 \\ -(\partial_y u_k)\partial\Omega_1 \\ J((\partial_z u_k)\partial\Omega_4) \\ J((\partial_z u_k)\partial\Omega_2) \end{bmatrix}; \quad \partial\Omega_{32} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} -J((\partial_y u_k)\partial\Omega_3) \\ -J((\partial_y u_k)\partial\Omega_1) \\ -J((\partial_z u_k)\partial\Omega_2) \\ -J((\partial_z u_k)\partial\Omega_4) \end{bmatrix}$$

$$\partial\Omega_{41} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \\ (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \end{bmatrix}; \quad \partial\Omega_{42} = \begin{bmatrix} (\partial_y u_j)\partial\Omega_1 \\ (\partial_y u_j)\partial\Omega_3 \\ (\partial_z u_j)\partial\Omega_4 \\ (\partial_z u_j)\partial\Omega_2 \end{bmatrix} = \begin{bmatrix} -(\partial_z u_k)\partial\Omega_2 \\ -(\partial_z u_k)\partial\Omega_4 \\ J((\partial_y u_k)\partial\Omega_1) \\ J((\partial_y u_k)\partial\Omega_3) \end{bmatrix}$$

Teniendo en cuenta todas las simetrías aplicadas, la carga computacional del problema se ha visto reducida en gran medida, si bien, debido al gran tamaño del mismo es necesario aplicar otras técnicas para poder abordar la resolución de este problema en un tiempo práctico y para un dominio con un tamaño de interés. Es por ello que se ha considerado de principal importancia la paralelización del código.

5. Paralelización del código

Como se ha comentado en el apartado anterior, el tamaño y la complejidad del problema ha hecho que sea necesario buscar distintos métodos para hacerlo más abarcable. Estos van desde el desarrollo de técnicas de CFD que permiten unas simulaciones más asequibles a la aplicación de condiciones de simetría en el dominio del problema, pero quizás una de las más útiles a la hora de hacer más abordable en este proyecto ha sido la paralelización del código.

Este proyecto partía de un código previo, el cual estaba desarrollado para su aplicación en serie, esto hizo que llegado un punto del desarrollo en el que era necesario realizar pruebas de mayor tamaño y complejidad el código precisase de al rededor de *6-7 horas* para poder calcular el primer subpaso del método de *Runge-Kutta*. Esto para un código que pretende calcular varios miles de pasos temporales resulta completamente inviable.

En el siguiente apartado se explicará someramente en qué consiste la paralelización de un código en *Fortran*, así como los métodos empleados para la misma y las ventajas que esto nos ha reportado.

5.1. Programación en serie y paralelo

Desde 1986 hasta 2003, el rendimiento de los microprocesadores ha aumentado en torno a un 50 % cada año [13]. Este aumento sin precedentes se vio detenido en 2003, momento a partir del cual la mejora anual del rendimiento de los monoprocesadores ha ido ralentizandose, llegando a disminuir a menos de un 4 % anual entre 2015 y 2017 [13]. Esta diferencia en el aumento de rendimiento anual ha provocado un cambio dramático, pasando del aumento de un 50 % anual, con el cual los rendimientos se multiplicarían casi por 60 en un periodo de 10 años, a un 4 %, con lo que el incremento sería de un 1,5.

Esta diferencia del aumento del rendimiento puede asociarse a un importante cambio en el diseño de los procesadores. A partir del 2005, los principales proveedores de microprocesadores decidieron que la forma más eficaz de aumentar el rendimiento de los mismos era a través del *paralelismo*, abandonando la idea de desarrollar monoprocesadores cada vez más rápidos, los fabricantes optaron por integrar múltiplices procesadores completos en un mismo circuito.

Esto causó un importante cambio a la hora de desarrollar software, ya que añadir más procesadores y por tanto mayor rendimiento no genera ningún impacto en la gran mayoría de programas en *serie*, es decir, programas que han sido diseñados para ejecutarse en un único procesador a la vez, debido a que estos programas no son conscientes de la existencia de múltiples procesadores, haciendo que su rendimiento sea el mismo que en un único procesador.

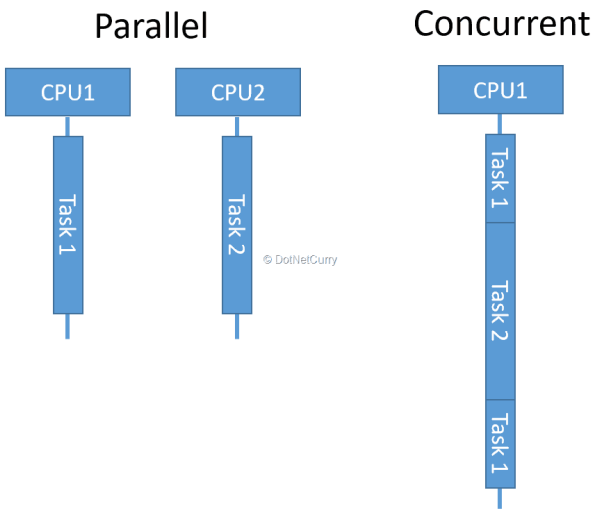

Figura 5.7: Representación esquemática de un proceso ejecutado de forma serial y de forma paralela

Esto plantea ciertas preguntas de por que es necesario el desarrollo de procesos en paralelo[14]:

- **¿Es necesario un aumento constante del rendimiento?**

  El aumento de la potencia de cálculo que se vio durante tanto tiempo fue el causante de muchos de los avances más importantes en multitud de campos. Avances tales como la decodificación del genoma humano, imágenes médicas cada vez más precisas o el procesamiento de imágenes cosmológicas están indudablemente ligados al gran aumento en los procesos de cálculo computacional.
  Es por ello que, a medida que aumenta la potencia de cálculo disponible, también aumenta el número de problemas que podemos considerar seriamente para resolver, problemas que anteriormente resultaban inabarcables, tales como la modelización del clima, el plegamiento de proteínas, descubrimiento de nuevos fármacos, *data analysis*,...

- **¿Es necesario a los sistemas en paralelo?**

  Gran parte del aumento en rendimiento de los monoprocesadores ha sido debido a la disminución en la densidad de los transistores de los circuitos. A medida que el tamaño de los transistores disminuye, su velocidad aumente y con ella la velocidad global del sistema, sin embargo, este aumento de la velocidad de los transistores conlleva a su vez un aumento del consumo de energía, la cual en gran parte es disipada como calor, causando un aumento de la temperatura del procesador que puede hacer que este deje de ser fiable. En la primera década del siglo XXI los sistemas de refrigeración por aire alcanzaron su límite de disipación [13].
  Esto se ve reflejado en como el aumento en la densidad de los transistores en estos últimos años se ha ralentizado drásticamente [15].
  Teniendo esto en cuenta, y sabiendo la necesidad imperante de aumentos en la capacidad de cálculo, la solución inmediata son los sistemas en paralelo, colocando varios procesadores completos en un único chip.

- **¿Es posible convertir programas seriales directamente en programas en paralelo?**

  La gran mayoría de los programas desarrollados para sistemas *single-core* no pueden aprovecharse de la presencia de múltiples núcleos, ya que si bien es posible ejecutar diversas partes del código en los distintos núcleos, esto no es lo que realmente se busca con la paralelización. El objetivo no es dividir un programa serial de forma que corra en varios núcleos a la vez, sino desarrollar un programa capaz de aprovechar al máximo la potencia prestada por todos los núcleos.
  Este proceso no resulta trivial, ya que aplicaciones sencillas en programación serial, como el cálculo de la traspuesta de una matriz o la multiplicación de dos matrices $n \times n$, pueden resultar tremendamente contraintuitivas en programación paralela. Es por esto que una implementación eficiente de un programa en serie puede implicar el desarrollo de un código completamente nuevo, en lugar de la paralelización de los distintos pasos del código original.

5.2. Estrategia de paralelización

La estrategia de paralelización adoptada para nuestro código ha sido una de arquitectura *Master-Slave*, en la cual se han dividido los procesadores a usar en un procesador *master*, el cual está encargado del tratamiento de los datos y de la

escritura de los mismos.

Con esto en mente, se ha desarrollado una división basada en planos paralelos al plano yz, método utilizado en casos similares al nuestro, con la diferencia de la existencia de dos direcciones periódicas en lugar de una [2]. El dominio de estudio consiste un total de N_x planos perpendiculares al plano yz, los cuales se repartirán entre todos los procesadores con los que se esté trabajando mediante la siguiente fórmula:

$$pb_i = \frac{N_x}{N_p} i; \qquad i = 0, 1, \ldots, N_p - 1 \tag{5,85}$$

$$pe_i = \frac{N_x}{N_p} (i + 1) - 1; \qquad i = 0, 1, \ldots, N_p - 1 \tag{5,86}$$

donde pb_i es el número del primer plano de trabajo de cada procesador i, pe_i es el último plano de trabajo del procesador i, N_x es el número de puntos en la dirección x, N_p es el número de procesadores.

De esta manera, es posible dividir de forma relativamente fácil la cantidad de planos con los que va a trabajar cada procesador.

Dado que los planos de trabajo, al estar contenidos en las direcciones yz, presentan paredes en todo el contorno, es necesario que cada procesador realice el cálculo de CFD para los planos que le correspondan, distribuyéndose así el mayor trabajo de cálculo entre todos los procesadores usados.

Esto, si bien reduce el tiempo de computación de forma drástica, presenta una importante limitación, ya que es imposible utilizar más procesadores en el cálculo en paralelo que el número de puntos en los que está dividida la dirección x, por lo que $N_p \leq N_x$, además de que siempre se fijarán unos N_x y N_p que sean una potencia de 2, de forma que siempre sean divisibles entre sí y todos los procesadores tengan igual carga de trabajo.

Para llevar a cabo la paralelización del código, se ha optado por la librería de *Fortran* MPI (*Message Passing Interface*), ya que en un primer lugar se optó por la implementación de un modelos de paralelización híbrido, basado en OpenMP y MPI, pero la primera de estas librerías entraba en conflicto con diferentes partes

de nuestro código, decidiéndose finalmente por el uso de MPI.

5.3. Message Passing Interface (MPI)

MPI consiste en una librería de rutinas que puede usarse para desarrollar programas en paralelo en Fortran. Esta librería tuvo su desarrollo en 1991, lanzándose su versión su primera versión en 1994 gracias a la colaboración de investigadores de distintas universidades de Estados Unidos y Europa junto con las principales empresas de computación (tales como IBM, Intel, TMC, Cray, Convex,...).

MPI es un protocolo de comunicación el cual permite desarrollar comunicaciones tanto punto a punto como colectivas, permitiendo así una gran versatilidad.

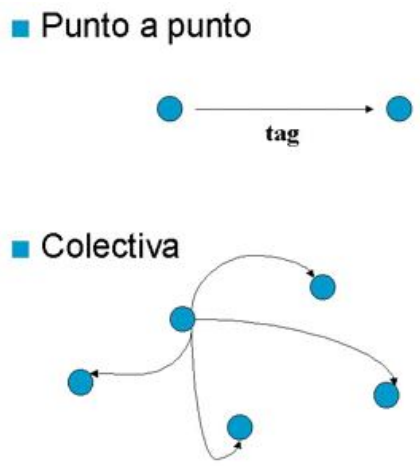

Figura 5.8: Esquemas de comunicación punto a punto y colectiva

Esto permite, con una librería no muy extensa de funciones, realizar todas las comunicaciones necesarias para llevar a cabo la paralelización de nuestro código, usando los cuatro movimientos colectivos de datos entre procesadores principales [16]:

- **Broadcast**
- **Gather**
- **Scather**
- **All gather**

A mayores de estos es de gran interés recurrir a rutinas de computación global como pueden ser *MPI_Reduce* o *MPI_AllReduce*, las cuales nos permiten combinar datos de todos los procesadores para devolver el resultado de las operaciones que se les desee hacer a un único procesador.

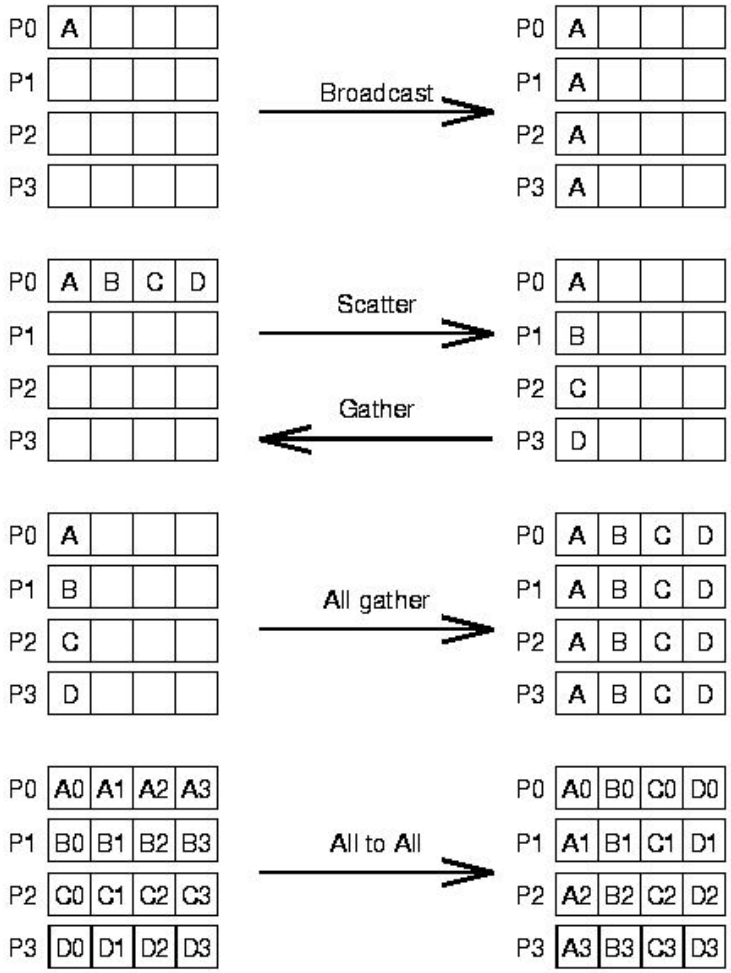

Figura 5.9: Tipos de movimientos colectivos de datos en el MPI

5.4. Beneficios de la paralelización

Las ventajas presentes al implementar la paralelización del código han sido claras desde el primer momento. Si bien esta paralelización ha resultado compleja, debido al gran tamaño del código original, ha permitido reducir los tiempos de computación de manera más que notable, pasando de tiempos del orden de *15 horas* para el cálculo de 10 pasos temporales, en un dominio de tamaño $(128, 151, 151)$ a tiempos inferiores a *30 minutos* para el mismo problema.

A continuación se presenta un gráfico de la evolución del tiempo de computación para una malla de menor tamaño con únicamente un paso temporal. Esto se hace con la intención de mostrar el ahorro de tiempo disponible gracias a la paralelización y la evolución asintótica del mismo.

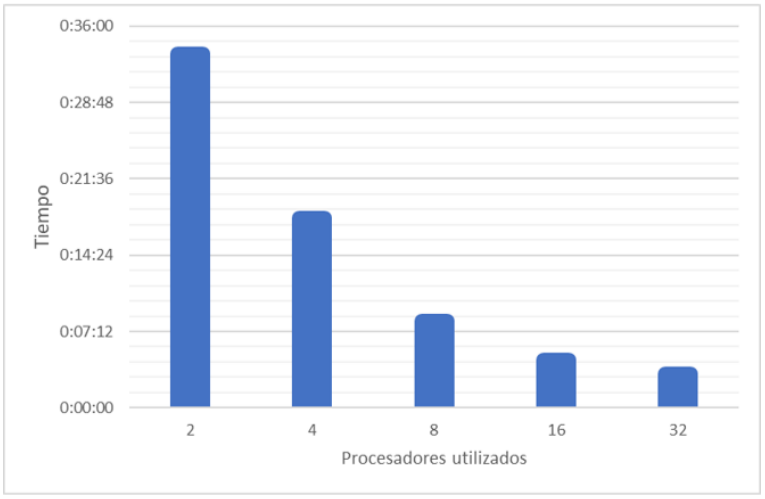

Figura 5.10: Evolución del tiempo de computación con respecto al número de procesadores

Como puede verse se presenta un ritmo de evolución lineal con los primeros aumentos del número de procesadores hasta llegar a un punto asintótico presente al pasar de 16 a 32 procesadores. Esto es debido a que el tiempo del código puede dividirse en tiempo de cálculo estándar y tiempo de comunicación. El tiempo de cálculo, siendo este el tiempo que tarda el código en realizar los cálculos de la simulación, se ve disminuido siempre que se aumenta el número de procesadores (como puede apreciarse en la evolución de 2 a 16 procesadores), sin embargo, el tiempo de comunicación, él cual representa el tiempo que tardan en comunicarse los procesadores entre sí durante el cálculo, aumenta según se va incrementando

el número de procesadores.

Esto da lugar a un comportamiento asintótico como el visto en la imagen, llegando a un punto en el que el tiempo de comunicación compensa el ahorro de tiempo de cálculo al introducir más procesadores, por lo que este aumento de procesadores deja de ser interesante.

Es importante comentar que para el caso mostrado el límite se presenta en 32 procesadores, siendo este un valor muy bajo debido al tamaño del problema seleccionado. Este caso de estudio no tiene mayor interés a parte del de ilustrar la tendencia del ahorro de tiempo de computación dada por la paralelización del código. En el caso los problemas que se pretenden afrontar con este código, teniendo estos un $Re = 4{,}000$, se ha planteado el lanzamiento en paralelo de varios cientos de procesadores, ya que el tiempo de computación total de estas simulaciones presenta un tiempo de cálculo tan elevado que no es necesario preocuparnos por llegar al límite del comportamiento asintótico.

Esta mejora ha sido posible gracias a disponer de un supercomputador de 64 procesadores, permitiéndonos tanto realizar simulaciones a gran velocidad, gracias a lo cual se ha podido depurar el código de errores en mucho menor tiempo, como realizar simulaciones de dominios y Re considerados inabarcables para códigos en serie.

6. Memoria de las variables

Una de los factores más importantes del código es la gran cantidad de memoria que es necesario utilizar para el mismo. Esta memoria viene impuesta en por la gran cantidad de variables y *work arrays* que son definidas en el documento *Mod.F90*, documento en el cual se allocatea la gran mayoría de la memoria necesaria en el código.

En los apartados siguientes se realizará un estudio de la memoria requerida por el código, tanto en variables como auxiliar, así como el proceso del grabado de los resultados a disco.

6.1. Memoria de las variables

A continuación se exponen las distintos módulos usados por el código así como las variables de los mismos y la memoria correspondiente a cada uno. Estos módulos están contenidos en el archivo *Mod.F90*, el cual tiene como objetivo allocatear las variables principales del código.

Este estudio de memoria se ha realizado para un dominio de tamaño:

$$mx = 128; \quad my = 151; \quad mz = 151;$$

El cual ha generado de un tamaño global de memoria allocateada por el archivo *Mod.F90* de unos $304,66$ MB aproximadamente, divididos entre los siguientes módulos:

- **ctes3D**: Contiene los parámetros de malla
- **alloc_dns**: Contiene todos los números adimensionales del código junto con la malla en las direcciones y y z
- **FFTW3**: Contiene variables relacionadas con el FFT
- **wave**: Aquí se guardan los números de onda
- **runge**: Valores de las variables del método R-K
- **point**: Contiene las variables para paralelización por puntos

- **statistics**: Contiene las estadísticas a estudiar del código (velocidades, vorticidades,...)
- **cfdiff**: Contiene parte de las variables del CFD
- **dealloaCfdiff**: Contiene parte de las variables del CFD
- **cfdiff1D**: Contiene parte de las variables del CFD

Para realizar la evaluación de la memoria usada por el código se han recopilado todas las variables de los distintos módulos, tanto sus dimensiones como los tipos de las mismas. De esta manera es posible realizar un estudio detallado de la memoria ocupada por cada módulo, así como de las variables de más interés dentro de ellos.

Figura 6.11: Distribución de la memoria total requerida en *Mod.F90* en función de sus distintos módulos

Como puede verse en la imagen (6.11), la cantidad de memoria requerida por *Mod.F90* viene en su mayoría únicamente de 3 subrutinas: *dealloaCfdiff*, *cfdiff* y *statistics*, agrupándo el resto de módulos en la componente *Otros*.

En porcentaje de memoria total estos módulos ocupan un:

- ***dealloaCfdiff***: 71,31 %
- ***cfdiff***: 27,74 %

- ***statistics***: 0.9 %

Por lo que el resto de módulos en conjunto contribuyen en menos de un $0,5\,\%$ al valor de la memoria requerida total.

La diferencia de tamaño entre estos módulos viene vinculada al hecho de que estos crecen con el tamaño de la malla, de forma que para la malla estudiada estos presentan unas exigencias de memoria mucho mayores que las del resto de módulos, los cuales en gran parte presentan valores independientes del tamaño de malla. Aun así, estos 3 módulos siempre van a representar más de un 97 % de la memoria del código, ya que contienen las variables más pesadas.

A continuación se modificarán distintos parámetros para que se vea como evolucionan:

- En primer lugar se compara la distribución de la memoria en las diferentes subrutinas para un caso en el que no se modifique el tamaño de maya, pero se disminuya el número de procesadores a $np = 8$. Para este caso, la memoria total empleada por *Mod.F90* es de 711,50 MB, con un aumento de un 133,5 % en la memoria requerida respecto al caso base. Puede verse como ha habido un aumento del porcentaje de memoria total requerida por el módulo *dealloaCfdiff*.

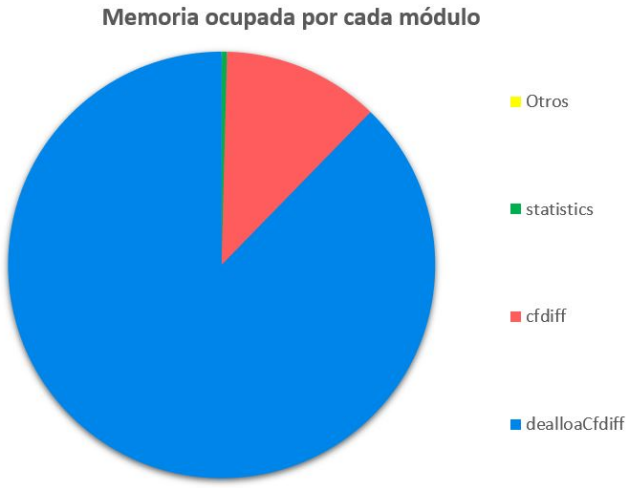

Figura 6.12: Distribución de la memoria total para un caso de *nx=128, ny=151, nz=151* y 8 procesadores

- A continuación se ha realizado una modificación de la malla base, disminuyendo su tamaño a *nx=64, ny=101, nz=101* y manteniendo fijo el número de procesadores. En este caso la memoria requerida por el código es de 99,84 MB, con una disminución de un 67 % de la memoria requerida para una reducción de casi un 80 % del tamaño de la malla. Puede verse una disminución del porcentaje de memoria requería da por la función *dealloaCfdiff*.

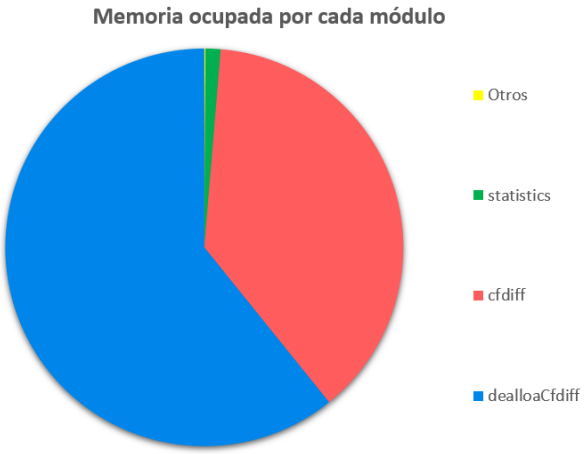

Figura 6.13: Distribución de la memoria total para un caso de *nx=64, ny=101, nz=101* y 32 procesadores

A continuación se hará un estudio de las principales variables de los dos módulos de mayor interés, *dealloaCfdiff* y *cfdiff*.

- *dealloaCfdiff*: Este módulo contiene distintas variables usadas en el CFD, con un tamaño para el caso base es de 217,26 MB. La mayoría de la memoria allocateada en este módulo viene de las variables *phi homogéneas* (ϕ_h), velocidades homogéneas (v_h) y diferentes variables auxiliares para el cálculo de las dos anteriores. Estas variables son fuertemente dependientes del tamaño de malla, concordando con lo presentado en la imagen (6.13), la cual representa la disminución del porcentaje de memoria del módulo *dealloaCfdiff* con la disminución del tamaño de malla. Por otro lado, dado que estas variables se han dividido de forma equivalente entre los diferentes procesadores, esto crea una dependencia directa con el número de procesadores.

- *cfdiff*: Este módulo contiene las matrices necesarias para la aplicación del método CFD ($A_y, A_z, B_y, B_z, G_{12}, \ldots$), tanto en sus versiones densas como dispersas. En este caso, todas estas matrices son dependientes únicamente

de las variables $n_y, n_z, nStencil$, por lo que a efectos prácticos la memoria requerida por este módulo es dependiente únicamente del tamaño de la malla elegido, ya que $nStencil$ toma un valor fijo de 5.

6.2. Memoria para $Re_\tau = 4{,}000$

A continuación se muestra el estudio de memoria realizado para el caso de simulaciones de números de Reynolds superiores al estado del arte actual. En este caso pasando de un $Re_\tau = 2000$ a un $Re_\tau = 4000$.

Para esto, se ha calculado del tamaño de malla necesario para poder realizar esta simulación con la precisión deseada, dando lugar a una malla de tamaño $(3072 \times 1101 \times 1101)$. El gran tamaño de esta malla hacer que sea inviable la simulación de la misma en un ordenador personal de sobre mesa o incluso en supercomputadores prestados por la universidad, es por ello que se han pedido un total de *35 millones de horas de cálculo* al SuperMUC, el supercomputador presente en el Centro Leibniz de Supercomputación, localizado en Alemania. Este supercomputador fue situado en el noveno puesto de supercomputadores más potentes del mundo en el pasado año 2020, por lo que la utilización del mismo es de vital importancia para el cálculo de problemas correspondientes a la nueva física.

Las características de hardware de este supercomputador son:

ComputeNodes	Thin Nodes	Fat Nodes	Total (Thin + Fat)
Processor	Intel Skylake Xeon Platinum 8174	Intel Skylake Xeon Platinum 8174	Intel Skylake Xeon Platinum 8174
Cores per Node	48	48	48
Memory per node (GByte)	96	768	NA
Number of Nodes	6,336	144	6,480
Number of Cores	304,128	6,912	311,040
PEAK @ nominal (PFlop/s)	26.3	0.6	26.9
Linpack (Pflop/s)	TBD	TBD	19.476
Memory (TByte)	608	111	719
Number of Islands	8	1	9
Nodes per Island	792	144	NA

Figura 6.14: Datos de hardware del supercomputador SuperMUC [17]

Siendo estas las características límite de las que disponemos, por lo que se ha realizado un estudio de memoria sobre estos valores para ver qué distribución de procesadores y de malla sería posible.

6.2.1. Memoria necesaria para el código original

Se ha realizado un estudio y reescalado de la memoria del dominio de simulación con el objetivo de modificar la memoria necesaria para este de forma que sea la menor posible.

Con la memoria necesaria por el código es posible calcular el número de procesadores mínimos que será necesario lanzar en paralelo en SuperMUC para realizar la simulación.

Tras realizar distintas aproximaciones se llegó a la conclusión de el mínimo número de procesadores que es necesario tomar del SuperMUC para poder almacenar toda la memoria de estas variables sería de 256 nodos, dando esto una memoria necesaria de 73,30 GB por procesador. Esta cantidad de nodos corresponde a un total de 12288 procesadores disponibles, teniendo una distribución de memoria en los módulos de la siguiente forma forma:

Figura 6.15: Distribución de la memoria total para el caso de simulación en SuperMUC.

Si bien 256 nodos es un valor que nos puede proporcionar SuperMUC, se considera bastante elevado, ya que este supercomputador debe dedicarse a otras funcio-

nes a mayores de calcular nuestra simulación. Con esto en mente, se ha desarrollado una mejora en la distribución de la memoria respecto al código original, la cual se esperaba que mejorase sustancialmente estos valores.

6.2.2. Memoria necesaria tras la mejora

A la hora de mejorar la memoria necesaria por el código se ha apuntado al módulo *dealloaCfdiff*, ya que este es el que presenta la mayor parte de la memoria necesaria, en especial la variable *phih*, encargada de almacenar las soluciones homogéneas de la variable ϕ, como puede verse en (4,75). Esta variable representaba alrededor de un 80 % de la memoria necesaria por el código, por lo que es especialmente crítico su almacenamiento correcto.

Con esto en mente, la solución que se ha adoptado ha sido la modificación del código, de forma que, respetando la física del problema. Es importante recordar que la forma que presentan las soluciones homogéneas es

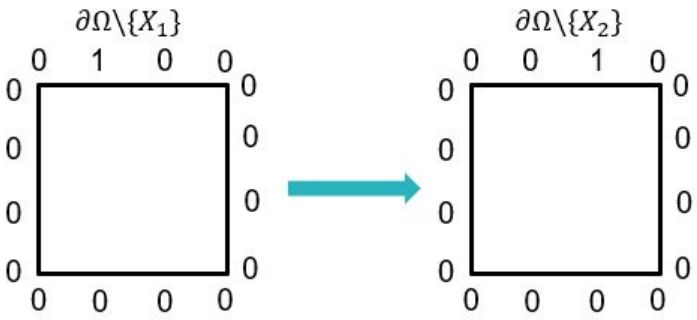

Figura 6.16: Forma que adoptan las soluciones ϕ_h a lo largo del contorno del dominio.

Por lo que cuando se empiece a multiplicar las soluciones homogéneas por las ponderaciones c presentes en la ecuación (4,77) y a superponerlas, las condiciones de contorno resultantes al obtener ϕ serán directamente los valores de las ponderaciones c.

Con esto en mente, es posible dejar de lado el allocateo de todas las matrices ϕ_h en paralelo, ya que solo será necesario trabajar con las variables de ponderación c, llevando esta parte del código a un cálculo serial o semiserial. Si bien esto disminuiría levemente la velocidad del código, genera un ahorro de memoria tan

sustancial que hace que esta solución sea más que recomendable, eliminando así la variable *phih* anterior y añadiendo una única variable *phih* para guardar sobre ella los cálculos de ϕ_h en serie, la cual tendría un tamaño despreciable con respecto al tamaño global con el que se está trabajando.

Con este cambio, la distribución de memoria del código tiene la siguiente forma:

Figura 6.17: Distribución de la memoria total tras la modificación.

Donde el número total de nodos necesarios para realizar este cálculo ha pasado de 256 a 32, una disminución de más del 87 % con respecto al código original tras implementar esta mejora. La memoria necesaria por nodo sería de 51,08 GB.

Esta enorme disminución permite correr el código con solvencia en un total de 1536 procesadores en paralelo, por lo que se podría lanzar sin mayor problema en SuperMUC junto a otros proyectos.

Esta mejora de la memoria del código junto con la mejora en el tiempo de computación dada por la paralelización son las que han hecho posible el plantearse problemas de nueva física como el caso de $Re_\tau = 4{,}000$.

7. Conclusiones

Este proyecto ha seguido los objetivos propuestos previamente en (1.2) de forma que se ha realizado un análisis del código LISO de Fortran, así como del código para la aplicación del esquema de CFD 2D. Estos dos códigos han sido modificados y depurados para adaptarlos al caso estudiado en este proyecto.

Para conseguir esto ha sido necesario modificar el código LISO en su totalidad, de forma que este permitiese aplicar CFD en dos de las tres direcciones del flujo. Para esto ha sido necesario analizar las más de 10.000 líneas de código que presenta LISO y modificarlas en consecuencia.

La perspectiva inicial de este proyecto era la obtención de un código capaz de realizar simulaciones de mayor o similar tamaño a las del estado del arte actual. Dado que esto solo es posible mediante el uso de un supercomputador con cálculo en paralelo se ha hecho especial hincapié en la paralelización completa del código, permitiendo así una mejora más que notable de los tiempos de cálculo y la simulación de dominios fuera del alcance de cualquier código en serie. Esta paralelización se ha hecho tanto del código LISO modificado como del esquema de CFD 2D mediante la implementación de MPI.

Tras esto, se han realizado diversos estudios de memoria, evaluando el porcentaje de almacenamiento necesario para cada módulo respecto al total, desarrollando e implementando un nuevo método de computación para el CDF 2D en dominios rectangulares. Esto ha permitido ahorrar casi un 87 % de los recursos computacionales que se estimaban necesarios para la simulación de un caso de $Re_\tau = 4000$ con el código de partida.

Durante todo este proceso, el código ha sido constantemente sometido a validación y depuración del mismo, habiéndose corregido la totalidad de los errores computacionales y adaptado la física detrás de este.

Finalmente, queda pendiente el lanzamiento del código en *SuperMUC*, el supercomputador del Centro Leibniz de Supercomputación, del cual disponemos de 35 millones de horas de computación para el lanzamiento de varios proyectos, este entre ellos. *SuperMUC* fue valorado como el noveno supercomputador más rápido del mundo en 2020. Con esto se pretenden obtener resultados de gran interés tanto para el estado del arte actual y como para el entendimiento del comportamiento

de los fluidos en canales turbulentos.

PRESUPUESTO

Introducción

En el siguiente apartado se desarrolla un presupuesto, permitiendo así realizar una estimación sobre el valor monetario del trabajo desempeñado en este proyecto y del coste del mismo en caso de haber sido desarrollado por una empresa privada.

Desglose del coste

A continuación se exponen las componentes del coste desglosadas de forma que estas se puedan ver más claramente.

Mano de obra

En este proyecto se consideran dos trabajadores principales:

- Autor principal. El autor principal de este trabajo es un Ingeniero Superior Junior con sueldo bruto anual de 22.000€
- Supervisor. El supervisor de este trabajo es un Doctor con más de diez años de experiencia y sueldo bruto anual de 45.000€

Se ha calculado el coste por hora asumiéndose que al año hay un total de 250 días laborales con una jornada diaria de 8 horas.

	Sueldo bruto anual	**S.S. a cargo de empresa anual**	**Total anual**	**Coste de mano de obra por hora**
Autor	22.000€	6.820€	29.820€	14.41€
Supervisor	71.755€	22.244€	94.000€	47,00€

Tabla 7.1: Cálculo del coste en de la mano de obra por hora.

Material informático

En este apartado se consideran los gastos derivados del uso de softwares así como del propio ordenador.

El gasto en softwares es de:

Descripción	Importe
Licencia anual de Matlab R2018b.	800,00€
Licencia de Microsoft Office 2018.	100,00€
LaTex.	0,00€
Eclipse	0,00€
Total	**900,00€**

Tabla 7.2: Coste del software empleado.

El coste en ordenador se ha desglosa en el uso tanto del ordenador portátil como del ordenador de sobremesa:

	Tiempo(h)	Coste total(EUR)	Importe(EUR)
Portatil	200	1300€	260€
Sobremesa	500	6000€	1.200€
		TOTAL	**1.460€**

Tabla 7.3: Coste de los dispositivos empleados.

Donde el coste total de los dispositivos es el precio de los mismos en el momento de compra. Estos dispositivos se consideran amortizados a los 5 años, por lo que al considerarse un año de trabajo de los mismos se obtiene un importe total por este año de trabajo de 1460€.

Material de oficina

El coste total del material de oficina se estima en 10€en concepto de material de oficina (libretas, bolígrafos, subrayadores, ...).

Impresión y encuadernación

El coste de cada impresión realizada del proyecto es de 30€. Considerando dos impresiones del mismo con sus encuadernaciones el coste asciende a 60€.

Costes de mano de obra

A continuación se desglosan los costes de mano de obra asociados a cada etapa del proyecto.

Documentación y familiarización con el código

	Horas(h)	Coste por hora(EUR/h)	Total(EUR)
Ingeniero	50	14,41	720,50€
Doctor	10	47,00	470.00€
TOTAL			**1.190,50€**

Tabla 7.4: Coste de mano de obra de documentación y familiarización del código.

Adaptación de LISO a 2D

	Horas(h)	Coste por hora(EUR/h)	Total(EUR)
Ingeniero	130	14,41	1.873,30€
Doctor	20	47,00	940,00€
TOTAL			**2.812,30€**

Tabla 7.5: Coste de mano de obra de adaptación de LISO a 2D.

Paralelización del código

	Horas(h)	Coste por hora(EUR/h)	Total(EUR)
Ingeniero	150	14,41	2.161,50€
Doctor	15	47,00	705,00€
TOTAL			**2.866,50€**

Tabla 7.6: Coste de mano de obra de paralelización del código.

Depuración y corrección de errores

	Horas(h)	Coste por hora(EUR/h)	Total(EUR)
Ingeniero	230	14,41	3.314,30€
Doctor	25	47,00	1175,00€
TOTAL			**4.489,30€**

Tabla 7.7: Coste de mano de obra de depuración y corrección de errores.

Elaboración de la memoria

	Horas(h)	Coste por hora(EUR/h)	Total(EUR)
Ingeniero	20	14,41	288,2€
Doctor	2	47,00	94,00€
TOTAL			**382,20€**

Tabla 7.8: Coste de mano de obra de elaboración de la memoria.

Coste global de mano de obra

Fases	Importe
Documentación y familiarización con el código	1.190,50€
Adaptación de LISO a 2D	2.813,30€
Paralelización del código	2.866,50€
Depuración y corrección de errores	4.489,30€
Elaboración de la memoria	382,20€
Total	**11.741,80€**

Tabla 7.9: Desglose de costes de mano de obra.

Presupuesto final

Finalmente se expone el presupuesto global del trabajo desglosado en conceptos, añadiendo el IVA del 21 % y el beneficio empresarial del 6 % sobre el subtotal sin IVA.

Concepto		Importe sin IVA	IVA	TOTAL
Mano de obra		11.741,80€	2.465,78€	14.207,58€
Material informático	Software	900,00€	189,00€	1.089,00€
	Ordenadores	1.460,00€	306,60€	1.766,60€
Material de oficina		10,00€	2,10€	12,10€
Impresión y encuadernación		60,00€	12,60€	72,60€
Beneficio empresarial		901,17€	189,25€	1.090,42€
Total		**15.022,11€**	**3154.64€**	**18.176,75€**

Tabla 7.10: Presupuesto final.

El presupuesto final de este proyecto asciende a DIECIOCHO MIL CIENTO SETENTA Y SEIS EUROS CON SETENTA Y CINCO CÉNTIMOS.

PLIEGO DE CONDICIONES

Condiciones del puesto de trabajo

Las condiciones de trabajo han de ser reguladas de forma que se minimicen los riesgos laborales vinculados a las mismas. Estas condiciones deben estar sometidas a la normativa recogida en el Real Decreto 488/1997 del 14 de abril [18], sobre disposiciones mínimas de seguridad y salud relativas al trabajo con equipos que incluyen pantallas de visualización (PVD). Respecto al puesto de trabajo, la definición que se da en el real decreto es 'el constituido por un equipo con pantalla de visualización provisto, en su caso, de un teclado o dispositivo de adquisición de datos, de un programa para la interconexión persona/máquina, de accesorios ofimáticos y de un asiento y mesa o superficie de trabajo, así como el entorno laboral inmediato'.

Las variables que se tendrán en cuenta a la hora de prever los diferentes riesgos a los que se puede atener el trabajador son:

- Tiempo trabajado con la pantalla de visualización.
- Tiempo de atención requerida a la pantalla, ya se continuo o discontinuo.
- Exigencias de la tarea realizada.
- Exigencias en la velocidad de obtención de la información.

A partir de esto, los riesgos posibles que experimenta el trabajador son:

- Seguridad en los contactos eléctricos.
- Iluminación.
- Ruido.
- Condiciones termohigrométricas .
- Ergonomía.
- Fatiga visual.
- Fatiga física.
- Fatiga mental.

Equipo

La utilización propia del entorno no debe suponer nunca un riesgo para los trabajadores. A continuación, se exponen las condiciones de los distintos equipos usados según el Real Decreto 488/1997 del 14 de abril [18]

Pantalla

Los caracteres de la pantalla deberán estar bien definidos y configurados de forma clara, y tener una dimensión suficiente, disponiendo de un espacio adecuado entre los caracteres y los renglones.

La imagen de la pantalla deberá ser estable, sin fenómenos de destellos, centelleos u otras formas de inestabilidad.

El usuario de terminales con pantalla deberá poder ajustar fácilmente la luminosidad y el contraste entre los caracteres y el fondo de la pantalla, y adaptarlos fácilmente a las condiciones del entorno.

La pantalla deberá ser orientable e inclinable a voluntad, con facilidad para adaptarse a las necesidades del usuario.

Podrá utilizarse un pedestal independiente o una mesa regulable para la pantalla.

La pantalla no deberá tener reflejos ni reverberaciones que puedan molestar al usuario.

Teclado

El teclado deberá ser inclinable e independiente de la pantalla para permitir que el trabajador adopte una postura cómoda que no provoque cansancio en los brazos o las manos.

Tendrá que haber espacio suficiente delante del teclado para que el usuario pueda apoyar los brazos y las manos.

La superficie del teclado deberá ser mate para evitar los reflejos.

La disposición del teclado y las características de las teclas deberán tender a facilitar su utilización.

Los símbolos de las teclas deberán resaltar suficientemente y ser legibles desde la posición normal de trabajo.

Mesa o superficie de trabajo

La mesa o superficie de trabajo deberán ser poco reflectantes, tener dimensiones suficientes y permitir una colocación flexible de la pantalla, del teclado, de los documentos y del material accesorio.

El soporte de los documentos deberá ser estable y regulable y estará colocado de tal modo que se reduzcan al mínimo los movimientos incómodos de la cabeza y los ojos.

El espacio deberá ser suficiente para permitir a los trabajadores una posición cómoda.

Asiento de trabajo

El asiento de trabajo deberá ser estable, proporcionando al usuario libertad de movimiento y procurándole una postura confortable. La altura del mismo deberá ser regulable.

El respaldo deberá ser reclinable y su altura ajustable.

Se pondrá un reposapiés a disposición de quienes lo deseen.

Lugar de trabajo

El entorno de trabajo debe cumplir las condiciones recogidas en el Real Decreto 486/1997 del 14 de abril [19] sobre condiciones mínimas de seguridad y salud.

Vías y salidas de evacuación

La empresa debe haber adoptado medidas de emergencia en las que se incluyan las vías y salidas de evacuación en caso de que se declare una emergencia. Las vías y salidas de evacuación deberán permanecer expeditas y desembocar lo más directamente posible en el exterior o en una zona de seguridad. Estas medidas deben darse a conocer a los trabajadores.

Condiciones de protección contra incendios

Los lugares de trabajo deberán ajustarse a lo dispuesto en la normativa que resulte de aplicación sobre condiciones de protección contra incendios.

En todo caso, y a salvo de disposiciones específicas de la normativa citada, dichos lugares deberán satisfacer las condiciones que se señalan en los siguientes puntos de este apartado.

Según las dimensiones y el uso de los edificios, los equipos, las características físicas y químicas de las sustancias existentes, así como el número máximo de personas que puedan estar presentes, los lugares de trabajo deberán estar equipados con dispositivos adecuados para combatir los incendios y, si fuere necesario, con detectores contra incendios y sistemas de alarma.

Los dispositivos no automáticos de lucha contra los incendios deberán ser de fácil acceso y manipulación. Dichos dispositivos deberán señalizarse conforme a lo dispuesto en el Real Decreto 485/1997, de 14 de abril [20], sobre disposiciones mínimas de señalización de seguridad y salud en el trabajo. Dicha señalización deberá fijarse en los lugares adecuados y ser duradera.

Instalaciones eléctricas

La instalación eléctrica de los lugares de trabajo deberá ajustarse a lo dispuesto en su normativa específica.

En todo caso, y a salvo de disposiciones específicas de la normativa citada, dicha instalación deberá satisfacer las condiciones que se señalan en los siguientes

puntos de este apartado.

La instalación eléctrica no deberá entrañar riesgos de incendio o explosión. Los trabajadores deberán estar debidamente protegidos contra los riesgos de accidente causados por contactos directos o indirectos. La instalación eléctrica y los dispositivos de protección deberán tener en cuenta la tensión, los factores externos condicionantes y la competencia de las personas que tengan acceso a partes de la instalación.

Condiciones termohigrométricas

En los locales de trabajo cerrados deberán cumplirse, en particular, las siguientes condiciones:

- La temperatura de los locales donde se realicen trabajos sedentarios propios de oficinas o similares estará comprendida entre 17 y 27 ºC. La temperatura de los locales donde se realicen trabajos ligeros estará comprendida entre 14 y 25 ºC.
- La humedad relativa estará comprendida entre el 30 y el 70 por 100, excepto en los locales donde existan riesgos por electricidad estática en los que el límite inferior será el 50 por 100.
- Los trabajadores no deberán estar expuestos de forma frecuente o continuada a corrientes de aire cuya velocidad exceda los siguientes límites:
  - Trabajos en ambientes no calurosos: 0,25 m/s.
  - Trabajos sedentarios en ambientes calurosos: 0,5 m/s.
  - Trabajos no sedentarios en ambientes calurosos: 0,75 m/s.

Reflejos y deslumbramientos

Los puestos de trabajo deberán instalarse de tal forma que las fuentes de luz, tales como ventanas y otras aberturas, los tabiques transparentes o translúcidos y los equipos o tabiques de color claro no provoquen deslumbramiento directo ni produzcan reflejos molestos en la pantalla.

Las ventanas deberán ir equipadas con un dispositivo de cobertura adecuado y regulable para atenuar la luz del día que ilumine el puesto de trabajo.

Ruido

Los niveles de ruido vienen determinados por lo regulado en el Real Decreto 28/2006, de 10 de marzo, sobre la protección de la salud y la seguridad de los trabajadores contra los riesgos relacionados con la exposición al ruido.

En nuestro caso el rango de ruido con el que se trabaja es muy reducido, gracias a los ventiladores y disipadores de calor más silenciosos de las distintas computadoras con las que se trabaja, por lo que esto no supone un riesgo.

Emisiones

Toda radiación, excepción hecha de la parte visible del espectro electromagnético, deberá reducirse a niveles insignificantes desde el punto de vista de la protección de la seguridad y de la salud de los trabajadores.

Humedad

Deberá crearse y mantenerse una humedad aceptable.

Interconexión ordenador/persona

Para la elaboración, la elección, la compra y la modificación de programas, así como para la definición de las tareas que requieran pantallas de visualización, el empresario tendrá en cuenta los siguientes factores:

- El programa habrá de estar adaptado a la tarea que deba realizarse.
- El programa habrá de ser fácil de utilizar y deberá, en su caso, poder adaptarse al nivel de conocimientos y de experiencia del usuario; no deberá utilizarse ningún dispositivo cuantitativo o cualitativo de control sin que los trabajadores hayan sido informados y previa consulta con sus representantes.
- Los sistemas deberán proporcionar a los trabajadores indicaciones sobre su desarrollo.

- Los sistemas deberán mostrar la información en un formato y a un ritmo adaptados a los operadores.
- Los principios de ergonomía deberán aplicarse en particular al tratamiento de la información por parte de la persona.

Condiciones de los recursos informáticos

Para poder llevar a cabo este proyecto es necesario disponer de dispositivos informáticos de altas prestaciones, debido a la gran exigencia en los cálculos a realizar. Las especificaciones técnicas de los equipos usados se muestran a continuación.

Hardware

Han sido utilizados como hardware en este proyecto un equipo portátil y un ordenador de sobremesa. El ordenador portátil ha sido utilizado tanto para acceder a internet, modificar y editar documentos, realizar reuniones telemáticas, realizar cálculos con diferentes software tales como *Fortran* y *Matlab*. En el caso del ordenador de sobremesa este ha sido utilizado para depurar el código con el que se ha trabajado, así como para realizar procesos de cálculo más pesados.

Las especificaciones técnicas de estos dispositivos son:

- **Equipo portátil:**
  - CPU: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz, 2592 Mhz, 6 procesadores principales, 12 procesadores lógicos.
  - RAM: DDR IV 8GB*2 (2666MHz)
  - Tarjeta gráfica: GeForce(R) RTX 2070 Super
  - Unidad de almacenamiento: 1TB NVMe PCIe SSD
- **Ordenador de sobremesa:**
  - CPU: AMD EPYC 7281 16-Core Processor x 2
  - RAM: 125 GB
  - Tarjeta gráfica: NVIDIA Corporation GP107GL [Quadro P620]
  - Unidad de almacenamiento: Disco duro SSD y disco duro HDD

Software

Los programas usados son:

- Matlab R2020
- Fortran 90
- LaTex
- Eclipse IDE 2022-06
- Windows 10

Conexión a internet

La conexión a internet usada ha sido tanto la estándar de una vivienda como la proporcionada por la UPV en el Instituto Universitario de Matemática Pura y Aplicada (IUMPA).

Referencias

[1] Jesús Amo-Navarro y col. "Two-Dimensional Compact-Finite-Difference Schemes for Solving the bi-Laplacian Operator with Homogeneous Wall-Normal Derivatives". En: *Mathematics* 9.19 (2021), pág. 2508.

[2] Federico Lluesma-Rodrıguez y col. "A code for simulating heat transfer in turbulent channel flow". En: *Mathematics* 9.7 (2021), pág. 756.

[3] ESA. *ESA*. 2011. URL: https://www.esa.int/Space_in_Member_States/Spain/Dosier_de_Prensa_-_Lanzamiento_Galileo_IOV_y_Soyuz_- (visitado 28-05-2021).

[4] Kenneth Holmberg y Ali Erdemir. "Global impact of friction on energy consumption, economy and environment". En: *Fme Trans* 43.3 (2015), págs. 181-185.

[5] Sanjiva K Lele. "Compact finite difference schemes with spectral-like resolution". En: *Journal of computational physics* 103.1 (1992), págs. 16-42.

[6] Steven A Orszag y GS Patterson Jr. "Numerical simulation of three-dimensional homogeneous isotropic turbulence". En: *Physical Review Letters* 28.2 (1972), pág. 76.

[7] John Kim, Parviz Moin y Robert Moser. "Turbulence statistics in fully developed channel flow at low Reynolds number". En: *Journal of Fluid Mechanics* 177 (1987), págs. 133-166. DOI: 10.1017/S0022112087000892.

[8] Sergio Hoyas y Javier Jiménez. "Scaling of the velocity fluctuations in turbulent channels up to Re τ= 2003". En: *Physics of fluids* 18.1 (2006), pág. 011702.

[9] Yoshinobu Yamamoto y Yoshiyuki Tsuji. "Numerical evidence of logarithmic regions in channel flow at R e τ= 8000". En: *Physical Review Fluids* 3.1 (2018), pág. 012602.

[10] Sergio Hoyas y col. "Wall turbulence at high friction Reynolds numbers". En: *Physical Review Fluids* 7.1 (2022), pág. 014602.

[11] Philippe R Spalart, Robert D Moser y Michael M Rogers. "Spectral methods for the Navier-Stokes equations with one infinite and two periodic directions". En: *Journal of Computational Physics* 96.2 (1991), págs. 297-324.

[12] *SPAKRSEKIT2*. http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm. Accessed: 2022-06-27.

[13] David A Patterson, John L Hennessy y David Goldberg. *Computer architecture: a quantitative approach*. Vol. 2. Morgan Kaufmann San Mateo, CA, 1990.

[14] Peter Pacheco y Matthew Malensek. *An Introduction to Parallel Programming*. Morgan Kaufmann, 2021.

[15] John Loeffler. *No more transistors: The end of Moore's law*. Feb. de 2022. URL: https://interestingengineering.com/transistors-moores-law.

[16] URL: https://users.dcc.uchile.cl/~ynakanis/pagina/presenta.html.

[17] *Hardware of supermuc-ng.* URL: https://doku.lrz.de/display/PUBLIC/Hardware+of+SuperMUC-NG.

[18] *Real Decreto 488/1997 del 14 de abril.* https://www.boe.es/buscar/pdf/1997/BOE-A-1997-8671-consolidado.pdf. Accessed: 2022-07-03.

[19] *Real Decreto 486/1997 del 14 de abril.* https://www.boe.es/buscar/act.php?id=BOE-A-1997-8669. Accessed: 2022-07-03.

[20] *Real Decreto 485/1997 del 14 de abril.* https://www.boe.es/buscar/doc.php?id=BOE-A-1997-8668. Accessed: 2022-07-03.