UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Máster Oficial en Sistemas Inteligentes y Aplicaciones Numéricas en Ingeniería

Trabajo Final de Master

CONTRIBUCIÓN EN LA SIMULACIÓN NUMÉRICA DE CORRIENTES MARINAS 3D CON ELEMENTOS FINITOS EN ZONAS COSTERAS

Román Grau Beránger

Tutores: Dr. Gabriel Winter Althaus
Dra. Begoña González Landín

Gran Canaria, Diciembre 2017

CONTRIBUCIÓN EN LA SIMULACIÓN NUMÉRICA DE CORRIENTES MARINAS 3D CON ELEMENTOS FINITOS EN ZONAS COSTERAS

Román Grau Beránger
roman.grau102@alu.ulpgc.es

Diciembre 2017

A mis padres
i mojej ukochanej

Agradecimientos

Agradezco en primer lugar a mi tutor D. Gabriel Winter por creer en mi y abrirme las puertas de un ámbito tan emocionante de la ciencia. Gracias por todo el conocimiento, la paciencia, la dedicación y las oportunidades.
Mi enorme agradecimiento también a mi tutora Dña. Begoña González, por todo el trabajo conmigo, la dedicación y por recibirme siempre con una sonrisa.

Le doy también las gracias al equipo del CEANI, que estos meses me han tratado tan amablemente y en especial a Ignacio y a Iñaki, por estar siempre predispuestos a ayudarme. Sin olvidar mencionar al instituto SIANI por facilitarme el acceso a su clúster de cálculo.

Por último, lo más importante, la familia, que ha sabido entenderme y apoyarme durante toda mi etapa académica y especialmente el año de máster. Todo es tan fácil gracias a ustedes.

Estudiar este máster ha sido una experiencia enormemente enriquecedora a nivel académico y personal, conociendo a grandes mentes dentro de grandes personas y llevándolas siempre conmigo en forma de buenos amigos. También se nos han dado unos conocimientos que, en buenas manos (las nuestras incluidas), de buena cuenta contribuirán y artificiarán grandes avances. Pero lo más importante que me llevo de este curso es a mi compañera y mi amor, la que de verdad hará que este año sea inolvidable y que cada momento sea especial. Gracias, Daria.

Índice

Lista de figuras

Lista de tablas

Lista de símbolos

α	Módulo de precisión de Gauss
β	Coeficiente de expansión térmica
ε	Valor muy pequeño que tiende a cero
κ	Conductividad térmica
λ	Multiplicador de Lagrange
μ	Viscosidad dinámica
ν	Viscosidad cinemática
Ω	Volumen dominio de $\mathbb{R}^3$
ρ	Densidad
θ	Temperatura potencial
∂_{jj}	Gradiente en notación de Einstein
∂_j	Derivada parcial respecto al tiempo
$\partial\Omega$, Γ	Frontera del volumen Ω
a	Constante de la espiral de Ekman
C_o	Número de Courant
C_p	Calor específico a presión constante
d	Distancia de un punto a una estación de medida
$\mathrm{d}t$	Diferencial de tiempo
f	Función, empleada en las ecuaciones NS, normalmente de densidad y gravedad
$\mathcal{F}_i$	Fuerzas de fricción en el fluido
g	Gravedad, 9.81 m/s
L_0	Longitud característica de la turbulencia
$\mathcal{L}_i$	Distancia a la superficie libre
$\mathbf{n}$	Vector normal
p	Presión
Q	Fuente o sumidero de calor
R_e	Número de Reynolds
S	Salinidad del agua de mar según TEOS-10
T	Temperatura *in situ* o en cada punto
$\mathcal{T}_i$	Tiempo medio entre colisiones de moléculas
$\mathbf{u}$	Vector velocidad
U_0	Velocidad característica en turbulencia
U_∞	Velocidad libre, en un punto lejano sin influenciar
$\mathcal{U}_i$	Velocidad rms de una molécula
w, v_i	Funciones test de la formulación variacional
z	Profundidad
Δt	Incremento de tiempo
∇	Gradiente
$\overline{\nabla}$	Divergencia media de un campo vectorial

Capítulo 1

Introducción

Los problemas ambientales tienen un gran impacto social, económico y científico. En este trabajo final de máster se propone la utilización y mejora de una serie de herramientas numéricas enfocadas a la simulación, lo más realista posible, de campos de corrientes marinas en entornos costeros.
Las posibles aplicaciones que requieren un conocimiento del comportamiento de estas corrientes pueden ser:

- Vertidos de contaminantes: una vez se produce un vertido de algún contaminante al medio marino, es fundamental conocer el punto de origen, las características del vertido y las condiciones del lugar para poder predecir y planificar el posible impacto en zonas de interés. Un conocimiento certero y rápido del comportamiento de las corrientes puede ayudar a defender y minimizar el impacto, por ejemplo, en zonas costeras

- Energías renovables: en la actualidad se está considerando el cambio climático como uno de los problemas más importantes de nuestro planeta y uno de los factores con mayor incidencia en este fenómeno son las fuentes de energía. Su generación tiene un enorme impacto medioambiental, comprometiendo el entorno en todos los puntos del proceso, desde la extracción y transporte de la materia prima hasta su procesado y transformación. Es por ello que la búsqueda de modelos energéticos que minimicen el impacto ambiental en su producción, es clave para la sostenibilidad del modelo y la protección del medio ambiente. Este cambio de modelo debe sostenerse sobre el pilar de la fuente de energía –que sea renovable, que no sea estacional, que sea abundante, etc.– sobre el pilar de la eficiencia –que una pequeña cantidad de materia prima sea capaz de generar abundante energía– y sobre el pilar del impacto ambiental en su procesado –los residuos generados, por ejemplo, gases nocivos, humos y partículas contaminantes, materiales radiactivos, etc–
  En este sentido, una de las fuentes alternativas es el océano, cuyas corrientes, en tanto que grandes masas de agua en movimiento con enormes

inercias, pueden aprovecharse de diversas maneras.

- Emplazamientos offshore: otra tendencia del sector de las energías renovables es la instalación de parques eólicos offshore, que garantizan un flujo de viento más constante y potente, además de las ventajas obvias frente a su instalación en tierra; no precisar enormes extensiones de terreno para su explotación, casi exclusiva. Sin embargo, la instalación de grandes generadores eólicos en el mar precisa de informes de viabilidad. En este sentido existen numerosos estudios sobre el comportamiento de un fluido alrededor de un cuerpo cilíndrico

- Estudios costeros: los estudios medioambientales llevados a cabo periódicamente por diversos organismos y entidades privadas o públicas necesitan de un conocimiento preciso de la dinámica marina en entornos costeros para el estudio de la dispersión biológica, procesos de transporte y sedimentación, etc.

Por estos motivos, a parte del evidente interés científico y académico que supone, se ha decidido recopilar y trabajar sobre una serie de herramientas que nos permitan simular con precisión y en un periodo de tiempo razonable, el comportamiento de las corrientes en un entorno costero. Para ello, es capital la importancia de contar con geometrías reales, cuya concordancia con el entorno geográfico que se desea estudiar sea razonable. Esto supone:

1. Encontrar una fuente de datos fiables, tanto de batimetrías como de líneas de costa.

2. Un procedimiento eficiente y eficaz de tratamiento de estos datos geográficos, de generación CAD, de un dominio computacional realista y computacionalmente viable.

3. Resolución de las Ecuaciones en Derivadas Parciales (EDP) de la dinámica de fluidos con aproximaciones que permitan su resolución en un tiempo razonable de cálculo y ajusten los resultados a la física real del problema, siendo capaz de tener en cuenta diversos factores externos como pueden ser la línea de costa, el efecto del viento, la incorporación de mediciones reales que se tengan de la corriente in situ o la orografía del fondo marino

4. Visualización gráfica y amigable de los resultados, que tenga un carácter fiel y riguroso a la vez que divulgativo y comprensible, para la difusión y transferencia de los resultados.

1.1. Objetivos

En este trabajo se pretende contribuir al estudio de herramientas cuya aplicación final sea a en los puntos descritos en el apartado anterior. Para ello,

a continuación se relacionan los objetivos que se desean comenzar a explorar en este trabajo:

- Reproducción de geometrías que se ajusten a las zonas geográficas reales objeto de estudio
- Estudio de la dinámica de corrientes marinas en zonas costeras en tres dimensiones utilizando el método de los elementos finitos
- Generación de mallados adecuados para el método de los elementos finitos
- Acoplamiento de modelos matemáticos existentes, en concreto el modelo de matriz de masa consistente (MMC) y el algoritmo de Chorin-Rannacher, y su implementación tridimensional.
- Visualización de los resultados de manera práctica y rigurosa

1.2. Antecedentes, revisión bibliográfica y estado del arte

Cabe destacar los enormes esfuerzos y avances que desde la Universidad de Las Palmas de Gran Canaria, y en concreto desde la división de Computación Evolutiva (CEANI) del instituto SIANI, se vienen haciendo en la línea de simulación de dispersión de contaminantes en medio marino. Entre los antecedentes científicos y las referencias relacionadas con la investigación desarrollada hasta el momento por el CEANI, podemos destacar:

– La tesis doctoral de Dña. Begoña González Landín titulada: *Determinación de localizaciones óptimas de puntos múltiples de vertidos de aguas residuales en zonas costeras mediante Algoritmos Genéticos*, donde se elaboró una metodología para la resolución eficiente del problema de encontrar localizaciones óptimas para vertidos de aguas residuales en un medio acuático (lagos, estuarios o zonas marinas) mediante la utilización de algoritmos de optimización heurísticos avanzados (denominados Algoritmos Evolutivos. En esta línea de investigación, el profesor D. Blas Gálvan González en cooperación con Dña. Silvia Alonso y otros investigadores del CEANI han desarrollado un nuevo método evolutivo denominado Agente de Evolución Flexible). Asimismo, se presentó la implementación de un modelo 3D con volúmenes finitos de simulación numérica de la concentración de los 10 contaminantes más usuales en estudios de impacto medio ambiental (modelo hidro-ecológico completo).

– El desarrollo de un nuevo modelo de campo cercano y de primera fase de campo lejano para vertidos de salmuera al mar mediante emisarios submarinos, en el que se tiene en consideración la diferencia entre las presiones osmóticas de

la salmuera y del agua ambiente en la formación de los jets (chorros o penachos) financiado por la Fundación Centro Canario del Agua y la Dirección General de Universidades del Gobierno de Canarias (Diciembre, 2002 – Octubre, 2003), y presentado en el IV Congreso Nacional AEDyR 2003: "DESALACIÓN Y REUTILIZACIÓN. MIRANDO HACIA EL FUTURO", que tuvo lugar del 19 al 21 Noviembre de 2003 en el Palacio de Congresos de Gran Canaria, Institución Ferial de Canarias.

– En cuanto a sumulaciones de evolución de vertidos de hidrocarburos, se han realizado más de setenta para los Planes de Contingencia de diferentes Puertos, con datos reales (batimetría, datos de laboratorio de evaporación (Fingas) y emulsificaciones de hidrocarburos (Fingas y Fieldhouse, 2006), y los últimos avances prácticos del estado del arte de la simulación de la evolución de los vertidos de hidrocarburos en medio marino, tanto en zonas costeras y puertos, como en océanos. Cabe también destacar el "Desarrollo y aplicación de modelos y sistemas integrados Alermac y SAMM para la predicción del comportamiento de derrames superficiales de hidrocarburos en el medio marino y de los procesos de transformación del vertido" trabajo codirigido por los profesores D. Blas Galván González y D. Gabriel Winter Althaus, que recibió el 2º Premio Nacional de Investigación e Innovación Tecnológica en la lucha contra la contaminación marítima y del litoral. Premio otorgado por el Ministerio de la Presidencia (BOE número 292, por la que se hace público el fallo del Jurado, Orden Pre/3718/2006, publicado el pasado 7 de Diciembre de 2006.

En cuanto a la revisión bibliográfica, se ha prestado especial atención a publicaciones relacionadas con la mecánica de fluidos clásica y con la Dinámica de Fluidos Computacional (CFD) para los aspectos teóricos, es especial las publicaciones de los autores J.M. Fernández Oro [1] y J.H. Ferziger & M. Peric [2].

Modelos oceanográficos computacionales

En este trabajo se han tenido en cuenta implementaciones de las ecuaciones de turbulencia marina y discretizaciones numéricas previas. Cabe destacar algunos modelos de columna de agua como las versiones mono y bidimensionales de *Princeton Ocean Model* (POM, *Blumberg and Mellor, 1987*, COHERENS (*Luyten et al.*, 1999) y PROBE (*Svensson*, 1998). Con todo, el modelo más completo de la revisión es el *General Ocean Turbulence Model* (GOTM, *Buchard et al.*, 1999), implementado en FORTRAN y disponible en `www.gotm.net`.
Existe también un marco europeo de trabajo sobre el modelado oceanográfico, en sus vertientes físicas y biológicas, llamado *Nucleus for European Modelling of the Ocean* (NEMO)[1].

[1]NEMO European Consortium, www.nemo-ocean.eu

Capítulo 2

Ecuaciones de la dinámica de un fluido

En esta sección se introducen las ecuaciones de Navier-Stokes, que son válidas para prácticamente toda la casuística de flujos, así como diferentes aproximaciones, simplificaciones y métodos de cálculo.

2.1. Conceptos previos

En el estudio de la Mecánica tenemos dos enfoques para la resolución de problemas:

- Lagrangiano: donde la solución buscada es la evolución temporal de las magnitudes de todas y cada una de las partículas de sistema que intervienen en el problema. Matemáticamente, las magnitudes se consideran funciones de la posición inicial de la partícula y del tiempo. Este planteamiento es frecuentemente empleado en la mecánica del sólido elástico.
- Euleriano: en el que se calculan los valores de las magnitudes de todas las partículas del sistema que en cada instante de tiempo t están ocupando un volumen de control fijado previamente. Matemáticamente, las magnitudes que se analizan son funciones de la posición del volumen de control y del tiempo. Por su parte, este enfoque es usado para la mecánica de fluidos

A continuación se describirán brevemente algunas herramientas matemáticas necesarias para el seguimiento de los conceptos tratados en el trabajo.

2.1.1. Derivada material

La derivada material es la derivada respecto al tiempo siguiendo una partícula material a lo largo del espacio.

La aceleración de un fluido con velocidad $\mathbf{u} = u(x,y,z,t)\overrightarrow{i} + v(x,y,z,t)\overrightarrow{j} + w(x,y,z,t)\overrightarrow{k}$ viene dada por el vector:

$$\frac{D\mathbf{u}}{Dt} = \left(\frac{Du}{Dt}, \frac{Dv}{Dt}, \frac{Dw}{Dt}\right) \tag{2.1}$$

Que en notación vectorial se escribe de la forma siguiente:

$$\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = \frac{\mathrm{D}u}{\mathrm{D}t}\overrightarrow{i} + \frac{\mathrm{D}v}{\mathrm{D}t}\overrightarrow{j} + \frac{\mathrm{D}w}{\mathrm{D}t}\overrightarrow{k} \tag{2.2}$$

Desarrollando cada una de las componentes se tiene:

$$\begin{cases} \frac{\mathrm{D}u}{\mathrm{D}t}\overrightarrow{i} = \frac{\partial u}{\partial t}\overrightarrow{i} + u\frac{\partial u}{\partial x}\overrightarrow{i} + v\frac{\partial u}{\partial y}\overrightarrow{i} + w\frac{\partial u}{\partial z}\overrightarrow{i} \\ \frac{\mathrm{D}v}{\mathrm{D}t}\overrightarrow{j} = \frac{\partial v}{\partial t}\overrightarrow{j} + u\frac{\partial v}{\partial x}\overrightarrow{j} + v\frac{\partial v}{\partial y}\overrightarrow{j} + w\frac{\partial v}{\partial z}\overrightarrow{j} \\ \frac{\mathrm{D}w}{\mathrm{D}t}\overrightarrow{k} = \frac{\partial w}{\partial t}\overrightarrow{k} + u\frac{\partial w}{\partial x}\overrightarrow{k} + v\frac{\partial w}{\partial y}\overrightarrow{k} + w\frac{\partial w}{\partial z}\overrightarrow{k} \end{cases} \tag{2.3}$$

Agrupando términos se obtiene:

$$\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = \frac{\partial}{\partial t}\mathbf{u} + \mathbf{u}\left(u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} + w\frac{\partial}{\partial z}\right) = \partial_t\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} \tag{2.4}$$

□

2.1.2. Semidiscretización de la derivada material

Una forma semi-discreta de expresar lo mismo que la derivada material es el cambio que experimenta una partícula en un espacio $X(\Omega)$ en un incremento de tiempo Δt.

$$\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} \approx \frac{u^n(x) - u^{n-1}\circ X^{n-1}}{\Delta t} \tag{2.5}$$

El término $u^{n-1}\circ X^{n-1} = u(x^{old}, t-1)$ se refiere a la velocidad de la partícula cuando estaba situada en un punto del espacio y el tiempo inmediatamente anterior.

2.1.3. Hipótesis de continuidad

En la mayoría de las situaciones podemos considerar los fluidos como medios continuos. En este sentido, la hipótesis de continuidad se basa en el hecho de que las escalas espaciales y temporales de las estructuras moleculares que lo componen son de un orden extremadamente pequeño (Buchard, 2002).
Aceptando esta hipótesis, algunas de las propiedades más relevantes de un fluido real tienen que ser parametrizadas de tal modo que queden adecuadamente representadas en el modelo de fluido continuo. Las propiedades más relevantes

son la viscosidad y la densidad. En un fluido continuo, la viscosidad dinámica μ relaciona las fuerzas de fricción en el fluido, $\mathcal{F}_i$, con los gradientes de velocidad[1], v_i:

$$\mathcal{F}_i = \mu \partial_{jj} v_i \tag{2.6}$$

Las propiedades de la viscosidad pueden ser explicadas considerando un gas perfecto, en el que las moléculas se mueven libremente sin ninguna fuerza entre ellas, excepto en las colisiones. Cuando se define la media cuadrática (rms) de la velocidad de una única molécula con $\mathcal{U}$ y el tiempo medio entre colisiones con $\mathcal{T}$, entonces la viscosidad ν de un gas perfecto se puede aproximar como:

$$\nu = \frac{\mu}{\rho} \approx \mathcal{U}^2 \mathcal{T} = \mathcal{L}\mathcal{U} \tag{2.7}$$

donde $\mathcal{L}$ es la distancia a la superficie libre $\mathcal{L} = \mathcal{U}\mathcal{T}$, la densidad ρ y la viscosidad cinemática ν. La densidad ρ se define como la media cuadrática de la masa por unidad de volumen y depende en gran medida de la temperatura y la presión del fluido (Batchelor, 1967). No obstante, en líquidos como el agua, las moléculas se encuentran mucho más cercanas las unas a las otras de tal modo que las fuerzas de atracción y repulsión entre ellas no pueden ser despreciadas. Eso da lugar a que el concepto de viscosidad en los líquidos reales se complique mucho más y no sea posible explicarlo imaginando estas colisiones moleculares ideales.

2.1.4. Ecuación de estado

En los océanos la densidad aumenta con la profundidad y la presión debido a la disminución de la temperatura y al aumento de la salinidad. Esta capa en la que se produce una variación rápida en la densidad es conocida como picnoclina y está determinada experimentalmente. De manera general, tendremos siempre que el agua menos densa, más caliente y más ligera se encuentra en la superficie, mientras que el agua más fría, salada y densa se encontrará en capas inferiores. Esto da lugar a movimientos de masas de agua hundiéndose o ascendiendo en función de sus características físicas, dando lugar a las corrientes de convección. La densidad es directamente proporcional a la temperatura y a la salinidad, por eso no es de extrañar que en las capas de picnoclina también sucedan cambios bruscos en las otras dos características; dando lugar a la termoclina (capa en la que varía rápidamente la temperatura) y la haloclina (capa en la que varía rápidamente la salinidad).

[1]La velocidad del flujo se define como la velocidad media de las moléculas sobre un volumen lo suficientemente largo; al menos más largo que la distancia a la superficie libre. Para una definición estadística más exacta de la velocidad del flujo, ver Batchelor [1967].

Existen dos principales ecuaciones de estado. La más antigua se trata de la *Equation of State EOS-80 for Seawater*, (Millero and Poisson, 1981b) y más recientemente existe la *Thermodynamic Equation Of Seawater - 2010* (TEOS-10). Esta última es una formulación basada en en una función de Gibbs de la que pueden ser derivadas todas las propiedades termodinámicas del agua del mar (densidad, entalpía, entropía, velocidad del sonido, etc.) de manera termodinámicamente consistente. TEOS-10 fue adoptada en la 25º Comisión Oceanográfica Intergubernamental en junio de 2009 para reemplazar a la antigua EOS-80 como descripción oficial de las propiedades del agua de mar y hielo en las ciencias marinas.
La principal diferencia es el uso de la Salinidad Abosulta S_A (fracción de masa de sal en agua de mar) en vez de la Salinidad Práctica S_P (que esencialmente se trata de una medida de la conductividad del agua) para describir la concentración de sal. La salinidad de los océanos ahora tiene las unidades en el SI g/kg. El resto de propiedades quedan descritas en función de la salinidad absoluta.

La densidad del agua de mar se puede definir en función de la salinidad del agua (S) de la temperatura potencial (θ) y de la presión (p): $\rho = \rho(S, \theta, p)$. Esta relación se denomina ecuación de estado [3].
La temperatura potencial es la temperatura que tendría una masa de agua que, situada en una determinada profundidad, fuera adiabáticamente transportada por advección hasta la superficie. Para los órdenes de magnitud de las profundidades que manejamos en este estudio, podemos aproximar $\theta \approx T$, donde T es la temperatura in situ; y consideramos también que la influencia de la presión sobre la densidad es despreciable.

La ecuación de estado que hemos utilizado es una actualización de la propuesta por la UNESCO en 1982 que únicamente tiene en cuenta la salinidad y la temperatura y que tiene la siguiente forma:

$$\begin{gathered}\rho_0(T,S) = 999{,}842594 + 6{,}793952\cdot 10^{-2}T - 9{,}09529\cdot 10^{-3} \\ T^2 + 1{,}001685\cdot 10^{-4}T^3 - 1{,}120083\cdot 10^{-6}T^4 + 6{,}536332\cdot 10^{-9}T^5 \\ +(0{,}824493 - 4{,}0899\cdot 10^{-3}T + 7{,}6438\cdot 10^{-5}T^2 \\ -8{,}2467\cdot 10^{-7}T^3 + 5{,}3875\cdot 10^{-9}T^4)S \\ +(-5{,}72466\cdot 10^{-3} + 1{,}0277\cdot 10^{-4}T - 1{,}6546\cdot 10^{-6} \\ T^2)S^{1,5} + 4{,}8314\cdot 10^{-4}S^2 \ (\mathrm{kg/m^3})\end{gathered} \tag{2.8}$$

En Canarias la temperatura media del agua superficial es de 17^{o}C en invierno y de 25^{o}C en verano, mientras que a 100 metros de profundidad encontramos una temperatura media constante de 10^{o}C. En cuanto a la salinidad, Canarias presenta valores típicos de 36 g/l [4].

2.2. Leyes de conservación

Asumiendo la hipótesis de continuidad descrita, podemos describir el comportamiento de un fluido en términos de sus propiedades macroscópicas y sus derivadas espaciales y temporales; de modo que una partícula de fluido o punto en el fluido será considerado como el menor elemento posible de fluido cuyas propiedades macroscópicas no estén influenciadas por moléculas individuales [5].
Un modelo compuesto por ecuaciones que describen el movimiento de un fluido debe representar las expresiones matemáticas de las siguientes leyes de conservación de la física, independientemente del fluido que se considere [2]

- La masa de un fluido se conserva
- La variación del momento por unidad de tiempo experimentada por una partícula de fluido, es igual a la suma de las fuerzas que actúan sobre ella (segunda ley de Newton)
- La variación de la energía por unidad de tiempo experimentada por una partícula de fluido es igual a la suma del calor absorbido por la partícula mas el trabajo realizado sobre ella (primera ley de la termodinámica).

2.3. Ecuaciones de Navier-Stokes

Las ecuaciones del movimiento de un fluido newtoniano fueron derivadas en el sigo XIX por Navier (1822) y Stokes (1845). Consideremos un fluido newtoniano con densidad ρ y vector velocidad $\mathbf{u}$. La ecuación de continuidad de conservación de la masa para este fluido será:

$$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{2.9}$$

Considerando un flujo incompresible podemos escribir la ecuación anterior como:

$$\nabla \cdot \mathbf{u} = 0 \tag{2.10}$$

Dicho de otro modo, la divergencia de la velocidad es nula para fluidos incompresibles. Considerando los principios anteriores y simplificando obtenemos la ecuación de balance de los momentos:

$$\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = \mathbf{f} \tag{2.11}$$

donde p es la presión dividida por la densidad, $\nu = \mu/\rho$ es la viscosidad cinemática y μ es la viscosidad dinámica.
Dada la variedad de fenómenos que se ven reflejados en esta ecuación podemos referirnos a ellos por los nombres que le han sido asignados:

[2] Todas ellas son descritas con detalle en (Winter y González, 2014)

- El primer término se trata del término **temporal** y representa la variación con el tiempo en el interior del volumen de control.
- En segundo lugar se encuentra el término **convectivo**, y representa el transporte de la variable de un punto a otro del dominio por medio de la velocidad del flujo
- El tercer término es conocido como **difusivo** y se corresponde con algunos de los fenómenos de transporte que ocurren a nivel molecular: la ley de Fourier para la difusión del calor, la ley de Fick para la difusión de masa o la ley de Newton para la difusión de la cantidad de movimiento por efectos viscosos
- En el segundo miembro nos encontramos la **fuente**, que tiene en cuenta posibles fuentes o sumideros.

Con todo ello, las ecuaciones (2.10) y (2.11) forman las llamadas *ecuaciones de Navier-Stokes*[3].

Cabe destacar que la existencia de unicidad en las soluciones de estas ecuaciones en los casos tridimensionales es aún un problema abierto; y es conocido que la estructura de las soluciones es compleja especialmente cuando aparecen turbulencias debidas a las bajas viscosidades y a las inestabilidades del flujo.

Se pueden escribir las ecuaciones de Navier-Stokes en forma adimensional si introducimos el número de Reynolds, un valor adimensional utilizado para caracterizar la intensidad de la turbulencia, que se define como:

$$Re = \frac{U_0 L_0}{\nu}$$

donde U_0 y L_0 son, respectivamente, la velocidad y longitud características, y son valores que dependen del problema[4] que se tienden a simplificar y asumir que $U_0 L_0 = 1$. Valores pequeños del número de Reynolds dan lugar a un flujo laminar llamado *efecto Stokes*, mientras que grandes valores del número de Reynolds dan lugar a un flujo turbulento. Vemos por tanto la implicación directa de la velocidad y la viscosidad en el fenómeno de la turbulencia. Para Reynolds intermedios se dice que el flujo está en régimen de transición inestable. En flujos dominados por efectos viscosos (pequeños Reynolds), el término inicial $\mathbf{u}\cdot\nabla\mathbf{u}$ es mucho menor al término de viscosidad en la ecuación (2.11), por lo que las ecuaciones de Navier-Stokes podrían aproximarse a las ecuaciones de Stokes:

$$\partial_t \mathbf{u} - \nu\Delta\mathbf{u} + \nabla p = \mathbf{f},\ \nabla\cdot\mathbf{u} = 0$$

[3]Para un desarrollo más detallado ver Winter y González (2014) [5]

[4]En escalas oceánicas, U_0 es la media cuadrática de la velocidad de fluctuación de las turbulencias y L_0 es la escala de longitud de las turbulencias [6]

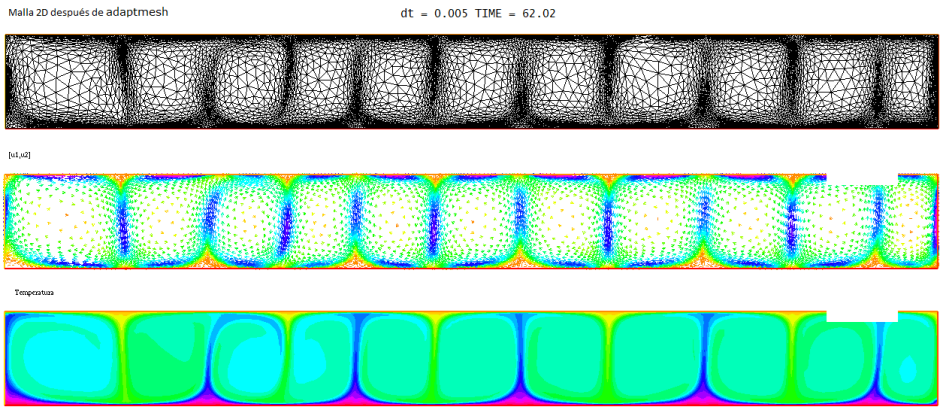

Figura 2.1: Simulación realizada por el Prof. Winter de un problema de Bènard, con la aproximación de Leray-Boussinesq. De arriba a abajo: malla, campo de velocidades, campo de temperaturas

En caso de que el flujo ha convergido a un estado estable, las ecuaciones de Stokes se pueden aproximar a las *ecuaciones estacionarias de Stokes*:

$$-\nu\Delta\mathbf{u} + \nabla p = \mathbf{f},\ \nabla\cdot\mathbf{u} = 0 \tag{2.12}$$

2.3.1. Aproximación de Boussinesq

La aproximación de Boussinesq se emplea a la hora de resolver problemas de flujos no-isotermos –por ejemplo problemas de convección natural– y su popularidad se debe a la reducción de coste computacional que conlleva. El caso test más reconocido para este fenómeno es el *problema de Bénard* [7], que consiste en el estudio de la convección de un fluido newtoniano incompresible situado en el interior de un recipiente calentado en su base, en un dominio bidimensional (ver figura 2.1)

Esta aproximación asume que las variaciones verticales de la densidad respecto al valor medio son pequeñas[5] [6] y la variación de densidad no tiene efectos sobre el fluido excepto en el término de la fuerza de flotación [8]. Típicamente, esta aproximación se usa para modelar líquidos con diferentes temperaturas, ventilación natural en edificios, flujos de aire causados por fuego en recintos confinados o la dispersión de gases densos en entornos industriales.

A partir de un fluido de tipo Boussinesq –con viscosidad $\nu = \mu/\rho$ constante y conductividad térmica κ consideramos el equilibrio de momentos de las ecuaciones de Navier-Stokes (2.11) en la tercera componente, sin simplificar la

[5]En términos de la aproximación de Boussinesq, la ecuación de conservación de la masa se simplifica a la condición de incompresibilidad (2.10)

densidad, donde actúa el campo de gravedad

$$\rho(\partial_t u_3 + \mathbf{u}\cdot\nabla u_3) - \mu\Delta u_3 + \nabla p = -\rho g \tag{2.13}$$

y la ecuación de la conducción térmica

$$\rho C_p\left(\frac{\partial T}{\partial t} + u_j\frac{\partial T}{\partial x_j}\right) = Q + \kappa\frac{\partial^2 T}{\partial x_j^2} \tag{2.14}$$

donde C_p es el calor específico a presión constante, T la temperatura (K), Q es una fuente de calor y κ la conductividad térmica.

Siendo $p = -\rho_0 g x_3$ la presión hidrostática a una profundidad $z = x_3, z > 0$, entonces

$$\frac{\partial p}{\partial x_3} = -\rho_0 g$$

y sustituyendo en (2.13) tenemos:

$$\rho(\partial_t u_3 + \mathbf{u}\cdot\nabla u_3) - \mu\Delta u_3 = \rho_0 g - \rho g = (\rho_0 - \rho)g \tag{2.15}$$

Añadiendo el coeficiente de expansión térmica β

$$\beta = \frac{-}{\rho}\frac{\partial\rho}{\partial T} \approx \frac{-1}{\rho}\left(\frac{\rho-\rho_0}{T-T_0}\right) \tag{2.16}$$

por lo que $\rho\beta(T-T_0) = \rho_0 - \rho$, donde T_0 y ρ_0 son una temperatura y densidad de referencia, respecto a la que se cuantifican los cambios. Sustituyendo también en (2.13) queda finalmente la expresión en función también de la variación de temperatura y el coeficiente de expansión térmica, es decir, que el fluido adquiere un carácter de flotabilidad debido a las variaciones de temperatura y densidad.

$$\begin{aligned}\rho(\partial_t u_3 + \mathbf{u}\cdot\nabla u_3) - \mu\Delta u_3 &= \rho\beta(T-T_0)\\ (\partial_t u_3 + \mathbf{u}\cdot\nabla u_3) - \nu\Delta u_3 &= \beta(T-T_0)\end{aligned} \tag{2.17}$$

Para el cálculo de temperatura simplemente resolvemos la ecuación (2.14):

$$\partial_t T + \mathbf{u}\cdot\nabla T - \frac{\kappa}{\rho C_p}(\Delta T) = 0 \tag{2.18}$$

2.4. Resolución numérica de las ecuaciones de Navier-Stokes

Consideremos las ecuaciones de Navier-Stokes evolutivas (dependientes del tiempo) con la aproximación pseudo-compresible, o penalización de Taylor, que afecta a la ecuación (2.10) en la que se introduce el término $\varepsilon \to 0$:

$$\begin{aligned}\partial_t\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} - \nu\Delta\mathbf{u} + \nabla p &= \mathbf{f}\\ \nabla\cdot\mathbf{u} - \varepsilon p &= 0\end{aligned} \tag{2.19}$$

Este sistema de ecuaciones se convierte en no lineal debido al término convectivo $\mathbf{u}\cdot\nabla\mathbf{u}$, causa de la dificultad matemática del análisis diferencial de los flujos. Una alternativa para aproximar esta ecuación es utilizando el método de características para el término convectivo y el método estándar de los elementos finitos para el resto de los términos.
Los términos $\partial_t\mathbf{u}+\mathbf{u}\cdot\nabla\mathbf{u}$ equivalen a la derivada material como se demostró en el apartado 2.1.1 de esta memoria.

2.4.1. Implementación vectorial

Al referirnos a un fluido nos interesa no simplificar la densidad en la ecuación del balance de los momentos (2.11), teniendo entonces

$$\partial_t(\rho\mathbf{u})+\nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u})-\nabla\cdot(\mu\nabla\mathbf{u})+\nabla p=\rho\mathbf{f}$$

donde $\otimes$ denota el producto tensorial de los vectores espaciales. Considerando ρ como una constante

$$\rho\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t}-\mu\Delta\mathbf{u}+\nabla p=\mathbf{f} \tag{2.20}$$

En la primera ecuación se simplifica ρ, se integran los términos en un dominio $\Omega\in\mathbb{R}^3$ y multiplicando cada uno de ellos por una función test suave $\mathbf{v}(v_1,v_2,v_3)$ que se anula en los contornos en los que se impone condición Dirichlet; se tiene, para $i=1,2,3$

$$\int_\Omega\frac{\mathrm{D}u_i}{\mathrm{D}t}v_i\,\mathrm{d}\Omega+\nu\int_\Omega\nabla u_i\nabla v_i\,\mathrm{d}\Omega+\int_\Omega\frac{1}{\rho}\frac{\partial p}{\partial x_i}v_i\,\mathrm{d}\Omega=\int_\Omega f_iv_i\,\mathrm{d}\Omega \tag{2.21}$$

En esta ecuación podemos sustituir la integral de presión por suma de integrales:

$$\int_\Omega\frac{1}{\rho}\frac{\partial p}{\partial x_i}v_i\,\mathrm{d}\Omega=-\int_\Omega\frac{\partial v_i}{\partial x_i}\frac{p}{\rho}\,\mathrm{d}\Omega+\int_{\partial\Omega}\frac{p}{\rho}v_i\,\mathrm{d}S \tag{2.22}$$

Si conocemos e imponemos la solución en los contornos, la condición Dirichlet anula las funciones test v_i en los contornos $\partial\Omega$, anulando por tanto las integrales de superficie. Análogamente con el término de la viscosidad y discretizando la derivada material según el apartado 2.1.2, la ecuación (2.21) queda:

$$\int_\Omega\frac{u^n-u^{n-1}}{\Delta t}v_i\,\mathrm{d}\Omega+\nu\int_\Omega\nabla u_i\nabla v_i\,\mathrm{d}\Omega-\int_\Omega\hat{p}\frac{\partial v_i}{\partial x_i}\,\mathrm{d}\Omega=\int_\Omega f_iv_i\,\mathrm{d}\Omega \tag{2.23}$$

donde $\hat{p}=p/\rho$.
En la segunda ecuación introducimos el término de penalización como se hizo en (2.19), pudiendo aproximar que $\nabla\cdot\mathbf{u}=\varepsilon p$. Se multiplica por una función test suave $q\in\mathrm{L}^2$ y se integra en el mismo dominio Ω. Tenemos:

$$-\int_\Omega\nabla\mathbf{u}\,q\,\mathrm{d}\Omega-\varepsilon\int_\Omega\hat{p}q\,\mathrm{d}\Omega=0 \tag{2.24}$$

Con todo lo anterior, desacoplando las ecuaciones en sus componentes vectoriales obtenemos:

$$\begin{aligned}&\int_\Omega \frac{1}{\Delta t}[(u_1v_1+u_2v_2+u_3v_3)-(u_1^{n-1}v_1+u_2^{n-1}v_2+u_3^{n-1}v_3)]\,\mathrm{d}\Omega\\&+\int_\Omega \nu(\nabla u_1^T\nabla v_1+\nabla u_2^T\nabla v_2+\nabla u_3^T\nabla v_3)\,\mathrm{d}\Omega-\varepsilon\int_\Omega \hat{p}\,q\,\mathrm{d}\Omega\\&-\int_\Omega \hat{p}\,\mathrm{div}(v_1,v_2,v_3)+q\,\mathrm{div}(u_1,u_2,u_3)\,\mathrm{d}\Omega=0\end{aligned}\tag{2.25}$$

Cabe destacar, llegados a este punto, que para la implementación de esta ecuación vectorial en Freefem++, el término u^{n-1} es calculado por Freefem++ con el operador `convect = ([u1,u2,u3],-dt,uXold)` donde `uXold` es el valor de la componente X(1,2,3) del vector **u** en la iteración anterior [9].

Como se comentaba al principio de este apartado, el término convectivo hace que esta ecuación sea no lineal, siendo necesario para su resolución por el método de los elementos finitos utilizar polinomios de grado dos en los espacios funcionales, cuyo coste computacional se traduce en mayor tiempo de cálculo.

Si queremos tener en cuenta un fluido de Boussinesq, debemos añadir a (2.25) la integral correspondiente, extraída de la formulación variacional de (2.17):

$$-\int_\Omega \beta g(T+T0)w\,\mathrm{d}\Omega\tag{2.26}$$

Para el cálculo de temperatura se procede análogamente, obteniendo la formulación variacional de (2.18) que queda

$$\int_\Omega \frac{T^n-T^{n-1}}{\mathrm{d}t}w\,\mathrm{d}\Omega+\frac{\kappa}{\rho C_p}\int_\Omega \nabla T\nabla w\,\mathrm{d}\Omega+\int_{\partial\Omega}\frac{\partial T}{\partial \mathbf{n}}w\,\mathrm{d}\Gamma\tag{2.27}$$

2.4.2. Algoritmo de Chorin-Rannacher

El método de proyección propuesto por Chorin [10] y posteriormente refinado y demostrado por Rannacher [11] busca linearizar las ecuaciones de Navier-Stokes, pudiendo resolverse utilizando espacios funcionales con polinomios de grado uno, disminuyendo notablemente el coste computacional del algoritmo.

En métodos de resolución estándar se calculan la velocidad **u** y la presión p simultáneamente, hallando una solución en la que ambas variables están unidas. Desde un punto de vista numérico, esto lleva a un enorme sistema, como el obtenido en el apartado anterior, (2.25), que resolver en cada paso de tiempo. Dado el caso, en tres dimensiones, considerando polinomios P1 y N grados de

libertad, el tamaño del sistema es $4N$.

Como la resolución de cuatro sistemas lineales de tamaño N es más sencillo que la resolución de un problema lineal de tamaño $4N$, se han desarrollado técnicas para resolver estos sistemas con las variables por separado. De este modo, el algoritmo de Chorin original divide en dos partes las ecuaciones de Navier-Stokes, resolviendo por un lado la ecuación de balance de los momentos olvidando la presión, para luego resolver únicamente este término.

Por su parte, Rannacher introduce de nuevo la presión en la primera parte de la resolución y añade un término q auxiliar para la presión en la segunda parte.

El algoritmo de Chorin parte de la ecuación semidiscretizada:

$$\frac{1}{dt}[u^n - u^{n-1}] - \nu\Delta\mathbf{u} + \nabla p = 0 \tag{2.28}$$

Multiplicando por una función test suave w e integrando a lo largo del dominio $\Omega \in \mathbb{R}^3$ tenemos:

$$\int_\Omega \left(\frac{u^n}{dt} - \frac{u^{n-1}}{dt}\right) w \, \mathrm{d}\Omega + \nu \int_\Omega \nabla\mathbf{u}\nabla w \, \mathrm{d}\Omega + \int_\Omega \nabla p \, w \, \mathrm{d}\Omega \tag{2.29}$$

cuya implementación en Freefem++ es inmediata, recordando el operador `convect`.
Para el cálculo de las presiones, Rannacher introduce las siguientes identidades:

$$-\Delta q = \nabla u - \overline{\nabla u} \tag{2.30}$$

$$p^n = p^{n-1} - q - \overline{p^{n-1} - q} \tag{2.31}$$

donde $\overline{\nabla u}$ es la divergencia del campo de velocidades promediado a lo largo de todo Ω, al igual que $\overline{p^{n-1} - q}$ es el promedio de la diferencia a lo largo del dominio. Anulando el término de la viscosidad tenemos una expresión en la que se relacionan las variables en cada punto con el valor promedio a lo largo de Ω.

$$\frac{1}{dt}[\overline{u^*} - u^*] + \nabla p = 0 \tag{2.32}$$

donde u^* representa la solución de $\mathbf{u}$ obtenida previamente en la ecuación (2.29). Obteniendo la divergencia de la ecuación (2.32) e igualando $p = -q$ tenemos

$$\frac{\overline{\nabla u^*} - \nabla u^*}{dt} - \Delta q = 0 \tag{2.33}$$

Multiplicando por una función test suave w e integrando por partes a lo largo de Ω se obtiene la formulación variacional del problema:

$$\int_\Omega \nabla q \nabla w \, \mathrm{d}\Omega - \int_\Omega \frac{\overline{\nabla u^*} - \nabla u^*}{dt} w \, \mathrm{d}\Omega \tag{2.34}$$

De la expresión (2.32) se despeja que

$$u^n = u^* + dt \nabla q \tag{2.35}$$

Las igualdades (2.31) y (2.35) son las que unifican los resultados del cálculo de velocidad y presión por separado y proporcionan el resultado final en cada iteración.

Por tanto, a modo de resumen concluimos que debemos calcular en primer lugar la solución a las ecuaciones (2.29) y (2.34) en un sistema formado por cuatro ecuaciones diferentes, separando las variables (u_1, u_2, u_3, p). Por último unificamos los resultados obtenidos mediante las igualdades (2.31) y (2.35). La implementación en Freefem++ de este algoritmo se encuentra en el Anexo B de esta memoria.

$$\int_\Omega \left(\frac{u^n}{dt} - \frac{u^{n-1}}{dt} \right) w \, \mathrm{d}\Omega + \nu \int_\Omega \nabla \mathbf{u} \nabla w \, \mathrm{d}\Omega + \int_\Omega \nabla p \, w \, \mathrm{d}\Omega$$

$$\int_\Omega \nabla q \nabla w \, \mathrm{d}\Omega - \int_\Omega \frac{\overline{\nabla u^*} - \nabla u^*}{dt} w \, \mathrm{d}\Omega$$

$$p^n = p^{n-1} - q - \overline{p^{n-1} - q}$$

$$u^n = u^* + dt \nabla q$$

En Freefem++ se implementan estas ecuaciones, respectivamente:

```
// -- Calculo por separado de las componentes de la velocidad --
solve pb4u(u1,w1,init=i,solver=UMFPACK)
=int3d(Th)(u1*w1/dt +nu*(dx(u1)*dx(w1)
+dy(u1)*dy(w1)+dz(u1)*dz(w1)))
-int3d(Th)((f/dt-dx(p))*w1)
// + condiciones de contorno;

solve pb4v(u2,w2,init=i,solver=UMFPACK)
= int3d(Th)(u2*w2/dt +nu*(dx(u2)*dx(w2)
+dy(u2)*dy(w2)+dz(u2)*dz(w2)))
-int3d(Th)((g1/dt-dy(p))*w2)
// + condiciones de contorno;

solve pb4w(u3,w3,init=i,solver=UMFPACK)
= int3d(Th)(u3*w3/dt +nu*(dx(u3)*dx(w3)
+dy(u3)*dy(w3)+dz(u3)*dz(w3)))
-int3d(Th)((g2/dt-dz(p))*w3)
// + condiciones de contorno;

//  -- Termino de presion --

real meandiv = int3d(Th)(dx(u1)+dy(u2)+dz(u3))/vol;

solve pb4p(q, w4, init = i, solver = UMFPACK)
= int3d(Th)(dx(q)*dx(w4) + dy(q)*dy(w4) + dz(q)*dz(w4))
- int3d(Th)((dx(u1) + dy(u2) + dz(u3) - meandiv) * w4/dt)
// + condiciones de contorno;

// -- Actualizacion de los valores --
real meanpq = int3d(Th)(pold - q)/vol;

p = pold - q - meanpq;
u1 = u1 + dx(q) * dt;
u2 = u2 + dy(q) * dt;
u3 = u3 + dz(q) * dt;
```

2.5. Condiciones de contorno

Existen múltiples posibilidades en cuanto a la elección de las condiciones de contorno. Sin embargo, lo más habitual es utilizar condiciones de contorno de Dirichlet o de Von Neumann. En las primeras, se fija el valor de la función en el contorno conocido (2.36). En las segundas se fija el valor de la derivada de la función o solución con respecto al vector normal n en cada punto de la frontera $\Gamma = \partial\Omega$ (2.37). El caso de contar con condiciones de ambos tipos por separado se conoce como condiciones de Cauchy. La condición de contorno de Robin es otro tipo de condición de frontera híbrida; siendo una combinación

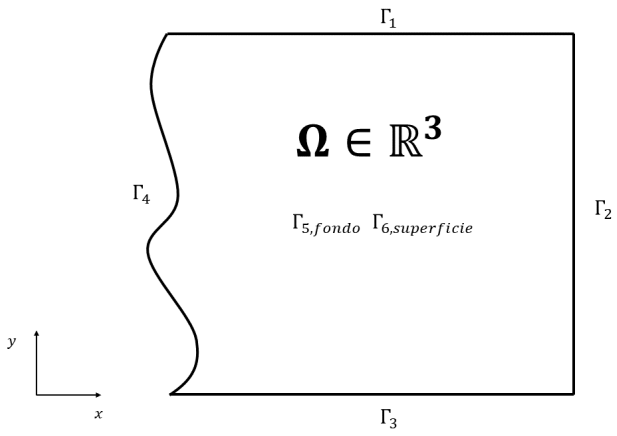

Figura 2.2: Esquema de dominio computacional con volumen Ω y contornos $\partial\Omega = \Gamma$

lineal de las dos primeras condiciones (2.38).

$$U\Big|_{\Gamma} = (u_x, u_y, u_z) \tag{2.36}$$

$$\frac{\partial U}{\partial n}\Big|_{\Gamma} = (u_x, u_y, u_z) \tag{2.37}$$

$$\left[aU + b\frac{\partial U}{\partial n}\right]_{\Gamma} = g \tag{2.38}$$

Las condiciones Dirichlet se utilizan por lo general para imponer valores conocidos en algún punto del dominio, por ejemplo, si conocemos una temperatura o velocidad. En las simulaciones de dinámica de fluidos es habitual considerar que, debido a la viscosidad del fluido, la velocidad del flujo es cero cuando está en contacto con las paredes; esta condición es la llamada *condición de no deslizamiento* (no flip condition). Las condiciones Neumann por su lado se suelen emplear para imponer o describir el comportamiento en el contorno. La derivada parcial de una función respecto a la normal en ese contorno igualada a cero quiere decir que la función no varía entre el último punto del contorno y el siguiente, es decir, es como si ese contorno no existiera, o fuera un contorno libre.

Cuando se trata del mar, en la superficie Γ_6 (ver figura 2.2), la velocidad de deriva debida al viento, U_d, es por lo general del 3 % al 4 % del vector velocidad del viento a 10 m por encima del nivel del mar: $U_d = F_d U_w$, con $F_d = 0{,}03$. En la literatura (Garrat, 1977) F_d varía entre el 1 % y el 4.5 %, pero valores del 3 - 3.5 % son más a menudo usados para vientos moderados en áreas de aguas abiertas (Henry and Heaps, 1976). Valores menores son a menudo usados

en embahiamientos cerrados o semi-encerrados. Por tanto, con una velocidad de viento de dirección y módulo constante, podemos establecer la condición de contorno $u|_{\Gamma_6} = u_c + 0{,}03U_w$ donde u_c es la velocidad de corriente en la superficie.

Condiciones de contorno de un modelo simple de corriente uniforme

Las condiciones de contorno impuestas en las simulaciones pueden variar su complejidad en función del nivel de simplificación que se desee aceptar. En este apartado se relacionan con las condiciones más sencillas que pueden ocurrir, que consiste en una entrada de flujo constante por el norte con velocidad v_c y el mismo flujo que sale por el sur. En el resto de contornos, en especial la costa, se imponen condiciones de no deslizamiento. En general (ver figura 2.2):

- Entrada: se impone una condición Dirichlet en la superficie, que representa un flujo de entrada constante a lo largo de todo el plano. Por ejemplo, si queremos una corriente de componente norte, la cara de entrada será Γ_1:

$$U\Big|_{\Gamma_1} = (u_x, -v_c, u_z) \tag{2.39}$$

  En Freefem++:

```
// Entrada, en eq de primera componente:
+ on(1, u1 = 0)
// Entrada, en eq de segunda componente:
+ on(1, u2 = -0.6)
// Entrada, en eq de tercera componente:
+ on(1, u3 = 0)
```

- Salida: las caras de salida son aquellas en las que se modela un contorno libre, porque el flujo puede salir del dominio computacional sin ningún tipo de restricción. Para ello se impone una condición Neumann en la cara sur, esto es, la ecuación (2.37) evaluada sobre el contorno Γ_3.
  Por su parte, en el algoritmo de Chorin-Rannacher este tipo de contornos debe implementarse como una condición Dirichlet en el que la velocidad se iguala al término convectivo; y del mismo modo, el término q relacionado con la presión debe igualarse a cero en estos contornos.

```
// Salida:
+ on(3, u1 = f)
// Salida:
+ on(3, u2 = g1)
```

```
// Salida:
+ on(3, u3 = g2)
```

- No deslizamiento: se impondrá, a través de una condición Dirichlet, una velocidad nula en el contorno de la costa, Γ_4, y el fondo, Γ_5, al considerarse que la velocidad de la corriente se reduce notablemente por efecto viscoso. Además, a efectos de simplificar los cálculos, también se impondrá esta condición en el resto de contornos (lateral y superficie, Γ_2 y Γ_6 respectivamente).

2.6. Matriz de Masa Consistente

El modelo de matriz masa consistente[6] (MMC) determina un campo de velocidades tridimensional, en un fluido incompresible, que se ajuste a un campo de velocidades generado a partir de datos medidos in situ (por ejemplo, mediante correntímetros) o, alternativamente, generado por expresiones experimentales de las que se disponga, o bien, a partir de información generada por otros modelos. La importancia de uso del modelo MMC obedece a que elimina la dificultad de definir de manera exacta en una aplicación numérica concreta, las condiciones de contorno en la resolución numérica del problema de contorno formulado mediante las ecuaciones de Navier- Stokes.

El modelo de Matriz de Masa Consistente supone una ventaja en términos de coste computacional frente a los algoritmos evolutivos que requieren un tiempo de cómputo significativamente mayor; además de adaptar el campo del dominio a unos datos empíricos, ayudando a conseguir una simulación más ajustada a la realidad. Este modelo fue adaptado al medio marino y al espacio tridimensional con el método de los volúmenes finitos por González (2001).
Como novedad en nuestro trabajo, la hemos adaptado al esquema de los elementos finitos, siendo posible utilizar la solución estacionaria de la MMC como condición inicial para la resolución del problema evolutivo, favoreciendo una convergencia más rápida y realista.

Para un dominio $\Omega \subset \mathbb{R}^3$ con frontera $\Gamma = \Gamma_1 \cup \Gamma_2$, representando Γ_1 el fondo marino, la costa y la superficie del mar, el modelo está basado en la ecuación de continuidad para un fluido incomprensible en Ω y condiciones de

[6] El modelo inicialmente ideado para el ajuste de campos de viento sobre terrenos con orografía compleja. Sasaki (1958, 1970), Sherman (1978), Dickerson (1978), Lalas (1985), Winter et al (1995), Ratto (1996). En González y Winter (1999, 2000), González (2001) se extiende su uso al ajuste de corrientes marinas y, en general, a campos de velocidades en fluidos incompresibles.

impermeabilidad en Γ_1:

$$\begin{cases} \nabla \mathbf{u} = 0 & \text{en} \quad \Omega \\ \mathbf{u} \cdot \mathbf{n} = 0 & \text{en} \quad \Gamma_1 \end{cases} \tag{2.40}$$

Se formula un problema de mínimos cuadrados en Ω con el objeto de ajustar $\mathbf{u}(u_x, u_y, u_z)$ a un campo inicial de velocidades $\mathbf{u}_0(u_{0x}, u_{0y}, u_{0z})$ obtenido de la interpolación de medidas experimentales o empíricas en el dominio de estudio.

$$J : K \subseteq [H^1(\Omega)]^3 \to \mathbb{R}$$
$$\mathbf{u} \to J(\mathbf{u})$$

$$J(\mathbf{u}) = \int_\Omega \left[\alpha_1^2 \left((u_x - u_{0x})^2 + (u_y - u_{0y})^2\right) + \alpha_2^2 (u_z - u_{0z})^2\right] \, \mathrm{d}\Omega$$

$$K = \{\mathbf{u} \in [H_1(\Omega)]^3 : \nabla \mathbf{u} = 0, \ \mathbf{u} \cdot \mathbf{n}|_{\Gamma_1} = 0\}$$

donde α_1 y α_2 son los módulos de precisión de Gauss. Minimizando el funcional o integral y operando (González, 2001), se obtienen finalmente las ecuaciones de Euler-Lagrange:

$$\mathbf{u} = \mathbf{u}_0 + \mathbf{P^{-1}grad}(\lambda) \text{ en } \Omega \tag{2.41a}$$
$$\lambda = 0 \text{ en } \Gamma_2 \tag{2.41b}$$

Donde P es la matriz

$$P = \begin{bmatrix} \alpha_1^2 & 0 & 0 \\ 0 & \alpha_1^2 & 0 \\ 0 & 0 & \alpha_2^2 \end{bmatrix}$$

Sustituyendo (2.41a) en (2.42) obtenemos:

$$\begin{cases} -\nabla[\mathbf{P^{-1}grad}(\lambda)] = \nabla \mathbf{u}_0 & \text{en} \quad \Omega \\ -\mathbf{P^{-1}grad}(\lambda) \cdot \mathbf{n} = \mathbf{u}_0 \cdot \mathbf{n} & \text{en} \quad \Gamma_1 \\ \lambda = 0 & \text{en} \quad \Gamma_2 \end{cases} \tag{2.42}$$

2.6.1. Campo inicial aproximado de velocidades

Si consideramos que m es el número de medidas observadas o datos disponibles sobre el dominio, el campo u_0, campo inicial aproximado de velocidades para iniciar el proceso de cálculo, se puede construir, por ejemplo, mediante la siguiente interpolación en los nodos (Ratto, 1996):

$$u_{0i} = \frac{\sum_{j=1}^{m} u_{ij} \frac{1}{d_j^2}}{\sum_{j=1}^{m} \frac{1}{d_j^2}} \tag{2.43}$$

siendo $1 \leqslant i \leqslant 3$ y d_j la distancia de cada nodo a la j-ésima estación de medida, u_{ij} es la i-ésima componente de la velocidad medida en la j-ésima estación de medida y u_{0i} es la i-ésima componente de la velocidad interpolada en un nodo.

Para el cálculo del perfil vertical de corrientes debemos tener en cuenta los efectos de la espiral de Ekman y de las aceleraciones de Coriolis en el mar, a lo largo de las capas que hay entre la superficie y el fondo. La expresión que se utiliza para la disminución del módulo con la profundidad es:

$$u_z = u_0 \mathrm{e}^{-az} \tag{2.44}$$

donde u_z es el módulo de la corriente a la profundidad z ($z \geq 0$), u_0 es el módulo de la corriente en superficie y a es el coeficiente del perfil de corrientes que se debe estimar de forma experimental. En el entorno de Gran Canaria se considera $a = 0{,}038$ [12].

Además, para calcular las componentes, si α es el ángulo que forma el vector de corriente en la superficie con la dirección Norte y en sentido antihorario, entonces las componentes del vector de corriente a una profundidad z ($z \geq 0$), u_0 serán:

$$\begin{aligned} (u_x)_z &= u_z \cos[(\pi/2) - \alpha - az] \\ (u_y)_z &= u_z \sin[(\pi/2) - \alpha - az] \end{aligned} \tag{2.45}$$

Este modelo lo emplearemos para calcular el campo inicial de velocidades (ver apartado 4.2).

2.7. Consideraciones sobre la discretización

2.7.1. La malla: discretizar el espacio

En general, la estrategia utilizada en los problemas de la dinámica de fluidos (CFD por sus siglas en inglés, *computerized fluid dynamics*) es la de remplazar un problema definido sobre un dominio continuo (hipótesis del continuo en Mecánica de Fluidos clásica) por un dominio discreto definido a partir de una malla.
En el continuo cada variable del flujo (presión, velocidad y temperatura) está definida en todos los puntos del espacio. Sin embargo, en el dominio discreto, cada variable del flujo está definida únicamente en los puntos (nodos) que configuran la malla. A este proceso se le denomina discretización espacial, porque el espacio se "discretiza" en un número finito de puntos.

Así, en una simulación CFD solamente se resuelven las variables de interés en los puntos que definen la malla. Los valores en otras posiciones se pueden determinar interpolando entre los valores resueltos en los nodos. Las ecuaciones de gobierno en derivadas parciales, así como las condiciones de contorno, están definidas matemáticamente mediante variables continuas –por ejemplo, velocidad o temperatura–. Es posible aproximar estas complejas ecuaciones

no lineales por una serie de ecuaciones algebraicas que relacionan las variables discretizadas. El sistema de ecuaciones resultante es un gran conjunto de ecuaciones algebraicas acopladas en variables discretas. Manipular este sistema y resolverlo (lo cual implica un problema de inversión matricial) requiere un número muy grande de cálculos respectivos, que deberán ser realizados por un ordenador.

El método de los elementos finitos (*MEF* o *FEM* por sus siglas en inglés) resuelve las ecuaciones en puntos representativos del espacio. El conjunto de estos puntos, por tanto, van a suponer una discretización del espacio; y su distribución a lo largo del mismo, o la forma de discretización, es de suma importancia.

Una malla de elementos finitos, por tanto, se define como una teselación regular de un subespacio dentro del espacio tridimensional, formada por elementos geométricos, usualmente cuadrados o triángulos (2D) y cubos o tetraedros (3D), distribuidos de tal manera que estos elementos solo se puedan intersectar a lo largo de una de sus caras, aristas o vértices, y nunca de otra manera. Los vértices, puntos donde se calculan las soluciones numéricas del problema que se resuelve, son llamados *nodos*, y deben estar distribuidos de la manera más uniformemente posible.
La generación de la malla es la parte más importante en la preparación de un modelo para la simulación por CFD. Ninguna simulación puede realizarse sin haber previamente definido una malla con una distribución de puntos apropiada.

Otro factor clave que define el tamaño de los elementos es el error cometido. De acuerdo con el lema de Ceá [13] , el error cometido en la aproximación de una solución exacta mediante elementos finitos viene acotado por el error de aproximación. Esto quiere decir que la solución obtenida mediante el MEF es tanto más buena cuanto mejor sea la aproximación del espacio discretizado al espacio real. Por tanto, cuanto menor sea el tamaño de los elementos, o mayor número de estos haya, tanto menor será el error de aproximación.

El tipo de conectividad que existe entre los diferentes nodos o celdas de la malla permite clasificar los mallados en dos grandes categorías básicas:

- Mallas estructuradas: en ellas, la retícula de las celdas se construye a partir de una red de familias de lineas ordenadas (figura 2.3-a)
- Mallas no estructuradas: en esta red no se sigue ningún tipo de dirección permanente ni predominante (figura 2.3-b). El desarrollo de este tipo de mallas ha sido consecuencia de la necesidad de desarrollar geometrías cada vez más complejas en las que no es fácil poder adecuar bloques paralelepipédicos con mallas ortogonales. El empleo de este tipo de mallas

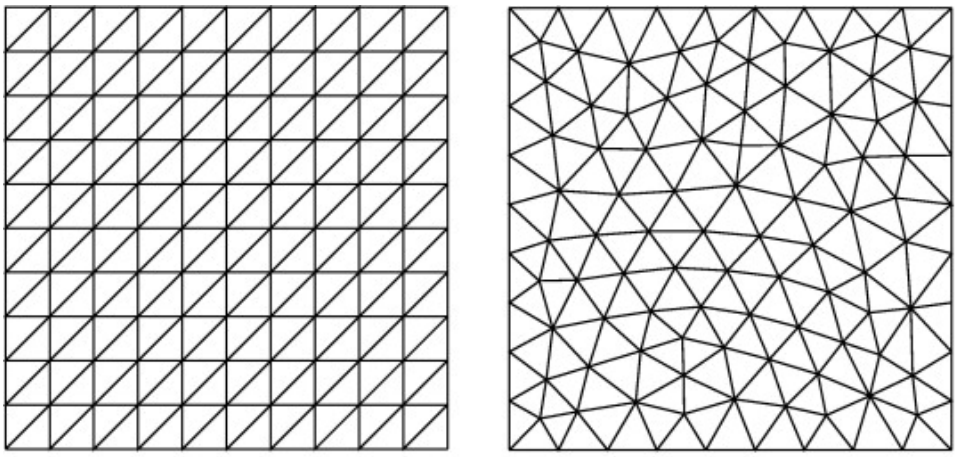

Figura 2.3: (a) malla estructurada, (b) malla no estructurada

consigue reducir significativamente los tiempos de construcción de los modelos, pero lógicamente, existe una penalización, tanto en términos de precisión como en coste computacional, en comparación con mallas estructuradas.

Independientemente del tipo de malla empleado, es esencial satisfacer una serie de requisitos básicos para conseguir una malla apta para el método de elementos finitos. Se destacan los siguientes requisitos [14]

- La malla debe ser generada con cierta previsión en función del tipo de flujo que se espera resolver
- Es necesario una mayor resolución en zonas donde el flujo presente importantes gradientes
- El mallado se debe distribuir por todo el dominio de la forma más regular posible, de modo que no haya variaciones importantes en la malla
- La resolución en las zonas donde se establezca una capa límite debe estar en consonancia con el modelo de turbulencia y de pared que se vaya a utilizar
- Se evitarán elementos singulares o deformados, como celdas muy angulosas
- Es interesante que el mallado sea capaz de adaptarse de forma dinámica a las variaciones de las variables en la solución del flujo
- El tamaño global de la malla debe ajustarse a las posibilidades y potencia de cálculo de los equipos en los que se vaya a resolver el problema

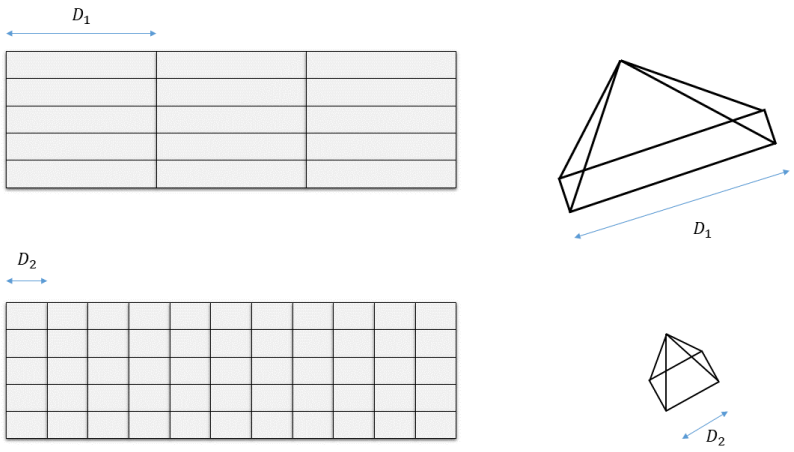

Figura 2.4: Formas que adquieren los tetraedros cuando se aplica un mallado más fino o más grosero, manteniendo constante el número de capas de nodos entre la superficie y el fondo

Otras consideraciones del mallado de entornos costeros

En una aplicación como la que ocupa este trabajo, la uniformidad del mallado obedece además a otros criterios: el mallado debe ser más fino en los lugares en los que se prevee una mayor velocidad del flujo, o bien la aparición de turbulencias; de lo contrario aparecerán errores numéricos en la resolución por elemenos finitos impidiendo la convergencia. Para el modelado de turbulencia, la experiencia en el trabajo en este área ha demostrado que también un menor tamaño de los elemenos captura mejor este tipo de fenómenos.
A efectos prácticos, en el mallado de entornos costeros, el tamaño de los elementos también está sujeto al correcto seguimiento de la geometría que se desea, es decir, que en zonas costeras debe ser lo suficientemente fino como para capturar los detalles que se deseen. En caso de estar compuesto por elementos groseros se obviarán entornos que pueden superar los 25 metros en la realidad, por lo que conviene, según el caso de estudio, que el tamaño de los elementos, en escala real, no supere los 5 metros.

Por otro lado, el factor que limita el tamaño máximo de los elementos en nuestra malla es la profundidad y cuántas capas de nodos son deseables. En cuanto a esto último, se ha considerado que el mínimo deseable es de cinco capas; de manera que las de superficie y fondo están impuestas por las condiciones de contorno, y queden tres nodos intermedios que puedan interactuar con los contornos y obtener sus propias soluciones. De tal suerte que, para que los elementos dispongan de una geometría regular, el tamaño máximo de los

elementos no debe superar demasiado una quinta parte de la máxima profundidad considerada en el dominio.

La importancia de una forma regular de los tetraedros radica en el valor de la matriz jacobiana que aparece en la transformación al espacio de referencia del MEF. Las coordenadas de un tetraedro deformado o irregular darán lugar a matrices jacobianas con valores cercanos a cero que distorsionarán el resultado; e incluso jacobianos negativos resultan de elementos con nodos enredados.

2.7.2. Discretización del tiempo

Como hemos visto, la idea es realizar una discretización espacial del dominio físico. Si las ecuaciones de gobierno de nuestro sistema presentan un comportamiento dependiente del tiempo, deberemos entonces discretizarlo también, de manera que resolveremos iterativamente nuestras ecuaciones en intervalos temporales. Una adecuada elección de este intervalo –también llamado incremento de tiempo, Δt o simplemente diferencial de tiempo dt– no es una cuestión baladí ni sencilla. Un elevado valor de dt hará que las soluciones obtenidas sean poco precisas o incluso sin sentido, ya que por el carácter retroalimentativo de los procesos iterativos, el comportamiento de una variable continua experimenta grandes cambios en poco tiempo, que no estaremos captando, llegando a enfrentarnos rápidamente a problemas de convergencia en la solución. Esta situación se ve acusada en problemas con flujos turbulentos. Por otro lado, un diferencial de tiempo demasiado pequeño elevará el coste computacional –en términos de tiempo de cómputo y memoria– pudiendo llegar a hacer físicamente inviable la resolución del problema.
El nivel de discretización temporal está íntimamente ligado a la discretización espacial. En este sentido, una herramienta orientativa que debemos tener en cuenta es el número de Courant, que se define como

$$C_o = \frac{\Delta t \cdot U_\infty}{\Delta x} \tag{2.46}$$

Este parámetro, que surge a partir de la casi axiomática terna $v = x/t$, establece la relación entre el paso de tiempo Δt, la velocidad de la corriente libre U_∞ y el tamaño mínimo de la celda Δx de la malla en la dirección de la velocidad. Para una buena precisión y estabilidad numérica, un número de Courant menor a 1 generalmente es lo recomendado, mientras que números de Courant mayores a la unidad carecen de sentido físico.

Capítulo 3

Geometrización de la orografía

3.1. Mallado de un volumen delimitado por una orografía

En general se trata de definir un volumen en forma de cubo cuya base sea la orografía o fondo marino de la zona que nos interesa estudiar y la altura máxima quede delimitada por la superficie del mar, de manera que este cubo represente el volumen de agua que existe entre el fondo marino y la superficie del mar. En el caso concreto de este trabajo, el fluido será agua de mar definida por los parámetros característicos del entorno de las Islas Canarias.

Para que se puedan resolver ecuaciones diferenciales que caractericen un estado en el interior de este volumen, éste se debe discretizar de manera adecuada para resolver las ecuaciones en puntos determinados y característicos. El fruto de esta discretrización es lo que llamamos la malla.

3.1.1. Creando la malla con Gmsh

Gmsh [15] es un software libre de generación de mallas 2D y 3D que incluye un motor CAD y un postprocesador. Gmsh se compone por cuatro bloques: *geometry*, *mesh*, *solver* y *post-processing*. La interacción con estos bloques puede ser interactiva a través de la interfaz gráfica o a través de scripts ASCII en el lenguaje propio de Gmsh.

3.1.2. Obtención de una nube de puntos

Para comenzar el proceso, debemos conocer los datos del terreno que queremos modelar. Estos datos vienen dados en forma de nube de puntos, que nos aportan una información discreta de la elevación del terreno respecto a un nivel de referencia. En este caso, los datos de batimetría son elevaciones negativas

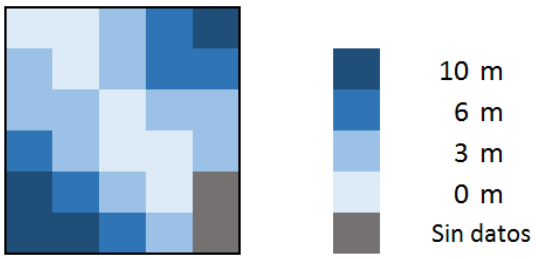

Figura 3.1: Ejemplo de datos de elevación de un terreno

sobre el nivel del mar.
La manera de localizar cada uno de estos datos caracteriza los diferentes formatos que existen.

ESRI ASCII

Se trata de un formato empleado para transferir información entre sistemas basados en celdas o entramados. Los archivos en este formato comenzarán con una cabecera que define las propiedades del entramado de datos y nos aporta la información necesaria para su procesado; esto es: tamaño de la celda, número de filas y columnas, coordenadas de referencia del entramado -esquina superior izquierda (NW) o centro- y el valor que se le asigna a los puntos en los que no se dispone de información.

Esta cabecera tendrá la siguiente forma:

```
NCOLS XXXX
NROWS XXXX
XLLCORNER XXXX | XLLCENTER XXXX
YLLCORNER XXXX | YLLCENTER XXXX
CELLSIZE XXXX
NODATA_VALUE XXXX
fila 1...
```

La cabecera viene precedida por el valor del parámetro –elevación, temperatura, densidad de población, etc.– de cada celda, separada cada columna por espacios y cada fila por un retorno de carro. Esto conforma una matriz de tamaño $NROWS \times NCOLS$. La extensión más común para estos archivos suele ser `.asc`

Se propone a continuación un ejemplo simplificado para entender la dinámi-

ca. Suponiendo que la figura 3.1 representa la elevación de un terreno discretizado, en el que cada celda representa un cuadrado de 0,3 × 0,3 m y su valor es el de la elevación media de esa superficie.
El fichero que contiene esta información en formato ESRI ASCII sería así:

```
NCOLS 5
NROWS 6
XLLCORNER 15,4211
YLLCORNER -28,4412
CELLSIZE 0,0001
NODATA_VALUE 3127
0 0 3 6 10
3 0 3 6 6
3 3 0 3 3
6 3 0 0 3
10 6 3 0 27
10 10 6 3 27
```

.xyz

El formato .xyz es otra de las formas de representar una nube de puntos del tipo *nube de puntos ASCII*. Estos archivos suelen contar con la extensión `.xyz`, `.txt` o `.csv` entre otros.
Se trata de una lista sin encabezado con tantas filas como puntos y cada fila cuenta con tres o más columnas. Las tres primeras son las coordenadas X,Y,Z de cada punto. En la multitud de posibles aplicaciones, las coordenadas pueden ir precedidas de otras columnas que aporten información adicional del punto, como magnitudes escalares (RGB, temperatura, elevación...), etiquetas o valores ordinales. La lista recorre los puntos de Oeste a Este y de Norte a Sur del perímetro establecido.

En el caso del ejemplo propuesto, descrito en el apartado anterior, una representación sencilla de la figura 3.1 sería:

```
15,4211 -28,4412    0
15,4212 -28,4412    0
15,4213 -28,4412    3
15,4214 -28,4412    6
15,4215 -28,4412    10
15,4211 -28,4411    3
15,4212 -28,4411    0
15,4213 -28,4411    3
15,4214 -28,4411    6
15,4215 -28,4411    6
15,4211 -28,4410    3
```

```
15,4212 -28,4410    3
15,4213 -28,4410    0
15,4214 -28,4410    3
15,4215 -28,4410    3
15,4211 -28,4409    6
15,4212 -28,4409    3
15,4213 -28,4409    0
15,4214 -28,4409    0
15,4215 -28,4409    3
15,4211 -28,4408    10
15,4212 -28,4408    6
15,4213 -28,4408    3
15,4214 -28,4408    0
15,4215 -28,4408    27
15,4211 -28,4407    10
15,4212 -28,4407    10
15,4213 -28,4407    6
15,4214 -28,4407    3
15,4215 -28,4407    27
```

A partir de estos datos, lo ideal sería crear una superficie que se ajuste a todos los puntos, extruir las caras laterales y obtener el volumen computacional, para posteriormente ser mallado en el software Gmsh y utilizar esta malla para el cálculo con elementos finitos. Se han probado múltiples alternativas y se ha concluido que este procedimiento no es trivial y conlleva una carga importante de trabajo en software CAD, que después de dedicar muchas horas de la programación se ha llegado a la conclusión de que su aplicación a este Trabajo de Fin de Máster no es viable. No obstante, se subraya como futura línea de investigación y se seguirá trabajando en ello.

3.2. Obtención de un dominio computacional tridimensional a partir de los datos de una línea de costa

En este trabajo se ha procedido a programar y automatizar un procedimiento para aproximar una geometría a la geografía real de un entorno costero.
Para ello es necesario obtener obtener las coordenadas .xyz de la línea de costa que nos interesa estudiar. Se ha escrito un script en Matlab que lee este archivo y solicita al usuario algunos parámetros con los que construirá un fichero .geo que contiene una geometría tridimensional en el lenguaje nativo del software Gmsh y que una vez importada, su mallado será inmediato.
Los datos solicitados al usuario definen de manera grosera algunos aspectos de la geometría final, como son: profundidad en la costa, profundidad en el extremo más alejado de la costa del dominio computacional, tamaño del dominio computacional entendido como la distancia comprendida entre el extremo

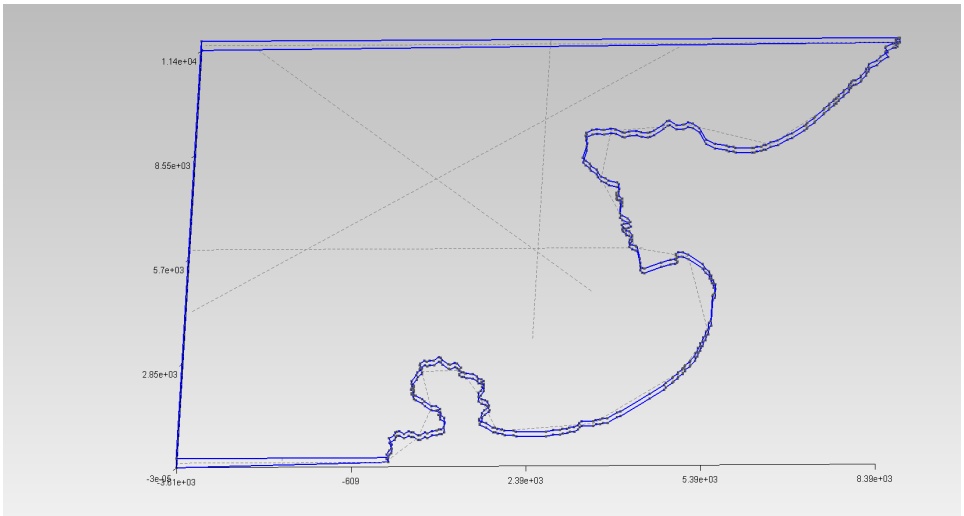

Figura 3.2: Geometría obtenida con el script Matlab a partir de línea de costa de Quintero, Chile.

norte de la línea de costa y el punto más oriental/occidental del contorno.

El código, llamado **escribe_puntos_cc.m** está detallado en el Anexo A de esta memoria. En la figura 3.2 se puede ver un ejemplo de aplicación de este procedimiento en la costa de Quintero (Chile) representada en la figura 3.3.

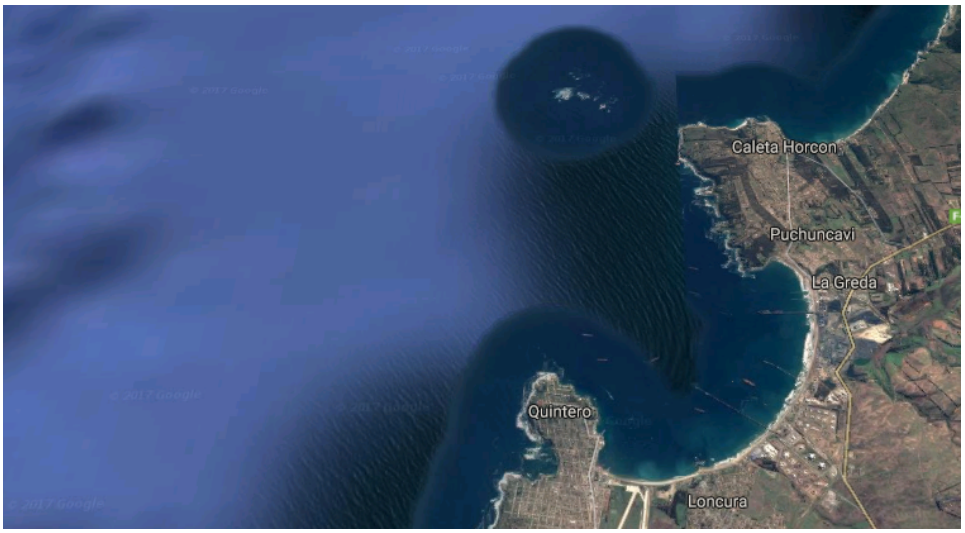

Figura 3.3: Ortofoto satelital del entorno geográfico de Quintero, Chile.

Capítulo 4

Metodología

En este capítulo se explica el proceso seguido para comprobar la corrección y cohesión de todos los pasos y procedimientos descritos anteriormente. En general y a modo de resumen, el proceso general seguido se puede representar mediante el flujograma de la figura 4.1.

4.1. Preproceso: Geometría y mallado de zona costera

La geometría utilizada para el caso test consiste en un volumen rectangular en una de sus caras delimitado por la línea de costa que comprende los alrededores de la central térmica de Barranco de Tirajana, en Arinaga [1] (ver figura 4.2).

La línea de costa se obtuvo a partir de los datos del portal REDMIC[2] del Observatorio Ambiental de Granadilla [3]. Estos datos se descargaron en formato de mapa ráster y convertidos a .xyz con el software libre QGIS [4].
Esta lista de puntos .xyz se introdujo en el código de Matlab descrito en el apartado 3.2 del que se obtuvo la geometría en formato Gmsh como se muestra en la figura 4.3. A continuación se realizó el mallado con el mismo software y se obtuvo la malla regular no estructurada de cinco capas de nodos y refinada en los contornos de la costa, que se muestra en las figuras 4.4 y 4.5.

[1] http://www.prtr-es.es/informes/fichacomplejo.aspx?Id_Complejo=1795

[2] El Repositorio de Datos Marinos Integrados de Canarias es un portal que proporciona datos libres en las aguas de las Islas Canarias. www.redmic.es

[3] El OAG es una fundación pública estatal creada en 2008 por la Autoridad Portuaria de Santa Cruz de Tenerife y el Gobierno de Canarias. www.oag-fundacion.org

[4] QGIS es un Sistema de Información Geográfica (SIG) de Código Abierto licenciado bajo GNU. www.qgis.org

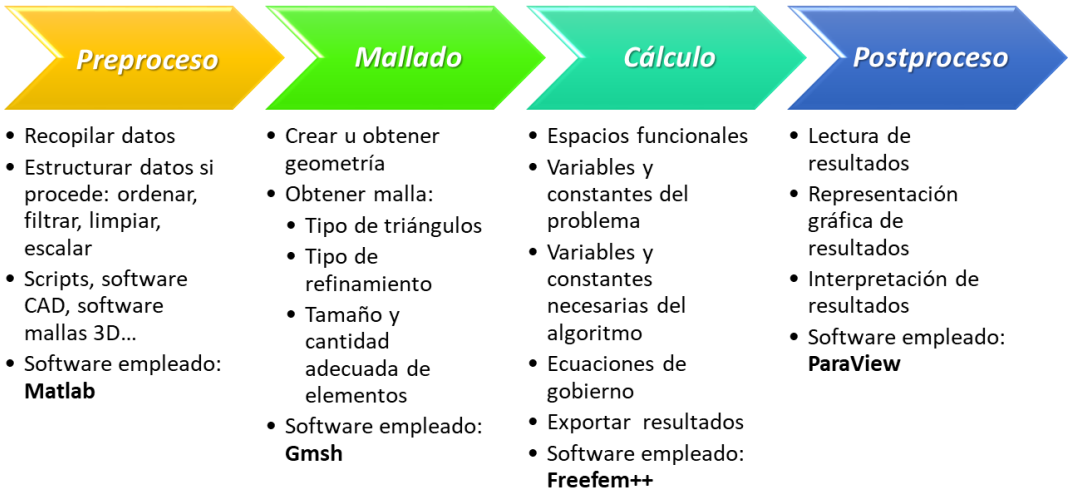

Figura 4.1: Flujograma del proceso seguido para realizar el caso test de esta memoria

Figura 4.2: Ortofoto satelital del entorno geográfico de la central térmica de Barranco de Tirajana en Arinaga.

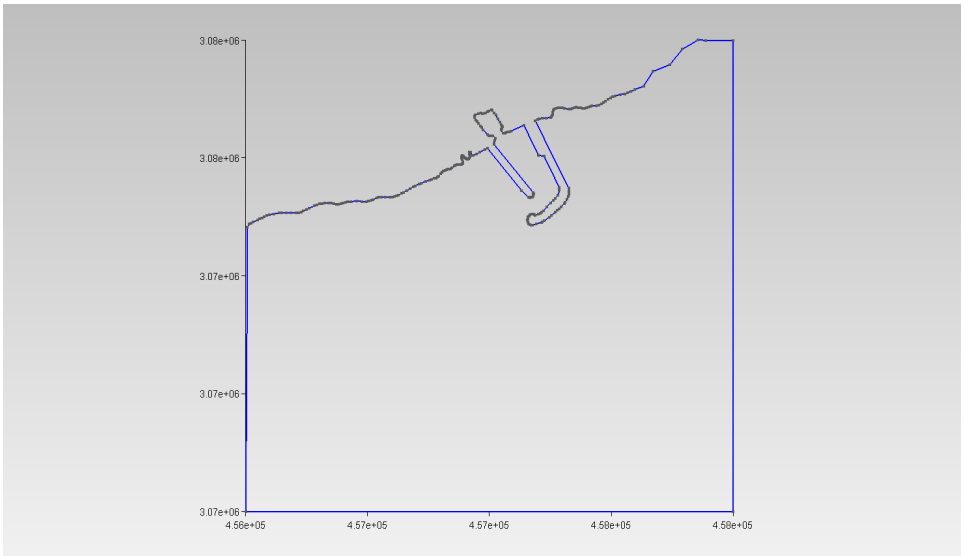

Figura 4.3: Geometría extraida del script Matlab a partir de la línea de costa

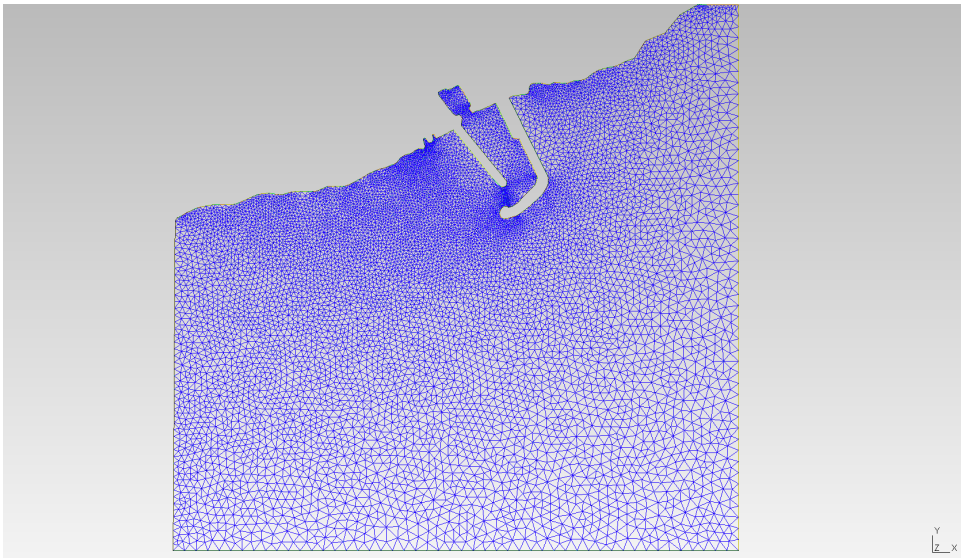

Figura 4.4: Mallado 3D del dominio en vista de planta

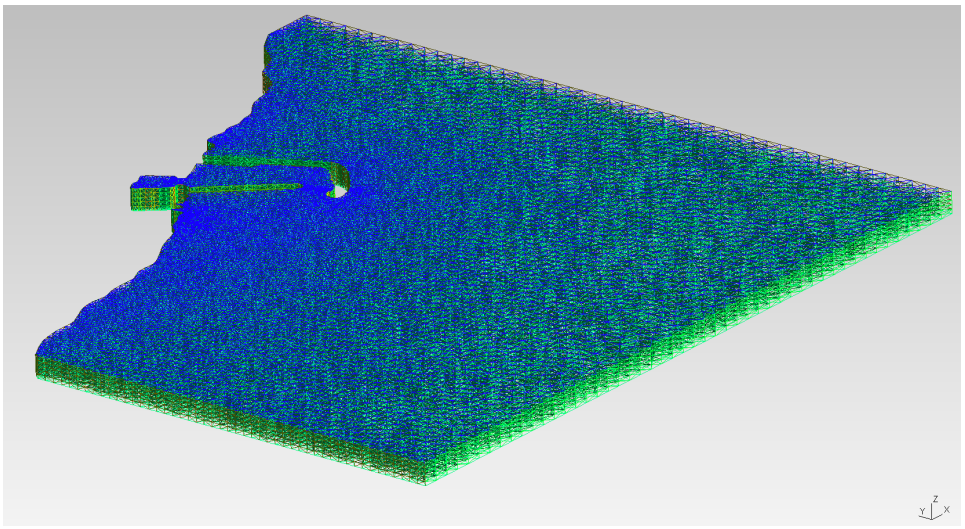

Figura 4.5: Mallado 3D del dominio en vista perspectiva

4.2. Cálculo

Para los cálculos se ha desarrollado un programa FreeFem++ en consonancia a todo lo descrito en el capítulo 2. A modo de resumen, el programa realiza las siguientes acciones:

1. Importa la malla desde archivo .msh
2. Cálculo MMC
   - *a*) Se declaran espacios funcionales y variables necesarias para cálculo de Matriz de Masa Consistente (MMC)
   - *b*) Se declaran las localizaciones y las medidas de cada estación de medida empírica de corriente y viento
   - *c*) MMC: interpolación y cálculo sobre el dominio y comprobación de divergencias
   - *d*) Guardar y exportar datos en formato .vtk
3. Algoritmo de Chorin-Rannacher
   - *a*) Se declaran espacios funcionales y se toman como condiciones iniciales los resultados de MMC
   - *b*) Declaración de variables necesarias para el modelo físico y para el algoritmo
   - *c*) Inicia el bucle para cálculo evolutivo de soluciones
   - *d*) Guarda valores de la solución anterior y calcula los términos convectivos

e) Calcula cada componente de la velocidad por separado teniendo en cuenta condiciones de contorno

f) Calcula presión, temperatura y ecuación de momento correspondiente con la presión calculada en iteración anterior. Actualiza valores finales.

g) Exporta resultados formato .vtk cada cierto número de iteraciones

La ejecución del programa requiere caracterizar muchas propiedades del océano para la veracidad del modelo. Las propiedades significativas que se han declarado para el caso test del entorno costero de Arinaga vienen definidas en la tabla 4.1.

Concepto	**Nombre variable**	**Valor**	**Fuente**
Coeficiente de Ekman	`a`	0.038	González (2001)
Salinidad Canarias g/m^3	`S`	0.36	[4]
Temperatura de referencia K	`T0`	16	
Densidad kg/m^3	`rho(T,S)`	Ecuación (2.8)	
Viscosidad cinemática m^2/s	`nu`	`0.001/ rho(TBou,S)`	
Calor específico p=cte $J/(KgK)$	`Cp`	3991.86	IOC (2010)
Conductividad térmica $W/(mK)$	`K`	0.596	NPL [5]
Coeficiente de expansión térmica	`beta`	0.1655	IOC (2010)

Tabla 4.1: Tabla de propiedades empleadas para el modelo físico del programa

Parámetros del algoritmo

Los parámetros empleados para el correcto funcionamiento del algoritmo deben ser cuidadosamente seleccionados y en algunos casos esta elección obedece a la experiencia del programador en un proceso informal. Es el caso, por ejemplo, del paso de tiempo que tiene un carácter decisivo en la convergencia del algoritmo evolutivo y que, aunque podemos delimitarlo a través del número de Courant, su ajuste muchas veces debe hacerse a través de la experiencia con el propio programa y de su comportamiento. En nuestro caso test, el paso de tiempo se ha fijado en `dt = 0.01`. Esto es, que en cada segundo se calculan cien soluciones. Este pequeño valor se debe al carácter turbulento que se ha detectado en el flujo del problema y al pequeño tamaño que tienen los elementos que componen la discretización numérica de la geometría.

Por otro lado, en la cabecera del programa se han cargado algunos paquetes adicionales, como son:

load 'gmsh' Se añade para poder leer ficheros en formato .msh nativos del software GMSH

load 'iovtk' Paquete que se utiliza para exportar la malla y la solución en formato .vtk y su representación gráfica en software ParaView

load 'msh3' Utilizado también para la importación y tratamiento de mallas tridimensionales

load 'MUMPS' Este paquete consiste en una librería que contiene resolvedores lineares para grandes sistemas compuestos por matrices tipo "sparseçon factorizaciones tipo LU y L^TDL. Esta librería además está preparada para paralelizar los procesos en las CPU del equipo.

En referencia al último punto cabe destacar que utilizando la función `solver=LU` incluida en la librería MUMPS se obtiene un muy buen rendimiento a la hora de resolver los sistemas que despeja velocidades y temperaturas.

Condiciones de contorno del caso test

En la malla se han declarado condiciones de contorno tipo Dirichlet y Neumann, según lo descrito en el apartado 2.5 de esta memoria. En la tabla 4.2 se definen las etiquetas dadas a los contornos de la malla de la figura 4.4.

Nombre	Contorno	C.C. en MMC	C.C. en Chorin
Γ_1	Línea de costa	Velocidad tangencial	No deslizamiento
Γ_2	Cara este	Contorno libre	Solución de MMC
Γ_3	Cara sur	Contorno libre	Contorno libre
Γ_4	Cara oeste	Contorno libre	Contorno libre
Γ_5	Fondo	Velocidad tangencial	No deslizamiento
Γ_6	Superficie	Contorno libre	Sol. MMC + viento

Tabla 4.2: Condiciones de contorno impuestas en el modelo de Matriz de Masa Consistente y en el algoritmo evolutivo de Chorin, respectivamente

4.3. Postproceso y resultados

Para la representación y visualización de los resultados, el propio `Freefem++` dispone del paquete "medit"que representa de manera sencilla los resultados. No obstante, por su mayor versatilidad y prestaciones se ha decidido utilizar para esta tarea el software ParaView.[6] Para exportar los resultados a un archivo de extensión .vtk que pueda abrirse en ParaView, en `Freefem++` se utiliza la siguiente línea de código:

[6]ParaView es una aplicación multiplataforma de código abierto para el análisis y visualización de datos. www.paraview.org

```
savevtk("VeloChorin-"+iter+".vtk",Th,[u1,u2,u3],...
dataname = "VelocidadChorin");
```

Este comando creará un archivo llamado `VeloChorin1.vtk`, con el número que corresponda a la iteración, y que contiene la malla `Th` y las magnitudes `u1,u2,u3` agrupadas en una variable llamada *VelocidadChorin*.

- Lo primero que veremos en la interfaz es la malla coloreada según el valor de la velocidad en cada elemento (figura 4.6). Se puede aplicar el filtro `Cell Data to Point Data` para interpolar los valores de la magnitud en cada nodo a todos los puntos para "suavizar"los resultados y hacerlos más atractivos (el rigor depende del tamaño de los elementos). También podemos hacer un corte transversal a la malla y observar las velocidades a la profundidad deseada (figura 4.7).

- Con el comando `Glyph` se representa la velocidad de manera vectorial a través de flechas (figura 4.8) pudiendo colorear los vectores por magnitud de velocidad (figura 4.9).

- Con el comando `Stream Tracer` podemos definir y ubicar una esfera y representar, con la densidad deseada, las líneas de corriente que la atraviesan. De este modo podemos ver la interacción de la corriente en las zonas que nos interesen, por ejemplo, en la bocana del dique (figura 4.10).

- Con el comando `Plot Over Line` podemos definir y ubicar una línea y representar gráficamente la velocidad (tanto en magnitud como separada por componentes) del flujo que atraviesa la línea (figura 4.11).

- En cuanto a la temperatura, impuesta la solución en el fondo (15^oC) y en la superficie (20^oC) podemos ver representado el campo escalar como superficies isotermas. Para ello, primero debemos aplicar el filtro `Cell Data to Point Data` para interpolar y homogeneizar la solución en todo el dominio y luego se aplica `Contour` para representar estas isosuperficies (figura 4.12)

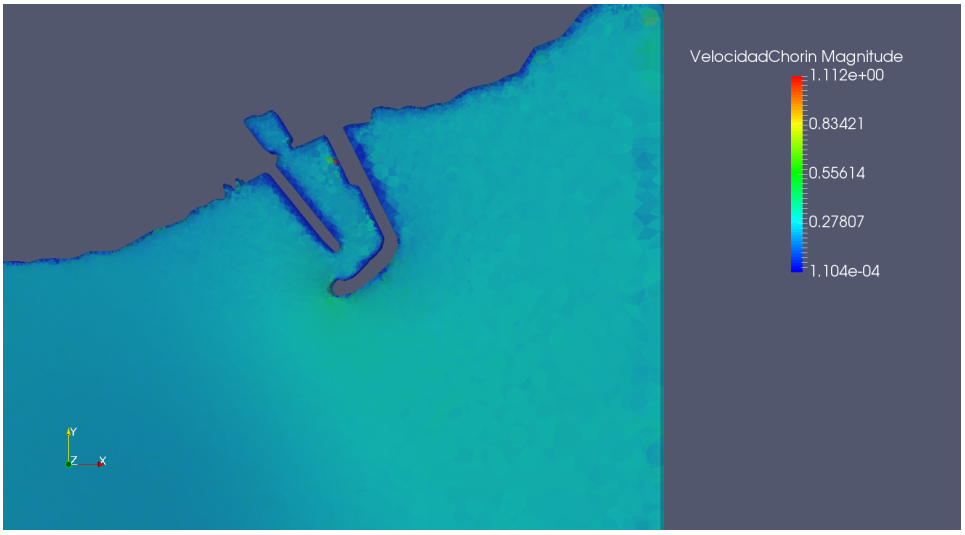

Figura 4.6: Malla coloreada por magnitud de velocidad en superficie

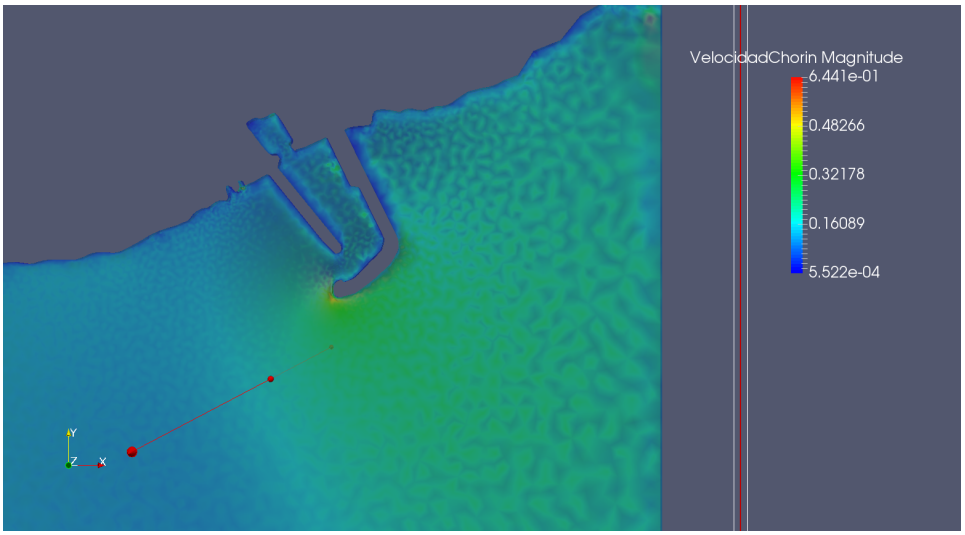

Figura 4.7: Malla coloreada por magnitud de velocidad a 25 metros de profundidad

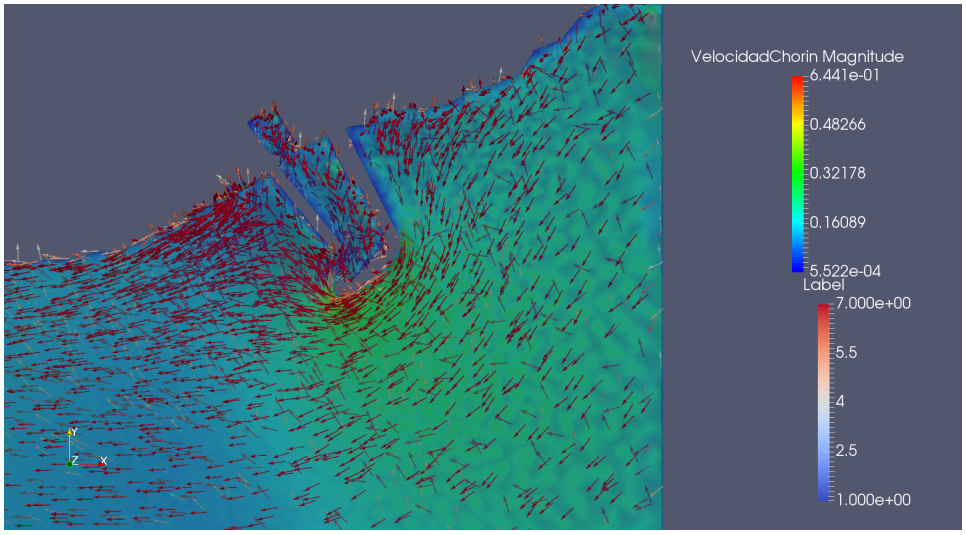

Figura 4.8: Representación uniforme de los vectores de velocidad

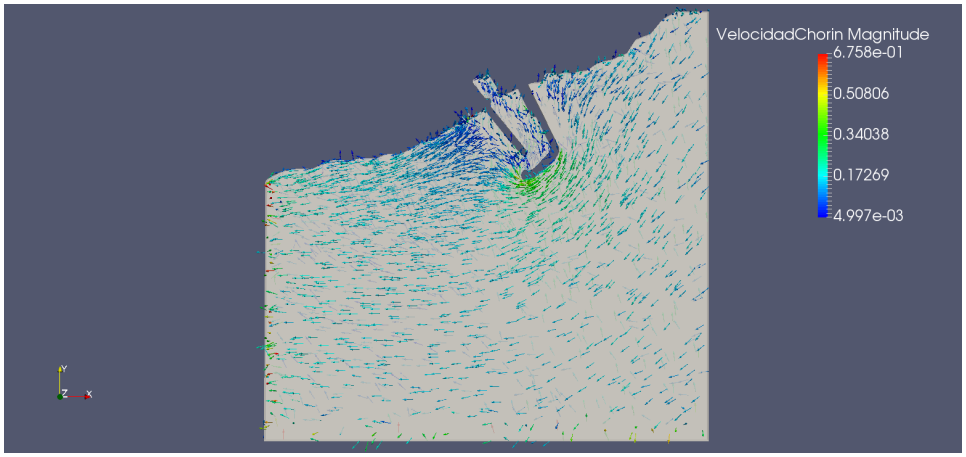

Figura 4.9: Representación uniforme de los vectores de velocidad coloreados por magnitud de velocidad

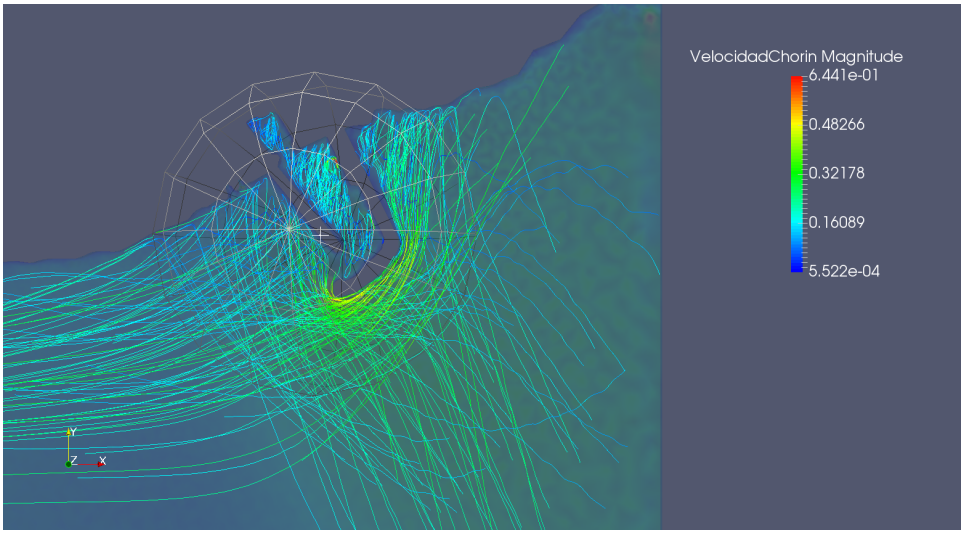

Figura 4.10: Representación de las líneas de corriente

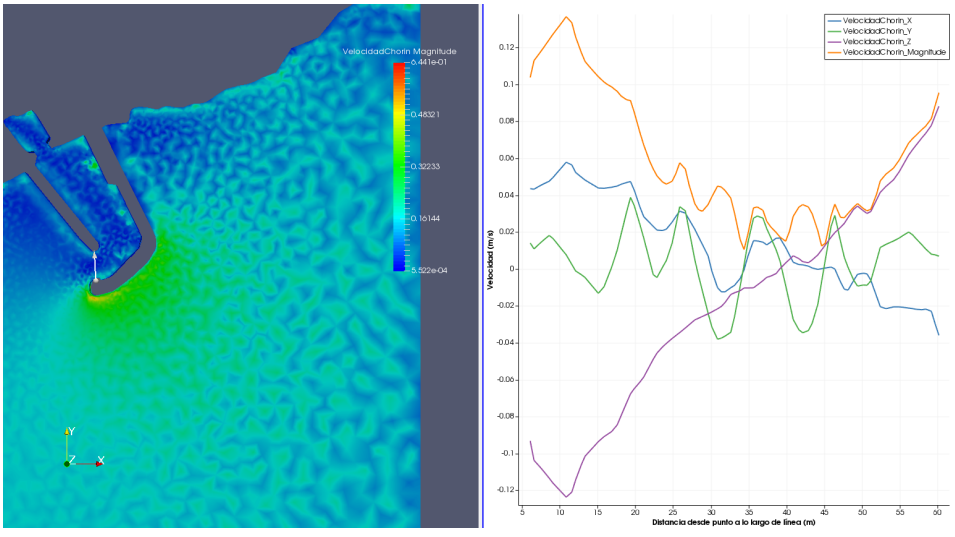

Figura 4.11: Representación gráfica de la velocidad a lo largo de una línea dispuesta en la bocana del espigón

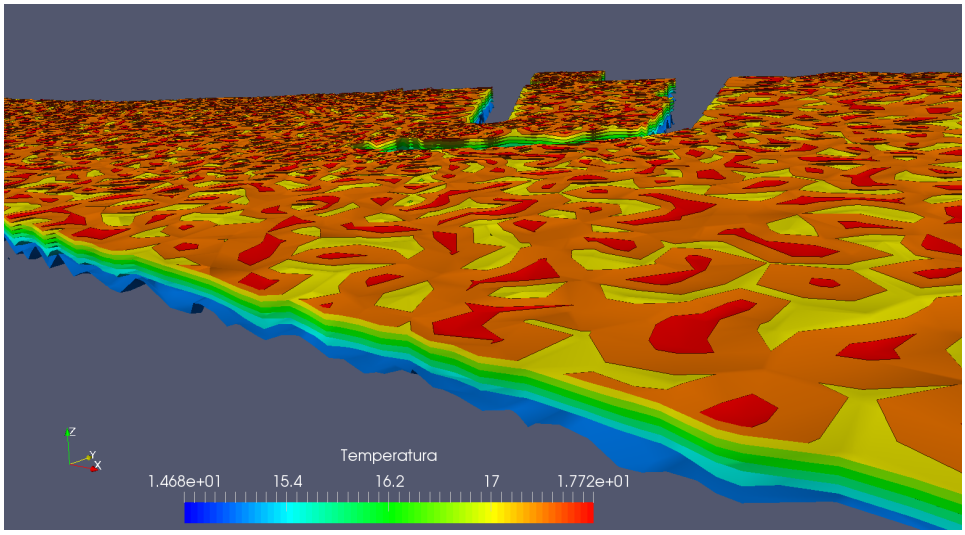

Figura 4.12: Isosuperficies de temperatura

Capítulo 5

Conclusiones

Este trabajo no pretende dar por concluido ningún procedimiento, todo lo contrario, pretende contribuir a un campo que ya está abierto y en el que hay aún mucho camino por recorrer. A lo largo del tiempo y de las pruebas realizadas hemos aprendido lo que se puede hacer y lo que se puede hacer con más esfuerzo del esperado, siempre con el convencimiento de que no existen tareas categóricamente imposibles.

- El proceso de obtención de datos para generar contornos computacionales a partir de datos de batimetrías es tedioso y poco accesible, convirtiéndose por lo general en un proceso de minería o de transformación de datos.

- El proceso de generar superficie y volumen es una tarea que puede realizarse mediante software CAD o bien tratando directamente los datos, de una manera más cruda y a través de algún entorno de *scripting* como Matlab o Phyton. Ambas tienen sus beneficios y perjuicios, y sin embargo, en futuras líneas de trabajo se apostará por la segunda opción ya que ofrece mayor posibilidad de "automatizar" el proceso.

- Se ha implementado satisfactoriamente en Freefem++ el modelo de Matriz de Masa Consistente con elementos finitos en un espacio de tres dimensiones, aplicando interpolación en el plano XY y efectos de Ekman en el eje Z.

- Se ha integrado en el código Freefem++ el algoritmo de Chorin-Rannacher como método de resolución directa (DNS) de las ecuaciones de Navier-Stokes, constatando el menor coste computacional gracias al empleo de polinomios de primer grado, P1, frente a los polinomios de grado dos presentes en las ecuaciones primitivas.

- Al algoritmo de Chorin-Rannacher se ha añadido satisfactoriamente un cálculo de la difusión de la temperatura y su efecto sobre el campo de velocidades a través de la aproximación de Boussinesq.

- Se presentan las soluciones de manera gráfica y visualmente atractivas, incluso interactivas, para su mejor entendimiento, tratamiento y difusión. No debemos olvidar que la transferencia es un pilar importante de la ciencia.

5.1. Limitaciones del modelo y futuras líneas de trabajo

Este Trabajo de Fin de Máster está contextualizado en una de las líneas de trabajo del proyecto de investigación CEI2017-14, enmarcado en el Campus de Excelencia Internacional, por lo que se tendrá la oportunidad de poder continuar trabajando sobre la mejora de las líneas ya tratadas y sobre las que están por abrirse. De manera detallada éstas son:

- En un principio se pretendía que el modelo numérico se ejecutara sobre un dominio computacional que contuviera la forma real del fondo marino. Se ha constatado que ésta es una tarea a la que hay que dedicar un número de horas notablemente superior a las que están programadas para un trabajo de fin de máster, y por tanto ha pasado a proponerse como línea con la que seguir trabajando.

- En relación al modelo numérico, puede incorporarse cerca de la costa el comportamiento de la corriente representado mediante fórmulas costeras empíricas que pueden ser del mismo modo implementadas en nuestro modelo a modo de datos in situ para considerar más variables en el problema, como periodo de ola o fricción con el fondo, lo que enriquecerá las simulaciones en zonas costeras.

- En cuanto a los modelos de turbulencia, desde una resolución directa (DNS) hasta un modelo RANS (Reynolds-Averanged Navier-Stokes) o LES (Large Eddy Simulation) para las diferentes escalas, pueden ser implementados para una mayor precisión de los resultados cuando aparecen estos fenómenos, ya sea en accidentes costeros abruptos como espigones o diques, rompeolas o estructuras sumergidas.

- La implementación de modelos de dispersión de contaminantes pueden ser añadidos con el fin de predecir su comportamiento teniendo en cuenta unos parámetros medioambientales realistas: orografía y batimetría, corriente y viento, salinidad y temperatura, etc.

- Un procedimiento riguroso y bastante automatizado con el que pasar de la nube de puntos del fondo hasta el modelo matemático fiel implementado en Freefem++, para obtener unos resultados que sean fácilmente

visualizables e interpretables, es la finalidad a la que este trabajo pretende contribuir.

Bibliografía

[1] J.M. Fernández Oro. *Técnicas Numéricas en Ingeniería de Fluidos. Introducción a la dinámica de fluidos computacional (CDF) por el método de los volúmenes finitos*. Reverté, 2012.

[2] J.H. Ferziger and M. Peric. *Computational Methods for Fluid Dynamics*. 3 ed. Berlin. Springer, 2002.

[3] Instituto Hidrográfico de Cantabria. Modelo hidrodinámico bidimensional. AQUALAB, Manual de Referencia.

[4] Oceanográfica; Gobierno de Canarias; ULPGC; Bioges. Nuestro mar canario.

[5] Gabriel Winter y Begoña González. Ecuaciones en medio ambiente: Navier-stokes, ecuación de energía y de trasnporte convectivo-difusivo. 2014. división de computación evolutiva ceani.

[6] Hans Burchard. *Applied Turbulence Modelling in Marine Waters*. Springer, 1959.

[7] Matthias Heil and Andrew Hazel. Boussinesq convection: Combining the navier-stokes and advection-diffusion equations.

[8] www.comsol.com. Multiphysics cyclopedia.

[9] F. Hecht. New development in freefem++. *J. Numer. Math.*, 20(3-4):251–265, 2012.

[10] Alexandre Joel Chorin. A numerical method for solving incompresible viscous flow problems. *Journal of Computational Physics*, (135):118–125, 1997.

[11] Rolf Rannacher. On chorin's projection method for the incompresible navier-stokes equations. *Universität Heidelberg. Institut für Angewandte Mathematik*.

[12] Begoña González Landín. *Determinación de localizaciones óptimas de puntos múltiples de vertidos de aguas residuales en zonas costeras mediante algoritmos genéticos*. PhD thesis, Departamento de Matemáticas, ULPGC, 2001.

[13] Ricardo G. Durán. Galerkin approximations and finite element methods. 2014.

[14] Manuel J. Chica González. Estimación computacional de la resistencia al avance de un cuerpo simple. Master's thesis, EIIC, UPGC, 2015.

[15] C. Geuzaine y J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities.

Anexo A

Programas Matlab

import_data.m

Código Matlab de importación de datos en forma de matriz para su posterior tratamiento.

```
% Script for importing data from text file:
%
% Auto-generated by MATLAB on 2017/10/26 10:50:55
%
%% Initialize variables.
global filename

filename;
%filename = 'E:\Master\TFM\matlab\lc_arinaga_ok.xyz';
delimiter = ',';

%% Format string for each line of text:
%   column1: double (%f)
%   column2: double (%f)
%   column3: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%f%f%f%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to format string.
% This call is based on the structure of the file used to
% generate this code. If an error occurs for a different file,
%  try regenerating the code from the Import Tool.
dataArray = textscan(fileID, formatSpec, 'Delimiter', delimiter,
 'MultipleDelimsAsOne', true,  'ReturnOnError', false);

%% Close the text file.
fclose(fileID);
```

```
%% Post processing for unimportable data.
% No unimportable data rules were applied during the import,
% so no post processing code is included. To generate code
% which works for unimportable data, select unimportable cells
% in a file and regenerate the script.

%% Create output variable
M = [dataArray{1:end-1}];  % Los datos se guardan en
% una matriz llamada M
%% Clear temporary variables
clear filename delimiter formatSpec fileID ans dataArray;
```

script_distancias.m

```
% Creado por Roman Grau Beranger en CEANI, noviembre 2017.
%
% Este programa importa datos en formato .xyz georeferenciados
% y los traslada a un origen de coordenadas situado en el extremo
% noroccidental, pasando ademas las coordenadas geograficas
% decimales a metros, para consonancia de unidades.
% Los nuevos datos se guardan en un fichero generado automatica-
% mente y llamado "coastline_ref.asc"
% -----------------------------------
% ESTA APROXIMACION SOLO ES VALIDA EN EL
% ENTORNO GEOGRAFICO DE GRAN CANARIA
% -----------------------------------

clear all
clc
import_data %importa los datos de entrada. Modificar a parte


%----- NO DATA VALUE---------
ndv = -32767;
%----------------------------


%load('M.mat')
M = coastline;

[filas,columnas] = size(M);

% Valores de referencia. Los mayores, es decir, el mas NW

refX = M(1,1);
refY = M(1,2);

% Matriz referenciada

Mref(:,1) = M(:,1)-refX;
Mref(:,2) = M(:,2)-refY;
Mref(:,3) = M(:,3);

% Entre el 28N y el 29N hay 111133.33 m
% Entre el 15W y el 14W hay 98289.142 m

% Adapto el eje x, primera columna

for f = 1:filas

    Mref(f,1) = Mref(f,1)*111133.33;
    Mref(f,2) = Mref(f,2)*98289.142;
```

```
end

%% Nombre y extension de archivo de salida
fid = fopen('coastline_ref.asc','w');
%% Cabecera (opcional)
% fprintf(fid,'%s','_MULTIPLE _POINT');
% fprintf(fid,'\n');
%%

for i = 1:size(Mref,1)

    if Mref(i,3) ~= ndv
    fprintf(fid,'%.5f,',Mref(i,1));
    fprintf(fid,'%.5f,',Mref(i,2));
    fprintf(fid,'%.5f',Mref(i,3));
    end
    fprintf(fid,'\n');

end
fclose(fid)
```

escribe_puntos_cc.m

```
% Creado por Roman Grau Beranger en CEANI, noviembre 2017.
% Este programa lee un fichero .xyz de la linea de costa en 1D
% y exporta la geometria para su lectura en GMSH (.geo) con
% fondo inclinado.
% Superficies deben declararse en sentido ANTIhorario
% ESTE PROGRAMA ES VALIDO PARA COSTAS QUE DAN AL ESTE

clear all
clc
import_data %importa los datos de entrada. Modificar a parte

[filas,columnas] = size(M);

%% Nombre y extension de archivo de salida
fid = fopen('lc_arinaga.geo','w');
%% Cabecera (opcional)
% fprintf(fid,'%s','_MULTIPLE _POINT');
% fprintf(fid,'\n');
%% Genera la primera serie de puntos que corresponde al contorno
% de la costa a nivel del mar (0 m)
contador = 1;
for i = 1:filas

    fprintf(fid,'Point(%i) = {%.5f, %.5f, 0, 100};\n', ...
    contador,M(i,1),M(i,2));
    contador = contador+1;
end
cont1 = contador;
% --------------------------
% Puntos de la capa inferior. PONER LA PROF. DE LA COSTA
% --------------------------


for i = 1:filas

    fprintf(fid,'Point(%i) = {%.5f, %.5f, -10};\n',contador, ...
    M(i,1),M(i,2)); % <-- AQUI
    contador = contador+1;

end
cont2 = contador;



% --------------------------
%
% Spline de la capa sup e inf
% --------------------------
```

```
n = [1:cont1-2];
fprintf(fid,'//+\nLine(1) = {');
fprintf(fid,'%i, ',n);
fprintf(fid,'%i};\n',n(end)+1);



n = [cont1:cont2-2];
fprintf(fid,'//+\nLine(2) = {');
fprintf(fid,'%i, ',n);
fprintf(fid,'%i};\n',n(end)+1);


% ---------------------------------
% Lineas que cierran los contornos
% ---------------------------------

% Principio
fprintf(fid,'//+\nLine(3) = {');
fprintf(fid,'1, %i};\n',cont1); %Siempre une el punto #1 con
% el punto n+1 (primero de la segunda fila)
% Fin
fprintf(fid,'//+\nLine(4) = {');
fprintf(fid,'%i, %i};\n',n(end)+1,cont1-1);

% Loop
fprintf(fid,'//+\nLine Loop(5) = {3, 2, 4, -1};\n'); %El loop,
% segun lo programado, siempre sera igual
% Surface
fprintf(fid,'//+\nRuled Surface(6) = {5}; // superficie ...
de la costa \n');
%El loop, segun lo programado, siempre sera igual

% Llegados a este punto, basta con abrir el .geo y darle a
% mesh 2D para mallar el contorno de la costa.

%% Volumen computacional a partir de la linea de costa

offset = 2000; % Distancia (m) del contorno computacional del
% primer punto de la costa.
% Ojo: valores positivos son contornos generados hacia el este
% (costa a la derecha, como Quintero)y
% valores negativos son contornos generados hacia el oeste (costa
% a la izquierda, como muelle de la luz).
prof_fuera = -50;
% Profundidad (m) en el extremo del dominio computacional

% Genera los puntos:

% Estremo norte superficie
    fprintf(fid,'//+\nPoint(%i) = {%.5f, %.5f, 0};\n', ...
```

```
    contador, M(1,1)+offset,M(1,2));
    idN = contador; % ID punto N superficie
    contador = contador+1;

% Estremo norte fondo
    fprintf(fid,'//+\nPoint(%i) = {%.5f, %.5f, %i};\n', ...
    contador, M(1,1)+offset,M(1,2),prof_fuera);
    contador = contador+1; %idNsup+1

% Estremo sur superficie
    fprintf(fid,'//+\nPoint(%i) = {%.5f, %.5f, 0};\n', ...
    contador, M(1,1)+offset,M(end,2));
    idS = contador; % ID punto S superficie
    contador = contador+1;

% Estremo sur fondo
    fprintf(fid,'//+\nPoint(%i) = {%.5f, %.5f, %i};\n', ...
    contador, M(1,1)+offset,M(end,2),prof_fuera);
    contador = contador+1; %idSsup+1

%% Lineas que cierran el contorno por el norte

% Norte
fprintf(fid,'//+\nLine(7) = {1, %i};\n',idN); % Es necesario
% definir cada superficie con al menos tres lineas diferentes
% para poder identificar la direccion de la
% normal. Se hacen en este caso por separado
% para evitar duplicidades
fprintf(fid,'//+\nLine(8) = {%i, %i};\n',idN,idN+1);
fprintf(fid,'//+\nLine(9) = {%i, %i};\n',idN+1,cont1);
fprintf(fid,'//+\nLine Loop(10) = {7, 8, 9, -3};\n');
% 3 es negativo porque en su momento
% se definio hacia abajo y ahora lo necesitamos hacia arriba.
% El orden de los puntos a la hora de definir la linea.
fprintf(fid,'//+\nPlane Surface(11) = {10}; ...
//surf cara norte\n');

% Lateral (este u oeste). OJO, SE ESTA PROGRAMANDO
% PARA COSTA A LA DCHA. IGUAL CON COSTA A LA
%IZQ HAY PROBLEMAS CON LAS NORMALES EN ESTE APARTADO.
fprintf(fid,'//+\nLine(12) = {%i, %i};\n',idN,idS);
fprintf(fid,'//+\nLine(13) = {%i, %i};\n',idS,idS+1);
fprintf(fid,'//+\nLine(14) = {%i, %i};\n',idS+1,idN+1);
fprintf(fid,'//+\nLine Loop(15) = {12, 13, 14, -8};\n');
fprintf(fid,'//+\nPlane Surface(16) = {15}; ...
//surf cara lateral\n');

% Sur
fprintf(fid,'//+\nLine(17) = {%i, %i};\n',idS,cont1-1);
fprintf(fid,'//+\nLine(18) = {%i, %i};\n',cont2-1,idS+1);
fprintf(fid,'//+\nLine Loop(19) = {17, -4, 18, -13};\n');
fprintf(fid,'//+\nPlane Surface(20) = {19}; //surf cara sur\n');
```

```
% Tapa superior
fprintf(fid,'//+\nLine Loop(21) = {-7, 1, -17, -12};\n');
fprintf(fid,'//+\nPlane Surface(22) = {21}; ...
//surf cara superior\n');

% Tapa inferior
fprintf(fid,'//+\nLine Loop(23) = {-2, -9, -14, -18};\n');
fprintf(fid,'//+\nPlane Surface(24) = {23}; ...
//surf cara inferior\n');

% Volumen
fprintf(fid,'//+\nSurface Loop(25) = {6, 11, 16, 20, 22, 24};\n');
fprintf(fid,'//+\nVolume(1) = {25}; //volumen\n');

fprintf(fid,'\n/*Si da problemas con interseccion, ...
cambiar algoritmo...
de mallado en Options->mesh->general*/\n\n');

%% Etiquetas de elementos fisicos
fprintf(fid,'//+\nPhysical Surface("superior") = ...
{22};\nPhysical Surface("inferior") = {24};...
\nPhysical Surface("costa")...
 = {6};\nPhysical Surface("norte") = {11};\n');
fprintf(fid,'Physical Surface("lateral") = {16}; ...
\nPhysical Surface("sur") ...
 = {20};\nPhysical Volume("dominio") = {1};');

fclose(fid)
fclose all
```

creaGEO_2D_multilinea.m

```
% Creado por Roman Grau Beranger en CEANI, noviembre 2017.
%
% Este programa lee un fichero .xyz de la linea de costa en 1D
% y exporta la geometria para su lectura en GMSH (.geo).
% Fondo plano (profundidad constante)
% Los puntos de la costa no forman una larga linea sino que
% se generan multiples caras entre cada 4 puntos.
%
% Superficies deben declararse en sentido ANTIhorario
% ESTE PROGRAMA ES VALIDO PARA COSTAS QUE DAN AL ESTE
clear all
clc

global filename

% Input:
filename = 'E:\Master\TFM\matlab\lc_arinaga_peque.xyz';

% Output
outname = 'arinaga_peque.geo';

import_data %importa los datos de entrada. Modificar a parte

% OPCIONES DE ENTRADA

offset = 1500; % Distancia (m) del contorno computacional
% del primer punto de la costa.
% Ojo: valores positivos para costas que dan hacia el este.
% Valores negativos para costas que dan al oeste

[filas,columnas] = size(M);

% Nombre y extension de archivo de salida
fid = fopen(outname,'w');

% Cabecera (opcional)
% fprintf(fid,'%s','_MULTIPLE _POINT');
% fprintf(fid,'\n');

% Genera la primera serie de puntos que corresponde
% al contorno de la costa a nivel del mar (0 m)

contador = 1;

for i = 1:filas

    fprintf(fid,'Point(%i) = {%.5f, %.5f, 0, 100};\n', ...
    contador,M(i,1),M(i,2));
    contador = contador+1;
```

```
end
cont1 = contador;

for i = 1:filas-1

    fprintf(fid,'Line(%i) = {%i,%i};\n',i,i,i+1);
end
contLin = i+1; % Guarda el valor del numero de
% puntos+1 que conforman la costa

% Contorno computacional a partir de la linea de costa

% Genera los puntos:
% Extremo norte superficie
    fprintf(fid,'//+\nPoint(%i) = {%.5f, %.5f, 0};\n', ...
    contador,M(1,1)+offset,M(1,2));
    idN = contador; % ID punto N superficie
    contador = contador+1;

% Extremo sur superficie
    fprintf(fid,'//+\nPoint(%i) = {%.5f, %.5f, 0};\n', ...
    contador,M(1,1)+offset,M(end,2));
    idS = contador; % ID punto S superficie
    contador = contador+1;

% Cierra la superficie con lineas
 % Linea norte
fprintf(fid,'//+\nLine(%i) = {%i, 1};\n',contLin,idN);
norte = contLin;
contLin = contLin+1;

 % Linea este
fprintf(fid,'//+\nLine(%i) = {%i, %i};\n',contLin,idS,idN);
este = contLin;
contLin = contLin+1;

 % Linea sur
fprintf(fid,'//+\nLine(%i) = {%i, %i};\n',contLin,cont1-1,idS);
sur = contLin;
contLin = contLin+1;

vectElementos = [1:filas+1];
% Superficie
fprintf(fid,'//+\nLine Loop(1) = {');
fprintf(fid,'%i, ',vectElementos);
fprintf(fid,'%i};\n',filas+2);

fprintf(fid,'//+\nPlane Surface(1) = {1};\n');

fclose(fid)
fclose all
```

Anexo B

Código Freefem++

MMC_Chorin_Temp.edp

```
/* ------------------------------------------------
 Roman Grau, Gabriel Winter, Begona Gonzalez
 Instituto SIANI division CEANI, noviembre 2017
 MMC 3D + Chorin-Rannacher + Temperatura

//------------------------------------------------
EN ESTE PROGRAMA SE REALIZAN LAS SIGUIENTES OPERACIONES:
1.- Importa la malla desde archivo .msh
2.- Calculo MMC
    2.1.- Se declaran espacios funcionales y variables necesarias
    para calculo de Matriz de Masa Consistente (MMC)
    2.2.- Se declaran las localizaciones y las medidas de cada
     estacion de medida empirica de corriente y viento
    2.3.- MMC: interpolacion y calculo sobre el dominio y
    comprobacion de divergencias
    2.4.- Guardar y exportar datos en formato .vtk
3.- Algoritmo de Chorin-Rannacher
    3.1.- Se declaran espacios funcionales y se toman como
    condiciones iniciales los resultados de MMC
    3.2.- Declaracion de variables necesarias para el modelo
    fisico y para el algoritmo
    3.3.- Inicia el bucle para calculo evolutivo de soluciones
    3.4.- Guarda valores de la solucion anterior y calcula los
    terminos convectivos
    3.5.- Calcula cada componente de la velocidad por separado
    teniendo en cuenta condiciones de contorno
    3.6.- Calcula presion y temperatura. Actualiza valores finales.
    3.7.- Exporta resultados formato .vtk cada cierto numero de
    iteraciones

FUTURAS APORTACIONES:
- Formulas costeras
```

```
- ...

-------------------------------------------------
Algoritmo de Chorin-Rannacher para resolucion de ecuaciones de
Navier-Stokes en 3D
Introducir etiquetado correcto, se puede copiar del .msh
No olvidar etiquetar el volumen
En contornos libres la codicion Dirichlet es el convect que le
corresponda, y q = 0
-------------------------------------------------*/

load "gmsh"
load "iovtk"
load "msh3"
load "MUMPS"
load "medit"

//verbosity = 11;


mesh3 Th = gmshload3("arinaga_peque.msh");
//plot(Th);

fespace VH(Th,P1);  // Espacios funcionales para modelo MMC
VH phi,ff,v,U0x,U0y,U0z;
VH ux,uy,uz,uu,error;

real a = 0.038; //coeficiente del perfil vertical de corrientes
// en Canarias
real eps = 1e-8; //para poner en componentes nulas y evitar
// futuros errores numericos

real T = 20;// temperatura superficie en oC para
// densidad de referencia
real S = 35; // salinidad en g/m3 media en Canarias

// Expresion UNESCO para la densidad en funcion
// de salinidad y temperatura


// Expresion UNESCO para la densidad en funcion de
// salinidad y temperatura
func real rho(real T, real s) {return
(999.842594+6.793952E-2*T-9.09529E-3*T^2+1.001685E-4*T^3
-1.120083E-6*T^4+6.536332E-9*T^5+(0.824493-4.0899E-3*T
+7.6438E-5*T^2-8.2467E-7*T^3+5.3875E-9*T^4)*s+(-5.72466E-3
+1.0277E-4*T-1.6546E-6*T^2)*s^(1.5)+4.8314E-4*s^2);
}

cout<<"\n Densidad UNESCO de referencia = "<<rho(T,S)<<endl;

//Datos Estaciones de medidas: corriente y viento
```

```
int Nest = 2;
real[int] estx(Nest);
estx[0] = 457589.1418581557; estx[1] = 456747.723961307;

real[int] esty(Nest);
esty[0] = 3074874.086884347; esty[1] = 3074490.658113498;

real[int] estz(Nest);
estz[0] = 25; estz[1] = 25;

real[int] estu(Nest);
estu[0] = -0.2; estu[1] = -0.2;  // NEW

real[int] estv(Nest);
estv[0] = -0.3; estv[1] = eps;    // NEW

real[int] estw(Nest);
estw[0] = 0; estz[1] = 0;

real[int] estAlpha(Nest); // rumbo de la corriente
// en cada estacion

/* -- Si el modulo esta en:
    -- primer cuadrante (ambos positivos) entonces se deja igual
    -- 2o cuadrante (+x,-y): sumar c2
    -- 3er cuadrante (-x,-y): sumar c3
    -- 4o cuadrante (-x,+y) o eje -y: sumar c4

-- NO PONER NINGUNA COMPONENTE 0, SINO EPSILON */

real c2 = pi, c3 = pi, c4 = 2*pi;

estAlpha[0] = c3 + atan(estu[0]/estv[0]);
estAlpha[1] = c4 + atan(estu[1]/estv[1]);

cout << "---------------------------------------- " << endl;
cout << " Rumbo de corriente en boya 1 = "
<< estAlpha[0]*180/pi << endl;
cout << " Rumbo de corriente en boya 2 = "
<< estAlpha[1]*180/pi << endl;
cout << "---------------------------------------- " << endl;

// Datos de viento

real vviento = 7 *  0.03; // Modulo vel viento en m/s
//proporcion de velocidad transferida a la capa superficial del mar
real rumboviento = 230 *pi/180; // Rumbo viento
//(0 es norte y posit. en sentido antihorario) y pasa a radianes

real vientox = vviento * sin(rumboviento);
real vientoy = vviento * cos(rumboviento);
```

```
//Interpolacion del campo inicial de velocidades
// "dist" es la distancia al cuadrado (dist^2)
//-----------------------------------------------
func real interpolax (int NEst, real[int] & Ex, real[int]
& Ey, real[int] & Ez, real[int] & Eu)
{
    real SMALL = 1.E-6;
    real dist;
    real sum = 0.;
    real uox = 0.;
    real distVert;
    for(int ii=0; ii<NEst; ++ii) {
        if(abs(Ex[ii]-x) > SMALL  || abs(Ey[ii]-y) > SMALL
        || abs(Ez[ii]-z) > SMALL)
        {
            dist = pow(Ex[ii]-x, 2.) + pow(Ey[ii]-y, 2.);
            // Distancia de cada nodo a la estacion

            if(dist == 0) {
            dist = eps; } //A veces alguna distancia es
            // cero aunque no deberia porque ya hay un condicional
            sum += 1/dist;
            uox += Eu[ii]/dist;

            distVert = z-Ez[ii]; //distancia vertical del nodo
            // a la estacion

            /* Componente */
            uox += exp(-a*distVert)*uox*cos((pi/2)-estAlpha[ii]
            -a*distVert);
            //modifica las componentes, = un giro del vector
        }
        else  /* ESTACION */
            return Eu[ii]*exp(-a*distVert);
    }
    if(sum > SMALL)
        return uox/sum;
    else
        return 0.;
}

func real interpolay (int NEst, real[int] & Ex, real[int] & Ey,
real[int] & Ez, real[int] & Ev)
{
    real SMALL = 1.E-6;
    real dist;
    real sum = 0.;
    real uoy = 0.;
    real distVert;
    for(int ii=0; ii<NEst; ++ii) {
        if(abs(Ex[ii]-x) > SMALL  || abs(Ey[ii]-y) > SMALL
```

```
        || abs(Ez[ii]-z) > SMALL)
        {
            dist = pow(Ex[ii]-x, 2.) + pow(Ey[ii]-y, 2.);
            if(dist == 0) {
            dist = eps; }
            sum += 1/dist;
            uoy += Ev[ii]/dist;

            distVert = z-Ez[ii]; //distancia vertical
            // del nodo a la estacion

            /* Componente */
            uoy += exp(-a*distVert)*uoy*sin((pi/2)
            -estAlpha[ii]-a*distVert);

        }
        else    /* ESTACION */
            return Ev[ii]*exp(-a*distVert);
    }
    if(sum > SMALL)
        return uoy/sum;
    else
        return 0.;
}

U0x = interpolax(Nest,estx,esty,estz,estu);
U0y = interpolay(Nest,estx,esty,estz,estv);


// ------------------------------------------------
// Comprobar la divergencia del campo de velocidades
// ------------------------------------------------

real vol = int3d(Th)(1.); // NEW
/* antes mas abajo aqui para hcer valores medios de la divergencia */

real DIVI,DIVF;
ff=dx(U0x)+dy(U0y)+dz(U0z);
DIVI=int3d(Th)(ff/vol);
cout << " Divergencia media campo velocidad inicial = "
<< DIVI << endl;

problem velocidadMMC (phi,v,solver=sparsesolver) =
int3d(Th)(dx(phi)*dx(v)+dy(phi)*dy(v)+dz(phi)*dz(v))
-int3d(Th)(ff*v)
+int2d(Th,1,5)((U0x*N.x+U0y*N.y+U0z*N.z)*v)
//Condicion Neumann sobre tierra
+on(2,3,4,6,phi=0); //superficies libres

velocidadMMC ;

ux=dx(phi)+U0x;
```

```
uy=dy(phi)+U0y;
uz=dz(phi)+U0z;

DIVF=int3d(Th)((dx(ux)+dy(uy)+dz(uz))/vol);
cout << " Divergencia media campo velocidad final = "
 << DIVF << endl;


int[int] ffordertt = [0];

// Parte del campo interpolado y busca un campo U que mejor
// se aproxime al interpolado.

    savevtk("Velocidad_MMC.vtk", Th, [ux,uy,uz],
    order = ffordertt, dataname = "U_MMC");
    cout << "\n **Generada grafica de velocidades
    del modelo MMC** " << endl;


// ---------------
// Inicia Chorin
// ---------------

// ETIQUETAS:
/*
2 1 "costa"
2 2 "este"
2 3 "sur"
2 4 "oeste"
2 5 "fondo"
2 6 "surf_mar"
3 7 "volumen"
*/

//Declaracion de espacios funcionales y condiciones inciales
fespace Vh(Th, P13d);
Vh u1 = ux, u2 = uy, u3 = uz, p = 0, q = 0, w, TBou = 15, TT;


real TIME;
real t = 0;
int iter = 0;
int nt = 800;
real dt = 0.01;


//real rho = 1025; // densidad
real T0 = 16;  // Temperatura de referencia en oC a la cual
// asigno densidad y conductividad
real nu = 0.001/rho(TBou,S); //visc. cinematica.
// Deberia estar en torno a 1.05e-6 (m2/s)
```

```
real gr = 9.81;// Gravedad (m/s)
real Cp = 4000;// Cp = 3991.86 (IOC et al. 2010) -- Los liquidos
// solo tienen Cp, calor esp. p=cte. [J/(Kg K)].
real K = 0.6;//0.596;// Conductividad termica [W/(m K)]

real kT = K / (rho(TBou,S) * Cp);
real beta = 0.003458; // 0.1655 (EOS-80), aprox del coeficiente
// de expansion termica NEW  no inversa de T(kelvin, gases)

// ------------------------------------------------------------

for(int i = 0; i < nt; ++i)
  {
    ++iter;
    t += dt;


Vh u1old = u1, u2old = u2, u3old = u3, pold = p, TBouold = TBou;
Vh f = convect([u1,u2,u3], -dt, u1old),
g1 = convect([u1,u2,u3], -dt, u2old),
g2 = convect([u1,u2,u3], -dt, u3old);

real epsi=0.00000001; //  NEW
real kT = K / (rho(TBou,S) * Cp);

solve pb4u(u1,w,init=i,solver=UMFPACK)
=int3d(Th)(u1*w/dt +nu*(dx(u1)*dx(w)+dy(u1)*dy(w)+dz(u1)*dz(w)))
-int3d(Th)((f/dt-dx(p))*w)
+int3d(Th)(epsi*p*w)  // NEW


// Entrada:
+ on(2, u1 = ux)             // Condicion de contorno en entrada
calculada por MMC
+ on(6, u1 = ux+vientox)     // Condicion de contorno en la
// superficie, calculada por MMC + el 3% del viento

// Salida:
+ on(4,3, u1 = f)

// Contornos donde la componente debe ser 0 (no flip condition):
+ on(1,5, u1 = 0);


solve pb4v(u2,w,init=i,solver=UMFPACK)
= int3d(Th)(u2*w/dt +nu*(dx(u2)*dx(w)+dy(u2)*dy(w)+dz(u2)*dz(w)))
-int3d(Th)((g1/dt-dy(p))*w)
+int3d(Th)(epsi*p*w)  // NEW

// Entrada:
+ on(2, u2 = uy)
+ on(6, u2 = uy+vientoy)
```

```
// Salida:
+ on(4,3, u2 = g1)

// Contornos donde la componente debe ser 0 (no flip condition):
+ on(1,6, u2 = 0);

solve pb4w(u3,w,init=i,solver=UMFPACK)
= int3d(Th)(u3*w/dt +nu*(dx(u3)*dx(w)+dy(u3)*dy(w)+dz(u3)*dz(w)))
-int3d(Th)((g2/dt-dz(p))*w)

-int3d(Th)(( beta*gr*TBouold + beta*gr*T0)*w)

// Entrada:
+ on(2, u3 = uz)

// Salida:
+ on(4,3, u3 = g2)

// Contornos donde la componente debe ser 0 (no flip condition):
+ on(1,6,5, u3 = 0);


// Resuelve temperatura
solve pb4T(TBou,TT,init=i,solver=UMFPACK) =

 -int3d(Th) ( (convect([u1, u2, u3], -dt, TBouold)) * TT/dt)
 +int3d(Th) ((dx(TBou)*dx(TT)+dy(TBou)*dy(TT)+dz(TBou)*dz(TT))
 *kT + TBou * TT/dt)

+on(5,TBou = 15) + on(6, TBou = 20); // Temperaturas en fondo
// y superficie respectivamente


real meandiv = int3d(Th)(dx(u1)+dy(u2)+dz(u3))/vol;

solve pb4p(q, w, init = i, solver = UMFPACK)
= int3d(Th)(dx(q)*dx(w) + dy(q)*dy(w) + dz(q)*dz(w))
- int3d(Th)((dx(u1) + dy(u2) + dz(u3) - meandiv) * w/dt)

+ on(4,3, q = 0); //salidas


real meanpq = int3d(Th)(pold - q)/vol;

p = pold - q - meanpq;
u1 = u1 + dx(q) * dt;
u2 = u2 + dy(q) * dt;
u3 = u3 + dz(q) * dt;

int[int] ffordert = [0];
```

```
TIME += dt;
    cout << " dt = " << dt << " TIME = "<<TIME<<endl;
    // Imprime en pantalla
    cout << " iteracion = " << iter << " TIME = "<<TIME<<endl;
    // Imprime en pantalla

     if ( !(iter %10))

    {
        savevtk("VeloChorinCosta-"+iter/10+".vtk", Th, [u1,u2,u3],
        order = ffordert, dataname = "VelocidadChorin");
        cout << "\n **Generada grafica .vtk de
        velocidades Chorin** " << endl;

        savevtk("TemperaturaCosta-"+iter/10+".vtk", Th, TBou,
        order = ffordert, dataname = "Temperatura");
        cout << "\n **Generada grafica .vtk de
        campo de temperaturas** " << endl;

    }

//plot(u1,u2,u3,nbiso=15,value=1);
//medit("solucion",Th,[u1,u2,u3],order=1);

}
```