Rev. Int. Mét. Num. Cálc. Dis. Ing. (2011) 27: 129-145

# PUMI: un código explícito no estructurado para resolver las ecuaciones de Euler

**R. Flores · E. Ortega · E. Oñate**

Recibido: Marzo 2008, Aceptado: Junio 2008


**Resumen** El código PUMI se ha desarrollado en CIMNE para resolver rápidamente problemas de flujo compresible alrededor de geometrías complejas. En la actualidad se llevan a cabo de manera rutinaria cálculos con mallas que contienen decenas de millones de elementos. PUMI fue diseñado específicamente para tratar problemas de esta escala usando hardware modesto; por tanto la eficiencia del código ha sido uno de los objetivos fundamentales durante su diseño. Se buscó una utilización mínima de memoria, excelente comportamiento en entornos de ejecución secuencial y escalado satisfactorio hasta un número moderado de procesos (para su uso eficiente en estaciones de trabajo multiprocesador). Para facilitar en lo posible las tareas de generación de mallas se eligió una formulación no estructurada basada en elementos finitos. El presente artículo describe los fundamentos teóricos del algoritmo así como detalles de implementación que mejoran la robustez y eficiencia del código.

## PUMI: AN EXPLICIT UNSTRUCTURED SOLVER FOR THE EULER EQUATIONS

**Summary** The PUMI flow solver has been developed at CIMNE in order to address the need for fast solutions of the flow field around complex geometries. Nowadays calculations involving a number of cells on the order of tens of millions are performed routinely. PUMI was created to deal with this kind of large-scale problem using modest hardware, therefore special emphasis was placed on the computational efficiency of the code. Design guidelines where minimum memory requirement, very fast single-threaded performance as well as satisfactory parallel scaling up to a moderate number of threads (as found on current entry-level SMP workstations) . In order to simplify the mesh generation activities an unstructured finite element formulation was selected. This paper describes the theoretical basis of the algorithm as well as details of the implementation that increase the robustness and efficiency of the code.

R. Flores, E. Ortega, E. Oñate
Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE)
C/ Gran Capitán s/n, Campus Nord UPC
08034 Barcelona España
Tel. 34 93 205 7076; Fax 34 93 401 6517
rflores@cimne.upc.edu; eortega@cimne.upc.edu; onate@cimne.upc.edu

## 1. Introducción

La incesante búsqueda por parte de la industria aeroespacial de mejoras en la economía, comodidad, seguridad y respeto por el medio ambiente ha impulsado el desarrollo de la mecánica de fluidos computacional. Los elevados costes y tiempos de preparación asociados con los ensayos en túnel de viento combinados con el enorme aumento de la capacidad de cálculo disponible han impulsado tremendamente el uso de la simulación numérica. A día de hoy configuraciones complejas en 3D (p. ej. aviones completos) se analizan de manera rutinaria. Se necesitan pues programas de cálculo eficientes que permitan obtener resultados detallados en un tiempo mínimo. El método más económico para obtener resultados realistas en régimen transónico (la principal área de interés de la industria aeronáutica) lo constituyen las ecuaciones de Euler (modelos más sencillos como los métodos de paneles no ofrecen resultados satisfactorios cuando los efectos de compresibilidad

se hacen importantes). Aunque en la actualidad existen métodos más precisos (RANS fundamentalmente, ya que LES es demasiado oneroso para su aplicación en tareas de producción) la solución no viscosa es a menudo adecuada para tareas de diseño preliminar. Además, esta solución puede mejorarse mediante el acoplamiento con un código de capa límite para capturar efectos viscosos manteniendo el coste computacional en niveles reducidos. Más aún, en el núcleo de cualquier código de NS se encuentra un solver de Euler y por tanto es una herramienta muy importante para desarrollos futuros. Puesto que CIMNE es un centro de investigación orientado al desarrollo de software y métodos numéricos no dispone de grandes recursos de cálculo. Por tanto, para cooperar efectivamente con la industria aeroespacial necesita un código de análisis adecuado para su ejecución en hardware modesto pero que proporciones soluciones en un tiempo razonable. PUMI se desarrolló para satisfacer esta necesidad. Además, para facilitar su integración con el resto de software de CIMNE, se optó por el método de los elementos finitos (actualmente CIMNE persigue la integración de sus códigos en un entorno de desarrollo común especialmente adaptado a los elementos finitos [1]). Para acelerar la ejecución y reducir los requisitos de memoria se eligió una estructura de almacenamiento por aristas. Como se verá mas adelante, estas decisiones conducen a una un algoritmo que presenta una notable similitud con una formulación de volúmenes finitos. Esto permite aplicar, sin necesidad de grandes cambios, numerosas técnicas desarrolladas por la comunidad de volúmenes finitos. Para permitir el uso eficiente del programa en estaciones de trabajo multi-procesador, el código ha sido paralelizado usando directivas de compilación Open-MP.

La primera sección del presente artículo describe las ecuaciones básicas, su formulación débil que permite la aplicación del método de los elementos finitos así como las particularidades debidas al almacenamiento por aristas. A continuación se presenta la implementación del algoritmo, incluyendo detalles sobre las técnicas de estabilización, aceleración de la convergencia y optimización del código. Para terminar, se muestra un caso práctico de aplicación a un problema de interés industrial para ilustrar las capacidades del software.

## 2. Ecuaciones básicas

Se parte de las ecuaciones de Euler formuladas en una referencia euleriana (fija en el espacio) y escritas en variables conservativas (para facilitar la obtención la ondas de choque posicionadas correctamente)

$$\frac{\partial \boldsymbol{\Phi}}{\partial t} + \frac{\partial \mathbf{F}_k}{\partial x_k} = 0 \quad \text{for} \quad k = 1..,3 \tag{1}$$

siendo $\boldsymbol{\Phi}$ el vector de variables conservativas y $\mathbf{F}_k$ el flujo convectivo en la k-ésima dirección

$$\boldsymbol{\Phi} = \begin{bmatrix} \rho \\ U_1 \\ U_2 \\ U_3 \\ e \end{bmatrix} \quad \mathbf{F}_i = \begin{bmatrix} U_i \\ u_i U_1 + p\delta_{i1} \\ u_i U_2 + p\delta_{i2} \\ u_i U_3 + p\delta_{i3} \\ u_i h \end{bmatrix} \tag{2}$$

donde $\boldsymbol{\delta}_{ij}$ is la delta de Kronecker. El vector de estado contiene la densidad, el momento ($U_i = \rho\, u_i$) y la energía total (interna y cinética) por unidad de volumen del fluido. Suponiendo que el fluido se comporta como un gas ideal, las expresiones para la energía total, entalpía y ecuación de estado son

$$\begin{aligned} &e = \rho\left(c_v T + \frac{u^2}{2}\right) \quad h = e + p = \rho\left(c_p T + \frac{u^2}{2}\right) \\ &p = \rho R T \quad R = c_p - c_v \end{aligned} \tag{3}$$

Para resolver las ecuaciones se necesitan condiciones de contorno apropiadas. En las paredes impermeables se impone la condición de deslizamiento (velocidad paralela a la superficie)

$$\mathbf{u}\,(\mathbf{x}) \cdot \mathbf{n} = 0 \;\; \forall \;\; \mathbf{x} \in \Sigma_S \tag{4}$$

donde $\sum_S$ representa a la frontera del sólido. También son precisas condiciones para el campo lejano, que varían según el flujo sea subsónico o supersónico

$$\begin{aligned} &\boldsymbol{\Phi} \to \boldsymbol{\Phi}_\infty \;\text{ for }\; \mathbf{x} \to \infty \;\text{ if }\; M_\infty < 1 \\ &\boldsymbol{\Phi} \to \boldsymbol{\Phi}_\infty \;\text{ for }\; \mathbf{x} \to \infty^- \;\text{ if }\; M_\infty \geq 1 \end{aligned} \tag{5}$$

El símbolo $\infty^-$ indica las condiciones del flujo sin perturbar (aguas arriba, lejos del sólido). Para mejorar el condicionamiento de las ecuaciones y obtener resultados más generales es recomendable escribir las ecuaciones en forma adimensional. Eligiendo las siguientes variables

$$\begin{aligned} &\tilde{t} = \frac{t\, c_\infty}{L} \quad \tilde{x}_i = \frac{x_i}{L} \quad \tilde{u}_i = \frac{u_i}{c_\infty} \quad \tilde{U}_i = \frac{u_i}{\rho_\infty c_\infty} \\ &\tilde{p} = \frac{p}{\rho_\infty c_\infty^2} \quad \tilde{e} = \frac{e}{\rho_\infty c_\infty^2} \quad \tilde{h} = \frac{h}{\rho_\infty c_\infty^2} \quad \tilde{T} = \frac{TR}{c_\infty^2} \end{aligned} \tag{6}$$

las ecuaciones se transforman en

$$\frac{\partial \tilde{\boldsymbol{\Phi}}}{\partial \tilde{t}} + \frac{\partial \tilde{\mathbf{F}}_k}{\partial \tilde{x}_k} = 0 \quad \text{for} \quad k = 1..,3 \tag{7}$$

En (6) L representa la longitud característica del problema, $\boldsymbol{\rho}_\infty$ y $\mathbf{a}_\infty$ la densidad y velocidad del sonido ($a^2 = \gamma R T$) del fluido sin perturbar (las condiciones

aguas arriba a gran distancia de las paredes). Los vectores adimensionales de incógnitas y flujos se convierten en

$$\tilde{\boldsymbol{\Phi}} = \begin{bmatrix} \tilde{\rho} \\ \tilde{U}_1 \\ \tilde{U}_2 \\ \tilde{U}_3 \\ \tilde{e} \end{bmatrix} \quad \tilde{\mathbf{F}}_i = \begin{bmatrix} \tilde{U}_i \\ \tilde{u}_i\tilde{U}_1 + \tilde{p}\delta_{i1} \\ \tilde{u}_i\tilde{U}_2 + \tilde{p}\delta_{i2} \\ \tilde{u}_i\tilde{U}_3 + \tilde{p}\delta_{i3} \\ \tilde{u}_i\tilde{h} \end{bmatrix} \tag{8}$$

Con este cambio las relaciones termodinámicas para el gas ideal resultan:

$$\begin{gathered} \tilde{e} = \tilde{\rho}\left(\frac{\tilde{T}}{\gamma-1} + \frac{\tilde{u}^2}{2}\right) \quad \tilde{h} = \tilde{\rho}\left(\frac{\gamma}{\gamma-1}\tilde{T} + \frac{\tilde{u}^2}{2}\right) \\ \tilde{p} = \tilde{\rho}\tilde{T} \quad \gamma = \frac{c_p}{c_v} \end{gathered} \tag{9}$$

Las condiciones para el campo lejano son ahora mucho más sencillas

$$\tilde{\boldsymbol{\Phi}}_\infty = \begin{bmatrix} 1 \\ M_\infty^X \\ M_\infty^Y \\ M_\infty^Z \\ \dfrac{1}{\gamma(\gamma-1)} + \dfrac{M_\infty^2}{2} \end{bmatrix} \tag{10}$$

Es fácil demostrar que tanto la presión como la temperatura adimensionales de la corriente no perturbada valen $\frac{1}{\gamma}$. Por tanto la solución adimensional es función únicamente del número de Mach, la relación de calores específicos y la forma (no el tamaño) del sólido. Un beneficio adicional es que los parámetros adimensionales son de orden unitario y mejoran el condicionamiento del sistema. De ahora en adelante, para mantener la claridad, se omitirá la tilde de las variables adimensionales. Se supondrá siempre que se está empleando la forma adimensional de las expresiones.

Para obtener la aproximación mediante elementos finitos del sistema (7) se parte de la forma débil de las ecuaciones diferenciales [2]. Sea $W$ una función de prueba genérica definida en el interior del dominio fluido. La forma débil del balance de flujos es:

$$\int_\Omega W(\mathbf{x}) \left(\frac{\partial \boldsymbol{\Phi}}{\partial t} + \frac{\partial \mathbf{F}_k}{\partial x_k}\right) d\Omega = 0 \ \forall \ W \tag{11}$$

En tanto (11) se cumpla para cualquier $W$, las dos formas son equivalentes. A continuación se usa el método de Galerkin para construir una solución aproximada $\tilde{\boldsymbol{\Phi}}$ que es una combinación lineal de los valores nodales y las funciones de interpolación $N_j$.

$$\begin{aligned} &\tilde{\boldsymbol{\Phi}}(\mathbf{x}) = N_j(\mathbf{x})\,\tilde{\boldsymbol{\Phi}}(\mathbf{x}^j) = N_j\,\tilde{\boldsymbol{\Phi}}^j \\ &W(\mathbf{x}) = N_i(\mathbf{x}) \end{aligned} \tag{12}$$

En (12) se supone que se suma sobre el índice repetido y que el superíndice indica valores nodales. Para el caso específico de los elementos finitos las funciones de interpolación son las funciones de forma que se definen de manera elemental. Cumplen la propiedad

$$N_i(\mathbf{x}^j) = \delta_{ij} \tag{13}$$

La forma semi-discreta de (11) es

$$\int_\Omega N_i \left(N_j\dot{\tilde{\boldsymbol{\Phi}}}^j + \frac{\partial \tilde{\mathbf{F}}_k}{\partial x_k}\right) d\Omega = 0 \ for \ \ i = 1...n_{node} \tag{14}$$

donde el punto sobre el vector de estado indica la derivada temporal. En (14) aparecen tantas ecuaciones como incógnitas (cinco por nodo) y por tanto el sistema puede resolverse para obtener los valores nodales de la solución aproximada (siempre y cuando se impongan correctamente las condiciones de contorno). Las integrales en (14) se calculan usando la cuadratura numérica de Gauss. La manera consistente de obtener los flujos en los puntos de integración sería:

$$\tilde{\mathbf{F}}_k^{IP} = \tilde{\mathbf{F}}_k\left(\tilde{\boldsymbol{\Phi}}^{IP}\right) = \tilde{\mathbf{F}}_k\left(N_j(\mathbf{x}^{IP})\tilde{\boldsymbol{\Phi}}^j\right) \tag{15}$$

Para mejorar la eficiencia del algoritmo se supondrá que pueden interpolarse los flujos en el interior de los elementos a partir de sus valores nodales [3]. Esto equivale a usar la cuadratura de Lobato para los flujos y no afecta sensiblemente al resultado final

$$\tilde{\mathbf{F}}_k \cong N_j\tilde{\mathbf{F}}_k(\mathbf{x}^j) = N_j\tilde{\mathbf{F}}_k^j \tag{16}$$

Con este cambio (14) se transforma en

$$\int_\Omega N_i \left(N_j\dot{\tilde{\boldsymbol{\Phi}}}^j + \frac{\partial N_j}{\partial x_k}\left(\tilde{\mathbf{F}}_k^j\right)\right) d\Omega = 0 \ for \ \ i = 1...n_{node} \tag{17}$$

Trasladando el término que contiene los flujos al lado derecho y empleando notación matricial la derivada temporal del vector de estado se puede escribir como

$$\begin{aligned} &\dot{\tilde{\Phi}}^j = \mathbf{M}^{-1}\,\mathbf{r} \\ &M = \int_\Omega N_i\,N_j\,d\Omega \\ &r = -\int_\Omega N_i\,\frac{\partial N_j}{\partial x_k}\,d\Omega\,\tilde{F}_k^j \end{aligned} \tag{18}$$

siendo $\mathbf{M}$ la matriz de masas consistente. Para evitar tener que resolver un sistema lineal de ecuaciones a cada paso la matriz de masas consistente suele reemplazarse por su variante diagonal definida de la siguiente manera

$$M_{ij}^d = \delta_{ij}\sum_j M_{ij} \tag{19}$$

Con este cambio el proceso de inversión resulta trivial. Para mejorar la eficiencia del algoritmo es preciso manipular el vector de residuos ($\mathbf{r}$). En primer lugar se divide el residuo de la i-ésima ecuación en dos partes:

$$r^i = -\sum_{j\neq i}\int_\Omega N_i\, N_{j,k}\, d\Omega\, \tilde{F}^j_k - \int_\Omega N_i\, N_{i,k}\, d\Omega\, \tilde{F}^i_k \qquad (20)$$

where $N_{j,k} = \dfrac{\partial N_j}{\partial x_k}$.

En la expresión anterior no existe sume sobre el índice $i$, y la suma sobre $j$ se extiende a todos los valores posibles salvo $i$. De ahora en adelante, para obtener expresiones más compactas, se omitirá el signo de suma pero esta salvedad se mantiene (esto es, $i$ no entra en la suma sobre $j$). Integrando (20) por partes se obtiene

$$\begin{aligned} r^i = &\int_\Omega N_{i,k}\, N_j\, d\Omega\, \tilde{F}^{ij}_k - \int_\Omega N_{i,k}\, N_j\, d\Omega\, \tilde{F}^i_k \\ &- \int_\Gamma N_i\, N_j\, n_k d\Gamma\, \tilde{F}^j_k - \frac{1}{2}\int_\Gamma N_i\, N_i\, n_k d\Gamma\, \tilde{F}^i_k \end{aligned} \qquad (21)$$

donde se ha definido el flujo numérico en la entrefase como

$$\tilde{\mathbf{F}}^{ij}_k = \tilde{\mathbf{F}}^i_k + \tilde{\mathbf{F}}^j_k \qquad (22)$$

Nótese que (22) es el doble de la media de los valores nodales. La expresión (21) debe simetrizarse para obtener el máximo beneficio de la estructura de datos por aristas

$$\begin{aligned} \mathbf{r}^i = &\frac{1}{2}\int_\Omega \left(N_{i,k}\, N_j - N_i\, N_{j,k}\right)\, d\Omega\, \tilde{\mathbf{F}}^{ij}_k \\ &+ \int_\Gamma N_i\, N_j\, n_k d\Gamma\, \tilde{\mathbf{F}}^{ij}_k - \int_\Omega N_{i,k}\, N_j\, d\Omega\, \tilde{\mathbf{F}}^i_k \\ &- \int_\Gamma N_i\, N_j\, n_k d\Gamma\, \tilde{\mathbf{F}}^j_k - \frac{1}{2}\int_\Gamma N_i\, N_i\, n_k d\Gamma\, \tilde{\mathbf{F}}^i_k \end{aligned} \qquad (23)$$

Empleando la propiedad de las funciones de forma $N_i = 1 - \sum_{j\neq i} N_j$ puede llegarse a la siguiente expresión para el residuo

$$\begin{aligned} \mathbf{r}^i &= d^{ij}_k \tilde{\mathbf{F}}^{ij}_k + b^{ij}_k \tilde{\mathbf{F}}^{ij}_k + c^i_k \tilde{\mathbf{F}}^i_k \\ d^{ij}_k &= \frac{1}{2}\int_\Omega \left(N_{i,k}\, N_j - N_i\, N_{j,k}\right)\, d\Omega \\ b^{ij}_k &= -\frac{1}{2}\int_\Gamma N_i\, N_j\, n_k d\Gamma \\ c^i_k &= -\int_\Gamma N_i\, N_i\, n_k d\Gamma \end{aligned} \qquad (24)$$

Las funciones de forma del nodo $i$ son nulas en cualquier elemento que no contenga dicho nodo; por tanto las integrales en (24) sólo deben ser calculadas para pares de nodos $ij$ que compartan algún elemento. A estas parejas se las denomina aristas computacionales. Cuando el elemento es un símplice (un triángulo en 2D o un tetraedro en 3D) las aristas computacionales coinciden con las geométricas; en otro caso aparecerán aristas internas adicionales. Nótese que los coeficientes $d$ son antisimétricos, esto significa que sólo es preciso almacenar la mitad del total. Más aún, los términos $b$ y $c$ son nulos para las aristas internas reduciendo en gran medida el espacio de almacenamiento preciso [4].

El hecho de que los coeficientes $d$ sean antisimétricos implica que el esquema (18) es conservativo. Esto es, la contribución neta del residuo es nula para cualquier arista interna (que no contenga nodos en la superficie)

$$\mathbf{r}^i_e + \mathbf{r}^j_e = d^{ij}_k \tilde{\mathbf{F}}^{ij}_k + d^{ji}_k \tilde{\mathbf{F}}^{ji}_k = d^{ij}_k \tilde{\mathbf{F}}^{ij}_k - d^{ij}_k \tilde{\mathbf{F}}^{ij}_k = 0 \qquad (25)$$

El empleo de la expresión (24) para el residuo proporciona importantes ahorros tanto en el número de operaciones como en términos de acceso a la memoria [5]. Cabe mencionar que para aristas interiores el término $\boldsymbol{d}$ que aparece en la expresión es muy similar al que se obtendría en el caso de un esquema de volúmenes finitos centrado en las celdas si los baricentros de éstas últimas se encontrasen en los nodos de la mala de elementos finitos. Por tanto, la mayoría de las técnicas desarrolladas para su uso en códigos de volúmenes finitos no estructurados pueden aplicarse también a la formulación de elementos finitos basada en aristas.

## 3. Estabilización convectiva

La formulación básica de Galerkin, al ser equivalente a un esquema de diferencias finitas de segundo orden adolece de las mismas limitaciones que éste. En particular, es susceptible a la aparición de oscilaciones espurias que contaminan la solución [6]. Para evitar la aparición de soluciones no físicas debe añadirse algún mecanismo de estabilización al esquema de Galerkin original. Puesto que las ecuaciones de Euler constituyen un sistema hiperbólico, representan la propagación de ondas. Por tanto las técnicas de *upwinding* desarrolladas para la ecuación de ondas se pueden extender exitosamente a las ecuaciones de Euler. Esto se logra efectuando un cambio de variables que desacople el sistema (7). Los jacobianos del flujo ($\mathbf{A}_i$) se definen como

$$\frac{\partial \mathbf{F}_i}{\partial x_i} = \mathbf{A}_i \frac{\partial \boldsymbol{\Phi}}{\partial x_i} \qquad (26)$$

Los jacobianos representan las derivadas de los vectores de flujo respecto a las variables de estado. Se da la circunstancia de que los flujos son funciones homogéneas de las variables conservativas, por tanto también es posible escribir

$$\mathbf{F}_i = \mathbf{A}_i \boldsymbol{\Phi} \qquad (27)$$

Sustituyendo (26) en (7) se obtiene la forma casi-lineal de las ecuaciones de Euler

$$\frac{\partial \boldsymbol{\Phi}}{\partial t} + \mathbf{A}_k \frac{\partial \boldsymbol{\Phi}}{\partial x_k} = 0 \tag{28}$$

Nótese que el sistema (28) no es realmente lineal ya que los jacobianos son funciones de $\boldsymbol{\Phi}$. Sin embargo, si se consideran pequeñas fluctuaciones alrededor de un estado de equilibrio, es aceptable linealizar el comportamiento y suponer que los jacobianos son constantes. En aras de simplicidad, la discusión que sigue supondrá un flujo unidimensional según la dirección $x$, los resultados son válidos para cualquier dirección arbitraria. Para el caso 1D (28) se reduce a

$$\frac{\partial \boldsymbol{\Phi}}{\partial t} + \mathbf{A} \frac{\partial \boldsymbol{\Phi}}{\partial x} = 0 \tag{29}$$

Se ha descartado el subíndice ya que sólo existe una dirección espacial. Que el sistema de ecuaciones sea hiperbólico implica que los autovalores del jacobiano son siempre reales. Puede construirse un juego completo de autovectores [9]

$$\mathbf{A}_{il}\mathbf{R}_{lj} = \lambda_j \mathbf{R}_{ij} \tag{30}$$

tales que el jacobiano se puede escribir como

$$\mathbf{A} = \mathbf{R}\,\boldsymbol{\Lambda}\,\mathbf{R}^{-1} \quad \text{donde} \quad \boldsymbol{\Lambda}_{ij} = \delta_{ij}\lambda_i \tag{31}$$

La matriz $\boldsymbol{\Lambda}$ es diagonal y contiene los autovalores del jacobiano. Multiplicando (29) por $\mathbf{R}^{-1}$ se obtiene

$$\mathbf{R}^{-1}\frac{\partial \boldsymbol{\Phi}}{\partial t} + \mathbf{R}^{-1}A(\mathbf{R}\mathbf{R}^{-1})\frac{\partial \boldsymbol{\Phi}}{\partial x} = 0 \tag{32}$$

Utilizando la descomposición en autovectores de $\mathbf{A}$ y el cambio de variable $\boldsymbol{\varphi} = \mathbf{R}^{-1}\boldsymbol{\Phi}$ el sistema se convierte en

$$\frac{\partial \boldsymbol{\varphi}}{\partial t} + \boldsymbol{\Lambda}\frac{\partial \boldsymbol{\varphi}}{\partial x} = 0 \tag{33}$$

Puesto que la matriz $\boldsymbol{\Lambda}$ es diagonal, todas las ecuaciones en (33) quedan desacopladas. La evolución de cada una de las componentes de $\boldsymbol{\varphi}$ (denominadas variables características) queda descrita por la ecuación de ondas. Las velocidades de onda son los autovalores del jacobiano. Existen tres autovalores diferentes:

$$\begin{aligned} \lambda_1 &= u + c \\ \lambda_2 &= u - c \\ \lambda_3 &= u \end{aligned} \tag{34}$$

Los dos primeros autovalores representan perturbaciones acústicas y el tercero ondas de entropía. En el caso multidimensional el tercer autovalor es múltiple debido a la existencia de ondas de vorticidad que viajan a la misma velocidad que el fluido.

Una técnica popular para estabilizar el comportamiento numérico de la ecuación de ondas es el *upwinding*. Al utilizar diferencias hacia atrás en lugar de centradas para el término convectivo se resuelve el problema de las oscilaciones espurias [7]. Este remedio puede aplicarse a las variables características en (33), sin embargo, debe recordarse que el sentido de diferenciación correcto no es el mismo para todas las componentes de la solución. En efecto, salvo en caso de flujo supersónico ($u>c$) no todos los autovalores en (34) tienen el mismo signo (es decir, no todas las ondas viajan en el mismo sentido). Por tanto, el sentido de diferenciación debe elegirse componente por componente. En el contexto de un código de elementos finitos por aristas el *upwinding* se consigue sustituyendo el flujo en la entrefase por su valor aguas arriba de la arista. Si se define un flujo característico en (33) como $\mathbf{f}_k = \lambda_k \boldsymbol{\varphi}_k$ (donde el subíndice k denota una componente característica concreta) puede reescribirse el sistema de la siguiente forma

$$\frac{\partial \boldsymbol{\varphi}_k}{\partial t} + \frac{\partial \mathbf{f}_k}{\partial x} = 0 \tag{35}$$

que tiene la misma forma que (7) y permite pues el uso de la discretización (24). Para una arista interna se tiene

$$\mathbf{r}_k^i = d^{ij}\mathbf{f}_k^{ij} = d^{ij}(\mathbf{f}_k^i + \mathbf{f}_k^j) \tag{36}$$

Supóngase que el eje apunta desde en nodo $i$ hacia el nodo $j$. Dependiendo del signo de la velocidad de propagación la aproximación aguas arriba de (36) es

$$\begin{aligned} \mathbf{r}_k^i\big|_{Upwind} &= d^{ij}(2\mathbf{f}_k^i) = d^{ij}\left(\mathbf{f}_k^{ij} - (\mathbf{f}_k^j - \mathbf{f}_k^i)\right) \quad \text{if } \lambda_k > 0 \\ \mathbf{r}_k^i\big|_{Upwind} &= d^{ij}(2\mathbf{f}_k^j) = d^{ij}\left(\mathbf{f}_k^{ij} + (\mathbf{f}_k^j - \mathbf{f}_k^i)\right) \quad \text{if } \lambda_k < 0 \end{aligned} \tag{37}$$

que puede reescribirse como

$$\begin{aligned} \mathbf{r}_k^i\big|_{Upwind} &= d^{ij}\left(\mathbf{f}_k^{ij} - \lambda_k(\boldsymbol{\varphi}_k^j - \boldsymbol{\varphi}_k^i)\right) \quad \text{if } \lambda_k > 0 \\ \mathbf{r}_k^i\big|_{Upwind} &= d^{ij}\left(\mathbf{f}_k^{ij} + \lambda_k(\boldsymbol{\varphi}_k^j - \boldsymbol{\varphi}_k^i)\right) \quad \text{if } \lambda_k < 0 \end{aligned} \tag{38}$$

Las dos expresiones en (38) se pueden combinar [6] en

$$\mathbf{r}_k^i\big|_{Upwind} = d^{ij}\left(\mathbf{f}_k^{ij} - |\lambda_k|\,(\boldsymbol{\varphi}_k^j - \boldsymbol{\varphi}_k^i)\right) \tag{39}$$

Definiendo la matriz $|\boldsymbol{\Lambda}|$ como aquella que contiene en su diagonal el valor absoluto de los autovalores, la aproximación aguas arriba de (35) se convierte en

$$\mathbf{r}^i\big|_{Upwind} = d^{ij}\left(\mathbf{f}^{ij} - |\boldsymbol{\Lambda}|\,(\boldsymbol{\varphi}^j - \boldsymbol{\varphi}^i)\right) \tag{40}$$

donde el subíndice $k$ se ha eliminado ya que (40) es aplicable al vector de estado completo (todas las variables

características). Puede deshacerse ahora el cambio de variables conservativas a características para obtener

$$\mathbf{r}^i\big|_{Upwind} = d^{ij}\left(\mathbf{F}^{ij} - \mathbf{R}\,|\boldsymbol{\Lambda}|\,\mathbf{R}^{-1}(\boldsymbol{\Phi}^j - \boldsymbol{\Phi}^i)\right) \tag{41}$$

La matriz $\mathbf{R}\,|\boldsymbol{\Lambda}|\,\mathbf{R}^{-1} = |\mathbf{A}|$ se llama jacobiano positivo. Se obtiene haciendo positivos todos los autovalores del jacobiano de flujo.

En este punto es importante recordar que en la discusión precedente se supuso que el comportamiento había sido linealizado, permitiendo así utilizar un jacobiano constante. En un caso real, sin embargo, la obtención de una matriz $\mathbf{A}$ tal que $\mathbf{F}^j - \mathbf{F}^i = \mathbf{A}(\boldsymbol{\Phi}^j - \boldsymbol{\Phi}^i)$ siendo los estados $i$ y $j$ sensiblemente diferentes precisa resolver un problema de Riemann. Aunque es factible llevar a cabo esta tarea, resulta extremadamente onerosa desde el punto de vista computacional. Como alternativa el código PUMI utiliza el solver aproximado de Roe [10]

$$\mathbf{F}^j - \mathbf{F}^i \approx \tilde{\mathbf{A}}_{Roe}(\boldsymbol{\Phi}^j - \boldsymbol{\Phi}^i) \quad \text{donde} \quad \tilde{\mathbf{A}}_{Roe} = \mathbf{A}(\tilde{\boldsymbol{\Phi}}^{ij}_{Roe}) \tag{42}$$

siendo $\tilde{\boldsymbol{\Phi}}^{ij}_{Roe}$ el estado intermedio aproximado de Roe, que puede calcularse fácilmente a partir de los estados $i$ y $j$. Así pues, (41) se sustituye por

$$\mathbf{r}^i\big|_{Upwind} = d^{ij}\left(\mathbf{F}^{ij} - \left|\tilde{\mathbf{A}}_{Roe}\right|(\boldsymbol{\Phi}^j - \boldsymbol{\Phi}^i)\right) \tag{43}$$

Es posible calcular el jacobiano positivo en (43) usando los autovalores (34) y el correspondiente juego de autovectores (para el cual se dispone de una expresión explícita [10]). Desgraciadamente, el coste de esta operación es muy alto. Una alternativa menos onerosa es calcular directamente el producto $\left|\tilde{\mathbf{A}}_{Roe}\right|(\boldsymbol{\Phi}^j - \boldsymbol{\Phi}^i)$ sin evaluar $\left|\tilde{\mathbf{A}}_{Roe}\right|$. En las referencias [11] y [12] pueden encontrarse algoritmos muy eficientes a tal efecto.

Cabe destacar que en los puntos de remanso o sónicos algún autovalor del jacobiano se anula [13]. Esto origina una pérdida de estabilización para alguna componente característica de la solución, con el potencial para la aparición de oscilaciones espurias. En el caso de PUMI este problema se trata imponiendo un límite inferior al valor absoluto de los autovalores según la expresión

$$|\boldsymbol{\Lambda}|_{ii} = \text{máx}(|\lambda_i|\,,\,\alpha\,c) \tag{44}$$

El parámetro $\boldsymbol{\alpha}$ en (44) puede ser ajustado por el usuario. El valor $\boldsymbol{\alpha} = 0,2$ es normalmente suficiente para obtener buenos resultados. Hasta este momento la discusión se ha centrado en la versión unidimensional de las ecuaciones de Euler. Puesto que PUMI resuelve las ecuaciones en 3D es preciso generalizar el resultado. El jacobiano de flujo para el transporte a lo largo de una dirección arbitraria (definida por el vector unitario $\mathbf{n}$) puede obtenerse como combinación lineal de los jacobianos correspondientes a las tres direcciones del sistema de referencia

$$\mathbf{A}^n = \sum_k \mathbf{n}_k \mathbf{A}_k \tag{45}$$

Puesto que las ecuaciones son hiperbólicas los autovalores de (45) son reales (independientemente de la elección de $\mathbf{n}$) y siempre es posible por tanto obtener el jacobiano positivo. Para estabilizar las ecuaciones tridimensionales debe elegirse una direccción de transporte para cada arista de manera que el jacobiano pueda obtenerse a partir de (45). En PUMI se ha elegido la dirección de la propia arista. El vector unitario correspondiente a la arista que conecta los nodos $i$ y $j$ es

$$\mathbf{n}^{ij} = \frac{\mathbf{x}^j - \mathbf{x}^i}{\|\mathbf{x}^j - \mathbf{x}^i\|} \tag{46}$$

Una vez definida la dirección, la contribución al vector de residuo incluyendo el término de estabilización resulta ser

$$\mathbf{r}^i\big|_{Upwind} = d^{ij}_k \mathbf{F}^{ij}_k - d^{ij}_k \mathbf{n}^{ij}_k \left|\tilde{\mathbf{A}}^{n^{ij}}_{Roe}\right| \Delta_{ij}\boldsymbol{\Phi} \tag{47}$$

donde se ha introducido la diferencia en la arista $\Delta_{ij}\boldsymbol{\Phi} = \boldsymbol{\Phi}^j - \boldsymbol{\Phi}^i$. Obviamente esta no es la única posibilidad, otras opciones resultan igualmente válidas. Una alternativa bastante extendida consiste en hacer el vector $\mathbf{n}$ paralelo al coeficiente $d^{ij}_k$ de manera que el producto $d^{ij}_k \mathbf{n}^{ij}_k$ sea máximo [3].

La discretización (47) es estable, pero puesto que ya no está centrada su orden de aproximación es menor que en su forma original (24) (primer orden en vez de segundo) [9]. Para desarrollar un código eficiente es necesario alcanzar al menos segundo orden de aproximación en el espacio sobre la mayor parte del dominio fluido. En PUMI esto se logra limitando el término estabilizador en (47) de manera que tienda a desaparecer en aquellas regiones en las que la solución se comporta suavemente. Para obtener expresiones más compactas se define el flujo artificial en (47) como

$$\mathbf{Fa}^{ij}_k = -\mathbf{n}^{ij}_k \left|\tilde{\mathbf{A}}^{\mathbf{n}^{ij}}_{Roe}\right| \Delta_{ij}\boldsymbol{\Phi} \tag{48}$$

Para acercarse en lo posible a la aproximación de segundo orden es preciso disminuir el flujo artificial (48) reduciendo el término de la diferencia en la arista

$$\mathbf{Fa}^{ij}_k\Big|_{\text{lím}\,ited} = -\mathbf{n}^{ij}_k \left|\tilde{\mathbf{A}}^{n^{ij}}_{Roe}\right| \left(\boldsymbol{\Phi}^{j-1/2} - \boldsymbol{\Phi}^{i+1/2}\right) \tag{49}$$

En (49) $\boldsymbol{\Phi}^{j-1/2}$ and $\boldsymbol{\Phi}^{i+1/2}$ representan dos aproximaciones de alto orden a la solución en la entrefase. Es

de esperar que el flujo limitado (49) sea menor que (48) reduciendo así el nivel de difusión artificial. Cuando se trabaja con mallas de espaciado uniforme resulta relativamente sencillo obtener las aproximaciones de alto orden en la entrefase [7]. Por el contrario, en el caso de una malla no estructurada típica del método de los elementos finitos, el proceso de reconstrucción de los valores en la entrefase resulta algo más complejo [8]. Para aprovechar las técnicas desarrolladas para mallas estructuradas se definen dos estados extrapolados para cada arista. Corresponden a dos puntos virtuales *i-1* y *j+1* tales que

$$\left.\begin{array}{l}\mathbf{x}^{i-1}=\mathbf{x}^{i}-\mathbf{u}^{ij}\\ \mathbf{x}^{j+1}=\mathbf{x}^{j}+\mathbf{u}^{ij}\end{array}\right\}\quad \text{donde}\quad \mathbf{u}^{ij}=\mathbf{x}^{j}-\mathbf{x}^{i} \tag{50}$$

Estos puntos no tienen porqué coincidir con nodos de la malla. De hecho, pueden encontrarse fuera del dominio fluido si la arista se encuentra cerca de la frontera. Existen múltiples formas de estimar el valor de la solución en los puntos virtuales [8]:

- Usando la aproximación usual de elementos finitos para el elemento en el que se encuentren
- Usando el valor del nodo más próximo
- Usando los valores nodales del gradiente de la solución

En el caso de PUMI se ha elegido la tercera opción. Cuando se resuelven las ecuaciones de Navier-Stokes (que el código también puede tratar) los gradientes nodales se utilizan para calcular los flujos difusivos. Por tanto dicha información se encuentra ya disponible. De esta manera también se simplifica el tratamiento de aquellos puntos virtuales que yacen fuera del dominio fluido. El proceso de recuperación de las derivadas se describe en la sección de detalles de implementación. A partir de la aproximación central de segundo orden de la derivada los estados extrapolados hacia delante y atrás se calculan como

$$\begin{array}{l}\boldsymbol{\Phi}^{j+1}\simeq\boldsymbol{\Phi}^{i}+2\mathbf{u}^{ij}\cdot\nabla\boldsymbol{\Phi}^{j}\\ \boldsymbol{\Phi}^{i-1}\simeq\boldsymbol{\Phi}^{j}-2\mathbf{u}^{ij}\cdot\nabla\boldsymbol{\Phi}^{i}\end{array} \tag{51}$$

A partir de estos valores se pueden definir dos nuevas diferencias (hacia atrás y hacia adelante)

$$\Delta_i^-=\boldsymbol{\Phi}^{i}-\boldsymbol{\Phi}^{i-1}\quad \Delta_j^+=\boldsymbol{\Phi}^{j+1}-\boldsymbol{\Phi}^{j} \tag{52}$$

Las aproximaciones de alto orden en la entrefase se obtienen mediante un esquema de reconstrucción de orden variable

$$\begin{array}{l}\boldsymbol{\Phi}^{i+1/2}=\boldsymbol{\Phi}^{i}+\dfrac{1}{4}\left[(1-k)\Delta_i^-+(1+k)\Delta_{ij}\right]\\ \boldsymbol{\Phi}^{j-1/2}=\boldsymbol{\Phi}^{j}-\dfrac{1}{4}\left[(1-k)\Delta_j^+ +(1+k)\Delta_{ij}\right]\end{array} \tag{53}$$

El parámetro $k$ en (53) controla el orden de extrapolación [3], algunos valores típicos son:

- $k$ = -1 esquema de segundo orden hacia atrás
- $k = 0$ esquema de From
- $k = 1/3$ esquema de tercer orden hacia atrás
- $k = 1$ esquema de diferencias centrales de tres puntos

Mediante e uso del flujo artificial de alto orden (49) la precisión del esquema mejora. Sin embargo, la aproximación de alto orden es susceptible a oscilaciones no físicas en las cercanías de choques o gradientes bruscos. Para superar esta limitación el esquema debería revertir a (48) (esto es, $\boldsymbol{\Phi}^{i+1/2}\rightarrow\boldsymbol{\Phi}^{i}$ and $\boldsymbol{\Phi}^{j-1/2}\rightarrow\boldsymbol{\Phi}^{j}$) siempre que aparezcan discontinuidades en la solución (en aquellas áreas donde la solución no es suave sólo el esquema de primer orden se encuentra libre de oscilaciones, tal como demuestra el teorema de Godunov's [7]). Los cambios bruscos en la solución se detectan en PUMI midiendo la curvatura local de la solución. En cada nodo se calcula un limitador basado en las diferencias hacia atrás y adelante. El limitador debería idealmente aproximarse a la unidad cuando las dos diferencias sean similares y anularse si resultan muy diferentes

$$\mathbf{l}^{i}=f(\Delta_i^-,\Delta_{ij})\quad \mathbf{l}^{j}=f(\Delta_{ij},\Delta_j^+) \tag{54}$$

Nótese que los limitadores son vectores, ya que sus valores pueden ser diferentes para cada una de las componentes del vector de estado. Consúltese la siguiente sección para más detalles sobre los limitadores disponibles en PUMI. Las expresiones en (53) se modifican para controlar el nivel de extrapolación mediante:

$$\begin{array}{l}\boldsymbol{\Phi}^{i+1/2}=\boldsymbol{\Phi}^{i}+\frac{l^i}{4}\left[(\mathbf{1}-k\mathbf{l}^{i})\Delta_i^-+(\mathbf{1}+k\mathbf{l}^{i})\Delta_{ij}\right]\\ \boldsymbol{\Phi}^{j-1/2}=\boldsymbol{\Phi}^{j}-\frac{l^j}{4}\left[(\mathbf{1}-k\mathbf{l}^{j})\Delta_j^+ +(\mathbf{1}+k\mathbf{l}^{j})\Delta_{ij}\right]\end{array} \tag{55}$$

Es fácil comprobar que las aproximaciones de alto orden se aproximan a $\boldsymbol{\Phi}^{i}$ y $\boldsymbol{\Phi}^{-1/2}$ cuando los limitadores tienden a cero, de esta manera se favorece la obtención de una solución monótona.

## 4. Integración temporal

El código PUMI se desarrolló para tratar principalmente problemas estacionarios. Por tanto, el esquema de integración temporal fue elegido en base a consideraciones de robustez y rapidez de convergencia, considerándose la precisión temporal de importancia secundaria. Se optó por un esquema explícito multipaso de Runge-Kutta a fin de aumentar el paso de tiempo admisible y conseguir una solución robusta [9]. Para avanzar (18) en cada paso temporal se crean estados intermedios

$\boldsymbol{\Psi}_j$ tales que:

$$\begin{array}{l} \boldsymbol{\Psi}_0 = \tilde{\boldsymbol{\Phi}}\Big|_{t_i} \\ \boldsymbol{\Psi}_j = \boldsymbol{\Psi}_0 + \theta_j \left(t_{i+1} - t_i\right) \mathbf{M}^{-1}\mathbf{r}(\boldsymbol{\Psi}_{j-1}) \\ \tilde{\boldsymbol{\Phi}}\Big|_{t_{i+1}} = \boldsymbol{\Psi}_n \\ \theta_n = 1 \end{array} \tag{56}$$

siendo $n$ el número de pasos del esquema. Diferentes esquemas se caracterizan por la elección de $n$ y $\theta_i$. Dos opciones populares y que proporcionan un nivel razonable de robustez son:

- Esquema de 3 pasos con $\theta_1 = \frac{3}{5}$ , $\theta_2 = \frac{3}{5}$ , $\theta_3 = 1$
- Esquema de 4 pasos con $\theta_1 = \frac{1}{4}$ , $\theta_2 = \frac{1}{3}$ , $\theta_3 = \frac{1}{2}$ , $\theta_4 = 1$

En la mayoría de las ocasiones el esquema de 4 pasos logra un mejor compromiso entre el paso de tiempo admisible y el coste computacional por paso. A la hora de calcular el vector de residuos en (56) no es necesario actualizar los flujos artificiales en cada paso intermedio. Hacerlo incrementaría sustancialmente el coste de cada paso y sin embargo proporcionaría únicamente un aumento marginal de robustez. Por tanto los términos de estabilización se calculan normalmente sólo en el primer paso. El esquema (56) es explícito y por tanto condicionalmente estable, existe pues un límite superior para el paso de tiempo admisible. Para un problema puramente convectivo el límite de estabilidad elemental puede calcularse como:

$$\Delta t_{el} = C\frac{h}{\lambda_{\text{máx}}} \tag{57}$$

donde $C$ denota el número de Courant admisible, $h$ es una medida del tamaño del elemento y $\lambda_{\text{máx}}$ es el máximo autovalor del jacobiano. Si es preciso analizar la respuesta transitoria del sistema, debe elegirse un paso de tiempo global igual al mínimo de todos los elementos. Por el contrario, cuando se busca únicamente la solución estacionaria puede emplearse un paso de tiempo local [15]. De esta manera la solución se propaga con mucha mayor rapidez, obteniéndose la convergencia en un menor número de pasos. El paso de tiempo local se calcula de forma nodal. A cada nodo de la malla se le asocia un tamaño dado por

$$h^i_{nod} = \text{mín}(h^j_{el}) \ / \ i \in el^j \tag{58}$$

esto es, al nodo $i$ se le asigna la mínima de las dimensiones de los elementos que contienen el nodo $i$. Como tamaño característico de un elemento se toma la mínima de sus alturas. Por ejemplo, para un elemento tetraédrico:

$$h^i_{el} = \frac{3\Omega^i}{\text{máx}(A^i_j)} \qquad j = 1, ..., 4 \tag{59}$$

siendo $\Omega^i$ el volumen del elemento y $A^i_j$ las áreas de sus caras. Esta elección del tamaño elemental aumenta la robustez del algoritmo cuando la malla contiene elementos severamente distorsionados. El incremento de tiempo admisible en cada nodo se obtiene como

$$\Delta t^i_{nod} = C\frac{h^i_{nod}}{\|\mathbf{v}^i\| + c^i} \tag{60}$$

El valor de $C$ en (60) es, en principio, global (el mismo para todos los nodos de la malla). Sin embargo, en áreas de baja velocidad, a causa del aumento de la rigidez de las ecuaciones, puede ser preciso reducir el valor de $C$ a fin de evitar oscilaciones. Para no degradar la convergencia global el usuario puede especificar dos valores diferentes del número de Courant. El valor inferior se usa en áreas en las que el número de Mach local se encuentre por debajo de un umbral ajustable. A fin de aumentar el paso de tiempo admisible PUMI emplea la técnica de suavizado implícito del residuo. Mediante la aplicación de un operador laplaciano se extiende el soporte de las funciones de forma, obteniendo así un aumento del número de Courant admisible. El valor suavizado del residuo viene dado por

$$\bar{\mathbf{r}}^i = \mathbf{r}^i + \varepsilon \sum_j \left(\bar{\mathbf{r}}^j - \bar{\mathbf{r}}^i\right) \tag{61}$$

para todos los nodos $j$ conectados a $i$.

Obtener el valor exacto de $\bar{\mathbf{r}}^i$ requeriría resolver un sistema lineal de ecuaciones, negando así las ventajas del esquema explícito. Debe recordarse que el suavizado tiene como misión única acelerar la convergencia sin afectar la solución estacionaria. Por tanto, sólo es necesaria una aproximación de $\bar{\mathbf{r}}^i$ para obtener el resultado deseado. Este valor aproximado se calcula usando un esquema iterativo de Jacobi:

$$\bar{\mathbf{r}}^i_n = \frac{\mathbf{r}^i + \varepsilon \sum_j \bar{\mathbf{r}}^j_{n-1}}{1 + \varepsilon \sum_j 1} \tag{62}$$

En la mayoría de las aplicaciones se obtiene un buen resultado con dos pasadas del esquema (62) y ajustando el coeficiente de difusión $\varepsilon$ a 0,1. No es preciso aplicar el suavizado en todos los pasos del esquema de Runge-Kutta. En el caso del esquema de 4 pasos, por ejemplo, aplicar el suavizado únicamente a los pasos 1 y 3 suele ser suficiente para doblar el límite de estabilidad.

## 5. Detalles adicionales de implementación

### Elección de limitadores

La mayor parte de los análisis con PUMI se llevan a cabo usando el limitador de Van Albada [3] que ofrece

buenos resultados en circunstancias comunes

$$\mathbf{l}^i = \text{máx}\left[\frac{2\Delta_i^- \Delta_{ij} + \varepsilon}{\left(\Delta_i^-\right)^2 + \left(\Delta_{ij}\right)^2 + \varepsilon},\ 0\right] \tag{63}$$

En (63) $\varepsilon$ es un número pequeño que evita divisiones por cero cuando el flujo es uniforme ($\varepsilon \sim 10^{-5}$). Para casos en los que resulta difícil lograr la convergencia existe la opción de utilizar el limitador MINMOD para aumentar la robustez (con la contrapartida de obtener una solución más difusiva) [5]

$$\mathbf{l}^i = \begin{cases} sgn(\Delta_i^-)\frac{\text{mín}\left[\left|\Delta_i^-\right|,\ |\Delta_{ij}|\right]}{\text{máx}\left[\left|\Delta_i^-\right|,\ |\Delta_{ij}|\right]} & \text{si } sgn(\Delta_i^-) = sgn(\Delta_{ij}) \\ 0 & \text{en otro caso} \end{cases} \tag{64}$$

En caso de elegir esta opción, los cálculos pueden agilizarse utilizando una versión simplificada del limitador MINMOD reemplazando (49) por

$$\mathbf{Fa}_k^{ij}\Big|_{\text{lím } ited} = -\mathbf{n}_k^{ij}\left|\tilde{\mathbf{A}}_{Roe}^{n^{ij}}\right| \Delta_{ij}^{\text{lím } ited}$$

$$\Delta_{ij}^{\text{lím } ited} = \begin{cases} \Delta_{ij} - \text{máx}\left[\text{mín}\left(\Delta_i^-, \Delta_j^+\right), 0\right] & \text{si } \Delta_{ij} > 0 \\ \Delta_{ij} - \text{mín}\left[\text{máx}\left(\Delta_i^-, \Delta_j^+\right), 0\right] & \text{en otro caso} \end{cases} \tag{65}$$

Al emplear (65) se hace innecesario el cálculo de los estados extrapolados (55) simplificando en cierta medida los cálculos.

Es común encontrar situaciones en las que la solución diverge durante los primeros pasos del análisis. Esto ocurre, por ejemplo, cuando las condiciones iniciales son extremadamente distintas de la solución estacionaria. Para mejorar el comportamiento en estos casos se ofrece al usuario la posibilidad de llevar a cabo un cierto número de pasos usando el esquema de primer orden (esto es, anulando todos los limitadores) al comienzo del análisis. Esto suele ser suficiente para lograr unas condiciones lo suficientemente próximas al estado estacionario como para que el esquema de alto orden converja sin problemas.

Hacia el final de la simulación, cuando el sistema se encuentra muy cerca del estado estacionario, la convergencia puede verse dificultada por la continua fluctuación de los limitadores. La no-linealidad debida a éstos reduce la tasa de convergencia sin aportar un aumento apreciable de precisión durante las etapas finales [8]. Para superar esta limitación existe la opción de congelar los limitadores una vez que la solución se encuentra suficientemente cerca de las condiciones estacionarias. Suele procederse a la congelación de los limitadores en el instante en el cual el residuo se ha reducido en tres órdenes de magnitud respecto a su valor original. El usuario puede también elegir llevar a cabo la congelación de los limitadores de una manera progresiva, reduciendo se frecuencia de actualización según progresa el cálculo. La disminución de la frecuencia de actualización tiene la ventaja adicional de reducir el coste computacional de los pasos de tiempo.

Para aquellas aristas que contengan nodos en la frontera el proceso de extrapolación se complica por el hecho de que alguno de los nodos virtuales suele yacer en el exterior del dominio fluido. Como la información empleada para recuperar el valor de la solución en un punto de estas características procede únicamente del lado interior, es de esperar una pérdida de precisión en el proceso de extrapolación. Esto puede tener como consecuencia que los limitadores en los nodos del contorno tengan valores demasiado bajos, introduciendo así un exceso de difusión artificial. El efecto puede apreciarse en la solución en forma de un exceso de curvatura de los contornos de número de Mach constante cerca de las paredes. El problema es especialmente severo en el caso de mallas groseras. Para mitigar este inconveniente el código ofrece la posibilidad de utilizar un valor unitario de los limitadores para los nodos superficiales. De esta manera el proceso de extrapolación se lleva a cabo únicamente usando información del nodo interior de la arista, donde la pérdida de precisión es mucho menor.

### Recuperación de derivadas

Para recuperar los valores nodales del gradiente de la solución se supone que el campo de derivadas puede aproximarse usando las funciones de forma y los valores nodales

$$\frac{\partial \tilde{\boldsymbol{\Phi}}}{\partial x_k} = \nabla_k \tilde{\boldsymbol{\Phi}} = N_j(\mathbf{x})\ \frac{\partial \tilde{\boldsymbol{\Phi}}}{\partial x_k}\bigg|_{x^j} = N_j\ \nabla_k \tilde{\boldsymbol{\Phi}}^j \qquad k = 1, 2, 3 \tag{66}$$

Por otro lado, los valores del gradiente en el interior del elemento pueden obtenerse derivando las funciones de forma

$$\frac{\partial \tilde{\boldsymbol{\Phi}}}{\partial x_k}\bigg|_{el} = \frac{\partial N_j}{\partial x_k}\tilde{\boldsymbol{\Phi}}^j = \nabla_k N_j\, \tilde{\boldsymbol{\Phi}}^j \tag{67}$$

Igualando las distribuciones (66) y (67) en el sentido débil se tiene

$$\int_\Omega W N_j\ \nabla_k \tilde{\boldsymbol{\Phi}}^j\ d\Omega = \int_\Omega W \nabla_k N_j\, \tilde{\boldsymbol{\Phi}}^j\, d\Omega\ \ \forall\ W \tag{68}$$

donde $W$ es una función de prueba genérica. Empleando el método de Galerkin (haciendo $W = N_i$) se llega a

$$\int_\Omega N_i N_j\ \nabla_k \tilde{\boldsymbol{\Phi}}^j\ d\Omega = \int_\Omega N_i \nabla_k N_j\, \tilde{\boldsymbol{\Phi}}^j\, d\Omega\ \forall\ W \tag{69}$$

que es un sistema lineal de ecuaciones para las incógnitas $\nabla_k \tilde{\boldsymbol{\Phi}}^j$.

$$\begin{array}{l} \mathbf{M}\nabla_k \tilde{\boldsymbol{\Phi}} = \mathbf{D}\tilde{\boldsymbol{\Phi}}^j \\ \mathbf{D} = \int_\Omega N_i \nabla_k N_j \; d\Omega \end{array} \tag{70}$$

Téngase en cuenta que el lado derecho de (70) puede ensamblarse por aristas ya que similar al segundo miembro de (18). La expresión anterior, que utiliza la matriz de masas consistente, requiere la solución de un sistema lineal de ecuaciones. En el caso de PUMI se ofrecen dos alternativas para resolver el sistema:

- Obtener una solución aproximada reemplazando $\mathbf{M}$ en (70) por la matriz de masas diagonal
- Resolver el sistema mediante un algoritmo iterativo que requiere invertir únicamente la matriz diagonal

Cuando se elige la segunda opción la solución se obtiene a partir del siguiente esquema.

$$\begin{array}{l} \mathbf{x}^0 = \mathbf{M}_d^{-1}\mathbf{b} \\ \mathbf{x}^n = \mathbf{M}_d^{-1}\left(\mathbf{b} - \mathbf{M}\mathbf{x}^{n-1}\right) \end{array} \tag{71}$$

Normalmente bastan 2 ó 3 iteraciones para que el algoritmo (71) obtenga una buena aproximación de la solución [3]. El incremento de tiempo de cálculo es pues muy pequeño. Una vez que la solución se aproxima al estado estacionario los gradientes del paso anterior constituyen una excelente aproximación inicial, acelerando aún mas la convergencia de (71).

### Condiciones de contorno

Las condiciones de contorno que pueden imponerse quedan englobadas en dos categorías; paredes (deslizamiento) y campo lejano (valores de la corriente libre).

Se supone que las paredes son impermeables, por tanto la componente normal de velocidad debe anularse sobre ellas. Esta condición puede imponerse de forma débil haciendo que el flujo a través de la frontera (24) tome el valor:

$$\mathbf{u}\cdot\mathbf{n} = 0 \;\Rightarrow\; \mathbf{F}_n = \mathbf{F}\cdot\mathbf{n} = \begin{bmatrix} 0 \\ p\,n_1 \\ p\,n_2 \\ p\,n_3 \\ 0 \end{bmatrix} \tag{72}$$

esto es, solo persiste el término de presión del flujo de cantidad de movimiento. Desafortunadamente, el uso de (72) por si mismo no garantiza que el flujo másico sea nulo en todos los puntos de la frontera (ya que la condición sólo se impone en término medio). Esta limitación es especialmente notoria en el punto de remanso delantero del cuerpo. Para obtener resultados satisfactorios se fuerza también a que la componente normal de velocidad en la superficie sea nula. La velocidad obtenida tras cada etapa del esquema de integración se rectifica mediante:

$$\mathbf{u}_{Corr} = \mathbf{u} - \alpha\,(\mathbf{u}\cdot\mathbf{n})\,n \qquad \alpha \in [0,1] \tag{73}$$

El parámetro $\alpha$ en (73) se eleva progresivamente desde cero al comienzo del análisis hasta la unidad al cabo de cierto número de pasos de tiempo [8]. De esta forma la condición de deslizamiento se impone de manera progresiva, evitando cambios bruscos en el primer paso que podrían afectar la convergencia. Esto es especialmente importante cuando las condiciones iniciales son muy diferentes de la solución estacionaria (p. ej. si como condición inicial se toma la del fluido sin perturbar). El vector normal en cada nodo se obtiene como la media ponderada de las normales de todas las caras externas que contengan a dicho nodo

$$\mathbf{n}^i_{nod} = \frac{\sum_k A_k \mathbf{n}^k_{el}}{\sum_k A_k} \tag{74}$$

donde el elemento $k$ contiene el nodo $i$.

Nótese que en caso de existir aristas vivas en la superficie del sólido la normal no está correctamente definida y (73) no puede imponerse correctamente. Se distinguen dos casos:

1. Si la arista reside en una concavidad de la superficie, la condición (73) se sustituye por

   $$\mathbf{u}_{Corr} = (\mathbf{u}\cdot\mathbf{t})\,\mathbf{t} \tag{75}$$

   donde $\mathbf{t}$ es el vector unitario tangente a la arista (es decir, se fuerza a que la velocidad sea paralela a la arista). La condición (72) se impone en las caras de los elementos que rodean a la arista.
2. Por el contrario, si la superficie es convexa, la dirección de la velocidad no se conoce *a priori* y por tanto no puede imponerse. La velocidad en la arista, en general, no será paralela a ninguna de las dos caras que concurren en la arista con lo cual (73) no es aplicable. En este caso el valor del flujo normal se calcula de la manera usual en cada una de las caras, sin imponer ninguna condición a la velocidad. Esto acontece, por ejemplo, en el borde de salida de un ala.

El tratamiento de la condición de campo lejano es algo más delicado. En la frontera exterior del dominio, donde el fluido entra o sale, sólo las componentes de la solución que se desplazan hacia el interior pueden ser impuestas. Por el contrario, aquellas que se mueven

hacia el exterior deben tomar su valor de la solución en el interior del dominio. No es sin embargo práctico realizar el cambio a variables características, elegir en función del sentido de propagación y finalmente deshacer la transformación para volver a variables conservativas. El mismo resultado se puede conseguir usando el solver aproximado de Roe para calcular los flujos en la frontera exterior [8]. Sea $\tilde{\mathbf{A}}^n_{Roe}$ el jacobiano de flujo en dirección normal a la frontera para el estado intermedio entre el de la frontera exterior y las condiciones del flujo sin perturbar. Se tiene:

$$\mathbf{F}^{\infty}_n - \mathbf{F}^{O}_n \simeq \tilde{\mathbf{A}}^n_{Roe}\left(\boldsymbol{\Phi}^{\infty} - \boldsymbol{\Phi}^{O}\right) \tag{76}$$

donde los superíndices $\infty$ y $O$ indican condiciones de corriente libre y en la frontera exterior respectivamente. El valor del flujo a prescribir, teniendo en cuenta las características de propagación de la solución, puede aproximarse usando el jacobiano positivo

$$\mathbf{F}^{Wave}_n = \frac{\mathbf{F}^{\infty}_n + \mathbf{F}^{O}_n}{2} - \left|\tilde{\mathbf{A}}^n_{Roe}\right| \frac{\boldsymbol{\Phi}^{\infty} - \boldsymbol{\Phi}^{O}}{2} \tag{77}$$

Utilizando el flujo en la frontera exterior dado por (77) se logra el comportamiento correcto sin necesidad de tener que prescribir directamente los valores en la frontera. Este método garantiza que las perturbaciones que se generan en el interior del dominio no se reflejen de vuelta en el contorno exterior. Se minimizan así las perturbaciones en la solución y no es preciso hacer ninguna distinción entre contornos de entrada/salida o subsónicos/supersónicos a la hora de general los archivos de entrada.

### Mejora de las prestaciones

Para reducir el tiempo de solución en ordenadores de sobremesa y estaciones de trabajo modernos se ha implementado un mecanismo de reordenación nodal a fin de reducir la probabilidad de fallos de cache durante la ejecución. Comenzando por el primer nodo (cuya elección es arbitraria) a todos los nodos conectados a él se le asigna una numeración consecutiva. A continuación se repite el proceso comenzando con los nodos recién renumerados. El ciclo continúa hasta que todos los nodos de la malla hayan recibido una nueva numeración. Así, cuando se opera con las aristas, disminuye la probabilidad de un fallo de cache puesto que nodos vecinos se almacenan unos cerca de otros en la memoria del ordenador. El resultado del proceso de renumeración depende obviamente de la elección del nodo inicial. Puede contrarrestarse este efecto haciendo varias pasadas del algoritmo usando como primer nodo en cada una el último punto del ciclo anterior [3].

Para acelerar la ejecución en sistemas procesadores con varios núcleos y en sistemas con varios procesadores el código se ha paralelizado utilizando directivas OPENMP. Para garantizar la ausencia de conflictos (*race conditions*) durante las escrituras, cada hilo de ejecución activo mantiene una copia privada del vector de residuos. De esta manera todas las etapas del ensamblaje del sistema se pueden llevar a cabo sin interacción entre los diferentes procesos. Una vez que todas las copias privadas del vector han sido ensambladas se procede a una etapa de reducción. En la mismo todas las copias privadas se fusionan (también en paralelo) para obtener el lado derecho completo de la ecuación. Una vez hecho esto la solución mediante la matriz de masas diagonal puede ejecutarse de manera concurrente. Este método de aislar los procesos lleva asociada una pequeña penalización en uso de memoria (ya que se almacenan varias copias parciales del vector, una por hilo) pero no afecta seriamente el comportamiento en tanto el número de procesos concurrentes no sea elevado. Puesto que PUMI ha sido diseñado para ejecutarse en equipos con un número limitado de procesadores (estaciones de trabajo de gama baja) la pérdida de eficiencia es muy limitada.

## 6. Ejemplo de aplicación

El código ha sido probado exitosamente en casos de relevancia industrial al llevar a cabo simulaciones de un avión de transporte subsónico completo. El estudio se ha completado en el ámbito del proyecto REMFI (Rear Empennage and Fuselage Flow Investigation) parte del 6° programa marco de la UE. Un estudio muy detallado del flujo en torno a las superficies de cola de un avión de gran capacidad se ha llevado a cabo. También se han estudiado los efectos aeroelásticos y de interferencia aerodinámica debidos al mecanismo de soporte doble empleado durante la campaña de ensayos experimentales [1,16]. Esta configuración permite la medición directa en el túnel aerodinámico de las cargas que actúan sobre la sección de cola. Debido al aumento de los esfuerzos de torsión sobre el ala es importante buscar un espaciado óptimo entre soportes. El objetivo es lograr un compromiso entre la reducción de la interferencia aerodinámica directa en la cola y las deformaciones de torsión en el ala (que podrían introducir errores adicionales debidos a los cambios en la deflexión de estela). Para mejorar la comprensión de los fenómenos involucrados debió simularse el modelo de avión completo, incluyendo los mecanismos de suspensión.

En la Figura 1 se muestran tres configuraciones diferentes estudiadas. La malla de superficie para uno de estos casos se muestra en la Figura 2 junto con un detalle de la zona de unión del ala al brazo de suspensión

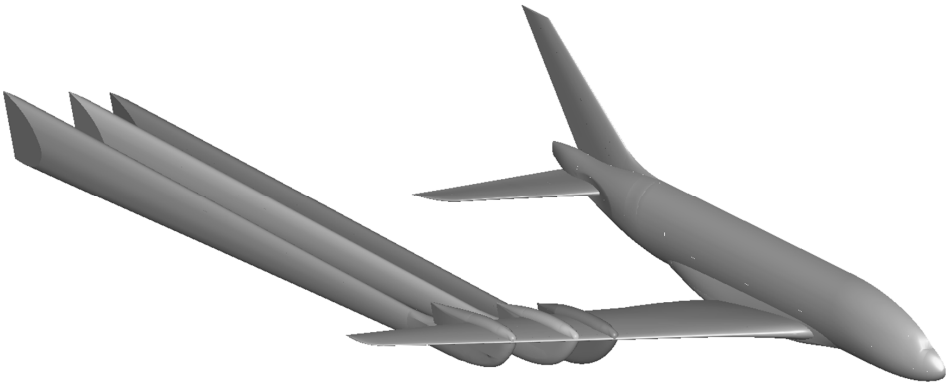

**Figura 1.** Tres diferentes posiciones del mecanismo de suspensión estudiadas

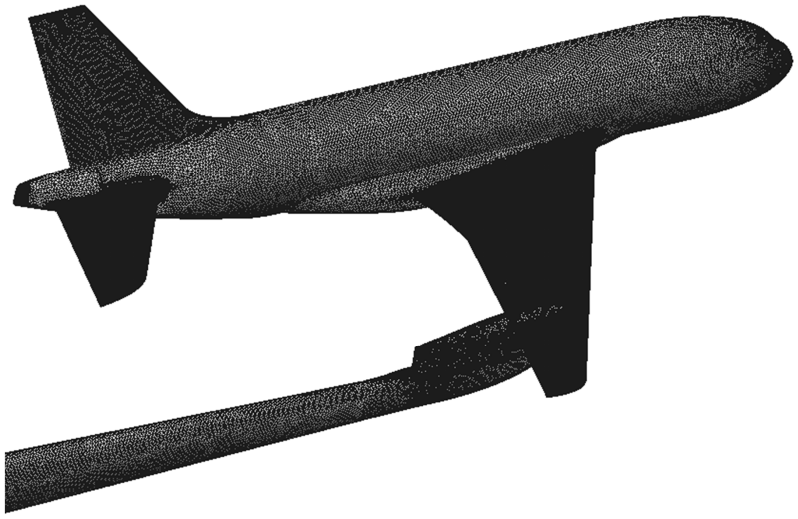

**Figura 2.** Malla de superficie del modelo, incluyendo los brazos de soporte y los adaptadores de unión ala-brazo

(Figura 3). Para discretizar de manera satisfactoria la geometría se precisó una malla de 1,53 millones de nodos formada por 8,27 millones de elementos tetraédricos. El uso de memoria típico del código es de 1GB por cada 10 millones de elementos. En particular, el modelo de la Figura 2 requiere un espacio de 853MB para su ejecución. Queda así demostrada la eficiencia en lo que a gestión de memoria se refiere. En un ordenador de sobremesa de gama media (Core 2 Q9400) el tiempo de CPU necesario para cada paso de tiempo (usando el esquema de integración de cuarto orden) es de 4,7s para la ejecución en serie. En caso de disponer de dos procesadores la disminución del tiempo de cálculo es del 40 %. En la Figura 4 se muestra la historia de convergencia de una solución típica. Puede comprobarse como a partir de 4000 pasos de tiempo la no linealidad introducida por los limitadores impide la obtención de una tasa de convergencia monótona. En el ejemplo mostrado se había configurado el programa para congelar los limitadores en el momento en el que los residuos disminuyesen 5 órdenes de magnitud respecto al valor inicial. Se aprecia claramente en la curva que la tasa se convergencia se acelera notablemente a partir de los 5600 pasos de tiempo. Se alcanza una reducción de seis órdenes de magnitud en el residuo (condición considerada universalmente como suficiente para la obtención de soluciones de calidad industrial) a los 6500 pasos. En el caso mostrado en la Figura 4 se ha retrasado intencionadamente la congelación de los limitadores para ilustrar de manera más dramática las oscilaciones debidas a su constante fluctuación. En aplicaciones prácticas suele ser apropiado proceder a la congelación cuando los residuos se han reducido 3, ó a lo sumo 4, órdenes de magnitud. Para el caso mostrado este punto corresponde a 2000 iteraciones. Se logra así la convergencia en sólo 4000 iteraciones. Esto permite obtener una solución de calidad industrial en menos de 5h (en el caso de ejecución en serie). Cabe destacar que el tiempo empleado en cada paso de tiempo se reduce a medida que progresa el análisis ya que la disminución de la frecuencia de actualización de los limitadores reduce el número de operaciones necesario.

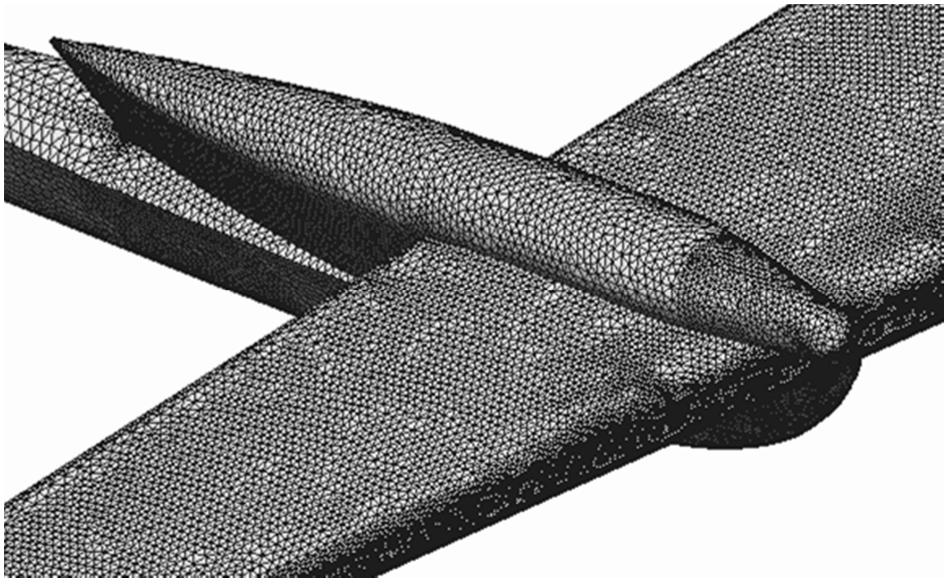

**Figura 3.** Detalle de la malla de superficie en la zona de anclaje

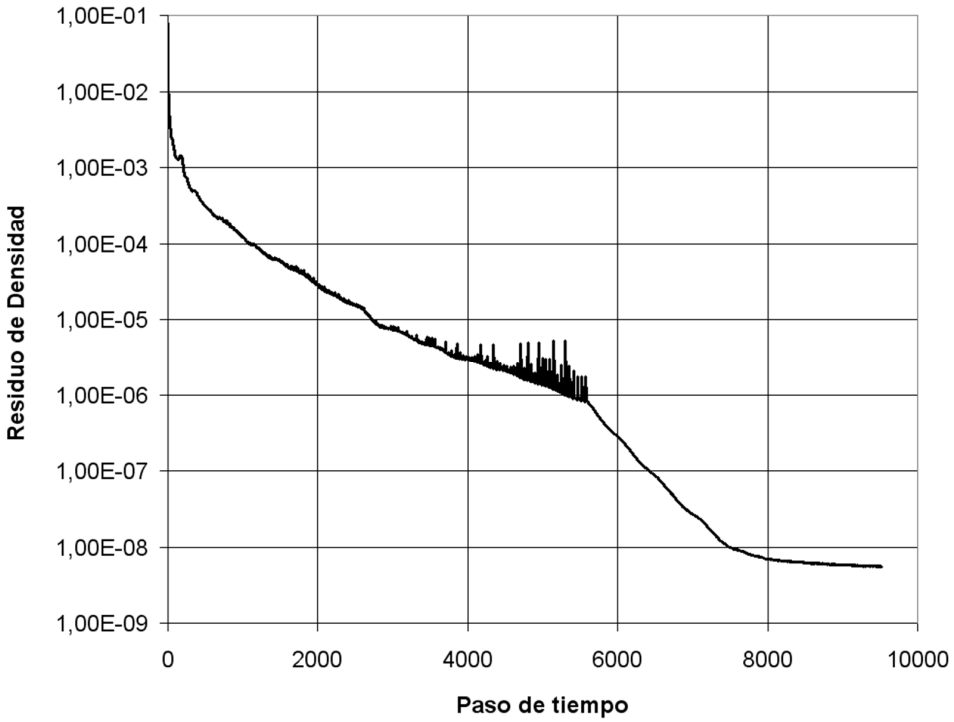


**Figura 4.** Historial de convergencia (residuo de densidad)

Se ensayaron diferentes espaciados de los brazos de soporte para un amplio rango de condiciones de vuelo, cubriendo desde el régimen subsónico bajo ($M = 0,35$) hasta el transónico alto ($M = 0,95$, muy por encima del régimen de crucero correspondiente al tipo de avión considerado). En la Figura 5 puede verse la distribución de presiones para un caso subsónico bajo ($M = 0,35$). Tal como corresponde a este régimen, se aprecia un pico de succión muy intenso cerca del borde de ataque seguido de un aumento de presión progresivo sobre la mayor parte del extradós del ala. La Figura 6 ilustra una condición próxima al punto de diseño del ala ($M = 0,85$). Puesto que ésta emplea perfiles supercríticos la variación de presión en la zona de succión es muy suave y la recompresión tiene lugar mediante una onda de choque situada relativamente cerca del borde de salida. Puede apreciarse como la onda de choque está bien definida en la zona exterior del ala mientras que resulta mucho

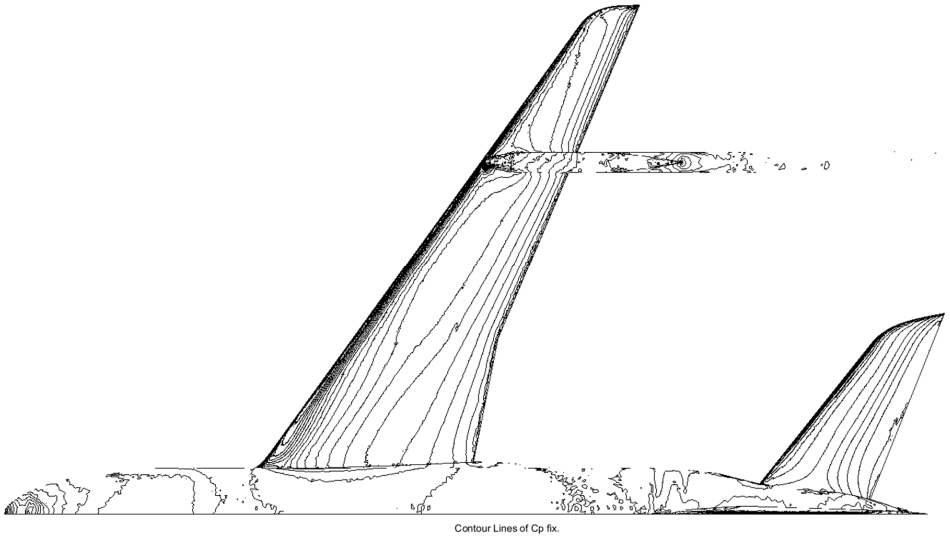


**Figura 5.** Isobaras en régimen subsónico bajo ($M = 0,35$)

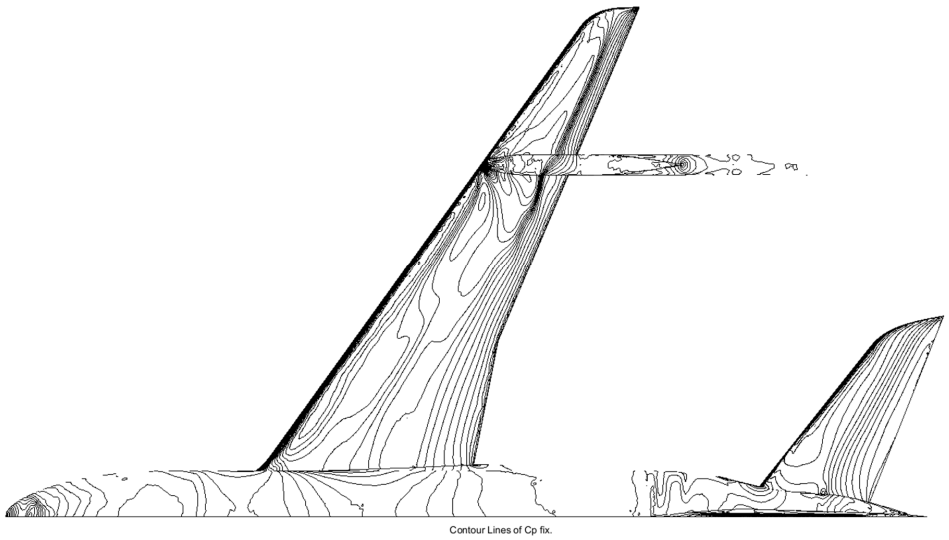


**Figura 6.** Isobaras en régimen transónico cerca del punto de diseño ($M = 0,85$)

más difusa en la raíz. Este comportamiento anómalo se debe a que para los ensayos en túnel se combinaron el fuselaje y el ala de dos modelos diferentes. Se manufacturó una nueva carena ala-fuselaje genérica (no optimizada aerodinámicamente) y por tanto el flujo en la zona interior del ala resultó perturbado. La Figura 8 muestra un detalle de las perturbaciones en la distribución de sustentación debidas a la presencia del mecanismo de soporte. Finalmente la Figura 7 corresponde al extremo de la envolvente de vuelo (condiciones de picado de diseño, $M = 0,95$). Se ha alcanzado la divergencia de sustentación y la onda de choque se ha desplazado hasta el borde de salida. Puede apreciarse también la aparición de una onda de choque en el extradós del estabilizador horizontal, señal inequívoca de que la aeronave opera fuera de sus parámetros de vuelo normales. Durante la campaña de ensayos del proyecto REMFI se tuvo la oportunidad de contrastar los resultados numéricos con mediciones en el túnel aerodinámico a fin de validar los resultados del código. En la Figura 9 la distribución de presión calculada al 70 % de la envergadura del estabilizador horizontal (en condiciones de crucero, $M = 0,85$) se compara con las mediciones experimentales procedentes del túnel del viento. Puede comprobarse como existe un muy buen acuerdo entre las dos series de datos. Téngase en cuenta que al tratarse de la comparación de una solución de Euler con resultados experimentales (afectados por la viscosidad) son de esperar ligeras diferencias.

En el marco REMFI también se llevaron a cabo estudios detallados de la actuación del timón de dirección. Puesto que la deflexión del timón rompe la simetría del

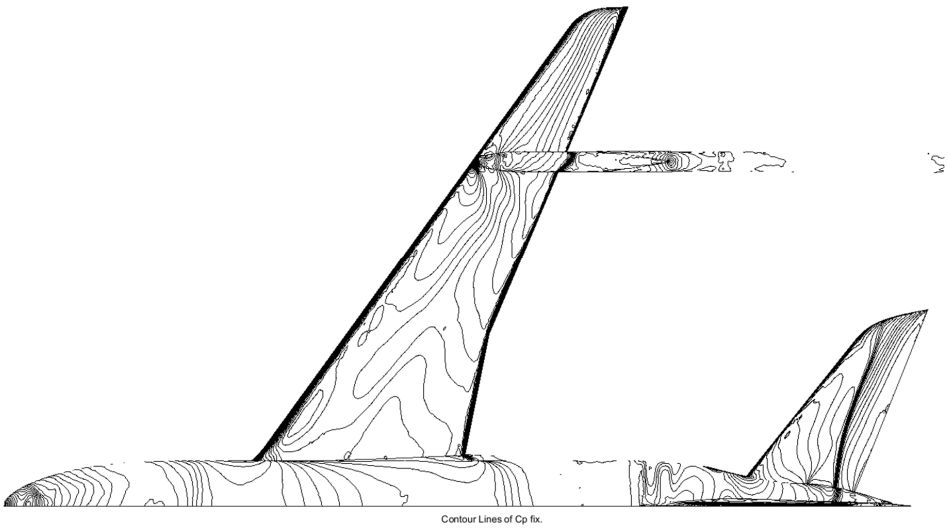


**Figura 7.** Isobaras en régimen transónico alto ($M = 0,95$))

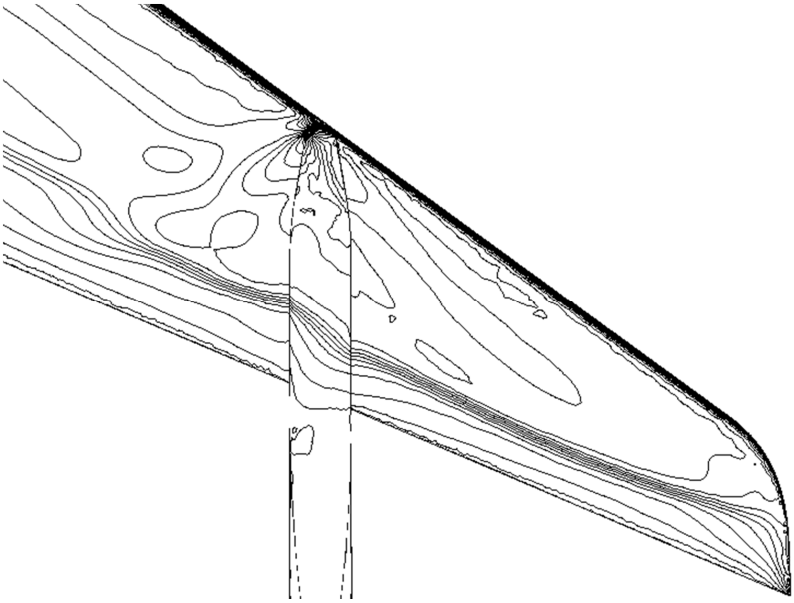

**Figura 8.** Detalle de la distribución de presión en el ala mostrando las alteraciones debidas a la presencia del mecanismo de soporte ($M = 0,85$))

modelo es preciso mallar la totalidad del mismo. Una de las geometrías analizadas, correspondiente a una deflexión de 20°, se muestra en la Figura 10. La malla correspondiente consta de 2,22 millones de nodos y 12,0 millones de celdas tetraédricas. Los requisitos de cómputo por paso de tiempo del solver crecen linealmente con el tamaño de la malla. Para este caso son necesarios 1,2GB de memoria y el coste por iteración es de 6,9s (en las etapas iniciales, cuando los limitadores se actualizan a cada paso). Es destacable que los requisitos de memoria del código son muy modestos por lo que es posible estudiar casos de elevada complejidad incluso en sistemas operativos de 32bits en los que cada proceso esté limitado a un máximo de 2GB. En la Figura 11 se muestra la distribución de presiones correspondiente a la deflexión del timón en régimen subsónico bajo.

## 7. Conclusiones

Se han expuesto los fundamentos matemáticos y detalles de implementación de un código de Euler 3D basado en elementos finitos. El programa ha sido diseñado para obtener resultados en tiempos reducidos con unas exigencias limitadas de memoria y capacidad de proceso. Adicionalmente, el software ha sido paralelizado para su ejecución en estaciones de trabajo multi-

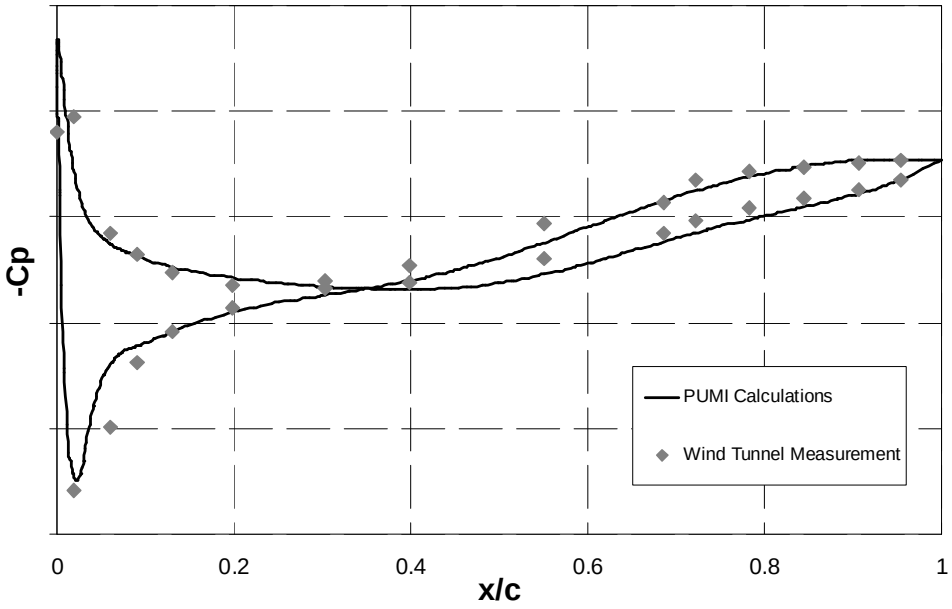


**Figura 9.** Distribución de presiones en el estabilizador horizontal. Comparación de la simulación con los registros obtenidos en túnel aerodinámico)

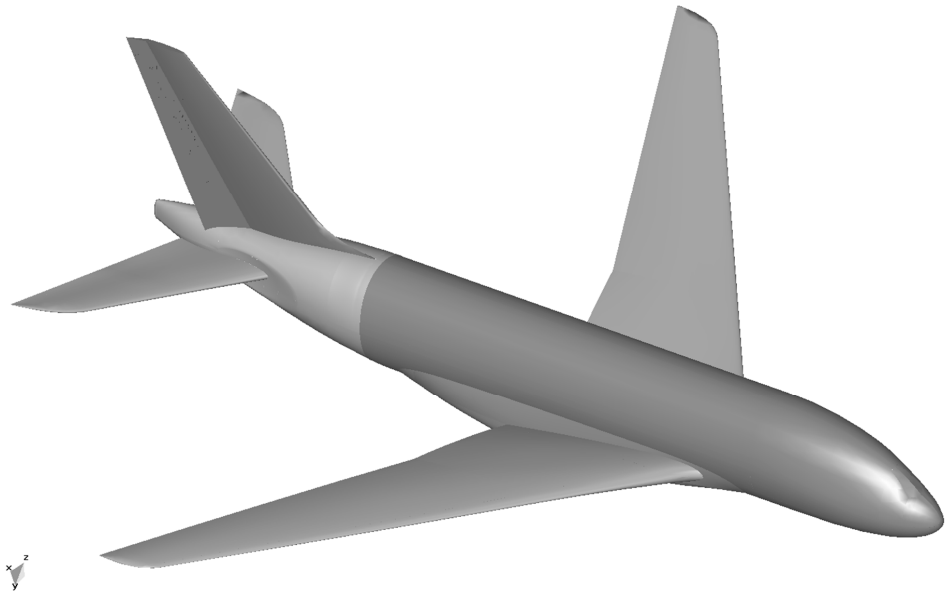

**Figura 10.** Geometría para el estudio de las actuaciones del timón de dirección (ángulo de deflexión $\delta = 20°$)

procesador del segmento bajo del mercado. El programa se encuentra especialmente adaptado para su uso en hardware de prestaciones modestas (ordenadores de sobremesa) reteniendo la capacidad para tratar configuraciones de interés industrial (mallas de decenas de millones de elementos). Al tiempo que el coste computacional se mantiene bajo, se obtienen resultados de buena calidad tal como demuestra la comparación con ensayos experimentales.

## AGRADECIMIENTOS

El presente trabajo se llevó a cabo con el apoyo financiero obtenido a través del proyecto REMFI, encuadrado en el 6 Programa Marco de la Comunidad Europea (contrato AST3-CT-2004-502895).

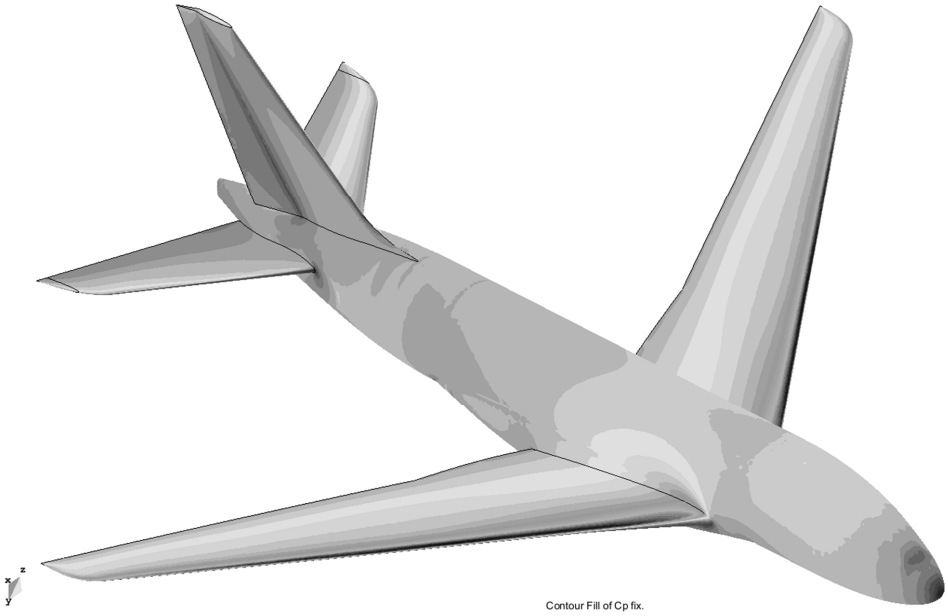


**Figura 11.** Distribución de presiones sobre el avión (deflexión de timón $\delta = 20°$, $M = 0,35$))

## REFERENCIAS


1. Dadvand P. (2007) A Framework for Developing Finite Element Codes for Multidisciplinary Applications. Ph.D. Thesis
2. Donea J., Huerta A. (2003) *Finite Element Methods for Flow Problems*. John Wiley & Sons Ltd.
3. Löhner R. (2001) *Applied CFD Techniques*. John Wiley & Sons Ltd.
4. Morgan K., Peraire J., Peiró J. (1992) Unstructured Grid Methods for Compressible Flows. In Report 787 – Special Course on Unstructured Grid Methods for Advection Dominated Flows. AGARD
5. Morgan K., Peraire J. (1998) Unstructured Grid Finite-Element Methods for Fluid Mechanics. *Rep. Prog. Phys.* 61:569-638
6. Lyra P.R.M., Morgan K. (2000) A Review and Comparative Study of Upwind Biased Schemes for Compressible Flow Computation. Part I: 1-D First-Order Schemes. *Arch. Comput. Methods Eng.* 7(1):19-55
7. Lyra P.R.M., Morgan K. (2000) A Review and Comparative Study of Upwind Biased Schemes for Compressible Flow Computation. Part II: 1-D Higher-order schemes. *Arch. Comput. Methods Eng.* 7(3):333-377
8. Lyra P.R.M., Morgan K. (2002) A Review and Comparative Study of Upwind Biased Schemes for Compressible Flow Computation. Part III: Multidimensional Extension on Unstructured Grids. *Arch. Comput. Meth. Eng.* 9:207-256
9. Lomax H. (2001) *Fundamentals of Computational Fluid Dynamics*. Springer-Verlag,
10. Hirsch C. (1990) *Numerical Computation of Internal and External Flows.* Volume 2. John Wiley & Sons.
11. Turkel E. (1988) Improving the Accuracy of Central Difference Schemes. ICASE Report 8853. September
12. Hu G. (2001) The Development and Applications of a Numerical Method for Compressible Vorticity Confinement in Vortex Dominant Flows. PhD Thesis. Virginia Polytechnic
13. Swanson R.C., Turkel E. (1997) Multistage Schemes with Multigrid for Euler and Navier-Stokes Equations. Components and Analysis'. NASA Technical Paper 3631. August
14. Hirsch C. (1990) *Numerical Computation of Internal and External Flows: Volume 2*. John Wiley & Sons
15. Zienkiewicz O., Taylor R. (2000) *The Finite Element Method: Volume 3. Fluid Dynamics*. th edition. Butterworth-Heinemann
16. Flores R., Ortega E., Oñate E. (2010) Numerical Investigation of wind-Tunnel Model Deformations Caused by the Twin-Sting Support System. *AIAA Journal of aircraft*. 47:708-714