Métodos Numéricos en Ingeniería

Editores
F. Navarrina
M. Casteleiro

VOLUMEN

2

SEMNI
Sociedad Española de Métodos Numéricos en Ingeniería

II Congreso de Métodos Numéricos en Ingeniería
F. Navarrina y M. Casteleiro (Eds.). ©SEMNI 1993

APLICACION DE PROCESOS READAPTATIVOS A PROBLEMAS EVOLUTIVOS BIDIMENSIONALES

L.Ferragut
Dpto. de Mat. Aplicada y Métodos Informáticos, Universidad Politécnica de Madrid, Ríos Rosas 21, 28003 Madrid, ESPAÑA

A. Plaza y **R. Montenegro**
Dpto. de Matemáticas, Univ.de Las Palmas de G. C., 35017 Las Palmas de Gran Canaria, ESPAÑA

RESUMEN

Hasta hace poco tiempo, para resolver problemas evolutivos los métodos adaptativos de elementos finitos usaban, por lo general, mallados no estructurados, sobre todo si la región de refinamiento variaba en cada instante de tiempo. Últimamente se ha demostrado que la combinación de un algoritmo de desrefinamiento con otro de refinamiento local, proceso readaptativo [1-4], dota a los mallados estructurados de la flexibilidad de que hacían gala los no estructurados y es muy útil para usarse junto con métodos multimalla en la resolución del sistema de ecuaciones asociado al método de elementos finitos.

Algunas propiedades de estos procesos son que la estrategia de refinamiento/desrefinamiento queda definida por el usuario al comienzo de cada ejecución con un pequeño número de parámetros, la adaptación de la malla es automática en cada instante de tiempo, esta adaptación es muy rápida y el número de nodos en el mallado -y por tanto el de incógnitas en el sistema algebraico asociado- varía poco en todo el proceso.

En esta comunicación se analiza la complejidad del algoritmo de desrefinamiento y se aplica el proceso readaptativo a la resolución de un problema evolutivo.

1. INTRODUCCIÓN.

La capacidad de generar automáticamente y de controlar adaptativamente las discretizaciones usadas en la solución numérica de las ecuaciones en derivadas parciales es muy importante para aplicar con éxito las técnicas del Análisis Numérico [5]. Se han dedicado muchos esfuerzos para lograr una adaptación de la malla automática que evite la necesidad de realizar varias pruebas antes de encontrar una estrategia óptima a fin de lograr la precisión deseada [6]. Este trabajo se inscribe en el marco de la generación de una malla óptima bidimensional y de la progresiva automatización de un método de elementos finitos, dentro del contexto de los mallados estructurados y de la utilización de métodos multimalla.

Para resolver problemas evolutivos, si el área de refinamiento varía con el tiempo se hace necesario combinar el refinamiento local con el desrefinamiento de mallado con objeto de mantener el número de nodos acotado a lo largo de todo el proceso. Nosotros usamos una versión del algoritmo 4-T de Rivara [7-9] para refinar la malla [10,11]. La elección del algoritmo de refinamiento es importante ya que el algoritmo de desrefinamiento se puede entender como su algoritmo inverso.

Hay cuatro apartados principales en este trabajo. En el siguiente se describe de forma general el algoritmo de desrefinamiento y se muestra cómo actúa en una secuencia de cinco niveles de malla. En el tercer apartado se estudia la complejidad y eficiencia del algoritmo de desrefinamiento. En el capítulo de aplicaciones se muestra un problema evolutivo de convección-difusión resuelto mediante un proceso readaptativo. Finalmente se resaltan las conclusiones más importantes.

2. EL ALGORITMO DE DESREFINAMIENTO.

La bibliografía sobre algoritmos de desrefinamiento en mallas estructuradas es todavía pequeña. Hasta donde nosotros alcanzamos, sólo M.C. Rivara [12] ha desarrollado otro algoritmo de desrefinamiento. Nuestro algoritmo puede encontrarse más desarrollado en [1-4].

En general, tenemos una secuencia de mallas anidadas:

$$\mathbf{T} = \{ \tau_1 \leq \tau_2 \leq \ldots \leq \tau_n \}$$

y queremos hallar $\mathbf{T}' = \{ \tau_1 \leq \tau_2 \leq \ldots \leq \tau_m \}$ donde $m \leq n$, tras desrefinar T.

2.1. Esquema del Algoritmo.

Un esquema básico del algoritmo de desrefinamiento es:

ENTRADA: Secuencia $\mathbf{T} = \{ \tau_1 \leq \tau_2 \leq \ldots \leq \tau_n \}$

Bucle en niveles de malla, desde la última hasta la segunda. Sea el nivel de malla j: Desde j = n hasta 2, hace:

1. Se recorren los *nodos propios* de τ_j y se evalúa la *condición de desrefinamiento.*
2. Se asegura la *conformidad* de la malla que se esta creando, minorando la zona que se desrefina.

3.a. Si algún nodo propio de τ_j debe ser eliminado, entonces:

3.a.1. Si algún nodo propio de τ_j debe permanecer, entonces:

Se definen las nuevas conexiones nodales para el nuevo nivel, sea éste τ'_j.

3.a.2. En caso contrario, i.e., si se eliminan todos los nodos propios de τ_j, entonces:

Se elimina el nivel *j* de los vectores de estructura.

Los cambios se heredan a los siguientes niveles de malla.

Fin del si.

3.b. Si todos los nodos propios de τ_j deben permanecer, entonces:

No se modifica el nivel *j*: $\tilde{\tau}_j = \tau_j$.

Fin del si.

4. Se obtiene una nueva secuencia de mallas que representamos por $T^j = \{ \tau_1 \leq \tau_2 \leq ... \leq \tau_{j-1} \leq \tilde{\tau}_j \leq ... \leq \tilde{\tau}_{nj} \}$.

Fin del bucle en niveles de malla.

SALIDA: Secuencia $T' = \{ \tau_1 \leq \tau'_2 \leq ... \leq \tau'_m \} = \{ \tau_1 \leq \tau^2_2 \leq ... \leq \tau^2_m \}$.

Dentro del algoritmo de desrefinamiento, podemos distinguir dos problemas: el geométrico y el algebraico. Al primero corresponde el esquema anterior y es el más complicado. El segundo consiste en la reasignación de nuevos números de ecuaciones a los grados de libertad que han quedado en la malla final después del desrefinamiento, a la vez que se conserva en los nodos que permanecen en el mallado la solución numérica anterior.

Nada se ha comentado de la estructura de datos utilizada por el algoritmo de desrefinamiento; ver, por ejemplo [4]. Se puede señalar que es del orden del número de nodos existente en el dominio, si bien, se almacena la numeración global de todas las caras y de todos los elementos de todos los niveles de malla en forma secuencial. Por esto, puede decirse que la memoria requerida para las caras y los elementos es del orden del número de niveles de malla, *n*, multiplicado por el número de nodos, *NN*: $O(n \cdot NN)$.

En la figura 1 se muestra la aplicación del algoritmo de desrefinamiento a una secuencia de cinco niveles de malla. Se ha omitido el primer nivel al ser el mismo en todas las secuencias. La primera línea representa la secuencia inicial; en ella se han resaltado los *nodos propios* de cada nivel, es decir los nodos creados en ese nivel de malla. En las siguientes líneas se representan las secuencias T^j que resultan en cada iteración del bucle en niveles de malla. En cada una de ellas se resalta el nivel $\tilde{\tau}_j$. Los nodos blancos son los nodos propios de cada nivel susceptibles de ser eliminados. Los nodos señalados en negro son nodos propios que deberán permanecer para que la conformidad de las mallas quede asegurada. Se puede observar cómo se heredan los cambios a las subsecuencias finales en cada iteración.

2.2. La condición de desrefinamiento.

Sobre la condición de desrefinamiento se puede resaltar que se utiliza la diferencia absoluta o la diferencia relativa, entre la solución numérica en cada nodo *K* y la solución interpolada de la solución numérica en los nodos extremos de la cara en la que *K* está en el punto medio (*cara-entorno de K*).

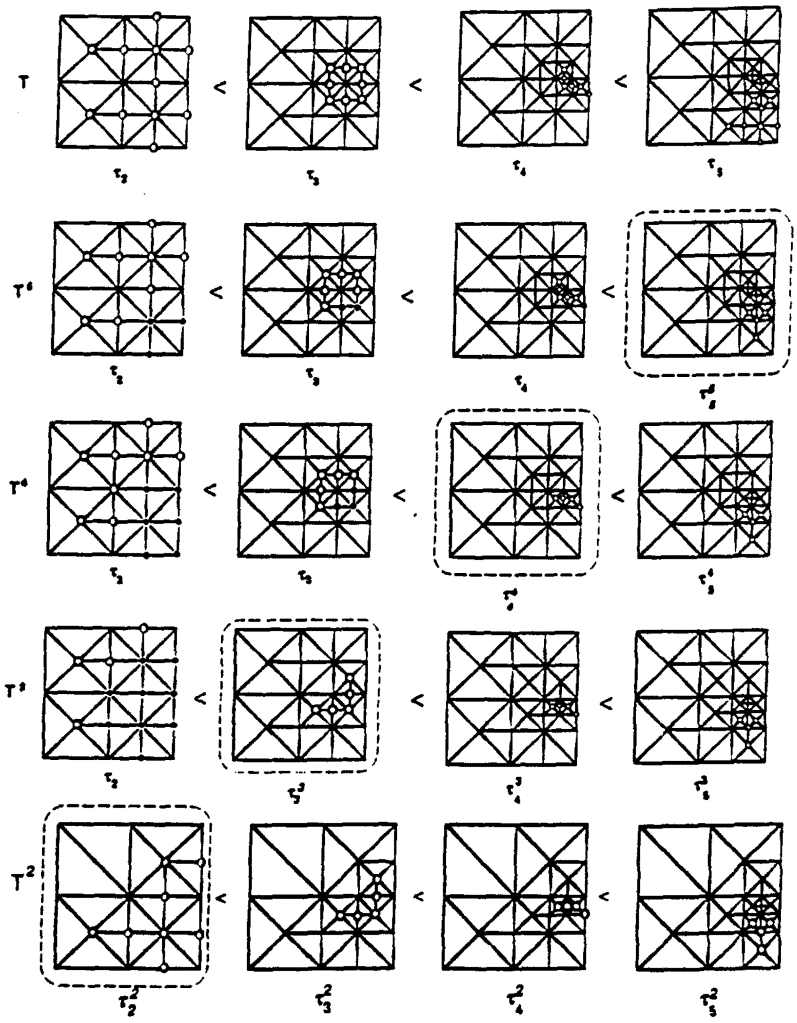


Figura 1.- Aplicación del Algoritmo de Desrefinamiento.

En la figura 2 se muestra un nodo propio K y su *cara-entorno* con nodos extremos K+1 y K-1. Para decidir si se puede eliminar el nodo K o no, se debe comprobar si para todos los grados de libertad por nodo, y respecto de un parámetro de desrefinamiento, fijado por el usuario en la entrada de datos, ε, se cumple la condición:

$$\left|u_h^l(K)-u_i^l(K)\right| \le \varepsilon \qquad (1)$$

o bien la condición:

$$\left|u_h^l(K)-u_i^l(K)\right| \le \varepsilon \cdot \left|u_h^l(K)\right|, \qquad (2)$$

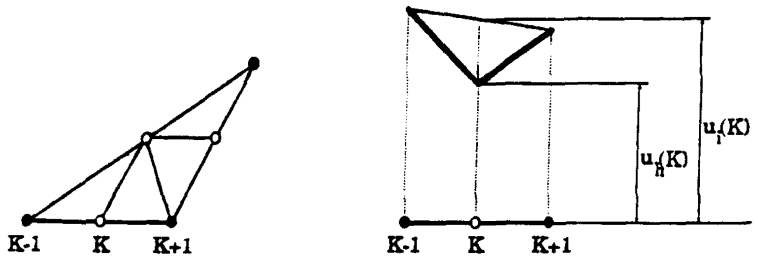


Figura 2.- La condición de desrefinamiento.

donde $u_h^l(K)$ es la solución numérica en el nodo K correspondiente al grado de libertad l, y $u_l^l(K)$ es la solución interpolada en K, es decir:

$$u_l^l(K) = \frac{u_h^l(K-1)+u_h^l(K+1)}{2} \qquad (3)$$

Hemos usado la condición de diferencia absoluta cuando tenemos alguna información sobre el rango de los valores que tomará la solución en el dominio, de acuerdo con las características físicas del problema. Sin embargo, si se usa la diferencia relativa, ha de tenerse cuidado con los errores de redondeo en las zonas en las que la solución sea muy cercana a cero.

2.3. Procedimiento de Conformidad.

Como se ve, la condición de desrefinamiento es muy fácil de verificar. Hay que señalar que, como queda reflejado en la figura 1, la condición de desrefinamiento se comprueba en un número de nodos mínimo, mediante el uso de unos indicadores de desrefinamiento. Más detalles pueden encontrarse en las referencias [2-4].

En cuanto a cómo se asegura la conformidad del naciente nivel de malla τ_j se puede decir lo siguiente: asegurar la conformidad de la malla implica, si es necesario, variar los indicadores de desrefinamiento de algunos nodos y caras; la conformidad se asegura minorando la zona desrefinada, es decir que por razones de conformidad algunos nodos, que respecto de la condición de desrefinamiento podrían ser quitados, han de permanecer en la malla. Este procedimiento se puede resumir como sigue:

ENTRADA: Especificaciones geométricas del nivel τ_j y del anterior, indicadores de desrefinamiento de nodos y caras.

NE(j-1)= nº de elementos de τ_{j-1}.
Mientras haya que lograr la conformidad, hace:
  Para i=1, hasta NE(j-1), hace:

t_i = i-ésimo elemento de τ_{j-1}.
Si t_i tiene hijos en τ_j,
Se logra la conformidad local.
Fin del si.
Fin del bucle en elementos.
Fin del mientras.

SALIDA: Especificaciones geométricas del nivel τ_j y del anterior, indicadores de desrefinamiento de nodos y caras, nº de nodos marcados para eliminar. Indicación de si el desrefinamiento es parcial o total.

La conformidad local se logra cambiando, si es necesario, el indicador de desrefinamiento del nodo que está en el punto medio del lado mayor. Ahora bien, si se cambia el indicador de algún nodo en la conformidad local se hace necesario, de nuevo, asegurar la conformidad con un nuevo bucle en elementos de la malla τ_{j-1}.

3. COMPLEJIDAD Y EFICIENCIA DEL ALGORITMO.

Hemos estimado el orden de operaciones que realiza el algoritmo de desrefinamiento en función de dos parámetros: el número de nodos en el mallado NN y el número de niveles de malla n. Daremos un cota superior del número de operaciones que conlleva el algoritmo de desrefinamiento, en su aspecto geométrico.

Teniendo en cuenta el esquema del algoritmo y denotando el número de nodos propios de la malla τ_j por $NNP(j)$, se tiene, en cuanto a número de operaciones: el paso 1 del algoritmo tiene una complejidad de $O(NNP(j))$; el asegurar la conformidad $O(NN \cdot NNP(j))$; los pasos 3.a.1 y 3.a.2: $O(NN)$; y, por último: los cambios se heredan a los siguientes niveles de malla: $O(n \cdot NN)$.

Teniendo en cuenta que el número de elementos de la malla τ_{j-1} lo podemos suponer del orden de NN, asegurar la conformidad tendrá un coste no superior a $NN \cdot NNP(j)$. Los pasos 1 y 2 se pueden hacer independientes del bucle en niveles de malla, pues cada nodo sólo se evalúa, como mucho una vez en todo el algoritmo. Es decir, sabiendo que

$$\sum_{j=2}^{n} NNP(j) \leq NN \tag{4}$$

se tiene:

$$O(n \cdot NNP(j)) = O(NN)$$
$$O(n \cdot NN \cdot NNP(j)) = O(NN^2) \tag{5}$$

Por tanto, el algoritmo tiene una complejidad del orden de

$$O(NN^2 + NN + n^2 \cdot NN) = O(NN^2 + n^2 \cdot NN) \qquad (6)$$

La conformidad, sin embargo, se puede asegurar *al mismo tiempo* que se recorren los nodos propios de cada nivel de malla y se evalúa la condición de desrefinamiento, evitándose los bucles en los elementos de la malla anterior e implicaría, salvo constantes multiplicativas, tantas operaciones como nodos propios de cada nivel. Es decir, los apartados 1 y 2 del algoritmo tendrían una complejidad de $O(NN)$. Además, en este caso, si está asegurado que el número de niveles de malla es acotado, se tendría una complejidad del algoritmo lineal $O(NN)$. En realidad, este análisis nos da una cota superior de la complejidad del algoritmo; la experiencia nos dice que el tiempo de cálculo empleado por la ejecución del algoritmo es despreciable frente al total (del orden del 1%).

4. APLICACIONES NUMÉRICAS.

En este apartado presentamos un problema modelo evolutivo de convección-difusión. Los principales aspectos numéricos sobre la formulación semi-implícita empleada para su resolución están recogidos en [13].

Sea un cuadrado de lado unidad, Ω, centrado en el punto $(0.5, 0.5)$, con condiciones de tipo Neumann nulas en su frontera Γ. Supongamos que en ese dominio tenemos un campo de velocidades giratorio $\vec{v} = (v_1, v_2)$:

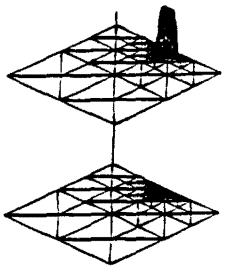
$$\left.\begin{aligned} v_1 &= \omega\, x_1(1-x_1)(x_2-0.5) \\ v_2 &= \omega\, x_2(0.5-x_1)(1-x_2) \end{aligned}\right\} \qquad (7)$$

de tal forma que $\vec{\nabla} \cdot \vec{v} = 0$ y $\vec{v}$ es tangente a Γ.

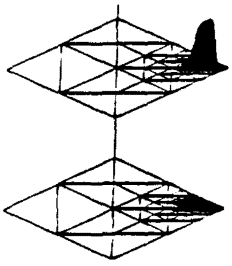
El máximo valor del módulo de la velocidad es $\omega/8$ y éste se alcanza en los puntos medios de los lados, mientras que en las esquinas del cuadrado la velocidad se hace cero. En esta aplicación hemos elegido $\omega = 2.5 \cdot 10^4$ y difusión $k = 1$. Por tanto, el número de Peclet es $3.125 \cdot 10^3$. Se han supuesto también fuerzas externas nulas: $f = 0$.

La solución inicial tiene la forma de una meseta semicilíndrica y ha sido aproximada usando una estrategia readaptativa, véase fig. 3(a). Partiendo de una malla inicial muy grosera, compuesta por 8 triángulos y 9 nodos, se realizan 3 refinamientos globales para capturar la solución inicial. Después se ha aplicado un desrefinamiento con parámetro $\varepsilon = 0.005$ y diferencia absoluta para disminuir el número de nodos en el dominio. Este proceso se ha repetido dos veces, pasándose de 1081

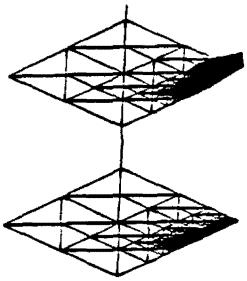
nodos en la última malla refinada a 234 nodos en la malla desrefinada (fig. 3(a)). Mediante esta estrategia logramos una buena aproximación con un número de nodos mínimo.



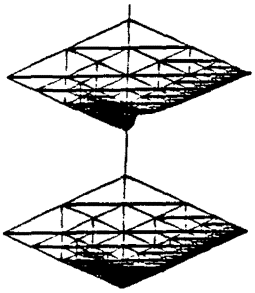
(a) Solución inicial, t = 0.0, 234 nodos.



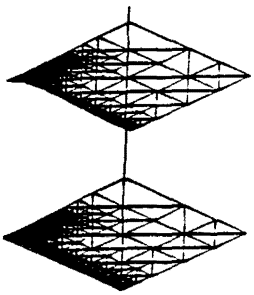
(b) t = 0.00014, 580 nodos.



(c) t = 0.00032, 801 nodos.



(d) t = 0.00060, 628 nodos.



(e) t = 0.00097, 337 nodos.

Figura 3 .- Mallas y soluciones para un problema evolutivo.

Hemos introducido un nuevo indicador, ε_l, para un elemento Ω_l:

$$\varepsilon_l = h_l^2 |\nabla u_h| \qquad (8)$$

siendo h_l el diámetro del elemento Ω_l y u_h la solución numérica lineal en el elemento triangular. Este indicador de refinamiento es una medida de la máxima variación de u_h en Ω_l ponderada con el diámetro h_l del elemento, con el fin de que las zonas de alto gradiente no se refinen excesivamente. Se ha utilizado como parámetro de refinamiento $\gamma = 0.6$, y de desrefinamiento $\varepsilon = 0.001$, y condición de desrefinamiento la diferencia absoluta.

Para llegar a la solución estacionaria, constante, fueron necesarios 2.892 pasos de tiempo. El paso de tiempo se calcula automáticamente para cada una de las diferentes mallas que resultan en todo el proceso, atendiendo a la cota de estabilidad propuesta en [13]. Se realizaron 723 desrefinamientos y 2.169 refinamientos locales. El máximo número de niveles de malla presentes en una de las secuencias manejadas fue de 167. La ejecución del programa se paró automáticamente al alcanzarse un número de niveles de malla, 3, que fue definido en la entrada de datos, cuando quedaban 81 nodos igualmente distribuidos en el dominio, tras 6 horas y 20 minutos de CPU, en un Stardent-3000 con compilación escalar.

5. CONCLUSIONES.

Los procesos readaptativos son muy útiles para resolver problemas evolutivos, especialmente aquellos que requieren zonas de refinamiento móviles. La combinación refinamiento, desrefinamiento y métodos multimalla promete ser muy eficaz para resolver problemas más complejos incluidos problemas no lineales, dado el bajo coste computacional de los métodos multimalla [14-15].

Experimentalmente se ha comprobado que el tiempo de CPU requerido por el algoritmo es despreciable con respecto al total de ejecución (menos del 1 %).

Es importante destacar que, mediante los procesos readaptativos, el número de nodos y su situación en el mallado está controlado automáticamente por el código de acuerdo con la estrategia utilizada y con la solución numérica en cada instante de tiempo.

6. AGRADECIMIENTOS.

Este trabajo ha sido parcialmente financiado por la FUNDACIÓN UNIVERSITARIA DE LAS PALMAS (GRAN CANARIA), con Patrocinador: REFINERÍA ACEITERA CANARIA, S.A. (RACSA).

7. REFERENCIAS.

1. L. Ferragut, A. Plaza y R. Montenegro, "Procesos readaptativos de mallas estructuradas: refinamiento/desrefinamiento", *Actas* XII C.E.D.Y.A., II Congreso de Matemática Aplicada, 1991, pp.453-459.
2. A. Plaza, L. Ferragut and R. Montenegro, "Derefinement algorithms of nested meshes", en J. van Leeuwen (ed.) "Algorithms, Software, Architecture", Proceedings IFIP World Computer Congress, Madrid. Elsevier Science Publishers B.V. (North-Holland), Amsterdam, *1992*, pp. *409-415*.
3. A. Plaza, R. Montenegro and L. Ferragut, "An adaptive refinement/derefinement algorithm of structured grids for solving time-dependent problems", en Ch. Hirsch et al. (eds.), "Numerical Methods in Engineering", Elsevier Science Publishers B.V., Amsterdam, 1992, pp. 225-232.
4. A. Plaza, *Algoritmos de desrefinamiento en mallados estructurados bidimensionales*, Tesis Doctoral, Universidad de Las Palmas de Gran Canaria, 1993.
5. M.S. Shephard and N.P. Weatheerill, Preface, *Int. J. Num. Meth. Eng.* Special Issue on Adaptive Meshing, 32, 651 (1991).
6. O.C. Zienkiewicz and J. Z. Zhu, Adaptivity and mesh generation, *Inter. J. Num. Meth. Eng.* 32, 783-810 (1991).
7. M.C. Rivara, Algorithms for refining triangular grids suitable for adaptive and multigrid techniques, *Int. J. Num. Meth. Eng.*, 20, 745-756 (1984).
8. M.C. Rivara, "A dinamic multigrid algorithm suitable for partial differential equations with singular solutions", in A. Balakrishnan and E. Polak (eds.), Proceedings IFIP Working Conference 1984. Santiago, Chile, "Lesture Notes in Control and Information Sciences", 1986, Cap. 20, pp. 359-370.
9. M.C. Rivara, A grid generator based on 4-triangles conforming. Mesh-refinement algorithms, *Int. J. Num. Meth. Eng.*, 24, 1343-1354 (1987).
10. L. Ferragut, "Una solucion al problema de la programación de métodos de elementos finitos autoadaptativos", Anales Ing. Mec. 5, 201-206 (1987).
11. L. Ferragut, *Neptuno, un sistema de elementos finitos autoadaptativo* (en Fortran), Depto. de Matemáticas Aplicadas y Métodos Informáticos, Madrid (1987).
12. M.C. Rivara, Selective refinement/derefinement algorithms for sequences nested triangulations, *Int. J. Num. Meth. Eng.*, 28, 2889-2906 (1989).
13. R. Montenegro, G. Montero, G. Winter y L. Ferragut, Aplicación de métodos finitos adaptativos a problemas de convección-difusión en 2-D, *Rev. Int. Métodos Num. para Cálculo y Diseño en Ing.*, 5, 535-560 (1989).
14. W. Hackbush and V. Trottengurg (eds.), *Multigrid Methods. Lectures Notes in Mathematics*, Springer-Verlag, Berlin, 1982.
15. W. Hackbush and V. Trottengurg (eds.), *Multigrid Methods II. Lectures Notes in Mathematics*, Springer-Verlag, Berlin, 1986.