Segones Jornades sobre Mètodes Numèrics en
Ciències Aplicades i Enginyería
VALL DE NÚRIA 2003
Laboratori de Càlcul Numèric (LaCàN)

GENERADOR DE MALLAS ADAPTADAS PARA LA SIMULACIÓN DE PROBLEMAS EN LA ATMÓSFERA

Rafael Montenegro*, Gustavo Montero*, José María Escobar* y Eduardo Rodríguez*

* Instituto Universitario de Sistemas Inteligentes y Aplicaciones Numéricas en Ingeniería,
Universidad de Las Palmas de Gran Canaria,
Campus Universitario de Tafira, 35017 Las Palmas de Gran Canaria, España
e-mail: rafa@dma.ulpgc.es

Palabras clave: Mallas de Tetraedros, Adaptabilidad, Elementos Finitos.

Resumen. *El objetivo de este trabajo es presentar los principales aspectos de un código capaz de generar automáticamente mallas de tetraedros a partir de una distribución adecuada de nodos en el dominio de estudio, así como plantear una versión de un algoritmo de refinamiento local basado en la subdivisión en 8-subtetraedros. En concreto, el dominio está limitado en su parte inferior por la superficie del terreno considerada como $z = f(x, y)$, tal que $\forall (x, y) \in [a, b] \times [c, d] \subset R^2 \Rightarrow \exists! z \in R$, y en su parte superior por un plano horizontal. Las paredes laterales están formadas por cuatro planos verticales, paralelos dos a dos. Las ideas básicas para la construcción de la malla combinan, por un lado, la utilización de un algoritmo de refinamiento y desrefinamiento para dominios bidimensionales [6], por otro lado, un algoritmo de generación de mallas de tetraedros basado en la triangulación de Delaunay [5], y además, una función para definir el espaciado vertical de los nodos distribuidos en el dominio. Este código ha sido utilizado anteriormente para generar mallas donde la superficie $z = f(x, y)$ correspondía a una digitalización topográfica del sur de la isla de la Palma, [19] y [20]. Asimismo, en el generador de mallas entran en juego un procedimiento para optimizar la malla resultante y diferentes estrategias para construir la nube de puntos. Una vez que se ha generado la malla atendiendo a la geometría del dominio, es necesario adaptarla a la solución numérica del problema. Por ello, dedicamos una segunda parte de este trabajo al desarrollo e implementación de un algoritmo de refinamiento local de mallas encajadas de tetraedros, cuyo principal interés es su aplicabilidad, rapidez y sencillez. El algoritmo propuesto se basa en la división expuesta en [16] cuyo análisis de calidad se estudia en [15].*

1 INTRODUCCIÓN

El trabajo se ha dividido en dos partes. La primera se desarrolla en la sección 2 y se dedica al análisis del generador automático de mallas de tetraedros de un dominio tridimensional definido sobre un terreno irregular. En la sección 3 presentamos el algoritmo de refinamiento local de mallas de tetraedros. Estas dos grandes líneas se engloban entre los aspectos básicos necesarios para la simulación numérica del problema de dispersión de contaminantes en la atmósfera, que pretendemos afrontrar en los próximos años.

2 GENERACIÓN EFICIENTE DE MALLAS SOBRE OROGRAFÍA IRREGULAR

Inicialmente, nos planteamos crear una malla de tetraedros que respete la topografía de la región de estudio con una precisión determinada. Para ello, se dispone únicamente de la información digitalizada del terreno. Por otra parte, deseamos que la malla esté adaptada con una mayor densidad de nodos donde sea necesario para definir las características geométricas de nuestro dominio a partir de una interpolación lineal a trozos. La malla generada podrá utilizarse como malla base para la simulación numérica de procesos naturales en el dominio; por ejemplo, ajuste de campos de viento [27] y [21], propagación de fuego [18], contaminación atmosférica, etc. Estos fenómenos tienen su mayor efecto en las zonas próximas al terreno, de ahí que también sea deseable que la densidad de nodos aumente al acercarnos a éste. Sobre esta malla base, que únicamente se adapta a las características geométricas del dominio, se podrán aplicar posteriormente algoritmos de refinamiento y desrefinamiento de tetraedros para mejorar la solución numérica del problema [16] y [15]. Estos algoritmos tendrán un especial interés en los problemas evolutivos.

Es bien conocido que para construir una triangulación de Delaunay es necesario definir una nube de puntos en el dominio y su frontera. Estos nodos serán precisamente los vértices de los tetraedros que conforman la malla. La generación de puntos en nuestro dominio se realizará sobre diferentes capas, reales o ficticias, definidas desde el terreno hasta la parte superior del dominio. En concreto, se propone plantear un malla rectangular con una distribución uniforme de puntos en el plano superior del dominio. Esta malla bidimensional puede ser obtenida a partir de la realización de un cierto número de refinamientos globales sobre una malla simple definida en la entrada de datos, o por ejemplo, puede también construirse realizando una triangulación de Delaunay sobre la distribución uniforme de puntos establecida. Consideraremos la malla obtenida como el nivel más bajo de la secuencia que define la distribución de los puntos en el resto de las capas. Sobre esta malla regular aplicamos a continuación el algoritmo de refinamiento y desrefinamiento, [6] y [23], para definir la distribución de los nodos de la capa correspondiente a la superficie del terreno. Para ello, en primer lugar se construye una función que interpola las cotas obtenidas a partir de una digitalización de la topografía de la zona rectangular estudiada. En segundo lugar, realizamos una serie de refinamientos globales sobre la malla uniforme hasta conseguir una malla regular capaz de captar la variación topográfica del terreno. El máximo grado de discretización viene definido por el nivel de detalle de la digitalización. Posteriormente, se realizará un desrefinamiento sobre estos últimos niveles de

malla utilizando como parámetro de desrefinamiento el máximo error de cotas permitido entre la superficie real del terreno y la superficie definida mediante la interpolación a trozos obtenida con la malla resultante. Los fundamentos de este proceso se resumen en la subsección 2.1 del trabajo.

Una vez que se ha definido la distribución de nodos sobre el terreno y sobre el plano superior del dominio, comenzamos a distribuir los nodos situados entre ambas capas. Esta distribución se puede realizar mediante diferentes estrategias, en las que interviene una función de espaciado vertical que se analiza en la subsección 2.2. La característica fundamental de esta función es que el grado de discretización obtenido sobre la vertical debe disminuir con la altura, o a lo sumo mantenerse constante.

La distribución de puntos en el dominio entrará como dato en el mallador tridimensional basado en la triangulación de Delaunay. Para evitar posibles problemas de conformidad con la superficie del terreno, se propone construir la malla de tetraedros con la ayuda de un paralelepípedo auxiliar. Sobre su cara inferior se situan todos los nodos distribuidos sobre el terreno, proyectados sobre un plano horizontal situado a la altura definida por la cota inferior de la región de estudio, y sobre su cara superior se situan los puntos distribuidos en el plano superior del dominio a su altura real. Esto conlleva a realizar una transformación, atendiendo a la función de espaciado sobre cada vertical, para situar el resto de puntos en el paralelepípedo auxiliar. Estos detalles nos asegurarán que la distancia máxima entre dos puntos consecutivos sobre la misma vertical del dominio real será siempre igual o inferior que la correspondiente distancia establecida en el paralelepípedo auxiliar.

Dedicamos la subsección 2.3 a la definición de la nube de puntos en el dominio real, así como su transformación al paralelepípedo auxiliar en el que se construye la malla mediante una versión del método de triangulación de Delaunay [5]. La calidad de la malla final, obtenida mendiante la transformación inversa al dominio real, depende de la distribución de los puntos definida en ambos dominios, ya que respetaremos la topología de la malla obtenida en la triangulación del paralelepípedo auxiliar. Proponemos una estrategia para determinar el número de puntos generados sobre la vertical de cada nodo de la malla bidimensional adaptada a la superficie del terreno, y analizamos sus características fundamentales. Con esta estrategia se generan capas virtuales, es decir, no se define un número concreto de superficies interiores al dominio sobre las que se situan los puntos. Por ello, diremos que el número de capas es variable, y será calculado automáticamente en función de los tamaños de los elementos existentes en la malla bidimensional que define el terreno, y en la correspondiente a la frontera superior del dominio. En concreto, se determina automáticamente, para cada nodo del terreno, una función de espaciado vertical con el objeto de respetar las distancias desde el primer punto generado hasta el terreno, y desde el último punto generado hasta la parte superior del dominio, en función de los tamaños de los elementos existentes sobre ambas superficies.

Una vez que se ha construido la triangulación de Delaunay de la nube de puntos en el paralelepípedo, procedemos a situar los puntos en sus posiciones reales manteniendo la topología de la malla. Hay que tener en cuenta que este proceso de compresión de la malla puede dar lugar a cruces de tedraedros que habrá que deshacer posteriormente. Asimismo, será aconsejable

aplicar una etapa de suavizado para mejorar la calidad de los elementos de la malla resultante. Los detalles sobre el proceso de triangulación se presentan en la subsección 2.4 del trabajo; los relativos al proceso de optimización de la malla se resumen en la subsección 2.5.

En la subsección 2.6 presentamos la malla generada para un problema relativo al sur de la isla de La Palma y su correspondiente malla optimizada, así como las mallas obtenidas en dominios con superficies *test* que presentan fuertes irregularidades. Finalmente, se establecen en la subsección 2.7 conclusiones y líneas futuras.

2.1 Discretización adaptada a la superficie del terreno

El proceso de generación de la malla tridimensional comienza con la determinación de los nodos situados sobre la superficie del terreno. Su distribución debe estar adaptada a las características orográficas con la finalidad de minimizar el número total de nodos necesario. El procedimiento construye inicialmente una secuencia de mallas encajadas $T = \{\tau_1 < \tau_2 < ... < \tau_m\}$ a partir de una triangulación regular τ_1 de la zona rectangular de estudio, tal que el nivel τ_j se obtiene mediante un refinamiento global del nivel anterior τ_{j-1} aplicando el algoritmo 4-T de Rivara [25]; todos los triángulos del nivel τ_{j-1} se dividen en cuatro subtriángulos mediante la introducción de un nuevo nodo en los centros de sus lados y uniendo el nodo introducido en el lado mayor con el vértice opuesto y los otros dos nuevos nodos. Por tanto, en el nivel τ_j aparecen nuevos nodos, lados y elementos que reciben el nombre de propios del nivel j. El número de niveles m de la secuencia está determinado por el grado de discretización de la digitalización del terreno, es decir, el diámetro de la triangulación τ_m debe ser del orden del paso espacial de la digitalización. De esta forma aseguramos que esta malla regular es capaz de captar toda la información orográfica mediante una interpolación de las cotas reales en los nodos de la malla. Finalmente, definimos una nueva secuencia $T' = \{\tau_1 < \tau'_2 < ... < \tau'_{m'}\}, m' \leq m$, aplicando el algoritmo de desrefinamiento, [6] y [23]. En este paso se introduce como dato el parámetro de desrefinamiento ε que determina la precisión con que se desea aproximar la topografía del terreno. La diferencia en valor absoluto de las cotas resultantes en cualquier punto de la malla $\tau'_{m'}$ y su correspondiente cota real será menor que ε. Asimismo, el algoritmo de desrefinamiento utiliza toda la información de la genealogía de elementos y lados definida en la secuencia. A continuación resumimos el algoritmo de desrefinamiento:

ENTRADA: Secuencia $T = \{\tau_1 < \tau_2 < ... < \tau_m\}$.

Bucle en niveles de T; Para $j = m$ hasta 2, hace:

1. Para cada nodo propio de τ_j se evalúa la condición de desrefinamiento y se marcan nodos y lados que podrían ser eliminados mediante los vectores de desrefinamiento.

2. Se asegura la conformidad del nuevo nivel de malla j minimizando la zona desrefinada.

3.a. Si algún nodo propio de τ_j permanece, entonces se definen nuevas conexiones nodales para el nuevo nivel j: τ_j^j. Se modifican los vectores de genealogía de τ_j^j y de τ_{j-1}.

3.b. En otro caso, el nivel actual j es eliminado de los vectores de estructura. Se modifican los vectores de genealogía de τ_{j-1}.

4. Los cambios en la malla se heredan a las mallas siguientes. Se comprimen los vectores de estructura.

5. Se obtiene una nueva secuencia de mallas encajadas $T^{\,j}$. Esta secuencia es la entrada en la siguiente iteración del bucle. $T^{\,j} = \{\tau_1 < \tau_2 < ... < \tau_{j-1} < \tau_j^j < ... < \tau_{m_j}^j\}$.

SALIDA: Secuencia desrefinada $T^{\,\prime} = T^{\,2} = \{\tau_1 < \tau_2' < ... < \tau_{m'}'\}$.

Como condición de desrefinamiento analizamos la diferencia absoluta entre la cota del nodo estudiado y el valor interpolado de las cotas correspondientes a los dos nodos extremos de su lado entorno, es decir, el lado en que ese nodo fue introducido en su punto medio durante el proceso de refinamiento. Si esa diferencia es menor que el parámetro de desrefinamiento ε, entonces el nodo podría ser eliminado, aunque en algunos casos deberá permanecer por razones de conformidad.

Destacamos que la malla bidimensional obtenida puede ser modificada al construir la triangulación de Delaunay en el dominio tridimensional, puesto que lo único que necesitamos y conservamos es la posición de sus nodos. También nos interesa tener presente el nivel en que cada nodo es propio, para proceder a la generación de nodos en el interior del dominio. Este último aspecto se utiliza en las estrategias propuestas.

2.2 Función de espaciado vertical

Como ya se ha indicado, interesa generar una nube de puntos con mayor densidad en la zona cercana al terreno. Para ello, cada nodo va estar situado atendiendo a una función del tipo,

$$z_i = a\, i^{\alpha} + b \tag{1}$$

tal que a medida que aumenta el exponente $\alpha \geq 1$, proporciona una mayor concentración de puntos cerca de la superficie del terreno; z_i es la cota correspondiente al i-ésimo punto insertado, del tal manera que para $i = 0$ se obtiene la cota del terreno y para $i = n$, la cota del último punto introducido que debe coincidir con la altura h del plano superior que delimita el dominio a discretizar. En estas condiciones el número de puntos definidos en la vertical sería $n + 1$ y la función se puede expresar como

$$z_i = \frac{h - z_0}{n^{\alpha}}\, i^{\alpha} + z_0 \quad ; \quad i = 0, 1, 2, ..., n \tag{2}$$

En ocasiones conviene expresar la altitud de un punto en función de la del punto anterior, evitando así tener que conservar en memoria el valor de z_0,

$$z_i = z_{i-1} + \frac{h - z_{i-1}}{n^{\alpha} - (i-1)^{\alpha}}\, [i^{\alpha} - (i-1)^{\alpha}] \quad ; \quad i = 1, 2, ..., n \tag{3}$$

En las ecuaciones (2) o (3), en general, una vez fijados los valores de α y n, los puntos a insertar quedan perfectamente definidos. No obstante, también puede ser interesante fijar la distancia del primer punto insertado ($i = 1$) a la superficie del terreno con el fin de mantener unos parámetros mínimos de calidad en la malla tridimensional que se pretende generar. Esto reduciría el número de grados de libertad a uno, bien sea α o bien n. Consideremos fijado y conocido el valor de esa distancia d tal que $d = z_1 - z_0$. Sustituyendo en la ecuación (2),

$$d = z_1 - z_0 = \frac{h - z_0}{n^{\alpha}} \tag{4}$$

Si fijamos α y dejamos libre el valor de n, de (4) se obtiene,

$$n = \left(\frac{h - z_0}{d} \right)^{1/\alpha} \tag{5}$$

No obstante, en la práctica se aproximará el valor de n al número natural más cercano. En cambio, si fijamos el valor de n y dejamos libre α, resulta,

$$\alpha = \frac{\log \frac{h-z_0}{d}}{\log\, n} \tag{6}$$

En ambos casos, dado uno de los dos parámetros, se calcula el otro mediante las expresiones (5) o (6), respectivamente. De esta forma, la distribución de puntos en la vertical respeta la distancia d entre z_1 y z_0.

Si además fijamos la distancia entre los dos últimos puntos introducidos, esto es $D = z_n - z_{n-1}$, entonces los parámetros α y n quedan perfectamente determinados. Supongamos que α es definido por la ecuación (6). Para $i = n - 1$, la ecuación (2) resulta

$$z_{n-1} = \frac{h - z_0}{n^\alpha}\ (n-1)^\alpha + z_0 \tag{7}$$

y por tanto, usando la ecuación (6),

$$\frac{\log\ (n-1)}{\log\, n} = \frac{\log \frac{h-z_0-D}{d}}{\log \frac{h-z_0}{d}} \tag{8}$$

A partir de las características con que se ha definido la malla buscada *a priori* se puede afirmar que $h - z_0 > D \geq d > 0$. Por tanto, el valor de n estará acotado, $2 \leq n \leq \frac{h-z_0}{d}$, y el de α no podrá ser inferior a 1. Asimismo, para llegar a introducir al menos un punto intermedio entre la superficie del terreno y la frontera superior del dominio, se tiene que verificar que $d + D \leq h - z_0$.

Llamando $k = \frac{\log \frac{h-z_0-D}{d}}{\log \frac{h-z_0}{d}}$, se puede comprobar fácilmente que $0 \leq k < 1$. De esta forma, la ecuación (8) se transforma en

$$n = 1 + n^k \tag{9}$$

Si denotamos $g(x) = 1 + x^k$, se puede comprobar que $g(x)$ es contractiva en $\left[2, \frac{h-z_0}{d}\right]$ con constante de Lipschitz $C = \frac{1}{2^{1-k}}$ y además está acotada,

$$2 \leq g(x) \leq 1 + \left(\frac{h - z_0}{d} \right)^k \leq \frac{h - z_0}{d} \tag{10}$$

En virtud del teorema del punto fijo, podemos asegurar que la ecuación (9) tiene solución única, y puede obtenerse numéricamente, por ejemplo, mediante el método del punto fijo, puesto que éste converge para cualquier aproximación inicial escogida en el intervalo $\left[2, \frac{h-z_0}{d}\right]$. No obstante, en general, la solución no tomará valores enteros. Consecuentemente, si aproximamos su valor al número natural más próximo, la condición impuesta con la distancia D no se cumplirá exactamente, sino de forma aproximada.

2.3 Definición de la nube de puntos

La generación de puntos se realizará en tres etapas. En la primera se define una malla regular bidimensional con la densidad de puntos que se desea obtener sobre la frontera superior del dominio. En segundo lugar, y sobre esta malla τ_1, se procederá a un proceso de refinamiento global y desrefinamiento para obtener la malla $\tau'_{m'}$ que define la distribución de puntos en la superficie del terreno. Para ilustrar estas dos primeras etapas, en la figura 1 se muestra la distribución de puntos sobre ambas superficies para un problema *test*. Una vez definida la distribución de puntos en la superficie del terreno y en la frontera superior del dominio, se procede a la generación de la nube de puntos distribuida entre estas dos capas. Para ello, sobre la vertical de cada nodo P de la malla del terreno $\tau'_{m'}$ situaremos puntos atendiendo a la función de espaciado vertical y al nivel j en el que P es propio, siendo $1 \leq j \leq m'$. La función de espaciado vertical quedará determinada mediante la definición de los siguientes parámetros: la cota topográfica z_0 de P; la altitud h de la frontera superior del dominio; el máximo número posible de puntos $n+1$ en la vertical de P, incluyendo el propio P y el de la frontera superior del dominio, caso de existir; el grado de la función de espaciado α; la distancia entre los dos primeros puntos generados $d = z_1 - z_0$; y la distancia entre los dos últimos puntos generados $D = z_n - z_{n-1}$. De esta forma, la cota del i-ésimo punto generado sobre la vertical de P viene dada por,

$$z_i = \frac{h - z_0}{n^\alpha}\, i^\alpha + z_0 \quad ; \quad i = 1, 2, ..., n-1 \tag{11}$$

Independientemente de la función de espaciado vertical definida, utilizaremos el nivel j en el que P es propio para determinar el número definitivo de puntos que se generan sobre la vertical de P excluyendo el terreno y la frontera superior. Distinguiremos:

1. Si $j = 1$, es decir, si el nodo P es propio de la malla base τ_1, se generan a partir de la ecuación (11) para $i = 1, 2, ..., n-1$.

2. Si $2 \leq j \leq m' - 1$, generamos nodos para $i = 1, 2, ..., min(m' - j, n - 1)$.

3. Si $j = m'$, esto es, el nodo P es propio del nivel más fino $\tau'_{m'}$, entonces no se genera ningún nodo nuevo.

Este proceso tiene su justificación, ya que la malla $\tau'_{m'}$ corresponde al nivel más fino de la secuencia de mallas encajadas $T' = \{\tau_1 < \tau'_2 < ... < \tau'_{m'}\}$, obtenida mediante el algoritmo de refinamiento y desrefinamiento, y por tanto el número de puntos introducidos decrece suavemente con la altura y además resultan distribuidos eficientemente con el fin de construir la malla tridimensional en el dominio.

En [20] se establecen cuatro estrategias para generar la nube de puntos entre el terreno y el plano superior del dominio. En la figura 2 puede observarse el comportamiento de cada una de las cuatro estrategias con la muestra un detalle de la triangulación obtenida en la pared vertical $ADD'A'$ correspondiente al problema test de la figura 1. En la primera estrategia se fijan por el usuario los valores de α y n. En la segunda estrategia imponemos también el mismo valor de n para todo punto P de $\tau'_{m'}$, en cambio, el valor de α se determina automáticamente en función

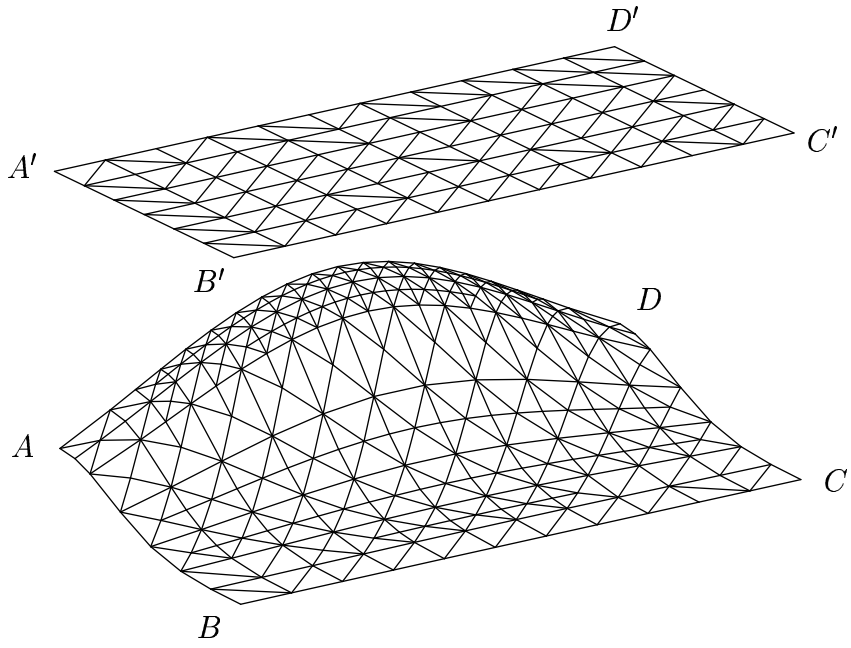


Figura 1. *Representación tridimensional de una distribución de puntos sobre el terreno ($ABCD$) y la frontera superior del dominio ($A'B'C'D'$).*

del tamaño de los elementos más próximos al terreno. En la tercera estrategia imponemos el mismo valor de α para todo punto P de $\tau'_{m'}$, pero el valor de n se determina automáticamente en función del tamaño de los elementos más próximos al terreno.

Finalmente, la cuarta estrategia determina automáticamente los valores de α y n para cada punto P de $\tau'_{m'}$, en función del tamaño de los elementos más próximos al terreno y a la parte superior del dominio. En primer lugar, comenzamos determinando el valor de d para cada punto P como el promedio de las longitudes de las aristas de los triángulos que contienen a P en la malla $\tau'_{m'}$ sobre la superficie del terreno. En segundo lugar, fijamos un único valor de D en función de la distancia deseada entre el último punto que teóricamente sería generado sobre las diferentes verticales y el plano superior del dominio. Esta distancia se determina fácilmente en función del tamaño de los elementos de la malla regular τ_1. Una vez obtenido d y D, para todo punto P de $\tau'_{m'}$, calculamos su correspondiente valor de n resolviendo la ecuación (9). Finalmente, la función de espaciado vertical queda determinada al obtener el valor de α mediante la ecuación (6). Esta estrategia posee, por tanto, como características principales que respeta las distancias exigidas entre la superficie del terreno y la primera capa generada, así como la distancia impuesta entre la última capa virtual generada y la frontera superior del dominio. Por último, se debe realizar la transformación de la nube de puntos,

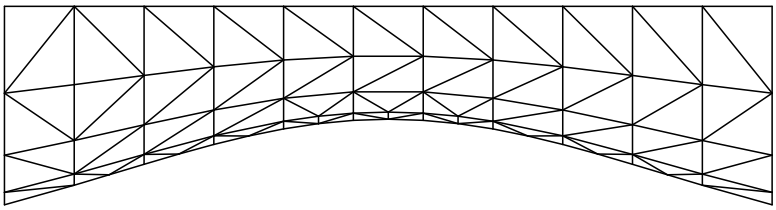
generada en el dominio real, a un paralelepípedo auxiliar. Puesto que tenemos capas virtuales, la transformación de los puntos se realiza según la función de espaciado vertical asociada a cada punto P de $\tau'_{m'}$.

2.4 Generación de la malla tridimensional

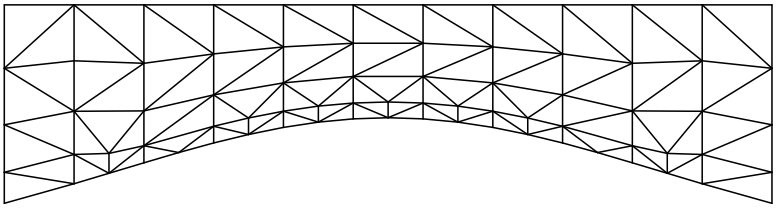
Una vez definida la nube de puntos será necesario construir una malla tridimensional que conecte esos puntos de forma adecuada y que sea conforme con la frontera del dominio, esto es, que respete todas las fronteras establecidas. Aunque la triangulación de Delaunay es apropiada para crear mallas de elementos finitos con un alto grado de regularidad para una nube de puntos dada, no lo es tanto en lo referente al problema de conformidad con la frontera, dado que ésta crea una malla de la envolvente convexa de la nube de puntos. Por esta razón, es posible que no se pueda recuperar la frontera del dominio a partir de las caras y aristas generadas por la triangulación. Para evitar esto, existen básicamente dos tipos diferentes de técnicas. Unas, *conforming Delaunay triangulation* [22], están basadas en la colocación de los puntos siguiendo ciertos criterios de espaciado, de manera que la triangulación resultante sea conforme con la frontera. Las otras, *constrained Delaunay triangulation* [10], están basadas en la modificación de la triangulación en las zonas próximas a la frontera no respetada, mediante intercambio de aristas y caras (*swaping*) de manera que ésta sea recuperada. En nuestro caso, la primera alternativa no resulta adecuada, debido a que deseamos que la malla resultante contenga ciertos puntos definidos de antemano. Además, dada la complejidad de la superficie que define el terreno, esta estrategia conllevaría un alto coste computacional. En principio, la segunda alternativa sería adecuada, si bien requiere unos algoritmos bastante complejos para recuperar la frontera del dominio.

Para construir triangulación tridimensional de Delaunay de la distribución de puntos del dominio comenzamos recolocándolos en un paralelepípedo auxiliar, tal que todos los puntos de la superficie del terreno estén situados en las coordenadas x, y originales, pero a una altura igual a la cota mínima del terreno, z_{min}. En el plano superior del paralelepípedo situamos los nodos del nivel τ_1, de la secuencia de mallas que define la superficie del terreno, a una altura igual a h. En general, los restantes puntos también mantienen sus coordenadas x, y, pero sus cotas se obtienen sustituyendo su correspondiente z_0 por z_{min} en (11). La triangulación de esta nube de puntos se realiza utilizando una variante del algoritmo incremental de Watson [5] que resuelve de manera efectiva los problemas derivados de los errores de redondeo que se producen cuando se trabaja con números en coma flotante. Una vez construida la triangulación en el paralelepípedo, obtenemos la malla final restableciendo sus cotas originales. Este último proceso puede entenderse como una compresión de toda la malla definida en el paralelepípedo, tal que su plano inferior se transforma en la superficie del terreno, con lo que se asegura la conformidad.

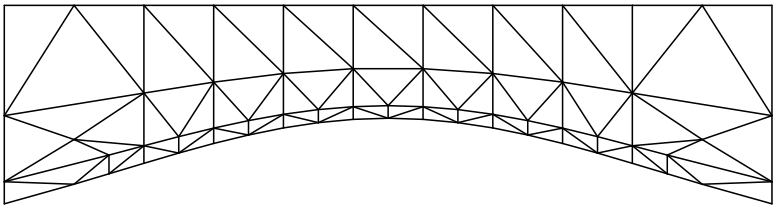
En ocasiones puede ocurrir que al restablecer las posiciones de los puntos a sus cotas reales se produzcan elementos de muy mala calidad o, incluso, *invertidos*, es decir, elementos para los que su volumen V_e, evaluado como el determinante jacobiano $|J_e|$ asociado a la transforma-



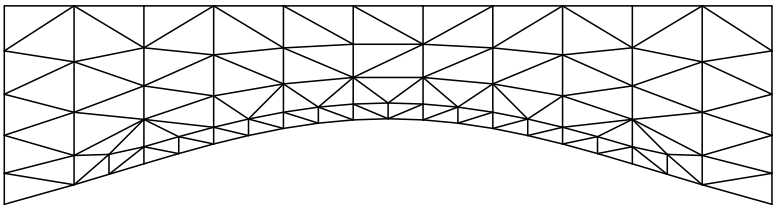
(a) *Estrategia 1*



(b) *Estrategia 2*



(c) *Estrategia 3*



(d) *Estrategia 4*

Figura 2. *Distribución de puntos obtenida sobre la frontera $ADD'A'$ para el ejemplo de la figura 1 mediante las diferentes estrategias.*

ción del tetraedro e al de referencia, pasa a ser negativo. Así, si las densidades de puntos van disminuyendo de forma progresiva con la altura y si los puntos se sitúan sobre capas *suficientemente* separadas en el paralelepípedo, entonces la probabilidad de existencia de elementos de baja calidad o de *cruces* se reduce. En cualquier caso, necesitamos de algún procedimiento capaz de deshacer los cruces que pudieran aparecer y suavizar la malla resultante. Esta cuestión será abordada en la siguiente subsección. Por otra parte, se debe tener en cuenta que la posibilidad de obtener una malla de elevada calidad mediante algoritmos de suavizado basados en movimientos de los nodos alrededor de sus posiciones iniciales depende, además del procedimiento concreto utilizado, de la "calidad topológica" de la malla. Se entiende que ésta es alta cuando la valencia de cada nodo, esto es el número de nodos conectados a él, se aproxima a la que tendría una malla regular formada por tetraedros equiláteros.

La triangulación de Delaunay es capaz de crear una malla de elevada calidad, óptima en 2-D, para una nube de puntos dada. Así pues, un adecuado criterio en la distribución de éstos tendrá como consecuencia la obtención de una malla inicial de elevada calidad. La malla de nuestro dominio conserva la calidad topológica de la triangulación obtenida en el paralelepípedo y, por tanto, un suavizado apropiado conduciría a mallas de gran calidad.

2.5 Optimización de la malla

Las técnicas más comúnmente utilizadas para mejorar la calidad de una triangulación *válida*, esto es, que no posea elementos invertidos, están basadas en el suavizado local. En esencia, dichas técnicas consisten en encontrar las nuevas posiciones que deben ocupar los nodos de la malla de manera que optimicen una determinada función objetivo, basada en alguna medida de la calidad de los tetraedros conectados al nodo ajustable o nodo *libre*. Obviamente, al tratarse de un proceso de optimización local, no se puede garantizar que la malla final sea óptima globalmente. Sin embargo, tras repetir este proceso un cierto número de veces se suele llegar a resultados bastante satisfactorios.

Usualmente, las funciones objetivo son adecuadas para la mejora de la calidad de una malla válida, pero no trabajan correctamente si existen elementos invertidos, debido a que presentan singularidades cuando los volúmenes de los tetraedros cambian de signo. Para evitar este problema se puede proceder según [7], donde se propone un método de optimización que consta de dos etapas. En la primera se resuelven los posibles cruces mediante un algoritmo que maximiza los determinantes jacobianos negativos correspondientes a los elementos invertidos y, en la segunda, se suaviza la malla resultante de la primera fase. Nosotros proponemos aquí una alternativa a este procedimiento, de manera que los procesos de *descruce* y suavizado se realicen en una misma etapa. Para ello, utilizaremos una modificación de la función objetivo propuesta en [4]. Así, supongamos que $N(v)$ designa la submalla formada por el conjunto de los s tetraedros incidentes en el nodo libre v, cuyo vector de posición denotaremos por $\mathbf{r} = (x, y, z)$. Entonces, la función objetivo a minimizar se define como

$$F(\mathbf{r}) = \sum_{e=1}^{s} f_e(\mathbf{r}) \tag{12}$$

siendo f_e la función objetivo asociada al tetraedro $e \in N(v)$, dada por

$$f_e(\mathbf{r}) = \frac{\sum_{i=1}^{6}(l_i^e)^2}{2V_e^{2/3}} \tag{13}$$

donde $l_i^e (i = 1, ..., 6)$ representan las longitudes de las aristas del tetraedro e y V_e su volumen. Si $N(v)$ es una submalla válida, entonces la minimización de F da lugar a posiciones de v para las que la calidad de la submalla $N(v)$ mejora [4]. Sin embargo, F no está acotada cuando el volumen de alguno de los tetraedros de $N(v)$ es nulo. Además, no podemos trabajar con F en el caso de que tengamos tetraedros invertidos. Por tanto, si $N(v)$ contiene algún elemento invertido o con volumen nulo, no será posible encontrar el mínimo relativo por procedimientos convencionales, tales como máximo descenso, gradiente conjugado, etc. Para evitar este problema hemos optado por modificar la función f_e de manera que la nueva función objetivo sea casi idéntica a F en las proximidades del mínimo, pero estando definida y siendo regular en todo $\mathbb{R}^3$. Para ello sustituiremos V_e en (13) por la función creciente

$$h(V_e) = \frac{1}{2}(V_e + \sqrt{V_e^2 + 4\delta^2}) \tag{14}$$

tal que $\forall V_e \in \mathbb{R}$, $h(V_e) > 0$, siendo el parámetro $\delta = h(0)$. De esta forma la nueva función objetivo que proponemos viene dada por

$$\Phi(\mathbf{r}) = \sum_{e=1}^{s} \phi_e(\mathbf{r}) \tag{15}$$

donde

$$\phi_e(\mathbf{r}) = \frac{\sum_{i=1}^{6}(l_i^e)^2}{2[h(V_e)]^{2/3}} \tag{16}$$

El comportamiento asintótico de $h(V_e)$, $h(V_e) \approx V_e$ cuando $V_e \to +\infty$, hará que para un valor de δ suficientemente pequeño y valores positivos de V_e, la función f_e y su correspondiente versión modificada ϕ_e sean tan próximas como se desee. Por otro lado, cuando $V_e \to -\infty$, tenemos que $h(V_e) \to 0$. Todo esto hace que para los tetraedros *más* invertidos tengamos un valor de ϕ_e más desfavorable que para los tetraedros *menos* invertidos. Además, con la función objetivo Φ evitamos los problemas que posee F para tetraedros con volúmenes próximos a cero; la singularidad que f_e presenta, desaparece en ϕ_e debido a la introducción del parámetro δ. A medida que elegimos valores de δ más pequeños, la función ϕ_e se comporta de forma más parecida a f_e. Como consecuencia de estas propiedades se deduce que las posiciones de v que minimizan las funciones objetivo F y Φ son prácticamente idénticas. Sin embargo, a diferencia de lo que ocurre para F, es posible encontrar el mínimo de Φ partiendo desde cualquier posición inicial del nodo libre. En particular, podemos partir de posiciones para las que $N(v)$ no es una submalla válida. Por tanto, con la función Φ podemos deshacer los cruces que aparezcan en la malla y, al mismo tiempo, mejorar su calidad. El valor de δ se escoge en función del punto v en consideración, haciendo que sea lo más pequeño posible y de forma que la evaluación del

mínimo de Φ no presente problemas computacionales. Por último, indicamos que el método utilizado para calcular los mínimos de las funciones objetivo ha sido el de máximo descenso.

2.6 Experimentos numéricos

En esta subsección presentamos una malla generada para un problema relativo al sur de la isla de La Palma y su correspondiente malla optimizada, así como las mallas obtenidas con superficies *test* que presentan fuertes irregularidades.

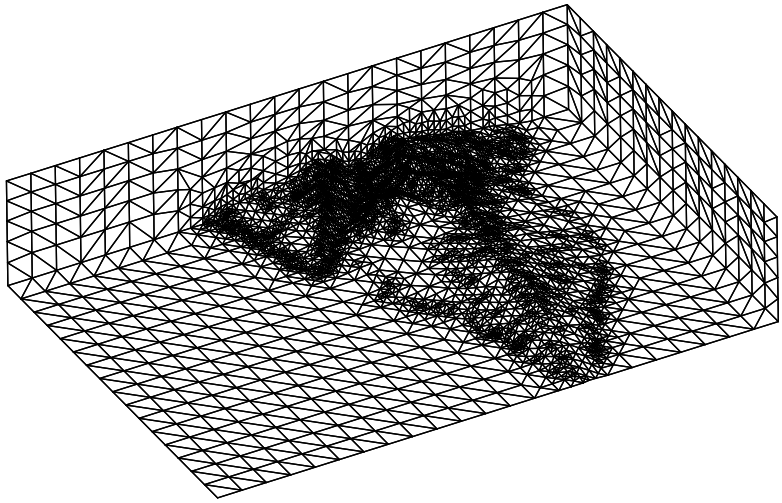
2.6.1 Superficie orográfica de La Palma

Como primera aplicación del generador de mallas se ha considerado una región rectangular del sur de la Isla de La Palma de $45.6 \times 31.2\ km$, en la que las cotas extremas varían de 0 a $2279\ m$ de altitud. La parte superior del dominio se ha establecido a una altitud $h = 9\ km$. Para definir la topografía se dispuso de una digitalización de la zona en la que las alturas estaban definidas sobre una cuadrícula con un paso espacial de $200\ m$ según los ejes x e y. A partir de una malla uniforme de la región rectangular con un tamaño de elementos aproximadamente de $2 \times 2\ km$, se realizaron cuatro refinamientos globales utilizando el algoritmo 4-T de Rivara [25]. Una vez que se interpolaron los datos digitalizados sobre esta malla refinada, se empleó el algoritmo de desrefinamiento desarrollado en [6] y [23] con un parámetro de desrefinamiento $\varepsilon = 40$ m. Esto asegura que la malla adaptada aproxima la superficie del terreno con un error menor que este valor. La distribución de nodos de la malla base regular utilizada antes de los refinamientos globales es la que se consideró sobre la frontera superior del dominio.

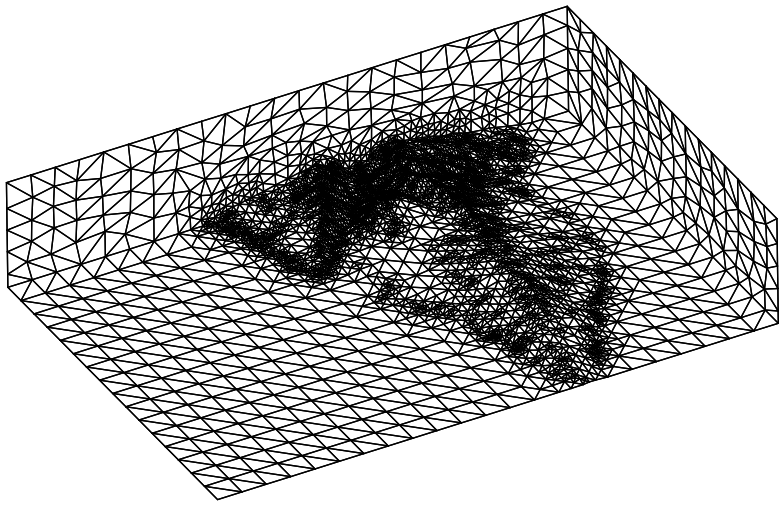
El resultado obtenido con la estrategia 4, propuesta en [20], se muestra en la figura 3 (a), fijando como único parámetro la distancia $D = 1.5\ km$. Se observa que con esta estrategia se conservan automáticamente las distancias existentes en la parte inferior y superior del dominio. En este caso, la malla contiene 57193 tetraedros y 11841 nodos, con una valencia máxima de 26. La distribución de nodos obtenida es de tal calidad que apenas se modifica cualitativamente después de cinco pasos del proceso de optimización (ver figura 3 (b)), exceptuando los cruces iniciales que son resueltos eficientemente (ver figura 4). Para evitar la aparición de tetraedros invertidos se ha aplicado de forma eficiente la técnica propuesta en la subsección anterior. De hecho, la peor medida de calidad de los tetraedros de la malla optimizada está entorno a 0.2. Podemos destacar que el número de parámetros necesarios para definir la malla resultante es muy reducido, así como el coste computacional.

2.6.2 Superficies *test*

Para analizar las posibilidades y limitaciones del generador de mallas, mostramos diversas aplicaciones considerando una serie de superficies *test* con irregularidades. En este apartado, todas las mallas expuestas han sido generadas por nuestro código antes de la aplicación del proceso de



(a) *Malla generada*



(b) *Malla optimizada*

Figura 3. *(a) malla generada y (b) malla resultante después de cinco pasos del proceso de optimización.*

optimización. Todas las superficies han sido definidas en una región rectangular de $10 \times 5\ km$. La parte superior del dominio se ha establecido a una altitud $h = 5\ km$. La primera aplicación corresponde a una superficie gaussiana relativamente suave y se representa en la figura 5 para la estrategia 2. Hemos fijado un parámetro de desrefinamiento $\varepsilon = 20$ m. Esta estrategia introduce 6 capas, 1155 nodos y no produce ningún tetraedro invertido.

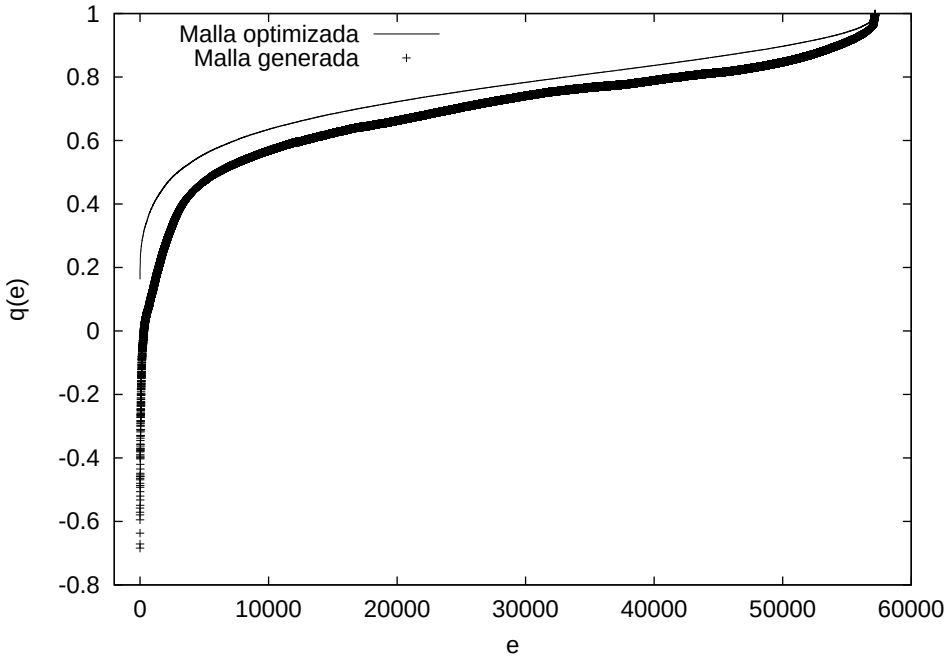


Figura 4. *Curvas de calidad de la malla generada y optimizada para la aplicación de La Palma.*

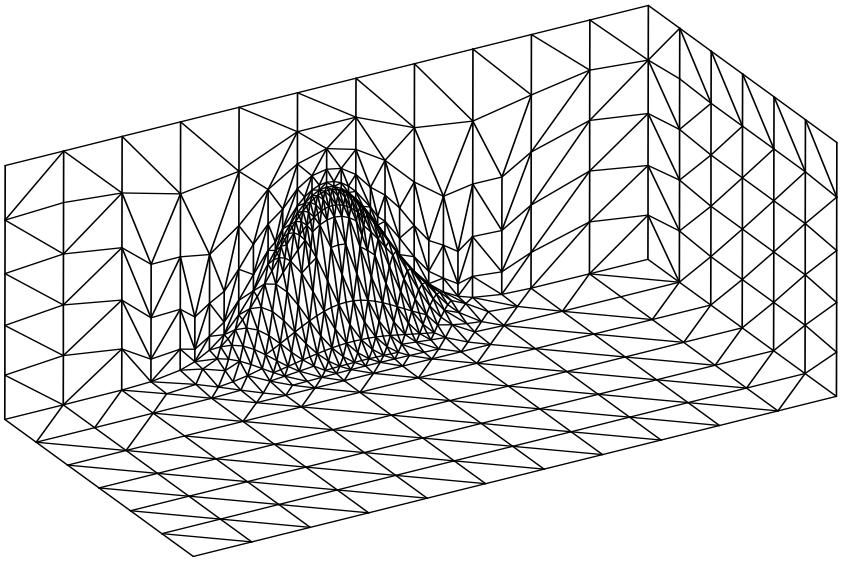


Figura 5. *Malla resultante con la estrategias 2, antes de aplicar el proceso de optimización, para una superficie gaussiana relativamente suave.*

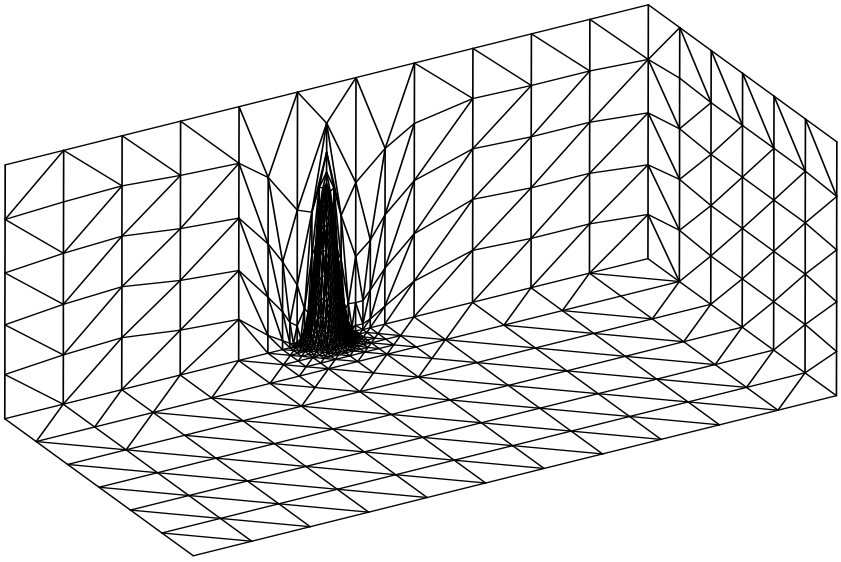


Figura 6. *Malla resultante con la estrategia 2, antes de aplicar el proceso de optimización, para una superficie gaussiana concentrada.*

La segunda aplicación corresponde a una superficie gaussiana muy concentrada en torno al punto de coordenadas $(5000, 5000)$; en la figura 6 se representa un detalle de la malla obtenida con la estrategia 2, con un parámetro de desrefinamiento $\varepsilon = 5$ m. En este caso, se introducen 6 capas, 1237 nodos y no se produce ningún tetraedro invertido. Mencionamos que con la estrategia 1, fijando 8 capas y un valor de $\alpha = 2$, se obtuvo una malla con 1364 nodos y 43 tetraedros invertidos. Para las estrategias 3 y 4 tenemos 228 y 248 tetraedros invertidos, con 989 y 1249 nodos, respectivamente.

La tercera aplicación corresponde a una superficie con forma de *volcán* que fue definida mediante la composición de dos superficies gaussianas; la cota máxima en el dominio es $h = 1$ km, mientras que la cota mínima del terreno es $z_{min} = -1$ km . En la figura 7 se representa un detalle de la malla obtenida con la estrategia 1, con un parámetro de desrefinamiento $\varepsilon = 5$ m. En este caso, se fijó el valor de $\alpha = 2$ y 8 capas. La malla resultante posee 3973 nodos sin existencia de cruces de tetraedros. Por otra parte, comentamos que con la estrategia 2, se obtuvo una malla con 3943 nodos, 6 capas y ningún tetraedro invertido. Para las estrategias 3 y 4 tenemos 430 y 576 tetraedros invertidos, con 3706 y 4013 nodos, respectivamente. Para llevar al límite las posibilidades del generador de mallas propuesto, consideramos una última aplicación con una superficie que presenta paredes prácticamente verticales. En la figura 8 se muestra un detalle de la malla obtenida con la estrategia 1, para $\alpha = 2$, $\varepsilon = 5$ m y 8 capas. La malla posee 1542 nodos y 266 tetraedros invertidos. Para conseguir una mejora en la calidad de la malla se puede desarrollar un procedimiento que genere puntos sobre las paredes verticales de la superficie, aumentar el grado de discretización y optimizar la malla; ninguna de las estrategias consiguió una malla sin tetraedros invertidos antes de su optimización.

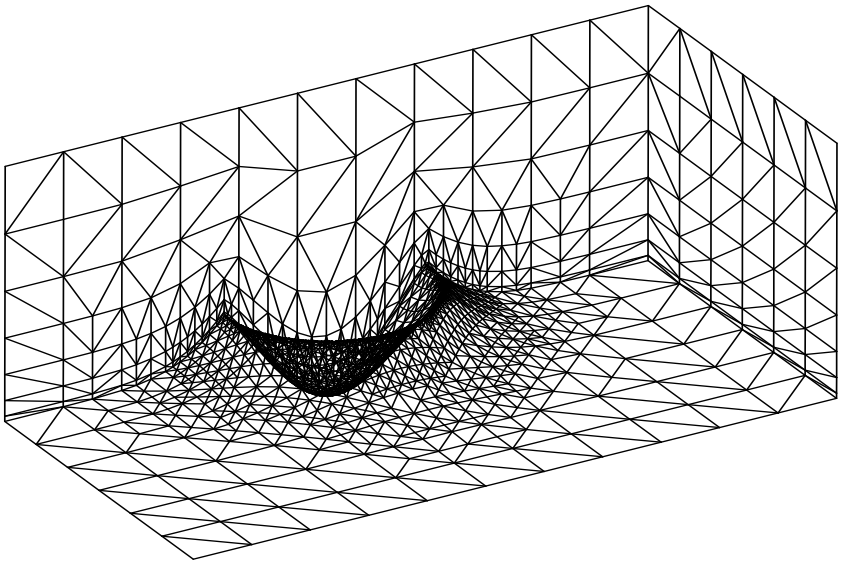


Figura 7. *Malla resultante con la estrategia 1, antes de aplicar el proceso de optimización, para una superficie definida mediante la composición de dos gaussianas.*

Finalmente, destacamos que en todas las aplicaciones analizadas en este apartado, la estrategia 2 fue la que obtuvo una malla con el menor número de tetraedros invertidos antes de aplicar el proceso de optimización.

2.7 Conclusiones

Hemos establecido y analizado los aspectos principales para generar una malla tridimensional de tetraedros que se adapta a una superficie definida sobre una región rectangular con una mínima intervención del usuario. En concreto se ha planteado una generación de puntos, bien distribuidos en el dominio de estudio, capaz de captar la información de la superficie y que posee una densidad menor a medida que aumenta la altura con respecto al terreno. Los puntos se generan aplicando técnicas de refinamiento/desrefinamiento en 2-D y la función de espaciado vertical presentada en este trabajo. Se ha analizado el comportamiento de las cuatro estrategias definidas para construir eficientemente una nube de puntos en el dominio tridimensional. Seguidamente, con la ayuda de un paralelepípedo auxiliar, se ha planteado un procedimiento basado en la triangulación de Delaunay para construir automáticamente la malla, asegurando la conformidad con la superficie del terreno. No obstante, la distribución de puntos obtenida también podría tener interés para generar la malla tridimensional con otras técnicas clásicas, tales como avance frontal [11] y *normal offsetting* [12]. Finalmente, el procedimiento propuesto para optimizar la malla generada ha permitido resolver al mismo tiempo los problemas de cruces de tetraedros y de calidad de la malla.

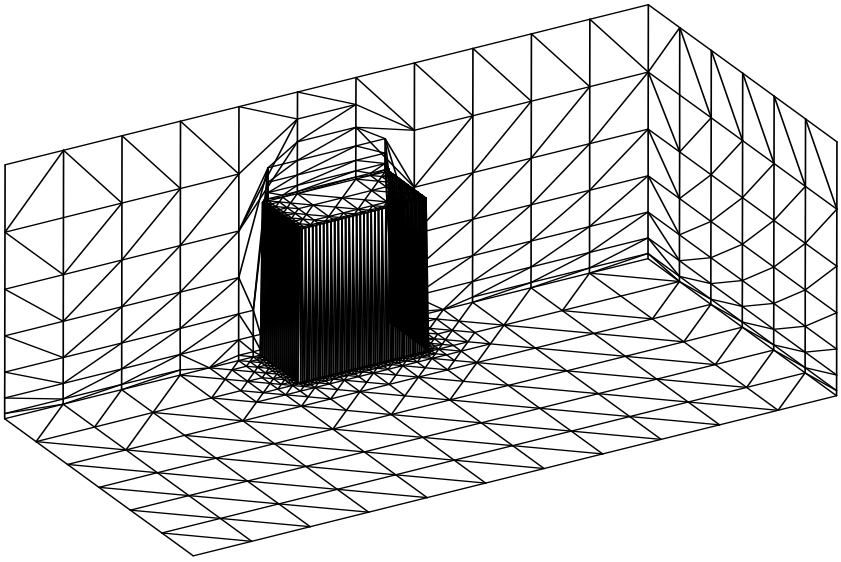


Figura 8. *Malla resultante con la estrategia 1, antes de aplicar el proceso de optimización, para una superficie con paredes verticales.*

3 REFINAMIENTO LOCAL DE MALLAS DE TETRAEDROS

En la actualidad, la mayor parte de los programas que utilizan el método de elementos finitos se apoya en técnicas adaptables basadas en una estimación del error cometido con nuestra solución numérica, o al menos en indicadores de error fiables que nos señalen los elementos que deben ser refinados o desrefinados en la malla.

En generación de mallas adaptables podemos considerar dos aspectos diferentes: la discretización del dominio atendiendo a su geometría o a la solución numérica. Existen muchas formas de abordar estos aspectos. La primera cuestión es: ¿mallas estructuradas o no estructuradas?. En este sentido, está claro que el uso de mallas no estructuradas nos proporciona más flexibilidad a la hora de mallar geometrías complejas utilizando un número óptimo de nodos. En este caso, los métodos más clásicos para la obtención de triangulaciones tridimensionales se basan fundamentalmente en algoritmos de avance frontal [17] o en algoritmos basados en la triangulación de Delaunay [10] y [5]. Una vez que se ha discretizado la geometría del dominio, la malla debe adaptarse atendiendo a las singularidades de la solución numérica. Este proceso implica la introducción (refinamiento) o eliminación (desrefinamiento) de nodos de la malla actual. Los cambios pueden afectar a la malla actual de forma local o global, dependiendo del método de triangulación elegido. Diferentes estrategias de refinamiento han sido desarrolladas para triangulaciones en 2-D, y han sido generalizadas a 3-D. Si se ha optado por un refinamiento que afecte localmente a la malla actual, cabe plantease otra cuestión: ¿mallas encajadas o no encajadas?. La respuesta en este caso no es tan clara. El uso de mallas encajadas tiene varias ventajas importantes. Podemos conseguir familias de secuencias de mallas encajadas en un mínimo tiempo de CPU. Además, se puede aplicar más fácilmente el método multimalla

para resolver el sistema de ecuaciones asociado al problema. Por otra parte, se puede controlar automáticamente la suavidad y la degeneración de la malla, y el mantenimiento de las superficies definidas en el dominio, en función de las características de la malla inicial. Si el dominio posee una geometría compleja, un buen modo de proceder es obtener la malla inicial empleando un generador de mallas no estructuradas y, posteriormente, aplicar una técnica de refinamiento y desrefinamiento local de mallas encajadas atendiendo a un indicador de error apropiado al problema. Además, si tratamos de resolver un problema evolutivo, podemos aproximar automáticamente cualquier solución inicial definida en el dominio. Con la técnica de refinamiento y desrefinamiento conseguimos un óptimo soporte de interpolación a trozos capaz de aproximar esta solución con la precisión deseada. En general, podría aplicarse esta técnica para cualquier función definida en el dominio de forma discreta o analítica.

La elección particular del algoritmo de refinamiento es muy importante, puesto que el algoritmo de desrefinamiento puede entenderse como el inverso del algoritmo de refinamiento. El algoritmo de refinamiento 4-T de Rivara posee buenas propiedades en cuanto a la suavidad y degeneración de la malla. Además de esto, el número de posibilidades que aparecen en la relación entre un elemento padre y sus hijos es menor que con otros algoritmos de refinamiento en 2-D, tras asegurar la conformidad de la malla. Por ejemplo, sería más complicado desarrollar un algoritmo de desrefinamiento, acoplado con el algoritmo de refinamiento local propuesto en [2]; todos los triángulos que deben ser refinados, atendiendo al indicador de error, se dividen en cuatro subtriángulos mediante la introducción de un nuevo nodo en los centros de sus lados y uniéndolos entre sí.

En 3-D, el problema es diferente. Aunque parezca paradógico, la extensión de un algoritmo adaptable que sea más simple que otro en 2-D, no tiene porqué ser también más simple en 3-D. Así, entre los algoritmos de refinamiento desarrollados en 3-D podemos mencionar los que se basan en la bisección del tetraedro [1], [26], [24], y los que utilizan la subdivisión en 8-subtetraedros [3], [15], [16]. En concreto, el algoritmo desarrollado en [24] se puede entender como la generalización a 3-D del algoritmo 4-T de Rivara, que a su vez está basado en la bisección del triángulo por su lado mayor. El problema que se produce en esta extensión a 3-D es el gran número de casos posibles en los que puede quedar dividido un tetraedro, respetando las diferentes posibilidades de la división 4-T en sus cuatro caras, durante el proceso de conformidad de la malla. Sin embargo, los algoritmos analizados en [3], [15], [16], que a su vez generalizan a 3-D la partición en cuatro subtriángulos propuesta en [2], son más sencillos debido a que el número de particiones posibles de un tetraedro es mucho menor que en el caso de la generalización del algoritmo 4-T. Por otra parte, puesto que la calidad de la malla está asegurada en todos estos casos, hemos optado por implementar una versión del algoritmo que utiliza la subdivisión en 8-subtetraedros, y que será presentada en la segunda subsección 3.1 de este trabajo. También en esta subsección se presentan algunos aspectos de la estructura de datos empleada en un código desarrollado en lenguaje C++. En la subsección 3.2 se presentan aplicaciones del algoritmo de refinamiento sobre mallas tridimensionales generadas con el mallador analizado en la sección 2 de este trabajo. Finalmente, se muestran algunas conclusiones y líneas futuras de investigación enmarcadas en la generación de mallas no estructuradas.

3.1 Descripción del algoritmo de refinamiento

En esta subsección presentaremos el algoritmo de refinamiento local que hemos desarrollado a partir de la subdivisión en 8-subtetraedros introducida en [16]. Consideremos una triangulación inicial τ_1 del dominio formada por un conjunto de n_1 tetraedros $t_1^1, t_2^1, ..., t_{n_1}^1$. Nuestro objetivo es construir una secuencia de m niveles de mallas encajadas $T = \{\tau_1 < \tau_2 < ... < \tau_m\}$, tal que el nivel τ_{j+1} se obtiene mediante un refinamiento local del nivel anterior τ_j. Si suponemos conocido el indicador de error ϵ_i^j asociado al elemento $t_i^j \in \tau_j$, decidimos que este elemento debe ser refinado si $\epsilon_i^j \geq \gamma \epsilon_{\max}^j$, siendo $\gamma \in [0, 1]$ el parámetro de refinamiento y $\epsilon_{\max}^j$ el máximo valor de los indicadores de error de los elementos de τ_j. Desde un punto de vista constructivo, plantearemos inicialmente la obtención de τ_2 partiendo de la malla base τ_1, atendiendo a las siguientes consideraciones:

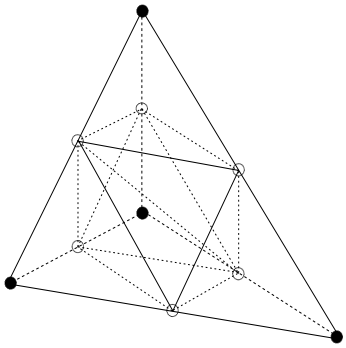
a) *Subdivisión en 8-subtetraedros*. Decimos que $t_i^1 \in \tau_1$ es un tetraedro de *tipo I* si se verifica que $\epsilon_i^1 \geq \gamma \epsilon_{\max}^1$. Este conjunto de tetraedros serán posteriormente subdivididos en 8 subtetraedros según la figura 1(a); se introducen 6 nuevos nodos en el punto medio de sus aristas y se subdividen cada una de sus cuatro caras en cuatro subtriángulos según la división propuesta por Bank [2], cuatro subtetraedros quedan determinados a partir de los cuatro vértices de t_i^1 y las nuevas aristas, y los otros cuatro subtetraedros se obtienen al unir los dos vértices opuestos más cercanos del octoedro que resulta en el interior de t_i^1. Esta sencilla estrategia es la que se propone en [16] y [3], frente a otras basadas en transformaciones afines a un tetraedro de referencia, como la analizada en [15], que aseguran la calidad de los tetraedros resultantes. Bornemann et al. [3] afirma que con ambas estrategias obtiene resultados comparables en sus experimentos numéricos, y Löhner y Baum [16] asegura que nuestra elección produce el menor número de tetraedros distorsionados en la malla refinada. Evidentemente, siempre se podría determinar la mejor de las tres opciones existentes para la subdivisión del octoedro interior, mediante el analisis de la calidad de sus cuatro subtetraedros, pero esto aumentaría el coste computacional del algoritmo.

Una vez que se ha definido el tipo de partición de los tetraedros *tipo I*, de cara a asegurar la conformidad de la malla, nos podemos encontrar con tetredros vecinos que pueden tener 6, 5,..., 1 ó 0 nuevos nodos introducidos en sus aristas. Hay que tener en cuenta que en todo este proceso únicamente estamos marcando las aristas de los tetraedros de τ_1 en las que se ha introducido un nuevo nodo, y en base al número de aristas marcadas se clasifica el correspondiente tetraedro. Es decir, hasta que no está asegurada la conformidad de τ_2, simplemente marcando aristas, no se procederá a la definición de esta nueva malla.

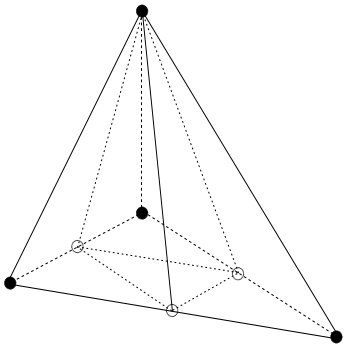
b) *Tetraedros con 6 nuevos nodos*. Aquellos tetraedros que por razones de conformidad tengan marcadas sus 6 aristas pasan automáticamente al conjunto de tetredros *tipo I*.

c) *Tetraedros con 5 nuevos nodos*. Aquellos tetraedros que tengan 5 aristas marcadas pasan también al conjunto de tetraedros *tipo I*. Previamente, habrá que marcar la arista en la que no había sido introducido ningún nuevo nodo.

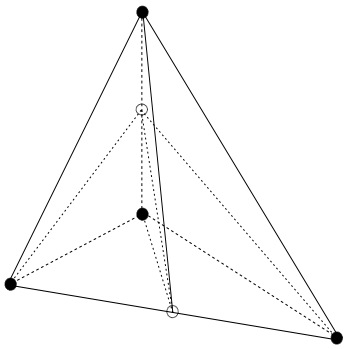
d) *Tetraedros con 4 nuevos nodos*. En este caso, se marcan las dos aristas restantes y pasa a ser considerado de *tipo I*.



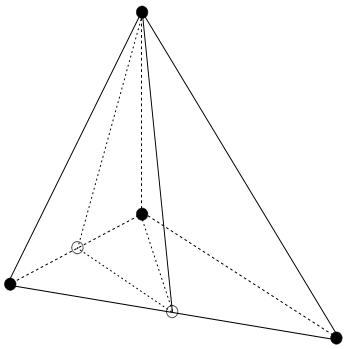
(a) *Tipo I*



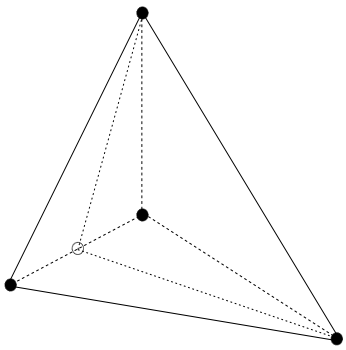
(b) *Tipo II*



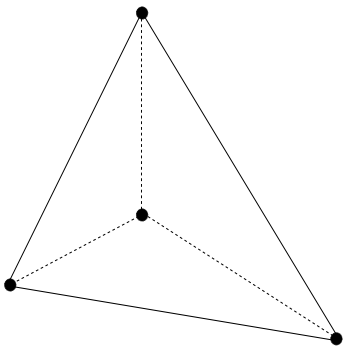
(c) *Tipo III.a*



(d) *Tipo III.b*



(e) *Tipo IV*



(f) *Tipo V*

Figura 9. *Clasificación de las subdivisiones de un tetraedro en función de los nuevos nodos indicados con círculo blanco.*

Hay que tener en cuenta que al proceder de la forma indicada en (b), (c) y (d) debido al refinamiento global considerado en (a) atendiendo al indicador de error, mejoramos la calidad de la malla y simplificamos considerablemente el algoritmo. Puede criticarse que este procedimiento puede aumentar la zona refinada, pero hay que tener en cuenta que sólo se introducirán 1 ó 2 nuevos nodos sobre un total de 6. Obsérvese que esta proporción es igual o inferior a la que surge en el refinamiento en 2-D del algoritmo 4-T de Rivara para un triangulo, en el que la probabilidad de encontrarnos con un nodo introducido en el lado mayor es $1/3$. Este fenómeno se acentúa en el algoritmo propuesto como su generalización a 3-D.

e) *Tetraedros con 3 nuevos nodos*. En este caso, hay que distinguir dos situaciones:

e.1) Si las correspondientes 3 aristas marcadas no están sobre la misma cara, entonces se marcan las 3 restantes y el tetraedro se introduce en el conjunto de tetraedros *tipo I*. Aquí podemos hacer la consideración antes mencionada, al comparar este paso con otros algoritmos basados en bisección por el lado mayor.

En los siguientes casos, ya no marcaremos ninguna nueva arista, lo cual implica que no se introducirá ningún nuevo nodo en el tetraedro que se pretende hacer conforme. Se procederá a subdividirlos de la forma que se indica a continuación, creando subtetraedros que llamaremos *transitorios*, ya que podrán desaparecer en posteriores etapas de refinamiento para asegurar que la malla no degenere.

e.2) Si las 3 aristas marcadas están sobre la misma cara del tetraedro, entonces se crearán 4-subtetradros transitorios como se muestra en la figura 1(b); se definen nuevas aristas uniendo entre sí los tres nuevos nodos y conectando éstos con el vértice opuesto a la cara que los contiene. Los tetraedros de τ_1 con estas características se englobarán en el conjunto de tetraedros de *tipo II*.

f) *Tetraedros con 2 nuevos nodos*. También distinguiremos dos situaciones:

f.1) Si las dos aristas marcadas no están sobre la misma cara, entonces se construirán 4-subtetraedros transitorios como se presenta en la figura 1(c), definidos a partir de las aristas que conectan los dos nuevos nodos y a éstos con los vértices opuestos de las dos caras que los contienen. Los tetraedros que se encuentren en esta situación se denominan de *tipo III.a.*

f.2) Si las dos aristas marcadas están sobre las misma cara, entonces se crearán 3-subtetraedros transitorios según se expone en la figura 1(d); se divide en tres subtriángulos la cara definida por las dos aristas marcadas, conectando el nuevo nodo situado en la arista mayor de éstas dos con el vértice opuesto y con el otro nuevo nodo, tal que estos tres subtriángulos y el vértice opuesto a la cara que los contiene definen los tres nuevos subtetraedros. Se destaca que de las dos posibles eleciones, se toma como referencia la mayor arista marcada para aprovechar en algunos casos las propiedades de la bisección por el lado mayor. Los tetraedros que se encuentren en esta situación se denominan de *tipo III.b.*

g) *Tetraedros con 1 nuevo nodo*. Se crearán dos subtetraedros transitorios según la figura 1(e); se une el nuevo nodo con los otros dos que no pertenecen a la correspondiente arista marcada. Este conjunto de tetraedros se denomina de *tipo IV*.

h) *Tetraedros con 0 nuevos nodos*. Estos tetraedros de τ_1 no se dividen y serán heredados a la malla refinada τ_2. Los denominaremos de *tipo V* y se representan en la figura 1(f).

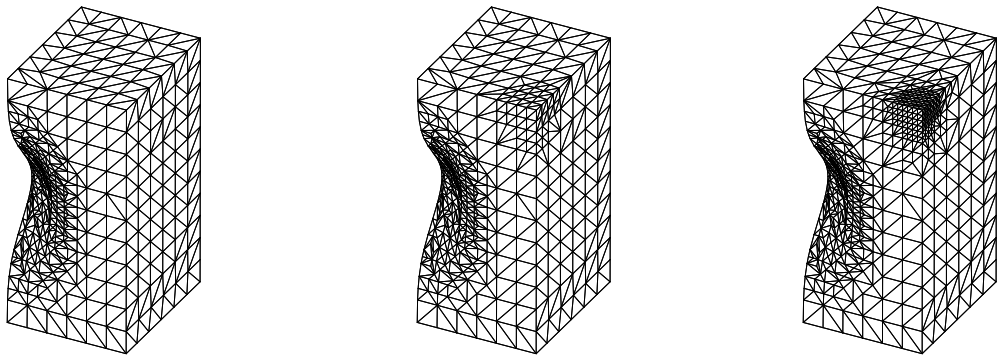
Este proceso de clasificación de los tetraedros de τ_1 se realiza simplemente marcando sus aristas. La conformidad de la malla se va asegurando a nivel local por vecindad entre los tetraedros que contienen una nueva arista marcada, mediante un proceso de expansión que comienza en los tetraedros de *tipo I* definidos en el apartado (a). Esto hace que, al finalizar el recorrido de este subconjunto de tetraedros de *tipo I*, la malla resultante sea conforme y refinada localmente. Además, el proceso resulta de un bajo coste computacional, ya que el proceso de expansión local finaliza cuando nos encontramos con tetraedros en los que no se tiene que marcar ninguna nueva arista.

En general, cuando queremos refinar el nivel τ_j en el que ya existen tetraedros *transitorios* se procederá de igual forma que en el paso de τ_1 a τ_2, con la siguiente variante: desde el momento en que se tenga que marcar una arista de un tetraedro transitorio, bien porque tenga que ser refinado atendiendo al indicador de error o bien por razones de conformidad, entonces se eliminan (proceso de borrado) todos los tetraedros transitorios "hijos" de su tetraedro "padre", marcamos todas las aristas de éste "padre" y se introduce en el conjunto de tetraedros de *tipo I*. Para optimizar el proceso, el bucle en tetraedros que deben ser refinados atendiendo a su indicador de error comienza por los tetraedros transitorios. Tras el proceso de borrado se debe asegurar localmente la conformidad de la malla mediante un proceso de expansión partiendo de las nuevas aristas marcadas del "padre" de los "hijos" transitorios. Hay que destacar que únicamente será necesario definir una variable que distinga si un tetraedro es o no transitorio.

La implementación del algoritmo de refinamiento ha sido desarrollada en lenguaje C++. Se ha elegido éste para aprovechar las facilidades en las gestión dinámica de la memoria y su capacidad como lenguaje orientado a objetos. Cada elemento de una malla (nodos, aristas, caras y tetraedros) ha sido representado mediante una clase, con sus métodos y propiedades. Cada objeto referencia los elementos por los que está compuesto y es referenciado por los elementos que compone. Además se mantienen enlaces entre todos los objetos y los "hijos" generados en cada proceso de refinamiento. El tratamiento de las referencias mediante punteros reduce el consumo de memoria al tiempo que aumenta la velocidad del proceso.

3.2 Aplicaciones

Como ejemplo de aplicación del algoritmo de refinamiento, se presenta en la figura 10(a) una triangulación inicial τ_1 formada por 5072 tetraedros y 1140 nodos, que fue generada a partir del código introducido en [5]. Para proceder a su refinamiento se ha utilizado un indicador de error atendiendo a las distancias desde los centros de gravedad de los tetraedros a un vértice del dominio. En las figuras 10(b) y 10(c) se representan las mallas refinadas después de 1 y 2 etapas de refinamiento, compuestas por 5386 tetraedros y 1201 nodos, y 6270 tetraedros y 1433 nodos, respectivamente. Destacamos la rapidez del proceso de refinamiento, así como la calidad de la malla resultante.



(a) (b) (c)

Figura 10. *Aplicación del algoritmo de refinamiento; (a) malla inicial, (b) y (c) mallas resultantes tras 1 y 2 etapas de refinamiento, respectivamente.*

3.3 Conclusiones

En este apartado se estudia un algoritmo de refinamiento local de mallas de tetraedros basado en la subdivisión en 8-subtetraedros propuesta en [3], [15] y [16]. Se ha aplicado de forma eficiente en mallas tridimensionales generadas mediante una versión del método de triangulación de Delaunay [5]. En trabajos futuros se propone desarrollar el correspondiente algoritmo de desrefinamiento y aplicarlo especialmente en problemas evolutivos de dispersión de contaminantes en la atmósfera.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente subvencionado por el Ministerio de Ciencia y Tecnología y fondos FEDER a través del proyecto REN2001-0925-C03-02/CLI.

REFERENCIAS

[1] D.N. Arnold, A. Mukherjee y L. Pouly, Locally adapted tetrahedral meshes using bisection, *SIAM J. Sci. Comput.*, **22**, 2, 431-448 (2000).

[2] R.E. Bank, A.H. Sherman y A. Weiser, Refinement algorithms and data structures for regular local mesh refinement, in *Scientific Computing IMACS*, Amsterdam, North-Holland, (1983), 3-17.

[3] F. Bornemann, B. Erdmann y R. Kornhuber, Adaptive multilevel methods in three space dimensions", *Int. J. Numer. Meth. Eng.*, **36**, 3187-3203 (1993).

[4] H.N. Djidjev, *Force-directed methods for smoothing unstructured triangular and tetrahedral meshes*, *Tech. Report*, Dep. of Computer Science, Univ. of Warwick, Coventry, UK, (2000). Disponible en http://www.andrew.cmu.edu/user/sowen/topics/new.html.

[5] J.M. Escobar y R. Montenegro, Several aspects of three-dimensional Delaunay triangulation, *Advances in Engineering Software*, **27**, 1/2, 27-39 (1996).

[6] L. Ferragut, R. Montenegro y A. Plaza, Efficient refinement/derefinement algorithm of nested meshes to solve evolution problems, *Comm. Num. Meth. Eng.*, **10**, 403-412 (1994).

[7] L.A. Freitag y P.M. Knupp, *Tetrahedral element shape optimization via the jacobian determinant and condition number*, en *Proceedings of the Eighth International Meshing Roundtable*, Sandia National Laboratories (1996), pp. 247-258.

[8] L.A. Freitag y P. Plassmann, Local optimization-based simplicial mesh untangling and improvement, *Int. J. Numer. Methods Engng.*, **49**, 109-125 (2000).

[9] L.A. Freitag y P.M. Knupp, Tetrahedral mesh improvement via optimization of the element condition number, *Int. J. Numer. Methods Engng.*, **53**, 1377-1391 (2002).

[10] P.L..George, F. Hecht y E. Saltel, Automatic mesh generation with specified boundary, *Comp. Meth. Appl. Mech. Eng.*, **92**, 269-288 (1991).

[11] H. Jin and R.I. Tanner, Generation of unstructured tetrahedral meshes by advancing front technique, *Int. J. Num. Meth. Eng.*, **36**, 1805-1823 (1993).

[12] B.P. Johnston y J.M. Sullivan, Jr., A normal offsetting technique for automatic mesh generation in three dimensions, *Int. J. Num. Meth. Eng.*, **36**, 1717-1734 (1993).

[13] P.M. Knupp, Algebraic mesh quality metrics, *SIAM J. Sci. Comput.*, **23**, 193-218 (2001).

[14] P.M. Knupp, Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part II-A frame work for volume mesh optimization and the condition number of the Jacobian matrix, *Int. J. Numer. Methods Engng.*, **48**, 1165-1185 (2000).

[15] A. Liu y B. Joe, Quality local refinement of tetrahedral meshes based on 8-subtetrahedron subdivision, *Mathematics of Computations*, **65**, 215, 1183-1200 (1996).

[16] R. Löhner y J.D. Baum, Adaptive *h*-refinement on 3D unstructured grids for transient problems, *Int. J. Num. Meth. Fluids*, **14**, 1407-1419 (1992).

[17] R. Löhner y P. Parikh, Three-dimensional grid generation by advancing front method, *Int. J. Num. Meth. Fluids*, **8**, 1135-1149 (1988).

[18] R. Montenegro, A. Plaza, L. Ferragut y I. Asensio, Application of a nonlinear evolution model to fire propagation, *Nonlinear Analysis, Th., Meth.& App.*, **30**, 5, 2873-2882 (1997).

[19] R. Montenegro, G. Montero, J.M. Escobar, E. Rodríguez y J.M. González-Yuste, Tetrahedral mesh generation for environmental problems over complex terrains, *Lecture Notes in Computer Science*, **2329**, 335-344 (2002).

[20] R. Montenegro, G. Montero, J.M. Escobar y E. Rodríguez, Efficient strategies for adaptive 3-D mesh generation over complex orography, *Neural, Parallel & Scientific Computation*, **10**, 1, 57-76 (2002).

[21] G. Montero, R. Montenegro y J.M. Escobar, A 3-D diagnostic model for wind field adjustment, *J. of Wind Eng. and Ind. Aerodynamics*, **74-76**, 249-261 (1998).

[22] M. Murphy, D.M. Mount y C.W. Gable, *A point-placement strategy for conforming Delaunay tetrahedralization*, en *Symposium on Discrete Algorithms*, (2000), pp. 67-74.

[23] A. Plaza, R. Montenegro and L. Ferragut, An improved derefinement algorithm of nested

meshes, *Advances in Engineering Software*, **27**, 1/2, 51-57 (1996).

[24] A. Plaza y G.F. Carey, Local refinement of simplicial grids based on the skeleton, *Appl. Numer. Math.*, **32**, 195-218 (2000).

[25] M.C. Rivara, A grid generator based on 4-triangles conforming. Mesh-refinement algorithms, *Int. J. Num. Meth. Eng.*, **24**, 1343-1354 (1987).

[26] M.C. Rivara y C. Levin, A 3-d refinement algorithm suitable for adaptive multigrid techniques, *J. Comm. Appl. Numer. Meth.*, **8**, 281-290 (1992).

[27] G. Winter, G. Montero, L. Ferragut y R. Montenegro, Adaptive strategies using standard and mixed finite elements for wind field adjustment, *Solar Energy*, **54**, 1, 49-56 (1995).