

IMPLEMENTACIÓN DE METODOLOGÍAS DE MALLA ADAPTATIVA PARA LA OPTIMIZACIÓN DE SIMULACIONES FLUVIOMORFOLÓGICAS EN OPENTELEMAC-MASCARET

IMPLEMENTATION OF ADAPTIVE MESH METHODOLOGIES FOR THE OPTIMIZATION OF FLUVIOMORPHOLOGICAL SIMULATIONS IN OPENTELEMAC-MASCARET

Novara, P.^a, Dominguez Ruben, L.^{b,c}, Franck, G.^a, Scacchi, G.^a y Mendoza, A.^d

^a*Facultad de Ingeniería y Ciencias Hídricas, Universidad Nacional del Litoral, pnovara@unl.edu.ar*

^b*Centro de Estudios Fluviales e Hidro-Ambientales del Litoral (CEFHAL), Facultad de Ingeniería y Ciencias Hídricas, Universidad Nacional del Litoral*

^c*Consejo Nacional de Investigaciones Científicas y Técnicas, CONICET, Buenos Aires, Argentina*

^d*Universidad Nacional Autónoma de México, Instituto de Ingeniería, CDMX, México*

Palabras clave: Malla adaptativa, modelación bidimensional, modelación fluviomorfológica

Resumen. El uso de mallas adaptativas en CFD constituye una de las principales estrategias para optimizar recursos computacionales y reducir considerablemente los costos de simulación. Esta metodología es particularmente valiosa en el estudio de procesos fluviomorfológicos en ambientes naturales, donde la variabilidad de escalas espaciales y temporales representa un desafío constante. La capacidad de capturar fenómenos que evolucionan durante décadas confiere a estas técnicas ventajas tanto cuantitativas como cualitativas en la modelación. Este trabajo describe el diseño y la implementación de una estrategia de refinamiento adaptativo para el sistema openTelemac. Se detalla el criterio de remallado, los algoritmos geométricos utilizados y el funcionamiento general del sistema. La validación se efectúa con un ensayo de canal de laboratorio para el cual se dispone una gran cantidad de datos experimentales que permiten validar el modelo.

Keywords: Adaptive mesh, Two-dimensional modeling, Fluviomorphological modeling

Abstract. Adaptive remeshing in CFD is one of the main strategies for optimizing the use of computational resources and reducing simulation costs. This methodology renders specially valuable when studying fluviomorphological processes in natural environments, where the time and space scales set a constant challenge for the simulation. The ability to capture phenomenons that evolve through decades confers this techniques many qualitative and quantitative advantages. In this paper we describe the design and implementation of an automatic and adaptive mesh refinement algorithm for the software package openTelemac. We present the general system operation, the remeshing criteria and the geometric algorithms applied. A laboratory channel experiment, with abundant empirical data, was used to validate the model.

1. INTRODUCCIÓN

Actualmente existe una amplia variedad de modelos hidrodinámicos bi- y tridimensionales que permiten la simulación de sistemas fluviales de diferentes jerarquías [US Army Corps of Engineers, Hydrologic Engineering Center \(2021\)](#); [Deltares \(2023\)](#); [DHI \(2023\)](#); [Hervouet \(2007\)](#). En ocasiones, la representación física de fenómenos que ocurren en múltiples escalas espaciales requiere el uso de métodos especializados como el refinamiento adaptativo de mallas (AMR, por sus siglas en inglés) [Berger y Oliger \(1984\)](#); [Rettenmaier et al. \(2019\)](#). Esta técnica permite adaptar con precisión una solución dentro de ciertas regiones sensibles o turbulentas de la simulación, de manera dinámica y durante el tiempo de cálculo.

Aunque la metodología AMR no es novedosa, su uso no es común, en parte debido a la variabilidad inherente de las grillas numéricas y la consecuente incertidumbre en la precisión de los resultados. Sin embargo, puede resultar de gran utilidad cuando se requieren resultados rápidos en tiempos razonables o cuando se desea modelar procesos morfológicos en períodos extendidos.

En este contexto, el sistema de cálculo openTelemac [Hervouet \(2007\)](#) constituye un paquete de software libre con implementación estable de elementos finitos y volúmenes finitos, paralelizable y orientado a la simulación de procesos fluvio-morfológicos en sistemas fluviales. Este software está diseñado para el modelado numérico de hidráulica de superficie libre, transporte de sedimentos y propagación de ondas en 1D, 2D o 3D. Desafortunadamente, el modelo no cuenta con un módulo de cálculo AMR en su versión 2D (la de mayor desarrollo), y aunque existen algunos esfuerzos previos orientados a fenómenos particulares [Langendoen et al. \(2016\)](#), estos no tuvieron continuidad debido a su implementación específica.

Este trabajo presenta una implementación de un algoritmo de refinamiento adaptativo automático sobre el modelo openTelemac. El enfoque propuesto introduce un método sencillo y fácil de aplicar ya que no requiere modificar el core de cálculo, no depende de ningún mecanismo específico para la generación de la malla inicial, y tampoco requiere adaptar las definiciones de un caso de estudio para su aplicación. Se presentan los resultados de una implementación preliminar para un caso de estudio de laboratorio desarrollado en la Facultad de Ingeniería y Ciencias Hídricas de la Universidad Nacional del Litoral (FICH-UNL), para su validación. El método integrado al cálculo reduce los costos computacionales y, aunque se obtienen resultados adecuados para equipos workstation de rendimiento medio, el potencial de la implementación radica en su aplicación a procesos morfológicos como los avances deltaicos y la posibilidad de realizar simulaciones de largo plazo de sistemas complejos que anteriormente resultaban computacionalmente prohibitivas.

2. METODOS

Como esquema general de cálculo se estableció el módulo TELEMATAC-2D del modelo openTelemac ([Hervouet \(2007\)](#)), utiliza las hipótesis de Saint-Venant para flujo de aguas poco profundas. El modelo es gratuito y permite el uso de cálculo en paralelo, lo cual agiliza el proceso de cálculo final. El caso físico se refiere a una simulación a fondo fijo llevada a cabo en el laboratorio de Hidráulica de la FICH-UNL, donde se evaluó el patrón hidrodinámico presente debido a la implantación de un estribo. Dicho caso será tomado y evaluado con la metodología de malla adaptativa aquí propuesta.

2.1. Modelo matemático hidrodinámico bidimensional

Las ecuaciones que resuelve TELEMAC-2D para la formulación de aguas poco profundas en la formulación no conservativa

$$\frac{\partial \vartheta h}{\partial t} + \vec{U} \cdot \nabla(\vartheta h) + \vartheta h \nabla \cdot \vec{U} = S_h \quad (1)$$

$$\frac{\partial u}{\partial t} + \vec{U} \cdot \nabla u = -g \frac{\partial H}{\partial x} + S_x + \frac{1}{\vartheta h} \nabla \cdot (\vartheta h \nu_t \nabla u) \quad (2)$$

$$\frac{\partial v}{\partial t} + \vec{U} \cdot \nabla v = -g \frac{\partial H}{\partial y} + S_y + \frac{1}{\vartheta h} \nabla \cdot (\vartheta h \nu_t \nabla v) \quad (3)$$

Donde h es la profundidad, \vec{U} es el vector de velocidades medias, y u y v es la velocidad en cada dirección, H es la superficie de elevación, t es tiempo, ν_t es la viscosidad turbulenta y S_h , S_x y S_y son los términos de fuente o sumideros en la ecuación de conservación y momento.

2.2. Escenarios de modelación

El planteamiento de cálculo corresponde al ensayo de laboratorio llevado a cabo en el Laboratorio de Hidráulica de FICH-UNL [Scacchi \(2017\)](#). De los ensayos efectuados solo se tomó los datos experimentales del Ensayo I (sin estribo y fondo fijo) para realizar la calibración del modelo numérico. Se realizaron relevamientos con velocímetro acústico Doppler ADV Vectrino-Nortek de 10 MHz de frecuencia acústica. El mismo permite contar con los valores de velocidad a diferentes alturas, descomponiendo la variable en las tres componentes espaciales. Aquí se presentarán solo las componentes en el plano x-y (promediadas), conocida como vista en planta del experimento.

Los ensayos de laboratorio se realizaron en un canal de sección rectangular de 60 m de largo, 1.48 m de ancho y 1 m de profundidad, perteneciente al Laboratorio de hidráulica de la FICH-UNL. Este canal contiene un lecho erosionable de 20 m de longitud y 0.4 m de espesor, constituido por arena uniforme de diámetro medio 0.001 m. En el tramo con fondo móvil se representó, junto a la margen izquierda, un estribo de puente prismático, con paredes de vidrio de 0.3 m de longitud y 0.12 m de ancho. El caudal seleccionado fue de aproximadamente 84 l/s, con lo que las condiciones del flujo no perturbado por el estribo estuvieron caracterizadas por una \bar{U} de 32.55 cm/s; u^* de 1.78 cm/s; Re^* de 34.51. Se efectuaron mediciones en la región en inmediaciones a la posición del estribo que alcanzaron el valor de 0.19 m. La condición de salida se determinó para el caso de 0.184 m, con base a los relevamientos llevados a cabo. En la Figura 1, a.1 y a.2 se presentan fotografías tomadas en el canal de ensayo que permite visualizar la zona del estribo y canal e instrumental utilizado, mientras que en 1b se muestra un esquema del dominio geométrico utilizado en este trabajo para la simulación.

Para la calibración del modelo, como se indicó previamente, se usó el ensayo sin presencia de estribo para la determinación de la rugosidad bajo la ley de Nikuradse. El paso de tiempo adoptado inicial fue de 0.03 s, aunque se utilizó un controlador para modificarlo automáticamente dependiendo de la resolución espacial de la malla de modo de obtener un número de Courant de aproximadamente 0.9. Se empleó un modelo de cierre de turbulencia tipo $k - \epsilon$ y un solver de gradiente conjugado, mientras que para la componente advectiva se utilizó la formulación tipo método de las características. El tratamiento del sistema lineal final es del tipo reducido conocido como ecuación de la onda.

Para el computo paralelo se dispone de una PC de alto rendimiento (Workstation) con dos procesadores Intel Xeon E2680v4 (que suman un total 28 núcleos reales) y 64GB de RAM.

2.3. Propuesta metodológica de mallado

El planteo geométrico se dividió en tres pasos progresivos: primero se implementaron dos generadores de mallas ad-hoc muy simples para calibrar el modelo con y sin la presencia del estribo, y luego se diseñó e implementó el software para el refinamiento adaptativo en tiempo de cálculo.

Para la calibración se generó una malla de triángulos estructurada con espaciamiento no uniforme, reduciendo el tamaño de elemento alrededor de las paredes. Esta geometría permitió evaluar los resultados del modelo y determinar el coeficiente de rugosidad adecuado con base a las mediciones de laboratorio. Además, para reducir el dominio en el siguiente paso, se utilizó esta malla para generar las condiciones de borde en la entrada: se impuso en la entrada el caudal conocido y se midió el perfil de velocidad aguas abajo una vez estabilizado. El perfil medido (validado contra los datos experimentales) sirve para imponer a la entrada en las siguientes simulaciones, permitiendo así reducir el margen necesario a la entrada para lograr dicha estabilización.

El segundo paso fue la incorporación a la geometría del estribo. En este caso se implementó un generador de malla ad-hoc parametrizado para poder variar fácilmente ciertos parámetros tales como longitud del canal y ubicación del estribo, tamaño del área a refinar alrededor del estribo, tamaños de elementos deseados en diferentes zonas, velocidad de transición entre zonas, perfil de velocidad impuesto a la entrada, etc. Este generador está basado en Triangle [Shewchuk \(1996\)](#), que utiliza un algoritmo incremental para generar una malla Delaunay restringida [Shewchuk \(2002\)](#) con ciertas garantías de calidad. Si bien se utilizó luego una malla relativamente gruesa generada por este generador como malla inicial para las pruebas de malla adaptativa, esto no es un requisito del sistema, ya que la funcionalidad de remallado no depende del algoritmo de generación de la malla inicial.

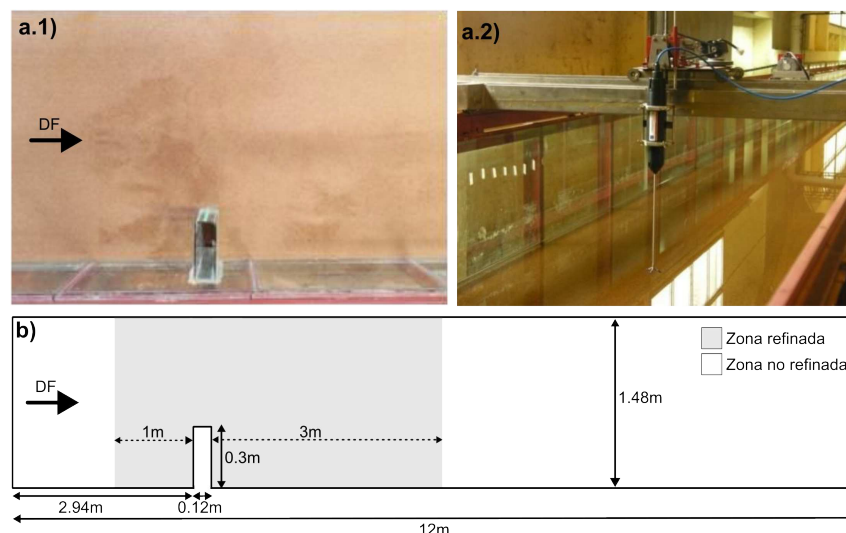


Figura 1: Vista en planta de ensayo con estribo a fondo fijo a.1) geometría de zona de estribo y a.2) equipamiento ADV empleado junto a carro, b) esquema geométrico de la simulación a llevar a cabo en una vista en planta. El ingreso del flujo se hace desde el extremo izquierdo hacia la derecha (DF, dirección de flujo)

El tercer paso fue la implementación del algoritmo de remallado y su interacción con la rutina de cálculo. A nivel software, la estrategia fue utilizar a openTelemac como 'caja negra' para no requerir modificaciones del mismo. El programa que implementa el remallado es el que en realidad controla la simulación: genera los datos de entrada para openTelemac, inicia

su ejecución, y queda en segundo plano analizando periódicamente los archivos temporales del cálculo hasta detectar la necesidad de remallado. En ese punto, interrumpe la ejecución de openTelemac, realiza el remallado, genera un nuevo caso para openTelamac, y vuelve a ejecutarlo. Este proceso se repite hasta que el cálculo finaliza. Para el remallado se utiliza una implementación propia basada en el algoritmo de refinamiento Red-Green. Los detalles de estos procesos se describen en mayor profundidad en la siguiente sección.

Paralelamente al desarrollo del remallado, se efectuó una prueba de convergencia de malla con el modelo ya calibrado. Se partió de una malla inicial producida por el generador ad-hoc, y se refinó el dominio completo 3 veces recursivamente. Se obtuvieron así 4 mallas diferentes, donde cada nuevo refinamiento cuadruplica la cantidad de elementos (ver Tabla 1). Es importante destacar que las mallas refinadas contienen a los nodos de la malla original como subconjunto de los propios, lo cual permite tomar las mediciones para el análisis en nodos coincidentes en todas ellas.

Malla	Calibr.	Conv. 0	Conv. 1	Conv. 2	Conv. 3	Adapt. t=800s
Cantidad de nodos	4000	3475	13097	50785	199937	7501
Cantidad de elementos	5000	6148	24592	98368	393472	14057
Tiempo de cálculo (s)	32	79	240	1920	18799	552

Tabla 1: Propiedades de las mallas utilizadas para la calibración, para el test de convergencia, y la malla final de la prueba con refinamiento automático

2.4. Propuesta metodológica de refinamiento y des-refinamiento

El algoritmo propuesto toma la malla inicial (a $t = 0$ s, que será parte de los datos de entrada del caso de estudio) como malla "base". A partir de ésta se irán generando bajo demanda versiones localmente refinadas. Mientras openTelemac corre, cada vez que guarda el resultado de un paso de tiempo, el algoritmo los detecta y analiza la calidad del mismo para determinar si es necesario remallar.

2.4.1. Métrica de calidad

Se definió una métrica de error por nodo. Si una cierta cantidad de nodos supera un umbral preestablecido se dispara el proceso de remallado. La métrica puede utilizar cualquiera de la variables resultado del cálculo y se basa en comparar en cada nodo el valor calculado con el valor aproximado mediante extrapolación a partir de los elementos vecinos. La extrapolación utiliza (al igual que solver) funciones de forma lineales, por lo que es de esperar que cuando la variable tenga un comportamiento no lineal y el nivel de refinamiento de la malla no sea suficiente como para aproximar correctamente ese comportamiento, el error medido sea alto. Para los ejemplos que aquí se presentarán, la variable de interés seleccionada fue la velocidad. Sería relativamente sencillo modificar la implementación para introducir otras formas de medir la calidad, pero en los resultados se observa que las zonas refinadas por el algoritmo propuesto coinciden con las que un usuario experto decidiría refinar manualmente. Por esto se priorizó avanzar sobre los demás aspectos del algoritmo, y se dejó la exploración de mejores métricas de calidad para trabajos futuros.

En los casos de prueba se definió un tamaño de arista mínimo para limitar el máximo refinamiento posible. Ciertas singularidades geométricas, como ocurre en el vértice del estribo, pueden llevar a que el algoritmo refine una y otra vez alrededor del mismo punto casi como

un bucle infinito. Una situación similar puede darse, dependiendo de las condiciones de borde, cerca de las paredes; y allí este parámetro también ayudaría a limitar el refinamiento en la zona de la capa límite.

2.4.2. Refinamiento

La entrada que recibe el algoritmo de remallado es la lista de nodos marcados como "de baja calidad", y esta marca se transfiere directamente a los elementos de dichos nodos. Una vez determinados los elementos a refinar, se utiliza la estrategia refinamiento red-green [Bank et al. \(1983\)](#). Cada elemento marcado se divide en 4 sub-triángulos generando puntos en el centro de cada arista (estos casos son los denominados rojos). Para evitar hanging-nodes en el resultado final y recuperar la conformidad de la malla, cuando un triángulo fuera del área inicialmente marcada tiene una de sus aristas divididas, se lo divide en dos sin generar ningún otro nodo nuevo ni cambios a sus elementos vecinos (estos son los elementos denominados 'verdes'). El caso en que un triángulo no marcado tiene dos de sus vecinos refinados no está contemplado en principio en ninguno de los dos colores: la alternativa es definir una nueva plantilla de refinamiento para este caso (lo cual da lugar a una variante también conocida como red-green-blue [Carstensen \(2004\)](#)), o convertirlo en rojo y extender el área refinada. Se optó por la segunda alternativa porque ofrece una transición más suave entre el área refinada y área preservada sin cambios, y porque el tercer patrón (los elementos 'azules') puede generar ángulos más pequeños.

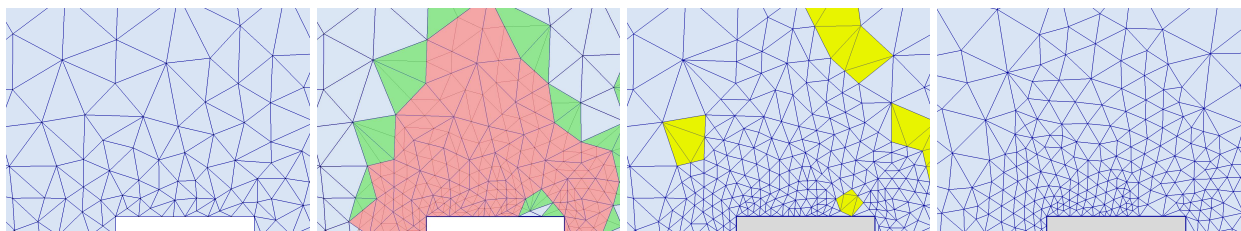


Figura 2: Malla original. Refinamiento red-green. Swaps de diagonales. Suavizado Laplaciano.

Al refinar un elemento rojo, la calidad de los nuevos triángulos es equivalente a la del elemento original. Sin embargo, al refinar un elemento verde, se divide uno de sus ángulos por la mitad y los nuevos elementos pueden tener además una relación de aspecto peor. En sucesivas aplicaciones del algoritmo el problema se agrava si se permite refinar verde sobre verde (refinar como verde un elemento que ha sido generado a partir de un refinamiento verde previo). Además de generarse elementos de mala calidad, se generan nodos con alta valencia (muchos elementos), lo cual también degrada la solución numérica. Para evitar esto, en la implementación propuesta si se requiere refinar como verde un elemento que en una iteración/nivel previo también fue verde, primero se retrocede al nivel anterior, se lo convierte en rojo (extendiendo en un elemento el área refinada previamente), y luego se lo refina (ahora será verde sobre rojo). De esta forma, además de resolver el problema planteado, se logra que la transición desde el tamaño de elemento inicial al refinado luego de varias aplicaciones resulte más suave y progresiva.

Una de las principales ventajas de este método es que respeta los nodos originales de la malla, y también las conectividades. Esto es, que los nodos originales no se eliminan ni cambian de posición (entonces no se alteran los valores nodales calculados), y que las aristas originales tampoco desaparecen (se dividen en 2). Esto, además de simplificar la implementación y con ello reducir el costo computacional, permite pensar guardar un historial de refinamiento de cada

elemento. Por ello, esta alternativa (frente al remallado completo), aunque más restrictiva en algunos aspectos, resulta muy eficiente (aunque por el tamaño de los problemas, en 2D esto no sería un problema grave) y dada su estructura jerárquica permitirá además desrefinar con cierta facilidad.

Se puede pensar en alguna instancia posterior de optimización de las posiciones de los nodos, pero con estrategias simples como aplicar algunos pasos de suavizado Laplaciano [Field \(1988\)](#) la calidad de los elementos no variaba sustancialmente, y sí puede empeorar cerca de los vértices geométricos de la frontera si no se lo restringe adecuadamente. Más aún, aplicarlo obliga a interpolar todos los valores nodales, incluyendo los de los nodos originales. Sin embargo sí se realiza una etapa de optimización de las conectividades que consiste en realizar swaps de diagonales mediante el criterio de Delaunay [Cherfilis y Hermeline \(1990\)](#). Esto mejora localmente algunos elementos con ángulos muy pequeños producto de un refinamiento de tipo verde, aunque rompe la estructura jerárquica de la malla que permitiría luego evitar el refinamiento verde sobre verde, o implementar el desrefinamiento. Por ello, este paso de optimización de conectividades se aplica al guardar la malla generada para openTelemac, pero no se registra en el árbol de elementos con el historial que guarda el algoritmo de remallado. Esta diferencia entre la malla que recuerda.^{el} sistema de remallado y la malla que recibe para openTelemac no es un problema, dado que el remallado solo requiere de parte del cálculo y del algoritmo de medición de calidad que coincidan los nodos.

Se consideró la alternativa de extraer los elementos marcados y generar una nueva malla no estructurada que respete la frontera interna del agujero generado, para luego re-conectarla. Esa estrategia podría dar mayor flexibilidad en el control del tamaño de elemento deseado, la posibilidad de probar con diferentes generadores, y también permitiría refinar o desrefinar con la misma implementación. Sin embargo, problemas como considerar la generación de muchos huecos pequeños disjuntos, la aparición de fronteras arbitrarias e irregulares a respetar, la difusión artificial introducida por la interpolación de valores nodales en los nodos nuevos, hicieron que se optara por el mecanismo basado en red-green.

2.4.3. Interpolación

Para generar los valores nodales (variables resultado de la simulación) para los nodos que se agregan durante el remallado se utiliza interpolación lineal. No hace falta construir la malla Delaunay de base, ni hacer búsqueda lineal para calcular los pesos gracias a que el método red-green siempre genera nodos justo en el centro de las aristas. Por ello, la lista de nodos nuevos, y los índices y pesos a utilizar para las interpolaciones se obtienen sin costo adicional.

Más aún, aprovechando que las simulaciones en 2D son considerablemente menos costosas computacionalmente que las 3D, el algoritmo, al detectar que un paso de tiempo t requiere remallado, retoma el cálculo desde de paso de tiempo previo a t . De esta forma, la interpolación se hace sobre una malla que según las mediciones previas no requería remallado, y por ende, garantiza un cota sobre el posible error de interpolación que en los nodos nuevos. Esta combinación de factores lleva a que se pueda utilizar un método de interpolación tan simple con ciertas garantías, aunque la exploración de mecanismos alternativos (como por ejemplo una interpolación de mayor orden basada en natural-neighbors [Sibson \(1981\)](#) como se utilizó en [Langendoen et al. \(2016\)](#) queda planteada como trabajo futuro.

Además de los valores nodales de las variables calculadas, puede ser necesario interpolar de forma especial las condiciones de borde cuando se agregan nodos en la frontera. Si en una arista de frontera cambia el tipo de condición, al agregar un nodo se elige con un criterio geométrico

una arista vecina para extrapolar las condiciones en lugar de interpolarlas.

2.4.4. Desrefinamiento

Para determinar cuándo o dónde desrefinar, se puede utilizar la misma métrica que para refinar definiendo un umbral de error por debajo del cual el nodo podría eliminarse. Por el mecanismo de refinamiento utilizado, se puede saber exactamente cuáles nodos fueron agregados y de cuáles otros dos nodos "dependen". Pero a diferencia del refinamiento, donde se podría lanzar el proceso aún para refinar alrededor de un solo nodo, aquí se necesita un clúster de nodos a desrefinar para que pueda aplicarse: para poder desrefinar un elemento rojo, se deben poder eliminar los nodos sobre sus 3 aristas, pero en muchos casos también desrefinar los elementos vecinos. Usar red-green-blue simplificaría el problema, pero el swap de diagonales generalmente haría desaparecer la arista del elemento azul sobre la cual se midió calidad para justificar el desrefinamiento.

Estrategias alternativas como el colapsado de aristas [Hoppe \(1996\)](#) se descartaron por su asimetría respecto al método red-green: se insertan nodos/elementos, pero se remueven aristas, entonces el resultado de insertar y luego remover termina siendo de menor calidad que el original (los ángulos se degradan, se pierde la posibilidad de evitar verde sobre verde, y se mueven nodos originales). Además, la jerarquía de elementos del método red-green hace que al desrefinar no se deba interpolar valores nodales preservando la precisión de los valores ya calculados.

Todo esto, sumado a que en el caso de estudio modelado casi no se presenta oportunidad para desrefinar, hace que el desrefinamiento no tenga impacto en los primeros resultados que aquí se presentan.

2.4.5. Implementación

La implementación desarrollada y utilizada para las pruebas consiste en un programa separado de openTelemac, que interactúa con el mismo invocándolo para su ejecución como proceso independiente y monitorizando los archivos temporales que éste va escribiendo durante el cálculo. De esta forma, openTelemac se utiliza como caja negra y no se requiere así ninguna versión en particular ni modificación. Como desventaja, cada remallado implica detener y relanzar proceso, lo cual suele ser más costoso (en tiempo, por la inicialización del cálculo) que el remallado mismo. Por este motivo se busca evitar remallar con frecuencia.

Está desarrollado en C++17 y no depende de ninguna biblioteca externa. Se implementaron versiones propias de los algoritmos geométricos necesarios (refinamiento red-green, swaps de diagonales por Delaunay, mecanismos de interpolación y suavizado Laplaciano) y las funciones para leer y escribir los formatos de archivo involucrados (Serafin, cas, cli). El programa toma por entrada los mismos datos para el caso de estudio que se utilizarían normalmente openTelemac (cas, cli y slf) y un archivo de texto adicional para configurar las opciones propias que no pueden deducirse de los primeros (ej: la cantidad procesadores para la ejecución en paralelo se toman del cas; los umbrales para la métrica de calidad de esta configuración adicional).

El programa genera los archivos cas, cli y slf para cada malla y lanza la ejecución de openTelemac en segundo plano. Mientras openTelemac calcula, este programa monitoriza los archivos temporales (ejemplo: T2DRES*) para detectar cada vez que se escribe un paso de tiempo y hacer el análisis. Si se determina la necesidad de remallar, se detiene el proceso de openTelemac, se procede al remallado, se generan nuevos archivos en entrada (cas, cli y slf), y se lanza

nuevamente openTelemac. En cada ejecución de un caso de estudio, se utiliza un único proceso principal, lo que permite conservar internamente el historial de refinamiento (información que no queda registrada en el archivo .slf). Sin embargo, openTelemac se invoca varias veces, y en cada ejecución calcula un período distinto dentro del intervalo total de simulación. Como resultado se obtendrán varios archivos slf con resultados parciales, cubriendo en cada uno, un intervalo de tiempo diferente dentro del intervalo total a simular, cada uno con una malla diferente. Estos resultados luego pueden fusionarse en un solo SLF con una herramienta adicional desarrollada para tal fin, que extrae/interpola los resultados de todos los archivos parciales sobre una única malla de referencia.

3. RESULTADOS

3.1. Calibración y prueba de convergencia de malla

Como se describió anteriormente, la calibración se realizó con base al experimento sin presencia de estribo en el laboratorio Scacchi (2017). Se evaluaron los resultados para el caso considerando una sección de control de velocidades transversales, sobre la cual se evaluaron diferentes coeficientes de rugosidad, en este caso el de Nikuradse (mejor representación física del grano). En total se evaluaron 5 coeficientes que van desde $k_s = 0.001$, 0.00191, 0.005 y 0.01. La generación de estos 4 valores iniciales se debe principalmente a la configuración del fondo, determinada por el material del lecho con $d_{50} = 1$ mm, y en particular el segundo valor proviene de los cálculos presentados en la bibliografía (Scacchi (2017)).

La Figura 3 presenta los resultados de la calibración a través de la sección transversal 3 a, la dispersión 3 b y el error 3 c. De la misma se desprenden además algunos estadísticos como el RMSE y el sesgo (Bias), los cuales muestran un ajuste adecuado para el segundo coeficiente de rugosidad propuesto ($k_s = 0.00191$). Adicionalmente, cabe destacar que no se impuso rugosidad en las paredes, dado que el material que las conforma es vidrio.

Respecto a la evaluación de convergencia de malla la Figura 4 muestra los resultados de velocidad en el paso de tiempo final para algunos puntos de interés. Se observa una tendencia hacia la convergencia, aunque en algunos puntos todavía se requeriría una iteración más de refinamiento. Esto podría deberse a la fuerte influencia de las paredes del canal en este caso de estudio y el hecho de que el modelo de turbulencia ($k - \epsilon$) no resuelve con precisión la capa límite sobre las mismas. Hacerlo elevaría tanto el número de elementos que anularía el efecto en el tiempo de cálculo del refinamiento sobre el estribo, por lo que se optó por poner una condición de borde deslizante en las mismas (y se validó que con esta configuración se reproducen los resultados experimentales).

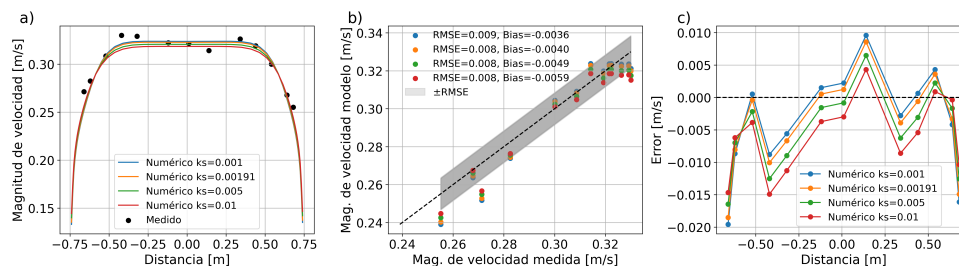


Figura 3: Resultados de la calibración de rugosidad del modelo hidrodinámico. (a) Perfil transversal de velocidades medidas y simuladas; (b) diagrama de dispersión con la banda de desvío ($\pm RMSE$), incluyendo los valores de RMSE y *bias* para cada caso; (c) error puntual en [m/s] entre las mediciones y las simulaciones a lo largo de la sección.

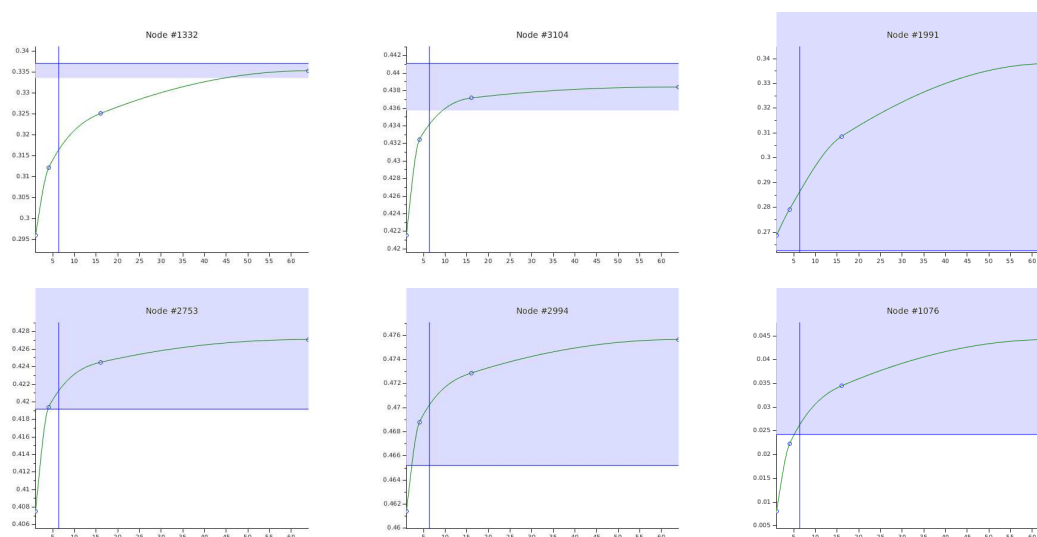


Figura 4: Velocidad (m/s) en puntos de interés para las diferentes mallas. El eje x marca en cuántos elementos se divide cada uno de la malla original durante el test de convergencia. La curva se forma con los resultados de este test. Las líneas horizontales/verticales indican el resultado y el costo equivalente (estimado en base al tiempo de ejecución) para la corrida con remallado adaptativo. 1

3.2. Influencia del remallado adaptativo

Se simuló, aplicando el remallado adaptativo automático, el mismo caso que para la prueba de convergencia de malla, partiendo de la malla 0 de dicho test. El tiempo total de cálculo es aproximadamente el doble que para la malla 1 (Ver Tabla 1), aunque el tamaño de la malla (cantidad de nodos y elementos) es aproximadamente la mitad. Esto se debe al ajuste automático del paso de tiempo que hace el solver para mantener el Courant.

Las líneas horizontales en la Figura 4 señalan el resultado de la simulación con remallado automático, y las verticales el costo computacional equivalente, con el objetivo de contrastar con las versiones de malla fija. La Figura 5 muestra las zonas donde el algoritmo refinó automáticamente. Se observa que en las zonas más refinadas, la versión con remallado logra en algunos lugares una precisión mucho mayor que la malla 2 del test de convergencia pero con un costo computacional mucho menor (ej: nodo #1332). Sin embargo, esta mejora de la precisión no es constante en toda la malla. La Figura 6 muestra resultados a $t = 800s$ y compara los del remallado contra la malla 3 del test de convergencia. La precisión no mejora sustancialmente en las zonas alejadas del estribo, aunque allí la variación de velocidad sea mucho más suave. De hecho en algunos casos el error es comparable al de la malla 0, por lo que la solución allí no sería buena. Se confirma además la fuerte influencia de la cercanía de la pared superior al estribo, dado que la zona de mayor error se ubica sobre esa pared, y no coincide con la zona de mayores velocidades ni de mayor turbulencia.

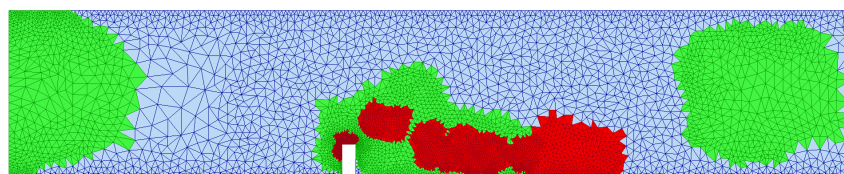


Figura 5: Ejemplo de remallado automático. Malla final para el caso de prueba. En celeste, los elementos de la malla inicial que no se refinaron, en verde los que se refinaron una vez, y en rojo los que se refinaron aun más.

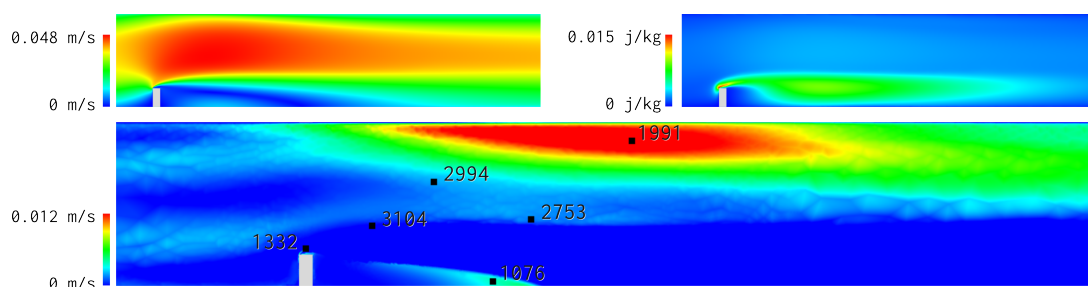


Figura 6: Arriba: velocidad y energía turbulenta a $t = 800$ s para la malla más fina del test de convergencia. Abajo: Diferencia de velocidad entre los resultados de dicha malla y de la malla adaptativa.

4. CONCLUSIONES Y TRABAJOS FUTUROS

Se implementó exitosamente un mecanismo de refinamiento automático para simulaciones con openTelemac que destaca por su simplicidad de uso. La solución no requiere versiones específicas del software ni modificaciones del mismo, utiliza los archivos de entrada estándar y solo necesita unos pocos parámetros adicionales (configurables por el usuario). El tiempo invertido en medir la calidad de la malla, refinarla y regenerar los archivos de entrada es mínimo comparado con el tiempo de cálculo de openTelemac (segundos versus minutos/horas). Sin embargo, interrumpir y relanzar la simulación sí implica un costo computacional significativo que hace prohibitivo el remallado en cada paso temporal. Cabe destacar que cuando se integre el módulo de transporte de sedimento al cálculo, no será necesario realizar modificaciones adicionales a esta implementación.

Los resultados del caso de estudio no permiten extraer conclusiones definitivas sobre la mejora en la relación resultados/costo computacional. Esta limitación probablemente se debe a los efectos de pared previamente discutidos, cuya solución requeriría incorporar variables adicionales a la métrica de calidad, como aquellas relacionadas con la turbulencia (entre otras). Esta línea de investigación queda planteada como trabajo futuro para completar la validación del caso de estudio. No obstante, para los problemas objetivo del remallado automático (ambientes fluviales), estos efectos de capa límite en las fronteras no tendrán influencia, aunque sí resultará relevante contemplar otras variables como la evolución del fondo.

El desarrollo alcanzó satisfactoriamente dos objetivos principales: la resolución del problema geométrico (selección metodológica, implementación del refinamiento, interpolación de valores nodales y condiciones de borde, estructuras de datos eficientes) y la integración con openTelemac (lectura y procesamiento de archivos de entrada, extracción directa de resultados parciales desde archivos temporales, generación automática de archivos para continuidad del cálculo).

REFERENCIAS

- Bank R.E., Sherman A.H., y Weiser A. Some Refinement Algorithms and Data Structures for Regular Local Mesh Refinement. En R.S. Stepleman, editor, *Scientific Computing*, páginas 3–17. North-Holland, Amsterdam, 1983.
- Berger M.J. y Oliger J. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53(3):484–512, 1984. [http://doi.org/10.1016/0021-9991\(84\)90073-1](http://doi.org/10.1016/0021-9991(84)90073-1).
- Carstensen C. An adaptive mesh-refining algorithm allowing for an h^1 stable l^2 projection onto courant finite element spaces. *Constructive Approximation - CONSTR APPROX*, 20:549–564, 2004. <http://doi.org/10.1007/s00365-003-0550-5>.

- Cherfils C. y Hermeline F. Diagonal swap procedures and characterizations of 2D-Delaunay triangulations. *ESAIM: Modélisation mathématique et analyse numérique*, 24(5):613–625, 1990.
- Deltares. *Delft3D-FLOW User Manual*. Deltares, Delft, The Netherlands, 2023. Version 6.03.
- DHI. *MIKE 21 Flow Model FM, Hydrodynamic Module, Scientific Documentation*. DHI Water & Environment, Hørsholm, Denmark, 2023. MIKE Powered by DHI.
- Field D.A. Laplacian smoothing and delaunay triangulations. *Communications in Applied Numerical Methods*, 4(6):709–712, 1988. <http://doi.org/https://doi.org/10.1002/cnm.1630040603>.
- Hervouet M. *Front Matter*. John Wiley & Sons, Ltd, 2007. ISBN 9780470319628. <http://doi.org/https://doi.org/10.1002/9780470319628.fmatter>.
- Hoppe H. Progressive meshes. En *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, página 99–108. Association for Computing Machinery, New York, NY, USA, 1996. ISBN 0897917464. <http://doi.org/10.1145/237170.237216>.
- Langendoen E.J., Mendoza A., Abad J.D., Tassi P., Wang D., Ata R., El kadi Abderrezak K., y Hervouet J.M. Improved numerical modeling of morphodynamics of rivers with steep banks. *Advances in Water Resources*, 93:4–14, 2016. ISSN 0309-1708. <http://doi.org/https://doi.org/10.1016/j.advwatres.2015.04.002>. Numerical modelling of river morphodynamics.
- Rettenmaier D., Deising D., Ouedraogo Y., Gjonaj E., De Gersem H., Bothe D., Tropea C., y Marschall H. Load balanced 2d and 3d adaptive mesh refinement in openfoam. *SoftwareX*, 10:100317, 2019. ISSN 2352-7110. <http://doi.org/https://doi.org/10.1016/j.softx.2019.100317>.
- Scacchi G. *Interacción entre el flujo turbulento y los procesos de erosión junto a estribos de puentes protegidos. Estudio experimental*. Tesis de Doctorado, Universidad Nacional del Litoral, Santa Fe, Argentina, 2017.
- Shewchuk J.R. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. En M.C. Lin y D. Manocha, editores, *Applied Computational Geometry: Towards Geometric Engineering*, volumen 1148 de *Lecture Notes in Computer Science*, páginas 203–222. Springer-Verlag, 1996. From the First ACM Workshop on Applied Computational Geometry.
- Shewchuk J.R. Delaunay refinement algorithms for triangular mesh generation. *Comput. Geom. Theory Appl.*, 22(1–3):21–74, 2002. ISSN 0925-7721. [http://doi.org/10.1016/S0925-7721\(01\)00047-5](http://doi.org/10.1016/S0925-7721(01)00047-5).
- Sibson R. A brief description of natural neighbour interpolation. *Interpreting Multivariate Data*, páginas 21–36, 1981.
- US Army Corps of Engineers, Hydrologic Engineering Center. *HEC-RAS River Analysis System, User's Manual*. U.S. Army Corps of Engineers, Davis, CA, USA, 2021. Version 6.0.