Escalabilidad de modelos numéricos del Río de la Plata

Pablo Ezzatti
Centro de Cálculo - Instituto de Computación
Facultad de Ingeniería - Universidad de la República
Montevideo - Uruguay
pezzatti@fing.edu.uy

Abstract

Uno de los modelos más utilizados para la simulación de mareas es el RMA-10 desarrollado por I. P. King (1993) y sus colaboradores. Entre las principales características se puede destacar sus cualidades de descripción, su capacidad de trabajar con grilla irregular y con distintos tipos de elementos y el tratamiento que realiza de la viscosidad turbulenta. Sin embargo en muchos casos, principalmente cuando se trabaja en 3 dimensiones, los excesivos costos computacionales, tiempo de cálculo, del modelo los transforman en una elección prohibitiva. Por estas razones, se diseñó y desarrolló una modificación del modelo centrada en la etapa de resolución de los sistemas lineales. Esta modificación se presentó en trabajos anteriores.Este trabajo se centra en evaluar la modificación propuesta anteriormente, estudiando la influencia de las estrategias de ordenamiento de pivotes en el tiempo de cómputo y la capacidad de escalamiento de la propuesta ante el aumento del tamaño de los problemas.

Keywords: Palabras Claves: Computación de alto desempeño, métodos numéricos, RMA-10.

1. INTRODUCCIÓN

En las últimas décadas, la simulación computacional de fluidos ha emergido como un área de intenso trabajo. En Uruguay, en la Facultad de Ingeniería (FIng) de la Universidad de la República, desde hace algunos años, se está trabajando en el modelado numérico del Río de la Plata. En un principio se construyeron modelos basados en la utilización de esquemas de diferencias finitas. Con el fin de obtener mejoras en diversas características del modelado, se realizó el pasaje del paradigma de diferencias finitas al de elementos finitos. A tales efectos se optó por el modelo RMA-10, ampliamente utilizado en el modelado de estuarios.

Las cualidades de descripción del modelo, su capacidad de trabajar con grilla irregular y con distintos tipos de elementos así como el tratamiento que realiza de la viscosidad turbulenta, permiten la obtención de resultados sumamente alentadores en diversas simulaciones realizadas. En contraposición a la mejora del modelado, se incrementan de forma abrupta los costos computacionales (tiempo de procesamiento), lo que implica un gran obstáculo para su utilización y para aplicar mejoras al modelo (mejorar la precisión y expandir eficientemente el modelo a tres dimensiones). Teniendo en cuenta la problemática anterior en trabajos previos se desarrolló una variante del modelo utilizando técnicas de alto desempeño [6].

Este trabajo se centra en evaluar la propuesta realizada anteriormente. Se presenta un breve estudio del efecto de usar distinas estrategias de ordenamiento, así como la evaluación de la escalabilidad ante el aumento de tamaño de los problemas a resolver de la propuesta de modificación del modelo RMA-10 presentada anteriormente.

Lo que resta de este documento se estructura de la forma que se describe a continuación. Primero se presenta una introducción al modelo RMA-10, luego se ofrece un resumen de la propuesta de modificación realizada anteriormente, que abarca la descripción de los casos de prueba, el estudio de los tiempos de ejecución del modelo original y el diseño de la propuesta. A continuación se describen los estudios realizados y los resultados obtenidos. Por último, se presentan las conclusiones y trabajos futuros planteados.

2. EL MODELO NUMÉRICO

El modelo RMA-10 [10], fue presentado en el año 1993 por I. King, para realizar modelados y simulaciones numéricas en tres dimensiones de la circulación hidrodinámica. Diseñado en particular para trabajar en situaciones donde las magnitudes de las velocidades verticales son importantes y la estratificación por densidad se presenta como un factor significativo.

El modelo describe las variables de estado, presión y velocidad (en tres dimensiones), utilizando la formulación de Galerkin [7] de elementos finitos, para resolver numéricamente las ecuaciones implicadas, que básicamente derivan de la combinación de las ecuaciones de Navier-Stokes, continuidad másica, advección-difusión, y una ecuación de estado que relaciona la densidad con la salinidad, temperatura y sedimento suspendido. En el modelo las fuerzas de fricción, el efecto de Coriolis y la tensión del viento en la superficie también son representadas.

El modelo RMA-10 puede ser utilizado para la simulación de situaciones que poseen una dependencia del tiempo, así como aplicado a la resolución de problemas estacionarios.

Además de ser utilizado en FIng para realizar diversos estudios preliminares del transporte de sedimentos en el Río de la Plata, el modelo RMA-10 ha sido aplicado con éxito en varios escenarios. Entre otras se destacan las aplicaciones realizadas en la Bahía de San Franciso, en el puerto de New York, en la Bahía Galveston y en el Río St. Lawrence [14].

Cuando se trabaja con el modelo en su versión tridimensional es necesario especificar en los archivos de entrada la cantidad de capas con las que se quiere realizar la simulación numérica [11]. Esta cantidad se puede definir uniformemente para todo el dominio de resolución o especificar la cantidad de capas para cada uno de los elementos definidos en el plano horizontal.

Las condiciones de borde se pueden especificar en términos de descarga, de velocidad o de nivel del agua. En caso de ser necesario, se puede utilizar una estructura vertical de la densidad.

Otro aspecto relevante del modelo RMA-10 es el tratamiento que realiza de la viscosidad turbulenta horizontal. Mientras que la mayoría de los modelos bidimensionales integrados en profundidad trabajan con un único valor del coeficiente de viscosidad turbulenta, el modelo RMA-10 permite la utilización de diferentes estrategias. Entre las opciones, la más simple es trabajar con un único valor del coeficiente de viscosidad turbulenta, el cual es constante. Una segunda opción es trabajar con el modelo de viscosidad propuesto por Smagorinsky.

La implementación del modelo RMA-10 se compone de diversos módulos codificados en Fortran. El Algoritmo 1 presenta un pseudo-código simplificado del modelo.

La primera etapa del Algoritmo 1 (paso 1) corresponde a una componente de inicialización del modelo. En esta etapa se obtienen la grilla de cálculo y los parámetros globales, como son la cantidad máxima de pasos y el criterio de convergencia del método de resolución de sistemas no lineales, la cantidad de pasos de tiempo que se desea simular, etc.

Algoritmo 1 Pseudo-código del RMA-10.

```
1:  Inicialización
2:  for 1:cantPasosSimulación do
3:      Calcular estructuras auxiliares
4:      for 1 : cantPasosNewton-Raphson do
5:          Resolver sistema lineal para calcular aporte
6:          Calcular velocidades verticales
7:          Desafectar ecuaciones que cumplen condición de convergencia
8:      end for
9:      Modificar las condiciones iniciales
10:     Registrar resultados
11: end for
12: Finalización
```

Luego, como se puede observar en el Algoritmo 1 entre los pasos 2 y 11, el programa se compone de un ciclo. En cada iteración del ciclo se efectúan los cálculos necesarios para realizar la simulación de un paso de tiempo. Las ecuaciones del planteo del método de elementos finitos (MEF) del modelo se componen principalmente de un sistema de ecuaciones no lineales. Como se puede apreciar en los pasos 4 al 8 del Algoritmo 1, para la resolución de estos sistemas de ecuaciones no lineales se utiliza la estrategia de Newton-Raphson.

Debido al importante costo computacional que implica la resolución de grandes sistemas lineales, por lo general es la etapa de los MEF más importante en cuanto a tiempo de cómputo, el modelo RMA-10 utiliza la estrategia de desafectar de la iteración de Newton-Raphson en pasos intermedios, aquellas ecuaciones que cumplan el criterio numérico de convergencia, especificado en el Algoritmo 1 por el paso 7. Por esta razón, el tamaño de las matrices a resolver durante la iteración normalmente va disminuyendo a partir del segundo paso del ciclo.

En las distintas aplicaciones dinámicas de MEF, es necesario realizar en cada paso de tiempo (o en cada paso de la iteración de Newton-Raphson en problemas no lineales) el cálculo de los valores de los nodos de los distintos elementos de la malla. Estos valores se utilizan para la generación, luego de una etapa de ensamblado, de la matriz de rigidez. En el modelo RMA-10 se dispone de la rutina **`FRONT`** (y varias sub-rutinas) para realizar estas tareas que se corresponden al paso 5 del Algoritmo 1.

Para efectivizar el cálculo de los coeficientes se utilizan distintas sub-rutinas. En tiempo de ejecución y dependiendo del tipo/forma del elemento que se esté procesando, se determina cual sub-rutina utilizar. Para el cálculo de los valores de los nodos es necesario efectuar integrales. Este cálculo se realiza utilizando una cuadratura de Gauss.

La estrategia de resolución de los sistemas lineales utilizada es el método frontal, presentada por Irons en 1970 [9] y extendida para matrices no simétricas por Hood en 1976 [8]. Los métodos frontales son versiones de la eliminación Gaussiana, relacionadas directamente con los elementos finitos y diseñadas en un principio para disminuir las necesidades de memoria de las estrategias estándar de resolución. Su estrategia de trabajo que se basa en ir cargando a memoria la matriz de a trozos (los distintos frentes), permitió a las distintas implementaciones de los métodos resolver grandes sistemas lineales en el hardware disponible en los años 70.

Luego, en el paso 6 del Algoritmo 1, se calculan las velocidades verticales.

Por último, el paso 12 del Algoritmo 1, corresponde a una etapa de finalización de la aplicación. Algunas de las tareas que se destacan en esta etapa son el registro de resultados finales, la liberación de recursos computacionales y la clausura de archivos.

3. TRABAJO PREVIO

El trabajo previo consistió en definir casos de prueba, evaluar los tiempos de ejecución del modelo original y diseñar una modificación del modelo RMA-10 que mejore el desempeño computacional. En el trabajo de Ezzatti y Piedra-Cueva [6] se puede profundizar en la propuesta.

3.1. Casos de prueba

Se utilizan 2 casos de prueba que difieren en la cantidad de elementos, nodos y capas con las que se discretiza el dominio a simular, así como en las condiciones de borde impuestas.

El caso N-31K se define por 1630 elementos y 5113 nodos en el plano, con cantidad de capas variables entre los distintos elementos, como máximo se utilizan 3 capas. Estos valores implican un máximo de 30787 ecuaciones en los sistemas lineales. Este caso corresponde al modelado real del Río de la Plata que se cuenta en FIng en la actualidad. Utilizando esta discretización se han realizado estudios en los cuales se validaron resultados empíricamente [7].

El caso N-40K utiliza para la descripción del dominio 4578 elementos y 9950 nodos en un plano, teniendo un número variable de capas para cada uno de los elementos. La cantidad de ecuaciones de las matrices de los primeros pasos de las iteraciones de Newton-Raphson es 39759. En ocasiones se utiliza una modificación de este caso que emplea cantidad de capas uniforme en el dominio, lo que implica sistemas lineales de mayor tamaño.

3.2. Evaluación de tiempos del modelo RMA-10

Para realizar esta etapa del trabajo se utilizó el caso de prueba N-31K, realizando el total de las ejecuciones en un computador Dell, con dos procesadores Pentium IV de 1.8 GHz, que dispone de 1 GB RAM, sistema operativo Linux Red Hat 7.3 y compilador Fortran Intel 7.0.

En el Cuadro 1 se resumen los tiempos de ejecución evaluados para las distintas etapas del modelo RMA-10 especificadas en el Algoritmo 1.

Etapa	Tiempo (seg)
Inicialización.	0.59
Calcular estructuras auxiliares.	0.02
Resolver sistema lineal para calcular aporte.	35.57
Resolver velocidad vertical.	0.55
Desafectar ecuaciones que cumplen condición de convergencia.	0.10
Modificar las condiciones iniciales y registrar resultados.	0.06

Cuadro 1: Tiempos de ejecución de las distintas etapas del modelo RMA-10.

Considerando los resultados resumidos en el Cuadro 1, se puede concluir que la mayor fuente de consumo de tiempo de cómputo es la etapa de cálculo y resolución de las matrices de rigidez. Esta etapa se realiza en cada paso de la iteración del método Newton-Raphson y es efectuada por la rutina **`FRONT`** del modelo RMA-10.

Debido a que la rutina **`FRONT`** resultó ser la más costosa, se realizó una segunda instancia de estudio orientada a evaluar los tiempos de cómputo de las distintas etapas de la rutina **`FRONT`**.

Como se mencionó anteriormente, la rutina **`FRONT`** utiliza el método frontal para la resolución de los sistemas lineales. El método no necesita realizar el cálculo de todos los coeficientes de la matriz de rigidez para comenzar a realizar la factorización, sino que su modalidad de trabajo es ir evaluando los distintos elementos y factorizar los distintos frentes. Sin embargo, para intentar identificar las zonas criticas del código, se delimitaron y calcularon los tiempos de los tres fragmentos lógicos que a priori posee la rutina: el cálculo de los coeficientes, la factorización de los distintos frentes de la matriz dispersa y la resolución.

Para calcular los tiempos de las etapas lógicas de la rutina **`FRONT`** se efectuó la sumatoria de los tiempos de las partes que las componen. Por ejemplo, el tiempo de factorización se calculó como la sumatoria de los tiempos de ejecución para factorizar cada uno de los frentes.

En el Cuadro 2 se puede observar la relación de tiempos de cómputo de las tres partes identificadas de la rutina **`FRONT`**. Los tiempos presentados son el promedio de 10 ejecuciones.

Etapa	Tiempo (seg)	Desv. Est.
Cálculo de coeficientes	2.88	0.11
Factorización	32.14	0.16
Sustitución hacia atrás	0.55	0.01
Total	35.57	0.19

Cuadro 2: Tiempos de ejecución de las distintas etapas lógicas de la rutina **`FRONT`**.

En base a los tiempos calculados se puede concluir que la factorización es en forma categórica la etapa más costosa de la rutina **FRONT**.

3.3. Propuesta realizada

La propuesta consiste en cambiar la metodología de trabajo que utiliza la rutina **FRONT**. Estableciendo una etapa de generación de la matriz de rigidez, luego utilizar un algoritmo para factorizar y resolver los sistemas lineales. Debido a las importantes dimensiones de las matrices de rigidez, para su manejo, hay que utilizar estrategias de almacenamiento disperso.

La lógica e implementación de la nueva rutina **FRONT** se puede dividir (a diferencia de la rutina **FRONT** original) en tres fases independientes, que se describen a continuación.

- Cálculo de los coeficientes: calcula todos los coeficientes de la matriz aportados por los distintos elementos. Los valores calculados se cargan en una estructura auxiliar.
- Generación de la matriz: en base a los resultados de los cálculos de la etapa previa, filtrando y reordenando distintos coeficientes se genera la matriz dispersa en algún formato.
- Resolución del sistema: se realiza la factorización y resolución del sistema lineal.

Cálculo de los coeficientes

La primera etapa de cálculo de los coeficientes se compone de una iteración sobre los elementos de la malla, en la cual para cada elemento se calcula el aporte a los coeficientes de la matriz de cada uno de los nodos para cada una de las variables libres. Para esto se utilizan las rutinas originales del sistema, que calculan los coeficientes y los almacenan en una estructura local que como máximo posee 80 valores (20 nodos por elemento y 4 variables libres).

A continuación se almacenan los valores anteriormente calculados en una estructura auxiliar de tipo tabla de dispersión abierta [2]. Esta estructura se utiliza para almacenar globalmente toda la matriz. La estrategia de las tablas de dispersión (hash) se basa en dividir el conjunto de datos a almacenar en una cantidad especifica de clases (cubetas). Para esta tarea es necesario la definición de una función de dispersión, la cual para cada objeto del conjuto de datos devuelve una única clase. Al utilizar hash abiertos, cada cubeta dispone de una lista de elementos.

En esta propuesta la función de dispersión utilizada queda definida por la fila del coeficiente a almacenar ($f_{dispersión}(coef_{ij}) = i$) y la lista de las cubetas se compone de varios nodos en donde cada uno de ellos es una dupla en el que se dispone del indicador de columna y el valor.

Para cargar los datos se aplica la función de dispersión a cada coeficiente (por ejemplo $coef_{ij}$), y una vez obtenida la cubeta (la i en el ejemplo) de la tabla de dispersión, se recorre la lista que tiene asociada la cubeta buscando si ya existe alguna entrada cuyo valor de columna sea igual al de la columna del coeficiente a almacenar (j en el ejemplo), en caso afirmativo se suma al valor de la entrada el valor del coeficiente calculado, en caso contrario se agrega un nuevo nodo al final de la lista con el valor e indicador de columna correspondiente.

Al finalizar esta etapa se cuenta con todos los coeficientes de la matriz en un formato disperso simple. Sin embargo, pueden existir entradas que posean valores nulos.

Generación de la matriz

La segunda etapa de la rutina propuesta consiste en llevar a cabo diversas tareas de depuración, que se realizan para poder invocar al método de resolución. Entre otras tareas se pueden filtrar los coeficientes nulos, ordenar los coeficientes y/o generar las estructuras auxiliares de forma de disponer de la matriz en el formato necesario para la siguiente etapa.

Si bien conceptualmente esta etapa es adaptable para cualquier solver, se desarrolló una versión particular para la biblioteca utilizada. En este caso se filtran los coeficientes nulos y no se establece ningún orden.

Resolución

La última etapa se encarga de la factorización y resolución del sistema lineales. Para esta tarea se utiliza la versión serial de la biblioteca de MUMPS 4.3 [5], biblioteca que utiliza los métodos multifrontales. La biblioteca MUMPS necesita disponer de la biblioteca BLAS, en el trabajo se utilizó la implementación de Goto disponible en su sitio web [1].

Resultados

Las matrices involucradas en la resolución del modelo RMA-10 aplicado al Río de la Plata son altamente mal condicionadas, generando que pequeñas perturbaciones en los cálculos puedan implicar grandes cambios en los resultados numéricos en pocos pasos de simulación. En este sentido, y con el objetivo de realizar la validación desde el punto de vista numérico, se comparó el resultado de efectuar una simulación de importancia media de 576 pasos de tiempo equivalente a 192 horas (8 días) de simulación sobre cinco puntos distribuidos en distintas zonas de la malla para el caso N-31K.

Nodo	Versión	Vel. Total	Elevación	Salinidad
Nodo 1519	Original	0.111316	0.784859	33.9156
Nodo 1519	Propuesto	0.111293	0.784851	33.9156
Nodo 2421	Original	-0.395645	0.885763	33.6970
Nodo 2421	Propuesto	-0.395656	0.885757	33.6970
Nodo 3021	Original	0.365118	1.19102	23.4713
Nodo 3021	Propuesto	0.365115	1.19102	23.4713
Nodo 3215	Original	-1.09168	0.888560	22.4547
Nodo 3215	Propuesto	-1.09166	0.888556	22.4547
Nodo 4226	Original	0.567655	2.08858	0.179923E-04
Nodo 4226	Propuesto	0.567654	2.08858	0.179923E-04

Cuadro 3: Resultados numéricos de la versión original y propuesta.

Los resultados obtenidos se pueden observar en el Cuadro 3, que presenta valores de velocidad total, elevación y salinidad. El estudio de la información que brinda la tabla permite afirmar que ambas versiones obtuvieron resultados sumamente similares, ya que las diferencias en los resultados numéricos finales en ninguna ocasión superan el 0,01 %.

En cuanto a la evaluación de tiempos de cómputo, el estudio se centró en la comparación de los tiempos de ejecución del modelo original y el resultante de las variaciones propuestas por un lado y en la comparación de los tiempos de las rutinas **FRONT** original y la propuesta.

En el Cuadro 4 se presentan los tiempos de ejecución de realizar una simulación de 576 pasos de tiempo utilizando el modelo en la versión original y la propuesta. Se puede observar que el tiempo de ejecución de la versión propuesta es casi una cuarta parte de la versión original.

Versión	Tiempo (minutos)
Original	1578,6
Propuesta	450,7

Cuadro 4: Comparativo de tiempos de la versión original y la propuesta.

En cuanto a los tiempos que insume la nueva rutina **FRONT** en el Cuadro 5 se presentan los tiempos de ejecución promedio de cada una de las tres etapas especificadas en la sección de diseño de la versión propuesta, así como el tiempo de ejecución de la rutina en su totalidad.

Estudiando los resultados del Cuadro 5 y comparándolos con los tiempos de ejecución de la rutina original se pude observar que la rutina **FRONT** original implica aproximadamente cuatro veces el tiempo que insume la rutina propuesta. Además, si comparamos el tiempo de factorización de la rutina original con la suma de los tiempos de cargar la estructura auxiliar, el reordenamiento, más el tiempo de factorización y resolución del sistema lineal de la rutina **FRONT** propuesta, es posible concluir que la relación de costo es de aproximadamente 6 a 1.

Etapa	Tiempo (seg)	Desv. Est.
Cálculo de coeficientes	3.48	0.10
Reordenamiento	0.20	0.02
Resolución	4.91	0.05
Total	8.59	0.06

Cuadro 5: Tiempo de cálculo de las distintas etapas de la rutina **FRONT** propuesta.

4. ESTUDIOS REALIZADOS

Este trabajo se centra en la realización de diferentes estudios del desempeño computacional de la propuesta de modificación del modelo RMA-10 realizada anteriormente. En particular, se evalúa el impacto de utilizar distintos estrategias de ordenamiento de pivotes y la capacidad de escalado de la propuesta ante el aumento en el tamaño de los problemas.

4.1. Estrategias de ordenamiento

La obtención de buenos tiempos de ejecución al resolver sistemas lineales utilizando métodos multifrontales esta condicionada por la aplicación de estrategias de ordenamiento de pivotes. Encontrar un ordenamiento óptimo de los pivotes es un problema NP-difícil, por lo que las distintas estrategias se basan en heurísticas que intentan lograr una solución de compromiso entre el tiempo de cálculo, los posibles errores numéricos y la minimización del fill-in (la

aparición de nuevos coeficientes no nulos en la matriz durante su factorización), explotando diferentes características de las matrices. Esto implica que la aplicación de una estrategia de ordenamiento puede ser muy buena para un tipo de matrices, pero no ser efectiva para otro. La biblioteca MUMPS dispone de las estrategias de ordenamientos AMD [4], AMF [12], PORD [13] y QAMD [3].

El estudio consistió en evaluar los tiempos de ejecución de la rutina **FRONT** sobre sistemas lineales del caso N-31K, utilizando las diferentes estrategias de ordenamiento. En el Cuadro 6 se pueden observar el tiempo de ejecución de la rutina al utilizar las distintas estrategias de reordenamiento.

Ordenamiento	Tiempo (seg)	Desv. Est.
AMD	8.59	0.06
AMF	8.75	0.04
PORD	8.85	0.06
QAMD	8.50	0.04

Cuadro 6: Tiempo de cálculo de la rutina **FRONT** utilizando diferentes estrategias de ordenamiento.

En base a los resultados obtenidos se puede afirmar que las estrategias evaluadas alcanzan resultados similares sobre la clase de sistemas lineales que se intentan resolver. Aunque las estrategias QAMD y AMD logran una pequeña ventaja, siendo esta última la estretegia que por defecto escoge la bibloteca para resolver los sistemas con los que se prueba en esta sección.

4.2. Escalabilidad

La segunda parte del estudio se centro en evaluar la escalabilidad de la propuesta ante el aumento de dimensiones de los problemas.

La Figura 1 muestra la evolución del tiempo de ejecución de la rutina **FRONT** original y la propuesta, al variar la cantidad de ecuaciones de los sistemas a resolver, para el caso N-31K.

Los resultados que se obtienen son sumamente alentadores con respecto a la rutina. Cuando se resuelven pocas ecuaciones (menos de 7000), el desempeño de la versión original de la rutina es igual o superior al de la versión propuesta. Sin embargo, a medida que se incrementa el número de ecuaciones a resolver, las diferencias en los tiempos de ejecución a favor de la rutina propuesta aumentan considerablemente hasta llegar a su punto máximo para el caso del mayor sistema lineal evaluado, de 30787 ecuaciones. En este último caso, la versión propuesta resulta 4 veces más rápida.

Una vez obtenidos los resultados sobre la escalabilidad de la versión propuesta con el caso de prueba N-31K, se intentaron validar dichos resultados trabajando con sistemas del caso N-40K. Al utilizar el caso N-40K no solo se validó la teoría, sino que se descubrió un hecho relevante. Una cualidad de los sistemas lineales de este caso es que si bien las matrices con las que trabaja poseen un número mayor de ecuaciones que en la resolución de los sistemas lineales del caso N-31K, la densidad (cantidad de coeficientes no nulos sobre cantidad total de coeficientes) de las matrices del caso N-40K es mucho menor. Como ejemplo de las diferencias de densidad mencionadas anteriormente se puede observar que las matrices de mayor tamaño

en el caso N-31K son de 30787 ecuaciones y poseen en el entorno de 2854000 coeficientes no nulos, lo que implica una densidad en el entorno de 0,0030. Por su parte, las mayores matrices en el caso N-40K son de 39759 ecuaciones y cuentan con aproximadamente 2333000 coeficientes no nulos implicando, una densidad cercana a 0,0015.

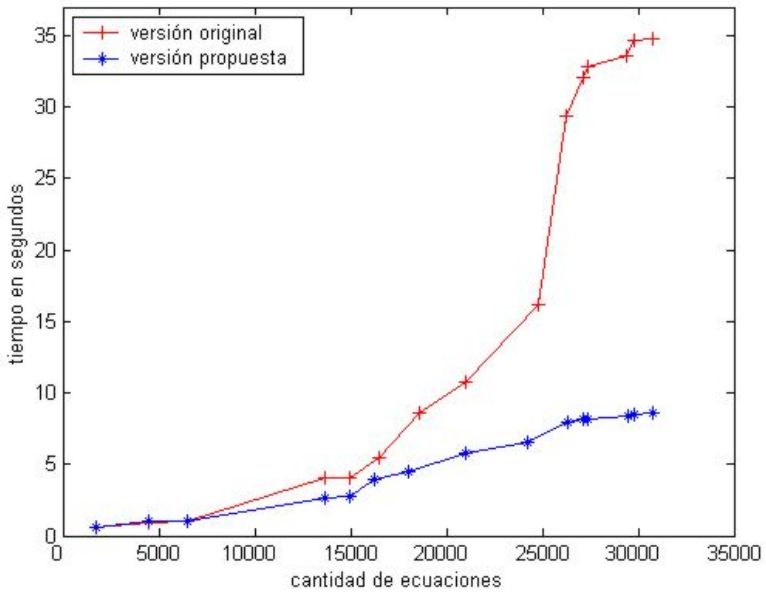

Figura 1: Escalabilidad de la rutina propuesta al aumentar el número de ecuaciones a resolver.

Las cualidades de los sistemas lineales presentes en ambos casos permiten estudiar la dependencia de los tiempos de cómputos de la rutina **FRONT** original y de la versión propuesta con respecto al número de ecuaciones, así como la dependencia con respecto a la cantidad de coeficientes no nulos presentes en las matrices. El Cuadro 7 presenta dichos resultados.

Caso	Coef. no nulos	Versión	Tiempo (seg)	Desv. Est
N-31K	2853756	Original	35.57	0.19
N-31K	2853756	Propuesta	8.59	0.06
N-40K	2332752	Original	54.01	0.14
N-40K	2332752	Propuesta	6.06	0.13

Cuadro 7: Tiempos de la rutina **FRONT** original y propuesta para los casos N-31K y N-40K.

Por último, se realizaron algunos experimentos para evaluar la escalabilidad de la rutina propuesta con sistemas lineales del caso N-40K modificado. Se evaluaron los tiempos de ejecución de la rutina **FRONT** propuesta para distintos sistemas lineales, los cuales poseen cantidades mayores de ecuaciones que los evaluados anteriormente. Los resultados de los experimentos se resumen en el Cuadro 8.

Al observar los datos de el Cuadro 8 se puede corroborar los resultados anteriores, ya que los tiempos de ejecución están ligados a la cantidad de coeficientes no nulos.

5. CONCLUSIONES

Se realizó un breve estudio de la influencia en el desempeño computacional de utilizar las distintas estrategias de ordenamiento de pivotes disponibles en la biblioteca MUMPS. La biblioteca incluye implementaciones de las estrategias AMF, AMD, QAMD y PORD. En todos los casos probados, los tiempos de ejecución para resolver los sistemas lineales involucrados en la aplicación del modelo RMA-10 al Río de la Plata con las distintas estrategias de ordenamiento fueron similares, mostrando una leve ventajas las estrategias AMD y QAMD.

Un aspecto relevante de la rutina **FRONT** propuesta son los resultados de escalabilidad ante el aumento de ecuaciones que presenta. Ante aumentos en las dimensiones del problema, la versión propuesta muestra su capacidad de escalar en mejor forma que la versión original, esta afirmación se confirma observando que al resolver sistemas lineales de aproximadamente 7000 ecuaciones los tiempos de ejecución de ambas versiones son equivalentes, mientras que al resolver sistemas del entorno de 30000 ecuaciones la versión propuesta es aproximadamente 4 veces más rápida que la original y al resolver sistemas de 40000 ecuaciones las diferencias entre los tiempos de ejecución llegan a ser de 9 a 1 a favor de la nueva rutina.

Otro aspecto importante de la rutina **FRONT** propuesta es que su tiempo de ejecución depende débilmente de la cantidad de ecuaciones de los sistemas lineales a resolver, y se encuentra fuertemente ligado a la cantidad de coeficientes no nulos. Este resultado es importante por dos razones, por un lado porque generalmente la densidad en las matrices vinculadas a la resolución de modelos de elementos finitos es baja, pero también porque brinda una vía de trabajo para disminuir el tiempo de cómputo al trabajar con sistemas lineales de grandes dimensiones, buscando disminuir la densidad de las matrices involucradas.

6. TRABAJO FUTURO

A continuación se describen las líneas de trabajo que se están llevando adelante en la actualidad o se plantean abordar en un futuro cercano.

El proceso de generación de la matriz de rigidez global puede mejorarse significativamente. Entre otras opciones, resulta interesante aprovechar la re-utilización de información de pasos anteriores de tiempo para la facilitar la ubicación de los coeficientes en la matriz de rigidez.

Otro aspecto que se considera promisorio es profundizar el estudio de las estrategia de ordenamiento de los pivotes al factorizar los sistemas lineales. Ya sea evaluando otras heurísticas generales o desarrollando estrategias particulares para el problema.

La resolución de sistemas lineales dispersos es un área de investigación sumamente dinámica y parece interesante el estudio de otras bibliotecas de resolución de sistemas lineales de uso público.

Ecuaciones	Coef. no nulos	Tiempo (seg)	Desv. Est.
67035	4676347	12,24	0,20
90021	6729391	25,84	0,33
113007	8782516	35,95	0,51

Cuadro 8: Tiempos de ejecución de la rutina **FRONT** para matrices del caso N-40K modificado.

Por último, parece importante abordar la inclusión de estrategias de programación paralela al modelo RMA-10 y en particular a la rutina **`FRONT`**.

REFERENCIAS

[1] Sitio web de K. Goto. `www.tacc.utexas.edu/~kgoto/`. Consultado abril 2005.

[2] A. Aho, J. Hopcroft, and J. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.

[3] P. Amestoy. Recent progress in parallel multifrontal solvers for unsymmetric sparse matrices. In *Proceedings of IMACS 97, Berlin*, 1997.

[4] P. Amestoy, T. Davis, and I. Duff. An approximate minimum degree ordering algorithm. *SIAM J. Matrix Anal. Appl.*, 17(4):886–905, 1996.

[5] P. Amestoy, I. Duff, , J. Koster, and J-Y. LÉxcellent. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM J. Matrix Anal. Appl.*, Vol 23 N 1:15–41, 2001.

[6] P. Ezzatti and I. Piedra-Cueva. Mejora del desempeño computacional del RMA-10. In *VIII Congreso Argentino de Mecánica Computacional (MECOM 05), Argentina*, 2005.

[7] M. Fossati. Modelación bidimensional del campo salino en el Río de la Plata. Tesis de Maestría en Mecánica de los Fluidos Aplicada, UdelaR – Uruguay, 2005.

[8] P. Hood. Frontal solution program for unsymmetric matrices. *International Journal for Numerical Methods in Engineering*, 10:379–400, 1976.

[9] B. M. Irons. A frontal solution program for finite element analysis. *International Journal for Numerical Methods in Engineering, 2*, 2:5–32, 1970.

[10] I. King. RMA-10, a finite element model for three-dimensional density stratified flow. TR prepared in co-operation with Australian Water and Coastal Studies for the Sydney Deepwater Outfalls Environmental Monitoring Program, 1993.

[11] I. King. A finite element model for stratified flow RMA-10, Users Guide. 2003.

[12] E. Rothberg and S. Eisenstat. Node selection strategies for bottom-up sparse matrix ordering. *SIAMJMat*, 19(3):682–695, 1998.

[13] J. Schulze. Towards a tighter coupling of bottom-up and top-down sparse matrix ordering methods. *BIT Numerical Mathematics*, 41(4):800–828, 2001.

[14] J. Warner. Barotropic and baroclinic convergence zones in tidal channels. Dissertation, Department of Civil and Environmental Engineering, University of California, Davis, 2000.