SOCIEDAD CHILENA DE INGENIERÍA HIDRÁULICA

XXVII CONGRESO CHILENO DE INGENIERÍA HIDRÁULICA

SWEpy: LIBRERÍA EN PYTHON PARA SIMULACIÓN DE FLUIDOS MEDIANTE LAS ECUACIONES DE AGUAS SOMERAS ACELERADA POR GPU

JUAN A. FUENZALIDA A.[1*]
JOAQUÍN MEZA A.[1*]
DANILO KUSANOVIC M.[2]
RODRIGO MENESES P.[3]

RESUMEN

La simulación precisa de fenómenos hidrológicos extremos, como rotura de presas, tsunamis e inundaciones urbanas, es esencial para la evaluación de riesgo y la planificación de sistemas de alerta temprana. En este trabajo presentamos SWEpy, un solver open-source implementado en Python y acelerado por GPU mediante CuPy, capaz de resolver las ecuaciones de aguas someras sobre mallas triangulares no estructuradas con alto orden de precisión. SWEpy integra un esquema de volúmenes finitos central-upwind, basado en reconstrucción cuadrática WENO y un integrador temporal SSP RK3, que garantiza el well-balancing y la preservación de la positividad en frentes secos/húmedos. Su arquitectura modular bajo licencia GPL facilita la extensión comunitaria y la incorporación de nuevos módulos para distintas necesidades. Validamos el código contra soluciones analíticas y experimentales, y lo verificamos en dos casos reales: la rotura de la presa de Malpasset (Francia, 1959) y la propagación del tsunami del Maule (Chile, 2010). Los resultados muestran errores aceptables en alturas máximas y tiempos de arribo, comparables o superiores a otros solvers consolidados. En términos de rendimiento, SWEpy alcanza aceleraciones de hasta 10×–20× sobre CPU secuencial, incluso en hardware de consumo considerado obsoleto. Estas características convierten a SWEpy en una herramienta competitiva y accesible para la comunidad de ingeniería hidráulica, promoviendo la colaboración abierta y la adopción de simulaciones de alto rendimiento en la investigación y docencia sin barreras comerciales ni necesidad de máquinas de alto rendimiento.

[1]Departamento de Obras Civiles, Universidad Técnica Federico Santa María, Chile
[2]Department of Civil and Environmental Engineering, University of California Davis, USA
[3]Escuela de Ingeniería Civil, Universidad de Valparaíso, Chile
[*]Autores correspondientes: juan.fuenzalidaasansano.usm.cl - joaquin.mezausm.cl

1. INTRODUCCIÓN

La simulación precisa de eventos hidrológicos peligrosos, como rupturas de presas, tsunamis e inundaciones urbanas, desempeña un papel crucial en la evaluación de riesgos, la planificación de emergencias y el funcionamiento de sistemas de alerta temprana (Behrens et al., 2010; Catalan et al., 2020; Fernández-Nóvoa et al., 2024; Harig, Immerz et al., 2019; Lin et al., 2015). A lo largo de los años, se han desarrollado múltiples herramientas numéricas para estos fines, desde modelos de código abierto hasta software comercial, como Fluent de ANSYS, Inc. (2013), o como se ve en la revisión de Jodhani et al. (2023). Estos solvers permiten generar datos altamente valiosos para la evaluación de la vulnerabilidad de las comunidades. Sin embargo, desafíos globales como la rápida urbanización, el cambio climático y las incertidumbres socioeconómicas—especialmente en regiones en desarrollo (United Nations, 2019)—imponen nuevas demandas a los marcos de gestión de riesgos, requiriendo herramientas que sean, a la vez, precisas, eficientes, flexibles, escalables y accesibles a diversos usuarios en investigación, planificación y operación.

Una gran mayoría de los modelos actuales enfocados al modelamiento de este tipo de fenómenos resuelven las ecuaciones de aguas someras, o Shallow Water Equations (SWE) en inglés, base del modelado bidimensional de flujos de superficie libre (Delis y Nikolos, 2021). Entre los solvers de SWE disponibles existe mucha diversidad entre características como geometría de malla (cartesiana vs. no estructurada), esquemas numéricos (Godunov, DG, etc.), paralelización (GPU, MPI) y alcance. Aunque muchos modelos sobresalen en aplicaciones específicas como escurrimiento pluvial (SERGHEI-SWE, cf. Caviedes-Voullième et al. (2023)), inundaciones urbanas (HMS, cf. Simons et al. (2013)), o propagación de tsunamis (HySEA, cf. Macias et al. (2014)), la mayoría emplea mallas cartesianas estructuradas, que limita la capacidad de refinamiento local en topografías complejas. Las mallas triangulares, en cambio, permiten refinamientos sin necesidad de acoplamientos ni anidación. Este tipo de refinamiento progresivo es altamente útil para capturar con detalle costas irregulares o entornos urbanos densos. Sin embargo, pocos solvers combinan estas mallas con esquemas numéricos de volúmenes finitos que incluyan reconstrucciones de alto orden que mantengan el equilibrio perfecto (well-balancing) y la preservación de la positividad en frentes seco/mojado, ya que su implementación puede volverse costosa computacionalmente para ejemplos reales, dejando un vacío en solvers que combinen mallas triangulares no estructuradas con métodos de volúmenes finitos de alto orden para mejorar la precisión en escenarios complejos.

Para sobrepasar la barrera computacional, los desarrolladores de dichos softwares han dedicado sus esfuerzos a implementar sus esquemas de forma paralelizada, ya sea a través de arquitecturas CPU o GPU. Estas implementaciones son, en general, desarrolladas en lenguajes de programación poco accesibles para el público general como C/C++ y FORTRAN, o a través de APIs como OpenMP o CUDA, lo que podría limitar su uso y potencial desarrollo. Por su parte, Python se presenta como una alternativa atractiva que, en la última década, ha adquirido capacidades de computación en paralelo a través de la librería CuPy (Okuta et al., 2017), que implementa el clásico paquete NumPy a través de núcleos CUDA, sin necesidad de entrar en programación especializada. El uso de estas herramientas se ve aún más atractivo considerando la filosofía Open Source, que fomenta la participación colaborativa de investigadores para adaptar y mejorar los modelos originales, asegurando la accesibilidad sin barreras comerciales.

Con el objetivo de cubrir las carencias mencionadas, en este trabajo se presenta SWEpy, un solver de código abierto en Python con aceleración GPU mediante CuPy. Las contribuciones principales del trabajo son: (1) la implementación y validación de un esquema FV central-upwind de orden superior, combinando reconstrucción cuadrática WENO y un integrador SSP RK3 para reducir la difusión numérica en escenarios cercanos (choques, frentes húmedo/seco) y lejanos (propagación de ondas). (2) Arquitectura modular bajo licencia GPL, que facilita la reproducibilidad y la extensión comunitaria (p. ej., incorporación de nuevos términos fuente). Y (3), implementación accesible en Python/CuPy para habilitar aceleración GPU en hardware de consumo, sin requerir conocimientos de bajo nivel. El software propuesto se valida contra soluciones analíticas y pruebas de laboratorio, y posteriormente se verifica a través de la simulación de dos casos reales: la rotura del 1959 de la represa Malpasset en Francia (Electricité de France, Direction des Études et Recherches (EDF-DER), 2000; Le y Viet, 2017), y la propagación del tsunami del Maule de 2010 (Benavente y Cummins, 2013; Harig, Zamora et al., 2022); ambos ejemplos altamente estudiados en la literatura.

2. METODOLOGÍA

2.1 ECUACIONES DE GOBIERNO

El problema físico ilustrdo en la siguiente figura:

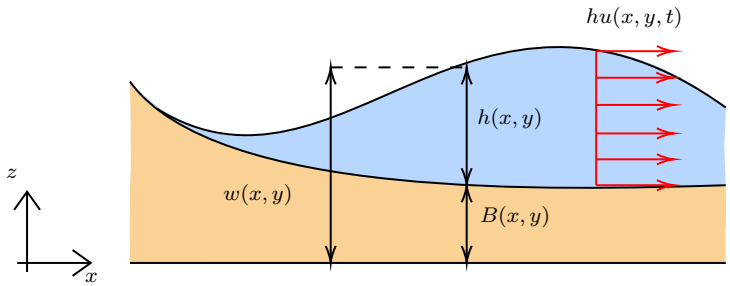

Figura 1: Problema físico y variables asociadas.

se puede modelar a través de las ecuaciones de aguas someras (SWE, por su sigla en inglés), derivadas de promediar en la vertical las ecuaciones de Navier-Stokes bajo la asunción hidrostática, presentadas a continuación incluyendo términos fuente:

$$\boldsymbol{q}_t + \boldsymbol{f}(\boldsymbol{q})_x + \boldsymbol{g}(\boldsymbol{q})_y = \boldsymbol{S}_B(\boldsymbol{q}) + \boldsymbol{S}_C(\boldsymbol{q}) + \boldsymbol{S}_F(\boldsymbol{q}) \tag{1}$$

donde

$$\boldsymbol{q} = (h, hu, hv)^\top, \tag{2}$$

$$\boldsymbol{f}(\boldsymbol{q}) = \left(hu, hu^2 + \frac{gh^2}{2}, huv\right)^\top \; ; \; \boldsymbol{g}(\boldsymbol{q}) = \left(hv, huv, hv^2 + \frac{gh^2}{2}\right)^\top, \tag{3}$$

$$\boldsymbol{S_B}(\boldsymbol{q}) = (0, -ghB_x, -ghB_y)^\top \quad ; \; \boldsymbol{S_C}(\boldsymbol{q}) = (0, fhv, -fhu)^T, \tag{4}$$

$$\boldsymbol{S_F}(\boldsymbol{q}) = \left(0, -g\frac{n^2}{h^{7/3}}hu\sqrt{(hu)^2 + (hv)^2}, -g\frac{n^2}{h^{7/3}}hv\sqrt{(hu)^2 + (hv)^2}\right)^T, \tag{5}$$

con g la aceleración de la gravedad, f el parámetro de Coriolis, y n la constante de Manning-Strickler.

2.2 ESQUEMA EN VOLÚMENES FINITOS

Integrando la ecuación (1) en un sistema Ω de volúmenes finitos (celdas, elementos) Ω_j, y usando el teorema de la divergencia, se llega a la formulación en volúmenes finitos

$$\frac{d}{dt}\boldsymbol{q}_j(t) + \frac{1}{|\Omega_j|}\sum_{k \in \mathcal{N}_j} \mathcal{F}_{jk}\, l_{jk} = \boldsymbol{S_B} + \boldsymbol{S}_C + \boldsymbol{S}_F \tag{6}$$

donde los vectores conservados ahora representan el valor promedio sobre la celda indexada, $\mathcal{F}_{jk}$ es el flujo numérico a través del lado Γ_{jk} que comparte la celda con su vecino Ω_{jk}, cuyo largo es l_{jk}.

Usando esta formulación, Kurganov y Petrova (2005) desarrollan un esquema de tipo Central-Upwind (CU) para mallas triangulares no estructuradas, en donde los flujos $\mathcal{F}_{jk}$ se calculan evitando el uso de Riemann solvers (Roe, 1981; Toro et al., 1994; Murillo y García-Navarro, 2012) a través de integraciones sobre abanicos de Riemann, y estimando los estados en el borde de los triángulos a través de operadores de reconstrucción polinomiales.

Las ecuaciones (6), junto con el flujo numérico $\mathcal{F}_{jk}$ de Bryson et al. (2011), definen una EDO para cada celda, la cual puede ser resuelta numéricamente a través de esquemas temporales como Forward Euler (FE) o Runge-Kutta (RK). En nuestro trabajo se utiliza FE y Runge-Kutta de 4 pasos y orden 3 (RK3,4) (Gottlieb, Shu y Tadmor, 2001), con un paso temporal Δt adaptativo, calculado mediante el cumplimiento de la condición CFL asociada al problema. Esta formulación en volúmenes finitos es naturalmente paralelizable, ya que cada término se expresa en fórmulas para una celda arbitraria, lo cual se puede calcular en términos vectoriales operando `arrays` de NumPy/CuPy.

2.3 DISCRETIZACIÓN DE TÉRMINOS FUENTE

Para preservar soluciones del tipo estacionario y lago en reposo, es necesario discretizar de for-

ma correcta los términos fuente utilizados de forma que se balanceen exactamente con los flujos calculados. Esta es la llamada condición de equilibrio exacto, buen-balanceo, o well-balanced en inglés.

Para el término fuente asociado al gradiente de la batimetría, se sigue la discretización realizada por Bryson et al. (2011). En el caso del término fuente relacionado a la fricción, se realiza un tratamiento siguiendo el procedimiento descrito por Chertock et al. (2015), incluyendo el paso correctivo semi-implícito en el tiempo. Para el término fuente de Coriolis, no es necesario una consideración especial, ya que si el flujo es nulo, el término de Coriolis también lo será.

2.4 OPERADORES DE RECONSTRUCCIÓN

Debido a la versatilidad que ofrece la familia de esquemas presentada por Kurganov y Petrova (2005), distintos operadores de reconstrucción son implementados y probados.

En primer lugar, el operador de reconstrucción más sencillo es el constante, que asume estados en los bordes de la celda idénticos al promedio. Este operador se demuestra bastante difusivo, como se puede ver en la sección siguiente.

Luego, una mejor reconstrucción se puede lograr a través de polinomios lineales. Como esta reconstrucción puede introducir oscilaciones espurias a la solución, se implementa un operador tipo minmod como el descrito por Bryson et al. (2011), que elige el polinomio interpolante, de entre tres construidos con los valores medios de la celda y dos de sus vecinos, minimizando la norma del gradiente.

Para reconstrucción cuadrática, se utiliza el operador WENO-Z3 (Zhu y Qiu, 2018), que combina un polinomio cuadrático construido a partir de información de dos capas de vecinos, con cuatro polinomios lineales construidos a partir de informacion de cuatro subconjuntos de este sistema de vecinos. Un dibujo esquemático del *stencil* de vecinos para cada celda se puede ver en la figura siguiente:

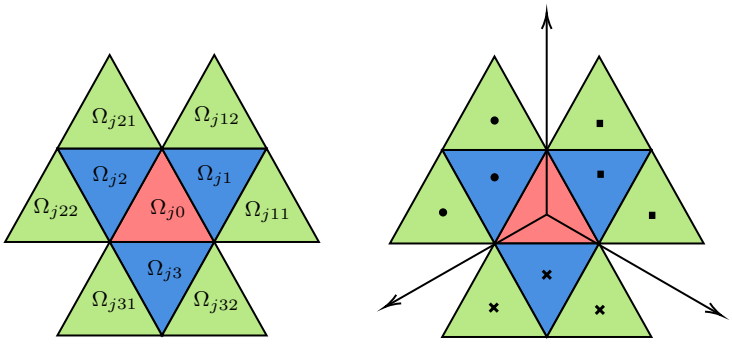

Figura 2: Diagrama del stencil para una celda Ω_j (izquierda) y división sectorial (derecha). los triángulos azules Ω_{jk} integran la primera capa de vecinos, mientras que los triángulos verdes son parte de la segunda capa de vecinos Ω_{jkl}.

La construcción de estos 5 polinomios por celda (uno cuadrático, cuatro lineales) se realiza a través de la solución por mínimos cuadrados de sistemas lineales sobredeterminados:

$$\boldsymbol{a}_j^{t_n} \overset{lsq}{=} (M_j^T M_j)^{-1} M_j^T \Delta q_j^{t_n}. \tag{7}$$

En donde el vector $a_j^{t_n}$ contiene los coeficientes del polinomio en el tiempo t_n, M_j es una matriz que depende sólo de información geométrica de la celda Ω_j y sus vecinos, y $\Delta q_j^{t_n}$ es un vector que contiene la diferencia entre la variable conservada en cada uno de los vecinos y la celda Ω_j, en el tiempo t_n. Como la multiplicación de matrices asociada al problema no depende del tiempo, se puede precalcular y guardar para ahorrar tiempos de simulación. Este precómputo es especialmente eficiente usando CuPy ya que el módulo `linalg` se encarga de estas operaciones matriciales sobre la GPU.

Esta estrategia es novedosa, ya que, hasta donde los autores tienen conocimiento, reconstrucciones de este tipo no han sido implementadas en el esquema CU para las SWE.

2.5 TRATAMIENTO SECO/MOJADO

Debido a la resolución *sub-celda* conseguida con los reconstructores, el programa debe encargarse de que la altura de agua en ningún vértice de ésta quede por debajo de la batímetría. De esta forma se evita la introducción de desbalances numéricos, y, principalmente, se conserva la fisicidad del modelo.

En el proceso de cálculo de la reconstrucción, se permite una ligera permeabilidad en la celda, resultando a veces alturas negativas de agua en el interior de las celdas en el frente seco/mojado. Sin embargo, en el paso inmediatamente siguiente se reemplaza la reconstrucción de la superficie de agua por una que mantenga todos los vértices por sobre la batimetría y conserve el volumen de fluido sobre la celda, siguiendo lo descrito por Bryson et al. (2011), como se muestra en la siguiente figura de forma esquemática

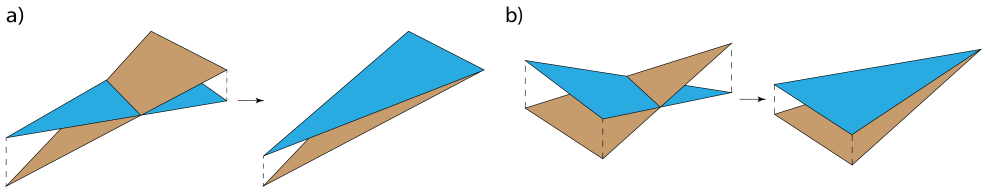

Figura 3: Tratamiento seco/mojado: (a) caso con dos puntos secos y (b) caso con un punto seco.

Este método es implementado en SWEpy mediante un algoritmo que aprovecha el indexado de las celdas y las capacidades del paquete CuPy para encontrar y corregir todas las celdas que lo requieren de forma simultánea.

3. RESULTADOS

3.1 VALIDACIÓN

Gracias a trabajos como los de Synolakis (1987), Briggs et al. (1995), Synolakis et al. (2008), o Delestre et al. (2012), se dispone ampliamente de pruebas canónicas o *benchmarks* para la validación de modelos numéricos como SWEpy. Algunos de estos benchmarks son usados para validar el programa, mostrando resultados satisfactorios. A continuación se presenta uno de estos resultados.
Entre las pruebas de validación realizadas, se simula el experimento de laboratorio de Briggs et al. (1995) en donde una ola solitaria (soliton) incide en una isla cónica. Un total de 27 medidores de presión son distribuidos en distintos puntos del estanque: en el fondo del estanque entre la ola y la isla, en la ladera de la isla de cara a la ola, en la ladera a ambos lados de la isla, y en la ladera contraria a la posición inicial de la ola (cf. figura 4a).

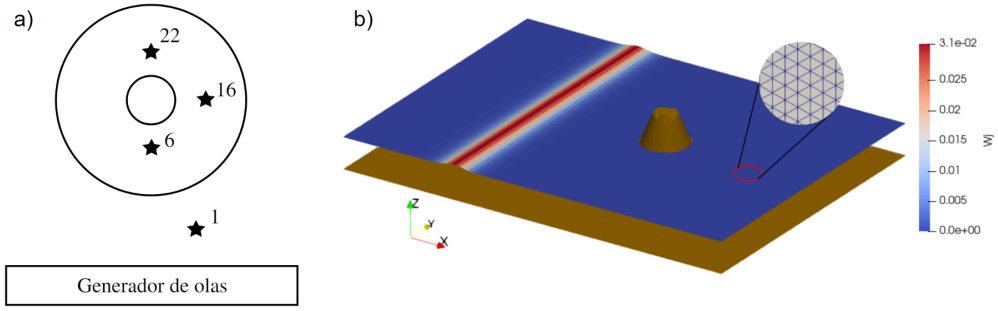

Figura 4: Posición (esquemática) de puntos de medición (a) y condiciones iniciales para la simulación numérica (b) con cuadro de vista que muestra la geometría de la malla utilizada.

Las condiciones iniciales se replican en el modelo computacional junto con medidores virtuales en los puntos correspondientes. Se generan series de tiempo con la altura de agua registrada en los medidores virtuales y se grafican junto con los datos experimentales en la siguiente figura:

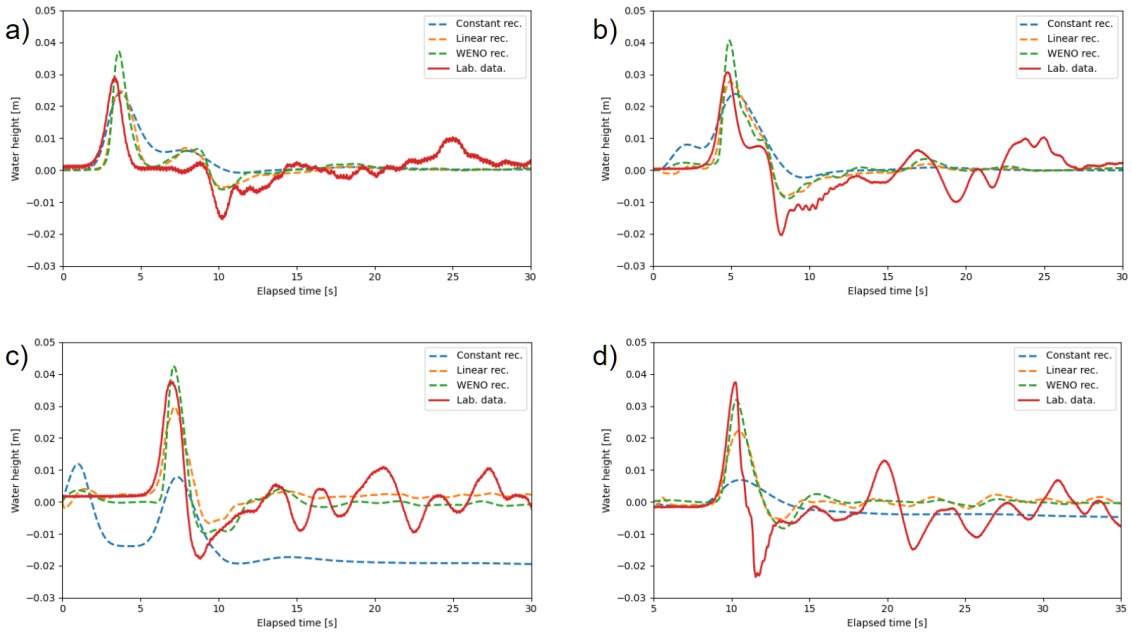

Figura 5: Series de tiempo para medidores 1 (a), 6 (b), 16 (c), y 22 (d) del experimento "Isla Cónica" realizado por Briggs et al. (1995). En línea roja sólida se ve la serie de tiempo medida, y en segmentadas las soluciones numéricas usando distintos reconstructores

Se puede ver que SWEpy modela bien el arribo de la ola en los distintos puntos, junto con la evolución de los primeros instantes de la incidencia del soliton. Se observa una clara ventaja del reconstructor WENO sobre los otros reconstructores implementados. Algunas de las diferencias entre las mediciones y las simulaciones podrían explicarse debido a factores como la generación del soliton en el experimento físico, ya que no alcanza la relación de altura/profundidad teórica de $0{,}1$, con mediciones de $0{,}093$ y $0{,}091$. Otro factor que podría explicar los errores, principalmente en las alturas negativas medidas, son las aceleraciones que podrían darse en la vertical para el caso físico, imposibles de capturar con el modelo de las SWE que asume presión hidrostática, entre otras hipótesis.

3.2 ROTURA DE MALPASSET

Para el caso de la rotura de la represa Malpasset, se replican las condiciones que se indican en el manual de validación del software TELEMAC (Electricité de France, Direction des Études et Recherches (EDF-DER), 2000), ampliamente utilizado en simulaciones de todo tipo. Esto incluye la malla, la bati-topografía de la zona, las condiciones iniciales ($w = 100$[m] aguas arriba de la represa, 0 aguas abajo), las condiciones de borde tipo muralla impermeable, el tiempo de simulación $t_{\text{máx}} = 4000$[s] y la constante de Manning $n = 0{,}03$. Además, gracias al registro de Electricité de France (EDF) se tienen los tiempos de fallo de 3 transformadores (puntos A, B, C), indicando el tiempo de arribo de la ola de inundación; y las alturas máximas registradas en una serie de puntos ubicados a lo largo del frente de inundación, a partir de un modelo en escala $1 : 400$. Estos datos se levantan también del dataset de TELEMAC.

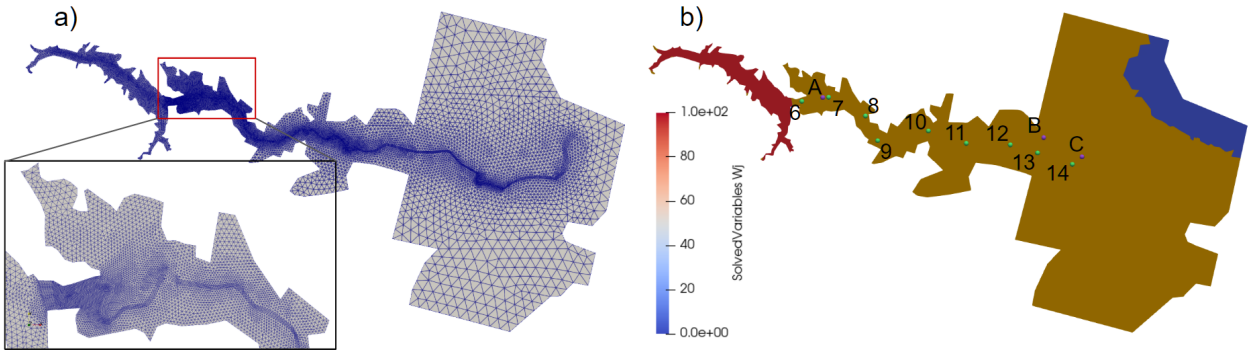

Figura 6: Malla utilizada en la simulación (a) con cuadro de vista de zona refinada al pie de la represa, y altura inicial de agua (b) con puntos de medición.

Medidores virtuales se colocan en dichos puntos para generar una serie de tiempo de la altura de columna de agua sobre ellos. La serie de tiempo es utilizada para identificar los tiempos de arribo y las alturas máximas de la ola al pasar. Se comparan los tiempos de arribo de la ola desde que pasa por A (como tiempo de referencia) hasta llegar a B y C (t_{arrA}), y las alturas máximas ($h_{\text{máx}}$) en el resto de los puntos, con los datos medidos, y los resultados de la validación de TELEMAC. Estos datos se pueden encontrar en la siguente tabla, junto a los errores relativos de cada uno.

Cuadro 1: Resultados para SWEpy y TELEMAC, comparados con datos medidos por EdF.

Punto	Datos registrados		SWEpy			TELEMAC		
	h_{max} [m]	t_{ArrA} [s]	h_{max} [m]	t_{ArrA} [s]	Err [%]	h_{max} [m]	t_{ArrA} [s]	Err [%]
A	-	-	-	-	-	-	-	-
B	-	1140	-	1071	-6	-	1142.9	0
C	-	1320	-	1088	-18	-	1387.3	5
P6	40.3	-	37.72	-	-6	81.58	-	102
P7	14.6	-	18.13	-	24	55.88	-	283
P8	24.0	-	22.46	-	-6	53.21	-	122
P9	12.8	-	18.6	-	45	48.14	-	276
P10	11.8	-	15.34	-	30	36.88	-	213
P11	8.3	-	6.21	-	-25	25.41	-	206
P12	10.1	-	5.89	-	-42	19.29	-	91
P13	6.8	-	12.21	-	80	17.74	-	161
P14	5.4	-	4.32	-	-20	12.71	-	135

Se observan predicciones de altura de agua por parte de SWEpy de hasta un orden de magnitud mejores que las obtenidas por TELEMAC, cuyo error porcentual promedio es de $176{,}6\,\%$. El punto P13 se considera un outlier debido a la resolución de la topografía utilizada, ya que puntos alrededor de éste presentaban alturas de agua simulados entre 4 y 7 metros, valores mucho más cercanos a los registrados por EdF. Además, debido al acercamiento progresivo de las alturas simuladas por TELEMAC a las registradas, se puede deducir que el modelo de TELEMAC introduce difusión a medida que la ola viaja a través del dominio, por lo que los tiempos de arribo podrían estar influenciados por este carácter difusivo, llegando más tarde de lo que simula SWEpy.

3.3 TSUNAMI DE MAULE 2010

Usando la inversión de Benavente y Cummins (2013) para calcular la deformación inicial via el modelo de Okada, y los datos batimétricos grillados que provee GEBCO (GEBCO Bathymetric Compilation Group 2024, 2024), se simula el tsunami producido por el terremoto de Maule del 2010 con el fin de evaluar la capacidad del programa de predecir tsunamis de larga distancia.

La grilla utilizada abarca las longitudes desde los $80°$ hasta los $300°$, y latitudes entre los $-60°$ y $60°$. La malla es más refinada en las zonas donde la batimetría varía fuertemente para no suavizar los cambios bruscos de pendiente. Se producen 2 mallas: una de alrededor de 250 mil triángulos (malla 1) con alturas promedio de ~ 44[km]; y otra de cerca de 1.4 millones de elementos (malla 2), con alturas promedio de ~ 18[km], con el objetivo de evaluar el rendimiento del programa, tanto en tiempos de computación como calidad de resultados, con distintos niveles de refinamiento. Como datos de referencia, se escogen las alturas de agua (rectificadas con respecto a mareas) medidas por las boyas DART (Mungov et al., 2005) número 21413 (SE de Kyoto), y 32412 (SE de Lima) en el evento, aunque se dispone de un total de 32 boyas DART con datos históricos. La batimetría utilizada y la ubicación de las boyas referenciadas se pueden apreciar en la siguiente figura:

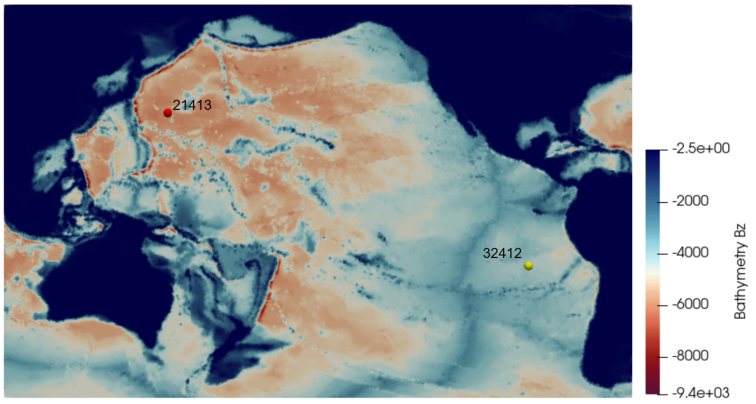

Figura 7: Batimetría utilizada (malla 1) y ubicación de boyas DART.

La batimetría es truncada en cada malla de forma que todo el dominio permanezca sumergido, con el objetivo de no considerar los efectos del tratamiento seco/mojado en el cálculo debido a su influencia en el buen-balanceo y los tiempos de cálculo. Medidores virtuales se colocan en las celdas que contienen la ubicación de las boyas DART y se registran series de tiempo para las alturas de agua sobre ellas. Los perfiles de tsunami simulados, usando distintos reconstructores espaciales e integradores temporales, junto con los perfiles medidos trasladados en el tiempo se grafican en las figuras a continuación.

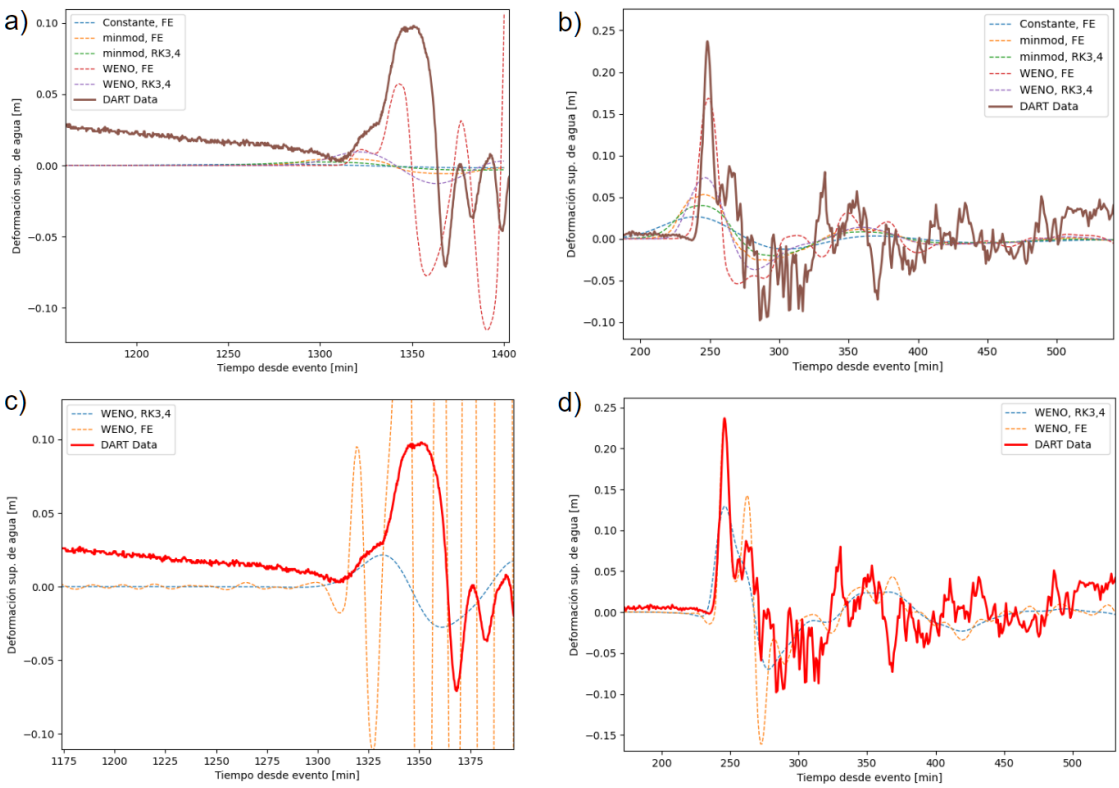

Figura 8: Perfiles de tsunami para las boyas DART en Lima (a,c) y Kyoto (b,d), sobre las mallas 1 (a,b) y 2 (c,d).

Se puede observar que, a pesar de entregar mejores resultados, el esquema FE presenta oscilaciones que eventualmente se salen de control, por lo que su mejora podría deberse a su carácter oscilatorio. Dichas oscilaciones se controlan mediante un paso temporal más pequeño, pero esto resulta en mayor difusividad. Por su parte, el esquema RK3,4 goza de la propiedad SSP Gottlieb, Shu y Ketcheson (2010) por lo que no introduce oscilaciones. Aunque parezca más difusivo, este esquema mejora considerablemente con una malla de mayor resolución como la malla 2, por lo que una malla con una refinación aún más adecuada entregaría los mejores resultados. Se comenta también que los tiempos de arribo calculados para puntos de rango lejano (como la boya de Kyoto) difieren de los reales por entre $-2{,}1\,\%$ y $3{,}6\,\%$ para la malla 1, y entre $2{,}7\,\%$ y $3{,}2\,\%$ para la malla 2. En cuanto a los reconstructores espaciales, el operador WENO es sin duda nuevamente el mejor.

En términos de tiempos de computación, la malla 1 registró tiempos de reloj de entre 2 minutos (FE con minmod) y 18 minutos (RK3,4 con WENO), sobre una GPU NVIDIA GeForce GTX 1650 promediando un 80 % de utilización. Para la malla 2, se registraron tiempos de ~ 70 minutos (FE con WENO), y ~ 130 minutos (RK3,4 con WENO), sobre una NVIDIA GeForce RTX 3080 promediando 81 % de utilización. En comparación con una versión serial (CPU, un hilo) del programa en tiempos de computación (CPU/GPU time), la versión GPU muestra aumentos de velocidad de hasta 2100 % para la malla 1 con RK3,4 y WENO, y 1300 % con FE y minmod. Todo esto quiere decir que el software es sumamente eficiente, a pesar de implementar algoritmos que requieren una gran cantidad de operaciones sobre una gran cantidad de elementos, incluso en hardware de nivel

consumidor de hace 4 generaciones.

4. CONCLUSIONES

En este trabajo se presentó SWEpy, un solver de código abierto en Python con aceleración GPU mediante CuPy, diseñado para resolver las SWE sobre mallas triangulares no estructuradas de forma precisa y eficiente.

El esquema central-upwind combinado con reconstrucción cuadrática WENO y el integrador SSP RK3 demuestra una reducción significativa de la difusión numérica y oscilaciones espurias, tanto en interacciones onda-obstáculo como en la propagación lejana de ondas. SWEpy mantiene el buen-balanceo y la positividad en todas las pruebas realizadas, lo que garantiza la conservación de estados de reposo y el comportamiento físico deseado en escenarios reales. Las simulaciones de la rotura de la represa Malpasset por su parte muestran un acuerdo con los datos de referencia, superando estudios previos como lo es la validación de TELEMAC. Por otro lado, las simulaciones del tsunami del Maule muestran la eficiencia de los esquemas implementados, a pesar de sus respectivas falencias. Esto confirma la capacidad de SWEpy para reproducir fenómenos hidrodinámicos complejos.

La implementación en Python/CuPy permite explotar el potencial de las GPUs para obtener aceleraciones de hasta un factor 21× respecto a una versión secuencial en CPU, incluso en hardware considerado obsoleto, manteniendo un código modular y legible. Esta aproximación, junto con la filosofía Open Source, democratizan el uso de cálculos paralelos en investigación y enseñanza, eliminando la barrera que suponen lenguajes de bajo nivel o licencias comerciales. Así, bajo licencia GPL, SWEpy fomenta la colaboración y extensión comunitaria, teniendo ya módulos adicionales en desarrollo gracias a su arquitectura modular que facilita la incorporación de nuevos términos fuente, esquemas temporales u operadores de reconstrucción, potenciando su evolución sin costos de licencia.

Actualmente, SWEpy (disponible bajo licencia abierta en https://github.com/joaquinmeza90/SWEpy) se centra en la resolución de las SWE bidimensionales con fricción de Manning y efecto Coriolis. Futuras líneas de investigación incluyen: (1) Extensión a modelos de sedimentos, transporte de sólidos, y flujos bifásicos, (2) incorporación de términos fuentes asociados a escurrimiento pluvial o infiltración y acoplamiento a modelos relacionados, (3) incorporación de términos correctores para flujos con fuertes pendientes o porosidad, y (4) optimización de la implementación para mejorar los tiempos de cálculo y admitir mallas de mayor tamaño.

En conjunto, SWEpy constituye una herramienta competitiva para la simulación de flujos superficiales en ingeniería hidráulica, que combina precisión, eficiencia y accesibilidad. Su desarrollo open-source abre el camino a una adopción amplia en la comunidad científica e industrial, contribuyendo al fortalecimiento de los sistemas de evaluación de riesgo y planificación de emergencias.

REFERENCIAS

ANSYS, Inc. (nov. de 2013). *ANSYS Fluent Theory Guide*. Version date: November 2013. ANSYS, Inc. 275 Technology Drive, Canonsburg, PA 15317.

Behrens, J. et al. (jun. de 2010). «A new multi-sensor approach to simulation assisted tsunami early warning». En: *Natural Hazards and Earth System Sciences* 10(6), págs. 1085-1100. ISSN: 1684-9981.

Benavente, Roberto y Phil R. Cummins (jul. de 2013). «Simple and reliable finite fault solutions for large earthquakes using the W-phase: The Maule (Mw = 8.8) and Tohoku (Mw = 9.0) earthquakes». En: *Geophysical Research Letters* 40(14), págs. 3591-3595. ISSN: 1944-8007.

Briggs, M. J. et al. (1995). «Benchmark Problem 2: Run-up of Solitary Waves on a Circular Island». En: *International Workshop on Long Wave Modeling of Tsunami Runup*. Vicksburg, MS.

Bryson, Steve et al. (mayo de 2011). «Well-balanced positivity preserving central-upwind scheme on triangular grids for the Saint-Venant system». en. En: *ESAIM: Mathematical Modelling and Numerical Analysis* 45(3). Number: 3 Publisher: EDP Sciences, págs. 423-446. ISSN: 0764-583X, 1290-3841.

Catalan, Patricio A. et al. (2020). «Design and operational implementation of the integrated tsunami forecast and warning system in Chile (SIPAT)». En: *Coastal Engineering Journal* 62(3), págs. 373-388.

Caviedes-Voullième, Daniel et al. (feb. de 2023). «SERGHEI (SERGHEI-SWE) v1.0: a performance-portable high-performance parallel-computing shallow-water solver for hydrology and environmental hydraulics». En: *Geoscientific Model Development* 16(3), págs. 977-1008. ISSN: 1991-9603.

Chertock, A. et al. (2015). «Well-balanced positivity preserving central-upwind scheme for the shallow water system with friction terms». En: *International Journal for Numerical Methods in Fluids* 78(6), págs. 355-383.

Delestre, Olivier et al. (oct. de 2012). «SWASHES: a compilation of shallow water analytic solutions for hydraulic and environmental studies». En: *International Journal for Numerical Methods in Fluids* 72(3), págs. 269-300. ISSN: 1097-0363.

Delis, Anargiros I y Ioannis K Nikolos (2021). *Shallow Water Equations in Hydraulics: Modeling, Numerics and Applications*.

Electricité de France, Direction des Études et Recherches (EDF-DER) (2000). *TELEMAC-2D Validation Document: Version 5.0*. Technical Report. Electricité de France, Direction des Études et Recherches.

Fernández-Nóvoa, Diego, José González-Cao y Orlando García-Feal (2024). «Enhancing Flood Risk Management: A Comprehensive Review on Flood Early Warning Systems with Emphasis on Numerical Modeling». En: *Water* 16(10). ISSN: 2073-4441.

GEBCO Bathymetric Compilation Group 2024 (2024). *The GEBCO_2024 Grid - a continuous terrain model of the global oceans and land.* en.

Gottlieb, Sigal, Chi-Wang Shu y David Ketcheson (mayo de 2010). *Strong Stability Preserving Runge-kutta And Multistep Time Discretizations*. World Scientific Publishing: Singapore, Singapore.

Gottlieb, Sigal, Chi-Wang Shu y Eitan Tadmor (2001). «Strong Stability-Preserving High-Order Time Discretization Methods». En: *SIAM Review* 43(1), págs. 89-112.

Harig, Sven, Antonia Immerz et al. (sep. de 2019). «The Tsunami Scenario Database of the Indonesia Tsunami Early Warning System (InaTEWS): Evolution of the Coverage and the Involved Modeling Approaches». En: *Pure and Applied Geophysics* 177(3), págs. 1379-1401. ISSN: 1420-9136.

Harig, Sven, Natalia Zamora et al. (jun. de 2022). «Systematic Comparison of Tsunami Simulations on the Chilean Coast Based on Different Numerical Approaches». En: *GeoHazards* 3(2), págs. 345-370. ISSN: 2624-795X. DOI: 10.3390/geohazards3020018. URL: http://dx.doi.org/10.3390/geohazards3020018.

Jodhani, Keval H, Dhruvesh Patel y N. Madhavan (2023). «A review on analysis of flood modelling using different numerical models». En: *Materials Today: Proceedings* 80, págs. 3867-3876. ISSN: 2214-7853.

Kurganov, Alexander y Guergana Petrova (2005). «Central-upwind schemes on triangular grids for hyperbolic systems of conservation laws». En: *Numerical Methods for Partial Differential Equations* 21(3), págs. 536-552. ISSN: 1098-2426. (Visitado 16-09-2023).

Le, Hien y Hung Viet (jun. de 2017). «SIMULATING MALPASSET (FRANCE) DAM-BREAK CASE STUDY BY A TWO-DIMENSIONAL SHALLOW FLOW MODEL». En.

Lin, Simon C. et al. (mayo de 2015). «Development of a tsunami early warning system for the South China Sea». En: *Ocean Engineering* 100, págs. 1-18. ISSN: 0029-8018.

Macias, Jorge et al. (mayo de 2014). «HySEA: An operational GPU-based model for Tsunami Early Warning Systems». En: *EGU General Assembly Conference Abstracts*. EGU General Assembly Conference Abstracts, 14217, pág. 14217.

Mungov, George, DOC/NOAA/NWS/NDBC >National Data Buoy Center, National Weather Service, NOAA, U.S. Department Of Commerce y DOC/NOAA/OAR/PMEL >Pacific Marine Environmental Laboratory, OAR, NOAA, U.S. Department Of Commerce (2005). *Deep-Ocean Assessment and Reporting of Tsunamis (DART(R))*.

Murillo, J. y P. García-Navarro (ago. de 2012). «Augmented versions of the HLL and HLLC Riemann solvers including source terms in one and two dimensions for shallow flow applications». En: *Journal of Computational Physics* 231(20), págs. 6861-6906. ISSN: 0021-9991.

Okuta, Ryosuke et al. (2017). «CuPy: A NumPy-Compatible Library for NVIDIA GPU Calculations». En: *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*.

Roe, P.L (oct. de 1981). «Approximate Riemann solvers, parameter vectors, and difference schemes». En: *Journal of Computational Physics* 43(2), págs. 357-372. ISSN: 0021-9991.

Simons, Franz et al. (feb. de 2013). «A model for overland flow and associated processes within the Hydroinformatics Modelling System». En: *Journal of Hydroinformatics* 16(2), págs. 375-391. ISSN: 1465-1734.

Synolakis, C. E. (1987). «The runup of solitary waves». En: *Journal of Fluid Mechanics* 185, págs. 523-545.

Synolakis, C. E. et al. (dic. de 2008). «Validation and Verification of Tsunami Numerical Models». En: *Pure and Applied Geophysics* 165(11–12), págs. 2197-2228. ISSN: 1420-9136.

Toro, E. F., M. Spruce y W. Speares (jul. de 1994). «Restoration of the contact surface in the HLL-Riemann solver». En: *Shock Waves* 4(1), págs. 25-34. ISSN: 1432-2153.

United Nations (2019). *World Urbanization Prospects: The 2018 Revision*. ST/ESA/SER.A/420. United Nations publication. United Nations: New York.

Zhu, Jun y Jianxian Qiu (2018). «New Finite Volume Weighted Essentially Nonoscillatory Schemes on Triangular Meshes». En: *SIAM Journal on Scientific Computing* 40(2), A903-A928.