GPU HACKATHON

#gpucesgahack

29 mayo - 1 junio
Santiago de Compostela

TEAM 4

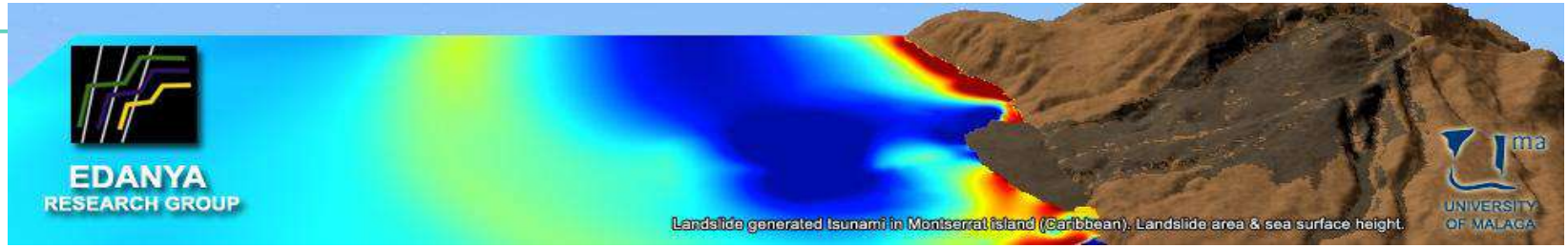
Sergio Ortega Acosta
Ernesto Guerrero Fernández

UNIVERSIDAD DE MÁLAGA

EDANYA

GPU HACKATHON

EDANYA group: Research Group on Differential Equations, Numerical Analysis and Applications



Shallow-Water type models.

- River floods, granular avalanches, submerged landslides, tsunami modeling, multilayer flows, . . .
- **Robust**, High Order and Well-balanced Finite Volume Schemes: path-conservative framework.
- In real problems: we try to find a good balance between **efficiency** and accurate results.
- True **validation** is necessary: laboratory and real events.

GPU HACKATHON

Simple wave on a simple beach - laboratory

Synolakis carried out laboratory experiments for incident solitary waves of multiple relative amplitudes, to study propagation, breaking and run-up over a planar beach with a slope 1 : 19.85[1].

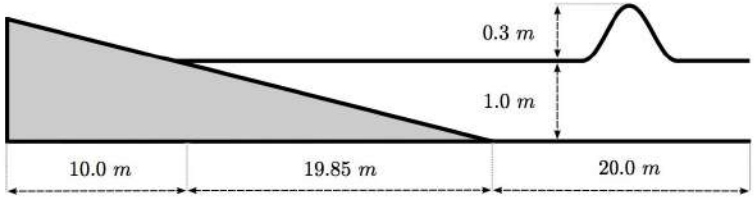


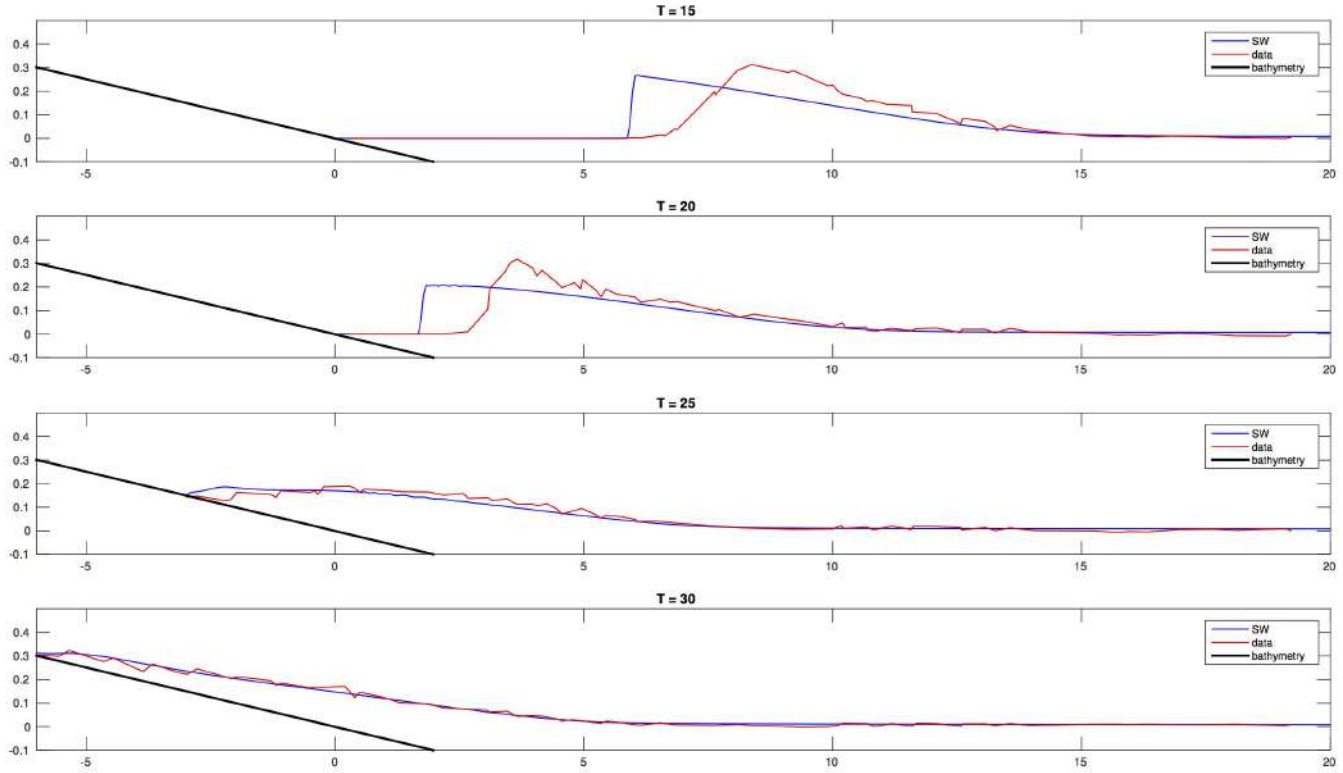
Figure: Sketch of the topography

A solitary wave propagates at constant speed and without change of shape over an horizontal bottom. An approximated expression of a solitary is

$$\eta(x,t) = A \cdot Sech^2\left[\sqrt{\frac{3A}{4h^3}}(x - ct))\right], \; u(x,t) = \frac{\sqrt{gh}}{h}\eta(x,t),$$

[1] C.E. Synolakis. "The runup of solitary waves". In: *Fluid Mechanics* 185 (1987), pp. 523–545.

Shallow-water results



GPU HACKATHON

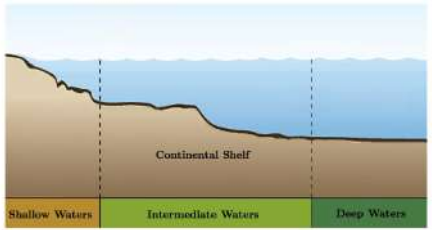
Dispersive models

During a tsunami propagation and inundation, nonuniform velocity profiles over depth are most often encountered. Missing physics:

- Include frecuency dispersion (propagation). Suitable in **intermediate waters** (close to continental shelf),
- Shoaling,
- Easily extended to **shallow water** areas (inundation).

Two big families for the approximations of NSE for coastal-type computations:

- **Boussinesq type Models**: Boussinesq, Peregrine, Madsen-Sorensen, Nwogu, Lynett.
- **Non-Hydrostatic Models**: Casulli, Yamazaki et al, Jaques Saint-Marie et al.

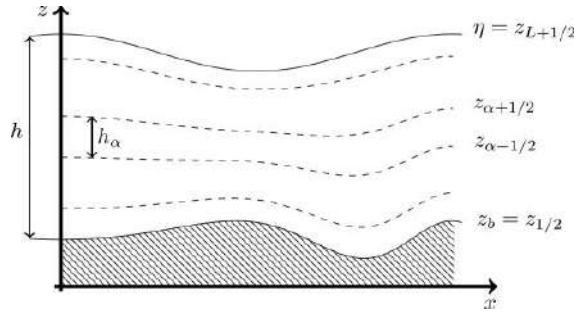


GPU HACKATHON

Non-Hydrostatic Models: Non-Hydrostatic Multi layer-averaged

E.D. Fernandez-Nieto, M. Parisot, Y. Penel, J. Sainte-Marie. 2017

- Layer averaged approximation of Euler.
- Total pressure is decomposed in hydrostatic and non-hydrostatic.
- Horizontal velocity has a constant vertical profile within each layer.
- Vertical velocity has a linear vertical profile within each layer.
- Exact energy balance.
- When the number of layers increases, the celerity converges to the first order theory.
- 1 layer case: JSM equations.



Non-Hydrostatic Models: Jaques Sainte-Marie et al

Jacques Sainte-Marie et al, 2015[2]

$$
\begin{cases}
\partial_t h + \nabla \cdot \boldsymbol{q} = 0, \\
\partial_t \boldsymbol{q} + \operatorname{div}\left(\dfrac{\boldsymbol{q} \otimes \boldsymbol{q}}{h}\right) + \dfrac{1}{2} g \nabla\left(h^2\right) + \dfrac{1}{2} \nabla(hp) = (gh + p) \nabla(H), \\
\partial_t (hw) + \operatorname{div}(hw\boldsymbol{u}) = p, \\
h \nabla \cdot \boldsymbol{q} - \boldsymbol{q} \cdot \nabla(2\eta - h) + 2hw = 0,
\end{cases}
$$

h is the thickness of the water, $\boldsymbol{q} = (q_1, q_2)$ is the discharge, $\boldsymbol{u} = (u, v)$ are the depth averaged velocities components in the x and y directions resp., w is the depth averaged velocity component in the z direction, p is the non-hydrostatic pressure at the bottom, H is the bottom and g is the gravitational constant.

[2] M.-O. Bristeau, A. Mangeney, J. Sainte-Marie, and N. Seguin. "An energy-consistent depth-averaged Euler system: Derivation and properties". In: *Discrete and Continuous Dynamical Systems Series B* 20.4 (2015), pp. 961–988.

A new two layer with improved dispersive properties

Enrique Fernández-Nieto et al

$$\partial_t h + \partial_x (hu) = 0,$$

$$\partial_t (hu_1) + \partial_x \left(hu_1^2 + \frac{1}{2} g h^2 \right) - gh\partial_x H - G_1(u) = -h(\partial_x p_1 + \sigma_1 \partial_z p_1),$$

$$\partial_t (hu_2) + \partial_x \left(hu_2^2 + \frac{1}{2} g h^2 \right) - gh\partial_x H - G_2(u) = -h(\partial_x p_2 + \sigma_2 \partial_z p_2),$$

$$\partial_t (hw_1) + \partial_x (hu_1 w_1) - G_1(w) = -\partial_z p_1,$$

$$\partial_t (hw_2) + \partial_x (hu_2 w_2) - G_2(w) = -\partial_z p_2,$$

$$\partial_x u_{\alpha-1/2} + \sigma_{\alpha-1/2} \partial_z u_{\alpha-1/2} + \frac{1}{h} \partial_z w_{\alpha-1/2} = 0, \ \alpha \in \{1, 2\}.$$

Numerical Scheme: Yamazaki Equations

In general, multi-layer system can be written in the compact form

$$\begin{cases} \partial_t \boldsymbol{U} + \partial_x \boldsymbol{F}_{\mathcal{SW}}(\boldsymbol{U}) + \boldsymbol{B}_{\mathcal{SW}}(\boldsymbol{U})\partial_x \boldsymbol{U} = \boldsymbol{G}_{\mathcal{SW}}(\boldsymbol{U})\partial_x H + \boldsymbol{\mathcal{T}}_{\mathcal{NH}}(h, h_x, H, H_x, p, p_x), \\ \mathcal{B}(\boldsymbol{U}, \boldsymbol{U}_x, H, H_x) = 0, \end{cases}$$

Spatial discretization (splitting technique)[3]**:**

- HLLC path-conservative second order **Finite-Volume** scheme for the underlying hyperbolic shallow-water system. MUSCL reconstruction on space.
- Central **finite-Difference** on a staggered-grid for the non-hydrostatic term (easy to impose the incompressibility condition).

Time discretization:

- Second order Runge-Kutta method.

The standard **CFL condition** related to the hyperbolic shallow-water system should be considered.

[3] M.J. Castro, C. Escalante, and T. Morales. "Non-Hydrostatic Pressure Shallow Flows: GPU Implementation Using Finite–Volume and Finite–Difference Scheme". In: *Submitted to Applied Mathematics and Computation* ().

Finite Volume

$$\mathbf{U}_i'(t) = -\frac{1}{\Delta x}\left(D^-_{i+1/2}(t) + D^+_{i-1/2}(t)\right),$$

where, avoiding the time dependence,

$$\begin{aligned} D^{\pm}_{i+1/2}(t) &= D^{\pm}_{i+1/2}(\mathbf{U}_i(t), \mathbf{U}_{i+1}(t), H_i, H_{i+1}) = \\ &= \frac{1}{2}\left(\mathbf{F}(\mathbf{U}_{i+1}) - \mathbf{F}(\mathbf{U}_i) - \mathbf{G}_{i+1/2}\left(H_{i+1} - H_i\right)\right) \\ &\pm \frac{1}{2} Q_{i+1/2}\left((\mathbf{U}_{i+1} - \mathbf{U}_i) - A^{-1}_{i+1/2}\mathbf{G}_{i+1/2}\left(H_{i+1} - H_i\right)\right), \end{aligned}$$

where

$$\mathbf{G}_{i+1/2} = \begin{pmatrix} 0 \\ g h_{i+1/2} \end{pmatrix},$$

and

$$A_{i+1/2} = \begin{pmatrix} 0 & 1 \\ u^2_{i+1/2} + g h_{i+1/2} & 2u_{i+1/2} \end{pmatrix}$$

is the Roe Matrix associated to the flux $\mathbf{F}(\mathbf{U})$ from the NSWE, being

$$h_{i+1/2} = \frac{h_i + h_{i+1}}{2}, \; u_{i+1/2} = \frac{u_i\sqrt{h_i} + u_{i+1}\sqrt{h_{i+1}}}{\sqrt{h_i} + \sqrt{h_{i+1}}}.$$

GPU HACKATHON

Finite Volume

$Q_{i+1/2}$ is the viscosity matrix associated to the numerical method. For PVM schemes, $Q_{i+1/2}$ is obtained by a polynomial evaluation of the Roe Matrix. In this work, the viscosity matrix is defined as

$$Q_{i+1/2} = \alpha_0 Id + \alpha_1 A_{i+1/2},$$

being,

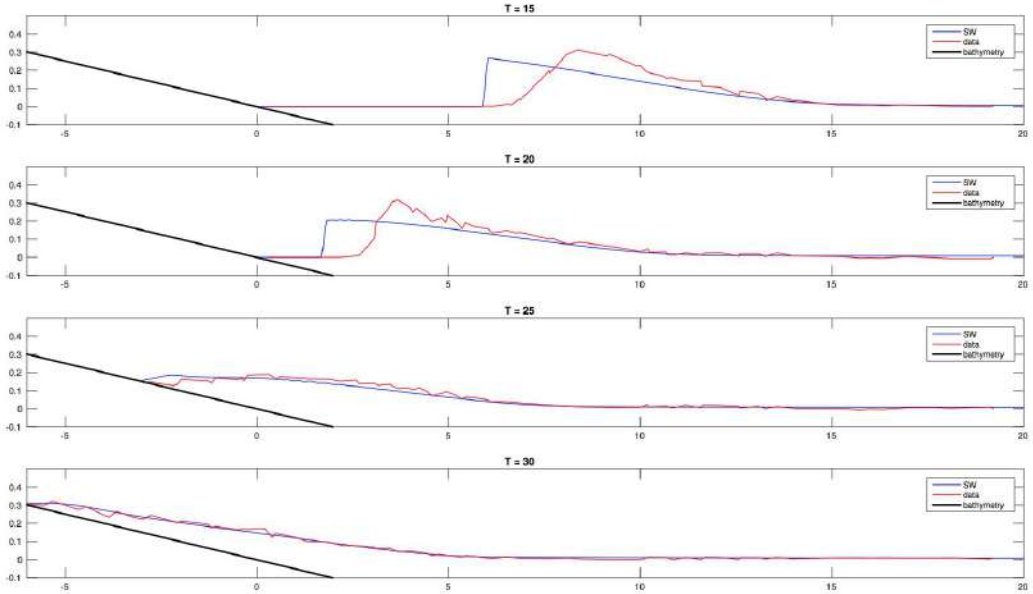
$$\alpha_0 = \frac{S_R|S_L| - S_L|S_R|}{S_R - S_L}, \quad \alpha_1 = \frac{|S_R| - |S_L|}{S_R - S_L},$$

where

$$S_L = \min\left(u_{i+1/2} - \sqrt{gh_{i+1/2}}, u_i - \sqrt{gh_i}\right),$$

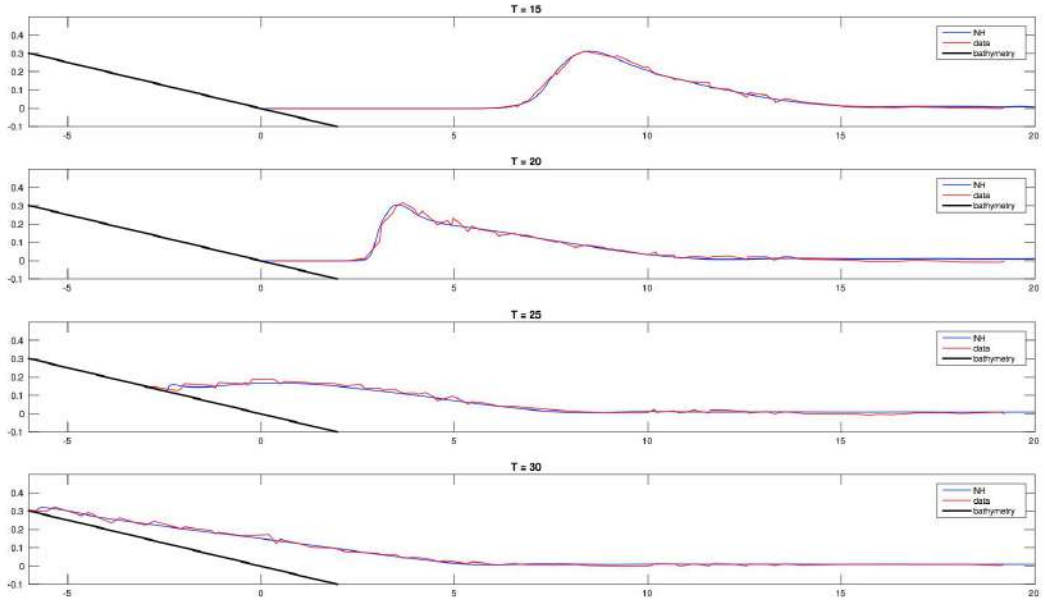
$$S_R = \max\left(u_{i+1/2} + \sqrt{gh_{i+1/2}}, u_{i+1} + \sqrt{gh_{i+1}}\right).$$

Runup of a solitary wave on a plane slope



[4] Castro M.J., Escalante C., Macias J., and Ortega S. "Performance benchmarking of Tsunami-HySEA model for NTHMP's inundation mapping activities". In: *Submitted to Pure and Applied Geophysics* ().

Runup of a solitary wave on a plane slope



[4] Castro M.J., Escalante C., Macias J., and Ortega S. "Performance benchmarking of Tsunami-HySEA model for NTHMP's inundation mapping activities". In: *Submitted to Pure and Applied Geophysics* ().

References I

- Adsuara, J.E., I. Cordero-Carrión, P. Cerdá-Durán, and M.A. Aloy. "Scheduled Relaxation Jacobi method: Improvements and applications". In: *Journal of Computational Physics* 321 (2016), pp. 369–413.
- Beji, S. and J.A. Battjes. "Experimental investigation of wave propagation over a bar". In: *Coastal Engineering* 19 (1993), pp. 151–162.
- Bristeau, M.-O., A. Mangeney, J. Sainte-Marie, and N. Seguin. "An energy-consistent depth-averaged Euler system: Derivation and properties". In: *Discrete and Continuous Dynamical Systems Series B* 20.4 (2015), pp. 961–988.
- Castro, M.J., C. Escalante, E.D. Fernández-Nieto, T. Morales, and G.Narbona-Reina. "A Non-Hydrostatic Bilayer Model for Bed–Load Sediment Transport That Tends to Classical SVE Model for Small Morphodynamic Time". In: *Preprint* ().
- Castro, M.J., C. Escalante, and T. Morales. "Non-Hydrostatic Pressure Shallow Flows: GPU Implementation Using Finite–Volume and Finite–Difference Scheme". In: *Submitted to Applied Mathematics and Computation* ().

References II

- Madsen, P.A. and O.R. Sorensen. "A new form of the Boussinesq equations with improved linear dispersion characteristics. Part 2: A slowing varying bathymetry". In: *Coastal Engineering* 18 (1992), pp. 183–204.
- M.J., Castro, Escalante C., and Macias J. "Performance assessment of Tsunami-HySEA model for NTHMP tsunami currents benchmarking. Part I Lab data". In: *Submitted to Coastal Engineering* ().
- M.J., Castro, Escalante C., Macias J., and Ortega S. "Performance benchmarking of Tsunami-HySEA model for NTHMP's inundation mapping activities". In: *Submitted to Pure and Applied Geophysics* ().
- Peregrine, D.H. "Long waves on a beach". In: *Fluid Mechanics* 27.4 (1967), pp. 815–827.
- Synolakis, C.E. "The runup of solitary waves". In: *Fluid Mechanics* 185 (1987), pp. 523–545.
- Yamazaki, Y., Z. Kowalik, and K.F. Cheung. "Depth-integrated, non-hydrostatic model for wave breaking and run-up". In: *Numerical Methods in Fluids* 61 (2008), pp. 473–497.
- Yang, Xiyang I.A. and Rajat Mittal. "Acceleration of the Jacobi iterative method by factors exceeding 100 using scheduled relaxation". In: *Journal of Computational Physics* 274 (2014), pp. 695–708.

GPU HACKATHON

ANALYSIS (DRAFT)

1. Algorithmic features
2. Parallel design patterns
3. Implementation features

Algorithmic features

- Spatial discretization?
  - Finite differences?
  - Finite elements?
  - Finite volumes?
  - Type of elements: triangle, square?
- Type of solver?
  - Direct vs Iterative (convergence criteria)?
  - Adaptive discretization of the domain?
  - Sparse/Irregular computations (e.g. sparse matrices -CRS-)?
  - Based on solving systems of linear equations?
- Compute-bounded?
- Memory-bounded?

Parallel design patterns

- Fully parallel loops?
- Parallel scalar reductions?
- Parallel sparse reductions?

Implementation features

- Shape of the data structure?
  - Pointer-based vs Array-based?
  - Recursive data structs (e.g. linked-list, tree)?
  - Array of structs (AoS) vs Struct of Arrays (AoS)?

GPU HACKATHON

#gpucesgahack

29 mayo - 1 junio
Santiago de Compostela

TEAM 4

May, 30th

Sergio Ortega Acosta
Ernesto Guerrero Fernández

UNIVERSIDAD DE MÁLAGA

EDANYA

GPU HACKATHON

Algorithmic features

- Spatial discretization?
  - Finite Volume.

- Type of solver?
  - Adaptive discretization of the domain? No, fixed mesh.
  - Sparse/Irregular computations? Yes.
  - Based on solving systems of linear equations? No.

Algorithmic features

- Found “critical” functions.
- Found loops with more workload.
- List of variables of the code, so as to distinguish among private and global ones.

Profiling

Flat profile:

Each sample counts as 0.01 seconds.

% time	cumulative seconds	self seconds	calls	self ms/call	total ms/call	name
32.31	0.73	0.73	2984982	0.00	0.00	calcula_Dmp(Eigen::M
21.68	1.22	0.49	2982000	0.00	0.00	reconstruccion_o2(E
19.03	1.65	0.43	994	0.43	2.24	calcula_iteracion(E
9.29	1.86	0.21	2982	0.07	0.28	reconstruye(Eigen::M
5.31	1.98	0.12	2982	0.04	0.04	actualiza(Eigen::Mat
3.10	2.05	0.07	17903928	0.00	0.00	velinterfaz(Eigen::M
2.66	2.11	0.06	17892000	0.00	0.00	minmod(double, doub
1.77	2.15	0.04	5964000	0.00	0.00	transferencia(double

index	% time	self	children	called	name
					<spontaneous>
[1]	100.0	0.00	2.26		main [1]
		0.43	1.80	994/994	calcula_iteracion(E
		0.00	0.02	994/994	calcula_dt(Eigen::M
		0.01	0.00	11/11	escribe_sol(int, in
		0.00	0.00	1000/995000	Smax(Eigen::Matrix<
		0.00	0.00	1000/5970964	filtro_estado(Eigen
		0.00	0.00	1000/1000	h_inicial_5p(double
		0.00	0.00	1000/1000	batimetria_5p(doubl
		0.00	0.00	12/12	void* Eigen::intern
		0.00	0.00	2/2	Eigen::PlainObjectB
		0.00	0.00	1/1	entrada_datos(doubl
		0.00	0.00	1/1	crea_ficheros(int,

		0.43	1.80	994/994	main [1]
	98.7	0.43	1.80	994	calcula_iteracion(Eigen
		0.73	0.12	2984982/2984982	calcula_Dmp(Eigen::
		0.21	0.61	2982/2982	reconstruye(Eigen::
		0.12	0.00	2982/2982	actualiza(Eigen::Ma
		0.01	0.00	5969964/5970964	filtro_estado(Eigen

		0.73	0.12	2984982/2984982	calcula_iteracion(E
	37.5	0.73	0.12	2984982	calcula_Dmp(Eigen::Matr
		0.05	0.00	11939928/17903928	velinterfaz(Eigen
		0.02	0.00	5969964/5969964	flujoC(double, Eige
		0.02	0.00	2984982/2984982	wetdry(double, doub
		0.01	0.00	5969964/5969964	flujoW(Eigen::Matri
		0.01	0.00	5969964/5969964	flujoG(double, doub
		0.01	0.00	2984982/2984982	presion(double, dou

		0.21	0.61	2982/2982	calcula_iteracion(E
[4]	36.4	0.21	0.61	2982	reconstruye(Eigen::Matr
		0.49	0.12	2982000/2982000	reconstruccion_o2(E

		0.49	0.12	2982000/2982000	reconstruye(Eigen::
[5]	27.1	0.49	0.12	2982000	reconstruccion_o2(Eigen
		0.06	0.00	17892000/17892000	minmod(double, do
		0.04	0.00	5964000/5964000	transferencia(doubl
		0.02	0.00	5964000/17903928	velinterfaz(Eigen:

GPU HACKATHON

Compilation

We have deleted the -g option from compilation.

Compilation option	Execution time
- -O1	21.86 sec
- -O2	21.24 sec
- -O3	12.31 sec
- -Ofast	12.29 sec
- + -march=native	6.9 sec

GPU HACKATHON

#gpucesgahack

29 mayo - 1 junio
Santiago de Compostela

TEAM 4

May, 31th

Sergio Ortega Acosta
Ernesto Guerrero Fernández

UNIVERSIDAD DE MÁLAGA

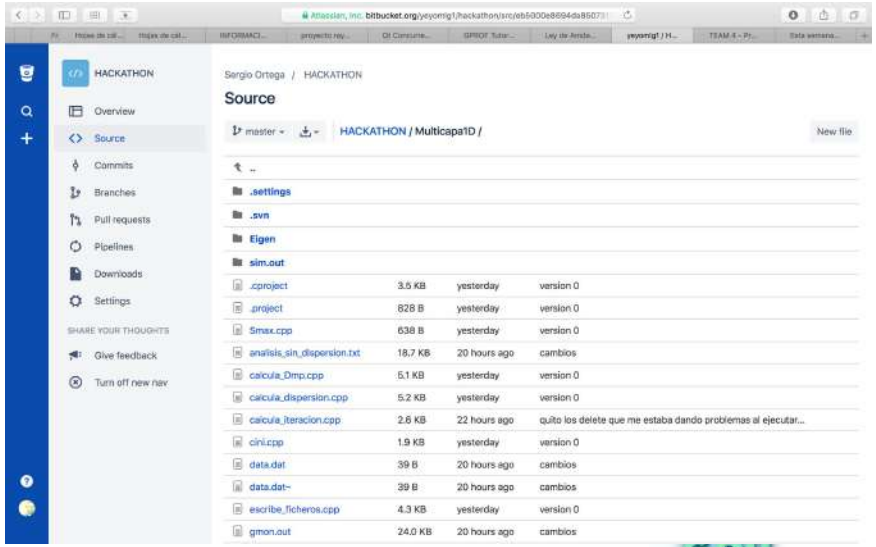
EDANYA

GPU HACKATHON

Bitbucket.org

Atlassian Bitbucket

We have prepared a repository in Bitbucket in order to be able to work collaboratively.



GPU HACKATHON

Pseudo-Code

MAIN

```
do {
  var1 = calcula_iteracion(var0,dt);
     dt = calcula_dt(var0);
} while(tiempo_actual<tiempo_final);
```

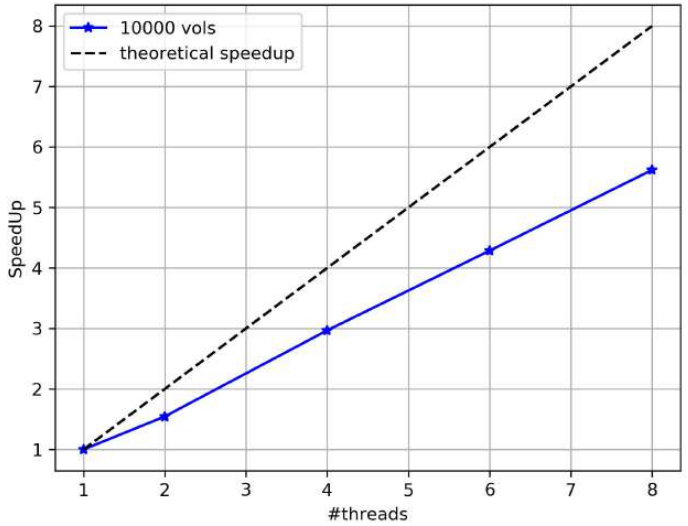
```
void calcula_iteracion(var0) {

    for (int l=0; l<orden; l++){
        var1 = 0;
        reconstructions,var1=reconstruye(var0);
        for (int i=0; i<npar; i++) {
            im = f(i)
            ip = g(i)
            fluxesm,fluxesp=calcula_Dmp(reconstructions,ncapas,dt,heps);
            var1[im]-=fluxesm
            var1[ip]-=fluxesp
        }
        for (int i=0; i<npar; i++) {
            var1 = rk(var0,var1,varn); //Runge-Kutta
        }
    }
}
```

```
#pragma omp parallel shared(...) private(...)
{
#pragma omp for
for (int i=0; i<npar; i++) {
        ...
        fluxesm,fluxesp=calcula_Dmp(reconstructions,ncapas,dt,heps);
        #pragma omp atomic update
        var1[im]-=fluxesm
        #pragma omp atomic update
        var1[ip]-=fluxesp
}
#pragma omp for
for (int i=0; i<npar; i++)
        var1 = rk(var0,var1,varn); //Runge-Kutta
}
```

SPEEDUP

Num_threads	Execution time
1	65.28982 sec
2	42.07812 sec
4	19.22562 sec
8	13.14961 sec



GPU HACKATHON

#gpucesgahack

29 mayo - 1 junio
Santiago de Compostela

TEAM 4

June, 1st

Sergio Ortega Acosta
Ernesto Guerrero Fernández

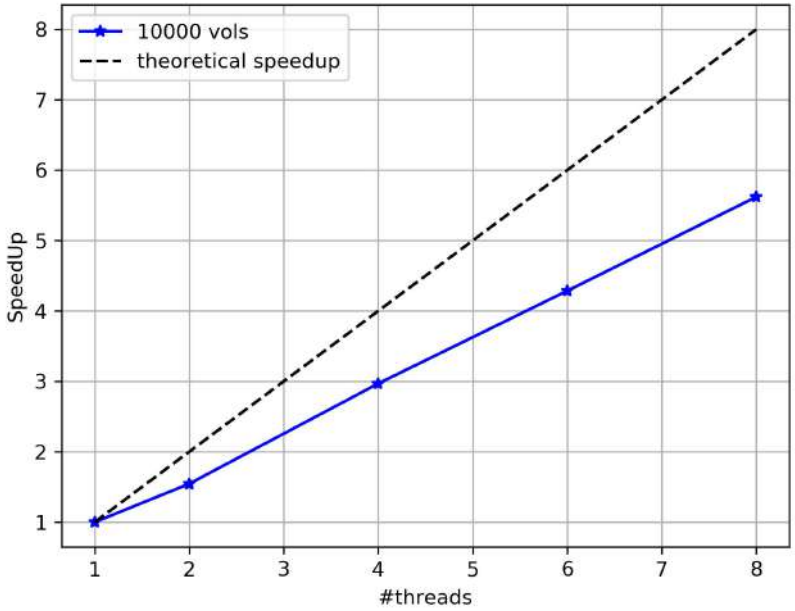
UNIVERSIDAD DE MÁLAGA

EDANYA

GPU HACKATHON

PROOF OF CONCEPT

#threads	Execution time (secs)
1	65.28982
2	42.07812
4	19.22562
6	13.14961
8	10.02978

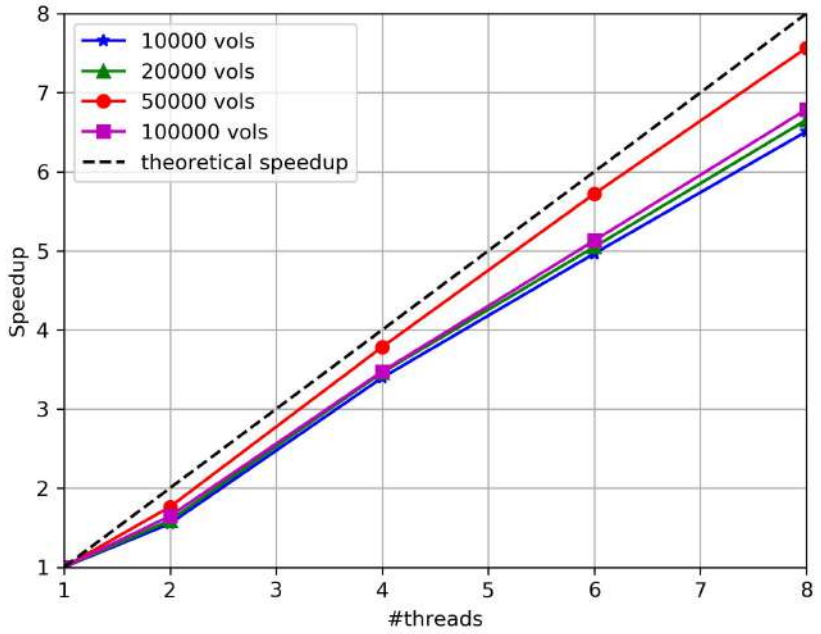


Problem size = #iter x order x (#vols/#threads) = 994 x 3 x (1000/#threads)

GPU HACKATHON

PERFORMANCE WITH BIGGER PROBLEM SIZES

#threads	10000 vols	20000 vols	50000 vols	100000 vols
1	65.28	260.03	1839.99	7276.63
2	42.07	163.56	1043.5	4420.28
4	19.22	75.11	486.22	2093.91
6	13.15	51.5	321.65	1418.57
8	10.03	39.07	243.21	1072.03



GPU HACKATHON

TODO LIST

- Parallelization OpenMP
  - Reconstruction → Fully parallel loop
  - Computation of fluxes → Done (parallel sparse reduction)
  - Linear combination (next state) → Fully parallel loop
  - Copy back → Fully parallel loop
- Minimize overhead accross parallel regions
  - 1 parallel region
    - Reconstruction+Computation of fluxes+Linear combination+Copy back
- Parallelization with OpenACC for GPU
- Parallelization with OpenMP+OpenACC

GPU HACKATHON