https://studiumeditore.it/riviste/coastal-and-offshore-science-and-engineering/
https://doi.org/10.53256/COSE_220202

Modeling passive tracers in the marine environment with high performance heterogeneous and hierarchical computing

D. Di Luccio[a,*], C. G. De Vita[a], G. Mellone[a], Enrico Zambianchi[a], and R. Montella[a]

[a] Department of Science and Technologies, University of Naples "Parthenope", Naples, Campania, Italy

[*] Corresponding author

ABSTRACT

The coastal area is a critical environment primarily for economic and social activities. However, these activities may, at the same time, yield several benefits while also having devastating effects on natural habitats. While the use of High-Performance Computing (HPC) for precise and accurate air quality forecasts is a common issue, similar services devoted to marine pollution in coastal areas remain a challenge. This paper presents Water quality Community Model Plus Plus (WaComM++), a Lagrangian model dedicated to simulating marine pollutants transport and diffusion processes, leveraging a parallelization schema enabling the users to run it on heterogeneous parallel architectures. We presented the model in a framework of a real-world application for pollutants forecasts in the Gulf of Naples (Campania, Italy).

KEYWORDS

Hierarchical parallelism, Environmental modeling, Lagrangian model, Multi-GPUs, Marine pollution forecast.

1 INTRODUCTION

Human activities are concentrated in the coastal area, which is, at the same time, one of the most vulnerable (Anfuso et al. 2021) and most critical environments for economic and social development. These activities can have devastating effects due to i) overexploitation of resources; ii) more or less irreversible damage due to pollution; iii) depletion of biodiversity (Lindeboom 2002). While the meteo-marine forecasts services (Di Luccio et al. 2018) have now reached spatial and temporal resolutions unthinkable until a few decades ago (Benassai & Ascione 2006), and the tools for the analysis of what-if scenarios are already available to decision-makers to the forecasting or reconstruction of events, similar services devoted to marine pollution in coastal areas (Montella et al. 2016a, Di Luccio et al. 2019) are still an open issue.

Data availability as initial and boundary conditions for numerical models was the main limitation in the past. However, today the affordable availability of computational resources could be the limiting factor.

Facing the pollutants like oil spill and organic matter as Lagrangian inert tracers floating in a computational domain set the need to simulate the movement of several particles. In this scenario, the computing problem is characterized by a magnitude of millions or billions of particles mapped at resolutions in the range of hundreds of meters to tens of meters (Hunter 1987).

The lack of availability of dedicated on-premises High-Performance Computing (HPC) resources can be mitigated by leveraging public cloud services (Li et al. 2011). In this scenario, heterogeneous computing based on various accelerators as, but not limited to, general-purpose GPUs arose, making the cost per core compatible with decision-makers budgets. Nevertheless, applications have to be designed (or re-designed) in order to combine different parallel paradigms and the corresponding computing frameworks enforcing the hierarchical parallelization approach (Álvarez-Farré et al. 2021).

This paper presents Water quality Community Model Plus Plus (WaComM++), a Lagrangian model dedicated to simulating marine pollutants transport and diffusion processes. WaComM++ is a model component of the operational model chain at the Center for Monitoring and Modelling Marine and Atmosphere applications (CMMMA[1]) of the University of Naples "Parthenope", based on the DagOnStar[2] workflow engine (Montella et al. 2018, Sánchez-Gallegos et al. 2019).

The WaComM++ system can be used in different ways: (i) an *ex-ante* decision-support tool, for example to aid in the selection of the best suitable areas for farming activity deployment; (ii) an ex-post simulation tool for improving the management of offshore activities.

WaComM++ supports three levels of hierarchical parallelization: (i) the distributed memory enforced by the use of the Message Passing Interface (MPI) library (Walker & Dongarra 1996); (ii) the shared memory paradigm to leverage on the modern multicore architectures thanks to the OpenMP library (OpenMP Architecture Review Board 2015); (iii) the multi-GPU computing implemented with the NVIDIA CUDA toolkit (Rustico et al. 2012).

In this study, we present the parallelization of WaComM++ using each level individually and a combination of them.

The rest of the paper is organized as follows: Section 2 is about research products related to this paper.

Section 3 provides an overview of the Lagrangian approach. Section 4 is devoted to the description of the design and the implementation of the parallel schema, while Section 5 details the configured actual case application and the overall workflow. Finally, discussion, conclusions, and the future research directions are in Section 6.

[1] https://meteo.uniparthenope.it

[2] http://github.com/dagonstar

2 RELATED WORK

The physical transport in the ocean is characterized by a large number of multiscale dynamical features. Advection and diffusion processes have been intensively studied using Lagrangian instruments (Davis 1991, Rossby 2007) or, more recently, the trajectory pair dispersion approach (Corrado et al. 2017). Another work studies the phytoplankton dynamics in coastal areas integrating the ecological information with the land-based remote observations of surface circulation (Cianelli et al. 2017), also utilized to improve search and rescue and oil spill response (Bellomo et al. 2015). Lagrangian particles models in atmospheric and marine environments have been proposed to study the turbulent dispersion of pollutants. This kind of approach has been used in both inhomogeneous and non-stationary flow scenarios (Monti & Leuzzi 2010). Many authors proposed their parameterization for standard stochastic models of single-particle dispersion in a two-dimensional turbulence where coherent vortices play a significant role (Pasquero et al. 2001, Periáñez 2020). As suggested in another work, there are large community-based Lagrangian codes (Van Sebille et al. 2018) such as the LAgrangian SEa MODel (LASEMOD), which can simulate the dispersion of a passive pollutant emitted from a stationary point source in the aquatic environment (Monti & Leuzzi 2010), and the "Probably A Really Computationally Efficient Lagrangian Simulator" (PARCELS) (Delandmeter & Van Sebille 2019), recently developed in the framework of the OceanParcels project for computing Lagrangian particle trajectories of passive and active particulates such as plankton (Dämmer et al. 2020), plastic (Van der Mheen et al. 2020), and fish (Schilling et al. 2020). Among others, the single-particle Lagrangian Assessment for Marine Pollution 3 Dimensional model (LAMP3D) has been used to simulate the dispersion of pollutants released from a marine fish farm (Doglioli et al. 2004). The same approach was used then for the design of a decision-making tool to evaluate the concentration of the pollutants in the proximity of mussel farms. Born as an evolution of the Lagrangian Assessment for Marine Pollution 3D (LAMP3D), WaComM uses old hierarchical and heterogeneous parallelization techniques, the predecessor of the present WaComM++. WaComM has been developed matching the hierarchical parallelization design requirements and tested in Intel X86 64 and IBM P8 multi-core environments and integrated with the FACE-IT Galaxy workflow (Montella et al. 2016b). Lagrangian models aim to compute the position changes of many particles fluctuating in a fluid in a typical practical application. However, because the problem size varies during the simulation run, the bottleneck controls the total execution time, which, when too long, limits the use of this approach in an operational fashion. Many authors explain the use of GPUs (Laccetti et al. 2013, Laccetti et al. 2019) to accelerate the Lagrangian model. One example presents the performance of the FLEXPART model using the global memory of Nvidia CUDA-enabled GPUs, evidencing that the use of a Tesla C1060 processor can improve the execution speed about ten times. (Zeng et al. 2010). More recently, the Lagrangian dispersion in the atmosphere has been simulated, presenting the performance of FlexOcl (Harvey et al. 2014). FlexOcl is an OpenCL/C++ version of FLEXCPP (GPU-accelerated version of the FLEXPART simulator) on different hardware platforms from different vendors and evidencing equivalent or better performance compared to its counterpart CUDA implementation.

3 LAGRANGIAN DISPERSION MODEL

The spatial and temporal evolution of pollutants in a fluid environment (just like any fluid property) can be described following two approaches: the Eulerian one, at a fixed point, and the Lagrangian one, or particle-following (Batchelor C. & Batchelor G. 2000). The choice of either one depends on several constraints, as well as on the desired kind of output: the Eulerian models provide concentration maps, while Lagrangian one, ones produce particle trajectories (Buffoni et al. 1997). In principle, the two approaches are equivalent, to the extent that the Eulerian tracer concentration is proportional to the probability of finding tracer particles in a given position at a given time (Csanady 1973). However, the full equivalence would require having a virtually infinite set of (Eulerian and Lagrangian) data. However,

the Lagrangian approach is more suitable for transport studies since dispersion properties can be obtained from single-particle (Taylor 1921) or multiple-particle statistics (Ottino 1989). Recent results of different dispersion regimes based on Lagrangian measurements have been observed in the ocean (Corrado et al. 2017). Moreover, Eulerian regular flow fields may exhibit irregular Lagrangian properties, resulting in an enhanced dispersion not detectable by Eulerian methods. Finally, the particle-tracking models may be more computationally efficient because the computational cost refers only to the region where
the particles are located, unlike the concentration model. Due to the nature of the Lagrangian model, its speed of calculation can be improved using parallel computing techniques. In WaComM++ the motion of a particle follows the equation:

$$\frac{d\mathbf{p}}{dt} = \mathbf{u} \qquad 1$$

(where $\mathbf{p} = (x_p, y_p, z_p)$ is the three-dimensional particle position (Garcıa-Martınez & Flores-Tovar 1999)), with appropriate horizontal and vertical boundary treatment when particles attain the coast (in the horizontal) or the bottom (in the vertical), i.e. when stranding or sedimentation occurs.
Phenomenologically, the instantaneous velocity $\mathbf{u}(\mathbf{p}, t)$ can be thought of as the sum of two components, an advective mean flow $\mathbf{U}(\mathbf{p}(t), t)$ and a diffusive stochastic component $\mathbf{u}'(\mathbf{p}(t), t)$ (Zannetti 1990). Rybalko et al. (Rybalko et al. 2012), provided that a scale separation exists between these two contributions ((e.g., (Holloway 1989, Zambianchi & Griffa 1994, Falco & Zambianchi 2011)). $\mathbf{U}$ represents the velocity field provided by the model, and $\mathbf{u}'$ is a parametrization of the unresolved subgrid-scale processes.
In our model, particles are supposed to be passive, i.e., their presence does not affect the circulation dynamics, but specific properties can be assigned to every particle as the time of decay and the sedimentation velocity.
At the current state of development, WaComM++ deals exclusively with the hypothesis of non-inertial particle motion. The particle position, at the time $t_1 + \Delta t$, s obtained by integrating Eq. 1 with the timestep $\Delta t = t_2 - t_1$:

$$\mathbf{p}(t_1 + \Delta t) = \mathbf{p}(t_1) + \int_{t_1}^{t_1+\Delta t} \mathbf{u}(\mathbf{p}(t), t)dt \qquad 2$$

The displacement at the nth step is thus computed as

$$\mathbf{p}_{n+1} - \mathbf{p}_n = \mathbf{U}_n(\Delta t) + \rho_n \qquad 3$$

In our case, the mean flow portion $\mathbf{U}$ of $\mathbf{u}$ is the model velocity field linearly interpolated in space and time at $\mathbf{p} = (x_p(t), y_p(t), z_p(t))$, while ρ is a random displacement (Pearson 1905, Allen 1982) from a normal distribution with a zero mean and standard deviation $\sigma = \sqrt{2K\Delta t}$, where K is the diffusivity which appears in the Eulerian diffusion equation, i.e. the asymptotic dispersion parameter based on single-particle statistics (for a thorough discussion on the equivalence between the two approaches in this context see (Zambianchi & Griffa 1994)). For each grid point, the σ value is set to decrease with depth as:

$$\sigma(z_n) = \sigma(0)\left(1 + \frac{z_n}{H}\right) \qquad 4$$

where z_n is the vertical coordinate of the particle and H is the local bathymetry value.
It is worth mentioning that this model complies with the criteria defined by (Thomson 1987) for the selection of random-walk models of dispersion in turbulent flows: the well-mixed condition; the small-time behavior of the velocity distribution of particles from a source point; the requirement of compatibility with the Eulerian equations; forward and reverse diffusion; the small-timescale limit.

4 ARCHITECTURE AND PARALLELIZATION SCHEMA

WaComM++ uses a particle-based Lagrangian approach relying on a tridimensional ocean dynamics field produced by a coupled Eulerian ocean model (Doglioli et al. 2004, Di Luccio et al. 2017, Montella et al. 2016b).

WaComM++ has been designed with hierarchical parallelism in mind. Nevertheless, some requirements have been strongly driven by the transport and diffusion Lagrangian model, for example, the need for data exchange using standard and well-known formats. TheWaComM++ overall structure design is shown in Fig. 1.

The ocean state, adapted by the OceanModelAdapter component to be used for the transport and diffusion computation, is described by the variables *u, v, w* (the horizontal and the vertical components of the sea current velocity vector), *zeta* (the sea surface height), and *AKT* (vertical diffusivity for the temperature) at a given time *T* in *K, J, I* position. The overall computation is performed over three nested cycles:

- Ocean state outer cycle: for each time-referenced dataset (typically 1-hour data), a WaComM component is instanced. It is also responsible for managing new emitted particles (sources) and "dead" particles, respectively adding/removing them from the workflow.
- Particles outer cycle: it assigns the particles to process using ocean data. It manages computational data and HPC operations.
- Particles inner cycle: it moves the particles within the considered time slice applying the Lagrangian transport and diffusion equations integrated on a given time step.

The outer and the inner cycles are characterized by strongly time-dependent iteration because the ocean dynamics are provided as domain conditions. However, the particles outer cycle has been parallelized using a hierarchical approach because each particle's movement is independent of the others. Therefore, the OceanModelAdapter component prepares and adapts data coming from ocean simulations or forecasts to the data structures managed by the transport and diffusion model. This kind of data is managed as multidimensional matrices usually stored in NetCDF files. Although most parts of environmental models use the NetCDF CF Conventions and Metadata[3], some processing is needed to prepare the ocean state forWaComM++.

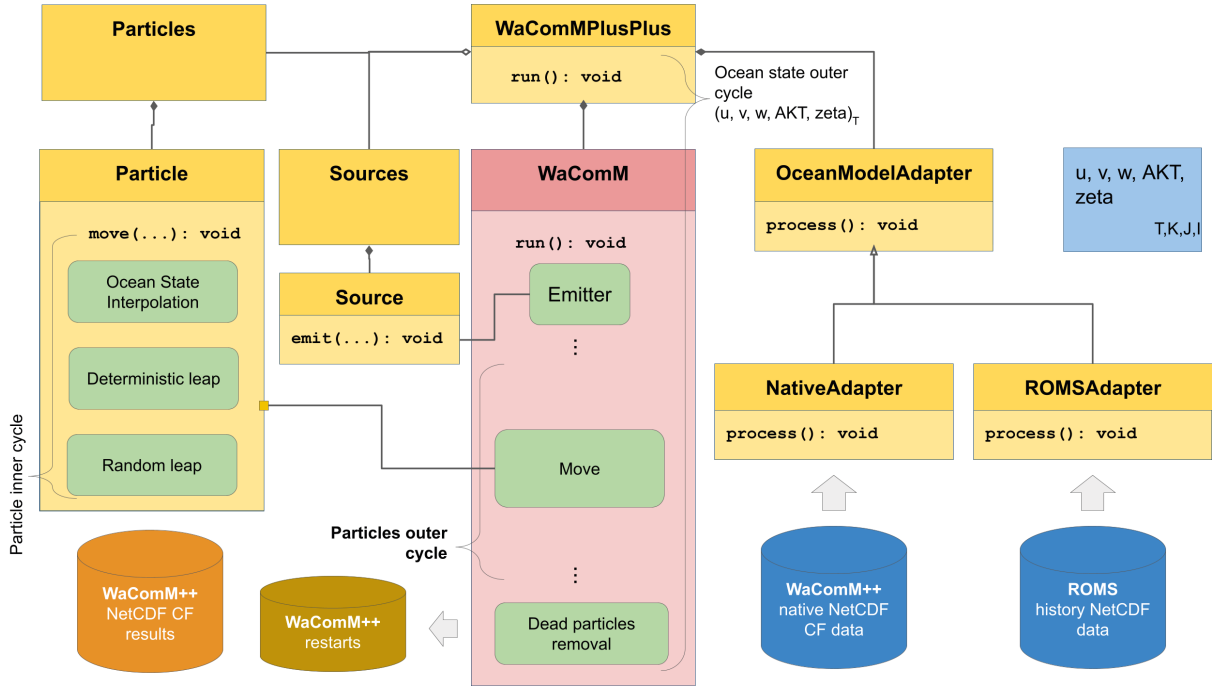


Figure 1. WaComM++ components design and I/O data flow schema.

The Source component implements the sea pollutant emission generator. Each emission source is defined by its position (latitude, longitude, and depth), the number of Lagrangian particles spilled out per hour, and the time interval of the emission. We designed WaComM++ using an Object-Oriented approach (Fig. 1) applied to the physics equation solver embedded in the Particle component. First, each particle is moved using the algorithm implemented in the Particle component This component represents a Lagrangian particle characterized by its *3D position, timestamp of generation, age,* and *health*. The *Particles* component is composed of the *Particle* components, and its duty deals with particle-related I/O operations as the state persistence among two simulations (restarting).

The WaComMPlusPlus component aggregates OceanModelAdapter, Sources, and Particles. This component manages the ocean state outer cycle providing, for each time-related dataset, theWaComM component with the momentum (u, v, and w components of the current sea vector), the variation of the sea surface height (zeta), and the vertical turbulent diffusion (AKT). The actual particle movement is encapsulated in the Particle component. TheWaComM component encapsulates the particles outer cycle of the transport and diffusion model (Montella et al. 2016b, Di Luccio et al. 2017).

For each step of the ocean state outer cycle, for each particle processed by the particles outer cycle, the particle inner cycle begins considering the particle n in its initial position stored in D_n. For each time step t, the particle is checked versus its health condition (Doglioli et al. 2004) and versus the closure constraints, as is the behaviour at the surface, at the bottom, and on the shore; eventually the current position becomes stored in D_n.

If the particle is still *alive*, the new position is computed performing two leaps (moves):

- Deterministic leap: it is driven by the ocean status $(u, v, w, zeta, AKT)_t$ at the given time step t computed interpolating $(u, v, w, zeta, AKT)_T$ on space and time
- Random leap: a stochastic component is added to the deterministic leap.

The new particle position is, again, checked versus the closure constraints. Finally, the particle is aged by the time step interval. Because the particles move independently from the others, therefore no interactions in terms of computation are required between them: the particle inner cycle can be considered as one of many in a many-task parallel paradigm (Diaz et al. 2012).

Our goal was to produce a multi-choice parallel framework, which can be used for the most common parallel architecture to achieve the best performance according to the HPC system available. The implemented parallelization scheme is based on different parallel sub-schemes, which can also be combinable with each other, providing a *hierarchical parallelization schema*. Fig. 2 represents the proposed parallelization schema, the data flow, and the active software components, using or not multi-GPU acceleration.

A standard paradigm in High-Performance Computing is domain decomposition.

The whole problem size (the number of moving particles in the case of Lagrangian modeling) is dived in lots and distributed to several executors. Each executor is an instance of a computer program (process) having in charge of computing the partition of the problem in its duty. Due to CPUs being composed of more computing cores, each process can decompose its part of the problem to each computing core running concurrently (threads). While the threads belonging to the same process share the same memory, processes communicate with each other by exchanging data messages. The general-purpose GPUs (Graphics Processing Units) are a relatively recent element in HPC architecture: designed to perform high-speed computation in real-time interactive graphics (i.e., games, scientific visualization, etc.), are used to accelerate the math computation in a performance and power consumption effective way. The problem size in charge of each thread can be decomposed again and distributed on each available GPU. As demonstrated later in the paper, the complete availability of the three levels of hierarchical problem decomposition makes the overall computing performance remarkable as the problem size increases

[3] https://cfconventions.org

consistently. In details, considering a shared and distributed memory scenario with at least one GPU device enabled, *np* represents the number of available processors, *nt* the number of threads used on each processor, and *ng* the number of GPU devices on each processor.

Assuming P_0,P_1, ...,P_{np} as a homogeneous set of heterogeneous computational resources, analysing Fig. 2, in P_0 the processor-level behavior is detailed and in P_1 how the computation is concurrently performed on *nt* threads and *ng* GPUs.

The domain decomposition is performed by P_0 dividing the particles at the ocean state time *T* in subsets sized as Δ0,Δ1, ...,Δ*np*. The subsets are distributed to each processor P_p. Each processor P_p has its local particles data ${}_pD$. For each processor P_p, the local particles dataset is divided in subsets sized as Δ0,Δ1, ...,Δ*nt* and assigned at each thread $t \in [0, nt]$. For each thread T_t, if there is no GPU device available, an algorithm is executed sequentially for all local thread particle data ${}_tD_n \in {}_pD_t$. Otherwise, if at least one GPU device is available, for each thread *Tt*, the local particles dataset is, again, split in subsets sized as Δ0,Δ1, ...,Δ*ng* and assigned to each GPU $g \in [0, ng]$, working with *CUDA Multi-threading*, then ocean state, seafloor depth and particles data are copied from the host to the GPU. The P_0 cycle ends gathering updated particles data from each processor and removing dead particles using the approach described in Doglioli et al. (Doglioli et al. 2004).

The parallelization schema (Fig. 2) enables the final user to choose any combination of the following execution modes:

- Single run: the single process P_0, calling the procedure move solves sequentially the problem for all particles of its domain D_T.
- Distributed memory run on np processes: each process P_p, $p \in [0, np]$, calling the procedure move , solves the problem for all particles of its sub-domain ${}_pD$.
- Shared memory runs on *nt* threads: using the shared distribution paradigm, a sub-domain of date *tD* is assigned at each thread of the multi-core environment. Each thread for $t \in [0, T]$, works on a sub-set *tD*, calling the procedure move.
- Heterogeneous Single/Multiple GPUs run: each thread *t* splits its domain *tD* in a sub-domain of data *gD* assigned to each GPU of the environment. Each GPU for $g \in [0, G]$ works on the subset *gD*.

Our hierarchical implementation expects that the last active hierarchical level processes particles at the current development stage.

5 REAL CASE APPLICATION AND PERFORMANCE EVALUATION

We present a real-world use case application of WaComM++ as a scientific workflow component to simulate tracers' sea dispersion (Fig. 3). The numerical results were compared with the Lagrangian data collected with a drifter buoy (Fig. 4) in the framework of the ABBaCo Project, founded by the Italian Ministry of University and Research - FISR in agreement with the Stazione Zoologica "A. Dohrn" (Castagno et al. 2020). As mentioned above, this workflow is operational at CMMMA.

The above-mentioned scientific workflow starting point is the Weather Research, and Forecasting (WRF) model, a mesoscale numerical weather prediction system designed for both atmospheric research and operational forecasting applications (Di Luccio et al. 2020b, Di Luccio et al. 2020c). Our implementation is based on a three telescopic two-way nested computational domain designed to reach the ground resolution of 1 Km in southern Italy. In contrast, 5 Km is the resolution in Italy and surrounding seas, and 25 Km on the Euro-Mediterranean area. WRF atmospheric initial and boundary conditions were provided by Global Forecast System (GFS), a weather forecast model on the entire globe produced by the National Centers for Environmental Prediction (NCEP), at a 0.50° horizontal resolution.

WRF model results are made available hourly as initial conditions (atmospheric forcing) to the Regional Ocean Model System (ROMS) coupled model. The ROMS configuration provides the sole grid domain covering, at the horizontal resolution of about 160 m, the geographic area between the southern Lazio Region and the northern Calabria Region (Italy).

ROMS initial and boundary conditions were

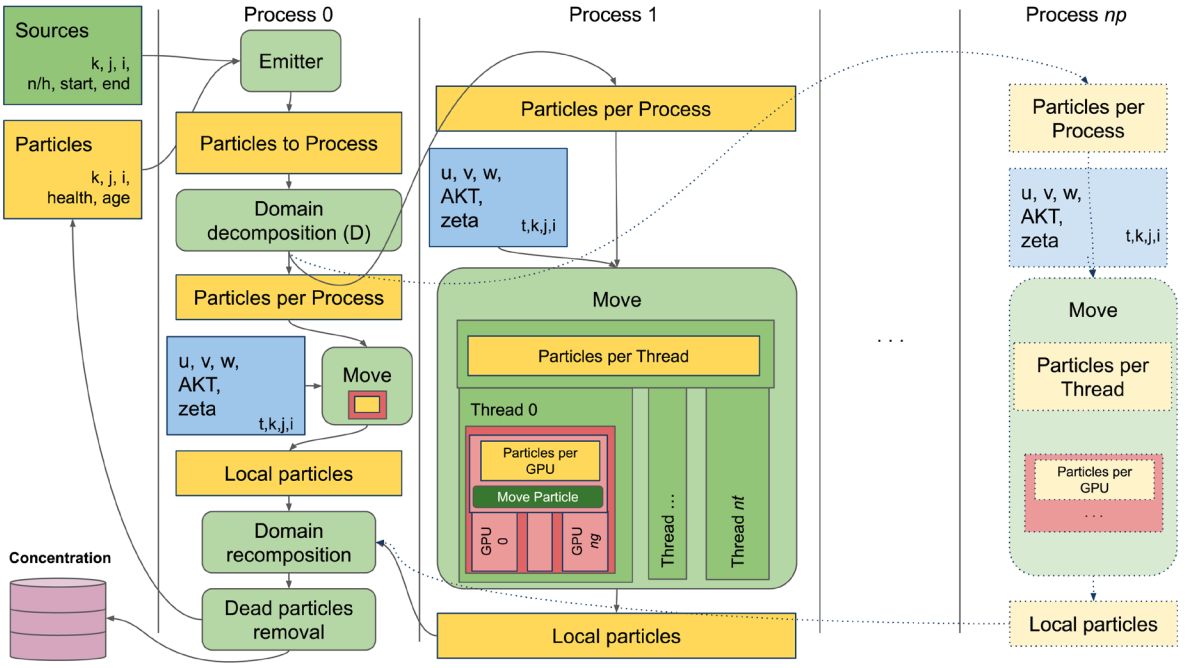


Figure 2. WaComM++ hierarchical parallelization schema using GPU paradigm.

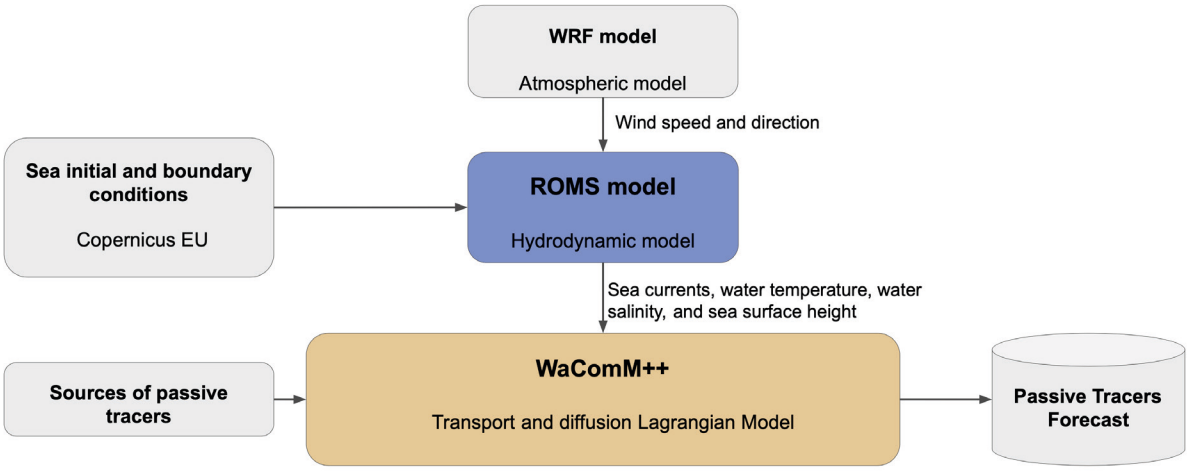
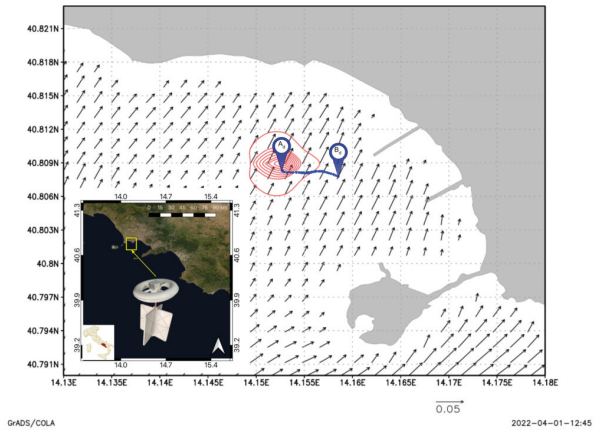
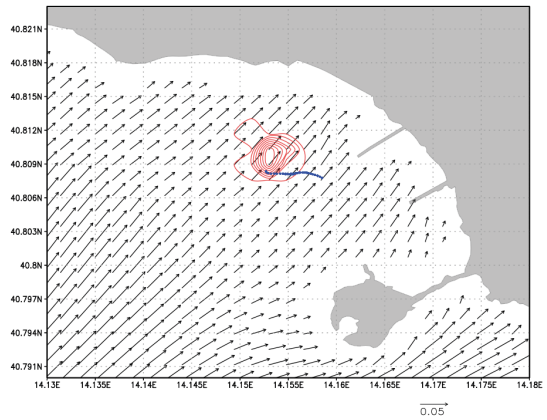


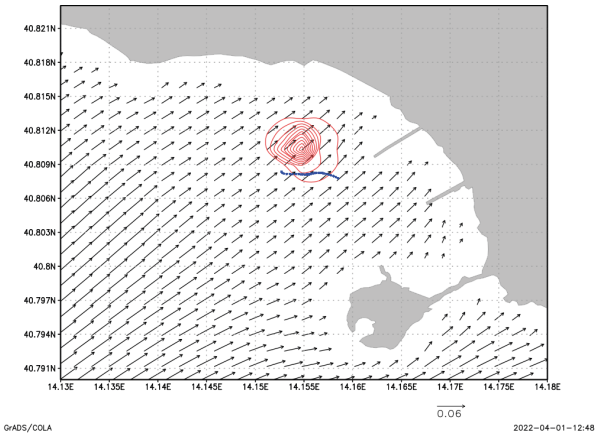
Figure 3. Block diagram of the configured use case (a). The atmospheric model WRF, the hydrodynamic model ROMS, and the Lagrangian model WacomM++ are involved in the framework of the CMMMA service at the University of Naples "Parthenope".



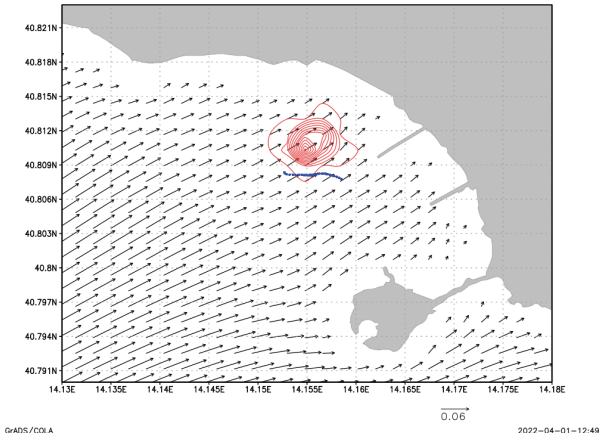
(a) *Simulation at time 09:00 UTC*



(b) *Simulation at time 10:00 UTC*



(c) *Simulation at time 11:00 UTC*



(d) *Simulation at time 12:00 UTC*

Figure 4. Wacomm++ lagrangian particles (red) and drifter buoy (blue) movement (West to East) under the sea state characterized by ROMS surface currents coming from SW (oriented vectors).

provided in the framework of the Copernicus project[4] with the horizontal grid resolution equal to 1/24° and 141 unavely spaced vertical levels.
The three-dimensional predictions of sea currents, water temperature, water salinity, and sea surface height are stored and used to force the WaComM++ coupled model. WRF-ROMS-WaComM++ hourly workflow results are stored and made available online, enriched with diagnostics (i,e. cloud fraction, 2 m relative humidity, updraft helicity), for science and engineering applications.

In this application, we consider a problem size consisting of a total of 20K particles spilled out by a single coastal source located in the Gulf of Pozzuoli (Campania, Italy), simulating the movements of a drifter buoy without the use of an *ad-hoc* configured restart file. The total simulation time is 4 hours.
As mentioned before, we compared the numerical results with a drifter path (Castagno et al. 2020) released in point A_d (40.808380 N, 14.152580 E) at 08:48 UTC and rescued in B_d (40.807620 N, 14.158720 E at 11:18 of 1st April 2019. Figure 4 shows the course of the drifter (blue) and the course of the particles simulated using WaComM++ mode (red). As evinced from the figures, the path of the simulated particles is consistent with the Eastern direction followed by the drifter buoy.

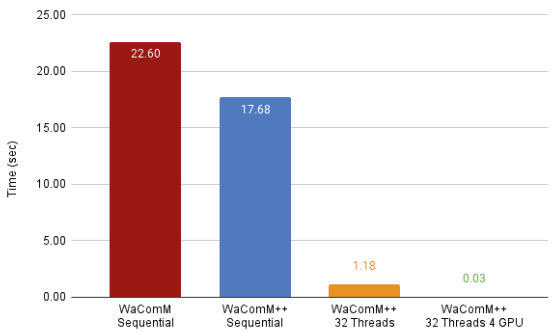


Figure 5. Particles Inner Cycle results comparing between WaComM and WaComM++.

To test the WaComM++ performance, we set up a testbed experiment using one computing node of the *purpleJeans*[5] HPC system equipped with two Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz (16 cores each) and 4 NVidia Tesla V100-SXM2-32GB (5120 CUDA cores each). We tested WaComM++ in an actual test case application involving all the I/O data flow (Fig. 1), and we compared results with WaComM, sharing the same configuration.

The same simulation was also performed on the old WaComM software and then computed using diverse WaComM++ configurations to compare and discuss results. Although WaComM and WaComM++ are two different types of software, we chose a complete sequential configuration without any form of parallelization to define a baseline. Then we extended the evaluation by considering the following configurations: i) a configuration using all available threads (32) on the same computing node; ii) a configuration using all the available threads (32) and all available GPU accelerators on a single computing node (4). From the tests carried out, as shown in Fig.5, a clear improvement of the new software is evident, especially using shared and distributed memory and GPUs. In particular, the sequential configuration of both types of software brings an improvement of about 22%.

Furthermore, while using shared memory, there is an improvement making the situation about 19 times better, and, finally, using all the computing power provided by a single purpleJeans node, a 750 times better result is obtained. It should be further emphasized that these results are achieved by comparing the Particles Inner Cycle, which is the portion of software that deals exclusively with simulating the movement of particles using a Lagrangian model.

6 DISCUSSION AND CONCLUSION

This paper presented WaComM++, a hierarchical heterogeneous Lagrangian model for marine pollutants transport and diffusion.

Although WaComM++ shares with its previous incarnation, WaComM, the same physics approach, it has been wholly redesigned leveraging three levels of parallelism.

It works with MPI distributed memory, OpenMP shared memory and GPU CUDA acceleration without any limitation regarding which parallelization technology must be used. This feature makes WaComM++ ideal for real-world operational applications where the availability of a dedicated high-end supercomputer is not the usual scenario.

As with all ongoing projects, WaComM++ is worthy of possible improvements Firstly, the particle motion can be improved by considering the wave-induced currents that have an essential effect on the particle trajectory, particularly when the particle source is located nearshore. Moreover, the physical model can be improved considering other particle features and behavior as, but not limited to, the flotation and the sinking. The emission model can be improved with more refined control over the emission rate. Finally, a different algorithm for particle concentration on Eulerian grids could produce higher resolution results at the exact computational costs. Crowdsourced marine data could significantly improve the model results by leveraging more detailed bathymetry (Montella et al. 2019, Di Luccio et al. 2020a).

The current parallelization scheme is based on using computational resources available locally. However, even if this implies obtaining exceptional performance, future research goals can move towards cloud computing and towards all those computational resources available *on-demand* which offer a certain kind of *malleability.*

[4] https://doi.org/10.25423/CMCC/MEDSEA ANALYSIS FORECAST PHY_006_013_EAS6

[5] https://rcf.uniparthenope.it

WaComM++ is available as open-source and can be considered mature enough for production[6].

ACKNOWLEDGMENTS

This study is part of the research agreements MytilAI (CUP I65F21000040002) and ADMIRE (956748-ADMIREH2020-JTI-EuroHPC-2019-1).
This study was partly supported by the 2017 PRIN project EMME (Exploring the fate of Mediterranean microplastic: from distribution pathways to biological effects), funded by the Italian Ministry for Research (grant agreement no. 2017WERYZP).

REFERENCES

Allen, C. M. (1982). Numerical simulation of contaminant dispersion in estuary flows. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences 381*(1780), 179–194.

Álvarez-Farré, X. et al. (2021). A hierarchical parallel implementation for heterogeneous computing. application to algebra-based cfd simulations on hybrid supercomputers. *Computers & Fluids 214*, 104768.

Anfuso, G. et al. (2021). Coastal sensitivity/vulnerability characterization and adaptation strategies: *A review. Journal of Marine Science and Engineering 9*(1), 72.

Batchelor C., K. & K. Batchelor G. (2000). *An introduction to fluid dynamics*. Cambridge university press.

Bellomo, L. et al. (2015). Toward an integrated hf radar network in the mediterranean sea to improve search and rescue and oil spill response: the tosca project experience. *Journal of Operational Oceanography 8*(2), 95–107.

Benassai, G. & I. Ascione (2006). Implementation and validation of wave watch iii model offshore the coastlines of southern italy. In *International Conference on Offshore Mechanics and Arctic Engineering,* Volume 47470, pp. 553–560.

Buffoni, G. et al. (1997). Dispersion processes and residence times in a semi-enclosed basin with recirculating gyres: An application to the tyrrhenian sea. *Journal of Geophysical Research: Oceans 102*(C8), 18699–18713.

Castagno, P. et al. (2020). Hydrographic and dynamical characterisation of the bagnoli-coroglio bay (gulf of naples, tyrrhenian sea). *Chemistry and Ecology 36*(6), 598–618.

Cianelli, D. et al. (2017). Disentangling physical and biological drivers of phytoplankton dynamics in a coastal system. *Scientific reports 7*(1), 1–15.

Corrado, R. et al. (2017). General characteristics of relative dispersion in the ocean. *Scientific reports 7*, 46291.

Csanady, G. (1973). Wind-induced barotropic motions in long lakes. *Journal of Physical Oceanography 3*(4), 429–438.

Dämmer, L. K. et al. (2020). Evaluation of isotopes and elements in planktonic foraminifera from the mediterranean sea as recorders of seawater oxygen isotopes and salinity. *Climate of the Past Discussions* 2020, 1–20.

Davis, R. E. (1991). Lagrangian ocean studies. *Annual Review of Fluid Mechanics 23*(1), 43–64.

Delandmeter, P. & E. Van Sebille (2019). The parcels v2. 0 lagrangian framework: new field interpolation schemes. *Geoscientific Model Development 12*(8), 3571–3584.

Di Luccio, D. et al. (2017). Some remarks about a community open source lagrangian pollutant transport and dispersion model. *Procedia computer science 113*, 490–495.

Di Luccio, D. et al. (2018). Wave run-up prediction and observation in a micro-tidal beach. *Natural Hazards and Earth System Sciences 18*(11), 2841–2857.

Di Luccio, D. et al. (2019). Shoreline rotation analysis of embayed beaches by means of in situ and remote surveys. *Sustainability 11*(3), 725.

[4] https://github.com/ccmmmma/wacommplusplus

Di Luccio, D. et al. (2020a). Coastal marine data crowdsourcing using the internet of floating things: Improving the results of a water quality model. *IEEE Access 8*, 101209–101223.

Di Luccio, D. et al. (2020b). Evidences of atmospheric pressure drop and sea level alteration in the ligurian sea. In *2019 IMEKO TC19 International Workshop on Metrology for the Sea: Learning to Measure Sea Health Parameters, MetroSea 2019*, pp. 22–27.

Di Luccio, D. et al. (2020c). An integrated approach of in-situ data, remote sensing measurements and numerical simulations to study storm events in the ligurian sea. *MetroSea 2020 - TC19 International Workshop on Metrology for the Sea*, 28–33.

Diaz, J. et al. (2012). A survey of parallel programming models and tools in the multi and many-core era. *IEEE Transactions on parallel and distributed systems 23*(8), 1369–1386.

Doglioli, A. et al. (2004). Development of a numerical model to study the dispersion of wastes coming from a marine fish farm in the ligurian sea (western mediterranean). *Aquaculture 231*(1-4), 215–235.

Falco, P. & E. Zambianchi (2011). Near-surface structure of the antarctic circumpolar current derived from world ocean circulation experiment drifter data. *Journal of Geophysical Research: Oceans 116*(C5).

Garcıa-Martınez, R. & H. Flores-Tovar (1999). Computer modeling of oil spill trajectories with a high accuracy method. *Spill Science & Technology Bulletin 5*(5-6), 323–330.

Harvey, P. et al. (2014). Accelerating lagrangian particle dispersion in the atmosphere with opencl across multiple platforms. In *Proceedings of the International Workshop on OpenCL 2013 & 2014*, pp. 1–8.

Holloway, G. (1989). Subgridscale representation. In *Oceanic Circulation Models: Combining Data and Dynamics*, pp. 513–583. Springer.

Hunter, J. (1987). The application of lagrangian particle-tracking techniques to modelling of dispersion in the sea. In *North-Holland Mathematics Studies, Volume 145*, pp. 257–269. Elsevier.

Laccetti, G. et al. (2013). A study on adaptive algorithms for numerical quadrature on heterogeneous gpu and multicore based systems. In *International Conference on Parallel Processing and Applied Mathematics,* pp. 704–713. Springer.

Laccetti, G. et al. (2019). An adaptive algorithm for high-dimensional integrals on heterogeneous cpu-gpu systems. *Concurrency and Computation: Practice and Experience 31*(19), e4945.

Li, A. et al. (2011). Comparing public-cloud providers. *IEEE Internet Computing 15*(2), 50–53.

Lindeboom, H. (2002). The coastal zone: an ecosystem under pressure. *Oceans 2020: Science, Trends and the Challenge of Sustainability,* 49–84.

Montella, R. et al. (2016a). Applications of the face-it portal and workflow engine for operational food quality prediction and assessment: Mussel farm monitoring in the bay of napoli, italy. In CEUR *Workshop Proceedings*, Volume 1800, pp. 64–68.

Montella, R. et al. (2016b). Wacomm: A parallel water quality community model for pollutant transport and dispersion operational predictions. In 2016 1*2th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pp. 717–724. IEEE.

Montella, R. et al. (2018). Dagon*: Executing direct acyclic graphs as parallel jobs on anything. In *2018 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*, pp. 64–73. IEEE.

Montella, R. et al. (2019). Workflow-based automatic processing for internet of floating things crowdsourced data. Future Generation Computer Systems 94, 103–119.

Monti, P. & G. Leuzzi (2010). Lagrangian models of dispersion in marine environment. *Environmental fluid mechanics 10*(6), 637–656.

OpenMP Architecture Review Board (2015). OpenMP application program interface version 4.5.

Ottino, J. M. (1989). The mixing of fluids. *Scientific American 260*(1), 56–67.

Pasquero, C. et al. (2001). Parameterization of dispersion in two-dimensional turbulence. *Journal of Fluid Mechanics 439*, 279.

Pearson, K. (1905). The problem of the random walk. *Nature 72*(1867), 342–342.

Periáñez, R. (2020). A lagrangian oil spill transport model for the red sea. *Ocean Engineering 217*, 107953.

Rossby, T. (2007). Evolution of lagrangian methods in oceanography. *Lagrangian Analysis and Prediction of Coastal and Ocean Dynamics*, 1–38.

Rustico, E. et al. (2012). Advances in multi-gpu smoothed particle hydrodynamics simulations. *IEEE Transactions on Parallel and Distributed Systems 25*(1), 43–52.

Rybalko, M. et al. (2012). A lagrangian particle random walk model for hybrid rans/les turbulent flows. *Powder technology 221*, 105–113.

Sánchez-Gallegos, D. D. et al. (2019). A microservice-based building block approach for scientific workflow engines: Processing large data volumes with dagonstar. In *2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS),* pp. 368–375. IEEE.

Schilling, H. T. et al. (2020). Multiple spawning events promote increased larval dispersal of a predatory fish in a western boundary current. *Fisheries Oceanography.*

Taylor, G. I. (1921). Experiments with rotating fluids. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character 100(703)*, 114–121.

Thomson, D. (1987). Criteria for the selection of stochastic models of particle trajectories in turbulent flows. *Journal of fluid mechanics 180*, 529–556.

Van der Mheen, M. et al. (2020). Beaching patterns of plastic debris along the indian ocean rim. *Ocean Science 16*(5), 1317–1336.

Van Sebille, E. et al. (2018). Lagrangian ocean analysis: Fundamentals and practices. *Ocean Modelling 121*, 49–75.

Walker, D. W. & J. J. Dongarra (1996). Mpi: a standard message passing interface. *Supercomputer 12*, 56–68.

Zambianchi, E. & A. Griffa (1994). Effects of finite scales of turbulence on dispersion estimates. *Journal of marine research 52*(1), 129–148.

Zannetti, P. (1990). Lagrangian dispersion models. In *Air Pollution Modeling,* pp. 185–222. Springer.

Zeng, J. et al. (2010). Using nvidia gpu for modelling the lagrangian particle dispersion in the atmosphere.