Computers & Geosciences 175 (2023) 105322

Contents lists available at ScienceDirect

Computers and Geosciences

journal homepage: www.elsevier.com/locate/cageo

Research paper

Efficiently simulating Lagrangian particles in large-scale ocean flows — Data structures and their impact on geophysical applications

Christian Kehl[a,b,1], Peter D. Nooteboom[a,c,2], Mikael L.A. Kaandorp[a,c,2], Erik van Sebille[a,c,*,3]

[a] Department of Physics, Institute for Marine and Atmospheric Research, Utrecht University, The Netherlands
[b] Department for Information and Computing Sciences, Utrecht University, The Netherlands
[c] Centre for Complex Systems Studies, Utrecht University, The Netherlands

ARTICLE INFO

Keywords:
Lagrangian simulations
Particle systems
Performance enhancement
Physical oceanography

ABSTRACT

Studying oceanography by using Lagrangian simulations has been adopted for a range of scenarios, such as the determining the fate of microplastics in the ocean, simulating the origin locations of microplankton used for palaeoceanographic reconstructions, and for studying the impact of fish aggregation devices on the migration behaviour of tuna. These simulations are complex and represent a considerable runtime effort to obtain trajectory results, which is the prime motivation for enhancing the performance of Lagrangian particle simulators. This paper assesses established performance enhancing techniques from Eulerian simulators in light of computational conditions and demands of Lagrangian simulators. A performance enhancement strategy specifically targeting physics-based Lagrangian particle simulations is outlined to address the performance gaps, and techniques for closing the performance gap are presented and implemented. Realistic experiments are derived from three specific oceanographic application scenarios, and the suggested performance-enhancing techniques are benchmarked in detail, so to allow for a good attribution of speed-up measurements to individual techniques. The impacts and insights of the performance enhancement strategy are further discussed for Lagrangian simulations in other geoscience applications. The experiments show that I/O-enhancing techniques, such as dynamic loading and buffering, lead to considerable speed-up on-par with an idealised parallelisation of the process over 20 nodes. Conversely, while the cache-efficient *structure-of-arrays* collection yields a visible speed-up, other alternative data structures fail in fulfilling the theoretically-expected performance increase. This insight demonstrates the importance of good data alignment in memory and caches for Lagrangian physics simulations.

1. Introduction

Simulating the particulate transport within the oceans, such as plastics by Duncan et al. (2018), Everaert et al. (2020), van Sebille et al. (2020), plankton as in Nooteboom et al. (2019), Dämmer et al. (2020), spilled oil particulates studied by Anguiano-García et al. (2019), Calzada et al. (2021), or biota transport- and migration studied by Scutt Phillips et al. (2018), Schilling et al. (2020), Lindo-Atichati et al. (2020), Le Gouvello et al. (2020) provides quantitative insight for ecological decision-making and for achieving goals of recently enacted policies, such as the European Union's *Green Deal* and the United Nations' Sustainable Development Goals. Estimating the distribution of physical-, chemical- or biological quantities (e.g. heat, salinity) in the oceans as well as the tracing of particulates can be done using either a Eulerian- or Lagrangian computational framework.

Both approaches are viable study methods: Eulerian simulations are based on mass-, momentum- and energy conservation and the flux of water masses between finite-volume cells of a discrete Cartesian grid (Batchelor, 2000), where tracer concentration is quantified within the finite-difference scheme. In contrast, Lagrangian simulations trace attributed particles along their trajectory (van Sebille et al., 2018), where particles are advected by the background current taken in turn from an Eulerian simulation. As a consequence, Lagrangian trajectories require Eulerian simulations as input for the hydrodynamical forcing of the (passive) advection by ocean currents and waves. Furthermore, Lagrangian simulations can trivially express particle 'behaviour' (floating, sinking, etc.) in a conceptual manner.

The Eulerian- and Lagrangian approach differ in terms of their computational characteristic. Eulerian simulations evaluate numerical, vectorised equations on gridded data (Batchelor, 2000). They require

* Corresponding author at: Department of Physics, Institute for Marine and Atmospheric Research, Utrecht University, The Netherlands.
E-mail address: e.vansebille@uu.nl (E. van Sebille).
1 Lead author.
2 Contributing author.
3 Research group leader.

https://doi.org/10.1016/j.cageo.2023.105322
Received 8 July 2022; Received in revised form 12 January 2023; Accepted 25 February 2023
Available online 3 March 2023

only few externally stored information while most required data are in memory. The major workload in Eulerian simulations comes from (i) implicit address calculations and (b) the arithmetic operations. Both operations are boosted by faster hardware processors and software parallelisation (Anderson et al., 1990; Harris et al., 2020). On the other hand, Lagrangian simulations evaluate simple equations per particle using auxiliary Eulerian data (van Sebille et al., 2018). The particle is part of an unordered collection. Lagrangian simulations require substantial external data beyond the memory capacity. Therefore, their major workload is (i) accessing external data and (ii) perform interpolations on external grids (Kehl et al., 2021). Those operations are significantly less impacted by processor technology, but rather by high-speed storage hardware interfaces and the employed data layouts.

The focus of this study is the development and assessment of performance-enhancing techniques of Lagrangian simulations. Fluid particle advection in *computational fluid dynamics (CFD)* has approaches in various computational disciplines such as computer graphics & visualisation (Post and Van Walsum, 1993; Harada et al., 2007; Harada, 2007; Ribicic et al., 2013), scientific computing (Crespo et al., 2011; Horváth et al., 2016), as well as computational engineering- and physics (Kelager, 2006; Crespo et al., 2015; Lowe et al., 2019; Kanehira et al., 2019; Morikawa et al., 2021). That said, Lagrangian physical oceanography goes beyond plain particle advection: Physical models, which are combined with the advection, not only linearly interpolate attributes, but represent active particle behaviour. Moreover, physical and biochemical models introduce non-linear effects to the hydrodynamic forcing that alter the particles' motion and transport. This effect was observed in studies on nearshore behaviour (Alsina et al., 2020), beaching (Daily et al., 2021; Onink et al., 2021) and biofouling (Kooi et al., 2017; Lobelle et al., 2021). Lastly, compared to demonstrated approaches in the literature, oceanographic simulations differ from common *dam break* scenarios as particle motion is affected by chemical interactions and thermal forces in addition to kinetic forcing.

Several established compute systems are available to the oceanographic community to simulate Lagrangian particle trajectories, which differ in terms the accepted Eulerian field input formats. *TRACMASS* (Döös et al., 2013), *OpenDrift* (Dagestad et al., 2018), *ARIANE* (Blanke and Raynaud, 1997) and *parcels* (Lange and van Sebille, 2017; Delandmeter and van Sebille, 2019) are oceanographic frameworks that work with structured grids, both rectilinear and curvilinear. While parcels is a Python framework with on-the-fly *C* kernel generation, TRACMASS is a *C*-written monolithic simulation programme. *Firedrake* (Rathgeber et al., 2016) and *FESOM* (Androsov et al., 2019) advect and trace particles on unstructured grids, which complicates field interpolations for the benefit of conserving memory. Both are *C*-developed programmes, whereas the recent *OceanTracker* (Vennell et al., 2021) is a Python framework for particle tracking on unstructured grids.

This paper investigates the performance improvement of Lagrangian ocean simulations. A previous performance study on synthetic data (Kehl et al., 2021) indicated high I/O load as bottleneck for Lagrangian simulations. Hence, this study quantifies the *input/output (I/O)* load on real oceanographic simulations. Different performance-enhancing techniques for faster data access are presented based on prior developments. The access-pattern enhancement introduced by Kehl et al. (2021) is dependent on the kind of simulation being performed, thus we assess the impact of in-memory performance improvements in different oceanic simulation scenarios. Next to the in-memory transactions, read-in and write-out operations consume a bulk of simulation time. Techniques such as chunking and caching theoretically boost external I/O operations. This manuscript also investigates how those techniques impact specific oceanographic scenarios. In conclusion to the assessment of individual techniques, we derive fine-grained performance metrics that are generally applicable to all Eulerian- and Lagrangian simulations. Those metrics provide insight into the performance profile of one's simulation.

2. Methodology

The study in this paper discusses performance implications and improvements in parcels, and their implications to comparable Lagrangian simulators. Parcels is a Python framework which integrates particle trajectories and tracers either in a Python-only (i.e. *scipy* by Virtanen et al. (2020)) or a *ctypes*, *just-in-time (JIT)*-compiled *C*-mode (i.e. *jit*). The framework is built around the concept of field sets and particle sets (see Fig. 1, take from https://oceanparcels.org). A field set is an aggregated vector of array buffers that stores the hydrodynamic- and supplementary fields. A particle set is a data collection that stores the current states of particles. During the integration, the timestamped particles are written to temporary files per integration step, which are later aggregated to *NetCDF* or *zarr*. The flexible particle set size and the constant read-in and write-out of data results in performance being capped by internal- (i.e. memory) and external (i.e. disk and network) I/O operations and the related data throughput. The high memory consumption, emerging from both the Eulerian hydrodynamic field data and the large particle set size, also mandates the use of *high-performance computing (HPC)* and cluster facilities to simulate real-world scenarios for particle tracing. Any performance optimisation needs to account for this computational setting of *common use-cases*.

The general performance characteristics in Section 1 proximally derive a strategy for speeding up Lagrangian simulations. A previous study by Kehl (2021) already presented memory- and time consumption profiles on a per-function bases, which states the system's the bottleneck operations. As a result, this paper transcends the trivial improvement strategies, and instead transfers small-scale insights into real-case performance benefits.

In application-domain communities, the prevalent idea is to exploit parallelism to achieve performance improvements (e.g. message passing via MPI, shared-memory via OpenMP). Lagrangian simulations are not easily parallelised and rapidly enter diminishing speed-up rates with an increase in *processing units (PUs)*. The diminishing performance improvement is most noticeable in scenarios with a dynamic particle set size. The reason of the diminishing improvement is the cost of data I/O as primary bottleneck. In an MPI-parallelised parcels setup, particles are associated with a PU on start-up, which stays fixed over the simulation runtime. Conversely, the sharp separation allows each PU to only load a distinct field area, which is the actual cause of runtime reductions. While this strategy works well at simulation start, the runtime reduction vanishes in later simulation stages. As particles are advected in the fluid, their positions change and thus, there is an increasing overlap in the field areas each PU is loading. In a fully stirred particle configuration, this load distribution has no speed-up (see supplementary material S4). It is indeed this load distribution that caps the performance improvement from parallelisation. Improving the strategy requires excessive synchronisation and communication, which limits the performance potential of parallelisation in general.

From the analysis of per-function runtime profiles (Kehl, 2021) with synthetic in-memory fields (Kehl et al., 2021), the five most expensive functions are two particle set loops (for adding and removing particles), two transposed array-copy operations of the field set buffers, and the actual kernel execution at each computation step. As a result, the runtime load can broadly be split into *compute*- and *I/O load*. In terms of computer architecture, the delay sources (Fig. 2) that are related to I/O have a major impact on performance.

Investigating the I/O load in detail, the simulations do not benefit from latest-generation parallel-processors because the load is governed by data transfer delays on data access (Fig. 2). Therefore, the primary goal for an I/O performance increase is to maximise data throughput by avoiding or mitigating data access delays, specifically *external I/O delays*, when moving data from disk or the network into memory. Memory buffers can hide external I/O delays resulting from the high latency of I/O components. Installing those buffers physically on *solid-state drives (SSDs)*, leading to SSD buffers, may significantly boost high-I/O

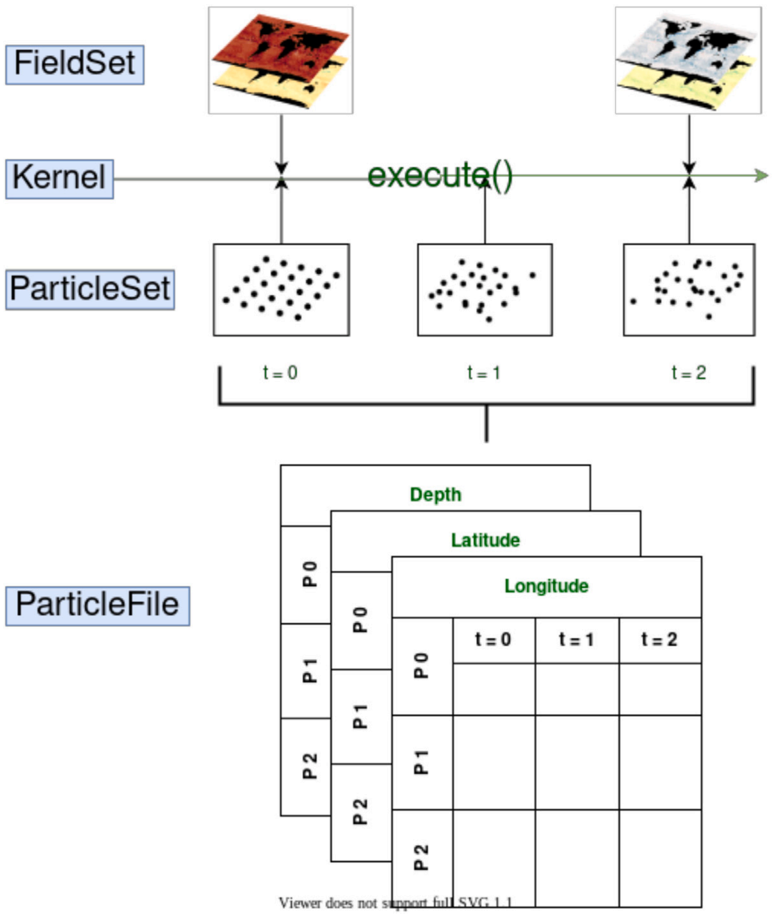


Fig. 1. Official diagram of parcels internal structure that is exposed and accessible to the user, as available at https://oceanparcels.org. It clarifies the interconnection between FieldSet, Kernel, ParicleSet and the ParticleFile.

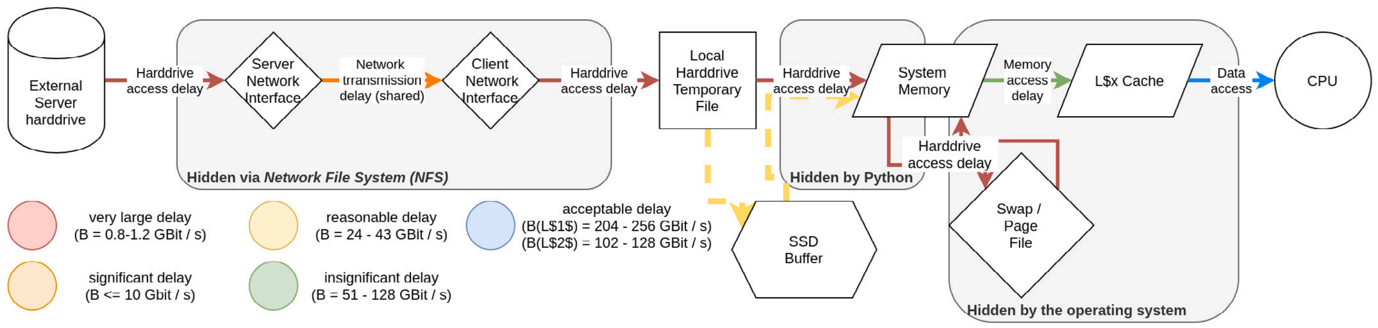

External Server harddrive
Harddrive access delay
Server Network Interface
Network trransmission delay (shared)
Client Network Interface
Harddrive access delay
Hidden via Network File System (NFS)
Local Harddrive Temporary File
Harddrive access delay
System Memory
Hidden by Python
Memory access delay
L$x Cache
Data access
CPU
Harddrive access delay
SSD Buffer
Swap / Page File
Hidden by the operating system
very large delay (B = 0.8-1.2 GBit / s)
reasonable delay (B = 24 - 43 GBit / s)
acceptable delay (B(L1) = 204 - 256 GBit / s) (B(L2) = 102 - 128 GBit / s)
significant delay (B <= 10 Gbit / s)
insignificant delay (B = 51 - 128 GBit / s)

Fig. 2. The diagram makes all active delay sources apparent in between issuing a data request DATA_REQ and having the data ready for calculation on the CPU. The impact of the delay sources varies depending on the connection bandwidth. In practice, some of those delays may be hidden from the user by computer processes, but they still exist and impact the computations. Certain delay-reducing shortcuts, such as the *solid-state drives (SSDs)*, are optional in this pipeline.

applications, even though such hardware is rare on HPC node infrastructure. Dynamic data loading via *chunks* or *slabs* reduce loading time by splitting the file-stored data in smaller, individually-loaded units. It thus prevents loading entire large files when only a small data subset is actually required. Both techniques – *memory buffers* and *dynamic data loading* – work independently and can be concurrently implemented. Next to external I/O delay mitigation, memory-related layout changes reduce the *internal I/O delay* between memory and CPU. Interleaved- or strided contiguous memory layouts (see Sedgewick and Wayne (2011), Amiri and Shahbahrami (2020) and supplementary material in Kehl et al. (2021)) for the particle sets are performance-enhancing changes for reducing internal I/O delays. Furthermore, the array-like structure (i.e. Numpy array (Harris et al., 2020)), as collection type of the particle set, is not ideal for particle insertions and removals at random indices, which are faster for list-like collections (see Sedgewick and Wayne (2011)). Hence, implementing different particle set collections potentially reduces those delays. This change has already been benchmarked by Kehl et al. (2021) for pure particle advection scenarios, while this paper investigates the I/O delay reduction in oceanographic scenarios with more extensively attributed particles.

Within the performance improvement strategy laid out here, measuring *performance* also exceeds a global runtime tracking, as still prevalent in the geosciences. Tracking individual timings for compute-, external I/O- and internal I/O operations is necessary to causally attribute runtime reductions to individual enhancement techniques. Moreover, in order to better split constant delay offset and scaling delay costs, the chosen approach tracks the global runtime, the runtime per kernel execution and the average per-particle runtime per timestep. Combining those individual metrics into (i) internal-versus-external I/O load ratio, (ii) compute-versus-internal I/O time ratio and (iii) compute-versus-external I/O time ratio enables explicit performance statements and provides performance guidance to individual flow scenarios.

Measuring the computational load with minimal systematic errors is key. Trivial particle generation for dynamic insertion and removal by generating 0 or N particles does not preserve computational load. Instead, the majority of timesteps would measure the processing of negligibly-small particle sets. Therefore, we modified the sampling so that the overall integral $\int_0^t |p| = N_{target}$. By this measure, $0 < N_{min} < N_{target} < N_{max} = N$ follows as a definition. The plots in supplementary material S5 illustrate the concept with $N_{target} = \sqrt{\frac{3}{2}}N$. Conversely, while the particle number linearly decreases by removal and linearly increases by insertion, a constant set is fixed in its particles from t_0 to t_x with $N_{target} = N$, whereas a *constantly kept* set experiences both insertion and removal while adhering to $\int_0^t |p| = N_{target}$ and $N_{target} \neq N$.

3. Datasets

The oceanographic studies in this work rely on two Eulerian datasets in terms of hydrodynamics, biochemistry and physical attributes, which are covered in the NEMO dataset by Megann et al. (2014) and Yool et al. (2013), and the *Surface and Merged Ocean Currents (SMOC)* dataset by Drillet et al. (2019). For all scenarios, we access the daily snapshots of each dataset in the period 2000–2010, with a periodic wrap in-time for longer simulation timeframes.

3.1. NEMO-MEDUSA dataset

The NEMO-MEDUSA dataset consists of hydrodynamic, physical, biological and biochemical Eulerian model data with a five-day temporal resolution and a horizontal resolution of $0.083° \times 0.083°$, as well as 50 vertical layers with an anisotropic layer thickness. The values are stored on a curvilinear ORCA C-grid, which thus requires dedicated interpolation schemes in parcels (Delandmeter and van Sebille, 2019). The grid uses a WGS84 coordinate system laterally, with depth stored in metres.

3.2. SMOC dataset

The SMOC dataset is a Eulerian hydrodymanic 2D flow model with a daily sample on a regular A-grid of $0.083° \times 0.083°$ for the first 15 m at the ocean surface (Drillet et al., 2019). It uses the WGS84 coordinate system laterally, with a bathymetric depth attribute given in metres.

The dataset is used for large-area and near-shore studies, such as the Galapagos case study. It provides hydrodynamic velocities of U and V from NEMO (Gasparin et al., 2018), as well as the stokes-drift fluid velocities at the sea surface, which are computed by the MeteoFrance Wave Action Model *WaveWatchIII* (Ardhuin et al., 2010), and tidal fluid velocities from FES2014 (Carrere et al., 2015).

4. Scenarios

In contrast to previous studies (Kehl et al., 2021), this article benchmarks the technical developments in operational oceanic simulations. The selected scenarios cover a range of computational conditions, as illustrated and referenced below.

4.1. Simulating the origin of sea level plastics around the Galapagos archipelago — Galapagos

The Galapagos Archipelago is home to one of the most iconic and unique ecosystems in the world, but it is also under pressure from human influences (Escobar-Camacho et al., 2021). In particular, large amounts of plastic wash up on some of the beaches around the Galapagos (Jones et al., 2021), carried by ocean currents from the mainland (van Sebille et al., 2019). Once in the Archipelago, the complex flow between the islands creates a pattern of capture-and-release of plastic on different shores of the islands (Ypma et al., 2022).

In order to analyse the plastic transport, particle simulations on the ocean surface with high-resolution SMOC field grids were employed. Furthermore, the effect of Stokes drift (Onink et al., 2019) is added via the WaveWatchIII data (Ardhuin et al., 2010).

A set of particles is released on a square grid around the Galapagos Islands, in the region bounded by (91.8W–89W, 1.4S–0.7N), and then advected for 14 days. A new set of particles is released every 7 days to capture the time-varying flow.

4.2. Simulating pathways and ocean surface origin locations of sedimentary microplankton — palaeo-plankton

Some near-surface living microplankton sink towards the ocean bottom as a part of their life cycle, where their remains can be preserved in sediments. As such, these sedimentary microplankton and their biogeochemical properties are representative of the climate at the ocean surface (Morey et al., 2005; Esper and Zonneveld, 2007). Therefore, fossil remains from these sedimentary microplankton can be used to make reconstructions of (near-)surface oceanographic conditions in past climates. Contrary to the accepted assumption of near-nadir sinking, microplankton is laterally transported by ocean currents and thus not representative for the overlying ocean surface conditions of its sediment location (Weyl, 1978; Nooteboom et al., 2022b).

Quantification of this *advection bias* effect (Nooteboom et al., 2019) is possible via backwards Lagrangian particle advection (Nooteboom et al., 2020). Within this *palaeo-parcels* Lagrangian method, plankton particles are periodically released every $\sim$ 1 day at the ocean floor for a few years. The particles are tracked back in time while being advected by the 3D hydrodynamic flow, accounting for the reversed sinking behaviour, until they reach the ocean surface. The environmental variables (e.g. Sea Surface Temperature (SST), sea surface salinity or primary productivity) are recorded during or at the end of transport, and compared to observations at the sedimentary release location. Once a particle reaches the ocean surfaces, it is removed from the particle set.

The palaeo-parcels method differentiates between microplankton types, which may impact performance. For instance, planktic foraminifera (van Sebille et al., 2015; Dämmer et al., 2020; Turney et al., 2020) and molecules produced by e.g. alkenones or TEX86 (Rice et al., 2022) typically sink faster compared to dinoflagellate cysts (Nooteboom et al., 2019, 2022a). Moreover, the required field variables depends on the type of microplankton and the environmental variables that are reconstructed. The simulation accesses all field variables via the NEMO-MEDUSA dataset.

4.3. Microplastics biofouling and its migration in the water column — biofouling

The biofouling simulations studies how plastic particles mix through the water column, and the resulting effect on horizontal transport in the global ocean. The most commonly used polymer types for consumer plastics, such as polyethylene, polypropylene, and polystyrene, are buoyant within seawater (Bond et al., 2018). However, an algae layer can grow on top of the plastic items over time. This can induce sinking of the plastics, as the biofilm is typically denser than seawater (Kooi et al., 2017). The biofouling simulations investigates how the realistic

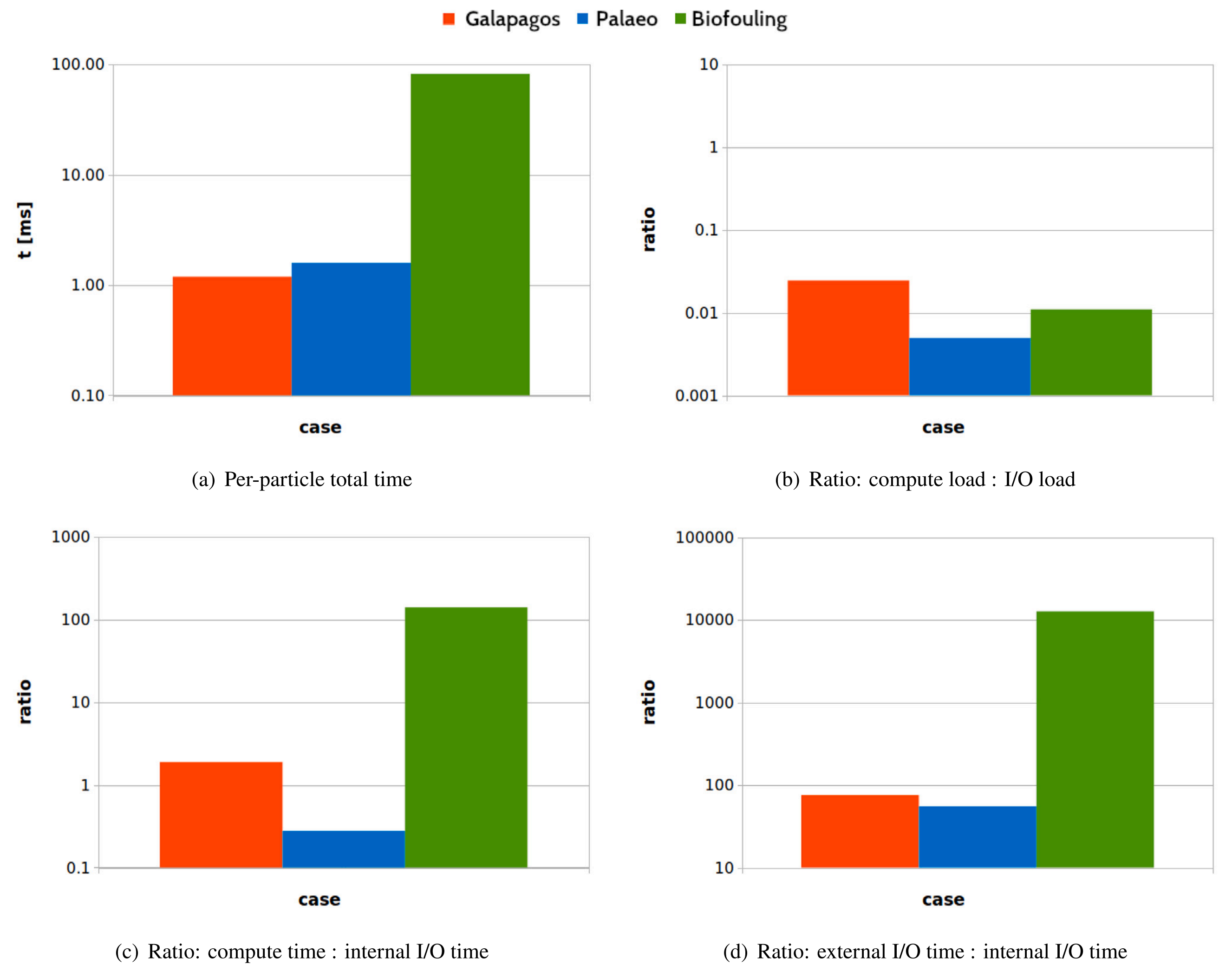


Fig. 3. Performance comparison of the different scenarios in terms of (a) per-particle total runtime, (b) compute load vs. I/O load, (c) compute time vs. internal (i.e. memory) I/O time, and (d) external- vs. internal I/O time.

algae growth on plastic particles, based on Fischer et al. (2022), affects the global dispersion of plastics. The simulations are done forward-in-time, focusing on the large spatial scales (i.e. global) and long time scales (i.e. months to years). Particles are seeded uniformly across the globe at varying depth levels in the ocean. In total, each partial simulation is run for a month with 2.3 million particles.

At first, a biofilm develops through collisions with algae in the water column, which is based on the algae concentrations, the particle's size, and particle's settling velocity. The algae concentrations is captured in two fields, one for diatom concentrations and one for nanophytoplankton concentrations. Then, the accumulated biofilm can grow. This growth is a function of the primary productivity, provided as NEMO-MEDUSA field, in the water column. In the end, the loss of algae is captured in the model via respiration. This is a function of the particle's accumulated algae and the seawater temperature, obtained from the fieldset. The combined growth and loss of algae leads to an oscillatory movement of particles in the water column, as discussed in Kooi et al. (2017), Fischer et al. (2022). The particle's settling velocity is a function of the particle's size, its density (Dietrich, 1982), and the seawater density. The seawater density is calculated using the relation from Roquet et al. (2015), based on the seawater salinity- and temperature field data. Furthermore, the particles experience vertical mixing through turbulence (Onink et al., 2022b), captured by a vertical turbulent diffusivity field. All the required fields are part of the NEMO-MEDUSA dataset and its biochemical components (see Section 3.1).

5. Results

This section presents the benchmark results, split up according to each of the three introduced performance-enhancing techniques: (a) different collection data structures to store the particle set, (b) dynamic data loading via Dask, and (c) external data buffers on SSD drives.

Initially, the three individual scenarios of Section 4 are compared so to form a discussion baseline and make subsequent measurements comparable. The simulation runtimes for the default setting of all three scenarios differ significantly due to a mismatch e.g. in particle numbers. Hence, comparing the scenarios in a meaningful way can only be done via ratios and per-particle metrics.

5.1. Assessing general performance characteristics of oceanographic scenarios

The palaeo-parcels case is hereby slightly slower than the Galapagos case, despite both simulations operating on a comparable area. The required fields alone would suggest a larger gap between both the biofouling- and the palaeo-plankton scenario. In contrast, due to the field data demands and the involved computational complexity of the kernel, the biofouling case exceeds the runtimes of the Galapagos- and

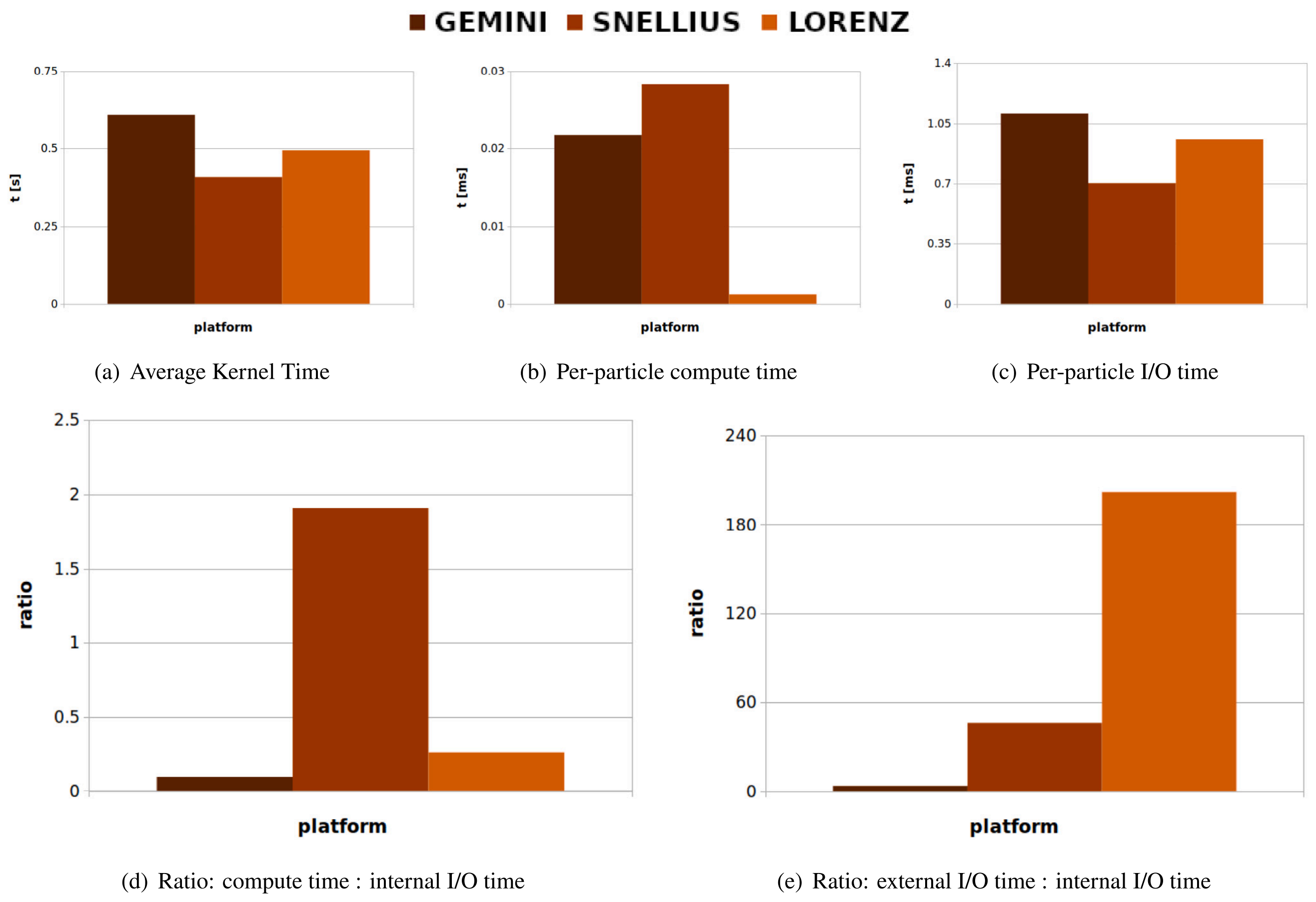


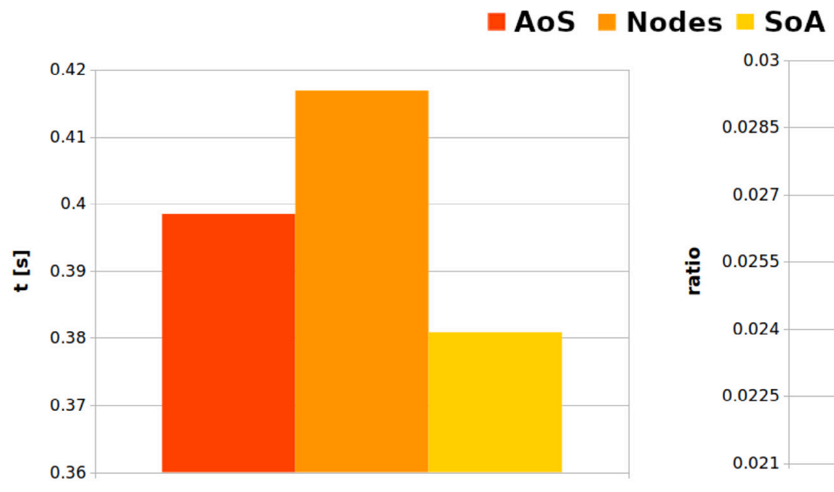
Fig. 4. Performance comparison of the three different compute platforms on the Galapagos scenario in terms of (a) average kernel runtime, (b) per-particle compute- and (c) I/O time, (d) compute time vs. internal (i.e. memory) I/O time, and (e) external- vs. internal I/O time.

palaeo-plankton scenario by two orders of magnitude. In further detail, the overall compute-to-I/O ratio (Fig. 3(b)) shows the expected behaviour: the comparably small number of fields results in a comparably high ratio for the Galapagos case, despite the simple advection kernel. The palaeo-plankton case requires more field data while having an equally simple kernel, and thus dropping the load ratio. The biofouling case has only a few more fields than the palaeo-plankton case, but a more complex computing kernel, thus its compute-to-I/O ratio is higher. Considering bottlenecks and delays (Figs. 3(c) and 3(d)), the palaeo-plankton scenario spends excessive time in data rearrangement due to particle deletion, which the other scenarios do not require. The high compute ratio compared to internal I/O for biofouling is rooted in the kernel, which is also visible because the external I/O time is four orders of magnitude higher than internal data procedures. The external-to-internal I/O ratio also shows that internal data rearrangement of the palaeo-plankton scenario is offset by its higher external I/O demands when comparing it to the Galapagos scenario.

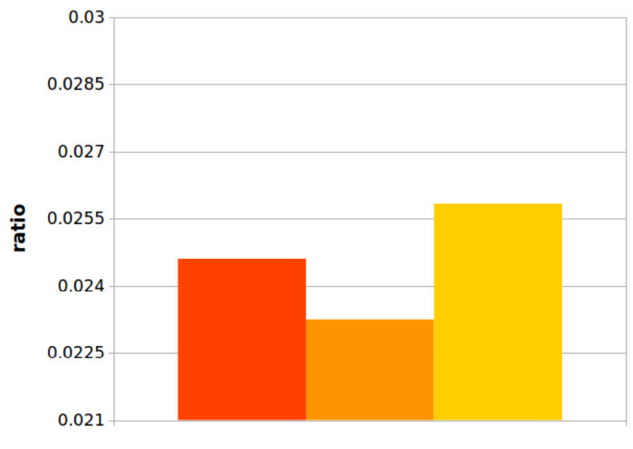
Furthermore, the gathered benchmarks can utilise different high-performance-, cluster- and distributed computing platforms. The GEMINI platform is a commodity cluster with a variable compute node hardware setup, running a non-preemptive *Sungrid Engine (SGE)* job scheduler with internal swap-space access. The SNELLIUS supercomputer is a homogeneously-equipped many-node platform with up to 256 GB per node. The supercomputer implements a preemptive SLURM job scheduler without swapping. While SNELLIUS is more strict in its usage policy, correct job preemption and the guarantee of data being in system memory makes the platform more reliable. The LORENZ cluster is the newest computing environment available. The cluster's setup is the same as for SNELLIUS, with the exception of the installed SSD buffer- or cache space on each compute node. In order to gauge the relative performance of all three platforms, Fig. 4 displays their runtimes for the Galapagos scenario using a jit-compiled kernel and an *array-of-structure (AoS)* particle set layout, similar to Kehl et al. (2021).

5.2. Impact of collection data structures and internal memory

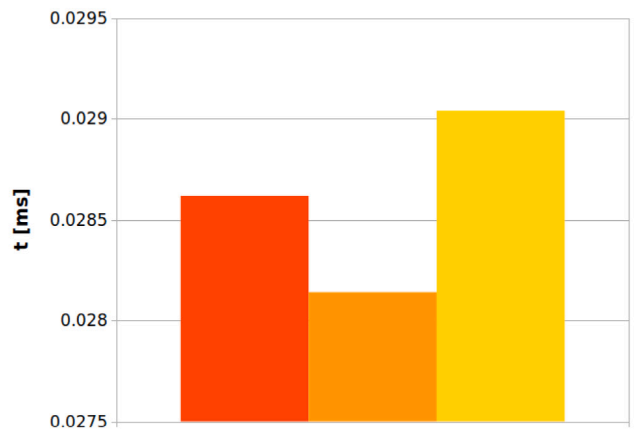
This section's experiments follow the Galapagos case, as this is the quickest scenario and the one easiest to reproduce. A first glance on the difference between the three collection structures of AoS (i.e. default option), *structure-of-arrays (SoA)* and the double-linked node-based list (i.e. *nodes*) is given with simulations of jit-based kernels and a constantly-held pool of 144 particles. The average kernel time (Fig. 5(a)) shows that the SoA collection is fastest for the computation, despite particle insertions- and removals at regular intervals, whereas the dynamic node-list is the slowest collection. A reason for this can be seen in the compute-to-I/O load (Fig. 5(b)), where SoA can allocate more time to actual computation, whereas the nodes incur a significant overhead for memory management. The interface binding to ctypes also imposes an overhead to the internal memory time. This hypothesis is supported by the per-particle compute- and I/O times (Figs. 5(c) and 5(d)): for SoA, the ctypes interface binding occurs during the kernel evaluation, thus raising the time consumption of SoA. Conversely, for node-based lists, the ctypes binding is part of the particle creation process, thus counting into the I/O time budget. That said, binding an array into ctypes is faster than binding individual elements, hence the per-particle binding process is overall slower. This is validated by compute-to-memory I/O (Fig. 5(e)) and external-to-internal I/O ratios (Fig. 5(f)), where the internal I/O delay per particle that occurs for nodes and AoS significantly limits the performances when compared



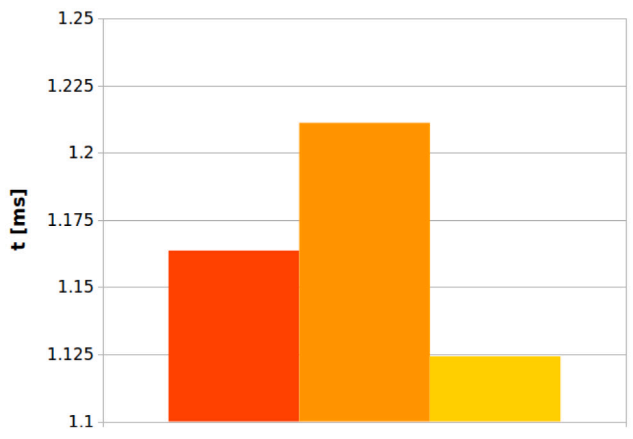
(a) Average Kernel Time



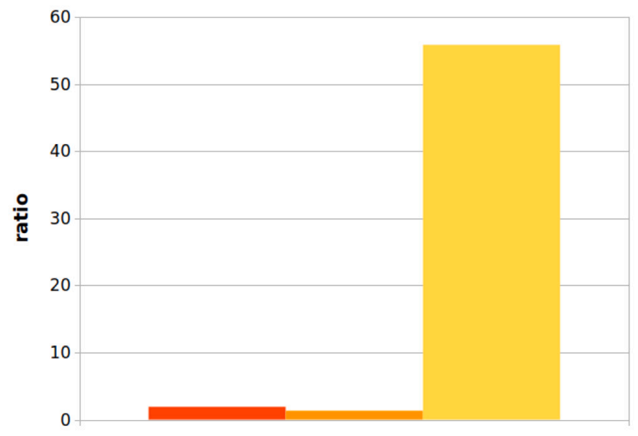
(b) Ratio: compute load : I/O load



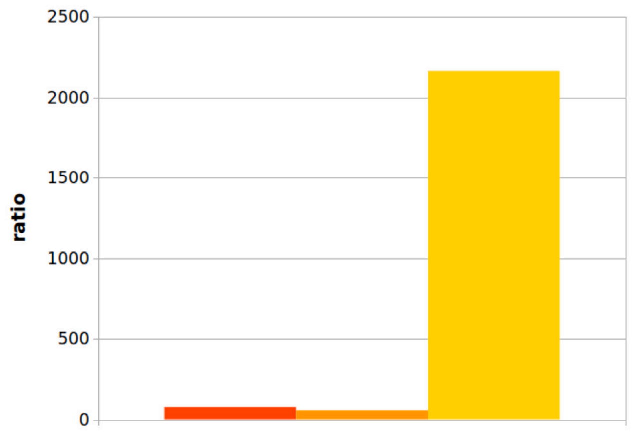
(c) Per-particle compute time



(d) Per-particle I/O time



(e) Ratio: compute time : internal I/O time



(f) Ratio: external I/O time : internal I/O time

Fig. 5. Performance comparison of the three different collection data structures using *jit* on the Galapagos scenario in terms of (a) average kernel runtime, (b) compute load vs. I/O load, (c & d) per-particle compute- and I/O time, (e) compute time vs. internal (i.e. memory) I/O time, (f) external- vs. internal I/O time.

to SoA. Another contributing hypothesis supported by previous studies the impact of SoA's cache-effective layout, as discussed in Section 2.

The costs of the ad-hoc or per-particle ctypes binding emerge when comparing the jit experiment above with an experiment just using Python and SciPy. As evident from the kernel runtime (Fig. 6(a)), the nodes is the most runtime-efficient collection, as expected from theory (Sedgewick and Wayne, 2011). In the compute-to-I/O ratio (Fig. 6(b)), we can see that without the explicit ctypes bindings, the AoS structure has the least management overhead, allowing for a maximum of computations, even though the computation itself proceeds slower. Furthermore, the overhead of the nodes is minimal when compared to SoA. The overhead for managing the list without ctypes bindings

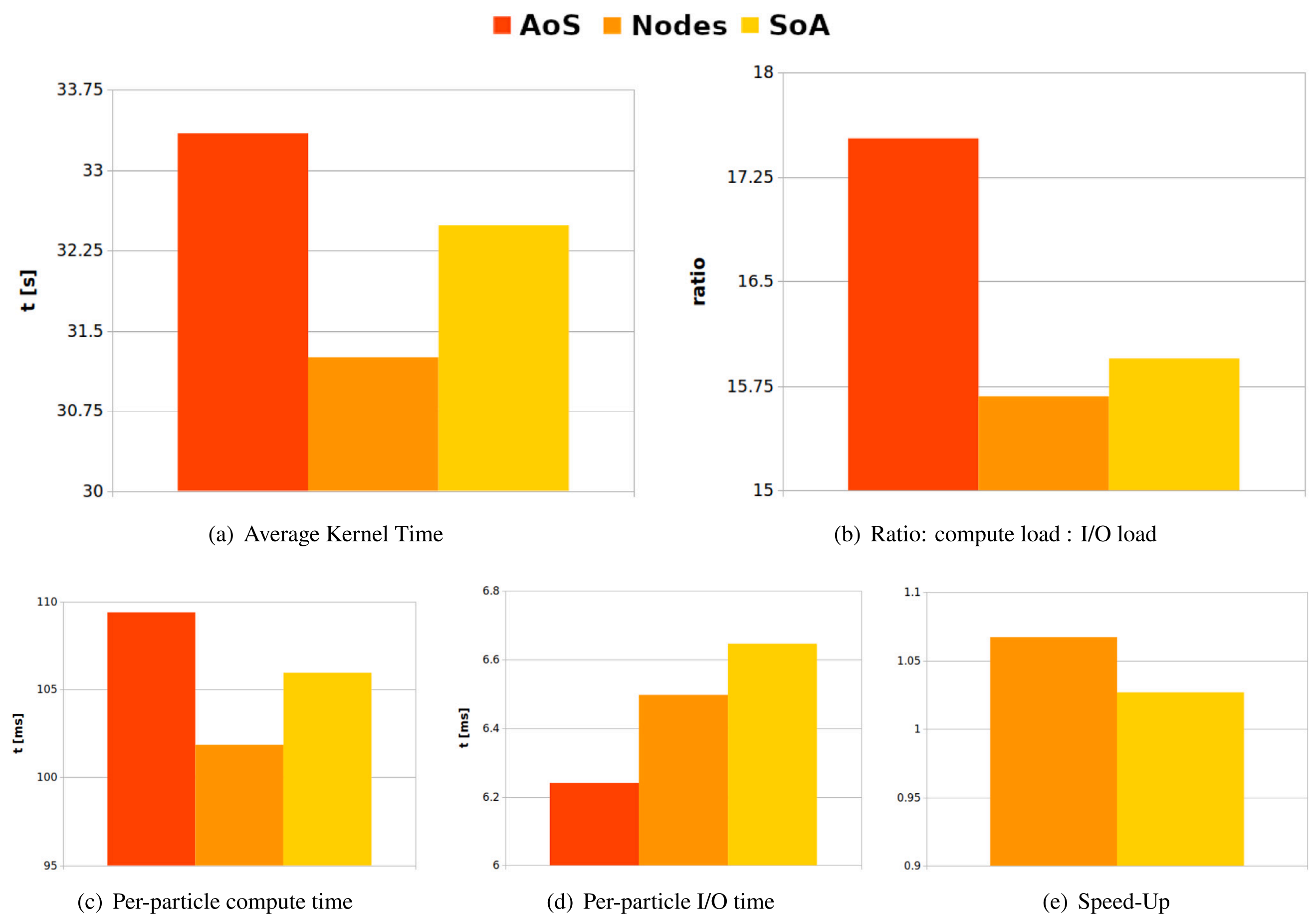


Fig. 6. Performance comparison of the three different collection data structures using **SciPy** on the Galapagos scenario in terms of (a) average kernel runtime, (b) compute-to-I/O load, (c) per-particle compute- and (d) I/O time, and (e) speed-up (relative to AoS).

for each node is smaller than the recurrent array re-allocations for SoA structures (see the per-particle I/O time in Fig. 6(c)), and it is mitigated by the more-efficient list traversal for small collections (emerging from the per-particle compute time in Fig. 6(d)). Considering the absolute speed-up of node lists and SoA over AoS (Fig. 6(e)), the actual difference for a small particle set of 144 elements is minimal, with the speed-up of the node-based list being at 1.065 and the one of SoA being at 1.025.

Considering the performance behaviour for larger datasets using the jit-interface for kernel evaluation, certain trends are clearly emerging:

1. the average kernel runtime of node-list particle sets rises exponentially with the number of particles, whereas array-like collections exhibit a linear runtime behaviour (Fig. 7(a));
2. object-organised structures (i.e. nodes and AoS) asymptotically approach a compute-to-I/O load of 0.8, whereas the array-organised SoA structure is more computationally efficient with an exponentially increasing compute-to-I/O ratio even beyond the 1.0 threshold (Fig. 7(b)), which has technical performance implications (e.g. vectorisation and parallelisation) for the future;
3. the ratio of compute-to-memory I/O stays constant for larger datasets for object-organised structures, whereas the portion of compute-operations rises linearly for SoA collections (see logarithmic plot in Fig. 7(c));
4. the impact of external file access overhead decreases linearly for all presented collection types (Fig. 7(d));
5. the double-linked node list does not deliver a consistent speed-up (Fig. 7(e)) compared to AoS, whereas SoA collections pay off with speed-ups rising linearly beyond 1.0 from a particle set size of 1500.

5.3. *Impact of dynamic data loading via dask chunking*

In order to judge the impact of chunking, a smallest running example with a pre-computed Bickley jet (Hadjighasem et al., 2017) flow field is compared to the Galapagos scenario with few (i.e. four) fields and the biofouling case with a large field number. Each of those scenarios is benchmarked in terms of overall runtime with disabled chunking (i.e. `nochk`), user-defined chunksizes (i.e. `dchk`) and auto-chunking (i.e. `achk`). As all three scenarios differ in particle set size and simulation timespan, it is advisable to compare the scenarios in terms of relative gains. The Bickley jet is the comparison baseline due to its simple 2D A-grid structure, representing the simplest case of external data access patterns in theory.

For this Bickley jet (Fig. 8), chunking in any form leads to a speed increase in the simulation. The speed-up of a user-defined chunksize is minimal compared to the automatically-derived chunksizes.

For a common application scenario, chunking introduces an overhead in computation. For a computationally simple advection case with few memory access-related interpolations, this overhead is not compensated by a computational enhancement. This can be seen in the runtime measurements for the Galapagos scenario in Fig. 9(a). Moreover, it is visible that the performance difference between a memory-optimised chunksize, as it is resulting from auto-chunking, and a suboptimal chunksize, as result of user-defined chunksizes, is significant in terms of simulation runtime. Inspecting the simulation in depth, the chunking process measurably rearranges previously stored

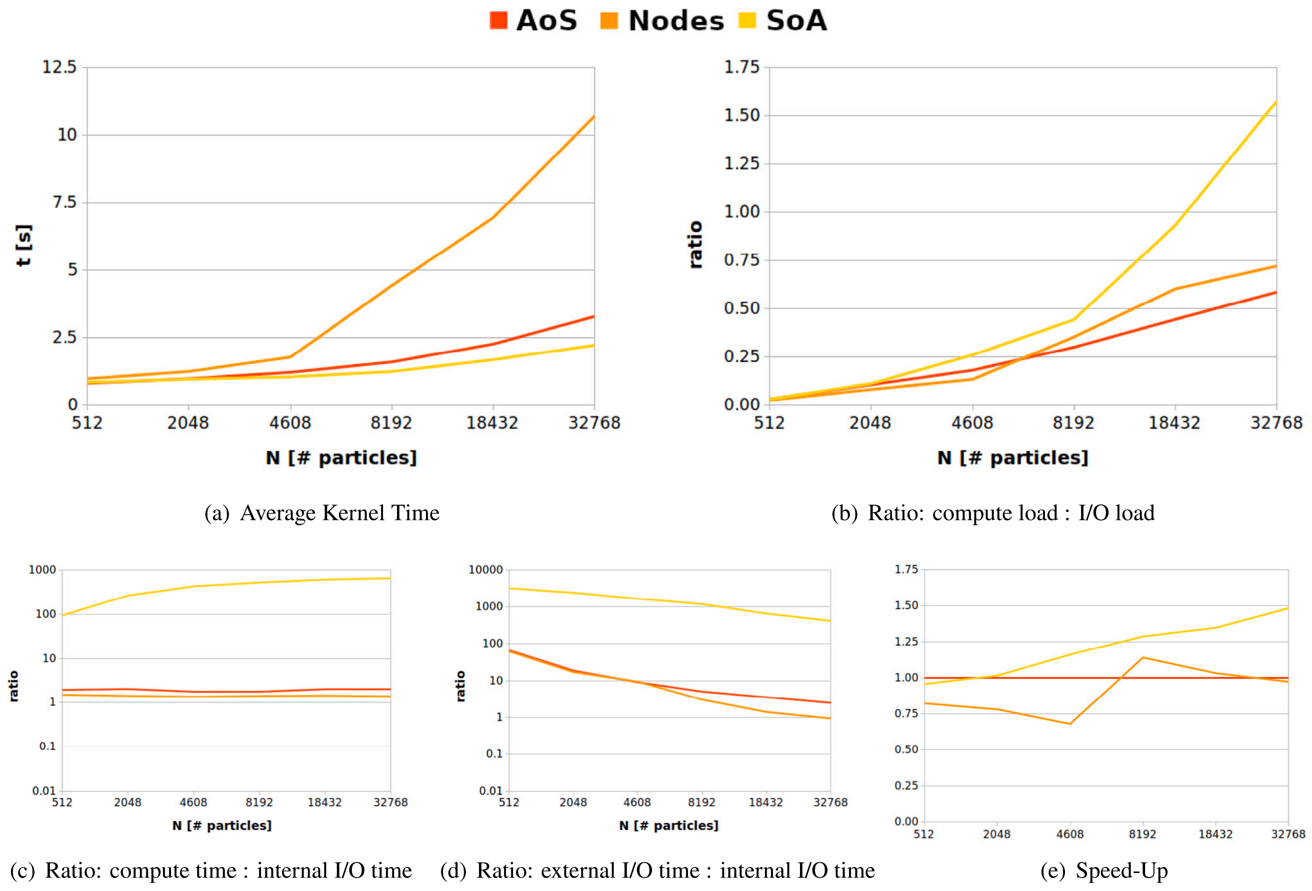


Fig. 7. Performance study of the three different collection data structures using **jit** on the Galapagos scenario for an increasing number of average simulated particles per kernel timestep in terms of (a) average kernel runtime, (b) compute-to-I/O load, (c) per-particle compute- and (d) I/O time, and (e) speed-up.

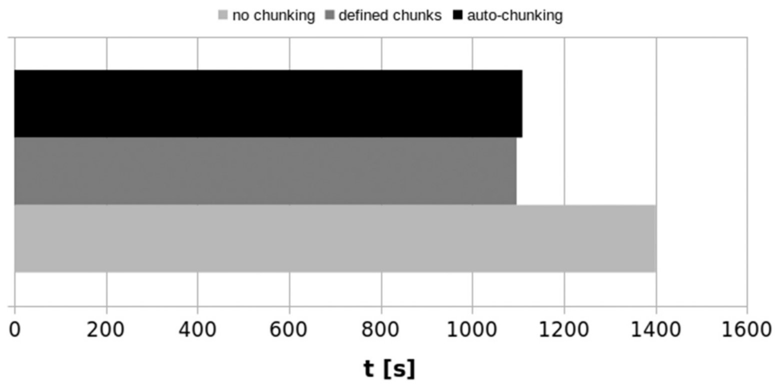


Fig. 8. Performance comparison on simulation runtime for the Bickley Jet synthetic fieldset scenario for no active chunking (light grey), user-defined chunksizes (dark grey) and auto-chunking (black).

data grids into a tree of chunked virtual cells for each field file on each file opening operation. This offset can only be compensated if the resulting chunks are small enough to reduce the loaded data, while equally being large enough so that the number of chunks do not require excessive parsing within its managing tree structure. It can be observed from the Bickley jet- and the Galapagos scenario that 2D flow computations benefit from larger, possibly non-chunk cells to reduce the parsing overhead.

The large, 3D field set scenario of the biofouling simulation behaves differently in terms of chunksizes and chunk setups (see Fig. 9(b)). A user-defined chunksize trims the runtime to only 37.14% of the same simulation without any chunking. Letting the memory-observant auto-chunking define the chunk boundaries trims this further down to only 12.54% of the user-defined chunksize runtime. Thus, overall the optimal auto-chunking procedure has a speed-up of 21.47 over the non-chunked simulation.

5.4. Impact of external data buffers

The introduction of external file buffers on high-throughput hard-drives has shown negligible benefit to the actual performance enhancement across all platforms for the Galapagos case. Actual measurements comparing a regular, low-throughput cluster (i.e. GEMINI) with a high-throughput cluster (i.e. LORENZ) can be found in supplementary material S1 to S3.

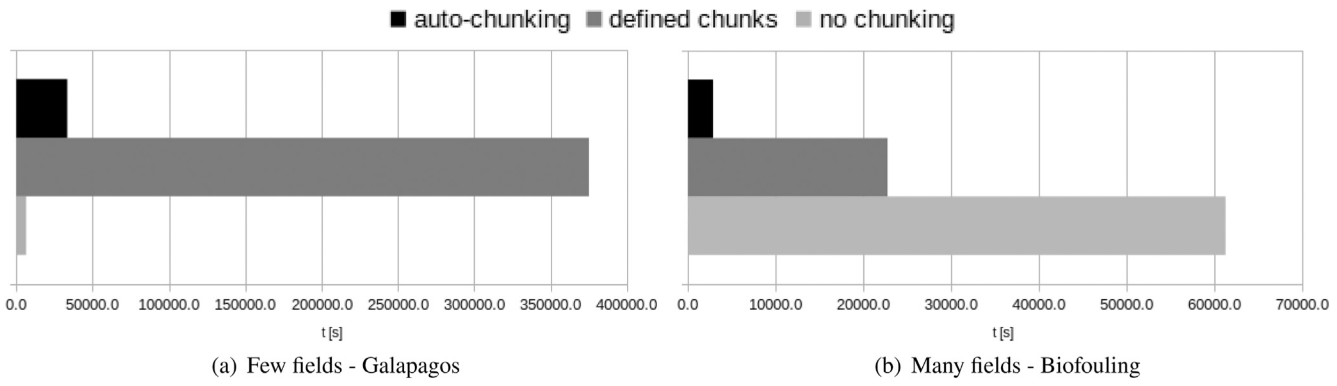


Fig. 9. Performance- and runtime comparison on simulation runtime for the Galapagos- (a) and biofouling (b) scenario with a large field set for the cases of no active chunking (light grey), user-defined chunksizes (dark grey) and auto-chunking (black).

6. Conclusions

The experiments analysed three different performance enhancing techniques. Using alternative data collection structures, such as a double-linked node-based list or an SoA layout of particles within NumPy arrays, has a significant impact on the runtime. As all evaluated advection kernels are computationally similar in their instruction composition, runtime differences in Section 5 are rooted in the effectiveness of internal- and external I/O procedures. In a jit kernel evaluation, the data need to be linked to the ctypes backend. This is very quick for array collections, whereas the node collection needs to link and refresh each element, imposing a considerable runtime overhead. Thus, in jit-based evaluations, the SoA structure outperforms the other two collection structures, which also scales well with an increasing number of particles (see Fig. 7(e)). In a SciPy setup, the need for special connections to any background framework is omitted. In this case, the measured performance follows theory, meaning that node lists outperform SoA- and AoS collections in a dynamic scenario of inserting and removing particles. A scalability study was out of scope of the displayed experiments. That said, all available information, including available previous studies (Kehl et al., 2021), suggest that this behaviour scales proportionally with the number of particles.

The other major time expense of external I/O, namely the interface to the Eulerian fields, can be reduced using dynamic loading procedures (i.e. Dask chunking). In the presented experiments, it is evident that the impact and runtime reduction achievable via chunking depends on the number and size of the required fields for each scenario. For scenarios with few and small fieldsets, only comprising the hydrodynamic velocities for advection, the performance improvement is negligible. For large-scale scenarios with multiple supplementary fields and high resolution (see supplementary material S3), the attainable performance improvement is significant and also unattainable by other means (e.g. parallelisation), with a speed-up ≥ 21 compared to non-chunked simulations.

At last, the introduction of SSD buffers for faster local data access does not show any performance improvement. There is no evidence for a specific reason why this performance enhancement strategy is not effective.

7. Discussion

The experimental results have implications for other Lagrangian simulations as well as I/O-bound process in general. Overall, the experiments validate that performance enhancement proceeds differently for compute-bound and I/O-bound processes and simulations. For I/O-bound processes, the data access delays need to be fully mitigated before compute-related enhancement techniques, such as parallelisation, yield any scalable speed-up. A deeper analysis of the performance profile also validates that runtime delays need to be profiled, and that a split between internal- and external I/O delay is beneficial to adequately address the delays. In this study, the experiments on alternative collection data structures (Section 5.2) demonstrated the response of internal I/O delays on the different collection data structures. In the related experiments, the external I/O time remains constant, and thus only the internal I/O delays affect the simulation runtime differences. Conversely, the dynamic loading and data buffering only affects the external I/O interface of fieldsets while internal I/O delays remain unaffected.

The experiments demonstrate that I/O-bound processes in general can be sped-up significantly with I/O reduction techniques, while parallelisation of the computing processes yields little to no benefit in terms of performance. Conversely, this result also demands from domain experts to comprehend the software characteristics, analyse the compute-to-I/O ratio for their individual compute scenarios, and base their performance enhancement strategy on this analysis.

This study analysed techniques for I/O optimisation for enhanced simulation performance. Alternatively, the constraining I/O delays can also be mitigated by simply raising the computational load of the simulation, with the goal to obtain more output data within the same simulation timeframe. Conversely, this approach is yet bound by the overall memory budget available to the simulation. In the presented realistic oceanographic scenarios, the available memory budget is exhausted at a significantly lower limit that what is needed to achieve a compute-to-I/O ratio of ≥ 1.0.

The chunking technique and the performance insights demonstrated here have recently been adopted by Kaandorp et al. (2022) to calculate the global, three-dimensional distribution of plastic litter in the oceans, including and merging the kernels of the Galapagos case (van Sebille et al., 2019) and the biofouling case (Lobelle et al., 2021). Additionally, the introduced SoA collection and the insights on particle set dynamics facilitate the modelling of particle–particle interaction in the oceans, which is applied by Nooteboom et al. (2023) for modelling tuna swarm motion around fish-aggregating devices. Most recently, the insights on particle dynamics facilitates speedy in-kernel particle-split and particle–particle-merge approaches in the future, which are essential for modelling plastic particle fragmentation on an oceanic scale. This research is ongoing, extending recent workflows by Onink et al. (2022a).

CRediT authorship contribution statement

Christian Kehl: Conception, Technical design, Technical implementation, Runtime experiment design, Runtime experiment execution, Runtime experiment analysis, Manuscript initiation, Manuscript writing. **Peter D. Nooteboom:** Curating physicist of the palaeo-plankton study in terms of technical design, Technical implementation, Experiment design and analysis, Manuscript writing of section 4 & 5. **Mikael L.A. Kaandorp:** Co-curating the biofouling project, Redesign and reimplementation of biofouling study, Manuscript writing of section 4 & 5. **Erik van Sebille:** Funding acquisition, Conceptualisation of Galapagos, Palaeo–plankton and biofouling scenario, Principal investigator of TOPIOS project, IMMERSE project and nanoplastics project, Manuscript writing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data are made available via the files for evaluation purposes.

Acknowledgements

All authors thank the OceanParcels group of Utrecht University's IMAU for their valuable feedback and the provided, interesting application scenarios. The research is supported by the "Tracking Of Plastic In Our Seas" (TOPIOS) project (grant agreement no. 715386), and partly by the IMMERSE project (grant agreement no. 821926), both funded by ERC's Horizon 2020 Research and Innovation program. Furthermore, the NWO Groot-funded nanoplastics project, Netherlands (ref. no. OCENW.GROOT.2019.043) supported the outlined development. Simulations were carried out on the Dutch National e-Infrastructure with the support of SURF Cooperative (project no. 16371 and 2019.034). We thank Phillipe Delandmeter, a former group member, for his considerable early contributions to the HPC developments in parcels. We further thank Delphine M.A. Lobelle, Stefanie L. Ypma and Reint P.B. Fischer for their microbiological- and physics curation of the biofouling study and the Galapagos cleanup project.

Code availability section

Package name: parcels

Contact: e.vansebille@uu.nl

Hardware requirements: laptop or workstation for small, synthetic examples; high-throughput workstation or cluster for realistic scenarios; scales to HPC facilities with MPI support via SGE, SLURM or PBS

Program language: Python

Software required: python package dependencies are lists in github's environment file; requires `mpi4py` for MPI distribution; requires `portalocker` for the hardware buffer branch.

Program size: 7.7 megabytes

The source codes are available for downloading at the link: https://github.com/oceanparcels/parcels

Installation guide, tutorials, training material and literature overview available at https://oceanparcels.org.

The package is available at conda-forge under https://anaconda.org/conda-forge/parcels.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.cageo.2023.105322.

References

Alsina, J.M., Jongedijk, C.E., van Sebille, E., 2020. Laboratory measurements of the wave-induced motion of plastic particles: Influence of wave period, plastic size and plastic density. J. Geophys. Res.: Oceans 125 (12), http://dx.doi.org/10.1029/2020JC016294, e2020JC016294, arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2020JC016294 e2020JC016294 2020JC016294.

Amiri, H., Shahbahrami, A., 2020. SIMD programming using Intel vector extensions. J. Parallel Distrib. Comput. 135, 83–100. http://dx.doi.org/10.1016/j.jpdc.2019.09.012, URL https://www.sciencedirect.com/science/article/pii/S074373151830813X.

Anderson, E., Bai, Z., Dongarra, J., Greenbaum, A., McKenney, A., Du Croz, J., Hammarling, S., Demmel, J., Bischof, C., Sorensen, D., 1990. LAPACK: A portable linear algebra library for high-performance computers. In: Proceedings of the 1990 ACM/IEEE Conference on Supercomputing. Supercomputing '90, IEEE Computer Society Press, Washington, DC, USA, pp. 2–11.

Androsov, A., Fofonova, V., Kuznetsov, I., Danilov, S., Rakowsky, N., Harig, S., Brix, H., Wiltshire, K.H., 2019. FESOM-C v. 2: Coastal dynamics on hybrid unstructured meshes. Geosci. Model Dev. 12 (3), 1009–1028.

Anguiano-García, A., Zavala-Romero, O., Zavala-Hidalgo, J., Lara-Hernández, J.A., Romero-Centeno, R., 2019. High performance open source Lagrangian oil spill model. In: Torres, M., Klapp, J., Gitler, I., Tchernykh, A. (Eds.), Supercomputing. Springer International Publishing, Cham, pp. 118–128.

Ardhuin, F., Rogers, E., Babanin, A.V., Filipot, J.-F., Magne, R., Roland, A., van der Westhuysen, A., Queffeulou, P., Lefevre, J.-M., Aouf, L., Collard, F., 2010. Semiempirical dissipation source functions for ocean waves. Part I: Definition, calibration, and validation. J. Phys. Oceanogr. 40 (9), 1917–1941. http://dx.doi.org/10.1175/2010JPO4324.1, URL https://journals.ametsoc.org/view/journals/phoc/40/9/2010jpo4324.1.xml.

Batchelor, G., 2000. An Introduction to Fluid Dynamics. In: Cambridge Mathematical Library, Cambridge University Press, URL https://books.google.de/books?id=aXQgAwAAQBAJ.

Blanke, B., Raynaud, S., 1997. Kinematics of the Pacific equatorial undercurrent: An Eulerian and Lagrangian approach from GCM results. J. Phys. Oceanogr. 27 (6), 1038–1053.

Bond, T., Ferrandiz-mas, V., Felipe-sotelo, M., van Sebille, E., 2018. The occurrence and degradation of aquatic plastic litter based on polymer physicochemical properties : A review. Cr. Rev. Environ. Sci. Technol. 48 (7–9), 685–722. http://dx.doi.org/10.1080/10643389.2018.1483155.

Calzada, A.E., Delgado, I., Ramos, C., Pérez, F., Reyes, D., Carracedo, D., Rodríguez, A., Chang, D., Cabrales, J., Lobaina, A., et al., 2021. Lagrangian model PETROMAR-3D to describe complex processes in marine oil spills. Open J. Mar. Sci. 11 (01), 17.

Carrere, L., Lyard, F., Cancet, M., Guillot, A., 2015. FES 2014, a new tidal model on the global ocean with enhanced accuracy in shallow seas and in the arctic region. In: EGU General Assembly Conference Abstracts. p. 5481.

Crespo, A.C., Dominguez, J.M., Barreiro, A., Gómez-Gesteira, M., Rogers, B.D., 2011. GPUs, a new tool of acceleration in CFD: Efficiency and reliability on smoothed particle hydrodynamics methods. PLoS One 6 (6), e20685.

Crespo, A.J., Domínguez, J.M., Rogers, B.D., Gómez-Gesteira, M., Longshaw, S., Canelas, R., Vacondio, R., Barreiro, A., García-Feal, O., 2015. DualSPHysics: Open-source parallel CFD solver based on smoothed particle hydrodynamics (SPH). Comput. Phys. Comm. 187, 204–216.

Dagestad, K.-F., Röhrs, J., Breivik, Ø., Ådlandsvik, B., 2018. OpenDrift v1.0: A generic framework for trajectory modelling. Geosci. Model Dev. 11 (4), 1405–1420. http://dx.doi.org/10.5194/gmd-11-1405-2018, URL https://gmd.copernicus.org/articles/11/1405/2018/.

Daily, J., Onink, V., Jongedijk, C.E., Laufkötter, C., Hoffman, M.J., 2021. Incorporating terrain specific beaching within a Lagrangian transport plastics model for lake Erie. Microplastics Nanoplastics 1 (1), 1–13.

Dämmer, L.K., de Nooijer, L., van Sebille, E., Haak, J.G., Reichart, G.-J., 2020. Evaluation of oxygen isotopes and trace elements in planktonic foraminifera from the Mediterranean sea as recorders of seawater oxygen isotopes and salinity. Clim. Past 16 (6), 2401–2414. http://dx.doi.org/10.5194/cp-16-2401-2020.

Delandmeter, P., van Sebille, E., 2019. The parcels v2.0 Lagrangian framework: New field interpolation schemes. Geosci. Model Dev. 12 (8), 3571–3584. http://dx.doi.org/10.5194/gmd-12-3571-2019.

Dietrich, W.E., 1982. Settling velocity of natural particles. Water Resour. Res. 18 (6), 1615–1626.

Döös, K., Kjellsson, J., Jönsson, B., 2013. TRACMASS—A Lagrangian trajectory model. In: Soomere, T., Quak, E. (Eds.), Preventive Methods for Coastal Protection: Towards the Use of Ocean Dynamics for Pollution Control. Springer International Publishing, Heidelberg, pp. 225–249. http://dx.doi.org/10.1007/978-3-319-00440-2_7.

Drillet, Y., Chune, S.L., Levier, B., Drevillon, M., 2019. SMOC: A new global surface current product containing the effect of the ocean general circulation, waves and tides. In: Geophysical Research Abstracts, Vol. 21.

Duncan, E.M., Arrowsmith, J., Bain, C., Broderick, A.C., Lee, J., Metcalfe, K., Pikesley, S.K., Snape, R.T., van Sebille, E., Godley, B.J., 2018. The true depth of the Mediterranean plastic problem: Extreme microplastic pollution on marine turtle nesting beaches in Cyprus. Mar. Pollut. Bull. 136, 334–340. http://dx.doi.org/10.1016/j.marpolbul.2018.09.019.

Escobar-Camacho, D., Rosero, P., Castrejón, M., Mena, C.F., Cuesta, F., 2021. Oceanic Islands and climate: Using a multi-criteria model of drivers of change to select key conservation areas in Galapagos. Reg. Environ. Chang. 21 (2), 47. http://dx.doi.org/10.1007/s10113-021-01768-0.

Esper, O., Zonneveld, K.A., 2007. The potential of organic-walled dinoflagellate cysts for the reconstruction of past sea-surface conditions in the Southern Ocean. Mar. Micropaleontol. 65 (3–4), 185–212. http://dx.doi.org/10.1016/j.marmicro.2007.07.002.

Everaert, G., De Rijcke, M., Lonneville, B., Janssen, C., Backhaus, T., Mees, J., van Sebille, E., Koelmans, A., Catarino, A., Vandegehuchte, M., 2020. Risks of floating microplastic in the global ocean. Environ. Pollut. 267, 115499. http://dx.doi.org/10.1016/j.envpol.2020.115499, URL http://www.sciencedirect.com/science/article/pii/S026974912036187X.

Fischer, R., Lobelle, D., Kooi, M., Koelmans, A., Onink, V., Laufkötter, C., Amaral-Zettler, L., Yool, A., van Sebille, E., 2022. Modelling submerged biofouled microplastics and their vertical trajectories. Biogeosciences 19 (8), 2211–2234.

Gasparin, F., Greiner, E., Lellouche, J.-M., Legalloudec, O., Garric, G., Drillet, Y., Bourdallé-Badie, R., Traon, P.-Y.L., Rémy, E., Drévillon, M., 2018. A large-scale view of oceanic variability from 2007 to 2015 in the global high resolution monitoring and forecasting system at Mercator océan. J. Mar. Syst. 187, 260–276. http://dx.doi.org/10.1016/j.jmarsys.2018.06.015, URL https://www.sciencedirect.com/science/article/pii/S0924796318300113.

Hadjighasem, A., Farazmand, M., Blazevski, D., Froyland, G., Haller, G., 2017. A critical comparison of Lagrangian methods for coherent structure detection. Chaos 27 (5), 053104. http://dx.doi.org/10.1063/1.4982720, arXiv:10.1063/1.4982720.

Harada, T., 2007. Real-time rigid body simulation on GPUs. GPU Gems 3, 611–632.

Harada, T., Koshizuka, S., Kawaguchi, Y., 2007. Smoothed particle hydrodynamics on GPUs. In: Computer Graphics International, Vol. 40. SBC Petropolis, pp. 63–70.

Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E., 2020. Array programming with NumPy. Nature 585, 357–362. http://dx.doi.org/10.1038/s41586-020-2649-2.

Horváth, Z., Perdigão, R.A., Waser, J., Cornel, D., Konev, A., Blöschl, G., 2016. Kepler shuffle for real-world flood simulations on GPUs. Int. J. High Perform. Comput. Appl. 30 (4), 379–395. http://dx.doi.org/10.1177/1094342016630800, arXiv:10.1177/1094342016630800.

Jones, J.S., Porter, A., Muñoz-Pérez, J.P., Alarcón-Ruales, D., Galloway, T.S., Godley, B.J., Santillo, D., Vagg, J., Lewis, C., 2021. Plastic contamination of a Galapagos Island (Ecuador) and the relative risks to native marine species. Sci. Total Environ. 789, 147704. http://dx.doi.org/10.1016/j.scitotenv.2021.147704, URL https://www.sciencedirect.com/science/article/pii/S0048969721027753.

Kaandorp, M., Lobelle, D., Kehl, C., van Sebille, E., 2022. A global 3D map of marine plastic litter: A data assimilated modelling framework. In: EGU General Assembly, Vol. 22. Copernicus Meetings, p. 5680. http://dx.doi.org/10.5194/egusphere-egu22-5680.

Kanehira, T., Mutsuda, H., Doi, Y., Taniguchi, N., Draycott, S., Ingram, D., 2019. Development and experimental validation of a multidirectional circular wave basin using smoothed particle hydrodynamics. Coast. Eng. J. 61 (1), 109–120.

Kehl, C., 2021. Performance Measurement Data for Speeding Up Lagrangian Fluid-Flow Particle Simulations in Python via Dynamic Collections. data repository 1, Utrecht University, http://dx.doi.org/10.24416/UU01-CV3OEH.

Kehl, C., van Sebille, E., Gibson, A., 2021. Speeding up python-based Lagrangian fluid-flow particle simulations via dynamic collection data structures. http://dx.doi.org/10.48550/ARXIV.2105.00057, arXiv.

Kelager, M., 2006. Lagrangian fluid dynamics using smoothed particle hydrodynamics, Vol. 2. University of Copenhagen: Department of Computer Science.

Kooi, M., Nes, E.H.v., Scheffer, M., Koelmans, A.A., 2017. Ups and downs in the ocean: Effects of biofouling on vertical transport of microplastics. Environ. Sci. Technol. 51 (14), 7963–7971. http://dx.doi.org/10.1021/acs.est.6b04702, arXiv:10.1021/acs.est.6b04702 PMID: 28613852.

Lange, M., van Sebille, E., 2017. Parcels v0.9: Prototyping a Lagrangian ocean analysis framework for the petascale age. Geosci. Model Dev. 10 (11), 4175–4186. http://dx.doi.org/10.5194/gmd-10-4175-2017, URL https://gmd.copernicus.org/articles/10/4175/2017/.

Le Gouvello, D.Z., Hart-Davis, M.G., Backeberg, B.C., Nel, R., 2020. Effects of swimming behaviour and oceanography on sea turtle hatchling dispersal at the intersection of two ocean current systems. Ecol. Model. 431, 109130. http://dx.doi.org/10.1016/j.ecolmodel.2020.109130, URL http://www.sciencedirect.com/science/article/pii/S0304380020302027.

Lindo-Atichati, D., Jia, Y., Wren, J.L.K., Antoniades, A., Kobayashi, D.R., 2020. Eddies in the Hawaiian Archipelago Region: Formation, characterization, and potential implications on Larval retention of reef fish. J. Geophys. Res.: Oceans 125 (5), http://dx.doi.org/10.1029/2019JC015348, e2019JC015348, arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2019JC015348 e2019JC015348 10.1029/2019JC015348.

Lobelle, D., Kooi, M., Koelmans, A., Laufkötter, C., Jongedijk, C., Kehl, C., van Sebille, E., 2021. Global modeled sinking characteristics of biofouled microplastic. J. Geophys. Res.: Oceans (in review, submitted Dec. 2020).

Lowe, R.J., Buckley, M.L., Altomare, C., Rijnsdorp, D.P., Yao, Y., Suzuki, T., Bricker, J., 2019. Numerical simulations of surf zone wave dynamics using smoothed particle hydrodynamics. Ocean Model. 144, 101481.

Megann, A., Storkey, D., Aksenov, Y., Alderson, S., Calvert, D., Graham, T., Hyder, P., Siddorn, J., Sinha, B., 2014. GO5. 0: the joint NERC–met office NEMO global ocean model for use in coupled and forced applications. Geosci. Model Dev. 7 (3), 1069–1092.

Morey, A., Mix, A.C., Pisias, N.G., 2005. Planktonic foraminiferal assemblages preserved in surface sediments correspond to multiple environment variables. Quat. Sci. Rev. 24, 925–950. http://dx.doi.org/10.1016/j.quascirev.2003.09.011.

Morikawa, D., Senadheera, H., Asai, M., 2021. Explicit incompressible smoothed particle hydrodynamics in a multi-GPU environment for large-scale simulations. Comput. Part. Mech. 8 (3), 493–510.

Nooteboom, P.D., Baatsen, M., Bijl, P.K., Kliphuis, M.A., van Sebille, E., Sluijs, A., Dijkstra, H.A., von der Heydt, A.S., 2022a. Improved model-data agreement with strongly eddying ocean simulations in the middle-late eocene. Paleoceanogr. Paleoclimatol. 37 (8), http://dx.doi.org/10.1029/2021PA004405, e2021PA004405, arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2021PA004405 e2021PA004405 2021PA004405.

Nooteboom, P., Bijl, P., Kehl, C., van Sebille, E., Ziegler, M., von der Heydt, A., Dijkstra, H., 2022b. Sedimentary microplankton distributions are shaped by oceanographically connected areas. Earth Syst. Dyn. 13 (357), 357–371. http://dx.doi.org/10.5194/esd-13-357-2022.

Nooteboom, P.D., Bijl, P.K., van Sebille, E., von der Heydt, A.S., Dijkstra, H.A., 2019. Transport bias by ocean currents in sedimentary microplankton assemblages: Implications for paleoceanographic reconstructions. Paleoceanogr. Paleoclimatol. 34 (7), 1178–1194. http://dx.doi.org/10.1029/2019PA003606, arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2019PA003606.

Nooteboom, P., P., D., van Sebille, E., Bijl, P., Dijkstra, H., von der Heydt, A., 2020. Resolution dependency of sinking Lagrangian particles in ocean general circulation models. PloS ONE 15 (9), 1–16. http://dx.doi.org/10.1371/journal.pone.0238650.

Nooteboom, P.D., Phillips, J.S., Kehl, C., Nicol, S., van Sebille, E., 2023. Modelling of tuna around fish aggregating devices: The importance of ocean flow and prey. Ecol. Model. 475, 110188. http://dx.doi.org/10.1016/j.ecolmodel.2022.110188.

Onink, V., Jongedijk, C.E., Hoffman, M.J., van Sebille, E., Laufkötter, C., 2021. Global simulations of marine plastic transport show plastic trapping in coastal zones. Environ. Res. Lett. 16 (6), 064053.

Onink, V., Kaandorp, M.L., van Sebille, E., Laufkötter, C., 2022a. Influence of particle size and fragmentation on large-scale microplastic transport in the Mediterranean sea. Environ. Sci. Technol. 56 (22), 15528–15540. http://dx.doi.org/10.1021/acs.est.2c03363.

Onink, V., van Sebille, E., Laufkötter, C., 2022b. Empirical Lagrangian parametrization for wind-driven mixing of buoyant particles at the ocean surface. Geosci. Model Dev. 15 (5), 1995–2012.

Onink, V., Wichmann, D., Delandmeter, P., van Sebille, E., 2019. The role of Ekman currents, geostrophy, and Stokes drift in the accumulation of floating microplastic. J. Geophys. Res.: Oceans 124 (3), 1474–1490. http://dx.doi.org/10.1029/2018JC014547, arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2018JC014547.

Post, F.H., Van Walsum, T., 1993. Fluid flow visualization. In: Focus on Scientific Visualization. Springer, pp. 1–40.

Rathgeber, F., Ham, D.A., Mitchell, L., Lange, M., Luporini, F., Mcrae, A.T.T., Bercea, G.-T., Markall, G.R., Kelly, P.H.J., 2016. Firedrake: Automating the finite element method by composing abstractions. ACM Trans. Math. Software 43 (3), 24:1–24:27. http://dx.doi.org/10.1145/2998441, arXiv:1501.01809.

Ribicic, H., Waser, J., Fuchs, R., Bloschl, G., Gröller, E., 2013. Visual analysis and steering of flooding simulations. IEEE Trans. Vis. Comput. Graphics 19 (06), 1062–1075. http://dx.doi.org/10.1109/TVCG.2012.175.

Rice, A., Nooteboom, P., van Sebille, E., Peterse, F., Ziegler, M., Sluijs, A., 2022. Limited lateral transport bias during export of sea surface temperature proxy carriers in the Mediterranean sea. Geophys. Res. Lett. 49 (4), 1–10. http://dx.doi.org/10.1029/2021GL096859.

Roquet, F., Madec, G., McDougall, T.J., Barker, P.M., 2015. Accurate polynomial expressions for the density and specific volume of seawater using the TEOS-10 standard. Ocean Model. 90, 29–43. http://dx.doi.org/10.1016/j.ocemod.2015.04.002, URL http://dx.doi.org/10.1016/j.ocemod.2015.04.002.

Schilling, H.T., Everett, J.D., Smith, J.A., Stewart, J., Hughes, J.M., Roughan, M., Kerry, C., Suthers, I.M., 2020. Multiple spawning events promote increased larval dispersal of a predatory fish in a western boundary current. Fisheries Oceanography 29 (4), 309–323. http://dx.doi.org/10.1111/fog.12473, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/fog.12473.

Scutt Phillips, J., Sen Gupta, A., Senina, I., van Sebille, E., Lange, M., Lehodey, P., Hampton, J., Nicol, S., 2018. An individual-based model of skipjack tuna (Katsuwonus pelamis) movement in the tropical Pacific ocean. Prog. Oceanogr. 164, 63–74. http://dx.doi.org/10.1016/j.pocean.2018.04.007, URL http://www.sciencedirect.com/science/article/pii/S0079661117302896.

Sedgewick, R., Wayne, K., 2011. Algorithms. Addison-Wesley, URL https://books.google.nl/books?id=MTpsAQAAQBAJ.

Turney, C., Jones, R., McKay, N., van Sebille, E., Thomas, Z., Hillenbrand, C., Fogwill, C., 2020. A global mean sea surface temperature dataset for the Last Interglacial (129–116 ka) and contribution of thermal expansion to sea level change. Earth Syst. Sci. Data 12, 3341–3356. http://dx.doi.org/10.5194/essd-12-3341-2020.

van Sebille, E., Aliani, S., Law, K.L., Maximenko, N., Alsina, J.M., Bagaev, A., Bergmann, M., Chapron, B., Chubarenko, I., Cózar, A., Delandmeter, P., Egger, M., Fox-Kemper, B., Garaba, S.P., Goddijn-Murphy, L., Hardesty, B.D., Hoffman, M.J., Isobe, A., Jongedijk, C.E., Kaandorp, M.L.A., Khatmullina, L., Koelmans, A.A., Kukulka, T., Laufkötter, C., Lebreton, L., Lobelle, D., Maes, C., Martinez-Vicente, V., Maqueda, M.A.M., Poulain-Zarcos, M., Rodríguez, E., Ryan, P.G., Shanks, A.L., Shim, W.J., Suaria, G., Thiel, M., van den Bremer, T.S., Wichmann, D., 2020. The physical oceanography of the transport of floating marine debris. Environ. Res. Lett. 15 (2), 023003. http://dx.doi.org/10.1088/1748-9326/ab6d7d.

van Sebille, E., Delandmeter, P., Schofield, J., Hardesty, B.D., Jones, J., Donnelly, A., 2019. Basin-scale sources and pathways of microplastic that ends up in the Galápagos Archipelago. Ocean Sci. 15 (5), 1341–1349. http://dx.doi.org/10.5194/os-15-1341-2019.

van Sebille, E., Griffies, S.M., Abernathey, R., Adams, T.P., Berloff, P., Biastoch, A., Blanke, B., Chassignet, E.P., Cheng, Y., Cotter, C.J., Deleersnijder, E., Döös, K., Drake, H.F., Drijfhout, S., Gary, S.F., Heemink, A.W., Kjellsson, J., Koszalka, I.M., Lange, M., Lique, C., MacGilchrist, G.A., Marsh, R., Mayorga Adame, C.G., McAdam, R., Nencioli, F., Paris, C.B., Piggott, M.D., Polton, J.A., Rühs, S., Shah, S.H., Thomas, M.D., Wang, J., Wolfram, P.J., Zanna, L., Zika, J.D., 2018. Lagrangian ocean analysis: Fundamentals and practices. Ocean Model. 121, 49–75. http://dx.doi.org/10.1016/j.ocemod.2017.11.008.

van Sebille, E., Scussolini, P., Durgadoo, J., Peeters, F., Biastoch, A., Weijer, W., Turney, C., Paris, C., Zahn, R., 2015. Ocean currents generate large footprints in marine palaeoclimate proxies. Nature Commun. 6 (6521), 8. http://dx.doi.org/10.1038/ncomms7521.

Vennell, R., Scheel, M., Weppe, S., Knight, B., Smeaton, M., 2021. Fast lagrangian particle tracking in unstructured ocean model grids. Ocean Dyn. 71 (4), 423–437.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al., 2020. SciPy 1.0: Fundamental algorithms for scientific computing in Python. Nature Methods 17 (3), 261–272.

Weyl, P.K., 1978. Micropaleontology and ocean surface climate. Science 202, 475–481.

Yool, A., Popova, E.E., Anderson, T.R., 2013. MEDUSA-2.0: An intermediate complexity biogeochemical model of the marine carbon cycle for climate change and ocean acidification studies. Geosci. Model Dev. 6 (5), 1767–1811.

Ypma, S.L., Bohte, Q., Forryan, A., Naveira Garabato, A.C., Donnelly, A., van Sebille, E., 2022. Detecting most effective cleanup locations using network theory to reduce marine plastic debris: A case study in the Galapagos Marine Reserve. EGUsphere 1–22. http://dx.doi.org/10.5194/egusphere-2022-426, URL https://egusphere.copernicus.org/preprints/2022/egusphere-2022-426/ Publisher: Copernicus GmbH.