Unlocking the Power of Parallel Computing: GPU technologies for Ocean Forecasting

Andrew R. Porter[1] and Patrick Heimbach[2]

[1]Science and Technology Facilities Council, Daresbury Laboratory, Hartree Centre, Daresbury, UK
[2]Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, USA

Correspondence to: Andrew Porter (andrew.porter@stfc.ac.uk)

~~Actions on this paper:~~

1. ~~Please, review the abstract and correct it.~~
2. ~~Please, go to the section "Competing interests" and if the default statement is wrong, please change.~~
3. ~~Additional sections "Data and/or code availability", "Authors contribution" and "Acknowledgements" can be completed also during the review phase, but if you prefer to complete now, please do.~~

~~Find here some information (suggested title and list of authors, including affiliations), that will help you to speed up the submission process:~~

~~Title~~	~~Unlocking the Power of Parallel Computing: GPU technologies for Ocean Forecasting~~

~~ID~~	~~First name (incl. middle names)~~	~~Last name~~	~~email~~	~~Affiliation (if you are registered, this is disabled, but I report in any case. It is mandatory ONLY for the Corresponding Author)~~
~~1~~	~~Andrew Robert~~	~~Porter~~	~~andrew.porter@stfc.ac.uk~~	~~Science and Technology Facilities Council, Daresbury Laboratory, Hartree Centre, Daresbury, UK~~

Abstract. Operational ocean forecasting systems are complex engines that must execute ocean models with high performance to provide timely products and datasets. Significant computational resources are then needed to run high-fidelity models and, historically, technological evolution of microprocessors has constrained data parallel scientific computation. Today, GPUs offer a rapidly growing ~~an additional~~ and valuable source of computing power rivaling ~~to~~ the traditional CPU-based machines: the exploitation of thousands of threads can significantly accelerate the execution of many models, ranging from traditional HPC workloads of finite-difference/volume/element modelling through to the training of deep neural networks used in machine learning and artificial intelligence. Despite the advantages, GPU usage in ocean forecasting is still limited due to the legacy of CPU-based model implementations and the intrinsic complexity of porting core models to GPU architectures. This review explores the potential use of GPU in ocean forecasting and how the

computational characteristics of ocean models can influence the suitability of GPU architectures for the execution of the overall value chain: it discusses the current approaches to code (and performance) portability, from CPU to GPU, ~~differentiating among~~including tools that perform code-transformation, easing the adaptation of Fortran code for GPU execution (like PSyclone)~~ or~~, direct use of OpenACC directives (like ICON-O),~~ to~~ adoption of specific frameworks that facilitate the management of parallel execution across different architectures, and also~~to~~ the ~~exploiting~~use of new programming languages and paradigms.

1 Introduction

Operational Ocean Forecasting Systems (OOFS) are computationally demanding, and large compute resources are required in order to run models of useful fidelity. However, this is a time of great upheaval in the development of computer architectures. The ever-shrinking size of transistors means that current leakage (and the resulting heat generated) now presents a significant challenge to chip designers. This breakdown of 'Dennard Scaling' (transistor power consumption is proportional to area as in Dennard et al., 1974) began in about 2006 and means that it is no longer straightforward to continually increase the clock frequency of processors. Historically this has been the main source of performance improvement from one generation of processor to the next (Figure 1). Although the number of transistors per device continues to rise, they are increasingly being used to implement larger numbers of execution cores. It is then the job of the application to make use of these additional cores to achieve a performance improvement. Graphical Processing Units (GPUs) are a natural consequence of this evolution. Originally developed to accelerate rendering of computer-generated images (a naturally data-parallel task thanks to the division of an image into pixels), scientists were quick to seize on their potential to accelerate data-parallel scientific computation. Therefore, manufacturers today produce HPC-specific "GPUs" that are purely intended for computation. The suitability of this hardware for the training of deep neural networks used in machine learning and artificial intelligence has stimulated massive development and competition amongst GPU vendors. Because of the exploding interest of AI applications in virtually all sectors of industry, the commercial HPC market is undergoing a seismic shift toward GPU-based hardware, with serious implications for available HPC architectures in the future, to which OOPC will have to adapt.

Unlike CPUs which tend to have relatively few but powerful (general purpose) processor cores, GPUs support hundreds of simpler cores running thousands of threads which can get data from memory very efficiently. The simplicity of these cores makes them more energy efficient and therefore GPUs tend to offer significantly greater performance per Watt. With energy consumption of large computing facilities now the key design criterion, GPUs are an important part of the technology being used in the push towards Exascale performance and beyond (e.g. Draeger and Siegel, 2023). As an illustration, in the November 2024~~2~~ incarnation of the Top500 list (Strohmaier et al., 202~~0~~4), nine~~eight~~ of the machines in the top ten are equipped with GPU~~some form of~~ accelerators ~~and the majority of those are GPUs~~ from ~~either~~ NVIDIA, Intel or AMD. Although CPUs are present in these machines, their primary role is to host the GPUs which provide the bulk of the compute performance. GPUs are therefore a major feature of the current HPC landscape, and their importance and pervasiveness is only set to increase.

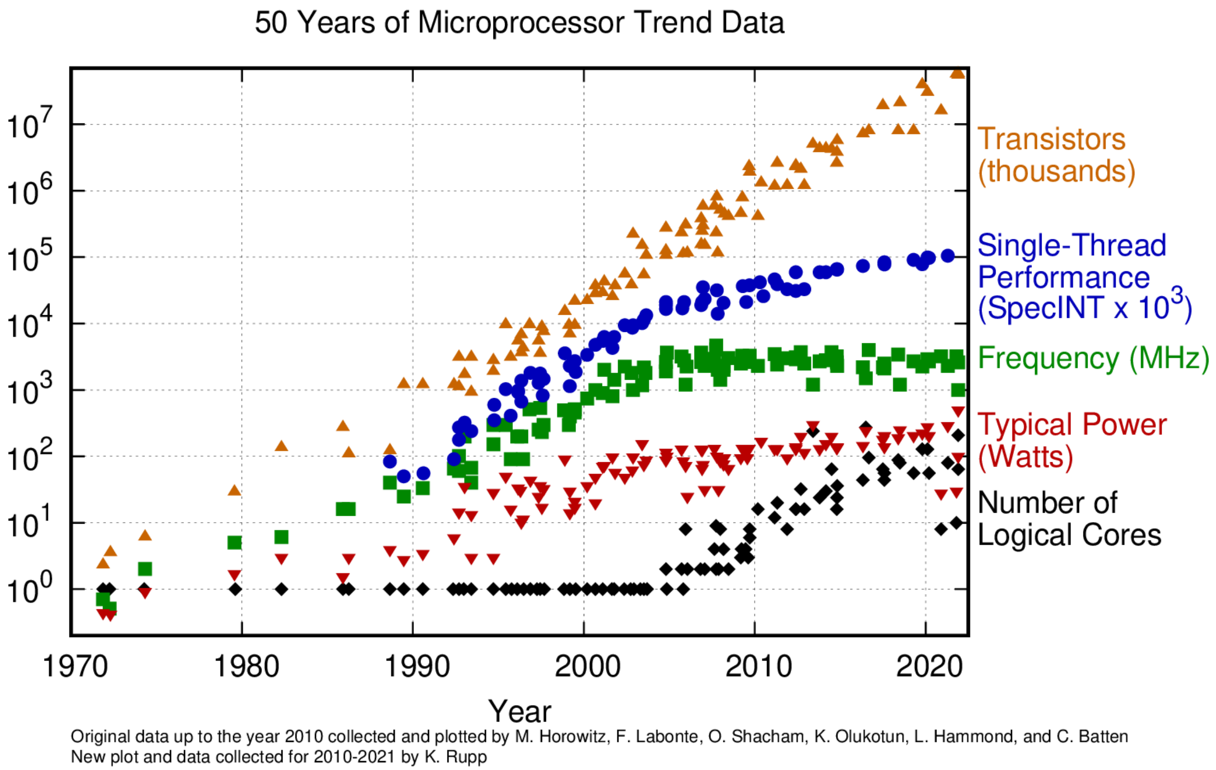

Figure 1: 50 years of microprocessor (CPU) evolution showing the breakdown of Dennard scaling (Rupp, 2022)

2 Computational Characteristics of Ocean Models

To understand why GPUs are well suited to running OOFS, it is important to consider their computational characteristics. The equations describing ocean evolution form a system of partial differential ~~equation~~equations that are ~~are~~ solved numerically by discretizing the model domain and then using a Finite Difference, Finite Volume or Finite Element scheme. In these forms, the bulk of the computational work takes the form of stencil computations where the update of a field at a given grid location requires that many other field values be read from neighbouring locations. This means that the limiting factor in the rate at which these computations can be done is how quickly all these values can be fetched from memory (so called 'memory bandwidth'). (Finite element schemes do have the advantage of shifting the balance in favour of doing more arithmetic operations but memory bandwidth still tends to dominate.) These computations are of course repeated across the entire model grid meaning that it is a Same Instruction Multiple Data (SIMD) problem. OOFS are therefore a very good fit for GPU architectures which naturally support massively data-parallel problems and typically provide much higher memory bandwidth than CPUs.

For execution on distributed-memory computers, OOFS typically use a geographical domain decomposition where each processor is assigned a part of the model domain. In order to handle stencil updates at the boundaries of a processor's sub-domain, it must exchange information with those processors operating on neighbouring sub-domains. Obviously, there is a cost associated with performing these exchanges which high-performance processor interconnects can only do so much to mitigate. As more processors are thrown at a problem in order to reduce the time to solution, the size of their sub-domains decreases and so too does the amount of computation that each must perform. Consequently, the relative cost of inter-processor communication becomes more significant and, after a certain point (the "strong-scaling limit"), will begin to dominate. At this point, using further processors will bring only limited performance improvements, if any.

Inter-processor communication on a GPU-based machine can be more costly as messages may have to go via the CPUs hosting the GPUs unless a machine has both hardware and software support for direct GPU-GPU communication. Communication avoiding/minimising strategies are therefore more important on these architectures. These can include algorithmic design (e.g. Silvestri et al, 2024) to allow for the overlap of communication and computation or simply the use of wider halo regions to reduce the frequency of halo exchanges.

3 The use of GPUs in Ocean Forecasting

Although GPUs are now a well-established HPC technology with potentially significant performance advantages for OOFS, they are not yet widely adopted in the ocean-forecasting community. For example, in Europe, NEMO (Madec et al., 202~~3~~4) is the most important ocean-modeling framework; it is used operationally by Mercator Ocean International, the European Centre for Medium-Range Weather Forecasts ~~ing~~ (ECMWF), the UK Met Office and the Euro-Mediterranean Centre on Climate Change, and other Institutes worldwide. NEMO is implemented in Fortran and parallelised with MPI and as such is limited to running on CPUs only. The German weather service (DWD) uses ICON-Ocean (Korn, 2017) which is also a Fortran model. Experiments are in progress with the use of OpenACC directives to extend this code to make use of GPUs but this functionality is not used operationally.

In the US, NOAA's Real-Time Ocean Forecast System (https://polar.ncep.noaa.gov/global/) is based on HYCOM (HYbrid Coordinates Ocean Model, Chassignet et al., 2009). HYCOM too is a Fortran code parallelised using a combination of OpenMP and MPI. Although not used operationally, the Energy Exascale Earth System Model (E3SM) is also significant. It utilizes the MPAS (Model for Prediction Across Scales) Ocean, Sea-Ice and Land-Ice models (Ringler et al., 2013) which again is implemented in Fortran with MPI. Although a port of this was attempted through the addition of OpenACC directives, it has been abandoned due to poor GPU performance (Petersen, 2024). Instead, a new, unstructured-mesh ocean model named Omega is being developed in C++ from the ground up. ~~(although some experimental ports have been performed using OpenACC directives).~~ Other widely used ocean general circulation models include the ~~The~~ MIT General

Circulation Model (MITgcm, Marshall et al., 1997) and the Modular Ocean Model, version 6 (MOM6; Adcroft et al., 2019)~~,~~ ~~and the Regional Ocean Modeling System (ROMS; Moore et al., 2004) is also widely used and~~, both of which again are ~~is~~ Fortran codes with support for distributed- and shared-memory parallelism on CPU.

The Japanese Meteorological Agency runs operational forecasts using the Meteorological Research Institute Community Ocean Model (MRI.COM) (Tsujino et al., 2010). As with the previous models, this too is implemented in Fortran with MPI and thus only runs on CPU.

For regional (as opposed to global) forecasts, the Rutgers Regional Ocean Modeling System (ROMS) (Shchepetkin and McWilliams, 2023) is used by centers worldwide including the Japan Fisheries Research and Education Agency, the Australian Bureau of Meteorology and the Irish Marine Institute. ROMS too is a Fortran code parallelised using either MPI or OpenMP (but not both combined) and thus is restricted to CPU execution. Although various projects have ported the code to different architectures (including the Sunway architecture for China's Tianhe machine, Liu et al., 2019), these are all standalone pieces of work that have not made it back into the main code base.

4 Discussion

From the preceding section, it is clear that OOFS are currently largely implemented in Fortran with no or limited support for execution on GPU devices. The problem here is that OOFS comprise of large and complex codes which typically have a lifetime of decades and are constantly being updated with new science by multiple developers. Maintainability, allowing for the fact that the majority of developers will be specialists in their scientific domain rather than in HPC, is therefore of vital importance. Given that such codes are often shared between organizations, they must also run with good performance on different types of architecture (i.e. be 'performance portable').

Previously, one generation of supercomputers looked much like the last and therefore the evolution of these computer models was not a significant problem. However, the proliferation of computer hardware (and, crucially, the programming models needed to target them) that has resulted from the breakdown of Dennard scaling has changed this (Balaji, 2021). With the average supercomputer having a lifetime of just some five years, OOFS are now facing the problem of adapting to future supercomputer architectures and this is difficult because the aims of performance, performance portability and code maintainability often conflict with each other (Lawrence et al., 2018).

Transformation of existing codes: To date there have been various approaches to this problem. NEMO v.5.0 ~~is in the process of~~(Madec et al., 2024) has adopt~~ing~~ed the PSyclone code-transformation tool (Adams et al., 2019) that enables an HPC expert to transform Fortran source code such that it may be executed on GPU using whichever programming model (i.e. OpenACC or OpenMP) is required. Previous, unpublished work ~~F~~found that for a low-resolution, (1 degree) global mesh, a single NVIDIA V100 GPU~~ gives a~~ perform~~ance~~ed some 3.6x better than an HPC-class Intel socket. For a high-resolution, (1/12th degree) global mesh, ~90 A100 GPUs g~~i~~ave the same performance as ~270 Intel sockets~~ (Porter et al., 2023 - in prep.)~~. In both cases this is an ocean-only configuration with virtually all compute being performed on the GPUs. This is important since any computation happening on the CPU incurs substantial data-transfer costs as data is moved from the GPU to the CPU, updated, and then transferred back to the GPU. (The advent of hardware support for unified CPU/GPU memory should reduce the cost of this.) As noted earlier, ICON-O is being extended manually with OpenACC directives ~~(although these are only supported on NVIDIA hardware~~). There are examples of recent (i.e. experimental) models that have moved away from Fortran in favor of higher-level programming approaches. Thetis (Kärnä et al., 2018) implements a Discontinuous Galerkin method for solving the 3D hydrostatic equations using the Firedrake framework. This permits the scientist to express their scheme in the Python implementation of Unified Form Language (Alnæs et al., 2014). The necessary code is then generated automatically. The Veros model (Häfner et al., 2021) takes a slightly different approach: its dynamical core is a direct Python translation of a Fortran code and thus retains explicit MPI parallelisation. The JAX system (http://github.com/google/jax) for Python is then used to generate performant code for both CPU and GPU. The authors report that the Python version running on 16 A100 GPUs gives the same performance as 2000 CPU cores for the Fortran version (although this comparison is slightly unfair as the CPUs used are several generations older than the GPUs).

Performance portability tools: Another popular approach to performance portability is to implement a model using a framework that takes care of parallel execution on a target platform. Frameworks such as Kokkos (Carter Edwards et al., 2014), SyCL and OpenMP are good examples and the new "Omega" ocean component of E3SM mentioned previously is being developed to use Kokkos. In principle this approach retains single-source science code, while enabling portability to a variety of different hardware. However, it is hard to insulate the oceanographer from the syntax of the framework (which are often only available in C++) and, while the framework may be portable, obtaining good performance often requires that it be used in a different way from one platform to another. In OpenMP for instance, the directives needed to parallelise a code for a multi-core CPU are not the same as those needed to offload code to an accelerator.

New programming languages: The Climate Modeling Alliance (CliMA) has adopted a radically new approach by rewriting ocean and atmospheric models from scratch using the programming language Julia (Perkel, 2019; Sridhar et al., 2022). Designed to overcome the “two-language problem” (Churavy et al., 2022), Julia is ideally suited to harness emerging HPC architectures based on GPUs (Besard et al., 2017; Bezanson et al., 2017). First results with CliMA’s ocean model, Oceananigans.jl (Ramadhan et al., 2020), run on 64 NVIDIA A100 GPUs exhibit 10 Simulation Years Per Day (SYPD) at 8 km horizontal resolution (Silverstri et al., 2024). This performance is similar to current-generation CPU-based ocean climate models run at much coarser resolution (order 25-50 km resolution). Similarly promising benchmarks have been obtained with a barotropic configuration of a prototype of the MPAS-Ocean model, rewritten in Julia (Bishnu ~~Strauss~~ et al., 2023). Such performance gains hold great promise for accelerating operational ocean prediction at high spatial resolution run on emerging HPC hardware.

Toward energy efficient simulations: Increased resolution, process representation, and data intensity in ocean and climate modeling is vastly expanding the need for compute cycles (more cores and smaller time steps). As a result, the ocean, atmosphere, and climate modeling community has recognized the need for their simulations to become more energy efficient and reduce their carbon footprint (Loft, 2020; Acosta et al., 2024; Voosen, 2024). Owing to their architecture, GPUs can play a significant role ~~at~~in reducing energy ~~requirement~~requirements. A related research frontier being spearheaded by the atmospheric modeling community is the use of mixed or reduced precision to speed up simulations (Freytag et al., 2022; Klöwer et al, 2022; Paxton et al., 2022), with a potentially desirable side effect of natively capturing stochastic parameterizations (Kimpson et al., 2023). GPUs are ideally suited for such approaches, but successful implementation depends heavily on the model's numerical algorithms.

Data-driven operational ocean forecasting: Operational weather and ocean forecasting are facing the potential of a paradigm shift with the advent of powerful, purely data-driven methods. The numerical weather prediction (NWP) community has spearheaded the development of machine learning-based emulators that perform several orders of magnitudes faster than physics-based models (e.g., Bouallègue et al., 2024; Rasp et al., 2024). Such emulators have the potential to revolutionize probabilistic forecasting and uncertainty quantification, among others. The computational patterns underlying the ML algorithms, such as parallel matrix multiplication, are ideally suited for general-purpose GPU architectures. Whereas these methods have been driven to a large extent by private sector entities and require access to increasingly large GPU-based HPC systems for training, corresponding efforts in operational ocean forecasting are only now beginning to catch up. A review of the rapidly changing landscape of AI methods in the context of ocean forecasting is attempted in Heimbach et al. (2024).

~~The focus of this paper is on the use of GPUs to accelerate traditional, numerical simulations of the ocean. However, we also note that~~

References

Acosta, M. C., Palomas, S., Ticco, S. V. P., Utrera, G., Biercamp, J., Bretonniere, P.-A., Budich, R., Castrillo, M., Caubel, A., Doblas-Reyes, F., Epicoco, I., Fladrich, U., Joussaume, S., Gupta, A. K., Lawrence, B., Sager, P. L., Lister, G., Moine, M.-P., Rioual, J.-C., … Balaji, V.: The computational and energy cost of simulation and storage for climate science: lessons from CMIP6. Geoscientific Model Development, 17(8), 3081–3098. https://doi.org/10.5194/gmd-17-3081-2024, 2024.

Adams, S. V., Ford, R. W., Hambley, M., Hobson, J. M., Kavcic, I., Maynard, C. M., Melvin, T., Mueller, E. H., Mullerworth, S., Porter, A. R., Rezny, M., Shipway, B. J. and Wong, R.: LFRic: Meeting the challenges of scalability and performance portability in Weather and Climate models. Journal of Parallel and Distributed Computing, 132, 383-396. https://doi.org/10.1016/j.jpdc.2019.02.007, 2019.

Adcroft, A., Anderson, W., Balaji, V., Blanton, C., Bushuk, M., Dufour, C. O., Dunne, J. P., Griffies, S. M., Hallberg, R., Harrison, M. J., Held, I. M., Jansen, M. F., John, J. G., Krasting, J. P., Langenhorst, A. R., Legg, S., Liang, Z., McHugh, C., Radhakrishnan, A., … Zhang, R.: The GFDL Global Ocean and Sea Ice Model OM4.0: Model Description and Simulation Features. Journal of Advances in Modeling Earth Systems, 11(10), 3167–3211. https://doi.org/10.1029/2019ms001726, 2019.

Alnæs, M. S., Logg, A., Ølgaard, K. B., Rognes, M. E. and Wells, G. N.: Unified Form Language: A Domain-Specific Language for Weak Formulations of Partial Differential Equations, ACM Trans. Math. Softw., 40(2), doi:10.1145/2566630

Carter Edwards, H., Trott, C.R., and Sunderland, D. (2014). Kokkos: enabling manycore performance portability through polymorphic memory access patterns, Journal of Parallel and Distributed Computing, 74(12), 3202-3216. https://www.osti.gov/servlets/purl/1106586, 2014.

Balaji, V.: Climbing down Charney's ladder: machine learning and the post-Dennard era of computational climate science. Philosophical Transactions of the Royal Society A, 379(2194), 20200085. https://doi.org/10.1098/rsta.2020.0085, 2021.

Besard, T., Foket, C. & Sutter, B. D.: Effective Extensible Programming: Unleashing Julia on GPUs. IEEE Transactions on Parallel and Distributed Systems, 30(4), 827–841. https://doi.org/10.1109/tpds.2018.2872064, 2017.

Bezanson, J., Edelman, A., Karpinski, S. and Shah, V. B.: Julia: A Fresh Approach to Numerical Computing. SIAM Review, 59(1), 65–98. https://doi.org/10.1137/141000671, 2017.

Bishnu, S., Strauss, R. R. & Petersen, M. R.: Comparing the Performance of Julia on CPUs versus GPUs and Julia-MPI versus Fortran-MPI: a case study with MPAS-Ocean (Version 7.1). Geoscientific Model Development, 16(19), 5539–5559. https://doi.org/10.5194/gmd-16-5539-2023;, 2023.

Bouallègue, Z. B., Clare, M. C. A., Magnusson, L., Gascón, E., Maier-Gerber, M., Janoušek, M., Rodwell, M., Pinault, F., Dramsch, J. S., Lang, S. T. K., Raoult, B., Rabier, F., Chevallier, M., Sandu, I., Dueben, P., Chantry, M. & Pappenberger, F.: The Rise of Data-Driven Weather Forecasting: A First Statistical Assessment of Machine Learning–Based Weather Forecasts in an Operational-Like Context. Bulletin of the American Meteorological Society, 105(6), E864–E883. https://doi.org/10.1175/bams-d-23-0162.1, 2024.

Chassignet, E.P., Hurlburt, H.E., Metzger, E.J., Smedstad, O.M., Cummings, J., Halliwell, G.R., Bleck, R., Baraille, R., Wallcraft, A.J., Lozano, C., Tolman, H.L., Srinivasan, A., Hankin, S., Cornillon, P., Weisberg, R., Barth, A., He, R., Werner, F., and Wilkin, J.: U.S. GODAE: Global Ocean Prediction with the HYbrid Coordinate Ocean Model (HYCOM). Oceanography, 22(2), 64-75. https://doi.org/10.5670/oceanog.2009.39, 2009.

Churavy, V., Godoy, W. F., Bauer, C., Ranocha, H., Schlottke-Lakemper, M., Räss, L., Blaschke, J., Giordano, M., Schnetter, E., Omlin, S., Vetter, J. S. & Edelman, A.: Bridging HPC Communities through the Julia Programming Language. arXiv. https://doi.org/10.48550/arxiv.2211.02740, 2022.

Dennard, R. H., Gaensslen, F., Yu, H., Rideout, L., Bassous, E., and LeBlanc, A.: Design of ion-implanted MOSFET's with very small physical dimensions. IEEE Journal of Solid-State Circuits. SC-9 (5), 256–268, 1974.

Draeger, E. W. and Siegel, A.: Exascale Was Not Inevitable; Neither Is What Comes Next. Computing in Science and Engineering, 25(3), 79–83. https://doi.org/10.1109/mcse.2023.3311375, 2023.

Freytag, G., Lima, J. V. F., Rech, P. & Navaux, P. O. A. Impact of Reduced and Mixed-Precision on the Efficiency of a Multi-GPU Platform on CFD Applications. Lecture Notes in Computer Science, 570–587. https://doi.org/10.1007/978-3-031-10542-5_39, 2022.

Häfner, D., Nuterman, R. and Jochum, M.: Fast, cheap, and turbulent — Global ocean modeling with GPU acceleration in Python. Journal of Advances in Modeling Earth Systems, 13. doi:10.1029/2021MS002717, 2021.

Heimbach, P., F. O'Donncha, T. A. Smith, J.M. Garcia-Valdecasas, A. Arnaud, and L. Wan: Crafting the Future: Machine Learning for Ocean Forecasting. State of the Planet, submitted. doi:10.5194/sp-2024-18, 2024.

Kärnä, T., Kramer, S. C., Mitchell, L., Ham, D. A., Piggott, M. D. and Baptista, A. M. .: Thetis coastal ocean model: discontinuous Galerkin discretization for the three-dimensional hydrostatic equations. Geoscientific Model Development, 11(11), 4359-4382. https://doi.org/10.5194/gmd-11-4359-2018, 2018.

Kimpson, T., Paxton, E. A., Chantry, M. & Palmer, T.: Climate-change modelling at reduced floating-point precision with stochastic rounding. Quarterly Journal of the Royal Meteorological Society, 149(752), 843–855. https://doi.org/10.1002/qj.4435, 2023.

Klöwer, M., Hatfield, S., Croci, M., Düben, P. D. & Palmer, T. N.: Fluid Simulations Accelerated With 16 Bits: Approaching 4x Speedup on A64FX by Squeezing ShallowWaters.jl Into Float16. Journal of Advances in Modeling Earth Systems, 14(2), e2021MS002684. https://doi.org/10.1029/2021ms002684, 2022.

Korn, P.: Formulation of an unstructured grid model for global ocean dynamics, Journal of Computational Physics, 339, 525-552. https://doi.org/10.1016/j.jcp.2017.03.009, 2017.

Lawrence, B. N., Rezny, M., Budich, R., Bauer, P., Behrens, J., Carter, M., Deconinck, W., Ford, R., Maynard, C., Mullerworth, S., Osuna, C., Porter, A., Serradell, K., Valcke, S., Wedi, N. and Wilson, S.: Crossing the chasm: how to develop weather and climate models for next generation computers? Geoscientific Model Development, 11(5), 1799-1821. https://doi.org/10.5194/gmd-11-1799-2018, 2018.

Loft, R.: Earth System Modeling Must Become More Energy Efficient. Eos, Transactions American Geophysical Union, 101. https://doi.org/10.1029/2020eo147051, 2020.

~~Madec, G. and NEMO System Team: NEMO ocean engine, Scientific Notes of Climate Modelling Center, 27, Institut Pierre-Simon Laplace (IPSL), doi:10.5281/zenodo.1464816, 2023.~~Madec, G. and the NEMO System Team, 2024. NEMO Ocean Engine Reference Manual, doi:10.5281/zenodo.1464816, 2024.

Marshall, J., Adcroft, A., Hill, C., Perelman, L., and Heisey, C.: A finite-volume, incompressible Navier Stokes model for studies of the ocean on parallel computers. J. Geophysical Res., 102(C3), 5753-5766. https://doi.org/10.1029/96JC02775, 1997.

Moore, A. M., Arango, H. G., Lorenzo, E. D., Cornuelle, B. D., Miller, A. J. & Neilson, D. J.: A comprehensive ocean prediction and analysis system based on the tangent linear and adjoint of a regional ocean model. Ocean Modelling, 7(1–2), 227–258. https://doi.org/10.1016/j.ocemod.2003.11.001, 2004.

Paxton, E. A., Chantry, M., Klöwer, M., Saffin, L. & Palmer, T.: Climate Modeling in Low Precision: Effects of Both Deterministic and Stochastic Rounding. Journal of Climate, 35(4), 1215–1229. https://doi.org/10.1175/jcli-d-21-0343.1, 2022.

Perkel, J. M.; Julia: come for the syntax, stay for the speed. 141–142. http://www.nature.com/articles/d41586-019-02310-3, 2019.

Petersen, M. R., Private communication, 2024.

~~Porter, A. R., Dearden, C., Ford, R. W., Henrichs, J., Siso, S., Nobre, N., Müller, S. A., Coward, A., Bell, M. (2023). Using PSyclone 3.4 to achieve Performance Portability for NEMO 4.0 and NEMO-MEDUSA Ocean Models, in preparation.~~

Ringler, T. Petersen, M., Higdon, R. L., Jacobsen, D., Jones, P. W. and Maltrud, M.: A multi-resolution approach to global ocean modeling, Ocean Modelling, 69, 211-232. https://doi.org/10.1016/j.ocemod.2013.04.010, 2013.

Shchepetkin, A. F., and McWilliams, J.C.: A method for computing horizontal pressure-gradient force in an oceanic model with a nonaligned vertical coordinate. Journal of Geophysical Research: Oceans, 108(C3), 3090. https://doi.org/10.1029/2001JC001047, 2003.

Ramadhan, A., Wagner, G., Hill, C., Campin, J.-M., Churavy, V., Besard, T., Souza, A., Edelman, A., Ferrari, R. & Marshall, J.: Oceananigans.jl: Fast and friendly geophysical fluid dynamics on GPUs. Journal of Open Source Software, 5(53), 2018. https://doi.org/10.21105/joss.02018, 2020.

Rasp, S., Hoyer, S., Merose, A., Langmore, I., Battaglia, P., Russell, T., Sanchez-Gonzalez, A., Yang, V., Carver, R., Agrawal, S., Chantry, M., Bouallegue, Z. B., Dueben, P., Bromberg, C., Sisk, J., Barrington, L., Bell, A. & Sha, F.:

WeatherBench 2: A Benchmark for the Next Generation of Data-Driven Global Weather Models. Journal of Advances in Modeling Earth Systems, 16(6). https://doi.org/10.1029/2023ms004019, 2024.

Rupp, K: Microprocessor Trend Data, https://github.com/karlrupp/microprocessor-trend-data, 2022. Accessed 12/09/2024.

Silvestri, S., Wagner, G. L., Constantinou, N. C., Hill, C. N., Campin, J.-M., Souza, A. N., Bishnu, S., Churavy, V., Marshall, J. C. and Ferrari, R.: A GPU-based ocean dynamical core for routine mesoscale-resolving climate simulations. ESSOAr. https://doi.org/10.22541/essoar.171708158.82342448/v1, 2024.

Sridhar, A., Tissaoui, Y., Marras, S., Shen, Z., Kawczynski, C., Byrne, S., Pamnany, K., Waruszewski, M., Gibson, T. H., Kozdon, J. E., Churavy, V., Wilcox, L. C., Giraldo, F. X. & Schneider, T.: Large-eddy simulations with ClimateMachine v0.2.0: a new open-source code for atmospheric simulations on GPUs and CPUs. Geoscientific Model Development, 15(15), 6259–6284. https://doi.org/10.5194/gmd-15-6259-2022, 2022.

~~Strauss, R. R., Bishnu, S. and Petersen, M. R.: Comparing the Performance of Julia on CPUs versus GPUs and Julia-MPI versus Fortran-MPI: a case study with MPAS-Ocean (Version 7.1). Geosci. Model Dev., 16, 5539-5559, https://doi.org/10.5194/gmd-16-5539-2023, 2023 EGUsphere, 2023, 1–22. https://doi.org/10.5194/egusphere-2023-57, 2023.~~

Strohmaier, E., Dongarra, J., Simon, H., Meuer, M. and Meuer, H.: The Top 500. https://www.top500.org/lists/top500/list/202~~2~~4/11/, 202~~2~~4.

Tsujino, H., Motoi, T., Ishikawa, I., Hirabara, M., Nakano, H., Yamanaka, G., Yasuda, T., and Ishizaki, H.: Reference manual for the Meteorological Research Institute Community Ocean Model (MRI.COM) version 3. Technical Reports of the Meteorological Research Institute, 59, 273, 2010.

Voosen, P.: Climate modelers grapple with their own emissions. Science, 384(6695), 494–495. https://doi.org/10.1126/science.adq1772, 2024.

Competing interests

Author A. Porter has declared that he is an author of the PSyclone package and a co-chair of the NEMO HPC Working Group.

Data and/or code availability

No data or code is associated with this work.~~This can also be included at a later stage, so no problem to define it for the first submission.~~

Authors contribution

~~This can also be included at a later stage, so no problem to define it for the first submission.~~AP created the first draft of this work. PH assisted with updating the text in the light of the reviews received.

Acknowledgements

~~This can also be included at a later stage, so no problem to define it for the first submission.~~The authors would like to thank the reviewers for bringing the E3SM work to their attention.