DEMSI: A Performance Portable Sea Ice Model

Kara Peterson[1], Adrian Turner[2], Paul Kuberry[1], Dan Bolintineanu[1], Andrew Roberts[2], Travis Davis[3], and Min Wang[2]

[1]Sandia National Laboratories, [2]Los Alamos National Laboratory, [3]Naval Postgraduate School

U.S. DEPARTMENT OF ENERGY
Office of Science

Overview

Sea ice is an important component of the Earth's climate system. It reflects solar radiation, insulates the ocean from atmosphere, and influences ocean circulation. Sea ice is composed of rigid plates called floes that are driven by surface winds and ocean currents.

The sea ice component the Energy Exascale Earth System Model (E3SM) treats the sea ice cover as a continuum with internal forces given by an isotropic constitutive model. Observations suggest that these approximations are not effective at the high-resolutions used now in Earth system modeling. To overcome these deficiencies we are developing a new sea ice dynamical core based on the Discrete Element Method (DEM).

The objective of this project is to develop a computationally efficient DEM sea ice model suitable for inclusion as the sea-ice component in E3SM and capable of running on next-generation architectures.

DEMSI

A complete model of sea ice must include the dynamics of the interacting floes and the seasonal thermodynamic response to incoming solar and thermal radiation. Each element in DEMSI represents a collection of floes that enables the capture of the anisotropic, heterogeneous nature of sea ice deformation. In addition, each element represents a region of sea ice with varying thickness described by an ice thickness distribution. Two well-established libraries have been leveraged by DEMSI to facilitate the development of the dynamical core and thermodynamics:

- ***Large-scale Atomic/Modelcular Massively Parallel Simulator (LAMMPS)*** (, 1995)
  - Particle based molecular dynamics code
  - Includes support for DEM and history-dependent contact models
  - Computationally efficient
- ***CICE consortium Icepack library*** (, 2018)
  - State-of-the-art sea ice column thermodynamics package
  - Vertical thermodynamics, salinity, shortwave radiation,snow, melt ponds, ice thickness distribution, biogeochemistry

Challenges

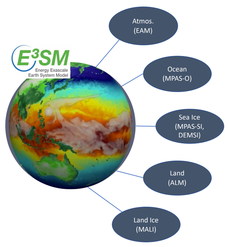
Performance: How to make a DEM sea ice model fast enough for global Earth system applications?

Incorporation of DEM sea-ice models into ESMs has been impeded partly by the large computational requirements of these models, which have previously been used primarily for regional-scale and short-term simulations.

To address this challenge we are leveraging the well-tested computational performance of LAMMPS for the DEM dynamical core. The use of advanced accelerated computing architectures, currently being adopted by leadership computing facilities, offers an opportunity to improve DEM computational performance by exploiting the increased on-node parallelism.

Interpolation and Coupling: How to couple Lagrangian particles to an Eulerian mesh efficiently and conservatively?

ESM components are typically discretized on a fixed grid making interpolation and coupling of the components relatively straightforward. The Lagrangian nature of DEM makes it more difficult to accurately and conservatively map sea-ice particle data to a fixed ocean or atmosphere mesh.

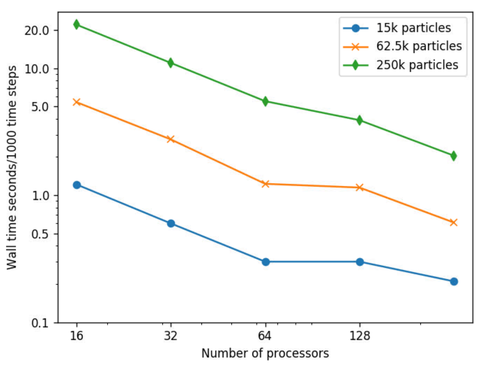


To address this challenge we are incorporating a moving least squares method to interpolate from particles to grid and investigating an optimization-based algorithm to enforce conservation and bounds preservation.

MPI Performance

LAMMPS has been designed for massive parallel performance and DEMSI inherits its parallel infrastructure. The initial implementation of the DEMSI sea ice contact model without Kokkos data structures achieves good scaling for pure MPI performance.

The results below are from a simple shear and compression test case for a lattice arrangement of monosized particles. These initial results suggest that DEMSI will be competitive with the sea ice model currently implemented within E3SM. There are opportunities for additional performance increase by using the Kokkos library on new computing architectures and decreasing stiffness to allow for larger time step.

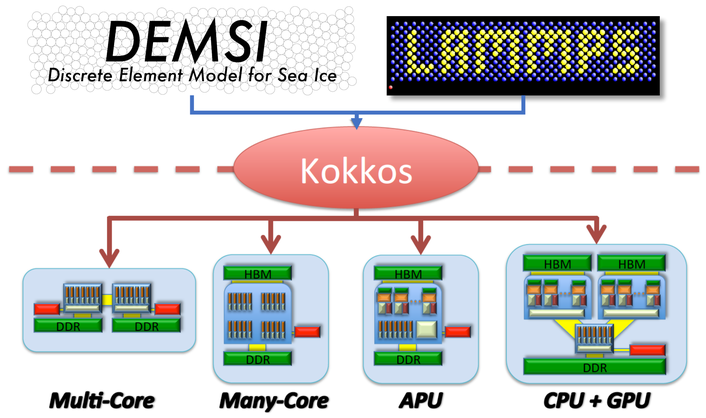


Kokkos Implementation

To enable performance portability of the DEMSI code base, we have designed the code to use Kokkos, an open-source C++ library providing a programming model that allows for a single algorithm implementation to be compatible with multiple HPC architectures (, 2014).

Kokkos supplies computational kernels associated with an execution space and multi-dimensional arrays associated with memory spaces, which have a polymorphic data layout that depends on the device architecture. This generic framework allows Kokkos to provide optimized code for particular instantiations of the execution and memory spaces. The Kokkos library has been successfully implemented with other E3SM codes, including a refactored version of the atmospheric dynamical core (, 2019).

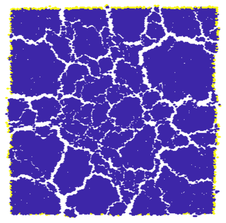
DEMSI has been designed from the start to use Kokkos multidimensional arrays and more recent work has included an effort to convert all particle loops in the code to Kokkos `parallel_for` and `parallel_reduce` execution modes. Some recent work has been done to convert portions of the LAMMPS DEM implmentation to Kokkos. We leveraged this existing capability as much as possible in the DEMSI development while converting additional LAMMPS code as needed including new sea ice contact models and data structures.



https://github.com/kokkos

Preliminary GPU Performance

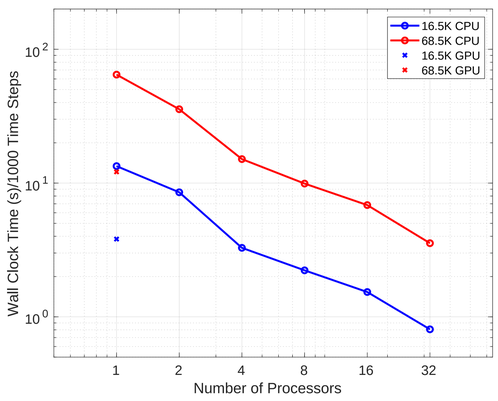
The Kokkos-refactored DEMSI code has been run in Serial, OpenMP, and CUDA execution spaces. For the CUDA implementation, we have built the code on White, a Heterogenous Advanced Architecture Platform (HAAP) at Sandia with eight Garrison dual socket Power8 compute nodes and Nvidia Tesla P100 GPUs.



The refactored code has been tested on a uniform stress test case where an initial polydisperse random packing of particles undergoes fracture. We ran two cases that differed in the number of particles (16.5K versus 65.8K) and varied the number of processors for MPI-only runs.

On the White testbed we ran each configuration on a single GPU. The total number of timesteps for this example is 150,000 and a timestep of 0.04 seconds was used.

The wall clock time per 1000 time steps for MPI runs with varying processor counts is shown below. These results are for MPI-only runs with no threading on an Intel Xenon E7-4480 v2 CPU. The two additional points for one processor indicate performance for a single GPU node on White. The code scales well for the small processor counts in this example. We also note that we achieve a substantial speedup on GPU over a single node CPU with a 3.5 times speedup for the 16.5K particle simulation and 5.3 times speedup for the 65.8K simulation.



Acknowledgements

Support for DEMSI is provided through the Scientific Discovery through Advanced Computing (SciDAC) program funded by the US Department of Energy (DOE), Office of Science, Biological and Environmental Research, and Advanced Scientific Computing Research programs.

Particle-to-Grid Interpolation

One challenge in coupling a Lagrangian particle code into the E3SM framework is to accurately and conservatively map sea-ice particle data to a fixed ocean or atmosphere mesh. Our solution is a moving least squares (MLS) interpolation method within DEMSI. An MLS approach (, 2005) enables an accurate reconstruction of a field from a set of scattered data points without the need for any grid structure.

Given a set of element or particle locations $\{\mathbf{x}\}_{e=1}^{N}$ and function values $f_e = f(\mathbf{x}_e)$ we can find a local approximation of the function $f^h(\mathbf{x})$ that is exact for all $q \in P_m$, where P_m is the space of polynomials of degree less than equal to m. We can write this function as a sum over the particles as

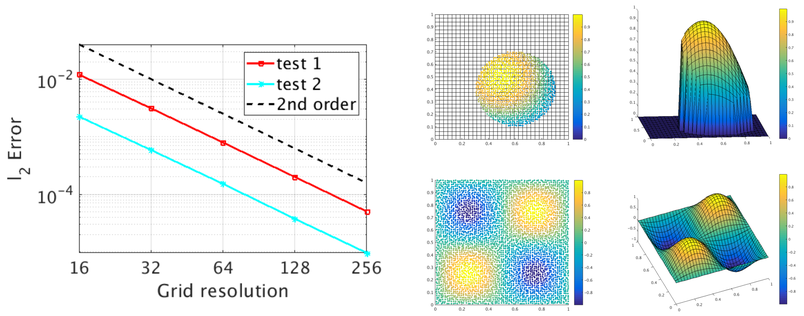
$$f^h(\mathbf{x}_e) = \sum_e \psi_e(\mathbf{x}_e) f_e \,.$$

The function ψ_e depends on the polynomial basis function for the reconstruction. We choose the linear basis vector for the DEMSI implementation of the form $b^T(\mathbf{x}_e) = [1(x - x_e)(y - y_e)]$. Given a moment matrix that depends on the polynomial basis and a weight function w as $M = \sum_e bb^T w(\mathbf{x}_e)$ we obtain $\psi_e(\mathbf{x}_e) = bM^{-1}b^T w(\mathbf{x}_e)$.

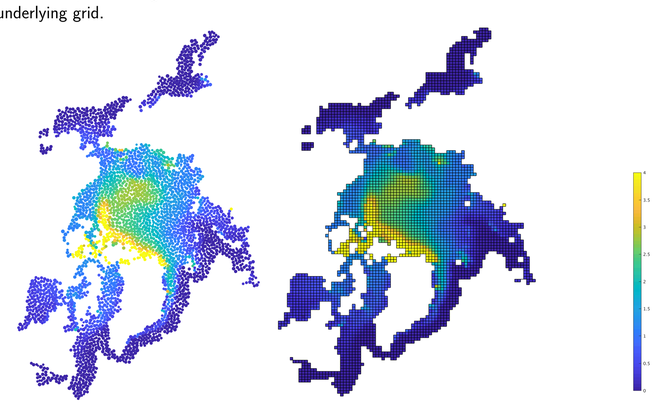
The linear MLS coupling is not computationally expensive, but the method has been implemented in DEMSI with Kokkos data structures and a test case has been run on GPUs as well as CPUs. Future improvements to the implementation will investigate more general MLS interpolation weights that can be chosen to ensure the particle distribution within the support is sufficient to achieve a given approximation order. Additionally, higher-order MLS interpolation can be used, which would increase the computational expense, but provide benefits in accuracy and more work for on-node parallelism.

Interpolation Testing

The MLS coupling method has been tested by interpolating simple smooth functions from particles to grid. Particles are initialized with a random perturbation from a structured arrangement with approximately four particles per grid cell. For testcases where the particles fill the domain and where particles are located in only part of the domain we see the expected second-order convergence for linear MLS.



The implementation in DEMSI has been completed for interpolation from particles to a structured grid. We have tested the interpolation algorithm on an Arctic basin test case where particle sea ice thickness has been mapped to the underlying grid.



Next Steps

Active development in continuing in DEMSI. As new DEM contact models and other enhancements are implemented they are being converted to Kokkos and tested on multiple platforms. In support of realistic Arctic basin simulations and future global simulations we will be extending our coupling implementation and working to optimize performance. Particular future plans include

- Complete more rigorous performance testing on different platforms
- Optimize code performance on leadership systems such as Cori and Summit
- Investigate load balancing and test performance within E3SM

- Enable coupling on more general grids using Compadre Toolkit (, 2019)
- Implement optimization-based property preservation.

References

Bertagna, L., M. Deakin, O. Guba, D. Sunderland, A. Bradley, I. Teazur, M. Taylor, and A. Salinger (2019), Hommexx 1.0: a performance portable atmosphere dynamical core for the energy exascale earth system model, *GMD*, *12*, 1423–1441, doi:doi:10.5194/gmd-12-1423-2019.

Edwards, H. C., C. Trott, and D. Sunderland (2014), Kokkos: enabling manycore performance portability through polymorphic memory accesss patterns, *J. Parallel Distrib. Comput.*, *74*, 3202–3216.

Hunke, E., R. Allard, D. Bailey, P. Blain, T. Craig, A. Damsgaard, F. Dupont, A. DuVivier, R. Grumbine, M. Holland, N. Jeffery, J.-F. Lemieux, A. Roberts, M. Turner, and M. Winton (2018), Cice consortium/icepack version 1.1.0 (version icepack1.1.0), *Zenodo*, doi: 10.5281/zenodo.1891650.

Kuberry, P., P. Bosler, and N. Trask (2019), Compadre toolkit, doi:10.5281/zenodo.2560287.

Plimpton, S. (1995), Fast parallel algorithms for short-range molecular dynamics, *J. Comp. Phys.*, *117*, 1–19.

Wendlund, H. (2005), *Scattered Data Approximation*, Cambridge University Press.