The Journal of Open Source Software

PhaseFieldPet: An Open-Source Phase Field Modeling Software for Heterogeneous Architectures

Shore Salle Chota [1,4¶], **Martin Reder** [1,2], **Daniel Schneider** [1,2], **Harald Koestler** [4], **and Britta Nestler** [1,2,3]

1 Institute for Applied Materials (IAM), Karlsruhe Institute of Technology, Karlsruhe, 76131, Germany **2** Institute of Digital Materials Science (IDM), Karlsruhe University of Applied Sciences, Karlsruhe, 76133, Germany **3** Institute of Nanotechnology (INT), Karlsruhe Institute of Technology, Karlsruhe, 76131, Germany **4** Chair for System Simulation, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Phase field method has emerged as a powerful computational tool for simulating complex, evolving interfaces at mesoscale in materials science and Engineering , fluid dynamics, cell migration and other fields. However, achieving scalability and efficiency for multicomponent multiphase across diverse hardware architectures remains a challenge. This paper presents PhaseFieldPet, an open-source, message passing interface (mpi) based software package for large-scale phase field simulations, specifically designed to leverage heterogeneous architectures such as CPUs and GPUs. PhaseFieldPet is built upon the Portable, Extensible Toolkit for Scientific Computation (PETSc), to efficiently handle large-scale simulations on high-performance computing platforms. The software's modular design facilitates and easily integrates various phase field models such as Multiphase-Field and Multi-Order Parameter models with various choice of gradient and potential energy contributions. Performance benchmarks demonstrate the software's capability to handle simulations involving from small to millions of degrees of freedom with excellent scalability.

The Phase Field Equation

The phase-field approach represents surfaces and interfaces implicitly using continuous scalar fields (order parameter) $\phi_\alpha(\mathbf{r},t)$, $\alpha \in \{1,2,\dots,N\}$, which is constant in bulk phases and transition smoothly—but sharply—across a diffuse boundary. $\phi_\alpha(\mathbf{r},t)$ represents different grains or phases such as solid, liquid, gas and their state like crystallographic orientation, polarization, volume fraction. For example, in a solid-solid phase transition, a solid phase can be represented by N order parameters $\phi_\alpha(\mathbf{r},t)$ based on N crystallographic orientations or grains.

Microstructure evolution (and hence evolution of order parameter $\phi_\alpha(\mathbf{r},t)$) can be obtained from functionals of entropy, or free energy, or grand potential (Hötzer et al., 2018). Following energy functional, one can write total energy functional of the system as

$$\mathcal{F}(\phi,\nabla\phi) = \int_V f dV = \int_V f_{grad}(\phi,\nabla\phi) + f_{pot}(\phi) + f_{bulk}(\phi,\dots)\, dV$$

The first two terms contributes to interfacial energy , and exists during phases are in contact with each other and is responsible to keep the diffuse interface at finite width with interplay of gradient energy density $f_{grad}(\phi,\nabla\phi)$ which diffuses the interfaces, while the potential term $f_{pot}(\phi)$counter acts it. The bulk contribution $f_{bulk}(\phi,\dots)$ can be of various type depending

on the problem at hand such as chemical, thermal, mechanical, electrical, elastic, magnetic and etc.

There exists various formulations to $f_{grad}(\phi, \nabla\phi)$ and $f_{pot}(\phi)$ by many scholars in the Phase field community (Daubner et al., 2023). Nestler et al. (2005) formulated these terms in their PACE3D software (Hötzer et al., 2018) by

$$f_{grad}(\phi, \nabla\phi) = \varepsilon \sum_{\alpha} \sum_{\beta>\alpha} \gamma_{\alpha\beta} |\phi_\alpha \nabla\phi_\beta - \phi_\beta \nabla\phi_\alpha|^2$$

$$f_{pot}(\phi) = \frac{16}{\varepsilon\pi^2} \sum_{\alpha} \sum_{\beta>\alpha} \gamma_{\alpha\beta}\phi_\alpha\phi_\beta + \frac{1}{\varepsilon} \sum_{\alpha} \sum_{\beta>\alpha} \sum_{\delta>\beta} \gamma_{\alpha\beta\delta}\phi_\alpha\phi_\beta\phi_\gamma$$

See Daubner et al. (2023) table B.4 for other formulations of these terms and their explanations currently considered in PhaseFieldPet.

Phase field evolution equation is in general given by Allen-Cahn approach for each order parameter following total energy minimization principle of the system by

$$\frac{\partial\phi_\alpha}{\partial t} = -L\frac{\delta\mathcal{F}}{\delta\phi_\alpha} = -L\left(\frac{\partial f}{\partial\phi_\alpha} - \nabla\cdot\frac{\partial f}{\partial\nabla\phi_\alpha}\right)$$

Where L is kinetic coefficient. We differentiate Phase field equation based on whether order parameters are conserved (where each order parameter represents volume fraction and hence the sum of all phase fields at a point in space is 1), which are called Multi Phase Field (MPF) models and and those that do not put such a conservation requirement directly, called Multi order Parameter model (MOP). All these models and variants are treated and solved in PhaseFieldPet, see the usage section below.

Statement of need

For the past couple of decades, phase field software has been being developed and used with in house codes, and Open source phase field software started to be available from 2007 (Hong & Viswanathan, 2020). Many existing open source softwares are limited to one or two spatial dimensions, focus on binary systems, use only one type time step solver (usually explicit time stepping), works only on one CPU core (serial code) or are not capable of using heterogeneous compute resources available such as GPUs for compute and energy efficiency. Notable large scale, distributed computing capable open source phase field softwares include: The open source Multiphysics Object Oriented Simulation Environment (MOOSE) (Schwen et al., 2017) - which is a powerful toolset for implementing phase field models using the finite element method and targets CPUs, PRISMS-PF - massively parallel finite element code for conducting phase field (DeWitt et al., 2020), OpenPhase (Tegeler et al., 2017) uses finite difference for spatial discretization and uses explicit time stepping algorithm and targets CPUs. Among non open source software, distributed machines capable software is a Parallel Algorithms for Crystal Evolution in 3D (PACE3D) (Hötzer et al., 2018) is a multiphase field software that uses explicit time stepping along with finite difference spatial discretizations.

PhaseFieldPet is a Finite difference based software built on top of TS solver from PETSc (Abhyankar et al., 2018) and it extends our previous work (Daubner et al., 2023) to 3D and N phases, and fills the above mentioned gaps in existing softwares. Among others:

1. It paves the way to opensource PACE3D.
2. Allows multiphase simulation in 1D / 2D / 3D.

3. Decouple the numerical solution methods from the physical modeling such that one can choose various solution methods without restricting to one time step solver (i.e. one can use methods like semi implicit, implicit time stepping algorithms, various underlying nonlinear solver , linear solvers and preconditioners, etc) based on composability features of PETSc (Balay et al., 2024). This makes it easier for newcomers to the phase field community and advanced users alike. We formulate the phase field equation using the Implicit Explicit (IMEX) scheme such that the either the whole Phase field equation is treated implicitly or the stiff part, allowing longer time steps compared to the explicit methods which require small time steps for stability.

4. Works on single core, multicore to multi node High Performance Computing cluster/supercomputer coupled with accelerators such as GPUs (Mills et al., 2021). GPUs are increasingly available for computing purposes with thousands of compute cores and small energy usage. We can leverage their compute power using PhaseFieldPet from one code base for both CPUs and GPUs.

Usage

To use PhaseFieldPet, all you need is to install PETSc (Balay et al., 2024), compile it using

```
make PhaseFieldPet
```

run the executable generated with default solver settings (E.g. with 4 mpi process) by

```
mpiexec -n 4 PhaseFieldPet
```

By default PhaseFieldPet simulates a static triple junction, see Daubner et al. (2023), using dot gradient term (`grad_dot`) and well potential of Toth (`pot_toth`) using lagrangian based phase field equations (`pfe_mpfl`). The default time stepping solver being Adaptive Runge Kutta Implicit-Explicit (IMEX) method, where the stiff part of the equation is treated implicitly. `Pfe_mpfl` adds a lagrangian term to the phase field equation to constrain the sum of ϕ_α to be 1 at every point in the domain(Nestler et al., 2005).

To simulate with non default combinations, for instance to solve phase field equation with gradient and potential energy terms above with $f_{bulk}(\phi, ...) = 0$, we give the corresponding options to the run time as

```
mpiexec -n 4 PhaseFieldPet -grad_weighted -pot_nestler  -simplex
```

This means that we are using weighted (generalized) gradient energy formulation (`grad_weighted`), the obstacle potential (`pot_nestler`) as outlined above and apply Gibbs simplex constraint (`simplex`) to constrain each phase field $\phi_\alpha >= 0$, $\sum_\alpha \phi_\alpha = 1$ at each point in the simulation domain. One can also use different Phase field equations (pfe) with optionlike `pfe_mop` (Multiorder parameter model) which does not put any restriction on order parameters, `pfe_mpf` (Non lagrangian based Multi-Phase field Equation) (Tegeler et al., 2017) along with other gradient and potential energy contributions. For details of usage not mentioned here, including your own energy contributions, see the associated Github page to this paper.

Example Performance Result

Here we report the strong scalabilty of PhaseFieldPet for simulation static triple junction using second order, Adaptive Backward Euler (fully implicit) time step solver by increasing grid points along x and y direction using

```
mpiexec -n # PhaseFieldPet -simplex -ts_type bdf -da_grid_x 256 -da_grid_y 256
```

Figure 1 shows the result on Meggie cluster at NHR@FAU obtained by running up to 80 mpi process on 4 compute nodes, where each nodes have two Intel Xeon E5-2630v4 "Broadwell" chips (10 cores per chip) running at 2.2 GHz with 25 MB Shared Cache per chip and 64 GB of RAM. The result indicated excellent agreement with theoretical expectations that the log-log plot is a straight decreasing line with slope of -1.

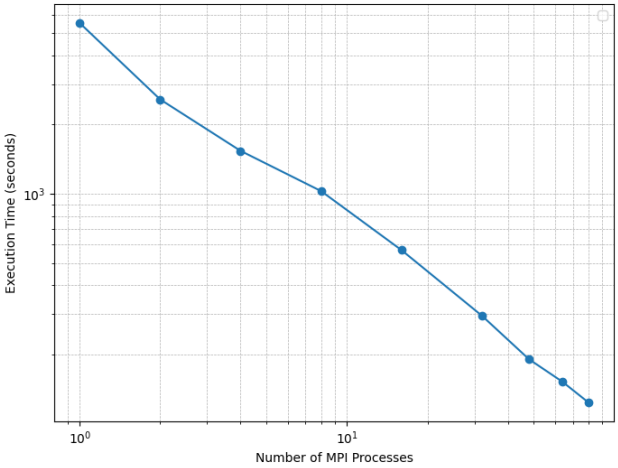

Figure 1: Log-Log Plot of Execution Time vs MPI Processes.

Conclusions

PhaseFieldPET provides users with flexible methods to solve phase field equations, along with various energy contributions on heterogeneous hardware architectures. More specifically, a user can include or choose what gradient energy term, potential energy term, the type of the phase field equation, type of numerical algorithm to use in order to solve the differential equation (pfe) (along with the choice of the underlying nonlinear equation solver, Linear equation solver, preconditioners) and etc. Inline with PACE3D software (Hötzer et al., 2018) and extension of it, the future version of PhaseFieldPet will include various other modules corresponding to different applications of phase field.

Acknowledgement

We acknowledge discussions we had with Dr. Simon Daubner during the genesis of this project. The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU). The hardware is funded by the German Research Foundation (DFG).

Funding

The authors would like to thank the NHR-Verein e.V for supporting this work/project within the NHR Graduate School of National High Performance Computing (NHR).

References

Abhyankar, S., Brown, J., Constantinescu, E. M., Debojyoti Ghosh, Smith, B. F., & Zhang, H. (2018). PETSc/TS: A modern scalable ODE/DAE solver library. In *arXiv e-preprints*. https://arxiv.org/abs/1806.01437

Balay, S., Abhyankar, S., Adams, M. F., Benson, S., Brown, J., Brune, P., Buschelman, K., Constantinescu, E. M., Dalcin, L., Dener, A., Eijkhout, V., Faibussowitsch, J., Gropp, W.

D., Hapla, V., Isaac, T., Jolivet, P., Karpeev, D., Kaushik, D., Knepley, M. G., … Zhang, J. (2024). *PETSc Web page*. https://petsc.org/

Daubner, S., Hoffrogge, P. W., Minar, M., & Nestler, B. (2023). Triple junction benchmark for multiphase-field and multi-order parameter models. *Computational Materials Science*. https://doi.org/10.1016/j.commatsci.2022.111995

DeWitt, S., Rudraraju, S., D., M., Andrews, W. B., & Thornton, K. (2020). PRISMS-PF: A general framework for phase-field modeling with a matrix-free finite element method. *Npj Computational Materials*. https://doi.org/10.1038/s41524-020-0298-5

Hong, Z., & Viswanathan, V. (2020). Open-Sourcing Phase-Field Simulations for Accelerating Energy Materials Design and Optimization. *ACS Energy Letters*. https://doi.org/10.1021/acsenergylett.0c01904

Hötzer, J., Reiter, A., Hierl, H., Steinmetz, P., Selzer, M., & Nestler, B. (2018). The parallel multi-physics phase-field framework Pace3D. *Journal of Computational Science*. 10.1016/j.jocs.2018.02.011

Mills, R. T., Adams, M. F., Balay, S., Brown, J., & Dener, A. (2021). Toward performance-portable PETSc for GPU-based exascale systems. *Parallel Computing*, *108*, 102831. https://doi.org/10.1016/j.parco.2021.102831

Nestler, B., Garcke, H., & Stinner, B. (2005). Multicomponent alloy solidification: Phase-field modeling and simulations. *Physical Review E*. https://doi.org/10.1016/j.commatsci.2022.111995

Schwen, D., Aagesen, L. K., Peterson, J. W., & Tonks, M. R. (2017). Rapid multiphase-field model development using a modular free energy based approach with automatic differentiation in MOOSE/MARMOT. *Computational Materials Science*, *132*, 36–45. https://doi.org/10.1016/j.commatsci.2017.02.017

Tegeler, M., Shchyglo, O., Kamachali, R. D., Monas, A., Steinbach, I., & Sutmann, G. (2017). Parallel multiphase field simulations with OpenPhase. *Computer Physics Communications*, *215*, 173–187. https://doi.org/10.1016/j.cpc.2017.01.023