3/27/2023

Daniel Schwen
Computational Methods
Development Group Lead

Wen Jiang
Som Dhulipala

Parallel sampling for computing rare event probabilities, inverse Bayesian inference and, uncertainty quantification with MOOSE

IPAM, Exascale Workshop

Idaho National Laboratory

Multiphysics Object-Oriented Simulation Environment (MOOSE)

A DoE sponsored finite element / finite volume based modeling framework

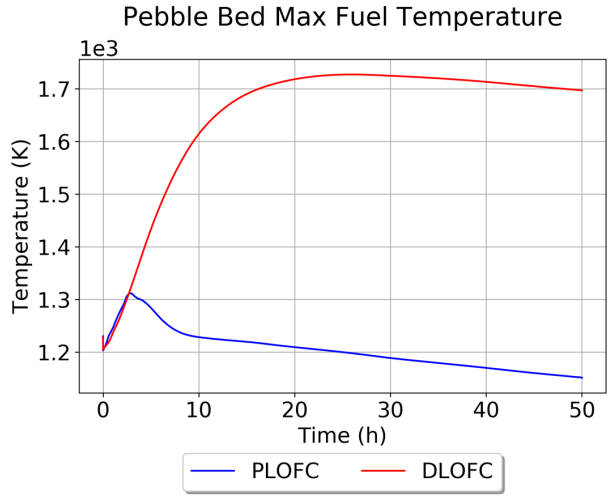
Idaho National Laboratory Position Nationally

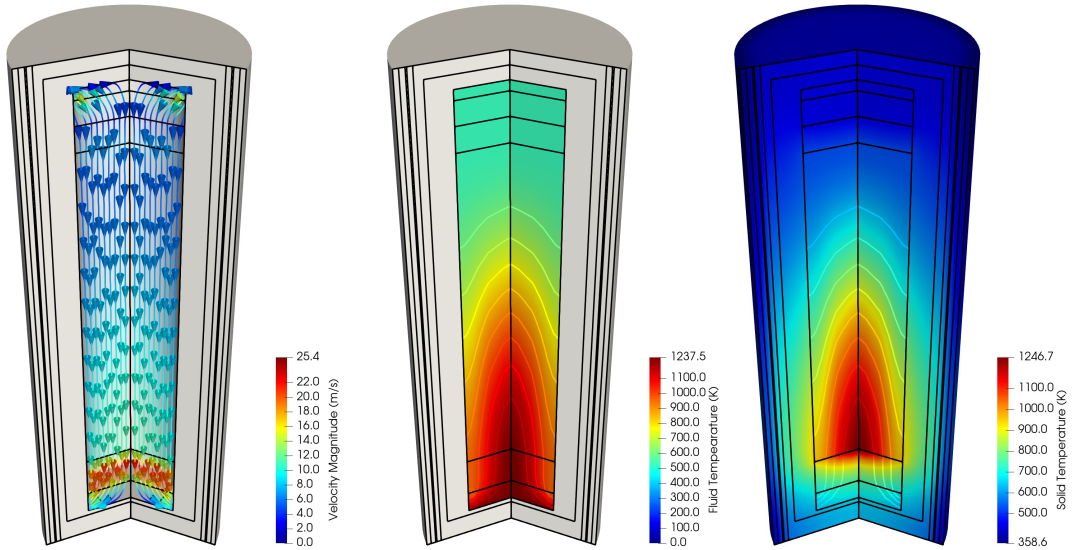
- One of 9 large DOE multi-program Labs
- DOE's Lead Lab for Nuclear Energy
- ~ 5,700 employees
- ~ 900 mi^2

INL is advancing clean, safe, and secure energy for the future

Reactor Modeling requires Multiphysics

- Reactors are inherently nonlinear, multiscale, multiphysics problems
  - Neutron transport, heat conduction, solid-mechanics, fluid flow, material degradation, chemistry, corrosion, etc.
- Predictive reactor simulation requires multiphysics (always has)
  - Doppler broadening, moderator density, etc.
- Licensing of advanced reactors will heavily rely on multiphysics mod/sim
  - Uncertainty analysis, accident scenarios, refueling, material selection, etc.

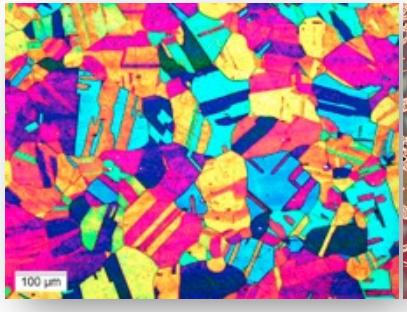




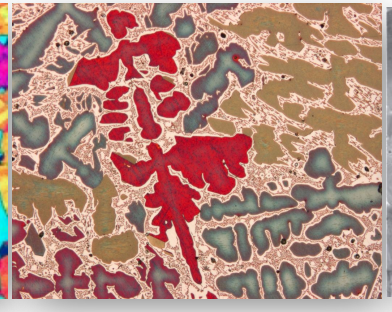
Pebble-bed PLOFC/DLOFC calculations using Pronghorn, Griffin, MOOSE-THM, MOOSE-HC
Credit: Sebastian Schunert, Guillaume Guidicelli, Alexander Lindsay, Paolo Balestra

Materials Microstructure and Properties

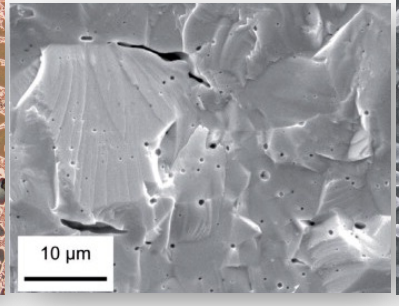
- A key objective of materials science is to understand the impact of microstructure on macroscale material behavior.



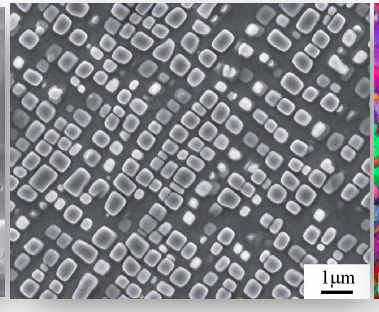
Annealed Stainless Steel



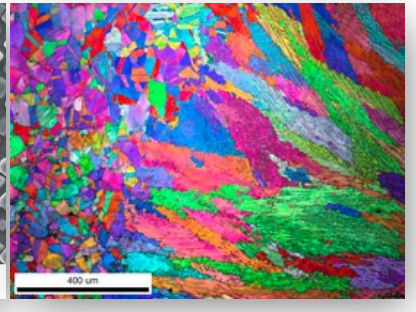
Cast Bronze



Sintered UO2

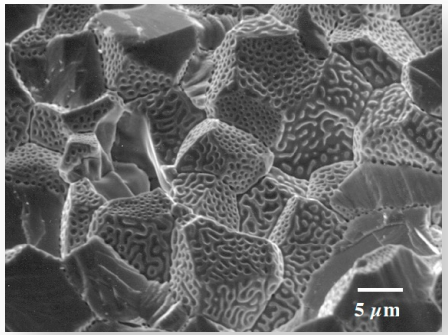


Co-Al-W Superalloy

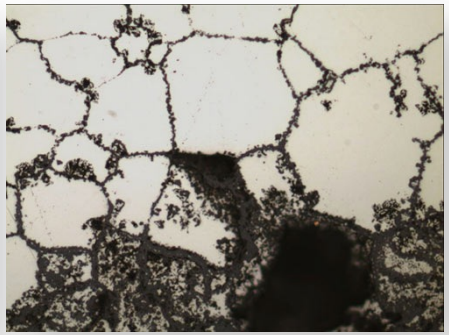


Friction stir welded stainless steel

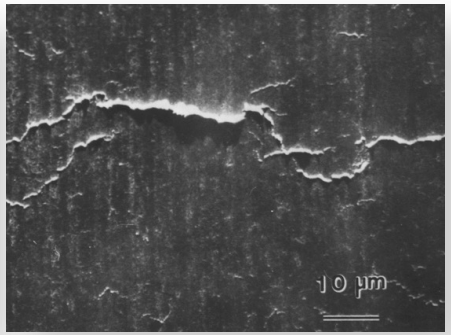
- An essential part of that is predicting the impact of microstructure evolution.



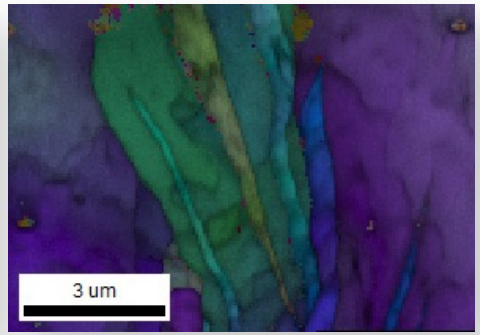
Irradiated UO_2 fuel



Corrosion in stainless steel



Micro-cracking in steel



Hydride in Zircaloy

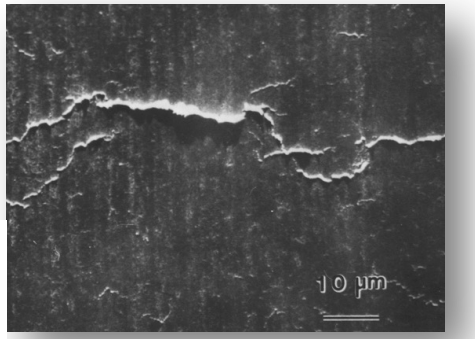
Material Behavior is Multiphysics

- Material behavior is influenced by many different physics, for example:

Mechanics

- Dislocations
- Cracking
- Stress-driven Diffusion

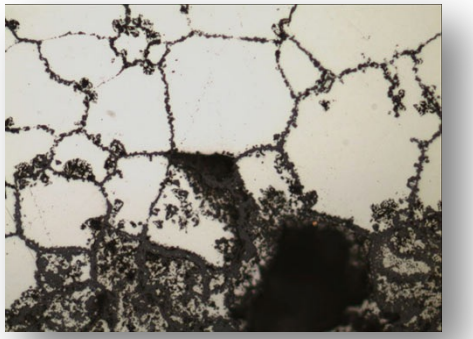
$$\nabla \cdot (\boldsymbol{\sigma} + \boldsymbol{\sigma}_0) + \mathbf{b} = \mathbf{0} \text{ in } \Omega$$
$$\mathbf{u} = \mathbf{g} \text{ in } \Gamma_{\mathbf{g}}$$
$$\boldsymbol{\sigma} \cdot \mathbf{n} = \boldsymbol{\iota} \text{ in } \Gamma_{\boldsymbol{\iota}}$$



Chemistry

- Corrosion
- Oxidation
- Reactive transport

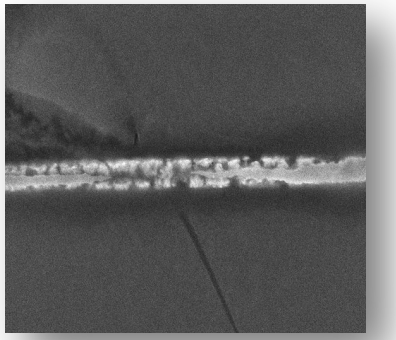
$$\frac{\partial u}{\partial t} = D_u \nabla^2 u - uv^2 + F(1-u),$$
$$\frac{\partial v}{\partial t} = D_v \nabla^2 v + uv^2 - (F+k)v.$$



Electricity/Magnetism

- Electromigration
- Ferroelectricity
- Ferromagnetism

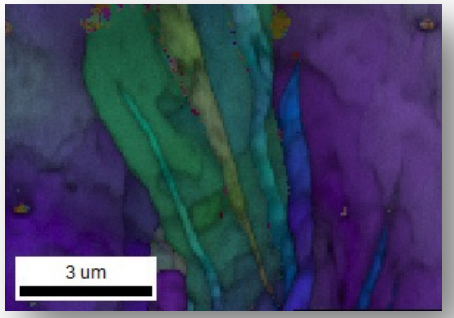
$$\nabla \cdot \mathbf{D} = \rho$$
$$\nabla \cdot \mathbf{B} = 0$$
$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$
$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}$$



Heat Conduction

- Species transport
- Melting
- Precipitation

$$\frac{\partial U}{\partial t} = \nabla^2 U$$



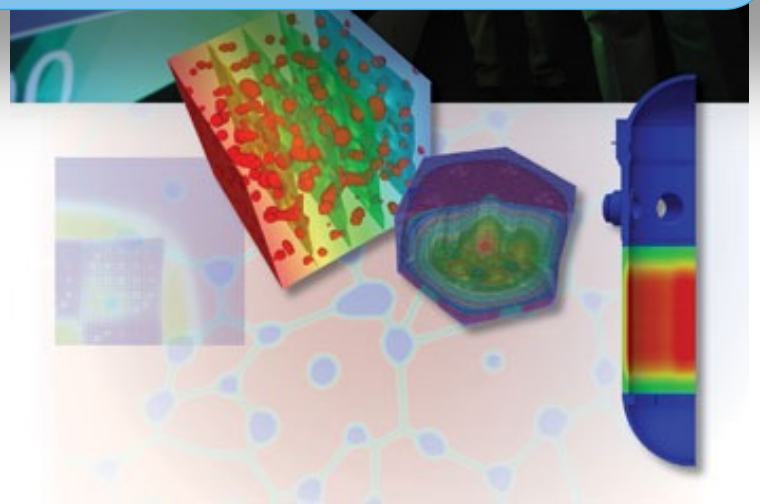
Multiphysics Object Oriented Simulation Environment

- MOOSE is a finite element, multiphysics framework that **simplifies the development** of advanced numerical applications.
- It provides a high-level interface to **sophisticated nonlinear solvers** and **massively parallel computational capability**.
- Open Source, available at https://mooseframework.inl.gov

MOOSE

MOOSE is in use across the world to solve:

- **Phase field**
- Solid mechanics
- Heat conduction
- Neutronics
- Comp. fluid dynamics
- Stochastic modeling
- Thermal hydraulics
- Geomechanics
- Reactive transport
- Corrosion
- Crystal plasticity
- Fracture
- Porous flow
- Electromagnetics



INL Idaho National Laboratory

Argonne NATIONAL LABORATORY

Los Alamos NATIONAL LABORATORY

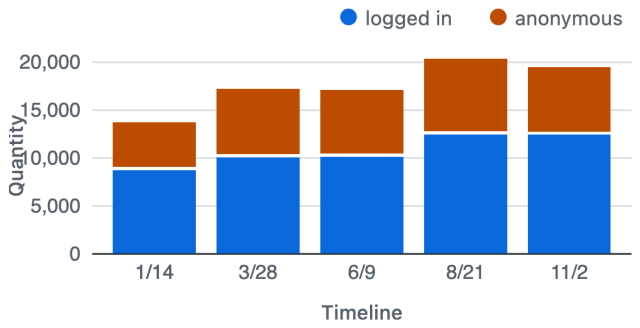
OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY

IDAHO NATIONAL LABORATORY

MOOSE Development and Community

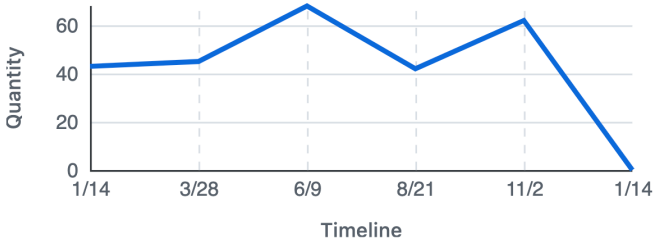
Discussions page views

Total page views to Discussions segmented by logged in vs anonymous users.

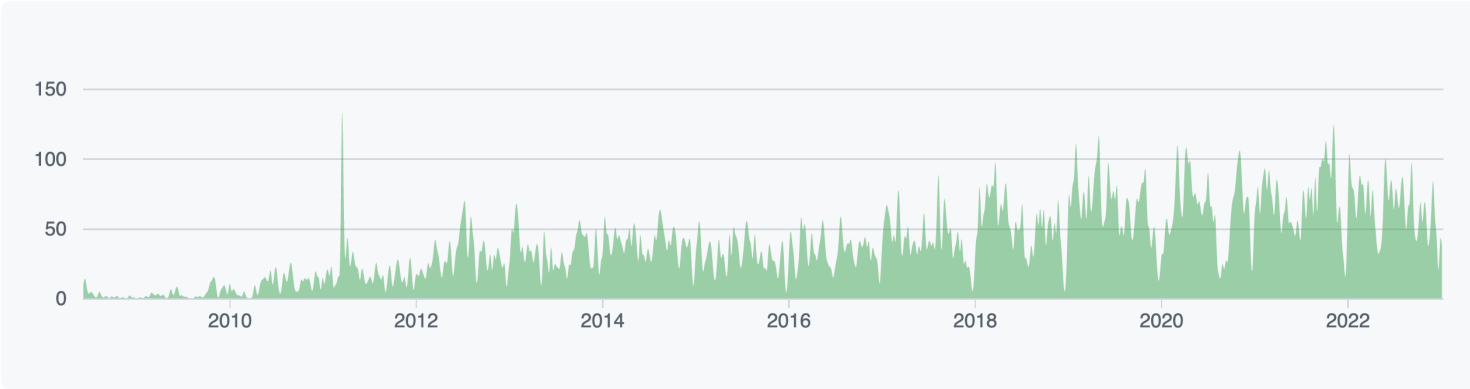


Discussions new contributors

Count of unique new users to Discussions who have reacted, upvoted, marked an answer, commented, or posted in the selected period.



- **Active development on GitHub**
  - Transparent development process
  - >850 git forks, >300 unique cloners / 2wk, 1.1k ⭐
  - >9,000 closed issues
- **Active User Community**
  - Hosted on GitHub Discussions
  - 1000s of views / 100-200 unique visitors / day
  - Hundreds of unique users contributing on Discussions each month



SoftwareX 20 (2022) 101202

Contents lists available at ScienceDirect

SoftwareX

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

Software update

2.0 - MOOSE: Enabling massively parallel multiphysics simulation

Alexander D. Lindsay [a], Derek R. Gaston [a], Cody J. Permann [a], Jason M. Miller [a], David Andrš [a], Andrew E. Slaughter [a], Fande Kong [a], Joshua Hansel [a], Robert W. Carlsen [a], Casey Icenhour [a], Logan Harbour [a], Guillaume L. Giudicelli [a,*], Roy H. Stogner [a], Peter German [a], Jacob Badger [a], Sudipta Biswas [a], Leora Chapuis [a], Christopher Green [f], Jason Hales [a], Tianchen Hu [b], Wen Jiang [a], Yeon Sang Jung [b], Christopher Matthews [c], Yinbin Miao [b], April Novak [b], John W. Peterson [d], Zachary M. Prince [a], Andrea Rovinelli [c], Sebastian Schunert [a], Daniel Schwen [a], Benjamin W. Spencer [a], Swetha Veeraraghavan [g], Antonio Recuero [a], Dewen Yushu [a], Yaqi Wang [a], Andy Wilkins [e], Christopher Wong [a]

[a] Idaho National Laboratory, Idaho Falls, ID, 83415, United States of America
[b] Argonne National Laboratory, Lemont, IL 60439, United States of America
[c] Los Alamos National Laboratory, Los Alamos, NM 87545, United States of America
[d] Akselos Inc., 2101 West Blvd., Houston, TX, 77042, United States of America
[e] CSIRO Mineral Resources, PO Box 883, Kenmore 4069, Australia
[f] CSIRO Energy, Private Bag 10, Clayton South, VIC, 3169, Australia
[g] Department of Civil Engineering, Indian Institute of Science, Bangalore, Karnataka 560012, India

ARTICLE INFO

Article history:
Received 12 August 2022
Accepted 26 August 2022

Keywords:
Multiphysics
Object-oriented
Finite-element
Framework

ABSTRACT

The last 2 years have been a period of unprecedented growth for the MOOSE community and the software itself. The number of monthly visitors to the website has grown from just over 3,000 to now averaging 5,000. In addition, over 1,800 pull requests have been merged since the beginning of 2020, and the new discussions forum has averaged 600 unique visitors per month. The previous publication has been cited over 200 times since it was published 2 years ago. This paper serves as an update on some of the key additions and changes to the code and ecosystem over the last 2 years, as well as recognizing contributions from the community.

Code metadata

Current code version	*V2.0*
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-22-00239
Permanent link to reproducible capsule	https://github.com/idaholab/moose/tree/2022-06-10-release
Legal code license	*GNU LGPL*
Code versioning system used	*git*
Software code languages, tools, and services used	*C++, MPI, OpenMP, python*
Compilation requirements, operating environments and dependencies	*Requirements: GCC/Clang C++17 compliant compiler; 16 GB memory (debug builds); 64-bit x86 + Apple Silicon support; 30 GB disk space* *Operating environments: Linux, macOS > 10.12* *Dependencies: PETSc, libMesh*
If available, link to developer documentation/manual	https://mooseframework.inl.gov/
Support email for questions	https://github.com/idaholab/moose/discussions

1. Application developer-oriented changes

The automatic differentiation [1] system has moved toward inserting derivatives based off the global degree of freedom indices. This allows construction of residuals that have highly arbitrary

DOI of original article: https://doi.org/10.1016/j.softx.2020.100430.
* Corresponding author.
E-mail address: guillaume.giudicelli@inl.gov (Guillaume L. Giudicelli).

https://doi.org/10.1016/j.softx.2022.101202
2352-7110/

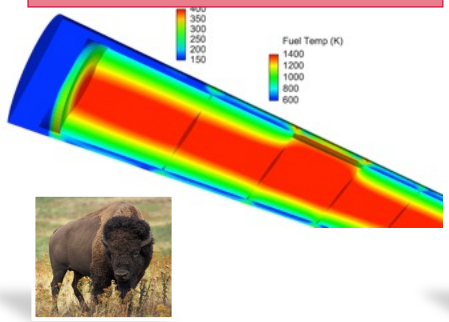
MOOSE Testing and Continuous Integration

- **Continuous integration and testing**
  - Public: https://civet.inl.gov
  - Continuous integration with ~10,000 tests in framework and modules
  - Tested in various…
    - …parallelization schemes
    - …mesh modes, AD modes
    - …compilers, operating systems, CPU architectures
    - …documentation and code coverage
  - Test on HPC
    - pull requests (before code is merged)
    - master merge (production code) (Valgrind, parallel sweeps)
  - Test >50 internal and **external** apps
  - Going on 3k cores for CI (Linux/mac,Arm/x64)

Yes, we can test YOUR app!

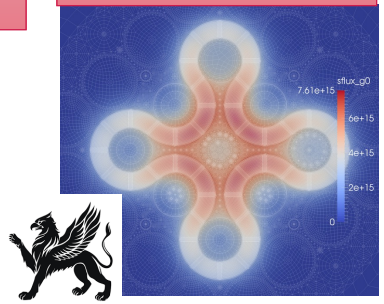
Some MOOSE-Based Applications

Bison
Nuclear Fuel Performance

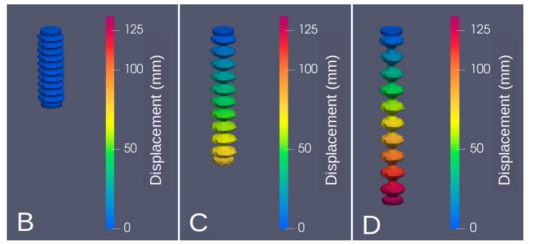


Grizzly
Structural Mechanics for Component Aging

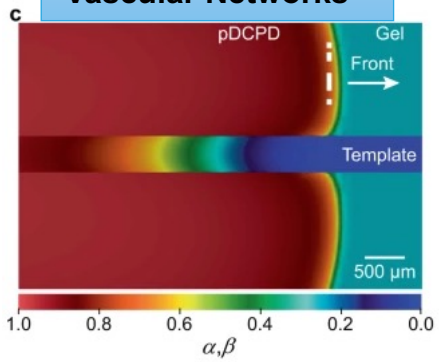
Griffin
Radiation Transport



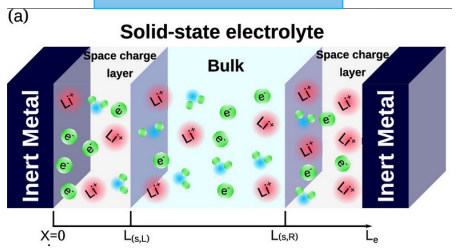
Soft Robotics [5]



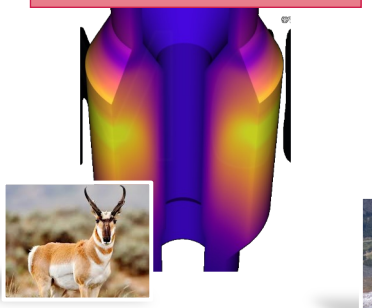
Bioinspired Vascular Networks [1]



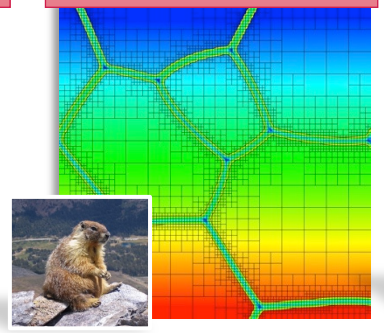
Solid-state Electrolytes [3]



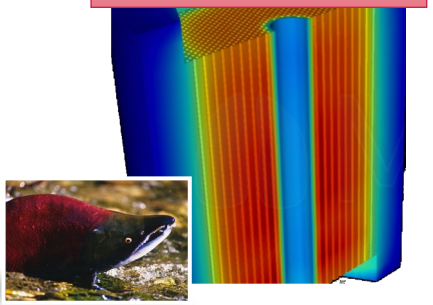
Pronghorn
Medium-fidelity CFD



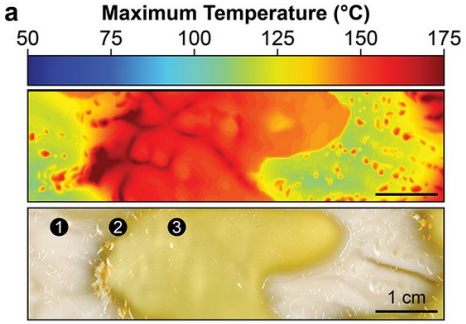
Marmot
Mesoscale Materials



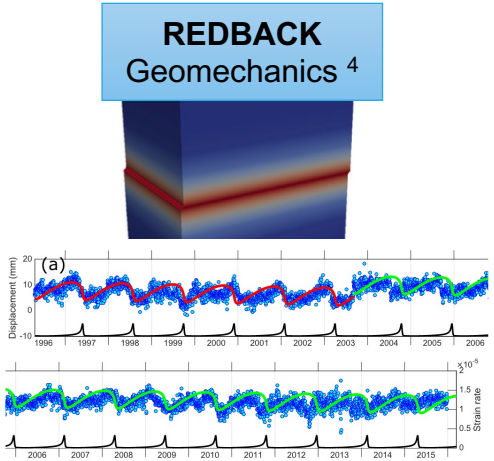
Sockeye
Heat pipe Simulation



Frontal Polymerization [2]



REDBACK
Geomechanics [4]

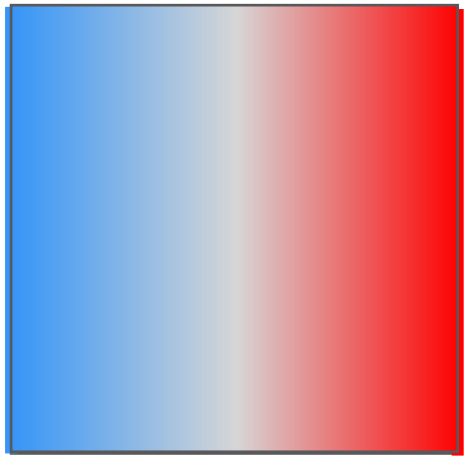


1. Garg, Mayank, et al. "Rapid synchronized fabrication of vascularized thermosets and composites." Nature communications 12.1 (2021): 1-9.
2. Lloyd, Evan M., et al. "Spontaneous Patterning during Frontal Polymerization." ACS central science 7.4 (2021): 603-612.
3. Liu, Yao, et al. "Impedance Modeling of Solid-State Electrolytes: Influence of the Contacted Space Charge Layer." ACS Applied Materials & Interfaces 13.4 (2021): 5895-5906.
4. Veveakis, E., S. Alevizos, and T. Poulet. "Episodic tremor and slip (ETS) as a chaotic multiphysics spring." Physics of the Earth and Planetary Interiors 264 (2017): 20-34.
5. K. Wandke, Y Z, "MOOSE-Based Finite Element Hyperelastic Modeling for Soft Robot Simulations." IEEE Access 9 (2021), 139627 - 139635

The MOOSE “Framework”

- MOOSE is not an “application”, but a “framework” that provides
  - C++ infrastructure to write your own…
    - physics (PDE weak forms)
    - Initial and boundary conditions
    - On-the-fly postprocessing (e.g. max/min concentration, precipitate counting, line scans)
  - An input file parser to combine the MOOSE C++ objects into a simulation at runtime, and add/configure
    - Variables
    - Solver and solver parameters
    - Outputs

```
[Mesh]
  type = GeneratedMesh
  dim = 2
  nx = 10
  ny = 10
[]

[Variables]
  [u]
  []
[]

[Kernels]
  [diff]
    type = Diffusion
    variable = u
  []
[]

[BCs]
  [left]
    type = DirichletBC
    variable = u
    boundary = left
    value = 0
  []
  [right]
    type = DirichletBC
    variable = u
    boundary = right
    value = 1
  []
[]

[Executioner]
  type = Steady
  solve_type = 'PJFNK'
  petsc_options_iname = '-pc_type -pc_hypre_type'
  petsc_options_value = 'hypre boomeramg'
[]

[Outputs]
  exodus = true
[]
```



Writing MOOSE Code

Implementing your physics in MOOSE

1. Derive your PDE term weak form
2. Create small boilerplate C++ .C and .h file
3. Type in weak form

$$\vec{v}\nabla u - \frac{\partial u}{\partial t} = 0$$

$$(\psi, \vec{v}\nabla u) - \left(\psi, \frac{\partial u}{\partial t}\right) = 0$$

```
_test[_i][_qp] * _v[_qp] * _grad_u[_qp];
```

```
registerMooseObject("MyAwesomeApp", ADAdvection);

InputParameters
ADAdvection::validParams()
{
  InputParameters params = ADKernel::validParams();
  params.addRequiredParam<MaterialPropertyName>("velocity_vector",
    "Advection velocity vector");
  return params;
}

ADAdvection::ADAdvection(const InputParameters & parameters)
  : ADKernel(parameters),
    _v(getADMaterialProperty<RealVectorValue>("velocity_vector"))
{
}

ADReal
ADAdvection ::computeQpResidual()
{
  return _test[_i][_qp] * _v[_qp] * _grad_u[_qp];
}
```

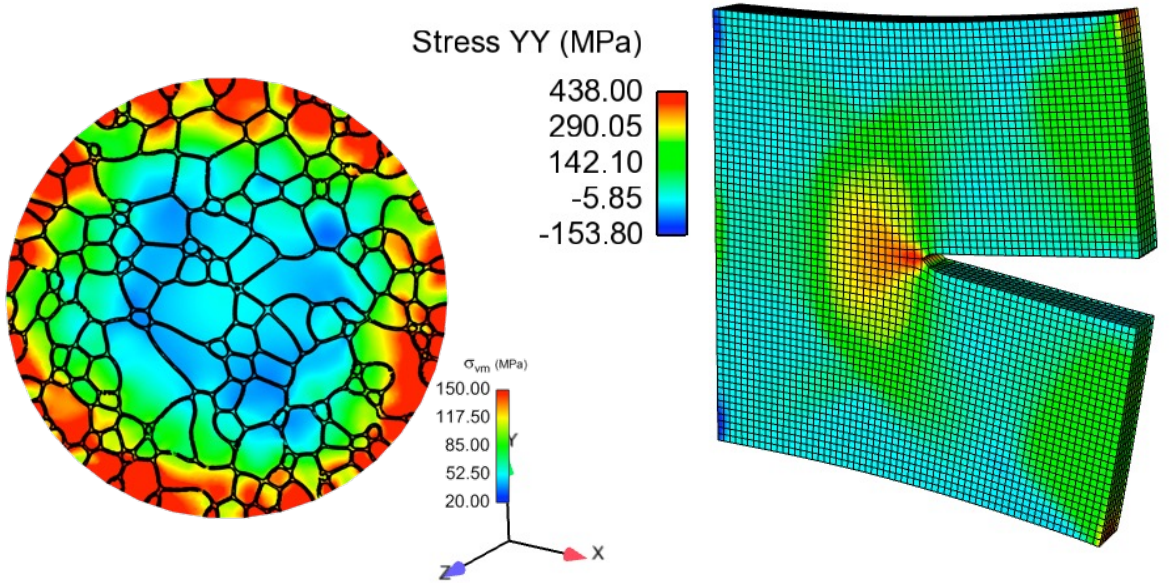
```
[Kernels]
  [convection]
    type = ADCoupledConvection
    variable = u
    velocity_vector = v
  []
  [dt]
    type = ADTimeDerivative
    variable = u
  []
[]
```

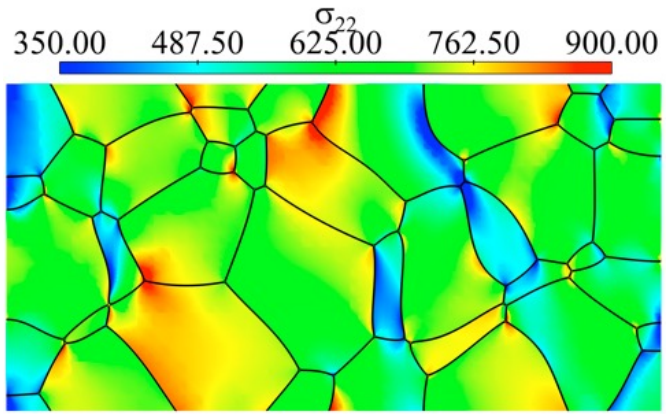
Mesoscale Materials Modeling

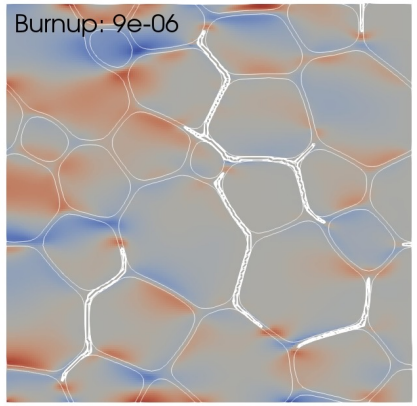
Brief overview over MOOSE mesoscale modeling capabilities

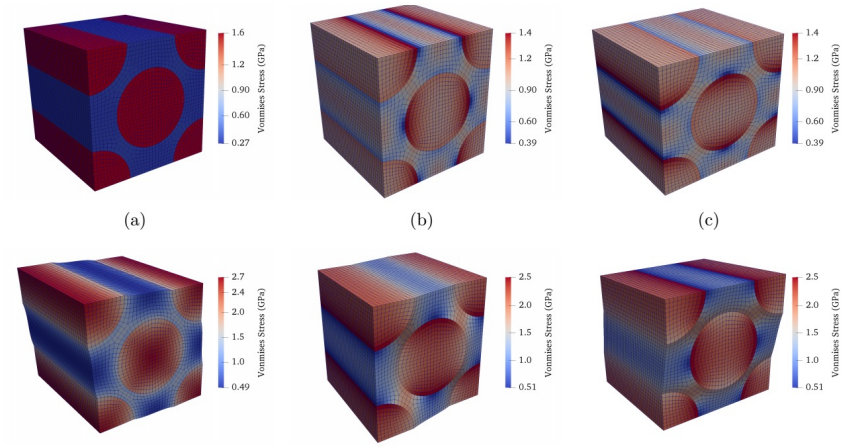
MOOSE Tensor Mechanics Module

- Provides the tools necessary for modeling mechanical deformation and stress at the mesoscale.
- Strain (small, finite, incremental)
- Eigenstrains / Eigenstresses
- Elastic stress, Modular inelastic stress system
  - Creep
  - Plasticity (J2, Crystal Plasticity)
  - Fracture
- Strain periodicity (representative volume elements)
- Fully couplable to phase field



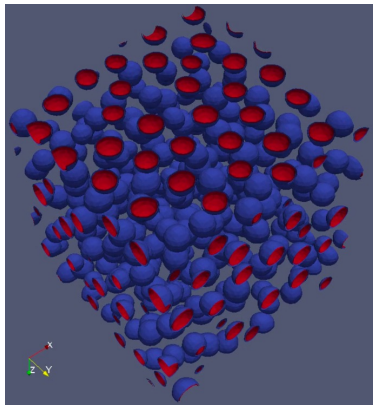


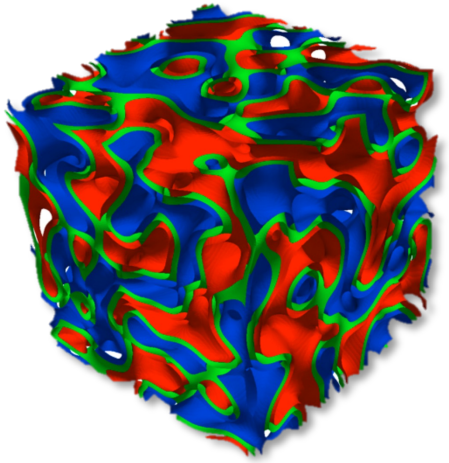


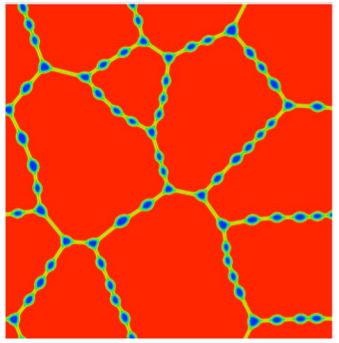


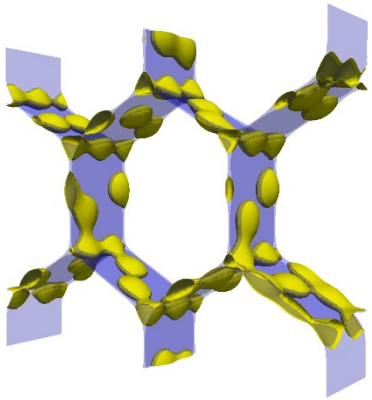
MOOSE Phase Field Module

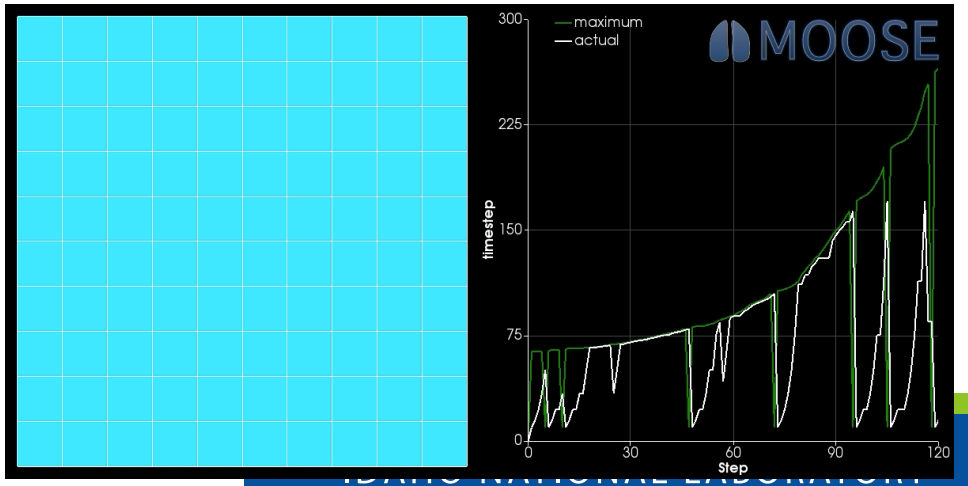
- Provides all the tools necessary to develop a massively parallel phase field code using FEM.
- Base classes for solving Cahn Hilliard equations
  - Direct solution / Split solution
  - Grand Potential, KKS
- Base classes for Allen-Cahn equations
- Grain growth model
- Grain remapping algorithm for efficient models
- Initial conditions
- Postprocessors for characterizing microstructure
- Nucleation



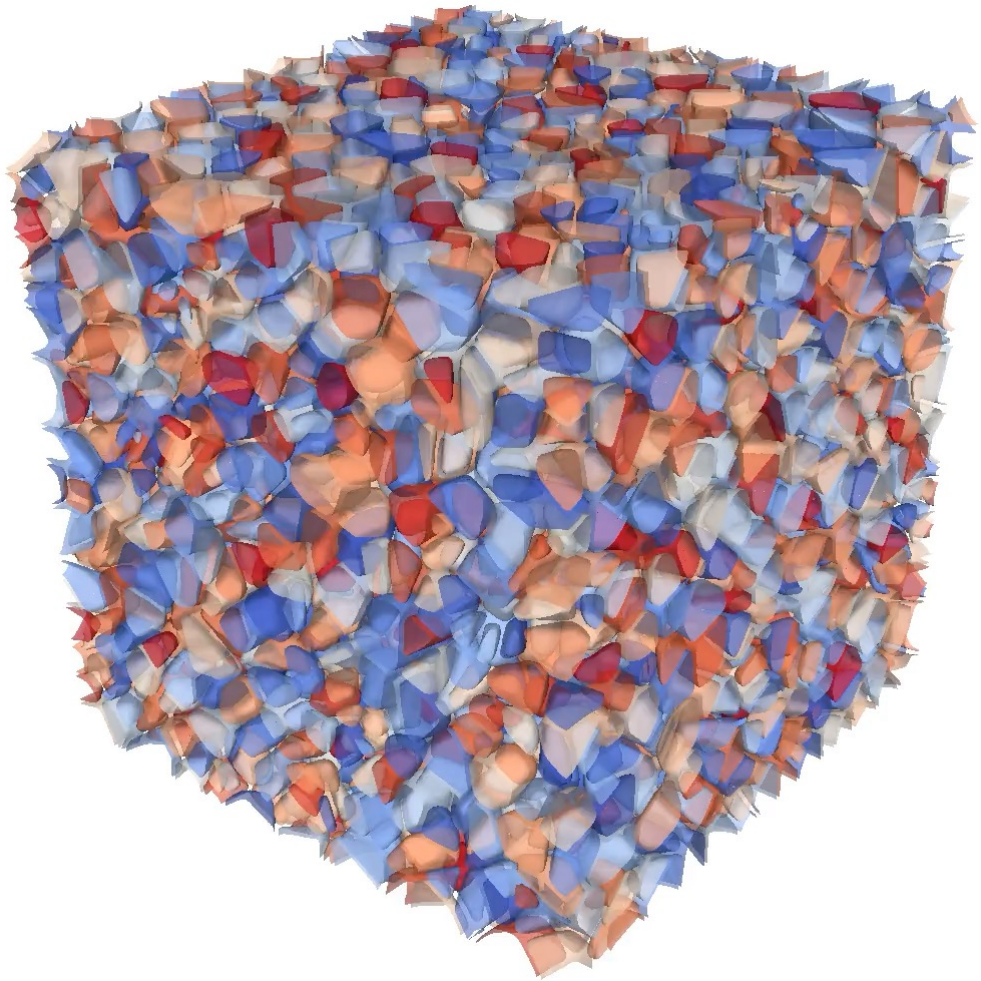




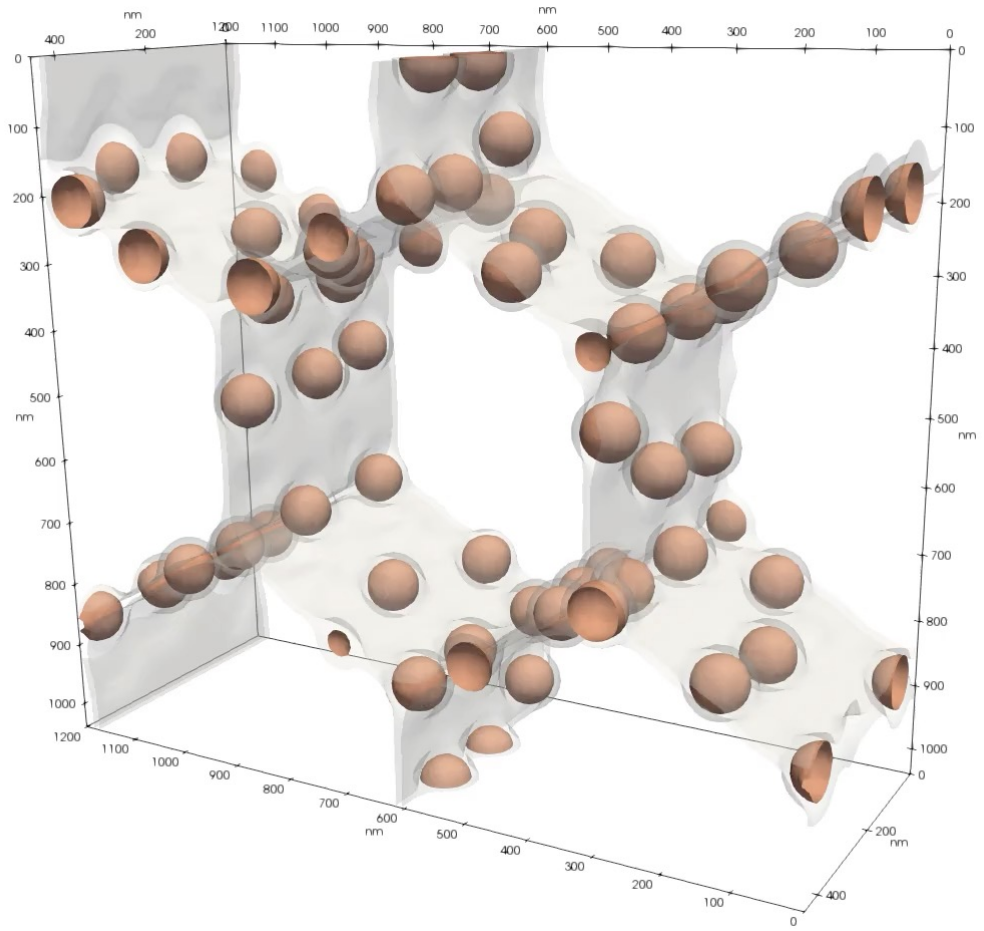




Phase Field Examples

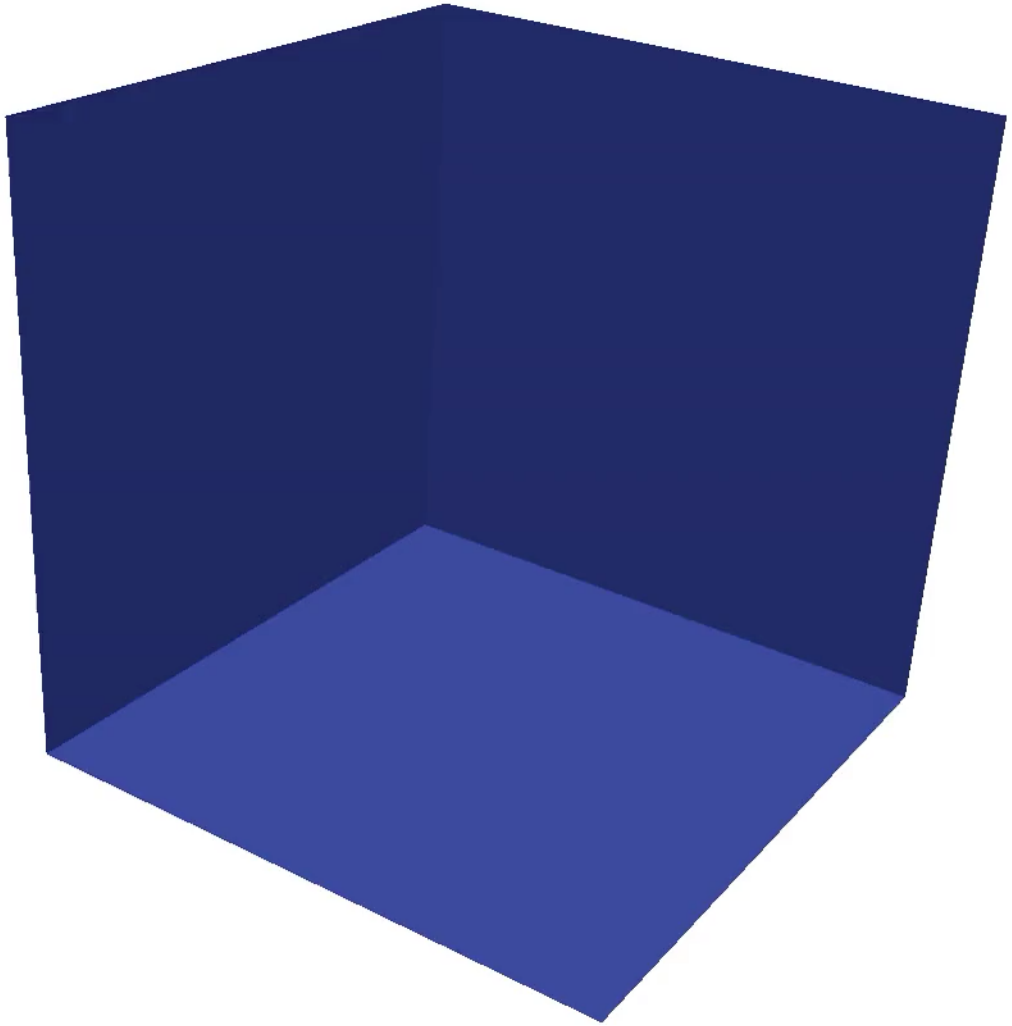


Grain growth model (6000 grains) with OP remapping
150 million DOFs on ~580 CPU cores

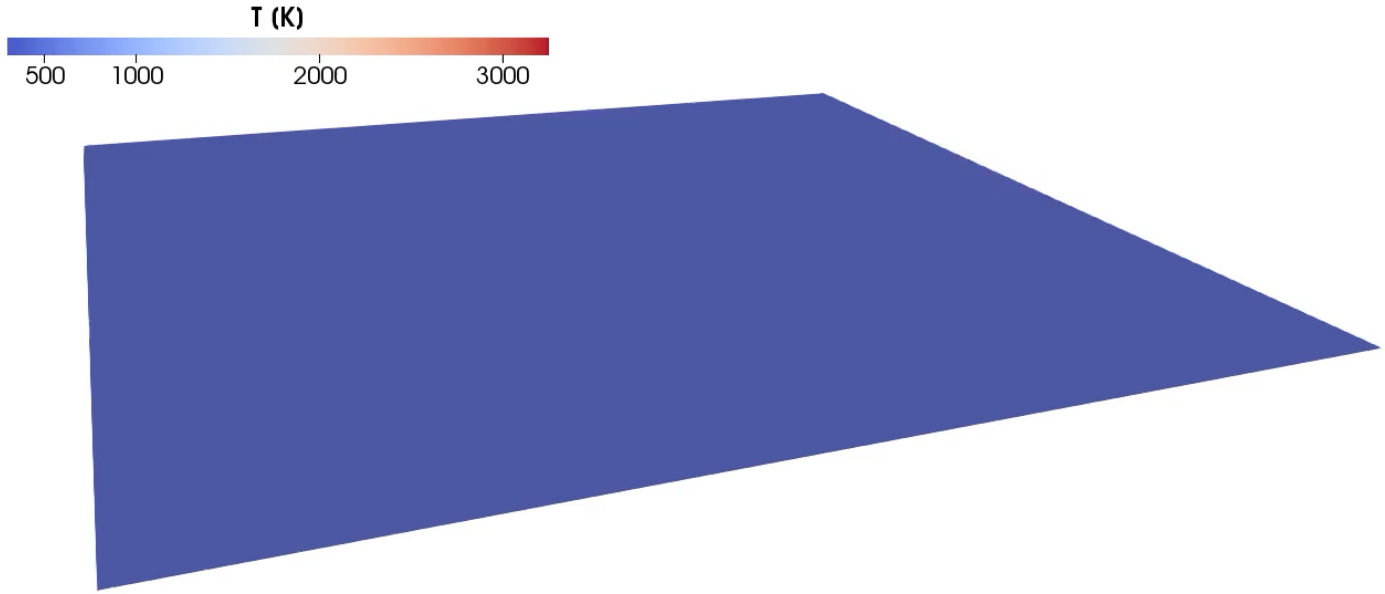


Grain boundaries decorated with growing gas bubbles. (Larry Aagesen)

Phase Field / Laser Melting



Void lattice formation due to anisotropic interstitial diffusion. (D. Schwen)

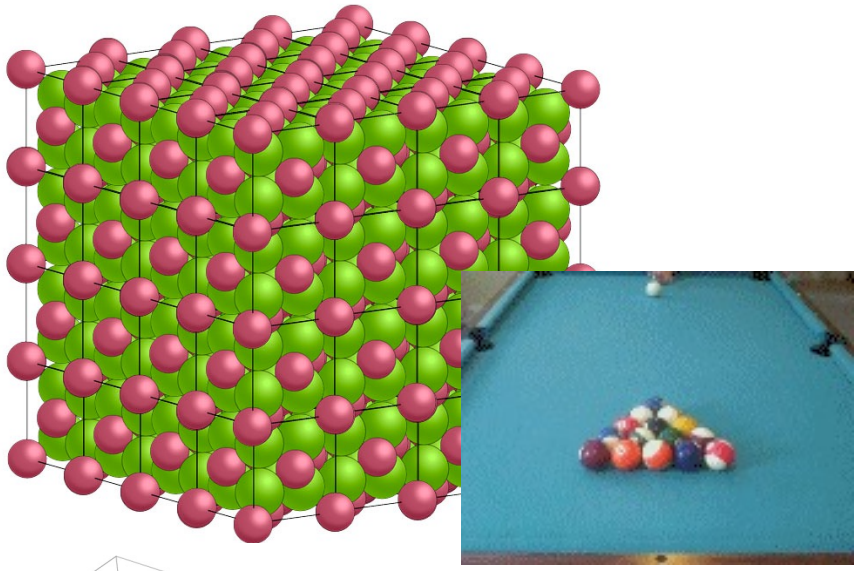


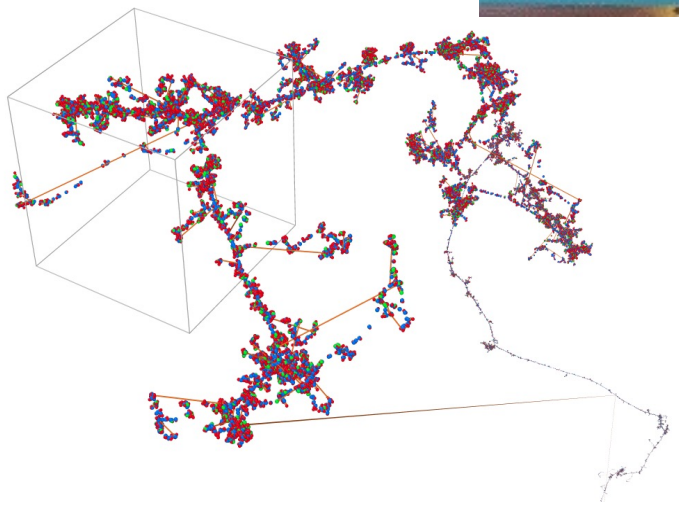
Local laser melting – heat transport, ablation, and Navier-Stokes using ALE. (Alexander Lindsey)

Simulating radiation and microstructure

atomic scale

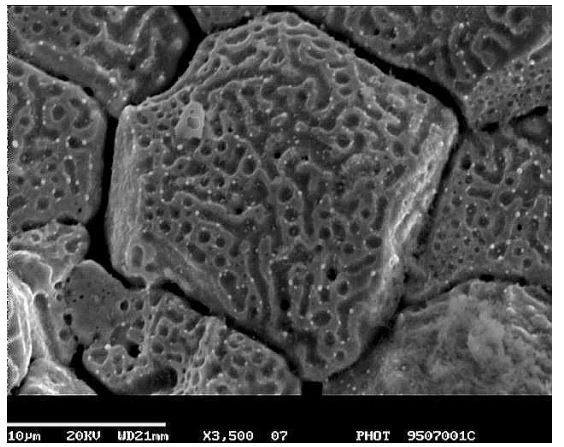
1 Å

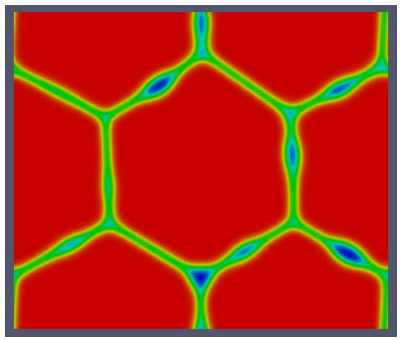


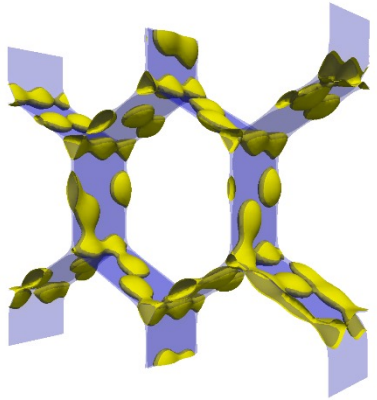


mesoscale

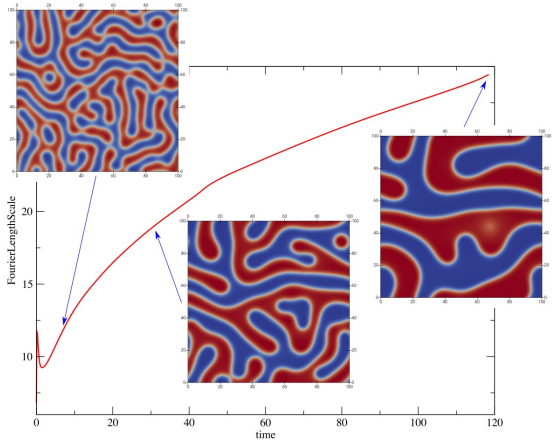
10 nm–
1 µm





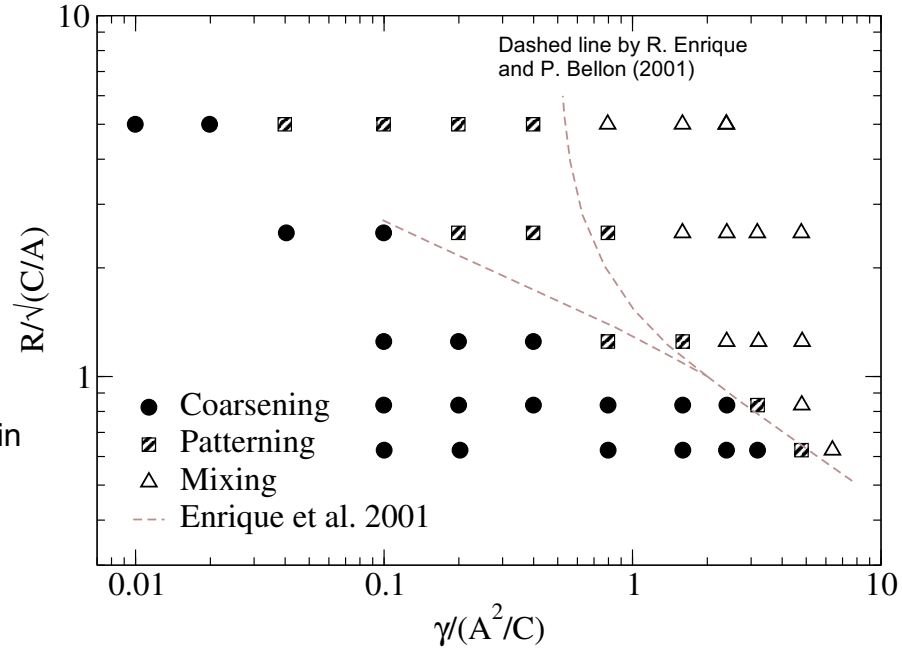


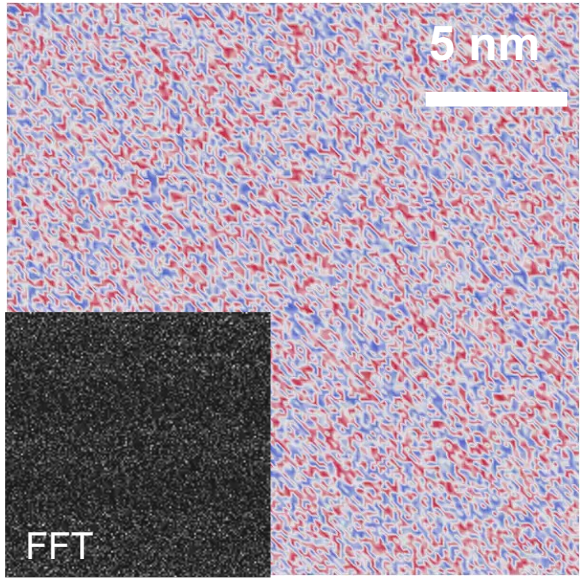
Nanoscale ballistic mixing



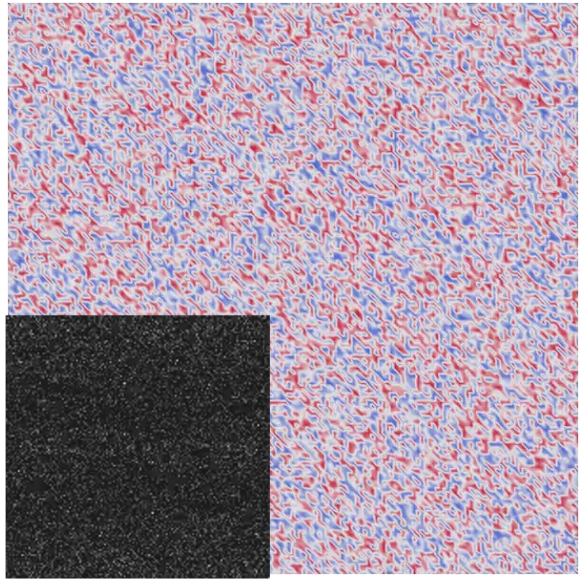
Coupling to FFTW3 to get Fourier space data, such as power spectrum and characteristic length scale.

Patterning, mixing, coarsening in an immiscible alloy under irradiation. Axes are displacement rate and distance

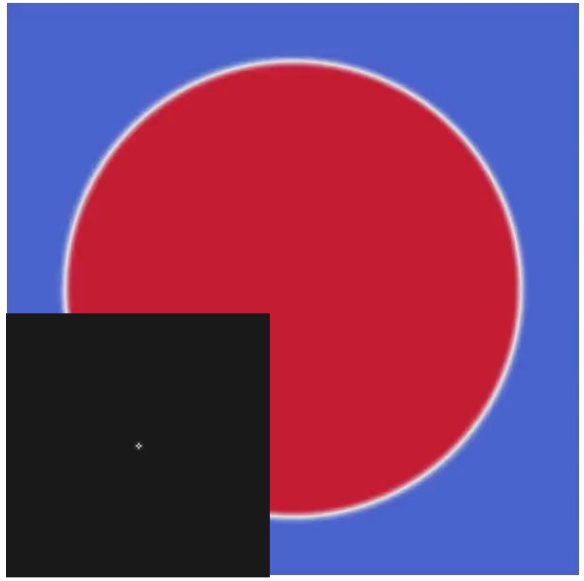




No irradiation



Irradiation



- Phase separating immiscible alloy (CuAg)
- 100,000s of recoil cascades run (50 keV Xenon ions)
- Steady-state length scale
- Applications: fission-gas bubbles, FeCrAl precipitates

Thermodynamic Database Coupling

CALPHAD approach
⇢ Free energy description of countless systems!
⇢ *de facto* standard TDB file

$$H_a(x) = x(1-x)\left[\sum_{i=0}^{5} a_i (1-2x)^i\right] + a_6 x + a_7(1-x) \quad (1)$$

$$H_{\xi\in\{b,c,d\}}(c) = \sum_{i=0}^{3} \xi_i \cdot (1-x)^i \quad (2)$$

$$H_e(x) = -\frac{H_a(x)}{T_0} + H_c(x)T_0 + H_b(x)\log T_0 + H_d(x)\frac{T_0^2}{2} \quad (3)$$

$$G_0(x) = x(1-x)\left[\sum_{i=0}^{5} f_i (1-2x)^i\right] + f_6 x + f_7(1-x) \quad (4)$$

$$G(x,T) = G_0(x)\frac{T}{T_0} + H_a(x) - H_b(x)T\log T - H_c(x)T^2 + H_d(x)\frac{T^3}{2} + H_e(x)T + Tk_B(x\log x + (1-x)\log(1-x)) \quad (5)$$

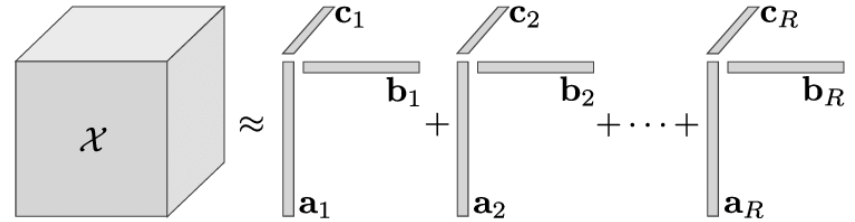
Compound Energy Formalism

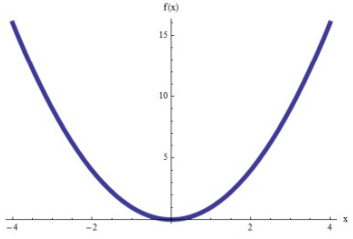
- No sublattices / internal DOFs
  - Directly export free energy expressions
- Solve local equilibrium
  - Concurrently couple thermodynamic software (OpenCalphad, pycalphad, ThermoCalc)
  - Parabolic fits
  - Tabulate free energy (and chemical potentials) in state space
    - Polyadic Tensor decomposition (Moelans, KU Leuven) ⇢ MOOSE
    - Neural Networks

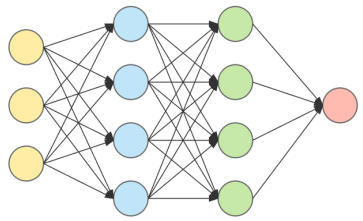
pycalphad

Thermo-Calc Software

OpenCalphad





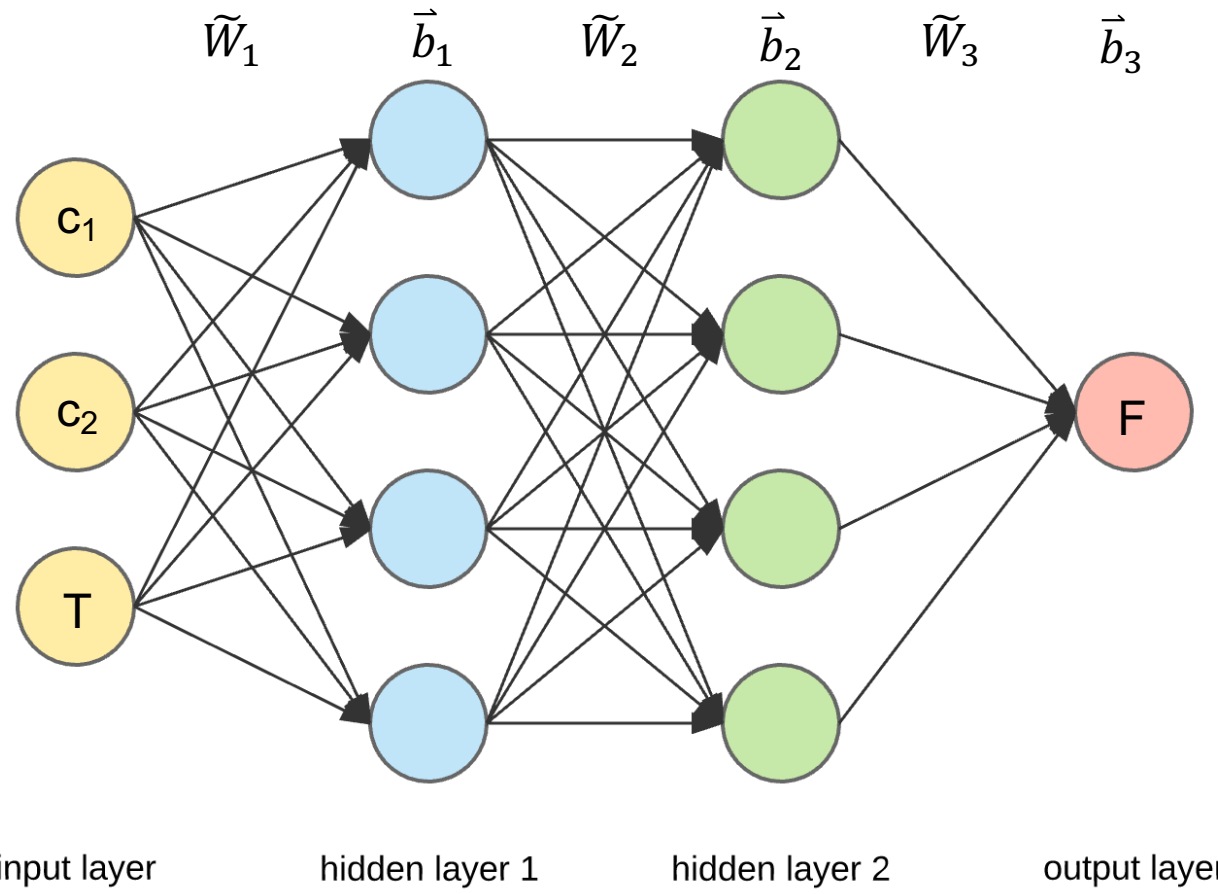


Fitting a neural net to TDB data

- Universal Approximation Theorem
  A feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of R^n
- PyTorch
  - Mini batch learning
  - Cost function includes chemical potentials dF/dc_1 etc.
  - Initial input/output weight/bias guess
    no manual normalization required
  - g: SoftSign, Sigmoid, tanh
- **W**eights & **b**iases ⇢ Text file ⇢ MOOSE
- Derivative of NN ⇢ chemical potential

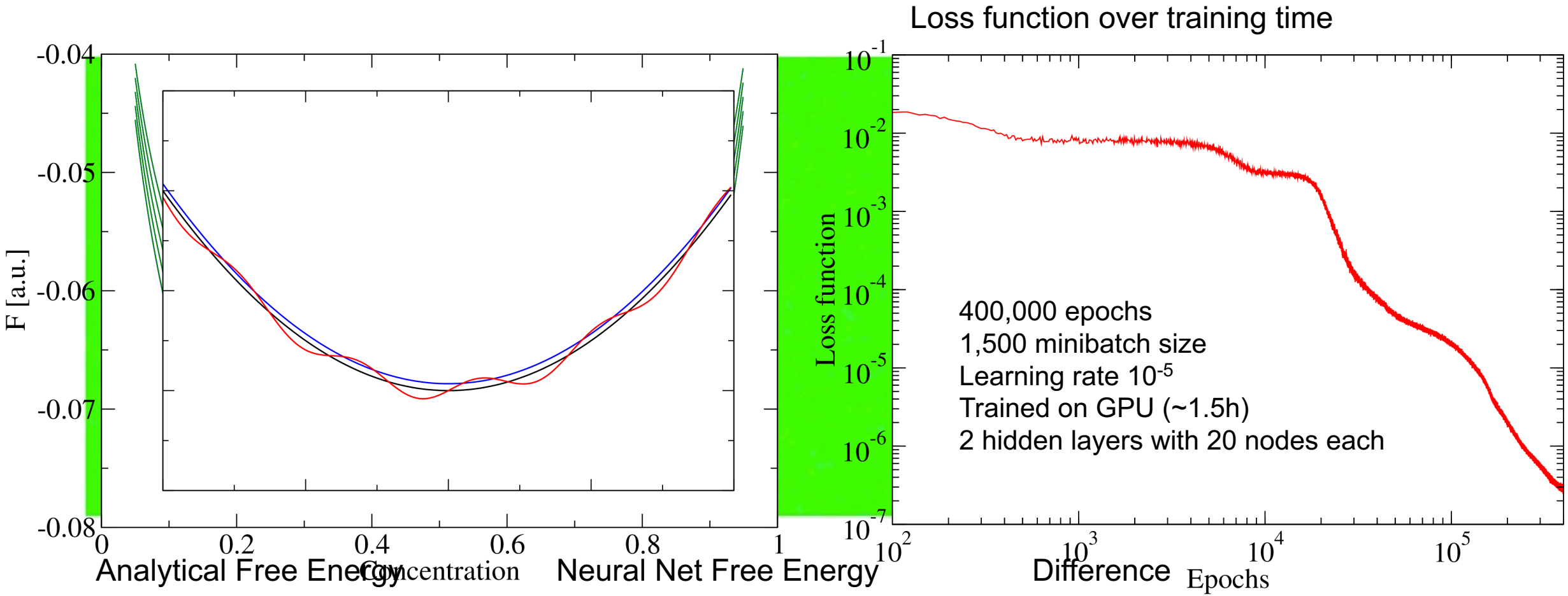
$$F = \left[\left[\begin{pmatrix} c_1 \\ c_2 \\ T \end{pmatrix} \cdot \widetilde{W}_1 + \vec{b}_1\right]_g \cdot \widetilde{W}_2 + \vec{b}_2\right]_g \cdot \widetilde{W}_3 + \vec{b}_3$$

PyTorch

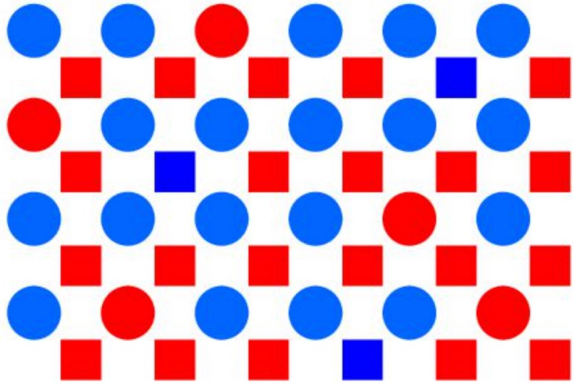


Fitting data from a regular solution free energy

$$f = c(1-c) + 10^{-3}T(c\log c + (1-c)\log(1-c))$$



Sublattice Kim-Kim-Suzuki Model



Original KKS Model

$$F = \sum_j h_j F_j$$

Free energy is weighted sum of phase F_j

$$c_i = \sum_j h_j c_{ij}$$

Physical concentration is weighted sum of phase c_{ij}

$$\frac{\partial F_j}{\partial c_{ij}} = \frac{\partial F_{j'}}{\partial c_{ij'}} = \mu_i$$

Equal chemical potential in phase j and j'

New Sublattice KKS Model

$$F = \sum_j h_j F_j$$

Free energy is weighted sum of phase F_j

$$g = -c_{ij} + \sum a_{jk} c_{ijk}$$

$$\nabla_{c_{ijk}} F = \lambda \nabla_{c_{ijk}} g$$

$$c_i = \sum_j h_j \sum_j a_{jk} c_{ijk}$$

Physical concentration is weighted sum of **sublattice** c_{ijk}

$$\frac{1}{a_{jk}} \frac{\partial F_j}{\partial c_{ijk}} = \frac{1}{a_{j'k'}} \frac{\partial F_{j'}}{\partial c_{ij'k'}} = \frac{\partial F_j}{\partial c_{ij}} = \mu_i$$

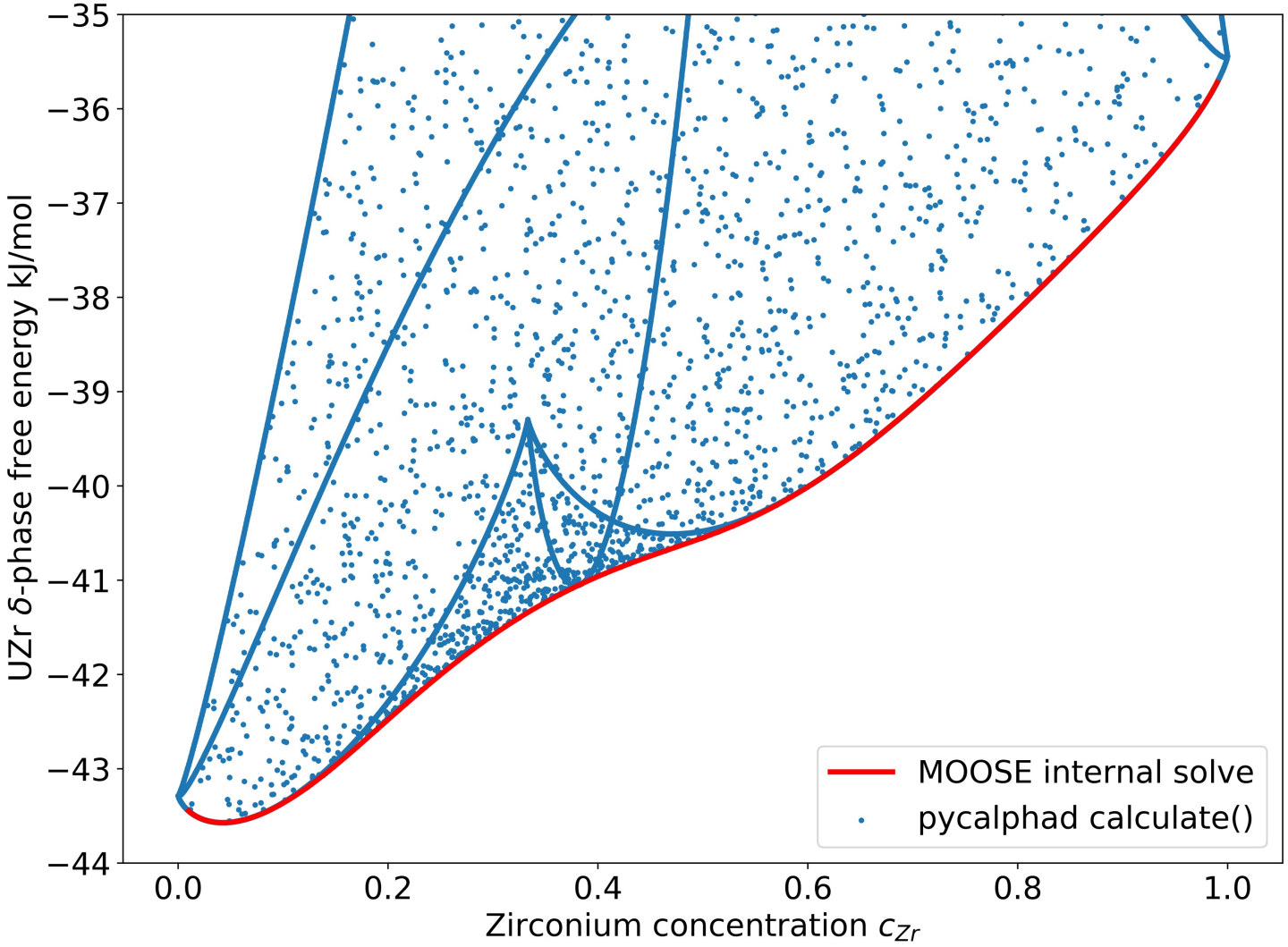
Equal chemical potential between sublattices, derived from Lagrange multiplier constrained minimization of F_j

D. Schwen, et al., A sublattice phase-field model for direct CALPHAD database coupling, Comp. Mat. Sci., 195 (2021) 110466

Internal DOF minimization – Phase free energy

- Solving PDEs for constrained minimization of δ-UZr internal DOFs
- Obtain effective phase free energy for δ-UZr
- Phase chemical potential can be derived as a function of the sublattice chemical potential

$$\frac{\partial F_j}{\partial c_{ij}} = \frac{1}{a_{jk}} \frac{\partial F_j}{\partial c_{ijk}}$$

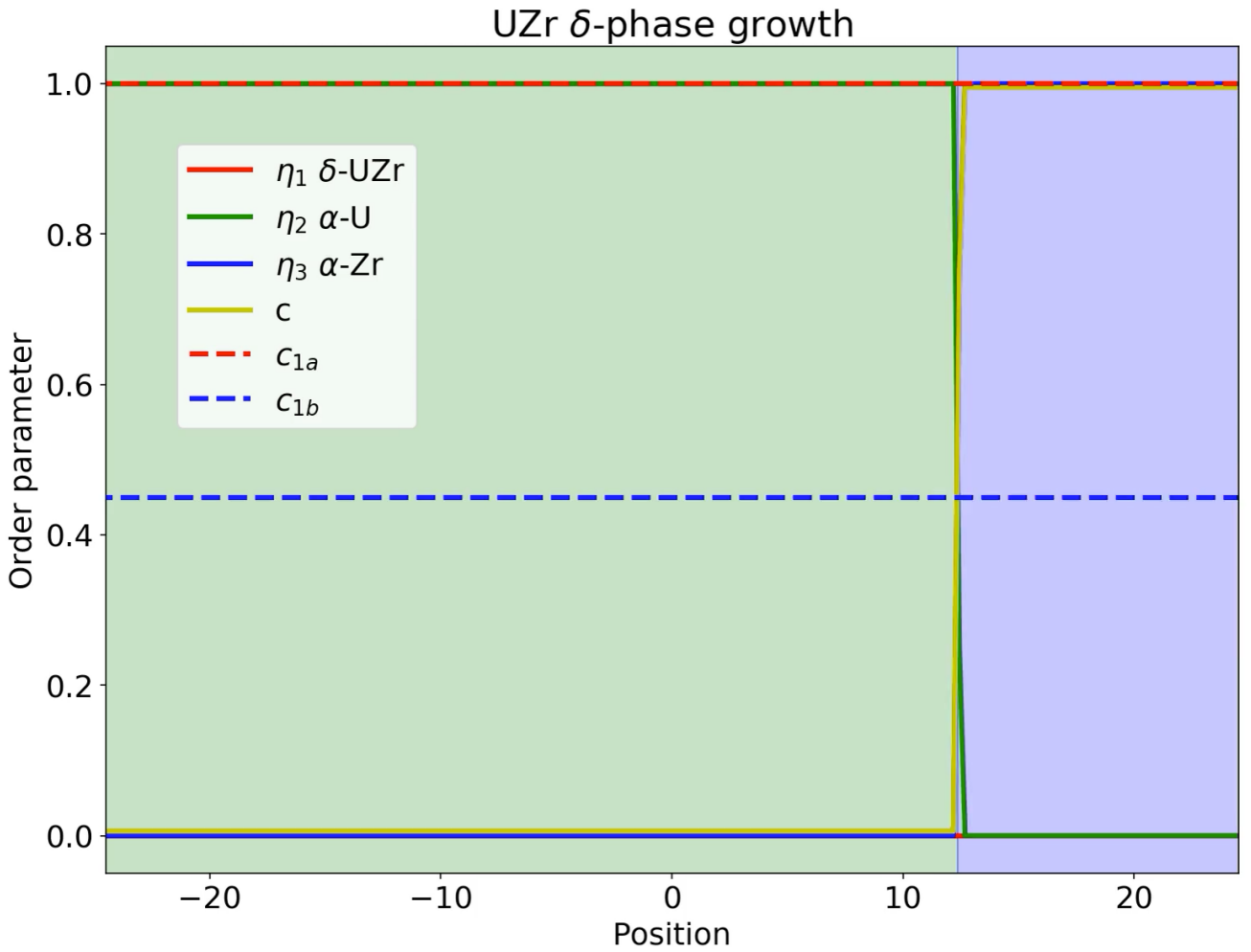


UZr phase field

- Ready to run the SLKKS phase field model
- Require multi phase formulation for appropriate switching functions h_j
- Thermodynamically consistent model for three phases implemented in MOOSE
- Formulations for n>3 phases exist Implementation WIP

- U/Zr bilayer
- Interdiffusion and formation of δ-UZr

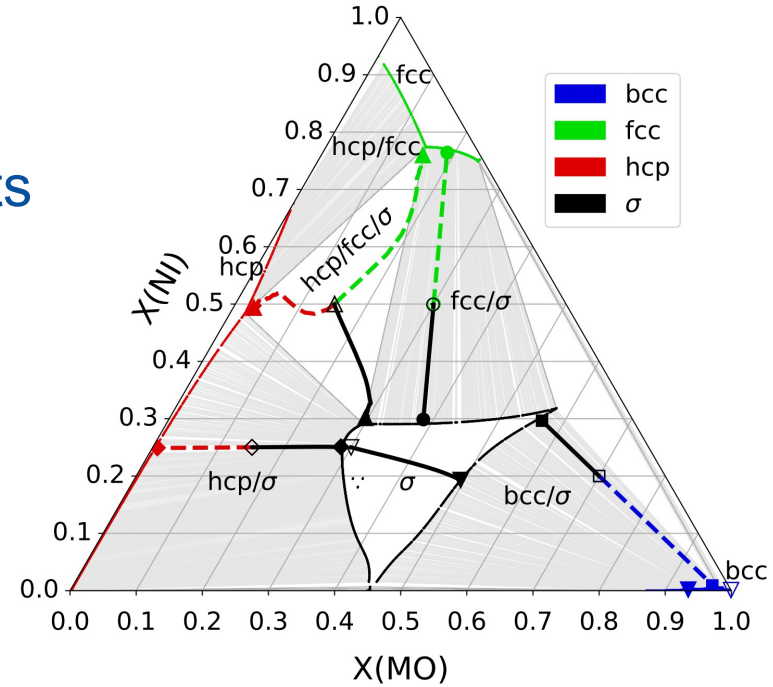
(Interfacial free energies 10mJ/m^2 and mobilities not based on physical values, all phases have the same atomic volume)

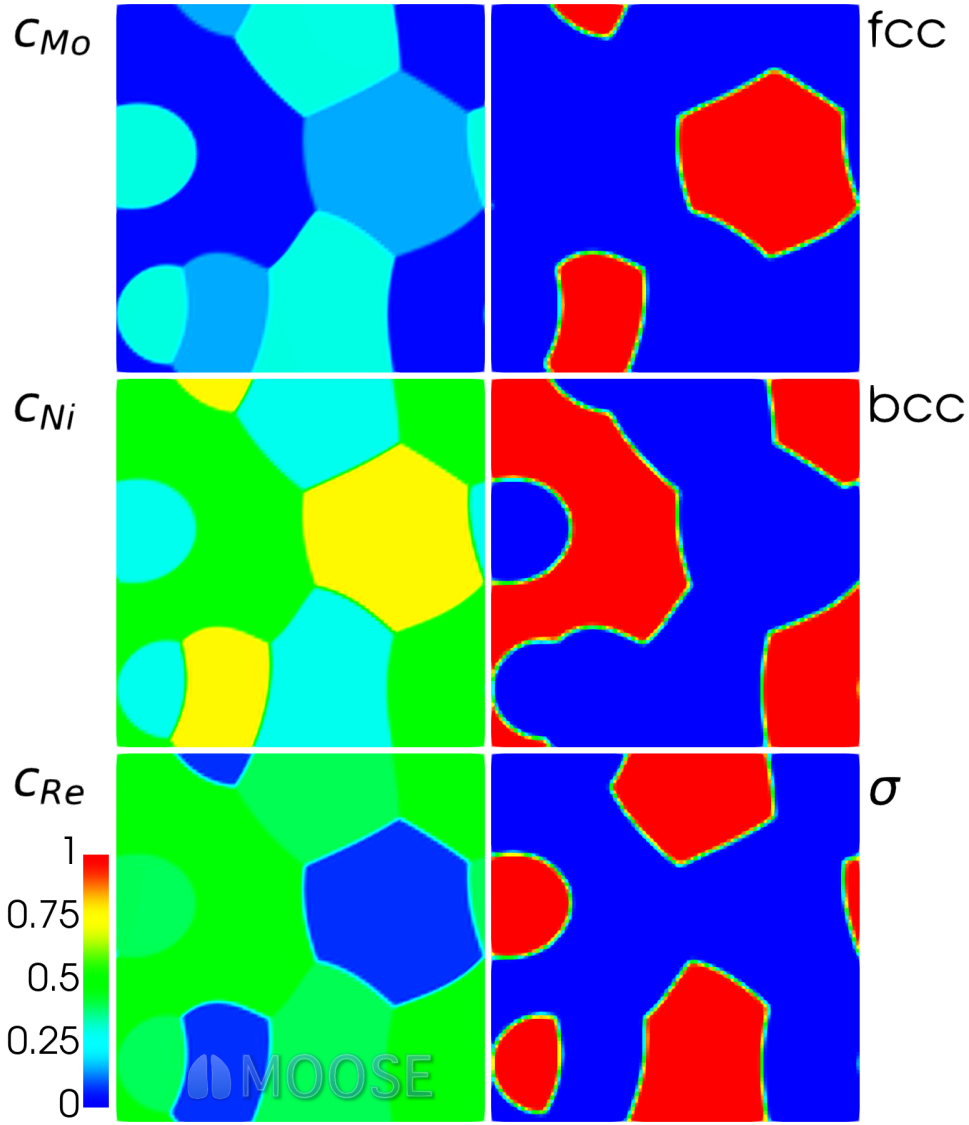


Mo-Ni-Re phase field

- Mo-Ni-Re High temperature Ni-based superalloy
  J.-C. Crivello, R.Souques, A.Breidi, N.Bourgeois, J.-M.Joubert,
  Calphad 51(2015) 233-240
- Ternary system
- Five(!) sublattices in the σ-phase
  $(Mo,Ni,Re)_2(Mo,Ni,Re)_4(Mo,Ni,Re)_8(Mo,Ni,Re)_8(Mo,Ni,Re)_8$
- Transient simulation
  Not at equilibrium
- Interfacial energy impacts bulk concentration

(Interfacial free energies 10mJ/m^2 and mobilities not based on physical values, all phases have the same atomic volume)

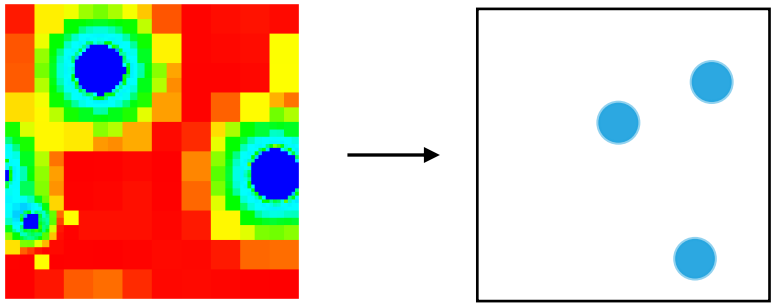




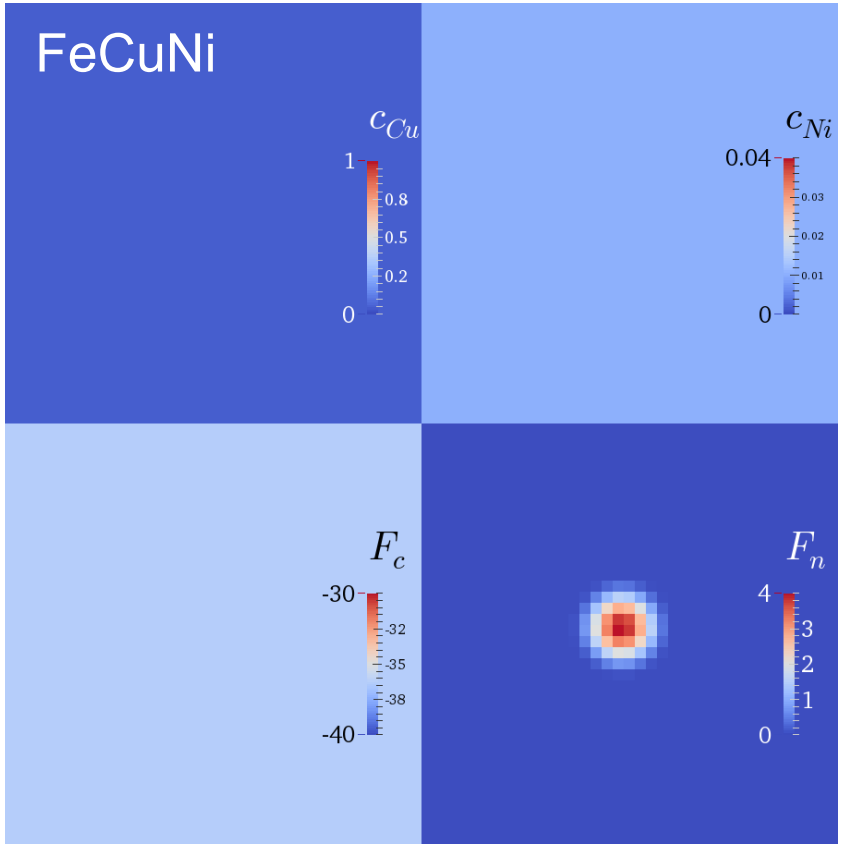
Phase separated microstructure evolved from a homogeneous initial composition of $Mo_3Ni_{10}Re_7$, which is located in the hcp, fcc, and σ-phase coexistence region.

Explicit Nucleation - Modifying free energy density

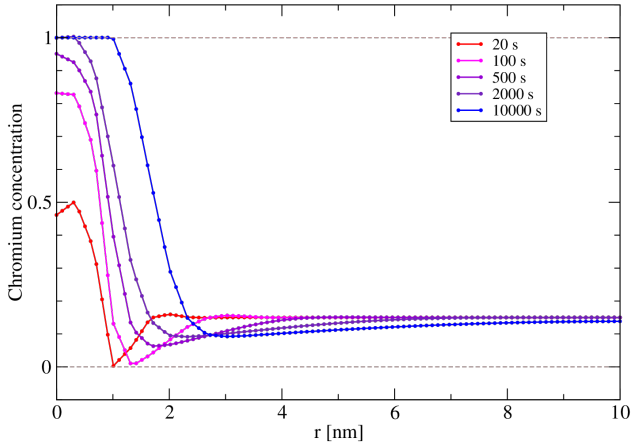
- Insert nucleation sites base on rate density

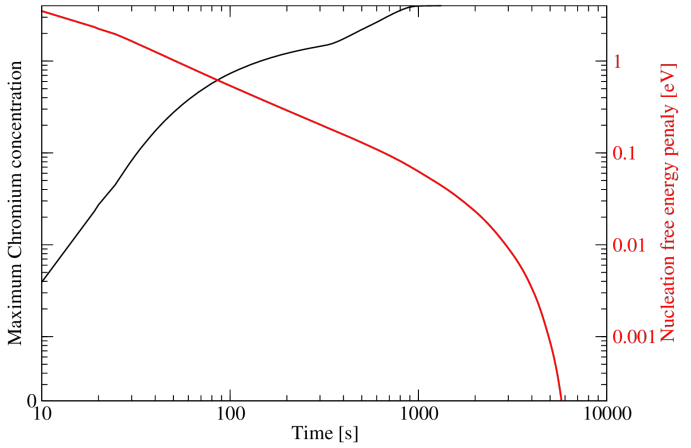


- Modulate free energy at nucleation sites (additive contribution)
- Driving force $\frac{\partial F_n}{\partial c}$ coerces precipitate to form
  - Can be applied to conserved or non-conserved order parameters

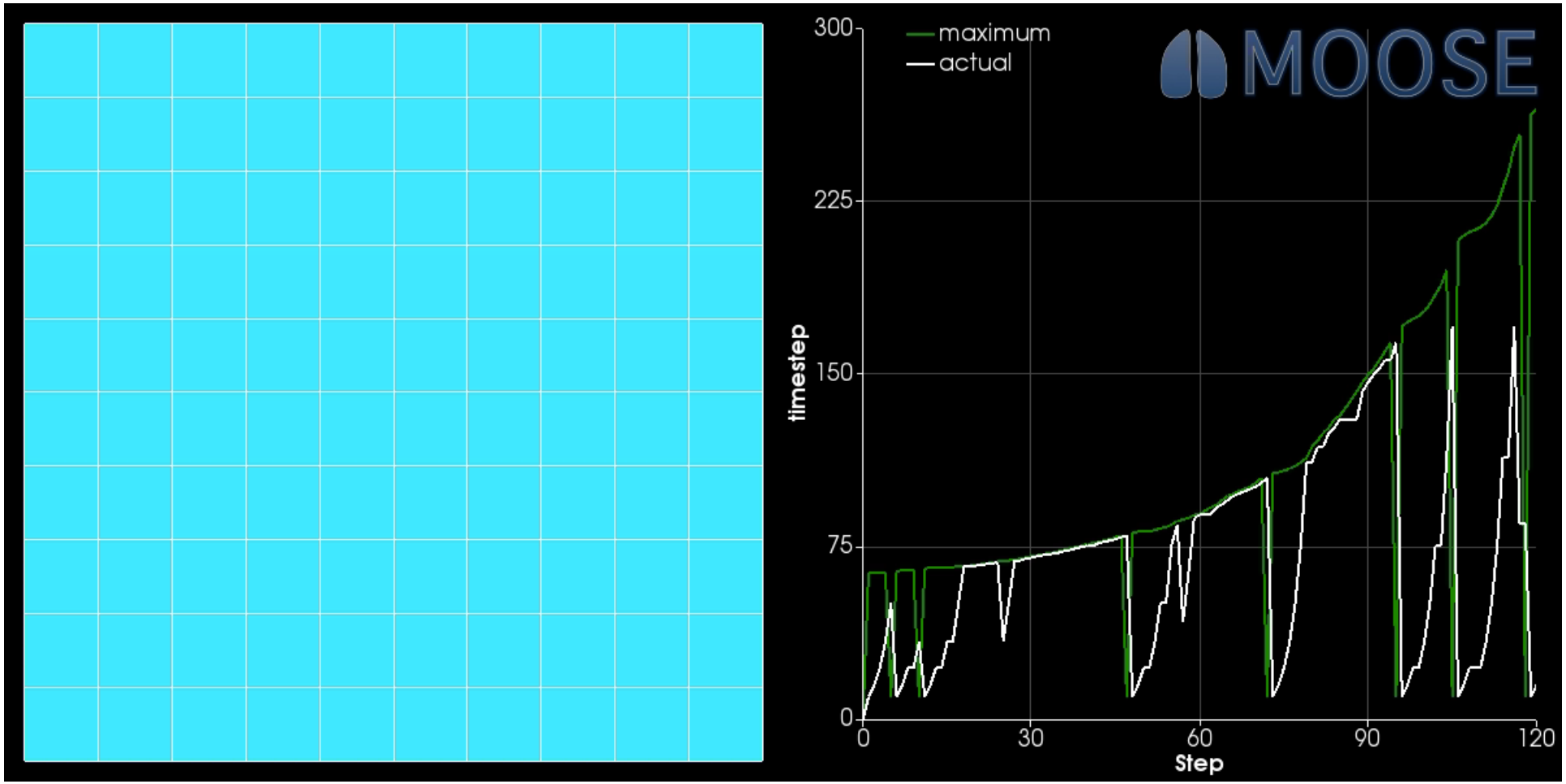


Notice the vanishing nucleation energy penalty F_n as the nucleus forms. In this example c_{Cu} is controlled and c_{Ni} follows automatically.





Nucleation example

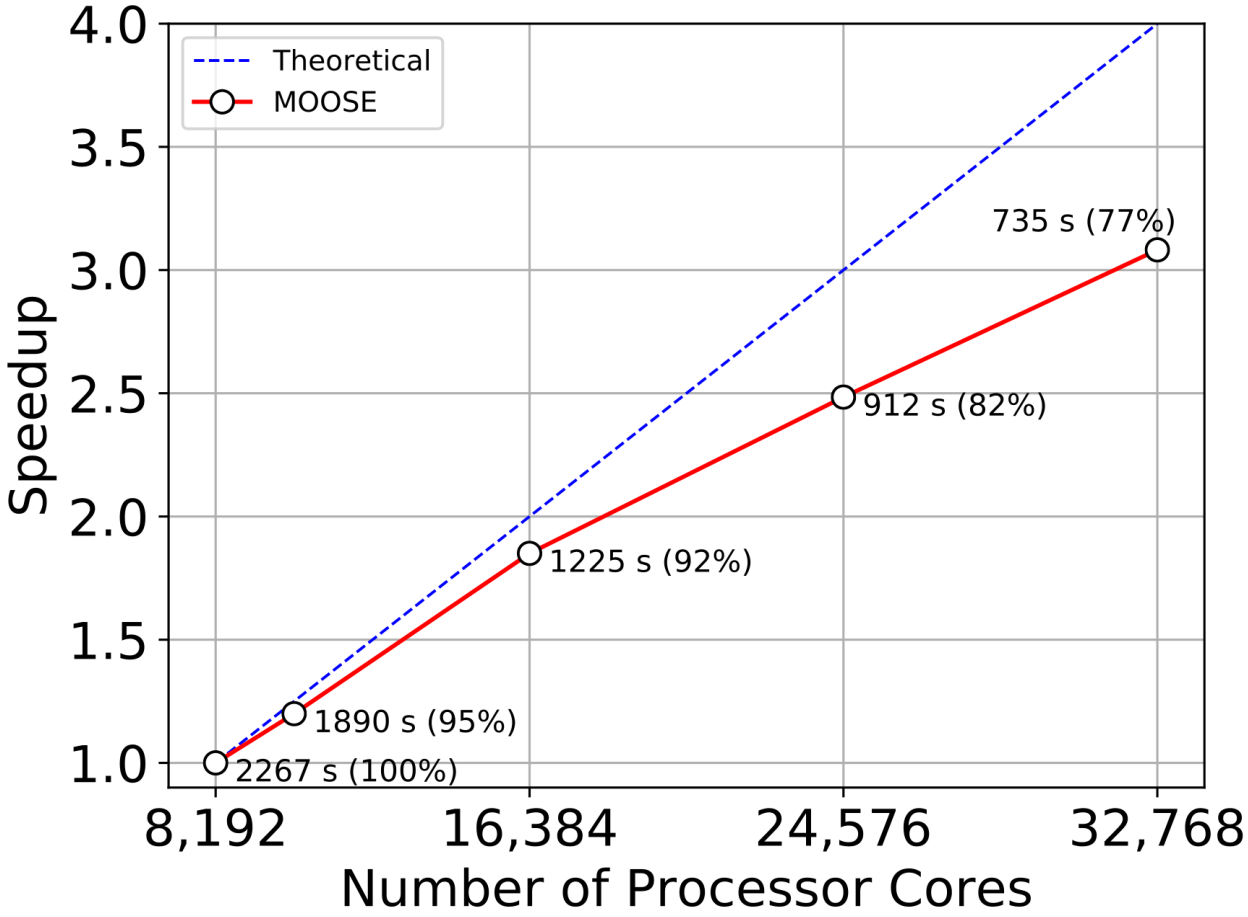


Preemptive mesh refinement, timestep limited by total nucleation probability, and cut-back at each nucleation event.

How could MOOSE utilize Exascale Computing Resources?

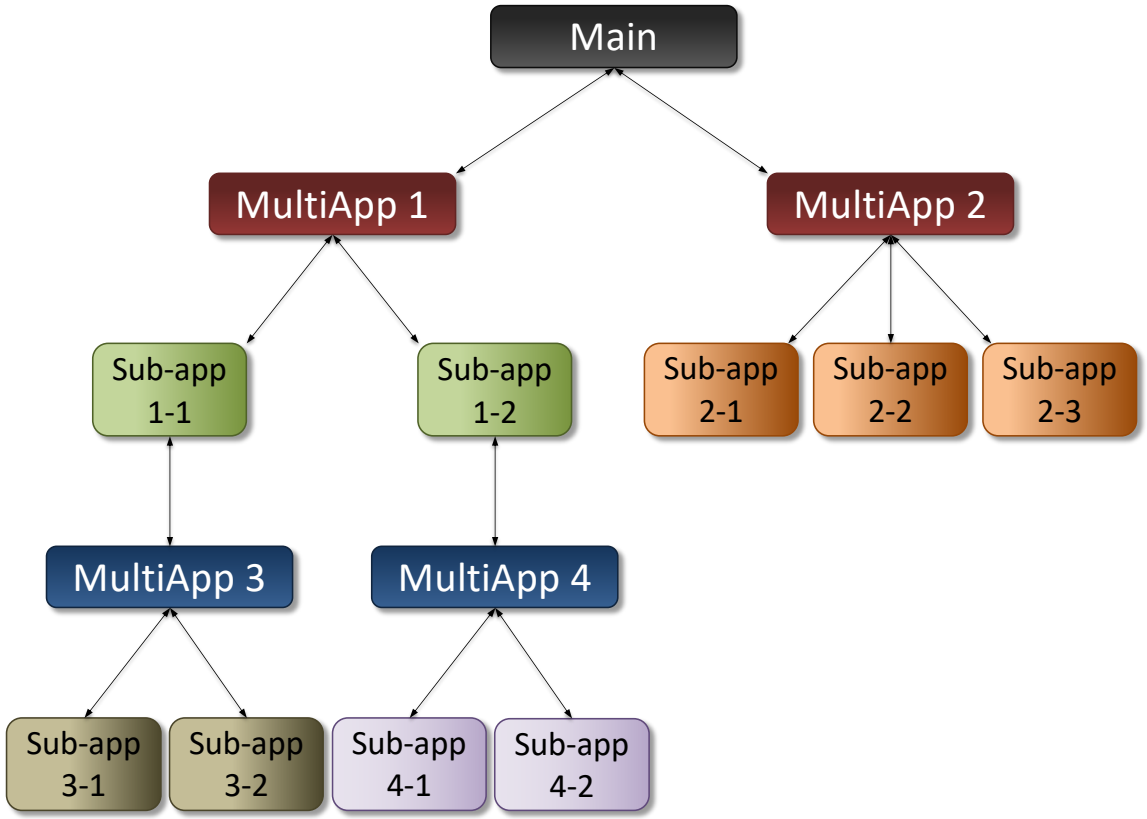
MOOSE Scalability

- MOOSE currently runs on CPUs
  - Hypre non-linear solver has GPU support, but that comes with lots of data transfers CPU↔GPU – for now)
- We are currently working with the MFEM team to define a path forward that will allow MOOSE-based applications to more directly utilize GPUs



MultiApps: Enabling Multiscale Simulation

- MOOSE-based solves can be nested to achieve Multiscale-Multiphysics simulations
  - Macroscale simulations can be coupled to embedded microstructure simulations
- Arbitrary levels of solves
- Each solve is spread out in parallel to make the most efficient use of computing resources
- Efficiently ties together multiple teams' codes
- MOOSE-wrapped apps (wrap arbitrary third party codes, e.g. NEK5000 -> Cardinal)



MOOSE design goals vs. Exascale

- User (Developer, Researcher, and Analyst) time is precious
  - Need versatile simulation tools
  - Easy to implement new physics
  - Hardware agnostic

- Virtually nobody is doing “Exascale computing” yet
  - neither are we
  - but we should be forward looking

- Do we even know what Exascale computing looks like?
  - Frontier ~1.1 EFLOPS
    heterogeneous: half a million CPU cores…
  - Fugaku <0.5 EFLOPS
    7 million CPU cores, **no** accellerators
  - El Capitan
    AMD MI300, zero-copy

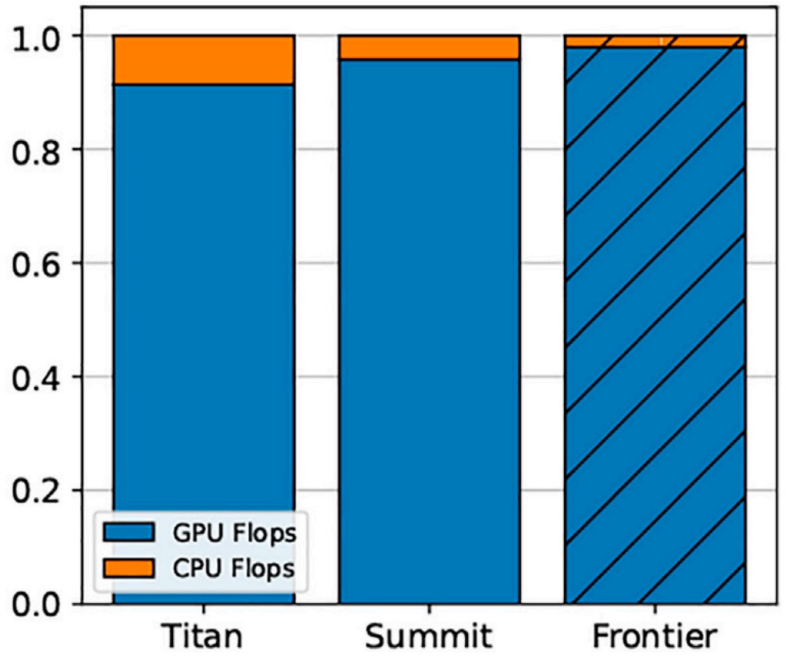
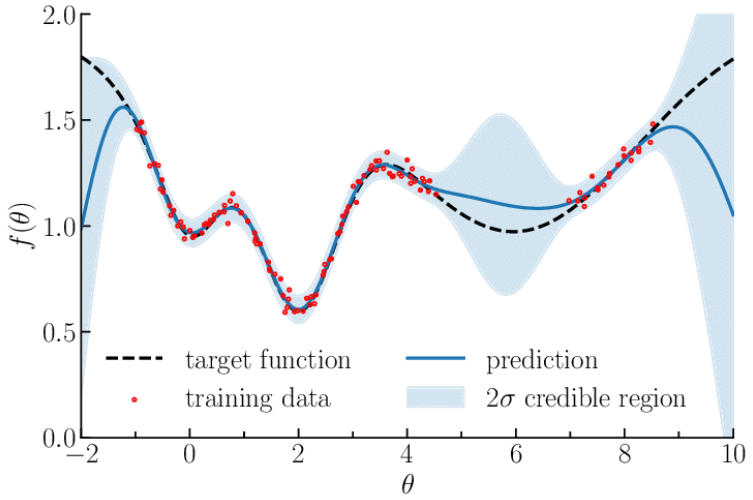


Figure: Apositakis et al., doi: 10.3389/fphy.2022.913510

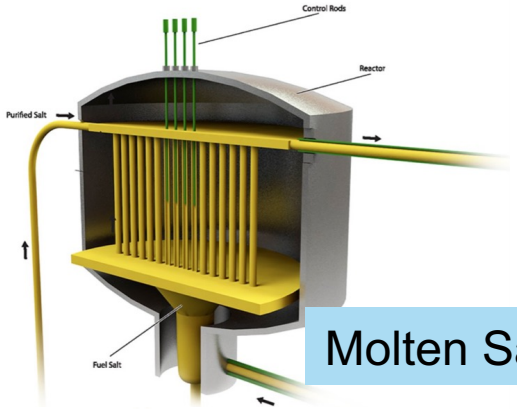
Utilizing Exascale resources

- Parallelizing to extend scales
  - Hard to get good *strong scaling* important for extended time scales
  - Not always easy to get good *weak scaling* important for extended length scales
- How about improving accuracy and *confidence*?
- Improve accuracy using multiscale modeling (MultiApps)
  - Expensive to run many HF/LLS models
- Improve confidence through UQ with stochastic methods (MultiApps)
  - MCMC as the gold standard for UQ is expensive
- Detect rare events / failure probability

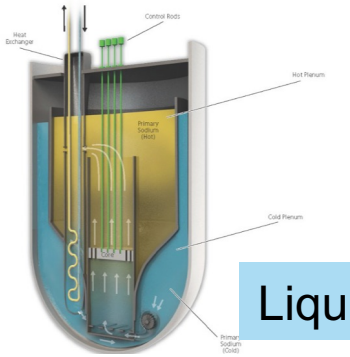


Failure probability analysis

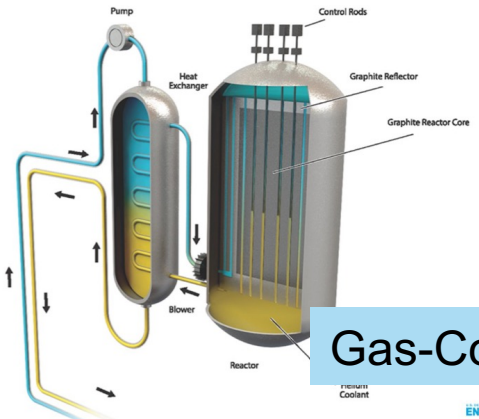
TRISO particles for Advanced Nuclear Reactor



Molten Salt



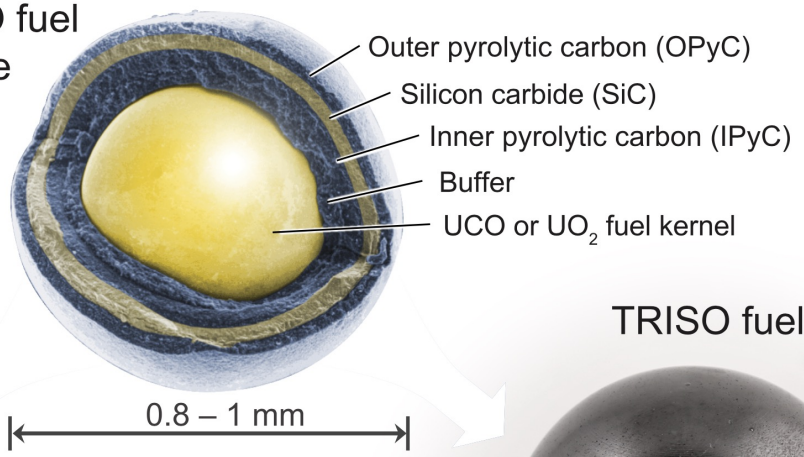
Liquid Metal



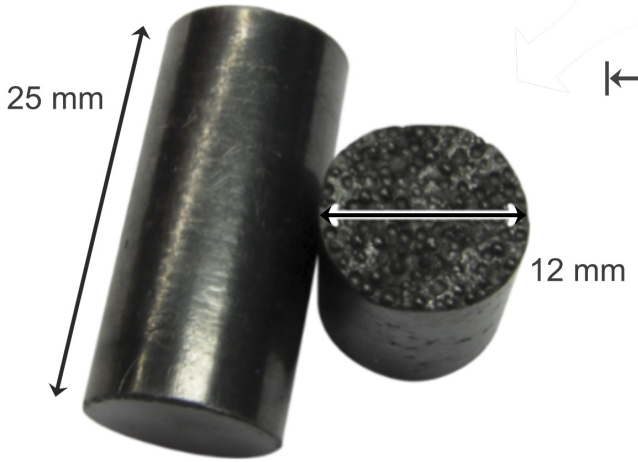
Gas-Cooled

TRISO stands for Tri-structural ISOtropic particle fuel. Reactor vendors such as Kairos Power, X-energy, BWXT, USNC, Westinghouse, Radiant are planning to use TRISO fuel for their small modular and micro-reactor designs.

TRISO fuel particle



TRISO fuel compacts

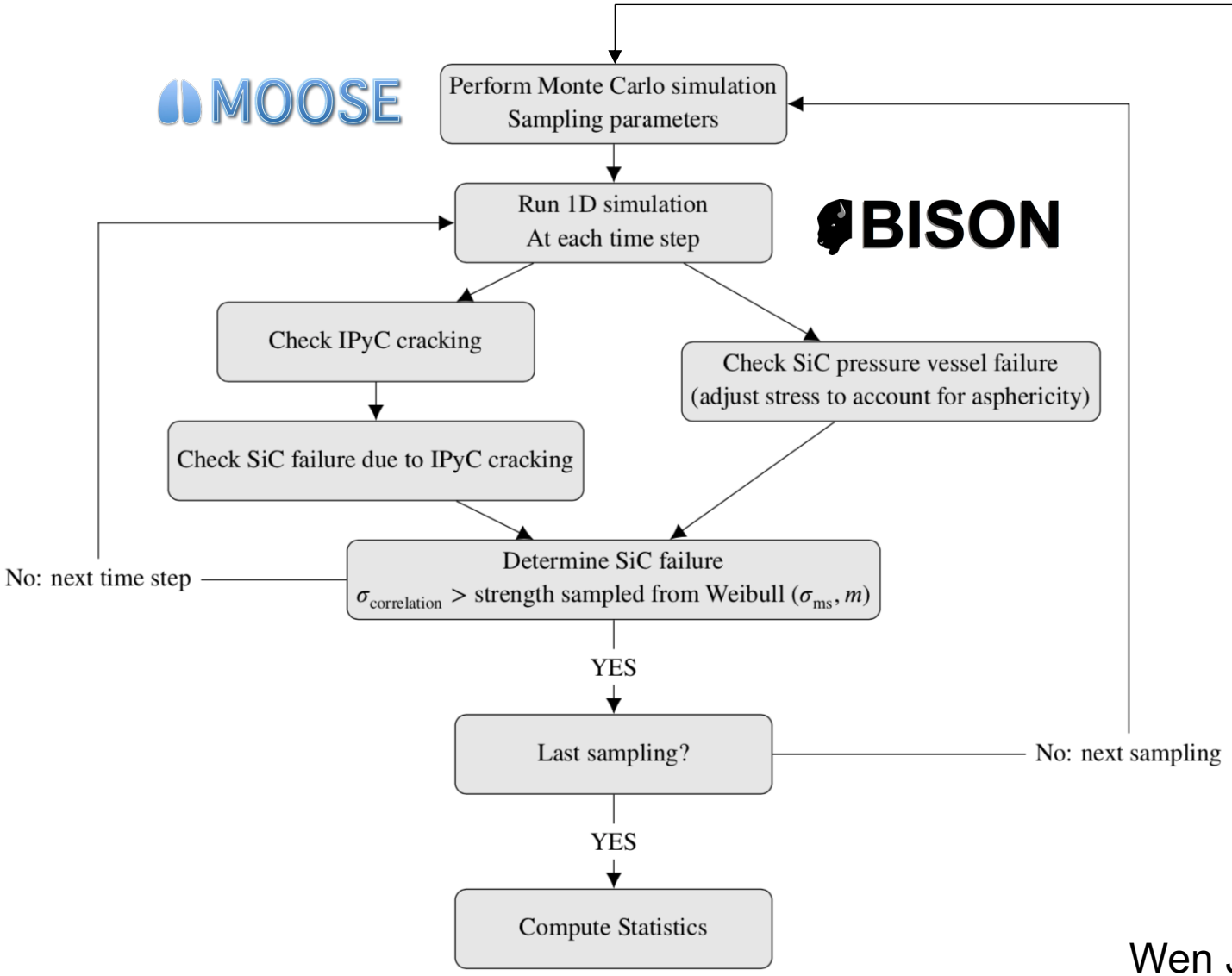


Each pebble contains **~10,000** TRISO particles.

Reactor core contains **~360,000** pebbles.

TRISO fuel pebble

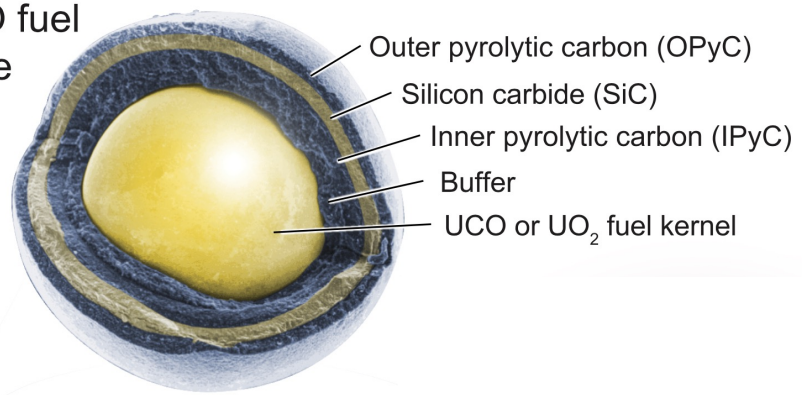
TRISO Failure Analysis using Monte Carlo Simulation



Category	Parameter	Nominal values ± Standard Deviation
Particle geometry	Kernel diameter (μm)	425±10
	Buffer thickness (μm)	100±10
	IPyC/OPyC thickness (μm)	40±3
	SiC thickness (μm)	35±2
	Particle asphericity (SiC aspect ratio)	1.04
Fuel properties	IPyC density (g/cm^3)	1.90±0.02
	OPyC density (g/cm^3)	1.90±0.02
	IPyC/OPyC BAF	1.05±0.005

Metropolis Monte Carlo Sampling

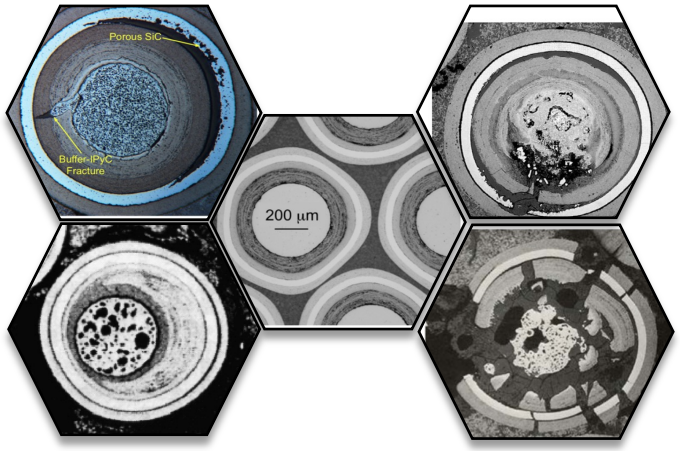
TRISO fuel particle



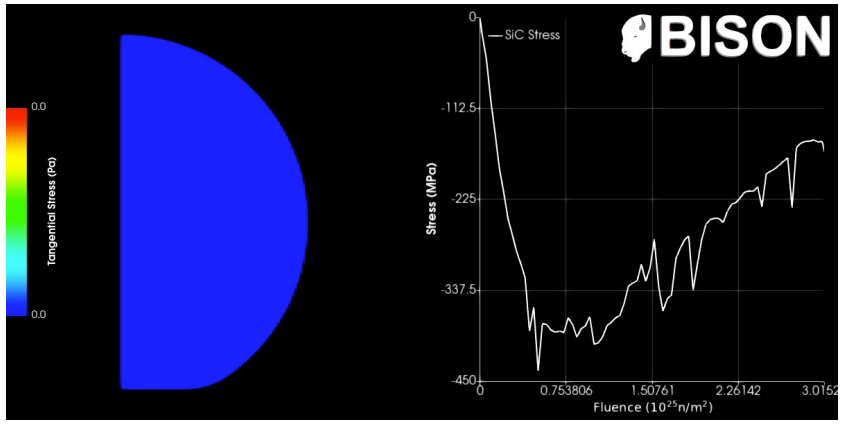
Wen Jiang, INL

Correlate 2d to 1d (stress factors)

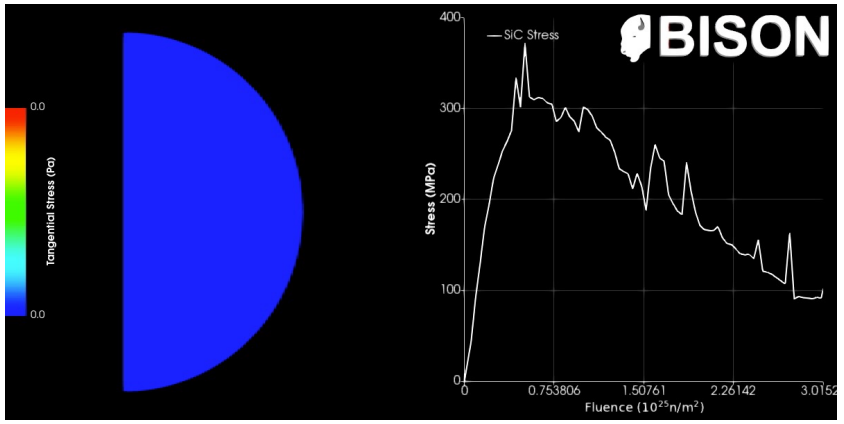
TRISO Failure Analysis using Monte Carlo Simulation



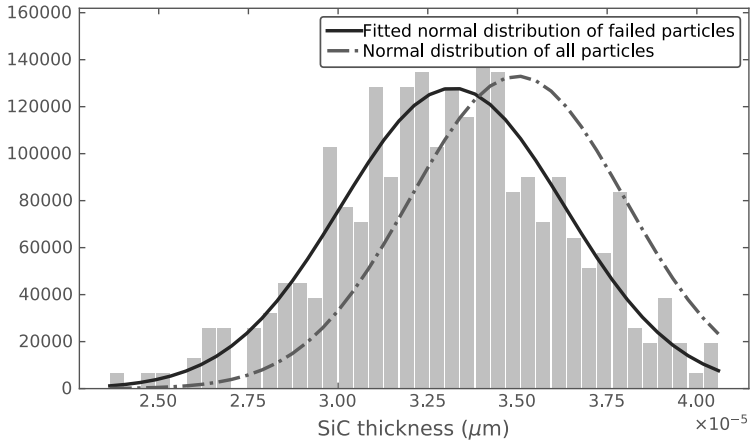
Aspherical (2D)

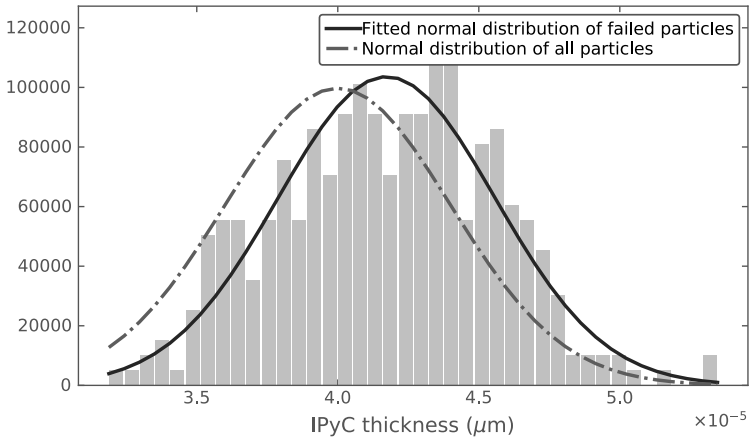


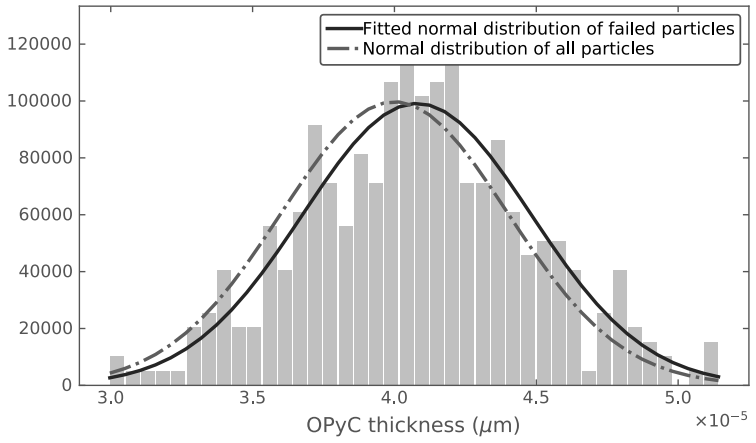
Layer cracking



2D simulations are used to create a 1D surrogate based on stress correlation factors





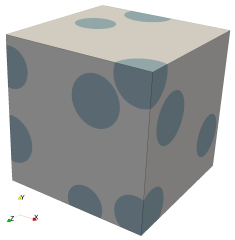


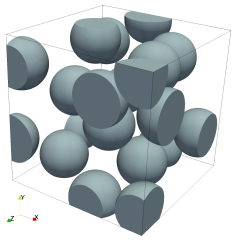
100 million 1D samples with
7600 cores on INL HPC

Wen Jiang, INL

Fuel elements modeling with MOOSE's Multi-App system

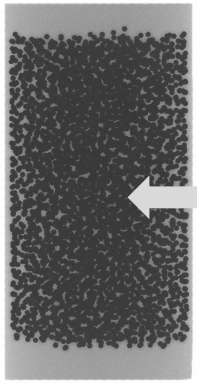
Material property homogenization

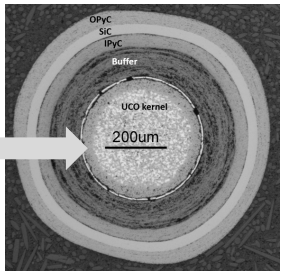




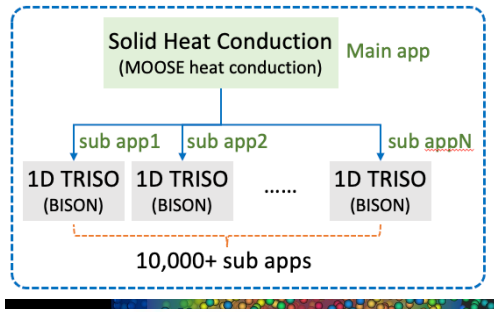
Main App is 3D homogenized matrix

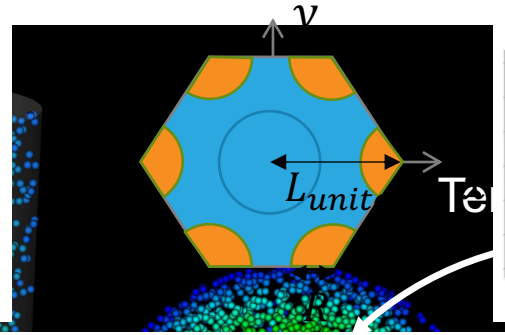
Each TRISO particle is solved individually as a sub-app

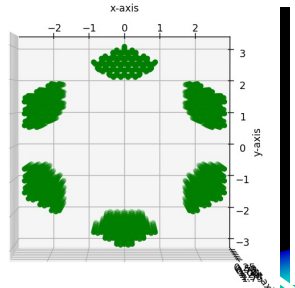


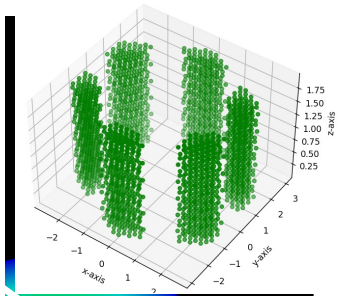


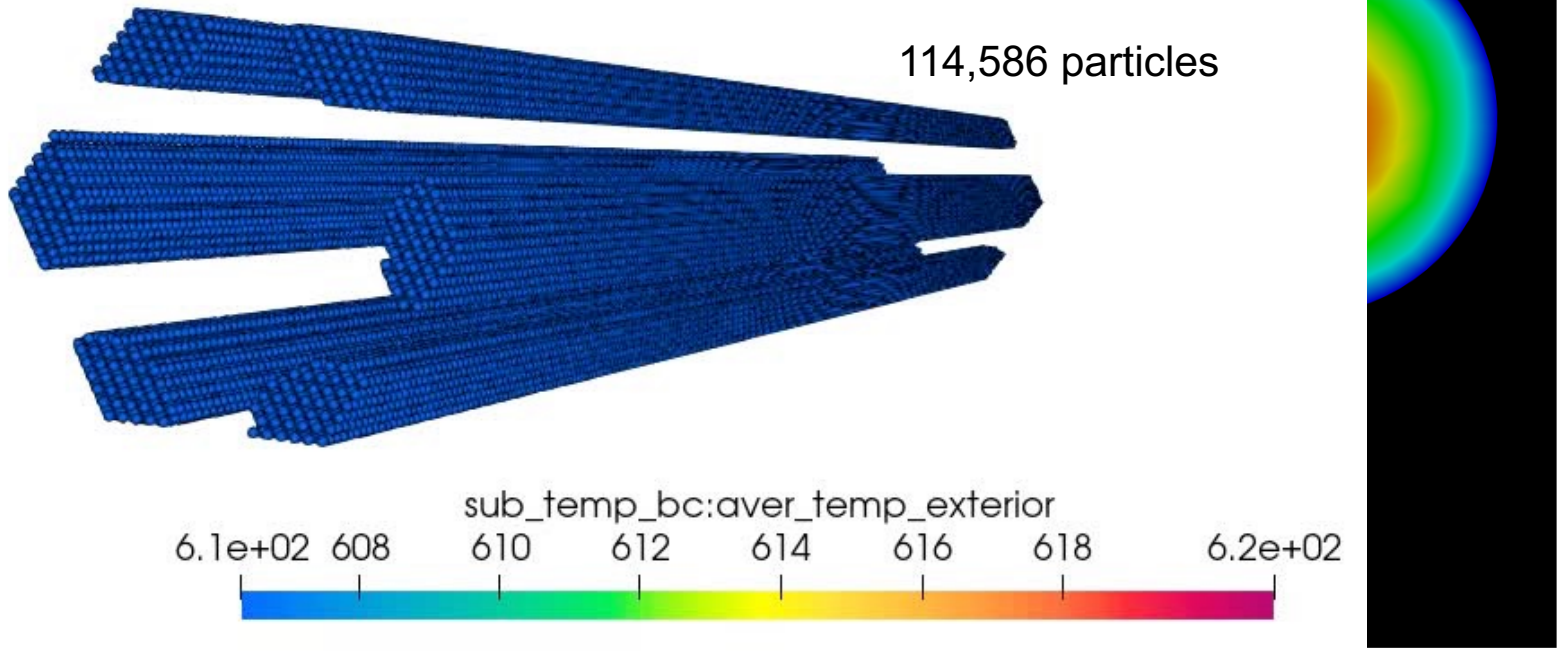
efficient two-way data transfer







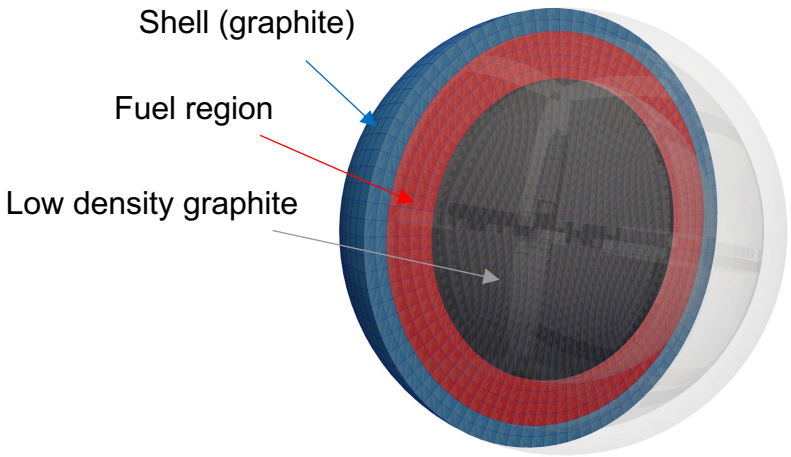


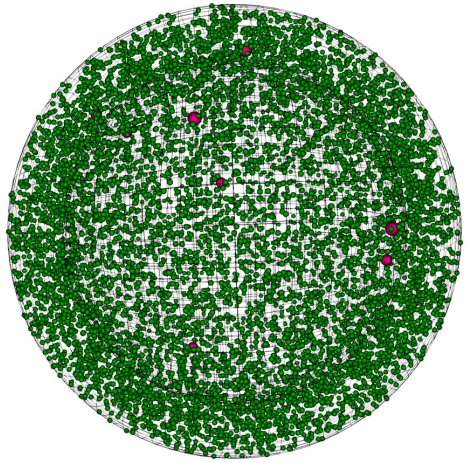


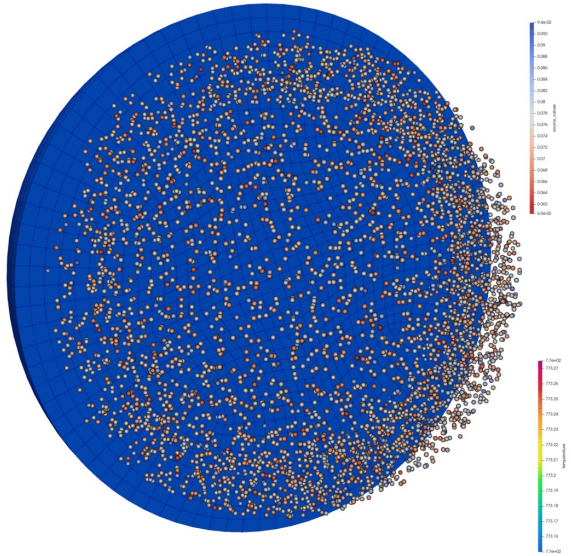
Wen Jiang, Yifeng Che, INL

Fission product diffusion in a pebble

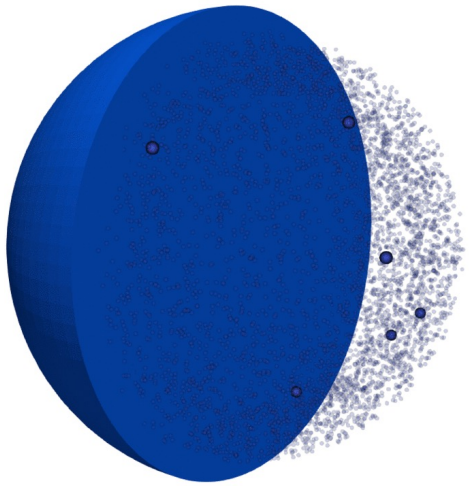
Radius(cm)	**2.000**
Shell layer thickness (cm)	**0.200**
Fuel layer thickness (cm)	**0.420**
(AGR-5/6/7) TRISOs	**9022**
U-235 Enrichment (% wt)	**19.55**

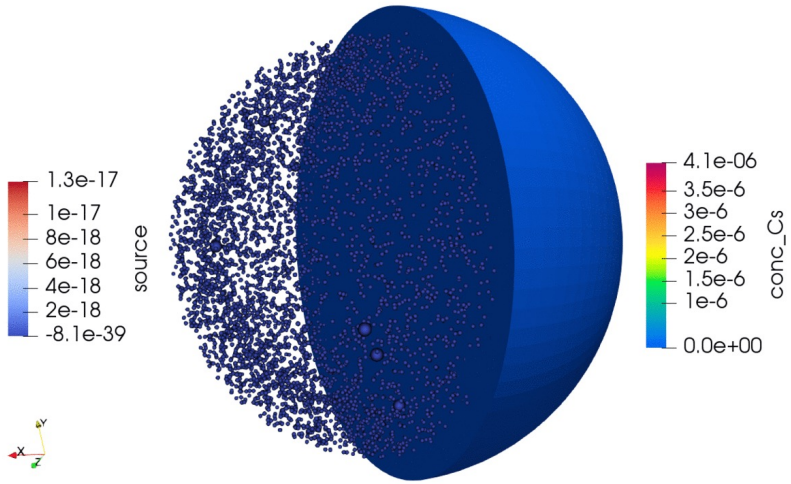




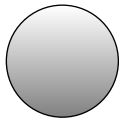


Heat point source

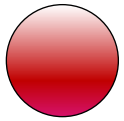




Cs point source



9012 intact particles

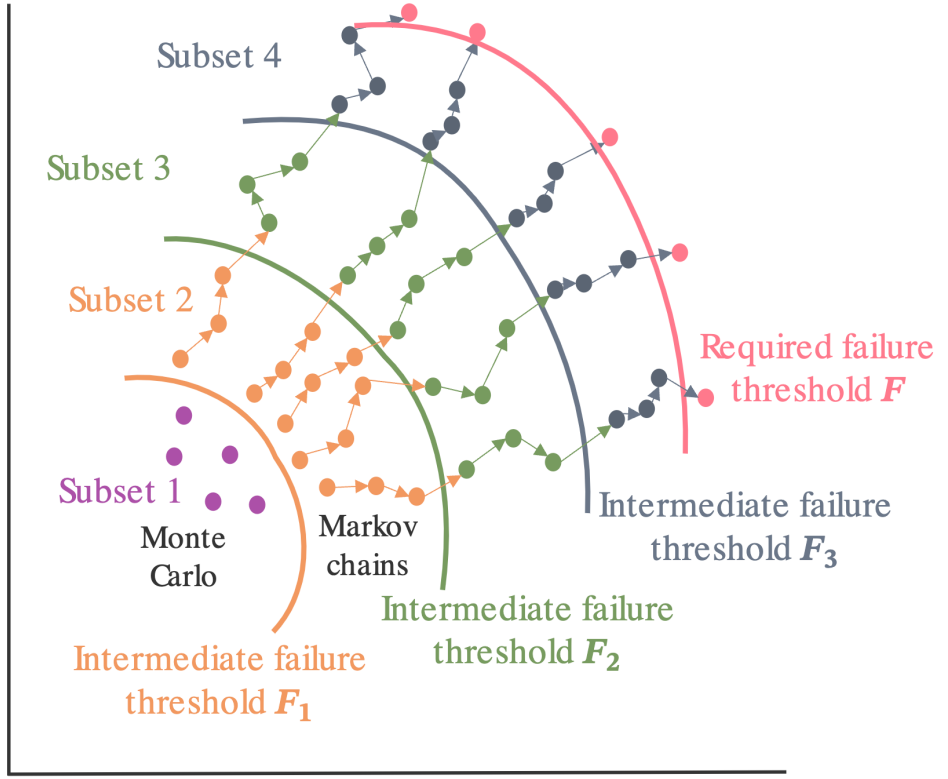


10 failed particles

Exascale opportunities

- Failure probability calculation
  - Higher fidelity lower scale models 1D → 2D → 3D
    - Capture asphericity
    - Capture failure modes
      - Crack formation
  - More advanced sampling methods
    - Parallel Subset Sampling (in MOOSE)
    - Surrogate models with active learning (LDRD)

- Multiscale coupling
  - Higher fidelity lower scale models 1D → 2D → 3D
  - On the fly particle failure

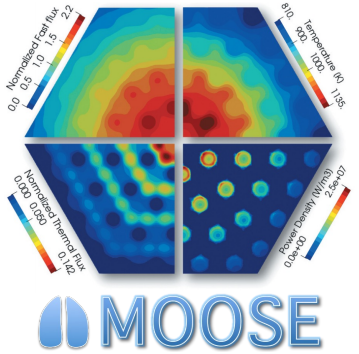


Bayesian Inference

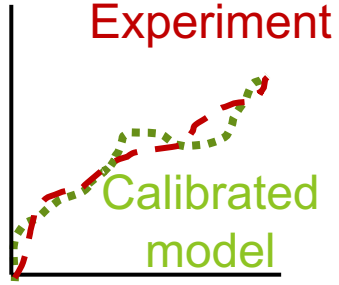
State of practice of computational model validation

Input parameters:
Fabrication
Material
System

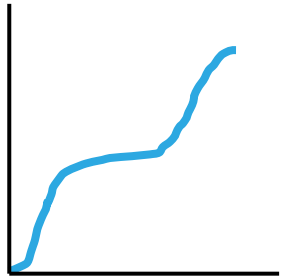
Step ①:
model and inputs



Step ②:
validation and calibration

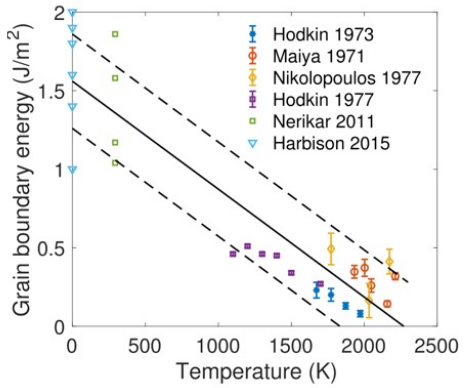


Step ③:
forward prediction

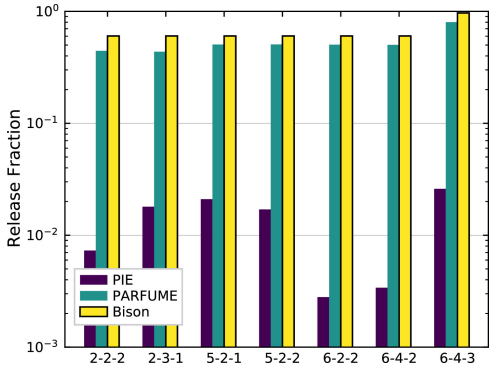


Key questions:

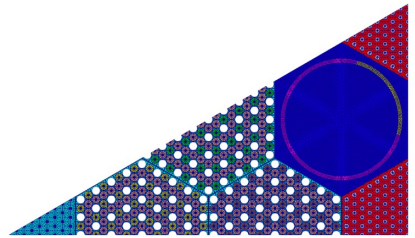
Input parameter uncertainties
(UO_2 material)



Model prediction uncertainties
(TRISO fuel Ag release)

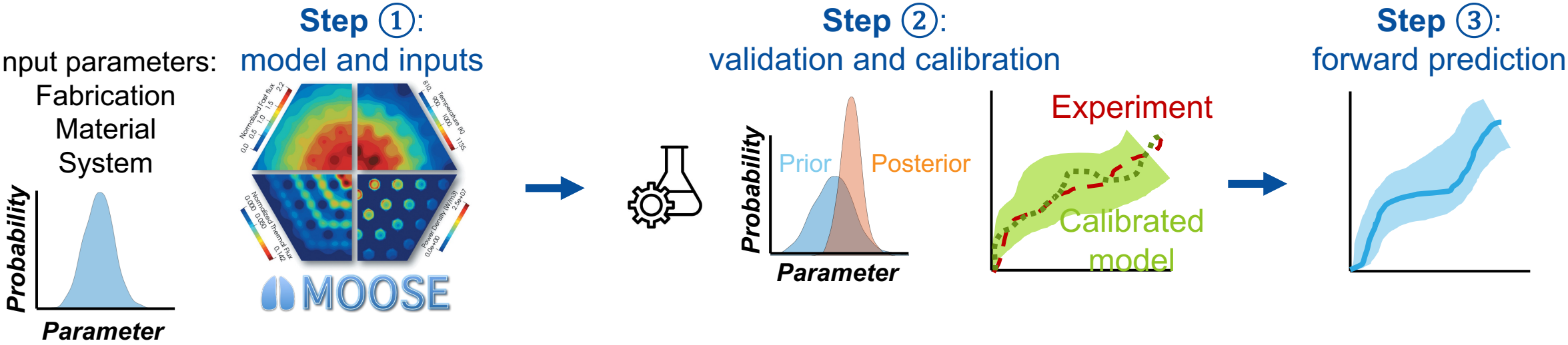


Limited experimental data
(Heat pipe reactor)



(**References:** Ikonen et al. 2014, Bratton et al. 2014, Tonks et al. 2021, Hales et al. 2022)

Bayesian calibration and UQ of computational models



- Identifies and propagates parameter uncertainties
- Reflects model mismatches with experiments during forward simulations
- Permits Bayesian optimal experimental design

(**References:** Wu et al. 2021, Hoff 2009, Che et al. 2021)

Bayesian inference with computational models

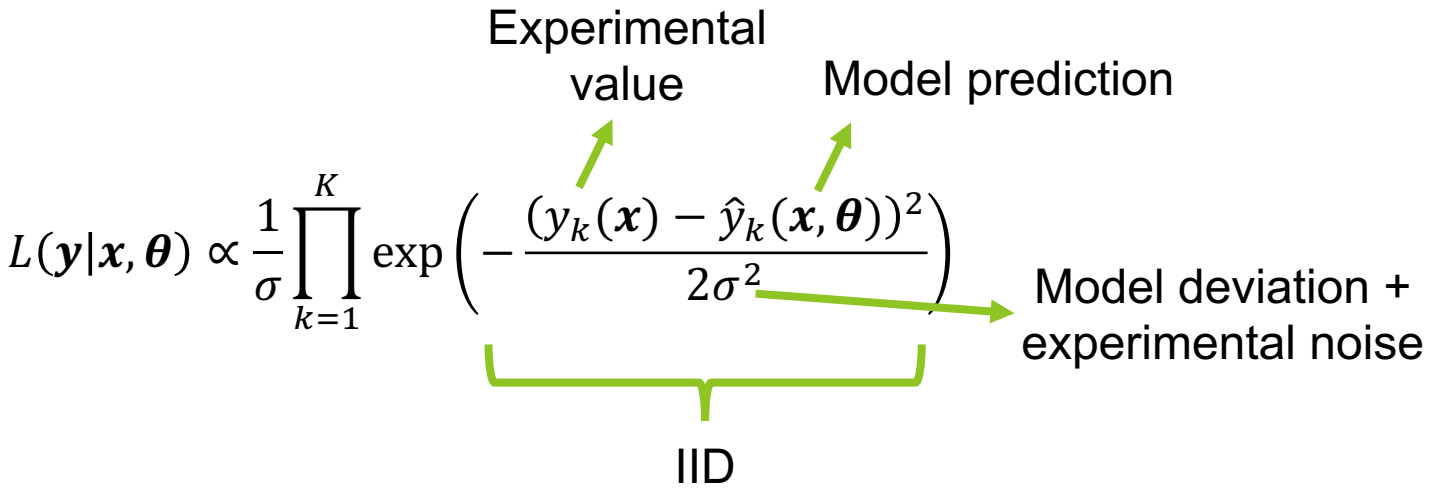
$$f(\boldsymbol{\theta}|\boldsymbol{x},\boldsymbol{y}) \propto L(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})\, f(\boldsymbol{\theta})$$

$f(\boldsymbol{\theta})$: $Prior\ of\ the\ uncertain\ model\ parameters$

$L(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})$: $Likelihood\ of\ observing\ the\ data\ given\ model\ inputs, parameters, and\ predictions$

$f(\boldsymbol{\theta}|\boldsymbol{x},\boldsymbol{y})$: $Posterior\ of\ the\ uncertain\ model\ parameters\ given\ model\ inputs\ and\ predictions$

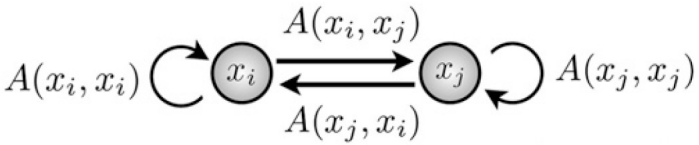
$f(\boldsymbol{\theta})$: Any user-specified distribution (uniform, bounded Normal)



Inferred params include **θ** + $\boldsymbol{\sigma}$

Computational aspects: serial and parallel MCMC

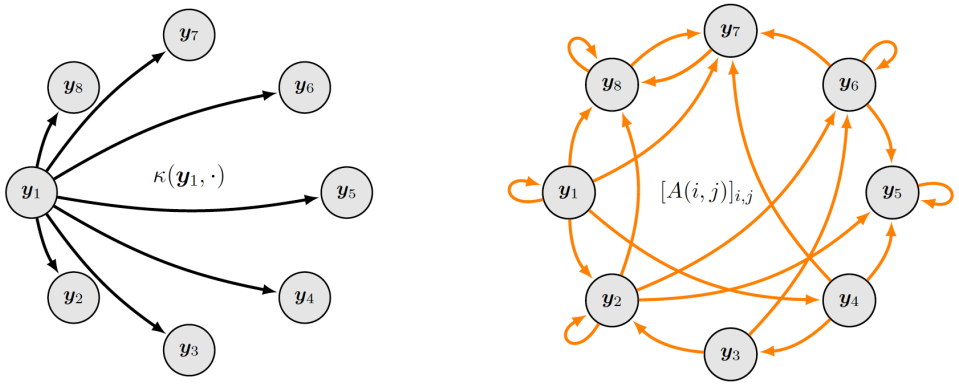
Standard Metropolis-Hastings



- Propose a new input
- Evaluate model and likelihood
- Compute the TPM, $A(i,j)$
- Draw an integer b/w [0, 1] with weights $A(i,j)$
- Assign new input

***M* model evaluations in serial (impractical for us)**

Parallel Metropolis-Hastings

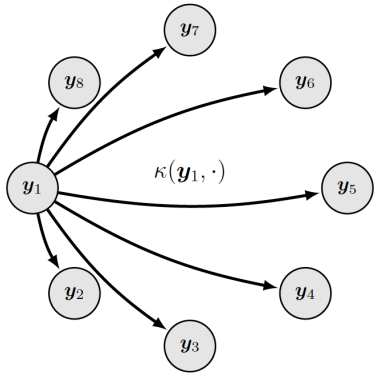


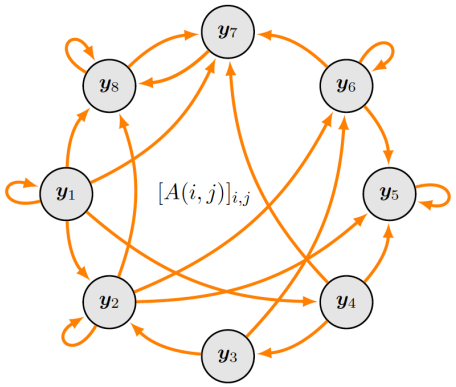
- Propose *N* new inputs
- Evaluate models and likelihoods (parallelization)
- Compute the TPM, $A(i,j)$
- Sub-sampling (generate *N* new points):
  - Draw an integer b/w [0, N] with weights $A(i,j)$
  - Assign new input

***M/N* model evaluations in serial**

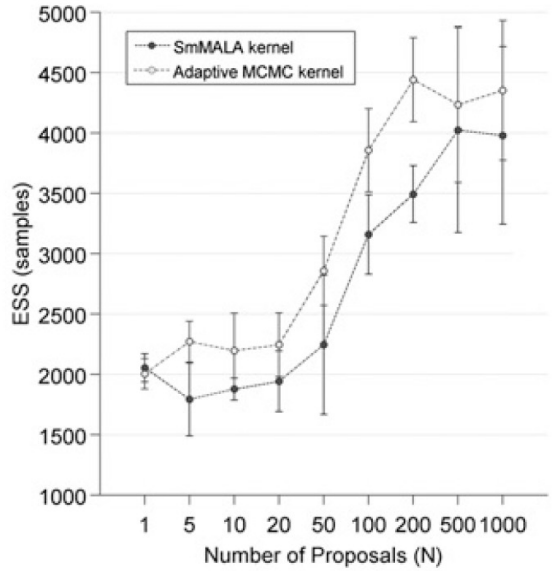
Calderhead (2014) *Proceedings of the National Academies of Sciences*.
Schwedes and Calderhead (2021) *International Conference on AI and Statistics*.

Computational aspects: parallel MCMC





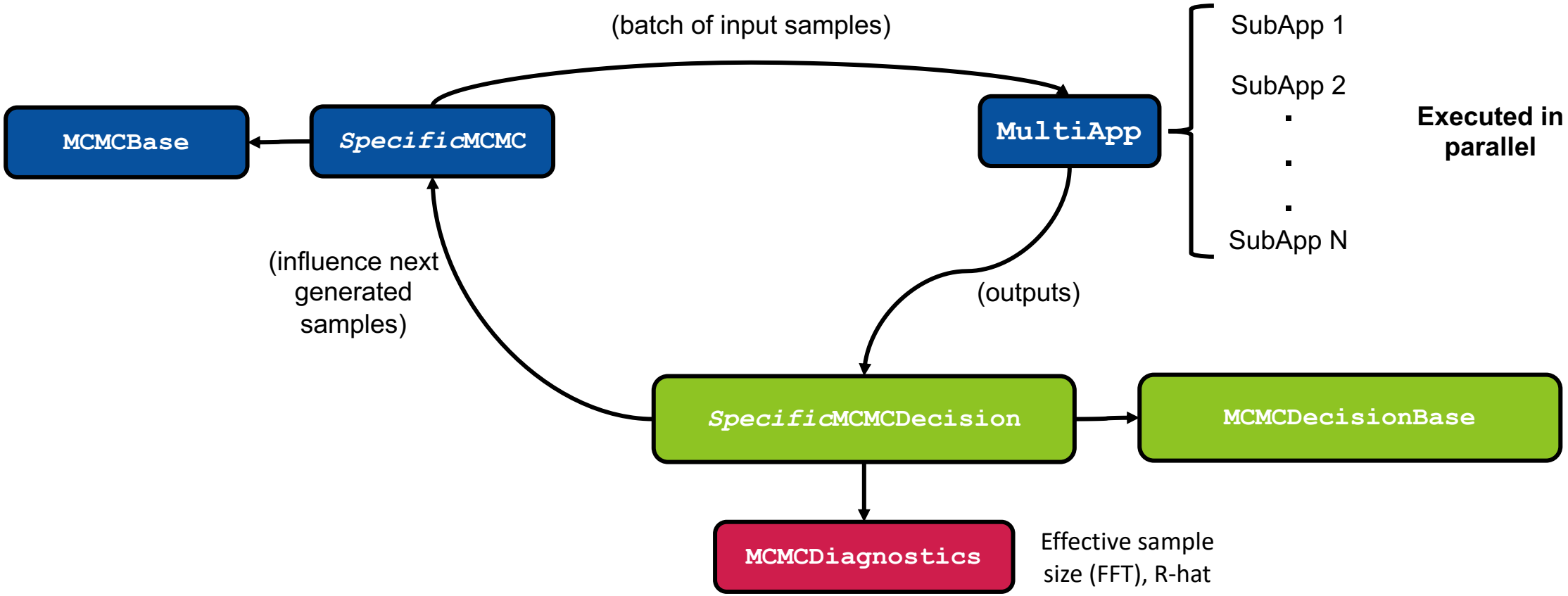
Algorithm by Calderhead 2014



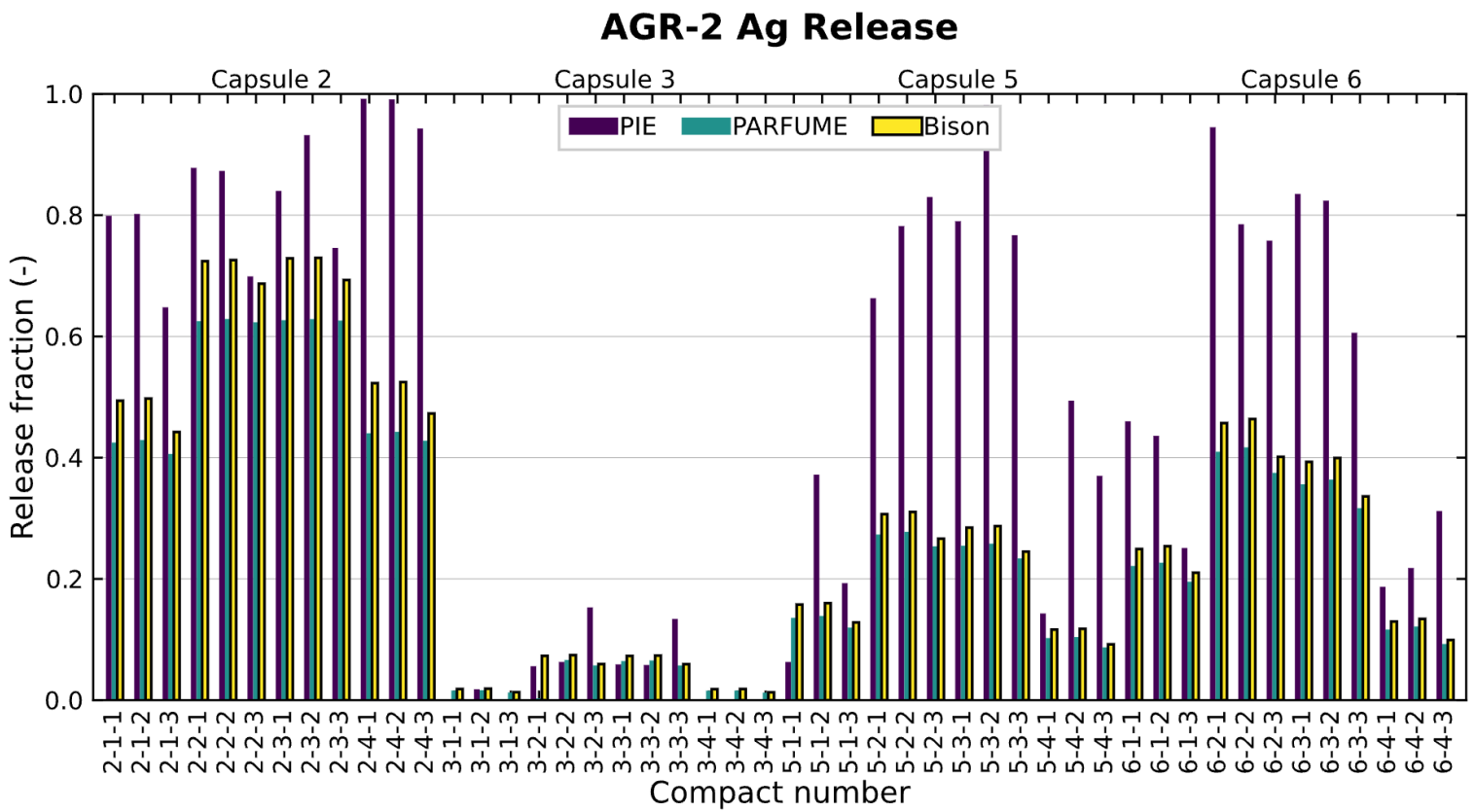
- *N* can be between 1 and 1000 (massive parallelization)
- *N*=1, standard serial MCMC
- Proposal kernel is flexible: random-walk (adaptive, delayed rejection), Langevin, Hamiltonian
- Improved performance compared to serial M-H
- Limitations??

Calderhead (2014) *Proceedings of the National Academies of Sciences*.
Schwedes and Calderhead (2021) *International Conference on AI and Statistics*.

MOOSE implementation



Case Study: TRISO AGR-2 experiments



- Advanced Gas Reactor
  Bison model prediction: *Ag release*
- 36 Post Irradiation Examination (PIE) data for UCO fuel kernel
- Considerable uncertainties in model predictions compared to experiments
- Model params: Pre-factor *A* and activation energy E_a for SiC, PyC, and fuel kernel (6 params)
- Experimental configurations:
  T, fluence, heat rate (time varying)

Case Study : TRISO AGR-2 experiments

Inverse analysis

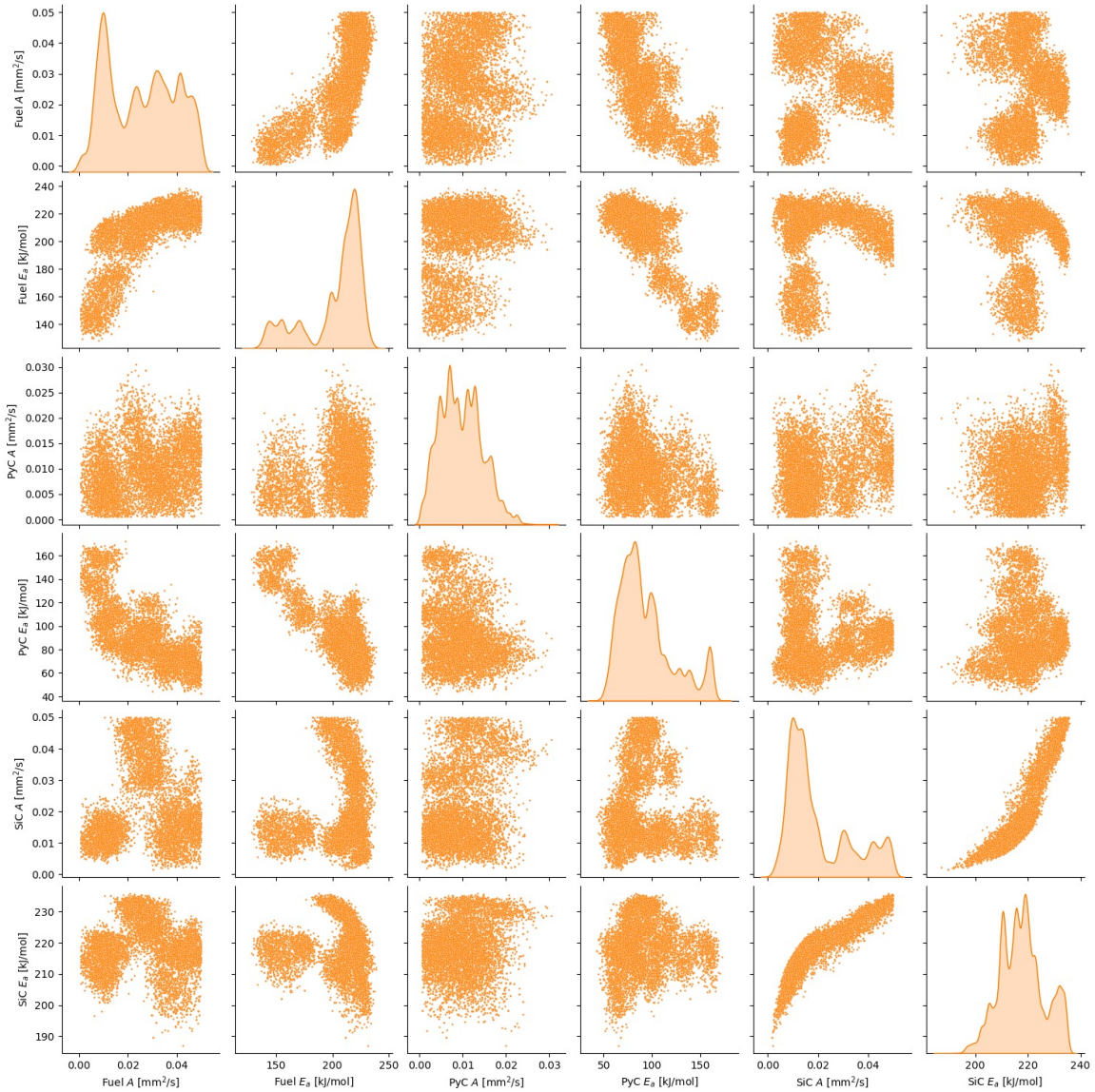
$$f(\boldsymbol{\theta}|\boldsymbol{x},\boldsymbol{y}) \propto L(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})\, f(\boldsymbol{\theta})$$

Infer 6 params (pre-factor/activation E)

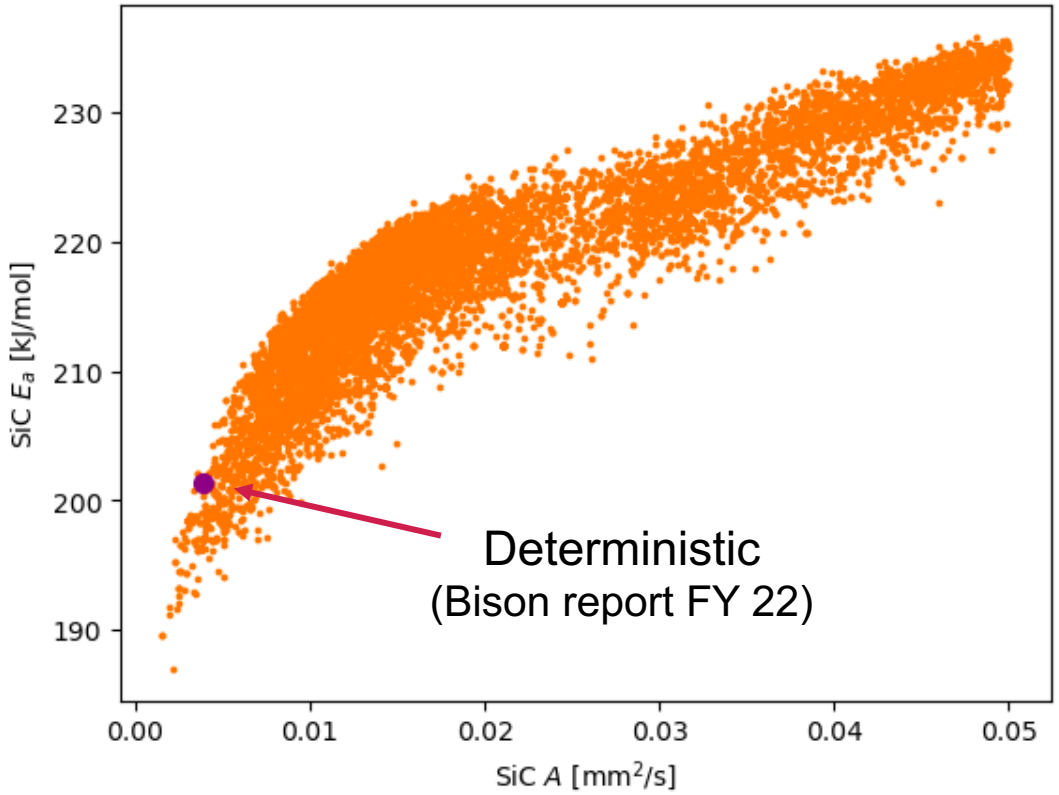
Sigma fixed to 0.1

- 50 parallel proposals
- Each proposal requires 36 model evals: 36 experimental configurations
- Analyzed using 1800 procs on Sawtooth
- Lower bounds: `5e-10; 165e2`
- Upper bounds: `5e-8; 165e4`
- Proposal stds: `3e-9; 5e3`

Case Study : TRISO AGR-2 experiments

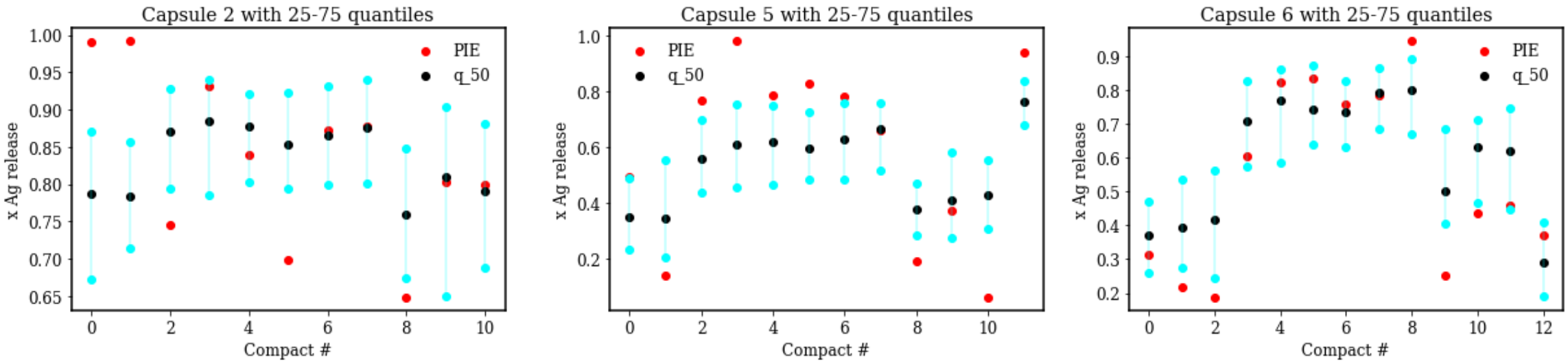


Inverse analysis



Case Study : TRISO AGR-2 experiments

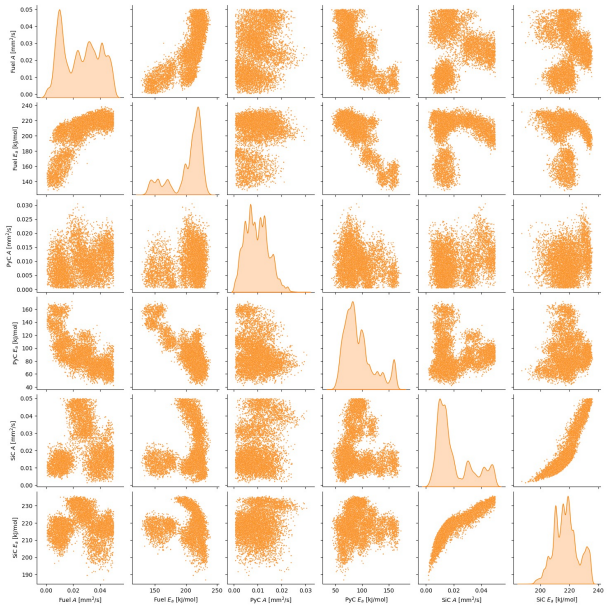
Forward analysis

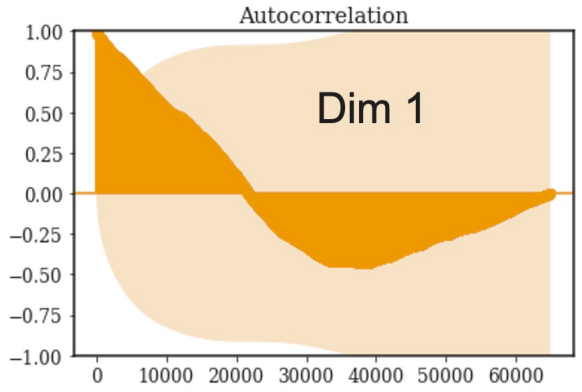


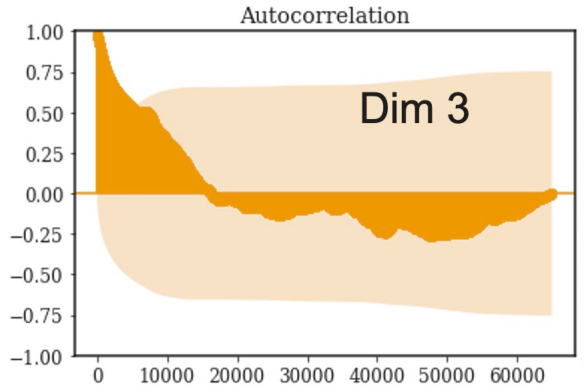
Sigma fixed to 0.1

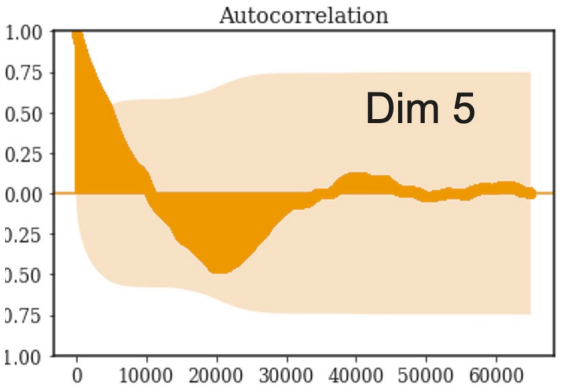
Uncertainty in the uncertainty estimation

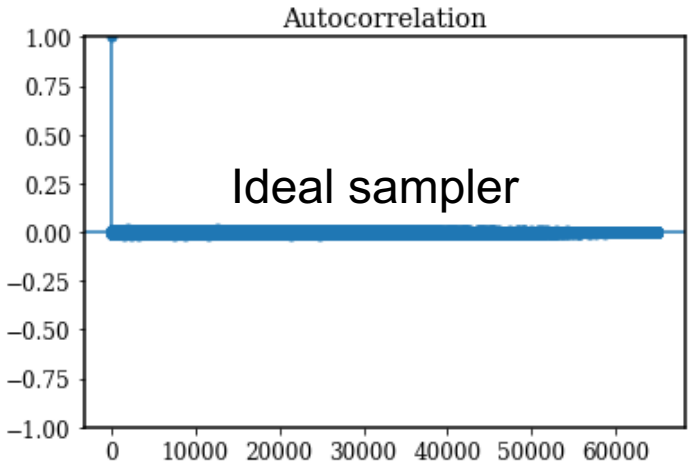
Sampling quality











Total samples: 65,000

Acceptance rate: 12%

Effective sample size (ESS):
6.99857546, 11.38694062,
4.91145346, 9.7290921 , 2.95958647,
3.36655714

Parallelization of MCMC is still better than serial MCMC

Affine Invariant Differential Evolution Sampler (AIDES)

- Uses two states other than the current state from the ensemble of walkers to propose a differential step
- This differential step is added to the current state to become the new proposal for the walker
- Procedure repeated across all ensemble of walkers
- Mathematically, the differential component of the proposal is (γ, ξ are the internal params)

$$\boldsymbol{\delta X} = \gamma\left(\boldsymbol{X}^a - \boldsymbol{X}^b\right) + \boldsymbol{N}(0, \xi)$$

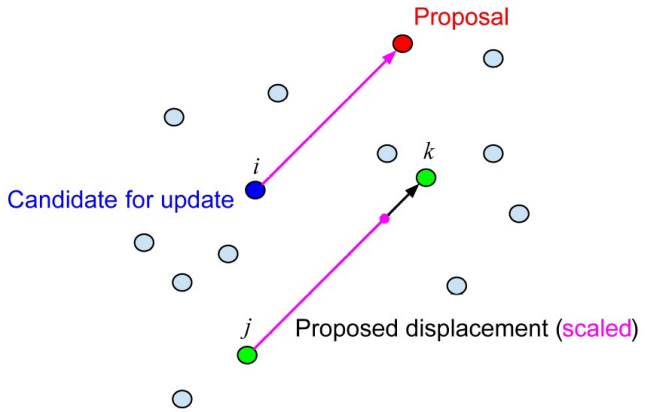
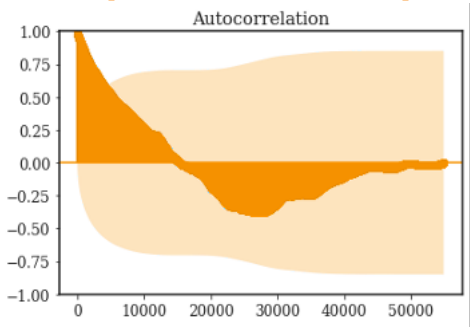


Figure 1. An illustration of the DEMCMC process. For a trial step for state i (blue), we consider two additional states, j and k (both green). A proposal vector is drawn between j and k, and the vector is scaled by γ (magenta) before being added to state i to generate a trial state (red).

(Cajo and Braak 2006; Vrugt 2016)

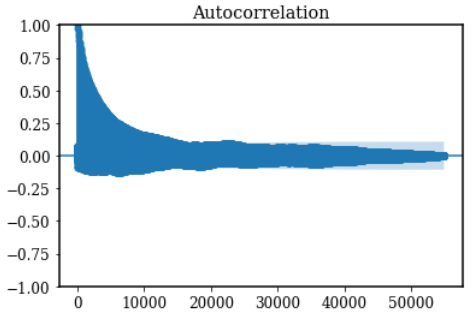
AIDES applied to TRISO FGR AGR-2

Independent Metropolis-Hasting



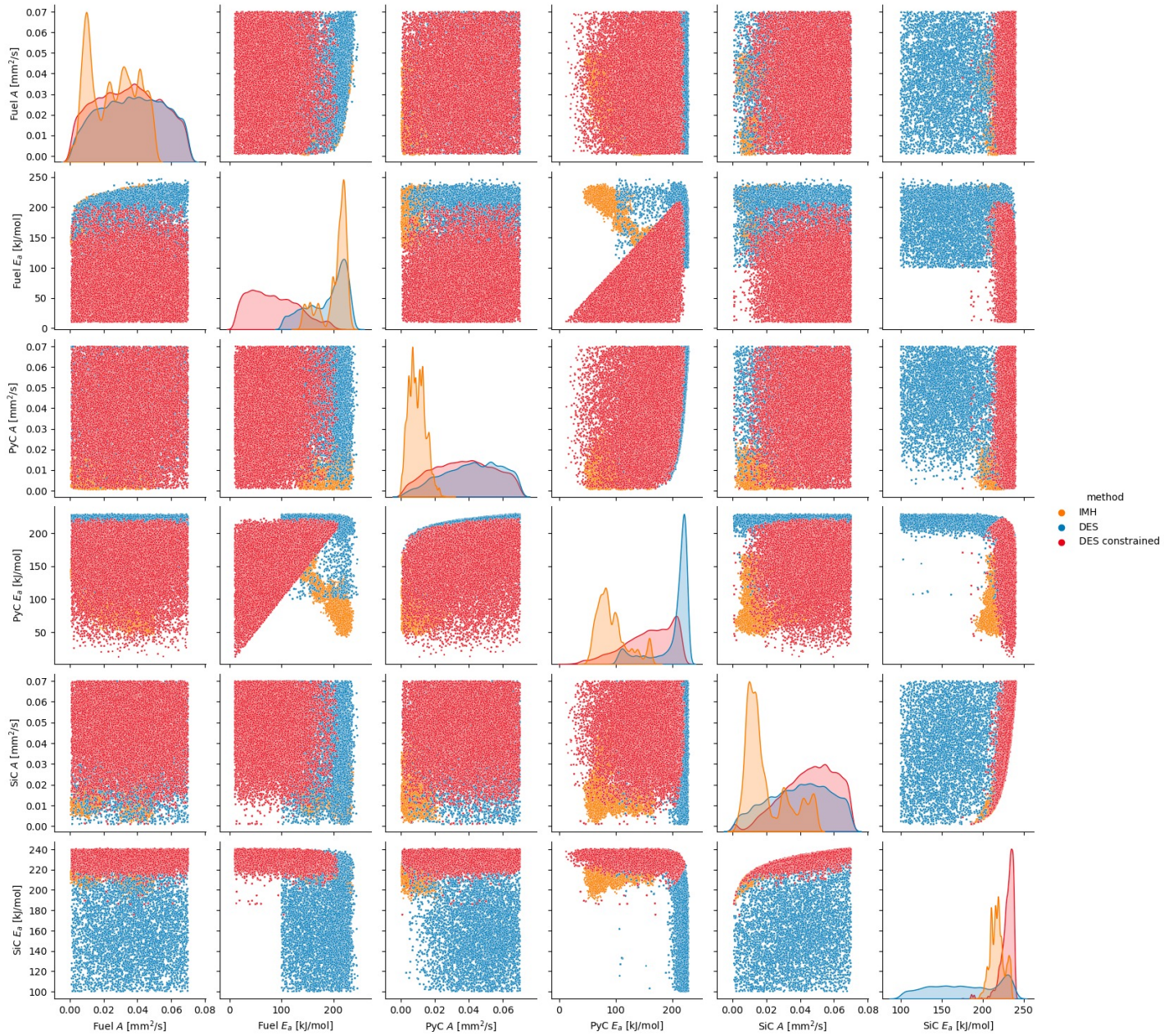
Effective sample sizes:
3.89, 4.44,
24.41, 4.97,
3.79, 4.40

Differential Evolution



Effective sample sizes:
54668.42, 49552.78,
53355.39, 46674.88,
54900., 49619.29

- 55,000 total samples with 50 parallel proposals **Effective parallelization**
- 1D Ag diffusion through the particles
- Better exploration of parameter space with Affine Invariant Differential Evolution Sampler



Som Dhulipala, INL

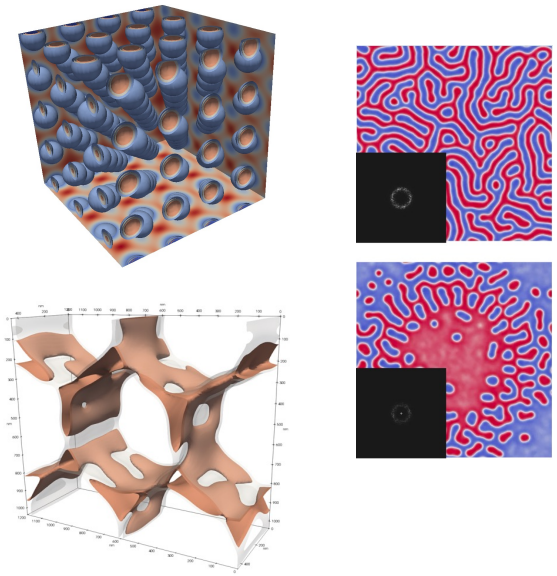
Exascale opportunities

- Parallel sampling
  - Parallelize over experimental data points
  - Parallelize individual model evaluations
  - Parallel Markov Chains
- Infer model parameters with uncertainties from complex experiments (e.g. Taylor impact test)
- ToDo: Gradient based methods
  - Gradient computation requires software retooling and comes with a substantial cost
  - No established benchmarks for parallel gradient based MCMC

Summary

Materials Modeling

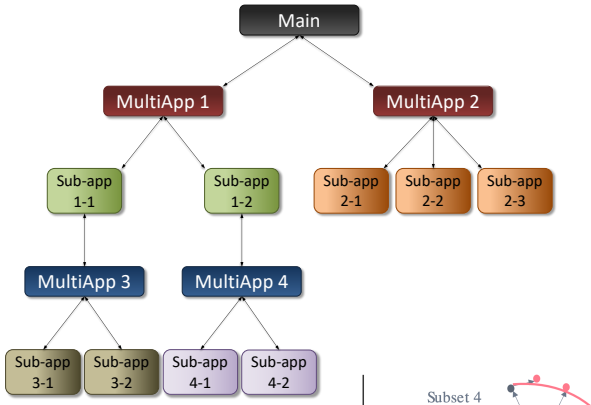
- Rapid model development



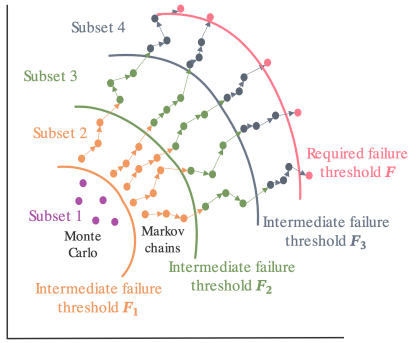
- Multiphysics coupling
- Effortless parallelism

Parallel multiscale architecture

- Sub-application system
- All data *in memory*
- Wrap external codes

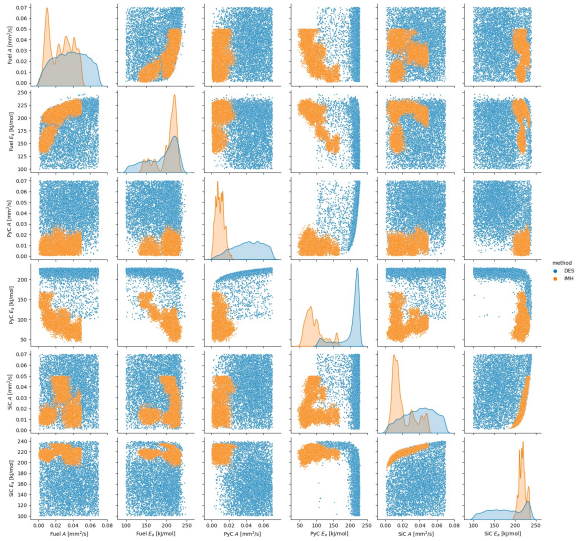


- Failure probability / rare events sampling



Stochastic Tools

- Parallel MCMC
- Inverse and forward Bayesian inference



Thank you! Questions?