INL/EXT-15-35108

Assessment of MARMOT: A Mesoscale Fuel Performance Code

M. R. Tonks, D. Schwen, Y. Zhang, P. Chakraborty, X. Bai, B. Fromm, J. Yu, M. C. Teague, D. A. Andersson

April 2015

INL/EXT-15-35108

Assessment of MARMOT: A Mesoscale Fuel Performance Code

M. R. Tonks, D. Schwen, Y. Zhang, P. Chakraborty, X. Bai, B. Fromm, J. Yu, M. C. Teague, D. A. Andersson

April 2015

Idaho National Laboratory

Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Office of Nuclear Science
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

Assessment of MARMOT:
A Mesoscale Fuel Performance Code

M. R. Tonks, D. Schwen, Y. Zhang
P. Chakraborty, X. Bai, B. Fromm, J. Yu, M. C. Teague

Computational Materials Science Group
Fuels Modeling & Simulation Department
Idaho National Laboratory
Idaho Falls, ID

D. A. Andersson
MST-8
Los Alamos National Laboratory
Los Alamos, NM

April 2015

Abstract

MARMOT is the mesoscale fuel performance code under development as part of the US DOE Nuclear Energy Advanced Modeling and Simulation Program. In this report, we provide a high level summary of MARMOT, its capabilities, and its current state of validation. The purpose of MARMOT is to predict the coevolution of microstructure and material properties of nuclear fuel and cladding due to stress, temperature, and irradiation damage. It accomplishes this using the phase field method coupled to solid mechanics and heat conduction. MARMOT is based on the Multiphysics Object-Oriented Simulation Environment (MOOSE), and much of its basic capability in the areas of the phase field method, mechanics, and heat conduction come directly from MOOSE modules. While some validation of MARMOT has been completed in the areas of fission gas behavior and grain growth, much more validation needs to be conducted. However, new mesoscale data is required in order to complete this validation.

Contents

1 Introduction

1.1 Mechanistic Material Model Development

Fuel and cladding materials undergo significant microstructure evolution during reactor operation. This evolution also changes the fuel properties, directly impacting fuel performance and safety. Traditional fuel performance codes account for these changes in properties using materials models that are empirical fits to experimental data and are correlated to burn-up and temperature. However, these models can only be interpolated within conditions where the tests were conducted and cannot be trusted when the irradiation conditions change, because burn-up is not a unique measure of the history of the fuel material. The Fuels Product Line (FPL) in the Nuclear Energy Advanced Modeling and Simulation (NEAMS) program is developing new materials models that are mechanistic and are based on microstructure rather than burn-up.

In this microstructure based approach, the current state of the microstructure is defined by a set of microstructure variables, e.g. average grain size, grain boundary coverage, and intragranular gas bubble porosity. These variables are evolved with time using mechanistic equations defining the physics of the phenomena. In turn, the material properties are functions of these variables as defined by other mechanistic equations. Thus, this set of variables and mechanistic models describes the interplay of the various microstructure changes that take place within the fuel during irradiation and predicts the resultant degradation in material performance. This approach is illustrated in Fig. 1.1.

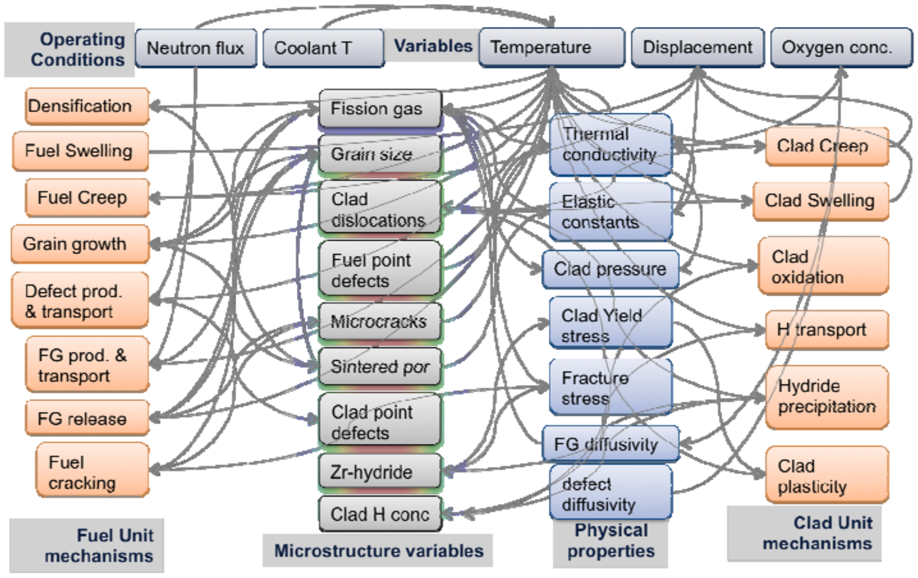


Figure 1.1: Schematic showing the NEAMS FPL plan to develop materials models for fuel performance codes based on microstructure rather than burn-up. Note that the variables and models listed in the figure are not a complete list.

While experimental data informs the development of these new materials models as much as possible, much of the required data is unavailable and would be very difficult and expensive to obtain. Therefore, NEAMS is using a multiscale modeling and simulation approach to supplement the difficult to obtain experimental data. In this approach, first principles calculations are used to investigate basic mechanisms that take place within the bulk of the material, and to quantify material properties defined by these mechanisms. Molecular dynamics simulations are used to investigate phenomena involving lattice defects such as interfaces in the materials and the corresponding properties, as well as dynamic properties such as mobilities. The information regarding the critical mechanisms and material property values is then used to develop mesoscale models that predict the coevolution of the microstructure and the effective macroscale material properties. These data are used to inform the development of the various mechanistic models required to complete the microstructure-based set of materials models. This approach is summarized in Fig. 1.2.

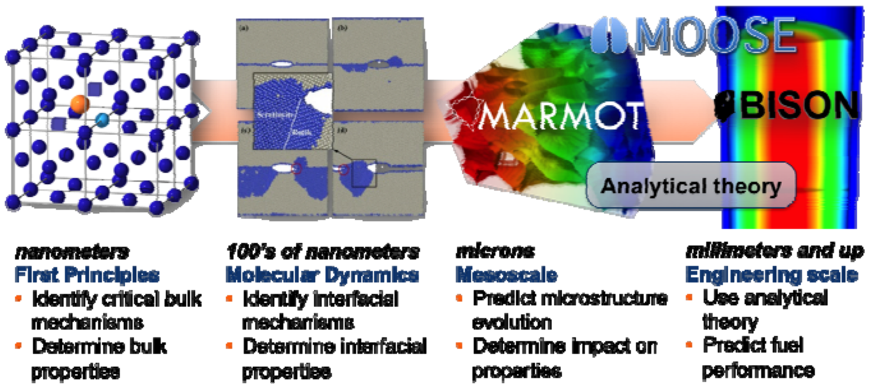


Figure 1.2: Schematic illustrating the multiscale approach being taken by the NEAMS FPL to develop materials models for fuel performance codes based on microstructure rather than burn-up.

1.2 MARMOT Development

While powerful numerical tools for first principles density functional theory and MD simulations are available and have been successfully applied to model nuclear fuel, a robust numerical tool for mesoscale modeling of fuel performance did not exist. Therefore, the NEAMS FPL developed the MARMOT code to predict the coevolution of microstructure and properties in fuel and cladding materials. MARMOT accomplishes this using the phase field method coupled with finite strain mechanics and heat conduction. MARMOT is based on the open source Multiphysics Object-Oriented Simulation Environment (MOOSE) [1] and solves the coupled partial differential equations defining the physics using the finite element method [2]. MARMOT is being developed in order to facilitate the development of improved materials models for fuel performance, but it is also being developed as a powerful tool in and of itself for the simulation of mesoscale fuel performance.

While the goal of the MARMOT tool is focused on investigating fuel and cladding materials, many of the capabilities employed by the code could also be applied to other materials and applications. Therefore, the general capabilities for the phase field method, solid mechanics, and heat conduction are contained in physics modules that are distributed with the MOOSE framework. Any user that downloads MOOSE can instantly use these tools to rapidly develop multiphysics mesoscale simulation tools for a wide range of different applications. These are the phase_field, tensor_mechanics, and heat_conduction modules. MARMOT builds on these modules and adds specific materials and models for fuel and cladding materials, as illustrated in Fig. 1.3.

1.3 Report Overview

In this report, we assess the current capabilities of the MARMOT tool, sumarizing capabilities, verification, and validation. We start in Sections 2 and 3 discussing the phase field and tensor mechanics

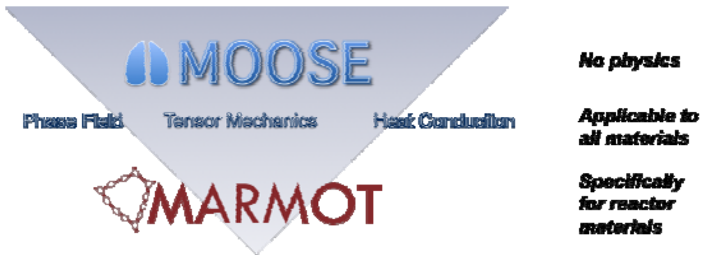


Figure 1.3: Schematic showing the relationship between MOOSE, the MOOSE modules, and MARMOT.

modules, respectively. We then discuss the capabilities of MARMOT in Section 4. Section 5 summarizes the approach used in MARMOT for uncertainty quantification and verification, and discusses MARMOT validation.

2 Phase Field Module

The phase field method has emerged as a powerful and flexible tool for quantitative modeling of the coevolution of microstructure and physical properties at the mesoscale. In the phase field method, the microstructure is described by a system of continuous variables, where the microstructure interfaces have a finite width over which the variables exhibit smooth transitions. The evolution of the microstructure is defined in terms of the free energy of the system, and can be coupled to other physics to provide a complete view of the material behavior. Phase field simulations range from hundreds of nanometers to hundreds of microns and evolve at diffusive time scales [3].

The phase_field module in MOOSE contains the necessary tools to solve the partial differential equations for the phase field method that define the microstructure variable evolution to minimize the overall free energy. The evolution of non-conserved order parameters η_i (representing phase regions and grains) is governed by the Allen-Cahn equation (2.1) and conserved order parameters c_i (representing concentrations) are evolved using the Cahn-Hilliard equation (2.2).

$$\frac{\partial \eta_j}{\partial t} = -L_j \frac{\delta F}{\delta \eta_j} \tag{2.1}$$

$$\frac{\partial c_i}{\partial t} = \nabla \cdot M_i \nabla \frac{\delta F}{\delta c_i} \tag{2.2}$$

2.1 Free Energy based model development

F is the total free energy of the modeled system as a function of the phase field variables, which can be formulated as a volume integral

$$F = \int_\Omega \left[f_{\text{loc}}(c, \eta) + f_{\text{gr}}(\nabla c, \nabla \eta) + E_{\text{d}} \right] dV, \tag{2.3}$$

over multiple free energy density contributions, where Ω is the simulation domain, f_{loc} is the local free energy density, f_{gr} is the gradient energy contribution, and E_{d} is the contribution of other sources of energy. The c, η and $\nabla c, \nabla \eta$ indicate a functional dependence on all conserved and non-conserved order parameters in the domain and their gradients, respectively. Executing the variational derivatives in (2.1) and (2.2) yields terms containing the derivatives of the local free energy density f_{loc} with respect to all order parameters.

The thermodynamic properties of the modeled system are determined by the thermodynamic potential in f_{loc}. The gradient contribution f_{gr} is the reason the phase field model represents interfaces with a diffuse width, and only contributes to the interfacial energy. f_{loc} is therefore the primary input needed to formulate a new phase field material model. In the phase_field module, the residuals for the generic phase field equations are provided as kernels, while the free energy and its derivatives are supplied by material objects. We use a special material interface to provide material properties for all necessary derivatives of the free energy. In general, users use the provided kernels without modification, and only create material objects defining different free energies.

The standard MOOSE solver uses the preconditioned Jacobian-free Newton Krylov method (PJFNK), provided by the PETSc library [4]. To improve the convergence of the solve, the chosen preconditioning matrix should be as close as possible to the actual Jacobian of the problem. Computing the Jacobian matrix entries effectively means providing the derivatives of the residual vector with respect to all non-

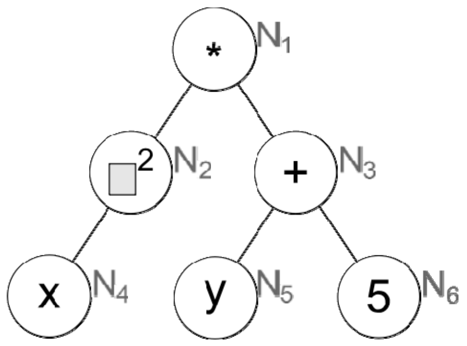


Figure 2.1: Schematic example of the tree representing the mathematical expression $x^2(y+5)$. The nodes N1 and N3 represent the multiplication operator and the sum operator with two arguments each, the internal node N2 represents a square function with one argument, and the leaf nodes N4-N6 represent the variables x, y, and the constant 5.

linear variables of the problem, thus requiring additional derivatives (including cross-derivatives) of the free energy functional.

2.1.1 Parsed functions and automatic differentiation

To create a material object that defines the free energy for a phase field model, code must be written that defines the thermodynamic free energy expression, but also its derivatives. For non-conserved order parameters, the 2nd derivatives are needed and for conserved order parameters, up to the 3rd derivatives could be required. This is complicated even more when a free energy is a function of multiple variables, because all cross derivatives are also required. To avoid having to take and implement all the derivatives, we have implemented automatic differentiation, where the thermodynamic free energy is only required to be entered in the input file.

To allow user defined thermodynamic free energy functions to be supplied via input files, without having to recompile the application code, MOOSE uses the Function Parser library that is included as a third-party plugin in the underlying libMesh finite element library [5]. The Function Parser Library accepts a mathematical function definition given as a plain text string. This string is lexically parsed into an intermediate tree representation and then transformed into a stack machine bytecode. This bytecode can then be executed by the function parser bytecode interpreter module as often as necessary without further transformations.

This intermediate tree representation of the function parser expressions lends itself to algorithmic transformations, such as an automatic differentiation procedure. In this tree structure, leaf nodes can correspond to constants or variables, and internal nodes correspond to mathematical operators and functions with the arguments contained in the respective child nodes or subtrees. The derivative of the leaf nodes yields 0 for all nodes that do not represent the variable the derivative is taken with respect to, and 1 for all nodes that do represent the variable. The derivatives of the internal nodes are constructed recursively according to a set of elementary derivative rules.

Construction of the derivative starts at the root node of the expression tree. For the example expression tree in fig. 2.1, which represents the expression $x^2(y+5)$, the root node holds the multiplication $N_1 = N_2 \cdot N_3$. To obtain the derivative with respect to x of the given expression we need to calculate the derivative of the root node d_xN_1. We set $d_xN_1 = d_xN_2 \cdot N_3 + N_2 \cdot d_xN_3$ according to the product rule. This expression contains derivatives of the nodes N_2 and N_3, which are recursively constructed, until the leaf nodes are reached which have a trivial vanishing 0 derivative in all cases except the d_xN_4, which evaluates to 1. The full derivative expression that is constructed this way is $(2x \cdot 1) \cdot (y+5) + x^2(0+0)$.

The function parser library provides a comprehensive algebraic optimizer that groups, reorders, and transforms the function expression into an equivalent but faster to evaluate form. This optimization stage

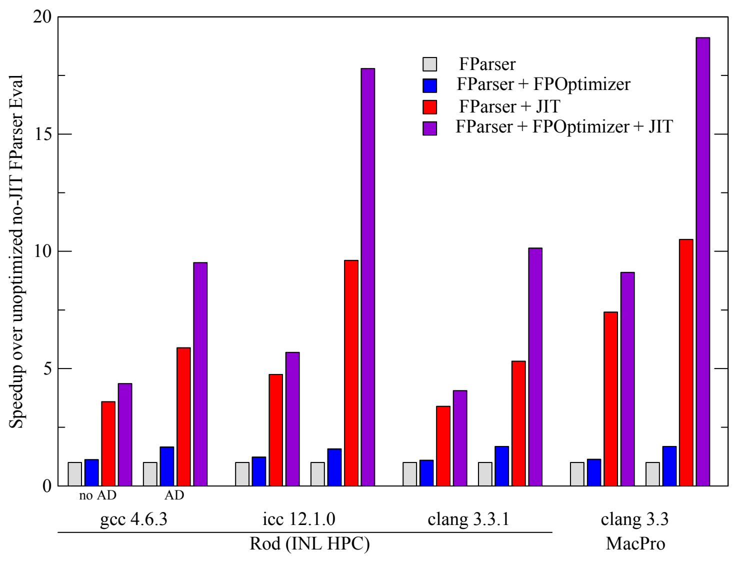


Figure 2.2: Performance comparison for the parsed functions showing the speed-up gained by the just-in-time compilation (JIT) and the algebraic optimization (FPOptimizer). Various compilers were compared on the INL HPC system and on a desktop computer. The combination of JIT and algebraic optimization can yield speed-ups of up to about a factor of twenty.

delivers a speedup of a factor of two, on average. The algebraic simplifications are essential to remove the trivial leaf node derivatives which may lead to evaluation errors such as divisions by zero, that can be avoided by simple term cancellations. In the above example, the simplifications reduce the derivative expression to $2x(y+5)$.

2.1.2 Just-in-time compilation

To further improve the performance of the parsed and runtime interpreted functions, we have developed a just-in-time (JIT) compilation module. At runtime, the generated bytecode sequences are automatically transformed into small C source code files. A compiler is dispatched to compile each function file into a dynamically linkable library, which then is loaded on the fly using the dlopen POSIX system call. If at any stage the JIT compilation fails the function evaluation falls back on the bytecode interpreter, otherwise the generated machine code is called. The time overhead of the additional compilation step is on average of the order of 0.1s per function expression or below, depending on the system the simulation is executed on. This is further mitigated by a caching system. A unique hash is computed from the function bytecode and the compiled functions are stored permanently using the hash as a filename. Recompilation will only occur if the bytecode, and thus the function expression, changes. Trivial function changes, namely the modification of constants, will in most cases not trigger a recompilation.

In Fig. 2.2 the performance of unoptimized interpreted function parser evaluations is compared to combinations of optimized and JIT compiled function evaluations for a variety of compilers under Linux and MacOS. Two function sets were used for the comparison. The left data labeled no AD were obtained using a set of mathematical expressions as they appear in free energy models. The right bar sets labeled AD were obtained by applying the automatic differentiation to the former functions. Two conclusions can be drawn from this comparison, firstly the JIT compilation alone delivers speedups up to a factor of 10. The algebraic optimizer can deliver speedups up to a factor of two on certain functions. The efficacy of the optimizer is largest on the AD function set, which contains lots of trivial terms from the leaf node derivatives, such as entire sub terms that end up being multiplied by 0.

Through this automatic differentiation system we achieve a significant reduction in developer time and remove a source of developer errors that are difficult to track down and debug. The resulting models

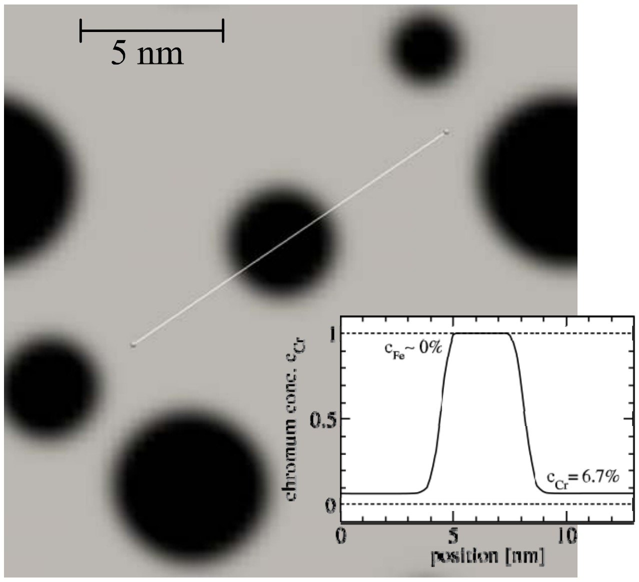


Figure 2.3: Snapshot of a phase field simulation of spinodal decomposition and formation of chromium rich precipitates in an iron chromium alloy obtained using a runtime parsed and automatically differentiated free energy expression. The line scan plotted in the inset shows the precipitate and matrix concentrations.

offer optimal convergence properties due to the complete implementation of the full Jacobian matrix.

2.1.3 Dealing with entropy terms

Free energies that contain a term for the configurational entropy derived from the random solution model will contain terms of the form $c \ln c + (1-c)\ln(1-c)$, where c is a conserved order parameter. As the natural logarithm is only defined for strictly positive numbers, this expression restricts the domain of the free energy to numbers on the interval $(0, 1)$. This poses numerical challenges for systems with equilibrium concentrations close to the edges of this domain.

To improve the convergence behavior, we developed a smoothly extrapolated logarithm surrogate function in which for arguments below a limit ε we evaluate a Taylor expansion of the logarithm around ε instead. For all arguments at and above ε we evaluate the logarithm as before. The function defined in this way extends to negative arguments, is continuous and differentiable to the 2^{nd} order. In the resulting free energy expressions the extension to negative arguments manifests as a free energy penalty which drives the solve back to physical concentrations without incurring numerical errors. Care has to be taken to choose ε small enough to not impact the thermodynamic properties of the simulated system. In particular, large values of ε can change the phase diagram by moving the common tangent points to larger concentrations.

2.1.4 Example

In [6], the free energy surface of an iron chromium binary alloy as a function of chromium concentration and temperature was determined. We entered the full free energy expression from this publication into a MOOSE input file using a parsed function material with automatic differentiation. A running phase field model was obtained in a matter of minutes. Figure 2.3 shows a simulation snapshot obtained using this free energy. The system is in the particle coarsening stage having previously undergone spinodal decomposition. A line scan was performed on the center precipitate and its results are plotted in the inset. In agreement with the published free energy surface and resulting phase diagram we observe practically no solubility of iron in the chromium precipitates, while the chromium solubility in the iron matrix is at around 6.7% at the simulation temperature of 500 K.

Apart from the free energy, the user has to provide a mobility model, which for this example we inferred from experimental chromium tracer diffusivity in iron. For the chosen simulation length scale and mesh size an appropriate interfacial energy parameter κ has to be chosen. An initial choice is trivial, as the order of magnitude of kappa rarely changes at a given length scale, as common excess free energies are of similar magnitude for many alloy systems. Further refinement of the κ value may require a few simulation trial runs, which on one dimensional test systems only take seconds.

2.2 Multiphase models

Multiphase model development requires the construction of a global free energy functional spanning the entire phase space of the system. One common approach is utilizing a linear combination of the free energies F_j of each phase in the system.

$$F(c,\eta) = \left[\sum_j h(\eta_j) F_j(c)\right] + W g(\eta) \tag{2.4}$$

A switching function $h(\eta)$ smoothly changes from 0 to 1 as η goes from 0 to 1. The total weight of all phase free energy contributions at each point in the simulation volume is exactly unity, which translates to the need to enforce the constraint $k(\eta) = 0$ for

$$k(\eta) = \left[\sum_j h(\eta_j)\right] - 1. \tag{2.5}$$

2.2.1 Constraint enforcement

Two phase systems can easily be modeled using a single order parameter η_1 and the explicit constraint $\eta_2 = 1 - \eta_1$, which, for a symmetric switching function with $h(\eta) = 1 - h(1-\eta)$, satisfies the constraint k. For n-phase systems with $n > 2$ it becomes advantageous to use n order parameters. In this case the constraint k is not automatically satisfied and needs to be enforced by other means. In the MOOSE phase field module we offer two methods to enforce the switching function sum constraint, a hard constraint utilizing the Lagrange multiplier technique and a soft constraint through a penalty term added to the free energy.

The hard constraint is applied by introducing a Lagrange multiplier λ as a field variable. With $a_j(\eta, c, v)$ being the weak form (Allen-Cahn) residual for the jth non-conserved order parameter, we need to find (η, λ) satisfying the boundary conditions such that

$$a_j(\eta,c,v) + \int_\Omega \lambda \frac{\partial k}{\partial \eta_j} v\,dx = 0 \tag{2.6}$$

$$\int_\Omega q \frac{\partial(\lambda k)}{\partial \lambda}\,dx = 0 \tag{2.7}$$

holds for every test function v and q. We note that these equations alter the character of the Jacobian matrix of the non-linear problem substantially by introducing a zero block on the Jacobian diagonal. This can complicate the solve substantially. By replacing the constraint k with a modified constraint

$$\bar{k}(\eta,\lambda) = k(\eta) - \frac{\varepsilon}{2}\lambda, \tag{2.8}$$

the Jacobian fill term $\frac{\varepsilon}{2}\lambda$ introduces a small λ dependence in the constraint through an ε (which defaults to 10^{-9}). This results in an on-diagonal Jacobian value of $-\varepsilon$ in the kernel of Eq. (2.7), while it drops out in the residual of Eq. (2.6). This is necessary to force a Jacobian matrix with full rank, avoiding *Zero pivot* PETSc-Errors, and greatly improves convergence. This approach results in a violation of the constraint by about ε, though this violation can be kept small by using an ε as small as possible.

As an alternative we implemented a soft constraint by constructing a penalty contribution f_p to the free energy as

$$f_p = \chi \left(1 - \sum_j h(\eta_j) \right)^2 , \tag{2.9}$$

where χ is a configurable penalty factor.

2.2.2 Kim-Kim-Suzuki model

An additional multiphase model implemented in the phase_field module is the Kim-Kim-Suzuki (KKS) multiphase model [7]. KKS addresses the issue of systems with large phase free energy differences in the interfacial regions. One relevant example is the Xenon gas bubble problem shown in the capabilities chapter in Fig. 4.3. Here the gas solubility in the solid UO_2 matrix is very low, with large free energy penalties for large gas concentrations. In the bubble phase the equilibrium gas concentration is near unity. In the bubble matrix interfacial region both the order parameter as well as the concentration change from the bubble equilibrium values to the matrix equilibrium values over a finite distance due to the soft interface approximation. In that interfacial region the phase free energy of both phases is computed for the intermediate concentration range, which results in large free energy densities from the solid phase contribution. This effectively increases the interfacial free energy of the bubbles significantly to an unphysical value.

The KKS model solves this by introducing the concept of phase concentrations, which are effectively the fractions of the total concentration held in a given phase. In this model the gas concentration is largely shifted to the gas phase to avoid the free energy penalty. Solving for these added variables requires additional differential equations. In the KKS model these are given by mass conservation equations and a constraint that requires the chemical potentials of each component to be pointwise equal across all phases.

The phase_field module currently implements a two-phase version of the KKS model in the form of kernels implementing the phase field equations as well as the KKS constraint equations. The free energy is supplied to those kernels using the derivative material system outlined above.

2.2.3 Example

Figure 2.4 demonstrates an immiscible three phase system consisting of a matrix phase (grey) and two precipitate phases with anisotropic Eigenstrains, where the white phase has a 1% expansion along the x direction and the black phase has a 1% expansion along the y direction. The stiffness of the precipitate phases was chosen larger than the stiffness of the matrix. Periodic boundary conditions are applied for the phase field variables and the displacements. The mesh displacement is plotted with an amplification factor of 20.

The recent developments in the phase_field module constitute a substantial capability expansion which was not singularly driven by immediate program needs, but the long term plan to position the MOOSE phase_field module as a state of the phase field research platform.

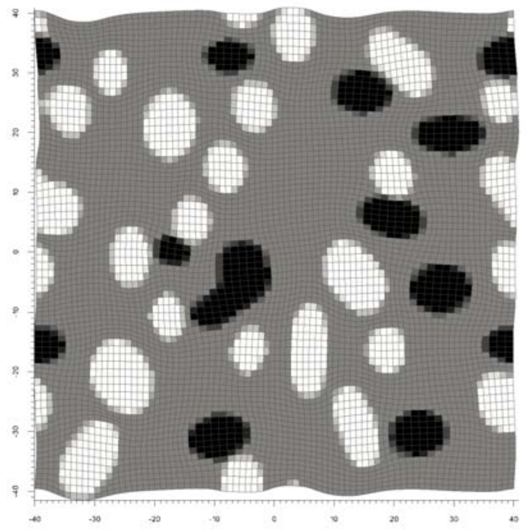


Figure 2.4: Three phase precipitation problem with phase field - mechanics coupling and anisotropic Eigenstrains. Displacements are exaggerated by a factor of 20.

3 Tensor Mechanics Module

The tensor_mechanics module in MOOSE has been developed to solve the weak form of the stress equilibrium equation for the kinematically admissible displacement fields where the second and fourth order tensors are treated as 3×3 and 3×3×3×3 systems, respectively. This representation, instead of the Voigt notation, can be particularly useful if the stress tensor becomes asymmetric due to the presence of micro-moments or if the time integration of asymmetric rank two tensors needs to be incorporated. The full representation of the tensors also facilities easy and error-free code development, with little overhead on the computation time. The tensor_mechanics module follows the MOOSE architecture and currently has the kernels, materials, bcs, auxkernels, actions and userobjects blocks. Several functions for rank two and rank four tensor algebraic operations have also been implemented as utilities (utils). Though the general framework has been developed for fully anisotropic tensors, utility functions have also been implemented for tensors with specific symmetries (isotropic, cubic, etc.) to minimize FLOPS and storage. An overview of the different blocks in the tensor_mechanics module is provided in the following section.

3.1 Blocks in tensor mechanics

3.1.1 Kernels

The kernels block sets up the contribution to the weak form from the stress divergence component Eq. (3.1b) of the equilibrium equation Eq. (3.1a). Capabilities already exist in MOOSE to handle the inertial, body force and traction boundary terms.

$$\rho\frac{D^2\underline{u}}{Dt^2} = \nabla.\underline{\underline{\sigma}} + \underline{b} \tag{3.1a}$$

$$\int_{\Omega} \delta\underline{u}.\nabla\underline{\underline{\sigma}}d\Omega = -\int_{\Omega}\nabla\delta\underline{u}.\underline{\underline{\sigma}}d\Omega + \int_{\Gamma}\delta\underline{u}.\underline{t}d\Gamma = 0 \tag{3.1b}$$

In the above equation, $\underline{u}$ is the displacement vector, ρ is the density, $\underline{b}$ is the body force vector, $\underline{\underline{\sigma}}$ is the Cauchy stress tensor, $\underline{t}$ is the applied traction vector and $\delta\underline{u}$ is the vector of test functions. For finite deformation, an updated Lagrangian formulation is utilized to solve for the displacement variables. In Eq. (3.1b) the stress tensor can be a non-linear function of the displacements and is calculated in the materials block.

3.1.2 Materials

The Material block supports both small strain and finite deformation to obtain the Cauchy stress. In finite deformation, both hypo and hyper elasticity based time integration schemes have been implemented. For hypo elasticity, an incrementally objective stress integration algorithm is used. In the algorithm, the incremental strain is obtained from a polar decomposition of the incremental deformation gradient tensor and the Cauchy stress is updated using a co-rotational formulation. The workability of the incrementally objective stress integration algorithm is demonstrated in sub-section 3.2.1. In the hyper elastic formulation, the logarithmic strain in the undeformed or intermediate configuration is calculated from the deformation gradient. The 2nd Piola-Kirchhoff stress is then evaluated and pushed forward to the current configuration to obtain the Cauchy stress. For both the cases, the B-bar method has been implemented to avoid volumetric locking.

Some of the non-linear materials models currently available in the tensor_mechanics module are hypo-elasticity based J_2 plasticity with both rate independent and dependent versions; multi-surface plasticity, weak plane shear and hyper-elasticity based crystal plasticity. All these models can be extended quite easily to incorporate new flow rules, plastic potentials and state variables by inheriting from the corresponding base classes. Some of the models have also been verified with ABAQUS. A comparison of the necking deformation using rate independent J_2 plasticity between MOOSE and ABAQUS is described in sub-section 3.2.2.

The base classes defining the kinetics and kinematics in the materials block have been implemented separately. This facilitates easy swapping between small and finite deformation. Also different strain measures can be used for the same consistent stress measure and integration scheme. Stress-free strains have also been incorporated as a general feature in the materials block. The stress-free strain can be effectively used to apply misfit strains in phase-field simulations and thermal strains in coupled thermo-mechanics problems.

3.1.3 Auxkernels

The specific operations implemented in this block are for generating output of components of the tensors and scalar measures such as Von-mises stress, equivalent strain, norm, etc. of rank two tensors.

3.1.4 UserObjects

The UserObjects block currently consists of a number of plasticity routines and hardening laws that can be used interchangeably in the input file. The UserObjects format of defining multi-surface plasticity also facilitates the simultaneous use of various associative and non-associative flow rules with little or no extra coding.

3.2 Verification

3.2.1 Example of the Stress Integration Algorithm

In this example, a square box is stretched along the x-direction and then rotated by 90^o along the z-axis as shown in Fig. 3.1(a). The normal and shear components of the stress tensor are plotted on the Mohr's circle in Fig. 3.1(b). As can be observed from Fig. 3.1(b), the objectivity of the stress tensor is maintained with the superposed rigid body rotation.

3.2.2 Verification of the J_2 plasticity material model

The necking of a 3D unit cube is simulated and compared with ABAQUS to demonstrate the workability of the "StressDivergence" kernel and rate independent J_2 plasticity material class. An offset of 0.05 mm is provided to introduce necking deformation and is shown in Fig. 3.2(a). Comparisons of the load-displacement curves, and the contours of equivalent plastic strain and stress are shown in Fig. 3.2(b) and 3.3, respectively. As can be observed from the figures, the results agree very well. The slight mismatch in the contours is due to the different averaging schemes used for plotting between ABAQUS and MOOSE.

3.3 Examples

3.3.1 Crystal Plasticity based material model

An example of heterogeneous deformation in a representative volume element (RVE) of polycrystalline microstructure obtained using the crystal plasticity material class in tensor_mechanics is demonstrated here. The initial microstructure is generated using an utils function that performs Voronoi tessellation

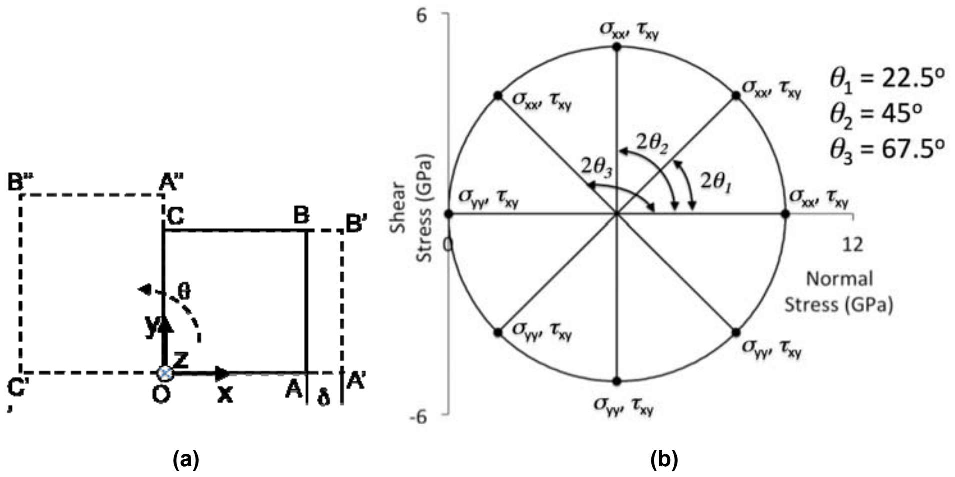


Figure 3.1: (a) Geometry of the large deformation finite element problem solved in MOOSE to test incremental objectivity. OABC is the original geometry stretched by δ along x to OA'B'C and then rotated by 90^o along z to the final configuration OA"B"C". (b) Stress components on the Mohr circle at various rotation angles.

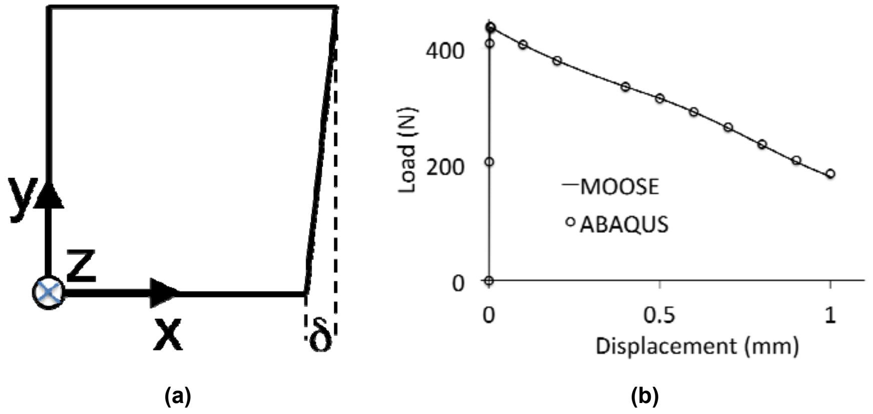


Figure 3.2: (a) Geometry of the necking problem solved in ABAQUS and MOOSE to test the accuracy of the rate independent J_2 plasticity model. δ = 0.05 mm is the offset value to introduce necking. (b) Comparison of the load-displacement evolution.

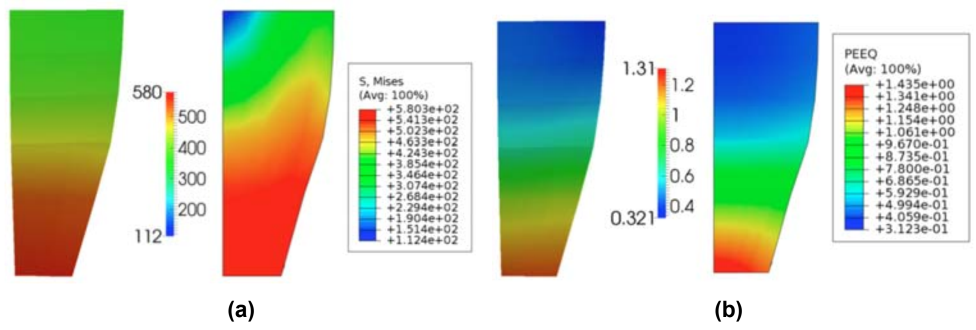
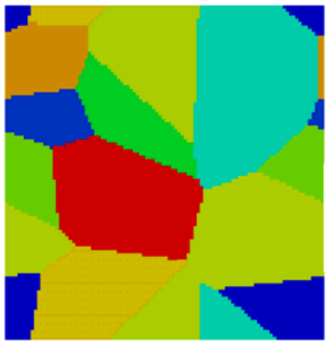
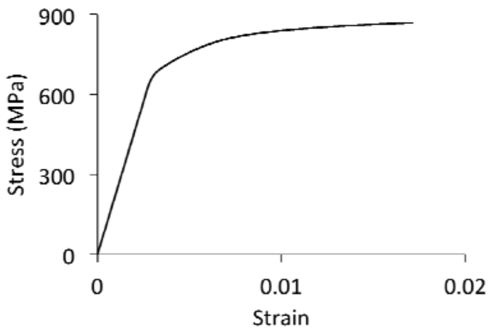


Figure 3.3: Comparison of (a) Von-Mises stress and (b) Equivalent plastic strain distribution in the necked specimen.

of rectangular 2D and 3D geometries, and is shown in Fig. 3.4(a). The RVE is subjected to uniaxial tensile strain and the engineering stress-strain evolution is shown in Fig. 3.4(b). The distribution of stress and plastic deformation gradient components along the loading direction in the microstructure at the final timestep is shown in Fig. 3.5.

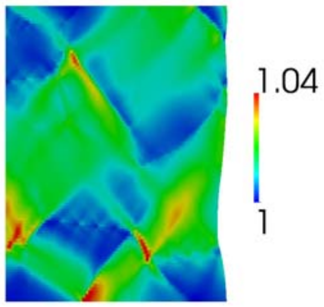


(a)

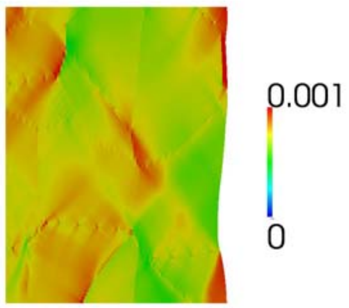


(b)

Figure 3.4: (a) Geometry of the 2D polycrystalline microstructure. (b) Engineering stress-strain evolution under uniaxial tension.



(a)



(b)

Figure 3.5: Distribution of, (a) Plastic Deformation and (b) Stress component along the loading direction, in the microstructure at the last time step. The units of stress are in 10^6 MPa.

3.3.2 Coupling of tensor_mechanics and PhaseField for modeling fracture

A phase-field fracture model has been developed under MARMOT to capture the complicated microstructure scale crack propagation and is now available in the phase_field module in MOOSE. The model couples the Allen-Cahn and stress divergence equations to evolve a damage variable (c) non-locally that degrades the material and causes failure. The model is described in detail in [8]. The model has been assessed by simulating typical fracture mechanics specimens, such as single edge notch (SENT) specimen (Fig. 3.6). In the rate independent sharp crack limit, the model performance has been compared with linear elastic fracture mechanics results. Currently the model is being extended to elasto-plastic fracture as well.

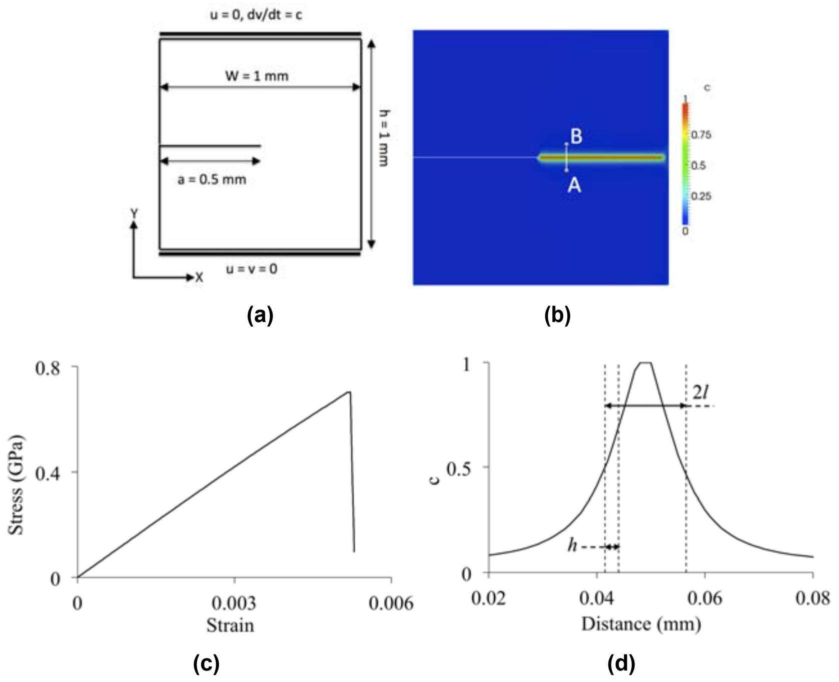


Figure 3.6: Simulation of brittle Mode I crack propagation in single edge notch (SENT) specimen using phase-field based fracture model: (a) Specimen geometry and boundary conditions. The displacements *u* and *v* are along the directions *X* and *Y* respectively; (b) Final configuration; (c) Stress-Strain along the loading direction; (d) Variation of damage variable, *c*, along line A-B shown in (b). Taken from [8].

4 MARMOT Capabilities

MARMOT builds on the capabilities from the phase_field and tensor_mechanics modules, along with the basic heat conduction capability from the heat conduction module, to model the coevolution of microstructure and properties in fuel and cladding materials. While the modules are open source and openly accessible using Github, MARMOT access is not open. There is no license or cost associated with MARMOT access; users simply have to contact a member of the MARMOT development team and request access.

The capability development of the modules has been general, to supply general tools to a broad user-base, but the capability development in MARMOT has been driven by the development of the microstructure-based materials models for fuel performance tools. Thus, the development has been focused on high impact areas of light water reactor fuel. Recently, development has begun on models of zirconium cladding as well as alternative fuel types other than UO_2. In this section we summarize the work that has been conducted using MARMOT and give the references for interested readers to get more detail. We begin with the microstructure evolution models and then summarize how we predict the changes in material properties.

4.1 Microstructure Evolution

Many microstructural changes take place with fuel and cladding materials during reactor operation, including creep, grain growth, fracture, and fission gas migration and release. For the fuel, the changes that most significantly impact the fuel performance are fission gas behavior and fracture. However, the current grain size impacts both fracture and fission gas. Thus, we have focused our initial MARMOT development efforts on modeling fission gas behavior, cracking, and grain growth in UO_2.

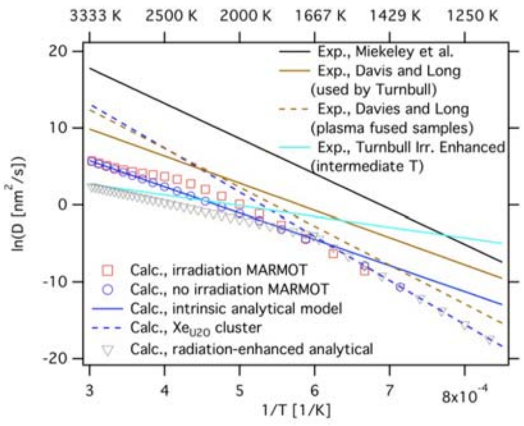
4.1.1 Fission Gas Behavior in UO_2

Gaseous fission products are produced as a product of the fissioning taking place within the fuel. There are various constituents to the fission gas, though the largest concentration is Xenon (Xe). The gas atoms migrate through the fuel until they are trapped in small intragranular bubbles or in larger grain boundary (GB) bubbles. The intragranular bubbles are kept small (with radii on the order of 5 nm) due to resolution caused by fission fragments, but the GB bubbles get to be much larger. These large GB bubbles contribute significantly to swelling of the material, and also interconnect to form open channels for fission gas release. Completed work in MARMOT has focused on fission gas transport and segregation to GBs, and general bubble growth behavior. Current development is coupling these two areas to model the GB bubble growth and interconnection in polycrystalline materials.

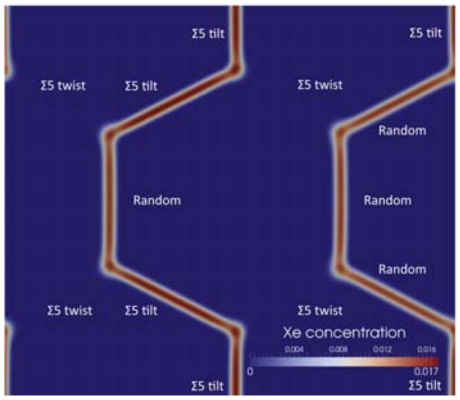
Xenon Transport and Grain Boundary Segregation

To accurately predict the swelling and fission gas release in UO_2 fuel, we must first understand the diffusion of fission gas atoms through grains and interaction with grain boundaries. Based on the mechanisms established from earlier density functional theory (DFT) and empirical potential calculations [9], diffusion models for Xe, uranium (U) vacancies and U interstitials in UO_2 have been derived for both intrinsic (no irradiation) and irradiation conditions (for more detail on this work, please see the original paper [10]). These atomistically determined properties were used to develop an in-depth model of Xe diffusion in MARMOT, accounting for various defect clusters and the reactions between clusters. The

MARMOT simulations were used to predict the effective diffusivity under different irradiation conditions and they were compared to experimental results, as shown in Fig. 4.1(a)



(a)



(b)

Figure 4.1: Simulation results using MARMOT to model Xe transport in UO_2. The diffusivity under irradiation conditions compared to the intrinsic diffusivity (Calc., MARMOT), the corresponding analytical expressions and experimental data are shown in (a). Xe distribution for a microstructure composed of random, $\Sigma5$ tilt and $\Sigma5$ twist grain boundaries is shown in (b). The domain size is 75×65 nm and the initial state was a homogeneous distribution of Xe ($c_{Xe} = 0.001$). Taken from [10].

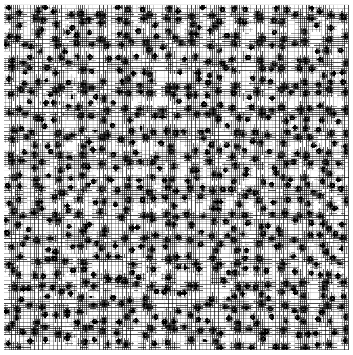
Segregation of Xe to grain boundaries was described by combining the bulk diffusion model with a model for the interaction between Xe atoms and three different grain boundaries in UO_2 ($\Sigma5$ tilt, $\Sigma5$ twist and a high angle random boundary), as derived from atomistic calculations. The model does not attempt to capture nucleation or growth of fission gas bubbles at the grain boundaries, though a model that does model bubbles nucleation and growth is under development. The point defect and Xe diffusion and segregation models were implemented in MARMOT to simulate Xe redistribution for a few simple microstructures, as shown in Fig. 4.1(b).

Intragranular Bubble Growth

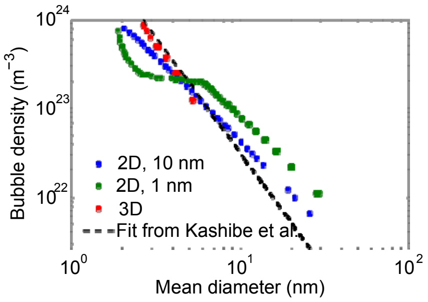
Fission gas bubble growth is the dominate driver for fuel swelling. Thus, it is essential that MARMOT be able to accurately model this phenomena. To validate the MARMOT bubble growth model, we used it to simulate the post-irradiation annealing of UO_2 described in the experimental work by Kashibe et al., 1993 [11]. The simulations were carried out in 2-D and 3-D using MARMOT. We employed a simple model in which only the Xe gas atoms are modeled, and all other defects are assumed to exist in sufficient quantities and to migrate fast enough that the Xe diffusion dominates the growth (for more details, see the original proceedings paper [12]). Mesh adaptivity was employed to reduce the computational expense, (see Fig. 4.2(a) for an image of the initial adapted mesh). The 2-D results compared fairly well with the experiments, in spite of the assumptions made in the model. The 3-D results compared even more favorably to experiments, indicating that diffusion in all three directions must be considered to accurately represent the bubble growth, as shown in Fig. 4.2(b). These results validate the model, but also indicate the importance of 3-D simulations.

Advanced Fission Gas Model

While the simple bubble growth model was sufficient to simulation post-irradiation annealing, fission gas bubble behavior under irradiation conditions is more complicated. Specifically, the bubble pressure has a



(a)



(b)

Figure 4.2: MARMOT simulations of post irradiation annealing, where (a) shows the initial adapted mesh of a 2-D simulation and (b) shows the predicted bubble density versus mean bubble diameter and the experimental fit. The 3-D results shown excellent agreement with the experiments. Taken from [12].

large impact on the gas behavior and on swelling. Therefore, we are currently implementing an advanced fission gas model in MARMOT taken from [13]. This model currently predicts the evolution of Uranium vacancies and gas atoms, though Uranium interstitials will also be added. The model is implemented using the KKS approach, as discussed in Section 2. Because the Xe and vacancies concentrations are tracked separately, we can add a pressure within the bubbles that is a function of the two concentrations. Figure 4.3 shows a simple demonstration of the model currently under development. This model will eventually be combined with the fission gas transport and segregation work from [10] to predict the growth and interconnection of fission gas bubbles in polycrystalline UO_2.

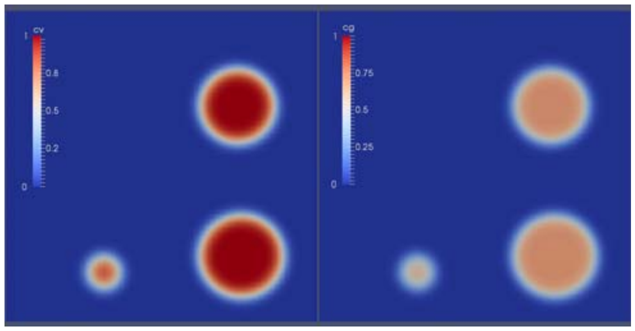


Figure 4.3: Example of bubble formation modeled with the KKS UO_2 fission gas model, where the vacancy concentration is shown on the left and the Xe concentration is shown on the right.

4.1.2 Grain Growth in UO_2

GBs migrate to reduce the overall free energy of the system. This reduction could be due to various sources of energy, including reduction in grain boundary energy, elastic energy, or defect energy. The velocity at which the GBs migrate is a function of the magnitude of the respective driving forces and the GB mobility. The GB mobility is temperature dependent, so the higher the temperature, the faster the GBs migrate.

In UO_2 fuel during LWR operation, a large temperature gradient forms across the radius of the fuel pellet, with the center temperature on the order of 1800 K and the edge temperature 800 K. These high temperatures and temperature gradients result in grain growth in the hotter portions of the fuel and thus a gradient in the average grain size across the pellet radius. The average grain size of the fuel directly impacts fission gas release, swelling, creep, thermal conductivity, and cracking within the fuel. Thus, it is very important to understand the grain growth within the fuel, which requires a detailed understanding of the various driving forces. In this section, we will summarize the work we have done with MARMOT investigating the importance of the temperature gradient driving force and on pinning of the GBs due to GB bubbles with a distribution of bubble radii.

Temperature Gradient Driving Force

Grain boundaries (GBs) are driven to migrate up a temperature gradient. While this driving force is often neglected because temperature gradients are small in most applications, the large temperature gradients that form in UO_2 fuel may make this driving force significant. We used MARMOT simulations to investigate the impact of temperature gradients on isotropic grain growth in LWR fuel pellets (Details about these simulations are available in [14]). GB motion in 2-D UO_2 polycrystals was predicted under increasing temperature gradients, as shown in Fig. 4.4. We found that the temperature gradient does not significantly impact the average grain growth behavior, because the curvature driving force is dominant (see Fig. 4.5). However, it does cause significant local migration of the individual grains. In addition, the temperature dependence of the GB mobility results in larger grains in the hot portion of the polycrystal.

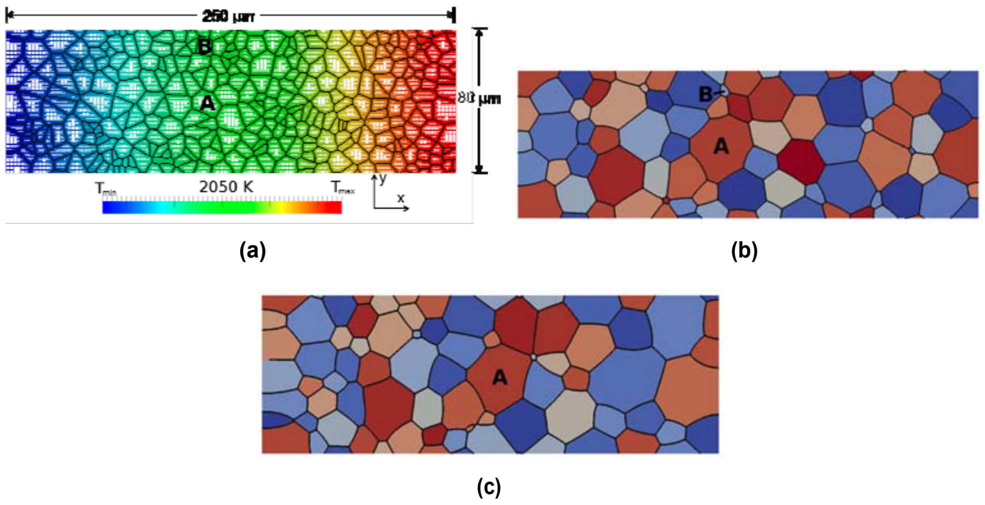


Figure 4.4: Polycrystal simulation domain, where the initial grain configuration showing the adapted mesh is shown in (a). The constant temperature gradient is also shown. Note that the values for T_{min} and T_{max} vary for the different gradients, ranging from $T_{min} = T_{max} = 2050$ K for no gradient to $T_{min} = 1950$ K and $T_{max} = 2150$ K for $\nabla T = 0.8$ K/μm. The centroid position with time is recorded for the grains labeled A and B. The final grain configuration after 2000 minutes is shown in (b) with no temperature gradient and in (c) with a gradient of 0.8 K/μm, where grain B has disappeared. Note that with the temperature gradient, the grains are smaller on the cold side and larger on the hot side. Taken from [14].

Pinning by Bubbles with a Distribution of Bubble Radii

While the mobility of a GB is an intrinsic property, the actual GB migration is severely influenced by impurities and defects [15, 16]. For example, particles and pores resist GB motion, slowing and even halting grain growth [17, 18]. The earliest treatment of GB pinning was by Zener [19]. Zener determined

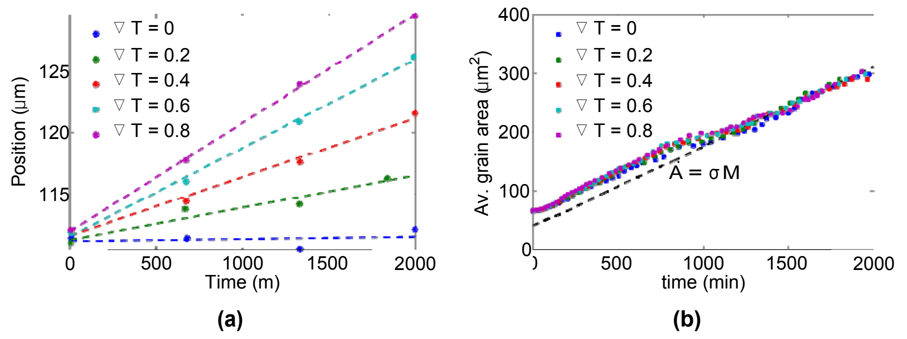


Figure 4.5: Plots of the impact of the temperature gradient on the grain growth behavior, where (a) shows the impact of the temperature gradient on the centroid position of grain A and (b) shows the average grain size for various temperature gradients. Note that the average grain size is independent of the temperature gradient and compares well with the analytical model for curvature driven grain growth. Taken from [14].

the resistive pressure for a given volume fraction of spherical, incoherent, immobile, and rigid particles. Zener was able to derive a simple expression for the pinning pressure by making several simplifying assumptions; he assumed that the GBs were flat, the particles were randomly distributed, only particles within one radius influence the GBs, and that all particles exert the maximum pinning force throughout the interaction. We have expanded on Zener's work to consider fission gas bubbles aligned on a GB and to also account for a distribution of particle sizes (see [20] for the details of this work). We began by developing a phase field model that describes GB and pore interactions, and verified it by comparing to molecular dynamics simulations (see Section 5 for more detail). We then developed an analytical pinning model that is a function of the GB fractional coverage, the percentage of the GB covered by gas bubbles. The model also considers the impact of the bubble size distribution, in terms of the mean and standard deviation of the bubble radius. The analytical model was verified by comparing to MARMOT simulation results and those from a simple Monte Carlo model (see Fig. 4.6). A significant finding from the model is that the *mean* value of the resistive pressure decreases with increasing *standard deviation* of the bubble radius.

Grain Growth Model Validation

In order to validate the grain growth model, it is not sufficient to only compare the average grain size to experimental data, but it is also important to compare the topology. In order to validate the grain growth model for UO_2, we need access to data that characterizes the microstructure before and after annealing, to allow direct comparison between the simulation results and the data. In addition, the best comparison would be with 3-D data. However, most approaches for characterizing 3-D microstructures destroy the samples. We have been working with Don Brown from Los Alamos National Laboratory (LANL) to use UO_2 data collected using High Energy X-Ray Diffraction Microscopy (HEDM) for validating the grain growth model in MARMOT. Don and collaborators used HEDM to collect 3-D data on a sample of UO_2 and then annealed the same sample. However, they have not completed the processing of the data from the annealed sample. We have reconstructed the sample in MARMOT and simulated the annealing process. Once the data from LANL is available, we will begin the comparison to validate the predictions. See Fig. 4.7 for an example of the microstructure before and after annealing, as predicted by MARMOT.

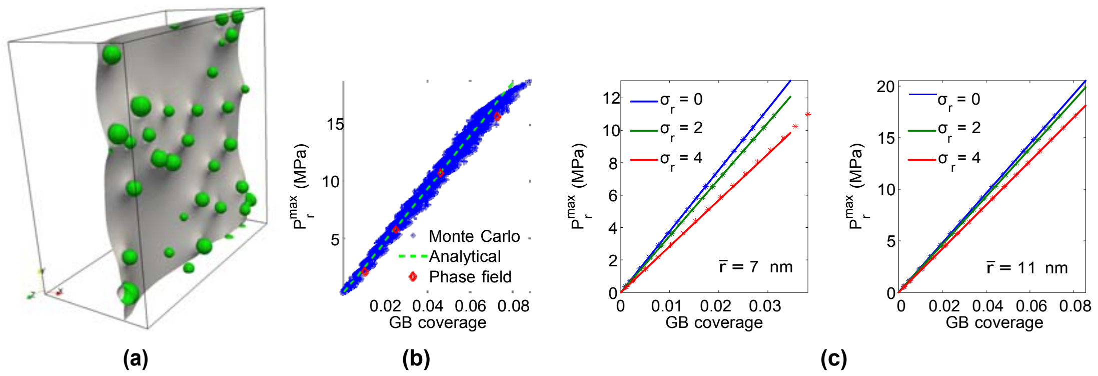


Figure 4.6: Verification of our resistive pressure model that is a function of the GB fractional coverage and considers bubble size distributions. An example of a phase field simulation with bubble radii following a normal distribution with a mean radius of $\bar{r} = 11$ nm and standard deviation of $\sigma_r = 2$ nm is shown in (a), a comparison of the analytical model, Monte Carlo results and the phase field simulations for $\bar{r} = 11$ nm and $\sigma_r = 2$ nm is shown in (b), and the analytical model shows excellent agreement with the phase field simulations and the Monte Carlo model. The variation of the resistive pressure with standard deviation for two mean radii is shown in (c), where the line shows the analytical model and the symbols show the mean values from the Monte Carlo simulations. Taken from [20].

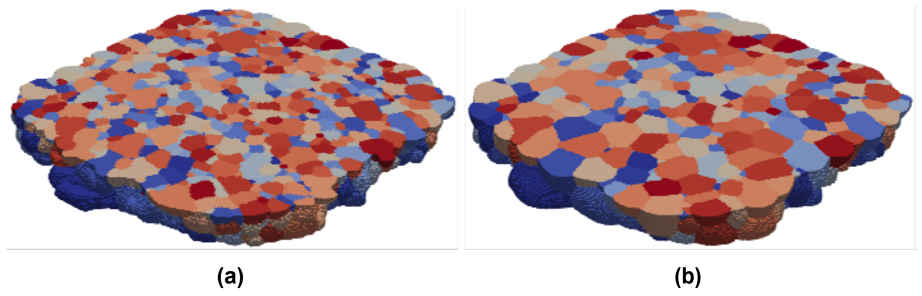


Figure 4.7: MARMOT grain growth simulation of a reconstructed microstructure taken from HEDM data of UO_2, where the initial microstructure is shown in (a) and the final, after 200 minutes of annealing at 2000°C, is shown in (b). 600 of the original 1620 grains are present after the annealing.

4.1.3 Fuel Cracking

UO_2 cracks during reactor operation, because it is a brittle material. It initially cracks radially during reactor start up, but it also cracks during reactor operation and shutdown. These cracks significantly impact the fuel behavior, reducing the stress, providing avenues for fission gas release, and reducing the thermal conductivity (primarily circumferential cracks). Cracking within the fuel primarily occurs along GBs, thus as the GBs get covered with fission gas bubbles, their fracture stress will be significantly reduced. To predict this reduction in fracture stress, we have developed a phase field fracture model that couples the phase field equations with solid mechanics, as discussed in Section 3. The simulation used a different fracture toughness for the grain interiors and the GB, with the values taken from MD simulations [21]. Increasing numbers of bubbles were placed on the grain boundaries and the cracking behavior was predicted using MARMOT. The crack propagation in a typical UO_2 RVE subjected to a strain controlled loading with ratio $\varepsilon_2/\varepsilon_1=0.7$ is shown in Fig. 4.8. The porosity dependent stress-strain evolution and a comparison with experiment is shown in Fig. 4.9. The deviation from the experiments are likely due to difference between the perfect crystals modeled by MD and the physical materials with defects and impurities in the experiment. In addition, it could be due to our conducting the simulations in 2-D. 3-D simulations will be conducted in the future.

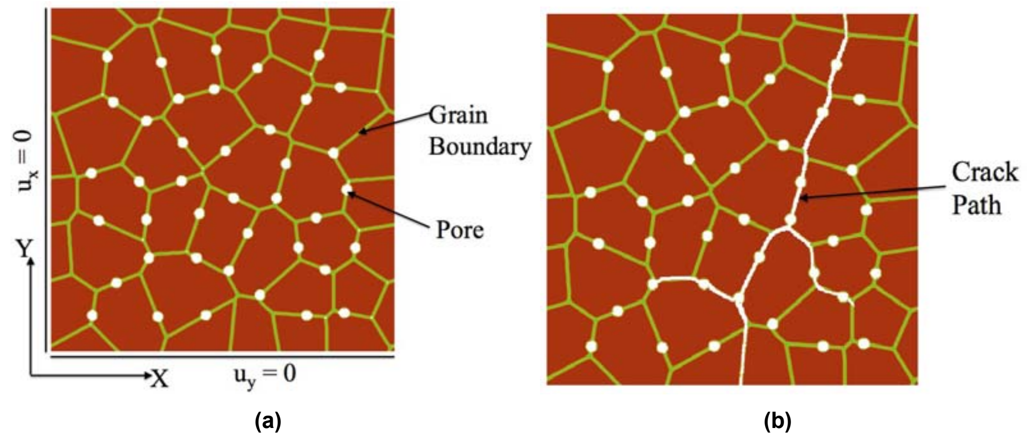


Figure 4.8: (a) Geometry of the 2-D representative volume element. (b) Phase-field based brittle crack propagation under strain controlled loading with $\varepsilon_2/\varepsilon_1=0.7$.

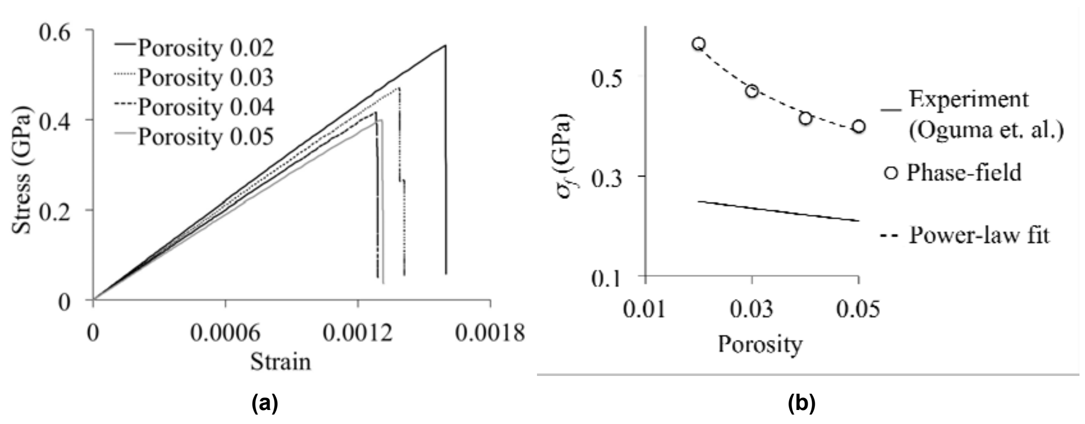


Figure 4.9: (a) Stress-strain evolution obtained from phase-field based fracture simulations of RVEs with varying porosity. (b) Variation of fracture strength with porosity.

4.1.4 Other Fuel Types

While the NEAMS program has been focused on modeling of UO_2, we are beginning to apply MARMOT to model other fuel materials. As part of the Accident Tolerant Fuel High Impact Problem, we will begin developing models of Uranium Silicide (U-Si) phases in MARMOT. However, our initial development efforts are focused on developing an empirical potential for U-Si and has not involved MARMOT development up to now.

Other DOE NE programs are also beginning to fund development work using MARMOT to model metallic fuels. The Advanced Fuel Cycle program is funding some development work on Uranium Zirconium (U-Zr), focusing on modeling constituent redistribution in a temperature gradient. This work will be validated by directly comparing to diffusion couple data that has been collected by the campaign. The NNSA RERTR program has also begun to fund the development of MARMOT to model Uranium Molybdenum (U-Mo). Initial development efforts have been focusing on implementing a simple model of fission gas swelling in U-Mo, as shown in Fig. 4.10. This model only tracks the fission gas and can't account for bubble pressure. Future work will develop an improved model of fission gas behavior using the KKS model and will investigate the impact of phase separation on swelling.

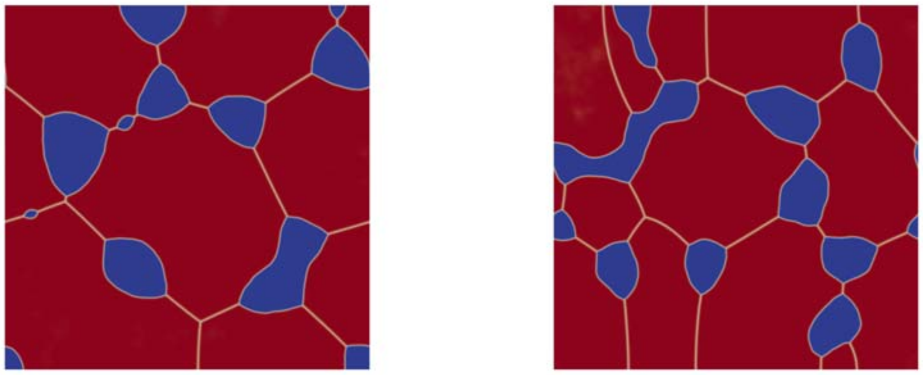


Figure 4.10: Fission gas bubble formation in U-Mo using a simple model only tracking fission gas irradiated to a fission density of 17×10^{27} fissions/m^3. Simulation results with an average grain size of 2.4 μm are shown in (a) and 1.2 μm in (b).

4.1.5 Cladding Materials

Under the NEAMS program, the MARMOT code has not been funded to develop models of the LWR zircaloy cladding. However, using internal INL funding, we are implementing a multiphase model that predicts the formation of the δ-hydride phase in the cladding. This model accounts for the lattice mismatch between the α-Zr phase and the δ-hydride phase using a stress free strain coupled to the phase field variables. The model currently predicts the phase growth but needs to be coupled to a robust nucleation model to account for the hydride nucleation. Simulations with this model will investigate the impact of applied stress on the hydride orientation under irradiation.

4.2 Effective Material Properties

As the microstructure evolves, the effective material properties of the material also change. For example, irradiated fuel has a much lower thermal conductivity than fresh fuel due to the formation of gas bubbles, solid fission products, and point defects. MARMOT can be used to quantify the effective properties of a given microstructure by locally changing the properties as a function of the microstructure. Then, the local properties are homogenized across the structure to determine the effective results. This approach can be used to determine effective thermal conductivities, elastic constants, diffusivity, or even fracture strength. Here, we summarize the approaches used for homogenization in MARMOT, and give several examples of how MARMOT has been applied to determine the effective thermal conductivity of various microstructures.

4.2.1 Approach

It requires more than just a spatial average to accurately homogenize local properties to obtain the effective property across the domain. One approach is to simulate a given physical phenomenon and then to determine the effective properties from the material response. For example, to determine the effective thermal conductivity, the steady state heat conduction equation can be solved across the microstructure with an applied heat flux on one side and a fixed temperature on the other. However, an alternative approach is to use the asymptotic expansion homogenization (AEH) technique that has shown to be less sensitive to boundary conditions and can be used for a large range of different properties. AEH has been implemented in MARMOT to determine effective thermal conductivities and elastic constants, as discussed in detail in [22]. Fig. 4.11 demonstrates the approach for effective thermal conductivity, showing more consistent results when using AEH than directly applying a temperature gradient.

4.2.2 Fuel Thermal Conductivity

The fuel thermal conductivity is one of the most dominant properties influencing fuel performance. Thus, it is critical to understand how the microstructure within the fuel impacts the thermal conductivity. For this reason, we have conducted a significant amount of research looking at the impact of microstructure on thermal conductivity.

Impact of GB Bubbles on Thermal Conductivity

GB bubbles were shown to have a larger impact on the thermal conductivity than randomly distributed bubbles [23]. Thus, we have conducted a series of simulations to quantify their impact as a function of the percentage of the GB covered by fission gas. In [24] we ran 2-D simulations of various grain structures with increasing concentrations of fission gas bubbles on the GB and different GB thermal resistances R_k, as shown in Fig. 4.12. 3-D bicrystal simulations were also conducted [25] to develop a robust set of data to quantify the impact of GB bubbles. These data are playing an essential role to inform the development of an analytical model of the impact of GB bubbles on thermal conductivity.

Thermal Conductivity in Reconstructed Irradiated Microstructures

The current understanding of fuel microstructure has been limited by the difficulty in studying the structure and chemistry of irradiated fuel samples at the mesoscale. We took advantage of recent advances in experimental capabilities to characterize the microstructure of irradiated mixed oxide (MOX) fuel in 3-D taken from two radial positions in the fuel pellet. We reconstructed these microstructures using MARMOT and calculated the impact of microstructure heterogeneities on the effective thermal conductivity using mesoscale heat conduction simulations, as shown in Fig. 4.13. See [26] for more detail. The predicted thermal conductivities of both samples were higher than the bulk MOX thermal conductivity due to the formation of metallic precipitates and because we do not currently consider phonon scattering due to defects smaller than the experimental resolution. We also used the results to investigate the

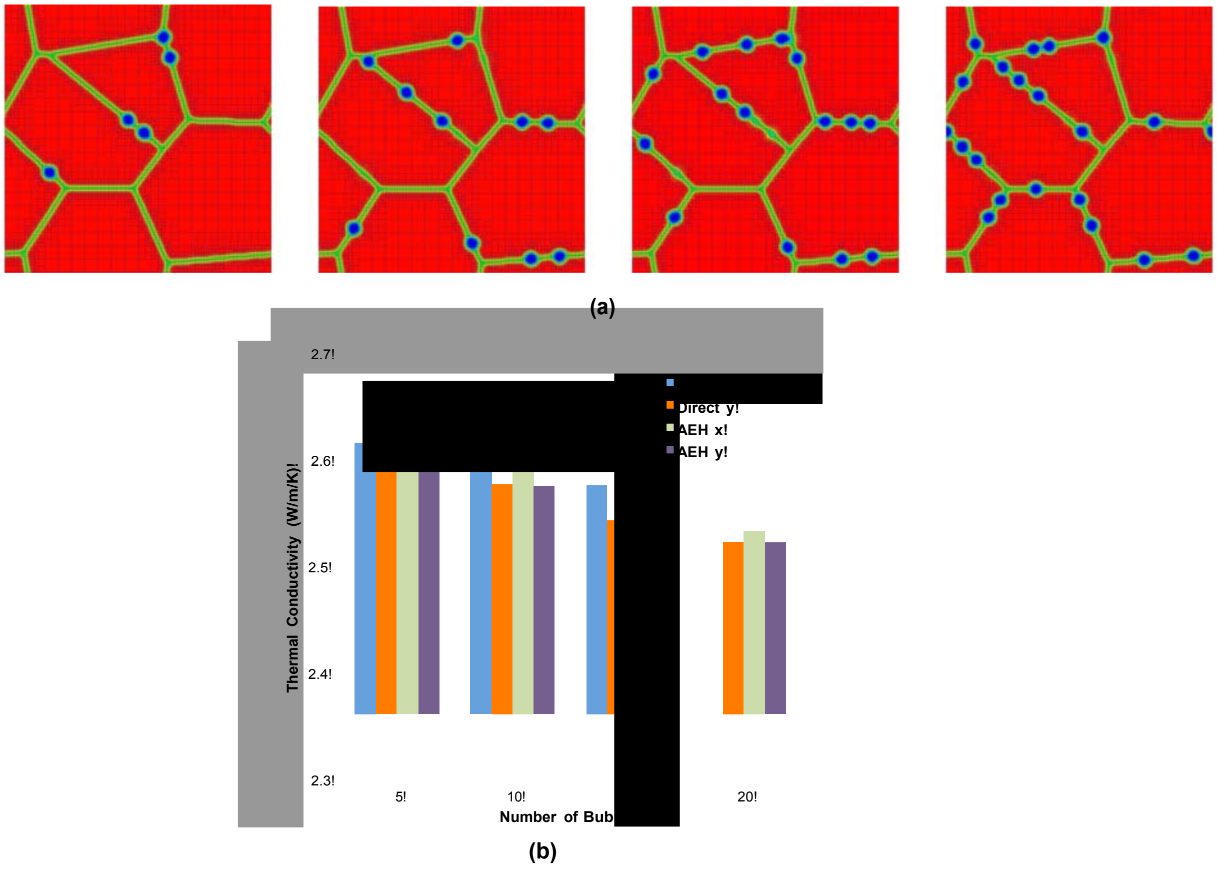


Figure 4.11: Homogenized thermal conductivity for the direct and AEH methods in the *x* and *y* directions, where (a) shows the 2-D microstructures and (b) gives values of the homogenized thermal conductivity using AEH and the direct approach. The direct method gives a significantly lower thermal conductivity in the *x* direction for the case of 20 bubbles due to a bubble on the domain boundary. Taken from [22].

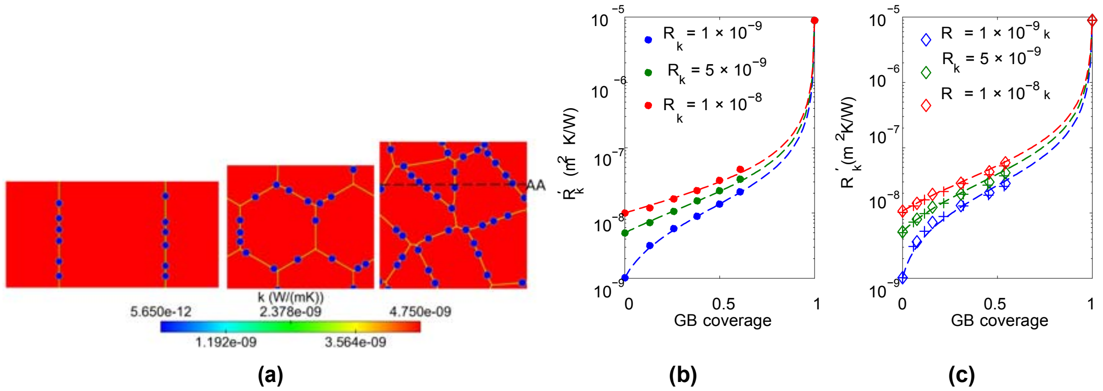


Figure 4.12: Mesoscale heat conduction simulations, where (a) shows example microstructures for the bicrystal, four grain hexagonal polycrystal and the seven grain polycrystal shaded by thermal conductivity and (b) and (c) are the effective GB thermal resistance vs GB coverage for bicyrstals and polycrystals, respectively. Note that the effect of the GB coverage on the thermal resistance is significantly smaller for the bicrystal simulations than for the two polycrystal simulations. The mechanistic model predictions are shown (dashed lines) with the corresponding mesoscale simulation data for the bicrystal and the two polycrystals. Taken from [24].

accuracy of simple thermal conductivity approximations and equations to convert 2-D thermal conductivities to 3-D. It was found that these approximations struggle to predict the complex thermal transport interactions between metal precipitates and voids (see Table 4.1).

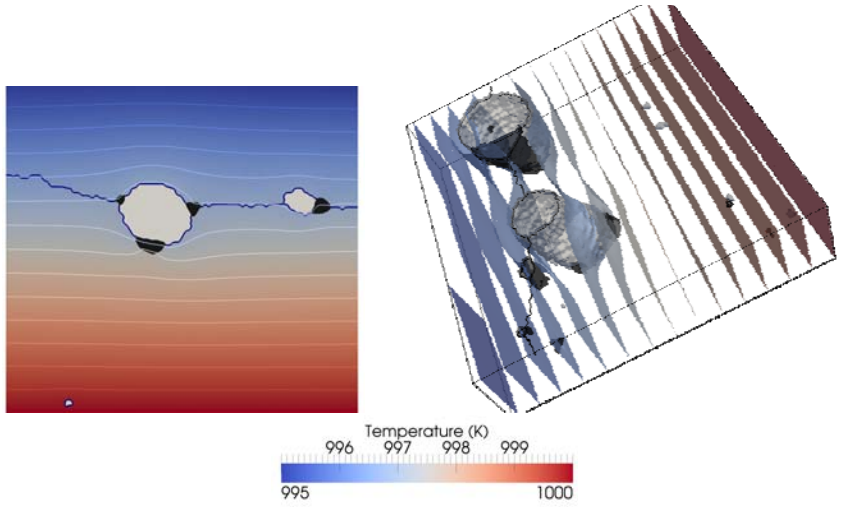


Figure 4.13: The calculated temperature profile across the Sample 1 microstructure, showing the 2-D (left) and 3-D (right) results. The large metallic precipitates are shown in grey, the voids in black, and the temperature contours are shaded by the temperature value.

Table 4.1: Effective thermal conductivities calculated across the reconstructed 2-D and 3-D microstructures, where k^x_{eff} and k^y_{eff} are the thermal conductivities in the *x*- and *y*-directions, respectively. Approximations assuming parallel and series arrangement are also shown, along with approximate 3-D values converted from the 2-D values using expressions from [27]. Note that the parallel and series approximations are isotropic and so give the same values in the *x*- and *y*-directions.

Sample 1	k^x_{eff} (W/mK)	k^y_{eff} (W/mK)
2-D simulation	2.57	2.55
3-D simulation	2.64	2.67
Parallel Appr.	3.68	3.68
Series Appr.	1.59	1.59
Converted 3-D	2.61	2.56
Sample 2		
2-D simulation	2.49	2.49
3-D simulation	2.48	2.48
Parallel Appr.	2.60	2.60
Series Appr.	2.47	2.47
Converted 3-D	2.44	2.44

5 Quality Assurance, Verification, and Validation

Critical aspects of developing MARMOT as a robust tool for mesoscale fuel performance modeling are quality assurance (QA), verification, and validation. Excellent quality assurance practices are necessary to ensure that the MARMOT tool produces consistent high quality results even as new features and capabilities are added to the tool. It also essential that rigorous verification takes place, ensuring that the equations are correctly implemented and are being accurately solved. Finally, the materials models must be validated by comparing to experimental data. In this Chapter we provide an overview of the QA practices used in the development of the MARMOT code base, summarize the verification approach used with the tool, and discuss the validation of MARMOT.

5.1 QA Practices

The MARMOT tool follows the same QA approach taken by the MOOSE framework, which is Nuclear Quality Assurance grade one (NQA-1) certified. The MARMOT code base uses the Git version control system, which allows for collaborative code development by researchers located at various locations. The code is hosted on an internal INL server using the Gitlab software (see about.gitlab.com for more information), which provides easy to use tools that facilitate the QA approach taken for MARMOT. The QA approach employs issue tracking, merge requests, automated testing, and collaborative code review to facilitate collaborative development while ensuring that the code quality does not drop.

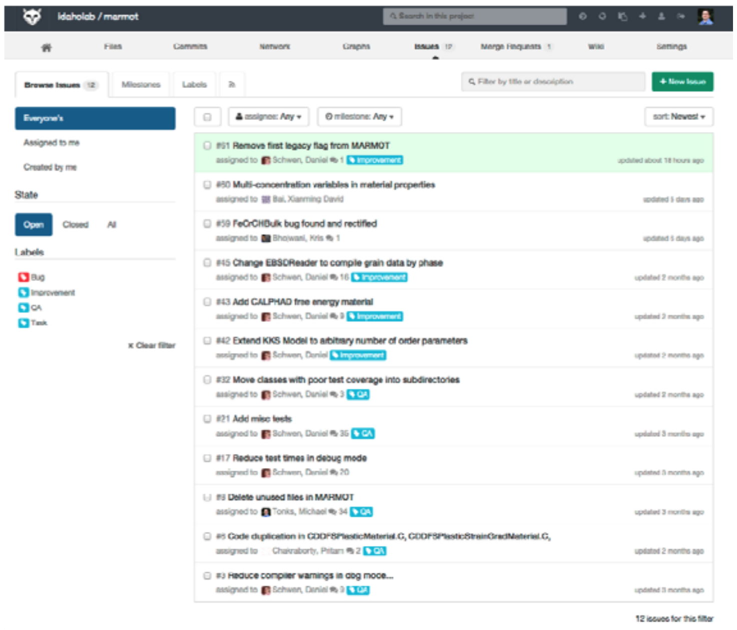
5.1.1 Issue Tracking

Whenever changes need to be made to the MARMOT code base, whether to add a feature or fix a bug, an issue must be submitted on the MARMOT Gitlab repository. Issues are submitted by the MARMOT development team as well as by any user that has access to the MARMOT repository. When an issue is submitted, the submitter provides a title, description, and can select labels and a milestone that the issue applies to. Then, the head of the MARMOT development team assigns the issue to either a developer or the person that submitted the issue. An example of the list of issues for MARMOT is shown in Fig. 5.1.

Once an issue is submitted, it is tracked in the system until it is closed. Some issues may be closed by a single set of changes to the code while others may be long standing and cover changes made for longer periods of time. Closed issues are still saved in the system, providing an automatic record of the history of the code.

5.1.2 Merge Requests

When a researcher has made changes to the code that address an existing issue, they can request that the changes be merged into the MARMOT code base by submitting a merge request on the MARMOT Gitlab website. When the merge request is made, it must be linked to an existing issue on the Gitlab website. The changes to the code go through basic checks once the merge request is submitted to ensure that the new code meets basic code standards. These include details like no trailing white space in the code. Once these basic checks are completed, the automated testing system is launched. The changes are merged with the code once the merge request is accepted by one of the primary MARMOT developers.



#61 Remove first legacy flag from MARMOT
assigned to Schwen, Daniel 1 Improvement — updated about 18 hours ago

#60 Multi-concentration variables in material properties
assigned to Bai, Xianming David — updated 5 days ago

#59 FeCrCHBulk bug found and rectified
assigned to Bhowani, Kris 1 — updated 5 days ago

#45 Change EBSDReader to compile grain data by phase
assigned to Schwen, Daniel 16 Improvement — updated 2 months ago

#43 Add CALPHAD free energy material
assigned to Schwen, Daniel 9 Improvement — updated 2 months ago

#42 Extend KKS Model to arbitrary number of order parameters
assigned to Schwen, Daniel Improvement — updated 2 months ago

#32 Move classes with poor test coverage into subdirectories
assigned to Schwen, Daniel 3 QA — updated 2 months ago

#21 Add misc tests
assigned to Schwen, Daniel 35 QA — updated 3 months ago

#17 Reduce test times in debug mode
assigned to Schwen, Daniel 20 — updated 3 months ago

#8 Delete unused files in MARMOT
assigned to Tonks, Michael 34 QA — updated 3 months ago

#5 Code duplication in CDDFSPlasticMaterial.C, CDDFSPlasticStrainGradMaterial.C,
assigned to Chakraborty, Pritam 2 QA — updated 2 months ago

#3 Reduce compiler warnings in dbg mode...
assigned to Schwen, Daniel 3 QA — updated 3 months ago

12 issues for this filter

Figure 5.1: Example of the list of issues on the INL MARMOT Gitlab repository website.

5.1.3 Automated Testing

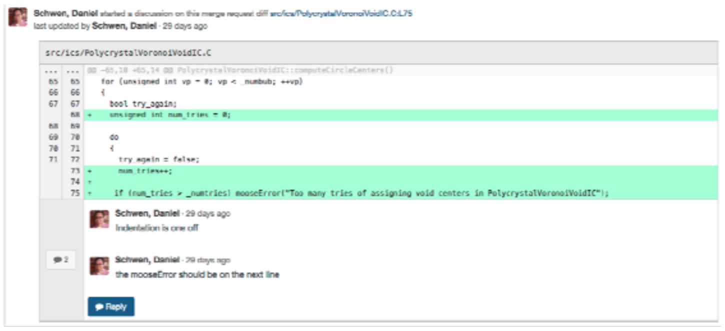
Over one hundred and fifty tests have been created to ensure that the MARMOT code base continues to give consistent predictions. The goal is to have every file within MARMOT be tested. Each of these tests are small simulations that can run in two seconds or less on a single processor. Every time a merge request is made, all the tests are rerun to ensure that the changes do not change the solutions in any unexpected ways. When new additions are added to the code, new tests must be added as well. Once a merge request is made, all the tests must pass before the changes will be merged into the code. If any tests fail, the researcher must make additional changes and resubmit the merge request.

5.1.4 Collaborative Code Review

Once all the tests pass, the changes to the code are reviewed by members of the MOOSE development team other than the one that submitted the merge request on the Gitlab repository website. On the website, comments can be made on the lines of the code that were changed, suggesting alternative approaches that could be taken or problems with the syntax. See Fig. 5.2 for an example of comments on a merge request. This collaborative review of the code ensures optimal quality and consistency of the entire code base. Once all the comments have been addressed and all developers are satisfied with the changes to the code, the merge request can be accepted and the issue closed.

5.2 Tool Verification

MARMOT models the coevolution of microstructure and material properties in irradiated fuel and cladding materials by solving the coupled phase field, solid mechanics, and heat conduction equations. To solve the equations, we use the finite element method (FEM), which approximates the field variables using shape functions across elements in a discretized domain. This solution approach, as with all numerical methods, has some error associated with it. In addition, mistakes in the code can lead to additional error. Therefore, it is essential to verify that every file implemented in MARMOT can be solved accurately and does not have errors. Verification in MARMOT is conducted in three ways: comparison against manufactured solutions, comparison against analytical models, and comparison against


Schwen, Daniel started a discussion on this merge request diff src/ics/PolycrystalVoronoiVoidIC.C:L75
last updated by Schwen, Daniel · 29 days ago

```
src/ics/PolycrystalVoronoiVoidIC.C
@@ -65,10 +65,14 @@ PolycrystalVoronoiVoidIC::computeCircleCenters()
   for (unsigned int vp = 0; vp < _numbub; ++vp)
   {
     bool try_again;
+    unsigned int num_tries = 0;

     do
     {
       try_again = false;
+      num_tries++;
+
+     if (num_tries > _numtries) mooseError("Too many tries of assigning void centers in PolycrystalVoronoiVoidIC");
```

Schwen, Daniel · 29 days ago
Indentation is one off

Schwen, Daniel · 29 days ago
the mooseError should be on the next line

Figure 5.2: Example of collaborative code review on the MARMOT Gitlab repository for a merge request.

atomistic simulation results.

5.2.1 Manufactured Solutions

The method of manufactured solutions is used to verify the discretized solution of a partial differential equation (PDE) or system of PDEs [28]. In the method, a closed analytic form of the solution is selected for a PDE test problem. Then, the analytic solution is substituted into the base PDEs to generate new or modified source terms in the equations. Finally, initial and boundary conditions are obtained by evaluating the selected solution form at zero time and at the boundaries. These source terms, initial, and boundary conditions are used in the numerical solution of the PDE and the error in the solution is determined by comparing to the analytical solution. Thus, the error can be calculated exactly, providing a robust means of evaluated the solution approach.

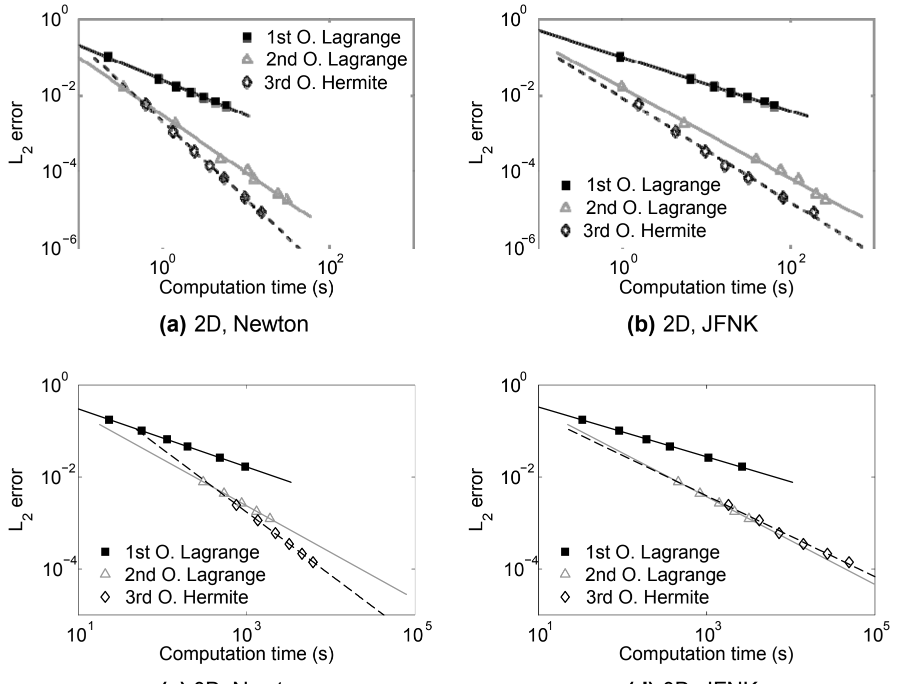
The Cahn-Hilliard (CH) equation is a time-dependent fourth-order partial differential equation that is used to solve for the evolution of conserved concentrations in the phase field method. When solving the CH equation via FEM in MARMOT, the domain is discretized by C^1-continuous basis functions or the equation is split into a pair of second-order PDEs, and discretized via C^0-continuous basis functions. In [29], a quantitative comparison between C^1 Hermite and C^0 Lagrange elements was carried out using MARMOT. The different discretizations were evaluated using the method of manufactured solutions solved with Newton's method and Jacobian-Free Newton Krylov (JFNK). It was found that the use of linear Lagrange elements provided the fastest computation time for a given number of elements, while the use of cubic Hermite elements provided the lowest error. When both computational time and accuracy were considered, the cubic Hermite elements achieved the lowest error per unit of computational time in 2D, while in 3D the Hermite elements and the quadratic Lagrange elements were more closely matched in this metric, as shown in Fig. 5.3.

5.2.2 Analytical Models

MARMOT solves systems of coupled equations across 2D and 3D domains to predict material behavior. However, often the same material behavior can be modeled analytically for much simpler systems and using simplifying assumptions. Therefore, MARMOT predictions can also be verified by comparing predictions for simple domains to the analytical solutions. While this approach can never prove that the predictions are accurate for all conditions, it does provide another means of evaluating the error in the solution approach used to solve the PDEs.

Grain Boundary Migration

A well-established analytical model exists defining grain boundary (GB) migration due to the curvature driving force. The MARMOT grain growth model has been compared to this analytical model, verifying



(a) 2D, Newton

(b) 2D, JFNK

(c) 3D, Newton

(d) 3D, JFNK

Figure 5.3: Computational efficiency (computation time vs. L^2-error) comparison of the three approaches in 2D using Newton's method and JFNK ((a) and (b), respectively), and in 3D using Newton's method and JFNK ((c) and (d), respectively). The computation time reported is the total wall time for one time step, not including the startup time. Taken from [29].

that we are accurately solving the phase field grain growth equations, as shown in Fig. 5.4. Results that have been verified against the analytical model have been checked into the testing system as the correct solution, ensuring that the code always maintains a good comparison with the analytical model. Both the analytical model and the phase field simulations require the GB energy and the GB mobility as material properties for a given material.

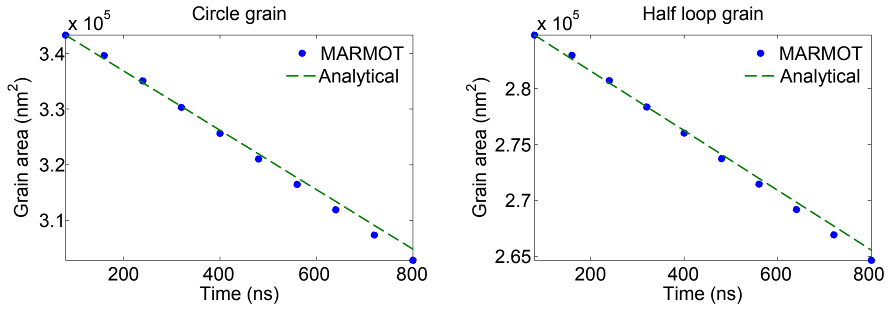


Figure 5.4: Verification of the MARMOT grain growth model for a circular grain (left) and half loop grain (right).

Fission Gas Migration

Analytical models exist to predict behavior related to fission gas. The fission gas transport at low concentrations can be verified by comparing to the solution of the diffusion equation. Both solutions should give very similar results, and all the fission gas transport models have been verified using this comparison. For the growth of a fission gas bubble, there exists an analytical 1D growth model that predicts the rate of growth given the bulk gas concentration and the diffusivity. Rate theory equations can also be used to verify the code and ensure that it is predicting the correct behavior. As with the grain growth model, verified results from the fission gas models have been used as the standard in the testing system.

5.2.3 Atomistic Simulation Results

Strictly speaking, verification ensures the accuracy with which the equations are being discretized and solved. Validation ensures the accuracy of the model by comparing to experimental data. However, there is another approach to investigate the accuracy of MARMOT that lies somewhere in between verification and validation, and that is comparing to higher fidelity models. Molecular dynamics (MD) simulations describe the microstructure evolution at small length and time scales by describing the interactions between individual atoms. Therefore, the accuracy of MARMOT simulations can be evaluated by directly comparing to MD simulations results. We have employed this approach to investigate the accuracy of the MARMOT models for grain growth, GB and bubble interaction, and fracture. Often these comparisons must be conducted using simpler materials systems for which MD empirical potentials exist in the literature.

Grain Growth

To compare the MARMOT and MD predictions of grain growth, we modeled the grain evolution in a bicrystal system with a circular grain embedded in a larger matrix grain. The MARMOT and MD simulations used the same simulation conditions and domain size (see [30] for a detailed description of this work). We took advantage of the mesh adaptivity capability in MARMOT to use a small GB width in the simulations ($w_b = 0.5$ nm) without requiring a large computational expense. The largest element size used in the simulations is 7.4 nm, while the smallest is 0.23 nm. The MARMOT predictions compared very well with the MD simulation results, as shown in Fig. 5.5

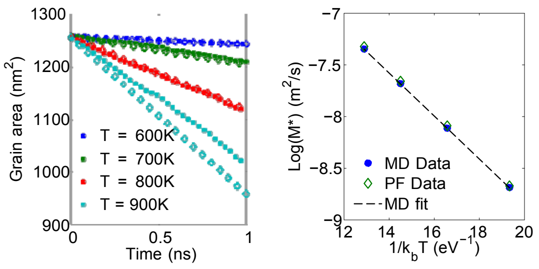


Figure 5.5: MD and MARMOT results for the embedded-circle bicrystal. The projected area of the embedded crystal as functions of time is shown on the left and the reduced mobility as a function of inverse temperature is shown on the right. In the left figure, the circles represent the MD data and the diamonds represent the PF data. The slope of fits to the data in the left figure are the values plotted in the right figure. The dashed line in the right figure is a liner fit to the MD data, which takes the form $M^* = 2.1526 \times 10^{-5} \exp(-0.4788/k_b T)$ m^2/s. Note the excellent agreement between the MARMOT and MD simulation results. Taken from [30].

GB and Bubble Interaction

Accurately predicting the interaction between GBs and bubbles is essential to understand grain growth in UO_2 fuel and fission gas behavior. Thus, it is essential to ensure that the model in MARMOT can accurately represent this behavior. Therefore, to verify that the MARMOT model accurately captures the interaction between pores and GBs, we compared the MARMOT results to MD simulation results representing identical molybdenum (Mo) systems containing helium (He) bubbles (for more detail see [20]). The reduction in volume of a circular grain with an initial radius of 20 nm was predicted by both simulation approaches, with ten bubbles randomly distributed along the GB. The 3D domain was 64.4 nm × 64.4 nm × 1.93 nm with periodic boundary conditions and the center axis of the circular grain was parallel with the *z*-axis. The ten He bubbles had a radius of 0.3 nm (ten He atoms were used to represent the bubbles in the MD simulations). The simulations were carried out at a temperature of 2700 K. Three simulations were run with different random initial positions of the ten He bubbles using both approaches. The phase field simulations employed a GB width of $l_{GB} = 0.2$ nm. Examples of the phase field and MD domains are shown in Fig. 5.6(a).

Both approaches predicted an initially slow reduction in the grain volume (see Fig. 5.6(b)), while all ten bubbles are in contact with the GB. However, the rate increases as the GB releases from more and more of the ten bubbles until the final rate of reduction is defined by the intrinsic GB mobility, as no bubbles are in contact with the GB. The comparison between these two distinct methods is reasonably good, though the phase field simulations seem to predict somewhat slower decrease in the grain size at the early stages of the volume reduction. Thus, the GB and bubble interaction model in MARMOT appears to accurately capture the behavior.

Fracture

In order to model the fracture behavior in UO_2, it is important to ensure that correct fracture stresses are predicted by the MARMOT simulations. Therefore, we compared phase field simulations of fracture along UO_2 GBs using MARMOT to results from MD simulations (more details can be found in [8]). The fracture strength from the MD simulations [21] was used as an input parameter for the MARMOT simulations and the comparison ensured that that the two methods produce consistent results. The system dimensions are 30.5 × 36.0 × 3.2 nm, with an elliptical hole of size 8 × 2 nm along the grain boundary in the middle. The loading is applied by adjusting the *y*-coordinate of each atom by 10^{-4} for every 1 ps, corresponding to a nominal engineering strain rate of 10^8/s. To evaluate the atomistic stresses, the Virial stress formulation is utilized to obtain the stress-strain curve. The nominal stress is represented by

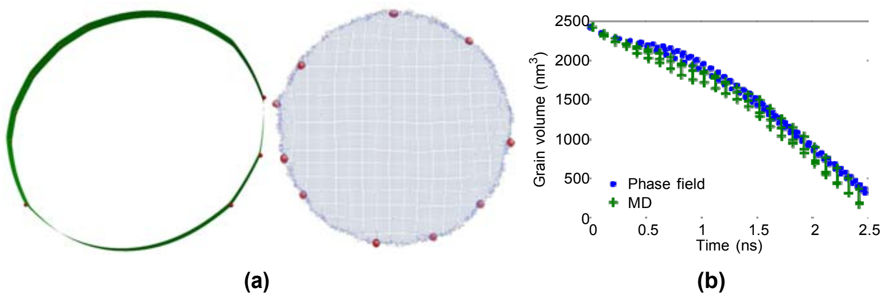


Figure 5.6: Verification of the phase field model by comparing to MD results. A shrinking Mo circular grain with ten He bubbles distributed along the grain boundary is modeled by both approaches, where examples of the phase field (left) and MD (right) simulation domains are shown in (a). The volume of the shrinking grain versus time predicted by both methods is shown in (b). Note that points from three different simulations for each approach with different initial bubbles positions are shown in the plot. The MARMOT phase field results show reasonably good agreement with the MD results. Taken from [20].

the average of the atomistic stresses over the entire volume. An identical domain was considered in the phase-field MARMOT simulations as shown in Fig. 5.7(b). Symmetry boundary conditions are applied to the RVE as opposed to periodicity in MD. In the simulations, a good agreement is obtained as shown in Fig. 5.7(c). The final configuration of the volume element is shown in Fig. 5.7(e).

5.3 MARMOT Validation

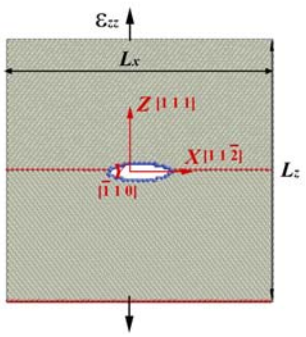
Before any MARMOT model can be used with confidence, the model predictions must be validated by comparing to robust experimental data. However, validating MARMOT requires unique data sets, because both the average microstructure evolution, e.g. average grain size, and the local microstructure changes must be validated. In addition, many of the material properties used by MARMOT to make material specific models have been calculated via atomistic simulations and must also be validated experimentally. Some validation has been completed and was discussed in Section 4. In this section, we discuss the unique aspects of validating spatially resolved microstructure simulations, and summarize many of the validation data that are needed in order to fully validate the MARMOT tool.

5.3.1 Microstructure Validation

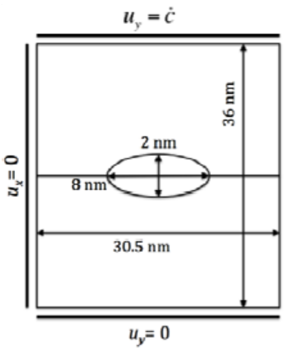
MARMOT predicts microstructure evolution in 2D and 3D, and spatially resolves the individual microstructural features. In addition, we often compute average properties across the spatially resolved microstructure, e.g. average grain size, porosity, to characterize some attributes of the microstructure. To validate a simulation tool that spatially resolves the microstructure, four critical points must be considered: average validation data is not sufficient, the simulations should be conducted in 3D, both initial and final microstructure characterization is needed, and simulations must accurately describe the validation experiment.

Average Data is Not Sufficient

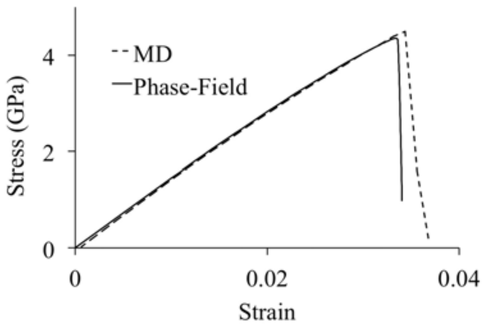
Often, microstructure data available in the literature comes in the form of average microstructural properties over time or burn-up. However, when we compare properties averaged across the simulated microstructure to these measured properties averaged across an experimental microstructure, the local predictions of the model are not validated even if the average properties compare well. This is because significant local phenomena may not even be apparent in the averaged microstructure properties. For



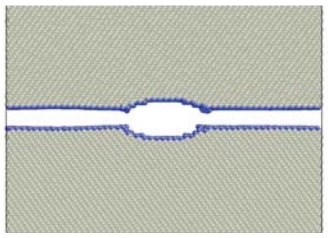
(a)



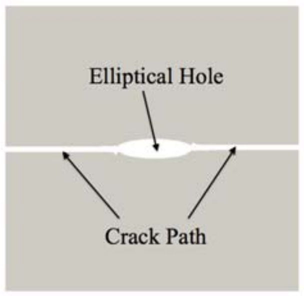
(b)



(c)



(d)



(e)

Figure 5.7: Comparison between MD and MARMOT simulations of fracture in UO_2, where (a) is the simulation cell used in MD, (b) is a schematic of the MARMOT domain, (c) compares the stress-strain evolution between MARMOT and MD simulations, (d) is the final configuration of the MD simulation and (e) is the final configuration of the MARMOT simulation. The MARMOT and MD simulation results show good agreement. Taken from [8].

example, in our MARMOT simulations of temperature gradient driven grain growth (see Fig. 4.4), the average grain size showed no dependence on temperature gradient. However, there was significant temperature gradient dependence in the local behavior of the individual grains. The MARMOT models can only be fully validated by directly comparing microstructure characterization data to the predicted microstructures. This can be accomplished by directly comparing the experimental and the predicted microstructures, or by comparing statistical descriptors of the microstructures such as grain size distributions, average disorientation angles, etc.

Simulations Should be Conducted in 3D

Material behavior in nature is always a 3D phenomena (with a very few exceptions), therefore a 2D simulation is always an approximation of the real material behavior. For this reason, simulations should be run in 3D whenever possible when comparing to data. Most microstructure characterization methods are surface methods and produce a 2D image of the surface microstructure. However, the microstructure evolution below the surface will significantly impact the surface microstructure. So, even when comparing to 2D surface characterization, the simulations should be run in 3D. 3D characterization data will provide the best comparison to the 3D simulation results, however 3D characterization data is often difficult to obtain and the methods are often destructive.

Both Initial and Final Microstructure Characterization is Needed

While MARMOT predicts the evolution of the microstructure, the final microstructures that result from this evolution will be highly sensitive to the initial condition of the microstructure. Therefore, microstructure characterization data after the experiment (e.g., annealing, mechanical deformation, or irradiation) is conducted, is not sufficient to validate the model. We must also have the same characterization data for the initial microstructure in order to correctly reconstruct the initial condition for the simulations. The best results are obtained when the microstructure is characterized before and after the experiment in the same region of the sample, though this is only possible when non-destructive characterization approaches are employed. When destructive characterization is used on the initial microstructure, a batch of samples (with similar microstructural features) can be employed, where different samples are characterized before and after the experiments.

Simulations Must Accurately Describe the Validation Experiment

Validation experiments are conducted to obtain the necessary data required to fully validate a specific model of interest. Due to procedural, technical, or financial constraints, if may be impossible to collect the necessary data using typical application conditions. For example, it may be difficult to obtain the required data from a fuel sample in a commercial LWR so a test reactor may be used for the irradiation instead. When conducting the simulation that will be compared to the validation data, it is essential that the simulation capture the conditions of the validation experiment rather than the conditions of the typical application, e.g. the simulation should use the test reactor conditions rather than typical LWR conditions. At times the differences between the validation experiment and the application may be quite drastic. For example, an experiment designed to validate a grain growth model in MARMOT uses an annealed sample and does not involve irradiation at all.

5.3.2 Validation Data Needs

Though some validation has been conducted on MARMOT models, there is still a significant amount of validation that needs to be conducted. The most common limitations with existing data from the literature for this validation are that there is very little initial characterization of the microstructure and often only averaged microstructure properties are reported. Therefore, a significant amount of new data needs to be collected to validate MARMOT simulation results and the atomistically-determined material properties. In this section we summarize the data needs organized by the physical phenomena. All of the data needs are applicable to UO_2 but also to other fuel materials.

Fission Gas Behavior

- Fission gas behavior in fabricated samples with gas atoms already present (no irradiation)
  - Fission gas transport in single crystals
  - Fission gas transport and bubble migration in a temperature gradient
  - Quantification of fission gas segregation for different GB types
  - Bubble nucleation, growth, and interconnection for different GB types
- Fission gas behavior in irradiated samples
  - Characterization of 3D percolated bubble structures in polycrystalline samples
  - Impact of point defect concentration on fission gas transport

Grain Growth

- Bicrystal experiments to obtain properties of individual GBs
  - Measurement of GB mobility
  - Impact of irradiation on GB mobility
  - Interaction of single GB with impurities/bubbles
- Polycrystal experiments
  - Annealing of well characterized microstructures in very pure, fully dense material
  - Annealing with various pore densities and well characterized microstructures
  - Grain growth under irradiation
  - Grain growth under high temperature gradients

Fuel Cracking

- In situ nano-indentation
  - Single crystal or large grains
  - Nanocrystalline material
  - Irradiated material (neutron or ion)
- Micro-indentation
  - Well-characterized fully dense polycrystals
  - Well-characterized microstructures with GB bubbles and/or irradiation

Effective Material Properties

- Measurement of GB thermal resistance
- Local characterization of local properties of different microstructural features
- Measurement of effective material properties of characterized microstructures
- Fundamental measurement of intrinsic properties
- Impact of externally applied fields on property values

6 Conclusions

The MARMOT tool being developed by the NEAMS FPL predicts the coevolution of microstructure and material properties in fuel and cladding materials due to temperature, stress, and irradiation damage. MARMOT is built using the capabilities available in the phase_field, tensor_mechanics, and heat_conduction modules in the open source MOOSE framework. Though MARMOT is being developed as a mesoscale fuel performance tool, it is also being used to inform the development of materials models that are based on microstructure rather than burn-up for fuel performance codes.

MARMOT has been funded by the NEAMS program to develop models for UO_2. Up to now, this development has been focused on models of fission gas behavior, grain growth, cracking, and effective material property calculation. Starting in fiscal year 2015, MARMOT is also being funded to develop models of U-Si fuel, though this efforts are waiting on the development of a U-Si inter-atomic potential. In FY 2015, the Advanced Fuel Cycle program and the RERTR program are also funding the development of metal fuels models in MARMOT.

The MARMOT code base undergoes rigorous quality assurance and verification throughout its development, and the models are also being validated when data is available. MARMOT is developed by a community of users and the code base is managed with the Git version control system. It is hosted on an internal INL Gitlab repository which has automated tools for issue tracking, automated testing, and collaborative code review. The code is verified using the method of manufactured solution, comparison to analytical models, and comparison to molecular dynamics simulations. Because MARMOT spatially resolves the microstructure, average validation data is not sufficient, the simulations should be conducted in 3D, both initial and final microstructure characterization is needed, and simulations must accurately describe the validation experiment. A list of data needed for validation of MARMOT is given in Section 5.

Bibliography

[1] D. Gaston, C. Newman, G. Hansen, and D. Lebrun-Grandié. MOOSE: A parallel computational framework for coupled systems of nonlinear equations. *Nucl. Eng. Design*, 239:1768–1778, 2009.

[2] M.R. Tonks, D. Gaston, P.C. Millett, D. Andrs, and P. Talbot. An object-oriented finite element framework for multiphysics phase field simulations. *Comp. Mat. Sci.*, 51(1):20–29, 2012.

[3] L.Q. Chen. Phase-field models for microstructure evolution. *Annu. Rev. Mater. Res.*, 32:113–40, 2002.

[4] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[5] Benjamin S. Kirk, John W. Peterson, Roy H. Stogner, and Graham F. Carey. libMesh: A C++ library for parallel adaptive mesh refinement/coarsening simulations. *Engineering with Computers*, 22(3-4):237–254, January 2006.

[6] D Schwen, E Martinez, and A Caro. On the analytic calculation of critical size for alpha prime precipitation in fecr. *Journal of Nuclear Materials*, 439(1):180–184, 2013.

[7] Seong Gyoon Kim, Won Tae Kim, and Toshio Suzuki. Phase-field model for binary alloys. *Phys. Rev. E*, 60:7186–7197, Dec 1999.

[8] MR Tonks, Y Zhang, A Butterfield, and X Bai. Multi-scale modeling of microstructure dependent intergranular brittle fracture using a quantitative phase-field based method. *Computer Methods in Applied Mechanics and Engineering*, Submitted, 2015.

[9] DA Andersson, P Garcia, X-Y Liu, G Pastore, M Tonks, P Millett, B Dorado, DR Gaston, D Andrs, RL Williamson, et al. Atomistic modeling of intrinsic and radiation-enhanced fission gas (xe) diffusion in uo2$\pm$x: Implications for nuclear fuel performance modeling. *Journal of Nuclear Materials*, 451(1):225–242, 2014.

[10] David A. Andersson, Michael R. Tonks, Luis Casillas, Shyam Vyas, Pankaj Nerikar, Blas P. Uberuaga, and Christopher R. Stanek. Multiscale simulation of xenon diffusion and grain boundary segregation in $\{$UO2$\}$. *Journal of Nuclear Materials*, 462:15 – 25, 2015.

[11] S Kashibe, K Une, and K Nogita. Formation and growth of intragranular fission gas bubbles in uo 2 fuels with burnup of 6–83 gwd/t. *Journal of nuclear materials*, 206(1):22–34, 1993.

[12] Michael R Tonks, S Bulent Biner, PC Mille, and David A Andersson. Comparison between phase field simulations and experimental data from intragranular bubble growth in uo 2. In *Proceedings of the 2013 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering-M and C 2013*, 2013.

[13] Yulan Li, Shenyang Hu, Robert Montgomery, Fei Gao, and Xin Sun. Phase-field simulations of intragranular fission gas bubble evolution in uo 2 under post-irradiation thermal annealing. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 303:62–67, 2013.

[14] Michael R. Tonks, Yongfeng Zhang, Xianming Bai, and Paul C. Millett. Demonstrating the temperature gradient impact on grain growth in UO_2 using the phase field method. *Materials Research Letters*, 2(1):23–28, 2014.

[15] John W Cahn. The impurity-drag effect in grain boundary motion. *Acta metallurgica*, 10(9):789–798, 1962.

[16] Andreas Michels, CE Krill, H Ehrhardt, R Birringer, and DT Wu. Modelling the influence of grain-size-dependent solute drag on the kinetics of grain growth in nanocrystalline materials. *Acta materialia*, 47(7):2143–2152, 1999.

[17] RD Doherty, DJ Srolovitz, AD Rollett, and MP Anderson. On the volume fraction dependence of particle limited grain growth. *Scripta metallurgica*, 21(5):675–679, 1987.

[18] Burton R Patterson and Yixiong Liu. Quantification of grain boundary-pore contact during sintering. *Journal of the American Ceramic Society*, 73(12):3703–3705, 1990.

[19] C Zener quoted by C Smith. *Trans Met Soc AIME*, 175:15–51, 1948.

[20] MR Tonks, Y Zhang, A Butterfield, and X Bai. Development of a grain boundary pinning model that considers particle size distribution using the phase field method. *Modeling and Simulation of Materials Science and Engineering*, In press, 2015.

[21] Y. Zhang, P. C. Millett, M. R. Tonks, X M Bai, and S B Biner. Molecular dynamics simulations of intergranular fracture in {UO2} with nine empirical interatomic potentials. *Journal of Nuclear Materials*, 452(1–3):296 – 303, 2014.

[22] JD Hales, MR Tonks, K Chockalingam, DM Perez, SR Novascone, BW Spencer, and RL Williamson. Asymptotic expansion homogenization for multiscale nuclear fuel analysis. *Computational Materials Science*, 99:290–297, 2015.

[23] Paul C. Millett and Michael Tonks. Meso-scale modeling of the influence of intergranular gas bubbles on effective thermal conductivity. *J. Nucl. Mater.*, 412(3):281 – 286, 2011.

[24] Michael R Tonks, Paul C Millett, Pankaj Nerikar, Shiyu Du, David Andersson, Christopher R Stanek, Derek Gaston, David Andrs, and Richard Williamson. Multiscale development of a fission gas thermal conductivity model: Coupling atomic, meso and continuum level simulations. *Journal of Nuclear Materials*, 440(1):193–200, 2013.

[25] Paul C Millett, Michael R Tonks, K Chockalingam, Yongfeng Zhang, and SB Biner. Three dimensional calculations of the effective kapitza resistance of UO_2 grain boundaries containing intergranular bubbles. *Journal of Nuclear Materials*, 439(1):117–122, 2013.

[26] Melissa C Teague, Bradley S Fromm, Michael R Tonks, and David P Field. Using coupled mesoscale experiments and simulations to investigate high burn-up oxide fuel thermal conductivity. *JOM*, 66(12):2569–2577, 2014.

[27] K. Bakker. Using the finite element method to compute the influence of complex porosity and inclusion structures on the thermal and electrical conductivity. *International Journal of Heat and Mass Transfer*, 40(15):3503 – 3511, 1997.

[28] P. J. Roache. Code verification by the method of manufactured solutions. *Journal of Fluids Engineering*, 124:4, 2002.

[29] Liangzhe Zhang, Michael R Tonks, Derek Gaston, John W Peterson, David Andrs, Paul C Millett, and Bulent S Biner. A quantitative comparison between c0 and c1 elements for solving the cahn–hilliard equation. *Journal of Computational Physics*, 236:74–80, 2013.

[30] Michael R Tonks, Yongfeng Zhang, SB Biner, Paul C Millett, and Xianming Bai. Guidance to design grain boundary mobility experiments with molecular dynamics and phase-field modeling. *Acta Materialia*, 61:1373–1382, 2013.