

Benchmarking and Performance of the NASA Multiscale Analysis Tool

Evan J. Pineda,¹ Trenton M. Ricks,²
Brett A. Bednarczyk,³ and Steven M. Arnold⁴
NASA Glenn Research Center, Cleveland, OH 44135, U.S.A.

The NASA Multiscale Analysis Tool (NASMAT) is as a “plug and play,” software package which utilizes multiscale recursive micromechanics as a platform for massively multiscale modeling of hierarchical materials and structures subjected to thermomechanical. This paper is intended to give an overview of the design of NASMAT and how the design supports modularity, upgradability and maintainability, interoperability, and utility. First, the software architecture and hierarchy will be explored. Details on each of the 11 NASMAT procedures and the arrangement of NASMAT data will be presented. Application program interfaces (APIs) that were developed to facilitate the communication of NASMAT with other programs will be described. The intended application for NASMAT is massively multiscale modeling on high performance computing systems. As such, results benchmarking the performance of the integration of NASMAT with the Abaqus commercial finite element method software are also presented.

I. Overview of NASMAT

The development of the NASA Multiscale Analysis Tool (NASMAT) has focused on modularity, upgradability and maintainability, interoperability, and utility. The code has been designed such that the various functionalities are compartmentalized into a set of 11 procedures: Pre-processing, Driver, Engine, Solution, Homogenization, Material Model, Localization, State Change, Output, Post-processing, and Utilities. Each procedure has access to a library of modules. Specific modules can be swapped in and out, as needed, to solve the multiscale problem of interest. Numerous procedures, and the subroutines used within the modules for those procedure, are recursive. Moreover, recursive data types are used extensively to handle the large quantities of data associated with each length scale considered by the multiscale model. The recursive nature of the NASMAT procedures and data enabled the development of a multiscale recursive micromechanics (MsRM) support massively multiscale modeling (M³) on high performance computing systems (HPC). Finally, application program interfaces (APIs) have been developed to facilitate the integration of NASMAT into other programs (commercial, research, and user-defined) as well as the integration of other codes into NASMAT itself. Some preliminary benchmarking results are presented in the last section of this paper.

II. Procedure Hierarchy

To facilitate “plug and play” functionality, the necessary NASMAT computational operations were categorized into 11 procedures. Those procedures include *Pre-processing*, Driver, Engine, Solution, *Homogenization*, Material Model, *Localization*, *State Change*, Output, *Post-processing*, and Utilities. The italicized procedures indicate procedures that must be recursive to support multiscale modeling. The hierarchy of these procedures is given in Figure 1. The function of these procedures along with the current, and possible future, modules available for use in

¹ Research Aerospace Engineer, Multiscale and Multiphysics Modeling Branch, AIAA Senior Member.

² Research Aerospace Engineer, Multiscale and Multiphysics Modeling Branch, AIAA Member.

³ Research Materials Engineer, Multiscale and Multiphysics Modeling Branch, AIAA Associate Fellow.

⁴ Technical Lead, Multiscale and Multiphysics Modeling Branch, AIAA Senior Member.

each procedure will be described in the following subsections. Note, the Utilities and Post-processing procedures do not fall within the NASMAT hierarchy because they are ancillary.

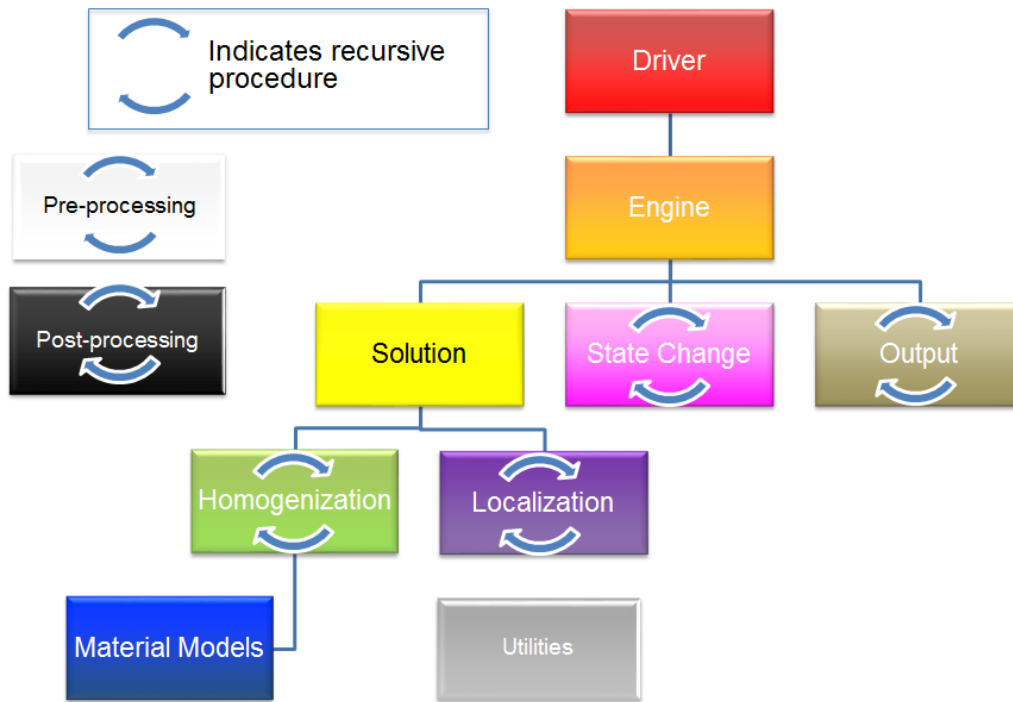


Fig. 1 Hierarchy of NASMAT procedures. Each procedure has access to a library of modules that can be exchanged to solve the particular multiscale problem of interest. The curved arrows indicate that the procedure is recursive.

Pre-processing: In this ancillary procedure all the necessary input files are read including the NASMAT .mac file and any files need for user-defined modules. The NASMAT data structures are also allocated in this procedure. The data allocation must be recursive to support MsRM.

Driver: The Driver is used to provide the global thermal and mechanical load increments. The currently available Driver modules include user-defined, quasi-static load, integration within Abaqus finite element method (FEM) software, and integration with any other third-party structural analysis software through the NASMAT MacroAPI.

Engine: The Engine is the core of the NASMAT software package and serves as the interface between the Driver and Solution procedures. The Engine is unique and does not contain any modules. All NASMAT analyses must integrate the engine into the workflow.

Solution: The Solution dictates the appropriate solution method needed to solve the problem of interest. The current Solution modules in NASMAT are the constituent solution, which is used to solve problems involving monolithic materials, and the micromechanics solution, which is used to analyze heterogeneous materials defined using a repeating unit cell (RUC) with periodic boundary conditions. Although not currently available, a classical lamination theory solution module will be implemented future versions of NASMAT [1].

Homogenization: In order to obtain the necessary information to solve a non-linear thermomechanical problem, the Solution procedure must call two other procedures. The first is the Homogenization procedure. The modules within this procedure include various micromechanics theories including the generalized method of cells (GMC), the high-fidelity GMC (HFGMC), the parametric HFGMC (PHFGMC), the Carerra unified formulation (CUF), and the Mori-Tanaka mean field theory (MT) [2-5]. The Homogenization procedure provides the Solution procedure with the effective thermomechanical properties of the RUC considering material nonlinearity in the RUC constituents. This procedure must be recursive to enable MsRM. CLT homogenization, and homogenization utilizing user-defined micromechanics, through the NASMAT MicroAPI, will be available with future releases of NASMAT.

Material Models: The Homogenization procedure must have access to the Material Models procedure in order to obtain the constitutive response of the material points in the RUC. The currently available Material Models include

linear elasticity, linear thermal expansion, and material point elimination (MPE). With MPE, all of the components of the stiffness matrix for the material point are reduced to a near zero value after satisfaction of a failure criterion which is evaluated in the State Changes procedure. Progressive damage and viscoplastic modules will be added to the Material Models solution for a future release of NASMAT.

Localization: In order to solve a nonlinear micromechanics, or multiscale, problem, the Solution procedure must also invoke the Localization procedure. The theory used in the Localization procedure must be complimentary to what is used in the Homogenization procedure. The Localization procedure is responsible for calculating the local fields (stress, strain, etc.) within the RUC. This procedure must be recursive to support MsRM.

State Changes: The NASMAT Engine calls the recursive State Changes procedure to determine if the state of the material changes and set the appropriate flags. These flags will be utilized in Material Models procedure to elicit the appropriate constituent response. Currently, only material point failure criteria are supported within the State Changes procedure, but other possible State Changes modules include crack growth, oxidation, microstructural changes, and interfacial debonding.

Output: The recursive Output module writes the desired output data to the hard disk. Currently data can be output into an ASCII text-based format and/or a HDF5 database.

Utilities: There are several ancillary utility functions that are made available throughout the entirety of the NASMAT program.

Post-processing: The Post-processing procedure is handled outside of the NASMAT analysis hierarchy and involves visualization of the NASMAT data. It is necessary that this procedure is recursive to enable multiscale visualization.

III. Data Management

To utilize shared-memory multi-processing programming, such as Open Multi-processing (Open MP), global data variables (i.e., public) must be carefully used to store local, independent data from each core. However, while private variables are available to each core, a significant memory penalty is incurred. Moreover, M3 requires the storage of a tremendous number of independent variables. Therefore, to exploit HPC and to organize the data for MsRM, the data used in NASMAT is categorized into 7 types: Deck, Fields, Properties, Flags, Step, Solution, and RUC Data. Derived type data structures are defined for each of the data categories, and these data structures are passed by reference throughout the code. The data structures are themselves recursive. I.e., a member within a particular structure is of the same derived type as the parent structure. This allows for seamless MsRM framework in which the same data variables, and the same recursive subroutines, can be used for computations at any scale, including the global. The data types are categorized as follows:

Deck: Deck contains all of the information read from the .mac, or user-defined, input files. The Deck data structure is static upon application and population and remains as a reference.

Fields: Fields are the quantities which culminate into the observed deformation of the RUC. NASMAT Fields include stress, strain, and temperature, for example.

Properties: The Properties data structure contains all of the material property information needed to invoke the appropriate Material Models solution module. Examples include material stiffness matrix, compliance matrix, coefficients of thermal expansion (CTEs), and failure criteria allowables.

Flags: Flags are data, usually binary, that indicate outcome of a Boolean operation within NASMAT. These flags are initialized in the Pre-processing solution and are updated, as needed, throughout the code. An example of a Flag datum the failure of a material point. This Flag is false until a failure criteria associated with that material point is satisfied.

Step: This data structure contains information on the global time increment and non-linear solution iteration. Since, all of this data is global, and NASMAT does not yet support disparate time scales or nonlinear solution procedures, this structure is not recursive.

Solution: All of the data that is needed in the Solution, Homogenization, and Localization procedures modules to complete the desired operations is stored in the Solution data structure. An example of Solution data is the strain concentration matrix in GMC [2].

RUC Data: RUC Data contains all of the geometric data pertaining to the RUCs in the analysis. It also contains indicators of the materials that comprise each constituent.

IV. NASMAT Application Programming Interface

Several NASMAT API are under development to support user defined procedure modules within NASMAT. These API are written to support Fortran, C, and C++ interfaces. The two API that are near completion and will be

available in NASMAT version 2.0 are MacroAPI and Micro API. The MacroAPI integrates NASMAT within a third-party structural analysis software, such as an FEM solver, to provide material point properties. The MicroAPI integrates a user-defined micromechanics theory within the NASMAT MsRM framework. The MicroAPI is still under development and will not be presented in this paper, but two external micromechanics theories (PHFGMC and CUF) have already easily been integrated as modules within NASMAT [3, 4].

MacroAPI: Figure 2 contains a diagram outline of the user-defined procedures that must be deployed in order to interface NASMAT with a third party structural analysis code to provide material point information. The uncoloured shapes with dashed outline represent operations that must be defined by the user. The actual NASMAT interface that is used is dependent on the language that the higher-level code is written in (Fortran, C/C++). The MacroAPI has already been used to interface the Abaqus, CalculiX, and the internal General Electric (GE) Scalable Parallel FEA (SPAF) FEM codes. Figure 2 is partitioned into three separate user-defined procedures.

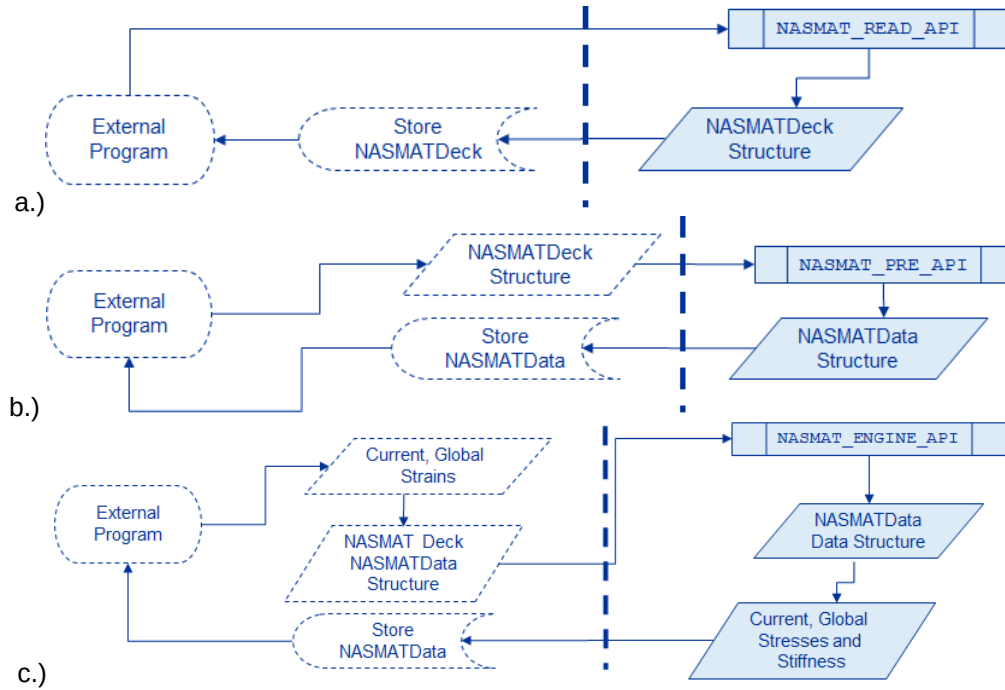


Fig. 2 Diagram show three necessary user defined procedures in the NASMAT MacroAPI which interfaces NASMAT with external, structural analysis codes. The uncolored dashed shapes indicate that the operation must be defined by the user. a.) NASMAT input file reading. b.) NASMAT pre-processing. c.) NASMAT analysis.

First, shown in Figure 2a, the user must call the front end subroutine for the Pre-processing module that reads the NASMAT input files: `NASMAT_READ_API`. NASMAT will then allocate and populate the `NASMATDeck` data structure and pass it back to the user subroutine. The user is responsible for storing this data structure and passing it back unaltered to other MacroAPI procedures. This user-defined procedure must be implemented in the user code for each unique NASMAT input file needed in the analysis.

Next, the user must call subroutines that perform NASMAT data allocation and initialization through `NASMAT_PRE_API`, as shown in Figure 2b. The user must pass the `NASMATDeck` data structure, associated with the current user-defined material point, to `NASMAT_READ_API`, and `NASMAT_READ_API` will return the `NASMATData` data structure which contains all of the other data structures described in Section 3 as children, except `NASMATDeck`. This user-defined procedure must be invoked for every user-defined material point that is to be linked to a NASMAT analysis.

Finally, the user must run the NASMAT analysis for each desired material point via `NASMAT_ENGINE_API`. The user must pass the current material point strain, and the appropriate `NASMATDeck` and `NASMATData` structures associated with this material point. `NASMAT_ENGINE_API` will return homogenized thermomechanical properties

of the relevant RUC, including stiffness and CTE, as well as the global volume averaged stiffness that is work conjugate to the strains that were provide by the user. that comprise each constituent.

V. Finite Element Method Multiscale Analysis Tool

One of the primary development goals pertaining to NASMAT is operation on HPC systems. The first step towards this goal is ensuring the program is thread safe and can execute in a parallel computing environment. As such, the NASMATv2.0 code is not currently parallelized itself, but can interface with higher level structural analysis codes that are parallelized for multiscale modeling.

The integration of NASMAT with an FEM model employs synergistic multiscale modeling. For a given time increment, the material behavior of the element integration point (IP) are directly linked to a micromechanics RUC model. Global strains at the element IP are passed to the RUC, and a localization procedure is performed to obtain the local fields. The local fields are then used to update local material properties, inelastic strains and state variables. A homogenization procedure is then performed to obtain the effective material properties of the RUC and the global stress which are passed back to finite element IP. Once the solution to the global FEM model has converged, the FEM solver marches forward to the next increment and the scales are again traversed for each multiscale element. The synergistic multiscale modeling method can be computationally intensive. Thus, it is crucial that it scales well on multiple CPUs.

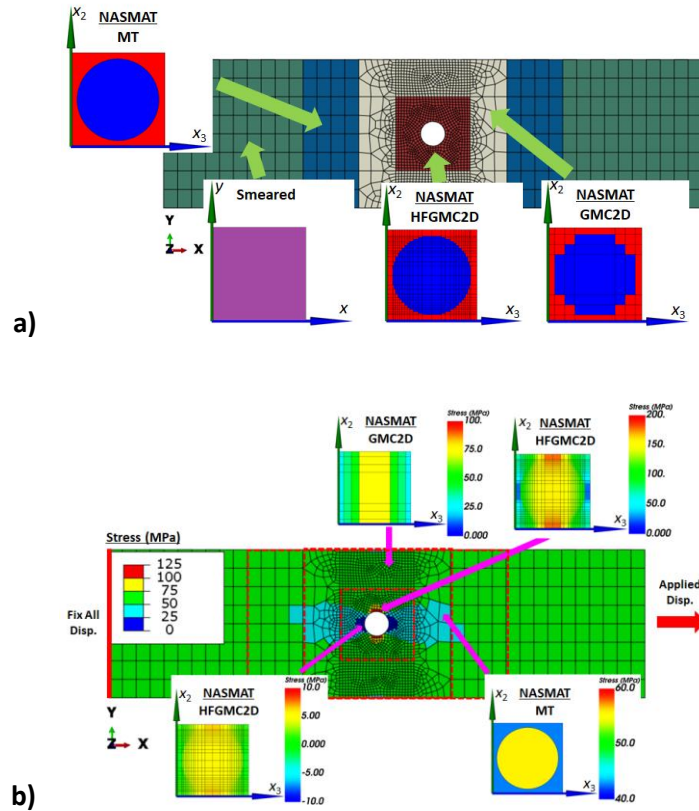


Fig. 1 Multiscale NASMAT model of a unidirectional composite lamina with a hole. a) The macroscopic domain is strategically linked to micromechanics models with various degrees of fidelity. Red: HFGMC, gray: GMC, blue: MT, green: effective properties. b) The global and local stresses when a displacement is applied to the edge of the specimen.

A multiscale example is presented in Figure 3a. In this example, the macroscale domain is a plate with a hole subjected to uniaxial loading. Each element is synergistically linked to a NASMAT micromechanics model of a RUC containing a single carbon fiber in an epoxy matrix. Thus, the effective response of the entire model is that of a unidirectional composite lamina. The macroscopic domain is divided into four different regions, as indicated with the different colors in Figure 3a. The fidelity of the micromechanics model used decreases as the elements are located

farther away from the hole. In the direct vicinity of the notch, the red elements are linked to HFGMC RUCs and the surrounding gray elements are tied to GMC RUCs. In the blue elements, located more than half the width of the specimen away from the notch, MT is utilized. Finally, in the acreage regions (green elements), no RUC is defined and simply the effective homogenized properties for the composite are used. This method of varying the fidelity of the micromechanics spatially is an effective way to reduce the computational time of the multiscale model but retain the desired accuracy. Higher fidelity models are required in regions where the field gradients are large, such as near a cut-out, but this level of accuracy may not be need where the fields do not change drastically and are not large enough to invoke a nonlinear material response. Figure 3b shows the global and local fields when the plate is loaded, assuming the behavior is elastic.

VI. Performance of NASMAT Integrated with Abaqus on Multiple CPUs

To demonstrate the performance of NASMAT in a parallel computing environment, NASMAT is integrated into the commercial FEM code Abaqus using the NASMAT MacroAPI and Abaqus/Standard UMAT and UEXTERNALDB user subroutines. Parametric studies were conducted with macroscale FEM model and microscale GMC micromechanics models of varying size. The FEM model consisted of 64, 1000, 5832, and 10645 C3D8R reduced integration 3D brick elements. The doubly-periodic RUCs used in this study contained 4, 49 and 196 subcells. During the analysis each element was linked to an RUC. Multiscale modeling simulations composed of permutations of the macroscale FEM models and microscale RUCs were performed on 1, 2, 8, 16, 32, and 64 CPUs on a Windows 10 workstation with two Intel Xeon Silver 4126 processors containing 32 CPUs each.

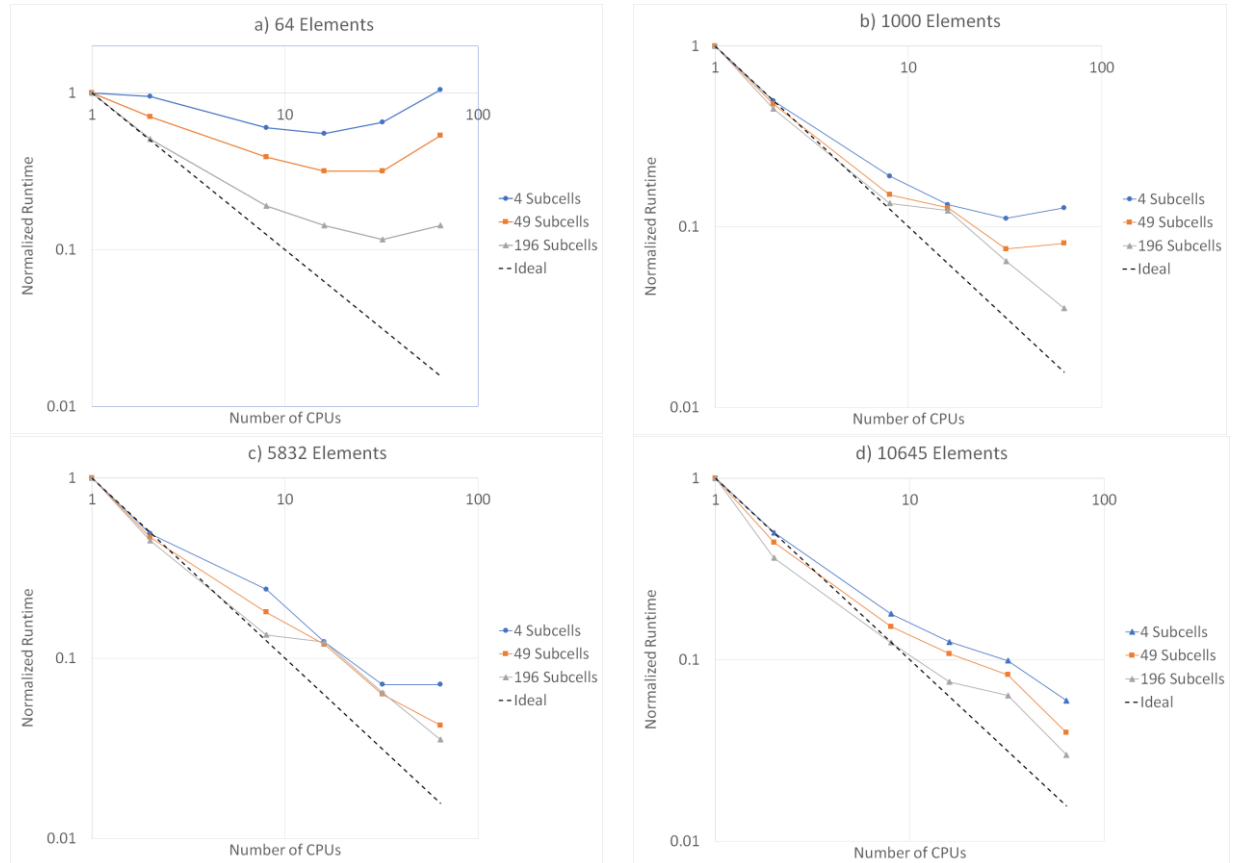


Fig. 4 Normalized runtime versus number of CPUs for various macroscale models containing 4, 49, and 196 subcells. Ideal normalized runtime versus number of CPUs plotted for reference. a) 64 elements. b) 1000 elements. c) 5832 elements. d) 10645 elements.

Figure 4 shows the computational runtime (normalized against the timer required for an analysis using a single CPU) versus the number of CPUs used to run the simulation on a log-log scale. Each graph contains data for a fixed size FEM model. Data obtained using different RUC sizes are plotted on each chart. Each subfigure in Figure 4 also contains a line representing the expected runtime for ideal scaling: $\text{runtime} = 1/\text{number of CPUs}$. For a given number of elements, the scaling trends are the similar for the different RUCs. For the cases with 64 elements shown in Figure 4a, computational time actually begins to increase when more CPUs are used beyond 16. If 64 CPUs are used, the runtime is actually slower than when a single CPU is used. This is expected because the workload distributed to each computational node is not substantial enough to overcome the overhead of passing data back to the master node. For the other meshes the software scales well as a function of the number of CPUs. The simulations do see some reduction in the efficiency of adding the additional CPUs after 8 CPUs, but it is not tremendous. For all meshes the scaling improves as the size of the RVE is increased. The normalized runtime versus number of CPUs for all the analysis utilizing an RUC with 196 subcells is plotted in Figure 5. From this figure, it does not appear that the runtime scaling is significantly affected by the number of elements in the problem (at or above 1000). All data points only represent a single analysis, and it is expected that there is some scatter in the runtime even if the conditions do not change. Therefore, the results may change if a statistical average is taken, but it is not expected that this variability would change any of the observations made in this discussion.

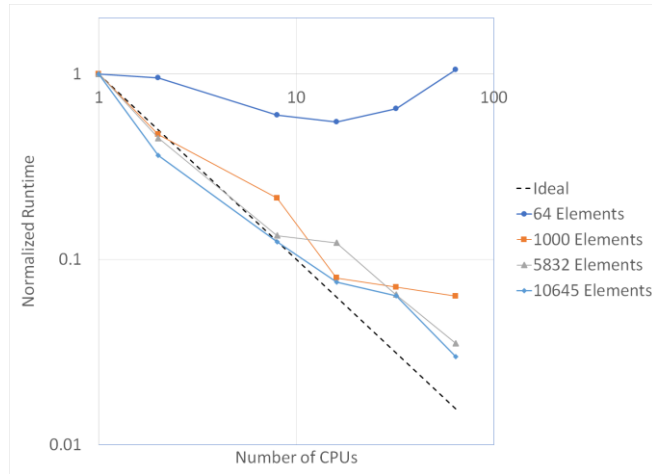


Fig. 5 Normalized runtime versus number of subcells for 64, 1000, 5832, and 10645 elements. Ideal normalized runtime versus number of CPUs plotted for reference.

VII. Conclusion

NASMAT is intended to be a virtual platform for simulating the thermomechanical response of heterogeneous materials and structures. This was achieved by developing an MsRM framework which utilizes recursive subroutines and data structures. To facilitate “plug and play” interoperability with other software as well as modularity, upgradability, maintainability, the NASMAT hierarchy is categorized into 11 key procedures which are described in detail in this manuscript. Each of these procedures has access to a library of modules which can be used to solve the specific problem of interest. The NASMAT MacroAPI was developed so that NASMAT can be easily integrated, through a Fortran or C/C++ interface, into external structural analysis codes for multiscale simulation of structures. A multiscale (NASMAT synergistically linked to FEM) demonstration was presented showing varied fidelity in the micromechanics models used throughout the FEM domain. Finally, it was shown that the multiscale model scales well with parallel processing and scales better the larger the RUC that is used.

Acknowledgments

This work is supported by the NASA Transformational Tools and Technologies (TTT) program.

References

- [1] Jones, R.M., *Mechanics of Composite Materials*, 2nd Ed., Taylor and Francis, Inc., Philadelphia, 1999.
- [2] Aboudi, J., Arnold, S. M., and Bednarczyk, B. A., *Micromechanics of Composite Materials: A Generalized Multiscale Analysis Approach*, Elsevier, 2013.
- [3] Haj-Ali, R., and Aboudi, J., “A New and General Formulation for the Parametric HFGMC Micromechanical Method for Two and Three-dimensional Multi-phase Composites,” *International Journal of Solids and Structures*, Vol. 50, 2013, pp. 907-919.
- [4] Carrera, E., Maiarú, M., Petrolo, M., and Guinta, G., “A Refined 1D Element for the Structural Analysis of single and multiple fiber/matrix cells, *Composite Structures*, Vol. 96, 2013, pp. 455-468.
- [5] Mori, T., and Tanaka, K., “Average Stress in Matrix and Average Elastic Energy of Materials with Misfitting Inclusions,” *Acta Metallurgica*, Vol. 21, 1973, pp. 571-574.