julia for Enhancing Nuclear Engineering Simulations (JENES): Introduction to the JENES Project and Platform

Muhammad Ramzy Altahhan, Gregory Delipei, David Holler
Jason Hou, Maria Avramova, Kostadin Ivanov

Department of Nuclear Engineering
North Carolina State University, Raleigh, NC 27695, USA

mraltahh@ncsu.edu, gkdelipe@ncsu.edu, dmholler@ncsu.edu
jhou8@ncsu.edu, mnavramo@ncsu.edu, knivanov@ncsu.edu

doi.org/10.13182/PHYSOR22-37830

ABSTRACT

Nuclear engineering education and research and development depend strongly on computer software and can stagnate if there is a large gap between theory and implementation. For this reason, we introduce a computational framework and platform to incorporate state-of-the-art coding paradigms and computer programs into nuclear engineering simulations. Since a computer code is characterized (first and foremost) by its programming language, we test a modern programming language called `julia` for its viability in nuclear engineering simulations and for the required state-of-the-art paradigm intended. `julia` is considered a Just-Ahead-Of-Time (JAOT) programming language that has elements of both Just-In-Time (JIT) and Ahead-Of-Time (AOT) compiled languages. `julia` has dedicated open-source modern mathematics, plotting, pre/post processing, Machine Learning (ML), Artificial Intelligence (AI), Automatic Differentiation (AD) and Optimization libraries that can be oriented together in tandem towards systematic computing. To compare `julia`'s capabilities against `Fortran` and `MATLAB`, we transferred a Nodal Expansion Method (NEM) code directly from `Fortran` to both `julia` and `MATLAB` and tested their computation speed using the IAEA-3D steady-state reactor simulation benchmark. The preliminary results show that the compiled `julia` code indeed has comparable speed to the `Fortran` version. We also demonstrated data plotting and pre/post processing using the `julia` code showing its edge in such tasks, leading to the possibility of both closed- and open-sourced complete nuclear engineering platform.

KEYWORDS: julia, Fortran, MATLAB, IAEA-3D, Nuclear Engineering, DNA

1. INTRODUCTION

Computational methods and simulations have been important in the field of nuclear engineering due to the difficulty and cost of conducting experiments in complex environments such as nuclear reactors. During the operation of a nuclear reactor many multi-physics and multi-scale phenomena occur that cannot always be separated. Nuclear engineers then mostly rely on mathematical equations implemented in computer codes to model and imitate real life designs and scenarios. For such purposes, the `Fortran` computer language is one of the major languages being used in nuclear engineering codes (in both industry and research institutes) and for fare reasons. First, `Fortran` is an Ahead-Of-Time (AOT) compiled language. This means the compiled code can be executed fast and an executable can be generated and can be shared without the source code being included, with both properties important for the industrial sector. Second, `Fortran` is a column-major order language. This means that dealing with arrays and matrices (which are the backbone of scientific computing) is easy and straightforward. Third, much of the legacy codes (that have been verified, and possibly validated) are written in `Fortran`.