INL/EXT-08-14884

High-Order Spatio-Temporal Schemes for Coupled, Multi-Physics Reactor Simulations

Vijay S. Mahadevan
Jean C. Ragusa

September 2008

INL/EXT-08-14884

High-Order Spatio-Temporal Schemes for Coupled, Multi-Physics Reactor Simulations

Vijay S. Mahadevan[1]
Jean C. Ragusa[1]

[1]Texas A & M

September 2008

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Through the INL LDRD Program
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

Final Technical Report:

High-Order spatio-temporal schemes for coupled, multi-physics reactor simulations

Prepared for Idaho National Laboratory

September 26, 2008

Sponsor Contract Number: 00072764
TEES Project Number: A1240

Mr. Vijay S. Mahadevan
Dr. Jean C. Ragusa

Department of Nuclear Engineering
Texas A&M University

Contents

1 Introduction

High-fidelity modeling of nuclear reactors requires the solution of a nonlinear coupled multi-physics problem. Safety analysis in nuclear reactors require nonlinearly consistent and accurate numerical schemes since the system of equations arising from the discretization of the various physics components is stiff and has widely varying time and length scales that need to be resolved correctly. A numerical method that converges the implicit nonlinear terms to a small tolerance is often referred to as nonlinearly consistent. This nonlinear consistency is still lacking in the vast majority of coupling techniques today.

Over the past decades, high fidelity modeling of nonlinear multi-physics problems has been subdivided into several distinct domains of physics and solved individually as mono disciplinary blocks without rigorous coupling between the different physics, a technique mathematically referred to as Operator-Splitting (OS). Although naïve, this is still the most widely used coupling strategy for nonlinear multi-physics simulations, including in reactor analysis, desing and safety. OS is based on coupling several existing specialized single physics codes with a "black-box" strategy, where the input of one code becomes the output of other, thereby producing solutions that are weakly coupled. More specifically, the OS method decomposes the system of PDEs into simpler sub-problems and solves the resulting system individually using specialized numerical schemes. The strategy is non-iterative and hence the nonlinearities in the system due to the coupling are not resolved, reducing the accuracy in the time stepping procedure to first order, even though high order time integration might be used in the individual physics components. Although it does allow parts of the problem to be treated implicitly, the lack of iterations over the nonlinear coupling terms leads to low accurate solutions. Despite these obvious drawbacks, this is still one of the major coupling paradigms used today for solving nonlinear coupled multi-physics systems.

The fundamental inefficiency and essential drawback of the OS strategy is that it is based on the explicit linearization of the coupled physics terms in the problem and, therefore, does not resolve nonlinearities between physics over a time step. Such an inconsistent treatment of the nonlinear terms usually results in a loss of the convergence order in the final solution and requires the use of excessively small time steps due to stability constraints and loss of convergence order. The direct implication of using smaller time steps to achieve a reasonable accuracy is that the computations needs in terms of CPU time and resources are large. On the other hand, the attractive feature of the explicit coupling strategy is that the legacy of many man-years of mono disciplinary, code development and Verification and Validation (V&V) are preserved.

In this report, an alternate scheme to OS, based on a Newton's method for the whole nonlinear system of equations, is presented and implemented. This Jacobian-Free Newton-Krylov (JFNK) method preserves and resolves the strong coupling between the physics components, such that high order accuracy in space and time are retained. The JFNK method enables the solution of the nonlinear equations without the need for the expensive Jacobian matrix. It has been proposed by Brown and Saad, in the early 1990's ([1]) and has enjoyed much success, see, for instance, the recent JFNK review paper by Knoll and Keyes ([2]).

The present work is a continuation of the work described in ([5]) and we present the implementation and verification of JFNK method for coupled physics applied to reactor analysis and safety. The KARMA computer code developed in that regard serves as a test bed code for methods development and contains the following features:

1. state-of-the-art computer science toolbox and libraries for efficient spatial discretizations, handling and interfacing of the various physics components, fast and robust linear algebra for parallel computing platforms;

2. coarse grain physics models for rapid testing and verification; finer grain models can replace the existing ones in a straightforward fashion through the common C++ interface.

A discussion of the algorithms used and the results from the implementation given in the next sections. Section 2 presents the coupling techniques currently in use, describes their drawbacks, and puts in context this work. Section 3 describes the development and implementation choices for KARMA, a test bed code for multi-physics applications in reactor analysis and safety. The numerical method is explained in Section 4; the various physics components are detailed in Section 5 and results are provided in Section 6.

2 Current coupling techniques in use for reactor analysis and the way forward

Traditional multi-physics coupled codes for reactor analysis problems employ extensively validated and verified efficient mono-physics codes that are coupled with either a PVM or MPI architecture to achieve the loose coupling in an OS methodology. Such "divide and conquer" methods provide flexibility in the usage (and re-use) of standard industrial codes and avoid replicating man years of development and testing. Examples of such existing coupled codes to analyze reactor transients are PARCS/TRACE and NESTLE/RELAP where an explicit coupling of the physics codes is performed by exchanging the solution fields from each physics at every time step as only boundary condition to the other physics. Nonetheless, these conventional coupling paradigms utilized to couple the different physics components in reactor analysis problems are inconsistent in their treatment of the nonlinear terms due to the explicit treatment of the coupling terms. Such methods without modifications have been proven ([5]) to be first-order accurate in time, i.e., $O(\Delta t)$, requiring considerably smaller time steps to obtain an accurate solution. We note that a few second-order OS techniques exist like Strang splitting methods ([7]), Marchuk splitting ([8]) but are somewhat intricate in their implementation.

Past research on the effect of the OS methods in terms of accuracy as compared to fully implicit coupling methods have been analyzed and documented ([4]). Mousseau ([6]) demonstrated that coupled problems with diffusion and two-phase flow using a consistent and accurate numerical scheme based on Jacobian-Free Newton Krylov (JFNK) framework is more effective to resolve the nonlinearities than traditional OS methods. Such a scheme will preserve the higher orders of accuracy in time integration in each nonlinear physics and the fully coupled solution. Also, since the length and time scales vary by orders of magnitude in such problems, an adaptive methodology in both space and time can be employed to efficiently resolve the solution evolution. Even though we have not yet performed any spatial adaptivity at this moment, our choice of library for spatial discretization allows such treatment.

An important factor for the consideration of a new code is that most of the mono-disciplinary codes were written one to three decades ago and to run on computers that existed during that period. Due to the current advances in computing, it would be imprudent to develop a new code for single processor machines but one must rather take advantage of the state-of-the-art multi-core, multi-processor parallel architectures that are available now and in the near future.

To overcome the issues stated above, a fully implicit treatment of the coupling terms is used to preserve accuracy and obtain unconditional stability. The difficulties in implementing such a scheme is that the spatial and temporal discretizations of all the physics need to be nonlinearly consistent. With such discretizations, the coupling terms in the physics are treated implicitly and higher order accuracy is ensured. Currently, no production codes exist with such capabilities and this is an effort in that direction. In the next section, the new code system is introduced; it is based on the JFNK framework with higher order spatio-temporal discretization of all the physics.

3 Current status of KARMA, a C(K)ode for Accident and Reactor Modeling Analysis

KARMA is a fully implicit coupled multi-physics transient analysis test bed code that could eventually be used to analyze reactor accidents. The rationale for KARMA is to provide a software environment in which

1. new coupling methodologies can be implemented and verified and
2. code architectures and software design for the next generation of safety analysis code can be tested.

KARMA's plug-in architecture employed makes it straightforward to modify it to add and coupled additional physics components and the framework can be used to seamlessly integrate the existing consistent numerical models with the new physics models that can be created. The idea behind this new code system is solving strongly coupled physics using loosely coupled software methodology. KARMA is written entirely in C++ making use of the advanced object-oriented features such as abstraction, encapsulation and inheritance, to create loosely coupled objects for seamless integration of new physics models. A prime concern in the code design is to achieve high level of efficiency while still maintaining the object oriented philosophy in mind. Careful planning in this aspect of the computational domain was done and a decision to use well tested linear algebra and other general purpose libraries was made in order to reduce the overhead in designed implementation. This also follows closely the principles of object and code re-use whenever possible thereby preserving man years of effort on code verification.

In the following sections, details on some of the supporting libraries such as PETSc, LibMesh, SLEPc, ParMETIS, GMSH, VISIT and the different components of KARMA are discussed.

3.1 Linear algebra: PETSc

Considering the current advances in computing, any state-of-the-art code needs to have some level of parallelism incorporated at its core. Based on this idea, a decision to use well tested parallel data structures provided by an external library such as PETSc ([9]) was made. Since PETSc has been used in several different fields and for various problems with tremendous success, it proves to be a perfect candidate to handle all the linear algebra needs by KARMA.

PETSc, the Portable, Extensible Toolkit for Scientific computation, was developed at Argonne National Laboratory, in the Mathematics and Computer Science Division ([10]), as a general purpose suite of tools for the scalable solution of partial differential equations and related problems. It provides sets of tools for the parallel (as well as serial), numerical solution of PDEs that require solving large-scale, sparse nonlinear systems of equations. It includes nonlinear and linear equation solvers that employ a variety of Newton techniques and Krylov subspace methods and provides several parallel vector formats and sparse matrix formats, including compressed row, block compressed row, and block diagonal storage.

PETSc is designed to facilitate extensibility. Thus, users can incorporate customized solvers and data structures when using the package. PETSc also provides an interface to several external software packages, including Matlab, ParMeTis, PVODE, and SPAI, and is fully usable from C and C++, and runs on most UNIX based-systems. Due to the advanced design, users can create complete application programs for the parallel solution of nonlinear PDEs without writing much explicit message-passing code themselves. Parallel vectors and sparse matrices can be easily and efficiently assembled through the mechanisms provided by PETSc. Furthermore, PETSc enables a great deal of runtime control for the user without any additional coding cost. The runtime options include control over the choice of solvers, preconditioners and problem parameters as well as the generation of performance logs. These options also can be used to manipulate whether a completely matrix-free approach (-snes_mf: Action of Jacobian and Preconditioner is matrix-free) is used to solve the coupled system or if a partially matrix-free approach is used (-snes_mf_operator: Action of Jacobian is matrix-free but the Preconditioner matrix is formed and stored in memory).

3.2 Spatial integration: LibMesh

One of the important aspects of a physics package depends strongly on its spatial treatment. Instead of creating a library from scratch, a generic Finite Element (FE) library called LibMesh ([11]), which is also written in C++ language has been made to interface with KARMA. LibMesh makes use of data structures exposed by PETSc and provides objects to handle and manipulate different types of Finite Elements, the unstructured mesh in the domain, the connectivity of the unknowns and association of material properties to each element. Currently, it is possible to use in LibMesh both Continuous Galerkin (CG) method with Lagrange basis functions and Discontinuous Galerkin (DG) method with Legendre basis functions. KARMA builds upon LibMesh to spatially resolve the parabolic problems using CG and hyperbolic problems using DG methods accurately with piecewise high-order polynomial basis functions. An example discretization for an elliptic and a hyperbolic system of equations is shown below.

3.2.1 Elliptic system: Continuous Galerkin discretization

Consider a diffusion-reaction PDE given by

$$-\bigtriangledown .(b(r)\bigtriangledown u) + c(r)u = f \tag{1}$$

Multiplying by a smooth function v (lagrange polynomial of order p) and integrating by parts over each element k in the finite element mesh Ω yields the following weak form of the problem

$$\sum_{elements(k)} \int_k \{v'bu' + vcu\} = \sum_{elements(k)} \int_k \{vf\} \tag{2}$$

and results in a linear system of equations of the form

$$(K + M + G)U = S + H \tag{3}$$

where K, M, S are the stiffness, mass matrices and load vector respectively. The matrix G and vector H arise from the boundary terms.

3.2.2 Hyperbolic system: Discontinuous Galerkin discretization

Consider a hyperbolic conservation equation of the form

$$\bigtriangledown.(F(u)) = f \tag{4}$$

Multiplying by a smooth function v (legendre polynomial of order p) and integrating by parts over each element k in the finite element mesh Ω yields the following weak form of the problem

$$\sum_{elements(k)} \left\{ -\int_k \{F(u) : v'\} + \int_{\partial k} \{F(u).nv\} \right\} = \sum_{elements(k)} \int_k \{vf\} \tag{5}$$

The flux $F(u).n$ is be replaced by a numerical flux function. In the current work, the (local) Lax-Friedrichs flux function is used. The DG method can then be written as

$$\sum_{elements(k)} \left\{ -\int_k \{F(u) : v'\} + \int_{\partial k} \left\{H(u^+, u^-, n)v^+\right\} \right\} = \sum_{elements(k)} \int_k \{vf\} \tag{6}$$

where the (local) Lax-Friedrichs flux $H(u^+, u^-, n)$ is

$$H(u^+, u^-, n) = \frac{1}{2}\left\{F(u^+).n + F(u^-).n + \lambda(u^+ - u^-)\right\} \tag{7}$$

with $\lambda = \max\{eig(Jacobian)\}$.

3.2.3 Advanced capabilities

It is also possible to perform Automatic Mesh Refinement (AMR) using LibMesh, when necessary, based on the needs of the physics being solved in KARMA. Also in order to make use of parallelism, LibMesh provides built in domain decomposition routines for structured meshes and interfaces to use external libraries such as ParMetis to partition unstructured meshes. This translates to performing parallel assembly of stiffness, mass matrices and source vectors that are necessary to solve the linear system arising from discretization of the PDE, thereby reducing overall computational times.

3.3 Eigensolver: SLEPc and ARPACK

For the purpose of solving generalized eigenvalue problems that occur in nuclear reactor analysis problems, KARMA has been configured to interface with SLEPc library ([12]), which implements several standard eigenvalue solvers. The power iteration, shifted-inverse-power iteration and subspace iteration methods implemented in this library ensure efficient convergence to the fundamental eigenvalue (criticality).

Alternately, another library named ARPACK ([13]) which implements the powerful and robust Arnoldi algorithm is also interfaced through SLEPc. The original Arnoldi algorithm proposed by Saad, is a powerful extension of the subspace iteration in that it builds an orthonormal basis of the Krylov subspace and factorizes the matrix to an upper Hessenberg matrix. The central idea behind the Arnoldi factorization is to construct eigenpairs of the original matrix from eigenpairs of the factorized, small, Hessenberg matrix. Sorensen (1992) improved the Arnoldi method by introducing several shifted QR iterations on the Hessenberg matrix which reduces the overall number of required matvec operations. Several other variations and optimizations have been added to this Arnoldi method and has been successfully implemented in ARPACK. The Implicitly Restarted Arnoldi Method (IRAM) is considered one of the state-of-the-art scheme to compute several dominant eigenpairs for large, sparse linear systems.

3.4 Temporal integration: Runge Kutta schemes

A generic temporal integrator was written based on KARMA framework, making use of the description of a scheme purely based on its Butcher-Coefficient matrix and associated coefficients. Currently, KARMA can handle Explicit RK (ERK) and Diagonally Implicit RK (DIRK) methods and future efforts will include Fully Implicit RK (FIRK) schemes also. Since the integrator is generic enough, arbitrary orders of temporal accuracy can be obtained using higher order RK schemes without any modifications in the user code. A brief description of an s-stage Runge-Kutta method (s functions evaluations per step) can be written as follows:

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i \tag{8}$$

with

$$k_i = f\left(t_n + hc_i \ , \ y_n + h \sum_{j=1}^{s} a_{ij} k_j \right) \tag{9}$$

The method's parameters are often represented using a Butcher tableau:

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array}$$

where $b = [b_1, \ldots, b_s]^T$, $c = [c_1, \ldots, c_s]^T$, and $A = [a_{ij}]$. These parameters define the method's characteristics (explicit vs implicit, convergence order, stability properties etc...). Hence to completely describe a new RK scheme, a user needs to only specify these coefficients at run-time and the temporal integration scheme will be used to step forward in time, without any further plumbing from user standpoint.

Previously ([3]), more than 70 Runge-Kutta methods were analyzed in terms of

1. explicitness/implicitness,
2. absolute stability (i.e.,the domain S such that $S = \{z \in \mathbb{C}; |R(z)| \leq 1\}$ where R is the method's characteristic polynomial),
3. L-stability (i.e., $\lim\limits_{z \to \infty} R(z) = 0$ in order to avoid oscillatory behaviors for large time steps)
4. coherence of the embedded feature (i.e., the stability regions of both pairs should be identical).

Based on this analysis, future improvements to the temporal integrators will be included to use more robust implicit schemes as compared to DIRK and SDIRK schemes that are currently implemented.

3.5 Meshing capability: Gmsh

Gmsh ([14]) is an automatic three-dimensional finite element mesh generator with a built-in CAD engine and pre-, post-processing functionalities. Its design goal is to provide a simple meshing tool for academic problems with parametric input and advanced visualization capabilities. All geometrical, mesh, solver and post-processing instructions are prescribed either interactively using the graphical user interface (GUI) or in ASCII data files using Gmsh's own scripting language. Interactive actions generate language bits in the input files, and vice versa. This makes it possible to automate all treatments, using loops, conditionals and external system calls. KARMA uses LibMesh's capability to read and write Gmsh format files (.msh). This enables KARMA to solve on problem domains with arbitrary geometries that can be represented with Gmsh, along with associated material numbers for each region, and create a conforming finite element mesh using triangles or tetrahedrons in 2-D and 3-D respectively.

Alternately, with few extensions, KARMA could be made to use the open source softwares namely Triangle (2-D) and TetGen (3-D) since the formats used by these libraries are quite straightforward.

3.6 Visualization: VisIt

VisIt ([15]) is a free interactive parallel visualization and graphical analysis tool for viewing scientific data on Unix and PC platforms. Users can quickly generate visualizations from their data, animate them through time, manipulate them, and save the resulting images for presentations. VisIt contains a rich set of visualization features so that you can view your data in a variety of ways. It can be used to visualize scalar and vector fields defined on two- and three-dimensional (2D and 3D) structured and unstructured meshes. LibMesh provides handles to read and write mesh and solution fields in the text and binary formats for both structured and unstructured meshes. VisIt is known to have rich feature set for scalar, vector, and tensor field visualization and provides powerful, full-featured graphical user interface (GUI).

3.7 Domain decomposition: ParMETIS

ParMETIS ([16]) is an MPI-based parallel library that implements a variety of algorithms for partitioning unstructured graphs, finite element meshes, and for producing fill-reducing orderings of sparse matrices. The algorithms implemented in ParMETIS are based on the multilevel recursive-bisection, multilevel k-way, and multi-constraint partitioning schemes. ParMETIS includes routines that are especially suited for parallel AMR computations and large scale numerical simulations which can be used in conjunction with LibMesh and PETSc.

3.8 Other toolboxes/libraries

TinyXML: This is a small C++ library that can handle reading, manipulation and writing of XML data with very little memory overhead. KARMA will use input decks in the form of XML files for convenience since most of the user input errors commonly seen in traditional codes with large input files can be avoided.

CSVParser: This is another supporting C++ library that acts as a parser to read and write Comma Separated Values (CSV) to aid in reading data from spreadsheets like Excel or from data exported from MATLAB (csvwrite, csvread).

3.9 Schematics of KARMA

A brief schematic of the overall interaction of KARMA with other supporting libraries is shown in Fig. 1. The overview figure clearly indicates the global interaction of KARMA library with an User Interface (UI) which is the primary driver interacting with the end user i.e., in this case, a reactor designer or analyst. KARMA library itself makes use of the PETSc parallel library for the linear algebra data structures such as Matrix, Vectors, Distributed arrays and several PETSc modules that handle linear solvers, nonlinear solvers, numerical preconditioners and optionally even temporal integration. KARMA also interacts with other libraries such as LibMesh for spatial integration, SLEPc and ARPACK for eigenvalue problems, ParMetis for parallel domain decomposition, VTK for visualization and TinyXML for input processing.

Also internally, KARMA has several different modules namely

1. PHYSICS: Handles the description of the physics models by discretizing the corresponding PDE in space and time
2. NUMERICS: Handles discretization and provides data structures to accurately resolve the different individual physics components
3. INTERFACE: Handles the coupling between the different physics in a generic fashion and serves as the primary rendezvous point for User Interfaces, if any.
4. IO: Handles all input data processing and output data manipulation in various formats in a generic fashion.

A more descriptive schematic of the interaction between these different modules inside KARMA is shown in Fig. 2. In the next section, a closer look at the time integration carried out using the generic Temporal integrator in KARMA is described.

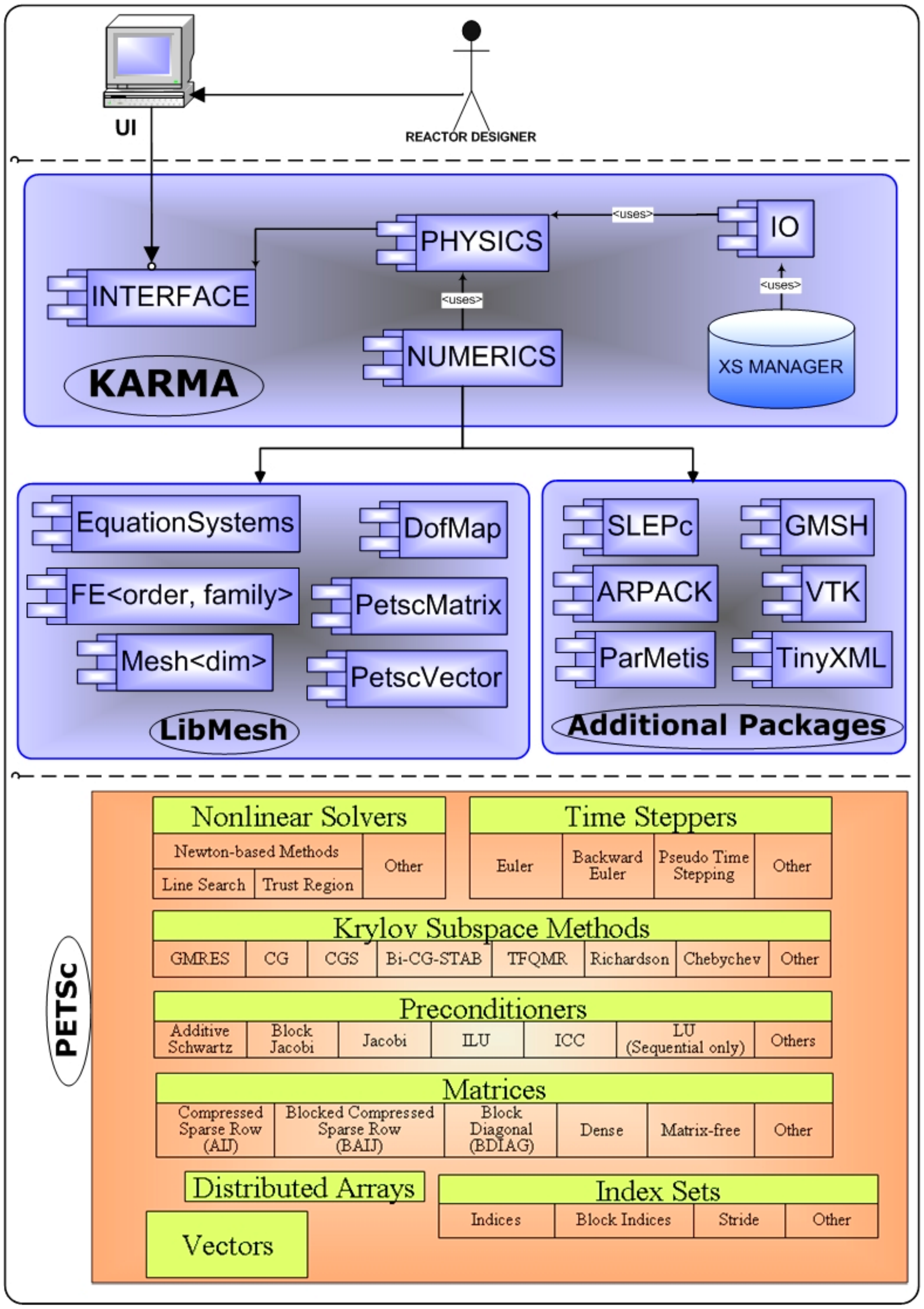


Figure 1: Interaction of KARMA with other packages and libraries.

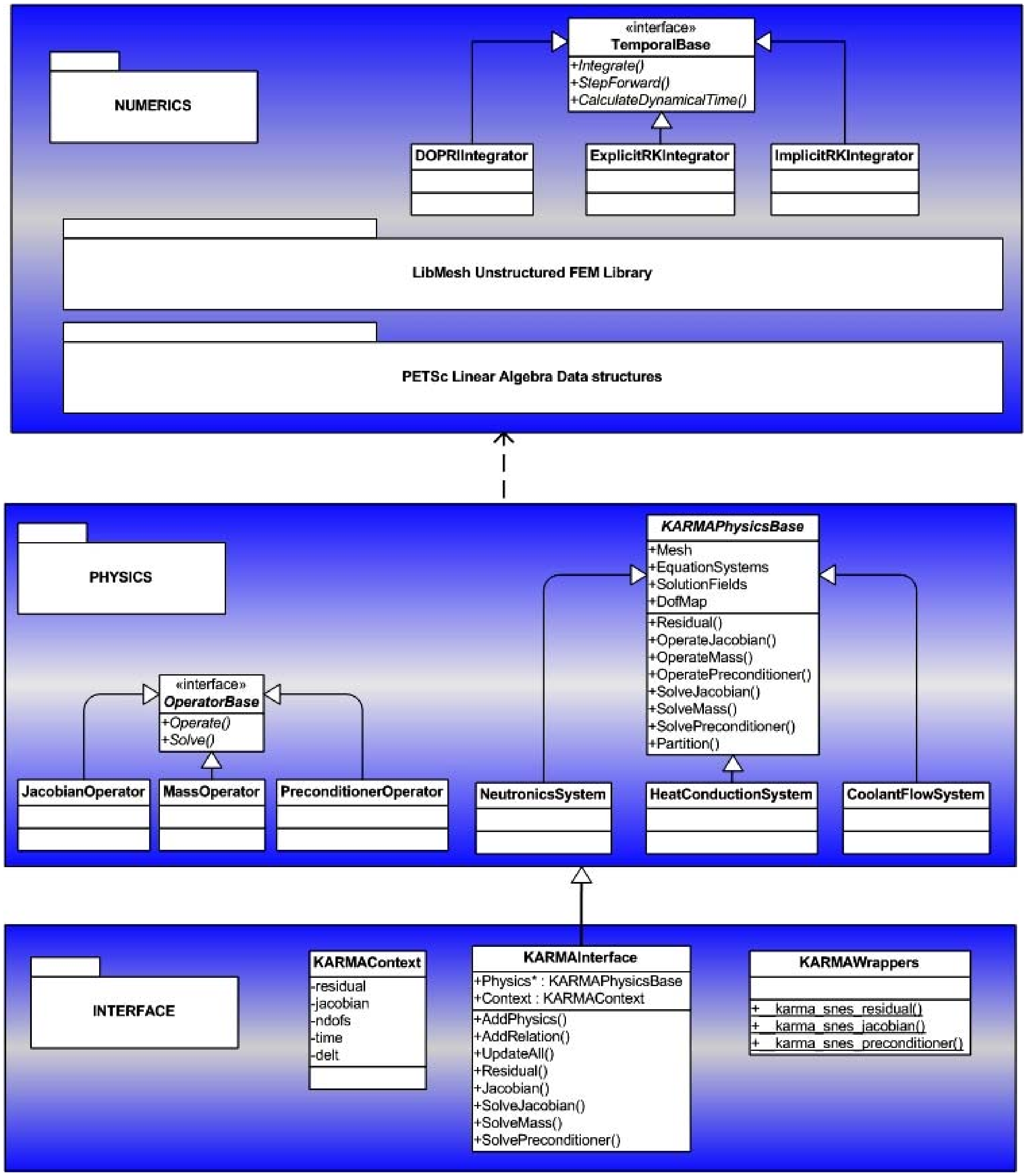


Figure 2: KARMA Package interaction schematic.

4 Jacobian-Free Newton Krylov (JFNK) Algorithm

4.1 Introduction

The JFNK method employs Newton's method (outer nonlinear solves) and Krylov methods (inner linear solves) to solve a set of nonlinear equations effectively and accurately. The Jacobian-Free approximation implies that the algorithm can be implemented without explicitly building the Jacobian matrix needed in the linear solve. Often, building the Jacobian matrix can be costly in CPU time and memory, especially when different physics components reside in multiple codes. The matrix-free nature of the solvers relies on (i) the fact that Krylov solvers build a solution subspace using matrix-vector operations and (ii) these matrix-vector operations can be approximated using a finite difference formula that does not require knowledge of the matrix at all. Nevertheless, Krylov methods may require a certain number of basis vector to be stored in order to find an accurate solution (i.e., the size of the subspace may be large). The Krylov space size and the overall computing time can be significantly reduced by the use of an appropriate preconditioner. Therefore, the JFNK algorithm consists of 3 levels of iterations: 1) Newton iteration 2) Krylov iteration 3) Preconditioner iteration. Let us now briefly look into the equations involved in implementing the algorithm.

4.2 Newton's Method

Let us consider a system of nonlinear equations of the form $F(y) = 0$, e.g., obtained by implicit time differentiation of a system of ordinary differential equations after applying spatial discretization. Newton's method iteratively seeks a root as follows:

$$\begin{aligned} J(y^k)\delta y &= -F(y^k) && (10) \\ y^{k+1} &= y^k + \delta y && (11) \end{aligned}$$

where $J(y^k) = \frac{\partial F(y^k)}{\partial y}$ is the Jacobian matrix of the system at the current Newton iterate y^k, δy is the increment update, solution of the linear solve, and the next Newton iterate is given by y^{k+1}.

It is quite clear that the above equation requires forming the Jacobian matrix explicitly in order to solve the system for δy. This can be expensive in terms of computational time if a finite difference procedure by perturbing $F(y)$ is used to find J element by element (numerical Jacobian). Alternately, a different algorithm can be employed in the linear solve such that the Jacobian matrix itself is not required but only its action on a given vector. This operation will be required during the linear solve of (10). For a given vector v, the action of the Jacobian on the vector can be computed using the following equation

$$Jv \approx \frac{F(y + \epsilon v) - F(y)}{\epsilon} \tag{12}$$

where ϵ is a parameter used to control the magnitude of perturbation. Optimal equation for choosing the perturbation parameter ϵ has also been derived in reference paper ([2]). The class of linear solvers to be employed in the matrix-free algorithm are the Krylov solvers. Since the Jacobian matrix is non-symmetric, a natural choice for the Krylov linear solver is the GMRES algorithm.

4.3 Krylov Method: GMRES

The GMRES (Generalized Minimum RESidual) method, introduced by Saad and Schultz ([17]), is a popular and efficient Krylov subspace method used to solve nonsymmetric system of equations. The GMRES algorithm generates a sequence of orthogonal vectors, and because the matrix being inverted is not symmetric, short recurrence relations cannot be used as in the case of the Conjugate Gradient algorithm. Instead, all previously computed vectors in the orthogonal sequence have to be retained. One matrix-vector product is required per iteration and the matrix-free approximation introduced earlier in (12) can be used to create the Jacobian-Free framework. Detailed information on the exact numerics and implementation of GMRES in the JFNK framework has been shown in ([2]).

4.4 Preconditioners for the Linear Iteration

As introduced earlier, highly nonlinear problems typically possess a preconditioning loop inside every GMRES iteration. A right-preconditioning technique applied to the above Jacobian-Free Newton framework yields

$$\text{Solve for } w: \qquad (JP^{-1})w = \quad -F \tag{13}$$

$$\text{Solve for } \delta y: \qquad \delta y = P^{-1}w \tag{14}$$

where P is the preconditioning matrix.
The preconditioner P is usually a good approximation of the Jacobian and should be easier to form and solve as compared to the Jacobian matrix itself. Then, the inherently two-step process shown above requires the computation of the action of P^{-1} on any vector v, rather than actually forming the preconditioning matrix itself. Such an algorithm would strictly be "matrix-free" and studies for higher dimensional real-world problems can be conducted to determine the efficiency of this approach.

The preconditioned JFNK algorithm can be explained as follows: the operation of Eq.(13) requires the action of JP^{-1} on any Krylov vector v and has to be done at each GMRES iteration; it is performed in two steps:

1. Preconditioning: solve for x in $Px = v$
2. Perform the matrix-free product Jx $(= JP^{-1}v)$ by using the finite difference formula (12)

Only the matrix elements required for the action of P^{-1} are formed. There are two primary choices to be made:

1. What linearization should be used to form the matrices required in P^{-1}?

   Based on the physics, depending on which modes are dominant, the linearization should implicitly handling the dominant modes while treating the other modes in a linearized fashion. This will reduce the spread of the eigenvalues of JP^{-1} thereby creating the desired effect in the Krylov iteration (reduce the total number of required iterations). An example of the physics-based preconditioning technique which does precisely the above mentioned idea was implemented by Mousseau et al. ([18]) and was proven to be efficient in reducing the total number of linear iterations, during a newton solve. Another example is the Implicit Continuous fluid Eulerian (ICE) scheme introduced by Harlow and Amsden ([19]) that treats the acoustic waves implicitly while treating the advection terms explicitly, which acts as an efficient preconditioner for low-mach flow problems.

2. What linear iterative method should be used for $y = P^{-1}v$?

   For nontrivial multidimensional problems, iterative methods would be the best option instead of a direct solve using Banded solvers or by Gaussian elimination. An obvious option would be to use GMRES again to solve the preconditioner problem in a matrix-free way in which case, we would only require action of P on a vector v. This form is recursive and care is needed to implement this framework in a modular fashion.

More discussion on physics-based preconditioning will be included below, when dealing with individual physics components.

5 Physics models

The physics of nuclear systems are usually sub-divided into 4 primary domains for extensive and rigorous calculations based on the nature of the physics. They are given as:

1. Neutronics - Description of the neutron population distribution in the reactor core as a function of position, time, energy and angle.
2. Fuel heat conduction - Description of the temperature fields within the nuclear fuel pin
3. Coolant flow - Description of the coolant flow fields that include density, momentum and total energy
4. Structural mechanics - Description of the effects of expansion in the fuel pins and structures in the core

The aim of the current work is primarily focused on testing out accurate numerical methods for coupled multi-physics calculations. Hence, a decision was made to use coarse grain fidelity physical models rather than investing additional effort into the derivation of high-fidelity models each of the physics component. The architecture of KARMA allows for straightforward addition of new physics models and should be considered in the future. With this in view, coarse grained physical models were chosen for each of the dominant physics (Neutronics, Heat conduction, Coolant flow) and have been solved in conjunction with the numerical methods already introduced. A brief discussion on the physics models is given below.

5.1 Neutron diffusion model

Neutronics is the branch of physics that deals with the calculation of neutron flux and neutron reaction rates in the different materials inside the core. It is required to determine accurately the spatial distribution of the nuclear power, for subsequent treatment as conduction in the fuel pellets and convection in the coolant. In turns, the temperature solution fields in the fuel and coolant will feedback into the neutronics model via the temperature dependence of the cross sections.

The energy production in a mesh cell K is given by

$$Q(r,t) = \int_K dV \sum_{g=1}^{G} dE \; [\kappa_{f,g}\Sigma_{f,g} + \kappa_{c,g}\Sigma_{c,g}]\,(r,t)\phi_g(r,t) \tag{15}$$

where $\Sigma_{f,g}$ is the fission cross section in group g, $\Sigma_{c,g}$ is the capture cross section in group g, the κ coefficients are the amount of energy released per reaction event, and $\phi_g(r,t)$ is the neutron scalar flux in group g.

Our neutronics model is based on the multigroup diffusion equation for homogenized fuel assemblies whose solution provides the neutron flux

$$\begin{aligned} \frac{1}{v}\frac{\partial\phi_g(r,t)}{\partial t} - \bigtriangledown . D_g(r,t) \bigtriangledown \phi_g(r,t) + \Sigma_{t,g}(r,t)\phi_g(r,t) \quad = \quad & \sum_{g'=1}^{G} \Sigma_{s,g'\to g}(r,t)\phi_{g'}(r,t) \\ & + \chi_{p,g} \sum_{g'=1}^{G} (1-\beta)\nu\Sigma_{f,g'}(r,t)\phi_{g'}(r,t) \\ & + \sum_{j=1}^{J} \chi_{d,g,j}\lambda_j C_j(r,t) \end{aligned} \tag{16}$$

For clarity, the diffusion equation can be expressed in operator notation

$$\frac{1}{v}\frac{\partial \phi}{\partial t} = (F_p - M)\phi + S_d \tag{17}$$

where
- F_p is the prompt fission source;
- S_d is the delayed neutron source;
- M is the net removal of neutrons via absorption and scattering plus net leakage of neutrons to other points in the reactor;

The delayed neutron source results from the radioactive decay of the precursors. Assuming that there are j precursor groups, with respective decay constants λ_j, we can then write the delayed neutron source as

$$S_{d,g}(r,t) = \sum_{j=1}^{j} \chi_{d,g,j}\lambda_j C_j(r,t) \tag{18}$$

where $\chi_{d,g,j}$ is the delayed neutron emission spectrum (different from the prompt neutron emission spectrum $\chi_{p,g}$). The precursor concentrations are given by the precursor depletion equations as

$$\frac{\partial C_j(r,t)}{\partial t} + \lambda_j C_j(r,t) = \beta_j \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(r,t)\phi_{g'}(r,t) \tag{19}$$

Even though this multi-group diffusion model is easier to solve than a full blown transport model, it still can be complex in multi-dimensional problems and provides a reasonable model for the flux distribution on a reactor.

For the multi-group diffusion model solved on homogenized fuel assemblies, it is necessary to generate multigroup cross sections from lattice physics codes such as CASMO or TransLAT, which solve with a high degree of fidelity the original fuel assembly geometry using a 2D infinite lattice transport model. The averaged cross sections are then obtained so as to preserve the reaction rates in the coarse grain model. An possible future step could be to replace the 3-D neutron diffusion model with a neutron transport model to capture a finer description of the physics.

5.2 Thermal heat conduction model

Heat conduction physics deals with the conduction of the energy deposited in the fuel element and transfered to the coolant flowing around the fuel pin. The energy released by the nuclear fission reactions appears primarily as kinetic energy of various fission products. The bulk of this fission product energy is rapidly deposited as heat in the fuel material, very close to the location of the fission event. The heat is then transported via thermal conduction across the fuel element, through the gap separating the fuel from the clad, and then across the clad to the outer fuel pin surface. It is then transferred from the clad outer surface to the coolant by forced convection. The bulk motion of the coolant then carries the thermal energy up and out of the reactor core, either as sensible heat (i.e., coolant temperature rise in PWR) or as latent heat (i.e., thermally induced phase change by boiling in BWR). Finally, radiative capture reactions also lead to some energy release in the fuel element. The entire contribution is given by the $Q(r,t)$, previously defined in Eq. 15. We note that a fraction of the radiative capture energy (as well a direct gamma heating) is also released in the coolant channels, we account for this in the description of the coolant flow physics in the next section.

The temperature field in the fuel element is obtained by solving the heat conduction equation:

$$\rho C_p \frac{\partial T_f(r,t)}{\partial t} - \bigtriangledown \cdot (k(T_f) \bigtriangledown T_f(r,t)) = Q(r,t) \tag{20}$$

with $k(r,T)$ the temperature-dependent thermal conductivity, leading to a nonlinear parabolic equation for the heat conduction alone.

In the current coarse grain model, the heat conduction equation is not solved for fuel pins but on an homogenized fuel assembly, i.e., using the same geometrical description as for the neutron diffusion. This allows for a simpler testing of the coupling between physics, where meshes are identical in both physics (neutronics and heat conduction). The future step to obtain a finer grain model would be to model one average fuel pin per assembly in $r - z$ coordinates.

5.3 Coolant flow model

The coolant flowing in a channel, outside the clad of the fuel pin gains enthalpy by convection and removes the heat from the fuel elements. The thermal hydraulics physics and heat conduction are nonlinearly coupled due to the heat transferred from the fuel to the coolant by means of forced convection. The temperature of the coolant is directly dependent on the temperature of the outer clad surface, which, in turn, is a direct function of the fission reaction rate thereby making all physics coupled to one another. In addition, a volumetric heat source can be used in the bulk of the coolant to model radiative capture energy release and direct gamma heating.

The current work only involves the solution of a single phase flow system in 1-D, where multiple 1-D channels can be used. We start by introducing the 1-D set of Navier-Stokes equations which describe the flow of coolant in a single channel. We can employ multiple flow channels per fuel assembly if needed but the initial work only deals with a single channel (average channel) per assembly.

The governing equations for the fluid flow in terms of the conservative set of variables are then given as

$$\frac{\partial \rho}{\partial t} - \bigtriangledown \cdot (\rho u) = 0 \tag{21}$$

$$\frac{\partial \rho u}{\partial t} - \bigtriangledown \cdot (\rho u^2) + \bigtriangledown P = \bigtriangledown \tau \tag{22}$$

$$\frac{\partial \rho E}{\partial t} - \bigtriangledown . (u(\rho E + P)) = \bigtriangledown \cdot (u\tau) + \bigtriangledown q + S \tag{23}$$

where $\rho-$ density, $\rho u-$ momentum, $\rho E-$Total energy, $P-$ Pressure, $\tau-$ viscous stress ($f_w \|u\| u$), f_w- wall friction factor, $q-$ thermal heat flux in the fluid, $S-$ external source terms (energy from fuel pin).

$$P = f(\rho, \rho e) \tag{24}$$

where $\rho e-$ internal energy $= \rho E - \frac{1}{2}\rho u^2$.

This system of equations are stiff in the flow regimes of concern in nuclear reactors where low Mach flow dominates. As the flow velocity of the fluid decreases, it is very difficult or impossible to solve low speed flows with a conventional compressible algorithm because of slow convergence. The difficulty in solving the compressible equations for low Mach numbers is associated with the large disparity between the acoustic wave speed and the waves propagating at the fluid speed, which is more generally represented as eigenvalue stiffness.

The discretization of this set of equations can be performed using DG in space and the higher order RK schemes in time. Recent work by Van Leer [20] and Nourgaliev et al [21] on a higher order extension of the Piecewise Parabolic Method (PPM), that is popular in the Finite Volume community, for diffusion operators in the Navier-Stokes equations systems promise improved orders of accuracy than from traditional DG method. For details on the rDG scheme, the reader is directed to papers by Van Leer and Nourgaliev.

5.4 Adding a new physics model in KARMA

The core principles in KARMA are based on the fully implicit treatment of each physics in the JFNK framework. The advantage of this design decision is that minimal requirements are mandated from a coding perspective for each of the physics. Each physics requires to implement only few main functions namely

1. Residual($F(u)$): Calculating the nonlinear residual for the physics occurring from the discretization of the PDE; This is the primary requirement and should completely describe a single physics.
2. OperateJacobian($J(u)v$): Action of the Jacobian of the nonlinear system on a given vector (v); If the Jacobian can be formed easily, this method may be an alternative to the matrix-free approximation in Eq. (12).
3. OperatePreconditioner(Pv): Action of the Preconditioner, which resembles the Jacobiam matrix, on a vector (v); This is useful if the preconditioner matrix is solved in a matrix-free fashion.
4. SolvePreconditioner($P^{-1}v$): This is the solution of a preconditioner system for a given vector (v); Here the preconditioner matrix is explicitly formed and then solved using one of PETSc's linear Krylov solvers.

Hence, any new physics in KARMA requires only 3 primary operators

1. Jacobian Operator: Primary Residual and Jacobian for nonlinear system;
2. Mass Operator: Arising from temporal discretization of the PDE;
3. Preconditioner operator: A preconditioner resembling the Jacobian;

And each operator requires 2 methods:

1. Operate: The action of the operator on a vector
2. Solve: The inversion of the operator acting on a vector

The idea is that a single physics can have an array of preconditioner operators and hence depending on the problem or transient being simulated, an appropriate choice can be made by the user. For instance, a point-Jacobi preconditioner can be used if the system is linear and reaction physics dominate while a linearized physics based preconditioner is effective when solving a nonlinear physics coupled to other physics.

These guidelines can be followed to implement, say, a $r - z$ fuel pin heat conduction model to replace the current implementation with homogenized assembly representations. Alternately, finer neutronics models can also be programmed based on the mature P_n or S_n methods for neutron transport.

6 Results

The results section is organized as follows: Firstly, each of the physics model is tested for spatial and temporal consistency (convergence order); Secondly, a coupled neutronics/heat-conduction problem is simulated and accuracy of the coupling effects are discussed.

6.1 Neutronics physics: Steady state problems - Eigenvalue calculation

Generalized eigenvalue problems occur often in reactor analysis and the fundamental mode (mode associated with largest eigenvalue in magnitude) is indicative of the reactor criticality configuration. But, while simulating problems like BWR instabilities, the influence of the lower modes can be significant and can play an important role during the transient. Although such a technique is promising in predicting the instability well, the computational cost for obtaining several eigenmodes and eigenvalues for analysis of large discretized coupled problems can be prohibitive. This is primarily due to lack of cost-effective schemes to determine higher modes other than fundamental mode accurately, when the number of required modes becomes large.

In the past several years, alternatives to power iteration and subspace iteration techniques, such as Jacobi-Davidson method and Arnoldi's method with implicit deflation for computing several eigenvalues, have been proposed and furthered by several researchers such as Saad (1992) [22] and Lehoucq, Sorenson (1996) [23] . In the current work, the Neutron eigenvalue problem is solved primarily using Implicitly Restarted Arnoldi Method (IRAM) which has been implemented in ARPACK library. The central idea behind the Arnoldi factorization is to construct eigenpairs of the original matrix from eigenpairs of the factorized, small, Hessenberg matrix with very few required number of matvec operations.

Generalized algebraic eigenvalue problems usually result from the discretization of multi-group neutron transport or diffusion equation in reactor analysis problems. The lambda modes equation is of the form

$$L\phi = \frac{1}{k} P\phi \tag{25}$$

where L is the Loss operator containing leakage, absorption and scattering terms, P is the Production operator containing fission terms.

Based on this approach, a BWR model benchmark problem was tested with a fine unstructured mesh using IRAM. The 2-D domain is discretized using triangular mesh elements. The spatial discretization of the generalized eigenvalue problem is performed using piecewise linear, Lagrange elements on triangles. The results presented below for the eigenvalue computation were from a discretization with 40960 elements and 20737 dofs/group. The results obtained for the first five fast and thermal group profiles are shown below in Fig. 3, 4, 5. It has also been found that the eigenvalues and eigenmodes converge quadratically in space to the true solution as we refine the mesh, as expected from theoretical considerations.

In this context, a scheme based on Inexact Newton [24] has been formulated to solve eigenvalue problems by treating it as a weakly nonlinear system. This method has proven to be successful and is based on the JFNK framework that has already been implemented in KARMA.

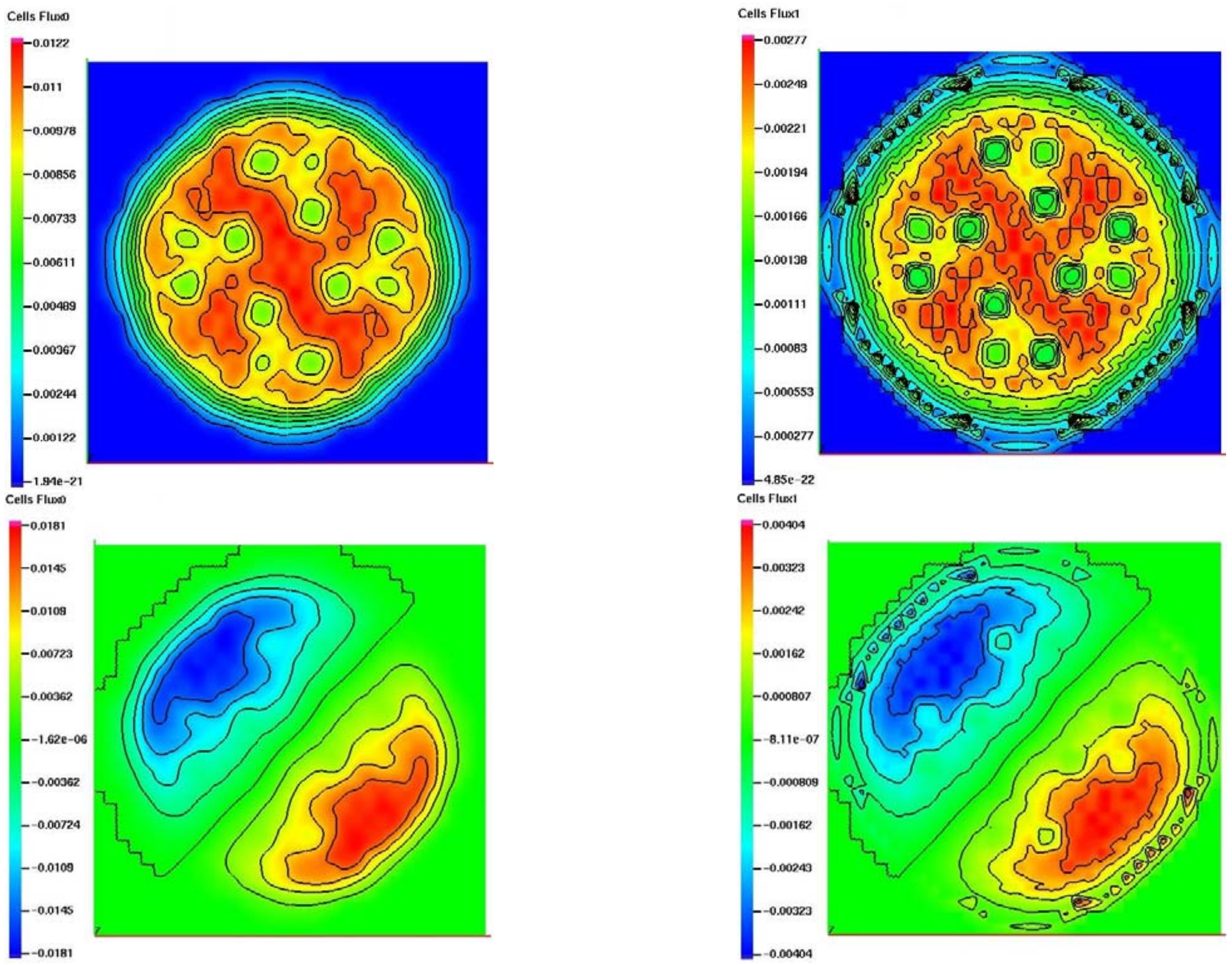


Figure 3: First two eigenmodes for a BWR stability test; fast fluxes (left), thermal fluxes(right).

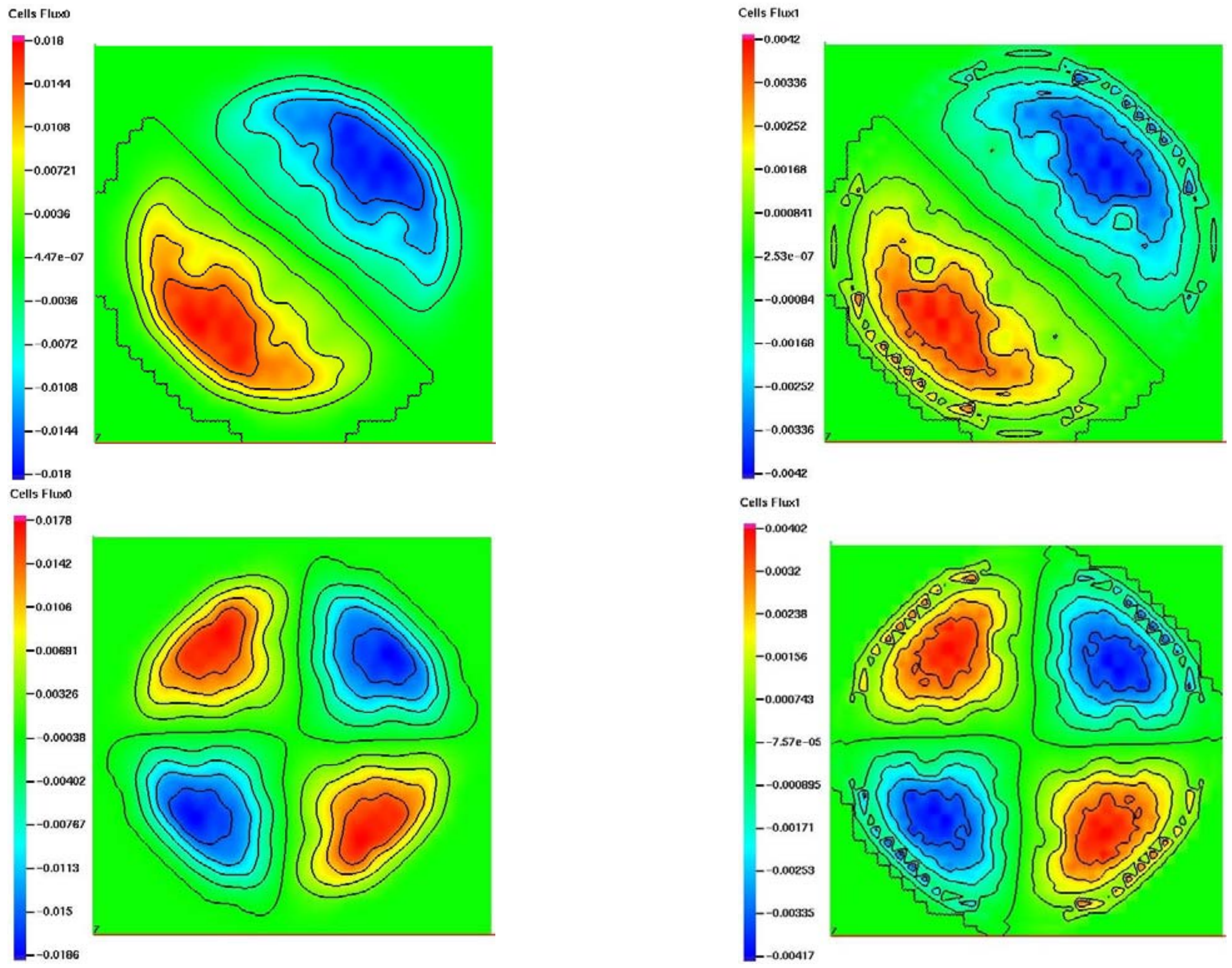


Figure 4: Third and fourth eigenmodes for a BWR stability test; fast fluxes (left), thermal fluxes(right).

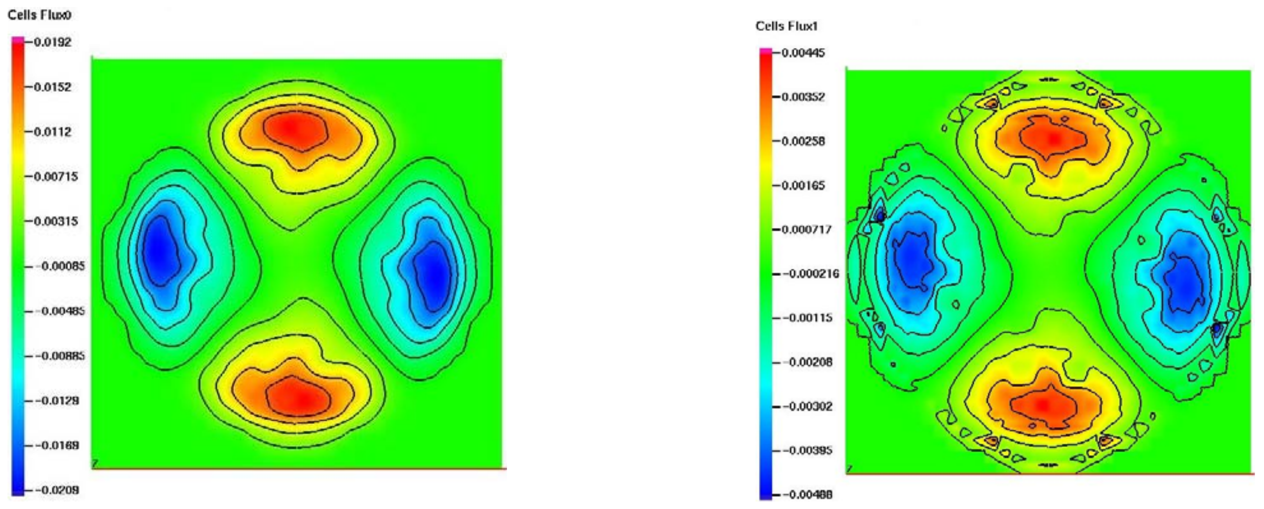
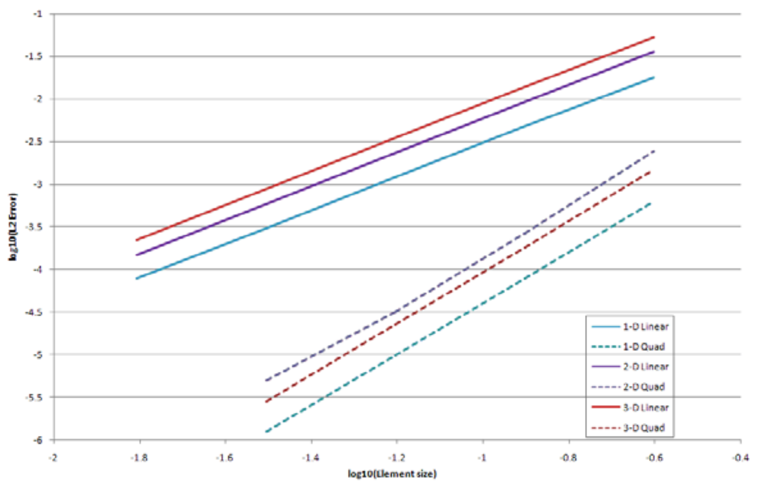


Figure 5: Fifth eigenmode for a BWR stability test; fast fluxes (left), thermal fluxes(right).

6.2 Nonlinear Heat conduction problem

Using the MMS method, a 1-, 2-, and 3-D test problem, with a diffusion coefficient varying as $k(T) = T^2$, has been devised, where the exact solution is taken to be $T(r,t) = \tanh(t)\prod_{i=1}^{dim}\sin(\pi r_i)$ where $r = [x, y, z]$. Since the exact solution is known, the spatial and temporal error discretization can be quantified and the solution methodology can be verified to be consistent.

The order of accuracy plots for space with piecewise continuous Lagrange shape functions up to $p = 3$ are shown in Fig. 6 (top). With a fine resolution of the mesh, the order of accuracy plot for time with Backward Euler, Crank Nicolson and SDIRK schemes were then found and plotted in in Fig. 6 (bottom). It can be seen that the nonlinear solution method with JFNK framework is high order accurate in space and time and is consistent with the expected theoretical orders of accuracy based on the discretization.



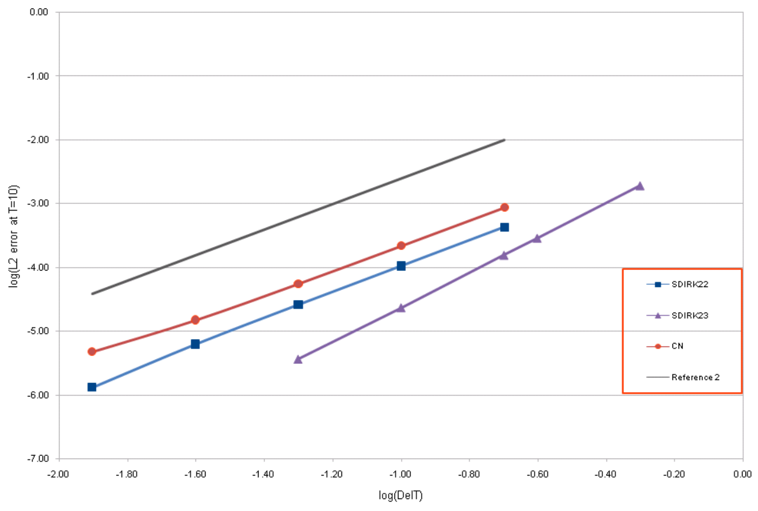


Figure 6: Spatial accuracy (top), temporal accuracy (bottom), conduction.

6.3 Coolant flow problem

Using the MMS, profiles for the state variables are assumed and accordingly forcing functions for the continuity, momentum and energy equations are found. The exact solution assumed for density, velocity and total energy are shown below.

$$\rho = \rho_{\min} + (\rho_{\max} - \rho_{\min}) \operatorname{sech}(\frac{x - \varpi t}{\delta}) \tag{26}$$

$$v = v_{\min} + (v_{\max} - v_{\min}) \operatorname{sech}(\frac{x - \varpi t}{\delta}) \tag{27}$$

$$E = E_{\min} + (E_{\max} - E_{\min}) \tanh(\frac{x - \varpi t}{\delta}) \tag{28}$$

Using these exact solutions, the source functions were found and the solution procedure was implemented to check the order of convergence in space and time. The closure relation for the pressure equation was chosen to be the ideal gas equation represented by

$$P = \rho e(\gamma - 1) \tag{29}$$

where $\gamma = \frac{C_p}{C_v}$.

Let us now look at the convergence results for the nonlinear Euler equations, using JFNK method. The profiles of the solutions as a function of time for the density, velocity and energy solution fields are shown in the figures 7. The spatial order of accuracy was measured using the above MMS problem and is plotted for different orders of the piecewise polynomials used with DG method. The accuracy orders are as expected theoretically and prove that the spatial treatment of the 1-D fluid equations are consistent.

With a fine mesh resolution and using second order DG finite elements with Legendre basis functions, the order of convergence for density solution for a final time $t = 2$ was calculated and plotted on Fig. 7.

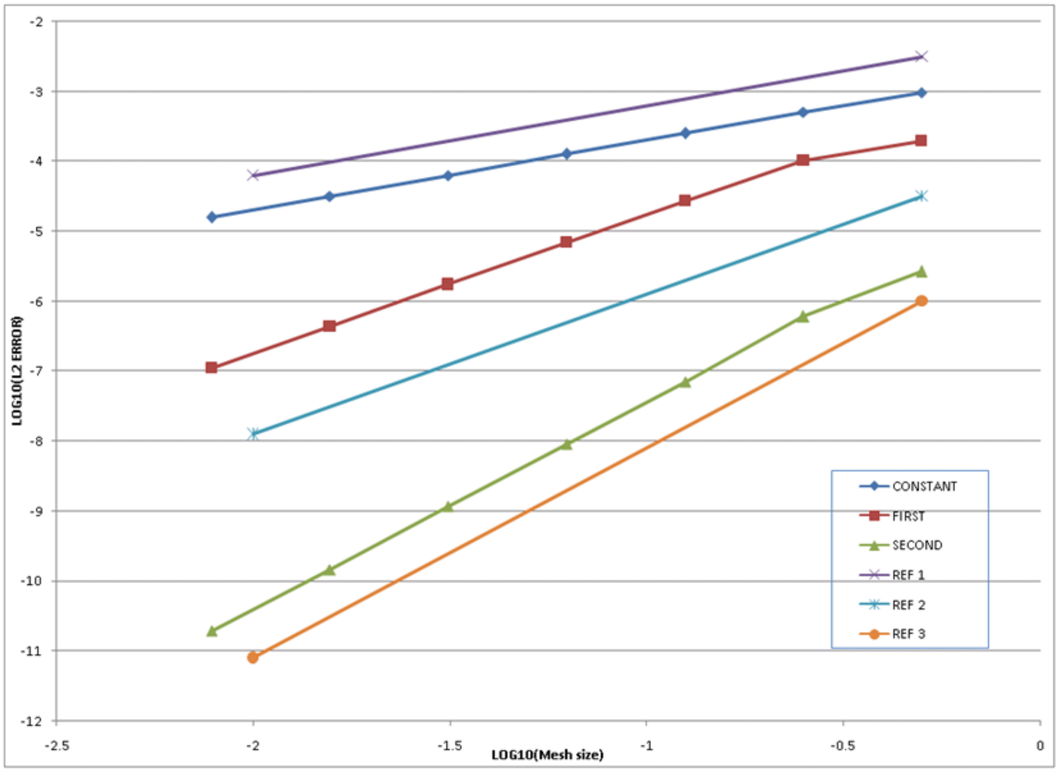


Figure 7: Spatial accuracy, fluid flow.

6.4 Coupled Neutronics/Heat-conduction problem

Neutronics and Heat conduction physics are tightly coupled and it is necessary to test the effectiveness of the coupling methodology currently implemented based on the JFNK framework.

Again, making use of the MMS techniques, a test problem was used to verify convergence of the method to exact solutions. Since the coupling between neutronics and conduction is nonlinear, and due to the fact that the conduction physics is nonlinear by itself, a script in MATLAB was written to obtain the forcing functions based on the following assumptions.

Firstly, the exact solution for the fields are taken to be

$$\phi(x, y, z) = (1 + \tanh(t)) \sin(\pi x) \sin(\pi y) xy \tag{30}$$

$$T(x, y, z) = (1 + \tanh(t)) \sin(\pi x) \sin(\pi y) \tag{31}$$

In this test, the precursors are not considered in the calculation. Hence, the transient is dominated by the prompt neutrons and is strongly affected by the doppler effects. Now, the coupling coefficients between the two physics are given by

$$\Sigma_f(T) = \Sigma_{f0}(T) + \frac{\partial \Sigma_f}{\partial T}(T - T_0) \tag{32}$$

$$\Sigma_a(T) = \Sigma_{a0}(T) + \frac{\partial \Sigma_a}{\partial T}(T - T_0) \tag{33}$$

and the linearized conductivity coefficient

$$k(T) = k_0 + \frac{\partial k}{\partial T}(T - T_0) \tag{34}$$

With these parameters, the problem was tested for spatial and temporal accuracy. The profiles obtained for neutronics and conduction fields for a sample transient are shown in Fig. 8. The spatial and temporal accuracy plots using MMS solutions are shown in Fig. 9.

It is quite clear that the coupled solutions in all physics are high order accurate. By varying the coupling coefficients in Equations (32, 33, 34), stiffer transients were created and convergence to the true solution was still observed. The higher order temporal schemes are efficient and allow the usage of larger time steps as compared to say a first order Backward Euler scheme which might be traditional used for coupled physics problems.

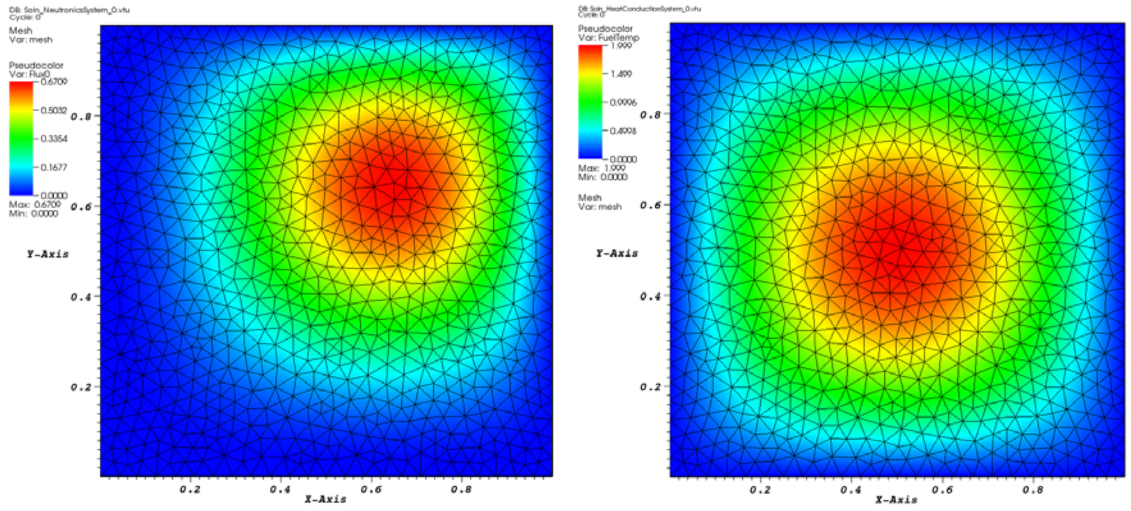
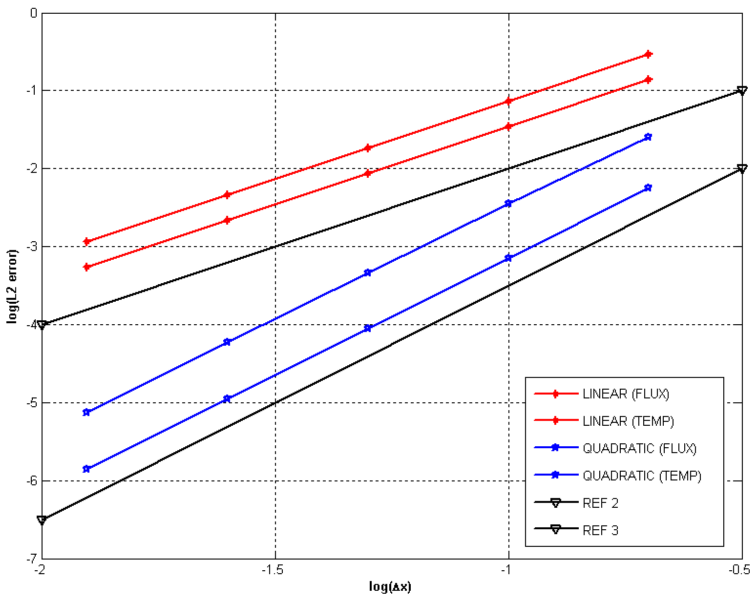


Figure 8: Coupled neutronics/heat conduction solutions (left: neutronics, right: heat conduction).



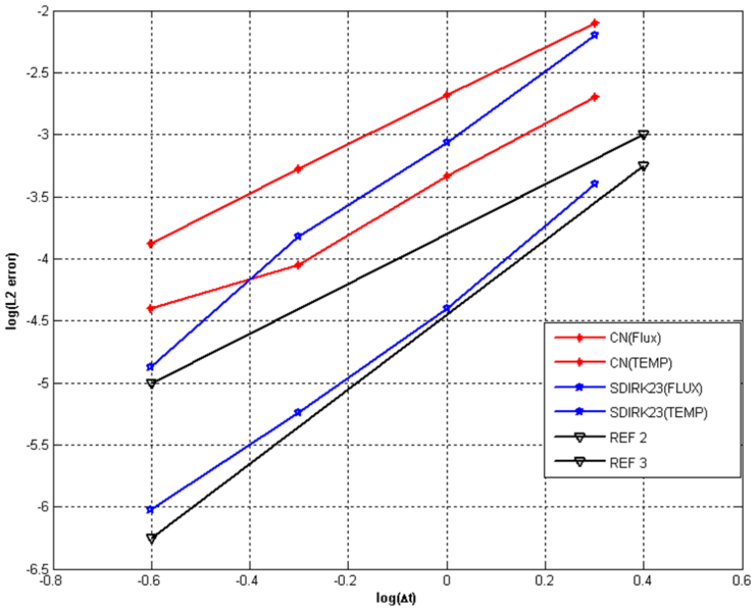


Figure 9: Spatial and temporal accuracy, Coupled neutronics/heat conduction.

7 Conclusions

KARMA, a new test bed code for multi-physics applications in reactor analysis is being developed and implemented. The rationale for KARMA is to provide a software environment in which

1. new coupling methodologies can be implemented and verified and
2. code architectures and software design for the next generation of safety analysis code can be tested.

KARMA is written in C++, with object oriented principles, and future extensions to add new physics in the future are straightforward.

KARMA's method is based on a Jacobian-Free Newton-Krylov technique. The numerics are based on state-of-the-art libraries, such as PETSc (linear and nonlinear algebra), LibMesh (finite element spatial discretization), Gmsh (mesh generation) and high-order Runge Kutta time integrators.

Some coarse grain multi-physics models have been implemented and tested. Further analysis is necessary to quantify the accuracy vs efficiency of the fully implicit JFNK scheme in comparison to traditional OS methods for problems of interest in LWR and SFR.

References

[1] Peter N. Brown , Youcef Saad, "Hybrid Krylov methods for nonlinear systems of equations", SIAM Journal on Scientific and Statistical Computing, 11, 3, 450-481 (May 1990)

[2] Knoll D. A., Keyes D. E., "Jacobian-free Newton-Krylov methods: a survey of approaches and applications", Journal of Computational Physics, 193, 2, 357-397 (2004)

[3] V.S. Mahadevan, J.C. Ragusa, "Coupling Schemes for Multiphysics Reactor Simulation", INL/EXT-07-13490, `www.osti.gov/servlets/purl/920420-s3W0S0/`, (Sep 2007)

[4] Knoll D.A., Chacon L., Margolin L.G., Mousseau V.A., "On balanced approximations for time integration of multiple time scale systems, Journal of Computational Physics, 185, 2, 583-611 (Mar. 2003)

[5] Vijay S Mahadevan, "Nonlinearly consistent schemes for coupled problems in reactor analysis", Master of Science thesis document (Dec 2006)

[6] Mousseau V.A, " Implicitly balanced solution of the two-phase flow equations coupled to nonlinear heat conduction" , Journal of computational physics, 200, 1, 104-132 (Oct 2004)

[7] Strang G, "On the construction and comparison of difference schemes", SIAM J Num Anal, 5, 506517 (1968)

[8] G.I. Marchuk, "On the theory of the splitting-up method", in Proceedings of the Second Symposium on Numerical Solution of Partial Differential Equations", 469500 (1970)

[9] Satish Balay, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, Hong Zhang, "PETSc Web page", `http://www.mcs.anl.gov/petsc`.

[10] Mathematics and Computer Science Division (MCS), Argonne National Laboratory, Argonne, Illinois

[11] B. Kirk, J. W. Peterson, R. H. Stogner, and G. F. Carey, "libMesh: A C++ Library for Parallel Adaptive Mesh Refinement/Coarsening Simulations", Engineering with Computers, 22, 3-4, 237-254 (2006)

[12] V. Hernandez, J. E. Roman and V. Vidal, "SLEPc: A Scalable and Flexible Toolkit for the Solution of Eigenvalue Problems", ACM Trans. Math. Softw. 31(3), 351-362 (2005)

[13] Lehoucq, R.B., Sorensen, D.C., Vu, P., "ARPACK: An implementation of the implicitly re-started Arnoldi iteration that computes some of the eigenvalues and eigenvectors of a large matrix" (1995)

[14] Gmsh: an automatic 3-D finite element mesh generator, available from `http://geuz.org/gmsh/`

[15] VisIt: an interactive parallel visualization tool, available from `https://wci.llnl.gov/codes/visit/home.html`

[16] George Karypis and Vipin Kumar, "A Parallel Algorithm for Multilevel Graph Partitioning and Sparse Matrix Ordering", 10th Intl. Parallel Processing Symposium, 314-319 (1996)

[17] Saad, Y. and Schultz, M. "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems." SIAM J. Sci. Statist. Comput. 7, 856-869 (1986)

[18] Mousseau,V.A., Knoll, D.A., Rider W.J., "Physics-Based Preconditioning and the Newton-Krylov Method for Non-equilibrium Radiation Diffusion", Journal of computational physics, 160, 2, 743-765, (Feb 2000)

[19] F.H. Harlow, A.A. Amsden, A numerical fluid dynamics calculation for all flow speeds, J. Comput. Phys. 8, 197 (1971)

[20] van Leer, B., Lo, M., and van Raalte, M., "A Discontinuous Galerkin Method for Diffusion Based on Recovery", AIAA 2007-4083, 18th AIAA Computational Fluid Dynamics Conference, Miami, Florida, USA (June 2007)

[21] Robert Nourgaliev, Theo Theofanous, HyeongKae Park, Vincent Mousseau, Dana Knoll, "Direct Numerical Simulation of Interfacial Flows: Implicit Sharp-Interface Method (I-SIM)", in proceedings of conference: American Institute of Aeronautics and Astronautics, Reno, USA (Jan 2008)

[22] Saad, Y., Numerical methods for large eigenvalue problems, Manchester University Press, Manchester (1992)

[23] R. B. Lehoucq and D. C. Sorensen, "Deflation Techniques for an Implicitly Restarted Arnoldi Iteration", SIAM. J. Matrix Anal. & Appl., 17, 4, 789-821 (Oct 1996)

[24] Vijay Mahadevan, Jean Ragusa, "Novel hybrid scheme to compute several dominant eigenmodes for reactor analysis problems", Proceedings of conference PHYSOR 2008 at Interlaken/Switzerland (September 2008)