

# **THE APPLICATION OF A MULTI-PHYSICS TOOLKIT TO SPATIAL REACTOR DYNAMICS**

**Clifford I**

Pebble Bed Modular Reactor (Pty.) Ltd.  
PO Box 9396, Centurion, 0046, South Africa  
ivor.clifford@pbmr.co.za

**Jasak H**

Wikki Ltd.  
London, United Kingdom  
h.jasak@wikki.co.uk

## **ABSTRACT**

Traditionally coupled field nuclear reactor analysis has been carried out using several loosely coupled solvers, each having been developed independently from the others. In the field of multi-physics, the current generation of object-oriented toolkits provides robust close coupling of multiple fields on a single framework. This paper describes the initial results obtained as part of continuing research in the use of the OpenFOAM multi-physics toolkit for reactor dynamics application development. An unstructured, three-dimensional, time-dependent multi-group diffusion code DiffusionFOAM has been developed using the OpenFOAM multi-physics toolkit as a basis. The code is based on the finite-volume methodology and uses a newly developed block-coupled sparse matrix solver for the coupled solution of the multi-group diffusion equations. A description of this code is given with particular emphasis on the newly developed block-coupled solver, along with a selection of results obtained thus far. The code has performed well, indicating that the OpenFOAM toolkit is suited to reactor dynamics applications. This work has shown that the neutronics and simplified thermal-hydraulics of a reactor may be represented and solved for using a common calculation platform, and opens up the possibility for research into robust close-coupling of neutron diffusion and thermal-fluid calculations. This work has further opened up the possibility for research in a number of other areas, including research into three-dimensional unstructured meshes for reactor dynamics applications.

*Key Words:* Diffusion, Finite-Volume, Dynamics, Multi-Physics

## **1. INTRODUCTION**

Nuclear reactor analysis deals with the coupled solution of the many physical processes taking place in a nuclear reactor. The solution of these individual physical processes has traditionally been carried out using several loosely-coupled solvers, each having been developed independently from the others. In particular, the calculation of the distribution of neutrons in space, energy and time is traditionally separated completely from the heat transfer calculation. This separation was introduced in the past for a number of reasons, namely; the solution of each class of problem is typically undertaken by specialists in each field, the complexities of the problems differ, and there are numerical differences between the classes of problems being modeled. This separation leads to problems when coupling the solvers. Often differences in data

management and spatial discretization require complex interface codes to be developed for the mapping and passing of data. Often independent source code is written to perform the same tasks in each solver and there is a significant amount of duplication. This in turn makes the verification of the coupled codes a time consuming and often labour-intensive task.

This particular problem is also encountered in the field of general multi-physics, which deals with the coupled solution of multiple fields. The field of general multi-physics analysis has advanced rapidly over recent years, embracing newer programming methodologies such as object-oriented programming. This has in turn led to the development of several multi-physics toolkits, allowing the solution of many classes of computational continuum mechanics (CCM) engineering problems in a simultaneous fashion, and readily extendable to new classes of problems. One such example is the Open-source Field Operation and Manipulation (OpenFOAM) toolkit [1], a set of classes written in the C++ programming language, which solves general partial differential equations using the finite-volume (FV) approach.

This paper describes the initial results obtained as part of continuing research in the use of the OpenFOAM multi-physics toolkit for reactor dynamics application development, in particular for pebble bed high temperature reactors (HTR). This work is a continuation of the research completed by Clifford [2]. A newly developed time-dependent multi-group neutron diffusion equation solver, DiffusionFOAM, has been developed as part of this research and is described along with a selection of the results obtained thus far. We further introduce an implicit block-coupled solution implementation that has allowed robust and stable time-dependent solutions to the diffusion equation to be obtained for an arbitrary number of energy groups.

## 2. MATHEMATICAL MODEL

The mathematical model employed in DiffusionFOAM is based on that of the TINTE [3] code system, which has been rewritten in a form more suited to the OpenFOAM toolkit. Although the methodology and design of this legacy code are somewhat outdated, the TINTE theoretical basis was chosen as a starting point for development because it is a well developed, tested and understood example of a pebble bed reactor dynamics code.

We consider the time-dependent multi-group diffusion equation with an arbitrary number of delayed neutron precursor groups. The two-dimensional, two-group diffusion equation as implemented in TINTE has been extended to three dimensions and an arbitrary number of energy groups. The influence of saturation fission products, such as  $^{135}\text{Xe}$ , is taken into account for an arbitrary number of fission product pairs. A pure heat conduction temperature calculation is assumed. This very simple temperature model has been included so that testing of cross-section feedback effects may be carried out.

## 3. MODEL DISCRETIZATION

The multi-group diffusion and solid material heat conduction equations are discretized spatially using the finite-volume method. This method is well known and understood and the details of this discretization will therefore not be supplied herein. The diffusion coefficients at the cell faces are calculated using harmonic interpolation, which conserves the neutron current on both

sides of the cell faces. Similarly, harmonic interpolation has been employed in the temperature calculation to conserve the heat flux on both sides of the cell faces.

Time integration of the multi-group diffusion, delayed neutron concentration, and saturation fission product concentration equations is carried out in an identical fashion to that of the TINTE code system. Here we assume that power varies linearly with time, which allows the direct integration of the delayed neutron and saturation fission product equations. The final set of discretized equations is second order accurate in time.

#### 4. SOLUTION STRATEGY

The steady-state solution algorithm differs from the TINTE approach, in that a simple power iteration is employed in calculating the eigenvalue. While this approach is currently not optimal, the power iteration algorithm can be readily modified and optimized in any number of ways. In contrast, the pseudo-transient approach used by the TINTE code system cannot be easily modified and this would ultimately limit the DiffusionFOAM code's future development.

The transient solution algorithm is derived from the TINTE code system. Cross-sections are pre-calculated using polynomial expansions as a function of the state parameters (temperature, buckling, etc.) on a per material basis. The neutron flux and saturation fission product concentrations are solved in an iterative fashion. Thereafter, the delayed neutron precursor concentrations are updated. A fully-explicit, staggered approach is employed for the coupled neutronic/temperature solution.

The coupled solution of the multi-group diffusion equation makes use of a newly developed block-coupled solver, which will be discussed in detail in the next section. Krylov space solver algorithms [4] are used for all matrix solutions. For all examples presented herein, the temperature (heat conduction) solution uses the preconditioned conjugate gradient algorithm.

#### 5. BLOCK-COUPLED SOLVER

A significant property of the multi-group diffusion equation system is strong inter-equation coupling due to the coupled nature of source terms and variation in time-scales between group diffusion equations. Efficient solution of the coupled system needs to account for physical properties of the equation set.

Traditionally, Finite Volume Discretization in fluid flows resorts to a *segregated* algorithm, which solves coupled transport equations one at a time, lagging inter-equation coupling terms. This is sufficient for all cases with dominant transport. Outer iterations are used to converge source term coupling; in practice, and due to a small time-step, this is seldom necessary.

For exceptional situations, e.g. stiff chemistry, where source term coupling dominates, a predictor-corrector step is used. Here, a "local equilibrium" step resolves local source term coupling by inverting a  $n \times n$  coupled system on a cell-by-cell basis, followed by a transport solution handled in a segregated manner. Thus, for problems dominated by local equilibrium, a

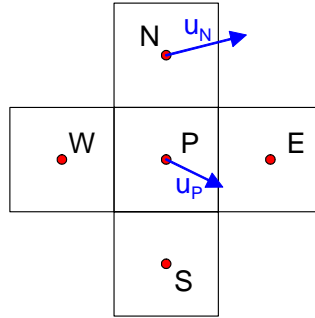
predictor step provides a good estimate of the solution and is followed by a segregated transport step.

Both approaches have been attempted on the coupled neutron diffusion system, without success. This indicates that the dominant physics of the problem is the inter-play between local source term and the range of transport time-scales. It is thus essential to solve the source balance and coupled transport in a fully implicit manner.

It remains to examine the most efficient manner of assembling the block-coupled matrix and to write a linear solver for it. In the framework of the Finite Volume Method (FVM), the global matrix preserves the properties of segregated systems. We shall examine this on a block-coupled system for a vector variable  $\mathbf{u}$ .

### 5.1. Block-Coupled Discretization

In the spirit of FVM and looking at the discretization molecule for a cell  $P$  in Figure 1, one can state that an off-diagonal coefficient between two cells will exist only if they share a face, e.g.  $P$  and  $N$ . A block-coupled system solves all components of the vector  $\mathbf{u} = (u_x, u_y, u_z)$  in a single discretization matrix.



**Figure 1: Discretisation for a block-coupled equation set**

The coupled nature of the system states that  $(u_x, u_y, u_z)$  in cell  $P$  is dependent on the values of  $(u_x, u_y, u_z)_P$  in the same cell and in any of its neighbours,  $(u_x, u_y, u_z)_N$ . The *global* coupling will still be of the same nature, with off-diagonal entries present only for cells which share a face. A result of discretization for a cell  $P$  reads:

$$a_P \mathbf{u}_P + \sum_N a_N \mathbf{u}_N = \mathbf{b} \quad (1)$$

Assembling Equation (1) for all cells yields a sparse system in the usual manner:

$$[A][\mathbf{u}] = [b] \quad (2)$$

In this framework,  $a\mathbf{u}$  can be written as a *tensorial product*, with  $a$  being a tensorial coupling coefficient:

$$a\mathbf{u} = \begin{bmatrix} a_{xx} & a_{xy} & a_{xz} \\ a_{yx} & a_{yy} & a_{yz} \\ a_{zx} & a_{zy} & a_{zz} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \quad (3)$$

In this framework, local source-term coupling is recognized by a tensorial central coefficient  $a_p$  in Equation (1). A situation where component  $\mathbf{u}_p$  depends on values in the neighbouring cell  $\mathbf{u}_N$  creates a tensorial off-diagonal coefficient  $a_N$ .

## 5.2. Levels of Block Coupling

Special cases arising in this framework are listed below.

- In a segregated system,  $a_p$  and  $a_N$  are scalar. For example, a rate of change term creates:

$$a_p = \frac{V_p}{\Delta t} \mathbf{I} \quad (4)$$

where  $\mathbf{I}$  is a unit tensor. We shall denote this as a “scalar” (diagonal or off-diagonal) coefficient. This is the case with all segregated equation sets;

- For a source-coupled system, inter-equation coupling creates a  $3 \times 3$  block  $a_p$ . For source-only coupled systems, like those present in chemical reactions  $a_N$  will remain scalar, since the velocity and diffusivity seen in the transport terms for all species will be identical;
- When the diffusion coefficient  $\nu$  is constant across components of  $\mathbf{u}$ , the diffusion coefficient  $a_N$  is similar to above:

$$a_N = \frac{\nu |\mathbf{s}_f|}{|\Delta|} \mathbf{I} \quad (5)$$

which is effectively a scalar coefficient. For a system with variable diffusion time-scales, the coupled diffusion term will create a diagonal-only  $a_N$ :

$$a_N = \begin{bmatrix} \nu_x & 0 & 0 \\ 0 & \nu_y & 0 \\ 0 & 0 & \nu_z \end{bmatrix} \quad (6)$$

A fully coupled  $a_N$  matrix would result from terms like  $\nabla \bullet [\gamma(\nabla \mathbf{u})^T]$  (present in stress analysis) or  $\mathbf{u} \bullet (\nabla \mathbf{v})^T$  (adjoint convection), creating a fully coupled system in the off-

diagonal. Consistent with the discretization, tensorial  $a_N$  results in tensorial  $a_p$  as well. The final form will be termed a “fully block-coupled system”.

### 5.3. Polyhedral Mesh Support

Three generalizations are now required. Support for unstructured polyhedral meshes in comparison with a structured mesh in Figure 1 implies a variable number of off-diagonal coefficients  $a_N$  per row, i.e. in Equation (1) thus, no generalization is needed.

### 5.4. Variable Group Size

The second generalization handles the fact that the number of coupled species in the discretized system is not necessarily three, as shown in Equation (3) but varies from model to model. However, Equation (3) shows a straightforward matrix-vector product and is readily generalized for a variable size of the equation set. One should note that for a coupled set of  $n$  equations, the size of each  $a_p$  and  $a_N$  is  $n \times n$ , which may involve considerable storage and computational effort. Thus, for an 8-component coupled system, each  $a_N$  holds 64 entries. If inter-equation coupling is not dominant, this may simply be a very expensive way of writing and solving the coupled equation set.

### 5.5. Linear Solver Algorithms

The final generalization is related to the form of the matrix and iterative solver algorithms. A combination of a large sparse matrix and a tensorial coefficient is simply a convenient way of writing  $[A][x]=[b]$ , where  $[x]$  contains all coupled variables for all cells. The form in Equations (1) and (2) simply recognizes that the sparseness pattern is identical for all components of  $\mathbf{u}$  and provides programming convenience.

Mathematically, iterative solution algorithms relate on simple basic operations: a Gauss-Seidel sweep, a vector-matrix multiplication and matrix decomposition in preconditioning. All operations of this type readily generalize for a matrix with tensorial coefficients. Even in Algebraic Multigrid (AMG), the only complication relates to assembling an appropriate measure of  $a_N$  used in agglomeration for agglomerative AMG (AAMG) or equation selection in selective AMG (SAMG). In this study, Krylov space solvers are used as a preference [4], including preconditioned Conjugate Gradient (CG) for symmetric matrices. For the tests cases described herein, the conjugate gradient algorithm was used with diagonal preconditioning.

### 5.6. A Note of Implementation

Efficiency of vector-matrix multiplication and coefficient storage is paramount. For this reason, three efficiency tweaks are used.

- Decoupled discretisation matrices, such as the ones from rate-of-change, convection, and scalar diffusion all create a scalar matrix coefficient. Storing a scalar coefficient instead of a complete tensor is a substantial saving in storage and CPU time (matrix-vector) product and is regularly present. Thus, a matrix coefficient field can take 3 forms and can

be expanded as needed:

- A scalar coefficient array, **where  $a$  is a spherical tensor:**

$$a = \alpha \mathbf{I} \quad (7)$$

- A linear (diagonal) coefficient array, **where  $a$  equals zero off the diagonal:**

$$a_N = \begin{bmatrix} \alpha_x & 0 & 0 \\ 0 & \alpha_y & 0 \\ 0 & 0 & \alpha_z \end{bmatrix} \quad (8)$$

- A fully coupled coefficient, **where  $a$  is a full  $n \times n$  tensor.**

Morphing between various forms is well-defined and will happen as needed by the algorithm;

- The matrix structure recognizes the fact that for a source-coupled coefficient, only the matrix diagonal needs to be expanded to a  $n \times n$  form. Thus, the matrix diagonal and off-diagonal are stored and expanded separately;
- For symmetric matrices, only one half of the off-diagonal coefficients are stored. For a vector-matrix product in the lower triangle, the tensorial coefficient from the upper is used in transpose form. Asymmetric matrices will clearly need to store separate upper and lower triangular coefficients.

This form of matrix storage, on-the-fly coefficient expansion and separated storage has proven to yield both a flexible and efficient block-coupled solver.

## 6. TEST CASES

The total suite of tests conducted for the verification of the DiffusionFOAM solver ranges from pure analytical single energy group comparisons using fixed cross-sections to numerical comparisons for eight-group time-dependent cases with temperature feedback. The test cases included herein are derived from the OECD PBMR 400MW coupled neutronic/thermal-hydraulic benchmark [6], a two-group axisymmetric high-temperature reactor (HTR) benchmark based on the Pebble-Bed Modular Reactor design. Included in the benchmark specification are a number of steady-state and transient test cases. Comparisons are made for steady-state exercise 3 and transient exercises 4 and 5 of the benchmark. The two-group comparisons use the TINTE code system [3] as a reference, while the multi-group comparisons use the MGT [5] code system as a reference.

Given that the current DiffusionFoam implementation includes only a pure conduction temperature feedback model, the benchmark model has been modified somewhat to allow transient comparisons to be made. These modifications include the following:

- The number of materials in the model has been reduced to three; namely core, reflector and control rod materials. The upper plenum is assumed to be filled with reflector material;

- All non-temperature related terms (buckling, Xe-135 concentration, etc.) have been removed from the cross-section representations, yielding a set of temperature-only dependent cross-sections;
- The thermal-hydraulic model has been reduced to a solid cylindrical pure conduction model. Moderator and fuel temperatures are assumed equal.

Both two-dimensional (2D) and three-dimensional (3D) DiffusionFOAM models have been created and the results compared with the TINTE code. Additionally fixed four- and eight-group cross-sections were derived based on the isotopics of the two-group model. For these multi-group models the steady-state results are compared with the MGT code system. A summary of the broad energy group structure used is given in Table I. Table II shows the comparison of  $k$ -effective for TINTE/MGT and DiffusionFOAM for three operational states; 400 MW operation with control rods at nominal operational depth, 400 MW operation with control rods fully withdrawn, and 160 MW operation. Figure 2 gives a comparison of steady-state radial flux profiles for the MGT and DiffusionFOAM models using eight energy groups.

**Table I. Group structure used for test calculations**

| Energy Groups | Group Structure    | Group Energy Boundaries [eV]  |
|---------------|--------------------|---|
| 2             | 1 fast / 1 thermal | 2.5E-3 / 3.06 / 10E6  |
| 4             | 3 fast / 1 thermal | 2.5E-3 / 3.06 / 130 / 67.4E3 / 10E6                                 |
| 8             | 4 fast / 4 thermal | 2.5E-3 / 0.075 / 0.27 / 0.825 / 3.06 / 61.4 / 15E3 / 1.354E6 / 10E6 |

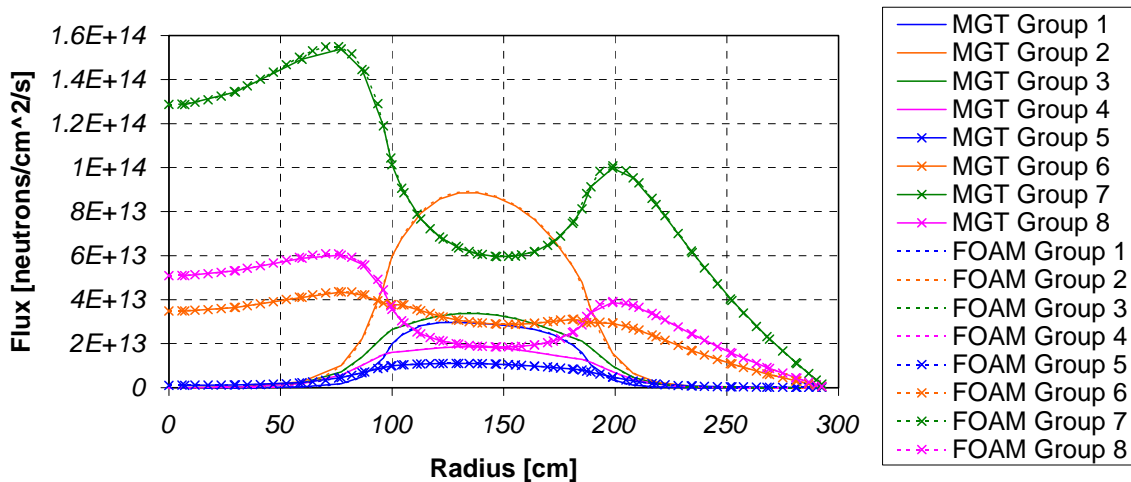
**Table II. K-effective comparisons for the simplified OECD PBMR benchmark with temperature feedback**

| Case              | Group Structure | K-effective |                  |                  |
|-------------------|-----------------|-------------|------------------|------------------|
|                   |                 | TINTE/MGT   | DiffusionFOAM 2D | DiffusionFOAM 3D |
| <b>400 MW</b>     | 2-group         | 0.99858     | 0.99804          | 0.99663          |
|                   | 4-group *       | 1.00218     | 1.00213          | -                |
|                   | 8-group *       | 1.00650     | 1.00645          | -                |
| <b>400 MW CRW</b> | 2-group         | 1.00439     | 1.00384          | 1.00236          |
| <b>160 MW</b>     | 2-group         | 1.02293     | 1.02253          | 1.02109          |

Although DiffusionFOAM supports arbitrary unstructured meshes, all test calculation models use a rectangular mesh to ensure consistency between the test models. All 2D models use identical mesh spacing (80 radial  $\times$  116 axial subdivisions). Note, however, that DiffusionFOAM

\* Fixed cross-sections used

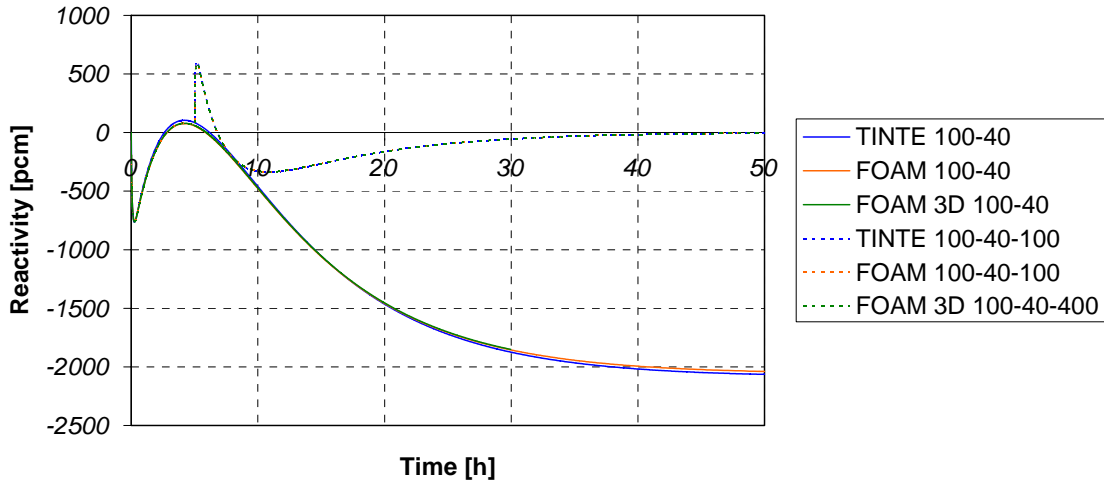
makes use of finite-volume numerics, whereas the TINTE and MGT codes both use the leakage iteration numerical scheme of Naito et al. [7] for the solution of the multi-group diffusion equation. Further, TINTE and MGT calculate material cross-sections, fission product concentrations and delayed neutron precursor concentrations using a coarse material mesh (20 radial  $\times$  29 axial subdivisions), whereas DiffusionFOAM calculates these using the more refined calculation mesh. It is expected that these numerical differences will influence the results to some extent. The 3D DiffusionFOAM model uses a relatively coarse mesh (50 radial  $\times$  75 axial  $\times$  75 azimuthal subdivisions) to reduce calculational time. The results for the 3D model are therefore expected to be less accurate than for the 2D models.



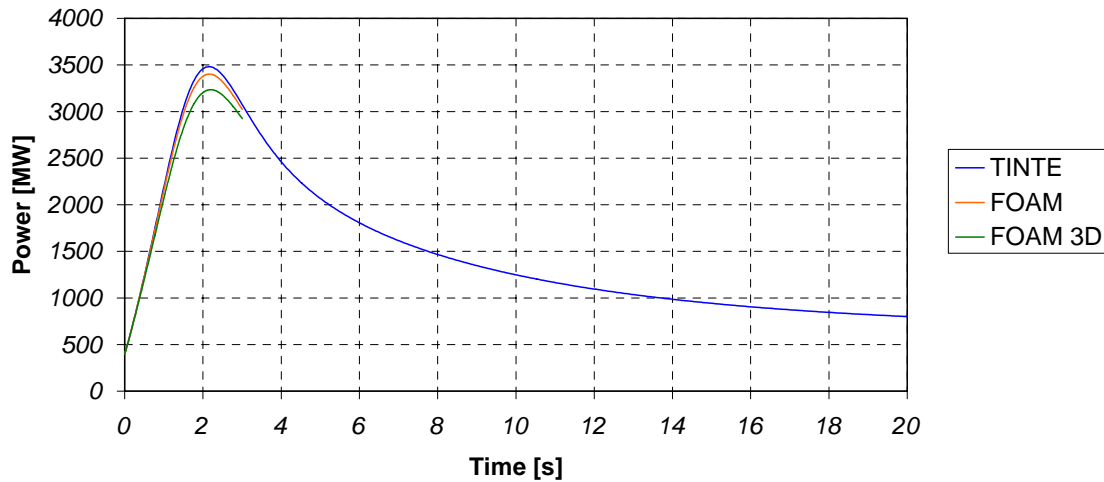
**Figure 2. Eight-group steady-state radial neutron flux profile comparison**

Transient exercise 4 of the OECD PBMR benchmark is a load-follow case. The reactor power is stepped from 400 MW (100%) to 160 MW (40%) at  $t=0$  s. After 5 h the power is stepped up to 400 MW once more. No control rod movement is modeled but rather the external reactivity, which is required to maintain a critical core, is monitored for the duration of the calculation. A second test calculation was performed for this test case assuming that the reactor power remains at 40% power. A fixed time step of 20 s is used throughout. Comparisons of the results are given in Figure 3. Differences during the 400 MW operational periods are negligible. During 160 MW operation there is approximately a 15 pcm difference between TINTE and DiffusionFOAM results. This is consistent with the  $k$ -effective comparisons of Table II.

Transient exercise 5 of the OECD PBMR benchmark is a control rod ejection case. Here, all control rods are ejected to the top of the core and the power response of the reactor monitored. A fixed time step of 0.05 s is used throughout. This power response comparison for the TINTE and DiffusionFOAM codes is given in Figure 4. The results compare well for the 2D cases, with a difference in peak power of 2%. For the 3D case there are larger differences in the peak power (7%) but the response is otherwise consistent. This is likely because of the coarser mesh used for the 3D model.



**Figure 3. Reactivity response comparisons for a load-follow calculation**



**Figure 4. Power response comparisons for a control rod ejection calculation**

A comparison of the computational time required by each model for the control rod ejection and load-follow calculations is given in Table III. DiffusionFOAM, in general, requires approximately 20% more computational time than does the TINTE code. These time differences are reasonable, given that DiffusionFOAM is a 3D unstructured solver while TINTE is limited to 2D rectangular meshes.

**Table III. Comparison of TINTE and DiffusionFOAM code execution times <sup>†</sup>**

| Case                          | TINTE  | DiffusionFOAM | DiffusionFOAM 3D      |
|-------------------------------|--------|---------------|-----------------------|
| <b>Control Rod Ejection</b>   | 1261 s | 1476 s        | 37 500 s <sup>‡</sup> |
| <b>100-40-100 Load-Follow</b> | 1089 s | 1291 s        | 44 274 s              |

## 7. FURTHER DISCUSSION

From the test cases described in Section 6 it is clear that the OpenFOAM toolkit has proven capable of solving reactor dynamics problems. We now consider what advantages the use of such a toolkit provides. We first consider some of the basic features of the toolkit. The OpenFOAM toolkit:

- Comes complete with support for 3D arbitrary unstructured meshes, including mesh generation, conversion, manipulation, mapping of data, and post-processing of results;
- Provides functionality for the construction and solution of matrices based on the finite-volume methodology;
- Allows any number of scalar, vector and tensor field variables to be defined, manipulated, and solved for;
- Can be extended to provide new and customized linear solvers, differencing schemes, boundary conditions, material properties, etc. to suite the user's specific problem.

OpenFOAM is not a replacement for mathematical and numerical research tools such as Mathematica and Matlab. It is a library of C++ classes that assist in the rapid development of computational continuum mechanics (CCM) solvers. The toolkit is designed in a hierarchical fashion, starting with low-level concepts such as scalars, vectors and tensors. At the next level, meshes, field variables and matrices are defined. On top of this, finite-volume discretization of partial differential equations is provided. At the highest level, solvers are provided for a variety of problem types including fluid flow, heat transfer, stress analysis and magneto-hydrodynamics.

This layered, modular design allows the developer to compartmentalize the problem. If, for example, a new equation operator, differencing scheme or linear solver is required then this may be developed in parallel without the risk of altering the existing code. Because the lower level concepts such as field variables are defined in a generic fashion, one may solve any number of physical problems in a consistent fashion. While the linear solution algorithms, differencing schemes, and boundary condition types may vary from one problem to another, the modular design of the toolkit encourages a consistent top-level solver design, which in turn helps to reduce the complexities associated with the coupled solution of multiple physical phenomena.

---

<sup>†</sup> Test platform uses 3GHz Intel Core 2 Duo processor with 2GB memory

<sup>‡</sup> Estimated by extrapolation

## 8. CONCLUSIONS

The OpenFOAM multi-physics toolkit has shown promise as a robust and versatile tool for the development of reactor analysis codes. Its strength lies not only in the availability of the numerous numerical tools and classes that are provided but rather in the toolkit's design, which allows extensions to be added in a modular fashion. The toolkit was successfully expanded to provide a fully-implicit block-coupled solution for an arbitrary number of coupled species, thus enabling the solution of the time-dependent neutron diffusion equation for an arbitrary number of energy groups. Test cases have shown good agreement between the DiffusionFOAM code and the TINTE and MGT codes. For the two-group two-dimensional test cases execution speed is slightly slower but comparable with the TINTE code.

While the current DiffusionFOAM solver uses a simplified thermal-hydraulics model and employs a segregated algorithm for the coupled neutronics/thermal-hydraulics solution, this work has shown that the neutronics and thermal-hydraulics of a reactor may be represented and solved for using a common calculation platform, and opens up the possibility for further research in a number of areas, including:

- Three-dimensional unstructured mesh optimization for reactor dynamics applications;
- The use of moving meshes to eliminate control rod cusping effects;
- Robust close-coupling of neutron diffusion and thermal-fluid calculations.

## REFERENCES

1. H. G. Weller, G. Tabora, H. Jasak and C. Fureby, "A Tensorial Approach to Computational Continuum Mechanics using Object-Oriented Techniques," *Computers in Physics*, **Vol. 12**, No. 6 (1998).
2. I. D. Clifford, "Object-Oriented Multi-Physics Applied to Spatial Reactor Dynamics," *North West University* (2007).
3. H. Gerwin, W. Scherer and E. Teuchert, "The TINTE modular code system for computational simulation of transient processes in the primary circuit of a pebble-bed high-temperature gas-cooled reactor," *Nuclear Science and Engineering*, **Vol. 103:3**, pp. 302-312 (1989).
4. Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd Edition, Society for Industrial Mathematics, Philadelphia, PA (2003).
5. C. Druska, S. Kasselmann and A. Lauer, "Investigations of space-dependent safety-related parameters of a PBMR-like HTR in transient operating conditions applying a multi-group diffusion code," *Nuclear Engineering and Design*, **Vol. 239:3**, pp. 508-520 (2009).
6. F. Reitsma, K. Ivanov, T. Downar, H. Haas, H. Gougar, "The OECD/NEA/NSC PBMR Coupled Neutronics/Thermal Hydraulics Transient Benchmark: The PBMR-400 Core Design," *Proceedings of the PHYSOR 2006 Conference*, Vancouver, Canada (2006).
7. Y. Naito, M. Maekawa and K. Shibuya, "A Leakage Iterative Method for Solving the Three-Dimensional Neutron Diffusion Equation," *Nuclear Science and Engineering*, **Vol. 58**, pp. 182 (1975).