SANDIA REPORT

SAND2011-9153
Unlimited Release
Printed December 2011

Verification of the Coupled Fluid/Solid Transfer in a CASL Grid-to-Rod-Fretting Simulation

Kevin D. Copps

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone:	(865) 576-8401
Facsimile:	(865) 576-5728
E-Mail:	reports@adonis.osti.gov
Online ordering:	http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone:	(800) 553-6847
Facsimile:	(703) 605-6900
E-Mail:	orders@ntis.fedworld.gov
Online ordering:	http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2011-9153
Unlimited Release
Printed December 2011

Verification of the Coupled Fluid/Solid Transfer in a CASL Grid-to-Rod-Fretting Simulation

A Technical Brief on the Analysis of Convergence Behavior and Demonstration of Software Tools for Verification

Kevin D. Copps

Sandia National Laboratories
Validation and Uncertainty Quantification Processes, Dept. 1544
P.O. Box 5800
Albuquerque, NM 87185-0897
kdcopps@sandia.gov

Abstract

For a CASL grid-to-rod fretting problem, Sandia's *Percept* software was used in conjunction with the Sierra Mechanics suite to analyze the convergence behavior of the data transfer from a fluid simulation to a solid mechanics simulation. An analytic function, with properties relatively close to numerically computed fluid approximations, was chosen to represent the pressure solution in the fluid domain. The analytic pressure was interpolated on a sequence of grids on the fluid domain, and transferred onto a separate sequence of grids in the solid domain. The error in the resulting pressure in the solid domain was measured with respect to the analytic pressure. The error in pressure approached zero as both the fluid and solids meshes were refined. The convergence of the transfer algorithm was limited by whether the source grid resolution was the same or finer than the target grid resolution. In addition, using a feature coverage analysis, we found gaps in the solid mechanics code verification test suite directly relevant to the prototype CASL GTRF simulations.

Acknowledgements

This work was funded by and conducted by the DOE CASL Energy Innovation Hub for Modeling & Simulation for Nuclear Reactors, led by Oak Ridge National Laboratory, in which Sandia National Laboratories is a core partner. The work is part of a cross-focus area CASL effort involving its VUQ and VRI elements. It benefitted from coordinating support by CASL focus area lead Jim Stewart (SNL). William Rider (SNL) coordinated the collaboration between VUQ and the GTRF analysis teams.

The simplified 3D model of the solid mechanical assembly for the GTRF problem relied on problem statements, models, and technical guidance provided by Westinghouse Electric Company LLC. Other direct contributors to the solid mechanical GTRF model definition, setup, and support were Nathan Crane (SNL), Mary White (SNL), and Rick Garcia (SNL). An earlier fluids GTRF model definition for Sierra Fuego was supported by Dan Turner (SNL) and Salvador Rodriguez (SNL). John Shadid (SNL) and Tom Smith (SNL) provided the sequence of grids and the fluid simulation pressure results using the Drekar software package. Drekar was developed along with Shadid and Smith, by Eric Cyr (SNL) and Roger Pawlowski (SNL). William Rider (SNL) completed the verification study on the fluid results. Dan Turner (SNL) and Brian Carnes (SNL) helped setup the projection of the pressure transfer from the fluid and solid grids using the Sierra Encore software package. Nathan Crane (SNL) and Walter Witkowski (SNL) analyzed earlier versions of solid mechanics model in the Sierra Mechanics Presto software. Prateek Nath (ORNL), Sam Sham (ORNL) and Nathan Crane (SNL) set up and performed subsequent solid mechanics vibration analysis in the Sierra Solid Mechanics software, using the transferred pressure data as a boundary condition. Stephen Kennon (SNL) provided Percept development and support, and Matthew Staten (SNL) developed the CUBIT CAD geometry integration with Percept, as part of the DOE ASC program. Additional hardware and software support was provided by SNL's Computer Science Research Institute, and SNL's Computational Computing and Network Services Center.

Many of the related activities—of THM members (Shadid, Smith, Weber, Cyr, Pawlowski), VRI members (Turner, Rodriguez, Garcia), MPO members (Sham, Nath) and other VUQ staff (Rider, Witkowski)—are documented in separate CASL reports. These are referenced at the end of this report.

Contents

List of Figures

List of Tables

1 Executive Summary

The success of the Consortium for Advanced Simulation of Light Water Reactors (CASL) depends critically on the ability to predict the performance of engineering systems using computer software. Predictions cannot be made reliably and accurately without verifying both the code and calculations of the software. Verification is difficult and expensive in terms of human and computer resources because it requires extra testing and calculations an order of magnitude above the work required to create simulations alone. The extra testing is required to prove there are no bugs in the capabilities offered by the software, testing which must be automated and executed daily as the software changes. The extra calculations are required because the simulation is an approximate solution of a mathematical model of a real physical system—the numerical error in the approximation must be quantified or estimated.

This technical brief documents CASL Level 3 Milestone L3.VUQ.VVDA.P3.01, *GTRF CFD-to-Mechanics Data Transfer Verification*, which was successfully completed in October 2011. The VUQ focus area led this effort. We document an example of the verification of one small part of the GTRF (Grid to Rod Fretting) challenge problem of CASL. Specifically, we examine the transfer, or projection, of the pressure field, which enables the coupling between the fluid and solid simulation of the GTRF problem. The transfer of pressure occurs between two separate software simulation codes used in the GTRF: Drekar is the fluid simulation code, and Sierra Solid Mechanics the solids code. In verifying the transfer, which is important in its own right, we demonstrate the use of software tools for reducing the burden of code and calculation verification.

The tools we demonstrate are

Encore A parallel code for pre- and post-processing, data transfer, and the method of manufactured solutions—part of the Sierra Mechanics suite.

Percept An open source parallel code for pre- and post-processing. It provides a capability for dividing and refining massive computational grids while respecting the CAD geometry definitions of engineering parts and structures.

Feature Coverage An integrated component of Sierra Mechanics that reports on how capabilities of a simulation code are tested in the code's test suite.

We demonstrate the usefulness of dedicated tools to perform verification activities. These tools are necessary across the spectrum of physics, engineering segments, and the sets of simulation codes intended to provide predictive analyses of nuclear reactor design for CASL. Some specialized versions of these same kinds of verification tools, albeit with a more limited set of capabilities, have been created as side projects during the past development of simulation codes. For example, at Sandia National Laboratories similar capabilities have been created for the Alegra, Sceptre, and for Sierra Mechanics. With the development of the Percept software package we intend to provide a one stop shop for these broadly applicable tools, licensed as open source, and offer a package that can be shared freely, developed, and collaborated on with laboratories, universities, and other members of engineering and physics disciplines beyond nuclear engineering.

In the results of our verification of the pressure transfer we show that (with the existing grid sizes used in the current Drekar and Sierra Mechanics simulations) the error in the discretized pressure projection can be adequately controlled, provided the grid sizes used in the two simulations are balanced in that one of the two grids is not much coarser than the other. Depending on the chosen grid resolutions, the relative error in a measure of the pressure can be reduced to less than two percent.

Issues that may require further investigation are:

1. Quantifying the actual sensitivity of the Sierra Solid Mechanics outputs to the errors in the transferred pressure. This is a tie in with UQ activities.

2. Analyzing the effect of unsmoothness in the transferred pressure caused by any modeling differences in the geometries of the fluid and solid domains. These differences occur when different simplifications are made to the fluid or solid domains that result in gaps in the interface between fluid and solid. In the presence of these gaps, the transfer or projection scheme must extrapolate to get complete results.

3. Analyzing the effect of non-conservative transfers and projections. The current method for coupling fluid and solid simulations does not attempt to ensure energy conservation, neither locally nor globally.

2 Introduction

The fluid flow through a fuel rod bundle causes vibrational excitation of the fuel rods in pressurized water nuclear reactors. This phenomenon is known as "Grid-to-Rod-Fretting" or GTRF[1]. GTRF wear is currently one of the main causes of fuel rod leaking in pressurized water reactors[2]. The Consortium for Advanced Simulation of Light Water Reactors (CASL) has identified GTRF as one of the challenge problems that drive the modeling and computational simulation environment for predictive simulation of light water reactors. An understanding of the GTRF phenomena through high fidelity CFD and solid mechanics simulations will reduce fuel rod cladding time-to-failure, improve reactor core performance, and reduce total costs.

The CASL simulation of GTRF links forces computed in a CFD (computational fluid dynamics) code to the detailed mechanical response computed in a structural analysis code. The link proceeds via a boundary condition in order to predict the vibrational response of a fuel rod. The boundary condition is time dependent due to the variability of the CFD solution. The CASL GTRF effort will produce numerical error estimates for both the CFD and mechanics simulations. And the computation of numerical error for the boundary condition link is also necessary to fully characterize the uncertainty.

This technical brief provides a verification of the data transfer between the two simulations with the underlying goal of demonstrating working tools for code and calculation verification. The document describes technical aspects of the work on the Level 3 CASL Milestone L3.VUQ.VVDA.P3.01, *GTRF CFD-to-Mechanics Data Transfer Verification*.

2.1 Background

Details of the Westinghouse model for CFD and fuel rod assembly were provided in [3] and [1]. The CASL VRI and THM team members completed CFD modeling activities using the Fuego[4] and Drekar[5] codes respectively. The CASL VUQ team completed a calculation verification study on these CFD models[6], which resulted in any CFD grids and other data used in this report.

Current activities in the CASL efforts on the GTRF problem involve the teams analyzing prototypes of sub-scale rod-bundle assemblies. Separate fluid and structural dynamic simulations have been conducted where the rod excitation predicted by the fluid simulations is transferred to the structural code through surface pressure boundary conditions. The initial prototype problem is a turbulent transient flow over the 3×3 rod assembly, with WEC V5H grid spacer, as defined by CASL AMA[3]. This report applies verification methodologies to the transfer from fluid code to the structural, or solid, code. Typical fluid and solid grids used in the current CASL prototype studies are shown in Figure 1.

In this study, we use both Sandia's Sierra Mechanics *Encore* software package[7]

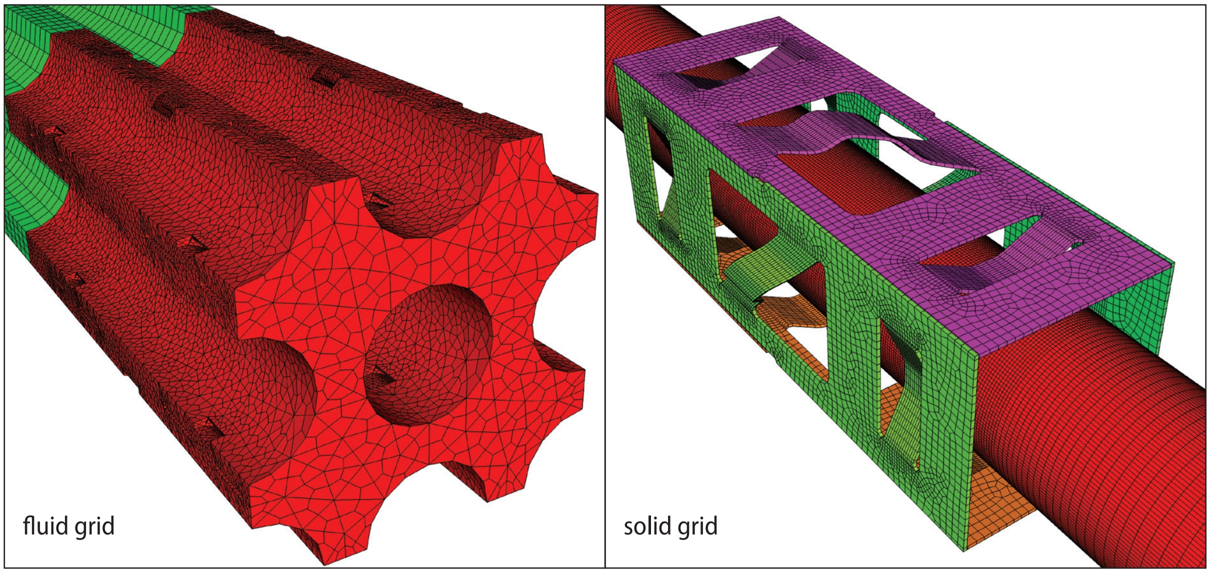

Figure 1. A snapshot of a coarse version of the fluid and solid grids recently used in the CASL GTRF coupled fluid/solid modeling activity. One block of elements in the fluid grid has been removed so that we can see detail inside the red block. The domain interface between the two grids may not closely match except near the bare fuel rod surface.

and the newer software package *Percept* (which is an open source licensed packaged component of the *Trilinos* system[8]).

2.2 Outline of the Report

In section 3 we outline the method of *feature coverage analysis*, a technique for obtaining evidence for code coverage relevant to a specific simulation. In subsection 3.1, we perform a feature coverage analysis on one of the CASL GTRF prototype solid mechanics models. The results are given in subsection 3.2.

Then in section 4, we describe verification of the transfer algorithm used to project pressure from the GTRF fluid simulations onto the boundary condition for the GTRF solid mechanics simulation. In subsection 4.1 we examine the transfer of a prototype simulation results from the Drekar code onto a solid mechanics model. In subsection 4.2 we define an analytic solution for pressure which allows precise metrics to be computed for error in the transfer algorithm. In subsection 4.3 we define the metric used to compute error in the pressure. And in subsection 4.4 we analyze the convergence behavior of the transfer algorithm using sequences of grids from both the fluid and solid models.

In section 5, we discuss our conclusions and recommendations for further study.

3 Feature Coverage

In a quotation summarizing years of experience at Hewlett-Packard, Robert Grady said, “Testing done without measuring code coverage typically exercises only about 55% of code.” Here, we must distinguish between the idea of *code coverage*, which measures the lines of the software *source code* executed by running a test suite, and the idea of *feature coverage*, which measures the possible lines of *input syntax* (assuming the input to the code can be represented in a textual form) exercised by a test suite. Code coverage is more developer centric, whereas feature coverage is user centric—and both are important. Results of both code and feature coverage analysis have the added benefit for the code developers that they may more easily target where additional tests are needed. This latter point is especially important for code verification testing because of the relatively large expense of creating verification tests.

Since feature coverage is the result of analyzing the set of all possible features (or input commands) to a simulation code and reporting the tests in the test suite that exercise those features, a strict feature coverage analysis can be an important component of the larger set of code verification activities. In Sandia’s verification and validation process and the PCMM[9] (Predictive Capability Maturity Model) this activity is referred to as *feature and capability coverage*. For simplification, we denote this idea as one-way feature coverage, which answers the question: “for each feature, is it tested in a test suite?” In addition, a two-way feature coverage analysis answers the question: “given any two features, is there one or more tests in the test suite that test both features at the same time?”

We demonstrate the process of feature coverage using a new FCT (Feature Coverage Tool) on the Sierra Solid Mechanics[10] test suite, given one of the solid mechanics models used as part of the present GTRF coupled modeling effort in CASL. The FCT is being actively improved and folded into the standard set of Sierra Mechanics tools. The process of setting up the data used by Sierra’s FCT is shown schematically in Figure 2. In this process, the inputs are all the input models in the test suite as well the user input for their specific model. The Sierra application transforms all these inputs in a form suitable for the feature coverage tool: the *log of coverage* files (shown in green). The Sierra application also provides a complete hierarchical tree of the features, basically a listing of all possible input commands: the *full syntax* tree (shown in red). The feature coverage tool collects and organizes which tests intersect each of features, and filters this result down to only those features used in the specific user input model. Results can be displayed as web pages or in a spreadsheet.

3.1 Feature coverage of the solid mechanics model

We performed a feature coverage analysis on a GTRF solid mechanics simulation input using the FCT. This was possible because the GTRF solid mechanics vibration model was created within the Sierra Mechanics system.

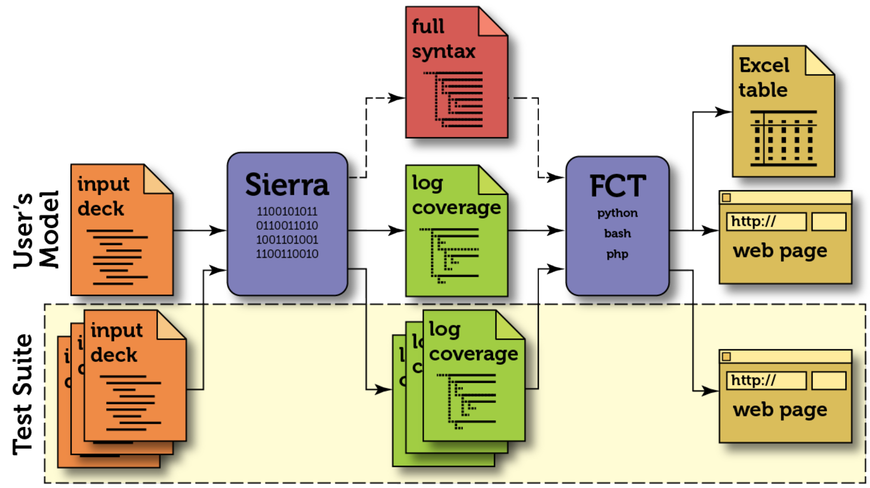

Figure 2. The Sierra FCT (feature coverage tool) digests and combines a specific user input model and all the tests in a test suite to produce an analysis. It is dependent on a user input file, a test suite, and the physics application code.

To complete a feature coverage analysis on a model, i.e., your own Sierra input file, there are a few prerequisites. You must have access to a *certificate of coverage* for a specific version of Sierra, a physics application, and a specific test suite. A test suite could be a subset of all the regression, integration, and system tests, such as a *verification* test suite: a set of high quality tests with known exact solutions and/or convergence rate tests. The certificate of coverage file has the `*.ccv` extension. In the future, the certificates of coverage will distributed with each version of Sierra. The Sierra Solid Mechanics input that we used to complete the feature coverage analysis is given in Appendix A.

3.1.1 Procedure to perform the feature coverage analysis

The procedure used to produce feature coverage results for a solid mechanics input model is as follows. This detail is included for readers that are interested in using this capability.

1. Place the `*.ccv` certificate of coverage in the same directory as the model input. Typically this is one for a verification test suite or regression test suite.

2. Make sure to `module load sierra-devel` (this should give you access to the `feature_coverage` tool, as well as the sierra command.

3. Run sierra to create a coverage log `*.icv` for your sierra input file. For example, if your input file is `my_input.i`, then execute:

```
sierra presto -i my_input.i -O "--command-coverage --check-syntax"
```

4. Run the feature coverage tool with the `-r -i -o` options to produce a comma separated values output file (`*.csv` file). Run the feature coverage tool with the -r -i -o options to produce a comma separated values output file.

```
feature_coverage -r solid_mechanics_verification.ccv -i my_input.icv
                 -o my_input_verification.csv
```

where

`solid_mechanics_verification.ccv` is a coverage certificate file,

`my_input.icv` is the coverage log you just created with sierra, and

`my_input_verification.csv` is your name for the output, containing a comma-separated values file.

5. Open the output `*.csv` in Microsoft Excel, or other spreadsheet program.

3.2 Results of feature coverage analysis

The results of the one-way coverage analysis of the GTRF solid mechanics model are illustrated in Table 1. This revealed that four of the features used in the solid mechanics model are not tested by any of the tests in the Sierra Solid Mechanics *verification* test suite. Athough, the same four features may be tested in other kinds of tests within Sierra Mechanics.

Also using the FCT on the same input file, again with respect to the verification test suite, we analyzed the two-way feature coverage. This output is shown in Table 2. In this two-way analysis, the features are expressed in a compressed hierarchical form, where related features are grouped together. The hierarchical levels are separated by the vertical bar character "|". The feature interaction in the test suite is represented by a matrix of rows and columns; a feature is tested with another feature in one or more tests if a black square appears in the corresponding row and column. We note that the one-way coverage is also present in the two-way matrix as the diagonal. One can see that the four empty rows/columns are those previously seen in the one-way coverage.

Table 1. The results of the one-way feature coverage analysis on one of the solid mechanics model input shows the number of high quality tests in the Sierra Solid Mechanics verification suite that excercise these features (input commands). This data provides evidence that a feature used in an actual simulation is well tested. This evidence could be provided as part of a PCMM report on the simulation.

<table>
<tr><th>Number of Tests</th><th>Feature (actual input command)</th></tr>
<tr><td>315</td><td>Begin Sierra <jobidentifier></td></tr>
<tr><td>25</td><td>Define Axis <axisname: string> With Point <pointname: string> {Direction|Point} <directname: string></td></tr>
<tr><td>275</td><td>Define Direction <directname: string> With Vector <components: real[3]></td></tr>
<tr><td>48</td><td>Define Point <pointname: string> With Coordinates <coordinates: real[3]></td></tr>
<tr><td>315</td><td>Begin Adagio Procedure <procedurename></td></tr>
<tr><td>315</td><td>Begin Adagio Region <regionname></td></tr>
<tr><td>315</td><td>Use Finite Element Model <modelname: string> [Model Coordinates Are <nodal_variable_name: string>]</td></tr>
<tr><td>85</td><td>Begin Contact Definition <contactname></td></tr>
<tr><td>4</td><td>Compute Contact Variables = {Off|On}</td></tr>
<tr><td>11</td><td>Contact Formulation Type = {Acme|Ars|Dash}</td></tr>
<tr><td>67</td><td>Contact Surface <surface_name: string> Contains <list_of_instances: string+></td></tr>
<tr><td>5</td><td>Begin Constant Friction Model <name></td></tr>
<tr><td>5</td><td>Friction Coefficient = <coeff: real></td></tr>
<tr><td>20</td><td>Begin Interaction Defaults</td></tr>
<tr><td>3</td><td>Friction Model = <name: string></td></tr>
<tr><td>20</td><td>General Contact = {Off|On}</td></tr>
<tr><td>202</td><td>Begin Fixed Displacement <name></td></tr>
<tr><td>38</td><td>Block = <block: string+></td></tr>
<tr><td>27</td><td>Component = {X|Y|Z}</td></tr>
<tr><td>17</td><td>Surface = <surface: string+></td></tr>
<tr><td>62</td><td>Begin Fixed Rotation <name></td></tr>
<tr><td>19</td><td>Block = <block: string+></td></tr>
<tr><td>3</td><td>Component = {X|Y|Z}</td></tr>
<tr><td>2</td><td>Begin Mpc <name></td></tr>
<tr><td>0</td><td>Tied Nodes = <id: integer[2:]></td></tr>
<tr><td>103</td><td>Begin Prescribed Displacement <name></td></tr>
<tr><td>48</td><td>Component = {X|Y|Z}</td></tr>
<tr><td>89</td><td>Function = <functionname: string></td></tr>
<tr><td>80</td><td>Scale Factor = <scalefactor: real></td></tr>
<tr><td>11</td><td>Surface = <surface: string+></td></tr>
<tr><td>50</td><td>Begin Pressure <name></td></tr>
<tr><td>0</td><td>Node Set Subroutine = <subroutinename: string></td></tr>
<tr><td>0</td><td>Subroutine Integer Parameter: <variablename: string> = <variablevalue: integer></td></tr>
<tr><td>0</td><td>Subroutine Real Parameter: <variablename: string> = <variablevalue: real></td></tr>
<tr><td>50</td><td>Surface = <surface: string+></td></tr>
<tr><td>313</td><td>Begin Results Output <label></td></tr>
<tr><td>227</td><td>At Time <dt1: real> {Increment|Interval} = <dt2: real></td></tr>
</table>

–continued on next page...

...continued from previous page

Number of Tests	Feature (actual input command)
313	`Database Name = <streamname: string>`
298	`Database Type = {Exodus|exodusII|Generated|Genesis|Xdmf}`
259	`Element Variables = [<variablelist: string+>]`
237	`Global Variables = [<variables: string+>]`
313	`Nodal Variables = [<variablelist: string+>]`
315	`Begin Time Control`
315	`Termination Time = <tend: real>`
315	`Begin Time Stepping Block <blockname>`
315	`Start Time = <tstart: real>`
247	`Begin Parameters For Presto Region <presto_region_name>`
244	`Step Interval = <step_interval: integer(>=0)>`
50	`Begin Definition For Function <functionname>`
18	`Abscissa = <name: string+> [Scale = <scale: real> Offset = <offset: real>]`
4	`Evaluate Expression = <expr: expression>`
18	`Ordinate = <name: string+> [Scale = <scale: real> Offset = <offset: real>]`
50	`Type = {Analytic|Constant|Multicolumn Piecewise Linear|Piecewise Analytic|...}`
50	`Begin Values <empty>`
50	`<xyvalues: real+>`
315	`Begin Finite Element Model <label>`
315	`Database Name = <streamname: string>`
291	`Database Type = {Exodus|exodusII|Generated|Genesis|Xdmf}`
315	`Begin Parameters For Block <blockname>`
102	`Material <matname: string>`
190	`Section = <sectionname: string>`
43	`Solid Mechanics Use Model <modelname: string>`
46	`Begin Property Specification For Material <materialname>`
46	`Density = <density: real(>=0)>`
43	`Begin Parameters For Model Elastic`
38	`Begin Rigid Body <name>`
117	`Begin Solid Section <solid_section_name>`
31	`Rigid Body = <rigid_body_name: string>`
9	`Begin Spring Section <spring_section_name>`
9	`Default Stiffness = <preload_stiffness: real>`
9	`Mass Per Unit Length = <mass_per_unit_length: real(>=0)>`

Adding and improving the verification test suite of a code is a time consuming activity, especially given that thousands of possible inputs exist to typical simulation codes. Nonetheless, we conclude that some higher quality tests could be added to the Sierra solid mechanics test suite to fill the four gaps revealed in our coverage analysis.

Table 2. The two-way feature coverage analysis on one of the solid mechanics model inputs. A black square indicates two features of the code were tested together in one or more tests in the verification suite. Note that the blank rows/columns coincide with the one-way results.

Feature		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67
adagio procedure	1	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
adagio procedure|time control	2	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
adagio procedure|time control|termination time	3	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
adagio procedure|time control|time stepping block	4	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
adagio procedure|timecontrol|time stepping block|parameters for presto region	5	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
adagio procedure|timecontrol|time stepping block|parameters for presto region|step interval	6	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
adagio procedure|time control|time stepping block|start time	7	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
boundary_condition|fixed displacement	8	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
boundary_condition|fixed displacement|block	9	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■			■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
boundary_condition|fixed displacement|component	10	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■			■		■	■	■	■	■	■	■	■	■		■	■	■	■	■	■						■	■	■	■	■						■	■	■	■	■	■	■		■				
boundary_condition|fixed displacement|surface	11	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■						■		■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■		■	■	■	■	■	■				■	■	■	■	■	■	■	■		■				
contact	12	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■			■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
contact|compute contact variables	13	■	■	■	■			■	■	■	■		■	■	■	■	■				■		■	■	■	■	■	■	■	■	■	■				■	■	■	■	■						■	■	■	■	■						■		■	■	■	■	■		■				
contact|constantfriction	14	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■		■	■	■	■	■	■	■	■	■	■				■	■	■	■	■						■	■	■	■	■						■	■	■	■	■	■	■		■				
contact|constantfriction|friction coefficient	15	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■		■	■	■	■	■	■	■	■	■	■				■	■	■	■	■						■	■	■	■	■						■	■	■	■	■	■	■		■				
contact|contact surface	16	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
contact|interaction defaults	17	■	■	■	■	■	■	■	■	■	■		■				■	■	■	■	■		■		■	■	■	■	■	■	■	■				■	■	■	■	■											■				■	■	■	■	■	■	■	■		■				
contact|interaction defaults|friction model	18	■	■	■	■	■	■	■	■	■	■		■					■	■	■			■		■	■			■	■	■								■	■																■	■	■	■	■	■	■		■				
contact|interaction defaults|general contact	19	■	■	■	■	■	■	■	■	■	■		■				■	■	■	■	■		■		■	■	■	■	■	■	■	■				■	■	■	■	■											■				■	■	■	■	■	■	■	■		■				
contact|library	20	■	■	■	■	■	■	■	■	■			■	■	■	■	■	■		■	■		■	■	■	■	■	■	■	■	■	■				■	■	■	■	■																■	■	■	■	■	■	■		■				
define axis	21	■	■	■	■	■	■	■	■	■			■				■					■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■	■				■		■	■							■	■	■	■	■	■	■	■	■	■			
define direction	22	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
define point	23	■	■	■	■	■	■	■	■	■			■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■	■				■	■	■	■		■				■	■	■	■	■	■	■	■	■	■	■			
finite element model	24	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
finite element model|block	25	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
finite element model|block|material	26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■			■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
finite element model|block|model	27	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
finite element model|block|section	28	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■			■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
finite element model|path	29	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
finite element model|type	30	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
function	31	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
function|abscissa	32	■	■	■	■	■	■	■	■	■	■	■	■				■						■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
function|evaluate expression	33	■	■	■	■	■	■	■	■														■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■					■	■	■	■							■	■	■	■	■	■	■						
function|ordinate	34	■	■	■	■	■	■	■	■	■	■	■	■				■						■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
material	35	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
material|density	36	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
material|elastic	37	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■		■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
mesh	38	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
mesh|finiteelementmodel	39	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
mesh|fixed rotation	40	■	■	■	■	■	■	■	■	■			■				■					■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■		■				■	■	■	■	■	■	■	■	■	■	■			
mesh|fixed rotation|block	41	■	■	■	■	■	■	■	■	■			■				■					■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■			■	■	■	■		■				■	■	■	■	■	■	■	■	■	■	■			
mesh|fixed rotation|component	42	■	■	■	■	■	■	■	■	■			■										■		■	■	■		■	■	■								■	■	■	■	■														■	■	■	■	■	■	■	■	■	■			
mesh|mpc	43	■	■	■	■	■	■	■	■			■											■		■	■				■	■								■	■				■													■	■	■	■	■	■	■						
mesh|mpc|tied nodes	44																																																																		
mesh|prescribeddisplacement	45	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■					■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
mesh|prescribeddisplacement|component	46	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■						■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
mesh|prescribeddisplacement|function	47	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■					■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
mesh|prescribeddisplacement|scale factor	48	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■					■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
mesh|prescribeddisplacement|surface	49	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■						■		■	■	■	■	■	■	■	■	■		■	■	■		■	■						■	■	■	■	■	■				■	■	■	■	■	■	■	■		■				
mesh|pressure	50	■	■	■	■	■	■	■	■	■		■	■				■	■		■			■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
mesh|pressure|node set subroutine	51																																																																		
mesh|pressure|subroutine integer parameter	52																																																																		
mesh|pressure|subroutine real parameter	53																																																																		
mesh|pressure|surface	54	■	■	■	■	■	■	■	■	■		■	■				■	■		■			■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■				■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
output	55	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
output|at time	56	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
output|element variables	57	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
output|global variables	58	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
output|nodal variables	59	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
output|path	60	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
output|type	61	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■	■	■	■
rigid body	62	■	■	■	■	■	■	■	■	■			■				■					■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■			■	■	■	■		■				■	■	■	■	■	■	■	■	■	■	■			
solid section	63	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■			■	■	■	■	■	■				■	■	■	■	■	■	■	■	■	■	■			
solid section|rigid body	64	■	■	■	■	■	■	■	■	■			■				■					■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■			■	■	■	■		■				■	■	■	■	■	■	■	■	■	■	■			
spring section	65	■	■	■	■	■	■	■	■														■		■	■			■	■	■					■	■	■	■	■																■	■		■	■	■	■				■	■	■
spring section|default stiffness	66	■	■	■	■	■	■	■	■														■		■	■			■	■	■					■	■	■	■	■																■	■		■	■	■	■				■	■	■
spring section|mass per unit length	67	■	■	■	■	■	■	■	■														■		■	■			■	■	■					■	■	■	■	■																■	■		■	■	■	■				■	■	■

4 Verification of the Transfer Algorithm

Our primary technique for performing code verification studies in this work is a definition of an analytic function for the pressure field, a stand-in for an "exact" solution, or manufactured solution of the pressure. We characterize the numerical pressure field from an actual simulation in subsection 4.1. Next we formulate the definition of an analytic pressure field function in subsection 4.2. We comment on the use of global norms, which are commonly used as a measure of error in the convergence analysis of finite element methods in subsection 4.3. And we show the results of our convergence analysis in subsection 4.4.

4.1 Examination of the pressure field from simulation results

We examined the fluid results output by the Drekar code in previous GTRF simulations. Four existing meshes for the fluid results already existed, being the output of the verification study for the fluid results[6]. These four meshes contained the following number of elements: 600K, 1M, 3M, and 6M (approximately).

The transfer process used by all the prototype GTRF simulations to couple the fluids/solids models is implemented in Sierra Mechanics suite, and is executed directly by the Encore application[7]. A listing of the Encore input for the transfer is given in Appendix B. We examined the resulting pressure on a coarse solid mechanics grid resulting from the transfer of the numerical pressure from the 600K fluids grid. Not only did this give us adequate bounds on the pressure and allow us to characterize our analytic solution, p_{ex} (discussed further below), but it also showed some anomalies in the pressure. The anomalies were sharp changes in the pressure value at the interfaces of two neighboring elements on the surface.

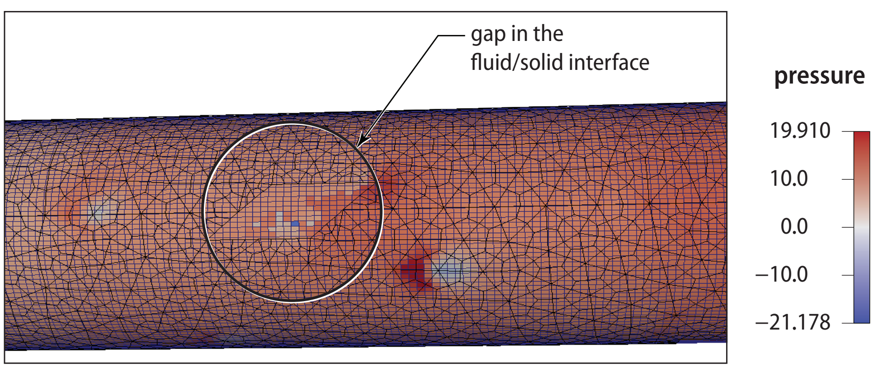

Figure 3. A closeup of the transferred pressure on the fuel rod grid (from a previously calculated Drekar numerical simulation) shows unsmoothness in the pressure boundary condition. The transfer algorithm extrapolates results from the nearest possible grid points on the source grid when the source and target grids exhibit gaps at their interface surfaces.

There was unsmooth behavior in the resulting pressure on the fuel rod. This is a result of two factors: (1) the fluid grid surface in the region of the fuel rod is defined by both tetrahedra and hexahedra while the solid mechanics grid is made up of hexahedral elements with quadrilateral surfaces; and (2) the intersection of the fluid and solid domains covered the respective grids do not match completely—some gaps are present between the complex surfaces due to modeling simplifications. These effects can be seen in one view of the results of the pressure transfer, shown in Figure 3.

When the respective grids in a transfer do not match, the Sierra Mechanics transfer attempts to extrapolate from the source field at the nearest node on the source grid. This is a forgiving algorithm, which allows non-matching source and target grids/domains, but is not guaranteed to result in a smooth field on the target domain. Further, and possibly more importantly, the transfers provided by the Sierra Mechanics are neither globally or locally conservative. There is no numerical conservation of energy between our fluid and solid domains.

4.2 Analytic Solution for Pressure

Our strategy for verifying the transfer process is similar to the method of nearby problems for estimating error presented by Roy, Raju and Hopkins[11]. This method takes some inspiration from the *method of manufactured solutions*[12, 13] intended for *code* verification and carries those ideas through into *calculation* verification.

> The method of nearby problems is developed as an approach for estimating numerical errors due to insufficient mesh resolution. A key aspect of this approach is the generation of accurate, analytic curve fits to an underlying numerical solution. Accurate fits are demonstrated using fifth-order Hermite splines that provide for solution continuity up to the third derivative, which is recommended for second-order differential equations[11].

Here, however, we do not assume that the result will give a necessarily accurate estimate on the error. Although we generate an analytic fit to an underlying numerical solution (i.e., the fluid pressure), this is not a piecewise curve fit as in [11]. Instead, the analytic function is chosen to roughly represent the minimum and maximum values of the numerical pressure and maintain some oscillatory behavior about the circumference of the fuel rod geometry. With this choice, we are only attempting to show that the transfer algorithm converges with decreasing mesh size. Because we use meshes and geometries taken from the actual modeling activity by the GTRF fluid and solids teams, we will still manage to see the effect of the relative discretization error and any errors inherent in the algorithm.

Thus, for purposes of measuring a very precise error in the transfer algorithm, we posit the existence of an exact solution for the pressure, p_{ex}. We chose the form of this function in an attempt to mimic the actual approximations to pressure we observed as output from the fluids code, Drekar.

4.2.1 Verification Procedure

1. Implement a subroutine for evaluating $p_{ex}(x, y, z, t)$ at any point and time.

2. Interpolate p_{ex} on the fluids grid by evaluating p_{ex} at the discrete grid points, resulting in p_{int} (the interpolated exact pressure).

3. Execute the transfer algorithm: transfer the interpolated p_{int} to the solid mechanics domain in the exact same way as the transfer procedure that is used for coupling the Drekar/Sierra (fluid/solid) simulations. The pressure on the solid grid is now the transferred pressure p_t.

4. Compute a measure of error between the original exact pressure, p_{ex} and the transferred pressure, p_t.

5. Repeat this procedure while varying both the fluids grid size and the solid grid size independently.

What did we use for the exact pressure, p_{ex}? A form that oscillated around the circumference of the fuel rod, and at the same time reduced in magnitude farther from the fuel rod. Let p_{ex} be a function of time t and space in cylindrical coordinates, (r, θ, z), where the z-axis is the centroidal axis of the fuel rod,

$$p_{ex}(x, y, z) = g(r)h(z)f(\theta, t) \tag{1}$$

where

$$g(r) = \text{sech}\left(\frac{r - R}{3R}\right) \tag{2a}$$

$$h(z) = \text{sech}\left(\frac{2(z - z_0)}{z_1 - z_0}\right) \tag{2b}$$

$$f(\theta, t) = c_0 + c_1(2 + \sin(t)) \sum_{i=1}^{2} a_i \sin(b_i\theta) \tag{2c}$$

and the constants are given by

$$a_0 = 1/2 \qquad a_1 = 1/4$$
$$b_0 = 4 \qquad b_1 = 16$$
$$c_0 = 45 \qquad c_1 = 4$$
$$z_0 = 0 \qquad z_1 = 0.169658$$

This function was coded as a C++ language module which we used to interpolate to grids and evaluate the error using the Encore software package. A listing of the source code is available upon request from the author.

We chose this function such that the maximum and minimum bound the numerical solutions coming out of Drekar. The values of the pressure field are of more importance near the surface of the fuel rod, but the values away from the fuel rod may also be used because of the extrapolation from nearby points that happen in the transfer algorithm. A plot of the function $p_{ex}(x, y, z = 0, t = 0)$ is shown in Figure 4.

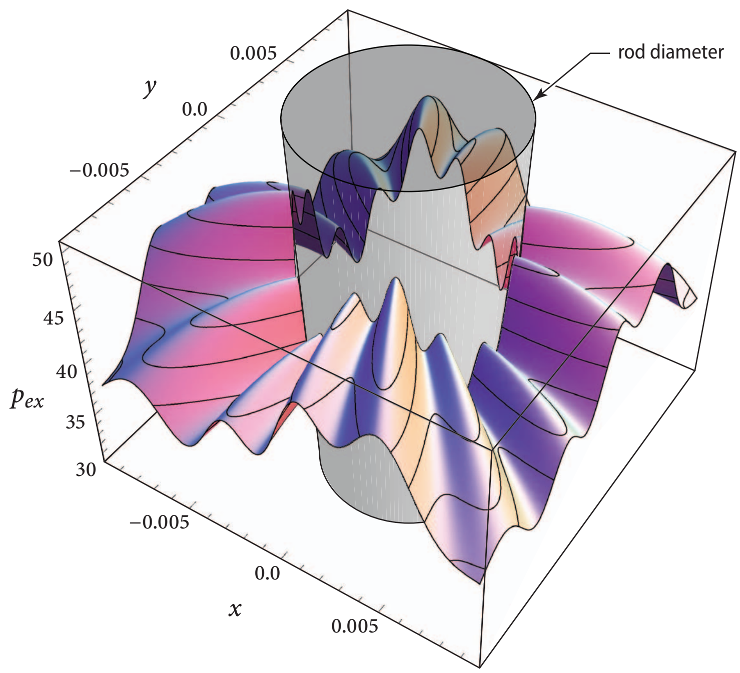

Figure 4. A plot of the analytic pressure function, $p_{ex}(x, y, z = 0, t = 0)$, shows the amplitude and oscillations about the circumference θ and the reduction in magnitude with increasing r.

4.3 Measure of Error

We will consider the $L^2(\Gamma)$ norm as a measure of the pressure on the surface of the solid mechanics domain. This norm is a semi-global measure of the value of a scalar field. We can turn this norm into a measure of accuracy if we suppose we have an exact form for the pressure field, p_{ex}. Let the error, $e = p_t - p_{ex}$, be the error between the transferred pressure and supposed exact pressure. We will compute the norm of the error in the discretized finite element sense.

$$\|e\|_{L^2(\Gamma)} \equiv \left(\int_\Gamma |e|^2 \, ds \right)^{1/2} \approx \left(\sum_{\Gamma_e} \sum_q |e(x_q)|^2 \, |J(x_q)| \, w_q \right)^{1/2} \tag{3}$$

where on an element Γ_e, we have the quadrature points x_q, the Jacobians $|J(x_q)|$ and the weights w_q. The norm is approximated using a suitable element quadrature rule. We used a quadrature rule that was fourth order accurate for all elements, using a large number of sample points per element.

We could have also consider some other error in a quantity of interest Q, defined abstractly as

$$\mathscr{E}(u, u_h) \equiv Q(u) - Q(u_h).$$

The L^2 norm over the surface, however, and other global norms have various useful properties that serve us well in comparing numerical and analytic functions. The use of functional norms is standard practice in proofs of convergence and enjoys a history of use in the verification of finite element methods[14]. We would expect the norm of the error to behave monotonically as the grids are refined, and it is guaranteed not to change sign.

4.4 Convergence Results

We used the Percept software to create a sequence of four successively uniformly refined grids for the fuel rod. We started with a coarse grid for the rod we generated in Sandia's CUBIT mesh generation software, but with an additional option to output the CAD geometry entities defining the surfaces of the model. The geometry is contained in a `*.3dm` file. The mesh contains indexing information with references back to the geometrical entities in the `*.3dm` file. A listing of the CUBIT journal file used to create the coarse mesh is given in Appendix C.

The coarse mesh we generated had the same grid size as used in coarse models in the prototype solid mechanics GTRF modeling effort. The Percept software can read the CAD geometry from the `*.3dm` file and use it during refinement to place new nodes, and conform all surface features to the original CAD geometry.

We noted early on that the results for transient cases were nearly the same as if we only examined a single time plane, and therefore in the remainder of the analysis we only computed results for $t = 0$. We found the simple linear interpolation in the time plane during the transfer had no significant effect. The only significant errors in the pressure were spatial.

We interpolated the p_{ex} function to the four existing meshes from the previous fluid results calculations. We then computed the transferred pressure on each of the four solid mechanics models of the fuel rod, resulting in $(3 \times 4 = 12)$ twelve separate results. We then computed the L^2 norm of the transferred pressure and L^2 norm of the exact pressure on each of the four solid grids. Results are shown in Table 3.

Table 3. Norms of transferred pressure from a series of three fluid grids, and the norm of the exact pressure, computed on a series of four solid mechanics grids. The number of surface elements are the number of element faces on the surface of the solid mechanics mesh of the fuel rod.

	fluid source grid			
surface elements	**600K** $\|\|p_t\|\|_{L^2}$	**1M** $\|\|p_t\|\|_{L^2}$	**3M** $\|\|p_t\|\|_{L^2}$	$\|\|p_{ex}\|\|_{L^2}$
880	2.2184336	2.2223336	2.2238410	2.2223430
3520	2.2236564	2.2276236	2.2268108	2.2265136
14080	2.2249751	2.2274896	2.2279980	2.2281246
56320	2.2253092	2.2279969	2.2281926	2.2284603

Next we computed the L^2 norm of the error between the transferred pressure and our exact pressure function. Results are shown in Table 4. We also plot these results in Figure 5.

Table 4. Relative error in transferred pressure from a series of three fluid grids, computed on a series of four solid mechanics grids. Surface elements are the number of element faces on the surface of the solid mechanics mesh of the fuel rod.

	fluid source grid		
surface elements	**600K** $\frac{\|\|p_t - p_{ex}\|\|_{L^2}}{\|\|p_{ex}\|\|_{L^2}}$	**1M** $\frac{\|\|p_t - p_{ex}\|\|_{L^2}}{\|\|p_{ex}\|\|_{L^2}}$	**3M** $\frac{\|\|p_t - p_{ex}\|\|_{L^2}}{\|\|p_{ex}\|\|_{L^2}}$
880	0.094755078	0.074101525	0.064353987
3520	0.065625911	0.027400385	0.038468772
14080	0.055905757	0.029512620	0.020368168
56320	0.053160623	0.020392383	0.015499714

The error of the pressure on the four solids grids from the 600K fluids grid eventually levels off, and never reduces below five percent. We would expect this trend, as the discretization of the source 600K fluids grid is constant and on a relatively coarse mesh. This demonstrates the desire for the source representation to be an equal or finer resolution than the target. The error using the 1M fluid grid as a source exhibits non-monotonic behavior; the reason for this is not clear, but may be due to the non-smooth pressure field due to the gaps in the fluid/solid interface originating from the extrapolation of the transfer algorithm. The 3M fluid element grid shows the best behavior, again a good reason to desire the source originate from a finer grid than the target.

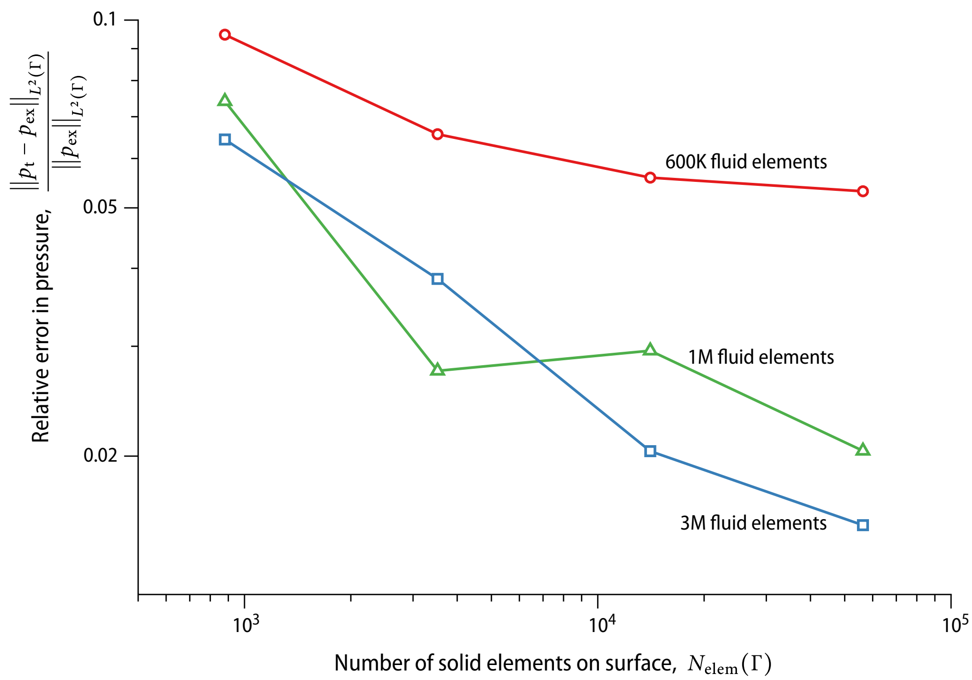

Figure 5. The error in the pressure transferred from the fluid to the solid domain was measured using the L^2 norm. The three fluids grid sizes, and their discretized interpolation of the exact pressure, p_{ex}, were transferred to a sequence of four successively uniformly refined solids grids. The relative error ranged from ten percent down to less than two percent.

5 Conclusions and Recommendations

We have presented a discussion of the FCT (feature coverage tool), a new tool provided in the Sierra Mechanics suite. The FCT can help collate and make easily accessible important evidence necessary for completing a PCMM analysis for a specific simulation. We demonstrated the use of the tool and its products on one of the solid mechanics models used in the CASL GTRF activity. We found four gaps in the Sierra Solid Mechanics verification test suite—gaps of untested features that were used in the CASL GTRF model.

We examined the transfer algorithm used in the prototype CASL GTRF fluid/-solid coupled modeling and simulation. The transfer algorithm is implemented in Sandia's Sierra Mechanics. To get precise convergence measurements of the transfer, we used an analysis technique similar to the MMS (method of manufactured solutions), or the method of nearby problems. With this method we were able to show that the source mesh resolution should be finer than the target mesh, if at possible. Error in our nearby problem ranged from between two to ten percent.

In addition, we found that gaps in the fluid/structure domains (due to modeling differences and simplifications on the respective surfaces) cause unsmoothness in the transferred pressure. This effect may or may not have a strong influence on the solid mechanics simulation results. We suggest an uncertainty quantification study in order to quantify the sensitivity of solid mechanics output to variation in the input pressure.

We also point out that the transfer algorithm is neither locally nor globally conservative. Energy is not conserved in the transfer. As coupled physics modeling and simulation become more prevalent, we advocate further development of a general production level capability for conservative transfers between grids and loosely coupled simulations.

We found the capabilities of the Sierra Encore software and the open source Trilinos Percept software package to be useful in verification studies. We used the mesh refinement capabilities of Percept to easily create relatively fine meshes while minimizing any geometry errors by respecting the CAD model. We also used the software capabilities to define exact analytic functions and compute their differences with discretized numerical fields. This post processing capability, including different orders of quadrature integration in parallel, is an important capability for putting verification into practice.

References

[1] R. Y. Lu, 17x17 V5H VIPER Test and Instrumented Fuel Rod General Mechanical Information Transmittal to CASL Project, Non-proprietary, Westinghouse Electric Company Memo, 2011.

[2] R. Y. Lu, Z. Karoutas, and T. L. Sham, *JOM* **63**, 53 (2011).

[3] J. Wan, V5H Fuel Assembly CAD Model and CFD Model Transmittal, Westinghouse Electric Company Memo, 2010.

[4] S. B. Rodriguez and D. Z. Turner, Sierra/Fuego Calculations to Support Milestone VRI.CPI.Y1.01: Assessment of Existing Capabilities for GTRF, Technical report, CASL VRI, Albuquerque, New Mexico 87185, 2011.

[5] J. N. Shadid, T. M. Smith, R. P. Pawlowski, and E. C. Cyr, A Summary of SNL Drekar Large-scale Parallel CFD Code for use in the Demonstration and Evaluation of Methods for VERA-CFD, Technical report, Sandia National Laboratories, Albuquerque, New Mexico 87185, 2011.

[6] W. J. Rider, Solution Verification Applied to Drekar and Fuego Calculations of Grid-to-Rod-Fretting (GTRF), Technical Report SAND2011-5580C, Sandia National Laboratories, Albuquerque, New Mexico 87185, 2011.

[7] K. D. Copps and B. R. Carnes, Encore User Guide, Technical Report SAND2009-7432, Sandia National Laboratories, Albuquerque, New Mexico 87185, 2009.

[8] M. Heroux, R. Bartlett, V. H. R. Hoekstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, and A. Williams, An Overview of Trilinos, Technical Report SAND2003-2927, Sandia National Laboratories, Albuquerque, New Mexico 87185, 2003.

[9] W. L. Oberkampf, M. Pilch, and T. G. Trucano, Predictive Capability Maturity Model for Computational Modeling and Simulation, Technical Report SAND2007-5948, Sandia National Laboratories, Albuquerque, New Mexico 87185, 2007.

[10] SIERRA Solid Mechanics Team, Sierra/SolidMechanics 4.22 User's Guide, Technical Report SAND2011-7597, Sandia National Laboratories, Albuquerque, New Mexico 87185, 2011.

[11] C. J. Roy, A. Raju, and M. M. Hopkins, *AIAA Journal* **45**, 1232 (2007).

[12] W. L. Oberkampf and C. J. Roy, *Verification and Validation in Scientific Computing*, Cambridge University Press, New York, 2010.

[13] P. J. Roache, *Journal of Fluids Engineering* **124** (2002).

[14] I. Babuška and T. Strouboulis, *The Finite Element Method and Its Reliability*, Oxford University Press, London, 2001.

Appendix A: Input to the Sierra Solid Mechanics Application

This input listing is from a preliminary version of a solid mechanics study for the CASL GTRF problem and may not necessarily reflect more current features in more recent simulations. This listing was used as the input for demonstrating the feature coverage analysis in subsection 3.2.

```
begin sierra cpuzzle

#
#  Analysis time periods
#   0.0->1.0e-3  :  Pre load, move grid into position, load up top spring
#
#
#
#

  begin definition for function ramp
    type is analytic
    ordinate is x
    abscissa is y
    EVALUATE EXPRESSION  = "0.5*(1-cos(x * 100.0 * pi));"
   end

  begin definition for function grid_pre
    type is piecewise linear
    begin values
      0       0
      1.0e-3  1
    end
  end

  begin   definition for function gravity_load_up
    type is piecewise linear
    begin values
      0.0   0.0
     1.0e-3 1.0
    end
  end


  define direction x_dir with vector 1 0 0
  define direction z_dir with vector 0 0 1
  define point zero with coordinates 0 0 0
  define axis x_axis with point zero direction x_dir

  begin property specification for material zirconium
    density         = 10900
    begin parameters for model elastic
      youngs modulus  = 68887e+6
      poissons ratio  = 0.342
    end
  end

  begin property specification for material uranium
    density         = 10980
    begin parameters for model elastic
      youngs modulus  = 1.9e+11
      poissons ratio  = 0.342
```

```
    end
  end

  begin property specification for material spring_mat
    density         = 10980
    begin parameters for model elastic
      youngs modulus  = 1.9e+09
      poissons ratio  = 0.342
    end
  end

  begin rigid body 101
  end

  begin rigid body 102
  end

  begin rigid body 502
  end

  begin solid section rigid_101
    rigid body = 101
  end

  begin solid section rigid_102
    rigid body = 102
  end

  begin solid section rigid_502
    rigid body = 502
  end



  begin spring section spring
    default stiffness = 1.0
    mass per unit length = 1.0e-1
#    preload = -80.0
#    preload duration = 1.0e-3
  end

#  begin definition for function force_strain
#    type is piecewise linear
#    ordinate is force
#    abscissa is engineering_strain
#    begin values
#      -1.0     -1e5
#       1.0      1e5
#    end values
#  end

#  begin superelement section super_xx_2_node
#    begin map
#      1 1
#      2 1
#      1 4
#      2 4
#    end
#    begin stiffness matrix
#      1.0e+05 -1.0e+05 0 0        $ map: 1 1 -> node 1 dof 1
#     -1.0e+05  1.0e+05 0 0        $ map: 2 1 -> node 2 dof 1
```

```
#      0 0  3.0e+02 -3.0e+02       $ map: 1 4 -> node 1 dof 4
#      0 0     -3.0e+02  3.0e+02   $ map: 2 4 -> node 2 dof 4
#    end
#    begin damping matrix
#      1.0e-1 0.0    0.0    0.0
#      0.0    1.0e-1 0.0    0.0
#      0.0    0.0    1.0e-1 0.0
#      0.0    0.0    0.0    1.0e-1
#    end
#    begin mass matrix
#         0.5e-3  0.0 0.0 0.0
#         0.0  0.5e-3 0.0 0.0
#         0.0  0.0 0.5e-3 0.0
#         0.0  0.0 0.0 0.5e-3
#    end
#  end

  begin finite element model mesh1
    Database Name = combined.g
    Database Type = exodusII
    begin parameters for block block_1
      material zirconium
      solid mechanics use model elastic
    end
    begin parameters for block block_101
      material zirconium
      solid mechanics use model elastic
      section = rigid_101
    end
    begin parameters for block block_102
      material zirconium
      solid mechanics use model elastic
      section = rigid_102
    end
    begin parameters for block block_2000 block_2001 block_2002 block_2003
      material zirconium
      solid mechanics use model elastic
    end
    begin parameters for block block_500 block_501
      material uranium
      solid mechanics use model elastic
    end
    begin parameters for block block_502
      material uranium
      solid mechanics use model elastic
      section = rigid_502
    end
    begin parameters for block block_503
      material spring_mat
      solid mechanics use model elastic
      section = spring
    end
  end

  begin presto procedure Apst_Procedure

    begin time control
        begin time stepping block p1
          start time = 0.0
          begin parameters for presto region presto
            step interval = 100
```

```
          end
        end time stepping block p1
        termination time = 10.0
    end time control

  begin presto region presto
    use finite element model mesh1

    ### output description ###
    begin Results Output output_presto
      Database Name = rod3d_explicit.e
      Database Type = exodusII
      At time 0.0, increment = 1.0e-3
      nodal Variables = displacement
      nodal variables = mass
      nodal variables = velocity
      nodal variables = force_internal
      nodal variables = force_contact
      nodal variables = force_external
      element variables = stress as stress
      global Variables = kinetic_energy as ke
      global Variables = internal_energy as ie
      global variables = external_energy as ExternalEnergy
      global variables = momentum as Momentum
      global variables = timestep as timestep

      nodal variables = CONTACT_NORMAL_TRACTION_MAGNITUDE as cnor

      nodal variables = CONTACT_ACCUMULATED_SLIP_VECTOR as slip_vec

      nodal variables = CONTACT_ACCUMULATED_SLIP as slip_mag

      nodal variables = CONTACT_FRICTIONAL_ENERGY as fric_en

      nodal variables = CONTACT_FRICTIONAL_ENERGY_DENSITY as fric_en_dens

    end

    begin fixed displacement
      block = block_101 block_102
      component = xyz
    end
    begin fixed rotation
      block = block_101 block_102
      component = z
    end


#    begin gravity
#      include all blocks
#      direction = z_dir
#      scale factor = -1.0
#      Gravitational Constant = 9.8
#      function = gravity_load_up
#    end


    begin prescribed displacement
      surface = surface_220
      component = x
      scale factor = -0.00001
```

```
      function = grid_pre
    end
    begin fixed displacement
      surface = surface_220
      component = yz
    end

    begin prescribed displacement
      surface = surface_221
      component = y
      scale factor = -0.00001
      function = grid_pre
    end
    begin fixed displacement
      surface = surface_221
      component = xz
    end

    begin prescribed displacement
      surface = surface_222
      component = x
      scale factor = 0.00001
      function = grid_pre
    end
    begin fixed displacement
      surface = surface_222
      component = yz
    end

    begin prescribed displacement
      surface = surface_223
      component = y
      scale factor = 0.00001
      function = grid_pre
    end
    begin fixed displacement
      surface = surface_223
      component = xz
    end


    begin pressure
      surface = surface_1 surface_2 surface_3 surface_4 surface_5 surface_6 surface_7 surface_8
     surface_9 surface_10
      surface = surface_11 surface_12 surface_13 surface_14 surface_15 surface_16 surface_17
     surface_18 surface_19
      node set subroutine = rod_radial_pressure

      subroutine real parameter: axis_origin_x = 0.0
      subroutine real parameter: axis_origin_y = 0.0
      subroutine real parameter: axis_origin_z = 0.0

      subroutine real parameter: axis_dir_x = 0.0
      subroutine real parameter: axis_dir_y = 0.0
      subroutine real parameter: axis_dir_z = 1.0

      subroutine real parameter: outer_radius = 0.00475
      subroutine real parameter: f1 = 5
      subroutine real parameter: f2 = 100
      subroutine real parameter: amp = 7.82801e-6
      subroutine real parameter: p0 = 50.0
```

```
      subroutine integer parameter: num_strip = 5
      subroutine real parameter: span = 0.521970
    end

    begin contact definition
      contact formulation type = dash

      compute contact variables = on

      contact surface pellet1 contains block_500
      contact surface pellet2 contains block_501
      contact surface top_pellet contains block_502
      contact surface grid contains block_2000 block_2001 block_2002 block_2003
      contact surface clad contains block_1 block_101 block_102

      begin interaction defaults
        general contact = on
        friction model = fric
      end
      begin constant friction model fric
        friction coefficient = 0.3
      end
    end

    #end node+3, tied to pellet
    begin MPC
      tied nodes = 10981 30461
    end

    #end node+1 tied to end cap
    begin MPC
      tied nodes = 10982 30459
    end


  end presto region presto
  end presto procedure Apst_Procedure
end
```

Appendix B: Input for Performing Transfer of Transient Pressure Field

This listing is the same input file used to transfer the transient fluid results to the solid mechanics models in actual simulations run during recent prototype GTRF analyses. This is input to the Sierra Mechanics Encore[7] application.

```
Begin Sierra Encore

    Title Tests Transfer from Coarse to Fine

    Begin Finite Element Model source_mesh
        Database Name = flow_solution_joined.e
    End

    Begin Finite Element Model target_mesh
        Database Name = grid12rod_sep25.g
    End

    Begin Encore Procedure encore_procedure

        Begin Solution Control Description
            Use System main
            Begin System main
                Begin Transient encore_trans
                    Advance source_region
                    Transfer source_to_target
                    Advance target_region
                End

                Simulation Start Time = 0
                Simulation Termination Time  =  0.397630
                Simulation Max Global Iterations = 100000 #Arbitrarily large
            End
        End

        begin Transfer source_to_target
            Interpolate surface Nodes From source_region To target_region
            search coordinate field model_coordinates state none to model_coordinates state none
            send block block_1 block_3 block_100 block_300 to surface_67 surface_69
            nodes outside region = extrapolate
            Send Field fluid_pressure State New To pressure State New
            From nodes to elements
        end

        Begin Encore Region source_region
            Use Finite Element Model source_mesh Model Coordinates Are model_coordinates
            Import Field solution->pressure as Nodal Field fluid_pressure
            Disable Compute Timestep
        End

        Begin Encore Region target_region
           Use Finite Element Model target_mesh Model Coordinates Are model_coordinates

            # verify this
            Constant Timestep Is 5.0E-4

            Create Face Field pressure Of Type REAL and dimension 1 On surface_67 surface_69

```

```
            Begin Results Output output
                database name = fluid_loads.e
                database type = exodusII
                at step 0 increment = 1
                face variables = pressure
            End results output output
        End

    End Encore Procedure
End Sierra
```

Appendix C: Input to Generate a Grid Including CAD Geometry

This listing is a journal of the input we used to the CUBIT software to generate a cylindrical grid for the fuel rod. In lines 40–41 are the commands to output the `*.3dm` file containing the definition of the CAD geometry as well as the coarse mesh. The Percept refine command can use the CAD geometry in the `*.3dm` file to place all new nodes on the original curved geometrical surfaces.

```
1 reset
2 create Cylinder height 0.52197 radius 0.0047498
3 move Volume 1  x 0 y 0 z 0.260985 include_merged
4 webcut volume 1  with plane zplane offset 0.20940615 imprint merge
5 webcut volume 1  with plane zplane offset 0.37906415 imprint merge
6 ##move Volume 1 2 3  x 0 y 0 z 0.00302885 include_merged
7 move Volume 1 2 3  x 0 y 0 z -0.20940615 include_merged
8 surface 2 interval 16
9 surface 2 scheme circle
10 mesh surface 2
11 surface 3 interval 16
12 surface 3 scheme circle
13 mesh surface 3
14 surface 4 interval 16
15 surface 4 scheme circle
16 mesh surface 4
17 surface 8 interval 16
18 surface 8 scheme circle
19 mesh surface 8
20 volume 1  interval 50
21 mesh volume 1
22 volume 3  interval 55
23 mesh volume 3
24 volume 2  interval 61
25 mesh volume 2
26
27 # element block 1
28 set duplicate block elements off
29 block 1 volume 1 2 3
30
31 # refinements
32 refine volume 1 2 3 numsplit 1
33 refine volume 1 2 3 numsplit 1
34 refine volume 1 2 3 numsplit 1
35
36 # sideset
37 Sideset 1 surface 11
38
39 # output mesh and geometry
40 set dev on
41 refine parallel No_execute
42
43 #export mesh "/Users/kdcopps/Documents/CASL/GTRF pressure/cyl_lvl3.e" overwrite
```

Sandia Internal Distribution

1	MS 1318	1441	J. R. Stewart
1	MS 1318	1441	B. M. Adams
1	MS 1318	1441	M. S. Eldred
1	MS 1318	1441	J. R. Kamm
1	MS 1318	1441	P. Knupp
1	MS 1318	1441	T. G. Trucano
1	MS 1320	1441	V. G. Weirs
1	MS 1320	1442	T. M. Smith
1	MS 1323	1443	W. J. Rider
1	MS 1321	1444	R. M. Summers
1	MS 1321	1444	J. N. Shadid
1	MS 0380	1540	D. E. Womble
1	MS 0828	1541	R. B. Bond
1	MS 0828	1541	S. P. Domino
1	MS 0828	1541	D. Z. Turner
1	MS 0380	1542	J. Jung
1	MS 0380	1542	M. W. Heinstein
1	MS 0380	1542	N. K. Crane
1	MS 0828	1544	K. F. Alvin
1	MS 0828	1544	B. Carnes
4	MS 0897	1544	K. D. Copps
1	MS 0828	1544	W. R. Witkowski
1	MS 0372	1545	M. W. Glass
1	MS 0899	9532	RIM-Reports Management (*electronic copy*)

External Distribution

Department of Energy Laboratories

Oak Ridge National Laboratory (4)
P.O. Box 2008
Oak Ridge, TN 37831

Attn:	Sam Sham,	MS 6155
	Prateek Nath,	MS 6155
	Jeff Banta,	MS 6003
	Douglas B. Kothe,	MS 6003

Sandia National Laboratories