INTRODUCTION

Examples of multiphysics problems in the field of nuclear engineering are numerous. For light water reactors the coupling of neutronics with thermal-hydraulics is the most predominant. Often this coupling is modeled by combining existing codes in a Picard Iteration scheme, that is, by solving one physics then using that answer to solve the other and iterating until both answers stabilize. However, simply solving to a high precision on every Picard (multiphysics) iteration is much less efficient than progressively solving to higher precisions. This unneeded work is known as over-solving.

Over-solving in single-physics solvers has been studied previously for the Newton method [1] and Chebyshev iteration [2]. Both of these methods solve an inner linear problem that determines the next update step for the outer loop. If the inner problem is solved to a high precision in every outer iteration, the result will be substantially more computational expense (over-solving) and a higher probability of reaching the “wrong” solution [1]. Methods for reducing over-solving in partitioned multiphysics codes have only been developed recently [3, 4]. This work introduces further improvements, including an investigation of over-solving methods on both the multiphysics and single-physics levels.

DESCRIPTION OF THE ACTUAL WORK

Coupling Methods

Picard Iteration (PI) is a partitioned coupling method commonly used to solve multiphysics problems. For each time step the constituent (single-physics) solvers are solved repeatedly until their feedback effects have been resolved. Picard Iteration is a preferred method because it can be relatively easy to implement and existing solvers can be used with only minor modifications.

A notable weakness of Picard Iteration is an inefficiency known as over-solving. The overall accuracy of the simulation is initially quite poor, therefore it is inefficient to solve to a tight tolerance. The constituent solvers only need to achieve higher precision as the simulation progresses.

In our previous study [5], the Residual Balance (RB) method was developed as a modification of Picard Iteration in order to mitigate over-solving in multiphysics problems. This method applies to multiphysics simulations in which the constituent solvers have an iterative structure and they provide the norm of the solution residual. This is indeed the case for the Jacobian-Free Newton-Krylov finite element solvers implemented in the MOOSE framework [6] that we use for this work. Throughout this work, norms (denoted by \( \| \cdot \| \)) refer exclusively to the \( L_2 \), or Euclidean, norm.

The RB method works by adjusting the tolerance of the constituent solver to a level just below the initial residual of the other solvers. This ensures that each solver catches up to the progress made by the other solvers. For a problem with two constituent solvers (S1 and S2) the termination criteria are adjusted to

\[
\| R_{1,j}^{S_1} \| < a \| R_{1-1,0}^{S_2} \|, \quad (1a)
\]

\[
\| R_{1,j}^{S_2} \| < a \| R_{1-1,0}^{S_1} \|, \quad (1b)
\]

where the index \( i \) refers to the Picard iteration and \( j \) refers to the constituent iteration. A user-specified value of \( a = 0.1 \) often works quite well [3, 4], but an automatic scheme for choosing the best value for this parameter is desirable. In this paper, we present a new scheme that does this based on the Picard (multiphysics) convergence rate.

Improved Schemes

The effective convergence rate of the coupled problem can be estimated by the ratio of the current and previous combined residual norm,

\[
\rho_i = \frac{\| R_{1,i}^{\text{tot}} \|}{\| R_{1,i-1}^{\text{tot}} \|}, \quad (2)
\]

where \( \| R_{1,i}^{\text{tot}} \| \) is evaluated at the beginning of each Picard iteration as

\[
\| R_{1,i}^{\text{tot}} \| = \| R_{1,i}^{S_1} \| + \| R_{1,i}^{S_2} \|. \quad (3)
\]

\( J \) is the number of constituent iterations needed to (inexactly) solve S2, thus \( \| R_{1-1,0}^{S_2} \| \) is the final residual norm of the second solver in Picard iteration \( i-1 \). The estimated convergence rate \( \rho_i \) can be used to project the appropriate solver tolerance for the current Picard iteration. In order to maintain this rate of convergence, the active solver must drive its residual norm at least a factor of \( a = \rho_i \) below the initial residual norm of the other solver (i.e. \( \| R_{1-1,0}^{S_2} \| \)). If \( \rho_i \) starts off too large (i.e. slower than the asymptotic convergence rate) the result may be extra Picard iterations (known as under-solving). In order to push \( \rho_i \) toward the asymptotic convergence rate, the reduction factor \( \alpha = \rho_i / 2 \) is used. A thorough investigation of the optimal reduction factor is needed and will be part of future efforts. Thus we have,

\[
\| R_{1,j}^{S_1} \| < \frac{\rho_i}{2} \| R_{1-1,0}^{S_1} \|, \quad (4a)
\]

\[
\| R_{1,j}^{S_2} \| < \frac{\rho_i}{2} \| R_{1-1,0}^{S_2} \|. \quad (4b)
\]

After one Picard iteration, the estimate for the convergence rate \( \rho \) is available. With the logic of Equation 4, the appropriate termination criterion will be selected without further user input. This is much more convenient than previous work [4], because the user is no longer held responsible for selecting a good value for \( a \).
The RB method deals with over-solving at the multi-physics level, but nested solvers may also suffer from over-solving at the constituent/single-physics level. We now turn our attention to one technique for mitigating over-solving in Newton solvers. Newton methods solve \( R(x) = 0 \) by updating the solution via \( x_{j+1} = x_j + \Delta x_j \). For inexact Newton Methods, the update step \( \Delta x_j \) is accepted when

\[
\| R(x_j) + R'(x_j) \Delta x_j \| \leq \eta_j \| R(x_j) \|. \tag{5}
\]

\( R' \) is the Jacobian of the residual/function, and \( \eta \) is known as the “forcing term.” Eisenstat and Walker’s (EW) termination criterion [1] is given by

\[
\eta_j = \gamma \rho_j^\alpha, \tag{6}
\]

where \( \rho \) is analogous to Equation 2, \( \gamma \in [0, 1] \), and \( \alpha \in (1, 2] \). In this work, the EW method uses the values \( \eta_0 = 0.1 \), \( \gamma = 1 \), and \( \alpha = \frac{1+\sqrt{5}}{2} \) (see [1]); whereas \( \eta_1 = 10^{-4} \) is used otherwise. In the next section, a reactor transient example problem is solved to investigate whether applying inexact methods to multiple levels of a nested solver (RB at the multiphysics level, and EW at the single-physics level) is an efficient scheme.

Example Problem

The LRA BWR benchmark [7, 8] is used to demonstrate the performance of the Residual Balance method with its latest improvements in comparison to standard Picard Iteration. The LRA benchmark problem models a control rod ejection accident in a boiling water reactor which results in a super-prompt critical reactivity excursion. Cross sections for two-group diffusion are provided as well as a model for adiabatic heat up and Doppler feedback. The reactor can be modeled in either two or three dimensions. In this work the two-dimensional version is used. The governing equations are as follows

\[
\frac{1}{v_1} \frac{\partial \phi_1}{\partial t} - \nabla D_1 \nabla \phi_1 + (\Sigma_{s1} + B^2 D_1 + \Sigma_{i-2}) \phi_1
\]

\[
- \nu(1-\beta)(\Sigma_{f1} \phi_1 + \Sigma_{f2} \phi_2) - \lambda_1 C_1 - \lambda_2 C_2 = 0, \tag{7}
\]

\[
\frac{1}{v_2} \frac{\partial \phi_2}{\partial t} - \nabla D_2 \nabla \phi_2 + (\Sigma_{s2} + B^2 D_2) \phi_2 - \Sigma_{i-2} \phi_1 = 0, \tag{8}
\]

\[
\frac{\partial C_i}{\partial t} + \lambda_i C_i - \nu \beta (\Sigma_{f1} \phi_1 + \Sigma_{f2} \phi_2) = 0; \quad i = 1, 2, \tag{9}
\]

\[
\frac{\partial T}{\partial t} - \alpha (\Sigma_{f1} \phi_1 + \Sigma_{f2} \phi_2) = 0. \tag{10}
\]

Further specifications can be found elsewhere [7, 8].

By using the multi-app system in MOOSE, the problem can be solved with partitioned methods. Equations 7, 8 and 9 are treated simultaneously in a neutronics solver, and the other partition consists of solving Equation 10. Although this problem is quite simplified, it still has the features required to demonstrate multiphysics coupling methods.

The transient is solved with direct numerical simulation using the second order backward difference formula for time integration. The neutron diffusion and delayed neutron precursor equations are solved on a fine mesh of 7744 linear finite elements—64 finite elements per fuel assembly. The power distribution is transferred to the heat transfer solver with a conservative \( L_2 \) projection method. In order to match the benchmark, the temperature is represented by a constant value for each assembly-sized element (78 in total, see Figure 1). The temperature is transferred to the neutron diffusion solver by direct sampling (interpolation). Nested within the Picard Iteration multiphysics solver, both single-physics solvers use the Jacobian-Free Newton-Krylov method preconditioned with LU factorization. Certainly this problem does not require these methods, but it still has enough complexity that the their performance behavior can be used to establish informative trends.

**RESULTS**

To evaluate the problem’s implementation, the results are compared to other published results [8, Table 4]. The eigenvalue is 0.99639 which lies within the reported bounds of 0.99631 and 0.99655. The time to peak power (1.432 s) and the peak power (5581 W/cm³) also fall in the range of the other results (1.426–1.445 s and 5451–5734 W/cm³). Figure 2 shows the core-averaged power density and temperature for the 3 seconds of the transient.

![Fig. 1. Final temperature distribution with assembly-sized mesh elements.](Image)

![Fig. 2. Core-averaged behavior.](Image)
TABLE I. Performance comparison for the LRA benchmark problem with inexact solvers at various levels.

<table>
<thead>
<tr>
<th>Method</th>
<th>Picard</th>
<th>Constituent</th>
<th>Linear</th>
<th>CPU Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>PI</td>
<td>841</td>
<td>2522</td>
<td>3223</td>
<td>1</td>
</tr>
<tr>
<td>PI + EW</td>
<td>841</td>
<td>2704</td>
<td>3658</td>
<td>1.124</td>
</tr>
<tr>
<td>RB</td>
<td>866</td>
<td>1758</td>
<td>2407</td>
<td>0.649</td>
</tr>
<tr>
<td>RB + EW</td>
<td>840</td>
<td>1679</td>
<td>1679</td>
<td>0.608</td>
</tr>
</tbody>
</table>

The performance of various inexact methods are compared next. Inexact methods—either the Residual Balance method in the multiphysics loop, or the inexact Newton method in the single-physics loop—can be applied to reach the solution with less computational cost. Every method comes to the same solution because the outermost (Picard) tolerance is the same.

There are several interesting results found in Table I. First, Picard Iteration does not benefit from the EW method using these parameters. The initially small values of $\eta_j$ cause under-solving at the single-physics level. Those additional constituent iterations result in additional linear iterations, and all this adds up to 12.4% more wall-clock time. Comparing the Residual Balance method to Picard Iteration, a large reduction in computer time (35%) is driven by a reduction in the number of constituent iterations, as shown in Figure 3. However, some under-solving at the multiphysics level is apparent by the increase in Picard iterations. Both Table I and Figure 4 show that this is a minor issue. Interestingly, when EW is added to RB it reduces both constituent and Picard iterations. This is most likely due to the fact that EW keeps the solution from going too far in the wrong direction when the underlying linear model is a poor representation of the actual function. Overall, the combination of RB and EW gets to the correct solution 1.64 times faster than PI.

CONCLUSIONS

Over-solving can be dealt with effectively using the Residual Balance method. The potential performance benefit is problem dependent, but for this reactivity insertion example the computational time is reduced by 35% (1.54x speedup). A further benefit is obtained by pairing the Residual Balance method with an inexact Newton method at the single-physics level. Not only does it improve performance by 6.3%, it also reduces the number of Picard iterations. The two inexact methods are not just compatible, but synergistic as well.

ACKNOWLEDGMENTS

This material is based upon work supported by a Department of Energy, Office of Nuclear Energy, Integrated University Program Graduate Fellowship.

REFERENCES