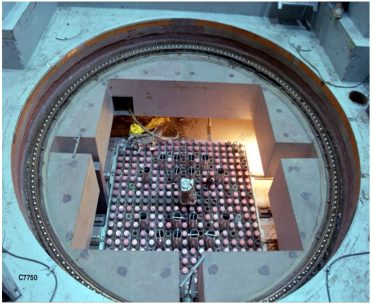


TREAT Support: Improved Quasi-Static Method

Zachary M. Prince, Jean C. Ragusa

Department of Nuclear Engineering, Texas A&M University, College Station, TX

November 22, 2016

email: zachmprince@tamu.edu

Outline

1. IQS Review
   - Equations
   - Process
   - Transient-15
   - M8CAL
2. Step Doubling
   - Process
   - LRA
   - Tran15
3. Multiphysics Updates
   - Process
   - LRA
4. IQS Refactoring
5. Wrap-up

IQS Traditional

Factorization

$$\phi^g(\vec{r},t) = p(t)\varphi^g(\vec{r},t)$$

Shape equations

$$\frac{1}{v^g}\frac{\partial \varphi^g}{\partial t} = \frac{\chi_p^g}{k_{eff}}\sum_{g'=1}^{G}(1-\beta)\nu^{g'}\Sigma_f^{g'}\varphi^{g'} - \left(-\vec{\nabla}\cdot D^g\vec{\nabla} + \Sigma_r^g + \frac{1}{v^g}\frac{1}{p}\frac{dp}{dt}\right)\varphi^g$$

$$+\sum_{g'\neq g}^{G}\Sigma_s^{g'\to g}\varphi^{g'} + \frac{1}{p}\sum_{i=1}^{I}\chi_{d,i}^g\lambda_i C_i\ , \quad 1\leq g\leq G$$

$$\frac{dC_i}{dt} = p\sum_{g=1}^{G}\nu_{d,i}\Sigma_f^g\varphi^g - \lambda_i C_i, \quad 1\leq i\leq I$$

PRKE

$$\frac{dp}{dt} = \left[\frac{\rho-\bar{\beta}}{\Lambda}\right]p + \sum_{i=1}^{I}\bar{\lambda}_i\xi_i$$

$$\frac{d\xi_i}{dt} = \frac{\bar{\beta}_i}{\Lambda}p - \bar{\lambda}_i\xi_i \quad 1\leq i\leq I$$

IQS Predictor-Corrector

Predicted Flux → Corrected Flux

IQS P-C linearizes the system and avoids iterations on the shape:

1. Evaluate multigroup diffusion equation to get predicted flux $\phi_{n+1}^{g,pred}$
2. Scale predicted flux to obtain shape:

$$\varphi_{n+1}^{g} = \phi_{n+1}^{g,pred} \frac{\sum_{g=1}^{G} \left(\phi^{*g}, \frac{1}{v^g}\phi_0^g\right)}{\sum_{g=1}^{G} \left(\phi^{*g}, \frac{1}{v^g}\phi_{n+1}^{g,pred}\right)} = \phi_{n+1}^{g,pred} \frac{K_0}{K_{n+1}}$$

3. Compute PRKE parameters at t_{n+1}
4. Evaluate PRKE along micro step using interpolated parameters to obtain p_{n+1}
5. Scale φ_{n+1}^{g} to obtain corrected flux:

$$\phi_{n+1}^{g,corr} = p_{n+1} \times \varphi_{n+1}^{g}$$

Solution Process

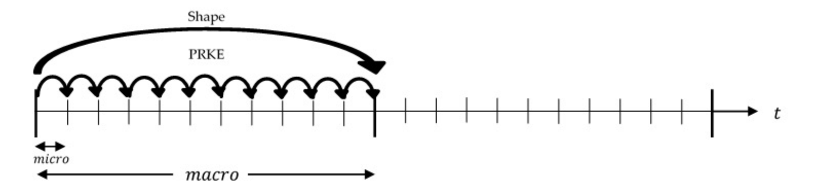
Factorization leads to a nonlinear system

The amplitude and shape equations form a system of nonlinear coupled equations:

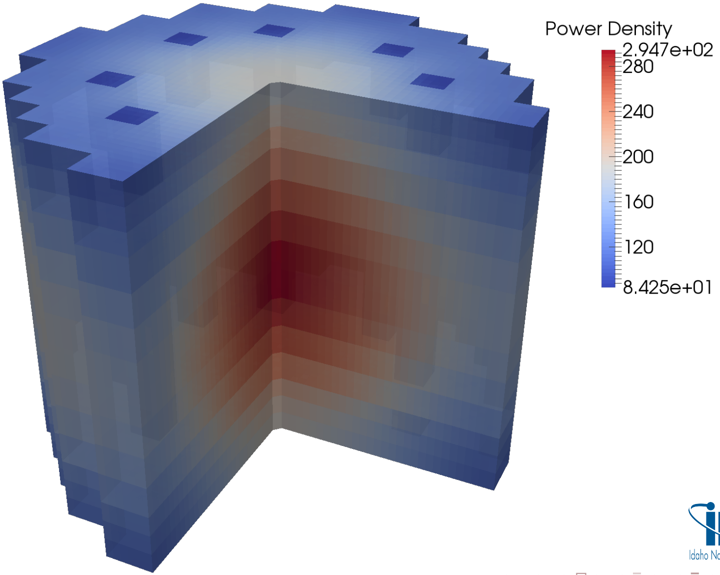
1. the coefficients appearing in the PRKE's depend upon the shape solution,
2. the shape equation has a kernel dependent on amplitude and its derivative,

Time scales and IQS method solution process

Because solving for the shape can be expensive, especially in two or three dimensions, it is attractive to make the assumption that the shape is weakly time-dependent so the shape can be computed after a multitude of PRKE calculations:



TREAT: Transient-15



Tran15 Results

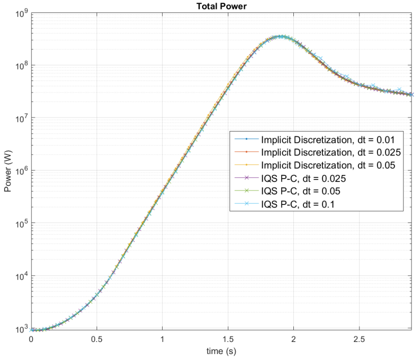


Figure: Tran15 Power Profile

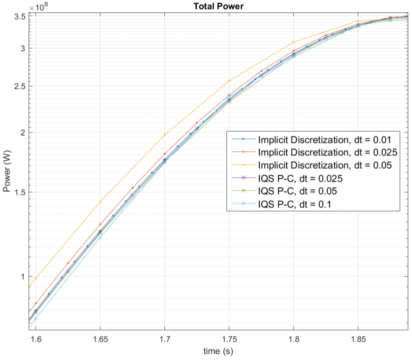


Figure: Tran15 Peak Power Profile

TREAT: M8CAL

Figure: M8CAL Power Profile

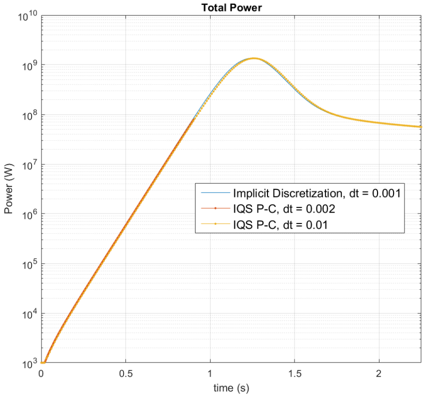
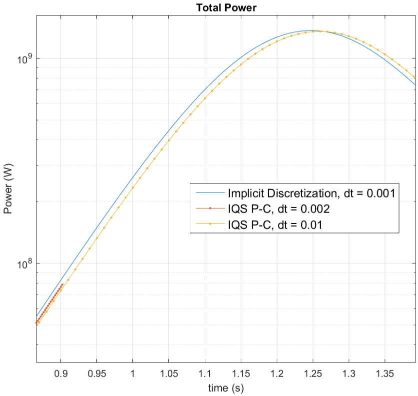


Figure: M8CAL Peak Power Profile

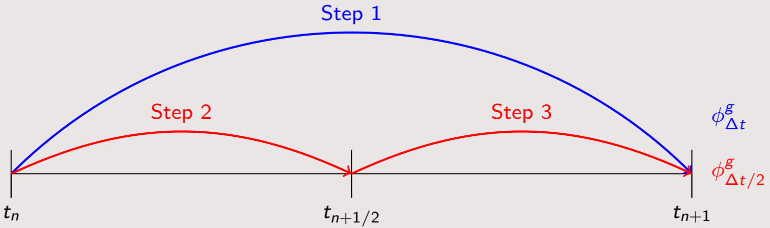


Step Doubling

1. IQS Review
   - Equations
   - Process
   - Transient-15
   - M8CAL
2. Step Doubling
   - Process
   - LRA
   - Tran15
3. Multiphysics Updates
   - Process
   - LRA
4. IQS Refactoring
5. Wrap-up

Step Doubling Solution Process

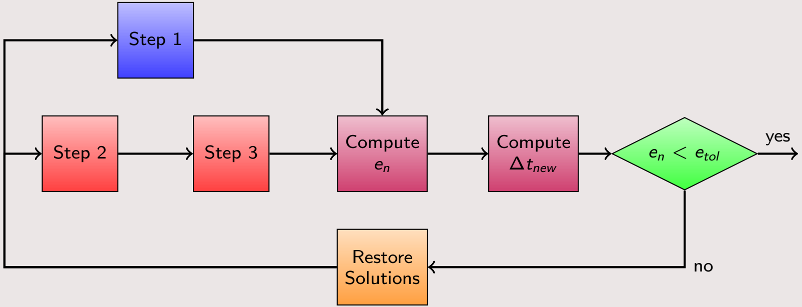
Solution Process with IQS



$$e_n = \frac{\left\| \sum_{g=1}^{G} \left(\phi^g_{\Delta t/2} - \phi^g_{\Delta t} \right) \right\|_{L^2}}{\max\left(\left\| \sum_{g=1}^{G} \phi^g_{\Delta t/2} \right\|_{L^2}, \left\| \sum_{g=1}^{G} \phi^g_{\Delta t} \right\|_{L^2} \right)} \qquad \Delta t_{new} = S_f \Delta t \left[\frac{e_{tol}}{e_n} \right]^{1/(p+1)}$$

Step Doubling Solution Process

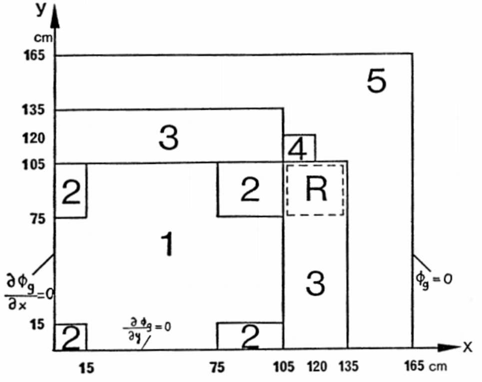
Programming Visualization

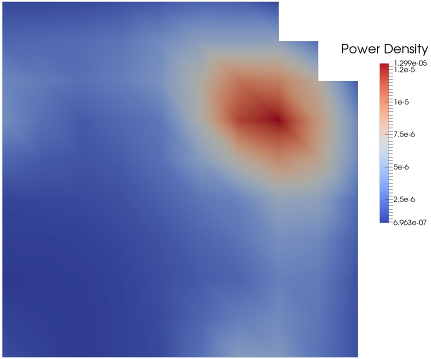


Each Step undergoes:

- Shape evaluation
- PRKE evaluations
- Multiphysics evaluations
- Iterations for convergence of amplitude, shape, and multiphysics

LRA Benchmark





LRA Results

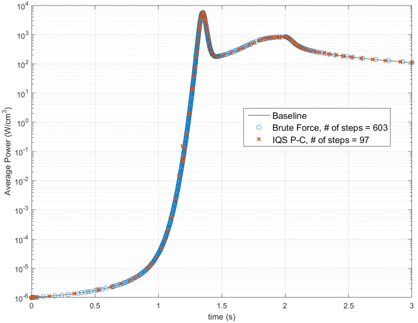


Figure: LRA Power Profile

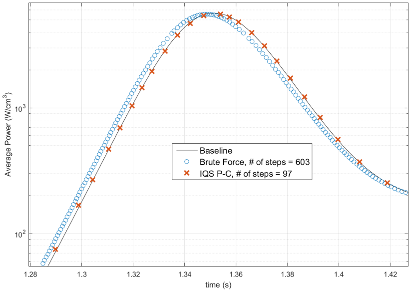


Figure: LRA Peak Power Profile

	Brute Force			IQS P-C		
Event	Power (W/cm^3)	Error	Steps	Power (W/cm^3)	Error	Steps
Max Power	5567.3	0.019454	423	5568.3	0.019274	47
End (3 s)	109.66	2.3650e-4	603	109.65	3.0622e-4	97

Tran15 Results

Figure: Tran15 Power Profile

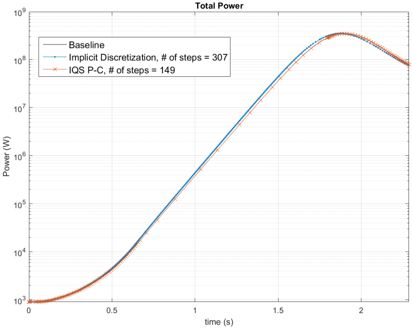
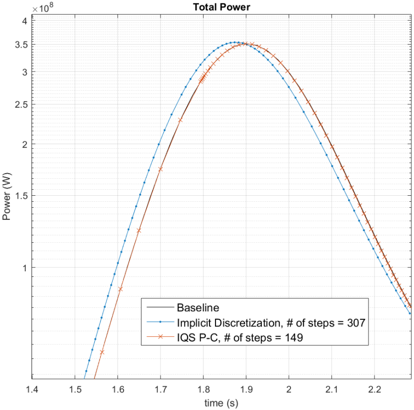


Figure: Tran15 Peak Power Profile



Multiphysics Updates

1. IQS Review
   - Equations
   - Process
   - Transient-15
   - M8CAL
2. Step Doubling
   - Process
   - LRA
   - Tran15
3. Multiphysics Updates
   - Process
   - LRA
4. IQS Refactoring
5. Wrap-up

Motivation

Figure: LRA convergence at $t = 1.44s$

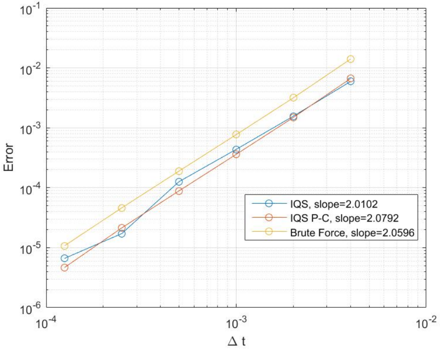
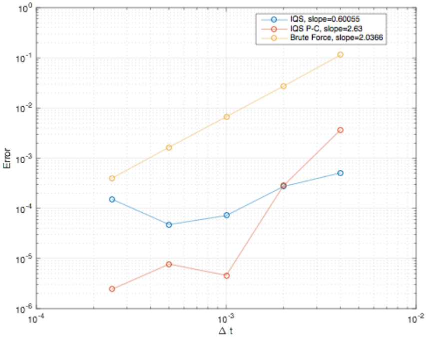
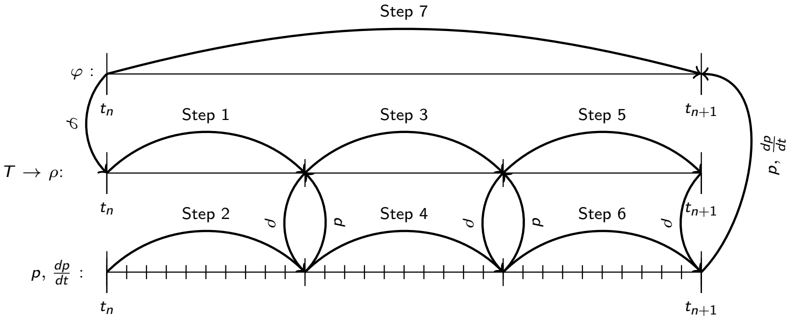


Figure: LRA convergence at $t = 1.40s$



Multiphysics Timescale



LRA with New Timescale

Figure: LRA multiphysics updates convergence

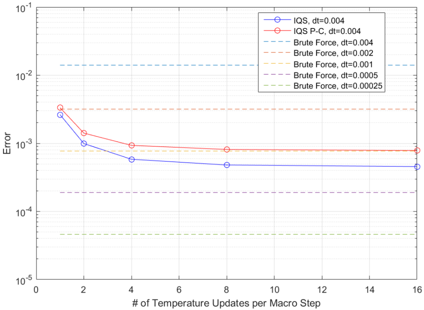
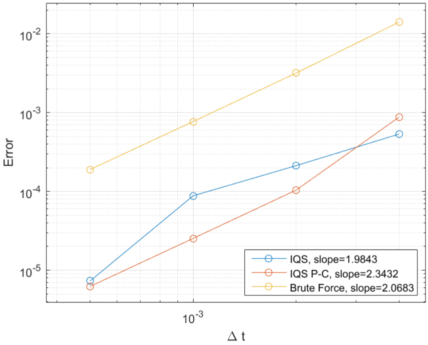


Figure: LRA convergence at $t = 1.44s$

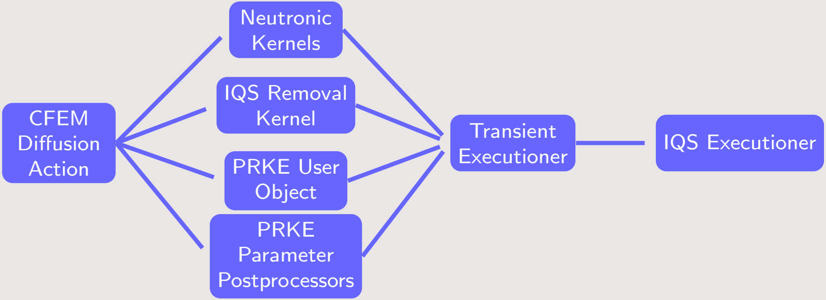


IQS Refactoring

1. IQS Review
   - Equations
   - Process
   - Transient-15
   - M8CAL
2. Step Doubling
   - Process
   - LRA
   - Tran15
3. Multiphysics Updates
   - Process
   - LRA
4. IQS Refactoring
5. Wrap-up

Executioner, User Object, Postprocessor, Postprocessor, Postprocessor, ...

Current Implementation

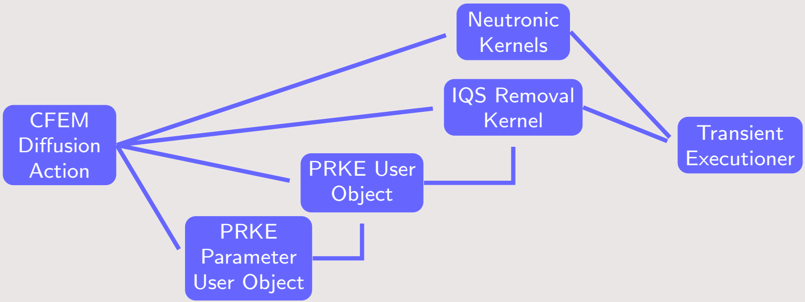


What's Wrong?

- Multitudinous postprocessors
- Weakly defined ρ with `save_in`
- Lots of duplicate code between Transient and IQS executioner

User Objects

Proposed Implementation



What's Good?

- Less files and duplicate code
- IQS iteration at PFJNK level (Picard currently)
- Easier integration of other transport systems

Idaho National Laboratory

Wrap-up

1. IQS Review
   - Equations
   - Process
   - Transient-15
   - M8CAL
2. Step Doubling
   - Process
   - LRA
   - Tran15
3. Multiphysics Updates
   - Process
   - LRA
4. IQS Refactoring
5. Wrap-up

IQS Update

Honey-Done List

- Initial IQS implementation to TREAT examples
- IQS testing with step doubling
- Initial thoughts on IQS multiphysics

Honey-Do List

- Waiting on MOOSE to finish refactoring time steppers so Rattlesnake can have step doubling
- IQS multiphysics with multi-apps in mind
- Refactor IQS executioner to user object

Questions about IQS?

Thank you

- Yaqi Wang (INL, Rattlesnake lead)
- Mark DeHart (INL, TREAT M&S lead)
- NEAMS

What people think when I say I work on TREAT

Idaho National Laboratory

Uh oh ... More? Performance Analysis on Tran15

Computing Time

96 CPUs, 65 Time Steps

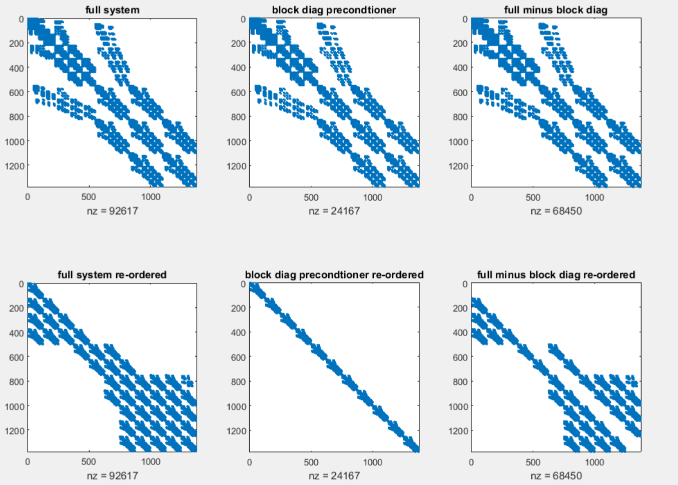
Process	Time (hr)	Time per Step (sec)	% of Time
`compute_residual()`	6.87	381	36%
`solve()`	6.66	369	35%
`update_aux_vars_elemental()`	5.01	277	27%
Total	18.85	1044	100%

Number of Iterations

- Number of Steps: 65
- Number of Nonlinear iterations per step: 3
- Number of Linear iterations per step: 180
- Number of Linear iterations per Nonlinear iteration: 60

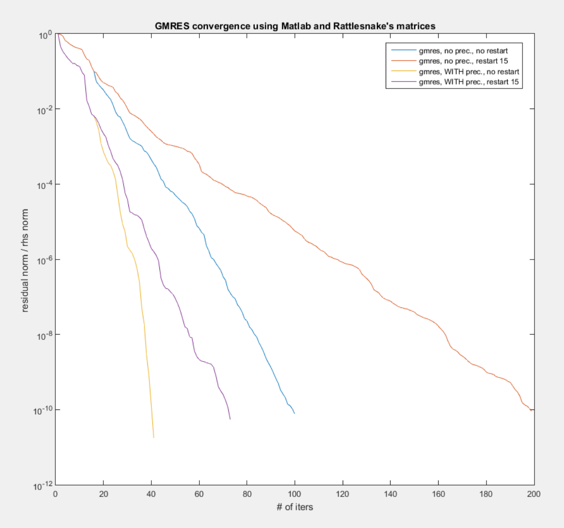
Preconditioning

Jacobian Sparsity



Preconditioning

GMRES Convergence



Whew!