High-Order Time Discretization and Nonlinear Iteration Analysis of the Improved Quasi-Static Method Time-adaptive Multiphysics Reactor Simulations based on the Improved Quasi-Static Method with high-order temporal discretization Multiphysics Reactor-core Simulations Using the Improved Quasi-Static Method

Zachary M. Prince[a], Jean C. Ragusa[a]

[a] *Texas A&M University, Department of Nuclear Engineering, College Station, TX 77840, USA*

Abstract

The improved quasi-static method (IQS) is a rigorous space/time multiscale approach whereby the neutron flux is represented by a time-dependent amplitude and a time-space-energy dependent shape. The objective of the IQS factorization is to evaluate amplitude and shape on different time scales in order to reduce computational burden associated with solving the multi-dimensional flux equations, while maintaining solution accuracy. IQS factorization leads to a nonlinear system of equations that requires iteration of shape and amplitude. IQS iteration techniques involve fixed-point (Picard) iteration with various convergence criteria and Newton iteration, namely preconditioned Jacobian-free Newton Krylov (PJFNK) method. Nonlinear convergence of each of these techniques is investigated. Verification of IQS with analysis of time step convergence is also investigated. Proper time step convergence is vital for implementation of time adaptive methods and error prediction. The time derivative of the shape function is discretized through fourth order using implicit-Euler, Crank-Nicholson, backward difference formulae (BDF), and singly-diagonally-implicit Runge-Kutta (SDIRK) methods.

Keywords: Reactor dynamics, reactor kinetics, Improved quasi-static method, temporal convergence

Zach: todo!
I change the template to that of an elsevier journal but I get a ton of bibtex errors. can you look into th

Email addresses: zachmprince@tamu.edu (Zachary M. Prince), jean.ragusa@tamu.edu (Jean C. Ragusa)

1. Introduction

The improved quasi-static method (IQS) is a numerical technique devised for nuclear reactor transient analysis. It involves factorizing the neutron flux solution into a time-only-dependent component, the amplitude, and a space- and time-dependent component, the shape [**?** **?** **?** **?** **?**]. The amplitude solution satisfies the point reactor kinetic equations (PRKE) where the shape solution have been used to generate the PRKE coefficients (reactivity, effective fraction of delayed neutrons, mean generation time). The shape solution satisfies a modified time-dependent neutron balance equation. The rationale for the IQS method lies in the assumption that the shape function is weakly dependent on time. Therefore, it is expected that the modified time-dependent neutron balance equations be less stiff than the original time-dependent neutron balance equations for the flux. As a result, the shape may not require to be solved for at the same frequency as the amplitude, but only on larger macro-time steps, which is expected to yield wall clock savings, especially in multi-dimensional geometries. The PRKE form a small system of ordinary differential equations and solving them on a fine temporal grid is not a computational burden. As opposed to the standard PRKE approach, whereby a shape function is selected, typically once for an entire transient without updates, the IQS technique is obtained in a rigorously consistent manner from the time-dependent neutron balance equation.

Due to the factorization of the flux into a shape and an amplitude, the latter two variables are nonlinearly coupled. Ott in [**?**] first investigated the coupling of shape and amplitude in a quasi-static nature, but did not include the time derivative in the shape equation. Later, in [**?**], Ott incorporated the time derivative of shape in the equation, yielding better results but requiring a fixed-point approach to resolve the nonlinearities. This also led to the technique's name: the Improved Quasi-Static method although one may argue that such a denomination does not make it immediately clear that the technique fully solves a time-dependent problem.

Nonlinear problems are solved in an iterative manner, typically using either a fixed-point (Picard) approach or Newton's iterations. Sissaoui et al. [**?**], Koclas et al. [**?**], Devooght et al. [**?**], and Monier [**?**] all use fixed-point iterative techniques for their IQS simulations, the main difference among them being their criteria for convergence. Devooght et al. in [**?**] proposed a Newton-based iteration technique. This paper discusses the different nonlinear iteration techniques and tests the rigor of their implementation. IQS can also be linearized by using the IQS Predictor-Corrector method (IQS P-C) [**?**]. IQS P-C entails evaluating the flux equation then correcting its amplitude using an amplitude evaluation. This method has proven to be effective for problems requiring a significant amount of shape updates when IQS is implemented [**?**].

This? Ours? his? I suppose you mean ours. If so, you need to complete the lit review by talking about IQS-PC from Dulla and then their time-adaptive IQS work. After that, start a new paragraph focusing on your paper

IQS has yet to be rigorously tested with high-order discretization Question jcr→zep spatial discretization? of the shape equation. All all? publications that test IQS do not go beyond first-order time discretization I thought the Canadians used theta scheme, hence crank nicholson, hence 2nd Testing IQS with higher-order schemes for proper time step error convergence is essential for implementation of time adaptive methods and quantification of model uncertainty explain a bit or rephase. This paper discusses implementation of up though fourth-order time discretization for the shape equation of IQS and the flux equation of IQS P-C. Step doubling time adaptation is also implemented to test IQS and IQS P-C performance with adaptation of shape/flux evaluation.

To test IQS and IQS P-C nonlinear iteration, time step convergence, and time adaptation performance (five?) problems were applied. (Three?) problems are purely neutronics and involve varying magnitudes of reactor size and complexity. Two problems involve temperature feedback and test an intermediate time scale for temperature evaluation.

2. Background Theory on IQS

In this Section, we recall the equations for the IQS method, starting from multi-group neutron conservation statements in operator form:

$$\frac{1}{v^g}\frac{\partial \phi^g}{\partial t} = \sum_{g'=1}^{G} \left(H^{g'\to g} + P_p^{g'\to g} \right) \phi^{g'} - L^g \phi^g + S_d^g \,, \tag{1a}$$

$$\frac{dC_i}{dt} = \sum_{g=1}^{G} P_{d,i}^g \phi^g - \lambda_i C_i \;, \quad 1 \le i \le I \,. \tag{1b}$$

where $H^{g'\to g}$ is the scattering operator, $P_p^{g'\to g}$ is the prompt fission operator, L^g is the leakage and interaction operator, S_d^g is the delayed neutron source, and $P_{d,i}^g$ is the delayed-neutron fission operator. We assumed G neutron groups and I precursors groups. The operator L^g is block diagonal in energy groups: for neutron transport, $L^g = \mathbf{\Omega}\cdot\nabla + \Sigma_t^g$ whereas for neutron diffusion, $L^g = -\nabla\cdot D^g\nabla + \Sigma_r^g$, with Σ_t^g and Σ_r^g the total and removal macroscopic cross sections. In the following, we specialize the IQS derivation for the diffusion approximation without loss of generality. Obtaining the IQS formulation for neutron transport is similarly straightforward.

The flux factorization in IQS leads to a decomposition of the multigroup flux into the product of a time-dependent amplitude (p) and a space-/time-dependent multigroup shape (φ):

$$\phi^g(\boldsymbol{r},t) = p(t)\varphi^g(\boldsymbol{r},t) \,. \tag{2}$$

After reporting the above factorization in the balance equations, the shape diffusion equations result:

$$\frac{1}{v^g}\frac{\partial \varphi^g}{\partial t} = \sum_{g'=1}^{G}\left(H^{g'\to g} + P_p^{g'\to g}\right)\varphi^{g'} - \left(L^g + \boxed{\frac{1}{v^g}\frac{1}{p}\frac{dp}{dt}}\right)\varphi^g + \boxed{\frac{1}{p}}S_d^g \tag{3a}$$

see the red + above. I think you had a typo

$$\frac{dC_i}{dt} = \boxed{p}\sum_{g=1}^{G} P_{d,i}^g \varphi^g - \lambda_i C_i \ , \quad 1 \le i \le I \tag{3b}$$

Note that the time-dependent shape equations are similar to the time-dependent flux equations. One may introduce a new block diagonal operator $\tilde{L}^g = L^g + \frac{1}{v^g}\frac{1}{p}\frac{dp}{dt}$ where the reaction term (total/removal cross section) simply needs to be augmented by $\frac{1}{v^g}\frac{1}{p}\frac{dp}{dt}$. Note that the shape equations are now nonlinearly coupled (boxed terms) to the amplitude equations.

To obtain the amplitude equations, the multigroup shape equations are multiplied by a time-independent weighting function, typically the initial adjoint multigroup flux (ϕ^{*g}), and then integrated over the phase-space domain. For brevity, the inner product over space will be represented with parenthetical notation ($(\phi^{*g}, f^g) = \int_D \phi^{*g}(\boldsymbol{r}) f^g(\boldsymbol{r}) d^3r$). In order to impose uniqueness of the factorization, one requires that

$$K(t) = \sum_{g=1}^{G}\left(\phi^{*g}, \frac{1}{v^g}\varphi^g(t)\right) \tag{4}$$

to be constant (hence $K(t) = K_0$). After some manipulation, the point reactor kinetics equations (PRKE) for the amplitude solution are finally obtained:

$$\frac{dp}{dt} = \left[\frac{\rho - \bar{\beta}}{\Lambda}\right]p + \sum_{i=1}^{I}\bar{\lambda}_i\xi_i \tag{5a}$$

$$\frac{d\xi_i}{dt} = \frac{\bar{\beta}_i}{\Lambda}p - \bar{\lambda}_i\xi_i \quad 1 \le i \le I \tag{5b}$$

where the functional coefficients are calculated using the space-/time-dependent shape function as follows:

$$\frac{\rho - \bar{\beta}}{\Lambda} = \frac{\sum_{g=1}^{G}\left(\phi^{*g}, \sum_{g'}(H^{g'\to g} + P_p^{g'\to g} - L^{g'}\delta_{g'g})\varphi^{g'}\right)}{\sum_{g=1}^{G}\left(\phi^{*g}, \frac{1}{v^g}\varphi^g\right)} \tag{6a}$$

$$\frac{\bar{\beta}}{\Lambda} = \sum_{i=1}^{I}\frac{\bar{\beta}_i}{\Lambda} = \sum_{i=1}^{I}\frac{\sum_{g=1}^{G}(\phi^{*g}, P_{d,i}^g\varphi^g)}{\sum_{g=1}^{G}\left(\phi^{*g}, \frac{1}{v^g}\varphi^g\right)} \tag{6b}$$

$$\bar{\lambda}_i = \frac{\sum_{g=1}^{G}(\phi^{*g}, \chi_{d,i}^{g}\lambda_i C_i)}{\sum_{g=1}^{G}(\phi^{*g}, \chi_{d,i}^{g} C_i)} \tag{6c}$$

Solving for the shape in Eq. (3a) can become expensive, especially in two or three spatial dimensions, and even more so when using the transport equations in lieu of the diffusion equations. Using IQS, one expects the time dependence of the shape to be weaker than that of the flux itself, thus allowing for larger time step sizes in updating the shape. The PRKE equations form a small ODE system and can be solved using a much smaller time step size. In transients where the shape varies much less than the flux, IQS can be very computationally effective. The two-time scale solution process, a micro scale for the PRKE and a macro scale for the shape, is illustrated in Fig. 1.

technically there is an intermediate step for reactivity updates. a reviewer will criticize you easily for not me

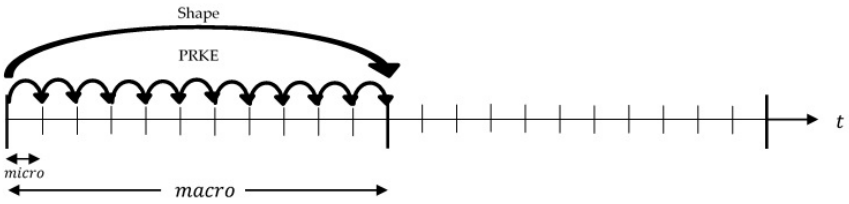


Figure 1: IQS method visualization

It is important to note that the PRKE parameters are evaluated at each macro step and interpolated for the PRKE evaluation. In order to preserve the error convergence rate of high order temporal discretization schemes for shape, higher order interpolation of the parameters will be required.

2.1. IQS Iterative Schemes

As noted in the previous section, shape-PRKE equations form a nonlinear system and must be solved in a iterative manner. Over each macro time step, one can use the latest end-time shape iterate to compute/interpolate the PRKE coefficents over the micro time step intervals. Sissaoui et al. from [**?**], Koclas et al. from [**?**], Devooght et al. from [**?**], and Monier from [**?**] all use iterative techniques for their IQS implementations. They all undergo a similar process:

Step 1: Compute the PRKE parameters at the end of the macro step using the last computed shape

Step 2: Linearly interpolate the computed PRKE parameters over the macro step

Step 3: Solve the PRKE on micro steps over the entire macro step

Step 4: Solve the shape equation on the macro step using the computed values of p and dp/dt.

Step 5: Check if the shape solution has converged:

- *No:* Repeat the same macro time step

– *Yes:* Move on to the next macro time step

The major difference between the methods of these authors is the convergence criteria used. Sissaoui and Koclas [**?** **?**] use fixed point iteration where the criteria is the simply the normalized difference between the last two computed shapes. Monier in [**?**] also employs fixed point iterations with the same criteria, except that the solution is rescaled by $\frac{\sum_{g=1}^{G}\left(\phi^{*g}, \frac{1}{v^g}\varphi^g(t_n)\right)}{\sum_{g=1}^{G}\left(\phi^{*g}, \frac{1}{v^g}\varphi^g(t_{n+1})\right)}$ after each iteration. can you give a small numerical insight into the scaling in addition to mentioning it; it would make th also, isn't the scaling simply K_n/K_{n+1}? finally, wasn't there another option where the shape is renormaliz Devooght in [**?**] uses a Newton-SOR iteration where the residual of the shape function evaluation is the convergence criteria and next iteration's solution is computed using Newton-Raphson method.

These techniques are by no means an exhaustive list of the possible iteration techniques for IQS. Dulla et al. in [**?**] provide an in depth analysis of the fixed point iteration technique most similar to Sissaoui and Koclas, involving convergence rates and solution results. However, no comprehensive analysis of iteration techniques exists, comparing both Newton and fixed-point convergence rates. The following describes each iteration convergence criterion investigated in this paper: you never show Newton results, do you?

- L^∞ norm of shape:
$$\frac{\max\left|\varphi_n^{(k+1)} - \varphi_n^{(k)}\right|}{\max\left|\varphi_n^{(k+1)}\right|} < \epsilon_\varphi$$

- L^2 norm of shape:
$$\frac{\left\|\varphi_n^{(k+1)} - \varphi_n^{(k)}\right\|_{L^2}}{\left\|\varphi_n^{(k+1)}\right\|_{L^2}} < \epsilon_\varphi$$

- Reactivity convergence:
$$(\rho/\Lambda)^{(k+1)} - (\rho/\Lambda)^{(k)} < \epsilon_\rho$$
why no time subscripts?

- Amplitude convergence:
$$p_n^{(k+1)} - p_n^{(k)} < \epsilon_p$$

- Uniqueness consistency:
$$(K_n^{(k+1)} - K_0)/K_0 < \epsilon_K$$
where k denotes the nonlinear iteration index within a given macro time step interval.

why not $n+1$ in the subscripts? that would make more sense since you use n+1 as the end-time afterwards

In order to ensure that uniqueness criteria is preserved, some authors (which ones?) explicitly scale the shape solution, either at each nonlinear iteration or upon exiting the nonlinear loop on the macro time time step. This re-scaling is simply

$$\varphi_n^g \leftarrow \varphi_n^{g,(\text{last})} \frac{K_0}{K_n^{(\text{last})}} \,. \tag{7}$$

so, how does this jive with the criterion on K?

2.2. IQS Predictor-Corrector Scheme

The Predictor-Corrector (P-C) version of IQS factorizes the flux and derives the PRKE the same way as the standard version, but the solution of the coupled system of equations is different. In the IQS P-C version, the flux equations (not the shape equations) are solved (represented by Eqs. (1a) and (1b)) in order to obtain a predicted flux solution. This predicted flux is then converted to a shape by rescaling it as follows:

$$\varphi_{n+1}^g = \underbrace{\phi_{n+1}^g}_{\text{predicted}} \frac{K_0}{K_{n+1}} \tag{8}$$

where the scaling factors are given by

$$K_{n+1} = \sum_{g=1}^{G} \left(\phi^{*g}, \frac{1}{v^g} \phi_{n+1}^g \right) \tag{9}$$

$$K_0 = \sum_{g=1}^{G} \left(\phi^{*g}, \frac{1}{v^g} \varphi_{n+1}^g \right) = \sum_{g=1}^{G} \left(\phi^{*g}, \frac{1}{v^g} \phi_0^g \right) \tag{10}$$

The PRKE parameters are then computed with this shape using Eqs. (6a)-(6c) and interpolated over the macro step, then the PRKE ODE system is solved on the micro time scale. With the newly computed amplitude, the shape is rescaled into a flux and the final corrected flux is given by:

$$\underbrace{\phi_{n+1}^g}_{\text{corrected}} = p_{n+1} \times \varphi_{n+1}^g \,. \tag{11}$$

The advantage to the predictor-corrector method is there is no iteration necessary for this method and, in turn, is much simpler and faster than the standard IQS. Ikeda et al. in [**?**] and Goluoglu et al. in [**?**] both use IQS P-C for complex, three-dimensional problems. Their results prove IQS P-C to be impressively effective, despite the de-coupling of the system. Dulla et al. in [**?**] also describes an in depth comparison of IQS P-C with traditional IQS.

2.3. Temperature Feedback Treatment

Two problems that are tested with IQS and IQS P-C involve adiabatic temperature feedback. The governing temperature equation is described by Eq. (12) with cross-section feedback described by Eq. (13).

$$\rho c_p \frac{\partial T(\boldsymbol{r},t)}{\partial t} = \kappa_f \sum_{g=1}^{G} \Sigma_f^g \phi^g(\boldsymbol{r},t) \tag{12}$$

$$\Sigma_a^{thermal}(\boldsymbol{r},t) = \Sigma_a^{thermal}(\boldsymbol{r},0)\left[1+\gamma\left(\sqrt{T}-\sqrt{T_0}\right)\right] \tag{13}$$

A typical implicit solver would simply use the interpolated flux at end of the temperature time step for the right hand side of the equation, a theta-method discretization. However, IQS has much more information about the profile of the flux along the time step because of the micro-step amplitude evaluation. Therefore, it is possible to solve for temperature using a semi-analytical approach, shown by Eq. (14).

$$T^{n+1} = T^n + \frac{\kappa_f}{\rho c_p}\left(a_2\varphi^{n+1} + a_1\varphi^n\right) \tag{14}$$

Where n corresponds to the beginning of the temperature step. a_1 and a_2 are integration coefficients defined by Eq. (15) and Eq. (16). Any interpolation of the amplitude along the micro steps is possible for the integration, this application uses piece-wise linear.

$$a_1 = \int_{t_n}^{t_{n+1}} \left(\frac{t_{n+1}-t'}{\Delta t}\right) p(t')dt' \tag{15}$$

$$a_2 = \int_{t_n}^{t_{n+1}} \left(\frac{t'-t_n}{\Delta t}\right) p(t')dt' \tag{16}$$

Temperature feedback affects both the shape equation and the reactivity of the PRKE; thus, it is an additional nonlinear component to the already coupled shape-amplitude equations. In foresight to the application of this component, temperature is much more time dependent than the shape, but less so than the amplitude. Therefore, the evaluation of temperature will have its own time scale. A possible solution process for a problem with temperature feedback will have time three time scales portrayed in Fig. 2. The first time scale is the shape solve, the second is the temperature evaluation as well as the computation of PRKE parameters, and the third is the PRKE scale. It is important to note that the number of time steps in each scale is arbitrary and meant only for visual purposes.

Iteration processes are needed for each time scale. The amplitude and temperature need to be iterated on the middle time scale until convergence on each temperature step. Then another iterative process needs to occur in the shape time scale on all three variables. Fig. 3 shows the programming diagram implement to execute this process. The time increment of $\Delta t/3$ for the temperature solve is arbitrary and is meant only to match Fig. 2 where three temperature updates have been used as illustration.

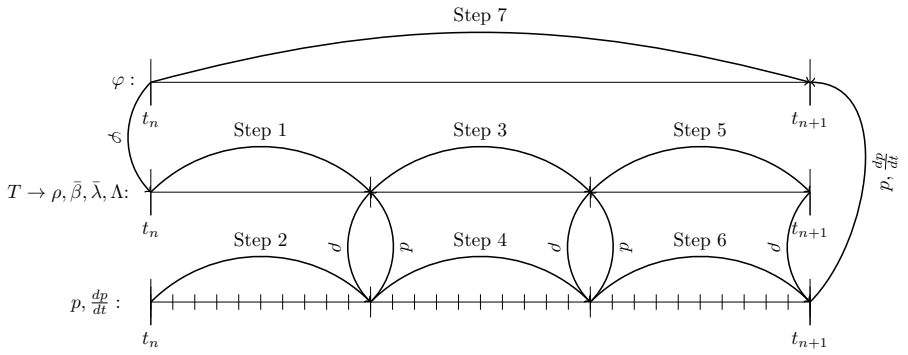


Figure 2: Time scales and process of IQS with temperature feedback

2.4. Delayed Neutron Precursor Treatment

This section presents the time-integration method used to solve coupled flux/shape and precursor equations, represented by Equations (1a)/(3a) and (1b)/(3b). First, we note that we could keep this system of two time-dependent equations and solve it as a coupled system. However, this is unnecessary and a memory expensive endeavor because the precursor equation is only an ODE and not a PDE. Instead, one may discretize in time the shape equation, which typically requires the knowledge of the precursor concentrations at the end of the time step. This precursor value is taken from the solution, numerical or analytical, of the precursors ODE.

A common precursor evaluation technique is using the theta method; applying this to Eq. (3b):

$$C^{n+1} = \frac{1-(1-\theta)\Delta t\lambda}{1+\theta\Delta t\lambda}C^n + \frac{(1-\theta)\Delta t\beta}{1+\theta\Delta t\lambda}S_f^n p^n + \frac{\theta\Delta t\beta}{1+\theta\Delta t\lambda}S_f^{n+1}p^{n+1} \tag{17}$$

Where S_f is the fission source equivalent for shape ($S_f^n = (\nu\Sigma_f)^n\varphi^n$). Another technique to evaluating the precursor equation is to use a exponential operator and integrate the time derivative analytically. Applying this operation to Eq. (3b) yields:

$$C^{n+1} = C^n e^{-\lambda(t_{n+1}-t_n)} + \int_{t_n}^{t_{n+1}} \beta(t')S_f(t')p(t')e^{-\lambda(t_{n+1}-t')}dt' \tag{18}$$

Again, this can be applied to implicit discretization by altering the definition of the fission source and eliminating p. Because S_f is not known continuously over the time step, the integration can be done using any scheme (Riemann, trapezoid, Simpson's, etc.). However, there is a very accurate representation of $p(t)$ over the macro step from the PRKE solve. In order to utilize this information, another possibility is to interpolate S_f linearly over the macro step. Such that:

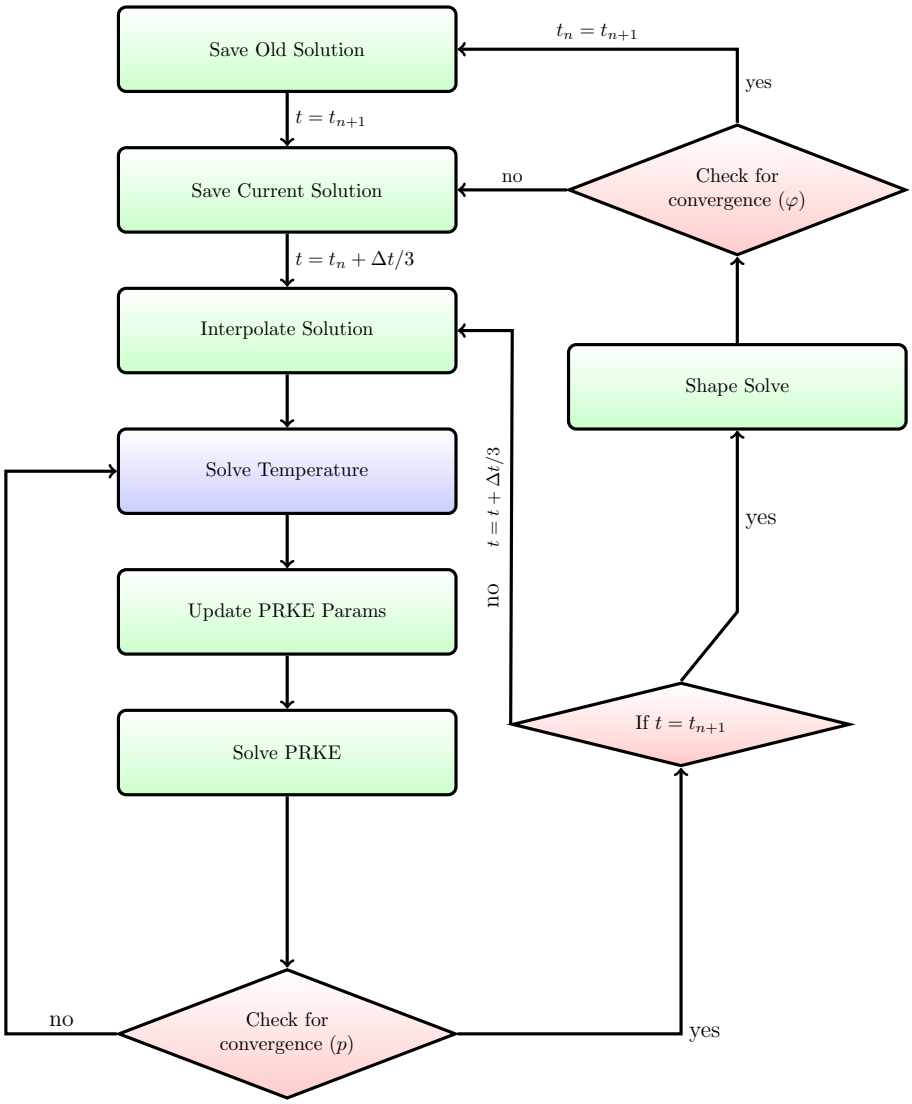


Figure 3: Visualization of fixed-point iteration and temperature update process for IQS

$$S_f(t) = \frac{t_{n+1} - t}{t_{n+1} - t_n} S_f^n + \frac{t - t_n}{t_{n+1} - t_n} S_f^{n+1} \quad t_n \leq t \leq t_{n+1} \tag{19}$$

Applying this to Eq. (18) yields:

$$C^{n+1} = C^n e^{-\lambda \Delta t} + \left(\hat{a}_2 S_f^{n+1} + \hat{a}_1 S_f^n \right) \beta \tag{20}$$

With integration coefficients defined as:

$$\hat{a}_1 = \int_{t_n}^{t_{n+1}} \frac{t_{n+1} - t'}{\Delta t} p(t') e^{-\lambda (t_{n+1} - t')} dt' \tag{21}$$

$$\hat{a}_2 = \int_{t_n}^{t_{n+1}} \frac{t' - t_n}{\Delta t} p(t') e^{-\lambda (t_{n+1} - t')} dt' \tag{22}$$

2.5. Step Doubling Time Adaptation

Further enhancements to IQS performance can be gained by using time adaptation (or time step control) in order to increase or reduce the macro time step size for the shape evaluation, depending on error estimates. A step-doubling technique is chosen as the time adaptation technique [**?**]. The step doubling technique involves estimating the local error for a certain time step by taking the difference between a solution with one full step ($\varphi^g_{\Delta t}$) and a solution with two half steps ($\varphi^g_{\Delta t/2}$). Note: φ is changed to ϕ for implicit discretization and IQS P-C.

The relative error is computed as follows:

$$e_n = \frac{\left\| \sum_{g=1}^G \varphi^g_{\Delta t/2} - \sum_{g=1}^G \varphi^g_{\Delta t} \right\|_{L^2}}{\max\left(\left\| \sum_{g=1}^G \varphi^g_{\Delta t/2} \right\|_{L^2}, \left\| \sum_{g=1}^G \varphi^g_{\Delta t} \right\|_{L^2} \right)} \tag{23}$$

If the error is smaller than the user-specified tolerance, e_{tol}, the time step is accepted. In addition, a new time step size is estimated as follows:

$$\Delta t_{new} = S \Delta t \left(\frac{e_{tol}}{e_n} \right)^{\frac{1}{1+q}} \tag{24}$$

Where q is the convergence order of the time integration scheme being used and $S \simeq 0.8$ is a safety factor. If the error is larger than the user-specified tolerance, the time step is rejected. A new time step size is estimated using Eq. (23) as well. This process can be visualized by Figs. 4 and 5. Where a step involves a full convergence of shape, amplitude, and any multiphysics on the respective time step.

To investigate IQS's performance with step-doubling time adaptation, the adaptation will be applied to implicit discretization method, traditional IQS, and IQS P-C. Each of these methods will be applied to several diffusion problems; the number of time steps taken and the resulting error will be used to compare the methods.

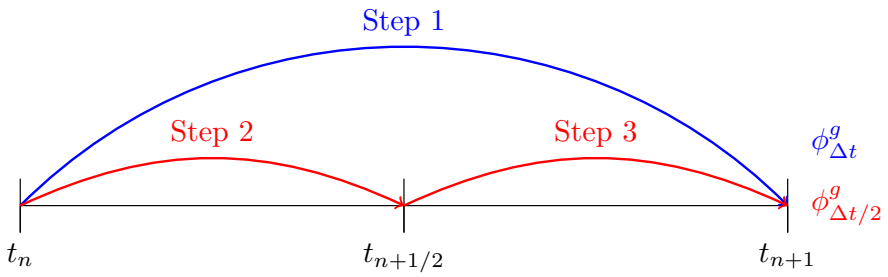


Figure 4: Visualization of step doubling process on time-line

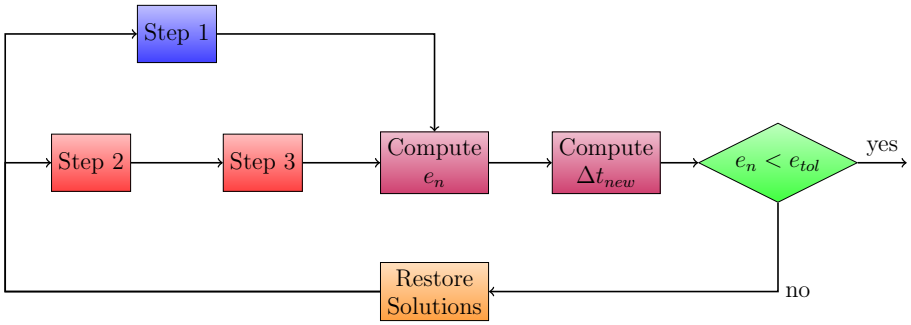


Figure 5: Visualization of step doubling process with coding logic

3. Kinetics Results

This section describes kinetics examples that IQS is tested with and analysis of its performance. The examples range in complexity and application. The first is a one-dimensional problem, designed for the prototype code in MATLAB. The next one is from the Argonne National Lab (ANL) Benchmark Problem Book (BPB), and is a common problem for testing codes and developing methods [**?**].

3.1. One-Dimensional Problem

This example is very simple and computes quickly; it entails a one dimensional, homogeneous 400 cm slab with a heterogenous perturbation in absorption cross section. Fig. 6 shows how the regions of the slab are divided and Table 1 shows the initial material properties. Regions 2, 3, and 4 have slope perturbations at different points in time, Table 2 shows the values of the absorption cross-section in each region at the times of interest. The values of Σ_a between these times of interest are linear interpolations between the given values.

Fig. 7 shows the resulting baseline flux and relative power profile of the one-dimensional problem. The baseline was computed using MATLAB's ode15s function which is a embedded Runge-Kutta time adaptive method for stiff problems, the error tolerance was set very tightly (10^{-12}). The flux distribution

1	1	1	1	2	3	1	1	1	1	1	1	1	1	4	4	1	1	1	1

Figure 6: 1-D slab region identification

Table 1: 1-D slab material properties and problem parameters

$D(cm)$	$\Sigma_a(cm^{-1})$	$\nu\Sigma_f(cm^{-1})$	$v(cm/s)$	β	$\lambda(s^{-1})$
1.0	1.1	1.1	1,000	0.006	0.1

Table 2: 1-D slab absorption cross-section at times of interest

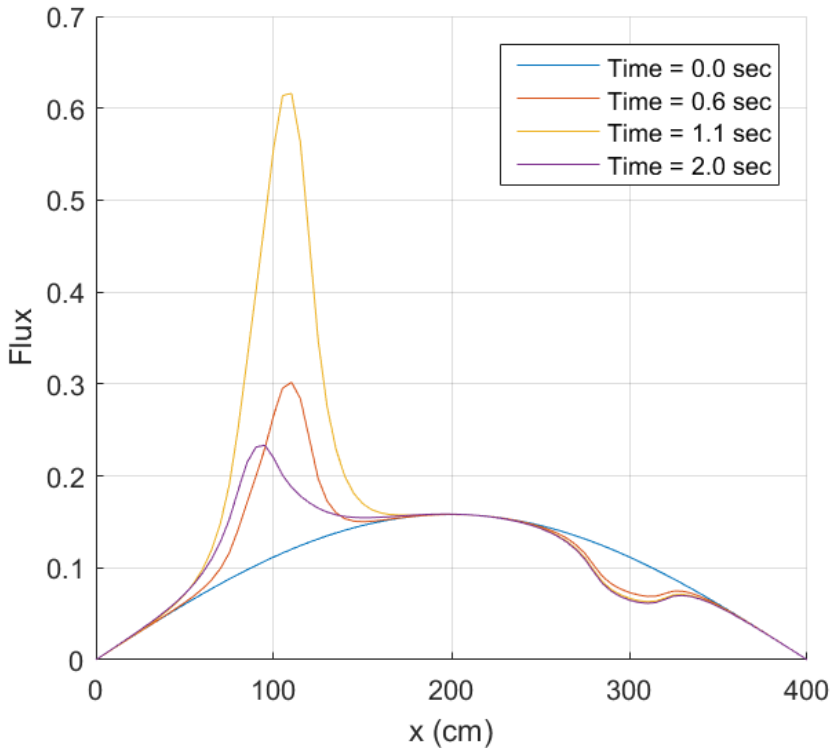
Region	Material Property	0.0 s	0.1 s	0.6 s	1.0 s	1.7 s
2	$\Sigma_a(cm^{-1})$	1.1	1.1	1.095	1.095	1.095
3	$\Sigma_a(cm^{-1})$	1.1	1.1	1.09	1.09	1.1
4	$\Sigma_a(cm^{-1})$	1.1	1.1	1.105	1.105	1.105

shows that the core is relatively large, thus different regions are weakly coupled. This baseline computation is used to compute the error of the other time discretization methods. Fig. 8a shows the shape profile at various times during the transient. It is apparent that the shape is very time-dependent, so it is expected that IQS has marginal accuracy gain for a given time step size.

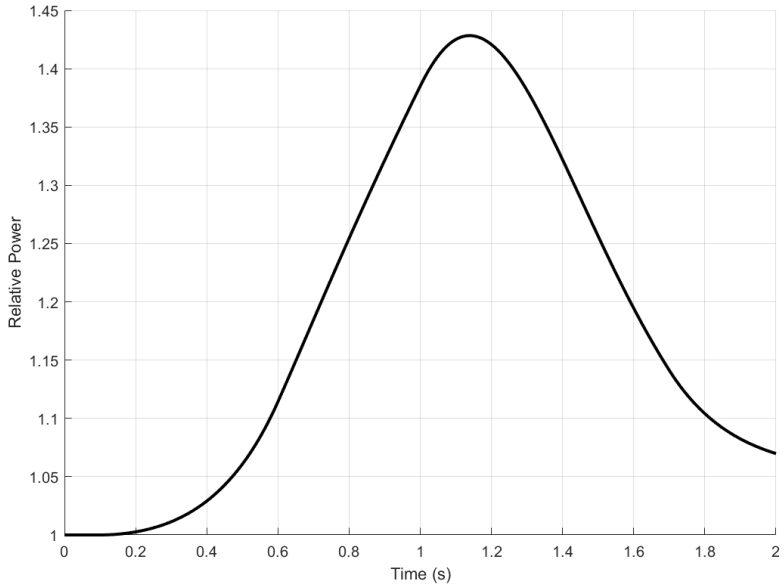
3.1.1. Iteration Analysis

IQS, as previously stated, is a system of nonlinear equations between shape and amplitude. These equations needed to be iterated to numerically converge to an accurate solution. Section 2.1 lists various iteration techniques for fixed-point and Newton schemes. Fig. 9 shows the number of fixed-point iterations required for a 10^{-11} tolerance over the transient. The criteria listed in the legend correspond to the list as such:

1. $L^\infty \rightarrow 1$.
2. $L^2 \rightarrow 2$.
3. Reactivity $\rightarrow 3$.
4. Amplitude $\rightarrow 4$.
5. K criteria $\rightarrow 5$.
6. All properties $\rightarrow$ 3.-5.

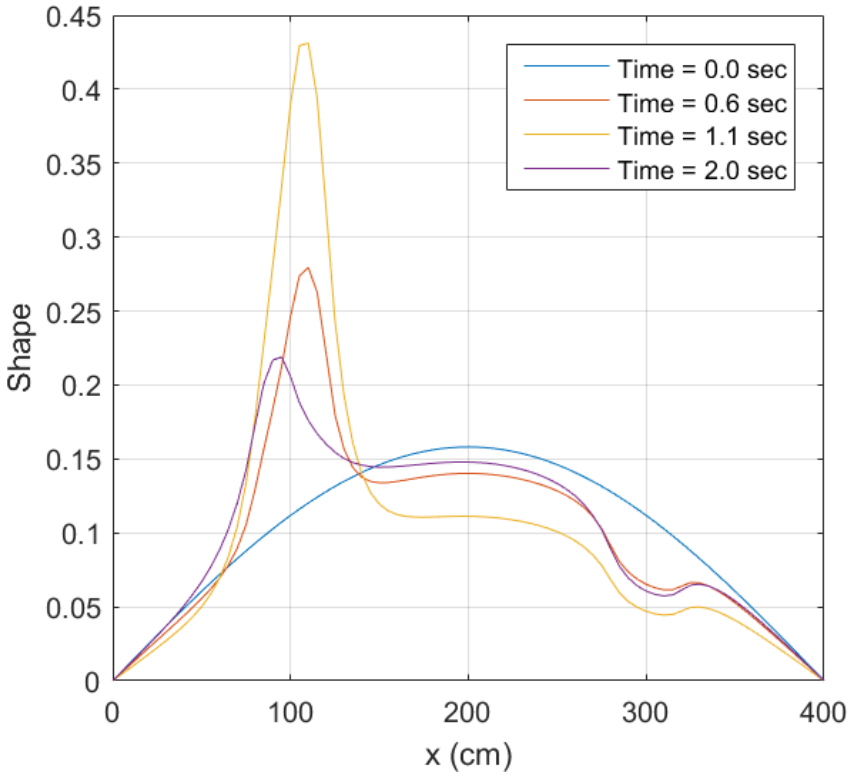


(a) Flux profile at various times

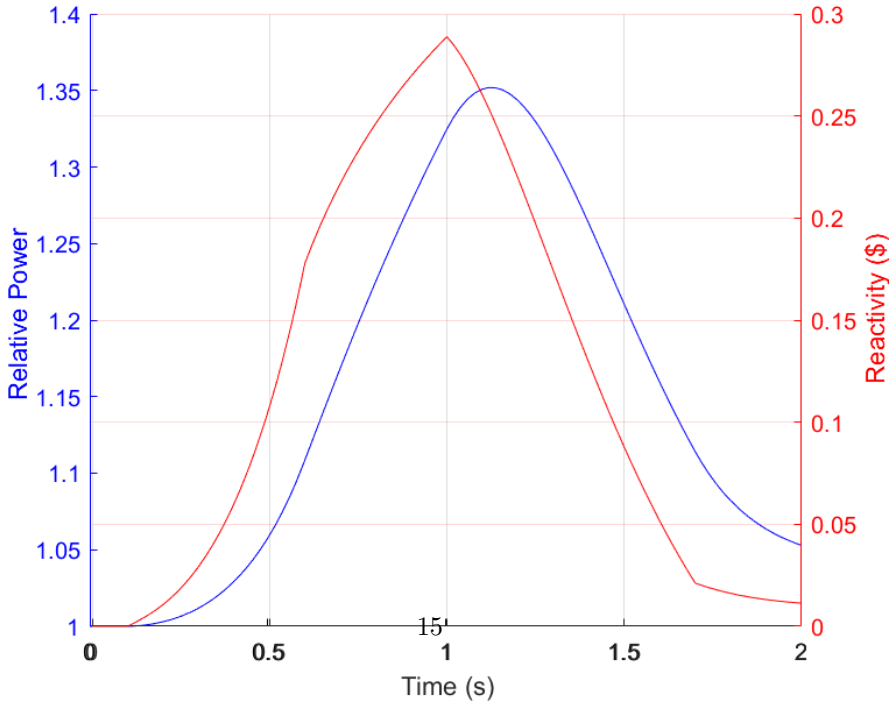


(b) Power profile over transient

Figure 7: Baseline flux and power distribution



(a) Shape profile at various times



(b) Power and reactivity profile over transient

Figure 8: IQS flux and power distribution

This plot shows that 1-4 have approximately the same convergence behavior, but the K criteria converges to a certain error. Fig. 10 shows the resulting error the K criteria converges to for different points of rescaling the shape. The rescaling is described by Eq. (7). Rescaling shape more frequently helps the error. However, rescaling every iteration is somewhat artificial because it does not consider changes spatially. Regardless, an error of 10^{-5} is quite large and it is expected that the magnitude is due to the explicit treatment of precursors (Eq. (17)). Switching to an analytical elimination (Eq. (20)) does not converge as well, but the error is much smaller, seen in Fig. 11.

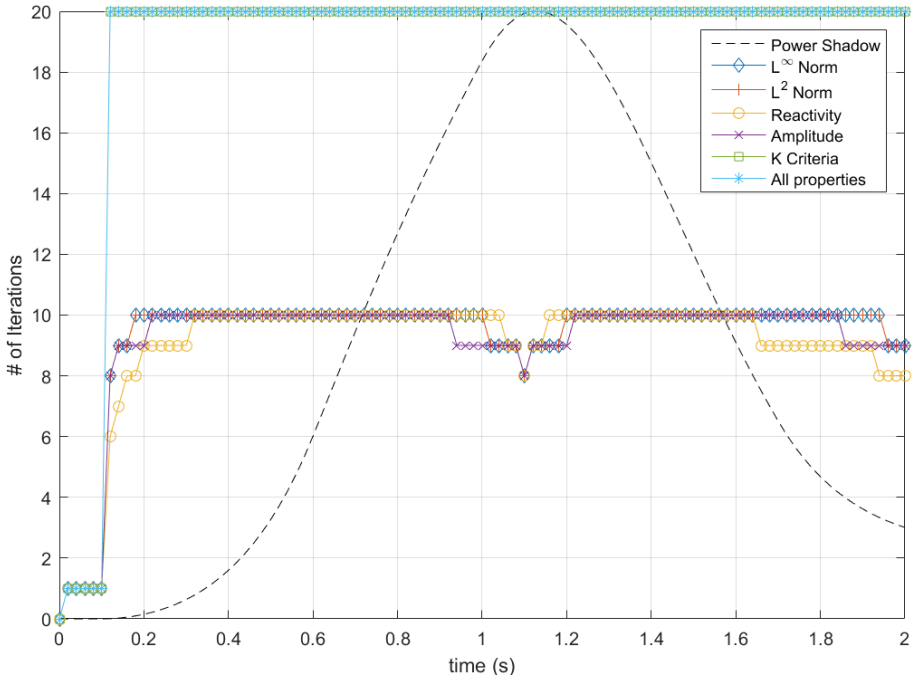


Figure 9: # of iterations for various convergence criteria, tolerance= 10^{-11}, max iterations= 20

3.1.2. One-Dimensional Problem Time Step Convergence Analysis

Time step convergence analysis involves evaluating a problem with various refinements in step size and comparing the resulting errors with the time step size. Plotting error versus Δt on a log-scale should produce a relatively strait line with a slope equal to the order of the time discretization method. In order to evaluate the performance and error convergence of IQS, the slab was simulated with varying time discretization methods and time step sizes. Fig. 12 shows these convergence plots of five different discretization methods for implicit discretization, IQS, and IQS P-C. These plots were generated from the results using the MATLAB prototype program. The plots show that IQS and IQS

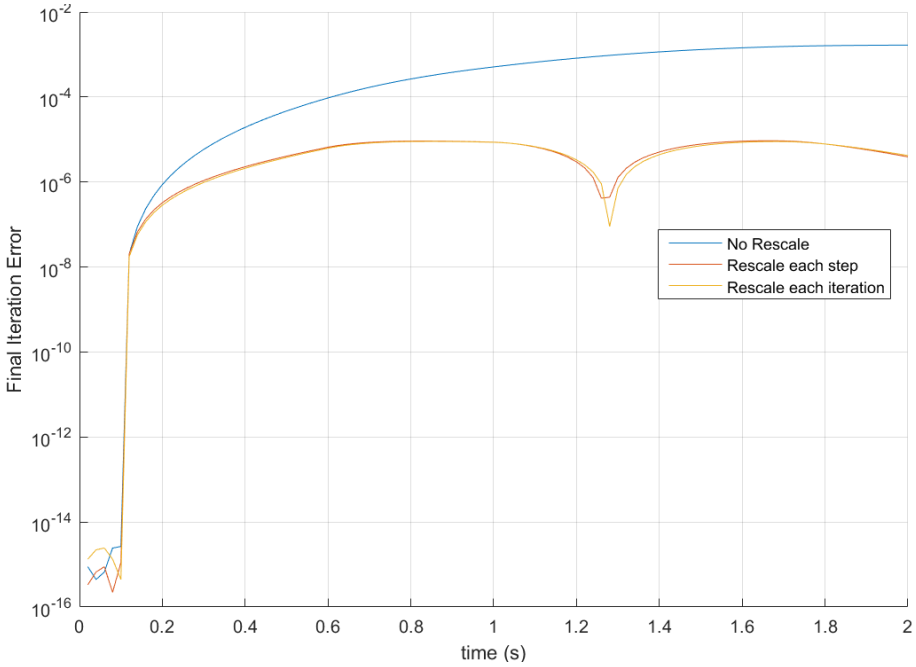


Figure 10: Final iteration error for K convergence criteria

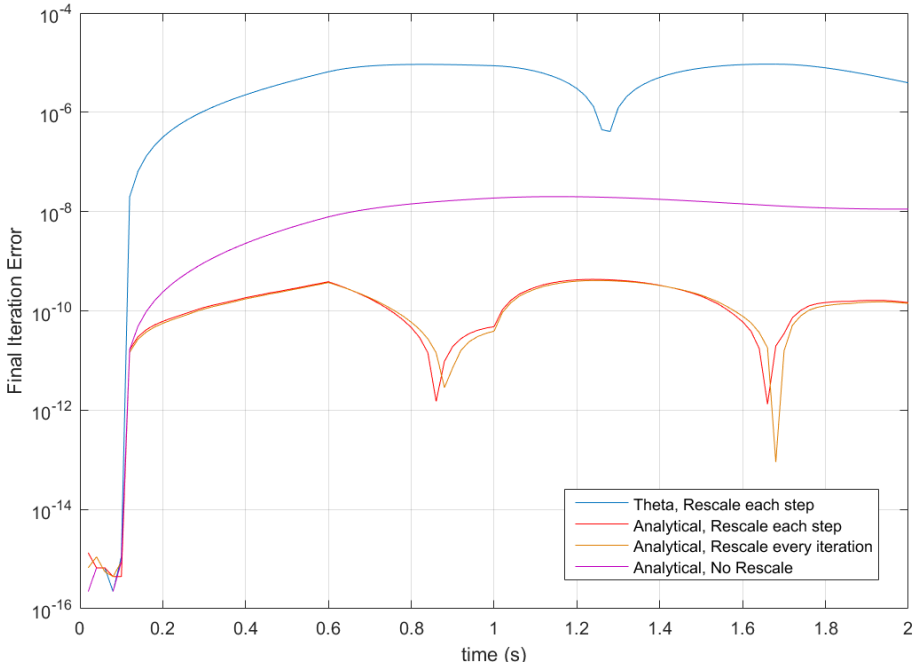


Figure 11: Final iteration error for K convergence criteria with analytical precursor elimination

P-C are convergent through fourth order BDF. Third order SDIRK did not show third order convergence, but, through extensive testing, SDIRK shows non-convergent behavior for too stiff of problems.

There are higher discretization order that can be tested, but most practical application do not go beyond second order. This paper shows an analysis of the first publicized application of IQS with higher than second order discretization, which exposed unforeseen properties of IQS. When using higher order techniques, the interpolation of PRKE parameters and shape for precursor integration become important to consider. Every other application that was investigated linearly interpolates parameters for the PRKE evaluation. Similarly, the shape used for the integration of the ODE for the precursors needs to have higher order interpolation to preserve high order error convergence. This interpolation was done using Lagragian and Hermite methods, both leading to successful convergence.

3.1.3. One-Dimensional Mini-Core Problem

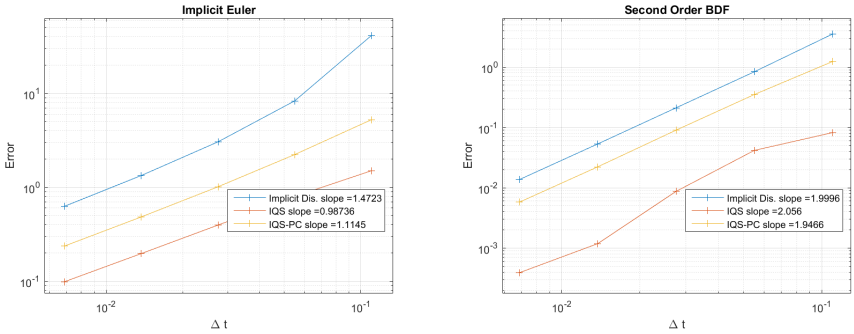
This problem is exactly the same as the previous one-dimensional problem, except the the core was reduced to 80 cm in length. The purpose of testing this problem is to determine if a more tightly coupled core will yield better performance for IQS. Fig. 13 shows the resulting baseline flux and relative power profile of the one-dimensional mini-core problem. This plot shows that the perturbed regions affect the domain more evidently. Fig. 14 shows the shape profile at various times during the transient. This plot shows that the shape is much less time-dependent than the previous large core. Fig. 15 shows that IQS performs significantly better with this example than the large core.

3.2. TWIGL Benchmark

This benchmark problem originates from the Argonne National Lab Benchmark Problem Book [**?**]. It is a 2D, 2-group reactor core model with no reflector region [**?**]. This example is meant to be of progressive complexity from the previous example. The transient of this reactor is very geometrically symmetrical with very little temporal shape change. Therefore, IQS is expected to perform significantly better than the implicit discretization method. Figs. 16 show the IQS solution as compared with the implicit discretization solution. This shows that IQS is consistent in more complex, higher dimensional problems. These plots also serve to illustrate that IQS has a much more accurate solution, even at a significantly larger time step than the implicit discretization.

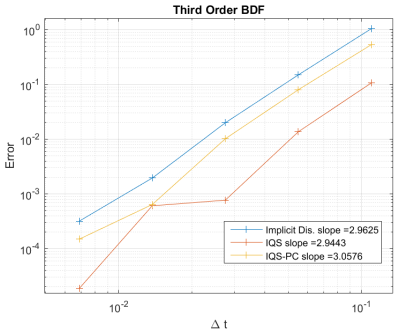
3.2.1. TWIGL Time Step Convergence Analysis

In order to demonstrate asymptotic convergence of IQS, implicit Euler (IE) and second order BDF (BDF2) were applied to the TWIGL simulation. Fig. 17 plots the error convergence of IQS and the implicit discretization methods. The curves show the impressive convergence of IQS for the highly transient TWIGL example. The slope indicated in the legend are the linear slope of curves on the log plot, these slopes should be similar to the order of the method (1 for IE and

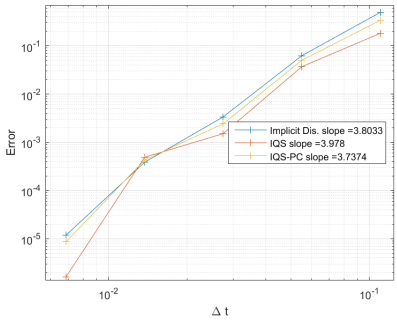


(a) Implicit Euler

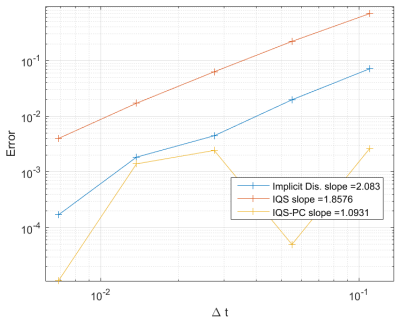
(b) BDF2



(c) BDF3

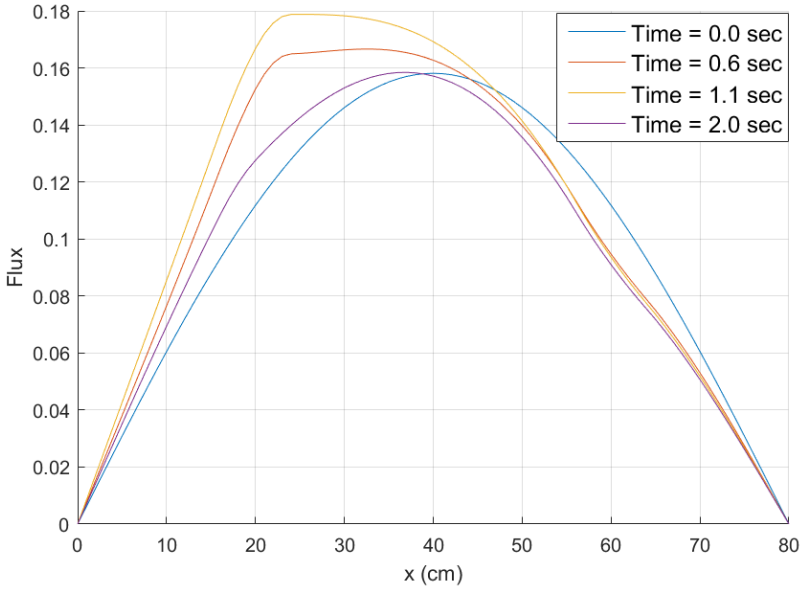


(d) BDF4

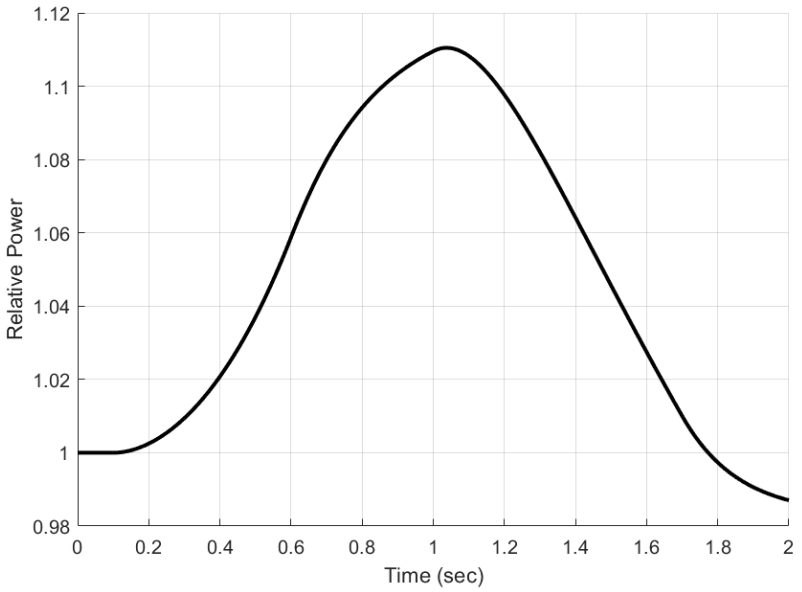


(e) SDIRK33

Figure 12: Error convergence plots of implicit discretization, IQS, and IQS P-C with various time discretization schemes



(a) Flux profile at various times



(b) Power profile over transient

Figure 13: Mini-core baseline flux and power distribution

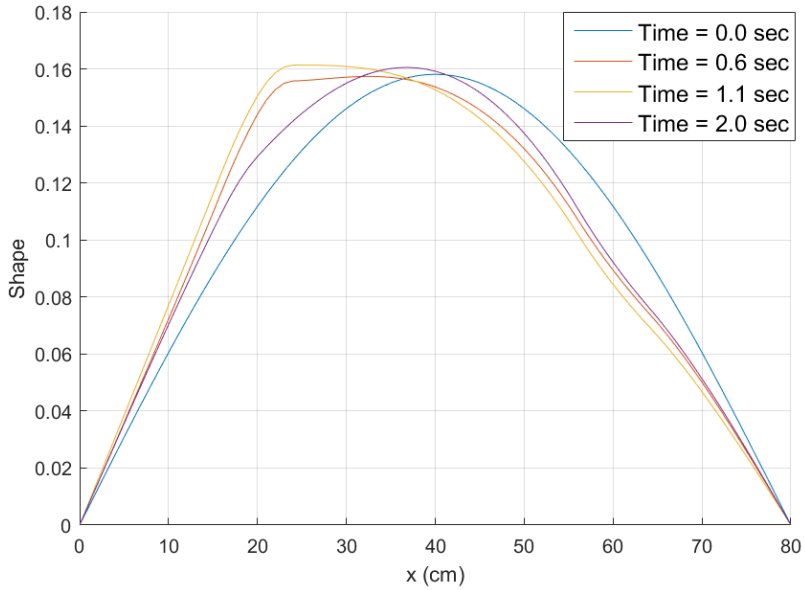


Figure 14: Shape profile at various times for one-dimensional mini-core

2 for BDF2). IQS shows a increased order because the PRKE is performing much of accuracy convergence and it is computed using SDIRK33, a third order method.

3.2.2. TWIGL Time Adaptation

Table 3 and Fig. 18 show the results for TWIGL with time adaptation. The results show that both IQS methods perform exceptionally well compared to implicit discretization. It also shows that traditional IQS performed better with large e_{tol}, while IQS P-C was better with smaller e_{tol}.

Table 3: TWIGL step doubling results

		Implicit Discretization			IQS			IQS P-C		
Test	e_{tol}	Error	Steps	Solves	Error	Steps	Solves	Error	Steps	Solves
1	0.05	0.00012677	9	29	0.03380433	4	20	0.00323100	4	9
2	0.01	3.5555e-05	11	35	0.00166991	5	40	0.00263068	5	12
3	0.005	4.0364e-05	11	31	0.00886584	5	40	0.00160486	6	21
4	0.001	0.00294822	33	122	0.02976305	5	36	1.7527e-05	10	35
5	0.0005	0.00099778	39	131	0.00143781	6	55	1.4185e-05	16	74
6	0.0001	0.00019510	78	236	0.00016175	8	65	6.2903e-06	19	78
7	5.0e-05	0.00018372	112	342	6.0328e-05	12	163	1.5247e-06	24	92
8	1.0e-05	8.0564e-05	263	794	7.7103e-05	379	5729	9.8321e-07	48	210

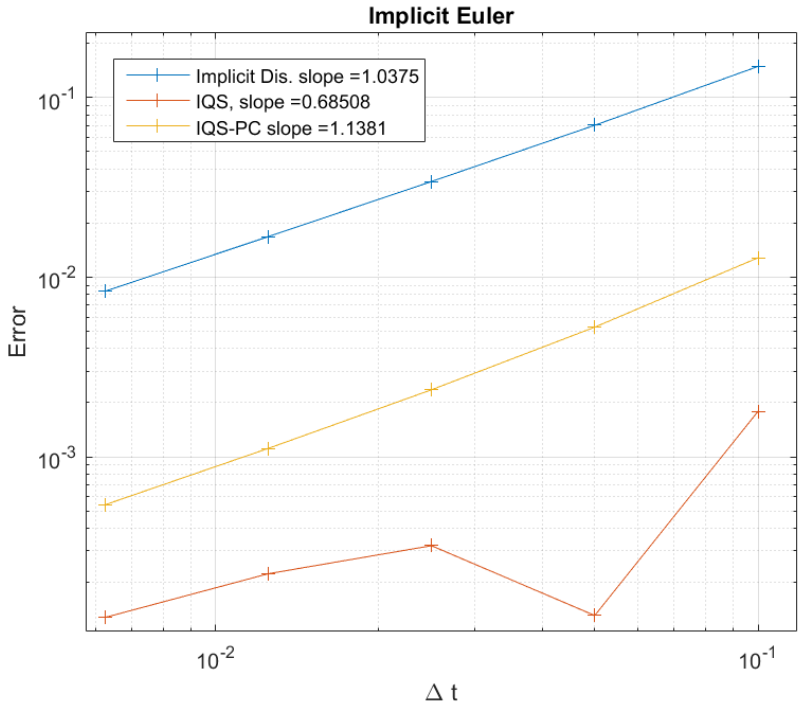
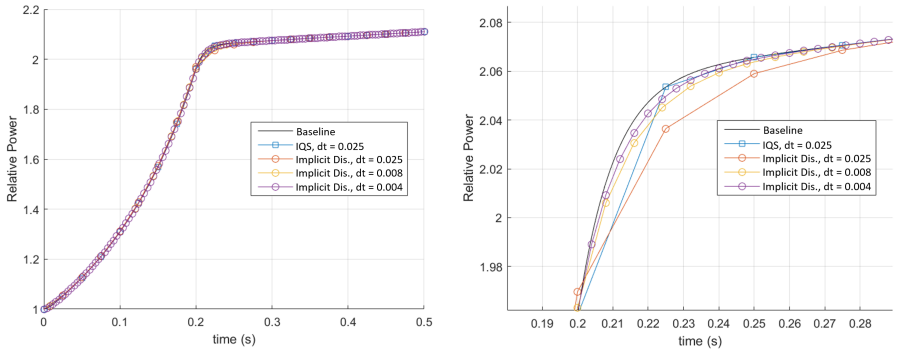


Figure 15: Time step convergence for one-dimensional mini-core with implicit Euler discretization



(a) Power profile for entire transient

(b) Power at cusp of profile

Figure 16: Power level comparison of TWIGL Benchmark

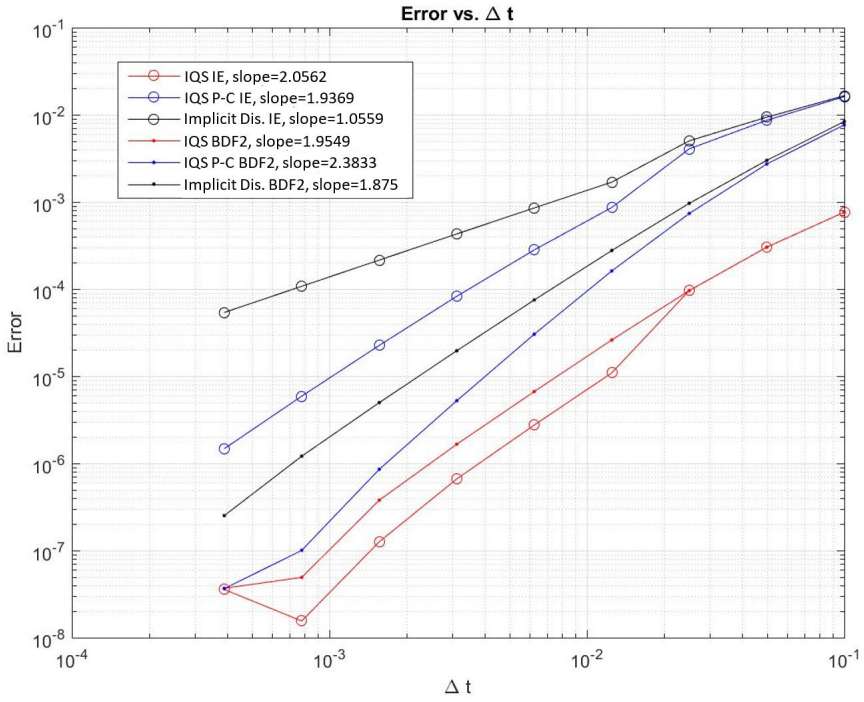


Figure 17: Error convergence comparison of TWIGL Benchmark

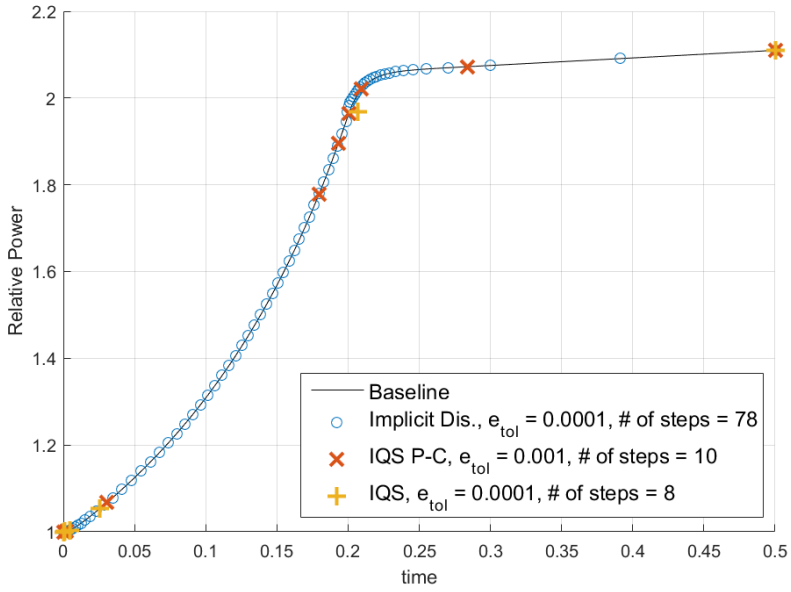


Figure 18: Power level comparison of TWIGL Benchmark with time adaptation

3.3. LMW Benchmark?

4. Dynamics Results

This section describes two dynamics examples, including the LRA benchmark and a TREAT experiment. These examples are of increased complexity from the previous kinetics examples. This section also analyzes IQS's performance with these, which is vital for verification of IQS in real-world problems.

4.1. LRA Benchmark

The LRA benchmark is a two-dimensional, two-group neutron diffusion problem with adiabatic heat-up and Doppler feedback in thermal reactor. It is a super prompt-critical transient. The execution of the benchmark was performed by the Rattlesnake/MOOSE framework at Idaho National Laboratory (INL) [**?**]. The spacial discretization was performed using continuous finite element method with first order Lagrangian basis functions. The mesh consisted of blocks 11×11 with five uniform refinements, totaling $165,165$ elements and $124,609$ nodes. Three different temporal techniques were applied: implicit discretization of the flux equation, IQS, and IQS-PC. Crank-Nicholson time discretization scheme was used for the diffusion evaluation of each technique. Third order Runge-Kutta discretization with step doubling adaptation was used for the PRKE evaluation. The performance of IQS and the temperature updates

were measured by its improvement in accuracy at peak power over the implicit discretization method.

Fig. 19 shows the baseline power and temperature transient profile for the LRA benchmark. The baseline results are compared to the results achieved by Sutton and Aviles in [**?**] and presented in Table 4. The relative difference in the magnitude of the peak power ($t \approx 1.44s$) from the baseline was used for error comparison.

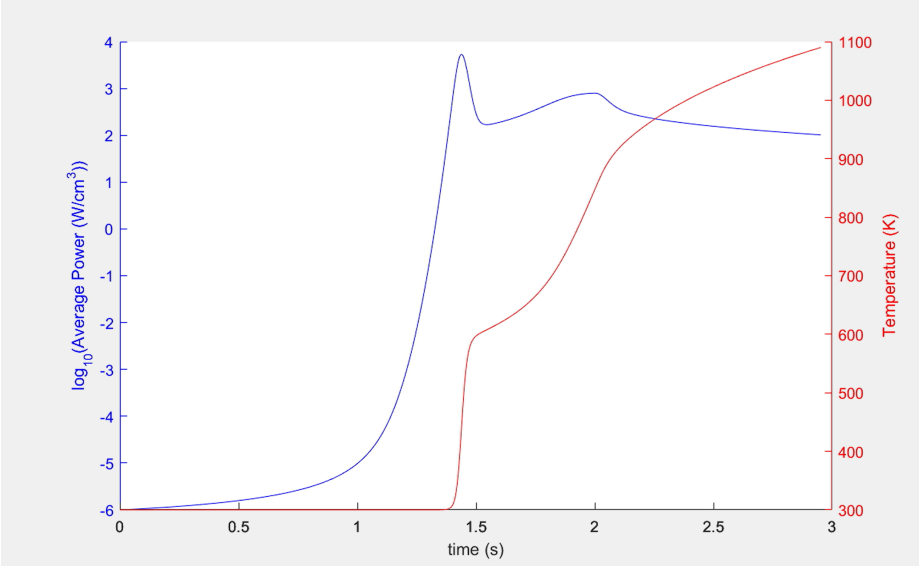


Figure 19: LRA baseline temperature and power profile

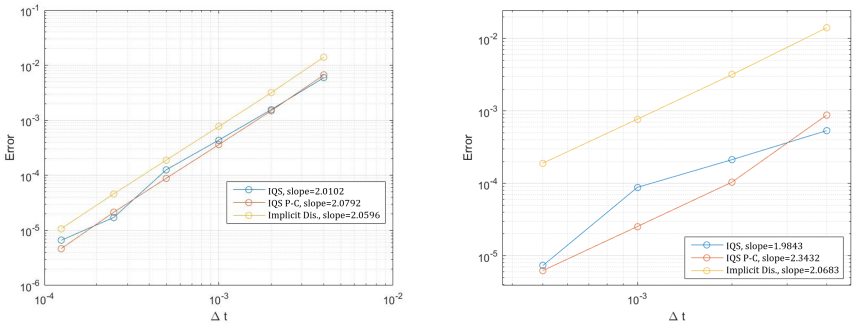
Table 4: LRA baseline verification

Calculation	Baseline	Sutton (Spandex 1936)
No. of Spatial Nodes	3872	1936
Eigenvalue	0.99637	0.99637
No. of Time Steps	6000	23,890
Time to Peak Power (s)	1.441	1.441
Peak Power (W/cm^3)	5456	5461

4.1.1. LRA Results

This section shows the time step error convergence of IQS for the LRA benchmark, as well as the effect of the intermediate temperature time scale. Fig. 20a is an error convergence plot comparing the three techniques where temperature

is evaluated only on the macro step (1 temperature update). Fig. 20b is an error convergence plot comparing the three techniques where temperature is evaluated 5 times within a macro step (5 temperature updates). Finally, Fig. 21 shows the effect of various temperature updates. The dashed lines correspond to implicit discretization at different flux step sizes, while the IQS macro step size is kept constant.



(a) Only one temperature update per macro step

(b) Five temperature updates per macro step

Figure 20: LRA error convergence plots

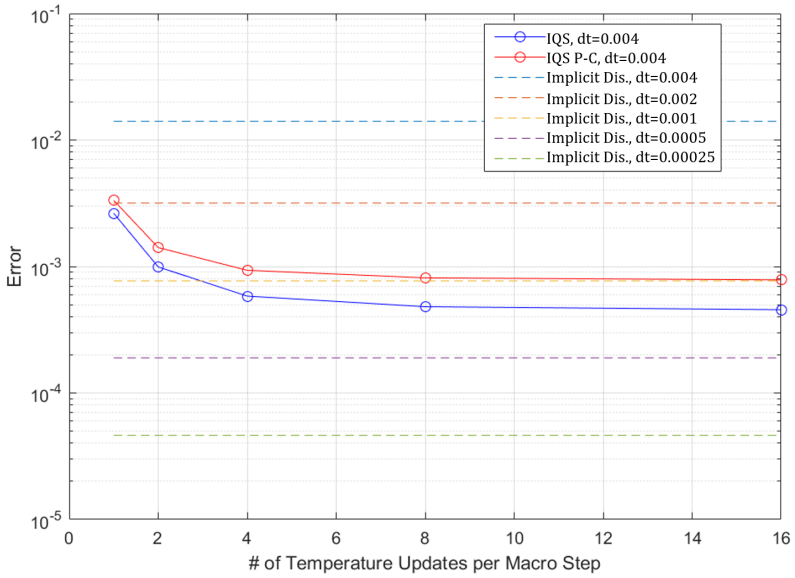


Figure 21: Error plot with various temperature updates per macro step

The convergence plots show that updating temperature and the PRKE pa-

rameters within a macro step has a significant effect on the performance of IQS. With only one update, IQS was only slightly better than implicit discretization, implicit discretization required about 150% more time steps than IQS for the same error. While 5 temperature updates showed a much more significant IQS performance, implicit discretization required about 400% more time steps than IQS for the same error. Fig. 21 shows that error has a convergent behavior for the number of temperature updates. This convergence makes sense because temperature can only be so accurate before the error in shape is dominating. Table 5 shows the run time results for the implicit discretization calculations. The number of GMRES linear iterations is included because it is proportional measure of the computational effort. Tables 6 and 7 present the IQS run-times with various numbers of temperature updates. These run-times are based on total alive time of the execution where the diffusion evaluation is distributed over 24 processors. These run-times show a marginal performance for IQS and impressive performance for IQS P-C. Some of the execution times were able to decrease from implicit discretization with the same number of macro steps because IQS is better equipped to resolve the nonlinearity between temperature and amplitude. Furthermore, there does seem to be an ideal number of temperature updates to optimize execution time: IQS only needs one and IQS P-C seems to be ideal at 4 updates. This discrepancy in the number of updates shows that a adaptive type implementation of the updates would be ideal, and could enforce a constant error over the transient. It is also important to compare the error of implicit discretization with IQS at one update and IQS P-C at 4 updates. IQS shows an error comparable to implicit discretization at $\Delta t = 0.002$, signifying an actual increase in runtime by -34.1%. IQS P-C shows an error less than implicit discretization at $\Delta t = 0.002$, signifying an actual increase in runtime by ¡-34.9%.

Table 5: Implicit discretization run time results

Run	Δt	Error	Runtime (hr)	Linear Iter.
1	4.0e-3	1.407e-2	4.11	7.13e4
2	2.0e-3	3.174e-3	6.01	9.49e4
3	1.0e-3	7.690e-4	10.38	1.45e5
4	5.0e-4	1.892e-4	21.91	2.08e5
5	2.5e-4	4.590e-5	25.23	3.16e5

4.1.2. LRA Time Adaptation

Fig. 22 shows the power profile of the LRA with time adaptation of implicit discretization and IQS P-C, and Table 8 compiles the results. These time adaptation results show the significant decrease in macro time steps required for IQS P-C. These profiles were obtaining with only one temperature update per macro step; so based on previous results, the IQS P-C performance would improve even more with more updates.

Table 6: IQS run time results with $\Delta t = 0.004$

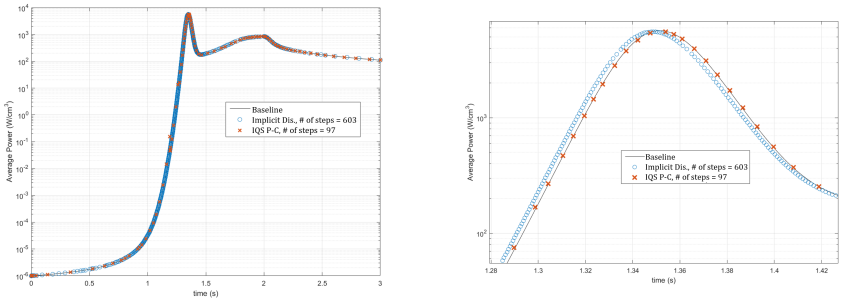
Run	Temperature Updates	Error	Runtime (hr)	% Increase in Runtime*
1	1	2.612e-3	3.96	-3.18%
2	2	9.893e-4	6.02	47.1%
3	4	5.796e-4	7.87	92.3%
4	8	4.772e-4	12.61	207.9%
5	16	4.516e-4	22.14	440.7%

* difference in runtime from $\Delta t = 0.004$ implicit discretization

Table 7: IQS PC run time results with $\Delta t = 0.004$

Run	Temperature Updates	Error	Runtime (hr)	% Increase in Runtime*
1	1	3.488e-3	2.91	-28.9%
2	2	1.349e-3	3.73	-9.00%
3	4	9.161e-4	3.97	-3.04%
4	8	8.052e-4	5.39	31.7%
5	16	7.905e-4	8.19	100%

* difference in runtime from $\Delta t = 0.004$ implicit discretization



(a) Full power profile

(b) Peak power profile at peak

Figure 22: LRA power profile with time adaptation of implicit discretization and IQS P-C

Table 8: LRA step doubling adaptation results with implicit discretization and IQS P-C

	Implicit Dis.			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