TEXAS A&M UNIVERSITY
Dwight Look College of Engineering
Department of Nuclear Engineering

Research Memo

From: Zachary M. Prince

To: Distribution
Date: October 8, 2015

Subject: Precursor Evaluation

1. Precursor Solve for Shape Evaluation

This document shows two different methods to solve precursor equations or IQS shape evaluation. To begin, below is the shape equation coupled with its precursors; for simplicity, the problem is single group with a single precursor group.

$$\frac{1}{v}\frac{\partial\varphi}{\partial t} = \nu\Sigma_f(1-\beta)\varphi - \left(-\vec{\nabla}\cdot D\vec{\nabla} + \Sigma_a + \frac{1}{v}\frac{1}{p}\frac{dp}{dt}\right)\varphi + \frac{1}{p}\lambda C \tag{1}$$

$$\frac{dC}{dt} = \beta\nu\Sigma_f\varphi p - \lambda C \tag{2}$$

It is possible to keep these equations separate and discretize them together; however, this is unnecessary and memory expensive endeavor. So solving the precursor equation using some discretization or integration scheme to solve the precursors with respect to the shape first, then insert that into the shape equation, solving only one equation at a time. This document will discuss two techniques to solving the precursor equation. First is a time discretization method that is currently being implemented in YAK. The second is a analytical integration of the precursors, this method has proven to be more beneficial for IQS.

2. Time Discretization using Theta Method

The simplest way to evaluate the precursor equation is to perform a implicit/explicit finite difference approximation known as the theta method. The method entails discretizing the the precursor in time and including a weight (θ) to the implicit and explicit components. Most generally, if there is a function u defined as:

$$\frac{du}{dt} = f(u,t) \tag{3}$$

Then the discretization would look like:

$$\frac{u^{n+1}-u^n}{\Delta t} = (1-\theta)f(u^n,t) + \theta f(u^{n+1},t) \tag{4}$$

Applying this to the precursor equation:

$$\frac{C^{n+1}-C^n}{\Delta t} = (1-\theta)\beta(\nu\Sigma_f)^n\varphi^n p^n - (1-\theta)\lambda C^n + \theta\beta(\nu\Sigma_f)^{n+1}\varphi^{n+1}p^{n+1} - \theta\lambda C^{n+1} \tag{5}$$

Rearranging to solve for the precursor at the new time step:

$$C^{n+1} = \frac{1-(1-\theta)\Delta t\lambda}{1+\theta\Delta t\lambda}C^n + \frac{(1-\theta)\Delta t\beta(\nu\Sigma_f)^n}{1+\theta\Delta t\lambda}\varphi^n p^n + \frac{(1-\theta)\Delta t\beta(\nu\Sigma_f)^{n+1}}{1+\theta\Delta t\lambda}\varphi^{n+1}p^{n+1} \tag{6}$$

This equation can then be plugged into the shape equation and solving for shape alone. θ is a somewhat arbitrary constant $0 \le \theta \le 1$. However, the purely explicit method shows when $\theta = 0$, implicit when $\theta = 1$, and Crank-Nicholson when $\theta = 1/2$. Due to instability, the explicit method is not a preferable option. YAK currently implements both implicit and Crank-Nicholson as options for precursor evaluation.

3. **Analytical Integration**

Through prototyping, it has been found that neither implicit nor Crank-Nicholson time discretization of precursors are preferable methods for solving the shape equation in IQS. It has been found that these discretizations result in a lack of convergence of the shape over the IQS iteration process. In order to remedy the error, a analytical representation of the precursors was implemented in the prototype and the shape solution was able to converge. The following section shows how this method was implemented in the prototype and the desired implementation in YAK.

Using an exponential operator, the precursor equation can be analytically solved for:

$$Ce^{\lambda t} = \int \beta\nu\Sigma_f\varphi p e^{\lambda t'}dt' \tag{7}$$

The right hand side can be integrated over a time step in order to acquire the precursor at the next time step:

$$C^{n+1}e^{\lambda t_{n+1}} = \beta\int_{t_n}^{t_{n+1}} \nu\Sigma_f(t')\varphi(t')p(t')e^{\lambda t'}dt' + \lambda C^n e^{\lambda t_n} \tag{8}$$

Dividing by $e^{\lambda t_{n+1}}$:

$$C^{n+1} = \beta\int_{t_n}^{t_{n+1}} \nu\Sigma_f(t')\varphi(t')p(t')e^{-\lambda(t_{n+1}-t')}dt' + \lambda C^n e^{-\lambda\Delta t} \tag{9}$$

Because $\nu\Sigma_f$ and φ being integrated are not known continuously over the time step, they can be interpolated linearly over the macro step. Such that:

$$f(t) \cong \frac{t - t_n}{t_{n+1} - t_n}(f^{n+1} - f^n) \quad t_n \leq t \leq t_{n+1} \tag{10}$$

So:

$$C^{n+1} = \beta \left(a_3(\nu\Sigma_f)^{n+1}\varphi^{n+1} + a_2(\nu\Sigma_f)^{n+1}\varphi^n + a_2(\nu\Sigma_f)^n\varphi^{n+1} + a_1(\nu\Sigma_f)^n\varphi^n \right) + \lambda C^n e^{-\lambda\Delta t} \tag{11}$$

Where the integration coefficients are defined as:

$$a_1 = \int_{t_n}^{t_{n+1}} \left(\frac{t_{n+1} - t'}{\Delta t} \right)^2 p(t') e^{-\lambda(t_{n+1} - t')} dt' \tag{12}$$

$$a_2 = \int_{t_n}^{t_{n+1}} \frac{(t' - t_n)(t_{n+1} - t')}{(\Delta t)^2} p(t') e^{-\lambda(t_{n+1} - t')} dt' \tag{13}$$

$$a_3 = \int_{t_n}^{t_{n+1}} \left(\frac{t' - t_n}{\Delta t} \right)^2 p(t') e^{-\lambda(t_{n+1} - t')} dt' \tag{14}$$

The amplitude (p) is included in the integration coefficient because it has been highly accurately calculated in the micro step scheme, so a piecewise interpolation between those points can be done to maximize accuracy. However, this term in the integration makes it impossible to do analytically. The prototype uses Matlab software to interpolate the amplitude between micro steps and a quadrature integration for the coefficients. So the challenge for YAK is to replicate this procedure: passing the amplitude vector to the DNP auxkernel, interpolating it, and integrating the coefficients. Applying these coefficients to the ”fission source” term seems trivial after the procedure is completed.

ZMP:zmp

Distribution:
Yaqi Wang, INL, *yaqi.wang@inl.gov*,
Jean Ragusa, TAMU, *jean.ragusa@tamu.edu*

Cy:
File