LA-UR- 11-01534

Approved for public release; distribution is unlimited.

Title:	Nonlinear Acceleration of Transport Criticality Problems
Author(s):	C.K. Newman -T-3 D. Knoll- T-3 R. Park- Idaho National Lab
Intended for:	International Conference on Mathematics and Computational Methods May 8-12, 2011 Rio de Jan

Los Alamos
NATIONAL LABORATORY
EST. 1943

Form 836 (7/06)

International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2011)
Rio de Janeiro, RJ, Brazil, May 8-12, 2011, on CD-ROM, Latin American Section (LAS) / American Nuclear Society (ANS)
ISBN 978-85-63688-00-2

NONLINEAR ACCELERATION OF TRANSPORT CRITICALITY PROBLEMS

H. Park
Multiphysics Methods Group
Idaho National Laboratory
2525 North Fremont Ave.
Idaho Falls, ID 83415
Ryosuke.Park@inl.gov

D.A. Knoll and C.K. Newman
Theoretical Division, MS B216
Los Alamos National Laboratory
Los Alamos, NM 87545
nol@lanl.gov, cnewman@lanl.gov

ABSTRACT

We present a nonlinear acceleration algorithm for the transport criticality problem. The algorithm combines the well-known nonlinear diffusion acceleration (NDA) with a recently developed, Newton-based, nonlinear criticality acceleration (NCA) algorithm. The algorithm first employs the NDA to reduce the system to scalar flux, then the NCA is applied to the resulting drift-diffusion system. We apply a nonlinear elimination technique to eliminate the eigenvalue from the Jacobian matrix. Numerical results show that the algorithm reduces the CPU time a factor of 400 in a very diffusive system, and a factor of 5 in a non-diffusive system.

Key Words: nonlinear acceleration, Jacobian-free Newton-Krylov, nonlinear elimination, criticality problems

1. INTRODUCTION

The dominant eigenvalue (k_{eff})-eigenvector pair of a criticality problem is often computed by traditional iterative methods such as the (inverse) power method. However, the power method suffers from a slow convergence rate when the dominance ratio, $\rho = \frac{k_2}{k_1}$ with $k_1 = k_{\text{eff}}$, of the problem is close to one. There have been several algorithm developments for accelerating the convergence rate. Most commonly applied methods are Wielandt shift [1] and Chebyshev acceleration [2]. The Wielandt shift method treats a fraction of the fission source implicitly, which effectively reduces the dominance ratio to $\tilde{\rho} = \frac{k_2 - k_s}{k_{\text{eff}} - k_s}$. On the other hand, the Chebyshev acceleration method constructs an optimal polynomial using previous iterates to extrapolate the eigenvalue-eigenvector pair.

There are several new efforts found in the literature for accelerating the convergence of criticality problems. One approach is to apply a Krylov based iterative method, such as the implicitly

restarted Arnoldi method (IRAM) [3]. The other is to cast the eigenvalue problem as a nonlinear-PDE constraint problem [4–6].

We present a new efficient approach to solve the criticality transport problem. The algorithm is based on the combination of two nonlinear acceleration schemes; nonlinear diffusion acceleration (NDA) [7, 8] and nonlinear criticality acceleration (NCA) [5, 6]. We compare the methodology with two other transport criticality algorithms.

2. NUMERICAL METHODS

2.1 Background: Transport Criticality Problems and Power Methods

For simplicity, we consider a 1D-slab, 1-group form of the neutron transport equation. Consider the transport equation,

$$\mu \frac{\partial \psi(\mu, x)}{\partial x} + \Sigma_t \psi(\mu, x) = \frac{1}{2} \left(\Sigma_s \phi(x) + \frac{\nu \Sigma_f}{k_{\text{eff}}} \phi \right), \tag{1}$$

as the governing equation. We refer to this as the high-order (HO) problem. We can write Eq. (1) using operator notation,

$$\mathcal{L}\psi = \left(\mathcal{S} + \frac{1}{k_{\text{eff}}} \mathcal{F} \right) \phi, \tag{2}$$

where $\mathcal{L}, \mathcal{S}$ and $\mathcal{F}$ denote transport, scattering and fission operators. Often Eq. (2) is solved with the power iteration. In this study, we consider discrete ordinates (S_N) as the angular discretization. In the case of S_N, (scattering) source iterations and power iterations can be combined. This produces the following simple inner-outer iterative method:

Algorithm 1: Standard power iterations: PI

initialize k_{eff}^0, ϕ^0 ;
for $m = 1, M$ **do**
  set $\Phi^0 = \phi^{m-1}$;
  for $n = 1, N$ **do**
    $\psi^n = \mathcal{L}^{-1} \left(\mathcal{S}\Phi^{n-1} + \frac{1}{k_{\text{eff}}^{m-1}} \mathcal{F}\Phi^0 \right)$;
    $\Phi^n = \int d\mu \; \psi^n$;
  end
  $\phi^m = \Phi^N$;
  $k_{\text{eff}}^m = k_{\text{eff}}^{m-1} \frac{\int dV \nu \Sigma_f \phi^m}{\int dV \nu \Sigma_f \phi^{m-1}}$;
  if $\frac{k_{\text{eff}}^m - k_{\text{eff}}^{m-1}}{k_{\text{eff}}^m} < \eta$ **then**
    break;
  end
end

In Algorithm 1, the outer $m-$loop is the power iteration loop, while inner $n-$loop is the source iteration loop. The convergence rate of this iterative approach (Algorithm 1) is determined by

two factors: the scattering ratio and dominance ratio. The scattering source term, $\Sigma_s\phi$, may converge slowly when the scattering ratio of the problem is large. Secondly, the problem with dominance ratio close to unity is slow to converge with the power method. For realistic reactor physics problems, both the scattering ratio and dominance ratio can be close to one, and the method suffers from very slow convergence. Our goal is to reduce both scattering source and fission source iterations via nonlinear acceleration methods.

2.2 Accelerating Scattering Source via NDA

To remedy slow convergence in the scattering source, we apply NDA, or commonly known as Coarse-Mesh Finite Difference (CMFD) to the HO problem [7, 8]. The NDA method integrates Eq.(1) over the angle, μ, to arrive at the following low-order equation:

$$\frac{dJ}{dx} + (\Sigma_t - \Sigma_s)\,\phi = \frac{\nu\Sigma_f}{k_{\text{eff}}}\phi. \tag{3}$$

We define the current, J, as,

$$J \equiv \int_{-1}^{+1} d\mu\ \mu\psi(x,\mu). \tag{4}$$

To define a low order (LO) problem, we use the following current closure,

$$J = -\frac{1}{3\Sigma_t}\frac{d\phi}{dx} + \hat{D}\phi. \tag{5}$$

We then obtain the following LO equation by substitution of Eq. (5) into Eq. (3),

$$\frac{d}{dx}\left[-\frac{1}{3\Sigma_t}\frac{d\phi}{dx} + \hat{D}\phi\right] + (\Sigma_t - \Sigma_s)\,\phi = \frac{\nu\Sigma_f}{k_{\text{eff}}}\phi, \tag{6}$$

or in the operator notation,

$$\mathcal{D}\phi = \frac{1}{k_{\text{eff}}}\mathcal{F}\phi, \tag{7}$$

where $\mathcal{D} \equiv \int d\mu(\mathcal{L} - \mathcal{S})$ denotes the drift-diffusion operator. With a proper definition of $\hat{D}$ [7], the LO equation (7) has exactly the same scalar flux (hence k_{eff}) as Eq. (2). Then, we can use the LO equation for

- implicit treatment of scattering operator $\mathcal{S}$ (Algorithm 2) and
- reducing the dimensionality (space-angle-energy to space-energy) of the power iterations (Algorithm 3).

Algorithm 2: CMFD accelerated power iterations: CMFD-PI(0)

initialize k_{eff}^0, ϕ^0 ;
for $m = 1, M$ **do**
  $\psi^m = \mathcal{L}^{-1}\left(\mathcal{S} + \frac{1}{k_{\text{eff}}^{m-1}}\mathcal{F}\right)\phi^{m-1}$;
  $\hat{D}^m = \frac{\int d\mu\mu\psi^m + \frac{1}{3\Sigma_t}\frac{d\phi^m}{dx}}{\phi^m}$;
  $\phi^m = \mathcal{D}^{-1}\left(\frac{1}{k_{\text{eff}}^{m-1}}\mathcal{F}\phi^{m-1}\right)$, with $\mathcal{D} = \frac{d}{dx}\left[-\frac{1}{3\Sigma_t}\frac{d}{dx} + \hat{D}^m\right] + (\Sigma_t - \Sigma_s)$;
  $k_{\text{eff}}^m = k_{\text{eff}}^{m-1}\frac{\int dV\nu\Sigma_f\Phi^m}{\int dV\nu\Sigma_f\Phi^{m-1}}$;
  if $\frac{k_{\text{eff}}^m - k_{\text{eff}}^{m-1}}{k_{\text{eff}}^{m-1}} < \eta_1$ **then**
    break;
  end
end

Algorithm 3: CMFD accelerated power iterations: CMFD-PI(M)

initialize k_{eff}^0, ϕ^0 ;
for $n = 1, N$ **do**
  $\psi^n = \mathcal{L}^{-1}\left(\mathcal{S} + \frac{1}{k_{\text{eff}}^{n-1}}\mathcal{F}\right)\phi^{n-1}$;
  $\hat{D}^n = \frac{\int d\mu\mu\psi^n + \frac{1}{3\Sigma_t}\frac{d\phi^n}{dx}}{\phi^n}$;
  for $m = 1, M$ **do**
    $\Phi^m = \mathcal{D}^{-1}\left(\frac{1}{k_{\text{eff}}}\mathcal{F}\Phi^{m-1}\right)$, with $\mathcal{D} = \frac{d}{dx}\left[-\frac{1}{3\Sigma_t}\frac{d}{dx} + \hat{D}^n\right] + (\Sigma_t - \Sigma_s)$;
    $K_{\text{eff}}^m = K_{\text{eff}}^{m-1}\frac{\int dV\nu\Sigma_f\Phi^m}{\int dV\nu\Sigma_f\Phi^{m-1}}$;
    if $\frac{K_{\text{eff}}^m - K_{\text{eff}}^{m-1}}{K_{\text{eff}}^{m-1}} < \eta_1$ **then**
      $k_{\text{eff}}^n = K_{\text{eff}}^m$;
      $\phi^n = \Phi^m$ break;
    end
  end
  if $\frac{k_{\text{eff}}^n - k_{\text{eff}}^{n-1}}{k_{\text{eff}}^{n-1}} < \eta_2$ **then**
    break;
  end
end

In Algorithm 2 a transport sweep is performed for each power iteration, and the inner drift-diffusion equation solves a fixed-source problem. Therefore, the number of transport sweeps is a function of both the dominance ratio and spectral radius [9]. The spectral radius defines the convergence rate of NDA. On the other hand, Algorithm 3 consists of inner-outer iterations. The transport sweep is performed at each outer iteration ($n-$loop) to obtain an intermediate drift-velocity term ($\hat{D}$), and the inner iterations solves the eigenvalue problem with given $\hat{D}$ via the power method. In Algorithm 3, the number of transport sweeps is a function of spectral radius

and independent of dominance ratio. Thus we expect the Algorithm 3 requires less number of transport sweeps, consequently more efficient than the Algorithm 2.

Note that Algorithm 3 is similar to [7], where the drift term, $\hat{D}$, is linearized at the previous iterate. There is merit in removing the Picard iteration and using the Newton iteration [8]. However, in many criticality problems, the dominance ratio may be higher than the scattering ratio and the Picard iteration is sufficient.

2.3 Full Nonlinear Algorithm for Transport Criticality Problems

With consistent HO and LO equations at hand, we can solve the inner iterations of CMFD-PI algorithm more efficiently by replacing the power iterations with Newton based JFNK approach. In this regard, we cast the LO equation as the following PDE constraint equation:

$$F_\phi(\phi, k_{\text{eff}}) = \frac{d}{dx}\left[-\frac{1}{3\Sigma_t}\frac{d\phi}{dx} + \hat{D}\phi\right] + (\Sigma_t - \Sigma_s)\,\phi - \frac{\nu\Sigma_f}{k_{\text{eff}}}\phi = 0, \tag{8}$$

$$F_{k_{\text{eff}}}(\phi, k_{\text{eff}}) = k_{\text{eff}} - \int dx\ \nu\Sigma_f\phi = 0. \tag{9}$$

Eq. (8-9) can be solved with the Jacobian-free Newton-Krylov (JFNK) method.

Jacobian-free Newton-Krylov method We solve the coupled Eqs. (8)-(9) using the JFNK method [10]. JFNK is a synergistic combination of Newton and Krylov methods. Newton's method iteratively solves the nonlinear system,

$$F(\mathbf{U}) = 0. \tag{10}$$

The first-order Taylor expansion of Eq. (10) about a k^{th} iterate is

$$F(\mathbf{U}) \approx F(\mathbf{U}^k) + \frac{\partial F}{\partial \mathbf{U}^k}(\mathbf{U} - \mathbf{U}^k). \tag{11}$$

We define the ij^{th} element of the Jacobian matrix as

$$\mathbb{J}_{i,j} \equiv \frac{\partial F_i}{\partial U_j}. \tag{12}$$

Then we solve the linear system of the form,

$$\mathbb{J}^k \delta\mathbf{U}^k = -F(\mathbf{U}^k), \tag{13}$$

at each Newton iteration, where $\delta\mathbf{U}^k = \mathbf{U}^{k+1} - \mathbf{U}^k$. For increased problem complexities, the explicit form of the Jacobian matrix soon becomes difficult or impossible to form. To remedy this difficulty, we take an advantage of the fact that Krylov methods require only the matrix-vector product, not the matrix itself. A JFNK method approximates the matrix vector product, $\mathbb{J}\mathbf{v}$ by,

$$\mathbb{J}\mathbf{v} \approx \frac{F(\mathbf{U} + \epsilon\mathbf{v}) - F(\mathbf{U})}{\epsilon}. \tag{14}$$

The matrix-vector product can be approximated with the finite difference form and only requires nonlinear residual functions.

Preconditioning Preconditioning is a key to efficiently solve a linear system iteratively. In JFNK, we use the right-preconditioned system,

$$\mathbb{J}\mathbb{M}^{-1}\mathbb{M}\delta\mathbf{U} = -F(\mathbf{U}), \tag{15}$$

where $\mathbb{M}$ and $\mathbb{M}^{-1}$ symbolize the preconditioning operator and preconditioning process. The corresponding matrix-vector product can be approximated by,

$$\mathbb{J}\mathbb{M}^{-1}\mathbf{v} \approx \frac{F(\mathbf{U} + \epsilon\mathbb{M}^{-1}\mathbf{v}) - F(\mathbf{U})}{\epsilon}. \tag{16}$$

Thus, the preconditioned system requires an extra step to perform $\mathbb{M}^{-1}\mathbf{v}$. Efficient operator-splitting and/or semi-implicit methods are often utilized for the preconditioning step [11–15].

2.3.1 Nonlinear elimination of eigenvalue (k_{eff})

With Newton-based linearization, the coupled nonlinear system, Eq. (8) and (9), can be expressed by the following matrix form,

$$\begin{bmatrix} \mathbb{J}_{\phi,\phi} & \mathbb{J}_{\phi,k_{\text{eff}}} \\ \mathbb{J}_{k_{\text{eff}},\phi} & \mathbb{J}_{k_{\text{eff}},k_{\text{eff}}} \end{bmatrix} \begin{bmatrix} \delta\phi \\ \delta k_{\text{eff}} \end{bmatrix} = - \begin{bmatrix} F_\phi(\phi, k_{\text{eff}}) \\ F_{k_{\text{eff}}}(\psi, k_{\text{eff}}) \end{bmatrix}, \tag{17}$$

where,

$$\mathbb{J}_{\phi,\phi} = -\frac{d}{dx}\left(\frac{1}{3\Sigma_t}\frac{d}{dx}\right) + \frac{d}{dx}\hat{D}(\psi^*) + (\Sigma_t - \Sigma_s) - \frac{\nu}{k_{\text{eff}}}\Sigma_f, \tag{18}$$

$$\mathbb{J}_{k_{\text{eff}},k_{\text{eff}}} = 1. \tag{19}$$

Note that $\mathbb{J}_{k_{\text{eff}},k_{\text{eff}}} = 1$ makes the inversion trivial. Therefore, we can reduce the system (17) via block Gaussian elimination. We refer to this technique as "nonlinear elimination" [16, 17]. Using the nonlinear elimination technique, we eliminate k_{eff} from Eq. (17) and obtain the following:

$$\left(\mathbb{J}_{\phi,\phi} - \mathbb{J}_{k_{\text{eff}},\psi}\mathbb{J}^{-1}_{k_{\text{eff}},k_{\text{eff}}}\mathbb{J}_{k_{\text{eff}},\phi}\right)\delta\phi = -F_\phi(\psi,\phi) + \mathbb{J}_{\phi,k_{\text{eff}}}\mathbb{J}^{-1}_{k_{\text{eff}},k_{\text{eff}}}F_{k_{\text{eff}}}(\phi, k_{\text{eff}}), \tag{20}$$

$$\tilde{\mathbb{J}}_{\phi,\phi}\delta\phi = -\tilde{F}_\phi(\phi). \tag{21}$$

Each GMRES iteration requires the matrix vector product (i.e., $\tilde{\mathbb{J}}_{\phi,\phi}\mathbf{v}$). The matrix vector product is approximated by the following two steps;

1. Evaluate k^*_{eff} from ϕ using Eq. (9).
2. Evaluate nonlinear residual function, Eq. (8), with k^*_{eff}.

Finally, the new algorithm (Algorithm 4) follows,

Algorithm 4: HO-LO nonlinear criticality search: CMFD-JFNK

initialize k_{eff}^0, ϕ^0 ;
for $n = 1, N$ **do**
  $\psi^* = \mathcal{L}^{-1}\left(\mathcal{S}\phi^{n-1} + \frac{1}{k_{\text{eff}}^{n-1}}\mathcal{F}\phi^{n-1}\right)$;
  compute $\hat{D}^* = \frac{\int d\mu\mu\psi^* + \frac{1}{3\Sigma_t}\frac{d\phi^*}{dx}}{\phi^*}$;
  solve $\tilde{\mathbb{J}}_{\phi,\phi}\delta\phi^{n-1} = -\tilde{F}_\phi(\phi^{n-1})$ with JFNK (and fixed $\hat{D} = \hat{D}^*$);
  set $\phi^n = \phi^{n-1} + d \times \delta\phi^{n-1}$;
  if $||F(\phi^n)||_2 < \gamma$ **then**
    break;
  end
end

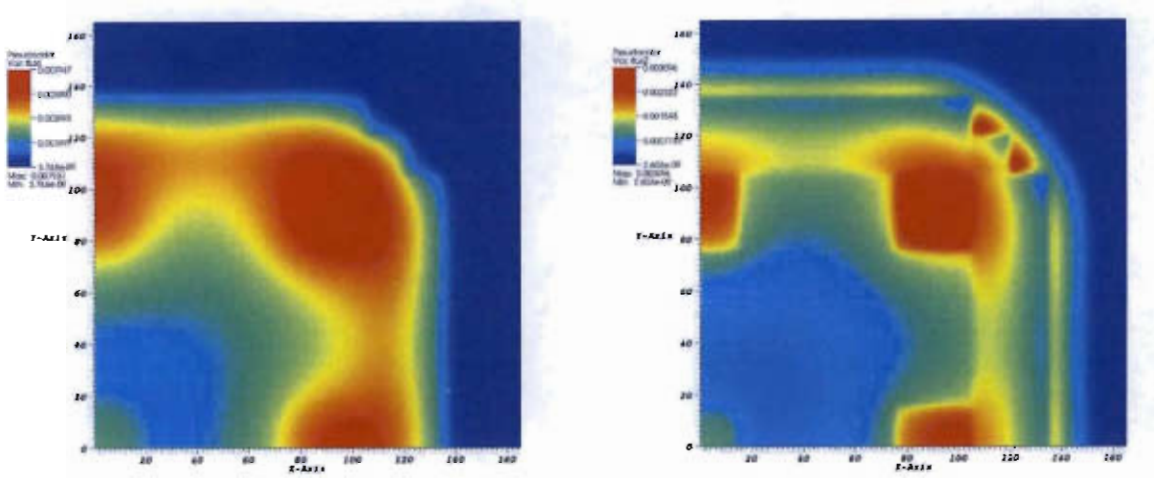
3. NUMERICAL RESULTS

In this section, we present a performance comparison among different transport criticality algorithms. To compare the efficiency of the developed algorithm, we report the number of transport sweeps and CPU time required for both the inner and outer iterations. There are two examples given below. Here we summarize the difference among all the algorithms.

- PI (Algorithm 1): This algorithm consists of the inner source iteration and outer power iteration. A transport sweep is performed at every inner iteration. The number of source iterations are adaptively changed with the following convergence criteria: $||\frac{\phi^m - \phi^{m-1}}{\phi^{m-1}}|| < 0.1||\frac{k_{\text{eff}}^n - k_{\text{eff}}^{n-1}}{k_{\text{eff}}^{n-1}}||$ or maximum of 50 iterations.
- CMFD-PI(0) (Algorithm 2): This algorithm requires a transport sweep per power iteration. The inner iteration loop solves the drift-diffusion equation with a fixed-source.
- CMFD-PI (Algorithm 3): This algorithm requires a transport sweep per outer iteration. The inner iteration loop solves the eigenvalue problem with a fixed drift velocity ($\hat{D}^*$) using standard power iterations.
- JFNK-CMFD (Algorithm 4): This algorithm requires a transport sweeps per outer iteration. The first 5 iterations are the same as Algorithm 2. This initialization step ensures that the initial guess is within a ball of convergence of subsequent Newton iterations. After initialization each outer iteration corresponds to one Newton step.

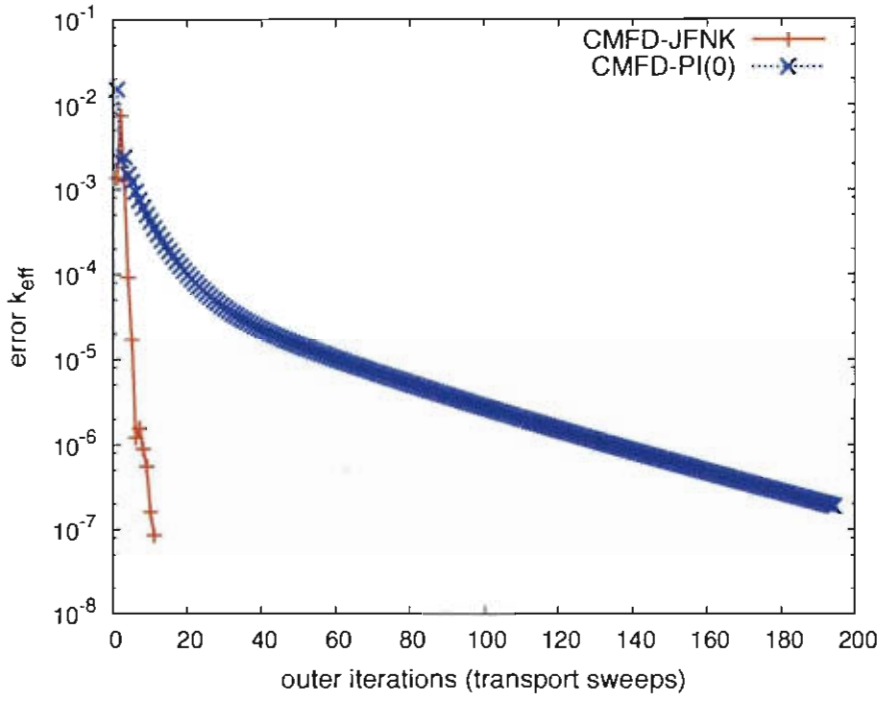
3.1 LRA-BWR Problem

The first example problem is the LRA-BWR problem [18]. This two-group two-dimensional problems represents simplified 1/4 BWR core. We solve this problem with S_N step-characteristic [19] spatial differencing on 165×165 mesh. Fig. 1(a) and 1(b) show the fast and thermal flux profiles of the problem and Fig. 1(c) shows k_{eff} convergence comparison between CMFD-PI(0)



(a) fast flux

(b) thermal flux



(c) k_{eff} convergence

Figure 1: Flux profiles and convergence plot in LRA-BWR transport problem.

and CMFD-JFNK with S_{16}. Fig. 1(c) clearly exhibits the advantage of CMFD-JFNK over CMFD-PI(0).

Table I, II and III show performance comparisons between PI, CMFD-PI(0), CMFD-PI and CMFD-JFNK. The reference k_{eff} values were obtained via Wynn-epsilon (W-e) convergence acceleration [20]. In this diffusive problem, convergence of the scattering source with CMFD is

a key for efficiency. Compared to the standard PI, employment of CMFD can reduce the CPU time by a factor of 14-400. CMFD-JFNK further reduces the CPU time by a factor of more than 10 compared to CMFD-PI(0) and 4-5 compared to the CMFD-PI in all cases. Though it is clear that reducing the number of transport sweep via CMFD is critical to decreasing the CPU time, reducing the number of inner iteration via NCA can also be very effective.

	Tran. Sweep	CPU inner (s)	CPU outer (s)	CPU total (s)	k_{eff}
PI	6605	–	–	1178.4	0.9988219
CMFD-PI(0)	195	24.9	39.4	82.65	0.9988257
CMFD-PI	20	22.0	6.0	46.9	0.9988257
CMFD-JFNK	12	4.7	2.6	8.3	0.9988329

Table I: S6 comparison of LRA-BWR problem, reference $k_{\text{eff}} = 0.9988327$

	Tran. Sweep	CPU inner (s)	CPU outer (s)	CPU total (s)	k_{eff}
PI	6645	–	–	2887.6	0.9989047
CMFD-PI(0)	196	25.0	78.8	132.7	0.9989083
CMFD-PI	20	22.6	8.0	52.8	0.9989083
CMFD-JFNK	11	4.6	4.4	10.7	0.9989156

Table II: S10 comparison of LRA-BWR problem, reference $k_{\text{eff}} = 0.9989153$

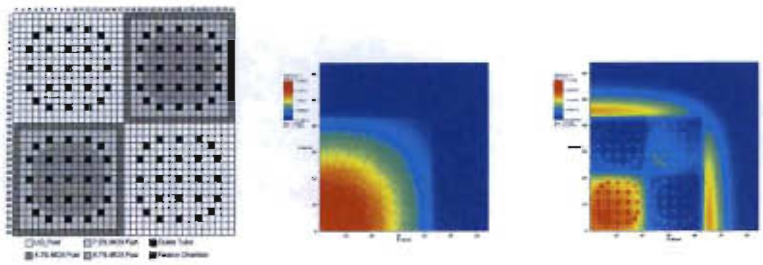
	Tran. Sweep	CPU inner (s)	CPU outer (s)	CPU total (s)	k_{eff}
PI	6620	–	–	6808.7	0.9989517
CMFD-PI(0)	198	25.2	198.0	248.2	0.9989556
CMFD-PI	20	22.1	20.0	63.4	0.9989555
CMFD-JFNK	11	4.5	11.0	17.3	0.9989629

Table III: S16 comparison of LRA-BWR problem, reference $k_{\text{eff}} = 0.9989626$

3.2 C5G7MOX Benchmark Problem

The second example is 2D-C5G7MOX benchmark problem[21]. The problem consists of 7-group with 4 up-scattering groups. The total number of unknowns in this problem is 892143 (i.e. 7×7 spatial mesh/fuel pin). This problem has a relatively small dominance ratio, therefore, the standard power iteration converges quickly. However, due to the detailed feature of the problem specification, the number of unknowns becomes large, and it is computationally demanding. Fig. 2(a) shows the layout of the fuel assemblies, while Fig. 2(b) and 2(c) depicts the fast and thermal flux profiles.

Table IV,V and VI compares the computational performance between CMFD-PI(0), CMFD-PI and CMFD-JFNK. Due to the small dominance ratio and scattering ratio of the problem, the CMFD-PI(0) only requires 39 iterations to converge. Yet for the large angular quadrature set (e.g., S16), CMFD-JFNK can significantly reduce the amount of CPU time.



(a) material layout (b) fast flux (c) thermal flux

Figure 2: **material layout and fast and thermal flux profiles of C5G7MOX benchmark problem**

Due to the small scattering ratio (and spectral radius) of the problem, only 4 transport sweeps were needed in CMFD-PI, and because of the initialization step of JFNK-CMFD, the number of transport sweeps required for JFNK-CMFD is larger than that of CMFD-PI. However, CMFD-JFNK was able to reduce the CPU time of inner iteration substantially (about a factor of 4), and consequently reduce the overall CPU time.

	Tran. Sweep	CPU inner(s)	CPU outer (s)	CPU total (s)	k_{eff}
CMFD-PI(0)	39	222.2	148.2	472.7	1.18536719
CMFD-PI	4	220.4	15.2	326.4	1.18537172
CMFD-JFNK	6	58.5	22.8	108.5	1.18537448

Table IV: S6 comparison of C5G7MOX benchmark problem, reference $k_{\text{eff}} = 1.18537429$

	Tran. Sweep	CPU inner(s)	CPU outer (s)	CPU total (s)	k_{eff}
CMFD-PI(0)	39	225.8	351.0	682.0	1.18590661
CMFD-PI	4	243.3	36.0	371.8	1.18591095
CMFD-JFNK	7	63.5	63.0	156.0	1.18591364

Table V: S10 comparison of C5G7MOX benchmark problem, reference $k_{\text{eff}} = 1.18591363$

	Tran. Sweep	CPU inner(s)	CPU outer (s)	CPU total (s)	k_{eff}
CMFD-PI(0)	39	222.3	834.6	1158.7	1.18649353
CMFD-PI	4	236.6	85.6	400.8	1.18649578
CMFD-JFNK	6	57.7	128.4	214.0	1.18650041

Table VI: S16 comparison of C5G7MOX benchmark problem, reference $k_{\text{eff}} = 1.18650054$

4. CONCLUSIONS

We have presented the nonlinear acceleration algorithms for the transport criticality problems. We combined the two nonlinear acceleration algorithms previously presented [5–8]. The comparison among the different algorithms shows that CMFD is very effective to reduce the required number of transport sweeps, while the HO-LO nonlinear acceleration algorithm can further reduce the CPU time by accelerating inner iterations (i.e., eigenvalue solver for the drift-diffusion

equations). Consequently, CMFD-JFNK reduces the CPU time by a factor of 400 in very diffusive problems compared to the standard power-source iteration.

ACKNOWLEDGEMENTS

The submitted manuscript has been authored by a contractor of the U.S. Government under Contract No. DEAC07-05ID14517 (INL/CON-11-20817). Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

REFERENCES

1. Y. Saad, Numerical Methods for Large Eigenvalue Problems, Halstead Press, New York, 1992.

2. Y. Saad, Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems, Math. Comput. 42 (166) (1984) 567–588.

3. J. Warsa, T. Wareing, J. Morel, J. McGhee, R. Lehoucq, Krylov subspace iterations for deterministic k-eigenvalue calculations, Nucl. Sci. Eng. 147 (1) (2004) 26–42.

4. V. Mahadevan, J. Ragusa, Novel hybrid scheme to compute several dominant eigenmodes for reactor analysis problems, in: International Conference on the Physics of Reactors, Interlaken, Switzerland, 2008.

5. D. Gill, Y. Azmy, Newton's method for solving k-eigenvalue problem in neutron diffusion theory, Nuclear Science and Engineering 167.

6. D. Knoll, H. Park, C. Newman, Acceleration of k-eigenvalue/criticality calculations using the Jacobiau-free Newton-Krylov method, Nuclear Science and Engineering 167.

7. K. Smith, J. Rhodes III, Full-core, 2-d LWR core calculatious with CASMO-4E, 2002, PHYSOR 2002, Seoul, Korea.

8. D. Knoll, H. Park, K. Smith, Application of the Jacobian-free Newton-Krylov method to nonlinear acceleration of transport source iteration in slub geometry, Nuclear Science and Engineering 167.

9. A. Yamamoto, Generalized coarse-mesh rebalance method for acceleration of neutron transport calculations, Nuclear Science and Engineering 151 (2005) 274–282.

10. D. A. Knoll, D. E. Keyes, Jacobian-free Newton-Krylov methods: a survey of approaches and applications, J. Comput. Phys. 193 (2) (2004) 357–397.

11. V. A. Mousseau, D. A. Knoll, W. J. Rider, Physics based preconditioning and the Newton-Krylov method for nonequilibrium radiation diffusion, J. Comput. Phys. 160 (2000) 743–765.

12. L. Chacon, D. A. Knoll, J. M. Finn, An implicit, nonlinear reduced resistive MHD solver, J. Comput. Phys. 178 (2002) 15–36.

13. J. M. Reisner, A. Wyszogrodzki, V. A. Mousseau, D. A. Knoll, An efficient physics-based preconditioner for the fully implicit solution of small-scale thermally driven atmospheric flows, J. Comput. Phys. 189 (2003) 30–44.

14. K. J. Evans, D. A. Knoll, M. A. Pernice, Development of a 2-D algorithm to simulate convection and phase transition efficiently, J. Comput. Phys. 219 (2006) 404–417.

15. H. Park, R. R. Nourgaliev, R. C. Martineau, D. A. Knoll, On physics-based preconditioning of the navier–stokes equations, Journal of Computational Physics 228 (24) (2010) 9131–9146.

16. D. Young, W. Huffman, R. Melvin, C. Hilmes, F. Johnson, Nonlinear elimination in aerodynamic analysis and design optimization, in: LARGE-SCALE PDE-CONSTRAINED OPTIMIZATION, Vol. 30 of LECTURE NOTES IN COMPUTATIONAL SCIENCE AND ENGINEERING, SPRINGER-VERLAG BERLIN, 2003, pp. 17–43.

17. P. Lanzdron, D. Rose, J. Wilkes, An analysis of approximate nonlinear elimination, SIAM Journal on Scientific Computing 17 (1996) 538–559.

18. Argonne code center: Benchmark problem book, ANL-7416 supplement 2, Argonne National Laboratory (1977).

19. K. Lathrop, Spatial differencing of the transport equation: Positivity vs. accuracy, Journal of Computational Physics 4 (1969) 475–498.

20. G. Baker, P. Grovers-Morris, Pade Apporoximations, Cambridge University Press, New York, 1960.

21. M. Smith, E. Lewis, B. Na, Benchmark on deterministic 2-d mox fuel assembly transport calculations without spatial homogenization, Progress in Nuclear Energy 45 (2-4) (2004) 107–118. doi:10.1016/j.pnuceene.2004.09.003.