CASL-X-2016-1106-000

Performance Improvements for Coarse Mesh Finite Difference Acceleration

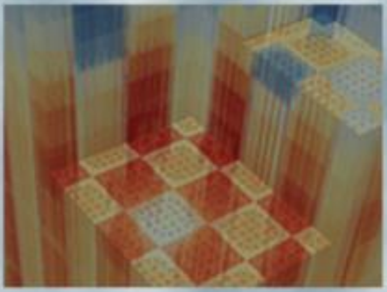
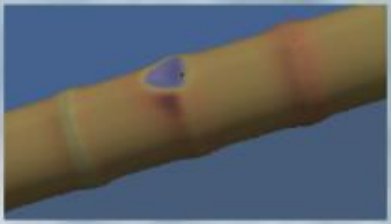
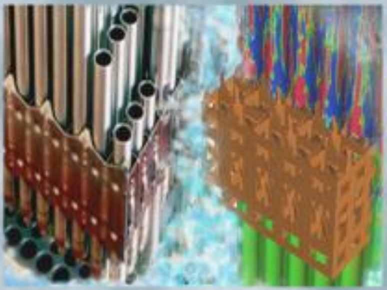
L3:RTM.PRT.P13.02

Benjamin Collins
Steven Hamilton
Shane Stimpson
Oak Ridge National Laboratory

Ben Yee
Edward Larsen
Brendan Kochunas
University of Michigan

5/31/2016

This research was supported by the Consortium for Advanced Simulation of Light Water Reactors (www.casl.gov), an Energy Innovation Hub (http://www.energy.gov/hubs) for Modeling and Simulation of Nuclear Reactors under U.S. Department of Energy Contract No. DE-AC05-00OR22725.

Oak Ridge National Laboratory

in partnership with

Electric Power Research Institute
Idaho National Laboratory
Los Alamos National Laboratory
Massachusetts Institute of Technology
North Carolina State University
Sandia National Laboratories
Tennessee Valley Authority
University of Michigan
Westinghouse Electric Company

and individual contributions from

Anatech Corporation
ASCOMP GmbH
CD-adapco, Inc
Core Physics, Inc.
City University of New York
Florida State University
Notre Dame University
Imperial College London
Pacific Northwest National Laboratory
Pennsylvania State University
Rensselaer Polytechnic Institute
Southern States Energy Board
Texas A&M University
University of Florida
University of Tennessee
University of Wisconsin

REVISION LOG

Revision	Date	Affected Sections	Revision Description
Rev. 0	5/31/2016	All	Initial release

1 Introduction

The development of VERA-CS in recent years has focused on developing the capability to simulate multiple cycles of operating commercial nuclear power plants. Now that these capabilities have advanced to the point where it is being deployed to users, the focus is on improving the computational performance of various components in VERA-CS. In this work, the focus is on the Coarse Mesh Finite Difference (CMFD) [1] solution in MPACT. CMFD serves multiple purposes in the 2D/1D solution methodology. First, it is a natural mechanism to tie together the radial MOC transport and the axial SP_3 solution. Because the CMFD system solves the multigroup three-dimensional core in one system, it pulls together the global response of the system. In addition, the CMFD solution provides a framework to accelerate the convergence of the eigenvalue problem.

The CMFD methodology is based on the 0th moment of the neutron transport equation

$$\nabla \bullet J_g + \Sigma_{t,g}\phi_g = \sum_{g'} \Sigma_{s,g'\to g}\phi_{g'} + \frac{\chi_g}{k_{eff}} \sum_{g'} \nu\Sigma_{f,g'}\phi_{g'} \ .$$

This equation is discretized onto a three dimensional Cartesian grid

$$\sum_{\substack{f\in n,s,e,\\ w,t,b}} J_{g,f} \bullet A_f + \overline{\Sigma}_{t,g,i}\overline{\phi}_{g,i}V_i = \sum_{g'} \overline{\Sigma}_{s,g'\to g,i}\overline{\phi}_{g',i}V_i + \frac{\overline{\chi}_{g,i}}{k_{eff}} \sum_{g'} \nu\overline{\Sigma}_{f,g',i}\overline{\phi}_{g',i}V_i \ .$$

So far there have been no approximations made to the multigroup transport equation but there are two unknowns, the current and the scalar flux. The CMFD methodology creates a relationship between the current and the flux as follows:

$$J_s = -\frac{2D^+D^-}{\Delta\left(D^+ + D^-\right)}\left(\overline{\phi}^+ - \overline{\phi}^-\right) + \hat{D}_s\left(\overline{\phi}^+ + \overline{\phi}^-\right) ,$$

where + and - denote the cells on each side of any given surface s. This equation defines the relationship between the current and the cell average flux using a coarse mesh diffusion approximation but an additional term ($\hat{D}_s$) is added to correct the error in this approximation. This equation is still exact but two unknowns still exist, the cell average flux, and a nonlinear correction coefficient, $\hat{D}_s$. The solution marching scheme used will approximate this correction coefficient using the high order MOC solution in the radial direction and the SP_3 solution in the axial direction for the current iterate;

$$\hat{D}_s = \frac{J_s^{MOC} + \frac{2D^+D^-}{\Delta\left(D^+ + D^-\right)}\left(\overline{\phi}^+ - \overline{\phi}^-\right)}{\left(\overline{\phi}^+ + \overline{\phi}^-\right)} \ .$$

The iteration scheme in MPACT first assumes $\hat{D}_s$ is zero and solves the full core CMFD equations. Once the coarse mesh fluxes are obtained, they are projected onto the fine MOC mesh and a single MOC sweep is performed. The MOC equations solver for the current on the boundaries of the coarse mesh which are used to estimate a new value for $\hat{D}_s$. This process is repeated until both the coarse and fine mesh solutions are converged.

Another key purpose of CMFD is to accelerate the solution to the eigenvalue problem. In order to consider this, it is easier to adopt a matrix notation of the CMFD system. There are four major components to the matrix notation: the diffusion operator $\mathbf{D}$, the collision operator $\mathbf{T}$, the scattering operator $\mathbf{S}$, and the fission operator $\mathbf{F}$. These four terms are combined into the generalized CMFD eigenvalue problem

$$\left(\mathbf{D}+\mathbf{T}-\mathbf{S}\right)\phi=\frac{1}{k_{eff}}\mathbf{F}\phi$$

and is simplified to

$$\mathbf{M}\phi=\frac{1}{k_{eff}}\mathbf{F}\phi\ .$$

The main focus of this work will be accelerating the convergence of this eigenvalue problem by looking at advanced solution schemes to solve for the dominant eigenvalue of this linear system.

2 Solution Methodology

Several methods have been proposed over the years to solve this generalized eigenvalue problem, this section will highlight many of the different approaches but will focus on the current methodology (power iteration with a fixed shift) and the new methodology (Generalized Davidson).

One of the most common methods to solve the CMFD eigenvalue system is to cast the generalized eigenvalue problem into a standard eigenvalue problem,

$$\mathbf{M}^{-1}\mathbf{F}\phi=k_{eff}\phi\,.$$

Since the migration operator is non-singular, it can be inverted and moved to the LHS of the equation. Once we have a standard eigenvalue problem, power iteration can be applied to obtain the dominant eigenvalue:

$$\phi^{(n+1)}=\mathbf{M}^{-1}\mathbf{F}\phi^{(n)}.$$

Typically the storing the full factorization of the migration operator is impractical for most problems so instead, a linear system is solved each iteration:

$$y=\mathbf{M}^{-1}\mathbf{F}\phi,$$
$$z=\mathbf{F}\phi,$$
$$\mathbf{M}y=z.$$

Although this iteration scheme is straightforward, the convergence of power iteration is highly dependent on the dominance ratio,

$$\rho=\frac{k_2}{k_1}\ ,$$

where k_1 and k_2 are the largest and second largest eigenvalues of system. This convergence rate can be improved by introducing a shift to the eigenvalue system,

$$\left(\mathbf{M}-\mu\mathbf{F}\right)^{-1}\mathbf{F}\phi=\tilde{k}\phi$$

where μ is an approximate to the inverse of the dominant eigenvalue. It is straightforward to show that the eigenvalue of the original system relates to the eigenvalue of the shifted system by

$$k_{eff} = \frac{\tilde{k}}{1+\mu\tilde{k}}$$

and the dominance ratio of the shifted system can be significantly reduced. Shifted power iteration does not come without risk. First, an incorrect choice of shift could alter the convergence to an eigenvalue which does not reflect the largest eigenvalue of the original system. Secondly, as the shift becomes closer to the true solution of the original system, the linear system becomes more and more difficult to be solved. The obvious extreme is if the shift is exactly the inverse of the true eigenvalue of the system, $\left(\mathbf{M}-\mu\mathbf{F}\right)$ is a singular matrix so care must be taken when selecting the choice of shift and solving the shifted system.

Traditionally in MPACT a fixed shift of 2/3 is used. This was determined to be a decent balance between performance of the solver and ensuring that the system is not over-shifted. Another approach that has been used in many other places is to have a variable shift based on the current eigenvalue iterate and potentially previous iterates. PARCS [2] uses a variable shift based on iteration history:

$$\mu^{(n)} = \mu_P^{(n)} = \max\left\{\frac{1}{k^{(n-1)}} - C_1\left|\frac{1}{k^{(n-1)}} - \frac{1}{k^{(n-2)}}\right| - C_0, \mu_{\min}\right\}$$

where C$_0$, C$_1$, and $\mu_{\min}$ are constants (0.02, 10, and 1/3 respectively). This method tries to keep enough margin from the actual eigenvalue to ensure stability of the method.

Another common iteration dependent approach is the Rayleigh Quotient Iteration (RQI) [3] which uses the current eigenvalue estimate as the shift. The RQI iteration strategy becomes

$$\phi^{(n+1)} = \left(\mathbf{M}-\mu^{(n)}\mathbf{F}\right)^{-1}\mathbf{F}\phi^{(n)},$$

$$\mu^{(n+1)} = \frac{\left\langle \phi^{(n+1)}, \mathbf{M}\phi^{(n+1)}\right\rangle}{\left\langle \phi^{(n+1)}, \mathbf{F}\phi^{(n+1)}\right\rangle},$$

where $\langle \bullet,\bullet \rangle$ denotes the inner product of two vectors. The main advantage of RQI is the convergence is quadratic and thus very little iteration is required for convergence. But the linear system becomes increasingly more difficult to solve. RQI is also not guaranteed to converge to the dominate eigenvalue.

Recent work at the University of Michigan [4] has considered a space dependent shift which looks at the local infinite eigenvalue of each CMFD cell and uses that locally as the shift,

$$\phi^{(n+1)} = \left(\mathbf{M}-\mu_c\mathbf{F}\right)^{-1}\mathbf{F}\phi^{(n)},$$

$$\mu_c = \mu_{\infty,c} = \frac{1}{\left\langle \left(\Sigma_{t.c} - \Sigma_{s,c}\right)^{-1}\chi_c, \nu\Sigma_{f,c}\right\rangle}.$$

This method was further enhanced to ensure the source term is positive by adapting the shift to consider the smaller of the infinite shift and the current shift

$$\mu_{PS,c}^{(n)} = \min\left\{\mu_{\infty,c}, \frac{1}{k^{(n)}}\right\}.$$

Another variant of the space dependent shift is to use the PARCS shifting scheme instead of the current iterate to ensure the solution does not approach a singular system and become more difficult to solve,

$$\mu_{IPS,c}^{(n)} = \min\left\{\mu_{\infty,c}, \mu_{P}^{(n)}\right\}.$$

These variations are implemented in MPACT but are still undergoing evaluation to determine the effectiveness of this method.

Thus far, all of the methods discussed have been fixed-point methods (i.e., the next estimate of the solution depends only on the estimate immediately preceding it). An alternative to fixed-point iterations is subspace eigenvalue solvers in which information from several previous vectors is used to generate the next approximate solution. In order to extract the solution, an estimate of the eigenvalue is obtained as a linear combination of the subspace basis vectors through a Rayleigh-Ritz procedure which solves the projected eigenvalue problem:

$$\mathbf{V}^T\mathbf{M}\mathbf{V}y = \lambda\mathbf{V}^T\mathbf{F}\mathbf{V}y$$

where $\mathbf{V}$ contains a set of (typically orthogonal) basis vectors for the current subspace. For an appropriate selection of the subspace, the eigenvalues of the projection problem will closely approximate the original system and the vectors of $\mathbf{V}y$ will approximate the eigenvector. The approximate eigenvalues and eigenvectors obtained from the Rayleigh-Ritz procedure are generally referred to as Ritz values and Ritz vectors respectively.

The determination of the new vectors to place into the subspace is generally what distinguishes the majority of subspace eigenvalue solvers. The Arnoldi method [5], the subspace is taken to the Krylov subspace corresponding to the operator $\mathbf{M}^{-1}\mathbf{F}$, which is similar to power iteration. Unlike power iteration, the convergence of Arnoldi's method is not dictated by the dominance ratio and so the number of iterations required to converge can be significantly smaller.

Another subspace eigenvalue solver that is the main focus for implementation of this work is the Davidson method [6]. The idea behind the Davidson method is that at iteration n, one should seek a correction $t^{(k)}$, such that the eigenvalue equation given by

$$\mathbf{M}\left(\phi^{(n)} + t^{(n)}\right) = \lambda^{(n)}\mathbf{F}\left(\phi^{(n)} + t^{(n)}\right).$$

This equation can be rearranged to

$$\left(\mathbf{M} - \lambda^{(n)}\mathbf{F}\right)t^{(n)} = \left(\mathbf{M} - \lambda^{(n)}\mathbf{F}\right)\phi^{(n)} \equiv -r^{(n)},$$

where $r^{(n)}$ is the residual of the eigenvalue problem. The evaluation of $t^{(n)}$ would require a linear system solution of a nearly singular system. In order to avoid this computational expense, $\left(\mathbf{M}-\lambda^{(n)}\mathbf{F}\right)$ is approximated using a preconditioner $\mathbf{P}$, which leads to the Davidson correction equation

$$\mathbf{P}t^{(n)} = -r^{(n)}.$$

The Davidson method is extremely attractive because unlike all of the previous methods discussed, a linear system solve is not required. Instead, only a preconditioner application to an approximate matrix is needed. The choice of preconditioner is important for Davidson to quickly converge. In order to avoid singularities in the preconditioner system, the preconditioner is chosen based on the migration operator $\mathbf{M}$ ranther than the shifted operator $\left(\mathbf{M}-\lambda^{(n)}\mathbf{F}\right)$. Several different preconditioners based on $\mathbf{M}$ are explored in this work.

3 Linear Solvers and Preconditioners

There are several methods available in MPACT to gauge the performance of the new CMFD eigenvalue iteration. There were several preexisting solvers prior to this work but only two work in parallel and are the focus of this work. The first was a custom red-black Successive Over-relaxation solver (RBSOR) which works in performs local and global sweeps to determine the coarse mesh flux. The RBSOR method uses a dynamic relaxation factor by determining the spectral radius of the problem over the first several iterations. The second parallel solver leverages the PETSc linear solver [7] package developed by Argonne National Laboratory. PETSc has multiple Krylov subspace solvers including (GMRES and BICGStab) and a range of preconditioners. PETSc was attractive because of the availability of native Fortran interfaces to many of the features. The PETSc sovlers have been a key component of MPACT for several years. In both solvers, the shifted power method was employed using a fixed shift.

As part of this work, several additional directions were pursued. Based on the Insilico SP$_N$ work [8] that was focused on in CASL, pursuit of the Krylov Subspace eigenvalue problems was strongly pursued. The first attempt at using these new solutions types was to use the SLEPc package [9] developed by Universitat Politecnica de Valencia, Spain. This package wraps many of the features in PETSc and implements several of the Krylov subspace based eigenvalue solvers discussed earlier. Although initial comparisons showed very promising results, it was determined that these solvers began to struggle for large systems (3D quarter core). It is strongly believed that these methods could be used efficiently if a stronger preconditioner was available in PETSc but currently one was not found. The RBSOR solver in MPACT was even attempted as a preconditioner for these problems with marginal success. It is possible that extended work in this area could show improvements.

Instead, the decision was made to pursue the solvers in the Trilinos package [10] developed at Sandia National Laboratory. Because of the expertise in CASL, using Trilinos makes a lot of sense and it has been proven to work well on very similar problems with the previous SP$_N$ work. The drawback to the Trilinos approach is that Fortran interfaces do not exist so they were developed as part of this work. Future enhancements of Trilinos may consider adding in more general Fortran support. In order to expose the features needed, Fortan to C to C++ wrappers were added for Epetra (vectors and matrices), Belos (linear solvers), Anasazi (Eigenvalue solvers), IFPACK (incomplete factorization preconditioners), and ML (algebraic multigrid preconditioners). While these interfaces do not completely expose the functionality in these packages, this represents a subset of capability needed for this work.

4 Results

In order to demonstrate the performance of MPACT and the CMFD solver a range of cases and core counts were run on two different machines. The majority of the cases for this work were run on the Titan computer maintained by OLCF but a few cases are demonstrated on the Falcon computer maintained by INL-HPC. While it has been observed that the overall performance of MPACT is significantly better on Falcon, its availability has been significantly limited because of other VERA-CS runs.

The first case considered is the VERA Progression Problem 5-2D, which is a 2D slice of a quarter core PWR. This case is run on 15, 73 (assembly based), and 257 (fully decomposed) processors. The 15 processor partition is not necessarily optimal but it is as close as possible to obtain an ~1000 core solution for a quarter core problem. Table 1 shows the number of iterations, the total run time for MPACT, and the CMFD solve time. Three cases are shown, a constant shift of 1.5 with PETSC, the RBSOR solver with a constant 1.5 shift, and generalized Davidson with the algebraic multigrid solver.

Table 1: Problem 5-2D Performance on Titan

	15 procs			**73 procs**			**257 procs**		
	Its.	Total	CMFD	Its.	Total	CMFD	Its.	Total	CMFD
Shift=1.5	12	46:50.7	35:58.1	11	10:15.4	07:26.1	15	05:13.0	04:08.3
RBSOR (1.5)	12	22:56.1	12:13.1	11	06:50.8	04:09.2	15	02:14.5	01:13.7
Gen. Davidson - AGM	9	09:54.3	01:20.0	11	02:15.8	00:18.3	17	01:13.7	00:13.2

It can be seen that across all problems, the CMFD run time is significantly decreased which results in reductions in runtime of 4-5x.

An extension to this problem is a 2D core depletion to 500 EFPD to demonstrate the impact of the methods during depletion. Unfortunately, the wall time on Titan is insufficient to deplete the 2D core with 15 processors with the default solver. Both MPACT with the default solver and the Generalized Davidson solver were run with the maximum 6 hour wall time. Table 2 shows the time per iteration and the total run time for each state.

Table 2: Problem 5-2D Depletion Times

	Shift=1.5		**Gen. Davidson - AMG**	
	Time Per Iteration	**Total**	**Time Per Iteration**	**Total**
1	42:52.8	46:56.8	05:50.3	0:09:54
2	36:35.2	1:23:32	13:19.6	0:23:13
3	41:56.0	2:05:28	10:13.6	0:33:27
4	44:33.0	2:50:01	11:20.2	0:44:47
5	49:31.0	3:39:32	11:46.0	0:56:33
6	49:09.0	4:28:41	12:01.9	1:08:35
7	51:06.0	5:19:47	14:34.0	1:23:09
8			15:00.0	1:38:09
9			16:43.0	1:54:52
10			15:06.0	2:09:58
11			17:07.0	2:27:05

12		16:00.0	2:43:05
13		14:55.0	2:58:00
14		14:03.0	3:12:03
15		18:16.0	3:30:19
16		14:36.0	3:44:55
17		15:43.0	4:00:38
18		14:56.0	4:15:34
19		14:37.0	4:30:11
20		14:41.0	4:44:52
21		15:12.0	5:00:04
22		15:34.0	5:15:38
Average	**45:06.1**	**14:09.8**	

As it can be seen, the new Generalized Davidson solver can solve the full depletion in less time than 7 state points of the current solver. The performance is roughly the same throughout the entire depletion.

The second case considered is the full height quarter core VERA Progression Problem 5a. This case is run on 870 (15 radial partition) and 4234 (assembly radial partition) processors. This is a target capability for MPACT to solve. The 15 core radial partition with 58 axial planes yields a less than 1000 core solution to quarter core PWR geometry which is the target platform for VERA-CS.

Table 3 shows the performance of the problem 5a performance on titan. Again, the three solution methods are evaluated at the ~1000 core solution and the current 4234 core solution. The RBSOR result for the 4234 core solution is not currently available. Like the 2D cases, significant decreases in CMFD and total run times are observed in all cases.

Table 3: Problem 5a Performance on Titan

	870 procs			**4234 procs**		
	Its.	Total	CMFD	Its.	Total	CMFD
Shift=1.5	13	1:05:25	47:23.8	12	19:57.9	12:18.8
RBSOR (1.5)	14	37:54.7	19:57.2	--	--	--
Gen. Davidson - AGM	12	19:27.8	03:48.2	12	08:54.0	02:11.1

An extension to this problem is to run in full core to determine the difference in performance as the problem size is increased. These cases used 4408 processors on Titan. Only the default method and the Generalized Davidson method were used for this case. As seen in Table 4, the performance is consistent with the previous cases.

Table 4: Problem 5 Full Core Performance on Titan

	Its.	**Total**	**CMFD**
Shift=1.5	13	1:04:55	43:46.4
Gen. Davidson - AGM	12	0:27:03	08:46.8

The final case considered in this work is VERA Progression Problem 7, which couples the problem 5 cases used before to COBRA-TF and is run with full thermal hydraulic feedback. It is expected that the same trends seen in problem 5a will translate to this case but since problem 7 represents a fundamental capability of VERA-CS, it is necessary to demonstrate how these improvements translate to the coupled problem.

The extension to the coupled problem only considers the Generalized Davidson solver on both Titan and Falcon. Unfortunately, there are still issues with the algebraic multigrid on Falcon. The incomplete factorization preconditioner was successful in solving the coupled problem. It was determined that the ILU preconditioner is approximately 25% slower than the AMG preconditioner for similar problems on Titan. Regardless of the issues with the preconditioner, Table 5 shows promise for the Generalized Davidson solver on both Titan and Falcon.

Table 5: Problem 7 Performance on Titan and Falcon

	870 procs			4234 procs		
	Its.	Total	CMFD	Its.	Total	CMFD
Titan (AMG)	12	36:08.2	04:41.2	9	22:39.9	02:40.1
Falcon (ILU)	14	24:30.3	03:16.9	11	14:51.5	02:21.6

This demonstrates that improved performance on Falcon even though it is using a preconditioner that is not as optimal. Ultimately, on Falcon, the computational resources for problem 7 has dropped from 1048 CPU-hrs in April 2016 to 355 CPU-hrs by reducing the core count to 870 cores with similar run times.

5 Conclusions and Future Work

In this work, a new eigenvalue solution method was implemented into the CMFD solver in MPACT. This method leverages the experience and expertise that has been developed in CASL through the Insilico SP_N work. Trilinos interfaces were implemented to allow Fortran to call the eigenvalue solver Anasazi and the Generalized Davidson solver was used to solve the CMFD eigenvalue system. Both algebraic multigrid and incomplete LU preconditioners were used successfully. Overall, the CMFD solve time was reduced from the default MPACT methods by approximately 13x which yields an overall reduction in MPACT solve times of approximately 3.5x. This allows MPACT and VERA-CS production runs to require approximately 1000 cores instead of the 4000+ cores currently required without a significant change in runtime. VERA-CS can also be solved significantly faster if the same core count is used.

Although significant improvements in the runtime are observed, there is still room for improvement. Particularly, further testing is needed on the Trilinos interface to ensure that the capability is sufficiently maintained. Also, further testing of full core coupled depletion to ensure robustness and stability is needed. Understanding the performance of the algebraic multigrid preconditioner issues on Falcon also needs investigation. Lastly, there are minor performance tuning activities that can be made in CMFD such as adaptive iteration control, optimizing how often the preconditioner is updated, and optimizing parameters in the preconditioners.

6 Acknowledgements

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

This research made use of the resources of the High Performance Computing Center at Idaho National Laboratory, which is supported by the Office of Nuclear Energy of the U.S. Department of Energy under Contract No. DE-AC07-05ID14517.

7 Reference

1. K. Smith. Nodal Method Storage Reduction by Nonlinear Iteration. Transactions of the American Nuclear Society. v.44. p.265. (1983)
2. T. Downar. PARCS: Purdue Advanced Reactor Core Simulator. Physor. Seoul, South Korea. (2002)
3. F. Scheben, I. Graham. Iterative Methods for Neutron Transport Eigenvalue Problems. SIAM Journal of Scientific Computing. v.33. p.2785-2804. (2011).
4. B. Yee, E. Larsen, B. Kochunas, Y. Xu. Space-Dependent Wielandt Shift Methods for Multigroup Diffusion Eigenvalue Problems. Transactions of the American Nuclear Society. (Accepted 2016).
5. W. Arnoldi. The Principle of Minimized Iterations in the Solution of Matrix Eigenvalue Problems. Quarterly of Applied Mathematics. v.9. p.17-29. (1951).
6. R. Morgan. Davidson's Method and Preconditioning for Generalized Eigenvalue Problems. Journal of Computational Physics. v.89. p.241-245. (1990).
7. S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. Gropp, D. Kaushik, M. Knepley, L. McInnes, K. Rupp, B. Smith, S. Zampini, H. Zhang, and H. Zhang. PETSc User's Manual. ANL-95/11 Rev. 3.7. Argonne National Laboratory. (2016). http://www.mcs.anl.gov/petsc
8. S. Hamilton, T. Evans. Efficient Solution of the Simplified P_N Equations. CASL-U-2014-0352-000. Oak Ridge National Laboratory. (2014). http://www.casl.gov/docs/CASL-U-2014-0352-000.pdf
9. V. Hernandez, J. E. Roman, V. Vidal. SLEPc: A Scalable and Flexible Toolkit for the Solution of Eigenvalue Problems. ACM Trans. Math. Software, 31(3):351-362, (2005).
10. M. Heroux, J. Willenbring. Trilinos Users Guide. SAND2003-2952. Sandia National Laboratory. (2003).