Toward Restarting Strategies Tuning for a Krylov Eigenvalue Solver

France Boillod-Cerneux[1], Serge G. Petiton[1,2], Christophe Calvin[3] and Leroy A. Drummond[4]

[1] CNRS LIFL, Cité Scientifique, Bâtiment M3, 59655 Villeneuve d'Ascq, France
[2] Maison de la Simulation, Bâtiment 565 Digitéo, CEA Saclay, 91191 Gif-sur-Yvette, France
[3] CEA/DEN/DANS/DM2S/, CEA Saclay, 91191 Gif-sur-Yvette, France
[4] Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, 94720-8150 California, USA

Abstract. Krylov eigensolvers are used in many scientific fields, such as nuclear physics, page ranking, oil and gas exploration, etc... In this paper, we focus on the ERAM Krylov eigensolver whose convergence is strongly correlated to the Krylov subspace size and the restarting vector v_0, a unit norm vector. We focus on computing the restarting vector v_0 to accelerate the ERAM convergence. First, we study different restarting strategies and compare their efficiency. Then, we mix these restarting strategies and show the considerable ERAM convergence improvement. Mixing the restarting strategies optimizes the "numerical efficiency" versus "execution time" ratio as we do not introduce neither additionnal computation nor communications.

1 Introduction

In the large shade of nuclear physics applications and simulations, solving an eigenvalue problem is a common occurrence. One of the best example is the neutron transport equation which is the heart of physical processes in nuclear reactor simulations: Solving this equation requires to compute the dominant eigenpair of a large non-Hermitian matrix. In some other cases, we must compute a subset of eigenpairs as will be illustrated with the Fission matrix [3]: The more eigenpairs we compute, the more we improve the Monte-Carlo process convergence. The Krylov methods are good candidates for such problems [1],[2], as they can compute either the dominant or a subset of eigenpairs.

In this paper, we will focus on the Explicitly Restarted Arnoldi Method (ERAM) [4], [5]. The ERAM has some lacks to ensure the system convergence. Based on the Arnoldi process, it iteratively builds a Krylov subspace $\mathbb{K}_{m,v_0} = span\{v_0, Av_0, ..., A^{m-1}v_0\}$ and its associated matrices $H \in \mathbb{C}^{(m+1)\times m}$ and $V \in \mathbb{C}^{n\times(m+1)}$ that verify $AV = VH$, where H is the unitary projection of A onto $\mathbb{K}_{m,v_0}$ [4]. The H eigenvalues may be good approximations of A eigenvalues. We will denote by $s \in [1, m]_{\mathbb{N}}$ the number of desired eigenpairs and by $\theta_j \in \mathbb{C}, j \in [1, s]_{\mathbb{N}}$ the j^{th} real largest modulus eigenvalue. The H eigenvectors are projected onto $\mathbb{K}_{m,v_0}$ basis, leading to approximated eigenvectors corresponding to the θ_j eigenvalue. We will denote the approximated eigenvector associated to θ_j by $u_j \in \mathbb{C}^n$. We then compute the associated residual of approximated eigenpairs $res_j = \frac{||Au_j - \theta_j u_j||_2}{|\theta_j|}$: If this residual is small enough, for each desired eigenpairs, the ERAM has reached the convergence. Otherwise, we must restart the process until convergence, by using a new restarting vector v_0. This aims to force the new Krylov subspace $\mathbb{K}_{m,v_0}$ convergence to the desired eigensubspace. Throughout all the following paper, we will retain these notations. It is assumed that only a subset of m computed eigenpairs will be a good approximations of the non-Hermitian matrix $A \in \mathbb{C}^{n\times n}$ eigenpairs [4],[5] and many scientific research has been conducted to fix the s and m values in consequence. The ERAM is parallel and efficient on actual supercomputers but its convergence is not ensured, depending on many parameters which are tricky to fix correctly before the ERAM execution.

One of the most influent parameter of the ERAM convergence is the size m of $\mathbb{K}_{m,v_0}$. It is assumed that the larger m is, the better the ERAM convergence is [4], [5]. However, increasing m implies two issues that are part of the extreme-scale computing barriers. Firstly, we increase the data size, especially the $V \in \mathbb{C}^{n\times(m+1)}$ (dense matrix) size. Secondly, it implies more operations to execute the Arnoldi process leading to more blocking and global communications. A large m may rapidly lead to a bottleneck due to the Arnoldi process global/blocking communications. Many research has been done for the Krylov methods applied to the linear system resolution to fix the m value such as it optimizes the parallel time execution versus the numerical convergence ratio of the method [6],[7]. In the context of up-coming exascale computing, optimizing this ratio is fundamental and implies to re-design algorithms, even mathematical methods themselves [8].

In this paper, we aim to improve the ERAM convergence by changing only the restarting vector v_0, as it influences the ERAM convergence. Regarding the ERAM, there is no general method to compute v_0 and especially no method to ensure the $\mathbb{K}_{m,v_0}$ convergence to the desired eigensubspace. Usually, v_0 is a linear combination of the Ritz vectors u_j that are uniformly weighted. In this paper, we choose to focus on finding a pertinent u_j combination to compute a new v_0 as it requires neither parallel communications nor complex operations. In the first part we propose to use different restarting strategies and study their influence on the ERAM convergence. In the second part, we will mix the restarting strategies and show the convergence improvement of the ERAM using mixed restarting strategies versus the ERAM using a single restarting strategy. All this work is the basis of an upcoming smart-tuning to dynamically mix the restarting strategies and obtain better numerical performances without being affected by additionnal operations neither additionnal communications.

2 The ERAM Restarting Strategies

The ERAM starts its $i+1^{th}$ restart with a restarting vector $v_0^{(i+1)}$ using the last computed Ritz vectors $u_j^{(i)}, j \in [1,s]_{\mathbb{N}}$. Throughout all the following paper, we will annotate all previous definitions by their computing restart (i). We introduce $\alpha_j^{(i)}$, the restarting coefficient associated to j^{th} eigenpair for the i^{th} restart. Then, the restarting vector $v_0^{(i+1)}$ is computed as follows:

$$v_0^{(i+1)} = \sum_{j=1}^{s} \alpha_j^{(i)} u_j^{(i)}, s \in [1,m]_{\mathbb{N}} \tag{1}$$

We summarize the restarting strategies used in this paper in the Table 1. Throughout all this paper, we will refer to the restarting strategy using their restarting strategies abreviations. In the scientific litterature, α_{Def} is the most commonly used restarting strategy [5].

Restarting Strategy	Abreviation	$\alpha_j^{(i)}$ Value
Default	α_{Def}	1
Residual	α_{Res}	$\lvert 1 - \lvert res_j^{(i)} \rvert \rvert$
Linear	α_{Li}	$(s-j+1)$
Linear Residual	α_{LiRes}	$(s-j+1) \times \lvert 1 - \lvert res_j^{(i)} \rvert \rvert$
Lambda	α_{La}	$\lvert \theta_j^{(i)} \rvert$
Lambda Residual	α_{LaRes}	$\lvert \theta_j^{(i)} \rvert \times \lvert 1 - \lvert res_j^{(i)} \rvert \rvert$

Table 1. The ERAM Restarting Strategies: $j \in [1,s]_{\mathbb{N}}$ and $n >> m \geq s$.

3 The ERAM Restarting Strategies Influence

We summarize the target matrices properties in the Table 2. For each figures presented bellow, we use $s = 4$ and the Arnoldi process uses a CGSR orthogonalization process [9]. For all results presented in this section, the subspace size m is fixed during all the ERAM execution.

Matrix Name	Size	nnz	Field
Bayer04	20545	85,537	chemical process simulation problem
Bcspwr09	1723	6,511	power network problem
Ex11	16614	1,096,948	computational fluid dynamics problem
Fission	10000	100,000,000	nuclear physics reactor simulation
Rim_dense	22560	508,953,600	computational fluid dynamics problem

Table 2. the Target Matrices.

For each target matrix, we execute the ERAM with exactly the same parameters (m, s, the orthogonalization process, the threshold, the number of MPI tasks and the same hardware , cf the Table 3) excepted the restarting strategy. Such executions will emphasize the ERAM convergence with respect to the restarting strategy used. In this context, we use the number of ERAM restarts until convergence as the reference metric to compare our results. The restarting strategies do not impact the execution time per restart but only the number of restarts to reach the convergence (therefore the global execution time of the ERAM solver). As an illustration, the execution time per restart using α_{Def} is the same as α_{La} and all other restarting strategies.

We first present the results for each ERAM using only one restarting strategy during its execution (Table 3). The # symbols means the ERAM did not reach the convergence after 500 restarts.

Matrix Name/ERAM param.	Target Machine	α_{Def}	α_{Res}	α_{La}	α_{LaRes}	α_{Li}	α_{LiRes}
Bayer04, $m = 15, tol = 10^{-14}$	1 Intel i5-2430M	93	48	118	27	72	106
Bcspwr09, $m = 25, tol = 10^{-14}$	1 Intel i5-2430M	#	454	395	#	125	#
Bcspwr09, $m = 30, tol = 10^{-14}$	1 Intel i5-2430M	30	34	35	38	25	91
Ex11, $m = 20, tol = 10^{-14}$	1 Intel i5-2430M	148	45	55	10	229	#
Fission, $m = 10, tol = 10^{-13}$	400 Intel Sandy Bridge	46	#	80	#	300	#
Rim_dense, $m = 15, tol = 10^{-13}$	480 Intel Sandy Bridge	152	65	135	138	69	309

Table 3. Target matrices, ERAM executions using a Single Restarting Strategy, Results. The Fission and Rim_dense matrices ERAMs have been executed on the PRACE Curie (TGCC Saclay France) supercomputer thin nodes.

We will detail the restarting strategy efficiency of the Bayer04 matrix, as the conclusions can be generalized to all other target matrices. The best restarting strategy is α_{LaRes} leading to convergence with only 27 restarts. Then α_{Res} converges at restart 48, α_{Li} at restart 70 and α_{Def} at restart 93. α_{LaRes} saves 66 restarts compared to α_{Def} which is the most commonly used restarting strategy. For this configuration, all ERAMs converge with the parameters fixed before the runtime execution, however the convergence behavior between the best and the worst restarting strategies is really different.

The number of restarts to reach the convergence varies for each configuration and each target matrix: There are no tools to prevent before the runtime execution which restarting strategy will provide the best convergence scheme. We can not emphasize a "best restarting strategy" that ensure the best convergence scheme for all target matrices, as restarting strategies efficiency depends on the matrix itself and the ERAM execution.

4 The ERAM with Mixed Restarting Strategies

In this paper, we focus on the ERAM using α_{Def} during its complete execution. The same work has been done for all other restarting strategies, leading to the same conclusions. We choose to illustrate the mixed restarting strategies based on ERAM using α_{Def} as it is the most commonly used restarting strategy. All the following results considering the ERAM with mixed restarting strategies are executed on the same number of MPI tasks as presented in the Table 3.

We executed an algorithm that detects the ERAM convergence based on $res_j^{(i)}$ analysis (not detailed in this paper). The convergence algorithm detects the stagnation or divergence states for the ERAM using α_{Def}. We retain the restarts listed by the convergence algorithm where such divergence or stagnation status have been detected. We re-executed the ERAM using α_{Def} and switch α_{Def} by another restarting strategy at the corresponding restarts, leading to an ERAM using mixed restarting strategies. We experimented different combinations of mixed restarting strategies and operated the switch restarting strategies at different restarts. The Figure 1 presents the process used to evaluate the ERAM with mixed restarting strategy performances versus an ERAM using only α_{Def}.

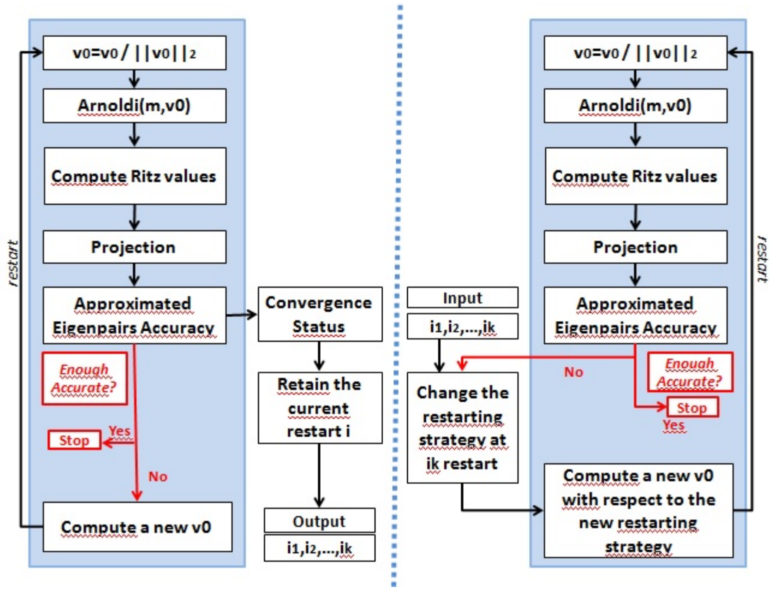

Fig. 1. We execute an ERAM using α_{Def} during the complete execution. We evaluate its convergence status and retain only the $i^{(th)}$ restarts (that we denote by i_k) when a stagnation or a divergence status has been detected. We re-execute the ERAM using α_{Def} and change α_{Def} by another restarting strategy at i_k restart.

The following figure shows the ERAM with mixed restarting strategies applied to the Bayer04 matrix. The results regarding the other target matrices will be summarized in tables.

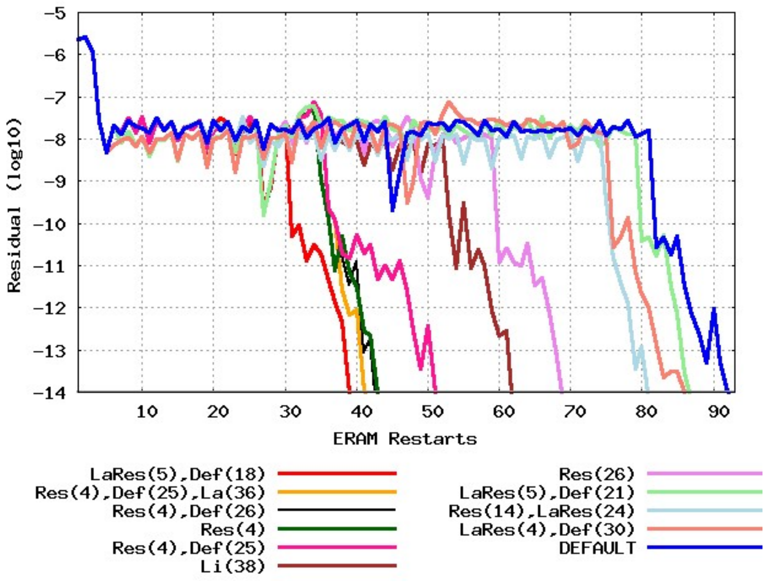

Fig. 2. Bayer04 Matrix, $m = 15, tol = 10^{-14}$. ERAMs (using initially α_{Def}) with Mixed Restarting Strategies efficiency compared to the ERAM using α_{Def} only (blue line). Results have been executed on a single Intel i5-2430M.

The Figure 2 shows the ERAMs with mixed restarting strategies for the Bayer04 matrix. As a reference, we added the ERAM using α_{Def} during its complete execution (blue line). We experimented several switches: at restart 4, 5, 16, 18, 26 and finally 38 according to our convergence algorithm. A divergence or stagnation was detected at these restarts therefore we changed the restarting strategy in order to avoid such convergence behavior. We re-executed some of the ERAM with mixed restarting strategies and combined at maximum three restarting strategies. Throughout all the following tables showing restarting strategies mixed, we indicated the restart where we changed the restarting strategy in parenthesis. As an illustration, the "Res(4),Def(25)" legend on figure 2 means we started with α_{Def} restarting strategy, then switch it by α_{Res} at restart 4 and switch it again by α_{Def} at restart 25.

The best configuration is obtained by changing α_{Def} by α_{LaRes} at restart 5 and then by α_{Def} at restart 18 (LaRes(5),Def(18) on Figure 2): we obtain the convergence in 39 restarts instead of 93 for the original convergence. All changes presented on Figure 2 show a gain from 54 to 10 restarts. Using α_{La} combined with other restarting strategies saves 50 restarts (green line, Res(4),Def(25),La(36) on Figure 2.), while α_{La} has the worst convergence on Table 3. One restarting strategy alone may not be very efficient while mixing it with the others may considerably improve the ERAM convergence.

In what follows, we will present the results for other matrices in tables. All the ERAM with mixed restarting strategies began their process using α_{Def}. For all matrices listed on Table 2, we will compare the ERAM using only α_{Def} as restarting strategy and the ERAM using mixed restarting strategies. One may note that we executed the same work with the other restarting strategies, id est ERAM using initially other restarting strategies than α_{Def}. This leads to the same conclusion that we present in this paper. As α_{Def} is the classic restarting strategy, we choose to focus on this one in this paper.

bcspwr09	α_{Li}(18,10,2)	α_{Res}(10)	α_{La}(10)
Restart CV.	245,132,225	285	300

Table 4. *Bcspwr*09 Matrix, $m = 25, tol = 10^{-15}$, ERAMs (using initially α_{Def}) with Mixed Restarting Strategy Results. Results have been executed on a single Intel i5-2430M.

On Table 4 we present the mixed restarting strategies convergence for bcspwr09 matrix. As an illustration, α_{Li}(18,10,2) refers to three distinct ERAM switching their restarting strategy to α_{Li} at respectively restart 18, 10 and 2. We indicate their restart convergence: the ERAM using α_{Li} at restart 18 converges at restart 245, the ERAM using α_{Li} at restart 10 converges at restart 132 etc... Throughout all this paper, we will retain this notation. The best scheme is obtained by switching α_{Def} by restarting strategy α_{Li} at restart 10: the ERAM with these mixed restarting strategies converges in only 132 restarts, while the initial ERAM using α_{Def} did not reached the convergence (see Table 3). Choosing the right restart is important: one may note that switching α_{Def} by restarting strategy α_{Li} at restart 18 has poor performance compared to the same switch at restart 10. Another observation is that mixing α_{Def} with α_{Res} or α_{La} has good performance compared to results presented in Table 3: α_{Res} and α_{La} restarting strategies have poor performances while mixing them may accelerate considerably the ERAM convergence and turn them to efficient restarting strategies.

bcspwr09	α_{LaRes}(8,7,5)	α_{LiRes}(8)	α_{Li}(8,6,5)	α_{La}(8,6,5)	α_{Res}(8,6,5)
Restart CV.	30,29,34	31	30,26,29	29,29,34	30,34,28

Table 5. *Bcspwr*09 Matrix, $m = 30, tol = 10^{-15}$, ERAMs (using initially α_{Def}) with Mixed Restarting Strategy Results. Results have been executed on a single Intel i5-2430M.

We ran the same tests with a bigger subspace size. The Table 3 shows that α_{LiRes} has still difficulties to converge. We denote a difference of 10 restarts only between the best restarting strategy α_{Li} and α_{La}. α_{Li} provides a gain of 70 restarts compared to α_{LiRes} restarting startegy. On Table 5, we changed the α_{Def} restarting strategy to evaluate if we can still improve the convergence, even in case of a strong ERAM convergence. We have a maximum gain of 4 restarts with mixed restarting strategies compared to the initial configuration. By mixing restarting strategies, whether we have still a small gain, whether we converge in a comparable number of restarts.

ex11	α_{Li}(8,2)	α_{LaRes}(13,8,7,4,2)	α_{La}(13,5)	α_{Res}(8)	α_{Li}(1),α_{LaRes}(4)
Restart CV.	139,97	84,40,49,39,29	154,104	122	142

Table 6. *Ex*11 Matrix, $m = 20, tol = 10^{-15}$, ERAMs (using initially α_{Def}) with Mixed Restarting Strategy Results. Results have been executed on a single Intel i5-2430M.

On Table 3, the ERAM using α_{LaRes} converges in only 10 restarts, while α_{Def} converges at restart 148. We have a gain of 138 restarts, which is clearly not negligible in terms of execution time. By mixing the restarting strategies (Table 6), we obtain in the best configuration a convergence in 29 restarts. This offers a gain of 119 restarts compared to the original α_{Def} ERAM convergence. Switching α_{Def} by α_{La} (see Table 6) provides comparable results as the original restarting strategy α_{Def}. The best restarting strategy we could find is α_{Li} initially and then α_{LaRes} at restart 3. This configuration converges in 6 restarts only, which is even better than α_{LaRes} on Table 3. This means that we can even accelerate the best ERAM convergence by mixing the restarting strategies alltogether. For this matrix, mixing α_{LaRes} and α_{Def} restarting strategies provide very satisfiable results in terms of number of restarts to reach the convergence.

Fission	α_{Res}(8,7,2)	α_{Li}(7,2)
Restart CV.	46,46,56	32,32

Table 7. *Fission* Matrix, $m = 10, tol = 10^{-13}$, ERAMs with Mixed Restarting Strategy Results. Results have been executed on 400 Intel Sandy Bridge from the PRACE Curie supercomputer.

On Table 3, the ERAM using α_{Def} as a single restarting strategy converges in 46 restarts, which is the best configuration for each ERAM using a single restarting strategy. In Table 7, mixing α_{Def} with α_{Li} converges in 32 restarts only, which is a considerable gain. On Table 3, one may observe that the ERAM using α_{Li} restarting strategy alone has a poor convergence. Mixing α_{Li} with α_{Def} had turned it to a very efficient restarting strategy. Finally, mixing α_{Def} with α_{Res} at restart 2 considerably improve the ERAM using single restarting strategy α_{Res} presented in Table 3: the original configuration did not reached convergence while this new one could.

Rim_dense	α_{LaRes}(25,4)	α_{Li}(26,25)	α_{LiRes}(25)	α_{La}(25)	α_{Res}(26,4)
Restart CV.	94,59	92,72	115	129	137,82

Table 8. *Rim_dense* Matrix, $m = 15, tol = 10^{-13}$, ERAMs with Mixed Restarting Strategy Results. Results have been executed on 480 Intel Sandy Bridge from the PRACE Curie supercomputer.

On Table 3, the ERAM using α_{Def} converges at the 152 restart, which is more than two times worst than α_{Res} (which converges at 65 restart). The best mixed restarting strategy mixed is obtained by switch α_{Def} by α_{LaRes} at restart 4 ($\alpha_{LaRes}(4)$ on Table). Firstly we improve considerably the initial ERAM configuration, secondly, this configuration is even better than the best ERAM using a single restarting strategy (Table 3, ERAM using α_{Res}).

5 Conclusion on the ERAM Restarting Strategies

We presented in section 3 diverse ERAM configurations using several restarting strategies. Their efficiency clearly depends on the matrix and the ERAM convergence itself, as these restarting strategies are using residuals or computed eigenvalues. There are no tools to fix the ERAM parameters to ensure the convergence or have the optimal ERAM convergence. There is a need to change the ERAM parameters during the runtime execution. This first necessitates to study and affect to the ERAM convergence a status. Does the ERAM converges, diverges or stagnates? Such status must be detected earliest as possible so we can improve the ERAM convergence and save execution time. Based on the ERAM convergence study, we know at which restart we must change the ERAM restarting strategy. On the presented figures, we changed the restarting strategy when the ERAM using α_{Def} tends to stagnate or diverge. After an analysis of each ERAM restarting strategy, we mixed the restarting strategies and improved the efficiency of the original ERAM, used as a reference to compare our results. Mixing the ERAM restarting strategies has a great improvement on the ERAM convergence as we can find better convergence with mixed restarting strategies compared to the best ERAM using a single restarting strategy.

In this paper, we presented the changes for the ERAM using α_{Def} initially, but the same work has been done for every restarting strategies id est we ran the ERAMs using respectively α_{Res}, α_{La}, α_{LaRes}, α_{Li} and α_{LiRes} to begin their process, showing promising results, as each restarting strategy can be ameliorated. We ran the same tests on other matrices, leading to the same conclusion, meaning that we can considerably improve the ERAM convergence with respect to its initial parameters. The next step is to automate the restarting strategies mix. Our results show promising accelerations for the ERAM convergence with mixed restarting strategies, but we will automate this process to explore more restarting strategies mix based on the exisiting results. We are now focused on the restarting strategy choice: which restarting strategy should we choose when a stagnation or divergence is detected and how to optmize the restarting strategy mix? The presented work constitues the basis of the ongoing algorithm to tune the restarting strategies and ensure the ERAM convergence as fast as possible in terms of number of restarts and parallel execution time. The most important point for this amelioration is the abscence of additionnal operations nor communications. Computing the restarting vector $v_0^{(i+1)}$ is a parallel operation that requires vectors additions and especially scal operations to weight them with the restarting coefficients. The restarting coefficients are using already computed data, we only focus on reusing computed data to improve the ERAM convergence at no cost in terms of memory, number of operations and most of all, communications. This offers a very good ratio of the ERAM numerical convergence versus parallel time execution, as the first is ameliorated while the second is unchanged.

There is however a necessity to mix this restarting strategy tuning with the existing subspace-size tuning for the Krylov solver applied to linear system resolution [6], [7], especially in the cases when ERAM do not reach convergence whatever the restarting strategy is. We aim to improve the ERAM convergence using as a priority the restarting strategy tuning and if it is inefficient, we increase the subspace size. With such restarting strategies and subspace size tuning, we aim to ensure the ERAM convergence as fast as possible with a low parallel communication scheme whathever the ERAM initial parameters are.

References

1. G. G. Davidson and al.: Massively Parallel, Three-Dimensional Transport Solutions for the k-Eigenvalue Problem. NSE, (2013), vol(75), 283–291
2. T. M. Evans: Full Core Reactor Analysis: Running Denovo on Jaguar. PHYSOR 2012, (2012)
3. J. Dufex and W. Gudowski: The semi-source fission matrix method for accelerating the monte-carlo eigenvalue calculations. AlbaNova University Center, Stockholm, Sweden, (2007)
4. F. Châtelin: Valeurs Propres de Matrices. (1988), Masson
5. Y. Saad: Numerical Solution of Large Non-symmetric Eigenvalue Problems. Comp. Phy. Comm. (1989)
6. A. Baker, E. R. Jessup and T.V. Kolev: A Simple Strategy for Varying the Restart Parameter in GM-RES(m). Journal of Comp. and App. Math., (2009), 751–761
7. T. Katagiri, P-Y. Aquilanti and S. Petiton: A Smart-Tuning Strategy for Restart Frequency of GM-RES(m) with Hierarchical Cache Sizes. High Perf. Comp. for Comp. Scie. VECPAR (2012) 314–328
8. J. Dongara and all: Applied Mathematics Research for Exascale Computing. March (2014), U.S. Department of Energy
9. J. Dubois, C. Calvin and S. Petiton: Performance and numerical accuracy evaluation of heterogeneous multicore systems for Krylov orthogonal basis computation. High Perf. Comp. for Comp. Scie. VEC-PAR(2010) **LNCS 6449** (2010) 45–57