Interpolation-restart strategies for resilient eigensolvers

Emmanuel Agullo, Luc Giraud, Pablo Salas, Mawussi Zounon

Interpolation-restart strategies for resilient eigensolvers

E. Agullo, L. Giraud, P. Salas, M. Zounon

RESEARCH REPORT
N° 8625
June 2015
Project-Teams HiePACS

ISSN 0249-6399 ISRN INRIA/RR--8625--FR+ENG

Interpolation-restart strategies for resilient eigensolvers

E. Agullo[*], L. Giraud[*], P. Salas[†], M. Zounon[*]

Project-Teams HiePACS

Research Report n° 8625 — June 2015 — 26 pages

[*] Inria Bordeaux Sud-Ouest, 200 avenue de la Vieille Tour, F-33400 Talence, France
[†] Sherbrooke University, Canada

RESEARCH CENTRE
BORDEAUX – SUD-OUEST

200 avenue de la Vielle Tour
33405 Talence Cedex

Abstract: The solution of large eigenproblems is involved in many scientific and engineering applications when, for instance stability analysis is a concern. For large simulation in material physics or thermo-acoustics, the calculation can last for many hours on large parallel platforms. On future large-scale systems, the time interval between two consecutive faults is forecast to decrease so that many faults could occur during the solution of large eigenproblems. Consequently it becomes critical to design parallel eigensolvers which can survive faults. In that framework, we mainly investigate the relevance of approaches relying on numerical techniques that might be combined with more classical techniques for real large scale parallel implementations. Because we focus on numerical remedies we do not consider parallel implementations nor parallel experiments but only numerical experiments. We assume that a separate mechanism ensures the fault detection and that a system layer provides support for setting back the environment (processes, . . .) in a running state. Once the system is in a running state, after a fault, our main objective is to provide robust resilient schemes so that the eigensolver may keep converging through the fault without restarting the calculation from scratch. For this purpose, we extend the interpolation-restart (IR) strategies introduced in a previous work [2] for linear systems. For a given numerical scheme, the IR strategies consist in extracting relevant spectral information from available data after a fault. After data extraction, a well selected part of the missing data is regenerated through interpolation strategies to constitute meaningful input to restart the numerical algorithm. A main feature of this numerical remedy that it does not require extra resources, *i.e.*, computational unit or computing time, when no fault occurs. In this paper, we revisit a few state-of-the-art methods for solving large sparse eigenvalue problems namely the Arnoldi methods, subspace iteration methods and the Jacobi-Davidson method, in the light of our IR strategies. For each considered eigensolver, we adapt the IR strategies to regenerate as much spectral information as possible. Through intensive numerical experiments, we illustrate the qualitative behavior of the resulting schemes when the number of faults and the amount of lost data are varied.

Key-words: resilience; fault tolerance; eigenvalue problems; linear algebra; HPC; numerical methods; Arnoldi; IRAM; subspace iteration; Jacobi-Davidson.

Stratégies d'interpolation-restart pour des solveurs aux valeurs propres résilients

Résumé : Le calcul des paires propres (valeurs propres et vecteurs propres) de matrices creuses de grande taille intervient dans de nombreuses applications scientifiques et d'ingénierie, par exemple dans les problèmes d'analyse de stabilité. Pour des simulations de grande dimension en physique des matériaux ou en thermo-acoustique, ces calculs peuvent durer plusieurs heures même sur des supercalculateurs. Pour les futurs systèmes de calcul haute performance à grande échelle, il est attendu que l'intervalle de temps entre deux pannes consécutives diminue de sorte que plusieurs pannes pouraient intervenir au cours d'une même résolution. Il est donc nécessaire de concevoir des solveurs parallèles de problèmes aux valeurs propres qui survivent aux pannes. Dans ce contexte, nous considérons essentiellement des solutions numériques pour en évaluer la pertinence, celles-ci pourraient se combiner avec des techniques de résilience plus classiques pour des mises en œuvre efficaces qui passent à l'échelle. Afin de se concentrer sur des solutions numériques, nous faisons l'hypothèse qu'un mécanisme distinct assure la détection des pannes et qu'une couche système s'occupe de ramener l'environnement (processus, . . .) dans un état fonctionnel. Une fois que le système est de nouveau opérationnel, après une panne, notre principal objectif est d'étudier des stratégies robustes de résilience qui permettent de converger sans avoir à redémarrer complètement le calcul. À cette fin, nous étendons les techniques Interpolation-Restart (IR) introduites dans un précédent travail [2] pour les systèmes linéaires. Les stratégies IR consistent à regéner des valeurs pour les données perdues en utilisant les données qui sont disponibles; l'ensemble de ces données est utilisé pour redemarrer efficacement les différents solveurs. Un atout de ces approches est qu'elles ne nécessitent pas de ressources supplémentaires (calculatoire ou mémoire) en l'absence de pannes. Dans ce travail, à la lumière des stratégies d'interpolation-restart, nous avons revisité des méthodes itératives de l'état de l'art à savoir les méthodes Arnoldi, les méthodes d'itérations de sous-espace et la méthode de Jacobi-Davidson. Pour chacune de ces méthodes, nous avons adapté les stratégies d'interpolation-restart pour régénérer autant d'information spectrale que possible. Nous illustrons, grâce à de nombreuses expériences, le comportement numérique qualitatif de nos stratégies lorsque le nombre de pannes et la quantité de données perdues varient.

Mots-clés : résilience; tolérance aux pannes; problèmes aux valeurs propres; algèbre linéaire; HPC; méthodes numériques; Arnoldi; IRAM; itération de sous-espace; Jacobi-Davidson.

1 Introduction

The computation of eigenpairs (eigenvalues and eigenvectors) of large sparse matrices is involved in many scientific and engineering applications such as when stability analysis is a concern. To name a few, it appears in structural dynamics, thermodynamics, thermo-acoustics, quantum chemistry. With the permanent increase of the computational power of high performance computing (HPC) systems by using a larger and larger number of CPU cores or specialized processing units, HPC applications are increasingly prone to faults. To cope with these unstable situations, numerical simulations need to be equipped with resilient mechanisms that enable them to complete a calculation even though several faults might occur during its realization. We investigate in this paper numerical remedies in the framework of the solution of eigenproblems. We consider the standard eigenproblem of the form:

$$Au = \lambda u,$$

where $A \in \mathbb{C}^{n\times n}$, with $u \neq 0$, $u \in \mathbb{C}^n$ and $\lambda \in \mathbb{C}$. The couple (λ, u) is called an eigenpair of A where the vector u is an eigenvector with the associated eigenvalue λ.

In this paper, we extend the interpolation-restart (IR) strategies introduced for the solution of linear systems [1, 2] to a few state-of-the-art eigensolvers. More precisely, the Arnoldi [3], Implicitly restarted Arnoldi [9], subspace iteration [11], and Jacobi-Davidson algorithms have been revisited to make them resilient in the presence of faults. Most of the considered eigensolvers naturally implement a restart mechanism to cope with memory constraints. We incorporate an additional restart in our scheme after faults. We attempt to regenerate as much as possible relevant spectral information to perform the restart effectively and efficiently.

The remainder of the paper is structured as follows: in Section 2 we describe how the interpolation techniques can be extended to regenerate meaningful spectral information. We briefly present the eigensolvers that we have considered in Section 3 and detail how the recovery ideas can be tuned for each one of them. Section 4 is devoted to the numerical experiments; we discuss the robustness of the various resilient numerical schemes and conclude with some perspectives in Section 5.

2 Interpolation-restart principles

In this section, we describe how the interpolation strategies can be used to regenerate meaningful spectral information. Contrarily to what has been proposed for the Krylov linear solvers where only a meaningful iterate is computed to serve as a new initial guess for restarting the iterations [1, 2], more flexibility exists in the framework of eigensolution where similar ideas can be adapted to better exploit the numerical features of the individual eigensolvers. The main reasons are that some of the considered eigensolvers do not rely on a central equality or a sophisticated short term recurrence (such as Conjugate Gradient for linear system solution). Furthermore, we consider also situations where a few *nev* eigenpairs are sought, which also provides additional freedom. We present in details different variants for selecting and computing the relevant subspaces to perform the restart for each particular considered eigensolver in Section 3.

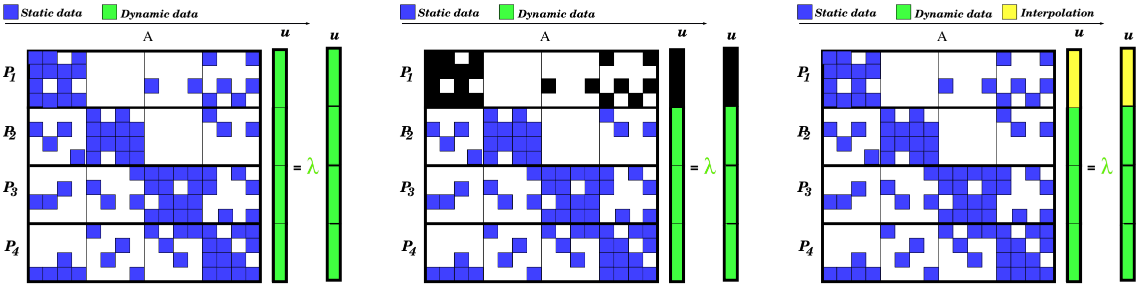
2.1 Context

We introduce the governing ideas that underline the design of the IR strategies. For the sake of exposure, we restrict ourselves to parallel distributed environments, although these strategies can be extended to other HPC contexts, assuming that the faults can be detected and reported by a lower layer of the operating system stack [7].

Assumption 1. *In our parallel computational context, all the vectors or matrices of dimension n are distributed by blocks of rows in the memory of the different computing nodes but scalars or low dimensional matrices are replicated.*

According to Assumption 1, the eigenvector u and the matrix A are distributed according to a block-row partition as well as all vectors of dimension n generated during the solution whereas scalars (for example λ) or low dimensional matrices are replicated on all nodes. Let N be the number of partitions, such that each block-row is mapped to a computing node. For all p, $p \in [1, N]$, I_p denotes the set of row indices mapped to node p. With respect to this notation, node p stores the block-row $A_{I_p,:}$ and u_{I_p} as well as the entries of all the vectors involved in the solver associated with the corresponding row indices of this block-row. If the block A_{I_p,I_q} contains at least one non zero entry, node p is referred to as neighbor of node q as communication will occur between those two nodes to perform a parallel sparse matrix-vector product.

When a fault occurs on a node, all data in its memory are lost. We consider the formalism proposed in [8] where lost data are classified into three categories: the *computational environment*, the *static* data and the *dynamic* data. The computational environment is all the data needed to perform the computation (code of the program, environment variables, . . .). The static data are those that are set up during the initialization phase and that remain unchanged during the computation such as the coefficient matrix A. The dynamic data is all data whose value may change during the computation. The iterates λ and u are examples of dynamic data. In Figure 1a, we depict a block-row distribution on four nodes. The data in blue is the static data associated with the eigenproblem (*i.e.*, the matrix) while the data in green is the dynamic data (here, only the eigenpair (λ, u) is shown). If node P_1 fails, the first block-row of A as well as the first entries of u are lost (in black in Figure 1b). The eigenvalue λ is a scalar replicated on all nodes and thus remains available on non failed ones.



(a) Before fault

(b) Faulty iteration

(c) Interpolation

Figure 1: General interpolation scheme. The matrix is initially distributed with a block row partition, here on four nodes (a). When a fault occurs on node P_1, the corresponding data is lost (b). Whereas static data can be immediately restored, dynamic data that has been lost cannot and we investigate numerical strategies for interpolating it (c).

We assume that, when a fault occurs, the failed node is replaced and the associated computational environment and static data are restored. In Figure 1c for instance, the first matrix block-row is restored as it is a static data. However, the eigenvector iterate u, being a dynamic data, its entries u_{I_1} are definitely lost and we present strategies for regenerating them through some interpolations that follow. Our strategies do not attempt to interpolate all the dynamic data but for each eigensolver, we study which part of the dynamic data could be interpolated. Because not all data is regenerated and/or that the interpolation computed approximation of the lost data, part of the information remains lost. As a consequence, it is furthermore required to restart the numerical process. We will present in detail propositions for selecting (or producing) relevant data for performing that restart for each particular eigensolver considered in Section 3. We assume in the rest of this paper that a fault occurs during iteration $f + 1$ and the proposed interpolation strategies are thus based on the numerical values computed by the algorithms at iteration f.

2.2 Interpolation methods

The IR strategies consist in interpolating lost data by using non corrupted data. Let $u^{(f)}$ be an approximated eigenvector when a fault occurs. After the fault, the entries of $u^{(f)}$ are correct, except those stored on the failed node p. Assuming that in a parallel distributed environment, the current eigenvalue λ_f is naturally replicated in the memory of the different computing nodes, we present two strategies to compute a new approximate solution. The first strategy, referred to as linear interpolation and denoted LI, consists in solving a local linear system associated with the submatrices A_{I_p,I_p} of the failed node. The second one relies on the solution of a least squares interpolation and is denoted LSI. Those two alternatives result from considering $(\lambda_f, u^{(f)})$ as an exact eigenpair. We may have a block-row view point, which defines the LI variant. If node p fails, LI computes a new approximation of the eigenvector $u^{(LI)}$ as follows

$$\begin{cases} u^{(LI)}_{I_q} = u^{(f)}_{I_q} & \text{for } q \neq p, \\ u^{(LI)}_{I_p} = (A_{I_p,I_p} - \lambda \mathcal{I}_{I_p,I_p})^{-1}(-\sum\limits_{q\neq q} A_{I_p,I_q} u^{(f)}_{I_q}). \end{cases}$$

Alternatively, we can have a block-column point of view, which leads to the LSI variant that computes $u^{(LSI)}$ via

$$\begin{cases} u^{(LSI)}_{I_q} = u^{(f)}_{I_q} & \text{for } q \neq p, \\ u^{(LSI)}_{I_p} = \underset{u_{I_p}}{\operatorname{argmin}} \|(A_{:,I_p} - \lambda \mathcal{I}_{:,I_p}) u_{I_p} - \sum\limits_{q\neq p}(A_{:,I_q} - \lambda \mathcal{I}_{:,I_q}) u^{(f)}_{I_q})\|. \end{cases}$$

Here, $\mathcal{I} \in \mathbb{C}^{n\times n}$ is the identity matrix and we furthermore assume that $(A_{I_p,I_p} - \lambda I_{I_p,I_p})$ is non singular and that $(A_{:,I_p} - \lambda \mathcal{I}_{:,I_p})$ has full rank. The matrix involved in the least squares problem, $(A_{:,I_p} - \lambda \mathcal{I}_{:,I_p})$, is sparse of dimensions $|J_p| \times |I_p|$ where its number of rows $|J_p|$ depends on the sparsity structure of $A_{:,I_p}$. Consequently the LSI strategy may have a higher computational cost. In the rest of this paper, we use IR indifferently to denote LI and/or LSI.

3 Interpolation-restart strategies for general purpose eigensolvers

In this section, we briefly describe a few of the most popular eigensolvers for general sparse matrices that we have considered in this work to compute *nev* eigenpairs. For each eigensolver, we describe which data are regenerated after a fault to make them resilient. We also briefly present the numerical approach and associated algorithm. We then describe how the IR strategies can be applied to compute a few of the key numerical quantities of the solver when some data is lost as well as how this data can be used to perform an effective and efficient restart.

3.1 Subspace iterations to compute *nev* eigenpairs

Brief description of the numerical algorithm: The subspace iteration method is a block variant of the power method [6]. It starts with an initial block of $nev + s$ (with $s \geq 0$) linearly independent vectors corresponding to matrix $U^{(0)} = [u^{(0)}_1, \ldots, u^{(0)}_{nev+s}] \in \mathbb{C}^{n\times(nev+s)}$. Under certain assumptions [11], the sequence of $U^{(k)} = A^k U^{(0)}$ generated by the algorithm, converges to the $nev + s$ eigenpairs of A associated with the eigenvalues of the largest magnitude. To guarantee the full column rank in $U^{(k)}$ for large values of k, the Q factor of its QR factorization may be used at each iteration.

Algorithm 1 Basic subspace iteration

```
1: Choose $U^{(0)} = [u_1^{(0)}, ..., u_{nev+s}^{(0)}] \in \mathbb{C}^{n \times (nev+s)}$
2: for $k = 0 \ldots$ until convergence do
3:     Orthonormalize $U^{(k)}$
4:     Compute $W^{(k)} = AU^{(k)}$
5:     Form the Rayleigh quotient $C^{(k)} = {W^{(k)}}^H A W^{(k)}$
6:     Compute the eigenvectors $G^{(k)} = [g_1, \ldots, g_{nev+s}]$ of $C^{(k)}$
       and eigenvalues $\sigma(C^{(k)}) = (\lambda_1, \ldots, \lambda_{nev+s})$
7:     Update Ritz vectors : $U^{(k+1)} = W^{(k)} G^{(k)}$
8: end for
```

To compute the eigenpairs associated with the smallest eigenvalues in magnitude, or eigenpairs associated with a given set of eigenvalues in a given region of the complex plane, the basic subspace iteration depicted in Algorithm 1 is no longer appropriate. For example, to compute eigenpairs associated the eigenvalues nearest to $\sigma \in \mathbb{C}$, it is possible to combine the subspace iterations with the shift-invert technique [5]. With shift-invert spectral transformation, the subspace iteration method will be applied to the matrix $(A - \sigma I)^{-1}$. Alternatively, a polynomial acceleration [10], can be used as a preconditioning technique to approximate eigenvalues near σ. This polynomial acceleration consists in applying a given polynomial $\mathcal{P}$ to the matrix A. In Algorithm 1 line 4 would change and become $W^{(k)} = \mathcal{P}(A)U^{(k)}$. The polynomial should act as a filter to damp eigencomponents in some undesired part of the spectrum.

Interpolation-restart policy: In the subspace iteration method depicted in Algorithms 1, according to Assumption 1, the Ritz vectors $U^{(k)}$ are distributed, whereas the Rayleigh quotient $C^{(k)}$ and Ritz values are replicated. When a fault occurs, we distinguish two cases. During an iteration, a fault may occur before or after the computation of the Rayleigh quotient $C^{(f+1)}$.

1. When a fault occurs before the computation of the Rayleigh quotient $C^{(f+1)}$ (Algorithm 1, lines 2 to 5) the surviving nodes cannot compute the Rayleigh quotient $C^{(f+1)}$ because entries of $W^{(f+1)}$ are missing. In this case, we consider the available entries of the Ritz vectors $U^{(f)}$ and its corresponding eigenvalues $\sigma(C^{(f)})$. We interpolate the m Ritz vectors individually $(u_\ell^{(f)}, 1 \leq \ell \leq nev + s)$ using *LI* or *LSI*. In the particular case of the computation of polynomial acceleration, all the computation in the filtering step are lost.

2. When a fault occurs after the computation of the Rayleigh quotient $C^{(f+1)}$ (Algorithm 1, line 6 to 7) all surviving nodes can compute the entries of $U^{(f+1)}$ relying on a local replicate of $C^{(f+1)}$ and the local entries of $W^{(f+1)}$. The missing entries of each Ritz vector $(u_\ell^{(f+1)}, 1 \leq \ell \leq nev + s)$ can be individually interpolated using *LI* or *LSI* relying on the corresponding eigenvalues $\sigma(C^{(f+1)})$.

After the interpolation, the subspace iteration algorithm is restarted with the matrix $U^{(IR)} = [u_1^{(IR)}, \ldots, u_{nev+s}^{(IR)}] \in \mathbb{C}^{n \times (nev+s)}$ until convergence or the next fault.

3.2 Arnoldi method to compute one eigenpair

Brief description of the numerical algorithm: The Arnoldi method is an efficient procedure for approximating eigenvalues lying in the periphery of the spectrum of A. For a prescribed dimension m, referred to as the restart, the method starts from an initial guest vector v_1 of norm one and builds an orthonormal basis V_m of the search space spanned by $span\{v_1, Av_1, ..., A^{m-1}v_1\}$ as well as an upper Hessenberg matrix $H_m \in \mathbb{C}^{m \times m}$. Those matrices satisfy the so called Arnoldi equality:

$$AV_m = V_m H_m + \beta_m v_{m+1} e_m^T, \tag{1}$$

where e_m denotes the last column of the $m \times m$ identity matrix. By construction, H_m is the Rayleigh quotient matrix associated with V_m that is used to compute approximate eigenpairs of A. If the computed Ritz pair corresponding to the targeted eigenvalue is not accurate enough, the procedure is restarted using the best current Ritz vector as the new initial guess. This space expansion and restart sequence is repeated until convergence.

Interpolation-restart policy: According to Assumption 1, we assume that the low dimension Hessenberg matrix H_k is replicated on each node. Consequently, regardless of the step in which the fault occurs during the iteration, each surviving node q can solve the eigenproblem $H_f g = \lambda g$ redundantly, then compute its entries of the Ritz vector $u_{I_q} = V_f(I_p, :)g$. The next step is the computation of the Ritz vector entries allocated on the failed node using LI or LSI. The resulting vector becomes the new initial guess to restart the Arnoldi iterations.

3.3 Implicitly restarted Arnoldi method to compute nev eigenpairs

Brief description of the numerical algorithm: Developed by Lehoucq and Sorensen in [9], the implicitly restarted Arnoldi method (IRAM) depicted in Algorithm 2 is commonly used for the solution of large eigenvalue problems. IRAM is an extension of the Arnoldi method that starts with an Arnoldi equality of size m. From the Arnoldi equality of dimension m, IRAM performs a contraction of the equality from size m down to size $\tilde{m}$ ($nev \leq \tilde{m} \leq m$). This is achieved by applying a polynomial filter of degree $\ell = m - \tilde{m}$ that reduces the size of the Arnoldi equality down to $\tilde{m}$ (see Algorithm 2, line 12), that is then expanded again to size m before checking the accuracy of the Ritz eigenpairs. The expansion and contraction steps are repeated until convergence. The contraction step acts as a filter to focus the spectral information in a target region of the spectrum from the m dimensional Krylov subspace while maintaining the Arnoldi equality.

Algorithm 2 Implicitly restarted Arnoldi method with restart m

```
 1: Compute Arnoldi equality $AV_m = V_m H_m + f_m e_m^T$.
 2: for $k = 0, 1, \ldots,$ until convergence, do
 3:     Compute $\sigma(H_m)$ and select $\ell$ shifts $\mu_1, \ldots, \mu_\ell$ ($\ell = m - \tilde{m}$).
 4:     $Q = \mathcal{I}_m$
 5:     for $i = 1, \ldots, \ell$ do
 6:         QR Factorize $Q_i R_i = H_m - \mu_i I$
 7:         $H_m = {Q_i}^H H_m Q_i$
 8:         $Q = Q Q_i$
 9:     end for
10:     $\beta_{\tilde{m}} = H_m(\tilde{m} + 1, \tilde{m})$
11:     $f_{\tilde{m}} = v_{\tilde{m}+1} \beta_{\tilde{m}} + f_m Q(m, \tilde{m})$
12:     $V_{\tilde{m}} = V_m Q(:, 1 : \tilde{m})$; $H_{\tilde{m}} = H_m(1 : \tilde{m}, 1 : \tilde{m})$
13:     Starting with $AV_{\tilde{m}} = V_{\tilde{m}} H_{\tilde{m}} + f_{\tilde{m}} e_{\tilde{m}}^T$,
        perform $\ell$ steps of Arnoldi algorithm to get $AV_m = V_m H_m + f_m e_m^T$
14: end for
```

Interpolation-restart policy: When a fault occurs during iteration $f + 1$, it may be during the expansion of the Krylov subspace (Algorithm 2, line 13) or during the contraction step (Algorithm 2, line 3 to 12).

1. The contraction implicitly implements the shifted QR mechanism. When a fault occurs during the contraction step each surviving node q can compute this step concurrently and redundantly using the replicated Hessenberg matrix H_m. The outcome of this local calculation is that all surviving nodes have $V_{\tilde{m}}(I_q, :)$ as well as the reduced Hessenberg matrix $H_{\tilde{m}}$.

2. When the fault occurs in the expansion phase during step $f+1$ with $\tilde{m} < f+1 < m$, each surviving node can compute the implicitly shift QR update of H_f to compute $H_{\tilde{m}}$ using the shift defined by $\sigma(H_f)$. Using the result of the shift QR calculations, each node q can also compute $V_{\tilde{m}}(I_q, :)$.

From $V_{\tilde{m}}(I_q, :)$ and $H_{\tilde{m}}$, the surviving nodes may then compute eigenvectors $G = [g_1, \ldots, g_{nev}]$ and eigenvalues $(\lambda_1, \ldots, \lambda_{nev})$ of $H_{\tilde{m}}$. Consequently, the entries of the Ritz vectors are computed by

$$U^{(f)}(I_q, :) = V_{\tilde{m}}(I_q, :)G. \tag{2}$$

The missing entries I_p of Ritz vectors may be interpolated using either *LI* or *LSI* of the interpolation algorithms. In that situation, the available entries of $V_{\tilde{m}}$ no longer satisfy any Arnoldi equality. To take into account all the available spectral information, we compute the linear combination of the interpolated eigenvectors, $u = \sum_{j=1}^{\tilde{m}} u_j^{(IR)}$, and restart with the normalized linear combination $v_1 = \frac{u}{\|u\|}$ as initial vector. The motivation is that Arnoldi converges within the first expansion if started from a vector v that is a linear combination of $k \leq m$ eigenvectors [6].

3.4 The Jacobi-Davidson method to compute *nev* eigenpairs

Brief description of the numerical algorithm: The Jacobi–Davidson method, proposed by Sleijpen and van der Vorst in [14], is a widely used eigensolver, especially for eigenpairs in the interior of the spectrum. The basic ingredient of Jacobi-Davidson is depicted in Algorithm 3 for the computation of one eigenpair whose eigenvalue is close to a given target τ. It starts with a given normalized vector v and constructs a basis V extended using the Jacobi orthogonal correction method. At each iteration the algorithm computes the Ritz pairs associated with V and selects the eigenpair whose eigenvalue is the closest to the target τ.

Algorithm 3 $[\lambda, u]$ = Basic-Jacobi-Davidson(v, τ)
Jacobi–Davidson algorithm to compute the eigenvalue of A closest to a target value τ

```
Set $V_1 = [v]$
for $k = 1, 2, \ldots$,until convergence do
    Compute Rayleigh quotient: $C_k = V_k^H A V_k$, and eigenpairs of $C_k$
    Select Ritz pair $(\lambda_k, u_k)$ such that $\lambda_k$ is the closest to $\tau$
    $r_k = Au_k - \lambda_k u_k$
    Perform a few steps of GMRES to solve the correction equation
        $(I - u_k u_k^H)(A - \tau I)(I - u_k u_k^H)v = -r_k$, so that $v \perp u_k$
    Compute $w$ by orthonormalizing $v$ against $V_k$: $w = v - V_k(V_k^H v)$
    Set $V_{k+1} = [V_k, w]$
end for
```

Algorithm 3 converges to one eigenpair. If more than one eigenpair needs to be computed, Algorithm 3 can be accommodated to compute a partial Schur decomposition of A. In that respect, the next iterations are enforced to generate a search space orthogonal to the space spanned by the *nconv* already converged eigenvectors. This is achieved by representing this space using the corresponding Schur vectors. Let $AZ_{nconv} = Z_{nconv}T_{nconv}$ denote the partial Schur form where the columns of the orthonormal matrix Z_{nconv} span the converged eigenspace and the diagonal of the upper triangular matrix T_{nconv} are the associated converged eigenvalues.

Algorithm 4 corresponds to the Jacobi-Davidson style QR algorithm presented in [4]. It is conceived to be used by a higher level routine that decides the number of wanted eigenpairs *nev*, the target point τ, the maximum and the minimum size of the basis V, etc. The inputs for the algorithm are the existing converged Schur vectors Z_{nconv} of A, the current size k of the basis

V_k, $W_k = AV_k$, and the Rayleigh quotient C_k. The focal point τ, the maximum dimension m affordable for the space spanned by V, the size of the restarted basis $\tilde{m}$ $(1 \leq \tilde{m} < m)$ and the maximum number of restarts allowed are also provided. Outputs are μ, z and t, such that

$$A(Z_{nconv}\ z) = (Z_{nconv}\ z)\begin{pmatrix} T_{nconv} & t \\ 0 & \mu \end{pmatrix}$$

is a partial Schur decomposition of one higher dimension.

Algorithm 4 $[\mu, z, t]$=JDQR(Z(:,1:$nconv$),V(:,1:k),W(:,1:k),C(1:k,1:k), τ,$\tilde{m}$,m,$maxiter$)
Jacobi-Davidson style QR algorithm for expansion of partial Schur decomposition

```
Set $iter = 0$; $k_{init} = k$; $tr = 0$
while iter<maxiter do
  $iter = iter + 1$
  for $k = k_{init}, \dots, m$ do
    % Computation of the Schur decomposition $CQ = QT$
    % so that the eigenvalues on the diagonal of $T$
    % are sorted by increasing distance to $\tau$
    $[Q, T] = SortSchur(C(1:k, 1:k), \tau, k)$,
    Choose $\mu = T(1,1)$ and $g = Q(:,1)$, the Ritz pair closest to $\tau$
    Approximate eigenvector of $A$: $z = V(:, 1:k)g$, and $Az$: $y = W(:, 1:k)g$
    Compute the residual $r = y - \mu z$, orthogonalize it against $Z$(:,1:$nconv$) and compute its
    norm: $rnorm$=norm(r)
    % Convergence test:
    if $rnorm$ is small enough then
      $nconv = nconv + 1$
      % Prepare outputs and deflate:
      $t = Z^H y$; $V = V(:, 1:k)Q(:, 2:k)$;
      $W = W(:, 1:k)Q(:, 2:k); C = T(2:k, 2:k)$.
      return
    else if  $k = m$ then
      % Restart:
      $V(:, 1:\tilde{m}) = V(:, 1:m)Q(:, 1:\tilde{m})$;
      $W(:, 1:\tilde{m}) = W(:, 1:m)Q(:, 1:\tilde{m})$;
      $C(1:\tilde{m}, 1:\tilde{m}) = T(1:\tilde{m}, 1:\tilde{m})$;
      $k_{init} = \tilde{m}$
    end if
    %No convergence reached and $k < m$.
    Solve the correction equation:
        $(I - zz^H)(A_{\perp} - \tau I)(I - zz^H)v = -r$
    Orthogonalize $v$ against $V(:, 1:k)$ and $Z(:, 1:nconv)$
    % Extend the Rayleigh basis and the Rayleigh quotient:
    $V(:, k+1) = v$, $W(:, k+1) = Av$, $C(k+1$,1:$k)$=$v^H W$(:,1:$k$),
    $C$(1:$k$,$k+1$)=$V$(:,1:$k$)$^H W$(:,$k+1$), $C(k+1$,$k+1)$=$v^H W$(:,$k+1$)
  end for
end while
```

The higher level routine must furnish the necessary inputs to Algorithm 4. If the process starts from the beginning, there are then two situations. The first one corresponds to the case when the computation starts from a single random vector. Then the higher level routine computes an

Arnoldi decomposition of size $\tilde{m}$

$$AV_{\tilde{m}} = V_{\tilde{m}} H_{\tilde{m}} + \beta v_{\tilde{m}+1} e_{\tilde{m}}^T,$$

and Jacobi-Davidson starts with $U = [\ ]$, $V = V_{\tilde{m}}$, $W = AV$ and $C = H_{\tilde{m}}$. The second case is when the process starts from a given number k of initial vectors. The initial block of vectors is then orthonormalized to obtain V_k and the process can start as indicated previously, with $Z = [\ ]$, $V = V_k$, $W = AV_k$ and $C = V_k^H AV_k = V_k^H W$.

Once a partial Schur form of size nev is available, the eigenpairs (λ_ℓ, u_ℓ) (with $\ell = 1, \ldots, nev$) of A can be computed. The eigenvalue λ_ℓ is the Ritz value of T_{nev} associated with the Ritz eigenvector g_ℓ so that $u_\ell = Z_{nev} g_\ell$.

Interpolation-restart policy: According to Assumption 1, the Schur vectors $Z_{nconv} = [z_1, \ldots, z_{nconv}]$, and the basis $V_f = [v_1, \ldots, v_f]$ are distributed among the computing units as the matrix $T_{nconv} \in \mathbb{C}^{nconv \times nconv}$, and the Rayleigh quotient matrix $C_f \in \mathbb{C}^{f \times f}$ are replicated.

The Jacobi-Davidson algorithm enables more possibilities to regenerate a meaningful context for the restart after a fault. There are mainly two reasons. First, the algorithm does not rely on an equality that is incremented at each iteration such as Arnoldi; preserving such an incremental equality after a fault is very challenging. Second, the algorithm can start from a set of vectors and its convergence will be fast if these vectors are rich in the sought spectral information.

When the fault occurs on node p while $nconv$ ($nconv > 0$) Schur vectors were converged, good approximations of the associated converged eigenvectors can easily be computed as follows. Each non-faulty node q performs:

1. the spectral decomposition of the partial Schur matrix T_{nconv}

$$T_{nconv} G_{nconv} = G_{nconv} D \text{ with } G_{nconv} = [g_1, \ldots, g_{nconv}],$$

2. and the computation of its entries of the converged eigenvectors

$$u_\ell(I_q) = Z_{nconv}(I_p, :) g_\ell \text{ for } \ell = 1, \ldots, nconv.$$

The missing entries of the eigenvectors can be computed using IR to build $U_{nconv}^{(IR)} = [u_1^{(IR)}, \ldots, u_{nconv}^{(IR)}]$.

In addition to $U_{nconv}^{(IR)}$, further information can be extracted from the search space V_f and the Rayleigh quotient matrix C_f. Following the same methodology, spectral information built from C_f and V_f can be computed to generate additional directions to expand the initial search space ($U_{nconv}^{(IR)}$) used to restart the Jacobi-Davidson algorithm. Each non-faulty node q computes:

1. the sorted Schur decomposition of C_f, that writes $C_f \tilde{G}_f = \tilde{G}_f S_f$ so that the eigenvalues on the diagonal of S_f are sorted by increasing distance to τ, (Algorithm 4, line 5),

2. and the entries of the Ritz vectors $\tilde{u}_\ell(I_q) = V_f(I_q, :)\tilde{g}_\ell$ for $\ell = 1, \ldots, s$, where s is the number of Ritz vectors we want to interpolate. Because $\tilde{G}_f$ has been sorted, these vectors may be considered as the s best candidates to expand $U_{nconv}^{(IR)}$. That is, $\tilde{u_1}$ is the Ritz vector associated with $S(1,1)$ which is the Ritz value the closest to the target τ, that is improved by Jacobi-Davidson iterations.

In addition, the missing entries I_p of the Ritz vectors $\tilde{u}_\ell$ can be computed using LI or LSI, $U^{(IR)} = [u_1^{(IR)}, \ldots, u_{nconv}^{(IR)}, \tilde{u}_1^{(IR)}, \ldots, \tilde{u}_s^{(IR)}]$. Once $U^{(IR)}$ has been computed, the vectors in $U^{(IR)}$ are then orthonormalized to obtain $V_{restart}$. The Jacobi-Davidson algorithm can be restarted with $Z = [\ ]$, $V = V_{restart}$, $W_{restart} = AV_{restart}$, $C = V_{restart}^H W$.

Remark 1. *Let us assume that the partial Schur decomposition has converged in exact arithmetic ($AZ_{nconv} = Z_{nconv}T_{nconv}$), and that the nconv eigenpairs are exact solutions ($Au_\ell = \lambda_\ell u_\ell$) still in exact arithmetic. Under this assumption, the eigenvectors ($u_\ell^{(IR)}$) computed by IR are the same exact eigenvectors as long as $\left(A(I_p, I_p) - \lambda_\ell I_{I_p,I_p}\right)$ is nonsingular or $\left(A_{:,I_p} - \lambda_\ell \mathcal{I}_{:,I_p}\right)$ is full column rank for LI and LSI, respectively. As a consequence, if Jacobi-Davidson is restarted with the initial basis V_{nconv} obtained from the orthonormalization of the vectors of $U_{nconv}^{(IR)}$ then, the nconv already converged Schur vectors will be retrieved in the initial basis V_{nconv}. In floating point arithmetic, there is no guarantee of retrieving the already converged nconv Schur vectors by restarting with V_{nconv}, although in practice this is likely to happen as we will see in the numerical experiments.*

Although in principle s has only to satisfy $0 \leq s \leq f$, because of Remark 1, a natural choice for s is $s = nev$ (we interpolate $nconv + nev$ vectors) so that the initial search space after a fault will be at least of dimension nev.

4 Numerical experiments

In this section, we investigate the numerical behavior of the eigensolvers in the presence of faults when the IR policies are applied. In Section 4.1, we present results for subspace iteration methods and study the robustness of the IR strategies when converging eigenvectors associated with both smallest and largest eigenvalues. We assess the robustness of our resilient Arnoldi procedure in Section 4.2, whereas Section 4.3 analyzes the robustness of the resilient algorithm designed for IRAM. In Section 4.4, we discuss results obtained for Jacobi-Davidson and the impact of different variants on the convergence behavior. We define

$$\tilde{\eta}(u^{(k)}, \lambda_k) = \frac{\|Au^{(k)} - \lambda_k u^{(k)}\|}{|\lambda_k|}, \tag{3}$$

the scaled residual associated with the approximate eigenpair $(\lambda_k, u^{(k)})$ for nonzero eigenvalue approximation. This scaled residual is a lower bound of the normwise backward error of the eigenpair $(u^{(k)}, \lambda_k)$. Given a threshold ε, the widely used stopping criterion to detect convergence is defined by

$$\tilde{\eta}(u^{(k)}, \lambda_k) \leq \varepsilon.$$

For all the experiments reported in this section we set $\varepsilon = 10^{-6}$.

We have performed extensive numerical experiments and only report here on the qualitative numerical behavior observed on a few examples that are representative of our observations. Although many test matrices have been considered for evaluating the qualitative behavior of the resilient schemes, we kept only one example in this section that comes from thermo-acoustic instabilities calculation in combustion chambers [12, 13]. Indeed this test case exhibits many illustrative features. The matrix is unsymmetric; its spectrum lies in the left plane and it has small eigenvalues close to zero that can be computed without shift invert techniques using the different eigensolvers we have considered. Although its size is rather small ($n = 1500$), it exhibits numerical difficulties that are encountered on real life large scale problems [12, 13].

In order to study the numerical behavior of the IR strategy and illustrate their possible robustness and weaknesses we consider three additional executions in our numerical experiments. The first one is the non faulty (NF) execution case that provides the best expectation. In order to evaluate the impact of the restart and the quality of the interpolated values, we also report on what is referred to as the Enforced Restart (ER) execution. It consists in enforcing the solver to restart at iteration f using the available quantities at this iterations, that are:

- for the subspace iteration, ER is equivalent to NF because this numerical scheme is a fixed point iteration,

- for regular Arnoldi, it corresponds to classical Arnoldi with variable restarts,
- for IRAM and Jacobi-Davidson, the ER calculation is identical to the IR strategies except that none entries are interpolated; everything is computed using the values available at iteration f,

Finally, to illustrate the benefit and robustness of the interpolation policies, we also report on the numerical behavior of the so called Reset variant, where all the quantities interpolated by the IR strategies are simply replaced by random values.

Faults are injected using the Weibull probability distribution [15] that is supposed to be the most realistic probabilistic model that characterizes the normal behavior of large-scale computational platforms. We vary some of its parameters to increase or decrease the number of faults for a given calculation. Finally, for the sake of fair comparison, for a set of experiments (e.g, Figure 2a), faults are injected at the same iterations (e.g, 485 and 968) and during the same instructions for all cases (except NF, of course).

4.1 Resilient subspace iteration methods to compute *nev* eigenpairs

In this section, we analyze the robustness of the proposed resilient IR subspace iteration methods in the presence of faults. To analyze the robustness of our strategies, we simulate stressful conditions by increasing the fault rate and the volume of lost data. We present results for two variants of subspace iteration methods:

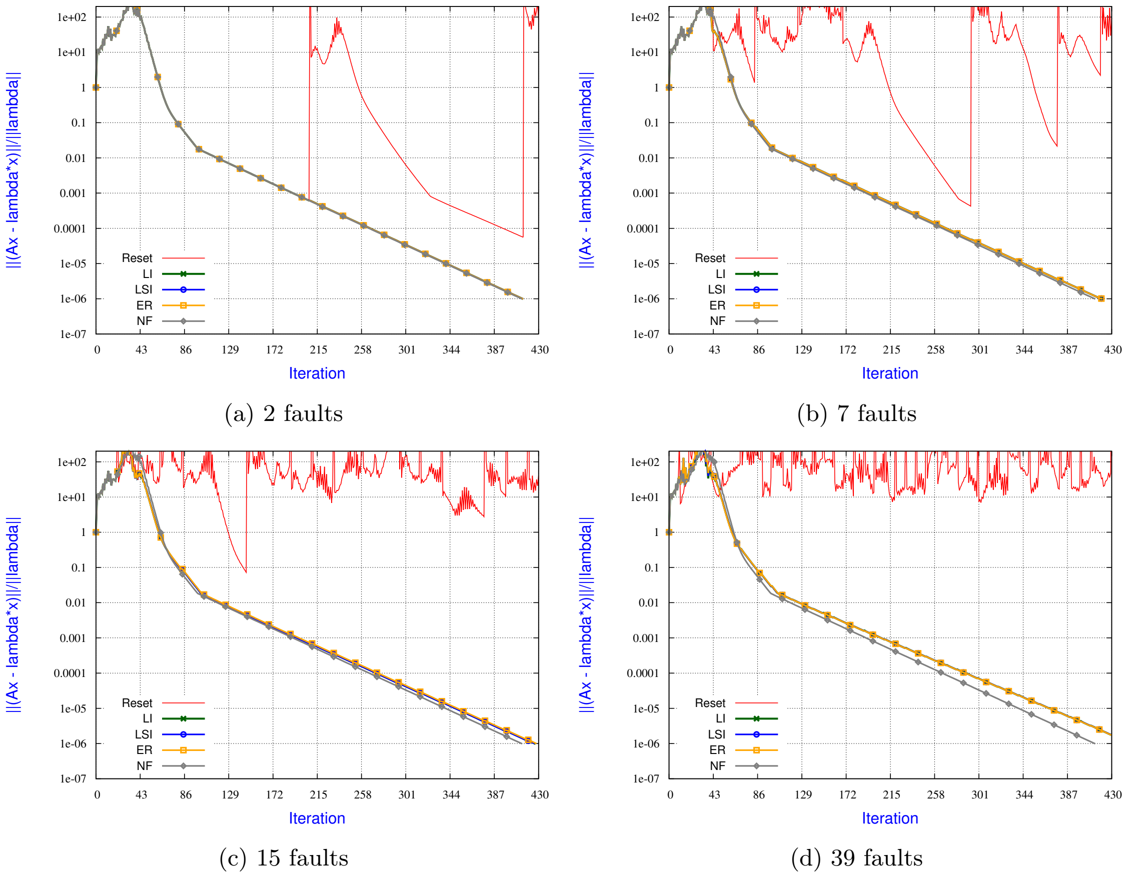
1. The subspace iteration with Chebyshev polynomial acceleration is used for the computation of the five eigenpairs corresponding to the smallest eigenvalues (Figures 2 and 3). In practice, a certain amount of information about the spectrum is needed in order to build the ellipse associated with the Chebyshev filter polynomial. The ellipse chosen must be as small as possible, but large enough to enclose the unwanted part of the spectrum. We mention that for thermo-acoustic calculations [12, 13] this prerequisite information is computationally affordable.

2. The classical method for the computation of the five eigenpairs corresponding to the largest magnitude eigenvalues (Figures 4 and 5).

For both calculations (the five largest and the five smallest eigenvalues), we report the maximum of the individual scaled residual norms (defined by Equation (3)) of the five Ritz pairs at each iteration. The execution ends when the five Ritz pairs satisfy the stopping criterion, i.e., when the maximum of the scaled residual norms is lower than the selected threshold ε.

When converging the Ritz pair associated with the smallest Ritz value, the scaled residual norm increases during the first iterations before it begin to decrease slowly reaching the target threshold in 420 iterations as in the NF case (Figures 2 and 3). With the reset strategy, the convergence history is characterized by large jumps after each fault that strongly delay the convergence. The delay is significant even with a very small fault rate (Figure 2a). When the fault rate is relatively large (Figure 2d), this strategy is likely to stagnate completely. It does exhibit large convergence peak even when a little amount of data is lost as it can be observed in Figure 3 when the amount of lost data ranges from 0.2% to 6%. Contrary to Reset, both LI and LSI are extremely robust and resilient. Indeed, regardless the number of faults and the volume of lost data, LI and LSI almost consistently overlap with ER and NF, except in the presence of a very large fault rate (Figure 2d). However, the resilience capability of LI and LSI is preserved because they overlap with ER when varying either the fault rate (Figure 2) or the volume of lost data (Figure 3).

When converging the Ritz pair associated with the largest eigenvalues, NF converges in 485 iterations as depicted in Figures 4 and 5. The Reset strategy again exhibits large peaks in the scaled residual norm after each fault, but this time, it can converge when only a few faults occur (Figure 4a) or only a small amount of data is lost (Figure 5). Regarding the robustness of both IR



(a) 2 faults

(b) 7 faults

(c) 15 faults

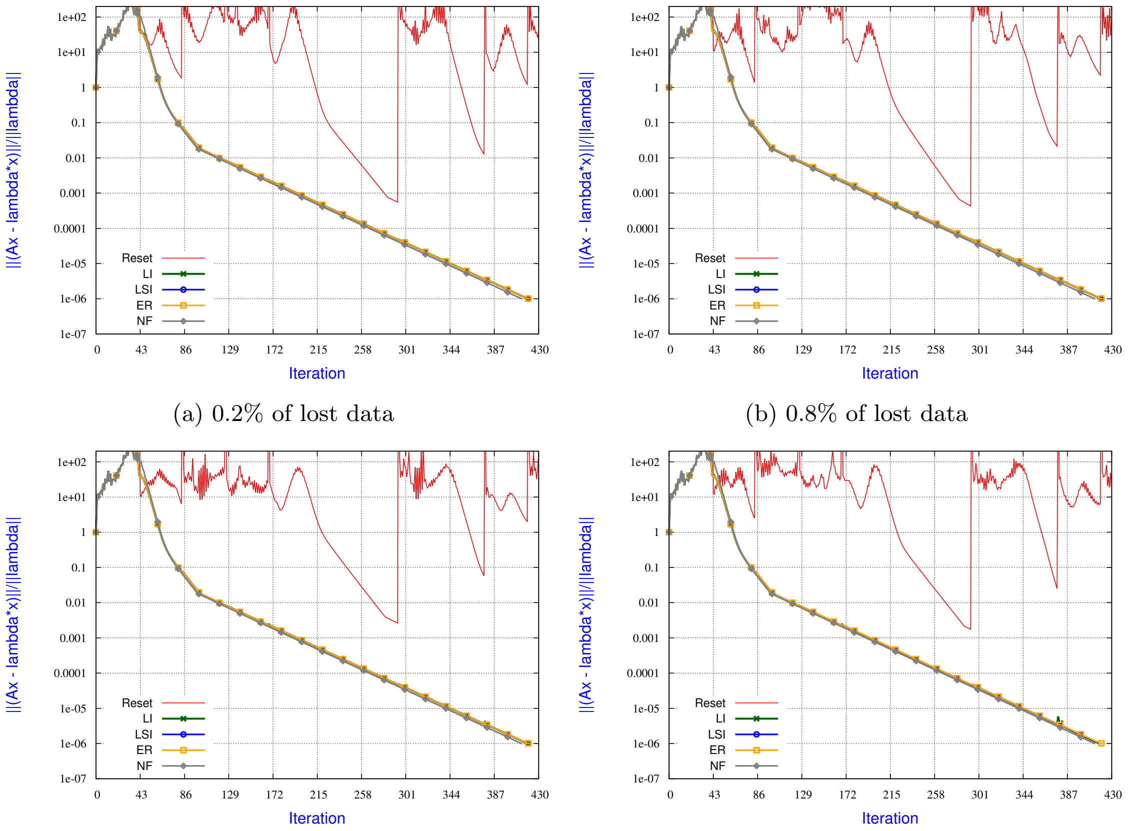
(d) 39 faults

Figure 2: **Impact of the fault rate** on the resilience of IR strategies when converging the **five eigenpairs** associated with the **smallest eigenvalues** using subspace iteration method accelerated with **Chebyshev**. A proportion of 0.8 % of data is lost at each fault. Note that LI, LSI, ER and NF almost coincide.

strategies, the convergence histories of LI and LSI almost consistently overlap again the NF curve regardless the fault rate (Figure 4) or the volume of lost data (Figure 5).

4.2 Arnoldi method to compute one eigenpair

In this section, we assess the robustness of our resilient Arnoldi for computing the eigenpair associated with the largest eigenvalue in magnitude. Because this is an easy calculation for Arnoldi, we select a rather small restart parameter $m = 7$ so that the convergence is not too fast in order to leave room to inject a few faults. One iteration consists of building a Krylov basis of size m followed by the approximation of the desired eigenpair. Because this computation requires only a few (outer) iterations we consider one single fault rate and we focus on the impact of proportion of lost data. Because the fault rate is constant the number of faults displayed in the different plots might differ depending on the convergence penalty they induce for the different resilient strategies. We report the convergence histories in Figure 6. NF converges smoothly in 9 (outer) iterations whereas Reset strategy exhibits a large peak in the scaled residual norm after each fault. When the amount of lost data is low (0.2% in Figure 6a), the Reset penalty remains reasonable (2 extra iterations), but it becomes more significant (7 extra iterations) if that amount increases (6% in Figure 6d). Because ER has to restart more often than NF, its convergence history exhibits some



(a) 0.2% of lost data

(b) 0.8% of lost data

(c) 3% of lost data

(d) 6% of lost data

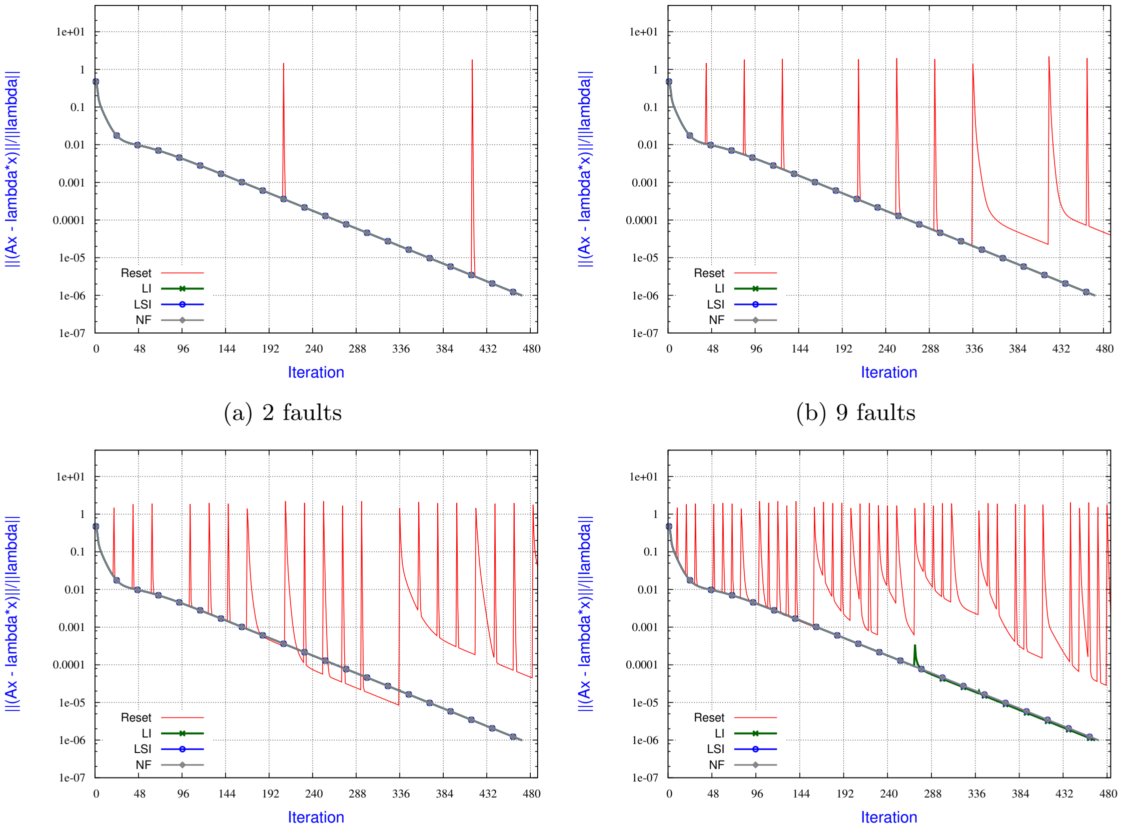
Figure 3: **Impact of the amount of lost data** on the resilience of IR strategies when converging the **five eigenpairs** associated with the **smallest eigenvalues** using subspace iteration method accelerated with **Chebyshev**. The volume of lost data varies from 0.2% to 6% whereas the fault rate is constant (7 faults). LI, LSI, ER and NF coincide.

delay compared to NF. On the other hand, both IR strategies are again extremely robust. Indeed, LI and LSI convergence coincide with ER, regardless the proportion of lost data (Figure 6). Note that if the proportion of lost data is very large (Figure 6d), LI and LSI may slightly differ from ER. The fact that LI and LSI convergence coincide with ER, indicates that the spectral information regenerated by the LI and LSI is as good as the one computed by the regular solver.

4.3 Implicitly restarted Arnoldi method to compute *nev* eigenpairs

To investigate the robustness of IR strategies designed for IRAM, we compute the five eigenpairs ($nev = 5$) that correspond to the largest magnitude eigenvalues. At each iteration, we report the maximum of the scaled residual norms of those five sought eigenpairs. We consider a restart parameter m=10 (see Algorithm 2). One iteration thus consists of building a Krylov subspace of size 10, followed by the computation of the approximate eigenpairs. If the eigenpairs do not satisfy the stopping criterion, the next iteration starts with a contracted Arnoldi equality of size $\tilde{m} = 5$.

The NF calculation computes the five sought eigenvectors in 11 (outer) iterations (index k in Algorithm 2). The Reset strategy exhibits a large peak in the scaled residual norm after each fault, its scaled residual norm increases further than the initial one. As a consequence, convergence is very much delayed. Furthermore, Reset is also sensitive to the amount of lost data. The larger the



(a) 2 faults

(b) 9 faults

(c) 19 faults

(d) 36 faults

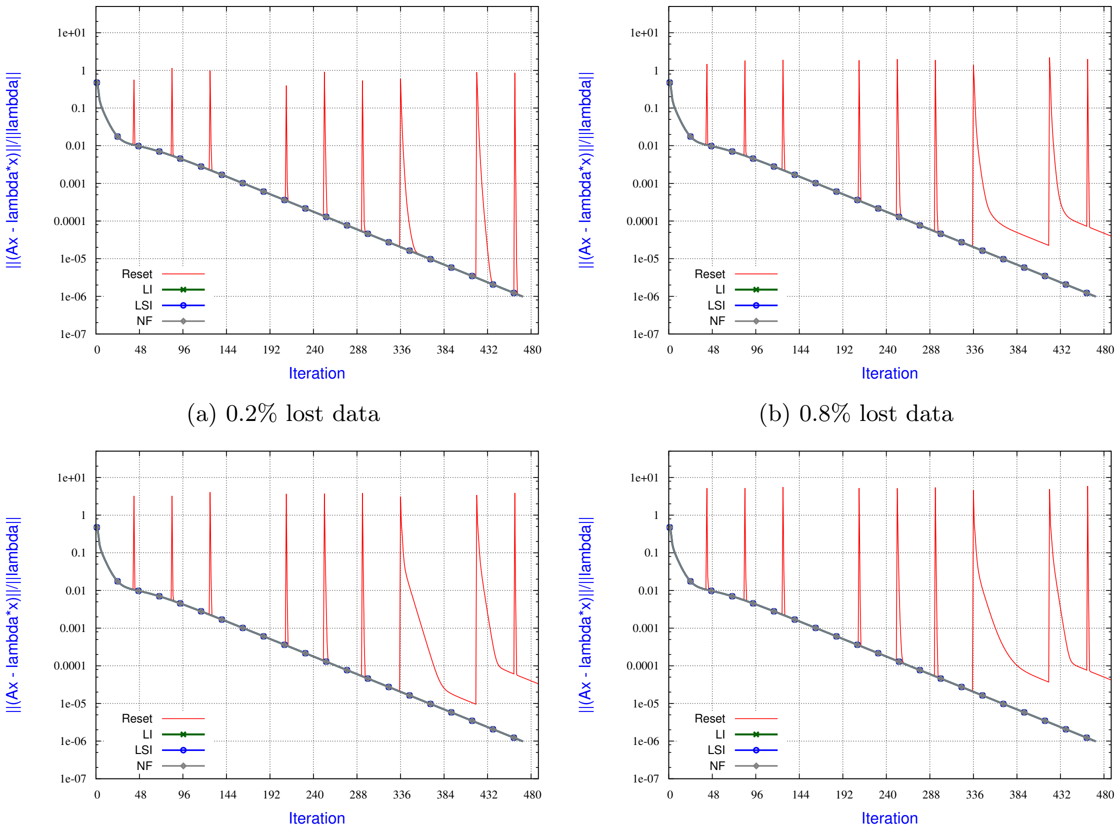
Figure 4: **Impact of the fault rate** on the resilience of IR strategies when converging the **five eigenpairs** associated with the **largest eigenvalues** using the **basic subspace iteration method**. A proportion of 0.8 % of data is lost at each fault. LI, LSI and NF coincide in (a), (b) and (c).

volume of lost data, the more its convergence is delayed (Figure 7).

On the other hand, both IR strategies are much more robust than Reset. However, they still require a few extra iterations than NF. Because the high coincidence with ER, it can be concluded that this slight penalty is not due to the quality of interpolation but to the necessity of restarting with the information of the five dimension space compressed in one single direction.

4.4 Jacobi-Davidson method to compute *nev* eigenpairs

In this section, we investigate the resilience of the IR strategies designed for Jacobi-Davidson. In all the experiments, we seek for the five ($nev = 5$) eigenpairs whose eigenvalues are the closest to zero ($\tau = 0$). The correction equation is solved using 30 iterations of GMRES. To facilitate the readability and the analysis of the convergence histories plotted in this section, we use vertical green lines to indicate the convergence of new eigenpairs (such as iterations 95, 130, 165, 200 and 242 in Figure 8a), and vertical red lines to indicate faulty iterations (such as iterations 148 and 228 for the sixth and ninth fault, respectively, in Figure 8a). According to Remark 1, although very likely to happen, there is no guarantee of retrieving all the already converged Schur vectors in the basis used to restart. As a consequence, we indicate the number of Schur vectors retrieved in the basis used to restart in red color under the vertical red line corresponding to the associated



(a) 0.2% lost data

(b) 0.8% lost data

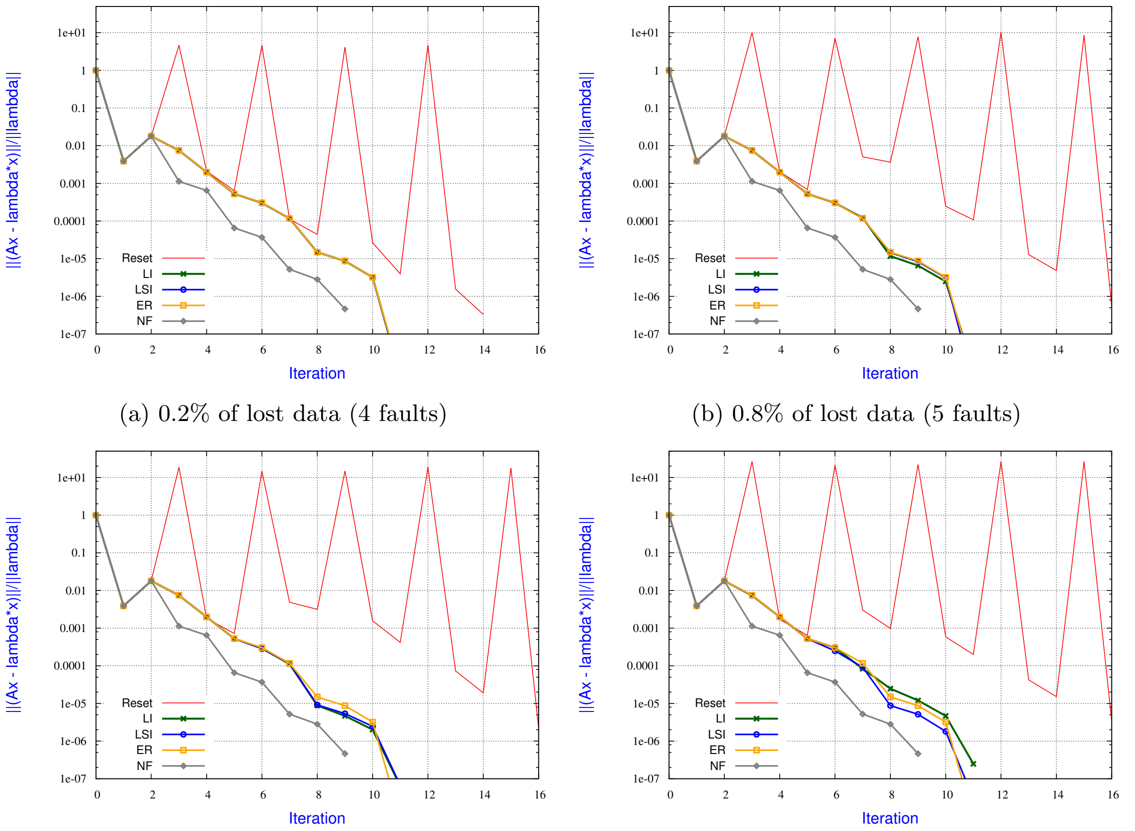
(c) 3% lost data

(d) 6% lost data

Figure 5: **Impact of the amount of lost data** on the resilience of IR strategies when converging the **five eigenpairs** associated with the **largest eigenvalues** using **basic subspace iteration method**. The volume of lost data varies from 0.2% to 6% whereas the fault rate is constant (9 faults). LI, LSI and NF coincide.

fault. For instance, 2 already converged Schur vectors are immediately retrieved at restart, after the fault at iteration 148 in Figure 8a. In the Jacobi-Davidson method there is some flexibility for selecting the number of vectors (*i.e.*, the dimension of the space generated for restarting) that can be interpolated after a fault. For our experiments we choose to interpolate the converged Schur vectors as well as nev of the best candidates for Schur vectors extracted from the search space V_f. For the calculation of the five smallest eigenvalues, the NF algorithm converges in 210 iterations while faulty executions have extra iterations. For the sake of comparison, we consider only the first 300 iterations of all the runs so that the graphs have exactly the same scales and range of iteration count.

We first consider all the different restarting strategies. We report in Figure 8 their convergence histories in different subplots for a fixed fault rate to evaluate the quality of the basis used for the restarts. The curves in Figure 8a show the impact of the enforced restarts (35 additional iterations for ER compared to NF) and will serve as reference to evaluate the quality and relevance of the interpolated directions considered for LI, LSI and Reset. The first comment goes to the Reset approach that completely fails and is unable to compute any eigenpair. The LI interpolation behaves much better but only succeeds to compute four out of the five eigenpairs in 300 iterations. For this example, LSI is slightly more robust and computes the five eigenpairs with a few extra iterations compared to ER. For both IR approaches it can be observed that the converged Schur



(a) 0.2% of lost data (4 faults)

(b) 0.8% of lost data (5 faults)

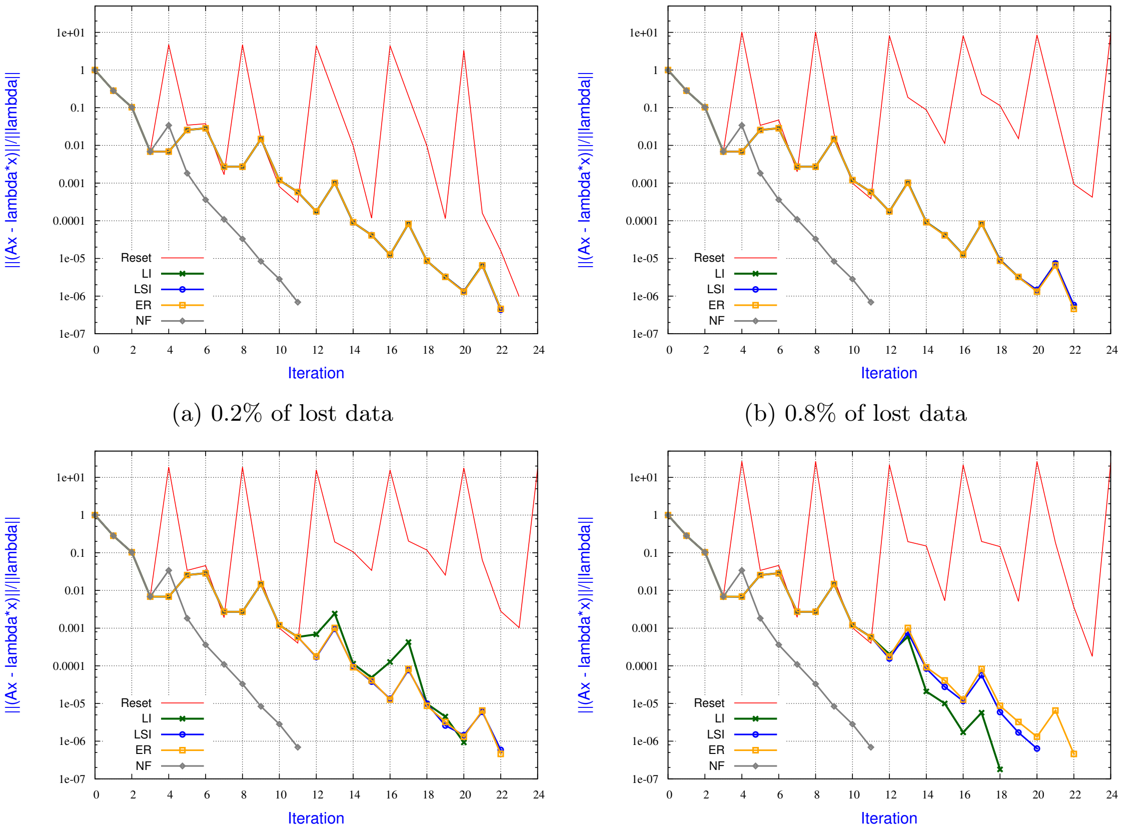
(c) 3% of lost data (5 faults)

(d) 6% of lost data (5 faults)

Figure 6: **Impact of the amount of lost data** on the resilience of IR strategies when converging the eigenpair associated with the **largest eigenvalue** using **Arnoldi method**. LI, LSI and NF coincide in (a) (b) and (c).

vectors are immediately recovered after a fault by the first step of the Jacobi-Davidson method that is the Raleigh quotient procedure. As illustrated in Figures 8c and 8b for instance, when a fault occurs, the *nconv* converged Schur vectors before a fault are found immediately from the interpolated vectors at restart (e.g, in Fig 8b, the value 4 under the vertical red line at the faulty iteration 275 means that all four Schur vectors converged before the fault are immediately rediscovered). Finally we notice that even if, in Figure 8, experiments are performed with the same fault rate, the number of faults varies depending on the interpolation strategy. Indeed, the less robust the strategy, the more the convergence is delayed, which increases the probability for additional faults to occur.

In order to reduce the number of graphs to illustrate the impact of the amount of lost data (see Figure 9) and the influence of the fault rate (see Figure 10), we only consider the LSI approach in the rest of this section. The influence of the volume of lost data is displayed in Figure 9 where its proportion varies from 0.2 % to 6 % while the fault rate remains constant. As one would expect, the general trend is: the larger the amount of lost data, the larger the convergence penalty. A large amount of lost data at each fault prevents LSI from converging the five eigenpairs; only four are eventually computed within 300 iterations for 6 % lost. Notice that allowing for more than the 300 iterations would probably enable the solver to succeed in the calculation of the five eigenpairs.



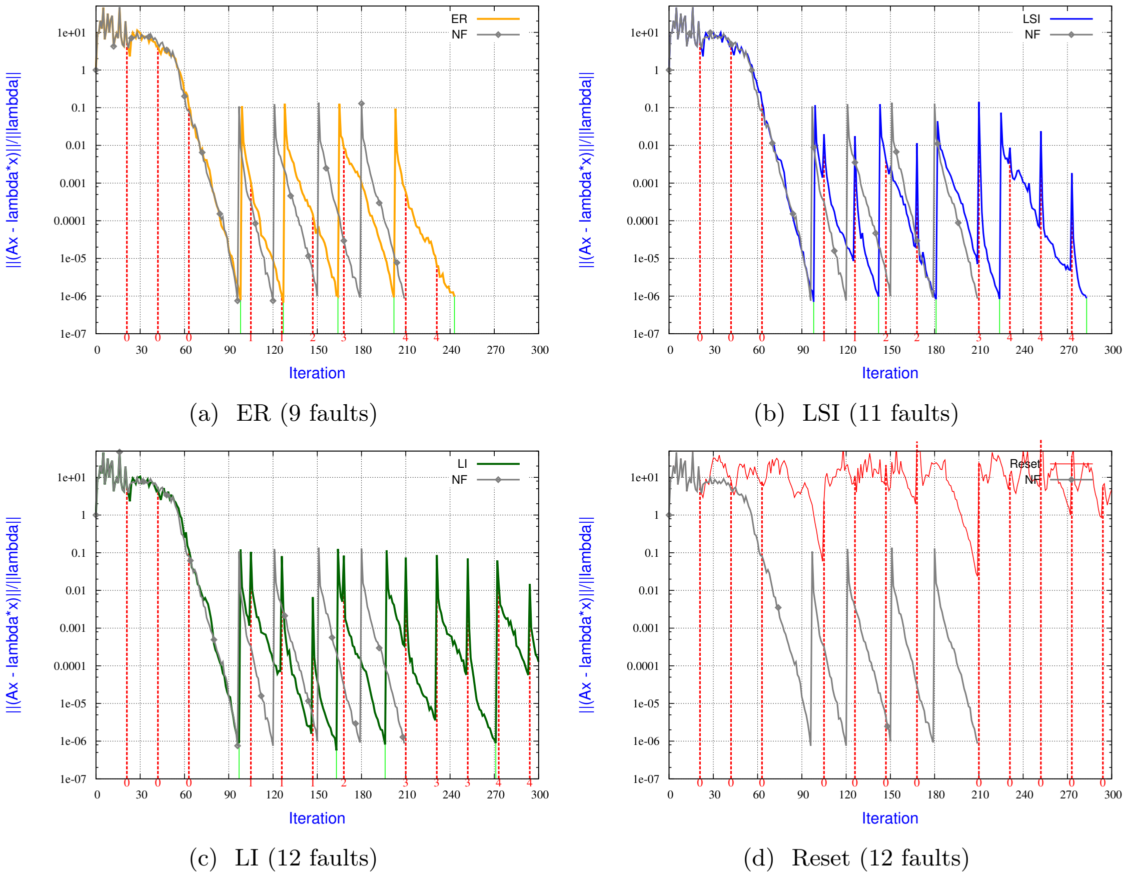
(a) 0.2% of lost data

(b) 0.8% of lost data

(c) 3% of lost data

(d) 6% of lost data

Figure 7: **Impact of the amount of lost data** on the resilience of IR strategies when converging the **five eigenpairs** with **largest eigenvalues** using **IRAM**. The volume of lost data varies from 0.2% to 6% whereas the fault rate is constant. LI, LSI and ER coincide in (a) and (b).



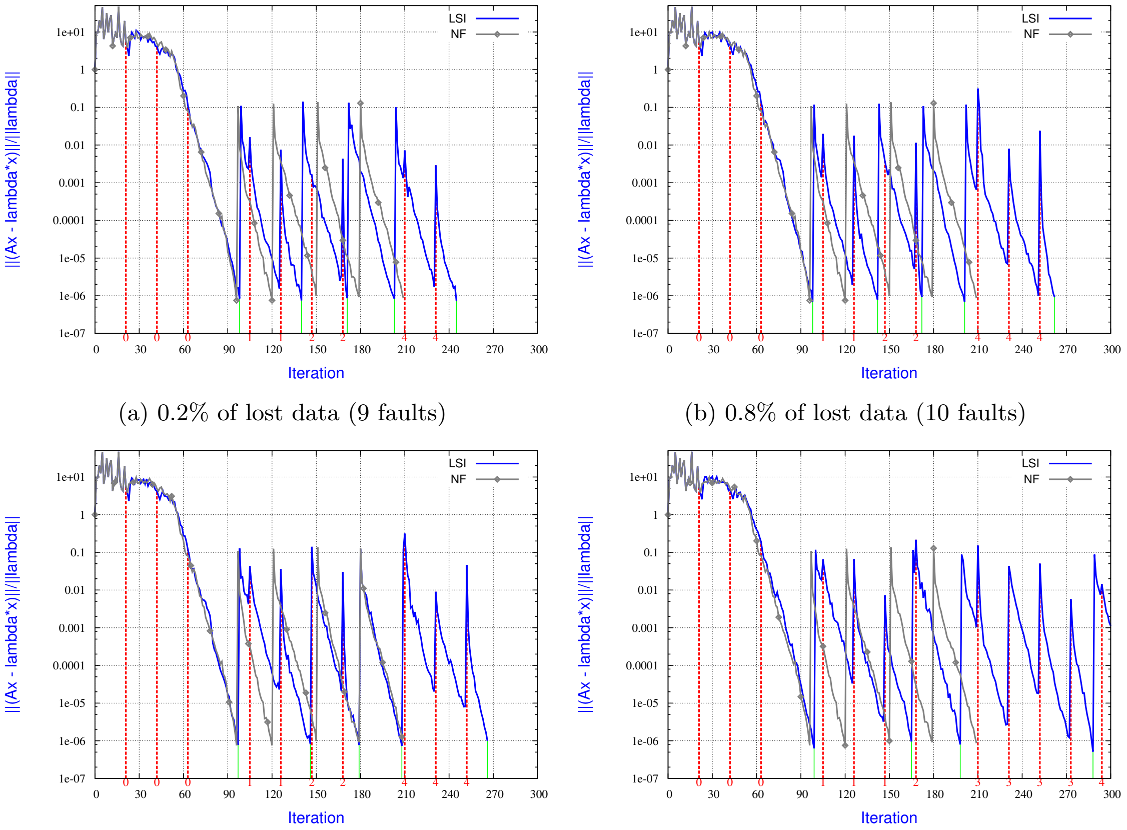
(a) ER (9 faults)

(b) LSI (11 faults)

(c) LI (12 faults)

(d) Reset (12 faults)

Figure 8: **Comparison of IR strategies** using *nev* regenerated vectors when converging the **five eigenpairs** associated with the **smallest eigenvalues** using **Jacobi-Davidson**. The fault rate is the same for all strategies and a proportion of 0.8 % data is lost at each fault.



(a) 0.2% of lost data (9 faults)

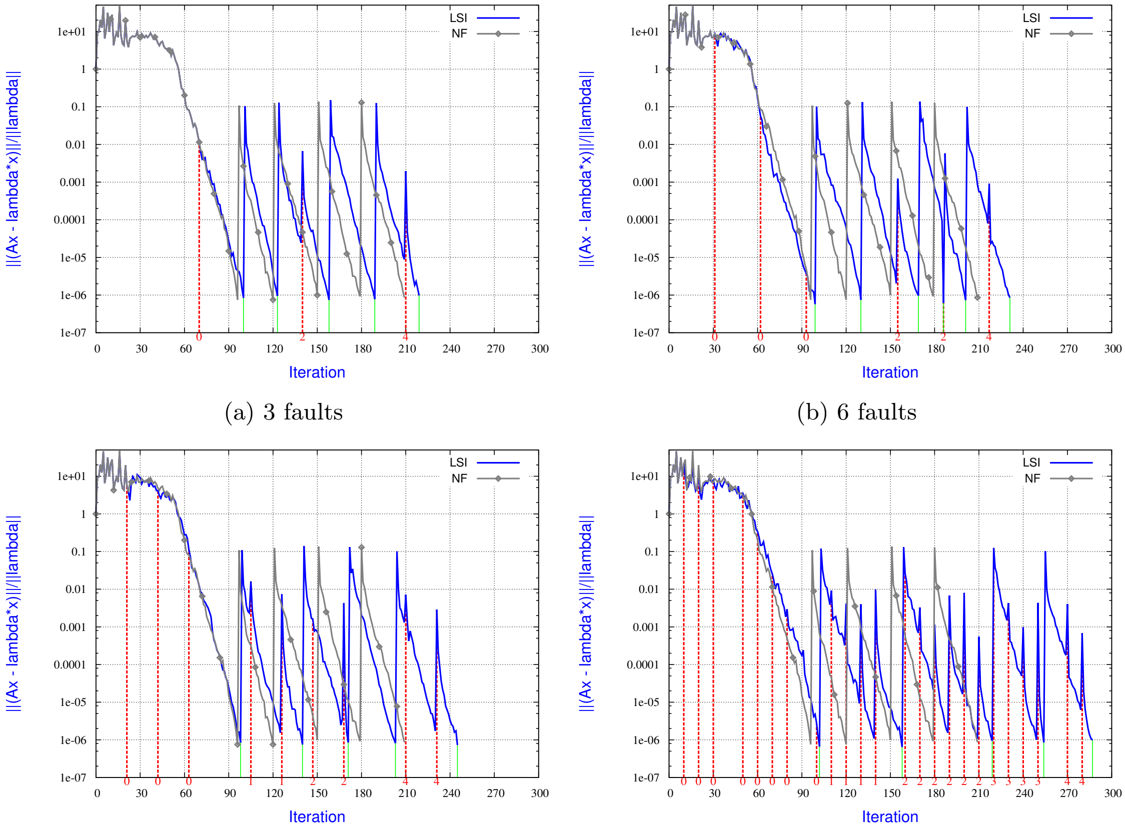
(b) 0.8% of lost data (10 faults)

(c) 3% of lost data (10 faults)

(d) 6% of lost data (12 faults)

Figure 9: **Impact of the amount of lost data** on the resilience of **LSI** using $(nev + nconv)$ regenerated vectors when converging the **five eigenpairs** associated with the **smallest eigenvalues** using **Jacobi-Davidson**. The volume of lost data varies from 0.2% to 6% whereas the fault rate is constant. All converged Schur vectors are found immediately after interpolation followed by restart.

Finally, in Figure 10 we depict the convergence histories of the various restarting strategies when the fault rate varies leading to a number of faults that varies from 3 to 24; as expected, the larger the number of faults, the larger the convergence delay. However, the IR policy is rather robust and succeeds in converging the five eigenpairs in less than 300 iterations.



(a) 3 faults

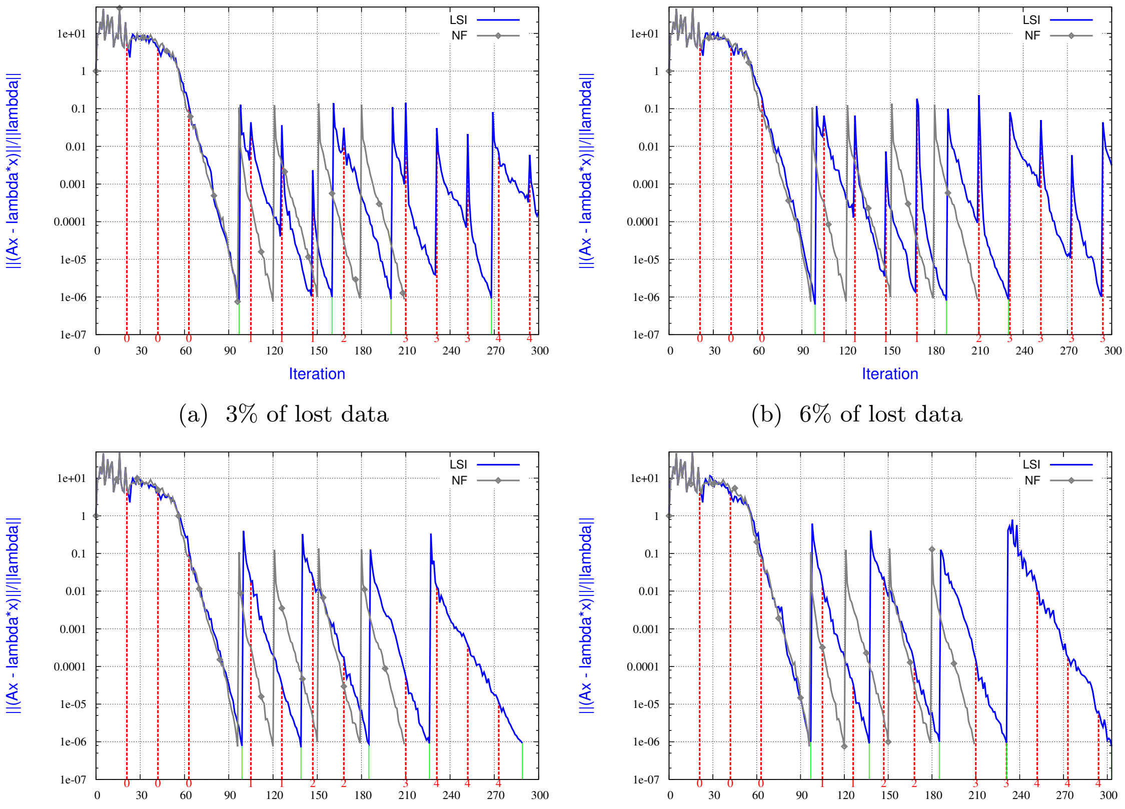
(b) 6 faults

(c) 9 faults

(d) 24 faults

Figure 10: **Impact of the fault rate** on the resilience of **LSI** using $(nev + nconv)$ regenerated vectors when converging the **five eigenpairs** associated with the **smallest eigenvalues** using **Jacobi-Davidson**. The fault rate varies whereas a proportion of 0.2 % of data is lost at each fault. At each fault, all the already converged Schur vectors are retrieved in the basis of restart.

Despite the robustness of the resilient schemes, peaks of the residual norm associated with the current Schur vector are often observed after each fault. A possible remedy consists of using a hybrid approach where we interpolate the $nconv$ Schur vector while reusing the best candidate Schur vector available in V_f (as if we had checkpointed this single direction) when the fault occurs and interpolate $nev-1$ directions but the first to recover additional meaningful spectral information from V_f. This procedure is beneficial for improving the convergence independently of the size of the space used for the restart. It is best highlighted in the most difficult situations that are large amount of lost data (see Figure 11) and large number of faults (see Figure 12). In these figures, the convergence of the scaled residual norm no longer exhibits peak after the faults when the best Schur candidate vector is used in the set of directions for the restart. The most relevant information on the next Schur vector to converge seems to be concentrated in the current best Schur candidate vector. Intensive experiments show that the interpolation strategies do not succeed in regenerating accurately enough the lost entries of this special direction of V_f. Consequently, a reasonable trade-off may consist in checkpointing only the current Schur vector in order to increase the robustness of the resilient Jacobi-Davidson method.



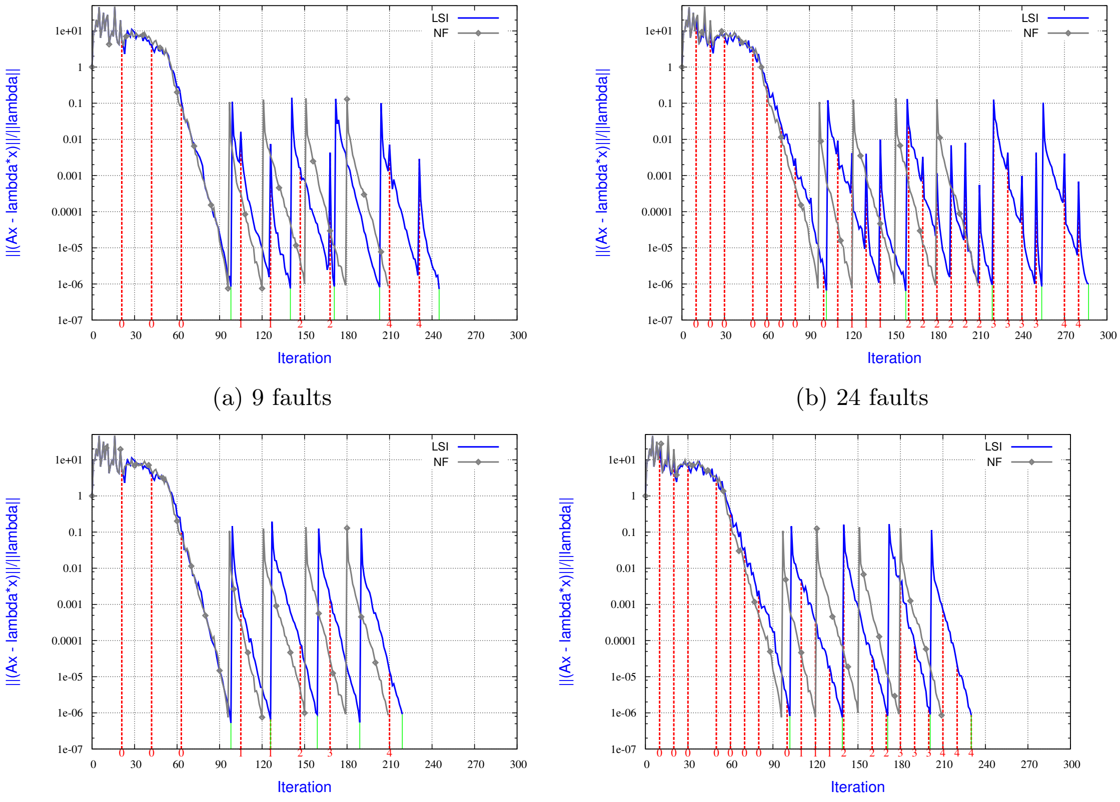
(a) 3% of lost data

(b) 6% of lost data

(c) 3% of lost data

(d) 6% of lost data

Figure 11: Impact of **keeping the best Schur vector candidate** in the search space after a fault combined with $nev + nconv - 1$ interpolated directions. Top two plots nev interpolated directions, bottom two plots $nev - 1$ interpolated directions plus the best Schur candidate. Calculation of the **five eigenpairs** associated with the **smallest eigenvalues** using **Jacobi-Davidson**. The fault rate is constant over all sub-figures. The proportion of lost data is either 3% in Figures (a) and (c) or 6% in Figures (b) and (d). All converged Schur vectors are found immediately after interpolation followed by restart.



(a) 9 faults

(b) 24 faults

(c) 8 faults

(d) 20 faults

Figure 12: Impact of **keeping the best Schur vector candidate** in the search space after a fault combined with $nev + nconv - 1$ interpolated directions. Top two plots $nev + nconv$ interpolated directions, bottom two plots $nev + nconv - 1$ interpolated directions plus the best Schur candidate. Calculation of the **five eigenpairs** associated with the **smallest eigenvalues** using **Jacobi-Davidson** with a proportion of 0.2 % of data is lost at each fault.

5 Concluding remarks

Many scientific and engineering applications require the computation of eigenpairs of large sparse matrices. The objective of the paper has been to study numerical schemes suitable for the design of resilient parallel eigensolvers. For that purpose, we have proposed two interpolation procedures to regenerate meaningful spectral information for restarting the eigensolver after a fault. To evaluate the qualitative behavior of the resilient schemes, we have simulated stressful conditions by increasing the fault rate and the volume of lost data.

We have considered two variants of subspace iteration methods. On the one hand, we have considered the subspace iteration with Chebyshev polynomial acceleration for the computation of eigenpairs corresponding to the smallest eigenvalues. On the other hand, we have considered the classical method for the computation of eigenpairs corresponding to the largest magnitude eigenvalues. For both methods, the Reset strategy strongly penalizes the convergence at each fault, while both LI and LSI are extremely robust and resilient, regardless the number of faults and the volume of lost data. The same numerical behavior is observed for our resilient Arnoldi for computing the eigenpair associated with the largest eigenvalue in magnitude. Our LI/LSI resilient IRAM for the computation of a few eigenpairs are much more robust than Reset. However, they do exhibit a slight penalty, not due to the quality of interpolation, but to the restarting policy that leads to compressing in a single direction the eigenspace under calculation when faults occur.

We have had a stronger emphasis on the Jacobi-Davidson method. The motivation is twofold: the Jacobi-Davidson method is widely used in many real-life applications, and, in addition, it offers some flexibility to select different spectral information to construct an efficient restart mechanism after a fault. We have observed that despite the increase of the amount of recovered data, the peak of the residual norm associated with the current Schur vector persists after a fault. For a possible remedy of these effects, we have designed a hybrid approach that consists in combining interpolation techniques with classical checkpoint for a single vector. This illustrates that numerical resilient strategies can be effectively combined with the state-of-the-art fault tolerant policies to design efficient and robust methods as demonstrated for the Jacobi-Davidson algorithm by the combination of light checkpointing and a numerical resilience.

Finally, the effectiveness of the proposed approaches has been illustrated from a numerical view point. Their effectiveness and scalable implementation in a parallel computing environment deserves to be studied once the fault tolerant supports have been made available in the MPI standard.

Acknowledgments

This work was partially supported by the French research agency ANR in the framework of the RESCUE project (ANR-10-BLANC-0301), in particular the PhD thesis of the fourth author was funded by this project. This research also benefited from the G8-ECS project.

References

[1] E Agullo, L Giraud, and M Zounon. On the resilience of a parallel sparse hybrid solver. Research Report 8744, INRIA Bordeaux, June 2015.

[2] Emmanuel Agullo, Luc Giraud, Abdou Guermouche, Jean Roman, and Mawussi Zounon. Towards resilient parallel linear Krylov solvers: recover-restart strategies. Research Report RR-8324, INRIA, July 2013.

[3] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Q. Appl. Math*, 9(17):17–29, 1951.

[4] Diederik R. Fokkema, Gerard L. G. Sleijpen, and Henk A. Van der Vorst. Jacobi-Davidson style QR and QZ algorithms for the partial reduction of matrix pencils. *SIAM J. SCI. COMPUT*, 20:94–125, 1996.

[5] Melina A. Freitag and Alastair Spence. Shift-invert Arnoldi's method with preconditioned iterative solves. *SIAM J. Matrix Anal. Appl.*, 31(3):942–969, August 2009. ISSN 0895-4798. doi:10.1137/080716281.

[6] G.W. Stewart. *Matrix algorithms – Volume II: Eigensystems.* SIAM, 2001.

[7] L. Jaulmes, M. Casas, M. Moretó, E. Ayguade, J. Labarta, and M. Valero. Exploiting asynchrony from exact forward recovery for DUE in iterative solvers. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC'15)*. Austin, USA, November 2015.

[8] Julien Langou, Zizhong Chen, George Bosilca, and Jack Dongarra. Recovery Patterns for Iterative Methods in a Parallel Unstable Environment. *SIAM J. Sci. Comput.*, 30:102–116, November 2007. ISSN 1064-8275. doi:10.1137/040620394.

[9] R.B. Lehoucq and D. C. Sorensen. Deflation techniques for an implicitly re-started Arnoldi iteration. *SIAM J. Matrix Anal. Appl*, 17:789–821, 1996.

[10] Y. Saad. *Iterative Methods for Sparse Linear Systems.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2003. ISBN 0898715342.

[11] Yousef Saad. *Numerical Methods for Large Eigenvalue Problems.* Manchester University Press, Manchester, UK, 1992.

[12] Pablo Salas. *Physical and numerical aspects of thermoacoustic instabilities in annular combustion chambers.* Ph.D. thesis, Université Bordeaux 1, November 2013.

[13] Pablo Salas, Luc Giraud, Yousef Saad, and Stéphane Moreau. Spectral recycling strategies for the solution of nonlinear eigenproblems in thermoacoustics. Research Report RR-8542, INRIA, May 2014.

[14] Gerard L. G. Sleijpen and Henk A. Van der. A Jacobi–Davidson iteration method for linear eigenvalue problems. *SIAM Rev.*, 42(2):267–293, June 2000.

[15] Wallodi Weibull. A statistical distribution function of wide applicability. *Journal of Applied Mechanics*, 18:293–297, 1951.

RESEARCH CENTRE
BORDEAUX – SUD-OUEST

200 avenue de la Vielle Tour
33405 Talence Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399