SAND2018-0720C

Chapter 1
The Resiliency of Multilevel Methods on Next Generation Computing Platforms: Probabilistic Model and Its Analysis

Mark Ainsworth and Christian Glusa

Abstract The reduced reliability of next generation exascale systems means that the resiliency properties of a numerical algorithm will become an important factor in both the choice of algorithm, and in its analysis. The multigrid algorithm is the workhorse for the distributed solution of linear systems but little is known about its resiliency properties and convergence behavior in a fault-prone environment. In the current work, we propose a probabilistic model for the effect of faults involving random diagonal matrices. We summarize results of the theoretical analysis of the model for the rate of convergence of fault-prone multigrid methods which show that the standard multigrid method will not be resilient. Finally, we present a modification of the standard multigrid algorithm that will be resilient.

1.1 Introduction

Exascale computing is anticipated to have a huge impact on computational simulation. However, as the number of components in a system becomes larger, the likelihood of one or more components failing or function abnormally during an application run increases. The problem is exacerbated by

Mark Ainsworth
Division of Applied Mathematics, Brown University, 182 George St, Providence, RI 02912, USA, e-mail: mark_ainsworth@brown.edu

Christian Glusa
Division of Applied Mathematics, Brown University, 182 George St, Providence, RI 02912, USA

Center for Computing Research, Sandia National Laboratories, Albuquerque, NM 87185, USA, e-mail: caglusa@sandia.gov
This research was performed at Brown University as part of Christian Glusa's dissertation [11].

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

the decreasing physical size of basic components such as transistors, and the accompanying increased possibility of quantum tunneling corrupting logic states [6, 7].

Current day petascale systems already exhibit a diverse range of faults that may occur during computation. These faults can arise from failures in the physical components of the system, or intermittent software faults that appear only in certain application states. One source of faults is cosmic radiation with charged particles, which can lead to memory bit-flips or incorrect behavior of logic units. Future HPC systems are expected to be built from even larger numbers of components than current systems, and the rate of faults in the system will increase accordingly. It is generally accepted that future large-scale systems must operate within a 20MW power envelope. This will require the usage of lower voltage logic thresholds. Moreover, cost constraints will result in greater utilization of consumer grade components, with accompanying reduced reliability [7].

Roughly speaking, faults can be classified as follows [3]: *hard* or *stop-fail* faults are faults which would otherwise lead to an immediate program termination, unless treated on the system level. *Soft* faults are those leading to program or data corruption, and which might only result in an erroneous program termination after some delay.

Reported fault rates seem to vary significantly from system to system. On current machines, hard faults have been reported as often as every 4 to 8 hours on the Blue Waters system [7], and (detected) L1-cache soft errors as often as every 5 hours on a large BlueGene/L system [8]. The next-generation supercomputers could have a mean-time to failure of about 30 minutes [21].

Many of the existing algorithms in use today were derived and analyzed without taking account of the effect of these kinds of faults. We believe that the dawning of the exascale era poses new, and exciting, challenges to the numerical analyst in understanding and analyzing the behavior of numerical algorithms on a fault-prone architecture. Our view is that on future exascale systems, the possible impact of faults on the performance of a numerical algorithm must be taken fully into account in the analysis of the method.

In order to alleviate the impact of faults and ensure resilience in a fault-prone environment, several techniques have been proposed and implemented in various parts of the hardware-software stack. Checkpointing on the system and the application level as well as replication of critical program sections are widely used [15, 7, 5]. These techniques can be coupled with statistical analysis, fault models, and hardware health data [7]. On the application level, Algorithm-Based Fault Tolerance (ABFT) describes techniques that duplicate application data to create redundancy [16]. ABFT has been explored in the context of sparse linear algebra [20, 19], and specifically for matrix-vector products in stationary iterative solvers [10, 8, 9, 17, 22]. All methods have in common that a balance needs to be struck between protecting against corruption of results and keeping the overhead reasonable.

The multigrid method is the workhorse for distributed solution of linear systems but little is known about about its resiliency properties and convergence behavior in a fault-prone environment [12, 17]. The current article presents a summary of our recent work addressing this problem [1, 2].

The outline of the remainder of this article is as follows: We give a short introduction to multi-level methods in Section 1.2. In Section 1.3, we introduce a model for faults and show simulations of the convergence behavior of a fault-prone two-level method for a finite element method. Finally, in Section 1.4, we summarize the analytic bounds on the convergence rate, and illustrate their behavior with further simulations. We refer the interested reader for further details and proof to the articles [1, 2].

1.2 Multi-level methods

Let $\Omega \subset \mathbb{R}^d$ be a polygonal domain and set $V := H_0^1(\Omega)$. Starting from an initial triangulation $\mathcal{T}_0$ of Ω into simplices, we obtain $\mathcal{T}_l$ through uniform refinement of $\mathcal{T}_{l-1}$. We define the finite element spaces $V_l := \{v \in H_0^1(\Omega) \cap C(\bar{\Omega})$ such that $v\big|_K \in \mathbb{P}_1(K), \ \forall K \in \mathcal{T}_l\}$, and set $n_l := \dim V_l$. For $f \in H^{-1}(\Omega)$, consider the well-posed problem:

$$\text{Find } u \in V \text{ such that } \quad a(u,v) = L(v), \quad \forall v \in V,$$

where $a(u,v) = \int_\Omega \nabla u \cdot \nabla v$ and $L(v) = \int_\Omega f v$. The discretized problem is:

$$\text{Find } u \in V_l \text{ such that } \quad a(u,v) = L(v), \quad \forall v \in V_l.$$

Let $\phi_l^{(i)}$ for $i = 1, \dots, n_l$ be the global shape function basis of V_l, and ϕ_l the vector of global shape functions. Then the stiffness matrix and the load vector are defined as $A_l := a(\phi_l, \phi_l)$ and $b_l := L(\phi_l)$, so that the problem becomes:

$$\text{Find } u = \phi_l \cdot x_l \in V_l \text{ such that } \quad A_l x_l = b_l. \tag{1.1}$$

Since $V_{l-1} \subset V_l$, there exists a restriction matrix r_{l+1}^l satisfying $\phi_l = r_{l+1}^l \phi_{l+1}$ along with the corresponding prolongation matrix $p_l^{l+1} = \left(r_{l+1}^l\right)^T$. In particular, this means that the stiffness matrix on level l can be expressed in terms of the matrix at level $l+1$:

$$A_l = a(\phi_l, \phi_l) = r_{l+1}^l a(\phi_{l+1}, \phi_{l+1}) p_l^{l+1} = r_{l+1}^l A_{l+1} p_l^{l+1}.$$

We shall omit the sub- and superscripts on r and p whenever it is clear which operator is meant. We shall consider solving the system (1.1) using the multigrid method [4, 18, 13, 14, 23]. The coarse-grid correction is given by $x_l \leftarrow x_l + pA_{l-1}^{-1} r(b_l - A_l x_l)$, and has iteration matrix $C_l := I - pA_{l-1}^{-1} r A_l$, while the damped Jacobi smoother is given by $S_l = I - \theta D_l^{-1} A_l$, where D_l is the diagonal of A_l and θ the relaxation parameter. The multi-level method for the solution of $A_L x_L = b_L$ is given in Algorithm 1. Here, ν_1 and ν_2 are the number of pre- and post-smoothing steps, γ is the number of coarse-grid corrections, and θ is the smoothing parameter.

Function $\mathcal{M}_l$(*right-hand side* b_l, *initial guess* x_l)
- **if** $l = 0$ **then return** $A_0^{-1} x_0$ (Exact solve on coarsest grid)
- **else**
  - **for** $i \leftarrow 1$ **to** ν_1 **do**
    - $x_l \leftarrow x_l + \theta D_l^{-1} (b_l - A_l x_l)$ (Pre-smoothing)
  - $d_{l-1} \leftarrow r (b_l - A_l x_l)$ (Restriction to coarser grid)
  - $e_{l-1}^{(0)} \leftarrow 0$
  - **for** $j \leftarrow 1$ **to** γ **do**
    - $e_{l-1}^{(j)} \leftarrow \mathcal{M}_{l-1} \left(d_{l-1}, e_{l-1}^{(j-1)} \right)$ (Solve on coarser grid)
  - $x_l \leftarrow x_l + p e_{l-1}^{(\gamma)}$ (Prolongation to finer grid)
  - **for** $i \leftarrow 1$ **to** ν_2 **do**
    - $x_l \leftarrow x_l + \theta D_l^{-1} (b_l - A_l x_l)$ (Post-smoothing)
  - **return** x_l

Algorithm 1: Multi-level method $\mathcal{M}_l$

1.3 Fault model

The first issue is to decide on how the effect of a fault should be incorporated into the analysis of the algorithm. The simplest and most convenient course of action if a component is subject to corruption, or fails to return a value, is to overwrite the value by zero. We therefore propose to model the effect of a fault on a vector using a random diagonal matrix $\boldsymbol{\mathcal{X}}$, of the form

$$\boldsymbol{\mathcal{X}} = \begin{pmatrix} \chi_1 & & \\ & \ddots & \\ & & \chi_n \end{pmatrix}, \qquad \chi_i = \begin{cases} 1 & \text{with probability } 1-q, \\ 0 & \text{with probability } q. \end{cases} \tag{1.2}$$

In particular, if a vector $x \in \mathbb{R}^n$ is subject to faults, then the corrupted version of x is given by $\boldsymbol{\mathcal{X}} x$. If all χ_i are independent, we will call the random matrix a matrix of *component-wise* faults. More generally, we shall make the following assumption on the set $\mathcal{S}$ of all the involved faults matrices $\boldsymbol{\mathcal{X}}$:

(A) There exist constants v, $C_e \geq 0$, and for each $\boldsymbol{\mathcal{X}} \in \mathcal{S}$ there exists $e_{\boldsymbol{\mathcal{X}}} \geq 0$ such that for all $\boldsymbol{\mathcal{X}} \in \mathcal{S}$

a. $\boldsymbol{\mathcal{X}}$ is a random diagonal matrix.

b. $\|\mathrm{Var}\left[\boldsymbol{\mathcal{X}}\right]\|_2 = \max_{i,j} |\mathrm{Cov}\left[\boldsymbol{\mathcal{X}}_{ii}, \boldsymbol{\mathcal{X}}_{jj}\right]| \leq v$.
c. $\mathrm{e}\boldsymbol{\mathcal{X}} = e_{\boldsymbol{\mathcal{X}}} I$.
d. $|e_{\boldsymbol{\mathcal{X}}} - 1| \leq C_e v$.

We will think of v as being small. This means that each of the fault matrices $\boldsymbol{\mathcal{X}}$ is close to the identity matrix with high probability. Obviously, the model for component-wise faults introduced above satisfies these assumptions.

In the remainder of this work, we write random matrices in bold letters. If a symbol appears twice, the two occurrences represent the same random matrix and are therefore dependent. If the power of a random matrix appears, we mean the product of identically distributed independent factors.

In summary, we shall model the application of a fault-prone Jacobi smoother as

$$x_l \leftarrow x_l + \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\mathrm{pre/post})} \theta D_l^{-1} \left(b_l - A_l x_l\right),$$

which has the same form as a standard Jacobi smoother in which the iteration matrix has been replaced by a random iteration matrix

$$\boldsymbol{S}_{\boldsymbol{l}}^{(\mathrm{pre/post})} = I - \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\mathrm{pre/post})} \theta D_l^{-1} A_l.$$

Here and in what follows, $\boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\cdot)}$ are generic fault matrices. Suppose that only the calculation of the update can be faulty, and that the previous iterate is preserved. This could be achieved by writing the local components of the current iterate to non-volatile memory or saving it on an adjacent node. The matrices $\boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\mathrm{pre/post})}$ and D_l^{-1} commute, so that without loss of generality, we can assume that there is just one fault matrix, because any faults in the calculation of the residual can be included in $\boldsymbol{\mathcal{X}}_l^{(\mathrm{pre/post})}$ as well. Moreover, while the application of D_l^{-1} and A_l to a vector is fault-prone, we assume that the entries of D_l^{-1} and A_l itself are not subject to corruption, since permanent changes to them would effectively make it impossible to converge to the correct solution. The matrix entries are generally computed once and for all, and can be stored in non-volatile memory which is protected against corruption. The low writing speed of NVRAM is not an issue since the matrices are written at most once.

The fault-prone two-level method has iteration matrix

$$\boldsymbol{E}_{\boldsymbol{TG},\boldsymbol{l}}\left(\nu_1, \nu_2\right) = \left(\boldsymbol{S}_{\boldsymbol{l}}^{(\mathrm{post})}\right)^{\nu_2} \boldsymbol{C}_{\boldsymbol{l}} \left(\boldsymbol{S}_{\boldsymbol{l}}^{(\mathrm{pre})}\right)^{\nu_1},$$

where

$$\boldsymbol{C}_l = I - \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\boldsymbol{p})} p A_{l-1}^{-1} \boldsymbol{\mathcal{X}}_{\boldsymbol{l-1}}^{(\boldsymbol{r})} r \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\boldsymbol{A})} A_l.$$

Function $\boldsymbol{\mathcal{M}_l}$(*right-hand side* b_l, *initial guess* x_l)
  if $l = 0$ **then return** $A_0^{-1} x_0$ `(Exact solve on coarsest grid)`
  else
    for $i \leftarrow 1$ **to** ν_1 **do**
      $x_l \leftarrow x_l + \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\mathrm{pre})} \theta D_l^{-1} (b_l - A_l x_l)$ `(Pre-smoothing)`
    $d_{l-1} \leftarrow \boldsymbol{\mathcal{X}}_{\boldsymbol{l-1}}^{(\boldsymbol{r})} r \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\boldsymbol{A})} (b_l - A_l x_l)$ `(Restriction to coarser grid)`
    $e_{l-1}^{(0)} \leftarrow 0$
    for $j \leftarrow 1$ **to** γ **do**
      $e_{l-1}^{(j)} \leftarrow \boldsymbol{\mathcal{M}_{l-1}} \left(d_{l-1}, e_{l-1}^{(j-1)}\right)$ `(Solve on coarser grid)`
    $x_l \leftarrow x_l + \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\boldsymbol{p})} p e_{l-1}^{(\gamma)}$ `(Prolongation to finer grid)`
    for $i \leftarrow 1$ **to** ν_2 **do**
      $x_l \leftarrow x_l + \boldsymbol{\mathcal{X}}_{\boldsymbol{l}}^{(\mathrm{post})} \theta D_l^{-1} (b_l - A_l x_l)$ `(Post-smoothing)`
    return x_l

Algorithm 2: Fault-prone multi-level method $\mathcal{M}_l$

Similar arguments as for the smoother can be used to justify the model of faults for the coarse-grid correction. The fault-prone multi-level algorithm is given in Algorithm 2.

In order to illustrate the effect of the faults on the convergence of the algorithm, we apply the two-level version of Algorithm 2 with one step of pre- and post-smoothing using a damped Jacobi smoother with optimal smoothing parameter $\theta = \frac{2}{3}$ for a piecewise linear discretization of the Poisson problem on a square domain.

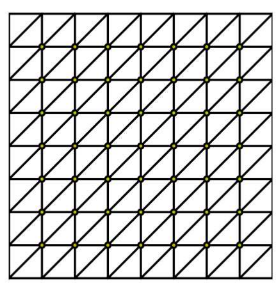

Fig. 1.1 Mesh for the square domain.

The domain is partitioned by a uniform triangulation (Figure 1.1), and we inject component-wise faults as given in eq. (1.2). We plot the evolution of the residual norm over 30 iterations for varying number of degrees of freedom n_L and different probabilities of faults q in Figure 1.2 on page 11. We can see that as q increases, the curves start to fan out, with a slope depending on the number of degrees of freedom n_L.

1.4 Summary of results on convergence

In [1, 2], a framework for the analysis of fault-prone stationary iterations was proposed. We summarize the obtained convergence results whose proofs can be found in [1, 2].

Theorem 1 ([1]). *Let $\Omega \subset \mathbb{R}^d$ with $\partial\Omega \in C^2$ or Ω convex and let A_l be the stiffness matrices associated to the finite element discretization of a second order elliptic PDE on a hierarchy of quasi-uniform meshes, and let*

$$\boldsymbol{E_{TG,L}}(\nu_1, \nu_2) = \left(\boldsymbol{S_L^{(post)}}\right)^{\nu_2} \boldsymbol{C_L} \left(\boldsymbol{S_L^{(pre)}}\right)^{\nu_1}$$

be the iteration matrix of the two-level method with component-wise faults of rate q in prolongation, restriction, residual and smoother:

$$\begin{aligned} \boldsymbol{C_L} &= I - \boldsymbol{\mathcal{X}_L^{(p)}} p A_{L-1}^{-1} \boldsymbol{\mathcal{X}_{L-1}^{(r)}} r \boldsymbol{\mathcal{X}_L^{(A)}} A_L, \\ \boldsymbol{S_L^{(pre/post)}} &= I - \boldsymbol{\mathcal{X}_L^{(pre/post)}} D_L^{-1} A_L. \end{aligned}$$

Assume that the usual conditions for multigrid convergence hold. Then the rate of convergence of the fault-prone two-level method is bounded as

$$\varrho\left(\boldsymbol{E_{TG,L}}(\nu_1, \nu_2)\right) \leq \left\| E_{TG,L}(\nu_1, \nu_2) \right\|_A + C \begin{cases} q n_L^{\frac{4-d}{2d}} & d < 4, \\ q (\log n_L)^{\frac{1}{2}} & d = 4, \\ q & d > 4, \end{cases}$$

where $E_{TG,L}$, C_L and S_L are the unperturbed two-level iteration matrix, coarse-grid correction and Jacobi smoother and $\|\cdot\|_A$ is the energy norm. C is independent of L and q.

In Figure 1.3 (top) on page 12, we plot the estimated rate of convergence of the two-level method for the 2d Poisson problem introduced above. We use 1000 iterations to estimate $\varrho\left(\boldsymbol{E_{TG,L}}(1,1)\right)$ for component-wise faults with varying probability q and varying problem size n_L. Moreover, we plot the behavior predicted by Theorem 1 and the level of $\varrho\left(\boldsymbol{E_{TG,L}}(1,1)\right) = 1$. We can see that their slope matches.

Experimentally, it can be observed that the result also holds for the case of an L-shaped domain and for block-wise faults, provided the size of the blocks is fixed, even though the conditions of Theorem 1 are not satisfied.

The above results indicate that two-level methods without protection of some components can not be used in a fault-prone environment. In order to preserve convergence independent of the number of degrees of freedom, we will have to protect one of the fault-prone operations. The cheapest operations are the restriction and the prolongation. The next result shows that the two-level method converges, if the prolongation is protected.

Theorem 2 ([1]). *Let*

$$\boldsymbol{E_{TG,L}}(\nu_1,\nu_2) = \left(\boldsymbol{S_L^{(post)}}\right)^{\nu_2} \boldsymbol{C_L} \left(\boldsymbol{S_L^{(pre)}}\right)^{\nu_1}$$

with smoother and coarse-grid correction given by

$$\begin{aligned} \boldsymbol{S_L^{(pre/post)}} &= I - \boldsymbol{\mathcal{X}_L^{(pre/post)}} D_L^{-1} A_L, \\ \boldsymbol{C_L} &= I - pA_{L-1}^{-1} \boldsymbol{\mathcal{X}_{L-1}^{(r)}} r \boldsymbol{\mathcal{X}_L^{(A)}} A_L. \end{aligned}$$

Provided the usual conditions for multigrid convergence and Assumption (A) with

$$\mathcal{S} = \left\{ \boldsymbol{\mathcal{X}_{L-1}^{(r)}}, \boldsymbol{\mathcal{X}_L^{(A)}}, \boldsymbol{\mathcal{X}_L^{(pre)}}, \boldsymbol{\mathcal{X}_L^{(post)}} \right\}$$

hold for some $v \geq 0$, we find for any level L that the fault-prone two-level method converges with a rate bounded as

$$\varrho\left(\boldsymbol{E_{TG,L}}(\nu_1,\nu_2)\right) \leq \left\| E_{TG,L}(\nu_2,\nu_1) \right\|_2 + Cv.$$

and C is independent of v and L.

We note that the result holds for more general types of faults including block-wise faults. In Figure 1.3 (bottom) on page 12, we plot the rate of convergence of the two-grid method for the already discussed example, this time with protected prolongation. We can see that the rate is essentially independent of the size of the problem, and even is smaller than one for large values of q. The protection can by achieved by standard techniques such as replication. In order to retain performance, the protected prolongation could be overlapped with the application of the post-smoother.

The following theorem shows that the result carries over to the multi-level case:

Theorem 3 ([2]). *Provided usual conditions for multigrid convergence and Assumption (A) with*

$$\mathcal{S} = \bigcup_{l=1}^{L} \left\{ \boldsymbol{\mathcal{X}_{l-1}^{(r)}}, \boldsymbol{\mathcal{X}_l^{(A)}}, \boldsymbol{\mathcal{X}_l^{(pre)}}, \boldsymbol{\mathcal{X}_l^{(post)}} \right\}$$

hold, the number of smoothing steps is sufficient and that v sufficiently small, the perturbed multi-level method converges with a rate bounded by

$$\varrho\left(\boldsymbol{E_L}(\nu_1,\nu_2,\gamma)\right) \leq \begin{cases} \frac{\gamma}{\gamma-1}\xi + Cv, & \gamma \geq 2, \\ \frac{2}{1+\sqrt{1-4C_*\xi}}\xi + Cv, & \gamma = 2, \end{cases}$$

where

$$\xi = \max_{l \leq L} \left\| E_{TG,l}(\nu_2,\nu_1) \right\|_2,$$

and C_* and C depend on ν_1, ν_2 and the convergence rate of the two-level method, but are independent of L and v.

We also plot the rate of convergence of fault prone multi-level algorithms with one coarse-grid correction for component-wise faults and protected prolongation in Figure 1.4 on page 13, and observe the predicted behavior.

In the current work, we proposed a probabilistic model for the effect of faults involving random diagonal matrices. We gave a summary of the theoretical analysis of the model for the rate of convergence of fault-prone multigrid methods which show that the standard multigrid method is not resilient. Finally, we presented a modification of the standard multigrid algorithm that is fault resilient.

References

1. Ainsworth, M., Glusa, C.: Is the Multigrid Method Fault Tolerant? The Two-Grid Case. SIAM Journal on Scientific Computing **39**(2), C116–C143 (2017). DOI 10.1137/16M1100691
2. Ainsworth, M., Glusa, C.: Is the multigrid method fault tolerant? The Multilevel Case. SIAM Journal on Scientific Computing (Accepted)
3. Avižienis, A., Laprie, J.C., Randell, B., Landwehr, C.: Basic concepts and taxonomy of dependable and secure computing. IEEE Transactions on Dependable and Secure Computing **1**(1), 11–33 (2004)
4. Bramble, J.H.: Multigrid methods, vol. 294. CRC Press (1993)
5. Cappello, F.: Fault tolerance in petascale/exascale systems: Current knowledge, challenges and research opportunities. International Journal of High Performance Computing Applications **23**(3), 212–226 (2009)
6. Cappello, F., Geist, A., Gropp, B., Kale, L., Kramer, B., Snir, M.: Toward exascale resilience. International Journal of High Performance Computing Applications **23**(4), 374–388 (2009)
7. Cappello, F., Geist, A., Gropp, W., Kale, S., Kramer, B., Snir, M.: Toward exascale resilience: 2014 update. Supercomputing frontiers and innovations **1**(1), 5–28 (2014)
8. Casas, M., de Supinski, B.R., Bronevetsky, G., Schulz, M.: Fault Resilience of the Algebraic Multi-grid Solver. In: Proceedings of the 26th ACM International Conference on Supercomputing, ICS '12, pp. 91–100. ACM, New York, NY, USA (2012). DOI 10.1145/2304576.2304590
9. Cui, T., Xu, J., Zhang, C.S.: An error-resilient redundant subspace correction method. Computing and Visualization in Science **18**(2), 65–77 (2017). DOI 10.1007/s00791-016-0270-6. URL http://dx.doi.org/10.1007/s00791-016-0270-6
10. Elliott, J., Mueller, F., Stoyanov, M., Webster, C.G.: Quantifying the impact of single bit flips on floating point arithmetic. Tech. Rep. ORNL/TM-2013/282, Oak Ridge National Laboratory (2013)
11. Glusa, C.: Multigrid and domain decomposition methods in fault-prone environments. Ph.D. thesis, Brown University (2017)
12. Göddeke, D., Altenbernd, M., Ribbrock, D.: Fault-tolerant finite-element multigrid algorithms with hierarchically compressed asynchronous checkpointing. Parallel Computing **49**, 117–135 (2015)
13. Hackbusch, W.: Multi-grid methods and applications, vol. 4. Springer-Verlag Berlin (1985). DOI 10.1007/978-3-662-02427-0

14. Hackbusch, W.: Iterative solution of large sparse systems of equations, *Applied Mathematical Sciences*, vol. 95. Springer-Verlag, New York (1994). DOI 10.1007/978-1-4612-4288-8
15. Herault, T., Robert, Y.: Fault-Tolerance Techniques for High-Performance Computing. Springer International Publishing (2015). DOI 10.1007/978-3-319-20943-2
16. Huang, K.H., Abraham, J.: Algorithm-based fault tolerance for matrix operations. Computers, IEEE Transactions on **100**(6), 518–528 (1984)
17. Huber, M., Gmeiner, B., Rude, U., Wohlmuth, B.: Resilience for massively parallel multigrid solvers. SIAM Journal on Scientific Computing **38**(5), S217–S239 (2016)
18. McCormick, S.F., Briggs, W.L., Henson, V.E.: A multigrid tutorial. SIAM, Philadelphia (2000)
19. Shantharam, M., Srinivasmurthy, S., Raghavan, P.: Characterizing the Impact of Soft Errors on Iterative Methods in Scientific Computing. In: Proceedings of the International Conference on Supercomputing, ICS '11, pp. 152–161. ACM, New York, NY, USA (2011). DOI 10.1145/1995896.1995922
20. Sloan, J., Kumar, R., Bronevetsky, G.: Algorithmic approaches to low overhead fault detection for sparse linear algebra. In: Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on, pp. 1–12. IEEE, Boston, MA, USA (2012)
21. Snir, M., Wisniewski, R.W., Abraham, J.A., Adve, S.V., Bagchi, S., Balaji, P., Belak, J., Bose, P., Cappello, F., Carlson, B., et al.: Addressing failures in exascale computing. International Journal of High Performance Computing Applications **28**(2), 129–173 (2014)
22. Stoyanov, M., Webster, C.: Numerical Analysis of Fixed Point Algorithms in the Presence of Hardware Faults. SIAM Journal on Scientific Computing **37**(5), C532–C553 (2015). DOI 10.1137/140991406
23. Trottenberg, U., Oosterlee, C.W., Schüller, A.: Multigrid. Academic Press Inc., San Diego, CA (2001). With contributions by A. Brandt, P. Oswald and K. Stüben

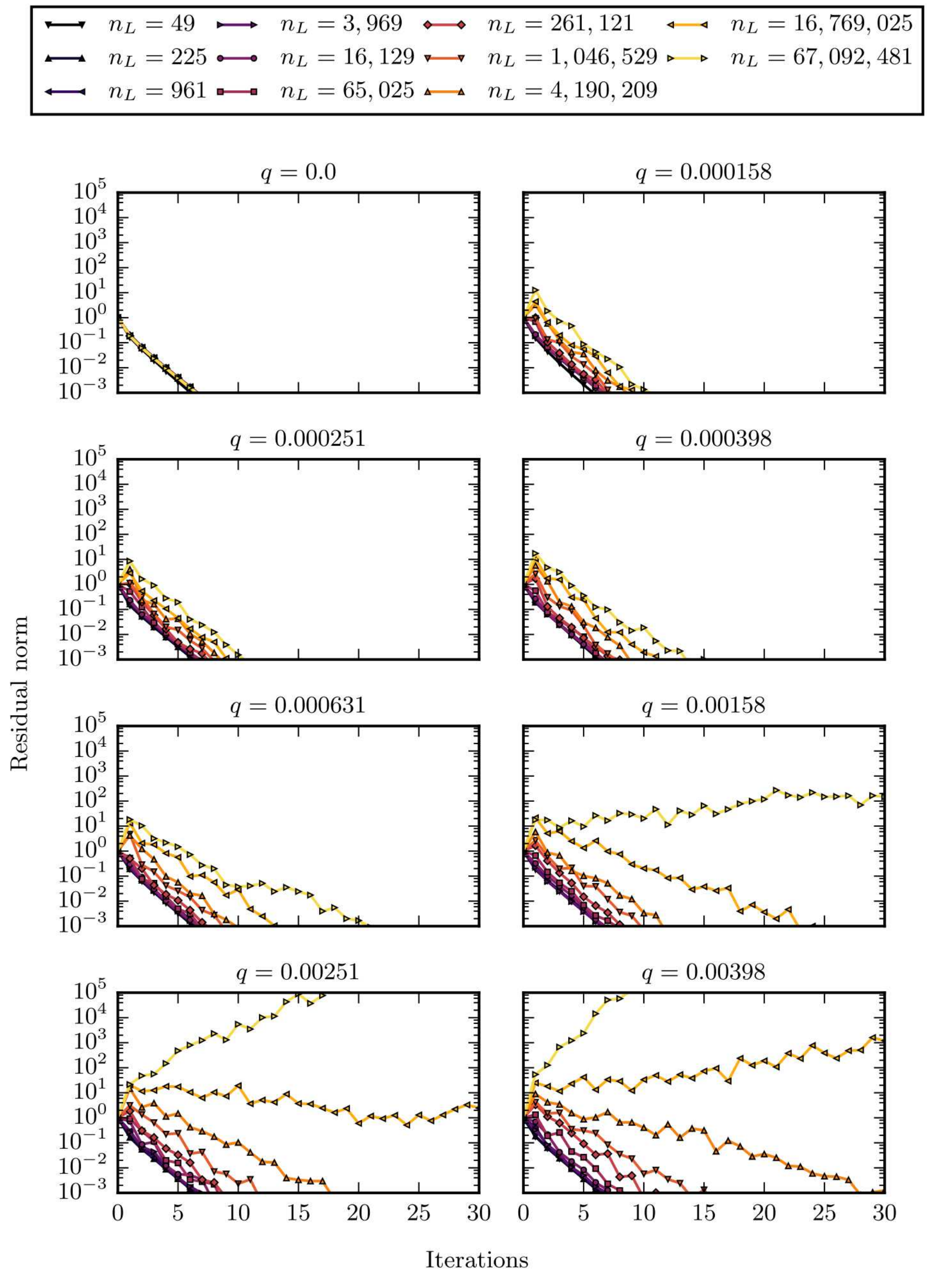

Fig. 1.2 Evolution of the norm of the residual of the two-level method for the 2d Poisson problem on square domain and component-wise faults in prolongation, restriction, residual and smoother.

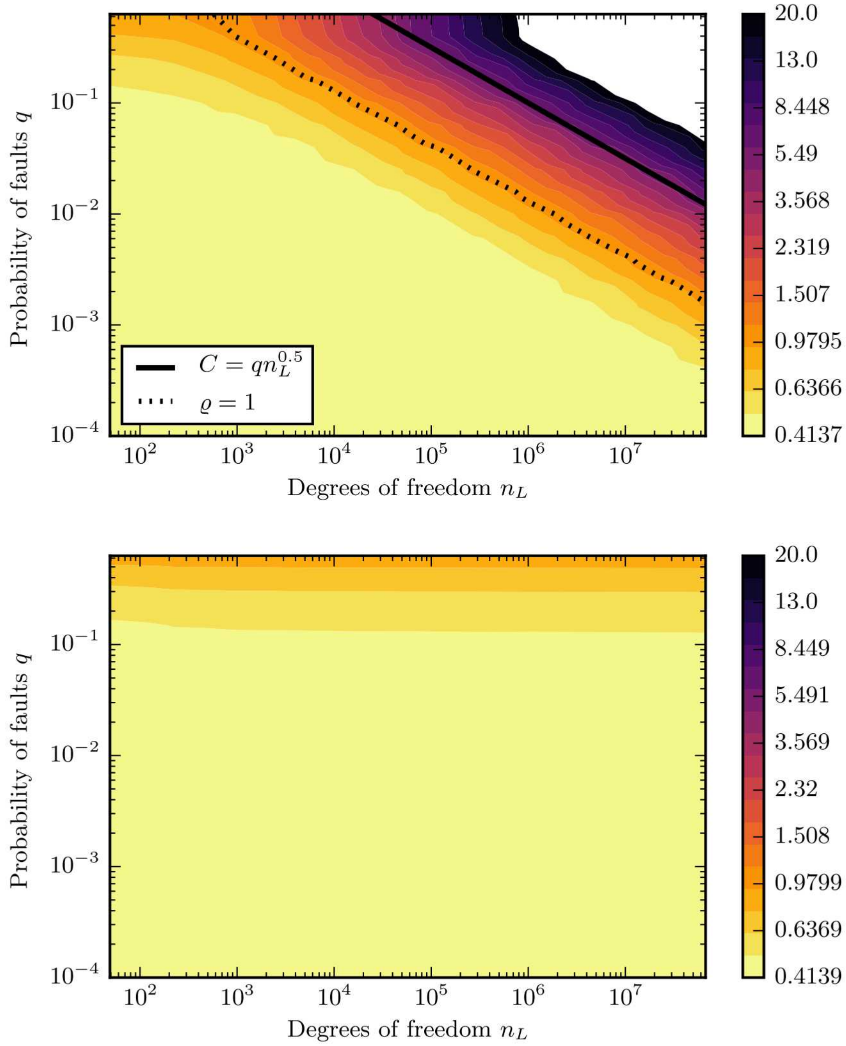

Fig. 1.3 Asymptotic convergence rate $\varrho\left(\boldsymbol{E_{TG,L}}(1,1)\right)$ of the fault-prone two-level method for the 2d Poisson problem on square domain with component-wise faults in prolongation, restriction, residual and smoother (top) and protected prolongation (bottom).

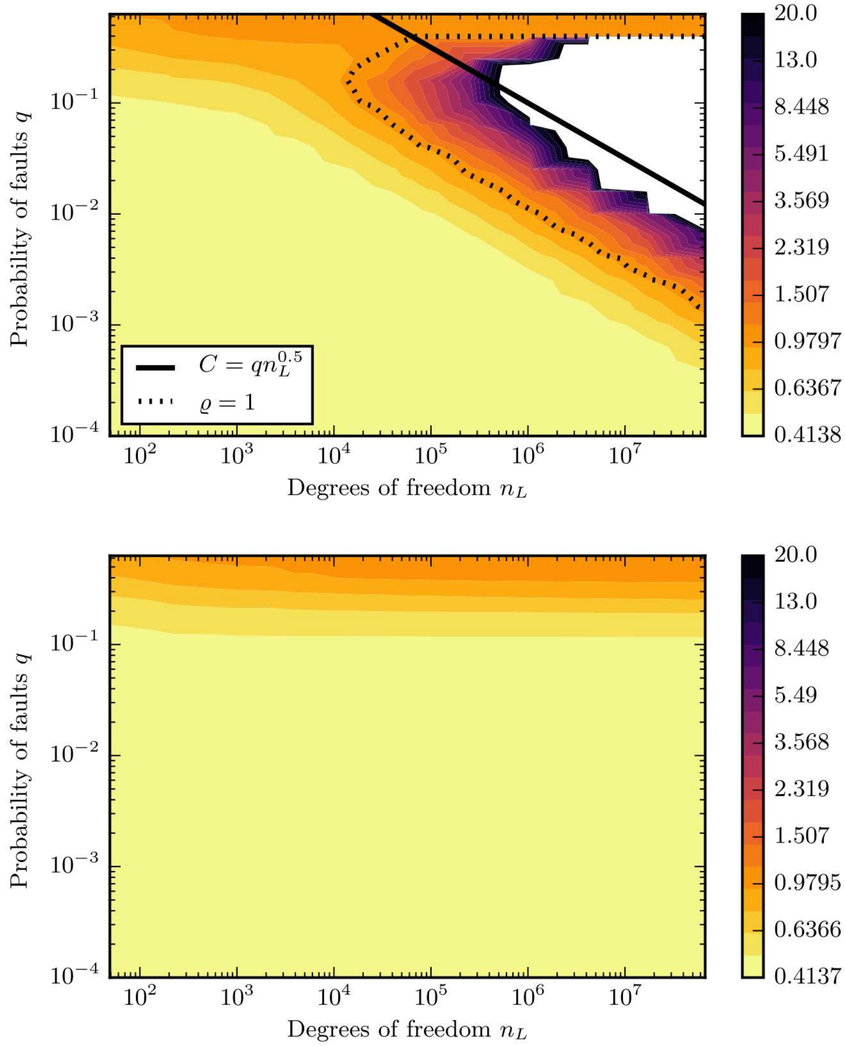

Fig. 1.4 Asymptotic convergence rate $\varrho\,(\boldsymbol{E_L}(1,1,2))$ of the fault-prone multi-level method for the 2d Poisson problem on square domain with component-wise faults in prolongation, restriction, residual and smoother (top) and protected prolongation (bottom).