Resilient geometric finite-element multigrid algorithms using minimised checkpointing

Dominik Göddeke, Mirco Altenbernd, Dirk Ribbrock

Institut für Angewandte Mathematik (LS3)
Fakultät für Mathematik
TU Dortmund
dominik.goeddeke@math.tu-dortmund.de

Parallel Matrix Algorithms and Applications 2014
Lugano, Switzerland, July 2, 2014

technische universität dortmund
fakultät für mathematik

Wrap-up of this mini-symposium

- 'Why and how' of faults and failures
- Asynchronous algorithms
- Synchronisation-avoiding algorithms
- Latency hiding
- Iterative solvers, preconditioners
- User level failure mitigation in future MPI
- Organiser benefit: first and last slot
- Saves me the trouble to explain the general classification of this talk

Scope of this talk

Larger setting of our work

- (Geometric) multigrid and finite element discretisations
- Important building block in PDE solvers, for all kinds of quasi-elliptic (sub-)problems

Standard assumption

- Assume problem data (matrix and rhs) never affected by faults
- Consider soft, transient and persistent faults

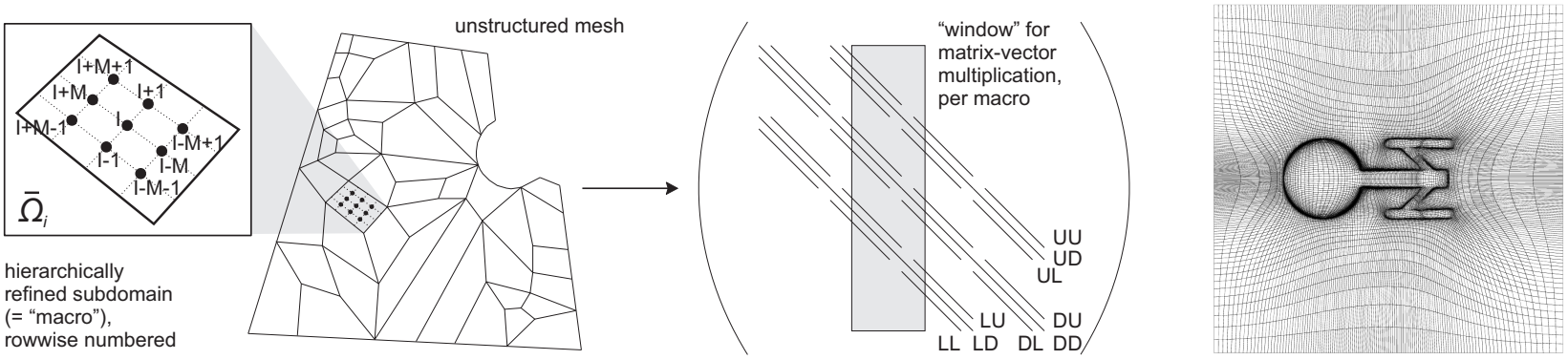
Intermediate goals of our project

- Understand multigrid convergence behaviour in case of faults
- Aim for ABFT and C/R techniques with small overhead and no impact on fault-free case
- Derive a technique that works for all fault types

FEAST

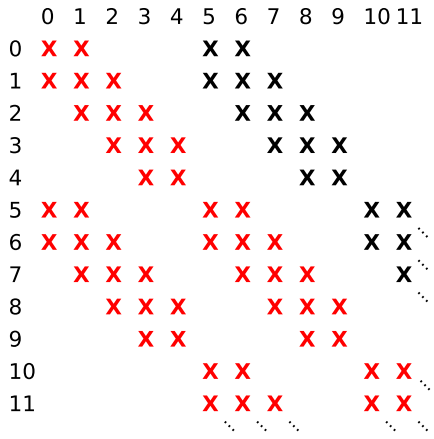
Parallel multilevel solvers

Globally unstructured locally structured meshes



- Unstructured macro-mesh for geometric flexibility
- Structured reefinement of each patch for good hardware exploitation
- Row-wise local numbering: banded local matrices, high locality
- Global operations $\rightarrow$ sequences of local operations
- Fully variable coefficients or stencils

Parallel multigrid in FEAST



global Krylov
preconditioned by
global MG (V 1+1)
additively smoothed by
for all Ω_i: local **ADI-TRI-GS**

- Weighted loadbalancing: m patches to $p < m$ MPI ranks
- Multigrid essentially straight forward, sequences of local operations
- Replace global smoother by block-Jacobi patchwise smoother
- Exploit structure to design strong local smoothers

Fault scenarios in this project

Persistent and transient faults

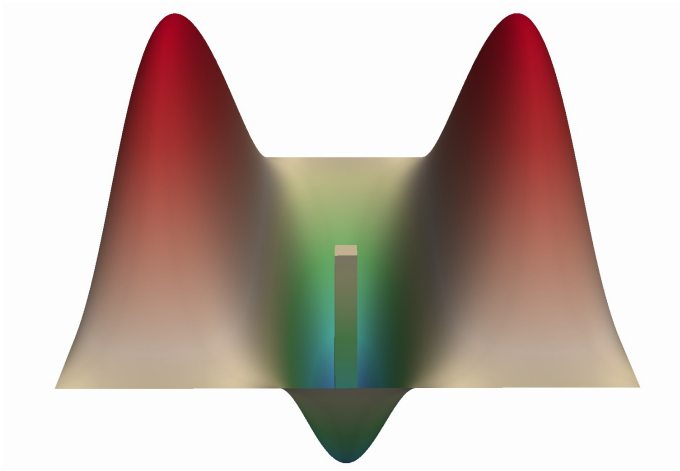
- Nodes die and get replaced during solver iterations
- Contiguous portions of global solution corresponding to patches lost
- Bitflips in the iterate: tiny localised disturbance

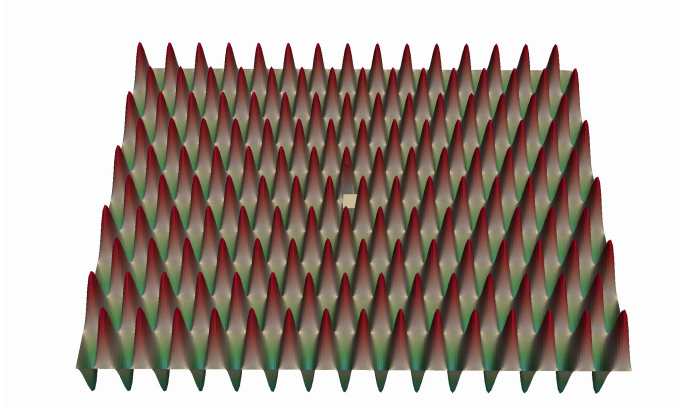
Current stage of the project

- Interested only in numerical behaviour
- Exploit h and H-independence of multigrid
- Scale down: node failure and patch loss $\rightarrow$ elimination of a few neighbouring element layers and DOF
- Design schemes with scalable parallel implementation in mind

Robustness of multigrid

Experimental setup and fault injection





- Poisson problem $-\Delta u = f$ on $\Omega = [0,1]^2$, Dirichlet BCs
- Conforming biquadratic finite elements, 1 M DOF
- V cycle multigrid, relative residual reduction convergence control
- Smooth problem: $f = -\Delta(\sin(\pi x)\sin(3\pi y))$
- Non-smooth problem: $f = -\Delta(\sin(10\pi x)\sin(30\pi y))$
- Fault injection into some patch of fine grid iterate (figure: into converged solution, with different elevation magnifier for clarity)

Experimental parameter space pruning

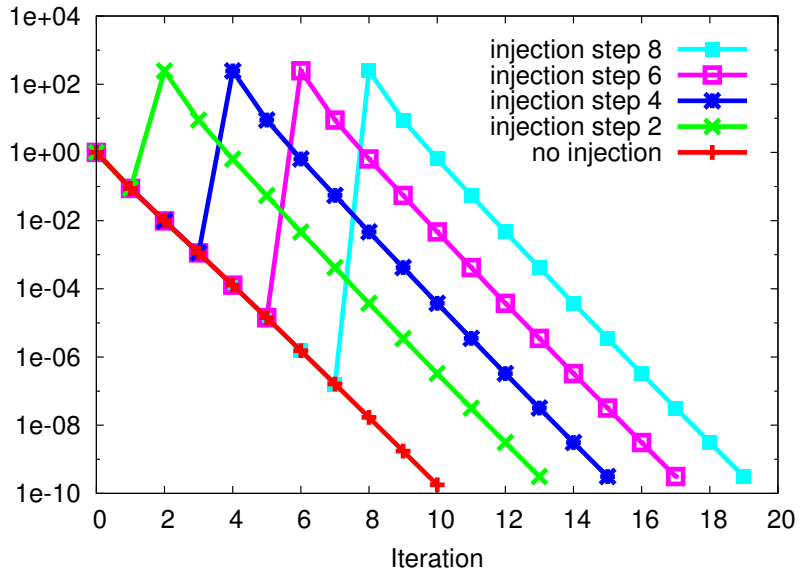
Qualitatively identical and quantitatively comparable behaviour

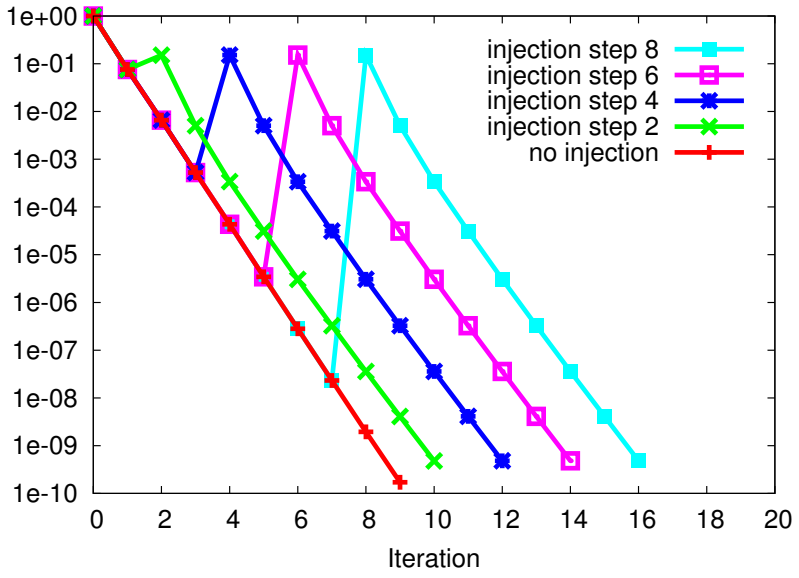
- Elimination of 0.01,. . . ,10 % of all values
- Same behaviour for well-behaved bitflips
- Zeroing, random values in $-2 \max |u|, \ldots, 2 \max |u|$, sign flip
- Fine-grid residuals instead of iterates $\Leftrightarrow$ slightly larger fault in fine-grid iterate with one iteration delay (initial residual is data)
- Consequence: same behaviour for restricted residuals and prolongated corrections on coarser levels

Sufficient: consider faulty values in fine grid iterates only

- Resulting free parameters: point in time (iteration number), location (in domain), frequency (single, every i-th iteration)

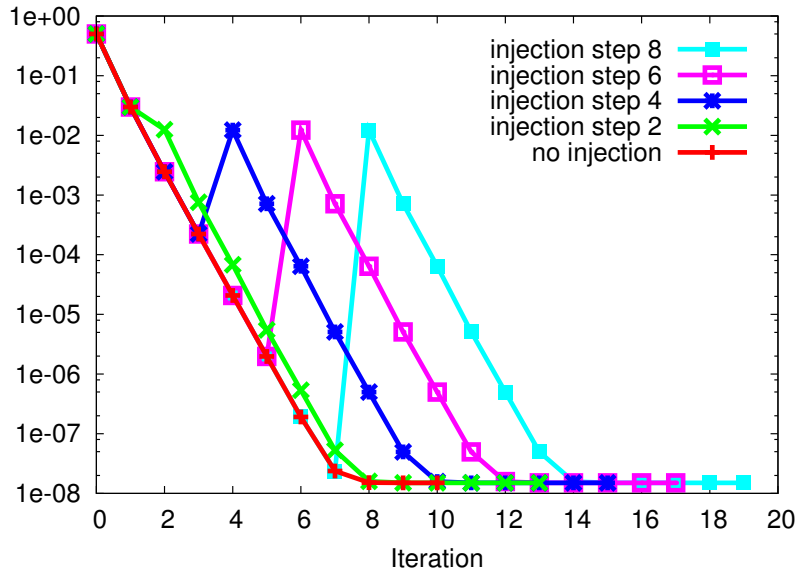
Single fault injection at different iterations

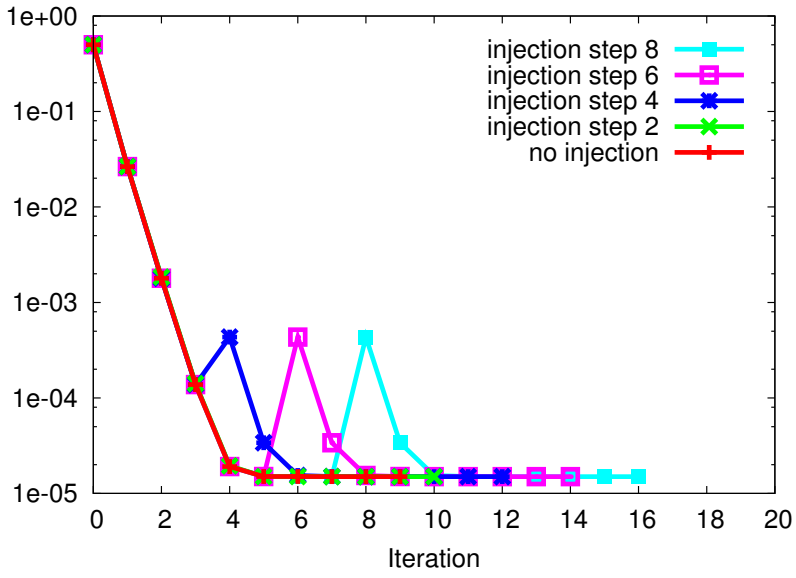




- Convergence behaviour (residuals), smooth and non-smooth problem
- MG always converges despite fault injection, at most 2x iterations
- Large jump: infinite / discontinuous curvature at ‘fault boundary’
- Analogy: global MG restart with much worse initial guess

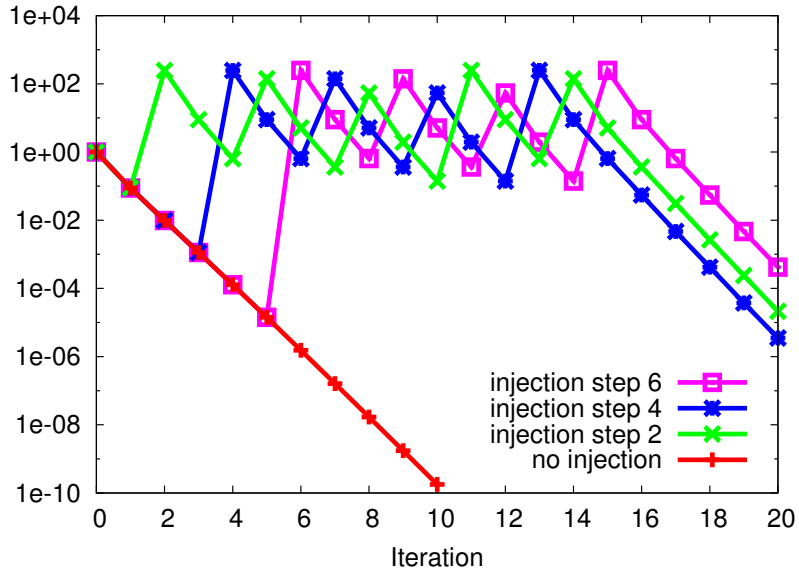
Single fault injection at different iterations

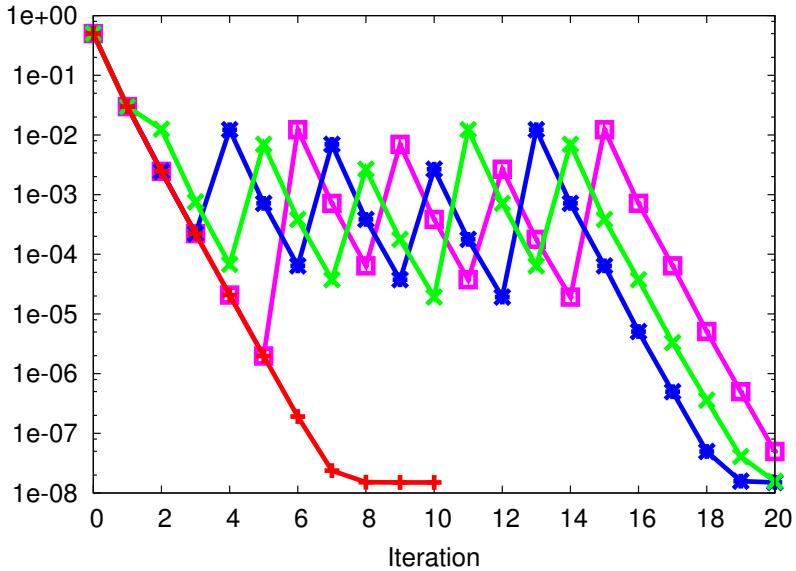




- Convergence behaviour (L_2 error), smooth and non-smooth problem
- Impact only visible if fault dominates the L_2 error
- Less extra work compared to residual-based convergence control
- Use some error estimation for convergence control?

Repeated fault injection at different locations





- Smooth problem: residuals (left) and L_2 errors (right), fault injection at alternating locations after every third iteration
- Good: MG converges nonetheless
- Bad: MG only converges after fault injection has ended

Resilient multigrid

with minimised checkpointing

Resilient multigrid with minimised checkpointing

Classical and proposed avenues towards built-in resilience

- Global C/R: too slow, data volume too high
- Redundancy and voting: too much energy
- Both overkill for linearised subproblem in real simulations
- ABFT-checksums: no $O(n^2)$ over $O(n)$ benefit to hide overhead

Our approach: explicitly exploit existing multigrid hierarchy

- Checkpoint (to memory): store last iterate on a coarser scale
- Restart (from memory): prolongate backup solution to fine scale
- Exploit exponentially decreasing data, comm and comp volume
- Conforming FEM: $2^d\times$ savings per refinement level (!)

Resilient multigrid with minimised checkpointing

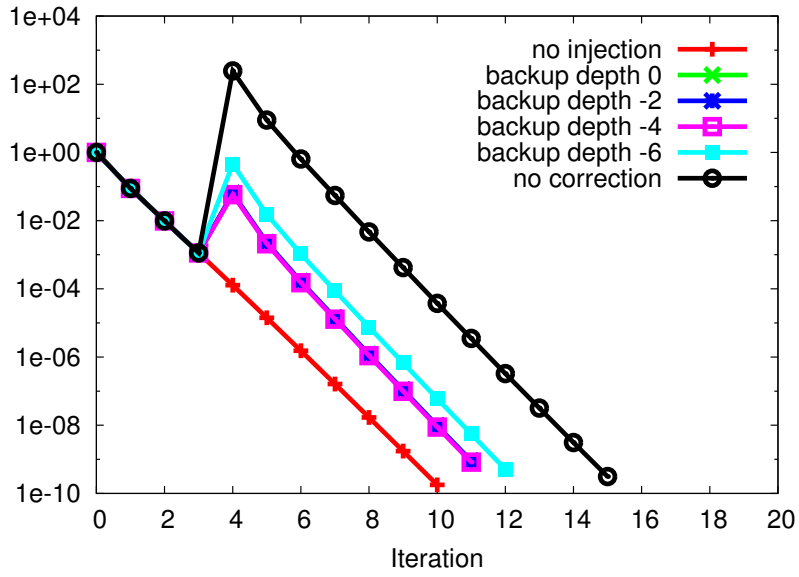
Checkpoint-to-memory

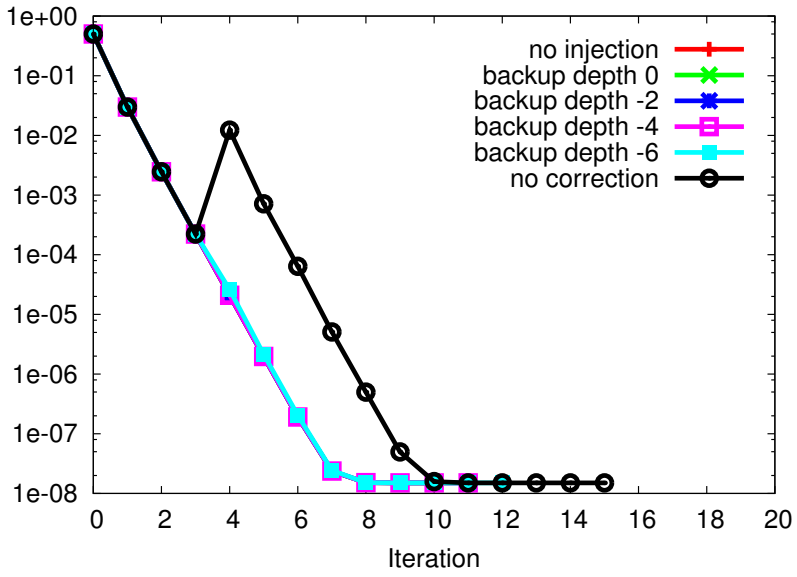
- Restriction of iterate can be done completely asynchroneously
- Extra work: down cycle without smoothing
- `MPI_Isend()` or `MPI_Put()` to some other node
- Fault-free performance barely impacted

Restart-from-memory

- Local prolongation on 'backup rank' or replacement node
- Extra work: up-cycle without smoothing
- `MPI_Irecv()`, `MPI_Isend()`, or one-sided communication
- Implies P2P load imbalance instead of global sync as in C/R
- Current project stage: numerical 'quality' of the local checkpoint?

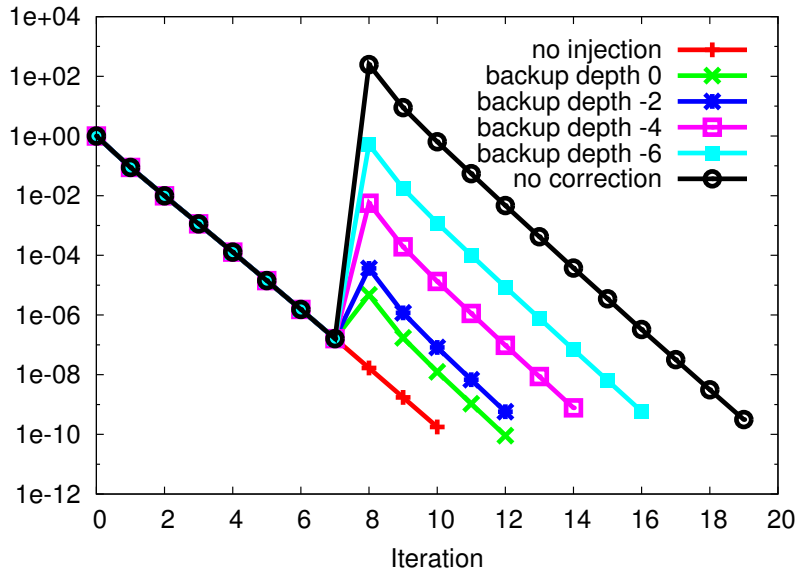
Early single fault injection and local repair

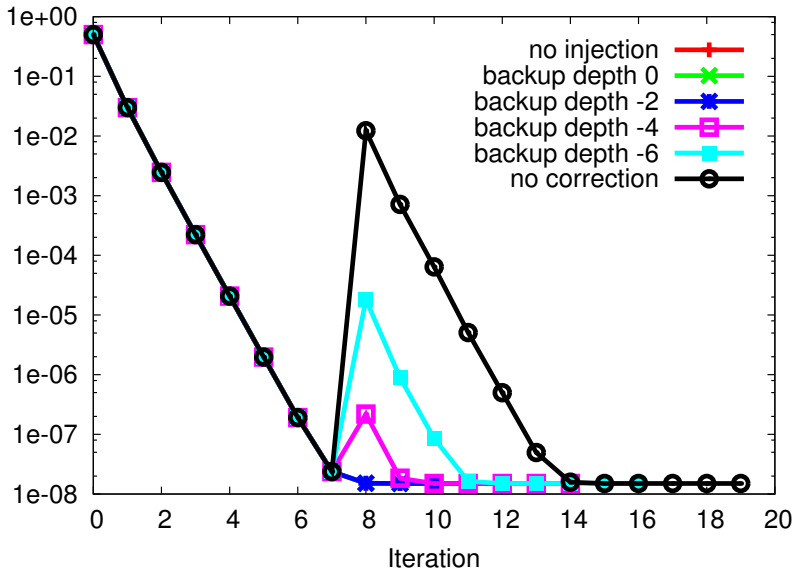




- Smooth problem: residuals (left) and L_2 errors (right)
- Same behaviour for non-smooth problem
- Cyan plot corresponds to 4096x smaller checkpoint (!)
- 0–2 extra iterations at most, up to 2x uncorrected
- No jumps in L_2 error at all

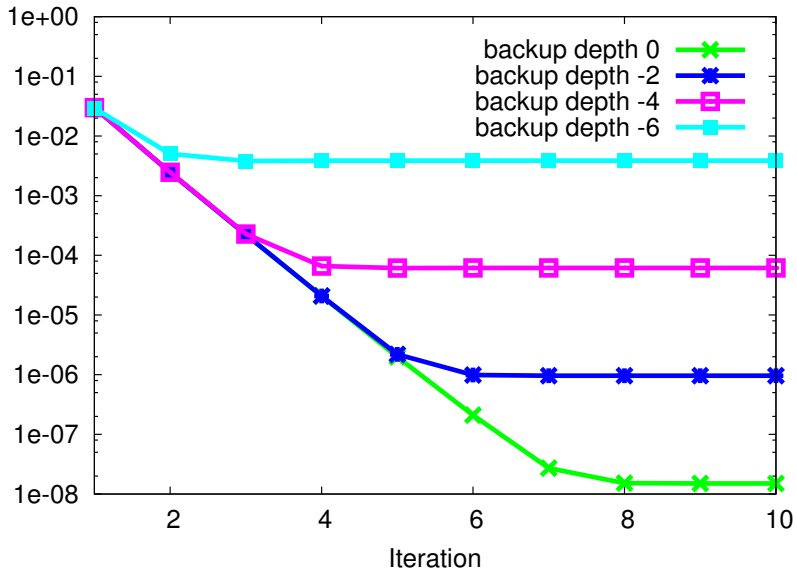
Late single fault injection and local repair

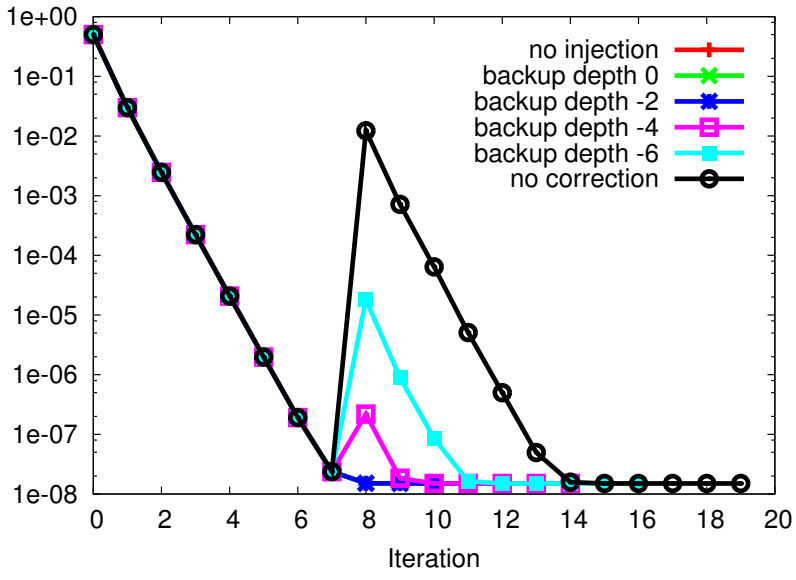




- Smooth problem: residuals (left) and L_2 errors (right)
- Fault injection into already converged solution in L_2 sense
- Moderate backup depth: same behaviour as before
- Large backup depth: no further restauration of convergence, explained on next slide

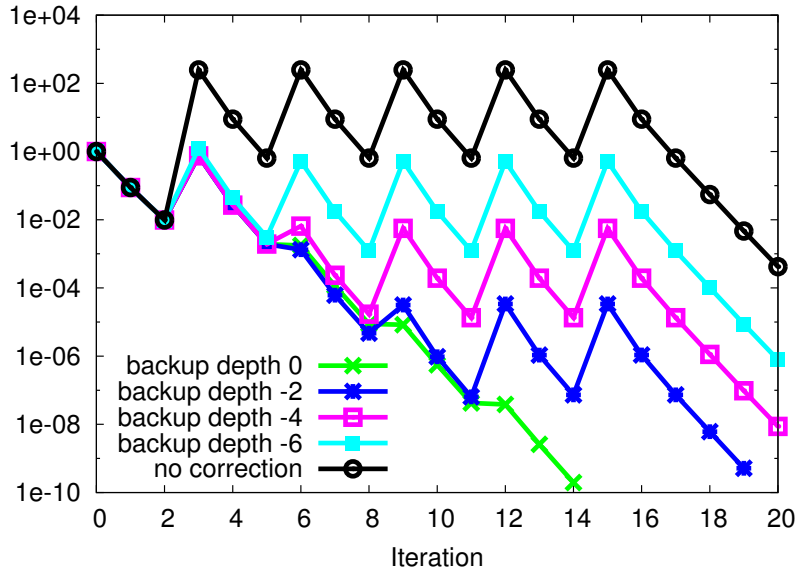
Quality of the backup in L_2 sense

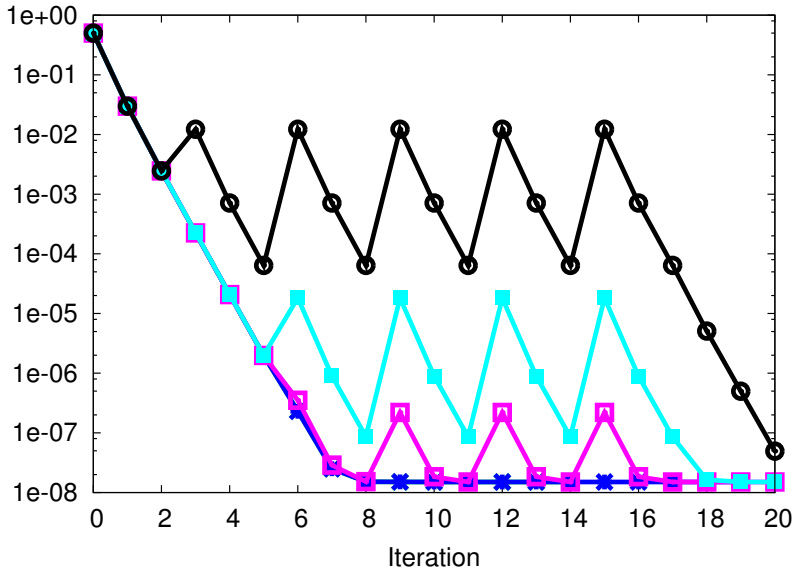




- Left: L_2 quality of backup at different iterations
- Right: convergence in L_2 norm with fault injection and repair
- Compressed backup of solution does not improve at some point
- Jump in L_2 error relates to L_2 error of backup, weighted with size of fault injection

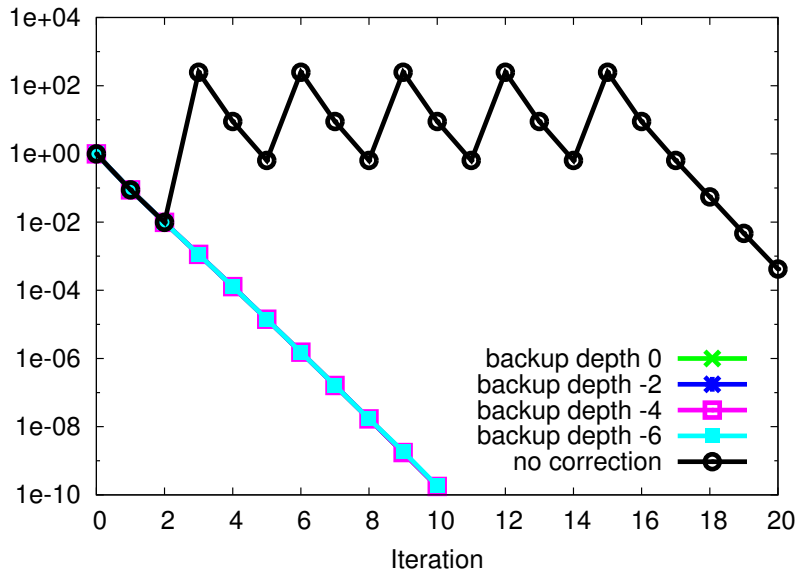
Recurrent fault injection and local repair

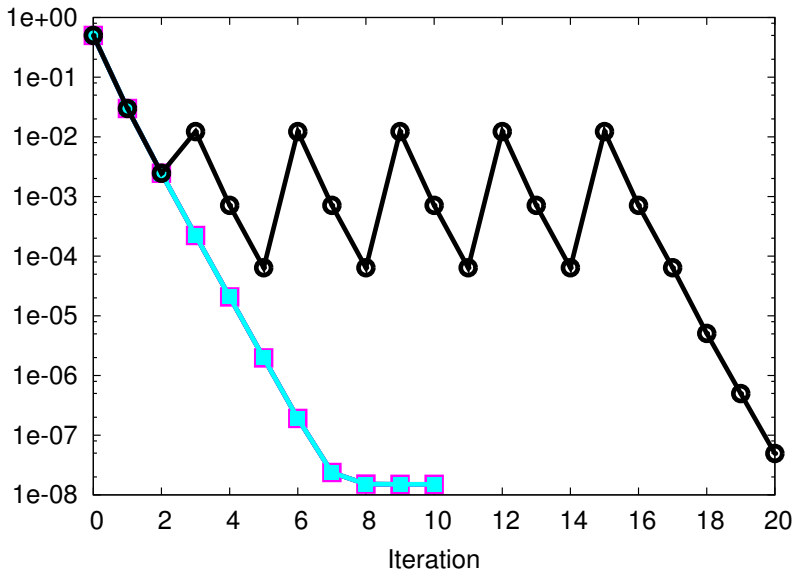




- Smooth problem: residuals (left) and L_2 errors (right)
- Same explanation as before: stagnation of backup quality in L_2 sense
- Not enough recovery from the jumps during fault-free cycles
- No convergence in residuals until fault injection ends
- L_2 convergence for moderate compression

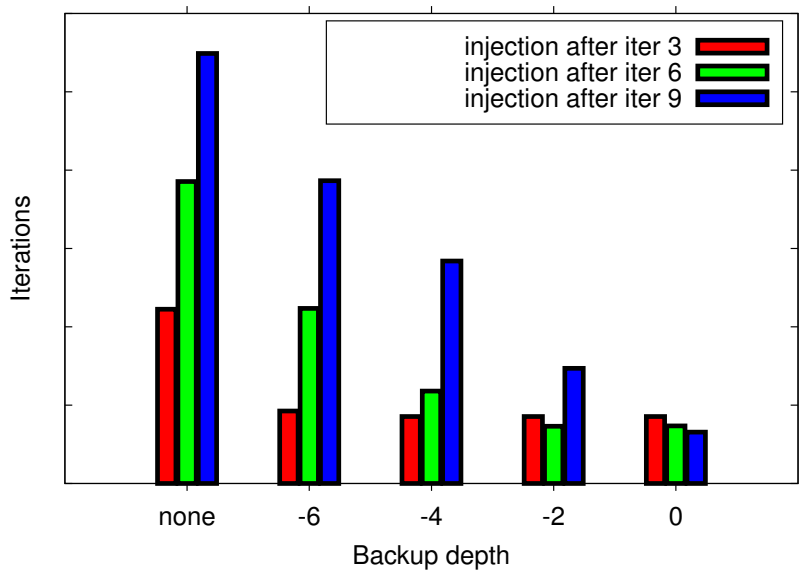
Local auxiliary solve instead of repair





- Smooth problem: residuals (left) and L_2 errors (right)
- Auxiliary solve only neccessary for recurrent fault injections
- Solve auxiliary problem on lost patch using Dirichlet data from neighbouring patches as BC
- Convergence perfectly restored
- Only applying the smoother insufficient, results similar to last slide
- Must solve exactly (same criterion as outer solve) for convergence

Combined approach for recurrent faults



- Solving auxiliary problem contradicts our approach, seemingly
- But: we can use the backup as an initial guess (!)
- 1.5x–4x less local iterations depending on backup depth
- Combined approach especially beneficial when either one alone fails

Summary

Future work

Summary

- Multigrid is surprisingly robustness for single faults
- Different fault locations and types lead to same convergence behaviour
- Exploit grid hierarchy and FEM interpolation to construct a strongly ‘compressed’ backup solution
- Perturbed iterate can be restored by compressed backup of previous iterate
- Numerical quality depends on dominating error component (discretisation error, discretisation error on backup level, etc.)
- Restoration further improved by solving local auxiliary problems

Future work and acknowledgements

Future work

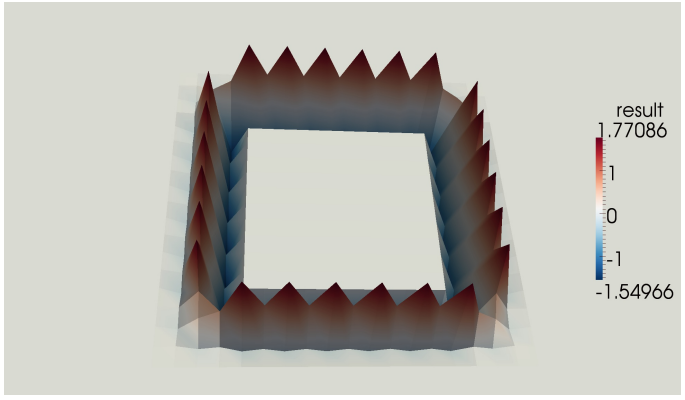
- Error predictors, observed behaviour much more meaningful in L_2
- Tradeoff: backup depth vs. effort
- Combination with async solvers, other patches can make progress towards solution
- Criteria to switch between strategies
- Relax standard assumption
- Parallel implementation

Acknowledgements

- Mercator Research Center Ruhr, grant AN-2013-0019
- DFG Priority Program 1648 *'Software for Exascale Computing'*

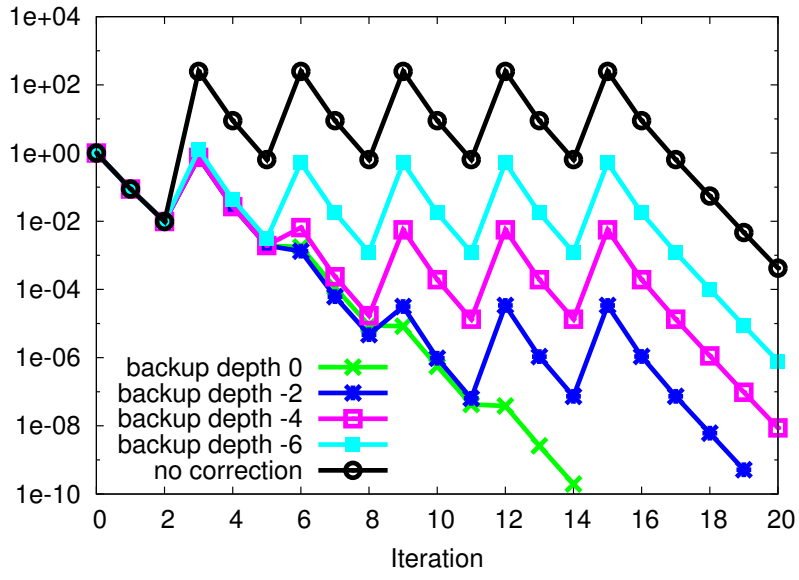
Backup slides

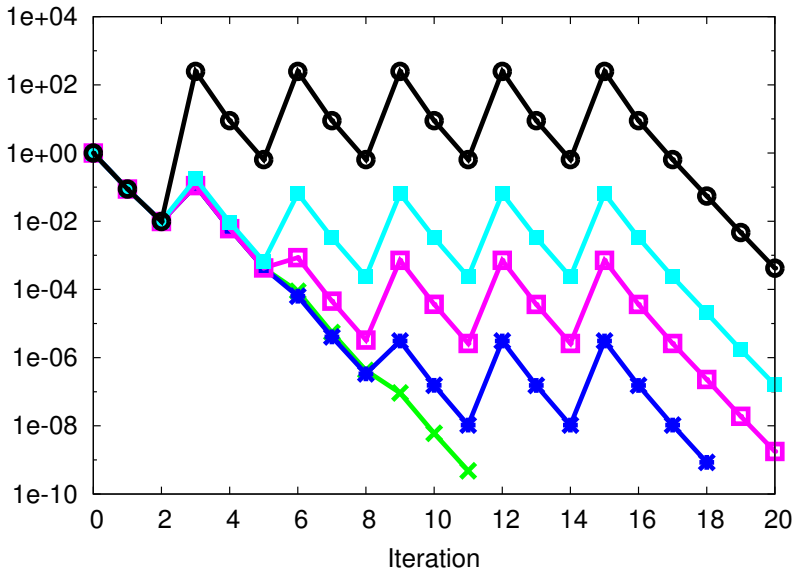
Large jumps to same plateau (without correction)



- Zoom on residual, computed immediately after fault injection into iterate
- Laplace operator in essence 2nd derivatives (curvature)
- Two ‘infinite curvature’ (discontinuous) jumps along the ‘fault boundary’
- FEM translates them into under- and overshoots

Improved restoration by auxiliary solves





- Residuals, smooth problem
- Left: local repair as above
- Right: solving local auxiliary problem up to a 2 digit improvement
- No significantly improved solution quality, still no converge until fault injection ends