Self-stabilizing Iterative Solvers

Piyush Sao, Richard Vuduc

School of Computational Science & Engineering
Georgia Institute of Technology

SIAM PP-14

Self-stabilization

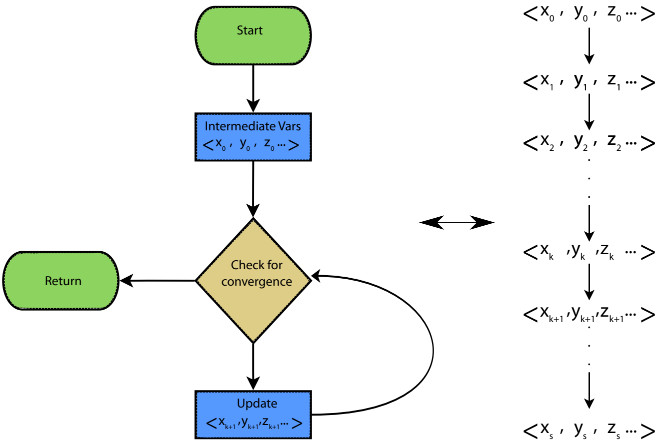
Informally, **self-stabilization** (Dijkstra 1974) is a property of a system that guarantees it will enter a valid state no matter what its initial state is.

- We describe a self-stabilizing version the conjugate gradients method, which is resilient to transient soft faults.

Fault tolerant Iterative Algorithms

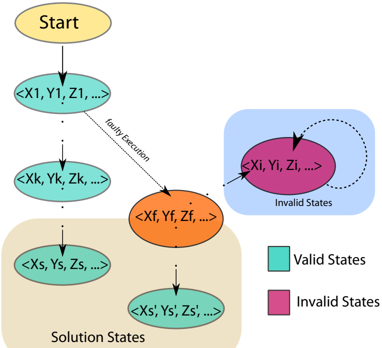
- Can an iterative algorithm still converge if a fault has occured ?

Iterative Algorithms



Self-stabilizing Algorithms

- An algorithm is *self-stabilizing*, if starting from any state (valid or invalid), it comes back to a valid state within finite number of "steps" , otherwise not.



Self-stabilizing Algorithms

Making an Algorithm Self-stabilizing

- Naturally self-stabilizing (e.g., Newton, SOR, Jacobi)
- Restart from a checkpoint
- Restart (such as restarted-GMRES)
- *Our strategy: Correction step*

Periodic correction step

- Restore sufficient conditions for convergence
- Mathematically “equivalent” to original in a fault-free execution
- Eliminates need for detecting faults
- Executing correction step periodically ensures resuming correct behavior in finite number of steps

Conjugate Gradient Algorithm

- Solve $Ax = b$ for x for SPD A;
- Quadratic optimization problem

$$F(x) = \frac{1}{2}x^T Ax - x^T b$$

- $F(x)$ represents N-dimensional paraboloid
- CG finds the optimum by taking appropriately constructed steps

Conjugate Gradient (CG) Algorithm

State variables

- x_k = present estimate
- p_k = search direction
- $r_k = b - Ax_k$ = direction of steepest descent

Transition function

$$
\begin{array}{llll}
1. & q_k & \leftarrow & Ap_k \\
2. & \alpha_k & \leftarrow & \frac{r^T r}{p^T q} \\
3. & x_{k+1} & \leftarrow & x_k + \alpha_k r_k \\
4. & r_{k+1} & \leftarrow & r_k - \alpha_k q_k \\
5. & \beta_k & \leftarrow & \frac{\|r_{k+1}\|^2}{\|r_k\|^2} \\
6. & p_{k+1} & \leftarrow & r_{k+1} + \beta_k p_k
\end{array}
$$

Self-stabilizing Conjugate Gradient

- It is a Krylov subspace method, $\{p_k\}$, $\{r_k\}$ spans Krylov subspace $\mathcal{K}(A, r_0, m)$

$$\mathcal{K}(A, r_0, m) = span\{r_0, Ar_0, \ldots A^{m-1} r_0\}$$

- Global orthogonality properties

$$\begin{aligned} p_i^T A p_j &= 0 \quad \text{if } i \neq j; \\ r_i^T r_j &= 0 \quad \text{if } i \neq j; \text{ and} \\ r_i^T p_j &= 0 \quad \text{if } i > j. \end{aligned}$$

- Finite termination in exact arithmetic

Effects of faults on Conjugate Gradient

- In general, most of Krylov subspace properties are lost
- Multiple potential outcomes due to faults
  1. Error in $r_k \Rightarrow$ Converge to incorrect value
  2. Error in $p_k \Rightarrow$ Diverge, stagnation, slow convergence
- Difficult to detect validity of state

Self-stabilizing Conjugate Gradient

We identify the following relations that are sufficient to guarentee convergence (a corollary to Zoutendijk condition)

- Residual condition : $r_k = b - Ax_k$
- Optimal step length : $\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k}$
- Correct search direction : $\frac{\left(p_k^T r_k\right)}{\|p_k\|\|r_k\|} > c_1$
- Local orthogonality relation : $p_{k+1}{}^T A p_k = 0$

Experiments

- Assume: *selective reliability mode*, i.e., correction step can be done reliably
- Inject faults in sparse matrix-vector (SpMV) product by flipping bits in matrix entry at a specified rate
  1. Bit flips in mantissa and sign bits - 40 bit flips in every unreliable SpMV
  2. Bit flips can occur any where (including exponent) - 4 bit flips in every unreliable SpMV

Problems

- We test self-stabilizing CG (CG-SS) on three problems with different convergence profiles and conditioning

Name	N	NNZ	$\kappa(A)$	Convergence profile
K3D	27000	183600	646	Quadratic
DIAG	10000	10000	990100	Linear
THERMAL1	82654	574458	496250	Sub-linear

Table : Different problems used for experimentation

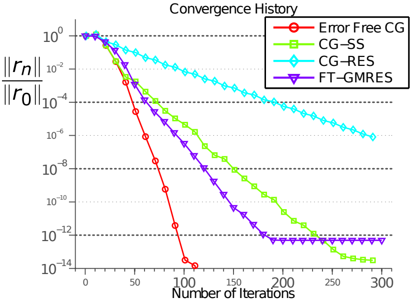
Solvers

- We compare performance of CG-SS against following solvers

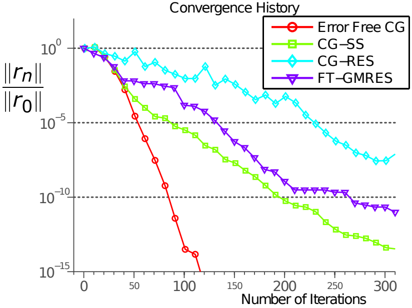
  1. *Reliable-CG* : Where all the computations are done reliably
  2. *CG-SS* : Self-stabilizing CG with correction done every 10^{th} iteration
  3. *CG-RES* : Restarted CG: restart every 10^{th} iteration
  4. *FT-GMRES* : Inner outer iteration based fault tolerant GMRES, where outer iteration is done reliably

K3D : Quadratic Convergence

- In presence of faults, only linear convergence is observed



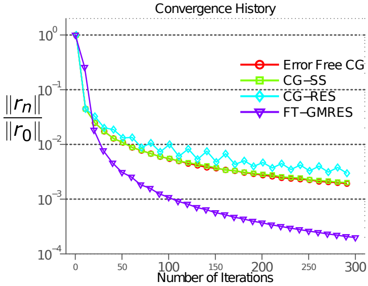
(a) Bounded errors (mantissa and sign bit flips only)



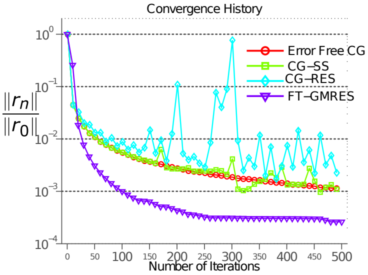
(b) Unbounded errors (including exponent also)

THERMAL1 : Sub-linear Convergence

- Convergence rate for CG-SS and CG-RES does not change by much; FT-GMRES shows better convergence due to pre-conditioning



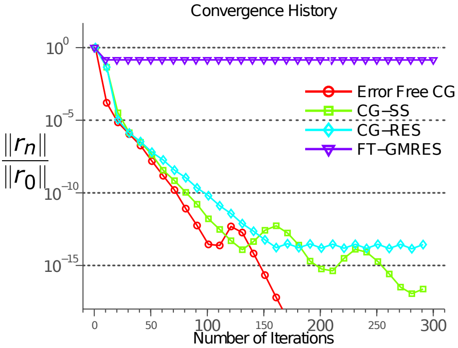
(a) Bounded errors (mantissa and sign bit flips only)



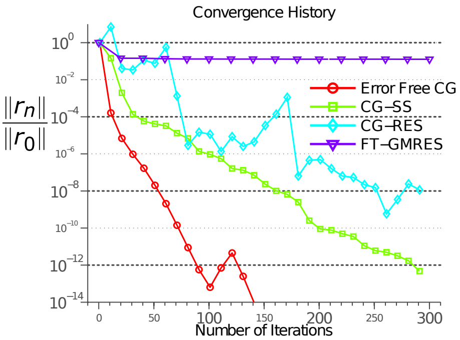
(b) Unbounded errors (including exponent also)

DIAG : Linear Convergence

- Linear convergence is maintained. However, slight slow-down in convergence is observed



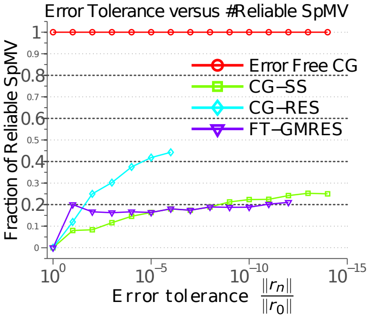
(a) Bounded errors (mantissa and sign bit flips only)



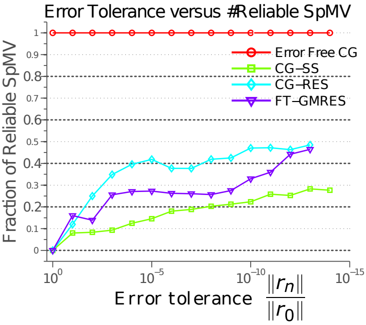
(b) Unbounded errors (including exponent also)

Amount of Reliable Computation Required

- Compared to Reliable-CG, CG-SS requires <30 % reliable SpMV to reach same error tolerance



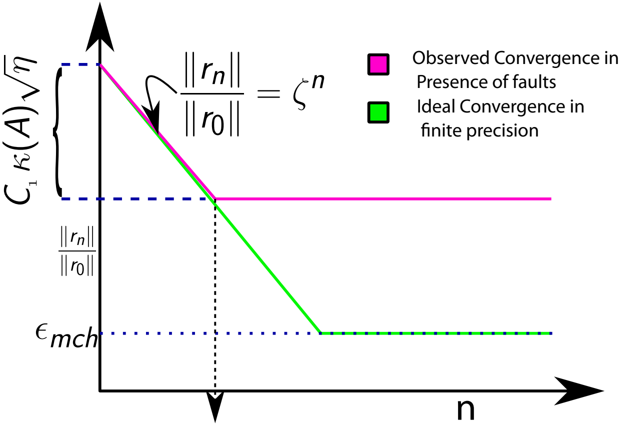
(a) Bounded errors (mantissa and sign bit flips only)



(b) Unbounded errors (including exponent also)

Analysis

- Can show that if $\kappa(A)$ is the condition number and η is rate of bit flips per SpMV, then



$$i_{opt} = log_{\zeta}\left(C_1 \kappa(A)\sqrt{\eta}\right)$$

Conclusion & Future Work

Conclusion

- Two mathematically equivalent algorithms can have different resilience. Concept of self-stabilization may give hints
- Self-stabilized CG tolerates high rates of faults
  1. Trade off between reliable and unreliable computation
  2. Reliable computation varies logarithmically with fault rates $\implies$ Scalable

Future Work

- The balance between reliable and unreliable computation : to optimize for energy / power / time
- Hybridized fault-tolerance techniques