Forum Teratec 2017
June 28, 2017
Palaiseau, France

A FEW CHALLENGES REVISITED AT SCALE

parallel performance v.s. attainable accuracy
robustness to soft errors

L. Giraud

joint work with

E. Agullo (Inria), S. Cools (Antwerpen Univ.), W. Vanroose (Antwerpen Univ.), F. Yetkin (Istanbul Kemerburgaz Univ.)

HiePACS - Inria Project
Inria Bordeaux Sud-Ouest

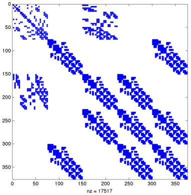
Forewords

HiePACS objectives: Contribute to the design of effective tools for frontier simulations arising from challenging research and industrial multi-scale applications towards extreme computing

- Study and design of novel numerical algorithms for emerging computing platforms
- Analyse their possible weaknesses and possible remedies

Scientific context: numerical linear algebra

Goal: solving $Ax = b$, where A is sparse



Appears in many academic and industrial simulation codes for various engineering applications: accelerator physics, chemical process simulations, earth and environmental sciences, fluid flow, fusion energy, structural analysis, structural biology, ...

- Still promising solution techniques based on Krylov subspace methods (Aleksei Nikolaevich Krylov, 1863-1945)
- Oldest but still effective solver : the conjugate gradient method (CG) [M .R. Hestenes and E. Stiefel, JNRBS, 1952]

1. Improving attainable accuracy

2. Detecting soft error in the Conjugate Gradient method

Outline

1. Improving attainable accuracy
2. Detecting soft error in the Conjugate Gradient method

Original algorithm at a glance

for $i = 0, \ldots$ **do**
    $s_i := Ap_i$
    $\alpha_i := r_i^T u_i / s_i^T p_i$
    $x_{i+1} := x_i + \alpha_i p_i$
    $r_{i+1} := r_i - \alpha_i s_i$
    $u_{i+1} := M^{-1} r_{i+1}$
    $\beta_{i+1} := r_{i+1}^T u_{i+1} / r_i^T u_i$
    $p_{i+1} := u_{i+1} + \beta_{i+1} p_i$
end for

Original algorithm at a glance

for $i = 0, \ldots$ **do**
$s_i := Ap_i$
$\alpha_i := r_i^T u_i / s_i^T p_i$
$x_{i+1} := x_i + \alpha_i p_i$
$r_{i+1} := r_i - \alpha_i s_i$
$u_{i+1} := M^{-1} r_{i+1}$
$\beta_{i+1} := r_{i+1}^T u_{i+1} / r_i^T u_i$
$p_{i+1} := u_{i+1} + \beta_{i+1} p_i$
end for

- Parallel performance bottleneck: 2 separated synchronizing scalar-products

Original algorithm at a glance

for $i = 0, \ldots$ **do**
$s_i := Ap_i$
$\alpha_i := r_i^T u_i / s_i^T p_i$
$x_{i+1} := x_i + \alpha_i p_i$
$r_{i+1} := r_i - \alpha_i s_i$
$u_{i+1} := M^{-1} r_{i+1}$
$\beta_{i+1} := r_{i+1}^T u_{i+1} / r_i^T u_i$
$p_{i+1} := u_{i+1} + \beta_{i+1} p_i$
end for

- Parallel performance bottleneck: 2 separated synchronizing scalar-products
- Many variants have been designed to overcome this drawback, one of the most recent and promising is pipelined CG
  [P. Ghysels and W. Vanroose, ParCo, 2014]

Pipelined CG - p-CG

for $i = 0, \ldots$ **do**

$\alpha_i := r_i^T u_i \quad \beta_i := w_i^T u_i$

$m_i := M^{-1} w_i \quad v_i := A m_i$

$z_i := v_i + \beta_i z_{i-1} \quad q_i := m_i + \beta_i q_{i-1}$

$s_i := w_i + \beta_i s_{i-1} \quad p_i := u_i + \beta_i p_{i-1}$

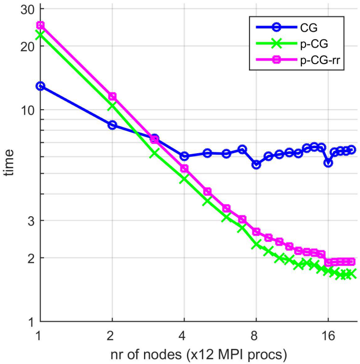
$x_{i+1} := x_i + \alpha_i p_i \quad r_{i+1} := r_i - \alpha_i s_i$

$u_{i+1} := u_i - \alpha_i q_i \quad w_{i+1} := w_i - \alpha_i z_i$

end for

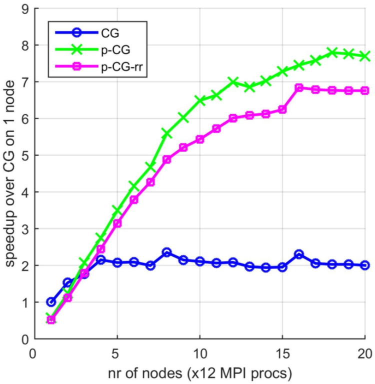
- A single non-synchronizing double scalar-products $\rightarrow$ possible overlap of mat-vec and preconditioning with non-blocking reduction

Parallel performance: 1 Mdof 2D Poisson



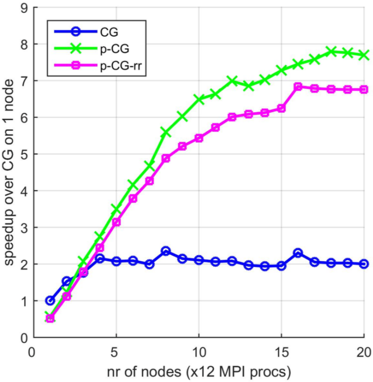
- In sequential extra computation makes p-CG (green curve) slower
- Quickly parallel p-CG outperforms regular parallel CG (blue curve)

Parallel performance: 1 Mdof 2D Poisson



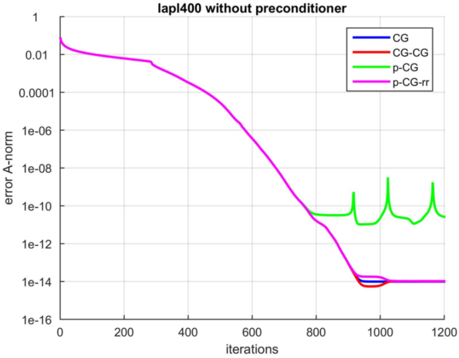
- In sequential extra computation makes p-CG (green curve) slower
- Quickly parallel p-CG outperforms regular parallel CG (blue curve)

Parallel performance: 1 Mdof 2D Poisson



- In sequential extra computation makes p-CG (green curve) slower
- Quickly parallel p-CG outperforms regular parallel CG (blue curve)
- Teasing: purple curve

Why shall we mind ?



- Attainable accuracy of p-CG worse than classical CG
- Known/expected behaviour for three-term recurrence variants
  [M.H.Gutknecht, Z.Strakoš, SIMAX, 2000]

Possible remedy

- Develop a (tedious) rounding-error analysis based on known results (see e.g. N. Higham, SIAM book, 2002]) to compute the propagation of local rounding errors in pipelined CG

$$\mathrm{fl}(a \text{ op } b) = (a \text{ op } b)(1+\epsilon), \quad |\epsilon| \leq \psi$$

$$\begin{aligned} f_{i+1} &= (b - A\bar{x}_{i+1}) - \bar{r}_{i+1} \\ &= b - A(\bar{x}_i + \bar{\alpha}_i \bar{p}_i + \delta_i^x) - (\bar{r}_i - \bar{\alpha}_i \bar{s}_i + \delta_i^r) \\ &= f_i - \bar{\alpha}_i g_i - A\delta_i^x - \delta_i^r \end{aligned}$$

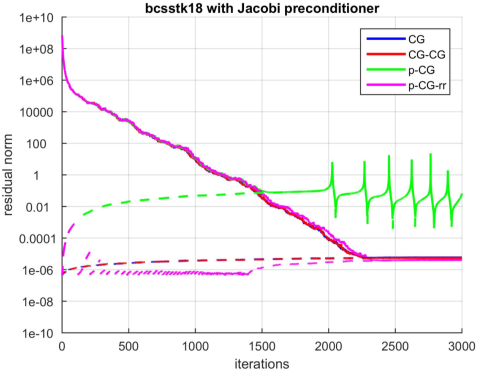
- Design a residual replacement strategy [H. van der Vorst, Q. Ye, SISC, 2000]

$$\|f_i\| \leq \sqrt{\psi}\|\bar{r}_i\| \quad \text{and} \quad \|f_{i+1}\| > \sqrt{\psi}\|\bar{r}_{i+1}\|.$$

Features of the new algorithm

At a negligible extra computational cost

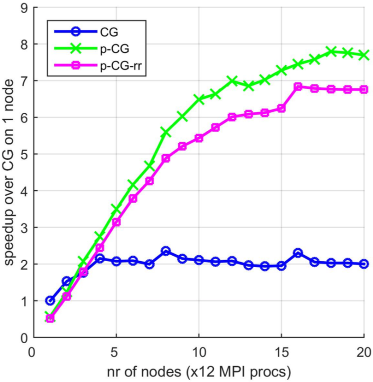
- Attainable accuracy is recovered



Features of the new algorithm

At a negligible extra computational cost

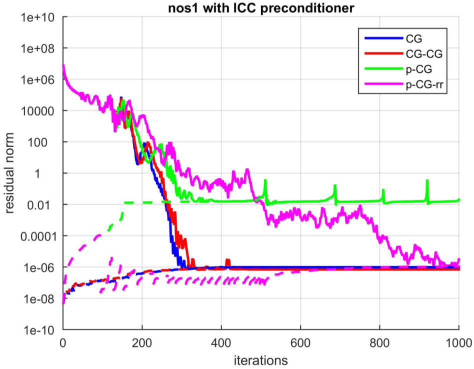
- Attainable accuracy is recovered
- Parrallel performance not “much affected” (purple vs green)



Not ended story

A few still open questions

- Analysis of the convergenve delay
- Relax some theoretical hypothesis that might not hold in practice



Outline

1. Improving attainable accuracy
2. Detecting soft error in the Conjugate Gradient method

Why soft errors occur?

What is soft error?

- Possible causes : voltage reduction, electricity fluctuations, cosmic particle effects, etc...
- Appears on: memories, registers, pipeline of the processor

Extreme scale platforms

Why soft errors occur?

What is soft error?

- Possible causes : voltage reduction, electricity fluctuations, cosmic particle effects, etc...
- Appears on: memories, registers, pipeline of the processor

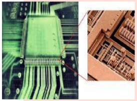
Extreme scale platforms

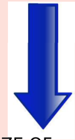
Why soft errors occur?

What is soft error?

- Possible causes : voltage reduction, electricity fluctuations, cosmic particle effects, etc...
- Appears on: memories, registers, pipeline of the processor

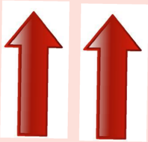
Extreme scale platforms





SIZE OF DEVICES

POSSIBLE SOFT ERROR RATES



OF COMPONENTS

AREA EFFECTED BY RADIATION

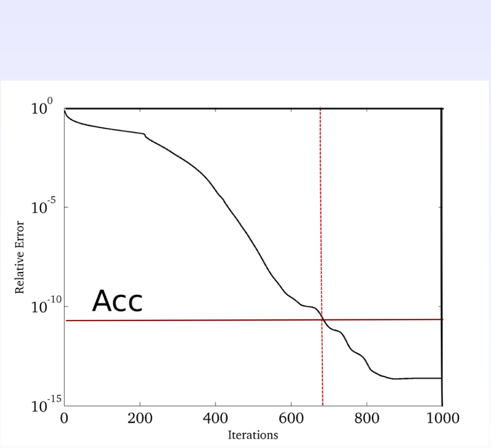
How soft errors occur?

```
for i = 0, ... do
    s := A p_i
    α := r_i^T u_i / s^T p_i
    x_{i+1} := x_i + α p_i
    r_{i+1} := r_i − α s
    u_{i+1} := M^{-1} r_{i+1}
    β := r_{i+1}^T u_{i+1} / r_i^T u_i
    p_{i+1} := u_{i+1} + β p_i
end for
```

We consider transient soft errors

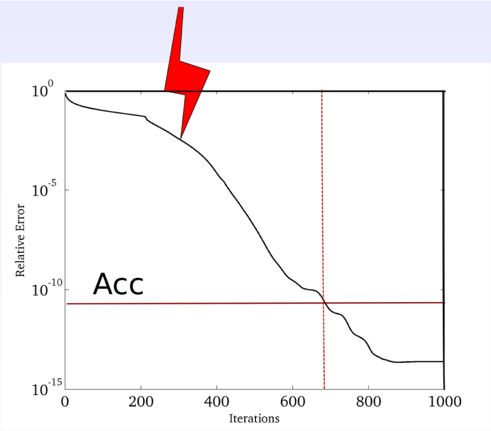
- in the most computationally expensive kernels
- other "cheaper" kernels could be protected by redundancy

Protocol for sensitivity study



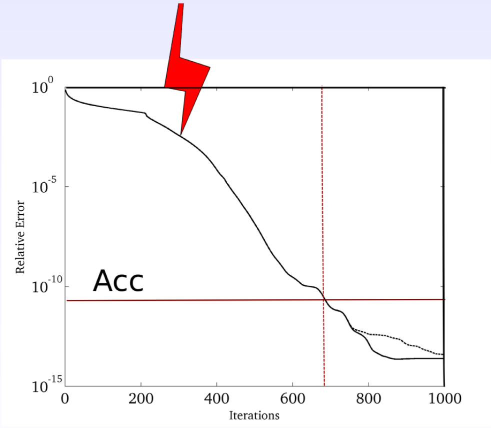
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



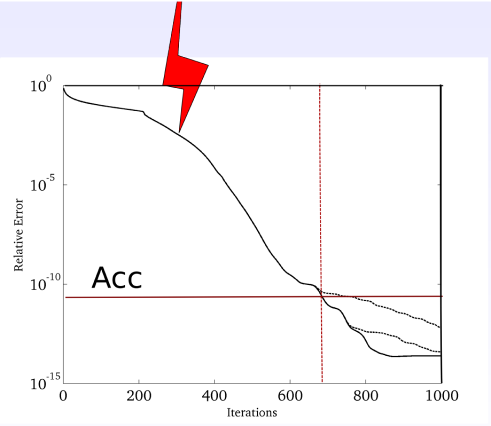
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



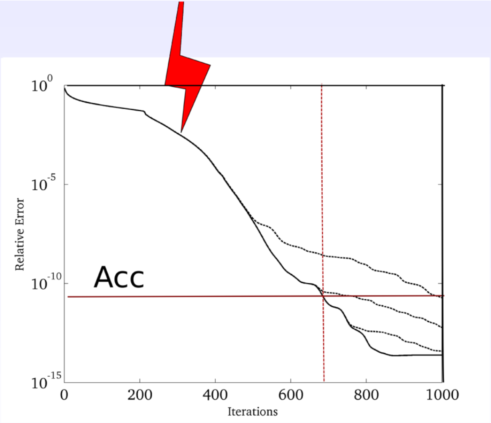
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



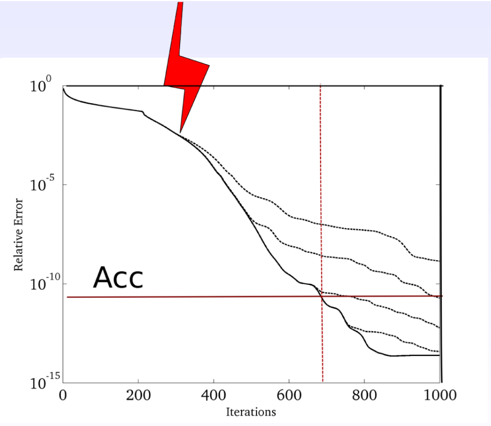
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



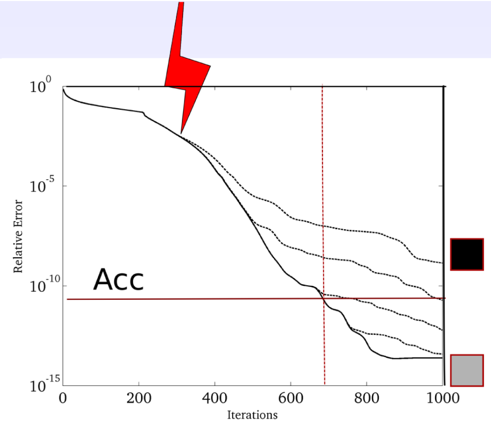
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



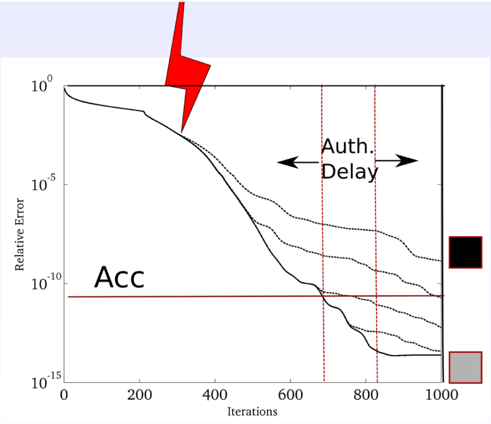
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



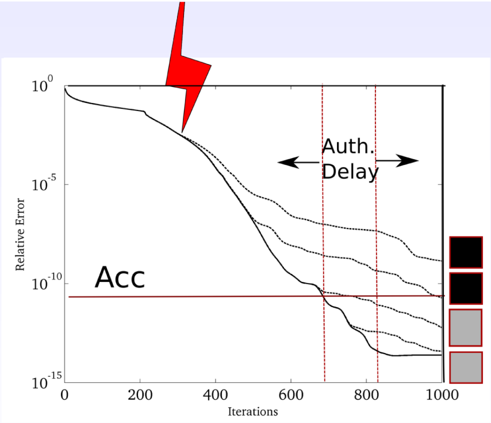
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



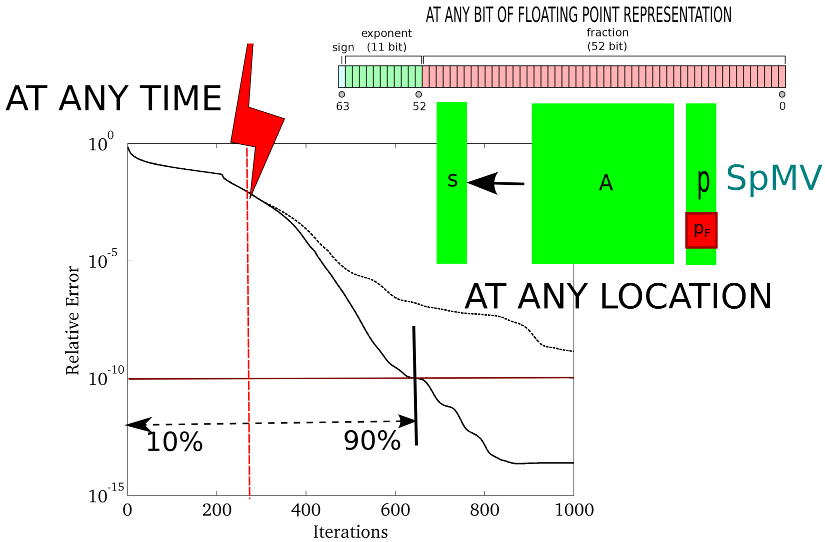
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study

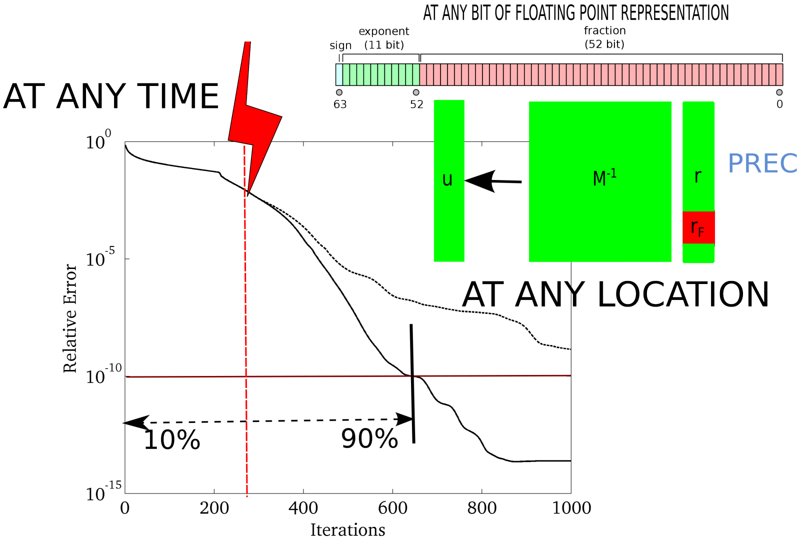


	Effect on the convergence
Converged	
Not Converged	

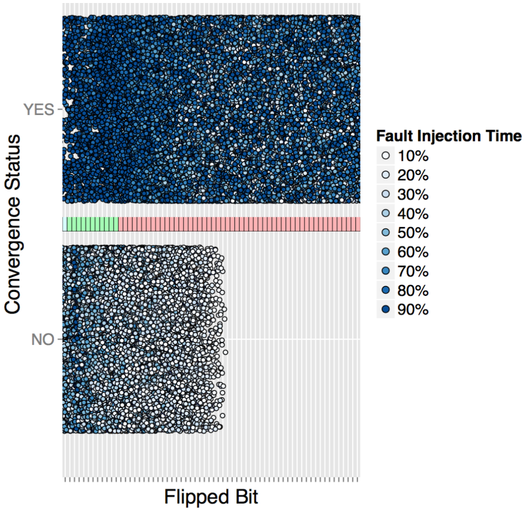
Fault injection methodology in 64-bit



Fault injection methodology in 64-bit

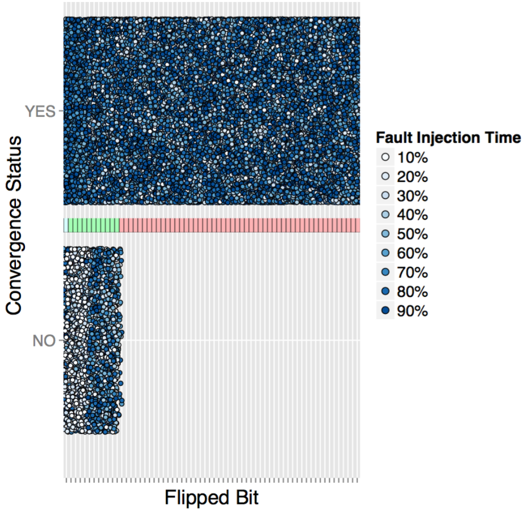


Sensitivity to soft-errors in mat-vec



- Many soft-errors are silent
- Mainly early bit-flip on high order bits are critical when computing $s_i = Ap_i$

Sensitivity to soft-errors in precond



- Even many more soft-errors are silent
- Mostly bit-flip in sign/exponent are critical when computing $u_{i+1} := M^{-1} r_{i+1}$

Main Observations

Sensitivity

- PCG algorithm is more sensitive for soft-errors on SpMV
- Soft-errors have different propagation patterns which influence the parameters of the algorithm in distinct ways

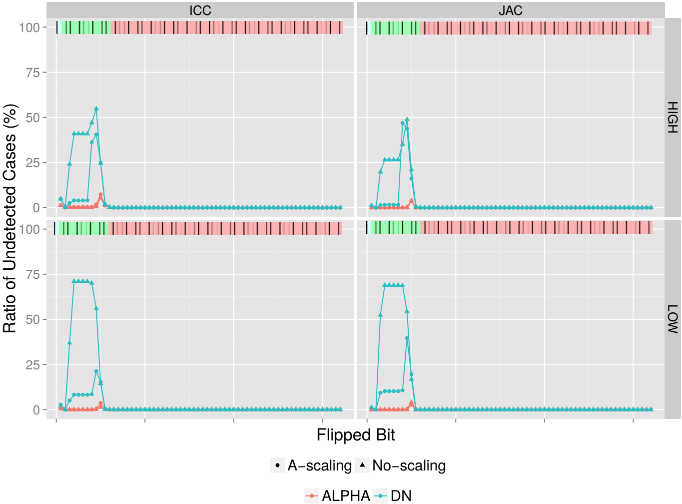
Main Observations

Sensitivity

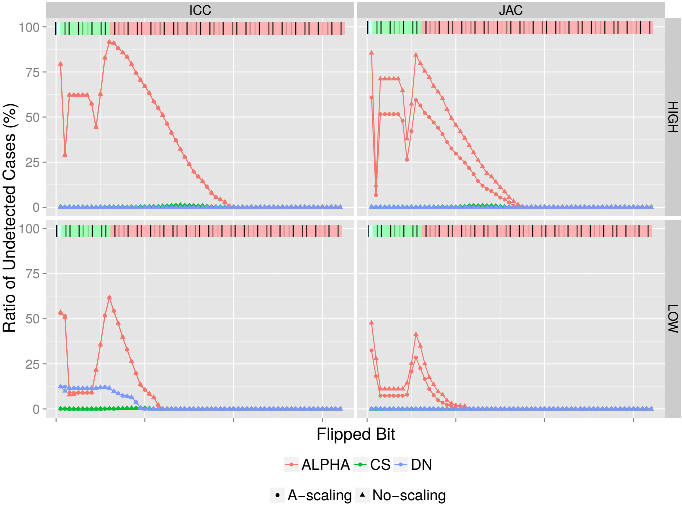
- PCG algorithm is more sensitive for soft-errors on SpMV
- Soft-errors have different propagation patterns which influence the parameters of the algorithm in distinct ways

What about soft-error detection based on rounding error analysis [Vorst & Yee, SISC, 2000] a safe interval for α parameter $(\lambda_{\text{max}}^{-1} \leq \alpha_i \leq \lambda_{\text{min}}^{-1})$ [Hestenes & Steifel, :1 JNRBS, 1952] ?

Detection robustness v.s. Preco. bit-flip location



Detection robustness v.s. SpMV bit-flip location



Main Observations

Detection

- Deviation is a good criterion candidate for SpMV faults but not for preconditioner faults
- Control frequency for deviation should be investigated
- α criterion works well for preconditioner faults but not for SpMV
- An estimation of extremal eigenvalues of preconditioned matrix is needed for α criterion (often known for scalable preconditioners)

Acknowlegement for financial support:

- French ANR: RESCUE project
- European FP7 : Exa2CT project
- G8 : ECS project

More information on
http://hiepacs.bordeaux.inria.fr/

Merci for your attention

Questions ?

Acknowlegement for financial support:

- French ANR: RESCUE project
- European FP7 : Exa2CT project
- G8 : ECS project

More information on
http://hiepacs.bordeaux.inria.fr/