RAIM 2017
October 25, 2017
Lyon, France

Effects of roundoff and soft error propagation in CG on attainable accuracy

E. Agullo

joint work with

S. Cools (Antwerpen Univ.),
L. Giraud (Inria), W. Vanroose (Antwerpen Univ.), F. Yetkin (Istanbul Kemerburgaz Univ.)

HiePACS - Inria Project
Inria Bordeaux Sud-Ouest

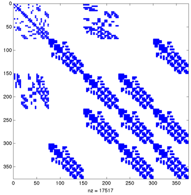
Forewords

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 (in parallel) $Ax = b$, where A is a large, sparse matrix



- Appears in many academic and industrial simulation codes;
- Iterative methods (vs direct methods)
  - 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. Roundoff errors: improving attainable accuracy of scalable variants (in p-CG)

2. Soft errors: impact on attainable accuracy and detection (in classical CG)

Outline

1. Roundoff errors: improving attainable accuracy of scalable variants (in p-CG)

2. Soft errors: impact on attainable accuracy and detection (in classical CG)

Original CG algorithm at a glance

1: **for** $i = 0, \dots$ **do**
2: $s_i := Ap_i$
3: $\alpha_i := r_i^T u_i / s_i^T p_i$
4: $x_{i+1} := x_i + \alpha_i p_i$
5: $r_{i+1} := r_i - \alpha_i s_i$
6: $u_{i+1} := M^{-1} r_{i+1}$
7: $\beta_{i+1} := r_{i+1}^T u_{i+1} / r_i^T u_i$
8: $p_{i+1} := u_{i+1} + \beta_{i+1} p_i$
9: **end for**

- Recurrences ... induce accumulation of roundoff errors
- Parallel performance bottleneck: 2 distinct dot products
- Many variants designed to overcome this drawback, including:
  - Chronopoulous-Gear CG (CG-CG) [A.T. Chronopoulous and C.W. Gear, JCAM , 1989]
  - pipelined CG (p-CG) [P. Ghysels and W. Vanroose, ParCo, 2014]

Original CG algorithm at a glance

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

- Recurrences ... induce accumulation of roundoff errors
- Parallel performance bottleneck: 2 distinct dot products
- Many variants designed to overcome this drawback, including:
  - Chronopoulous-Gear CG (CG-CG) [A.T. Chronopoulous and C.W. Gear, JCAM , 1989]
  - pipelined CG (p-CG) [P. Ghysels and W. Vanroose, ParCo, 2014]

Original CG 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

- Recurrences ... induce accumulation of roundoff errors
- Parallel performance bottleneck: 2 distinct dot products
- Many variants designed to overcome this drawback, including:
  - Chronopoulous-Gear CG (CG-CG) [A.T. Chronopoulous and C.W. Gear, JCAM , 1989]
  - pipelined CG (p-CG) [P. Ghysels and W. Vanroose, ParCo, 2014]

Original CG algorithm at a glance

1: **for** $i = 0, \ldots$ **do**
2: $\quad s_i := Ap_i$
3: $\quad \alpha_i := r_i^T u_i / s_i^T p_i$
4: $\quad x_{i+1} := x_i + \alpha_i p_i$
5: $\quad r_{i+1} := r_i - \alpha_i s_i$
6: $\quad u_{i+1} := M^{-1} r_{i+1}$
7: $\quad \beta_{i+1} := r_{i+1}^T u_{i+1} / r_i^T u_i$
8: $\quad p_{i+1} := u_{i+1} + \beta_{i+1} p_i$
9: **end for**

- Recurrences ... induce accumulation of roundoff errors
- Parallel performance bottleneck: 2 distinct dot products
- Many variants designed to overcome this drawback, including:
  - Chronopoulous-Gear CG (CG-CG) [A.T. Chronopoulous and C.W. Gear, JCAM , 1989]
  - pipelined CG (p-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 dot product $\rightarrow$ possible overlap of mat-vec and preconditioning with non-blocking reduction
- But additional recurrences ... induce amplification of roundoff errors

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 dot product
  $\rightarrow$ possible overlap of mat-vec and preconditioning with non-blocking reduction
- But additional recurrences ... induce amplification of roundoff errors

Pipelined CG (p-CG)

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

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

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

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

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

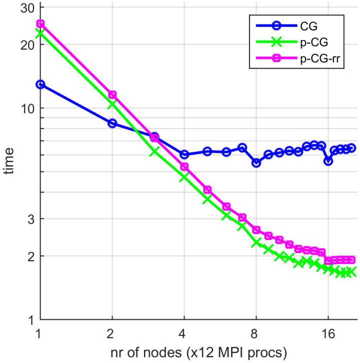
$\quad x_{i+1} := x_i + \alpha_i p_i \quad r_{i+1} := r_i - \alpha_i s_i$

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

end for

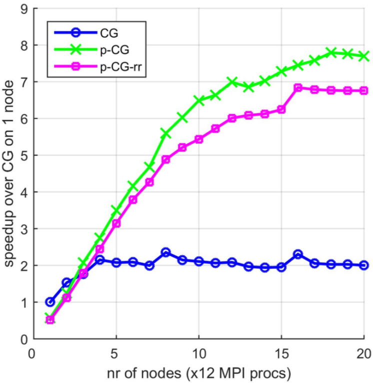
- A single non-synchronizing double dot product
  $\rightarrow$ possible overlap of mat-vec and preconditioning with non-blocking reduction
- But additional recurrences ... induce amplification of roundoff errors

Parallel performance: 1 Mdof 2D Poisson



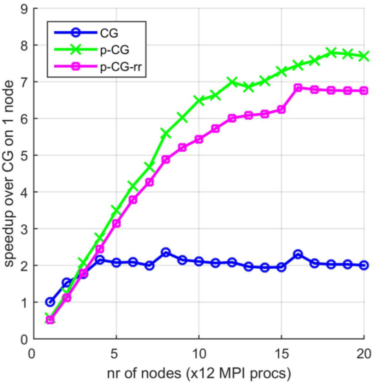
- Extra computation makes p-CG (green curve) slower (sequential reference overhead)
- p-CG eventually outperforms classical CG (blue curve)
- Teasing: purple curve

Parallel performance: 1 Mdof 2D Poisson



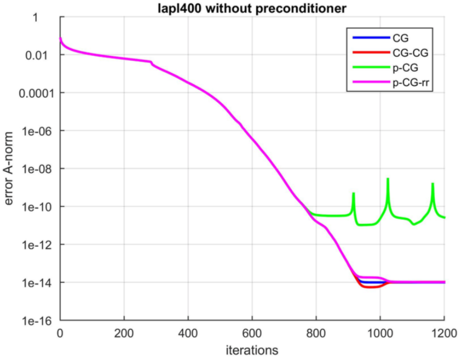
- Extra computation makes p-CG (green curve) slower (sequential reference overhead)
- p-CG eventually outperforms classical CG (blue curve)
- Teasing: purple curve

Parallel performance: 1 Mdof 2D Poisson



- Extra computation makes p-CG (green curve) slower (sequential reference overhead)
- p-CG eventually outperforms classical 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 on the deviation in pipelined CG

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

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

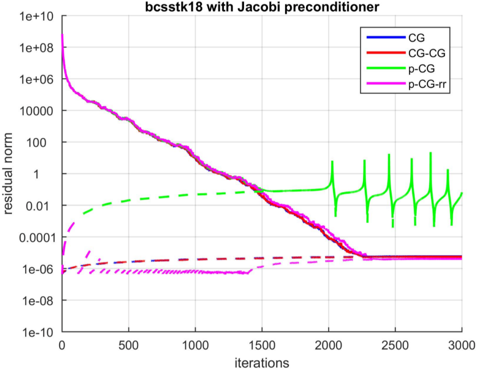
- Design a residual replacement strategy based on a practical estimate [H. van der Vorst, Q. Ye, SISC, 2000]

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

Features of the new algorithm (p-CG-rr)

At a negligible extra computational cost:

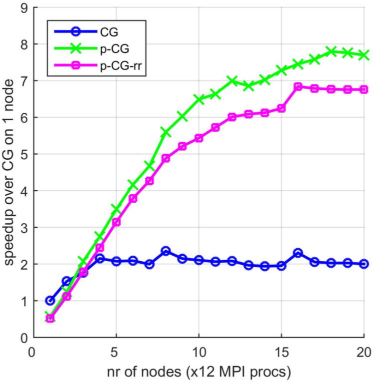
- Attainable accuracy is recovered



Features of the new algorithm (p-CG-rr)

At a negligible extra computational cost:

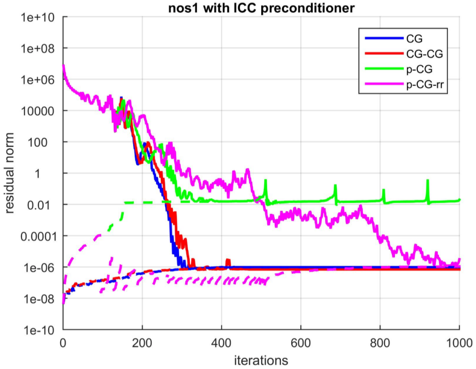
- Attainable accuracy is recovered
- Parrallel performance not "much affected" (purple vs green)



To be continued!

A few still open questions

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



Outline

1. Roundoff errors: improving attainable accuracy of scalable variants (in p-CG)

2. Soft errors: impact on attainable accuracy and detection (in classical CG)

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

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

How soft errors occur?

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

We consider transient soft errors

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

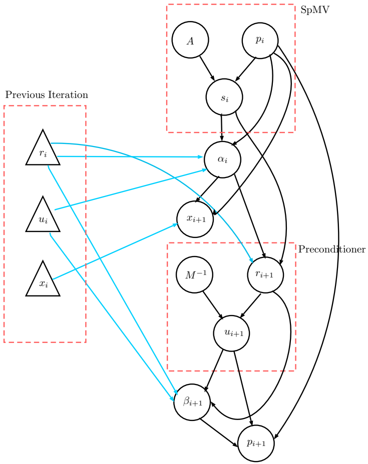
Soft errors in this study

Elliott, Hoemmen, Mueller, 2013'

- "Transient faults occur once, and while the fault is transient the effect of the fault may be persistent."
- SDC. "In this work, we address a very specific type of fault, i.e., a fault that silently introduces bad data, while not persistently tainting the data that was used in the calculation. For example, let $a = 2$ and $b = 2$, then $c = a + b = 10$, while simplistic, this model presumes no knowledge of the nature of the fault, only that c is incorrect."

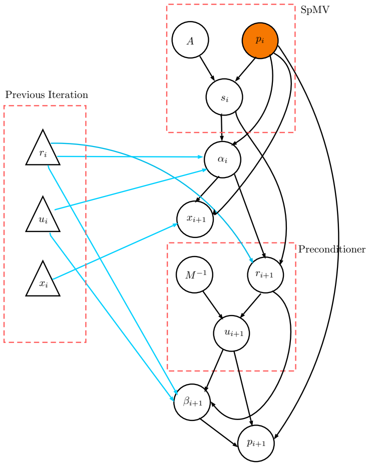
Propagation of SDC in SpMV

$r_0 := b - Ax_0$
$u_0 := M^{-1}r_0$; $p_0 := r_0$
for $i = 0, \dots$ **do**
  $s := Ap_i$
  $\alpha := (r_i, u_i)/(s, p_i)$
  $x_{i+1} := x_i + \alpha p_i$
  $r_{i+1} := r_i - \alpha s$
  $u_{i+1} := M^{-1} r_{i+1}$
  $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
  $p_{i+1} := u_{i+1} + \beta p_i$
end for



Propagation of SDC in SpMV

$r_0 := b - Ax_0$
$u_0 := M^{-1} r_0$; $p_0 := r_0$
for $i = 0, \dots$ **do**
    $s := A\textcolor{orange}{p_i}$
    $\alpha := (r_i, u_i)/(s, p_i)$
    $x_{i+1} := x_i + \alpha p_i$
    $r_{i+1} := r_i - \alpha s$
    $u_{i+1} := M^{-1} r_{i+1}$
    $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
    $p_{i+1} := u_{i+1} + \beta p_i$
end for



Propagation of SDC in SpMV

$r_0 := b - Ax_0$

$u_0 := M^{-1}r_0$; $p_0 := r_0$

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

 $s := Ap_i$

 $\alpha := (r_i, u_i)/(s, p_i)$

 $x_{i+1} := x_i + \alpha p_i$

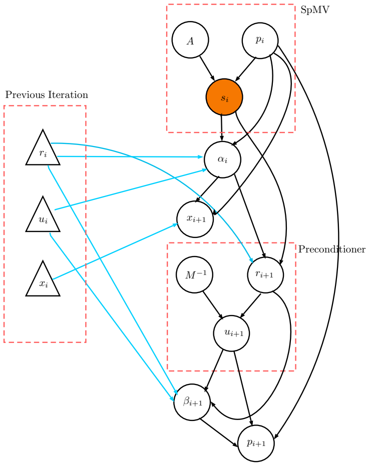
 $r_{i+1} := r_i - \alpha s$

 $u_{i+1} := M^{-1} r_{i+1}$

 $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$

 $p_{i+1} := u_{i+1} + \beta p_i$

end for



Propagation of SDC in SpMV

$r_0 := b - Ax_0$

$u_0 := M^{-1} r_0$; $p_0 := r_0$

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

  $s := Ap_i$

  $\alpha := (r_i, u_i)/(s, p_i)$

  $x_{i+1} := x_i + \alpha p_i$

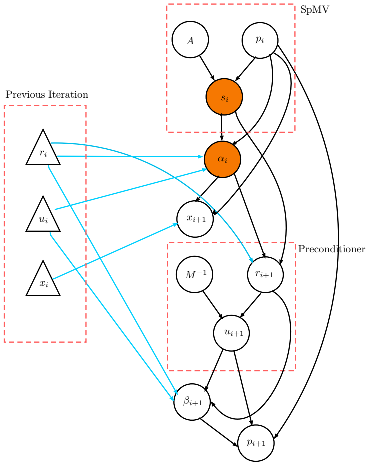
  $r_{i+1} := r_i - \alpha s$

  $u_{i+1} := M^{-1} r_{i+1}$

  $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$

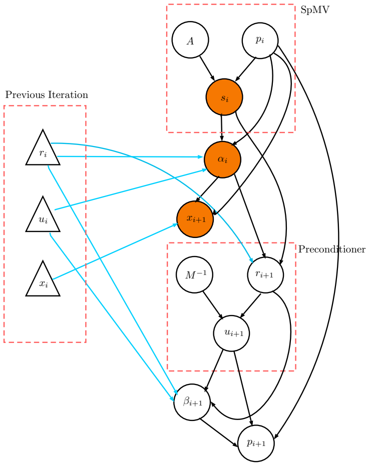
  $p_{i+1} := u_{i+1} + \beta p_i$

end for



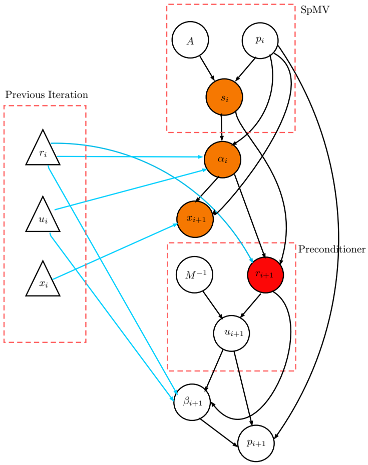
Propagation of SDC in SpMV

$$
\begin{array}{l}
r_0 := b - Ax_0 \\
u_0 := M^{-1} r_0;\ p_0 := r_0 \\
\textbf{for } i = 0, \ldots \textbf{ do} \\
\quad s := Ap_i \\
\quad \alpha := (r_i, u_i)/(s, p_i) \\
\quad x_{i+1} := x_i + \alpha p_i \\
\quad r_{i+1} := r_i - \alpha s \\
\quad u_{i+1} := M^{-1} r_{i+1} \\
\quad \beta := (r_{i+1}, u_{i+1})/(r_i, u_i) \\
\quad p_{i+1} := u_{i+1} + \beta p_i \\
\textbf{end for}
\end{array}
$$



Propagation of SDC in SpMV

$r_0 := b - Ax_0$
$u_0 := M^{-1} r_0$; $p_0 := r_0$
for $i = 0, \ldots$ **do**
  $s := Ap_i$
  $\alpha := (r_i, u_i)/(s, p_i)$
  $x_{i+1} := x_i + \alpha p_i$
  $r_{i+1} := r_i - \alpha s$
  $u_{i+1} := M^{-1} r_{i+1}$
  $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
  $p_{i+1} := u_{i+1} + \beta p_i$
end for



Propagation of SDC in SpMV

$r_0 := b - Ax_0$

$u_0 := M^{-1}r_0;\ p_0 := r_0$

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

$\quad s := Ap_i$

$\quad \alpha := (r_i, u_i)/(s, p_i)$

$\quad x_{i+1} := x_i + \alpha p_i$

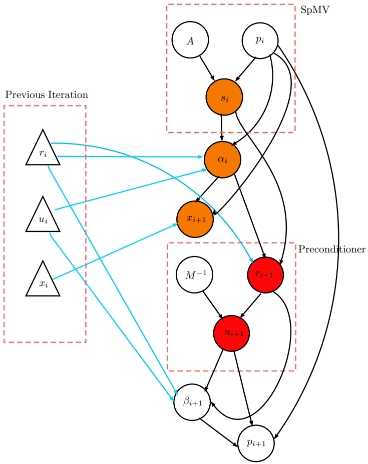
$\quad r_{i+1} := r_i - \alpha s$

$\quad u_{i+1} := M^{-1} r_{i+1}$

$\quad \beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$

$\quad p_{i+1} := u_{i+1} + \beta p_i$

end for



Propagation of SDC in SpMV

$r_0 := b - Ax_0$

$u_0 := M^{-1} r_0$; $p_0 := r_0$

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

  $s := Ap_i$

  $\alpha := (r_i, u_i)/(s, p_i)$

  $x_{i+1} := x_i + \alpha p_i$

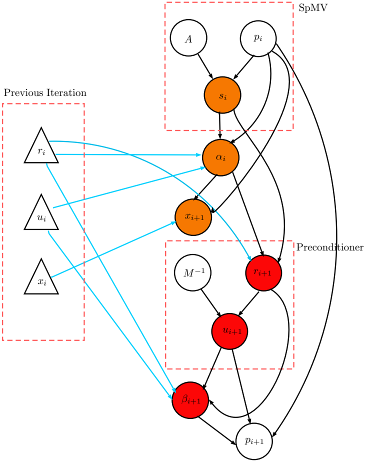
  $r_{i+1} := r_i - \alpha s$

  $u_{i+1} := M^{-1} r_{i+1}$

  $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$

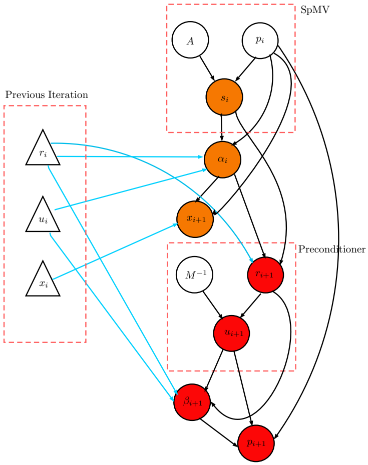
  $p_{i+1} := u_{i+1} + \beta p_i$

end for



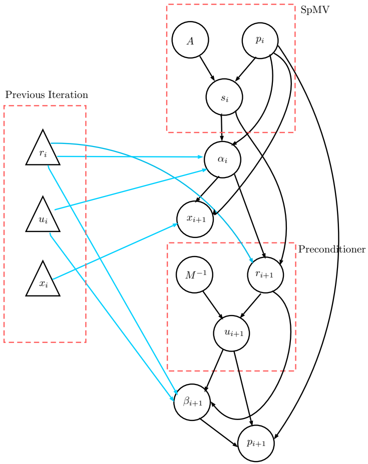
Propagation of SDC in SpMV

$r_0 := b - Ax_0$
$u_0 := M^{-1} r_0; \; p_0 := r_0$
for $i = 0, \dots$ **do**
  $s := Ap_i$
  $\alpha := (r_i, u_i)/(s, p_i)$
  $x_{i+1} := x_i + \alpha p_i$
  $r_{i+1} := r_i - \alpha s$
  $u_{i+1} := M^{-1} r_{i+1}$
  $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
  $p_{i+1} := u_{i+1} + \beta p_i$
end for



Propagation of SDC in preconditioner application

$r_0 := b - Ax_0$
$u_0 := M^{-1} r_0$; $p_0 := r_0$
for $i = 0, \dots$ **do**
    $s := Ap_i$
    $\alpha := (r_i, u_i)/(s, p_i)$
    $x_{i+1} := x_i + \alpha p_i$
    $r_{i+1} := r_i - \alpha s$
    $u_{i+1} := M^{-1} r_{i+1}$
    $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
    $p_{i+1} := u_{i+1} + \beta p_i$
end for



Propagation of SDC in preconditioner application

$r_0 := b - Ax_0$

$u_0 := M^{-1} r_0$; $p_0 := r_0$

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

$\quad s := Ap_i$

$\quad \alpha := (r_i, u_i)/(s, p_i)$

$\quad x_{i+1} := x_i + \alpha p_i$

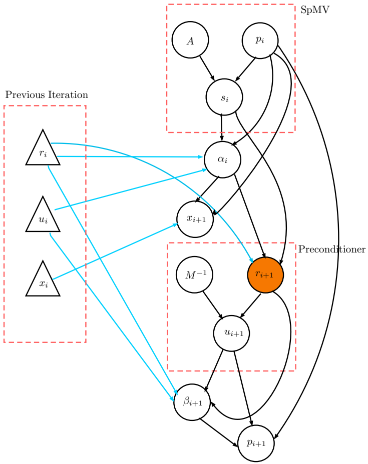
$\quad r_{i+1} := r_i - \alpha s$

$\quad u_{i+1} := M^{-1} r_{i+1}$

$\quad \beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$

$\quad p_{i+1} := u_{i+1} + \beta p_i$

end for



Propagation of SDC in preconditioner application

$r_0 := b - Ax_0$

$u_0 := M^{-1}r_0$; $p_0 := r_0$

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

$\quad s := Ap_i$

$\quad \alpha := (r_i, u_i)/(s, p_i)$

$\quad x_{i+1} := x_i + \alpha p_i$

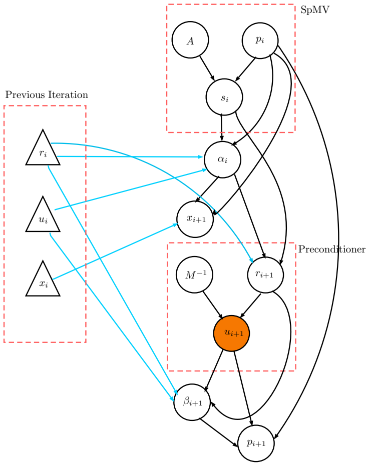
$\quad r_{i+1} := r_i - \alpha s$

$\quad u_{i+1} := M^{-1} r_{i+1}$

$\quad \beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$

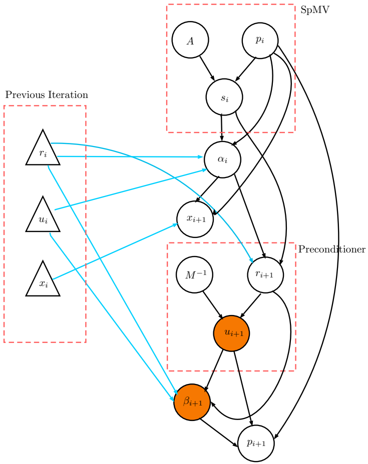
$\quad p_{i+1} := u_{i+1} + \beta p_i$

end for



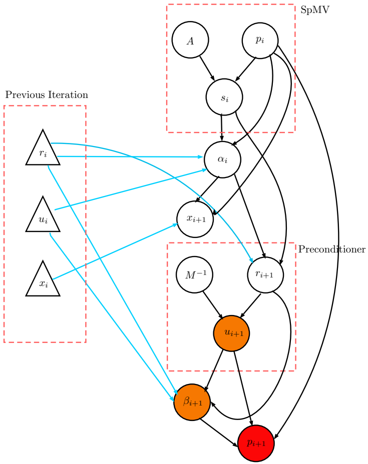
Propagation of SDC in preconditioner application

```
r_0 := b − A x_0
u_0 := M^{-1} r_0; p_0 := r_0
for i = 0, . . . do
    s := A p_i
    α := (r_i, u_i)/(s, 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}, u_{i+1})/(r_i, u_i)
    p_{i+1} := u_{i+1} + β p_i
end for
```



Propagation of SDC in preconditioner application

$r_0 := b - Ax_0$
$u_0 := M^{-1} r_0$; $p_0 := r_0$
for $i = 0, \dots$ **do**
    $s := Ap_i$
    $\alpha := (r_i, u_i)/(s, p_i)$
    $x_{i+1} := x_i + \alpha p_i$
    $r_{i+1} := r_i - \alpha s$
    $u_{i+1} := M^{-1} r_{i+1}$
    $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
    $p_{i+1} := u_{i+1} + \beta p_i$
end for



Propagation of SDC in preconditioner application

$r_0 := b - Ax_0$

$u_0 := M^{-1}r_0; \; p_0 := r_0$

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

$\quad s := A\color{red}{p_i}$

$\quad \alpha := (r_i, u_i)/(s, p_i)$

$\quad x_{i+1} := x_i + \alpha p_i$

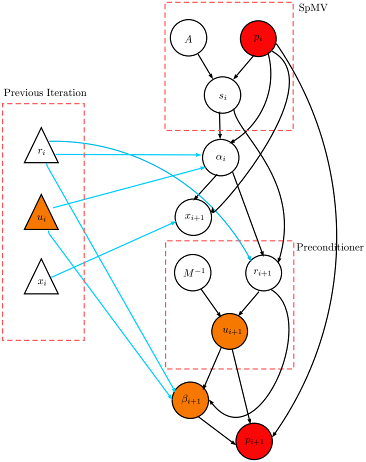
$\quad r_{i+1} := r_i - \alpha s$

$\quad \color{orange}{u_{i+1}} := M^{-1} r_{i+1}$

$\quad \color{orange}{\beta} := (r_{i+1}, \color{orange}{u_{i+1}})/(r_i, u_i)$

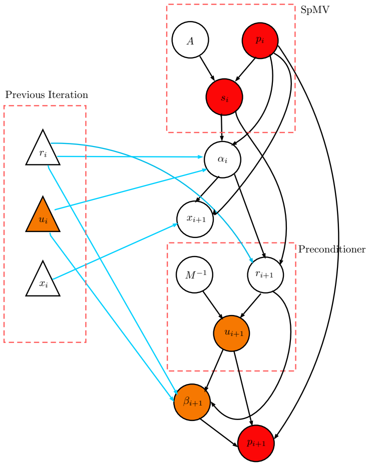
$\quad \color{red}{p_{i+1}} := \color{orange}{u_{i+1}} + \color{orange}{\beta} p_i$

end for



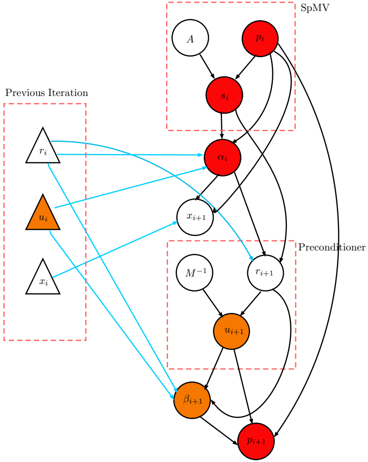
Propagation of SDC in preconditioner application

$r_0 := b - Ax_0$
$u_0 := M^{-1} r_0$; $p_0 := r_0$
for $i = 0, \ldots$ **do**
    $s := Ap_i$
    $\alpha := (r_i, u_i)/(s, p_i)$
    $x_{i+1} := x_i + \alpha p_i$
    $r_{i+1} := r_i - \alpha s$
    $u_{i+1} := M^{-1} r_{i+1}$
    $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
    $p_{i+1} := u_{i+1} + \beta p_i$
end for



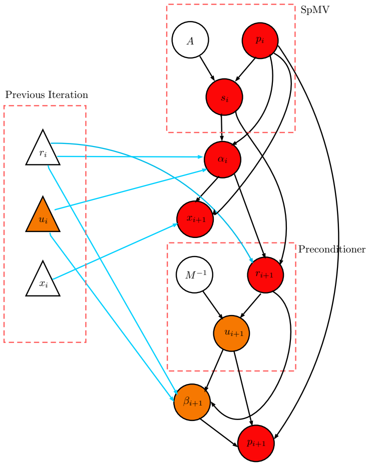
Propagation of SDC in preconditioner application

```
r_0 := b - A x_0
u_0 := M^{-1} r_0; p_0 := r_0
for i = 0, ... do
    s := A p_i
    α := (r_i, u_i)/(s, 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}, u_{i+1})/(r_i, u_i)
    p_{i+1} := u_{i+1} + β p_i
end for
```

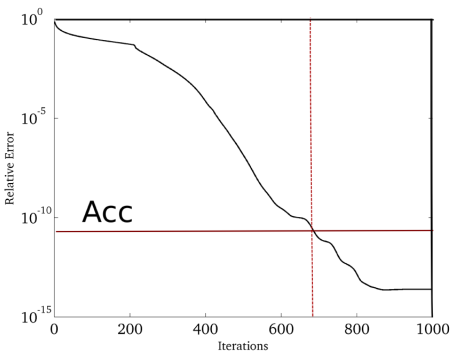


Propagation of SDC in preconditioner application

$r_0 := b - Ax_0$
$u_0 := M^{-1}r_0$; $p_0 := r_0$
for $i = 0, \ldots$ **do**
    $s := Ap_i$
    $\alpha := (r_i, u_i)/(s, p_i)$
    $x_{i+1} := x_i + \alpha p_i$
    $r_{i+1} := r_i - \alpha s$
    $u_{i+1} := M^{-1}r_{i+1}$
    $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
    $p_{i+1} := u_{i+1} + \beta p_i$
end for

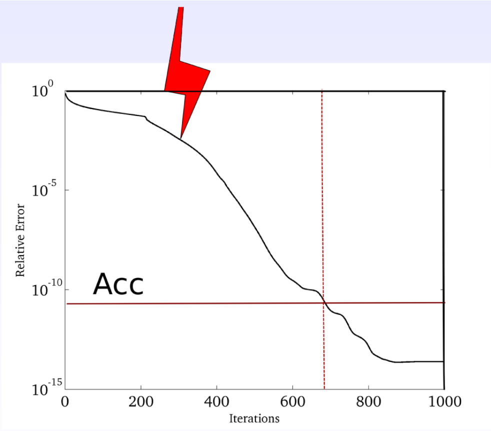


Protocol for sensitivity study



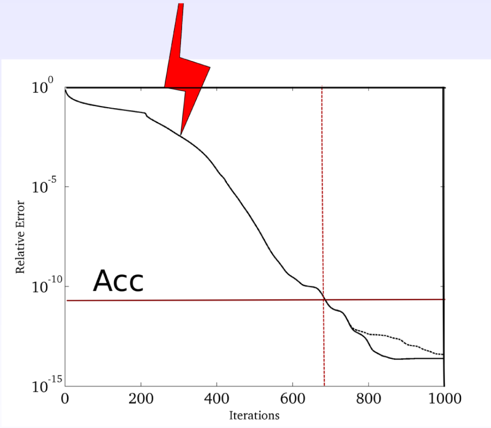
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



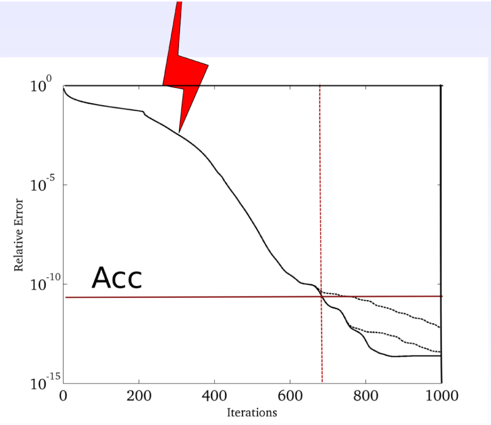
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



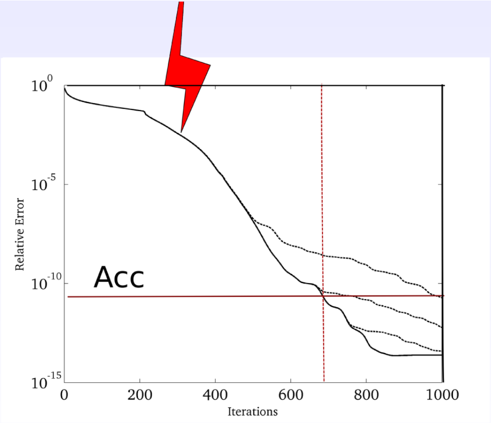
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



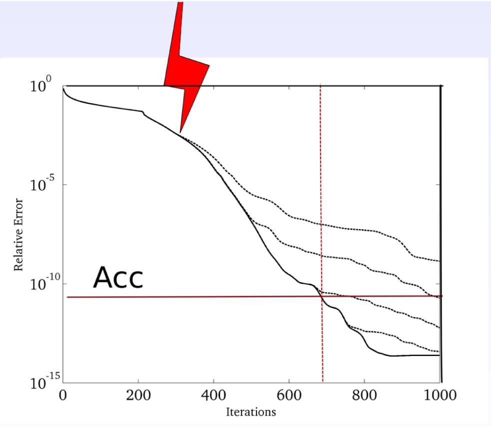
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



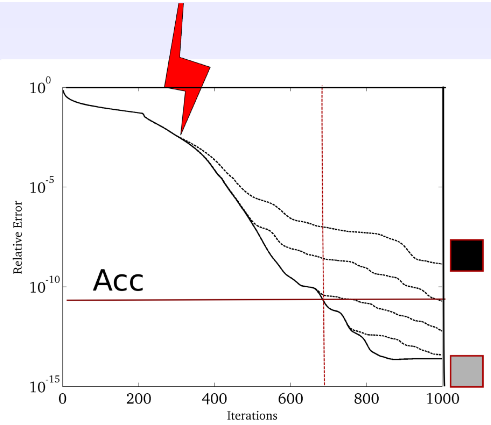
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



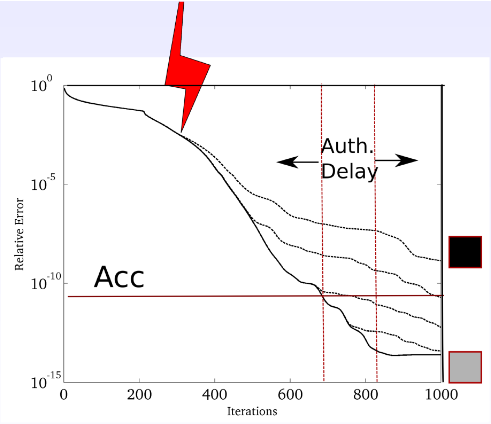
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



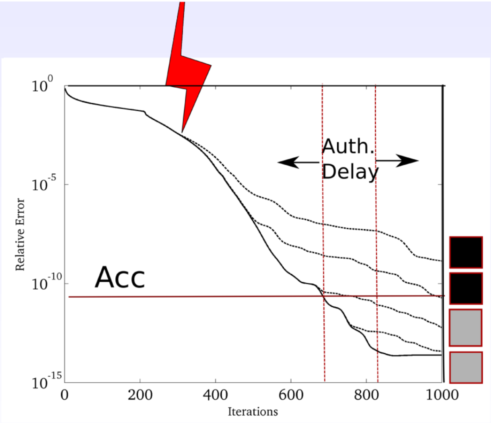
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study



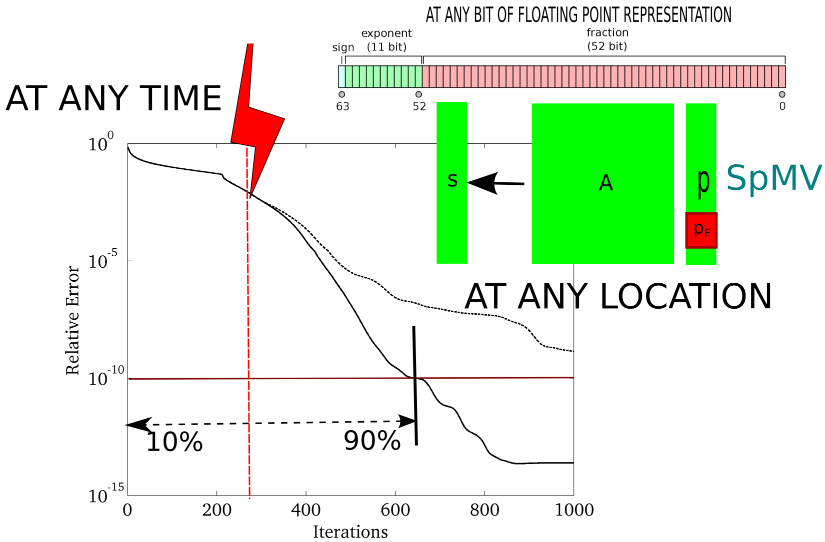
	Effect on the convergence
Converged	
Not Converged	

Protocol for sensitivity study

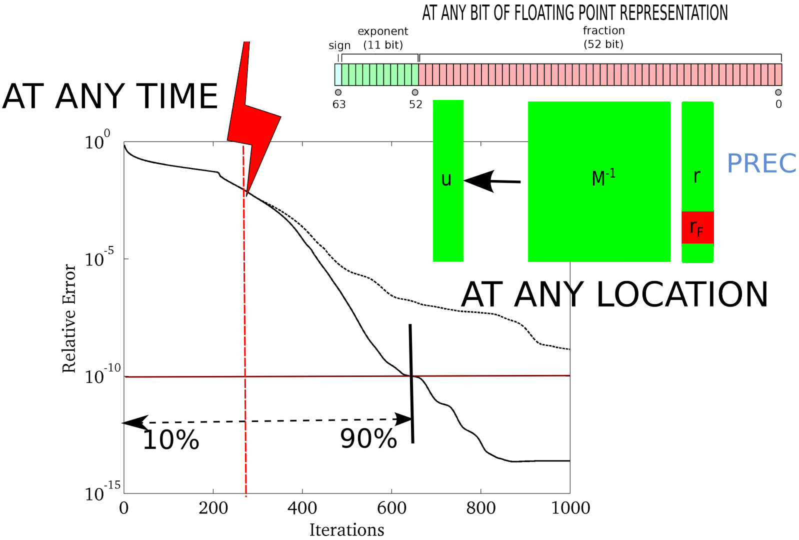


	Effect on the convergence
Converged	
Not Converged	

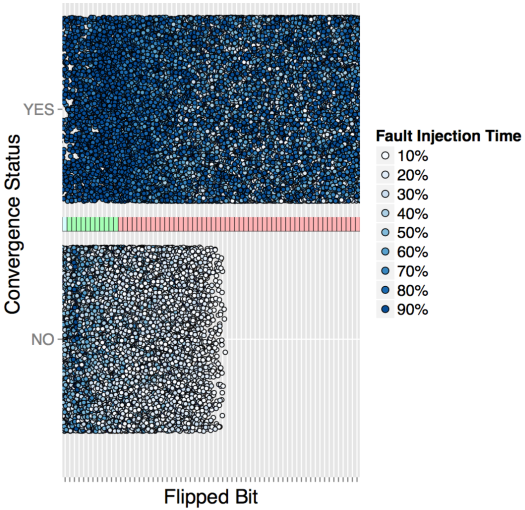
Fault injection methodology in 64-bit



Fault injection methodology in 64-bit

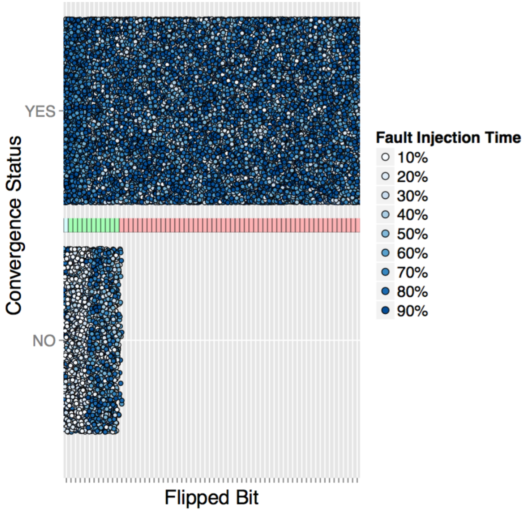


Sensitivity to soft-errors in mat-vec



- 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

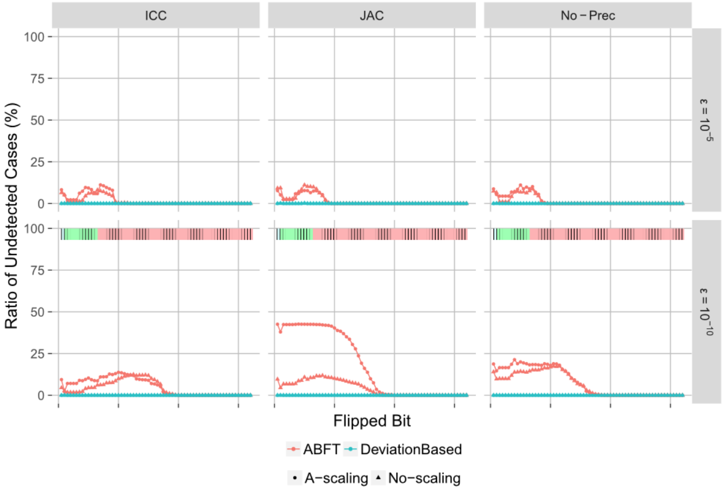


- Even many more soft-errors are silent
- Mostly bit-flip in sign/exponent are critical when computing $u_{i+1} := M^{-1} r_{i+1}$

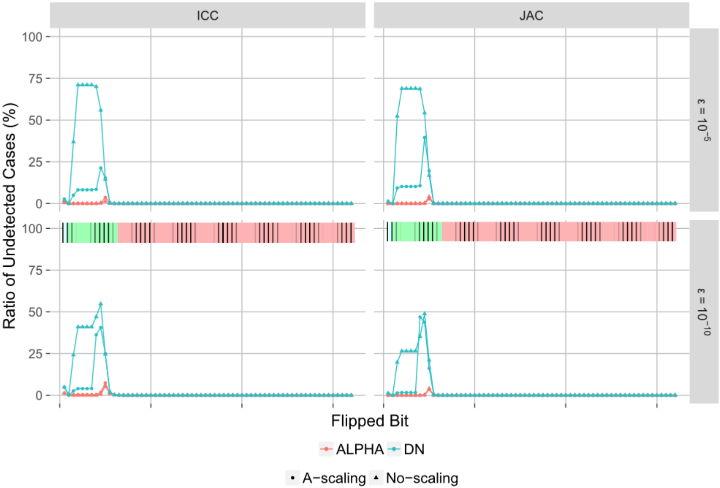
Detection criteria

- ABFT [Huang & Abraham, IEEE TOC, 1984] ("Note: since there may be roundoff errors for floating point operations, a small tolerance should be allowed for in the the comparison")
- Deviation-based (DN) [Vorst & Yee, SISC, 2000]
- α-based (alpha): $(\lambda_{\max}^{-1} \leq \alpha_i \leq \lambda_{\min}^{-1})$ [Hestenes & Steifel, JNRBS, 1952] (a safe interval for α parameter for distinguishing "normal rounding-off errors" from "errors of the computer" (sic!))

Detection robustness v.s. SpMV bit-flip location



Detection robustness v.s. Preco. 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/