GMRES-based iterative refinement in up to five precisions

Speaker : Bastien Vieublé
Joint work with : Patrick Amestoy, Alfredo Buttari, Jean-Yves L'Excellent, Nicholas J. Higham, Theo Mary

SIAM CSE 2021

Generalized iterative refinement

Algorithm Generalized iterative refinement

1: Compute the LU factorization $A = LU$ (u_f)
2: Solve $Ax_0 = b$ (u_f)
3: **while** not converged **do**
4: Compute $r_i = b - Ax_i$ (u_r)
5: Solve $Ad_i = r_i$ (u_s)
6: Compute $x_{i+1} = x_i + d_i$ (u)
7: **end while**

- The solver at step 5 is arbitrary.
- u_s expresses the precision of the computed solution d_i provided by this solver ($\neq$ unit roundoff of an arithmetic precision).

E. Carson and N. J. Higham. *"Accelerating the solution of linear systems by iterative refinement in three precisions"*. In : SIAM, 2018.

Generalized iterative refinement

Algorithm Generalized iterative refinement

Compute the LU factorization $A = LU$ (u_f)
Solve $Ax_0 = b$ (u_f)
while not converged **do**
    Compute $r_i = b - Ax_i$ (u_r)
    Solve $Ad_i = r_i$ (u_s)
    Compute $x_{i+1} = x_i + d_i$ (u)
end while

Two main properties determined by the set of precisions :

- **The convergence condition :** the maximal value of $\kappa(A)$ for which convergence is guaranteed. (u_f, u_s)
- **The limiting accuracies :** the accuracies at which the forward and backward errors converge. (u, u_r)

LU-IR3

Algorithm LU-based iterative refinement in three precisions

1: Compute the LU factorization $A = LU$ (u_f)
2: Solve $Ax_0 = b$ (u_f)
3: **while** not converged **do**
4: Compute $r_i = b - Ax_i$ (u_r)
5: Solve $Ad_i = r_i$ by $d_i = \hat{U}^{-1}\hat{L}^{-1}r_i$. (u_f)
6: Compute $x_{i+1} = x_i + d_i$ (u)
7: **end while**

Step 5 : Solver LU – $u_s \equiv u_f$.

E. Carson and N. J. Higham. *"Accelerating the solution of linear systems by iterative refinement in three precisions"*. In : SIAM, 2018.

LU-IR3

Algorithm LU-based iterative refinement in three precisions

```
Compute the LU factorization A = LU                 (u_f)
Solve A x_0 = b                                     (u_f)
while not converged do
    Compute r_i = b − A x_i                         (u_r)
    Solve A d_i = r_i by d_i = Û^{-1} L̂^{-1} r_i.   (u_f)

    Compute x_{i+1} = x_i + d_i                     (u)
end while
```

	Convergence condition	Forward error
LU-IR3	$\kappa(A) < u_f^{-1}$	$u_r\kappa(A) + u$

Very low precision factorization (e.g fp16, bfloat16) leads to a very restrictive convergence condition for LU-IR3 (e.g 2×10^3).

GMRES-IR3

Algorithm GMRES-based iterative refinement in three precisions

1: Compute the LU factorization $A = LU$ (u_f)
2: Solve $Ax_0 = b$ (u_f)
3: **while** not converged **do**
4: Compute $r_i = b - Ax_i$ (u_r)
5: Solve $\widetilde{A}d_i = \hat{U}^{-1}\hat{L}^{-1}Ad_i = \hat{U}^{-1}\hat{L}^{-1}r_i$ by GMRES at precision (u) with matrix vector products with $\widetilde{A}$ at precision (u^2).
6: Compute $x_{i+1} = x_i + d_i$ (u)
7: **end while**

Step 5 : Preconditioned GMRES in two precision – $u_s \equiv u$.

E. Carson and N. J. Higham. *“A new analysis of iterative refinement and its application to accurate solution of ill-conditioned sparse linear systems”*. In : SIAM, 2017.

GMRES-IR3

Algorithm GMRES-based iterative refinement in three precisions

Compute the LU factorization $A = LU$ (u_f)
Solve $Ax_0 = b$ (u_f)
while not converged **do**
    Compute $r_i = b - Ax_i$ (u_r)
    Solve $\widetilde{A}d_i = \hat{U}^{-1}\hat{L}^{-1}Ad_i = \hat{U}^{-1}\hat{L}^{-1}r_i$ by GMRES at precision (u) with matrix vector products with $\widetilde{A}$ at precision (u^2).
    Compute $x_{i+1} = x_i + d_i$ (u)
end while

	Convergence condition	Forward error
LU-IR3	$\kappa(A) < u_f^{-1}$	$u_r\kappa(A) + u$
GMRES-IR3	$\kappa(A) < u^{-1/2}u_f^{-1}$	$u_r\kappa(A) + u$

If u_f is fp16, then the condition on LU-IR3 is 2×10^3, on GMRES-IR3 is 2×10^{11} !

Practical issues of GMRES-IR3

In GMRES-IR3, LU solves are performed at precision u^2 : this is a **major practical issue**.

Practical issues of GMRES-IR3

In GMRES-IR3, LU solves are performed at precision u^2 : this is a **major practical issue**.

- Increases cost per iteration.
- If u^2 is fp128, requires a quad precision solver.
- Cast the LU factors from precision u_f to precision u^2

  $\Rightarrow$ huge memory increase

Other issue : Do we need to run the other GMRES operations in precision u ?

Practical issues of GMRES-IR3

In GMRES-IR3, LU solves are performed at precision u^2 : this is a **major practical issue**.

- Increases cost per iteration.
- If u^2 is fp128, requires a quad precision solver.
- Cast the LU factors from precision u_f to precision u^2

  $\Rightarrow$ huge memory increase

Other issue : Do we need to run the other GMRES operations in precision u ?

$\Rightarrow$ What if we **relax the precision $\mathbf{u}^2$** on the preconditioning **and $\mathbf{u}$** on the rest of the operations ?

GMRES-IR5

Algorithm GMRES-IR3

Compute the LU factorization $A = LU$ (u_f)

Solve $Ax_0 = b$ (u_f)

while not converged **do**

  Compute $r_i = b - Ax_i$ (u_r)

  Solve $\widetilde{A}d_i = \hat{U}^{-1}\hat{L}^{-1}Ad_i = \hat{U}^{-1}\hat{L}^{-1}r_i$ by GMRES at precision $(u\ \ )$ with matrix vector products with $\widetilde{A}$ at precision (u^2).

  Compute $x_{i+1} = x_i + d_i$ (u)

end while

GMRES-IR5

Algorithm GMRES-IR3

Compute the LU factorization $A = LU$ (u_f)

Solve $Ax_0 = b$ (u_f)

while not converged **do**

  Compute $r_i = b - Ax_i$ (u_r)

  Solve $\widetilde{A}d_i = \hat{U}^{-1}\hat{L}^{-1}Ad_i = \hat{U}^{-1}\hat{L}^{-1}r_i$ by GMRES at precision $(u\ \ )$ with matrix vector products with $\widetilde{A}$ at precision (u^2).

  Compute $x_{i+1} = x_i + d_i$ (u)

end while

GMRES-IR5

Algorithm GMRES-IR5

Compute the LU factorization $A = LU$ (u_f)
Solve $Ax_0 = b$ (u_f)
while not converged **do**
  Compute $r_i = b - Ax_i$ (u_r)
  Solve $\widetilde{A}d_i = \hat{U}^{-1}\hat{L}^{-1}Ad_i = \hat{U}^{-1}\hat{L}^{-1}r_i$ by GMRES at precision (u_g) with matrix vector products with $\widetilde{A}$ at precision (u_p).
  Compute $x_{i+1} = x_i + d_i$ (u)
end while

- u_p : precision at which we apply the **preconditioned** matrix-vector products.
- u_g : precision at which we apply the other **GMRES** operations.

Possibly $u_p > u^2$ (and $u_g > u$).

Preconditioned MGS-GMRES in 2 precisions

Theorem (Stability of preconditioned MGS-GMRES in 2 precisions)

Consider solving a preconditioned linear system

$$\widetilde{A}d = s, \quad \widetilde{A} = \hat{U}^{-1}\hat{L}^{-1}A, \quad A \in \mathbb{R}^{n\times n},$$

with a MGS-GMRES in precision u_g except for the products with $\widetilde{A}$ applied in precision u_p.

The computed solution $\hat{d}$ achieves a backward error of order

$$\mathsf{u_s} \equiv \mathsf{u_g} + \mathsf{u_p}\kappa(\mathsf{A})$$

⇒ It generalizes the **backward stability** of **MGS-GMRES** to a preconditioned MGS-GMRES in **2 precisions**.

C. Paige, M. Rozložník and Z. Strakoš. *“Modified Gram-Schmidt (MGS), least squares, and backward stability of MGS-GMRES”*. In : SIAM, 2006.

Convergence condition of GMRES-IR5

IR	Convergence condition
LU-IR3	$\kappa(A)u_f \ll 1$
GMRES-IR5	$\mathbf{(u_g + u_p\kappa(A))\kappa(A)^2u_f^2 \ll 1}$
GMRES-IR3	$\kappa(A)u^{1/2}u_f \ll 1$

If u_f is fp16, the condition on LU-IR3 is 2×10^3, on GMRES-IR5 (with $u_g = u_p =$ fp64) is 3×10^7, on GMRES-IR3 is 2×10^{11}

Meaningful combinations

With five arithmetics (fp16, bfloat16, fp32, fp64, fp128) GMRES-IR5 can be declined in over **3000 different combinations!**

They are not all relevant!

Filter principle : Useless to have high precision when we can use low precision without impacting the limiting accuracy and convergence condition.

Filtering rules

- $u^2 \le u_r \le u \le u_f$
- $u_p \le u_g$
- $u_p < u_f$
- $u_p < u$, $u_p = u$, and $u_p > u$
- $u_g = u$ and $u_g > u$
- $u_g < u_f$, $u_g = u_f$, and $u_g > u_f$

These rules are based on the limiting accuracy and convergence condition formulas.

Possible arithmetic precisons

	ID	Signif. bits	Exp. bits	Range	Unit roundoff u
fp128	Q	113	15	$10^{\pm 4932}$	1×10^{-34}
fp64	D	53	11	$10^{\pm 308}$	1×10^{-16}
fp32	S	24	8	$10^{\pm 38}$	6×10^{-8}
fp16	H	11	5	$10^{\pm 5}$	5×10^{-4}
bfloat16	B	8	8	$10^{\pm 38}$	4×10^{-3}

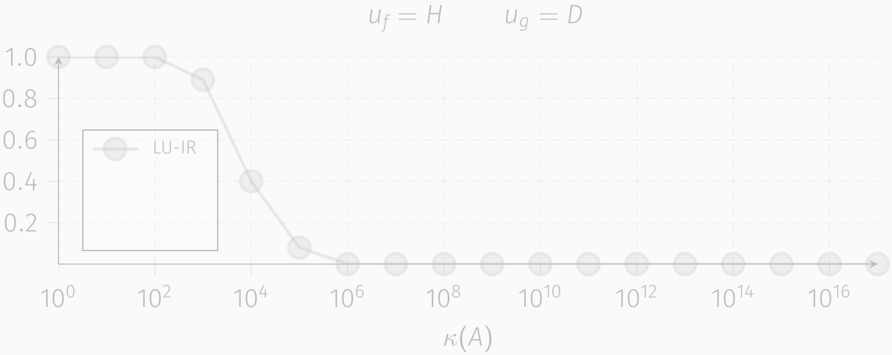
Theoretical robustness over $\kappa(A)$

u_g	u_p	Convergence Condition $\max(\kappa(A))$
LU-IR3		2×10^3
B	S	3×10^4
H	S	4×10^4
H	D	9×10^4
S	D	8×10^6
D	D	3×10^7
GMRES-IR3		2×10^{11}

Meaningful combinations of GMRES-IR5 for $u_f = H$ and $u = D$.

Five combinations between LU-IR3 and GMRES-IR3 $\Rightarrow$ More flexible precisions choice to fit at best the **hardware constraints** and the **problem difficulty**.

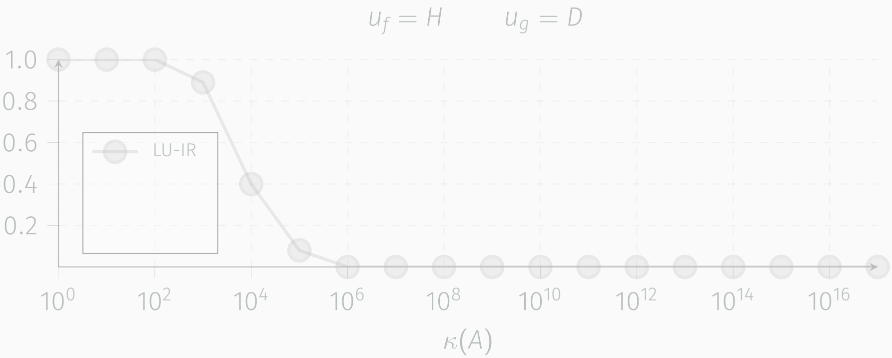
Experimental robustness over $\kappa(A)$



We want to study the experimental **robustness** on $\kappa(A)$ of the following variants :

- $\mathsf{u} = D$, $\mathsf{u_r} = Q$, and $\mathsf{u_f} = H$ **fixed**.
- GMRES precision $\mathsf{u_g}$, preconditioning precision $\mathsf{u_p}$ **varying**.

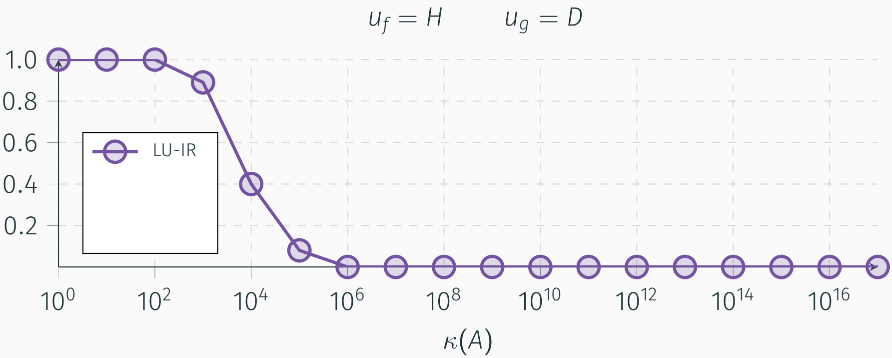
Experimental robustness over $\kappa(A)$



Evaluate the robustness ? **Success rate of convergence** seems to be a good measure.

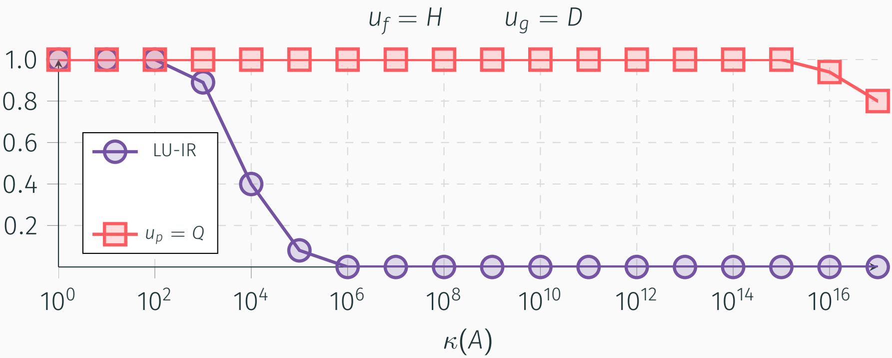
It converges when it reaches the theoritical limiting accuracy ($u = D$ and $u_r = Q \Rightarrow$ forward error $= 10^{-16}$).

Experimental robustness over $\kappa(A)$



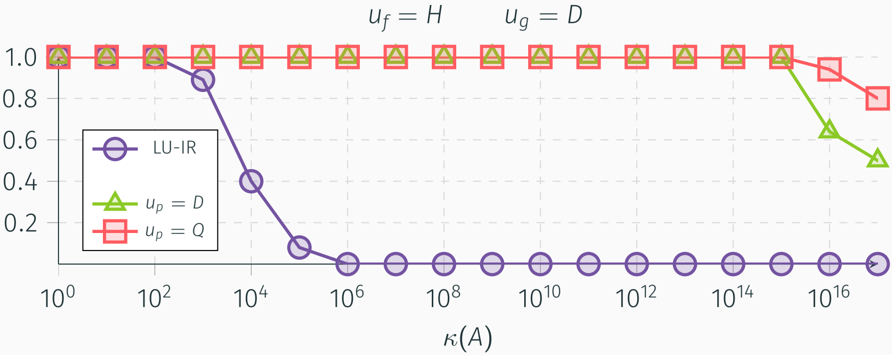
- Success rate of convergence of LU-IR3 and GMRES-IR5. Each success rate computed from 100 50×50 randsvd dense matrices.
- The more **the breaking point** of the success rate is high the more the method is robust (ex : LU-IR $\approx 10^2 - 10^3$).

Experimental robustness over $\kappa(A)$



- GMRES-IR5 with $u_g = D$ and $u_p = Q$ far more robust on $\kappa(A)$.

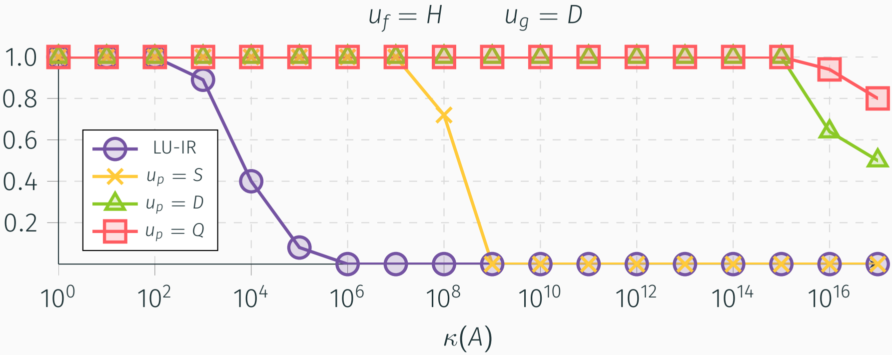
Experimental robustness over $\kappa(A)$



- When $u_p = Q \rightarrow D$, lose really little in robustness.

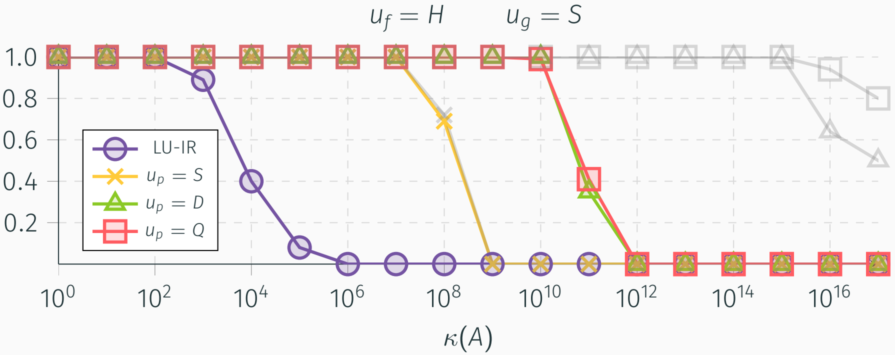
$\Rightarrow$ No compromise by not using Q (maybe not hardware supported)!

Experimental robustness over $\kappa(A)$



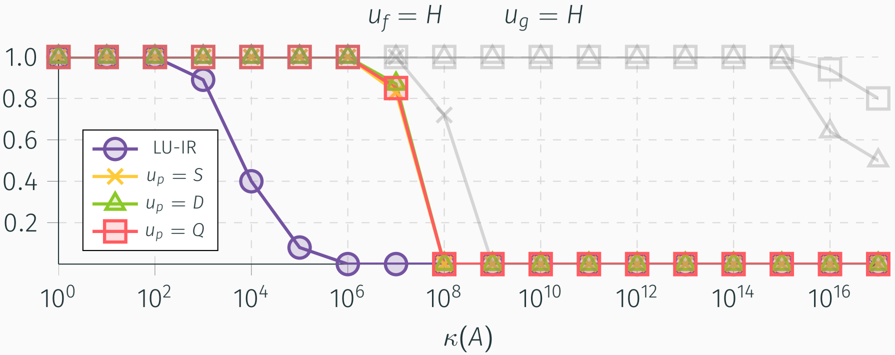
- When $u_p = D \to S$, lose in robustness but still far more robust than LU-IR3.

Experimental robustness over $\kappa(A)$



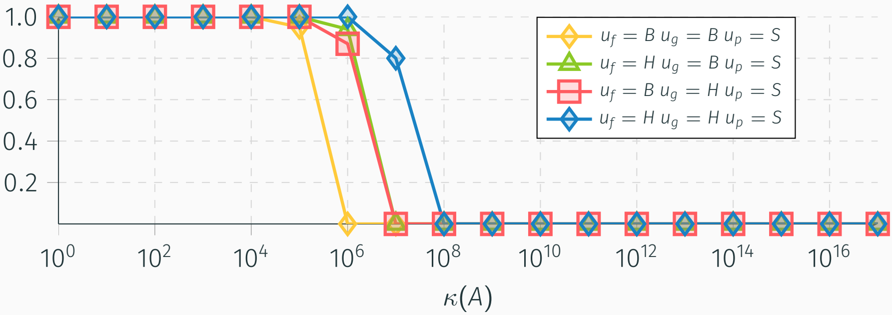
- When $u_g = D \to S$:
  - $u_p = D$ and $u_p = Q$ lose in robustness.
  - $u_p = S$ same as $u_g = D \Rightarrow$ better use $u_g = S$.

Experimental robustness over $\kappa(A)$



- When $u_g = S \to H$, still more robust than LU-IR3 with u_g in really low precision.

Five-precisions combinations

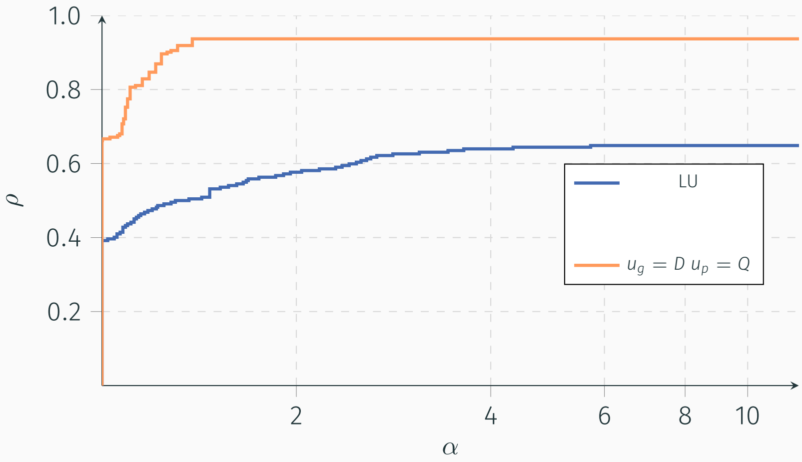


2 **five-precisions** combinations meaningful theoretically :

$$(u_f = B, u = D, u_r = Q, u_g = H, u_p = S) \qquad (u_f = H, u = D, u_r = Q, u_g = B, u_p = S)$$

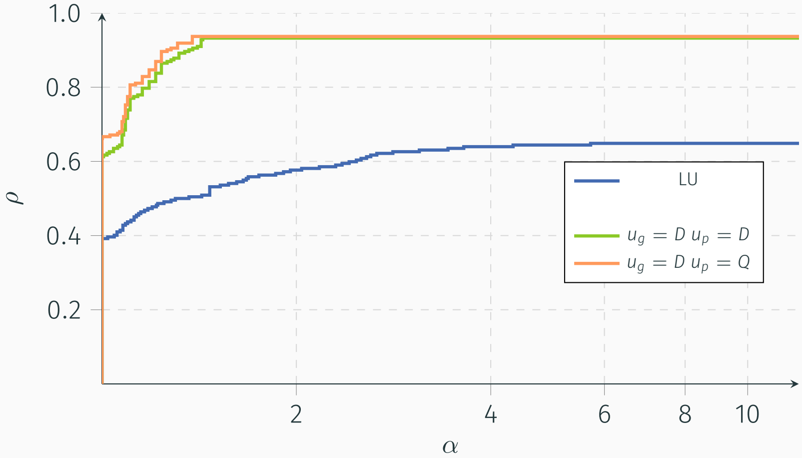
$\Rightarrow$ Tradeoff between 2 four-precisions combinations allowing even finer setup of convergence conditions.

Cumulated number of LU solve calls ($\approx$ nb iterations)



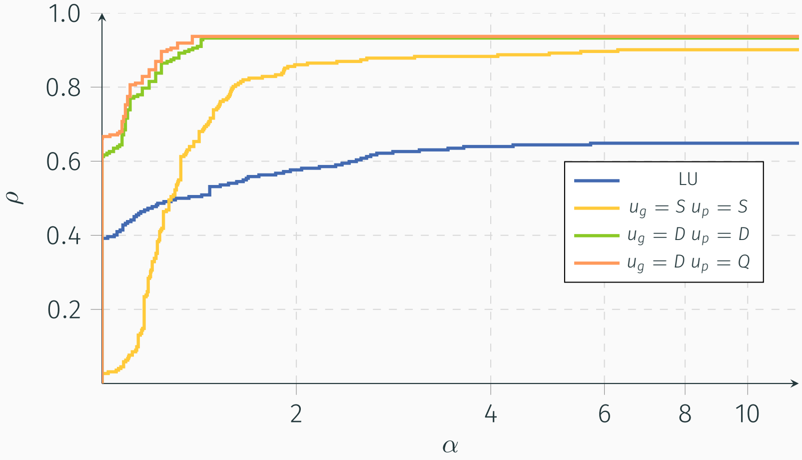
Performance profile on 230 little **Suite Sparse real life** matrices. ρ indicates the % of matrices for which a given combination requires less than α times the number of LU solves required by the best combination. $u_f = H$ fixed.

Cumulated number of LU solve calls (≈ nb iterations)



Performance profile on 230 little **Suite Sparse real life** matrices. ρ indicates the % of matrices for which a given combination requires less than α times the number of LU solves required by the best combination. $u_f = H$ fixed.

Cumulated number of LU solve calls ($\approx$ nb iterations)



Performance profile on 230 little **Suite Sparse real life** matrices. ρ indicates the % of matrices for which a given combination requires less than α times the number of LU solves required by the best combination. $u_f = H$ fixed.

Conclusion

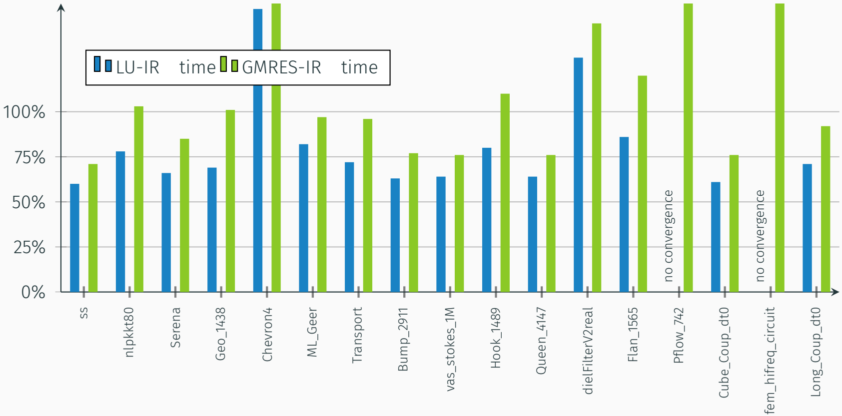
Contributions

- **GMRES-IR5 + error analysis :** high versatility on precisions allowing better fit of precisions combinations according to problem difficulty and hardware.
- **Numerical experiments :** Validate the theoretical convergence condition on hundreds generated and real life matrices.

Future work : High performance parallel implementation within distributed memory for the solution of sparse systems.

Amestoy, Buttari, Higham, L'Excellent, Mary, Vieublé. *"Five precisions GMRES-based iterative refinement"*. In : Submission soon.

Sparse - Time Performance (No MPI - 18 threads)



LU-IR $u_f = S$ **Vs** GMRES-IR $u_f = S$ $u_g = D$ $u_p = D$ normalized by LU direct solver in full D; multifrontal solver MUMPS;

- LU-IR 1.4 –1.7× faster on most of the matrices!

- GMRES-IR slower than LU-IR, but converges on all the matrices ⇒ more robust.