université PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

LISN LABORATOIRE INTERDISCIPLINAIRE DES SCIENCES DU NUMÉRIQUE

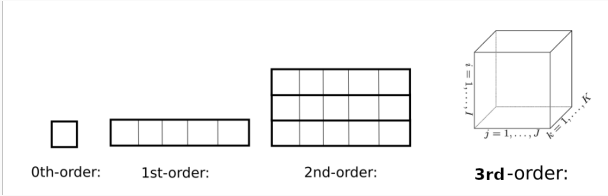
Mixed Precision Iterative Refinement for Low-Rank Matrix and Tensor Approximations

Matthieu Robeyns

Joint work with Marc Baboulin, Oguz Kaya, and Theo Mary

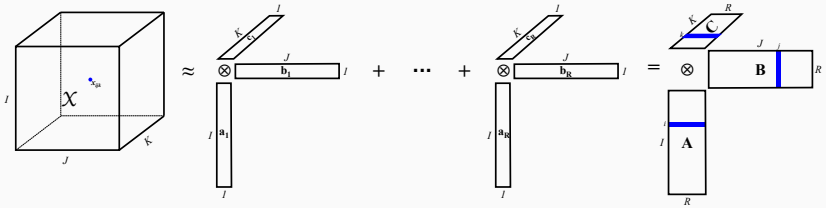
SIAM CSE 2023, March 2, 2023

Tensor definition



- A **tensor**, denoted as $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$, is a multidimensional array of data.
- The **order** of the tensor is its number of dimensions.
- Powerful tool in many applications : data analysis, quantum computing, AI, signal processing, scientific computing, ...
- The size of a tensor is exponential in its order $\Rightarrow$ curse of dimensionality.

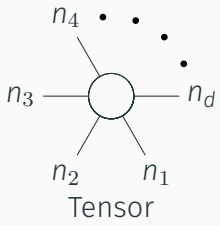
Low-rank approximation

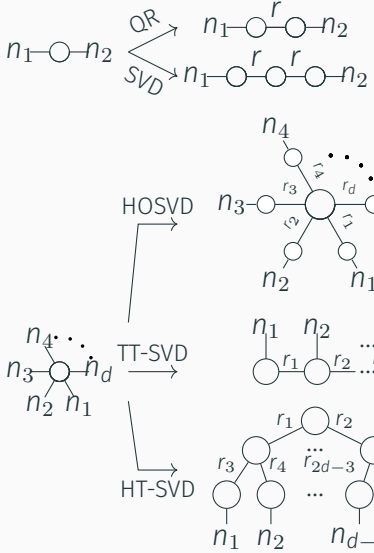


Rank $\vec{r}$ CP decomposition of $\boldsymbol{\mathcal{X}}$

- **Low-rank approximation (LRA)** is the problem of approximating an order d tensor by smaller tensors with **inner dimensions** as the **rank** of decomposition $\Rightarrow$ curve of dimensionality solved.

Different types of low-rank decompositions





Low-rank approximation

Direct methods

Solve the LRA problem using a fixed error threshold, and give a quasi-optimal approximation.

Iterative methods

Solve the LRA problem with a sequence of approximations, each approximation improving upon the previous one. No guarantee of convergence.

Low-rank approximation

Direct methods

Solve the LRA problem using a fixed error threshold, and give a quasi-optimal approximation.

Iterative methods

Solve the LRA problem with a sequence of approximations, each approximation improving upon the previous one. No guarantee of convergence.

- **Low-rank approximation** of tensors heavily uses linear algebra operations. Computationally intensive.

Accelerating tensor computations

⇒ Need HPC methods (parallelism, accelerators) to speed up computations.

		Range	Precision	Speed (Tflop/s)*
fp64	double	$10^{\pm 308}$	1×10^{-16}	32
fp32	single	$10^{\pm 38}$	6×10^{-8}	64
fp16	half	$10^{\pm 5}$	5×10^{-4}	1000
bfloat16		$10^{\pm 38}$	4×10^{-3}	
fp8 (e4m3)	quarter	$10^{\pm 2}$	6×10^{-2}	2000
fp8 (e5m2)		$10^{\pm 5}$	1×10^{-1}	

*on NVIDIA Hopper H100 SXM5

⇒ **Low precision** computations achieve very high performance with resulting **low accuracy**.

Mixed Precision algorithm

- Many LRA applications need an accuracy higher than 16-bit.
- We will use **mixed precision** algorithms with the hope of achieving both high performance and high accuracy.

Proposed approach : **Iterative refinement**

1. Apply method to original input in low precision.
2. Compute the residual in high precision.
3. Apply method to the residual in low precision.
4. Combine (1) and (3) to obtain an approximation in high precision.

Iterative refinement

Input :

$\boldsymbol{\mathcal{X}}$: Tensor or matrix

LRA : Some LRA method

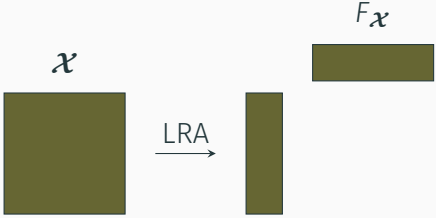
---

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ

2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$

3 : $F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ

4 : $F'_{\boldsymbol{\mathcal{X}}} \leftarrow F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}$

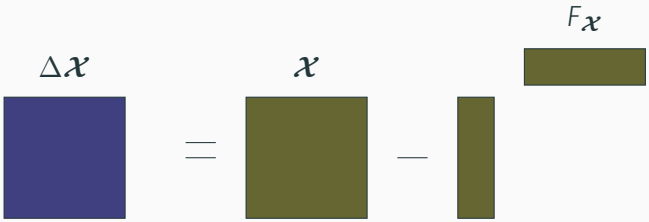


Iterative refinement

Input :
$\mathcal{X}$: Tensor or matrix
LRA : Some LRA method

---

1 : $F_{\mathcal{X}} \leftarrow \text{LRA}(\mathcal{X})$ ▷ In low precision u_ℓ
2 : $\Delta\mathcal{X} \leftarrow \mathcal{X} - \text{Decompress}(F_{\mathcal{X}})$
3 : $F_{\Delta\mathcal{X}} \leftarrow \text{LRA}(\Delta\mathcal{X})$ ▷ In low precision u_ℓ
4 : $F'_{\mathcal{X}} \leftarrow F_{\mathcal{X}} + F_{\Delta\mathcal{X}}$



Iterative refinement

Input :
$\boldsymbol{\mathcal{X}}$: Tensor or matrix
LRA : Some LRA method

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$
3 : $F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
4 : $F'_{\boldsymbol{\mathcal{X}}} \leftarrow F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}$

Iterative refinement

Input :

$\boldsymbol{\mathcal{X}}$: Tensor or matrix

LRA : Some LRA method

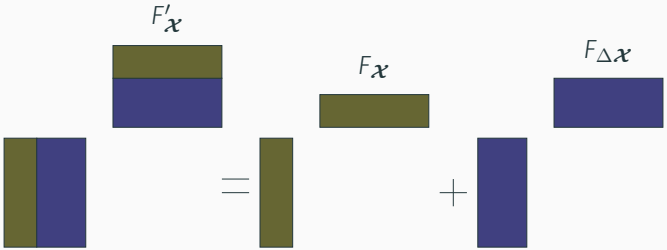
---

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ

2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$

3 : $F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ

4 : $F'_{\boldsymbol{\mathcal{X}}} \leftarrow F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}$

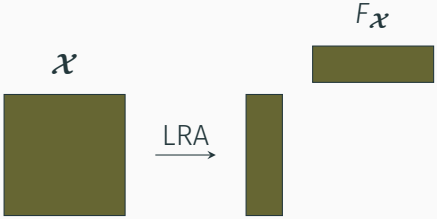


Iterative refinement

Input :
$\boldsymbol{\mathcal{X}}$: Tensor or matrix
LRA : Some LRA method

---

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$
3 : $F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
4 : $F'_{\boldsymbol{\mathcal{X}}} \leftarrow F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}$



Rank analysis :

- $\text{rank}(\boldsymbol{\mathcal{X}}) \leq \vec{r}$.
- $\text{rank}(F_{\boldsymbol{\mathcal{X}}}) \leq \vec{r}$.

Error analysis :

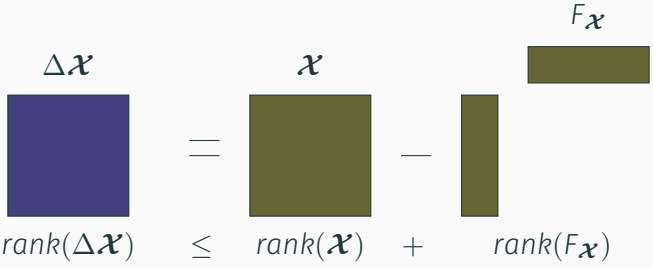
- $\|\boldsymbol{\mathcal{X}} - F_{\boldsymbol{\mathcal{X}}}\| \leq c_1 u_\ell \|\boldsymbol{\mathcal{X}}\|$.

  Where u_ℓ is the low precision unit roundoff.

Iterative refinement

Input :
$\mathcal{X}$: Tensor or matrix
LRA : Some LRA method

1 : $F_{\mathcal{X}} \leftarrow \text{LRA}(\mathcal{X})$ ▷ In low precision u_ℓ
2 : $\Delta\mathcal{X} \leftarrow \mathcal{X} - \text{Decompress}(F_{\mathcal{X}})$
3 : $F_{\Delta\mathcal{X}} \leftarrow \text{LRA}(\Delta\mathcal{X})$ ▷ In low precision u_ℓ
4 : $F'_{\mathcal{X}} \leftarrow F_{\mathcal{X}} + F_{\Delta\mathcal{X}}$



Rank analysis :

- $\text{rank}(\Delta\mathcal{X}) \leq \text{rank}(\mathcal{X}) + \text{rank}(F_{\mathcal{X}})$.

=> $\text{rank}(\Delta\mathcal{X}) \leq 2\vec{r}$.

Error analysis :

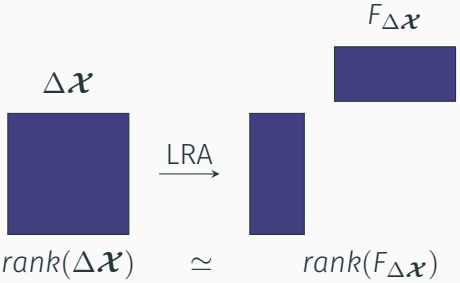
- $\|F_{\mathcal{X}} - \tilde{F}_{\mathcal{X}}\| \leq c_2 u \|F_{\mathcal{X}}\|$, where $\tilde{F}_{\mathcal{X}} = \text{DECOMPRESS}(F_{\mathcal{X}})$, and u is the high precision unit roundoff.

Iterative refinement

Input :
$\boldsymbol{\mathcal{X}}$: Tensor or matrix
LRA : Some LRA method

---

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$
3 : $F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
4 : $F'_{\boldsymbol{\mathcal{X}}} \leftarrow F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}$



Rank analysis :

- $\text{rank}(F_{\Delta\boldsymbol{\mathcal{X}}}) \leq 2\vec{r}$.

Error analysis :

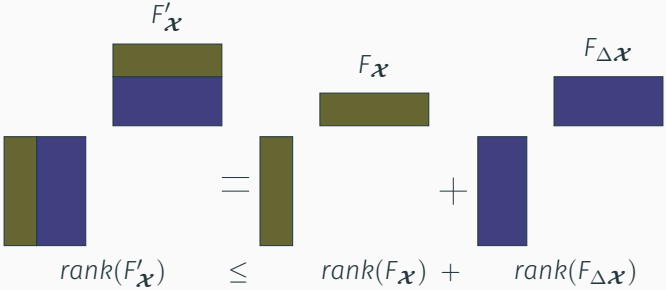
- $\|\Delta\boldsymbol{\mathcal{X}} - F_{\Delta\boldsymbol{\mathcal{X}}}\| \leq c_1 u_\ell \|\Delta\boldsymbol{\mathcal{X}}\|$.

Iterative refinement

Input :
$\boldsymbol{\mathcal{X}}$: Tensor or matrix
LRA : Some LRA method

---

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$
3 : $F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$ ▷ In low precision u_ℓ
4 : $F'_{\boldsymbol{\mathcal{X}}} \leftarrow F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}$



Rank analysis :

=> $\text{rank}(F'_{\boldsymbol{\mathcal{X}}}) \leq 3\vec{r}$.

Error analysis :

=> $\|\boldsymbol{\mathcal{X}} - F'_{\boldsymbol{\mathcal{X}}}\| \leq ((c_1 u_\ell)^2 + c_2 u)\,\|\boldsymbol{\mathcal{X}}\|$.

Iterative refinement

Input :
$\mathcal{X}$: Tensor or matrix
LRA : Some LRA method
n_{iter} : Number of iterations

---

1 : $F_{\mathcal{X}} \leftarrow \text{LRA}(\mathcal{X})$
2 : $\Delta\mathcal{X} \leftarrow \mathcal{X}$
3 : for $i = 1, \ldots, n_{iter}$ do
4 : $\quad \Delta\mathcal{X} \leftarrow \Delta\mathcal{X} - \text{Decompress}(F_{\mathcal{X}})$
5 : $\quad F_{\Delta\mathcal{X}} \leftarrow \text{LRA}(\Delta\mathcal{X})$
6 : $\quad F_{\mathcal{X}} \leftarrow F_{\mathcal{X}} + F_{\Delta\mathcal{X}}$
7 : end for

Theorem

After i iterations, the computed $F_{\mathcal{X}}$ satisfies

$$\|\mathcal{X} - F_{\mathcal{X}}\| \leq (\phi^{(i+1)} + \xi + O(u_\ell u)) \|\mathcal{X}\|$$

- $\phi = O(u_\ell)$ is the convergence speed
- $\xi = O(u)$ is the attainable accuracy

Rank analysis : $\text{rank}(F_{\mathcal{X}}) \leq \sum_{j=1}^{i} 2^j \vec{r}$.

Iterative refinement

Input :
$\boldsymbol{\mathcal{X}}$: Tensor or matrix
LRA : Some LRA method
n_{iter} : Number of iterations

---

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$
2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}}$
3 : for $i = 1, \ldots, n_{\text{iter}}$ do
4 : $\quad \Delta\boldsymbol{\mathcal{X}} \leftarrow \Delta\boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$
5 : $\quad F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$
6 : $\quad F_{\boldsymbol{\mathcal{X}}} \leftarrow \textsc{Recompress}(F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}, \phi^{(i+1)})$
7 : end for

Theorem

After i iterations, the computed $F_{\boldsymbol{\mathcal{X}}}$ satisfies

$$\|\boldsymbol{\mathcal{X}} - F_{\boldsymbol{\mathcal{X}}}\| \leq (\phi^{(i+1)} + \xi + O(u_\ell u)) \|\boldsymbol{\mathcal{X}}\|$$

- $\phi = O(u_\ell)$ is the convergence speed
- $\xi = O(u)$ is the attainable accuracy

Rank analysis : $\text{rank}(F_{\boldsymbol{\mathcal{X}}}) \leq \vec{r}$.

- $\textsc{Recompress}(F,\varepsilon)$: Tighten the ranks of a decomposition F with non-optimal ranks, by finding a lower rank decomposition within its ε vicinity.

  $\Rightarrow$ Should recompress $F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}$ back to rank $\vec{r}$.

Cost Analysis

Input :
- $\mathcal{X}$: Tensor or matrix
- LRA : Some LRA method
- n_{iter} : Number of iterations

---

1 : $F_{\mathcal{X}} \leftarrow \text{LRA}(\mathcal{X})$
2 : $\Delta\mathcal{X} \leftarrow \mathcal{X}$
3 : **for** $i = 1, \ldots, n_{iter}$ **do**
4 : $\quad \Delta\mathcal{X} \leftarrow \Delta\mathcal{X} - \text{Decompress}(F_{\mathcal{X}})$
5 : $\quad F_{\Delta\mathcal{X}} \leftarrow \text{LRA}(\Delta\mathcal{X})$
6 : $\quad F_{\mathcal{X}} \leftarrow \text{RECOMPRESS}(F_{\mathcal{X}} + F_{\Delta\mathcal{X}}, \phi^{(i+1)})$
7 : **end for**

Let $P = \prod dims$ and $S = \sum dims$.
Let r_i be the rank of $F_{\mathcal{X}}$ at iteration i, with $r_i \leq r$ (final rank).

- $\text{cost}(LRA) = c_1 P \sum r_i$ flops ($c_1 \simeq [4 : 10]$).
- $\text{cost}(Decompress) = c_2 P \sum r_i$ flops ($c_2 = 2$).
- $\text{cost}(Recompress) = O(S \sum r_i)$ flops.

$$\text{cost}(LRA) \geq \text{cost}(Decompress) \gg \text{cost}(Recompress)$$

$\downarrow$ low precision

$\downarrow$ high precision

Cost Analysis

Input :
$\boldsymbol{\mathcal{X}}$: Tensor or matrix
LRA : Some LRA method
n_{iter} : Number of iterations

---

1 : $F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\boldsymbol{\mathcal{X}})$
2 : $\Delta\boldsymbol{\mathcal{X}} \leftarrow \boldsymbol{\mathcal{X}}$
3 : for $i = 1, \ldots, n_{iter}$ do
4 : $\quad \Delta\boldsymbol{\mathcal{X}} \leftarrow \Delta\boldsymbol{\mathcal{X}} - \text{Decompress}(F_{\boldsymbol{\mathcal{X}}})$
5 : $\quad F_{\Delta\boldsymbol{\mathcal{X}}} \leftarrow \text{LRA}(\Delta\boldsymbol{\mathcal{X}})$
6 : $\quad F_{\boldsymbol{\mathcal{X}}} \leftarrow \text{RECOMPRESS}(F_{\boldsymbol{\mathcal{X}}} + F_{\Delta\boldsymbol{\mathcal{X}}}, \phi^{(i+1)})$
7 : end for

Two scenarios where the proposed IR can be much faster :

1. Hardware with much faster (x3) low precision arithmetic and/or mixed precision FMA (e.g., GPU tensor cores)
2. $r_i \ll r$ for small i (rapid decay of singular values).

Experimental Setting

- Code developed in MATLAB(2019a) with the following decompositions :
  - **Randomized SVD** (Matrix), [Martinsson & Voronin, 2016],
  - **QRCP** (Matrix),
  - **HOSVD** (Tucker) [De Lathauwer & al., 2000] TENSOR TOOLBOX,
  - **TTSVD** (Tensor-Train) [Oseledets & al., 2011] TT-TOOLBOX,
  - **HTSVD** (Hierarchical-Tensor) [Grasedyck, 2010] HTUCKER_1.2.

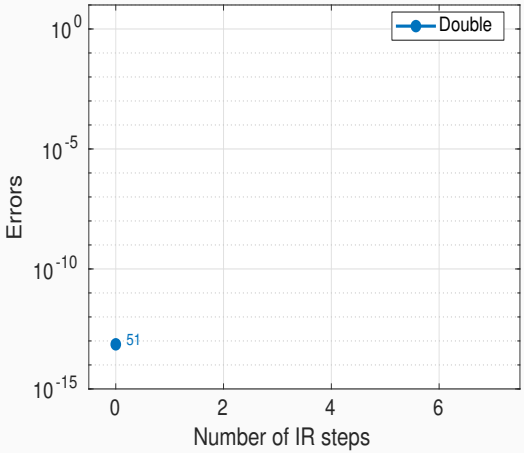
CHOP : to simulate low precision.

Experimental Setting

- Code developed in MATLAB(2019a) with the following decompositions :
  - **Randomized SVD** (Matrix), [Martinsson & Voronin, 2016],
  - **QRCP** (Matrix),
  - **HOSVD** (Tucker) [De Lathauwer & al., 2000] TENSOR TOOLBOX,
  - **TTSVD** (Tensor-Train) [Oseledets & al., 2011] TT-TOOLBOX,
  - **HTSVD** (Hierarchical-Tensor) [Grasedyck, 2010] HTUCKER_1.2.

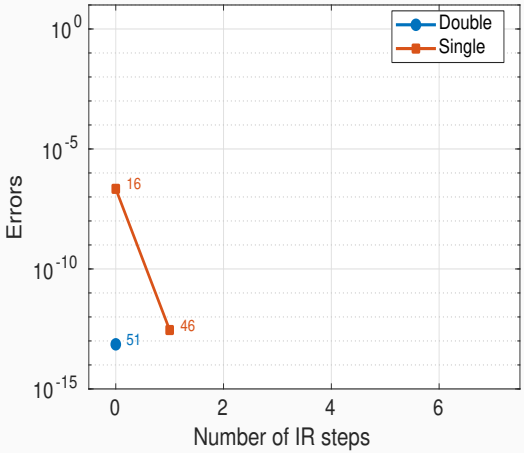
  CHOP : to simulate low precision.
- Our benchmarks involve small matrices ($\simeq 300 \times 300$) and tensors ($\simeq 100 \times 100 \times 100$) randomly generated with a specific decreasing behavior of singular values.

Experiments on matrices



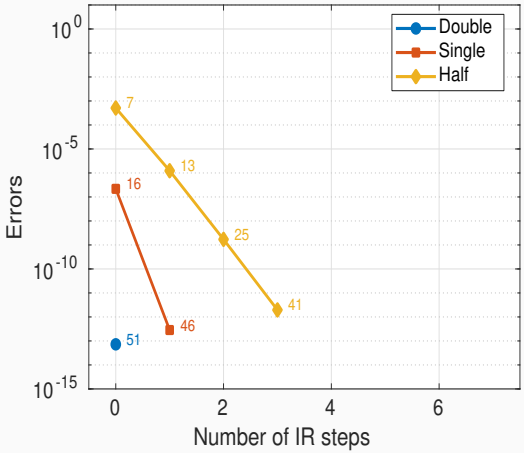
(a) RandSVD computation with a Poisson matrix of size $253x252$

Experiments on matrices



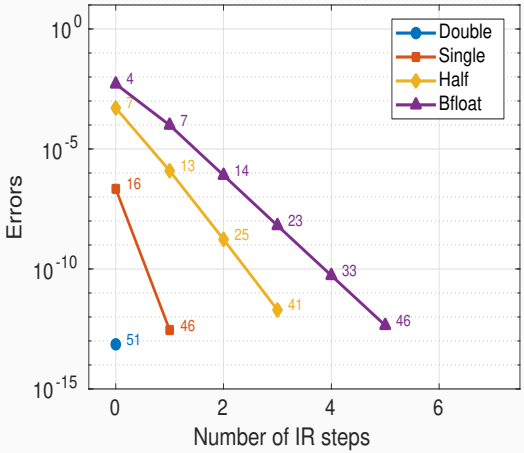
(a) RandSVD computation with a Poisson matrix of size $253x252$

Experiments on matrices



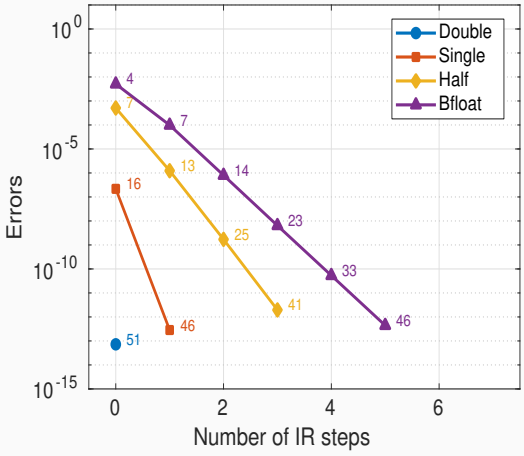
(a) RandSVD computation with a Poisson matrix of size 253×252

Experiments on matrices

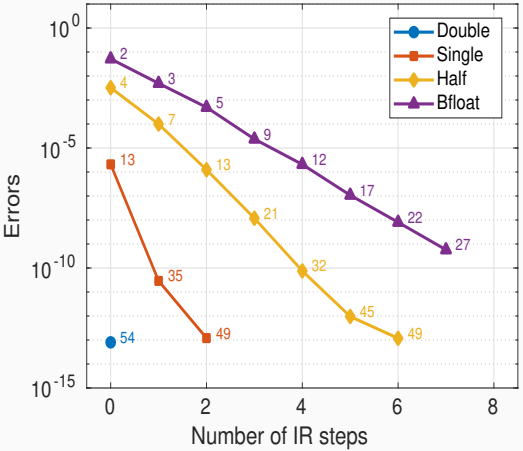


(a) RandSVD computation with a Poisson matrix of size 253×252

Experiments on matrices

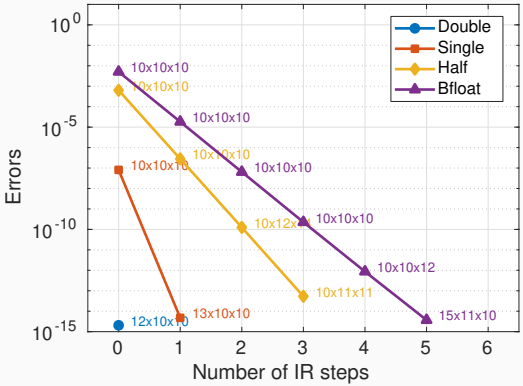


(a) RandSVD computation with a Poisson matrix of size 253×252

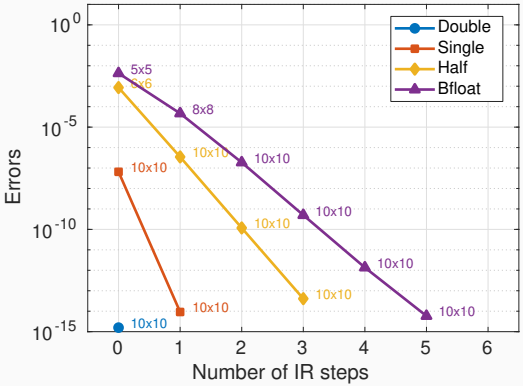


(b) QRCP with the same matrix

Experiments on tensors

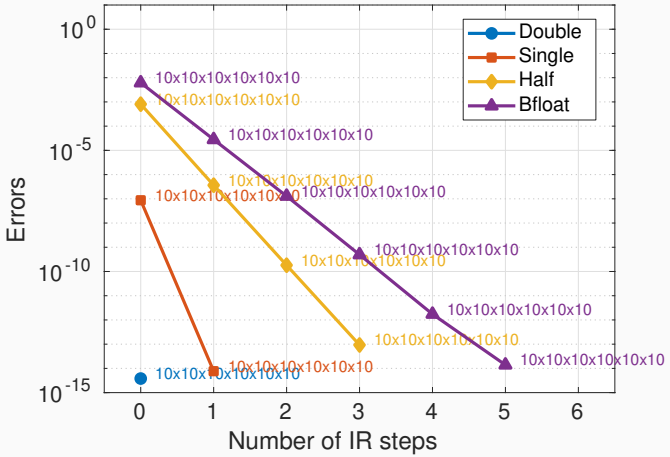


(a) HOSVD with $\vec{r}$ = 10x10x10



(b) TT-SVD with $\vec{r}$ = 10x10

Experiments on tensors



(a) HT-SVD with
$\vec{r}$ = 10x10x10x10x10x10

Conclusion

Contributions

- We propose a **new** iterative refinement method for tensor LRA.
- We achieve high accuracy with most operations in low precision.

Future work

- Test the method on real-life tensor benchmarks.
- Develop a parallel high performance implementation of the method with performance analysis, e.g., on GPUs.

Preprint in preparation : M. Baboulin, O. Kaya, T. Mary, M. Robeyns,
Mixed precision iterative refinement for low-rank matrix and tensor approximations.

THANK YOU ! Questions ?