Optimal Quantization of Rank-One Matrices in Floating-Point Arithmetic

Elisa Riccietti

LIP – École Normale Supérieure de Lyon

RAIM meeting 2025, 3-7 November 2025

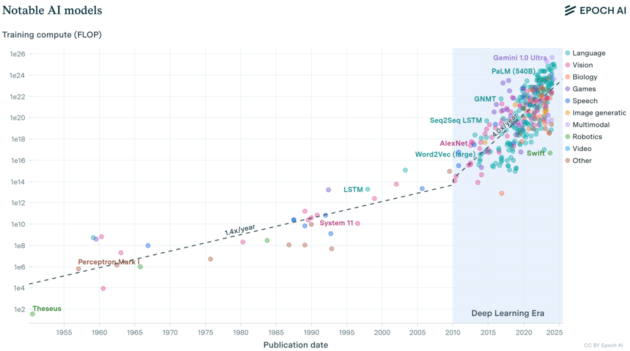
Collaboration

Rémi Gribonval (INRIA - LIP)

Théo Mary (CNRS - LIP6)

Motivation

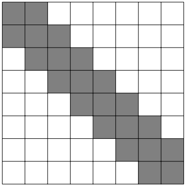
Growing size of models and datasets



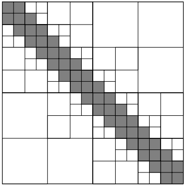
[J. Sevilla et all. 2022]

$\rightarrow$ approximate computing

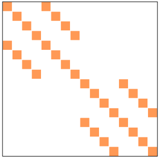
- Low-rank, structured, data sparse matrices



BLR matrix



$\mathcal{H}$-matrix



Butterfly matrix

Sparse matrix factorization

Given a dense matrix A, find *multiple* factors $S_1, S_2, \dots S_L$ such that:

$$A \approx S_1 S_2 \dots S_L$$

where S_i are *sparse* matrices.

Sparse matrix factorization

Given a dense matrix A, find *multiple* factors $S_1, S_2, \dots S_L$ such that:

$$A \approx S_1 S_2 \dots S_L$$

where S_i are *sparse* matrices.

$$\underbrace{A}_{dense} \approx \underbrace{S_1 S_2 \dots S_L}_{sparse} \quad \Rightarrow \quad Ax \approx S_1(S_2(\dots(S_L x)))$$

Sparse matrix factorization

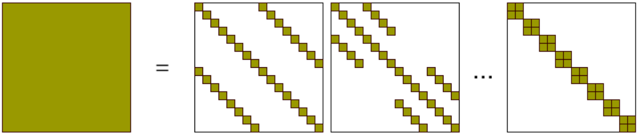
Given a dense matrix A, find *multiple* factors $S_1, S_2, \ldots S_L$ such that:

$$A \approx S_1 S_2 \ldots S_L$$

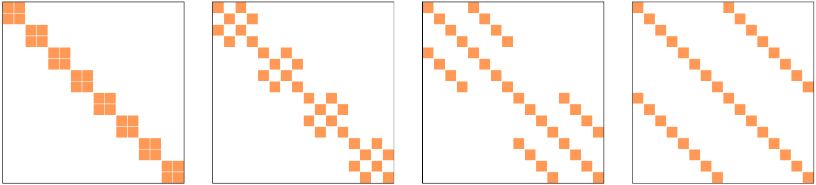
where S_i are *sparse* matrices.

$$\underbrace{A}_{dense} \approx \underbrace{S_1 S_2 \ldots S_L}_{sparse} \quad \Rightarrow \quad Ax \approx S_1(S_2(\ldots(S_L x)))$$

Application: fast transforms, networks compression...



The butterfly factorization



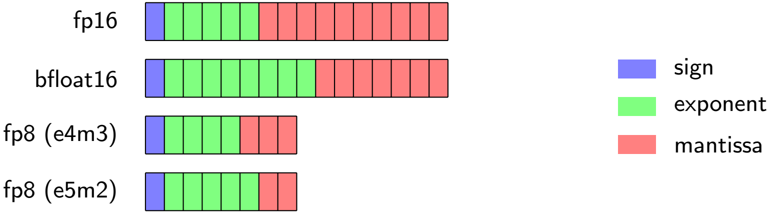
- Butterfly matrices are extremely sparse yet highly expressive, they appear in many fast linear transforms
- Butterfly factorization: given dense A $n \times n$ matrix :

$$A \approx B_1 \dots B_L$$

with $L = \log_2 n \Rightarrow O(n \log n)$ complexity

Approximate computing (II)

- Quantization to low precision floating-point arithmetic



Quantized butterfly factorization

Butterfly factorization + quantization

$$A \approx \hat{B}_1 \dots \hat{B}_L$$

Problem: Given a butterfly factorization $B_1 \dots B_L$, compute $\hat{B}_1 \dots \hat{B}_L$ with optimal error

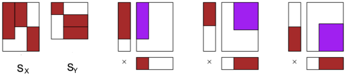
$$\frac{\|B_1 \dots B_L - \hat{B}_1 \dots \hat{B}_L\|}{\|B_1 \dots B_L\|}$$

Key property: optimal two factors quantization

- If $L = 2$ the problem can be optimally solved.
- Given two butterfly factors B_1, B_2

$$B_1 B_2^T = \sum_{i=1}^{n} b_1(:, i) b_2(:, i)^T$$

where the rank-one matrices $b_1(:, i) b_2(:, i)^T$ have disjoint support.

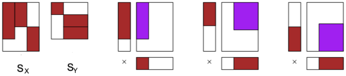


Key property: optimal two factors quantization

- If $L = 2$ the problem can be optimally solved.
- Given two butterfly factors B_1, B_2

$$B_1 B_2^T = \sum_{i=1}^{n} b_1(:, i) b_2(:, i)^T$$

where the rank-one matrices $b_1(:, i) b_2(:, i)^T$ have disjoint support.



- We can optimally quantize the product $B_1 B_2^T$ by quantizing each $b_1(:, i) b_2(:, i)^T$ optimally and independently

Part I

Quantization of rank-one matrices

Quantization of rank-one matrices

Goal: quantize the rank-one matrix

$$xy^T \to \widehat{x}\widehat{y}^T \qquad (x \in \mathbb{R}^m,\ y \in \mathbb{R}^n)$$

where the coefficients of $\widehat{x}$, $\widehat{y}$ have t bits of mantissa

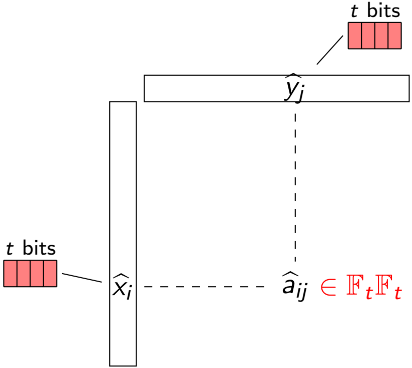
- The standard approach uses round-to-nearest (RTN) and leads to an error of order $u = 2^{-t}$: if $\widehat{x} = \text{round}(x)$, $\widehat{y} = \text{round}(y)$ then

$$\begin{aligned} \|\widehat{x} - x\| &\leq u\|x\| \\ \|\widehat{y} - y\| &\leq u\|y\| \\ \Rightarrow \|\widehat{x}\widehat{y}^T - xy^T\| &\leq (2u + u^2)\|x\|\|y\| \end{aligned}$$

- We will show this is far from optimal!

An analogy

Let $\mathbb{F}_t$ be the set of t-bit floating-point numbers and
$\mathbb{F}_t\mathbb{F}_t = \{a = xy,\ x \in \mathbb{F}_t,\ y \in \mathbb{F}_t\}$

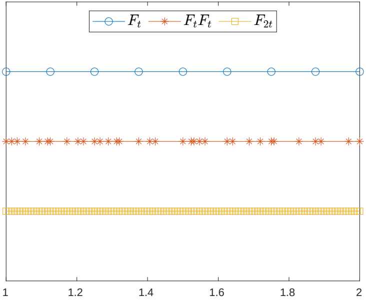


- What we really care about is the accuracy of $\widehat{a}_{ij} = \widehat{x}_i\widehat{y}_j$
- Think of multiword arithmetic: $a \approx \widehat{x} + \widehat{y}$ with $\widehat{x} = \mathrm{round}(a)$ and $\widehat{y} = \mathrm{round}(a - \widehat{x}) \to 2t$-bit accuracy
- What about $a = xy$? (Which $\widehat{x}$,$\widehat{y}$ yields the best approximation $\widehat{x}\widehat{y}$?)

The set $\mathbb{F}_t\mathbb{F}_t$

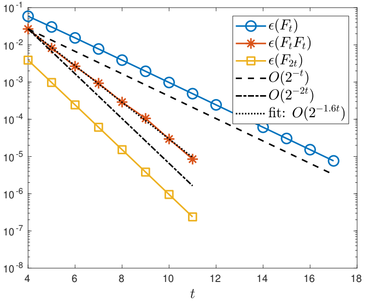
- Let $\mathbb{F}_t$ be the set of t-bit floating-point numbers. We are interested in the set

$$\mathbb{F}_t\mathbb{F}_t = \{a = xy,\ x \in \mathbb{F}_t,\ y \in \mathbb{F}_t\}$$



- No closed form expression of its elements, but we can simply enumerate all of them for small t

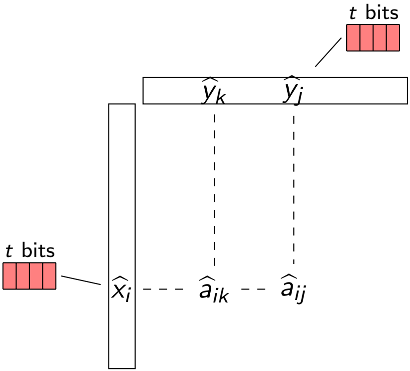
The set $\mathbb{F}_t\mathbb{F}_t$



- $\epsilon(S) = \sup_{z\neq 0} \frac{d(z,S)}{|z|}$: the worst-case relative error of quantizing an element $z \in \mathbb{R}$ on S
- $\epsilon(\mathbb{F}_t) = \frac{2^{-t}}{1+2^{-t}}$

$\Rightarrow$ $\epsilon(\mathbb{F}_t\mathbb{F}_t)$ error of order $2^{-1.6t}$

A constrained combinatorial problem



We don't just have one scalar, but a rank-one matrix $\Rightarrow$

- We have constraints: $\widehat{x}_i$ must be the same in $\widehat{a}_{ij} = \widehat{x}_i \widehat{y}_j$ and $\widehat{a}_{ik} = \widehat{x}_i \widehat{y}_k$
- How can we find the optimal quantization? Combinatorial problem!

$$\min_{\widehat{x} \in \mathbb{F}_t^m, \widehat{y} \in \mathbb{F}_t^n} \|xy^T - \widehat{x}\widehat{y}^T\|$$

Intuition to simplify the problem

In exact arithmetic:

$$xy^T = (\lambda x)\left(\frac{1}{\lambda} y\right)^T$$

In floating point arithmetic

$$\mathrm{round}(xy^T) \neq \mathrm{round}(\lambda x)\,\mathrm{round}\left(\frac{1}{\lambda} y\right)^T$$

- Can we find the optimal scaling λ^*?
- Can we reduce the problem to a scalar problem?

Characterization of the optimum

Theorem

$$\min_{\widehat{x}\in\mathbb{F}_t^m,\widehat{y}\in\mathbb{F}_t^n} \|xy^T - \widehat{x}\widehat{y}^T\| = \min_{\lambda\in\mathbb{R}} \|xy^T - \mathrm{round}(\lambda x)\,\mathrm{round}(\mu(\lambda)y)^T\|$$

The optimal quantization $\widehat{x}\widehat{y}^T$ is given by

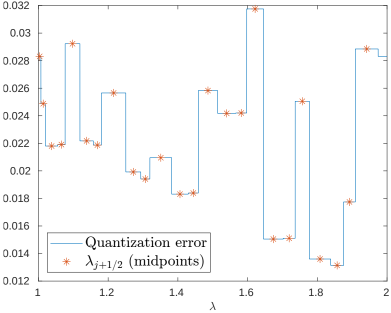
$$\widehat{x} = \mathrm{round}(\lambda x)$$
$$\widehat{y} = \mathrm{round}(\mu(\lambda)y^T)$$

where $\lambda \in \mathbb{R}$ and $\mu(\lambda) = \frac{x^T\widehat{x}}{\|\widehat{x}\|^2}$.

- It suffices to find the optimal λ to find the optimal $\widehat{x}\widehat{y}^T$!

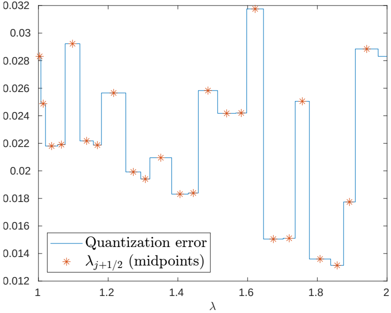
Finding λ

- How do we find the optimal $\lambda \in \mathbb{R}$?
- The optimum is stable under sign flip and multiplication by powers of two $\rightarrow$ restrict the search to $\lambda \in [1, 2]$
- Only a finite number of values of λ change the value of round(λx). Denoting these "breakpoints" as λ_j, we can enumerate the midpoints $\lambda_{j+1/2} = (\lambda_j + \lambda_{j+1})/2$



Finding λ

- How do we find the optimal $\lambda \in \mathbb{R}$?
- The optimum is stable under sign flip and multiplication by powers of two $\rightarrow$ restrict the search to $\lambda \in [1, 2]$
- Only a finite number of values of λ change the value of round(λx). Denoting these "breakpoints" as λ_j, we can enumerate the midpoints $\lambda_{j+1/2} = (\lambda_j + \lambda_{j+1})/2$

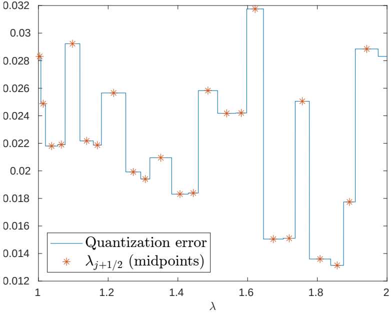


Algorithm:

- Build the set of midpoints
- For each midpoint $\lambda_{j+1/2}$:
  - Build $\widehat{x} = \text{round}(\lambda_{j+1/2} x)$
  - Compute $\mu(\widehat{x}) = x^T \widehat{x} / \|\widehat{x}\|^2$
  - Build $\widehat{y} = \text{round}(\mu y)$
  - Test the accuracy of $\widehat{x}\widehat{y}^T$

Finding λ

- How do we find the optimal $\lambda \in \mathbb{R}$?
- The optimum is stable under sign flip and multiplication by powers of two $\to$ restrict the search to $\lambda \in [1, 2]$
- Only a finite number of values of λ change the value of round(λx). Denoting these "breakpoints" as λ_j, we can enumerate the midpoints $\lambda_{j+1/2} = (\lambda_j + \lambda_{j+1})/2$

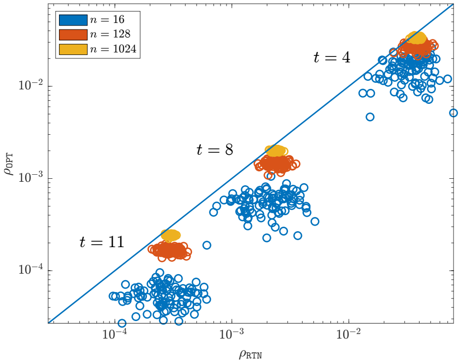


Algorithm:

- Build the set of midpoints
- For each midpoint $\lambda_{j+1/2}$:
  - Build $\widehat{x} = \text{round}(\lambda_{j+1/2} x)$
  - Compute $\mu(\widehat{x}) = x^T \widehat{x} / \|\widehat{x}\|^2$
  - Build $\widehat{y} = \text{round}(\mu y)$
  - Test the accuracy of $\widehat{x}\widehat{y}^T$

$O(mn2^t)$ complexity $\Rightarrow$ tractable for large matrices and low precisions

Experiments



$$\rho_{OPT} = \|xy^T - \widehat{x}\widehat{y}^T\| / \|xy^T\|,$$
$$\rho_{RTN} = \|xy^T - \mathrm{round}(x)\,\mathrm{round}(y)^T\| / \|xy^T\|.$$

100 randomly chosen couples $(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$

Part II

Application to butterfly factorization

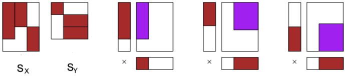
Aim: Given $B_1 \dots B_L$, compute $\hat{B}_1 \dots \hat{B}_L$

- Key property: optimal two factor quantization
- For any partial product XY^T of consecutive factors

$$B_1 \dots B_j \underbrace{B_{j+1} \dots B_k}_{X} \underbrace{B_{k+1} \dots B_\ell}_{Y^T} B_{\ell+1} \dots B_L$$

$$XY^T = \sum_{i=1}^{n} x_i y_i^T$$

where the rank-one matrices $x_i y_i^T$ have disjoint support.



Quantization of butterfly factorization

- We can use our optimal algorithm to quantize each $x_i y_i^T$ optimally and independently: $\widehat{x_i} = \text{round}(\lambda_i x_i)$, $\widehat{y_i} = \text{round}(\mu_i y_i)$
- We then obtain

$$\widehat{X} = \text{round}(X\Lambda), \quad \Lambda = \text{diag}(\lambda_i)$$
$$\widehat{Y} = \text{round}(YM), \quad M = \text{diag}(\mu_i)$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$B_1 \, B_2 \, B_3 \, B_4 \ldots B_L$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{B_1 \, B_2}_{XY^T} \, B_3 \, B_4 \ldots B_L$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1 \, \widehat{B}_2}_{XY^T} \, B_3 \, B_4 \ldots B_L$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1 \, \widehat{B}_2}_{XY^T} \underbrace{B_3 \, B_4}_{XY^T} \ldots B_L$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1 \, \widehat{B}_2}_{XY^T} \underbrace{\widehat{B}_3 \, \widehat{B}_4}_{XY^T} \ldots B_L$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1\,\widehat{B}_2}_{XY^T}\underbrace{\widehat{B}_3\,\widehat{B}_4}_{XY^T}\ldots\underbrace{\widehat{B}_L}_{RTN}$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1\,\widehat{B}_2}_{XY^T}\underbrace{\widehat{B}_3\,\widehat{B}_4}_{XY^T}\ldots\underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$B_1\,B_2\,B_3\ldots B_{L-1}B_L$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1 \, \widehat{B}_2}_{XY^T} \underbrace{\widehat{B}_3 \, \widehat{B}_4}_{XY^T} \dots \underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$\underbrace{B_1}_{X} \underbrace{B_2 \, B_3 \dots B_{L-1} B_L}_{Y^T}$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1\,\widehat{B}_2}_{XY^T}\,\underbrace{\widehat{B}_3\,\widehat{B}_4}_{XY^T}\ldots\underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$\underbrace{\widehat{B}_1}_{X}\;\underbrace{\color{red}{M_2}\,B_2\,B_3\ldots B_{L-1}\,B_L}_{Y^T}$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1\,\widehat{B}_2}_{XY^T}\,\underbrace{\widehat{B}_3\,\widehat{B}_4}_{XY^T}\ldots\,\underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$\underbrace{\widehat{B}_1}_{X}\;\underbrace{\underbrace{\color{red}{M_2}B_2}_{X}\,\underbrace{B_3\ldots B_{L-1}B_L}_{Y^T}}_{Y^T}$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1 \widehat{B}_2}_{XY^T} \underbrace{\widehat{B}_3 \widehat{B}_4}_{XY^T} \dots \underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$\underbrace{\widehat{B}_1}_{X} \underbrace{\underbrace{\widehat{B}_2}_{X} \underbrace{\color{red}{M_3} B_3 \dots B_{L-1} B_L}_{Y^T}}_{Y^T}$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1 \widehat{B}_2}_{XY^T} \underbrace{\widehat{B}_3 \widehat{B}_4}_{XY^T} \ldots \underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$\underbrace{\widehat{B}_1}_{X} \underbrace{\underbrace{\widehat{B}_2}_{X} \underbrace{\widehat{B}_3 \ldots \underbrace{\color{red}{M_{L-1}}B_{L-1}}_{X} \underbrace{B_L}_{Y^T}}_{Y^T}}_{Y^T}$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

$$\underbrace{\widehat{B}_1 \widehat{B}_2}_{XY^T} \underbrace{\widehat{B}_3 \widehat{B}_4}_{XY^T} \dots \underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$\underbrace{\widehat{B}_1}_{X} \underbrace{\underbrace{\widehat{B}_2}_{X} \underbrace{\widehat{B}_3 \dots \underbrace{\widehat{B}_{L-1}}_{X} \underbrace{\widehat{B}_L}_{Y^T}}_{Y^T}}_{Y^T}$$

Heuristics for the L-factor butterfly quantization

When $L > 2$, need heuristics to decide how to order/group the factors

- Pairwise heuristic:

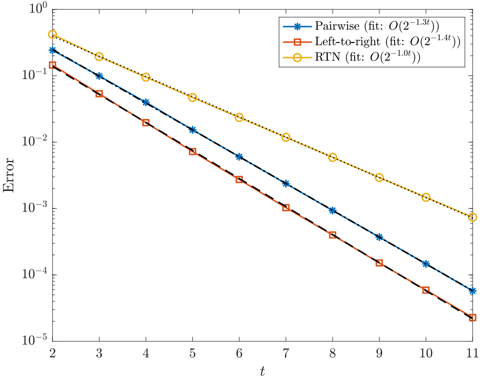
$$\underbrace{\widehat{B}_1 \widehat{B}_2}_{XY^T} \underbrace{\widehat{B}_3 \widehat{B}_4}_{XY^T} \dots \underbrace{\widehat{B}_L}_{RTN}$$

- Left-to-right heuristic:

$$\underbrace{\widehat{B}_1}_{X} \underbrace{\underbrace{\widehat{B}_2}_{X} \underbrace{\widehat{B}_3 \dots \underbrace{\widehat{B}_{L-1}}_{X} \underbrace{\widehat{B}_L}_{Y^T}}_{Y^T}}_{Y^T}$$

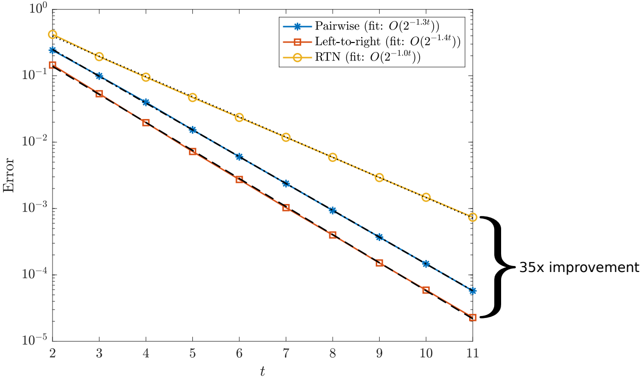
- L2R more expensive because it densifies the factors
- If $L > 2$ the optimality is lost

Experimental results



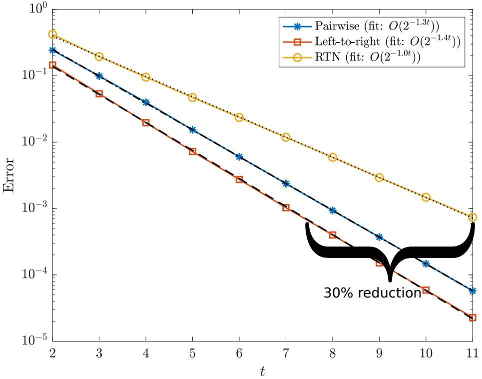
- Randomly generated butterfly factors

Experimental results



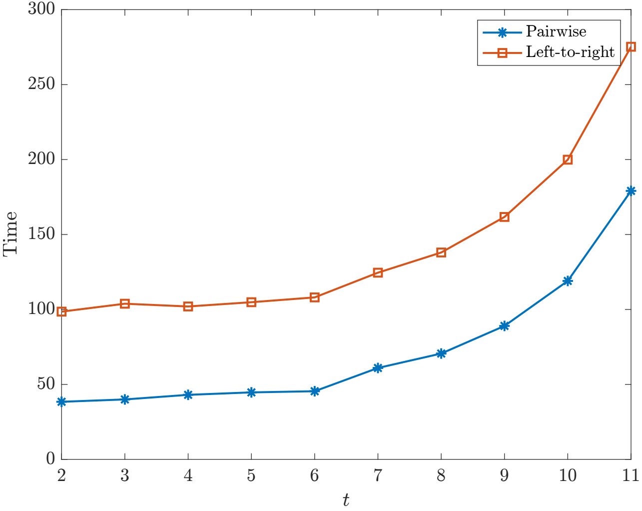
- Randomly generated butterfly factors
- Significant accuracy improvement...

Experimental results



- Randomly generated butterfly factors
- Significant accuracy improvement...
- ...or, equivalently, can reduce storage by about 30% with no loss of accuracy

Experimental results



Conclusion

Key results:

- Characterized optimal quantization of xy^T as $\mathrm{round}(\lambda x)\,\mathrm{round}(\mu y)^T$
- Proposed algorithm to find the optimal λ in $O(mn2^t)$ complexity
- Proposed two heuristics to apply method to butterfly factorization and obtained storage reductions of 30% with no loss of accuracy

Limitation: most of the fast transforms involving butterflies are complex-valued $\rightarrow$ Maël Chaumette's talk tomorrow !

Perspectives: Butterfly matrices are only one possible application, many other perspectives: rank-r matrices, tensors, DNNs, . . .

In practice: the FAμST library

FAμST library: an implementation of the hierarchical algorithm, fast GPU matrix vector multiplication of butterfly matrices, quantization algorithm in C++ via Python and Matlab wrappers FAμST 3.25 toolbox at `https://faust.inria.fr/`.

To know more:

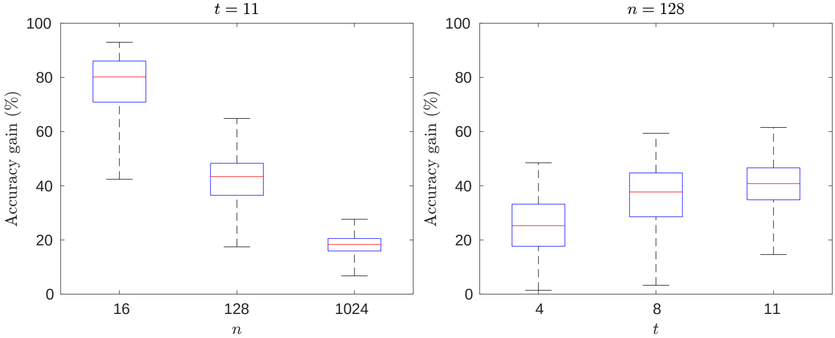
R. Gribonval, T. Mary, E. Riccietti (2024), Optimal quantization of rank-one matrices in floating point arithmetic - with applications to butterfly factorizations, in revision for SISC.

Q.-T. Le, E. Riccietti, and R. Gribonval (2023), Spurious Valleys, Spurious Minima and NP-hardness of Sparse Matrix Factorization With Fixed Support, SIMAX.

L. Zheng, E. Riccietti, and R. Gribonval (2023), Efficient Identification of Butterfly Sparse Matrix Factorizations, SIMODS.

Q.-T. Le, L. Zheng, E. Riccietti, and R. Gribonval (2022), Fast learning of fast transforms, with guarantees, ICASSP 2022

Experiments



$$100 \times \left(1 - \frac{\rho_{OPT}}{\rho_{RTN}}\right)$$

Experiments