Accelerating Nonnegative-X by extrapolation

X $\in$ {Least Square, Matrix Factorization, Tensor Factorization}

Andersen Ang

Mathématique et recherche opérationnelle
UMONS, Belgium

Email: manshun.ang@umons.ac.be Homepage: angms.science

International Conference on Continuous Optimization
August 7, 2019 Berlin, Germany

Outline

Non-negative Matrix Factorization (NMF)

- **Given** matrix $\mathbf{M} \in \mathbb{R}_+^{m \times n}$, positive integer r.
- **Find** matrices $\mathbf{U} \in \mathbb{R}_+^{m \times r}, \mathbf{V} \in \mathbb{R}_+^{r \times n}$ s.t. $\mathbf{M} = \mathbf{UV}$.
- Everything is non-negative.

Non-negative Matrix Factorization (NMF)

$$[\mathbf{U}, \mathbf{V}] = \underset{\mathbf{U}\geq\mathbf{0},\mathbf{V}\geq\mathbf{0}}{\operatorname{argmin}} \|\mathbf{M} - \mathbf{U}\mathbf{V}\|_F.$$

- $\geq$ are element-wise (not positive semi-definite)
- NTF (upgrade)
- NNLS (downgrade)

Question : Why study these problems?
Short answer : They are useful.

Setting

- Always with non-negativity constraint ≥ 0
- Always Quadratic problems[1]
- Always Euclidean (Frobenius) norm
- Always work on matrices (for NMF, NTF)
  i.e. no vec$(\mathbf{X})$, dimension explosion

- I am sorry :
  - single-machine algorithm : no parallelisation
  - deterministic algorithm : no randomization, no compression, no sketching, no projection
  - heuristic : no theoretical convergence result (very difficult)
  - algorithmic : not on applications (there are tons of them)

[1] No other function or divergence in this talk.

$$(\mathcal{P}) \quad : \min_{\mathbf{U},\mathbf{V}} f(\mathbf{U},\mathbf{V}) = \|\mathbf{M} - \mathbf{U}\mathbf{V}\|^2$$

Algorithm Block Coordinate Descent[2]

Input: $\mathbf{M} \in \mathbb{R}_+^{m\times n}$, $r \in \mathbb{N}$, initialization $\mathbf{U} \in \mathbb{R}_+^{m\times r}$, $\mathbf{V} \in \mathbb{R}_+^{r\times n}$

Output: $\mathbf{U}, \mathbf{V}$

1: **for** $k = 1, 2, \ldots$ **do**

2: $\quad \mathbf{U}^{k+1} = \underset{\mathbf{U}\geq 0}{\operatorname{argmin}} f(\mathbf{U}, \mathbf{V}^k)$, initialized at $\mathbf{U}^k$

3: $\quad \mathbf{V}^{k+1} = \underset{\mathbf{V}\geq 0}{\operatorname{argmin}} f(\mathbf{U}^{k+1}, \mathbf{V})$, initialized at $\mathbf{V}^k$

4: **end for**

We have non-increasing sequence

$$f(\mathbf{U}^{k+1}, \mathbf{V}^{k+1}) \leq f(\mathbf{U}^{k+1}, \mathbf{V}^k) \leq f(\mathbf{U}^k, \mathbf{V}^k).$$

(Actually not enough, need sufficient decrease condition)

[2] Other names : Gauss-Seidel iteration, alternating minimization (for 2 blocks)

Extrapolated BCD

$$(\mathcal{P}) \ : \min_{\mathbf{U},\mathbf{V}} f(\mathbf{U},\mathbf{V}) = \|\mathbf{M}-\mathbf{U}\mathbf{V}\|^2$$

Algorithm Heuristic Extrapolation with Restarts (HER)

Input: $\mathbf{M} \in \mathbb{R}_+^{m\times n}$, $r, \mathbf{U}, \mathbf{V}$, $\hat{\mathbf{U}} = \mathbf{U}, \hat{\mathbf{V}} = \mathbf{V}$

Output: $\mathbf{U}, \mathbf{V}$

1: **for** $k = 1, 2, \ldots$ **do**

2: $\quad \mathbf{U}^{k+1} = \underset{\mathbf{U}\geq 0}{\operatorname{argmin}} f(\mathbf{U}, \hat{\mathbf{V}}^k)$, initialized at $\mathbf{U}^k$

3: $\quad$ Extrapolate[$\mathbf{U}$] : $\hat{\mathbf{U}}^{k+1} = \mathbf{U}^{k+1} + \beta_k(\mathbf{U}^{k+1} - \mathbf{U}^k)$.

4: $\quad \mathbf{V}^{k+1} = \underset{\mathbf{V}\geq 0}{\operatorname{argmin}} f(\hat{\mathbf{U}}^{k+1}, \mathbf{V})$, initialized at $\mathbf{V}^k$.

5: $\quad$ Extrapolate[$\mathbf{V}$] : $\hat{\mathbf{V}}^{k+1} = \mathbf{V}^{k+1} + \beta_k(\mathbf{V}^{k+1} - \mathbf{V}^k)$.

6: $\quad$ Restarts (safe guard mechanism) if needed.

7: **end for**

- Extrapolation may destroy the non-increasing sequence property
- Instead of 2-3-4-5, can do 2-4-3-5
- How to do 6

Outline

Algorithm Heuristic Extrapolation with Restarts (HER)

Input: $\mathbf{M} \in \mathbb{R}_+^{m\times n}$, $r, \mathbf{U}, \mathbf{V}, \hat{\mathbf{U}}, \hat{\mathbf{V}}$,
$\beta_0 \in [0,1]$, $\gamma \geq 1$, $\bar{\gamma} \geq 1, \eta \geq 1, \bar{\beta}_0 = 1$

1: **for** $k = 1, 2, \ldots$ **do**
2: $\quad \mathbf{U}^{k+1} = \underset{\mathbf{U}\geq 0}{\operatorname{argmin}} f(\mathbf{U}, \hat{\mathbf{V}}^k)$, initialized at $\mathbf{U}^k$
3: $\quad \hat{\mathbf{U}}^{k+1} = \mathbf{U}^{k+1} + \beta_k(\mathbf{U}^{k+1} - \mathbf{U}^k)$
4: $\quad \mathbf{V}^{k+1} = \underset{\mathbf{V}\geq 0}{\operatorname{argmin}} f(\hat{\mathbf{U}}^{k+1}, \mathbf{V})$, initialized at $\mathbf{V}^k$.
5: $\quad \hat{\mathbf{V}}^{k+1} = \mathbf{V}^{k+1} + \beta_k(\mathbf{V}^{k+1} - \mathbf{V}^k)$
6: $\quad \hat{e}^{k+1} = f(\hat{\mathbf{U}}^{k+1}, \mathbf{V}^{k+1})$
7: $\quad$ **if** $\hat{e}^{k+1} > \hat{e}^k$
$\quad\quad$ Restarts
$\quad\quad$ Decay β_k, update $\bar{\beta}_k$
8: $\quad$ **else** $(\hat{e}^{k+1} \leq \hat{e}^k)$
$\quad\quad$ Grow β_k, update $\bar{\beta}_k$
9: $\quad$ **end if**
10: **end for**

Important: the argmin is not really necessary. i.e. It can be inexact BCD.

Facts

- NMF is non-cvx problem
- Direct application of Nesterov's β gives erratic convergence behaviour Mitchell, et al. "Nesterov Acceleration of Alternating Least Squares for Canonical Tensor Decomposition." arXiv:1810.05846 (2018)

Why heuristics?

- Non-cvx problem is hard :0)
- No better idea :0)
- Currently no convergence analysis (even for NNLS)

What's good ?

- Just a parameter tuning
- Easy implementation
- extension to other models : exact / inexact BCD
- (Empirical) Improvement on convergence speed

Computing β by restarts

With $\bar{\beta}_1 = 1$ and $\beta_0 \in [0, 1]$, update β as

$$\beta_{k+1} = \begin{cases} \min\{\gamma\beta_k, \bar{\beta}\} & \text{if } \hat{e}^k \leq \hat{e}^{k-1} \\ \dfrac{\beta_k}{\eta} & \text{if } \hat{e}^k > \hat{e}^{k-1} \end{cases}$$

Also update $\bar{\beta}_k$

$$\bar{\beta}_{k+1} = \begin{cases} \min\{\bar{\gamma}\bar{\beta}_k, 1\} & \text{if } \hat{e}^k \leq \hat{e}^{k-1} \text{ and } \bar{\beta}_k < 1 \\ \beta_k & \text{if } \hat{e}^k > \hat{e}^{k-1} \end{cases}.$$

The idea of updating β_k

Case 1. "error" decreases : $\hat{e}^k \leq \hat{e}^{k-1}$

- Means the current β is "good"

The idea of updating β_k

Case 1. "error" decreases : $\hat{e}^k \leq \hat{e}^{k-1}$

- Means the current β is "good"
- Be more ambitious on next extrapolation
  - i.e., make β larger
  - How : multiplying it with a growth factor $\gamma > 1$

$$\beta_{k+1} = \beta_k \gamma$$

The idea of updating β_k

Case 1. "error" decreases : $\hat{e}^k \leq \hat{e}^{k-1}$

- Means the current β is "good"
- Be more ambitious on next extrapolation
  - i.e., make β larger
  - How : multiplying it with a growth factor $\gamma > 1$

$$\beta_{k+1} = \beta_k \gamma$$

- Growth of β cannot be indefinite : put a ceiling
  - How :

$$\beta_{k+1} = \min\{\beta_k \gamma, \bar{\beta}_k\}$$

  - $\bar{\beta}$ is also updated with a growth factor $\bar{\gamma}$ with ceiling 1

The idea of updating β_k

Case 2. "error" increases : $\hat{e}^k > \hat{e}^{k-1}$

- Means the current β value is "bad" (too large)

The idea of updating β_k

Case 2. "error" increases : $\hat{e}^k > \hat{e}^{k-1}$

- Means the current β value is "bad" (too large)
- Be less ambitious on the next extrapolation
  - i.e., make β smaller
  - How : divide it with a decay factor $\eta > 1$

$$\beta_{k+1} = \frac{\beta_k}{\eta}$$

The idea of updating β_k

Case 2. "error" increases : $\hat{e}^k > \hat{e}^{k-1}$

- Means the current β value is "bad" (too large)
- Be less ambitious on the next extrapolation
  - i.e., make β smaller
  - How : divide it with a decay factor $\eta > 1$

$$\beta_{k+1} = \frac{\beta_k}{\eta}$$

- As f is continuous and smooth, for β_k being too large, it "should also be" too large in the near future
  - i.e., have to avoid β_{k+1} to grow back to the "bad" β_k too soon
  - How : we set the ceiling parameter

$$\bar{\beta}_{k+1} = \beta_k$$

Variation of the HER algorithm

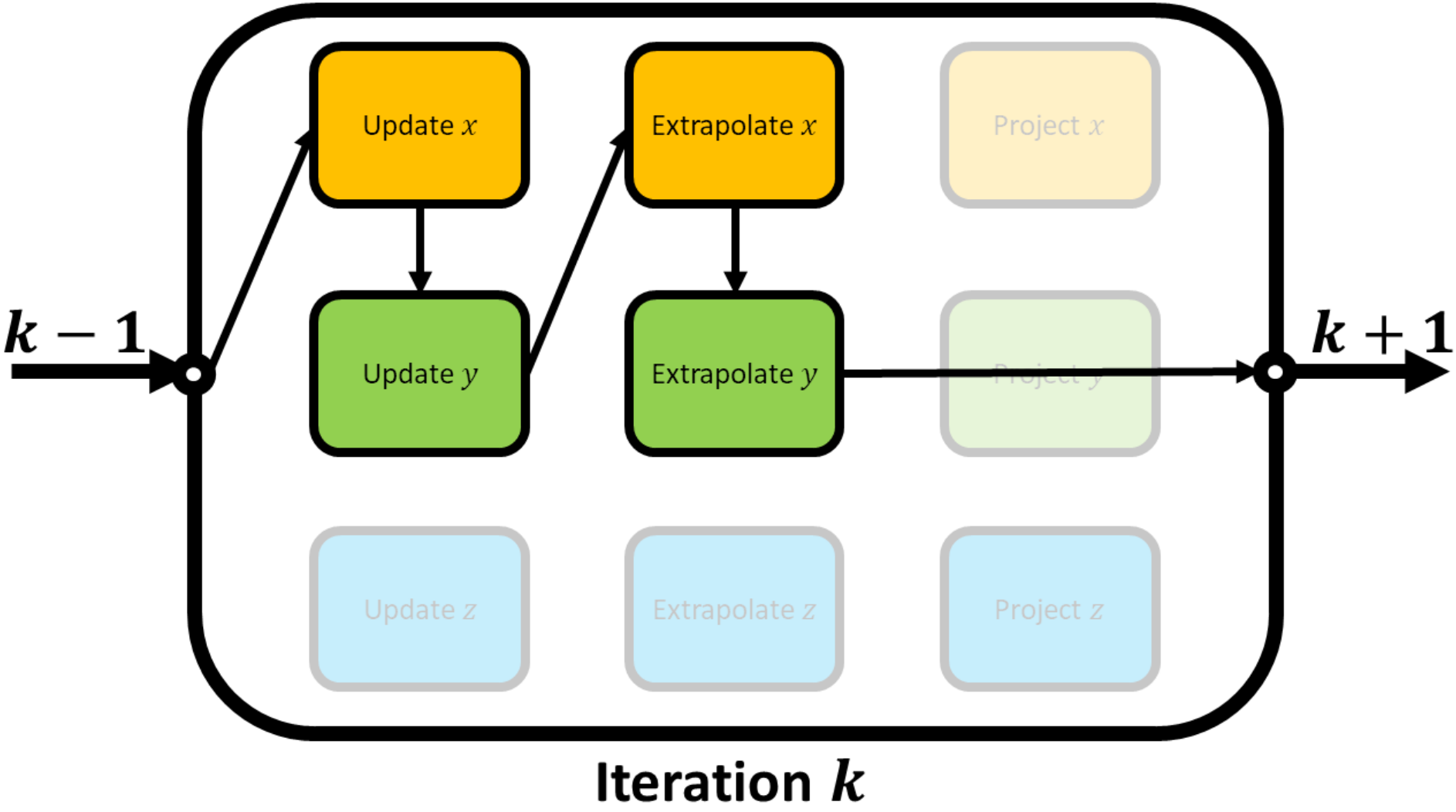
Together with

$$\beta_{k+1} = \begin{cases} \min\{\gamma\beta_k, \bar{\beta}\} & \text{if } \hat{e}^k \leq \hat{e}^{k-1} \\ \dfrac{\beta_k}{\eta} & \text{if } \hat{e}^k > \hat{e}^{k-1} \end{cases}$$

$$\bar{\beta}_{k+1} = \begin{cases} \min\{\bar{\gamma}\bar{\beta}_k, 1\} & \text{if } \hat{e}^k \leq \hat{e}^{k-1} \text{ and } \bar{\beta}_k < 1 \\ \beta_k & \text{if } \hat{e}^k > \hat{e}^{k-1} \end{cases}.$$

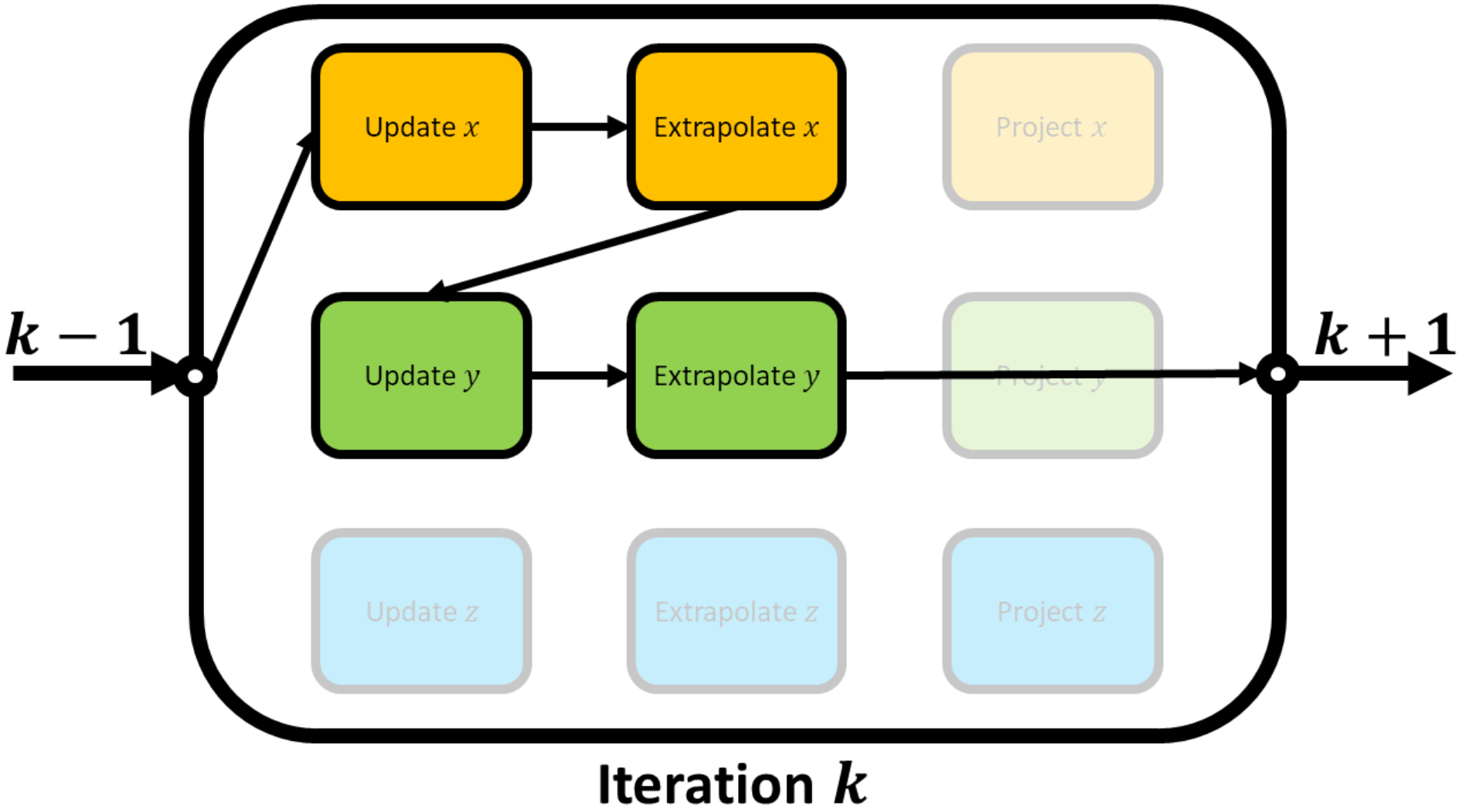
There are variations on the update-extrapolate chain :

- Update $\mathbf{U} \to$ extrapolate $\mathbf{U} \to$ update $\mathbf{V} \to$ extrapolate $\mathbf{V}$
- Update $\mathbf{U} \to$ extrapolate $\mathbf{U} \to$ project $\mathbf{U} \to$ update $\mathbf{V} \to$ extrapolate $\mathbf{V} \to$ project $\mathbf{V}$
- Update $\mathbf{U} \to$ update $\mathbf{V} \to$ extrapolate $\mathbf{U} \to$ extrapolate $\mathbf{V}$
- Update $\mathbf{U} \to$ update $\mathbf{V} \to$ extrapolate $\mathbf{U} \to$ extrapolate $\mathbf{V} \to$ project $\mathbf{U} \to$ project $\mathbf{V}$



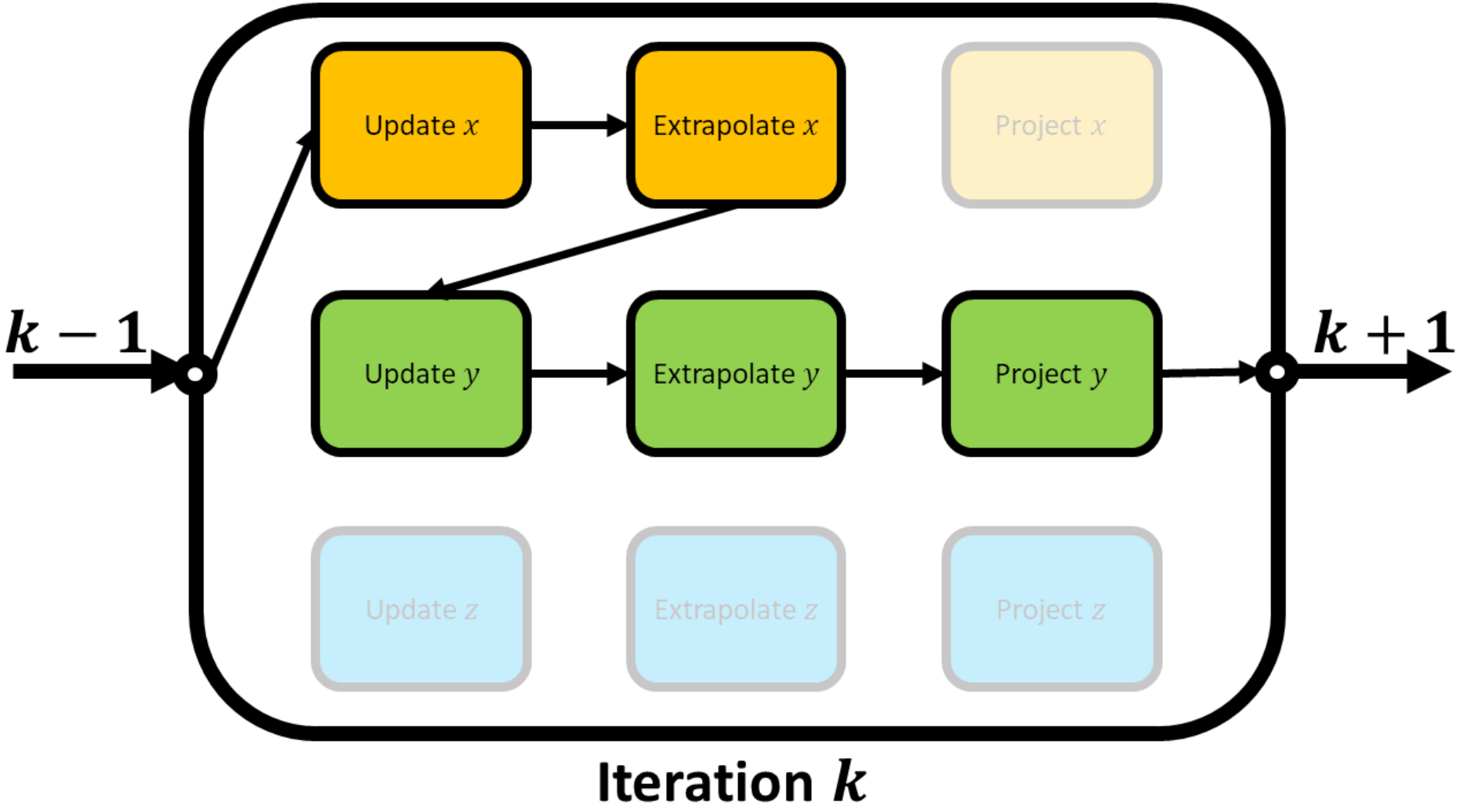
For empirical results, see paper **A**.-Gillis, "Accelerating Non-negative matrix factorization by extrapolation", *Neural Computation*, Feb, 2019. arXiv : 1805.06604

Open question : why certain structure has a better performance than others



For empirical results, see paper **A**.-Gillis, "Accelerating Non-negative matrix factorization by extrapolation", *Neural Computation*, Feb, 2019. arXiv : 1805.06604

Open question : why certain structure has a better performance than others



For empirical results, see paper **A**.-Gillis, "Accelerating Non-negative matrix factorization by extrapolation", *Neural Computation*, Feb, 2019. arXiv : 1805.06604

Open question : why certain structure has a better performance than others

Why $\hat{e}^k = \|\mathbf{M} - \hat{\mathbf{U}}\mathbf{V}\|$ not $e^k = \|\mathbf{M} - \mathbf{U}\mathbf{V}\|$

- $\hat{\mathbf{V}}$ is updated according to $\hat{\mathbf{U}}$
- It gives the algorithm some degrees of freedom to possibly increase the objective function (a vague statement)
- Computationally cheaper (main reason)
  Compute $\|\mathbf{M} - \mathbf{U}\mathbf{V}\|_F$ cost $O(mnr)$ instead of $O(mr^2)$ by re-using previous computed terms :

$$\|\mathbf{M} - \mathbf{U}\mathbf{V}\|_F^2 = \|\mathbf{M}\|_F^2 - 2\left\langle \mathbf{U}, \mathbf{M}\mathbf{V}^\top \right\rangle + \left\langle \mathbf{U}^\top\mathbf{U}, \mathbf{V}\mathbf{V}^\top \right\rangle$$

  Significant if $r \ll n$, which is true in low-rank model.
  Says $r = 5 \sim 50$ with $n = 10^3 \sim 10^6$ or more.
- (If converge) In the long run, $\mathbf{U}$, $\hat{\mathbf{U}}$ is effectively the same :
  $\mathbf{U}^\infty = \hat{\mathbf{U}}^\infty$ after projection

Why $\hat{e}^k = \|\mathbf{M} - \hat{\mathbf{U}}\mathbf{V}\|$ not $e^k = \|\mathbf{M} - \mathbf{U}\mathbf{V}\|$

It gives the algorithm some degrees of freedom to possibly increase the objective function (a vague statement)

By definition of the algorithm, we have

- $\hat{\mathbf{U}}^{k+1} = \mathbf{U}^{k+1} + \beta^{k+1}(\mathbf{U}^{k+1} - \mathbf{U}^k)$
- $\mathbf{V}^{k+1} = \underset{\mathbf{V}\geq 0}{\operatorname{argmin}} f(\hat{\mathbf{U}}^{k+1}, \mathbf{V})$

$$
\begin{aligned}
\underbrace{\|\mathbf{M} - \hat{\mathbf{U}}^{k+1}\mathbf{V}^{k+1}\|}_{\hat{e}^{k+1}} &= \|\mathbf{M} - \mathbf{U}^{k+1}\mathbf{V}^{k+1} + \beta^{k+1}(\mathbf{U}^k - \mathbf{U}^{k+1})\mathbf{V}^{k+1}\| \\
&\overset{\Delta IQ.}{\leq} \underbrace{\|\mathbf{M} - \mathbf{U}^{k+1}\mathbf{V}^{k+1}\|}_{e^{k+1}} + \beta^{k+1}\|(\mathbf{U}^k - \mathbf{U}^{k+1})\mathbf{V}^{k+1}\| \\
\hat{e}^{k+1} &\leq e^{k+1} + \beta^{k+1}\underbrace{\|\mathbf{U}^k - \mathbf{U}^{k+1}\|}_{\searrow 0 \text{ if converge}}\|\mathbf{V}^{k+1}\|
\end{aligned}
$$

A talk in this conference on similar topics

On Accelerated Alternating Minimization

Sergey Guminov
Moscow Institute of Physics and Technology;
Institute for Information Transmission Problems

Pavel Dvurechensky
Weierstrass Institute for Applied Analysis and Stochastics;
Institute for Information Transmission Problems

Alexander Gasnikov
Moscow Institute of Physics and Technology;
Institute for Information Transmission Problems

Abstract

Alternating minimization (AM) optimization algorithms have been known for a long time and are of importance in machine learning problems, among which we are mostly motivated by approximating optimal transport distances. AM algorithms assume that the decision variable is divided into several blocks and minimization in each block can be done explicitly or cheaply with high accuracy. The ubiquitous Sinkhorn's algorithm can be seen as an alternating minimization algorithm for the dual to the entropy-regularized optimal transport problem. We introduce an accelerated alternating minimization method with a $1/k^2$ convergence rate, where k is the iteration counter. This improves over known bound $1/k$ for general AM methods and for the Sinkhorn's algorithm. Moreover, our algorithm converges faster than gradient-type methods in practice as it is free of the choice of the step-size and is adaptive to the local smoothness of the problem. We show that the proposed method is primal-dual, meaning that if we apply it to a dual problem, we can reconstruct the solution of the primal problem with the same convergence rate. We apply our method to the entropy regularized optimal transport problem and show experimentally, that it outperforms Sinkhorn's algorithm.

A talk in this conference on similar topics

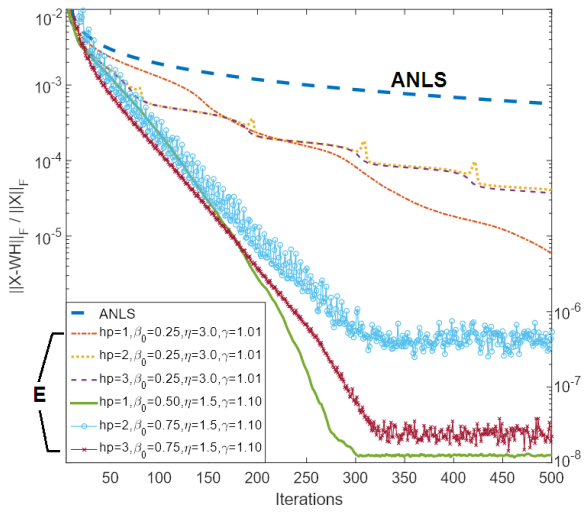
Their work has convergence result, but . . .

- They consider convex problem. NMF/NTF are not.
- Their algo2,3 has is to solve unconstrained smooth minimization problem $\min_x f(x)$. Not useful for NMF/NTF.
- Step9 of Algo2 and Step6 of Algo3 require closed form sol. of sub-minimization $\min f(x) : x \in S$
  Not useful for NMF/NTF : NNLS no close form sol.
- (Statistical) Fact : minimize a (1st/2nd-order) majorization fun. is often easier to have closed form sol. (that's why in their test they pick linear objective fun)
  Not useful for quadratic NMF/NTF; also sub-min. problem is NNLS
- The way they update parameters $\sim$ do line search using info of objective fun. $\sim$ comp. cost as HER

These some how tells why convergence analysis of general extrapolated BCD on non-cvx problems (with no close form sol. in sub-minimization) are hard, not to mention restart is involved.

Fancy graphs showing numerics

NMF literature use $(\mathbf{X}, \mathbf{W}, \mathbf{H})$ instead of $(\mathbf{M}, \mathbf{U}, \mathbf{V})$
Here the plots are using e (not $\hat{e}$)



Low-rank synthetic data

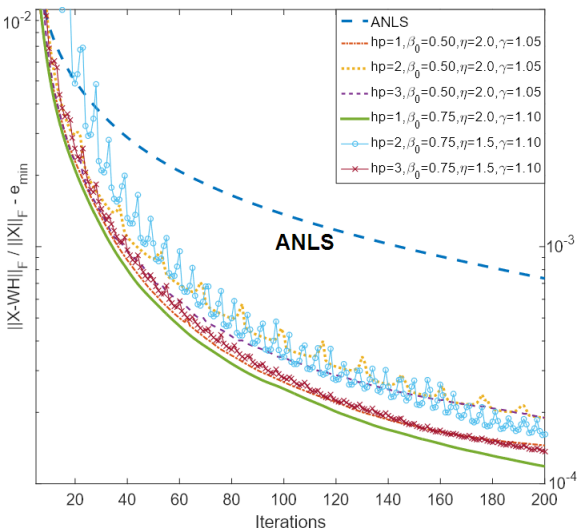


Image data

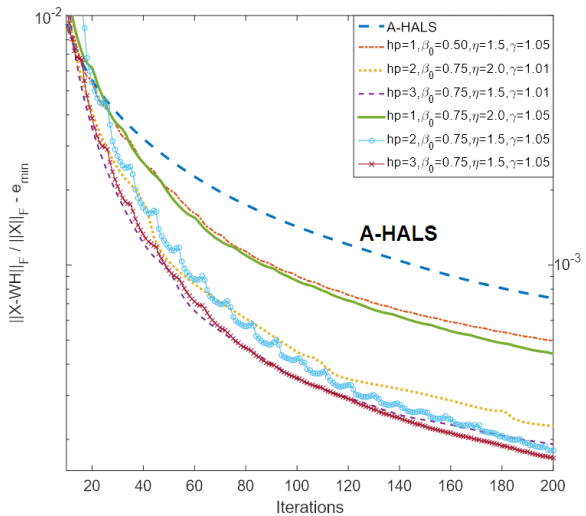
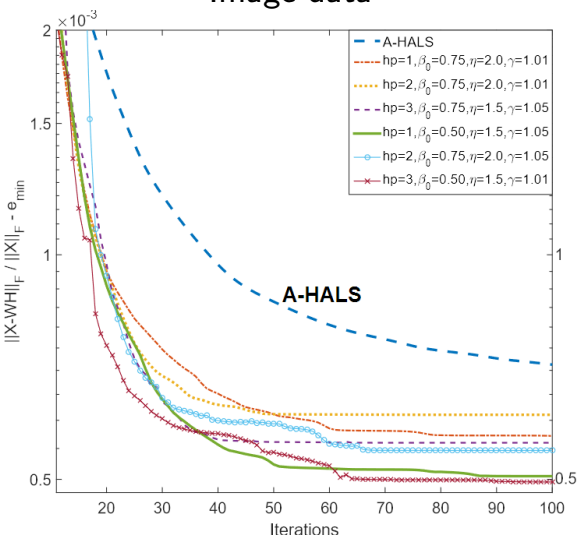
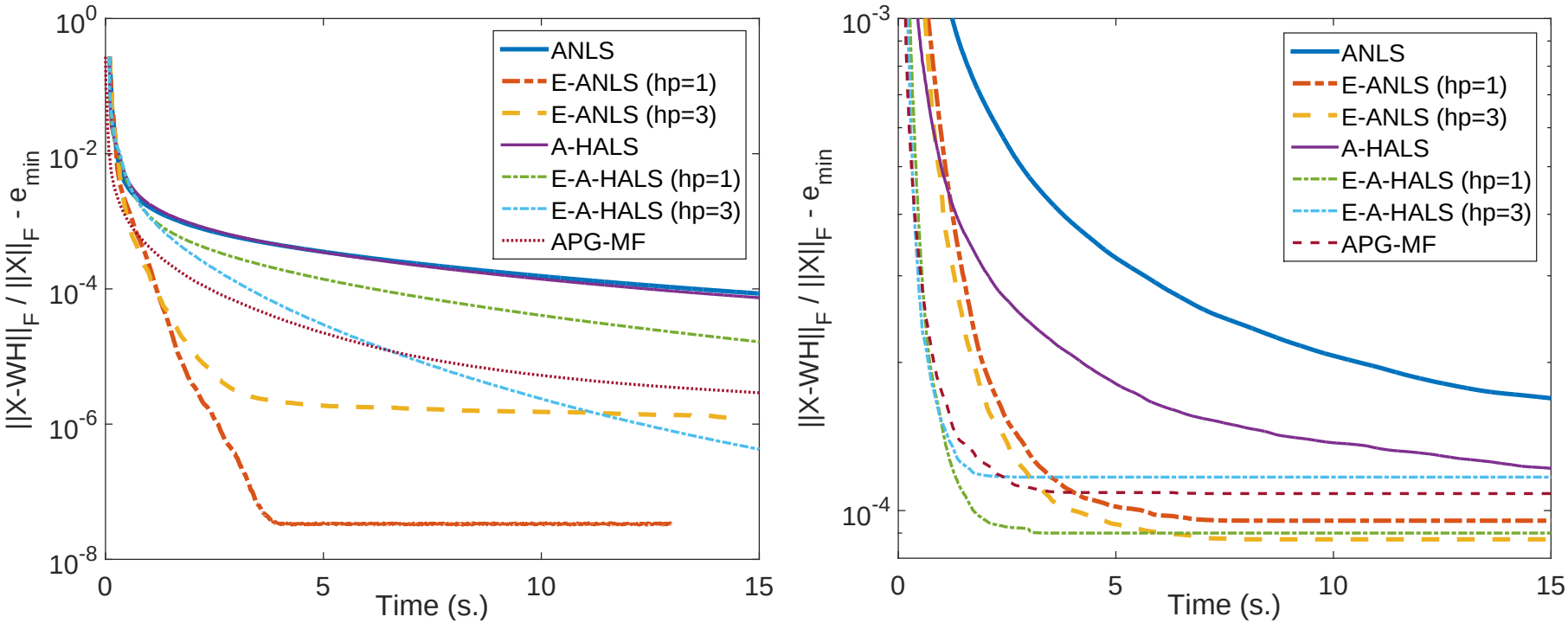


Image data



Text data

HER (the "E-") beats the APG-MF of (Xu-Yin, 2013) A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. SIAM J. Img Sci.



Similar results in sparse text data, dense image data : ORL, Umist, CBCL, Frey.

Details in paper : A.-Gillis, "Accelerating Non-negative matrix factorization by extrapolation", *Neural Computation*, Feb, 2019. (arXiv : 1805.06604)

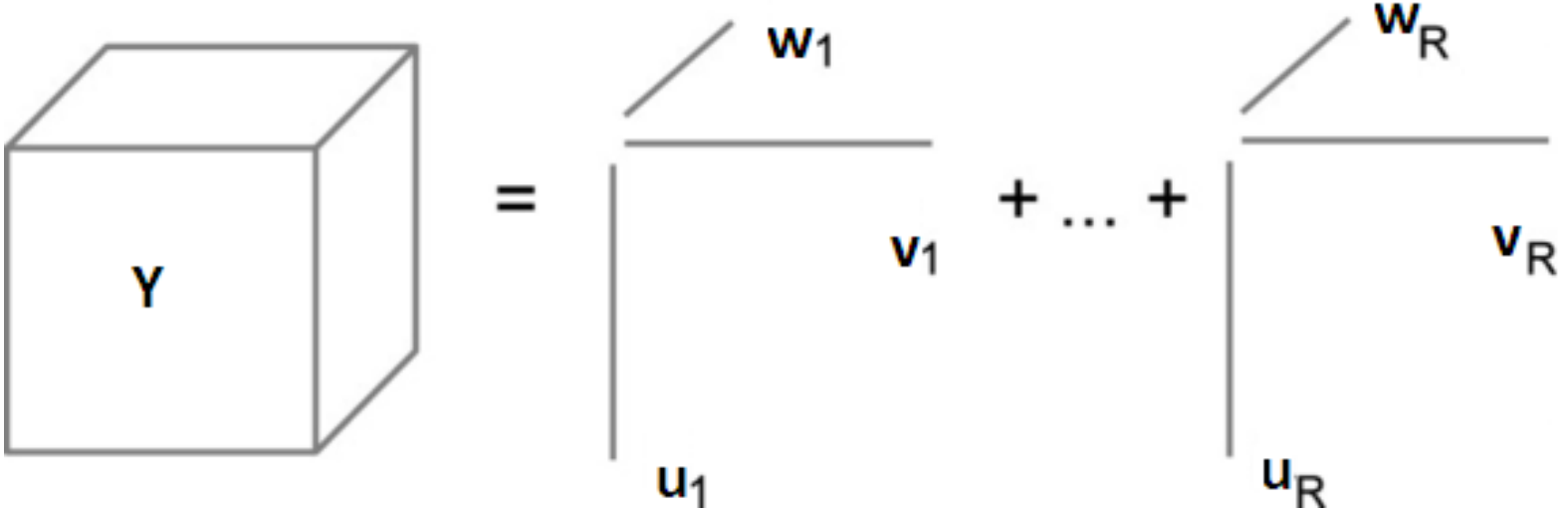
Outline

Non-negative Canonical Polyadic Decomposition

(Joint-work with Jeremy E. Cohen of IRISA, Rennes, France)
A.-Cohen-Gillis, "Accelerating Approximate Nonnegative Canonical Polyadic Decomposition using Extrapolation", 2019.

For example, order-3 tensor :

$$\begin{aligned} \underset{\mathbf{U}\geq 0,\mathbf{V}\geq 0,\mathbf{W}\geq 0}{\operatorname{argmin}} \quad f(\mathbf{U},\mathbf{V},\mathbf{W}) &= \|\mathcal{Y} - \mathbf{U} * \mathbf{V} * \mathbf{W}\| \\ &= \|\mathcal{Y} - \sum_{i}^{r} \mathbf{u}_i * \mathbf{v}_i * \mathbf{w}_i\| \end{aligned}$$



(Not on Tucker Model in this talk.)

$$(\mathcal{P}) \ : \min_{\mathbf{U}\geq 0, \mathbf{V}\geq 0, \mathbf{W}\geq 0} f(\mathbf{U},\mathbf{V},\mathbf{W}) = \|\mathcal{Y} - \mathbf{U} * \mathbf{V} * \mathbf{W}\|^2$$

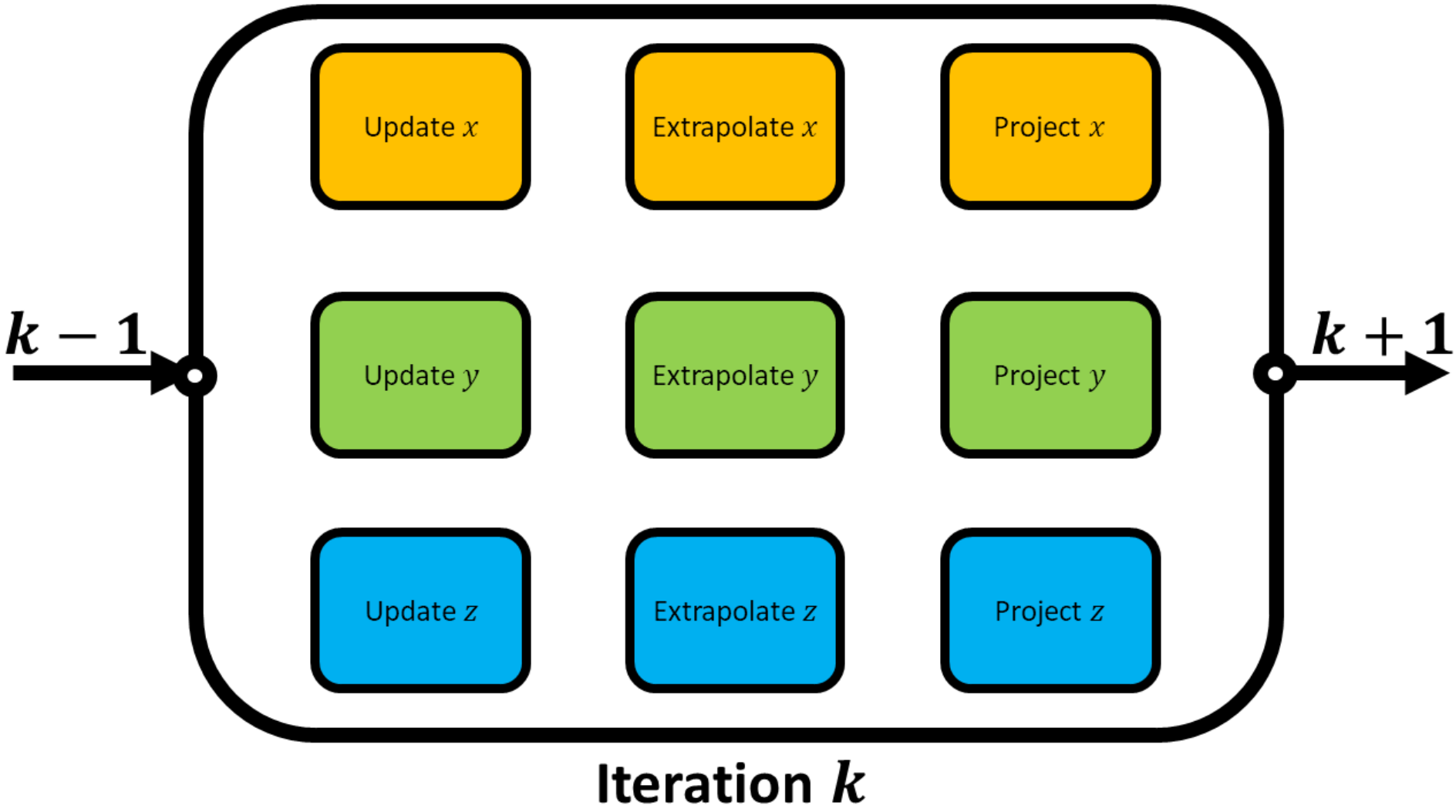
Algorithm HER

Input: $\mathcal{Y} \in \mathbb{R}_+^{I\times J\times K}$, r, $\mathbf{U}, \mathbf{V}, \mathbf{W}$, $\hat{\mathbf{U}} = \mathbf{U}, \hat{\mathbf{V}} = \mathbf{V}, \hat{\mathbf{W}} = \mathbf{W}$
Output: $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$

1: **for** $k = 1, 2, \ldots$ **do**
2: **for** $\mathbf{U}, \mathbf{V}, \mathbf{W}$ **do**
3: $\mathbf{U}^{k+1} = \underset{\mathbf{U}\geq 0}{\operatorname{argmin}} f(\mathbf{U}, \hat{\mathbf{V}}^k, \hat{\mathbf{W}}^k)$
4: $\hat{\mathbf{U}}^{k+1} = \mathbf{U}^{k+1} + \beta_k(\mathbf{U}^{k+1} - \mathbf{U}^k)$.
5: **end for**
6: $\hat{e}^{k+1} = f(\hat{\mathbf{U}}^{k+1}, \hat{\mathbf{V}}^{k+1}, \mathbf{W}^{k+1})$
7: Update $\beta_k, \bar{\beta}_k$ and restarts (if needed)
8: **end for**

- $\hat{e}^k$ is implicitly computed by reusing already compute component : $O(mnr) \to O(mr^2)$ with $m = K$, $n = IJ \ggg r$ (insane!)
- 3 MTTKRP (Matricized tensor times Khatri-Rao product) if using 1st order solver
- Many variation. e.g. project after extrapolation

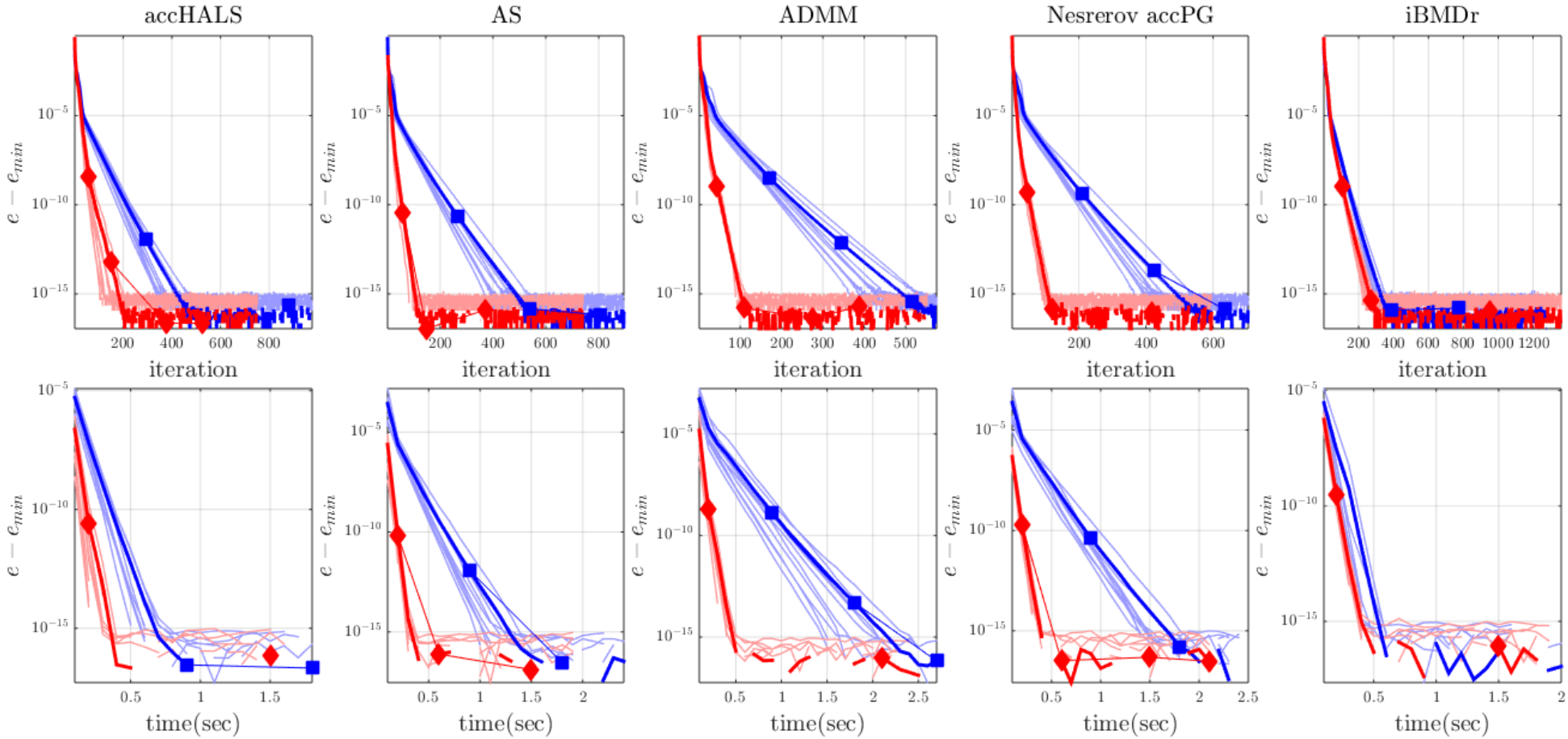
Unsolved problem : NNCPD has even higher variability on the chain structure.



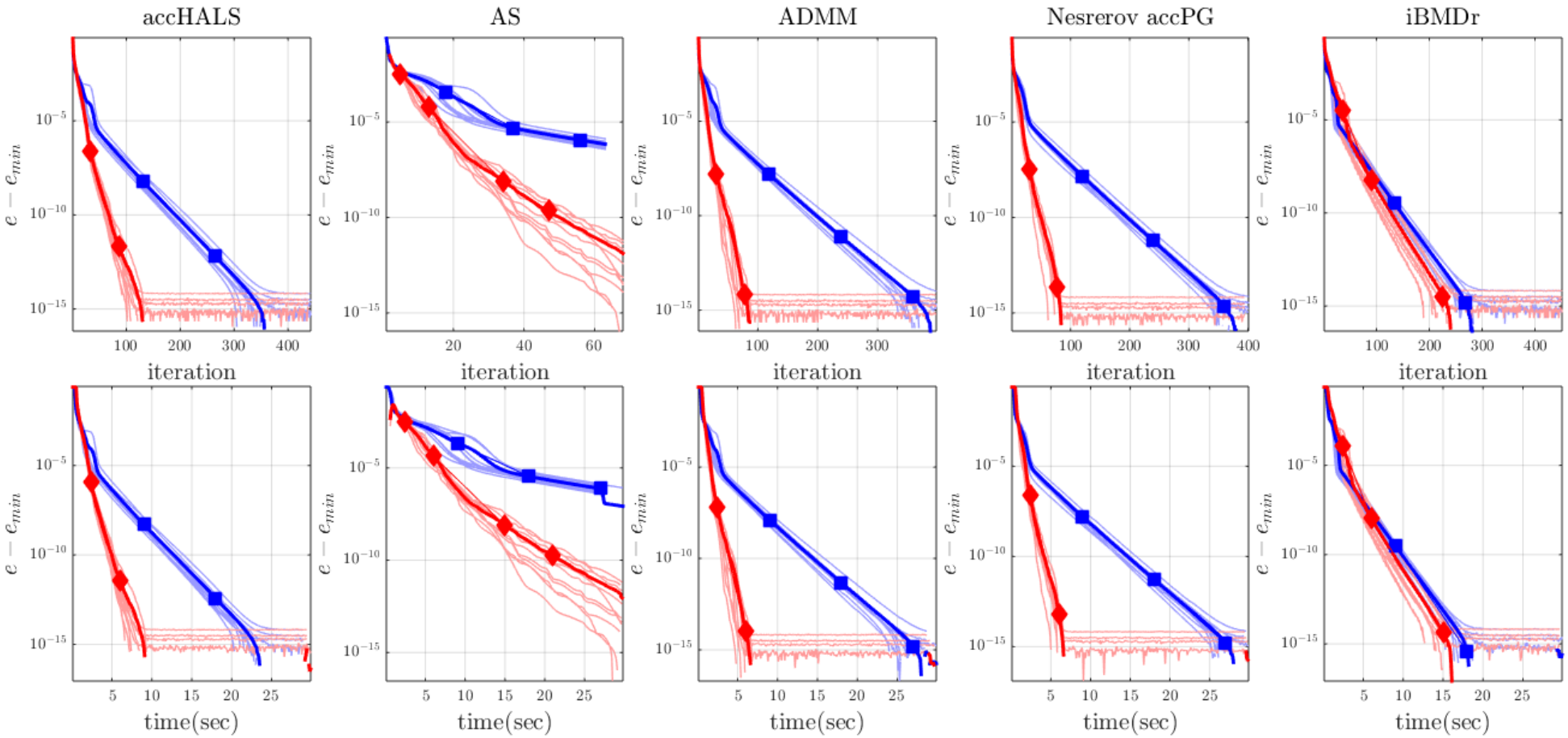
Understanding the relationship between the data structure (rank size, size of each mode) and the chain structure will be crucial.

More fancy graphs showing numerics

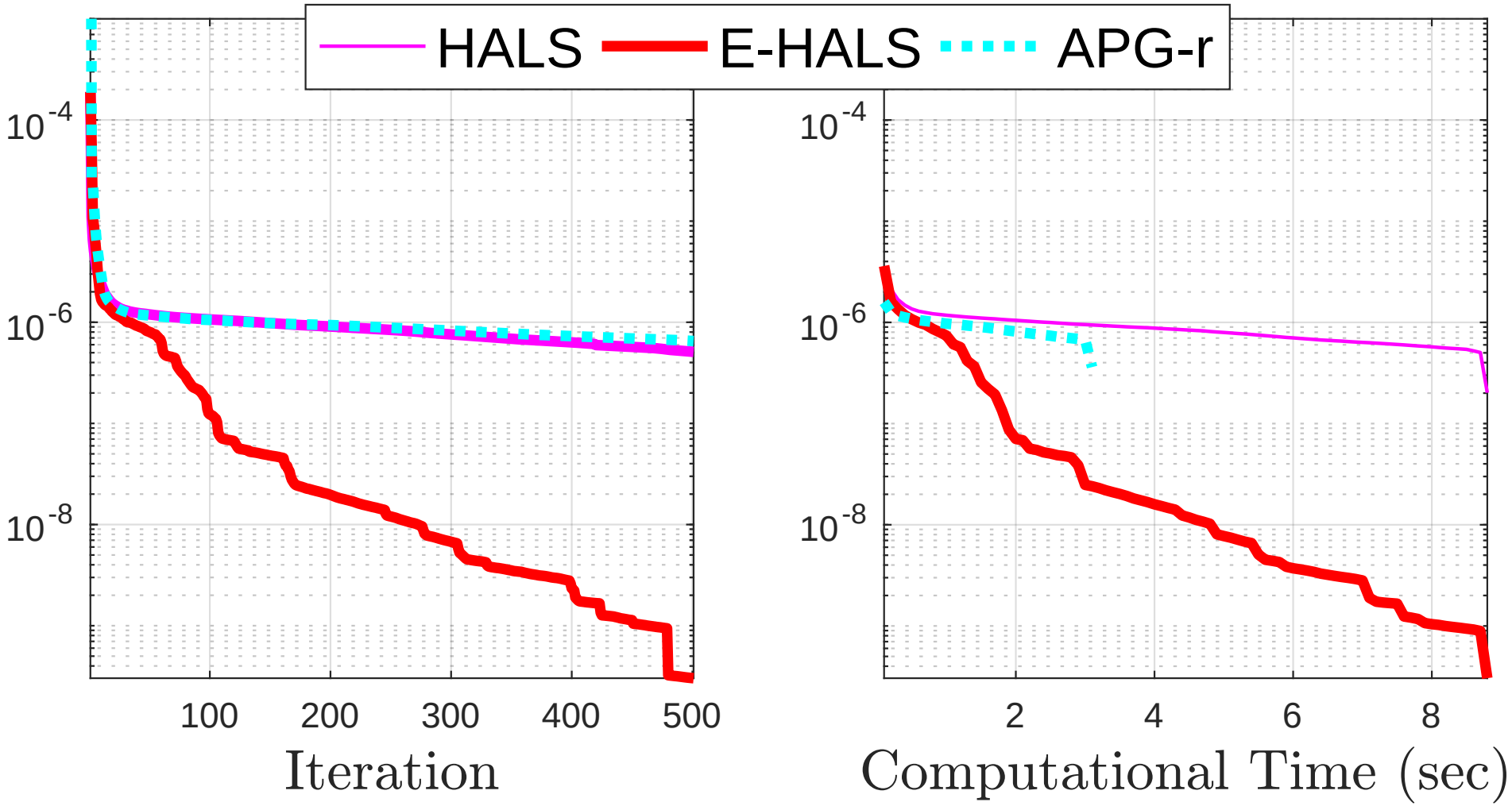
[I,J,K,r,σ] = [50,50,50,10,0.0] on algorithms with HER(red), sans HER(blue)



[I,J,K,r,σ] = [150,1000,100,10,0.0] on algorithms with HER(red), sans HER(blue)

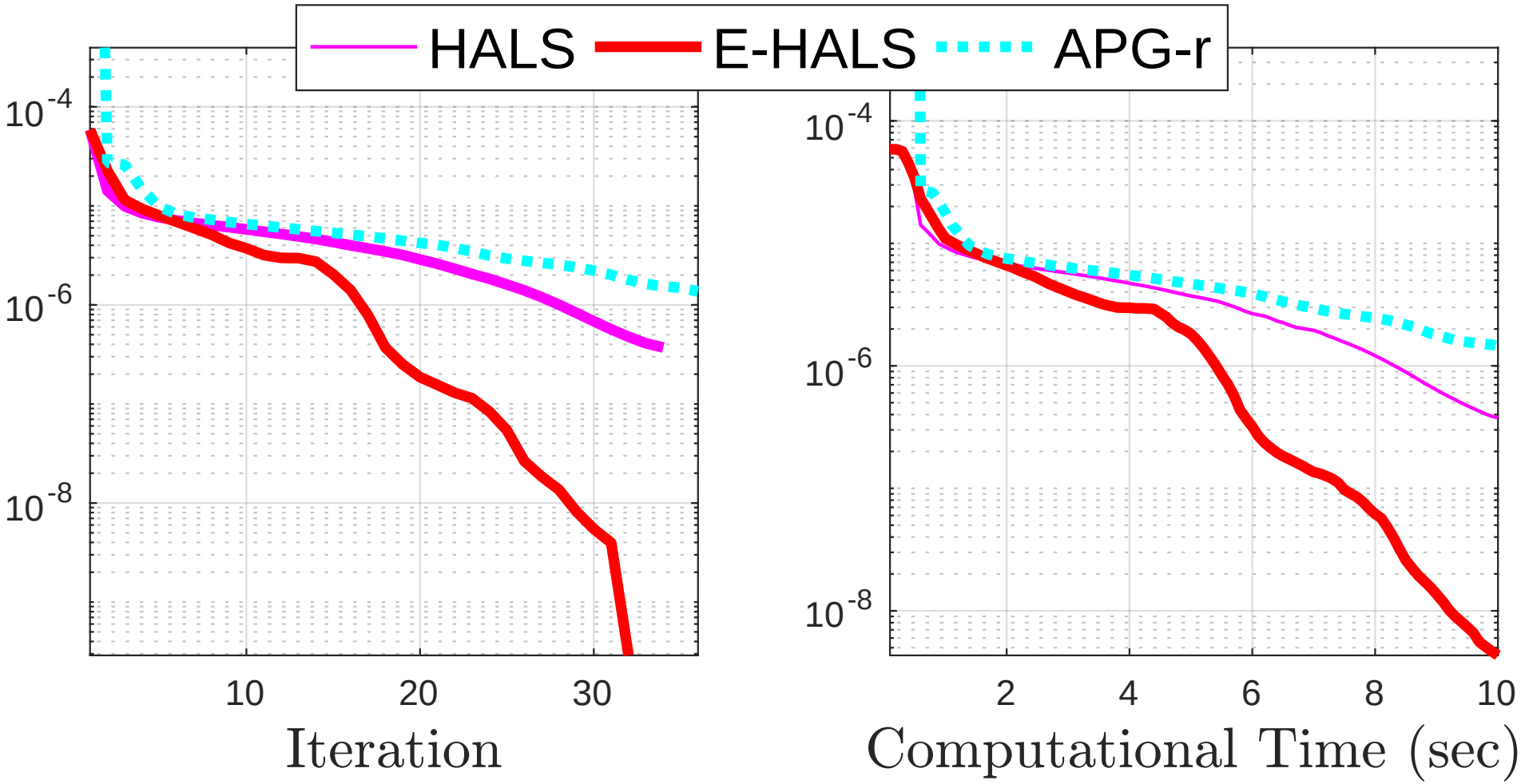


On low-rank, cubic size, ill-condition data



A.-Cohen-Gillis, "Accelerating Approximate Nonnegative Canonical Polyadic Decomposition using Extrapolation", 2019.

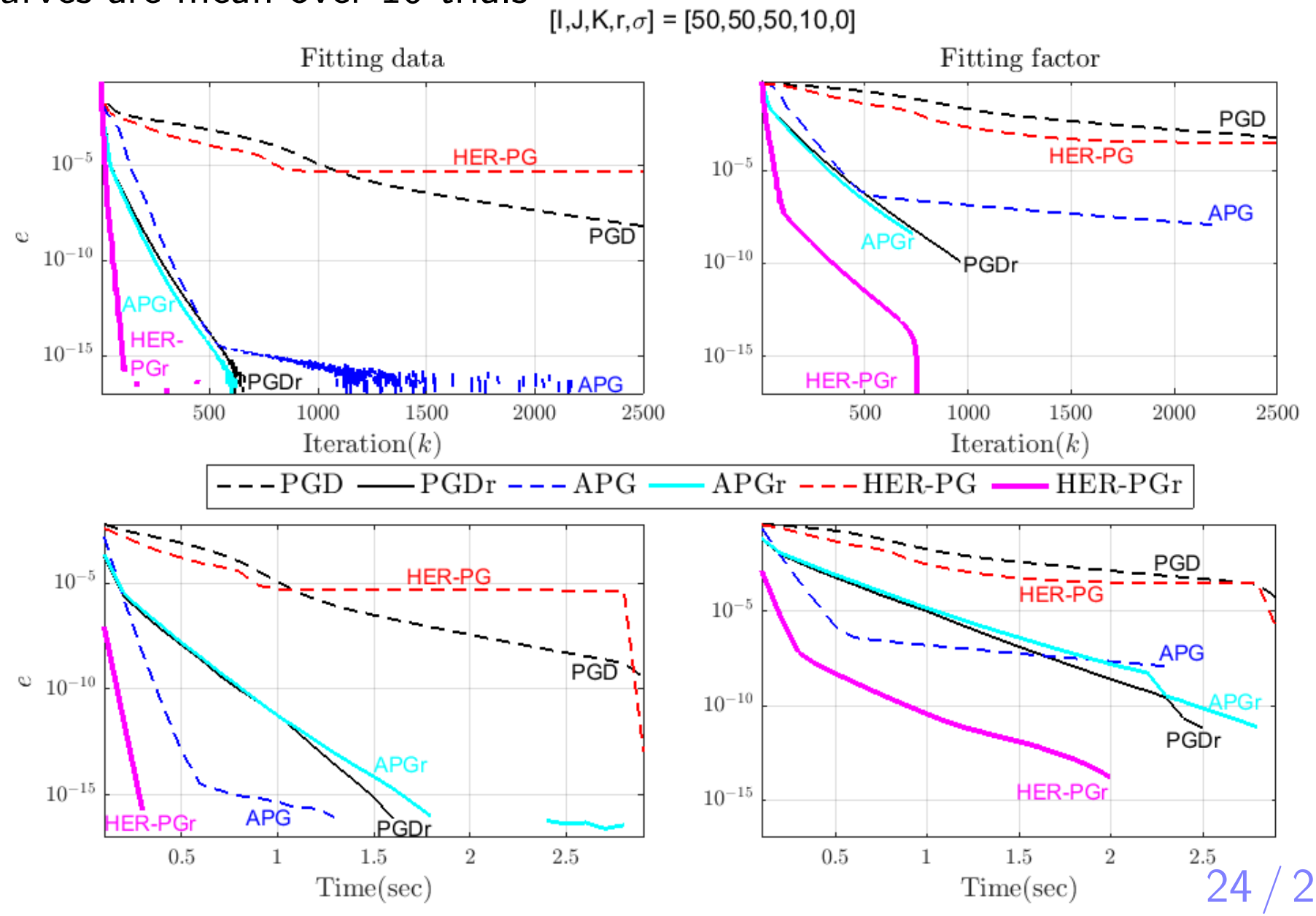
Medium rank, unbalanced sizes (short-fat-thin) data



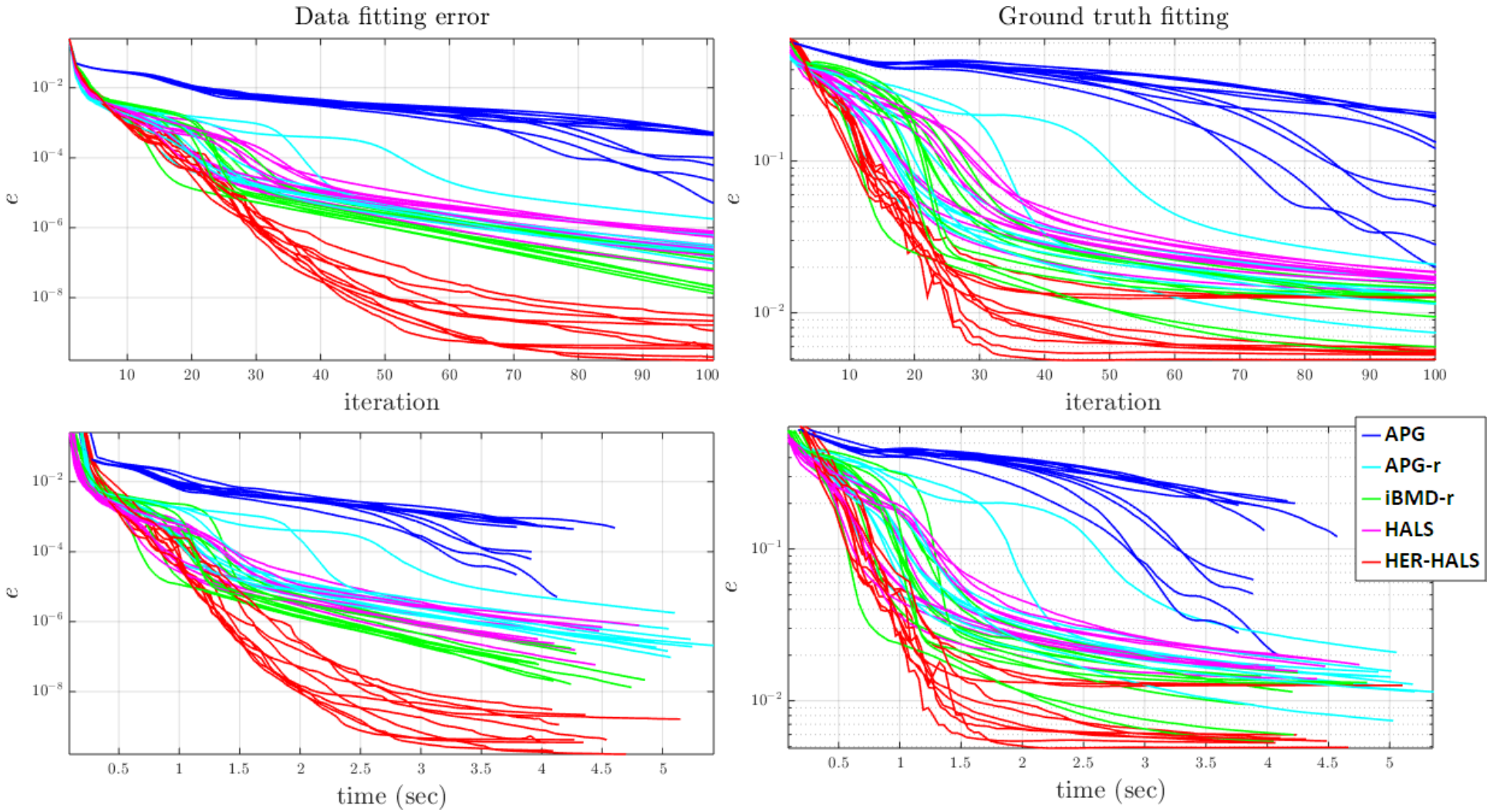
A.-Cohen-Gillis, "Accelerating Approximate Nonnegative Canonical Polyadic Decomposition using Extrapolation", 2019.

On inexact BCD : gradient update

Curves are mean over 10 trials



Comparing with other different inexact BCD using different extrapolations on PG.

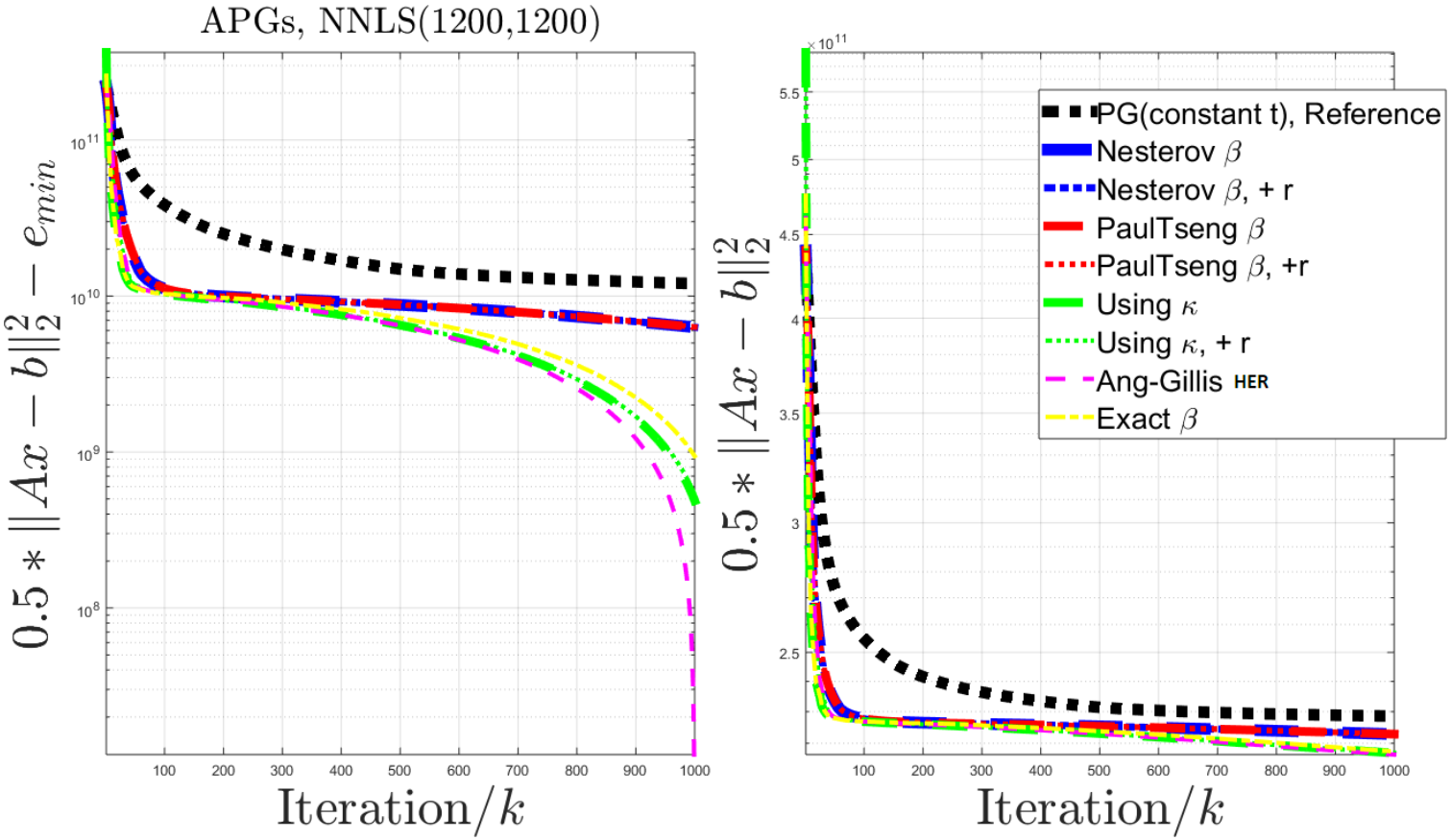


APG : (Xu-Yin 2013) as before
iBMD : L. T. K. Hien, N. Gillis, P. Patrinos, "Inertial Block Mirror Descent Method for Non-Convex Non-Smooth Optimization", March 2019.

Outline

Non-negative Least Square

$$\mathbf{x} = \underset{\mathbf{x}\geq\mathbf{0}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$$



We suspect HER-PG (inexact BCD) just share the same rate as other extrapolated gradients, but again no proof (even NNLS is convex). For HER-exact BCD, even harder.

Various applications

- Title : Accelerating Nonnegative-X by extrapolation, X $\in \mathcal{P} :=$ {LS, MF, TF}
- Actually (empirically) you can enlarge $\mathcal{P}$ to include Regularized-MF, Regularized-TF, MC, TC, DL, ...

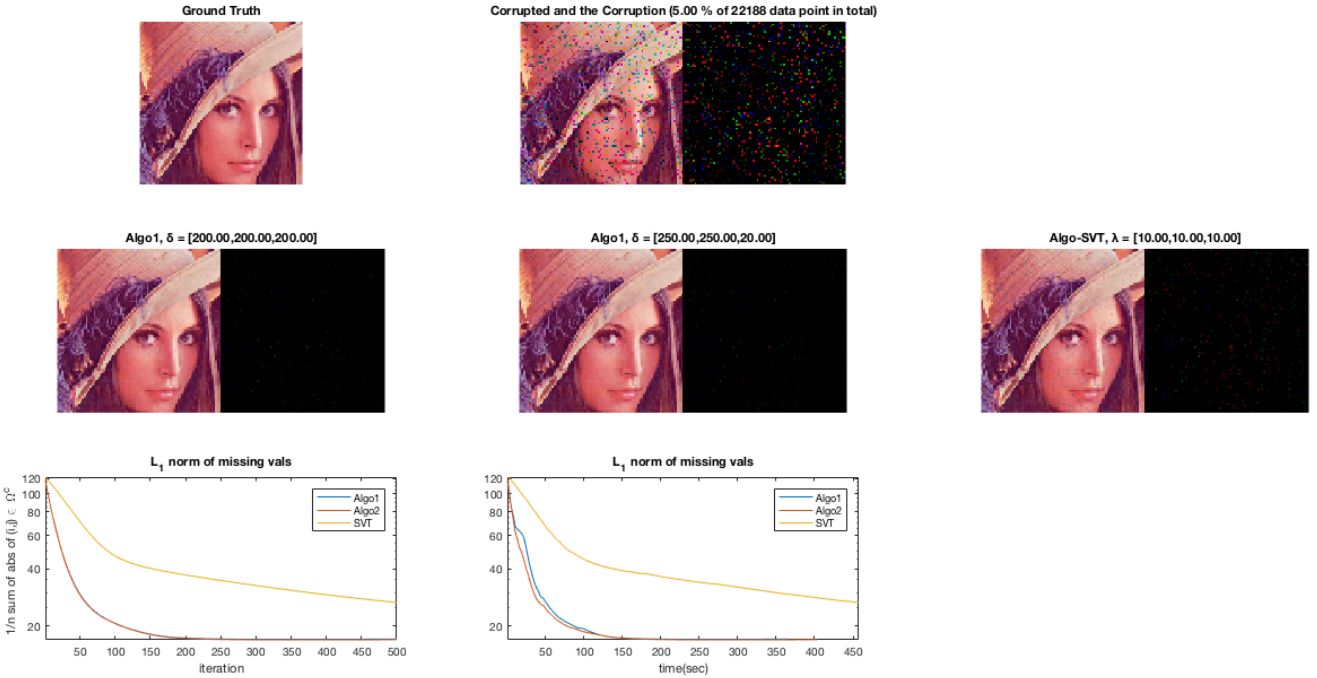


Figure: A toy example on tensor completion. A 9-times speed up (0.11-fraction of time) on the nuclear norm SVT algo.

- Why : HER is highly flexibility – there always exists a suitable parameter for the problem (a hypothesis hard to prove theoretically

Summary : HER

Heuristic Extrapolation with "Restarts" for exact / in-exact BCD on NMF, NTF and NNLS.

Paper :

- NMF paper **A**.-Gillis, "Accelerating Non-negative matrix factorization by extrapolation", *Neural Computation*, Feb, 2019.
- NTF paper **A**.-Cohen-Gillis, "Accelerating Approximate Nonnegative Canonical Polyadic Decomposition using Extrapolation", 2019.

  A longer version on accelerating different algos is working in progress.

Not discussed

- Accelerating other X
- Applications

Open problems

- Convergence theory (at least for the convex NNLS)
- The chain structure variation

Slide, code, preprint in *angms.science*

Workshop on Low-Rank Models and Applications (LRMA)

- Mons, Belgium, September 12-13, 2019
- Topics : Low-rank model $\in \{$computer science, information theory, mathematics and signal processing$\}$
- Plenary speakers
  - Cedric Fevotte (CNRS, IRIT Toulouse)
  - Valeria Simoncini (U. Bologna)
  - Nicola Guglielmi (U. L'Aquila)
  - Vincent Tan (NUS)
  - Zhihui Zhu (Johns Hopkins U.)
  - Christian Grussler (Cambridge U.)
  - Andre Uschmajew (Max Planck Institute)
  - Stephen Vavasis (U. Waterloo)
- Program now available. Don't forget to register. Registration is free.

https://sites.google.com/site/lowrankmodels/