Learning with Low Rank Approximations

Jeremy E. Cohen

Team Panama, IRISA, CNRS

Sisyphe Seminar, 06 Mai 2021

Roadmap

1. An introduction to tensor methods
2. Nonnegative Tucker decomposition of music for automatic segmentation
3. Heuristic extrapolation of alternating algorithms for nonnegative tensor decomposition

Separability: a fundamental property

Definition: Separability

Let $f : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. Map f is said to be separable if there exist real maps f_1, f_2, f_3 so that

$$f(x, y, z) = f_1(x) f_2(y) f_3(z)$$

Of course, any order (i.e. number of variables) is fine.

<u>Examples</u>:

$$(xyz)^n = x^n y^n z^n \ , \ \ e^{x+y} = e^x e^y, \quad \int_x \int_y h(x) g(y) dx dy = \left(\int_x h(x) dx\right) \left(\int_y g(y) dy\right)$$

Some usual function are not separable, but are written as a few separable ones!

- $\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b)$
- $\log(xy) = \log(x)\mathbb{1}_{y \in \mathbb{R}} + \mathbb{1}_{x \in \mathbb{R}} \log(y)$

Separability and matrix rank

Now what about discrete spaces? $(x, y, z) \rightarrow \{(x_i, y_j, z_k)\}_{i\in I, j\in J, k\in K}$
$\rightarrow$ Values of f are contained in a tensor $\mathcal{T}_{ijk} = f(x_i, y_j, z_k)$.

Example: e^{x_i} is a vector of size I. Let us set $x_i = i$ for $i \in \{0, 1, 2, 3\}$.

$$\begin{bmatrix} e^0 \\ e^1 \\ e^2 \\ e^3 \end{bmatrix} = \begin{bmatrix} e^0 e^0 \\ e^0 e^1 \\ e^2 e^0 \\ e^2 e^1 \end{bmatrix} = \begin{bmatrix} e^0 \\ e^2 \end{bmatrix} \otimes_K \begin{bmatrix} e^0 \\ e^1 \end{bmatrix}$$

Here, this means that a matricized vector of exponential is a rank one matrix.

$$\begin{bmatrix} e^0 & e^1 \\ e^2 & e^3 \end{bmatrix} = \begin{bmatrix} e^0 \\ e^2 \end{bmatrix} \begin{bmatrix} e^0 & e^1 \end{bmatrix} = \begin{bmatrix} e^0 \\ e^2 \end{bmatrix} \otimes \begin{bmatrix} e^0 \\ e^1 \end{bmatrix}$$

Setting $i = j2^1 + k2^0$, $f(j, k) = e^{2j+k}$ is separable in (j, k).

Conclusion: A rank-one matrix can be seen as a separable function on a grid.

Tensor rank?

We can also introduce a third-order tensor here:

$$\begin{bmatrix} e^0 \\ e^1 \\ e^2 \\ e^3 \\ e^4 \\ e^5 \\ e^6 \\ e^7 \end{bmatrix} = \begin{bmatrix} e^0e^0e^0 \\ e^0e^0e^1 \\ e^0e^2e^0 \\ e^0e^2e^1 \\ e^4e^0e^0 \\ e^4e^0e^1 \\ e^4e^2e^0 \\ e^4e^2e^1 \end{bmatrix} = \begin{bmatrix} e^0 \\ e^4 \end{bmatrix} \otimes_K \begin{bmatrix} e^0 \\ e^2 \end{bmatrix} \otimes_K \begin{bmatrix} e^0 \\ e^1 \end{bmatrix}$$

By "analogy" with matrices, we say that a tensor is rank-one if it is the discretization of a separable function.

From separability to matrix/tensor rank

From now on, we identify a function $f(x_i, y_j, z_k)$ with a three-way array $\mathcal{T}_{ijk}$.

Definition: rank-one tensor

A tensor $\mathcal{T}_{ijk} \in \mathbb{R}^{I \times J \times K}$ is said to be a [decomposable] [separable] [simple] [rank-one] tensor iff there exist $a \in \mathbb{R}^I, b \in \mathbb{R}^J, c \in \mathbb{R}^K$ so that

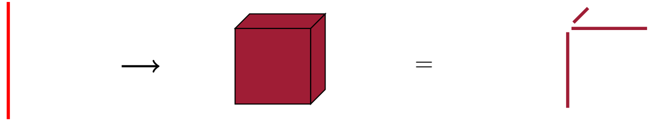
$$\mathcal{T}_{ijk} = a_i b_j c_k$$

or equivalently,

$$\mathcal{T} = a \otimes b \otimes c$$

where $\otimes$ is a multiway equivalent of the exterior product $a \otimes b = ab^t$.

What matters in practice may be to find the right description of the inputs!!



$$f(x, y, z, t, \ldots) \qquad \mathcal{T} \qquad = \qquad a \otimes b \otimes c$$

ALL tensor decomposition models are based on separability

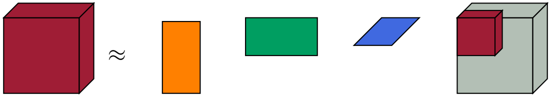
Canonycal Polyadic Decomposition:

$$\mathcal{T} = \sum_{q=1}^{r} a_q \otimes b_q \otimes c_q$$

$$\mathcal{T} = a_1 \otimes b_1 \otimes c_1 + \cdots + a_r \otimes b_r \otimes c_r$$

Tucker Decomposition:

$$\mathcal{T} = \sum_{q_1,q_2,q_3=1}^{r_1,r_2,r_3} g_{q_1 q_2 q_3} a_{q_1} \otimes b_{q_2} \otimes c_{q_3}$$



$$\mathcal{T} \approx (A \otimes B \otimes C)\ \mathcal{G}$$

Definition: tensor [CP] rank (also applies for other decompositions)

$$rank(\mathcal{T}) = \min\{r \mid \mathcal{T} = \sum_{q=1}^{r} a_q \otimes b_q \otimes c_q\}$$

Tensor CP rank coincides with matrix “usual” rank! (on virtual board)

If I were in the audience, I would be wondering:

- Why should I care??
  → I will tell you now.
- Even if I cared, I have no idea how to know if my data is somehow separable or a low-rank tensor!
  → I don't know, this is the difficult part but at least you may think about separability in the future.
  → It will probably not be low rank, but it may be approximately low rank!

Making use of low-rank representations

Let $A = [a_1, a_2, \dots, a_r]$, B and C similarly built.

Uniqueness of the CPD

Under mild conditions

$$krank(A) + krank(B) + krank(C) - 2 \geq 2r, \tag{1}$$

the CPD of $\mathcal{T}$ is essentially unique (i.e.) the rank-one terms are unique.

This means we can interpret the rank-one terms a_q, b_q, c_q
$\rightarrow$ Source Separation!

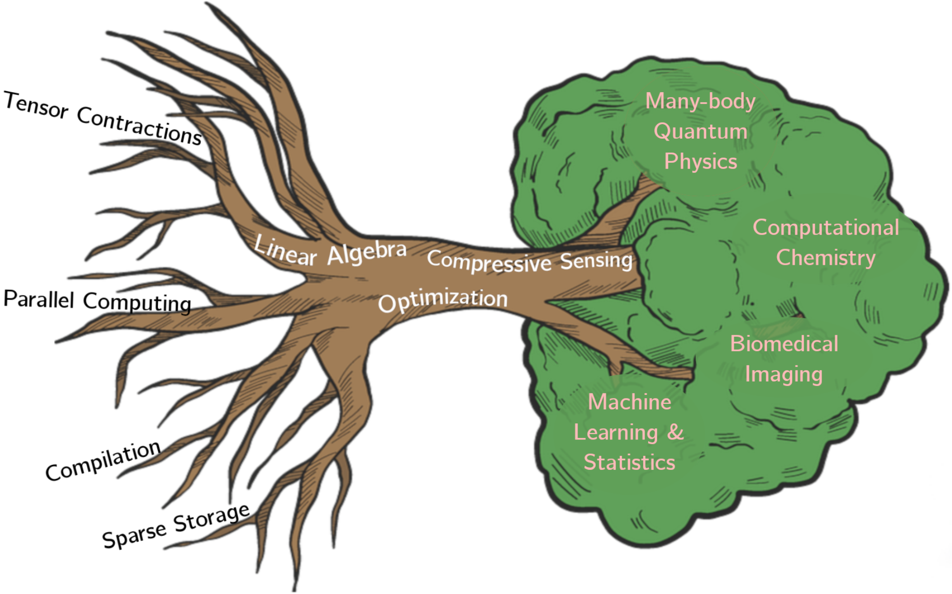
Compression (also true for other models)

The CPD involves $r(I + J + K - 2)$ parameters, while $\mathcal{T}$ contains IJK entries.

If the rank is small, this means huge compression/dimentionality reduction!

- missing values completion, denoising
- function approximation
- imposing sparse structure to solve other problems (PDE, neural networks, dictionary learning...)

The landscape of research on tensors



My ongoing research projects

LoRAiA (ANR JCJC)

Semi-supervision and Tensors:

- Dictionaries/sparse coding
- Optimal Transport

with efficient implementations/algorithms!

Automatic Transcription

With semi-supervision and NMF.

Tensoptly (Inria)

Tensorly optimization layer:

- Constrained models
- Faster algorithms
- Customization

Music Segmentation

PhD of Axel Marmoret.

Sparse/Fast Optimization

Long-term collaboration with N. Gillis (UMONS).

Multimodality

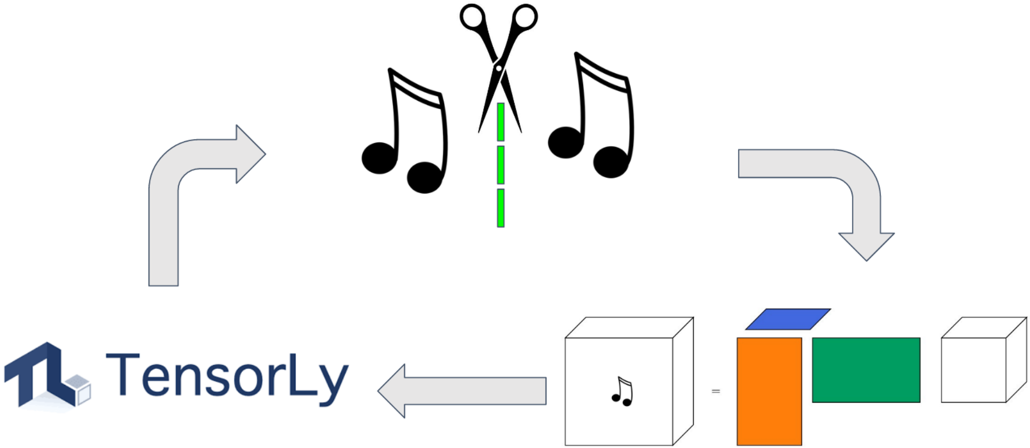
Long-term collaboration with E. Acar (SimulaMet).

A common trait: nonnegativity!

Roadmap

1. An introduction to tensor methods
2. Nonnegative Tucker decomposition of music for automatic segmentation
3. Heuristic extrapolation of alternating algorithms for nonnegative tensor decomposition

The NTD project in a glance

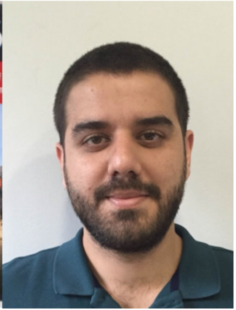


A team effort

Axel Marmoret
Doctorant UR1

Nancy Bertin
CR CNRS

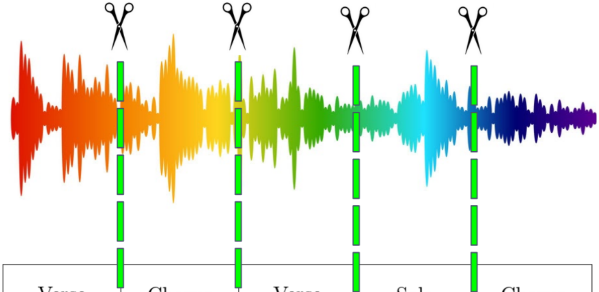
Frederic Bimbot
DR CNRS



Caglayan Tuna
Ingénieur Inria

Axel Marmoret, Jérémy Cohen, Nancy Bertin, Frédéric Bimbot. Uncovering Audio Patterns in Music with Nonnegative Tucker Decomposition for Structural Segmentation. ISMIR 2020 - 21st International Society for Music Information Retrieval, Oct 2020, Montréal (Online), Canada. pp.1-7

Segmenting a song?

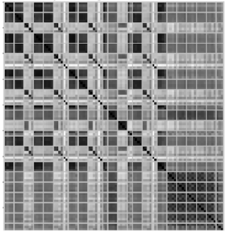


Organisation of the song:	Verse	Chorus	Verse	Solo	Chorus

Large scale structure:	A	B	A	C	B’

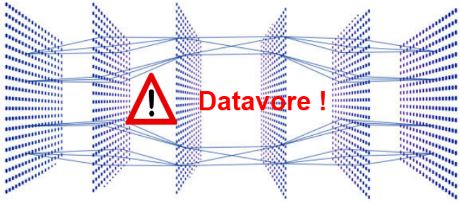
Small scale structure:	a	b	c	c	a	b	d	e	f	c	c’

Unsupervised



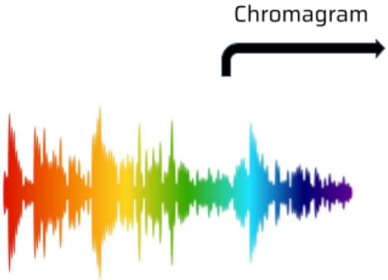
Signal Autosimilarity + post-processing

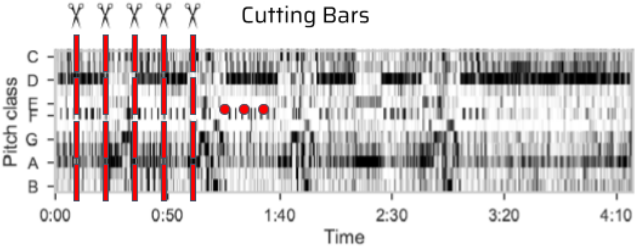
Supervised



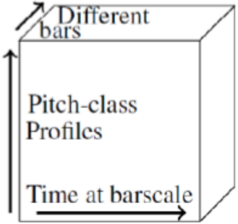
Deep learning

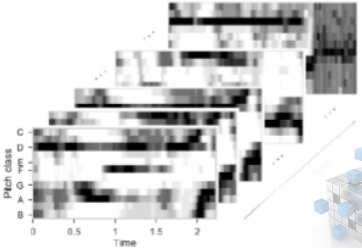
Our idea: a chromagram tensor...





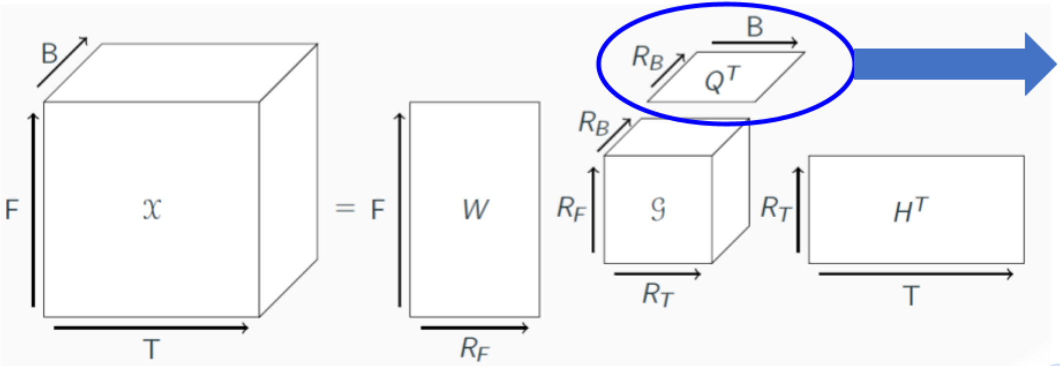
Chromagram of "Come Together", by The Beatles.





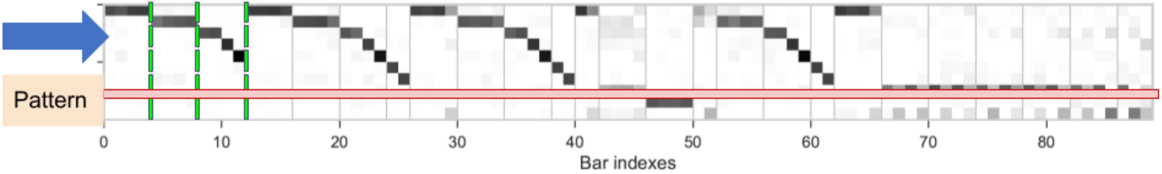
...decomposed to find redundancies!

Approximate Nonnegative Tucker Decomposition $\mathcal{X} \approx (W \otimes H \otimes Q)\mathcal{G}$

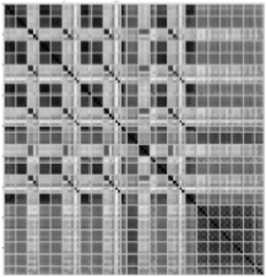


$\mathcal{X} \in \mathbb{R}_+^{F \times T \times B}, W \in \mathbb{R}_+^{F \times R_F}, H \in \mathbb{R}_+^{T \times R_T}, Q \in \mathbb{R}_+^{B \times R_B}, \mathcal{G} \in \mathbb{R}_+^{R_F \times R_T \times R_B}$

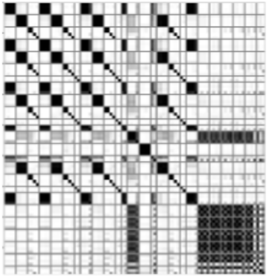
Back to segmentation



Signal Autosimilarity



Patterns autosimilarity



State-of-the-art unsupervised results!

Algorithm		$P_{0.5}$	$R_{0.5}$	$F_{0.5}$	P_3	R_3	F_3
NTD-based autosimilarity		53.3%	62.1%	56.6%	66.8%	78.1%	71.1%
Barwise chromagram autosimilarity		43.1%	45.7%	43.9%	64.8%	68.0%	65.8%
Foote Novelty[Foote2000]	Original	29.7%	22.3%	25.1%	63.9%	48.6%	54.5%
	Aligned	42.0%	30.0%	34.5%	67.1%	47.7%	55.0%
ConvexNMF[Nieto2013]	Original	22.8%	21.5%	21.5%	46.8%	45.1%	44.7%
	Aligned	31.6%	28.1%	28.8%	50.7%	45.4%	46.5%
Spectral Clustering[McFee2014]	Original	31.2%	30.5%	29.4%	60.7%	60.8%	58.1%
	Aligned	49.2%	45.0%	45.0%	65.5%	60.6%	60.3%

Table: Averaged segmentation scores, and their comparison with several “blind” reference methods.

Algorithm	$P_{0.5}$	$R_{0.5}$	$F_{0.5}$	P_3	R_3	F_3
NTD, with “oracle ranks” for each song	67.1%	78.2%	71.5%	78.5%	90.2%	83.1%
Neural Networks[Grill2015]	80.4%	62.7%	69.7%	91.9%	71.1%	79.3%

Table: Averaged segmentation scores in the “oracle ranks” condition, compared to the current state-of-the-art (non-blind) method.

An algorithmic road

HALS principles
~2008
Nonnegative Matrix factorization

Nicolas Gillis, UMONS

Implementation and acceleration
~2012

PARAFAC decomposition
~2019

Nonnegative Tucker
~2020

Packages nnfac and MusicNTD

An algorithmic road

nnfac

Inria

TensorLy

TensorFlow PyTorch NVIDIA CUDA

Roadmap

1. An introduction to tensor methods

2. Nonnegative Tucker decomposition of music for automatic segmentation

3. Heuristic extrapolation of alternating algorithms for nonnegative tensor decomposition

Another team effort

Andersen Ang	Thi Khanh Hien Le	Nicolas Gillis
Post-doc, Univ. Waterloo	Post-doc, UMONS	Ass. Prof, UMONS

A. M. S. Ang, J. E. Cohen, N. Gillis, L. T. K. Hien, "Accelerating Block Coordinate Descent for Nonnegative Tensor Factorization", Numerical Linear Algebra Appl., 2021;e2373.

Approximate CPD

- Often, $\mathcal{T} \approx \sum_{q}^{r} a_q \otimes b_q \otimes c_q$ for small r.
- However, the generic rank (i.e. rank of random tensor) is very large.
- Therefore if $\mathcal{T} = \sum_q^r a_q \otimes b_q \otimes c_q + \mathcal{N}$ with $\mathcal{N}$ some small Gaussian noise, it has approximatively rank lower than r but its exact rank is large.

Best low-rank approximate CPD

For a given rank r, the cost function

$$\eta(A, B, C) = \|\mathcal{T} - \sum_{q=1}^{r} a_q \otimes b_q \otimes c_q\|_F^2$$

has the following properties:

- it is infinitely differentiable.
- it is non-convex in (A, B, C), but quadratic in A and B and C.
- its minimum may not be attained (ill-posed problem).

Approximate Nonnegative CPD

Low-rank r approximate NCPD

Given a tensor $\mathcal{T}$, find tensor $\mathcal{G}^* = \sum_{q=1}^{r} a_q \otimes b_q \otimes c_q$ that minimizes

$$\eta(A, B, C) = \|\mathcal{T} - \sum_{q=1}^{r} a_q \otimes b_q \otimes c_q\|_F^2 \text{ so that } a_q \geq 0, b_q \geq 0, c_q \geq 0$$

- The minimum is always attained (coercivity)!
- The cost is not smooth anymore.

Well-posedness

Approximate NCPD is well posed:

- the best low nonnegative rank approximation $\mathcal{G}^*$ exists. [Lim, Comon 2009]
- *most of the time*, tensor $\mathcal{G}^*$ is unique [Qi, Lim, Comon 2016]

My favorite class of algorithms to solve aNCPD: block-coordinate descent!

Nonconvex optimization algorithms, an incomplete list

All at once

- Conjugate gradient
- ADMM
- Nonlinear Least Squares (second order)
- Levenberg Marquardt

nonnegativity imposed by interior point methods, squaring or active set.

X ADMM $<$ AOADMM, PG $<$ APG

X Typically slower than BCD

O Very efficient near optimum

Block coordinate (alternating)

- Alternating proximal gradient
- Alternating nonnegative least squares (ANLS)
- HALS
- Multiplicative updates
- AOADMM

nonnegativity imposed mostly by proximal step.

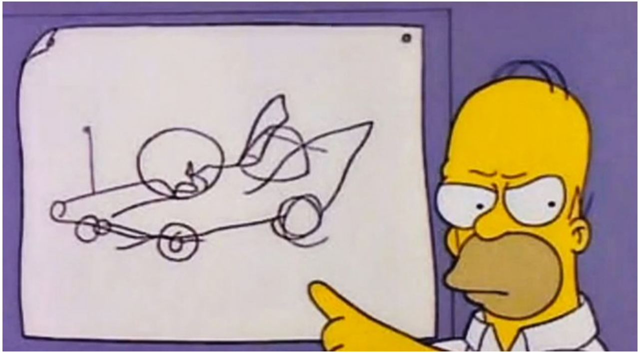
O Easy to design and implement

O Convex optimization tools

O Fast in practice

Problematic

Be cheap, be fast.



How to make tensor algorithms faster?

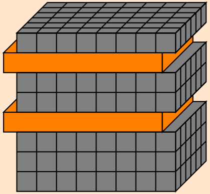
HPC

Not my expertise...

- n-mode product
- NNLS
- ??

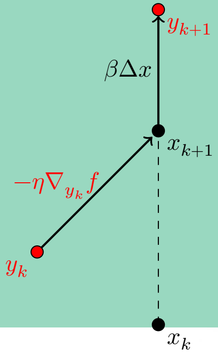
Sampling and Randomization

- Compression
- Sketching
- Subtensor sampling
- Fiber sampling
- Element-wise sampling



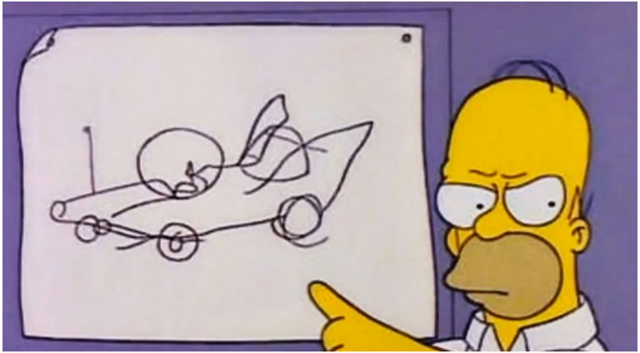
Acceleration

- Adagrad
- Momentum
- Quantification
- Extrapolation



Problematic

Be cheap, be fast.



Proposed solution: Extrapolated ANLS.

Some reminders on optimization:

- ANLS
- Nonnegative least squares
- Nesterov Extrapolation

Reminder 1: Alternating nonnegative least squares for aNCPD

Problem:

$$\underset{a_q\geq 0, b_q\geq 0, c_q\geq 0}{argmin} \|\mathcal{T} - \textstyle\sum_{q=1}^{r} a_q \otimes b_q \otimes c_q\|_F^2$$

Equivalent problem:

$$\underset{A\geq 0, B\geq 0, C\geq 0}{argmin} \|T_{[1]} - A(B \odot C)^T\|_F^2$$

where $T_{[1]}$ is an unfolding of $\mathcal{T}$ and $\odot$ is the Khatri Rao product and $A = [a_1, \ldots, a_r]$.

The ANLS algorithm (or any typicaly BCD algorithm)

loop until convergence:

- Update A using NNLS($T_{[1]}, B \odot C$)
- Update B using NNLS($T_{[2]}, A \odot B$)
- Update C using NNLS($T_{[3]}, A \odot C$)

Reminder 2: NonNegative Least Squares

U update problem: NNLS

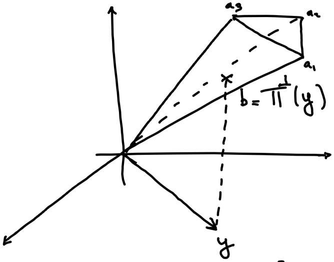
$$\underset{X \geq 0}{argmin} \; \|Y - AX\|_F^2$$

Convex!

Algorithms:

- Active set [Lawson Hanson 1974, Bro 1997]
- **Hierarchical Alternating Least Squares (HALS)**
- Block Principal Pivoting [Kim Park 2011]
- Any proximal gradient method

Note: HALS is also a BCD algorithm.



$$b = \underset{z \in cl_+(A)}{argmin} \; \|y - z\|_2^2 = Ax$$

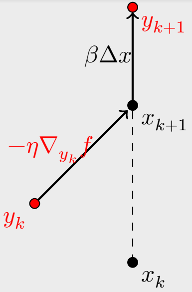
Reminder 3: Nesterov extrapolation for convex optimization

Given a (strongly) convex differentiable form f, L Lipschitz continuous, solve

$$\underset{x\in[0,1]^n}{argmin}\ f(x)$$

Fast gradient algorithm (simplified)

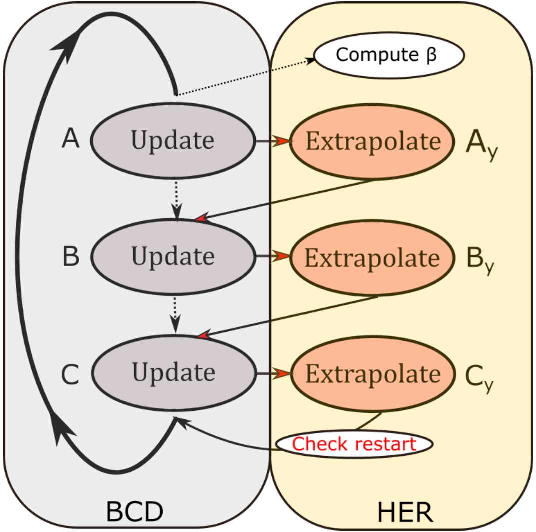
- $\eta = 1/L$; initialize x; $y = x$
- loop until convergence:
  1. $x_{old} = x$
  2. $\beta =$ some formula(β)
  3. $x = y - \eta\nabla_y f$
  4. $y = x + \beta(x - x_{old})$



Note: Step 3. can be replaced by a proximal gradient step to account for constraints.

Improves gradient descent convergence rate for strongly convex maps from $\mathcal{O}(\frac{1}{k})$ to $\mathcal{O}(\frac{1}{k^2})$.

Contribution: Heuristic Extrapolation in BCD algorithms



Heuristic Extrapolation with Restart (HER)

- Introduce pairing variables
- Update a block, then extrapolate heuristically
- Perform restart if error increases

Different from

- using extrapolation in the updates
- using extrapolation after each outer loop

Extrapolation for ANLS using HALS with restart: E-HALS

The E-HALS algorithm

- initialize A, B, C; $A_y = A, B_y = B, C_y = C$
- loop until convergence:
  1. $A_{old} = A, B_{old} = B; C_{old} = C$
  2. Update β with heuristic (next slide)
  3. Update A using NNLS($T_{[1]}, B_y \odot C_y$)
  4. Extrapolate $A_y = [A + \beta(A - A_{old})]_+$
  5. Update B using NNLS($T_{[2]}, A_y \odot C_y$)
  6. Extrapolate $B_y = [B + \beta(B - B_{old})]_+$
  7. Update C using NNLS($T_{[3]}, A_y \odot B_y$)
  8. Extrapolate $C_y = [C + \beta(C - C_{old})]_+$
- if cost function increases, restart $A_y = A, B_y = B, C_y = C$

At each iteration,

1. if error has decreased, increase β up to a threshold β_{max}.
2. if error has increased, decrease β and β_{max}.

In any case, $\beta \in]0, \beta_{max}]$ with $\beta_{max} \leq 1$.

Experimental Results: setup

Balanced dimensions, ill-conditioned factors

- $r = 10$
- $I = J = K = 50$
- Uniform A, B, C
- $a_1 = 0.01a_1 + 0.99a_2$

Unbalanced dimensions, ill-conditioned factors

- $r = 12$
- $I = 150$
- $J = 10^3$
- $K = 35$
- Uniform A, B, C
- $a_1 = 0.01a_1 + 0.99a_2$

Difficulty:

We test with HALS and ADMM nnls solvers, more in the paper!

Plots

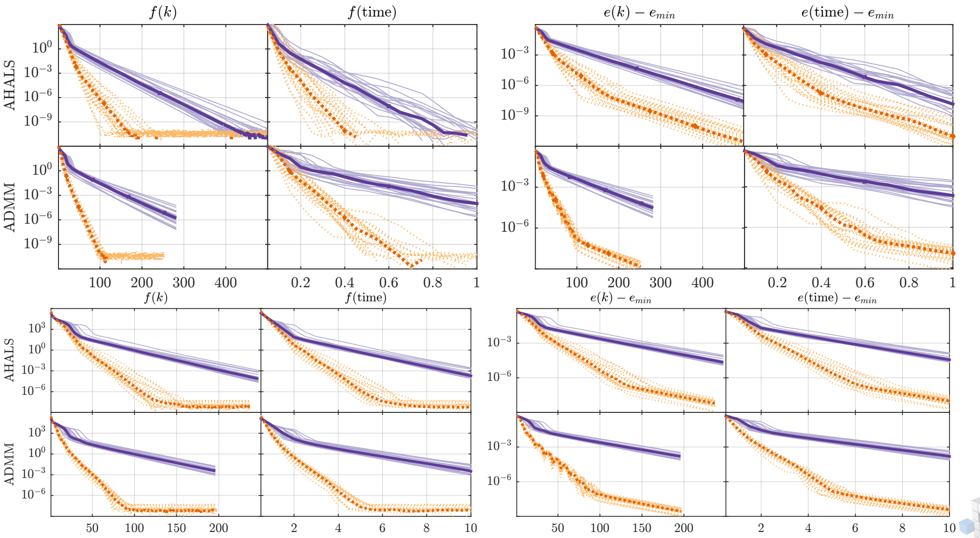


Figure: Convergence of algorithms : A-HALS and AO-ADMM without HER (solid purple) and with HER (dotted orange). Balanced dimensions, ill-conditionned factors

A few other extrapolation methods

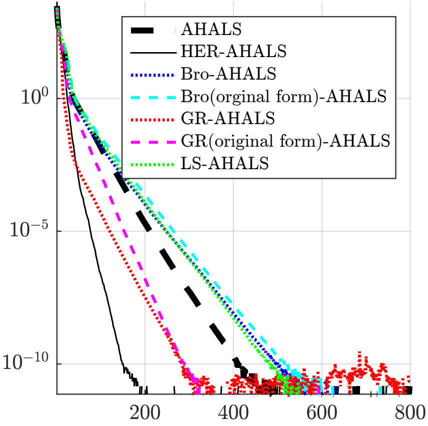


Figure: Comparing AHALS with different acceleration frameworks on synthetic datasets

My ongoing research projects

LoRAiA (ANR JCJC)

Semi-supervision and Tensors:

- Dictionaries/sparse coding
- Optimal Transport

with efficient implementations/algorithms!

Automatic Transcription

With semi-supervision and NMF.

Tensoptly (Inria)

Tensorly optimization layer:

- Constrained models
- Faster algorithms
- Customization

Music Segmentation

PhD of Axel Marmoret.

Sparse/Fast Optimization

Long-term collaboration with N. Gillis (UMONS).

Multimodality

Long-term collaboration with E. Acar (SimulaMet).

Thank you for your attention