# CPD with Structural Constraints: From BCD to Stochastic Mirror Descent

Xiao Fu

‡School of EECS, Oregon State University

**SIAM LA 2021, Virtual Conference**

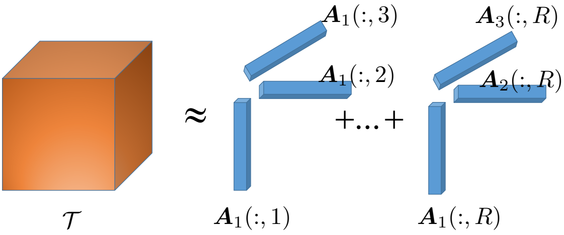
ack: Shahana Ibrahim, Kejun Huang, Wenqiang Pu, Mingyi Hong, Hoi-To Wai, Nico Vervilet, Nicolas Gillis and Lieven De Lathauwer

August 13, 2021

# This talk is based on

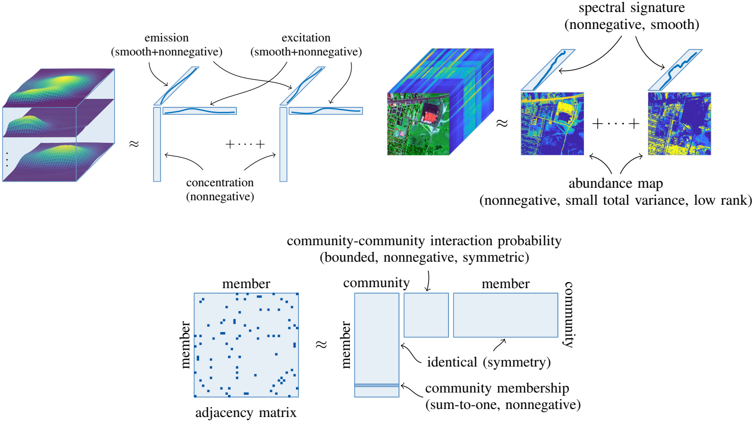
- X. Fu, N. Vervliet, L. De Lathauwer, K. Huang and N. Gillis, "Computing Large-Scale Matrix and Tensor Decomposition With Structured Factors: A Unified Nonconvex Optimization Perspective," in IEEE Signal Processing Magazine, vol. 37, no. 5, pp. 78-94, Sept. 2020.
- X. Fu, S. Ibrahim, H. Wai, C. Gao and K. Huang, "Block-Randomized Stochastic Proximal Gradient for Low-Rank Tensor Factorization," in IEEE Transactions on Signal Processing, vol. 68, pp. 2170-2185, 2020
- W. Pu, S. Ibrahim, X. Fu, and M. Hong, "Stochastic Mirror Descent for Low-Rank Tensor Decomposition Under Non-Euclidean Losses", submitted to IEEE Transaction on Signal Processing, April, 2021.

# Canonical Polyadic Decomposition (CPD)




$\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ $\quad$ $\boldsymbol{A}_n \in \mathbb{R}^{I_n \times R},\ n = 1, 2, 3$

# Decomp. with Constraints/Regularization



# CPD with Constraints/Regularization

General problem of interest:

$$\min_{\text{model param.}} \mathsf{dist}\,(\text{data}||\text{model}) + \left( \begin{array}{c} \text{penalty for} \\ \text{structure violation} \end{array} \right)$$
$$\text{under} \quad \text{structural constraints},$$

For example [Beutel et al., 2014]:

$$\min_{\{\boldsymbol{A}_n\}_{n=1}^{N}} \frac{1}{2} \|\mathcal{T} - [\![\boldsymbol{A}_1, \ldots, \boldsymbol{A}_N]\!]\|_{\mathrm{F}}^2 + \lambda \sum_{n=1}^{N} \|\boldsymbol{A}_n\|_1$$
$$\text{s.t. } \boldsymbol{A}_n \geq 0.$$

# CPD with Constraints/Regularization

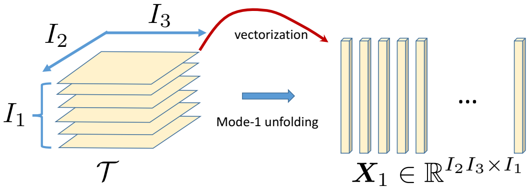
Structural info. on latent factors are useful for

- fending against noise (acting as priors);
- enhancing interpretability;
- avoiding ill-posedness [Lim and Comon, 2009];
- improving identifiability (esp.in matrix fact.) [Fu et al., 2019a];

Frequently seen constraints/regularization:

- nonnegativity: $\boldsymbol{A}_n \geq 0$ [Chi and Kolda, 2012]
- sparsity: $\|\boldsymbol{A}_n\|_1$
- prob. simplex: $1^\top \boldsymbol{A}_n = 1^\top,\ \boldsymbol{A}_n \geq 0$ [Kargas et al., 2018, Yeredor and Haardt, 2019]
- boundedness: $a \leq \boldsymbol{A}_n(i,r) \leq b$
- column/row sparsity $\|\boldsymbol{A}_n\|_{2,1}$ or $\|\boldsymbol{A}_n\|_{1,2}$ [Yang et al., 2015].
- **more**: monotonicity, smoothness, total variation, orthogonality, symmetry ... see [Sidiropoulos et al., 2017]

# First glance: Not so hard?




For all $i_1, \ldots, i_N$, define *matrix unfolding*:

$$\boldsymbol{X}_n(j, i_n) = \mathcal{T}(i_1, \ldots, i_N),$$

where $j = 1 + \sum_{\ell=1, \ell \neq n} (i_\ell - 1) J_\ell$, $J_\ell = \prod_{m=1, m\neq n}^{\ell-1} I_m$.

$$\boldsymbol{X}_n = \boldsymbol{H}_n \boldsymbol{A}_n^\top,$$

where the matrix $\boldsymbol{H}_n \in \mathbb{R}^{(\prod_{\ell=1,\ell\neq n}^{N} I_n) \times R}$ is defined as:

$$\boldsymbol{H}_n = \boldsymbol{A}_N \odot \ldots \odot \boldsymbol{A}_{n+1} \odot \boldsymbol{A}_{n-1} \odot \ldots \odot \boldsymbol{A}_1.$$

# First glance: Not so hard?

BCD updates:

$$\boldsymbol{A}_n^{(t+1)} \leftarrow \arg\min_{\boldsymbol{A}_n} \ \frac{1}{2}\|\boldsymbol{X}_n - \boldsymbol{H}_n^{(t)}\boldsymbol{A}^\top\|_{\mathrm{F}}^2 + h_n(\boldsymbol{A}_n), \tag{1}$$

where $\boldsymbol{H}_n^{(t)} = \boldsymbol{A}_N^{(t)} \odot \ldots \odot \boldsymbol{A}_{n+1}^{(t)} \odot \boldsymbol{A}_{n-1}^{(t+1)} \odot \ldots \odot \boldsymbol{A}_1^{(t+1)}$, since $\boldsymbol{A}_\ell$ for $\ell < n$ has been updated.

- The subproblem is often convex - "easy" to solve.
  - ADMM [Huang et al., 2016]
  - proximal/projected gradient descent [Lin, 2007]
  - accelerated proximal/projected gradient descent [Liavas et al., 2017]
  - inexact accelerated gradient descent [Xu and Yin, 2013]
- Convergence well understood [Razaviyayn et al., 2013].

# Bottleneck: MTTKRP

**Common challenge**: matricized tensor times Khatri–Rao product (MTTKRP), i.e., $\boldsymbol{X}_n^\top \boldsymbol{H}_n^{(t)}$.

- This single step costs $O(\prod_{n=1}^{N} I_n R)$ flop.
- It also could use up to $O(\prod_{n=1}^{N} I_n)$ memory (depending on implementation).

**Memory-efficient implementations:**

- Large-scale sparse tensor [Phipps and Kolda, 2019, Smith et al., 2015]
- Large-scale dense tensor [Ravindran et al., 2014, Kolda and Bader, 2006]
- `tensor_toolbox` has nice modules for MTTKRP if you use Matlab

# Stochastic Optimization in One Slide

General stochastic optimization under finite sum:

$$\min_{\boldsymbol{\theta}} \ \frac{1}{L}\sum_{\ell=1}^{L} f_\ell(\boldsymbol{\theta}) + h(\boldsymbol{\theta}),$$

Stochastic proximal gradient

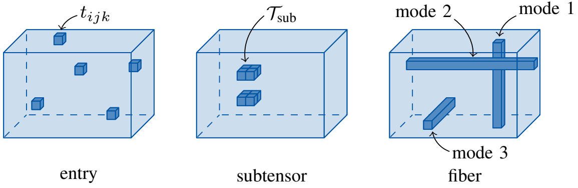
$$\boldsymbol{\theta}^{(t+1)} \leftarrow \mathrm{Prox}_h\left(\boldsymbol{\theta}^{(t)} - \alpha^{(t)}\boldsymbol{g}(\boldsymbol{\theta}^{(t)})\right),$$

where $\boldsymbol{g}(\boldsymbol{\theta}^{(t)})$ is a "stochastic oracle" evaluated at $\boldsymbol{\theta}^{(t)}$:

- **Example**: $\boldsymbol{g}(\boldsymbol{\theta}^{(t)}) = {}^{1}/_{|\mathcal{S}^{(t)}|}\sum_{\ell\in\mathcal{S}^{(t)}} \nabla f_\ell(\boldsymbol{\theta}^{(t)})$.
- $\mathcal{S}^{(t)} \subset [L]$ is a random subset.

# Sampling Tensor Data for Stochastic Opt.

A natural way to circumvent MTTKRP - working with sampled data




- Entry sampling [Beutel et al., 2014, Hong et al., 2020, Kolda and Hong, 2020]
- Subtensor sampling [Vervliet and De Lathauwer, 2016]
- Fiber sampling [Battaglino et al., 2018]

- The Euclidean loss based CPD problem can be written as:

$$\min_{\{\boldsymbol{A}_n\}_{n=1}^N} \frac{1}{L} \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} \underbrace{\left( \mathcal{T}(i_1, \ldots, i_N) - \sum_{r=1}^{R} \prod_{n=1}^{N} \boldsymbol{A}_n(i_n, r) \right)^2}_{f_{i_1,\ldots,i_N}(\boldsymbol{\theta})} + h(\boldsymbol{\theta})$$

  where $L = \prod_{n=1}^N I_n$, $h(\boldsymbol{\theta}) = \sum_{n=1}^N h_n(\boldsymbol{A}_n)$ and $\boldsymbol{\theta} = [\mathrm{vec}(\boldsymbol{A}_1)^\top, \ldots, \mathrm{vec}(\boldsymbol{A}_N)^\top]^\top$.
- The corresponding SGD update:

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \mathrm{Prox}_h \left( \boldsymbol{\theta}^{(t)} - \frac{\alpha^{(t)}}{|\mathcal{B}^{(t)}|} \sum_{(i_1,\ldots,i_N) \in \mathcal{B}^{(t)}} \nabla f_{i_1,\ldots,i_N}(\boldsymbol{\theta}^{(t)}) \right).$$

- very small per-iteration complexity.

# Stochastic Optimization for CPD

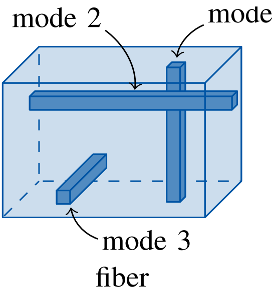
- **Challenge 1**: constraint enforcing - random sample may create some problems.
  - $\mathcal{T}(i,j,k)$ only contains info of $\boldsymbol{A}_1(i,:)$; hard to enforce constraints like $1^\top \boldsymbol{A}_1(:,r) = 1$.
- **Challenge 2**: step size scheduling - what is the best practice?

  *"One of the major issues in stochastic gradient descent (SGD) methods is how to choose an appropriate step size while running the algorithm." [Tan et al., 2016]*

  *"Determining a good learning rate becomes more of an art than science for many problems." [Zeiler, 2012]*
- **Challenge 3**: convergence analysis

# Fiber Sampling and Sketched LS

$$\underbrace{\mathcal{T}(i_1, \ldots, i_{n-1}, :, i_{n+1}, \ldots, i_N)}_{\text{a mode-}n\text{ fiber}} = \boldsymbol{X}_n(j_n, :) = \boldsymbol{H}_n(j_n, :)\boldsymbol{A}_n^\top$$




Sample a set of mode-$n$ fibers, indexed by $\mathcal{Q}_n^{(t)}$ and solve a 'sketched least squares' problem [Battaglino et al., 2018]:

$$\boldsymbol{A}_n^{(t+1)} \leftarrow \arg\min_{\boldsymbol{A}_n} \left\| \boldsymbol{X}_n(\mathcal{Q}_n^{(t)}, :) - \boldsymbol{H}_n^{(t)}(\mathcal{Q}_n^{(t)}, :)\boldsymbol{A}_n^\top \right\|_{\mathrm{F}}^2,$$

# Challenges

**Pros**:

- $\boldsymbol{A}_n^{(t+1)} \leftarrow (\boldsymbol{H}_n^{(t)}(\mathcal{Q}_n^{(t)},:)^\dagger \boldsymbol{X}_n(\mathcal{Q}_n^{(t)},:))^\top$ updates the entire $\boldsymbol{A}_n$.
- No step size selection.

**Challenges**:

- $|\mathcal{Q}_n^{(t)}| \geq R$ (suggested as $|\mathcal{Q}_n^{(t)}| = 10R \log R$ in [Battaglino et al., 2018]); can be costly when $R$ is large ($R$ could reach $O(I^2)$).
- Convergence unknown.

# Proposed Approach

**For constraints/reg.**: Use SGD instead of LS:

- Construct $\boldsymbol{G}_n^{(t)} = \boldsymbol{A}_n \boldsymbol{B}^\top \boldsymbol{B} - \boldsymbol{X}_n(\mathcal{Q}_n, :)^\top \boldsymbol{B}$ with $\boldsymbol{B} = \boldsymbol{H}^{(t)}(\mathcal{Q}_n^{(t)}, :)$. Then

$$\boldsymbol{A}_n^{(t+1)} \leftarrow \mathsf{Prox}_{h_n}\left(\boldsymbol{A}_n^{(t)} - \alpha^{(t)} \boldsymbol{G}_n^{(t)}\right).$$

- can deal with a large number of constraints with closed-from/semi-algebraic solutions.
- small memory footprint and lightweight in terms of flops.

**Remaining Challenges**:

- Step size is back - tuning can be irritating.
- $\mathbb{E}[\boldsymbol{g}] \neq \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$: not desired for SGD convergence, where $\boldsymbol{g} = [\mathrm{vec}(\boldsymbol{G}_1)^\top, \ldots, \mathrm{vec}(\boldsymbol{G}_N)^\top]^\top$.

# Simple Fix

- **Unbiased gradient estimation** - block randomization
  - for each iteration, sample a block $n$ to update (uniformly).
  - for the sampled block, sample corresponding fibers and do stochastic proximal gradient.
  - $\mathbb{E}[\boldsymbol{g}] = c\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ now holds for $c > 0$.
- **Step size rule** - ideas from deep learning

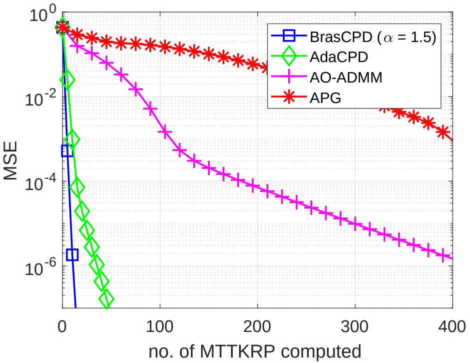
$$[\boldsymbol{\eta}_n^{(t)}]_{i,r} \leftarrow \frac{1}{\left(b + \sum_{q=1}^{t} [\boldsymbol{G}_n^{(q)}]_{i,r}^2\right)^{1/2+\epsilon}},$$

where $b$ and $\epsilon$ are for regularization purpose.

$$\boldsymbol{A}_n^{(t+1)} \leftarrow \mathrm{Prox}_{h_n}\left(\boldsymbol{A}_n^{(t)} - \boldsymbol{\eta}_n^{(t)} \circledast \boldsymbol{G}_n^{(t)}\right).$$

- This is the `adagrad` scheme [Duchi et al., 2011]; see [Kolda and Hong, 2020] for using `adam` [Kingma and Ba, 2014] for entry sampling.

# Convergence




- **Setting**: $I_1 = I_2 = I_3 = 100$, $R = 10$, NN latent factors.
- **Baselines**: AO-ADMM [Huang et al., 2016], APG [Xu and Yin, 2013].
- **Proposed**: BrasCPD (fine-tuned diminishing step size). AdaCPD (adaptive step size).
  - $|\mathcal{B}^{(t)}| = 9$ fibers sampled per iteration.

# Convergence Results - Details in [Fu et al., 2019b]

**Proposition 1.** Consider the case where $h_n(\cdot) = 0$ for all $n$, that $\alpha^{(t)}$ satisfies the Robinson-Monroe rule, and that the solution sequence is not unbounded. `BrasCPD` satisfies:

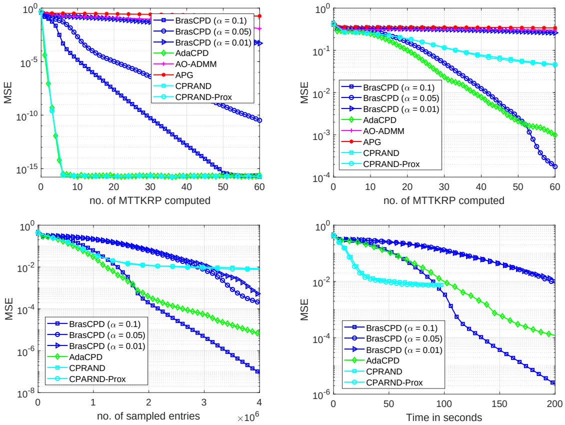
$$\liminf_{t \to \infty} \mathbb{E}\left[\left\|\nabla f(\boldsymbol{\theta}^{(t)})\right\|^2\right] = 0.$$

**Proposition 2.** In addition to the assumptions in Prop. 1, also assume that $\Pr(\xi^{(t)} = n) = 1/N$ for all $t$ and $n$. `AdaCPD` satisfies

$$\Pr\left(\liminf_{t \to \infty} \|\nabla f(\boldsymbol{\theta}^{(t)})\|^2 = 0\right) = 1.$$

**Proposition 3.** In addition to the assumptions in Prop. 1, also assume that the gradient estimation's variance diminishes when $t \to \infty$. Every limit point of `BrasCPD`'s solution sequence is a stationary point.

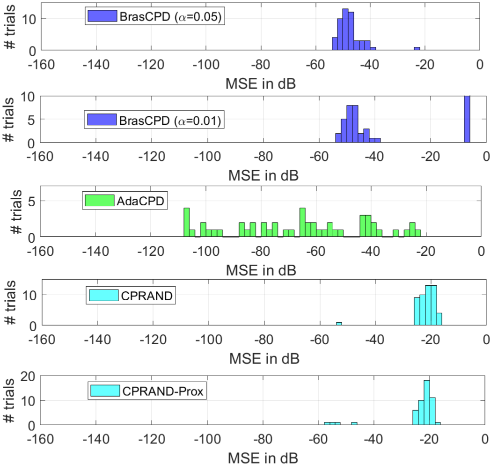
# More Results




**Setting**: $I = 300, N = 3$. Left upper: $R = 10$; Others: $R = 200$. NN constraints. 50 trials.

- CPRAND [Battaglino et al., 2018]: fiber sampling and sketched LS (no constraint).

# More Results




**Setting**: $I = 300, N = 3$. $R = 100$. 50 trials. NN constraints.

# Summary

- Stochastic optimization for tensor decomposition has become more important.
- Stochastic optimization's key considerations:
  - sampling schemes
  - step size scheduling (automatic/adaptive steps size is preferrable)
  - constraints/reg.
  - convergence supports
- The block-randomized fiber sampling strategy seems to offer a promising solution package.

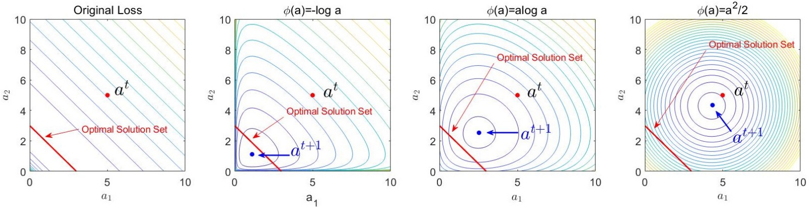
# Generalization for Non-Euclidean Losses

General problem of interest:

$$\min_{\text{model param.}} \textsf{dist}\,(\text{data}||\text{model}) + \left( \begin{array}{c} \text{penalty for} \\ \text{structure violation} \end{array} \right)$$
$$\text{under} \quad \text{structural constraints},$$

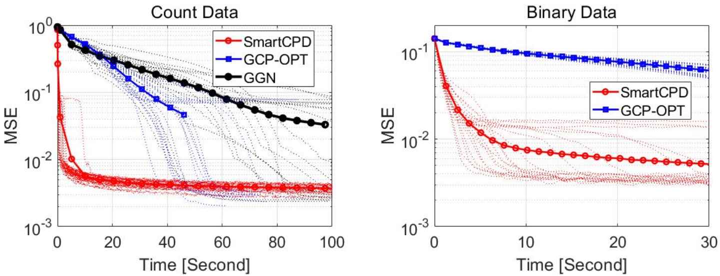
- How about $\beta$-divergence, KL-divergence, IS-divergence, Logistic loss, etc? [Cichocki et al., 2015, Chi and Kolda, 2012, Févotte et al., 2009]
  - used in integer, binary, and scaling-sensitive data analysis.
  - Entry sampling-based non-Euclidean CPD [Kolda and Hong, 2020, Hong et al., 2020].
    - SGD + `Adam`
- **Proposed**: Fiber sampling + stochastic **mirror descent**.

# More Results




MD adapts to loss function and constraint's geometry.

# SMD for CPD (SmartCPD)




- **Proposed**: SmartCPD - fiber sampling, block randomization, SMD
- **Baseline**: GCP-OPT [Hong et al., 2020, Kolda and Hong, 2020]. GGN [Vandecappelle et al., 2020].
- **Freshly baked manuscript**:
  - W. Pu, S. Ibrahim, X. Fu, and M. Hong, "Stochastic Mirror Descent for Low-Rank Tensor Decomposition Under Non-Euclidean Losses", arXiv:2104.14562 [stat.ML]

Battaglino, C., Ballard, G., and Kolda, T. G. (2018).
A practical randomized CP tensor decomposition.
39(2):876–901.

Beutel, A., Talukdar, P. P., Kumar, A., Faloutsos, C., Papalexakis, E. E., and Xing, E. P. (2014).
Flexifact: Scalable flexible factorization of coupled tensors on Hadoop.
In *Proc. SIAM SDM 2014*, pages 109–117. SIAM.

Chi, E. C. and Kolda, T. G. (2012).
On tensors, sparsity, and nonnegative factorizations.
33(4):1272–1299.

Cichocki, A., Mandic, D., De Lathauwer, L., Zhou, G., Zhao, Q., Caiafa, C., and Phan, H.-A. (2015).
Tensor decompositions for signal processing applications: From two-way to multiway component analysis.
32(2):145–163.

Duchi, J., Hazan, E., and Singer, Y. (2011).

Adaptive subgradient methods for online learning and stochastic optimization.
12(Jul):2121–2159.

Févotte, C., Bertin, N., and Durrieu, J.-L. (2009).
Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis.
*Neural computation*, 21(3):793–830.

Fu, X., Huang, K., Sidiropoulos, N. D., and Ma, W.-K. (2019a).
Nonnegative matrix factorization for signal and data analytics: Identifiability, algorithms, and applications.
*IEEE Signal Process. Mag.*, 36(2):59–80.

Fu, X., Ibrahim, S., Wai, H.-T., Gao, C., and Huang, K. (2019b).
Block-randomized stochastic proximal gradient for low-rank tensor factorization.
*arXiv preprint arXiv:1901.05529*.

Hong, D., Kolda, T. G., and Duersch, J. A. (2020).

Generalized canonical polyadic tensor decomposition.
*SIAM Review*, 62(1):133–163.

Huang, K., Sidiropoulos, N. D., and Liavas, A. P. (2016).
A flexible and efficient algorithmic framework for constrained matrix and tensor factorization.
*IEEE Trans. Signal Process.*, 64(19):5052–5065.

Kargas, N., Sidiropoulos, N. D., and Fu, X. (2018).
Tensors, learning, and 'Kolmogorov extension'for finite-alphabet random vectors.
*IEEE Trans. Signal Process.*, 66(18):4854–4868.

Kingma, D. P. and Ba, J. (2014).
Adam: A method for stochastic optimization.
*arXiv preprint arXiv:1412.6980*.

Kolda, T. G. and Bader, B. W. (2006).
Matlab tensor toolbox.
Technical report, Sandia National Laboratories.

Kolda, T. G. and Hong, D. (2020).

Stochastic gradients for large-scale tensor decomposition.
*SIAM Journal on Mathematics of Data Science*, 2(4):1066–1095.

Liavas, A. P., Kostoulas, G., Lourakis, G., Huang, K., and Sidiropoulos, N. D. (2017).
Nesterov-based alternating optimization for nonnegative tensor factorization: Algorithm and parallel implementation.
*IEEE Trans. Signal Process.*, 66(4):944–953.

Lim, L.-H. and Comon, P. (2009).
Nonnegative approximations of nonnegative tensors.
23(7-8):432–441.

Lin, C.-J. (2007).
Projected gradient methods for nonnegative matrix factorization.
19(10):2756–2779.

Phipps, E. and Kolda, T. G. (2019).
Software for sparse tensor decomposition on emerging computing architectures.

*SIAM Journal on Scientific Computing*, 41(3):C269–C290.

Ravindran, N., Sidiropoulos, N. D., Smith, S., and Karypis, G. (2014).
Memory-efficient parallel computation of tensor and matrix products for big tensor decomposition.
In *2014 48th Asilomar Conference on Signals, Systems and Computers*, pages 581–585. IEEE.

Razaviyayn, M., Hong, M., and Luo, Z.-Q. (2013).
A unified convergence analysis of block successive minimization methods for nonsmooth optimization.
23(2):1126–1153.

Sidiropoulos, N. D., De Lathauwer, L., Fu, X., Huang, K., Papalexakis, E. E., and Faloutsos, C. (2017).
Tensor decomposition for signal processing and machine learning.
*IEEE Trans. Signal Process.*, 65(13):3551–3582.

Smith, S., Ravindran, N., Sidiropoulos, N. D., and Karypis, G. (2015).

Splatt: Efficient and parallel sparse tensor-matrix multiplication.
In *2015 IEEE International Parallel and Distributed Processing Symposium*, pages 61–70.

Tan, C., Ma, S., Dai, Y.-H., and Qian, Y. (2016).
Barzilai-borwein step size for stochastic gradient descent.
*arXiv preprint arXiv:1605.04131*.

Vandecappelle, M., Vervliet, N., and De Lathauwer, L. (2020).
A second-order method for fitting the canonical polyadic decomposition with non-least-squares cost.
*IEEE Transactions on Signal Processing*, 68:4454–4465.

Vervliet, N. and De Lathauwer, L. (2016).
A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors.
*IEEE J. Sel. Topics Signal Process.*, 10(2):284–295.

Xu, Y. and Yin, W. (2013).

A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion.
6(3):1758–1789.

Yang, Y., Feng, Y., and Suykens, J. A. (2015).
A rank-one tensor updating algorithm for tensor completion.
*IEEE Signal Processing Letters*, 22(10):1633–1637.

Yeredor, A. and Haardt, M. (2019).
Maximum likelihood estimation of a low-rank probability mass tensor from partial observations.
*IEEE Signal Process. Lett.*, 26(10):1551–1555.

Zeiler, M. D. (2012).
Adadelta: an adaptive learning rate method.
*arXiv preprint arXiv:1212.5701*.