Bregman Proximal Gradient Framework for Deep Linear Neural Networks

Peter Ochs

Mathematical Optimization Group

— 16.03.2021 —

A class of matrix factorization problems

- Decompose matrix A into product UZ with matrices U and Z:

$$A \approx UZ$$

- Applications require certain properties of U and Z, e.g. sparsity, non-negativity, uni row/column sum, low-rank, ...

- Non-smooth non-convex Optimization problem:

$$\min_{U,Z} \frac{1}{2}\|A - UZ\|_F^2 + \mathcal{R}_1(U) + \mathcal{R}_2(Z)$$

- $\mathcal{R}_1$ and $\mathcal{R}_2$ can be non-convex regularization terms or constraints.

Outlook: Deep Linear Neural Networks / Deep Matrix Factorization

$$\min_{W_1,\dots,W_N} \frac{1}{2}\|Y - W_1 W_2 \cdots W_N X\|_F^2 + \sum_{i=1}^{N} \mathcal{R}_i(W_i)$$

Matrix Factorization: Alternating Minimization

How to solve the problem?

$$\min_{U,Z} Q(U,Z) + \mathcal{R}_1(U) + \mathcal{R}_2(Z)\,, \quad Q(U,Z) := \frac{1}{2}\|A - UZ\|_F^2$$

Matrix Factorization: Alternating Minimization

How to solve the problem?

$$\min_{U,Z} Q(U,Z) + \mathcal{R}_1(U) + \mathcal{R}_2(Z)\,, \quad Q(U,Z) := \frac{1}{2}\|A - UZ\|_F^2$$

- **Alternating Minimization:**

$$U^{(k+1)} \in \operatorname*{argmin}_U Q(U, Z^{(k)}) + \mathcal{R}_1(U)$$
$$Z^{(k+1)} \in \operatorname*{argmin}_Z Q(U^{(k+1)}, Z) + \mathcal{R}_2(Z)$$

- Often biased towards one of the variables.
- Can be slow.
- **Almost no convergence guarantees in non-smooth setting.**
- *Variant:* HALS [Cichocki, Phan 09]:
  AM on columns of U and Z (closed form updates for NMF).

Matrix Factorization: PALM

$$\min_{U,Z} Q(U,Z) + \mathcal{R}_1(U) + \mathcal{R}_2(Z)\,, \quad Q(U,Z) := \frac{1}{2}\|A - UZ\|_F^2$$

- **Proximal Alternating Linearized Min. (PALM)** [Bolte, Sabach, Teboulle 14]

$$U^{(k+1)} \in \operatorname{prox}_{\tau_k \mathcal{R}_1}\Big(U^{(k)} - \tau_k \nabla_U Q(U^{(k)}, Z^{(k)})\Big)$$
$$Z^{(k+1)} \in \operatorname{prox}_{\sigma_k \mathcal{R}_2}\Big(Z^{(k)} - \sigma_k \nabla_Z Q(U^{(k+1)}, Z^{(k)})\Big)$$

  with step sizes $0 < \tau_k < 1/L_1(Z^{(k)})$ and $0 < \sigma_k < 1/L_2(U^{(k+1)})$.

- Computing L_1 and L_2 can be costly or require severe overestimation.
- Often biased towards one of the variables.
- Guarantees **convergence to stationary point**.
- *Variants:* PALM, iPALM, BCD, BC-VMFB, . . .

Matrix Factorization: Proximal Gradient Method

$$\min_{U,Z} Q(U,Z) + \mathcal{R}_1(U) + \mathcal{R}_2(Z)\,, \quad Q(U,Z) := \frac{1}{2}\|A - UZ\|_F^2$$

- **Proximal Gradient Descent**

$$(U^{(k+1)}, Z^{(k+1)}) \in \mathrm{prox}_{\tau_k \mathcal{R}_1 \oplus \mathcal{R}_2}\Big((U^{(k)}, Z^{(k)}) - \tau_k \nabla Q(U^{(k)}, Z^{(k)})\Big)$$

  with step sizes τ_k computed by (backtracking) line search.

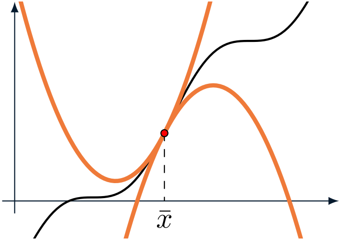
- ∇Q is not Lipschitz continuous.
- Procedure can be arbitrarily slow.
- Line search requires extra loop and function evaluations.
- Guarantees **convergence to stationary point**.

Lack of Lipschitz Continuity

Relative Smoothness:

- Concept proposed by [Birnbaum, Devanur, Xiao 2011].
- Popularized by [Bauschke, Bolte, Teboulle 2017] and [Bolte et al. 2018].
- **Idea:** Key for convergence analysis is the Descent Lemma.

⇝ Quadratic upper and lower bounds.

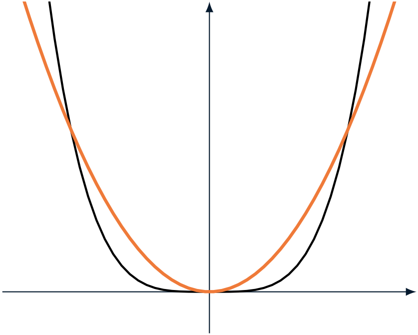


∇f is L-Lipschitz

$$\Longrightarrow |f(x) - f(\bar{x}) - \langle \nabla f(\bar{x}), x - \bar{x} \rangle| \leq \frac{L}{2} \|x - \bar{x}\|^2$$

Relative Smoothness (cont.):

- Simple functions like x^4 **do not** allow for quadratic bounds:



- Same situation in Matrix Factorization, e.g., for $Z = U^\top$:

$$Q(U, U^\top) = \frac{1}{2}\|A - UU^\top\|_F^2 \quad \text{polynomial of degree } 4 \text{ in } U$$

Relative Smoothness (cont.):

- **Remedy: Generalized Descent Lemma** w.r.t. Bregman distances:

$$-\underline{L}D_h(x,\bar{x}) \leq f(x) - f(\bar{x}) - \langle \nabla f(\bar{x}), x - \bar{x} \rangle \leq \overline{L}D_h(x,\bar{x})$$

$\rightsquigarrow$ **Define:** f is L-relatively smooth w.r.t. h. *(Also called L-smad).*

- **Bregman distance:** (generalized distance measure)

$$D_h(x,\bar{x}) := h(x) - h(\bar{x}) - \langle \nabla h(\bar{x}), x - \bar{x} \rangle \,.$$

- h is assumed to have good properties *(Legendre function)*.

$\rightsquigarrow$ **upper and lower bounds are adapted to the objective**.

Bregman Proximal Gradient Algorithm [Bolte, Sabach, Teboulle, Vaisbourd 18]

- **BGP for Matrix Factorization Problem:** [Mukkamala, O. 19]

$$\min_{U,Z} Q(U,Z) + \mathcal{R}_1(U) + \mathcal{R}_2(Z)$$

Q is L-relatively smooth w.r.t. some h *(see next slide)*.

- **Update step:**

$$C^{(k)} := \nabla Q(U^{(k)}, Z^{(k)}) - \frac{1}{\tau}\nabla h(U^{(k)}, Z^{(k)})$$

$$(U^{(k+1)}, Z^{(k+1)}) \in \operatorname*{argmin}_{U,Z} \mathcal{R}_1(U) + \mathcal{R}_2(Z) + \langle C^{(k)}, (U,Z)\rangle + \frac{1}{\tau}h(U,Z)$$

with step size $\tau < 1/L$.

- Guarantees **convergence to a stationary point.**

New Bregman Distance for Matrix Factorization [Mukkamala, O. 19]

- $Q(U, Z) = \frac{1}{2}\|A - UZ\|_F^2$ is relatively smooth w.r.t.

$$h(U, Z) = 3\left(\frac{\|U\|_F^2 + \|Z\|_F^2}{2}\right)^2 + \|A\|_F\left(\frac{\|U\|_F^2 + \|Z\|_F^2}{2}\right).$$

- The update step can be computed efficiently (in closed form) for $\|\cdot\|_F^2$, $\|\cdot\|_1$, $\|\cdot\|_*$, ℓ_0-sparsity constraints, non-negativity constraints.

- Usually reduces to a nesting of the Euclidean proximal mapping with a one dimensional root finding problem of a cubic polynomial.

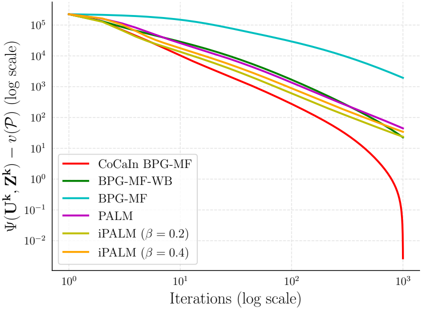
- Symmetric MF setting developed in [Dragomir, Bolte, d'Aspremont 19].

New Bregman Distance for Matrix Factorization

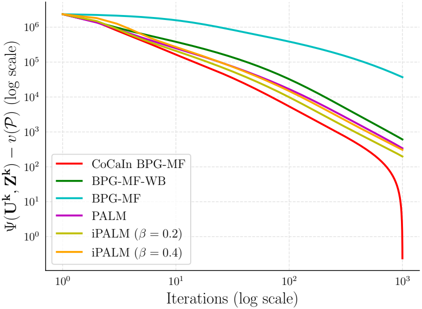
Modifications:

- Matrix completion
- All Bregman based algorithms can be used !
- BPG for MF can be extended to **inertial algorithms** such as CoCaIn [Mukkamala, Ochs, Pock, Sabach 20].
- There are stochastic variants of BPG.

Matrix Completion on MovieLens Datasets:



MovieLens-100K



MovieLens-1M

Deep Linear Neural Networks or Deep Matrix Factorization:

$$\min_{W_1,\ldots,W_N} \frac{1}{2}\|Y - W_1 W_2 \cdots W_N X\|_F^2 + \sum_{i=1}^{N} \mathcal{R}_i(W_i)$$

Deep Linear Neural Networks or Deep Matrix Factorization:

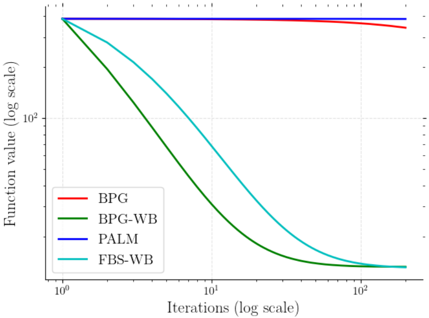
$$\min_{W_1,\ldots,W_N} \frac{1}{2}\|Y - W_1 W_2 \cdots W_N X\|_F^2 + \sum_{i=1}^{N} \mathcal{R}_i(W_i)$$

- Write: $\boldsymbol{W} := (W_1, \ldots, W_N)$ and $\|\boldsymbol{W}\|_F^2 := \sum_{i=1}^N \|W_i\|_F^2$.
- [Mukkamala et al. 2021] shows relative smoothness w.r.t.

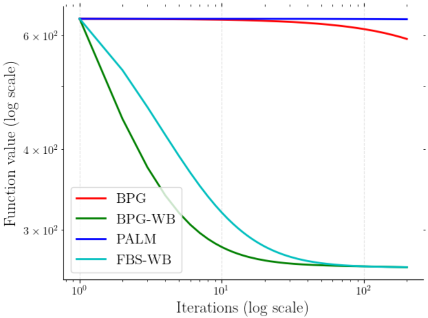
$$h(\boldsymbol{W}) = \begin{cases} \frac{\|X\|_F^2}{N^{N-2}}\|\boldsymbol{W}\|_F^{2N} + \frac{\|Y\|_F\|X\|_F}{(N-2)^{\frac{N-2}{2}}}\|\boldsymbol{W}\|_F^N, & \text{if } N \text{ is even} \\ \frac{\|X\|_F^2}{N^{N-2}}\|\boldsymbol{W}\|_F^{2N} + \frac{\|Y\|_F\|X\|_F}{(N-1)^{\frac{N-1}{2}}}\left(\|\boldsymbol{W}\|_F^2 + 1\right)^{\frac{N+1}{2}}, & \text{if } N \text{ is odd}. \end{cases}$$

⇝ optimization / training with **constant step size rule** using BPG.

Matrix Completion on MovieLens Datasets:



L2-Regularization ($N = 4$)
MovieLens-100K



L1-Regularization ($N = 4$)
MovieLens-100K

Summary:

- Matrix factorization problems are usually solved by alternating minimization or a line search based Proximal Gradient Algorithm.
- Remedy by Bregman Proximal Gradient Algorithm and the concept of relative smoothness.

$\rightsquigarrow$ Adapts algorithm to geometry of given problem.

- Objective in Matrix Factorization and Deep Linear Networks are relatively smooth.
- Allows variants of BPG to be applied.