Cyclic Block Optimization

How they work, Why they work, and Where they work

Hanbaek Lyu

Department of Mathematics

University of Wisconsin – Madison

Partially supported by NSF DMS #2206296

Michigan AIM Seminar

Feb. 21, 2025

Students and Collaborators

Yuchen Li
(UW Math)

Rahul Choudhary
(UW CS)

Joowon Lee
(UW Stats)

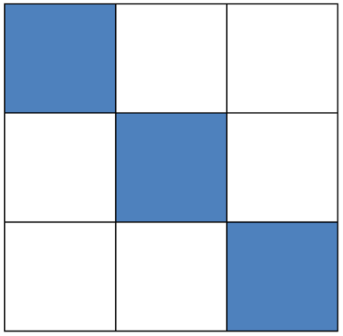
Sumit Mukherjee
(Columbia Stats)

Laura Balzano
(Michigan EECS)

Deanna Needell
(UCLA Math)

Cyclic Block Optimization

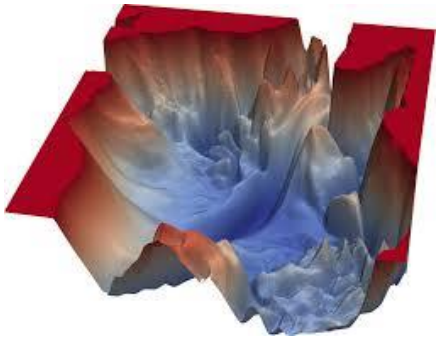
Optimize one block at a time!



Introduction

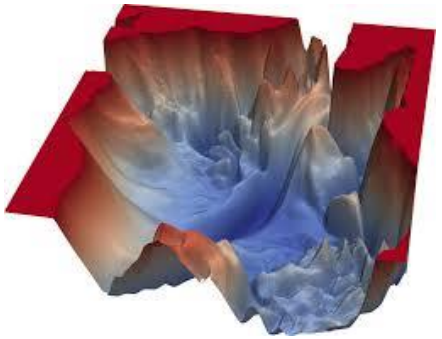
$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta} \in \boldsymbol{\Theta} \subseteq \mathbb{R}^p}{\arg\min}\, f(\boldsymbol{\theta})$$

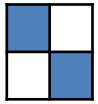
- f: Objective function
  - Nonconvex, smooth (later non-smooth)
  - Often represents model fitness to training data
    - e.g., DNN, LLM

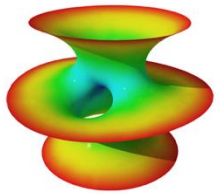


$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta}\in\boldsymbol{\Theta}\subseteq\mathbb{R}^p}{\arg\min}\, f(\boldsymbol{\theta})$$

- f: Objective function
  - Nonconvex, smooth (later non-smooth)
  - Often represents model fitness to training data
    - e.g., DNN, LLM

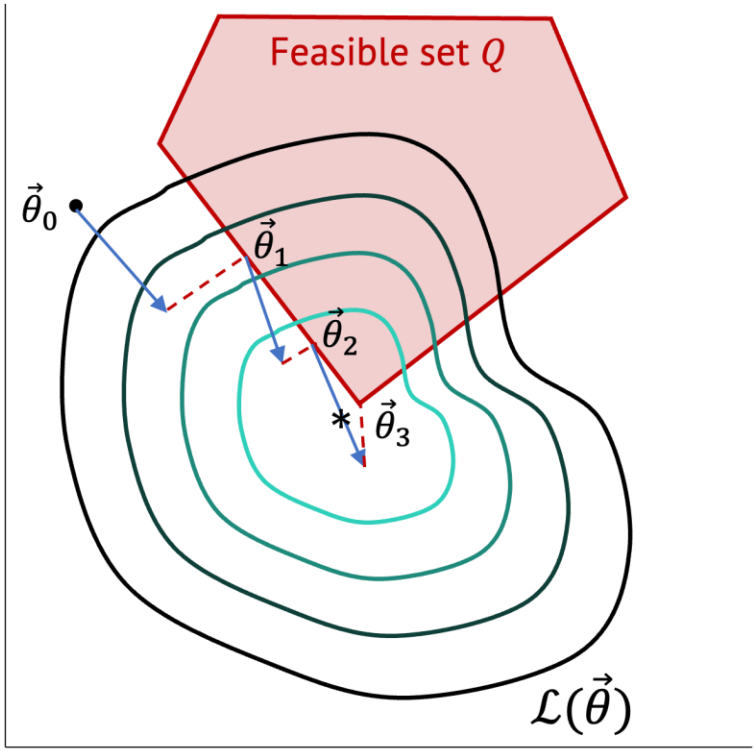


- $\boldsymbol{\Theta}$: Parameter space with various structures:
  - Convex 
  - block structure: $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_m)$ 
  - Riemannian manifold (e.g., $\boldsymbol{\Theta}$=set of orthogonal matrices)



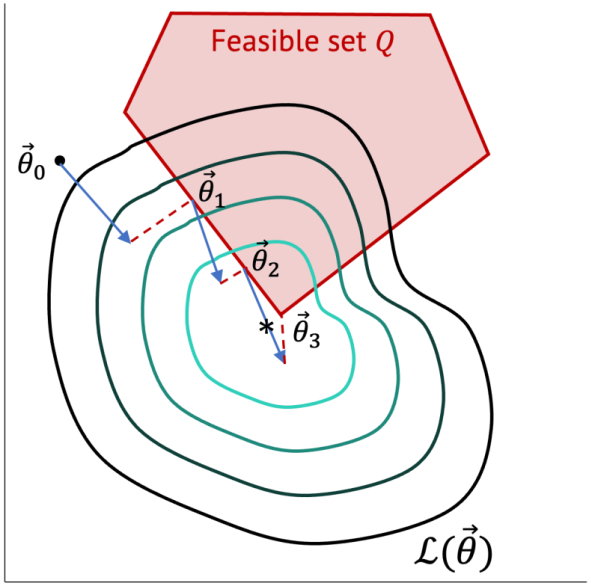
The simplest yet commonly used method

(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n) \right)$$



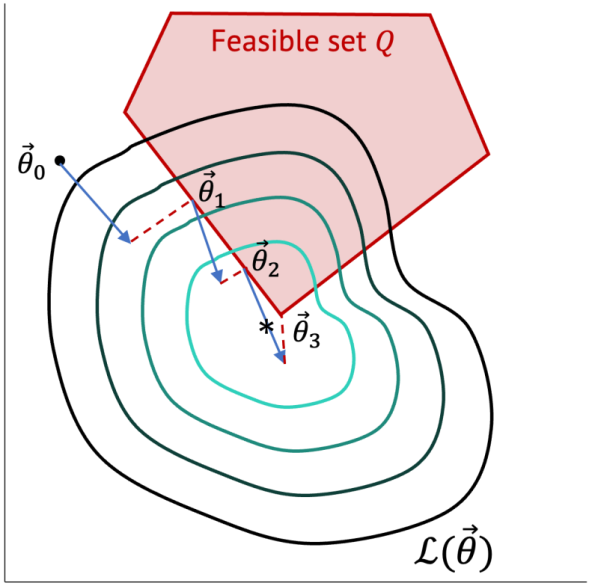
(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n) \right)$$

- Projected Gradient Descent (PGD)
  - Widely used in ML, Statistics, Scientific Computing
  - Reduces to Gradient Descent for unconstrained problems
  - Easy to implement, cheap to run
  - Several of modifications (e.g., momentum, adaptive stepsizes)



(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n) \right)$$

- Projected Gradient Descent (PGD)
  - Widely used in ML, Statistics, Scientific Computing
  - Reduces to Gradient Descent for unconstrained problems
  - Easy to implement, cheap to run
  - Several of modifications (e.g., momentum, adaptive stepsizes)



(PSGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \alpha_n \widehat{\nabla} f(\boldsymbol{\theta}_n) \right)$$

Stochastic gradient $\approx \nabla$

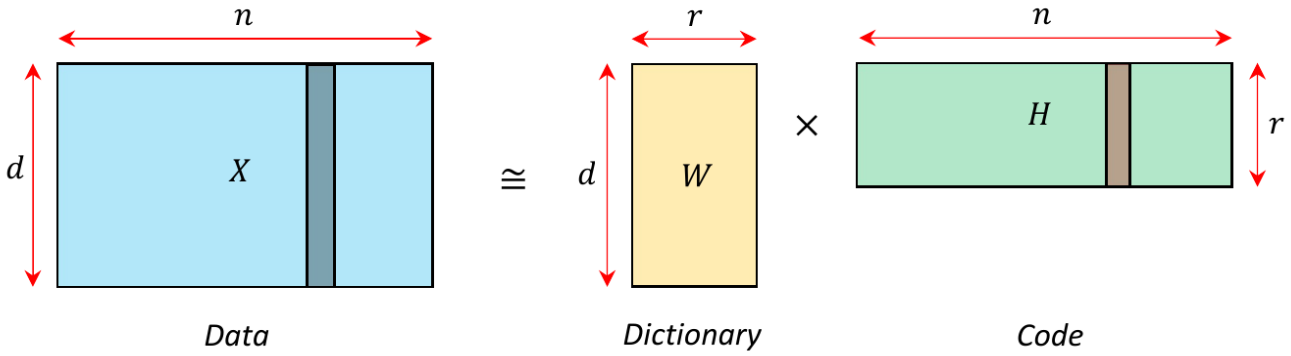
- Least Squares: Classical setting for linear regression

$$\min_{\mathbf{H}} \|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2$$

- Least Squares: Classical setting for linear regression

$$\min_{\mathbf{H}} \|\mathbf{X} - \mathbf{WH}\|_F^2$$

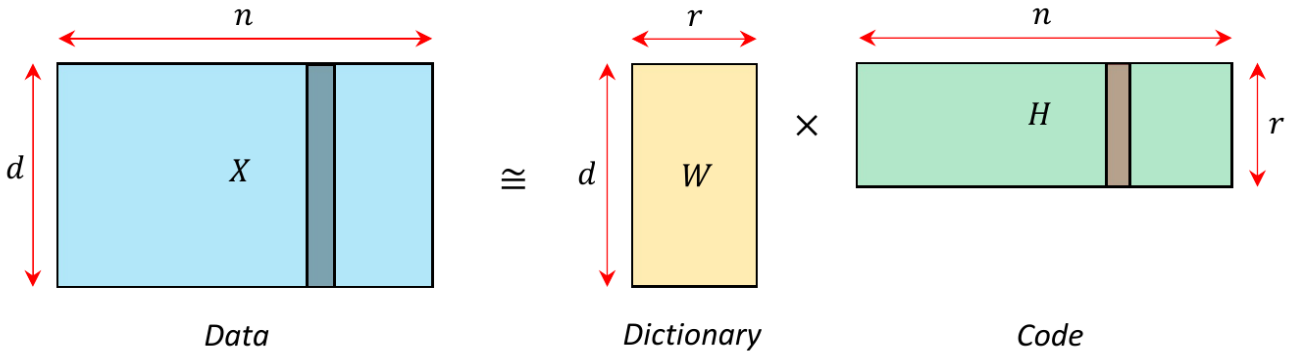
- Data $\approx$ Linear combination of $\overbrace{\text{basis features}}^{\text{cols. of } W}$



- Least Squares: Classical setting for linear regression

$$\min_{\mathbf{H}} \|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2$$

  - Data $\approx$ Linear combination of $\overbrace{\text{basis features}}^{\text{cols. of } W}$

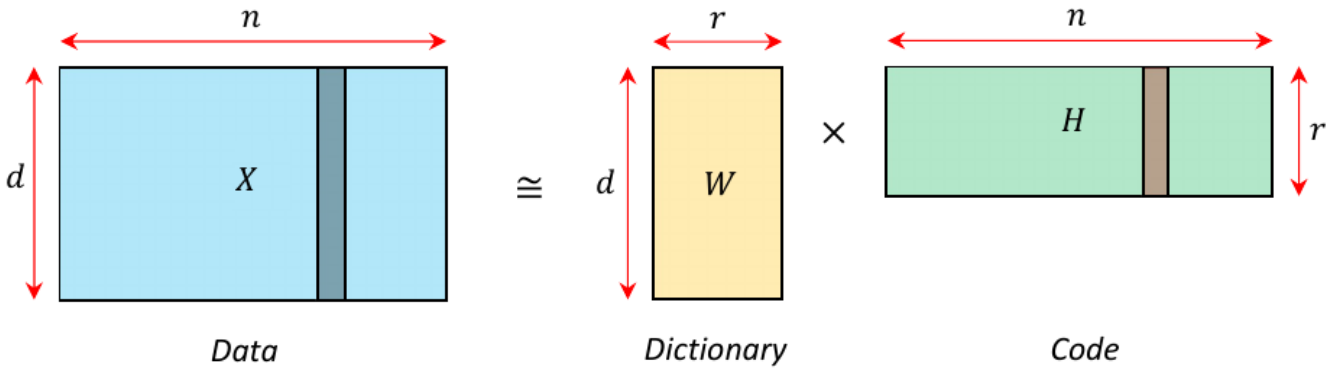


  - Convex optimization problem with closed-form solution
    - $\widehat{\mathbf{H}} = \left(\mathbf{W}^{\mathbf{T}}\mathbf{W}\right)^{\dagger}\mathbf{W}^{\mathbf{T}}\mathbf{X}$

- Q: What if we don't know what basis features $\mathbf{W}$ to use?

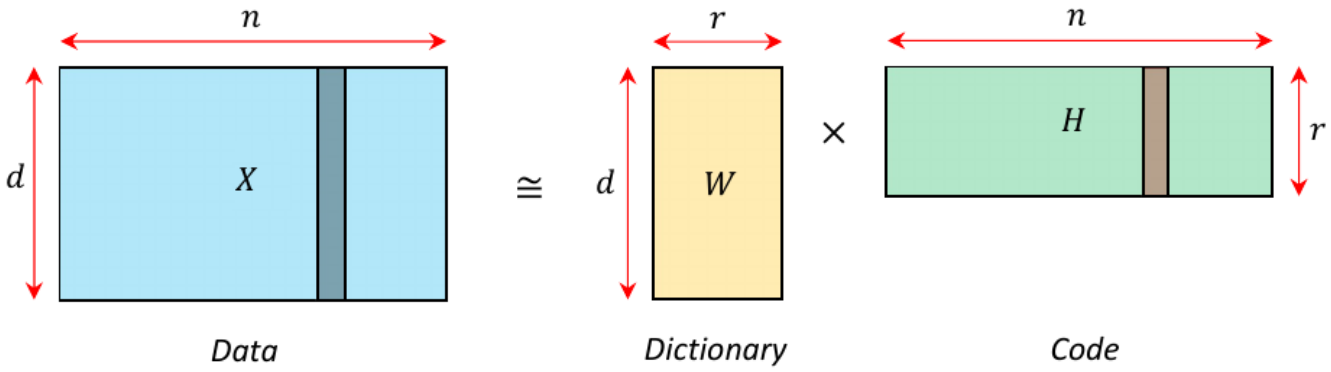
- Q: What if we don't know what basis features $\mathbf{W}$ to use?
  - Simultaneously find the basis $\mathbf{W}$ and the linear representation $\mathbf{H}$ for the data $\mathbf{X}$?

- Q: What if we don't know what basis features $\mathbf{W}$ to use?
  - Simultaneously find the basis $\mathbf{W}$ and the linear representation $\mathbf{H}$ for the data $\mathbf{X}$?
- Matrix factorization is a fundamental tool in dictionary learning problems.



Data $\approx$ Linear combination of $\overbrace{\text{latent features}}^{\text{cols. of } W}$

- Q: What if we don't know what basis features $\mathbf{W}$ to use?
  - Simultaneously find the basis $\mathbf{W}$ and the linear representation $\mathbf{H}$ for the data $\mathbf{X}$?
- Matrix factorization is a fundamental tool in dictionary learning problems.

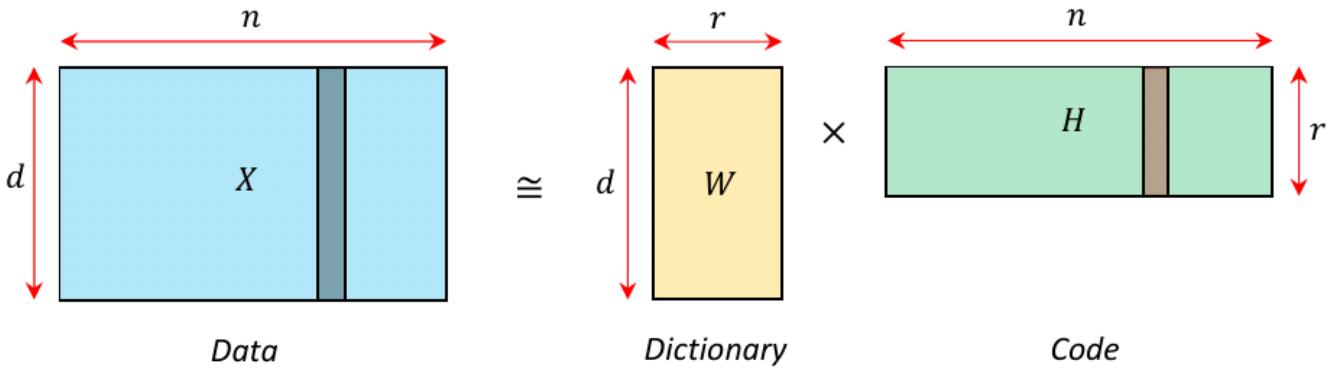


Data ≈ Linear combination of $\overbrace{\text{latent features}}^{\text{cols. of } W}$

- Formulated as a nonconvex optimization problem:

$$\begin{cases} \min_{\mathbf{W},\mathbf{H}} & \|\mathbf{X}-\mathbf{W}\mathbf{H}\|_F^2 & \text{(Reconstruction error)} \\ \text{subject to} & \mathbf{W}\in\mathscr{C},\ \mathbf{H}\in\mathscr{C}' & \text{(Constraints)} \end{cases}$$

- Q: What if we don't know what basis features $\mathbf{W}$ to use?
  - Simultaneously find the basis $\mathbf{W}$ and the linear representation $\mathbf{H}$ for the data $\mathbf{X}$?
- Matrix factorization is a fundamental tool in dictionary learning problems.



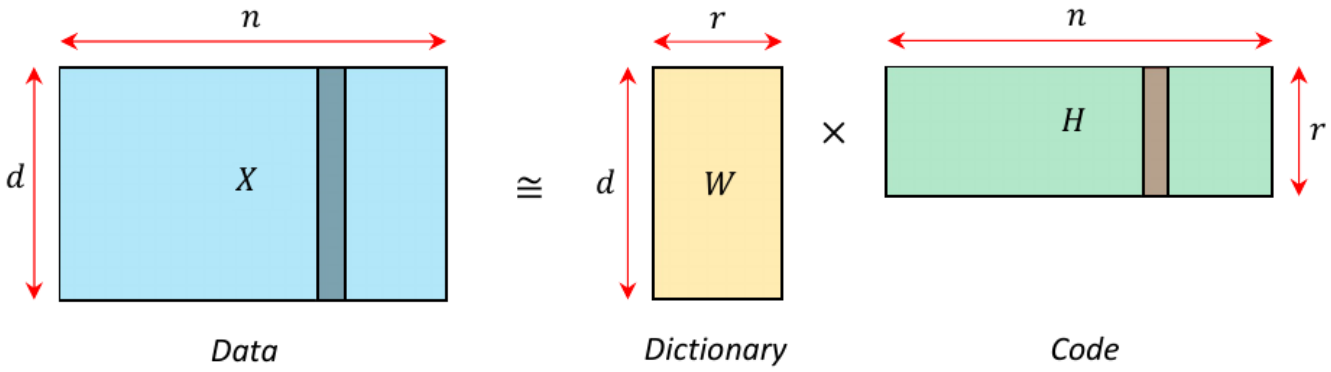
Data ≈ Linear combination of $\overbrace{\text{latent features}}^{\text{cols. of } W}$

- Formulated as a nonconvex optimization problem:

$$\begin{cases} \min_{\mathbf{W},\mathbf{H}} & \|\mathbf{X}-\mathbf{W}\mathbf{H}\|_F^2 & \text{(Reconstruction error)} \\ \text{subject to} & \mathbf{W}\in\mathscr{C}, \mathbf{H}\in\mathscr{C}' & \text{(Constraints)} \end{cases}$$

  - Unconstrained MF ($\mathscr{C}=\mathbb{R}^{d\times r}$, $\mathscr{C}'=\mathbb{R}^{r\times n}$): Global min attained by SVD

- Q: What if we don't know what basis features $\mathbf{W}$ to use?
  - Simultaneously find the basis $\mathbf{W}$ and the linear representation $\mathbf{H}$ for the data $\mathbf{X}$?
- Matrix factorization is a fundamental tool in dictionary learning problems.



Data ≈ Linear combination of $\overbrace{\text{latent features}}^{\text{cols. of } W}$

- Formulated as a nonconvex optimization problem:

$$\begin{cases} \min_{\mathbf{W},\mathbf{H}} & \|\mathbf{X}-\mathbf{W}\mathbf{H}\|_F^2 & \text{(Reconstruction error)} \\ \text{subject to} & \mathbf{W}\in\mathscr{C}, \mathbf{H}\in\mathscr{C}' & \text{(Constraints)} \end{cases}$$

  - Unconstrained MF ($\mathscr{C}=\mathbb{R}^{d\times r}$, $\mathscr{C}'=\mathbb{R}^{r\times n}$): Global min attained by SVD
  - Nonnegative Matrix Factorization (NMF): $\mathscr{C}=\mathbb{R}^{d\times r}_{\geq 0}$, $\mathscr{C}'=\mathbb{R}^{r\times n}_{\geq 0}$

- How do we solve NMF?

$$\min_{\mathbf{W}\in\mathbb{R}_{\geq 0}^{d\times r},\mathbf{H}_{\geq 0}^{r\times n}} \left[f(\mathbf{W},\mathbf{H}) := \|\mathbf{X}-\mathbf{W}\mathbf{H}\|_F^2 \right]$$

- How do we solve NMF?

$$\min_{\mathbf{W}\in\mathbb{R}^{d\times r}_{\geq 0},\mathbf{H}^{r\times n}_{\geq 0}} \left[f(\mathbf{W},\mathbf{H}) := \|\mathbf{X}-\mathbf{W}\mathbf{H}\|_F^2 \right]$$

  - Can't find both $\mathbf{W}$ and $\mathbf{H}$ at the same time, so alternate!

$$\mathbf{H}_{t+1} \leftarrow \operatorname*{argmin}_{\mathbf{H}\in\mathbb{R}^{r\times n}_{\geq 0}} f(\mathbf{W}_t,\mathbf{H}) \qquad (NLS)$$

$$\mathbf{W}_{t+1} \leftarrow \operatorname*{argmin}_{\mathbf{W}\in\mathbb{R}^{d\times r}_{\geq 0}} f(\mathbf{W},\mathbf{H}_{t+1}) \qquad (NLS)$$

- How do we solve NMF?

$$\min_{\mathbf{W}\in\mathbb{R}^{d\times r}_{\geq 0},\mathbf{H}^{r\times n}_{\geq 0}} \left[f(\mathbf{W},\mathbf{H}) := \|\mathbf{X}-\mathbf{W}\mathbf{H}\|_F^2 \right]$$

  - Can't find both $\mathbf{W}$ and $\mathbf{H}$ at the same time, so alternate!

$$\mathbf{H}_{t+1} \leftarrow \operatorname*{argmin}_{\mathbf{H}\in\mathbb{R}^{r\times n}_{\geq 0}} f(\mathbf{W}_t,\mathbf{H}) \qquad (NLS)$$

$$\mathbf{W}_{t+1} \leftarrow \operatorname*{argmin}_{\mathbf{W}\in\mathbb{R}^{d\times r}_{\geq 0}} f(\mathbf{W},\mathbf{H}_{t+1}) \qquad (NLS)$$

  - Block Coordinate Descent for NMF (a.k.a. Alternating Least Squares)
  - NOT guaranteed to converge to global optimum (will come back to this point later)

$$\min_{\boldsymbol{\theta}=(\theta_1,\ldots,\theta_m)\in\Theta^{(1)}\times\cdots\times\Theta^{(m)}} f(\theta_1,\ldots,\theta_m)$$

$$\min_{\boldsymbol{\theta}=(\theta_1,\ldots,\theta_m)\in\Theta^{(1)}\times\cdots\times\Theta^{(m)}} f(\theta_1,\ldots,\theta_m)$$

- Block Coordinate Descent (BCD): For $n=1,\ldots,N$ and for $i=1,\ldots,m$:

$$\theta_n^{(i)} \in \operatorname*{argmin}_{\theta\in\Theta^{(i)}} f\left(\theta_n^{(1)},\cdots,\theta_n^{(i-1)},\theta,\theta_{n-1}^{(i+1)},\cdots,\theta_{n-1}^{(m)}\right).$$

$$\min_{\boldsymbol{\theta}=(\theta_1,\ldots,\theta_m)\in\Theta^{(1)}\times\cdots\times\Theta^{(m)}} f(\theta_1,\ldots,\theta_m)$$

- Block Coordinate Descent (BCD): For $n=1,\ldots,N$ and for $i=1,\ldots,m$:

$$\theta_n^{(i)} \in \operatorname*{argmin}_{\theta\in\Theta^{(i)}} f\left(\theta_n^{(1)},\cdots,\theta_n^{(i-1)},\theta,\theta_{n-1}^{(i+1)},\cdots,\theta_{n-1}^{(m)}\right).$$

  - Sequentially update each block coordinate (by PGD) while fixing the rest

$$\min_{\boldsymbol{\theta}=(\theta_1,\ldots,\theta_m)\in\Theta^{(1)}\times\cdots\times\Theta^{(m)}} f(\theta_1,\ldots,\theta_m)$$

- Block Coordinate Descent (BCD): For $n=1,\ldots,N$ and for $i=1,\ldots,m$:

$$\theta_n^{(i)} \in \operatorname*{argmin}_{\theta\in\Theta^{(i)}} f\left(\theta_n^{(1)},\cdots,\theta_n^{(i-1)},\theta,\theta_{n-1}^{(i+1)},\cdots,\theta_{n-1}^{(m)}\right).$$

  - Sequentially update each block coordinate (by PGD) while fixing the rest
  - For m=2 blocks, a.k.a. **alternating minimization/maximization**

$$\min_{\boldsymbol{\theta}=(\theta_1,\ldots,\theta_m)\in\Theta^{(1)}\times\cdots\times\Theta^{(m)}} f(\theta_1,\ldots,\theta_m)$$

- Block Coordinate Descent (BCD): For $n=1,\ldots,N$ and for $i=1,\ldots,m$:

$$\theta_n^{(i)} \in \operatorname*{argmin}_{\theta\in\Theta^{(i)}} f\left(\theta_n^{(1)},\cdots,\theta_n^{(i-1)},\theta,\theta_{n-1}^{(i+1)},\cdots,\theta_{n-1}^{(m)}\right).$$

  - Sequentially update each block coordinate (by PGD) while fixing the rest
  - For m=2 blocks, a.k.a. **alternating minimization/maximization**

- Block PGD: For $n=1,\ldots,N$ and for $i=1,\ldots,m$:

$$\theta_n^{(i)} \leftarrow \Pi_{\Theta^{(i)}}\left(\theta_{n-1}^{(i)} - \alpha_n^{(i)}\nabla_i f(\boldsymbol{\theta}_{n+\frac{i-1}{m}})\right)$$

Matrix Scaling, Optimal Transport, and Sinkhron Algorithms

- Matrix Scaling Problem:

  Given a matrix $\mathbf{A}$ and target row and column sums ($\mathbf{r}$, $\mathbf{c}$),

  find diagonal matrices $\mathbf{D_1}$ and $\mathbf{D_2}$ s.t. $\mathbf{D_1 A D_2}$ satisfies margin ($\mathbf{r}$, $\mathbf{c}$)

- Matrix Scaling Problem:

  Given a matrix **A** and target row and column sums (**r**, **c**),

  find diagonal matrices $\mathbf{D_1}$ and $\mathbf{D_2}$ s.t. $\mathbf{D_1 A D_2}$ satisfies margin (**r**, **c**)

  - Sinkhorn's matrix scaling algorithm (1964)
    - Normalize the rows of **A** to match the target row sum **r;** Obtain **A'**
    - Normalize the columns of **A'** to match the target column sum **c;** Obtain **A"**
    - Repeat

- Matrix Scaling Problem:

  Given a matrix **A** and target row and column sums (**r**, **c**),

  find diagonal matrices $\mathbf{D_1}$ and $\mathbf{D_2}$ s.t. $\mathbf{D_1 A D_2}$ satisfies margin (**r**, **c**)

  - Sinkhorn's matrix scaling algorithm (1964)
    - Normalize the rows of **A** to match the target row sum **r;** Obtain **A'**
    - Normalize the columns of **A'** to match the target column sum **c;** Obtain **A"**
    - Repeat
  - Sinkhorn's algorithm is known to solve the following relative entropy minimization problem:

$$\min_{\mathbf{x} \in \mathcal{T}(\mathbf{r},\mathbf{c})} \sum_{i,j} x_{ij} \log x_{ij}$$

Transportation polytope with margin (**r**, **c**)

- Matrix Scaling Problem:

  Given a matrix **A** and target row and column sums (**r**, **c**),

  find diagonal matrices $\mathbf{D_1}$ and $\mathbf{D_2}$ s.t. $\mathbf{D_1 A D_2}$ satisfies margin (**r**, **c**)

  - Sinkhorn's matrix scaling algorithm (1964)
    - Normalize the rows of **A** to match the target row sum **r;** Obtain **A'**
    - Normalize the columns of **A'** to match the target column sum **c;** Obtain **A"**
    - Repeat
  - Sinkhorn's algorithm is known to solve the following relative entropy minimization problem:

$$\min_{\mathbf{X} \in \mathcal{T}(\mathbf{r},\mathbf{c})} \sum_{i,j} x_{ij} \log x_{ij}$$

Transportation polytope with margin (**r**, **c**)

  - Why? It is in fact the **alternating maximization** on the "dual"!

- Entropic Optimal Transport

Given a margin ($\mathbf{r}$, $\mathbf{c}$) and cost matrix $\mathbf{C} = (c_{ij})$, c_{ij}=cost of moving a unit mass from location i to j, find the most efficient coupling $\boldsymbol{\pi}^*$:

$$\underset{\pi \in \mathcal{T}(\mathbf{r},\mathbf{c})}{\operatorname{argmin}} \sum_{ij} c_{ij}\pi_{ij} + \varepsilon D_{KL}(\pi \| \mathbf{r} \otimes \mathbf{c})$$

- Entropic Regularization (Cuturi, NeurIPS '13)

- Entropic Optimal Transport

Given a margin $(\mathbf{r}, \mathbf{c})$ and cost matrix $\mathbf{C} = (c_{ij})$, c_{ij}=cost of moving a unit mass from location i to j, find the most efficient coupling $\boldsymbol{\pi}^*$:

PGD??

$$\underset{\pi \in \mathcal{T}(\mathbf{r},\mathbf{c})}{\operatorname{argmin}} \sum_{ij} c_{ij}\pi_{ij} + \varepsilon D_{KL}(\pi \| \mathbf{r} \otimes \mathbf{c})$$

- Entropic Regularization (Cuturi, NeurIPS '13)

- Entropic Optimal Transport

Given a margin $(\mathbf{r}, \mathbf{c})$ and cost matrix $\mathbf{C} = (c_{ij})$, c_{ij}=cost of moving a unit mass from location i to j, find the most efficient coupling $\boldsymbol{\pi}^*$:

$$\underset{\pi \in \mathcal{T}(\mathbf{r},\mathbf{c})}{\operatorname{argmin}} \sum_{ij} c_{ij}\pi_{ij} + \varepsilon D_{KL}(\pi \| \mathbf{r} \otimes \mathbf{c})$$

- Entropic Regularization (Cuturi, NeurIPS '13)

$$\pi^* = \mathbf{W} \odot \exp(\boldsymbol{\alpha}^* \oplus \boldsymbol{\beta}^*) \quad \text{where} \quad \mathbf{W}_{ij} = e^{-c_{ij}/\varepsilon}\mathbf{r}_i\mathbf{c}_j$$

- Schrödinger potentials

Entropic Optimal Transport

Given a margin $(\mathbf{r}, \mathbf{c})$ and cost matrix $\mathbf{C} = (c_{ij})$, c_{ij}=cost of moving a unit mass from location i to j, find the most efficient coupling $\boldsymbol{\pi}^*$:

$$\underset{\pi \in \mathcal{T}(\mathbf{r},\mathbf{c})}{\operatorname{argmin}} \sum_{ij} c_{ij}\pi_{ij} + \varepsilon D_{KL}(\pi \| \mathbf{r} \otimes \mathbf{c})$$

- Entropic Regularization (Cuturi, NeurIPS '13)

$$\pi^* = \mathbf{W} \odot \exp(\boldsymbol{\alpha}^* \oplus \boldsymbol{\beta}^*) \quad \text{where} \quad \mathbf{W}_{ij} = e^{-c_{ij}/\varepsilon}\mathbf{r}_i\mathbf{c}_j$$

- Schrödinger potentials

- Sinkhorn Algorithm

$$\begin{cases} \forall 1 \le i \le n,\ \boldsymbol{\beta}_k(j) \leftarrow \log \dfrac{\mathbf{c}(j)}{\sum_{i=1}^m \mathbf{W}_{ij} \exp(\boldsymbol{\alpha}_{k-1}(i))}, \\ \forall 1 \le i \le m,\ \boldsymbol{\alpha}_k(i) \leftarrow \log \dfrac{\mathbf{r}(i)}{\sum_{j=1}^n \mathbf{W}_{ij} \exp(\boldsymbol{\beta}_k(j))}. \end{cases}$$

Entropic Optimal Transport

Given a margin $(\mathbf{r}, \mathbf{c})$ and cost matrix $\mathbf{C} = (c_{ij})$, c_{ij}=cost of moving a unit mass from location i to j, find the most efficient coupling $\boldsymbol{\pi}^*$:

$$\operatorname*{argmin}_{\pi \in \mathcal{T}(\mathbf{r},\mathbf{c})} \sum_{ij} c_{ij}\pi_{ij} + \varepsilon D_{KL}(\pi \| \mathbf{r} \otimes \mathbf{c})$$

- Entropic Regularization (Cuturi, NeurIPS '13)

$$\pi^* = \mathbf{W} \odot \exp(\boldsymbol{\alpha}^* \oplus \boldsymbol{\beta}^*) \quad \text{where} \quad \mathbf{W}_{ij} = e^{-c_{ij}/\varepsilon}\mathbf{r}_i\mathbf{c}_j$$

- Schrödinger potentials

- Sinkhorn Algorithm

$$\begin{cases} \forall 1 \le i \le n,\ \boldsymbol{\beta}_k(j) \leftarrow \log \dfrac{\mathbf{c}(j)}{\sum_{i=1}^m \mathbf{W}_{ij} \exp(\boldsymbol{\alpha}_{k-1}(i))}, \\ \forall 1 \le i \le m,\ \boldsymbol{\alpha}_k(i) \leftarrow \log \dfrac{\mathbf{r}(i)}{\sum_{j=1}^n \mathbf{W}_{ij} \exp(\boldsymbol{\beta}_k(j))}. \end{cases}$$

- Kantorovich dual

$$\sup_{\boldsymbol{\alpha},\boldsymbol{\beta}} \Big(\langle \mathbf{r}, \boldsymbol{\alpha} \rangle + \langle \mathbf{c}, \boldsymbol{\beta} \rangle - \langle \mathbf{W}, \exp(\boldsymbol{\alpha} \oplus \boldsymbol{\beta}) \rangle \Big)$$

Entropic Optimal Transport

Given a margin $(\mathbf{r}, \mathbf{c})$ and cost matrix $\mathbf{C} = (c_{ij})$, c_{ij}=cost of moving a unit mass from location i to j, find the most efficient coupling $\boldsymbol{\pi}^*$:

$$\operatorname*{argmin}_{\pi \in \mathcal{T}(\mathbf{r},\mathbf{c})} \sum_{ij} c_{ij}\pi_{ij} + \varepsilon D_{KL}(\pi \| \mathbf{r} \otimes \mathbf{c})$$

- Entropic Regularization (Cuturi, NeurIPS '13)

$$\pi^* = \mathbf{W} \odot \exp(\boldsymbol{\alpha}^* \oplus \boldsymbol{\beta}^*) \quad \text{where} \quad \mathbf{W}_{ij} = e^{-c_{ij}/\varepsilon}\mathbf{r}_i\mathbf{c}_j$$

- Schrödinger potentials

Alternating maximization

- Sinkhorn Algorithm

$$\begin{cases} \forall 1 \le i \le n,\ \boldsymbol{\beta}_k(j) \leftarrow \log \dfrac{\mathbf{c}(j)}{\sum_{i=1}^m \mathbf{W}_{ij} \exp(\boldsymbol{\alpha}_{k-1}(i))}, \\ \forall 1 \le i \le m,\ \boldsymbol{\alpha}_k(i) \leftarrow \log \dfrac{\mathbf{r}(i)}{\sum_{j=1}^n \mathbf{W}_{ij} \exp(\boldsymbol{\beta}_k(j))}. \end{cases}$$

- Kantorovich dual

$$\sup_{\boldsymbol{\alpha},\boldsymbol{\beta}} \Big(\langle \mathbf{r}, \boldsymbol{\alpha} \rangle + \langle \mathbf{c}, \boldsymbol{\beta} \rangle - \langle \mathbf{W}, \exp(\boldsymbol{\alpha} \oplus \boldsymbol{\beta}) \rangle \Big)$$

- Random matrix conditioned on row/column sums *(L., Mukherjee 24+)*

  X = RM w/ i.i.d. Exp(1) entries. Condition its row/column sums as (**r**, **c**).

- Random matrix conditioned on row/column sums *(L., Mukherjee 24+)*

  X = RM w/ i.i.d. Exp(1) entries. Condition its row/column sums be (**r**, **c**).

  Thm. $\mathbf{X} \approx \mathbf{Y} = (\mathbf{Y_{ij}})$, where $\mathbf{Y_{ij}} \sim \mathrm{Exp}\left(\frac{-1}{\alpha_i^* + \beta_j^*}\right)$ indep.

- Random matrix conditioned on row/column sums *(L., Mukherjee 24+)*

  X = RM w/ i.i.d. Exp(1) entries. Condition its row/column sums be (**r**, **c**).

  Thm. $\mathbf{X} \approx \mathbf{Y} = (\mathbf{Y_{ij}})$, where $\mathbf{Y_{ij}} \sim \mathrm{Exp}\left(\frac{-1}{\alpha_i^* + \beta_j^*}\right)$ indep.

  - Maximum Likelihood exponential tilting parameters

- Random matrix conditioned on row/column sums *(L., Mukherjee 24+)*

  X = RM w/ i.i.d. Exp(1) entries. Condition its row/column sums be (**r**, **c**).

  Thm. $\mathbf{X} \approx \mathbf{Y} = (\mathbf{Y}_{ij})$, where $\mathbf{Y}_{ij} \sim \mathrm{Exp}\left(\frac{-1}{\alpha_i^* + \beta_j^*}\right)$ indep.

  - Maximum Likelihood exponential tilting parameters

$$\sup_{\boldsymbol{\alpha},\boldsymbol{\beta}} \left(\langle \mathbf{r}, \boldsymbol{\alpha} \rangle + \langle \mathbf{c}, \boldsymbol{\beta} \rangle - \langle \mathbf{1}\mathbf{1}^\top, \psi(\boldsymbol{\alpha} \oplus \boldsymbol{\beta}) \rangle \right) \quad \text{where} \quad \psi(t) = -\log(-t)$$

  - Log-likelihood of observing margin (**r**, **c**)

- Random matrix conditioned on row/column sums *(L., Mukherjee 24+)*

  X = RM w/ i.i.d. Exp(1) entries. Condition its row/column sums be (**r**, **c**).

  Thm. $\mathbf{X} \approx \mathbf{Y} = (\mathbf{Y}_{ij})$, where $\mathbf{Y}_{ij} \sim \mathrm{Exp}\left(\frac{-1}{\alpha_i^* + \beta_j^*}\right)$ indep.

  - Maximum Likelihood exponential tilting parameters

$$\sup_{\boldsymbol{\alpha},\boldsymbol{\beta}} \left(\langle \mathbf{r}, \boldsymbol{\alpha} \rangle + \langle \mathbf{c}, \boldsymbol{\beta} \rangle - \langle \mathbf{1}\mathbf{1}^\top, \psi(\boldsymbol{\alpha} \oplus \boldsymbol{\beta}) \rangle \right) \quad \text{where} \quad \psi(t) = -\log(-t)$$

  - Log-likelihood of observing margin (**r**, **c**)

Alternating maximization

Generalized Sinkhorn

$$\begin{cases} \text{For } 1 \le j \le n,\ \boldsymbol{\beta}_k(j) \leftarrow \text{unique } \beta \in \mathbb{R} \text{ s.t. } \mathbf{c}(j) = \sum_{i=1}^m \psi'(\boldsymbol{\alpha}_{k-1}(i) + \beta), \\ \text{For } 1 \le i \le m,\ \boldsymbol{\alpha}_k(i) \leftarrow \text{unique } \alpha \in \mathbb{R} \text{ s.t. } \mathbf{r}(i) = \sum_{j=1}^n \psi'(\alpha + \boldsymbol{\beta}_k(j)). \end{cases}$$

- Random matrix conditioned on row/column sums *(L., Mukherjee 24+)*

X = RM w/ i.i.d. Exp(1) entries. Condition its row/column sums be (**r**, **c**).

Thm. $\mathbf{X} \approx \mathbf{Y} = (\mathbf{Y}_{ij})$, where $\mathbf{Y}_{ij} \sim \mathrm{Exp}\left(\frac{-1}{\alpha_i^* + \beta_j^*}\right)$ indep.

- Maximum Likelihood exponential tilting parameters

$$\sup_{\boldsymbol{\alpha},\boldsymbol{\beta}} \left(\langle \mathbf{r}, \boldsymbol{\alpha} \rangle + \langle \mathbf{c}, \boldsymbol{\beta} \rangle - \langle \mathbf{1}\mathbf{1}^\top, \psi(\boldsymbol{\alpha} \oplus \boldsymbol{\beta}) \rangle \right) \quad \text{where} \quad \psi(t) = -\log(-t)$$

- Log-likelihood of observing margin (**r**, **c**)

Alternating maximization

Thm. GS converges exponentially fast.

Generalized Sinkhorn

$$\begin{cases} \text{For } 1 \le j \le n,\ \boldsymbol{\beta}_k(j) \leftarrow \text{unique } \beta \in \mathbb{R} \text{ s.t. } \mathbf{c}(j) = \sum_{i=1}^m \psi'(\boldsymbol{\alpha}_{k-1}(i) + \beta), \\ \text{For } 1 \le i \le m,\ \boldsymbol{\alpha}_k(i) \leftarrow \text{unique } \alpha \in \mathbb{R} \text{ s.t. } \mathbf{r}(i) = \sum_{j=1}^n \psi'(\alpha + \boldsymbol{\beta}_k(j)). \end{cases}$$

Why does it work well in practice?

Stepsizes in PGD

(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n) \right)$$

- α_n: Stepsizes. How to choose them?

Stepsizes in PGD

(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n) \right)$$

- α_n: Stepsizes. How to choose them?
  - "*Small enough stepsize*": $\alpha_n \leq 1/L$, where
    - L = **Lipschitz constant for ∇f over Θ**
      ≈ **Largest absolute eigenvalue of $\nabla^2 f$ over Θ**

(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}}\left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n)\right)$$

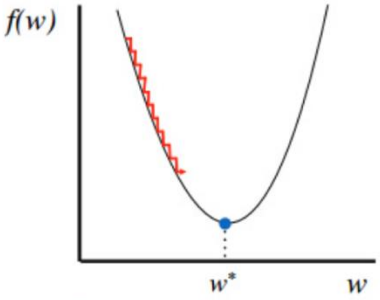
- α_n: Stepsizes. How to choose them?
  - "*Small enough stepsize*": $\alpha_n \leq 1/L$, where
    - L = **Lipschitz constant for ∇f over Θ**
      $\approx$ **Largest absolute eigenvalue of $\nabla^2 f$ over Θ**
  - But $\alpha_n = 1/L$ is TOO SMALL!
    - Could use "line search" to find larger α_n that works
    - L might be unknown and hard to estimate

very slowly even diverge

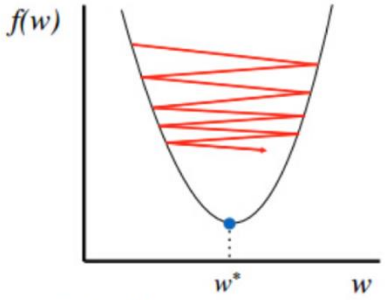
(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\Theta}\left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n)\right)$$

- α_n: Stepsizes. How to choose them?
  - "*Small enough stepsize*": $\alpha_n \leq 1/L$, where
    - L = **Lipschitz constant for ∇f over Θ**
      $\approx$ **Largest absolute eigenvalue of $\nabla^2 f$ over Θ**
  - But $\alpha_n = 1/L$ is TOO SMALL!
    - Could use "line search" to find larger α_n that works
    - L might be unknown and hard to estimate

In practice, performance of P(S)GD depends very sensitively on α_ns



Too small: converge very slowly



Too big: overshoot and even diverge

Two-Block structure : $\boldsymbol{\theta} = (A, B)$, $\boldsymbol{\Theta} = \Theta_A \times \Theta_B$

(Block PGD)
(or BCD)

$$A_{n+1} \leftarrow \Pi_{\Theta_A}\left(A_n - \alpha_n \nabla_A f(A_n, B_n)\right)$$
$$B_{n+1} \leftarrow \Pi_{\Theta_B}\left(B_n - \beta_n \nabla_B f(A_{n+1}, B_n)\right)$$

Block PGD = BCD

Two-Block structure : $\boldsymbol{\theta} = (A, B)$, $\boldsymbol{\Theta} = \Theta_A \times \Theta_B$

(Block PGD) (or BCD)

$$A_{n+1} \leftarrow \Pi_{\Theta_A}\left(A_n - \alpha_n \nabla_A f(A_n, B_n)\right)$$
$$B_{n+1} \leftarrow \Pi_{\Theta_B}\left(B_n - \beta_n \nabla_B f(A_{n+1}, B_n)\right)$$

- Known to be much more robust against stepsize choices than PGD (**Why??**)
  - e.g., low-rank matrix factorization, dictionary learning, tensor factorization, kernel learning, linear tranceiver design

Two-Block structure : $\boldsymbol{\theta} = (A, B)$, $\boldsymbol{\Theta} = \Theta_A \times \Theta_B$

(Block PGD) (or BCD)

$$A_{n+1} \leftarrow \Pi_{\Theta_A}\left(A_n - \alpha_n \nabla_A f(A_n, B_n)\right)$$
$$B_{n+1} \leftarrow \Pi_{\Theta_B}\left(B_n - \beta_n \nabla_B f(A_{n+1}, B_n)\right)$$

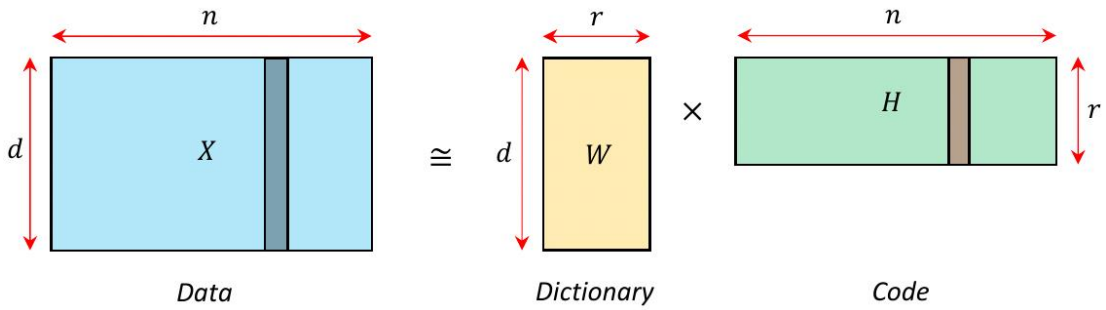
- Known to be much more robust against stepsize choices than PGD (**Why??**)
  - e.g., low-rank matrix factorization, dictionary learning, tensor factorization, kernel learning, linear tranceiver design

- [Large $L \Leftrightarrow \nabla f$ changes wildly]

  → **Sensitive dependence on stepsize**

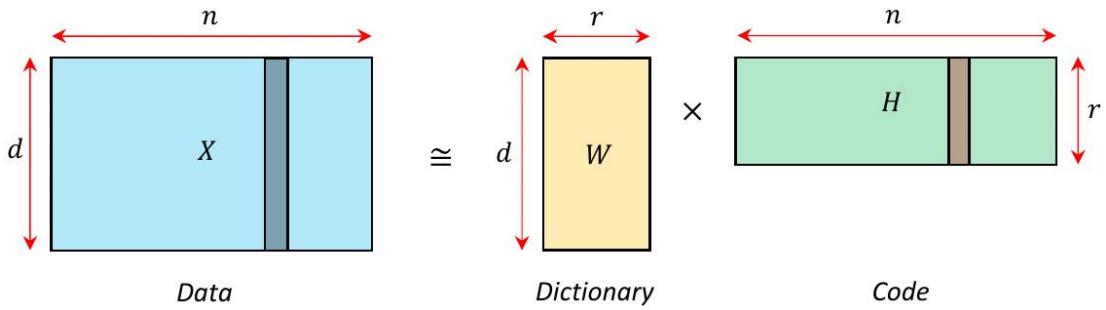
- **Exploiting block structure → The "effective L" is reduced**

Motivating example : **Nonnegative Matrix Factorization (Lee & Seung, Nature '99)**



$$\begin{cases} \text{minimize} & f(W,H) = \|X - WH\|_F^2 & \text{(Reconstruction error)} \\ \text{subject to} & W \in \mathbb{R}^{p\times r}_{\geq 0},\ H \in \mathbb{R}^{r\times n}_{\geq 0} & (\textit{Constraints}) \end{cases}$$

Motivating example : **Nonnegative Matrix Factorization (Lee & Seung, Nature '99)**



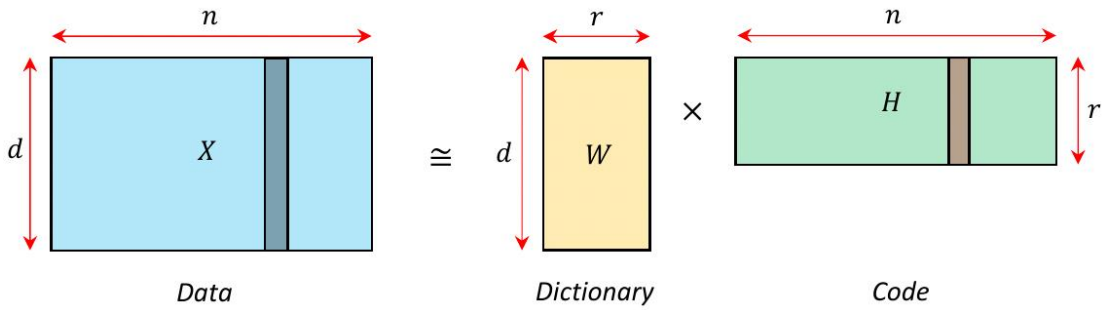
$$\begin{cases} \text{minimize} & f(W,H) = \|X - WH\|_F^2 & \text{(Reconstruction error)} \\ \text{subject to} & W \in \mathbb{R}^{p\times r}_{\geq 0},\ H \in \mathbb{R}^{r\times n}_{\geq 0} & \text{(Constraints)} \end{cases}$$

PGD $\begin{cases} (W_{n+1},H_{n+1}) \leftarrow (W_n,H_n) - \alpha_n(\nabla_W f(W_n,H_n), \nabla_H f(W_n,H_n)) \\ (W_{n+1},H_{n+1}) \leftarrow \max\big(0,(W_{n+1},H_{n+1})\big) \end{cases}$

(Almost no one uses this ☺)

- $f(W,H)$ = Bi-convex
- $\nabla_W f(W,H) = (WH - X)H^T$
- $\nabla_H f(W,H) = W^T(WH - X)$

Motivating example : **Nonnegative Matrix Factorization (Lee & Seung, Nature '99)**



$$\begin{cases} \text{minimize} & f(W,H) = \|X - WH\|_F^2 & \text{(Reconstruction error)} \\ \text{subject to} & W \in \mathbb{R}^{p\times r}_{\geq 0},\ H \in \mathbb{R}^{r\times n}_{\geq 0} & \text{(Constraints)} \end{cases}$$

PGD $\begin{cases} (W_{n+1},H_{n+1}) \leftarrow (W_n,H_n) - \alpha_n(\nabla_W f(W_n,H_n), \nabla_H f(W_n,H_n)) \\ (W_{n+1},H_{n+1}) \leftarrow \max\big(0,(W_{n+1},H_{n+1})\big) \end{cases}$

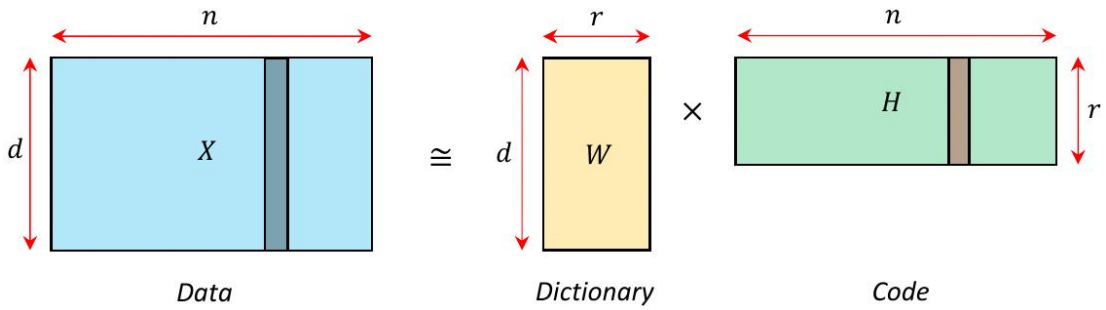
(Almost no one uses this ☺)

- $f(W,H)$ = Bi-convex
- $\nabla_W f(W,H) = (WH - X)H^T$
- $\nabla_H f(W,H) = W^T(WH - X)$

- $\nabla^2 f = \begin{matrix} \\ \mathrm{vec}(\mathbf{W}) \\ \mathrm{vec}(\mathbf{H}) \end{matrix} \begin{matrix} \begin{matrix} \mathrm{vec}(\mathbf{W})^T & \mathrm{vec}(\mathbf{H})^T \end{matrix} \\ \begin{bmatrix} \mathbf{H}\mathbf{H}^T \otimes \mathbf{I}_p & A_{12} \\ A_{12}^T & \mathbf{I}_n \otimes \mathbf{W}^T\mathbf{W} \end{bmatrix} \end{matrix}$

$$A_{12} = [(\mathbf{H}\otimes\mathbf{W}) + \mathbf{I}_r \otimes (\mathbf{W}\mathbf{H} - \mathbf{X})]\,\mathbf{C}^{(r,n)}$$

Motivating example : **Nonnegative Matrix Factorization (Lee & Seung, Nature '99)**



$$\begin{cases} \text{minimize} & f(W,H) = \|X - WH\|_F^2 \\ \text{subject to} & W \in \mathbb{R}^{p\times r}_{\geq 0},\ H \in \mathbb{R}^{r\times n}_{\geq 0} \end{cases} \qquad \begin{matrix} \text{(Reconstruction error)} \\ \text{(Constraints)} \end{matrix}$$

PGD $\begin{cases} (W_{n+1},H_{n+1}) \leftarrow (W_n,H_n) - \alpha_n(\nabla_W f(W_n,H_n), \nabla_H f(W_n,H_n)) \\ (W_{n+1},H_{n+1}) \leftarrow \max\big(0,(W_{n+1},H_{n+1})\big) \end{cases}$

(Almost no one uses this ☺)

- $f(W,H)$ = Bi-convex
- $\nabla_W f(W,H) = (WH - X)H^T$
- $\nabla_H f(W,H) = W^T(WH - X)$

- $\nabla^2 f = \begin{matrix} \\ \text{vec}(\mathbf{W}) \\ \text{vec}(\mathbf{H}) \end{matrix} \begin{matrix} \begin{matrix} \text{vec}(\mathbf{W})^T & \text{vec}(\mathbf{H})^T \end{matrix} \\ \begin{bmatrix} \mathbf{H}\mathbf{H}^T \otimes \mathbf{I}_p & A_{12} \\ A_{12}^T & \mathbf{I}_n \otimes \mathbf{W}^T\mathbf{W} \end{bmatrix} \end{matrix}$

$$A_{12} = [(\mathbf{H} \otimes \mathbf{W}) + \mathbf{I}_r \otimes (\mathbf{W}\mathbf{H} - \mathbf{X})]\,\mathbf{C}^{(r,n)}$$

L = Max eval over all (W,H)

= *Unbounded!*

In **Supervised Matrix Factorization** ***(Lee, L., Yao, ICML '24)***

Adaptive BCD:

$$\mathbf{W} \leftarrow \Pi\left(\mathbf{W} - \frac{1}{\lambda_{\max}(\mathbf{H}\mathbf{H}^T) + \varepsilon}\,(\mathbf{W}\mathbf{H} - \mathbf{X})\mathbf{H}^T\right),$$

$$\mathbf{H} \leftarrow \Pi'\left(\mathbf{H} - \frac{1}{\lambda_{\max}(\mathbf{W}\mathbf{W}^T) + \varepsilon}\,\mathbf{W}^T(\mathbf{W}\mathbf{H} - \mathbf{X})\right)$$

In **Supervised Matrix Factorization** ***(Lee, L., Yao, ICML '24)***

Adaptive BCD:

$$\mathbf{W} \leftarrow \Pi\left(\mathbf{W} - \frac{1}{\lambda_{\max}(\mathbf{H}\mathbf{H}^T) + \varepsilon}(\mathbf{W}\mathbf{H} - \mathbf{X})\mathbf{H}^T\right),$$
$$\mathbf{H} \leftarrow \Pi'\left(\mathbf{H} - \frac{1}{\lambda_{\max}(\mathbf{W}\mathbf{W}^T) + \varepsilon}\mathbf{W}^T(\mathbf{W}\mathbf{H} - \mathbf{X})\right)$$

- $$\nabla^2 f = \begin{matrix} \\ \mathrm{vec}(\mathbf{W}) \\ \mathrm{vec}(\mathbf{H}) \end{matrix} \begin{matrix} \mathrm{vec}(\mathbf{W})^T & \mathrm{vec}(\mathbf{H})^T \end{matrix} \begin{bmatrix} \mathbf{H}\mathbf{H}^T \otimes \mathbf{I}_p & A_{12} \\ A_{12}^T & \mathbf{I}_n \otimes \mathbf{W}^T\mathbf{W} \end{bmatrix}$$

$\nabla^2_W f(\cdot, \mathbf{H})$ $\quad$ $\nabla^2_H f(\mathbf{W}, \cdot)$

In **Supervised Matrix Factorization** ***(Lee, L., Yao, ICML '24)***

Adaptive BCD:

$$\mathbf{W} \leftarrow \Pi\left(\mathbf{W} - \frac{1}{\lambda_{\max}(\mathbf{H}\mathbf{H}^T) + \varepsilon}(\mathbf{W}\mathbf{H} - \mathbf{X})\mathbf{H}^T\right),$$
$$\mathbf{H} \leftarrow \Pi'\left(\mathbf{H} - \frac{1}{\lambda_{\max}(\mathbf{W}\mathbf{W}^T) + \varepsilon}\mathbf{W}^T(\mathbf{W}\mathbf{H} - \mathbf{X})\right)$$

- $$\nabla^2 f = \begin{matrix} \\ \mathrm{vec}(\mathbf{W}) \\ \mathrm{vec}(\mathbf{H}) \end{matrix} \begin{matrix} \mathrm{vec}(\mathbf{W})^T & \mathrm{vec}(\mathbf{H})^T \end{matrix} \begin{bmatrix} \mathbf{H}\mathbf{H}^T \otimes \mathbf{I}_p & A_{12} \\ A_{12}^T & \mathbf{I}_n \otimes \mathbf{W}^T\mathbf{W} \end{bmatrix}$$

$\nabla^2_W f(\cdot, \mathbf{H})$ $\qquad$ $\nabla^2_H f(\mathbf{W}, \cdot)$

Largest Eval of *diagonal blocks* of the Hessian

$\ll$ Largest Eval of the entire Hessian

How does it work?

General principle for Cyclic Block Optimization?

Frank-Wolfe (1956)

- **Taylor expand the objective f near θ_n:**

$$f(\boldsymbol{\theta}) \approx f(\boldsymbol{\theta}_n) + \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle$$

- **Minimize the 1st-order Taylor approximation**

$$\boldsymbol{\theta}'_n = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle$$

$$\boldsymbol{\theta}_{n+1} = \text{Convex combination of } \boldsymbol{\theta}_n \text{ and } \boldsymbol{\theta}'_n$$

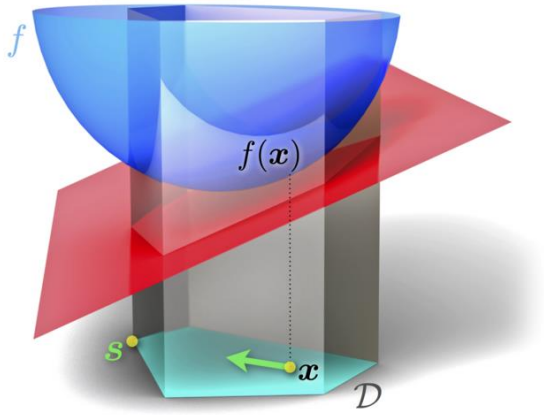


Image credit: Wikipedia

Newton's Method (1690)

(Assume $\Theta = \mathbb{R}^p$)

- **Taylor expand the objective f near θ_n:**

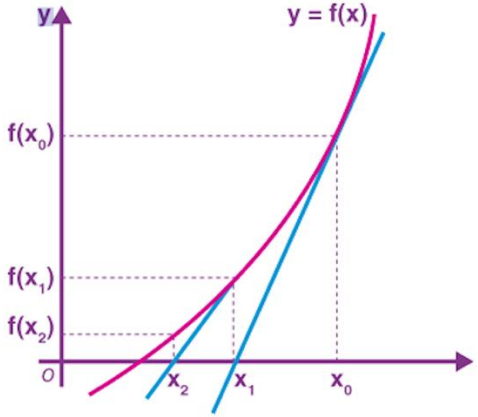
$$f(\boldsymbol{\theta}) \approx f(\boldsymbol{\theta}_n) + \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{1}{2} \left\langle \boldsymbol{\theta} - \boldsymbol{\theta}_n, \nabla f^2(\boldsymbol{\theta}_n)(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \right\rangle$$

- **Minimize the 2nd-order Taylor expansion**

$$\begin{aligned} \boldsymbol{\theta}_{n+1} &= \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^p} f(\boldsymbol{\theta}_n) + \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{1}{2} \left\langle \boldsymbol{\theta} - \boldsymbol{\theta}_n, \nabla f^2(\boldsymbol{\theta}_n)(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \right\rangle \\ &= \boldsymbol{\theta}_n - \left(\nabla^2 f(\boldsymbol{\theta}_n) \right)^{-1} \nabla f(\boldsymbol{\theta}_n) \end{aligned}$$

- Undefined if the Hessian is not PD
- Inverting the Hessian is expensive

Super-fast, quadratic convergence if works ☺



A Quasi-Newton Method

(Assume $\Theta = \mathbb{R}^p$)

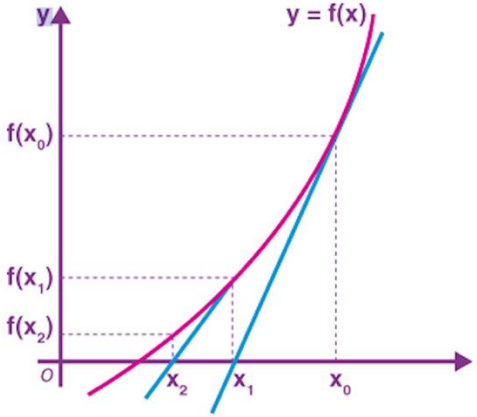
- **Taylor expand the objective f near θ_n:**

$$f(\boldsymbol{\theta}) \approx f(\boldsymbol{\theta}_n) + \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{1}{2} \left\langle \boldsymbol{\theta} - \boldsymbol{\theta}_n, \nabla f^2(\boldsymbol{\theta}_n)(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \right\rangle$$

- **Minimize the 2nd-order Taylor expansion with regularization**

$$\begin{aligned} \boldsymbol{\theta}_{n+1} &= \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{1}{2} \left\langle \boldsymbol{\theta} - \boldsymbol{\theta}_n, \nabla f^2(\boldsymbol{\theta}_n)(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \right\rangle + \frac{\gamma_n}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_n\|^2 \\ &= \boldsymbol{\theta}_n - \left(\nabla^2 f(\boldsymbol{\theta}_n) + \gamma_n \mathbf{I}_p \right)^{-1} \nabla f(\boldsymbol{\theta}_n) \end{aligned}$$

- Levenberg-Marquardt regularization ('44 '63)
- Can make the regularized Hessian PD
- Matrix inversion still expensive



Trust-Region (1970's)

- **Taylor expand the objective f near θ_n:**

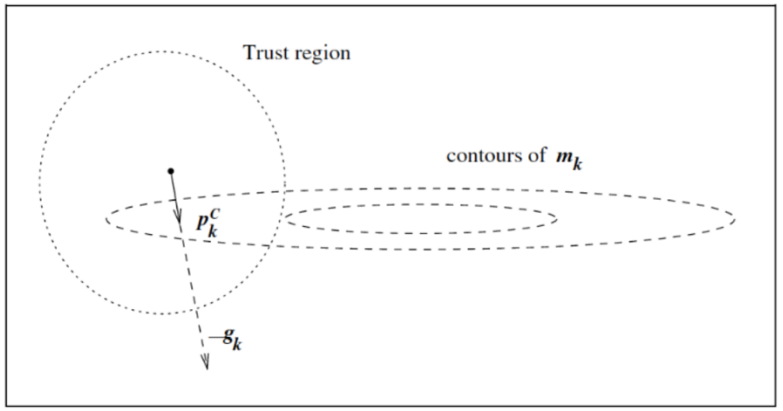
$$f(\boldsymbol{\theta}) \approx f(\boldsymbol{\theta}_n) + \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{1}{2} \langle \boldsymbol{\theta} - \boldsymbol{\theta}_n, \nabla f^2(\boldsymbol{\theta}_n)(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \rangle$$

- **Minimize a "quadratic model" within a trust-region**

$$\boldsymbol{\theta}_{n+1} = \underset{\boldsymbol{\theta} \in \boldsymbol{\Theta},\, \|\boldsymbol{\theta} - \boldsymbol{\theta}_n\| \le r_n}{\arg\min} \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{1}{2} \langle \boldsymbol{\theta} - \boldsymbol{\theta}_n, \mathbf{B}_n(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \rangle$$

- Trust-region with radius r_n
- Next radius r_{n+1} is computed adaptively based on the performance of the previous update

- PD, and no need to be close to the Hessian



PGD (Late 1950's)

- **Taylor expand the objective f near θ_n:**

$$f(\boldsymbol{\theta}) \approx f(\boldsymbol{\theta}_n) + \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{1}{2} \langle \boldsymbol{\theta} - \boldsymbol{\theta}_n, \nabla f^2(\boldsymbol{\theta}_n)(\boldsymbol{\theta} - \boldsymbol{\theta}_n) \rangle$$

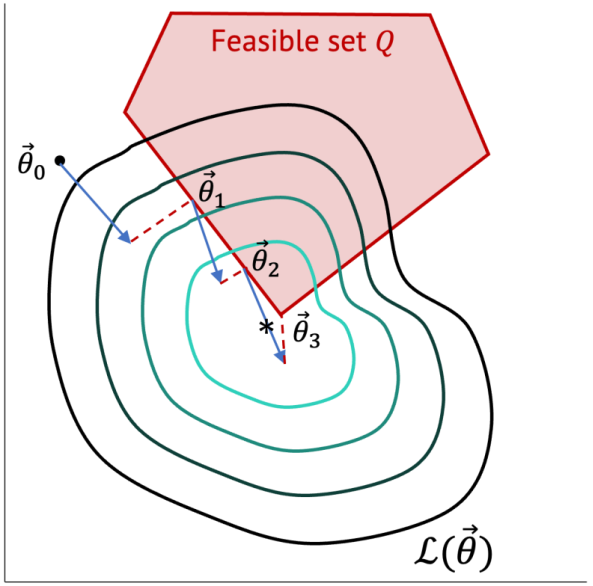
- **Replace the Hessian with L times the identity**

L = largest absolute eigenvalue of $\nabla^2 f(\boldsymbol{\theta})$ over all $\boldsymbol{\theta}$

$$f(\boldsymbol{\theta}) \le f(\boldsymbol{\theta}_n) + \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{L}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_n\|^2$$

- **Minimize the quadratic surrogate**

$$\begin{aligned} \boldsymbol{\theta}_{n+1} &= \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \ \langle \nabla f(\boldsymbol{\theta}_n), \boldsymbol{\theta} - \boldsymbol{\theta}_n \rangle + \frac{L}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_n\|^2 \\ &= \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \ \left\| \boldsymbol{\theta} - \left(\boldsymbol{\theta}_n - \frac{1}{L} \nabla f(\boldsymbol{\theta}_n) \right) \right\|^2 \\ &= \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \frac{1}{L} \nabla f(\boldsymbol{\theta}_n) \right) \end{aligned}$$

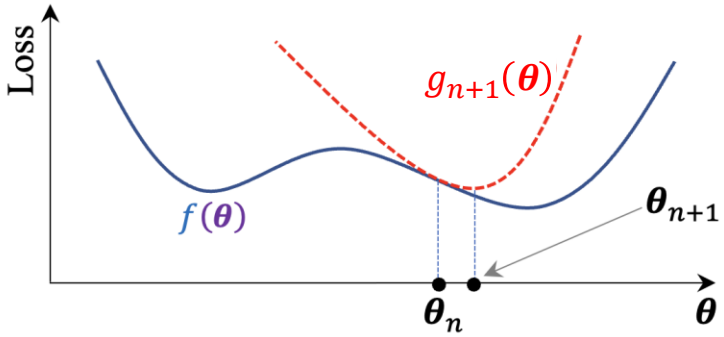


Majorization-Minimization (MM) (1970's, originally EM in statistics)

- **Find a majorizing surrogate g_{n+1} of f at θ_n**

$$f(\boldsymbol{\theta}) \le g_{n+1}(\boldsymbol{\theta}), \qquad f(\boldsymbol{\theta}_n) = g_{n+1}(\boldsymbol{\theta}_n)$$

nonconvex convex tangent



- **Minimize the surrogate**

$$\boldsymbol{\theta}_{n+1} = \operatorname*{argmin}_{\boldsymbol{\theta} \in \Theta} g_{n+1}(\boldsymbol{\theta})$$

- Nonconvex minimization
  ← A seq. of convex minimizations

- Sub-problems often admit close-form solutions
  - e.g., PGD, Multiplicative Update for NMF, Poisson regression
- Otherwise use convex solver for each step
  - e.g., PGD, Newton

Block MM (a.k.a. BSUM by Hong et al. 2015)

$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta}=[\theta^{(1)},\dots,\theta^{(m)}]\in\boldsymbol{\Theta}}{\arg\min} \; (F(\boldsymbol{\theta}) := f(\boldsymbol{\theta}) + p(\boldsymbol{\theta}))$$

Convex, nonsmooth

Nonconvex, smooth

$$\textbf{BMM} \quad \begin{cases} g_n^{(i)} \leftarrow \left[\begin{array}{c} \text{Majorizing surrogate of} \\ \theta \mapsto f_n^{(i)}(\theta) := f\left(\theta_n^{(1)}, \cdots, \theta_n^{(i-1)}, \theta, \theta_{n-1}^{(i+1)}, \cdots, \theta_{n-1}^{(m)}\right) \end{array} \right] \\ p_n^{(i)}(\theta) := p\left(\theta_n^{(1)}, \cdots, \theta_n^{(i-1)}, \theta, \theta_{n-1}^{(i+1)}, \cdots, \theta_{n-1}^{(m)}\right) \\ \theta_n^{(i)} \in \arg\min_{\theta\in\Theta^{(i)}} \left(G_n^{(i)}(\theta) := g_n^{(i)}(\theta) + p_n^{(i)}(\theta)\right). \end{cases}$$

- BCD, Block PGD, Sinkhorn, Alt. Min, Block EM..

Only majorize the smooth part

What do we know about them?

Convergence guarantees?

- For general nonconvex optimization, look for **first-order** guarantees
  - Iteration complexity = # of iterations to reach an ϵ-stationary points?

    " $\|\nabla f(\theta_n)\| \leq \epsilon$ "
  - Asymptotic stationarity = convergence to stationary points?

Methods	Objective	Block update	Complexity	Asymp. conv.	Inexact computation
BPGD [39]	C & NS	cyclic	Depends on KL-ineq.	✓	✗
BPGD [4]	C & S	cyclic	$\widetilde{O}(\varepsilon^{-1})$	✗	✗
BCD-PR [21]	NC & S	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM [34]	NC & NS	random	$O_{\mathbb{E}}((1+\rho^{-1}L_g^2+\rho^{-1})\varepsilon^{-2})$	✓	✗
BMM [34]	NC & NS	cyclic	✗	✓	✗

  - Use ρ-strongly convex, L_g-smooth surrogates
  - C = Convex, NC = Nonconvex, S=Smooth, NS=Non-smooth

- For general nonconvex optimization, look for **first-order** guarantees
  - Iteration complexity = # of iterations to reach an ϵ-stationary points?

    " $\|\nabla f(\theta_n)\| \le \epsilon$ "
  - Asymptotic stationarity = convergence to stationary points?

Methods	Objective	Block update	Complexity	Asymp. conv.	Inexact computation
BPGD [39]	C & NS	cyclic	Depends on KL-ineq.	✓	✗
BPGD [4]	C & S	cyclic	$\widetilde{O}(\varepsilon^{-1})$	✗	✗
BCD-PR [21]	NC & S	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM [34]	NC & NS	random	$O_{\mathbb{E}}((1+\rho^{-1}L_g^2+\rho^{-1})\varepsilon^{-2})$	✓	✗
BMM [34]	NC & NS	cyclic	✗	✓	✗
BMM [L., Li, SIOPT 25]	NC & NS	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓

- BMM could be slow with flat surrogates

- For general nonconvex optimization, look for **first-order** guarantees
  - Iteration complexity = # of iterations to reach an ϵ-stationary points?

    " $\|\nabla f(\theta_n)\| \le \epsilon$ "
  - Asymptotic stationarity = convergence to stationary points?

Methods	Objective	Block update	Complexity	Asymp. conv.	Inexact computation
BPGD [39]	C & NS	cyclic	Depends on KL-ineq.	✓	✗
BPGD [4]	C & S	cyclic	$\widetilde{O}(\varepsilon^{-1})$	✗	✗
BCD-PR [21]	NC & S	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM [34]	NC & NS	random	$O_{\mathbb{E}}((1+\rho^{-1}L_g^2+\rho^{-1})\varepsilon^{-2})$	✓	✗
BMM [34]	NC & NS	cyclic	✗	✓	✗
BMM	NC & NS	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM + Trust-Region	NC & NS	cyclic	$\widetilde{O}((1+L_g)\varepsilon^{-2})$	✓	✓

[L., Li, SIOPT 25]

- Still fast with flat surrogates

- For general nonconvex optimization, look for **first-order** guarantees
  - Iteration complexity = # of iterations to reach an ϵ-stationary points?

    " $\|\nabla f(\theta_n)\| \le \epsilon$ "
  - Asymptotic stationarity = convergence to stationary points?

Methods	Objective	Block update	Complexity	Asymp. conv.	Inexact computation
BPGD [39]	C & NS	cyclic	Depends on KL-ineq.	✓	✗
BPGD [4]	C & S	cyclic	$\widetilde{O}(\varepsilon^{-1})$	✗	✗
BCD-PR [21]	NC & S	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM [34]	NC & NS	random	$O_{\mathbb{E}}((1+\rho^{-1}L_g^2+\rho^{-1})\varepsilon^{-2})$	✓	✗
BMM [34]	NC & NS	cyclic	✗	✓	✗
BMM	NC & NS	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM + Trust-Region	NC & NS	cyclic	$\widetilde{O}((1+L_g)\varepsilon^{-2})$	✓	✓

[L., Li, SIOPT 25]

Quite technical

- For general nonconvex optimization, look for **first-order** guarantees
  - Iteration complexity = # of iterations to reach an ϵ-stationary points?

" $\|\nabla f(\theta_n)\| \le \epsilon$ "

  - Asymptotic stationarity = convergence to stationary points?

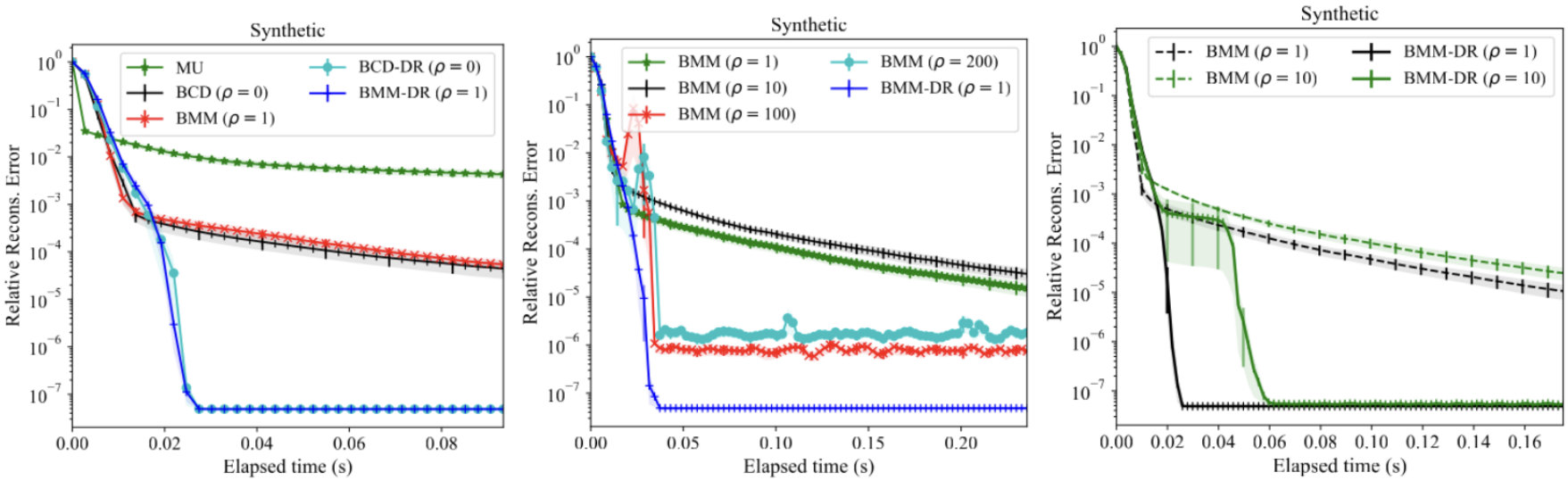
Methods	Objective	Block update	Complexity	Asymp. conv.	Inexact computation
BPGD [39]	C & NS	cyclic	Depends on KL-ineq.	✓	✗
BPGD [4]	C & S	cyclic	$\widetilde{O}(\varepsilon^{-1})$	✗	✗
BCD-PR [21]	NC & S	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM [34]	NC & NS	random	$O_{\mathbb{E}}((1+\rho^{-1}L_g^2+\rho^{-1})\varepsilon^{-2})$	✓	✗
BMM [34]	NC & NS	cyclic	✗	✓	✗
BMM	NC & NS	cyclic	$\widetilde{O}((1+L_g+\rho^{-1})\varepsilon^{-2})$	✓	✓
BMM + Trust-Region	NC & NS	cyclic	$\widetilde{O}((1+L_g)\varepsilon^{-2})$	✓	✓

[L., Li, SIOPT 25]

Half of the work

= Defining the right notion of approximate stationarity

A very flat nonconvex problem (NMF)



MU=Multiplicative Update

BCD = Block Coordianate Descent

BMM= Block Majorization-Minimization (in this case BCD+prox. Reg.)

A few words about measures of stationarity..

- $\boldsymbol{\theta}^* \in \text{Interior}(\boldsymbol{\Theta})$ is stationary for $F = f + p$ iff $\|\nabla f(\boldsymbol{\theta}^*) + \partial p(\boldsymbol{\theta}^*)\| = 0$

A few words about measures of stationarity..

- $\boldsymbol{\theta}^* \in \text{Interior}(\boldsymbol{\Theta})$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\|\nabla f(\boldsymbol{\theta}^*) + \partial p(\boldsymbol{\theta}^*)\| \leq \epsilon$

A few words about measures of stationarity..

- $\boldsymbol{\theta}^* \in \text{Interior}(\boldsymbol{\Theta})$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\|\nabla f(\boldsymbol{\theta}^*) + \partial p(\boldsymbol{\theta}^*)\| \leq \epsilon$
- $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is stationary for $\mathrm{F} = f + p$ iff

$$\sup_{\boldsymbol{\theta} \in \boldsymbol{\Theta},\, \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| \leq 1} \langle -\nabla f(\boldsymbol{\theta}^*) - \partial p(\boldsymbol{\theta}^*),\, \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \leq 0,$$

$$\text{iff} \qquad \text{dist}(\mathbf{0}, \partial F(\boldsymbol{\theta}^*) + \mathcal{N}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}^*)) \leq 0$$

A few words about measures of stationarity..

- $\boldsymbol{\theta}^* \in \text{Interior}(\boldsymbol{\Theta})$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\|\nabla f(\boldsymbol{\theta}^*) + \partial p(\boldsymbol{\theta}^*)\| \leq \epsilon$
- $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\quad \text{dist}(\mathbf{0}, \partial F(\boldsymbol{\theta}^*) + \mathcal{N}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}^*)) \leq \epsilon$??

A few words about measures of stationarity..

- $\boldsymbol{\theta}^* \in \text{Interior}(\boldsymbol{\Theta})$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\|\nabla f(\boldsymbol{\theta}^*) + \partial p(\boldsymbol{\theta}^*)\| \leq \epsilon$
- $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\quad \text{dist}(\mathbf{0}, \partial F(\boldsymbol{\theta}^*) + \mathcal{N}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}^*)) \leq \epsilon$??

  - Davis & Drusvyatskiy in SIOPT 2019

    "This is a highly discontinuous measure and hard to work with"

A few words about measures of stationarity..

- $\boldsymbol{\theta}^* \in \text{Interior}(\boldsymbol{\Theta})$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\|\nabla f(\boldsymbol{\theta}^*) + \partial p(\boldsymbol{\theta}^*)\| \leq \epsilon$
- $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\quad \text{dist}(\mathbf{0}, \partial F(\boldsymbol{\theta}^*) + \mathcal{N}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}^*)) \leq \epsilon$??

  - Davis & Drusvyatskiy in SIOPT 2019

    "This is a highly discontinuous measure and hard to work with"

- **Our proposal:** $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is ϵ-stationary for $\mathrm{F} = f + p$ iff

$$\sup_{\boldsymbol{\theta} \in \boldsymbol{\Theta},\, \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| \leq 1} \left[V(\boldsymbol{\theta}^*, \boldsymbol{\theta}) := \langle -\nabla f(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle + p(\boldsymbol{\theta}^*) - p(\boldsymbol{\theta}) \right] \leq \epsilon$$

"p is non-smooth, so use objective value gap instead of sub-gradient"

A few words about measures of stationarity..

- $\boldsymbol{\theta}^* \in \text{Interior}(\boldsymbol{\Theta})$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\|\nabla f(\boldsymbol{\theta}^*) + \partial p(\boldsymbol{\theta}^*)\| \leq \epsilon$
- $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is ϵ-stationary for $\mathrm{F} = f + p$ iff $\quad \text{dist}(\mathbf{0}, \partial F(\boldsymbol{\theta}^*) + \mathcal{N}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}^*)) \leq \epsilon$??

  - Davis & Drusvyatskiy in SIOPT 2019

    "This is a highly discontinuous measure and hard to work with"

- **Our proposal:** $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is ϵ-stationary for $\mathrm{F} = f + p$ iff

$$\sup_{\boldsymbol{\theta} \in \boldsymbol{\Theta},\, \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| \leq 1} \left[V(\boldsymbol{\theta}^*, \boldsymbol{\theta}) := \langle -\nabla f(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle + p(\boldsymbol{\theta}^*) - p(\boldsymbol{\theta}) \right] \leq \epsilon$$

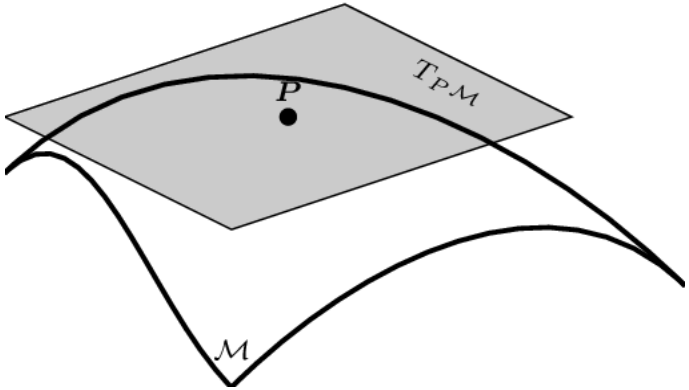
"p is non-smooth, so use objective value gap instead of sub-gradient"

  - $\epsilon = 0$ iff stationary point
  - If p is continuous (but still non-smooth), V is continuous

A geometric approach:

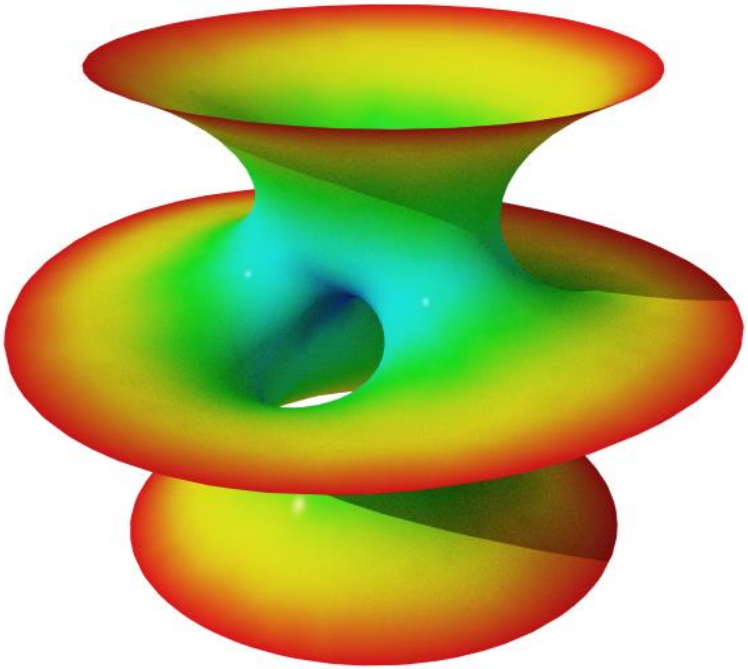
Riemannian Optimization

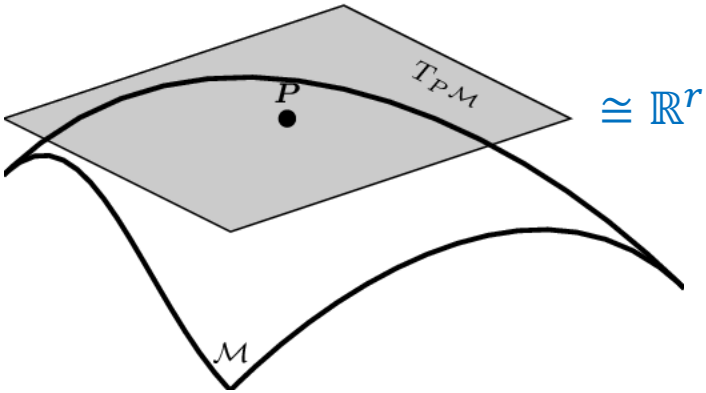
Tangent planes!



- Parameter spaces often has some intrinsic, ***low-dimensional geometric structures***
- **(r-dim) Riemannian manifold**

  = Smooth surface in $\mathbb{R}^p$ where every point has r-dim tangent spaces

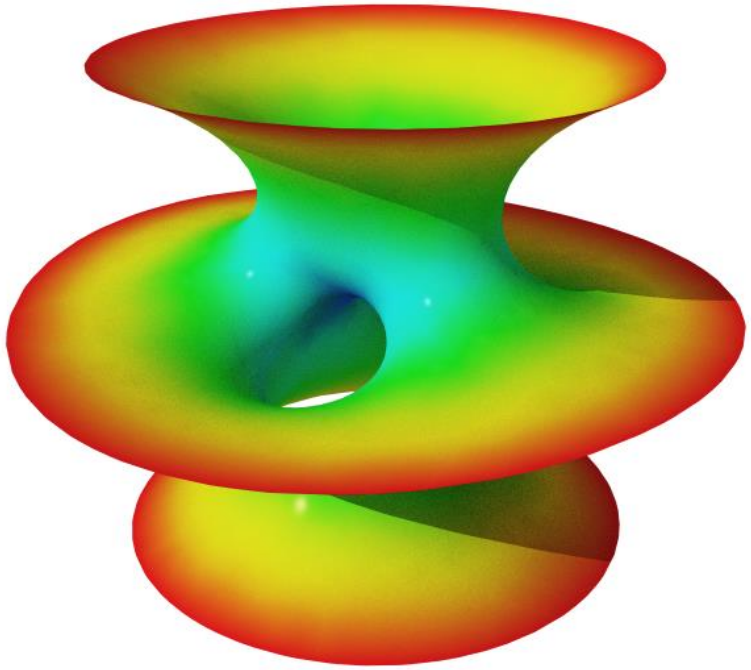




e.g.,
- Low-rank matrix manifolds
- Orthogonal frames
- PSD matrices
- DNNs with spectrum-constrained matrix weights

- Parameter spaces often has some intrinsic, ***low-dimensional geometric structures***
- **(r-dim) Riemannian manifold**

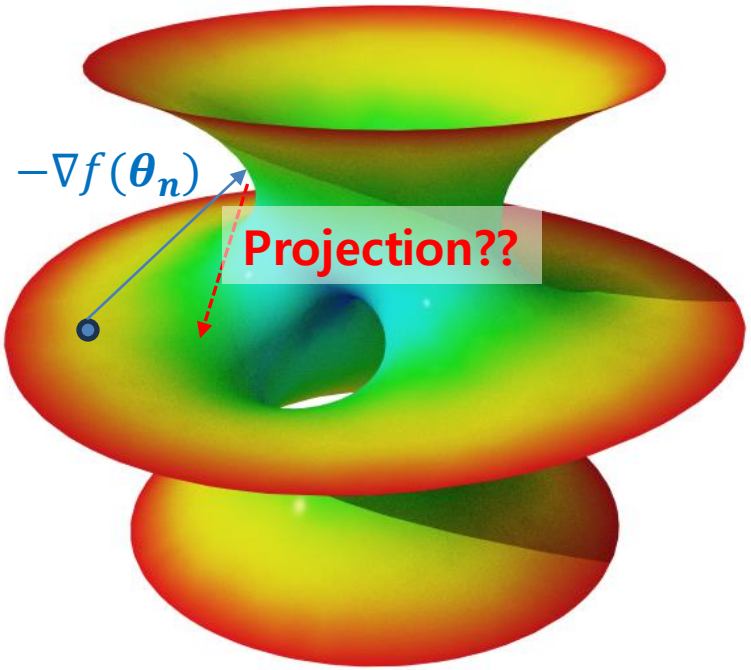
  = Smooth surface in $\mathbb{R}^p$ where every point has r-dim tangent spaces



Riemann
(1826-1866)

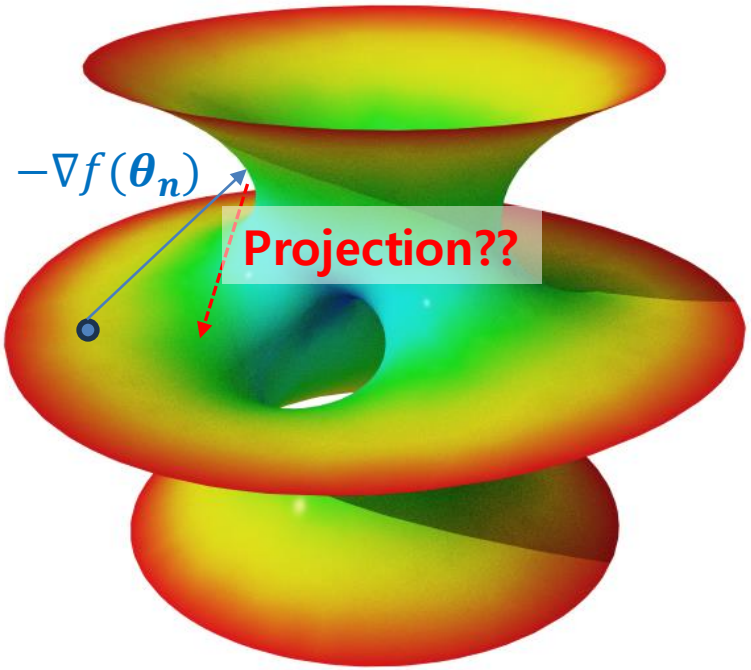
Gauss
(1777-1855)

(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}} \left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n) \right)$$



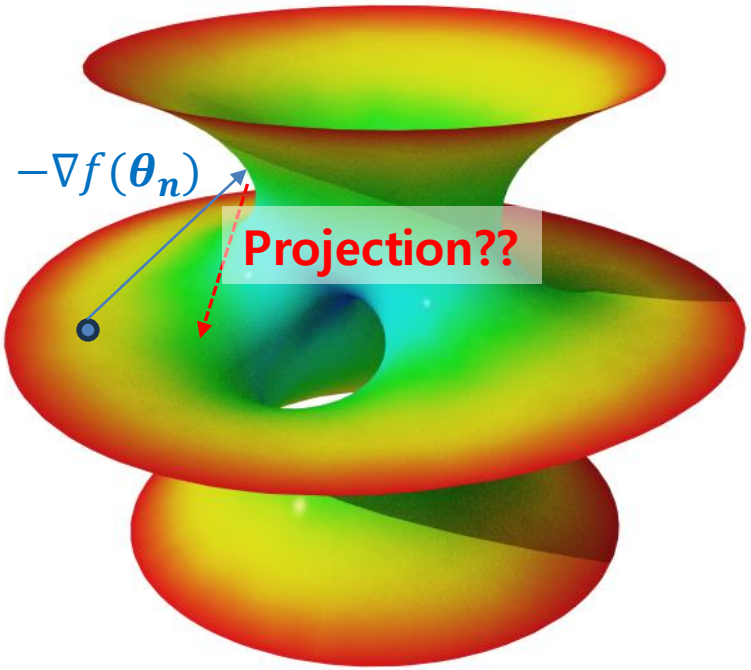
- $\Pi_{\boldsymbol{\Theta}}$: Projection onto $\boldsymbol{\Theta}$:
  - Not always easy!

(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}}\left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n)\right)$$



- $\Pi_{\boldsymbol{\Theta}}$: Projection onto $\boldsymbol{\Theta}$:
  - Not always easy!
- If $\boldsymbol{\Theta}$ is low-dim (say $r \ll p$)
  - Work only with r-dim stuffs?

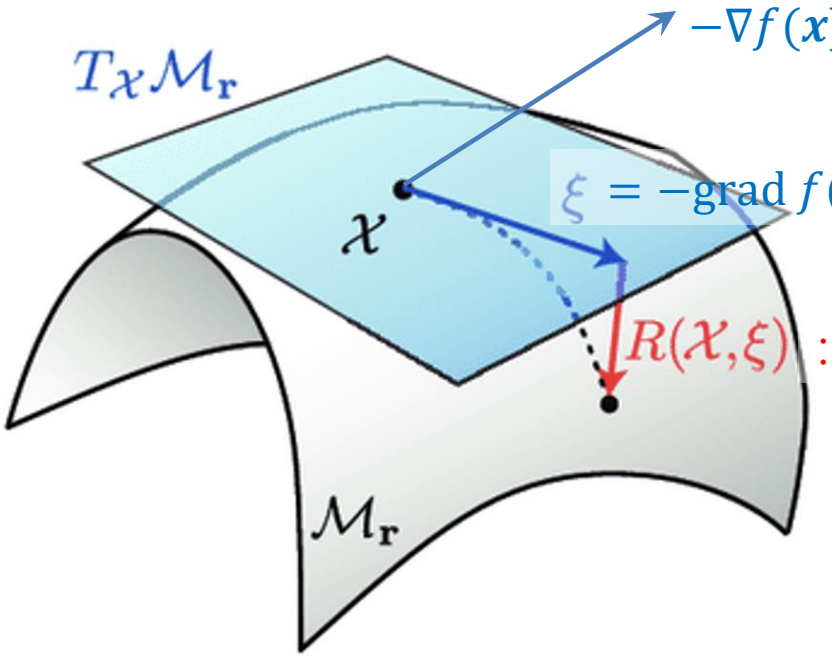
(PGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \Pi_{\boldsymbol{\Theta}}\left(\boldsymbol{\theta}_n - \alpha_n \nabla f(\boldsymbol{\theta}_n)\right)$$



- $\Pi_{\boldsymbol{\Theta}}$: Projection onto $\boldsymbol{\Theta}$:
  - Not always easy!
- If $\boldsymbol{\Theta}$ is low-dim (say $r \ll p$)
  - Work only with r-dim stuffs?
- *For certain low-rank matrix factorization problems, PGD with rSVD as projection can be useful*

  [Lee, Lyu, Yao NeurIPS '23]

(Riemannian GD) $\boldsymbol{\theta}_{n+1} \leftarrow \mathrm{Rtr}_{\boldsymbol{\theta}_n}\left(-\alpha_n \operatorname{grad} f(\boldsymbol{\theta}_n)\right)$



(Proj from $T_x \to \mathcal{M}$)

- RGD = gradient descent within the tangent space + Retraction

(RGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \mathrm{Rtr}_{\boldsymbol{\theta}_n}\left(-\alpha_n \operatorname{grad} f(\boldsymbol{\theta}_n)\right)$$

- Limitation: ***grad*** and ***Retraction*** *can be expensive to compute*

(RGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \mathrm{Rtr}_{\boldsymbol{\theta}_n}\left(-\alpha_n \operatorname{grad} f(\boldsymbol{\theta}_n)\right)$$

- Limitation: ***grad*** and ***Retraction*** *can be expensive to compute*

[Li, Lyu, Balzano, Needell ICML '24, JMLR (minor revision) '24+]

(Inexact RGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \widehat{\mathrm{Rtr}}_{\boldsymbol{\theta}_n}\left(-\alpha_n \widehat{\operatorname{grad}} f(\boldsymbol{\theta}_n)\right)$$

- ***It's OK to use approximate grad*** **and** ***Retraction***

  (Both mathematically and practically)

(RGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \mathrm{Rtr}_{\boldsymbol{\theta}_n}\left(-\alpha_n \operatorname{grad} f(\boldsymbol{\theta}_n)\right)$$

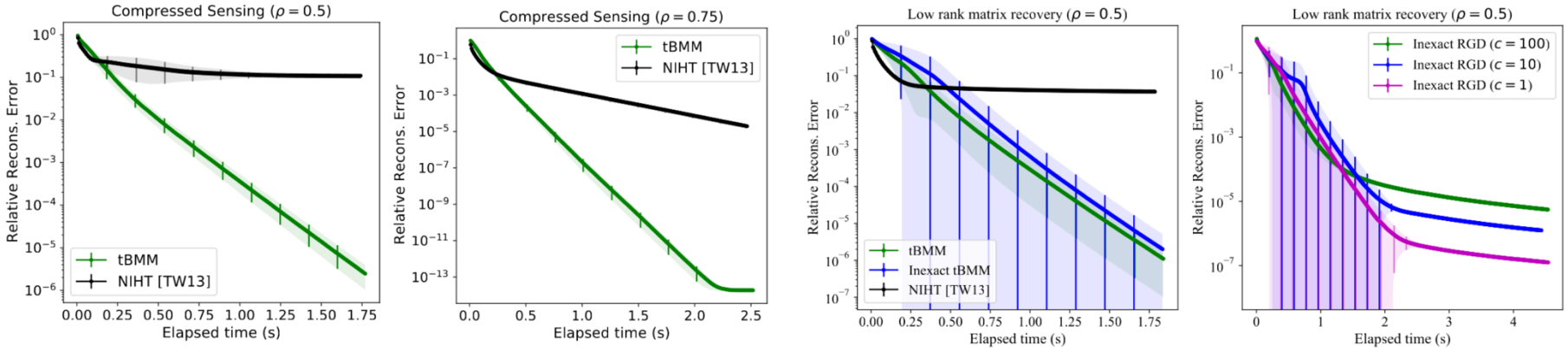
- Limitation: ***grad*** and ***Retraction*** *can be expensive to compute*

[Li, Lyu, Balzano, Needell ICML '24, JMLR (minor revision) '24+]

(Inexact RGD) $$\boldsymbol{\theta}_{n+1} \leftarrow \widehat{\mathrm{Rtr}}_{\boldsymbol{\theta}_n}\left(-\alpha_n \widehat{\operatorname{grad}} f(\boldsymbol{\theta}_n)\right)$$

- ***It's OK to use approximate grad* and *Retraction***

(Both mathematically and practically) *Block version too!*



Takeaways

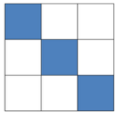
- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization

Takeaways

- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization
  - Alternating least squares – MF/NMF
  - Sinkhorn's algorithm – Matrix Scaling / Entropic OT / Random Matrices

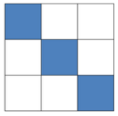
Takeaways

- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization
  - Alternating least squares – MF/NMF
  - Sinkhorn's algorithm – Matrix Scaling / Entropic OT / Random Matrices
- BPGD is practically more robust than PGD since they allow larger range of stepsizes



Takeaways

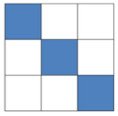
- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization
  - Alternating least squares – MF/NMF
  - Sinkhorn's algorithm – Matrix Scaling / Entropic OT / Random Matrices
- BPGD is practically more robust than PGD since they allow larger range of stepsizes



- BMM allows very flexible framework with use-designed surrogates

Takeaways

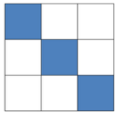
- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization
  - Alternating least squares – MF/NMF
  - Sinkhorn's algorithm – Matrix Scaling / Entropic OT / Random Matrices
- BPGD is practically more robust than PGD since they allow larger range of stepsizes



- BMM allows very flexible framework with use-designed surrogates
- The iteration complexity of BMM for the general case is $\tilde{O}(\epsilon^{-2})$

Takeaways

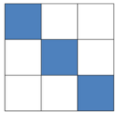
- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization
  - Alternating least squares – MF/NMF
  - Sinkhorn's algorithm – Matrix Scaling / Entropic OT / Random Matrices
- BPGD is practically more robust than PGD since they allow larger range of stepsizes



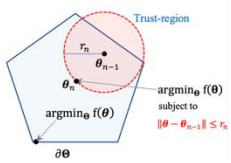
- BMM allows very flexible framework with use-designed surrogates
- The iteration complexity of BMM for the general case is $\tilde{O}(\epsilon^{-2})$
  - The implied constant blows up with "flat" surrogates

Takeaways

- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization
  - Alternating least squares – MF/NMF
  - Sinkhorn's algorithm – Matrix Scaling / Entropic OT / Random Matrices
- BPGD is practically more robust than PGD since they allow larger range of stepsizes

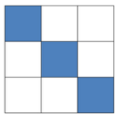


- BMM allows very flexible framework with use-designed surrogates
- The iteration complexity of BMM for the general case is $\tilde{O}(\epsilon^{-2})$
  - The implied constant blows up with "flat" surrogates
  - BMM+Trust Region overcomes this limitation

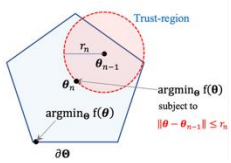


Takeaways

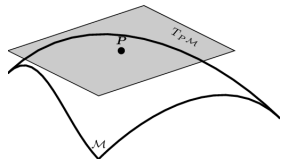
- Cyclic Block Optimization is a simple and powerful paradigm for general nonc onvex nonsmooth optimization
  - Alternating least squares – MF/NMF
  - Sinkhorn's algorithm – Matrix Scaling / Entropic OT / Random Matrices
- BPGD is practically more robust than PGD since they allow larger range of stepsizes



- BMM allows very flexible framework with use-designed surrogates
- The iteration complexity of BMM for the general case is $\tilde{O}(\epsilon^{-2})$
  - The implied constant blows up with "flat" surrogates
  - BMM+Trust Region overcomes this limitation



- BMM is available on Riemannian manifolds with similar theoretical guarantees



Thank you very much!

References

1. Joowon Lee, Hanbaek Lyu, and Weixin Yao, "*Exponentially Convergent Algorithms for Supervised Matrix Factorization*", NeurIPS 2023
2. Joowon Lee, Hanbaek Lyu, and Weixin Yao, "*Constrained Matrix Factorization: Local Landscape Analysis and Applications*" ICML 2024
3. Hanbaek Lyu, "*Stochastic regularized block majorization-minimization with weakly convex and multi-convex surrogates*" JMLR 24
4. Hanbaek Lyu, Christopher Strohmeier, and Deanna Needell, "*Online nonnegative tensor factorization and CP-Dictionary Learning for Markovian data*" JMLR 23(148):1–50, 2022
5. Hanbaek Lyu and Yuchen Li, "*Block majorization-minimization with diminishing radius for constrained nonconvex optimization*" To appear in SIOPT
6. Yuchen Li, Laura Balzano, Deanna Needell, Hanbaek Lyu, "*Convergence and Complexity Guarantee for Inexact First-order Riemannian Optimization Algorithms.*" ICML 2024
7. Yuchen Li, Laura Balzano, Deanna Needell, Hanbaek Lyu, "*Convergence and complexity of block majorization-minimization for constrained block-Riemannian optimization*" Minor revision in JMLR
8. Hanbaek Lyu and Sumit Mukherjee, "*Large random matrices with given margins*", arXiv:2407.14942