Stochastic semidefinite optimization via low-rank factorization: algorithms, theory and applications

Jinshan ZENG (HKUST)

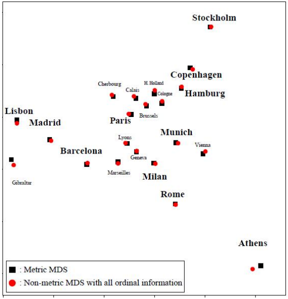
Joint work with Ke MA (IIE, CAS) and Prof. Yuan YAO (HKUST)

October 27, 2017

Outline

- Background
- Algorithms
- Numerical performance
- Convergence guarantees

Background

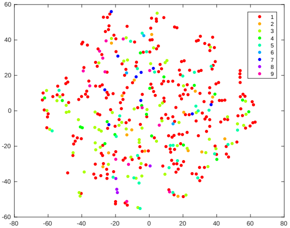


- **eurodist** dataset (stat.ethz.ch)
  - given over 20,000 ordinal distance comparisons, like

$$\mathrm{dist}(\mathrm{Rome}, \mathrm{Milan}) \leq \mathrm{dist}(\mathrm{Paris}, \mathrm{Athens})$$

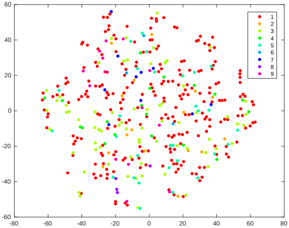
  - Task: draw a map between 21 cities in 2-dimensional space

Background



- **Music artist** dataset (Ellis et al. 2002)
  - Web-based survey: 1,032 users, 412 music artists
  - given 213,472 triplets (i, j, k) like $d_{ij}^2 \le d_{ik}^2$ means "*music artist* i *is more similar to artist* j *than artist* k"
  - nine music genres (rock, metal, pop, dance, hip hop, jazz, country, gospel, and reggae)
  - Task: *label the genres* for all music artists

Background



- **Music artist** dataset (Ellis et al. 2002)
  - Web-based survey: 1,032 users, 412 music artists
  - given 213,472 triplets (i, j, k) like $d_{ij}^2 \leq d_{ik}^2$ means "*music artist* i *is more similar to artist* j *than artist* k"
  - nine music genres (rock, metal, pop, dance, hip hop, jazz, country, gospel, and reggae)
  - Task: *label the genres* for all music artists

- such kind of problems called the ordinal embedding

Problem description

- a set of $\{o_1, \ldots, o_p\}$ in an abstract space $\mathbf{O}$

Problem description

- a set of $\{o_1, \ldots, o_p\}$ in an abstract space $\mathbf{O}$
- dissimilarity function $\xi : \mathbf{O} \times \mathbf{O} \to \mathbb{R}^+$: existing but unknown
  assigns the dissimilarity value ξ_{ij} for a pair of objects (o_i, o_j)

Problem description

- a set of $\{o_1, \ldots, o_p\}$ in an abstract space $\mathbf{O}$
- dissimilarity function $\xi : \mathbf{O} \times \mathbf{O} \to \mathbb{R}^+$: existing but unknown
  assigns the dissimilarity value ξ_{ij} for a pair of objects (o_i, o_j)
- ordinal constraint set:
  $\mathcal{C} = \{(i, j, l, k) \mid$ if exist o_i, o_j, o_k, o_l satisfy $\xi_{ij} < \xi_{lk}\}$

Problem description

- a set of $\{o_1, \ldots, o_p\}$ in an abstract space $\mathbf{O}$
- dissimilarity function $\xi : \mathbf{O} \times \mathbf{O} \to \mathbb{R}^+$: existing but unknown
  assigns the dissimilarity value ξ_{ij} for a pair of objects (o_i, o_j)
- ordinal constraint set:
  $\mathcal{C} = \{(i, j, l, k) \mid$ if exist o_i, o_j, o_k, o_l satisfy $\xi_{ij} < \xi_{lk}\}$
- Task: obtain representations of $\{o_1, \ldots, o_p\}$ in $\mathbb{R}^r$, $r(\ll p)$: embedding dimension

Low-rank semidefinite optimization formulation

- $\mathbf{U} \in \mathbb{R}^{p \times r}$: an embedding preserving the ordinal constraints in $\mathcal{C}$

$$(i, j, l, k) \in \mathcal{C} \Leftrightarrow \xi_{ij} < \xi_{lk} \Leftrightarrow d_{ij}^2(\mathbf{U}) < d_{lk}^2(\mathbf{U})$$

$d_{ij}^2(\mathbf{U}) = \|\mathbf{U}_i - \mathbf{U}_j\|^2$ the squared Euclidean norm

Low-rank semidefinite optimization formulation

- $\mathbf{U} \in \mathbb{R}^{p \times r}$: an embedding preserving the ordinal constraints in $\mathcal{C}$

$$(i, j, l, k) \in \mathcal{C} \Leftrightarrow \xi_{ij} < \xi_{lk} \Leftrightarrow d_{ij}^2(\mathbf{U}) < d_{lk}^2(\mathbf{U})$$

$d_{ij}^2(\mathbf{U}) = \|\mathbf{U}_i - \mathbf{U}_j\|^2$ the squared Euclidean norm

- Distance matrix $\mathbf{D} = \{d_{ij}^2(\mathbf{U})\}$: can be determined by $\mathbf{X} = \mathbf{U}\mathbf{U}^T$

$$\mathbf{D} = \mathrm{diag}(\mathbf{X})\mathbf{1}^T - 2\mathbf{X} + \mathbf{1}\mathrm{diag}(\mathbf{X})^T$$

Low-rank semidefinite optimization formulation

- $\mathbf{U} \in \mathbb{R}^{p \times r}$: an embedding preserving the ordinal constraints in $\mathcal{C}$

$$(i,j,l,k) \in \mathcal{C} \Leftrightarrow \xi_{ij} < \xi_{lk} \Leftrightarrow d_{ij}^2(\mathbf{U}) < d_{lk}^2(\mathbf{U})$$

$d_{ij}^2(\mathbf{U}) = \|\mathbf{U}_i - \mathbf{U}_j\|^2$ the squared Euclidean norm

- Distance matrix $\mathbf{D} = \{d_{ij}^2(\mathbf{U})\}$: can be determined by $\mathbf{X} = \mathbf{U}\mathbf{U}^T$

$$\mathbf{D} = \operatorname{diag}(\mathbf{X})\mathbf{1}^T - 2\mathbf{X} + \mathbf{1}\operatorname{diag}(\mathbf{X})^T$$

- finding $U \Rightarrow$ finding $\mathbf{X}$ via SDP (Agarwal et al., 2007)

$$\min_{\mathbf{X} \in \mathbb{R}^{p \times p}} \ f(\mathbf{X}) = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \ell_c(\mathbf{X}) + \lambda \cdot \mathrm{tr}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{X} \succeq 0$$

ℓ_c: some loss function like ℓ_2, hinge, logistic loss, et al. (shown latter)

Low-rank semidefinite optimization formulation

- $\mathbf{U} \in \mathbb{R}^{p \times r}$: an embedding preserving the ordinal constraints in $\mathcal{C}$

$$(i, j, l, k) \in \mathcal{C} \Leftrightarrow \xi_{ij} < \xi_{lk} \Leftrightarrow d_{ij}^2(\mathbf{U}) < d_{lk}^2(\mathbf{U})$$

$d_{ij}^2(\mathbf{U}) = \|\mathbf{U}_i - \mathbf{U}_j\|^2$ the squared Euclidean norm

- Distance matrix $\mathbf{D} = \{d_{ij}^2(\mathbf{U})\}$: can be determined by $\mathbf{X} = \mathbf{U}\mathbf{U}^T$

$$\mathbf{D} = \mathrm{diag}(\mathbf{X})\mathbf{1}^T - 2\mathbf{X} + \mathbf{1}\mathrm{diag}(\mathbf{X})^T$$

- finding $U \Rightarrow$ finding $\mathbf{X}$ via SDP (Agarwal et al., 2007)

$$\min_{\mathbf{X} \in \mathbb{R}^{p \times p}} \; f(\mathbf{X}) = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \ell_c(\mathbf{X}) + \lambda \cdot \mathrm{tr}(\mathbf{X}) \quad \text{s.t.} \quad \mathbf{X} \succeq 0$$

ℓ_c: some loss function like ℓ_2, hinge, logistic loss, et al. (shown latter)

- U: the largest r eigenvectors of $\mathbf{X}$ via SVD

Typical loss functions in ordinal embedding

- $\triangle_c := d_{ij}^2(\mathbf{U}) - d_{lk}^2(\mathbf{U}),\ c = (i, j, l, k).$

Typical loss functions in ordinal embedding

- $\triangle_c := d_{ij}^2(\mathbf{U}) - d_{lk}^2(\mathbf{U}),\ \ c = (i, j, l, k).$
- hinge loss (Agarwal et al. 2007, Generalized Non-Metric Multidimensional Scaling (GNMDS)):

$$\ell_c(\mathbf{X}) = \max\{0, 1 + \triangle_c\}$$

Typical loss functions in ordinal embedding

- $\triangle_c := d_{ij}^2(\mathbf{U}) - d_{lk}^2(\mathbf{U}),\ c = (i, j, l, k).$
- hinge loss (Agarwal et al. 2007, Generalized Non-Metric Multidimensional Scaling (GNMDS)):

$$\ell_c(\mathbf{X}) = \max\{0, 1 + \triangle_c\}$$

- logistic loss (van der Maaten and Weinberger 2012, Stochastic Triplet Embedding (STE)):

$$\ell_c(\mathbf{X}) = -\log(1 + \exp(\triangle_c))$$

Typical loss functions in ordinal embedding

- $\triangle_c := d_{ij}^2(\mathbf{U}) - d_{lk}^2(\mathbf{U}),\ c = (i, j, l, k).$
- hinge loss (Agarwal et al. 2007, Generalized Non-Metric Multidimensional Scaling (GNMDS)):

$$\ell_c(\mathbf{X}) = \max\{0, 1 + \triangle_c\}$$

- logistic loss (van der Maaten and Weinberger 2012, Stochastic Triplet Embedding (STE)):

$$\ell_c(\mathbf{X}) = -\log(1 + \exp(\triangle_c))$$

- logistic loss with the Student-t kernel with degree α (van der Maaten and Weinberger 2012, Stochastic Triplet Embedding with Student-t (TSTE)):

$$\ell_c(\mathbf{X}) = -\log \frac{\left(1 + \frac{d_{ij}^2(\mathbf{U})}{\alpha}\right)^{-\frac{\alpha+1}{2}}}{\left(1 + \frac{d_{ij}^2(\mathbf{U})}{\alpha}\right)^{-\frac{\alpha+1}{2}} + \left(1 + \frac{d_{lk}^2(\mathbf{U})}{\alpha}\right)^{-\frac{\alpha+1}{2}}}$$

Beyond ordinal embedding: matrix sensing

- In this talk, we consider *Nonlinear stochastic semidefinite optimization*

$$\text{(P1)} \qquad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

Beyond ordinal embedding: matrix sensing

- In this talk, we consider *Nonlinear stochastic semidefinite optimization*

$$(\text{P1}) \qquad \min_{X \in \mathbb{R}^{p \times p}} \; f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- **Matrix sensing**: recovering a low-rank matrix from linear measurements (Jain-Netrapalli-Sanghavi'13, Tu et al.'16)

$$\min_{X} \; f(X) = \frac{1}{2n} \sum_{i=1}^{n} (b_i - \langle A_i, X \rangle)^2, \text{ s.t. } \operatorname{rank}(X) \leq r, X \succeq 0$$

$\mathcal{A} : \mathbb{R}^{p \times p} \to \mathbb{R}^m$ linear sensing mechanism, $[\mathcal{A}(X)]_i = \langle A_i, X \rangle$,
$A_i \in \mathbb{R}^{p \times p}$: sub-Gaussian independent measurement matrices.

Beyond ordinal embedding: phase retrieval

- **Phase retrieval**: recovering the phase from magnitude measurements
- arising in X-ray crystallography and related disciplines

Beyond ordinal embedding: phase retrieval

- **Phase retrieval**: recovering the phase from magnitude measurements
- arising in X-ray crystallography and related disciplines
- given phaseless measurements

$$y_i = |\langle a_i, x \rangle|^2, \quad k = 1, \ldots, m.$$

$a_k \in \mathbb{C}^p$: sampling vectors

Beyond ordinal embedding: phase retrieval

- **Phase retrieval**: recovering the phase from magnitude measurements
- arising in X-ray crystallography and related disciplines
- given phaseless measurements

$$y_i = |\langle a_i, x \rangle|^2, \quad k = 1, \dots, m.$$

  $a_k \in \mathbb{C}^p$: sampling vectors

- Convex relaxation (Candes et al.'13, 15)

$$\min_X \; f(X) = \frac{1}{2n} \sum_{i=1}^{n} |y_i - \mathrm{tr}(a_i a_i^* X)|^2 + \lambda \cdot \mathrm{tr}(X) \quad \text{s.t.} \quad X \succeq 0,$$

  $X \in \mathbb{S}^p_+$, $\mathrm{tr}(a_i a_i^* X) = y_i, i = 1, \dots, m.$

More applications $\ldots \widehat{\odot\odot}$

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$\text{(P1)} \qquad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$\text{(P1)} \qquad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- Projected gradient descent (Nestrov & Nemiroovski'89)
  - $X^{t+1} = \mathrm{Proj}_{\mathbb{S}_+^p}(X^t - \eta \nabla f(X^t))$, $\eta > 0$: step size

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$\text{(P1)} \quad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- Projected gradient descent (Nestrov & Nemiroovski'89)
  - $X^{t+1} = \mathrm{Proj}_{\mathbb{S}^p_+}(X^t - \eta \nabla f(X^t))$, $\eta > 0$: step size
  - SVD-based: $O(p^3)$,

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$\text{(P1)} \qquad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- Projected gradient descent (Nestrov & Nemiroovski'89)
  - $X^{t+1} = \mathrm{Proj}_{\mathbb{S}_+^p}(X^t - \eta \nabla f(X^t))$, $\eta > 0$: step size
  - SVD-based: $O(p^3)$, computational burden when $p \gg 1$

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$(\text{P1}) \qquad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- Projected gradient descent (Nestrov & Nemiroovski'89)
  - $X^{t+1} = \mathrm{Proj}_{\mathbb{S}_+^p}(X^t - \eta \nabla f(X^t))$, $\eta > 0$: step size
  - SVD-based: $O(p^3)$, computational burden when $p \gg 1$
  - batch method:

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$\text{(P1)} \qquad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- Projected gradient descent (Nestrov & Nemiroovski'89)
  - $X^{t+1} = \mathrm{Proj}_{\mathbb{S}_+^p}(X^t - \eta \nabla f(X^t))$, $\eta > 0$: step size
  - SVD-based: $O(p^3)$, computational burden when $p \gg 1$
  - batch method: evaluation of full gradient is expense when $n \gg 1$

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$(\text{P1}) \quad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- Projected gradient descent (Nestrov & Nemiroovski'89)
  - $X^{t+1} = \text{Proj}_{\mathbb{S}_+^p}(X^t - \eta \nabla f(X^t))$, $\eta > 0$: step size
  - SVD-based: $O(p^3)$, computational burden when $p \gg 1$
  - batch method: evaluation of full gradient is expense when $n \gg 1$
  - time and memory-intensive with poor scability

Existing methods

- Recall the stochastic semidefinite optimization problem (f_i's are smooth)

$$(\text{P1}) \qquad \min_{X \in \mathbb{R}^{p \times p}} f(X) = \frac{1}{n} \sum_{i=1}^{n} f_i(X) \quad \text{s.t.} \quad X \succeq 0$$

- Projected gradient descent (Nestrov & Nemiroovski'89)
  - $X^{t+1} = \text{Proj}_{\mathbb{S}_+^p}(X^t - \eta \nabla f(X^t))$, $\eta > 0$: step size
  - SVD-based: $O(p^3)$, computational burden when $p \gg 1$
  - batch method: evaluation of full gradient is expense when $n \gg 1$
  - time and memory-intensive with poor scability
- Other (not-scalable too, about 1000 dimensions)
  - Interior point method (Alizadeh'95, Ye'96)
  - Path-following interior point method (see, Monteiro'03)

Existing methods

- Factored gradient descent (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$\text{(P2)} \quad \underset{U\in\mathbb{R}^{p\times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

Existing methods

- Factored gradient descent (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$\text{(P2)} \quad \underset{U\in\mathbb{R}^{p\times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - Problem (P1) $\Leftrightarrow$ Problem (P2) if $r = r^* (:= \mathrm{rank}(X^*))$

Existing methods

- Factored gradient descent (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$\text{(P2)} \quad \underset{U\in\mathbb{R}^{p\times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - Problem (P1) $\Leftrightarrow$ Problem (P2) if $r = r^*(:= \text{rank}(X^*))$
  - Some pioneer work for SDP (Homer and Peinado 1997; Burer and Monteiro 2001; 2003; 2005)

Existing methods

- Factored gradient descent (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p \times r}$, letting $X = UU^T$

$$(\text{P2}) \quad \underset{U \in \mathbb{R}^{p \times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - Problem (P1) $\Leftrightarrow$ Problem (P2) if $r = r^* (:= \text{rank}(X^*))$
  - Some pioneer work for SDP (Homer and Peinado 1997; Burer and Monteiro 2001; 2003; 2005)
  - Two benefits: (a) PSD is eliminated and (b) $pr \ll p(p+1)/2$

Existing methods

- Factored gradient descent (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$(\text{P2}) \quad \underset{U\in\mathbb{R}^{p\times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - Problem (P1) $\Leftrightarrow$ Problem (P2) if $r = r^*(:= \text{rank}(X^*))$
  - Some pioneer work for SDP (Homer and Peinado 1997; Burer and Monteiro 2001; 2003; 2005)
  - Two benefits: (a) PSD is eliminated and (b) $pr \ll p(p+1)/2$ but "No free lunch": Problem (P2) is generally nonconvex

Existing methods

- Factored gradient descent (FGD) (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$\text{(P2)} \quad \underset{U\in\mathbb{R}^{p\times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

[1] This is deduced by noting that $\nabla g(U) = (\nabla f(UU^T) + \nabla f(UU^T)^T)U = 2\nabla f(UU^T)U$ due to f is assumed to be symmetric. "factored" is named as the constant '2' is absorbed.

[2] Here, *linear global convergence* means that the algorithm converges to a global optimum exponentially fast.

Existing methods

- Factored gradient descent (FGD) (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p \times r}$, letting $X = UU^T$

$$\text{(P2)} \quad \underset{U \in \mathbb{R}^{p \times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - How to solve?

[1] This is deduced by noting that $\nabla g(U) = (\nabla f(UU^T) + \nabla f(UU^T)^T)U = 2\nabla f(UU^T)U$ due to f is assumed to be symmetric. "factored" is named as the constant '2' is absorbed.

[2] Here, *linear global convergence* means that the algorithm converges to a global optimum exponentially fast.

Existing methods

- Factored gradient descent (FGD) (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$(\mathrm{P2}) \quad \underset{U\in\mathbb{R}^{p\times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - How to solve? a simple "gradient descent" $\Rightarrow$ FGD [1]:

$$U^{k+1} = U^k - \eta\nabla f(X^k)U^k$$

  where $X^k := U^k {U^k}^T$

[1] This is deduced by noting that $\nabla g(U) = (\nabla f(UU^T) + \nabla f(UU^T)^T)U = 2\nabla f(UU^T)U$ due to f is assumed to be symmetric. "factored" is named as the constant '2' is absorbed.

[2] Here, *linear global convergence* means that the algorithm converges to a global optimum exponentially fast.

Existing methods

- Factored gradient descent (FGD) (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$\text{(P2)} \quad \underset{U\in\mathbb{R}^{p\times r}}{\text{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - How to solve? a simple "gradient descent" $\Rightarrow$ FGD [1]:

$$U^{k+1} = U^k - \eta \nabla f(X^k) U^k$$

    where $X^k := U^k U^{k\,T}$

  - SVD-free with cheaper computation

[1] This is deduced by noting that $\nabla g(U) = (\nabla f(UU^T) + \nabla f(UU^T)^T)U = 2\nabla f(UU^T)U$ due to f is assumed to be symmetric. "factored" is named as the constant '2' is absorbed.

[2] Here, *linear global convergence* means that the algorithm converges to a global optimum exponentially fast.

Existing methods

- Factored gradient descent (FGD) (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p\times r}$, letting $X = UU^T$

$$(\mathrm{P2}) \quad \underset{U\in\mathbb{R}^{p\times r}}{\mathrm{minimize}} \; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - How to solve? a simple "gradient descent" $\Rightarrow$ FGD [1]:

$$U^{k+1} = U^k - \eta \nabla f(X^k) U^k$$

    where $X^k := U^k {U^k}^T$

  - SVD-free with cheaper computation
  - certain *linear global convergence*[2] is established in the strongly convex case and under other strict conditions

[1] This is deduced by noting that $\nabla g(U) = (\nabla f(UU^T) + \nabla f(UU^T)^T)U = 2\nabla f(UU^T)U$ due to f is assumed to be symmetric. "factored" is named as the constant '2' is absorbed.

[2] Here, *linear global convergence* means that the algorithm converges to a global optimum exponentially fast.

Existing methods

- Factored gradient descent (FGD) (Bhojanapalli et al.'16)
  - Low-rank factorization: Introducing $U \in \mathbb{R}^{p \times r}$, letting $X = UU^T$

$$\text{(P2)} \quad \underset{U \in \mathbb{R}^{p \times r}}{\text{minimize}} \;\; g(U) := f(UU^T) \quad \text{where} \quad r \ll p.$$

  - How to solve? a simple "gradient descent" $\Rightarrow$ FGD [1]:

$$U^{k+1} = U^k - \eta \nabla f(X^k) U^k$$

    where $X^k := U^k {U^k}^T$

  - SVD-free with cheaper computation
  - certain *linear global convergence*[2] is established in the strongly convex case and under other strict conditions
  - but still a batch method

[1] This is deduced by noting that $\nabla g(U) = (\nabla f(UU^T) + \nabla f(UU^T)^T)U = 2\nabla f(UU^T)U$ due to f is assumed to be symmetric. "factored" is named as the constant '2' is absorbed.

[2] Here, *linear global convergence* means that the algorithm converges to a global optimum exponentially fast.

Aim of this talk

- Adapt more efficient algorithms for the SOP with
  - reduced computational cost
  - low memory storage
  - theoretical guarantees

Main Results

- adapt the stochastic FGD (SFGD) to solve the considered semidefinite optimization problem and establish its $O(1/k)$-*global convergence rate* under much more relaxed conditions than the existing results.

---

[3] SVRG is firstly proposed by Johnson&Zhang'2013 to accelerate the SGD for the stochastic optimization problem in the vector space

Main Results

- adapt the stochastic FGD (SFGD) to solve the considered semidefinite optimization problem and establish its $O(1/k)$-*global convergence rate* under much more relaxed conditions than the existing results.
- adapt a faster algorithm called SVRG [3] for this semidefinite optimization problem and establish its *linear global convergence rate* under almost the same conditions of SFGD

[3] SVRG is firstly proposed by Johnson&Zhang'2013 to accelerate the SGD for the stochastic optimization problem in the vector space

Main Results

- adapt the stochastic FGD (SFGD) to solve the considered semidefinite optimization problem and establish its $O(1/k)$-*global convergence rate* under much more relaxed conditions than the existing results.
- adapt a faster algorithm called SVRG [3] for this semidefinite optimization problem and establish its *linear global convergence rate* under almost the same conditions of SFGD
- In ordinal embedding application, we introduce a new step size called stabilized Barzilai-Borwein (SBB) for SVRG, and establish the $O(1/k)$-rate of convergence to a stationary point in the *smooth* case but not necessary strong convexity and even convexity.

[3] SVRG is firstly proposed by Johnson&Zhang'2013 to accelerate the SGD for the stochastic optimization problem in the vector space

Algorithms: SFGD and SVRG

Stochastic Factored Gradient Descent

- Main idea
  - low-rank factorization used in FGD $\Rightarrow$ reduced computational cost
  - stochastic approximation (Robbins & Monro 1951) $\Rightarrow$ low memory storage and also speedup

Stochastic Factored Gradient Descent

- Main idea
  - low-rank factorization used in FGD $\Rightarrow$ reduced computational cost
  - stochastic approximation (Robbins & Monro 1951) $\Rightarrow$ low memory storage and also speedup
- **Stochastic FGD (SFGD)**: randomly pick $i_k \in \{1, \ldots, n\}$

$$U^{k+1} = U^k - \eta_k \nabla f_{i_k}(X^k) U^k,$$

$\eta_k > 0$: diminishing step size satisfying

$$\sum_{k=0}^{\infty} \eta_k = \infty, \text{ and } \sum_{k=0}^{\infty} \eta_k^2 < \infty.$$

Stochastic Factored Gradient Descent

- Main idea
  - low-rank factorization used in FGD $\Rightarrow$ reduced computational cost
  - stochastic approximation (Robbins & Monro 1951) $\Rightarrow$ low memory storage and also speedup
- **Stochastic FGD (SFGD)**: randomly pick $i_k \in \{1, \ldots, n\}$

$$U^{k+1} = U^k - \eta_k \nabla f_{i_k}(X^k) U^k,$$

  $\eta_k > 0$: diminishing step size satisfying

$$\sum_{k=0}^{\infty} \eta_k = \infty, \text{ and } \sum_{k=0}^{\infty} \eta_k^2 < \infty.$$

- $\eta_k = \frac{\eta}{k+1}$ for some small $\eta > 0$

SFGD: theoretical guarantees

- $O(1/k)$-rate of convergence to a global optimum in (restricted) strongly convex case, starting from a good initialization (shown latter in this talk)

SFGD: theoretical guarantees

- $O(1/k)$-rate of convergence to a global optimum in (restricted) strongly convex case, starting from a good initialization (shown latter in this talk)
- $O(1/\sqrt{k})$-rate of convergence to a stationary point beyond strongly convex case, regardless of the initialization (Ghadimi-Lan'13)

SFGD: theoretical guarantees

- $O(1/k)$-rate of convergence to a global optimum in (restricted) strongly convex case, starting from a good initialization (shown latter in this talk)
- $O(1/\sqrt{k})$-rate of convergence to a stationary point beyond strongly convex case, regardless of the initialization (Ghadimi-Lan'13)
- Q: are there faster stochastic algorithms to achieve the linear convergence rate in the strongly convex case?

SVRG: A faster one

- Stochastic variance reduced gradient (Johnson&Zhang'13)

SVRG: A faster one

- Stochastic variance reduced gradient (Johnson&Zhang'13)
- Main idea: introducing an inner loop to reduce the inherent variance

Algorithm 1 Stochastic Variance Reduced Gradient (SVRG) for (1)

Parameters: update frequency m, step size (or learning rate) $\{\eta_k\}$, initial point $\tilde{U}^0 \in \mathbb{R}^{p\times r}$

for k=0,1,... **do**

$\quad \tilde{X}^k := \tilde{U}^k \tilde{U}^{k^T}$

$\quad g_k = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\tilde{X}^k)\tilde{U}^k$

$\quad U^0 = \tilde{U}^k$

$\quad$ **for** $t = 0, \ldots, m-1$ **do**

$\quad\quad X^t = U^t U^{t^T}$

$\quad\quad$ Randomly pick $i_t \in \{1, \ldots, n\}$

$\quad\quad U^{t+1} = U^t - \eta_k(\nabla f_{i_t}(X^t)U^t - \nabla f_{i_t}(\tilde{X}^k)\tilde{U}^k + g_k)$

$\quad$ **end for**

$\quad \tilde{U}^{k+1} = U^m$

end for

SVRG: A faster one

- Stochastic variance reduced gradient (Johnson&Zhang'13)
- Main idea: introducing an inner loop to reduce the inherent variance

Algorithm 1 Stochastic Variance Reduced Gradient (SVRG) for (1)

Parameters: update frequency m, step size (or learning rate) $\{\eta_k\}$, initial point $\tilde{U}^0 \in \mathbb{R}^{p \times r}$

for k=0,1,... **do**

$\tilde{X}^k := \tilde{U}^k \tilde{U}^{k^T}$

$g_k = \frac{1}{n} \sum_{i=1}^n \nabla f_i(\tilde{X}^k) \tilde{U}^k$

$U^0 = \tilde{U}^k$

for $t = 0, \ldots, m-1$ **do**

$X^t = U^t U^{t^T}$

Randomly pick $i_t \in \{1, \ldots, n\}$

$U^{t+1} = U^t - \eta_k (\nabla f_{i_t}(X^t) U^t - \nabla f_{i_t}(\tilde{X}^k) \tilde{U}^k + g_k)$

end for

$\tilde{U}^{k+1} = U^m$

end for

- not a "true" but an "artificial" stochastic algorithm

SVRG: A faster one

- Stochastic variance reduced gradient (Johnson&Zhang'13)
- Main idea: introducing an inner loop to reduce the inherent variance

Algorithm 1 Stochastic Variance Reduced Gradient (SVRG) for (1)

Parameters: update frequency m, step size (or learning rate) $\{\eta_k\}$, initial point $\tilde{U}^0 \in \mathbb{R}^{p\times r}$
for k=0,1,... **do**
  $\tilde{X}^k := \tilde{U}^k {\tilde{U}^k}^T$
  $g_k = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\tilde{X}^k)\tilde{U}^k$
  $U^0 = \tilde{U}^k$
  for $t = 0, \ldots, m-1$ **do**
    $X^t = U^t {U^t}^T$
    Randomly pick $i_t \in \{1, \ldots, n\}$
    $U^{t+1} = U^t - \eta_k(\nabla f_{i_t}(X^t)U^t - \nabla f_{i_t}(\tilde{X}^k)\tilde{U}^k + g_k)$
  end for
  $\tilde{U}^{k+1} = U^m$
end for

- not a "true" but an "artificial" stochastic algorithm
- a comparable one called "SAG" (Schmidt-Le Roux-Bach'16)

SVRG: step size strategies

- Fixed step size: $\eta_k \equiv \eta$
  η should be tuned by hand

SVRG: step size strategies

- Fixed step size: $\eta_k \equiv \eta$
  η should be tuned by hand

- Barzilai-Borwein (BB) step size (Barzilai-Borwein'1988; Tan et al'16)

$$\eta_k = \frac{1}{m} \cdot \frac{\|\tilde{X}^k - \tilde{X}^{k-1}\|_F^2}{\langle \tilde{X}^k - \tilde{X}^{k-1}, \tilde{g}_k - \tilde{g}_{k-1} \rangle}, \text{ where } \tilde{g}_k := \nabla f(\tilde{X}^k)$$

  adaptive, but effective only for strongly convex

SVRG: step size strategies

- Fixed step size: $\eta_k \equiv \eta$

  η should be tuned by hand

- Barzilai-Borwein (BB) step size (Barzilai-Borwein'1988; Tan et al'16)

$$\eta_k = \frac{1}{m} \cdot \frac{\|\tilde{X}^k - \tilde{X}^{k-1}\|_F^2}{\langle \tilde{X}^k - \tilde{X}^{k-1}, \tilde{g}_k - \tilde{g}_{k-1} \rangle}, \text{ where } \tilde{g}_k := \nabla f(\tilde{X}^k)$$

  adaptive, but effective only for strongly convex

- Stabilized BB (SBB_ϵ) step size (Ma-Z-Yao-et al'17): given small $\epsilon \geq 0$,

$$\eta_k = \frac{1}{m} \cdot \frac{\|\tilde{X}^k - \tilde{X}^{k-1}\|_F^2}{|\langle \tilde{X}^k - \tilde{X}^{k-1}, \tilde{g}_k - \tilde{g}_{k-1} \rangle| + \epsilon \|\tilde{X}^k - \tilde{X}^{k-1}\|_F^2}$$

  adaptive, effective for all smooth cases but not limited to strongly cvx

SVRG: step size strategies

- Fixed step size: $\eta_k \equiv \eta$
  η should be tuned by hand

- Barzilai-Borwein (BB) step size (Barzilai-Borwein'1988; Tan et al'16)

$$\eta_k = \frac{1}{m} \cdot \frac{\|\tilde{X}^k - \tilde{X}^{k-1}\|_F^2}{\langle \tilde{X}^k - \tilde{X}^{k-1}, \tilde{g}_k - \tilde{g}_{k-1} \rangle}, \text{ where } \tilde{g}_k := \nabla f(\tilde{X}^k)$$

  adaptive, but effective only for strongly convex

- Stabilized BB (SBB_ϵ) step size (Ma-Z-Yao-et al'17): given small $\epsilon \geq 0$,

$$\eta_k = \frac{1}{m} \cdot \frac{\|\tilde{X}^k - \tilde{X}^{k-1}\|_F^2}{|\langle \tilde{X}^k - \tilde{X}^{k-1}, \tilde{g}_k - \tilde{g}_{k-1} \rangle| + \epsilon \|\tilde{X}^k - \tilde{X}^{k-1}\|_F^2}$$

  adaptive, effective for all smooth cases but not limited to strongly cvx

- when applied to a strongly cvx function, BB $\Leftrightarrow$ SBB_0

SVRG: theoretical guarantees

- linear rate of convergence to a global optimum in (restricted) strongly convex case, starting from a good initialization (shown latter in this talk)

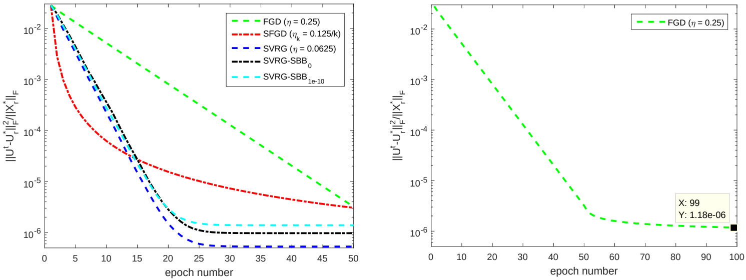
SVRG: theoretical guarantees

- linear rate of convergence to a global optimum in (restricted) strongly convex case, starting from a good initialization (shown latter in this talk)
- $O(1/k)$-rate of convergence to a stationary point beyond strongly convex case, regardless of the initialization (Reddi et al.'16; Allen-Zhu-Hazan'16; Ma-Z-Yao et al'17)

Numerical Experiments

Matrix Sensing

- $\min_{X \succeq 0}\ f(X) = \frac{1}{2n}\sum_{i=1}^{n}(b_i - \langle A_i, X\rangle)^2$
- $p = 5000$, $n = 10p$, $r^* = \mathrm{rank}(X^*) = 5$; For SVRG $m = n$



(a) Iterative error (b) Iterative error of FGD

Figure: Experiments for matrix sensing problem.

Ordinal Embedding: *eurodist* data

- 21 cities in Europe, 21,945 quadruplet comparisons

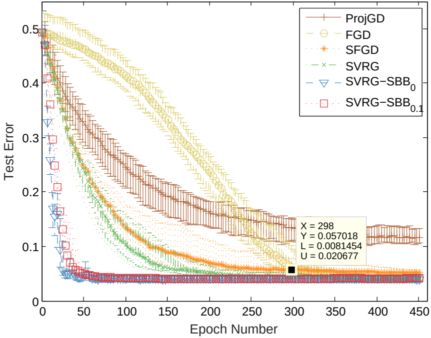
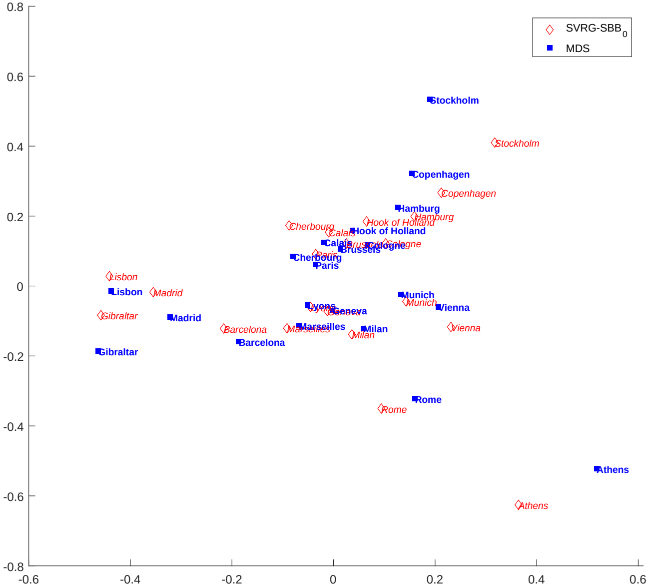
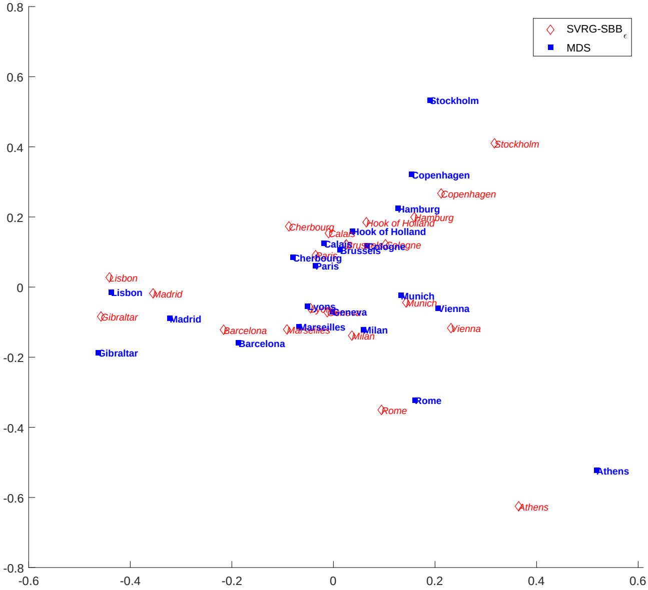


Figure: Experiments for *eurodist* dataset. To achieve the test error 0.057, about 40 epochs for SVRG-SBB$_0$ and SVRG-SBB$_{0.1}$, and 150 epochs for SVRG with a fixed step size, and 300 epochs for SFGD and FGD, and more than 450 epochs for ProjGD are required.

Ordinal Embedding: *eurodist* data (embedding result)



Ordinal Embedding: *eurodist* data (embedding result)



Ordinal Embedding: Music Artist data

- 412 music artists from 9 music genres (rock, metal, pop, dance, hip hop, jazz, country, gospel, and reggae)
- $213,472$ triplet comparisons

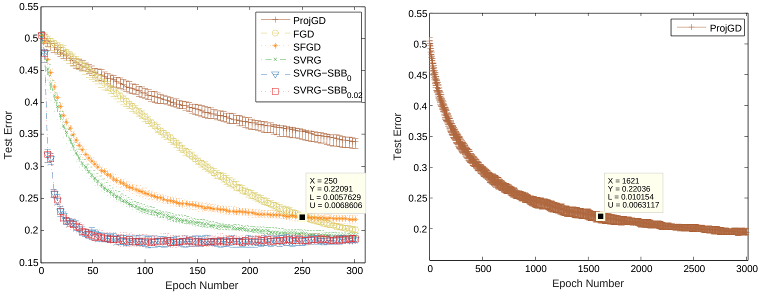
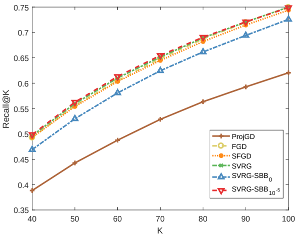


Figure: Experiments for Music artist data. To achieve the test error 0.22, about 40 epochs for SVRG-SBB$_0$ and SVRG-SBB$_{0.02}$, and 130 epochs for SVRG (fixed step size), and 260 epochs for both SFGD and FGD, and 1600 epochs for ProjGD are required.

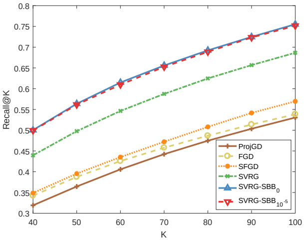
Ordinal Embedding: ***SUN 397*** data

- Task: image retrieval
- around 108K images from 397 scene categories
- each image is represented by a $1,600$-dimensional feature vector extracted by PCA from $12,288$-dimensional Deep Convolution Activation Features

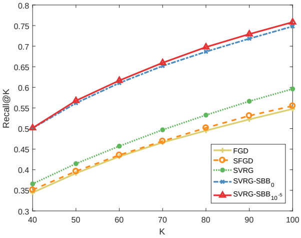
Ordinal Embedding: *SUN 397* data



(a) GNMDS



(b) STE



(c) TSTE

Figure: Recall@K with 10% noise on SUN397.

Ordinal Embedding: *SUN 397* data

Table: Image retrieval performance (MAP and Precision@40) on SUN397 when $p = 19$.

	0%		5%		10%	
	MAP	Precision@40	MAP	Precision@40	MAP	Precision@40
GNMDS						
ProjGD	0.2691	0.3840	0.2512	0.3686	0.2701	0.3883
FGD	0.3357	0.4492	0.3791	0.4914	0.3835	0.4925
SFGD	0.3245	0.4379	0.3635	0.4772	0.3819	0.4931
SVRG	0.3348	0.4490	0.3872	0.4974	0.3870	0.4965
SVRG-SBB$_0$	0.3941	0.5040	0.3700	0.4836	0.3550	0.4689
SVRG-SBB$_\epsilon$	0.3363	0.4500	0.3887	0.4981	0.3873	0.4987
STE						
ProjGD	0.2114	0.3275	0.1776	0.2889	0.1989	0.3190
FGD	0.2340	0.3525	0.2252	0.3380	0.2297	0.3423
SFGD	0.3369	0.4491	0.2951	0.4125	0.2390	0.3488
SVRG	0.3817	0.4927	0.3654	0.4804	0.3245	0.4395
SVRG-SBB$_0$	0.3968	0.5059	0.3958	0.5054	0.3895	0.5002
SVRG-SBB$_\epsilon$	0.3940	0.5036	0.3921	0.5012	0.3896	0.4992
TSTE						
FGD	0.2268	0.3470	0.2069	0.3201	0.2275	0.3447
SFGD	0.2602	0.3778	0.2279	0.3415	0.2402	0.3514
SVRG	0.3481	0.4617	0.3160	0.4332	0.2493	0.3656
SVRG-SBB$_0$	0.3900	0.4980	0.3917	0.5018	0.3914	0.5007
SVRG-SBB$_\epsilon$	0.3625	0.4719	0.3845	0.4936	0.3897	0.5013

- MAP: Mean Average Precision
- Precision@K $= \frac{1}{n}\sum_i^n p_i^K = \frac{1}{n}\sum_i^n \frac{\mathrm{TP}_i^K}{\mathrm{TP}_i^K + \mathrm{FP}_i^K}$
- Recall@K $= \frac{1}{n}\sum_i^n r_i^K = \frac{1}{n}\sum_i^n \frac{\mathrm{TP}_i^K}{\mathrm{TP}_i^K + \mathrm{FN}_i^K}$.

Theoretical Guarantees

Definitions

- L-**Lipschitz differentiable:** f is smooth and ∇f is L-Lipschitz
- μ-**strongly convex:**
  $f(Y) \geq f(X) + \langle \nabla f(X), Y - X \rangle + \frac{\mu}{2} \| Y - X \|_F^2, \ \forall X, Y \in \mathbb{S}_+^p$
- (μ, r)-**restricted strongly convex:**
  $f(Y) \geq f(X) + \langle \nabla f(X), Y - X \rangle + \frac{\mu}{2} \| Y - X \|_F^2, \ \forall X, Y \in \mathbb{S}_+^p$ with rank-r
- $\mathcal{E}$-**metric:** for any $U, V \in \mathbb{R}^{p \times r}$,
  - for SFGD: $\mathcal{E}(U, V) := \min_{R \in \mathcal{O}} \| U - VR \|_F^2$
  - for SVRG: $\mathcal{E}(U, V) := \| U - V \|_F^2$

Convergence Guarantees: Assumptions

Assumption (rank-r approximation error)

Let X^* be a global optimum of problem (P1), X_r^* be the rank-r best approximation of X^* for a given positive integer $r \leq r^* := \mathrm{rank}(X^*)$. The following holds

$$\|X_r^* - X^*\|_F < (\sqrt{2} - 1)\xi^{1/2}\kappa^{-1} \cdot \sigma_r(X_r^*),$$

where $\kappa := \frac{L}{\mu}$, and $\sigma_r(X_r^*)$ is the r-th largest singular value of X_r^*.

Assumption (Unbiasedness of stochastic direction)

$\{\nabla f_{i_k}(X^k)U^k\}$ *satisfies* $\mathbb{E}_{i_k}[\nabla f_{i_k}(X^k)U^k] = \nabla f(X^k)U^k, \ \forall k \in \mathbb{N}$.

Notations

- $\kappa := \frac{L}{\mu}, \quad \tau(X_r^*) = \frac{\sigma_1(X_r^*)}{\sigma_r(X_r^*)}$
- $\bar{\eta} := \min\left\{ \frac{(1-\sqrt{\gamma_0})^2}{\left[\frac{\|\nabla f(X_r^*)\|_F}{L\cdot\|X_r^*\|_2} + \frac{2\sqrt{\gamma_0}+\gamma_0}{\tau(U_r^*)}\right]\cdot\tau(X_r^*)}, 1 \right\}$, where $\gamma_0 := (\sqrt{2}-1)\kappa^{-1}$
- $\xi := \bar{\eta}(1-\bar{\eta}/2) \sim O(\frac{1}{\tau(X_r^*)})$
- $\Delta := \frac{(\sqrt{2}-1)^2\xi^2\sigma_r^2(X_r^*)}{\kappa^2} - \xi\|X_r^* - X^*\|_F^2$
- $\gamma_l := \frac{(\sqrt{2}-1)\xi\sigma_r(X_r^*)}{\kappa} - \sqrt{\Delta} \sim O\left(\frac{\kappa\|X_r^*-X^*\|_F^2}{\sigma_r(X_r^*)}\right)$
- $\gamma_u := \frac{(\sqrt{2}-1)\xi\sigma_r(X_r^*)}{\kappa} + \sqrt{\Delta} \sim O(\frac{\sigma_r(X_r^*)}{\kappa\tau(X_r^*)})$
- **Variance**: $B_0 := \sup_{\{U:\mathcal{E}(U,U_r^*)\le\gamma_u\}} \mathbb{E}_i[\|\nabla f_i(UU^T)U - \nabla f(UU^T)U\|_F^2]$
- **Upper bound of step size**: $\eta_{\max} := \min\left\{\frac{L\Delta}{4\xi B_0}, \frac{\xi}{8L(\gamma_u+\|X_r^*\|_F)}\right\}$

$O(1/k)$ Convergence of SFGD

- **diminishing step size:** (a) $\eta_k \geq 0$, (b) $\sum_{k=0}^{\infty} \eta_k = \infty$, and (c) $\sum_{k=0}^{\infty} \eta_k^2 < \infty$.
- $\eta_k = \frac{\eta}{k+1}$, for some $\eta \in (0, \eta_{\max})$.

Theorem ($O(1/k)$ convergence of SFGD)

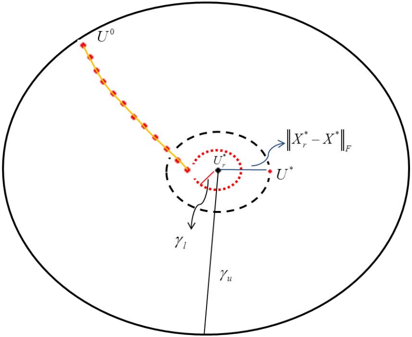
Let $\{U^k\}$ be a sequence generated by SFGD with diminishing step sizes η_k. Suppose that $f : \mathbb{S}_+^p \to \mathbb{R}$ is L-Lipschitz differentiable and (μ, r)-restricted strongly convex, and that Assumptions 1 and 2 hold. The following hold:

(1) *Assume that U^0 satisfies $\gamma_l < \mathcal{E}(U^0, U_r^*) < \gamma_u$. Then there hold*

- (a1) $\{\mathbb{E}[\mathcal{E}(U^k, U_r^*)]\}$ *is monotonically decreasing,*
- (a2) $\mathbb{E}[\|X^k\|_F] \leq 2(\gamma_u + \|X_r^*\|_F)$, *and*
- (a3) $\mathbb{E}[\mathcal{E}(U^k, U_r^*)] - \gamma_l = O(\frac{1}{k})$.

(2) *If $\mathcal{E}(U^0, U_r^*) \leq \gamma_l$, then $\mathbb{E}[\mathcal{E}(U^k, U_r^*)] \leq \gamma_l$ for any $k \in \mathbb{N}$.*

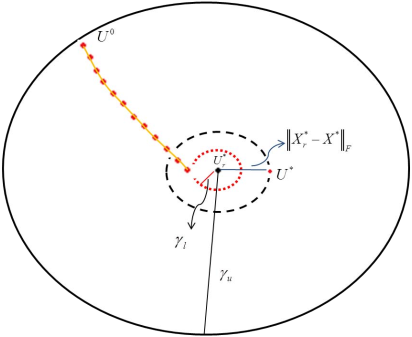
Interpretation of convergence of SFGD



(a) Convergence path of SFGD

- for SFGD
  - initialization lies in the ball $\mathcal{B}(X_r^*, \gamma_u)$
  - converges at a sublinear rate until reaching the ball $\mathcal{B}(X_r^*, \gamma_l)$
  - stagnates and never escapes from $\mathcal{B}(X_r^*, \gamma_l)$

Interpretation of convergence of SFGD



(b) Convergence path of SFGD

- for SFGD
  - initialization lies in the ball $\mathcal{B}(X_r^*, \gamma_u)$
  - converges at a sublinear rate until reaching the ball $\mathcal{B}(X_r^*, \gamma_l)$
  - stagnates and never escapes from $\mathcal{B}(X_r^*, \gamma_l)$
  - when $\|X_r^* - X^*\|_F^2 = 0$, then $\gamma_l = 0 \Rightarrow$ exact recovery

Linear convergence of SVRG

Theorem (Linear convergence of SVRG)

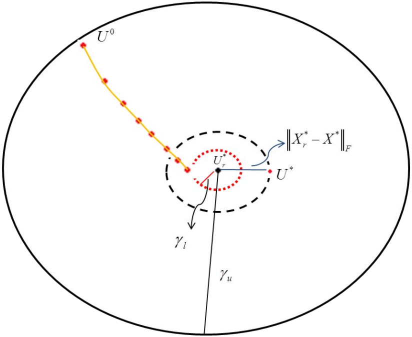
Let $\{\tilde{U}^k\}$ *be a sequence generated by the outer-loop of SVRG. Suppose that* $f : \mathbb{S}_+^p \to \mathbb{R}$ *is* L*-Lipschitz differentiable and* (μ, r)*-restricted strongly convex, and that Assumptions 1 and 2 hold, and that* $\eta_k \in (0, \eta_{\max}), \forall k \in \mathbb{N}$. *The following hold: if* $\gamma_l < \|\tilde{U}^0 - U_r^*\|_F^2 < \gamma_u$, *there hold*

(1) $\{\mathbb{E}[\mathcal{E}(\tilde{U}^k, U_r^*)]\}$ *is monotonically decreasing, and*

(2) $\mathbb{E}[\|\tilde{U}^k - U_r^*\|_F^2] - \gamma_l \leq \rho^k \cdot (\|\tilde{U}^0 - U_r^*\|_F^2 - \gamma_l)$, *for some* $0 < \rho < 1$.

In addition, if $\|\tilde{U}^0 - U_r^*\|_F^2 \leq \gamma_l$, *then* $\mathbb{E}[\|\tilde{U}^k - U_r^*\|_F^2] \leq \gamma_l$ *for any* $k \in \mathbb{N}$.

Interpretation of convergence of SVRG



(c) Convergence path of SVRG

- for SVRG
  - initialization lies in the ball $\mathcal{B}(X_r^*, \gamma_u)$
  - converges at a linear rate until reaching the ball $\mathcal{B}(X_r^*, \gamma_l)$
  - stagnates and never escapes from $\mathcal{B}(X_r^*, \gamma_l)$
  - when $\|X_r^* - X^*\|_F^2 = 0$, then $\gamma_l = 0 \Rightarrow$ exact recovery

Linear convergence of SVRG

Corollary (Linear convergence of SVRG with different step sizes)

Under assumptions of Theorem 2, then all claims in Theorem 2 hold, if one of the following conditions holds:

(1) *a fixed step size* η *is used with* $\eta \in (0, \eta_{\max})$*;*

(2) *the BB step size is used with* $m > \frac{1}{\mu \eta_{\max}}$*;*

(3) *the stabilized BB step size is used with* $m > \frac{1}{(\mu+\epsilon)\eta_{\max}}$ *for any* $\epsilon \geq 0$*.*

Comparisons with existing results

Algorithm	$\|X^* - X_r^*\|_F$	$\mathcal{E}(U^0, U_r^*)$	recovery error
FGD ([1])	$O(\frac{\sigma_r(X_r^*)}{\kappa^{1.5}\tau(X_r^*)})$	$O(\frac{\sigma_r(X_r^*)}{\kappa^2\tau^2(X_r^*)})$	$O(\frac{\kappa\|X_r^*-X^*\|_F^2}{\sigma_r(X_r^*)})$
SFGD (this talk)	$O(\frac{\sigma_r(X_r^*)}{\kappa})$	$O(\frac{\sigma_r(X_r^*)}{\kappa})$	$O(\frac{\kappa\|X_r^*-X^*\|_F^2}{\sigma_r(X_r^*)})$
SVRG (this talk)	$O(\frac{\sigma_r(X_r^*)}{\kappa})$	$O(\frac{\sigma_r(X_r^*)}{\kappa})$	$O(\frac{\kappa\|X_r^*-X^*\|_F^2}{\sigma_r(X_r^*)})$

Table: Comparisons on convergence results between (Bhojanapalli, Kyrillidis, and Sanghavi 2016) and this talk in the restricted strongly convex case. The second and third columns show the requirements on the rank-r approximation error and initialization to guarantee the convergence. The fourth column shows the recovery errors obtained by different algorithms, which are measured by $\lim_{k\to\infty}\mathcal{E}(U^k, U_r^*)$ for both FGD and SFGD or $\lim_{k\to\infty}\|U^k - U_r^*\|_F^2$ for SVRG.

Main proof techniques

- Perturbation theory for matrix eigenvalues
- Descent lemma for Lipschitz differentiable function
- Establishing a *second-order descent lemma* for both SFGD and SVRG shown latter

Summary

- Adapt SFGD and SVRG for fast solving the stochastic semidefinite optimization problem
- The considered algorithms have reduced computational cost and good scalability
- Establish their global convergence guarantees in the (restricted) strongly convex case
- Conduct a series of experiments to show their effectiveness