Sparse and Low-Rank Optimization for Dense Wireless Networks

Part II: Algorithms and Theory

Jun Zhang

HKUST

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Yuanming Shi

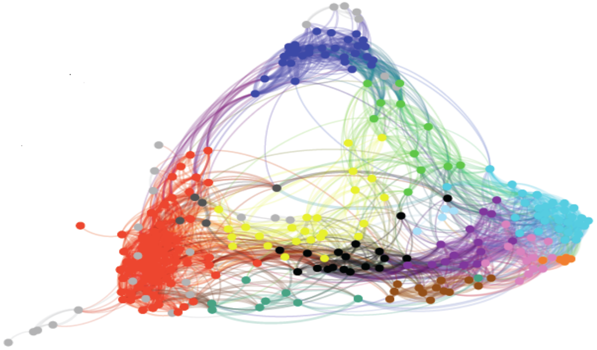
ShanghaiTech University

上海科技大学
ShanghaiTech University

Outline

- **Motivations**
  - Issues on computation, storage, nonconvexity,…
- **Two Vignettes:**
  - Large-scale convex optimization
    - Motivation: Why convex optimization?
    - Large-Scale Convex Optimization Algorithms
  - Scalable nonconvex optimization on manifolds
    - Motivation: Why Nonconvex Optimization?
    - Riemannian Optimization Algorithms
- **Future Directions**

Motivation: Optimization for Dense Wireless Networks

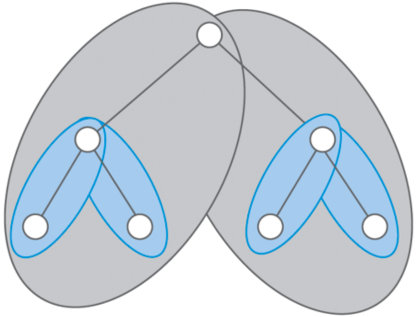


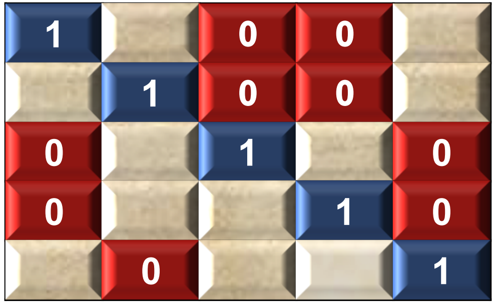
Motivations

- **The era of dense wireless networks**
  - Lead to new issues related to modeling and computing
- **Part I: Modeling issue**
  - Sparse and low-rank modeling frameworks for dense wireless networks
- **Part II: Computational issue**
  - Excessively large problem dimension, parameter size
  - Real-time communication requirements: polynomial-time algorithms often not fast enough
  - Non-convexity in general formulations

Issue A: Large-scale structured optimization

- Explosion in scale and complexity of the optimization problem in dense wireless networks

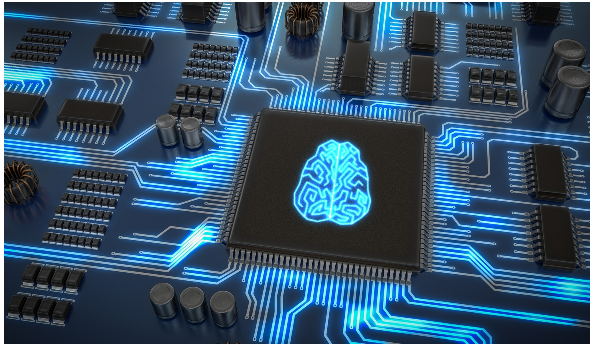




- **Questions:**
  - How to exploit the low-dimensional structures (e.g., sparsity and low-rankness) to assist efficient algorithms design?

Issue B: Real-time convex optimization

- Polynomial-time algorithms often not fast enough for real-time communications: parallel computing and approximations are essential



- **Questions:**
  - When is there a gap between polynomial-time and exponential-time algorithms?
  - How to reduce computational complexity while retaining optimality and accuracy?

Issue C: Scalable nonconvex optimization

- Nonconvex optimization may be super scary

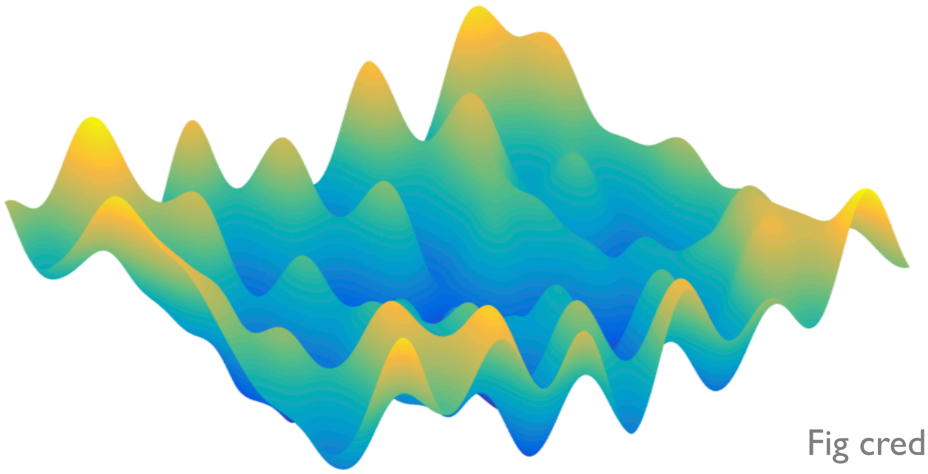
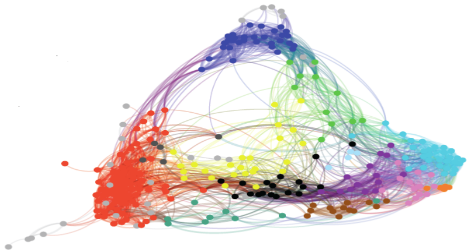


Fig credit: Chen

- **Question:**
  - ➢ How to exploit the geometry of nonconvex programs to guarantee optimality and enable scalability in computation and storage?

Vignettes A: Large-Scale Convex Optimization

1. Motivation: Why Convex Optimization?
   1) Theory I: Convexify sparse functions
   2) Theory II: Geometry of convex relaxation
2. Large-Scale Convex Optimization Algorithms
   1) Matrix stuffing for homogeneous self-dual embedding transforming
   2) Operator splitting for homogeneous self-dual embedding solving



Motivation: Why Convex Optimization?

Convex optimization – classical form

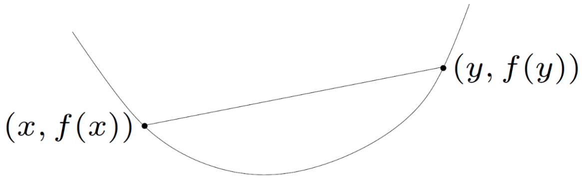
- Convex optimization problem in classical form

$$\begin{aligned} \underset{\boldsymbol{z}}{\text{minimize}} \quad & f_0(\boldsymbol{z};\boldsymbol{\alpha}) \\ \text{subject to} \quad & f_i(\boldsymbol{z};\boldsymbol{\alpha}) \leq g_i(\boldsymbol{z};\boldsymbol{\alpha}), i=1,\ldots,m \\ & u_i(\boldsymbol{z};\boldsymbol{\alpha}) = v_i(\boldsymbol{z};\boldsymbol{\alpha}), i=1,\ldots,p. \end{aligned}$$

  - f_i convex, g_i concave, u_i, v_i affine

- **Convex functions:** have nonnegative (upward) curvature

$$f_i(\theta\mathbf{x}+(1-\theta)\mathbf{y}) \leq \theta f_i(\mathbf{x}) + (1-\theta) f_i(\mathbf{y})$$



Convex optimization – conic form

- Convex optimization in *modern* canonical form

$$\begin{aligned} \underset{\boldsymbol{\nu},\boldsymbol{\mu}}{\text{minimize}} \quad & \mathbf{c}^T\boldsymbol{\nu} \\ \text{subject to} \quad & \mathbf{A}\boldsymbol{\nu} + \boldsymbol{\mu} = \mathbf{b} \\ & (\boldsymbol{\nu}, \boldsymbol{\mu}) \in \mathbb{R}^n \times \mathcal{K}. \end{aligned}$$

  - $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_q \in \mathbb{R}^m$ is a Cartesian product of closed convex cones
    - Nonnegative reals: $\mathbb{R}_+ = \{z \in \mathbb{R} | z \geq 0\}$ (LP)
    - Second-order cone: $\mathcal{Q}^d = \{(z, \mathbf{x}) \in \mathbb{R} \times \mathbb{R}^{d-1} | \|\mathbf{x}\| \leq z\}$ (SOCP)
    - Positive semidefinite cone: $\mathbf{S}_+^n = \{\boldsymbol{M} \in \mathbb{R}^{n\times n} | \boldsymbol{M} = \boldsymbol{M}^T, \boldsymbol{M} \succeq \mathbf{0}\}$ (SDP)

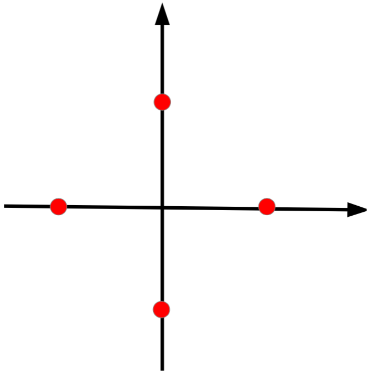
Why?

- **Theoretical foundations:** Beautiful, nearly complete theory
  - Duality, optimality conditions, convex geometry,…
- **Effective algorithms:** Convex optimization problems can be solved effectively with global optimality
  - Use generic methods for not huge problems: high level language support (CVX/CVXPY/Convex.jl) makes prototyping easy
  - Develop custom methods for huge problems (e.g., stochastic gradient descent)
- **Lots of applications:** Machine learning, signal processing, statistics, wireless communications, …

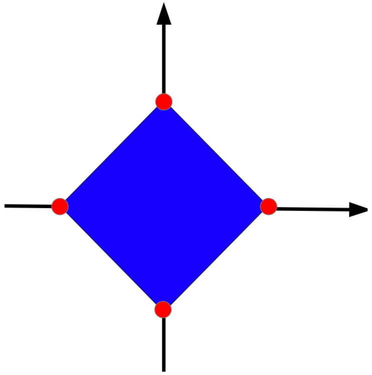
Theory I: Convexify Sparse Functions

Geometric view: sparsity

- Sparse approximation via convex hull $\mathcal{D} := \mathrm{conv}\left(\{\pm \boldsymbol{e}_i | i \in [n]\}\right)$



$\Rightarrow$



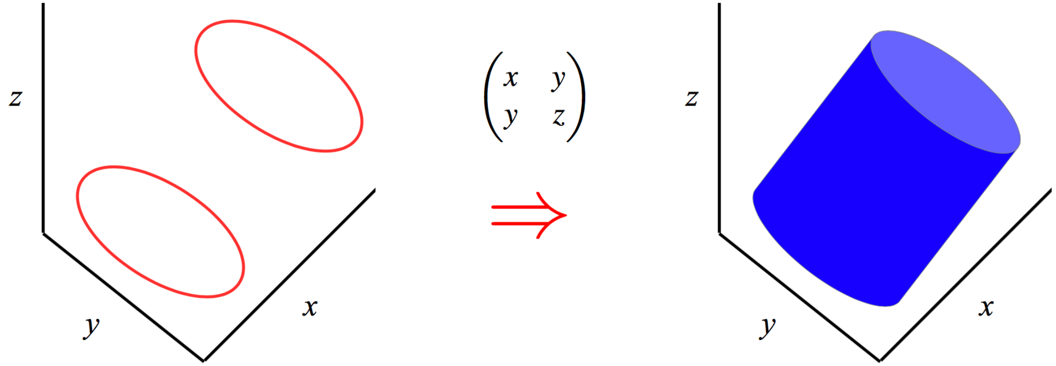
1-sparse vectors of
Euclidean norm 1

convex hull: ℓ_1-norm

$$\|\boldsymbol{z}\|_1 = \sum_{i=1}^{n} |z_i|$$

Geometric view: low-rank

- Low-rank approximation via convex hull



$$\begin{pmatrix} x & y \\ y & z \end{pmatrix} \Rightarrow$$

2x2 rank 1 symmetric matrices (normalized)

convex hull: nuclear norm

$$\|M\|_* = \sum_i \sigma_i(M)$$

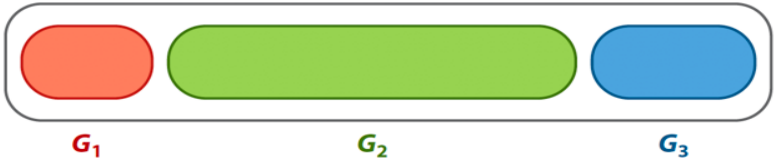
Structured sparsity

- ℓ_p-regularized combinatorial penalties of the form

$$F_p(\boldsymbol{z}) = \mu F(\mathrm{Supp}(\boldsymbol{z})) + \nu \|\boldsymbol{z}\|_p^p$$

  - μ and ν are positive scalar coefficients, $p \in (1, \infty]$
  - Positive-valued set-function F : control the structure of a model with non-zero patterns
  - ℓ_p-norm: control the magnitude of the coefficients

- **Examples:** 1) individual sparsity $F(A) = |A|$; 2) group sparsity



$$F(A) = \sum_{i=1}^{T} 1_{\{A \cap G_j \neq \emptyset\}}$$

Structure preserved by convex relaxations

- The tightest positively homogeneous lower bound ($1/p + 1/q = 1$)

$$F_h(\boldsymbol{z}) = (q\mu)^{1/q}(p\nu)^{1/p}Q(\boldsymbol{z})$$

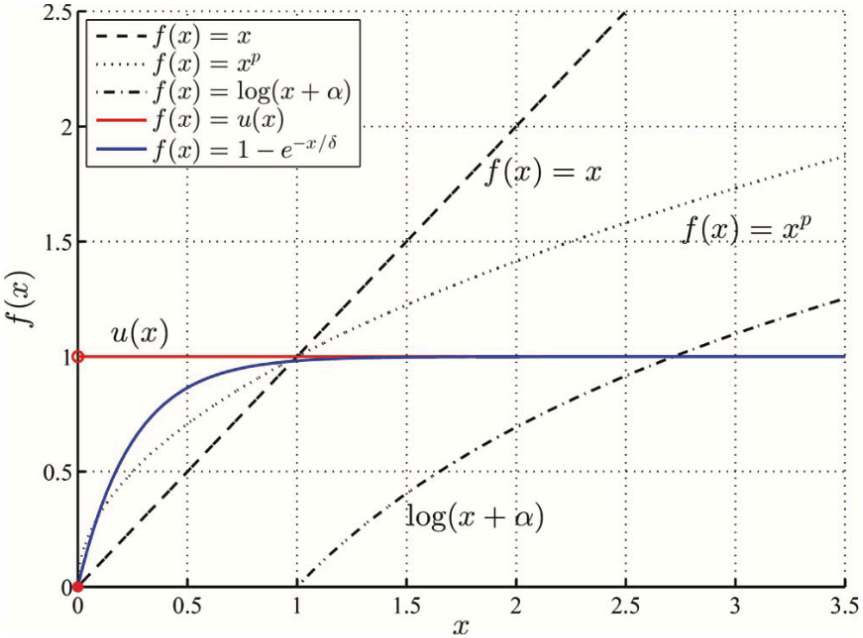
- The convex envelope of Q is given by the norm Ω_p with dual norm as

$$\Omega_p^*(\boldsymbol{s}) := \max_{A\subset V, A\neq\emptyset} \frac{\|\boldsymbol{s}_A\|_q}{F(A)^{1/q}}$$

- **Examples:**
  - 1) ℓ_1-norm (Lasso): If $F(A) = |A|$, then $\Omega_p(\boldsymbol{z}) = \|\boldsymbol{z}\|_1$, since $\Omega_p^*(\boldsymbol{s}) = \|\boldsymbol{s}\|_\infty$
  - 2) ℓ_p-norm: If $F(A) = 1_{\{A\neq\emptyset\}}$, then $\Omega_p(\boldsymbol{z}) = \|\boldsymbol{z}\|_p$, since $\Omega_p^*(\boldsymbol{s}) = \|\boldsymbol{s}\|_q$
  - 3) ℓ_1/ℓ_p-norm (Group Lasso): If $F(A) = \sum_{i=1}^T 1_{\{A\cap G_j\neq\emptyset\}}$, then $\Omega_p(\boldsymbol{z}) = \sum_{i=1}^T \|\boldsymbol{z}_{G_i}\|_p$

Enhance sparsity via sequential convex programming

- **Goal:** Provide tight approximation for sparsity function $u(x) = 1_{\{x \neq 0\}}$



Non-convex approximation:

$$\|\mathbf{x}\|_0 = \lim_{p \to 0} \|\mathbf{x}\|_p^p = \lim_{p \to 0} \sum |x_i|^p$$

At the origin, ℓ_0 function is better approximated by the log-sum function

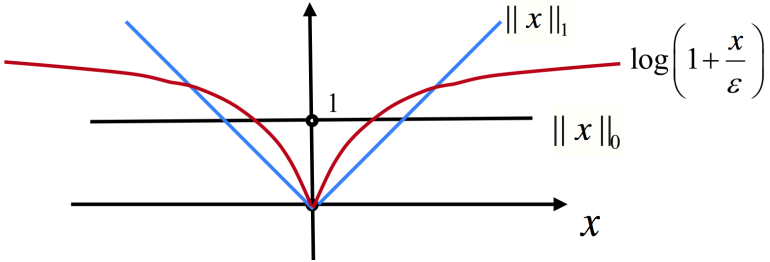
(check the slop at the origin)

Iterative reweighted-ℓ_1 algorithm (I)

- Consider the following (non-convex) sparse optimization problem

$$\underset{\boldsymbol{z}\in\mathbb{C}^n}{\text{minimize}} \quad \|\boldsymbol{z}\|_0 \quad \text{subject to} \quad \boldsymbol{z}\in\mathcal{C}, \boldsymbol{z}\succeq \mathbf{0}$$

- Approximate $\operatorname{card}(z)\approx\log(1+z/\epsilon)$, where $\epsilon>0, z\in\mathbb{R}_+$



- Using this approximation, we get (non-convex) problem

$$\underset{\boldsymbol{z}\in\mathbb{C}^n}{\text{minimize}} \quad \sum_{i=1}^{n}\log(1+z_i/\epsilon) \quad \text{subject to} \quad \boldsymbol{z}\in\mathcal{C}, \boldsymbol{z}\succeq \mathbf{0}$$

Iterative reweighted-ℓ_1 algorithm (II)

- Find a local solution by linearizing objective at current point

$$\sum_{i=1}^{n} \log(1 + z_i/\epsilon) \approx \sum_{i=1}^{n} \log(1 + z_i^{[k]}/\epsilon) + \sum_{i=1}^{n} \frac{z_i - z_i^{[k]}}{\epsilon + z_i^{[k]}}$$

- **Solve resulting convex problem**

$$\underset{\boldsymbol{z}\in\mathbb{C}^n}{\text{minimize}} \quad \sum_{i=1}^{n} \omega_i^{[k]} z_i \quad \text{subject to} \quad \boldsymbol{z} \in \mathcal{C}, \boldsymbol{z} \succeq \mathbf{0}$$

with $\omega_i^{[k]} = 1/(\epsilon + x_i^{[k]})$, to get next iterate

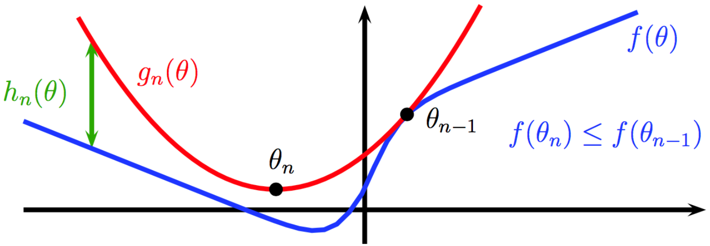
- Repeat until convergence to get a local solution

Iterative reweighted-ℓ_2 algorithm

- Adopt $\|\boldsymbol{z}\|_p (0 < p < 1)$ to approximate $\|\boldsymbol{z}\|_0$: $\|\boldsymbol{z}\|_0 = \lim_{p \to 0} \|\boldsymbol{z}\|_p^p$
- Solve the following (non-convex) smoothed ℓ_p-minimization problem

$$\underset{\boldsymbol{z} \in \mathbb{C}^n}{\text{minimize}} \quad \sum_{i=1}^{n} (z_i^2 + \epsilon^2)^{p/2} \quad \text{subject to} \quad \boldsymbol{z} \in \mathcal{C}$$

- Construct an upper bound for objective function $Q(\boldsymbol{z}; \boldsymbol{\omega}^{[k]}) := \sum_{i=1}^{n} \omega_i^{[k]} z_i^2$



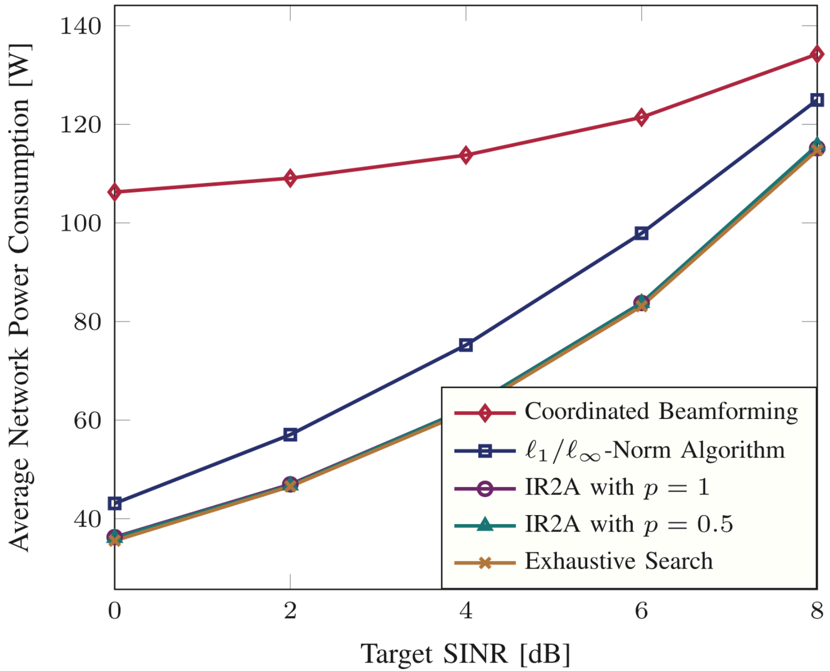
$$\omega_i^{[k]} = \frac{p}{2} \left[\left(z_i^{[k]} \right)^2 + \epsilon^2 \right]^{\frac{p}{2} - 1}$$

majorization-minimization algorithm

- Find the local solution via convex iterates $z^{[k+1]} := \arg\min_{\boldsymbol{z} \in \mathcal{C}} \ Q(\boldsymbol{z}; \boldsymbol{\omega}^{[k]})$

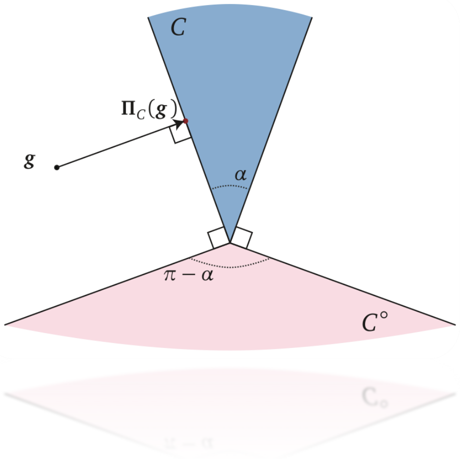
Simulation results: enhanced sparsity

- Network power minimization via group sparse beamforming



Group sparse beamforming for network power minimization (IR2A: iterative reweighted ℓ_2-algorithm)

Theory II: Geometry of Convex Relaxation



Linear inverse problems

- Let $\boldsymbol{x}^\natural \in \mathbb{R}^d$ be a structured, unknown vector
  - ➢ Group sparsity for user activity detection
- Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex function that reflects structure, e.g., ℓ_1-norm
- Let $\boldsymbol{A} \in \mathbb{R}^{m \times d}$ be a measurement operator
- **Observe** $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}^\natural$
- Find estimate $\hat{\boldsymbol{x}}$ by solving **convex program**

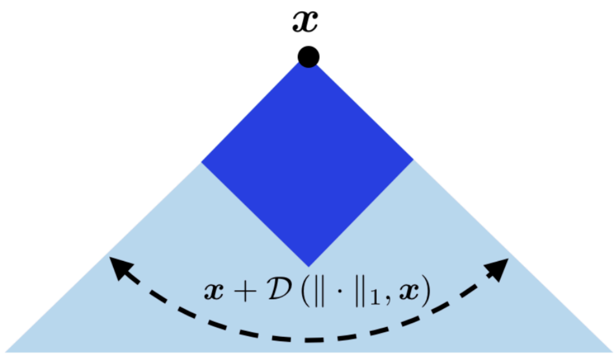
$$\text{minimize} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{z}$$

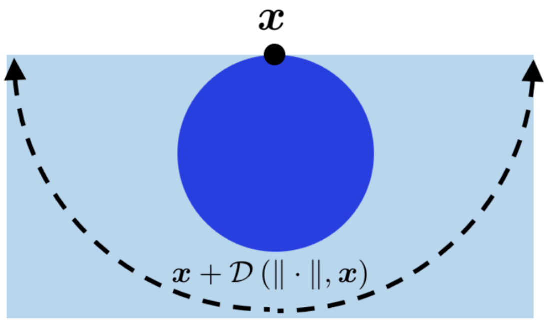
- **Hope:** $\hat{\boldsymbol{x}} = \boldsymbol{x}^\natural$

Geometry of linear inverse problems

- **Descent cone** of a function f at a point x is

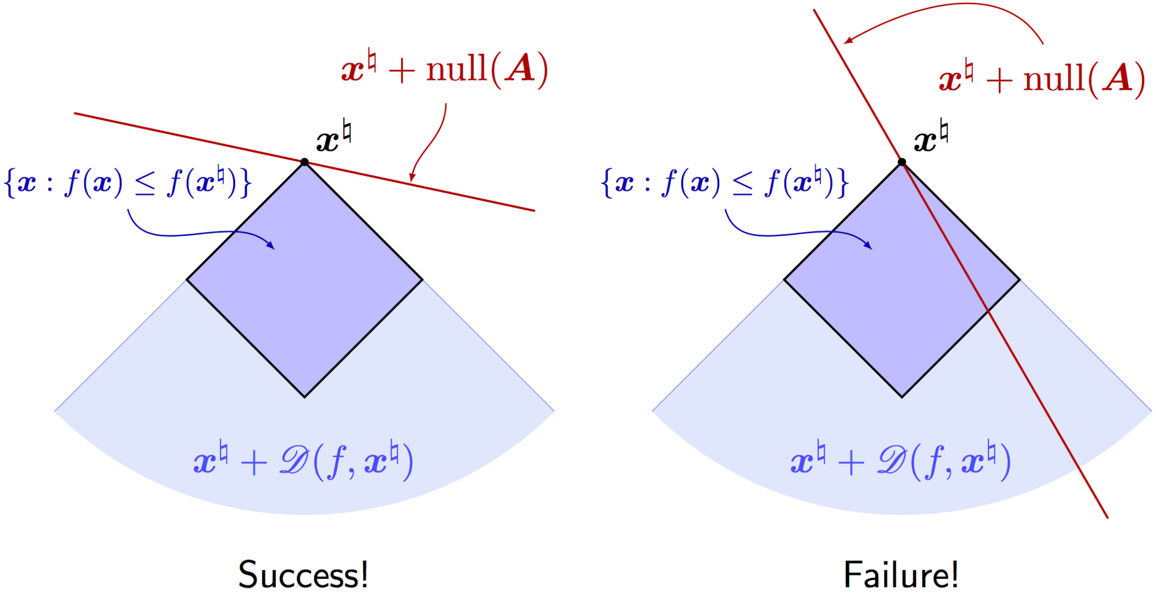
$$\mathscr{D}(f, \boldsymbol{x}) := \{\boldsymbol{d} : f(\boldsymbol{x} + \epsilon \boldsymbol{d}) \leq f(\boldsymbol{x}), \quad \text{for some } \epsilon > 0\}$$





References: Rockafellar 1970

Geometry of linear inverse problems



References: Candes–Romberg–Tao 2005, Rudelson–Vershynin 2006, Chandrasekaran et al. 2010, Amelunxen et al. 2013

Linear inverse problems with random data

- **Assume**
  - The vector $\boldsymbol{x}^\natural \in \mathbb{R}^d$ is unknown
  - The observation $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}^\natural$ where $\boldsymbol{A} \in \mathbb{R}^{m\times d}$ is standard normal
  - The vector $\hat{\boldsymbol{x}}$ solves

$$\text{minimize} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{z}$$

- **Then**

$$m \gtrsim \delta(\mathscr{D}(f, \boldsymbol{x}^\natural)) \quad \Longrightarrow \quad \hat{\boldsymbol{x}} = \boldsymbol{x}^\natural, \quad \text{w.h.p.}$$
$$m \lesssim \delta(\mathscr{D}(f, \boldsymbol{x}^\natural)) \quad \Longrightarrow \quad \hat{\boldsymbol{x}} \neq \boldsymbol{x}^\natural, \quad \text{w.h.p.}$$

statistical dimension [Amelunxen-McCoy-Tropp'13]

Examples for statistical dimension

- **Example 1:** ℓ_1-minimization for compressed sensing
  - $\boldsymbol{x}^{\natural} \in \mathbb{R}^d$ with s non-zero entries

$$\delta\left(\mathscr{D}(\|\cdot\|_1, \boldsymbol{x}^{\natural})\right) = \inf_{\tau \geq 0} \left\{ s(1+\tau^2) + (d-s)\sqrt{\frac{2}{\pi}} \int_{\tau}^{\infty} (z-\tau)^2 e^{-z^2} dz \right\}$$

- **Example 2:** ℓ_1/ℓ_2-minimization for massive device connectivity
  - $\boldsymbol{X}^{\natural} \in \mathbb{R}^{N \times M}$ with s non-zero rows

$$\delta\left(\mathscr{D}(\|\cdot\|_{2,1}, \boldsymbol{X}^{\natural})\right) = \inf_{\tau \geq 0} \left\{ s(M+\tau^2) + (N-s)\frac{2^{1-M/2}}{\Gamma(M/2)} \int_{\tau}^{\infty} (u-\tau)^2 u^{M-1} e^{-\frac{u^2}{2}} \mathrm{d}u \right\}$$

Numerical phase transition

- Compressed sensing with ℓ_1-minimization

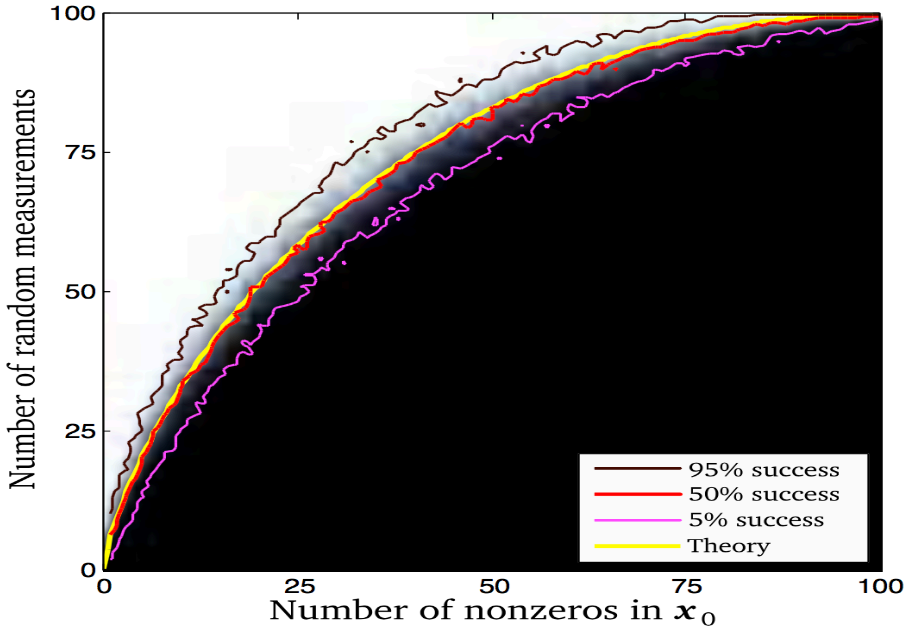
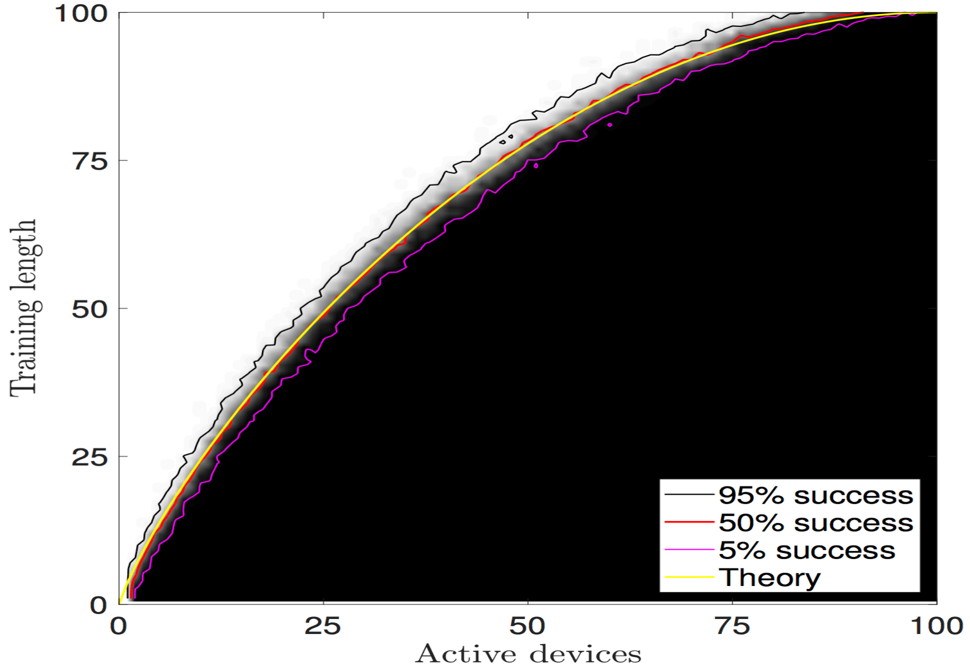


Figure credit: Amelunxen-McCoy-Tropp'13

Numerical phase transition

- User activity detection via ℓ_1/ℓ_2-minimization



group-structured sparsity estimation

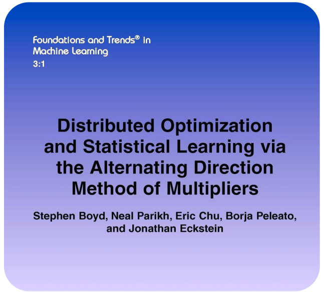
Summary of convex optimization

- Theoretical foundations for sparse optimization
  - Convex relaxation: convex hull, convex analysis
  - Fundamental bounds for convex methods: convex geometry, high-dimensional statistics
- Computational limits for (convexified) sparse optimization
  - Custom methods (e.g., stochastic gradient descent): not generalizable for complicated problems
  - Generic methods (e.g., CVX): not scalable to large problem sizes

Can we design a unified framework for general large-scale convex programs?

Large-Scale Convex Optimization Algorithms

$$\mathscr{P}_{\text{Original}} \longrightarrow \boxed{\text{Matrix Stuffing}} \xrightarrow{\mathscr{P}_{\text{HSD}}} \boxed{\text{ADMM Solver}} \xrightarrow{\mathbf{x}^\star}$$

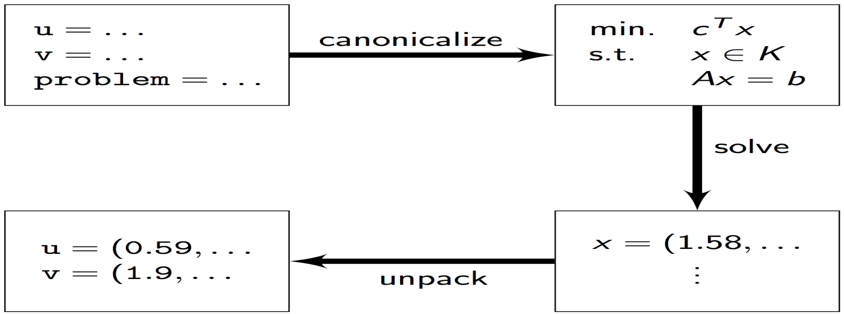

Foundations and Trends® in
Machine Learning
3:1

Distributed Optimization
and Statistical Learning via
the Alternating Direction
Method of Multipliers

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato,
and Jonathan Eckstein

Modeling languages

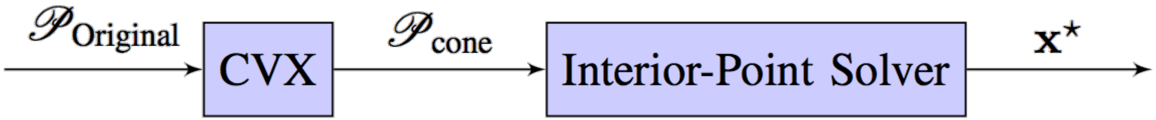
- High level language support for convex optimization
  - **Stage one:** problem description automatically transformed to standard form
  - **Stage two:** solved by standard solver, transformed back to original form



- **Implementation:** YALMIP, CVX (Matlab), CVXPY (Python), Convex.jl (Julia)

Modeling languages

- Disciplined convex programming framework [Grant & Boyd ’08]

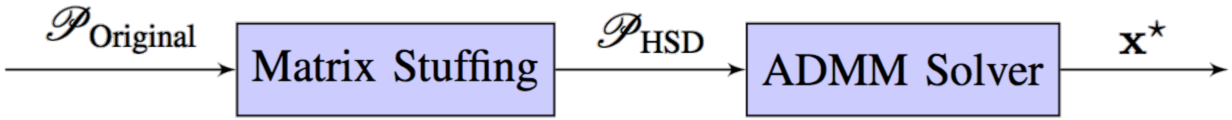


  - enable rapid prototyping (for small and medium problems)
  - widely used for applications with medium scale problems
  - shifts focus from *how to solve* to *what to solve*
  - **Large-scale problems:** time consuming in modeling phase & solving phase

- **Goal:** Scale to large problem sizes in modeling phase and solving phase

Large-scale convex optimization

- **Proposal:** Two-stage approach for large-scale convex optimization



fast homogeneous self-dual embedding (HSD) transformation

large-scale homogeneous self-dual embedding solving

  - Matrix stuffing: Fast homogeneous self-dual embedding (HSD) transformation
  - Operator splitting (ADMM): Large-scale homogeneous self-dual embedding

Stage I: Matrix Stuffing

Smith form reformulation

- **Goal:** transform the classical form to conic form

$$\begin{aligned} \underset{\boldsymbol{z}}{\text{minimize}} \quad & f_0(\boldsymbol{z};\boldsymbol{\alpha}) \\ \text{subject to} \quad & f_i(\boldsymbol{z};\boldsymbol{\alpha}) \leq g_i(\boldsymbol{z};\boldsymbol{\alpha}), \\ & u_i(\boldsymbol{z};\boldsymbol{\alpha}) = v_i(\boldsymbol{z};\boldsymbol{\alpha}). \end{aligned} \quad \Rightarrow \quad \begin{aligned} \underset{\boldsymbol{\nu},\boldsymbol{\mu}}{\text{minimize}} \quad & \mathbf{c}^T\boldsymbol{\nu} \\ \text{subject to} \quad & \mathbf{A}\boldsymbol{\nu} + \boldsymbol{\mu} = \mathbf{b}, \\ & (\boldsymbol{\nu},\boldsymbol{\mu}) \in \mathbb{R}^n \times \mathcal{K}. \end{aligned}$$

- **Key idea:** Introduce a new variable for each subexpression in classical form [Smith '96]
  - ➢ The Smith form is ready for standard cone programming transformation

Example

- Coordinated beamforming problem family

$$\mathscr{P}_{\text{Original}}: \text{minimize} \quad \|\boldsymbol{v}\|_2^2$$

$$\text{subject to} \quad \|\boldsymbol{D}_l \boldsymbol{v}\|_2 \leq \sqrt{P_l}, \forall l, \quad \text{Per-BS power constraint} \qquad (1)$$

$$\|\boldsymbol{C}_k \boldsymbol{v} + \boldsymbol{g}_k\|_2 \leq \beta_k \boldsymbol{r}_k^T \boldsymbol{v}, \forall k. \quad \text{QoS constraints} \qquad (2)$$

- Smith form reformulation

$$\mathcal{G}_1(l): \begin{cases} (y_0^l, \mathbf{y}_1^l) \in \mathcal{Q}^{KN_l+1} \\ y_0^l = \sqrt{P_l} \in \mathbb{R} \\ \mathbf{y}_1^l = \boldsymbol{D}_l \boldsymbol{v} \in \mathbb{R}^{KN_l} \end{cases}$$

Smith form for (1)

$$\mathcal{G}_2(k): \begin{cases} (t_0^k, \mathbf{t}_1^k) \in \mathcal{Q}^{K+1} \\ t_0^k = \beta_k \mathbf{r}_k^T \mathbf{v} \in \mathbb{R} \\ \mathbf{t}_1^k = \mathbf{t}_2^k + \mathbf{t}_3^k \in \mathbb{R}^{K+1} \\ \mathbf{t}_2^k = \mathbf{C}_k \mathbf{v} \in \mathbb{R}^{K+1} \\ \mathbf{t}_3^k = \mathbf{g}_k \in \mathbb{R}^{K+1} \end{cases}$$

Smith form for (2)

The Smith form is readily to be reformulated as the standard cone program

Reference: Grant-Boyd'08

Optimality condition

- KKT conditions (necessary and sufficient, assuming strong duality)
  - Primal feasibility: $\mathbf{A}\boldsymbol{\nu}^\star + \boldsymbol{\mu}^\star - \mathbf{b} = \mathbf{0}$
  - Dual feasibility: $\mathbf{A}^T\boldsymbol{\eta}^\star - \boldsymbol{\lambda}^\star + \mathbf{c} = \mathbf{0}$
  - Complementary slackness: $\mathbf{c}^T\boldsymbol{\nu}^\star + \mathbf{b}^T\boldsymbol{\eta}^\star = 0$ zero duality gap
  - Feasibility: $(\boldsymbol{\nu}^\star, \boldsymbol{\mu}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{\eta}^\star) \in \mathbb{R}^n \times \mathcal{K} \times \{0\}^n \times \mathcal{K}^*$

no solution if primal or dual problem infeasible/unbounded

Homogeneous self-dual (HSD) embedding

- **HSD embedding** of the primal-dual pair of transformed standard cone program (based on KKT conditions) [Ye et al. 94]

$$\boxed{\begin{array}{ll} \underset{\boldsymbol{\nu},\boldsymbol{\mu}}{\text{minimize}} & \mathbf{c}^T\boldsymbol{\nu} \\ \text{subject to} & \mathbf{A}\boldsymbol{\nu}+\boldsymbol{\mu}=\mathbf{b} \\ & (\boldsymbol{\nu},\boldsymbol{\mu})\in\mathbb{R}^n\times\mathcal{K}. \end{array}} + \boxed{\begin{array}{ll} \underset{\boldsymbol{\eta},\boldsymbol{\lambda}}{\text{maximize}} & -\mathbf{b}^T\boldsymbol{\eta} \\ \text{subject to} & -\mathbf{A}^T\boldsymbol{\eta}+\boldsymbol{\lambda}=\mathbf{c} \\ & (\boldsymbol{\lambda},\boldsymbol{\eta})\in\{0\}^n\times\mathcal{K}^* \end{array}} \Longrightarrow \boxed{\begin{array}{ll} \mathscr{F}_{\mathrm{HSD}}: \text{find } (\mathbf{x},\mathbf{y}) & \\ \text{subject to } \mathbf{y}=\mathbf{Q}\mathbf{x} & \\ \mathbf{x}\in\mathcal{C},\mathbf{y}\in\mathcal{C}^* & \end{array}}$$

$$\underbrace{\begin{bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \kappa \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{A}^T & \mathbf{c} \\ -\mathbf{A} & \mathbf{0} & \mathbf{b} \\ -\mathbf{c}^T & -\mathbf{b}^T & \mathbf{0} \end{bmatrix}}_{\mathbf{Q}} \underbrace{\begin{bmatrix} \boldsymbol{\nu} \\ \boldsymbol{\eta} \\ \tau \end{bmatrix}}_{\mathbf{x}}$$

finding a *nonzero* solution

- This feasibility problem is homogeneous and self-dual

Recovering solution or certificates

- Any HSD solution $(\boldsymbol{\nu}, \boldsymbol{\mu}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \tau, \kappa)$ falls into one of three cases:
  - Case 1: $\tau > 0,\ \kappa = 0$, then $\hat{\boldsymbol{\nu}} = \boldsymbol{\nu}/\tau, \hat{\boldsymbol{\eta}} = \boldsymbol{\eta}/\tau, \hat{\boldsymbol{\mu}} = \boldsymbol{\mu}/\tau$ is a solution
  - Case 2: $\tau = 0,\ \kappa > 0$, implies $\mathbf{c}^T\boldsymbol{\nu} + \mathbf{b}^T\boldsymbol{\eta} < 0$
    - If $\mathbf{b}^T\boldsymbol{\eta} < 0$, then $\hat{\boldsymbol{\eta}} = \boldsymbol{\eta}/(-\mathbf{b}^T\boldsymbol{\eta})$ certifies primal infeasibility
    - If $\mathbf{c}^T\boldsymbol{\nu} < 0$, then $\hat{\boldsymbol{\nu}} = \boldsymbol{\nu}/(-\mathbf{c}^T\hat{\boldsymbol{\nu}})$ certifies dual infeasibility
  - Case 3: $\tau = \kappa = 0$, nothing can be said about original problem
- **HSD embedding:** 1) obviates need for phase I / phase II solves to handle infeasibility/unboundedness; 2) used in all interior-point cone solvers

Matrix stuffing for fast transformation

- HSD embedding of the primal-dual pair of standard cone program

$$\underbrace{\begin{bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \kappa \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{A}^T & \mathbf{c} \\ -\mathbf{A} & \mathbf{0} & \mathbf{b} \\ -\mathbf{c}^T & -\mathbf{b}^T & \mathbf{0} \end{bmatrix}}_{\mathbf{Q}} \underbrace{\begin{bmatrix} \boldsymbol{\nu} \\ \boldsymbol{\eta} \\ \tau \end{bmatrix}}_{\mathbf{x}}$$

- **Matrix stuffing:** fast HSD embedding transformation
  - Generate and keep the structure $\mathbf{Q}$
  - Copy problem instance parameters to update the entries in $\mathbf{Q}$

Stage II: Operator Splitting

$$\begin{aligned} \mathscr{F}_{\text{HSD}} : \text{find} \quad & (\mathbf{x}, \mathbf{y}) \\ \text{subject to} \quad & \mathbf{y} = \mathbf{Q}\mathbf{x} \\ & \mathbf{x} \in \mathcal{C}, \mathbf{y} \in \mathcal{C}^* \end{aligned}$$

Alternating direction method of multipliers

- **ADMM:** an operator splitting method solving convex problems in form

$$\mathscr{P}_{\text{ADMM}}: \text{minimize} \quad f(\mathbf{x}) + g(\boldsymbol{z}) \quad \text{subject to} \quad \mathbf{x} = \boldsymbol{z}$$

  - f, g convex, not necessarily smooth, can take infinite values

- The basic ADMM algorithm [Boyd et al., FTML 11]

$$\mathbf{x}^{[k+1]} = \arg\min_{\mathbf{x}} \left(f(\mathbf{x}) + (\rho/2)\|\mathbf{x} - \boldsymbol{z}^{[k]} - \lambda^{[k]}\|_2^2 \right)$$
$$\boldsymbol{z}^{[k+1]} = \arg\min_{\boldsymbol{z}} \left(g(\boldsymbol{z}) + (\rho/2)\|\mathbf{x}^{[k+1]} - \boldsymbol{z} - \lambda^{[k]}\|_2^2 \right)$$
$$\lambda^{[k+1]} = \lambda^{[k]} - \mathbf{x}^{[k+1]} + \boldsymbol{z}^{[k+1]}$$

  - $\rho > 0$ is a step size; λ is the dual variable associated the constraint

Alternating direction method of multipliers

- **Convergence of ADMM:** Under benign conditions ADMM guarantees
  - $f(\mathbf{x}^k) + g(\boldsymbol{z}^k) \to p^\star$
  - $\lambda^k \to \lambda^\star$, an optimal dual variable
  - $\mathbf{x}^k - \boldsymbol{z}^k \to 0$
- Same as many other operator splitting methods for consensus problem, e.g., Douglas-Rachford method
- **Pros:** 1) with good robustness of method of multipliers; 2) can support decomposition

Operator splitting

- Transform HSD embedding $\mathscr{F}_{\mathrm{HSD}}$ in ADMM form: Apply the operating splitting method (ADMM)

$$\mathscr{P}_{\mathrm{ADMM}} : \underset{\mathbf{x},\tilde{\mathbf{x}},\mathbf{y},\tilde{\mathbf{y}}}{\text{minimize}} \quad I_{\mathcal{C}\times\mathcal{C}^*}(\mathbf{x},\mathbf{y}) + I_{\mathbf{Q}\tilde{\mathbf{x}}=\tilde{\mathbf{y}}}(\tilde{\mathbf{x}},\tilde{\mathbf{y}})$$
$$\text{subject to} \quad (\mathbf{x},\mathbf{y}) = (\tilde{\mathbf{x}},\tilde{\mathbf{y}})$$

- **Final algorithm**

$$\begin{aligned}
\tilde{\mathbf{x}}^{[i+1]} &= (\mathbf{I}+\mathbf{Q})^{-1}(\mathbf{x}^{[i]}+\mathbf{y}^{[i]}) && \text{subspace projection}\\
\mathbf{x}^{[i+1]} &= \Pi_{\mathcal{C}}(\tilde{\mathbf{x}}^{[i+1]}-\mathbf{y}^{[i]}) && \text{parallel cone projection}\\
\mathbf{y}^{[i+1]} &= \mathbf{y}^{[i]}-\tilde{\mathbf{x}}^{[i+1]}+\mathbf{x}^{[i+1]} && \text{computationally trivial}
\end{aligned}$$

Parallel cone projection

- **Proximal algorithms** for parallel cone projection [Parikn & Boyd, FTO 14]
  - Projection onto the second-order cone: $\mathcal{Q}^d = \{(z, \mathbf{x}) \in \mathbb{R} \times \mathbb{R}^{d-1} | \|\mathbf{x}\| \leq z\}$

$$\Pi_{\mathcal{C}}(\boldsymbol{\omega}, \tau) = \begin{cases} 0, \|\boldsymbol{\omega}\|_2 \leq -\tau \\ (\boldsymbol{\omega}, \tau), \|\boldsymbol{\omega}\|_2 \leq \tau \\ (1/2)(1 + \tau/\|\boldsymbol{\omega}\|_2)(\boldsymbol{\omega}, \|\boldsymbol{\omega}\|_2), \|\boldsymbol{\omega}\|_2 \geq |\tau|. \end{cases}$$

    - Closed-form, computationally scalable (we mainly focus on SOCP)
  - Projection onto positive semidefinite cone: $\mathbf{S}_+^n = \{\boldsymbol{M} \in \mathbb{R}^{n \times n} | \boldsymbol{M} = \boldsymbol{M}^T, \boldsymbol{M} \succeq \mathbf{0}\}$

$$\Pi_{\mathcal{C}}(\boldsymbol{V}) = \sum_{i=1}^n (\lambda_i)_+ \boldsymbol{u}_i \boldsymbol{u}_i^T$$

    - SVD is computationally expensive

Numerical results

- Power minimization coordinated beamforming problem

Network Size (L=K)		20	50	100	150
CVX+SDPT3	Modeling Time [sec]	**0.7563**	4.4301	N/A	N/A
	Solving Time [sec]	4.2835	**326.2513**	N/A	N/A
	Objective [W]	12.2488	6.5216	N/A	N/A
Matrix Stuffing+ADMM	Modeling Time [sec]	**0.0128**	0.2401	2.4154	9.4167
	Solving Time [sec]	0.1009	**2.4821**	23.8088	81.0023
	Objective [W]	12.2523	6.5193	3.1296	2.0689

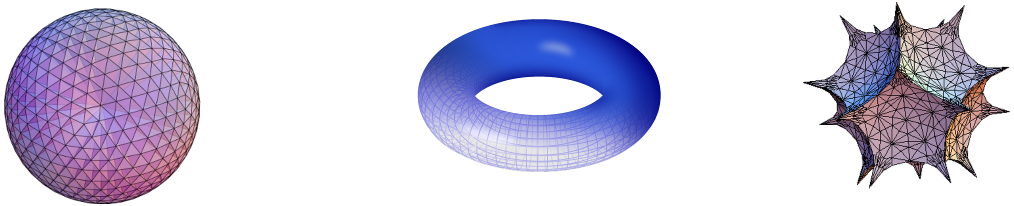
Matrix stuffing can speedup **60x** over CVX

ADMM can speedup **130x** over the interior-point method

[Ref] Y. Shi, J. Zhang, B. O'Donoghue, and K. B. Letaief, "Large-scale convex optimization for dense wireless cooperative networks," IEEE Trans. Signal Process., vol. 63, no. 18, pp. 4729-4743, Sept. 2015. **(The 2016 IEEE Signal Processing Society Young Author Best Paper Award)**

Vignette B: Scalable Optimization on Manifolds

1. Motivation: Why Nonconvex Optimization?
   1) Geometry of Nonconvex Optimization
2. Riemannian Optimization Algorithms



Optimization over Riemannian Manifolds (non-Euclidean geometry)

*Motivation: Why **Nonconvex** Optimization?*

Low-rank matrix optimization

- Rank-constrained matrix optimization problem

$$\underset{\boldsymbol{M}\in\mathbb{R}^{n\times n}}{\text{minimize}} \quad f(\mathcal{A}(\boldsymbol{M})) \quad \text{subject to} \;\; \text{rank}(\boldsymbol{M}) = r$$

  - $\mathcal{A} : \mathbb{R}^{n\times n} \to \mathbb{R}^d$ is a real linear map on $n \times n$ matrices
  - $f : \mathbb{R}^d \to \mathbb{R}$ is convex and differentiable
  - A prevalent **model** in signal processing, statistics, and machine learning
- **Challenge I:** Reliably solve the low-rank matrix problem at scale
- **Challenge II:** Develop optimization algorithms with optimal storage $\Theta(rn)$

A brief biased history of convex methods

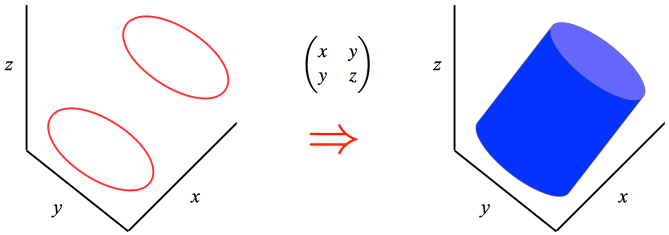
- **1990s:** Interior-point methods (computationally expensive)
  - Storage cost $\boldsymbol{\Theta}(n^4)$ for Hessian
- **2000s:** Convex first-order methods
  - (Accelerated) proximal gradient, spectral bundle methods, and others
  - Store matrix variable $\boldsymbol{\Theta}(n^2)$
- **2008-Present:** Storage-efficient convex first-order methods
  - Conditional gradient method (CGM) and extensions
  - Store matrix in low-rank form $\mathcal{O}(tn)$ after t iterations: no storage guarantees

Interior-point: Nemirovski & Nesterov 1994; ... **First-order:** Rockafellar 1976; Helmberg & Rendl 1997; Auslender & Teboulle 2006; ... **CGM:** Frank & Wolfe 1956; Levitin & Poljak 1967; Jaggi 2013; ...

Convexity: Why bother?

- **Convex relaxation fails:** always return the identity matrix!

$$\begin{aligned} \underset{\boldsymbol{M} \in \mathbb{C}^{K \times K}}{\text{minimize}} \quad & \|\boldsymbol{M}\|_* \\ \text{subject to} \quad & M_{ii} = 1, i = 1, \ldots, K \\ & M_{ij} = 0, \forall (i,j) \in \mathcal{S} \end{aligned}$$



  - **Fact:** $\mathrm{Trace}(\boldsymbol{M}) \leq \|\boldsymbol{M}\|_*$

- **The dilemma:** Convex methods have slow memory hogs, high computational complexity, sometimes fail

Can we solve the nonconvex matrix optimization problem directly?

Recent advances in nonconvex optimization

- **2009–Present: Nonconvex heuristics**
  - Burer–Monteiro factorization idea + various nonlinear programming methods
  - Store low-rank matrix factors $\Theta(rn)$
- **Guaranteed solutions:** Global optimality with statistical assumptions
  - Matrix completion/recovery: [Sun-Luo'14], [Chen-Wainwright'15], [Ge-Lee-Ma'16],…
  - Phase retrieval: [Candes et al., 15], [Chen-Candes' 15], [Sun-Qu-Wright'16]
  - Community detection/phase synchronization [Bandeira-Boumal-Voroninski'16], [Montanari et al., 17],…

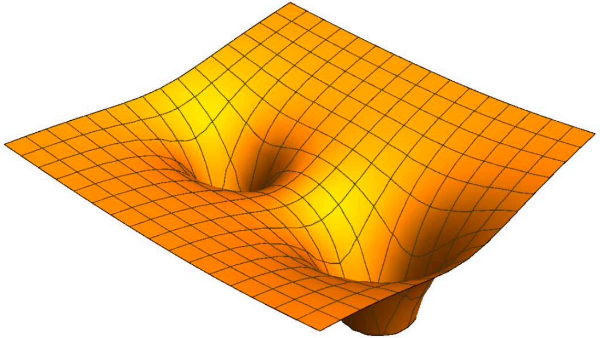
When are nonconvex optimization problems not scary?

Geometry of Nonconvex Optimization

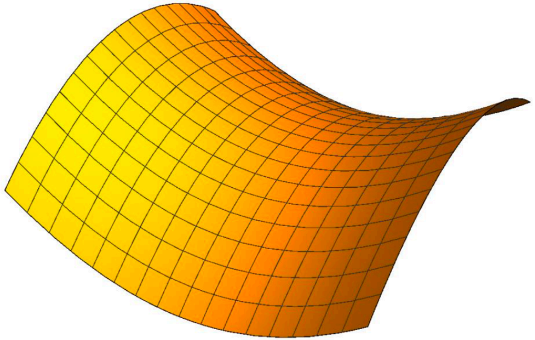
First-order stationary points

- Saddle points and local minima:

$$\lambda_{\min}(\nabla^2 f(\boldsymbol{z})) \begin{cases} > 0 & \text{local minimum} \\ = 0 & \text{local minimum or saddle point} \\ < 0 & \text{strict saddle point} \end{cases}$$



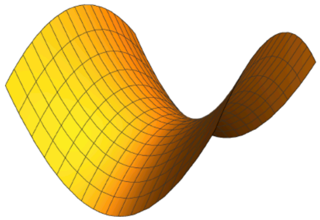
Local minima



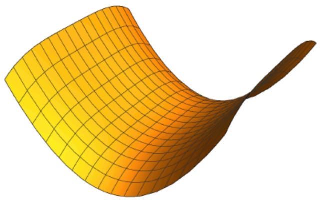
Saddle points/local maxima

First-order stationary points

- **Applications:** PCA, matrix completion, dictionary learning etc.
  - **Local minima:** Either all local minima **are** global minima or all local minima **as good as** global minima
  - **Saddle points:** **Very poor** compared to global minima; **Several** such points



Strict saddle point



Non-strict saddle point

- **Bottomline:** Local minima much more desirable than saddle points

Summary of motivations

- **Convex methods:**
  - Slow memory hogs
  - Convex relaxation fails sometimes, e.g., topological interference alignment
  - High computational complexity, e.g., eigenvalue decomposition
- **Nonconvex methods:** fast, lightweight
  - Under certain statistical models with benign global geometry: no spurious local optima

How to escape saddle points efficiently?

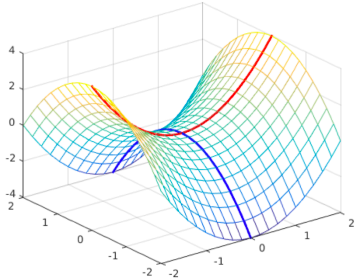
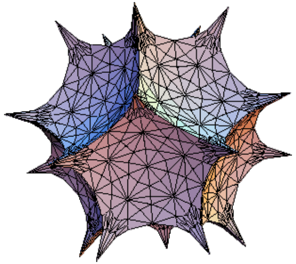


Fig credit: Sun, Qu & Wright

Riemannian Optimization Algorithms

Escape saddle points via manifold optimization

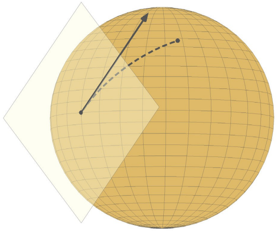


What is manifold optimization?

- Manifold (or manifold-constrained) optimization problem

$$\underset{\boldsymbol{M}\in\mathbb{C}^{m\times n}}{\text{minimize}} \quad f(\boldsymbol{M}) \quad \text{subject to} \quad \boldsymbol{M}\in\mathcal{M}$$

  - $f:\mathbb{R}^{m\times n}\to\mathbb{R}$ is a smooth function
  - $\mathcal{M}$ is a **Riemannian manifold:** spheres, orthonormal bases (Stiefel), rotations, positive definite matrices, ***fixed-rank matrices***, Euclidean distance matrices, semidefinite fixed-rank matrices, linear subspaces (Grassmann), phases, essential matrices, fixed-rank tensors, Euclidean spaces...



Escape saddle points via manifold optimization

- Convergence guarantees for Riemannian **trust regions**
  - Global convergence to **second-order critical points**
  - Quadratic convergence rate locally
  - Reach ϵ-second order stationary point $\|\mathrm{grad}\, f(\boldsymbol{z})\| \leq \epsilon$ and $\nabla^2 f(\boldsymbol{z}) \succeq -\epsilon \boldsymbol{I}$ in $\mathcal{O}(1/\epsilon^3)$ iterations under Lipschitz assumptions [Cartis & Absil'16]

Escape *strict* saddle points via finding second-order stationary point

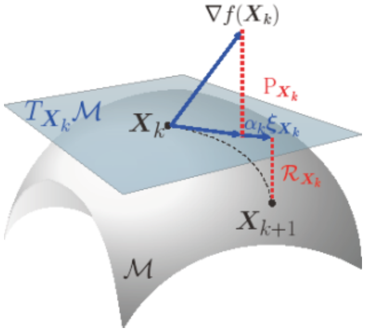
- **Other approaches:** Gradient descent by adding noise [Ge et al., 2015], [Jordan et al., 17] (slow convergence rate in general)

Recent applications of manifold optimization

- Matrix/tensor completion/recovery: [Vandereycken'13], [Boumal-Absil'15], [Kasai-Mishra'16],…
- Gaussian mixture models: [Hosseini-Sra'15], Dictionary learning: [Sun-Qu-Wright'17], Phase retrieval: [Sun-Qu-Wright'17],…
- Phase synchronization/community detection: [Boumal'16], [Bandeira-Boumal-Voroninski'16],…
- **Wireless transceivers design:** [Shi-Zhang-Letaief'16], [Yu-Shen-Zhang-K. B. Letaief'16], [Shi-Mishra-Chen'16],…

The power of manifold optimization paradigms

- Generalize Euclidean gradient (Hessian) to *Riemannian gradient (Hessian)*



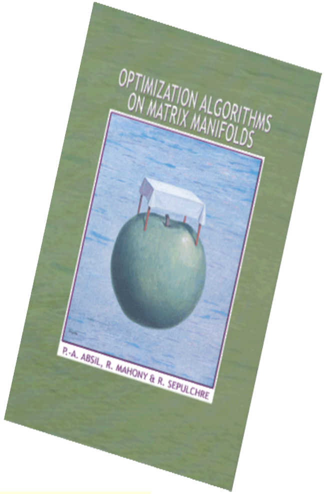
$$\underbrace{\nabla_{\mathcal{M}} f(\mathbf{X}^{(k)})}_{\text{Riemannian Gradient}} = P_{\mathbf{X}^{(k)}}(\underbrace{\nabla f(\mathbf{X}^{(k)})}_{\text{Euclidean Gradient}})$$

$$\mathbf{X}^{(k+1)} = \underbrace{\mathcal{R}_{\mathbf{X}^{(k)}}}_{\text{Retraction Operator}}(-\alpha^{(k)} \nabla_{\mathcal{M}} f(\mathbf{X}^{(k)}))$$

- We need Riemannian geometry: 1) linearize search space $\mathcal{M}$ into a tangent space $T_{\mathbf{X}}\mathcal{M}$; 2) pick a metric on $T_{\mathbf{X}}\mathcal{M}$ to give intrinsic notions of gradient and Hessian

An excellent book

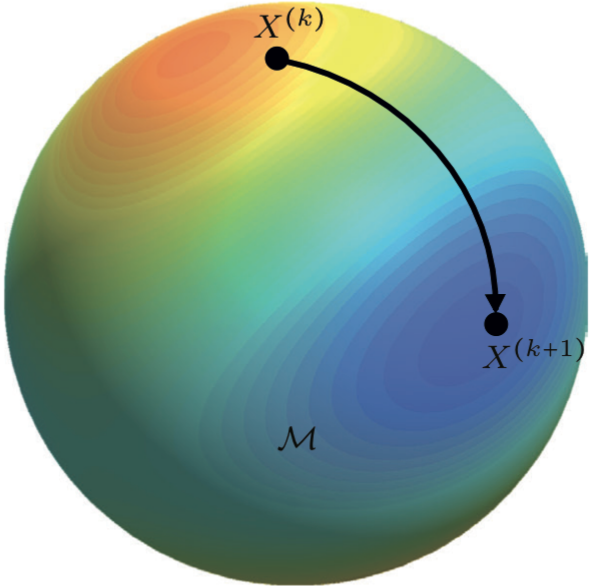
Optimization algorithms on matrix manifolds



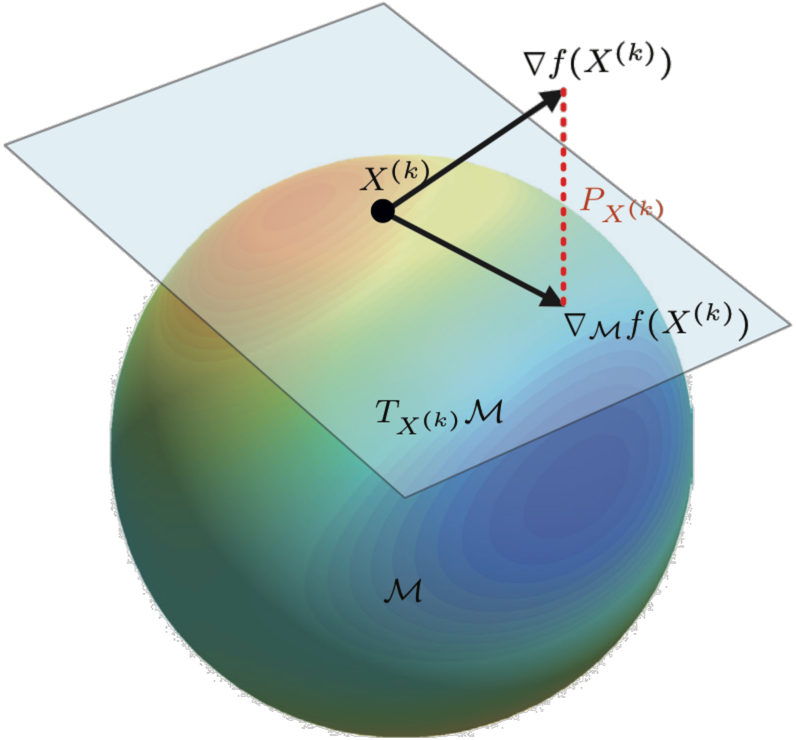
A Matlab toolbox

Taking A Close Look at ***Gradient Descent***

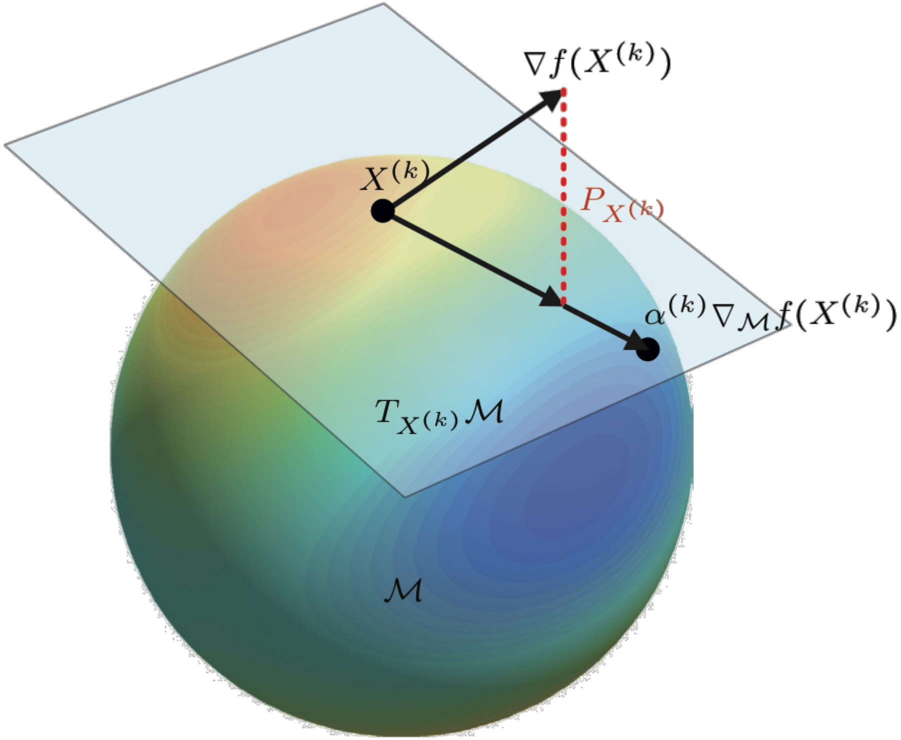
Optimization on the manifold: main idea



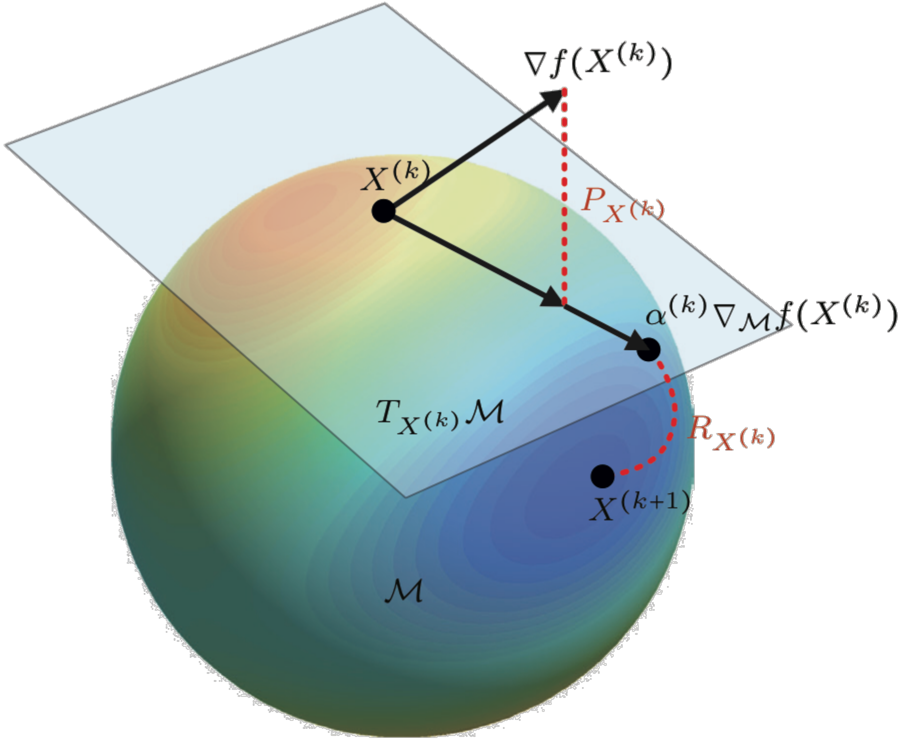
Optimization on the manifold: main idea



Optimization on the manifold: main idea



Optimization on the manifold: main idea



Example: Rayleigh quotient

- Optimization over (sphere) manifold $\mathbb{S}^{n-1} = \{x \in \mathbb{R}^n : x^T x = 1\}$

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; f(x) = -x^T A x \quad \text{subject to} \;\; x^T x = 1$$

  - The cost function is smooth on $\mathbb{S}^{n-1}$, symmetric matrix $A \in \mathbb{R}^{n \times n}$

- Step 1: Compute the Euclidean gradient in $\mathbb{R}^n$

$$\nabla f(x) = -2Ax$$

- Step 2: Compute the Riemannian gradient on $\mathbb{S}^{n-1}$ via projecting $\nabla f(x)$ to the tangent space using the orthogonal projector $\mathrm{Proj}_x u = (I - xx^T)u$

$$\mathrm{grad} f(x) = \mathrm{Proj}_x \nabla f(x) = -2(I - xx^T)Ax$$

Example: Generalized low-rank optimization

- Generalized low-rank optimization for topological interference alignment via Riemannian optimization

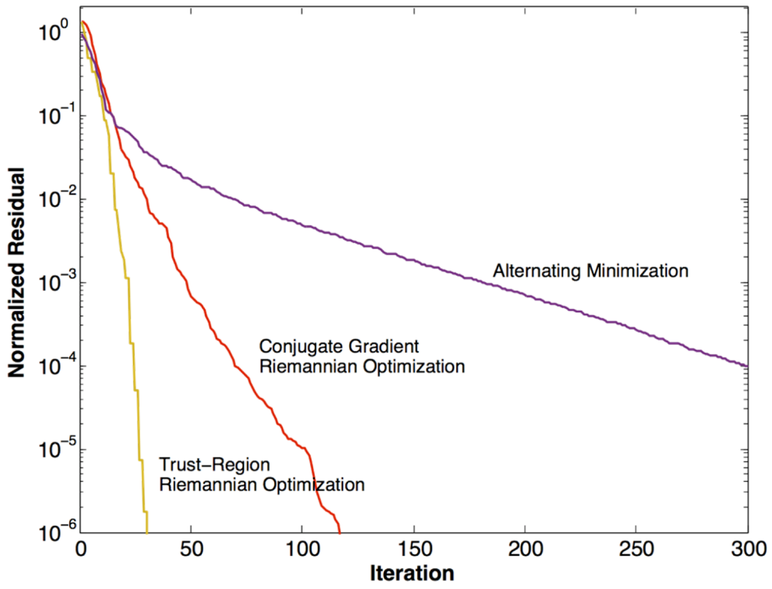
$$\underset{\boldsymbol{M}\in\mathbb{C}^{m\times n}}{\text{minimize}} \quad f(\boldsymbol{M}), \qquad \text{subject to} \quad \text{rank}(\boldsymbol{M}) = r$$

OPTIMIZATION-RELATED INGREDIENTS FOR PROBLEM $\mathscr{P}_r$

	$\mathscr{P}_r$: $\text{minimize}_{\mathbf{X}\in\mathcal{M}_r}\ f(\mathbf{X})$
Matrix representation of an element $\mathbf{X}\in\mathcal{M}_r$	$\mathbf{X} = (\mathbf{U}, \boldsymbol{\Sigma}, \mathbf{V})$
Computational space $\mathcal{M}_r$	$\mathrm{St}(r, M) \times \mathrm{GL}(r) \times \mathrm{St}(r, M)$
Quotient space	$\mathrm{St}(r, M) \times \mathrm{GL}(r) \times \mathrm{St}(r, M)/(\mathcal{O}(r) \times \mathcal{O}(r))$
Metric $g_{\mathbf{X}}(\xi_{\mathbf{X}}, \zeta_{\mathbf{X}})$ for $\xi_{\mathbf{X}}, \zeta_{\mathbf{X}} \in T_{\mathbf{X}}\mathcal{M}_r$	$g_{\mathbf{X}}(\xi_{\mathbf{X}}, \zeta_{\mathbf{X}}) = \langle \xi_U, \zeta_U \Sigma\Sigma^T\rangle + \langle \xi_\Sigma, \zeta_\Sigma\rangle + \langle \xi_V, \zeta_V \Sigma^T\Sigma\rangle$
Riemannian gradient $\mathrm{grad}_{\mathbf{X}} f$	$\mathrm{grad}_{\mathbf{X}} f = (\xi_U, \xi_\Sigma, \xi_V)$ (30)
Riemannian Hessian $\mathrm{Hess}_{\mathbf{X}} f[\xi_{\mathbf{X}}]$	$\mathrm{Hess}_{\mathbf{X}} f[\xi_{\mathbf{X}}] = \Pi_{\mathcal{H}_{\mathbf{X}}\mathcal{M}_r}(\nabla_{\xi_{\mathbf{X}}} \mathrm{grad}_{\mathbf{X}} f)$ (40)
Retraction $\mathcal{R}_{\mathbf{X}}(\xi_{\mathbf{X}}) : \mathcal{H}_{\mathbf{X}}\mathcal{M}_r \to \mathcal{M}_r$	$(\mathrm{uf}(\mathbf{U} + \xi_{\mathbf{X}}), \boldsymbol{\Sigma} + \xi_\Sigma, \mathrm{uf}(\mathbf{V} + \xi_V))$

Convergence rates

- Optimize over fixed-rank matrices (quotient matrix manifold)



Riemannian algorithms:

1. Exploit the rank structure in a principled way
2. Develop second-order algorithms systematically
3. Scalable, SVD-free

[Ref] Y. Shi, J. Zhang, and K. B. Letaief, "Low-rank matrix completion for topological interference management by Riemannian pursuit," *IEEE Trans. Wireless Commun.*, vol. 15, no. 7, Jul. 2016.

Concluding remarks

- **Large-scale convex optimization**
  - Convex geometry and analysis provide optimality guarantees
  - Matrix stuffing for fast HSD embedding transformation
  - Operator splitting for solving large-scale HSD embedding
- **Future directions:**
  - Optimality guarantees for more complicated problems, e.g., group sparse beamforming
  - Operator splitting for general large-scale SDP problems, e.g., using approximated cone projection
  - More applications: deep neural network compression via sparse optimization

Concluding remarks

- **Scalable nonconvex optimization algorithms**
  - Nonconvex statistical optimization may not be that scary: no spurious local optima
  - Riemannian optimization is powerful: 1) Exploit the manifold geometry of fixed-rank matrices; 2) Escape saddle points

- **Future directions:**
  - Geometry of neural network loss surfaces: saddle points, local/global optima
  - More applications: blind deconvolution for IoT, big data analytics (e.g., ranking)

To learn more...

- **Web:** http://shiyuanming.github.io/sparserank.html
- **Papers:**
- Y. Shi, J. Zhang, and K. B. Letaief, "Group sparse beamforming for green Cloud-RAN," *IEEE Trans. Wireless Commun.*, vol. 13, no. 5, pp. 2809-2823, May 2014. (The 2016 Marconi Prize Paper Award)
- Y. Shi, J. Zhang, B. O'Donoghue, and K. B. Letaief, "Large-scale convex optimization for dense wireless cooperative networks," *IEEE Trans. Signal Process.*, vol. 63, no. 18, pp. 4729-4743, Sept. 2015. t. 2015. (The 2016 IEEE Signal Processing Society Young Author Best Paper Award)
- Y. Shi, J. Zhang, K. B. Letaief, B. Bai and W. Chen,"Large-scale convex optimization for ultra-dense Cloud-RAN," *IEEE Wireless Commun. Mag.*, pp. 84-91, Jun. 2015.
- Y. Shi, J. Zhang, W. Chen, and K. B. Letaief, "Generalized sparse and low-rank optimization for ultra-dense networks," *IEEE Commun. Mag.*, to appear.

To learn more...

- Y. Shi, J. Zhang, and K. B. Letaief, "Optimal stochastic coordinated beamforming for wireless cooperative networks with CSI uncertainty," *IEEE Trans. Signal Process.*, vol. 63,, no. 4, pp. 960-973, Feb. 2015.
- Y. Shi, J. Zhang, and K. B. Letaief, "Robust group sparse beamforming for multicast green Cloud- RAN with imperfect CSI," *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4647-4659, Sept. 2015.
- Y. Shi, J. Cheng, J. Zhang, B. Bai, W. Chen and K. B. Letaief, "Smoothed L_p-minimization for green Cloud-RAN with user admission control," *IEEE J. Select. Areas Commun.*, vol. 34, no. 4, pp. 1022-1036, Apr. 2016.
- X. Yu, J.-C. Shen, J. Zhang, and K. B. Letaief, "Alternating minimization algorithms for hybrid precoding in millimeter wave MIMO systems," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 3, pp. 485-500, Apr. 2016.
- Y. Shi, J. Zhang, and K. B. Letaief, "Low-rank matrix completion for topological interference management by Riemannian pursuit," *IEEE Trans. Wireless Commun.*, vol. 15, no. 7, pp. 4703-4717, Jul. 2016.
- Y. Shi, B. Mishra, and W. Chen, "Topological interference management with user admission control via Riemannian optimization," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7362-7375, Nov. 2017.
- X. Peng, Y. Shi, J. Zhang, and K. B. Letaief, "Layered group sparse beamforming for cache-enabled wireless networks," *IEEE Trans. Commun.*, to appear.