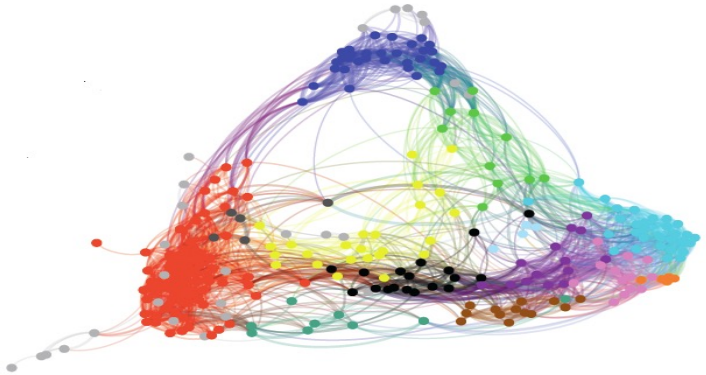
Convex Optimization and Its Applications in Information Science

Fall 2024

Yuanming Shi

上海科技大学
ShanghaiTech University

Vignettes A: Optimization Model, Algorithm, Hardware



Optimization problem

- General optimization problem in standard form:

$$\begin{array}{ll} \text{minimize} & f_0(\boldsymbol{x}) \\ \text{subject to} & f_i(\boldsymbol{x}) \leq 0, i = 1, \ldots, m \end{array}$$

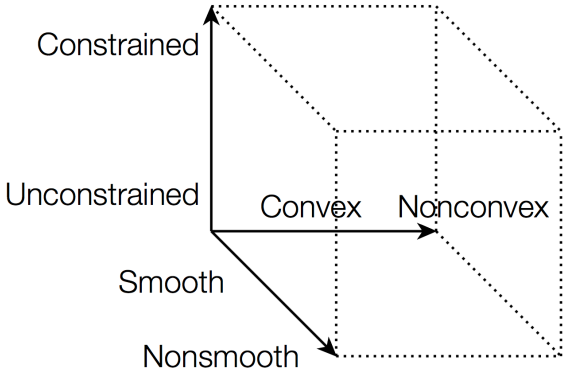
  - $\boldsymbol{x} = (x_1, \ldots, x_n)$: optimization variables
  - $f_0 : \mathbb{R}^n \to \mathbb{R}$: objective function
  - $f_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, m$: constraint functions

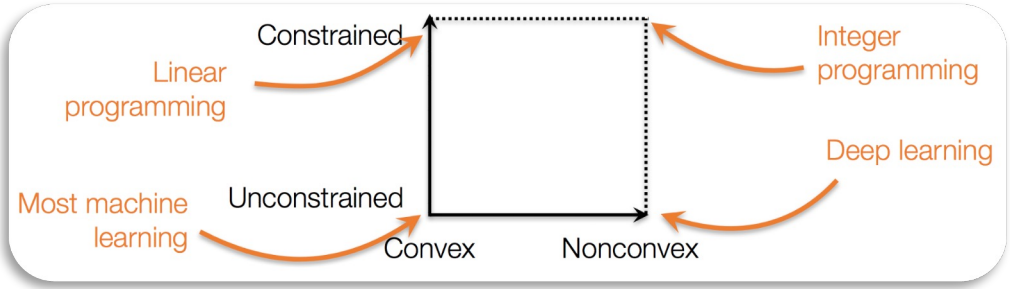
- **Goal:** find optimal solution $x^\star$ minimizing f_0 while satisfying constraints
- **Three basic elements:** 1) variables, 2) constraints, and 3) objective

Optimization **Model**

Classes of optimization problems

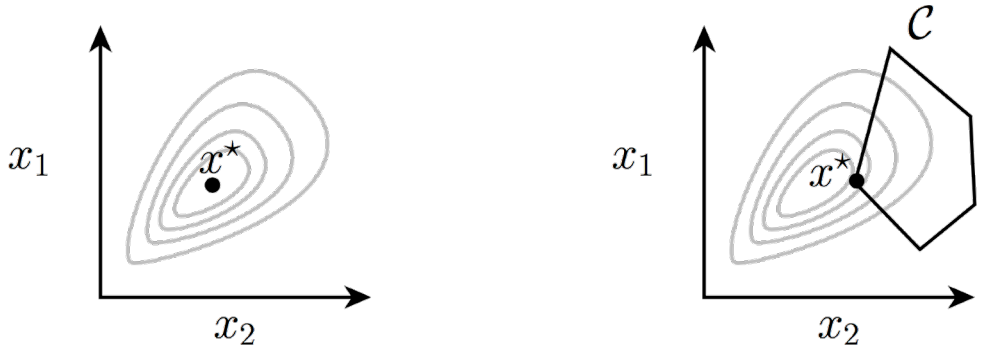
- Types of optimization problems: linear programming, nonlinear programming, integer programming, geometric programming, …





- We focus on three dimensions: ***unconstrained vs. constrained, convex vs. nonconvex, and smooth vs. nonsmooth***

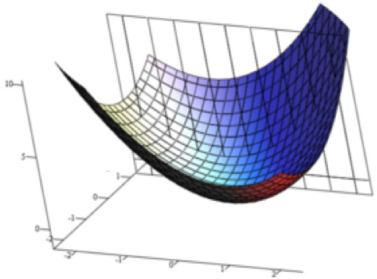
Constrained vs. unconstrained optimization

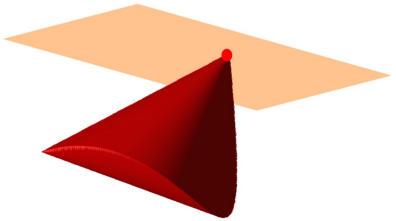


- **Unconstrained optimization:** every point $\boldsymbol{x} \in \mathbb{R}^n$ is feasible, so only focus is on minimizing $f(\boldsymbol{x})$
- **Constrained optimization:** it may be difficult to even *find* a feasible point $\boldsymbol{x} \in \mathcal{C}$

Typically lead to different classes of algorithms

Convex vs. nonconvex optimization





Convex optimization:

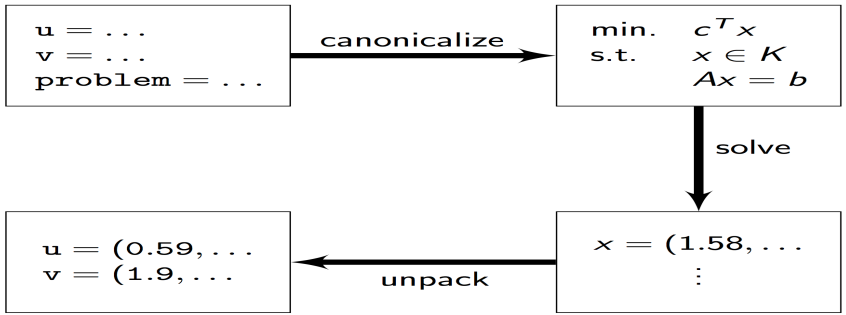
1) All local optima are global optima
2) Can be solved in polynomial-time

"... the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity"

— R. Rockafellar '1993

Modeling languages

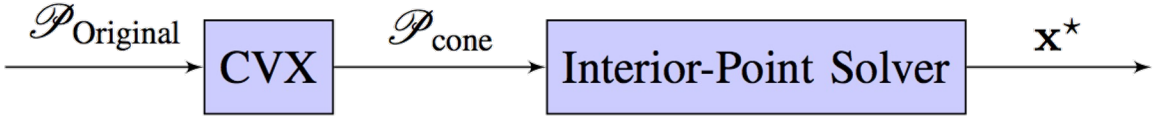
- High level language support for convex optimization
  - **Stage one:** problem description automatically transformed to standard form
  - **Stage two:** solved by standard solver, transformed back to original form



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

Modeling languages

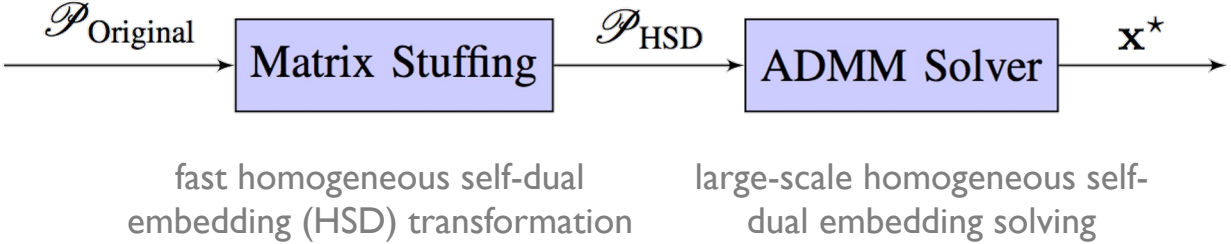
- Disciplined convex programming framework [Grant & Boyd '08]



  - 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



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

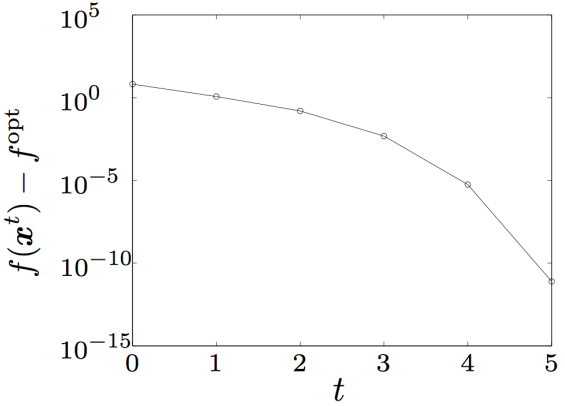
Optimization **Algorithm**

Scaling issues: solvability vs. scalability

- Polynomial-time algorithms might be *useless* in large-scale applications
- **Example:** Newton's method

$$\text{minimize}_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x})$$

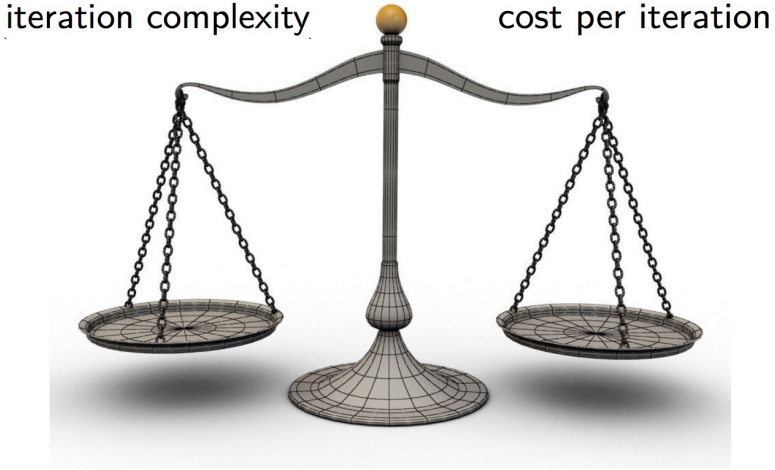
$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - (\nabla^2 f(\boldsymbol{x}^t))^{-1}\nabla f(\boldsymbol{x}^t)$$



- Attains ϵ accuracy within $\mathcal{O}(\log\log\frac{1}{\epsilon})$ iterations; requires $\nabla^2 f(\boldsymbol{x}) \in \mathbb{R}^{n\times n}$
- A single iteration may last forever; prohibitive storage requirement

Iteration complexity vs. per-iteration cost

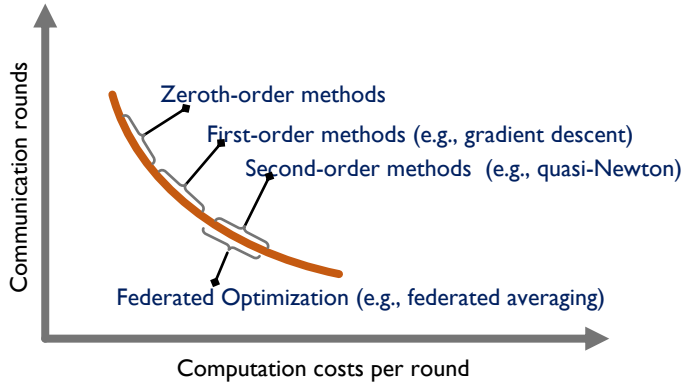
computational cost = iteration complexity (#iterations) x cost per iteration



large-scale problems call for methods with *cheap iterations*

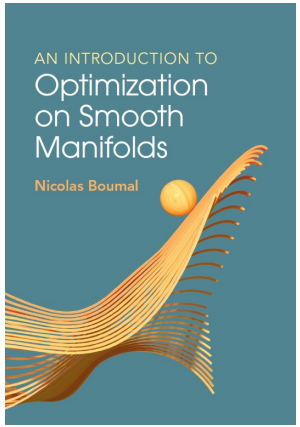
Zero/first/second-order methods

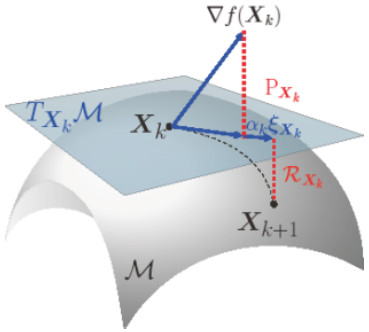
- **First-order methods:** methods that exploit only information on function values and (sub)gradients without using Hessian information
  - cheap iterations
  - low memory requirements



Euclidean space vs. manifolds

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





$$\nabla_{\mathcal{M}} f(\mathbf{X}^{(k)}) = P_{\mathbf{X}^{(k)}}(\nabla f(\mathbf{X}^{(k)}))$$

Riemannian Gradient Euclidean Gradient

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

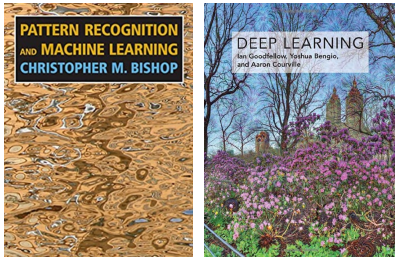
Retraction Operator

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

Machine learning vs. mathematical optimization

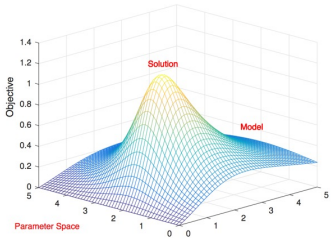
mathematical optimization

Goal: scalable, real-time, parallel, distributed, automatic, etc.

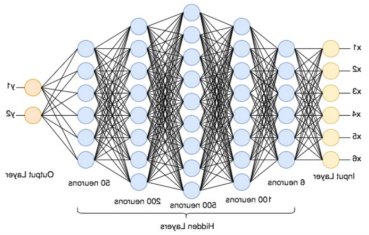


machine learning

- mathematical optimization
  - relies on well-defined model
  - network optimization follows "model-then-optimize"
  - efficient solution often relies on convex landscape

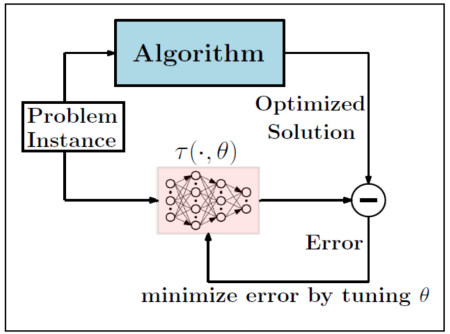


- machine learning
  - data-driven thereby skipping models altogether
  - universal functional mapping via supervised learning
  - highly parallel implementation architecture

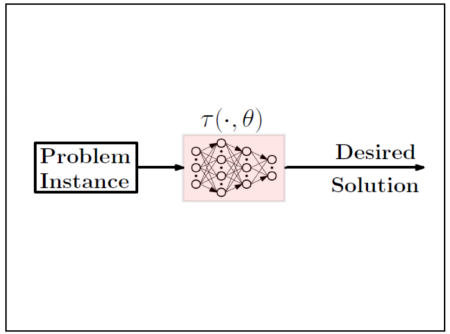


Machine learning for mathematical optimization

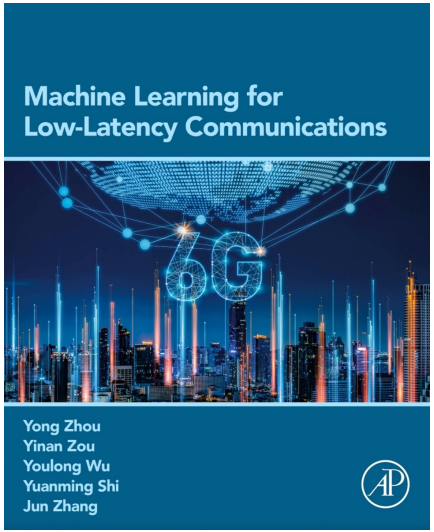
- **Learning to optimize:** obtain near-optimal solution at affordable time by learning an algorithm with neural networks



(a) Training Stage



(b) Testing Stage

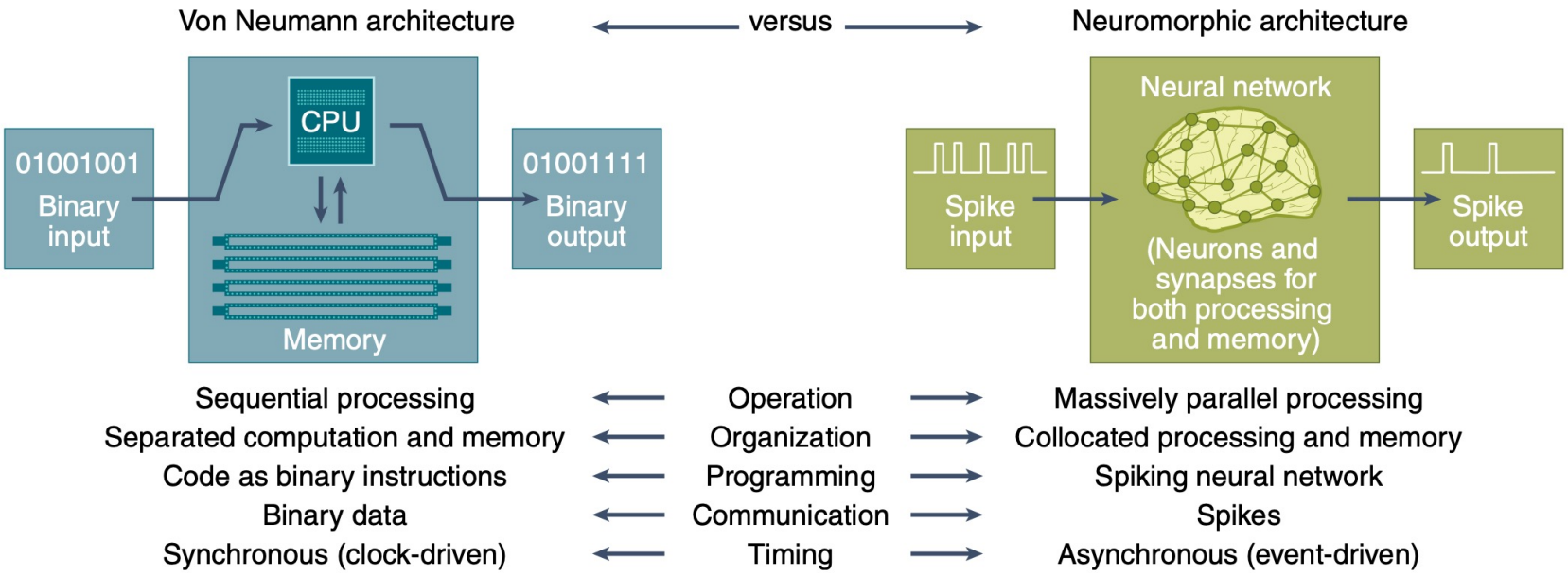


- Advantages
  - hundreds or thousands of speedups
  - better approximation for NP-hard problems
  - system optimization with implicit channel estimation
  - reduce the required pilot length

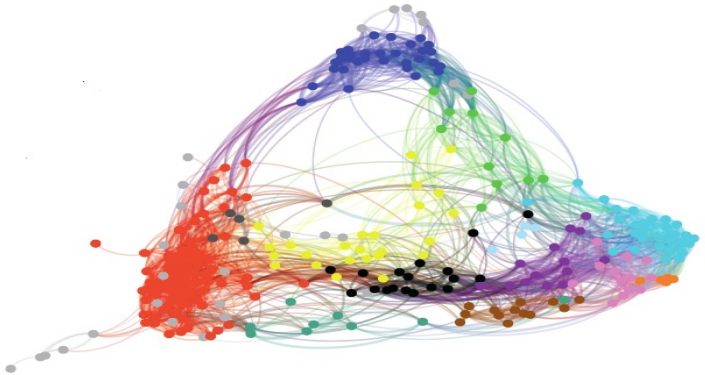
- Challenges
  - design neural networks for different system objectives
  - generalizability to different system parameters
  - interpretability of the obtained solution
  - large amount of training data

Optimization ***Hardware***

Von Neumann vs. Neuromorphic architecture

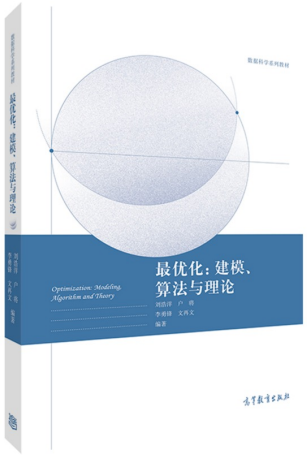


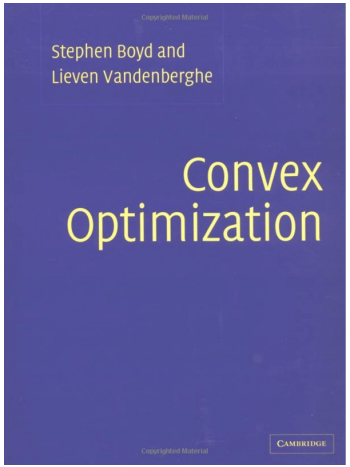
Topics and Grading



Theoretical foundations

- **Main topics:** convex sets, convex functions, convex problems, Lagrange duality and KKT conditions, disciplined convex programming





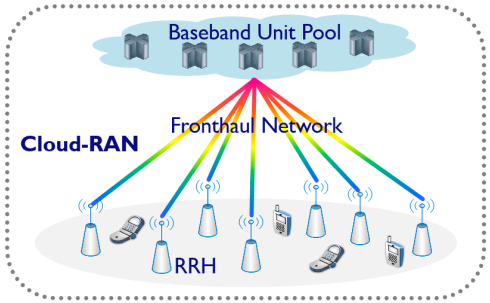
CVX RESEARCH

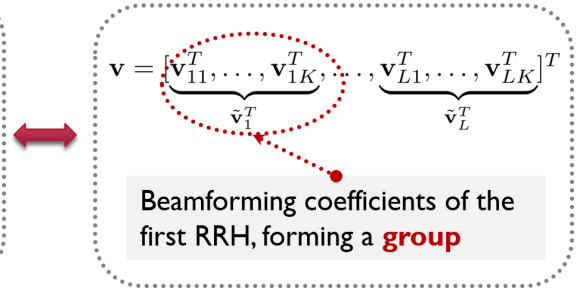
Convex Optimization, by S. Boyd and L. Vandenberghe, Cambridge University Press, 2003.

Applications in wireless networks

- Dense wireless networks: explosion in scale and complexity of the power control and beamforming optimization problems
  - **Questions:** how to exploit the low-dimensional structures (e.g., sparsity and low-rankness) to assist efficient algorithms design?

- **Example:** group sparse beamforming





switch off the l-th RRH

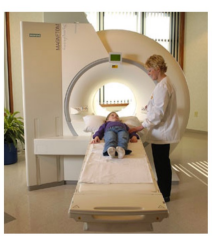
$$\tilde{\mathbf{v}}_l = \mathbf{0}$$

group sparsity structure in $\mathbf{v}$

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. 2014. **(The 2016 Marconi Prize Paper Award)**

Applications in signal processing: compressed sensing

- Let $\boldsymbol{x}^\natural \in \mathbb{R}^d$ be an unknown structured sparse signal
  - ➢ Individual sparsity for compressed sensing
- 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:** $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}^\natural$



MR scanner

MR image

- Find estimate $\hat{\boldsymbol{x}}$ by solving **convex program**

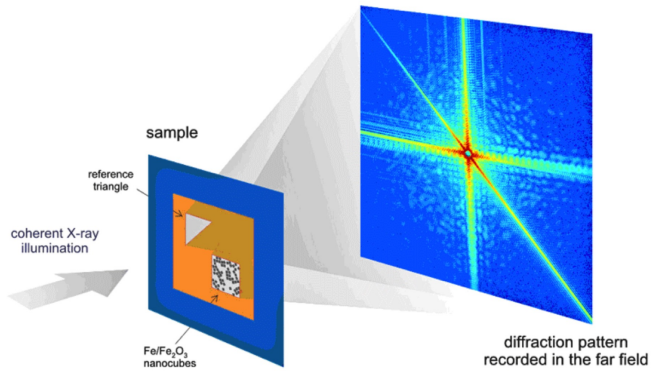
$$\text{minimize} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{z}$$

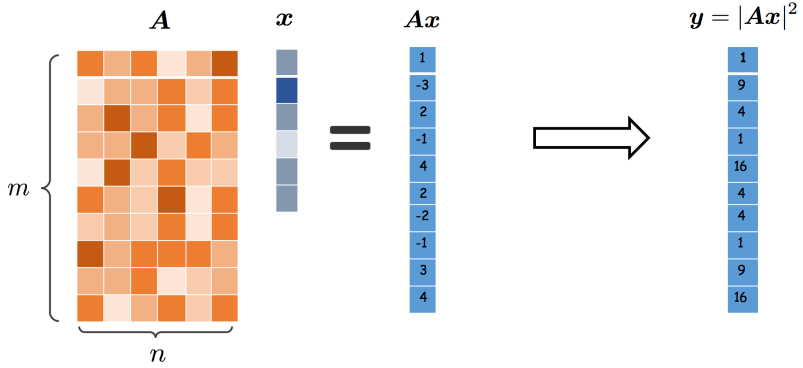
- **Hope:** $\hat{\boldsymbol{x}} = \boldsymbol{x}^\natural$

Y. Shi, J. Dong, and J. Zhang, *Low-overhead Communications in IoT Networks: Structured Signal Processing Approaches*, Springer, Apr. 2020.

Applications in signal processing: phase retrieval

- **Phase retrieval:** recover signal from intensity (missing phase)





- Recover $z^\natural \in \mathbb{R}^n$ from m random quadratic measurements

$$\text{find} \quad z \quad \text{subject to} \quad y_r = |\langle a_r, z \rangle|^2, \quad r = 1, 2, \ldots, m$$

Applications in signal processing: matrix completion

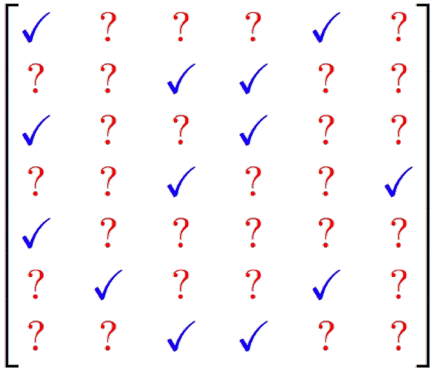


Fig. credit: Candès

- Given partial samples Ω of a low-rank matrix $M^\natural$, fill in missing entries

$$\underset{\boldsymbol{M}\in\mathbb{C}^{m\times n}}{\text{minimize}} \quad \text{rank}(\boldsymbol{M}) \quad \text{subject to} \quad Y_{i,k} = M_{i,k}, \quad (i,k)\in\Omega$$

Applications in machine learning

- **Data:** n observations $\{\boldsymbol{x}_i, y_i\}_{i=1}^n \in \mathcal{X} \times \mathcal{Y}$
- **Prediction function:** $h(\boldsymbol{x}, \boldsymbol{\theta}) \in \mathbb{R}$ parameterized by $\boldsymbol{\theta} \in \mathbb{R}^d$
  - linear predictions: $h(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{\theta}^T \Phi(\boldsymbol{x})$ using features $\Phi(\boldsymbol{x})$
  - neural networks: $h(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{\theta}_m^T \sigma(\boldsymbol{\theta}_{m-1}^T \sigma(\cdots \boldsymbol{\theta}_2^T \sigma(\boldsymbol{\theta}_1^T \boldsymbol{x})))$
- Estimating $\boldsymbol{\theta}$ parameters is an optimization problem (ℓ: loss function)

$$\text{minimize} \quad f(\boldsymbol{\theta}) := \frac{1}{n} \sum_{i=1}^n \ell(h(\boldsymbol{x}_i, \boldsymbol{\theta}), y_i) \quad \text{subject to} \quad \mathcal{R}(\boldsymbol{\theta}) \leq \tau$$

  - $\mathcal{R}$:regularization function encoding prior information (e.g., sparse) on $\boldsymbol{\theta}$

Key benefits of looking at problems in machine learning as optimization problems: separate out the *definition* of the problem from the *method for solving it!*

Applications in robotics

optimal control problem

$$\underset{x(\cdot),u(\cdot),\lambda(\cdot)}{\text{minimize}} \quad \text{Cost}(x(\cdot),u(\cdot),\lambda(\cdot)) \tag{1a}$$

$$\text{subject to} \quad M(q)\,\dot{\nu} + C(q,\nu)\,\nu + \tau_g(q) = S^T\tau + J_c(q)^T\lambda \tag{1b}$$

$$\text{ContactConstraints}(x(t),\lambda(t),u(t),\text{Env}) \tag{1c}$$

$$\text{KinematicsConstraints}(x(t)) \tag{1d}$$

$$\text{InputConstraints}(u(t),x(t)) \tag{1e}$$

$$\text{TaskConstraints}(x(t),u(t),\lambda(t)) \quad \forall t\,. \tag{1f}$$

IEEE TRANSACTIONS ON ROBOTICS, VOL. 40, 2024

Optimization-Based Control for Dynamic Legged Robots

Patrick M. Wensing, *Senior Member, IEEE*, Michael Posa, *Member, IEEE*, Yue Hu, *Member, IEEE*, Adrien Escande, *Member, IEEE*, Nicolas Mansard, and Andrea Del Prete, *Member, IEEE*

Grading

- **Homework:** 4 homework sets
- **Final exam:** 1.5-hours open book exam (end of 12-th week)

- **Course project:**
  - individually
  - list of topics (end of 8-th week); report & slides (end of 12-th week)

$$\text{grade} = 0.2H + 0.5E + 0.3P$$

  - H : homework; E : final exam; P : project

Course information

- **Instructor:** Yuanming Shi (http://faculty.sist.shanghaitech.edu.cn/faculty/shiym/)
  - Email: shiym@shanghaitech.edu.cn
  - Office location: Room 2-302.F, SIST Building
  - Office hours: by appointments

- TA
  - Hanzhe Yang (yanghzh2022@shanghaitech.edu.cn)
  - Office hours: Thursday 15:00-16:00
  - Office location: Room 2-205, SIST Building

- Post all the course materials on Blackboard

Thanks