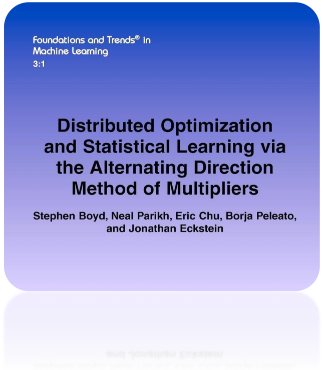
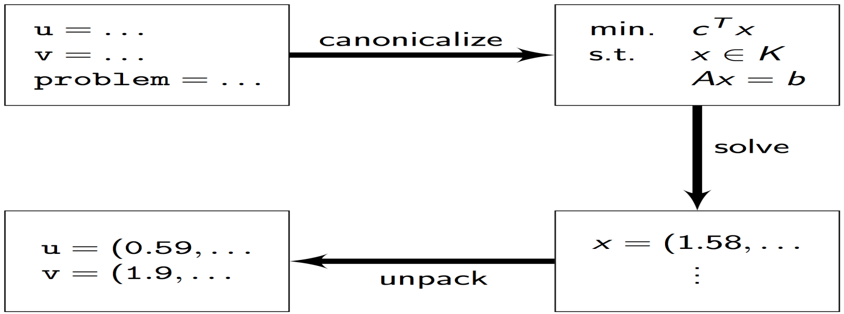
Large-Scale Convex Optimization Algorithms

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



Modeling languages

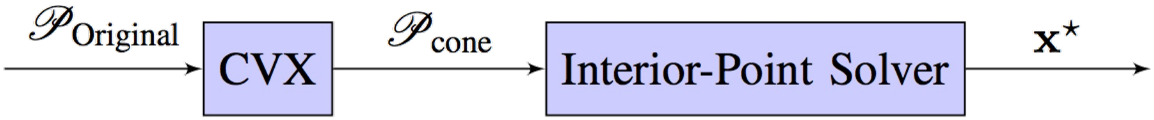
- High level language support for convex optimization
  - **Stage one:** problem description automatically transformed to standard form
  - **Stage two:** solved by standard solver, transformed back to original form



- **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

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

- **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

$$
\begin{aligned}
\mathscr{P}_{\text{Original}} : \text{minimize} \quad & \|\boldsymbol{v}\|_2^2 \\
\text{subject to} \quad & \|\boldsymbol{D}_l \boldsymbol{v}\|_2 \le \sqrt{P_l}, \forall l, && \text{Per-BS power constraint} \quad (1) \\
& \|\boldsymbol{C}_k \boldsymbol{v} + \boldsymbol{g}_k\|_2 \le \beta_k \boldsymbol{r}_k^T \boldsymbol{v}, \forall k. && \text{QoS constraints} \quad (2)
\end{aligned}
$$

- Smith form reformulation

$$
\mathcal{G}_1(l) : \begin{cases} (y_0^l, \mathbf{y}_1^l) \in \mathcal{Q}^{KN_l+1} \\ y_0^l = \sqrt{P_l} \in \mathbb{R} \\ \mathbf{y}_1^l = \boldsymbol{D}_l \boldsymbol{v} \in \mathbb{R}^{KN_l} \end{cases}
\qquad
\mathcal{G}_2(k) : \begin{cases} (t_0^k, \mathbf{t}_1^k) \in \mathcal{Q}^{K+1} \\ t_0^k = \beta_k \mathbf{r}_k^T \mathbf{v} \in \mathbb{R} \\ \mathbf{t}_1^k = \mathbf{t}_2^k + \mathbf{t}_3^k \in \mathbb{R}^{K+1} \\ \mathbf{t}_2^k = \mathbf{C}_k \mathbf{v} \in \mathbb{R}^{K+1} \\ \mathbf{t}_3^k = \mathbf{g}_k \in \mathbb{R}^{K+1} \end{cases}
$$

Smith form for (1)

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)

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

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

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

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

- **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

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

Alternating direction method of multipliers

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

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

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

- The basic ADMM algorithm is

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

  - $\rho > 0$ 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**

$$\tilde{\mathbf{x}}^{[i+1]} = (\mathbf{I}+\mathbf{Q})^{-1}(\mathbf{x}^{[i]} + \mathbf{y}^{[i]})$$ subspace projection

$$\mathbf{x}^{[i+1]} = \Pi_{\mathcal{C}}(\tilde{\mathbf{x}}^{[i+1]} - \mathbf{y}^{[i]})$$ parallel cone projection

$$\mathbf{y}^{[i+1]} = \mathbf{y}^{[i]} - \tilde{\mathbf{x}}^{[i+1]} + \mathbf{x}^{[i+1]}$$ computationally trivial

Parallel cone projection

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

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

    - 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)**