A Brief Introduction to Prox-affine Forms in Convex Optimization

Junzi Zhang, Anqi Fu, Stephen Boyd

EE & ICME Departments
Stanford University

Prox-affine Form of Generic Convex Optimization

We consider the following **prox-affine** [WWK2015, FZB2019] formulation of a **generic** convex optimization problem:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^N f_i(x_i) \\ \text{subject to} & \sum_{i=1}^N A_i x_i = b. \end{array}$$

with variable $x = (x_1, \ldots, x_N) \in \mathbf{R}^{n_1+\cdots+n_N}$, $A_i \in \mathbf{R}^{m\times n_i}$, $b \in \mathbf{R}^m$.

- $f_i : \mathbf{R}^{n_i} \to \mathbf{R} \cup \{+\infty\}$ is closed, convex and proper (CCP).
- Each f_i can **only** be accessed through its proximal operator:

$$\mathbf{prox}_{tf_i}(v_i) = \mathrm{argmin}_{x_i} \left(f_i(x_i) + \frac{1}{2t}\|x_i - v_i\|_2^2 \right).$$

Prox-affine Form of Generic Convex Optimization

Remark on $\mathbf{prox}_{tf}$

- Generalization of projection: take $f(x) = \mathcal{I}_{\mathcal{C}}(x)$, $\mathbf{prox}_{tf} = \Pi_{\mathcal{C}}$.
- $\mathbf{prox}_{tf}$ is a smoothing of f.
- $\mathbf{prox}_{tf}(x) = x$ iff $x \in \operatorname{argmin}_x f(x)$.
- For many f, $\mathbf{prox}_{tf}$ is closed-form.
- $\mathbf{prox}_{t\sum_{i=1}^{N} f_i(x_i)}(v_1, \ldots, v_N) = (\mathbf{prox}_{tf_1}(v_1), \ldots, \mathbf{prox}_{tf_N}(v_N))$.

Prox-affine Form of Generic Convex Optimization

Why **prox-affine** form?

- **Separability:** suitable for parallel and distributed implementation.
- **Black-box proximal:** suitable for peer-to-peer optimization with privacy requirements.
- **Compact representation:** alternative to **conic** standard form.
  - Cone programs can be represented in prox-affine form by consensus without complication (but NOT vice versa).
  - With log, exp, det involved, prox-affine form is much more compact.

Conic Form as Prox-affine Form

Conic form: ($\mathcal{K}$ is a nonempty, closed and convex cone)

$$\begin{array}{ll} \text{minimize} & c^T x + \frac{1}{2} x^T Q x \\ \text{subject to} & Ax = b, \quad x \in \mathcal{K}. \end{array}$$

- Used in most solvers: `CPLEX`, `MOSEK`, `GUROBI`, `SCS`, `ECOS`, `OSQP`.
- Target standard form of most modeling languages: `CVX*`, `YALMIP`.

Prox-affine form of cone programs via consensus:

$$\begin{array}{ll} \text{minimize} & c^T x_1 + \frac{1}{2} x_2^T Q x_2 + \mathcal{I}_{\mathcal{K}}(x_3) \\ \text{subject to} & Ax_1 = b, \quad x_1 = x_2 = x_3. \end{array}$$

- Consensus already used in most conic solvers, so no complication.
- Can be further parallelized when $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_m$.

Example: Portfolio Optimization

Portfolio optimization with transaction costs and risk constraints:

$$\begin{array}{ll} \text{maximize} & r^T x - \sum_{j=1}^{n} (a_j |x^j| + b_j |x^j|^{3/2}) \\ \text{subject to} & (w+x)^T \Sigma (w+x) \leq \rho, \quad \mathbf{1}^T x = 0. \end{array}$$

Here $x = (x^1, \ldots, x^n)$ (and similarly for other variables hereafter).

Example: Portfolio Optimization

Conic form: (*e.g.*, what `MOSEK`, `GUROBI`, `SCS` & `ECOS` accept)

$$\begin{array}{ll} \text{minimize} & -r^T x + \sum_{j=1}^n (a_j t_1^j + b_j t_2^j) \\ \text{subject to} & \Sigma^{1/2}(w + x) = g,\ \|g\|_2 \le \alpha,\ \alpha = \sqrt{\rho},\ \mathbf{1}^T x = 0, \\ & x^j + t_1^j \ge 0,\ x^j - t_1^j \le 0,\ x - z \le 0,\ x + z \ge 0, \\ & (z^j)^2 \le 2 s^j t_2^j,\ (w^j)^2 \le 2 v^j u^j,\ z = v,\ s = w,\ u = \frac{1}{8}\mathbf{1}, \\ & s \ge 0,\ t_2 \ge 0,\ v \ge 0,\ u \ge 0,\ j = 1, \ldots, n. \end{array}$$

Variables: $x, t_1, t_2, g, \alpha, z, v, s, w, u$ – dimension $= 9n + 1$.

- complicated transformation and redundancy.
- **more consensus variable copies** are implicitly created to *separate the projections* in the solvers (*e.g.*, `SCS`) – dimension $> 9n + 1$.

Example: Portfolio Optimization

Prox-affine form: (`Epsilon` & `a2dr`)

$$\begin{array}{ll} \text{minimize} & -r^T x_1 + \sum_{j=1}^n a_j |x_2^j| + \sum_{j=1}^n b_j |x_3^j|^{3/2} + \mathcal{I}_{\|x_5\|_2 \le \sqrt{\rho}}(x_5), \\ \text{subject to} & \Sigma^{1/2}(w + x_4) = x_5, \quad \mathbf{1}^T x_1 = 0, \quad x_1 = x_2 = x_3 = x_4 = x_5. \end{array}$$

Variables: x_1, x_2, x_3, x_4, x_5 – dimension $= 5n$.

- Straightforward & compact: more dramatic with log, det, exp.
- Separation into low dimensional problems (easy parallelization): $\mathbf{prox}_{t\sum_{i=1}^N f_i(x_i)}(v_1, \ldots, v_N) = (\mathbf{prox}_{tf_1}(v_1), \ldots, \mathbf{prox}_{tf_N}(v_N))$.
- Closed-form $\mathbf{prox}_{tr^T x_1}$, $\mathbf{prox}_{ta_j|x_2|^j}$, $\mathbf{prox}_{tb_j|x_3^j|^{3/2}}$ and $\Pi_{\{\|x_5\|_2 \le \sqrt{\rho}\}}$.
- No additional consensus variable copies are needed: can be directly solved by `Epsilon` & `a2dr`.

Related Solvers and Links

`Epsilon` (2015)

- expression tree compiler for transforming convex optimization problems into prox-affine forms
- `https://github.com/mwytock/epsilon`.

`POGS` (2015)

- first-order GPU-compatible solver for *graph form* convex optimization problems; graph form is similar to prox-affine form
- `http://foges.github.io/pogs/`.

`a2dr` (2019)

- (Anderson) accelerated Python solver for prox-affine distributed convex optimization
- `https://github.com/cvxgrp/a2dr`.

References

Wytock, M., Wang, P. W. and Kolter, J. Z. (2015).
Convex Programming with Fast Proximal and Linear Operators.

Fougner, C. and Boyd, S. P. (2015).
Parameter Selection and Preconditioning for a Graph Form Solver.

Fu, A.*, Zhang, J.* and Boyd, S. P. (2019). (*equal contribution)
Anderson Accelerated Douglas-Rachford Splitting.