Disciplined Biconvex Programming

Hao Zhu

Department of Computer Science

University of Freiburg

CVXPY Workshop, Stanford University

February 20, 2026

Overview

disciplined biconvex programming (DBCP):

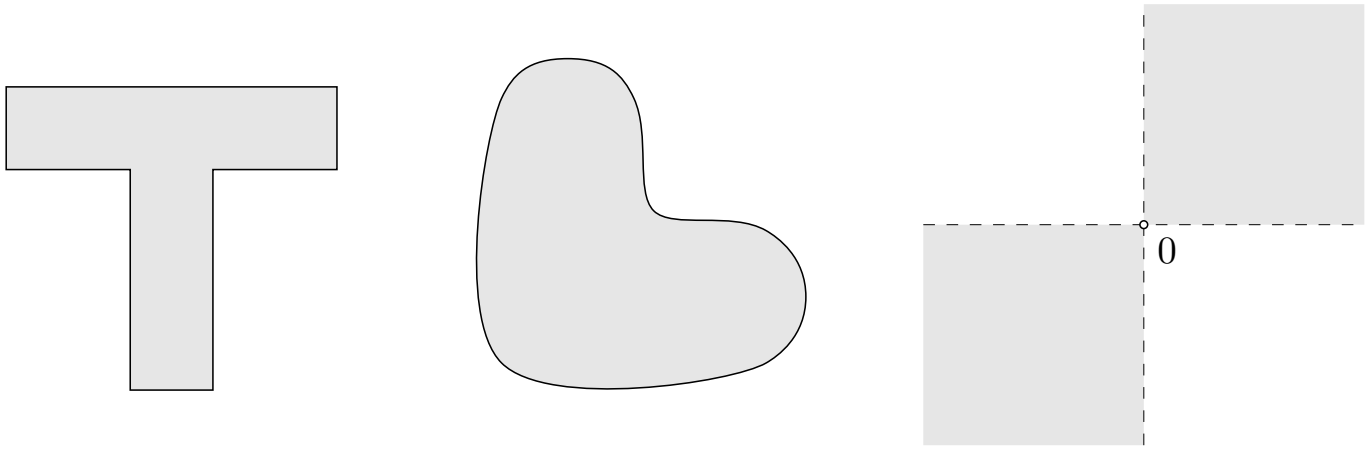
- domain specific language for biconvex programming
- Python package `dbcp` as an extension of CVXPY
- (heuristic) solution methods based on alternate minimization [GPK07]
- fast prototyping of many machine learning problems
  - clustering with constraints
  - generalized principal component analysis
  - fitting latent factor models
  - . . .

Biconvex sets

$B \subseteq \mathcal{X} \times \mathcal{Y}$ is a **biconvex set**, if

- fix $\tilde{y} \in \mathcal{Y} \implies B_{\tilde{y}} = \{x \in \mathcal{X} \mid (x, \tilde{y}) \in B\}$ convex
- fix $\tilde{x} \in \mathcal{X} \implies B_{\tilde{x}} = \{y \in \mathcal{Y} \mid (\tilde{x}, y) \in B\}$ convex

examples:



- biconvex sets can be unconnected: $\{(x, y) \in \mathbf{R}^2 \mid x, y < 0 \text{ or } x, y > 0\}$
- intersection of (any number of) biconvex sets is biconvex

Biconvex functions

$f\colon \mathcal{X} \times \mathcal{Y} \to \mathbf{R}$ is a **biconvex function** if

- fix $\tilde{y} \in \mathcal{Y} \implies f_{\tilde{y}}\colon \mathcal{X} \to \mathbf{R}$, $x \mapsto f(x, \tilde{y})$ convex
- fix $\tilde{x} \in \mathcal{X} \implies f_{\tilde{x}}\colon \mathcal{Y} \to \mathbf{R}$, $y \mapsto f(\tilde{x}, y)$ convex

define biconcave, biaffine, and bilinear functions similarly

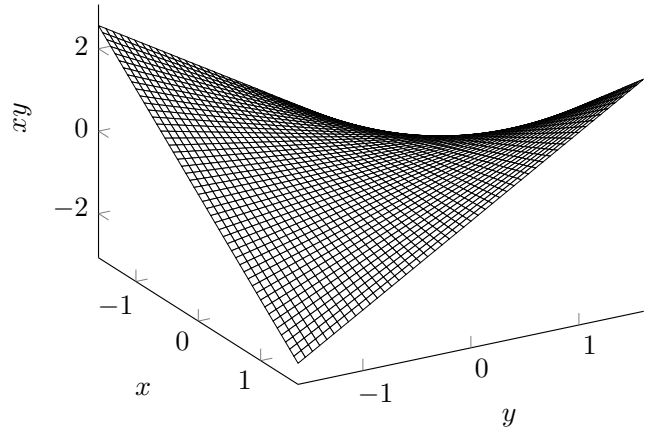
some properties:

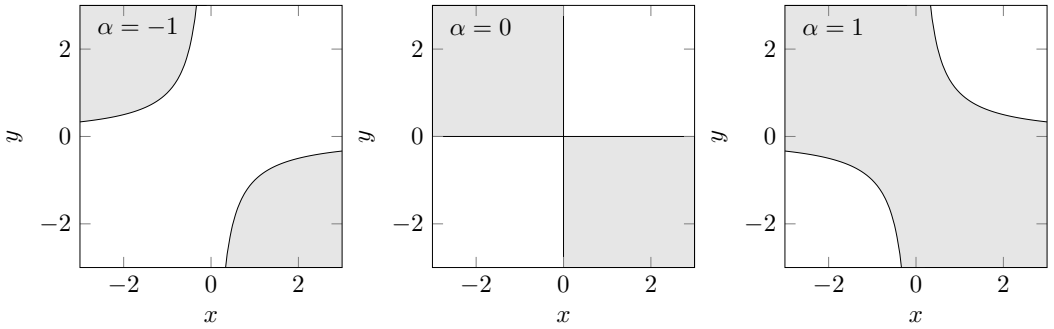
- sublevel sets of a biconvex function are biconvex
- operations that preserve biconvexity:
  - nonnegative weighted sum
  - pointwise maximum and supremum
  - biaffine precomposition:

    $h\colon \mathbf{R} \to \mathbf{R}$ convex $\implies h((Ax+b)^T(Cy+d))$ biconvex

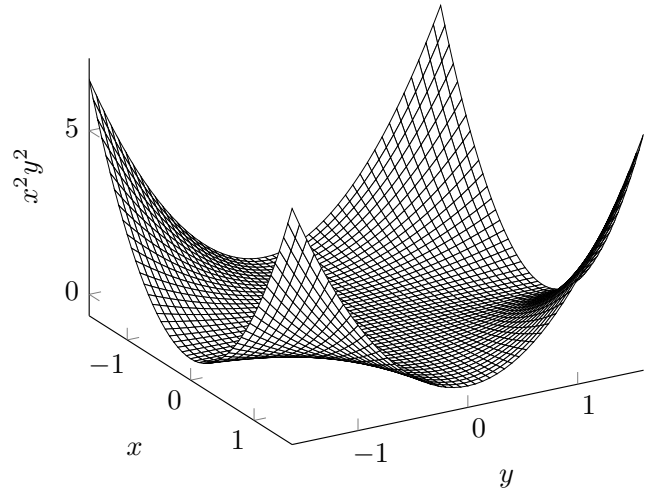
examples:

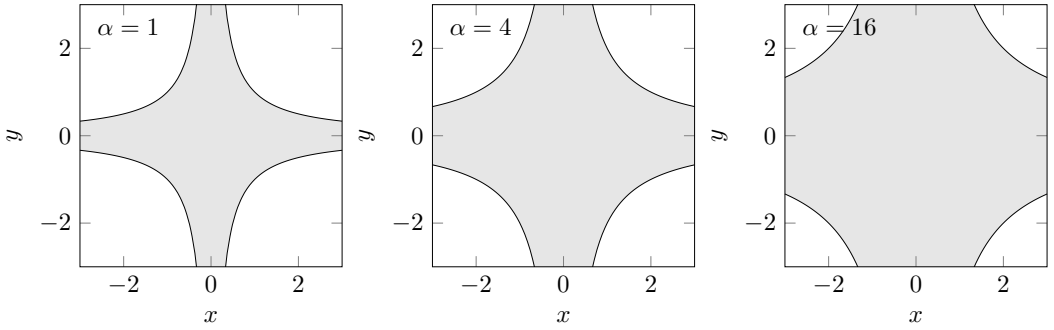
- $f(x, y) = xy$:





- $f(x, y) = x^2y^2$:





Biconvex optimization problems

$$\begin{array}{ll} \text{minimize} & f_0(x,y) \\ \text{subject to} & f_i(x,y) \leq 0, \quad i = 1,\ldots,m \\ & (Ax+b)^T(Cy+d) = 0 \end{array}$$

variables $x \in \mathcal{X}$, $y \in \mathcal{Y}$; $f_0, f_1, \ldots, f_m$ biconvex

- difficult to solve in general (mostly NP-hard)
- let $\mathcal{D}$ be the feasible set, $(x^\star, y^\star) \in \mathcal{D}$ is **partially optimal** if

$$f_0(x^\star, y^\star) \leq f_0(x, y^\star) \quad \text{and} \quad f_0(x^\star, y^\star) \leq f_0(x^\star, y)$$

  for all $x \in \mathcal{D}_{y^\star}$, $y \in \mathcal{D}_{x^\star}$

  - every stationary point is partially optimal, and vice versa
  - not necessarily globally or even locally optimal
  - turn out to work quite well in practice

Alternate convex search

basic idea: alternate between convex subproblems in x and y

Algorithm 1 ALTERNATE CONVEX SEARCH.

given a starting point $(x^{(0)}, y^{(0)}) \in \mathcal{D}$

$k := 0$.

repeat

1. $x^{(k+1)} := \underset{x \in \mathcal{X}}{\operatorname{argmin}} \left\{ f_0(x, y^{(k)}) \;\middle|\; \begin{array}{l} f_i(x, y^{(k)}) \leq 0, \;\; i = 1, \ldots, m \\ h_i(x, y^{(k)}) = 0, \;\; i = 1, \ldots, p \end{array} \right\}$

2. $y^{(k+1)} := \underset{y \in \mathcal{Y}}{\operatorname{argmin}} \left\{ f_0(x^{(k+1)}, y) \;\middle|\; \begin{array}{l} f_i(x^{(k+1)}, y) \leq 0, \;\; i = 1, \ldots, m \\ h_i(x^{(k+1)}, y) = 0, \;\; i = 1, \ldots, p \end{array} \right\}$

3. $k := k + 1$.

until stopping criteria is satisfied.

- theoretical convergence guarantee to stationary points
- **variations**: proximal regularization, penalty method infeasible start, . . .

DBCP product ruleset

most DBCP rules are inherited from DCP, with additional product rules:

1. products involving variables on both sides should be one of:

   affine * *affine*

   affine-nonneg * *convex*/*affine-nonpos* * *concave*

   convex-nonneg * *convex-nonneg*/*concave-nonpos* * *concave-nonpos*

2. no loop allowed in the variable products, $e.g.$,

   ```
   x * y + y * z + z * x
   ```

   is not a valid DBCP expression

Some simple examples

principal component analysis:

$$\text{minimize} \quad \|XY - A\|_F^2$$

biconvex in factor matrices $X \in \mathbf{R}^{m\times k}$ and $Y \in \mathbf{R}^{k\times n}$

k-**means clustering**:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^m z_i^T(\|\bar{x}_1 - x_i\|_2^2, \ldots, \|\bar{x}_k - x_i\|_2^2) \\ \text{subject to} & 0 \preceq z_i \preceq \mathbf{1}, \quad \mathbf{1}^T z_i = 1, \quad i = 1, \ldots, m \end{array}$$

biconvex in cluster centers $\bar{x}_i \in \mathbf{R}^n$ and (soft) cluster labels $z_i \in \mathbf{R}^k$

```
xbars, zs = cp.Variable((k, n)), cp.Variable((m, k), nonneg=True)
obj = cp.sum(cp.multiply(zs, cp.vstack([cp.sum(cp.square(xs - c), axis
  =1) for c in xbars]).T))
constr = [zs <= 1, cp.sum(zs, axis=1) == 1]
prob = BiconvexProblem(cp.Minimize(obj), [[xbars], [zs]], constr)
```

allows fast model prototyping, close to the idea of **customized modeling**

Example: Input-output hidden Markov model

given dataset $\{(x(t), y(t))\}_{t=1}^m$ generated from a K-state HMM:

- $x(t) \in \mathbf{R}^n$ are the features, $y(t) \in \{0, 1\}$ are the labels
- $\tilde{z}(t) \in \{1, \ldots, K\}$: hidden state labels, modeled as a Markov chain

$$\tilde{z}(t) \sim \begin{cases} \mathrm{Cat}(p_{\mathrm{init}}) & t = 0 \\ \mathrm{Cat}(p_{\tilde{z}(t-1)}) & t > 0 \end{cases}$$

  - $p_{\mathrm{init}} \in \mathbf{R}^K$ $(\mathbf{1}^T p_{\mathrm{init}} = 1)$: initial state distribution
  - $P_{\mathrm{tr}} \in \mathbf{R}^{K \times K}$ $(P_{\mathrm{tr}}\mathbf{1} = \mathbf{1})$: state transition matrix
  - $p_{\tilde{z}(t-1)} \in \mathbf{R}^K$ is the $\tilde{z}(t-1)$th row of P_{tr}
- response $y(t)$ generated from a logistic model:

$$\mathbf{prob}(y(t) = 1) = 1/(1 + \exp(-x(t)^T \theta_{\tilde{z}(t)}))$$

  - $\theta_{\tilde{z}(t)} \in \{\theta_1, \ldots, \theta_K\} \subseteq \mathbf{R}^n$ is the coefficient

fitting problem: estimate model parameters P_{tr}, $\theta_1, \ldots, \theta_K$

- variables:
  - hidden state probability: $z(t) \in \mathbf{R}_+^K$ s.t. $\mathbf{1}^T z(t) = 1$, $t = 1, \ldots, m$
  - logistic model coefficients: $\theta_1, \ldots, \theta_K \in \mathbf{R}^n$
- objective:

$$L = -\sum_{t=1}^{m} \sum_{k=1}^{K} z_k(t) \left(y(t) x(t)^T \theta_k - \log(1 + \exp(x(t)^T \theta_k)) \right)$$

  - negative log-likelihood of the observed data
  - biconvex in $z(t)$ and θ_k
- some (widely considered) prior information:
  - coefficients θ_k should be small
  - hidden state transition should be sparse, $i.e.$, P_{tr} close to identity
  - constraints on coefficients θ_k ($e.g.$, nonnegativity, monotonicity, . . .)

biconvex program formulation:

$$\begin{array}{ll} \text{minimize} & L + \alpha_\theta \sum_{k=1}^{K} \|\theta_k\|_2^2 + \alpha_z \sum_{t=1}^{m-1} D_{\mathrm{kl}}(z(t), z(t+1)) \\ \text{subject to} & 0 \preceq z(t) \preceq \mathbf{1}, \quad \mathbf{1}^T z(t) = 1, \quad t = 1, \ldots, m \\ & \theta_k \in \mathcal{C}_k, \quad k = 1, \ldots, K \end{array}$$

- variables $z(t) \in \mathbf{R}^K$ and $\theta_k \in \mathbf{R}^n$
- regularization weights $\alpha_\theta, \alpha_z \geq 0$
- L: biconvex negative log-likelihood objective
- D_{kl} is the Kullback-Leibler divergence
- $\mathcal{C}_k$ are convex sets specifying (hard) structural constraints on θ_k

numerical example:

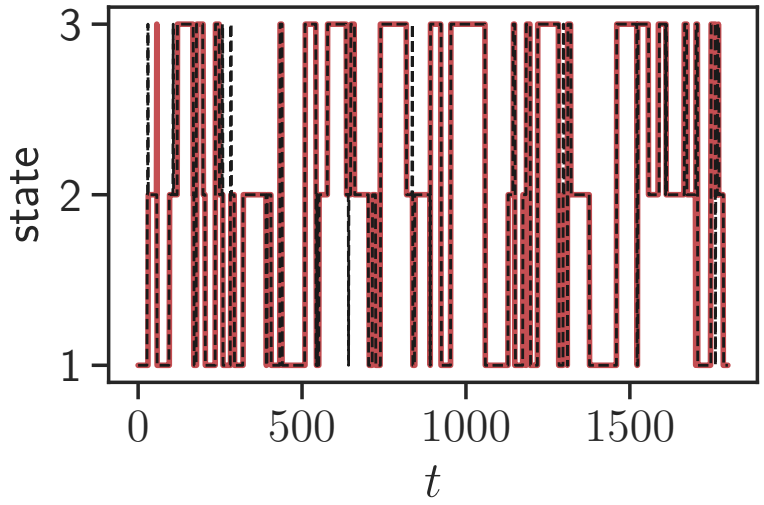
- $m = 1800$ samples, feature dimension $n = 2$ (with intercept)
- $K = 3$ hidden states with $P_{\text{tr}} = \begin{bmatrix} 0.95 & 0.025 & 0.025 \\ 0.025 & 0.95 & 0.025 \\ 0.025 & 0.025 & 0.95 \end{bmatrix}$
- constraints: $\theta_{1,1} \leq 0, \quad \theta_{2,1} \geq 0, \quad \theta_{3,1} \geq 0, \quad \theta_{2,2} \geq \theta_{3,2}$
- to specify the problem in `dbcp`:

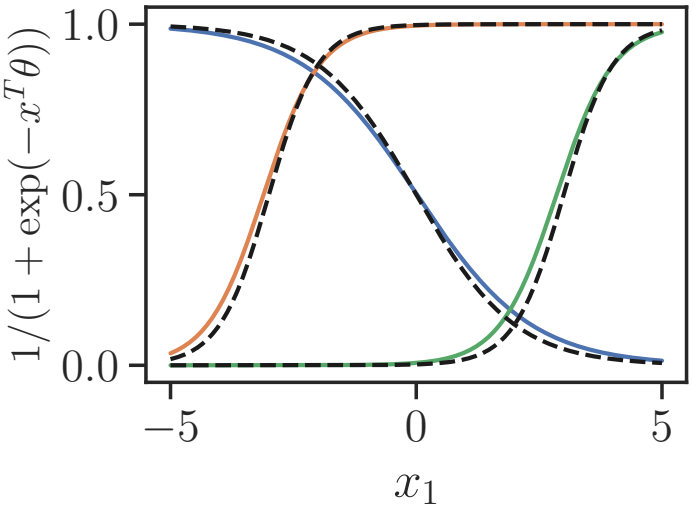
```
thetas = cp.Variable((K, n))
zs = cp.Variable((m, K), nonneg=True)

rs = [-cp.multiply(ys, xs @ thetas[k]) + cp.logistic(xs @ thetas[k])
    for k in range(K)]
obj = cp.Minimize(cp.sum(cp.multiply(zs, cp.vstack(rs).T))
    + alpha_theta * cp.sum_squares(thetas)
    + alpha_z * cp.sum(cp.kl_div(zs[:-1], zs[1:])))
constr = [zs <= 1, cp.sum(zs, axis=1) == 1,
    thetas[0][0] <= 0, thetas[1][0] >= 0,
    thetas[2][0] >= 0, thetas[1][1] >= thetas[2][1]]

prob = BiconvexRelaxProblem(obj, ([zs], [thetas]), constr)
```

- results with $\alpha_\theta = 0.1, \alpha_z = 2$ (ground truth shown dashed):





$$\hat{P}_{\text{tr}} = \begin{bmatrix} 0.96 & 0.02 & 0.02 \\ 0.03 & 0.95 & 0.02 \\ 0.01 & 0.02 & 0.97 \end{bmatrix}$$

Resources

- paper:

  Disciplined biconvex programming.
  H. Zhu and J. Boedecker. *arXiv:2511.01813*, November 2025.

- Python package `dbcp`:

  `https://github.com/nrgrp/dbcp`

  – installation: `pip install dbcp`

- examples and tutorial:

  `https://github.com/nrgrp/dbcp/tree/main/examples`

Any other applications of DBCP? Let us know!

References

[GPK07] J. Gorski, F. Pfeuffer, and K. Klamroth. Biconvex sets and optimization with biconvex functions: A survey and extensions. *Mathematical Methods of Operations Research*, 66(3):373–407, 2007.

[ZB25] H. Zhu and J. Boedecker. Disciplined biconvex programming. *arXiv Preprint arXiv:2511.01813*, 2025.