Mathematical Optimization Models and Applications

Yinyu Ye

Department of Management Science and Engineering

Stanford University

Stanford, CA 94305, U.S.A.

`http://www.stanford.edu/˜yyye`

Chapters 1, 2.1-2, 6.1-2, 7.2, 11.1, 11.4

Model Classifications

Optimization problems are generally divided into Unconstrained, Linear and Nonlinear Programming based upon the objective and constraints of the problem

- Unconstrained Optimization: Ω is the entire space R^n
- Linear Optimization: If both the objective and the constraint functions are linear/affine
- Conic Linear Optimization: If both the objective and the constraint functions are linear/affine, but variables in a convex cone.
- Nonlinear Optimization: If the constraints contain general nonlinear functions
- Stochastic Optimization: Optimize the expected objective function with random parameters
- Fixed-Point Computation: Optimization of multiple agents with zero-sum objectives
- There are integer program, mixed-integer program etc.

We present a few optimization examples in this lecture that we would cover through out this course.

Logistic Regression I

Given a training data point $\mathbf{a}_i \in R^n$, according to the logistic model, the probability that it's in one class C is represented by

$$\frac{e^{\mathbf{a}_i^T\mathbf{x}+x_0}}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}.$$

Thus, for some training data points, we like to determine intercept x_0 and slope vector $\mathbf{x} \in R^n$ such that

$$\frac{e^{\mathbf{a}_i^T\mathbf{x}+x_0}}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}} = \begin{cases} 1, & \text{if } \mathbf{a}_i \in C \\ 0, & \text{otherwise} \end{cases}.$$

Then the probability to give a “right classification answer” for all training data points is

$$\left(\prod_{\mathbf{a}_i \in C} \frac{e^{\mathbf{a}_i^T\mathbf{x}+x_0}}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}\right)\left(\prod_{\mathbf{a}_i \notin C} \frac{1}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}\right)$$

Logistic Regression II

Therefore, we like to maximize the probability when deciding intercept x_0 and slope vector $\mathbf{x} \in R^n$

$$\left(\prod_{\mathbf{a}_i \in C} \frac{e^{\mathbf{a}_i^T\mathbf{x}+x_0}}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}\right)\left(\prod_{\mathbf{a}_i \notin C} \frac{1}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}\right) = \left(\prod_{\mathbf{a}_i \in C} \frac{1}{1+e^{-\mathbf{a}_i^T\mathbf{x}-x_0}}\right)\left(\prod_{\mathbf{a}_i \notin C} \frac{1}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}\right),$$

which is equivalently to maximize

$$-\left(\sum_{\mathbf{a}_i \in C} \ln(1+e^{-\mathbf{a}_i^T\mathbf{x}-x_0})\right) - \left(\sum_{\mathbf{a}_i \notin C} \ln(1+e^{\mathbf{a}_i^T\mathbf{x}+x_0})\right).$$

Or

$$\min_{x_0,\mathbf{x}} \quad \left(\sum_{\mathbf{a}_i \in C} \ln(1+e^{-\mathbf{a}_i^T\mathbf{x}-x_0})\right) + \left(\sum_{\mathbf{a}_i \notin C} \ln(1+e^{\mathbf{a}_i^T\mathbf{x}+x_0})\right).$$

This is an unconstrained optimization problem, where the objective is a convex function of decision variables: intercept x_0 and slope vector $\mathbf{x} \in R^n$.

Linear Programming and Conic Linear Programming

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \in K. \end{array}$$

Linear Programming (LP): when K is the nonnegative orthant cone

Second-Order Cone Programming (SOCP): when K is the second-order cone

Semidefinite Cone Programming (SDP): when K is the semidefinite matrix cone

$$\begin{array}{ll} \text{min} & 2x_1 + x_2 + x_3 \\ \text{s.t.} & x_1 + x_2 + x_3 = 1, \\ & (x_1; x_2; x_3) \geq \mathbf{0}; \end{array}$$

$$\begin{array}{ll} \text{min} & 2x_1 + x_2 + x_3 \\ \text{s.t.} & x_1 + x_2 + x_3 = 1, \\ & \sqrt{x_2^2 + x_3^2} \leq x_1. \end{array}$$

$$\begin{array}{ll} \text{min} & 2x_1 + x_2 + x_3 \\ \text{s.t.} & x_1 + x_2 + x_3 = 1, \\ & \begin{pmatrix} x_1 & x_2 \\ x_2 & x_3 \end{pmatrix} \succeq \mathbf{0}, \end{array}$$

LP Examles: Sparsest Data Fitting

We want to find a sparsest solution to fit exact data measurements, that is, to minimize the number of non-zero entries in $\mathbf{x}$ such that $A\mathbf{x} = \mathbf{b}$:

$$\begin{array}{ll} \text{minimize} & \|\mathbf{x}\|_0 = |\{j : x_j \neq 0\}| \\ \text{subject to} & A\mathbf{x} = \mathbf{b}. \end{array}$$

Sometimes this objective can be accomplished by LASSO:

$$\begin{array}{ll} \text{minimize} & \|\mathbf{x}\|_1 = \sum_{j=1}^n |x_j| \\ \text{subject to} & A\mathbf{x} = \mathbf{b}. \end{array}$$

It can be equivalently represented by (?)

$$\begin{array}{ll} \text{minimize} & \sum_{j=1}^n y_j \\ \text{subject to} & A\mathbf{x} = \mathbf{b},\ -\mathbf{y} \le \mathbf{x} \le \mathbf{y}; \end{array} \quad \text{or} \quad \begin{array}{ll} \text{minimize} & \sum_{j=1}^n (x'_j + x''_j) \\ \text{subject to} & A(\mathbf{x}' - \mathbf{x}'') = \mathbf{b},\ \mathbf{x}' \ge \mathbf{0},\ \mathbf{x}” \ge \mathbf{0}. \end{array}$$

Both are linear programs!

Sparsest Data Fitting continued

A better approximation of the objective can be accomplished by

$$\begin{array}{rl} \text{minimize} & \|\mathbf{x}\|_p := \left(\sum_{j=1}^n |x_j|^p\right)^{1/p} \\ \text{subject to} & A\mathbf{x} = \mathbf{b}; \end{array} \quad \text{or} \quad \text{minimize} \quad \|A\mathbf{x} - \mathbf{b}\|^2 + \mu \left(\sum_{j=1}^n |x_j|^p\right)^{1/p}$$

for some $0 < p < 1$, where $\mu > 0$ is a regularization parameter.

Or simply

$$\begin{array}{rl} \text{minimize} & \|\mathbf{x}\|_p^p := \left(\sum_{j=1}^n |x_j|^p\right) \\ \text{subject to} & A\mathbf{x} = \mathbf{b}; \end{array} \quad \text{or} \quad \text{minimize} \quad \|A\mathbf{x} - \mathbf{b}\|^2 + \beta \left(\sum_{j=1}^n |x_j|^p\right)$$

The former is a linearly constrained (nonconvex) optimization problem!

SOCP Example: Facility Location

$\mathbf{c}_j$ is the location of client $j = 1, 2, ..., m$, and $\mathbf{y}$ is the location of a facility to be built.

$$\text{minimize} \quad \sum_j \|\mathbf{y} - \mathbf{c}_j\|_p.$$

Or equivalently (?)

$$\begin{array}{ll} \text{minimize} & \sum_j \delta_j \\ \text{subject to} & \mathbf{y} + \mathbf{x}_j = \mathbf{c}_j, \ \|\mathbf{x}_j\|_p \le \delta_j, \ \forall j. \end{array}$$

This is a p-order conic linear program for $p \geq 1$.

In particular, when $p = 2$, it is an SOCP problem.

For simplicity, consider $m = 3$.

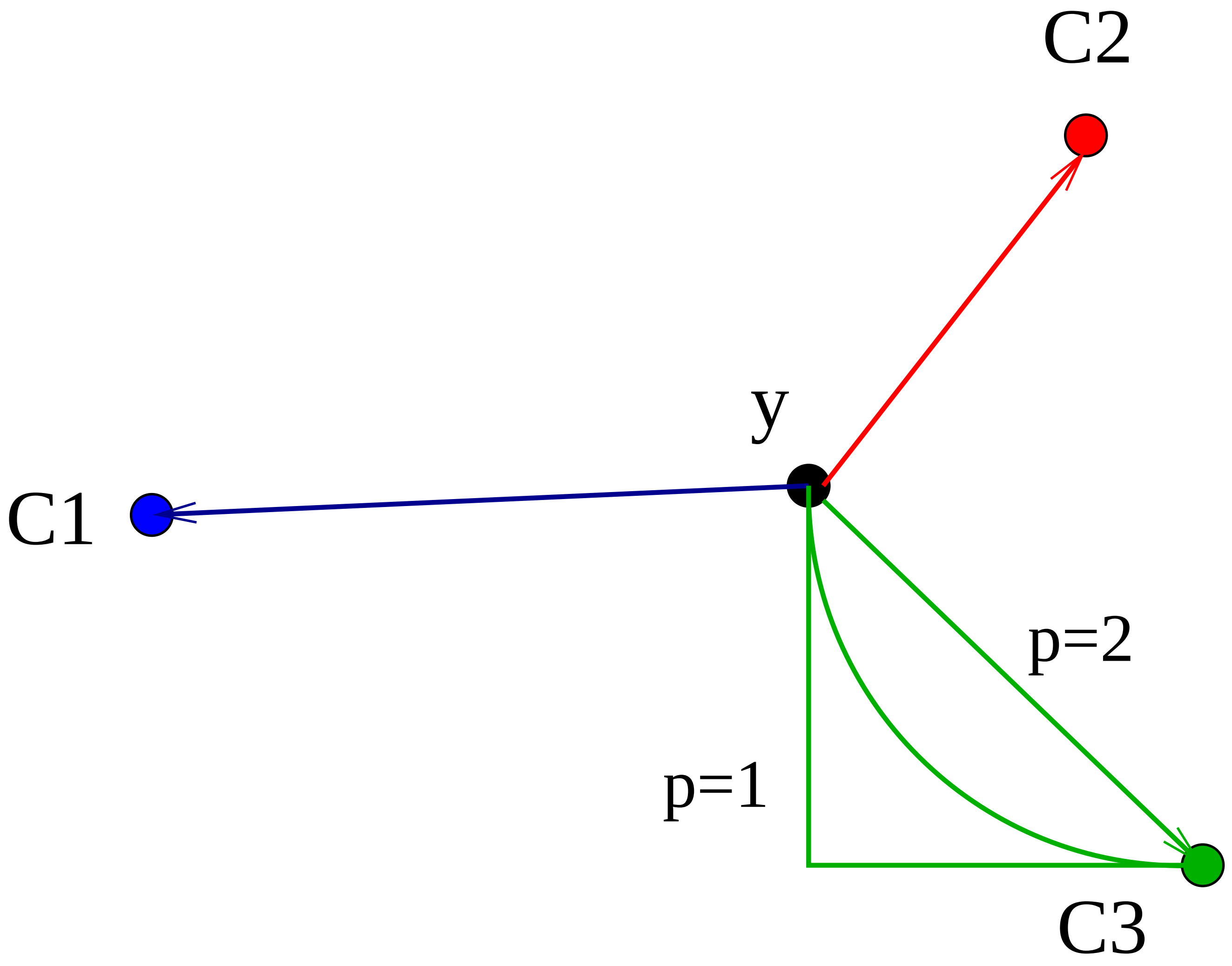


Figure 1: Facility Location at Point $\mathbf{y}$.

Graph Realization and Sensor Network Localization

Given a graph $G = (V, E)$ and sets of non–negative weights, say $\{d_{ij} : (i, j) \in E\}$, the goal is to compute a realization of G in the Euclidean space $\mathbf{R}^d$ for a given low dimension d, where the distance information is preserved.

More precisely: given anchors $\mathbf{a}_k \in \mathbf{R}^d$, $d_{ij} \in N_x$, and $\hat{d}_{kj} \in N_a$, find $\mathbf{x}_i \in \mathbf{R}^d$ such that

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 = d_{ij}^2, \ \forall\, (i, j) \in N_x, \ i < j,$$
$$\|\mathbf{a}_k - \mathbf{x}_j\|^2 = \hat{d}_{kj}^2, \ \forall\, (k, j) \in N_a,$$

This is a set of Quadratic Equations.

Does the system have a localization or realization of all $\mathbf{x}_j$'s? Is the localization unique? Is there a certification for the solution to make it reliable or trustworthy? Is the system partially localizable with certification?

It can be relaxed to SOCP (change "$=$" to "$\leq$") or SDP.

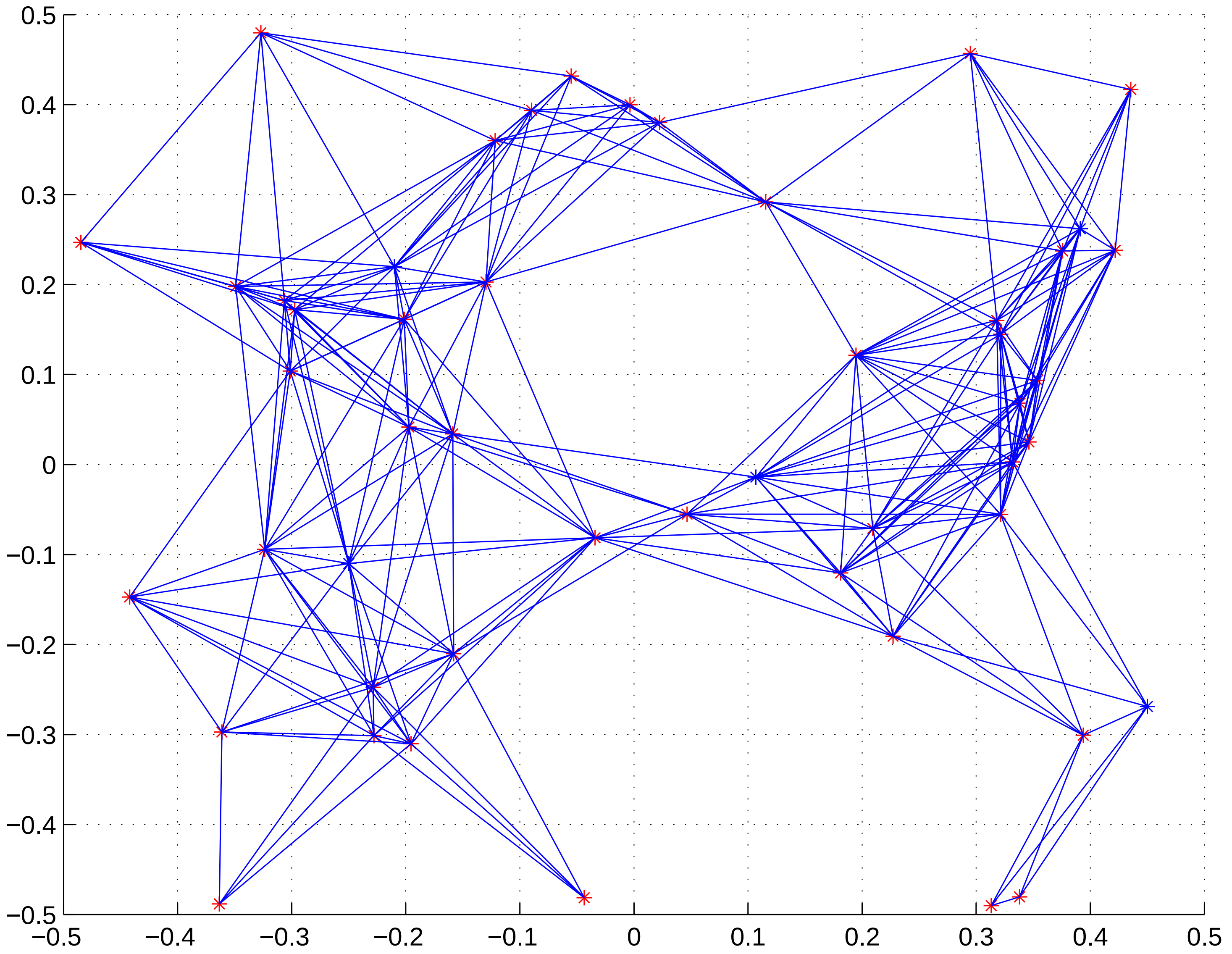


Figure 2: 50-node 2-D Sensor Localization.

Matrix Representation of SNL and SDP Relaxation

Let $X = [\mathbf{x}_1\ \mathbf{x}_2\ ...\ \mathbf{x}_n]$ be the $d \times n$ matrix that needs to be determined and $\mathbf{e}_j$ be the vector of all zero except 1 at the jth position. Then

$$\mathbf{x}_i - \mathbf{x}_j = X(\mathbf{e}_i - \mathbf{e}_j) \quad \text{and} \quad \mathbf{a}_k - \mathbf{x}_j = [I\ \ X](\mathbf{a}_k; -\mathbf{e}_j)$$

so that

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 = (\mathbf{e}_i - \mathbf{e}_j)^T X^T X(\mathbf{e}_i - \mathbf{e}_j)$$

$$\|\mathbf{a}_k - \mathbf{x}_j\|^2 = (\mathbf{a}_k; -\mathbf{e}_j)^T [I\ \ X]^T [I\ \ X](\mathbf{a}_k; -\mathbf{e}_j) =$$

$$(\mathbf{a}_k; -\mathbf{e}_j)^T \begin{pmatrix} I & X \\ X^T & X^T X \end{pmatrix} (\mathbf{a}_k; -\mathbf{e}_j).$$

Or, equivalently,

$$
\begin{aligned}
&(\mathbf{e}_i - \mathbf{e}_j)^T Y (\mathbf{e}_i - \mathbf{e}_j) = d_{ij}^2, \ \forall\, i, j \in N_x, \ i < j, \\
&(\mathbf{a}_k; -\mathbf{e}_j)^T \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} (\mathbf{a}_k; -\mathbf{e}_j) = \hat{d}_{kj}^2, \ \forall\, k, j \in N_a, \\
&Y = X^T X.
\end{aligned}
$$

Relax $Y = X^T X$ to $Y \succeq X^T X$, which is equivalent to matrix inequality:

$$
\begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} \succeq \mathbf{0}.
$$

This matrix has rank at least d; if it's d, then $Y = X^T X$, and the converse is also true.

The problem is now an SDP problem.

More CLP Examples: Portfolio Management

For expected return vector $\mathbf{r}$ and co-variance matrix V of an investment portfolio, one management model is:

$$\begin{array}{ll} \text{minimize} & \mathbf{x}^T V \mathbf{x} \\ \text{subject to} & \mathbf{r}^T\mathbf{x} \geq \mu, \\ & \mathbf{e}^T\mathbf{x} = 1,\ \mathbf{x} \geq \mathbf{0}, \end{array} \qquad \text{or simply} \qquad \begin{array}{ll} \text{minimize} & \mathbf{x}^T V \mathbf{x} \\ \text{subject to} & \mathbf{r}^T\mathbf{x} \geq \mu, \\ & \mathbf{e}^T\mathbf{x} = 1, \end{array}$$

where $\mathbf{e}$ is the vector of all ones.

This is a convex quadratic program.

More CLP Examples: Robust Portfolio Management

In applications, $\mathbf{r}$ and V may be estimated under various scenarios, say $\mathbf{r}_i$ and V_i for $i = 1, ..., m$. Then, we like

$$\begin{array}{rl} \text{minimize} & \max_i \mathbf{x}^T V_i \mathbf{x} \\ \text{subject to} & \min_i \mathbf{r}_i^T \mathbf{x} \geq \mu, \\ & \mathbf{e}^T \mathbf{x} = 1, \ \mathbf{x} \ \geq \ \mathbf{0}. \end{array} \quad \Rightarrow \quad \begin{array}{rl} \text{minimize} & \alpha \\ \text{subject to} & \mathbf{r}_i^T \mathbf{x} \geq \mu, \ \forall i \\ & \sqrt{\mathbf{x}^T V_i \mathbf{x}} \leq \alpha, \ \forall i \\ & \mathbf{e}^T \mathbf{x} = 1, \ \mathbf{x} \ \geq \ \mathbf{0}. \end{array}$$

If factorize $V_i = R_i^T R_i$ and let $\mathbf{y}_i = R_i \mathbf{x}$, we can rewrite the problem as

$$\begin{array}{rl} \text{minimize} & \alpha \\ \text{subject to} & \mathbf{r}_i^T \mathbf{x} \geq \mu, \ \mathbf{y}_i - R_i \mathbf{x} = \mathbf{0}, \ \forall i \\ & \|\mathbf{y}_i\| \leq \alpha, \ \forall i \\ & \mathbf{e}^T \mathbf{x} = 1, \ \mathbf{x} \ \geq \ \mathbf{0}. \end{array}$$

This is an SOCP-LP problem with additional benefits.

Recall Data Classification: Supporting Vector Machine

Like in logistic regression, suppose we have two-class discrimination data. We assign the first class with 1 and the second with -1 for a binary varible. A powerful discrimination method is the Supporting Vector Machine (SVM).

Let the first class data points i be given by $\mathbf{a}_i \in R^d$, $i = 1, ..., n_1$ and the second class data points j be given by $\mathbf{b}_j \in R^d$, $j = 1, ..., n_2$. We like to find a hyperplane to separate the two classes:

$$\begin{array}{rl}
\text{minimize} & \beta + \mu\|\mathbf{x}\|^2 \\
\text{subject to} & \mathbf{a}_i^T\mathbf{x} + x_0 + \beta \geq 1, \ \forall i, \\
 & \mathbf{b}_j^T\mathbf{x} + x_0 - \beta \leq -1, \ \forall j, \\
 & \beta \geq 0,
\end{array}$$

where μ is a fixed positive regularization parameter.

This is a quadratic program.

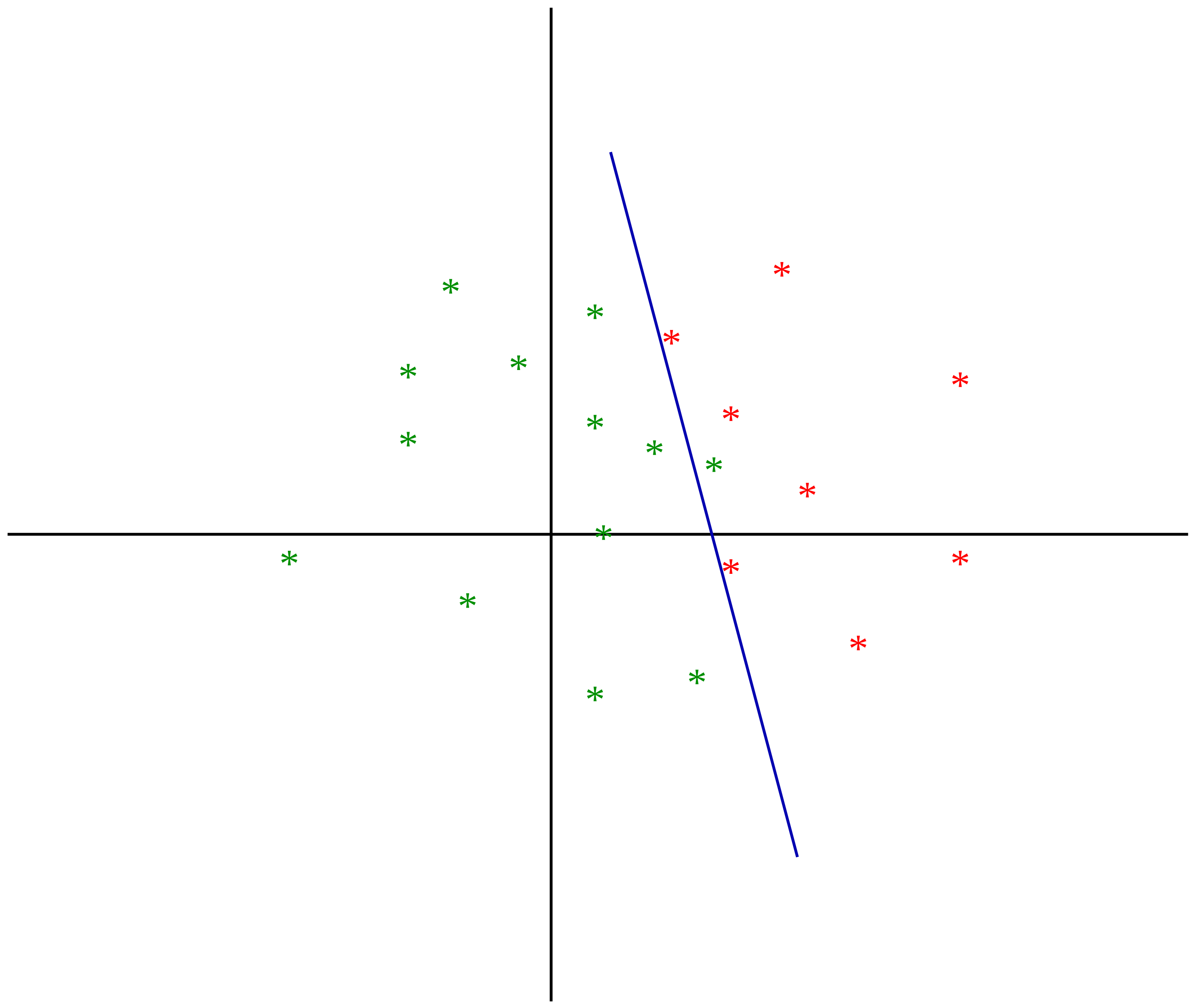


Figure 3: Linear Support Vector Machine

Stochastic Optimization and Learning

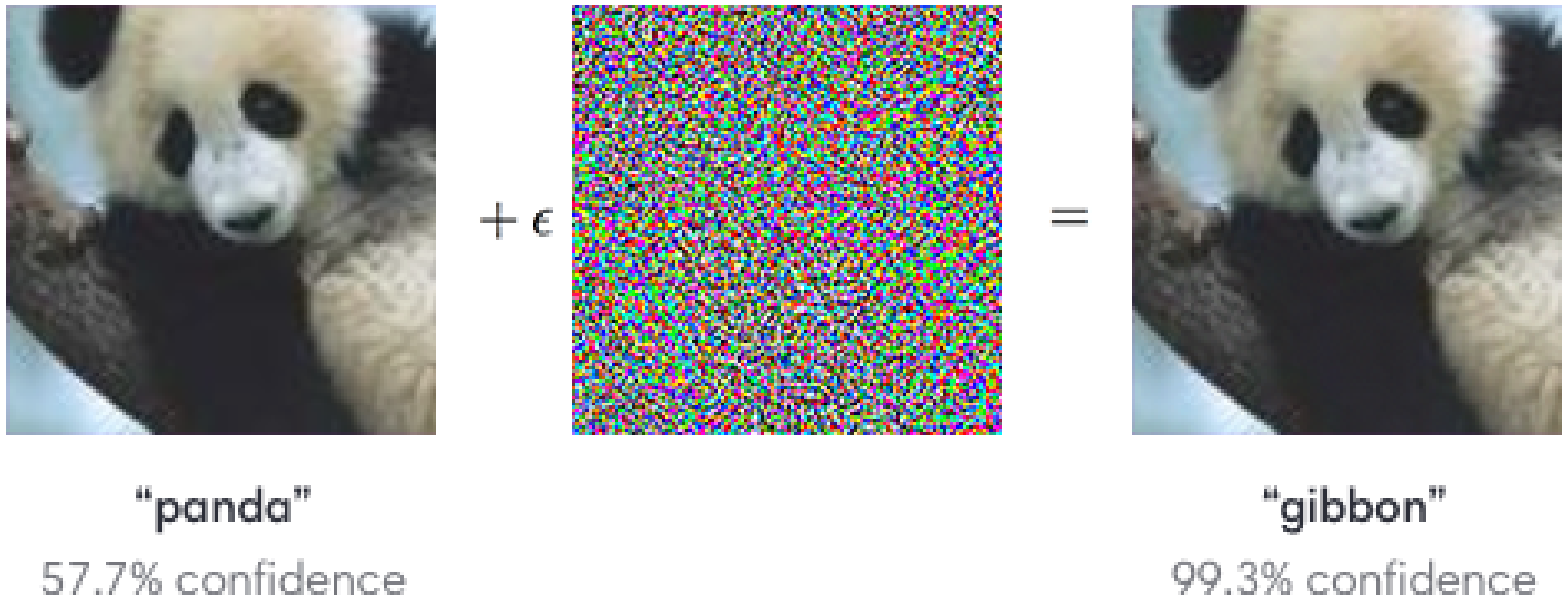
In real world, we most often do

$$\text{mimimize}_{\mathbf{x}\in X} \quad \mathbf{E}_{F_\xi}[h(\mathbf{x},\xi)] \tag{1}$$

where ξ represents random variables with the joint distribution F_ξ.

- Pros: In many cases, the expected value is a good measure of performance
- Cons: One has to know the exact distribution of ξ to perform the stochastic optimization so that we most frequently use sample distribution. Then, deviant from the assumed distribution may result in sub-optimal solutions. Even know the distribution, the solution/decision is generically risky.

Learning with Noises/Distortions



Goodfellow et al. [2014]

Distributionally Robust Optimization and Learning

Why error? Believing that the sample distribution is the true distribution...

In practice, although the exact distribution of the random variables may not be known, people usually know certain observed samples or training data and other statistical information. Thus, we can consider an enlarged distribution set $\mathcal{D}$ that confidently containing the sample distribution, and do

$$\text{minimize}_{\mathbf{x}\in X} \quad \max_{F_\xi\in\mathcal{D}} \mathbf{E}_{F_\xi}[h(\mathbf{x},\xi)] \tag{2}$$

In DRO, we consider a set of distributions $\mathcal{D}$ and choose one to minimize the expected value for the worst distribution in $\mathcal{D}$. When choosing $\mathcal{D}$, we need to consider the following:

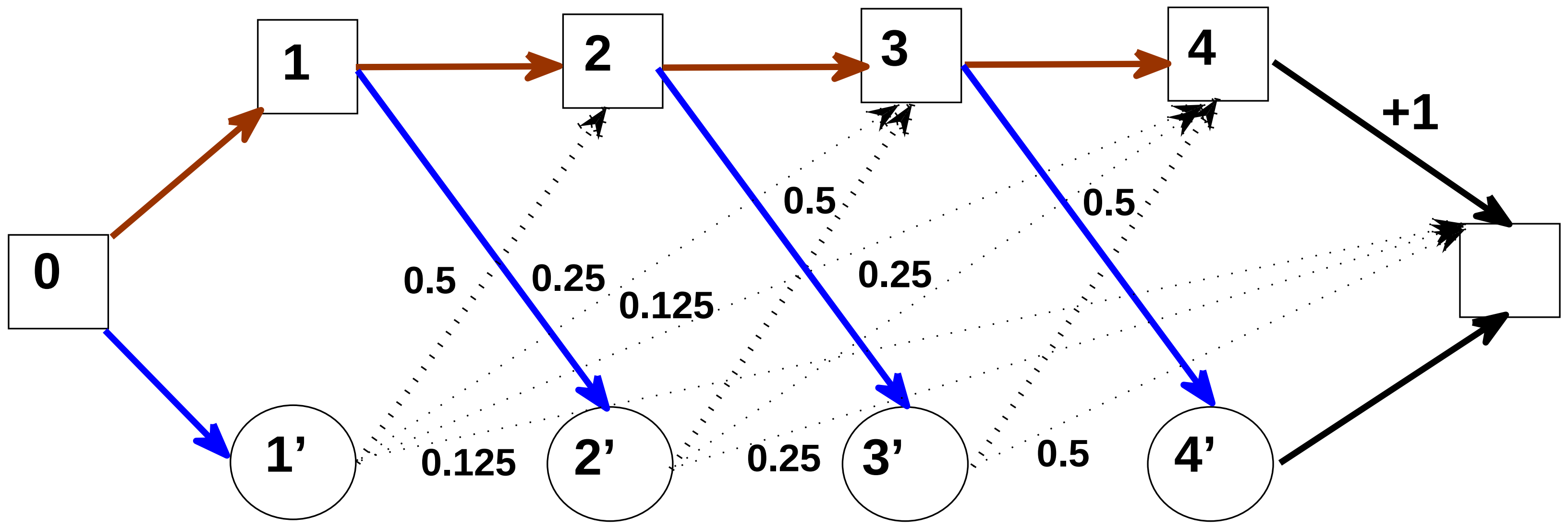
- Tractability
- Practical (Statistical) Meanings
- Performance (the potential loss comparing to the benchmark cases)

This is a nonlinear Min-Max optimization/zero-sum-game problem

The Markov Decision/Game Process

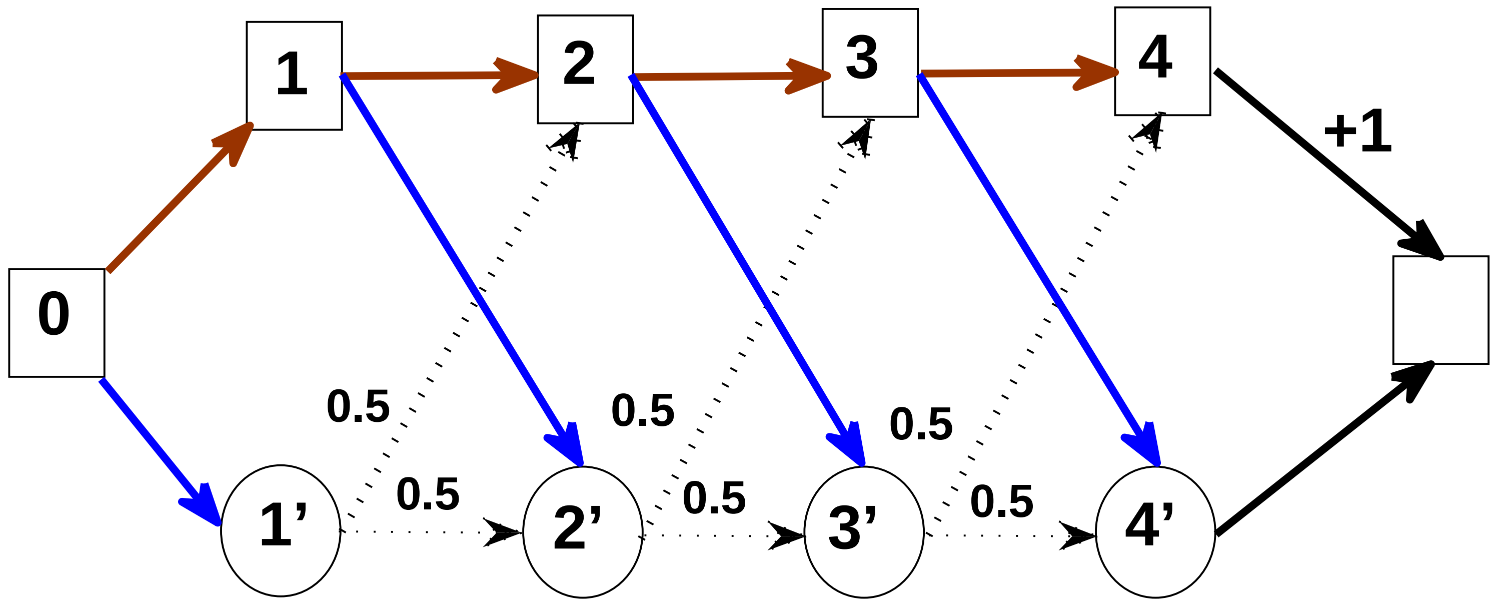
- Markov decision processes (MDPs) provide a mathematical framework for modeling sequential decision-making in situations where outcomes are partly random and partly under the control of a decision maker.
- Markov game processes (MGPs) provide a mathematical slidework for modeling sequential decision-making of two-person turn-based zero-sum game.
- MDGPs are useful for studying a wide range of optimization/game problems solved via dynamic programming, where it was known at least as early as the 1950s (cf. Shapley 1953, Bellman 1957).
- Modern applications include dynamic planning under uncertainty, reinforcement learning, social networking, and almost all other stochastic dynamic/sequential decision/game problems in Mathematical, Physical, Management and Social Sciences.

An MDGP Toy Example: Non-Game Setting



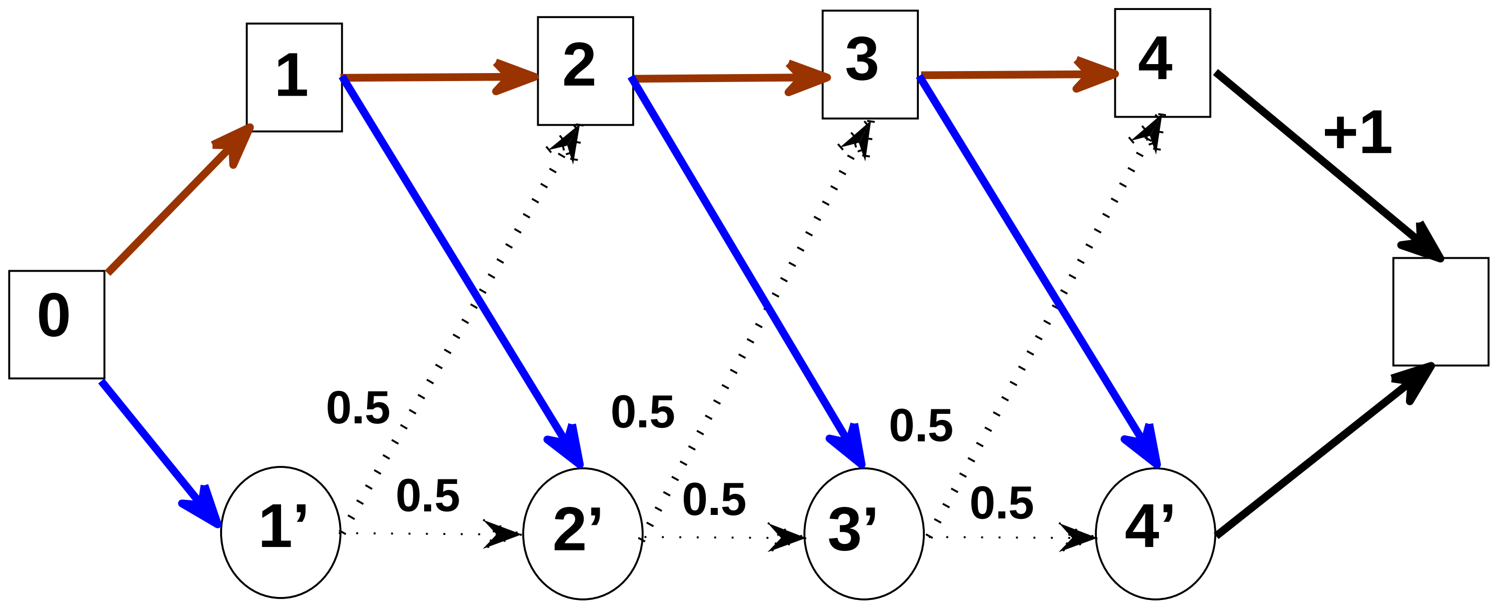
Actions are in red, blue and black; and all actions have zero cost except the state 4 to the termination state. Which actions to take from every state to minimize the total repeated cost?

Toy Example: Simplified Representation



Actions are in red, blue and black; and all actions have immediate zero cost except the state 4 to the termination state.

Toy Example: Game Setting



States $\{0,1,2\}$ minimize, while States $\{3,4\}$ maximize.

MDP Stationary Policy and Cost-to-Go Value

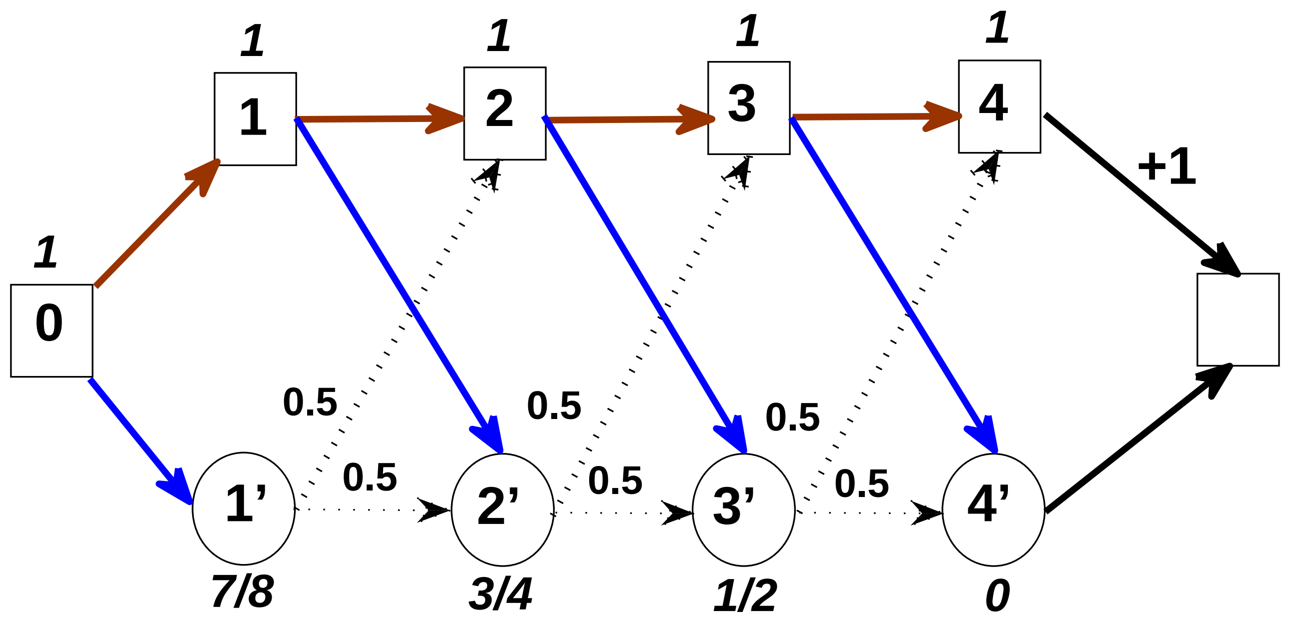
- A stationary policy for the decision maker is a function $\pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ that specifies an action in each state, $\pi_i \in \mathcal{A}_i$, that the decision maker will always choose; which also lead to a cost-to-go value for each state

- The MDP is to find a stationary policy to minimize/maximize the expected discounted sum over the infinite horizon with a discount factor $0 \leq \gamma < 1$:

$$\sum_{t=0}^{\infty} \gamma^t E[c^{\pi_{i^t}}(i^t, i^{t+1})].$$

- If the states are partitioned into two sets, one is to minimize and the other is to maximize the discounted sum, then the process becomes a two-person turn-based zero-sum stochastic game.

The Cost-to-Go Values of the States



Cost-to-go values on each state when actions in red are taken (the current policy is not optimal).

The Optimal Cost-to-Go Value Vector

Let $\mathbf{y} \in \mathbf{R}^m$ represent the cost-to-go values of the m states, ith entry for ith state, of a given policy.

The MDP problem entails choosing the optimal value vector $\mathbf{y}^*$ such that it satisfies:

$$y_i^* = \min\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*, \ \forall j \in \mathcal{A}_i\}, \ \forall i,$$

with optimal policy

$$\pi_i^* = \arg\min\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*, \ \forall j \in \mathcal{A}_i\}, \ \forall i.$$

In the Game setting, the conditions becomes:

$$y_i^* = \min\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*, \ \forall j \in \mathcal{A}_i\}, \ \forall i \in I^-,$$

and

$$y_i^* = \max\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*, \ \forall j \in \mathcal{A}_i\}, \ \forall i \in I^+.$$

They both are fix-point or saddle-point problems.