

SEMIDEFINITE PROGRAMMING

WHAT'S AN SDP

linear program

$$\underset{x}{\text{minimize}} \quad \langle x, c \rangle$$

$$\text{subject to} \quad Ax = b$$

$$x \geq 0$$

semdefinite program

$$\underset{X}{\text{minimize}} \quad \langle X, C \rangle$$

$$\text{subject to} \quad \langle A_i, X \rangle = b_i, \quad \forall i$$

$$X \succeq 0$$

WHAT'S AN SDP

semdefinite program

$$\underset{X}{\text{minimize}} \quad \langle X, C \rangle$$

$$\text{subject to} \quad \langle A_i, X \rangle = b_i, \quad \forall i$$

$$X \succeq 0$$

LP with infinite constraint

$$\underset{X}{\text{minimize}} \quad \langle X, C \rangle$$

$$\text{subject to} \quad \langle A_i, X \rangle = b_i, \quad \forall i$$

$$u^T X u = \langle u u^T, X \rangle \geq 0, \quad \forall u$$

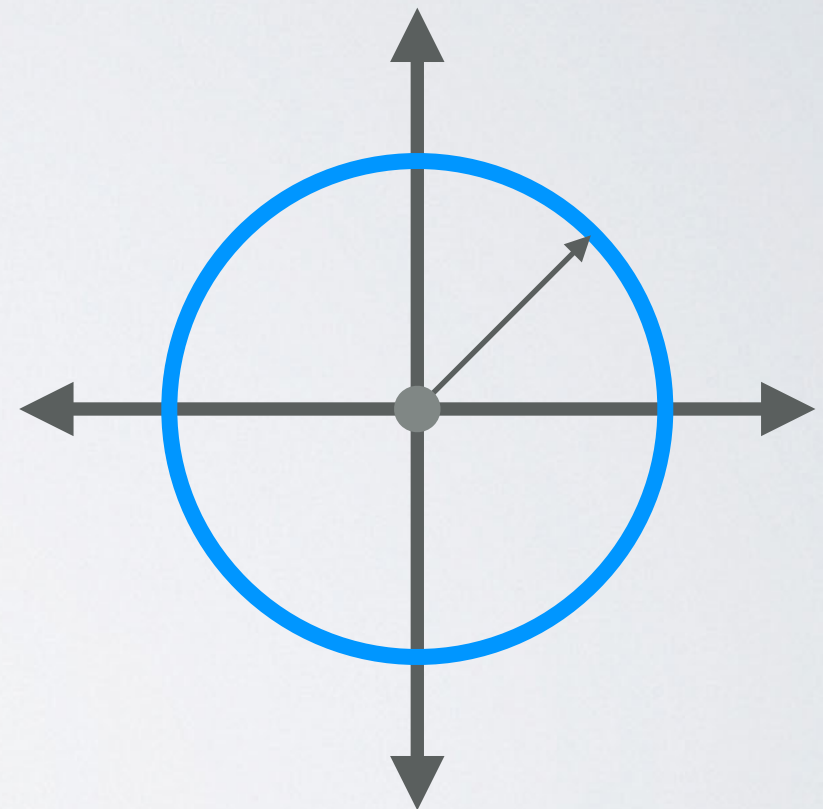
WHY DO WE CARE?

$$Y = \begin{pmatrix} | & | & | & | \\ y_1 & y_2 & \cdots & y_n \\ | & | & | & | \end{pmatrix}$$

minimize $f(Y)$
subject to $\|y_i\| = 1$

Is this convex?

constraint: $\|y_i\| = 1$



WHY DO WE CARE?

$$Y = \begin{pmatrix} | & | & | & | \\ y_1 & y_2 & \cdots & y_n \\ | & | & | & | \end{pmatrix}$$

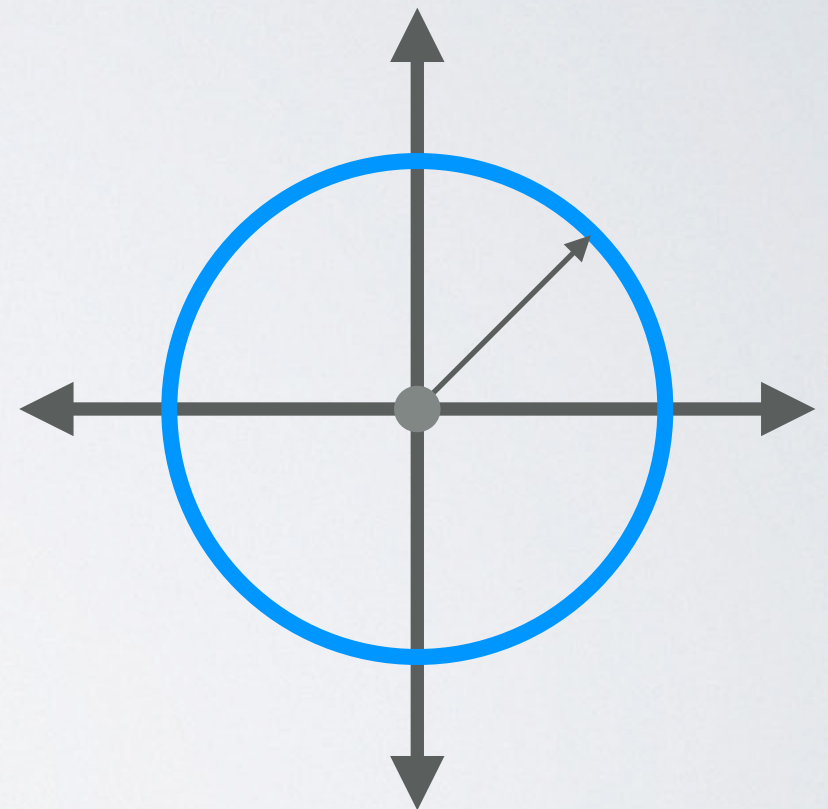
$$\begin{array}{ll} \text{minimize} & f(Y) \\ \text{subject to} & \|y_i\| = 1 \end{array}$$

convexify

$$\text{Let } X = Y^T Y \xrightarrow{\text{red arrow}} X_{ii} = \|y_i\|^2$$

$$\begin{array}{ll} \text{minimize} & g(X) \\ \text{subject to} & X_{ii} = 1 \end{array}$$

$$\text{constraint: } \|y_i\| = 1$$



EXAMPLE: STRUCTURAL PROGRAMMING

$$Y = \begin{pmatrix} | & | & | & | \\ y_1 & y_2 & \dots & y_n \\ | & | & | & | \end{pmatrix}$$

Find a structure given the following...

I tell you the **length** of all vectors

$$\|y_i\| = l_i$$

I tell you the **distance** between vectors

$$d_{ij}^l \leq \|y_i - y_j\| \leq d_{ij}^u$$

I tell your the **angle** between vectors

$$a_{ij}^l \leq \frac{\langle y_i, y_j \rangle}{\|y_i\| \|y_j\|} \leq a_{ij}^u$$

$$\text{Let } X = Y^T Y$$


EXAMPLE: STRUCTURAL PROGRAMMING

$$X = Y^T Y = \begin{pmatrix} y_1^T y_1 & y_1^T y_2 & y_1^T y_3 \\ y_2^T y_1 & y_2^T y_2 & y_2^T y_3 \\ y_3^T y_1 & y_3^T y_2 & y_3^T y_3 \end{pmatrix}$$

I tell you the **length** of all vectors

$$\|y_i\| = l_i$$

**convex
constraint**

$$X_{ii} = l_i$$


EXAMPLE: STRUCTURAL PROGRAMMING

$$X = Y^T Y = \begin{pmatrix} y_1^T y_1 & y_1^T y_2 & y_1^T y_3 \\ y_2^T y_1 & y_2^T y_2 & y_2^T y_3 \\ y_3^T y_1 & y_3^T y_2 & y_3^T y_3 \end{pmatrix}$$

I tell you the **distance** between vectors $d_{ij}^l \leq \|y_i - y_j\| \leq d_{ij}^u$

$$\begin{aligned} \|y_i - y_j\|^2 &= \|y_i\|^2 + 2\langle y_i, y_j \rangle + \|y_j\|^2 \\ &= l_i^2 + X_{ij} + l_j^2 \end{aligned}$$

convex constraint

$$(d_{ij}^l)^2 - l_i^2 - l_j^2 \leq X_{ij} \leq (d_{ij}^u)^2 - l_i^2 - l_j^2$$

EXAMPLE: STRUCTURAL PROGRAMMING

$$X = Y^T Y = \begin{pmatrix} y_1^T y_1 & y_1^T y_2 & y_1^T y_3 \\ y_2^T y_1 & y_2^T y_2 & y_2^T y_3 \\ y_3^T y_1 & y_3^T y_2 & y_3^T y_3 \end{pmatrix}$$

I tell your the **angle** between vectors $a_{ij}^l \leq \frac{\langle y_i, y_j \rangle}{\|y_i\| \|y_j\|} \leq a_{ij}^u$

$$\frac{X_{ij}}{l_i l_j} = a_{ij}$$

convex constraint

$$l_i l_j a_{ij}^l \leq X_{ij} \leq l_i l_j a_{ij}^u$$

EXAMPLE

Find “largest” X that satisfies structural constraints

maximize $\log \det X$

subject to

length constraints

$$X_{ij} = l_{ij}$$

angle/distance constraints

$$\alpha_{ij} \leq X_{ij} \leq \beta_{ij}$$

SDP constraint

$$X \succeq 0$$



why?

recovery solution using **Cholesky** factorization

$$X = Y^T Y$$

what if the problem is infeasible?

SOLUTION SYMMETRY

Is it possible to find a convex problem in Y that solves this?

I tell you the **length** of all vectors

$$\|y_i\| = l_i$$

I tell you the **distance** between vectors $d_{ij}^l \leq \|y_i - y_j\| \leq d_{ij}^u$

I tell your the **angle** between vectors $a_{ij}^l \leq \frac{\langle y_i, y_j \rangle}{\|y_i\| \|y_j\|} \leq a_{ij}^u$

recovery solution using **Cholesky** factorization

$$X = Y^T Y = Y^T U^T U Y = (U Y)^T (U Y)$$

CONVEX RELAXATION

replace a non-convex set
with a (larger) convex set

relaxation



FAMOUS EXAMPLE: MAX CUT

Value of cut =
number of edges sliced

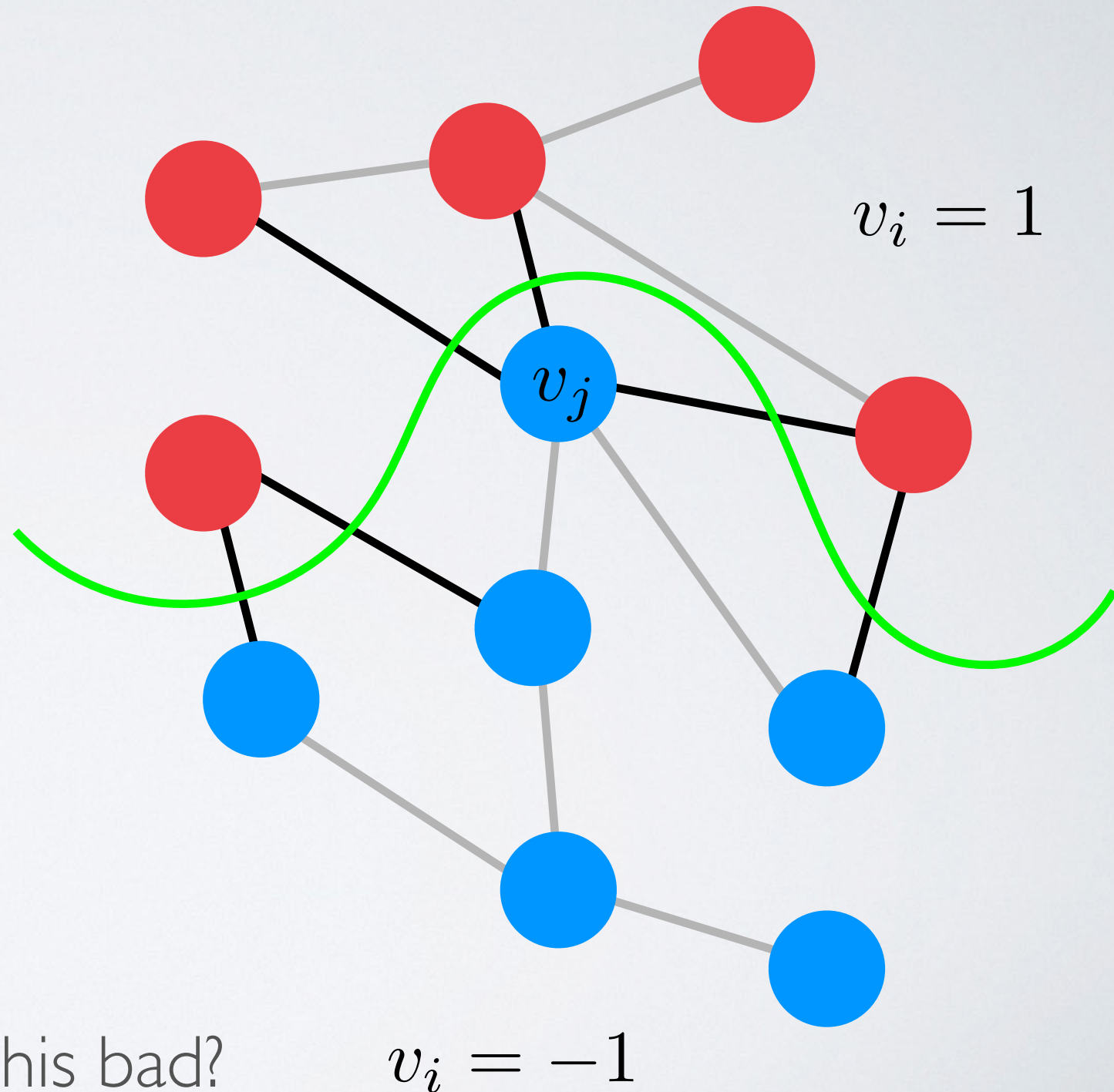
$$\text{maximize} \quad \sum_{(ij) \in E} \frac{1 - v_i v_j}{2}$$

subject to $v_i \in \{-1, 1\}$

easy relaxation

$$v_i \in [-1, 1]$$

why is this bad?

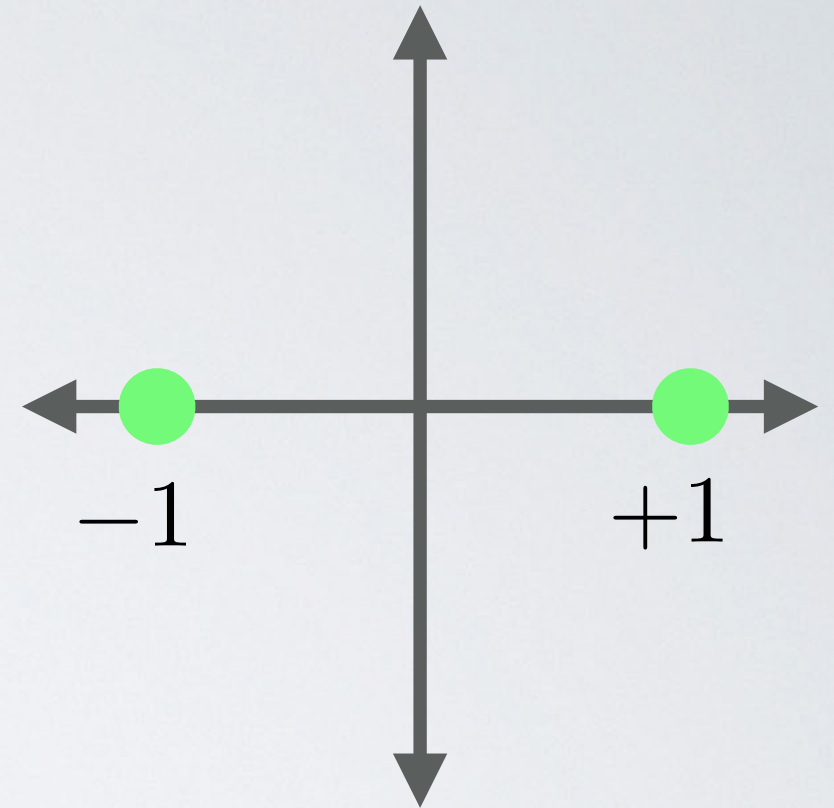


GOEMANS-WILLIAMSON

integer program

$$\text{maximize} \quad \sum_{(ij) \in E} \frac{1 - v_i v_j}{2}$$

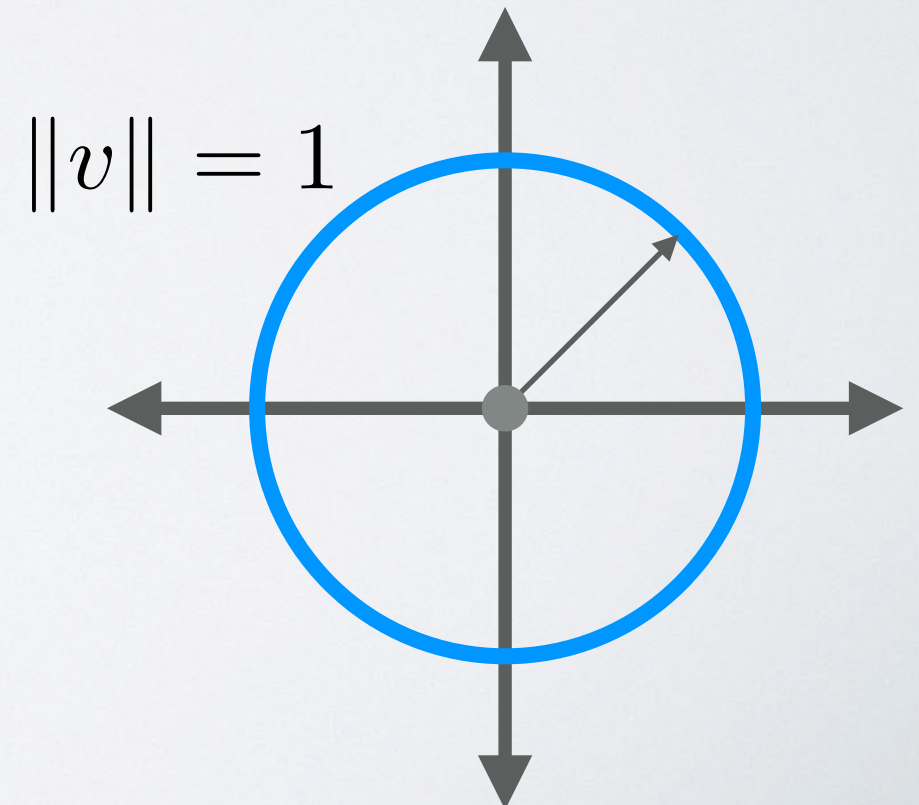
$$\text{subject to} \quad v_i \in \{-1, 1\}$$



relaxation

$$\text{maximize} \quad \sum_{(ij) \in E} \frac{1 - \langle v_i, v_j \rangle}{2}$$

$$\text{subject to} \quad \|v_i\| = 1$$



GOEMANS-WILLIAMSON

relaxation

$$\text{maximize} \quad \sum_{(i,j) \in E} \frac{1 - \langle v_i, v_j \rangle}{2}$$

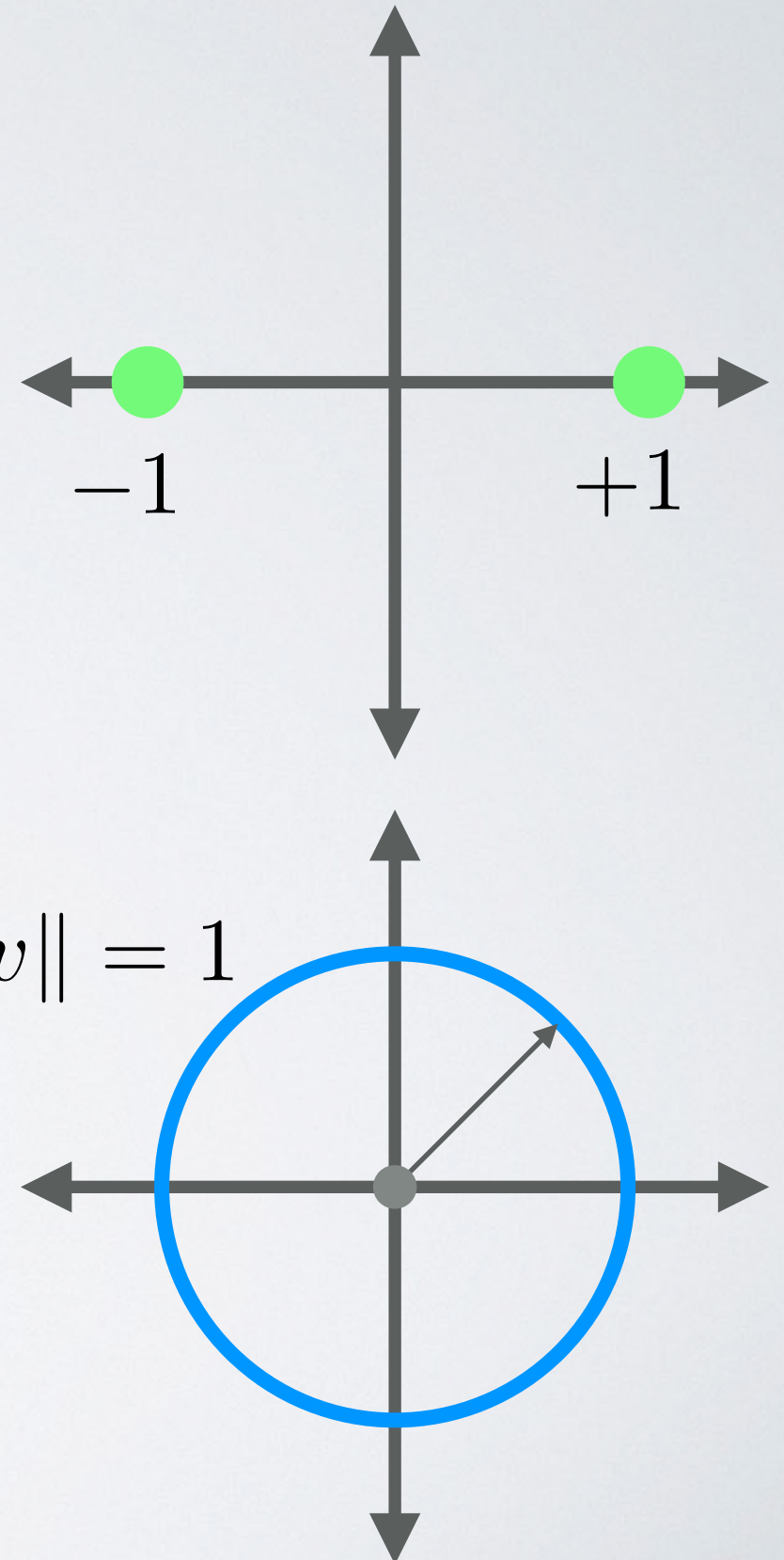
$$\text{subject to} \quad \|v_i\| = 1$$

SDP

$$\text{maximize} \quad \sum_{(i,j) \in E} \frac{1 - X_{ij}}{2}$$

$$\text{subject to} \quad X_{ii} = 1, \quad \forall i$$

$$X \succeq 0$$



GOEMANS-WILLIAMSON

SDP

$$\text{maximize} \quad \sum_{(i,j) \in E} \frac{1 - X_{ij}}{2}$$

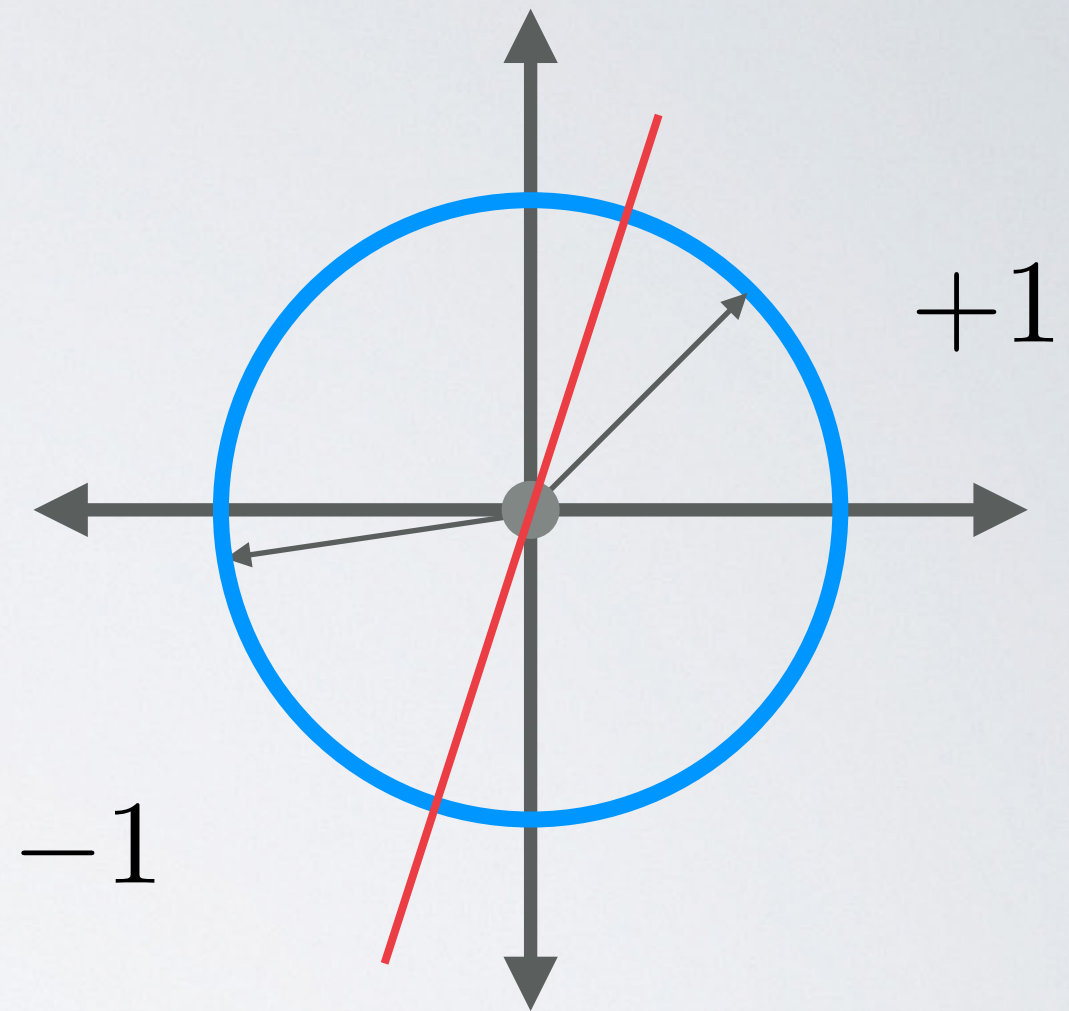
$$\text{subject to} \quad X_{ii} = 1, \quad \forall i$$

$$X \succeq 0$$

Use random rounding
to convert to original solution

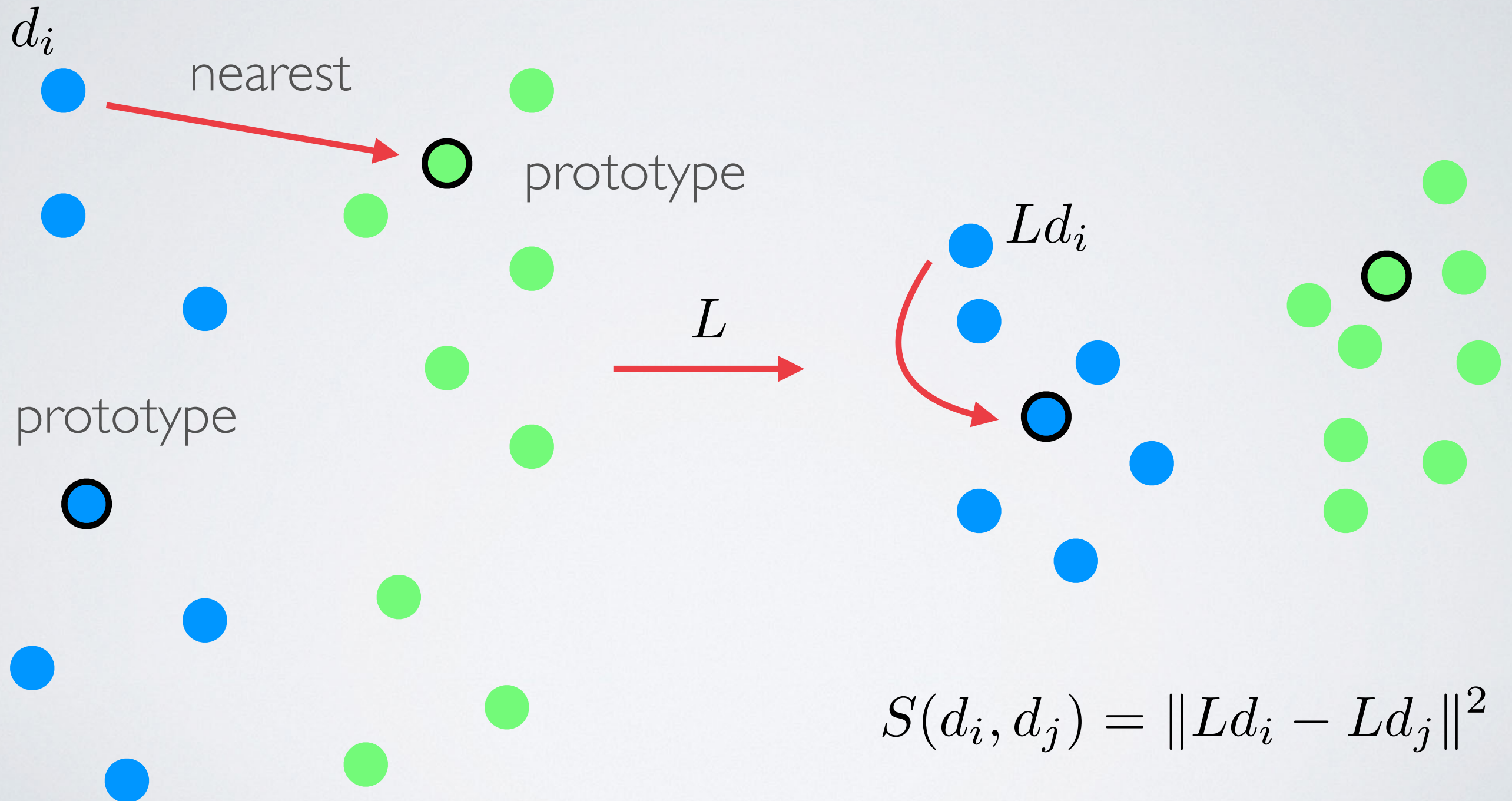
Expected value = 0.87856 (optimal)

This is the **best** polynomial-time approximation
if you believe the unique games conjecture



METRIC LEARNING

learn a similarity measure between data point
that preserves classification



LARGE-MARGIN NEAREST NEIGHBORS

given data with multiple labels

feature vectors

$$d_i \in \mathbb{R}^n$$

labels

$$l_i \in \{1, 2, 3, \dots, c\}$$

neighborhood

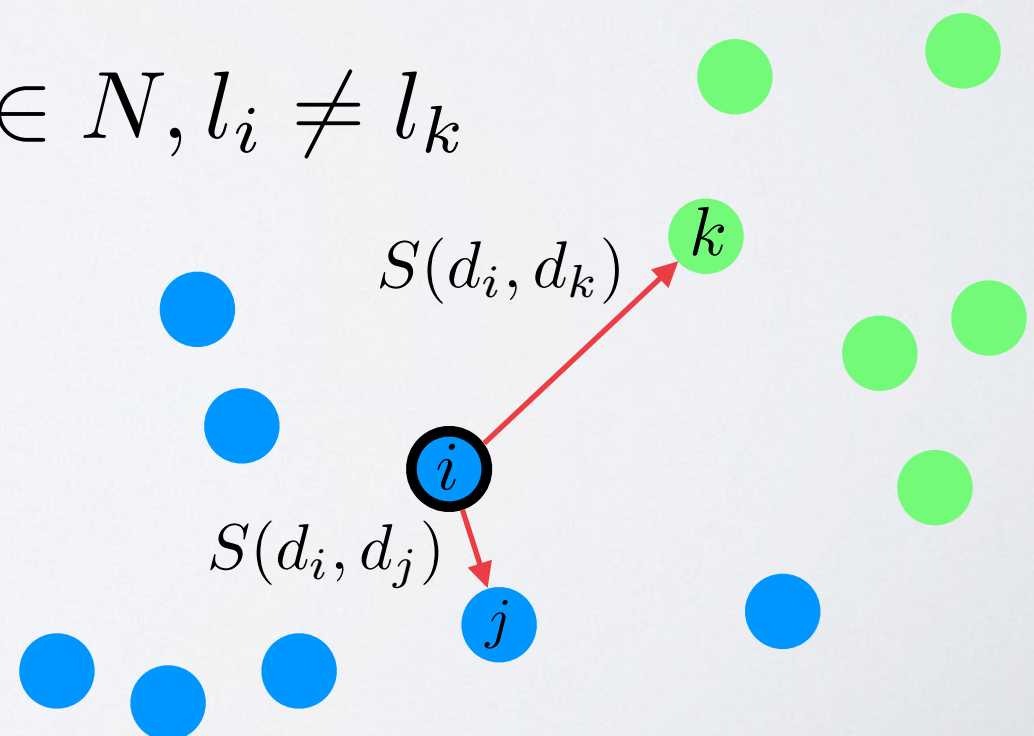
$$N_i = \{d_j\}$$

Every point should lie closer to “neighbors”
than members of another other class

$$S(d_i, d_j) + 1 \leq S(d_i, d_k), \quad \forall (i, j) \in N, l_i \neq l_k$$

remember

$$S(d_i, d_j) = \|Ld_i - Ld_j\|^2$$



CONVEX FORMULATION

neighbors should be close together

minimize $\sum_{(i,j) \in N} S(d_i, d_j) + \sum_{i,j,k} \eta_{ijk}$

subject to $S(d_i, d_j) + 1 \leq S(d_i, d_k) + \eta_{ijk}, \quad \forall (i,j) \in N, l_i \neq l_k$

$\eta_{ijk} \geq 0$

classes should be far apart

convexify

$$S(d_i, d_j) = \|Ld_i - Ld_j\|^2 = (d_i - d_j)^T L^T L (d_i - d_j) = (d_i - d_j)^T M (d_i - d_j)$$

CONVEX FORMULATION

$$\text{minimize} \quad \sum_{(i,j) \in N} S(d_i, d_j) + \sum_{i,j,k} \eta_{ijk}$$

$$\text{subject to} \quad S(d_i, d_j) + 1 \leq S(d_i, d_k) + \eta_{ijk}, \quad \forall (i, j) \in N, l_i \neq l_k$$

$$\eta_{ijk} \geq 0$$

convexify

$$S(d_i, d_j) = \|Ld_i - Ld_j\|^2 = (d_i - d_j)^T L^T L (d_i - d_j) = (d_i - d_j)^T M (d_i - d_j)$$

linear objective / linear constraints

$$\text{minimize} \quad \sum_{(i,j) \in N} \|d_i - d_j\|_M^2 + \sum_{i,j,k} \eta_{ijk}$$

$$\text{subject to} \quad \|d_i - d_j\|_M^2 + 1 \leq \|d_i - d_k\|_M^2 + \eta_{ijk}, \quad \forall (i, j) \in N, l_i \neq l_k$$

$$\eta_{ijk} \geq 0$$

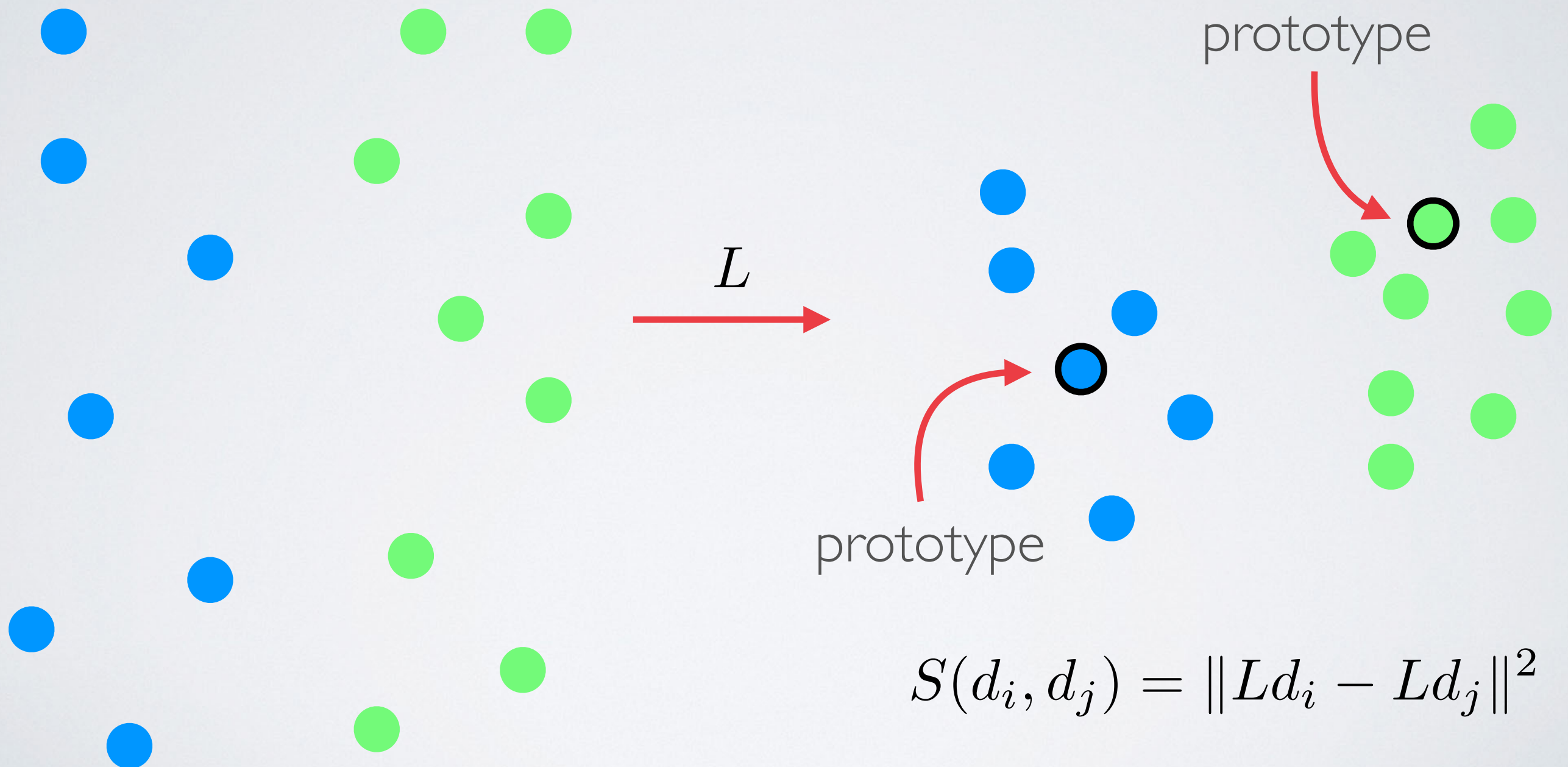
$$\text{SDP} \rightarrow M \succeq 0$$

linear



GENERALIZATION

learn prototypes and mapping at the same time



GENERALIZATION

forward model

Given

Data matrix: D

Prototypes: P

Learn the linear transform

$$L = \arg \min_L \sum_i \|p_i - Ld_i\|^2 + \lambda \|L\|^2$$

$$L = \arg \min_L \|PJ - LD\|^2 + \lambda \|L\|^2$$

ridge
penalty



column selector



L is a function of P

$$L = \underbrace{PJ D^T (D D^T + \lambda I)^{-1}}_A = PA$$

PROJECTED DATA

$$L = PJD^T(DD^T + \lambda I)^{-1} = PA$$

Projected data: $LD = PAD = P\hat{D}$

$$\|p_{l_i} - Ld_i\|^2 + 1 \leq \|p_{l_j} - Ld_i\|^2, \quad \forall l_j \neq l_i$$

$$\|p_{l_i} - PAD_i\|^2 + 1 \leq \|p_{l_j} - PAD_i\|^2, \quad \forall l_j \neq l_i$$

$$\|p_{l_i} - P\hat{d}_i\|^2 + 1 \leq \|p_{l_j} - P\hat{d}_i\|^2, \quad \forall l_j \neq l_i$$

in matrix form

$$\|Pe_{l_i} - P\hat{d}_i\|^2 + 1 \leq \|Pe_{l_j} - P\hat{d}_i\|^2, \quad \forall l_j \neq l_i$$

FORMULATION

$$\underset{P}{\text{minimize}} \quad \sum_{ijk} \eta_{ijk}$$

$$\text{subject to} \quad \|Pe_{l_i} - P\hat{d}_i\|^2 + 1 \leq \|Pe_{l_j} - P\hat{d}_i\|^2 + \eta_{ijk}, \quad \forall l_j \neq l_i$$

convexify

$$\|Pz\|^2 = z^T P^T P z = z^T M z$$

SDP

$$\underset{M}{\text{minimize}} \quad \sum_{ijk} \eta_{ijk}$$

$$\text{subject to} \quad \|e_{l_i} - \hat{d}_i\|_M^2 + 1 \leq \|e_{l_j} - \hat{d}_i\|_M^2 + \eta_{ijk}, \quad \forall l_j \neq l_i$$

$$M \succeq 0$$

NUMERICS

semidefinite program

$$\underset{X}{\text{minimize}} \quad \langle X, C \rangle$$

$$\text{subject to} \quad \langle A_i, X \rangle = b_i, \quad \forall i$$

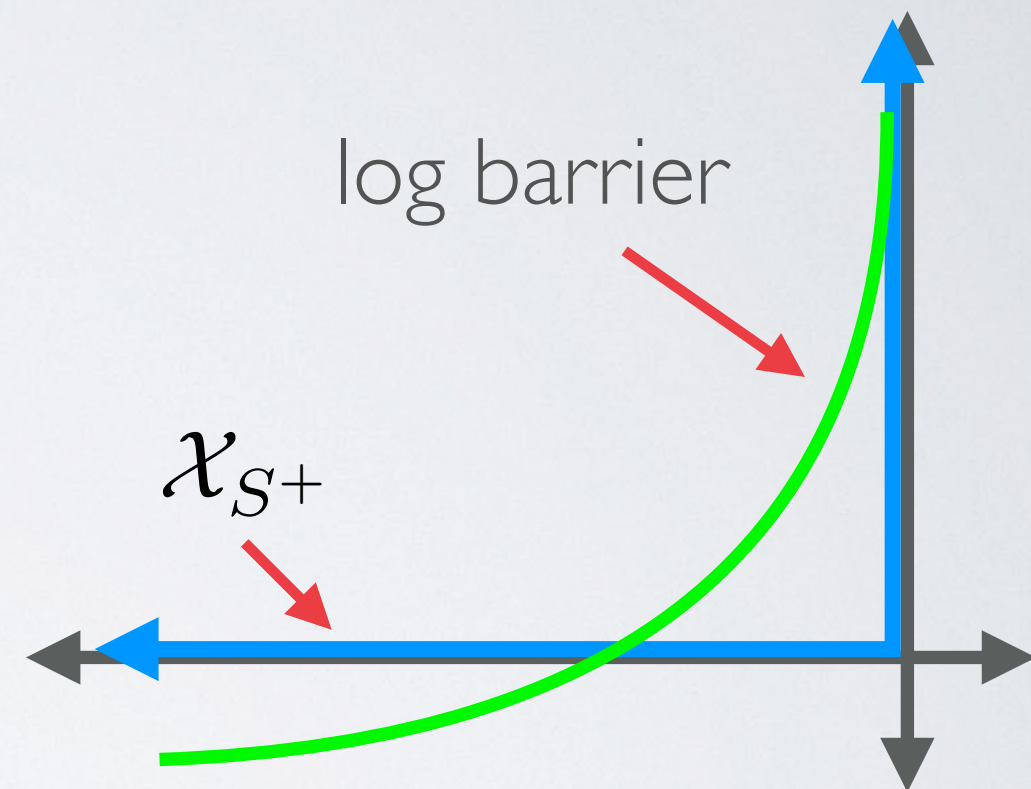
$$X \succeq 0$$

barrier: $-\log \det X$

$$\underset{X}{\text{minimize}} \quad \langle X, C \rangle - \mu \log \det X$$

$$\text{subject to} \quad \langle A_i, X \rangle = b_i, \quad \forall i$$

$$X \succeq 0$$



NUMERICS

log-barrier form

$$\begin{aligned} &\text{minimize} && \langle X, C \rangle - \mu \log \det X \\ &\text{subject to} && \langle A_i, X \rangle = b_i, \quad \forall i \\ &&& X \succeq 0 \end{aligned}$$

solve by Newton's method

barrier (primal) method

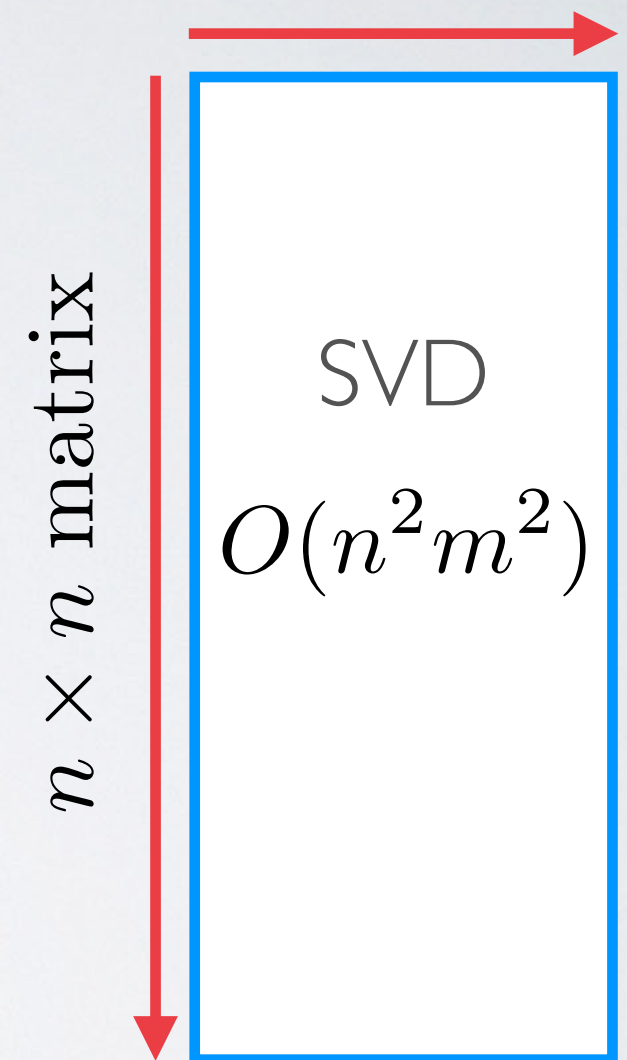
$n \times n$ matrix $\rightarrow n^2 \times n^2$ system
 $O(n^6)$ complexity

primal-dual method

$n^2 \times m$ matrix

in practice

#constraints = m



Why IP methods?
you're **doomed** to
factorization

ALTERNATIVES

simple methods
may be faster for specific problem forms

FBS / ADMM methods

$$\begin{array}{ll} \text{minimize} & f(X) \\ \text{subject to} & X \succeq 0 \end{array} \quad \longrightarrow \quad L(X, Y, \lambda) = f(X) + \mathcal{X}_{S^+}(Y) + \frac{\tau}{2} \|X - Y - \lambda\|^2$$

Approximation methods

$$X = L^T L$$

$$\begin{array}{ll} \text{minimize} & f(X) \\ \text{subject to} & X \succeq 0 \end{array} \quad \longrightarrow \quad \underset{L}{\text{minimize}} \quad f(L^T L)$$

example: Wiberg method

WHEN TO USE SDP?

Only when you have to!

your problem is non-convex

can be effectively relaxed

you have a great solver

small number of unknowns:

m constraints, $n \times n$ matrix: $O(mn^2)$ runtime