Projected Power Method: An Efficient Algorithm for Discrete Assignment

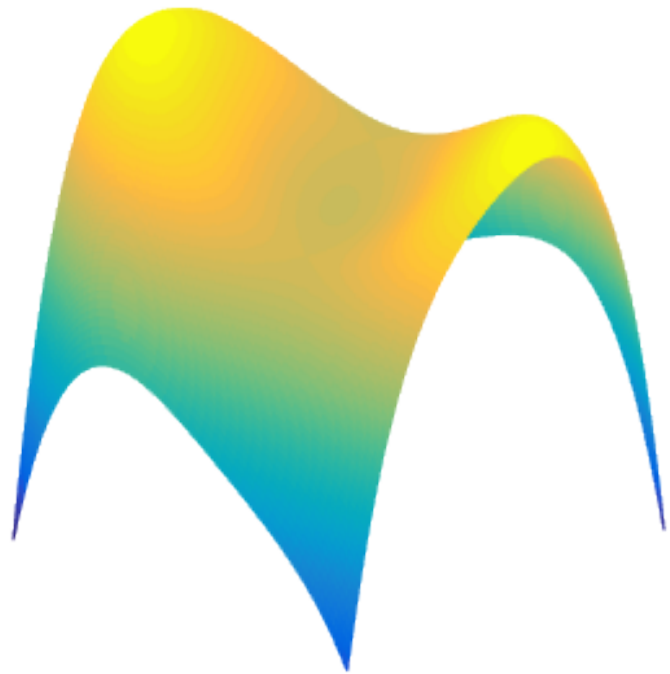
Yuxin Chen

Department of Electrical Engineering, Princeton University
Joint work with Emmanuel Candes, David Tse, Govinda Kamath, Changho Suh, Tao Zhang

Nonconvex problems are everywhere

Maximum likelihood is usually nonconvex

$$\begin{array}{rcll} \text{maximize}_{\boldsymbol{x}} & \ell(\boldsymbol{x};\boldsymbol{y}) & \rightarrow & \text{may be nonconvex} \\ \text{subj. to} & \boldsymbol{x} \in \mathcal{S} & \rightarrow & \text{may be nonconvex} \end{array}$$

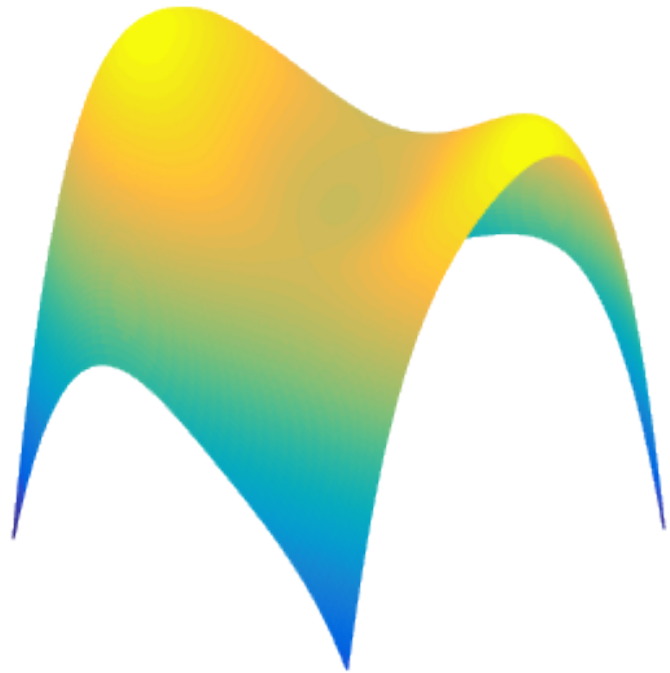


Nonconvex problems are everywhere

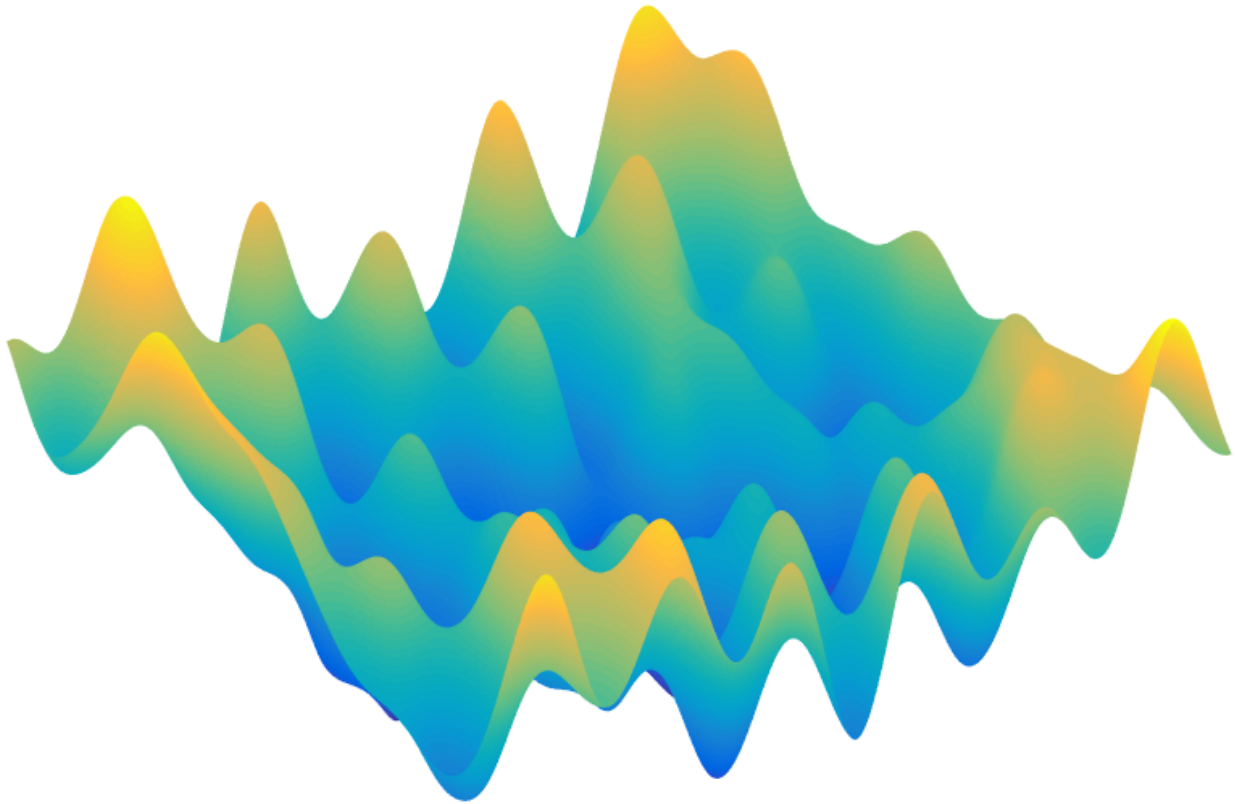
Maximum likelihood is usually nonconvex

$$\begin{array}{rcll} \text{maximize}_{\boldsymbol{x}} & \ell(\boldsymbol{x};\boldsymbol{y}) & \rightarrow & \text{may be nonconvex} \\ \text{subj. to} & \boldsymbol{x} \in \mathcal{S} & \rightarrow & \text{may be nonconvex} \end{array}$$

- low-rank matrix completion
- graph clustering
- dictionary learning
- graph matching
- ...



Nonconvex optimization may be super scary



There may be bumps everywhere and exponentially many local optima

e.g. 1-layer neural net (Auer, Herbster, Warmuth '96; Vu '98)

Nonconvex optimization may be super scary

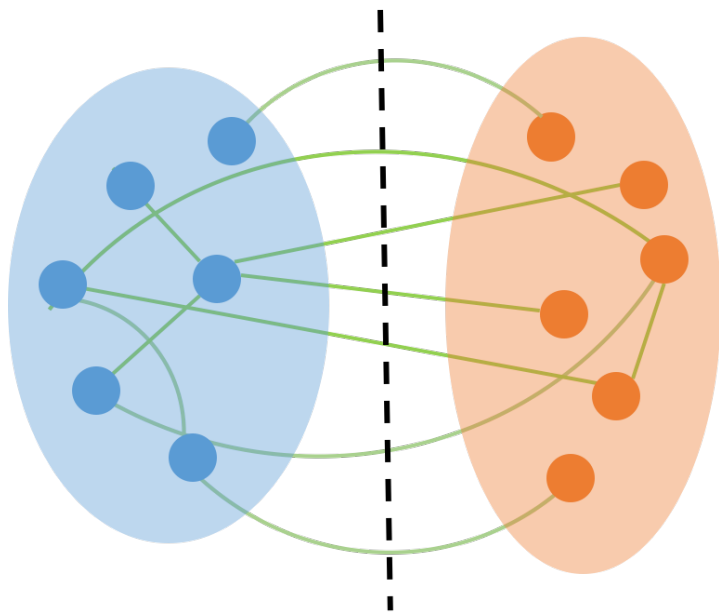
There may be bumps everywhere and exponentially many local optima

e.g. 1-layer neural net (Auer, Herbster, Warmuth '96; Vu '98)

Solving discrete problems is hard

Finding maximum cut in a graph is

$$\begin{aligned} \text{maximize}_{\boldsymbol{x}} \quad & \boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{x} \\ \text{subj. to} \quad & x_i^2 = 1, \quad i = 1, \cdots, n \end{aligned}$$



Solving discrete problems is hard

"I can't find an efficient algorithm, but neither can all these people."

Fig credit: coding horror

Convex relaxation

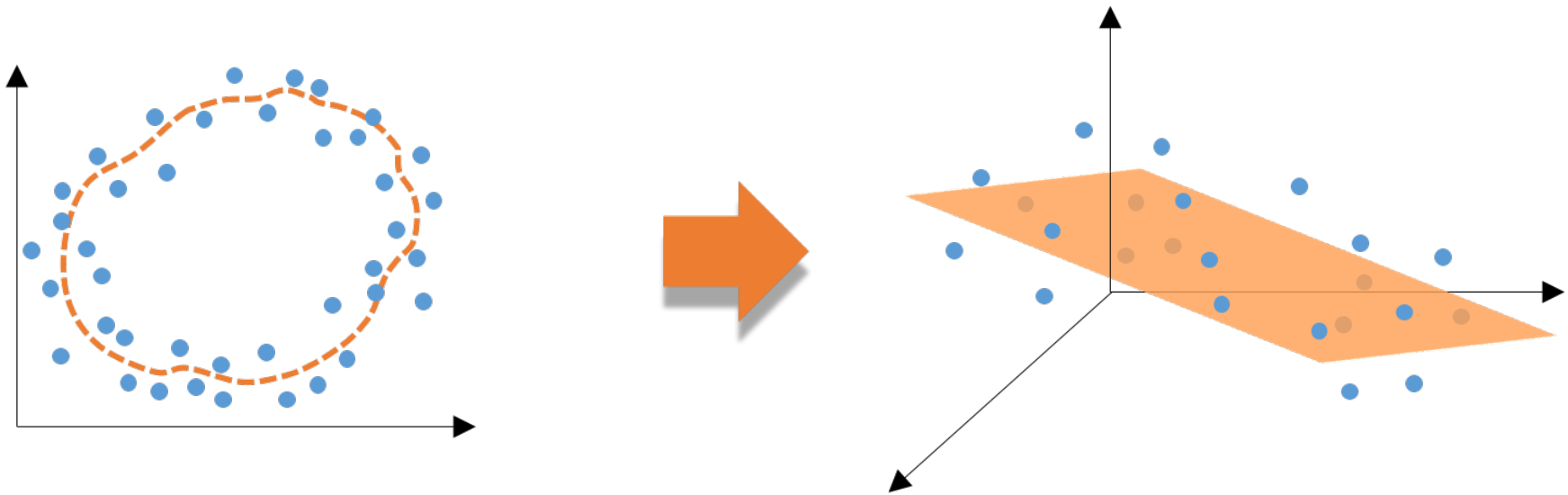
Can relax into convex problems by

- finding convex surrogates (e.g. compressed sensing, matrix completion)

Convex relaxation

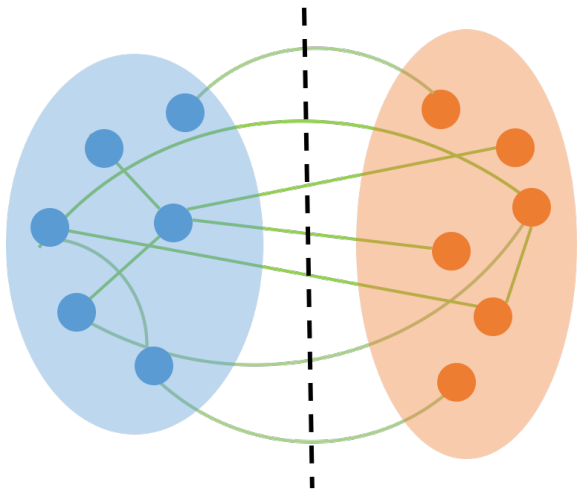
Can relax into convex problems by

- finding convex surrogates (e.g. compressed sensing, matrix completion)
- lifting the problem into higher dimensions (e.g. Max-Cut, phase retrieval)



Example of lifting: Max-Cut

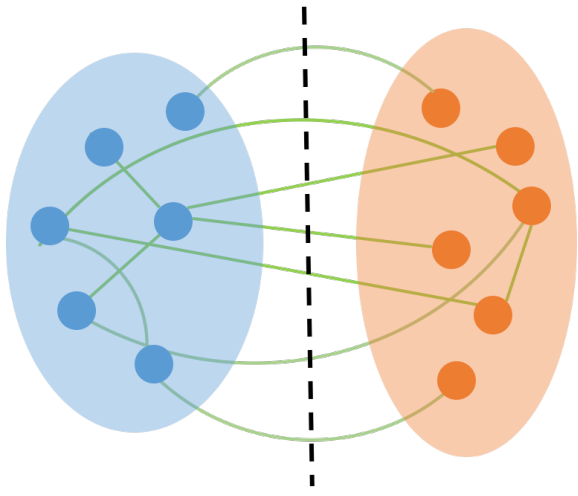
Goemans, Williamson '95



$$\begin{aligned} \text{maximize}_{\boldsymbol{x}} \quad & \boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{x} \\ \text{subj. to} \quad & x_i^2 = 1, \quad i = 1, \cdots, n \end{aligned}$$

Example of lifting: Max-Cut

Goemans, Williamson '95



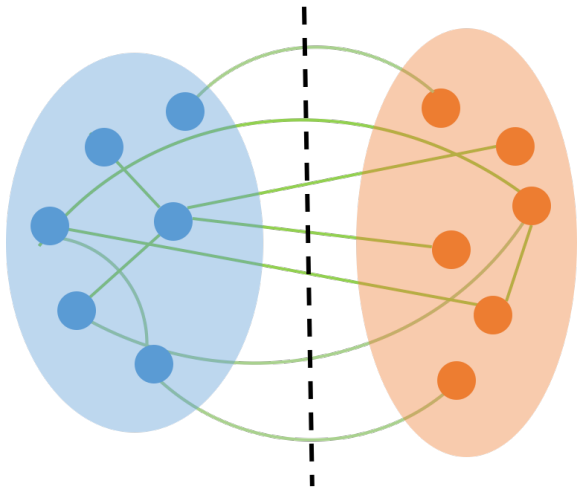
$$\begin{aligned} \text{maximize}_{\boldsymbol{x}} \quad & \boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{x} \\ \text{subj. to} \quad & x_i^2 = 1, \quad i = 1, \cdots, n \end{aligned}$$

$\Downarrow$ let $\boldsymbol{X}$ be $\boldsymbol{x}\boldsymbol{x}^\top$

$$\begin{aligned} \text{maximize}_{\boldsymbol{X}} \quad & \langle \boldsymbol{X}, \boldsymbol{W} \rangle \\ \text{subj. to} \quad & \boldsymbol{X}_{i,i} = 1, \quad i = 1, \cdots, n \\ & \boldsymbol{X} \succeq \boldsymbol{0} \\ & \text{rank}(\boldsymbol{X}) = 1 \end{aligned}$$

Example of lifting: Max-Cut

Goemans, Williamson '95



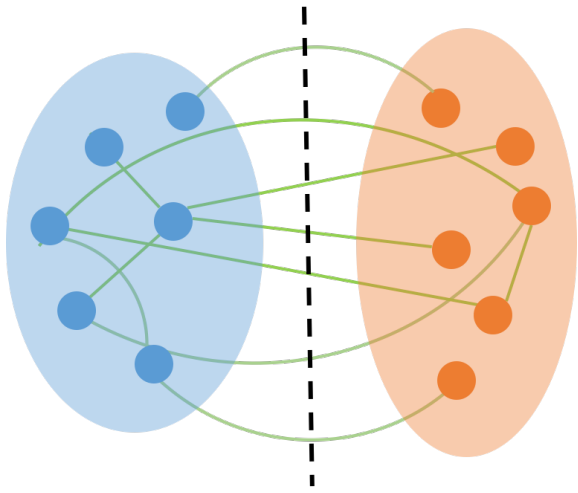
$$\begin{aligned} \text{maximize}_{\boldsymbol{x}} \quad & \boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{x} \\ \text{subj. to} \quad & x_i^2 = 1, \quad i = 1, \cdots, n \end{aligned}$$

$\Downarrow$ let $\boldsymbol{X}$ be $\boldsymbol{x}\boldsymbol{x}^\top$

$$\begin{aligned} \text{maximize}_{\boldsymbol{X}} \quad & \langle \boldsymbol{X}, \boldsymbol{W} \rangle \\ \text{subj. to} \quad & \boldsymbol{X}_{i,i} = 1, \quad i = 1, \cdots, n \\ & \boldsymbol{X} \succeq \boldsymbol{0} \\ & \cancel{\text{rank}(\boldsymbol{X}) = 1} \end{aligned}$$

Example of lifting: Max-Cut

Goemans, Williamson '95



$$\begin{aligned} \text{maximize}_{\boldsymbol{x}} \quad & \boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{x} \\ \text{subj. to} \quad & x_i^2 = 1, \quad i = 1, \cdots, n \end{aligned}$$

$\Downarrow$ let $\boldsymbol{X}$ be $\boldsymbol{x}\boldsymbol{x}^\top$

$$\begin{aligned} \text{maximize}_{\boldsymbol{X}} \quad & \langle \boldsymbol{X}, \boldsymbol{W} \rangle \\ \text{subj. to} \quad & \boldsymbol{X}_{i,i} = 1, \quad i = 1, \cdots, n \\ & \boldsymbol{X} \succeq \boldsymbol{0} \\ & \cancel{\text{rank}(\boldsymbol{X}) = 1} \end{aligned}$$

Problem: explosion in dimensions ($\mathbb{R}^n \to \mathbb{R}^{n \times n}$)

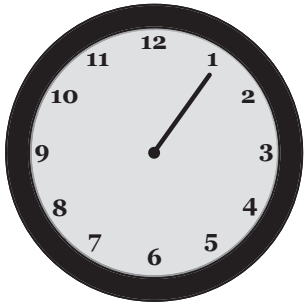
How about solving nonconvex problems directly without lifting?

How about solving nonconvex problems directly without lifting?

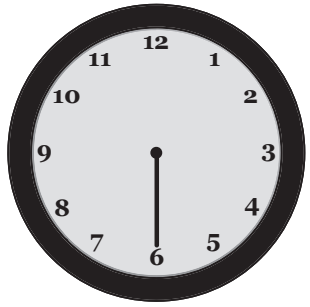
This talk: an efficient paradigm for discrete problems

Joint alignment from pairwise differences

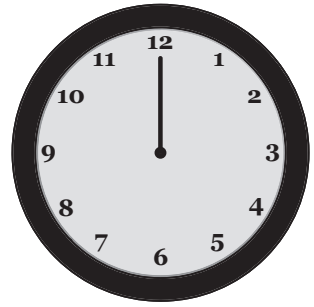
- n unknown variables: $x_1, \cdots, x_n$
- m possible states: $x_i \in \{1, 2, \cdots, m\}$



$x_1 = 1$



$x_2 = 6$

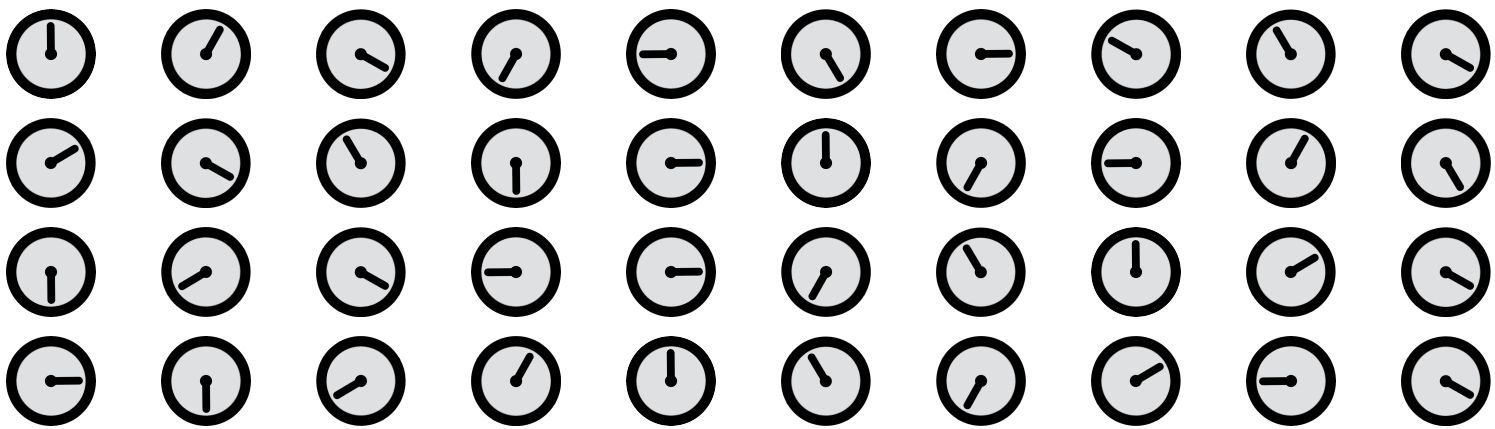


$x_3 = 12$

$\cdots$

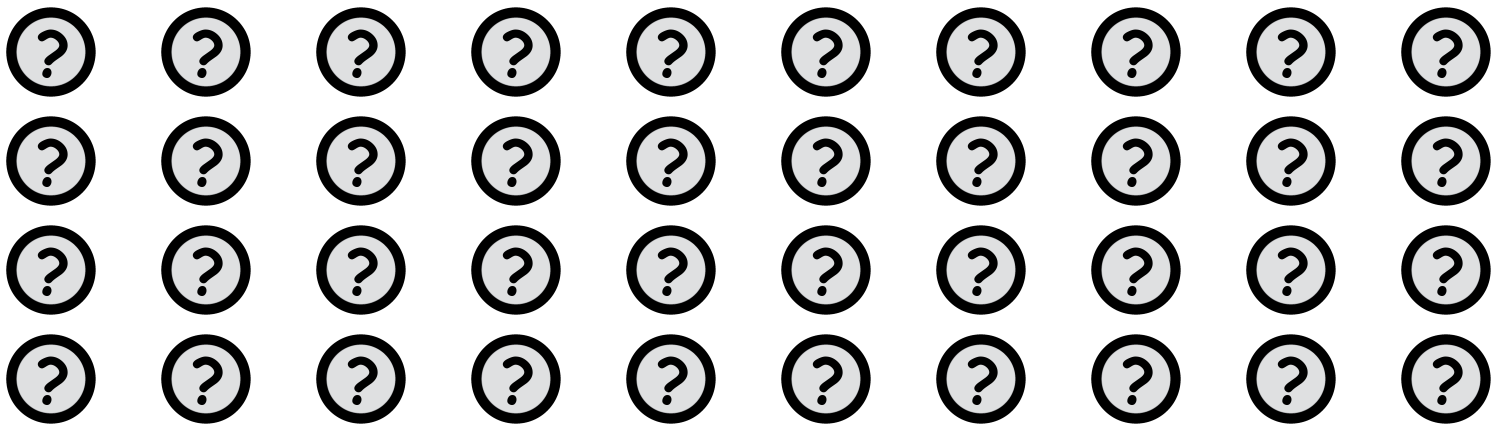
Joint alignment from pairwise differences

- n unknown variables: $x_1, \cdots, x_n$
- m possible states: $x_i \in \{1, 2, \cdots, m\}$



Joint alignment from pairwise differences

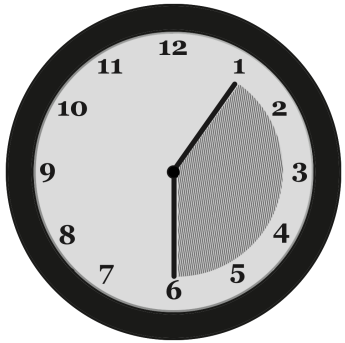
- n unknown variables: $x_1, \cdots, x_n$
- m possible states: $x_i \in \{1, 2, \cdots, m\}$



Joint alignment from pairwise differences

- **Measurements:** pairwise differences

$$y_{i,j} \overset{\text{ind.}}{=} x_i - x_j + \underbrace{\eta_{i,j}}_{\text{noise}} \bmod m, \qquad i \neq j$$



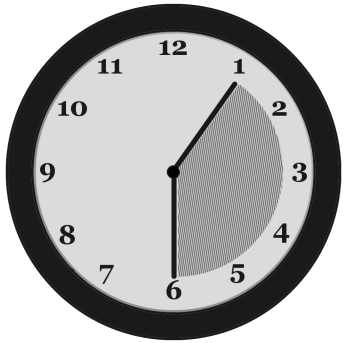
$x_i - x_j \bmod m$

Bandeira, Charikar, Singer, Zhu '13; Chen, Guibas, Huang '14

Joint alignment from pairwise differences

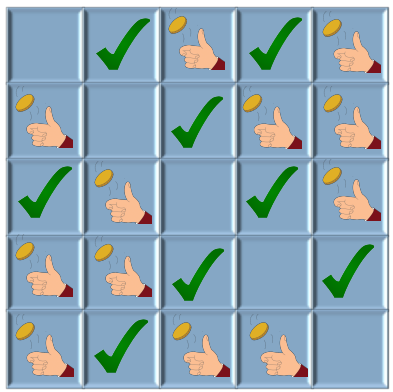
- **Measurements:** pairwise differences

$$y_{i,j} \overset{\text{ind.}}{=} x_i - x_j + \underbrace{\eta_{i,j}}_{\text{noise}} \bmod m, \qquad i \neq j$$



$x_i - x_j \bmod m$

— e.g. random corruption model



$$y_{i,j} \overset{\text{ind}}{=} \begin{cases} x_i - x_j \bmod m & \text{with prob. } \pi_0 \\ \text{Uniform}\,(m) & \text{else} \end{cases}$$

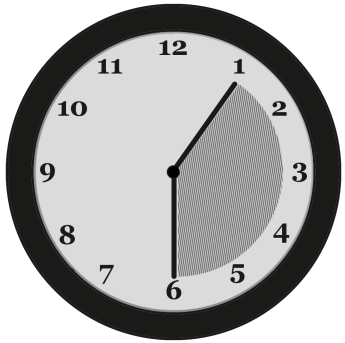
- π_0: non-corruption rate

Bandeira, Charikar, Singer, Zhu '13; Chen, Guibas, Huang '14

Joint alignment from pairwise differences

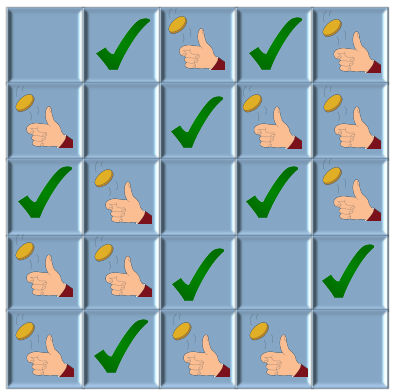
- **Measurements:** pairwise differences

$$y_{i,j} \overset{\text{ind.}}{=} x_i - x_j + \underbrace{\eta_{i,j}}_{\text{noise}} \bmod m, \qquad i \neq j$$



$x_i - x_j \bmod m$

— e.g. random corruption model



$$y_{i,j} \overset{\text{ind}}{=} \begin{cases} x_i - x_j \bmod m & \text{with prob. } \pi_0 \\ \text{Uniform}\,(m) & \text{else} \end{cases}$$

- π_0: non-corruption rate

- **Goal:** recover $\{x_i\}$ (up to global offset)

Bandeira, Charikar, Singer, Zhu '13; Chen, Guibas, Huang '14

Motivation: community recovery

Community structures are common in many social networks

Fig. credit: The Future Buzz

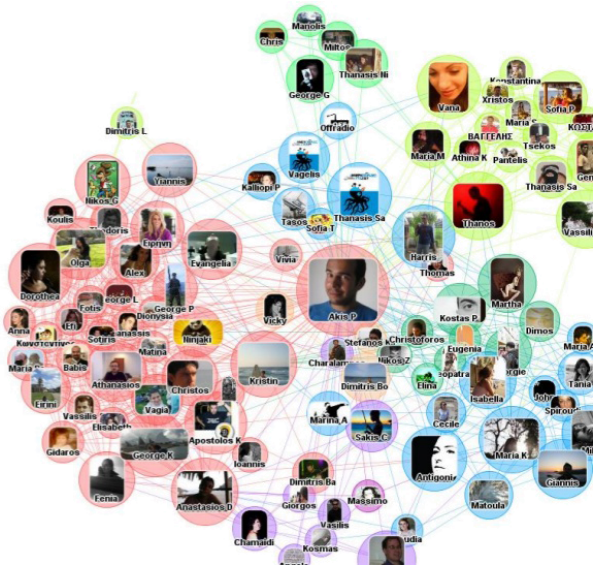


Fig. credit: S. Papadopoulos

Motivation: community recovery

Community structures are common in many social networks

Fig. credit: The Future Buzz

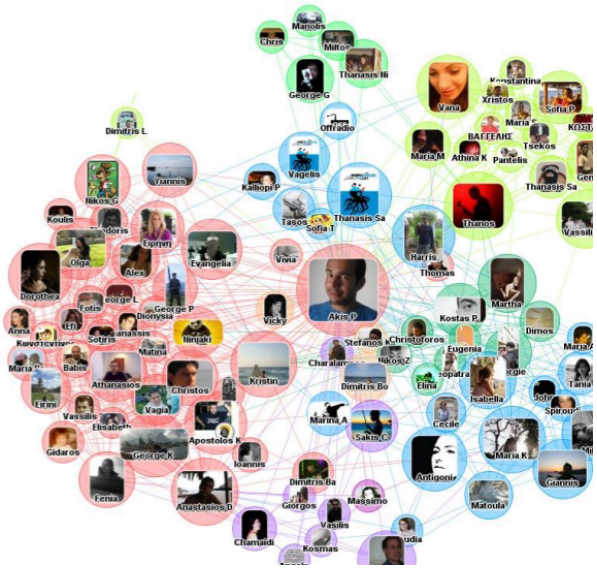


Fig. credit: S. Papadopoulos

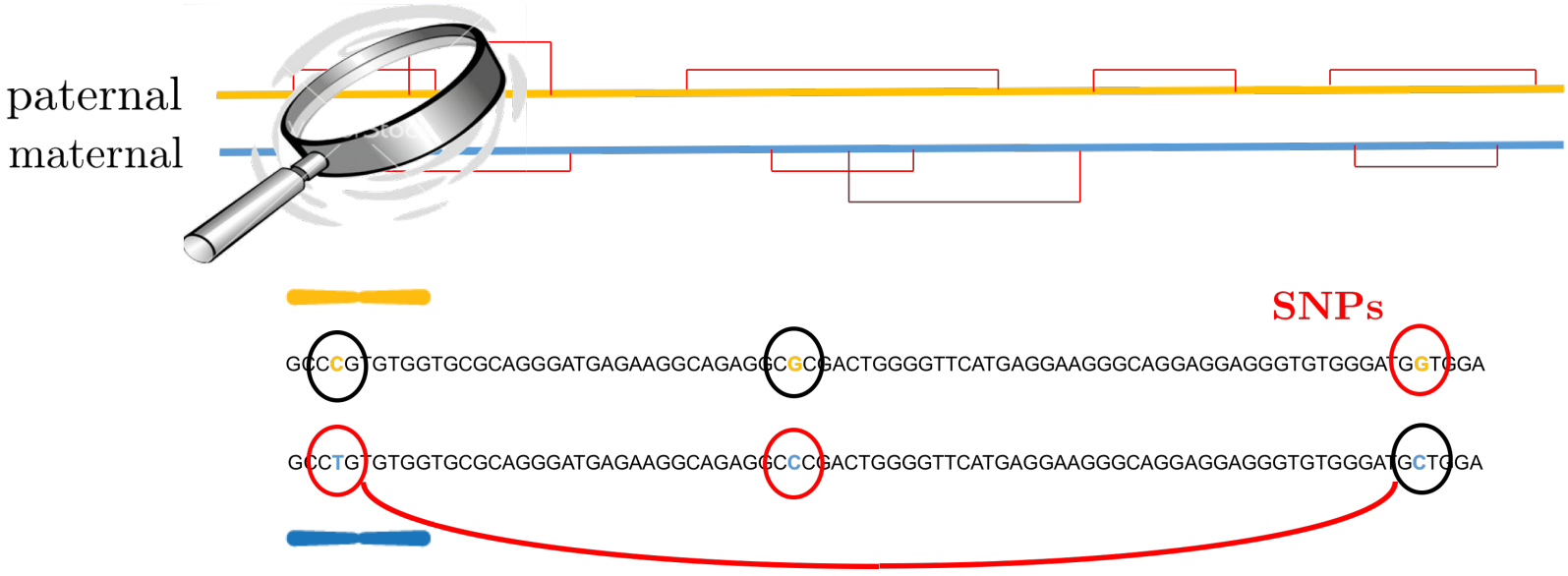
Community recovery: partition users into several clusters based on their friendships / similarities

Motivation: genome phasing

Motivation: genome phasing

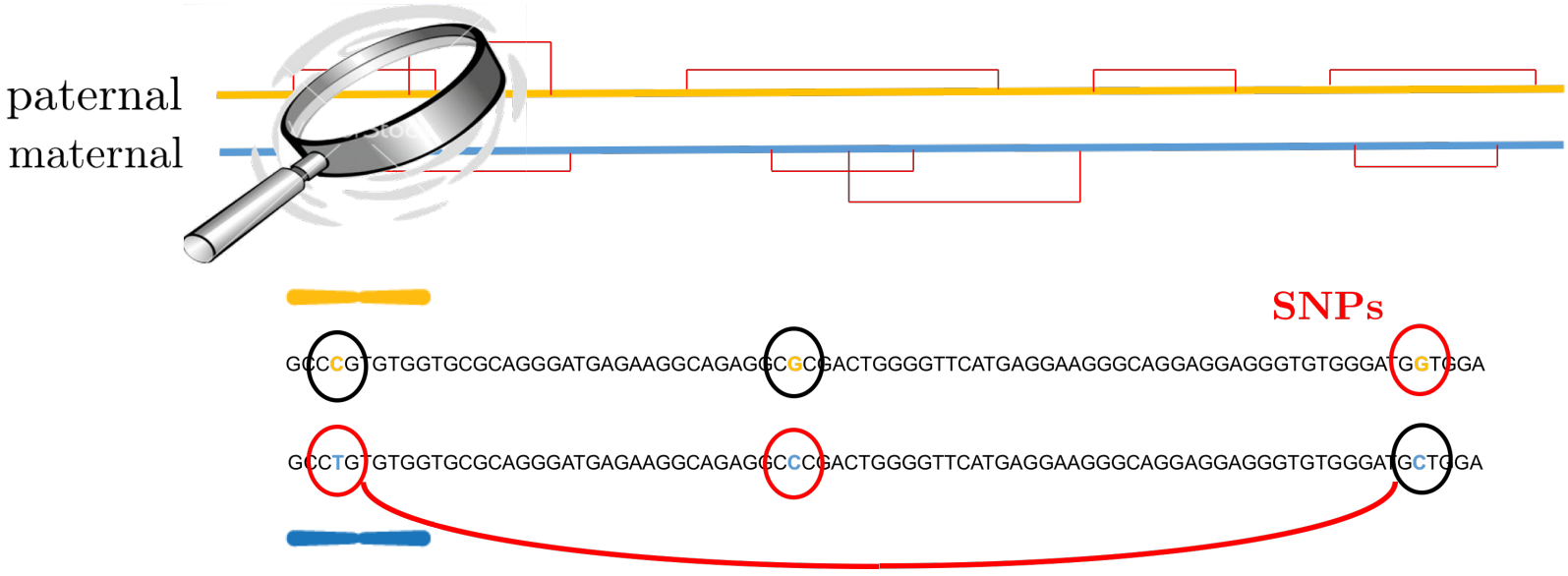
- **phase info x_i for each SNP:**
  (1) $x_i = 0$: maternally inherited (2) $x_i = 1$: paternally inherited

Motivation: genome phasing



- **phase info x_i for each SNP:**
  (1) $x_i = 0$: maternally inherited (2) $x_i = 1$: paternally inherited
- **paired reads:** whether 2 SNPs are on same phase $(x_i \oplus x_j)$

Motivation: genome phasing



- **phase info x_i for each SNP:**
  (1) $x_i = 0$: maternally inherited (2) $x_i = 1$: paternally inherited
- **paired reads:** whether 2 SNPs are on same phase $(x_i \oplus x_j)$

Phasing: retrieve phase info (haplotype) of all SNPs from paired reads

Motivation: multi-image alignment

Jointly align a collection of images/shapes of the same physical object

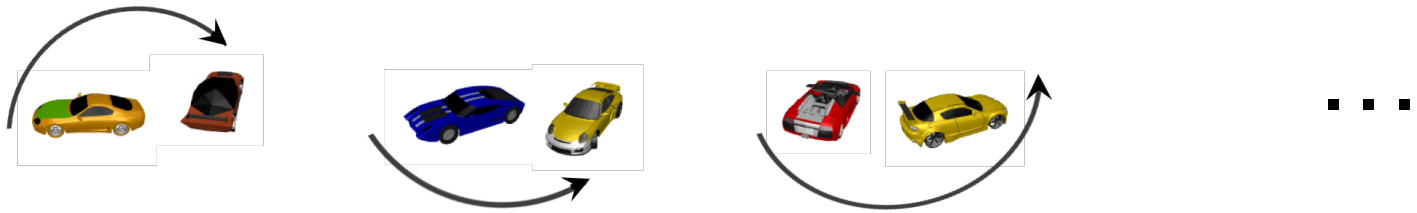
Motivation: multi-image alignment

Jointly align a collection of images/shapes of the same physical object

- x_i: angle of rotation associated with each shape

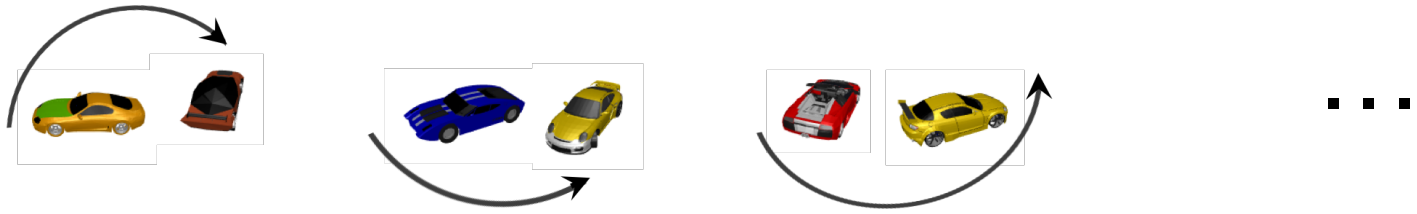
Motivation: multi-image alignment

Step 1: compute pairwise estimates of relative angles of rotations $\{x_i - x_j\}$



Motivation: multi-image alignment

Step 1: compute pairwise estimates of relative angles of rotations $\{x_i - x_j\}$



Step 2: aggregate these pairwise information for joint alignment

Many other related applications ...

- Structure from motion in computer vision
- Cryo-EM in structural biology
- Water-fat separation in MRI
- ...

Maximum likelihood estimates (MLE)

$$\begin{aligned} \text{maximize}_{\{x_i\}} \quad & \sum_{i,j} \ell\left(x_i, x_j; y_{i,j}\right) \\ \text{subj. to} \quad & x_i \in \{1, \cdots, m\}, \quad 1 \leq i \leq n \end{aligned}$$

- Log-likelihood function ℓ may be complicated

Maximum likelihood estimates (MLE)

$$\begin{aligned} \text{maximize}_{\{x_i\}} \quad & \sum_{i,j} \ell\left(x_i, x_j; y_{i,j}\right) \\ \text{subj. to} \quad & x_i \in \{1, \cdots, m\}, \quad 1 \leq i \leq n \end{aligned}$$

- Log-likelihood function ℓ may be complicated
- Discrete input space

Maximum likelihood estimates (MLE)

$$\begin{aligned} \text{maximize}_{\{x_i\}} \quad & \sum_{i,j} \ell\left(x_i, x_j; y_{i,j}\right) \\ \text{subj. to} \quad & x_i \in \{1, \cdots, m\}, \quad 1 \leq i \leq n \end{aligned}$$

- Log-likelihood function ℓ may be complicated
- Discrete input space
- Looks daunting

Alternative representation of discrete variables

Discrete variables $\rightarrow$ orthogonal vectors in higher-dimensional space

$$x_i = 1 \qquad \Longleftrightarrow \qquad \boldsymbol{x}_i = \boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$x_i = 2 \qquad \Longleftrightarrow \qquad \boldsymbol{x}_i = \boldsymbol{e}_1 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

$$\vdots$$

$$x_i = j \qquad \Longleftrightarrow \qquad \boldsymbol{x}_i = \boldsymbol{e}_j$$

Matrix representation

Pairwise sample $y_{i,j}$ $\rightarrow$ encode $\ell(x_i, x_j)$ by $\boldsymbol{L}_{i,j} \in \mathbb{R}^{m \times m}$

$$[\boldsymbol{L}_{i,j}]_{\alpha,\beta} = \ell(x_i = \alpha, x_j = \beta)$$

Matrix representation

Pairwise sample $y_{i,j}$ $\rightarrow$ encode $\ell(x_i, x_j)$ by $\boldsymbol{L}_{i,j} \in \mathbb{R}^{m \times m}$

$$[\boldsymbol{L}_{i,j}]_{\alpha,\beta} = \ell(x_i = \alpha, x_j = \beta)$$

- e.g. random corruption model

$$y_{i,j} = \begin{cases} x_i - x_j, & \text{w.p. } \pi_0 \\ \text{Unif}(m), & \text{else} \end{cases} \quad \Rightarrow \quad \ell(x_i, x_j) = \begin{cases} \log(\pi_0 + \frac{1-\pi_0}{m}), & \text{if } x_i - x_j = y_{i,j} \\ \log(\frac{1-\pi_0}{m}), & \text{else} \end{cases}$$

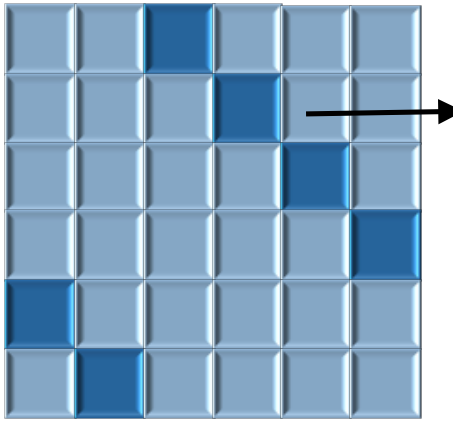
Matrix representation

Pairwise sample $y_{i,j}$ $\rightarrow$ encode $\ell(x_i, x_j)$ by $\boldsymbol{L}_{i,j} \in \mathbb{R}^{m \times m}$

$$[\boldsymbol{L}_{i,j}]_{\alpha,\beta} = \ell(x_i = \alpha, x_j = \beta)$$

- e.g. random corruption model

$$y_{i,j} = \begin{cases} x_i - x_j, & \text{w.p. } \pi_0 \\ \text{Unif}(m), & \text{else} \end{cases} \quad \Rightarrow \quad \ell(x_i, x_j) = \begin{cases} \log(\pi_0 + \frac{1-\pi_0}{m}), & \text{if } x_i - x_j = y_{i,j} \\ \log(\frac{1-\pi_0}{m}), & \text{else} \end{cases}$$



$$\ell(x_i = 2, x_j = 5; y_{i,j} = 2)$$

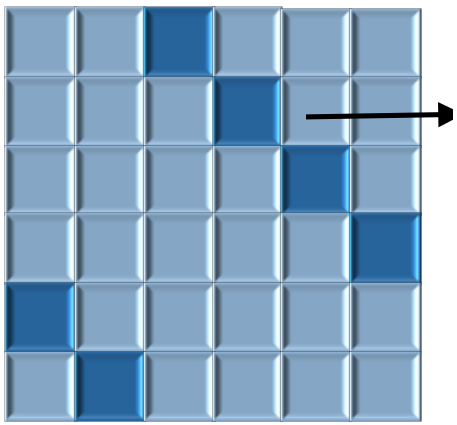
Matrix representation

Pairwise sample $y_{i,j}$ $\rightarrow$ encode $\ell(x_i, x_j)$ by $\boldsymbol{L}_{i,j} \in \mathbb{R}^{m \times m}$

$$[\boldsymbol{L}_{i,j}]_{\alpha,\beta} = \ell(x_i = \alpha, x_j = \beta)$$

- e.g. random corruption model

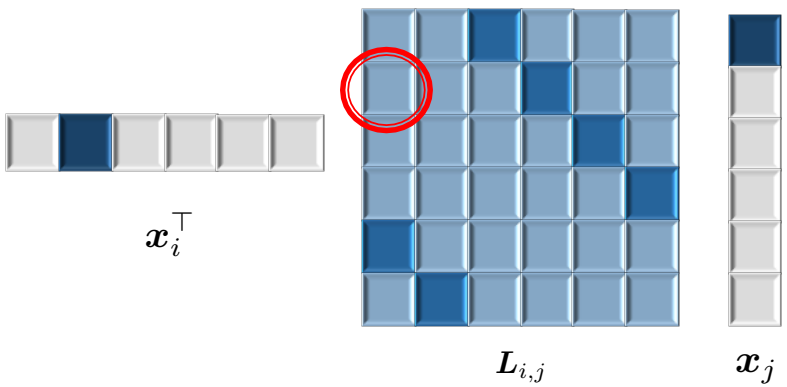
$$y_{i,j} = \begin{cases} x_i - x_j, & \text{w.p. } \pi_0 \\ \text{Unif}(m), & \text{else} \end{cases} \quad \Rightarrow \quad \ell(x_i, x_j) = \begin{cases} \log(\pi_0 + \frac{1-\pi_0}{m}), & \text{if } x_i - x_j = y_{i,j} \\ \log(\frac{1-\pi_0}{m}), & \text{else} \end{cases}$$



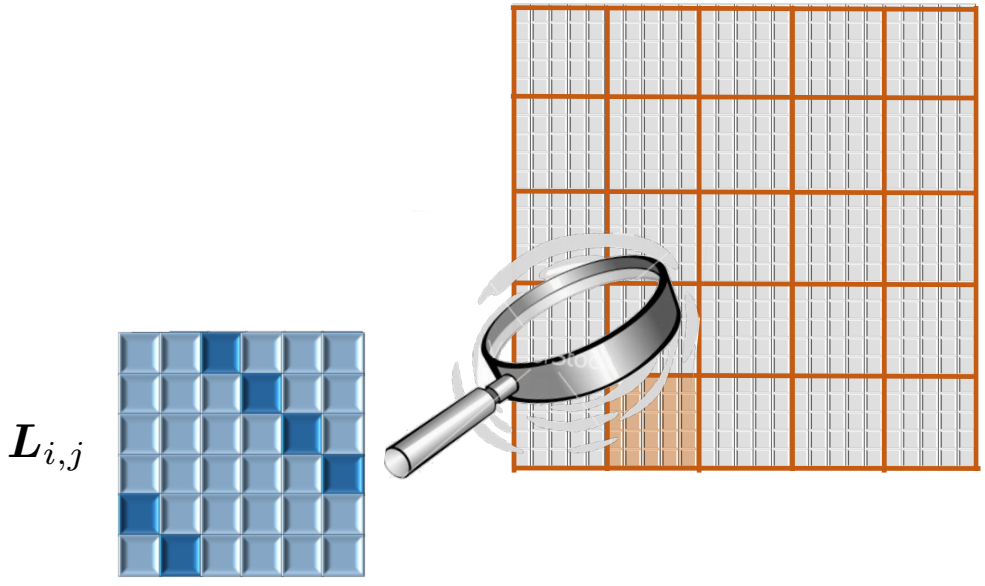
$$\ell(x_i = 2, x_j = 5; y_{i,j} = 2)$$

This enables quadratic representation

$$\ell(x_i, x_j) \;=\; \boldsymbol{x}_i^\top \boldsymbol{L}_{i,j} \boldsymbol{x}_j$$



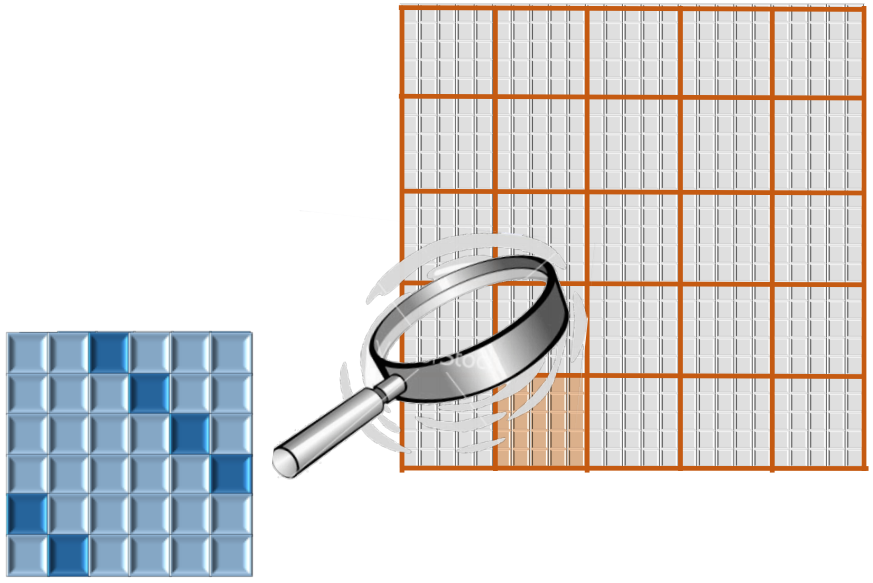
MLE is equivalent to a binary quadratic program



$\boldsymbol{L}_{i,j}$

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_{1,1} & \cdots & \boldsymbol{L}_{1,n} \\ \vdots & \ddots & \vdots \\ \boldsymbol{L}_{n,1} & \cdots & \boldsymbol{L}_{n,n} \end{bmatrix}$$

MLE is equivalent to a binary quadratic program



$\boldsymbol{L}_{i,j}$

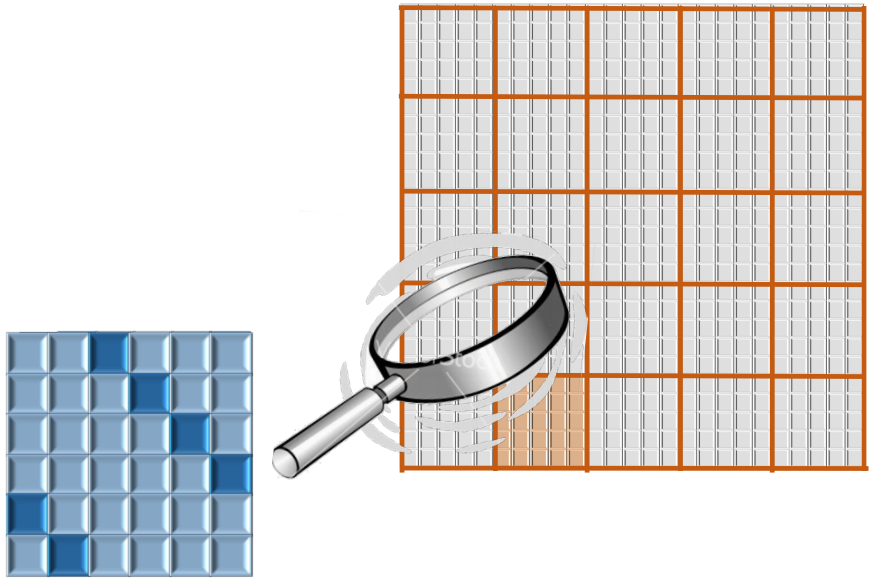
$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_{1,1} & \cdots & \boldsymbol{L}_{1,n} \\ \vdots & \ddots & \vdots \\ \boldsymbol{L}_{n,1} & \cdots & \boldsymbol{L}_{n,n} \end{bmatrix}$$

$$\begin{aligned} \text{maximize} \quad & \sum_{i,j} \ell(x_i - x_j; y_{ij}) \\ \text{subj. to} \quad & x_i \in \{1, \cdots, m\} \end{aligned}$$

$$\Longleftrightarrow$$

$$\begin{aligned} \text{maximize}_{\boldsymbol{x}} \quad & \boldsymbol{x}^\top \boldsymbol{L} \boldsymbol{x} \\ \text{subj. to} \quad & \boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix} \\ & \boldsymbol{x}_i \in \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_m\} \end{aligned}$$

MLE is equivalent to a binary quadratic program



$\boldsymbol{L}_{i,j}$

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_{1,1} & \cdots & \boldsymbol{L}_{1,n} \\ \vdots & \ddots & \vdots \\ \boldsymbol{L}_{n,1} & \cdots & \boldsymbol{L}_{n,n} \end{bmatrix}$$

maximize $\quad \sum_{i,j} \ell(x_i - x_j; y_{ij})$

subj. to $\quad x_i \in \{1, \cdots, m\}$

$\Longleftrightarrow$

maximize$_{\boldsymbol{x}}$ $\quad \boldsymbol{x}^\top \boldsymbol{L} \boldsymbol{x}$

subj. to $\quad \boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix}$

$\boldsymbol{x}_i \in \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_m\}$

This is essentially nonconvex constrained PCA

How to solve nonconvex constrained PCA?

PCA

$$\begin{array}{ll} \text{maximize}_{\boldsymbol{x}} & \boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} \\ \text{subj. to} & \|\boldsymbol{x}\| = 1 \end{array}$$

Power method:

for $t = 1,\ 2,\ \cdots$

$$\begin{aligned} \boldsymbol{z}^{(t)} &= \boldsymbol{L}\boldsymbol{z}^{(t-1)} \\ \boldsymbol{z}^{(t)} &\leftarrow \text{normalize}\left(\boldsymbol{z}^{(t)}\right) \end{aligned}$$

How to solve nonconvex constrained PCA?

PCA

$$\begin{array}{ll} \text{maximize}_{\boldsymbol{x}} & \boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} \\ \text{subj. to} & \|\boldsymbol{x}\| = 1 \end{array}$$

Constrained PCA

$$\begin{array}{ll} \text{maximize}_{\boldsymbol{x}} & \boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} \\ \text{subj. to} & \boldsymbol{x}_i \in \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_m\} \end{array}$$

Power method:

for $t = 1,\ 2,\ \cdots$

$$\begin{aligned} \boldsymbol{z}^{(t)} &= \boldsymbol{L}\boldsymbol{z}^{(t-1)} \\ \boldsymbol{z}^{(t)} &\leftarrow \text{normalize}\left(\boldsymbol{z}^{(t)}\right) \end{aligned}$$

Projected power method:

for $t = 1,\ 2,\ \cdots$

$$\begin{aligned} \boldsymbol{z}^{(t)} &= \boldsymbol{L}\boldsymbol{z}^{(t-1)} \\ \boldsymbol{z}^{(t)} &\leftarrow \text{Project}_{\Delta^n}\left(\mu \boldsymbol{z}^{(t)}\right) \end{aligned}$$

- μ: scaling factor

Projection onto standard simplex

$$\text{maximize}_{\boldsymbol{x}=\{\boldsymbol{x}_i\}} \quad \boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} \qquad \text{s.t. } \boldsymbol{x}_i \in \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_m\}$$

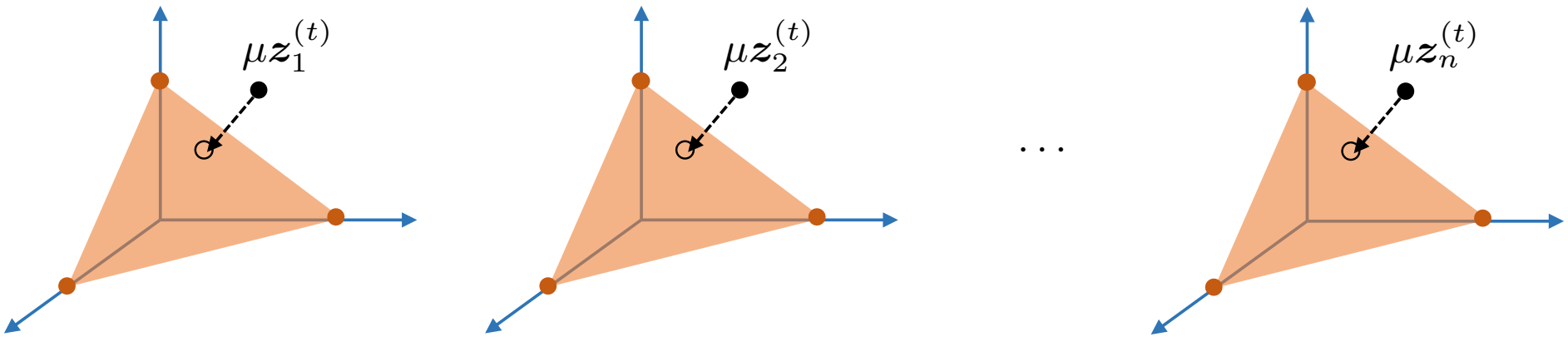
$$\begin{aligned} \boldsymbol{z}^{(t)} &= \boldsymbol{L}\boldsymbol{z}^{(t-1)} \\ \boldsymbol{z}^{(t)} &\leftarrow \text{Project}_{\triangle^n}\left(\mu \boldsymbol{z}^{(t)}\right) \end{aligned}$$

Projection onto standard simplex

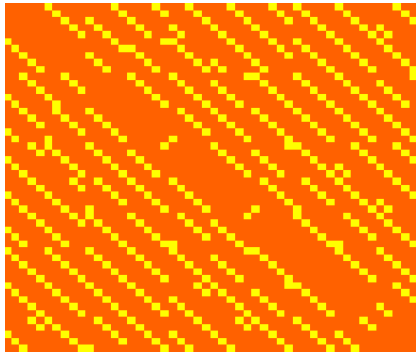
$$\text{maximize}_{\boldsymbol{x}=\{\boldsymbol{x}_i\}} \quad \boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} \qquad \text{s.t. } \boldsymbol{x}_i \in \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_m\}$$

$$\begin{aligned} \boldsymbol{z}^{(t)} &= \boldsymbol{L}\boldsymbol{z}^{(t-1)} \\ \boldsymbol{z}^{(t)} &\leftarrow \text{Project}_{\Delta^n}\left(\mu \boldsymbol{z}^{(t)}\right) \end{aligned}$$

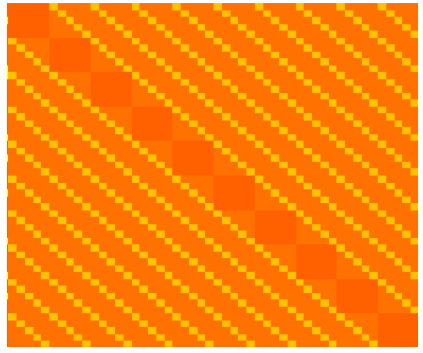
Δ^n is convex hull of feasibility set,
i.e. $\left\{ \boldsymbol{z} = [\boldsymbol{z}_i]_{1 \le i \le n} \;\middle|\; \forall i : \; \mathbf{1}^\top \boldsymbol{z}_i = 1; \; \boldsymbol{z}_i \ge \mathbf{0} \right\}$

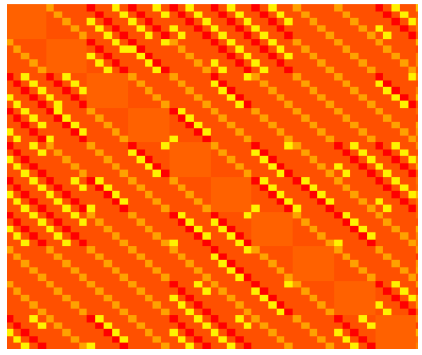


Initialization?

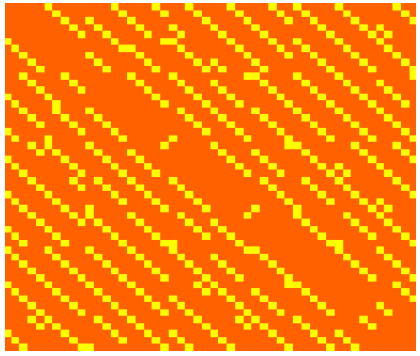


$$\boldsymbol{L} = \underbrace{\mathbb{E}[\boldsymbol{L}]}_{\text{approx. low-rank}} + \boldsymbol{L} - \mathbb{E}[\boldsymbol{L}]$$

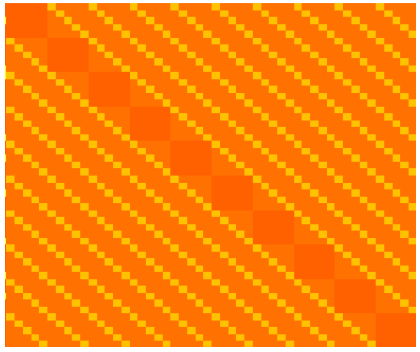




Initialization?



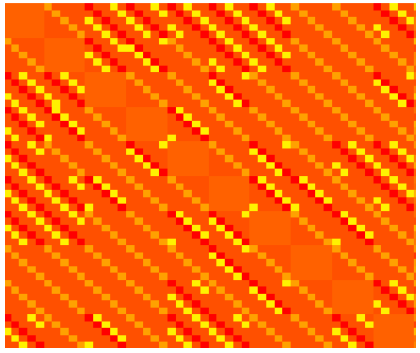
$\boldsymbol{L}$ $=$



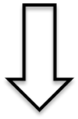
$\underbrace{\mathbb{E}[\boldsymbol{L}]}$

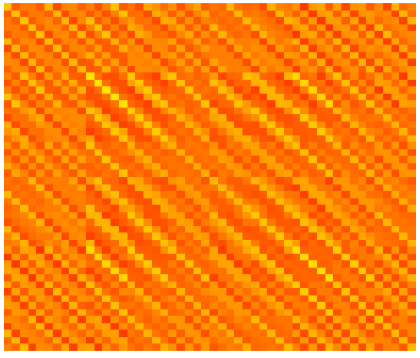
approx. low-rank

$+$



$\boldsymbol{L} - \mathbb{E}[\boldsymbol{L}]$



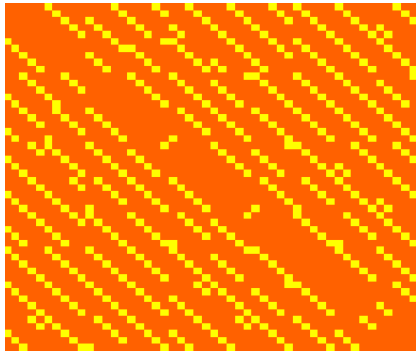


$\hat{\boldsymbol{L}}$

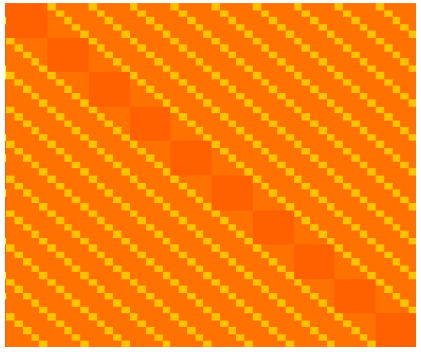
Spectral initialization

1. $\hat{\boldsymbol{L}} \leftarrow$ rank-m approximation of $\boldsymbol{L}$

Initialization?



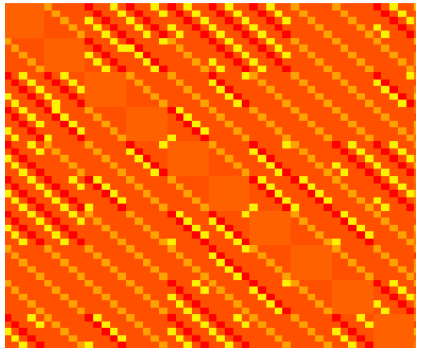
$\boldsymbol{L}$ $=$



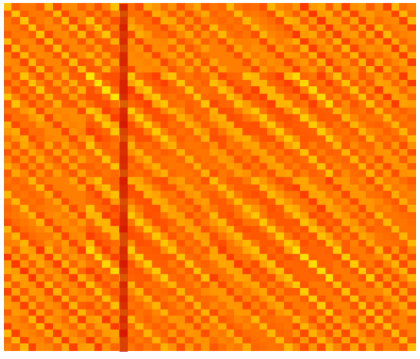
$\underbrace{\mathbb{E}[\boldsymbol{L}]}$

approx. low-rank

$+$



$\boldsymbol{L} - \mathbb{E}[\boldsymbol{L}]$

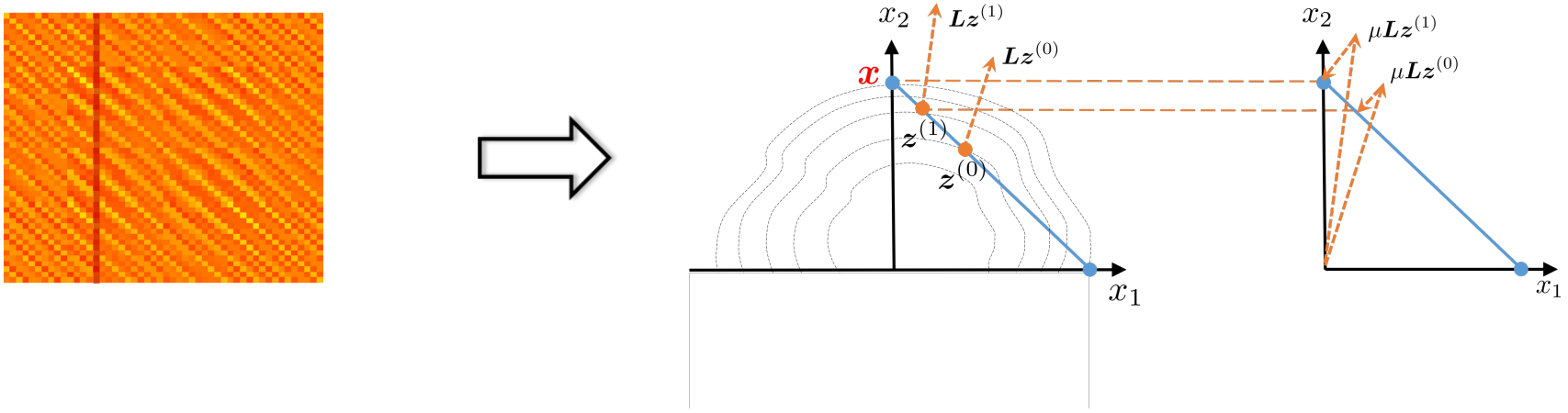


$\hat{\boldsymbol{L}}$

Spectral initialization

1. $\hat{\boldsymbol{L}} \leftarrow$ rank-m approximation of $\boldsymbol{L}$
2. $\boldsymbol{z}^{(0)} \leftarrow \text{Project}_{\Delta^n}(\mu \hat{\boldsymbol{z}})$, where $\hat{\boldsymbol{z}}$ is a random column of $\hat{\boldsymbol{L}}$

Summary of projected power method (PPM)



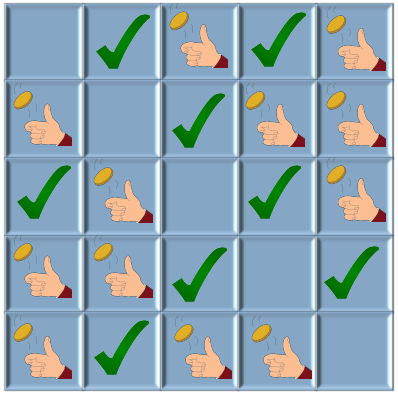
1. Spectral initialization

2. For $t = 1, 2, \cdots$

$$\boldsymbol{z}^{(t)} \quad \leftarrow \quad \text{Project}_{\Delta^n}\left(\mu \boldsymbol{L}\boldsymbol{z}^{(t-1)}\right)$$

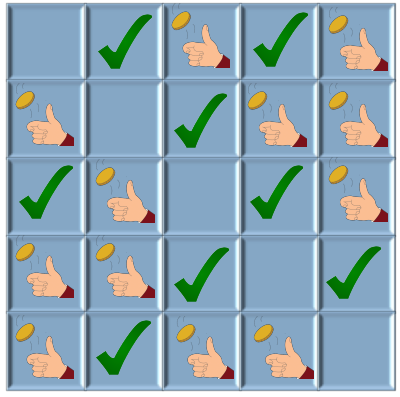
Random corruption model

$$y_{i,j} \overset{\text{ind}}{=} \begin{cases} x_i - x_j \ \mathsf{mod}\ m & \text{with prob. } \pi_0 \\ \mathsf{Uniform}\,(m) & \text{else} \end{cases}$$



Random corruption model

$$y_{i,j} \overset{\text{ind}}{=} \begin{cases} x_i - x_j \bmod m & \text{with prob. } \pi_0 \\ \text{Uniform}\,(m) & \text{else} \end{cases}$$



Theorem (C. & Candès '16) Fix $m > 0$ and set $\mu \gtrsim 1/\sigma_2(\boldsymbol{L})$. With high prob., PPM recovers the truth exactly within $O(\log n)$ iterations if

- signal-to-noise ratio (SNR) not too small: $\pi_0 > 2\sqrt{\frac{\log n}{mn}}$

Implications

Theorem (C. & Candès '16) $\cdots$ PPM succeeds within $O(\log n)$ iterations if

$$\text{non-corruption rate } \pi_0 > 2\sqrt{\frac{\log n}{mn}}$$

- PPM succeeds even when most (i.e. $1 - O(\sqrt{\frac{\log n}{n}})$) entries are corrupted

Implications

Theorem (C. & Candès '16) $\cdots$ PPM succeeds within $O(\log n)$ iterations if

$$\text{non-corruption rate } \pi_0 > 2\sqrt{\frac{\log n}{mn}}$$

- PPM succeeds even when most (i.e. $1 - O(\sqrt{\frac{\log n}{n}})$) entries are corrupted
- Nearly linear time algorithm

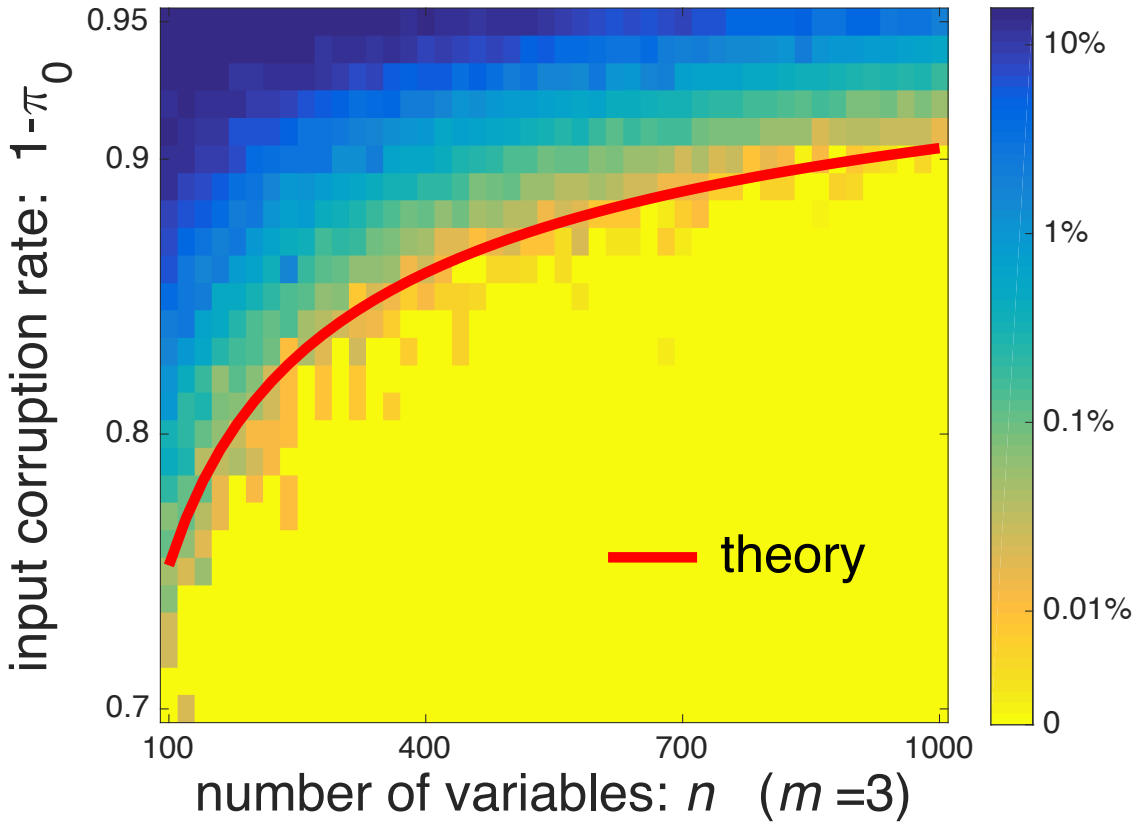
Implications

Theorem (C. & Candès '16) $\cdots$ PPM succeeds within $O(\log n)$ iterations if

$$\text{non-corruption rate} \quad \pi_0 > 2\sqrt{\frac{\log n}{mn}}$$

- PPM succeeds even when most (i.e. $1 - O(\sqrt{\frac{\log n}{n}})$) entries are corrupted
- Nearly linear time algorithm
- Works for any initialization obeying $\|\boldsymbol{z}^{(0)} - \boldsymbol{x}\| < 0.5\|\boldsymbol{x}\|$

Empirical misclassification rate



Misclassification rate when n and π_0 vary $\quad (\mu = 10/\sigma_2(\boldsymbol{L}))$

More general noise models

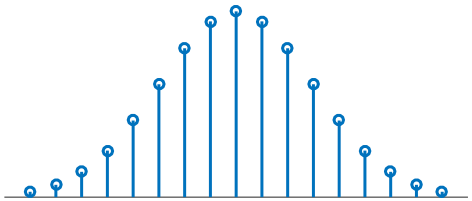
$$y_{i,j} = x_i - x_j + \eta_{i,j} \text{ mod } m, \qquad \text{where } \eta_{i,j} \overset{\text{i.i.d.}}{\sim} P_0$$

More general noise models

$$y_{i,j} = x_i - x_j + \eta_{i,j} \text{ mod } m, \qquad \text{where } \eta_{i,j} \overset{\text{i.i.d.}}{\sim} P_0$$

Distributions of $y_{i,j}$ under different hypotheses

P_0

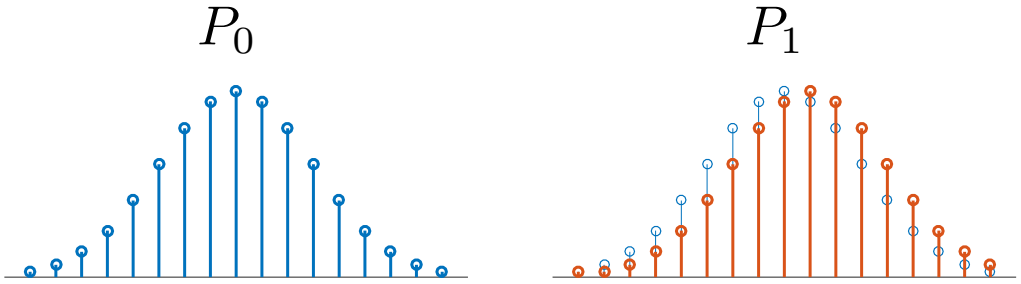


$x_i - x_j = 0$

More general noise models

$$y_{i,j} = x_i - x_j + \eta_{i,j} \text{ mod } m, \qquad \text{where } \eta_{i,j} \overset{\text{i.i.d.}}{\sim} P_0$$

Distributions of $y_{i,j}$ under different hypotheses

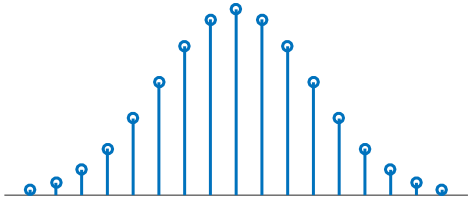


More general noise models

$$y_{i,j} = x_i - x_j + \eta_{i,j} \text{ mod } m, \qquad \text{where } \eta_{i,j} \overset{\text{i.i.d.}}{\sim} P_0$$

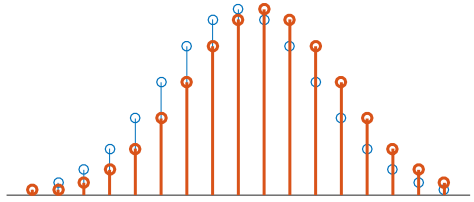
Distributions of $y_{i,j}$ under different hypotheses

P_0



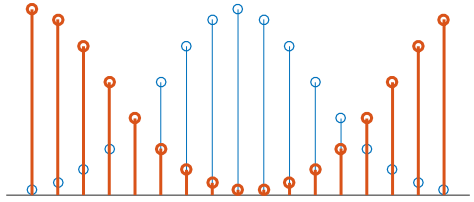
$x_i - x_j = 0$

P_1



$x_i - x_j = 1$

P_9



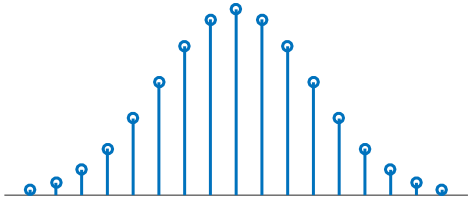
$x_i - x_j = 9$

More general noise models

$$y_{i,j} = x_i - x_j + \eta_{i,j} \text{ mod } m, \qquad \text{where } \eta_{i,j} \overset{\text{i.i.d.}}{\sim} P_0$$

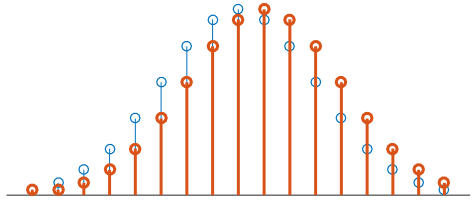
Distributions of $y_{i,j}$ under different hypotheses

P_0



$x_i - x_j = 0$

P_1

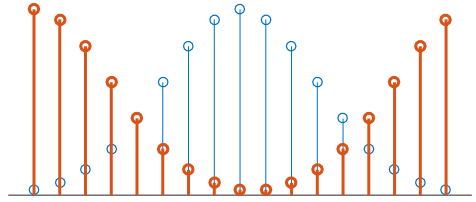


$x_i - x_j = 1$

$\Downarrow$

$\mathsf{KL}(P_0 \parallel P_1)$

P_9



$x_i - x_j = 9$

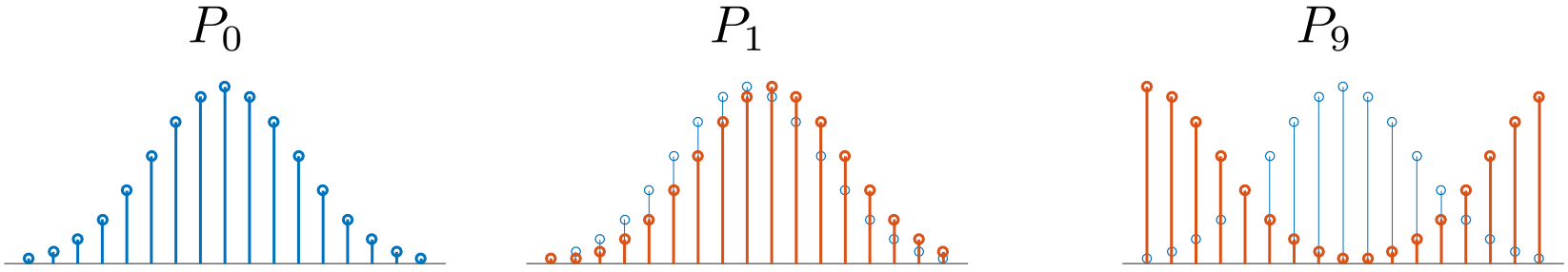
$\Downarrow$

$\mathsf{KL}(P_0 \parallel P_9)$

More general noise models

$$y_{i,j} = x_i - x_j + \eta_{i,j} \text{ mod } m, \qquad \text{where } \eta_{i,j} \overset{\text{i.i.d.}}{\sim} P_0$$

Distributions of $y_{i,j}$ under different hypotheses



$x_i - x_j = 0$

$x_i - x_j = 1$ $\Downarrow$ $\mathsf{KL}(P_0 \parallel P_1)$

$x_i - x_j = 9$ $\Downarrow$ $\mathsf{KL}(P_0 \parallel P_9)$

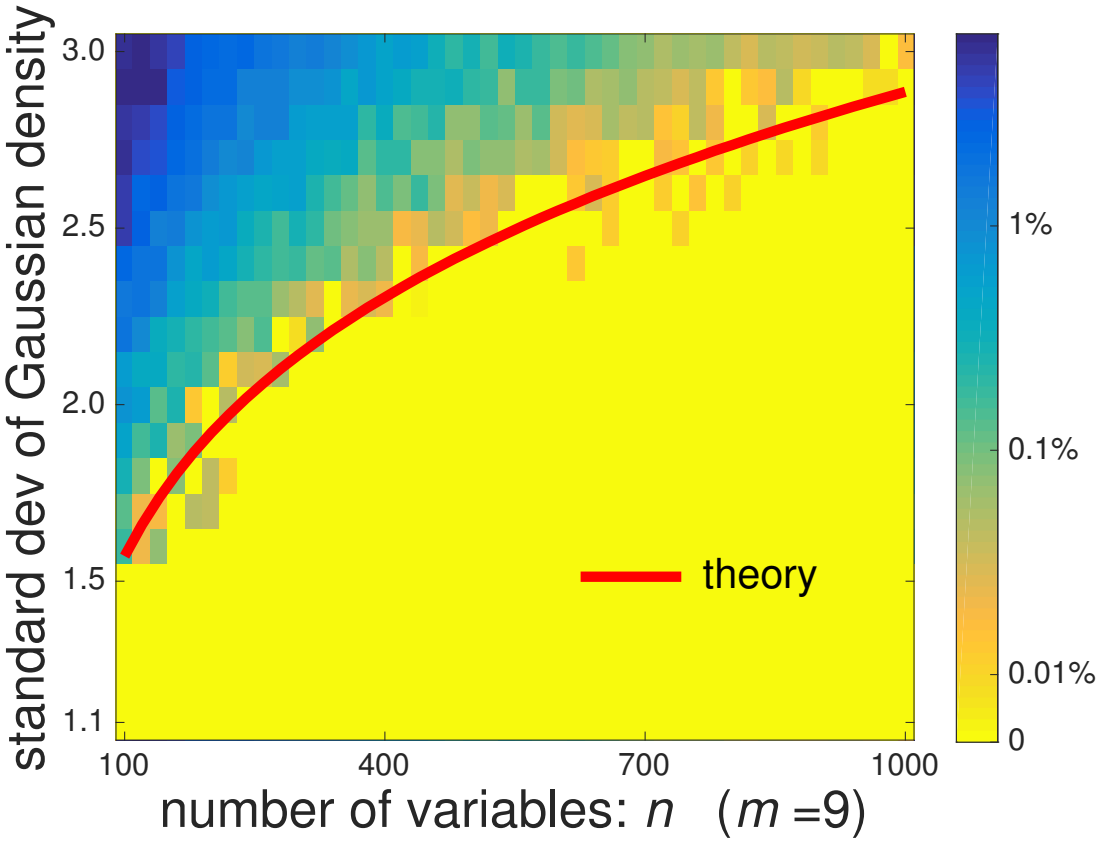
Theorem (C. & Candès '16) Fix $m > 0$ and set $\mu \gtrsim 1/\sigma_2(\boldsymbol{L})$. Under mild conditions, PPM succeeds within $O(\log n)$ iterations with high prob., provided that

$$\mathsf{KL}_{\min} := \min_{1 \le l < m} \mathsf{KL}(P_0 \parallel P_l) \; > \; \frac{4 \log n}{n}$$

Empirical misclassification rate

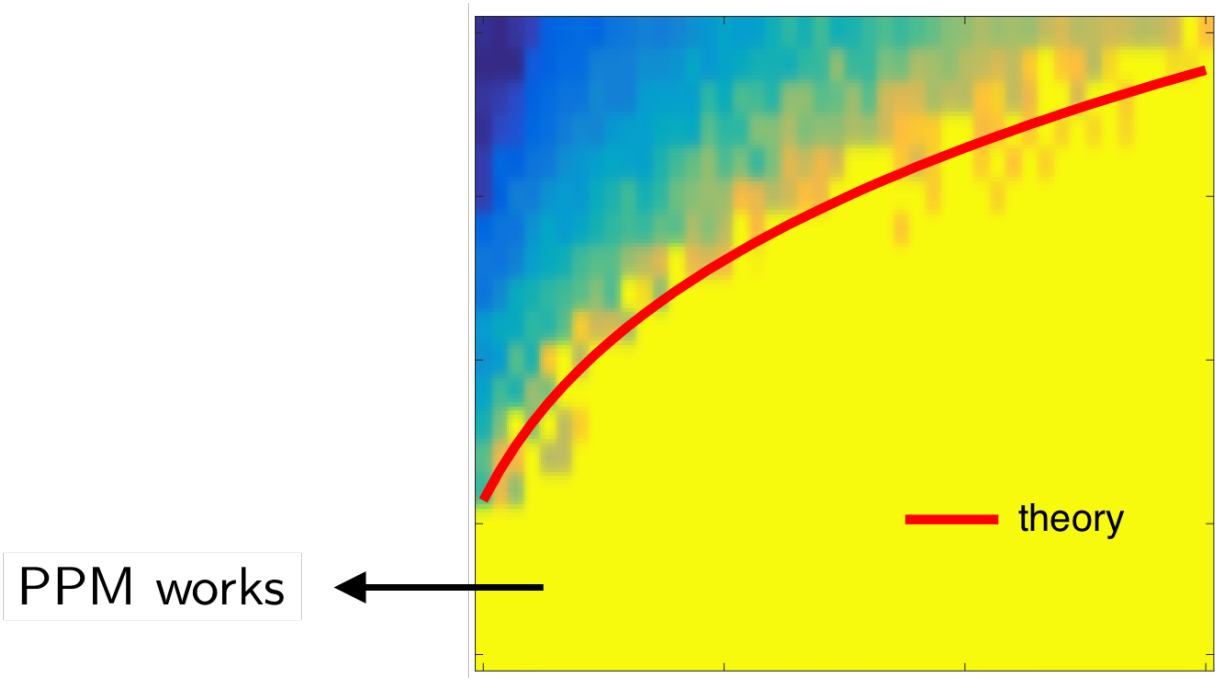
Modified Gaussian noise model:

$$\mathbb{P}\{\eta_{i,j} = z\} \;\propto\; \exp\left(-\frac{z^2}{2\sigma^2}\right), \quad |z| \le \frac{m-1}{2}$$

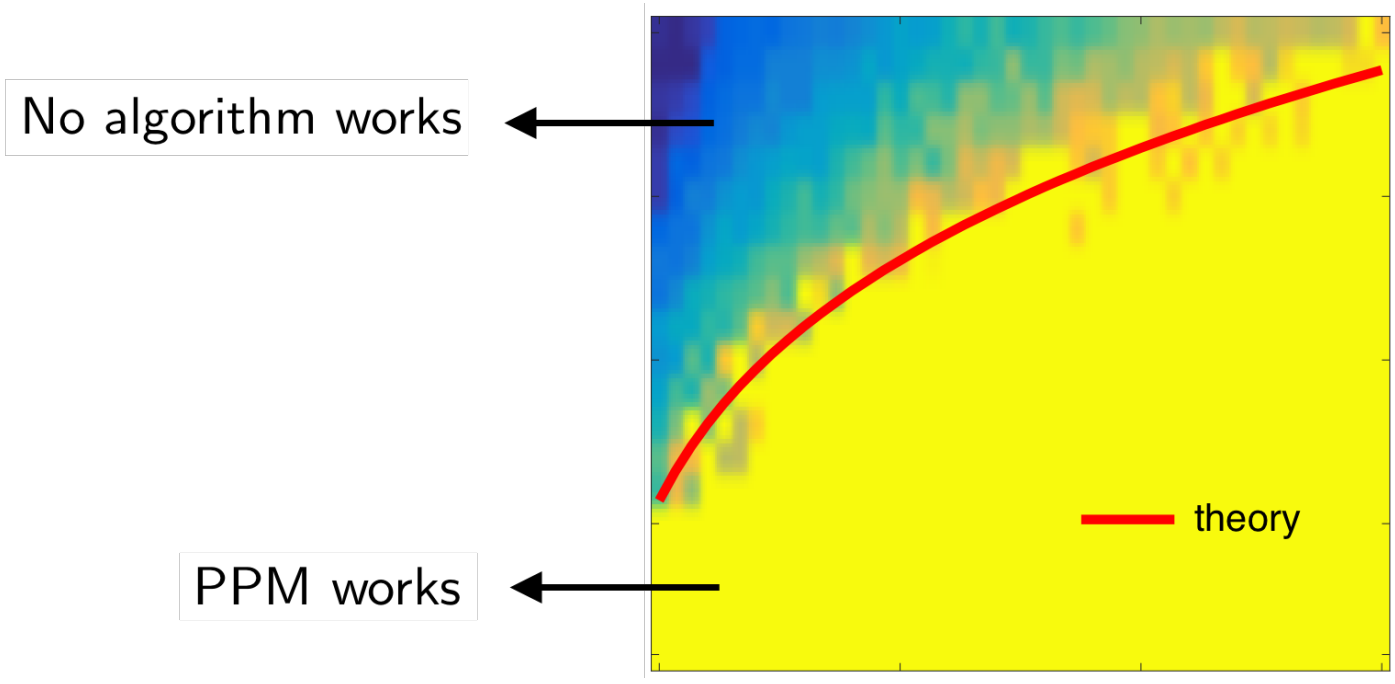


$$\mu = 20/\sigma_m(\boldsymbol{L})$$

PPM is information-theoretically optimal



PPM is information-theoretically optimal



Theorem (Chen-Candès'16) Fix $m > 0$. No method achieves exact recovery if

$$\mathsf{KL}_{\min} < \frac{4 \log n}{n}$$

Large-m case: random corruption model

$$y_{i,j} = \begin{cases} x_i - x_j, & \text{with prob. } \pi_0 \\ \text{Unif}(m), & \text{else} \end{cases}$$

Theorem (C. & Candès '16) Suppose $\log n \lesssim m \lesssim \text{poly}(n)$. PPM succeeds if

$$\pi_0 \gtrsim \frac{1}{\sqrt{n}}$$

Singer'09; Wang & Singer'12; Bandeira et al'14; Boumal'16; Liu et al'16, Perry et al'16 ...

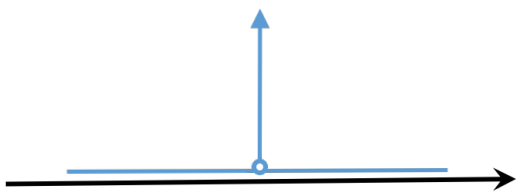
Large-m case: random corruption model

$$y_{i,j} = \begin{cases} x_i - x_j, & \text{with prob. } \pi_0 \\ \text{Unif}(m), & \text{else} \end{cases}$$

Theorem (C. & Candès '16) Suppose $\log n \lesssim m \lesssim \text{poly}(n)$. PPM succeeds if

$$\pi_0 \gtrsim \frac{1}{\sqrt{n}}$$

- Spiky model: when $m \gg n$, model converges to



$$x_i \in [0, 1), \qquad y_{i,j} = \begin{cases} x_i - x_j, & \text{with prob. } \pi_0 \\ \text{Unif}(0, 1), & \text{else} \end{cases}$$

Singer'09; Wang & Singer'12; Bandeira et al'14; Boumal'16; Liu et al'16, Perry et al'16 ...

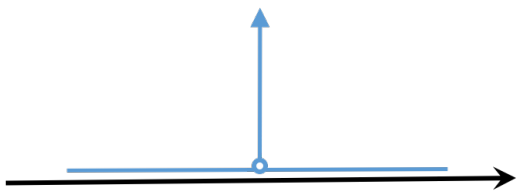
Large-m case: random corruption model

$$y_{i,j} = \begin{cases} x_i - x_j, & \text{with prob. } \pi_0 \\ \mathsf{Unif}(m), & \text{else} \end{cases}$$

Theorem (C. & Candès '16) Suppose $\log n \lesssim m \lesssim \mathrm{poly}(n)$. PPM succeeds if

$$\pi_0 \gtrsim \frac{1}{\sqrt{n}}$$

- Spiky model: when $m \gg n$, model converges to



$$x_i \in [0,1), \qquad y_{i,j} = \begin{cases} x_i - x_j, & \text{with prob. } \pi_0 \\ \mathsf{Unif}(0,1), & \text{else} \end{cases}$$

  - Succeeds even if a dominant fraction $1 - O(1/\sqrt{n})$ of inputs are corrupted

Singer'09; Wang & Singer'12; Bandeira et al'14; Boumal'16; Liu et al'16, Perry et al'16 ...

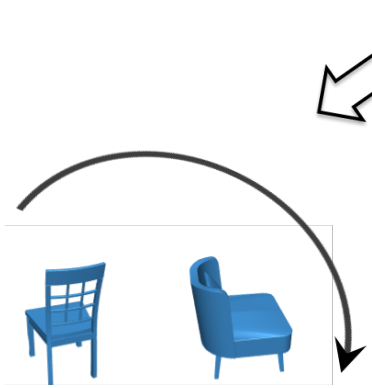
Joint shape alignment: Chair dataset from ShapeNet[1]

20 representative 3D shapes (out of 50)

[1] We add extra noise to each point of the shapes to make it more challenging.

Joint shape alignment: Chair dataset from ShapeNet[1]

20 representative 3D shapes (out of 50)

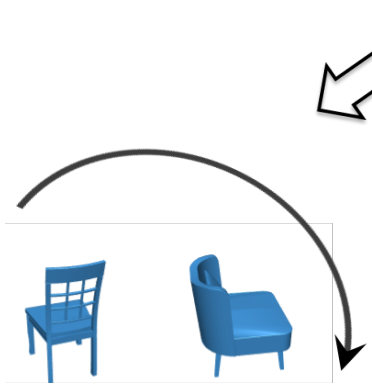


pairwise cost $-\ell_{i,j}(x_i, x_j)$:
avg nearest-neighbor squared distance

[1]We add extra noise to each point of the shapes to make it more challenging.

Joint shape alignment: Chair dataset from ShapeNet[1]

20 representative 3D shapes (out of 50)

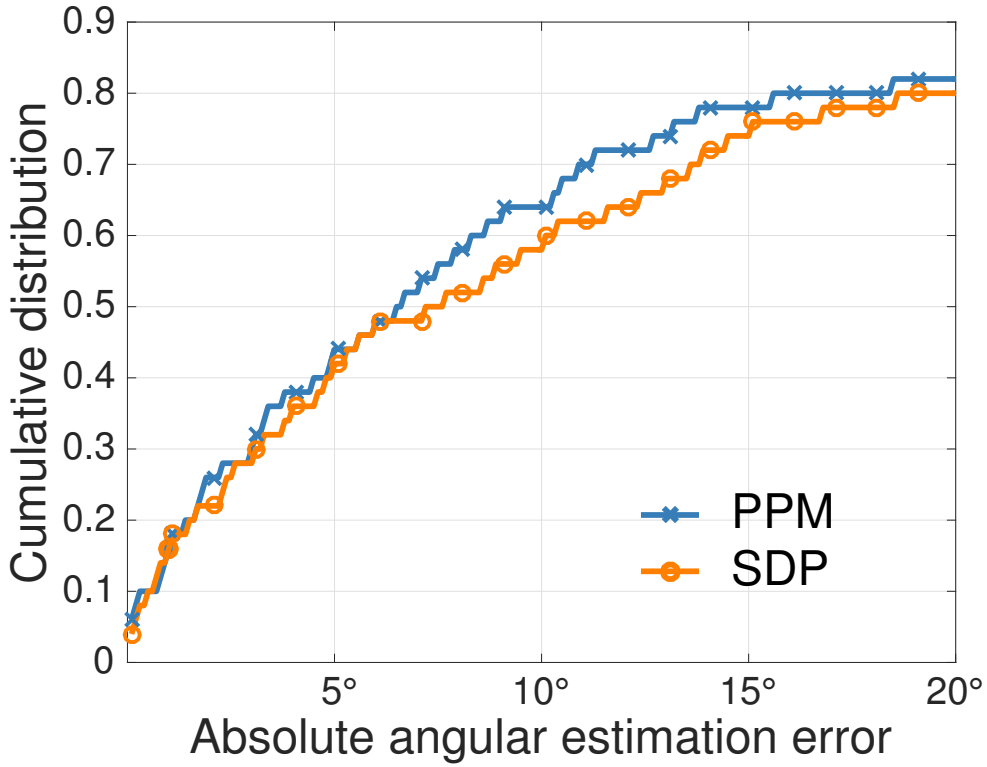


pairwise cost $-\ell_{i,j}(x_i, x_j)$:
avg nearest-neighbor squared distance

aligned shapes

[1]We add extra noise to each point of the shapes to make it more challenging.

Joint shape alignment: angular estimation errors[2]



	projected power method	semidefinite relaxation
Runtime	2.4 sec	895.6 sec

[2]We add extra noise to each point of the shapes to make it more challenging.

Joint graph matching: CMU House dataset

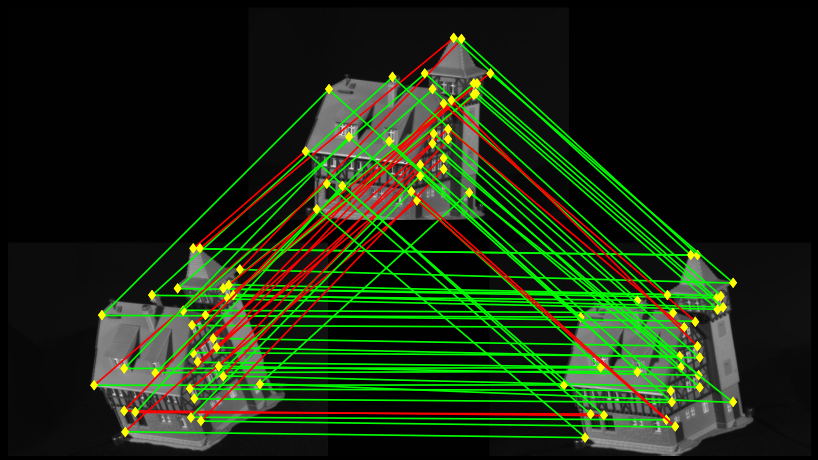
...

111 images of a toy house

Joint graph matching: CMU House dataset

...

111 images of a toy house



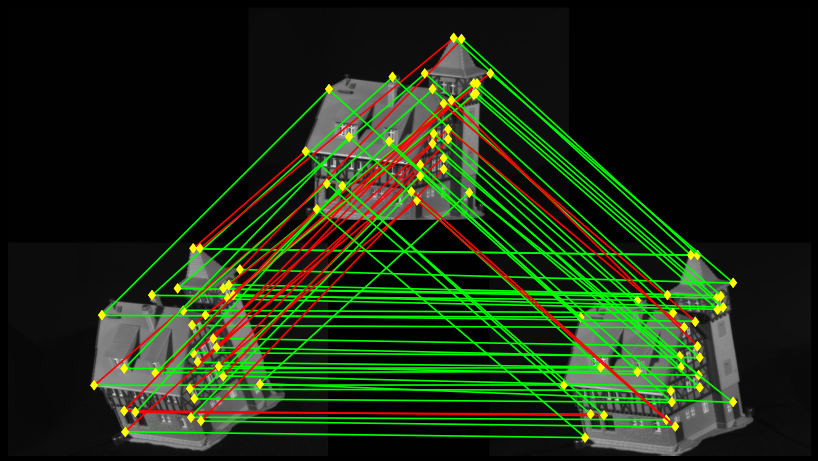
input matches

3 representative images

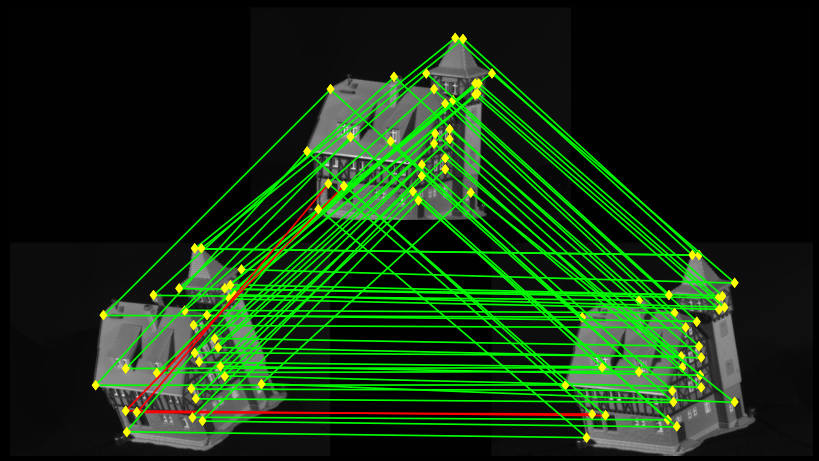
Joint graph matching: CMU House dataset

...

111 images of a toy house



input matches



optimized matches

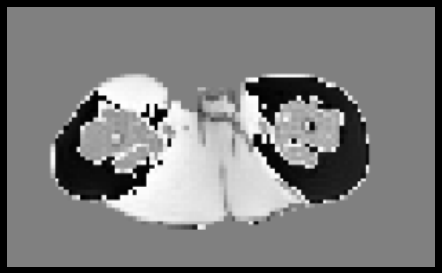
3 representative images

Dixon imaging in body MRI

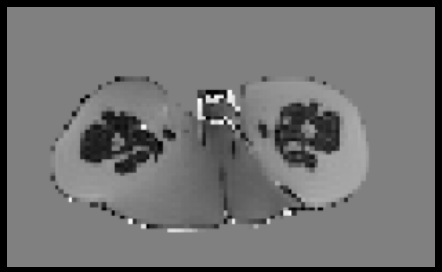
Zhang et al., Magn. Reson. Med., 2016

2 phasor candidates for field inhomogeneity at *each voxel*

candidate 1



candidate 2

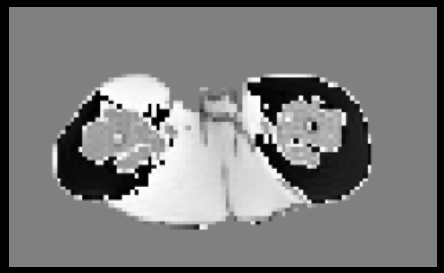


Dixon imaging in body MRI

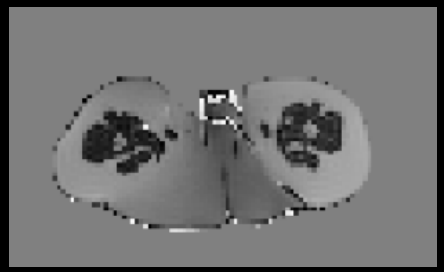
Zhang et al., Magn. Reson. Med., 2016

2 phasor candidates for field inhomogeneity at *each voxel*

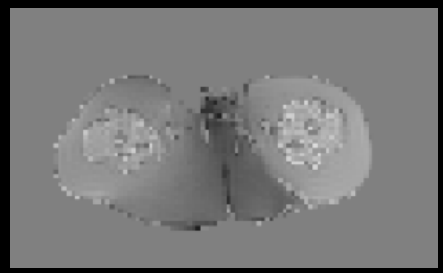
candidate 1



candidate 2



optimize some pariwise cost function



recovery

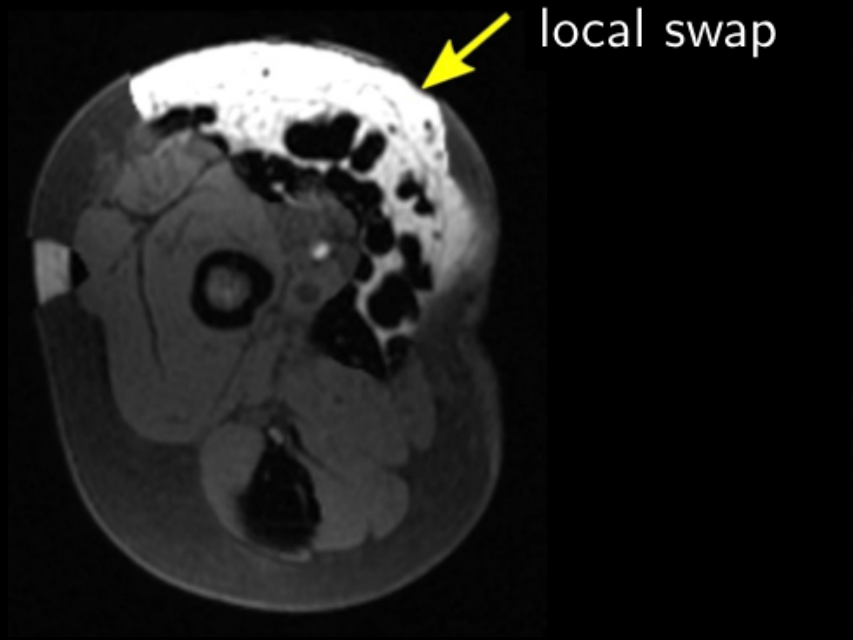
maximize $\sum \ell(x_i, x_j)$

subject to $x_i \in \{1, 2\}$

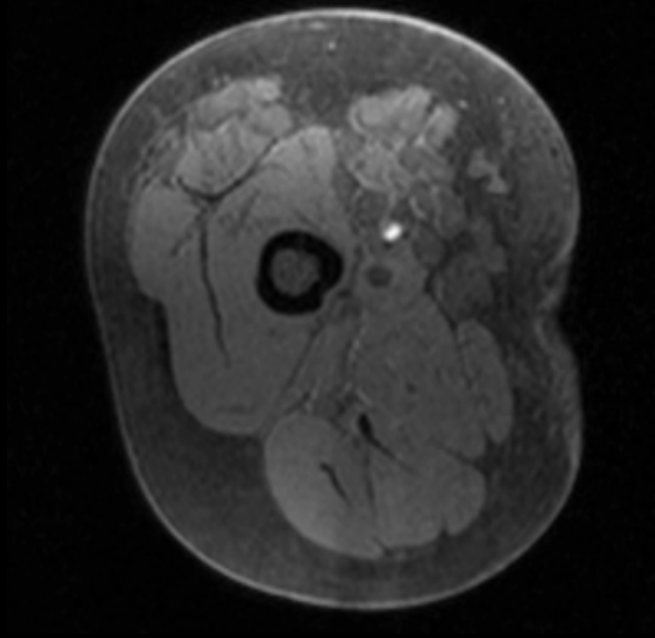
Dixon imaging in body MRI

Zhang et al., Magn. Reson. Med., 2016

Representative cases of water signal recovery



commercial software



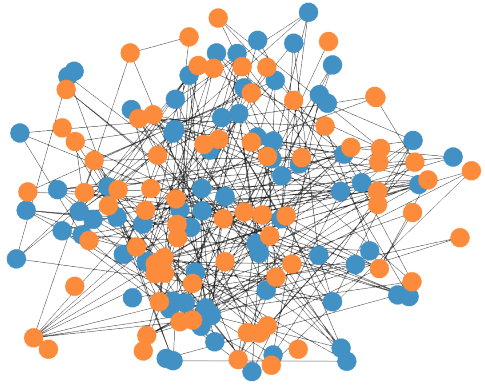
projected power method

Another important issue: missing data + sample locality

Nodes often have locality

Most prior work: (almost) equally likely to sample between any pair of nodes

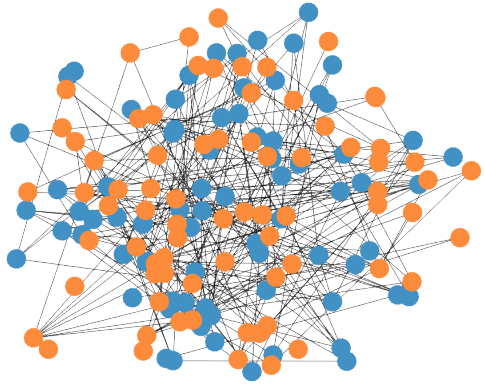
- *Condon et al., Jalali et al., Chen et al., Abbe et al., Mossel et al., Hajek et al., Chin et al...*



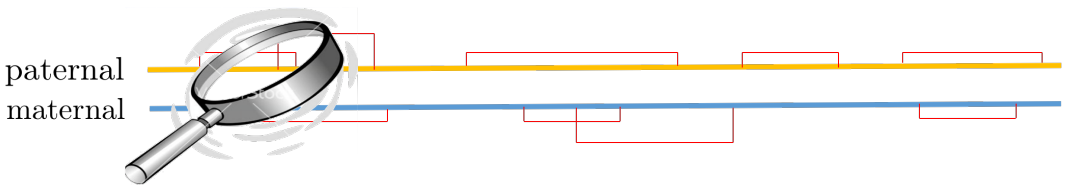
Nodes often have locality

Most prior work: (almost) equally likely to sample between any pair of nodes

- *Condon et al., Jalali et al., Chen et al., Abbe et al., Mossel et al., Hajek et al., Chin et al...*



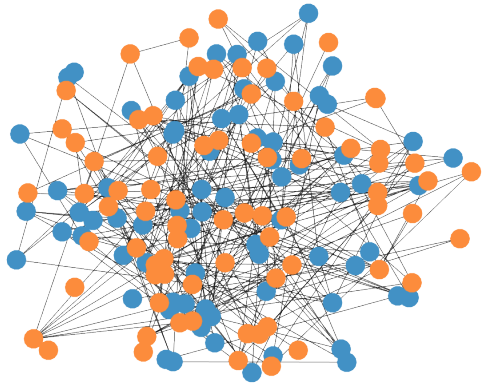
More realistically: samples come mainly (or exclusively) from nearby nodes



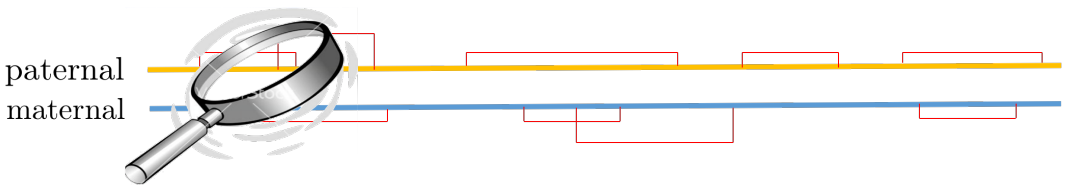
Nodes often have locality

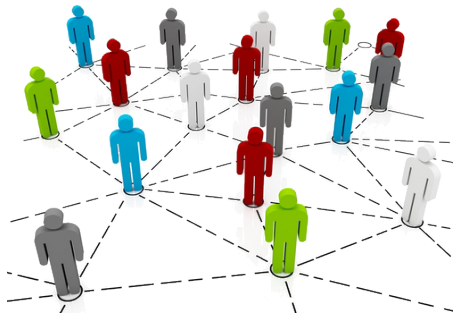
Most prior work: (almost) equally likely to sample between any pair of nodes

- *Condon et al., Jalali et al., Chen et al., Abbe et al., Mossel et al., Hajek et al., Chin et al...*



More realistically: samples come mainly (or exclusively) from nearby nodes

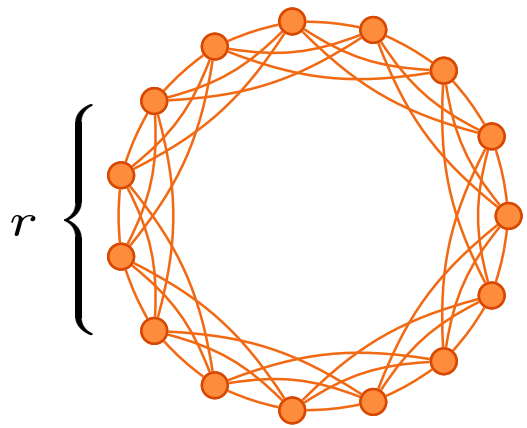




In new technologies like 10x-Genomics: (1) $n \sim 10^5$ SNPs; (2) linking range ~ 100 SNPs

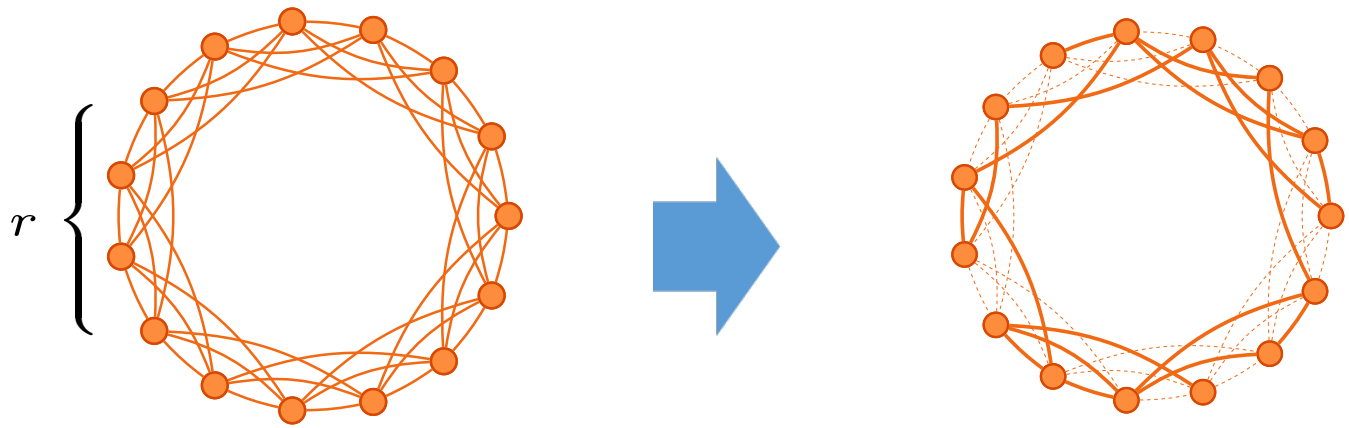
Modeling locality via graphs

- **Constraint graph $\mathcal{G}$**



Modeling locality via graphs

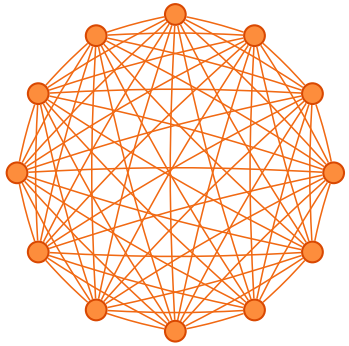
- **Constraint graph** $\mathcal{G}$



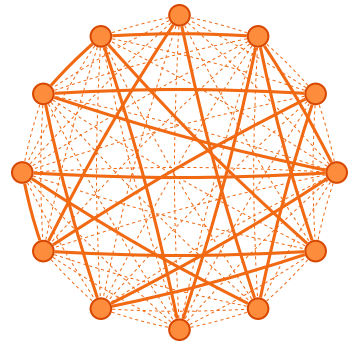
- **Random sampling:** pick m randomly chosen edges of $\mathcal{G}$

Modeling locality via constraint graph

Global / long-range measurements



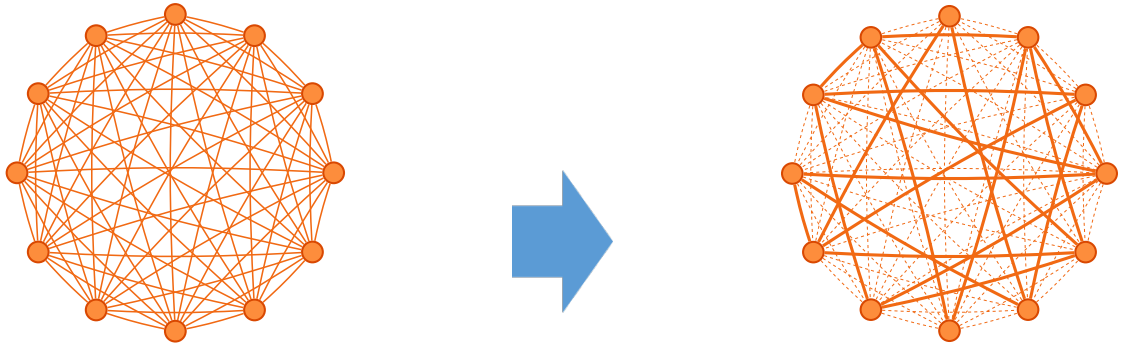
constraint graph



randomly picked edges

Modeling locality via constraint graph

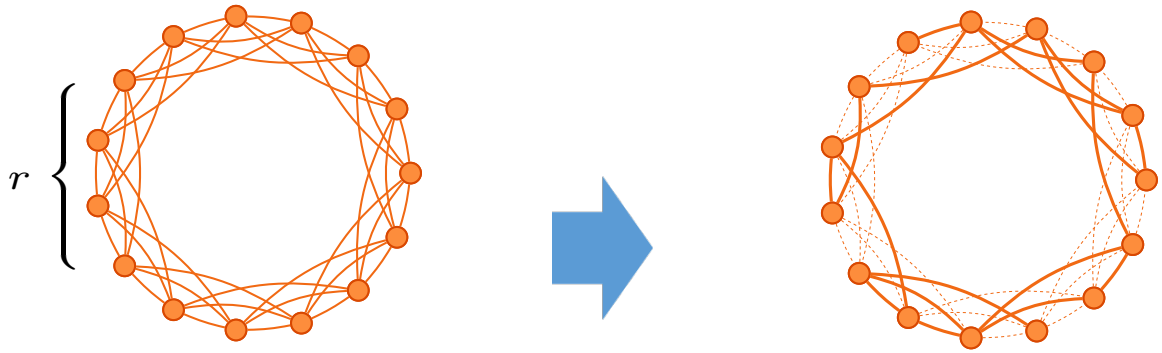
Global / long-range measurements



constraint graph

randomly picked edges

Local measurements



constraint graph
(e.g. $r \sim n^{0.4}$ for 10x)

randomly picked edges

Information and computation limits

1. How many samples are needed to recover $\{x_i\}$ reliably (up to global offset)?

Information and computation limits

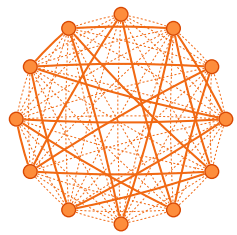
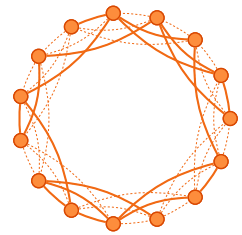
1. How many samples are needed to recover $\{x_i\}$ reliably (up to global offset)?

2. How to extend our paradigm to deal with locality efficiently?

Information and computation limits

1. How many samples are needed to recover $\{x_i\}$ reliably (up to global offset)?

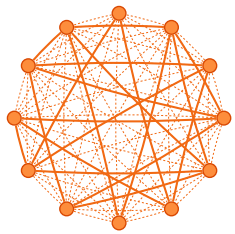
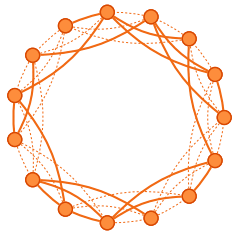
2. How to extend our paradigm to deal with locality efficiently?

	Global samples	Local samples
		
prior works		

Information and computation limits

1. How many samples are needed to recover $\{x_i\}$ reliably (up to global offset)?

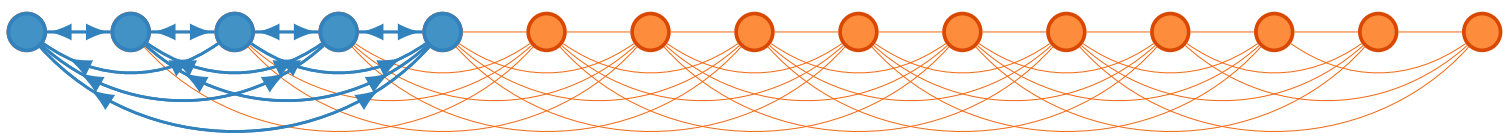
2. How to extend our paradigm to deal with locality efficiently?

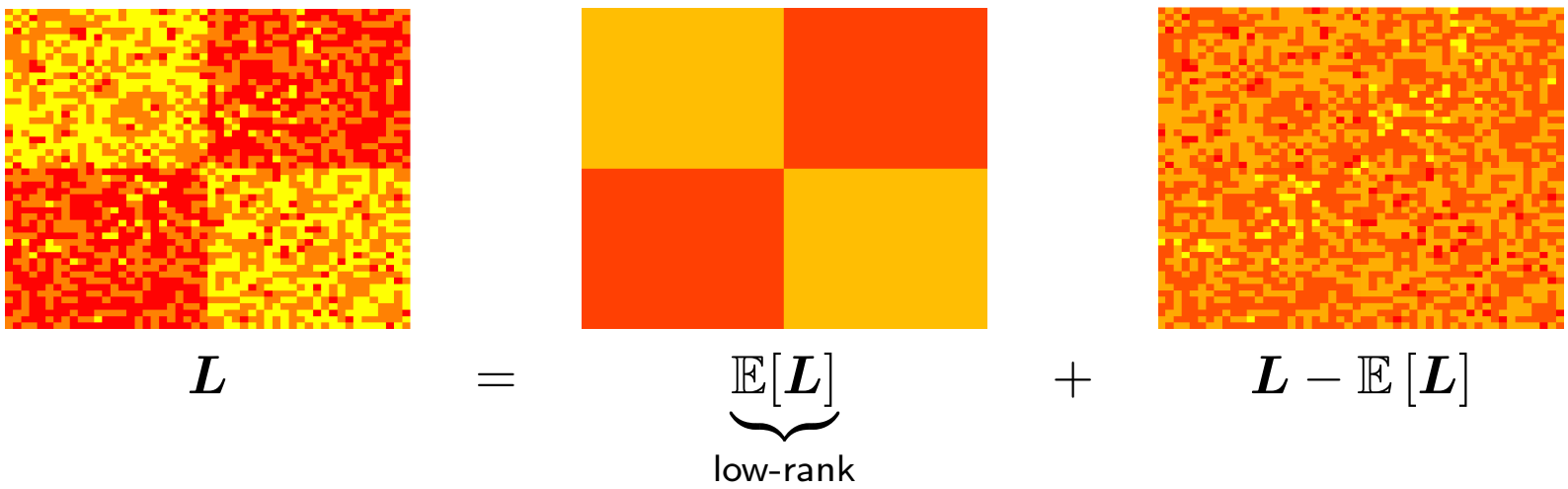
	Global samples	Local samples
		
prior works		

Encouraging news: one can obtain efficient recovery within linear time

Spectral-Stitching: Stage 1

Start by running spectral method on core complete subgraphs

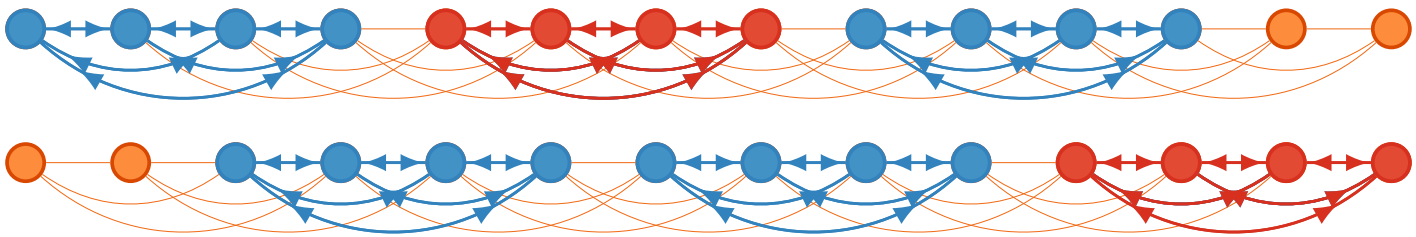




- Compute low-rank approximation of $\boldsymbol{L}$ (*sample matrix restricted to the subgraph*)

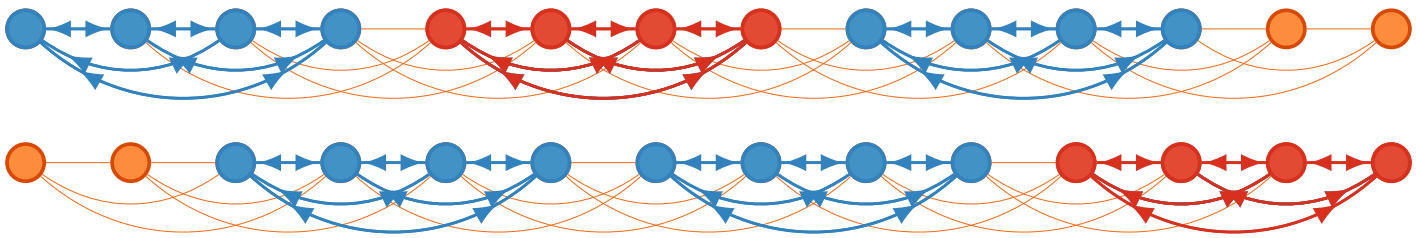
Spectral-Stitching: Stage 1

Split all nodes into overlapping subsets and run spectral methods separately



Spectral-Stitching: Stage 1

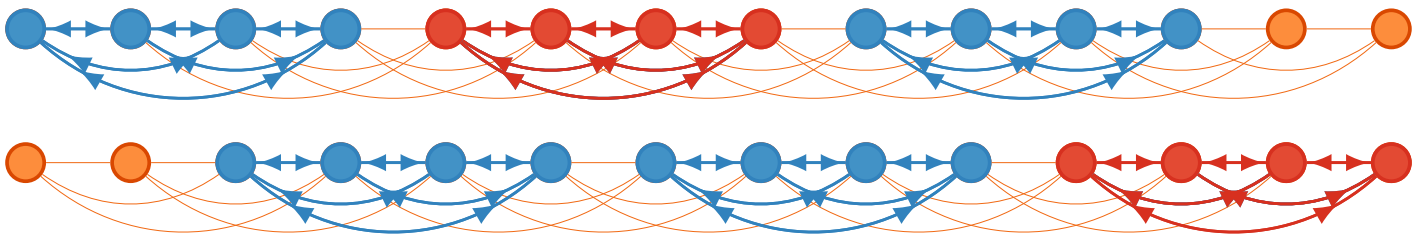
Split all nodes into overlapping subsets and run spectral methods separately



- Approximate solution within each subgraph

Spectral-Stitching: Stage 1

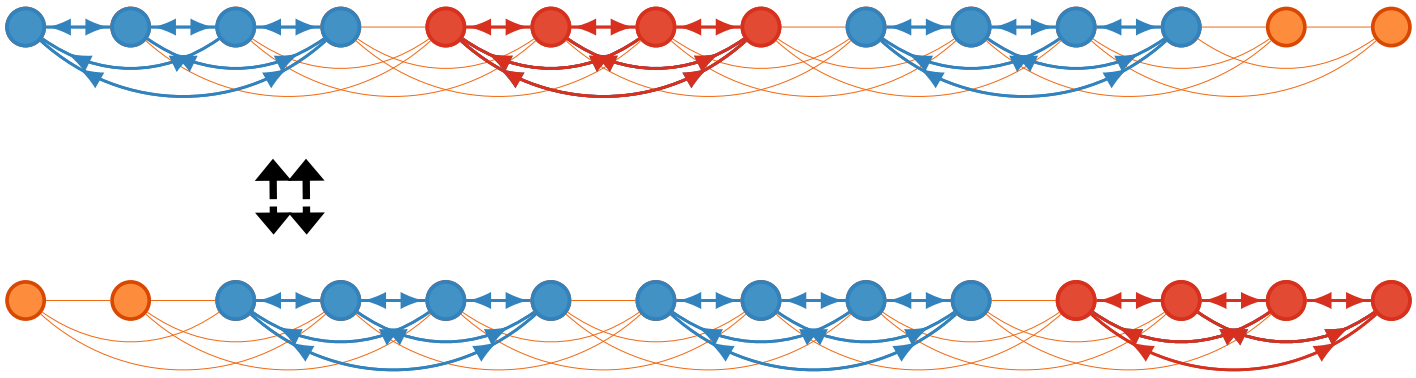
Split all nodes into overlapping subsets and run spectral methods separately



- Approximate solution within each subgraph
- Inconsistent global phases across subgraphs

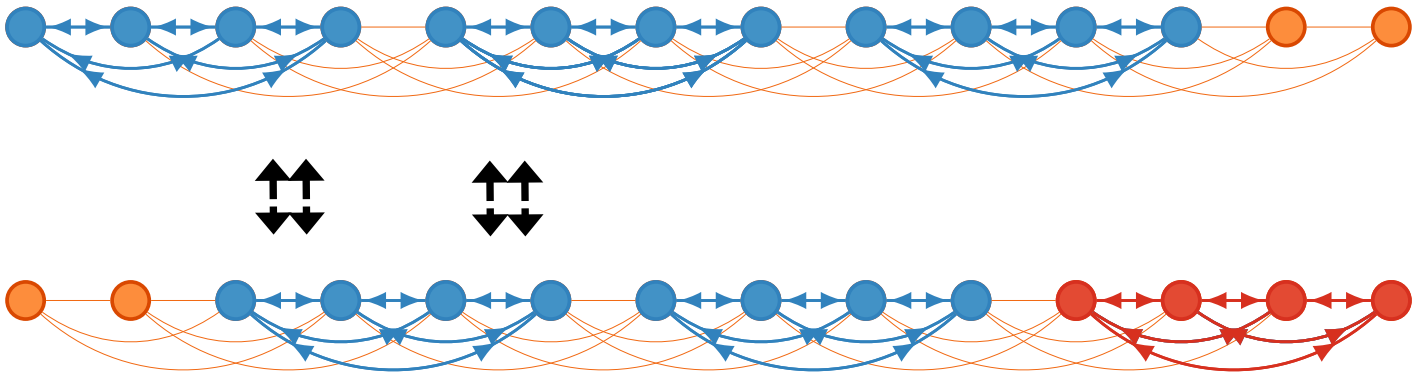
Spectral-Stitching: Stage 2

Calibrate phases across subgraphs by checking their correlations



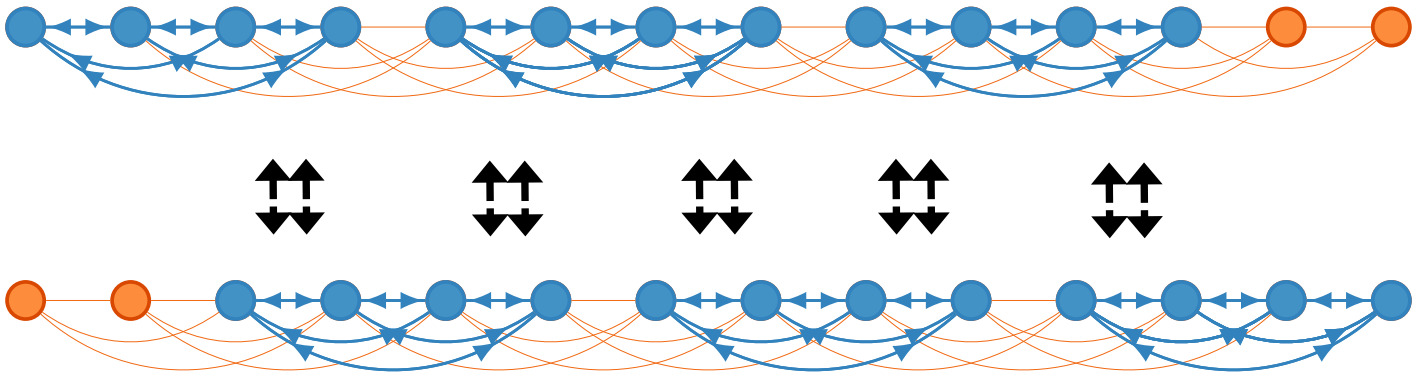
Spectral-Stitching: Stage 2

Calibrate phases across subgraphs by checking their correlations



Spectral-Stitching: Stage 2

Calibrate phases across subgraphs by checking their correlations

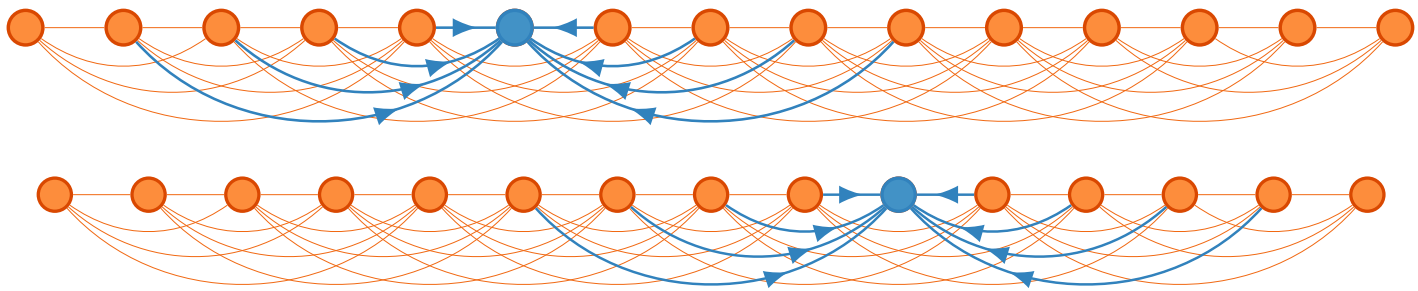


Purpose of Stages 1-2: obtain approximate solution of all nodes

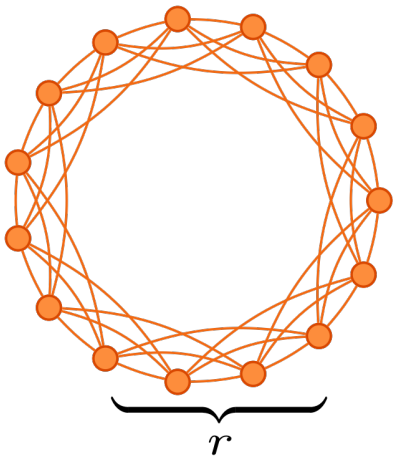
Spectral-Stitching: Stage 3

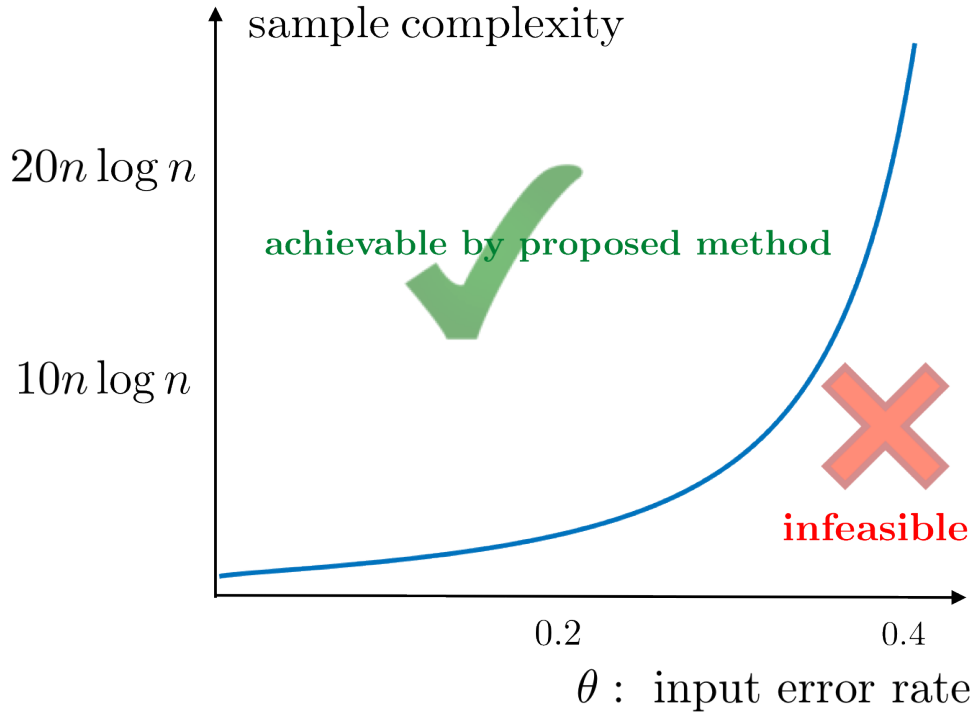
Clean up all remaining errors by iterative refinement (e.g. projected power method)

- local refinement using *all* samples

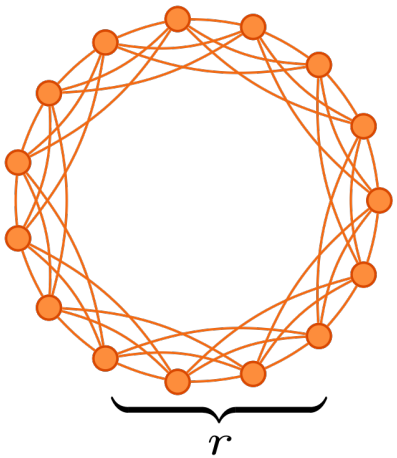


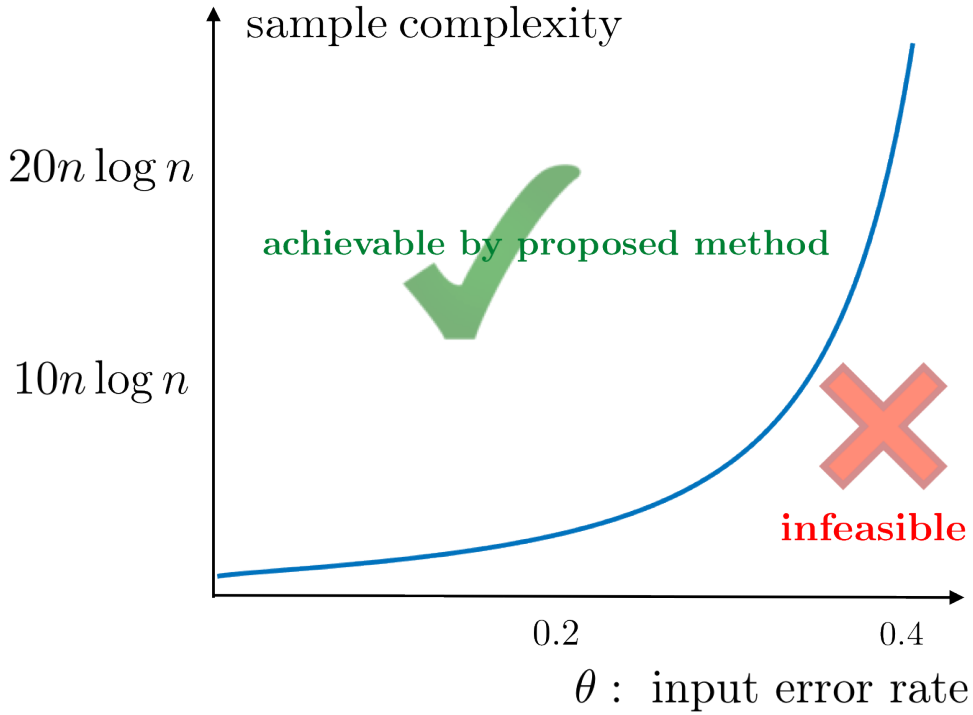
Main results: rings





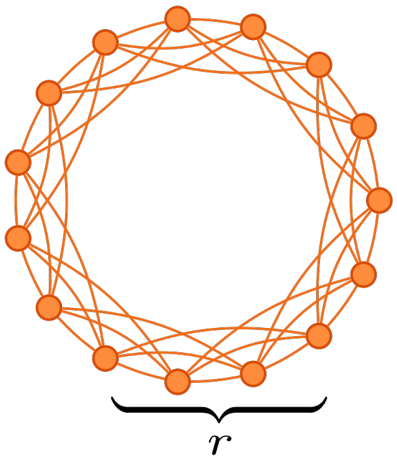
Main results: rings

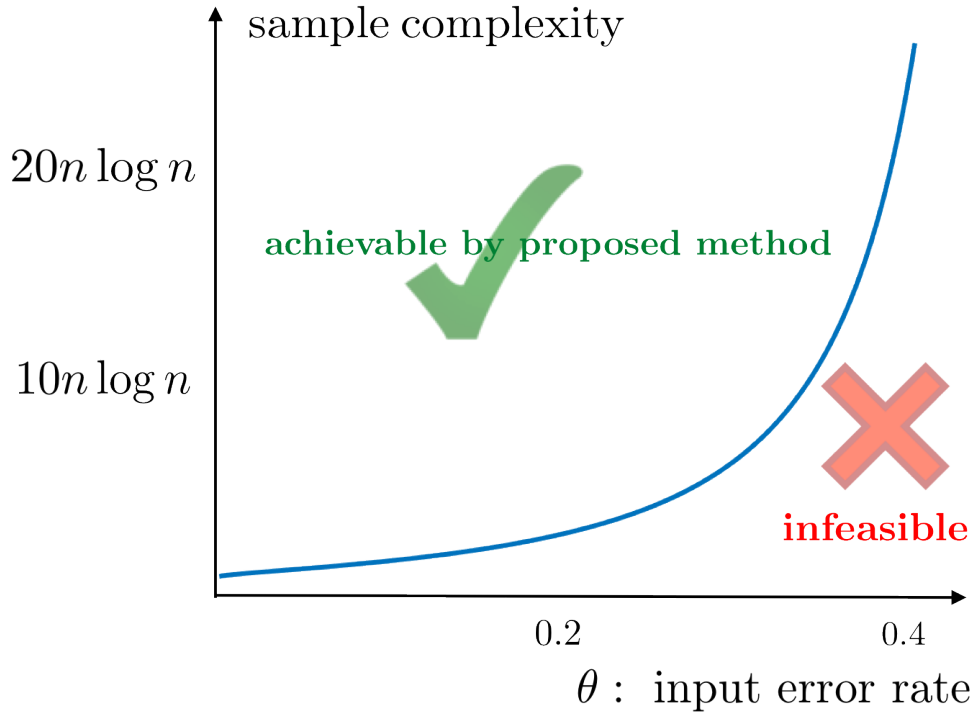




Theorem: minimum sample complexity $= \frac{0.5n \log n}{1-\exp\{-\text{Chernoff}-\text{info})}$

Main results: rings

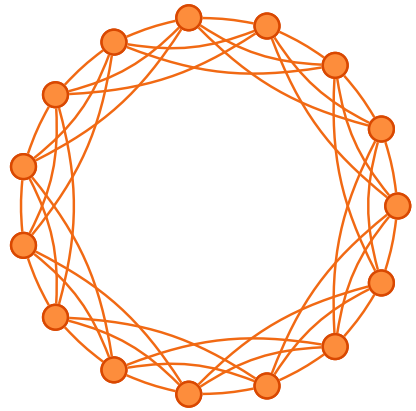




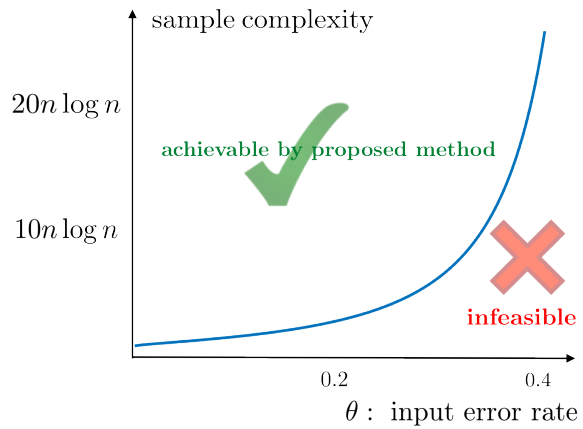
Theorem: minimum sample complexity $= \frac{0.5n \log n}{1-\exp\{-\text{Chernoff}-\text{info})}$

Info and comput. limits meet!

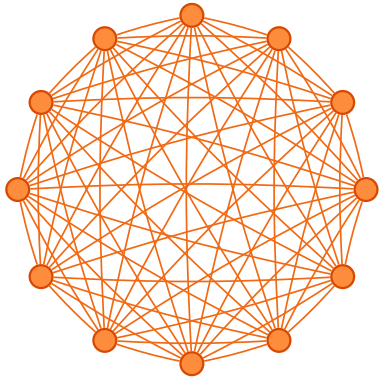
An insensitivity phenomenon



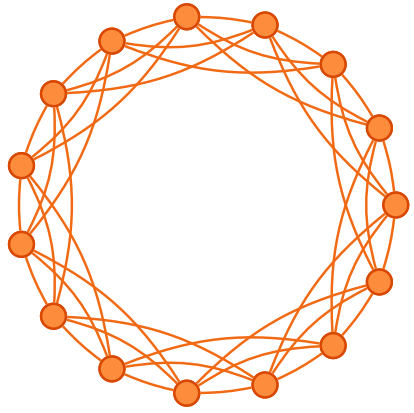
ring



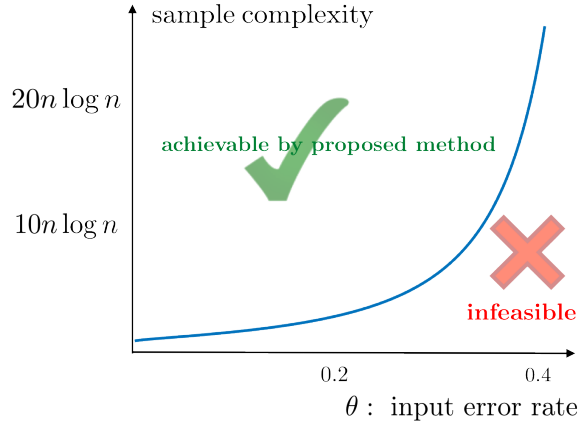
An insensitivity phenomenon



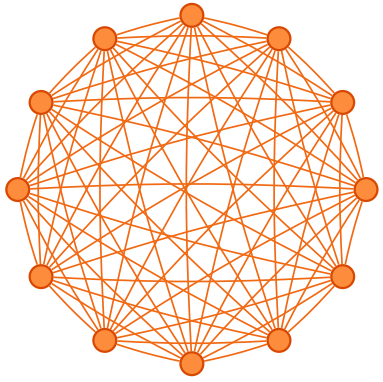
complete graph



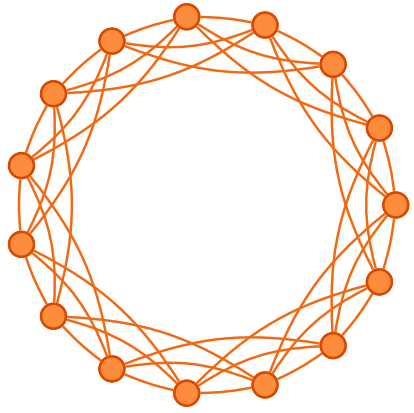
ring



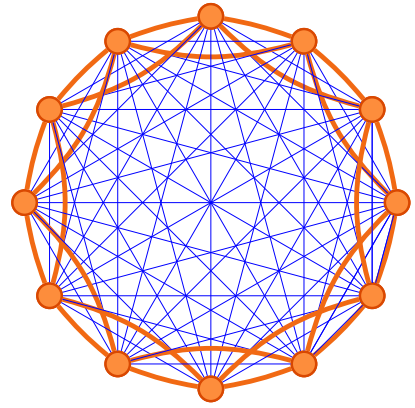
An insensitivity phenomenon



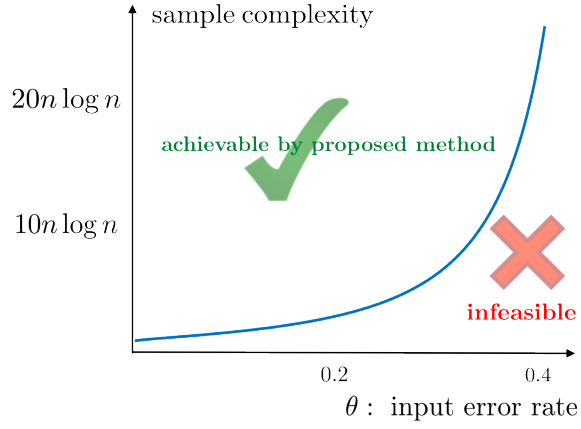
complete graph



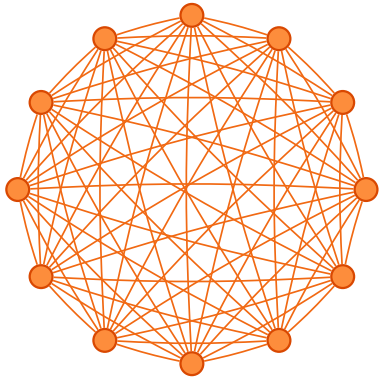
ring



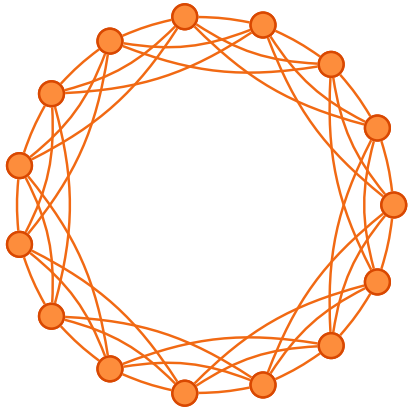
small-world



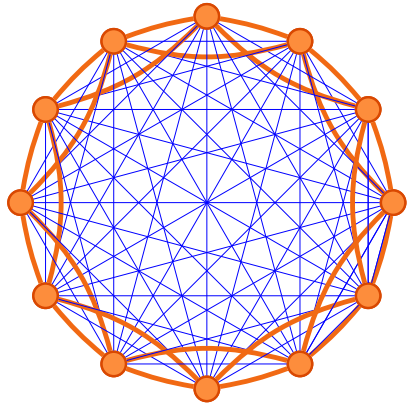
An insensitivity phenomenon



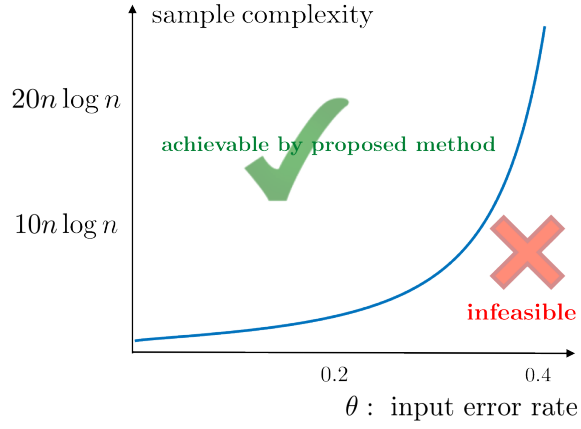
complete graph



ring

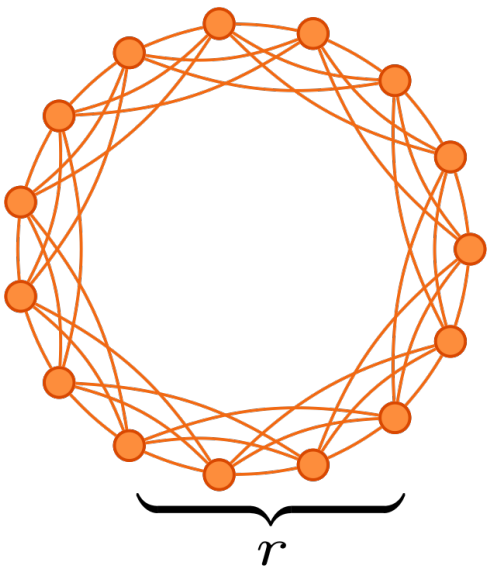


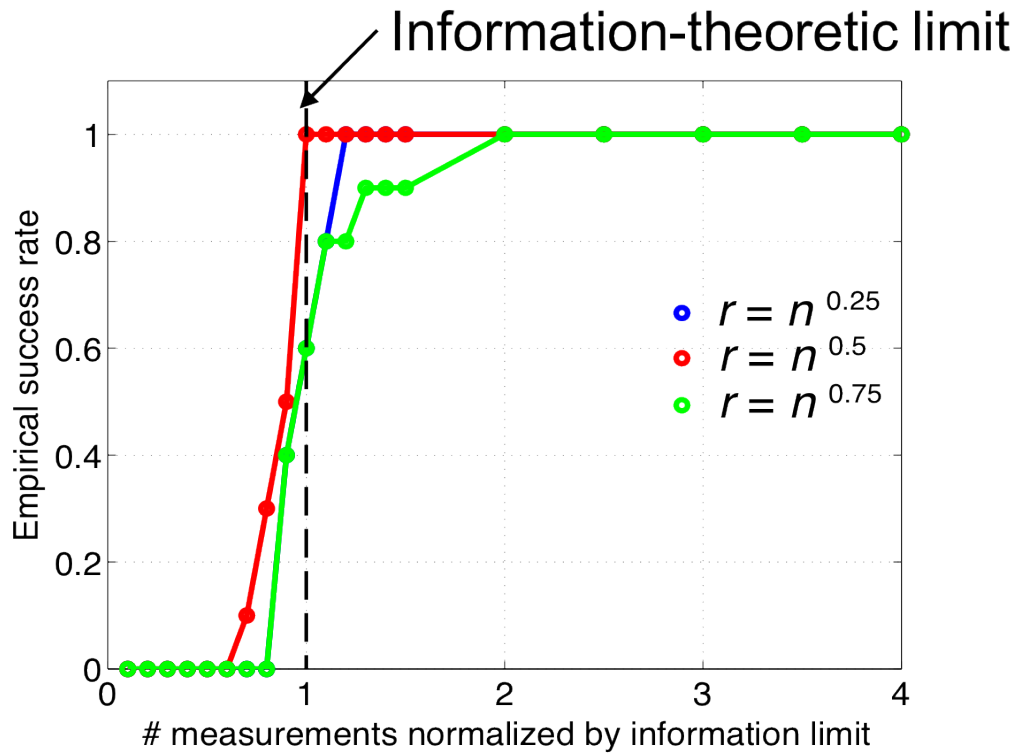
small-world



Info and comput. limits are identical for many spatially invariant graphs

Empirical success rate vs. sample size



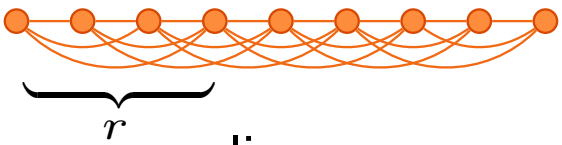


$n = 100,000,$ input error rate $= 0.2$

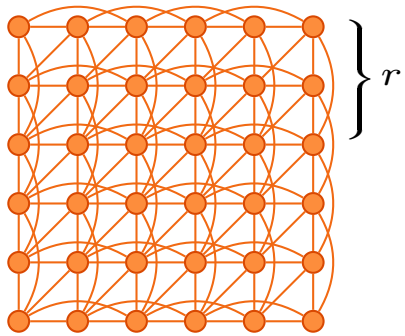
10 Monte Carlo runs to get each point

Each run takes $\sim$6.4 sec on a Mac Pro

Extension: beyond spatially invariant graphs

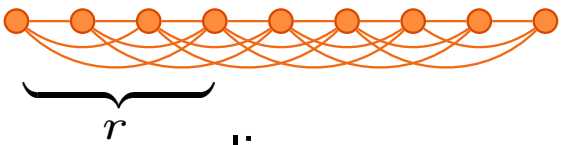


lines

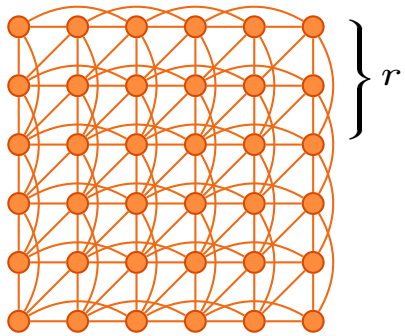


grids

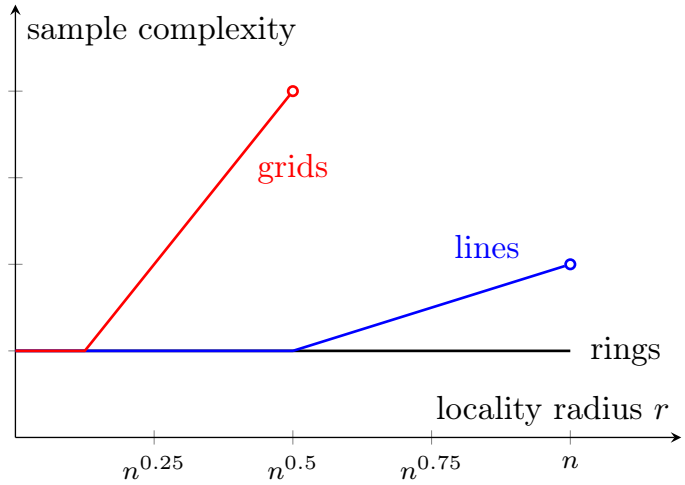
Extension: beyond spatially invariant graphs



lines



grids



Info limit vs. r

Infomation and comput. limits achievable by same algorithm

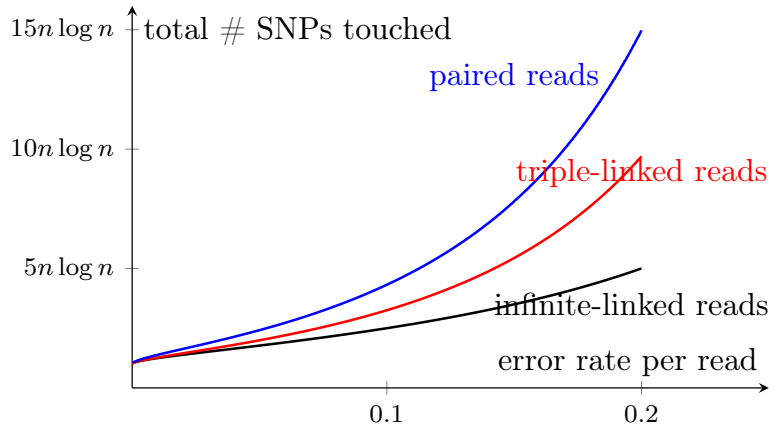
Extension: beyond pairwise measurements

New technologies (e.g. 10x) provide multi-linked reads from same chromosome, not just two

Extension: beyond pairwise measurements

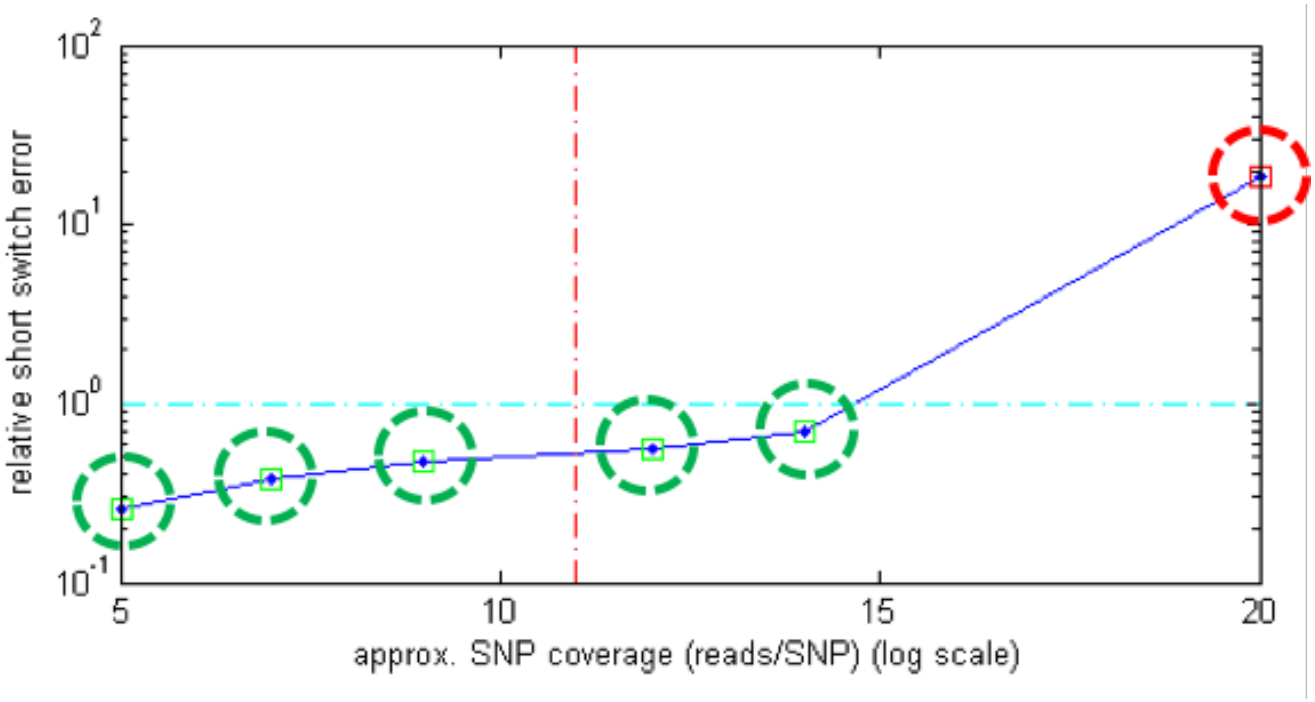
New technologies (e.g. 10x) provide multi-linked reads from same chromosome, not just two

Algorithm and theory can be easily extended to see performance gain



Real data (haplotype phasing)

NA12878_WGS dataset from 10x genomics (# SNPs $n : 34240 \sim 191829$)

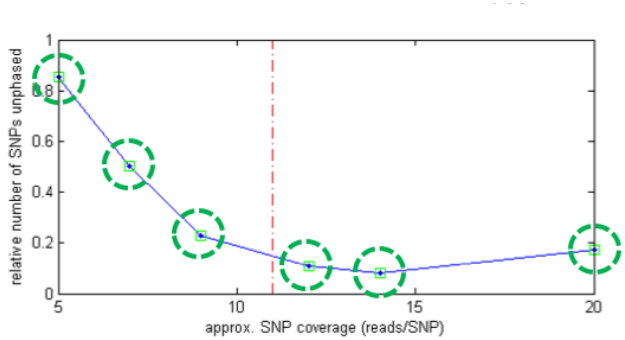


Short switch error rate vs. coverage depth (Spectral-Stitching vs. 10X algorithm)
(green circle: improvement; red circle: loss of performance)

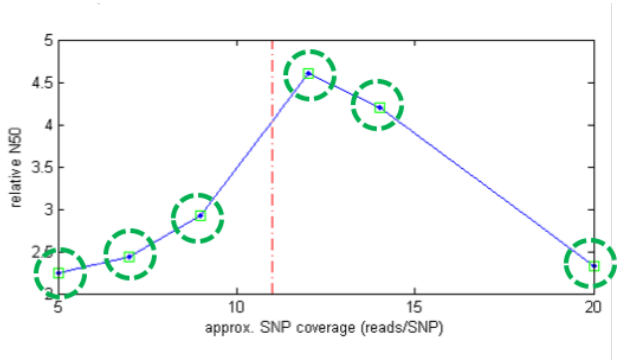
Fig. credit: Prof. David Tse, Stanford

Real data (haplotype phasing)

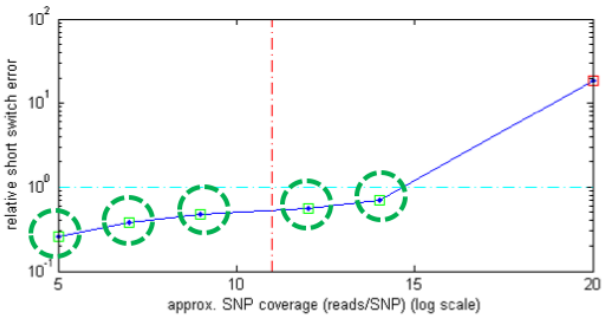
NA12878_WGS dataset from 10x genomics (# SNPs n : $34240 \sim 191829$)



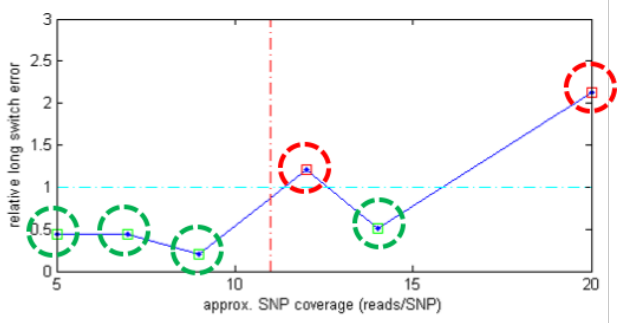
1) # SNPs unphased;



2) relative N50



3) short switch error rate;
(green circle: improvement;



4) long switch error rate
red circle: loss of performance)

Fig. credit: Prof. David Tse, Stanford

Concluding remarks

- Nonconvex procedures are efficient for many discrete optimization problems
- Information limits can be achieved in linear time for a broad family of models

Papers:

1. "The projected power method: an efficient algorithm for joint alignment from pairwise differences", Y. Chen and E. Candès, 2016

2. "Community recovery in graphs with locality", Y. Chen, G. Kamath, C. Suh, and D. Tse, International Conference on Machine Learning, 2016

3. "Resolving phase ambiguity in dual-echo Dixon imaging using a projected power method", T. Zhang et al, Magnetic Resonance in Medicine, 2016