Distance Geometry and Data Science

Leo Liberti, CNRS & Ecole Polytechnique
liberti@lix.polytechnique.fr

Winter School in GCS @FieldsInstitute 210111-15

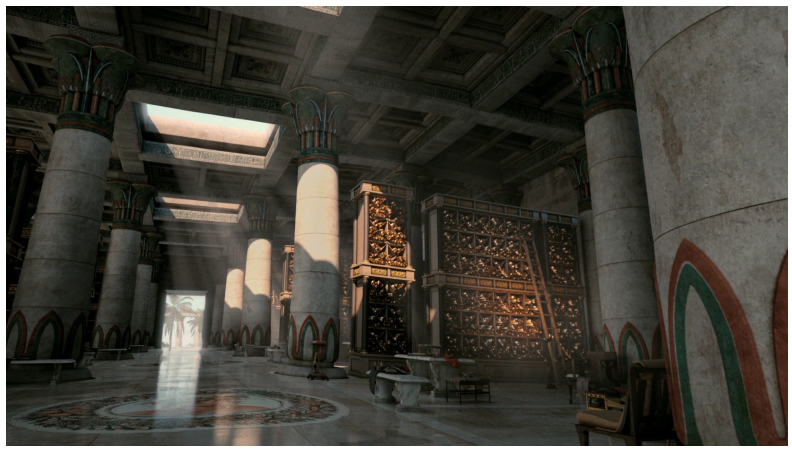
Most material from [L., TOP 28:271-339, 2020]

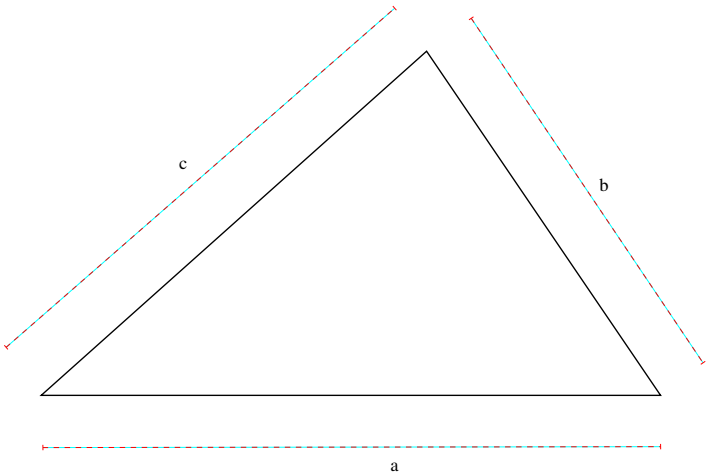
http://www.lix.polytechnique.fr/~liberti/dgds.pdf

DG's best known result

A gem in Distance Geometry

- *Heron's theorem*
- Heron lived around year 0
- Hang out at Alexandria's library





$$A = \sqrt{s(s-a)(s-b)(s-c)}$$

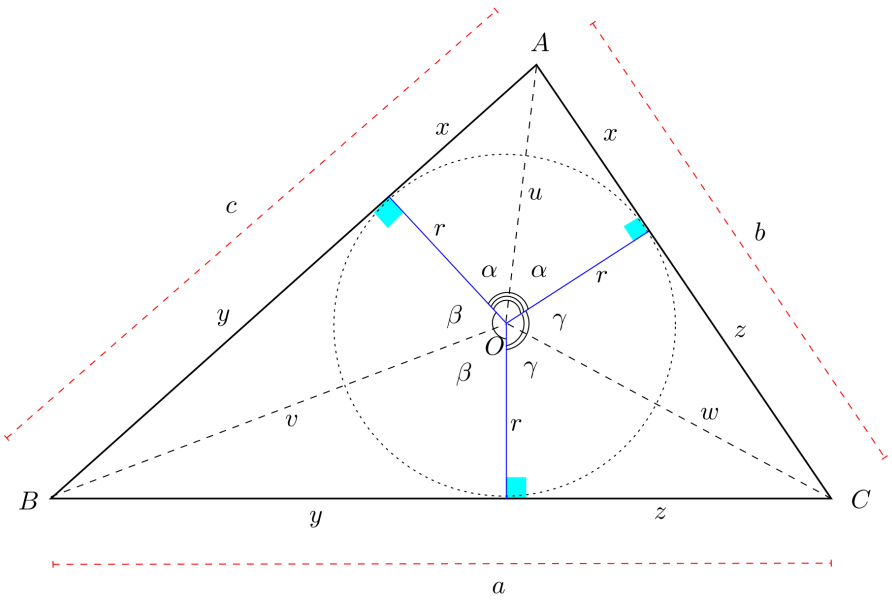
- $A =$ area of triangle
- $s = \frac{1}{2}(a+b+c)$

Useful to measure areas of agricultural land

Subsection 1

Proof of Heron's theorem

Heron's theorem: *Proof* [M. Edwards, high school student, 2007]



A. $2\alpha + 2\beta + 2\gamma = 2\pi \Rightarrow \alpha + \beta + \gamma = \pi$

$$\begin{aligned} r + ix &= ue^{i\alpha} \\ r + iy &= ve^{i\beta} \\ r + iz &= we^{i\gamma} \end{aligned}$$

$\Rightarrow (r+ix)(r+iy)(r+iz) = (uvw)e^{i(\alpha+\beta+\gamma)} =$

$uvw\, e^{i\pi} = -uvw \in \mathbb{R}$

$\Rightarrow \mathsf{Im}((r+ix)(r+iy)(r+iz)) = 0$

$\Rightarrow r^2(x+y+z) = xyz \Rightarrow r = \sqrt{\frac{xyz}{x+y+z}}$

B. $s = \frac{1}{2}(a+b+c) = x+y+z$

$$\begin{aligned} s - a &= x + y + z - y - z = x \\ s - b &= x + y + z - x - z = y \\ s - c &= x + y + z - x - y = z \end{aligned}$$

$$\mathcal{A} = \frac{1}{2}(ra + rb + rc) = r\frac{a+b+c}{2} = rs = \sqrt{s(s-a)(s-b)(s-c)}$$

Metric spaces representability

Representing metric spaces in $\mathbb{R}^n$

- Given metric space (X, d) with dist. matrix $D = (d_{ij})$, *embed X in a Euclidean space with same dist. matrix*
- Consider i-th row $x_i = (d_{i1}, \ldots, d_{in})$ of D
- Embed $i \in X$ by vector $U^D(i) = x_i \in \mathbb{R}^n$
  define $U^D : \{1, \ldots, n\} \to \mathbb{R}^n$ s.t. $U^D(i) = x_i$
- Thm.: (U^D, ℓ_∞) is a metric space with dist. matrix D
  i.e. $\forall i, j \leq n \; \|x_i - x_j\|_\infty = d_{ij}$
- U^D is called Universal Isometric Embedding (UIE)
  also known as Fréchet's embedding
- Practical issue: *embedding is high-dimensional ($\mathbb{R}^n$)*

[Kuratowski 1935]

Proof

- Consider $i, j \in X$ with distance $d(i,j) = d_{ij}$
- Then

$$\|x_i - x_j\|_\infty = \max_{k \le n} |d_{ik} - d_{jk}| \le \max_{k \le n} |d_{ij}| = d_{ij}$$

ineq. $\le$ *above from triangular inequalities in metric space:*

$$\begin{aligned} & d_{ik} \le d_{ij} + d_{jk} \quad \wedge \quad d_{jk} \le d_{ij} + d_{ik} \\ \Rightarrow \quad & d_{ik} - d_{jk} \le d_{ij} \quad \wedge \quad d_{jk} - d_{ik} \le d_{ij} \\ \Rightarrow \quad & |d_{ik} - d_{jk}| \le d_{ij} \end{aligned}$$

If valid $\forall i, j$ then valid for max

- $\max |d_{ik} - d_{jk}|$ over $k \le n$ achieved when $k \in \{i, j\}$

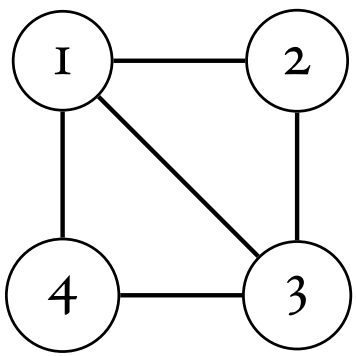
$$\Rightarrow \|x_i - x_j\|_\infty = d_{ij}$$

Subsection 1

Missing distances

UIE from incomplete metrics

- If your metric space is missing some distances
- Get incomplete distance matrix D
- Cannot define vectors $U^D(i)$ in UIE
- *Note*: D defines a graph

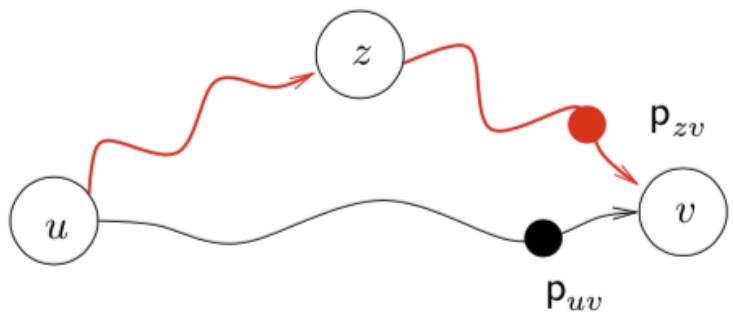


$$D = \begin{pmatrix} 0 & 1 & \sqrt{2} & 1 \\ 1 & 0 & 1 & ? \\ \sqrt{2} & 1 & 0 & 1 \\ 1 & ? & 1 & 0 \end{pmatrix}$$

- Complete this graph with shortest paths: $d_{24} = 2$

Floyd-Warshall algorithm 1/2

- Given $n \times n$ partial matrix D computes
  all shortest path lengths
- For each triplet u, v, z of vertices in the graph, test:
  when going $u \to v$, is it convenient to pass through z?



- If so, then change the path length

Floyd-Warshall algorithm 2/2

```
# initialization
for u ≤ n, v ≤ n do
    if d_uv =? then
        d_uv ← ∞
    end if
end for
# main loop
for z ≤ n do
    for u ≤ n do
        for v ≤ n do
            if d_uv > d_uz + d_zv then
                d_uv ← d_uz + d_zv
            end if
        end for
    end for
end for
```

Subsection 2

Noisy distances

Schoenberg's theorem

- [I. Schoenberg, *Remarks to Maurice Fréchet's article "Sur la définition axiomatique d'une classe d'espaces distanciés vectoriellement applicable sur l'espace de Hilbert"*, Ann. Math., 1935]
- Question:
  When is a given matrix a *Euclidean Distance Matrix* (EDM)?

  Thm.
  $D = (d_{ij})$ is an EDM iff $\frac{1}{2}(d_{1i}^2 + d_{1j}^2 - d_{ij}^2 \mid 2 \leq i, j \leq n)$ is PSD of rank K

Gram in function of EDM

- $x = (x_1, \ldots, x_n) \subseteq \mathbb{R}^K$, written as $n \times K$ matrix
- matrix $G = xx^\top = (x_i \cdot x_j)$ is the *Gram matrix* of x

  Lemma: $G \succeq 0$ *and each* $M \succeq 0$ *is a Gram matrix of some* x
- Useful variant of Schoenberg's theorem

  Relates EDMs and Gram matrices

$$G = -\frac{1}{2} J D^2 J \qquad (\star)$$

- where $D^2 = (d_{ij}^2)$ and

$$J = I_n - \tfrac{1}{n}\mathbf{1}\mathbf{1}^\top = \begin{pmatrix} 1 - \frac{1}{n} & -\frac{1}{n} & \cdots & -\frac{1}{n} \\ -\frac{1}{n} & 1 - \frac{1}{n} & \cdots & -\frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{n} & -\frac{1}{n} & \cdots & 1 - \frac{1}{n} \end{pmatrix}$$

Multidimensional scaling (MDS)

- Often get approximate EDMs $\tilde{D}$ from raw data
  (dissimilarities, discrepancies, differences)
- $\tilde{G} = -\frac{1}{2} J \tilde{D}^2 J$ is an approximate Gram matrix
- Approximate Gram $\Rightarrow$ spectral decomposition $P \tilde{\Lambda} P^\top$ has $\tilde{\Lambda} \not\geq 0$
- Let Λ closest PSD diagonal matrix to $\tilde{\Lambda}$:
  zero the negative components of $\tilde{\Lambda}$
- $x = P\sqrt{\Lambda}$ is an "approximate embedding" of $\tilde{D}$

Classic MDS: Main result

1. Prove lemma: matrix is Gram iff it is PSD
2. Prove that $G = -\frac{1}{2} J D^2 J$

Proof of lemma

- *Gram* $\subseteq$ *PSD*
  - x is an $n \times K$ real matrix
  - $G = xx^\top$ its Gram matrix
  - For each $y \in \mathbb{R}^n$ we have

    $$yGy^\top = y(xx^\top)y^\top = (yx)(x^\top y^\top) = (yx)(yx)^\top = \|yx\|_2^2 \geq 0$$

  - $\Rightarrow G \succeq 0$
- *PSD* $\subseteq$ *Gram*
  - Let $G \succeq 0$ be $n \times n$
  - *Spectral decomposition*: $G = P\Lambda P^\top$

    (P orthogonal, $\Lambda \geq 0$ diagonal)
  - $\Lambda \geq 0 \Rightarrow \sqrt{\Lambda}$ exists
  - $G = P\Lambda P^\top = (P\sqrt{\Lambda})(\sqrt{\Lambda}^\top P^\top) = (P\sqrt{\Lambda})(P\sqrt{\Lambda})^\top$
  - Let $x = P\sqrt{\Lambda}$, then G is the Gram matrix of x

Schoenberg's theorem proof (1/2)

- Assume zero centroid WLOG (can translate x as needed)
- Expand: $d_{ij}^2 = \|x_i - x_j\|_2^2 = (x_i - x_j)(x_i - x_j) = x_i x_i + x_j x_j - 2x_i x_j \quad (*)$
- Aim at "inverting" $(*)$ to express $x_i x_j$ in function of d_{ij}^2
- Sum $(*)$ over i: $\sum_i d_{ij}^2 = \sum_i x_i x_i + n x_j x_j - 2x_j \cancelto{0}{\sum_i x_i}$ 0 by zero centroid
- Similarly for j and divide by n, get:

$$\frac{1}{n}\sum_{i\leq n} d_{ij}^2 = \frac{1}{n}\sum_{i\leq n} x_i x_i + x_j x_j \quad (\dagger)$$

$$\frac{1}{n}\sum_{j\leq n} d_{ij}^2 = x_i x_i + \frac{1}{n}\sum_{j\leq n} x_j x_j \quad (\ddagger)$$

- Sum $(\dagger)$ over j, get:

$$\frac{1}{n}\sum_{i,j} d_{ij}^2 = n\frac{1}{n}\sum_i x_i x_i + \sum_j x_j x_j = 2\sum_i x_i x_i$$

- Divide by n, get:

$$\frac{1}{n^2}\sum_{i,j} d_{ij}^2 = \frac{2}{n}\sum_i x_i x_i \quad (**)$$

Schoenberg's theorem proof (2/2)

- Rearrange $(*)$, $(\dagger)$, $(\ddagger)$ as follows:

$$2x_ix_j = x_ix_i + x_jx_j - d_{ij}^2 \quad (1)$$

$$x_ix_i = \frac{1}{n}\sum_j d_{ij}^2 - \frac{1}{n}\sum_j x_jx_j \quad (2)$$

$$x_jx_j = \frac{1}{n}\sum_i d_{ij}^2 - \frac{1}{n}\sum_i x_ix_i \quad (3)$$

- Replace LHS of Eq. (2)-(3) in RHS of Eq. (1), get

$$2x_ix_j = \frac{1}{n}\sum_k d_{ik}^2 + \frac{1}{n}\sum_k d_{kj}^2 - d_{ij}^2 - \frac{2}{n}\sum_k x_kx_k$$

- By $(**)$ replace $\frac{2}{n}\sum_i x_ix_i$ with $\frac{1}{n^2}\sum_{i,j} d_{ij}^2$, get

$$2x_ix_j = \frac{1}{n}\sum_k (d_{ik}^2 + d_{kj}^2) - d_{ij}^2 - \frac{1}{n^2}\sum_{h,k} d_{hk}^2 \quad (\star)$$

expressing x_ix_j in function of D *(showing $(\star) \equiv (2G = -JD^2J)$ is messy but easy)*

Subsection 3

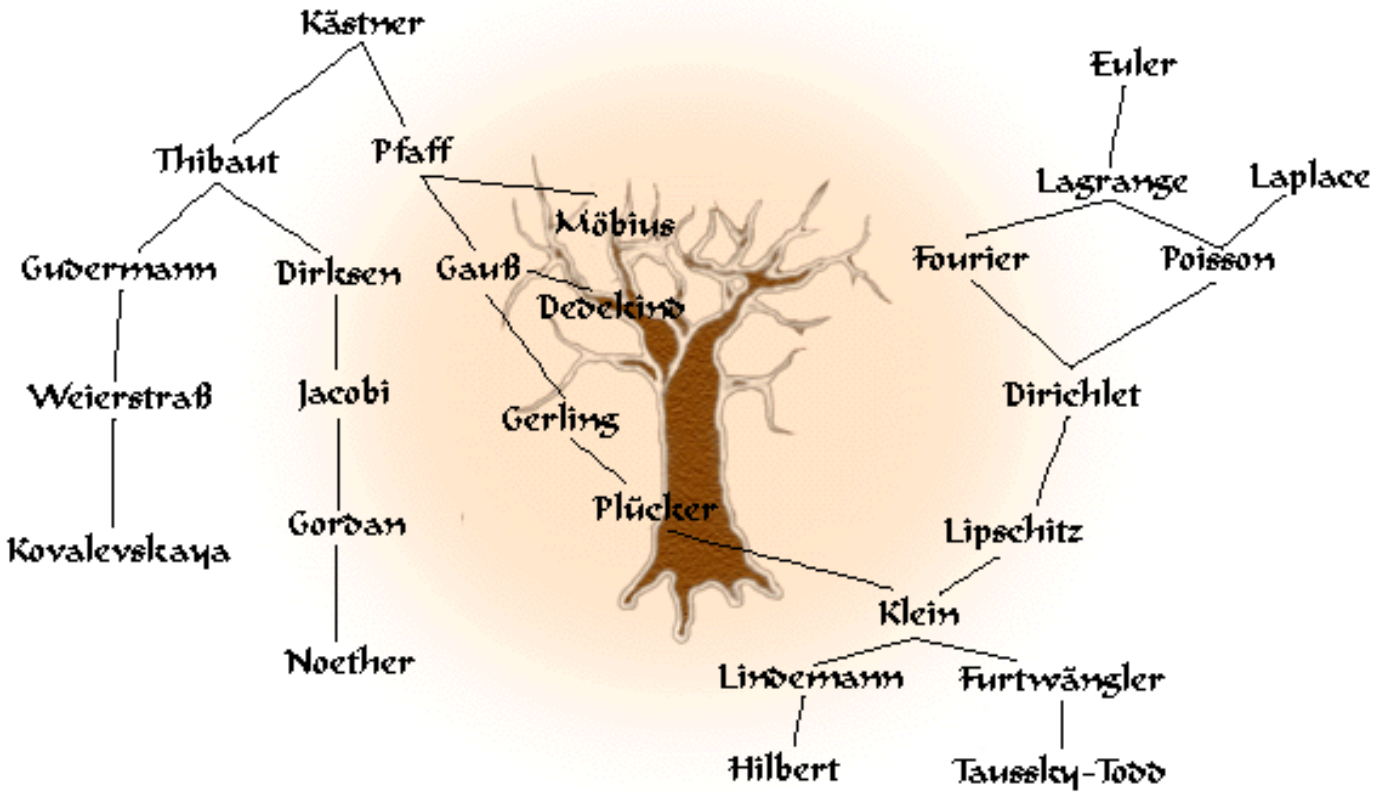
Principal Component Analysis

Principal Component Analysis (PCA)

- Given an approximate distance matrix D
- find $x = \mathrm{MDS}(D)$
- However, you want $x = P\sqrt{\Lambda}$ in K dimensions
  but $\mathsf{rank}(\Lambda) > K$
- Only keep K largest components of Λ
  zero the rest
- Get embedding in desired (lower) dimension

Example 1/3

Mathematical genealogy skeleton



Example 2/3

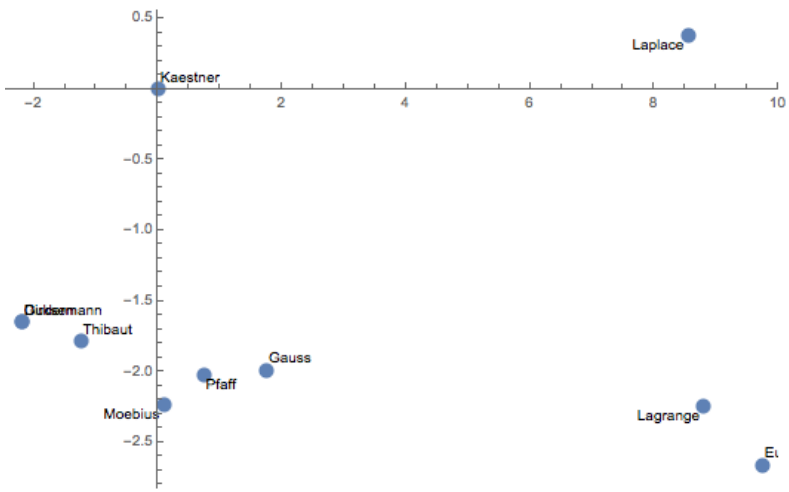
A partial view

	Euler	Thibaut	Pfaff	Lagrange	Laplace	Möbius	Gudermann	Dirksen	Gauss
Kästner	10	1	1	9	8	2	2	2	2
Euler		11	9	1	3	10	12	12	8
Thibaut			2	10	10	3	1	1	3
Pfaff				8	8	1	3	3	1
Lagrange					2	9	11	11	7
Laplace						9	11	11	7
Möbius							4	4	2
Gudermann								2	4
Dirksen									4

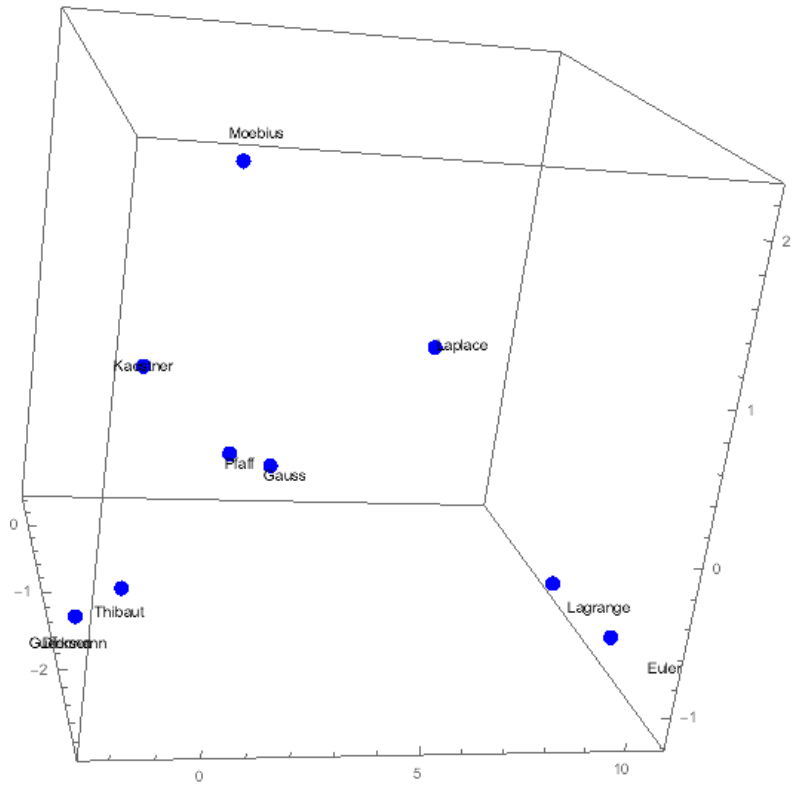
$$D = \begin{pmatrix} 0 & 10 & 1 & 1 & 9 & 8 & 2 & 2 & 2 & 2 \\ 10 & 0 & 11 & 9 & 1 & 3 & 10 & 12 & 12 & 8 \\ 1 & 11 & 0 & 2 & 10 & 10 & 3 & 1 & 1 & 3 \\ 1 & 9 & 2 & 0 & 8 & 8 & 1 & 3 & 3 & 1 \\ 9 & 1 & 10 & 8 & 0 & 2 & 9 & 11 & 11 & 7 \\ 8 & 3 & 10 & 8 & 2 & 0 & 9 & 11 & 11 & 7 \\ 2 & 10 & 3 & 1 & 9 & 9 & 0 & 4 & 4 & 2 \\ 2 & 12 & 1 & 3 & 11 & 11 & 4 & 0 & 2 & 4 \\ 2 & 12 & 1 & 3 & 11 & 11 & 4 & 2 & 0 & 4 \\ 2 & 8 & 3 & 1 & 7 & 7 & 2 & 4 & 4 & 0 \end{pmatrix}$$

Example 3/3

In 2D



In 3D

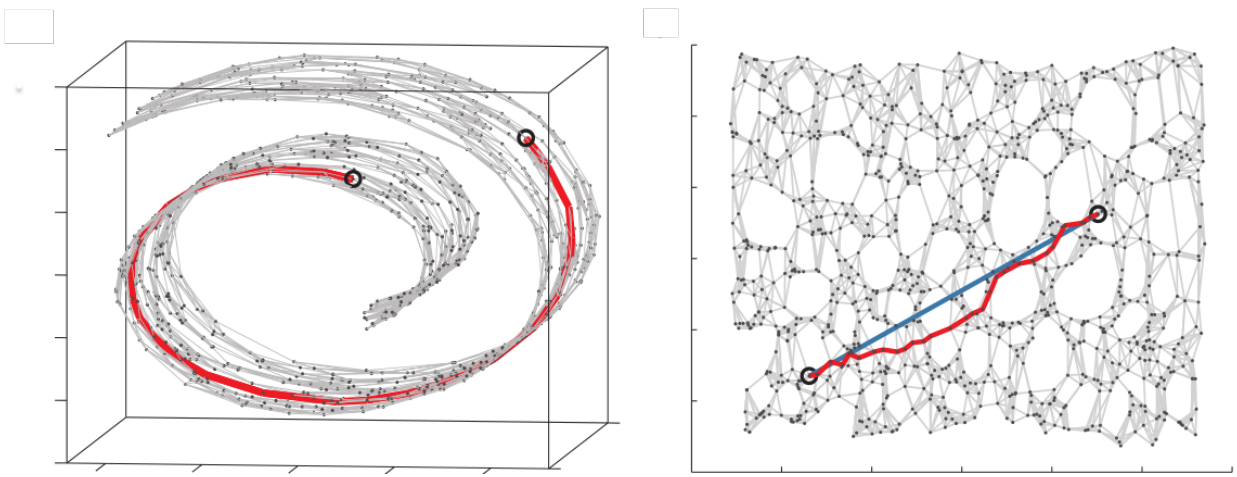


Subsection 4

Summary: Isomap

Isomap for DG

1. Let D' be the (square) weighted adjacency matrix of G
2. Complete D' to *approximate* EDM $\tilde{D}$
3. Perform PCA on $\tilde{D}$ given K dimensions
   (a) Let $\tilde{B} = -(1/2)J\tilde{D}J$, where $J = I - (1/n)\mathbf{1}\mathbf{1}^\top$
   (b) Find eigenval/vects Λ, P so $\tilde{B} = P^\top \Lambda P$
   (c) Keep $\leq K$ largest nonneg. eigenv. of Λ to get $\tilde{\Lambda}$
   (d) Let $\tilde{x} = P^\top \sqrt{\tilde{\Lambda}}$



Vary Step 2 to generate Isomap heuristics

Variants for Step 2

A. Floyd-Warshall all-shortest-paths algorithm on G (classic Isomap)

B. Find a spanning tree (SPT) of G and compute a random realization in $\bar{x} \in \mathbb{R}^K$, use its sqEDM

C. *and more…*

[Liberti *et al.*, SEA 17]

Distance geometry problem

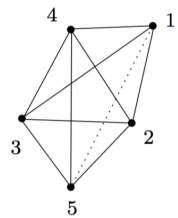
The Distance Geometry Problem (DGP)

Given $K \in \mathbb{N}$ and $G = (V, E, d)$ with $d : E \to \mathbb{R}_+$, find $x : V \to \mathbb{R}^K$ s.t.

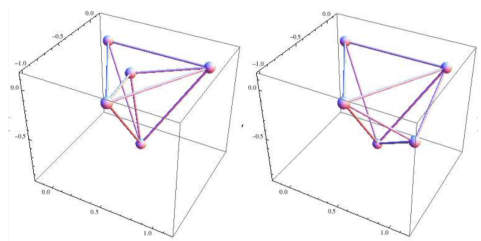
$$\forall \{i, j\} \in E \quad \|x_i - x_j\|_2^2 = d_{ij}^2$$

Defn.

x is a *realization* of G in $\mathbb{R}^K$

Given a weighted graph 

, draw it so edges are drawn as segments

with lengths = weights 

Subsection 1

Applications

Some applications

- clock synchronization ($K = 1$)
- sensor network localization ($K = 2$)
- molecular structure from distance data ($K = 3$)
- autonomous underwater vehicles ($K = 3$)
- distance matrix completion (whatever K)
- *finding graph embeddings*

Clock synchronization

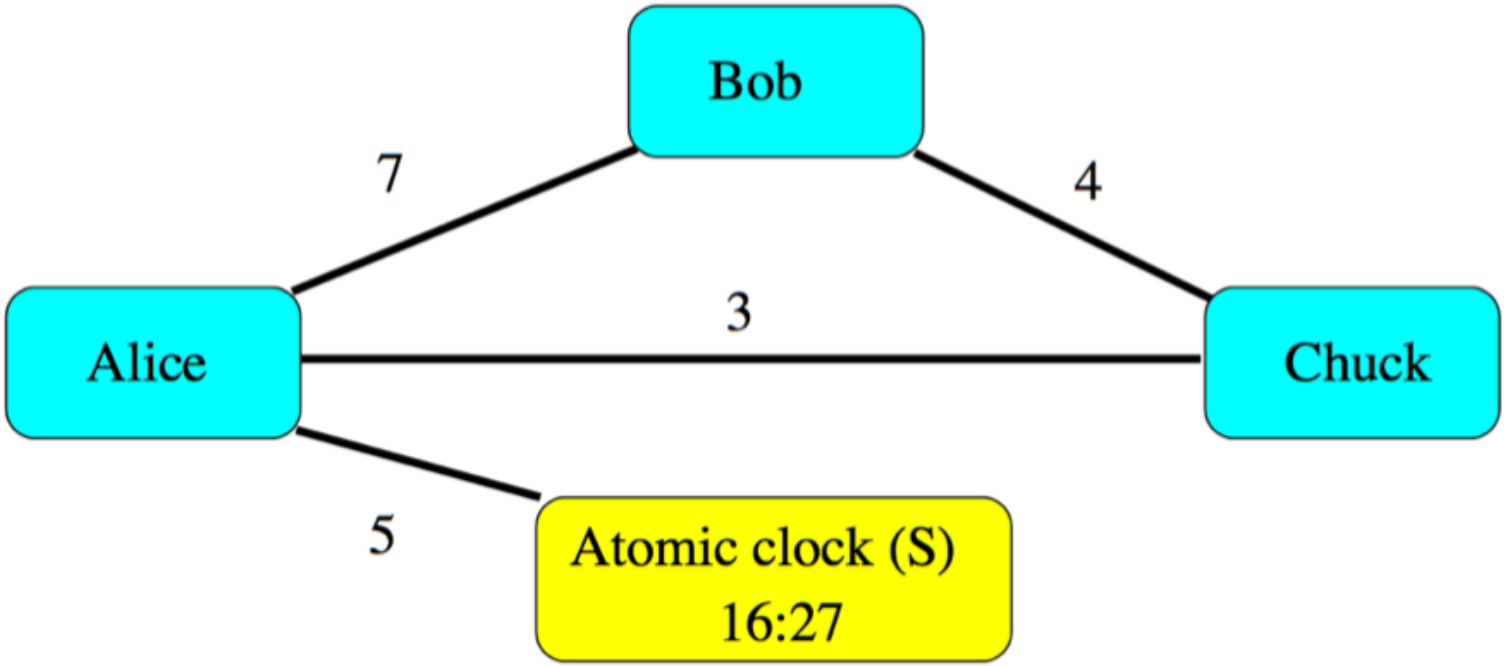
From [Singer, *Appl. Comput. Harmon. Anal.* 2011]

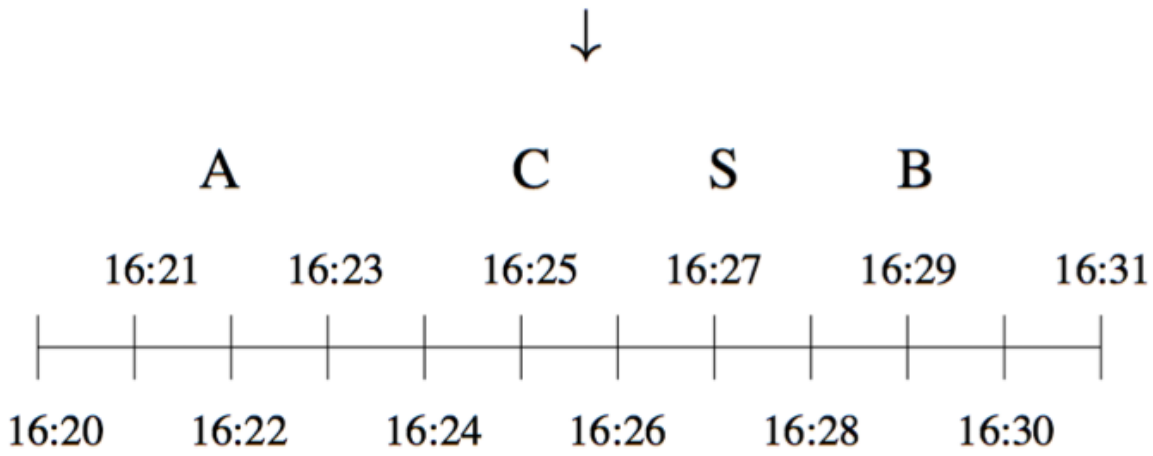
Determine a set of unknown timestamps from partial measurements of their time differences

- $K = 1$
- V: timestamps
- $\{u, v\} \in E$ if known time difference between u, v
- d: values of the time differences

Used in time synchronization of distributed networks

Clock synchronization





Sensor network localization

From [Yemini, *Proc. CDSN*, 1978]

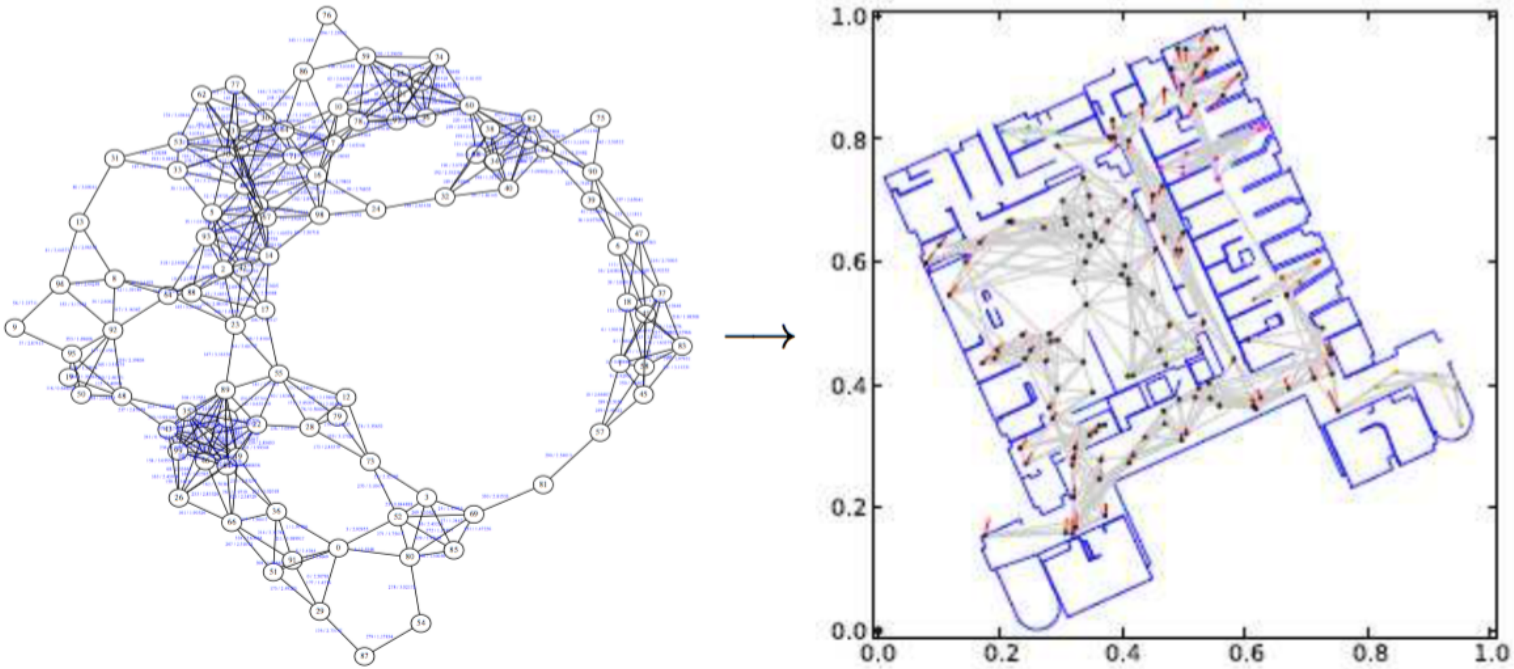
The positioning problem arises when it is necessary to locate a set of geographically distributed objects using measurements of the distances between some object pairs

- $K = 2$
- V: (mobile) sensors
- $\{u, v\} \in E$ iff distance between u, v is measured
- d: distance values

Used whenever GPS not viable (e.g. underwater)

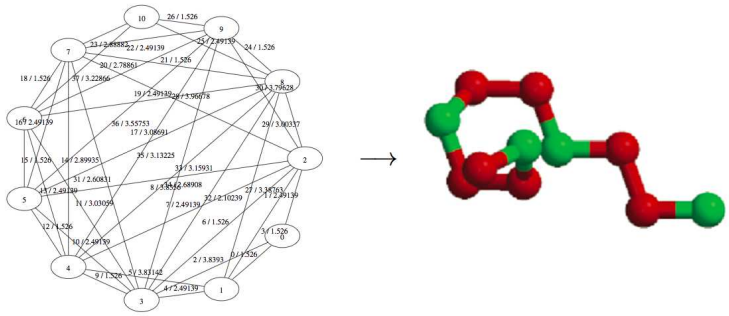
$d_{uv} \propto$ battery consumption in P2P communication betw. u, v

Sensor network localization



Molecular structure from distance data

From [Liberti et al., *SIAM Rev.*, 2014]



- $K = 3$
- V: atoms
- $\{u, v\} \in E$ iff distance between u, v is known
- d: distance values

Used whenever X-ray crystallography does not apply (e.g. liquid)
Covalent bond lengths and angles known precisely
Distances $\lessapprox 5.5$ measured approximately by NMR

Graph embeddings

- Relational knowledge best represented by graphs
- We have fast algorithms for clustering vectors
- Task: represent a graph in $\mathbb{R}^n$
- "Graph embeddings" and "distance geometry": almost synonyms
- Used in Natural Language Processing (NLP)
  obtain "word vectors" & "concept vectors"

Subsection 2

Complexity

Complexity

- DGP$_1$ with $d : E \to \mathbb{Q}_+$ is in **NP**
  - if instance YES $\exists$ realization $x \in \mathbb{R}^{n\times 1}$
  - if some component $x_i \notin \mathbb{Q}$ translate x so $x_i \in \mathbb{Q}$
  - consider some other x_j
  - let $\ell = |\text{sh. path } p : i \to j| = \sum_{\{u,v\}\in p} (-1)^{s_{uv}} d_{uv} \in \mathbb{Q}$

    for some $s_{uv} \in \{0, 1\}$
  - then $x_j = x_i \pm \ell \to x_j \in \mathbb{Q}$
  - $\Rightarrow$ verification of

$$\forall \{i, j\} \in E \quad |x_i - x_j| = d_{ij}$$

    in polytime
- DGP$_K$ may not be in **NP** for $K > 1$

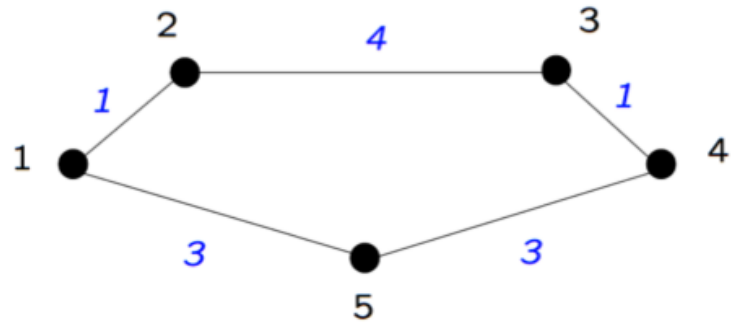
  don't know how to check $\|x_i - x_j\|_2 = d_{ij}$ in polytime for $x \notin \mathbb{Q}^{nK}$

Hardness

PARTITION *is* NP*-hard*

Given $a = (a_1, \ldots, a_n) \in \mathbb{N}^n$, $\exists\, I \subseteq \{1, \ldots, n\}$ s.t. $\sum_{i \in I} a_i = \sum_{i \notin I} a_i$?

- Reduce PARTITION to DGP_1
- $a \longrightarrow$ cycle C
  $V(C) = \{1, \ldots, n\}$, $E(C) = \{\{1, 2\}, \ldots, \{n, 1\}\}$
- For $i < n$ let $d_{i,i+1} = a_i$
  $d_{n,n+1} = d_{n1} = a_n$
- *E.g. for* $a = (1, 4, 1, 3, 3)$, *get cycle graph:*



[Saxe, 1979]

PARTITION is YES $\Rightarrow$ DGP$_1$ is YES

- Given: $I \subset \{1, \ldots, n\}$ s.t. $\sum_{i \in I} a_i = \sum_{i \notin I} a_i$
- Construct: realization x of C in $\mathbb{R}$
  1. $x_1 = 0$ `// start`
  2. induction step: suppose x_i known
     if $i \in I$
     let $x_{i+1} = x_i + d_{i,i+1}$ `// go right`
     else
     let $x_{i+1} = x_i - d_{i,i+1}$ `// go left`
- Correctness proof: by the same induction
  but careful when $i = n$: have to show $x_{n+1} = x_1$

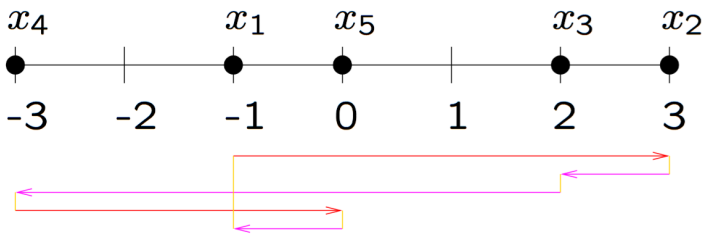
PARTITION is YES $\Rightarrow$ DGP$_1$ is YES

$$\begin{aligned}(1) = \sum_{i\in I}(x_{i+1}-x_i) &= \sum_{i\in I} d_{i,i+1} = \\ = \sum_{i\in I} a_i &= \sum_{i\notin I} a_i = \\ = \sum_{i\notin I} d_{i,i+1} &= \sum_{i\notin I}(x_i - x_{i+1}) = (2)\end{aligned}$$

$$\begin{aligned}(1) = (2) \Rightarrow \sum_{i\in I}(x_{i+1}-x_i) = \sum_{i\notin I}(x_i - x_{i+1}) \Rightarrow \sum_{i\leq n}(x_{i+1}-x_i) &= 0 \\ \Rightarrow (x_{n+1}-x_n)+(x_n-x_{n-1})+\cdots+(x_3-x_2)+(x_2-x_1) &= 0 \\ \Rightarrow x_{n+1} &= x_1\end{aligned}$$

PARTITION is NO $\Rightarrow$ DGP$_1$ is NO

- By contradiction: suppose DGP$_1$ is YES, x realization of C
- $F = \{\{u, v\} \in E(C) \mid x_u \leq x_v\}$,
  $E(C) \setminus F = \{\{u, v\} \in E(C) \mid x_u > x_v\}$
- Trace $x_1, \ldots, x_n$: follow edges in F ($\rightarrow$) and in $E(C) \setminus F$ ($\leftarrow$)



$$\sum_{\{u,v\}\in F} (x_v - x_u) = \sum_{\{u,v\}\notin F} (x_u - x_v)$$
$$\sum_{\{u,v\}\in F} |x_u - x_v| = \sum_{\{u,v\}\notin F} |x_u - x_v|$$
$$\sum_{\{u,v\}\in F} d_{uv} = \sum_{\{u,v\}\notin F} d_{uv}$$

- Let $J = \{i < n \mid \{i, i+1\} \in F\} \cup \{n \mid \{n, 1\} \in F\}$

$$\Rightarrow \quad \sum_{i\in J} a_i = \sum_{i\notin J} a_i$$

- So J solves Partition instance, contradiction
- $\Rightarrow$ DGP is NP-hard, DGP$_1$ is NP-complete

Subsection 3

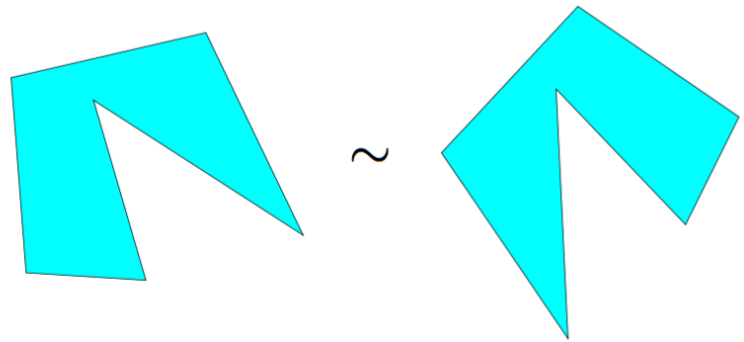
Number of solutions

Number of solutions

- (G, K): DGP instance
- $\tilde{X} \subseteq \mathbb{R}^{Kn}$: set of solutions
- *Congruence*: composition of translations, rotations, reflections
- $C =$ set of congruences in $\mathbb{R}^K$
- $x \sim y$ means $\exists \rho \in C\ (y = \rho x)$:
  distances in x are preserved in y through ρ
- $\Rightarrow$ if $|\tilde{X}| > 0$, $|\tilde{X}| = 2^{\aleph_0}$

Number of solutions modulo congruences

- Congruence is an *equivalence relation* $\sim$ on $\tilde{X}$
  (reflexive, symmetric, transitive)



- Partitions $\tilde{X}$ into *equivalence classes*
- $X = \tilde{X}/\sim$: sets of representatives of equivalence classes
- Focus on $|X|$ rather than $|\tilde{X}|$

Rigidity, flexibility and $|X|$

- infeasible $\Leftrightarrow |X| = 0$
- rigid graph $\Leftrightarrow |X| < \aleph_0$
- globally rigid graph $\Leftrightarrow |X| = 1$
- flexible graph $\Leftrightarrow |X| = 2^{\aleph_0}$
- $|X| = \aleph_0$: impossible by Milnor's theorem

Milnor's theorem implies $|X| \neq \aleph_0$

- System S of polynomial equations of degree 2

$$\forall i \leq m \quad p_i(x_1, \ldots, x_{nK}) = 0$$

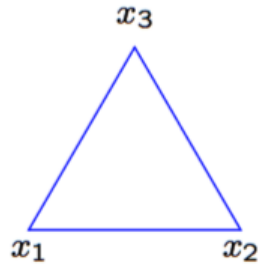
- Let X be the set of $x \in \mathbb{R}^{nK}$ satisfying S
- Number of connected components of X is $O(3^{nK})$
  [Milnor 1964]
- Assume $|X|$ is countable; then G cannot be flexible
  $\Rightarrow$ *each incongruent rlz is in a separate component*
  $\Rightarrow$ by Milnor's theorem, there's finitely many of them

Examples

$V^1 = \{1, 2, 3\}$

$E^1 = \{\{u, v\} \mid u < v\}$

$d^1 = 1$



ρ congruence in $\mathbb{R}^2$

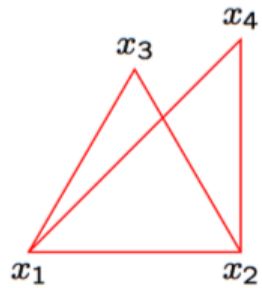
$\Rightarrow \rho x$ valid realization

$|X| = 1$

$V^2 = V^1 \cup \{4\}$

$E^2 = E^1 \cup \{\{1, 4\}, \{2, 4\}\}$

$d^2 = 1 \wedge d_{14} = \sqrt{2}$



ρ reflects x_4 wrt $\overline{x_1, x_2}$

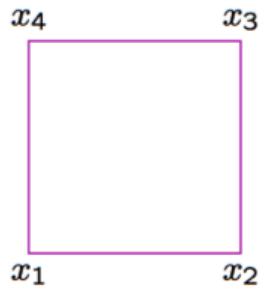
$\Rightarrow \rho x$ valid realization

$|X| = 2$ ()

$V^3 = V^2$

$E^3 = \{\{u, u+1\} | u \leq 3\} \cup \{1, 4\}$

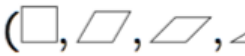
$d^1 = 1$



ρ rotates $\overline{x_2x_3}$, $\overline{x_1x_4}$ by θ

$\Rightarrow \rho x$ valid realization

$|X|$ is uncountable

( , $\ldots$)

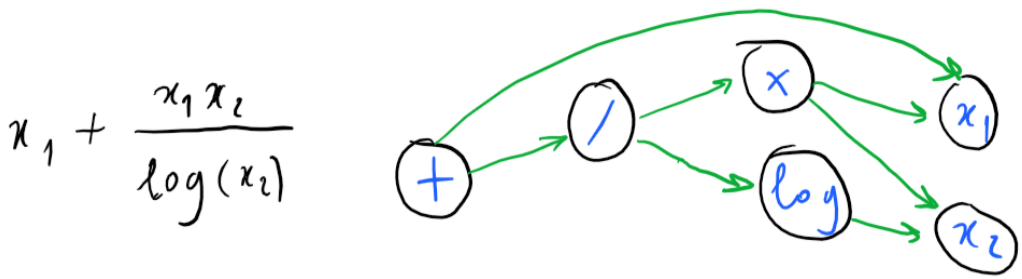
Mathematical Programming formulations

Short introduction to MP 1/3

- Formal language to describe optimization problems
- Sentences are called *formulations*

$$\left.\begin{array}{rlcl} \min & f(x) & & \\ \forall i \leq m & g_i(x) & \leq & 0 \\ \forall j \in Z & x_j & \in & \mathbb{Z} \end{array}\right\} \quad [P]$$

- f, g: represented by expression language



- Formulation entities:

set	param	dec. var	obj	constr
Z	m	x	$f(x)$	$g_i(x) \leq 0$ $x_j \in \mathbb{Z}$

Short introduction to MP 2/3

- Syntax highlights:
  - *no explicit open sets:* use $\leq, \geq, =$ not $<, >, \neq$
  - *no vars in quantifiers* (e.g. $\sum_{\substack{i \in I \\ x_i = 1}} y_i$)
- Semantics:
  assignment of values to decision variables
  - carried out by solver:
    solution alg. for MP subclass defined by given math. properties
  - requires formulation taxonomy w.r.t. solvers
    LP, SOCP, SDP, QP, QCP, QCQP, NLP,
    MILP, MIQP, MIQCP, MIQCQP, MINLP
    c(MI)QP, c(MI)QCP, c(MI)QCQP, c(MI)NLP,
    BLP, BQP, MOP, BLevP, SIP, ...
- MP shifts focus from *algorithmics* to *modelling*

Short introduction to MP 3/3

- Solvers can be:
  global or local, exact, approximate or heuristic, (worst-case) polytime or exponential, fast or slow
- Examples:
  - LP: simplex alg., interior point method (IPM); CPLEX, GuRoBi, XPressMP, GLPK, CLP; *global, exact, fast, IPM is polytime*
  - SOCP, SDP: IPM; Mosek, SCS; *global, ϵ-approx., fast, polytime*
  - QP, QCP, QCQP, NLP: Succ. Quadr. Prog. (SQP), IPM; Snopt, IPOPT; *local, heuristic (ϵ-approx. if cvx), fast*
  - BLP, MILP: Cutting plane (CP), Branch-and-Bound (BB); CPLEX, XPressMP, GLPK; *global, exact, slow, exponential*
  - cMIQP, cMIQCP, cMIQCQP, cMINLP: BB, Outer Approx. (OA), CP; GuRoBi, BonMin, alphaECP; *global, ϵ-approx., slow, exponential*
  - MIQP, MIQCP, MIQCQP, MINLP: spatial BB (sBB); Baron, GuRoBi, Couenne, Antigone; *global, ϵ-approx., slow, exponential*

Subsection 1

Formulations in position variables

QCP and QCQP

Original DGP system is a QCP:

$$\forall \{u, v\} \in E \quad \|x_u - x_v\|^2 = d_{uv}^2 \tag{4}$$

Computationally: useless

Reformulate using slack variables, get QCQP:

$$\min_{x,s} \left\{ \sum_{\{u,v\} \in E} {s_{uv}}^2 \;\middle|\; \forall \{u, v\} \in E \; \|x_u - x_v\|^2 = d_{uv}^2 + s_{uv} \right\} \tag{5}$$

Unconstrained NLP

$$\min_{x \in \mathbb{R}^{nK}} \sum_{\{u,v\} \in E} \left(\|x_u - x_v\|^2 - d_{uv}^2 \right)^2 \tag{6}$$

Sum of squares, but nonconvex in x (quartic polynomial)

Globally optimal obj. fun. value of (6) is 0 iff x solves (4)

Computational experiments in [Liberti et al., 2006]:

- Solvers from 15 years ago
- randomly generated protein data: ≤ 50 atoms
- cubic crystallographic grids: ≤ 64 atoms

"Push-and-pull" QCQP

- $\min_{x \in \mathbb{R}^{nK}} \sum_{\{u,v\} \in E} |\|x_u - x_v\|^2 - d_{uv}^2|$ exactly reformulates (4)
- Relax objective f to concave part, remove constant term, rewrite $\min -f$ as $\max f$
- Reformulate convex part of obj. fun. to convex constraints
- *Exact reformulation*

$$\left.\begin{array}{rl} \max_x & \sum_{\{u,v\} \in E} \|x_u - x_v\|^2 \\ \forall \{u,v\} \in E & \|x_u - x_v\|_2^2 \leq d_{uv}^2 \end{array}\right\} \quad (7)$$

Objective pulls points apart, constraints push them together

Thm.

At a glob. opt. x^* of a YES instance, all constraints of (7) are active

Multiplicative Weights Update (MWU) alg.

$$\left.\begin{array}{rl} \max_x & \sum\limits_{\{u,v\}\in E} \|x_u - x_v\|^2 \\ \forall \{u,v\} \in E & \|x_u - x_v\|_2^2 \le d_{uv}^2 \end{array}\right\} \quad \text{[QCQP]}$$

1. $\|x_u - x_v\|_2^2 = \langle (x_u - x_v), (x_u - x_v)\rangle = \langle w_{uv}, (x_u - x_v)\rangle$
   fix $w \in \mathbb{R}^K$, get cNLP
   $\min\{ \sum\limits_{\{u,v\}\in E} \langle w_{uv}, (x_u - x_v)\rangle \mid \forall \{u,v\} \in E \ \|x_u - x_v\|_2^2 \le d_{uv}^2\}$
   solve efficiently, get x' with EDM D'
2. $\forall \{u,v\} \in E$ define $\psi_{uv} \triangleq \frac{|D'_{uv} - d_{uv}|}{\max(D'_{uv}, d_{uv})}$
   relative error between D' and d
3. $\forall \{u,v\} \in E$ update $w_{uv} \leftarrow w_{uv}(1 - \eta\, \psi_{uv})$
   where η is a constant in $(0, 1)$
4. repeat for T iterations, record best solution

MWU relative guarantee

- index quantities by iteration t: $(x')^t, w^t, \psi^t$
- get distribution $\rho^t_{uv} = w^t_{uv}/\langle \mathbf{1}, w^t \rangle$
- $\sum_{\{u,v\}\in E} \psi^t_{uv}\rho^t_{uv} = \langle \psi^t, \rho^t \rangle$
  weighted relative error between D' and d at itn t
- can prove that

$$\min_{t\le T}\langle \psi^t, \rho^t \rangle \le \frac{1}{T}\left(2\ln m + \frac{3}{2}\min_{\{u,v\}\in E}\sum_{t\le T}\psi^t_{uv}\right)$$

- relative UB to error of MWU output in terms of best cumulative distance error

[D'Ambrosio *et al.*, DCG 17, Mencarelli *et al.* EJCO 17]

Subsection 2

Formulations in matrix variables

Linearization

Replace nonlinear terms with additional variables:

Back to original system $\forall \{i,j\} \in E \ \|x_i - x_j\|_2^2 = d_{ij}^2$

$$\Rightarrow \quad \forall \{i,j\} \in E \quad \|x_i\|_2^2 + \|x_j\|_2^2 - 2x_i \cdot x_j = d_{ij}^2$$

$$\Rightarrow \left\{ \begin{array}{lrcl} \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} & = & d_{ij}^2 \\ & X & = & x\,x^\top \end{array} \right.$$

Note: $X = xx^\top \Leftrightarrow [\forall i,j \leq n \ X_{ij} = x_i x_j]$

$$\Leftrightarrow \begin{pmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ X_{12} & X_{22} & \cdots & X_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{1n} & X_{2n} & \cdots & X_{nn} \end{pmatrix} = \begin{pmatrix} x_1^2 & x_1x_2 & \cdots & x_1x_n \\ x_1x_2 & x_2^2 & \cdots & x_2x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_nx_1 & x_nx_2 & \cdots & x_n^2 \end{pmatrix}$$

implies that X must have rank ≤ 1

Relaxation

Replace nonconvex set with a convex relaxation:

$$
\begin{aligned}
& & X &= x\,x^\top & \\
& \Rightarrow & X - x\,x^\top &= 0 & \text{all eigs.} = 0 \\
\text{(relax)} \quad & \Rightarrow & X - x\,x^\top &\succeq 0 & \text{all eigs.} \geq 0 \\
\Rightarrow \mathsf{Schur}(X, x) \triangleq & \begin{pmatrix} I_K & x^\top \\ x & X \end{pmatrix} &\succeq 0 &&
\end{aligned}
$$

- If x does not appear elsewhere in formulation can eliminate it (e.g. choose $x = 0$)
- $\Rightarrow$ Replace $\mathsf{Schur}(X, x) \succeq 0$ by $X \succeq 0$

SDP relaxation

$$\begin{aligned} & \min F \bullet X \\ \forall \{i,j\} \in E \quad X_{ii} + X_{jj} - 2X_{ij} &= d_{ij}^2 \\ X &\succeq 0 \end{aligned}$$

How do we choose F?

$F \bullet X = \mathsf{tr}(F^\top X)$

Some possible objective functions

- For protein conformation:

$$\min \sum_{\{i,j\} \in E} (X_{ii} + X_{jj} - 2X_{ij})$$

with $=$ changed to $\geq$ in constraints *(or* $\max$ *and* $\leq$*)*
SDP relaxation of "push-and-pull" QCQP

- [Ye, 2003], application to wireless sensors localization

$$\min \mathsf{tr}(X)$$

$\mathsf{tr}(X) = \mathsf{tr}(P^{-1}\Lambda P) = \mathsf{tr}(P^{-1}P\Lambda) = \mathsf{tr}(\Lambda) = \sum_i \lambda_i$
$\Rightarrow$ *hope to minimize rank*

- How about "just random"?

[Barvinok, DCG 1995]

How do you choose?

for want of some better criterion…

TEST!

- Download protein files from Protein Data Bank (PDB)

  they contain atom realizations

- Mimick a Nuclear Magnetic Resonance experiment

  Keep only pairwise distances < 5.5

- Try and reconstruct the protein shape from those weighted graphs

- Quality evaluation of results:
  - $\mathrm{LDE}(x) = \max_{\{i,j\} \in E} | \, \|x_i - x_j\| - d_{ij} \, |$
  - $\mathrm{MDE}(x) = \frac{1}{|E|} \sum_{\{i,j\} \in E} | \, \|x_i - x_j\| - d_{ij} \, |$

Empirical choice

- *Ye* very fast but often imprecise
- *Random* good but nondeterministic
- *Push-and-Pull*: can relax $X_{ii} + X_{jj} - 2X_{ij} = d_{ij}^2$ to $X_{ii} + X_{jj} - 2X_{ij} \geq d_{ij}^2$
  easier to satisfy feasibility, useful later on
- Heuristic: add $+\eta \mathsf{tr}(X)$ to objective, with $\eta \ll 1$
  might help minimize solution rank
- $\min \sum_{\{i,j\} \in E} (X_{ii} + X_{jj} - 2X_{ij}) + \eta \mathsf{tr}(X)$

Subsection 3

Diagonal dominance

When SDP solvers hit their size limit

- SDP solver: technological bottleneck
- Can we use an LP solver instead?
- Diagonally Dominant (DD) matrices are PSD
- Not *vice versa*: inner approximate PSD cone $Y \succeq 0$
- *Idea by A.A. Ahmadi [Ahmadi & Hall 2015]*

Diagonally dominant matrices

$n \times n$ symmetric matrix X is DD if

$$\forall i \leq n \quad X_{ii} \geq \sum_{j \neq i} |X_{ij}|.$$

E.g.
$$\begin{pmatrix} 1 & 0.1 & -0.2 & 0 & 0.04 & 0 \\ 0.1 & 1 & -0.05 & 0.1 & 0 & 0 \\ -0.2 & -0.05 & 1 & 0.1 & 0.01 & 0 \\ 0 & 0.1 & 0.1 & 1 & 0.2 & 0.3 \\ 0.04 & 0 & 0.01 & 0.2 & 1 & -0.3 \\ 0 & 0 & 0 & 0.3 & -0.3 & 1 \end{pmatrix}$$

Gershgorin's circle theorem

- Let A be symmetric $n \times n$
- $\forall i \leq n$ let $R_i = \sum_{j \neq i} |A_{ij}|$ and $I_i = [A_{ii} - R_i, A_{ii} + R_i]$
- Then $\forall \lambda$ eigenvalue of $A \quad \exists i \leq n$ s.t. $\lambda \in I_i$

Proof

- Let λ be an eigenvalue of A with eigenvector x
- Normalize x s.t. $\exists i \leq n \; x_i = 1$ and $\forall j \neq i \; |x_j| \leq 1$
  let $i = \text{argmax}_j |x_j|$, *divide* x *by* $\text{sgn}(x_i)|x_i|$
- $Ax = \lambda x \Rightarrow \sum_{j \neq i} A_{ij} x_j + A_{ii} x_i = \sum_{j \neq i} A_{ij} x_j + A_{ii} = \lambda x_i = \lambda$
- Hence $\sum_{j \neq i} A_{ij} x_j = \lambda - A_{ii}$
- Triangle inequality and $|x_j| \leq 1$ for all $j \neq i \Rightarrow$
  $|\lambda - A_{ii}| = |\sum_{j \neq i} A_{ij} x_j| \leq \sum_{j \neq i} |A_{ij}| \, |x_j| \leq \sum_{j \neq i} |A_{ij}| = R_i$
  hence $\lambda \in I_i$

DD $\Rightarrow$ PSD

- Assume A is DD, λ an eigenvalue of A
- $\Rightarrow \forall i \leq n \quad A_{ii} \geq \sum_{j \neq i} |A_{ij}| = R_i$
- $\Rightarrow \forall i \leq n \quad A_{ii} - R_i \geq 0$
- By Gershgorin's circle theorem $\lambda \geq 0$
- $\Rightarrow A$ is PSD

DD Linearization

$$\forall i \leq n \quad X_{ii} \geq \sum_{j \neq i} |X_{ij}| \qquad (*)$$

- linearize $|\cdot|$ by additional matrix var T
  $\Rightarrow$ *write* $|X|$ *as* T
- $\Rightarrow (*)$ becomes

$$X_{ii} \geq \sum_{j \neq i} T_{ij}$$

- add "sandwich" constraints $-T \leq X \leq T$
- Can easily prove $(*)$ in case $X \geq 0$ or $X \leq 0$:

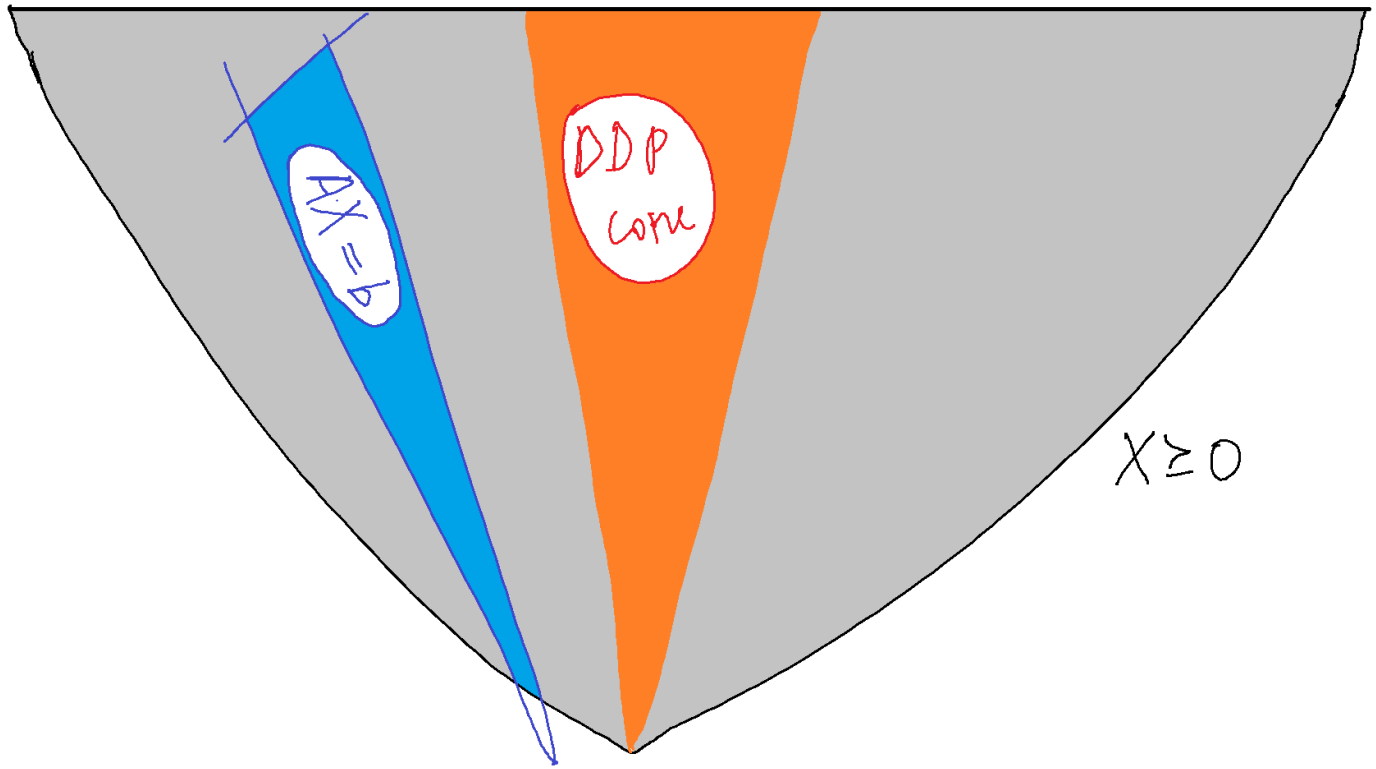
$$\begin{aligned} X_{ii} &\geq \sum_{j \neq i} T_{ij} \geq \sum_{j \neq i} X_{ij} \\ X_{ii} &\geq \sum_{j \neq i} T_{ij} \geq \sum_{j \neq i} -X_{ij} \end{aligned}$$

DD Programming (DDP)

$$\left.\begin{array}{rrcl} \forall\{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} & = & d_{ij}^2 \\ & X & \text{is} & \text{DD} \end{array}\right\}$$

$$\Rightarrow \left\{\begin{array}{rrcl} \forall\{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} & = & d_{ij}^2 \\ \forall i \leq n & \sum_{\substack{j \leq n \\ j \neq i}} T_{ij} & \leq & X_{ii} \\ & -T \leq X & \leq & T \end{array}\right.$$

The issue with inner approximations



DDP could be infeasible!

Exploit push-and-pull

- Enlarge the feasible region
- From

$$\forall \{i,j\} \in E \quad X_{ii} + X_{jj} - 2X_{ij} = d_{ij}^2$$

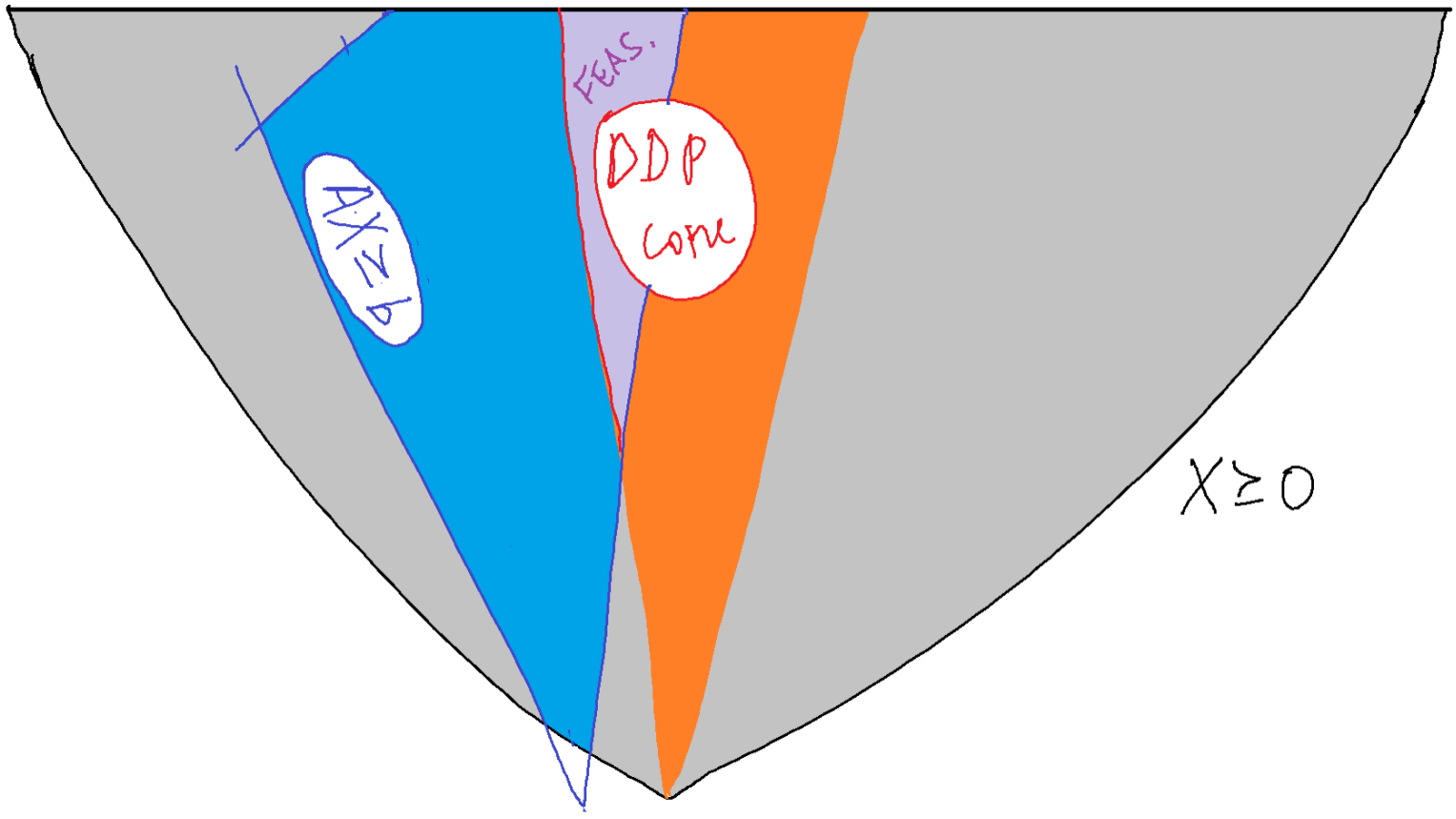
- Relax to "pull" constraints

$$\forall \{i,j\} \in E \quad X_{ii} + X_{jj} - 2X_{ij} \geq d_{ij}^2$$

- Then use "push" objective

$$\min \sum_{ij \in E} X_{ii} + X_{jj} - 2X_{ij}$$

Hope to achieve LP feasibility



DDP formulation for the DGP

$$
\left.
\begin{array}{rrrcl}
\min & \sum_{\{i,j\}\in E} (X_{ii} + X_{jj} - 2X_{ij}) & & & \\
\forall \{i,j\} \in E & & X_{ii} + X_{jj} - 2X_{ij} & \geq & d_{ij}^2 \\
\forall i \leq n & & \sum_{\substack{j \leq n \\ j \neq i}} T_{ij} & \leq & X_{ii} \\
& & -T \leq X & \leq & T \\
& & T & \geq & 0
\end{array}
\right\}
$$

Subsection 4

Dual DD

Cones

- Set C is a *cone* if:

$$\forall A, B \in C,\ \alpha, \beta \geq 0 \quad \alpha A + \beta B \in C$$

- If C is a cone, the *dual cone* is

$$C^* = \{y \mid \forall x \in C\ \langle x, y \rangle \geq 0\}$$

vectors making acute angles with all elements of C

- If $C \subset \mathbb{S}_n$ (set $n \times n$ symmetric matrices)

$$C^* = \{Y \mid \forall X \in C\ (Y \bullet X \geq 0)\}$$

- A $n \times n$ matrix cone C is *finitely generated* by $\mathcal{X} \subset \mathbb{R}^n$ if

$$\mathcal{X} = \{x_1, \ldots, x_p\} \wedge \quad \forall X \in C\ \exists \delta \in \mathbb{R}^p_+\ X = \sum_{\ell \leq p} \delta_\ell\, x_\ell x_\ell{}^\top$$

Representations of $\mathbb{DD}$

- Consider $E_{ii}, E_{ij}^{+}, E_{ij}^{-}$ in $\mathbb{S}_n$
  Define $\mathcal{E}_0 = \{E_{ii} \mid i \leq n\}$, $\mathcal{E}_1 = \{E_{ij}^{\pm} \mid i < j\}$, $\mathcal{E} = \mathcal{E}_0 \cup \mathcal{E}_1$
- $E_{ii} = \mathsf{diag}(0, \ldots, 0, 1_i, 0, \ldots, 0)$
- E_{ij}^{+} has minor $\begin{pmatrix} 1_{ii} & 1_{ij} \\ 1_{ji} & 1_{jj} \end{pmatrix}$, 0 elsewhere
- E_{ij}^{-} has minor $\begin{pmatrix} 1_{ii} & -1_{ij} \\ -1_{ji} & 1_{jj} \end{pmatrix}$, 0 elsewhere

- Thm. $\mathbb{DD}$ = cone generated by $\mathcal{E}$ [Barker & Carlson 1975]
  Pf. *Rays in $\mathcal{E}$ are extreme, all DD matrices generated by $\mathcal{E}$*
- Cor. $\mathbb{DD}$ finitely gen. by
  $\mathcal{X}_{\mathbb{DD}} = \{e_i \mid i \leq n\} \cup \{(e_i \pm e_j) \mid i < j \leq n\}$
  Pf. *Verify $E_{ii} = e_i e_i^{\top}$, $E_{ij}^{\pm} = (e_i \pm e_j)(e_i \pm e_j)^{\top}$, where e_i is the i-th std basis element of $\mathbb{R}^n$*

Finitely generated dual cone representation

Thm. If C finitely gen. by $\mathcal{X}$, then

$$C^* = \{Y \in \mathbb{S}^n \mid \forall x \in \mathcal{X}\ (Y \bullet xx^\top \geq 0)\}$$

recall $C^* \triangleq \{Y \in \mathbb{S}^n \mid \forall X \in C \quad Y \bullet X \geq 0\}$

- ($\supseteq$) Let Y s.t. $\forall x \in \mathcal{X}\ (Y \bullet xx^\top \geq 0)$
  - $\forall X \in C, X = \sum_{x \in \mathcal{X}} \delta_x xx^\top$ (by fin. gen.)
  - hence $Y \bullet X = \sum_x \delta_x Y \bullet xx^\top \geq 0$ (by defn. of Y)
  - whence $Y \in C^*$ (by defn. of C^*)
- ($\subseteq$) Suppose $Z \in C^* \setminus \{Y \mid \forall x \in \mathcal{X}\ (Y \bullet xx^\top \geq 0)\}$
  - then $\exists \mathcal{X}' \subset \mathcal{X}$ s.t. $\forall x \in \mathcal{X}'\ (Z \bullet xx^\top < 0)$
  - consider any $Y = \sum_{x \in \mathcal{X}'} \delta_x xx^\top \in C$ with $\delta \geq 0$
  - then $Z \bullet Y = \sum_{x \in \mathcal{X}'} \delta_x Z \bullet xx^\top < 0$ so $Z \notin C^*$
  - contradiction $\Rightarrow C^* = \{Y \mid \forall x \in \mathcal{X}\ (Y \bullet xx^\top \geq 0)\}$

Dual cone constraints

- Remark: $X \bullet vv^\top = v^\top X v$
- Use finitely generated dual cone theorem
- Decision variable matrix X
- Constraints:

$$\forall v \in \mathcal{X} \quad v^\top X v \geq 0$$

- Cor. $\mathbb{PSD} \subset \mathbb{DD}^*$

  Pf. $X \in \mathbb{PSD}$ iff $\forall v \in \mathbb{R}^n \; vXv \geq 0$, so certainly valid $\forall v \in \mathcal{X}$
- If $|\mathcal{X}|$ polysized, get compact formulation

  otherwise use column generation
- $|\mathcal{X}_{\mathbb{DD}}| = |\mathcal{E}| = O(n^2)$

Dual cone DDP formulation for DGP

$$\left.\begin{array}{rrcl} & \min \sum_{\{i,j\}\in E} (X_{ii} + X_{jj} - 2X_{ij}) & & \\ \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} & = & d_{ij}^2 \\ \forall v \in \mathcal{X}_{\mathbb{DD}} & v^\top X v & \geq & 0 \end{array}\right\}$$

- $v^\top X v \geq 0$ for $v \in \mathcal{X}_{\mathbb{DD}}$ equivalent to:

$$\begin{array}{rrcl} \forall i \leq n & X_{ii} & \geq & 0 \\ \forall \{i,j\} \notin E & X_{ii} + X_{jj} - 2X_{ij} & \geq & 0 \\ \forall i < j & X_{ii} + X_{jj} + 2X_{ij} & \geq & 0 \end{array}$$

Note we went back to equality "pull" constraints

Quantifier $\forall \{i,j\} \notin E$ should be $\forall i < j$ but we already have those constraints $\forall \{i,j\} \in E$

Properties of the Dual DDP formulation

- SDP relaxes original problem
- DualDDP relaxes SDP
  hence also relaxes original problem
- Yields extremely tight obj fun bounds w.r.t. SDP
- Solutions may have large negative rank

Subsection 5

Dimensional reduction

Retrieving realizations in $\mathbb{R}^K$

- SDP/DDP yield $n \times n$ PSD matrix X^*
- We need $n \times K$ realization matrix x^*
- Recall *PSD* $\Leftrightarrow$ *Gram*
- Apply PCA to X^*, keep K largest comps, get x'
- This yields solutions with errors
- Refinement:
  Use x' as starting point for local NLP solver on QCQP

Subsection 6

Barvinok's Naive Algorithm

Concentration of measure

From [Barvinok, 1997]

> *The value of a "well behaved" function at a random point of a "big" probability space X is "very close" to the mean value of the function.*

and

> *In a sense, measure concentration can be considered as an extension of the law of large numbers.*

Concentration of measure

Given Lipschitz function $f : X \to \mathbb{R}$ s.t.

$$\forall x, y \in X \quad |f(x) - f(y)| \leq L\|x - y\|_2$$

for some $L \geq 0$, there is *concentration of measure* if $\exists$ constants c, C s.t.

$$\forall \varepsilon > 0 \quad \mathsf{P}_x(|f(x) - \mathsf{E}(f)| > \varepsilon) \leq c\, e^{-C\varepsilon^2/L^2}$$

where $\mathsf{E}(\cdot)$ is w.r.t. given Borel measure μ over X

$\equiv$ *"discrepancy from mean is unlikely"*

Barvinok's theorem

Consider:

- for each $k \leq m$, manifolds $\mathcal{X}_k = \{x \in \mathbb{R}^n \mid x^\top Q^k x = a_k\}$
  where $m \leq \mathsf{poly}(n)$
- feasibility problem $F \equiv \left[\bigcap_{k \leq m} \mathcal{X}_k \stackrel{?}{\neq} \varnothing\right]$
- SDP relaxation $\forall k \leq m \ (Q^k \bullet X = a_k) \wedge X \succeq 0$ with soln. $\bar{X}$
- Algorithm: $T \leftarrow \mathsf{factor}(\bar{X})$; $y \sim \mathsf{N}^n(0,1)$; $x' \leftarrow Ty$

Then:

- $\exists c > 0, n_0 \in \mathbb{N}$ such that $\forall n \geq n_0$

$$\mathsf{Prob}\left(\forall k \leq m \quad \mathsf{dist}(x', \mathcal{X}_k) \leq c\sqrt{\|\bar{X}\|_2 \ln n}\right) \geq 0.9.$$

Algorithmic application

- x' is "close" to each $\mathcal{X}_k$: *try local descent from* x'
- $\Rightarrow$ Feasible QP solution from an SDP relaxation

Elements of Barvinok's formula

$$\mathsf{Prob}\left(\forall k \le m \quad \mathsf{dist}(x', \mathcal{X}_k) \le c\sqrt{\|\bar{X}\|_2 \ln n}\right) \ge 0.9.$$

- $\sqrt{\|\bar{X}\|_2}$ arises from T (a factor of $\bar{X}$)
- $\sqrt{\ln n}$ arises from concentration of measure
- 0.9 follows by adjusting parameter values in "union bound"

Application to the DGP

- $\forall \{i,j\} \in E \quad \mathcal{X}_{ij} = \{x \mid \|x_i - x_j\|_2^2 = d_{ij}^2\}$
- DGP can be written as $\bigcap_{\{i,j\}\in E} \mathcal{X}_{ij} \stackrel{?}{\neq} \varnothing$
- SDP relaxation $X_{ii} + X_{jj} - 2X_{ij} = d_{ij}^2 \wedge X \succeq 0$ with soln. $\bar{X}$

- Difference with Barvinok: $x \in \mathbb{R}^{Kn}$, $\mathrm{rk}(\bar{X}) \leq K$
- IDEA: sample $y \sim \mathsf{N}^{nK}(0, \frac{1}{\sqrt{K}})$
- Thm. *Barvinok's theorem works in rank K*

Proof structure

- Show that, on average, $\forall k \leq m\ \mathsf{tr}((Ty)^\top Q^k(Ty)) = Q^K \bullet \bar{X} = a_k$
  - compute multivariate integrals
  - bilinear terms disappear because y normally distributed
  - decompose multivariate int. to a sum of univariate int.
- Exploit concentration of measure to show errors happen rarely
  - a couple of technical lemmata yielding bounds
  - $\Rightarrow$ bound Gaussian measure μ of ε-neighbourhoods of

$$A_i^- = \{y \in \mathbb{R}^{n\times K} \mid \mathcal{Q}^i(Ty) \leq Q^i \bullet \bar{X}\}$$
$$A_i^+ = \{y \in \mathbb{R}^{n\times K} \mid \mathcal{Q}^i(Ty) \geq Q^i \bullet \bar{X}\}$$
$$A_i = \{y \in \mathbb{R}^{n\times K} \mid \mathcal{Q}^i(Ty) = Q^i \bullet \bar{X}\}.$$

  - use "union bound" for measure of $A_i^-(\varepsilon) \cap A_i^+(\varepsilon)$
  - show $A_i^-(\varepsilon) \cap A_i^+(\varepsilon) = A_i(\varepsilon)$
  - use "union bound" to measure intersections of $A_i(\varepsilon)$
  - appropriate values for some parameters $\Rightarrow$ result

The heuristic

1. Solve SDP relaxation of DGP, get soln. $\bar{X}$
   use DDP+LP if SDP+IPM too slow

2. a. $T = \mathsf{factor}(\bar{X})$
   b. $y \sim \mathsf{N}^{nK}(0, \frac{1}{\sqrt{K}})$
   c. $x' = Ty$

3. Use x' as starting point for a local NLP solver on formulation

$$\min_x \sum_{\{i,j\} \in E} \left(\|x_i - x_j\|^2 - d_{ij}^2 \right)^2$$

and return improved solution x

MP formulations for the DGP: Summary

- *Try nonconvex formulations in position vars*
- Quadratic nonconvex too difficult?
- *Solve SDP relaxation*
- SDP relaxation too large?
- *Solve DDP approximation*
- Get $n \times n$ matrix solution, need $K \times n$
- *Use PCA or Barvinok's alg. and refine w/local NLP*

High-dimensional weirdness

Subsection 1

Random projections

The gist

- Let A be a $m \times n$ data matrix (columns in $\mathbb{R}^m$, $m \gg 1$)
- T short & fat, normally sampled componentwise

$$\underbrace{\begin{pmatrix} \vdots & \vdots \\ \vdots & \vdots \end{pmatrix}}_{T} \underbrace{\begin{pmatrix} \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \end{pmatrix}}_{A} = \underbrace{\begin{pmatrix} \vdots & \vdots & \vdots \end{pmatrix}}_{TA}$$

- Then $\forall i < j \; \|A_i - A_j\|_2 \approx \|TA_i - TA_j\|_2$ "wahp"

Some more dimensional reduction!

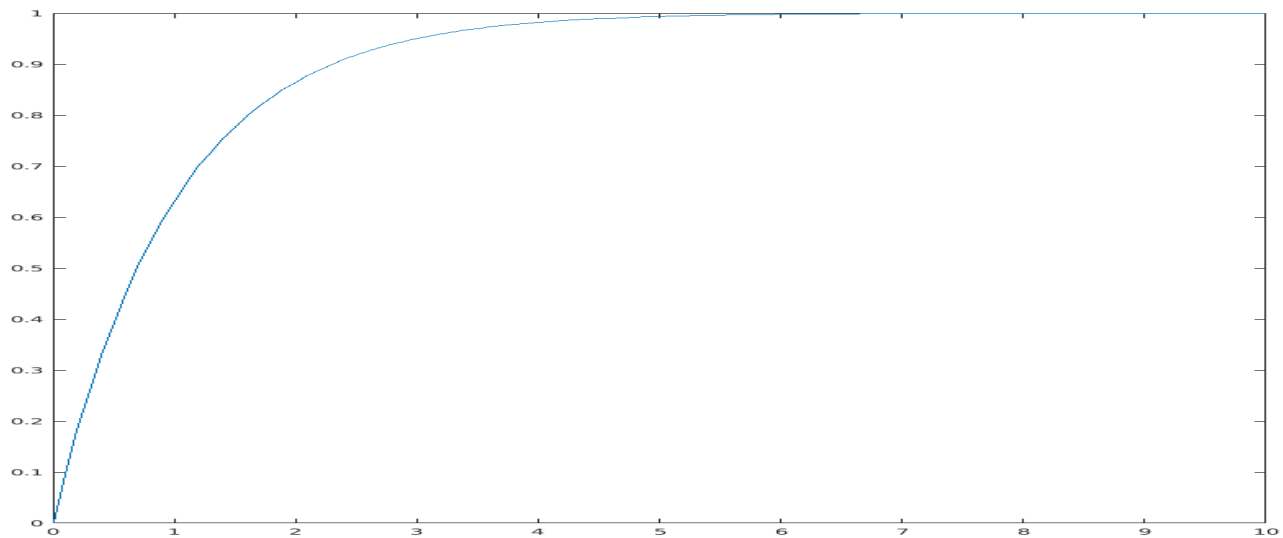
wahp

"wahp" = "with arbitrarily high probability"

the probability of E_k (depending on some parameter k)

approaches 1 *"exponentially fast"* as k increases

$$\mathrm{P}(E_k) \approx 1 - O(e^{-k})$$



Johnson-Lindenstrauss Lemma

Thm.

Given $A \subseteq \mathbb{R}^m$ with $|A| = n$ and $\varepsilon > 0$ there is $k \sim O(\frac{1}{\varepsilon^2} \ln n)$ and a $k \times m$ matrix T s.t.

$$\forall x, y \in A \quad (1-\varepsilon)\|x-y\| \;\leq\; \|Tx - Ty\| \;\leq\; (1+\varepsilon)\|x-y\|$$

If $k \times m$ matrix T is sampled componentwise from $N(0, \frac{1}{\sqrt{k}})$, then

$$\mathrm{P}(A \text{ and } TA \text{ are approximately congruent}) \geq \frac{1}{n}$$

result follows by probabilistic method

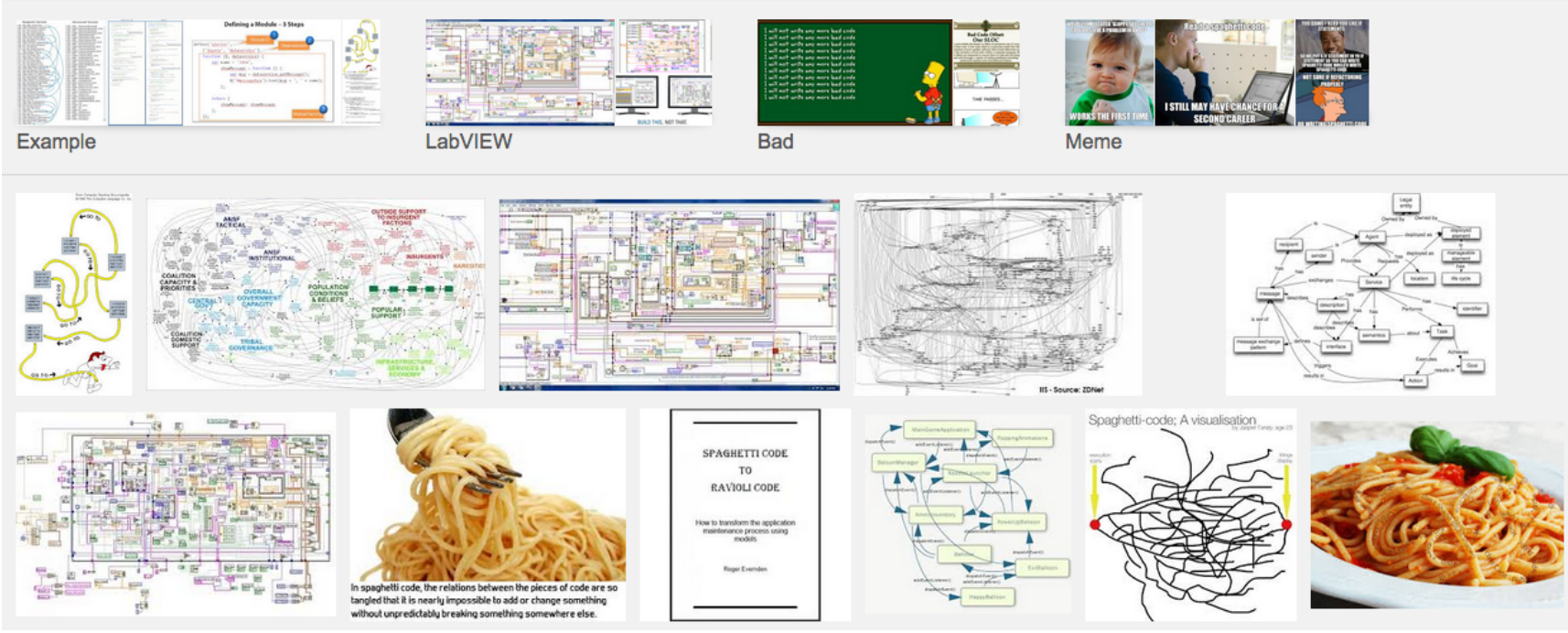
In practice

- P(A and TA are approximately congruent) $\geq \frac{1}{n}$
- re-sampling sufficiently many times gives wahp
- Empirically, sample T very few times (e.g. once will do!)
  $\mathbb{E}_T(\|Tx - Ty\|) = \|x - y\|$
  probability of error decreases exponentially fast

Surprising fact:

k is independent of the original number of dimensions m

Clustering Google images

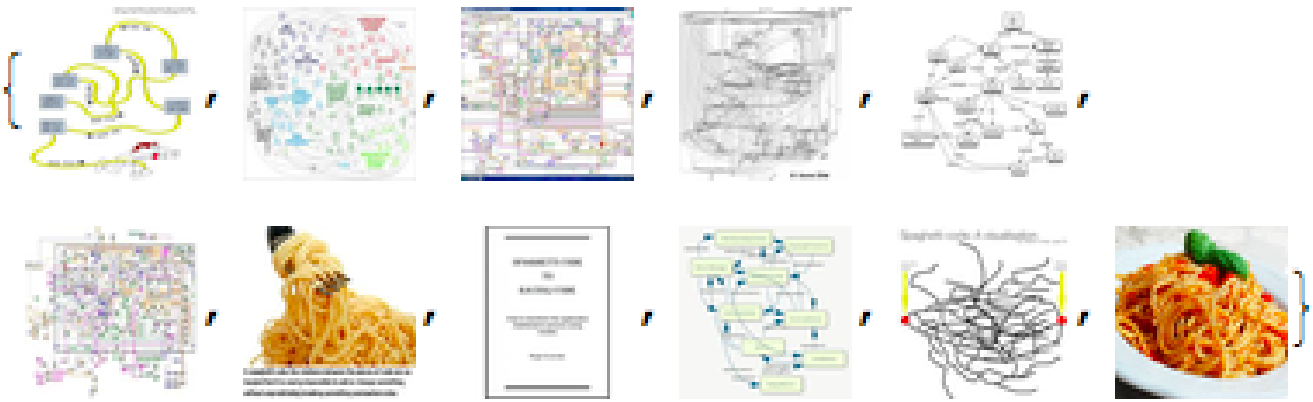


[L. & Lavor, 2017]

Clustering without random projections

```
VHimg = Map[Flatten[ImageData[#]] &, Himg];
```

Out[28]=



```
VHcl = Timing[ClusteringComponents[VHimg, 3, 1]]
```

```
Out[29]= {0.405908, {1, 2, 2, 2, 2, 2, 3, 2, 2, 2, 3}}
```

Too slow!

Clustering with random projections

```
Get["Projection.m"];
VKimg = JohnsonLindenstrauss[VHimg, 0.1];
VKcl = Timing[ClusteringComponents[VKimg, 3, 1]]
Out[34]= {0.002232, {1, 2, 2, 2, 2, 2, 3, 2, 2, 2, 3}}
```

From 0.405s CPU time to 0.00232s

Same clustering

Approximating the identity

- Let $n \in \mathbb{N}$ and $\varepsilon \in \mathbb{R}_+$ with $n \gg 1$ and $\varepsilon \in (0, 1/2)$
- Let T be a $k \times n$ Random Projection (RP) matrix with $k = O(\varepsilon^{-2} \ln n)$
- Look at relations between $TT^\top$ and I_k and $T^\top T$ and I_n
- By [Zhang *et al.*, COLT 13], $\exists \mathcal{C} \geq 1/4$ s.t. $\forall \delta \left(n \geq \frac{(k+1)\ln(2k/\delta)}{\mathcal{C}\varepsilon^{-2}} \right)$

$$\mathsf{Prob}\left(\left\| \frac{1}{n} TT^\top - I_k \right\|_2 \leq \varepsilon \right) \geq 1 - \delta$$

- By Thm. 7 in [L. TOP 20], given any fixed $x \in \mathbb{R}^n$

$$\mathsf{Prob}(-\mathbf{1}\varepsilon \leq T^\top T x - I_n x \leq \mathbf{1}\varepsilon) \geq 1 - 4e^{-d\mathcal{C}\varepsilon^2}$$

- Note: $T^\top T$ only behaves like I_n for a fixed x

  T is $k \times n$, so $T^\top T$ is $n \times n$ with rank $k < n$; hence there can be an arbitrary difference w.r.t. I_n acting on all vectors in $\mathbb{R}^n$

Projecting formulations

- Can we apply RPs to MP formulations?
  - decrease dimension of parameter vectors
  - decrease dimension of variable vectors
- Three major difficulties
  1. Globally optimal objective function value of projected formulation must be approximately the same as of original one
  2. Must be able to retrieve global optima of original formulation from those of projected
  3. Projecting variables is equivalent to projecting an infinite (possibly uncountable) number of vectors, JLL does not apply
- Can project LP, SDP, QP

[Vu *et al.* DAM 19, Vu *et al.* MOR 19, Vu *et al.* IPCO 19, D'Ambrosio *et al.* MPB 20]

Subsection 2

Distance instability

Nearest Neighbours

k-Nearest Neighbours (k-NN). *Given:*

- $k \in \mathbb{N}$
- *a distance function* $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$
- *a set* $\mathcal{X} \subset \mathbb{R}^n$
- *a point* $z \in \mathbb{R}^n \setminus \mathcal{X}$,

find the subset $\mathcal{Y} \subset \mathcal{X}$ *such that:*

(a) $|\mathcal{Y}| = k$

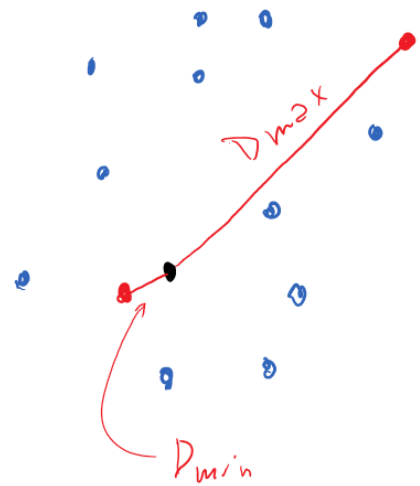
(b) $\forall y \in \mathcal{Y}, x \in \mathcal{X} \quad (d(z, y) \leq d(z, x))$

- basic problem in data science
- pattern recognition, computational geometry, machine learning, data compression, robotics, recommender systems, information retrieval, natural language processing and more
- Example: Used in Step 2 of k-means:

  assign points to closest centroid

[Cover & Hart 1967]

With random variables



- Consider 1-NN
- Let $\ell = |\mathcal{X}|$
- Distance function family
  $\{d^m : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+\}_m$
- For each m:
  - random variable Z^m with some distribution over $\mathbb{R}^n$
  - for $i \leq \ell$, random variable X_i^m with some distrib. over $\mathbb{R}^n$
  - X_i^m iid w.r.t. i, Z^m independent of all X_i^m
  - $D_{\min}^m = \min_{i \leq \ell} d^m(Z^m, X_i^m)$
  - $D_{\max}^m = \max_{i \leq \ell} d^m(Z^m, X_i^m)$

Distance Instability Theorem

- Let $p > 0$ be a constant
- If

$$\exists i \leq \ell \quad (d^m(Z^m, X_i^m))^p \textit{ converges as } m \to \infty$$

then, for any $\varepsilon > 0$,

closest and furthest point are at about the same distance

Note "$\exists i$" suffices since $\forall m$ we have X_i^m iid w.r.t. i

[Beyer et al. 1999]

Distance Instability Theorem

- Let $p > 0$ be a constant
- If

$$\forall i \leq \ell \lim_{m \to \infty} \mathsf{Var}\left(\frac{(d^m(Z^m, X_i^m))^p}{\mathbb{E}((d^m(Z^m, X_i^m))^p)} \right) = 0$$

then, for any $\varepsilon > 0$,

$$\lim_{m \to \infty} \mathbb{P}(D_{\text{max}}^m \leq (1 + \varepsilon) D_{\text{min}}^m) = 1$$

Note "$\exists i$" suffices since $\forall m$ we have X_i^m iid w.r.t. i

[Beyer et al. 1999]

Preliminary results

- *Lemma*. $\{B^m\}_m$ seq. of rnd. vars with finite variance and $\lim_{m\to\infty} \mathbb{E}(B^m) = b \wedge \lim_{m\to\infty} \mathsf{Var}(B^m) = 0$; then

$$\forall \varepsilon > 0 \quad \lim_{m\to\infty} \mathbb{P}(\|B^m - b\| \leq \varepsilon) = 1$$

denoted $B^m \to_{\mathbb{P}} b$

- *Slutsky's theorem*. $\{B^m\}_m$ seq. of rnd. vars and g a continuous function; if $B^m \to_{\mathbb{P}} b$ and $g(b)$ exists, then $g(B^m) \to_{\mathbb{P}} g(b)$

- *Corollary*. If $\{A^m\}_m, \{B^m\}_m$ seq. of rnd. vars. s.t. $A^m \to_{\mathbb{P}} a$ and $B^m \to_{\mathbb{P}} b \neq 0$ then $\frac{A^m}{B^m} \to_{\mathbb{P}} \frac{a}{b}$

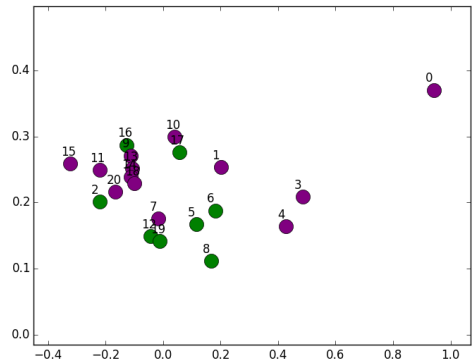
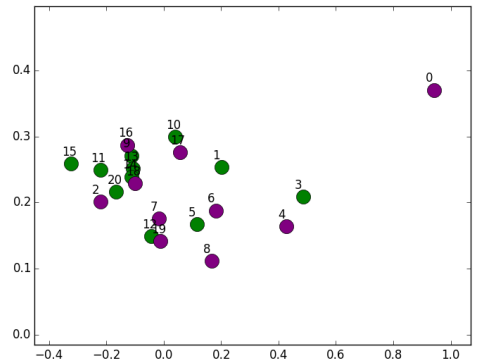
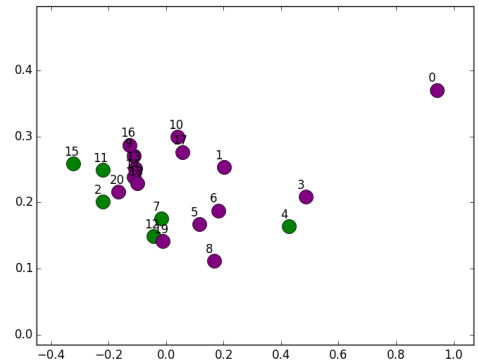
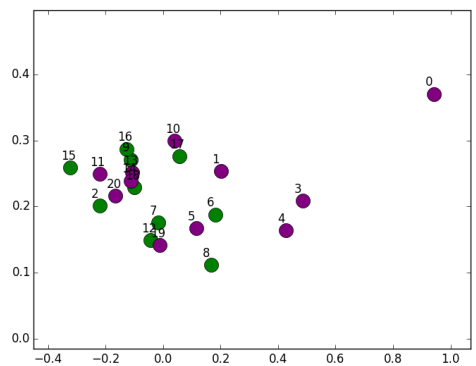
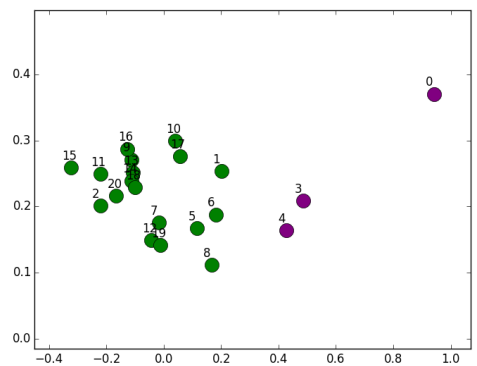
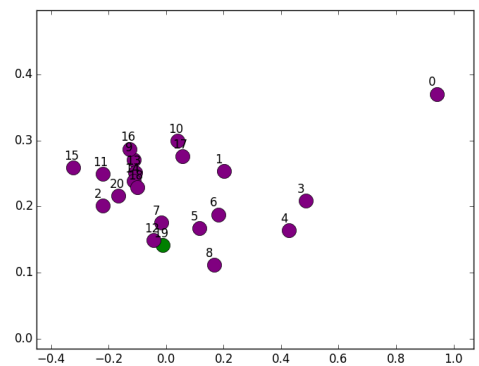
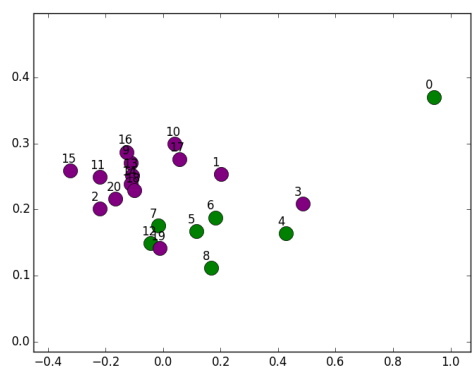
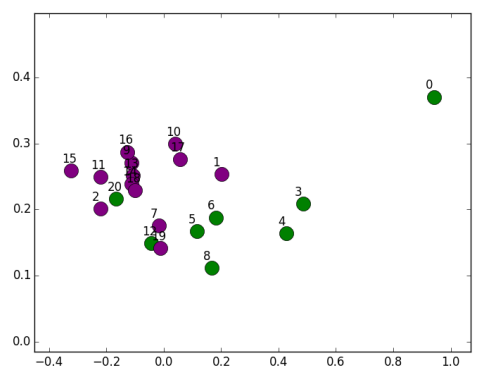
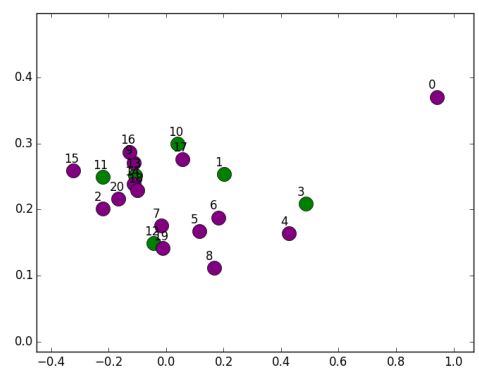
Proof

1. $\mu_m = \mathbb{E}((d^m(Z^m, X_i^m))^p)$ independent of i
   (since all X_i^m iid)
2. $V_m = \frac{(d^m(Z^m, X_i^m))^p}{\mu_m} \to_{\mathbb{P}} 1$:
   - $\mathbb{E}(V_m) = 1$ (rnd. var. over mean) $\Rightarrow \lim_m \mathbb{E}(V_m) = 1$
   - Hypothesis of thm. $\Rightarrow \lim_m \mathsf{Var}(V_m) = 0$
   - *Lemma* $\Rightarrow V_m \to_{\mathbb{P}} 1$
3. $\mathbf{V}^m = (V_m \mid i \leq \ell) \to_{\mathbb{P}} \mathbf{1}$ (by iid)
4. *Slutsky's thm.* $\Rightarrow \min(\mathbf{V}^m) \to_{\mathbb{P}} \min(\mathbf{1}) = 1$
   simy for max
5. *Corollary* $\Rightarrow \frac{\max(\mathbf{V}^m)}{\min(\mathbf{V}^m)} \to_{\mathbb{P}} 1$
6. $\frac{D^m_{\max}}{D^m_{\min}} = \frac{\mu_m \max(\mathbf{V}^m)}{\mu_m \min(\mathbf{V}^m)} \to_{\mathbb{P}} 1$
7. Result follows (defn. of $\to_{\mathbb{P}}$ and $D^m_{\max} \geq D^m_{\min}$)

When it applies

- iid random variables from any distribution
- Particular forms of correlation
  e.g. $U_i \sim \mathsf{Uniform}(0, \sqrt{i})$, $X_1 = U_1$, $X_i = U_i + (X_{i-1}/2)$ for $i > 1$
- Variance tending to zero
  e.g. $X_i \sim \mathsf{N}(0, 1/i)$
- Discrete uniform distribution on m-dimensional hypercube
  for both data and query
- Computational experiments with k-means:
  instability already with $n > 15$

Example of k-means in $\mathbb{R}^{100}$

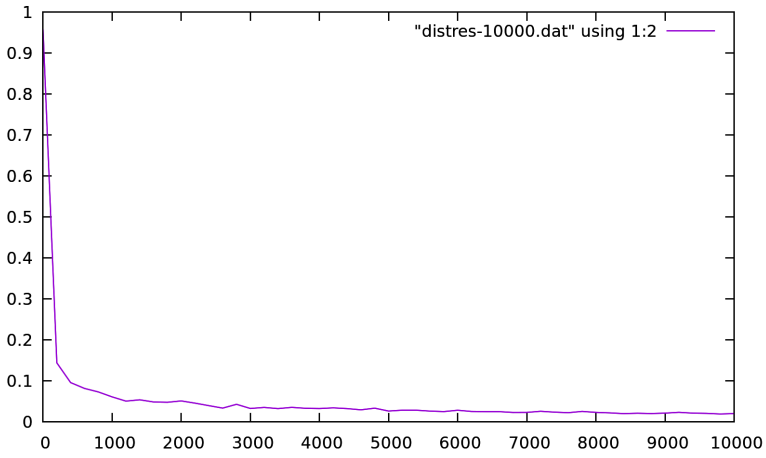










. . . and when it doesn't

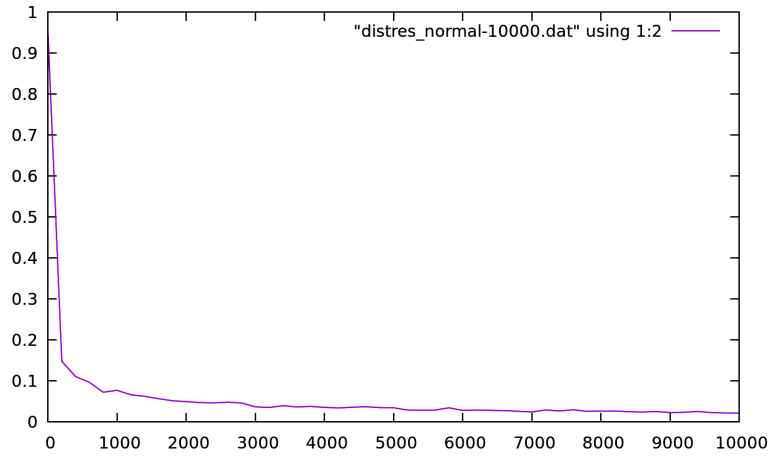
- Complete linear dependence on all distributions
  can be reduced to NN in 1D
- Exact and approximate matching
  query point = (or ≈) data point
- Query point in a well-separated cluster in data
- Implicitly low dimensionality
  project; but NN must be stable in lower dim.

Loss of resolution ε for $K \leq 10000$

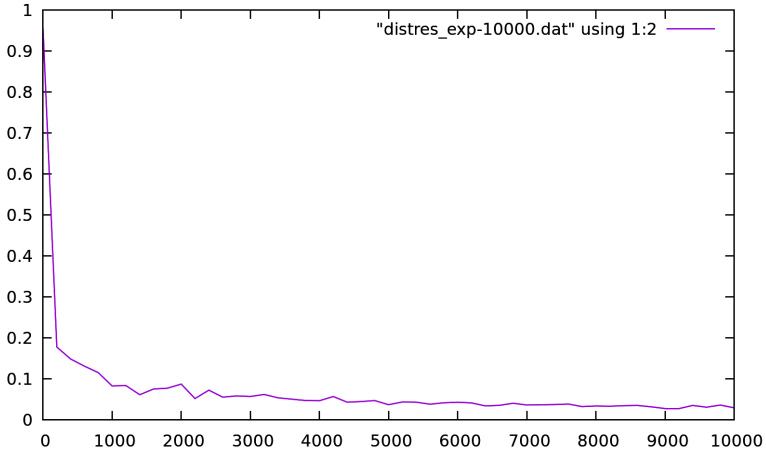
Uniform$(0, 1)$



Normal$(0, 1)$



Exponential(1)



- Resolution falls exponentially fast
- Hard to tell closest from furthest pt
  a fortiori, picking k closest neighb
- Generates algorithmic instability

Subsection 3

Are these results related?

Relations between distance instability and JLL?

- *On the one hand,*
  - n-dimensional point cloud $\mathcal{X}$ with $n \gg 1$
  - Distance instability $\Rightarrow D_{\max} \leq (1+\varepsilon) D_{\min}$
    all the useful information within ε
  - T a rnd. prj. $\Rightarrow$ ε-congruent $T\mathcal{X}$ in $k = O(\varepsilon^{-2} \ln |\mathcal{X}|)$ dim.
    all of the useful information is lost (?)
- *On the other hand,*
  - n-dimensional point cloud $\mathcal{X}$ with $n \gg 1$
  - $T = (T_{ij})$ with $T_{ij} \sim \mathcal{N}(0,1) \Rightarrow$ ε-congruent $T\mathcal{X} \subset \mathbb{R}^k$
  - Columns in $T\mathcal{X}$ are identically distributed
  - k still "largish" (because of ε^{-2})
    distance instability $\Rightarrow$ closest/furthest points could be mistaken

Graph embeddings for artificial Neural Networks

Artificial neural networks

- *Artificial Neural Network* (ANN):
  explicit approximate representation of a function oracle

$$\xi = f(\chi)$$

  where $\chi \in \mathbb{R}^n$ and $\xi \in \mathbb{R}^k$
- Parametrized discrete dynamical system with 2 phases
  1. training (slow)
  2. evaluation (fast)
- Learning phase assigns values to ANN parameters
  learning algorithm is based on a given training dataset
- Once parametrized, ANN evaluates given input to output

Formal definition

- ANN $\mathcal{A}$: triplet $(G, T, \phi : \mathbb{R} \to [-1, 1])$
  - G is a *Directed Acyclic Graph* (DAG)
  - T is a *training set*
  - ϕ is an *activation function*
- DAG $G = (V, A, v, b, w, I, O)$:
  - node weight function $v : V \to \mathbb{R}$ (variables)
  - node weight function $b : V \to \mathbb{R}$ (parameter to be learned)
  - arc weight function $w : A \to \mathbb{R}$ (parameter to be learned)
  - $I \subset V$ s.t. $|I| = n$ and $N^-(I) = \varnothing$: input nodes
  - $O \subset V$ s.t. $|O| = k$ and $N^+(O) = \varnothing$: output nodes
- Training set $T = (X \subset \mathbb{R}^n, Y \subset \mathbb{R}^k)$
  with $|X| = |Y| = \hat{R}$ and $R = \{1, \ldots, \hat{R}\}$

Training

- Finds values for b, w using the training data from T
- Decision variable map $u : R \times V \to \mathbb{R}$
- For subset $S \subseteq R$ and $U \subset V$ let $u[S, U]$ the submatrix of u with rows indexed by S and columns indexed by U
- Let $\mathsf{dist}(\cdot, \cdot)$ denote a function evaluating a non-negative difference score between two equally sized matrices
- Training problem:

$$\left.\begin{array}{rll} & \min\limits_{w,b,u} & \mathsf{dist}(u[R, O], Y) \\ & & u[R, I] = X \\ \forall t \in R, j \in V \setminus I & & u_{tj} = \phi\Big(\sum\limits_{i \in N^-(j)} w_{ij} u_{ti} + b_j\Big) \end{array}\right\}$$

Evaluation

- b, w have been fixed by training
- Evaluation problem:

$$\left.\begin{array}{rrcl} \forall j \in I & v_j & = & \chi_j \\ \forall j \in V \setminus I & v_j & = & \phi_j\Big(\sum\limits_{i \in N^-(j)} w_{ij} v_i + b_j\Big) \\ \forall j \in O & \xi_j & = & v_j \end{array}\right\}$$

- Since G is a DAG, for given χ can solve for ξ in linear time

Subsection 1

A clustering task

Natural language processing

- Aims at determining:
  - segmentation of text into sentences and words
  - lemmatization (removing desinence/conjugation syllables)
  - part-of-speech tagging (grammatical category of word)
  - finding the grammatical structure of sentences (parsing)
  - named entity recognition
  - relation extraction
  - machine translation
  - sentiment analysis
  - …
- Common methodologies:
  - deterministic algorithms (automaton-based)
  - symbolic artificial intelligence (e.g. LISP, Prolog)
  - supervised, semi-supervised, unsupervised machine learning
- In particular, ANNs very successful in machine translation

Clustering sentences

- Cluster sentences according to a similarity using an ANN
  assume input text is "clean" (lemmatized, stopwords removed)
- Sentences are linear sequences of words
  transform them into a graph on words
  - vertices are words
  - sliding window on word sequences:
    adjacency relation (word neighbourhoods)
  - contract vertices with same word,
    parallel edges represented by single edges with sum of weights
- ANN requires vectors in $\mathbb{R}^n$ as input
  we have differently sized graphs over the sentences
- Use DG methods, stack word vectors, pad with zeros,
  dimensional reduction

Some results

- *Simple ANN, one hidden layer*
- Training sets formed by k-means over sentence vectors
- formed with inc, qrt, sdp
- dimensional reduction with pca, rp
- *table below reports ANN loss function values*

Training set inputs	Training set outputs							
	μ / ρ	inc pca	inc rp	qrt pca	qrt rp	sdp pca	sdp rp	sum $\mu \in M$
	inc pca	*0.052*	*0.013*	0.106	0.164	0.079	0.161	0.575
	inc rp	*0.001*	*0.000*	0.106	0.167	0.080	0.159	0.513
	qrt pca	0.063	0.022	*0.038*	*0.218*	0.079	0.159	0.579
	qrt rp	0.062	0.024	*0.120*	*0.035*	0.076	0.164	0.481
	sdp pca	0.063	0.021	0.126	0.195	*0.033*	*0.149*	0.587
	sdp rp	0.059	0.021	0.121	0.176	*0.083*	*0.037*	0.497

Some of my references for this tutorial

1. L., *DG and Data Science*, TOP 28:271-339, 2020
2. L., *A New DG Method for Constructing Word and Sentence Vectors*, in *Proc. of WWW*, Companion Volume, p. 679-685, 2020
3. D'Ambrosio, L., Poirion, Vu, *Random projections for quadratic programs*, Math. Prog. B, 183:619-647, 2020
4. Vu, Poirion, D'Ambrosio, L., *Random Projections for Quadratic Programs over a Euclidean Ball*, in A. Lodi *et al.* (eds.), *Proc. of IPCO*, LNCS 11480:442-452, 2019
5. Vu, Poirion, L., *Random Projections for LP*, Math. Oper. Res., 43:1051-1071, 2019
6. Vu, Poirion, L., *Gaussian random projections for Euclidean membership problems*, Discr. Appl. Math, 253:93-102, 2019
7. L., Vu, *Barvinok's naive algorithm in DG*, Op. Res. Lett., 46:476-481, 2018
8. D'Ambrosio, Vu, Lavor, L., Maculan, *New Error Measures and Methods for Realizing Protein Graphs from Distance Data*, Discr. Comp. Geom., 57(2):371-418, 2017
9. Mencarelli, Sahraoui, L., *A multiplicative weights update algorithm for MINLP*, Eur. J. Comp. Opt., 5:31-86, 2017
10. L., Lavor, *Euclidean DG: an Introduction*, Springer, New York, 2017
11. L., D'Ambrosio, *The Isomap algorithm in distance geometry*, in Iliopoulos et al. (eds.), *Proceedings of SEA*, LIPICS 75(5)1:13, Dagstuhl Publishing, 2017
12. Dias, L., *Diagonally Dominant Programming in DG*, in Cerulli et al. (eds.) *Proceedings of ISCO*, LNCS 9849:225-246, Springer, Zürich, 2016
13. L., Lavor, Maculan, Mucherino, *Euclidean distance geometry and applications*, SIAM Review, 56(1):3-69, 2014
14. Lavor, L., Maculan, *Computational Experience with the Molecular DG Problem*, in Pintér (ed.) *Global Optimization: Scientific and Engineering Case Studies*, Springer, Berlin, 2006