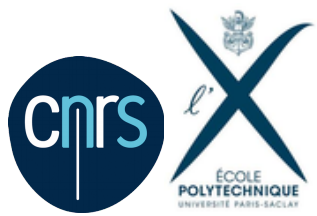
Distance Geometry in Data Science

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

CNMAC 2017



Line of reasoning for this talk

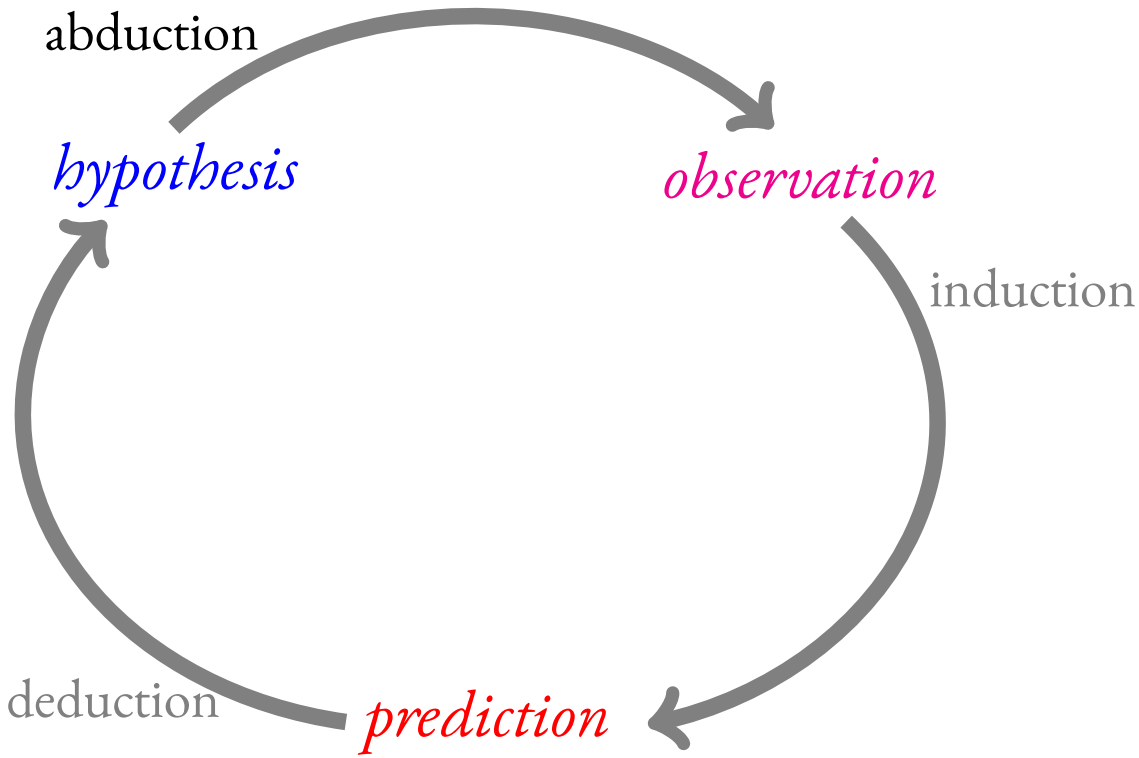
1. Graphs and weighted graphs necessary to model data
2. Computers can "reason by analogy" (clustering)
3. Clustering on vectors allows more flexibility
4. Need to embed (weighted) graphs into Euclidean spaces
5. High dimensions make clustering expensive/unstable
6. Use random projections to reduce dimensions

Outline

Reasoning

The philosophical motivation to distance geometry

Modes of rational thought



[Arist. 24a, Peirce CP, Putnam 79, Eco 83]

Modes of rational thought

$\underbrace{\textit{All humans are mortal}}_{\text{hypothesis}}$, $\underbrace{\textit{Socrates is human}}_{\text{prediction}}$, hence $\underbrace{\textit{Socrates is mortal}}_{\text{observation}}$

- deduction (hypothesis + prediction → observation)
  TRUTH; *logician, mathematician*
- induction (observation + prediction → hypothesis)
  CAUSALITY; *physicist, chemist, biologist*
- abduction (hypothesis + observation → prediction)
  PLAUSIBILITY; *everyone else*

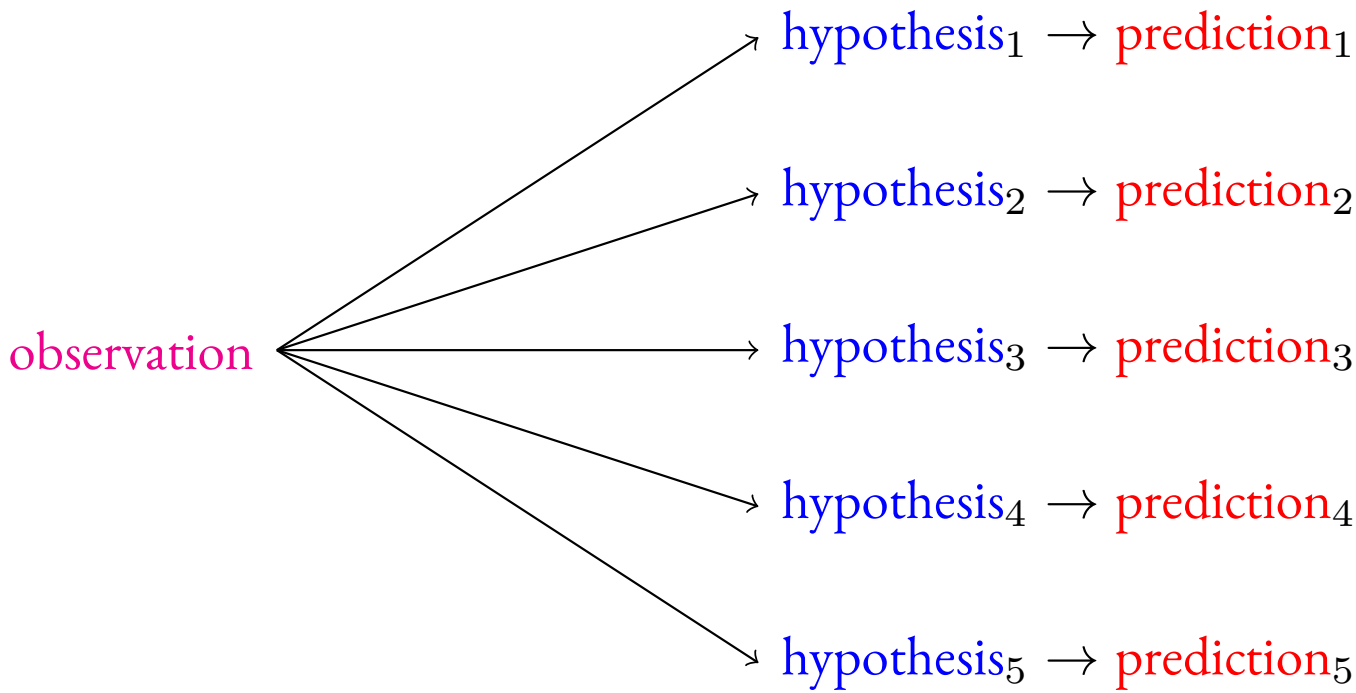
Abduction might infer falsehoods

- Peirce:
  1. All beans in this bag are white
  2. There is a white bean next to this bag
  3. The bean was in the bag
  4. *but what if the bean wasn't in the bag?*
- Sherlock Holmes wannabe:
  1. People who walk in the park have their shoes full of dirt
  2. John's shoes are dirty
  3. John walked in the park
  4. *but what if John did not walk in the park?*

Only deduction infers truth

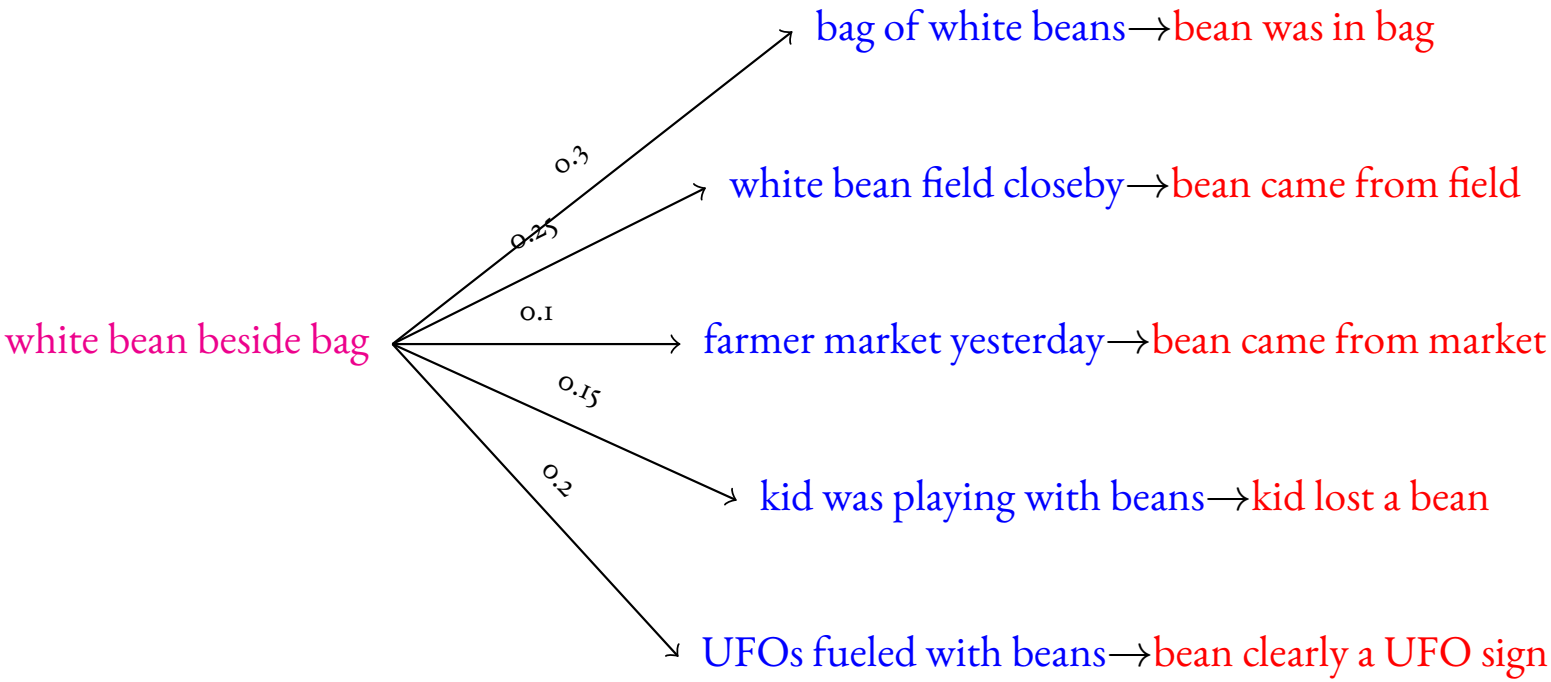
[Desclés, Jackiewicz 2006]

Statistician's abduction



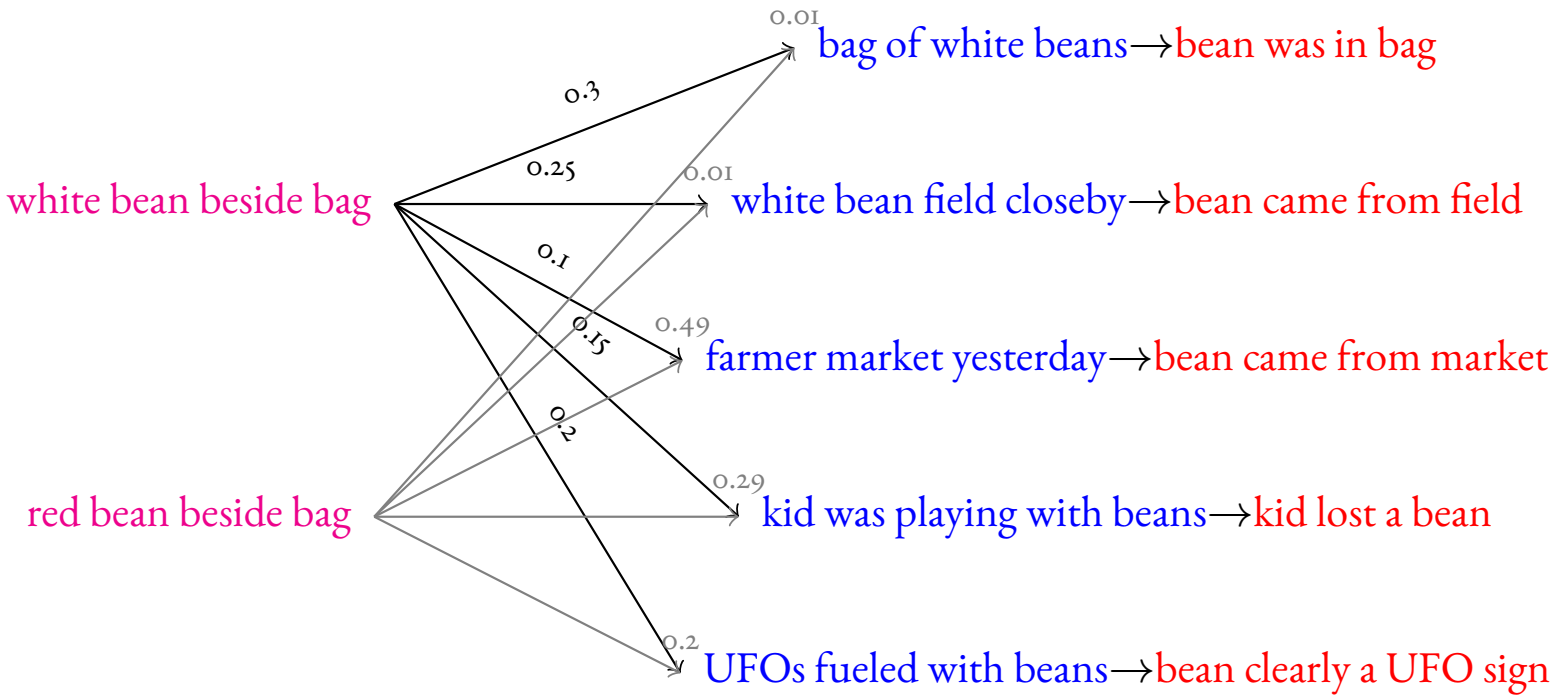
- Evaluate $\mathbb{P}(\text{observation} \mid \text{hypothesis}_i \rightarrow \text{prediction}_i)\ \forall i$
- Choose inference i with largest probability

Example



- Repeat experiences, collect data
- Probability distribution $\Leftarrow \begin{cases} \text{frequency} \\ \text{personal conviction} \end{cases}$

Compare different observations



- Repeat experiences, collect data
- Probability distribution $\Leftarrow \begin{cases} \text{frequency} \\ \text{personal conviction} \end{cases}$

Subsection 1

Relations, graphs, distances

Modelling a good prediction

Observation graph

- set V of observations
- set I of inferences (hypotheses $\wedge$ predictions)
- $\forall v \in V$ get probability distribution P^v on I
- relation E if $u, v \in V$ have similar distributions on I
- $\mathcal{F} = (V, E)$: observation graph
- relation $\sim$ if $h, k \in I$ not contradictory
- *Densest subgraphs U with every $h_u \sim k_u$ (for $u \in U$)*

richest observation sets with non-contradictory inferences

Think of Sherlock Holmes:
set of clues compatible with most likely consistent explanations

Example

- $V = \{u, v\}$ where $u =$ white bean, $v =$ red bean

  lots of beans (both red and white) found next to all-white bean bag
- largest combined probabilities:
  1. *farmer market*: 0.59
  2. *kid playing*: 0.34
  3. *UFO fuel*: 0.4
- UFO hovering above market square $\rightarrow$ farmer market disbanded

  $\Rightarrow 1 \sim 2 \wedge 2 \sim 3$ but $\neg(1 \sim 3)$
- *Observation graph*:
  - $\mathbb{P}(u \cup v \mid 1 \vee 2) = 0.93 > 0.74 = \mathbb{P}(u \cup v \mid 2 \vee 3)$
  - $\Rightarrow U = V = \{u, v\}, E = \{\{u, v\}\}$
  - with scaled edge weight $0.93/(0.93 + 0.74) = 0.55$

Where we are and where we are going

- Relations on observations
  encoding most likely compatible predictions ⇒ graphs
- Similarity
  probability / magnitude / intensity ⇒ weighted graphs

- "Machine intelligence" by analogy:
  clustering in graphs
- More refined clustering techniques?
  - pull in tools from linear algebra
  - work with vectors rather than graphs
  - Euclidean embeddings of weighted graphs
- Distances lose "resolution" in high dimensions
- Project into lower dimensional spaces

Outline

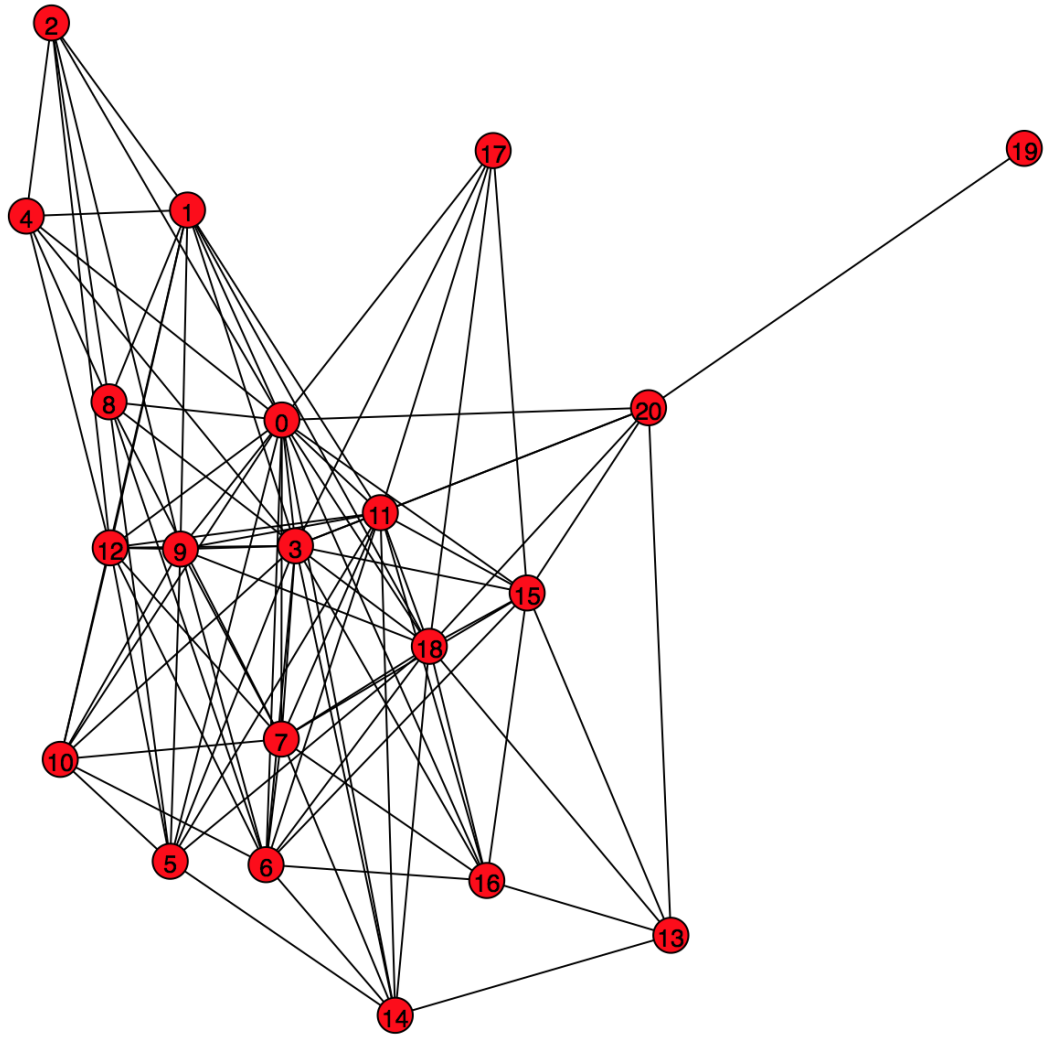
Clustering

"Machine intelligence":
analogy based on proximity

Subsection 1

Clustering in graphs

Example graph

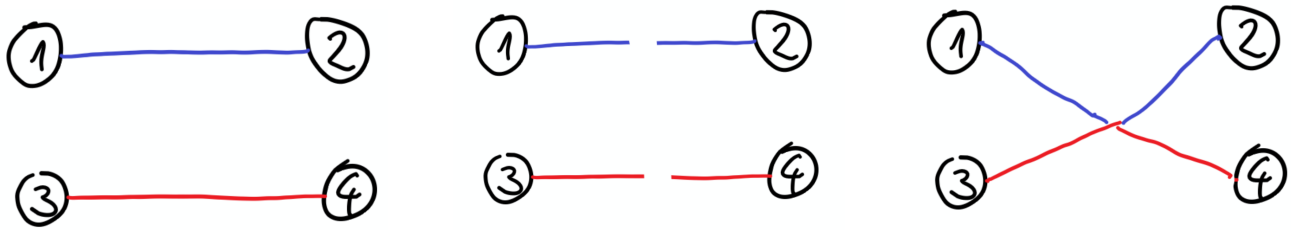


- Goal: *find partition in densest subgraphs*

Modularity clustering

"Modularity is the fraction of the edges that fall within a cluster minus the expected fraction if edges were distributed at random."

- *"at random"* = random graphs over same degree sequence
- *degree sequence* $= (k_1, \ldots, k_n)$ where $k_i = |N(i)|$
- *"expected"* = all possible "half-edge" recombinations



- expected edges between u, v: $k_u k_v/(2m)$ where $m = |E|$
- $\mathsf{mod}(u, v) = (A_{uv} - k_u k_v/(2m))$
- $\mathsf{mod}(G) = \sum_{\{u,v\} \in E} \mathsf{mod}(u, v) x_{uv}$

  $x_{uv} = 1$ if u, v in the same cluster and 0 otherwise
- *"Natural extension" to weighted graphs:* $k_u = \sum_v A_{uv}$, $m = \sum_{uv} A_{uv}$

[Girvan & Newman 2002]

Use modularity to define clustering

- What is the "best clustering"?
- *Maximize discrepancy between actual and expected*
  "as far away as possible from average"

$$\left.\begin{array}{rl} \max & \sum_{\{u,v\}\in E} \mathsf{mod}(u,v)x_{uv} \\ \forall u \in V, v \in V & x_{uv} \in \{0,1\} \end{array}\right\}$$

- Issue: trivial solution $x = 1$ "one big cluster"
- Idea: *treat clusters as cliques* *(even if zero weight)*
  then clique partitioning constraints for transitivity

$$\begin{array}{rrcl} \forall i<j<k & x_{ij} + x_{jk} - x_{ik} & \leq & 1 \\ \forall i<j<k & x_{ij} - x_{jk} + x_{ik} & \leq & 1 \\ \forall i<j<k & -x_{ij} + x_{jk} + x_{ik} & \leq & 1 \\ \forall \{i,j\} \notin E & x_{ij} & = & 0 \end{array}$$

if $i, j \in C$ and $j, k \in C$ then $i, k \in C$

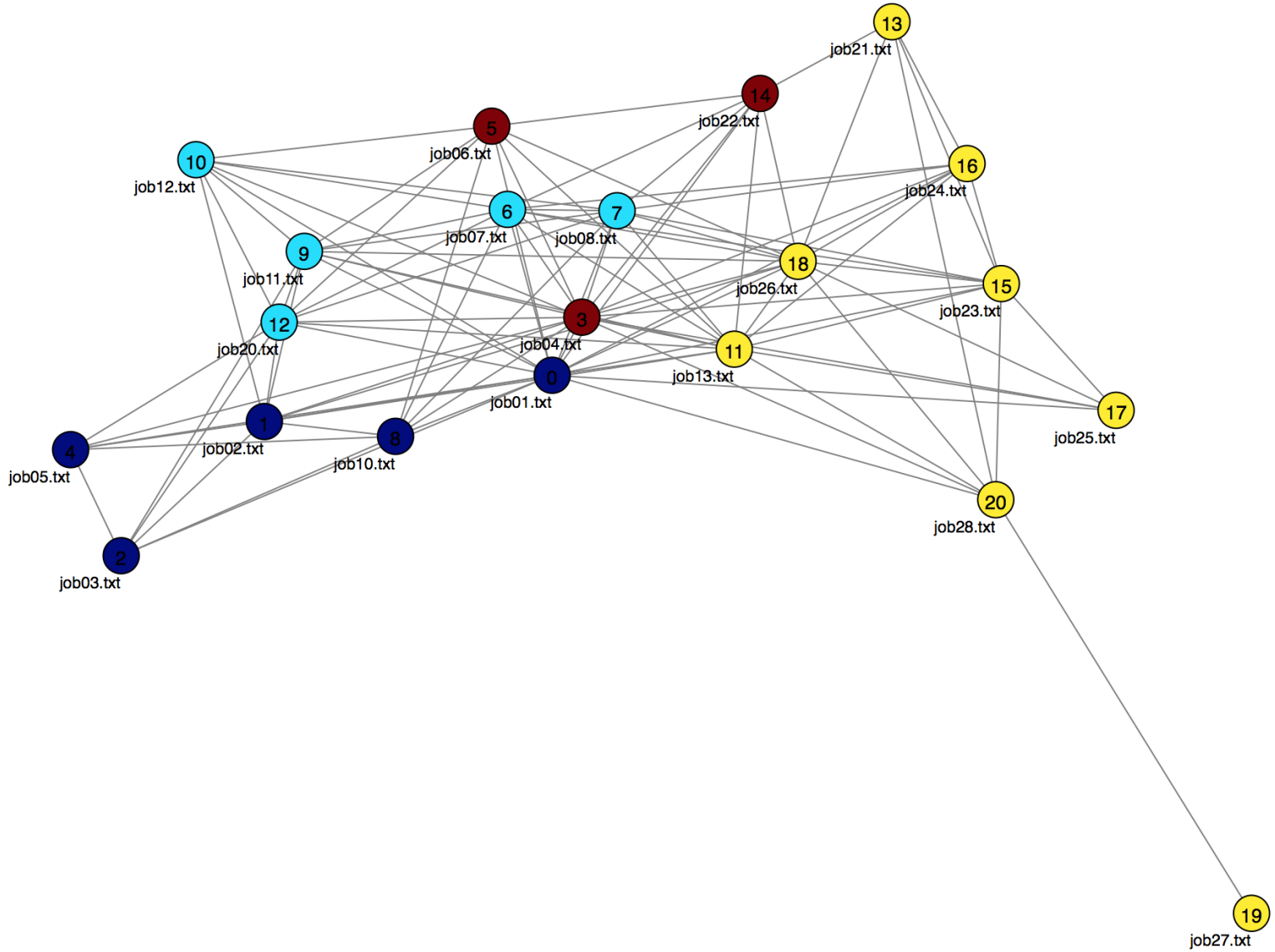
[Aloise et al. 2010]

Maximizing the modularity of a graph

- Formulation above is a Mathematical Program (MP)
  - *MP is a formal language for describing optimization problems*
  - *each MP consists of:*
    - *parameters (input)*
    - *decision variables (output)*
    - *objective function(s)*
    - *explicit and implicit constraints*
  - *broad MP classification:*
    LP, SDP, cNLP, NLP, MILP, cMINLP, MINLP
- Modularity Maximization MP is a MILP
- *MILP is* **NP***-hard but* $\exists$ *technologically advanced solvers*
- Otherwise, use (fast) heuristics
- *This method decides the number of clusters*

[Cafieri et al. 2014]

The resulting clustering



Subsection 2

Clustering in Euclidean spaces

Minimum sum-of-squares clustering

- MSSC, a.k.a. the *k-means problem*
- Given points $p_1, \ldots, p_n \in \mathbb{R}^m$, find clusters $C_1, \ldots, C_d$

$$\min \sum_{j \leq k} \sum_{i \in C_j} \|p_i - \text{centroid}(C_j)\|_2^2$$

where $\text{centroid}(C_j) = \frac{1}{|C_j|} \sum_{i \in C_j} p_i$

- k-means alg.: given initial clustering $C_1, \ldots, C_d$

1: $\forall j \leq d$ compute $y_j = \text{centroid}(C_j)$
2: $\forall i \leq n, j \leq d$ if y_j is the closest centroid to p_i let $x_{ij} = 1$ else 0
3: $\forall j \leq d$ update $C_j \leftarrow \{p_i \mid x_{ij} = 1 \wedge i \leq n\}$
4: repeat until stability

In "k-means", "k" is the number of clusters, here denoted by d

note that d is given

[MacQueen 1967, Aloise et al. 2012]

MP formulation

$$
\left.
\begin{array}{lrcl}
\min\limits_{x,y,s} & \sum\limits_{i\le n}\sum\limits_{j\le d} \|p_i - y_j\|_2^2\, x_{ij} & & \\
\forall j \le d & \frac{1}{s_j}\sum\limits_{i\le n} p_i x_{ij} & = & y_j \\
\forall i \le n & \sum\limits_{j\le d} x_{ij} & = & 1 \\
\forall j \le d & \sum\limits_{i\le n} x_{ij} & = & s_j \\
\forall j \le d & y_j & \in & \mathbb{R}^m \\
& x & \in & \{0,1\}^{nd} \\
& s & \in & \mathbb{N}^d
\end{array}
\right\} \quad \text{(MSSC)}
$$

MINLP: nonconvex terms; continuous, binary and integer variables

Reformulations

The (MSSC) formulation has the same optima as:

$$
\left.
\begin{array}{rrcl}
\min\limits_{x,y,P} & \sum\limits_{i\le n}\sum\limits_{j\le d} P_{ij}\, x_{ij} & & \\
\forall i \le n, j \le d & \|p_i - y_j\|_2^2 & \le & P_{ij} \\
\forall j \le d & \sum\limits_{i\le n} p_i x_{ij} & = & \sum\limits_{i\le n} y_j x_{ij} \\
\forall i \le n & \sum\limits_{j\le d} x_{ij} & = & 1 \\
\forall j \le d & y_j & \in & ([\min\limits_{i\le n} p_{ia}], \max\limits_{i\le n} p_{ia} \mid a \le d) \\
& x & \in & \{0,1\}^{nd} \\
& P & \in & [0, P^U]^{nd}
\end{array}
\right\}
$$

- Only nonconvexities:
  products of bounded by binary variables
- *Caveat*: cannot have empty clusters

Products of binary and continuous vars.

- Suppose term xy appears in a formulation
- Assume $x \in \{0, 1\}$ and $y \in [0, 1]$ is bounded
- means "either $z = 0$ or $z = y$"
- *Replace xy by a new variable z*
- Adjoin the following constraints:

$$\begin{aligned} z &\in [0, 1] \\ y - (1 - x) \leq z &\leq y + (1 - x) \\ -x \leq z &\leq x \end{aligned}$$

- $\Rightarrow$ Everything's linear now!

[Fortet 1959]

Products of binary and continuous vars.

- Suppose term xy appears in a formulation
- Assume $x \in \{0, 1\}$ and $y \in [y^L, y^U]$ is bounded
- means "either $z = 0$ or $z = y$"
- *Replace xy by a new variable z*
- Adjoin the following constraints:

$$\begin{aligned} z &\in [\min(y^L, 0), \max(y^U, 0)] \\ y - (1-x)\max(|y^L|, |y^U|) \leq z &\leq y + (1-x)\max(|y^L|, |y^U|) \\ -x\max(|y^L|, |y^U|) \leq z &\leq x\max(|y^L|, |y^U|) \end{aligned}$$

- $\Rightarrow$ Everything's linear now!

[L. et al. 2009]

MSSC is a convex MINLP

$$
\begin{array}{rrcll}
 & \min\limits_{x,y,P,\chi,\xi} \quad \sum\limits_{i\le n}\sum\limits_{j\le d} \chi_{ij} & & & \\
\forall i \le n, j \le d & 0 \le & \chi_{ij} & \le P_{ij} & \\
\forall i \le n, j \le qquad d & P_{ij} - (1 - x_{ij})P^U \le & \chi_{ij} & \le x_{ij} P^U & \\
\forall i \le n, j \le d & \|p_i - y_j\|_2^2 & \le & P_{ij} & \Leftarrow \text{ convex} \\
\forall j \le d & \sum\limits_{i\le n} p_i x_{ij} & = & \sum\limits_{i\le n} \xi_{ij} & \\
\forall i \le n, j \le d & y_j - (1 - x_{ij}) \max(|y^L|, |y^U|) \le & \xi_{ij} & \le y_j + (1 - x_{ij}) \max(|y^L|, |y^U|) & \\
\forall i \le n, j \le d & - x_{ij} \max(|y^L|, |y^U|) \le & \xi_{ij} & \le x_{ij} \max(|y^L|, |y^U|) & \\
\forall i \le n & \sum\limits_{j\le d} x_{ij} & = & 1 & \\
\forall j \le d & y_j & \in & [y^L, y^U] & \\
 & x & \in & \{0,1\}^{nd} & \\
 & P & \in & [0, P^U]^{nd} & \\
 & \chi & \in & [0, P^U]^{nd} & \\
\forall i \le n, j \le d & \xi_{ij} & \in & [\min(y^L, 0), \max(y^U, 0)] &
\end{array}
$$

y_j, ξ_{ij}, y^L, y^U are vectors in $\mathbb{R}^m$

Solving the MSSC

- k-means
  - heuristic (optimum not guaranteed)
  - fast, well-known, lots of analyses
  - scales reasonably well
  - implemented in practically all languages
- convex MINLP
  - exact (guaranteed global optima)
  - reasonably fast only for small sizes
  - scales exponentially
  - Solvers: KNITRO (commercial), Bonmin (free)
    need an MP language interpreter (AMPL)

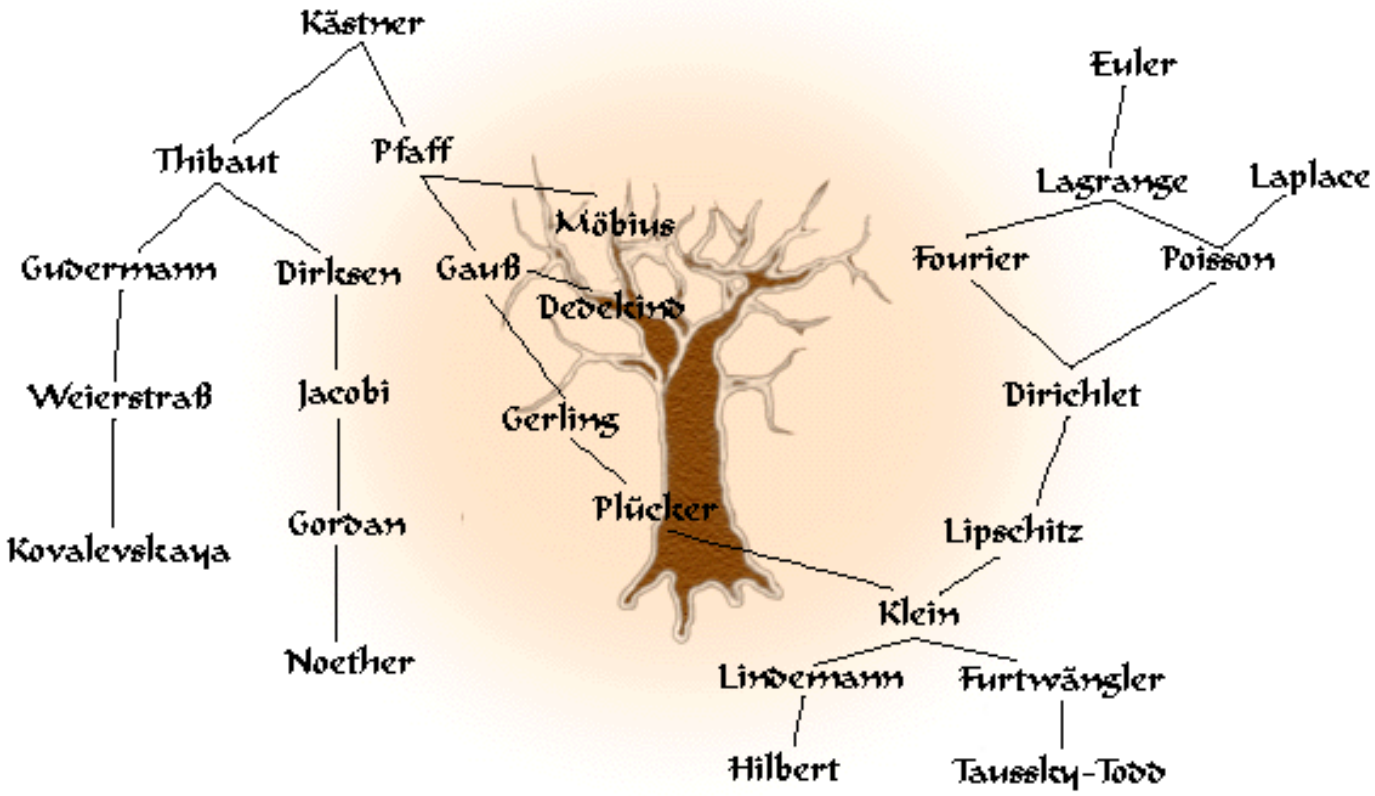
Outline

Metric embeddings

Mapping metric spaces into each other

Graphs with shortest path metric

E.g. mathematical genealogy skeleton



The distance matrix

(only for a subgraph)

	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

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

Subsection 1

Fréchet embeddings in ℓ_∞

ℓ_∞ embeddings

- Given a metric space (X, d) with distance matrix $D = (d_{ij})$
- Consider i-th row $\delta_i = (d_{i1}, \ldots, d_{in})$ of D
- Embed $i \in X$ by vector $\delta_i \in \mathbb{R}^n$
- Define $f(X) = \{\delta_1, \ldots, \delta_n\}$, $f(d(i,j)) = \|f(i) - f(j)\|_\infty$
- Thm.: $(f(X), \ell_\infty)$ is a metric space with distance matrix D
- If (X, d) not a metric space,
  $\exists i, j \in X \ (d(i,j) \neq \|\delta_i - \delta_j\|_\infty)$
- Issue: *embedding is high-dimensional ($\mathbb{R}^n$)*

[Kuratowski 1935]

Proof

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

$$f(d(i,j)) = \|\delta_i - \delta_j\|_\infty = \max_{k \le n} |d_{ik} - d_{jk}| \le \max_{k \le n} |d_{ij}| = d_{ij}$$

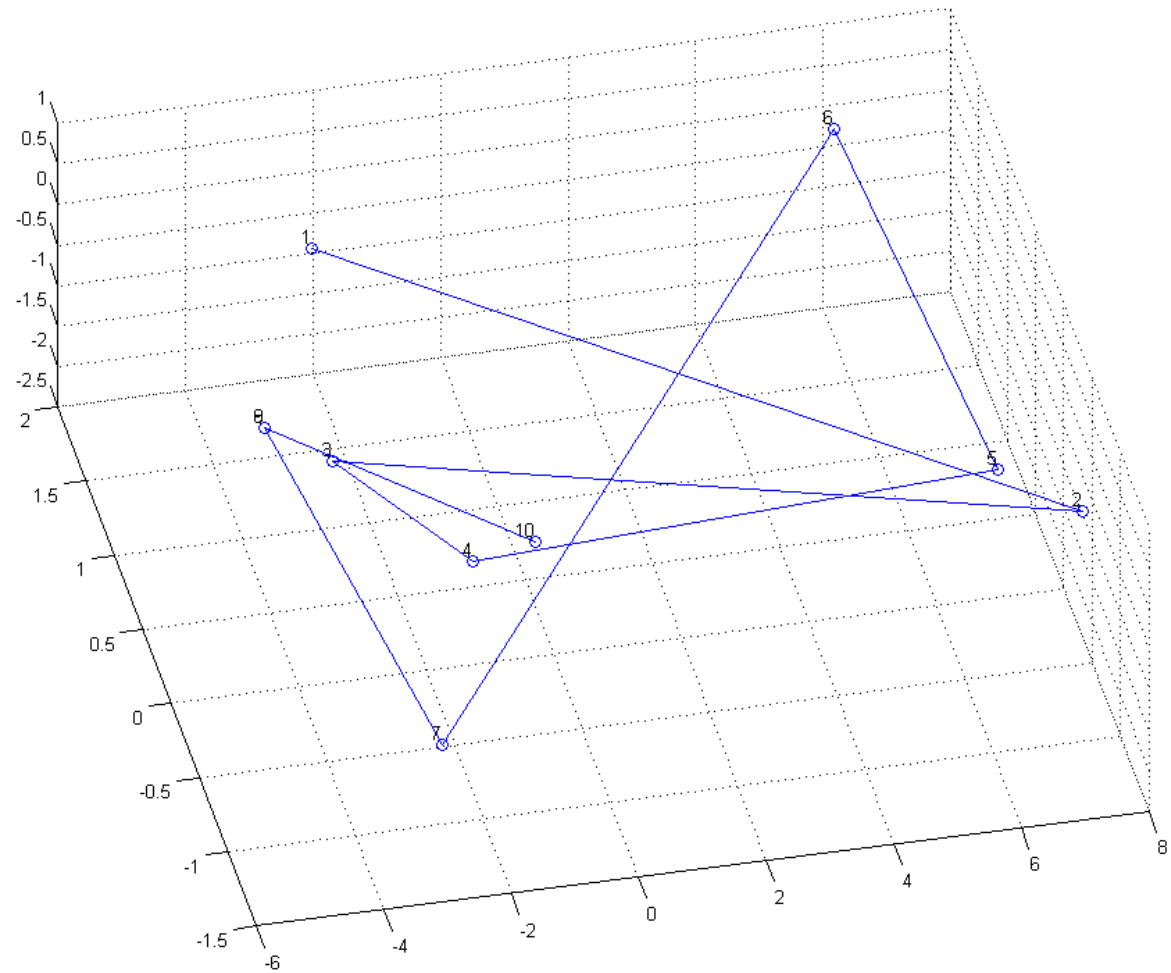
- $\max |d_{ik} - d_{jk}|$ over $k \le n$ is achieved when

$$k \in \{i, j\} \Rightarrow f(d(i,j)) = d_{ij}$$

Genealogical similarity example

Fréchet embedding

in the 3 most significant (rotated) dimensions



Subsection 2

Approximate embeddings in ℓ_2

Lower dimensional (approximate) embeddings

- Given distance matrix, find approximate Euclidean embedding
- Application: visualize a metric space
  e.g. embed genealogy tree in $\mathbb{R}^3$ *(some errors allowed)*
- For visualization purposes, $K \in \{1, 2, 3\}$
  for other purposes, $K < n$

Classical methods

- Multi-Dimensional Scaling (MDS)
- Principal Component Analysis (PCA)

Classic Multidimensional Scaling

Definition and history

- [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: Given $n \times n$ symmetric matrix D, what are necessary and sufficient conditions s.t. D is a EDM[1] corresponding to n points $x_1, \ldots, x_n \in \mathbb{R}^K$ with minimum K?
- Main theorem:
  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
- PSD: *positive semidefinite*, all eigenvalues ≥ 0

[1] *Euclidean Distance Matrix*

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
- Schoenberg's theorem: *relation between EDMs and Gram matrices*

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

- $D^2 = (d_{ij}^2)$, $J = I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$

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 be closest PSD diagonal matrix to $\tilde{\Lambda}$:
  zero the negative components of $\tilde{\Lambda}$
- $x = P\sqrt{\Lambda}$ is an "approximate realization" of $\tilde{D}$

Classic MDS: Main result

1. Prove Schoenberg's theorem: $G = -\frac{1}{2} J D^2 J$
2. Prove matrix is Gram iff it is PSD

Classic MDS: Proof 1/3

- 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:

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

[Borg 2010]

Classic MDS: Proof 2/3

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

$$2x_i x_j = x_i x_i + x_j x_j - d_{ij}^2 \quad (1)$$

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

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

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

$$2x_i x_j = \frac{1}{n}\sum_k d_{ik}^2 + \frac{1}{n} d_{kj}^2 - d_{ij}^2 - \frac{2}{n}\sum_k x_k x_k$$

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

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

which expresses $x_i x_j$ in function of D

Classic MDS: Proof 3/3

- *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

Principal Component Analysis

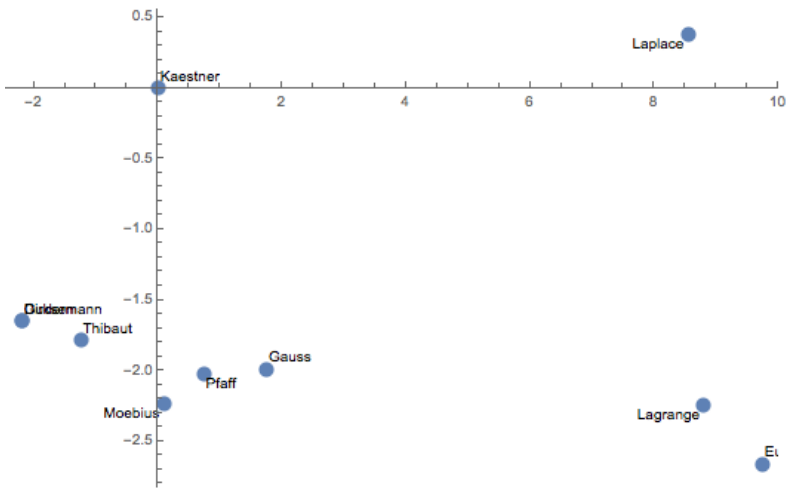
Principal Component Analysis (PCA)

- *MDS with fixed K*
- Motivation: "draw" $x = P\sqrt{\Lambda}$ in 2D or 3D
  but $\mathsf{rank}(\Lambda) = K > 3$
- Only keep 2 or 3 largest components of Λ
  zero the rest
- Get realization in desired space

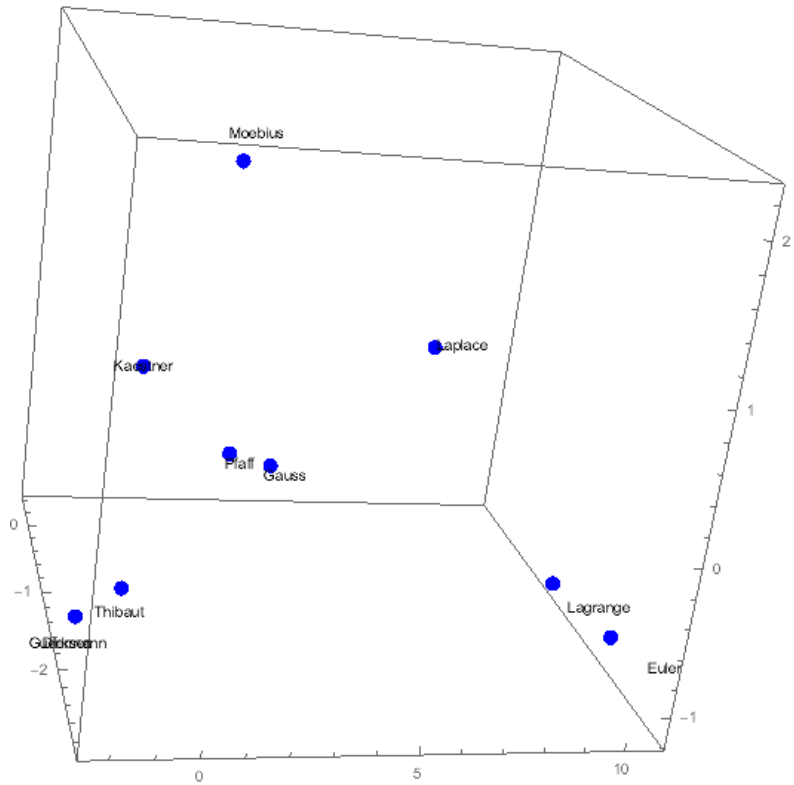
[Pearson 1901]

Mathematicians genealogy in 2D/3D

In 2D



In 3D



Outline

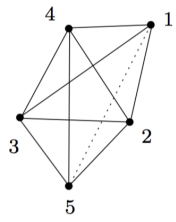
Distance Geometry

Embedding weighted graphs in Euclidean spaces

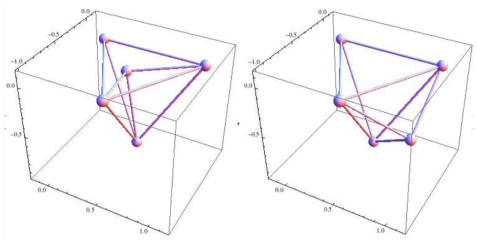
Distance Geometry Problem (DGP)

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.*

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

Given a weighted graph 

, draw it so edges are drawn as segments

with lengths = weights 

[Cayley 1841, Menger 1928, Schoenberg 1935, Yemini 1978]

Subsection 1

DGP 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)

Clock synchronization

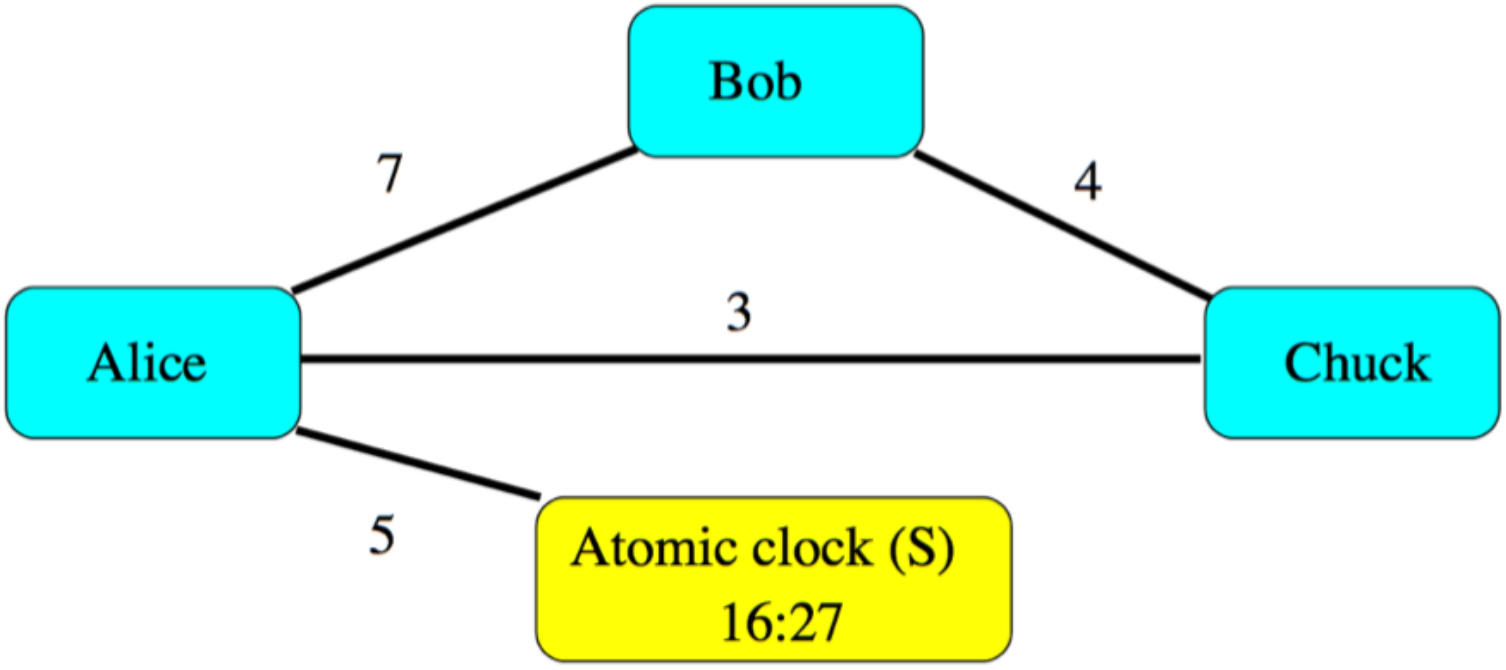
From [Singer, *Appl. Comput. Harmon. Anal.* 2011]

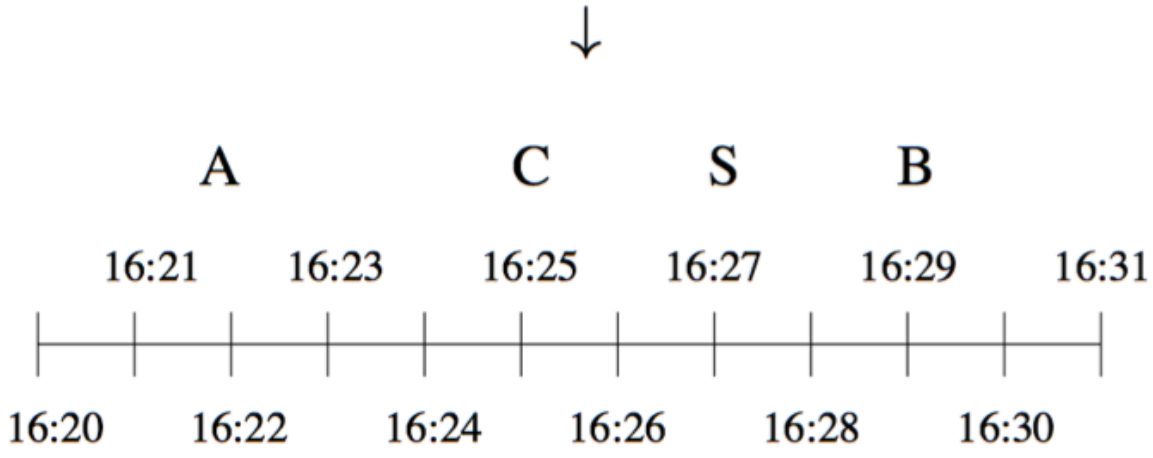
Determine a set of unknown timestamps from a partial measurements of their time differences

- $K = 1$
- 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]

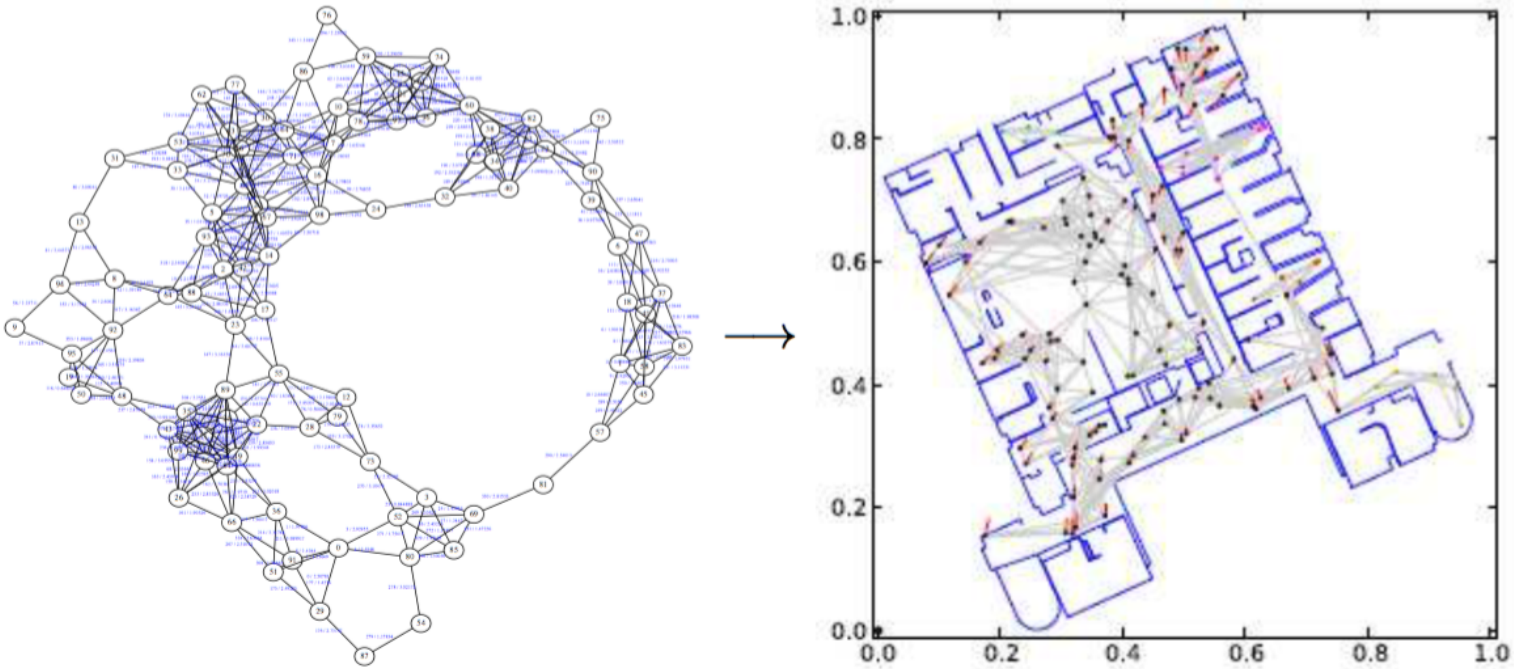
The positioning problem arises when it is necessary to locate a set of geographically distributed objects using measurements of the distances between some object pairs

- $K = 2$
- 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)

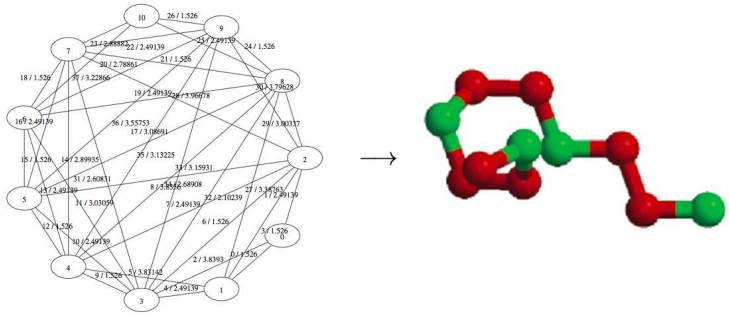
$d_{uv} \propto$ battery consumption in P2P communication betw. u, v

Sensor network localization



Molecular structure from distance data

From [L. 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

Subsection 2

Complexity of the DGP

Complexity class

- 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 $\ell = (\text{length sh. path } p : i \to j) = \sum_{\{u,v\} \in p} d_{uv} \in \mathbb{Q}$
  - 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 verify $\|x_i - x_j\|_2 = d_{ij}$ for $x \notin \mathbb{Q}^{nK}$

Hardness

- Want to show DGP_1 is **NP**-hard by reduction from PARTITION

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

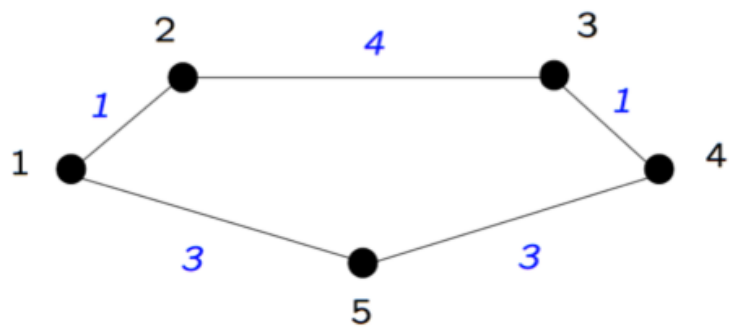
- $a \longrightarrow$ cycle C

  $V(C) = \{1, \ldots, n\}$, $E(C) = \{\{1, 2\}, \ldots, \{n, 1\}\}$

- 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 (1/2)

- 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 $x_{n+1} = x_1$

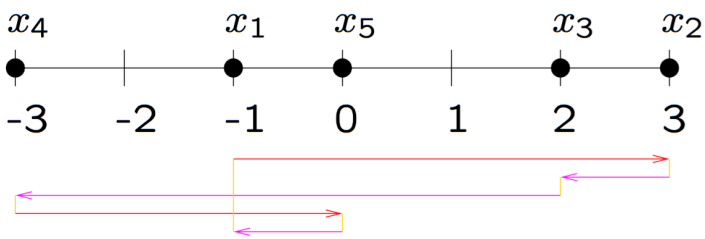
PARTITION is YES $\Rightarrow$ DGP$_1$ is YES (2/2)

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

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

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; since $\in$ **NP**, DGP$_1$ is also NP-complete

Subsection 3

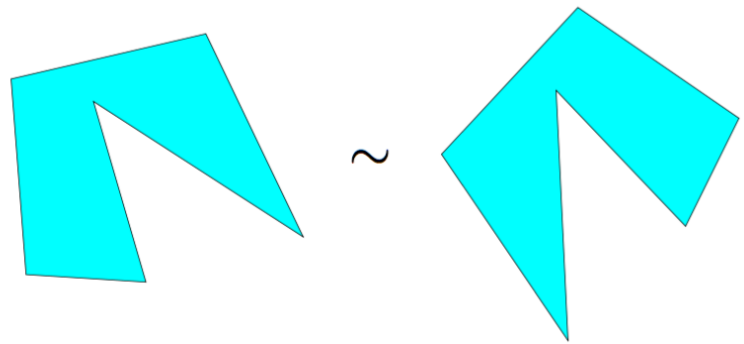
Number of solutions

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)



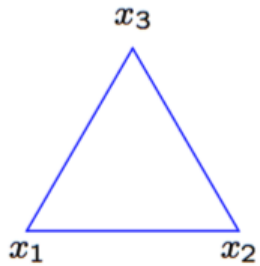
- Partitions $\tilde{X}$ into *equivalence classes*
- $X = \tilde{X}/\sim$: sets of representatives of equivalence classes
- Focus on $|X|$ rather than $|\tilde{X}|$

Examples

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

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

$d^1 = 1$



ρ congruence in $\mathbb{R}^2$

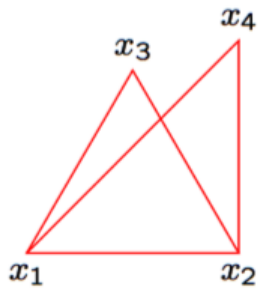
$\Rightarrow \rho x$ valid realization

$|X| = 1$

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

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

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



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

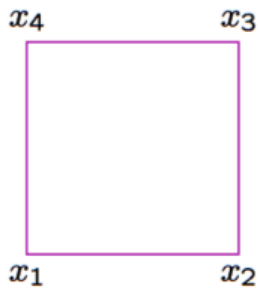
$\Rightarrow \rho x$ valid realization

$|X| = 2$ ()

$V^3 = V^2$

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

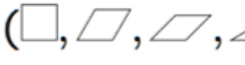
$d^1 = 1$



ρ rotates $\overline{x_2 x_3}$, $\overline{x_1 x_4}$ by θ

$\Rightarrow \rho x$ valid realization

$|X|$ is uncountable

()

Rigidity, flexibility and $|X|$

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

[Milnor 1964, L. et al. 2013]

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]
- If $|X|$ is countably ∞ then G cannot be flexible
  $\Rightarrow$ *incongruent elts of X are separate connected components*
  $\Rightarrow$ by Milnor's theorem, there's finitely many of them

Subsection 4

MP based solution methods

Direct methods

System of quadratic equations

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

Computationally: useless

(less than 10 vertices with $K = 3$ using Octave)

Unconstrained Global Optimization

$$\min_x \sum_{\{u,v\} \in E} (\|x_u - x_v\|^2 - d_{uv}^2)^2 \tag{5}$$

Globally optimal obj. fun. value of (5) is 0 iff x solves (4)

Computational experiments in [L. et al., 2006]*:*

- GO solvers from 10 years ago
- randomly generated protein data: ≤ 50 atoms
- cubic crystallographic grids: ≤ 64 atoms

Constrained global optimization

- $\min_x \sum_{\{u,v\}\in E} |\|x_u - x_v\|^2 - d_{uv}^2|$ 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*

$$\left.\begin{array}{rl} \max_x & \sum_{\{u,v\}\in E} \|x_u - x_v\|^2 \\ \forall \{u,v\} \in E & \|x_u - x_v\|^2 \leq d_{uv}^2 \end{array}\right\} \tag{6}$$

Theorem (Activity)

At a glob. opt. x^* of a YES instance, all constraints of (6) are active

[Mencarelli et al. 2017]

Semidefinite Programming

Linearization

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

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

$$\Rightarrow \quad \begin{cases} \forall \{i,j\} \in E & X_{ii} + X_{jj} - 2X_{ij} = d_{ij}^2 \\ & X = x\,x^\top \end{cases}$$

$$X = x\,x^\top \Leftrightarrow \forall i,j\ X_{ij} = x_i x_j$$

Relaxation

$$\begin{aligned} X &= x\,x^\top \\ \Rightarrow \quad X - x\,x^\top &= 0 \end{aligned}$$

$$\begin{aligned} \text{(relax)} \quad \Rightarrow \quad X - x\,x^\top &\succeq 0 \end{aligned}$$

$$\mathsf{Schur}(X,x) = \begin{pmatrix} I_K & x^\top \\ x & X \end{pmatrix} \succeq 0$$

If x does not appear elsewhere $\Rightarrow$ get rid of it (e.g. choose $x = 0$):

replace $\mathsf{Schur}(X,x) \succeq 0$ *by* $X \succeq 0$

Reason for this weird relaxation: there are efficient solvers for Semidefinite Programming (SDP)

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?

Some possible objective functions

- For protein conformation:

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

with $=$ changed to $\leq$ in constraints *(or* min *and* $\geq$*)* [Dias & L. 2016]
"push-and-pull" the realization

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

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

improve covariance estimator accuracy

- *How about "just random"?* [Barvinok 1995]

Computational evaluation

- 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:

  - $\mathrm{LDE}(x) = \max\limits_{\{i,j\}\in E} \left| \|x_i - x_j\| - d_{ij} \right|$
  - $\mathrm{MDE}(x) = \frac{1}{|E|} \sum\limits_{\{i,j\}\in E} \left| \|x_i - x_j\| - d_{ij} \right|$

Objective function tests

SDP solved with Mosek

SDP + PCA

Instance			LDE			MDE			CPU		
Name	$\|V\|$	$\|E\|$	*PP*	*Ye*	*Rnd*	*PP*	*Ye*	*Rnd*	*PP*	*Ye*	*Rnd*
C0700odd.1	15	39	3.31	4.57	4.44	1.92	2.52	2.50	0.13	0.07	0.08
C0700odd.C	36	242	10.61	4.85	4.85	3.02	3.02	3.02	0.69	0.43	0.44
C0700.odd.G	36	308	4.57	4.77	4.77	2.41	2.84	2.84	0.86	0.54	0.54
C0150alter.1	37	335	4.66	4.88	4.86	2.52	3.00	3.00	0.97	0.59	0.58
C0080create.1	60	681	7.17	4.86	4.86	3.08	3.19	3.19	2.48	1.46	1.46
tiny	37	335	4.66	4.88	4.88	2.52	3.00	3.00	0.97	0.60	0.60
1guu-1	150	959	10.20	4.93	4.93	3.43	3.43	3.43	9.23	5.68	5.70

SDP + PCA + NLP

Instance			LDE			MDE			CPU		
Name	$\|V\|$	$\|E\|$	*PP*	*Ye*	*Rnd*	*PP*	*Ye*	*Rnd*	*PP*	*Ye*	*Rnd*
1b03	89	456	0.00	0.00	0.00	0.00	0.00	0.00	8.69	6.28	9.91
1crn	138	846	0.81	0.81	0.81	0.07	0.07	0.07	33.33	31.32	44.48
1guu-1	150	959	0.97	4.93	0.92	0.10	3.43	0.08	56.45	7.89	65.33

[Dias & L., 2016]

Empirical considerations

- *Ye* very fast but often imprecise
- *Random* good but nondeterministic
- *Push-and-Pull* relaxes $X_{ii} + X_{jj} - 2X_{ij} = d_{ij}^2$ to $X_{ii} + X_{jj} - 2X_{ij} \geq d_{ij}^2$, easier to satisfy feasibility

...will be useful in DDP later on

Focus on Push-and-Pull objective

Diagonally Dominant Programming

When SDP solvers hit their size limit

- SDP solver: technological bottleneck
- How can we best use an LP solver?
- Diagonally Dominant (DD) matrices are PSD
- Not *vice versa*: inner approximate PSD cone $Y \succeq 0$
- *Idea by A.A. Ahmadi and co-authors*

[Ahmadi & Majumdar 2014, Ahmadi & Hall 2015]

Diagonally dominant matrices

$n \times n$ matrix X is DD if

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

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

DD Linearization

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

- introduce "sandwiching" variable T
- write $|X|$ as T
- add constraints $-T \leq X \leq T$
- by $\geq$ constraint sense, write $(*)$ as

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

DD Programming (DDP)

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

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

DDP formulation for the DGP

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

[Dias & L., 2016]

SDP vs. DDP: tests

Using "push-and-pull" objective in SDP
SDP solved with Mosek, DDP with CPLEX

SDP/DDP + PCA

	SDP			DDP		
Instance	*LDE*	*MDE*	*CPU modl/soln*	*LDE*	*MDE*	*CPU modl/soln*
`C0700odd.1`	0.79	0.34	0.06/0.12	0.38	0.30	0.15/0.15
`C0700.odd.G`	2.38	0.89	0.57/1.16	1.86	0.58	1.11/0.95
`C0150alter.1`	1.48	0.45	0.73/1.33	1.54	0.55	1.23/1.04
`C0080create.1`	2.49	0.82	1.63/7.86	0.98	0.67	3.39/4.07
`1guu-1`	0.50	0.15	6.67/684.89	1.00	0.85	37.74/153.17

Subsection 5

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.

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

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

Barvinok's theorem

Consider:

- for each $k \leq m$, manifolds $\mathcal{X}_k = \{x \in \mathbb{R}^n \mid x^\top Q^k x = a_k\}$
- a feasibility problem $x \in \bigcap_{k \leq m} \mathcal{X}_k$
- its SDP relaxation $\forall x \leq m \ (Q^k \bullet X = a_k)$ with soln. $\bar{X}$

Let $\boxed{T = \mathsf{factor}(\bar{X})}$, $\boxed{y \sim \mathcal{N}^n(0,1) \text{ and } x' = Ty}$

Then $\exists c$ and $n_0 \in \mathbb{N}$ s.t. if $n \geq n_0$,

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

IDEA: since x' is "close" to each $\mathcal{X}_k$
try local Nonlinear Programming (NLP)

Application to the DGP

- $\forall \{i,j\} \in E \quad \mathcal{X}_{ij} = \{x \in \mathbb{R}^{nK} \mid \|x_i - x_j\|_2^2 = d_{ij}^2\}$
- DGP can be written as $\bigcap_{\{i,j\} \in E} \mathcal{X}_{ij}$
- 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 $y \sim \mathcal{N}^{nK}(0, \frac{1}{\sqrt{K}})$
- Thm. Barvinok's theorem works in rank K

[L. & Vu, unpublished]

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. $y \sim \mathcal{N}^{nK}(0, \frac{1}{\sqrt{K}})$
   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

[Dias & L., 2016]

SDP+Barvinok vs. DDP+Barvinok

	SDP			DDP		
Instance	*LDE*	*MDE*	*CPU*	*LDE*	*MDE*	*CPU*
C0700odd.1	0.00	0.00	0.63	0.00	0.00	1.49
C0700.odd.G	0.00	0.00	21.67	0.42	0.01	30.51
C0150alter.1	0.00	0.00	29.30	0.00	0.00	34.13
C0080create.1	0.00	0.00	139.52	0.00	0.00	141.49
1b03	0.18	0.01	132.16	0.38	0.05	101.04
1crn	0.78	0.02	800.67	0.76	0.04	522.60
1guu-1	0.79	0.01	1900.48	0.90	0.04	667.03

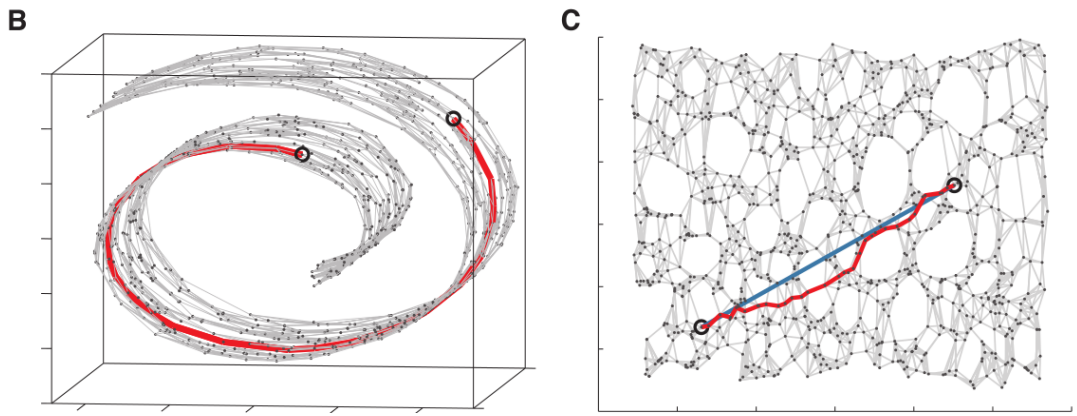
Most of the CPU time taken by local NLP solver

Subsection 6

Isomap for the DGP

Isomap for DG

1. Let D' be the (square) weighted adjacency matrix of G
2. Complete D' to *approximate* EDM $\tilde{D}$
3. MDS/PCA on $\tilde{D} \Rightarrow$ obtain embedding $x \in \mathbb{R}^K$ *for given* K



Vary Step 2 to generate Isomap heuristics

[Tenenbaum et al. 2000, L. & D'Ambrosio 2017]

Variants for Step 2

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

B. Find a spanning tree (SPT) of G, compute any embedding $\bar{x} \in \mathbb{R}^K$ for STP, use its EDM

C. Solve a push-and-pull SDP relaxation, get soln. $\bar{x} \in \mathbb{R}^n$, use its EDM

D. Solve an SDP relaxation with "Barvinok objective", find $\bar{x} \in \mathbb{R}^r$ (with $r \leq \lfloor (\sqrt{8|E|+1} - 1)/2 \rfloor$), use its EDM
haven't really talked about this, sorry

Post-processing: $\tilde{x}$ *as starting point for NLP descent in GO formulation*

[L. & D'Ambrosio 2017]

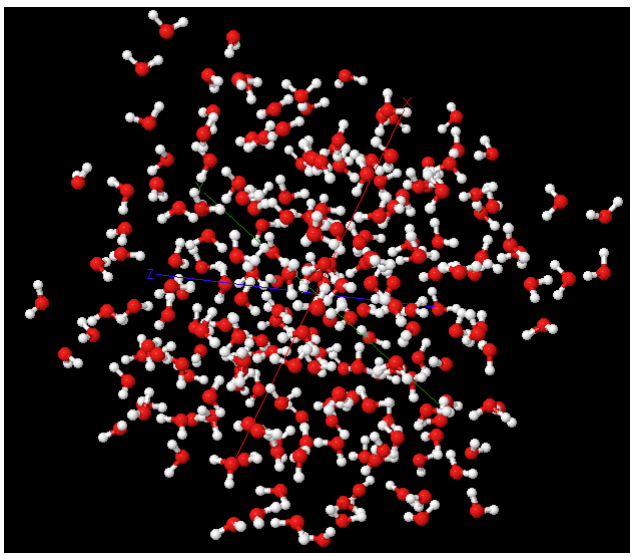
Results

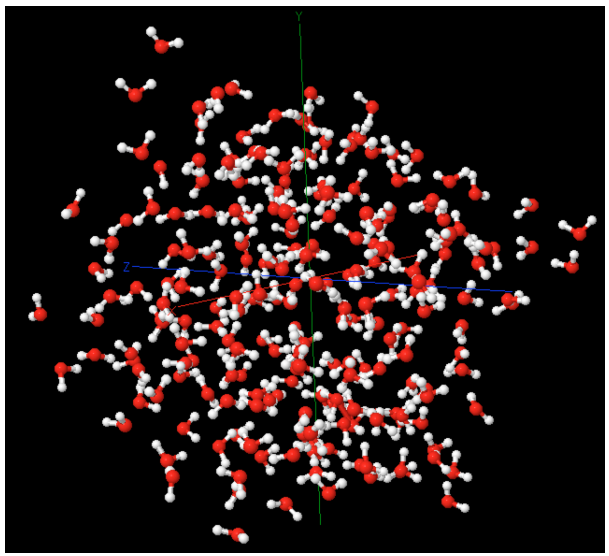
Comparison with `dgsol` [Moré, Wu 1997]

Instance			mde						lde						CPU					
Name	n	$\|E\|$	Isomap	IsoNLP	SPT	SDP	Barvinok	DGSol	Isomap	IsoNLP	SPT	SDP	Barvinok	DGSol	Isomap	IsoNLP	SPT	SDP	Barvinok	DGSol
C0700odd.1	15	39	0.585	0.001	0.190	0.068	**0.000**	0.135	0.989	0.004	0.896	0.389	**0.001**	0.634	**0.002**	1.456	1.589	0.906	1.305	1.747
C0700odd.2	15	39	0.599	**0.000**	0.187	0.086	**0.000**	0.128	0.985	**0.002**	0.956	0.389	0.009	1.000	**0.003**	1.376	1.226	1.002	1.063	0.887
C0700odd.3	15	39	0.599	**0.000**	0.060	0.086	**0.000**	0.128	0.985	**0.002**	0.326	0.389	0.009	1.000	**0.003**	1.259	1.256	0.861	1.167	0.877
C0700odd.4	15	39	0.599	**0.000**	0.283	0.086	0.001	0.128	0.985	**0.002**	2.449	0.389	0.008	1.000	**0.003**	1.347	1.222	0.976	1.063	1.033
C0700odd.5	15	39	0.599	**0.000**	0.225	0.086	0.000	0.128	0.985	**0.002**	0.867	0.389	0.007	1.000	**0.003**	1.284	1.157	0.987	1.100	0.700
C0700odd.6	15	39	0.599	**0.000**	0.283	0.086	0.000	0.128	0.985	**0.002**	1.520	0.389	**0.002**	1.000	**0.002**	1.372	1.196	0.998	1.305	0.909
C0700odd.7	15	39	0.585	0.001	0.080	0.068	**0.000**	0.135	0.989	0.004	0.361	0.389	**0.001**	0.634	**0.003**	1.469	1.322	0.894	1.093	1.719
C0700odd.8	15	39	0.585	0.001	0.056	0.068	**0.000**	0.135	0.989	0.004	0.275	0.389	**0.003**	0.634	**0.003**	1.408	1.306	0.692	1.079	1.744
C0700odd.9	15	39	0.585	0.001	0.057	0.068	**0.000**	0.135	0.989	0.004	0.301	0.389	**0.002**	0.634	**0.002**	1.430	1.172	0.791	1.093	1.745
C0700odd.A	15	39	0.585	0.001	0.043	0.068	**0.000**	0.135	0.989	**0.004**	0.316	0.389	**0.004**	0.634	**0.002**	1.294	1.269	0.722	1.220	1.523
C0700odd.B	15	39	0.585	0.001	0.151	0.068	**0.000**	0.135	0.989	**0.004**	1.022	0.389	**0.004**	0.634	**0.002**	1.297	1.279	0.871	1.111	1.747
C0700odd.C	15	39	0.835	**0.022**	0.033	0.039	0.031	0.025	1.012	**0.147**	0.393	0.211	0.294	0.167	**0.004**	6.803	6.369	7.371	7.030	7.000
C0700odd.D	36	242	0.835	**0.022**	0.041	0.039	0.042	0.025	1.012	**0.147**	0.423	0.211	0.268	0.167	**0.006**	6.806	6.575	7.422	7.603	7.095
C0700odd.E	36	242	0.835	**0.022**	0.064	0.039	0.031	0.025	1.012	**0.147**	0.894	0.211	0.260	0.167	**0.006**	6.911	6.638	7.365	6.979	7.008
C0700odd.F	36	242	0.599	**0.000**	0.047	0.086	**0.000**	0.128	0.985	**0.002**	0.308	0.389	0.005	1.000	**0.002**	1.299	1.310	1.008	1.100	1.040
C0150alter.1	37	335	0.786	0.058	0.066	0.014	0.015	**0.010**	0.992	0.571	0.693	0.256	0.285	**0.253**	**0.004**	9.492	9.456	10.276	10.120	9.272
C0080create.1	60	681	0.887	0.053	0.083	**0.024**	**0.024**	0.054	1.967	0.949	0.789	**0.511**	0.516	0.718	**0.012**	18.835	19.720	21.247	20.906	19.962
C0080create.2	60	681	0.887	0.053	0.047	**0.024**	**0.024**	0.054	1.967	0.949	0.585	**0.511**	0.512	0.718	**0.008**	18.791	20.009	21.728	20.885	19.740
C0020pdb	107	999	0.939	0.110	0.119	**0.059**	0.060	0.103	1.242	1.113	1.349	1.082	1.138	**0.798**	**0.035**	29.024	27.772	35.273	35.486	32.479
1guu	150	955	0.986	0.068	0.069	**0.057**	**0.057**	0.061	0.999	0.854	0.830	**0.735**	0.751	0.768	**0.048**	30.869	28.784	41.488	41.852	37.848
1guu-1	150	959	0.986	0.061	0.063	0.058	**0.057**	0.060	1.000	**0.711**	0.855	0.805	0.829	0.778	**0.053**	31.322	31.442	42.308	41.590	37.218
1guu-4000	150	968	0.974	0.081	0.080	0.072	**0.065**	0.079	1.000	0.901	**0.728**	0.760	0.961	0.826	**0.050**	30.352	29.856	42.330	39.832	42.015
C0030pkl	198	3247	0.961	0.112	0.160	**0.076**	0.077	0.137	**1.197**	1.354	2.230	1.995	2.054	1.401	**0.091**	105.175	104.775	149.192	146.360	111.859
1PPT	302	3102	0.984	**0.121**	0.129	0.128	0.129	0.123	**1.000**	1.519	1.219	1.944	1.956	1.224	**0.356**	112.448	110.345	185.815	187.182	118.681
100d	488	5741	0.987	0.146	0.146	0.155	0.157	**0.137**	**1.000**	1.577	1.397	1.764	1.749	1.358	**0.828**	229.809	213.136	659.638	659.280	233.115
GeoMean			0.74	**0.00**	0.09	0.06	**0.00**	0.08	1.07	**0.04**	0.73	0.50	0.06	0.66	**0.01**	6.30	6.04	5.93	6.63	6.30
Avg			0.76	0.04	0.11	0.07	**0.03**	0.10	1.09	**0.44**	0.88	0.63	0.47	0.77	**0.06**	26.12	25.21	49.69	49.55	27.96
StDev			0.17	0.05	0.07	**0.03**	0.04	0.04	**0.27**	0.55	0.57	0.52	0.65	0.34	**0.18**	51.69	48.82	135.08	134.97	53.26

Large instances

Instance			mde		lde		CPU	
Name	$\|V\|$	$\|E\|$	IsoNLP	dgsol	IsoNLP	dgsol	IsoNLP	dgsol
water	648	11939	0.005	0.15	0.557	0.81	26.98	15.16
3al1	678	17417	0.036	0.007	0.884	0.810	170.91	210.25
1hpv	1629	18512	0.074	0.078	0.936	0.932	374.01	60.28
il2	2084	45251	0.012	0.035	0.910	0.932	465.10	139.77
1tii	5684	69800	0.078	0.077	0.950	0.897	7400.48	454.375





Outline

Distance resolution limit

Clustering in high dimensions is unstable

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

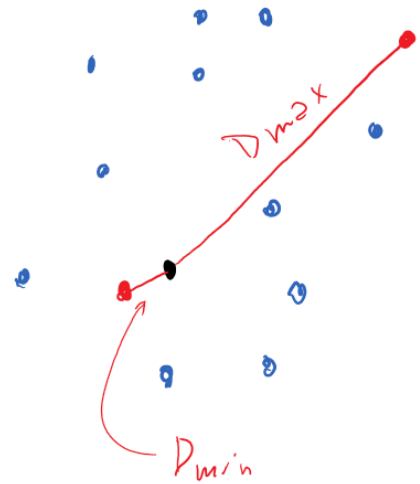
(b) $\forall y \in \mathcal{Y}, x \in \mathcal{X} \quad (d(z, y) \leq d(z, x))$

- 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
  - $D_{\text{min}}^m = \min_{i \leq \ell} d^m(Z^m, X_i^m)$
  - $D_{\text{max}}^m = \max_{i \leq \ell} d^m(Z^m, X_i^m)$

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

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

then, for any $\varepsilon > 0$,

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

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 $\left\{\frac{A^m}{B^m}\right\}_m \to_{\mathbb{P}} \frac{a}{b}$

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. $\mathbf{D}^m = ((d^m(Z^m, X_i^m))^p \mid i \leq \ell) \to_{\mathbb{P}} \mathbf{1}$ (X_i^m iid)
4. *Slutsky's thm.* $\Rightarrow \min(\mathbf{D}^m) \to_{\mathbb{P}} \min(\mathbf{1}) = 1$, simy for max
5. *Corollary* $\Rightarrow \frac{\max(\mathbf{D}^m)}{\min(\mathbf{D}^m)} \to_{\mathbb{P}} 1$
6. $\frac{D^m_{\max}}{D^m_{\min}} = \frac{\mu_m \max(\mathbf{D}^m)}{\mu_m \min(\mathbf{D}^m)} \to_{\mathbb{P}} 1$
7. Result follows (defn. of $\to_{\mathbb{P}}$ and $D^m_{\max} \geq D^m_{\min}$)

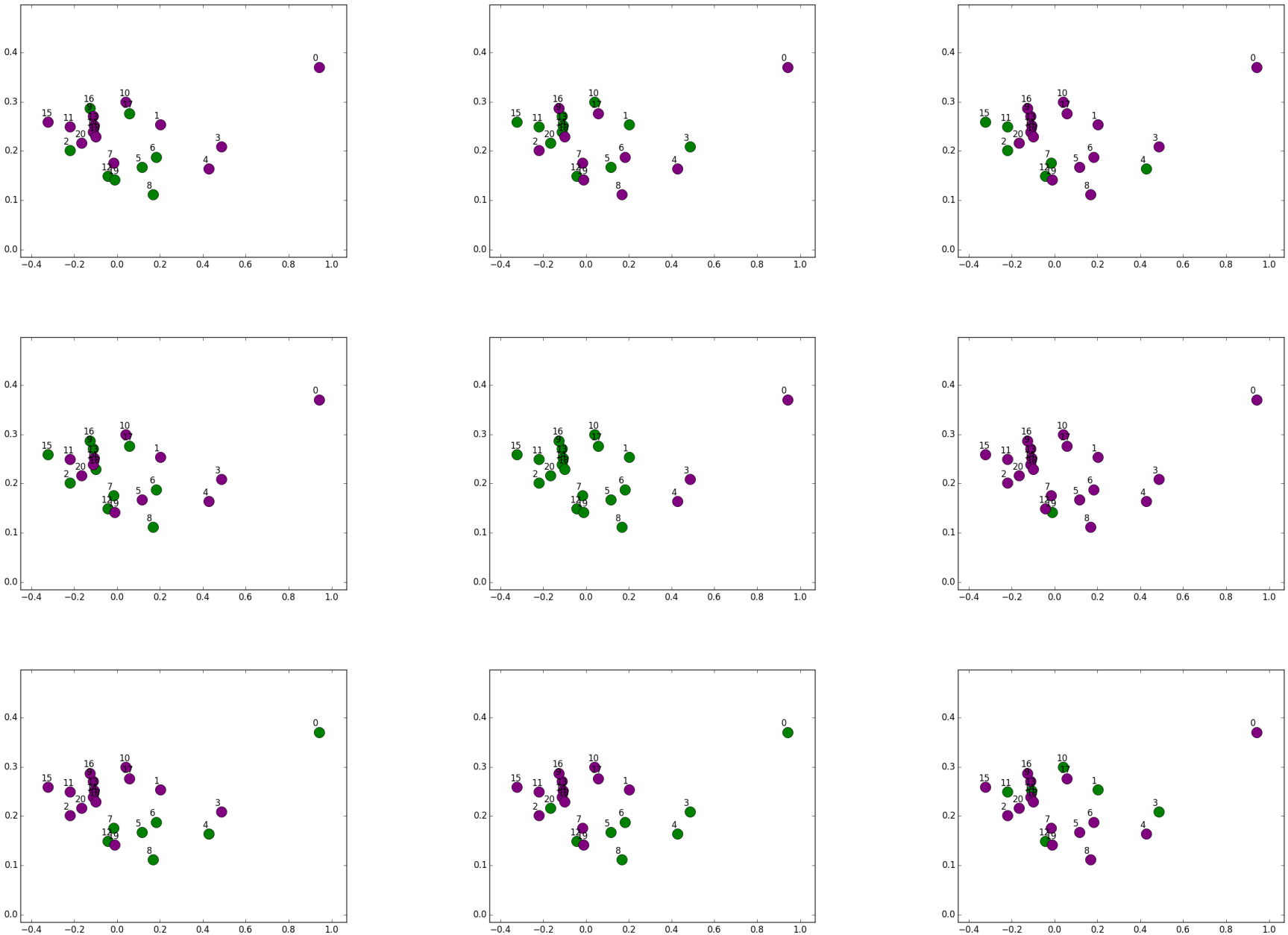
Subsection 1

When to start worrying

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: 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.

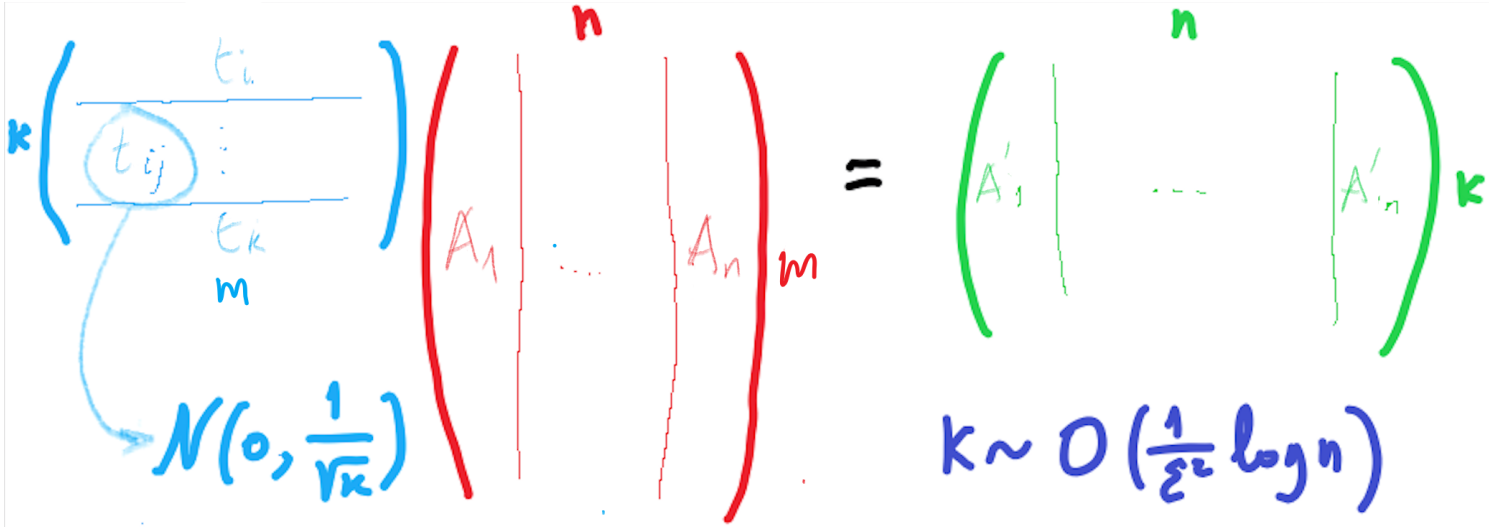
Outline

Random projections

The mathematics of big data

The magic of random projections

- *"Mathematics of big data"*
- In a nutshell



- *Clustering on* $\boxed{A'}$ *rather than* A
  yields approx. same results with arbitrarily high probability (wahp)

[Johnson & Lindenstrauss, 1984]

The magic of random projections

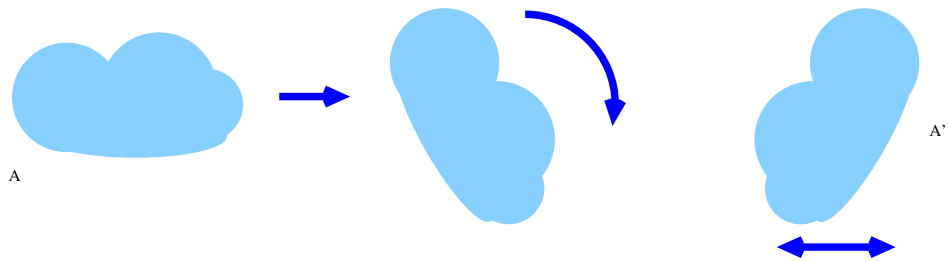
- *"Mathematics of big data"*
- In a nutshell
  1. Given points $A_i, \ldots, A_n \in \mathbb{R}^m$ with m large and $\varepsilon \in (0,1)$
  2. Pick "appropriate" $k \approx O(\frac{1}{\varepsilon^2} \ln n)$
  3. Sample $k \times d$ matrix T (*each comp. i.i.d.* $\mathcal{N}(0, \frac{1}{\sqrt{k}})$)
  4. Consider *projected* points $A'_i = TA_i \in \mathbb{R}^k$ for $i \leq n$
  5. With prob$\to 1$ exponentially fast as $k \to \infty$

$$\forall i,j \leq n \quad (1-\varepsilon)\|A_i - A_j\|_2 \leq \|A'_i - A'_j\|_2 \leq (1+\varepsilon)\|A_i - A_j\|_2$$

[Johnson & Lindenstrauss, 1984]

The shape of a set of points

- Lose dimensions but not too much accuracy
  Given $A_1, \ldots, A_n \in \mathbb{R}^m$ find $k \ll m$ and points
  $A'_1, \ldots, A'_n \in \mathbb{R}^k$ s.t. A and A' "have almost the same shape"
- What is the shape of a set of points?



congruent sets have the same shape

- Approximate congruence $\Leftrightarrow$ distortion:
  A, A' have almost the same shape if

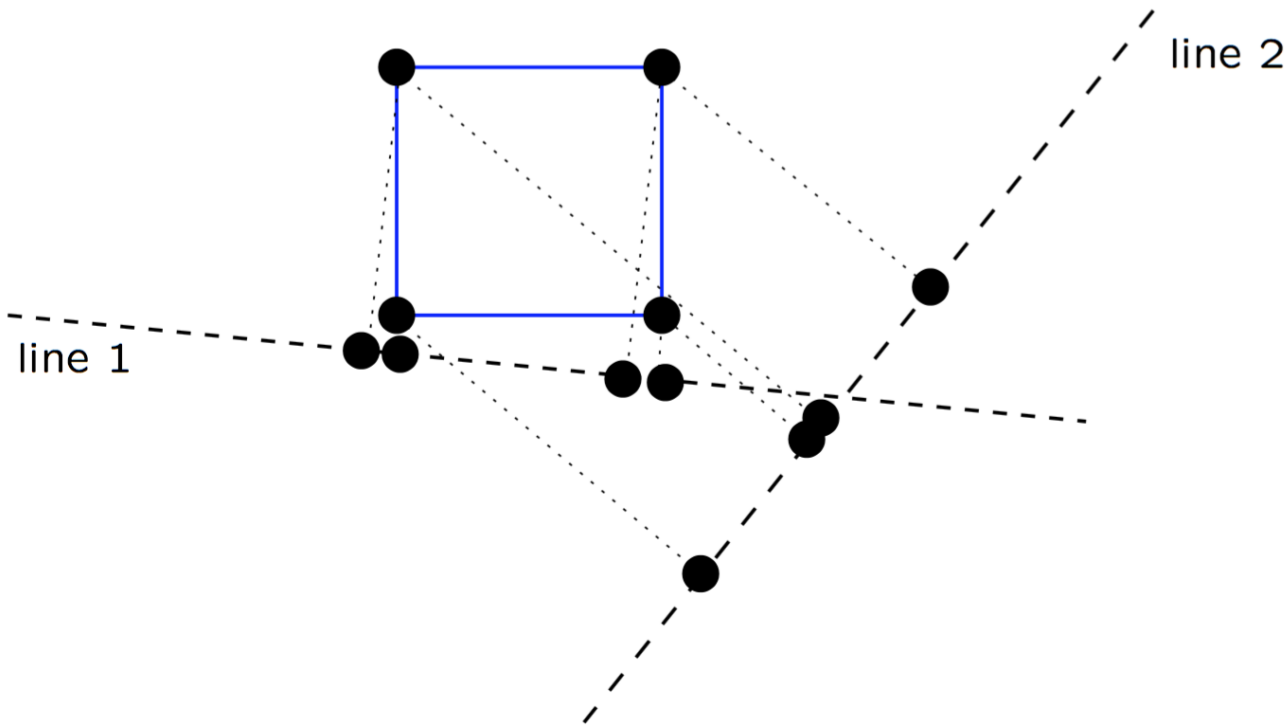
$$\forall i < j \leq n \quad (1-\varepsilon)\|A_i - A_j\| \;\leq\; \|A'_i - A'_j\| \;\leq\; (1+\varepsilon)\|A_i - A_j\|$$

  for some small $\varepsilon > 0$

Assume norms are all Euclidean

Losing dimensions = "projection"

In the plane, hopeless



In 3D: no better

Johnson-Lindenstrauss Lemma

Thm.

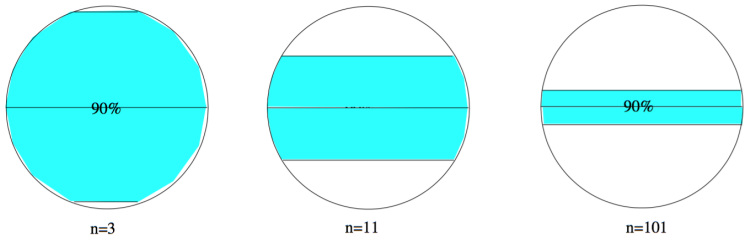
Given $A \subseteq \mathbb{R}^m$ with $|A| = n$ and $\varepsilon > 0$ there is $k \sim O(\frac{1}{\varepsilon^2} \ln n)$ and a $k \times m$ matrix T s.t.

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

If $k \times m$ matrix T is sampled componentwise from $\mathsf{N}(0, \frac{1}{\sqrt{k}})$, then A and TA have almost the same shape

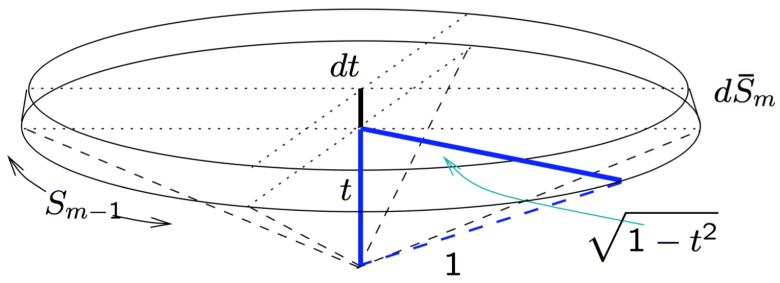
[Johnson & Lindenstrauss, 1984]

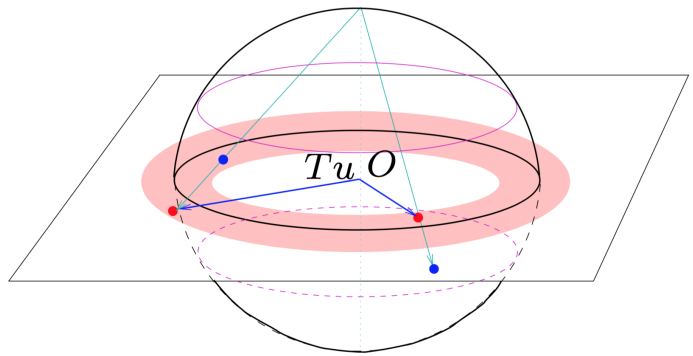
Sketch of a JLL proof by pictures

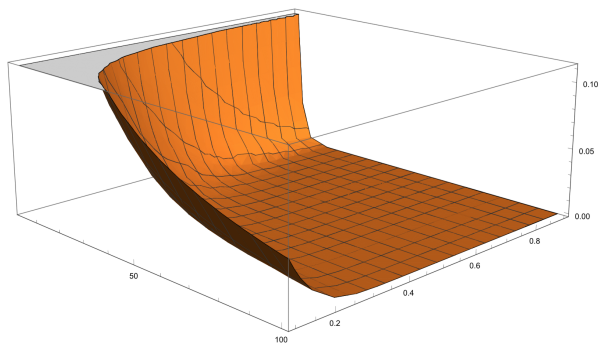


Thm.

Let T be a $k \times m$ rectangular matrix with each component sampled from $\mathsf{N}(0, \frac{1}{\sqrt{k}})$, and $u \in \mathbb{R}^m$ s.t. $\|u\| = 1$. Then $\mathbb{E}(\|Tu\|^2) = 1$







Sampling to desired accuracy

- Distortion has low probability:

$$\forall x, y \in A \quad \mathrm{P}(\|Tx - Ty\| \leq (1-\varepsilon)\|x-y\|) \quad \leq \quad \frac{1}{n^2}$$

$$\forall x, y \in A \quad \mathrm{P}(\|Tx - Ty\| \geq (1+\varepsilon)\|x-y\|) \quad \leq \quad \frac{1}{n^2}$$

- Probability $\exists$ pair $x, y \in A$ distorting Euclidean distance:
  union bound over $\binom{n}{2}$ *pairs*

$$\mathrm{P}(\neg(A \text{ and } TA \text{ have almost the same shape})) \quad \leq \quad \binom{n}{2}\frac{2}{n^2} = 1 - \frac{1}{n}$$

$$\mathrm{P}(A \text{ and } TA \text{ have almost the same shape}) \quad \geq \quad \frac{1}{n}$$

$\Rightarrow$ re-sampling T gives JLL with arbitrarily high probability

[Dasgupta & Gupta, 2002]

In practice

- Empirically, sample T very few times (e.g. once will do!)
  on average $\|Tx - Ty\| \approx \|x - y\|$, *and distortion decreases exponentially with* n

We only need a logarithmic number of dimensions in function of the number of points

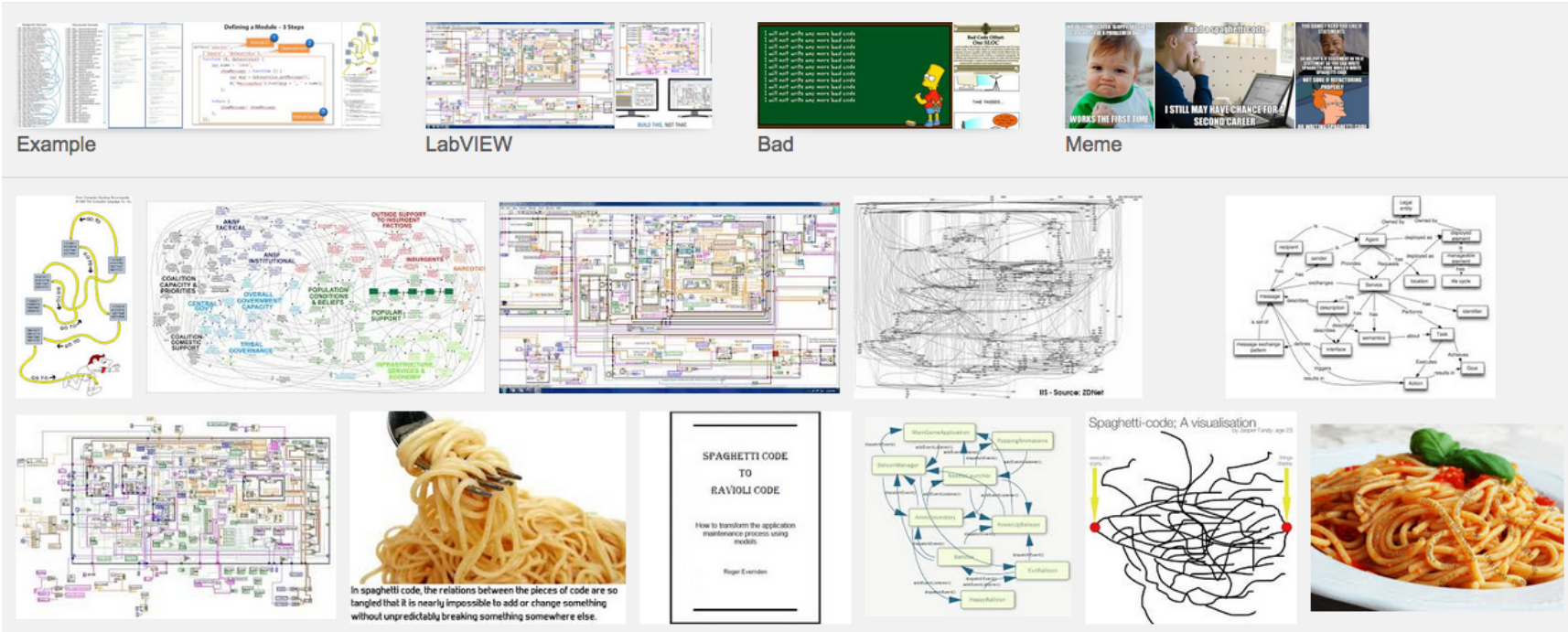
Surprising fact:

k is independent of the original number of dimensions m

Subsection 1

More efficient clustering

Clustering Google images



[L. & Lavor, *in press*]

Recall k-means

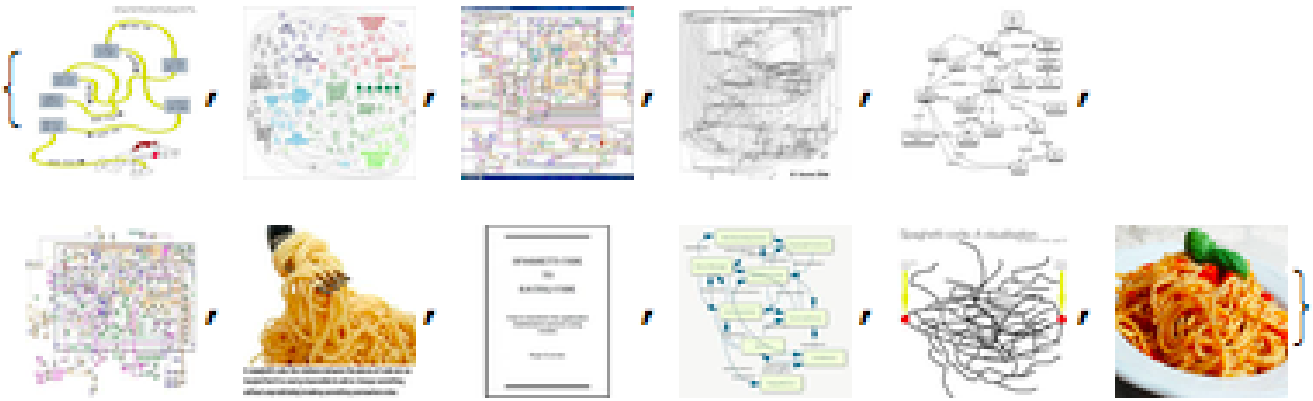
- *Input*: X = set of images (as vectors in $\mathbb{R}^{|\text{pixels}|}$)
- *Output*: a k-partition of X

1. pick k random centroid candidates
2. assign points to closest centroid
3. recompute correct centroid for assigned points
4. repeat from Step 2 until stability

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!

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 0.405 CPU time to 0.00232

Same clustering

Works on the MSSC MP formulation too!

$$
\left.
\begin{array}{lrcl}
\min\limits_{x,y,s} & \sum\limits_{i\le n}\sum\limits_{j\le d} \|Tp_i - Ty_j\|_2^2\, x_{ij} & & \\
\forall j \le d & \frac{1}{s_j}\sum\limits_{i\le n} Tp_i x_{ij} & = & Ty_j \\
\forall i \le n & \sum\limits_{j\le d} x_{ij} & = & 1 \\
\forall j \le d & \sum\limits_{i\le n} x_{ij} & = & s_j \\
\forall j \le d & y_j & \in & \mathbb{R}^m \\
& x & \in & \{0,1\}^{nd} \\
& s & \in & \mathbb{N}^d
\end{array}
\right\}
$$

where T is a $k \times m$ random projector

Works on the MSSC MP formulation too!

$$
\left.
\begin{array}{lrcl}
\min\limits_{x,y',s} & \sum\limits_{i\le n}\sum\limits_{j\le d} \|Tp_i - y'_j\|_2^2\, x_{ij} & & \\
\forall j \le d & \frac{1}{s_j}\sum\limits_{i\le n} Tp_i x_{ij} & = & y'_j \\
\forall i \le n & \sum\limits_{j\le d} x_{ij} & = & 1 \\
\forall j \le d & \sum\limits_{i\le n} x_{ij} & = & s_j \\
\forall j \le d & y'_j & \in & \mathbb{R}^k \\
& x & \in & \{0,1\}^{nd} \\
& s & \in & \mathbb{N}^d
\end{array}
\right\} \quad (\text{MSSC}')
$$

- *where* $k = O(\frac{1}{\varepsilon^2}\ln n)$
- less data, $|y'| < |y| \Rightarrow$ get solutions faster
- Yields smaller cMINLP

Subsection 2

Random projections in Linear Programming

The gist

- Let A, b be very large, consider LP

$$\min\{c^\top x \mid Ax = b \wedge x \geq 0\}$$

- T short & fat normally sampled
- Then

$$Ax = b \wedge x \geq 0 \;\Leftrightarrow\; TAx = Tb \wedge x \geq 0$$

wahp

[Vu & al. *to appear*]

Losing dimensions

Restricted Linear Membership (RLM_X)
Given $A_1, \ldots, A_n, b \in \mathbb{R}^m$ and $X \subseteq \mathbb{R}^n$, $\exists$? $x \in X$ s.t.

$$b = \sum_{i \leq n} x_i A_i$$

Given $X \subseteq \mathbb{R}^n$ and $b, A_1, \ldots, A_n \in \mathbb{R}^m$, find $k \ll m$, $b', A'_1, \ldots, A'_n \in \mathbb{R}^k$ such that:

$$\underbrace{\exists x \in X\ b = \sum_{i \leq n} x_i A_i}_{\textit{high dimensional}} \quad \text{iff} \quad \underbrace{\exists x \in X\ b' = \sum_{i \leq n} x_i A'_i}_{\textit{low dimensional}}$$

with high probability

Projecting feasibility

Projecting infeasibility (easy cases)

Thm.

$T : \mathbb{R}^m \to \mathbb{R}^k$ a JLL random projection, $b, A_1, \ldots, A_n \in \mathbb{R}^m$ a RLM_X instance. For any given vector $x \in X$, we have:

(i) If $b = \sum_{i=1}^{n} x_i A_i$ then $Tb = \sum_{i=1}^{n} x_i T A_i$

(ii) If $b \neq \sum_{i=1}^{n} x_i A_i$ then $\mathrm{P}\left(Tb \neq \sum_{i=1}^{n} x_i T A_i\right) \geq 1 - 2e^{-\mathcal{C}k}$

(iii) If $b \neq \sum_{i=1}^{n} y_i A_i$ for all $y \in X \subseteq \mathbb{R}^n$, where $|X|$ is finite, then

$$\mathrm{P}\left(\forall y \in X\, Tb \neq \sum_{i=1}^{n} y_i T A_i\right) \geq 1 - 2|X|e^{-\mathcal{C}k}$$

for some constant $\mathcal{C} > 0$ (independent of n, k).

[Vu & al. *to appear*, arXiv:1507.00990v1/math.OC]

Separating hyperplanes

When $|X|$ is large, project separating hyperplanes instead

- Convex $C \subseteq \mathbb{R}^m$, $x \notin C$: then $\exists$ hyperplane c separating x, C
- In particular, true if $C = \mathsf{cone}(A_1, \ldots, A_n)$ for $A \subseteq \mathbb{R}^m$
- We can show $x \in C \Leftrightarrow Tx \in TC$ with high probability
- As above, if $x \in C$ then $Tx \in TC$ by linearity of T
  Difficult part is proving the converse

We can also project point-to-cone distances

Projecting the separation

Thm.

Given $c, b, A_1, \ldots, A_n \in \mathbb{R}^m$ of unit norm s.t. $b \notin \mathsf{cone}\{A_1, \ldots, A_n\}$ pointed, $\varepsilon > 0$, $c \in \mathbb{R}^m$ s.t. $c^\top b < -\varepsilon$, $c^\top A_i \geq \varepsilon$ $(i \leq n)$, and T a random projector:

$$\mathrm{P}\big[Tb \notin \mathsf{cone}\{TA_1, \ldots, TA_n\}\big] \geq 1 - 4(n+1)e^{-\mathcal{C}(\varepsilon^2 - \varepsilon^3)k}$$

for some constant $\mathcal{C}$.

Proof

Let $\mathscr{A}$ be the event that T approximately preserves $\|c - \chi\|^2$ and $\|c + \chi\|^2$ for all $\chi \in \{b, A_1, \ldots, A_n\}$. Since $\mathscr{A}$ consists of $2(n+1)$ events, by the JLL Corollary (squared version) and the union bound, we get

$$\mathrm{P}(\mathscr{A}) \geq 1 - 4(n+1)e^{-\mathcal{C}(\varepsilon^2 - \varepsilon^3)k}$$

Now consider $\chi = b$

$$\begin{aligned}
\langle Tc, Tb\rangle &= \frac{1}{4}(\|T(c+b)\|^2 - \|T(c-b)\|^2) \\
\text{by JLL} \quad &\leq \frac{1}{4}(\|c+b\|^2 - \|c-b\|^2) + \frac{\varepsilon}{4}(\|c+b\|^2 + \|c-b\|^2) \\
&= c^\top b + \varepsilon < 0
\end{aligned}$$

and similarly $\langle Tc, TA_i\rangle \geq 0$

[Vu et al. *to appear*, arXiv:1507.00990v1/math.OC]

The feasibility projection theorem

Thm.

Given $\delta > 0$, $\exists$ sufficiently large $m \leq n$ such that:

for any LFP input A, b where A is $m \times n$

we can sample a random $k \times m$ matrix T with $k \ll m$ and

$$\mathrm{P}(\text{orig. LFP feasible} \iff \text{proj. LFP feasible}) \geq 1 - \delta$$

Projecting optimality

Notation

- $P \equiv \min\{cx \mid Ax = b \wedge x \geq 0\}$ *(original problem)*
- $TP \equiv \min\{cx \mid TAx = Tb \wedge x \geq 0\}$ *(projected problem)*
- $v(P) =$ optimal objective function value of P
- $v(TP) =$ optimal objective function value of TP

The optimality projection theorem

- Assume $\mathsf{feas}(P)$ is bounded
- Assume all optima of P satisfy $\sum_j x_j \leq \theta$ for some given $\theta > 0$

  (prevents cones from being "too flat")

Thm.

Given $\delta > 0$,

$$v(P) - \delta \leq v(TP) \leq v(P) \qquad (*)$$

holds wahp

in fact $(*)$ holds with prob. $1 - 4ne^{-\mathcal{C}(\varepsilon^2 - \varepsilon^3)k}$ where $\varepsilon = \delta/(2(\theta+1)\eta)$ and $\eta = O(\|y\|_2)$ where y is a dual optimal solution of P having minimum norm

[Vu & al. *to appear*]

The easy part

Show $v(TP) \leq v(P)$:

- Constraints of P: $Ax = b \ \wedge \ x \geq 0$
- Constraints of TP: $TAx = Tb \ \wedge \ x \geq 0$
- $\Rightarrow$ constraints of TP are lin. comb. of constraints of P
- $\Rightarrow$ any solution of P is feasible in TP
  (btw, the converse holds almost never)
- P and TP have the same objective function
- $\Rightarrow TP$ is a relaxation of $P \Rightarrow v(TP) \leq v(P)$

The hard part (sketch)

- Eq. (7) equivalent to P for $\delta = 0$

$$\left.\begin{array}{rcl} cx & = & v(P) - \delta \\ Ax & = & b \\ x & \geq & 0 \end{array}\right\} \qquad (7)$$

Note: for $\delta > 0$, Eq. (7) is infeasible

- By feasibility projection theorem,

$$\left.\begin{array}{rcl} cx & = & v(P) - \delta \\ TAx & = & Tb \\ x & \geq & 0 \end{array}\right\}$$

is infeasible wahp for $\delta > 0$

- Hence $cx < v(P) - \delta \wedge TAx = Tb \wedge x \geq 0$ infeasible wahp
- $\Rightarrow cx \geq v(P) - \delta$ holds wahp for $x \in \mathsf{feas}(TP)$
- $\Rightarrow v(P) - \delta \leq v(TP)$

Solution retrieval

Projected solutions are infeasible in P

- $Ax = b \;\Rightarrow\; TAx = Tb$ by linearity
- However,
  Thm.
  For $x \geq 0$ s.t. $TAx = Tb$, $Ax = b$ with probability zero
- Can't get solution for original LFP using projected LFP!

Solution retrieval from optimal basis

- Primal $\min\{c^\top x \mid Ax = b \wedge x \geq 0\} \Rightarrow$ dual $\max\{b^\top y \mid A^\top y \leq c\}$
- Let $x' = \mathsf{sol}(TP)$ and $y' = \mathsf{sol}(\mathsf{dual}(TP))$
- $\Rightarrow (TA)^\top y' = (A^\top T^\top) y' = A^\top (T^\top y') \leq c$
- $\Rightarrow T^\top y'$ is a solution of $\mathsf{dual}(P)$
- $\Rightarrow$ we can compute an optimal basis J for P
- Solve $A_J x_J = b$, get x_J, obtain a solution x^* of P

Solving large quantile regression LPs

Regression

- multivariate random var. X
  function $y = f(X)$
  sample $\{(a_i, b_i) \in \mathbb{R}^p \times \mathbb{R} \mid i \leq m\}$

- sample mean:

$$\hat{\mu} = \arg\min_{\mu \in \mathbb{R}} \sum_{i \leq m} (b_i - \mu)^2$$

- sample mean conditional to $X = A = (a_{ij})$:

$$\hat{\nu} = \arg\min_{\nu \in \mathbb{R}^p} \sum_{i \leq m} (b_i - \nu a_i)^2$$

Quantile regression

- *sample median:*

$$\begin{aligned}\hat{\xi} &= \arg\min_{\xi\in\mathbb{R}} \sum_{i\leq m} |b_i - \xi| \\ &= \arg\min_{\xi\in\mathbb{R}} \sum_{i\leq m} \left(\frac{1}{2}\max(b_i-\xi,0) - \frac{1}{2}\min(b_i-\xi,0)\right)\end{aligned}$$

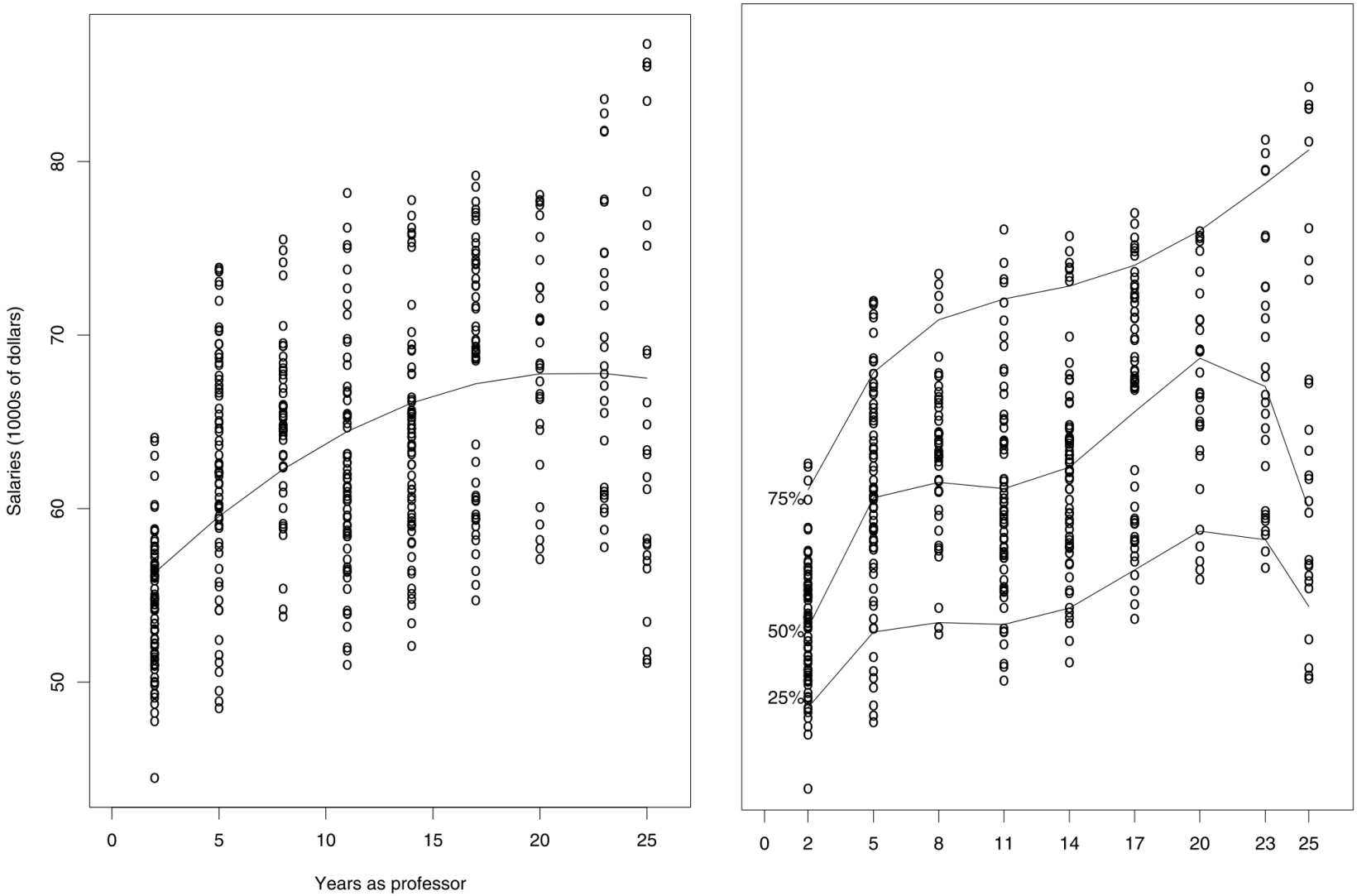
- *sample τ-quantile:*

$$\hat{\xi} = \arg\min_{\xi\in\mathbb{R}} \sum_{i\leq m} (\tau\max(b_i-\xi,0) - (1-\tau)\min(b_i-\xi,0))$$

- *sample τ-quantile conditional to $X = A = (a_{ij})$:*

$$\hat{\beta} = \arg\min_{\beta\in\mathbb{R}^p} \sum_{i\leq m} (\tau\max(b_i-\beta a_i,0) - (1-\tau)\min(b_i-\beta a_i,0))$$

Usefulness



A much better visibility of one's own poverty!

Linear Programming formulation

$$\left.\begin{array}{rrcl} \min & \tau u^+ + (1-\tau)u^- & & \\ & A(\beta^+ - \beta^-) + u^+ - u^- & = & b \\ & \beta, u & \geq & 0 \end{array}\right\}$$

- parameters: A is $m \times p$, $b \in \mathbb{R}^m$, $\tau \in \mathbb{R}$
- decision variables: $\beta^+, \beta^- \in \mathbb{R}^p$, $u^+, u^- \in \mathbb{R}^m$
- LP constraint matrix is $m \times (2p + 2m)$
  density: $p/(p+m)$ — can be high

Large datasets

- Russia Longitudinal Monitoring Survey, household data (`hh1995f`)
  - $m = 3783, p = 855$
  - $A =$ `hf1995f`, $b = \log \text{avg}(A)$
  - 18.5% dense
  - poorly scaled data, CPLEX yields infeasible (!!!) after around 70s CPU
  - `quantreg` in R fails
- 14596 RGB photos on my HD, scaled to 90×90 pixels
  - $m = 14596, p = 24300$
  - each row of A is an image vector, $b = \sum A$
  - 62.4% dense
  - CPLEX killed by OS after ≈30min (presumably for lack of RAM) on 16GB

Results on large datasets

Instance			Projection				Original		
τ	m	p	k	opt	CPU	feas	opt	CPU	qnt err
					`hh1995f`				
0.25	3783	856	411	0.00	8.53	0.038%	71.34	17.05	0.16
0.50				0.00	8.44	0.035%	89.17	15.25	0.05
0.75				0.00	8.46	0.041%	65.37	31.67	3.91
					`jpegs`				
0.25	14596	24300	506	0.00	231.83	0.51%	0.00	3.69E+5	0.04
0.50				0.00	227.54	0.51%	0.00	3.67E+5	0.05
0.75				0.00	228.57	0.51%	0.00	3.68E+5	0.05
					`random`				
0.25	1500	100	363	0.25	2.38	0.01%	1.06	6.00	0.00
0.50				0.40	2.51	0.01%	1.34	6.01	0.00
0.75				0.25	2.57	0.01%	1.05	5.64	0.00
0.25	2000	200	377	0.35	4.29	0.01%	2.37	21.40	0.00
0.50				0.55	4.37	0.01%	3.10	23.02	0.00
0.75				0.35	4.24	0.01%	2.42	21.99	0.00

$$\text{feas} = 100\frac{\|Ax-b\|_2}{\|b\|_1/m}$$

$$\text{qnt err} = \frac{\|\text{qnt} - \text{proj. qnt}\|_2}{\#\text{ cols}}$$

IPM with *no simplex crossover*:
solution w/o opt. guarantee
cannot trust results
simplex method won't work
due to ill-scaling and size

Outline

Summary

1. Graphs and weighted graphs necessary to model data
   Abduction, observation graphs, consistency relations
2. Computers can "reason by analogy" (clustering)
   Modularity clustering
3. Clustering on vectors allows more flexibility
   k-means, MSSC
4. Need to embed (weighted) graphs into Euclidean spaces
   Metric embeddings, Distance Geometry
5. High dimensions make clustering expensive/unstable
   Distance resolution limit
6. Use random projections to reduce dimensions
   Johnson-Lindenstrauss lemma

Some of *my* references

1. Vu, Poirion, L., *Random Projections for Linear Programming*, Math. Oper. Res., to appear
2. L., Lavor, *Euclidean Distance Geometry: an Introduction*, Springer, Zürich, in press
3. Mencarelli, Sahraoui, L., *A multiplicative weights update algorithm for MINLP*, Eur. J. Comp. Opt., 5:31-86, 2017
4. 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
5. Dias, L., *Diagonally Dominant Programming in Distance Geometry*, in Cerulli et al. (eds.) *Proceedings of ISCO*, LNCS 9849:225-246, Springer, Zürich, 2016
6. L., Lavor, Maculan, Mucherino, *Euclidean distance geometry and applications*, SIAM Review, 56(1):3-69, 2014
7. Cafieri, Hansen, L., *Improving heuristics for network modularity maximization using an exact algorithm*, Discr. Appl. Math., 163:65-72, 2014
8. L., Lavor, Mucherino, *The DMDGP seems easier on proteins*, in Mucherino et al. (eds.) *Distance Geometry: Theory, Methods, and Applications*, Springer, New York, 2013
9. Aloise, Hansen, L., *An improved column generation algorithm for minimum sum-of-squares clustering*, Math. Prog. A 131:195-220, 2012
10. Aloise, Cafieri, Caporossi, Hansen, Perron, L., *Column generation algorithms for exact modularity maximization in networks*, Phys. Rev. E, 82(4):046112, 2010
11. L., Cafieri, Tarissan, *Reformulations in Mathematical Programming: A Computational Approach*, in Abraham et al. (eds.) *Foundations of Computational Intelligence* vol. 3, SCI 203:153-234, Springer, Heidelberg 2009
12. Lavor, L., Maculan, *Computational Experience with the Molecular Distance Geometry Problem*, in Pintér (ed.) *Global Optimization: Scientific and Engineering Case Studies*, Springer, Berlin, 2006