Linear Algebra in Data Exploration

Daniel L Boley
University of Minnesota

Find Reduced Order Representations for Large Unstructured Data Collections to facilitate finding hidden patterns, connections, outliers, and to eliminate noise.

Explore the many ways linear algebra shows up in deep learning algorithms.

Study recent papers and do a project related to these topics.

Outline

- Organization and Administrative
- Clustering, Graph Analysis & Partitioning
  - Dimensionality Reduction: mapping
  - Spectral Partitioning
  - Network Link Analysis: Pagerank, HITS
  - Some Applications
- Deep Learning Algorithms
  - Basics (neural networks, backpropagation)
  - Sample of recent successful Algorithms
  - Convolutional Neural networks
  - Neural Networks with memory
  - Transformers: Attention Mechanism
  - Diffusion Models
  - Reinforcement Learning
- Deep Learning Analysis Tools
  - Mechanistic Interpretability
  - Probe network with local perturbations
  - Dissentangle concepts with sparse representations

Linear Algebra in Data Exploration

- Many large unstructured data sets are represented as tables and tables of numbers or connections.
  - Text documents (news, laws, WWW documents).
  - Gene expression profiles
  - Attributes for individual people, transactions, locations, ecosystems, ...

  tabular

  - Gene-gene or protein-protein interaction networks
  - WWW connectivity graph
  - Computer inter-connect in Internet
  - People-people affinities in Social Media

  graph

  - Many example datasets can easily have up to $O(10^{9+})$ data points.
  - Many datasets have much noise or many attributes.
  - Many example datasets are sampled, subject to sampling bias.
  - New DL algorithms learn structure implicitly.

  machine learning

Tools to Explore

- Dimensionality Reduction, Latent Space.
  - Represent each data sample with a reduced set of attribute values
  - Minimize loss of information
  - Implicit assumption: data is subject to some level of noise.
  - Expose essential patterns and/or reduce cost of subsequent algorithms
- Graph Properties
  - partitioning
  - identify important nodes or links
  - aggregrate properties
  - model propagation of influence/information
- Deep Learning
  - Matrix-Matrix operations & optimization at the heart of DL computations
  - Probability/Statistics at the heart of DL theory
  - Adopt Matrix methods: Low-Rank, Least Squares, Sparsity

Outline

- Organization and Administrative
- Clustering, Graph Analysis & Partitioning
  - Dimensionality Reduction: mapping
  - Spectral Partitioning
  - Network Link Analysis: Pagerank, HITS
  - Some Applications
- Deep Learning Algorithms
  - Basics (neural networks, backpropagation)
  - Sample of recent successful Algorithms
  - Convolutional Neural networks
  - Neural Networks with memory
  - Transformers: Attention Mechanism
  - Diffusion Models
  - Reinforcement Learning
- Deep Learning Analysis Tools
  - Mechanistic Interpretability
  - Probe network with local perturbations
  - Dissentangle concepts with sparse representations

Class Info

Pointers

- Instructor: Daniel Boley, office 4-225C Keller, e-mail boley@cs.umn.edu.
- Class meets in AkerH 227 on MW 4-5:15pm.
- Web page: `https://canvas.umn.edu/courses/518591`

Organization

- One introductory lecture reviewing linear algebra and some optimization.
- 21-23 lectures: Student presentation of a scheduled paper:

Web Site

- Semester Plan
  - the schedule of papers to be presented.
  - alternative papers (you can use one of these instead)
  - most papers are research papers. Long papers need a "highlight" talk.
- Canvas
  - All items should be submitted through Canvas.

Class Plan

- 21 lectures: A student presents a paper:
  - Select a paper from the list on the web site, in advance.
  - Make selection in advance to give everyone a chance to read the relevant paper in advance.
  - Everyone listening submits a short synopsis of what they got out of the lecture (the main take-away message), and separately some feedback to be passed back to each speaker.
  - All submissions are submitted through Canvas.
- Student Term Projects
  - Each student carries out an independent project.
  - Submit a short 1-page project proposal after about 3 weeks.
  - Give a 10-minute presentation of your project during the last 4 lectures (2 weeks) of the semester.
  - Submit a report of at least 10 pages at the end of the semester.

Papers

- Graphs and Random Walks

  A. Directed Graphs
  B. Graphs: Random Walks on Graphs
  C. Graphs: Extensions and Methods
  D. Graphs &Linear Algebra: Classic Results
  E. Graphs and Machine Learning

- Machine Learning

  F. ML: Deep Neural Networks
  G. ML: Convolutional Neural Nets
  H. ML: Latent
  I. ML: Learning
  J. ML: Transformers
  K. ML: Data Leakage

- Machine Learning (cont.)

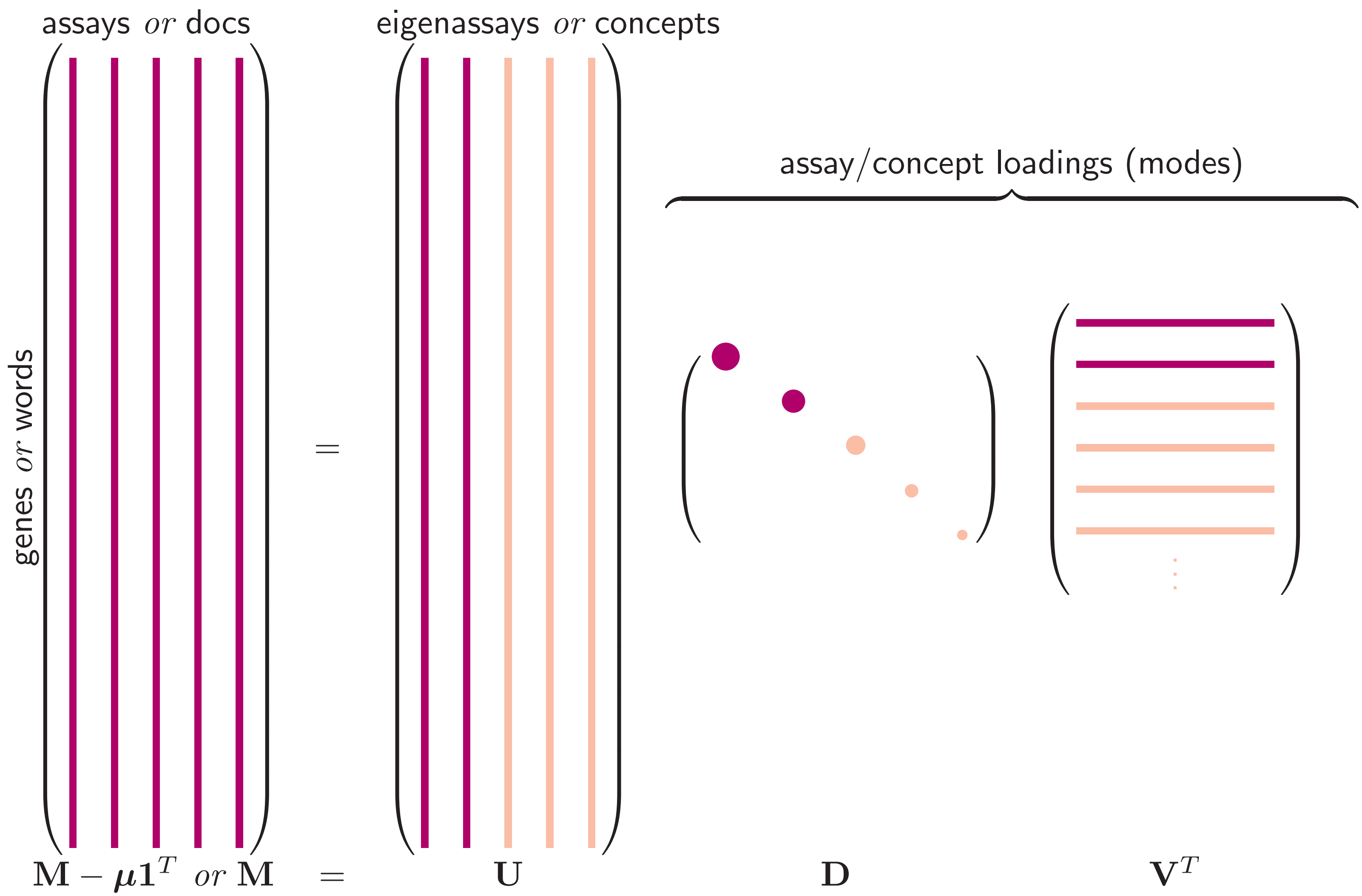
  L. ML: Linear Algebra
  M. ML: Generalization Performance
  N. ML: Physics Induced Neural Nets
  O. ML: Mechanistic Interpretability – Sparse Auto Encoders
  P. ML: Reinforcement Learning
  Q. ML: Diffusion Models
  R. ML: Recent Papers

- Optimization

  S. Optimization in ML

- T. Matrix Sketching

Singular Value Decomposition – SVD



Singular Value Decomposition – SVD

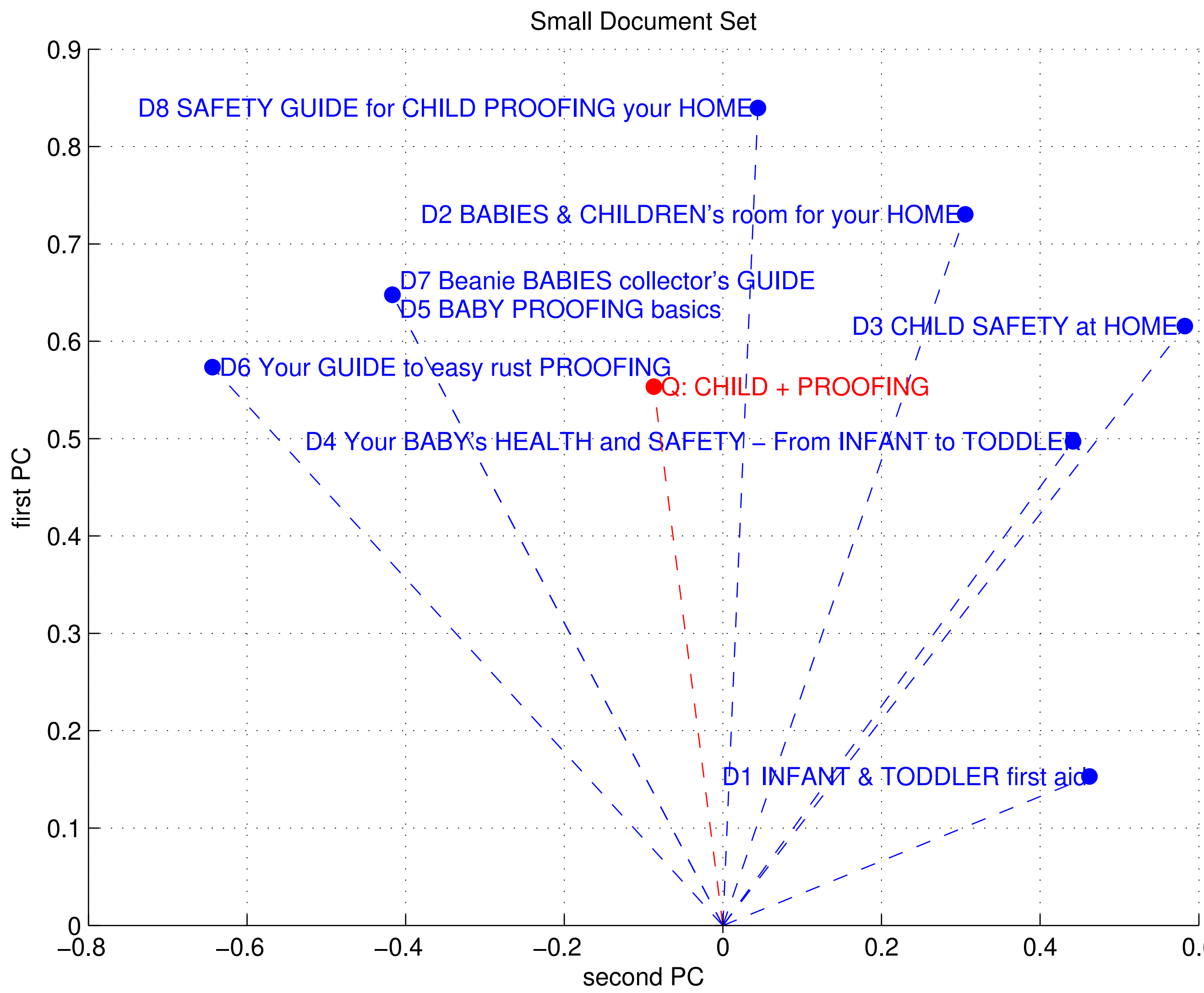
- Eliminate Noise
- Reduce Dimensionality
- Expose Major Components
- Suppose samples are columns of $m \times n$ matrix $\mathbf{M}$.
- Try to find k pseudo-data columns such that all samples can be represented by linear combinations of those k pseudo-data columns.
- Primary criterion: minimize the 2-norm of the discrepancy between the original data and what you can represent using k pseudo-data columns.
- Answer: Singular Value Decomposition of $\mathbf{M}$.
  If centered, get Principal Components of $\mathbf{M}$ (PCA).
- Sometimes, for statistical reasons, want to remove uniform signal:
  - $\mathbf{M} \leftarrow \mathbf{M} - \boldsymbol{\mu}\mathbf{1}^T$,
    where $\boldsymbol{\mu} = \mathbf{M} \cdot \mathbf{1}$.
  - Then $\mathbf{M}^T\mathbf{M}$ is the Sample Covariance Matrix.
  - Even without centering, $\mathbf{M}^T\mathbf{M}$ is a “Gram” matrix.

Text Documents – Data Representation

- Each document represented by n-vector $\mathbf{d}$ of word counts, scaled to unit length.
- Vectors assembled into Term Frequency Matrix $\mathbf{M} = (\,\mathbf{d}_1 \quad \cdots \quad \mathbf{d}_m\,)$.

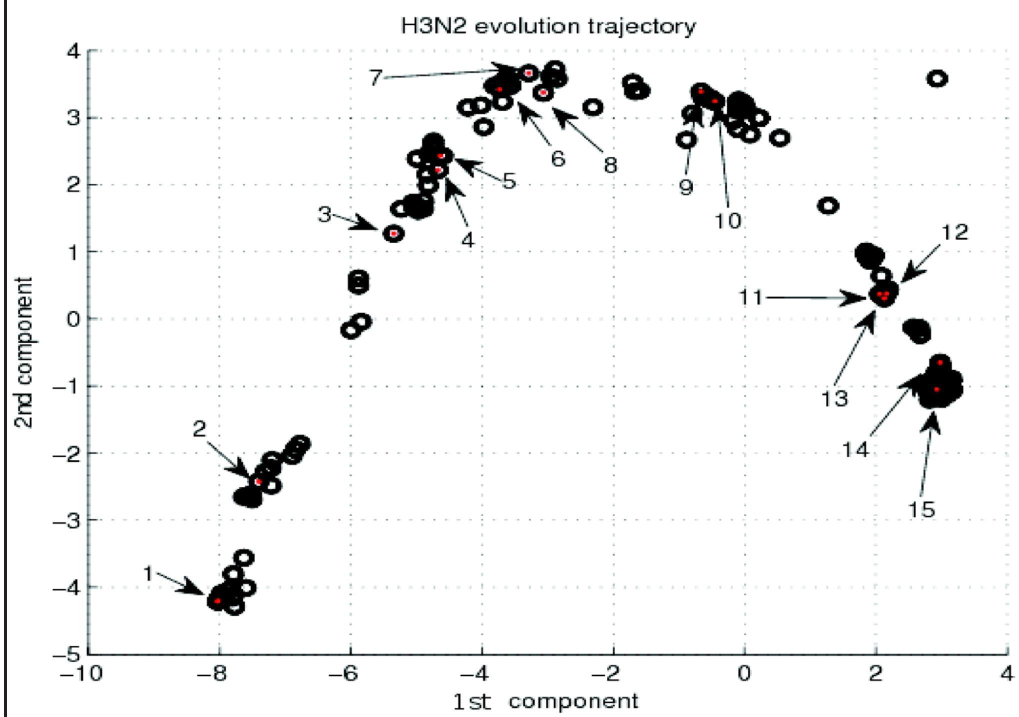
	D1 INFANT & TODDLER first aid	D2 BABIES & CHILDREN's room for your HOME	D3 CHILD SAFETY at HOME	D4 Your BABY's HEALTH and SAFETY - From INFANT to TODDLER	D5 BABY PROOFING basics	D6 Your GUIDE to easy rust PROOFING	D7 Beanie BABIES collector's GUIDE	D8 SAFETY GUIDE for CHILD PROOFING your HOME
BABY	0	$\sqrt{3}$	0	$\sqrt{5}$	$\sqrt{2}$	0	$\sqrt{2}$	0
CHILD	0	$\sqrt{3}$	$\sqrt{2}$	0	0	0	0	$\sqrt{5}$
GUIDE	0	0	0	0	0	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{5}$
HEALTH	0	0	0	$\sqrt{5}$	0	0	0	0
HOME	0	$\sqrt{3}$	$\sqrt{2}$	0	0	0	0	$\sqrt{5}$
INFANT	$\sqrt{2}$	0	0	$\sqrt{5}$	0	0	0	0
PROOFING	0	0	0	0	$\sqrt{2}$	$\sqrt{2}$	0	$\sqrt{5}$
SAFETY	0	0	$\sqrt{2}$	$\sqrt{5}$	0	0	0	$\sqrt{5}$
TODDLER	$\sqrt{2}$	0	0	$\sqrt{5}$	0	0	0	0

Latent Semantic Indexing – LSI



- Stay length-independent: compare using just angles.

Model Avian Influenza Virus



from Lam&Boley 2011

Number	Vaccine strain
1	A/Aichi/1968
2	A/Port Chalmers/1/1973
3	A/Philippines/2/1982
4	A/leningrad/360/1986
5	A/Shanghai/11/1987
6	A/Beijing/353/1989
7	A/Shangdong/9/1993
8	A/Johannesburg/33/1994
9	A/Sydney/5/1997
10	A/Moscow/10/1999
11	A/Fujian/411/2002
12	A/California/7/2004
13	A/Wisconsin/67/2005
14	A/Brisbane/10/2007
15	A/Perth/16/2009

- Evolution is a flow, naturally falls in chronological order.
- Without vaccine, picture is more a random cloud of points.
- Suggests vaccine use does affect evolution of virus.

Model Avian Influenza Virus

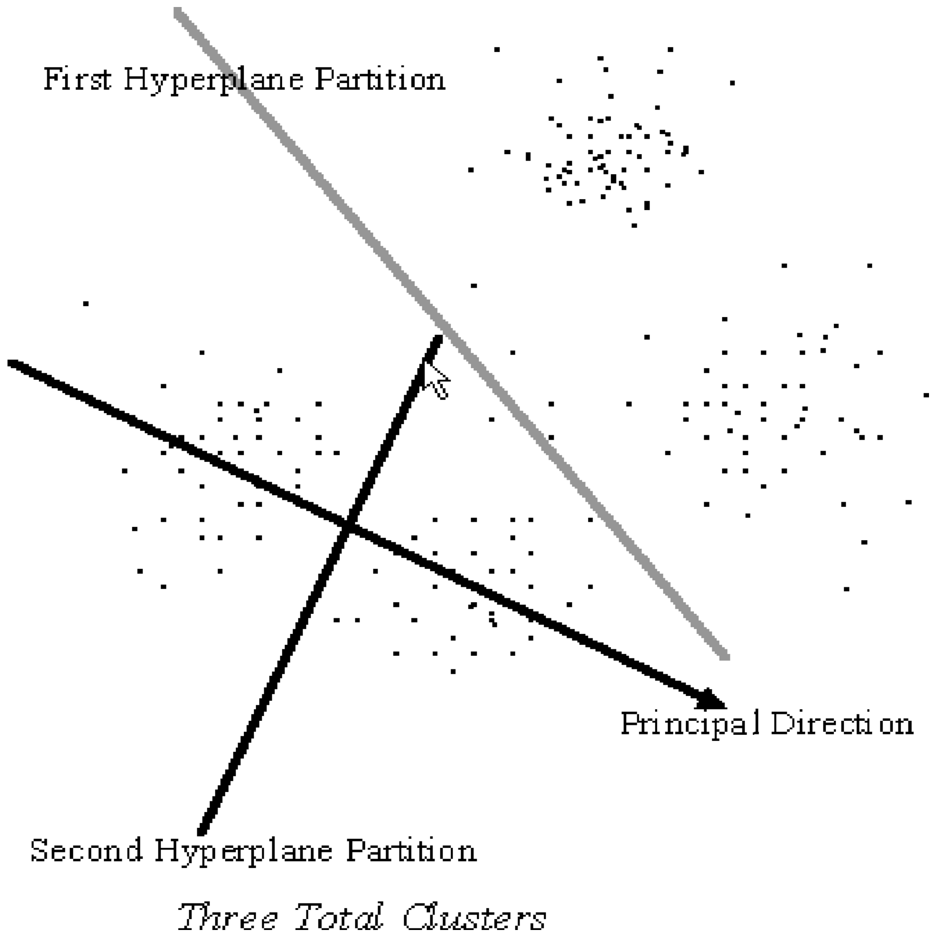
(Lam et al., 2012)

- Avian Flu Virus characterized by the HA protein, which the virus uses to penetrate the cell.
- The protein is described by a string of 566 symbols, each representing one of 20 Amino Acids.
- Embed in high dimensional Euclidean space by replacing each Amino Acid with a string of 20 bits:
  - E.g. 3rd amino acid $= \rightarrow$ 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
- Result is a vector of length $566 \cdot 20 = 11230$.
- Use PCA to reduce dimensions from 11320 to 6.
- Use first 2 components to track evolution of this protein in a simple visual way.

Outline

- Organization and Administrative
- Clustering, Graph Analysis & Partitioning
  - Dimensionality Reduction: mapping
  - Spectral Partitioning
  - Network Link Analysis: Pagerank, HITS
  - Some Applications
- Deep Learning Algorithms
  - Basics (neural networks, backpropagation)
  - Sample of recent successful Algorithms
  - Convolutional Neural networks
  - Neural Networks with memory
  - Transformers: Attention Mechanism
  - Diffusion Models
  - Reinforcement Learning
- Deep Learning Analysis Tools
  - Mechanistic Interpretability
  - Probe network with local perturbations
  - Dissentangle concepts with sparse representations

Principal Direction Divisive Partitioning



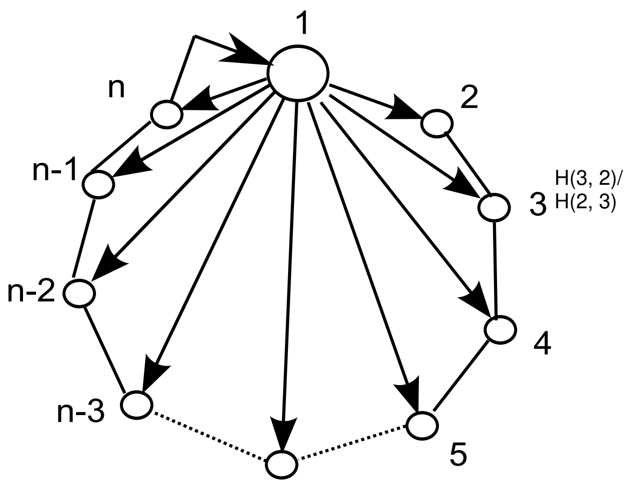
(Boley, 1998)

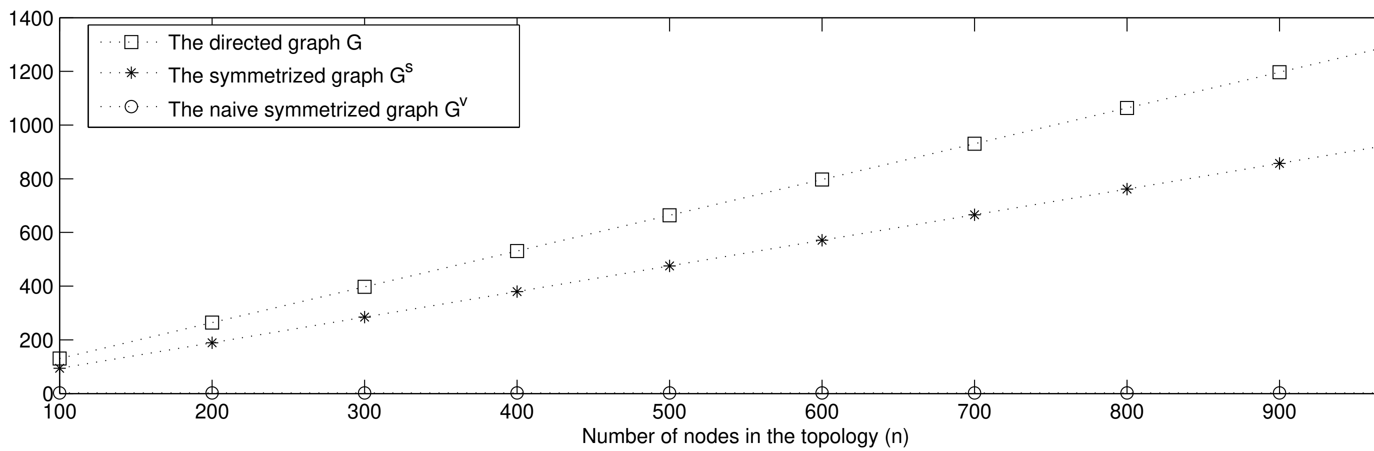
Divisive Partitioning for Unsupervised Clustering

- Unsupervised, as opposed to Supervised:
  - No predefined categories;
  - No previously classified training data;
  - No a-priori assumptions on the number of clusters.
- Top-down Hierarchical:
  - Imposes a tree hierarchy on unstructured data;
  - Tree is source for some taxomonic information for dataset;
  - Tree is generated from the root down.
  - Result is Principal Direction Divisive Partitioning. (Boley, 1998)
- Multiway Clustering – variations using similar ideas.
  - Project onto first k principal directions. Result: each data sample is represented by k components.
  - Apply classical k-means clustering to projected data.
  - Used for both Graph Partitioning and Data Clustering. (Dhillon, 2001)
- Empirically Best Approach: a hybrid method:
  - Use Divisive Partitioning first (deterministic).
  - Refine with K-means (avoids initialization issues). (Savaresi & Boley, 2004)

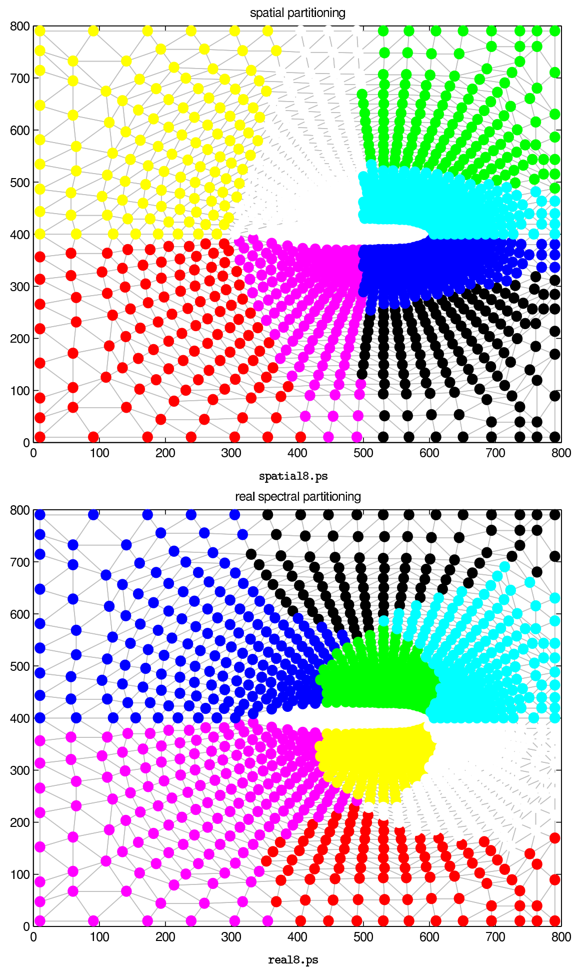
Spectral Methods on Graphs

- Model an undirected graph by a random walk.
- Recurring Node Probability $\Longleftrightarrow$ Pagerank [as in google].
- Average round-trip commute time is a metric-squared
- Can embed graph in Euclidean space preserving distances.
- Principal Direction splitting on embedding is equivalent to two-way Spectral Graph Partitioning.
- Can be extended to directed graphs (e.g., commute times still a metric). (Boley et al., 2011)

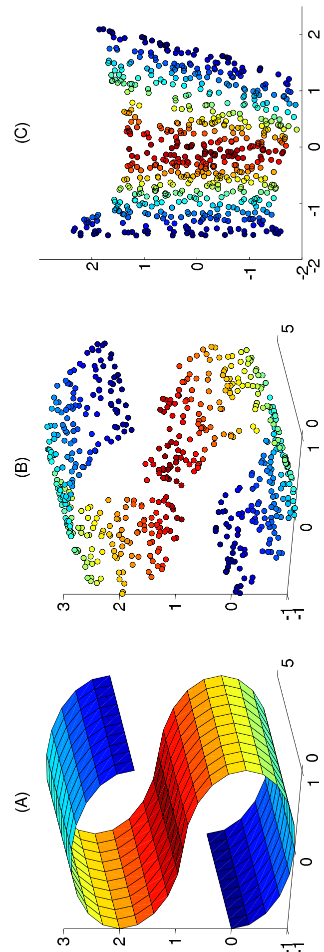




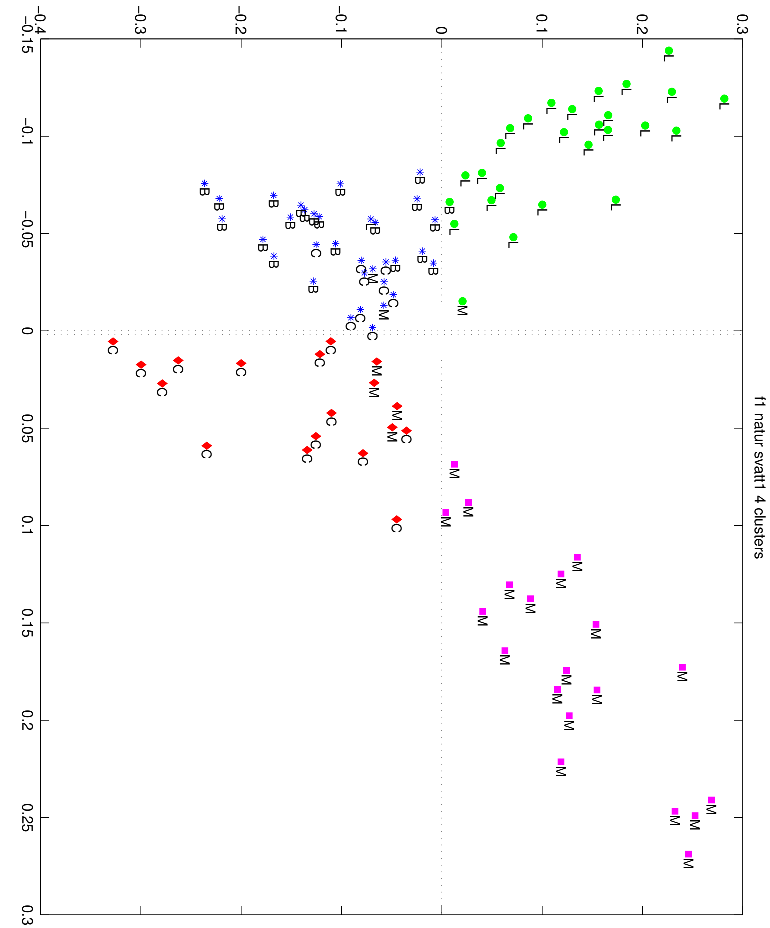
Graph Examples



graph partitioning

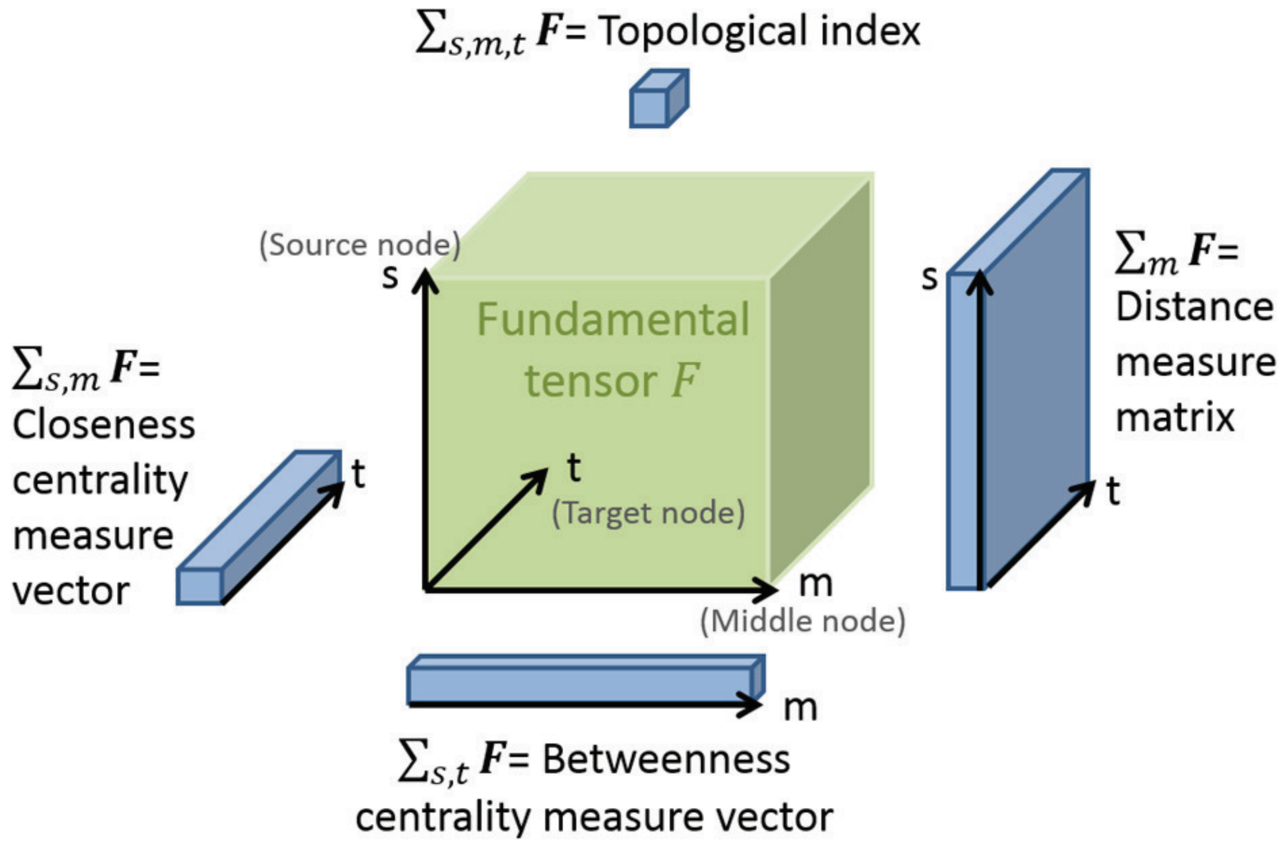


LLE



clustering

Tensor Applications



Tensor Applications

- Closeness(k): $\sum_{ij} \mathbf{N}(i, j, k)$
  (7.25000 30.5000 18.8333 21.3333 9.44444)

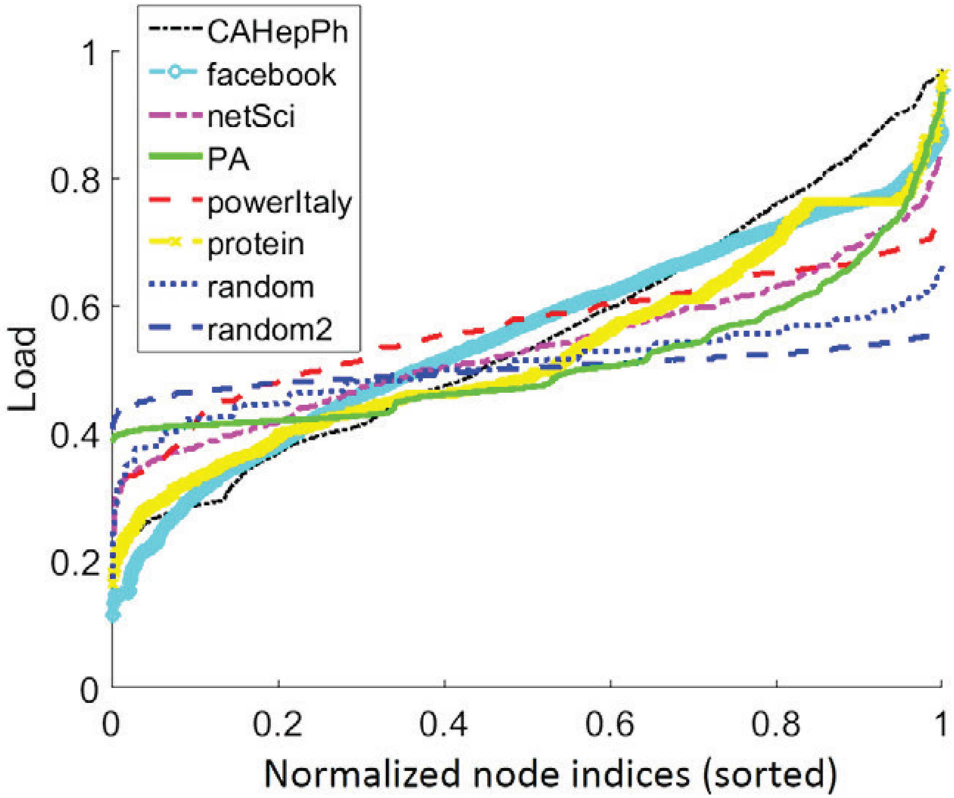
- Betweenness(j): $\sum_{ik} \mathbf{N}(i, j, k)$
  (28.3333 9.44444 14.1666 14.1666 21.2500)

- Probability of passage (i, j, k): $\widetilde{\mathbf{N}}(i, j, k) = \mathbf{N}(i, j, k)/\mathbf{N}(j, j, k)$
  [or zero if $i = k$ or $j = k$].

- node load: chance of passage averaged over all start/end nodes
  $\sum_{ik} \widetilde{\mathbf{N}}(i, j, k) \; / \; (n-1)^2$.
  (.796875 .475000 .572916 .531250 .748958)

- prob of passiing 2 when starting from 1 before reaching 5:
  .400 (full network); .500 (if 4 removed); .333 (if 4 is to be avoided)

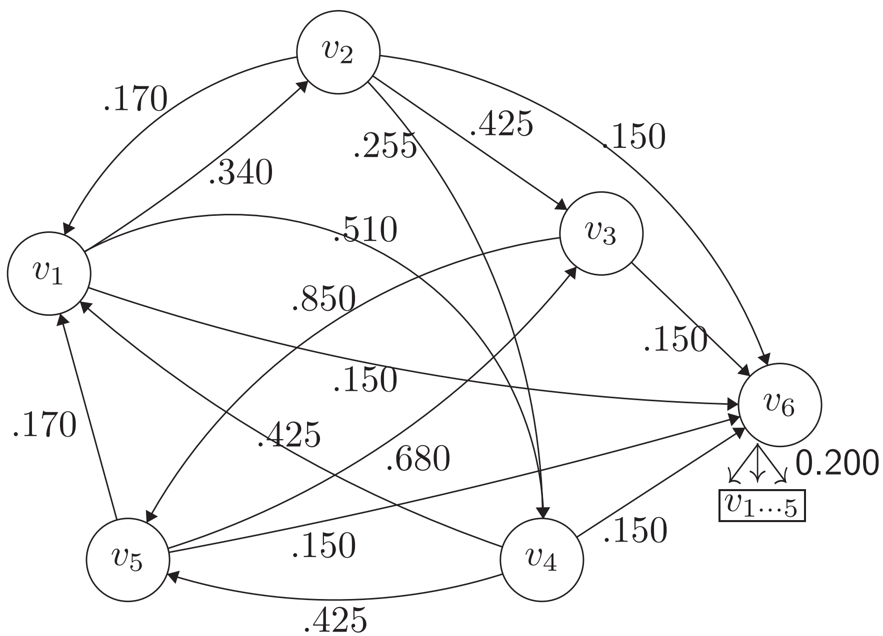
Load examples



- High Energy Physics Collaboration Network
- sampled Facebook network
- netSci: co-author network of network scientists
- Italian power grid
- protein-protein interaction network
- synthetic: Preferential Attachment
- synthetic: random Erdös-Rényi (8 or 40 initial links)

- Load is most balanced in ER network, less so in PA.
- Co-author networks: more imbalanced.
- power network: more balanced load.

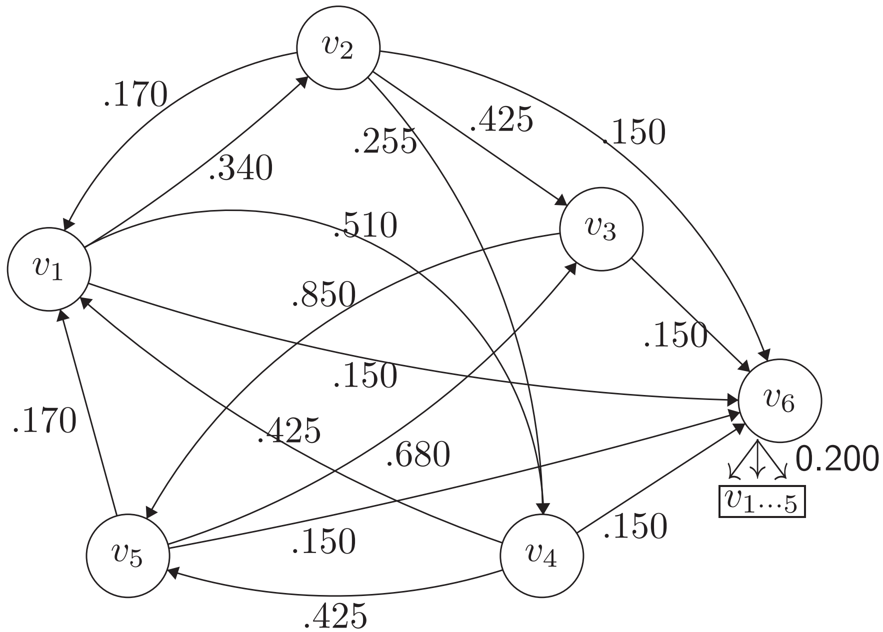
Influence Propagation



- Seek to predict trust between new pairs

- Network shows how much agent i trusts agent j, for $i, j{=}1, \ldots, 5$ after direct interaction.
- Node 6 is extra evaporation node to control influence locality.

Influence Propagation



- Network shows how much agent i trusts agent j, for $i,j{=}1,\ldots,5$ after direct interaction.
- Node 6 is extra evaporation node to control influence locality.

- Seek to predict trust between new pairs: predicted trust of j by i = $\mathbf{PHT}(i,j) = \mathbf{N}(i,j,6)/\mathbf{N}(j,j,6)$. $\boxed{\text{A}}$
- Example: from point of view of agent 4:

$$\mathbf{PHT}(4, 1{:}5) = (0.60,\ 0.29,\ 0.53,\ 1.00,\ 0.66),$$

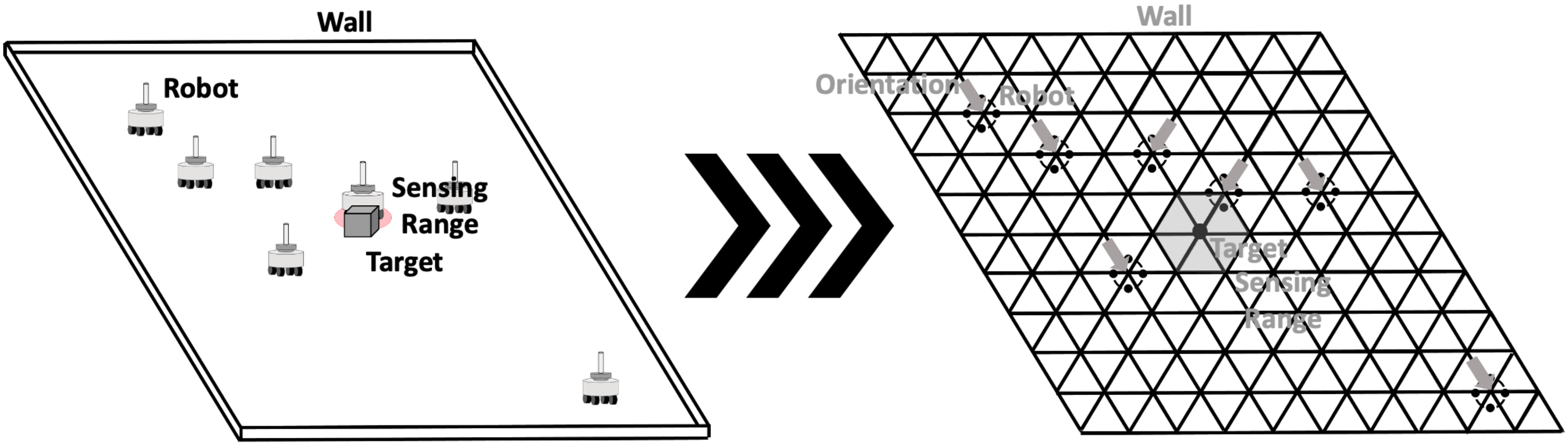
- If agent 2 is a bad actor to be avoided then we can use Fundamental Tensor to compute trust values based on avoiding 2:

$$\mathbf{PHT}_2(4, 1{:}5) = (0.60,\ 0.00,\ 0.39,\ 1.00,\ 0.54)$$

[plug $\widetilde{\mathbf{N}} = \mathbf{N}(:,:,6) - \frac{\boldsymbol{N}(:,2,6)}{\boldsymbol{N}(2,2,6)}\mathbf{N}(2,:,6)$ into $\boxed{\text{A}}$.

- Now most trustworthy node for 4 is node 1 instead of node 5.

Correlated Random Walk for Robots



Continuous domain $\Rightarrow$ random walk on discrete graph

- Fast computation of aggregate properties
- Use Hitting Time as an example

- Triangles to get equal link lengths
- State = (position, orientation)

Probability Transition Matrix

- $\mathbf{P}_{ij}$ = probability of moving to node j from i.
- Goal nodes lead directly to absorbing state, put last.
- Partition matrix where $\boxed{\mathbf{P} = \begin{bmatrix} \mathbf{Q} & \mathbf{r} \\ \mathbf{0} & \mathit{1} \end{bmatrix}}$

  - $\mathbf{Q} \Longleftrightarrow$ transient states
  - $\mathit{1} \Longleftrightarrow$ absorbing state (nest)
  - $\mathbf{r} \Longleftrightarrow$ transition from transient states to nest

Average Hitting Times

- $\mathbf{h}_\mu$ = vector of average Hitting Times (to reach nest) from any transient node
- $\mathbf{h}_{\sigma^2}$ = vector of variances of hitting times
- Obtain both values directly by solving system of linear equations [matrix of coefs = $(\mathbf{I}-\mathbf{Q})$]:

$$(\mathbf{I}-\mathbf{Q})\mathbf{h}_\mu = \mathbf{c} = \text{vector of all ones}$$

$$(\mathbf{I}-\mathbf{Q})\mathbf{h}_{\sigma^2} = 2\mathbf{h}_\mu - (\mathbf{I}-\mathbf{Q})\mathbf{h}_\mu - (\mathbf{I}-\mathbf{Q})\mathbf{h}_\mu^2$$

(Grinstead Snell 1997, Kemeny Snell 1976)

Solving The Linear System

- Dimension of system to solve: approx $6d^2 \times 6d^2$
  where $d =$ dimension of Arena
- System very sparse: at most 6 entries per row.
- Can be solved efficiently by iterative sparse matrix methods

arena size	matrix $\mathbf{P}$ dimension	number non-zeros	setup time (sec)	solve time (sec)
30×30	5767	31223	1.5	0.05
150×150	136807	807863	38.4	5.4
300×300	543607	3236663	44.9	35.6

References

Boley, D., Ranjan, G., & Zhang, Z.-L. (2011). Commute times for a directed graph using an asymmetric Laplacian. *Linear Algebra and Appl.*, *435*, 224–242.

Boley, D. L. (1998). Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, *2*, 325–344.

Dhillon, I. S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. *KDD* (pp. 269–274).

Lam, H. C., & Boley, D. (2011). Analyze influenza virus sequences using binary encoding approach. *11th International Workshop on Data Mining in Bioinformatics (BIOKDD'11)*. affiliated with ACM SIGKDD11.

Lam, H. C., Sreevatsan, S., & Boley, D. (2012). Analyzing influenza virus sequences using binary encoding approach. *Scientific Programming*, *20*, 3–13. expanded version of [(Lam & Boley, 2011)].

Savaresi, S. M., & Boley, D. (2004). A comparative analysis on the bisecting K-means and the PDDP clustering algorithms. *Intelligent Data Analysis*, *8*, 345–362.

Conclusions

- Classical methods for structured data
  - linear & non-linear separators
  - tabular & graph data
  - simple image manipulations
- Deep Learning implicitly learns underlying structure
  - General images, audio, unstructured text
  - Structure buried within network, hidden from view
- Aim of this course:
  - Explore hidden structure
  - Use hidden patterns to improve methods

Conclusions

- Classical methods for structured data
  - linear & non-linear separators
  - tabular & graph data
  - simple image manipulations
- Deep Learning implicitly learns underlying structure
  - General images, audio, unstructured text
  - Structure buried within network, hidden from view
- Aim of this course:
  - Explore hidden structure
  - Use hidden patterns to improve methods

THANK YOU