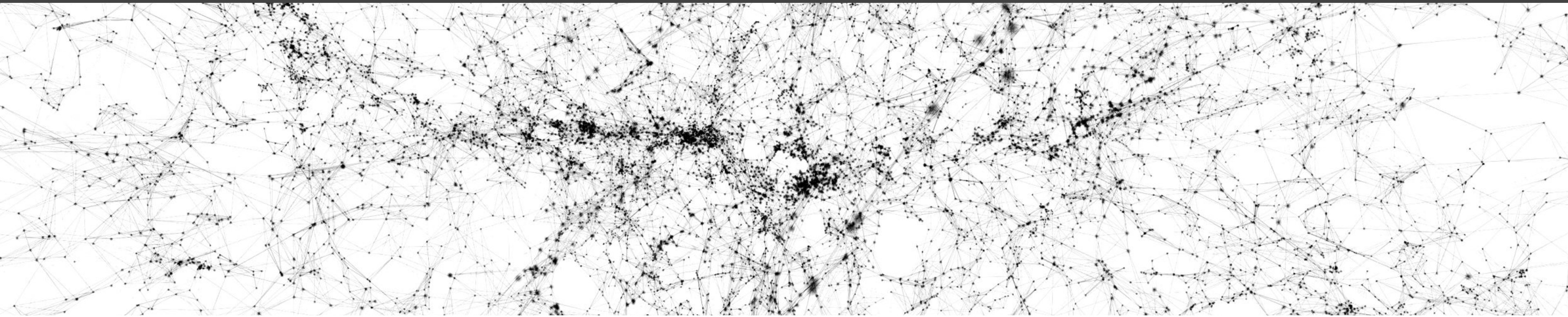
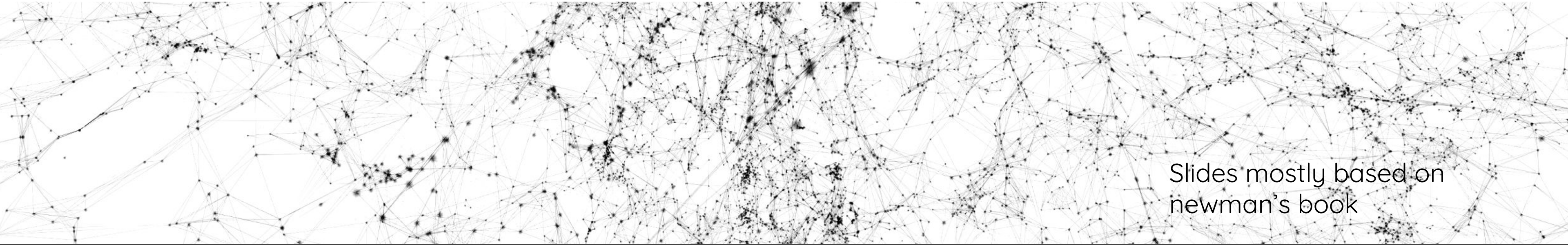


Visualization

Analysis of complex interconnected data



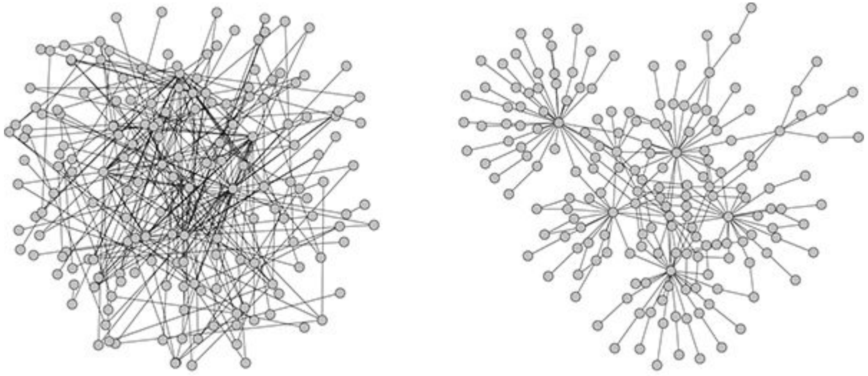
Slides mostly based on newman's book

Outline

- Graph layouts
- Drawing Graphs by Eigenvectors
- Force-Directed Layouts
- Learning the Layout and Transition between Layouts
- Some Plotting Examples
- Graph Visualization Tools

Network visualization

What is a good visualization? Make the network structure as clear as possible



(a) (b)

two different pictures of the same network

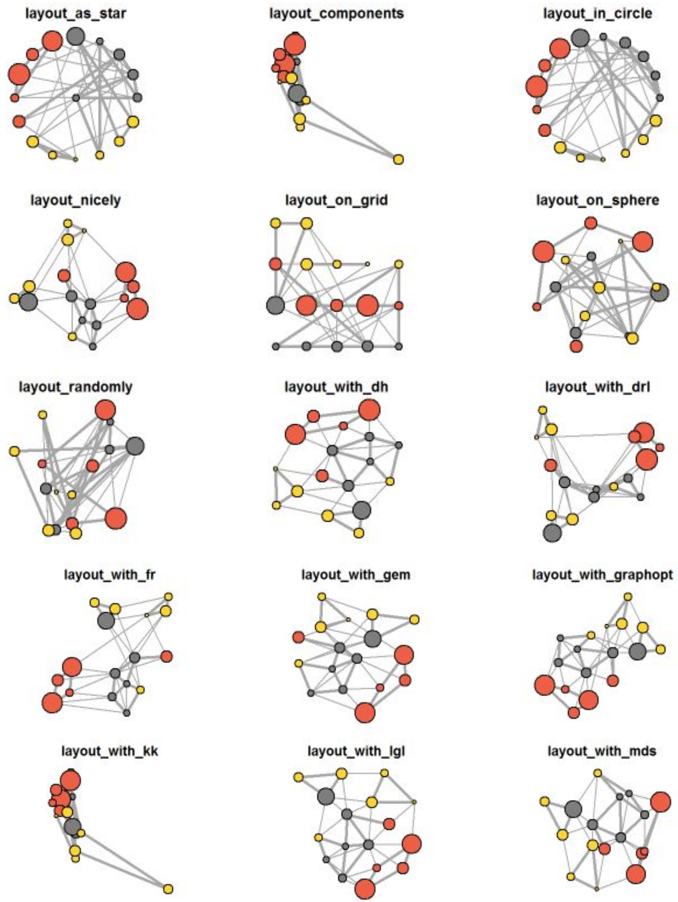
Layout algorithms map nodes into 2D space for plotting and visualization (Assign <x,y>)

Network visualization

Layout algorithms map nodes into 2D space for plotting and visualization (Assign <x,y>)

Which layout is the best?

the lengths of most edges in the network, as drawn on the page, are short



Layout Algorithm to Minimize Ink

Assign <x,y> to all nodes so that the lengths of most edges in the network, as drawn on the page, are short

Assume the 1D case:

$d(i,j) = |x_i - x_j|$ or $(x_i - x_j)^2$

$\Delta = \frac{1}{2} \Sigma_{ij} A_{ij}(x_i - x_j)^2$

$= \frac{1}{2} \Sigma_{ij} A_{ij}(x_i^2 + x_j^2 - 2 x_i x_j) = \frac{1}{2} [\Sigma_{ij} A_{ij} x_i^2 + \Sigma_{ij} A_{ij} x_j^2 - 2 \Sigma_{ij} A_{ij} x_i x_j] = \Sigma_{ij} A_{ij} x_i^2 - \Sigma_{ij} A_{ij} x_i x_j$

$= \Sigma_i x_i^2 \Sigma_j A_{ij} - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_i x_i^2 d_i - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_i x_i x_i d_i - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_{ij} x_i x_j d_i \delta_{ij} - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_{ij} (d_i \delta_{ij} - A_{ij}) x_i x_j$

Looks familiar?

$\delta_{ij} == 1$ iff $i == j$ {kronecker delta}

Layout Algorithm to Minimize Ink

Assign <x,y> to all nodes so that the lengths of most edges in the network, as drawn on the page, are short

Assume the 1D case:

$d(i,j) = |x_i - x_j|$ or $(x_i - x_j)^2$

$\boldsymbol{\Delta} = ½ \Sigma_{ij} A_{ij}(x_i - x_j)^2$ $= \mathbf{x^T L x}$

$= ½ \; \Sigma_{ij} A_{ij}(x_i^2 + x_j^2 - 2 x_i x_j) = ½ [\Sigma_{ij} A_{ij} x_i^2 + \Sigma_{ij} A_{ij} x_j^2 - 2 \Sigma_{ij} A_{ij} x_i x_j] = \Sigma_{ij} A_{ij} x_i^2 - \Sigma_{ij} A_{ij} x_i x_j$

$= \Sigma_i x_i^2 \Sigma_j A_{ij} - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_i x_i^2 d_i - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_i x_i x_i d_i - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_{ij} x_i x_j d_i \boldsymbol{\delta}_{ij} - \Sigma_{ij} A_{ij} x_i x_j = \Sigma_{ij} (d_i \boldsymbol{\delta}_{ij} - A_{ij}) x_i x_j = \Sigma_{ij} L_{ij} x_i x_j$

Looks familiar?

Layout Algorithm to Minimize Ink

Assign <x,y> to all nodes so that the lengths of most edges in the network, as drawn on the page, are short

Assume the 1D case:

$d(i,j) = |x_i - x_j|$ or $(x_i - x_j)^2$

$\mathbf{\Delta} = ½ \Sigma_{ij} A_{ij}(x_i - x_j)^2 = \mathbf{x^T L x}$

Spectral clustering objective, number of cut edges

Layout Algorithm to Minimize Ink

For more dimensions

$d(i,j) = \sum_k (x^k_i - x^k_j)^2$, squared Euclidean distance

$\Delta = \sum_{ij} A_{ij} d_{ij} = \sum_k (\mathbf{x}^k)^T \mathbf{L}\, \mathbf{x}^k$

(a) Spring embedding of citation network G

(b) Laplacian layout of co-citation graph $\mathcal{S}_C(G)$

visualizations of networks using the eigenvectors of the graph Laplacian

Drawing Graphs by Eigenvectors: Theory and Practice

Visualization of Bibliographic Networks with a Reshaped Landscape Metaphor

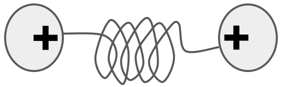
Plotting graphs: from math to art! Many algorithms and aspects

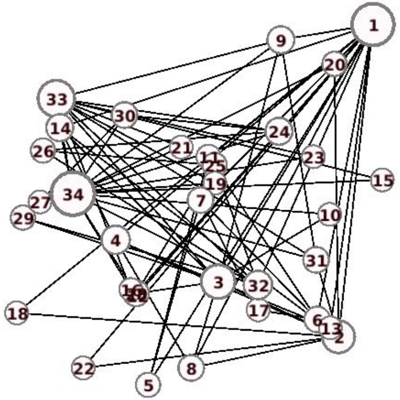
Outline

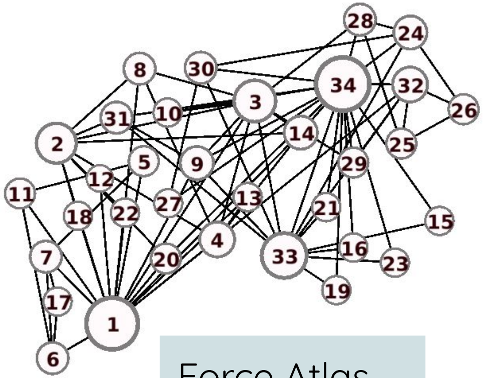
- Graph layouts
- Drawing Graphs by Eigenvectors
- **Force-Directed Layouts**
- Learning the Layout and Transition between Layouts
- Some Plotting Examples
- Graph Visualization Tools

Best choices

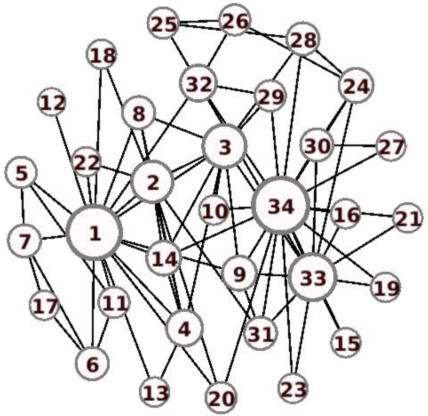
Force-directed algorithms



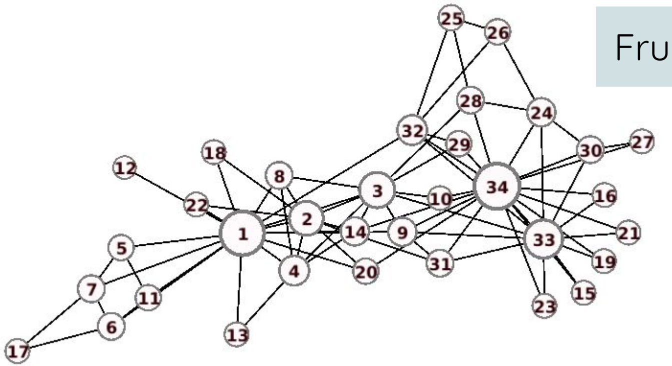




Force Atlas



Fruchterman Reingold



Yifan Hu

Physical simulations to minimize energy

- Pull edge endpoints close together
  - spring-like attractive forces on edges (Hooke's law)
- Seperate all nodes
  - electrically charged particles repulsive forces on nodes (Coulomb's law)

Edge Bundling

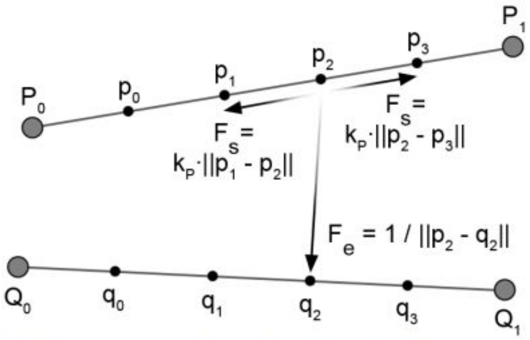


Figure 1: *Two interacting edges P and Q. The spring forces* $\mathbf{F_s}$ *and the electrostatic force* $\mathbf{F_e}$ *that are exerted on subdivision point* p_2 *by* p_1, p_3, *and* q_2 *are shown.*

[Force-Directed Edge Bundling for Graph Visualization]

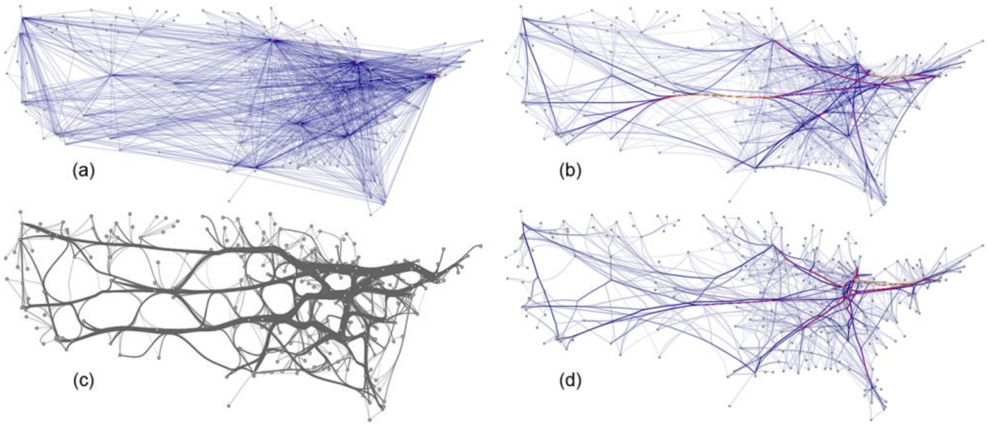


Figure 7: *US airlines graph (235 nodes, 2101 edges) (a) not bundled and bundled using (b) FDEB with inverse-linear model, (c) GBEB, and (d) FDEB with inverse-quadratic model.*

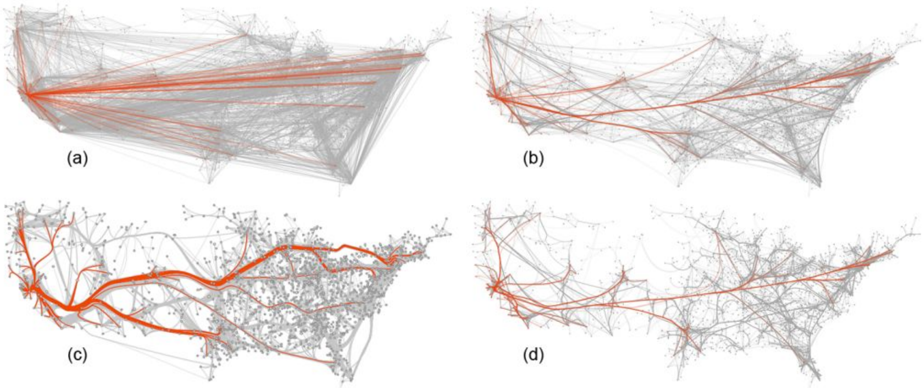


Figure 8: *US migration graph (1715 nodes, 9780 edges) (a) not bundled and bundled using (b) FDEB with inverse-linear model, (c) GBEB, and (d) FDEB with inverse-quadratic model. The same migration flow is highlighted in each graph.*

Outline

- Graph layouts
- Drawing Graphs by Eigenvectors
- Force-Directed Layouts
- **Learning the Layout and Transition between Layouts**
- Some Plotting Examples
- Graph Visualization Tools

Graph Embedding and Visualization

G → embedding/overlapping community assignments → 2D

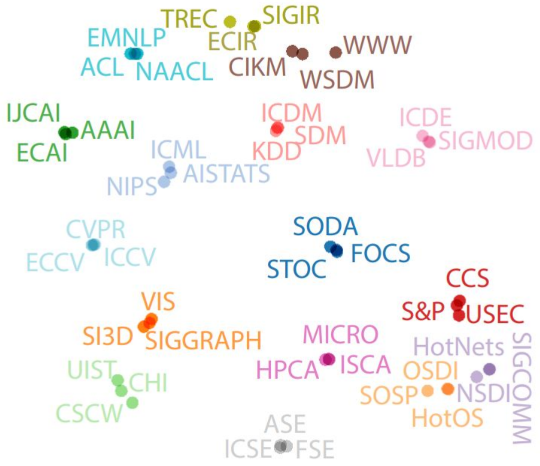
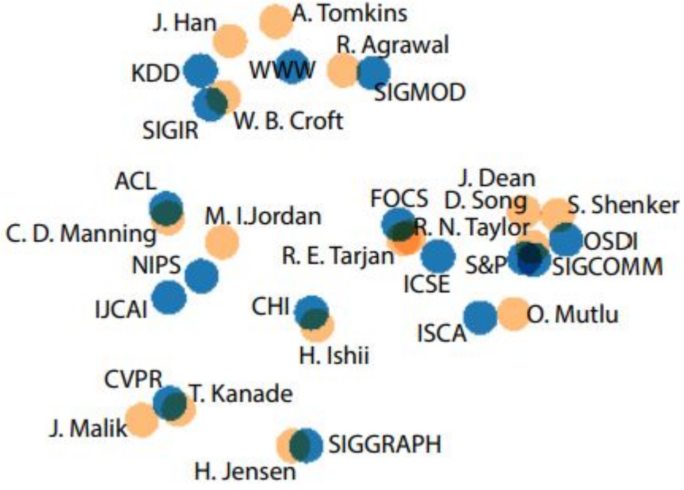


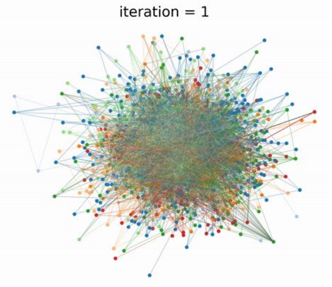
Figure 5: 2D t-SNE projections of the 128D embeddings of 48 CS venues, three each from 16 sub-fields.



(c) *metapath2vec*

[metapath2vec: Scalable Representation Learning for Heterogeneous Networks]

Learn the layout



GraphTSNE: "GCN using a modified t-SNE loss composed of two sub-losses: a graph clustering loss CG and a feature clustering loss"

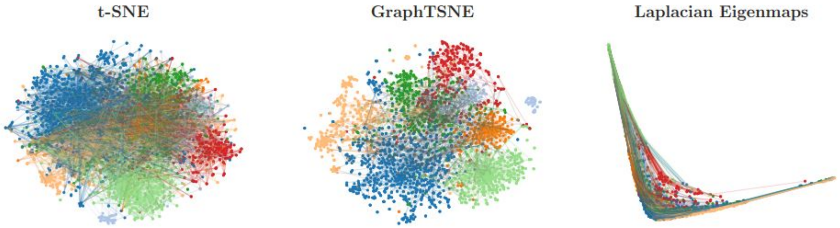


Figure 1: Three different visualizations of the CORA citation network. Compared to t-SNE (**left**) and Laplacian Eigenmaps (**right**), our proposed method GraphTSNE (**middle**) is able to produce visualizations which account for *both* graph structure and node features.

GRAPHTSNE: A VISUALIZATION TECHNIQUE FOR GRAPH-STRUCTURED DATA

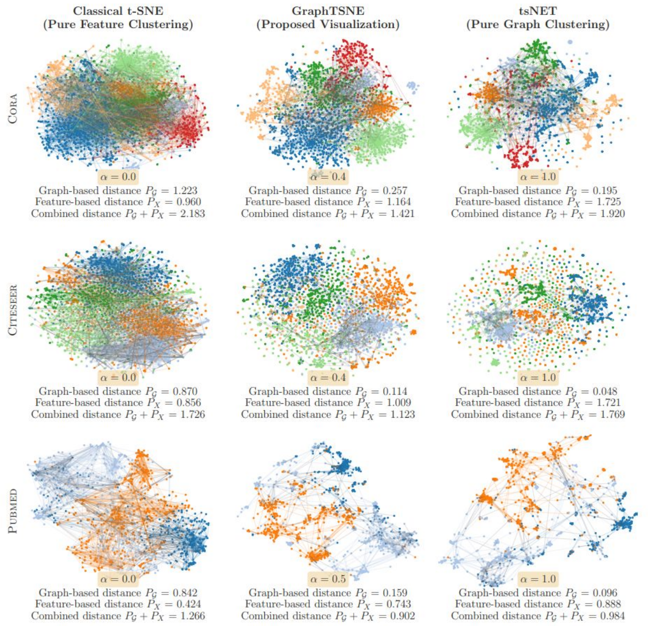
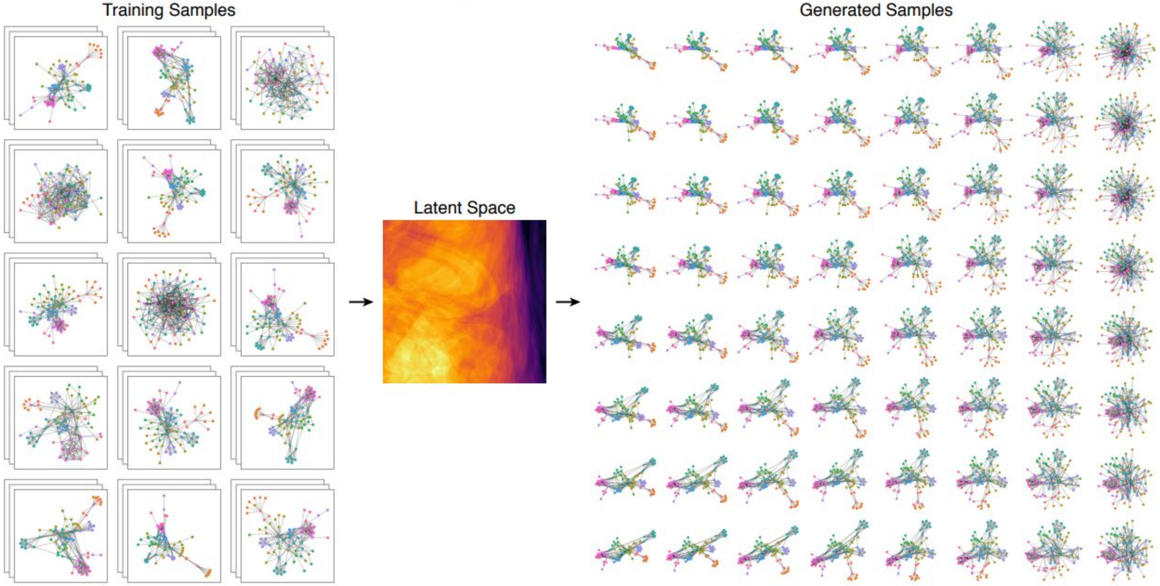


Figure 3: Comparison of visualization techniques on benchmark datasets. Colors denote document class which are not provided during training. GraphTSNE visualizations are produced with $\alpha = \alpha^*$.

Transitions between the different layouts

Demo, description

A Deep Generative Model for Graph Layout, InfoVis 2019



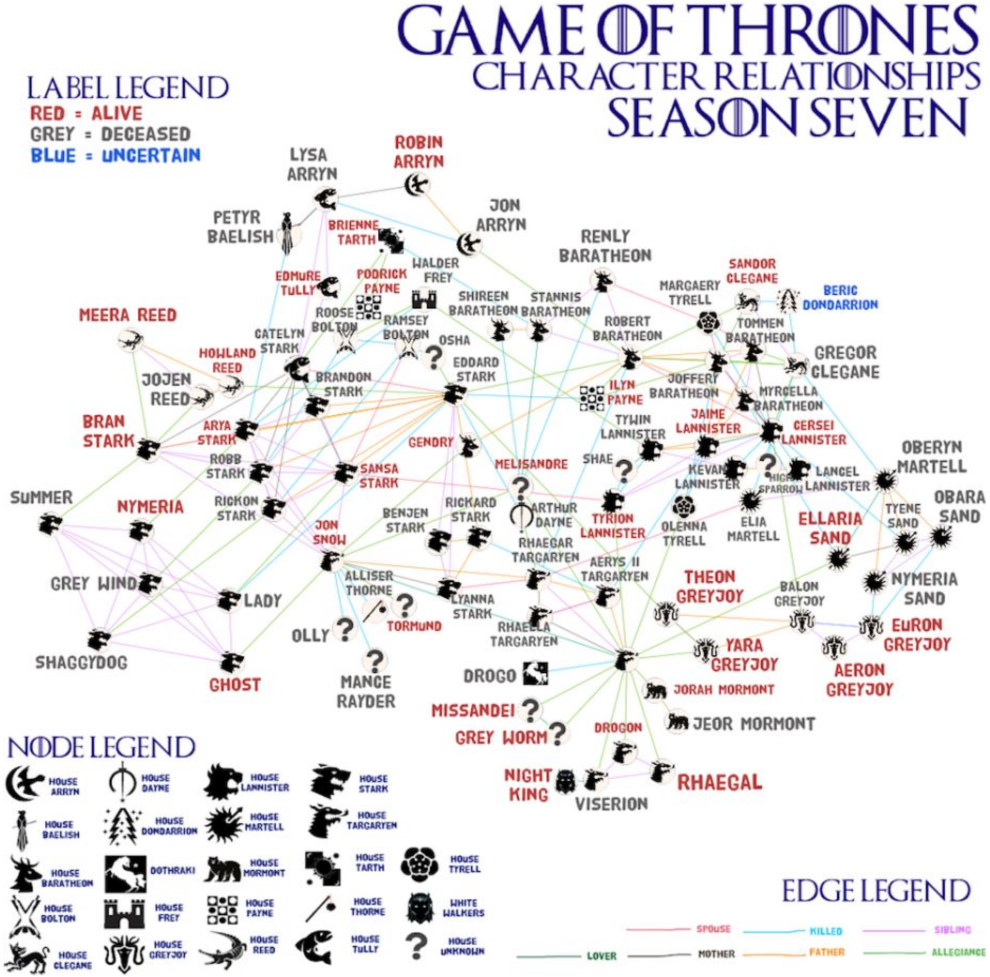
Outline

- Graph layouts
- Drawing Graphs by Eigenvectors
- Force-Directed Layouts
- Learning the Layout and Transition between Layouts
- **Some Plotting Examples**
- Graph Visualization Tools

GOT example

Annotated plot

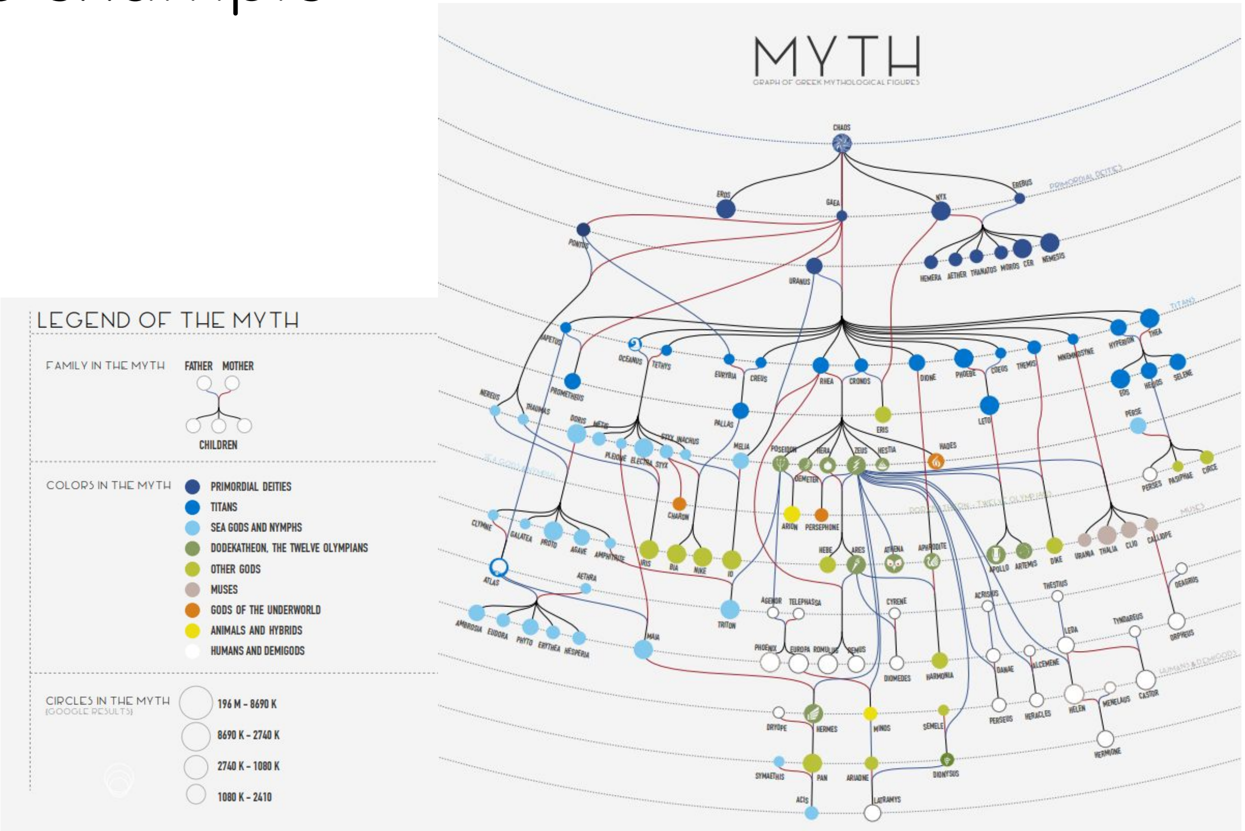
Graph Drawing Conference



Interactive demo:
https://bl.ocks.org/meltjl/raw/37a62aaa4b70e15e7753d46254b123e8/

GREEK GODs example

Hierarchical plot



Molecules example

Plotting small graphs

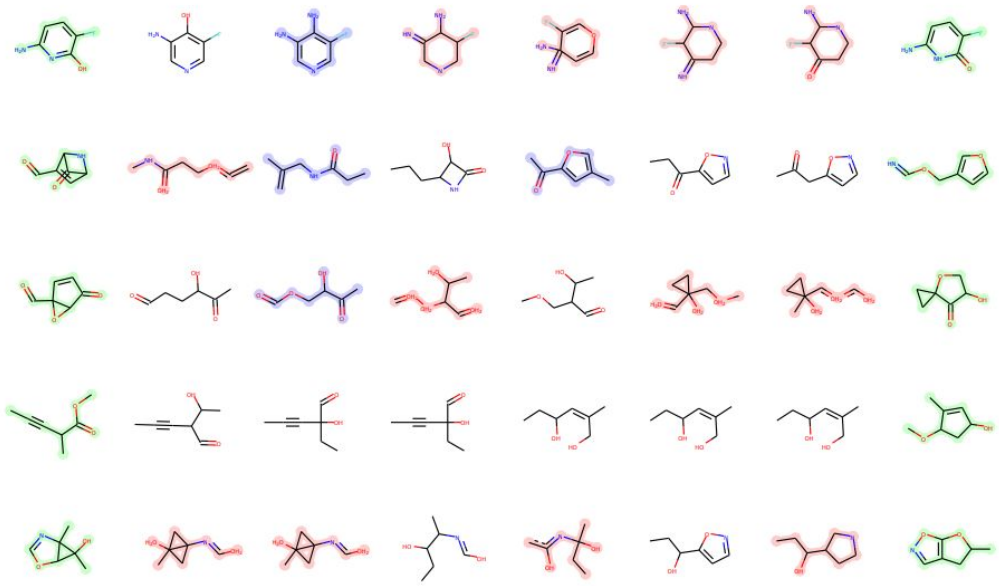
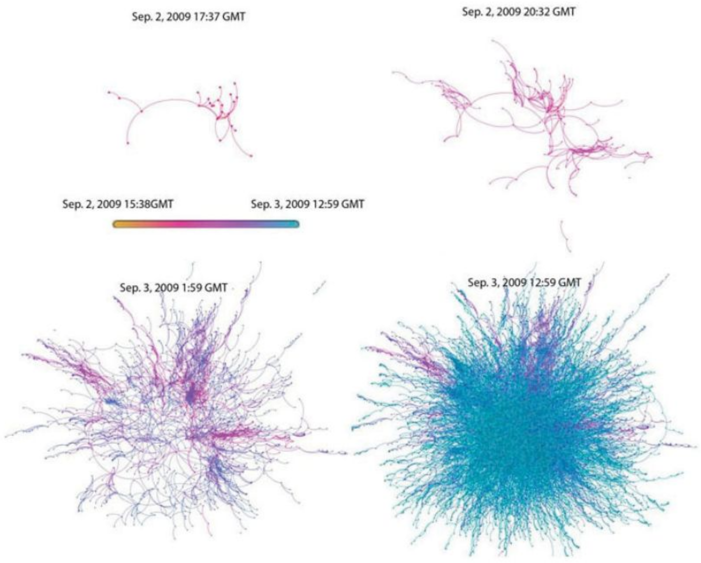


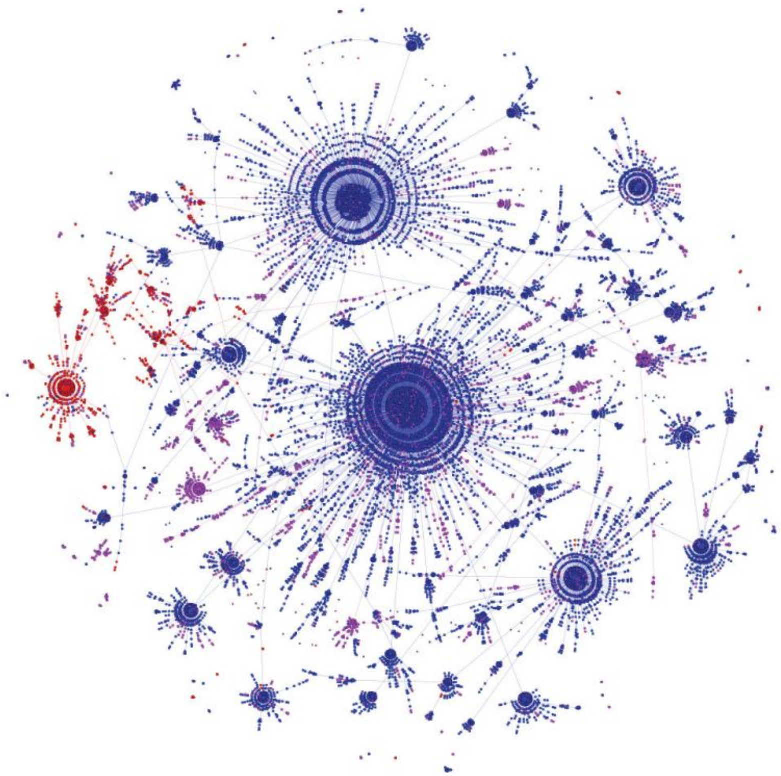
Figure 3. Linear interpolation between row-wise pairs of randomly chosen molecules in **z**-space of $c = 40$ in a conditional model. Color legend: encoder inputs (green), chemically invalid graphs (red), valid graphs with wrong label (blue), valid and correct (white).

[GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders]

Memes example

Plotting Temporal Graphs





[Information Evolution in Social Networks]

Cascades example

Plotting Diffusion cascade (copies of the same content) over time

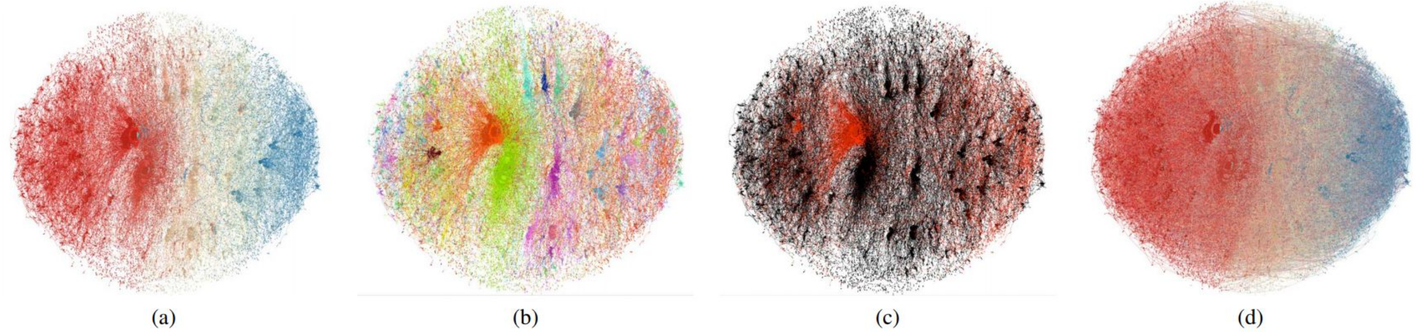
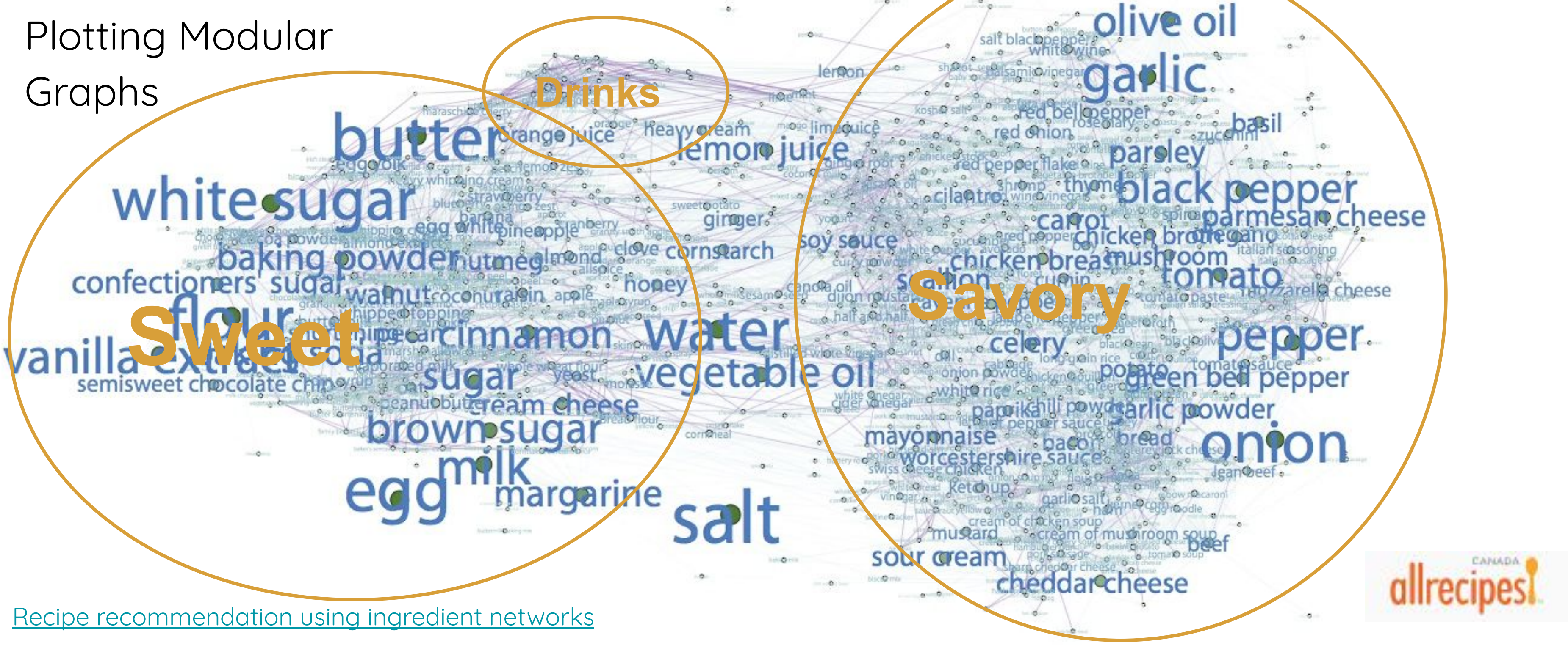


Figure 2: (a) The diffusion cascade of the example meme from Figure 1 as it spreads over time, colored from red (early) to blue (late). Only reshares that prompted subsequent reshares are shown. (b) The cascade is made up of separately introduced copies of the same content; in this drawing of the cascade from (a), each copy is represented in a different color. (c) Sometimes, individual copies experience a resurgence in popularity; again we draw the cascade from (a), but now highlight a single resurgent copy in red with the spread of all other copies depicted in black. (d) A different network on the same set of users who took part in the cascade, showing friendship edges rather than reshare edges. These edges span reshares across copies and time, showing that multiple copies of the meme are not well-separated in the friendship network.

[Do Cascades Recur?]

Ingredients example

Plotting Modular Graphs



[Recipe recommendation using ingredient networks]

Code, dependencies example

Plotting Large Graphs

http://anvaka.github.io/pm/#/?_k=32gfxk

Outline

- Graph layouts
- Drawing Graphs by Eigenvectors
- Force-Directed Layouts
- Learning the Layout and Transition between Layouts
- Some Plotting Examples
- **Graph Visualization Tools**

Graph Visualization Tools

Gephi, the common toolbox, open-source

Packages: [iGraph](#) (or see [this](#)), Networkx, [Javascript](#) and Graph databases