S&DS 365 / 665

Intermediate Machine Learning

Sparsity and Graphs

March 28

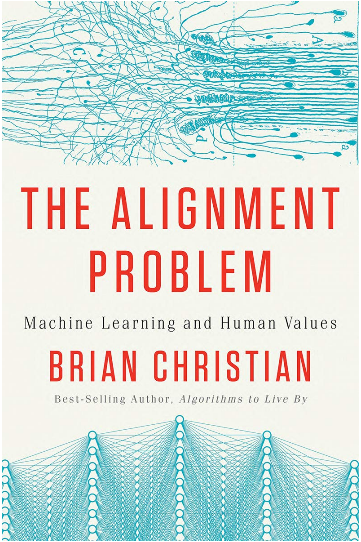
Yale

Welcome back!

For today:

- Where have we been? Where are we going?
- ~~Families of generative models~~
- Graphs of data/distributions

Announcement: Thursday at 4:30pm



Where have we been?

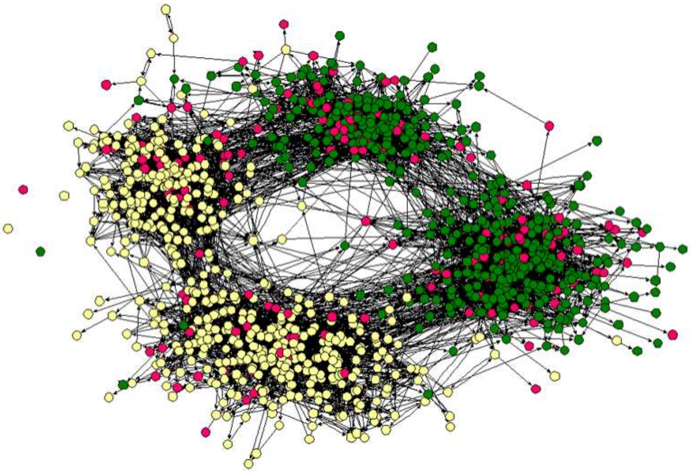
Week	Dates	Topics	Demos & Tutorials	Lecture Slides	Readings & Notes	Assignments & Exams
1	Jan 26, 28	Course overview	CO Python elements CO Pandas and regression CO Lasso example	Jan 26: Course overview Jan 28: Sparse regression	PML Section 11.4	
2	Jan 31, Feb 2	Smoothing and kernels	CO Smoothing example CO Using different kernels CO Mercer kernels	Jan 31: Smoothing Feb 2: Mercer Kernels	PML Sections 16.3, 17.1 Notes on Mercer kernels	
3	Feb 7, 9	Density estimation and risk bounds	CO Density estimation demo	Feb 7, 9: Sync up	Bias-variance tradeoff for density estimation	Feb 9: CO Assn1 out
4	Feb 14, 16	Neural networks for classification	TensorFlow playground CO Convolution demo CO Problem 4 warmup	Feb 9: Neural networks Feb 14: Convolutional neural networks Feb 16: CNNs continued	PML Sections 13.1, 13.2 Notes on backpropagation	Feb 16: Quiz 1
5	Feb 21, 23	Nonparametric Bayes	CO Parametric Bayes CO Gaussian processes CO Dirichlet processes	Feb 21: Gaussian processes Feb 23: Gaussian and Dirichlet processes	PML Section 17.2 Notes on Bayesian inference Notes on nonparametric Bayes	Feb 23: Assn 1 in; CO Assn2 out
6	Feb 28, Mar 2	Gibbs sampling	CO DP demo, ver. 2 Gibbs sampling demo (.mp4) (.mov)	Feb 28: Dirichlet processes Mar 2: Gibbs sampling	Notes on Gibbs sampling	Mar 2: Quiz 2
7	Mar 7, 9	Variational inference	CO Variational autoencoders	Mar 7: Introduction to approximate inference Mar 9: Variational inference and VAEs	PML Section 20.3 Notes on variational inference	Mar 9: Assn 2 in
8	Mar 14, 16	Review and midterm		Mar 14: VAEs and review Mar 16: Midterm	Practice midterm	Mar 16: Midterm exam

Where are we going?

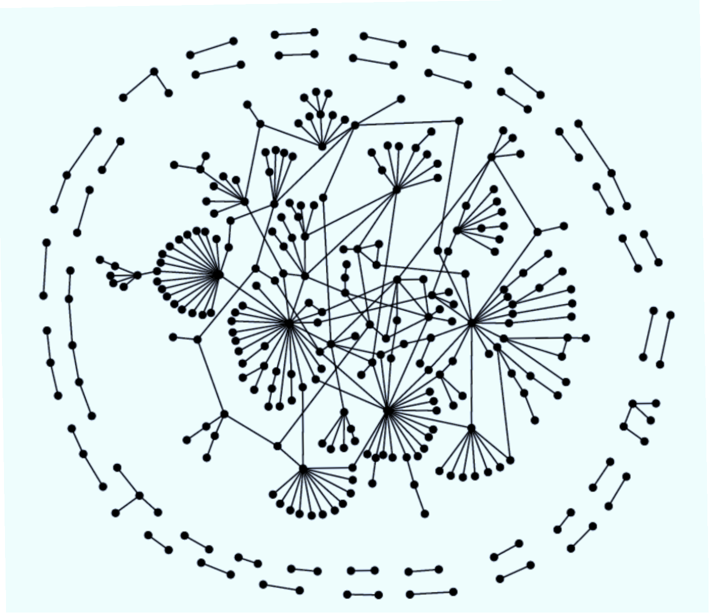
9	Mar 28, 30	Graphs and structure learning	CO Graphical lasso demo	Mar 28: Sparsity and graphs Mar 30: Discrete data and graph neural nets	Notes on graphs and structure learning PML Section 23.4	Mar 30: Assn 3 out
10	Apr 4, 6	Deep reinforcement learning	CO Q-learning	Apr 4: Reinforcement learning Apr 6: Deep reinforcement learning		Apr 6: Quiz 3
11	Apr 11, 13	Policy gradient methods				Apr 13: Assn 3 in; Assn 4 out
12	Apr 18, 20	Sequential and sequence-to-sequence models				Apr 20: Quiz 4
13	Apr 25, 27	Attention and language models				Apr 27: Assn 4 in
	May 7	Final exam, 2pm location TBD				Registrar: final exam schedule

Graphs

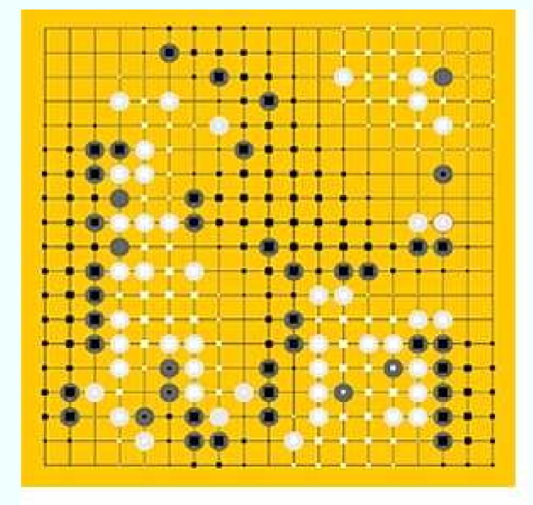
- A natural language for describing various data
- Give information about relationships between variables
- Associated with each multivariate distribution



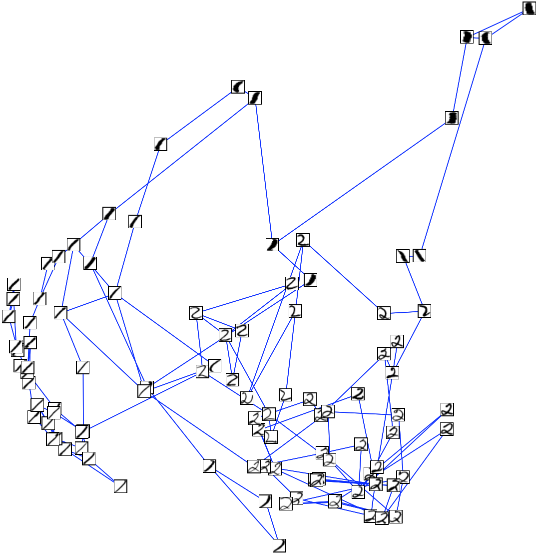
social networks



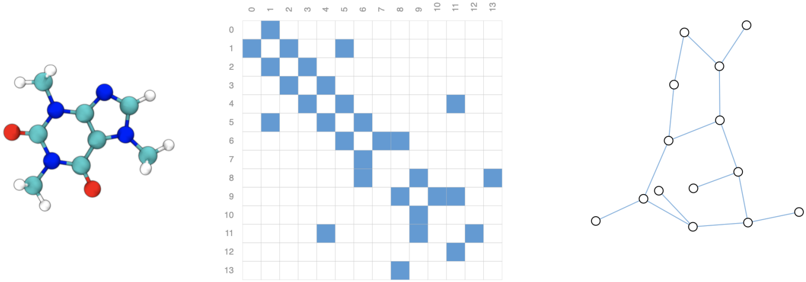
protein networks



games

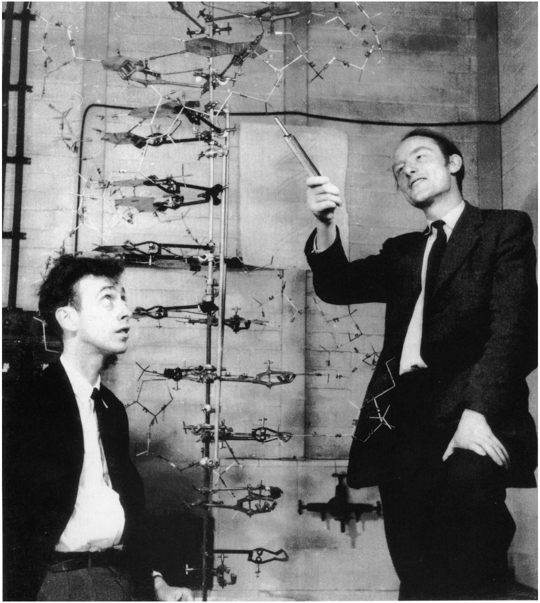


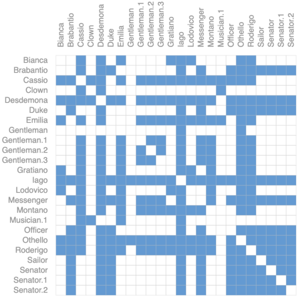
semi-supervised learning

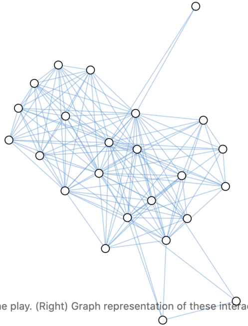


(Left) 3d representation of the Caffeine molecule (Center) Adjacency matrix of the bonds in the molecule (Right) Graph representation of the molecule.

https://distill.pub/2021/gnn-intro/



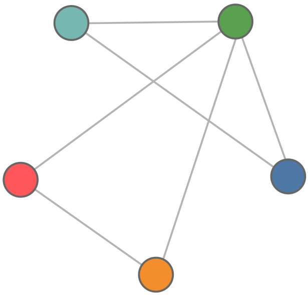




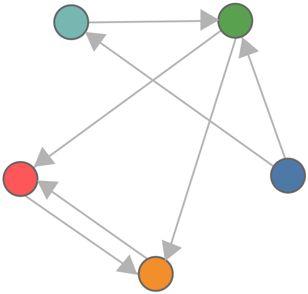
(Left) Image of a scene from the play "Othello". (Center) Adjacency matrix of the interaction between characters in the play. (Right) Graph representation of these interactions.

https://distill.pub/2021/gnn-intro/

Undirected graph



Directed graph



Undirected Graphs

A graph $G = (V, E)$ has vertices V, edges E.

If $X = (X_1, \ldots, X_p)$ is a random variable, we will study graphs where there are p vertices, one for each X_j.

The graph will encode conditional independence relations among the variables.

Graphs for data/distributions

- Graphs give us a new way of understanding data
- Allow us to make structural assumptions
- Central to causal inference

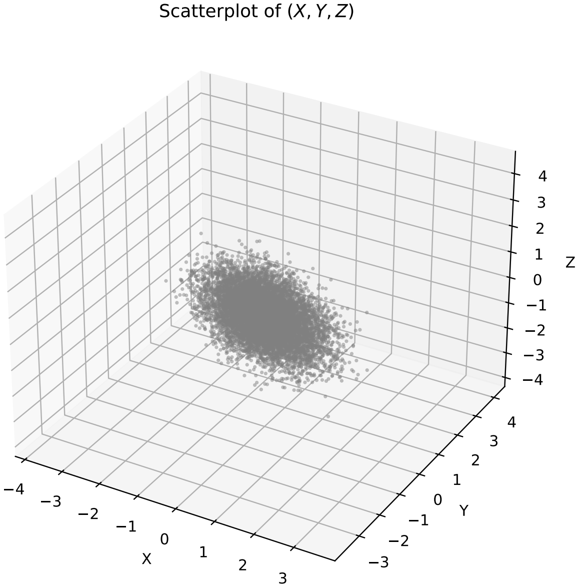
Example: Gaussian data

We have a three-dimensional Gaussian (X, Y, Z) with covariance

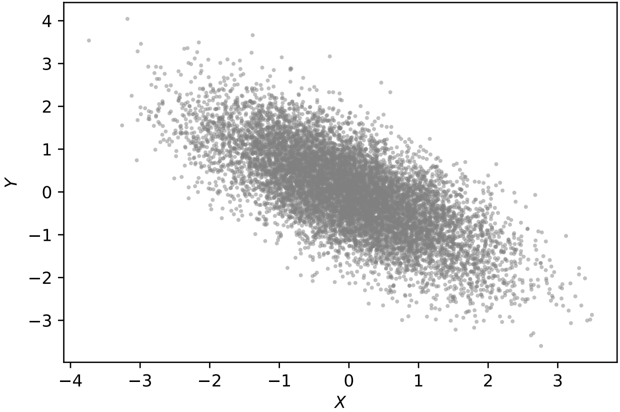
$$\Sigma = \begin{pmatrix} 3.1 & -2.4 & 2.1 \\ -2.4 & 2.6 & -2.4 \\ 2.1 & -2.4 & 3.1 \end{pmatrix}$$

So, all pairs are correlated

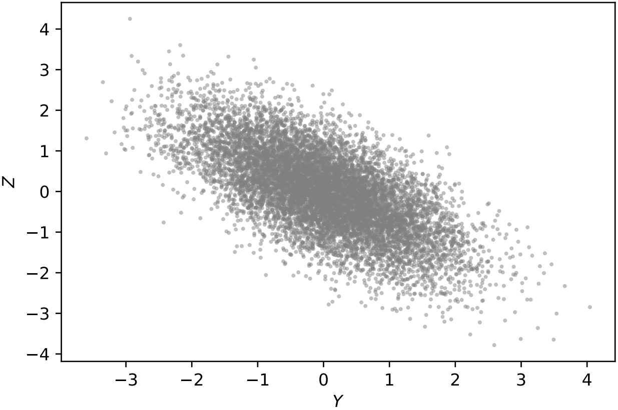
Example: Gaussian data



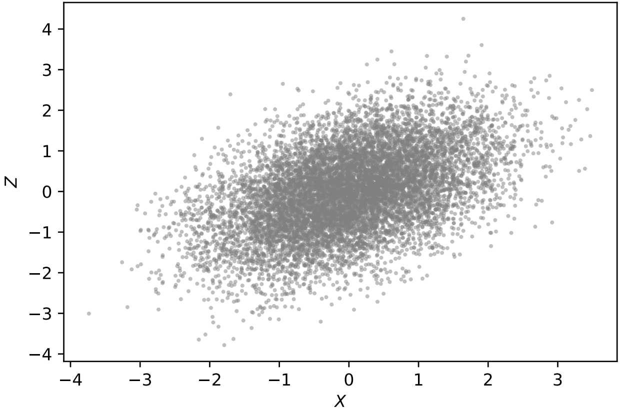
Example: Gaussian data



Example: Gaussian data

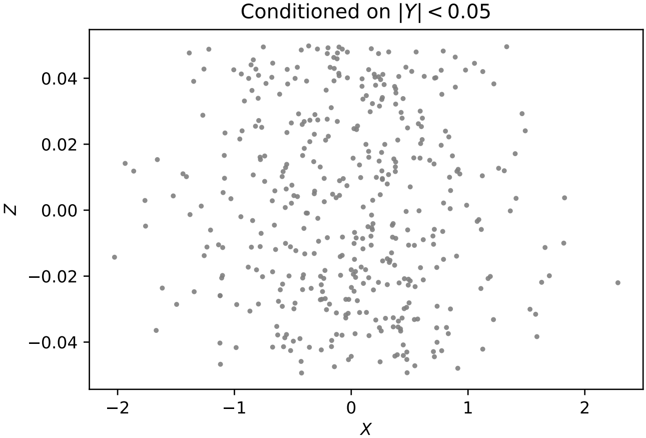


Example: Gaussian data



Example: Gaussian data

But when we condition on $Y \approx 0$:



Gaussian example

This is revealed in the "precision matrix"

$$\Omega \equiv \Sigma^{-1} = \begin{pmatrix} 1 & \frac{9}{10} & 0 \\ \frac{9}{10} & 2 & \frac{9}{10} \\ 0 & \frac{9}{10} & 1 \end{pmatrix}$$

The zeros lead to <u>conditional</u> independence assumptions

Undirected graphs

Simplest case:

Here $V = \{X, Y, Z\}$ and $E = \{(X, Y), (Y, Z)\}$.

This encodes the independence relation

$$X \perp\!\!\!\perp Z \mid Y$$

which means that *X and Z are independent conditioned on Y*.

Markov Property

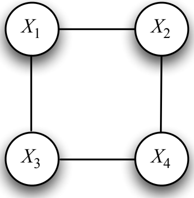
A probability distribution P satisfies the *global Markov property* with respect to a graph G if:

for any disjoint vertex subsets A, B, and C such that C separates A and B,

$$X_A \perp\!\!\!\perp X_B \mid X_C.$$

- X_A are the random variables X_j with $j \in A$.
- C separates A and B means that there is no path from A to B that does not pass through C.

Example

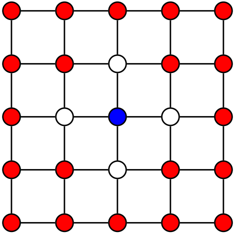


$$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$$

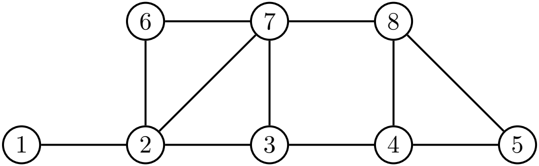
$$X_2 \perp\!\!\!\perp X_3 \mid X_1, X_4$$

Example: 2-dimensional grid

The blue node is independent of the red nodes given the white nodes.



Example



$C = \{3, 7\}$ separates $A = \{1, 2\}$ and $B = \{4, 8\}$. Hence,

$$\{X_1, X_2\} \perp\!\!\!\perp \{X_4, X_8\} \quad \Big| \quad \{X_3, X_7\}$$

Special case

If $(i, j) \notin E$ then

$$X_i \perp\!\!\!\perp X_j \;\Big|\; \{X_k : k \neq i, j\}$$

Special case

If $(i, j) \notin E$ then

$$X_i \perp\!\!\!\perp X_j \quad \Big| \quad \{X_k : k \neq i, j\}$$

Lack of an edge from i to j implies that X_i and X_j are independent given all of the other random variables.

Graph estimation

- A graph G represents the class of distributions, $\mathcal{P}(G)$, the distributions that are Markov with respect to G
- Graph estimation: Given n samples $X_1, \ldots, X_n \sim P$, estimate the graph G.

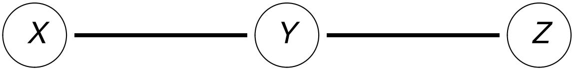
Gaussian case

Let $\Omega = \Sigma^{-1}$ be the precision matrix.

A zero in Ω indicates a *lack of the corresponding edge* in the graph

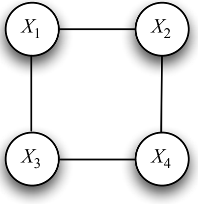
Gaussian case

$$\Omega \equiv \Sigma^{-1} = \begin{pmatrix} * & * & 0 \\ * & * & * \\ 0 & * & * \end{pmatrix}$$



Gaussian case

$$\Omega \equiv \Sigma^{-1} = \begin{pmatrix} * & * & * & 0 \\ * & * & 0 & * \\ * & 0 & * & * \\ 0 & * & * & * \end{pmatrix}$$



$$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$$

The machine learning problem

How do we estimate the graph from a sample of data?

Gaussian case: Algorithms

Two approaches:

- parallel lasso
- graphical lasso

Parallel Lasso:

1. For each $j = 1, \ldots, p$ (in parallel): Regress X_j on all other variables using the lasso.
2. Put an edge between X_i and X_j if each appears in the regression of the other.

Graphical Lasso (glasso)

- Assume a multivariate Gaussian model
- Subtract out the sample mean
- Minimize the negative log-likelihood of the data, subject to a constraint on the sum of the absolute values of the inverse covariance

Graphical Lasso (glasso)

The glasso optimizes the parameters of $\Omega = \Sigma^{-1}$ by minimizing:

$$\mathrm{trace}(\Omega S_n) - \log|\Omega| + \lambda \sum_{j \neq k} |\Omega_{jk}|$$

where $|\Omega|$ is the determinant and S_n is the sample covariance

$$S_n = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T$$

Derivation: Where does this come from?

Assume mean is zero. Then the probability density at a data point x is

$$\begin{aligned} p(x) &= \frac{1}{\sqrt{(2\pi)^p|\Sigma|}} \exp\left(-\frac{1}{2}x^T\Sigma^{-1}x\right) \\ &= \frac{1}{\sqrt{(2\pi)^p|\Sigma|}} \exp\left(-\frac{1}{2}x^T\Omega x\right) \\ &= \frac{1}{\sqrt{(2\pi)^p|\Sigma|}} \exp\left(-\frac{1}{2}\text{trace}(\Omega x x^T)\right) \end{aligned}$$

Therefore, using $\log|A| = -\log|A^{-1}|$, up to an additive constant,

$$-\log p(x) = \frac{1}{2}\log|\Sigma| + \frac{1}{2}\text{trace}(\Omega x x^T) = -\frac{1}{2}\log|\Omega| + \frac{1}{2}\text{trace}(\Omega x x^T)$$

Derivation: Where does this come from?

Summing over all the data we have

$$-\sum_{i=1}^{n} \log p(x_i) = \frac{1}{2}\sum_{i=1}^{n} \text{trace}(\Omega x_i x_i^T) - \frac{n}{2}\log|\Omega|$$
$$= \frac{n}{2}\text{trace}(\Omega S_n) - \frac{n}{2}\log|\Omega|$$

Rescaling by $2/n$ and adding the ℓ_1 penalty, we get the objective function

$$\mathcal{O}(\Omega) = \text{trace}(\Omega S_n) - \log|\Omega| + \lambda \sum_{k\neq j} |\Omega_{jk}|$$

This is a convex function of Ω

Graphical Lasso (glasso)

There is a simple blockwise gradient descent algorithm for minimizing this function. It is similar to the algorithm for the lasso that we studied.

Python packages: `sklearn.covariance.GraphicalLasso` and `sklearn.covariance.GraphicalLassoCV`

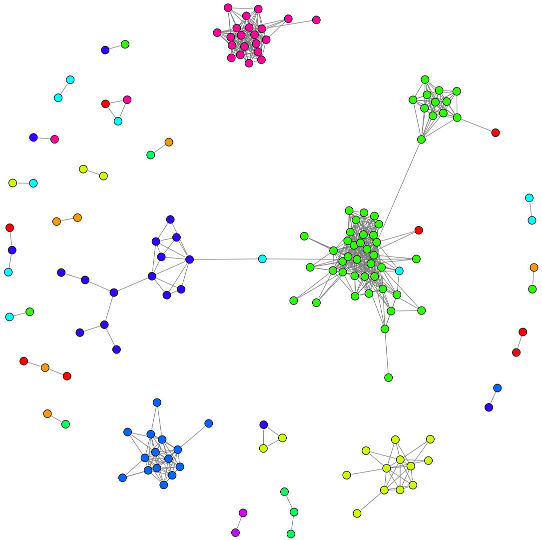
For a derivation of this algorithm, see “Notes on Graphs and Structure Learning”

Graphs on the S&P 500

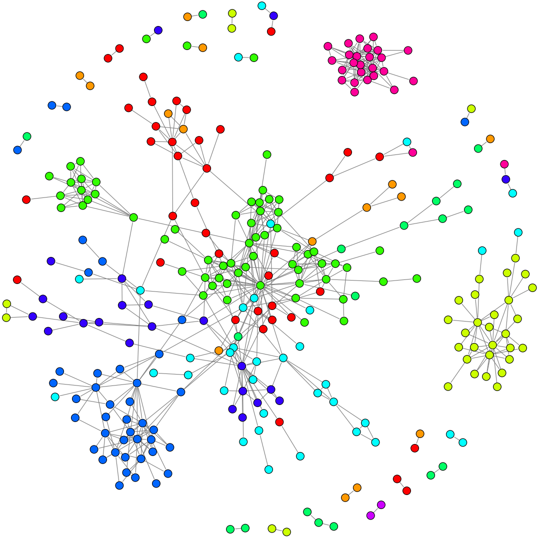
- Data from Yahoo! Finance (`finance.yahoo.com`).
- Daily closing prices for 452 stocks in the S&P 500 between 2003 and 2008 (before onset of the "financial crisis").
- Log returns $X_{tj} = \log\left(S_{t,j}/S_{t-1,j}\right)$.
- Outliers capped at $\pm 6\sigma$.
- In following graphs, each node is a stock, and color indicates an industry sector

```
Consumer Discretionary          Consumer Staples
Energy                          Financials
Health Care                     Industrials
Information Technology          Materials
Telecommunications Services     Utilities
```

S&P 500: Graphical Lasso



S&P 500: Parallel Lasso



Example Neighborhood

Yahoo Inc. (Information Technology):

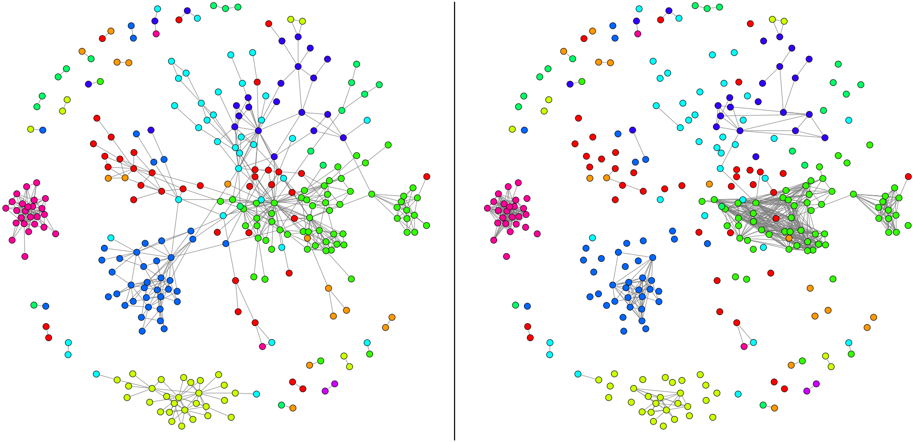
- Amazon.com Inc. (Consumer Discretionary)
- eBay Inc. (Information Technology)
- NetApp (Information Technology)

Example Neighborhood

Target Corp. (Consumer Discretionary):

- Big Lots, Inc. (Consumer Discretionary)
- Costco Co. (Consumer Staples)
- Family Dollar Stores (Consumer Discretionary)
- Kohl's Corp. (Consumer Discretionary)
- Lowe's Cos. (Consumer Discretionary)
- Macy's Inc. (Consumer Discretionary)
- Wal-Mart Stores (Consumer Staples)

Parallel vs. Graphical



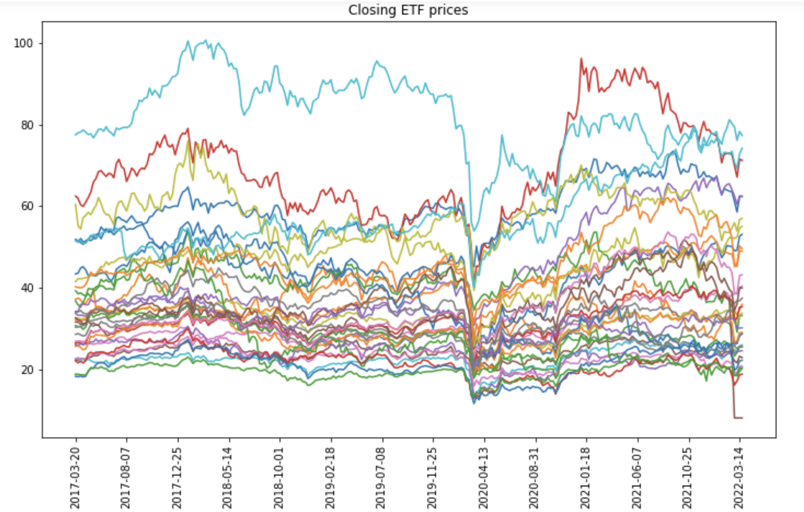
Choosing λ

Can use:

1. Cross-validation
2. BIC = log-likelihood - $(p/2)\log n$
3. AIC = log-likelihood - p

where p = number of parameters.

Let's go to the demo!



Summary

- Graphs encode conditional independence assumptions
- Sparse graphs represent low-dimensional structure in high dimensional data
- Gaussian case: Graph read off from precision matrix
- Graphical lasso used to estimate the graph