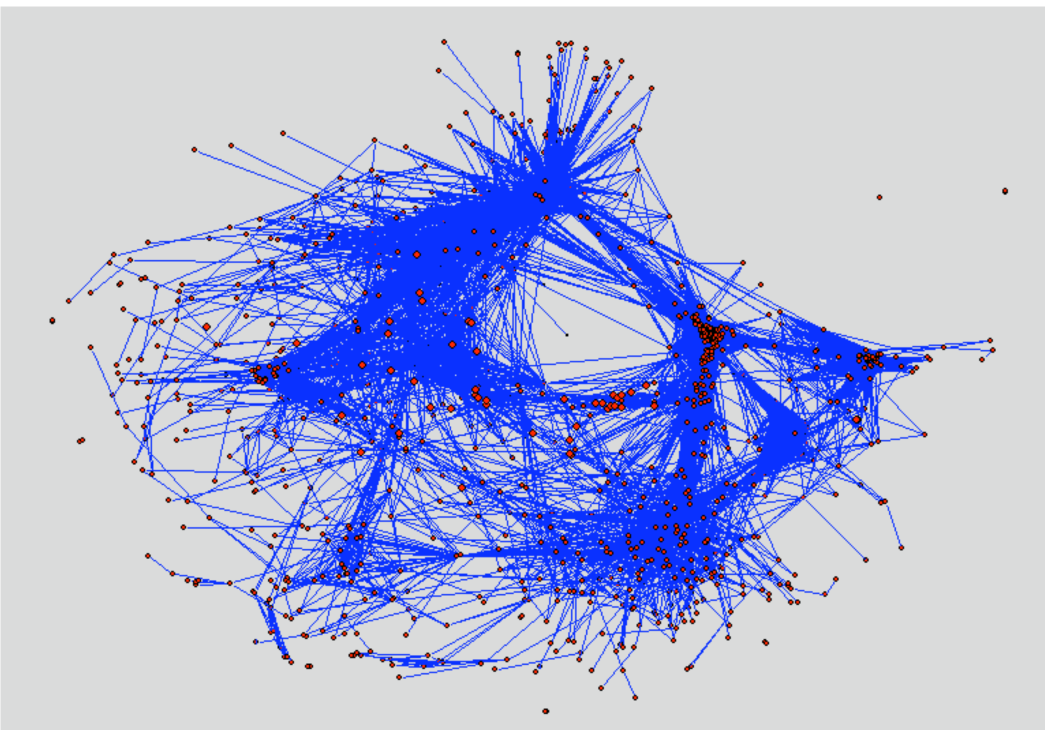
CSS07
Pattern recognition's role in the articulation and validation of Complex Structures
Part 2: Supervised Learning



New York Stock Exchange Network

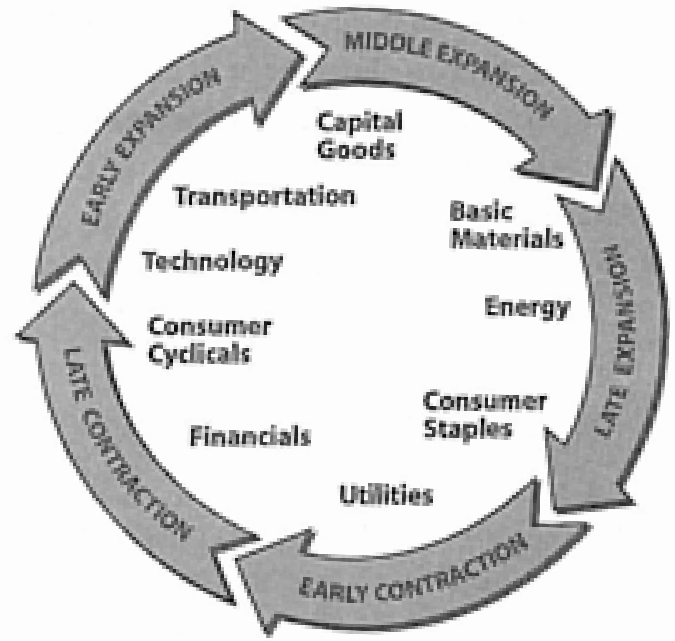


FIGURE 13.1 Technology and transportation leadership during 2003 fits Early Expansion phase.

Known cyclic structure

Gregory Leibon
Dartmouth College

A VERY basics of supervised learning

(See: *Pattern Classification* by Duda, Hart, and Stork OR *The Elements of Statistical Learning* by Hastie, Tibshirani, Friedman):

Suppose there are two possible sectors denoted k=1 and k=2, and real valued function X at a fixed time which depends on this sector.

Goal: Given X we would like to predict k's value.

To accomplish this goal, imagine we have data in the form: $\{(x_i, k_i)\}_{i=1}^{140}$

The desire is use the data to estimate the unknown density: $f_X(x,k)$

If we could approximate

$$f_X(x_0 \mid k) \quad \text{and} \quad p(k = K)$$

then using Bayes' Theorem

$$P(k=1 \mid x = x_0) = \frac{f_{X,h}(x_0 \mid 1)P(k=1)}{f_{X,h}(x_0 \mid 1)P(k=1) + f_{X,h}(x_0 \mid 2)P(k=2)}$$

$$P(k=2 \mid x = x_0) = \frac{f_{X,h}(x_0 \mid 2)P(k=2)}{f_{X,h}(x_0 1)P(k=1) + f_{X,h}(x_0 \mid 2)P(k=2)}$$

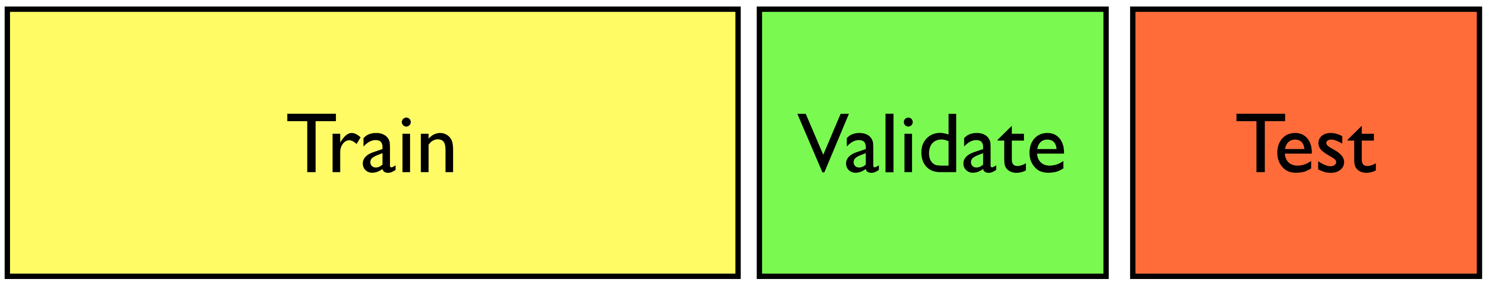
and we cab declare the stock x_0 to be of type k=1 if

$$P(k=1 \mid x = x_0) > P(k=2 \mid x = x_0)$$

and declare x_0 to be of type 2 otherwise.

How to do this?

DATA



$$\{(x_i,k_i)\}_{i=1}^{140} = \{(x_i,k_i)\}_{i=1}^{70} \bigcup \{(x_i,k_i)\}_{i=71}^{105} \bigcup \{(x_i,k_i)\}_{i=106}^{140}$$

Train some collection of models. Maybe initially:

Parzen Window Approximation

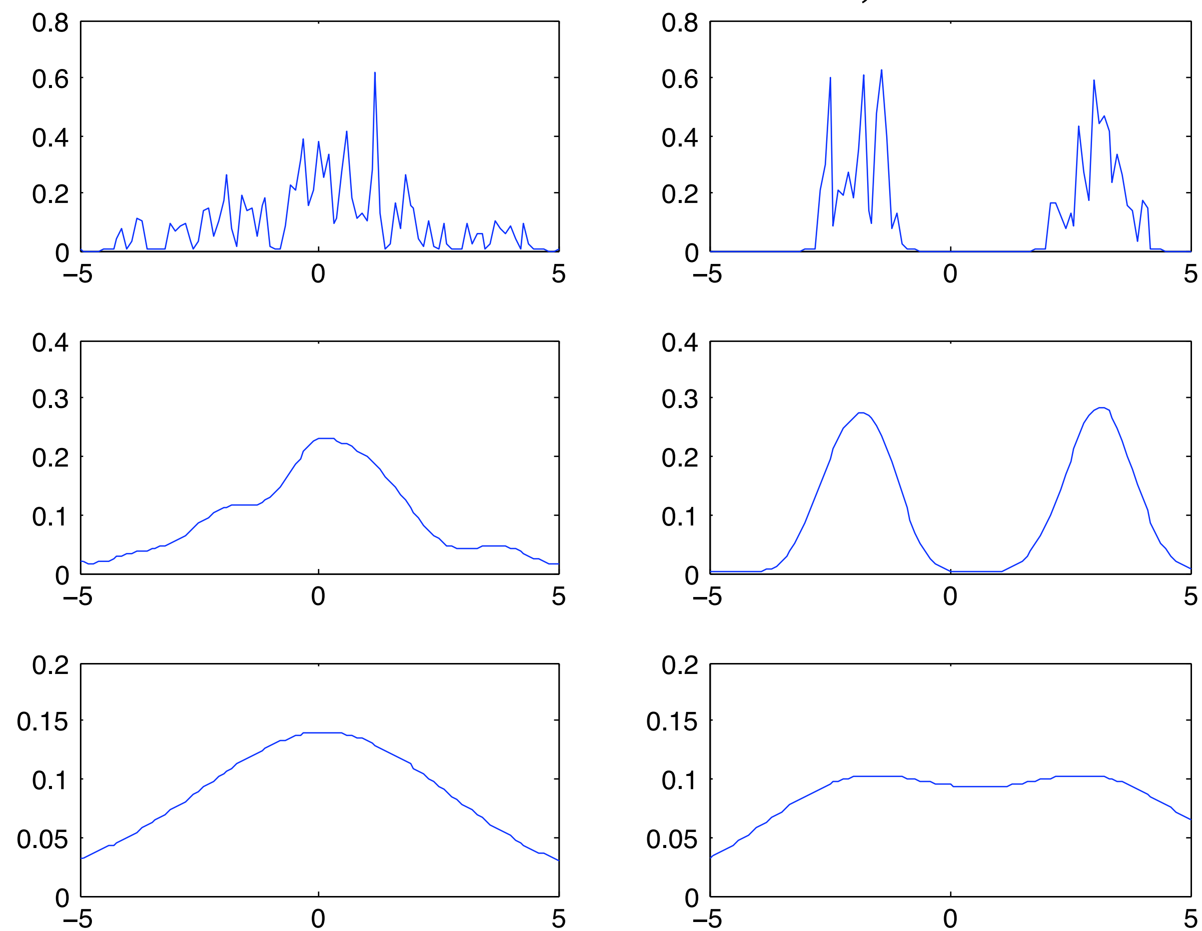
$$f_{X,h}(x \mid K) = \frac{1}{\mid k_i = K \mid} \sum_{i \mid k_i = K} \frac{e^{\frac{-(x-x_i)^2}{2h^2}}}{h\sqrt{2\pi}}$$

$f_{X,h}(x \mid 1)$ $\qquad$ $f_{X,h}(x \mid 2)$

h=1/20

h=1/2

h=2



In MatLab

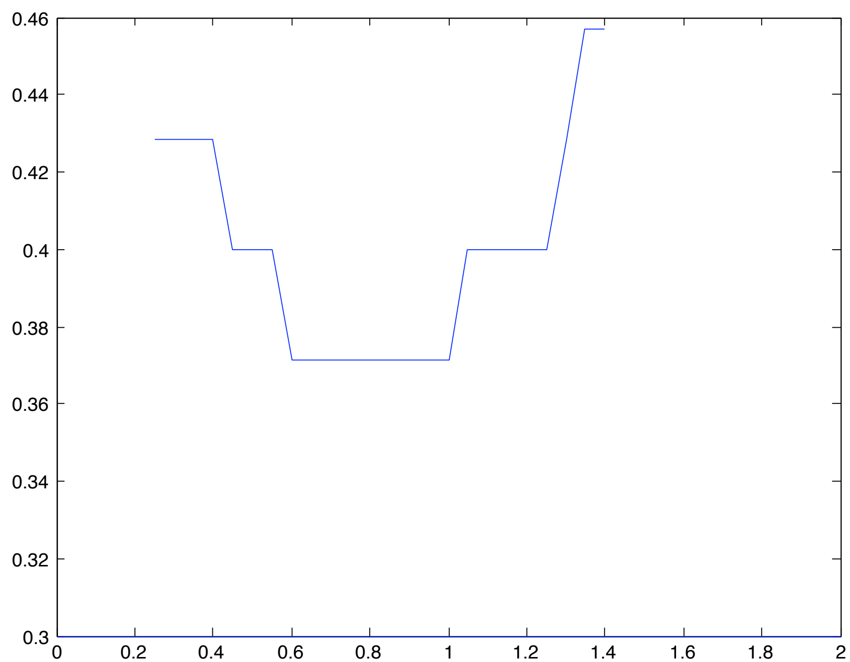
```
Interval=-5:.1:5;

h=.8;

[fX,xi2] = ksdensity(Data,Interval,'width',h,'kernel','normal');
```

NO FREE LUNCH!

To fit the distribution, you could and should try other local methods: other Kernels, nearest neighbor approximations,... as well as global (parametric) models like Gaussian, Mixture Gaussian, Polynomial fitting....

To decide which model to eventually go with, we use our validation data:

Percent Incorrect



Window width=h

Finally, using the test data and see with it there is roughly 60% success rate with h=0.8.

How to think about the error?

Imagine x is fixed...

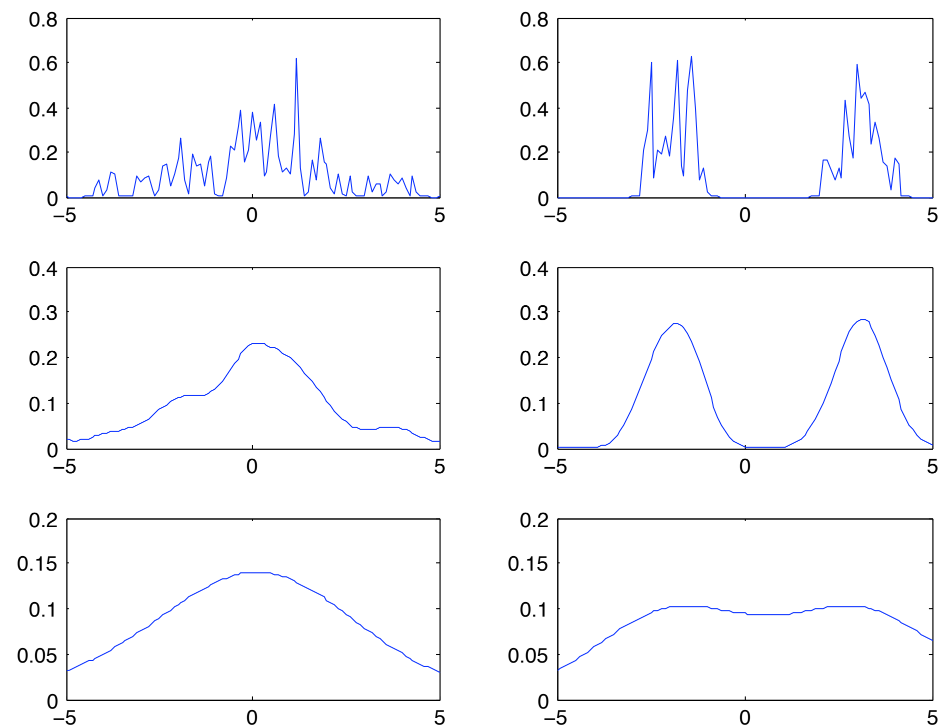
$$E\left(\left(f_h(x)-f(x)\right)^2\right)=\left(E\left(f_h(x)\right)-f(x)\right)^2+E\left(\left(f_h(x)-E\left(f_h(x)\right)\right)^2\right)$$

Error = Bias + Variance

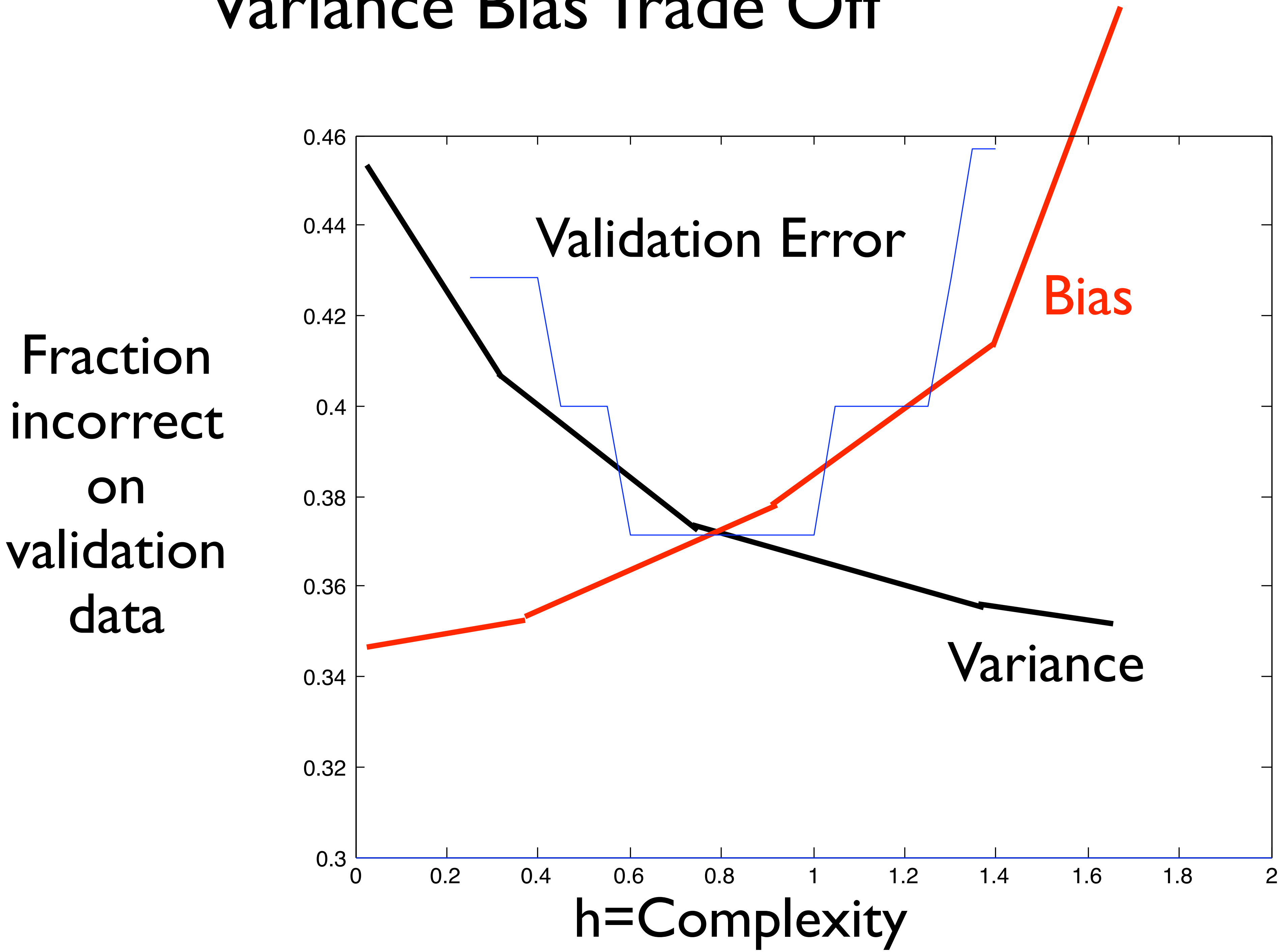
For

$$h << 1$$

$$E(f_h(x)) \approx f(x)$$

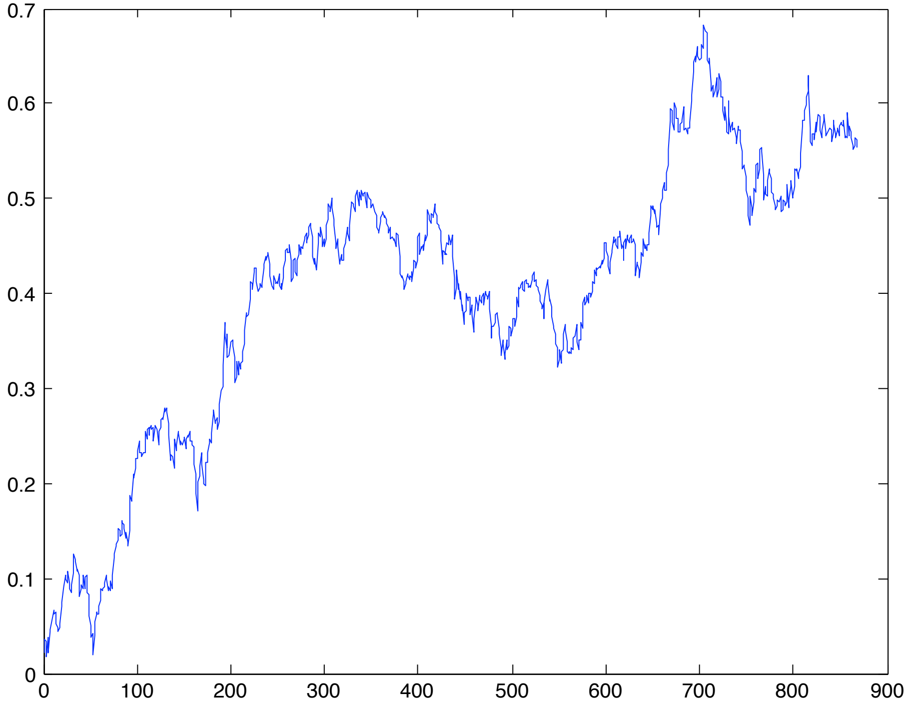


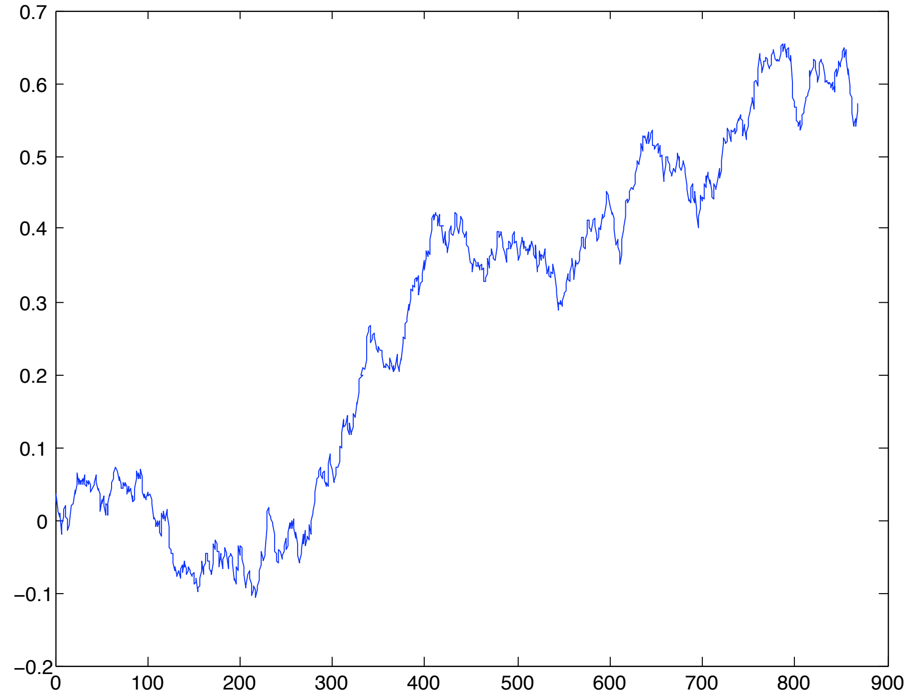
Variance Bias Trade Off



Recall our Equity's Times Series

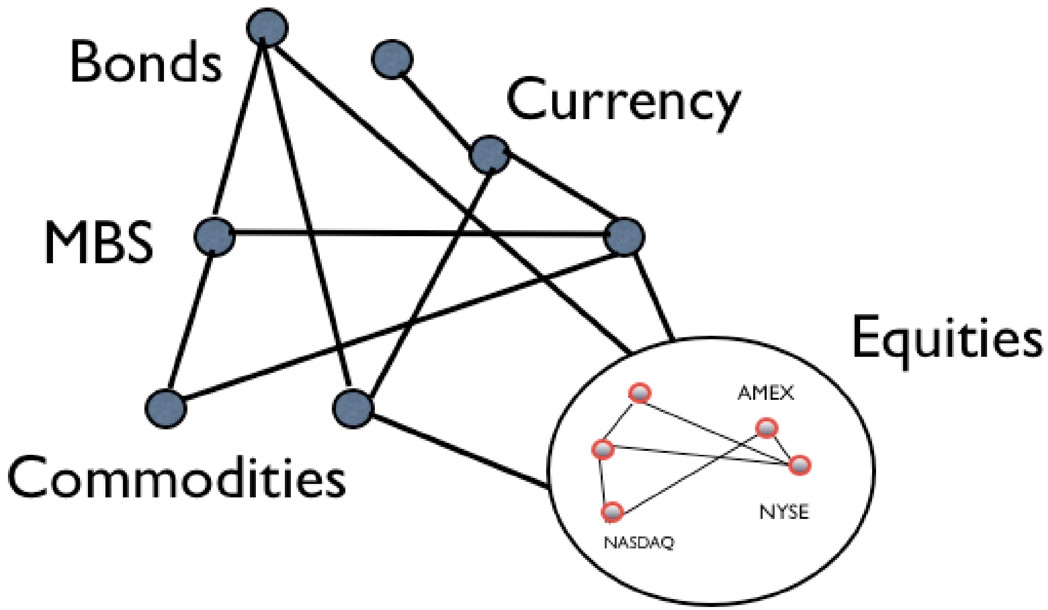
$$X_t = \frac{C_t - C_{t-1}}{C_{t-1}} \approx d(\ln(S_t))$$

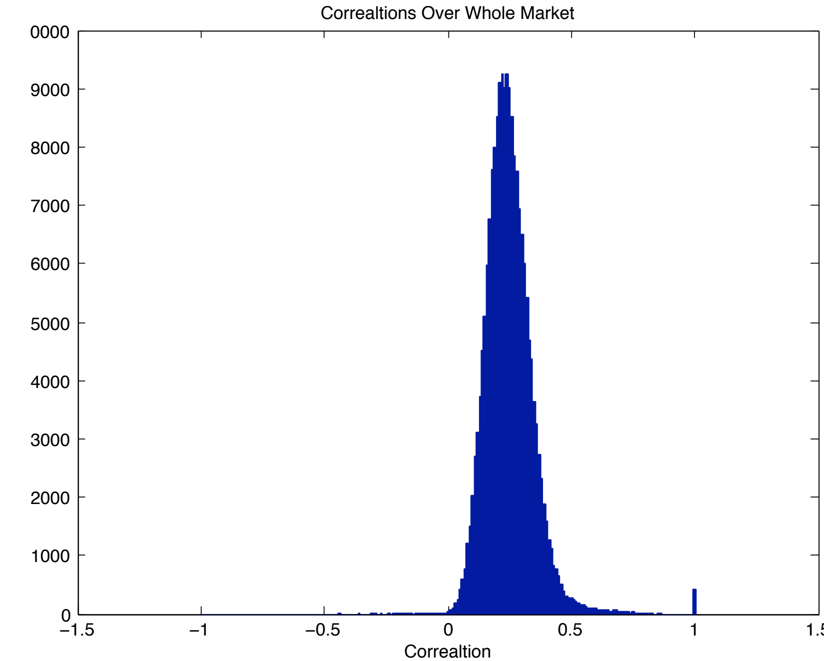




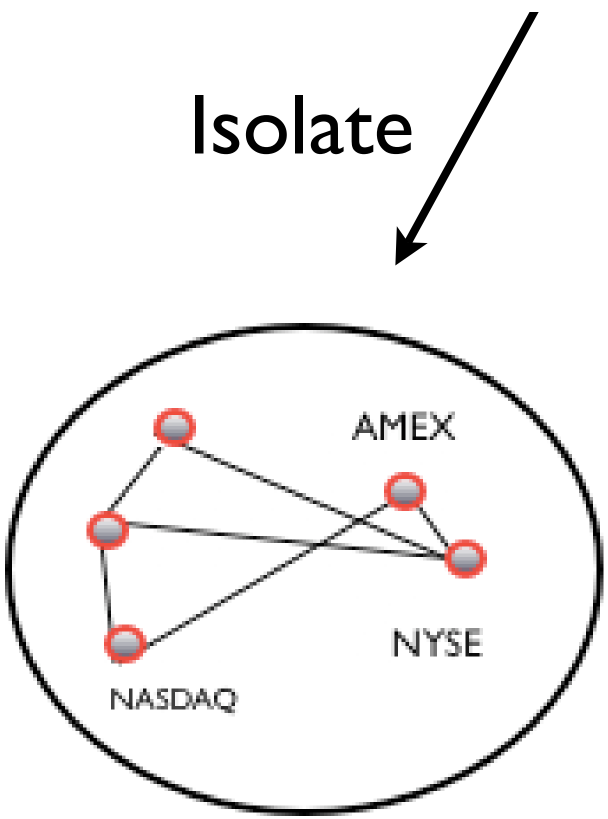
A rough approximation:

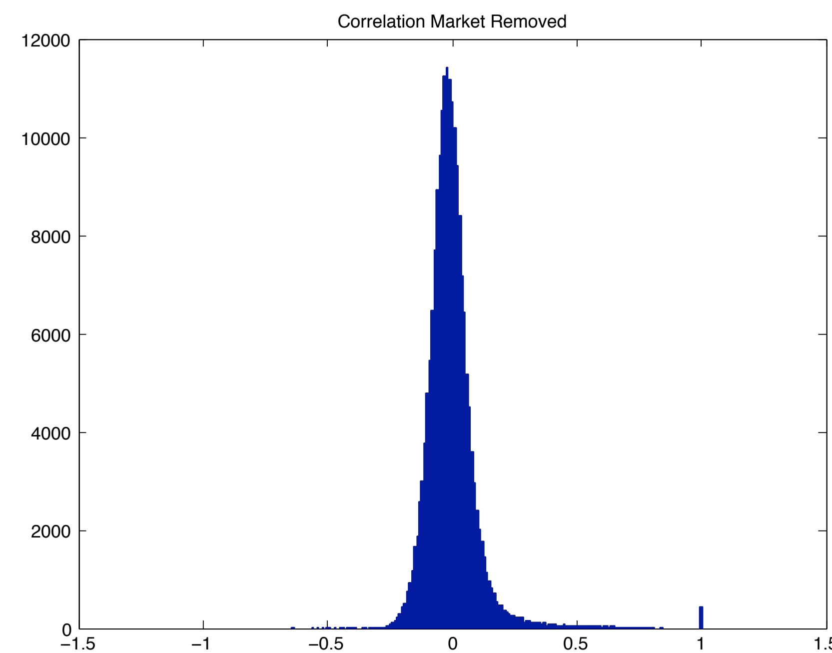
$$d(\ln(S_t)) = \sigma dB_t + c dt + \tau dF_t$$


Bonds
Currency
MBS
Equities
AMEX
Commodities
NYSE
NASDAQ

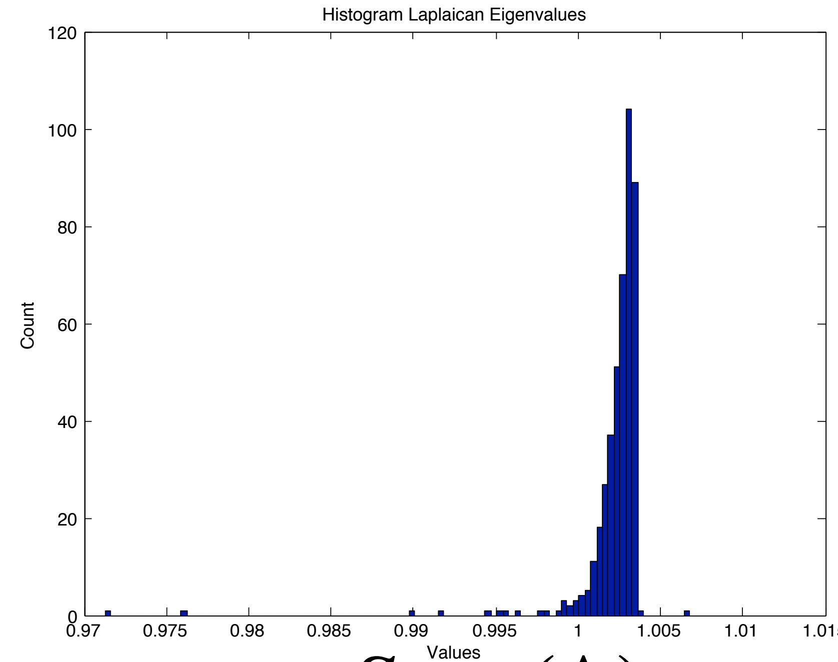

Correaltions Over Whole Market
Correaltion

NASDAQ

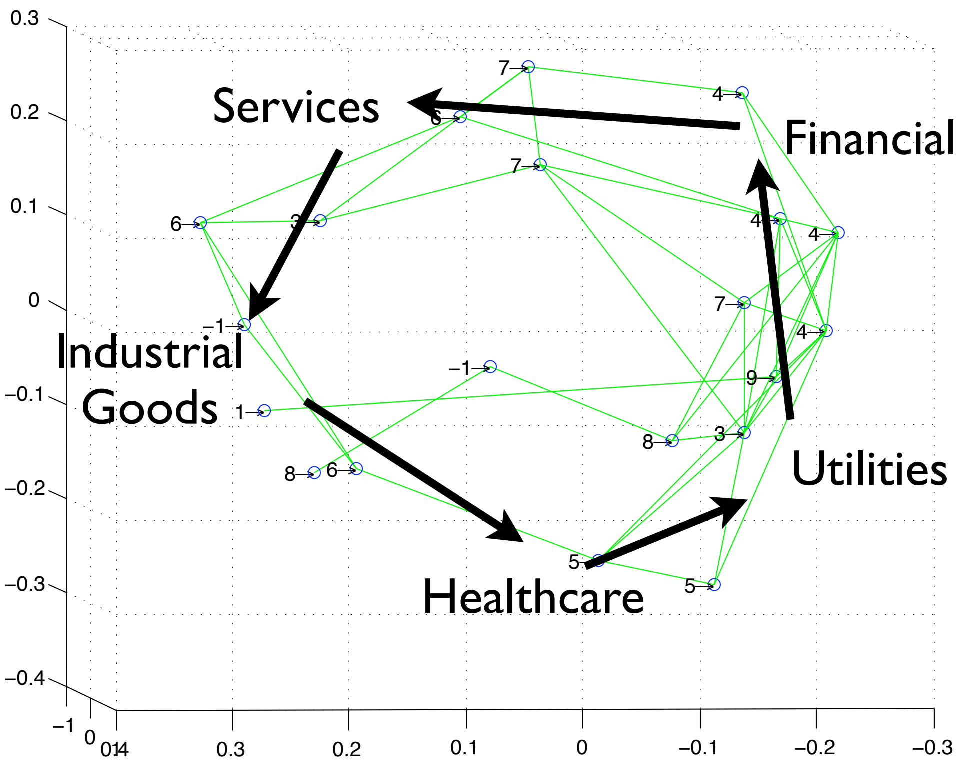

Isolate
AMEX
NYSE
NASDAQ

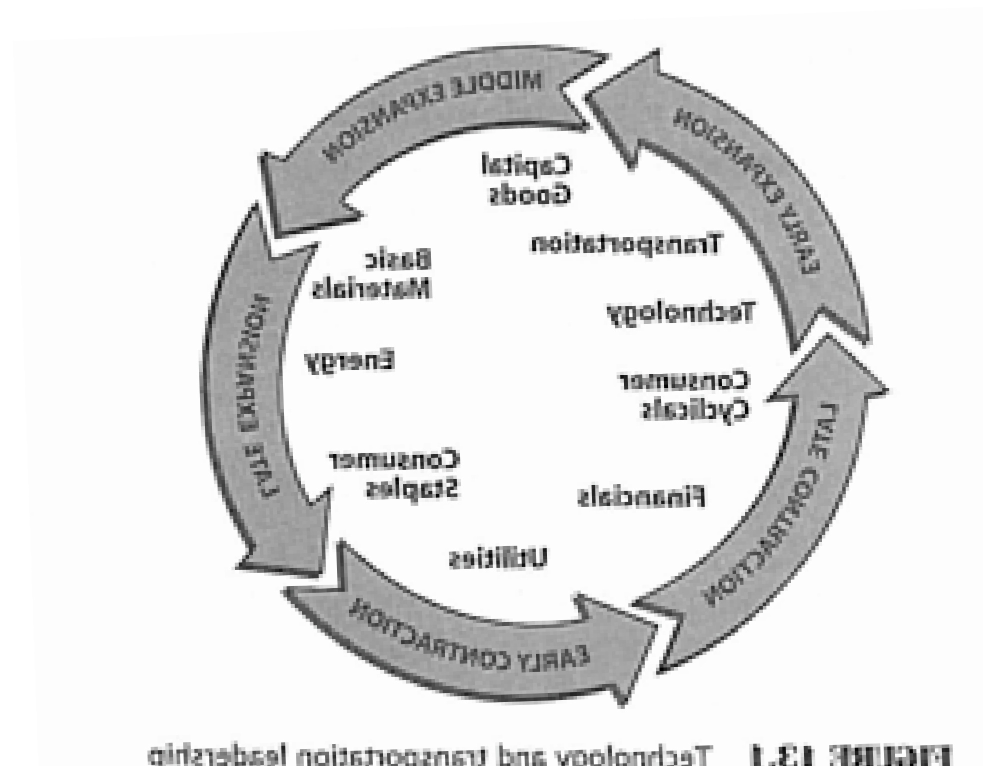

Correlation Market Removed

One Remaining goal is
to understand
the real nature of the
Laplace Spectrum.
One way is by producing
a generative model.

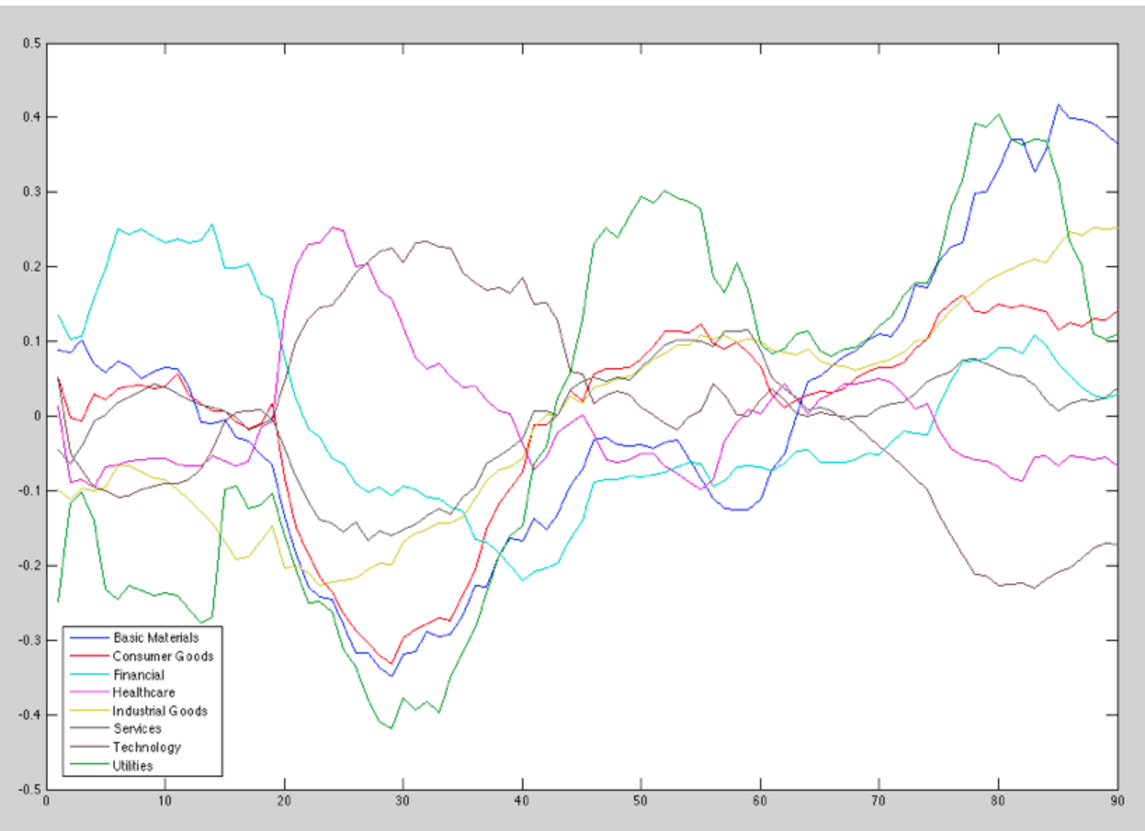


$$Spec(\Delta)$$

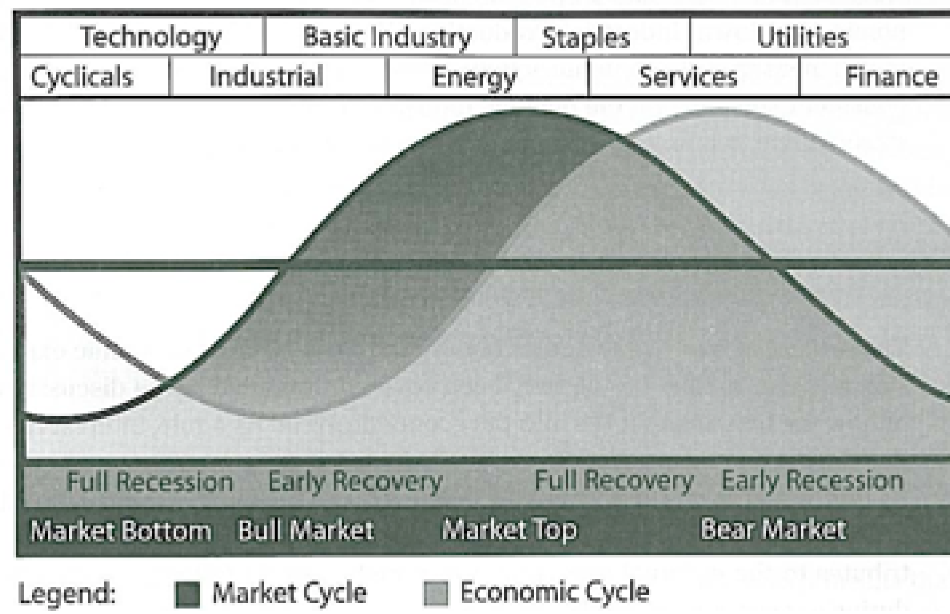




More importantly, generative models will allow us to deal with structured time series.

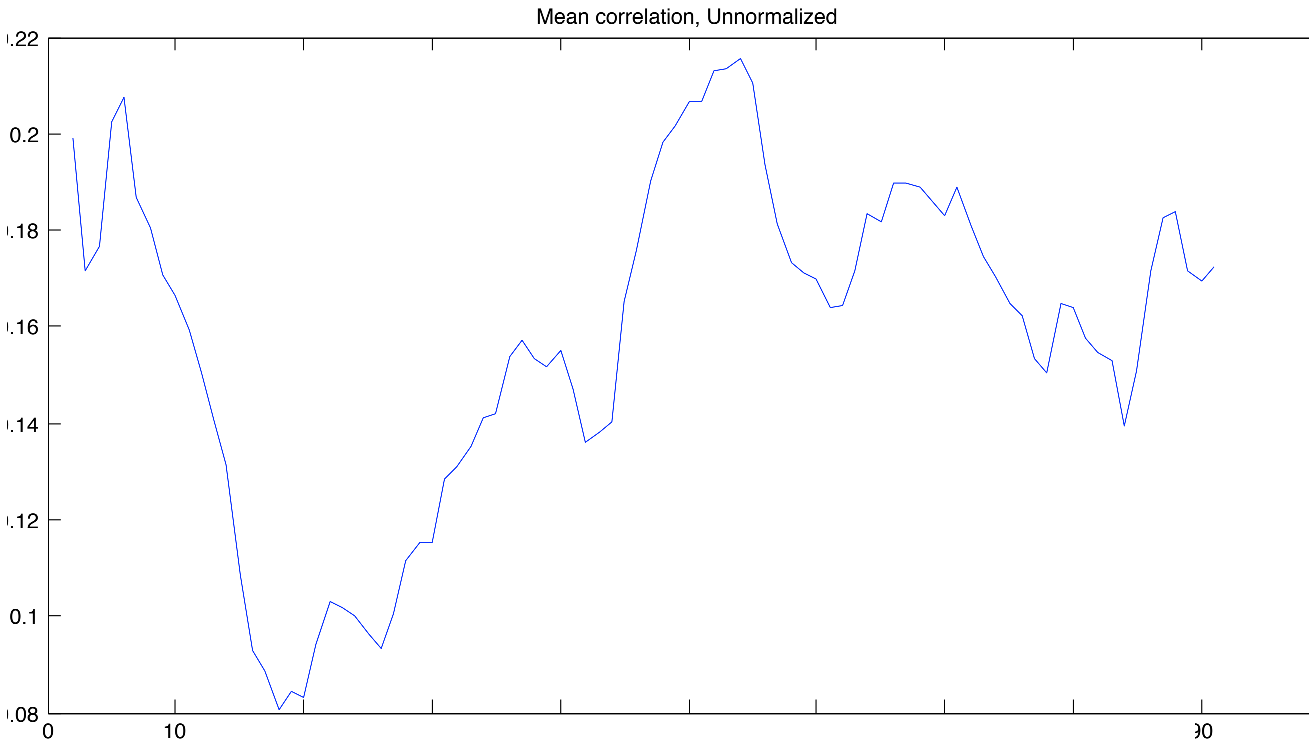


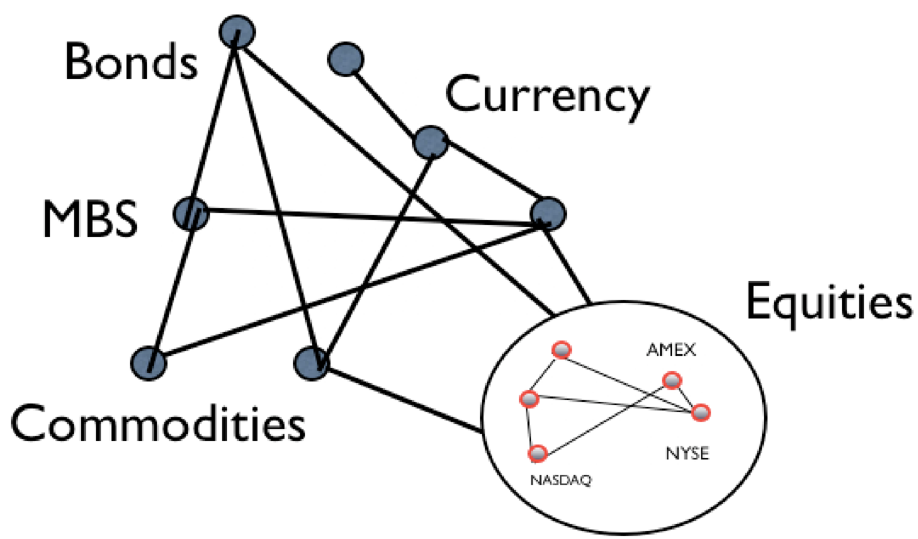
Correlation by Sector

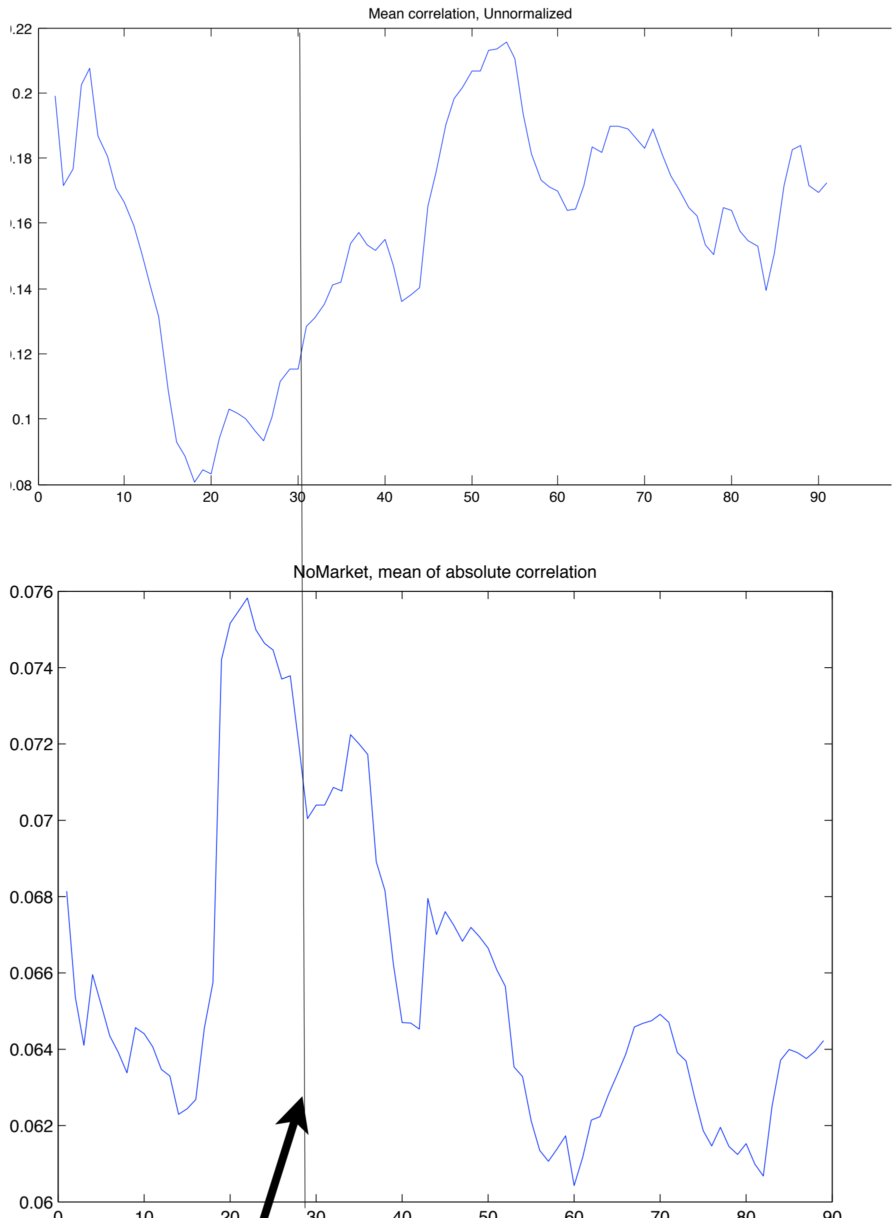


Market Wisdom

Dynamic Properties: Market Pressure







Full correlation pressure

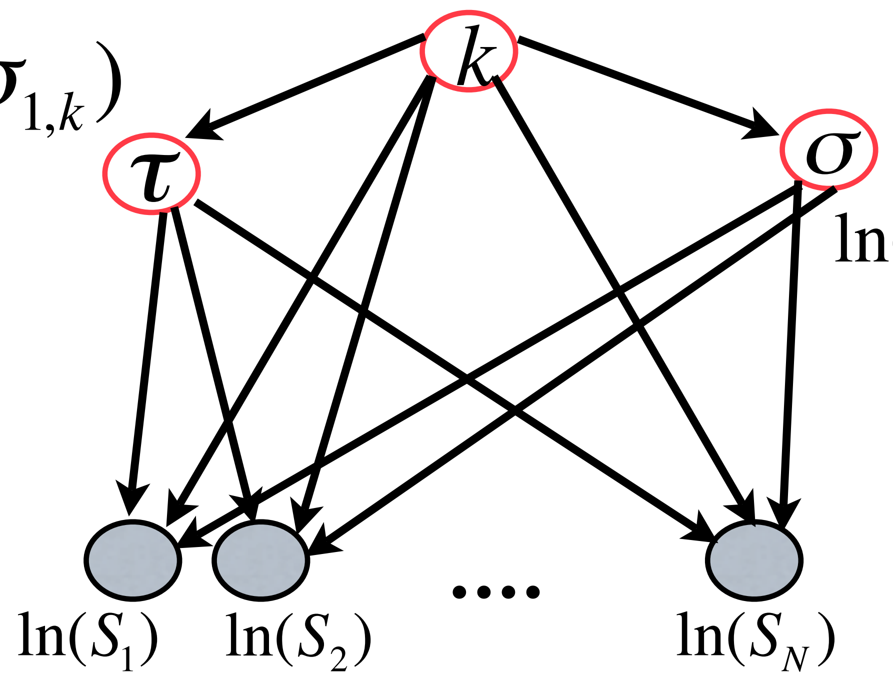
With the equities market isolated

NASDAQ Tech Stock Crash

Here is a very simple generative model of our market.
We imagine everything has some unique "sector" that it is associated to. (In the real market the sector notion is very fuzzy.)

$$\Pr(k) = p_k$$

$$\ln(\tau) = N(m_{1,k}, \sigma_{1,k})$$

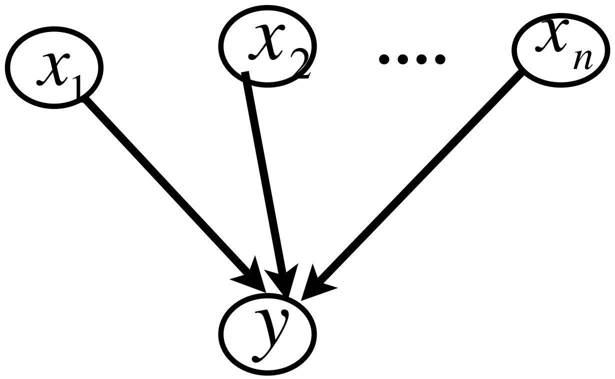


$$\ln(\sigma) = N(m_{2,k}, \sigma_{2,k})$$

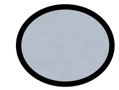
$$\ln(S_t) = \sigma N(0,1) + \tau \mu_{t,k}$$

Variables: k, τ, σ, S_t Parameters: $\mu_{t,k}, m_i, \sigma_i, p_k$

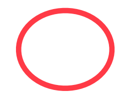
Bayesian Belief Network



$$p(y \mid \vec{x}, \vec{\alpha})$$



Observed



Latent

This is one way to build a complicated distribution from a graph. For example, in our simple example:

$$P\left(\vec{\vec{S}}, \vec{\tau}, \vec{\sigma} \mid m_{ik}, \sigma_{ik}, \mu_{tk}\right) = \prod_{j=1:M} \left(\left(\left(\frac{p_k e^{-\frac{(\ln(\tau)-m_{1k})^2}{2\sigma_{1k}^2}} e^{-\frac{(\ln(\sigma)-m_{2k})^2}{2\sigma_{2k}^2}}}{\sigma_{1k}\sqrt{2\pi}\,\sigma_{2k}\sqrt{2\pi}} \right) d(\ln(\tau))\, d(\ln(\sigma)) \prod_{t=1:N} \left(\frac{e^{-\frac{(\ln(S_t)-\tau\mu_{.t,k})^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}} \right) d(\ln(S_t)) \right) \right|_{\vec{S}_j, \tau_j, \sigma_j}$$

If it allows you easily simulate the observables, then we call it generative.

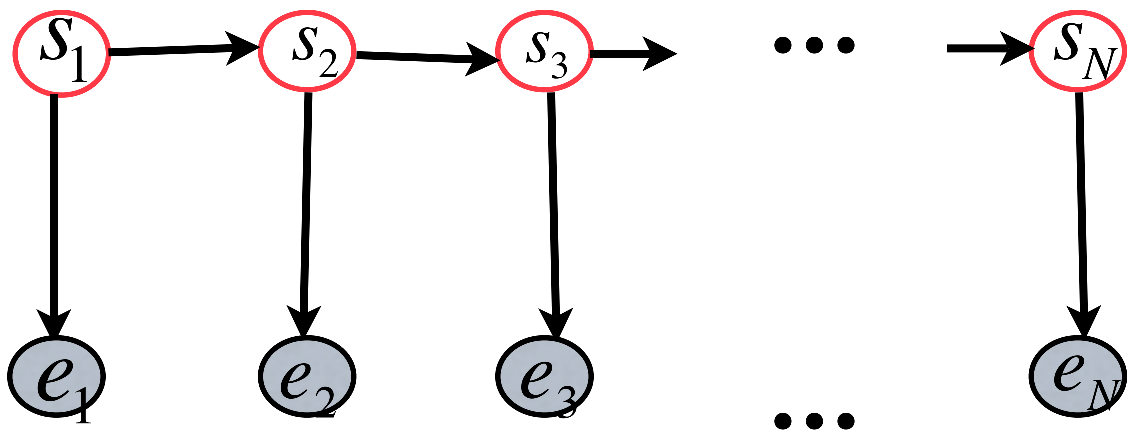
Things we must learn to do:

- Training: We need to find a good choice of parameters.
- Inference: Fore example, given a new stock how well can we predict its sector from its price series?

Simple Example: Hidden Markov Model, HMM

$$\pi_{s_1} = P(s_1)$$

$$T_{s_{t-1}s_t} = P(s_t \mid s_{t-1})$$



$$E_{s_t, e_t} = P(e_t \mid s_t)$$

Simple Training

Annotated Data: meaning a set of M list of emissions AND states.

Choose parameters that maximize:

$$\prod_{i=1:M}\left(\pi_{h_t^i}\prod_t E_{h_t^i,o_t^i}T_{h_{t-1}^i,h_t^i}\right)$$

In this case, the answer is simply to count!

MatLab Example

First we can fake some data...

```
trans = [0.95,0.05;
          0.10,0.90];
 emis = [   1/6  1/6   1/6   1/6   1/6 1/6;
           1/10 1/10 1/10 1/10 1/10 1/2];
[seq,states] = hmmgenerate(20,trans,emis);
[seq; states]
```

ans =

5	1	6	2	4	3	2	1	1	6	5	6	6	5	2	6	1	6	3	6
1	1	2	2	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2

[TRANS,EMIS] = hmmestimate(seq,states)

TRANS =

0.8462	0.1538
0.1667	0.8333

EMIS =

0.2308	0.1538	0.0769	0.0769	0.2308	0.2308
0.1429	0.1429	0.1429	0	0	0.5714

Given a model, find likely states given the emissions.

- Notice, there are exponentially many states with the same emissions (In our last example, every state list was possible).
- Most popular algorithm: Viterbi algorithm.
- In MatLab:

```
STATES = hmmviterbi(seq,TRANS,EMIS)
ans =
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2
```

For simplicity, we'll describe one of many closely related algorithms: the "forward-max" algorithm....

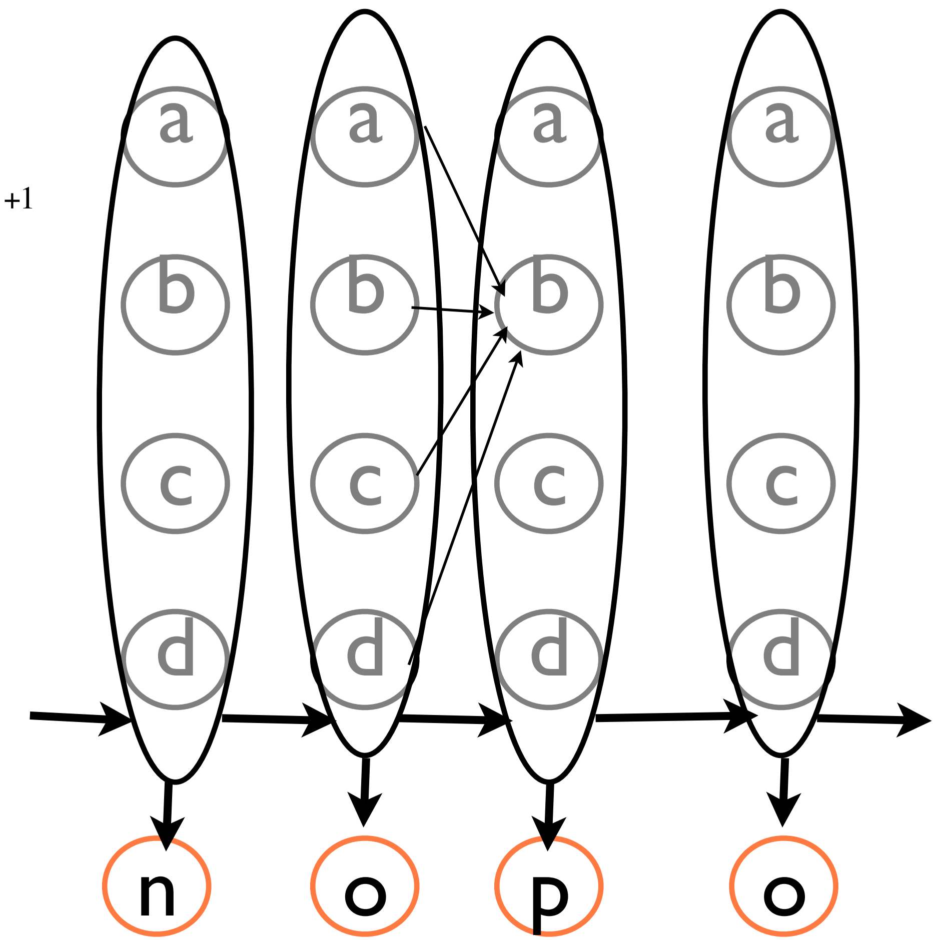
The "Forward-Max" Algorithm

$$\alpha_t(h_t) = P(e_1 = o_1, \ldots, e_t = o_t, s_t = h_t \mid T, E, \pi)$$

$$\alpha_{t+1}(h_{t+1}) = \sum_{h_t} \alpha_t(h_t) T_{h_t h+1} E_{h_{t+1} o_{t+1}}$$

$$f_t = \arg\max_h(\alpha_t(h))$$

Notice: $P(\vec{e} = \vec{o}) = \sum_{s_T} \alpha_T(s_T)$



Training without states

1. Start with guess at the parameters (T_0, E_0, π_0)

2. Using these parameters find fake sequences using "max forward"

3. Use the trivial training to up date values (T_k, E_k, π_k)

4. Repeat until this is small: $\max\left(\left|T_k - T_{k-1}\right|, \left|E_k - E_{k-1}\right|\right)$

Using expected value rather than the "fake value" this is an example of Expectation Maximization and, in the HMM case, called the Baum-Welch algorithm.

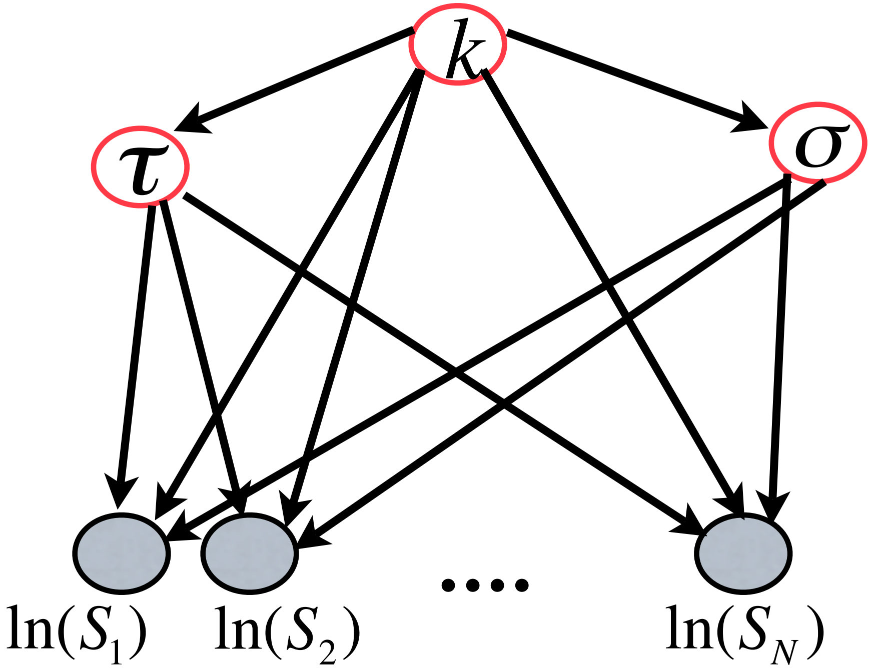
MatLab

[ESTTR,ESTEMIT] = hmmtrain(seq,TRGUESS,EMITGUESS)

In general...

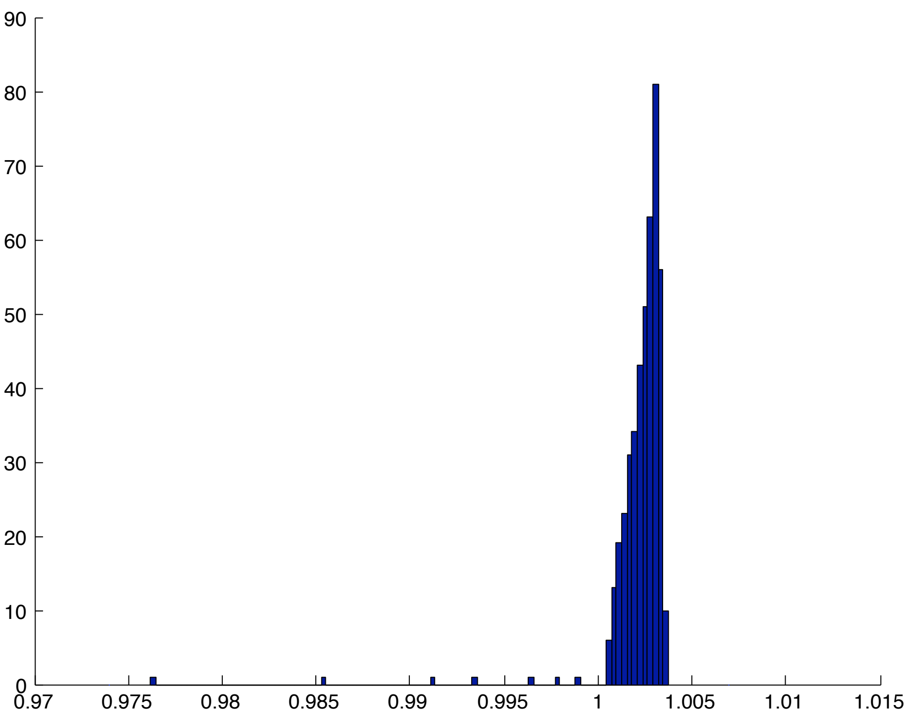
These methods can be extend to generative Bayesian belief networks.

There are many more methods for training and inference (and No Free Lunch!): Variable elimination, Clique Tree Propagation, Recursive Conditioning, Sum Product Algoithm, Stochastic MCMC simulation, Mini-buket elimination, loopy belief propagation, variational methods....

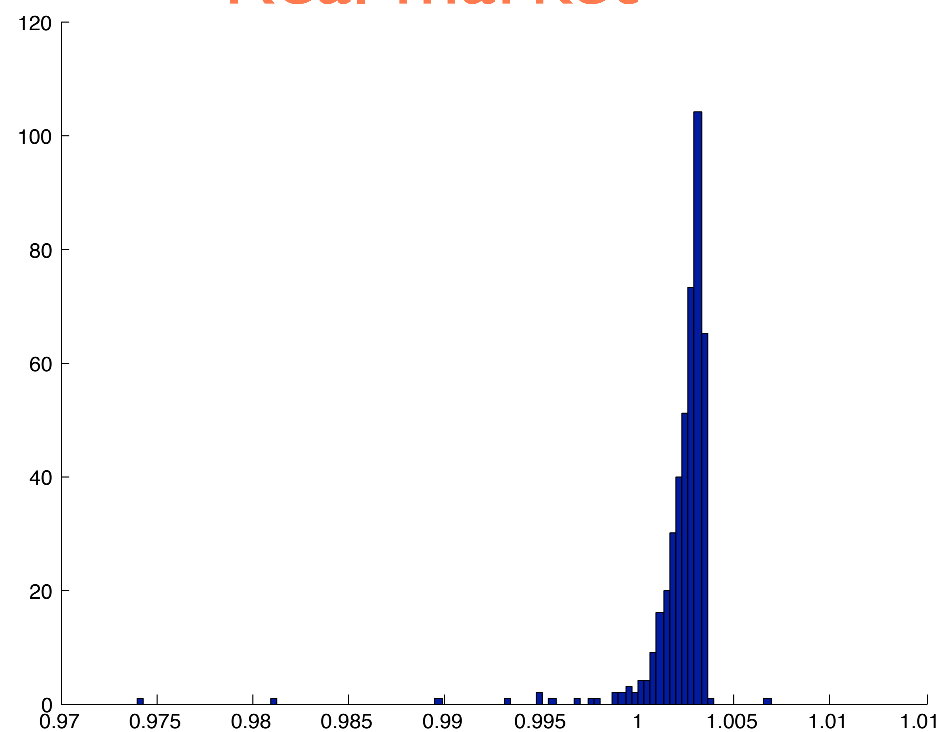


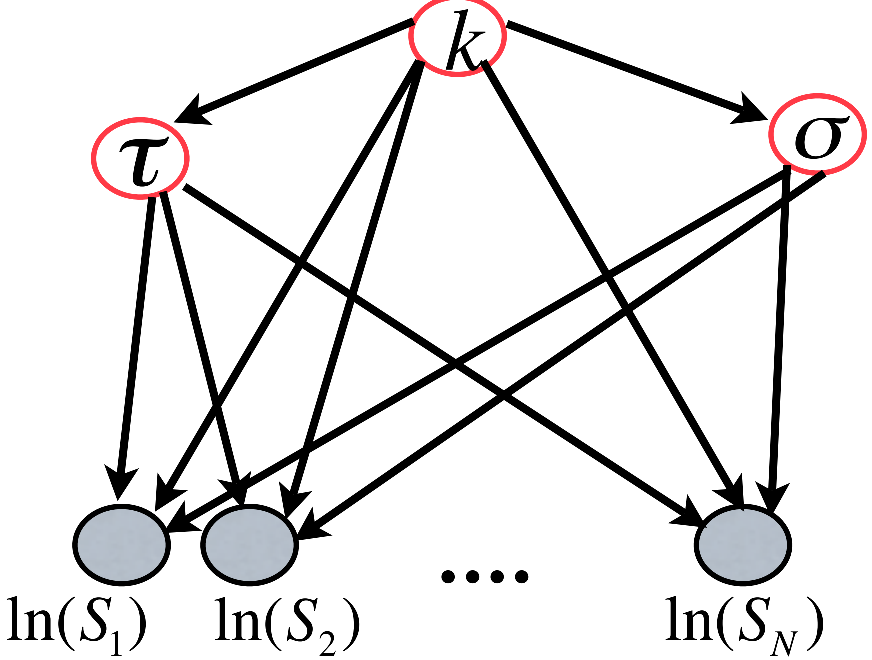
We can use expectation maximization to train our network. This allows us to simulate the market.

Sectors accounted for



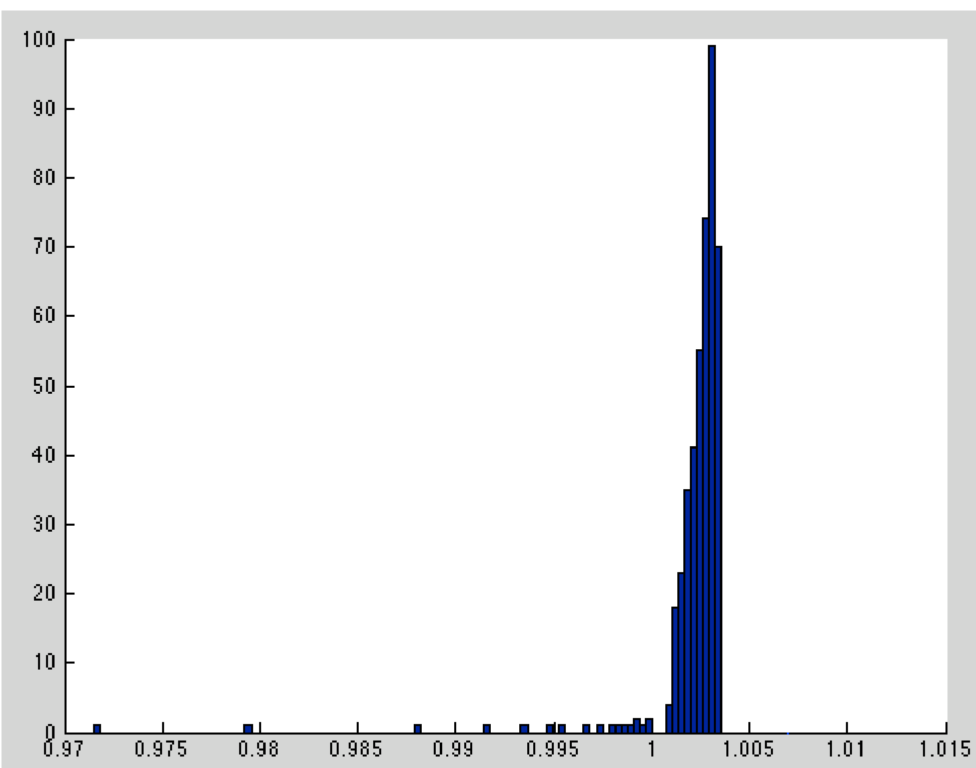
Real market



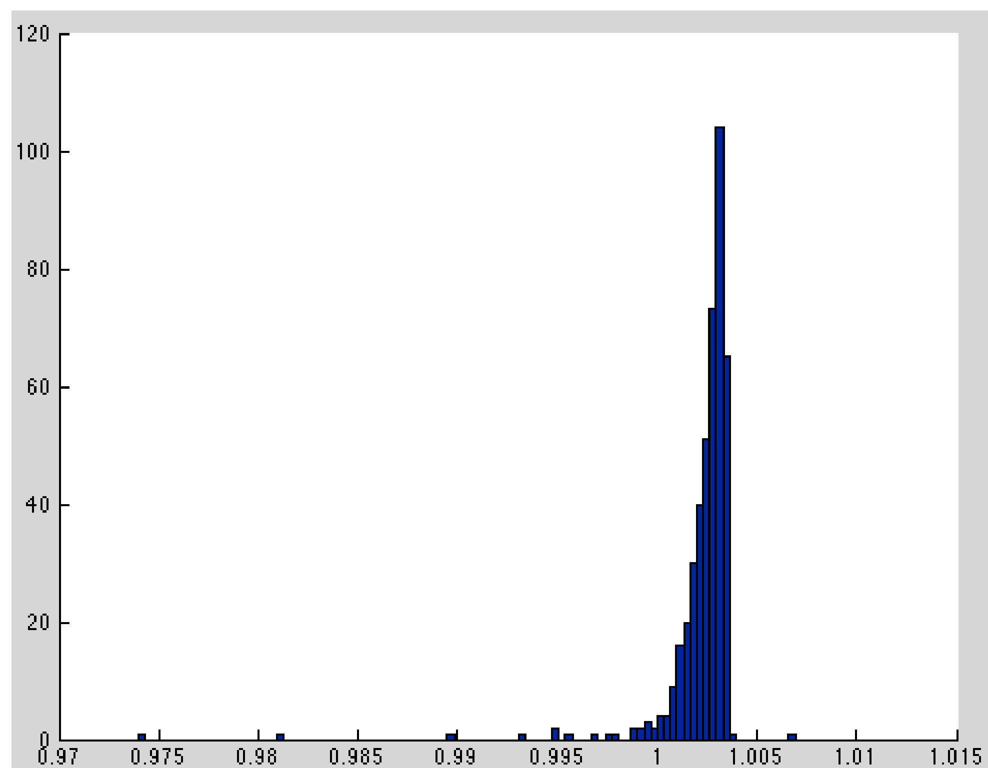


If we account for the 20 spectral clusters, we find:

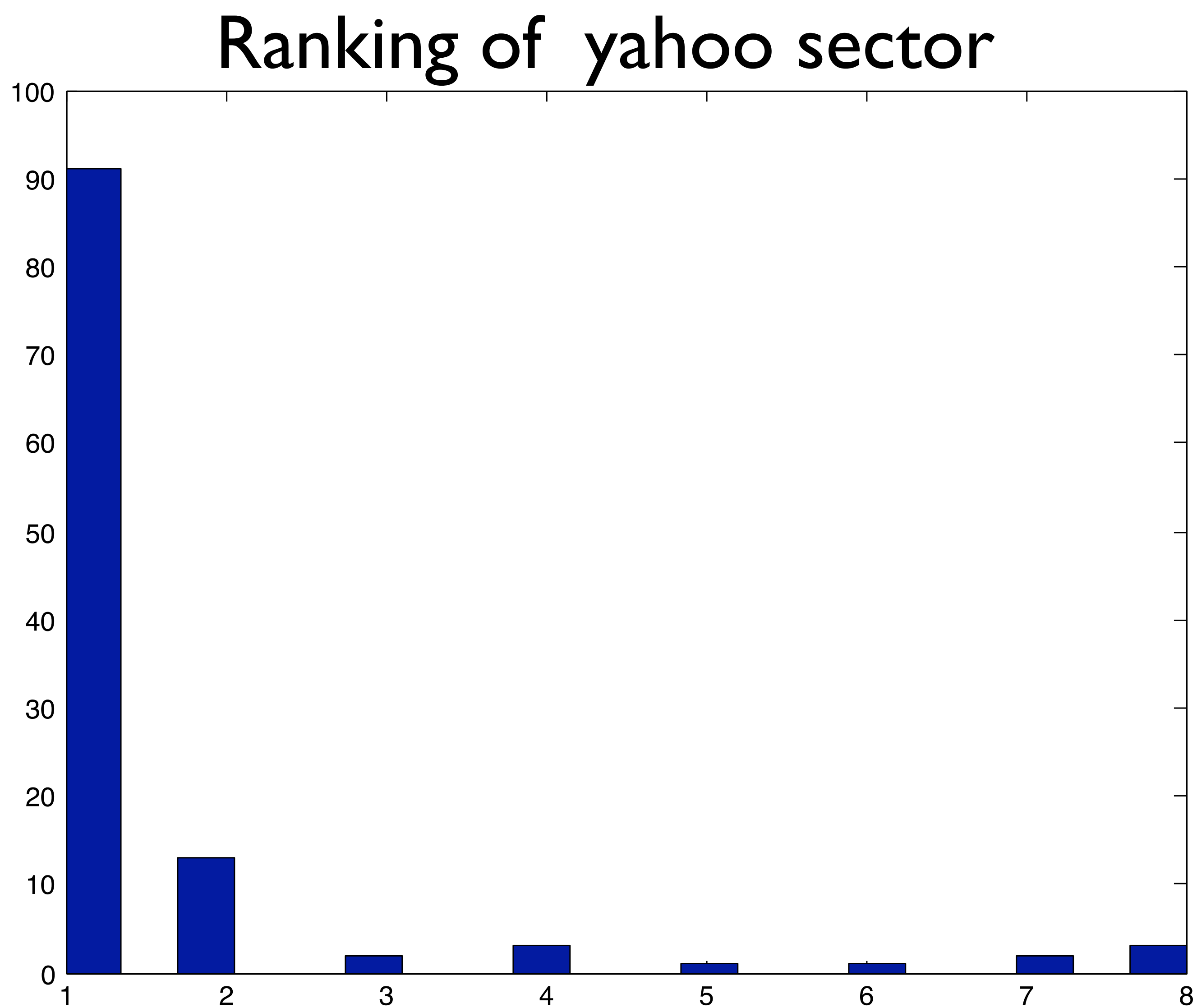
20 Spectral Clusters



Real market

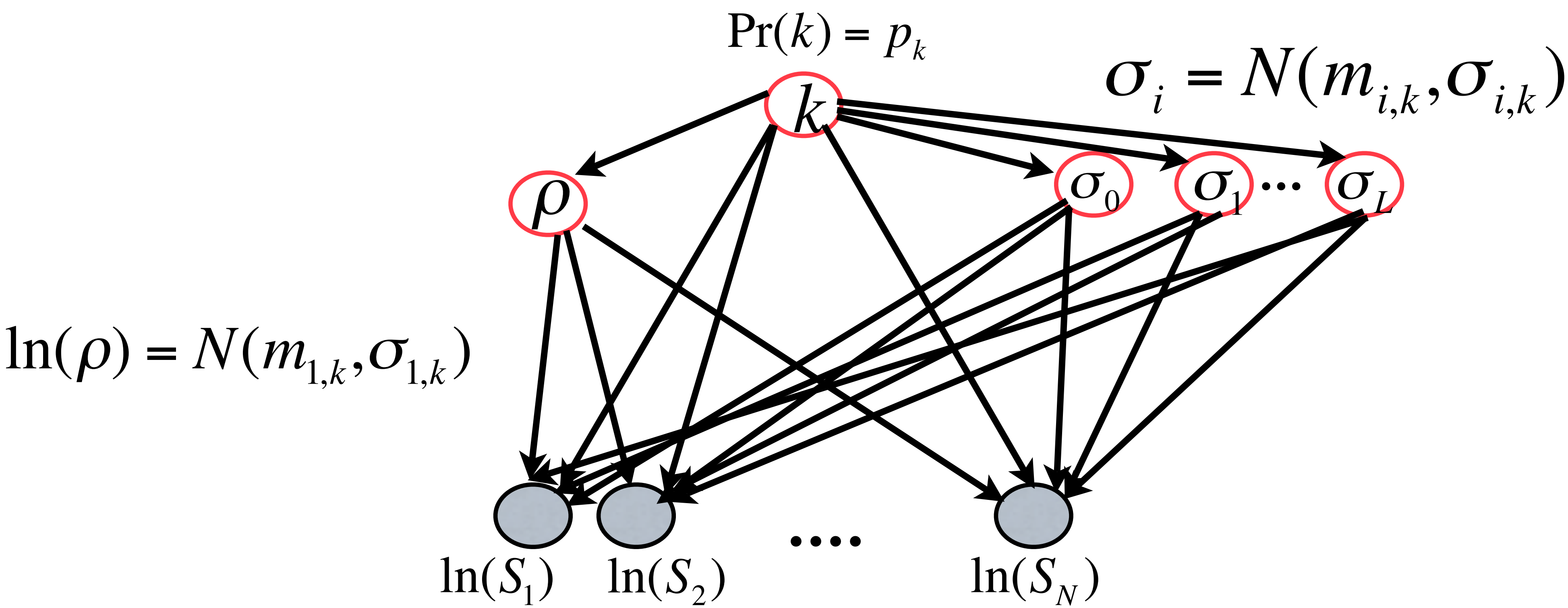


Validation Set (80 % Success)



We trained our model using the training data, and the above is using our validation data. The number is the ranking. For exmaple if a sample gets a 3, then the actual Yahoo! sector was assessed as the third most likely of the 8 possible sectors.

Now we can see if model enhancement improves of error rate.
For example, we can find the optimal L in this family of models:



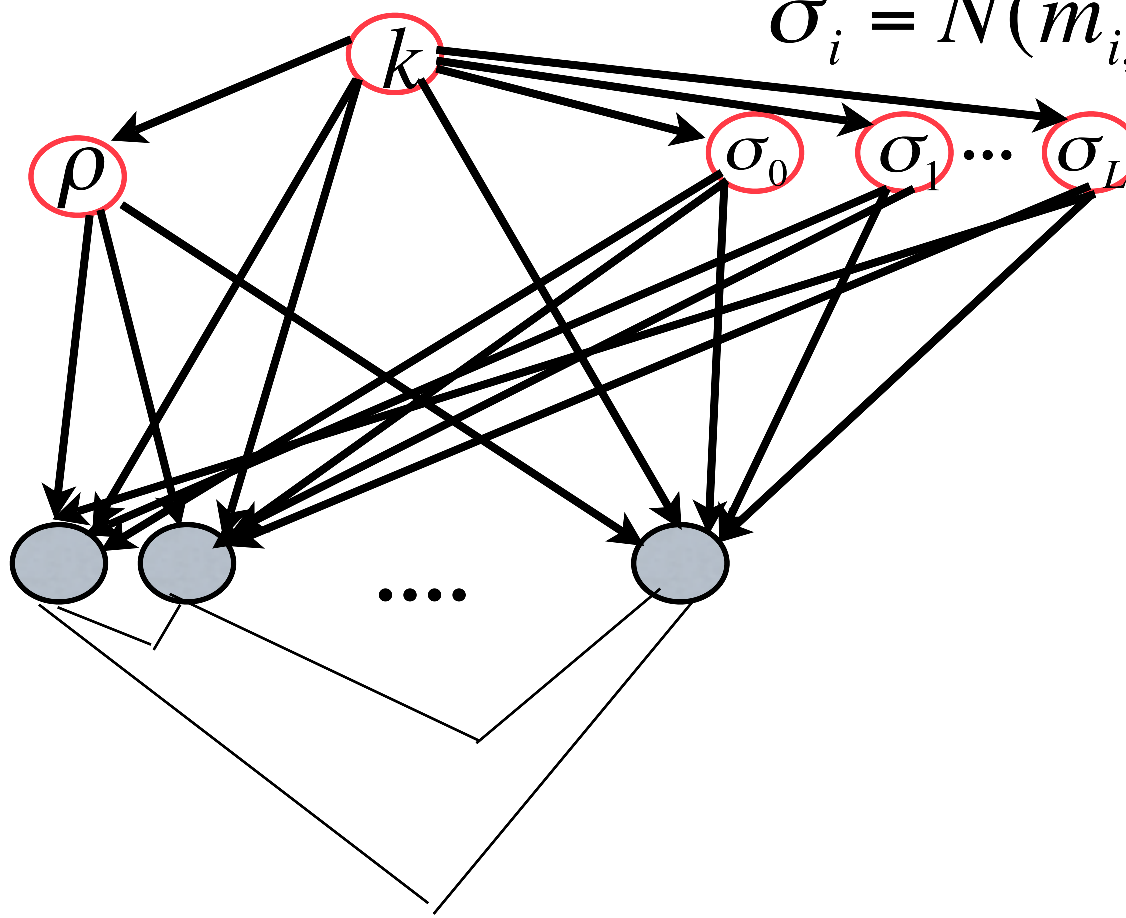
$$\ln(S_t) = e^{\sigma_0 + \sigma_1 t + \ldots + \sigma_L t^L} N(0,1) + \rho\mu_{t,k}$$

In the real world, the times series does NOT consist of independent terms, and we have;

$$\Pr(k) = p_k$$

$$\sigma_i = N(m_{i,k}, \sigma_{i,k})$$

$$\ln(\rho) = N(m_{1,k}, \sigma_{1,k})$$



Project: Look up Conditional Random Fields, explain why they allow for complicated dependencies between the prices.

See: An Introduction to Conditional Random Fields for Relational Learning. Charles Sutton and Andrew McCallum. In *Introduction to Statistical Relational Learning*.