# High level analysis of microarray data

## *Lecture 2*

Claudio Altafini

SISSA

http://people.sissa.it/∼altafini








- **Model-free methods**
  1. CLUSTERING ALGORITHMS
     - put together genes with similar expression profiles
  2. PRINCIPAL COMPONENT ANALYSIS
     - reduce the dimension of a data set keeping the most significant “directions”
  3. ONTOLOGICAL ENRICHMENT
     - add functional annotation (e.g. GO)
     - perform statistical tests on the ontological information

# High level analysis of microarray data



- **Gene network inference methods**
  1. "LESS" MODEL-DEPENDENT METHODS (e.g. probabilistic, statistical, etc.)
     - looking only for the core relationships of a network
     - not quantitative
     - can be used for large scale networks
  2. MODEL-DEPENDENT METHODS (e.g. ODEs)
     - provide both the network topology and the functional relationships
     - useful mostly for small/mid scale networks
     - quantitative

- warning: the classification is not sharp!!!!



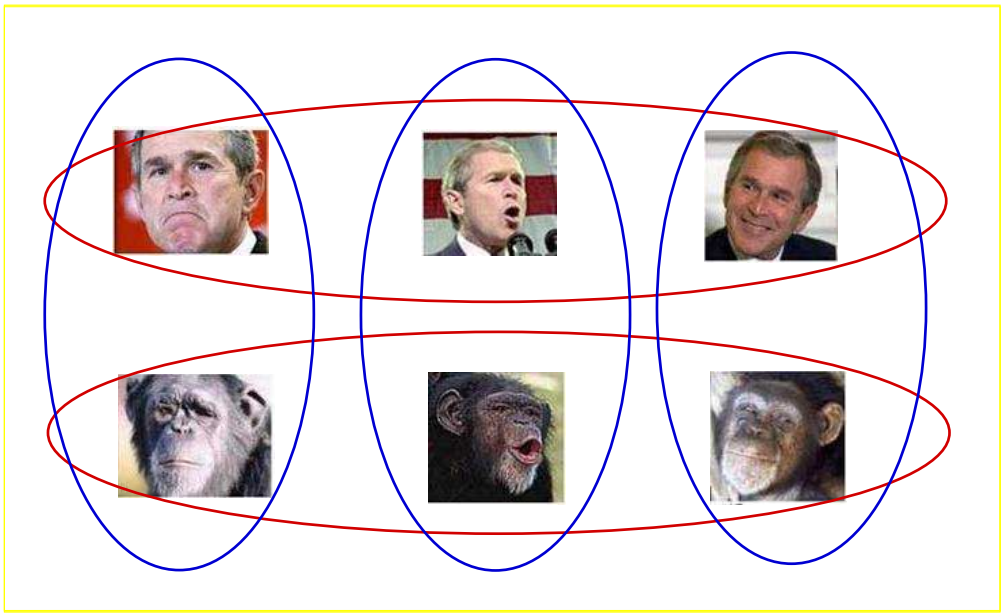


# Clustering

# Clustering



- example: cluster these





# Clustering



- Clustering = dividing a set of data into relatively homogeneous groups according to a user-defined metric

$$d(\mathbf{x},\,\mathbf{y}) > 0 \quad \text{such that} \quad \begin{cases} d(\mathbf{x},\,\mathbf{x}) & = 0 \\ d(\mathbf{x},\,\mathbf{y}) & = d(\mathbf{y},\,\mathbf{x}) \\ d(\mathbf{x},\,\mathbf{y}) & \leqslant d(\mathbf{x},\,\mathbf{z}) + d(\mathbf{z},\,\mathbf{y}) \end{cases}$$

- typically: $L_p$ norm

$$d(\mathbf{x},\,\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_p, \qquad p = 1,\ldots,\infty$$

example: Euclidean norm $p = 2$

$$d(\mathbf{x},\,\mathbf{y}) = \sqrt{(x_1-y_1)^2+\ldots+(x_n-y_n)^2}$$

- 3 main algorithms:
  - $k$-means
  - hierarchical clustering
  - SOM: Self Organizing Maps

# Clustering algorithms: $k$-means

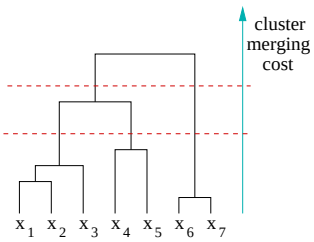


- **Inputs**: data $\mathbf{x}_1, \ldots, \mathbf{x}_n$, # of clusters $k$
- **Output**: $k$ clusters
- **Algorithm**:
  1. select $k$ *centroids*
  2. assign each element $\mathbf{x}_i$ to the cluster with nearest centroid
  3. recompute the centroid
  4. repeat until it converges
- Properties:
  - need to choose $k$
  - initialization step can change the result
  - sensitive to perturbations



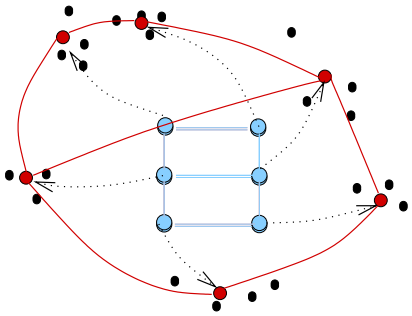
# Hierarchical Clustering






- **Inputs**: data $\mathbf{x}_1, \ldots, \mathbf{x}_n$
- **Output**: clustering tree
- **Algorithm**:
  - put each $\mathbf{x}_i$ in a cluster $C_i = \{\mathbf{x}_i\}$
  - compute the merging cost between each pair of clusters
  - merge the two clusters with cheapest merging cost
  - repeat until only 1 cluster is left
- cost of merging
  - single linkage $\min_{\mathbf{x}\in C_i, \mathbf{y}\in C_j} d(\mathbf{x}, \mathbf{y})$ ($\rightarrow$ loose clusters)
  - average linkage $\frac{1}{|C_i||C_j|}\sum_{\mathbf{x}\in C_i}\sum_{\mathbf{y}\in C_j} d(\mathbf{x}, \mathbf{y})$
  - complete linkage $\max_{\mathbf{x}\in C_i, \mathbf{y}\in C_j} d(\mathbf{x}, \mathbf{y})$ ($\rightarrow$ tight clusters)
- properties:
  - greedy algorithm
  - tends to build big clusters
  - need to choose a threshold on the # of clusters

# Clustering: Self Organizing Maps



- **Inputs**: data $\mathbf{x}_1, \ldots, \mathbf{x}_n$; SOM topology ($k$ nodes)
- **Output** $k$ clusters
- **Algorithm**:
  1. start with a simple topology
  2. select a random data $\mathbf{p}$
  3. move all nodes towards $\mathbf{p}$ according to the rule

$$f_{i+1}(\mathbf{x}) = f_i(\mathbf{x}) + \tau(d(\mathbf{x}, \mathbf{x}_p), i)(\mathbf{p} - f_i(\mathbf{x}))$$

     - $f_i(\mathbf{x}) =$ position of node $\mathbf{x}$ at iteration $i$
     - $\mathbf{x}_p =$ node closest to $\mathbf{p}$
     - $\tau = \tau(d, i)$ learning rate
  4. go to 2. until convergence
- properties
  - even more computationally costy, but more robust
  - neighboring clusters are similar: elements on the border can belong to both clusters





# Clustering: quality indices

- homogeneity

$$\frac{1}{n_{\text{genes}}} \sum_i d(\mathbf{x}_i, C(\mathbf{x}_i))$$

  - = average distance between each $\mathbf{x}$ and the centroid of the cluster it belongs to
  - reflects the compactness of the cluster
- separation

$$\frac{1}{\sum_{i \neq j} n_{C_i} n_{C_j}} \sum_{i \neq j} n_{C_i} n_{C_j} d(\bar{c}_i, \bar{c}_j))$$

  - weighted average distances between cluster centroids
  - reflects the distance between clusters
- siluette width: composition of the two indices

# More advanced clustering



- example: rather than a distance one can use a Pearson correlation

$$\tilde{d}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

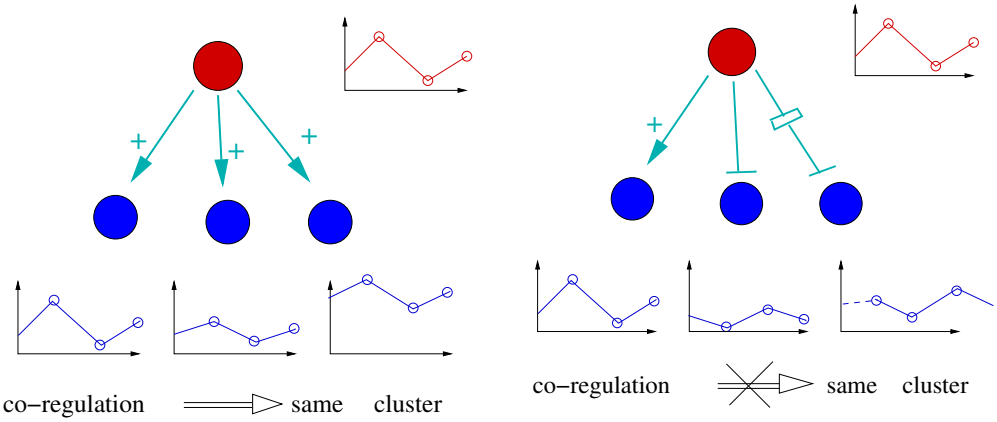
  where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i,\ \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$
- Pearson "metric":
  - uses differences from the mean rather than the mean
  - normalized by the standard deviation $\Longrightarrow \tilde{d}(\mathbf{x}, \mathbf{y}) \in [-1, 1]$
  - invariant to scaling and shifting of $\mathbf{x}$ and $\mathbf{y}$



# Clustering: drawbacks



- clustering:
  similar expression $\Longrightarrow$similar function
  Is it really usefull to inferr common function and co-regulation???
- example:

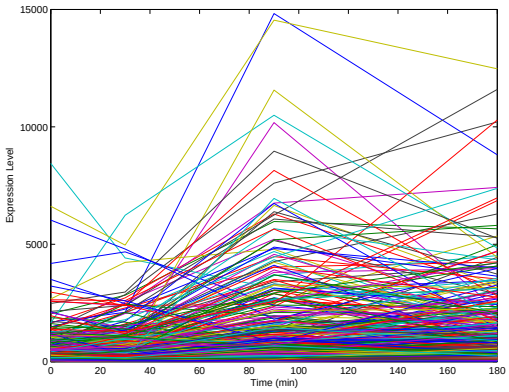


# Clustering: hippocampus time-series



| time instants | 0 min | 30 min | 90 min | 180 min |
|---|---|---|---|---|

- what does the time series looks like for all genes?



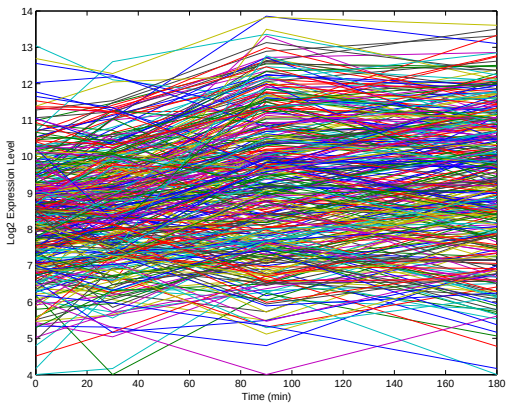

- mess....



# Hippocampus time series



- take the log



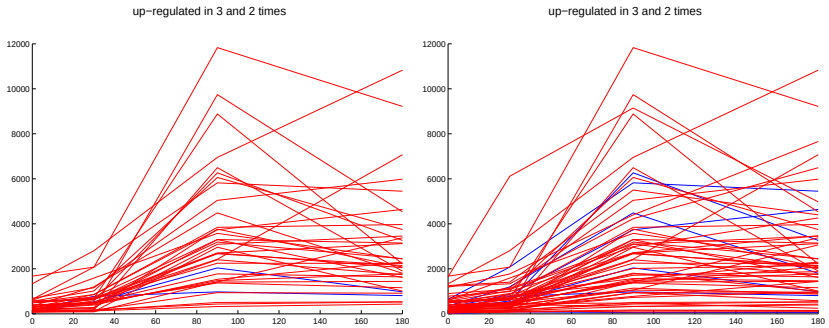

- still mess....

# Hippocampus time series



- to identify interesting genes:
  1. select only genes that show differential expression at a fold change analysis
     - blue = genes that stay up for all 3 time samples
     - red = genes that stay up for 2 out of 3 time samples

4-fold 3-fold



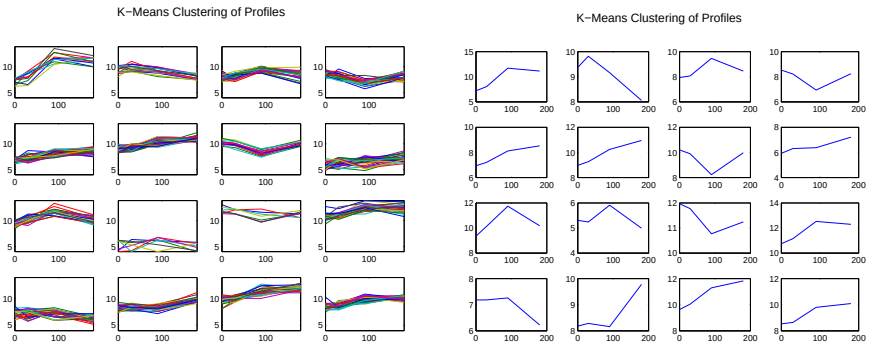

  2. select only genes with sufficiently high variance



# Similar pattern: clustering



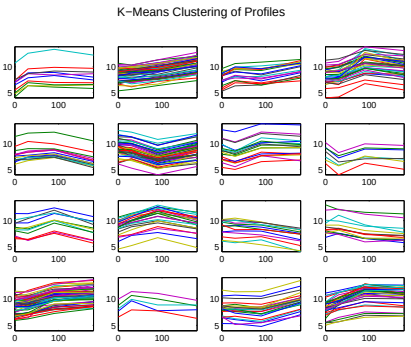
- clustering: similar gene expression time course $\Longrightarrow$similar function (or at least co-regulation)????
- if I filter out those with little variance (the majority) are cluster the remaining

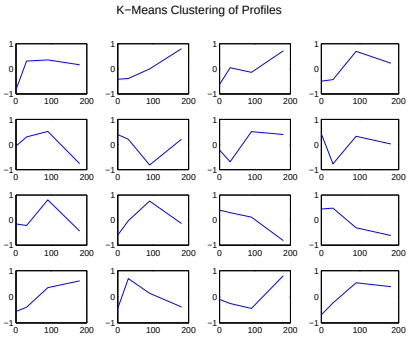


# Similar pattern: clustering



- clustering depends a lot on the algorithm
  - previous page: euclidean distance
  - here: Pearson correlation as distance








# Principal Component Analysis

# PCA: Principal Component Analysis



- PCA detects the directions that capture the most of the information available from the data
- PCA is performed by a linear transformation of the data set based on the Singular Value Decomposition (SVD)
  - idea of principal components analysis: take linear combinations of the $\mathbf{x}$ as "basis" elements so that the new basis elements are orthogonal $\Longrightarrow$they contain no redundant information
  - Successive principal components capture less and less information about the data
  - We can truncate the representation of the data to a limited number of principle components $\Longrightarrow$dimensionality reduction
- use SVD to decompose $X$ ($n \times m$ matrix):

$$X = U\Lambda V^T \qquad \begin{array}{ll} U & n \times m \text{ orthogonal} \quad UU^T = I_n \\ V & m \times m \text{ orthogonal} \quad VV^T = I_m \end{array}$$



# PCA: Principal Component Analysis



$$X = \underbrace{\begin{bmatrix} x_1^1 & \dots & x_1^m \\ x_2^1 & \dots & x_2^m \\ \vdots & & \vdots \\ x_n^1 & \dots & x_n^m \end{bmatrix}}_{1^{st}\text{exp.}\dots m^{th}\text{exp.}} = U\Lambda V^T \qquad \Lambda = \begin{bmatrix} 0 & & & & \\ & \ddots & & & \\ & & 0 & & \\ & & & \lambda_1 & \\ & & & & \ddots \\ & & & & & \lambda_\ell \end{bmatrix}$$

- $\ell = \text{rank}(X)$
- $\lambda_1, \dots, \lambda_\ell$ singular values: $\lambda_j = \sqrt{\mu_j}$
- $\mu_j = \text{eig}(\Sigma)$ = eigenvalues of the covariance matrix of $X$
- $\Sigma$ = covariance matrix of $X$ after centering: $\Sigma = (X - [\bar{x}^1 \ \dots \bar{x}^m])^T (X - [\bar{x}^1 \ \dots \bar{x}^m])$
- is PCA improving yor clustering algorithm? Not necessarily.... see

  K. Y. Yeung, W. L. Ruzzo *Principal Component Analysis for clustering gene expression data*, Bioinformatics 17 pages 763-774, 2001

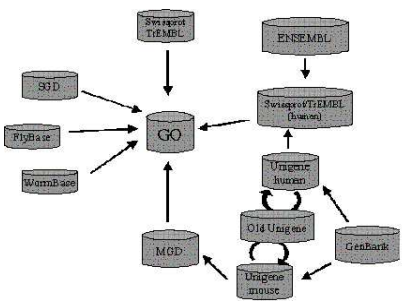
# Ontological enrichment





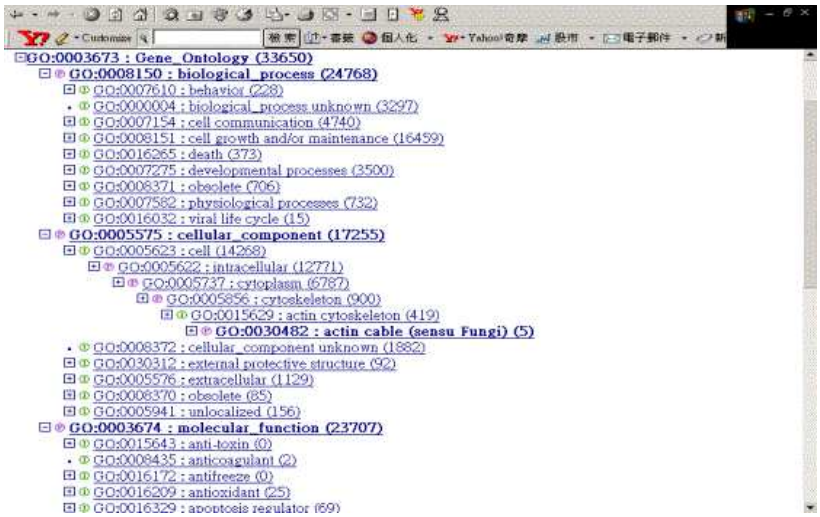
# Gene Ontology

- GO = Gene Ontology project provides a controlled vocabulary to describe gene and gene product attributes in any organism
- Genes are associated, with GO terms by trained curators
- GO annotations give “functions” label to genes
- cross-link to most common gene banks, pathways database, etc.




- http://www.geneontology.org

# Structure of GO




- GO terms:

1. Biological Process
2. Molecular Function
3. Cellular Component

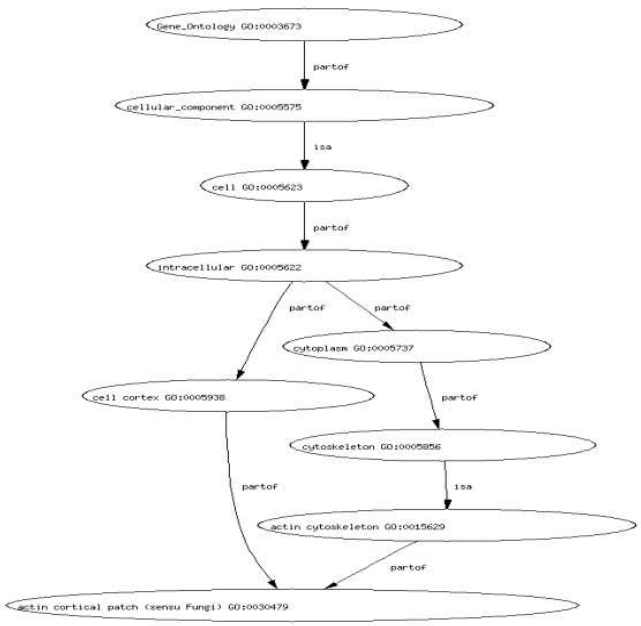
- a gene may belong to many categories



Structure of GO



- Ontologies are structured as a hierarchical directed acyclic graph (DAG)
- Terms can have more than one parent, and zero, one or more children



# Ontological enrichment



- Ontological enrichment: questions you would like to ask:
  - what is the “main” functional annotation of interesting genes (e.g. differentially expressed, or genes having a similar expression profiles)?
  - do genes involved in the same process/function have a similar profile of expression?
- Many tools exist that use GO to answer these questions: `http://www.geneontology.org/GO.tools.microarray.shtml`
- Most of these tools work in a similar way:
  - input a gene list and a subset of “interesting” genes
  - tool shows which GO categories have most interesting genes associated with them i.e. which categories are “enriched” for interesting genes
  - tool provides a statistical measure to determine whether enrichment is significant



# Ontological enrichment

1. select a *set of significant genes* (e.g. t-test)
2. attain all the GO categories corresponding to them
3. analyze GO terms for significance

- example of statistical measure: Hypergeometric test
  - $N$ genes on the microarray
  - `Bio` is a GO term $\begin{cases} M \text{ genes } \in \texttt{Bio} \\ N-M \text{ genes } \notin \texttt{Bio} \end{cases}$
  - $K$ = n. of significant genes
  - what is the probability of having exactly $x$ genes from $K$, of type `Bio`?

$$P(X = x \,|\, N,\, M,\, K) = \frac{\binom{M}{x}\binom{N-M}{K-x}}{\binom{N}{K}}$$

  - P-value = probability of having at least $x$ of $K$ genes (cumulative proability distribution)

$$\mathrm{p-val} = 1 - \sum_{i=0}^{x-1} \frac{\binom{M}{x}\binom{N-M}{K-x}}{\binom{N}{K}}$$


High level analysis of microarrays

Clustering

Principal Component Analysis

Ontological enrichment
- Ontological enrichment
- Onto-Express
- KEGG
- biclustering
- Biclustering example
- Cancer compendium
- Interpretation

Inferring Regulatory Networks

Bayesian Networks

# GO Tools



- Tools:

| Tool | Statistical model | Correction for multiple experiments |
|---|---|---|
| Onto-Express | $\chi^2$, binomial, hypergeometric, Fisher's exact test | sidák, Holm, Bonferroni, FDR |
| GoMiner | Fisher's exact test | Relative enrichment |
| EASEonline | Fisher's exact test | Bonferroni |
| GeneMerge | Hypergeometric | Bonferroni |
| FatiGO | Percentage | "Step-down minP, FDR |
| GOstat | $chi^2$„ Fisher's exact test" | FDR, Holm |
| GOToolBox | Hypergeometric, binomial, Fisher's exact test | Bonferroni, Holm, Hochberg, Hommel, FDR |
| GoSurfer | $\chi^2$ | q-value ,DAG |

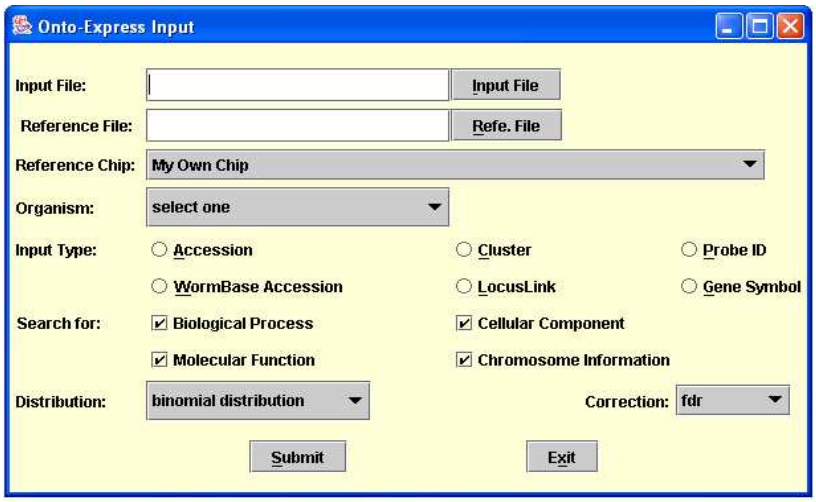
- Affymetrix also provide a Gene Ontology Mining Tool as part of their NetAffx Analysis Center which returns GO terms for probe sets



# Example: Onto-Express

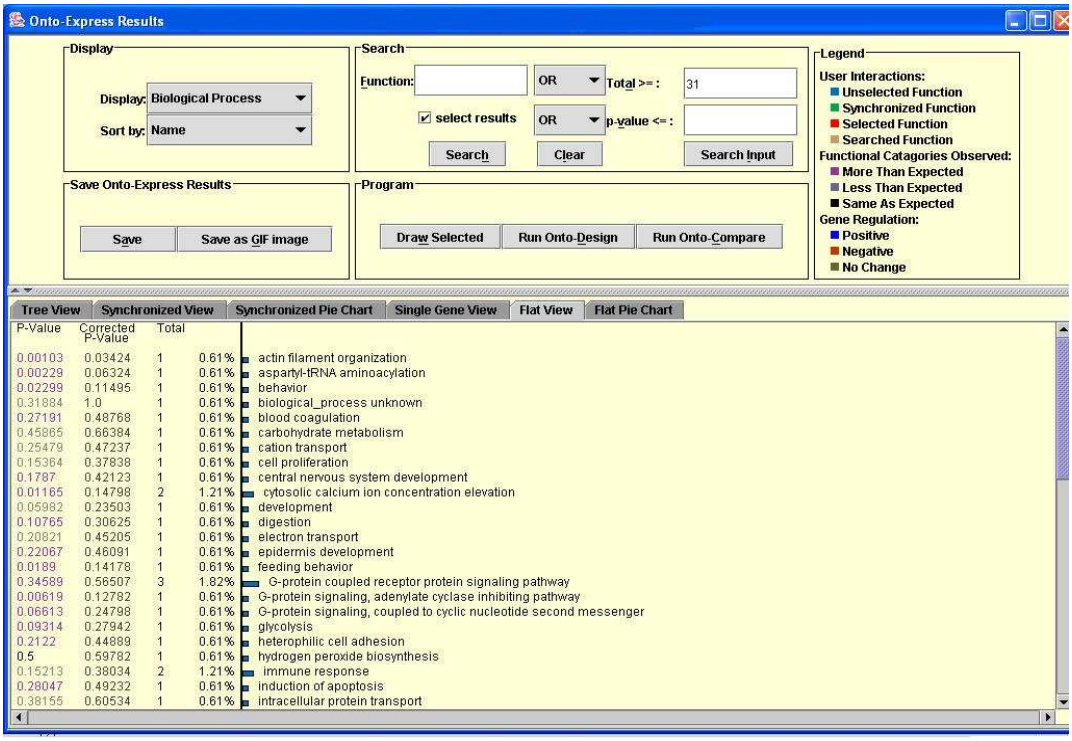


- Onto-Express is available at `http://vortex.cs.wayne.edu/projects.htm`



# Example: Onto-Express

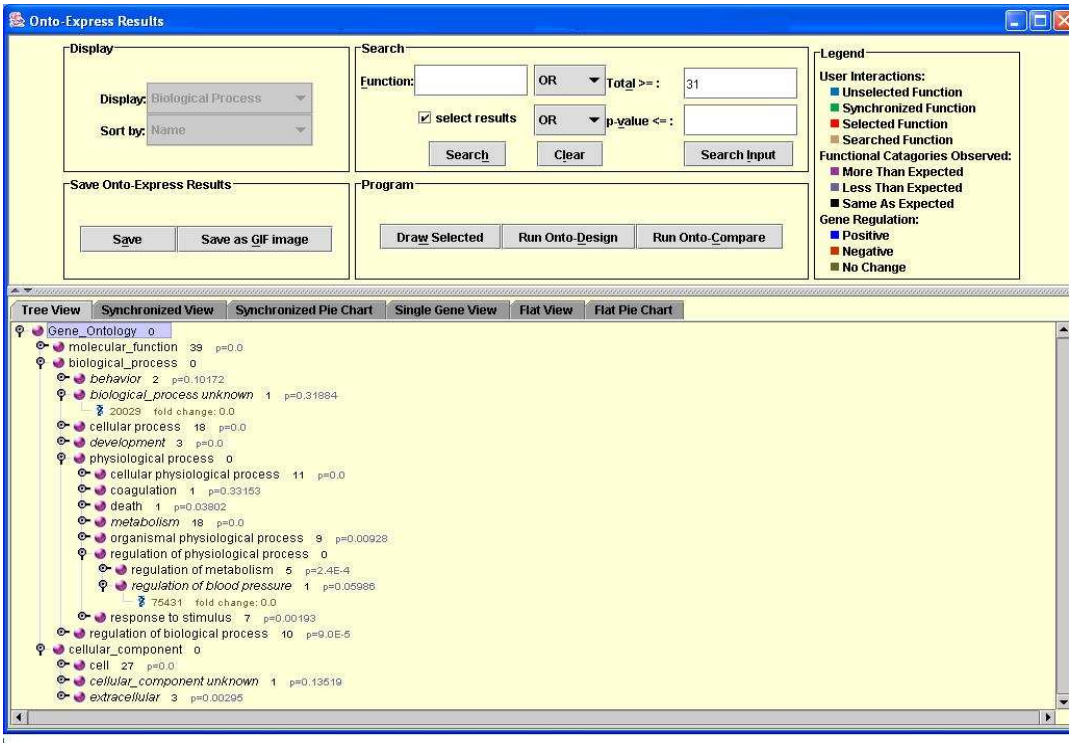






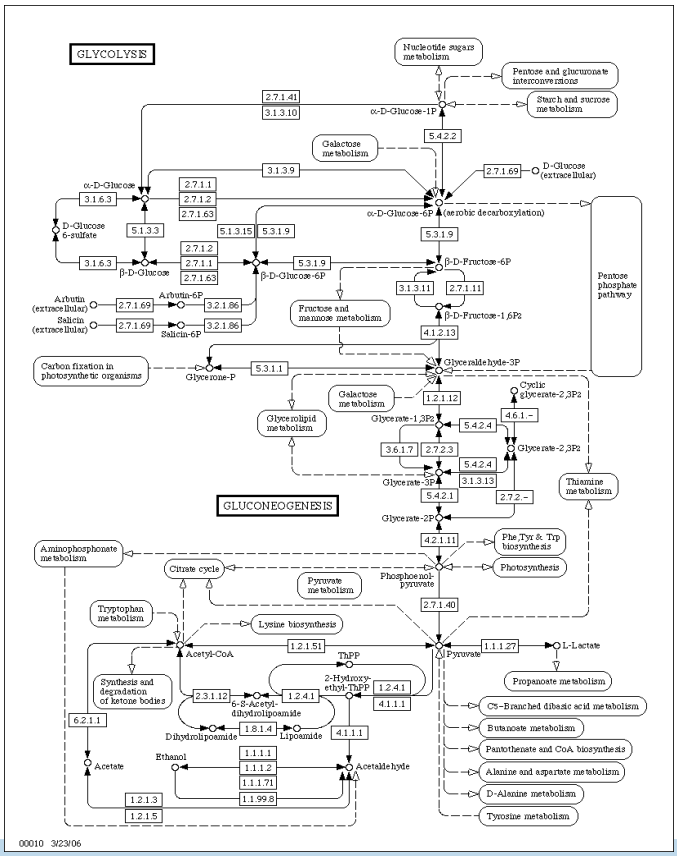

# Example: Onto-Express





# Other Ontologies: KEGG pathways



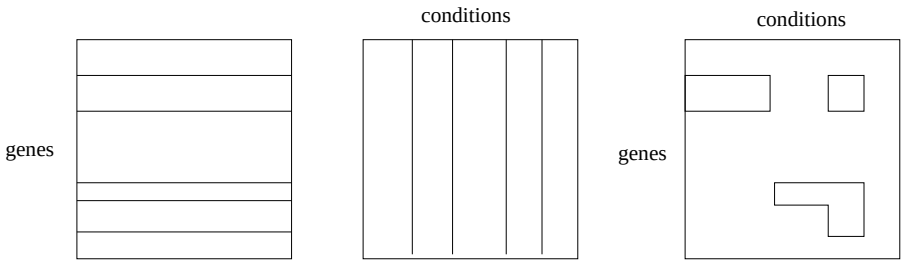





# Biclustering



- clustering can be carried out:
  - w.r.t gene expression
  - with respect to some other condition (e.g. clinical condition in which I take the sample, ontological information)
- two-axis clustering $\Longrightarrow$ biclustering

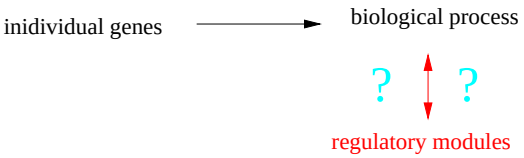


# Biclustering: expression + ontology



E. Segal, N. Friedman, D. Koller, A. Regev *A module map showing conditional activity of expression modules in cancer* Nature Genetics 36, 1090-1098 (2004)

- idea



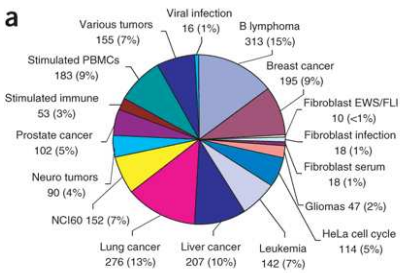

- rather than working with single genes and their regulatory mechanics is it possible to lump together genes into modules = set of genes that act in concert to carry out a specific function?
- here: DNA microarray data in a comprehensive analysis aimed at identifying the shared and unique molecular 'modules' underlying human malignancies.
- in the paper modules are extrated and used to characterize gene-expression profiles in tumors as a combination of activated and deactivated modules.

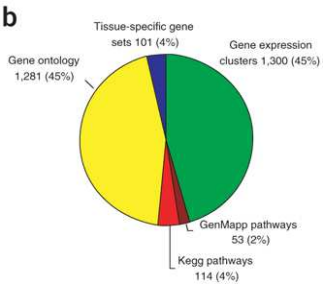


# A cancer compendium



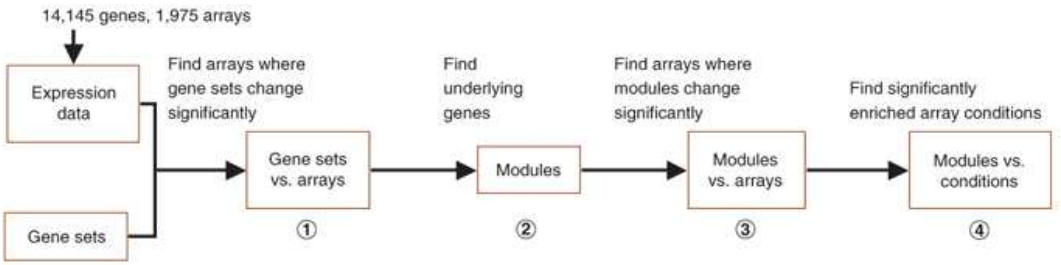
- 26 studies




- expression of 14.145 genes
- 1.975 arrays: Stanford Microarray Database
  Whitehead Institute Database
- 2849 gene sets: Gene Ontology (1281)
  KEGG: Kyoto Encycl. of Genes and Genomes (114
  Gene MicroArray Pathway Profiler (53)
  other: tissue-specific gene sets (101)
  other: clustered sets of coexpressed genes (1300)
- whole analysis: data mining tool called GeneXPress

# Method modules & clinical conditions




- *# of statistically significant modules = 456* (spanning various processes and functions, metabolism, transcription, degradation, cellular and neuronal signalling, growth, cell cycle, apoptosis, extracellular matrix and cytoskeleton components)
- next: *identify clinical conditions* according to the combination of active/deactive modules → *263 biological and clinical conditions* (tissue type, tumor type, diagnosis and prognosis info, molecular markers)





# Modules vs clinical conditions

# Interpretation of the network

- interpretation
  1. clinical conditions $\rightarrow$ modules
     - some modules (e.g. cell cycle) are common to many tumor types $\rightarrow$ tumorigenic processes?
     - some other are specifi c (e.g. neural processes repressed in a set of brain tumors)
  2. modules $\rightarrow$ clinical conditions
     - various tumors of hematologic nature involve similar immune, inflammation, growth regulation and signalling modules
- Conclusion:
  - large scale analysis between different tissues/conditions/experimental setting yelds results with statistical signifi cance $> 0$
- in studying tumors:
  - Activation of some modules is specifi c to particular types of tumor
  - Other modules are shared across a different clinical conditions, suggestive of common tumor progression mechanisms.

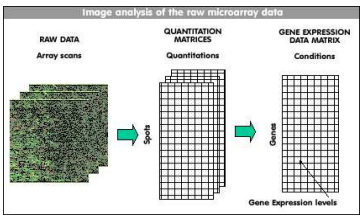


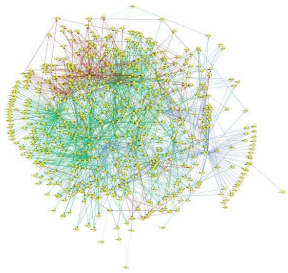


# Inferring Regulatory Networks

# Limitations of clustering/PCA

- clustering: methods of information extraction from data based on co-regulation:
  - similar expression pattern over a set of experiments $\Longrightarrow$similar function
  - all the clustering algorithms give the same results if the time points are randomly permuted
  - cannot reveal causal/dynamical connections
  - $\Longrightarrow$does not reveal what is behind the co-regulation
- more ambitious goal:
  find the transcriptional regulatory network



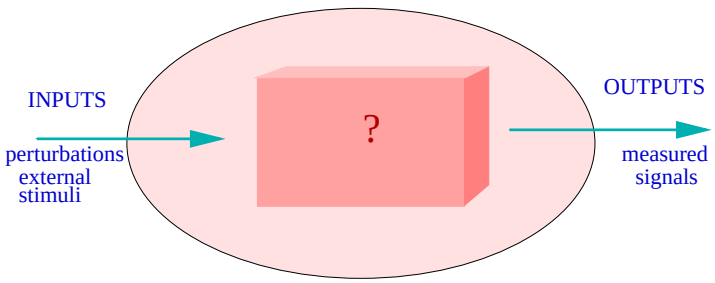







# The “reverse engineering” paradigm

- basic idea: *the architecture of the network is inferred (or reverse engineered) based on the observed response of the system to a series of experimental perturbations*




- measured signals: [mRNA], [proteins] [metabolites]
  - global response: measure the entire “state” vector
    - time series (e.g. cell cycle)
    - single time point (e.g. steady state)
- perturbations: experimental interventions that alter the state of interest

# The “reverse engineering” paradigm



- TASK: from gene expression profiles to a gene-gene graph
  - extract the network structure
  - quantify the interactions
- computationally the task is hard: a very large amount of data is required
  $\rightarrow$ data rich/data poor paradox
  many data $\not\Rightarrow$ significant data for network inference
- what are the significant data?
  Those obtained perturbing sistematically the variables of interest
  $\rightarrow$ regulation is dynamical
  we see it “static” because most time we cannot observe the transient period (in which the system reacts to the change), but just measure the new “steady state” in which the system resettles following a perturbation



# The “reverse engineering” paradigm



- what are then the perturbations?
  - everything that moves the cell from its “standard” working condition
  - biochemical, environmental, genetic, transcriptional, etc. examples: stress factors, starvation, infection, hormonal and growth factors; chemical inhibitors/activators, protein activity, metabolite concentration, gene overexpressions and inhibition, gene knockout and mutations, miRNA
  - perturbations could be
    - temporary (e.g. activating or inhibiting a signalling protein by phosphorylation) or permanent (e.g. gene knockout)
    - time dependent (e.g. time-varying stimulus) or static (e.g. gene knockout)
    - *local* (i.e., affecting a single gene) or *global* ( i.e., change in temperature or pH)
    - of *small amplitude* or *large amplitude*

# Network inference algorithms



- a few methods
  1. BAYESIAN NETWORK
     - attains a probabilistic graph through a bayesian learning
     - (exact) complexity: superexponential
  2. ASSOCIATION NETWORKS
     - learns a graph through a “similarity measure”
     - polynomial complexity
  3. LINEAR ODES MODELS
     - linear complexity
     - suffers from underdetermination
     - model-dependent



# Bayesian networks



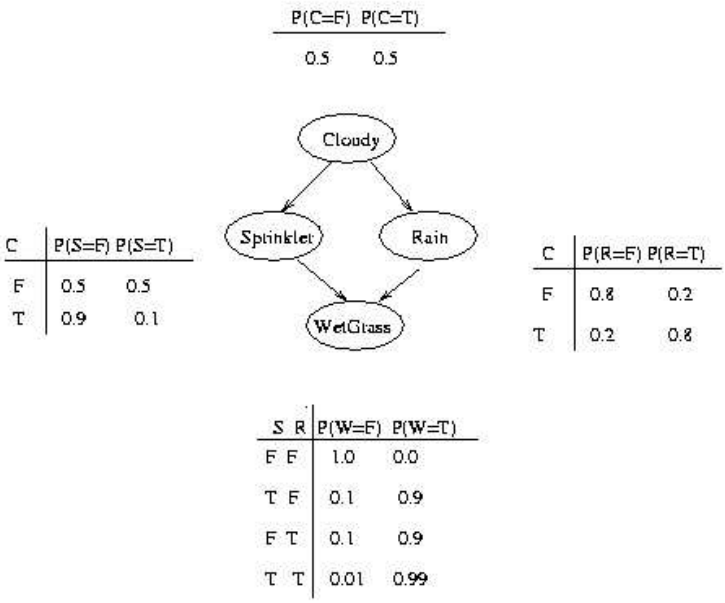
Bayesian networks

- are a probabilistic framework aiming at capturing the *conditional dependence* or *conditional independence* between “states” in a set of data.
- approach is statistic in nature $\Longrightarrow$able to cope with noisy data & not sufficiently many experimental data
- useful when each state depends only on a relatively small # of other components $\rightarrow$ networks with low connectivity
- Bayesian network can
  - learn the regulatory network
  - find the best set of parameters for the conditional distribution of that network
  - “best” is to be taken in a Bayesian sense as the most probable given the data

N. Friedman, M. Linial, I. Nachman, and D. Pe’er. *Using Bayesian Network to Analyze Expression Data* J. Computational Biology 7:601-620, 2000

# Bayesian networks

- example




- "A causes B" is the rule to construct the graph
- tables = conditional probability distribution





# Bayes rule

$$\text{posterior probability} = \frac{\text{marginal likelihood} \cdot \text{prior probability}}{\text{coefficient}}$$

- if $A$ and $B$ are independent events

$$p(A\,B) = p(A)\,p(B)$$

- if $A$ and $B$ are not independent

$$\begin{aligned} p(A\,B) &= p(A|B)p(B) \\ &= p(B|A)p(A) \end{aligned}$$

$\Longrightarrow$conditional probability

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

# Bayes rule: example



$$\begin{cases} p(C=1) & = 0.5 \\ p(C=0) & = 0.5 \end{cases} \qquad \begin{cases} p(R=1|C=1) & = 0.8 \\ p(R=1|C=0) & = 0.2 \end{cases}$$

- from the likelihood

$$\begin{aligned} P(R=1) &= p(C=0,\, R=1) + p(C=1,\, R=1) \\ &= p(R=1|C=0)p(C=0) + p(R=1|C=1)p(C=1) \\ &= 0.2 \cdot 0.5 + 0.8 \cdot 0.5 = 0.5 \end{aligned}$$

- Bayes rule

$$p(C|R) = \frac{p(R|C)p(C)}{p(R)}$$

- e.g. if we see it is raining $R = 1 \Longrightarrow$

$$p(C=0|R=1) = \frac{p(R=1|C=0)p(C=0)}{p(R=1)} = \frac{0.2 \cdot 0.5}{0.5} = 0.2$$

$$\left( \text{if instead } p(C=0) = 0.9 \quad \Longrightarrow \begin{cases} P(R=1) & = 0.26 \\ p(C=0|R=1) & = 0.69!! \end{cases} \right)$$



# Bayes rule: example



- how about if you observe the grass is wet?
  - is it because of
    - sprinkler

$$p(S=1|W=1) = \frac{p(S=1,\, W=1)}{p(W=1)} = 0.43$$

    - rain

$$p(R=1|W=1) = \frac{p(R=1,\, W=1)}{p(W=1)} = 0.7$$

  - from the joint probability, we deduce the different conditional probabilities
- Bayesian inference: find the probability of conditional events, given the Bayesian network, or find the conditional events and the network structure

# Bayesian networks



- Bayesian networks are graphical representations of *joint probability distributions* and consist of 2 components:
  1. an annotated *direct acyclic graph (DAG)* $G$ with
     - nodes = random variables $X_1, \ldots, X_n$ (e.g. $X_i$ = gene expression)
     - arcs = causal relationships between nodes $X_i \rightarrow X_j$
  2. conditional probability distributions $p(X_i|\text{ parents }(X_i))$ for each $X_i$
- the graph encodes the *Markov assumption*: each $X_i$ is independent of its non-descendants, given its parents
- joint distribution

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i|\text{ parents }(X_i))$$

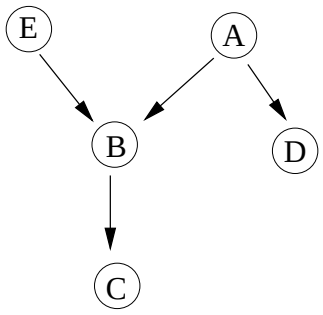
- on the joint distribution one can do inference and choose likely causalities (conditional distribution)



# Bayesian networks



- to reduce the number of conditionals to compute in the joint distribution: conditional independence
- from the Markov assumption, for all the non-descendent nodes there is conditional independence: $i(X;Y|Z)$ means $X$ is independent of $Y$ given $Z$
- example




  - conditional independeces

$$i(A;E),\quad i(B;D|A,E),$$
$$i(C;A,D,E|B)\quad i(D;B,C,E|A)$$

  - joint distribution

$$p(A,B,C,D,E) = p(A)p(B|A,E)p(C|B)p(D|A)p(E)$$

# Equivalence classes of Bayesian Networks



- a Bayesian network implies a set of independencies $I(G)$ (more than just the ones following the Markov assumption)
- Bayesian networks that have the same independencies belong to the same equivalence class
- example: $G:\ X \to Y$ and $G:\ Y \to X$ are equivalent
- rather than a DAG (Direct Acyclic Graph) a class is represented by a *PDAG: Partially Direct Acyclic Graph*: a graph such that
  - if there is a direct edge $X \to Y$ then all members of the equivalence class must contain the edge with the same direction
  - some edges may be nondirect $X - Y$ (meaning in the equivalence class both $X \to Y$ and $Y \to X$ are present)



# Variable representation



- Different types of representations for $X_1, \ldots, X_n$
  1. discrete variables: $X_i$ take values in a finite set
     - binary = $\{0,\ 1\}$
     - $\{$ low expression; normal; over-expressed $\}$ $\Longrightarrow$*multinomial distribution*
     - can capture combinatorial effects
     - discretization $\Longrightarrow$loss of information
  2. continuous variables: in order to compute posteriors in closed form one must use *linear Gaussian distributions*

$$p(X|u_1, \ldots, u_k) \sim N(a_0 + \sum_i a_i \cdot u_i,\ \sigma^2)$$

     - can capture only linear effects
  3. hybrid models: mix of the two cases

# Learning the network



- PROBLEM FORMULATION:

  given a training set $D = (x_1, \ldots, x_n)$ of independent instances of the random variables $X_1, \ldots, X_n$, find the network $G$ (or equivalence class of networks) that best matches $D$

- complete data: the entire "state vector" is measured $\Longrightarrow$"full observation, unknown structure" case.
- Learning the structure (e.g. via Bayesian score algorithms) is known to be a NP-hard problem (superexponential growth)
- from the Bayes rule

$$p(G|D) = \frac{p(D|G)p(G)}{p(D)}$$

  where
  - $p(G|D)$ = posterior probability on the network structure
  - $p(G)$ = prior probability on the network structure



# Learning the network



- take the log: Scoring function

$$\begin{aligned} S(G:\ D) &= \log p(G|D) \\ &= \log p(D|G) + \log p(G) + C \end{aligned}$$

  where
  - $C = -\log p(D)$ = const.
  - $p(D|G)$ = marginal likelihood = averages the probability of the data over all possible structures assignable to $G$

$$p(D|G) = \int p(D|G,\theta)p(\theta|G)d\theta$$

  - complete data $\Longrightarrow$integral is treatable
- solution:
  - model:

$$\max_G S(G:\ D)$$

  - parameters

$$\max_\theta S(\theta|G^*,\ D)$$

# Learning the network



- this is still NP-hard
- simplifications
  - complete data $\Longrightarrow G$ and $G'$ with equivalent graphs give the same posterior score
  - score is decomposable

$$S(G : D) = \sum_i \text{ScoreContribution}\,(X_i, \text{parents}(X_i) : D)$$

    contribution of each $X_i$ to the total score depends only on its own value and on the value of its parents in $G$
- heuristcs:
  - to cope with complexity: local search procedure that changes one arc at each move: evaluation of the gain made by adding/removing/reversing a single arc
  - further complexity reduction: # of parents is bounded ("fan-in") $\Longrightarrow$sparsness
  - greedy algorithm, but performing well in practice



# Discovering features



- result is a joint distribution over all random variables
- rather than obtaining a single "optimal" model $G^*$, one gets a set of models with different high scores
- idea: compare highly scoring models for common features
- simplest features: pairwise relations $\rightarrow$ *Markov Relations*
  - Markov blanket = minimal set of variables that shield $X$ from the rest of the variables in the model $\Longrightarrow X$ is independent from the rest given the blanket
  - 2 nodes $X$ and $Y$ in the blanket either are directly linked or share parenthood of a node
  - biologically it means that $X$ and $Y$ are related in a joint process
- assessing the confidence of a model: *bootstrap* = slightly perturb your data, re-apply the learning procedure and verify the overlap

# Drawbacks



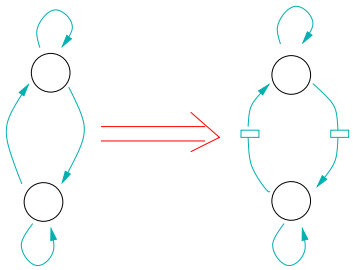
- finding the “best” structure is a NP-hard problem
- PDAG rather than DAG: not all cause-effect relations can be resolved: Bayesian network is a model of *dependencies* between variables rather than causality
- sparseness assumption is “initialized” by genes that are co-expressed in a clustering: this is reasonable but may arbitrarily and erroneously restrict the search space
- Graph must be *Acyclic*: the network found has no regulatory “loops”

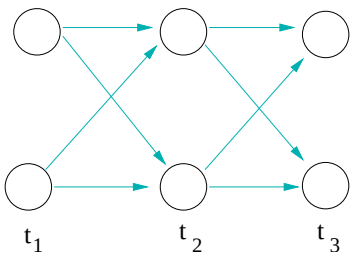


# Improvements and developements



- to cope with unmeasured quantities (e.g. missing data: part of the state vector not measured in some of the experiments): hidden Markov models
- to cope with acyclicity: Dynamic Bayesian Networks
  - idea: feedback is seen as a delay unfoding in time into an acyclic graph

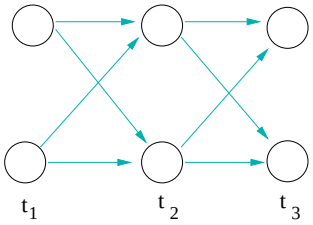




# Dynamic Bayesian Networks



D. Husmeier *Sensitivity and specifi city of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks* Bionformatics 19 p.2271-82, 2003




- each time slice is a Bayesian network
- to tame complexity: transition probabilities between slices is the same $\forall\ t$
  $\rightarrow$ homogeneous Markov model
- intraslice connections (i.e., instantaneous interactions) are not allowed
- directional ambiguity is avoided: temporal causality



# Dynamic Bayesian Networks: drawbacks



- the bottleneck is that the time series of data are short $\Longrightarrow$the posterior distribution over network structure is vague...
- other problems:

$$p(G|D) = \frac{p(D|G)p(G)}{p(D)}$$

  - prior on network structure $p(G)$ has a non-neglegible influence on posterior $p(G|D)$
  - $\Longrightarrow p(G)$ should capture known features of biological networks
  - $\Longrightarrow$need to know a lot to initialize $G$
  - needless to say: computational complexity