

AFRICAN CENTERS OF EXCELLENCE IN BIOINFORMATICS

KAMPALA, UGANDA

Gene Regulatory Networks 1



Yunhua Zhu,
PhD in stem cell biology

Computational Genomics
Specialist – Transcriptomics

Bachelor in biochemistry @NUS
PhD in stem cell biology @NUS
Postdoc in neurodegeneration, single cell biology
@JHU
Bioinformatician @ NIH

Recent projects

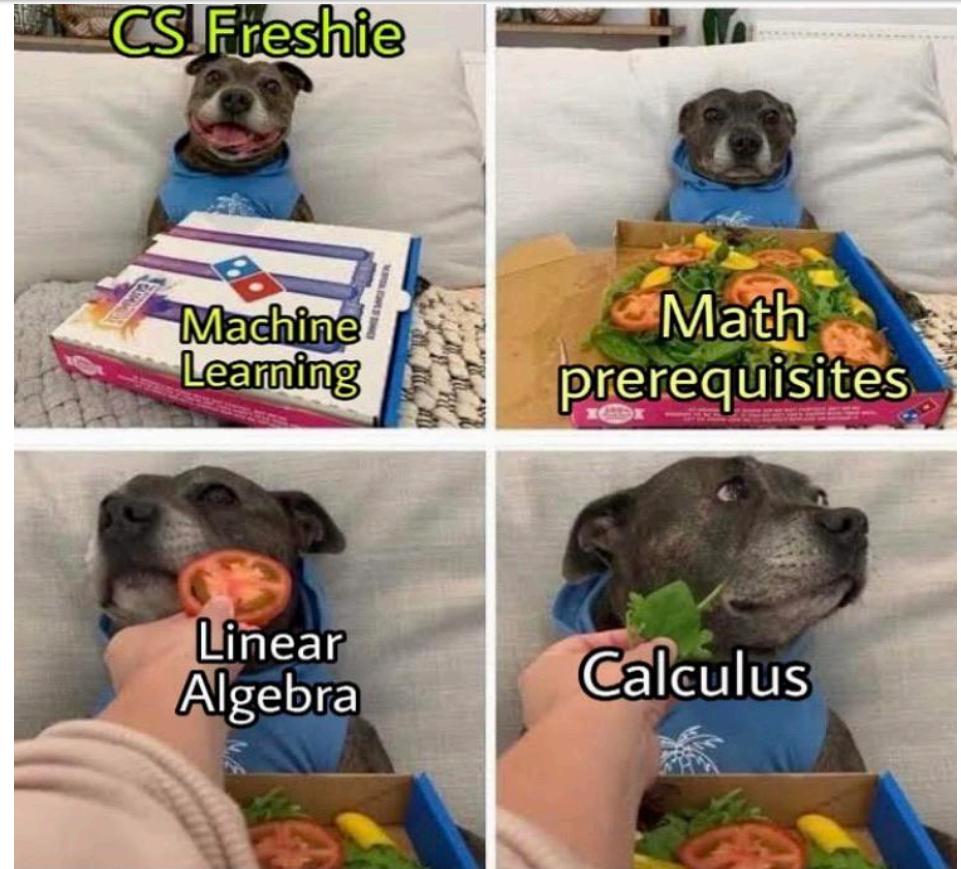
- Single cell analysis of cancer stem cells
- Single cell analysis of immune response in lung cancer
- Bulk RNA-seq on irradiation, HIV blood samples
- PLEASE JOIN THE DISCUSSION

Today's Instructor

- Bioinformatics and Computational Biosciences Branch (BCBB), NIAID
- National Institutes of Health, Bethesda, MD USA.
- Contact our team via email:
 - GitHub: git clone https://github.com/niaid/Gene_Regulatory_Networks
 - Instructor: zhuy16@nih.gov
 - Server: ssh <username@137.63.194.9>,
`/home/bcbb_teaching_files/`
 - Googledoc, <https://tinyurl.com/zhu-GRN>
 - Email: bioinformatics@niaid.nih.gov
 - LinkedIn: <https://tinyurl.com/zhu-linkedin>

Personal experience: learning bioinformatics

- Sounds Nice
- A lot of difficulties
- Luckily...





Biology in the modern era ...

Microarray
RNA sequencing
Epigenomes
Single cell transcriptomics and Single cell
Epigenomics
Microbiomes
Hard to digest
.....
... Where is my key factor to study?

The dog is FAAAT!!!



YouTube
Google
Wikipedia
stackoverflow

Statistics,
Mathematics,
Programming! All
good stuff !!!

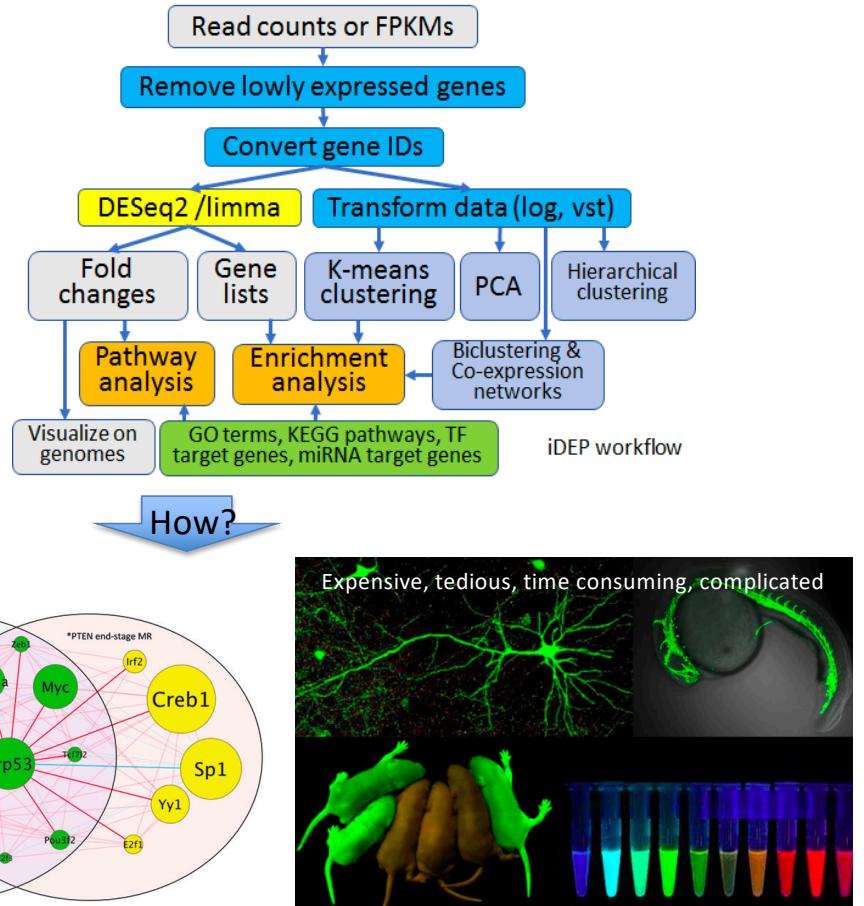


Objectives: achieve a general overview of gene regulatory network (GRN) analysis

- Biological rational
- Theoretical bases of theories, intuitions and tools.
- Practical demonstration
- Links to further resources in case interested
- A discussion, not a lecture

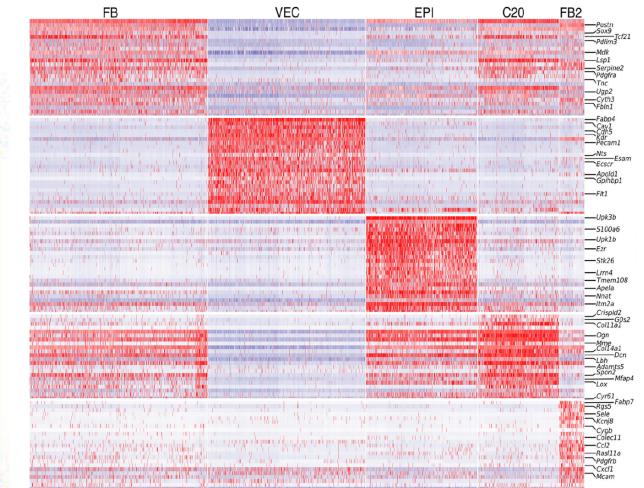
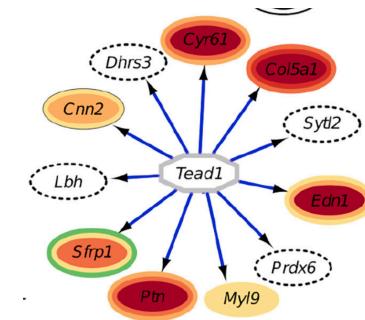
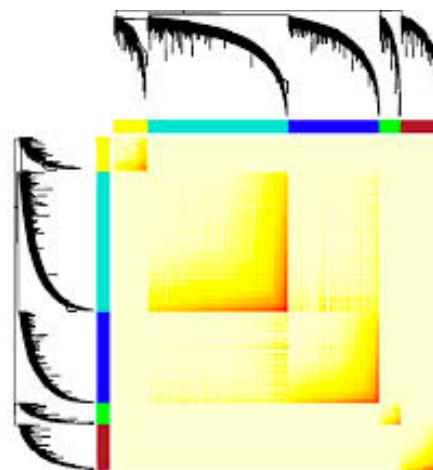
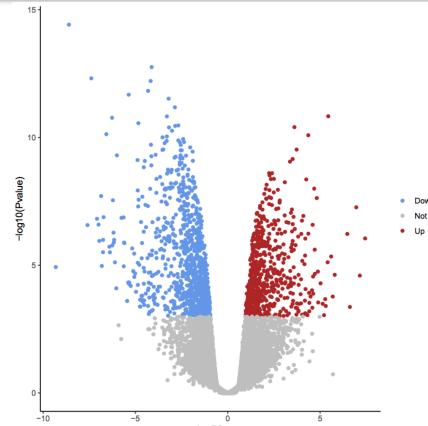
Why study gene regulatory networks

- Why
 - High throughput are screening procedures
 - Gene lists do not inform relationship between genes
 - Regulatory information is hidden in the statistical relationships
 - Down-stream validation is costly, tediously and risky
- Aims
 - To find regulatory network, to inform finding of key candidate genes/master regulators for validation
- Challenges
 - Data are massive, and mostly static that do not reflect dynamic regulation
 - Relationships can take various forms, linear or non-linear



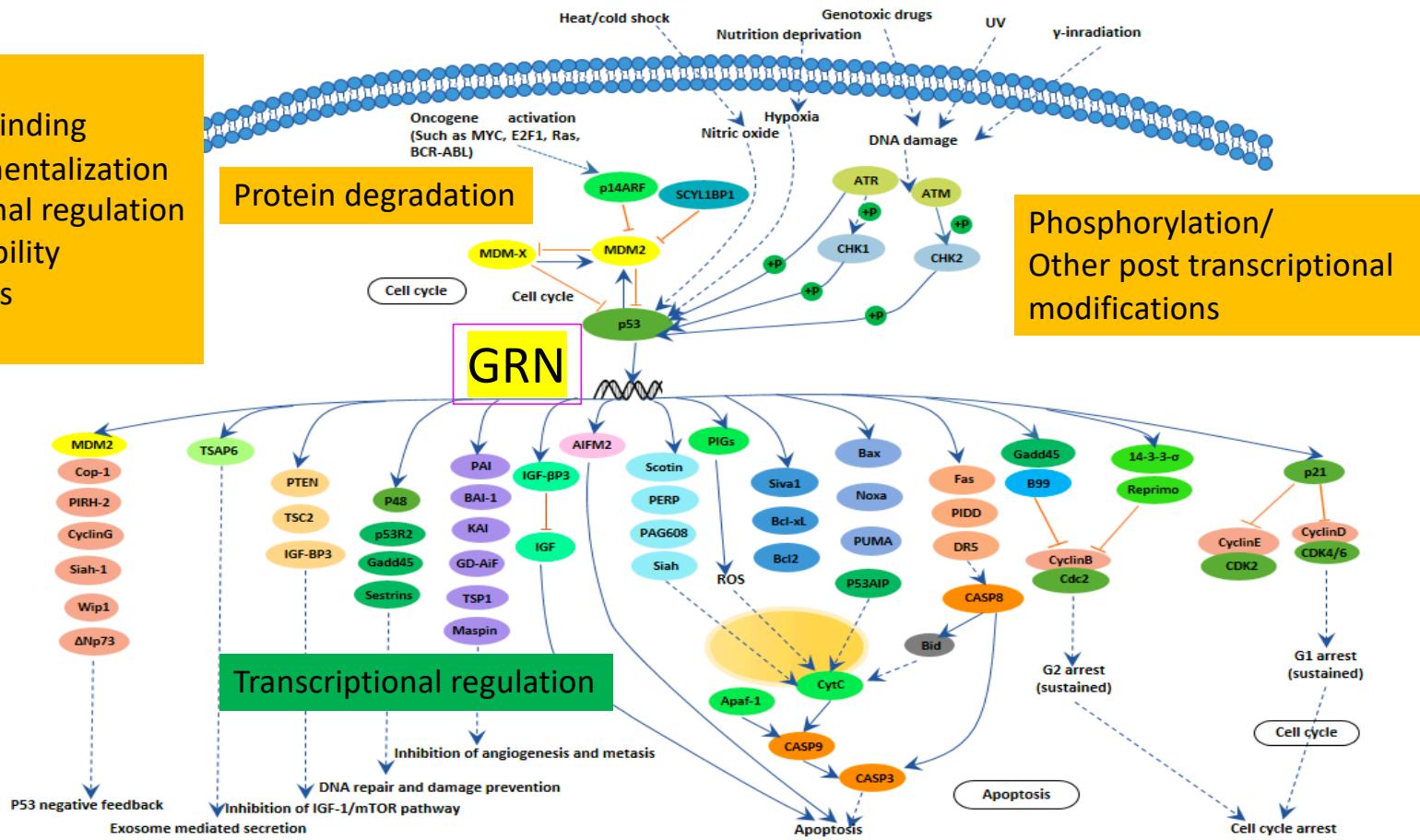
Prerequisite: identify a group of genes

- How to get a set of interesting genes?
 - Top variable genes
 - Differential expression analysis
 - Unsupervised Clustering
 - WGCNA: Weighted Gene Co-expression Network Analysis



A side note: GRN is only a subset of cellular networks

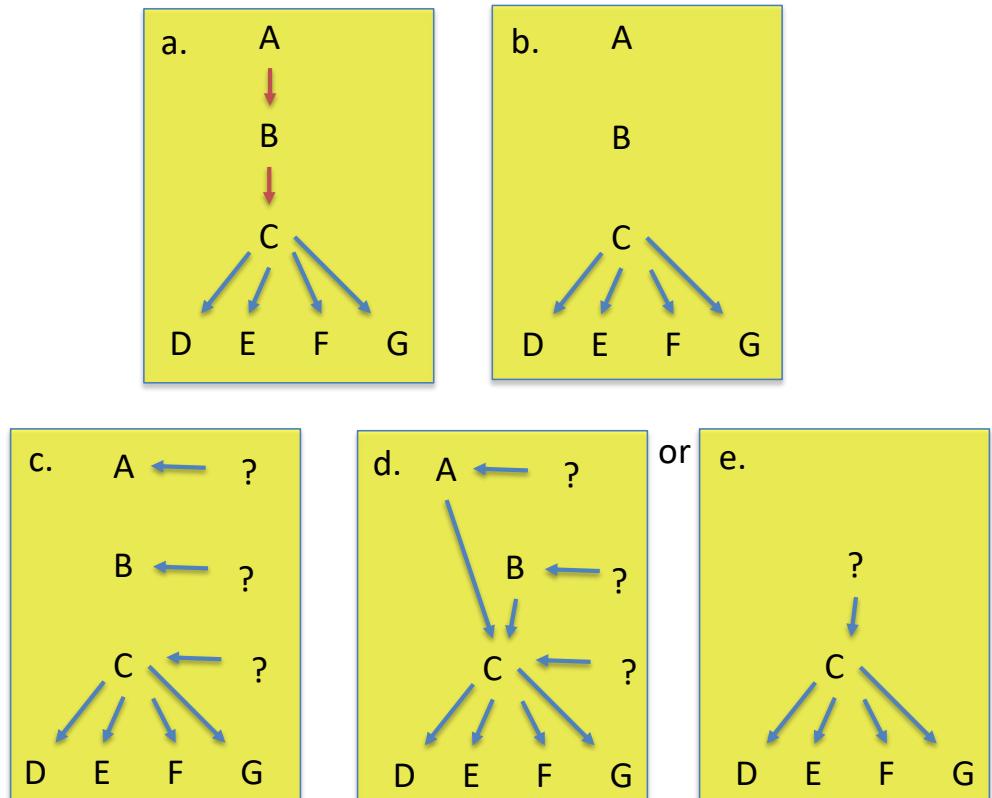
Others
 Cofactor binding
 Compartmentalization
 Translational regulation
 mRNA stability
 Epigenetics
 ...





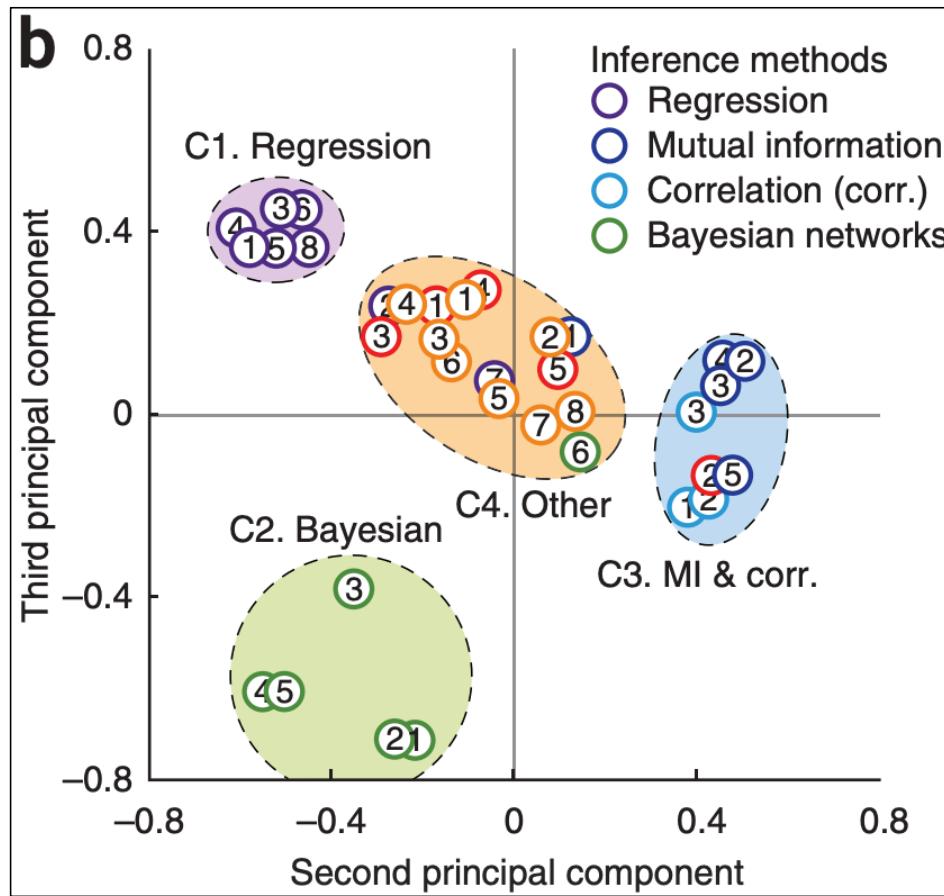
A schematic illustration on the difference

- (a) conventional structure of intracellular signaling
- (b) Biochemical link will be invisible in GRN
- (c) Other regulators kick in
- (d) Upstream factors may be independent from each other in the transcriptional GRN
- (e) Upstream factors may be totally invisible in the analysis



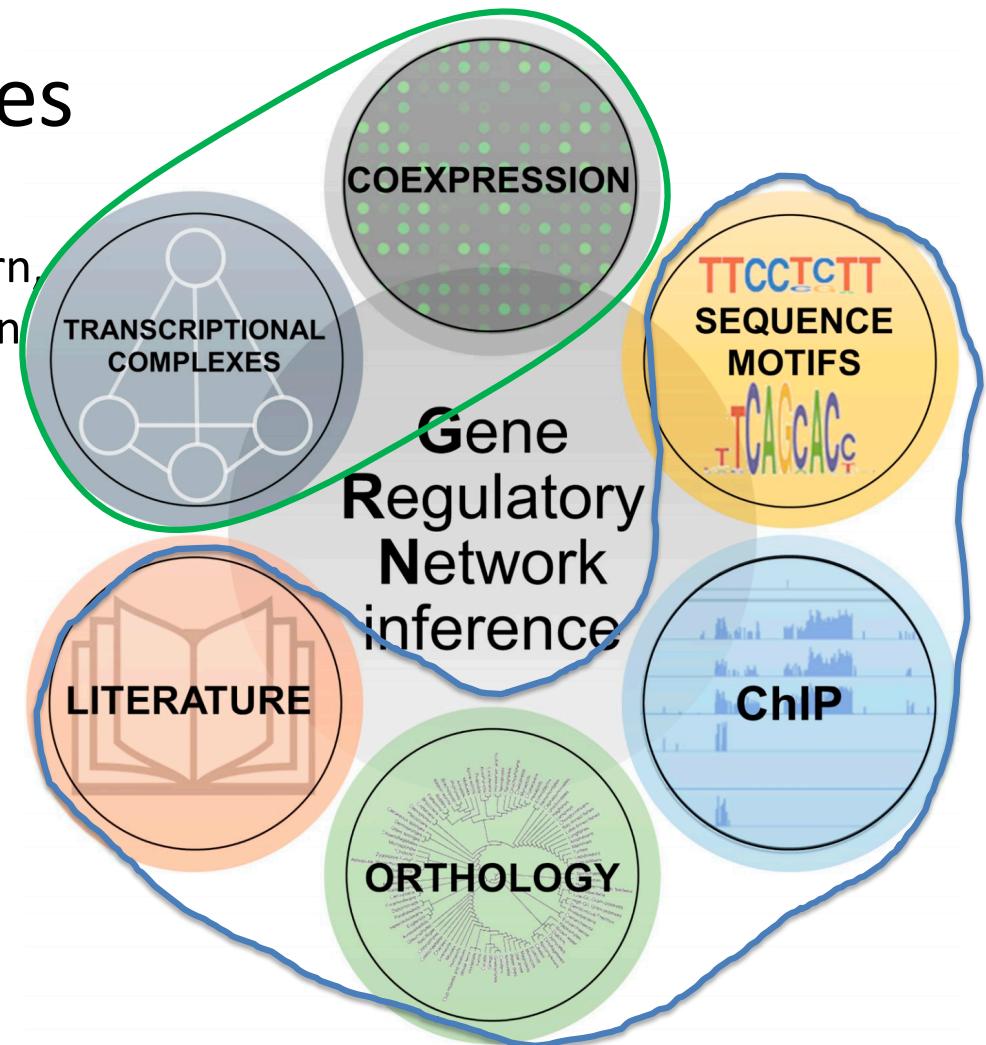
DREAM5– Network Inference Challenge (2010)

Dialogue for Reverse Engineering Assessments and Methods (Open science challenges)



Computational Strategies

- Infer network through expression pattern, based on statistical relationship between genes.
 - Linear relationships
 - Mutual information—linear/non-linear statistical relationships
 - Bayesian network /Bayes Net
 - Regression tree + random forest
- Predicting master regulators through promoter sequences
 - cis-Target and iRegulon
- Combinatory approach
 - SCENIC (GENIE3 and cis-Target)



Mercatelli et al., 2019

Overview of Tools in GRN inference

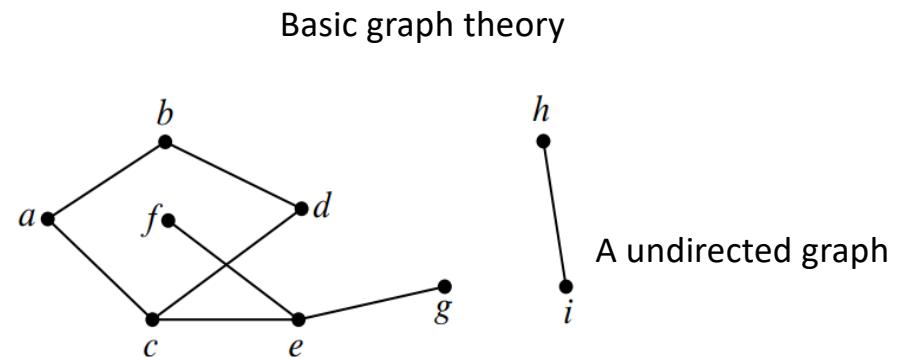
| Software | ARACNE | NetworkInference/ PIDC | bnlearn | GENIE3 | iRegulon | SCENIC |
|----------------------|--|---|---|---|---|---|
| semantics | Mutual information | Partial Information Decomposition | Bayes theory | Random Forest, Regression tree | Promoter and TF binding sequence, database | Combination of regression and promoter sequence |
| years published | 2006 | 2017 | 2009 | 2010 | 2014 | 2017 |
| No. of cited | 2179 | 82 | 894 | 658 | 337 | 265 |
| FullName/explanation | Algorithm for the Reconstruction of Accurate Cellular Networks | Using proportional unique contribution (PUC) to a target gene | Bayes net structure and parameter learning, causality | GEne Network Inference with Ensemble of trees | reverse-engineer the transcriptional regulatory network with regulatory sequence analysis | single-cell regulatory network inference and clustering |
| Implementation | GUI (geWorkbench) | Julia | R | R | GUI (Cytoscape) | R, Python |
| type of experiment | Microarray, bulk RNA-seq | Single cell data | General | single cell data | a list of gene names | single cell data |
| input format | csv | csv | csv | csv | a list | csv/loom file |
| output | network file | network file | directed network file | network file | network file/binding sequences | network file/heatmap |

Today

Next week

Graph/network theory: What are we estimating

- Graph (network) theory
 - Vertex (Vertices)/nodes
 - Genes
 - Edges, statistical interactions
 - Undirected graph
 - ARACNE, PIDC
 - Directed Acyclic Graph (DAG)
 - Bayes net
 - Edge weights

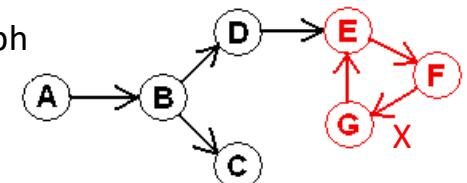


The graph in Figure 5.1 is expressed mathematically as $G = (V, E)$, where:

$$V = \{a, b, c, d, e, f, g, h, i\} \text{ Vertex/node}$$

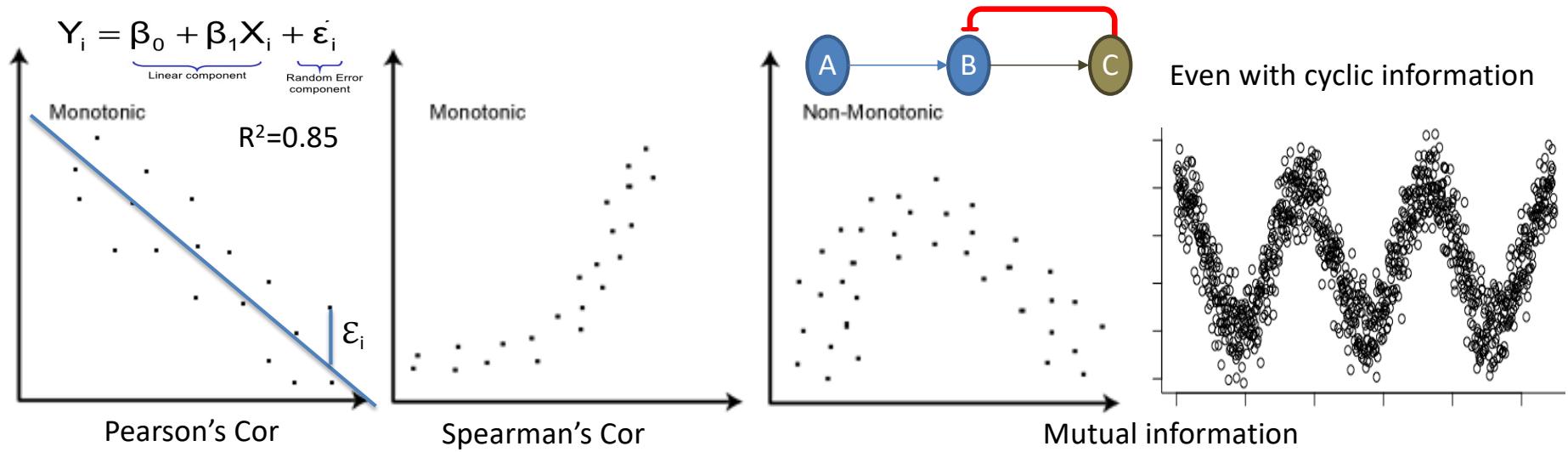
$$E = \{\{a, b\}, \{a, c\}, \{b, d\}, \{c, d\}, \{c, e\}, \{e, f\}, \{e, g\}, \{h, i\}\}.$$

Directed Acyclic Graph
(DAG)

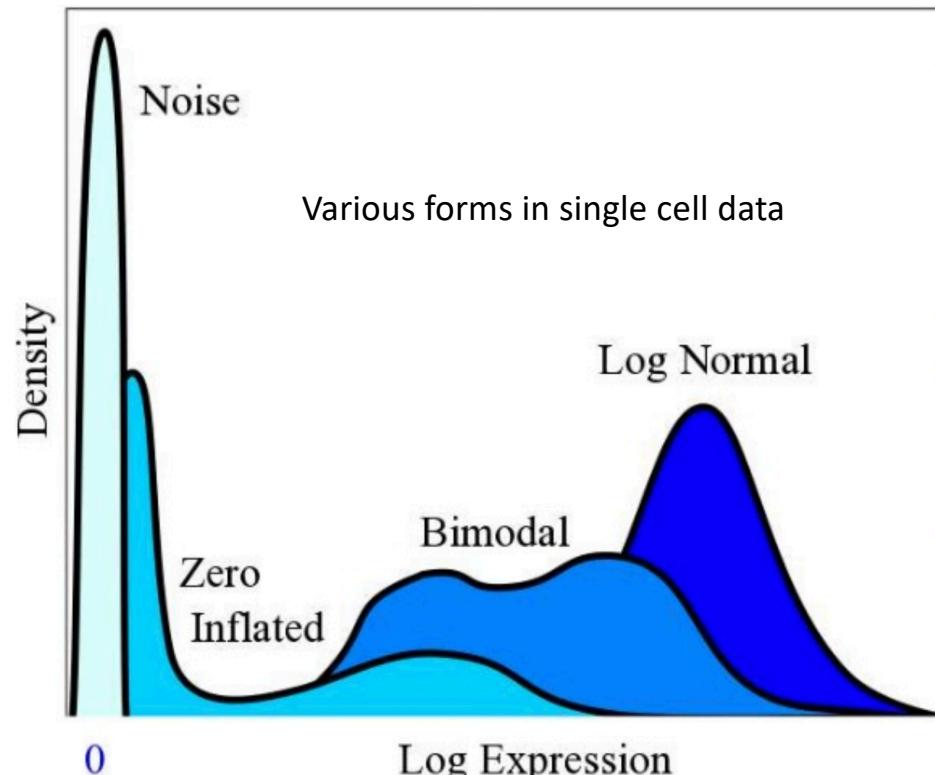


Types of statistical relationship

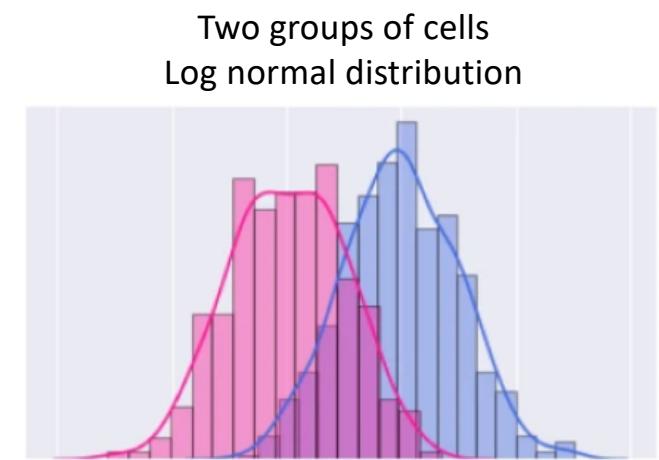
- Pearson's correlation, low computational complexity, and their capability of inferring genome-wide networks even when a relatively low number of samples ($n = 50$) is available
- Spearman's correlation, rank based, no assumption on normal distribution
- Mutual information, any statistical relationship. But requires N to be sufficiently large for a good approximation



Various types of distributions of gene expression



<https://www.slideshare.net/TimothyTickle/introduction-to-singlecell-rnaseq>

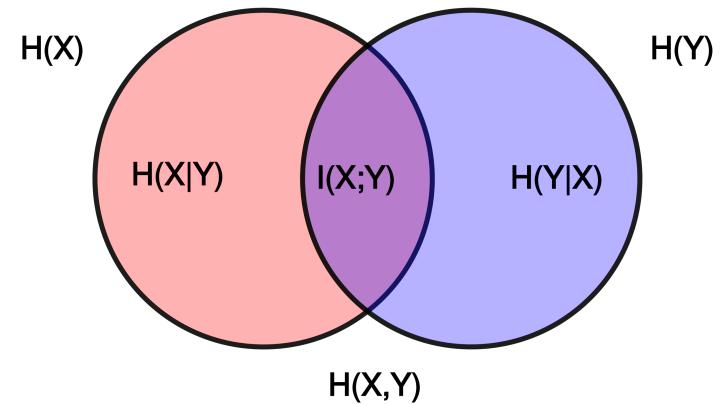


Gene expression
(Expression values of a given gene in different replicates /cells)

Information theory

- Entropy
 - A measure of information content (bit as its unit)
 - Information content is coded within the statistical distribution of variables and can be calculated as Shannon's entropy
 - Common statistical patterns (“correlation”) between two random variables can be calculated as their mutual information

$$\text{entropy}(\text{flip a coin once}) = -\log_2\left(\frac{1}{2}\right) = 1 \text{ (bit)}$$

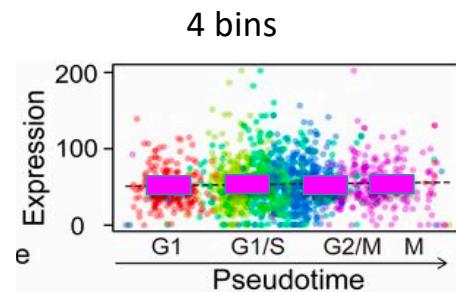


Shannon's Entropy—measure of information content. (Not the entropy in physics)

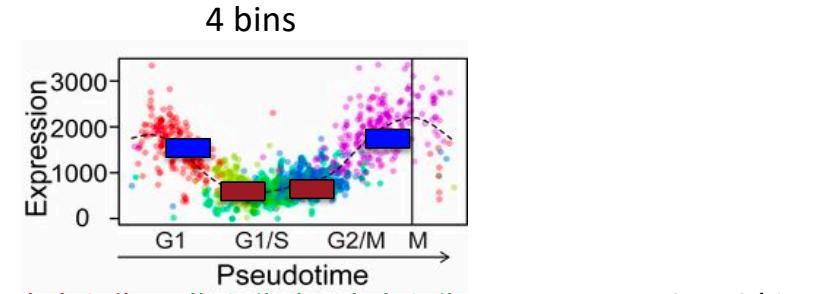
- Example:
 - Gene1**
 - Probability of gene expression at 50 is 100% because all states average the same
 - Gene2**
 - Probability of gene expressed at high level, 2/4=50%
 - at low level, 2/4=50%

$$\begin{aligned}
 H(x) &= - \sum_x P(x) \cdot \log P(x) \\
 &= \sum_x P(x) \cdot \log \left(\frac{1}{P(x)} \right)
 \end{aligned}$$

↓ ↓
 The prob. of event X WHAT IS THIS ?



$$H_1 = 1/1 \times \log_2(1/1) + \text{same} + \text{same} + \text{same} = 0 \text{ bits}$$



$$H_2 = 1/(50\%) \times \log_2(1/50\%) + 1/(50\%) \times \log_2(1/50\%) + \text{same} + \text{same} = 2 \times 4 = 8 \text{ bits}$$

PNAS Spatial transcriptome profiling by MERFISH reveals subcellular RNA

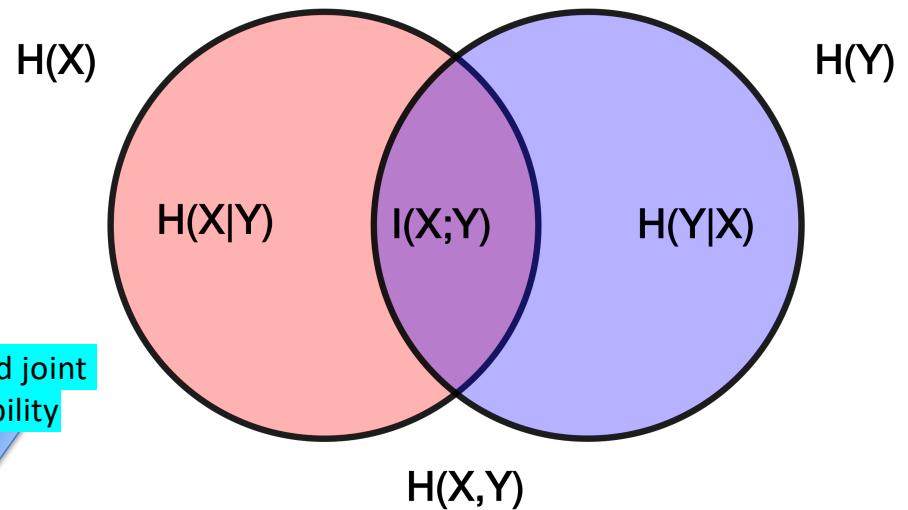


Method 1: ARACNE

Algorithm for the Reconstruction
of Accurate Cellular Networks

Mutual Information (MI)

- $H(X), H(Y)$ are the information contents for X and Y
- $I(X;Y)$ is the mutual information of X and Y



All the information you have got = entropy

Observed joint probability

Mutual information

$$MI_{ij} = \sum_{x_i} \sum_{x_j} p(x_i, x_j) \log_2 \frac{p(x_i, x_j)}{p(x_i)p(x_j)}$$

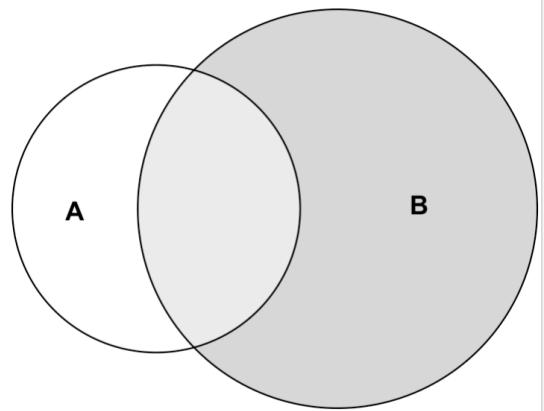
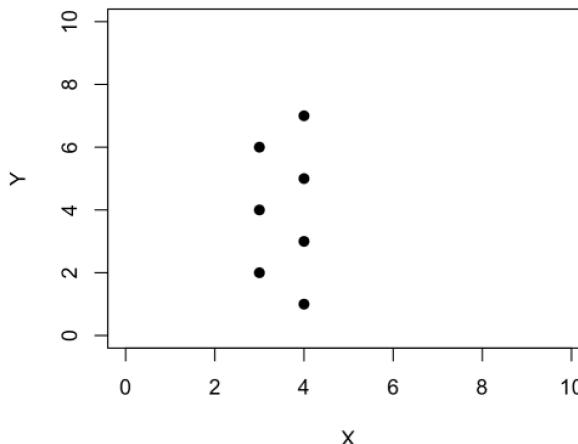
Equal to joint probability assuming x_i, x_j co-occur purely by chance.

Mutual information captures non-linear relationships

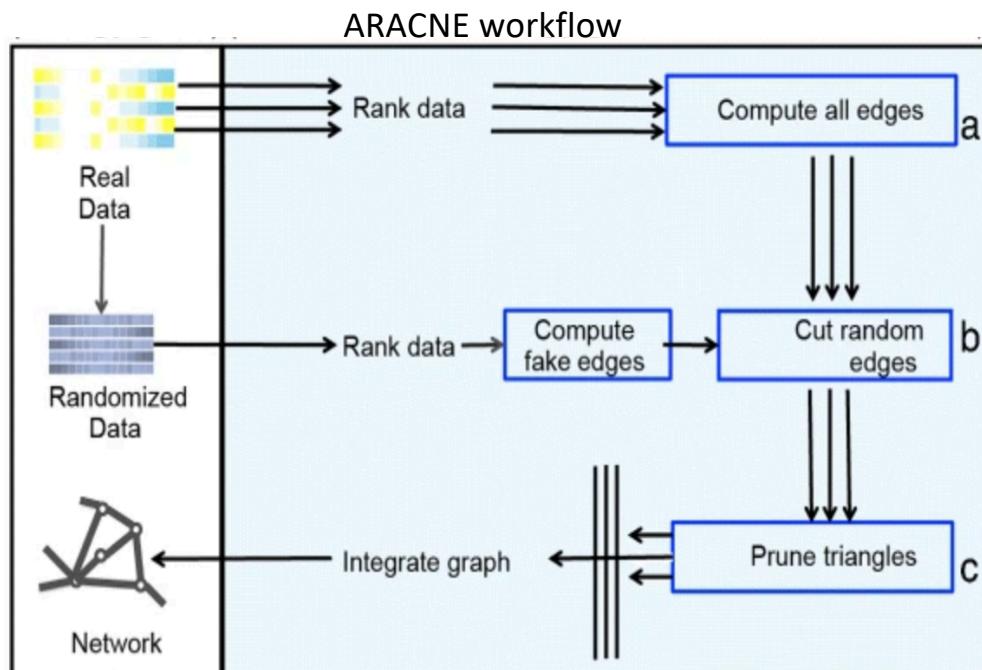
In R:

```
X=c(4,3,4,3,4,3,4)  
Y=c(1,2,3,4,5,6,7)  
plot(X,Y)
```

```
#install.packages("infotheo")  
library(infotheo)  
entropy(X)  
entropy(Y)  
mutinformation(X,Y)  
#install.packages("eulerr")  
library(eulerr)  
fit <- euler(c(A = entropy(X), B = entropy(Y), "A&B" =  
mutinformation(X,Y)))  
plot(fit)
```



Determine interaction through MI



Data processing inequality (DPI), to remove indirect interaction

$$x_i \rightarrow x_j \rightarrow x_k,$$

then,

$$I(x_i, x_k) \leq \min \{ I(x_i, x_j), I(x_j, x_k) \}$$

If i and k are not directly connected, the correlation of them will not be high as a direct connection.

Illustration of how mutual information is calculated with 3 variables
<https://www.youtube.com/watch?v=3iplAfGzzI>

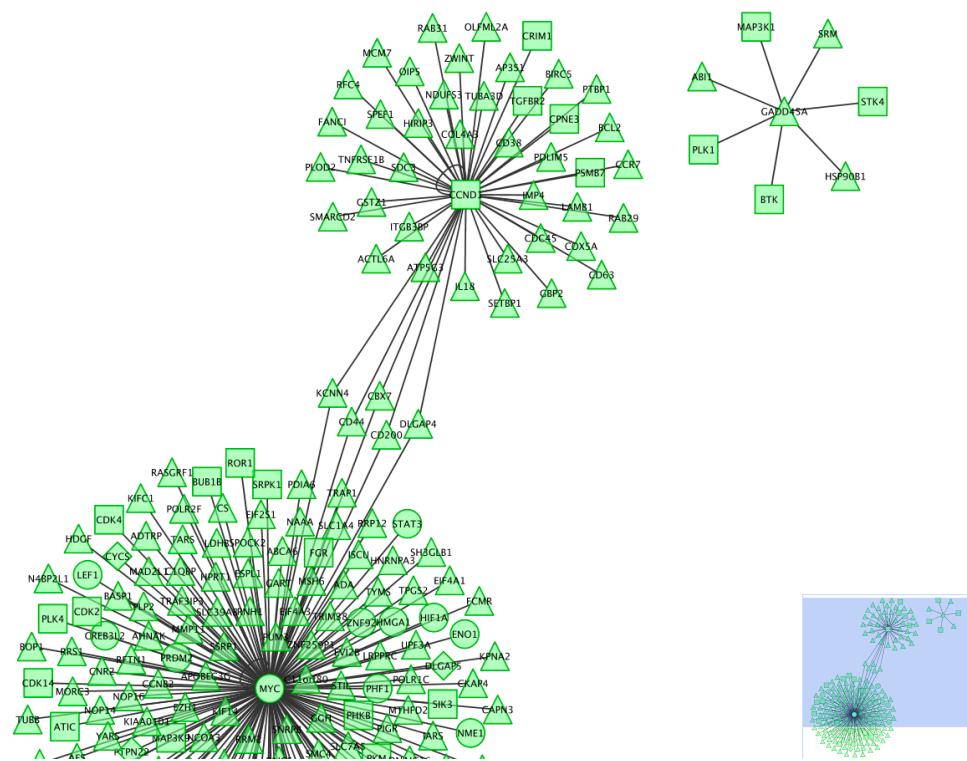


Notes on running ARACNE

- geWorkbench will have ARACNE by default
 - http://wiki.c2b2.columbia.edu/workbench/index.php/Download_and_Installation
- What data to use for ARACNE
 - The Bcell-100.exp as an expression file
 - http://wiki.c2b2.columbia.edu/workbench/index.php/Tutorial - Data#Tutorial_data_files
 - Use HG_U95Av2.na36.annot.csv - most updated version
 - To download file need registration on ThermoFisher website!!! Already downloaded in github.
- Video Tutorial: <https://www.youtube.com/watch?v=CU4nukYswt0>
- The layout of the geWorkbench is different from the video
 1. The project file cannot be created.
 2. The window looks different as well; no sub-field of windows at the bottom.

Demo ARACNE

- Using a microarray data
 - The interface of geWorkbench
 - setting parameters
 - Run ARACNE
 - Export from geWorkbench
 - Import to Cytoscape
 - Visualization





Discussion on ARACNE

- Good tool to find potential target genes with known TF
- Most widely used GRN software
- Straightforward concept of mutual information
- Processing inequality to break indirect linkages
- Too old, too slow; newer implementation may be useful

Bioinformatics, 2016 Jul 15;32(14):2233-5. doi: 10.1093/bioinformatics/btw216. Epub 2016 Apr 23.

ARACNe-AP: gene network reverse engineering through adaptive partitioning inference of mutual information.

Lachmann A¹, Giorgi FM¹, Lopez G¹, Califano A¹.

Author information

1 Department of Systems Biology, Columbia University, New York, NY, USA.

Abstract

The accurate reconstruction of gene regulatory networks from large scale molecular profile datasets represents one of the grand challenges of Systems Biology. The Algorithm for the Reconstruction of Accurate Cellular Networks (ARACNe) represents one of the most effective tools to accomplish this goal. However, the initial Fixed Bandwidth (FB) implementation is both inefficient and unable to deal with sample sets providing largely uneven coverage of the probability density space. Here, we present a completely new implementation of the algorithm, based on an Adaptive Partitioning strategy (AP) for estimating the Mutual Information. The new AP implementation (ARACNe-AP) achieves a dramatic improvement in computational performance (200x on average) over the previous methodology, while preserving the Mutual Information estimator and the Network inference accuracy of the original algorithm. Given that the previous version of ARACNe is extremely demanding, the new version of the algorithm will allow even researchers with modest computational resources to build complex regulatory networks from hundreds of gene expression profiles.

AVAILABILITY AND IMPLEMENTATION: A JAVA cross-platform command line executable and detailed usage guide are freely available on Sourceforge (<http://sourceforge.net/projects/aracne/>).

CONTACT: califano@c2b2.columbia.edu

SUPPLEMENTARY INFORMATION: Supplementary data are available at Bioinfo

© The Author 2016. Published by Oxford University Press.

Bioinformatics, 2019 Jun 1;35(12):2165-2166. doi: 10.1093/bioinformatics/bty907.

SJARACNe: a scalable software tool for gene network reverse engineering from big data.

Khatamian A¹, Paull EO², Califano A², Yu J¹.

Author information

1 Department of Computational Biology, St. Jude Children's Research Hospital, Memphis, TN, USA.

2 Department of Systems Biology, Columbia University, New York, NY, USA.

Abstract

SUMMARY: Over the last two decades, we have observed an exponential increase in the number of generated array or sequencing-based transcriptomic profiles. Reverse engineering of biological networks from high-throughput gene expression profiles has been one of the grand challenges in systems biology. The Algorithm for the Reconstruction of Accurate Cellular Networks (ARACNe) represents one of the most effective and widely-used tools to address this challenge. However, existing ARACNe implementations do not efficiently process big input data with thousands of samples. Here we present an improved implementation of the algorithm, SJARACNe, to solve this big data problem, based on sophisticated software engineering. The new scalable SJARACNe package achieves a dramatic improvement in computational performance in both time and memory usage and implements new features while preserving the network inference accuracy of the original algorithm. Given that large-sampled transcriptomic data is increasingly available and ARACNe is extremely demanding for network reconstruction, the scalable SJARACNe will allow even researchers with modest computational resources to efficiently construct complex regulatory and signaling networks from thousands of gene expression profiles.

AVAILABILITY AND IMPLEMENTATION: SJARACNe is implemented in C++ (computational core) and Python (pipelining scripting wrapper, ≥3.6.1). It is freely available at <https://github.com/jyyulab/SJARACNe>.

SUPPLEMENTARY INFORMATION: Supplementary data are available at Bioinformatics online.

© The Author(s) 2018. Published by Oxford University Press.

<https://github.com/califano-lab/ARACNe-AP>



Demonstration on ARACNE in geWorkbench



Method 2:

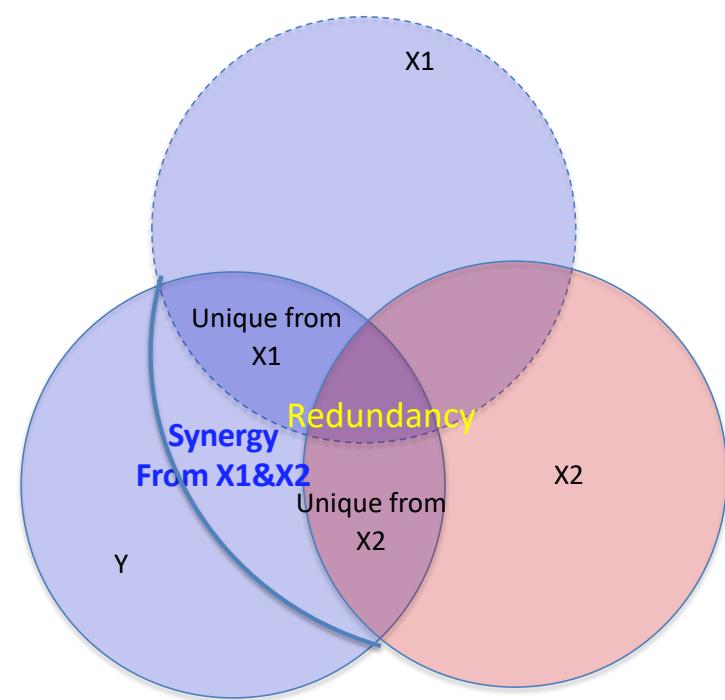
Partial Information Decomposition (NetworkInference for single cell data)

Partial Information Decomposition

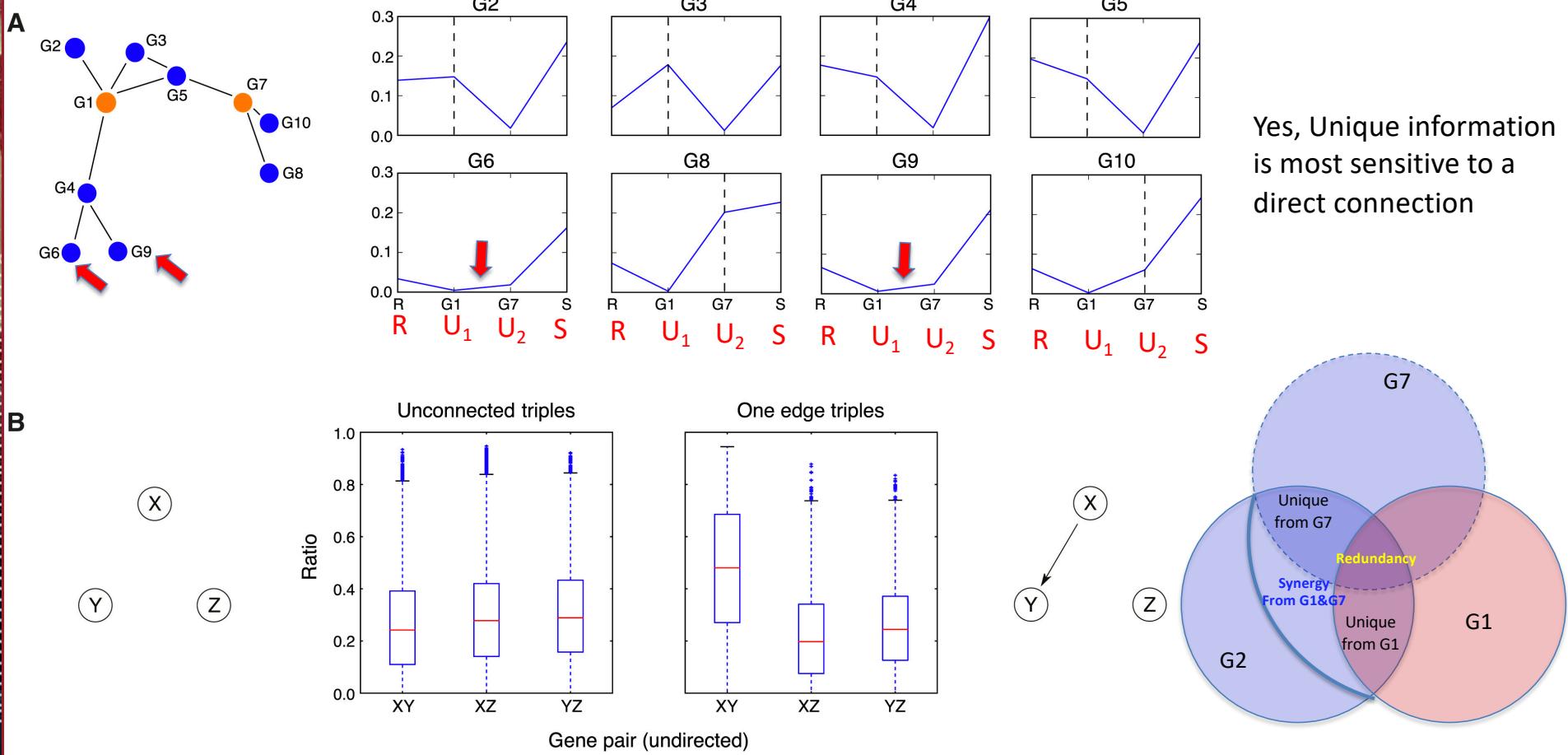
- Consider three variables X_1 , X_2 and Y
- Consider that Y is dependent on X_1, X_2
- The information from X_1 and X_2 that matches Y can be decomposed to 4 components (U_{X_1} , U_{X_2} , Redundancy and Synergy)

Information about Target Y provided by two sources X_1, X_2 .

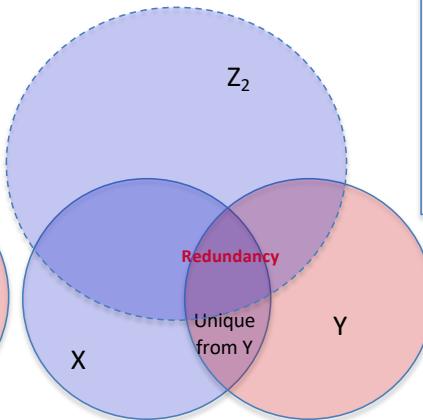
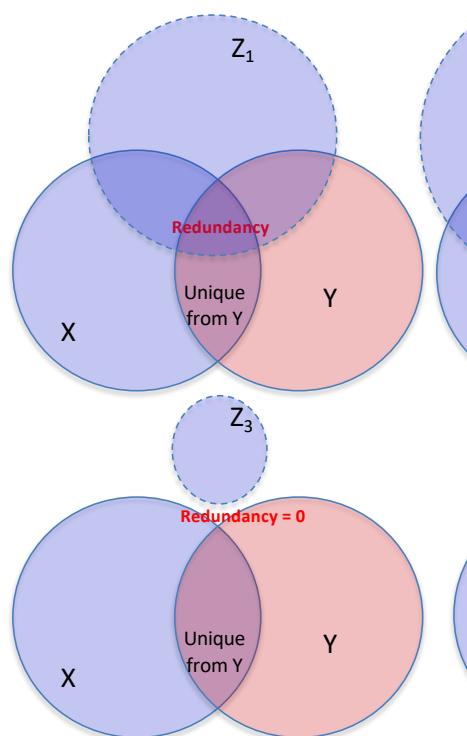
$$I(Y; X_1, X_2)$$



Are any of these components an indicator of a network connection?

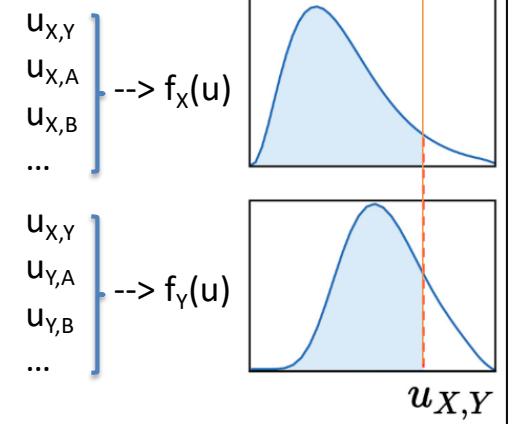
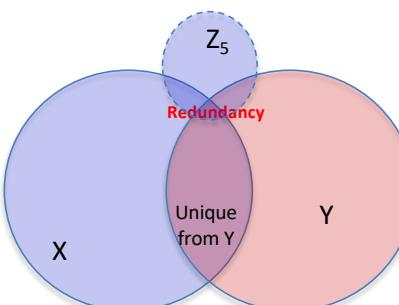
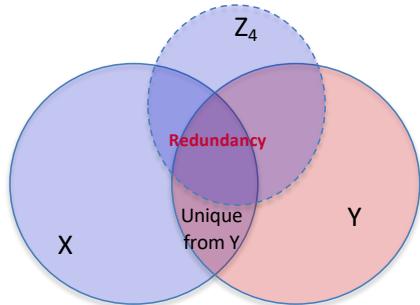
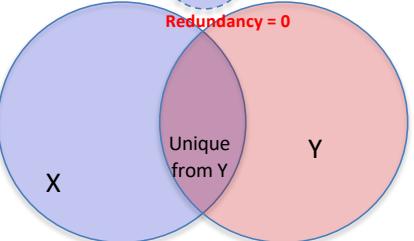


Proportional Unique Contribution



$$u_{X,Y} = \sum_{Z \in S \setminus \{X,Y\}} \frac{\text{Unique}_Z(X;Y)}{I(X;Y)} + \sum_{Z \in S \setminus \{X,Y\}} \frac{\text{Unique}_Z(Y;X)}{I(X;Y)}$$

... ...



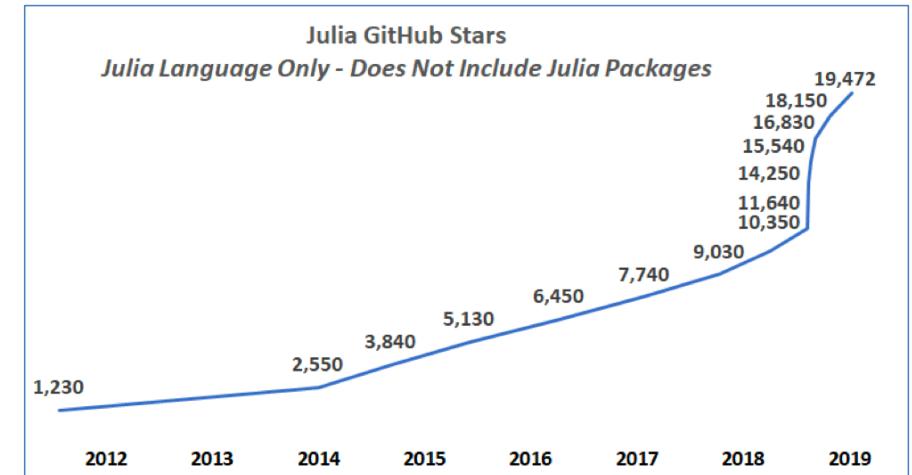
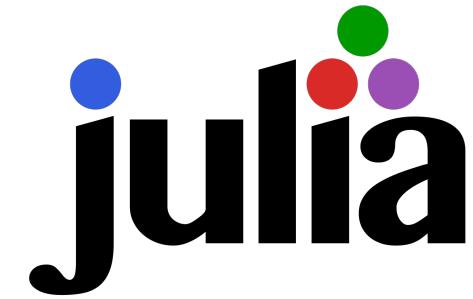


Discussion on NetworkInference

- Good at discovery of single-edge connection, may systematically loss two-edge, 3-edge connections
- Relatively new, not sufficiently validated
- Easy to use in Julia, may need HPC to run large data
- Undirected, and no attempt is made to find the directionality of a regulation

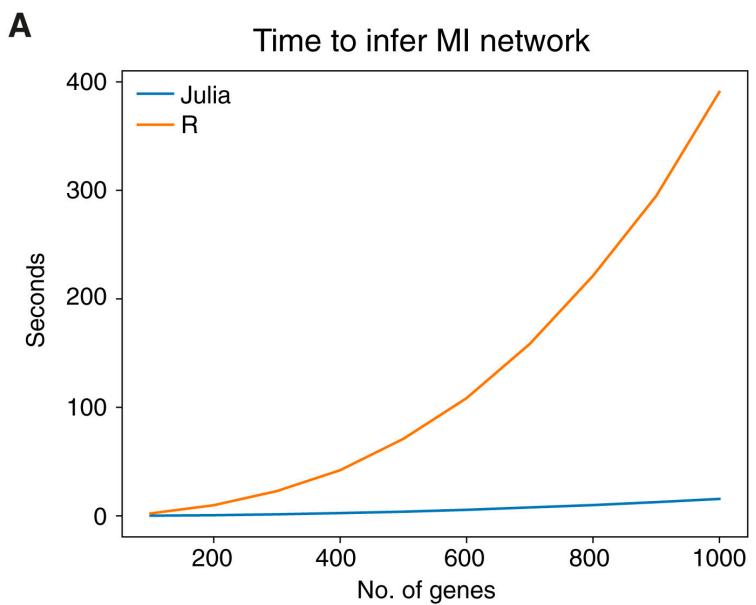
Julia: A fresh approach to numerical computing

- A young language developed by researchers in MIT on 2012
- Advantage, speed of C and intuitive as python
- Use all the libraries developed for python
- Disadvantage, changes a lot as versions upgrade – not very stable



Implementation in Julia

- Using NetworkInference in Julia
 - Install Julia
 - <https://julialang.org/downloads/>, download the binary file for Generic Linux 64-bit
 - Install NetworkInference, simple
 - use Pkg
 - Pkg.add("NetworkInference")
 - use NetworkInference

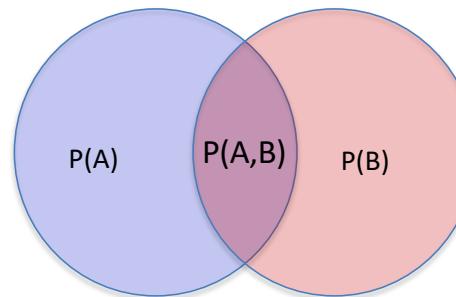




Demonstration on NetworkInference in Julia

- Import data
 - Run NetworkInference
 - Parameters
 - Export to CSV files
 - Import to Cytoscape for visualization

Method 3: Bayesian Network – conditional dependence



$$P(A, B) = P(A | B) * P(B) = P(B | A) * P(A)$$

$$\frac{P(A | B) * P(B)}{P(B | A) * P(A)}$$

Bayes rule:

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

Bayesian network – the conditional probability

If A and B are independent
A= winter (yes 25%, no 75%)
B= happy (yes 50%, no 50%)



$$\begin{aligned}P(A) &= 25\% \\P(B) &= 50\% \\P(\text{winter, happy}) &= \\P(A,B) &= 25\% * 50\% = 12.5\%\end{aligned}$$

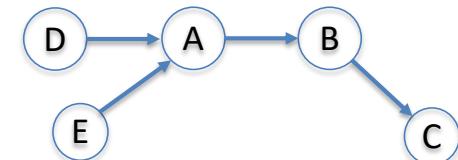
Correct

How about if B dependent on A:
A= winter (yes 25% or no 75%)
B= cold weather (overall yes 25%, no 75%)


$$\begin{aligned}P(A) &= 25\% \\P(B|A) &= 100\% \text{ on winter and } 0\% \text{ on other seasons.} \\&\text{Overall } 25\% \\&\text{The conditional probability of winter \& cold is} \\P(A=\text{winter}, B=\text{cold}) &= P(A)P(B) = 25\% * 25\% = 6.25\% \\&\text{wrong!}\end{aligned}$$

$$\begin{aligned}P(\text{winter, cold}) &= \\P(A=\text{winter}; B=\text{cold}) &= P(A)P(B|A) = 25\% * 100\% = 25\%\end{aligned}$$

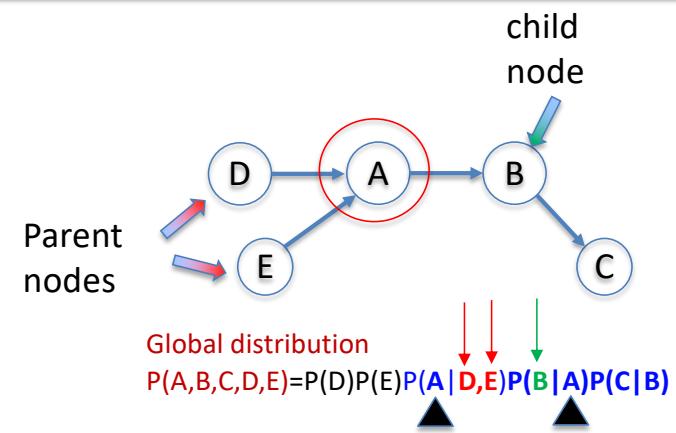
A= winter (yes or no 50% each)
B= cold weather (yes or no 100%)
C= wear sweater (yes or no, 50%)
D= month of a year
E= location on the earth



Global distribution
 $P(A,B,C,D,E) = P(D)P(E)P(A|D,E)P(B|A)P(C|B)$

Parent nodes and child node

- To a given node
 - Conditional nodes (reasons or contributing factors) are called parents.
 - The consequent nodes are child nodes.



Assign directions: Induction of Causality algorithm

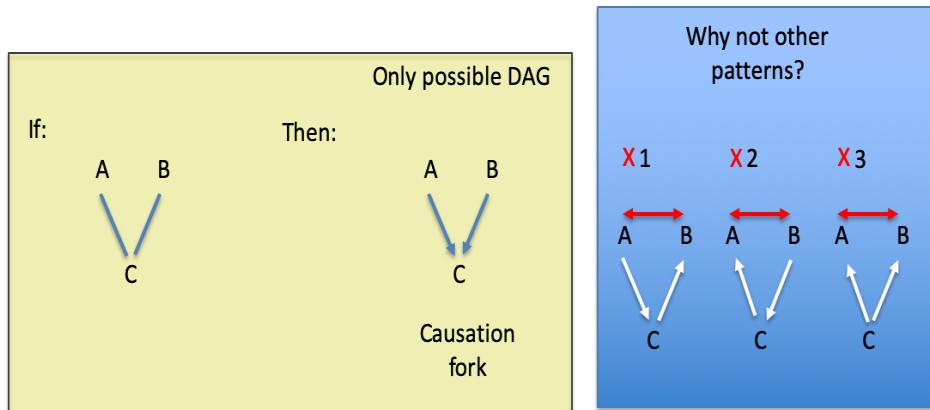
The Induction of Causality Algorithm:

If,

1. A, B are not connected through any other connection.
2. A and B have a common connection to C

Then:

arrows point from A and B to C



Learn the network structure and parameters

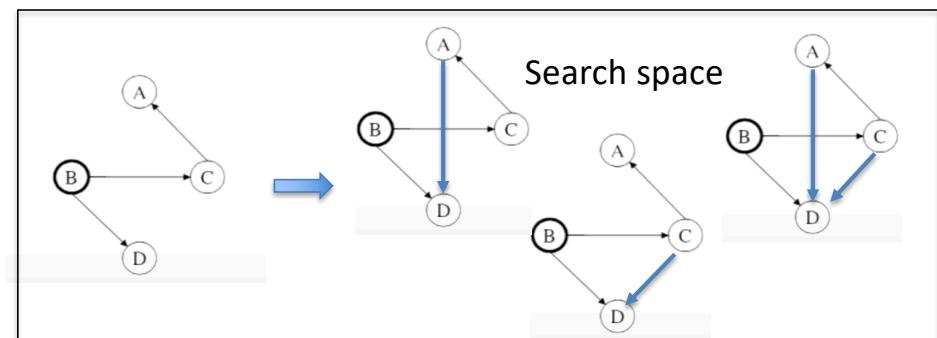
Divide the Bayes net into two components:

- Θ : the conditional probabilities
- G : the structure
 - Can be decomposed to each node with parents (X_i and Π_{xi}).

| The problem | Step 1 | Step 2 |
|--|--------|--------|
| $\underbrace{P(M \mathcal{D}) = P(\mathcal{G}, \Theta \mathcal{D})}_{\text{learning}} = \underbrace{P(\mathcal{G} \mathcal{D})}_{\text{structure learning}} \cdot \underbrace{P(\Theta \mathcal{G}, \mathcal{D})}_{\text{parameter learning}}$ | | |
| Probability of a model (M) given the data set (D). | | |

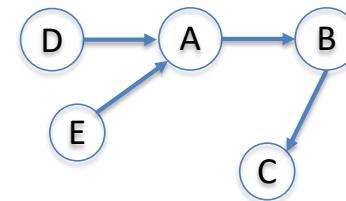
$$\text{Score}(G|\mathcal{D}) \stackrel{\text{def}}{=} \text{LL}(G|\mathcal{D}) - \underline{\psi(N) \cdot \|G\|}$$

BIC score = Bayesian Information Criterion



Curse of the dimensionality

- Complexity increases exponentially with respect to:
 - Number of bins each node can take.
 - Number of parent nodes that a node can have.
- Solutions
 - Breaking a huge network into conditionally independent units



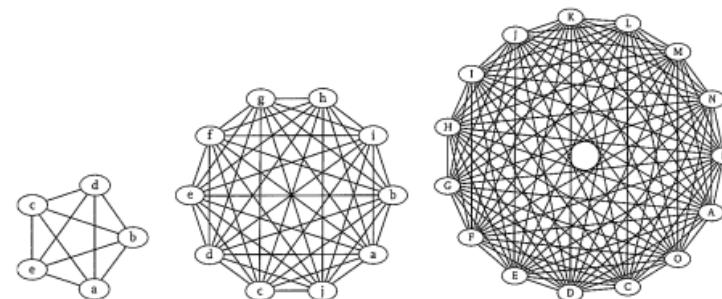
Global distribution

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

If each sample has 5 bins

Values to fully enumerate the distribution table: $5^5 = 1875$

Enumerate the Bayesian network: $5 + 5 + 125 + 25 + 25 = 185$



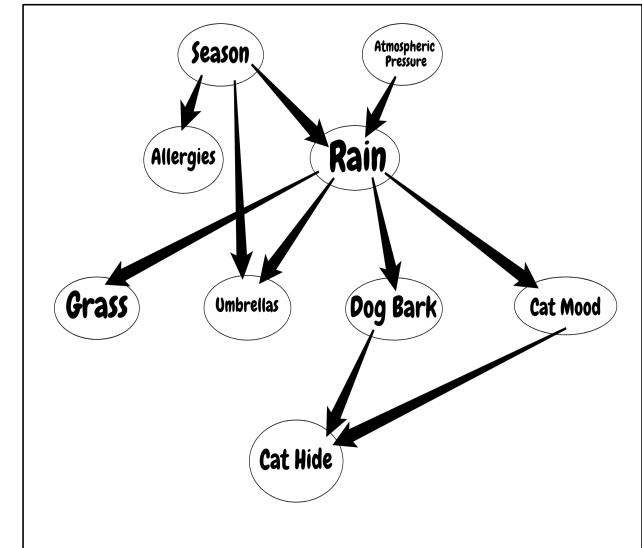
Discussion on Bayesian Network

Pros

- Easy interpretation

Cons

- Requires large computation
- Decomposition of Bayesian seems not efficient



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



Demonstration on bnlearn in R

- bnlearn in R
 - Demonstrate the import of data
 - Data structure
 - Output structure
 - Demonstrate how the graph are calculated
 - Import in to Cytoscape for visualization



Thanks!

- Questions?
 - Googledoc, <https://tinyurl.com/zhu-GRN>
 - GitHub: https://github.com/niaid/Gene_Regulatory_Networks
 - My email: zhuy16@nih.gov
 - BCBB email: bioinformatics@niaid.nih.gov

Key references

- Intuition understanding about entropy,
 - <https://www.youtube.com/watch?v=2s3aJfRr9gE>
- Calculation of mutual information using a concrete data sample.
 - <https://www.youtube.com/watch?v=3iplfAfGzI>
- CS262a: Learning and reasoning with Bayesian Networks Adnan Darwiche
 - 11a. Learning Parameters: Complete Data (Chapter 17)
 - <https://www.youtube.com/watch?v=gRVg0lZgLug>
 - 11b. Learning Parameters: Incomplete Data (Chapter 17)
 - <https://www.youtube.com/watch?v=NDoHheP2ww4>
 - 12a. Learning Network Structure I (Chapter 17)
 - https://www.youtube.com/watch?v=RV2lInyq_bI
 - 12b. Learning Network Structure II (Chapter 17)
 - <https://www.youtube.com/watch?v=o-urnZRNnbY>
- Bnlearn <https://www.youtube.com/watch?v=4JkddqxGrO0>



Other resources

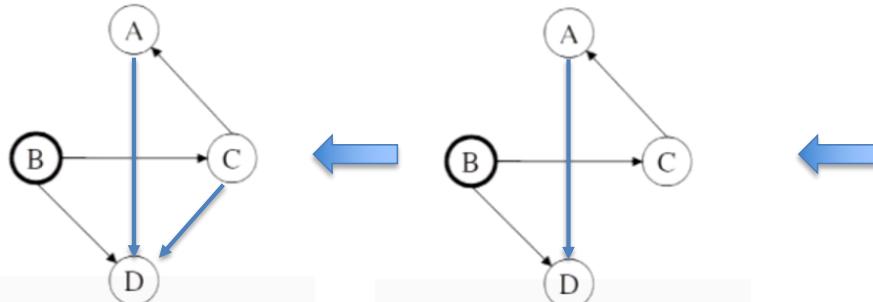
- GraphPlot, <http://juliagraphs.github.io/GraphPlot.jl/>
- Gamma distribution, <https://towardsdatascience.com/gamma-distribution-intuition-derivation-and-examples-55f407423840>
- R packages
 - `install.packages("igraph")`
 - `install.packages("bnlearn")`
 - `BiocManager::install("Rgraphviz")`
- iGraph for network analysis
<https://www.bioss.ac.uk/people/helen/igraphIntro.html>
- <https://www.r-bloggers.com/interactive-network-visualization-with-r/>

Learning the skeleton of a graph

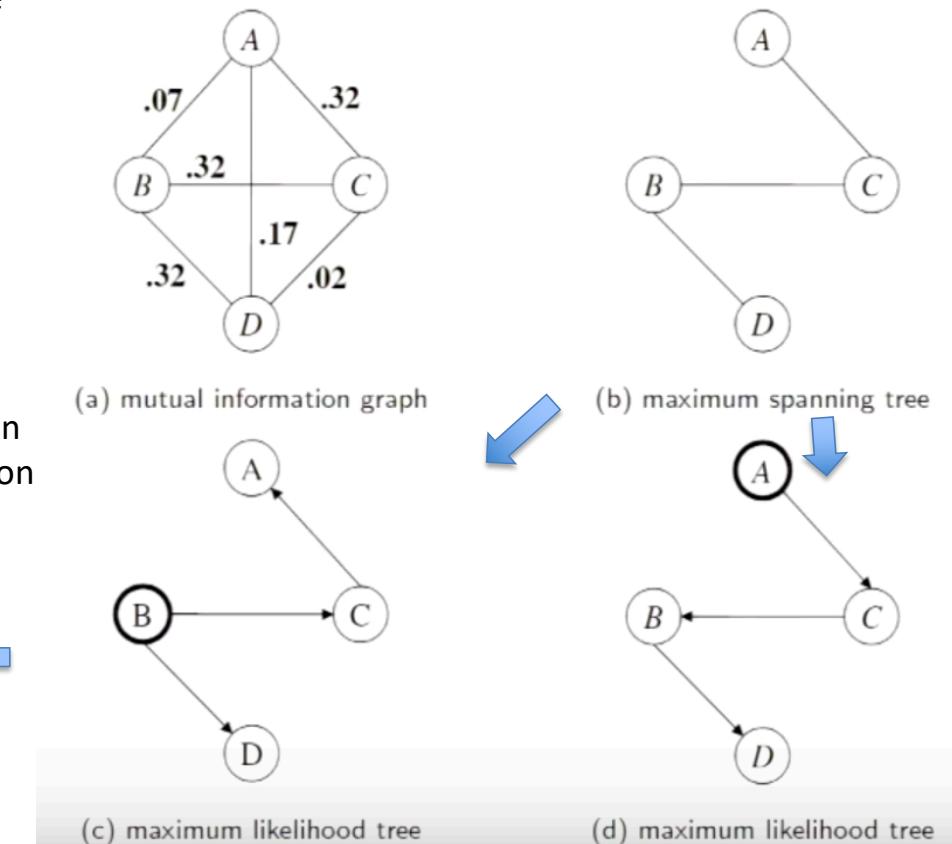
- Constructing a tree structure (each node can only have one parent node).
- Step 1, learning the Maximum Spanning Tree.
- Add direction through starting from a randomly node.
- The resulting directed graph will be used as seed for other algorithms.
- Optimization of a graph, a commonly used approach constrained on a penalty on the complexity ($||G||$) of the graph.

$$\text{Score}(G|\mathcal{D}) \stackrel{\text{def}}{=} \text{LL}(G|\mathcal{D}) - \underline{\psi(N) \cdot ||G||}$$

BIC score = Bayesian Information Criterion

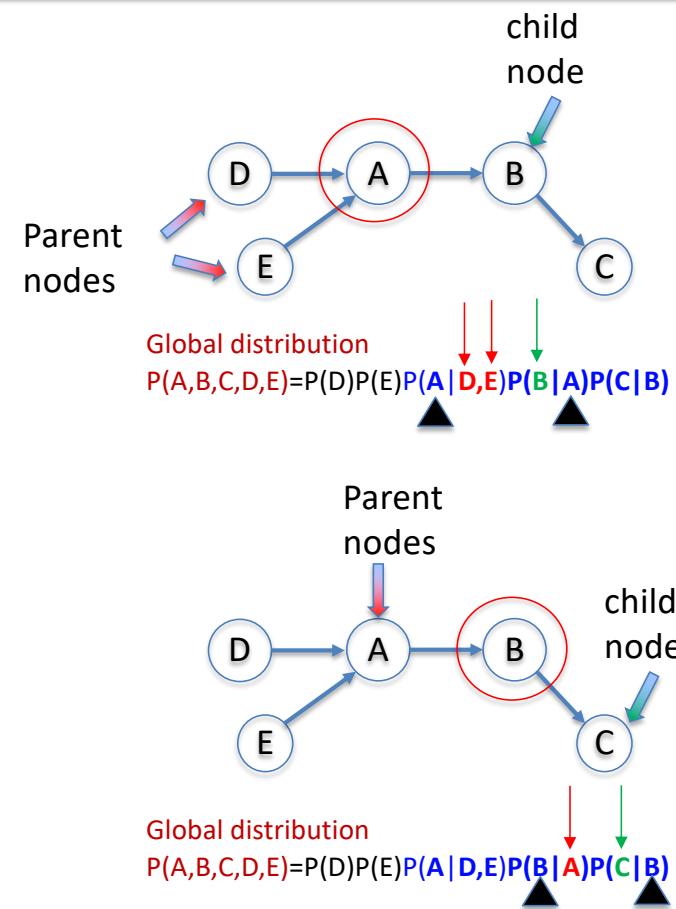


<https://www.youtube.com/watch?v=o-urnZRNnbY>



Parent nodes and child node

- To a given node
 - Conditional nodes (reasons or contributing factors) are called parents.
 - The consequent nodes are child nodes.



Objective function for optimizing structures

$$\underbrace{P(M | \mathcal{D}) = P(\mathcal{G}, \Theta | \mathcal{D})}_{\text{learning}} = \underbrace{P(\mathcal{G} | \mathcal{D})}_{\text{structure learning}} \cdot \underbrace{P(\Theta | \mathcal{G}, \mathcal{D})}_{\text{parameter learning}}$$

Probability of a model (M) given the data set (D).

$$MI_{\mathcal{D}}(X, U) \stackrel{\text{def}}{=} \sum_{x,u} \Pr_{\mathcal{D}}(x, u) \log \frac{\Pr_{\mathcal{D}}(x, u)}{\Pr_{\mathcal{D}}(x)\Pr_{\mathcal{D}}(u)}$$

$$\text{Score}(G|\mathcal{D}) \stackrel{\text{def}}{=} LL(G|\mathcal{D}) - \psi(N) \cdot ||G||$$

Degree of a network

$$||G|| \stackrel{\text{def}}{=} \sum_{i=1}^n ||X_i \mathbf{U}_i||$$

N: the number of nodes in the network

Sum of the number of possible states each node can possibly take

$$||X_i \mathbf{U}_i|| \stackrel{\text{def}}{=} (X_i^{\#} - 1) \mathbf{U}_i^{\#}$$

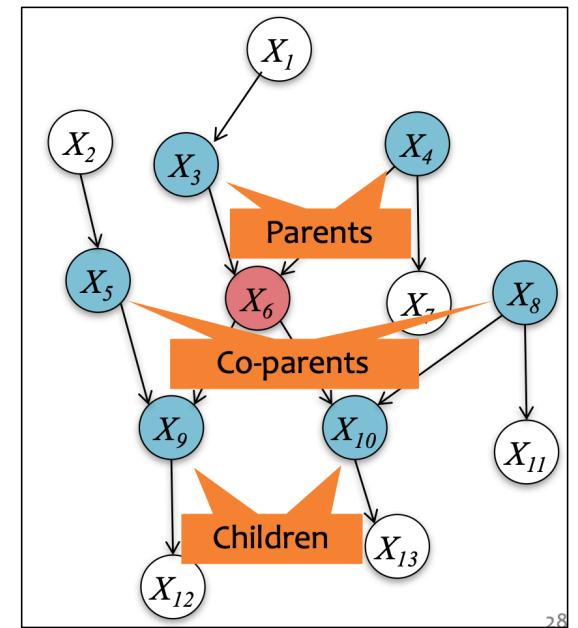


How to set up a tunnel to server

- `install.packages('IRkernel')` # Don't forget step 2/2!
- `IRkernel::installspec()`
- using Pkg
- `Pkg.add("IJulia")`
- `user@local_machine$ ssh -N -L localhost:8888:localhost:8888
user@remote_mahcine`
- Paste this to the url to access the Julia – "localhost:1234"

Computation: how to break up a huge network to small local problems

- Optimization of the local variables
 - For computation purpose, you need to break a huge network into smaller pieces.
 - for a node in a graphical model, **Markov blanket** contains all the variables needed to do approximation, that shield the node from the rest of the network.



Ref: Learning Bayesian network structure using Markov blanket decomposition, Bui 2012

Assign directions: Induction of Causality algorithm

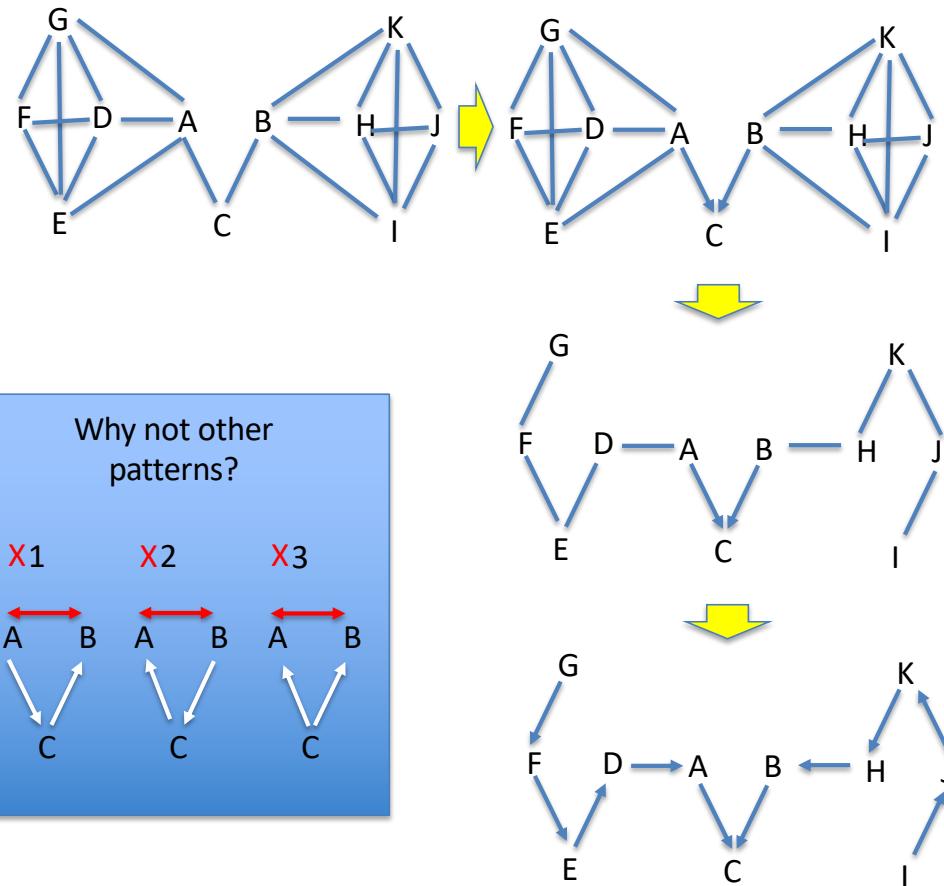
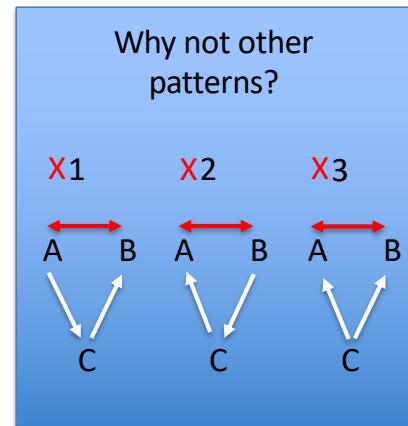
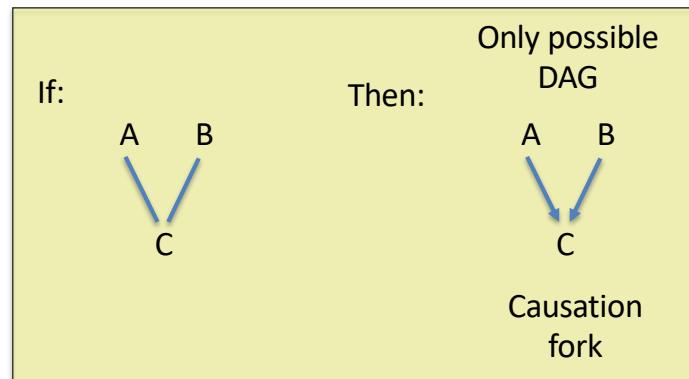
The Induction of Causality Algorithm:

If,

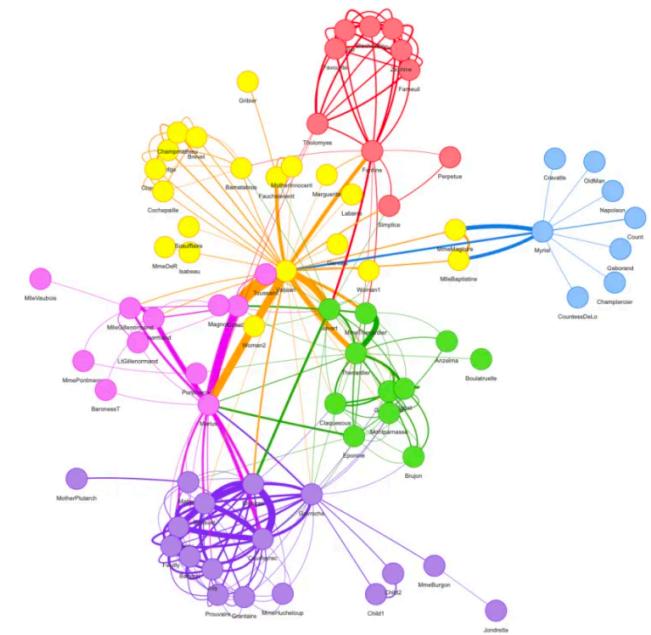
1. A, B are not connected through any other connection.
2. A and B have a common connection to C

Then:

arrows point from A and B to C



- > tpm=read.table("S_GSE71485_Single TPM.txt",row.names=1,header=T)
- > Tpm=tpm[rowMeans(tpm)>100,]
- > write.table(Tpm,"S_GSE71485_Single TPM.txt")



Parameters in Bayesian network

$$P(A | B) = \frac{\text{likelihood prior}}{\text{posterior Evidence}}$$
$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}$$

$$P(\text{cold} | \text{winter}) = \frac{\text{posterior likelihood prior}}{\text{Evidence}} = \frac{P(\text{winter} | \text{cold})P(\text{cold})}{P(\text{winter})} = \frac{P(\text{winter} | \text{cold})P(\text{cold})}{P(\text{winter})} = \frac{100\% * 25\%}{25\%} = 100\%$$

Infer the parameters in Bayesian network

Expectation Maximization

- Assign a set of random parameters θ^0 .
- Based on evidence samples, calculate the distribution of each sample.
- Go through all the samples.
- Construct the averaged distribution.
- Based on the averaged distribution, recalculate the set of parameters θ^1 .
- Repeat the previous step until θ^{K+1} and θ^K converges.

Using log-likelihood as a objective function to maximize

- Construct Log Likelihood formula.
- Use classical methods such as gradient decent to optimize the parameter with a assigned learning rate, to iterate and reach a local maximum.

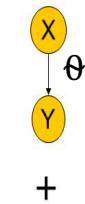
For complete data

$$\theta^* = \underset{\theta}{\operatorname{argmax}} L(\theta | \mathcal{D})$$

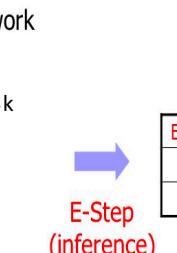
$$\text{iff } \theta_{x|\mathbf{u}}^* = \Pr_{\mathcal{D}}(x|\mathbf{u})$$

For incomplete data

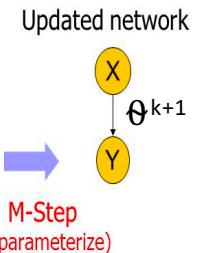
Initial network



+

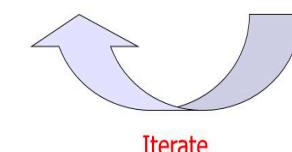


| Expected counts |
|-----------------|
| N(X) |
| N(X,Y) |



Training data

| X | Y |
|----------------|----------------|
| ? | y ⁰ |
| x ⁰ | y ¹ |
| ? | y ⁰ |



Iterate