Some Algorithms for Inferring Gene Regulatory Networks Using Gene Expression data

Changiz Eslahchi

Faculty of Mathematical Sciences, Department of Computer Science, Shahid Beheshti University, G.C., Tehran, Iran.

March, 2015
1 Challenge

2 Bayesian Network

3 Algorithms

4 Conclusion
Gene Expression

- DNA → transcription → mRNA → translation → Protein
- CCTGAGCCAACTATTGATGAA → CCUGAGCCAAACUAAUUGAUGAA → PEPTIDE
Inferring gene regulatory networks (GRNs) is a major issue in systems biology, which explicitly characterizes regulatory processes in the cell.
Gene Regulatory Network

- Differential Equation (Novak et al, 1998)
- Stochastic Petri Net (Goss et al, 1999)
- Boolean Network (Liang et al, 1998)
- Regression Method (Gardner et al, 2003)
- Bayesian Network (Friedman, 2004)
- Linear Programming (Wang et al, 2006)
- And so many others!
Independence

Recall the following equivalent characterizations of independence, $X \perp Y$:

$$P(X \mid Y) = P(X)$$

$$P(Y \mid X) = P(Y)$$

$$P(X, Y) = P(X)P(Y)$$

Intuitively, if $X \perp Y$ then knowledge of $X$ provides no information about $Y$. 
Conditional Independence

Characterizations of conditional independence, $X \independent Y \mid Z$:

$$P(X \mid Y, Z) = P(X \mid Z)$$

$$P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z),$$


Intuitively, $X \independent Y \mid Z$, then if $Z$ is known, $X$ provides no further knowledge of $Y$, and $Y$ provides no further knowledge of $X$. 
Entropy function is a suitable tool for measuring the average uncertainty of a variable $X$.

$$H(X) = E[I(X)] = - \sum_{i=1}^{n} P(X = x_i) \log P(X = x_i)$$

- $H(X, Y) = - \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} \log p_{ij}$
- $H(Y | X = i) = - \sum_{j=1}^{m} p_i(j) \log p_i(j)$
- $H(Y | X) = - \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} \log p_i(j)$
- $H(Y | X) = H(X, Y) - H(X)$,
- $H(X | Y) = H(X, Y) - H(Y)$. 

Ch. Eslahchi (Shahid Beheshti University)
Mutual Information

- \( MI(X, Y) = \sum_Y \sum_X P(x, y) \log \frac{P(x,y)}{P(x)P(y)} \)
- \( MI(X, Y) = H(X) + H(Y) - H(X, Y), \)
- \( MI(X, Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X). \)
- Conditional Mutual Information between \( X \) and \( Y \) given \( Z \) is determined by:
  \[
  CMI(X, Y \mid Z) = \sum_Z P(Z) \sum_{X,Y} P(X,Y \mid Z) \log_2 \frac{P(X,Y \mid Z)}{P(X \mid Z)P(Y \mid Z)}.
  \]
The graph is acyclic because we do not permit directed cycles: 
\[ A \rightarrow \cdots \rightarrow A \]
If there is an edge \( A \rightarrow B \) then \( A \) is said to be a parent of \( B \), and \( B \) is a child of \( A \).
Two vertices joined by an edge are said to be adjacent.
The set of parents of a vertex \( A \), is denoted \( \text{pa}(A) \). Example:
\[ \text{pa}(D) = \{B, C\} \]
We use \( X \) to denote the set of all vertices in the DAG.
Example

\[ Pa(B) = \{ A \} \]
\[ an(E) = \{ A, B, C, D \}, \quad an(D) = \{ A, B, C \}, \]
\[ de(A) = \{ B, C, D, E \}, \quad de(B) = \{ D, E \}. \]
Complete DAGs.

A DAG is complete if every pair of vertices are adjacent.
The term "Bayesian networks" was coined by Judea Pearl in 1985. A Bayesian network (BN) is a graphical representation of a joint probability distribution that includes two components[20, 13, 26]

- First, a directed acyclic graph (DAG) $G = (X, E_G)$ where $X = \{X_1, \ldots, X_n\}$

- The second component is a set of numerical parameters, which usually represent conditional probability distributions.
The term "Bayesian networks" was coined by Judea Pearl in 1985. A Bayesian network (BN) is a graphical representation of a joint probability distribution that includes two components [20, 13, 26]

- First, a directed acyclic graph (DAG) $G = (X, E_G)$ where $X = \{X_1, \ldots, X_n\}$
- The second component is a set of numerical parameters, which usually represent conditional probability distributions.
Definitions and Theorems

Definition

A DAG $G$ has a Markov property if each variable is conditionally independent of its non-descendants given its parent variables\[20, 13, 26\]:

For example based on this definition in the following figure we have:

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

$P(X_1, X_2, \ldots, X_n) = \prod_{X_i \in X} P(X_i | Pa_{X_i})$. 

Ch. Eslahchi  (Shahid Beheshti University)
A path $p$ from $X$ to $Y$ in $G$ is said to be blocked by a set of variables $Z$ if and only if [20, 13, 26]:

- $p$ contains a chain $X \rightarrow K \rightarrow Y$ or a fork $X \leftarrow K \rightarrow Y$ such that $K \in Z$, or
- $p$ contains a collider $X \rightarrow K \leftarrow Y$ such that $K$ and all the descendants of $K$ are not in $Z$.

A set $Z$ is said to d-separate $X$ from $Y$ in $G$ if and only if $Z$ blocks every path from $X$ to $Y$. 
A path $p$ between vertices $X$ and $Y$ consists of a sequence of distinct vertices that are adjacent. For example:

$$X \rightarrow A \leftarrow B \leftarrow C \rightarrow D \rightarrow Y$$

A non-endpoint vertex $V$ on a path (between $X$ and $Y$) is said to be a collider if the path takes the form $X \cdots \rightarrow V \leftarrow \cdots Y$. Non-endpoints that are not colliders are called non-colliders:

$$X \cdots \leftarrow V \rightarrow \cdots Y, X \cdots \rightarrow V \rightarrow \cdots Y, X \cdots \leftarrow V \leftarrow \cdots Y$$
Subtlety regarding colliders

A vertex is a collider or non-collider with respect to a given path. A vertex may be a collider on one path and a non-collider on another (even between the same endpoints).

Here $Z$ is a collider on the path $X \rightarrow Z \leftarrow H \rightarrow Y$, but $Z$ is a non-collider on the path $X \rightarrow Z \rightarrow Y$.

Grammar: $V$ is a collider with respect to the path.
So far we have considered separation between single variables $X$, $Y$ given a set $Z$.

We extend this to sets of variables as follows: A set $X$ is d-separated from a set $Y$ given $Z$ iff or all $X \in X, Y \in Y$, $X$ and $Y$ are d-separated given $Z$.

We use $S_{XY} = Z$ to denote $X$ is d-separated from $Y$ given $Z$. 

---

Ch. Eslahchi (Shahid Beheshti University)
Learning Bayesian Network contains Structure learning and Parameter Learning.

**Structure learning**

**Parameter Learning**
Structure learning

- **Constraint Learning Methods:** Constraint-based methods apply information about Conditional Independent (CI) test to determine dependency between vertices.

- **Scored and Search Methods:** The score-based searching methods produce a series of candidate networks, calculate a score for each candidate and return a candidate with the highest score.

- **Hybrid Methods:** The hybrid method is a combination of these two methods.
Structure learning

- **Constraint Learning Methods:**
  SGS Algorithm [29], FCI Algorithm [30], TIPDA Algorithm [10], PC Algorithm [29], IC Algorithm [27], QFCI Algorithm [3], GPC Algorithm [28] and *PCA – CMI* Algorithm [34].

- **Scored and Search Methods:**
  AIC Algorithm [2], BDeu Algorithm [6], K2 Algorithm [12], BD Algorithm [18], BIC Algorithm [19], MIT Algorithm [13] and Algorithm K2GA [15].

- **Hybrid Methods:** Suzuki’s Algorithm [7], HGC Algorithm [9] and *MMHC* Algorithm [32].
The Path Consistency (PC) algorithm is introduced by Spirtes et al. in 1993 as a constraint-based method to infer BNs.
PC Algorithm

Algorithm [PC]
1. Start with a complete undirected graph $S_{-1}$
2. $i = 0$,
   i denotes the order of algorithm, It means the separating sets of size $i$
3. Repeat
4. For each $X \in \mathbf{X}$
5. For each $Y \in ADJ(X)$
6. Determine if there is $M \subseteq ADJ(X)$ Or $ADJ(Y)$ with $|M| = i$ such that $X$ and $Y$ given $M$ are independent
7. If this set exists
8. Remove the edge between $X$ and $Y$ from $S_{i-1}$
9. $i = i + 1$
10. Until $\max\{|ADJ(X)|, |ADJ(Y)|\} \leq i + 1$. 
Example of PC Algorithm; $i = 0$

There is no pair of variables d-separated given $\emptyset$, so Graph unchanged.
Example of PC Algorithm; \( i = 1 \)

\[ S_{BC} = \{ A \} \]

\[ S_{AE} = \{ D \} \]

\[ S_{BE} = \{ D \} \]

\[ S_{CE} = \{ D \} \]
Example of PC Algorithm; \( i = 2 \)

\[ S_{AD} = \{B, C\} \]
Orientation Rules

1. If we have $X \rightarrow Z \rightarrow Y$ but no edge between $X$ and $Y$, and $Z \notin S_{XY}$ then orient as $X \rightarrow Z \leftarrow Y$.

2. Apply the following rules:
   (a) If $X$ and $Z$ are not adjacent by $X \rightarrow Y \rightarrow Z$ then orient as $Y \rightarrow Z$.
   (b) If $X \rightarrow Y \rightarrow Z$ and $X \rightarrow Z$ then orient as $X \rightarrow Z$.
   (c) If $X$ and $Z$ are not adjacent, but $X \rightarrow W \rightarrow Z$ and $X \rightarrow Y \leftarrow Z$ and $W \rightarrow Y$ then orient as $W \rightarrow Y$. 

Example of PC Algorithm

\[ D \notin S_{BC} = \{A\} \]
Scoring Function: Denotes the degree of fitness of candidate structure to data set

Search Methods

\[ G^* = \arg\max_{G \in F(n)} g(G : D), \]

in which \( g(G : D) \) denotes the degree of fitness of candidate \( G \) to data set and \( F(n) \) indicates all the possible DAGs defined on \( X \). The challenging part of search procedure is that the size of the space of all structures, \( f(n) \), is super-exponential in the number of variables[26],
There are many scoring functions to measure the degree of fitness of a DAG $G$ to a data set. These are generally classified as:

**Bayesian scoring functions** \([6, 18, 22]\)

**Information theory-based scores** \([11, 24, 4, 16, 13]\).
Search Methods

The definition of the search space determines the definition of the search operators used to move from one structure to another. In turn, these operators determine the neighborhood of a DAG, namely the DAGs that can be reached in one step from the current DAG. Typically, the operators consist of:

- **arc addition**: insert a single arc between two nonadjacent nodes.
- **arc deletion**: remove a single arc between two nodes.
- **arc reversal**: reverse the direction of a single arc.
In what follows we let $op(S, A)$ represent the result of performing the arc operation $A$ on the structure $S$, i.e., $op(S, A)$ is a DAG that differs from $S$ with respect to one arc.

We define

$$\Delta(X_i \rightarrow X_j) = \text{score}(X_j, pa(X_j) \cup X_i, D) - \text{score}(X_j, pa(X_j), D).$$
Search Methods

- The Greedy search Algorithm

Algorithm [Greedy search]
1. Let $S$ be an initial structure.
2. Repeat
   a) Calculate $\Delta(A)$ for each legal arc operation $A$
   Let $\Delta^*(A) = \max_A \Delta(A)$ and $A^* = \arg\max_A \Delta(A)$.
   b) If $\Delta^* > 0$, then
      Set $S = op(S, A^*)$.
3. Until $\Delta^* < 0$. 
Scored and Search Methods

- The Greedy search Algorithm

\[ \hat{G} = \arg \max_G \text{Score}(G) \]

- Heuristic search:
  - Greedy local search
  - Best-first search
  - Simulated annealing

NP-hard optimization
MIT Score

- MIT Score is one of the famous Information theory-based scores which contains two terms:
  Mutual information and Penalization Term.

\[
g_{MIT}(G, D) = \sum_{i=1, Pa_G(X_i) \neq \emptyset}^{n} (2NMI_D(X_i, Pa_B(X_i)) - \max_{\sigma_i} \sum_{j=1}^{s_i} \chi_{\alpha, l_i, \sigma_i(j)}).
\]  

(1)
Algorithms

In this work we introduce three algorithms for learning GRNs.


Information Theory

Let $\mathbf{X} = (X_1, ..., X_n)^T$ be an $n$-dimensional Gaussian vector with mean $\mathbf{\mu} = (\mu_1, ..., \mu_n)^T$ and covariance matrix $C(\mathbf{X}) = E(\mathbf{X} - \mathbf{\mu})(\mathbf{X} - \mathbf{\mu})^T$, i.e. $\mathbf{X} \sim N(\mathbf{\mu}, C(\mathbf{X}))$.

Then MI for two continuous variables $X$ and $Y$ can be easily calculated using the following formula.

$$MI(X, Y) = \frac{1}{2} \log \frac{\sigma_X^2 \cdot \sigma_Y^2}{\sigma_{XY}},$$ \hspace{1cm} (2)

Conditional Mutual Information for two continuous variables $X$ and $Y$ given $Z$ is determined by:

$$CMI(X, Y | Z) = \frac{1}{2} \log \frac{\det(C(X, Z)) \cdot \det(C(Y, Z))}{\det(C(Z)) \cdot \det(C(X, Y, Z))},$$ \hspace{1cm} (3)
PC Algorithm based on CMI test (PCA-CMI)

Algorithm [PCA-CMI]
1. Start with a complete undirected graph $S_{-1}$
2. $i = 0$
3. Repeat
4. For each $X \in \mathbf{X}$
5. For each $Y \in ADJ(X)$
6. Determine if there is $\mathbf{M} \subseteq V_{XY} = ADJ(X) \cap ADJ(Y)$ with $|\mathbf{M}| = i$
such that $X$ and $Y$ given $\mathbf{M}$ are independent
7. If this set exists
8. Remove the edge between $X$ and $Y$ from $S_{i-1}$
9. $i = i + 1$
10. Until $i \leq |V_{XY}|$
PC Algorithm based on CMI test (PCA-CMI)
Zero order of the Improvement of PC Algorithm based on CMI test

Algorithm [IPCA-CMI order 0]

1. Start with a complete undirected graph $S_{-1}$.
2. Repeat
3. For each $X \in \mathbf{X}$
4. For each $Y \in \text{ADJ}(X)$
5. If $X$ and $Y$ are independent based on the measure of MI
6. Remove the edge between $X$ and $Y$ from $S_{-1}$
7. The MIT score was utilized in the HC algorithm to construct $G_0$. 
The IPCA-CMI for $i > 0$

$A_{qZ}$ for $1 \leq q \leq 4$ are defined as follows:

$A_{1Z} = \{W|X \rightarrow Z \rightarrow W\}$, $A_{2Z} = \{W|X \leftarrow Z \leftarrow W\}$,

$A_{3Z} = \{W|X \leftarrow Z \rightarrow W\}$, $A_{4Z} = \{W|X \rightarrow Z \leftarrow W\}$.

$Weight_X(Z) = |A_{1Z}| + |A_{2Z}| + |A_{3Z}| - |A_{4Z}|$,

Let $R_{XY}$ be defined by:

$R_{XY} = \{Z|Weight_X(Z) \geq k \text{ or } Weight_Y(Z) \geq k, \forall Z \in \{ADJ(X) \cup ADJ(Y)\} \setminus \{X, Y\}\}$,
The IPCA-CMI for $i > 0$

Algorithm [The IPCA-CMI for $i > 0$]

1. Start with $G_0$
2. $i = 1$
3. Repeat
4. For each $X \in \mathbf{X}$
5. For each $Y \in ADJ(X)$
6. Test whether $\exists \mathbf{H} \subseteq R_{X Y}$ with $|\mathbf{H}| = i$ such that $X \parallel Y | \mathbf{H}$.
7. If this set exists
8. Remove the edge between $X$ and $Y$ from $G_{i-1}$
9. The MIT score was utilized in the HC algorithm to direct the structure.
10. For each $Z \in \{ADJ(X) \cup ADJ(Y)\} \setminus \{X, Y\}$
11. $Weight_X(Z) = |A_{1Z}| + |A_{2Z}| + |A_{3Z}| - |A_{4Z}|$
12. $R_{X Y} = \{Z | Weight_X(Z) \geq k \text{ or } Weight_Y(Z) \geq k, \forall Z \in \{ADJ(X) \cup ADJ(Y)\} \setminus \{X, Y\}\}$
13. $i = i+1$
14. Until $i \leq |R_{X Y}|$
The **Dialogue for Reverse Engineering Assessments and Methods** (DREAM) project was introduced in 2006. DREAM is a data set that allows researchers to evaluate the performance of their approaches in predicting biological networks.

The goal of the in silico network challenge is to reverse engineer gene regulation networks from simulated steady-state and time-series data. The true network structure of biological gene networks is in general unknown or only partly known, but the structures of in silico networks are known and so predictions can be validated.
You can download Data set from http://www.the-dream-project.org/challenges
Data Set

DREAM3 In Silico Size 10 and 50.
Evaluation

To assess the validity of algorithms, the True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN) values for proposed algorithms are computed. In addition, known measures such as Accuracy (ACC), False Discovery Rate (FDR), Positive Predictive Value (PPV), F-score measure (F-measure), Matthews Correlation Coefficient (MCC) and True Positive Rate (TPR) are considered. These measures are defined by:

\[
TPR = \frac{TP}{TP + FN}, \quad SPC = \frac{TN}{FP + TN}
\]

\[
PPV = \frac{TP}{TP + FP}, \quad NPV = \frac{TN}{TN + FN}
\]

\[
ACC = \frac{TP + TN}{TP + FP + TN + FN}, \quad FPR = \frac{FP}{FP + TN},
\]
Evaluation

\[
FDR = \frac{FP}{FP + TP}, \quad F = 2 \frac{PPV \times TPR}{PPV + TPR},
\]

\[
MCC = \frac{TP \times TN - FP \times FN}{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}.
\]
Evaluation

To make an evaluation of the performance of algorithms, in the present study, these algorithms are compared by several other popular Inference methods those which are given in Table. In this table the approaches used by each algorithm is also mentioned.

**Table:** Gene network inference methods which applied for comparison

<table>
<thead>
<tr>
<th>Method</th>
<th>Approach</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>LASSO</td>
<td>Regression</td>
<td>Royal Statistical Society[31]</td>
</tr>
<tr>
<td>LP</td>
<td>Linear Programing</td>
<td>Bioinformatics[33]</td>
</tr>
<tr>
<td>PCC</td>
<td>PCA based on partial CC</td>
<td>Machine Learning Research[21]</td>
</tr>
<tr>
<td>MI</td>
<td>Mutual Information</td>
<td>Reverse engineering cellular networks[25]</td>
</tr>
<tr>
<td>PCCMI</td>
<td>PCA based on CMI</td>
<td>Bioinformatics[34]</td>
</tr>
</tbody>
</table>
Result IPCA-CMI

A

True Network

B

First-order Network inferred by PCA-CMI

C

First-order Network inferred by IPCA-CMI
## Algorithms

### Result IPCA-CMI

**Table:** The result of Simulated and Real data sets in order 0

<table>
<thead>
<tr>
<th>Network</th>
<th>TP</th>
<th>FP</th>
<th>ACC</th>
<th>FPR</th>
<th>FDR</th>
<th>PPV</th>
<th>F</th>
<th>MCC</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DREAM10</td>
<td>9</td>
<td>1</td>
<td>0.95</td>
<td>0.02</td>
<td>0.10</td>
<td>0.9</td>
<td>0.90</td>
<td>0.87</td>
<td>0.90</td>
</tr>
<tr>
<td>DREAM50</td>
<td>36</td>
<td>54</td>
<td>0.92</td>
<td>0.05</td>
<td>0.6</td>
<td>0.4</td>
<td>0.43</td>
<td>0.39</td>
<td>0.46</td>
</tr>
<tr>
<td>DREAM100</td>
<td>70</td>
<td>58</td>
<td>0.96</td>
<td>0.01</td>
<td>0.45</td>
<td>0.55</td>
<td>0.47</td>
<td>0.46</td>
<td>0.42</td>
</tr>
<tr>
<td>SOS</td>
<td>18</td>
<td>4</td>
<td>0.72</td>
<td>0.33</td>
<td>0.18</td>
<td>0.82</td>
<td>0.78</td>
<td>0.40</td>
<td>0.75</td>
</tr>
</tbody>
</table>
Table: The result of gene expression data set DREAM3 Challenge with 10 genes and sample number 10

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>TP</th>
<th>FP</th>
<th>ACC</th>
<th>FPR</th>
<th>FDR</th>
<th>PPV</th>
<th>F</th>
<th>MCC</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA1</td>
<td>7</td>
<td>1</td>
<td>0.91</td>
<td>0.03</td>
<td>0.13</td>
<td>0.87</td>
<td>0.78</td>
<td>0.73</td>
<td>0.7</td>
</tr>
<tr>
<td>IPCA1</td>
<td>8.8</td>
<td>0</td>
<td>0.98</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.94</td>
<td>0.93</td>
</tr>
</tbody>
</table>
### Result IPCA-CMI

Table: The result of gene expression data set DREAM3 Challenge with 50 genes and sample number 50

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>TP</th>
<th>FP</th>
<th>ACC</th>
<th>FPR</th>
<th>FDR</th>
<th>PPV</th>
<th>F</th>
<th>MCC</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA1</td>
<td>24</td>
<td>23</td>
<td>0.93</td>
<td>0.02</td>
<td>0.49</td>
<td>0.51</td>
<td>0.39</td>
<td>0.37</td>
<td>0.31</td>
</tr>
<tr>
<td>PCA2</td>
<td>22</td>
<td>21</td>
<td>0.93</td>
<td>0.02</td>
<td>0.49</td>
<td>0.51</td>
<td>0.37</td>
<td>0.35</td>
<td>0.29</td>
</tr>
<tr>
<td>IPCA1</td>
<td>28</td>
<td>26.5</td>
<td>0.94</td>
<td>0.02</td>
<td>0.48</td>
<td>0.51</td>
<td>0.43</td>
<td>0.4</td>
<td>0.36</td>
</tr>
<tr>
<td>IPCA2</td>
<td>22.9</td>
<td>11.78</td>
<td>0.95</td>
<td>0.01</td>
<td>0.48</td>
<td>0.52</td>
<td>0.38</td>
<td>0.42</td>
<td>0.3</td>
</tr>
</tbody>
</table>
### Result IPCA-CMI

**Table**: The result of gene expression data set DREAM3 Challenge with 100 genes and sample number 100

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>TP</th>
<th>FP</th>
<th>ACC</th>
<th>FPR</th>
<th>FDR</th>
<th>PPV</th>
<th>F</th>
<th>MCC</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA1</td>
<td>49</td>
<td>25</td>
<td>0.971</td>
<td>0.005</td>
<td>0.34</td>
<td>0.66</td>
<td>0.41</td>
<td>0.43</td>
<td>0.28</td>
</tr>
<tr>
<td>PCA2</td>
<td>46</td>
<td>25</td>
<td>0.971</td>
<td>0.005</td>
<td>0.35</td>
<td>0.64</td>
<td>0.38</td>
<td>0.41</td>
<td>0.27</td>
</tr>
<tr>
<td>IPCA1</td>
<td>53.11</td>
<td>29.77</td>
<td><strong>0.972</strong></td>
<td>0.006</td>
<td>0.35</td>
<td>0.65</td>
<td><strong>0.43</strong></td>
<td><strong>0.44</strong></td>
<td>0.32</td>
</tr>
<tr>
<td>IPCA2</td>
<td>46.55</td>
<td>15.16</td>
<td><strong>0.973</strong></td>
<td>0.003</td>
<td>0.24</td>
<td>0.75</td>
<td>0.4</td>
<td>0.45</td>
<td>0.28</td>
</tr>
</tbody>
</table>
**Result IPCA-CMI**

---

**Table:** The result of experimental data from Escherichia coil containing 9 genes and sample number 9

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>TP</th>
<th>FP</th>
<th>ACC</th>
<th>FPR</th>
<th>FDR</th>
<th>PPV</th>
<th>F</th>
<th>MCC</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA1</td>
<td>18</td>
<td>4</td>
<td>0.72</td>
<td>0.33</td>
<td>0.18</td>
<td>0.82</td>
<td>0.78</td>
<td>0.40</td>
<td>0.75</td>
</tr>
<tr>
<td>IPCA1</td>
<td>18</td>
<td>1.8</td>
<td>0.73</td>
<td>0.32</td>
<td>0.17</td>
<td>0.82</td>
<td>0.79</td>
<td>0.41</td>
<td>0.75</td>
</tr>
</tbody>
</table>
IPCA-CMI algorithm is written in Matlab. It gets simulated data and real data as inputs. This algorithm applied for learning the skeleton of GRNs. The source of the program and data sets are available at http://www.bioinf.cs.ipm.ir/software/IPCA-CMI/.
 Algorithms

CN: A Consensus Algorithm for Inferring Gene Regulatory Networks Using SORDER Algorithm and Conditional Mutual Information Test†

- PC algorithm is not robust, because it achieves different network topologies if gene orders are permuted.

- In addition, the performance of this algorithm depends on the threshold value used for independence tests.
SORDER Algorithm

To overcome the disadvantage of PC algorithm i.e. the dependence on the sequential ordering of vertices, the SORDER algorithm is proposed to select a suitable sequential ordering of vertices (genes) based on the degrees of vertices in $S_0$ (skeleton of order 0 resulted by PCA-CMI).

Node selection in SORDER algorithm is based on the maximum degree of nodes. In each step of SORDER algorithm, node with maximum degree is selected, and the selection process are repeated until all nodes of graph are chosen. When some nodes have the same degree in one step, an induced subgraph of these nodes are constructed. In a similar manner, nodes in the induced subgraph are selected according to the maximum degree.
SORDER Algorithm

The result of applying SORDER algorithm for graph $G$ is $O = (c, a, b, d, f, e)$.
SORDER Algorithm

In order to investigate the performance of the SORDER algorithm, at first all permutations of \( n \) nodes of networks which contain small set of variables (DREAM3 with 10 nodes and SOS network with 9 nodes) are determined. As the number of possible permutations for the network is very large and cannot be fully enumerated, we have selected \( N \) random permutations (\( N = 10^5 \)) of gene order and run PCA-CMI. For \( i = 1, \ldots, 10^5 \), \( T_i \) and \( T_{SORDER} \) are defined as \( T_i = (TP_i, FP_i) \) and \( T_{SORDER} = (TP_{SORDER}, FP_{SORDER}) \), respectively. Let

\[
X = \{ i | TP_i > TP_{SORDER} \text{ and } FP_i < FP_{SORDER} \} \text{ for } i = 1, \ldots, N.
\]
### SORDER Algorithm

**Table:** The comparison between SORDER algorithm and random permutation of nodes. **Exceeding Value (EV)** denotes the percentage of random permutations (unordered) which perform better than SORDER algorithm out of $10^5$ random permutations.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>EV value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DREAM50</td>
<td>0.018</td>
</tr>
<tr>
<td>DREAM100</td>
<td>0.048</td>
</tr>
</tbody>
</table>
CN Algorithm

Figure: Diagram of the CN algorithm.
CNMIT Algorithm

Figure: Diagram of the CNMIT algorithm. (a) Result of CN algorithm. (b) Values of MIT score for $k$ possible DAGs. (c) The result of CNMIT algorithm.
Figure: ROC curves of different methods for DREAM3 challenge with 10 nodes. The red line is related to the ROC curve of CN algorithm with a AUC of 0.9734.
Figure: Result of SORDER and CN algorithms. (A) The true network with 10 nodes and 10 edges. (B) Network inferred by SPCA-CMI algorithm. The edge with red line G2-G4 is false positives, while edges G1-G2, G3-G5 and G4-G9 are false negative. (C) Network obtained by the CN algorithm. The edge with red line G2-G9 is false positives and G4-G9 is false negative, but edges G1-G2 and G3-G5 are successfully found by this algorithm in comparison with that of (B).
Algorithms

Result

(a) 

(b) 

(c) 

(d)
Figure: Comparison of TP and FP values of CN algorithm with other methods for learning (a) DREAM3 Challenge with 10 genes, (b) DREAM3 Challenge with 50 gene, (c) DREAM3 Challenge with 100 genes, (d) SOS network with 9 genes. Also, comparison of CNMIT algorithm with different methods for learning (e) DREAM3 Challenge with 50 genes, (f) DREAM3 Challenge with 100 genes, (LA2012: LASSO2012; Gsqrt: GENIE3-FR-sqrt0; Gall: GENIE3-FR-all).
Algorithms

Result

(a)  
(b)  
(c)  
(d)
**Figure:** Comparison of ACC value and F-measure of CN algorithm with other methods for learning (a) DREAM3 Challenge with 10 genes, (b) DREAM3 Challenge with 50 genes, (c) DREAM3 Challenge with 100 genes, (d) SOS network with 9 genes. Comparison of CNMIT algorithm with different methods for learning (e) DREAM3 Challenge with 50 genes and 77 edges, (f) DREAM3 Challenge with 100 genes and 166 edges, (LA2012: LASSO2012; Gsqrt: GENIE3-FR-sqrt0; Gall: GENIE3-FR-all).

<table>
<thead>
<tr>
<th>Method</th>
<th>LASSO</th>
<th>LP</th>
<th>PCA-PCC</th>
<th>MI</th>
<th>PCA-CMI</th>
<th>CN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUCD10</td>
<td>0.813</td>
<td>0.75</td>
<td>0.897</td>
<td>0.93</td>
<td>0.9642</td>
<td>0.9734</td>
</tr>
<tr>
<td>AUCD50</td>
<td>0.819</td>
<td>0.559</td>
<td>0.78</td>
<td>0.832</td>
<td>0.8421</td>
<td>0.8515</td>
</tr>
</tbody>
</table>
### Table: Comparison of different methods for Learning DREAM4 Challenge. Team-Name is the name of the team which registered for this challenge (PCA-CMI: Path Consistency Algorithm based on Conditional Mutual Information; CN: Consensus Network Algorithm, the best performer for the relative item is noted in bold).

<table>
<thead>
<tr>
<th>Method</th>
<th>Net1</th>
<th>Net2</th>
<th>Net3</th>
<th>Net4</th>
<th>Net5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team415</td>
<td>0.75</td>
<td>0.69</td>
<td>0.76</td>
<td>0.77</td>
<td>0.76</td>
</tr>
<tr>
<td>Team549</td>
<td>0.73</td>
<td>0.7</td>
<td>0.74</td>
<td>0.74</td>
<td>0.74</td>
</tr>
<tr>
<td>Team395</td>
<td>0.69</td>
<td>0.64</td>
<td>0.72</td>
<td>0.72</td>
<td>0.71</td>
</tr>
<tr>
<td>PCA-CMI</td>
<td>0.7</td>
<td>0.69</td>
<td>0.74</td>
<td>0.74</td>
<td>0.74</td>
</tr>
<tr>
<td>CN</td>
<td>0.75</td>
<td>0.74</td>
<td>0.76</td>
<td>0.7</td>
<td>0.76</td>
</tr>
</tbody>
</table>
Table: Comparison of AUC for real data sets (AUCSOS: AUC values for a SOS network with 9 genes, AUCECO: AUC values for E.coil network with 8 genes, the best performer for the relative item is noted in bold).

<table>
<thead>
<tr>
<th>Method</th>
<th>LASSO</th>
<th>LP</th>
<th>PCA-PCC</th>
<th>MI</th>
<th>PCA-CMI</th>
<th>CN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUCSOS</td>
<td>0.61</td>
<td>0.59</td>
<td>0.67</td>
<td>0.73</td>
<td>0.74</td>
<td><strong>0.75</strong></td>
</tr>
<tr>
<td>AUCECO</td>
<td>0.63</td>
<td>0.54</td>
<td>0.6</td>
<td>0.63</td>
<td>0.64</td>
<td><strong>0.84</strong></td>
</tr>
</tbody>
</table>
Conclusion

- In this study new algorithms called IPCA-CMI and CN for inferring GRNs from gene expression data was presented.

- Zhang [34] reported that the PCA-CMI performed better than linear programming method [33], multiple linear regression Lasso method [31], mutual information method [25] and PC-Algorithm based on partial correlation coefficient [21] for inferring networks from gene expression data such as DREAM3 Challenge and SOS DNA repair network.

- Software in the form of MATLAB and JAVA codes. The source of data sets and codes are available at http://bs.ipm.ir/softwares/IPCA-CMI/.
In this study, we proved that PCA-CMI is an order-dependent algorithm and the result can be changed considering different orders of $n$ nodes. The results showed that determining a suitable sequential ordering of nodes can improve the performance of the PCA-CMI.

Moreover, constraint-based methods that apply CMI tests are dependent on the selected threshold for CMI tests.

Without using the SORDER algorithm, applying the CN algorithm is meaningless.

Software in the form of MATLAB and JAVA codes. The source of data sets and codes are available at http://bs.ipm.ir/softwares/CN/.
Thank You
References


<table>
<thead>
<tr>
<th>Outline</th>
<th>Challenge</th>
<th>Bayesian Network</th>
<th>Algorithms</th>
<th>Conclusion</th>
</tr>
</thead>
</table>


