Systems biology
Learning signalling pathways and regulatory networks from postgenomic data

Static Bayesian networks
• Marriage between graph theory and probability theory.
• Directed acyclic graph (DAG) represents conditional independence relations.
• Markov assumption leads to a factorization of the joint probability distribution:

\[
P(A, B, C, D, E, F) = P(A) \cdot P(B|A) \cdot P(C|A) \cdot P(D|B, C) \cdot P(E|D) \cdot P(F|C, D)
\]

Bayesian networks versus causal networks
Bayesian networks represent conditional (in)dependency relations - not necessarily causal interactions.
### Equivalence classes of BNs

\[
\begin{align*}
P(A) \cdot P(B) \cdot P(C \mid A, B) &= P(A) \cdot P(B, C) \cdot P(C \mid A) \\
&= P(C \mid B) \cdot P(B) \cdot P(C) \cdot P(A) \cdot P(A^{-1}) \\
&= P(A \mid C) \cdot P(B) \cdot P(C \mid B) \\
&= P(A \mid C) \cdot P(B, C) \\
&= P(A \mid C) \cdot P(B \mid C) \cdot P(C)
\end{align*}
\]

- $P(A, B) \neq P(A) \cdot P(B)$
- $P(A, B, C) = P(A \mid C) \cdot P(B \mid C) \cdot P(C)$

- Completed partially directed graphs (CPDAGs)

### Bayesian networks

\[
P(\text{graph} \mid \text{data}) = \frac{P(\text{data} \mid \text{graph}) \cdot P(\text{graph})}{P(\text{data})} \propto P(\text{data} \mid \text{graph}) \cdot P(\text{graph})
\]

\[
= P(\text{graph}) \cdot \int P(\text{data}, \theta(\text{graph}) \mid \text{graph}) d\theta(\text{graph})
\]

**BGe metric:** closed form solution

\[
= \text{score}_{BGe}(\text{graph} \mid \text{data})
\]

**BDe metric:** closed form solution

\[
= \text{score}_{BDe}(\text{graph} \mid \text{data})
\]

- See lecture (29-Apr-09)
- See lecture (6-May-09)
Learning the network structure

\[ \text{graph} \rightarrow \text{score(graph)} \]

<table>
<thead>
<tr>
<th>n</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>#DAGs</td>
<td>543</td>
<td>3.7 \times 10^6</td>
<td>7.8 \times 10^{11}</td>
<td>4.2 \times 10^{18}</td>
</tr>
</tbody>
</table>

**Idea:** Heuristically searching for the graph \( M^* \) that is most supported by the data \( P(M^*|\text{data}) \geq P(\text{graph}|\text{data}) \), e.g.: greedy search algorithm

**Remarks**

- It may make sense to perform the Greedy Search algorithm several times from different initialisations to avoid getting trapped in local optima.
- Both the BDe and the BGe scoring metrics can be used, and it is almost certain that they will lead to different results.
- It holds: \[ \text{Score}(G) = \frac{P(D|G) \cdot P(G)}{P(D)} \] does not depend on \( G \) and can be ignored.
- Neighbour graphs of \( G \) means graphs that can be reached from \( G \) by single-edge operations. At least edge additions and edge deletions should be allowed. Edge reversals can be seen as optional (advantageous!) whereby it doesn’t matter whether one includes all edge reversals or edge reversals of compelled edges only.
- Neighbour graphs can have the same score, then one may randomly draw one of those neighbour graphs with the highest score.
- To include a possibility to cross ’posterior landscape valleys’ with equal scores the following improved version of the greedy search algorithm can be implemented:

**Simple Greedy Search algorithm**

- **Initialisation:** Start from an arbitrary graph \( G \) (e.g. the empty-seeded DAG without any edges) \( G \) and set: \( G_1 = G \).
- **Iteration:** Repeat for \( i=1,2,3,... \)
  
  Determine all \( N=|N(G_i)| \) neighbour graphs \( G_{i,1},...,G_{i,N} \) of \( G_i \) and compute their scores: \[ \text{Score}(G_{i,k}) = P(D|G_{i,k}) \cdot P(G_{i,k}) \]
  
  IF \( \text{Score}(G_i) \geq \text{Score}(G_{i,k}) \) for \( k=1,...,N \), then stop the iteration and output \( G_i \).
  
  ELSE set \( G_{i+1} = G^* \) where \( G^* \) is the neighbour graph of \( G_i \) with the highest score. That is, it holds: \( \text{Score}(G^*) \geq \text{Score}(G_{i,k}) \) for all \( k \) and perform the next iteration step.

**Greedy Search algorithm improved**

- **Initialisation:** Start from an arbitrary graph \( G \) (e.g. the empty-seeded DAG without any edges) \( G \) and set: \( G_1 = G \). Set Counter=1.
- **Iteration:** Repeat for \( i=1,2,3,... \)
  
  Determine all \( N=|N(G_i)| \) neighbour graphs \( G_{i,1},...,G_{i,N} \) of \( G_i \) and compute their scores: \[ \text{Score}(G_{i,k}) = P(D|G_{i,k}) \cdot P(G_{i,k}) \]
  
  IF \( \text{Score}(G_i) > \text{Score}(G_{i,k}) \) for \( k=1,...,N \), AND IF Counter\geq10
  
  then stop the iteration and output \( G_i \).
  
  ELSEIF \( \text{Score}(G_i) = \text{Score}(G_{i,j}) \) for one or more neighbour graphs
  
  Then set Counter=Counter+1,
  
  and randomly draw one of these neighbour graphs \( G^* \) of \( G_i \) with
  
  \( \text{Score}(G_i) = \text{Score}(G^*) \) and set: \( G_{i+1} = G^* \).
  
  Then perform the next iteration step.
  
  ELSE Set Counter=1 and \( G_{i+1} = G^* \) where \( G^* \) is the neighbour graph of \( G_i \) with the highest score. That is, it holds: \( \text{Score}(G^*) \geq \text{Score}(G_{i,k}) \) for all \( k \) and perform the next iteration step.
**Greedy Search algorithm**

- To include another possibility to escape from local optima the following **optional step** can be included:

**Greedy Search algorithm 2**

- **Initialization**: Start from an arbitrary graph $G$ (e.g. the empty-seeded DAG without any edges) $G$ and set: $G_1=G$.
- **Iteration**: Repeat for $i=1,2,3,...$

Determine all $N=N(G_i)$ neighbour graphs $G_{i,1},...,G_{i,N}$ of $G_i$ and compute their scores: $\text{Score}(G_{i,k})=P(D|G_{i,k}) \cdot P(G_{i,k})$.

IF $\text{Score}(G_i) \geq \text{Score}(G_{i,k})$ for all $k$

REPEAT for $j=1,...,10$

Set $G_{j,0}=G_i$ REPEAT for $m=1,...,50$

Randomly draw a neighbour graph $G^*$ of $G_{i,m-1}$ and set $G_{j,m}=G^*$ END

Determine the scores of $G_{1,50},...,G_{10,50}$ and select the graph $G_{1,50}$ with the highest score.

IF $\text{Score}(G_{1,50}) > \text{Score}(G_i)$

set $G_{i+1}=G_{1,50}$ and perform the next iteration step
ELSE

stop the iteration and output $G_i$

ELSE set $G_{i+1}=G_{i,u}$ where $G_{i,u}$ is the neighbour graph of $G_i$ with the highest score. That is, it holds: $\text{Score}(G_{i,u}) \geq \text{Score}(G_{i,k})$ for $k=1,...,N$ and perform the next iteration step.

**Einfache verbale Beschreibung des Zusatzschrittes:**


**Learning the network structure**

*Distribution of $P(\text{graph}|\text{data})$*

Data are sparse → Intrinsic uncertainty of inference

- Large data set $D$: Best network structure $M^*$ well defined
- Small data set $D$: Intrinsic uncertainty about $M^*$
Example: 2 genes → 16 different (dynamic) network structures

Best network: maximum score $P(D|M)$

Identify the best network structure
Ideal scenario: Large data sets, low noise

Uncertainty about the best network structure
Limited number of experimental replications, high noise

Sample of high-scoring networks
Can we generalize this scheme to more than 2 genes?  
In principle yes. However ...
Markov Chain Monte Carlo (MCMC) sampling of Bayesian networks

Devise a Markov chain

\[ P_{j+1}(G_i) = \sum_k T(G_i \mid G_k) \cdot P_j(G_k) \]

that converges in distribution to the posterior probability \( P(G|D) \)

\[ P_j(G) \rightarrow P(G \mid D) \quad \text{for } j \rightarrow \infty \]

The Markov transition matrix \( T \) is a matrix of transition probabilities, with \( T(G_i \mid G_k) \) denoting the probability of a transition from graph \( G_k \) to graph \( G_i \). The important feature of a Markov chain is that, under the fairly weak condition of ergodicity, the distribution \( P_j(G_k) \) converges (for \( j \rightarrow \infty \)) to a stationary distribution \( P_\infty(G_k) \)

The stationary distribution is independent of the initialisation of the Markov chain and uniquely determined by the Markov transition matrix \( T \).

It holds:

\[ P_\infty(G_i) = \sum_k T(G_i \mid G_k) \cdot P_\infty(G_k) \]

The idea is to construct a Markov transition matrix \( T \) in such a way that the resulting Markov Chain has the posterior probability \( P(G|D) \) as its stationary distribution, symbolically \( P_\infty(G) = P(G|D) \).

Bayesian networks

\[ P(G|D) = \frac{P(D|G) \cdot P(G)}{P(D)} \]

\( P(G|D) \) is the posterior probability of \( G \) given the data \( D \)
\( P(D|G) \) is the marginal likelihood of the graph \( G \)
\( P(G) \) is the graph-prior
\( P(D) \) is the probability of the data:

\[ P(D) = \sum_{G'} P(D, G') = \sum_{G'} P(D \mid G') \cdot P(G') \]

intractable for network domains with more than \( n=5 \) or \( n=6 \) variables, since this is a sum over all possible (valid/acyclic) graphs, and the number of valid graphs grows super-exponentially in the number of nodes.
Bayesian networks

\[ P(G \mid D) = \frac{P(D \mid G) \cdot P(G)}{P(D)} \]

- \( P(G \mid D) \) is the posterior probability of \( G \) given the data \( D \)
- \( P(D \mid G) \) is the marginal likelihood of the graph \( G \)
- \( P(G) \) is the graph prior
- \( P(D) \) is the probability of the data:

\[ P(D) = \sum_{G'} P(D, G') = \sum_{G'} P(D \mid G') \cdot P(G') \]

- \( P(D) \) can be intractable but ratios \( P(G \mid D) / P(G' \mid D) \) can be computed, since \( P(D) \) cancels out in these ratios!!!

Ergodicity of Markov chains

A Markov chain is **ergodic** if it is **aperiodic** and **irreducible**. An irreducible Markov chain is one in which all states are reachable from all other states. A sufficient test for aperiodicity is that each state has a self-loop, meaning that the probability that the next state is the same as the current state is non-zero.

In general it is difficult to prove that a Markov Chain is ergodic. However, ergodicity can be assumed to hold in most real-world applications.

Detailed balance

The equation of detailed balance:

\[ \frac{T(G_k \mid G_i)}{T(G_i \mid G_k)} = \frac{P(G_k \mid D)}{P(G_i \mid D)} \]

\[ \Leftrightarrow T(G_k \mid G_i) \cdot P(G_i \mid D) = T(G_i \mid G_k) \cdot P(G_k \mid D) \]

is a sufficient condition for \( P(G \mid D) \) to be the stationary distribution \( P_\infty(G) \) of an ergodic Markov chain with Markov transition matrix \( T(\cdot|\cdot) \).

Proof

We have to show that:

\[ P(G_i \mid D) = \sum_k T(G_i \mid G_k) \cdot P(G_k \mid D) \]

\[ \Leftrightarrow P(G_i \mid D) = \sum_k T(G_i \mid G_k) \cdot P(G_k \mid D) \]

\[ \Leftrightarrow P(G_i \mid D) = P(G_i \mid D) \cdot \sum_k T(G_k \mid G_i) = 1 \]
Construction of Markov chains

In practice setting up a Markov chain consists of two parts. First, given \( G_k \), a new graph is proposed with a proposal probability \( Q(G_i|G_k) \). In a second step, the new graph is then accepted with an acceptance probability \( A(G_i|G_k) \), or rejected otherwise. A transition probability \( T(G_i|G_k) \) is therefore given by the product of a proposal and an acceptance probability and can be written as:

\[
T(G_i|G_k) = Q(G_i|G_k) \cdot A(G_i|G_k) \quad (i \neq j)
\]

The proposal probabilities \( Q(G_i|G_k) \) are defined by the way one designs the moves in the graph space, and the acceptance probabilities \( A(G_i|G_k) \) depend on \( Q(G_i|G_k) \).

The goal is that the equation of detailed balance holds for the transition probabilities \( T(G_i|G_k) \).

How to specify \( A(G_k|G_i) \)?

\[
\frac{T(G_i|G_k)}{T(G_k|G_i)} = \frac{P(G_i|D)}{P(G_k|D)} \quad \iff \quad \frac{Q(G_i|G_k) \cdot A(G_i|G_k)}{Q(G_k|G_i) \cdot A(G_k|G_i)} = \frac{P(G_i|D)}{P(G_k|D)}
\]

\[
\iff \quad A(G_k|G_i) = \frac{P(G_i|D) \cdot Q(G_i|G_k)}{P(G_k|D) \cdot Q(G_k|G_i)}
\]

The equation of detailed balance is satisfied, if we set for \( i \neq j \):

\[
A(G_k|G_i) = \min \left\{ 1, \frac{P(G_i|D) \cdot Q(G_i|G_k)}{P(G_k|D) \cdot Q(G_k|G_i)} \right\}
\]

\[
\iff A(G_k|G_i) = \min \left\{ \frac{P(D|G_i) \cdot P(G_i)}{P(D|G_k) \cdot P(G_k)} \cdot \frac{Q(G_i|G_k)}{Q(G_k|G_i)} \right\}
\]

.Hastings ratio

Metropolis Hastings algorithm
structure MCMC for Bayesian networks

Initialisation: Start from an arbitrary initial graph \( G \)
(e.g. the empty-seeded graph) and set \( G_0 = G \).

Iteration: For \( i = 1, \ldots, T \)
- Obtain a new graph \( G_i \) from the proposal distribution \( Q(G_i|G_{i-1}) \)
- Accept the new graph with probability \( A(G_i|G_{i-1}) \) whereby
  \( A(\cdot) \) has to be specified as described above; otherwise reject
  \( G_{i-1} \) leave the Markov chain state unchanged; symbolically: \( G_i = G_{i-1} \).

END
Discard an initial 'burn-in' period to allow the Markov chain to reach stationarity, i.e to converge. For example discard the first \( I \cdot T \) MCMC samples.

Output: An MCMC sample from the posterior distribution \( P(G|D) \), symbolically:
MCMC sample: \( G_{1, \ldots, T} \)

Notes on Metropolis Hastings algorithm
structure MCMC for Bayesian networks

The proposal moves are based on single edge operations and randomly drawn neighbour graphs.

That is, the proposal probability \( Q(G_i|G_{i-1}) \) is a discrete uniform distribution on all neighbour graphs of \( G_{i-1} \).

It holds:

\[
Q(G|G_{i-1}) = 1/|N(G_{i-1})| \quad \text{for all neighbour graphs } G \text{ of } G_{i-1},
\]

and

\[
Q(G|G_{i-1}) = 0 \quad \text{for all non-neighbour graphs } G \text{ of } G_{i-1}.
\]

\(|N(G_{i-1})|\) is the cardinality of the neighbour graph set of \( G_{i-1} \).
Single-edge-operations - structure MCMC moves

MCMC moves
- Delete edge
- Reverse edge
- Create edge

MCMC algorithm converges to the posterior distribution $P(\text{graph}|\text{data})$.

Hastings ratio
$Q(G_i|G_{i-1}) \neq Q(G_{i-1}|G_i)$ is possible

Acceptance probability: $\min\left\{1, \frac{P(\text{graph}) P(M_{i=0})}{P(M_{i=0}) P(\text{graph})} \times \frac{Q(M_{i=0}|M_{i=0})}{Q(M_{i=0}|M_{i=0})}\right\}$

MCMC sampling of Bayesian networks

The DAG sample $G_1, G_2, G_3, \ldots G_T$ is generated via Markov Chain Monte Carlo (MCMC) simulations.

That is, via simulation of a Markov Chain $(M_t)_t$ in the space of DAGs whose distribution converges to the graph posterior distribution:

$P(M_t=\text{graph}|\text{data}) \to P(\text{graph}|\text{data})$

$t \to \infty$
MCMC sampling of Bayesian networks

The DAG sample \(G_1, G_2, G_3, \ldots, G_T\) is generated via Markov Chain Monte Carlo (MCMC) simulations. That is, via simulation of a Markov Chain \((M_t)_t\) in the space of DAGs whose distribution converges to the graph posterior distribution:

\[
P(M_t = \text{graph} | \text{data}) \rightarrow P(\text{graph} | \text{data})
\]

In practice: \(t\) is not infinite!

Convergence diagnostics

In theory, an ergodic Markov chain converges to the true posterior distribution \(P(G|D)\). In practice, it is difficult to decide whether an MCMC simulation has sufficiently converged. Simple convergence diagnostics are trace plots of the logarithmic scores. Another — more reliable — convergence diagnostic bases on scatter plots of marginal edge posterior probabilities will be presented later. Passing these diagnostic tests is only a necessary rather than a sufficient condition for convergence, as it may not distinguish between meta-stable disequilibrium and true equilibrium (e.g. locally optimal regions of the posterior landscape).

Trace plot diagnostics

<table>
<thead>
<tr>
<th>Bad convergence</th>
<th>Good convergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>different plateaus</td>
<td>same region is reached</td>
</tr>
</tbody>
</table>

MCMC sample trace plots of \(\log(\text{Scores}(G_i)) = \log(\text{P(D|G_i)P(G_i)})\) for differently seeded (independent) structure MCMC runs

Edge relation features

Utilise the DAG (CPDAG) sample \(G_{J+1}, \ldots, G_T\) for estimating the posterior probabilities of edge relation features:

\[
\hat{P}(A \rightarrow B) = \frac{1}{(T-J)} \sum_{i=J+1}^{T} I(G_i)
\]

where \(I(G_i) = 1\) if the CPDAG of \(G_i\) contains the directed edge \(A \rightarrow B\), and 0 otherwise.

interpretation superposition
Convergence check via independent MCMC runs and scatter plots of edge relation feature posterior probabilities.