Bayesian Network Modelling

with Examples

Marco Scutari

scutari@stats.ox.ac.uk
Department of Statistics
University of Oxford

November 28, 2016

What Are Bayesian Networks?

A Graph and a Probability Distribution

Bayesian networks (BNs) are defined by:

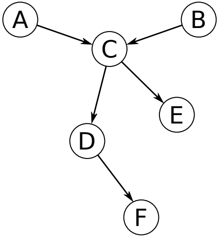
- a network structure, a directed acyclic graph $\mathcal{G} = (\mathbf{V}, A)$, in which each node $v_i \in \mathbf{V}$ corresponds to a random variable X_i;
- a global probability distribution $\mathbf{X}$ with parameters Θ, which can be factorised into smaller local probability distributions according to the arcs $a_{ij} \in A$ present in the graph.

The main role of the network structure is to express the conditional independence relationships among the variables in the model through graphical separation, thus specifying the factorisation of the global distribution:

$$\mathrm{P}(\mathbf{X}) = \prod_{i=1}^{p} \mathrm{P}(X_i \mid \Pi_{X_i}; \Theta_{X_i}) \quad \text{where} \quad \Pi_{X_i} = \{\text{parents of } X_i\}$$

How the DAG Maps to the Probability Distribution

DAG



$\Longrightarrow$

Graphical separation

$A \perp\!\!\!\perp_G B$
$A \perp\!\!\!\perp_G D \mid C$
$B \perp\!\!\!\perp_G D \mid C$
$A \perp\!\!\!\perp_G E \mid C$
$B \perp\!\!\!\perp_G E \mid C$
$D \perp\!\!\!\perp_G E \mid C$
$C \perp\!\!\!\perp_G F \mid D$
$\ldots$

$\Longrightarrow$

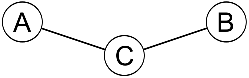
Probabilistic independence

$A \perp\!\!\!\perp_P B$
$A \perp\!\!\!\perp_P D \mid C$
$B \perp\!\!\!\perp_P D \mid C$
$A \perp\!\!\!\perp_P E \mid C$
$B \perp\!\!\!\perp_P E \mid C$
$D \perp\!\!\!\perp_P E \mid C$
$C \perp\!\!\!\perp_P F \mid D$
$\ldots$

Formally, the DAG is an independence map of the probability distribution of $\mathbf{X}$, with graphical separation ($\perp\!\!\!\perp_G$) implying probabilistic independence ($\perp\!\!\!\perp_P$).

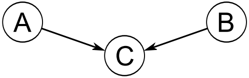
Graphical Separation in DAGs (Fundamental Connections)

separation (undirected graphs)

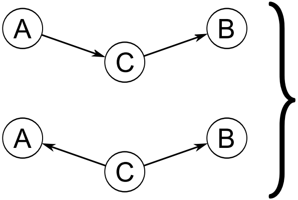


$$\mathbf{A} \perp\!\!\!\perp \mathbf{B} \mid \mathbf{C}$$
$$\mathrm{P}(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \mathrm{P}(\mathbf{A} \mid \mathbf{C})\,\mathrm{P}(\mathbf{B} \mid \mathbf{C})\,\mathrm{P}(\mathbf{C})$$

d-separation (directed acyclic graphs)



$$\mathbf{A} \not\perp\!\!\!\perp \mathbf{B} \mid \mathbf{C}$$
$$\mathrm{P}(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \mathrm{P}(\mathbf{C} \mid \mathbf{A}, \mathbf{B})\,\mathrm{P}(\mathbf{A})\,\mathrm{P}(\mathbf{B})$$



$$\mathbf{A} \perp\!\!\!\perp \mathbf{B} \mid \mathbf{C}$$
$$\begin{aligned}\mathrm{P}(\mathbf{A}, \mathbf{B}, \mathbf{C}) &= \\ &= \mathrm{P}(\mathbf{B} \mid \mathbf{C})\,\mathrm{P}(\mathbf{C} \mid \mathbf{A})\,\mathrm{P}(\mathbf{A}) \\ &= \mathrm{P}(\mathbf{A} \mid \mathbf{C})\,\mathrm{P}(\mathbf{B} \mid \mathbf{C})\,\mathrm{P}(\mathbf{C})\end{aligned}$$

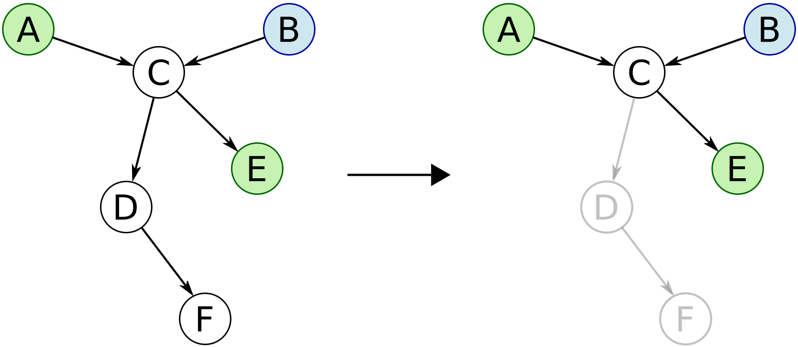
Graphical Separation in DAGs (General Case)

Now, in the general case we can extend the patterns from the fundamental connections and apply them to every possible path between **A** and **B** for a given **C**; this is how d-separation is defined.

> *If* $\mathbf{A}$, $\mathbf{B}$ *and* $\mathbf{C}$ *are three disjoint subsets of nodes in a directed acyclic graph* $\mathcal{G}$, *then* $\mathbf{C}$ *is said to d-separate* $\mathbf{A}$ *from* $\mathbf{B}$, *denoted* $\mathbf{A} \perp\!\!\!\perp_G \mathbf{B} \mid \mathbf{C}$, *if along every path between a node in* $\mathbf{A}$ *and a node in* $\mathbf{B}$ *there is a node* v *satisfying one of the following two conditions:*
>
> 1. v *has converging edges (i.e. there are two edges pointing to* v *from the adjacent nodes in the path) and none of* v *or its descendants (i.e. the nodes that can be reached from* v*) are in* $\mathbf{C}$.
> 2. v *is in* $\mathbf{C}$ *and does not have converging edges.*

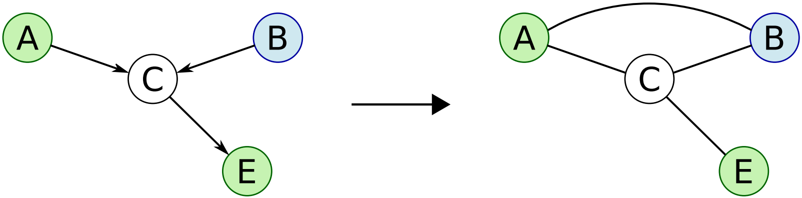
This definition clearly does not provide a computationally feasible approach to assess d-separation; but there are other ways.

A Simple Algorithm to Check D-Separation (I)



Say we want to check whether A and E are d-separated by B. First, we can drop all the nodes that are not ancestors (*i.e.* parents, parents' parents, etc.) of A, E and B since each node only depends on its parents.

A Simple Algorithm to Check D-Separation (II)

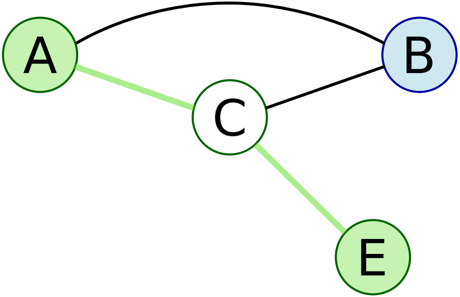


Transform the subgraph into its moral graph by

1. connecting all nodes that have one parent in common; and
2. removing all arc directions to obtain an undirected graph.

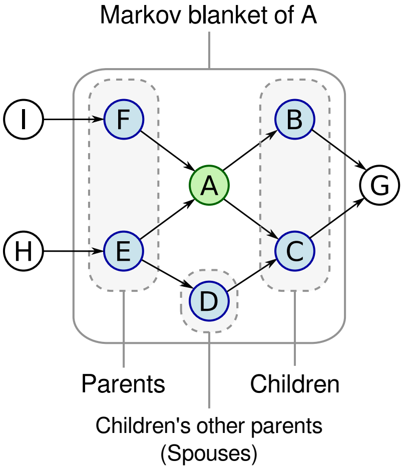
This transformation has the double effect of making the dependence between parents explicit by “marrying” them and of allowing us to use the classic definition of graphical separation.

A Simple Algorithm to Check D-Separation (III)



Finally, we can just perform *e.g.* a depth-first or breadth-first search and see if we can find an open path between A and B, that is, a path that is not blocked by C.

Completely D-Separating: Markov Blankets



We can easily use the DAG to solve the feature selection problem. The set of nodes that graphically isolates a target node from the rest of the DAG is called its Markov blanket and includes:

- its parents;
- its children;
- other nodes sharing a child.

Since $\perp\!\!\!\perp_G$ implies $\perp\!\!\!\perp_P$, we can restrict ourselves to the Markov blanket to perform any kind of inference on the target node, and disregard the rest.

Different DAGs, Same Distribution: Topological Ordering

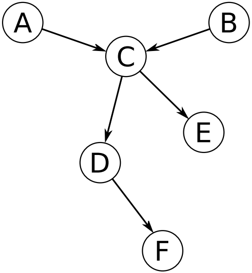
A DAG uniquely identifies a factorisation of $\mathrm{P}(\mathbf{X})$; the converse is not necessarily true. Consider again the DAG on the left:

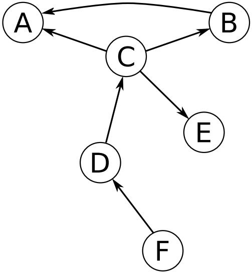
$$\mathrm{P}(\mathbf{X}) = \mathrm{P}(A)\,\mathrm{P}(B)\,\mathrm{P}(C \mid A, B)\,\mathrm{P}(D \mid C)\,\mathrm{P}(E \mid C)\,\mathrm{P}(F \mid D).$$

We can rearrange the dependencies using Bayes theorem to obtain:

$$\mathrm{P}(\mathbf{X}) = \mathrm{P}(A \mid B, C)\,\mathrm{P}(B \mid C)\,\mathrm{P}(C \mid D)\,\mathrm{P}(D \mid F)\,\mathrm{P}(E \mid C)\,\mathrm{P}(F),$$

which gives the DAG on the right, with a different topological ordering.





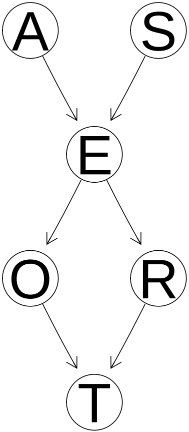
An Example: Train Use Survey

Consider a simple, hypothetical survey whose aim is to investigate the usage patterns of different means of transport, with a focus on cars and trains.

- Age (A): *young* for individuals below 30 years old, *adult* for individuals between 30 and 60 years old, and *old* for people older than 60.
- Sex (S): *male* or *female*.
- Education (E): *up to high school* or *university degree*.
- Occupation (O): *employee* or *self-employed*.
- Residence (R): the size of the city the individual lives in, recorded as either *small* or *big*.
- Travel (T): the means of transport favoured by the individual, recorded either as *car*, *train* or *other*.

The nature of the variables recorded in the survey suggests how they may be related with each other.

The Train Use Survey as a Bayesian Network (v1)

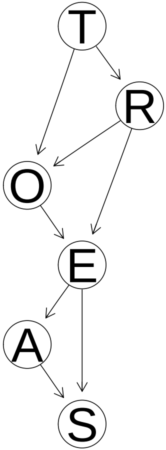


That is a prognostic view of the survey as a BN:

1. the blocks in the experimental design on top (e.g. stuff from the registry office);
2. the variables of interest in the middle (e.g. socio-economic indicators);
3. the object of the survey at the bottom (e.g. means of transport).

Variables that can be thought as "causes" are on above variables that can be considered their "effect", and confounders are on above everything else.

The Train Use Survey as a Bayesian Network (v2)



That is a diagnostic view of the survey as a BN: it encodes the same dependence relationships as the prognostic view but is laid out to have “effects” on top and “causes” at the bottom.

Depending on the phenomenon and the goals of the survey, one may have a graph that makes more sense than the other; but they are equivalent for any subsequent inference. For discrete BNs, one representation may have fewer parameters than the other.

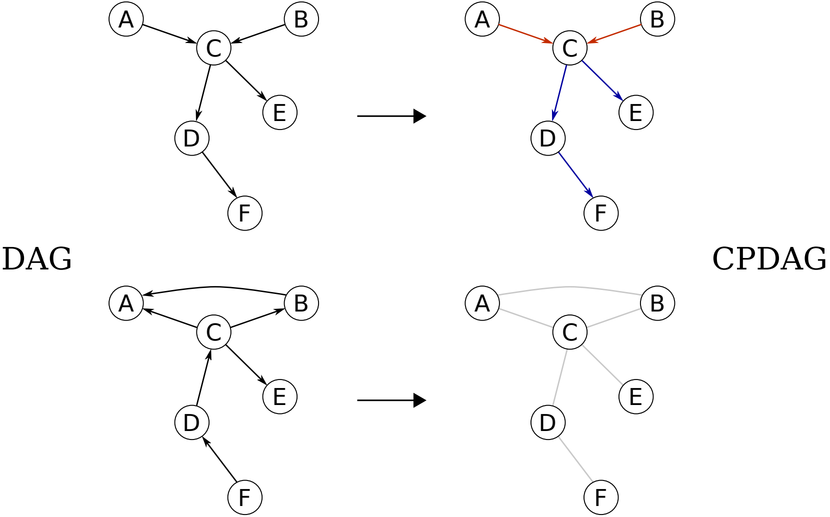
Different DAGs, Same Distribution: Equivalence Classes

On a smaller scale, even keeping the same underlying undirected graph we can reverse a number of arcs without changing the dependence structure of $\mathbf{X}$. Since the triplets $A \rightarrow B \rightarrow C$ and $A \leftarrow B \rightarrow C$ are probabilistically equivalent, we can reverse the directions of their arcs as we like as long as we do not create any new v-structure ($A \rightarrow B \leftarrow C$, with no arc between A and C).

This means that we can group DAGs into equivalence classes that are uniquely identified by the underlying undirected graph and the v-structures. The directions of other arcs can be either:

- uniquely identifiable because one of the directions would introduce cycles or new v-structures in the graph (compelled arcs);
- completely undetermined.

Completed Partially Directed Acyclic Graphs (CPDAGs)



DAG

CPDAG

What About the Probability Distributions?

The second component of a BN is the probability distribution $\mathrm{P}(\mathbf{X})$. The choice should such that the BN:

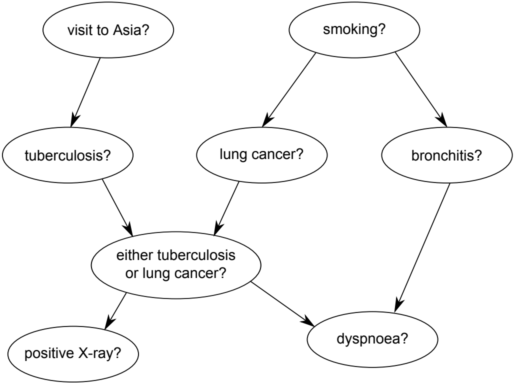
- can be learned efficiently from data;
- is flexible (distributional assumptions should not be too strict);
- is easy to query to perform inference.

The three most common choices in the literature (by far), are:

- discrete BNs (DBNs), in which $\mathbf{X}$ and the $X_i \mid \Pi_{X_i}$ are multinomial;
- Gaussian BNs (GBNs), in which $\mathbf{X}$ is multivariate normal and the $X_i \mid \Pi_{X_i}$ are univariate normal;
- conditional linear Gaussian BNs (CLGBNs), in which $\mathbf{X}$ is a mixture of multivariate normals and the $X_i \mid \Pi_{X_i}$ are either multinomial, univariate normal or mixtures of normals.

It has been proved in the literature that exact inference is possible in these three cases, hence their popularity.

Discrete Bayesian Networks



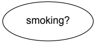
A classic example of DBN is the ASIA network from Lauritzen & Spiegelhalter (1988), which includes a collection of binary variables. It describes a simple diagnostic problem for tuberculosis and lung cancer.

Total parameters of $\mathbf{X}$: $2^8 - 1 = 255$

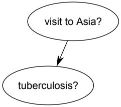
Conditional Probability Tables (CPTs)

visit to Asia?

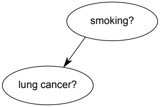
yes	no
0.01	0.99



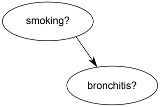
yes	no
0.50	0.50



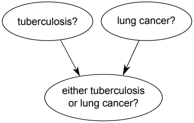
	yes	no
yes	0.05	0.95
no	0.01	0.99



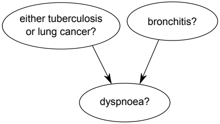
	yes	no
yes	0.10	0.90
no	0.01	0.99



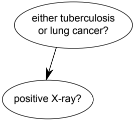
	yes	no
yes	0.60	0.40
no	0.30	0.70



	yes	no
yes:yes	1	0
yes:no	1	0
no:yes	1	0
no:no	0	1



	yes	no
yes:yes	0.90	0.10
yes:no	0.70	0.30
no:yes	0.80	0.20
no:no	0.10	0.90

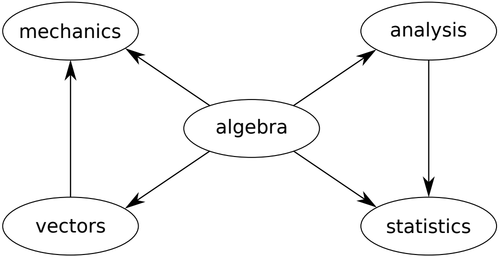


	yes	no
yes	0.98	0.02
no	0.05	0.95

The local distributions $X_i \mid \Pi_{X_i}$ take the form of conditional probability tables for each node given all the configurations of the values of its parents.

Overall parameters of the $X_i \mid \Pi_{X_i}$: 18

Gaussian Bayesian Networks



A classic example of GBN is the MARKS networks from Mardia, Kent & Bibby JM (1979), which describes the relationships between the marks on 5 math-related topics.

Assuming $X \sim N(\boldsymbol{\mu}, \Sigma)$, we can compute $\Omega = \Sigma^{-1}$. Then $\Omega_{ij} = 0$ implies $X_i \perp\!\!\!\perp_P X_j \mid \mathbf{X} \setminus \{X,X_j\}$. The absence of an arc $X_i \rightarrow X_j$ in the DAG implies $X_i \perp\!\!\!\perp_G X_j \mid \mathbf{X} \setminus \{X,X_j\}$, which in turn implies $X_i \perp\!\!\!\perp_P X_j \mid \mathbf{X} \setminus \{X,X_j\}$.

Total parameters of $\mathbf{X}$: $5 + 15 = 20$

Partial Correlations and Linear Regressions

The local distributions $X_i \mid \Pi_{X_i}$ take the form of linear regression models with the Π_{X_i} acting as regressors and with independent error terms.

$$\begin{aligned}
\mathsf{ALG} &= 50.60 + \varepsilon_{\mathsf{ALG}} \sim N(0, 112.8)\\
\mathsf{ANL} &= -3.57 + 0.99\mathsf{ALG} + \varepsilon_{\mathsf{ANL}} \sim N(0, 110.25)\\
\mathsf{MECH} &= -12.36 + 0.54\mathsf{ALG} + 0.46\mathsf{VECT} + \varepsilon_{\mathsf{MECH}} \sim N(0, 195.2)\\
\mathsf{STAT} &= -11.19 + 0.76\mathsf{ALG} + 0.31\mathsf{ANL} + \varepsilon_{\mathsf{STAT}} \sim N(0, 158.8)\\
\mathsf{VECT} &= 12.41 + 0.75\mathsf{ALG} + \varepsilon_{\mathsf{VECT}} \sim N(0, 109.8)
\end{aligned}$$

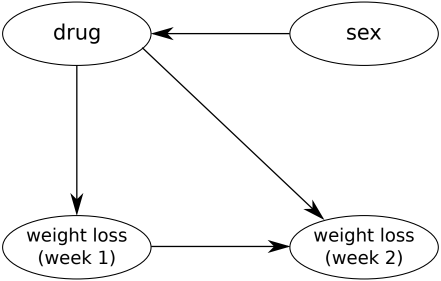
(That is because $\Omega_{ij} \propto \beta_j$ for X_i, so $\beta_j > 0$ if and only if $\Omega_{ij} > 0$. Also $\Omega_{ij} \propto \rho_{ij}$, the partial correlation between X_i and X_j, so we are implicitly assuming all probabilistic dependencies are linear.)

Overall parameters of the $X_i \mid \Pi_{X_i}$: $11 + 5 = 16$

Conditional Linear Gaussian Bayesian Networks

CLGBNs contain both discrete and continuous nodes, and combine DBNs and GBNs as follows to obtain a mixture-of-Gaussians network:

- continuous nodes cannot be parents of discrete nodes;
- the local distribution of each discrete node is a CPT;
- the local distribution of each continuous node is a set of linear regression models, one for each configurations of the discrete parents, with the continuous parents acting as regressors.



One of the classic examples is the RATS' WEIGHTS network from Edwards (1995), which describes weight loss in a drug trial performed on rats.

Mixtures of Linear Regressions

The resulting local distribution for the first weight loss for drugs D_1, D_2 and D_3 is:

$$\begin{aligned}
\mathrm{W}_{1,\mathrm{D}_1} &= 7 + \varepsilon_{\mathrm{D}_1} \sim N(0, 2.5) \\
\mathrm{W}_{1,\mathrm{D}_2} &= 7.50 + \varepsilon_{\mathrm{D}_2} \sim N(0, 2) \\
\mathrm{W}_{1,\mathrm{D}_3} &= 14.75 + \varepsilon_{\mathrm{D}_3} \sim N(0, 11)
\end{aligned}$$

with just the intercepts since the node has no continuous parents. The local distribution for the second loss is:

$$\begin{aligned}
\mathrm{W}_{2,\mathrm{D}_1} &= 1.02 + 0.89\beta_{\mathrm{W}_1} + \varepsilon_{\mathrm{D}_1} \sim N(0, 3.2) \\
\mathrm{W}_{2,\mathrm{D}_2} &= -1.68 + 1.35\beta_{\mathrm{W}_1} + \varepsilon_{\mathrm{D}_2} \sim N(0, 4) \\
\mathrm{W}_{2,\mathrm{D}_3} &= -1.83 + 0.82\beta_{\mathrm{W}_1} + \varepsilon_{\mathrm{D}_3} \sim N(0, 1.9)
\end{aligned}$$

Overall, they look like random effect models with random intercepts and random slopes.

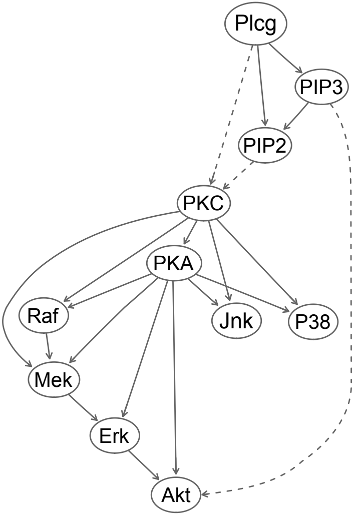
Case Study: Human Physiology

Source and Overview of the Data

Causal Protein-Signalling Networks Derived from Multiparameter Single Cell Data. Karen Sachs, *et al.*, Science, **308**, 523 (2005).

That is a landmark application of BNs because it highlights the use of interventional data; and because results are validated. The data consist in the 5400 simultaneous measurements of 11 phosphorylated proteins and phospholipids; 1800 are subjected to spiking and knock-outs to control expression.

The goal of the analysis is to learn what relationships link these 11 proteins, that is, the signalling pathways they are part of.



Bayesian Network Structure Learning

Learning a BN $\mathcal{B} = (\mathcal{G}, \Theta)$ from a data set $\mathcal{D}$ is performed in two steps:

$$\underbrace{\mathrm{P}(\mathcal{B} \mid \mathcal{D}) = \mathrm{P}(\mathcal{G}, \Theta \mid \mathcal{D})}_{\text{learning}} \quad = \quad \underbrace{\mathrm{P}(\mathcal{G} \mid \mathcal{D})}_{\text{structure learning}} \quad \cdot \quad \underbrace{\mathrm{P}(\Theta \mid \mathcal{G}, \mathcal{D})}_{\text{parameter learning}} .$$

In a Bayesian setting structure learning consists in finding the DAG with the best $\mathrm{P}(\mathcal{G} \mid \mathcal{D})$. We can decompose $\mathrm{P}(\mathcal{G} \mid \mathcal{D})$ into

$$\mathrm{P}(\mathcal{G} \mid \mathcal{D}) \propto \mathrm{P}(\mathcal{G})\,\mathrm{P}(\mathcal{D} \mid \mathcal{G}) = \mathrm{P}(\mathcal{G}) \int \mathrm{P}(\mathcal{D} \mid \mathcal{G}, \Theta)\,\mathrm{P}(\Theta \mid \mathcal{G}) d\Theta$$

where $\mathrm{P}(\mathcal{G})$ is the prior distribution over the space of the DAGs and $\mathrm{P}(\mathcal{D} \mid \mathcal{G})$ is the marginal likelihood of the data given $\mathcal{G}$ averaged over all possible parameter sets Θ; and then

$$\mathrm{P}(\mathcal{D} \mid \mathcal{G}) = \prod_{i=1}^{N} \left[\int \mathrm{P}(X_i \mid \Pi_{X_i}, \Theta_{X_i})\,\mathrm{P}(\Theta_{X_i} \mid \Pi_{X_i}) d\Theta_{X_i} \right] .$$

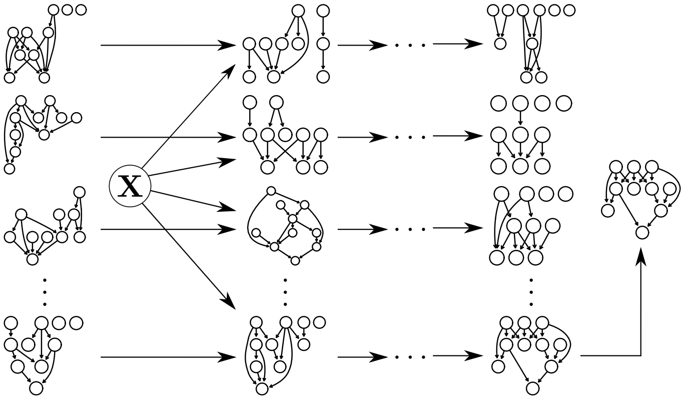
The Hill-Climbing Algorithm

The most common score-based structure learning algorithm, in which we are looking for the DAG that maximises a score such as the posterior $\mathrm{P}(\mathcal{G} \mid \mathcal{D})$ or BIC, is a greedy search such as hill-climbing:

```
1. Choose an initial DAG $\mathcal{G}$, usually (but not necessarily) empty.
2. Compute the score of $\mathcal{G}$, denoted as $Score_{\mathcal{G}} = \mathrm{Score}(\mathcal{G})$.
3. Set $maxscore = Score_{\mathcal{G}}$.
4. Repeat the following steps as long as $maxscore$ increases:
    4.1 for every possible arc addition, deletion or reversal not introducing cycles:
        4.1.1 compute the score of the modified DAG $\mathcal{G}^*$, $Score_{\mathcal{G}^*} = \mathrm{Score}(\mathcal{G}^*)$:
        4.1.2 if $Score_{\mathcal{G}^*} > Score_{\mathcal{G}}$, set $\mathcal{G} = \mathcal{G}^*$ and $Score_{\mathcal{G}} = Score_{\mathcal{G}^*}$.
    4.2 update $maxscore$ with the new value of $Score_{\mathcal{G}}$.
5. Return the DAG $\mathcal{G}$.
```

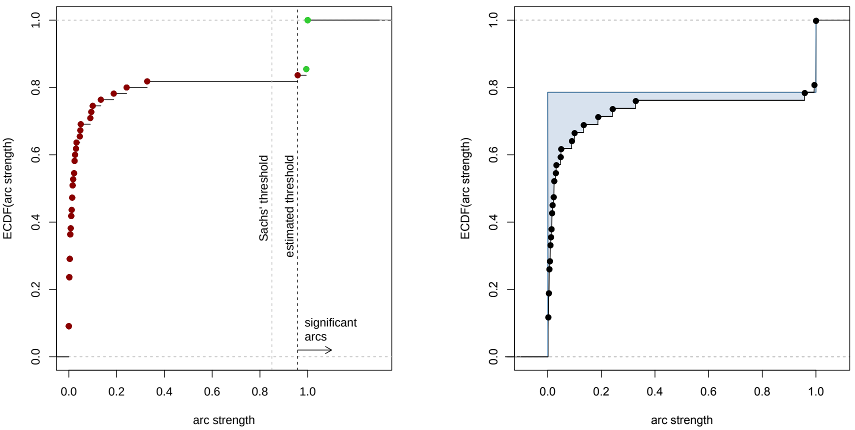
Only one local distribution changes in each step, which makes algorithm computationally efficient and easy to speed up with caching.

Learning Multiple DAGs from the Data



Searching from different starting points increases our coverage of the space of the possible DAGs; the frequency with which an arc appears is a measure of the strength of the dependence.

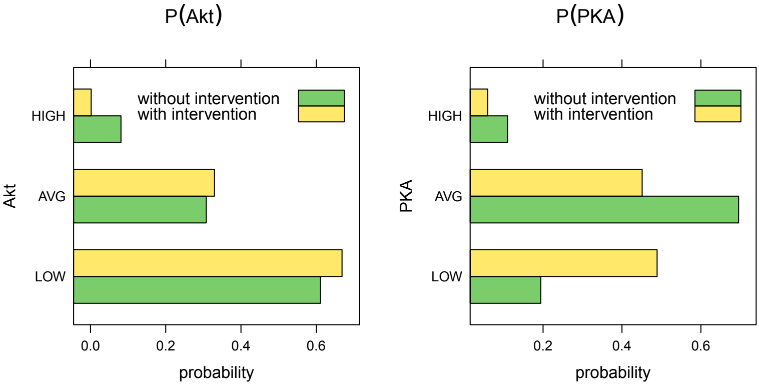
Model Averaging for DAGs



Arcs with significant strength can be identified using a threshold estimated from the data by minimising the distance from the observed ECDF and the ideal, asymptotic one (the blue area in the right panel).

Using The Protein Network to Plan Experiments

This idea goes by the name of hypothesis generation: using a statistical model to decide which follow-up experiments to perform. BNs are especially easy to use for this because they automate the computation of arbitrary events.



Conditional Probability Queries

DBNs, GBNs, CLGBNs all have exact inference algorithms to compute conditional probabilities for an event given some evidence. However, approximate inference scales better and the same algorithms can be used for all kinds of BNs. An example is likelihood weighting below.

Input: a BN $\mathcal{B} = (\mathcal{G}, \Theta)$, an query $\mathbf{Q} = \mathbf{q}$ and a set of evidence $\mathbf{E}$
Output: an approximate estimate of $\mathrm{P}(\mathbf{Q} \mid \mathbf{E}, G, \Theta)$.

1. Order the X_i according to the topological ordering in $\mathcal{G}$.
2. Set $w_{\mathbf{E}} = 0$ and $w_{\mathbf{E},\mathbf{q}} = 0$.
3. For a suitably large number of samples $\mathbf{x} = (x_1, \ldots, x_p)$:
    3.1 generate $x_{(i)}, i = 1, \ldots, p$ from $X_{(i)} \mid \Pi_{X_{(i)}}$ using the values $e_1, \ldots, e_k$ specified by the hard evidence $\mathbf{E}$ for $X_{i_1}, \ldots, X_{i_k}$.
    3.2 compute the weight $w_{\mathbf{x}} = \prod \mathrm{P}(X_{i^*} = e_* \mid \Pi_{X_{i^*}})$
    3.3 set $w_{\mathbf{E}} = w_{\mathbf{E}} + w_{\mathbf{x}}$;
    3.4 if $\mathbf{x}$ includes $\mathbf{Q} = \mathbf{q}$, set $w_{\mathbf{E},\mathbf{q}} = w_{\mathbf{E},\mathbf{q}} + w_{\mathbf{x}}$.
4. Estimate $\mathrm{P}(\mathbf{Q} \mid \mathbf{E}, G, \Theta)$ with $w_{\mathbf{E},\mathbf{q}} / w_{\mathbf{E}}$.

Case Study: Plant Genetics

Genetics and Quantitative Trait Analysis

DNA data (*e.g.* SNP markers) is routinely used in statistical genetics to understand the genetic basis of human diseases, and to breed traits of commercial interest in plants and animals. Multiparent (MAGIC) populations are ideal for the latter. Here we consider a wheat population: 721 varieties, 16K genetic markers, 7 traits. (I ran the same analysis on a rice population, 1087 varieties, 4K markers, 10 traits, with similar results.)

Phenotypic traits for plants typically include flowering time, height, yield, a number of disease scores. The goal of the analysis is to find key genetic markers controlling the traits; to indentify any causal relationships between them; and to keep a good predictive accuracy.

GSA®
Genetics Society of America

Multiple Quantitative Trait Analysis Using Bayesian Networks
Marco Scutari, *et al.*, Genetics, **198**, 129–137 (2014);
DOI: 10.1534/genetics.114.165704

Bayesian Networks in Genetics

If we have a set of traits and markers for each variety, all we need are the Markov blankets of the traits; most markers are discarded in the process. Using common sense, we can make some assumptions:

- traits can depend on markers, but not vice versa;
- dependencies between traits should follow the order of the respective measurements (*e.g.* longitudinal traits, traits measured before and after harvest, etc.);
- dependencies in multiple kinds of genetic data (*e.g.* SNP + gene expression or SNPs + methylation) should follow the central dogma of molecular biology.

Assumptions on the direction of the dependencies allow to reduce Markov blankets learning to learning the parents and the children of each trait, which is a much simpler task.

Parametric Assumptions

In the spirit of classic additive genetics models, we use a Gaussian BN. Then the local distribution of each trait T_i is a linear regression model

$$\begin{aligned} T_i &= \boldsymbol{\mu}_{T_i} + \Pi_{T_i}\boldsymbol{\beta}_{T_i} + \boldsymbol{\varepsilon}_{T_i} \\ &= \boldsymbol{\mu}_{T_i} + \underbrace{T_j\beta_{T_j} + \ldots + T_k\beta_{T_k}}_{\text{traits}} + \underbrace{G_l\beta_{G_l} + \ldots + G_m\beta_{G_m}}_{\text{markers}} + \boldsymbol{\varepsilon}_{T_i} \end{aligned}$$

and the local distribution of each marker G_i is likewise

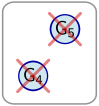
$$\begin{aligned} G_i &= \boldsymbol{\mu}_{G_i} + \Pi_{G_i}\boldsymbol{\beta}_{G_i} + \boldsymbol{\varepsilon}_{G_i} = \\ &= \boldsymbol{\mu}_{G_i} + \underbrace{G_l\beta_{G_l} + \ldots + G_m\beta_{G_m}}_{\text{markers}} + \boldsymbol{\varepsilon}_{G_i} \end{aligned}$$

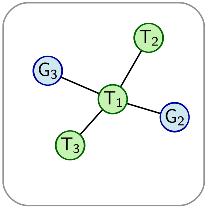
in which the regressors (Π_{T_i} or Π_{G_i}) are treated as fixed effects. Π_{T_i} can be interpreted as causal effects for the traits, Π_{G_i} as markers being in linkage disequilibrium with each other.

Learning the Bayesian Network (I)

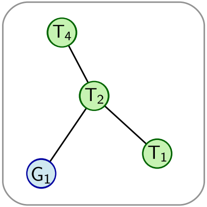
1. Feature Selection.
   1.1 Independently learn the parents and the children of each trait with the SI-HITON-PC algorithm; children can only be other traits, parents are mostly markers, spouses can be either. Both are selected using the exact Student's t test for partial correlations.
   1.2 Drop all the markers that are not parents of any trait.

Redundant markers that are not in the Markov blanket of any trait

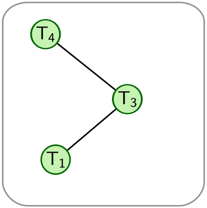




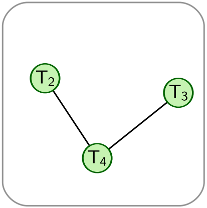
Parents and children of T_1



Parents and children of T_2



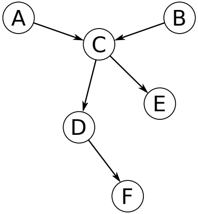
Parents and children of T_3



Parents and children of T_4

Constraint-Based Structure Learning Algorithms

CPDAG



⟸

Graphical separation

$$
\begin{aligned}
&A \perp\!\!\!\perp_G B\\
&A \perp\!\!\!\perp_G D \mid C\\
&B \perp\!\!\!\perp_G D \mid C\\
&A \perp\!\!\!\perp_G E \mid C\\
&B \perp\!\!\!\perp_G E \mid C\\
&D \perp\!\!\!\perp_G E \mid C\\
&C \perp\!\!\!\perp_G F \mid D\\
&\ldots
\end{aligned}
$$

⟸

Conditional independence tests

$$
\begin{gathered}
p_{A \perp\!\!\!\perp B} < \alpha\\
p_{A \perp\!\!\!\perp E} < \alpha\\
\vdots\\
p_{A \perp\!\!\!\perp E | C} < \alpha\\
p_{B \perp\!\!\!\perp E | D} < \alpha\\
p_{A \perp\!\!\!\perp B | C} > \alpha\\
p_{E \perp\!\!\!\perp F | D} < \alpha\\
\vdots\\
p_{F \perp\!\!\!\perp E | CD}
\end{gathered}
$$

The mapping between edges and conditional independence relationships lies at the core of BNs; therefore, one way to learn the structure of a BN is to check which such relationships hold using a suitable conditional independence test. Such an approach results in a set of conditional independence constraints that identify a single equivalence class.

The Semi-Interleaved HITON-PC Algorithm

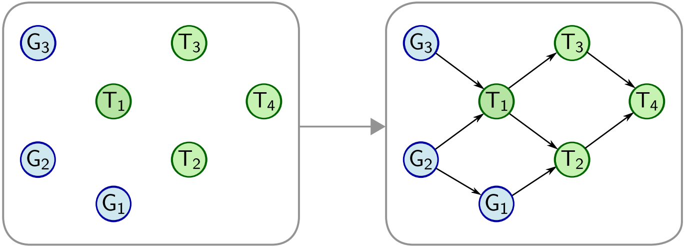
Input: each trait T_i in turn, other traits (T_j) and all markers (G_l), a significance threshold α.
Output: the set $\mathbf{CPC}$ parents and children of T_i in the BN.

1. Perform a marginal independence test between T_i and each T_j ($T_i \perp\!\!\!\perp T_j$) and G_l ($T_i \perp\!\!\!\perp G_l$) in turn.
2. Discard all T_j and G_l whose p-values are greater than α.
3. Set $\mathbf{CPC} = \{\varnothing\}$.
4. For each the T_j and G_l in order of increasing p-value:
    - 4.1 Perform a conditional independence test between T_i and T_j/G_l conditional on all possible subsets $\mathbf{Z}$ of the current $\mathbf{CPC}$ ($T_i \perp\!\!\!\perp T_j \mid \mathbf{Z} \subseteq \mathbf{CPC}$ or $T_i \perp\!\!\!\perp G_l \mid \mathbf{Z} \subseteq \mathbf{CPC}$).
    - 4.2 If the p-value is smaller than α for all subsets then $\mathbf{CPC} = \mathbf{CPC} \cup \{T_j\}$ or $\mathbf{CPC} = \mathbf{CPC} \cup \{G_l\}$.

NOTE: the algorithm is defined for a generic independence test, you can plug in any test that is appropriate for the data.

Learning the Bayesian Network (II)

2. Structure Learning. Learn the structure of the network from the nodes selected in the previous step, setting the directions of the arcs according to the assumptions above. The optimal structure can be identified with a suitable goodness-of-fit criterion such as BIC. This follows the spirit of other hybrid approaches (combining constraint-based and score-based learning) that have shown to be well-performing in the literature.

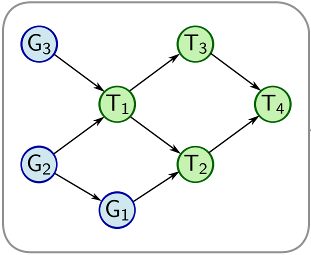


Empty network

Learned network

Learning the Bayesian Network (III)

3. Parameter Learning. Learn the parameters: each local distribution is a linear regression and the global distribution is a hierarchical linear model. Typically least squares works well because SI-HITON-PC selects sets of weakly correlated parents; ridge regression can be used otherwise.

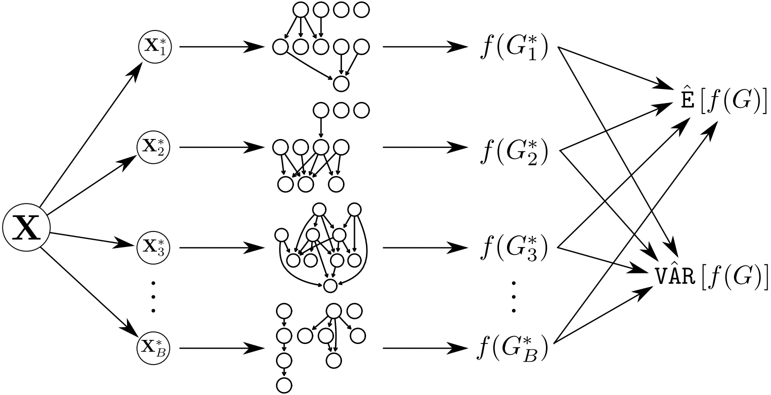


Learned network

$$
\begin{aligned}
G_1 &= \mu_{G_1} + G_2\beta_{G_2} + \varepsilon_{G_1}\\
G_2 &= \mu_{G_2} + \varepsilon_{G_2}\\
G_3 &= \mu_{G_3} + \varepsilon_{G_3}\\
T_1 &= \mu_{T_1} + G_2\beta_{G_2} + G_3\beta_{G_3} + \varepsilon_{T_1}\\
T_2 &= \mu_{T_2} + T_1\beta_{T_1} + G_1\beta_{G_1} + \varepsilon_{T_2}\\
T_3 &= \mu_{T_3} + T_1\beta_{T_1} + \varepsilon_{T_3}\\
T_4 &= \mu_{T_4} + T_2\beta_{T_2} + T_3\beta_{T_3} + \varepsilon_{T_4}
\end{aligned}
$$

Local distributions

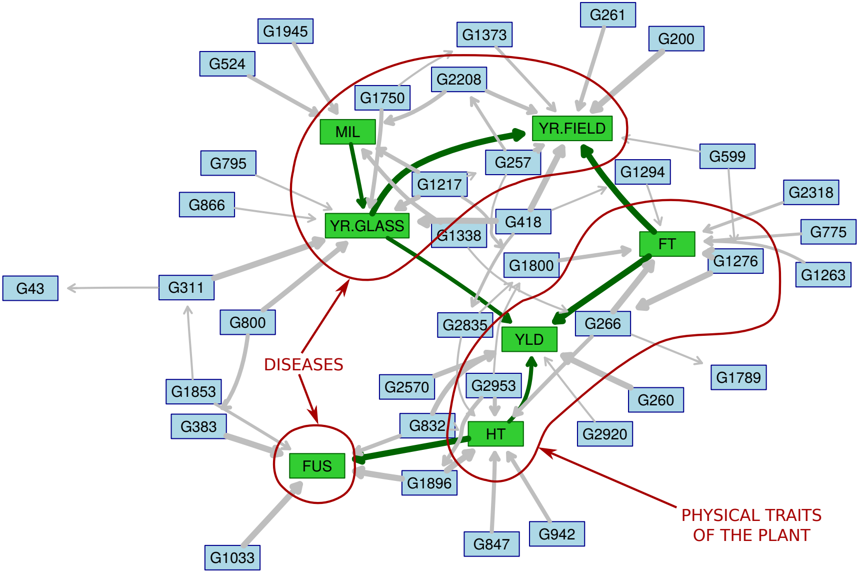
Model Averaging and Assessing Predictive Accuracy



We perform all the above in 10 runs of 10-fold cross-validation to

- assess predictive accuracy with *e.g.* predictive correlation;
- obtain a set of DAGs to produce an averaged, de-noised consensus DAG with model averaging.

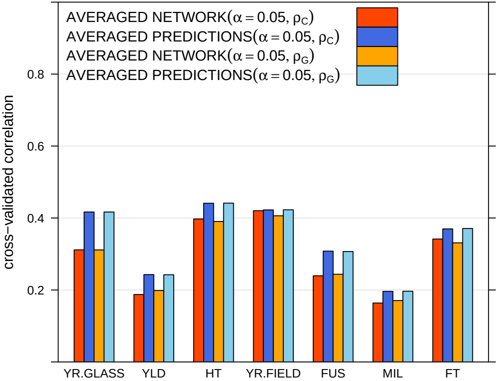
The Averaged Bayesian Network (44 nodes, 66 arcs)



Predicting Traits for New Individuals

We can predict the traits:

1. from the averaged consensus network;
2. from each of the 10×10 networks we learn during cross-validation, and average the predictions for each new individual and trait.



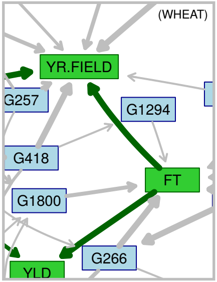
Option 2. almost always provides better accuracy than option 1.; 10×10 networks capture more information, and we have to learn them anyway. So: averaged network for interpretation, ensemble of networks for predictions.

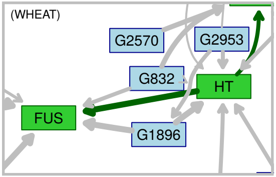
Causal Relationships Between Traits

One of the key properties of BNs is their ability to capture the direction of the causal relationships in the absence of latent confounders (the experimental design behind the data collection should take care of a number of them). Markers are causal for traits, but we do not know how traits influence each other, and we want to learn that from the data.

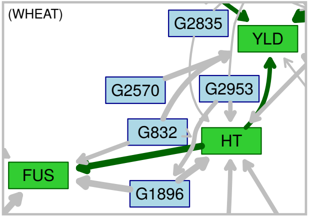
It works out because each trait will have at least one incoming arc from the markers, say $G_l \to T_j$, and then $(G_l \to) T_j \leftarrow T_k$ and $(G_l \to) T_j \to T_k$ are not probabilistically equivalent. So the network can

- suggest the direction of novel relationships;
- confirm the direction of known relationships, troubleshooting the experimental design and data collection.





Disentangling Direct, Indirect and Confounding Effects



Traits can interact in complex ways that may not be obvious when they are studied individually, but that can be explained by considering neighbouring variables in the network.
An example: in the WHEAT data, the difference in the mean YLD between the bottom and top quartiles of the FUS disease scores is +0.08.

So apparently FUS is associated with increased YLD! What we are actually measuring is the confounding effect of HT (FUS ← HT → YLD); conditional on each quartile of HT, FUS has a negative effect on YLD ranging from −0.04 to −0.06. This is reassuring since it is known that susceptibility to fusarium is positively related to HT, which in turn affects YLD.

Similarly, when a marker is shown to be associated to multiple traits, we should separate its direct and indirect effects on each of the traits. (Especially if the traits themselves are linked!)

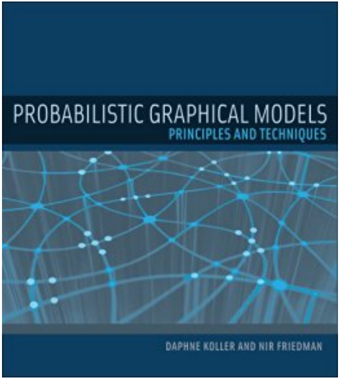
Conclusions

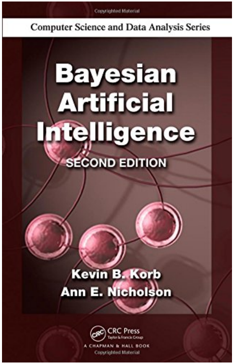
Conclusions and Remarks

- BNs provide an intuitive representation of the relationships linking heterogeneous sets of variables, which we can use for qualitative and causal reasoning.
- We can learn BNs from data while including prior knowledge as needed to improve the quality of the model.
- BNs subsume a large number of probabilistic models and thus can readily incorporate other techniques from statistics and computer science (via information theory).
- For most tasks we can start just reusing state-of-the-art, general purpose algorithms.
- Once learned, BNs provide a flexible tool for inference, while maintaining good predictive power.
- Markov blankets are a valuable tool for feature selection, and make it possible to learn just part of the BN depending on the goals of the analysis.

That's all, Thanks!

References


PROBABILISTIC GRAPHICAL MODELS
PRINCIPLES AND TECHNIQUES
DAPHNE KOLLER AND NIR FRIEDMAN



PROBABILISTIC REASONING IN INTELLIGENT SYSTEMS:
Networks of Plausible Inference
Revised & Updated
Judea Pearl
REVISED SECOND PRINTING

(Best perused as ebooks, the Koller & Friedman is $\approx 2\,{}^{1}\!/{}_{2}$ inches thick.)