Algorithms for Graphical Models (AGM)

Structure learning

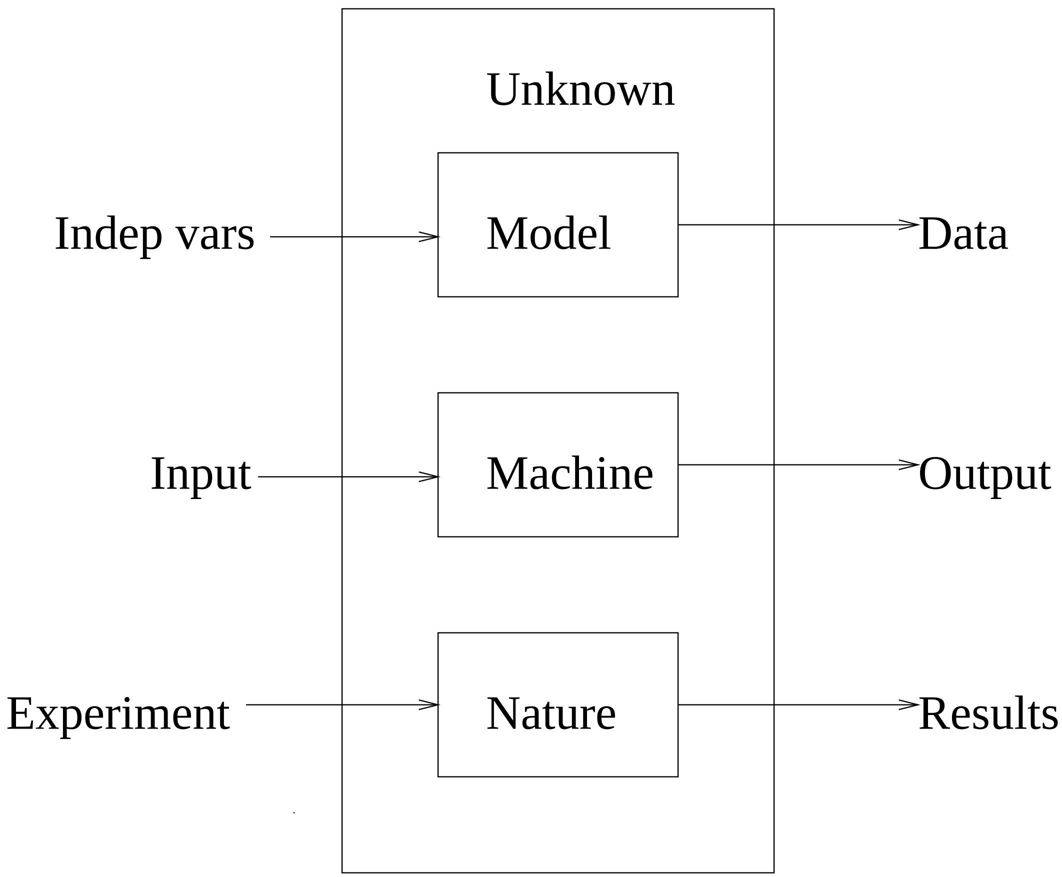
$Date: 2006/12/11 10:38:14 $

AGM-17

Overview

- Scoring BNs and decomposable models
- Essential graphs
- Searching

Statistical inference



Structure learning

Given Data (D)

Find A graphical model (M) (e.g. a BN) which 'fits' the data

The hope is that the 'learnt' model captures the true underlying conditional independence relations.

A Bayesian approach

Find a graphical model M which maximises $P(M|D)$. Bayes' theorem tells us that

$$P(M|D) = P(M)\frac{P(D|M)}{P(D)}$$

With a uniform prior, where $P(M) = \frac{1}{|\mathcal{M}|}$, what matters is the likelihood $P(D|M)$.

AGM-17

Fitted likelihoods vs marginal likelihoods

- Let's assume M is a model in the proper sense: a set of probability distributions. Let (M, θ) be a model with a particular choice of parameters.

- For example, M could be an ADG, and θ a corresponding set of CPTs so that (M, θ) is a BN.

- If $D = \mathbf{x}^1, \mathbf{x}^2, \ldots \mathbf{x}^n$ where each $\mathbf{x}^i$ is a joint instantiation (and the data is iid) then: $P(D|M, \theta) = \prod_{i=1}^{n} P(\mathbf{x}^i|M, \theta)$

- $P(D|M, \theta)$ is thus easy to compute, but what about $P(D|M)$?

Marginal likelihood

- To compute $P(D|M)$ we need to (conceptually) compute $P(D|M,\theta)$ for *all possible parameter vectors* θ and weight each summand by how likely each θ is.

- Since there are continuously infinitely many θs this is an integration: $P(D|M) = \int_\theta P(D|M,\theta)P(\theta)d\theta$

- $P(\theta)$ will be a (multi-dimensional) density function.

- This doesn't look for hopeful . . .

Dirichlets to the rescue

For BNs, suppose we assume Dirichlet distributions for each variable (and parent instantiation). Let i index variables, j index instantiations of the parents of variables and k index values of a variable, then:

$$P(D|M) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(n_{ij} + \alpha_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(n_{ijk} + \alpha_{ijk})}{\Gamma(\alpha_{ijk})}$$

where e.g. n_{ijk} is how often variable i had value k when its parents had instantiation j. The α_{ijk} are corresponding Dirichlet parameters.

Note that $P(D|M) = \prod_{i=1}^{n} \mathsf{Score}_i$.

We actually compute $\log P(D|M)$ which is a sum of scores for each variable.

AGM-17

For decomposable models

Let $\mathcal{C}$ be the cliques (hyperedges), and $\mathcal{S}$ the separators:

$$P(D|M) = \frac{\prod_{C \in \mathcal{C}} H(C)}{\prod_{S \in \mathcal{S}} H(S)}$$

where

$$H(C) = \frac{\Gamma(\alpha_C)}{\Gamma(n + \alpha_C)} \prod_{k=1}^{r_C} \frac{\Gamma(n_k + \alpha_k)}{\Gamma(\alpha_k)}$$

Here k indexes joint instantiations of the clique variables. Similarly for separators

Overfitting (and how to avoid it)

- Care has to be taken when using $P(D|M, \widehat{\theta})$, fitted likelihood, as a score.

- The more parameters M has, the easier it is to choose $\widehat{\theta}$ to fit the data: this can lead to *overfitting* which reflects chance regularities in the data.

- Marginal likelihood doesn't cherry pick: it considers *all* possible parameter values for a model and so prevents overfitting.

Learning as search

- There are too many models to score each exhaustively.
- It is thus necessary to heuristically *search* for high-scoring models.
- There are many options for searching.
- 'Greedy' search (hill-climbing) is one option: consider candidates and 'move' to whichever has the highest score. Stop if no candidate improves the score of the current model.
- The R package `deal` includes this approach.

Markov equivalence classes

- $A \to B$ and $A \leftarrow B$ are the same model: the set of all joint distributions for A and B: the saturated model.

- They have the same conditional independence relations and are thus in the same *Markov equivalence class*.

- Two BNs are Markov equivalent iff they have the same skeleton (graph ignoring arrow direction) and the same *immoralities*.

- $A \to B \to C$ is Markov equivalent to $A \leftarrow B \leftarrow C$, but not to $A \to B \leftarrow C$

Likelihood equivalence

- Unless the arrows in a BN actually represent causal relations,
- then it is a mistake to give different scores to BNs in the same Markov equivalence class.
- A scoring function that avoids this mistake is *likelihood equivalent*
- Most choices of Dirichlet priors are *not* likelihood equivalent.

Essential graphs

- An *essential graph* is a unique representative of a Markov equivalence class.

- For a given BN, it is possible to generate its essential graph.

- Searching (and scoring) can then take place in the space of essential graphs rather than ADGs.

- See (Chickering, 2002), for example.

Bayesian model averaging

- Returning a single 'best guess' for the true BN/HM fails to reflect the inherent uncertainty in learning.

- An alternative is to produce a posterior distribution over models,

- or at least an approximation to such a distribution.

MCMC over models

- A common approach to Bayesian model averaging is to use MCMC to approximately sample from the posterior.

- With the Metropolis-Hastings MCMC algorithm a new model is 'proposed' using a probabilistic mutation operator

- With a certain probability the proposed model is accepted, otherwise it is rejected and the chain doesn't change state.

- (Roughly) if the new model is more probable it is accepted, otherwise there's still a chance of acceptance if it is not much less probable.

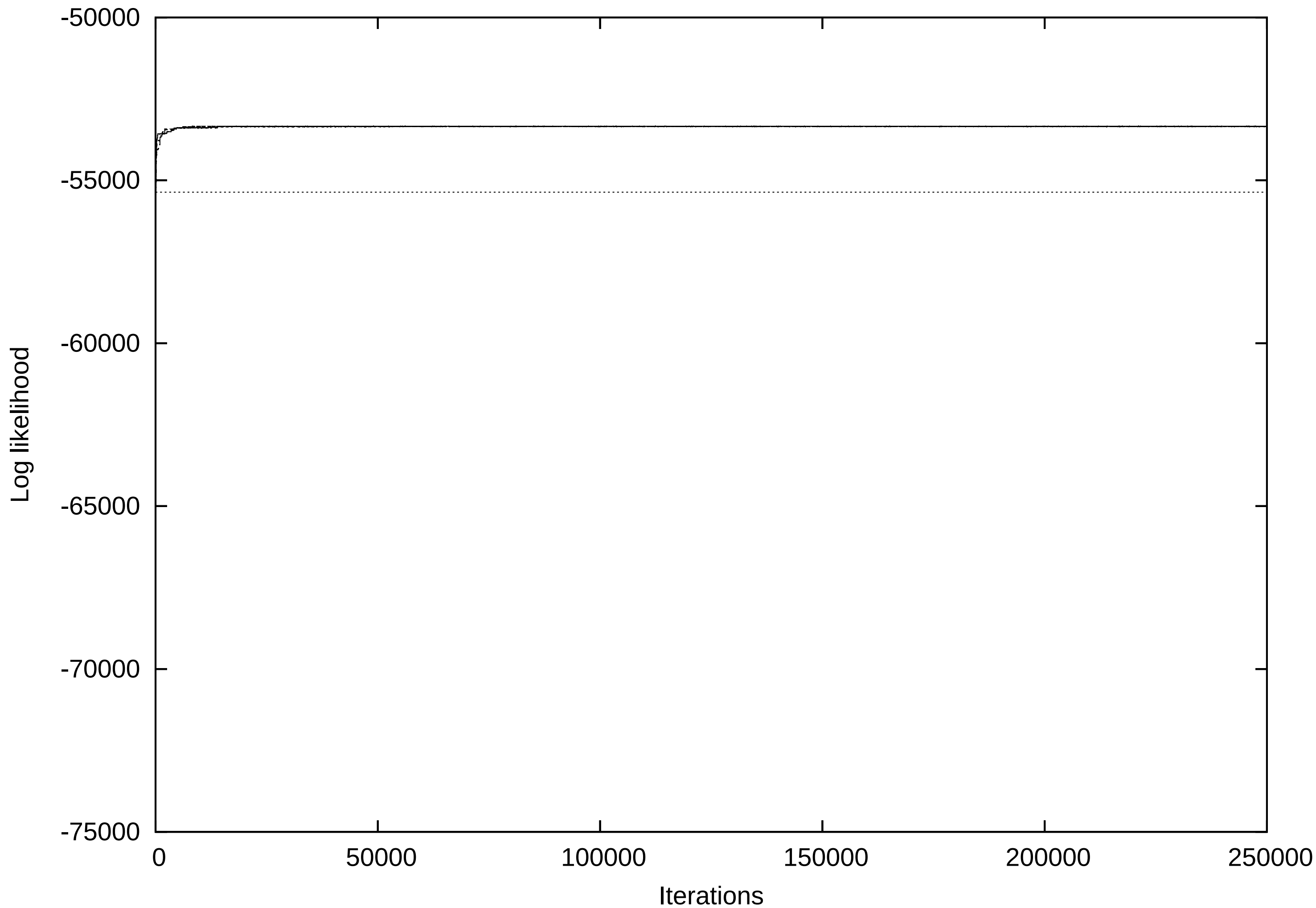
Tracking MCMC

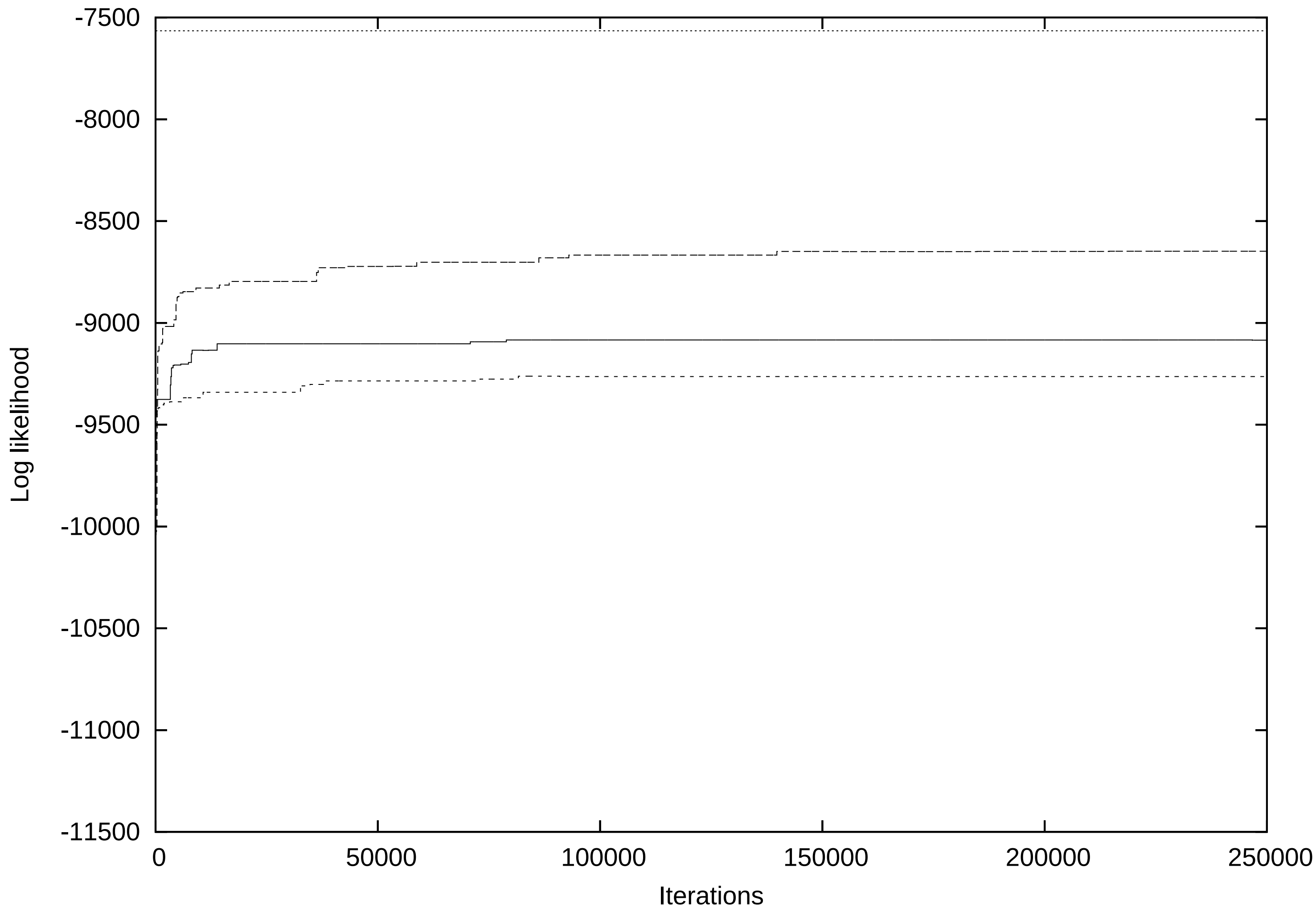
- It is often illuminating to track $\log P(D|M)$ as MCMC (or indeed a search algorithm) progresses.

- Here's some *log-likelihood trajectories* from my own research (with Nicos Angelopoulos)

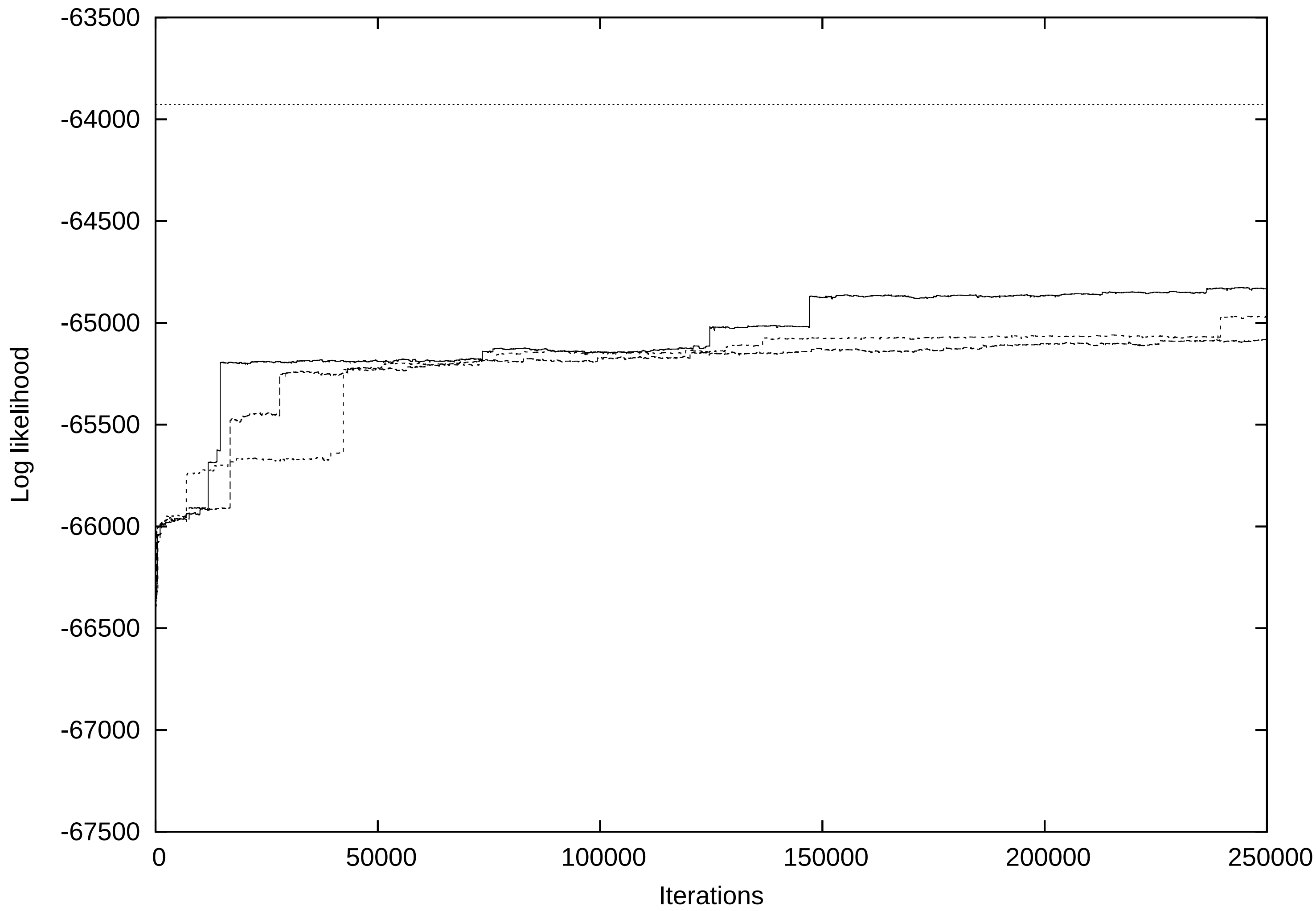
- In some cases, there is clear evidence of the MCMC getting stuck :-(

- We used artificial data sampled from a 'true' BN. The dotted line is the log-likelihood of this BN.


-50000
-55000
-60000
-65000
-70000
-75000
Log likelihood
0
50000
100000
150000
200000
250000
Iterations



-7500
-8000
-8500
-9000
-9500
-10000
-10500
-11000
-11500
Log likelihood
0
50000
100000
150000
200000
250000
Iterations


-63500
-64000
-64500
-65000
-65500
-66000
-66500
-67000
-67500
Log likelihood
0
50000
100000
150000
200000
250000
Iterations