Scaled Conjugate Gradients for Maximum Likelihood: An Empirical Comparison with the EM Algorithm

Kristian Kersting and Niels Landwehr
Institute for Computer Science, Machine Learning Lab
Albert-Ludwigs-University, Georges-Köhler-Allee, Gebäude 079,
D-79085 Freiburg i. Brg., Germany
{kersting,landwehr}@informatik.uni-freiburg.de

Abstract

To learn Bayesian networks, one must estimate the parameters of the network from the data. EM (Expectation-Maximization) and gradient-based algorithms are the two best known techniques to estimate these parameters. Although the theoretical properties of these two frameworks are well-studied, it remains an open question as to when and whether EM is to be preferred over gradients. In this paper, we answer this question empirically. More specifically, we first adapt scaled conjugate gradients well-known from neural network learning. This accelerated conjugate gradient avoids the time consuming line search of more traditional methods. Secondly, we empirically compare scaled conjugate gradients with EM. The experiments show that accelerated conjugate gradients are competitive with EM. Although, in general EM is the domain independent method of choice, gradient-based methods can be superior.

1 Introduction

Bayesian networks (Pearl, 1991) are one of the most important, efficient and elegant frameworks for representing and reasoning with probabilistic models. They specify joint probability distributions over finite sets of random variables, and have been applied to many real-world problems in diagnosis, forecasting, automated vision, sensor fusion and manufacturing control. Over the past years, there has been much interest in the problem of learning Bayesian networks from data. Here, the problem of *parameter estimation* is a fundamental task not only because of the inability of humans to reliably estimate the parameters, but also because it forms the basis for the overall learning problem (Heckerman, 1995). A classical method is *maximum likelihood (ML)* parameter estimation, i.e. one selects those parameters maximizing the likelihood which is the probability of the observed data as a function of the unknown parameters. Unfortunately in many real-world domains, the data cases available are incomplete, i.e. some values may not be observed. In presence of missing data, the ML estimate typically cannot be written in closed form. It is a numerical optimization problem, and all known algorithms involve nonlinear optimization and multiple calls to a Bayesian network inference as subroutines. The most prominent techniques within Bayesian networks are the EM (Expectation and Maximization) algorithm and gradient-based approaches.

The comparison between EM and (advanced) gradient techniques like conjugate gradient (CG) is not well understood. Both methods perform a greedy local search which is guaranteed to converge to stationary points. They both exploit expected sufficient statistics as their primary computation step. However, there are important differences. The EM is easier to implement, converges much faster than simple gradient, and is somewhat less sensitive to starting points. CGs estimate the step size (see below) with a line search involving several additional Bayesian network inferences com-

pared to EM. But, gradients are more flexible than EM, as they easily allow e.g. to learn non-multinomial parameterizations using the chain rule for derivatives (J.Binder et al., 1997) or to choose other scoring functions than the likelihood (Jensen, 1999). The EM algorithm may slowly converge, but can be sped up near the maximum using gradient-based approaches (Thiesson, 1995; McLachlan and Krishnan, 1997; Bauer et al., 1997; Ortiz and Pack Kaelbling, 1999).

In this paper, we report on an experimental comparison between EM and (accelerated) CGs to ML parameter estimation within Bayesian networks over finite random variables. In doing so, we follow Ortiz and Kaelbling's (Ortiz and Pack Kaelbling, 1999) suggestion for such a comparison in the context of (discrete) Bayesian network. To overcome the expensive line search, we adapted *scaled conjugate gradients* (SCG) (Møller, 1993a). This type of accelerated CGs are novel in the context of Bayesian networks. Other work investigated approximated line searches only to accelerate EM (Thiesson, 1995; Ortiz and Pack Kaelbling, 1999; Bauer et al., 1997). From the experimental results, we will argue that SCGs are competitive with EM. (Accelerate) gradient methods can be superior depending on properties of the domain such as the fraction of latent parameters and the prior information encoded the network structure, though EM seems to be the best *domain independent* choice.

We proceed as follows. After briefly introducing Bayesian networks in Section 2, Section 3 reviews the EM algorithm, basic (conjugate) gradients, and scaled conjugate gradients. We experimentally compare these approaches in Section 4. Before concluding, we discuss related work in Section 5.

2 Bayesian networks

Throughout the paper, we will use X to denote a random variable, x a state and $\mathbf{X}$ (resp. $\mathbf{x}$) a vector of variables (resp. states). A *Bayesian network* (Pearl, 1991) represents the joint probability distribution $\mathbf{P}(X_1, \ldots, X_n)$ over a set $\mathbf{X} = \{X_1, \ldots, X_n\}$ of random variables. In this paper, we restrict each X_i to have a finite set $x_i^1, \ldots, x_i^{k_i}$ of possible states. A Bayesian network is an augmented, acyclic graph, where each node corresponds to a variable X_i and each edge indicates a direct influence among the variables. We denote the parents of X_i in a graph-theoretical sense by $\mathbf{Pa}_i$. The family of X_i is $\mathbf{Fa}_i := \{X_i\} \cup \mathbf{Pa}_i$. To each node X_i, a conditional probability table* (CPT) is associated specifying the distribution $\mathbf{P}(X_i \mid \mathbf{Pa}_i)$. The table entries are

$$\theta_{ijk} = P(X_i = x_i^k \mid \mathbf{Pa}_i = \mathbf{pa}_i^j).$$

The network stipulates the assumption that each node X_i in the graph is conditionally independent of any subset $\mathbf{A}$ of nodes that are not descendants of X_i given a joint state of its parents. Thus, the joint probability distribution over $\mathbf{X}$ factors to

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbf{P}(X_i \mid \mathbf{Pa}_i).$$

In the rest of the paper, we will represent a Bayesian network by the vector θ of all θ_{ijk}'s.

3 Maximum Likelihood Estimation

Here, we will show how to estimate the maximum likelihood (ML) parameters of a given Bayesian network. Let $\mathbf{D} = \{\mathbf{d_1}, \ldots, \mathbf{d_N}\}$ be a set of data cases, i.e. partial joint states over $\mathbf{X}$. We assume that the data cases are independently sampled from identical distributions (iid). The likelihood $L(\mathbf{D}, \theta) := P(\mathbf{D} \mid \theta)$ simplifies because of the iid assumption to

$$L(\mathbf{D}, \theta) = \prod_{l=1}^{N} P(\mathbf{d}_l \mid \theta).$$

Thus, the search space $\mathcal{H}$ is spanned by the space over the possible values of θ and, we seek to find $\theta^* = \max_{\theta \in \mathcal{H}} L(\mathbf{D}, \theta)$. Due to the monotonicity of the logarithm, this yields

$$\theta^* = \max_{\theta \in \mathcal{H}} \sum_{l=1}^{N} \log P(\mathbf{d}_l \mid \theta).$$

*There are more sophisticated representation, but we will not discuss them here.

In the presence of missing data, the ML estimate typically cannot be written in closed form, and iterative optimization schemes like the EM or conjugate gradient algorithms are needed.

3.1 EM Algorithm

The Expectation-Maximization algorithm (Lauritzen, 1995) is a standard tool for ML estimation in the presence of incomplete data. Its basic idea is that if we know the values for all nodes, learning would be easy. It iteratively performs two steps: First, based on the current parameters θ^n and the observed data the expected values are computed using an inference algorithm. Then it treats the expected values as though they were observed, computing new parameters θ^{n+1}. Lauritzen (Lauritzen, 1995) showed that this idea leads to $\theta^*_{ijk} = \bar{n}(\mathbf{fa}_i^{kj} \mid \mathbf{D})/\bar{n}(\mathbf{pa}_i^j \mid \mathbf{D})$. The term

$$\bar{n}(\mathbf{a}) = \sum_{l=1}^{N} P(\mathbf{a} \mid \mathbf{d}_l, \theta^n)$$

is called the *expected counts*. To compute $P(\mathbf{a} \mid \mathbf{d}_l, \theta^n)$ a Bayesian network inference engine can be used.

3.2 Gradient-Ascent Learning

Gradient ascent, also known as *hill climbing*, is a classical method for finding a maximum of a (scoring) function. It iteratively performs two steps. First, it computes the *gradient* vector ∇_θ of partial derivatives of the scoring function with respect to the parameters of a Bayesian network at a given point $\theta \in \mathcal{H}$. Then it takes a small step in the direction of the gradient to the point $\theta + \delta\nabla_\theta$ where δ is the step-size parameter. The algorithm will converge to a local maximum for small enough δ (cf. (Luenberger, 1984)). Thus, we have to compute the partial derivatives of $P(\mathbf{D} \mid \theta)$ w.r.t. parameters θ_{ijk}. According to Binder *et al.* (J.Binder et al., 1997), they are given by

$$\frac{\partial log P(\mathbf{D} \mid \theta)}{\partial \theta_{ijk}} = \sum_{l=1}^{N} \frac{P(\mathbf{fa}_i^{kj} \mid \mathbf{d}_l, \theta)}{\theta_{ijk}} \qquad (1)$$

Again, $P(\mathbf{fa}_i^{kj} \mid \mathbf{d}_l, \theta)$ can be computed using an inference engine.

3.2.1 Constraint Satisfaction

Note that in the problem at hand, the described method has to be modified to take into account the constraint that the parameter vector θ consists of probability values, i.e. $\theta_{ijk} \in [0, 1]$ and $\sum_j \theta_{ijk} = 1$. There are two (J.Binder et al., 1997) ways to enforce this. (1) Projecting the gradient onto the constraint surface, and (2) reparameterizing the problem so that the new parameters automatically respect the constraints on θ_{ijk} no matter what their values are. We choose the latter approach as the reparameterized problem is fully unconstrained. More precisely, we define the parameters $\beta_{ijk} \in \mathbb{R}$ such that

$$\theta_{ijk} = \frac{\beta_{ijk}^2}{\sum_l \beta_{ilk}^2}.$$

This enforce the constraints given above, and a local maximum w.r.t. to β_{ijk} is also a local maximum w.r.t. θ_{ijk}, and vice versa. The gradient w.r.t the β_{ijk}'s can be found by computing the gradient w.r.t to the θ_{ijk} and then deriving the gradient w.r.t. the β_{ijk} using the chain rule, see (J.Binder et al., 1997).

3.3 (Scaled) Conjugate Gradients

The simple gradient ascent algorithm uses a fixed step size δ when following the gradient. As a consequence, it often converges very slowly. Though it will converge to a (local) maximum for small enough δ, it is not a priori clear how to choose δ.

Instead, it would be better to perform a series of *line searches* to choose δ, i.e. to do a one dimensional iterative search for δ in the direction of ∇_{θ_i} maximizing $L(\mathbf{D}, \theta + \delta \cdot \nabla_{\theta_i})$. The problem with the resulting algorithm is, that a maximization in one direction could spoil past maximizations. This problem is solved in *conjugate gradient* (CG) methods (see e.g (Luenberger, 1984; Nocedal, 1992)) by computing so-called conjugate directions $h_0, h_1, h_2, \ldots$, which are not interfering, and estimating the step size along these directions with line searches. But still there are several drawbacks of doing a

line search. First, it introduces new problem-dependent parameters such as stopping criterion. Second, the line search involves several likelihood evaluations, i.e. network inferences which are known to be NP-hard (Cooper, 1990). Thus, the line search dominates the computational costs of CGs resulting in a serious disadvantage compared to the EM which does one inference per iteration.

Therefore, it is not surprising that researchers in the UAI community used inexact line search to reduce the complexity (Thiesson, 1995; Ortiz and Pack Kaelbling, 1999; Bauer et al., 1997) when accelerating EM. However, the use of both exact and inexact line searches within CGs requires careful considerations (Ortiz and Pack Kaelbling, 1999; Nocedal, 1992). Thus, we decided to use a variant of CGs called *scaled conjugate gradients* (SCG) due to Møller (Møller, 1993a) which led to significant speed up in the context of learning neural networks while preserving accuracy.

Scaled conjugate gradients substitute the line search by a scaling of the step size δ depending on success in error reduction and goodness of a quadratic approximation of the likelihood. The approximation costs one additional inference compared to EM. Due to lack of space, we will only sketch the algorithm, details can be found in (Møller, 1993a), and for the sake of simplicity, we will illustrate the idea minimizing a general function $f : \mathbb{R}^k \mapsto \mathbb{R}$. Substituting f by $-f$ yields the corresponding maximization. Let

$$\tilde{f}_{\mathbf{w}}(\mathbf{x}) = f(\mathbf{w}) + f'(\mathbf{w})^T \cdot \mathbf{x} + \frac{1}{2}\mathbf{x}^T \cdot f''(\mathbf{w}) \cdot \mathbf{x}$$

be the quadratic approximation of f at $\mathbf{w}$, $\mathbf{v}_j$ the last parameter estimation, and h_{j+1} the current conjugate direction. Then, the step size δ_{j+1} minimizing $\tilde{f}_{\mathbf{v}_j}(\mathbf{v}_j + \delta \cdot h_{j+1})$ is given by

$$\delta_{j+1} = \frac{-h_{j+1}^T \cdot \tilde{f}'_{\mathbf{v}_j}(\mathbf{v}_j)}{h_{j+1}^T \cdot f''(\mathbf{v}_j) \cdot h_{j+1}}, \qquad (2)$$

provided that the Hessian $f''(\mathbf{v}_j)$ is positive definite. Assuming this, the next parameter estimation $\mathbf{v}_{j+1}$ would be $\tilde{f}_{\mathbf{v}_j}(\mathbf{v}_j + \delta_{j+1} \cdot h_{j+1})$. However, the computation of the second order term $f''(\mathbf{v}_j) \cdot h_{j+1}$ in (2) is expensive. It is approximated by

$$s_{j+1} = \frac{f'(\mathbf{v}_j + \sigma_{j+1} \cdot h_{j+1}) - f'(\mathbf{v}_j)}{\sigma_{j+1}} \qquad (3)$$

for some small σ_{j+1}. As Møller pointed out, the approximation in equation (3) is poor if the Hessian becomes indefinite, and the search might fail. Therefore, one checks whether $f'(\mathbf{v}_j)$ is positive definite or not by evaluating the sign of $\beta_{j+1} := h_{j+1}^T \cdot s_{j+1}$. If it is indefinite, i.e. $\beta_{j+1} \leq 0$, a scalar λ_{j+1} is introduced into equation (3) to regulate the indefiniteness of $f''(\mathbf{v}_j)$ by adding $\lambda_{j+1} \cdot h_{j+1}$ to the right-hand side of equation (3). By increasing λ_{j+1} one can ensure $\beta_{j+1} > 0$. Using a simple heuristic for rising λ_{j+1}, Møller showed that the step size δ_{j+1} is given by

$$\delta_{j+1} = \frac{-h_{j+1}^T \cdot f'(\mathbf{v}_j)}{h_{j+1}^T \cdot s_{j+1} + \lambda_{j+1} \cdot \mid h_{j+1} \mid^2}.$$

The values of λ_{j+1} directly scale the step size δ_{j+1} in the way, that the bigger the λ_{j+1}, the smaller the step size δ_{j+1}. Again, only the gradient needs to be computed, i.e. SCGs can be implemented using a Bayesian network inference engine

4 Experiments

In the experiments described below, we implemented all three algorithms using Netica API (`http://www.norsys.com`) for Bayesian network inference. We adapted the conjugate gradient (CG) described in (Press et al., 1993) to fullfil the constraints described above. Based on this code, we adapted the scaled conjugate gradient (SCG) as implemented in Bishop and Nabney's Netlab library (`http://www.ncrg.aston.ac.uk/netlab/`, see also (Bishop, 1995)) with an upper bound on the scale parameter of $2 \cdot 10^6$. To see how sensitive the methods to priors are, we also implemented each of them using BDeu priors (Heckerman et al., 1995). Data were generated from four Bayesian networks whose main characteristics are described in Table 1. From each target network, we generated a test set of 10000

Name	Description
Alarm	Well-known benchmark network for the ICU ventilator management. It consists of 37 nodes and 752 parameters (Beinlich et al., 1989). No latent variables.
Insurance	Well-known benchmark network for estimating the expected claim costs for a car insurance policyholder. It consists of 27 nodes (12 latent) and over 1419 parameters (J.Binder et al., 1997).
3-1-3, **5-3-5**	Two artificial networks with a feed-forward architecture known from neural networks (J.Binder et al., 1997). There are three fully connected layers of $3 \times 2 \times 3$ (resp. $5 \times 3 \times 5$) nodes. Nodes of the first and third layers have 3 possible states, nodes of the second 2.

Table 1: Description of the networks used in the experiments.

data cases, and training sets of 100, 200, 500 and 1000 data cases with a fraction of 0, 0.1, 0.2 and 0.3 of missing at random values of the observed nodes. The values of latent nodes were never observed. For each training set, five different random initial sets of parameters were tried. We ran each algorithm on each data set starting from each of the generated initial set of parameters. We used a simple, typical stopping criterion. We stopped when a limit of 200 iterations was exceeded or a change in average log-likelihood per case was less than 10^{-5} from one iteration to the next. We chose this low threshold because EM could be very slow if high accuracy of the estimate is necessary, whereas CGs work well under these conditions. Although discrete Bayesian network are said to be robust against small changes, there exist situations where small variations can lead to significant changes in computed queries (Chan and Darwiche, 2002). All learned models were evaluated on the test set using a standard measure, the *normalized-loss*

$$\frac{1}{M}\sum_{l=1}^{M}(\log P^*(\mathbf{d}_l) - \log P(\mathbf{d}_l))$$

where P^* is the probability of the generating distribution. Normalized-loss measures the additional penalty for using an approximation instead of the true model, and is also a cross-entropy estimate. The closer the normalized-loss is to zero, the better. We report on the average performance and running time. The latter is dominated by the Bayesian network inference, because except for the likelihood evaluations all computations are linear. Therefore, we consider *EM-equivalent* iterations, i.e. likelihood evaluations. Each SCG iteration consists of two EM-equivalent iterations.

4.1 Results

The results on the normalized-loss measure are shown in Table 2. We omit CG from further investigations because it did not terminated in reasonable time [†] on **Insurance** and **Alarm**. However, on **3-1-3** and **5-3-5**, it showed the expected behaviour. Considering only the normalized-loss, it reached normalized-losses slightly closer to zero than EM and SCG (mean difference around 0.001, priors). The number of iterations on **3-1-3** was on average two times higher than for EM, on **5-3-5** they were on average twice lower. However, the number of EM-equivalent iterations was in all cases much higher then for EM and SCG. Furthermore, CG was sensitive to the initial set of parameters.

EM and SCG reached similar normalized-losses. As expected, a higher number of data cases lead to a lower normalized-loss. Similarly, priors lead to a lower normalized-loss. The SCG was less sensitive to priors than EM. Both were insensitive to the initial set of parameters. The EM tend to have a slightly lower variation, but a *one-tailed, paired sampled t test* shows that only a 0.045 difference in mean of EM and SCG (with priors, $p = 0.05$) was significant. EM reached normalized-losses slightly closer to zero in all domains but **5-3-5** where a 0.02 difference in mean was significant ($p = 0.05$) favoring SCG. However, the t test takes all restarts into account, but in practice, one usually selects for

[†] No run was finished after one week on a Pentium III, 450 MHz.

Domain / # hidden (in %)	Meth.	Number of data cases			
		100	200	500	1000
Insurance	EM	0.99 ± 0.04(12.20 ± 2.03)	**0.64 ± 0.03**(9.18 ± 0.48)	0.37 ± 0.02(2.56 ± 0.41)	0.20 ± 0.01(0.72 ± 0.18)
0.0	SCG	**1.82 ± 1.76**(**8.40 ± 0.74**)	0.62 ± 0.02(**3.92 ± 0.76**)	**0.33 ± 0.03**(**1.33 ± 0.29**)	**0.51 ± 0.60**(**0.56 ± 0.19**)
0.1	EM	1.08 ± 0.05(20.98 ± 2.55)	0.68 ± 0.04(12.25 ± 2.15)	0.37 ± 0.02(3.03 ± 0.75)	**0.23 ± 0.01**(0.90 ± 0.15)
	SCG	**1.14 ± 0.35**(**8.10 ± 1.42**)	**0.69 ± 0.11**(**4.22 ± 1.19**)	**0.89 ± 0.67**(**1.76 ± 0.28**)	0.30 ± 0.08(**0.76 ± 0.04**)
0.2	EM	**1.07 ± 0.03**(22.94 ± 2.20)	0.76 ± 0.03(14.98 ± 3.01)	0.45 ± 0.02(4.60 ± 0.67)	0.25 ± 0.02(0.97 ± 0.19)
	SCG	1.35 ± 0.48(**8.85 ± 2.62**)	**0.76 ± 0.06**(**5.80 ± 0.29**)	**0.44 ± 0.06**(**2.64 ± 0.49**)	**0.37 ± 0.24**(**1.06 ± 0.19**)
0.3	EM	1.35 ± 0.04(28.78 ± 4.44)	0.94 ± 0.02(14.87 ± 1.63)	**0.52 ± 0.03**(6.11 ± 0.75)	0.29 ± 0.01(**1.47 ± 0.16**)
	SCG	**1.59 ± 0.71**(**12.26 ± 0.75**)	**1.25 ± 0.39**(**7.04 ± 1.17**)	0.62 ± 0.13(**3.62 ± 0.70**)	**0.38 ± 0.09**(1.43 ± 0.08)
Alarm	EM	0.81 ± 0.00(**−0.66 ± 0.01**)	**0.62 ± 0.00**(**−0.44 ± 0.02**)	**0.29 ± 0.00**(**−0.15 ± 0.01**)	0.16 ± 0.00(**−0.03 ± 0.00**)
0.0	SCG	**1.01 ± 0.25**(3.79 ± 0.18)	0.94 ± 0.37(2.41 ± 0.20)	0.33 ± 0.02(1.01 ± 0.05)	**0.17 ± 0.00**(0.53 ± 0.04)
0.1	EM	1.28 ± 0.00(13.34 ± 1.38)	0.68 ± 0.00(7.01 ± 0.87)	0.31 ± 0.00(3.30 ± 0.08)	0.18 ± 0.00(1.42 ± 0.07)
	SCG	**1.81 ± 0.74**(**5.94 ± 0.21**)	**0.87 ± 0.27**(**2.53 ± 0.13**)	**0.31 ± 0.00**(**1.04 ± 0.05**)	**0.19 ± 0.00**(**0.54 ± 0.02**)
0.2	EM	**1.10 ± 0.00**(18.47 ± 1.43)	0.64 ± 0.00(8.91 ± 0.76)	0.31 ± 0.00(2.75 ± 0.43)	**0.24 ± 0.00**(1.61 ± 0.06)
	SCG	1.31 ± 0.19(**6.06 ± 0.26**)	**1.25 ± 0.80**(**3.18 ± 0.09**)	**0.31 ± 0.00**(**1.09 ± 0.07**)	0.24 ± 0.01(**0.64 ± 0.03**)
0.3	EM	1.22 ± 0.00(16.63 ± 1.31)	**0.65 ± 0.00**(11.41 ± 0.79)	**0.41 ± 0.00**(3.98 ± 0.13)	**0.21 ± 0.00**(1.68 ± 0.05)
	SCG	**1.28 ± 0.08**(**6.76 ± 0.40**)	0.66 ± 0.01(**4.07 ± 0.27**)	1.12 ± 0.96(**1.70 ± 0.09**)	0.22 ± 0.00(**0.76 ± 0.04**)
3-1-3	EM	**0.15 ± 0.01**(**0.30 ± 0.26**)	0.07 ± 0.00(0.12 ± 0.00)	**0.03 ± 0.00**(0.03 ± 0.00)	0.02 ± 0.00(0.02 ± 0.00)
0.0	SCG	0.16 ± 0.02(0.57 ± 0.04)	**0.07 ± 0.00**(**0.12 ± 0.03**)	0.03 ± 0.00(**0.03 ± 0.00**)	**0.02 ± 0.00**(**0.02 ± 0.00**)
	CG	0.17 ± 0.04(0.56 ± 0.08)	0.07 ± 0.00(0.10 ± 0.00)	0.04 ± 0.02(0.03 ± 0.00)	0.02 ± 0.00(0.02 ± 0.00)
0.1	EM	0.13 ± 0.02(**0.46 ± 0.35**)	0.13 ± 0.03(**0.18 ± 0.02**)	0.05 ± 0.00(**0.06 ± 0.00**)	**0.02 ± 0.00**(**0.02 ± 0.00**)
	SCG	**0.13 ± 0.01**(0.31 ± 0.05)	**0.11 ± 0.01**(0.17 ± 0.02)	**0.06 ± 0.03**(0.06 ± 0.00)	0.02 ± 0.00(0.02 ± 0.00)
	CG	0.13 ± 0.02(0.31 ± 0.04)	0.11 ± 0.02(0.17 ± 0.02)	0.05 ± 0.00(0.06 ± 0.00)	0.02 ± 0.01(0.02 ± 0.00)
0.2	EM	**0.20 ± 0.01**(2.08 ± 0.90)	**0.16 ± 0.04**(0.28 ± 0.08)	**0.05 ± 0.00**(**0.07 ± 0.01**)	**0.03 ± 0.00**(**0.04 ± 0.00**)
	SCG	0.25 ± 0.05(**0.86 ± 0.31**)	0.10 ± 0.01(**0.20 ± 0.08**)	0.05 ± 0.00(0.11 ± 0.03)	0.04 ± 0.03(0.04 ± 0.00)
	CG	0.20 ± 0.03(0.79 ± 0.26)	0.10 ± 0.01(0.24 ± 0.09)	0.06 ± 0.00(0.11 ± 0.02)	0.03 ± 0.00(0.04 ± 0.00)
0.3	EM	**0.15 ± 0.00**(1.47 ± 0.18)	**0.16 ± 0.01**(0.90 ± 0.16)	**0.03 ± 0.00**(0.06 ± 0.00)	0.05 ± 0.05(0.06 ± 0.05)
	SCG	0.17 ± 0.02(**0.77 ± 0.28**)	0.17 ± 0.01(**0.68 ± 0.22**)	0.04 ± 0.03(**0.06 ± 0.00**)	**0.02 ± 0.00**(**0.02 ± 0.00**)
	CG	0.16 ± 0.02(0.79 ± 0.31)	0.18 ± 0.04(0.72 ± 0.16)	0.03 ± 0.00(0.06 ± 0.00)	0.02 ± 0.00(0.03 ± 0.00)
5-3-5	EM	0.23 ± 0.01(17.36 ± 0.92)	0.22 ± 0.01(8.38 ± 1.23)	0.15 ± 0.01(1.86 ± 0.41)	0.11 ± 0.00(0.43 ± 0.08)
0.0	SCG	**0.19 ± 0.03**(**6.00 ± 0.88**)	**0.21 ± 0.01**(**2.96 ± 1.32**)	**0.13 ± 0.02**(**0.94 ± 0.09**)	**0.10 ± 0.01**(**0.28 ± 0.02**)
	CG	0.14 ± 0.07(5.15 ± 0.84)	0.13 ± 0.07(3.28 ± 0.61)	0.09 ± 0.05(0.84 ± 0.12)	0.06 ± 0.03(0.29 ± 0.02)
0.1	EM	0.19 ± 0.01(19.37 ± 2.14)	0.21 ± 0.01(11.76 ± 2.05)	0.18 ± 0.00(3.03 ± 0.34)	0.13 ± 0.01(1.03 ± 0.22)
	SCG	**0.14 ± 0.04**(**7.52 ± 1.59**)	**0.20 ± 0.01**(**4.43 ± 0.66**)	**0.15 ± 0.02**(**1.47 ± 0.34**)	**0.09 ± 0.02**(**0.47 ± 0.05**)
	CG	0.12 ± 0.06(6.50 ± 0.65)	0.13 ± 0.07(4.08 ± 0.35)	0.11 ± 0.06(1.34 ± 0.07)	0.08 ± 0.04(0.44 ± 0.06)
0.2	EM	0.16 ± 0.01(24.81 ± 1.67)	0.18 ± 0.01(13.88 ± 2.74)	0.16 ± 0.01(4.09 ± 0.63)	0.14 ± 0.01(1.04 ± 0.16)
	SCG	**0.12 ± 0.03**(**7.64 ± 0.93**)	**0.14 ± 0.03**(**5.58 ± 0.66**)	**0.14 ± 0.02**(**1.56 ± 0.61**)	**0.13 ± 0.01**(**0.72 ± 0.07**)
	CG	0.09 ± 0.05(8.08 ± 1.00)	0.10 ± 0.05(4.50 ± 0.50)	0.09 ± 0.05(1.62 ± 0.24)	0.08 ± 0.04(0.72 ± 0.09)
0.3	EM	0.25 ± 0.01(26.53 ± 2.76)	0.17 ± 0.00(13.59 ± 2.25)	0.25 ± 0.01(4.88 ± 0.93)	0.15 ± 0.00(1.59 ± 0.30)
	SCG	**0.20 ± 0.04**(**8.32 ± 1.15**)	**0.16 ± 0.01**(**5.53 ± 0.70**)	**0.21 ± 0.03**(**2.68 ± 0.15**)	**0.13 ± 0.02**(**0.95 ± 0.19**)
	CG	0.15 ± 0.08(7.99 ± 0.72)	0.10 ± 0.05(4.45 ± 0.26)	0.15 ± 0.08(2.68 ± 0.45)	0.09 ± 0.05(0.92 ± 0.12)

Table 2: Average normalized-loss of **Insurance**, **Alarm**, **3-1-3**, and **5-3-5** on an independent test set consisting of 10000 data cases. The mean and standard deviation of five restarts are reported (bracket term: without BDeu priors). A bold term indicates the method which estimated the best model (CG not considered). When there is no value for conjugate gradient (CG) this indicates that one run of it took longer than one week on this domain.

each method the best model out of the set of restarts. A *sign test* yield that the number of cases, where the best model of SCG outperformed EM's best model, was large, 43/21, but there were variations over the domains: **Insurance** 12/4, **Alarm** 9/7, **3-1-3** 6/10, and **5-3-5** 16/0.

Table 3 summarizes the number of EM-equivalent iterations. EM was significantly faster than SCG on **Alarm** . However, this did not carry over to all domains. A *one-tailed, paired sampled t test* showed that EM ran significantly faster than SCG on **Alarm**, $p = 0.00005$, and on **3-1-3**, $p = 0.0005$. The difference in mean decreased from 82 for **Alarm** to 24 for **3-1-3**. This changed for **5-3-5** and **Insurance**. Here, the same test showed that SCG ran significantly faster than EM on **5-3-5**, $p = 0.0005$, and on **Insurance**, $p = 0.00005$. The difference in mean increased from 23 for **5-3-5** to 62 for **Insurance**.

To summarize, next to the quality of the estimated parameters, SCG seems to have some EM-like characteristics. It is fast, and insensitive to the initial set of parameters. This is remarkable given that CGs are usually considered to perform worse than EM (on discrete Bayesian networks). But the behaviors of both algorithms SCG and EM varied over the domain. It is reasonable to classify domains into "EM-easy" and "EM-hard" ones. Figure 1 shows the differences in average normalized-loss and in EM equivalent iterations. **Alarm** and **Insurance** are networks where prior knowledge is encoded into the

Domain / # hidden (in %)	Method	Number of data cases			
		100	200	500	1000
Insurance	EM	$100 \pm 28(144 \pm 34)$	$143 \pm 40(199 \pm 0)$	$188 \pm 17(199 \pm 0)$	$188 \pm 13(199 \pm 0)$
0.0	SCG	$95 \pm 40(307 \pm 75)$	$132 \pm 24(272 \pm 67)$	$138 \pm 42(380 \pm 34)$	$156 \pm 82(335 \pm 130)$
0.1	EM	$129 \pm 37(128 \pm 32)$	$178 \pm 19(193 \pm 9)$	$168 \pm 25(199 \pm 0)$	$177 \pm 27(199 \pm 0)$
	SCG	$42 \pm 14(280 \pm 36)$	$136 \pm 58(302 \pm 74)$	$91 \pm 67(318 \pm 43)$	$89 \pm 54(355 \pm 42)$
0.2	EM	$171 \pm 26(153 \pm 29)$	$161 \pm 42(199 \pm 0)$	$196 \pm 6(199 \pm 0)$	$178 \pm 19(199 \pm 0)$
	SCG	$64 \pm 25(251 \pm 68)$	$141 \pm 47(357 \pm 67)$	$112 \pm 42(335 \pm 44)$	$106 \pm 71(377 \pm 29)$
0.3	EM	$132 \pm 38(180 \pm 20)$	$156 \pm 27(199 \pm 0)$	$187 \pm 13(199 \pm 0)$	$180 \pm 36(199 \pm 0)$
	SCG	$77 \pm 41(272 \pm 69)$	$79 \pm 43(328 \pm 49)$	$89 \pm 57(312 \pm 59)$	$88 \pm 51(312 \pm 50)$
Alarm	EM	$1 \pm 0(1 \pm 0)$	$1 \pm 0(1 \pm 0)$	$1 \pm 0(1 \pm 0)$	$1 \pm 0(1 \pm 0)$
0.0	SCG	$66 \pm 18(97 \pm 12)$	$75 \pm 46(114 \pm 13)$	$104 \pm 20(148 \pm 15)$	$130 \pm 31(162 \pm 14)$
0.1	EM	$7 \pm 0(10 \pm 1)$	$8 \pm 0(13 \pm 3)$	$6 \pm 0(9 \pm 0)$	$6 \pm 0(8 \pm 0)$
	SCG	$83 \pm 36(130 \pm 8)$	$83 \pm 46(131 \pm 17)$	$140 \pm 18(147 \pm 12)$	$113 \pm 14(158 \pm 9)$
0.2	EM	$11 \pm 0(17 \pm 2)$	$11 \pm 0(16 \pm 3)$	$10 \pm 2(13 \pm 2)$	$9 \pm 0(12 \pm 1)$
	SCG	$84 \pm 22(167 \pm 26)$	$87 \pm 13(174 \pm 27)$	$100 \pm 7(170 \pm 32)$	$120 \pm 26(142 \pm 9)$
0.3	EM	$16 \pm 1(25 \pm 2)$	$14 \pm 0(23 \pm 2)$	$15 \pm 1(18 \pm 1)$	$13 \pm 1(17 \pm 1)$
	SCG	$83 \pm 27(163 \pm 22)$	$108 \pm 26(173 \pm 7)$	$69 \pm 45(164 \pm 6)$	$106 \pm 10(174 \pm 11)$
3-1-3	EM	$21 \pm 10(27 \pm 7)$	$21 \pm 2(22 \pm 2)$	$18 \pm 3(22 \pm 4)$	$16 \pm 1(18 \pm 1)$
0.0	SCG	$49 \pm 24(58 \pm 15)$	$63 \pm 6(72 \pm 30)$	$66 \pm 15(83 \pm 6)$	$60 \pm 7(67 \pm 14)$
0.1	EM	$28 \pm 4(46 \pm 13)$	$26 \pm 7(35 \pm 17)$	$27 \pm 7(36 \pm 8)$	$23 \pm 7(27 \pm 8)$
	SCG	$54 \pm 11(76 \pm 11)$	$57 \pm 13(72 \pm 35)$	$44 \pm 8(87 \pm 15)$	$62 \pm 5(76 \pm 8)$
0.2	EM	$24 \pm 9(36 \pm 11)$	$52 \pm 36(43 \pm 14)$	$30 \pm 6(36 \pm 6)$	$27 \pm 6(28 \pm 7)$
	SCG	$54 \pm 23(58 \pm 15)$	$60 \pm 7(89 \pm 10)$	$51 \pm 10(61 \pm 20)$	$54 \pm 17(80 \pm 11)$
0.3	EM	$37 \pm 9(40 \pm 8)$	$46 \pm 8(62 \pm 18)$	$32 \pm 3(36 \pm 3)$	$35 \pm 14(40 \pm 17)$
	SCG	$55 \pm 36(70 \pm 10)$	$68 \pm 26(78 \pm 12)$	$57 \pm 23(80 \pm 12)$	$57 \pm 12(74 \pm 16)$
5-3-5	EM	$97 \pm 7(38 \pm 6)$	$92 \pm 40(52 \pm 13)$	$120 \pm 39(78 \pm 17)$	$113 \pm 24(85 \pm 7)$
0.0	SCG	$75 \pm 37(151 \pm 60)$	$134 \pm 33(108 \pm 48)$	$151 \pm 50(166 \pm 7)$	$150 \pm 37(142 \pm 6)$
0.1	EM	$160 \pm 12(56 \pm 16)$	$116 \pm 19(67 \pm 14)$	$142 \pm 25(84 \pm 11)$	$162 \pm 14(126 \pm 20)$
	SCG	$80 \pm 48(207 \pm 45)$	$130 \pm 13(184 \pm 34)$	$112 \pm 22(141 \pm 21)$	$99 \pm 35(158 \pm 19)$
0.2	EM	$128 \pm 35(72 \pm 17)$	$190 \pm 19(78 \pm 16)$	$160 \pm 10(105 \pm 20)$	$141 \pm 10(100 \pm 19)$
	SCG	$99 \pm 62(180 \pm 38)$	$97 \pm 37(168 \pm 25)$	$150 \pm 58(140 \pm 37)$	$114 \pm 23(163 \pm 25)$
0.3	EM	$166 \pm 33(65 \pm 7)$	$99 \pm 15(78 \pm 16)$	$142 \pm 29(112 \pm 22)$	$157 \pm 23(131 \pm 17)$
	SCG	$72 \pm 42(170 \pm 21)$	$88 \pm 20(179 \pm 29)$	$121 \pm 28(151 \pm 9)$	$130 \pm 26(143 \pm 22)$

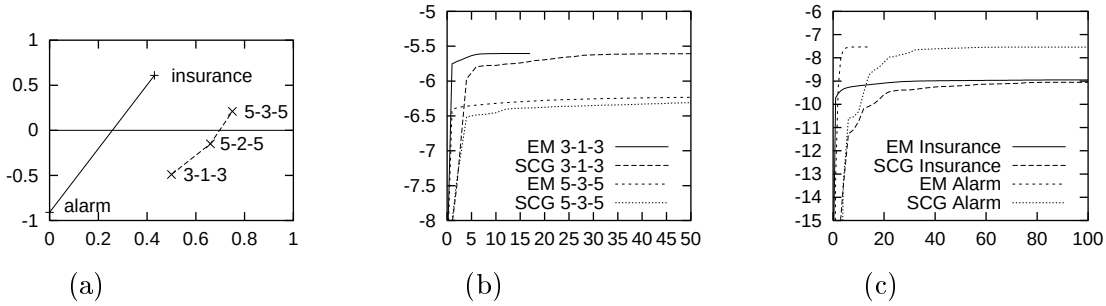
Table 3: Average EM equivalent iterations performed by EM and SCG. The mean and standard deviation of five restarts are reported (bracket term: without BDeu priors). An entry of 199 ± 0 for EM indicates that it exceeded the maximum number of iteration in all restarts.

structure, whereas **3-1-3** and **5-3-5** have uninformative structures. Figure 1 (a) shows that on average EM tends to yield better normalized-loss on prior knowledge networks. However, within both types of network structures, the fraction of latent parameters seems to correlate to "EM-hard" domains (in particular w.r.t. number of iterations). Figure 1 (b) shows that SCG converged faster on networks with a higher number of latent parameters, i.e. **Alarm** and **5-3-5**. Figure 2(a) shows the percentage of latent parameters vs. speed-up which is defined as the quotient of average EM-equivalent iterations (over corresponding restarts) of SCG and EM minus one. To test whether SCGs performs well on "EM-hard" domains, i.e. that these domains are "SCG-easy" ones, we investigated the network **5-2-5**. It is a variant of **5-3-5** with only two latent variables. The observable variables have two, the latent one three states. We tested **5-2-5** with the same experiments as before but restricted to 1000 data cases. SCG's speed-up on **5-2-5** was -0.15, i.e. between the speed-up of **3-1-3** and **5-3-5**. Although this is preliminary, the result is promising.

Finally, as Figures 2(b) and 2(c) show, EM is usually faster when far away from the solution. Therefore, the EM seems to be the domain independent method of choice. But, the results show that SCG is not only competitive but that it can be superior to EM, and in contrast to CG, it learned **Insurance** and **Alarm**.

5 Related Work

Both, EM and gradients are well-known parameter estimation techniques. For a general introduction see (McLachlan and Krishnan, 1997; Luenberger, 1984). Bishop (Bishop, 1995) and Møller (Møller, 1993a) show how to use (scaled) CGs to estimate parameters of neural networks. Lauritzen (Lauritzen, 1995) introduced EM for Bayesian networks (see also Heckerman (Heckerman, 1995) for a nice tutorial). The original work on gradient-based approaches in the context of Bayesian networks was done by Binder *et al.* (J.Binder et al., 1997), some recent work by Jensen (Jensen, 1999). However, we are neither aware of any direct experimental comparison between the two approaches



(a) (b) (c)

Figure 2: (a) Percentage of latent parameters vs. SCG's speed-up. The domains are grouped into network structures with and without encoded prior information. (b) Learning curves averaged over five restarts for 1000 data cases, no missing values. (c) Learning curves averaged over five restarts for 1000 data cases. Fraction of 0.3 missing values for **Alarm**, no missing values for **Insurance**.

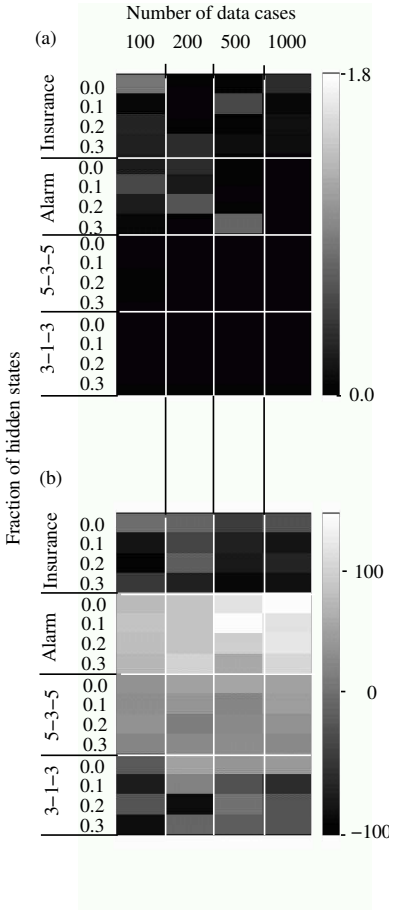


Figure 1: (a) Differences (SCG − EM) of average normalized-loss over **Insurance**, **Alarm**, **3-1-3**, and **5-3-5** on an independent test set consisting of 10000 data cases. (b) Difference (SCG − EM) of EM equivalent iterations. In both pictures, the mean of five restarts were subtracted.

nor of research investigating accelerated CGs such as SCGs. Binder *et al.* (J.Binder et al., 1997) used traditional CG, and did not compare more advanced gradient-based and EM-based approaches. Bauer *et al.* (Bauer et al., 1997) reported on experiments with different learning algorithms for Bayesian networks, but they focused on accelerations of the EM. They did not report on results of CGs. Thiesson (Thiesson, 1995) discussed (traditional) CG accelerations of the EM, but does not report on experiments. Ortiz and Kaelbling (Ortiz and Pack Kaelbling, 1999) conducted experiments with several learning algorithms (mainly accelerated EMs) for continuous models. They argued "that CG by itself is not a good idea since it requires a very precise line search when it is far away from a solution". Our experiments do not support this for discrete Bayesian networks. Ortiz and Kaelbling also classified their domains into "easy" and "hard" ones. Like in our case, accelerated methods improved convergence on hard problems. However, they accelerated EM and not CG.

6 Conclusions and future work

In this case study, we experimentally compared EM and gradient-based algorithms for ML parameter estimation on several Bayesian networks. To overcome the expensive line search of traditional conjugate gradients, we applied *scaled conjugate gradients*. They are novel in the context of Bayesian networks. The experiments

show that SCGs are competitive with EM in both quality of the estimate and convergence speed. To the best of our knowledge, this is the first time that gradient-based techniques are reported to be competitive with EM w.r.t quality and speed for discrete Bayesian networks. In general EM seems to be the domain independent method of choice, but which technique is superior depends on properties of the domain. We identified "EM-easy" and "EM-hard" domains, and Scaled conjugate gradients seem especially well suited in the presence of many latent parameters, i.e. "EM-hard" instance. Given that gradients are more flexible – as they allow us to learn non-multinomial parameterization (e.g. noisy-or) using the chain-rule of derivatives and conditional probability distributions using Bayes' rule – and that they are strongly related to methods in neural networks training, scaled conjugate gradients are a promising alternative to EM for parameter estimation of (discrete) Bayesian networks.

Our results suggest that the number of latent variables determines "EM-hard" instances. In the future, we will conduct more experiments to validate this and to find other properties. Comparing SCGs with gradient-based accelerated EMs should yield further insight into the nature of the learning problem. In particular, accelerating EM using SCG, and applying stochastic SCGs (Møller, 1993b) to the problem of *online-learning* of Bayesian networks are interesting lines of future research.

Acknowledgements

The authors would like to thank Luc De Raedt and Wolfram Burgard for helpful comments on the paper. Kristian Kersting was supported by the European Union IST programme, IST-2001-33053 (Application of Probabilistic Inductive Logic Programming - APRIL).

References

E. Bauer, D. Koller, and Y. Singer. 1997. Update Rules for Parameter Estimation in Bayesian Networks. In *Proceedings of the Thirteenth Conference on Unertainty in Artificial Intelligence (UAI)*, pages 3–13.

I. Beinlich, H. Suermondt, R. Chavez, and G. Cooper. 1989. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of Second European Conference on Artificial Intelligence and Medicine*, pages 247–256.

C. M. Bishop. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press.

H. Chan and A. Darwiche. 2002. When do numbers really matter? *Journal of Artificial Intelligence Research (JAIR)*, 17:265–287.

G. F. Cooper. 1990. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405.

D. Heckerman, D. Geiger, and D. M. Chickering. 1995. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243.

D. Heckerman. 1995. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, March.

J.Binder, D. Koller, S. Russell, and K. Kanazawa. 1997. Adaptive probabilistic networks with hidden variables. *Machine Learning*, pages 213–244.

F. V. Jensen. 1999. Gradient descent training of bayesian networks. In *Proceedings of European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU)*, number 1638 in LNAI, pages 190–200.

S. L. Lauritzen. 1995. The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19:191–201.

D. G. Luenberger. 1984. *Linear and Nonlinear Programming*. Addison-Wesley, 2. edition.

G. McLachlan and T. Krishnan. 1997. *The EM Algorithm and Extensions*. Wiley, New York.

M. Møller. 1993a. A Scaled Conjugate Gradient Algoritm for Fast Supervised Learning. *Neural Networks*, 6:525–533.

M. Møller. 1993b. Supervised Learning on Large Redundant Training sets. *International Journal Neural Systems*, 4(1):15–25.

J. Nocedal. 1992. Theory of algorithms for unconstrained optimization. *Acta Numerica*, pages 199–242.

L. E. Ortiz and L. Pack Kaelbling. 1999. Accelerating EM: An Empirical Study. In *Proceedings of Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 512–521.

J. Pearl. 1991. *Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 2. edition.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. 1993. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2. edition.

B. Thiesson. 1995. Accelerated quantification of Bayesian networks with incomplete data. In *Proceedings of the First ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 306–311.