Computational Statistics & Data Analysis 19 (1995) 177–189
North-Holland

On the effective implementation of the iterative proportional fitting procedure

Radim Jiroušek and Stanislav Přeučil
Institute of Information Theory and Automation, Prague, Czech Republic

Received January 1993
Revised July 1993

Abstract: The famous Iterative Proportional Fitting Procedure is known to be computationally extremely complex from both space and time point of view. Its implementation introduced in [6] was proposed to decrease space requirements. This paper presents the implementation in a more procedural way and shows, both analyzing the process and by examples, the fact that the implementation decreases the time complexity as well.

Keywords: Marginal problem; Maximum likelihood estimate; Algorithmic complexity

1. Introduction

In [3,4], Edwarads and Havránek proposed a procedure for model selection in contingency tables. Its goal is to find the simplest possible graphical model corresponding to a dependence structure of data, which is assumed to be summarized in a multidimensional contingency table. The high algorithmical complexity of the procedure consists in two crucial moments. The first one is the necessity to test great numbers of models before a solution is reached. In the mentioned original papers, this problem was to some extent reduced by utilizing the fact that graphical models form a lattice. The other bottleneck of the procedure is the statistical test itself. To perform it, one needs to calculate the maximum likelihood estimate of the multidimensional distribution under the assumption of the tested model. This estimate is usually obtained (in a general case) by the procedure known as Iterative Proportional Fitting. As the space and time complexity of this procedure is exponential, it is no wonder that existing programs cannot be applied to problems of more than 8 or 9 dimensions.

Correspondence to: R. Jiroušek, ÚTIA AVČR, Pod vodárenskou věži 4, 18208 Prague 8, Czech Republic.

SSDI 0167-9473(93)E0055-9

The present paper suggests the employment of a space saving modification of IPF procedure [6] and shows, both theoretically and on experimental comparisons, to what extent one can expect increase of efficiency of the modified algorithms.

2. Notation

A general notation in the field of multidimensional contingency tables is rather complicated and therefore authors often present their results on 4 or 5 dimensional examples. To avoid such a simplification and keep a general case, let us use the following symbolism.

Consider a finite index set V and assume that a contingency table under consideration represents realizations of a $|V|$-dimensional random variable $X^V = ((X_i)_{i \in V})$. $\mathbb{X}_i$, $i \in V$, will denote a finite set of values of the variable X_i, $|\mathbb{X}_i| \geq 2$, and the Cartesian product $\mathbb{X}^V = \times_{i \in V} \mathbb{X}_i$ correspond to the space of possible values of the (multidimensional) random variable X^V. For each $x \in \mathbb{X}^V$, $x = ((x_i)_{i \in V})$, $n^V(x)$ will be the entry of a given $|V|$-dimensional contingency table corresponding to the frequency of the respective combination of values $x = ((x_i)_{i \in V})$. The total number of experiments is denoted by n; so $n = \sum_{x \in \mathbb{X}^V} n^V(x)$.

For any subset $C \subset V$, $\mathbb{X}^C = \times_{i \in C} \mathbb{X}_i$ will denote the marginal subspace corresponding to observation of the variables $(X_i)_{i \in C}$ only. By the symbol $y \in \mathbb{X}^C$, we shall understand either a single point $y = ((y_i)_{i \in C})$ from the space $\mathbb{X}^C$, or a subset $y = \{x = ((x_i)_{i \in V}) \in \mathbb{X}^V : x_i = y_i \text{ for all } i \in C\}$ of points from $\mathbb{X}^V$ whose respective coordinates coincide with those of y. Using this notation, we can introduce the *marginal contingency table* corresponding to coordinates from C with entries $n^C(y)$ for all $y \in \mathbb{X}^C$ defined as

$$n^C(y) = \sum_{x \in y} n^V(x).$$

In consistency with this notation, whenever n^C occurs for $C = \emptyset$ it will mean nothing else but the total number of experiments n.

2.1 Example

Let $V = \{1, 2, 3\}$ and $\mathbb{X}_1 = \mathbb{X}_2 = \mathbb{X}_3 = \{0, 1\}$. Consider a 3-dimensional contingency table given in Table 1 for which $n = 16$. In this case, $n^V(0, 0, 0) = 5$ and $n^V(0, 1, 0) = 0$.

Table 1

X_1		0		1	
X_2		0	1	0	1
X_3	0	5	0	0	3
	1	0	3	3	2

Table 2

	$X_3=0$	$X_3=1$
$X_1=0$	5	3
$X_1=1$	3	5

Taking $C=\{1, 3\}\subset V$ we consider the marginal for X_1 and X_3. The frequencies corresponding to this $n^C(y)$ are in Table 2.

3. Graphical models

Consider a *simple*, i.e. unoriented, without loops and multiple edges, *graph* $G=(V, E)$. Let $C_1,\dots,C_K$ be the system of its *cliques* (maximal complete subgraphs). A probability distribution P on $\mathbb{X}^V$ is said to *factorize with respect to* G if there exist functions $\psi_1,\dots,\psi_k$ defined on $\mathbb{X}^{C_1},\dots,\mathbb{X}^{C_K}$ respectively such that

$$P\big((X_i=x_i)_{i\in V}\big)=\prod_{k=1}^{K}\psi_k\big((x_i)_{i\in C_k}\big).$$

For any graph $G=(V, E)$, Π_G will denote the set of all probability distributions on $\mathbb{X}^V$ factorizing with respect to G.

3.1. Example

The system of cliques of the graph from Figure 1 is $C_1=\{1, 2, 3\}$, $C_2=\{2, 5\}$, $C_3=\{3, 6\}$, $C_4=\{4\}$, $C_5=\{5, 6\}$, $C_6=\{6, 7\}$. Therefore, for any distribution P factorizing with respect to this graph there must exist functions $\psi_1,\dots,\psi_6$ such that

$$P(x_1,\dots,x_7)$$
$$=\psi_1(x_1, x_2, x_3)\psi_2(x_2, x_5)\psi_3(x_3, x_6)\psi_4(x_4)\psi_5(x_5, x_6)\psi_6(x_6, x_7).$$

The important step of the Edwards-Havránek procedure is the statistical test of the following hypothesis:

For a given simple graph $G=(V, E)$, the $|V|$-dimensional contingency table under consideration was generated by independent realizations of a $|V|$-dimensional random variable with probability distribution from Π_G

To perform this test, the following statistics is usually calculated

$$\frac{1}{n}\sum_{x\in\mathbb{X}^V}n^V(x)\log\frac{n^V(x)}{\hat{n}^V(x)}=\sum_{x\in\mathbb{X}^V}p(x)\log\frac{p(x)}{\hat{p}(x)},$$

where $p(x) = n^V(x)/n$ is a *relative frequency* of the entry of the contingency table corresponding to the point $x \in \mathbb{X}^V$ and $\hat{p}(x) = \hat{n}^V(n)/n$ is a *maximum likelihood estimate* of the respective probability under the assumption of the hypothesis. To get these maximum likelihood estimates the *Iterative Proportional Fitting* procedure may be used [2,5].

4. Iterative proportional fitting

Let us describe how the maximum likelihood estimate of relative frequencies for a given graph $G = (V, E)$ is obtained with the help of the Iterative Proportional Fitting procedure.

Let $C_1, \ldots, C_K$ be the cliques of $G = (V, E)$. First, it is necessary to compute, for each $k = 1, \ldots, K$, a marginal table containing relative frequencies corresponding to variables $((X_i)_{i \in C_k})$:

$$\forall_y \in \mathbb{X}^{C_k},\ p_k(y) = \sum_{x \in y} p(x) = \frac{1}{n} \sum_{x \in y} n^V(x) = \frac{1}{n} n^{C_k}(y).$$

Before starting the iterative process, which computes at each step new values $q(z)$ for all $z \in \mathbb{X}^V$, it is necessary to assign initial values. For several reasons, see [1] for details, it is reasonable to use the uniform distribution for this purpose:

$$q_0(z) = \frac{1}{|\mathbb{X}^V|} \quad \text{for all} \quad z \in \mathbb{X}^V.$$

The iterative process consists in recurrent computation of $|V|$-dimensional tables (distributions) $q_1, q_2, \ldots$ (i.e. $q_i(z)$ for all $z \in \mathbb{X}^V$) according to the following formula

$$q_i(z) = q_{i-1}(z) \frac{p_k(y)}{\sum_{x \in y} q_{i-1}(x)},$$

where $y \in \mathbb{X}^{C_k}$ for which $z \in y$ and $k = ((i-1) \bmod K) + 1$.

To make this (and similar ones) formula more self-explanatory let us introduce further notation consistent with that used above. For $z \in \mathbb{X}^V$, the symbol z^{C_k} denotes that $y \in \mathbb{X}^{C_k}$ for which $z \in y$. Then, the formula of the iterative process can be rewritten into the following form

$$q_i(z) = q_{i-1}(z) \frac{p_k(z^{C_k})}{\sum_{x \in z^{C_k}} q_{i-1}(x)}.$$

What does this formula mean from the point of view of computational complexity? First, the value has to be computed for all $z \in \mathbb{X}^V$; it means $|\mathbb{X}^V| = \prod_{i \in V} |\mathbb{X}_i| \geq 2^{|V|}$ values. Further, there is a summation in the expres-

sion. Fortunately, it is not necessary to perform this summation for every $z \in \mathbb{X}^V$ independently (it would need summing of $\prod_{i \in V - C_k} |\mathbb{X}_i|$ numbers) as it is possible to prepare all these values before starting computation of $q_i(z)$ just by one cycle over the whole space $\mathbb{X}^V$. Thus, one step of the iterative process requires two cycles of length $|\mathbb{X}^V|$. At each step, only simple arithmetic operations are performed so we can say that a time complexity of one step is linear with $|\mathbb{X}^V|$.

To enable the reader to understand the principle of the simplification proposed in [6] a definition and basic properties of decomposable models have to be introduced.

5. Decomposable models

Consider a probability distribution P on $\mathbb{X}^V$ factorizing with respect to a graph $G = (V, E)$. If the graph G is *triangulated* (i.e. contains no chordless cycles of length four or greater) then the distribution P is said to be *decomposable* with respect to G.

The main advantage of decomposable models lies in the fact that there is a direct way how to get the maximum likelihood estimates. In this case, namely, the iterative process stabilizes after computing the K-th distribution, i.e. $q_K = q_{K+1} = q_{K+2} = \ldots$. However, the same estimate can be obtained even in a simpler way. It is known [5] that all the cliques of a triangulated graph can be enumerated in such a way that the sequence $C_1, \ldots, C_K$ meets the so called *running intersection property*:

$$\forall_i = 2, \ldots, K \exists j (1 \leq j < i) \left(C_i \cap \left(\bigcup_{k=1}^{i-1} C_k \right) \subset C_j \right).$$

Having such an ordering of the cliques of a triangulated graph $G = (V, E)$, any probability distribution $P(x)$ (defined for all $x \in \mathbb{X}^V$) decomposable with respect to G can be expressed with the help of its marginals $P(x^{C_k})$ by the following formula:

$$P(x) = P(x^{C_1}) \frac{P(x^{C_2})}{P(x^{C_2 \cap C_1})} \cdots \frac{P(x^{C_K})}{P(x^{C_K \cap (C_1 \cup \ldots \cup C_{K-1})})},$$

(where, again, for $x \in \mathbb{X}^V$ x^C denotes that $y \in \mathbb{X}^C$ for which $x \in y$). And this is an analogy of the formula yielding directly the maximum likelihood estimate $\hat{p}(x)$:

$$\hat{p}(x) = p_1(x^{C_1}) \frac{p_2(x^{C_2})}{\sum_{z \in x^{C_2 \cap C_1}} p(z)} \cdots \frac{p_K(x^{C_K})}{\sum_{z \in x^{C_K \cap (C_1 \cup \ldots \cup C_{K-1})}} p(z)},$$

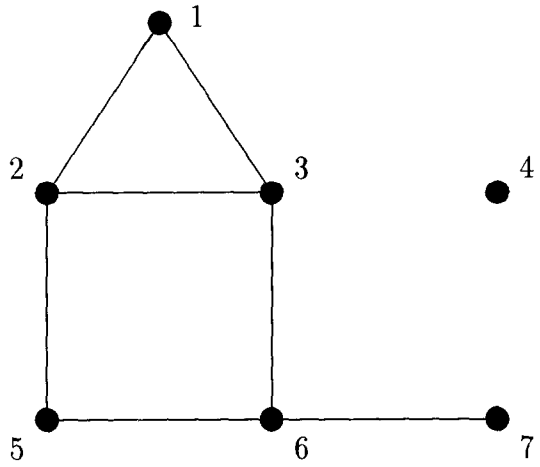

Fig. 1.

(naturally, in case that $C_k \cap (C_1 \cup \ldots \cup C_{k-1}) = \emptyset$ then

$$\sum_{z \in x^{C_K \cap (C_1 \cup \ldots \cup C_{K-1})}} p(z) = 1,$$

as $x^{\emptyset} = \mathbb{X}^V$).

Notice, that this is also the very property making possible space-saving representation of decomposable distributions. One need not store a $|V|$-dimensional distribution but only a system of its marginal distributions for the subspaces $\mathbb{X}^{C_1}, \mathbb{X}^{C_2}, \ldots, \mathbb{X}^{C_K}$. Therefore, in the case of binary variables, one needs to store only

$$\sum_{k=1}^{K} 2^{|C_k|},$$

instead of $2^{|V|}$ probabilities. From this property one can easily deduce situations when this type of representation is advantageous (sparse triangulated graphs).

5.1. Example

The graph from Figure 1 is not triangulated. It contains a cycle (2, 3, 6, 5) of length 4. However, adding (for example) the edge (2, 6) the situation changes substantially. The new graph is triangulated (see Figure 2) and contains only 5 cliques: $D_1 = \{1, 2, 3\}$, $D_2 = \{2, 3, 6\}$, $D_3 = \{2, 5, 6\}$, $D_4 = \{4\}$, $D_5 = \{6, 7\}$. Notice that the orderings $(D_1, D_2, D_3, D_4, D_5)$, $(D_2, D_1, D_3, D_4, D_5)$, $(D_3, D_2, D_1, D_5, D_4)$, $(D_4, D_5, D_3, D_2, D_1)$ and $(D_5, D_2, D_3, D_1, D_4)$ meet the running intersection property. This does not hold, for example, for the orderings $(D_1, D_3, D_2, D_4, D_5)$ and $(D_4, D_5, D_1, D_2, D_3)$.

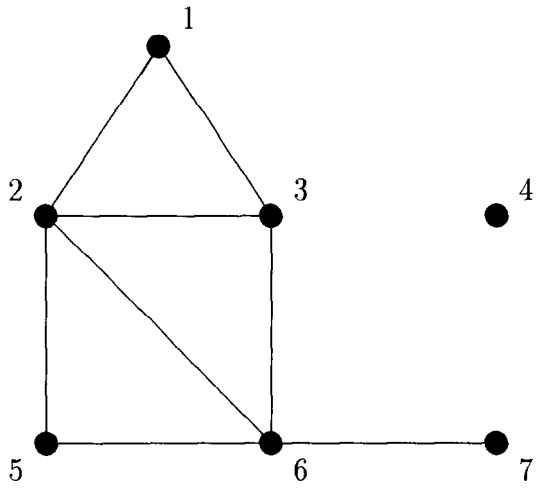

Fig. 2.

Using the properties mentioned above, for any probability distribution $P(x_1,\ldots,x_7)$ factorizing with respect to the graph from Figure 2 the following expression must hold:

$$P(x_1,\ldots,x_7)$$
$$= P(x_1, x_2, x_3)\frac{P(x_2, x_3, x_6)}{P(x_2, x_3)}\frac{P(x_2, x_5, x_6)}{P(x_2, x_6)}P(x_4)\frac{P(x_6, x_7)}{P(x_6)},$$

(all terms $\frac{0}{0}$ are taken to be 1).

The way how to utilize these advantageous properties of decomposable models even in cases of general graphical models, i.e. when the considered graph $G=(V, E)$ is not triangulated, was shown in [6]. It is based on the fact that both the result of the iterative procedure (i.e. the maximum likelihood estimate $\hat{p}$) and all the distributions $q_0, q_1, q_2,\ldots$ computed during the iterative process are decomposable with respect to any triangulated graph $H=(V, F)$ with $F \supset E$.

6. Space-saving implementation of IPF

Consider a general graph $G=(V, E)$ which is not necessarily triangulated. In [5,8] it is shown how to treat the situations if either

- there is a node $i \in V$ contained only in one clique, or
- the system of the cliques $C_1,\ldots,C_K$ can be ordered in such a way that for some k

$$(C_1 \cup C_2 \cup \ldots \cup C_k) \cap (C_k \cup C_{k+1} \cup \ldots \cup C_K) \subseteq C_k.$$

(Let us remark that for triangulated graphs with more than one clique both these conditions are fulfilled.) In these cases the problem can be decomposed and the maximum likelihood estimate can be obtained directly from the maximum likelihood estimates of the subproblems.

The method we are dealing with (and which is described in [6] in detail, see also the appendix) is recommendable mainly for situations when neither of the two conditions is fulfilled. As the simplest example of such a situation let us mention a cycle; i.e. the graph $G_m = (V_m, E_m)$ with

$$V_m = \{1, 2, \ldots, m\},$$

$$E_m = \{(1, 2), (2, 3), \ldots, (m-1, m), (m, 1)\},$$

which will be also used for computational experiments.

The initial phase of the space-saving algorithm is to add some edges to the original graph $G = (V, E)$ to get a triangulated graph $H = (V, F)$ $(F \supseteq E)$. To reach this, we have implemented the Tarjan-Yannakakis *fill-in algorithm* [9]. (Another method optimizing the triangulation process with respect to the storage requirements has been published by U. Kjærulff [7].) Further, it is reasonable to find K orderings of the cliques of the graph H (necessary in the step II(i); see the appendix). To describe this process denote by $C_1, \ldots, C_K$ all the cliques of G and by $D_1, \ldots, D_L$ the cliques of H. For each $k = 1, \ldots, K$ we need a permutation $\sigma_k(1), \ldots, \sigma_k(L)$ of indices $1, \ldots, L$ such that $D_{\sigma_k(1)}, \ldots, D_{\sigma_k(L)}$ meets the running intersection property and $D_{\sigma_k(1)} \supseteq C_k$. This is implemented with the help of the *restricted maximum cardinality search algorithm* described in [9,5]. For each k it is enough to enumerate all the nodes of C_k first and then proceed according to the maximum cardinality search. The cliques $D_1, \ldots, D_L$ are then ordered in a succession corresponding to the highest number assigned to a node of the respective clique.

The iterative phase of the algorithm realizes the computation according to the formula

$$q_i(z) = q_{i-1}(z) \frac{p_k(z^{C_k})}{\sum_{x \in z^{C_k}} q_{i-1}(x)}.$$

Owing to decomposability of q_i with respect to H, we do not store the distribution q_i but only its marginals $q_i^{D_1}, \ldots, q_i^{D_L}$. During each iterative step all these marginals have to be updated (corresponding to a computation of q_i from q_{i-1}; see II(ii) and (iii) at the appendix). The updating proceeds according to the ordering $\sigma_k(1), \ldots, \sigma_k(L)$ using the following three routines:

(1) The marginal $q_{i-1}^{D_{\sigma_k(l)}}$ is further marginalized (within an auxiliary store) to get $q_{i-1}^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}$. For $q_{i-1}^{D_{\sigma_k(1)}}$ (i.e. $l = 1$), the marginalization is done for $q_{i-1}^{C_k}$ (the permutation σ_k is such that $C_k \subseteq D_{\sigma_k(1)}$).

(2) The marginal $q_i^{D_{\sigma_k(j)}}$ is further marginalized to get $q_i^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}$, where j is the very index whose existence is guaranteed by the running intersection property:

$$1 \leq j < l \text{ and } D_{\sigma_k(l)} \cap \left(D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)}\right) \subset D_{\sigma_k(j)}.$$

For $l = 1$, p_k is used instead of $q_i^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}$ in the subsequent computations.

(3) For all $y \in \mathbb{X}^{D_{\sigma_k(l)}}$ compute

$$q_i^{D_{\sigma_k(l)}}(y) = q_i^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}\left(y^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}\right) \cdot \frac{q_{i-1}^{D_{\sigma_k(l)}}(y)}{q_{i-1}^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}\left(y^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}\right)}$$

(where, as was said above, p_k is used for $l = 1$ instead of $q_i^{D_{\sigma_k(l)} \cap (D_{\sigma_k(1)} \cup \ldots \cup D_{\sigma_k(l-1)})}$.

7. Algorithmic complexity

As for the *space complexity* of the implementation it is clear that the space requirements are the following:

- size of the distributions $p_1, \ldots, p_k$ is
$$\sum_{k=1}^{K} |\mathbb{X}^{C_k}| = \sum_{k=1}^{K} \prod_{i \in C_k} |\mathbb{X}_i|,$$
- size of the distributions $q_i^{D_1}, \ldots, q_i^{D_L}$ is
$$\sum_{l=1}^{L} |\mathbb{X}^{D_l}| = \sum_{l=1}^{L} \prod_{i \in D_l} |\mathbb{X}_i|,$$
- size of the auxiliary memory necessary for storing marginals computed at each step of the iterative process (routines 1 and 2) cannot exceed
$$2 \max_{l=1,\ldots,L} |\mathbb{X}^{D_l}|.$$

Therefore, it is clear that this implementation is advantageous when the cliques of the triangulated graph H are small in comparison with the dimension of the problem $|V|$.

The study of the *time complexity* is much more complicated. To simplify it we shall proceed in the same way as when discussing the complexity of the "classical" implementation of the Iterative Proportional Fitting procedure at the end of the respective paragraph.

Let us start with what must be done at each step of the iterative process. (It is worth mentionning that from the numerical standpoint the whole iterative process is not influenced by the implementation, therefore, the number of

iterative steps necessary to reach a satisfactory solution remains unchanged.) All q^{D_l} must be updated (though in different orderings). Therefore, the routine 3 must process all the entries of the table representing q^{D_l} – that is, $\prod_{i \in D_l} |\mathbb{X}_i|$ entries. As the complexity of the marginalization depends on the size of the input distribution the preceding estimate holds also for the preparatory routine 1. Thus, the time complexity of these parts of the algorithm (as with the "classical" implementation) is linear with the space necessary to represent the distribution. To estimate the complexity of the routine 2 which marginalize some other table – $q^{D_{\sigma_k(j)}}$ – we must use an upper boundary

$$\max_{j=1,\ldots,L} \prod_{i \in D_j} |\mathbb{X}_i|.$$

Numbers of summations, multiplications and divisions connected with any access to the memory are (approximately) the same as in the "classical" implementation. So we can summarize the above considerations in the following way: numbers of basic operations required to perform each iterative step of the procedure can be approximately expressed as a product of three parameters $\alpha \cdot \beta \cdot \gamma$, where

α – independent of implementation,

β – size of the memory necessary to represent the distribution,

γ – $\begin{cases} 2 & - \text{ for the "classical" implementation,} \\ > 3 & - \text{ for the space-saving implementation.} \end{cases}$

Therefore, the space-saving implementation is advantageous whenever requirements for storing the decomposable model is substantially less than storing directly the $|V|$-dimensional distribution. This may be guaranteed by keeping the cliques of the decomposable model small.

As the theoretical considerations may not be transparent and convincing let us conclude with an illustrative example.

7.1. Example

We shall present computational time necessary to perform 100 steps of the iterative process for graphs $G_m = (V_m, E_m)$ with

$$V_m = \{1, 2, \ldots, m\}$$

$$E_m = \{(1, 2), (2, 3), \ldots, (m-1, m), (m, 1)\}.$$

To be honest, this is the very type of a graph where the effectivity of the described method shows up most. The corresponding triangulated graphs H_m contain always $|F_m| = 2m - 3$ edges and all $(m-2)$ cliques consist of three nodes.

The second column of Table 3 contains results of experiments with medical data (rheumatic diseases) where the number of values were different for different variables ($2 \leq |\mathbb{X}_i| \leq 9$). Therefore, to exclude obvious dependence on the sizes of sample spaces of different variables we generated an artificial data

Table 3
Time necessary to perform 100 iterative steps (sec)

length of a cycle m	effective implementation		"classical" implementation
	medical data	artificial data	
3	0.65	0.12	0.09
4	0.77	0.22	0.19
5	1.11	0.33	0.65
6	2.32	0.41	1.70
7	3.23	0.52	4.50
8	3.29	0.62	11.87
9	3.74	0.68	29.50
10	4.73	0.76	60.07
11	4.97	0.86	148.21
12	5.01	0.98	380.34
13	5.06	1.08	812.53
14	5.05	1.18	–
15	5.40	1.30	–
16	5.88	1.40	–
17	6.06	1.50	–
18	6.27	1.67	–
19	6.30	1.79	–
20	6.42	1.93	–

file with binary variables. The results obtained with this data file is contained in the third column of Table 3. By request of the referees we added the fourth column containing the time necessary to perform 100 steps of the procedure implemented in a "classical" way. These computations were performed for the artificial data only. All computational experiments were performed on HP Apollo 9000 model 720 connected to a network. Therefore, the computational time is to some extent dependent on other activities of the computational system and thus the numbers must be taken with a reserve. The main reason is to prove that the implementation increases substantially the dimensionality of solvable problems. At this point, it is also worth mentioning that the number of necessary iterative steps to reach the solution with sufficient accuracy was always less than 50.

Appendix

The algorithm presented below is taken from [6]. All the changes performed have been required by the necessity to adapt the notation to the present paper.

Algorithm

Input. A system of marginal distributions $p_1, \ldots, p_K$ defined for variables corresponding to cliques $C_1, \ldots, C_K$ of a graph $G = (V, E)$.

Output. System of marginal distributions $q_1, \ldots, q_L$ defined for variables corresponding to cliques $D_1, \ldots, D_L$ (the ordering meets the running intersection property) of the triangulated graph $H = (V, F)$ for which $F \supseteq E$. The resulting maximum likelihood estimate is given by the formula

$$q_1\left[q_2/q_2^{D_2 \cap D_1}\right] \ldots \left[q_L/q_L^{D_L \cap (D_1 \cup \ldots \cup D_{L-1})}\right].$$

I. *Initial phase.* Find a triangulated graph $H = (V, F)$ with $F \supseteq E$. Denote its cliques $D_1, \ldots, D_L$. It is desirable to construct such a graph H with minimum possible value

$$\sum_{l=1}^{L} \prod_{j \in D_L} |\mathbb{X}_j|.$$

Some ideas how to achieve this optimal solution are in [10,7].

II. *Iterative phase.* Define

$$q_{0,l} = \frac{1}{\prod_{j \in D_l} |\mathbb{X}_j|},$$

for all $l = 1, \ldots, L$.

For $i = 1, 2, \ldots$ (untill it converges) perform the following procedure:

(i) Choose any permutation $\sigma(1), \ldots, \sigma(L)$ such that $(D_{\sigma(1)}, \ldots, D_{\sigma(L)})$ meets the running intersection property and $D_{\sigma(1)} \supseteq C_k$ for $k = ((i-1) \bmod K) + 1$.

(ii) Compute

$$q_{i,\sigma(1)} = p_k\left[q_{i-1,\sigma(1)}/q_{i-1,\sigma(1)}^{C_k}\right].$$

(iii) For $l = 2, \ldots, L$ compute the following two distributions:

(a) marginal distribution of the distribution

$$\hat{q}_{i,l} = q_{i,\sigma(1)}\left[q_{i,\sigma(2)}/q_{i,\sigma(2)}^{D_{\sigma(2)} \cap D_{\sigma(1)}}\right] \ldots \left[q_{i,\sigma(l-1)}/q_{i,\sigma(l-1)}^{D_{\sigma(l-1)} \cap (D_{\sigma(1)} \cup \ldots \cup D_{\sigma(l-2)})}\right],$$

for variables $\{X_m\}_{m \in D_{\sigma(l)} \cap (D_{\sigma(1)} \cup \ldots \cup D_{\sigma(l-1)})}$ only, and

(b)

$$q_{i,\sigma(l)} = \hat{q}_{i,l}^{D_{\sigma(l)} \cap (D_{\sigma(1)} \cup \ldots \cup D_{\sigma(l-1)})}\left[q_{i-1,\sigma(l)}/q_{i-1,\sigma(l)}^{D_{\sigma(l)} \cap (D_{\sigma(1)} \cup \ldots \cup D_{\sigma(l-1)})}\right].$$

References

[1] I. Csiszár, I-divergence geometry of probability distributions and minimization problems. *Ann. Probab.* **3** (1975), 146–158.

[2] W.E. Deming and F.F. Stephan, On a least square adjustment of a sampled frequency table when the expected marginal totals are known. *Ann. Math. Statist.* **11** (1940), 427–444.

[3] D. Edwards and T. Havránek, A fast procedure for model search in multidimensional contingency tables. *Biometrica* **72** (1985), 339–351.

[4] D. Edwards and T. Havránek, A fast model search in multidimensional contingency tables. *J. Am. Stat. Assoc.*, **82** (1987), 205–214.

[5] P. Hájek, T. Havránek and R. Jiroušek, Uncertain Information Processing in Expert Systems. CRC Press, Inc. Boca Raton, Ann Arbor, London, Tokyo, 1992.

[6] R. Jiroušek, Solution of the marginal problem and decomposable distributions. *Kybernetika* **27** (1991), 403–412.

[7] U. Kjærulff, Optimal decomposition of probabilistic networks by simulated annealing. *Statistics and Computing* **2** (1992), 7–17.

[8] F.M. Malvestuto, Computing the maximum-entropy extension of a given discrete probability distributions. *Comput. Statist. Data Anal.* **8** (1989), 299–311.

[9] R.E. Tarjan and M. Yannakakis, Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM J. Comput.* **13** (1984), 566–579.

[10] Lianwen Zhang, Studies on finding hypertree covers for hypergraphs. Working paper no. 198, School of Business, The Univerity of Kansas, Lawrence 1988.