New strategies for finding multiplicative decompositions of probability trees

Irene Martínez*

Dpt. Informatics
University of Almería
La Cañada de San Urbano s/n
04120 Almería, Spain

Serafín Moral

Dpt. Computer Science and Artificial Intelligence
University of Granada
C/ Daniel Saucedo Aranda, s/n,
18071 Granada, Spain

Carmelo Rodríguez, Antonio Salmerón

Dpt. Mathematics
University of Almería
La Cañada de San Urbano s/n
04120 Almería, Spain

Abstract

Probability trees are a powerful data structure for representing probabilistic potentials. However, their complexity can become intractable if they represent a probability distribution over a large set of variables. In this paper, we study the problem of decomposing a probability tree as a product of smaller trees, with the aim of being able to handle bigger probabilistic potentials. We propose exact and approximate approaches and evaluate their behaviour through an extensive set of experiments.

*Corresponding author

Email addresses: irene@ual.es (Irene Martínez), smc@decsai.ugr.es (Serafín Moral), crt@ual.es (Carmelo Rodríguez), antonio.salmeron@ual.es (Antonio Salmerón)

Preprint submitted to Applied Mathematics and Computation *October 9, 2013*

Key words: probability trees, multiplicative factorisation, approximation

1. Introduction

Probability trees [1] constitute a flexible and compact data structure used for representing probabilistic potentials (i.e. functions representing probabilistic information). They are especially useful in contexts where large probability distributions are handled, being Bayesian networks a remarkable example [6, 7, 14].

In scenarios of high complexity, representing probabilistic potentials in a factorised way can make difficult problems become tractable from a computational point of view, since a factorised representation is typically more compact than a joint one.

There are various ways of obtaining decompositions of probabilistic potentials. Canonical models [8] constitute a group of schemes for obtaining additive factorisations of conditional probability distributions. A more general approach is based on tensor decompositions [15, 16], where probabilistic potentials are approximated by a sum of potentials of the same arity, but each one of them expressed as a product of univariate functions.

The problem of obtaining multiplicative factorisations has been previously studied in the literature [12], being the most recent contribution the so-called *fast-factorisation* [4]. Though fast factorisation has the advantage of efficiency, as it can be computed quickly, it is only able to benefit of rather restrictive scenarios, namely those in which a potential can be decomposed as the product of two functions, one of them containing only one variable. In this paper, we follow the original ideas in [12] and develop the concept of exact and approximate factorisation of probability trees as a product of smaller trees. Our proposal is more general than fast factorisation and provides more accurate approximations than the method in [12].

The methods studied in this paper can be embedded in a more general data structure for representing probabilistic potentials, called *recursive probability trees* [2]: a factorised representation of a probability tree is suitable to be stored inside a recursive tree as a *list node* [3].

The rest of the paper is structured as follows: Section 2 is devoted to introduce the use of probability trees to represent potentials; in Section 3, exact procedures to decompose a potential as a product of two probability trees are given; Sections 4 and 5 describe approximate factorisation procedures; an experimental evaluation of the different decomposition methods is reported in Section 6; the paper ends with the conclusions in Section 7.

2. Probability trees

We will use the concept of *potential* to represent probabilistic information (including ‘a priori’, conditional and ‘a posteriori’ distributions and intermediate results of operations between them). A *potential* ϕ for a set of variables $\mathbf{X}$ is a mapping $\phi : \Omega_{\mathbf{X}} \rightarrow \mathbb{R}_0^+$, where $\mathbb{R}_0^+$ is the set of non-negative real numbers and $\Omega_{\mathbf{X}}$ is the set of possible cases of the set of variables $\mathbf{X}$. From now onwards, we will consider only discrete variables with a finite number of cases, and the *size* of a potential will be the highest number of values necessary to completely specify it; i.e. if ϕ is defined on $\Omega_{\mathbf{X}}$, its size is $|\Omega_{\mathbf{X}}|$. In this paper we are concerned with the representation of probabilistic potentials by means of probability trees.

A *probability tree* [1, 5, 14] is a directed labeled tree, where each internal node represents a variable and each leaf node represents a probability value. Each internal node has one outgoing arc for each state of the variable associated with that node. Each leaf contains a non-negative real number. The *size* of a tree $\mathcal{T}$, denoted as size($\mathcal{T}$), is defined as its number of leaves.

A probability tree $\mathcal{T}$ on variables $\mathbf{X}_I = \{X_i | i \in I, I \subset \mathbb{N}\}$ *represents a potential* $\phi : \Omega_{\mathbf{X}_I} \rightarrow \mathbb{R}_0^+$ if for each $\mathbf{x}_I \in \Omega_{\mathbf{X}_I}$ the value $\phi(\mathbf{x}_I)$ is the number stored in the leaf node that is reached by starting from the root node and selecting the child corresponding to coordinate x_i for each internal node labeled with X_i.

Figure 1 shows a potential ϕ and its representation using a probability table and a probability tree. The tree contains the same information as the table, but using five values instead of eight. Furthermore, trees allow to obtain even more compact representations in exchange of losing accuracy. This is achieved by pruning some leaves and replacing them by their average value -see [6] for details- as it is shown in the rightmost tree.

2.1. Operations over probability trees

Typical operations over probabilistic potentials required for probability calculus are *combination* (product) and *marginalisation* (projection). Both can be defined over probability trees.

The *combination* of two trees is done recursively. In each recursion step one of the trees is selected, and each child of its root node is combined with the other tree, and so on. This operation is illustrated in Figure 2 where the symbol $\otimes$ denotes the combination operator. The details of the algorithm can be found in [6].

Marginalisation is an operation to remove a variable by summing up over all its possible values. A variable is *marginalised out* from a probability tree by replacing it by the *addition* of its subtrees (corresponding to each of the branches starting from the variable). The *addition* of two probability trees is similar to the combination operation, and is also detailed in [6]. The marginalisation of a variable by summing probability trees is described in Figure 3 where symbol $\oplus$ represents the addition operator.

A third operation, called restriction, is also necessary to specify the tecniques in the following sections.

Definition 1 (Restriction). *Let $\mathcal{T}$ be a probability tree, $\mathbf{X}_J$ a set of variables, and $\mathbf{x}_J$ a configuration of values of the variables in $\mathbf{X}_J$. We define the* restriction of $\mathcal{T}$ to the values $\mathbf{x}_J$, *$\mathcal{T}^{R(\mathbf{X}_J=\mathbf{x}_J)}$, as the tree obtained by substituting in $\mathcal{T}$ every node corresponding to variable $X_k \in \mathbf{X}_J$ by the subtree $\mathcal{T}_k$ which is given by the value of $X_k=x_k$.*

Note that if $\mathbf{X}_J$ contains all of the variables between the root node and a leaf, $\mathcal{T}^{R(\mathbf{X}_J=\mathbf{x}_J)}$ represents the probability value in the leaf. On the other hand, if the configuration of variables is empty, $\mathcal{T}^{R(\emptyset)}$ equals $\mathcal{T}$. This operation is illustrated in Figure 4.

3. Factorisation of probability trees

We will distinguish between *exact* and *approximate* factorisation, depending on whether or not a probability tree can be decomposed, without loss of accuracy, into a product of smaller trees.

3.1. Exact factorisation of probability trees

The issue of decomposing probability trees can be characterised by the following definitions.

Definition 2 (Proportional tree below a variable). *Let $\mathcal{T}$ be a probability tree. Let $(\mathbf{X}_C = \mathbf{x}_C)$ be a configuration of variables leading from the root node in $\mathcal{T}$ to a variable X. We say that $\mathcal{T}$ is* proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$ *if for every $x_i, x_j \in \Omega_X$, $\exists \pi_{ji} > 0$ such that*

$$\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)} = \pi_{ji} \cdot \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)} \quad . \tag{1}$$

The value π_{ji} is called proportionality factor *of the subtree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}$ with respect to the subtree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}$. All the proportionality factors can be collected in a matrix $\boldsymbol{\pi} = \big(\pi_{ji}\big)$ which will be called* proportionality matrix. *Obviously, $\pi_{ii} = 1$ for each i, and $\pi_{ij} = 1/\pi_{ji}$ for each i, j.*

The tree shown in Figure 5 is proportional below X within context $W = 0$, because all the subtrees under X are proportional among them, and the proportionality matrix $\boldsymbol{\pi}$ is equal to:

$$\begin{pmatrix} 1 & 2 & 4 \\ \frac{1}{2} & 1 & 2 \\ \frac{1}{4} & \frac{1}{2} & 1 \end{pmatrix}$$

Definition 3 (Exact factorisation). *Let $\mathcal{T}$ be a probability tree proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors given by $\boldsymbol{\pi} = \big(\pi_{ji}\big)$. Given $x_i \in \Omega_X$, we define a* factor subtree *as any subtree $\mathcal{T}_i = \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}$. The* exact factorisation *of $\mathcal{T}$ respect to the factor subtree $\mathcal{T}_i$ is defined as the product*

$$\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \mathcal{T}_i) \otimes \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \mathcal{T}_i) \tag{2}$$

where:

- *$\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \mathcal{T}_i)$, called* core term of *$\mathcal{T}$ in the factorisation, is the tree obtained from $\mathcal{T}$ by replacing each subtree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}$ by its proportionality factor π_{ji} respect to the factor subtree $\mathcal{T}_i$.*

- *$\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \mathcal{T}_i)$, called* free term of *$\mathcal{T}$ in the factorisation, is the tree obtained from $\mathcal{T}$ by replacing subtree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C)}$ by $\mathcal{T}_i$, and any other subtree $\mathcal{T}^{R(\mathbf{X}_D=\mathbf{x}_D)}$ by a constant 1, for any context $(\mathbf{X}_D = \mathbf{x}_D)$ incompatible with $(\mathbf{X}_C = \mathbf{x}_C)$.*

Proposition 1. *Let $\mathcal{T}$ be a probability tree proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors given by $\boldsymbol{\pi} = \big(\pi_{ji}\big)$. Consider $x_i \in \Omega_X$, $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \mathcal{T}_i)$ and $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \mathcal{T}_i)$ as in Definition 3. Then*

$$\mathcal{T} = \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \mathcal{T}_i) \otimes \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \mathcal{T}_i). \tag{3}$$

Proof: The proof is straightforward following the combination operation over probability trees described in Section 2.1. $\square$

Figure 6 shows the resulting trees after applying *exact factorisation* to the probability tree in Figure 5 (proportional below X for context $W = 0$) with respect to the factor subtree $\mathcal{T}^{R(W=0,X=0)}$.

Observe that the values in the i-th row of $\boldsymbol{\pi}$ are the leaves below variable X in the *core term* when decomposing respect to the *factor subtree* $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}$. The *core term* has been obtained by replacing in $\mathcal{T}$ the subtrees $\mathcal{T}^{R(W=0,X=0)}$ by $\pi_{11} = 1$, $\mathcal{T}^{R(W=0,X=1)}$ by $\pi_{12} = 2$, and $\mathcal{T}^{R(W=0,X=2)}$ by $\pi_{13} = 4$. The rest of $\mathcal{T}$ remains unchanged. On the other hand, the *free term* is constructed from $\mathcal{T}$ replacing $\mathcal{T}^{R(W=0)}$ by the *factor subtree*, and a constant value 1 for the other contexts.

Notice that the *core* and *free terms* are both probability trees with size smaller than $\mathcal{T}$. Because of that, the nearer to the leaves of the tree the easier it is to detect proportionality inside a tree, since the *factor trees* will have smaller size. However, the closer we are to the root, the *core term* become smaller and, as this is usually the largest factor, the factorisation will be more effective, in the sense that it will produce small factors in relation to the original tree.

3.2. Exact factorisation with average free term

The exact factorisation described in 3.1 may introduce high values in the potential represented by the *core term* (see Figure 6 for instance in which a value of 4 is introduced). These values can be problematic in operations like tree pruning [6], in which a node in the tree is replaced by the average of its leaves if the potential represented by the new approximate tree minimises the distance to the original potential (Figure 1). As a result of using factorisation, the information measures obtained during these operations can be distorted by values highly distant in magnitude to the others. With the aim of avoiding this problem, in this section we propose a new factorisation strategy: instead of factorising with respect to an arbitrary subtree, factorise with respect to the average of the proportional subtrees.

Definition 4 (Average factor subtree). *Let $\mathcal{T}$ be a probability tree proportional below X for context $(\mathbf{X}_C = \mathbf{x}_C)$. We define the* average factor subtree*, denoted by $\bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)}$, as the subtree obtained averaging over all the subtrees proportional below context $(\mathbf{X}_C = \mathbf{x}_C)$, that is,*

$$\bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)} = \frac{1}{|\Omega_X|} \sum_{x_j \in \Omega_X} \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}. \tag{4}$$

Definition 5 (Exact factorisation with average free term). *Let $\mathcal{T}$ be a probability tree proportional below X for context $(\mathbf{X}_C = \mathbf{x}_C)$. We define the* exact factorisation of $\mathcal{T}$ with average free term *as the product*

$$\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \bar{\mathcal{T}}) \otimes \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \bar{\mathcal{T}}),$$

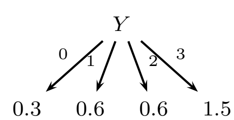
where:

- *The free term, $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \bar{\mathcal{T}})$, is computed as the tree obtained from $\mathcal{T}$ by replacing $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C)}$ by $\bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)}$ and replacing $\mathcal{T}^{R(\mathbf{X}_D=\mathbf{x}_D)}$ by a 1 for every context $(\mathbf{X}_D = \mathbf{x}_D)$ incompatible with $(\mathbf{X}_C = \mathbf{x}_C)$.*

- *The core term, $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \bar{\mathcal{T}})$, is the tree obtained from $\mathcal{T}$ by replacing each subtree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)}$ by a constant π_j given by:*

$$\pi_j = \frac{1}{\bar{\pi}_{\cdot j}} = \frac{|\Omega_X|}{\sum_{k:x_k\in\Omega_X} \pi_{kj}} \quad . \tag{5}$$

Figure 7 shows the *exact factorisation with average free term* of the tree in Figure 5 (proportional below X for context $W = 0$). In this case, the *average factor subtree* $\bar{\mathcal{T}}^{R(W=0)}$ is:



Theorem 1. *Let $\mathcal{T}$ be a probability tree proportional below X for context $(\mathbf{X}_C = \mathbf{x}_C)$ such that*

$$\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)} = \pi_{ji} \cdot \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)} \qquad x_i, x_j \in \Omega_X. \tag{6}$$

The average factor subtree $\bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)}$ *is proportional to all the subtrees $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)}$, and it holds that*

$$\mathcal{T} = \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \bar{\mathcal{T}}) \otimes \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \bar{\mathcal{T}}) \tag{7}$$

where $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \boldsymbol{\pi}, \bar{\mathcal{T}})$ and $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \bar{\mathcal{T}})$ are the core term and the free term given in Definition 5.

Proof: For each $i : x_i \in \Omega_X$, we have that

$$\bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)} = \frac{1}{|\Omega_X|}\sum_{x_k\in\Omega_X}\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_k)} = \frac{1}{|\Omega_X|}\sum_{x_k\in\Omega_X}\pi_{ki}\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}$$
$$= \frac{\left(\sum_{x_k\in\Omega_X}\pi_{ki}\right)}{|\Omega_X|}\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)} = \bar{\pi}_{\cdot i}\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}. \tag{8}$$

Therefore, $\bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)}$ is proportional to each one of the subtrees $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}$. Furthermore, for each $i : x_i \in \Omega_X$, it follows from Eq. (8) that

$$\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)} = \frac{1}{\bar{\pi}_{\cdot i}}\bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)}.$$

Hence, there is a constant π_i such that

$$\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)} = \pi_i \cdot \bar{\mathcal{T}}^{R(\mathbf{X}_C=\mathbf{x}_C)},$$

and it follows that

$$\mathcal{T} = \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = \bar{x}, \boldsymbol{\pi}) \otimes \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, \bar{\mathcal{T}}).$$

$\square$

Corollary 1. *Let $\mathcal{T}$ be a probability tree proportional under a given context. Under the conditions in Theorem 1, it holds that the core term resulting from an* exact factorisation with average free term *of $\mathcal{T}$ can be obtained from the core term of an* exact factorisation *of $\mathcal{T}$ -defined in Definition 3- by dividing by the average of its leaves.*

Proof: For each $x_j \in \Omega_X$, denote $\mathcal{T}_j = \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}$. Let $\alpha_1, \alpha_2, \ldots, \alpha_n$, with $n = |\Omega_X|$, be the leaves of the *core term* of the *exact factorisation* with respect to the *factor subtree* $\mathcal{T}_i$. That is, for each $j : x_j \in \Omega_X$, $\alpha_j = \pi_{ji}$, since $\mathcal{T}_j = \alpha_j\mathcal{T}_i$. Furthermore, for each $k, j = 1, \ldots, n$ it holds that

$$\alpha_k\mathcal{T}_i = \mathcal{T}_k = \pi_{kj}\mathcal{T}_j = \pi_{kj}\alpha_j\mathcal{T}_i$$

and therefore

$$\alpha_k = \pi_{ki}\alpha_j \quad . \tag{9}$$

Hence, the leaves of the *core term* of the *exact average factorisation*, $\pi_1, \pi_2, \ldots, \pi_n$ can be computed, according to Eq. (9) as

$$\pi_i = \frac{1}{\bar{\pi}_{\cdot i}} = \frac{\alpha_i}{\alpha_i \bar{\pi}_{\cdot i}} = \frac{\alpha_i}{\dfrac{1}{n}\sum_{k=1}^{n} \pi_{ki}\alpha_i} = \frac{\alpha_i}{\dfrac{1}{n}\sum_{k=1}^{n} \alpha_k} = \frac{\alpha_i}{\bar{\alpha}} \quad i = 1, \ldots, n.$$

$\square$

Corollary 2. *The sum of the values stored in the leaves of the core term of the factorisation of $\mathcal{T}$ with average free term is equal to its number of leaves or equivalently, to the number of proportional subtrees before the factorisation in the given context.*

Proof: Let $n = |\Omega_X|$, $\mathcal{T}_j = \mathcal{T}^{R(\mathbf{X}_C = \mathbf{x}_C, X = x_j)}$ and $\bar{\mathcal{T}} = \bar{\mathcal{T}}^{R(\mathbf{X}_C = \mathbf{x}_C)}$. Let $SC(\bar{\mathcal{T}})$ be the sum of the values stored in the leaves of the *core term* of the *exact factorisation with average free term.* Then,

$$SC(\bar{\mathcal{T}}) = \sum_{j=1}^{n} \pi_j = \sum_{j=1}^{n} \frac{1}{\bar{\pi}_{\cdot j}}. \tag{10}$$

We know that $\mathcal{T}_j = \pi_{ji}\mathcal{T}_i$ for each $i, j = 1, \ldots, n$, and therefore

$$\pi_j \bar{\mathcal{T}} = \mathcal{T}_j = \pi_{ji}\mathcal{T}_i = \pi_{ji}\pi_i\bar{\mathcal{T}} \tag{11}$$

which implies that

$$\pi_j = \pi_{ji}\pi_i \quad i, j = 1, \ldots, n \tag{12}$$

and, particularly,

$$\pi_j = \pi_{j1}\pi_1 \quad j = 1, \ldots, n. \tag{13}$$

Using Eq. (13) in Eq. (10), we obtain

$$SC(\bar{\mathcal{T}}) = \sum_{j=1}^{n} \pi_j = \sum_{j=1}^{n} \pi_{j1}\pi_1 = \frac{1}{\bar{\pi}_{\cdot 1}} \sum_{j=1}^{n} \pi_{j1} = \frac{n}{\sum_{j=1}^{n} \pi_{j1}} \sum_{j=1}^{n} \pi_{j1} = n \ .$$

$\square$

4. Approximate factorisation of probability trees

There are situations in which the ways of decomposing trees described so far may be of interest, even if the conditions of proportionality are not met. For instance, assume that we have three variables X, Y and Z for which X is independent of Y given Z and therefore a potential representing the joint probability of X, Y, and Z can be decomposed as product of a potential depending on X, Z and other one depending on Y, Z. But assume that due to the fact that the actual distribution of X, Y and Z has been estimated from a sample, the learnt distribution is not exactly the same, but very close to the true one. The exact factorisation would not be discovered in the estimated potential. Another scenario in which one could be interested in decomposing a tree is when space limitations do not allow to represent fully expanded probability trees, and then it is necessary to tradeoff accuracy for space requirements. This happens, for instance, when probability trees are used for carrying out inference in Bayesian networks [4].

The problem of *approximate factorisation* can be stated as follows. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two subtrees which are siblings for a given context (i.e. both subtrees are children of the same node), such that both have the same size and structure and their leaves contain only positive numbers. The goal of the *approximate factorisation* is to find a tree $\mathcal{T}_2^*$ with the same structure as $\mathcal{T}_2$, such that $\mathcal{T}_2^*$ and $\mathcal{T}_1$ become proportional, under the restriction that the potential represented by $\mathcal{T}_2^*$ must be as close as possible to the one represented by $\mathcal{T}_2$. Then, $\mathcal{T}_2$ can be replaced by $\mathcal{T}_2^*$ and the resulting tree, containing $\mathcal{T}_1$ and $\mathcal{T}_2^*$, can be decomposed as it would become proportional for the given context.

In order to obtain the approximate tree $\mathcal{T}_2^*$, two main questions arise: the *determination of the proportionality factors* and the *assessment of the accuracy of the approximation*. Both questions are connected, since it seems sensible to select the proportionality factors in such a way that the chosen divergence measure is minimised. Approximate factorisation is formalised in the next definition.

Definition 6 (δ-factorisable tree). *We say that a probability tree $\mathcal{T}$ is δ-*factorisable *below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors $\boldsymbol{\pi}$, and with respect to a divergence measure $\mathcal{D}$, if for each $x_j, x_i \in \Omega_X$ $\exists \pi_{ji} > 0$ it holds that*

$$\mathcal{D}(\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)}, \pi_{ji} \cdot \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)}) \leq \delta.$$

Parameter $\delta > 0$ is called the tolerance of the approximation.

Note that *proportional trees* below X are δ-factorisable, with $\delta = 0$.

The *basic approximate factorisation* was introduced in [12], where the formulae for computing the proportionality factors $\boldsymbol{\pi}$ were given according to several methods generally based on minimising divergence measures. From these formulae an approximate tree is obtained and factorised in the same way as the *exact factorisation* method introduced in Proposition 3. Algorithm 1 implements this factorisation.

Algorithm 1 Basic Approximate factorisation

1: **procedure** BasicApproxFactorisation($\mathcal{T}$,X,($\mathbf{X}_C = \mathbf{x}_C$),$\delta$,$\mathcal{D}$)

2: Let $\mathcal{T}_j = \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)}$, $j : x_j \in \Omega_X$, be the subtrees of $\mathcal{T}$ δ-factorisable below X for context ($\mathbf{X}_C = \mathbf{x}_C$), respect to divergence $\mathcal{D}$.

3: Select $\mathcal{T}_i = \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)}$, $i : x_i \in \Omega_X$, as the *factor subtree*.

4: Obtain $\boldsymbol{\pi}$, the matrix of proportionality factors, from the individual coefficients between the leaves in trees $\mathcal{T}_j$ respect to the *factor subtree*, using the formulae in [12].

5: Compute the approximate subtrees $\mathcal{T}_j^* = \mathcal{T}^{*R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)}$ proportional amongst them, with proportionality factors $\boldsymbol{\pi}$.

6: Apply the *exact factorisation* in Proposition 3, using subtrees $\mathcal{T}_j^*$.

Proportionality factors in step 4 are calculated under the restriction of minimising different measures of divergence, in such a way that if $\mathcal{T}_1$ and $\mathcal{T}_2$ are subtrees of $\mathcal{T}$ below a variable X, with leaves $\mathcal{P} = \{p_i : i = 1, \ldots, n; p_i \neq 0\}$ and $\mathcal{Q} = \{q_i : i = 1, \ldots, n\}$ respectively, and the divergence measure considered is, for instance, the χ^2 divergence, the distance between $\mathcal{T}_2$ and its approximate tree $\mathcal{T}_2^*$ can be defined as

$$\mathcal{D}_\chi(\mathcal{T}_2, \mathcal{T}_2^*) = \sum_{i=1}^{n} \frac{(q_i - \alpha p_i)^2}{q_i} \quad , \tag{14}$$

and is minimised for α equal to $\frac{\sum_{i=1}^n p_i}{\sum_{i=1}^n p_i/\phi_i}$, where $\phi_i = q_i/p_i, i = 1, \ldots, n$.

Other divergence measures, as the mean squared error (MSE), Kullback-Leibler or Hellinger divergence, and their corresponding proportionality factors, are considered in [12]. In general, different divergence measures yield different proportionality factors. Notice that using the MSE as divergence measure is equivalent to selecting the proportionality factors that minimise the Euclidean distance between the exact and approximate trees, and it can be solved using the Singular Value Decomposition (SVD) technique employed in Principal Component Analysis (see, for instance [10, pages 534-536]).

However, the Euclidean distance is not necessarily the most appropriate divergence measure from a statistical point of view. For instance, the Kullback-Leibler divergence has a probabilistic interpretation as an expected value, and is related to commonly used statistical procedures as the maximum likelihood estimation method.

5. Approximate factorisation with average free term

Another approach for carrying out approximate factorisation is to obtain an approximate tree, which may become factorisable, using an *average factor subtree* during the computation. Given a tree $\mathcal{T}$ δ-factorisable below X for context $(\mathbf{X}_C = \mathbf{x}_C)$, with respect to a divergence measure $\mathcal{D}$, we propose two methods for factorising it in an approximate way with respect to an average free term:

1. The approximate proportional subtrees below X are computed and then, the *exact factorisation with average free term* is applied. We shall refer to this strategy as **approximate-first factorisation with average free term**.
2. The *average factor subtree* is computed from the original subtrees below X, and the proportionality factors $\boldsymbol{\pi}$ are obtained by any approximation method, with respect to the *average factor subtree*, i.e. firstly, the subtrees are averaged and secondly, the best proportionality factors for approximating each of the subtrees are computed. We shall refer to this approach as **average-first factorisation with average free term**.

5.1. *Approximate-first factorisation with average free term*

Let $\mathcal{T}$ be a tree δ-factorisable below X for context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors $\boldsymbol{\pi}$ with respect to a divergence measure $\mathcal{D}$. Algorithm 2 implements the *Approximate-first factorisation with average free term.*

Algorithm 2 Approximate-First factorisation with average free term

1: **procedure** ApproxFirstAFT($\mathcal{T}$,X,($\mathbf{X}_C = \mathbf{x}_C$),$\delta$,$\mathcal{D}$)

2: Let $\mathcal{T}_j = \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}$, $j : x_j \in \Omega_X$, be the subtrees of $\mathcal{T}$ δ-factorisable below X for context ($\mathbf{X}_C = \mathbf{x}_C$), respect to divergence $\mathcal{D}$.

3: Select $\mathcal{T}_i = \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}$, $i : x_i \in \Omega_X$, as the *factor subtree*

4: Obtain $\boldsymbol{\pi}$, the matrix of proportionality factors, from the individual coefficients between the leaves in subtrees $\mathcal{T}_j$ respect to $\mathcal{T}_i$.

5: Compute the approximate subtrees $\mathcal{T}_j^* = \mathcal{T}^{*R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}$ proportional amongst them, with proportionality factors $\boldsymbol{\pi}$.

6: Apply the *exact factorisation with average free term* using subtrees $\mathcal{T}_j^*$.

This method computes the approximate proportional subtrees $\mathcal{T}^*$ in the same way as in [12]. The difference between them is the last step in Algorithms 1 and 2. In both cases the factorisation is exact, but the first one carries out exact factorisation respect to a *factor subtree*, and the second performs the factorisation respect to the *average factor subtree*. Due to the fact that in both cases the factorisation is exact, they give rise to the same distance from the combination of the factors to the original tree.

Example 1. *Figure 8 shows a probability tree δ-factorisable below X for context $W = 0$.*

Figures 9 and 10 show the subtrees of X for context $W = 0$ approximately proportional according to the invariant potential *method*[1]*, and the approximate factorisation resulting by applying* approximate-first factorisation *to the tree in Figure 8.*

5.2. Average-first factorisation with average free term

Let $\mathcal{T}$ be a tree δ-factorisable below X for context ($\mathbf{X}_C = \mathbf{x}_C$) with respect to a divergence measure $\mathcal{D}$. Algorithm 3 implements the *average-first factorisation with average free term.* In this case, the *average factor*

[1]Introduced in [12]. With this method the weights of the original and the approximate tree coincide: $\mathrm{sum}(\mathcal{T}_2^*) = \sum_{i=1}^n \alpha p_i = \sum_{i=1}^n q_i = \mathrm{sum}(\mathcal{T}_2)$, and $\alpha = \frac{\sum_{i=1}^n q_i}{\sum_{i=1}^n p_i}$.

subtree $\bar{\mathcal{T}}$ is computed before, and it is used as a *factor subtree* in the step 3 within the *basic approximate factorisation* in Algorithm 1. Therefore, the leaves of the core term are calculated using the individual proportionality coefficients between the leaves of each subtree $\mathcal{T}_i$ and $\bar{\mathcal{T}}$.

Algorithm 3 Average-first factorisation with average free term

1: **procedure** Average-FirstAFT($\mathcal{T}$, X, $(\mathbf{X}_C = \mathbf{x}_C)$, δ, $\mathcal{D}$)

2: Let $\mathcal{T}_j = \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)}$, $j : x_j \in \Omega_X$ be the subtrees of $\mathcal{T}$ δ-factorisable below X for context $(\mathbf{X}_C = \mathbf{x}_C)$, respect to divergence $\mathcal{D}$.

3: Compute the *average factor subtree* $\bar{\mathcal{T}}$ as the average of subtrees $\mathcal{T}_j$.

4: Apply *basic approximate factorisation* in Algorithm 1 to subtrees $\mathcal{T}_j$ with respect to $\bar{\mathcal{T}}$, that is, in step 3 select $\bar{\mathcal{T}}$ instead of $\mathcal{T}_i$.

Example 2. *Consider the tree in Figure 8, δ-factorisable below X for context $W = 0$. The* average factor subtree *for the δ-proportional subtrees below X is displayed in Figure 11.*

Figure 12 shows the *average-first factorisation with average free term* of the tree in Figure 8 using the method of minimum χ^2 divergence.

6. Experimental evaluation

6.1. Experiments with simulated trees

In this section we describe a set of experiments carried out to illustrate and evaluate the impact of using approximate factorisation in terms of the error of the corresponding approximation.

- The first experiment was run over the tree in Figure 8. Table 1 shows the leaves of the *core* and *free terms* of the different decompositions resulting from applying seven different approximation methods combined with *Basic Approximate factorisation* (BF), shown in Algorithm 1, *Approximate-First factorisation with average free term* (AF), and *Average-First factorisation with average free term* (VF). The methods used for obtaining the approximate factorisations were (see [12] for details): The invariant potential, the minimum χ^2-divergence (D_χ), the minimum mean squared error (MSE), the minimum weighted mean squared error (WMSE), the null Kullback-Leibler divergence (NKL), the minimum weighted average method (WA), in which the proportionality factor is computed as a weighted average of the ratios between the leaves of $\mathcal{T}_1$ and $\mathcal{T}_2$, and the minimum Hellinger distance (Hell).

Next, Table 2 displays the distances between the tree in Figure 8 and the different factorisations obtained. The considered distances have been: the extended Kullback-Leibler divergence (KL), the maximum absolute difference (MAXD) and the mean absolute difference (MAD) between the leaves of $\mathcal{T}_2$ and $\mathcal{T}_2^*$, the χ^2-divergence (D_χ), the normalised χ^2 divergence (ND_χ), the mean squared error (MSE), the weighted mean squared error (WMSE), and the Hellinger distance (Hell). Note that, obviously, the distances are the same for the *approximate factorisation* and the *approximate-first factorisation with average free term.* However, the application of the *average-first factorisation with average free term* shows remarkable reductions in the errors in most of the cases.

- As a second experiment, we have considered the case of a tree clearly far away from proportionality. More precisely, we have used the tree in Figure 13. The results of the different factorisations are displayed in Table 3. Again, it can be seen that the method of *average-first factorisation* clearly outperforms the *approximate-first factorisation.*

- Finally, in a third experiment, we have run a simulation over randomly generated trees with various features, and in each case, we have annotated which factorisation strategy (*approximate-first* factorisation or *average-first* factorisation) provides the best results. More precisely, we have considered three scenarios in this third experiment:

  1. 10000 runs, each one with a set of m subtrees (m generated at random between 2 and 102) with n leaves (n generated at random between 2 and 52), each leaf with a random real number between 0 and 10. Table 4 shows the proportion of runs in which the error of the *average-first* factorisation is lower, for the different divergence measures considered.
  2. The same settings as in the preceding scenario, but the leaves in each tree, instead of containing random numbers, contain real

numbers in increasing order. The results for this scenario can be seen in Table 5.

3. The same settings as in the previous scenarios, but in this case the leaves are generated in such a way that the resulting trees are δ-factorisable with $\delta = 0.01$. In this case, in all the runs the *average-first* method was more accurate than the *approximate-first* one.

The experiments described above show how the *Average-First factorisation with average free term* method is in general superior to *Approximate-First factorisation with average free term*. The extreme case is when the subtrees are δ-proportional, when all the runs showed a better accuracy of the VF method.

6.2. Experiments with real world problems

In order to test the behaviour of the factorisation techniques proposed in this paper when applied to distributions coming from real world domains, we have considered the task of computing the posterior distribution of the variables in two Bayesian networks that describe actual scenarios, that we will denote as Munin1 [13] and Link [11]. We have represented the distributions in both networks as probability trees, and carried out the calculations using the so-called *Lazy-penniless* architecture [7], modified in such a way that the probability trees are factorised using both methods of *approximate factorisation with average free term*, AF and VF. In addition, the *Basic Approximate factorisation* (BF) in [12] has been also used with the aim of comparing the three methods (AF, VF and BF).

Two values for δ, the tolerance of the approximation, have been considered, and five different sets of observed variables (called evidence) are used with each. In all of them the same divergence measure (normalised $\mathcal{D}_\chi$) for obtaining the proportionality factors is used. The distance between each tree and its approximation is measured by the mean squared error.

The following tables show the error values obtained in the experiments. The error is computed by comparing the results of the propagation for the exact and the approximate computations for the marginals of all the variables in the network conditional on the observations. The tables represent the resulting values of mean squared error (MSE), max absolute error, Kullback-Leibler error, and G defined in [9] as:

$$G(X_l) = \sqrt{\frac{1}{|\Omega_{X_l}|} \sum_{a_l \in \Omega_{X_l}} \frac{(p'(a_l|\mathbf{e}) - p(a_l|\mathbf{e}))^2}{p(a_l|\mathbf{e})(1 - p(a_l|\mathbf{e}))}} \tag{15}$$

where X_l is a variable, $\mathbf{e}$ is the evidence, $p(a_l|\mathbf{e})$ is the true *(a posteriori)* probability and $p'(a_l|\mathbf{e})$ is the estimated value. Given a set of variables $\{X_1, \ldots, X_n\}$, the error is computed as:

$$G(\{X_1, \ldots, X_n\}) = \sqrt{\sum_{i=1}^{n} G(X_i)^2} \tag{16}$$

The results for the Munin1 network are shown in Tables 6, 7 and 8. As in the simulated trees in the former sections, the *Average First* method obtains better approximations than the other two.

The results for the Link network are displayed in Tables 9, 10 and 11. Again, we can see that the best measures are obtained with the *Average First Factorisation* method.

6.3. Decompositions representing a data sample

The goal of this experiment is to explore the relationship between the tolerance of an approximate factorisation and the likelihood of a sample associated to a probability tree. In this way, we can interpret the meaning of parameter δ when the initial tree is obtained from a data sample by counting the frequencies corresponding to each leaf. We considered an initial tree $\mathcal{T}$ and a data sample S corresponding to it. The experiment consisted of factorising $\mathcal{T}$ for a series of values δ and measuring the likelihood of data sample S for each one of the factorisations.

We considered four scenarios for this experiment:

1. An initial tree that admits a close-to-exact factorisation, using the KL divergence as divergence measure and also as method for computing the α coefficients in the factorisation. The results of this setting are displayed in Figure 14.
2. An initial tree that admits a close-to-exact factorisation, using the normalised χ^2 divergence as divergence measure and the weight preserving technique as method for computing the α coefficients in the factorisation. The results of this setting are displayed in Figure 15.

3. An initial tree with values generated at random, and therefore not likely to be factorisable, using the KL divergence as divergence measure and also as method for computing the α coefficients in the factorisation. The results of this setting are displayed in Figure 16.
4. An initial tree with values generated at random, using the normalised χ^2 divergence as divergence measure and the weight preserving technique as method for computing the α coefficients in the factorisation. The results of this setting are displayed in Figure 17.

In the four settings, the results obtained suggest that parameter δ can be used as a means of controlling the model accuracy, in terms of likelihood of the sample used.

7. Conclusions

In this paper we have proposed a new procedure to obtain approximate factorisations of probability trees. The old method was based on taking one of the subtrees as basis and then, approximate the other ones to this. The new methods takes the average of the subtrees as basis and approximate each subtree with respect to the average. We have carried out an extensive experimentation in which it is shown that the new procedure consistently produces better approximations.

As a line of future research, we plan to study the integration of the factorisation method proposed in this paper into algorithms for approximate inference in Bayesian networks, as a way of finding a tradeoff between accuracy of the final results and time invested in computing them.

Acknowledgements

This work has been supported by the Spanish Ministry of Economy and Competitiveness, under projects TIN2010-20900-C04-01,02, MTM2010-20774-C03-03 and by EFRD (FEDER) funds.

References

[1] Boutilier, C., Friedman, N., Goldszmidt, M., Koller, D., 1996. Context-specific independence in Bayesian networks. In: Horvitz, E., Jensen, F. (Eds.), Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence. Morgan & Kaufmann, pp. 115–123.

[2] Cano, A., Gómez-Olmedo, M., Moral, S., Pérez-Ariza, C., 2009. Recursive probability trees for Bayesian networks. CAEPIA 2009. Lecture Notes in Artificial Intelligence 5988, 242–251.

[3] Cano, A., Gómez-Olmedo, M., Moral, S., Pérez-Ariza, C., Salmerón, A., 2012. Learning recursive probability trees from probabilistic potentials. International Journal of Approximate Reasoning 53, 1367–1387.

[4] Cano, A., Gómez-Olmedo, M., Pérez-Ariza, C., Salmerón, A., 2012. Fast factorisation of probabilistic potentials and its application to approximate inference in Bayesian networks. International Journal of Uncertainty, Fuzziness and Knowledge Based Systems 20, 223–243.

[5] Cano, A., Moral, S., 1997. Propagación exacta y aproximada con árboles de probabilidad. In: Actas de la VII Conferencia de la Asociación Española para la Inteligencia Artificial. pp. 635–644.

[6] Cano, A., Moral, S., Salmerón, A., 2000. Penniless propagation in join trees. International Journal of Intelligent Systems 15, 1027–1059.

[7] Cano, A., Moral, S., Salmerón, A., 2002. Lazy evaluation in Penniless propagation over join trees. Networks 39, 175–185.

[8] Díez, F., Druzdzel, M., 2001. Fundamentals of canonical models. In: Proceedings of the 9th Conference of the Spanish Assocciation for Artificial Intelligence (CAEPIA-TTIA'2001). pp. 1125–1134.

[9] Fertig, K., Mann, N., 1980. An accurate approximation to the sampling distribution of the studentized extreme-valued statistic. Technometrics 22, 83–90.

[10] Hastie, T., Tibshirani, R., Friedman, J., 2009. The elements of statistical learning. Data mining, inference and prediction, second edition Edition. Springer.

[11] Jensen, C., Kong, A., Kjærulff, U., 1995. Blocking Gibbs sampling in very large probabilistic expert systems. International Journal of Human-Computer Studies 42, 647–666.

[12] Martínez, I., Moral, S., Rodríguez, C., Salmerón, A., 2005. Approximate factorisation of probability trees. ECSQARU'05. Lecture Notes in Artificial Intelligence 3571, 51–62.

[13] Olesen, K., Kjærulff, U., Jensen, F., Jensen, F., 1989. A munin network for the median nerve - a case study on loops. Applied Artificial Intelligence 3, 385–404.

[14] Salmerón, A., Cano, A., Moral, S., 2000. Importance sampling in Bayesian networks using probability trees. Computational Statistics and Data Analysis 34, 387–413.

[15] Savický, P., Vomlel, J., 2007. Exploiting tensor rank-one decomposition in probabilistic inference. Kybernetika 43 (5), 747–764.

[16] Vomlel, J., 2011. Rank of tensors of l-out-of-k functions: an application in probabilistic inference. Kybernetika 47 (3), 317–336.

List of Tables

Tables

Table 1: Resulting trees after factorising the tree in Figure 8 in the first experiment.

Method	Factor	Core Term			Free Term			
Inv Pot	BF	1.00000000	2.00000000	5.99910000	0.10000000	0.20000000	0.20000000	0.50000000
	AF	0.33336667	0.66673334	1.99989999	0.29997000	0.59994000	0.59994000	1.49985000
	VF	0.33336667	0.66673334	1.99989999	0.30000000	0.59800000	0.60006667	1.50163333
D_χ	BF	1.00000000	1.99999984	5.99906166	0.10000000	0.20000000	0.20000000	0.50000000
	AF	0.33336810	0.66673614	1.99989576	0.29996872	0.59993743	0.59993743	1.49984358
	VF	0.33336573	0.66673136	1.99989853	0.30000000	0.59800000	0.60006667	1.50163333
MSE	BF	1.00000000	1.99994118	6.00385294	0.10000000	0.20000000	0.20000000	0.50000000
	AF	0.33319287	0.66636614	2.00044099	0.30012647	0.60025294	0.60025294	1.50063235
	VF	0.33319220	0.66636477	2.00044303	0.30000000	0.59800000	0.60006667	1.50163333
WMSE	BF	1.00000000	1.99987328	6.00713438	0.10000000	0.20000000	0.20000000	0.50000000
	AF	0.33307399	0.66610578	2.00082023	0.30023359	0.60046718	0.60046718	1.50116794
	VF	0.33307418	0.66610616	2.00082136	0.30000000	0.59800000	0.60006667	1.50163333
NKL	BF	1.00000000	2.00000008	5.99911913	0.10000000	0.20000000	0.20000000	0.50000000
	AF	0.33336596	0.66673194	1.99990210	0.29997064	0.59994128	0.59994128	1.49985320
	VF	0.33336714	0.66673433	1.99990072	0.30000000	0.59800000	0.60006667	1.50163333
WA	BF	1.00000000	2.00000016	5.99913822	0.10000000	0.20000000	0.20000000	0.50000000
	AF	0.33336525	0.66673055	1.99990420	0.29997128	0.59994256	0.59994256	1.49985640
	VF	0.33336761	0.66673533	1.99990144	0.30000000	0.59800000	0.60006667	1.50163333
Hell	BF	1.00000000	1.99999996	5.99909043	0.10000000	0.20000000	0.20000000	0.50000000
	AF	0.33336703	0.66673404	1.99989894	0.29996968	0.59993936	0.59993936	1.49984840
	VF	0.33336644	0.66673284	1.99989963	0.30000000	0.59800000	0.60006667	1.50163333

Table 2: Distances from the tree in Figure 8 to the decomposition, for the factorisations in the first experiment.

		Distance							
Method	Factor	KL	MAXD	MAD	D_χ	ND_χ	MSE	WMSE	Hellinger
Inv Pot	BF/AF	0.0000192	0.0058200	0.0010033	0.0062049	0.0017912	0.0023040	0.0010963	0.0031000
	VF	0.0000066	0.0019402	0.0006782	0.0036227	0.0010458	0.0009777	0.0004057	0.0018118
D_χ	BF/AF	0.0000192	0.0058123	0.0010052	0.0062048	0.0017912	0.0023063	0.0010985	0.0031000
	VF	0.0000066	0.0019393	0.0006780	0.0036227	0.0010458	0.0009776	0.0004058	0.0018118
MSE	BF/AF	0.0000211	0.0067706	0.0009407	0.0065060	0.0018781	0.0021607	0.0008815	0.0032489
	VF	0.0000072	0.0022649	0.0006291	0.0038005	0.0010971	0.0009155	0.0003251	0.0019010
WMSE	BF/AF	0.0000246	0.0074269	0.0009395	0.0070265	0.0020284	0.0022302	0.0008318	0.0035081
	VF	0.0000085	0.0024912	0.0006273	0.0041139	0.0011876	0.0009462	0.0003062	0.0020580
NKL	BF/AF	0.0000192	0.0058238	0.0010024	0.0062049	0.0017912	0.0023029	0.0010952	0.0031000
	VF	0.0000066	0.0019406	0.0006783	0.0036227	0.0010458	0.0009777	0.0004056	0.0018118
WA	BF/AF	0.0000192	0.0058276	0.0010014	0.0062049	0.0017912	0.0023018	0.0010941	0.0031000
	VF	0.0000066	0.0019411	0.0006784	0.0036227	0.0010458	0.0009777	0.0004055	0.0018118
Hell	BF/AF	0.0000192	0.0058181	0.0010038	0.0062049	0.0017912	0.0023046	0.0010968	0.0031000
	VF	0.0000066	0.0019400	0.0006782	0.0036227	0.0010458	0.0009777	0.0004057	0.0018118

Table 3: Errors when decomposing the tree in Figure 13.

		Distance							
Method	Factor	KL	MAXD	MAD	D_χ	ND_χ	MSE	WMSE	Hellinger
Inv Pot	BF/AF	17.6271158	4.5937500	4.6656250	9.3514345	0.8369031	5.7766857	3.2057776	3.8633656
	VF	4.4325765	2.0516854	2.7764045	6.7231720	0.7918182	3.1810756	2.1148633	2.6088844
D_χ	BF/AF	26.8345212	3.9569651	2.9825582	4.6503537	0.7332061	4.1742393	2.9791542	3.8164150
	VF	7.4003924	2.8321153	2.4575399	4.5620070	0.7299451	3.6972862	2.8458906	2.8697676
MSE	BF/AF	24.9399634	3.9475232	3.0877709	4.6808889	0.7343144	4.1637427	2.9608635	3.7381687
	VF	4.4418948	1.9868421	2.7785559	6.6156588	0.7894290	3.1782002	2.1209308	2.6027889
WMSE	BF/AF	21.2652928	3.9173577	3.4226413	5.1149957	0.7491079	4.2532739	2.9380192	3.6140052
	VF	6.4609504	2.1330226	3.5750888	12.3561437	0.8691626	3.9955638	1.6675103	3.4200239
NKL	BF/AF	38.9280129	27.2894013	12.4101050	39.6253280	0.9530531	19.9734791	7.8843081	7.2223377
	VF	5.3858799	1.6573232	3.2849139	10.2455820	0.8480632	3.5430334	1.7377670	3.0728248
WA	BF/AF	81.2939064	61.0389706	23.7212745	80.9508613	0.9761731	39.7949666	15.2310088	10.5982992
	VF	6.7421892	2.2175210	3.6513365	12.8202284	0.8730356	4.1151141	1.6719638	3.5025408
Hell	BF/AF	19.9697399	3.9047909	3.6349700	5.6099425	0.7640448	4.4151416	2.9509875	3.5931614
	VF	4.6662897	2.2685613	2.5794264	5.5987191	0.7637262	3.2356601	2.3250107	2.5493663

Table 4: Proportion of experiments where the average-first factorisation is more accurate in Scenario 1.

Method	KL	MAXD	MAD	D_χ	ND_χ	MSE	WMSE	Hellinger
Inv Pot	100	97.88	89.19	93.43	93.43	90	90.06	93.39
D_χ	100	97.97	89.16	93.28	93.28	90	90.12	93.47
MSE	100	97.85	89.22	93.33	93.33	89.99	90.12	93.38
WMSE	100	97.76	89.16	93.4	93.4	90	90.11	93.33
NKL	100	97.92	89.2	93.63	93.63	90.08	90.12	93.47
WA	100	97.82	89.34	93.85	93.85	90.27	90.22	93.59
Hell	100	97.88	89.13	93.38	93.38	89.97	90.06	93.39

Table 5: Proportion of experiments where the average-first factorisation is more accurate in Scenario 2.

Method	KL	MAXD	MAD	D_χ	ND_χ	MSE	WMSE	Hellinger
Inv Pot	100	99.98	99.98	99.56	99.56	99.96	99.96	99.95
D_χ	100	99.97	99.98	99.94	99.94	99.96	99.96	99.95
MSE	100	99.98	99.98	98.99	98.99	99.96	99.96	99.95
WMSE	100	99.99	99.97	97.17	97.17	99.96	99.96	99.95
NKL	100	99.99	99.97	99.95	99.95	99.96	99.96	99.95
WA	100	99.99	99.97	99.97	99.97	99.96	99.96	99.95
Hell	100	99.98	99.98	99.56	99.56	99.96	99.96	99.95

Table 6: Munin1 errors using *Average First Factorisation* (VF).

Evidence	δ	G	MSE	Max Abs Error	KL error
1	0.001	8.99E-07	2.72E-17	9.40E-08	9.22E-15
2	0.001	0.04825902	7.56E-08	0.00189964	9.44E-06
3	0.001	1.35E-04	1.26E-14	1.41E-06	1.71E-10
4	0.001	0.030646	9.84E-07	0.00919359	1.54E-05
5	0.001	0.00712411	1.12E-11	5.70E-05	1.68E-07
1	0.01	2.13E-04	3.08E-16	4.10E-07	6.99E-09
2	0.01	2.29935664	5.50E-04	0.16250689	0.00292949
3	0.01	4.37E-04	1.37E-14	1.41E-06	6.48E-10
4	0.01	0.18722285	3.91E-05	0.0715271	1.37E-04
5	0.01	0.00714148	1.41E-11	5.70E-05	1.71E-07

Table 7: Munin1 errors using *Approximate First Factorisation* (AF).

Evidence	δ	G	MSE	Max Abs Error	KL error
1	0.001	2.18E-04	7.56E-16	7.79E-07	7.43E-09
2	0.001	0.17134741	1.20E-05	0.09472202	4.93E-04
3	0.001	1.66E-04	3.29E-14	3.10E-06	4.17E-10
4	0.001	0.14061221	2.46E-05	0.0674813	2.83E-04
5	0.001	0.00324477	1.83E-10	1.70E-04	8.19E-08
1	0.01	2.18E-04	7.56E-16	7.79E-07	7.43E-09
2	0.01	0.17896897	1.21E-05	0.09472651	5.02E-04
3	0.01	1.66E-04	3.33E-14	3.09E-06	4.17E-10
4	0.01	0.15106329	2.83E-05	0.07161814	3.34E-04
5	0.01	0.02390923	1.26E-10	1.92E-04	8.94E-07

Table 8: Munin1 errors using *Basic Approximate Factorisation* (BF).

Evidence	δ	G	MSE	Max Abs Error	KL error
1	0.001	2.18E-04	7.56E-16	7.79E-07	7.43E-09
2	0.001	0.16507203	1.12E-05	0.09472202	4.86E-04
3	0.001	1.95E-04	1.14E-14	1.70E-06	4.02E-10
4	0.001	0.16799176	2.63E-05	0.0674813	3.26E-04
5	0.001	0.0032453	1.83E-10	1.70E-04	8.19E-08
1	0.01	2.18E-04	7.56E-16	7.79E-07	7.43E-09
2	0.01	0.32096843	8.62E-05	0.16207984	0.00115003
3	0.01	2.71E-04	1.40E-14	1.70E-06	1.26E-09
4	0.01	0.17347653	3.50E-05	0.07161814	5.37E-04
5	0.01	0.02402091	2.12E-09	6.66E-04	9.28E-07

Table 9: Link errors using *Average First Factorisation* (VF).

Evidence	δ	G	MSE	Max Abs Error	KL error
1	0.001	0.39335135	7.22E-05	0.14109113	3.90E-04
2	0.001	0.6445906	1.58E-04	0.18260543	9.56E-04
3	0.001	0.78562278	2.49E-04	0.25	0.00137449
4	0.001	0.2838208	4.26E-05	0.09375	1.92E-04
5	0.001	0.41843547	9.35E-05	0.15579897	3.69E-04
1	0.01	0.39470625	7.25E-05	0.14109113	3.94E-04
2	0.01	0.68858348	1.78E-04	0.18260543	0.0010874
3	0.01	0.78562278	2.49E-04	0.25	0.00137449
4	0.01	0.2838208	4.26E-05	0.09375	1.92E-04
5	0.01	0.3627544	6.76E-05	0.15579897	2.92E-04

Table 10: Link errors using *Approximate First Factorisation* (AF).

Evidence	δ	G	MSE	Max Abs Error	KL error
1	0.001	0.6641237	2.10E-04	0.14247104	0.00111876
2	0.001	0.82480467	3.11E-04	0.20569782	0.00180448
3	0.001	0.52747113	1.10E-04	0.14967471	6.46E-04
4	0.001	0.46437581	1.17E-04	0.15625	5.05E-04
5	0.001	0.262509	3.32E-05	0.09071757	1.82E-04
1	0.01	0.66844586	2.12E-04	0.14247104	0.00113907
2	0.01	0.95288837	4.06E-04	0.20585263	0.00234069
3	0.01	0.52747113	1.10E-04	0.14967471	6.46E-04
4	0.01	0.46437581	1.17E-04	0.15625	5.05E-04
5	0.01	0.26840516	3.30E-05	0.09071757	1.94E-04

Table 11: Link errors using *Basic Approximate Factorisation* (BF).

Evidence	δ	G	MSE	Max Abs Error	KL error
1	0.001	0.70855783	2.15E-04	0.14247104	0.0012569
2	0.001	0.76308514	2.65E-04	0.22615267	0.00154018
3	0.001	0.49794044	1.06E-04	0.13927838	6.11E-04
4	0.001	0.46359131	1.17E-04	0.15625	5.03E-04
5	0.001	0.45217569	8.96E-05	0.125	5.54E-04
1	0.01	0.71593735	2.18E-04	0.14247104	0.00129806
2	0.01	0.77002952	2.70E-04	0.22630277	0.00157727
3	0.01	0.49794044	1.06E-04	0.13927838	6.11E-04
4	0.01	0.49794044	1.06E-04	0.13927838	6.11E-04
5	0.01	0.41467981	7.38E-05	0.125	4.76E-04

List of Figures

Figures

X	Y	Z	$\Phi(X,Y,Z)$
0	0	0	0.2
0	0	1	0.5
0	1	0	0.7
0	1	1	0.7
1	0	0	0.3
1	0	1	0.5
1	1	0	0.3
1	1	1	0.3

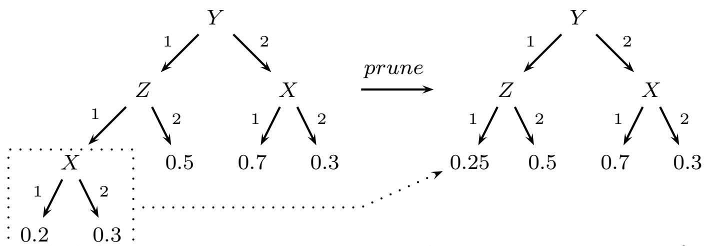


Figure 1: A potential ϕ, a probability tree representing it, and an approximation of it after pruning the branches beneath configuration $(Y = 1, Z = 1)$.

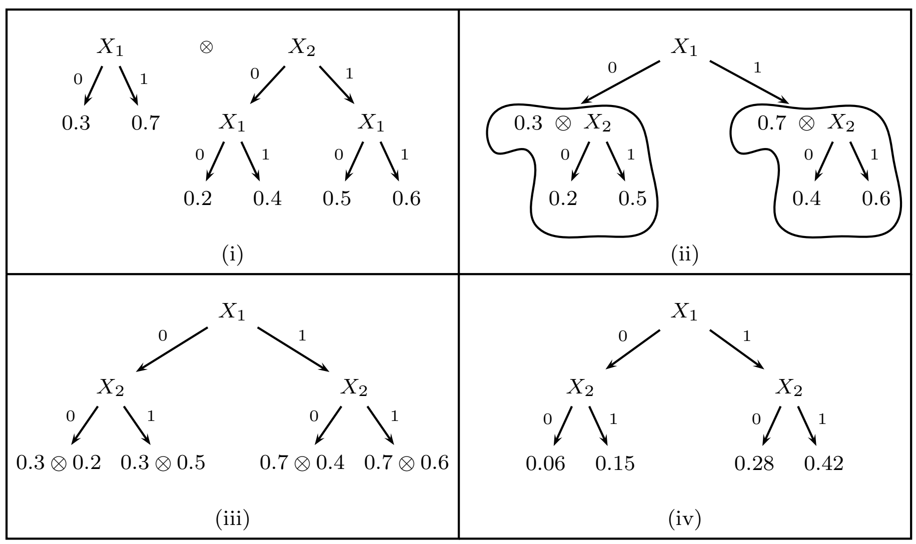


Figure 2: Combination of two trees.

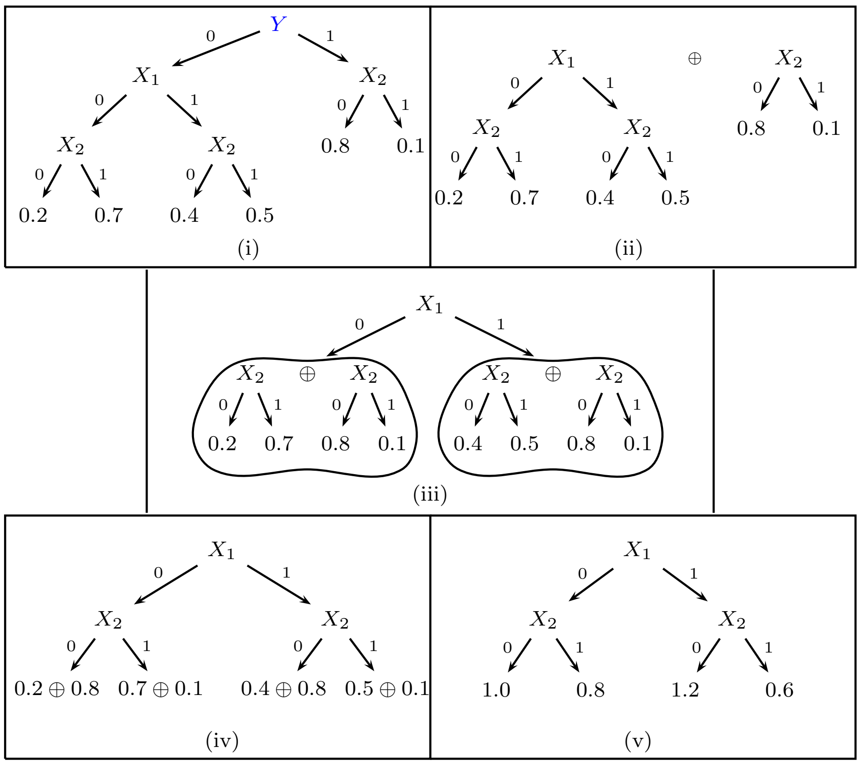


Figure 3: Variable Y is marginalised out by summing its two subtrees.

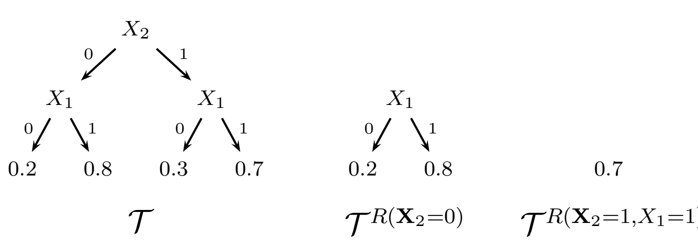


Figure 4: The restriction operation.

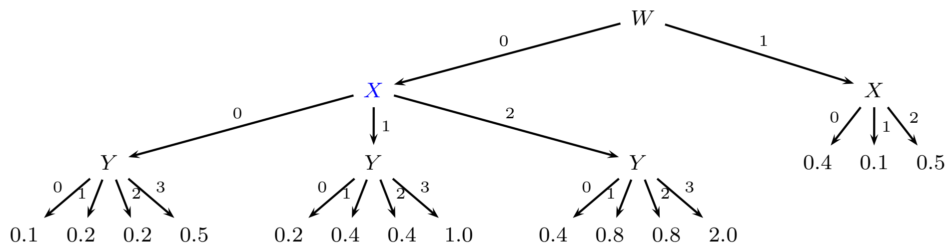


Figure 5: A probability tree $\mathcal{T}$ proportional below X for context $(W = 0)$.

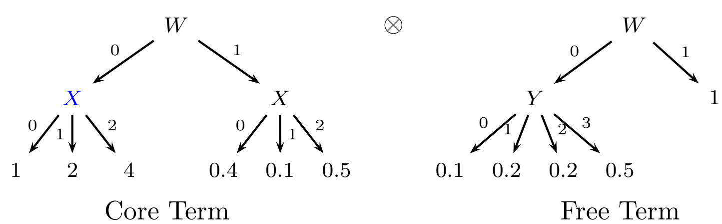


Figure 6: *Exact factorisation* of the tree in figure 5 with respect to variable X.

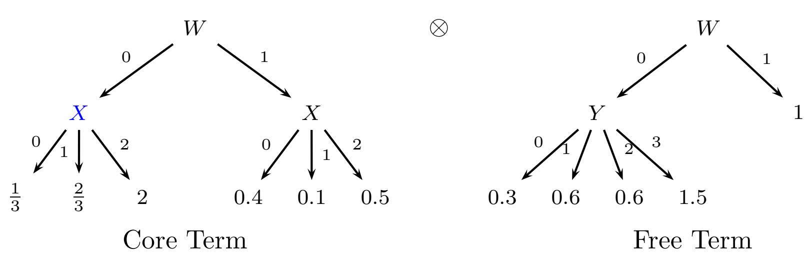


Figure 7: Factorisation with Average Free Term of the tree in Figure 5.

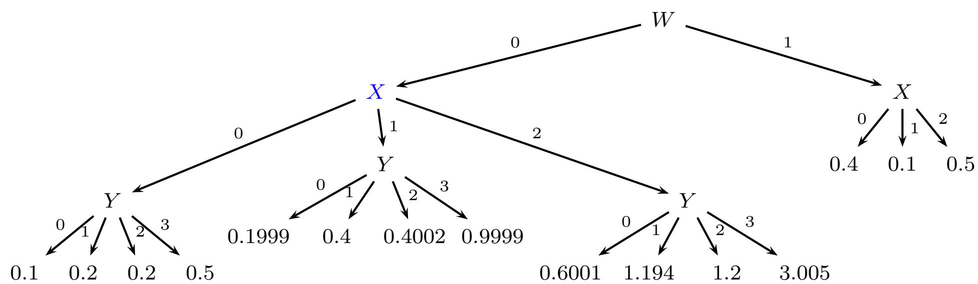


Figure 8: A probability tree δ-factorisable below X for context $W = 0$.

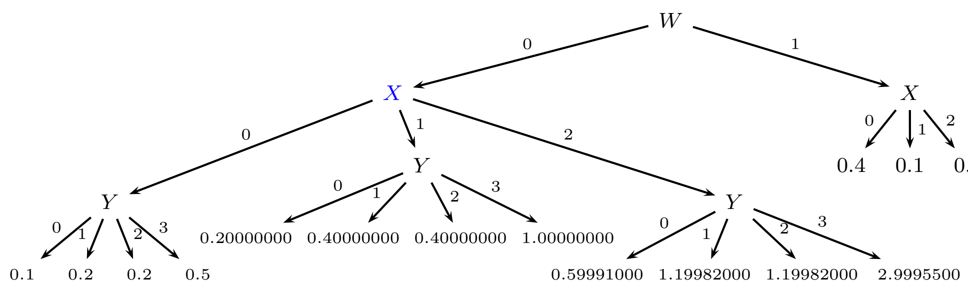


Figure 9: Approximation of the subtrees below X for context $W = 0$ by the *invariant potential* method (see [12]). The proportionality factors are 1, 2 and 5.99910000.

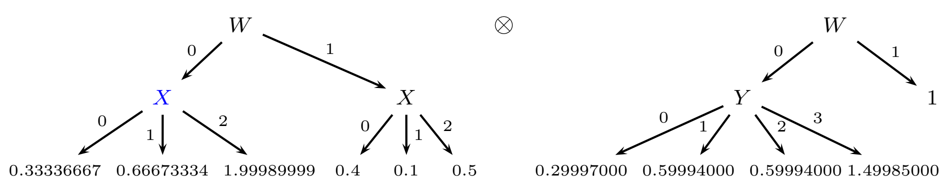


Figure 10: Approximate-First factorisation with average free term of the tree in Figure 8 according to the *invariant potential* method.

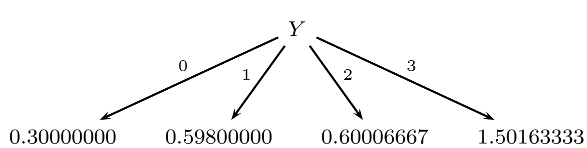


Figure 11: Average factor subtree obtained from the tree in Figure 8.

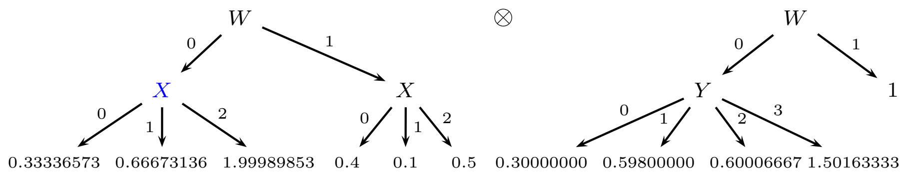


Figure 12: Average-First factorisation with average free term of the tree in Figure 8 using the method of minimum χ^2 divergence.

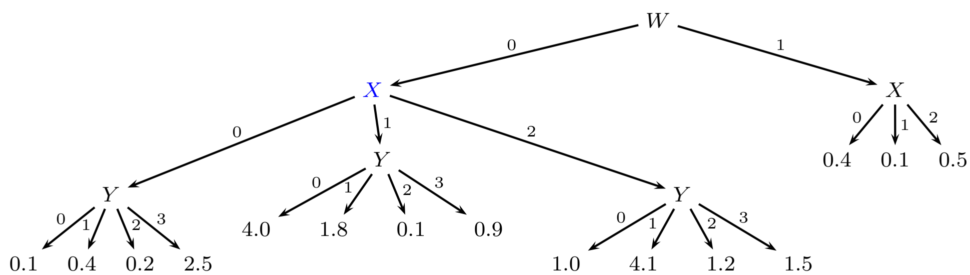


Figure 13: A tree far away from proportionality.

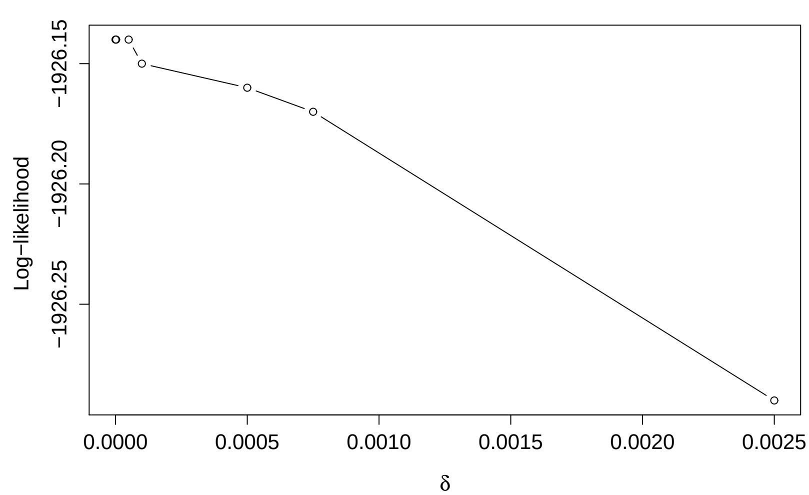


Figure 14: Log-likelihood vs. δ for the experiment with an initial tree that admits a close-to-exact factorisation, using the KL divergence as divergence measure and as method for computing the α coefficients.

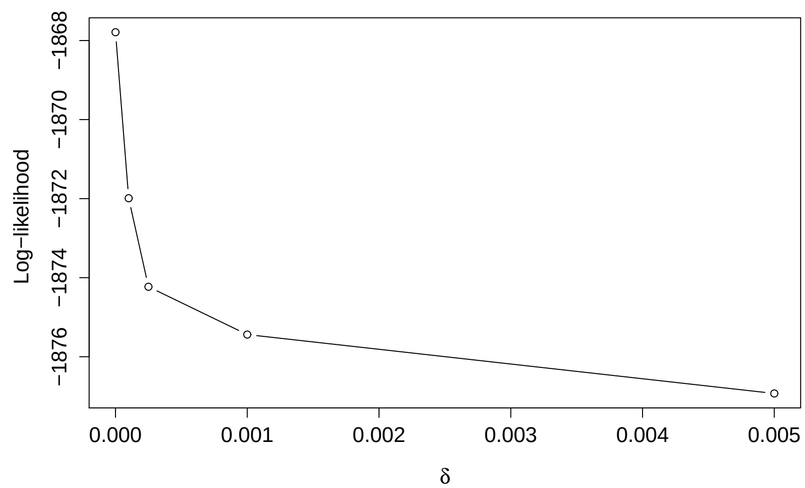


Figure 15: Log-likelihood vs. δ for the experiment with an initial tree that admits a close-to-exact factorisation, using the normalised χ^2 divergence as divergence measure and the weight preserving technique as method for computing the α coefficients.

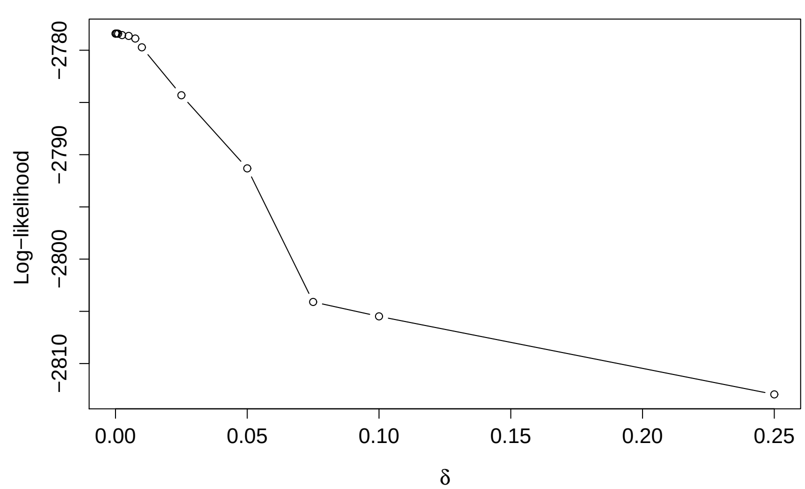


Figure 16: Log-likelihood vs. δ for the experiment with an initial tree generated at random, using the KL divergence as divergence measure and as method for computing the α coefficients.

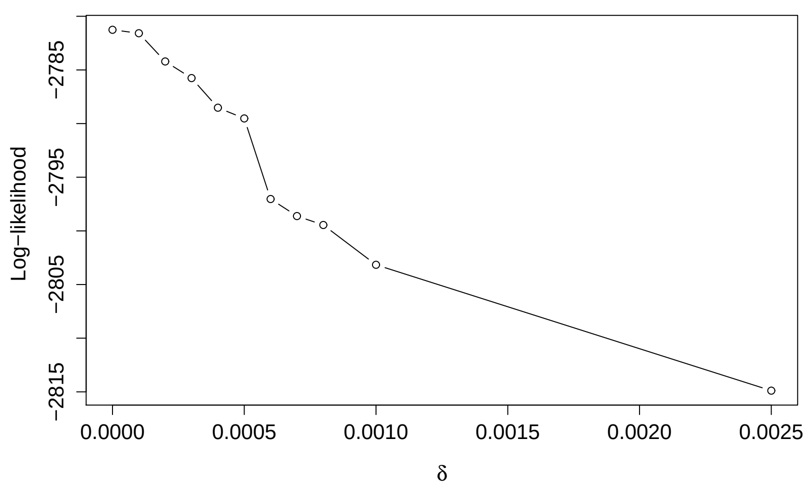


Figure 17: Log-likelihood vs. δ for the experiment with an initial tree generated at random, using the normalised χ^2 divergence as divergence measure and the weight preserving technique as method for computing the α coefficients.