Using Latent Binary Variables for Online Reconstruction of Large Scale Systems

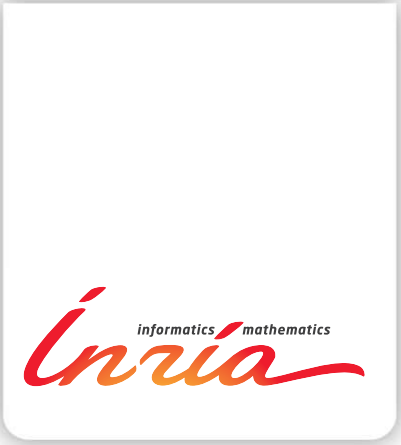
Victorin Martin, Jean-Marc Lasgouttes, Cyril Furtlehner

Using Latent Binary Variables for Online Reconstruction of Large Scale Systems

Victorin Martin, Jean-Marc Lasgouttes, Cyril Furtlehner

RESEARCH REPORT
N° 8435
décembre 2013
Project-Teams Imara and TAO

ISSN 0249-6399 ISRN INRIA/RR--8435--FR+ENG



Using Latent Binary Variables for Online Reconstruction of Large Scale Systems

Victorin Martin*, Jean-Marc Lasgouttes†, Cyril Furtlehner‡

Project-Teams Imara and TAO

Research Report n° 8435 — décembre 2013 — 34 pages

Abstract: We propose a probabilistic graphical model realizing a minimal encoding of real variables dependencies based on possibly incomplete observation and an empirical cumulative distribution function per variable. The target application is a large scale partially observed system, like e.g. a traffic network, where a small proportion of real valued variables are observed, and the other variables have to be predicted. Our design objective is therefore to have good scalability in a real-time setting. Instead of attempting to encode the dependencies of the system directly in the description space, we propose a way to encode them in a latent space of binary variables, reflecting a rough perception of the observable (congested/non-congested for a traffic road). The method relies in part on message passing algorithms, i.e. belief propagation, but the core of the work concerns the definition of meaningful latent variables associated to the variables of interest and their pairwise dependencies. Numerical experiments demonstrate the applicability of the method in practice.

Key-words: latent variables, Markov random field, belief propagation, inference, soft constraints

* Mines-Paristech, Paris, France, e-mail: victorin.martin@mines-paristech.fr
† Inria, Imara Project-Team, e-mail: jean-marc.lasgouttes@inria.fr
‡ Inria, TAO Project-Team, e-mail: cyril.furtlehner@inria.fr

RESEARCH CENTRE
PARIS – ROCQUENCOURT

Domaine de Voluceau, - Rocquencourt
B.P. 105 - 78153 Le Chesnay Cedex

Reconstruction en ligne dans des grands systèmes grâce à des variables binaires latentes

Résumé : On propose un modèle graphique probabiliste réalisant un encodage parcimonieux des dépendances d'un jeu de variables réelles à partir de corrélations possiblement incomplètes et de la fonction de répartition de chaque variable. L'application sous-jacente est un grand système partiellement observé, comme par exemple un réseau routier, où une petite proportion de variables réelles sont observées et les autres doivent être prédites. Il s'agit donc de trouver un algorithme permettant de travailler en temps réel sur des grands systèmes. Au lieu d'essayer d'encoder directement les dépendances du système, on propose une manière de le faire dans un espace de variables binaires latentes qui reflètent une vision sommaire des variables (par exemple embouteillé/fluide pour des routes). La méthode s'appuie en partie sur des algorithmes de passage de message comme « belief propagation », mais la difficulté est surtout de définir des variables binaires cohérentes avec celles de départ, ainsi que leurs corrélations. L'applicabilité de la méthode en pratique est démontrée par des expérimentation numériques.

Mots-clés : variables latentes, champ markovien aléatoire, propagation de croyances, inférence, contraintes bayésiennes

1 Introduction

Predicting behavior of large scale complex stochastic systems is a relevant question in many different situations where a (communication, energy, transportation, social, economic...) network evolves for instance with respect to some random demand and limited supply. This remains to a large extent an open and considerable problem, especially for partially observed systems with strong correlations (see e.g. Boyen [5]), though efficient methods, like Kalman and, by extension, particle filtering, see e.g. Doucet et al. [11], exist, but with limited scalability.

In the example which motivates this work, road traffic reconstruction from floating car data, the system is partially observed and the goal is to predict the complete state of the traffic network, which is represented as a high-dimensional real valued vector of travel times or alternatively speeds or densities. State of the art methods in this field exploit both temporal and spatial correlations with multivariate regression (Min and Wynter [24]) with rather restrictive linear hypothesis on the interactions. Here, we explore a different route for the encoding of spatial and potentially temporal dependencies, which we believe can simplify both the model calibration and the data reconstruction tasks for large scale systems.

The classical way to obtain data on a road traffic network is to install fixed sensors, such as magnetic loops. However, this is adapted to highways and arterial roads, but not to a whole urban network which typically scales up to 10^5 segments. As part of the Field Operational Test PUMAS [28] in Rouen (Normandy), we explored the possibility to acquire data with equipped vehicles that send geolocalized information, and to process it directly with a fast prediction scheme (Furtlehner et al. [12, 13]). While offline processing of historical data can be allowed to be time consuming, travel times predictions must instead be available in "real-time", which means in practice a few minutes. This "real-time" constraint implies some design choices: firstly, the predictions need to be computed online, even on large networks, which can be achieved using the message-passing algorithm Belief Propagation of Pearl [27]; secondly, our model shall be suitable for the use of this inference algorithm.

Stated in a more generic form, the problem at hand is to predict, from sparse data originating from non stationary locations, the value of the variables on the rest of the network. The set of nodes to predict is potentially varying from time to time because sensors are moving, like probe vehicles in the traffic context. Since only very sparse joint observations are available, purely data driven methods such as k nearest neighbors cannot be used and one has to resort to building some model. We study in this article the possibility of a probabilistic graphical model that avoids modeling the underlying complexity of the physical phenomena. Building a model of dependency between real-valued variables can be very costly both in terms of

statistics, calibration and prediction if one tries to account for the empirical joint probability distribution for each pair of variables. A possible way to proceed, compatible with the use of BP, is to build a multivariate Gaussian Copula, since then Gaussian belief propagation can be run efficiently on such models. However, in the traffic example, the variables are endowed with a binary perception (congested/non-congested), which make it likely that the joint distribution will be multi modal in general and therefore not very well suited for a Gaussian model, which admits only one reference state associated to one single belief propagation fixed point. What we propose instead is to abstract the binary perception as a latent state descriptor and to exploit it by encoding the dependencies between the real state variables at the level of the latent variables. This way we end up with a minimal parametric method, where both the prediction and the calibration are easy to perform and which is well adapted to multimodal distributions; each mode can be under certain conditions associated to a belief propagation fixed point [14]. Another route could be to build a general continuous copula model, compatible with the use of the expectation propagation (EP) algorithm of Minka [25]. The EP algorithm is indeed also very simple and efficient to use, but the model selection stage might be too complex for large scale applications. It requires to choose manually the exact exponential family for modeling the marginal and pairwise distributions, since no automatic procedures has emerged up to now. Also, compared to traditional methods like particle filtering, we expect a better scalability.

We formalize the model as follows: the state of the system is represented by a vector $\mathbf{X} = (X_i)_{i\in\mathbb{V}}$ of N real valued variables, attached to nodes $i \in \mathbb{V}$ and taking their respective values in the sets $\mathcal{X}_i \subset \mathbb{R}$. We assume that we never observe the full vector $\mathbf{X}$, but that only pairwise observations are available. For a given set $\mathbb{E} \subset \mathbb{V}^2$ of pair of variables, each pair (X_i, X_j) such as $(i, j) \in \mathbb{E}$ is observed N_{ij} times, all observations being independent. These pair samples are stored in the vector $\mathbf{x}$, which contains all the vectors $\mathbf{x}^k$:

$$(X_i, X_j) = (x_i^k, x_j^k) \text{ for } k \in \{1, \dots, N_{ij}\}. \tag{1}$$

The model goes as follows: to each variable X_i is attached a binary latent variable σ_i and the variables X_i are assumed to be independent, conditionally to the latent state $\boldsymbol{\sigma}$. This is a strong assumption, generally false, but we shall see in this paper that it can provide an efficient model for the prediction task. We wish to stress that it will be necessary to *construct* these latent variables and multiple choices could be meaningful. Somehow this is a choice of feature functions from $\mathcal{X}_i$ to $\{0, 1\}$. The problem at stake is not to infer the states of a hidden Markov model from noisy observations: the only variables of interest are the X_i's. To be able to infer the behavior of these variables, given a partial observation of the system, we use a pairwise Markov Random

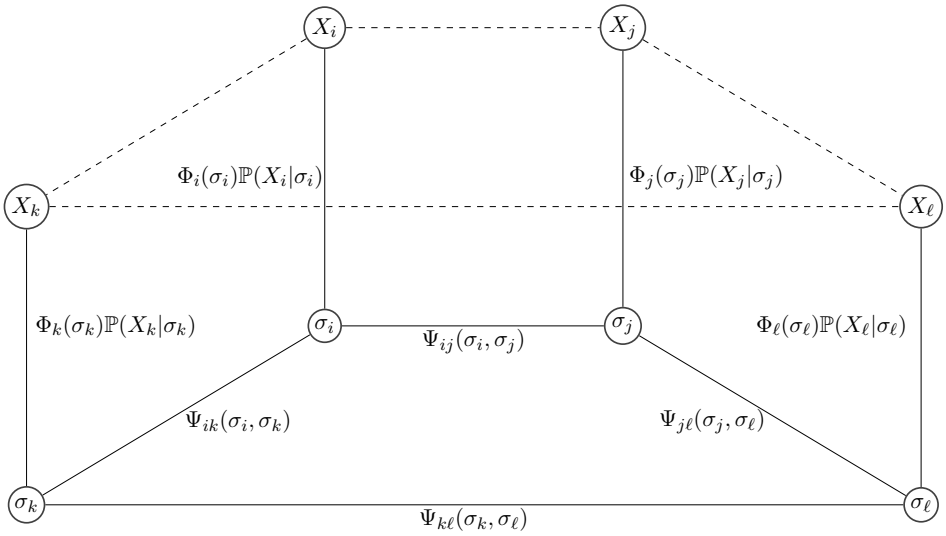


Figure 1: Markov random field $(\mathbf{X}, \boldsymbol{\sigma})$ for $\mathbb{V} = \{i, j, k, \ell\}$. The true model of the vector $\mathbf{X}$ (dashed lines) is approximated through the latent binary variables $\boldsymbol{\sigma}$ (plain lines).

Field (MRF) for the binary variables σ_i, i.e. an Ising model in statistical physics parlance (Baxter [1]). The joint measure for the variables $\mathbf{X}$ and $\boldsymbol{\sigma}$ factorizes as (Figure 1):

$$\mathbb{P}(\mathbf{X} \leq \mathbf{x}, \boldsymbol{\sigma} = \mathbf{s}) = \mathbb{P}(\boldsymbol{\sigma} = \mathbf{s}) \prod_{i \in \mathbb{V}} \mathbb{P}(X_i \leq x_i | \sigma_i = s_i),$$

$$\mathbb{P}(\boldsymbol{\sigma} = \mathbf{s}) = \frac{1}{Z} \prod_{(i,j) \in \mathbb{E}} \psi_{ij}(s_i, s_j) \prod_{i \in \mathbb{V}} \phi_i(s_i),$$

with Z a constant ensuring that $\mathbb{P}$ sums up to 1. Of course it will not be possible to model in a precise way the joint distribution of any random vector $\mathbf{X}$ through our latent Ising model. The task assign to the model is actually less ambitious: we wish to make predictions about this random vector $\mathbf{X}$. The problem we are trying to solve is simply a regression on the variables X_i, which is very different in nature from modeling the distribution $\mathcal{P}$. Note that with observations (1) Jaynes' maximum entropy criterion leads us to a pairwise interaction model which is compatible with our choice.

Based on these assumptions, we try to answer three main questions:

(i) How to define the latent variable σ_i and how to relate it to its real valued variable X_i?

(ii) How to construct the dependencies between latent variables σ_i in an efficient way in terms of prediction performance?

(iii) How can partial observations be inserted into the model to perform the predictions of the unobserved variables?

These three questions are of course highly interdependent. Once the model has been built, exact procedures to infer the behavior of the X_i's generally face an exponential complexity, and one has to resort to an approximate procedure. We rely here on Pearl's belief propagation (BP) algorithm [27] – widely used in the Artificial Intelligence and Machine Learning communities [20, 37] – as a basic decoding tool, which in turn will influence the MRF structure. While this algorithm is well defined for real-valued variables in the case of a Gaussian vector $\mathbf{X}$ (see Bickson [3]), the more general case requires other procedures, like the nonparametric BP algorithm proposed by Sudderth et al. [31], which involves much more computation than the classical BP algorithm does. We propose here a new BP-based method to tackle this same problem while keeping computations lightweight. The BP algorithm will be precisely defined in Section 4.

The paper is organized as follows: Section 2 is devoted to answering question (i), by finding a relevant mapping of an observation $X = x$ to the parameter of a Bernoulli variable σ. As we shall, see this is equivalent to the definition of a feature function. Section 3 focuses on question (ii) concerning the optimal encoding in the latent space of the dependency between the X_i's. In Section 4, we construct a variant of BP named "mirror BP" that imposes belief values of σ_i when X_i is observed; this addresses question (iii). Some experimental results of these methods are presented in Section 5.

2 Latent variables definition

Let X be a real-valued random variable with cumulative distribution function (cdf) $F(x) \stackrel{\text{def}}{=} \mathbb{P}(X \leq x)$. We focus in this section on a way to relate an observation $X = x$ to a latent binary variable σ. In the following we will call "σ-parameter" the value $\mathbb{P}(\sigma = 1)$.

2.1 A stochastically ordered mixture

A simple way to relate an observation $X = x$ to the latent variable σ is through a mapping Λ such that $\Lambda(x)$ is the σ-parameter. The mapping Λ will be referred to as the encoding function and can depend on the cdf F. σ being a latent variable, it will not be directly observed, but conditionally to an observation $X = x$, we define its distribution as:

$$\mathbb{P}(\sigma = 1 | X = x) \stackrel{\text{def}}{=} \Lambda(x). \tag{2}$$

For simplicity, we assume that Λ is continuous on right, limited on left (*corlol*) and increasing. Note that the condition "Λ is increasing" is equivalent to have a monotonic mapping Λ since choosing the mapping $1-\Lambda$ simply

inverts the states 0 and 1 of the variable σ. Moreover Λ shall increases from 0 to 1, without requiring that $\Lambda(\mathcal{X}) = [0,1]$, since Λ can be discontinuous. This constraint is expressed as the following:

$$\int_{\mathcal{X}} d\Lambda(X) = 1 \quad \text{and} \quad \inf_{x \in \mathcal{X}} \Lambda(x) = 0. \tag{3}$$

Let us emphasize again that σ is not just an unobserved latent random variable which estimation is required. It is a feature that we define in order to tackle the inference on $\mathbf{X}$. This encoding is part of the following global scheme

$$\begin{array}{ccc} X_i = x_i \in \mathcal{X}_i & \xrightarrow{\Lambda_i} & \mathbb{P}(\sigma_i = 1 | X_i = x_i) \in \Lambda_i(\mathcal{X}_i) \\ & & \Big\downarrow \text{mBP} \\ X_j = x_j \in \mathcal{X}_j & \xleftarrow{\Gamma_j} & b(\sigma_j = 1) \in [0,1] \end{array} \tag{4}$$

which is as follows:

- observations of variables X_i are encoded through the distribution of a latent binary random variable σ_i using the encoding function Λ_i,
- a marginalisation procedure is then performed on these latent variables $\boldsymbol{\sigma}$, in a way that will be described later,
- and finally the distributions of variables σ_j allows in turn to make predictions about the other real variables X_j.

This scheme requires that we associate to Λ an "inverse" mapping $\Gamma : [0,1] \mapsto \mathcal{X}$. Since Λ can be non invertible, the decoding function Γ cannot always be the inverse mapping Λ^{-1}. We will return to the choice of the function Γ in Section 2.3.

To understand the interaction between σ and X, let us define the conditional cdf's:

$$F^0(x) \stackrel{\text{def}}{=} \mathbb{P}(X \leq x | \sigma = 0),$$
$$F^1(x) \stackrel{\text{def}}{=} \mathbb{P}(X \leq x | \sigma = 1).$$

Bayes' theorem allows us to write

$$\mathbb{P}(\sigma = 1 | X = x) = \mathbb{P}(\sigma = 1) \frac{dF^1}{dF}(x),$$

and thus

$$dF^1(x) = \frac{\Lambda(x)}{\mathbb{P}(\sigma = 1)} dF(x). \tag{5}$$

Summing over the values of σ imposes

$$F(x) = \mathbb{P}(\sigma = 1) F^1(x) + \mathbb{P}(\sigma = 0) F^0(x), \tag{6}$$

and the other conditional cdf follows

$$dF^0(x) = \frac{1-\Lambda(x)}{\mathbb{P}(\sigma=0)} dF(x). \tag{7}$$

The choice of this class of *corlol* increasing functions has a simple stochastic interpretation given in the following proposition.

Proposition 1. *The choice of an increasing encoding function Λ yields a separation of the random variable X into a mixture of two stochastically ordered variables X^0 and X^1 with distributions dF^0 and dF^1. Indeed, we then have*

$$X \sim \mathbb{1}_{\{\sigma=0\}} X^0 + \mathbb{1}_{\{\sigma=1\}} X^1,$$

where $\sim$ is the equality in term of probability distribution. The stochastic ordering is the following

$$X^0 \preceq X \preceq X^1.$$

Proof. It is sufficient (and necessary) to prove that

$$\forall x \in \mathcal{X}, \quad F^1(x) \leq F(x) \leq F^0(x).$$

Consider first the left inequality ($F^1 \leq F$); If $x \in \mathcal{X}$ is such that $\Lambda(x) \leq \mathbb{P}(\sigma=1)$, then we have:

$$\begin{aligned} F^1(x) &= \int_{-\infty}^{x} dF^1(y) = \int_{-\infty}^{x} \frac{\Lambda(y)}{\mathbb{P}(\sigma=1)} dF(y), \\ &\leq \int_{-\infty}^{x} dF(y) = F(x), \end{aligned}$$

because, Λ being increasing, $\Lambda(y) \leq \mathbb{P}(\sigma=1)$ for all $y \in]-\infty, x]$. Conversely, when $\Lambda(x) \geq \mathbb{P}(\sigma=1)$,

$$\begin{aligned} F^1(x) &= 1 - \int_{x}^{+\infty} dF^1(y) = 1 - \int_{x}^{+\infty} \frac{\Lambda(y)}{\mathbb{P}(\sigma=1)} dF(x), \\ &\leq 1 - \int_{x}^{+\infty} dF(y) = F(x), \end{aligned}$$

using again the fact that Λ is increasing. The other inequality ($F \leq F^0$) is obtained using (6). □

Remark 1. *Since the encoding function Λ is increasing from 0 to 1, it can be considered as the cdf of some random variable Y,*

$$\mathbb{P}(Y \leq x) \stackrel{def}{=} \Lambda(x),$$

which allows to reinterpret the previous quantities in terms of Y*:*

$$\begin{aligned}\mathbb{P}(\sigma = 1) &= \int_{\mathcal{X}} \mathbb{P}(\sigma = 1|X = x)dF(x) \\ &= \int_{\mathcal{X}} \Lambda(x)dF(x) = \int_{\mathcal{X}} \mathbb{P}(Y \leq x)dF(x) \\ &= \mathbb{P}(Y \leq X),\end{aligned}$$

supposing that Y *and* X *are independent. The variable* σ *can therefore be defined as*

$$\sigma \stackrel{def}{=} \mathbb{1}_{\{Y \leq X\}},$$

which means that the variable Y *acts as a random threshold separating* X*-values that correspond to latent states* 0 *and* 1*. The stochastic ordering between* $(X|\sigma = 0)$ *and* $(X|\sigma = 1)$ *then appears quite naturally. Note that this interpretation leads to a natural extension to a larger discrete feature space for* σ *simply using multiple thresholds. When* $\Lambda = F$*, the conditional cdf's of* X *are:*

$$\begin{aligned}F^1(x) &= (F(x))^2 = \mathbb{P}(\max(X_1, X_2) \leq x), \\ F^0(x) &= F(x)(2 - F(x)) = \mathbb{P}(\min(X_1, X_2) \leq x),\end{aligned}$$

with X_1 *and* X_2 *two independent copies of* X*.*

2.2 Choosing a good encoding function Λ

Now that the nature of the mapping between X and σ has been described, it remains to find an "optimal" encoding function. It turns out to be difficult to find a single good criterion for this task. In this section, we therefore propose two different approaches, based respectively on the mutual information and on the entropy.

Mutual information. The idea here is to choose Λ (or equivalently σ), such that the mutual information $I(X, \sigma)$ between variables X and σ is maximized. In other words, a given information about one variable should lead to as much knowledge as possible on the other one.

Proposition 2. *Let* $q_X^{0.5}$ *be the median of* X*. The encoding function* Λ_{MI} *which maximizes the mutual information* $I(X, \sigma)$ *between variables* X *and* σ *is the step function*

$$\Lambda_{\text{MI}}(x) \stackrel{def}{=} \mathbb{1}_{\{x \geq q_X^{0.5}\}}.$$

Before turning to the proof of this proposition, let us remark that this definition of the binary variable σ is a natural one, σ being deterministic as a function of X. However, as we shall see in Section 5, it is usually suboptimal for the reconstruction task.

Proof. The function to maximize is

$$\begin{aligned} I(X,\sigma) &= \sum_s \int_{\mathcal{X}} \mathbb{P}(\sigma = s) \log\left(\frac{dF^s(x)}{dF(x)}\right) dF^s(x) \\ &= H(\mathbb{P}(\sigma = 1)) - \int_{\mathcal{X}} H(\Lambda(x)) dF(x), \end{aligned}$$

where $H(p) \stackrel{\text{def}}{=} -p\log p - (1-p)\log(1-p)$ is the binary entropy function. Among all random variables σ with entropy $H(\mathbb{P}(\sigma = 1))$, the ones which maximize $I(X,\sigma)$ are deterministic functions of X, or equivalently the ones for which Λ is an indicator function. Since we limit ourselves to the *corlol* class, we get $\Lambda = \mathbb{1}_{[a,+\infty[}$ for some $a \in \mathcal{X}$. It remains to maximize the entropy of the variable σ, which leads to $P(\sigma = 1) = 1/2$ and $a = q_X^{0.5}$. □

Max-entropy principle. Another possibility is, in order to maximize the information contained in the latent variable σ, to maximize the entropy of $U = \Lambda(X)$. This variable U is indeed the data that will be used to build the Ising model over the latent variables (see Section 3). We assume here that the variable X admits a probability density function (pdf). We add to the few constraints detailed in the previous section that Λ is a bijection between $\mathcal{X}$ and $[0,1]$. When dealing with continuous random variables, the entropy only makes sense relatively to some measure (see Jaynes [19, pp. 374-375]). Following Jaynes' [18] arguments, since U is the parameter of a Bernoulli variable with both outcomes possible, having no other prior knowledge leads us to the uniform measure as reference.

Proposition 3. *Let X be a random variable which admits a pdf. The (increasing) invertible function which maximizes the entropy of $U = \Lambda(X)$, taken relatively to the uniform measure, is the cumulative distribution function F of the variable X.*

Proof. The variable U with maximal entropy has a uniform pdf $h_\Lambda(u) = \mathbb{1}_{[0,1]}(u)$. The encoding function Λ such as $\Lambda(X)$ is a uniform variable on $[0,1]$ is the cdf of X, which concludes the proof. □

Let us quickly sum up the choices we proposed for the encoding function:

- Λ_{MI} which is a deterministic encoding: σ indicates the position of X w.r.t. its median;
- F the cdf of X, which corresponds to the less discriminating choice about the encoded data distribution.

We will see that these two encoding functions have very distinct properties: Λ_{MI} is much more conservative than F but is rather adapted to model precisely the joint distributions. Let us remark that in the first case Λ is the feature function while in the second case the feature function is a random variable.

2.3 Decoding function Γ

Before turning to the definition of the decoding function Γ, let us focus first on the following simple question:

> *What is the best predictor of a real-valued random variable X, knowing only its distribution?*

The answer will obviously depend on the loss function considered and this will in turn influence the choice of the decoding function Γ, which purpose is to predict the random variable X. Assuming a L^r norm as loss function, the optimal predictor $\hat{\theta}_r(X)$ is then defined as

$$\hat{\theta}_r(X) \stackrel{\text{def}}{=} \operatorname*{argmin}_{c\in\mathbb{R}} \mathbb{E}_X[|X-c|^r].$$

In the case $r=1$, the optimal predictor $\hat{\theta}_1(X)$ is simply the median of X; $r=2$ corresponds to $\hat{\theta}_2(X)=\mathbb{E}[X]$, the mean value of X. In the following we call "contextless prediction" the X-prediction performed without other information than the distribution of X.

When focusing on the definition of the "inverse" mapping Γ, two natural definitions arise. When Λ is a bijection, the simplest predictor of X, given $b=\mathbb{P}(\sigma=1)$, is $\Lambda^{-1}(b)$. Actually, it is the unique X-value such that $(\sigma|X=x)$ is distributed as $\mathbb{P}(\sigma=1|X=x)=b$ by definition (2) of Λ. We will denote this first choice for the decoding function

$$\Gamma^{\mathcal{L}} \stackrel{\text{def}}{=} \Lambda^{-1}.$$

$\Gamma^{\mathcal{L}}$ corresponds, in some sense, to a predictor based on maximum likelihood (ML). Indeed, suppose that the knowledge of $b=\mathbb{P}(\sigma=1)$ is replaced with a sample of M independent copies s^k of a binary variable distributed as $\mathbb{P}(\sigma|X=x)$. The ML estimate of x is then $\Lambda^{-1}(\sum_k s^k/M)$. So the choice Λ^{-1} as decoding function corresponds to the ML estimate from a sample with an empirical rate of success equal to b.

In the more general case of an increasing *corlol* encoding function, with a Bayesian point of view, the knowledge of the σ-parameter allows to update the distribution of X. Applying Jeffrey's update rule (see Chan and Darwiche [6]) yields the updated cdf $F^{\mathcal{B}}$

$$F^{\mathcal{B}}(x) = bF^1(x) + (1-b)F^0(x). \tag{8}$$

Let $X^{\mathcal{B}}$ be a random variable which distribution is $F^{\mathcal{B}}$. The predictor $\hat{\theta}(X^{\mathcal{B}})$ previously defined can be used irrespective of whether Λ is invertible or not. To refer to this second choice we will use the notation

$$\Gamma^{\mathcal{B}} \stackrel{\text{def}}{=} \hat{\theta}(X^{\mathcal{B}}).$$

Note that, while it may be costly in general to compute the wanted statistic of $X^{\mathcal{B}}$, some choices of Λ lead to explicit formulas.

Mutual information. We consider here the case of the step function Λ_{MI} as encoding function. This function is of course not invertible and only the Bayesian decoding function $\Gamma^{\mathcal{B}}$ can be used. Using (5)–(8), the cdf of $X^{\mathcal{B}}$ is

$$F^{\mathcal{B}}(x) = \begin{cases} 2(1-b)F(x), & \text{if } x \le q_X^{0.5}, \\ F^{\mathcal{B}}(q_X^{0.5}) + 2b\left(F(x) - F(q_X^{0.5})\right), & \text{if } x > q_X^{0.5}. \end{cases} \tag{9}$$

In order to compute $\hat{\theta}_1(X^{\mathcal{B}})$, we need to solve the equation $F^{\mathcal{B}}(x) = 1/2$, leading to the decoding function

$$\Gamma^{\mathcal{B}}(b) = \begin{cases} F^{-1}\left(\frac{1}{4(1-b)}\right), & \text{if } b \le \frac{1}{2}, \\ F^{-1}\left(\frac{4b-1}{4b}\right), & \text{if } b > \frac{1}{2}. \end{cases}$$

When F is not invertible, F^{-1} should be understood as the pseudo-inverse of F, commonly used to define quantiles:

$$F^{-1}(b) \stackrel{\text{def}}{=} \inf_x \{x \mid F(x) \ge b\}.$$

If we choose the predictor $\hat{\theta}_2$ based on a L^2 loss function, using the linearity of the expectation we get

$$\Gamma^{\mathcal{B}}(b) = \mathbb{E}\left[X^{\mathcal{B}}\right] = b\,\mathbb{E}\left[X \mid \sigma = 1\right] + (1-b)\mathbb{E}\left[X \mid \sigma = 0\right].$$

Max entropy principle. When one uses $\Gamma^{\mathcal{L}} = F^{-1}$ as decoding function, the contextless prediction, i.e. without any observation, is simply $F^{-1}(\mathbb{P}(\sigma = 1))$. Moreover, we know that $\mathbb{P}(\sigma = 1) = \mathbb{E}[F(X)] = 1/2$ – provided that X admits a pdf – so the ground prediction is the median of X. The choice $\Lambda = F$ and $\Gamma = F^{-1}$ is therefore optimal w.r.t. a L^1 loss function for the prediction error.

The other choice for the decoding function is to use $\Gamma^{\mathcal{B}}$ and to compute, for example, the predictor $\hat{\theta}_1(X^{\mathcal{B}})$. Using (5) – (8), we get the cdf of $X^{\mathcal{B}}$

$$F^{\mathcal{B}}(x) = \big((2b-1)F(x) - 2(b-1)\big)F(x),$$

and the sought function is solution of the following quadratic equation

$$\left((2b-1)F(x) - 2(b-1)\right)F(x) = \frac{1}{2},$$

with only one reachable root. Thus the Bayesian decoding function is

$$\Gamma^{\mathcal{B}}(b) = F^{-1}\left(\frac{2(b-1) + \sqrt{(2b-1)^2 + 1}}{4b-2}\right). \tag{10}$$

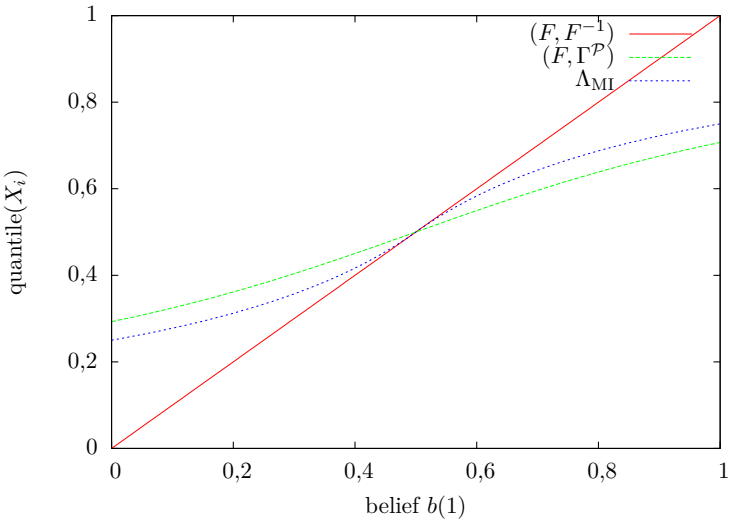


Figure 2: Prediction, expressed in quantiles, on the variable X_i for a given belief $b_i(1)$. The left and right values are $(1/4, 3/4)$ for Λ_{MI} and $(1-\sqrt{2}/2, \sqrt{2}/2)$ for $(F, \Gamma^{\mathcal{B}})$.

Let us remark that the Bayesian decoding function $\Gamma^{\mathcal{B}}$ is always more conservative than $\Gamma^{\mathcal{L}}$. Using the inverse F^{-1} allows us to make predictions spanning the whole set $\mathcal{X}$ of possible outcomes, which is not the case with the Bayesian decoding function. Figure 2 illustrates this.

Assume that two random variables X_1 and X_2 are equal with probability 1. Even if we build a latent model such that $\mathbb{P}(\sigma_1 = \sigma_2) = 1$, using $\Gamma^{\mathcal{B}}$ as decoding function will never predict $X_1 = X_2$. The decoding function $\Gamma^{\mathcal{B}}$ is in fact trying to approximate the joint distribution of (X_1, X_2) and this approximation can only be very rough when variables are strongly dependent (see Proposition 4). However, the choice (F, F^{-1}) is equivalent to performing a X-quantiles regression. We will see in Section 5 that this last choice is better when variables are strongly dependent.

In one wishes to choose the decoding function based on the ML estimate $\Gamma^{\mathcal{B}} = \Lambda^{-1}$, it is of interest to generalize the max-entropy criterion in order to get an encoding function Λ with an optimal contextless prediction w.r.t. a specific loss function. It is in fact quite simple to solve this problem and to obtain the sought encoding function which is based on the cdf (Martin [22, chapter 5]). We will use here only the cdf function because we are interested in L^1 error measure. Compared to a loss function based on the L^2 norm, it gives less weight to extreme values.

3 Building pairwise dependencies

It was shown in Section 2 how to relate the variable X_i to its latent state σ_i, by means of an encoding function Λ_i. The next question to address is how to encode the dependencies at the latent state level and, more generally, how to estimate the parameters of the underlying Ising model on $\boldsymbol{\sigma}$. Given two real-valued variables X_i and X_j, with respective cdf F_i and F_j, and two binary variables σ_i and σ_j, we want to construct a pairwise model as described in Figure 1. The probability distribution of the vector $(X_i, X_j, \sigma_i, \sigma_j)$ for this model is

$$\mathbb{P}(X_i \leq x_i, X_j \leq x_j, \sigma_i = s_i, \sigma_j = s_j) = p_{ij}(s_i, s_j) F_i^{s_i}(x_i) F_j^{s_j}(x_j). \tag{11}$$

Since σ_i and σ_j are binary variables, $p_{ij}(s_i, s_j)$ can be expressed with 3 independent parameters,

$$\begin{aligned} p_{ij}(s_i, s_j) = p_{ij}^{11} s_i s_j + \left(p_j^1 - p_{ij}^{11}\right) \bar{s}_i s_j + \left(p_i^1 - p_{ij}^{11}\right) s_i \bar{s}_j \\ + \left(1 - p_i^1 - p_j^1 + p_{ij}^{11}\right) \bar{s}_i \bar{s}_j, \end{aligned}$$

using the notation $\bar{s} \stackrel{\text{def}}{=} 1 - s$ and with

$$\begin{aligned} p_i^1 &\stackrel{\text{def}}{=} \mathbb{P}(\sigma_i = 1) = \mathbb{E}(\sigma_i), \\ p_{ij}^{11} &\stackrel{\text{def}}{=} \mathbb{P}(\sigma_i = 1, \sigma_j = 1) = \mathbb{E}(\sigma_i \sigma_j). \end{aligned}$$

The probability distribution is valid as soon as $(p_i^1, p_j^1) \in [0,1]^2$ and

$$p_{ij}^{11} \in \mathbb{D}(p_i^1, p_j^1) \stackrel{\text{def}}{=} \left[\max(0, p_i^1 + p_j^1 - 1), \min(p_i^1, p_j^1)\right].$$

Until now, we have been able to make optimal choices in some sense, but obviously the number of parameters is not enough to encode exactly any structure of dependency. This is shown in the following proposition

Proposition 4. *When the mutual information $I_{\mathcal{P}}(X_i, X_j)$ between the real variables is strictly greater than* $\log(2)$*, our model is not able to perfectly encode the joint distribution of X_i and X_j for any choice of encoding function.*

This result is compatible with intuition: whatever the definition of the binary variables, it will not be possible to share more than one bit of information between two of them. However, we shall see in Section 5 that it is still possible to obtain quasi-optimal performances for the prediction task even when the mutual information is strictly greater than $\log(2)$.

Proof. We will prove that the Kullback-Leibler divergence between the empirical joint distribution $\mathcal{P}$ of (X_i, X_j) and the joint distribution $\mathbb{P}$ within

our model is strictly positive as soon as $I_{\mathcal{P}}(X_i, X_j) > \log(2)$.

$$\begin{aligned} D_{\mathrm{KL}}(\mathcal{P}||\mathbb{P}) &= \int \mathcal{P}(x_i, x_j) \log \frac{\mathcal{P}(x_i, x_j)}{\mathbb{P}(x_i, x_j)} dx_i dx_j \\ &= I_{\mathcal{P}}(X_i, X_j) + \int \mathcal{P}(x_i, x_j) \log \frac{\mathcal{P}(x_i)\mathcal{P}(x_j)}{\mathbb{P}(x_i, x_j)} dx_i dx_j, \\ &\stackrel{\text{def}}{=} I_{\mathcal{P}}(X_i, X_j) - \mathbb{I}(X_i, X_j). \end{aligned}$$

Using the fact that $\mathbb{P}(x_i) = \mathcal{P}(x_i)$ and expanding w.r.t. σ_i and σ_j, one gets

$$\begin{aligned} \mathbb{I}(X_i, X_j) &= \int \mathcal{P}(x_i, x_j) \log \left(\sum_{\sigma_i, \sigma_j} \frac{\mathbb{P}(\sigma_i, \sigma_j)}{\mathbb{P}(\sigma_i)\mathbb{P}(\sigma_j)} \Lambda_i^{\sigma_i}(x_i) \Lambda_j^{\sigma_j}(x_j) \right) dx_i dx_j, \\ &\leq \log \left(\int \mathcal{P}(x_i, x_j) \sum_{\sigma_i, \sigma_j} \frac{\mathbb{P}(\sigma_i, \sigma_j)}{\mathbb{P}(\sigma_i)\mathbb{P}(\sigma_j)} \Lambda_i^{\sigma_i}(x_i) \Lambda_j^{\sigma_j}(x_j) dx_i dx_j \right), \end{aligned}$$

with $\Lambda^1 \stackrel{\text{def}}{=} \Lambda$ and $\Lambda^0 \stackrel{\text{def}}{=} 1 - \Lambda$. Defining $\mathcal{P}_{\sigma_i\sigma_j} \stackrel{\text{def}}{=} \mathbb{E}_{\mathcal{P}}[\Lambda_i^{\sigma_i}(X_i)\Lambda_j^{\sigma_j}(X_j)]$, we get the final expression

$$\mathbb{I}(X_i, X_j) \leq \log \left(\sum_{\sigma_i, \sigma_j} \frac{\mathbb{P}(\sigma_i, \sigma_j)}{\mathbb{P}(\sigma_i)\mathbb{P}(\sigma_j)} \mathcal{P}_{\sigma_i\sigma_j} \right) \leq \log(2),$$

because we have $\mathbb{P}(\sigma_i, \sigma_j) \leq \mathbb{P}(\sigma_j)$ and $\sum_{\sigma_j} \mathcal{P}_{\sigma_i\sigma_j} = \mathbb{P}(\sigma_i)$. □

We will focus first on the estimation of the pairwise distribution p_{ij} of (σ_i, σ_j), without discussing how to estimate the joint distribution of $\boldsymbol{\sigma}$ from them. We will come back to this problem in the end of this section.

3.1 Pairwise distributions estimation

The choice of the encoding functions Λ_i imposes the marginal distributions of the latent variables σ_i; indeed we have seen that

$$p_i^1 = \mathbb{P}(\sigma_i = 1) = \mathbb{E}[\Lambda_i(X_i)] = \int_{\mathcal{X}_i} \Lambda_i(x) dF_i(x).$$

These parameters can easily be estimated using empirical moments and it will only remain to estimate the correlation parameter p_{ij}^{11}. We propose here to carry out a maximum likelihood estimation. The estimation of each parameter p_{ij}^{11} is independent of the others and we carry out one unidimensional likelihood maximization per edge. For the sake of simplicity, we assume that the random variables admit probability distribution functions. The joint pdf of (X_i, X_j) associated to the distribution p_{ij} will be referred to as

$$f_{p_{ij}}^{ij}(x_i, x_j) \stackrel{\text{def}}{=} \sum_{s_i, s_j} p_{ij}(s_i, s_j) f_i^{s_i}(x_i) f_j^{s_j}(x_j),$$

where $f_i^{s_i}$ is the pdf associated to $dF_i^{s_i}$. Let us first express the logarithm of the likelihood of a distribution p_{ij} of (σ_i, σ_j) corresponding to the pairwise observations $\mathbf{x}$ described in (1).

$$\begin{aligned} L(\mathbf{x}, p_{ij}) &= \sum_{k=1}^{N_{ij}} \log f_{p_{ij}}^{ij}(x_i^k, x_j^k) \\ &= \sum_{k=1}^{N_{ij}} \log \left(\sum_{s_i, s_j} p_{ij}(s_i, s_j) f_i^{s_i}(x_i^k) f_j^{s_j}(x_j^k) \right). \end{aligned}$$

Because of the hidden variables σ_i and σ_j, a sum appears within the logarithms. Therefore, it will not be possible to find explicitly the distributions p_{ij} maximizing $L(\mathbf{x}, p_{ij})$. The usual approach is to use the Expectation Maximization algorithm (EM) first introduce by Dempster et al. [10]. It consists in building a sequence of (σ_i, σ_j)-distribution $p_{ij}^{(n)}$ with increasing likelihood. Using the following notation

$$p_{ij}^{(n)}(s_i, s_j | x_i, x_j) \stackrel{\text{def}}{=} \mathbb{P}^{(n)}(\sigma_i = s_i, \sigma_j = s_j | X_i = x_i, X_j = x_j),$$

the EM algorithm can be expressed as

$$p_{ij}^{(n+1)} \leftarrow \operatorname*{argmax}_{p_{ij}} \quad \ell(p_{ij} || p_{ij}^{(n)}) \stackrel{\text{def}}{=} \sum_k \sum_{s_i, s_j} p_{ij}^{(n)}(s_i, s_j | x_i^k, x_j^k) \log p_{ij}(s_i, s_j),$$

The derivative of $\ell(p_{ij} || p_{ij}^{(n)})$ with respect to p_{ij}^{11} is

$$\frac{\partial \ell(p_{ij} || p_{ij}^{(n)})}{\partial p_{ij}^{11}} = \sum_{k=1}^{N_{ij}} \sum_{s_i, s_j} \left(2\mathbb{1}_{\{s_i = s_j\}} - 1 \right) \frac{p_{ij}^{(n)}(s_i, s_j | x_i^k, x_j^k)}{p_{ij}(s_i, s_j)},$$

Stationary points yields an obvious solution, which is

$$p_{ij}(s_i, s_j) = \frac{1}{N_{ij}} \sum_{k=1}^{N_{ij}} p_{ij}^{(n)}(s_i, s_j | x_i^k, x_j^k).$$

The function that we maximize being concave, this solution is the unique stationary point of $\ell(p_{ij} || p_{ij}^{(n)})$. We obtain the following update rule for the EM algorithm

$$p_{ij}^{(n+1)}(1,1) \leftarrow \frac{1}{N_{ij}} \sum_{k=1}^{N_{ij}} \frac{\psi_{ij}^{(n)}(1,1) \Lambda_i(x_i^k) \Lambda_j(x_j^k)}{Z_{ij}(x_i^k, x_j^k)}, \tag{12}$$

with

$$\psi_{ij}^{(n)}(s_i, s_j) \stackrel{\text{def}}{=} \frac{p_{ij}^{(n)}(s_i, s_j)}{p_i^{(n)}(s_i) p_j^{(n)}(s_j)},$$

$$Z_{ij}(x_i, x_j) \stackrel{\text{def}}{=} \sum_{s_i, s_j} \psi_{ij}^{(n)}(s_i, s_j) \Lambda_i^{s_i}(x_i) \Lambda_j^{s_j}(x_j),$$

and $\Lambda^1 \stackrel{\text{def}}{=} \Lambda$, $\Lambda^0 \stackrel{\text{def}}{=} 1 - \Lambda$. The update rule (12) is quite simple, although one has to check that the estimated parameter is valid, i.e. $p_{ij}^{11} \in \mathbb{D}(p_i^1, p_j^1)$. If it is not the case, it means that the parameter saturates at one bound.

Now that we have proposed a way to estimate the pairwise marginal of the model, we will focus in next section on how to estimate the Ising model of $\boldsymbol{\sigma}$ from them.

3.2 Latent Ising model estimation compatible with BP

We now return to the problem of estimating the joint distribution $p_{\boldsymbol{\sigma}}$ of the random vector $\boldsymbol{\sigma}$ from its pairwise marginals $\{p_{ij}\}_{(i,j)\in\mathbb{E}}$. First, let us remark that (as discussed by Mackay et al. [21]) having compatible marginals does not guaranty the existence of a joint distribution $p_{\boldsymbol{\sigma}}$ such as

$$\forall (i,j) \in \mathbb{E}, \forall s_i, \sum_{\mathbf{s}_{\mathbb{V}\setminus\{i,j\}}} p_{\boldsymbol{\sigma}}(\boldsymbol{\sigma} = \mathbf{s}) = p_{ij}(s_i, s_j),$$

However, in the case where the graph $\mathcal{G} = (\mathbb{V}, \mathbb{E})$ contains no cycles, the joint distribution is entirely determined by its pairwise marginals. This joint distribution is expressed as

$$p_{\boldsymbol{\sigma}}(\boldsymbol{\sigma} = \mathbf{s}) = \prod_{(i,j)\in\mathbb{E}} \frac{p_{ij}(s_i, s_j)}{p_i(s_i) p_j(s_j)} \prod_{i\in\mathbb{V}} p_i(s_i), \tag{13}$$

with p_i the marginal of p_{ij} – independent of j.

In the more general case of a graph containing cycles, the situation is more complex. This inverse Ising model is much studied in statistical physics (see Cocco and Monasson [7] and references within). Potentially it is NP-hard and can have no solution. Only approximate methods can be used for graph of large size. Wainwright [34] proposed an approach of particular interest, which takes into account the fact that once the distribution $p_{\boldsymbol{\sigma}}$ is fixed in an approximate way, the marginalisation will also be performed in an approximate way. The idea is to use compatible approximations for these two tasks. In our case, we wish to use the BP algorithm, described in forthcoming Section 4, to compute the approximate marginals of $p_{\boldsymbol{\sigma}}$. It seems reasonable to impose that, without any observation, the answer given by BP is the historical marginals $\{p_{ij}\}$ and $\{p_i\}$. For doing so, the distribution $p_{\boldsymbol{\sigma}}$ should be chosen under the Bethe approximation (13) which is closely related to the BP algorithm, as we shall see in Section 4. If this choice is a good candidate as starting point, the Bethe approximation is usually too rough and overestimates correlations, and it is thus necessary to improve on it. This can be achieved using various results from linear response theory

(see Welling and Teh [35], Yasuda and Tanaka [36], Mézard and Mora [23]), when the level of correlation is not too high.

We use instead a simple but more robust approach, which is to modify the model using a single parameter α such as

$$p_{\boldsymbol{\sigma}}(\boldsymbol{\sigma} = \mathbf{s}) = \prod_{(i,j)\in\mathbb{E}} \left(\frac{p_{ij}(s_i, s_j)}{p_i(s_i)p_j(s_j)} \right)^{\alpha} \prod_{i\in\mathbb{V}} p_i(s_i). \tag{14}$$

α can roughly be interpreted as an inverse temperature, which role is to avoid overcounting interactions when the graph contains cycles. This parameter can easily be calibrated by finding a phase transition w.r.t. α. Indeed, for $\alpha = 0$, the BP output is exactly $\{p_i\}$ and, when α increases, it remains close to it until some discontinuity appears (see Furtlehner et al. [14]). In some sense, the best α corresponds to the maximal interaction strength such that the BP output remains close to $\{p_i\}$.

4 A message passing inference algorithm

According to the results of Section 2, observations about the real-valued random variable X_i are converted into knowledge of the marginal distribution of σ_i. In order to estimate the distributions of the others binary latent variables, we need an inference algorithm allowing us to impose this marginal constraint to node i when X_i is observed. For this task, we propose a modified version of the BP algorithm.

4.1 The BP algorithm

We present here the BP algorithm, first described by Pearl [27], in a way very similar to the one of Yedidia et al. [37]. We use in this section a slightly more general notation than in Section 1, since instead of considering only pairwise interactions, variables in the set $\mathbb{V}$ interact through factors, which are subsets $a \subset \mathbb{V}$ of variables. If $\mathbb{F}$ is this set of factors, we consider the following probability measure

$$\mathbb{P}(\boldsymbol{\sigma} = \mathbf{s}) = \prod_{a\in\mathbb{F}} \psi_a(\mathbf{s}_a) \prod_{i\in\mathbb{V}} \phi_i(s_i), \tag{15}$$

where $\mathbf{s}_a = \{s_i, i \in a\}$. It is also possible to see variables and factors as nodes of a same bipartite graph, in which case the shorthand notation $i \in a$ should be interpreted as "there is an edge between i and a". $\mathbb{F}$ together with $\mathbb{V}$ define a factor graph, such as defined by Kschischang et al. [20]. The set $\mathbb{E}$ of edges contains all the couples $(a, i) \in \mathbb{F} \times \mathbb{V}$ such that $i \in a$. We denote by d_i the degree of the variable node i. The BP algorithm is a message passing procedure, which output is a set of estimated marginal probabilities, the beliefs $b_a(\mathbf{s}_a)$ (including single nodes beliefs $b_i(s_i)$). The idea is to factor

the marginal probability at a given site as a product of contributions coming from neighboring factor nodes, which are the messages. With definition (15) of the joint probability measure, the updates rules read:

$$m_{a\to i}(s_i) \leftarrow \sum_{\mathbf{s}_{a\setminus i}} \psi_a(\mathbf{s}_a) \prod_{j\in a\setminus i} n_{j\to a}(s_j), \tag{16}$$

$$n_{i\to a}(s_i) \stackrel{\text{def}}{=} \phi_i(s_i) \prod_{a'\ni i, a'\neq a} m_{a'\to i}(s_i), \tag{17}$$

where the notation $\sum_{\mathbf{s}_a}$ should be understood as summing all the variables σ_i, $i \in a \subset \mathbb{V}$, over the realizations $s_i \in \{0, 1\}$. In practice, the messages are often normalized so that $\sum_{s_i} m_{a\to i}(s_i) = 1$.

At any point of the algorithm, one can compute the current beliefs as

$$b_i(s_i) \stackrel{\text{def}}{=} \frac{1}{Z_i} \phi_i(s_i) \prod_{a\ni i} m_{a\to i}(s_i), \tag{18}$$

$$b_a(\mathbf{s}_a) \stackrel{\text{def}}{=} \frac{1}{Z_a} \psi_a(\mathbf{s}_a) \prod_{i\in a} n_{i\to a}(s_i), \tag{19}$$

where Z_i and Z_a are normalization constants that ensure that

$$\sum_{\sigma_i} b_i(\sigma_i) = 1, \qquad \sum_{\boldsymbol{\sigma}_a} b_a(\boldsymbol{\sigma}_a) = 1. \tag{20}$$

When the algorithm has converged, the obtained beliefs b_a and b_i are compatible:

$$\sum_{\mathrm{s}_{a\setminus i}} b_a(\mathrm{s}_a) = b_i(s_i). \tag{21}$$

Yedidia et al. [37] proved that the belief propagation algorithm is an iterative way of solving a variational problem: namely it minimizes the Kullback-Leibler divergence $D_{KL}(b\|p)$ to the true probability measure (15) over all Bethe approximations on the factor graph, of the form

$$b(\mathbf{s}) = \prod_{a\in\mathbb{F}} \frac{b_a(\mathbf{s}_a)}{\prod_{i\in a} b_i(s_i)} \prod_{i\in\mathbb{V}} b_i(s_i),$$

subject to constraints (20)–(21). The approximation is actually exact when the underlying graph is a tree. The stationary points of the above variational problem are beliefs at a fixed point of the BP algorithm (see Yedidia et al. [37]). This alternative description of BP will be used in the next section to derive a new variant of the algorithm.

4.2 Imposing beliefs: mirror BP

In the following, $\mathbb{V}^*$ will be the set of nodes i such that X_i is observed. Assuming that the model (ψ_a and ϕ_i) is given, we wish to include in the algorithm some constraints on the beliefs of the form

$$\forall i \in \mathbb{V}^*, \forall s_i \in \{0,1\}, b_i(s_i) = b_i^*(s_i). \tag{22}$$

We suppose in the following that each b_i^* is normalized. The issue of how to convert real-valued observation to this distribution b_i^* has been studied in Section 2. We seek to obtain a new update rule from the Kullback-Leibler divergence minimization, with the additional constraints (22). Constraints of this form as sometimes referred to as "soft constraints" in the Bayesian community (Bilmes [4]).

We start from the Lagrangian of the minimization problem:

$$\begin{aligned}\mathcal{L}(b,\lambda) =& D_{KL}(b\|p) + \sum_{\substack{i\in\mathbb{V}\setminus\mathbb{V}^*\\ a\ni i,s_i}} \lambda_{ai}(s_i)\Big(b_i(s_i) - \sum_{\mathbf{s}_{a\setminus i}} b_a(\mathbf{s}_a)\Big) \\ &+ \sum_{\substack{i\in\mathbb{V}^*\\ a\ni i,s_i}} \lambda_{ai}(s_i)\Big(b_i^*(s_i) - \sum_{\mathbf{s}_{a\setminus i}} b_a(\mathbf{s}_a)\Big) + \sum_{i\in\mathbb{V}} \gamma_i \left(\sum_{s_i} b_i(s_i) - 1\right),\end{aligned}$$

with $D_{KL}(b\|p)$ defined as

$$D_{KL}(b\|p) \stackrel{\text{def}}{=} \sum_{a,\boldsymbol{s}_a} b_a(\boldsymbol{s}_a) \log \frac{b_a(\boldsymbol{s}_a)}{\psi_a(\boldsymbol{s}_a)} + \sum_{i,s_i} b_i(s_i) \log \frac{b_i(s_i)^{1-d_i}}{\phi_i(s_i)}.$$

The stationary points satisfy

$$\begin{cases} b_a(\mathbf{s}_a) = \psi_a(\mathbf{s}_a) \exp\Big(\sum_{i\in a} \lambda_{ai}(s_i) - 1\Big), \ \forall a \in \mathbb{F}, \\ b_i(s_i) = \phi_i(s_i) \exp\Big(\frac{\sum_{a\ni i} \lambda_{ai}(s_i)}{d_i - 1} + 1 - \gamma_i\Big), \forall i \notin \mathbb{V}^*, \\ b_i(s_i) = b_i^*(s_i), \ \forall i \in \mathbb{V}^*. \end{cases}$$

Following Yedidia et al. [37], we introduce the parametrization

$$\lambda_{ai}(s_i) = \log n_{i\to a}(s_i),$$

for all edges $(ai) \in \mathbb{E}$. Note that we do not consider any node in $\mathbb{V} \setminus \mathbb{V}^*$ of degree d_i equal to 1 since they play no role in the minimization problem. For all nodes $i \notin \mathbb{V}^*$ we also have

$$\lambda_{ai}(s_i) = \log\Big[\phi_i(s_i) \prod_{b\ni i, b\neq a} m_{b\to i}(s_i)\Big].$$

Then it follows that, whenever $i \notin \mathbb{V}^*$,

$$n_{i\to a}(s_i) = \phi_i(s_i) \prod_{b\ni i, b\neq a} m_{b\to i}(s_i).$$

Enforcing the compatibility constraints on nodes $i \notin \mathbb{V}^*$ shows that update rules (16)–(17) are still valid for these nodes. For $i \in \mathbb{V}^*$, the compatibility constraints yield

$$\begin{aligned} b_i^*(s_i) &= \sum_{\mathbf{s}_{a\setminus i}} b_a(\mathbf{s}_a) = \sum_{\mathbf{s}_{a\setminus i}} \psi_a(\mathbf{s}_a) \prod_{j\in a} n_{j\to a}(s_j) \\ &= n_{i\to a}(s_i) \sum_{\mathbf{s}_{a\setminus i}} \psi_a(\mathbf{s}_a) \prod_{j\in a} n_{j\to a}(s_j). \end{aligned}$$

Until now the message from a factor a to a variable $i \in \mathbb{V}^*$ has not been defined. For convenience we define it as in (16) and the preceding equation becomes

$$n_{i\to a}(s_i) m_{a\to i}(s_i) = b_i^*(s_i),$$

as in the usual BP algorithm. This leads to a definition that replaces (17) when $i \in \mathbb{V}^*$

$$n_{i\to a}(s_i) = \frac{b_i^*(s_i)}{m_{a\to i}(s_i)} = \frac{b_i^*(s_i)}{b_i(x_i)} \phi_i(s_i) \prod_{b\ni i, b\neq a} m_{b\to i}(s_i). \tag{23}$$

Therefore the message (23) is the BP message (17) multiplied by the ratio of the belief we are imposing over the "current belief" computed using (18). This is very similar to iterative proportional fitting (IPF, see Darroch and Ratcliff [9]). To sum up, the characteristics of this new variant of Belief Propagation are

- all factors and all variables which value has not been fixed send the same messages (16)–(17) as in classic BP;
- variables which value has been fixed use the new messages (23);
- beliefs for factors or for variables which value has not been fixed are still computed using (18)–(19);

In the classical BP algorithm, the information sent by one node can only go back to itself through a cycle of the graph. When (23) is used, however, the variable with fixed value acts like a mirror and sends back the message to the factor instead of propagating it through the graph. It is to emphasize this property that we call our new method the *mirror BP* (mBP) algorithm. Note that it could be defined for variables valued in any discrete alphabet.

A very similar algorithm to our mBP has been proposed by Teh and Welling [33]. Their algorithm is described as iterations of successive BP runs on unobserved nodes and IPF on nodes in $\mathbb{V}^*$. The update (23) is just obtained as direct IPF. The main drawback of their version is that it assumes a particular update ordering because they consider that the updates (17)–(23) are of different nature, which is in fact not really necessary and complicates its use.

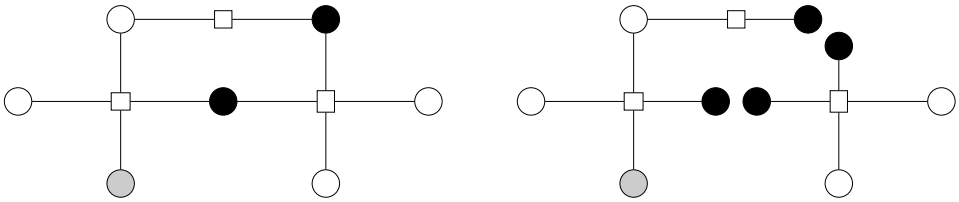


Figure 3: Illustration of Proposition 5. If only black nodes are in $\mathbb{V}^*$, Proposition 5 tells us that mBP converges since the resulting graph $\mathcal{T}(\mathcal{G}, \mathbb{V}^*)$ (right graph) contains two disconnected trees with exactly two nodes in $\mathbb{V}^*$. If we add the gray node in $\mathbb{V}^*$ then Proposition 5 does not apply, the right tree contains three nodes in $\mathbb{V}^*$, and we cannot conclude about convergence. However, on the other part of the graph Proposition 5 still holds.

It is known that BP can exhibit non convergent behavior in loopy networks, although sufficient conditions for convergence are known (see e.g. Mooij and Kappen [26], Tatikonda and Jordan [32], Ihler et al. [16]). Since the mirroring behavior of our algorithm seems to be quite different, we present some sufficient conditions for convergence.

Definition 1. *Let $\mathcal{T}(\mathcal{G}, \mathbb{V}^*)$ be the factor graph where each node $i \in \mathbb{V}^*$ has been cloned d_i times, each clone being attached to one (and only one) neighbor of i. We call "graph cutting at $\mathbb{V}^*$" the transformation $\mathcal{T}(\cdot, \mathbb{V}^*)$ applied to a factor graph $\mathcal{G}$ for a given set of variable nodes $\mathbb{V}^*$.*

Example of a such "graph cutting" $\mathcal{T}(\mathcal{G}, \mathbb{V}^*)$ is shown in Figure 3. The following proposition describes cases where the mBP algorithm is guaranteed to converge.

Proposition 5. *If the graph $\mathcal{T}(\mathcal{G}, \mathbb{V}^*)$ is formed by disconnected trees containing not more than two leaves cloned from $\mathbb{V}^*$, the mBP algorithm is stable and converges to a unique fixed point.*

Proof. See Appendix. □

5 Numerical experiments

In order to understand its behavior, we apply here the method described in this paper to synthetic data. We will consider three cases of increasing complexity:

- a pair (X_1, X_2) of real-valued random variables,
- a tree with interior connectivity fixed,
- a rough road traffic network.

For each case, we repeat the following *decimation experiment*: for an outcome of the random vector $\mathbf{X}$, we observe its components X_i in a random order and we make prediction about unobserved components using our method. This will allow us to compare the performance of the different choices of encoding and decoding functions when the proportion of observed variables varies.

We will consider the following choices for the encoding and decoding functions:

- the step function Λ_{MI} with the Bayesian decoding function (9),
- the cumulative distribution function F with its inverse $\Gamma^{\mathcal{L}} = F^{-1}$,
- the cumulative distribution function F with the decoding function $\Gamma^{\mathcal{B}}$ of (10).

Each of these choices yields an estimator θ for which we will compute the performance w.r.t. the L^1 norm

$$\mathbb{E}_X\Big[|\theta(X) - X|\Big]. \tag{24}$$

Model generation. These synthetic models are based on Gaussian copulas with support corresponding to one of the three cases previously described. More precisely, it corresponds to the support of the precision matrix, i.e. the inverse covariance matrix, of the Gaussian vector $\mathbf{Y}$. For doing so, we randomly generate the partial correlations, the entries of the precision matrix of $\mathbf{Y}$, with uniform random variables on $[-1, -0.2] \cup [0.2, 1]$. Since this will not always lead to a positive definite precision matrix, we use this matrix as a starting point and reduce the highest correlation until it becomes definite positive.

We can then generate outcomes of this Gaussian vector $\mathbf{Y}$ and transform them, using the function which maps a Gaussian variable $\mathcal{N}(0,1)$ into a variable of chosen cdf F_X. More precisely, each component of the vector $\mathbf{X}$ is defined as

$$X_i = F_X^{-1}\left(F_{\mathcal{N}(0,1)}(Y_i)\right),$$

where $F_{\mathcal{N}(0,1)}$ is the cdf of a $\mathcal{N}(0,1)$ variable. This procedure will allow us to perform exact inference using the Gaussian vector $\mathbf{Y}$ while the dependency of the vector $\mathbf{X}$ is based on a Gaussian copula.

We will sometimes consider the case of $\beta(a,b)$ variables, so let us recall their pdfs $f^{\beta}_{a,b}$ for $a, b \in]0, +\infty[$

$$f^{\beta}_{a,b}(x) = \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1}\mathbb{1}_{[0,1]}(x),$$

where the normalization constant $B(a,b)$ is the bêta function. These distributions are of particular interest because different cases arise depending

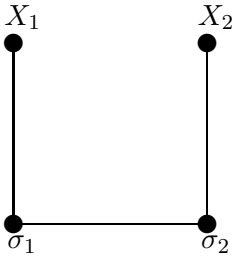


Figure 4: Model of the random vector $(X_1, X_2, \sigma_1, \sigma_2)$. The true distribution of (X_1, X_2) is approximated through the latent variables σ_1 and σ_2.

on the parameters a and b. Indeed, it is possible to obtain almost binary $(a, b \to 0)$, unimodal $(a, b > 1)$ or uniform $(a, b = 1)$ distributions on $[0, 1]$.

A pair (X_1, X_2) of real-valued random variables. Let us start with the simple case where the vector $\mathbf{X}$ is just two random real-valued variables (Figure 4). We repeat 100,000 times the decimation experiment. In this case, this experiment is just to observe either X_1 or X_2 for a given outcome of the vector (X_1, X_2) and to predict the other variable. In addition to the L^1 performance, we compute the biases of the different estimators θ

$$\mathbb{E}_X \left[\theta(X) - \hat{\theta}_1(X) \right]. \tag{25}$$

We recall that $\hat{\theta}_1(X)$ is the optimal predictor w.r.t. the L^1 distance i.e. the conditional median.

The results, for various values of a, b and $\rho \stackrel{\text{def}}{=} \text{cov}(Y_i, Y_j)$, are given in Table 1. The first line is the L^1 performance (24) and the second one the bias (25). Generally, with weak correlations all estimators have a satisfactory behavior. However the best choice is the cdf function $\Lambda = F$ with the inverse mapping $\Gamma^{\mathcal{L}} = F^{-1}$. As expected, the conservative property of the decoding $\Gamma^{\mathcal{B}}$ is a real drawback in the case of strong correlation because it prevents from predicting extreme values (see Figure 2). When considering symmetric variables X_i, all estimators biases are close to 0. Even with $\beta(2, 3)$ variables, this bias is negligible. In the case of asymmetrical variables, these biases are clearly non zero, but do not prevent from obtaining good performance.

Regular trees. We consider here the case of a tree with a given connectivity n for interior nodes: each non leaf node has exactly n neighbors. For $n = 3$, we get a binary tree. We perform the decimation experiments and results are presented in Figure 5. For the sake of comparison, we show three other predictors: the median (in red), the k nearest neighbors (k-NN, Cover and Hart [8]) predictor (in orange) and the perfect predictor (in black), which is obtained by computing the conditional mean of the vector $\mathbf{Y}$. The

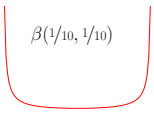
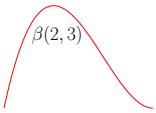
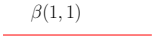
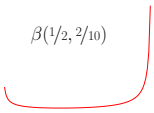
Density	Correlation	$(F, \Gamma^{\mathcal{D}})$	$(F, \Gamma^{\mathcal{P}})$	Λ_{MI}	Exact
	$\rho = 0.5$	**0.2%** -0.34	3.1% -0.79	3.3% 0.01	32.04 0
	$\rho = 0.9$	**0.1%** -0.15	22% -0.72	9.5% -0.06	13.62 0
	$\rho = 0.5$	**1.4%** -0.26	4.4% 0.47	6.7% 0.44	14.22 0
	$\rho = 0.9$	**1.3%** 0.1	68.8% 1.25	61.7% 0.92	6.89 0
	$\rho = 0.5$	**0.1%** 0.02	4.6% 0.01	7.3% -0.03	20.96 0
	$\rho = 0.9$	**0.4%** 0.02	66.2% -0.09	58.6% 0	9.83 0
	$\rho = 0.5$	**2.7%** 6.77	4.4% -5.14	7.5% -4.11	23 0
	$\rho = -0.7$	**4.1%** 5.74	16.9% -8.8	20.8% -5.16	18.21 0

Table 1: Performances of various predictors in the case of Figure 4. The first line is mean L^1 error in % of deviation from the optimal performance. The second one is its bias. Bold values are the best performing choices.

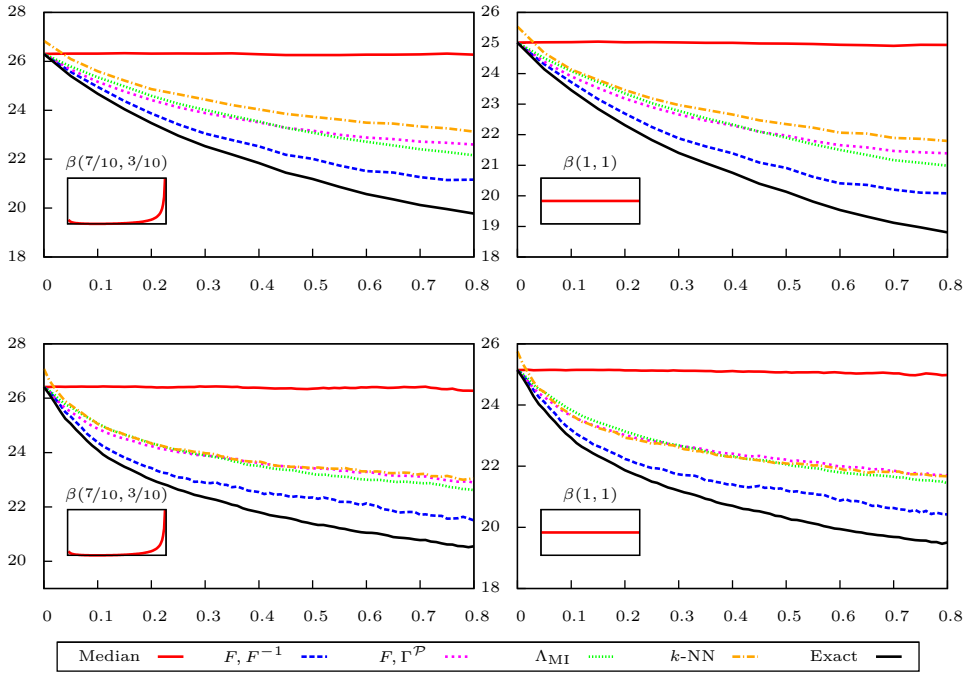


Figure 5: Mean L^1 prediction error of unobserved variables ($\times 100$) as a function of the proportion of revealed variables; the small embedded figures are the corresponding pdf of the bêta variables. The connectivity is $n = 3$ for top figures and $n = 5$ for the bottom ones. Each tree contains 100 variables.

k-NN predictor is manually optimized to the $k = 50$ nearest neighbors in the whole historical data used to build the model. This predictor is known to be a good choice for road traffic data (Smith et al. [30]), but its complexity is too high for large networks compared to BP. Moreover, it requires complete observations of the network, which are not available when dealing with probe vehicles data.

As a general rule, the choice (F, F^{-1}) seems to be the best one. Let us remark that, if we continue to increase the connectivity n, this situation can change. In fact, at very high connectivity ($n \sim 10$), the convergence of mBP can be very slow. Non convergent cases can then impact the result and one should rather use Λ_{MI}. Indeed, for the choice Λ_{MI} the mBP algorithm is strictly equivalent to BP. In this case, the BP algorithm is more stable since it is always converging on trees. At this point, we discard the choice $(F, \Gamma^{\mathcal{B}})$ which is clearly inferior to the other ones.

A simple road network model. Let us finally consider a new synthetic model, associated to the road network of Figure 6, which is a very rough description of a city network. The dependency graph of the vector $\mathbf{X}$ is basically the line graph of the road networks, i.e. there is a direct dependency

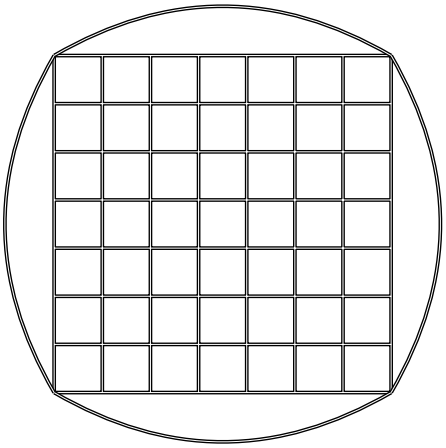


Figure 6: A simple model of urban road network with two-way streets. The inner grid represents the city itself and the 2×4 exterior edges form a ring road around it.

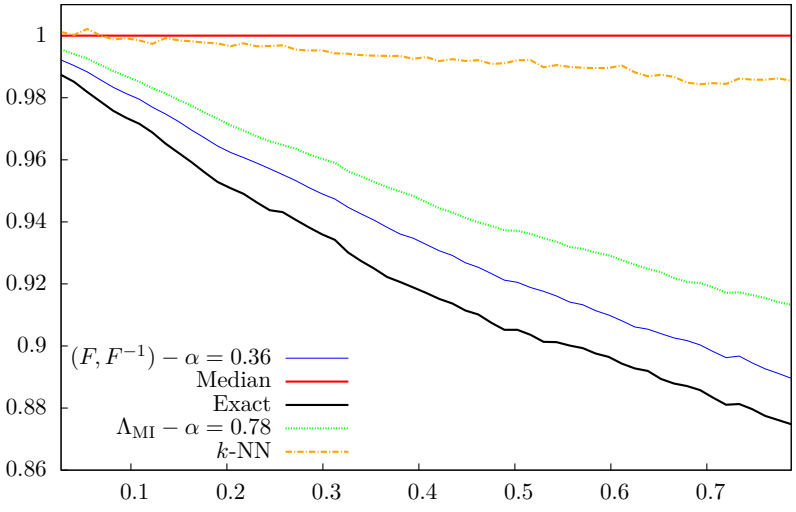


Figure 7: Mean L^1 prediction error on unobserved variables, as a function of the proportion of revealed variables, for the urban network of Figure 6. All values are relative to the error made by the "median" predictor.

between edges i and j iff they are adjacent in the road network. To model the impact of a ring road on its neighborhood, we set their partial correlations with adjacent edges to 0.3. The marginal distributions of travel times are real data coming from the Australian M4 motorway. We assume that the ring road links are always observed, by means of specific equipment such as magnetic loops.

Again, the decimation experiment is performed 1,000 times and the results are presented in Figure 7. Since the ring road links are always observed, the decimation curve begin at $\rho \sim 0.03$. Note that, in this case, the k-NN predictor performance is very bad, due to the fact that correlations are small compared to the vector dimension (Beyer et al. [2]). The parameter α of (14) is estimated with a dichotomy search on $[0, 1]$ up to a precision of 0.01. Once again, the best choice of encoding function is the cdf, which performs clearly better than Λ_{MI}.

6 Conclusion

We proposed a simple way to model the interaction between real-valued random variables defined over a graph from the following information:

- the empirical cumulative distribution function of each variable;
- an incomplete covariance matrix.

The choice of the cdf as encoding function and its inverse as decoding function seems to be the best one, as long as the graph connectivity is not too high. When this connectivity increases too much, the algorithm mBP loose its efficiency and one should rather choose the encoding function Λ_{MI}. An important but potentially difficult task has been discarded here: finding the dependency graph structure. This task can be performed using greedy heuristics (see Jalali et al. [17], Furtlehner et al. [15]) or L^1-based regularization method (see Ravikumar et al. [29])

Once the encoding/decoding functions are chosen and the marginals p_{ij} have been estimated, many available methods exists to define the latent Ising model, i.e. the set of Ising couplings. The best one will depend on the data and determining it will require tests on real data. However, the results presented here make us quite optimistic about applying this method to road traffic data, for which the underlying binary description seems natural.

Straightforward generalization of the approach presented here can be carried out to construct latent variables with a feature space larger than $\{0, 1\}$, by considering additional random thresholds defined in Section 2.1 or deterministic ones; the underlying principles remain unchanged. In particular, it is still possible to build decoding functions based on ML or Bayesian updating, to use the EM algorithm for pairwise distributions estimations and the mBP algorithm for inference.

References

[1] R. Baxter. *Exactly solved models in statistical mechanics*. Dover Publications, 2008.

[2] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *Database Theory-ICDT'99*, pages 217–235. Springer, 1999.

[3] D. Bickson. *Gaussian Belief Propagation: Theory and Application*. PhD thesis, Hebrew University of Jerusalem, 2008.

[4] J. Bilmes. On soft evidence in bayesian networks. Technical report, University of Washington, 2004.

[5] X. Boyen. *Inference and Learning in Complex Stochastic Processes*. PhD thesis, Stanford University, Computer Science Department, 2002. 229 pages.

[6] H. Chan and A. Darwiche. On the revision of probabilistic beliefs using uncertain evidence. *Artificial Intelligence*, 163(1):67–90, 2005.

[7] S. Cocco and R. Monasson. Adaptive cluster expansion for the inverse Ising problem: convergence, algorithm and tests. *Journal of Statistical Physics*, 147(2):252–314, 2012.

[8] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[9] J. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

[10] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[11] A. Doucet, N. de Freitas, and N. Gordon. An introduction to sequential Monte Carlo methods. *Sequential Monte Carlo methods in practice*, 2001.

[12] C. Furtlehner, J.-M. Lasgouttes, and A. de La Fortelle. A Belief Propagation approach to traffic prediction using probe vehicles. In *Proceedings of the 10th International IEEE Conference on Intelligent Transportation Systems*, pages 1022–1027, 2007.

[13] C. Furtlehner, Y. Han, J.-M. Lasgouttes, V. Martin, F. Marchal, and F. Moutarde. Spatial and temporal analysis of traffic states on large scale networks. In *Procceding of the 13th International IEEE Conference on Intelligent Transportation Systems*, pages 1215–1220, 2010.

[14] C. Furtlehner, J.-M. Lasgouttes, and A. Auger. Learning multiple Belief Propagation fixed points for real time inference. *Physica A: Statistical Mechanics and its Applications*, 389(1):149–163, 2010.

[15] C. Furtlehner, Y. Han, J.-M. Lasgouttes, and V. Martin. Pairwise MRF calibration by perturbation of the Bethe reference point. Rapport de recherche 8059, INRIA, 2012.

[16] A. Ihler, J. I. Fischer, and A. Willsky. Loopy Belief Propagation: Convergence and effects of message errors. *The Journal of Machine Learning Research*, 6:905–936, 2005.

[17] A. Jalali, C. Johnson, and P. Ravikumar. On learning discrete graphical models using greedy methods. *arXiv preprint arXiv:1107.3258*, 2011.

[18] E. T. Jaynes. Prior probabilities. *Systems Science and Cybernetics, IEEE Transactions on*, 4(3):227–241, 1968.

[19] E. T. Jaynes. *Probability Theory: The Logic of Science (Vol 1)*. Cambridge University Press, 2003. ISBN 0521592712.

[20] F. R. Kschischang, B. J. Frey, and H. Loeliger. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519, 2001.

[21] D. J. Mackay, J. S. Yedidia, W. T. Freeman, Y. Weiss, et al. A conversation about the Bethe free energy and sum-product. Available at `http://www.merl.com/publications/TR2001-018/`, 2001.

[22] V. Martin. *Modélisation Probabiliste et inférence par l'algorithme belief propagation*. PhD thesis, Mines-ParisTech, 2013.

[23] M. Mézard and T. Mora. Constraint satisfaction problems and neural networks: A statistical physics perspective. *Journal of Physiology-Paris*, 103(1-2):107–113, 2009.

[24] W. Min and L. Wynter. Real-time road traffic prediction with spatio-temporal correlations. *Transportation Research Part C*, 19:606–616, 2011.

[25] T. Minka. Expectation Propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in Artificial Intelligence*, pages 362–369, 2001.

[26] J. M. Mooij and H. J. Kappen. Sufficient conditions for convergence of the sum-product algorithm. *Information Theory, IEEE Transactions on*, 53(12):4422–4437, 2007.

[27] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference*. Morgan Kaufmann, 1988.

[28] PUMAS project. `http://team.inria.fr/pumas/` (in French).

[29] P. Ravikumar, M. J. Wainwright, and J. D. Lafferty. High-dimensional Ising model selection using L^1 regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319, 2010.

[30] B. L. Smith, B. M. Williams, and R. Keith Oswald. Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10(4):303–321, 2002.

[31] E. Sudderth, A. Ihler, M. Isard, W. Freeman, and A. Willsky. Nonparametric Belief Propagation. *Communications of the ACM*, 53(10): 95–103, Oct. 2010.

[32] S. Tatikonda and M. Jordan. Loopy Belief Propagation and Gibbs measures. In *Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence*, pages 493–50, 2002.

[33] Y. W. Teh and M. Welling. Passing and bouncing messages for generalized inference. Technical report, UCL, 2001.

[34] M. J. Wainwright. Estimating the "wrong" graphical model: benefits in the computation-limited setting. *The Journal of Machine Learning Research*, 7:1829–1859, 2006.

[35] M. Welling and Y. W. Teh. Approximate inference in Boltzmann machines. *Artificial Intelligence*, 143(1):19–50, 2003.

[36] M. Yasuda and K. Tanaka. Approximate learning algorithm in Boltzmann machines. *Neural computation*, 21(11):3130–3178, 2009.

[37] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized Belief Propagation algorithms. *Information Theory, IEEE Transactions on*, 51(7):2282–2312, 2005.

A Proof of Proposition 5

Let us focus first on the case of one factor with two binary variables σ_i and σ_j, both observed (Figure 8 with $n = 2$). The messages $m_{a\to i}$ are assumed to be normalized such that

$$\sum_{s_i} m_{a\to i}(s_i) = 1.$$

We introduce the following notation

$$u_n \stackrel{\text{def}}{=} m_{a\to i}(0), \quad v_n \stackrel{\text{def}}{=} m_{a\to j}(0),$$

so that $1-u_n = m_{a\to i}(1)$ and $1-v_n = m_{a\to j}(1)$. Using the update rules (16)–(17), one obtains:

$$u_{n+1} = \frac{\psi_{00}\alpha_j\bar{v}_n + \psi_{01}\bar{\alpha}_j v_n}{(\psi_{00}+\psi_{10})\alpha_j\bar{v}_n + (\psi_{01}+\psi_{11})\bar{\alpha}_j v_n}, \tag{26}$$

$$v_{n+1} = \frac{\psi_{00}\alpha_i\bar{u}_n + \psi_{10}\bar{\alpha}_i u_n}{(\psi_{00}+\psi_{01})\alpha_i\bar{u}_n + (\psi_{10}+\psi_{11})\bar{\alpha}_i u_n}, \tag{27}$$

where $\alpha_i \stackrel{\text{def}}{=} b_i^*(0)$, $\psi_{yz} \stackrel{\text{def}}{=} \psi(\sigma_i = y, \sigma_j = z)$ and using the convention $\bar{z} \stackrel{\text{def}}{=} 1-z$.

Lemma 6. *The sequences $(u_n)_{n\in\mathbb{N}}$ and $(v_n)_{n\in\mathbb{N}}$, defined recursively by (26) and (27), converge to a unique fixed point for any $(u_0, v_0) \in]0,1[^2$.*

Proof. Since the roles of u_n or v_n are symmetric, we will only prove the convergence of u_n. From (26) and (27), we obtain a recursive equation of the form $u_{n+2} = f(u_n)$ such as

$$f(x) = \frac{h_0 x + K_0}{(h_0+h_1)x + (K_0+K_1)},$$

with

$$\begin{aligned}
h_0 &\stackrel{\text{def}}{=} \psi_{00}\alpha_j(\bar{\alpha}_i\psi_{11} - \alpha_i\psi_{01}) + \psi_{01}\bar{\alpha}_j(\psi_{10}\bar{\alpha}_i - \psi_{00}\alpha_i),\\
h_1 &\stackrel{\text{def}}{=} \psi_{10}\alpha_j(\bar{\alpha}_i\psi_{11} - \alpha_i\psi_{01}) + \psi_{11}\bar{\alpha}_j(\psi_{10}\bar{\alpha}_i - \psi_{00}\alpha_i),\\
K_0 &\stackrel{\text{def}}{=} \psi_{00}\psi_{01}\alpha_i, \quad K_1 \stackrel{\text{def}}{=} \alpha_i(\psi_{10}\psi_{01}\alpha_j + \psi_{11}\psi_{00}\bar{\alpha}_j).
\end{aligned}$$

The derivative of f is

$$f'(x) = \frac{h_0K_1 - h_1K_0}{\left((h_0+h_1)x + (K_0+K_1)\right)^2},$$

which is of constant sign. If $f'(x) \geq 0$, then u_{2n} and u_{2n+1} are monotonic, and, because u_n is bounded, we can conclude that both u_{2n} and u_{2n+1} converge. If we could prove that there is a unique fixed point in the interval $[0,1]$, we would have proved that u_n converges.

Let us begin by discarding some trivial cases. First the case $f(1) = 1$ implies that $\alpha_i = 1$ and

$$\begin{aligned}
h_0 &= -\psi_{00}\psi_{01} = -K_0,\\
h_1 &= -\psi_{10}\psi_{01}\alpha_j - \psi_{11}\psi_{00}\bar{\alpha}_j = -K_1,
\end{aligned}$$

which leads to f being a constant function equal to $\frac{K_0}{K_0+K_1}$. When $f(0) = 0$, one has $\alpha_i = K_0 = K_1 = 0$, and f is again constant. The cases $f(1) = 0$ and $f(0) = 1$ are treated similarly and f is still a constant function, which implies the trivial convergence of u_n.

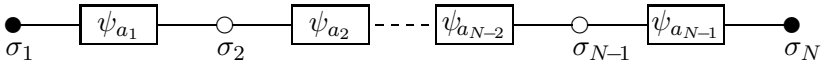


Figure 8: Chain of N pairwise factors, the extremal variables σ_1 and σ_N are observed.

Case 1: f is increasing At least one fixed point exists in $[0,1]$ since $f([0,1]) \subset [0,1]$. Studying the roots of $f(x) - x$ shows that the number of fixed points is at most 2 since these fixed points are roots of a degree 2 polynomial.

Since $f(0) > 0$, f being increasing and $f(1) < 1$ the number of fixed points has to be odd, indeed the graph of f must cross an odd number of times the first bisector. One can conclude that there is only one fixed point in $[0,1]$, so both u_{2n} and u_{2n+1} converge to the same fixed point.

Case 2: f is decreasing We just have to consider the sequence $(1 - u_n)_{n\in\mathbb{N}}$, which is similar, but will be defined by recurrence of the form $1 - u_{n+2} = g(1 - u_n)$ with a function g such as g' is positive and the result of Case 1 applies. □

The case we just studied is in fact much more general than it looks. Indeed, as soon as a tree gets stuck between exactly two nodes with fixed beliefs, the situation is equivalent and leads to the result of Proposition 5.

Proof of Proposition 5. First it is trivial to see that fixing the beliefs of a set of nodes $\mathbb{V}^* \subset \mathbb{V}$ has the effect of the graph cutting $\mathcal{T}(\cdot, \mathbb{V}^*)$ in term of messages propagation. To conclude the proof, it is enough to focus on proving the convergence on a tree with two leaves in $\mathbb{V}^*$. Consider the tree of Figure 8; one can show that it is equivalent to the case of Lemma 6 for a well chosen function ψ. Propagating the updates rules yields $m_{a_1\to 1}(s_1) \leftarrow \Theta$, with

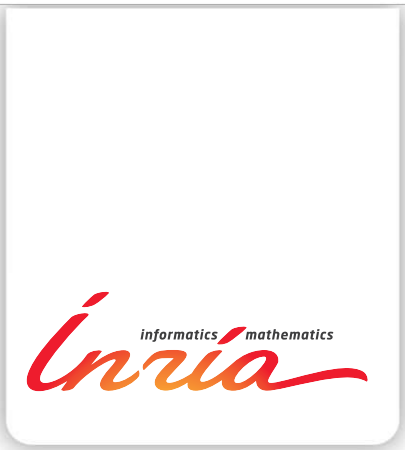
$$\Theta \propto \sum_{s_N} \Big(\sum_{s_1 \dots s_{N-2}} \prod_{i=1}^{N-2} \psi_{a_i}(\mathbf{s}_{a_i}) \phi_i(s_i) \Big) \frac{\psi_{a_{N-1}}(\mathbf{s}_{a_{N-1}}) b_r^*(s_N)}{m_{a_{N-1}\to N}(s_N)}.$$

We define $\tilde{\psi}$ such as

$$\tilde{\psi}(s_1, s_N) = \Big(\sum_{s_1 \dots s_{N-2}} \prod_{i=1}^{N-2} \psi_{a_i}(\mathbf{s}_{a_i}) \phi_i(s_i) \Big) \; \psi_{a_{N-1}}(\mathbf{s}_{a_{N-1}}),$$

then we use the results of Lemma 6 to obtain the convergence of messages on this tree. In the general case of a tree with two leaves in $\mathbb{V}^*$, leaves fixed on variables $\sigma_i, i \in \{1 \dots N\}$ will simply affect the local fields ϕ_i. The leaves fixed on factors a_i will affect the functions ψ_{a_i}. In fact, since the graph is a

tree, we know that the information sent by σ_1 and σ_N to these leaves will not come back to σ_1 and σ_N. These leaves send constant messages, which can be integrated into the functions ψ and ϕ, in order to recover the setting of Lemma 6. □



RESEARCH CENTRE
PARIS – ROCQUENCOURT

Domaine de Voluceau, - Rocquencourt
B.P. 105 - 78153 Le Chesnay Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399