A weighted belief-propagation algorithm for estimating volume-related properties of random polytopes

J. Stat. Mech. (2012) P11003

(http://iopscience.iop.org/1742-5468/2012/11/P11003)

Journal of Statistical Mechanics: Theory and Experiment
An IOP and SISSA journal

A weighted belief-propagation algorithm for estimating volume-related properties of random polytopes

Francesc Font-Clos[1,2], **Francesco Alessandro Massucci**[3] **and Isaac Pérez Castillo**[4]

[1] Centre de Recerca Matemàtica, Edifici C, Campus Bellaterra, E-08193 Bellaterra (Barcelona), Spain
[2] Departament de Matemàtiques, Universitat Autònoma de Barcelona, Edifici C, E-08193 Bellaterra (Barcelona), Spain
[3] Departament d'Enginyeria Química, Universitat Rovira i Virgili, 43007 Tarragona, Spain
[4] Department of Mathematics, King's College London, London WC2R 2LS, UK
E-mail: fontclos@crm.cat, francesco.massucci@urv.cat and isaac.perez_castillo@kcl.ac.uk

Received 11 August 2012
Accepted 14 October 2012
Published 2 November 2012

Online at stacks.iop.org/JSTAT/2012/P11003
doi:10.1088/1742-5468/2012/11/P11003

Abstract. In this work we introduce a novel weighted message-passing algorithm based on the cavity method for estimating volume-related properties of random polytopes, properties which are relevant in various research fields ranging from metabolic networks, to neural networks, to compressed sensing. We propose, as opposed to adopting the usual approach consisting in approximating the real-valued cavity marginal distributions by a few parameters, using an algorithm to faithfully represent the entire marginal distribution. We explain various alternatives for implementing the algorithm and benchmarking the theoretical findings by showing concrete applications to random polytopes. The results obtained with our approach are found to be in very good agreement with the estimates produced by the Hit-and-Run algorithm, known to produce uniform sampling.

Keywords: cavity and replica method, disordered systems (theory), message-passing algorithms, metabolic networks

ArXiv ePrint: 1208.1295

1742-5468/12/P11003+23$33.00

Contents

1. Introduction

There are many problems appearing in various research fields that can be mathematically formalized as requiring the determination of the solution set of a collection of linear equalities or inequalities for real-valued unknown variables. More precisely, given a rectangular $M \times N$ matrix $\boldsymbol{\xi}$ with real entries ξ_i^μ, $\mu = 1, \ldots, M$ and $i = 1, \ldots, N$, and given a vector $\boldsymbol{\gamma} \in \mathbb{R}^M$, with entries γ^μ with $\mu = 1, \ldots, M$, one looks for the set V of vectors $\boldsymbol{x} \in D \subseteq \mathbb{R}^N$ which are solutions to e.g. a set of equalities

$$\sum_{i=1}^{N} \xi_i^\mu x_i = \gamma^\mu, \qquad \mu = 1, \ldots, M, \tag{1}$$

or a set of inequalities

$$\sum_{i=1}^{N} \xi_i^\mu x_i \geq \gamma^\mu, \qquad \mu = 1, \ldots, M. \tag{2}$$

A first example of a problem related to a set of equalities like (1) is that of flux-balance analysis [1]. This is a technique designed to estimate the reaction rates of a set of chemical reactions in the stationary state. Having in mind future applications in this area, we devote some lines to explaining this method. Suppose we have a set of N chemical reactions that produce and consume M chemical substances, and let us denote as c^μ the concentration of the chemical substance μ. Then we can write down the following mass-balance evolution equations:

$$\frac{\mathrm{d}c^\mu(t)}{\mathrm{d}t} = \sum_{i=1}^{N} \xi_i^\mu x_i - \gamma^\mu$$

where x_i is the reaction rate of reaction i, γ^μ is a flux of exchange of chemical substance μ with the environment, and $(\xi_i^1, \ldots, \xi_i^M)$ are the stoichiometric coefficients of reaction i. Generally, each reaction rate x_i is a very complicated function of the concentrations and kinetic parameters, i.e. $x_i = x_i(\boldsymbol{c}, \text{kinetic coefficients, etc})$. Even in cases in which all of these functional relations are known, the resulting equations are highly non-linear and numerically hard to handle. However, if we have good reasons to assume that the system works in the stationary state, that is $\mathrm{d}c^\mu(t)/\mathrm{d}t = 0$, then the reaction rate vector $\boldsymbol{x}$ can be considered as an unknown vector for a set of equalities (1). Besides, due to biochemical constraints, each reaction rate x_i would be such that $x_i \in [x_i^{\min}, x_i^{\max}]$, so $D = \prod_{i=1}^{N} [x_i^{\min}, x_i^{\max}]$. Thus, in this context, estimating the volume of solutions corresponds to estimating reaction rates of a set of chemical reactions in the stationary state. Of course, the problem has been transformed from a potentially numerically intractable task of solving a set of coupled non-linear differential equations to the more tractable yet very time-consuming task of estimating the volume of solutions V of (1). Flux-balance analysis (FBA) goes one step further and simplifies the problem by replacing the whole volume V by one of its solutions. This is achieved by imposing an objective function which must be optimized (see [1] for details).

One example relating to the case of inequalities (2) is von Neumann's expanding model for linear economies [5]–[7], which, due to its interesting applications to metabolic networks [8]–[11], deserves a couple of lines of explanation. In the economic context in which it was originally introduced, this model assumes that an economy is constituted of N companies consuming and producing M commodities. Each company i is able to operate linearly with a scale of operation x_i. If $A = (a_i^\mu)$ and $B = (b_i^\mu)$ are the input and output matrices of such an economy, as some of the input will be used to produce output the ratio of input to output produced cannot be larger than the global growth rate ρ, that is

$$\rho \leq \frac{\sum_{i=1}^{N} b_i^\mu x_i}{\sum_{i=1}^{N} a_i^\mu x_i} \qquad \text{or} \qquad \sum_{i=1}^{N} \left(b_i^\mu - \rho a_i^\mu\right) x_i \geq 0, \qquad \forall\, \mu = 1, \ldots, M. \tag{3}$$

J. Stat. Mech. (2012) P11003

Thus for von Neumann's model of linear economies, one then wonders what the possible values for the vector of operations $\boldsymbol{x}$ are for a given growth rate, and how the optimal growth rate can be achieved.

These two examples are just a glimpse into the myriad problems, old (e.g. Gardner's optimal capacity problem [2]) and new (e.g. compressed sensing [3, 12]), coming from diverse research fields which can be mathematically formalized as either (1) or (2), and which have the same goal: find techniques, either analytic or numerical or a combination of these, for estimating the volume of solutions V and statistical properties related to it.

Thus, rather than focusing on a research field in particular, in the present paper we analyse the problem in general terms, and, for the sake of clarity, we present our new methodology and test it on simple examples, leaving more complex and certainly more exciting applications, particularly in the area of metabolic networks, for future research.

We have organized this paper as follows. In section 2 we set up the problem at hand and present it in a unifying way. In section 3 we apply the cavity method to obtain a set of self-consistency equations for the cavity marginals. We also discuss a set of self-consistency equations for the support of these marginals. As we will see, the set of equations for the supports decouples from the actual shape of the marginals, yielding an extremely simple set of equations, which can be readily solved. Section 4 contains the first main result, consisting in reweighting the cavity equations, in seeking for an efficient way to solve them. In section 5 we propose two main ways for implementing the weighted cavity equations: the method of the histograms and the weighted population dynamics in the instance. Besides this, for the second method we suggest two alternatives: (i) random assignment and variable locking and (ii) variable fixing. We have benchmarked all these novel ideas with some examples and reported the results in section 6. Further theoretical research and applications, particularly in the area of metabolic networks, are discussed in the conclusions in the last part, section 7.

2. Model definitions

As the analytical treatments for the two problems (1) and (2) are not that different, we choose to discuss the more general problem related to the set of inequalities (2). We are particularly interested in heterogeneous systems in which the rectangular matrix $\boldsymbol{\xi}$ is generally random. This can be summed up by saying quite simply that we focus here on estimating volume-related properties of random polytopes. In the so-called H-representation, a polytope is defined as the set of points $\boldsymbol{x} = (x_1, \dots, x_N) \in D = D_1 \times \cdots \times D_N \subseteq \mathbb{R}^N$ encapsulated by M hyperplanes $\{(\boldsymbol{\xi}^\mu, \gamma^\mu)\}_{\mu=1}^M$, with $\boldsymbol{\xi}^\mu = (\xi_1^\mu, \dots, \xi_N^\mu)$ the normal vector of the μ-hyperplane. Its volume can formally be written as

$$V = \left\{ \boldsymbol{x} \in D : \sum_{i=1}^{N} \xi_i^\mu x_i \geq \gamma^\mu, \mu = 1, \dots, M \right\}.$$

From all possible questions related to polytopes in H-representation, we are particularly interested in that of the volume V and its projection onto each axis, or in order words, the single-site marginal pdf $P_i(x_i)$. These two quantities can be mathematically written as

$$V = \int_D \mathrm{d}\boldsymbol{x} \prod_{\mu=1}^{M} \Theta\left[h_\mu(x_{\partial\mu})\right], \qquad P_i(x_i) = \frac{1}{V} \int_{D_{\setminus i}} \mathrm{d}\boldsymbol{x}_{\setminus i} \prod_{\mu=1}^{M} \Theta\left[h_\mu(x_{\partial\mu})\right], \quad (4)$$

J. Stat. Mech. (2012) P11003

where we have defined $h_\mu(x_{\partial\mu}) = \sum_{i\in\partial\mu}\xi_i^\mu x_i - \gamma^\mu$, $\Theta(x)$ is the Heaviside step function, and $\boldsymbol{x}_{\setminus i}$ denotes the vector $\boldsymbol{x}$ without the component x_i, and $D_{\setminus i} = D_1\times\cdots\times D_{i-1}\times D_{i+1}\times\cdots\times D_N$. Notice that, in a strict mathematical sense, the volume of the polytope V defined by equation (4) is always strictly 0, as this is the volume of an $N-M$-dimensional manifold in N dimensions. This can be formally dealt with as explained in [13], but with the definition used here this does not have an impact on the results concerning the marginal distributions.

3. Single-site marginals, cavity equations and supports

Using the cavity method (see appendix A) it is possible to express the single-site marginals $P_i(x_i)$ in terms of the local variables connected to them. Assuming that the bipartite graph associated with the rectangular matrix $\boldsymbol{\xi}$ is locally tree-like, we can arrive at the following set of cavity equations:

$$P_i^{(\nu)}(x_i) = \frac{1}{V_i^{(\nu)}}\prod_{\mu\in\partial i\setminus\nu} m_\mu^{(i)}(x_i), \qquad \forall\, i,\nu\in\partial i, \tag{5}$$

$$m_\mu^{(i)}(x_i) = \frac{1}{m_\mu^{(i)}}\int_{D_{\partial\mu\setminus i}} \mathrm{d}x_{\partial\mu\setminus i}\,\Theta\left(h_\mu(x_{\partial\mu\setminus i}) + \xi_i^\mu x_i\right)\prod_{\ell\in\partial\mu\setminus i} P_\ell^{(\mu)}(x_\ell), \qquad \forall\, i,\mu\in\partial i. \tag{6}$$

Here the $V_i^{(\nu)}$ and $m_\mu^{(i)}$ are normalizing constants and we have labelled the variable-nodes with italic indices, $i, j, \ldots$, and the factor-nodes with Greek ones, $\mu,\nu,\ldots$. Besides that, we have used the following standard notation: ∂i denotes the set of factor-nodes neighbouring i, $\partial\mu$ denotes the set of variable-nodes neighbouring the factor-node μ, and $x_{\partial\mu}$ is the set of dynamical variables on the set of variable-nodes $\partial\mu$. Finally, if A is a set of indices and $i\in A$, then $A\setminus i$ is the set A without the index i.

Once the set of equations (5) and (6) is solved, the actual single-site marginals are given in terms of the marginals $\{m_\mu^{(i)}(x_i)\}$:

$$P_i(x_i) = \frac{1}{V_i}\prod_{\mu\in\partial i} m_\mu^{(i)}(x_i), \tag{7}$$

with V_i a normalizing constant. It is important to point out that the support of the marginal $P_i(x_i)$ is not the integration domain for $x_i\in D_i$, but rather the result of intersecting the polytope and D, the intersection then being projected onto the x_i-axis. Looking at equations (5) and (6), one notices that it is possible to write down self-consistency equations for the support of the marginals in the cavity equations. To do so, let us denote as $R_\mu^{(i)}$ and $K_i^{(\nu)}$ the supports of the marginals $m_\mu^{(i)}(x_i)$ and $P_i^{(\nu)}(x_i)$, respectively. Then from equation (5) we can write that

$$K_i^{(\nu)} = \bigcap_{\mu\in\partial i\setminus\nu} R_\mu^{(i)}. \tag{8}$$

To be able to write down the support $R_\mu^{(i)}$ in terms of $K_i^{(\nu)}$, we notice from equation (6) that, geometrically, we are integrating on a parallelotope $S_\mu^{(i)}\subseteq D_{\partial\mu\setminus i}$ defined as the Cartesian product of supports $\{K_\ell^{(\mu)}\}_{\ell\in\partial\mu\setminus i}$, that is $S_\mu^{(i)} = \times_{\ell\in\partial\mu\setminus i}K_\ell^{(\mu)}$. This parallelotope has $2^{|\partial\mu\setminus i|}$ vertices, whose set we denote as $\mathcal{V}_\mu^{(i)}$. As geometrically suggested by equation (6),

J. Stat. Mech. (2012) P11003

when x_i is varied the hyperplane $h_\mu(x_{\partial\mu\backslash i}) + \xi_i^\mu x_i = 0$ will intersect with all these vertices. Let us denote as $\mathcal{I}(\mathcal{V}_\mu^{(i)})$ the set of $2^{|\partial\mu\backslash i|}$ values of x_i at which this intersection occurs. Then the support $R_\mu^{(i)}$ is the maximal set such that

$$R_\mu^{(i)} = \left[\min_{x\in\mathcal{I}} \mathcal{I}(\mathcal{V}_\mu^{(i)}), \max_{x\in\mathcal{I}} \mathcal{I}(\mathcal{V}_\mu^{(i)})\right] \bigcap R_\mu^{(i)}. \tag{9}$$

Notice that once the collection of supports $R_\mu^{(i)}$ are known, then supports K_i for the single-site marginals $P_i(x_i)$ are given by

$$K_i = \bigcap_{\mu\in\partial i} R_\mu^{(i)}. \tag{10}$$

The set of equations (8) and (9) is decoupled from the set of the cavity equations (5) and (6), in the sense that the actual shape of the marginals is not needed to obtain information solely about their support. This rather simple observation is quite relevant for two reasons. Firstly, to apply the algorithms that we devise in section 4 we need to know the support of the marginals beforehand in order to avoid rejection. This will become clearer in the explanation of the method, but in anticipation and with a modest amount of foresight we note that by simply inspecting equation (8) we find that either x_i lies within the intersection of all the supports $\{R_\mu^{(i)}\}_{\mu\in\partial i\backslash\nu}$ or (at least) one of the m-functions is zero for that value, and therefore the value x_i should not be allowed. Secondly, as we will see in the derivation below, the self-consistency equations for the supports are much simpler than the corresponding equations for the marginals, meaning that they can be solved extremely quickly. Thus, as the supports give valuable information on allowed values for the dynamical variables, these self-consistency equations are important in their own right, as they can be applied, for instance, to study the behaviour of metabolic networks under perturbations (e.g. the effect of gene knockout in reaction rates) or, as we illustrate below in section 6.2, to determine the critical line for the global growth rate in von Neumann's model described in equation (3).

3.1. Expressing the self-consistency equations for the supports in terms of endpoints

We now move on to implement the set of (8) and (9) to obtain each support $K_i^{(\mu)}$. As the marginals are the result of projecting a convex polytope, we expect them to be singly supported. We then define $K_i^{(\mu)} = [k_{i,-}^{(\mu)}, k_{i,+}^{(\mu)}]$ and similarly $R_\mu^{(i)} = [r_{\mu,-}^{(i)}, r_{\mu,+}^{(i)}]$. From the Heaviside function in equation (6) we know that $h_\mu(x_{\partial\mu\backslash i}) + \xi_i^\mu x_i \geq 0$. This implies that for an auxiliary variable z_μ we can write $h_\mu(x_{\partial\mu\backslash i}) + \xi_i^\mu x_i = z_\mu$ or $x_i = (1/\xi_i^\mu)[z_\mu - h_\mu(x_{\partial\mu\backslash i})]$. If the reader finds the introduction of the variable z_μ rather obscure, an alternative interpretation will be given in section 4. Next, let us now denote as $x^i_{\mu,\sigma_{\partial\mu\backslash i}}$ the $2^{|\partial\mu\backslash i|}$ values of x_i at each of the vertices of the parallelotope $S_\mu^{(i)} = \times_{\ell\in\partial\mu\backslash i} K_\ell^{(\mu)}$, that is

$$x^{(i)}_{\mu,\sigma_{\partial\mu\backslash i}} = \frac{1}{\xi_i^\mu} y^{(i)}_{\mu,\sigma_{\partial\mu\backslash i}}, \qquad y^{(i)}_{\mu,\sigma_{\partial\mu\backslash i}} = z_\mu + \gamma^\mu - \sum_{\ell\in\partial\mu\backslash i} \xi_\ell^\mu k^{(\mu)}_{\ell,\sigma_\ell}.$$

To look for the maximum and minimum in the set of values $\{x^{(i)}_{\mu,\sigma_{\partial\mu\backslash i}}\}$, we notice that we do not need to check all $2^{|\partial\mu\backslash i|}$ values, but rather we can write down directly an expression

J. Stat. Mech. (2012) P11003

for them. Indeed, let us define first

$$t^{(i)}_{\mu,-} = z_{\min} + \gamma^{\mu} - \sum_{\ell\in\partial\mu\setminus i} \max\{\xi^{\mu}_{\ell} k^{(\mu)}_{\ell,\sigma_\ell}\}_{\sigma_\ell\in\{-,+\}},$$
$$t^{(i)}_{\mu,+} = z_{\max} + \gamma^{\mu} - \sum_{\ell\in\partial\mu\setminus i} \min\{\xi^{\mu}_{\ell} k^{(\mu)}_{\ell,\sigma_\ell}\}_{\sigma_\ell\in\{-,+\}} \tag{11}$$

where $z_{\min} = 0$ and $z_{\max}$ are the minimum and maximum values that the variable z_μ can take[5]. Then the maximum and minimum values of x_i are

$$o^{(i)}_{\mu,+} = \max\{t^{(i)}_{\mu,\sigma}/\xi^{\mu}_{i}\}_{\sigma\in\{-,+\}}, \qquad o^{(i)}_{\mu,-} = \min\{t^{(i)}_{\mu,\sigma}/\xi^{\mu}_{i}\}_{\sigma\in\{-,+\}}. \tag{12}$$

From here we can have finally write

$$r^{(i)}_{\mu,-} = \max\{o^{(i)}_{\mu,-}, r^{(i)}_{\mu,-}\}, \qquad r^{(i)}_{\mu,+} = \min\{o^{(i)}_{\mu,+}, r^{(i)}_{\mu,+}\}$$
$$k^{(\nu)}_{i,-} = \max\{r^{(i)}_{\mu,-}\}_{\mu\in\partial i\setminus\nu}, \qquad k^{(\nu)}_{i,+} = \min\{r^{(i)}_{\mu,+}\}_{\mu\in\partial i\setminus\nu}. \tag{13}$$

The set of equations (11)–(13) is a set of self-consistency cavity equations for the endpoints defining the supports of the cavity marginals. These cavity equations can be solved in the standard manner by using belief propagation, i.e. the fixed-point iteration method, using as a starting support $K^{(\mu)}_i = D_i = [m_{i,-}, m_{i,+}]$ for all $i = 1, \ldots, N$ and for all $\mu \in \partial i$. From equation (10) and once the values of $\{r^{(i)}_{\mu,-}\}$ have been estimated, we can calculate the support K_i of the single-site marginals $P_i(x_i)$, namely

$$k_{i,-} = \max\{r^{(i)}_{\mu,-}\}_{\mu\in\partial i}, \qquad k_{i,+} = \min\{r^{(i)}_{\mu,+}\}_{\mu\in\partial i}.$$

4. A weighted belief-propagation algorithm

Before presenting our method let us discuss briefly the various alternatives found in the literature as regards how to estimate volume-related properties of polytopes. They can be grouped into two main categories: (i) Monte Carlo simulations and (ii) theoretical approaches.

By Monte Carlo simulations we generally mean numerical ways of directly sampling—one hopes, uniformly—the volume of a polytope. The Hit-and-Run algorithm [4], explained in appendix B, is a Monte Carlo method that uniformly samples the volume of a polytope. This method is reasonably efficient only for small dimensions as its mixing time goes as $\mathcal{O}(N^3)$. The MinOver$^+$ algorithm [14] was originally introduced for neural networks to check whether a solution to a problem of the type (2) exists. This algorithm was subsequently applied to sample the volume of polytopes in [7], [9]–[11]. Compared with the Hit-and-Run algorithm, MinOver$^+$ can deal with larger systems, yielding samples that become more uniform as the system becomes larger.

By theoretical approaches what we mean is that one identifies the quantities of interest and under not unreasonable assumptions one then tries to write down closed equations for these quantities. This is what we have done here thus far (and also in [7]), by taking as quantities of interest the single-site marginals, and finding the set of closed equations (5) and (6) by assuming the Bethe approximation for the problem (2). For FBA, that is for

[5] Since the polytope is restricted to being at most D, then z_μ does indeed have a finite maximum value. For practical purposes one simply sets a cut-off maximum value.

J. Stat. Mech. (2012) P11003

problems of the type (1), this was done in e.g. [13], while in the context of compressed sensing, similar equations were derived in [3, 12]. Of course the problem is then how to solve the resulting equations (e.g. the set of equations (5) and (6) in our case) in an efficient way.

The reader should note that in the simplest possible analysis, we expect the computational time associated with solving the cavity equations to go linearly with the system size N (as opposed to $\sim N^3$ for the Hit-and-Run algorithm), as this is precisely how the number of cavity equations grows with N. Leaving this aside there are, however, more urgent and pressing matters. Indeed, in general terms it is considered a daunting task trying to solve the exact cavity equations by any numerical means when the dynamical variables are continuous, as they are in our case, in the cases of [3, 12, 13] or, more generally, for any continuous-valued spin models on diluted graphs. To overcome this numerical burden, one first approximates the cavity equations somehow and then solves numerically the approximated equations. This can be presented in various ways, but the general approach is first noticing that the marginals can be parametrized using an infinite number of parameters, that is, $P_i^{(\nu)}(x_i|\boldsymbol{\alpha}_i^{(\nu)})$ and $m_\mu^{(i)}(x_i|\boldsymbol{\beta}_\mu^{(i)})$ where we have defined $\boldsymbol{\alpha}_i^{(\nu)} = (\alpha_{i,1}^{(\nu)}, \alpha_{i,2}^{(\nu)}, \ldots)$ and $\boldsymbol{\beta}_\mu^{(i)} = (\beta_{\mu,1}^{(i)}, \beta_{\mu,2}^{(i)}, \ldots)$. For instance, in this framework the Gaussian approximation means choosing the parameters of the marginals to be their cumulants and assuming that only the first two cumulants are different from zero.

However, for this type of problem consisting in counting the number of solutions of a set of either linear equalities or inequalities, as for our case, it is possible to tackle the cavity equations (5) and (6) without approximation. To whet the appetite for the forthcoming discussion, let us first rewrite equation (6) as follows:

$$m_\mu^{(i)}(x_i) = \frac{1}{m_\mu^{(i)}} \int_{\mathbb{R}^+} \mathrm{d}z_\mu \int_{D_{\partial\mu\setminus i}} \mathrm{d}x_{\partial\mu\setminus i}\, \delta(h_\mu(x_{\partial\mu\setminus i}) + \xi_i^\mu x_i - z_\mu) \prod_{\ell\in\partial\mu\setminus i} P_\ell^{(\mu)}(x_\ell), \qquad \forall\, i,\mu \in \partial i \tag{14}$$

where we have expressed the Theta functions in terms of a Dirac delta using the integral identity $\theta(x-a) = \int_{\mathbb{R}^+} \mathrm{d}y\, \delta[(x-a)-y]$. Note that the new integration variable z_μ can be understood as a random variable with uniform density in a subset of $\mathbb{R}^+$. Indeed, due to the constraints of the problem at hand, the random variable z_μ will generally take values not on the whole positive line but rather on a compact set of it, and it will be therefore normalizable.

Looking at the cavity equation (14) we notice that, written in this way, the Dirac delta suggests estimating the integral expression by using the method of population dynamics (note: population dynamics provides a clever way to estimate the integral using a Monte Carlo method when the integrand is unknown) as in, for instance, solving the ensemble cavity equations for discrete spin models on locally tree-like graphs. There are two important differences, though: (i) the cavity equation (14) is still in the instance; (ii) not all updates $x_i \leftarrow (z_\mu - h_\mu(x_{\partial\mu\setminus i}))/\xi_i^\mu$ suggested by equation (14) are allowed, as there may be integration regions such that $x_i \notin R_\mu^{(i)}$ and therefore the update must be rejected. The first difference is unimportant, but the second one is not: if we do not find a way to avoid the rejection region, the resulting population dynamics performed in the instance will be quite inefficient.

J. Stat. Mech. (2012) P11003

Fortunately, there is a way to avoid the rejection region completely. To illustrate how, we set aside all notational complications of the cavity equations and focus on the problem at hand with the following simpler example:

$$m(x) = \frac{1}{m}\int_{[0,1]^n}\left[\prod_{i=1}^{n} \mathrm{d}y_i\,\rho_i(y_i)\right]\delta\left(x+\sum_{i=1}^{n} a_i y_i - \gamma\right), \qquad x \geq 0$$

where the $\rho_i(y_i)$ for $i = 1, \ldots, n$ are some arbitrary pdfs with $y_i \in [0, 1]$. We note that this integral is zero unless $x = \gamma - \sum_{i=1}^{n} a_i y_i \geq 0$. Thus the effective integration region is the one enclosed by the hypercube $[0, 1]^n$ and the hyperplane $\gamma = \sum_{i=1}^{n} a_i y_i$. Deciding to carry out the integration in a specific order we write

$$m(x) = \frac{1}{m}\int_{\mathcal{R}_1(\gamma)} \mathrm{d}y_1\,\rho_1(y_1)\int_{\mathcal{R}_2(y_1,\gamma)} \mathrm{d}y_2\,\rho_2(y_2)\cdots \times \int_{\mathcal{R}_n(y_1,\ldots,y_{n-1},\gamma)} \mathrm{d}y_n\,\rho_n(y_n)\delta\left(x+\sum_{i=1}^{n} a_i y_i - \gamma\right).$$

As we want to evaluate the above integral by Monte Carlo methods, we need to reweight each density ρ_i on the new region $\mathcal{R}_i(y_1, \ldots, y_{i-1}, \gamma)$. We write

$$\rho_i(y_i|y_1, \ldots, y_{i-1}, \gamma) = \frac{\rho_i(y_i)}{w_i(y_1, \ldots, y_{i-1}, \gamma)},$$
$$w_i(y_1, \ldots, y_{i-1}, \gamma) \equiv \int_{\mathcal{R}_i(y_1,\ldots,y_{i-1},\gamma)} \mathrm{d}y_i\,\rho_i(y_i).$$

Thus the expression for $m(x)$ becomes

$$m(x) = \frac{1}{m}\left[\prod_{i=1}^{n}\int_{\mathcal{R}_i(y_1,\ldots,y_{i-1},\gamma)} \mathrm{d}y_i\,\rho_i(y_i|y_1, \ldots, y_{i-1}, \gamma)\right]\left[\prod_{i=1}^{n} w_i(y_1, \ldots, y_{i-1}, \gamma)\right] \times \delta\left(x+\sum_{i=1}^{n} a_i y_i - \gamma\right).$$

In this new form, the integral $m(x)$ can be evaluated by Monte Carlo methods without rejection. This is done as follows:

(1) draw y_1 according to $\rho_1(y_1|\gamma)$, draw y_2 according to $\rho_2(y_2|y_1, \gamma)$, etc, and finally draw y_n according to $\rho_n(y_n|y_1, \ldots, y_{n-1}, \gamma)$;

(2) assign $x \leftarrow \gamma - \sum_{i=1}^{n} a_i y_i$ with weight $\Omega \equiv \prod_{i=1}^{n} w_i(y_1, \ldots, y_{i-1}, \gamma)$.

This process can be then repeated $\mathcal{N}_p$ times to obtain a collection of pairs $\{(x_\alpha, \Omega_\alpha)\}_{\alpha=1,\ldots,\mathcal{N}_p}$, from which we can reconstruct $m(x)$ as $m(x) \approx 1/m\sum_{\alpha=1}^{\mathcal{N}_p}\Omega_\alpha\delta(x - x_\alpha)$ with $m = \sum_{\alpha=1}^{\mathcal{N}_p}\Omega_\alpha$. This reweighting technique avoids the rejection region, allowing us to estimate the integrals involved much more efficiently.

J. Stat. Mech. (2012) P11003

Thus, if we apply this reweighting technique to cavity equations (14), we obtain

$$m_\mu^{(i)}(x_i) = \frac{1}{m_\mu^{(i)}} \left[\prod_{j=1}^{k_\mu^{(i)}} \int_{\mathcal{R}_j^{(\mu)}(x_{\ell_1},\ldots,x_{\ell_{j-1}},\gamma^\mu)} \mathrm{d}x_{\ell_j}\, P_{\ell_j}^{(\mu)}(x_{\ell_j}|x_{\ell_1},\ldots,x_{\ell_{j-1}},\gamma^\mu) \right] \times \int_{\mathcal{R}_j^{(\mu)}(x_{\partial\mu\setminus i},\gamma^\mu)} \frac{\mathrm{d}z_\mu}{\omega^{(\mu)}(x_{\partial\mu\setminus i},\gamma^\mu)} \delta\left(h_\mu(x_{\partial\mu\setminus i}) + \xi_i^\mu x_i - z_\mu\right) \times \left[\prod_{j=1}^{k_\mu^{(i)}} \omega_j^{(\mu)}(x_{\ell_1},\ldots,x_{\ell_{j-1}},\gamma^\mu) \right] \omega^{(\mu)}(x_{\partial\mu\setminus i},\gamma^\mu), \qquad \forall\, i,\ \mu\in\partial i \tag{15}$$

where we have used the notation $k_\mu^{(i)} = |\partial\mu \setminus i|$ and $\ell_j \in \partial\mu\setminus i$ for $j = 1,\ldots,k_\mu^{(i)}$ and we have defined

$$P_{\ell_j}^{(\mu)}(x_{\ell_j}|x_{\ell_1},\ldots,x_{\ell_{j-1}},\gamma^\mu) = \frac{P_{\ell_j}^{(\mu)}(x_{\ell_j})}{\omega_j^{(\mu)}(x_{\ell_1},\ldots,x_{\ell_{j-1}},\gamma^\mu)},$$

$$\omega_j^{(\mu)}(x_{\ell_1},\ldots,x_{\ell_{j-1}},\gamma^\mu) = \int_{\mathcal{R}_j^{(\mu)}(x_{\ell_1},\ldots,x_{\ell_{j-1}},\gamma^\mu)} \mathrm{d}x_{\ell_j}\, P_{\ell_j}^{(\mu)}(x_{\ell_j})$$

$$\omega^{(\mu)}(x_{\partial\mu\setminus i},\gamma^\mu) = \int_{\mathcal{R}_j^{(\mu)}(x_{\partial\mu\setminus i},\gamma^\mu)} \mathrm{d}z_\mu = |\mathcal{R}_j^{(\mu)}(x_{\partial\mu\setminus i},\gamma^\mu)|.$$

Although we could have chosen any order of integration for writing down equation (15), it is convenient, as we will explain below, to integrate the variable z_μ first[6]. We will henceforth generally call the set of equations (15) and (5) *the weighted cavity equations* and the algorithm used to solve them the *weighted belief-propagation algorithm* or the *weighted message-passing algorithm*.

5. Implementing the weighted belief-propagation algorithm

We move on to discussing the actual implementation of the algorithm to numerically solve the set of equations (5) and (15). As we will see shortly, although we have found a nice method for avoiding rejection, further complications appear on the horizon which may put into question the discussion that we have given thus far. Fortunately, we can describe a way to overcome them.

We present here two alternative implementations: *the method of histograms* and *the weighted population dynamics in the instance*.

5.1. The method of histograms

Ideally, to solve numerically the weighted cavity equations we would firstly represent the marginals $P_i^{(\mu)}(x_i)$ and $m_\mu^{(i)}(x_i)$ with populations of pairs $\{(w_{i,\alpha}^{(\mu)}, x_{i,\alpha}^{(\mu)})\}_{\alpha=1,\ldots,\mathcal{N}_p}$ and $\{(v_{i,\alpha}^{(\mu)}, x_{\mu,\alpha}^{(i)})\}_{\alpha=1,\ldots,\mathcal{N}_p}$, respectively, so that $P_i^{(\mu)}(x_i) \approx (1/P_i^{(\mu)})\sum_{\alpha=1}^{\mathcal{N}_p} w_{i,\alpha}^{(\mu)}\delta(x_i - x_{i,\alpha}^{(\mu)})$ and $m_\mu^{(i)}(x_i) \approx (1/m_\mu^{(i)})\sum_{\alpha=1}^{\mathcal{N}_p} v_{\mu,\alpha}^{(i)}\delta(x_i - x_{\mu,\alpha}^{(i)})$, where $P_i^{(\mu)}$ and $m_\mu^{(i)}$ are normalization factors.

[6] Note that while z_μ is the variable to formally be integrated first, within our Monte Carlo method, z_μ is the last variable to be estimated.

J. Stat. Mech. (2012) P11003

There is a drawback though: while it is possible—and desirable—to use the population of the Ps to evaluate equation (15) updating the population for the ms, it is difficult to see how to use this representation directly to update the population for the ms according to equation (5). Of course, one could always construct histograms for the set of functions $\{m_\mu^{(i)}(x_i)\}_{\mu\in\partial i/\nu}$ to calculate a histogram for $P_i^{(\nu)}(x_i)$ according to equation (5), and then draw a population for $P_i^{(\nu)}(x_i)$ from its histogram. But if we need to construct histograms at some point in the algorithm, it is actually a waste of time to go back and forth between a representation of histograms and a representation of populations.

Thus, due to equation (5), we are unfortunately obliged to implement a weighted message-passing algorithm where the functions $P_i^{(\mu)}(x_i)$ and $m_\mu^{(i)}(x_i)$ are quite simply represented by histograms. We call this implementation *the method of histograms*.

Discretization of the unweighted cavity equations and their Fourier transforms constituted the method used in [12, 13] in the context of FBA and compressive sensing. Let us see how it is possible to avoid discretization altogether.

5.2. Weighted population dynamics in the instance

Fortunately, it is possible to overcome the previous problem of having to go through the m-functions via equation (5). To illustrate the method we again ignore all tediousness regarding notation and analyse the following simpler example that captures the complications arising from combining equations (5) and (15), namely, the appearance of multiple Dirac deltas:

$$r(x) = \frac{1}{r}\int_{[0,1]^n}\left[\prod_{i=1}^n \mathrm{d}y_i\,\rho_i(y_i)\right]\delta\left(x+\sum_{i=1}^n a_i y_i-\gamma\right)\times\int_{[0,1]^m}\left[\prod_{i=1}^m \mathrm{d}z_i\,\psi_i(z_i)\right]\delta\left(x+\sum_{i=1}^m b_i z_i-\delta\right),$$

with $x\geq 0$ and where $\rho_i(y_i)$, $i=1,\ldots,n$, and $\psi_i(z_i)$, $i=1,\ldots,m$, are some arbitrary pdfs with $y_i, z_i\in[0,1]$. As we have already discussed the problem of rejection, we first simply reweight this equation accordingly:

$$\begin{aligned} r(x) = \frac{1}{r}&\left[\prod_{i=1}^n\int_{\mathcal{R}_i^{(\rho)}(y_1,\ldots,y_{i-1},\gamma)}\mathrm{d}y_i\,\rho_i(y_i|y_1,\ldots,y_{i-1},\gamma)\right]\\ &\times\left[\prod_{i=1}^n w_i^{(\rho)}(y_1,\ldots,y_{i-1},\gamma)\right]\delta\left(x+\sum_{i=1}^n a_i y_i-\gamma\right)\\ &\times\left[\prod_{i=1}^m\int_{\mathcal{R}_i^{(\psi)}(z_1,\ldots,z_{i-1},\gamma)}\mathrm{d}z_i\,\psi_i(z_i|z_1,\ldots,z_{i-1},\delta)\right]\\ &\times\left[\prod_{i=1}^m w_i^{(\psi)}(z_1,\ldots,z_{i-1},\delta)\right]\delta\left(x+\sum_{i=1}^m b_i z_i-\delta\right). \end{aligned} \tag{16}$$

As there are two Dirac deltas in the expression (16), it is natural to ask how to resolve the issue of assignment for the variable x so that we can estimate the integrals by Monte

J. Stat. Mech. (2012) P11003

Carlo methods. We describe two options: *random assignment and variable locking* and *variable fixing*.

5.2.1. Random assignment and variable locking. In this case we use one of the Dirac deltas to assign a value to x, locking such values in the other Dirac deltas. Suppose, for instance, that in our example we use the first Dirac delta to assign a value to x. The first steps of the algorithm would be almost as before:

(1) draw y_1 according to $\rho_1(y_1|\gamma)$, draw y_2 according to $\rho_2(y_2|y_1,\gamma)$, etc, and finally draw y_n according to $\rho_n(y_n|y_1,\ldots,y_{n-1},\gamma)$;
(2) assign $x \leftarrow x^\star = \gamma - \sum_{i=1}^n a_i y_i$;
(3) calculate the weight $\Omega \equiv \prod_{i=1}^n w_i^{(\rho)}(y_1,\ldots,y_{i-1},\gamma)$.

In the second Dirac delta, the value of x has been locked to $x^\star$. The integration regions will depend on $x^\star$ and we denote this by writing $\mathcal{R}_i^{(\psi)}(z_1,\ldots,z_{i-1},\gamma,x^\star)$. The multiple integration over $\boldsymbol{z}$ is then done as follows:

(1) draw z_1 according to $\psi_1(z_1|\delta,x^\star)$, draw z_2 according to $\psi_2(z_2|z_1,\delta,x^\star)$, etc, and finally draw z_{m-1} according to $\psi_{m-1}(z_{m-1}|z_1,\ldots,z_{m-2},\delta,x^\star)$;
(2) the Dirac delta locks the value $z_m \leftarrow (\delta - x^\star - \sum_{i=1}^{m-1} b_i z_i)/b_m$ allowing us to integrate the last integral over z_m;
(3) calculate the weight $\Gamma \equiv \psi_m(z_m|z_1,\ldots,z_{m-1},\delta,x^\star)\prod_{i=1}^m w_i^{(\psi)}(z_1,\ldots,z_{i-1},\delta,x^\star) = \psi_m(z_m)\prod_{i=1}^{m-1} w_i^{(\psi)}(z_1,\ldots,z_{i-1},\delta,x^\star)$;
(4) assign $x^\star$ an overall weight $\Delta \equiv \Omega\Gamma$.

As before this process can be repeated $\mathcal{N}_p$ times to obtain a collection of pairs $\{(x_\alpha^\star,\Delta_\alpha)\}_{\alpha=1,\ldots,\mathcal{N}_p}$, from which we can reconstruct $r(x)$ as $r(x) \approx 1/r\sum_{\alpha=1}^{\mathcal{N}_p}\Delta_\alpha\delta(x-x_\alpha^\star)$ with $r = \sum_{\alpha=1}^{\mathcal{N}_p}\Delta_\alpha$.

5.2.2. Variable fixing. In this case we fix the value of x from the beginning, then evaluate each integral according to the second part of the algorithm before. For our example at hand the steps are:

(1) fix a value of x to $x^\star$ within its support for $r(x)$;
(2) for the integrals involving function ρ:
   (a) draw y_1 according to $\rho_1(y_1|\gamma,x^\star)$, draw y_2 according to $\rho_2(y_2|y_1,\gamma,x^\star)$, etc, and finally draw y_{n-1} according to $\rho_{n-1}(y_{n-1}|y_1,\ldots,y_{n-2},\gamma,x^\star)$;
   (b) the Dirac delta locks the value $y_n \leftarrow (\delta - x - \sum_{i=1}^{n-1} a_i z_i)/a_n$ allowing us to integrate the last integral over z_n;
   (c) calculate the weight $\Gamma_\rho \equiv \psi_n(y_n|y_1,\ldots,y_{n-1},\gamma,x^\star)\prod_{i=1}^n w_i^{(\rho)}(y_1,\ldots,y_{i-1},\gamma,x^\star) = \rho_n(y_n)\prod_{i=1}^{n-1} w_i^{(\rho)}(y_1,\ldots,y_{i-1},\gamma,x^\star)$;
(3) repeat the same process for function ψ and calculate the weight Γ_ψ;
(4) assign a weight $\Delta \equiv \Gamma_\rho\Gamma_\psi$ to the fixed value $x^\star$.

This process can be repeated $\mathcal{N}_p$ times, scanning the support of $r(x)$ to obtain a collection of pairs $\{(x_\alpha,\Delta_\alpha)\}_{\alpha=1,\ldots,\mathcal{N}_p}$, from which we can reconstruct $r(x)$ as $r(x) \approx (1/r)\sum_{\alpha=1}^{\mathcal{N}_p}\Delta_\alpha\delta(x-x_\alpha)$ with $r = \sum_{\alpha=1}^{\mathcal{N}_p}\Delta_\alpha$.

J. Stat. Mech. (2012) P11003

This implementation has two clear advantages with respect the previous one. The first one is that since we are not using the Dirac delta to randomly assign a value of x, this can be used to perform one less integration. For this reason, we have found it more convenient to arrange the order of integration in equation (15) such that the integral over z_μ is the last one to be estimated by the algorithm, and so the contributing weight is unity. The second advantage lies in the fact that at a fixed value of x, what we should obtain is an averaged value of the weight. The weight is itself a random variable whose variance, even in simple cases, can be rather large. Thus it is appropriate to refine the estimates of the weights by replacing them by averaged values. To do this we simply calculate T estimates for each weight $\{\Gamma_{\rho,t}\}_{t=1,\ldots,T}$ and $\{\Gamma_{\psi}\}_{t=1,\ldots,T}$, calculate the average weights $\overline{\Gamma_\rho} = 1/T\sum_{t=1}^{T}\Gamma_{\rho,t}$ and $\overline{\Gamma_\psi} = 1/T\sum_{t=1}^{T}\Gamma_{\psi,t}$, and assign instead the weight $\overline{\Gamma_\rho}\,\overline{\Gamma_\psi}$ to the fixed value $x^\star$.[7]

5.2.3. Implementation for our problem. Considering all the points discussed previously, while hoping that the tedious but necessary notation that we are about to use does not distract the reader, the core of the algorithm for the weighted population dynamics for solving the cavity equations (5) and (15) is as follows. We assume that all supports $K_i^{(\mu)}$ for each marginal $P_i^{(\mu)}(x_i)$ are known and we represent each marginal by a population of pairs $\{(w_{i,\alpha}^{(\mu)}, x_{i,\alpha}^{(\mu)})\}_{\alpha=1,\ldots,\mathcal{N}_p}$. Then for the *random assignment and variable locking* the essential steps are the following:

(1) choose a variable-node i, a factor-node $\nu \in \partial i$, and a value $x_i \in K_i^{(\nu)}$;
(2) chose a factor-node $\mu_0 \in \partial i/\nu$;
   (a) draw x_{ℓ_1} with probability $P_{\ell_1}^{(\mu_0)}(x_{\ell_1}|\gamma^{\mu_0})$, draw x_{ℓ_2} with probability $P_{\ell_2}^{(\mu_0)}(x_{\ell_2}|x_{\ell_1},\gamma^{\mu_0})$, etc, draw $x_{\ell_{k_{\mu_0}^{(i)}}}$ with probability $P_{\ell_{k_{\mu_0}^{(i)}}}^{(\mu_0)}(x_{\ell_1}|x_{\ell_1},\ldots,x_{\ell_{k_{\mu_0}^{(i)}-1}},\gamma^{\mu_0})$, and draw a uniform random variable z^μ in the segment $\mathcal{R}_j^{(\mu_0)}(x_{\partial\mu_0\setminus i},\gamma^{\mu_0})$;
   (b) assign $x_i \leftarrow [z_{\mu_0} - h_\mu(x_{\partial\mu_0\setminus i})]/\xi_i^{\mu_0}$;
   (c) calculate $\Omega_{\mu_0}^{(i)} = \omega^{(\mu)}(x_{\partial\mu\setminus i},\gamma^\mu)\prod_{j=1}^{k_\mu^{(i)}}\omega_j^{(\mu)}(x_{\ell_1},\ldots,x_{\ell_{j-1}},x_i,\gamma^\mu)$;
(3) for all $\mu \in \partial i/(\nu\cup\mu_0)$:
   (a) draw x_{ℓ_1} with probability $P_{\ell_1}^{(\mu)}(x_{\ell_1}|x_i,\gamma^\mu)$, draw x_{ℓ_2} with probability $P_{\ell_2}^{(\mu)}(x_{\ell_2}|x_{\ell_1},x_i,\gamma^\mu)$, etc, draw $x_{\ell_{k_\mu^{(i)}}}$ with probability $P_{\ell_{k_\mu^{(i)}}}^{(\mu)}(x_{\ell_1}|x_{\ell_1},\ldots,x_{\ell_{k_\mu^{(i)}-1}},x_i,\gamma^\mu)$;
   (b) calculate $\Omega_\mu^{(i)} = \prod_{j=1}^{k_\mu^{(i)}}\omega_j^{(\mu)}(x_{\ell_1},\ldots,x_{\ell_{j-1}},x_i,\gamma^\mu)$;
(4) calculate $\Gamma_i^{(\nu)} = \prod_{\mu\in\partial i/\nu}\Omega_\mu^{(i)}$ and replace a pair of the population of the marginal $P_i^{(\nu)}(x_i)$ by the new pair $(\Gamma_i^{(\nu)}, x_i)$.

For the second implementation, that is *variable fixing*, we enumerate the following basic steps:

(1) choose a variable-node i, a factor-node $\nu \in \partial i$, and a value $x_i \in K_i^{(\nu)}$;

[7] Clearly, T does not have to be fixed, but is chosen so as to achieve a certain accuracy; nor does it have to be the same for all weights.

J. Stat. Mech. (2012) P11003

(2) for all $\mu \in \partial i/\nu$:

(a) for $t = 1, \ldots, T$:

(i) draw x_{ℓ_1} with probability $P^{(\mu)}_{\ell_1}(x_{\ell_1}|x_i, \gamma^\mu)$, draw x_{ℓ_2} with probability $P^{(\mu)}_{\ell_2}(x_{\ell_2}|x_{\ell_1}, x_i, \gamma^\mu)$, etc, draw $x_{\ell_{k^{(i)}_\mu}}$ with probability $P^{(\mu)}_{\ell_{k^{(i)}_\mu}}(x_{\ell_1}|x_{\ell_1}, \ldots, x_{\ell_{k^{(i)}_\mu - 1}}, x_i, \gamma^\mu)$;

(ii) calculate $w_t = \prod_{j=1}^{k^{(i)}_\mu} \omega^{(\mu)}_j(x_{\ell_1}, \ldots, x_{\ell_{j-1}}, x_i, \gamma^\mu)$;

(b) set $\Omega^{(i)}_\mu = \frac{1}{T}\sum_{t=1}^{T} w_t$;

(3) calculate $\Gamma^{(\nu)}_i = \prod_{\mu \in \partial i/\nu} \Omega^{(i)}_\mu$ and replace a pair of the population of the marginal $P^{(\nu)}_i(x_i)$ by the new pair $(\Gamma^{(\nu)}_i, x_i)$.

It is important to notice here that in the last step the replacement must be done according to how the value of x_i is selected in the first place; otherwise we could generate a biased sampling of the support $K^{(\nu)}_i$, that is, if in the first step x_i is chosen uniformly randomly, or fixed, then the replacement must be done in the same manner. Besides this, instead of replacing, other alternatives would be, for instance, increasing the populations in those regions where the sampling is rather poor, or using a non-uniform sampling taking more points wherever they are most needed.

6. Numerical tests

To illustrate the previously discussed results we apply them to some simple examples.

6.1. A toy example regarding supports of marginals

Consider the polyhedron described by the following set of inequalities: $\boldsymbol{\xi}_\epsilon \boldsymbol{x} \geq \boldsymbol{\gamma}$ with

$$\boldsymbol{\xi}_\epsilon = \begin{pmatrix} -1 & -1 & -1 \\ 1 & -\epsilon & 0 \\ 0 & 1 & -\epsilon \\ -\epsilon & 0 & 1 \end{pmatrix}, \qquad \boldsymbol{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \qquad \boldsymbol{\gamma} = \begin{pmatrix} -4 \\ 1-\epsilon \\ 1-\epsilon \\ 1-\epsilon \end{pmatrix}. \tag{17}$$

For $\epsilon = 0$ the graph associated with $\boldsymbol{\xi}_\epsilon$ is exactly a tree, as shown in figure 1, so iteration of the cavity equations for the support converges at one step. On the other hand, if $\epsilon \neq 0$ the graph is loopy and, depending on the value of ϵ, the estimates from the cavity equations can be very bad indeed. Yet for small ϵ one obtains very precise results. In figure 2 we plotted the results of iterating the cavity equations (11)–(13) for the supports. Due to symmetry of the problem, the support and marginals are the same for the three variables and we parametrize the support as $[a, b]$. We have taken a value of $\epsilon = 0.05$.

It is important to notice that as the dynamical variables are continuous we have at our disposal an ample set of transformations that we can perform on the system. This can be used to transform a very loopy network into a more tree-like one. For instance, in this

J. Stat. Mech. (2012) P11003

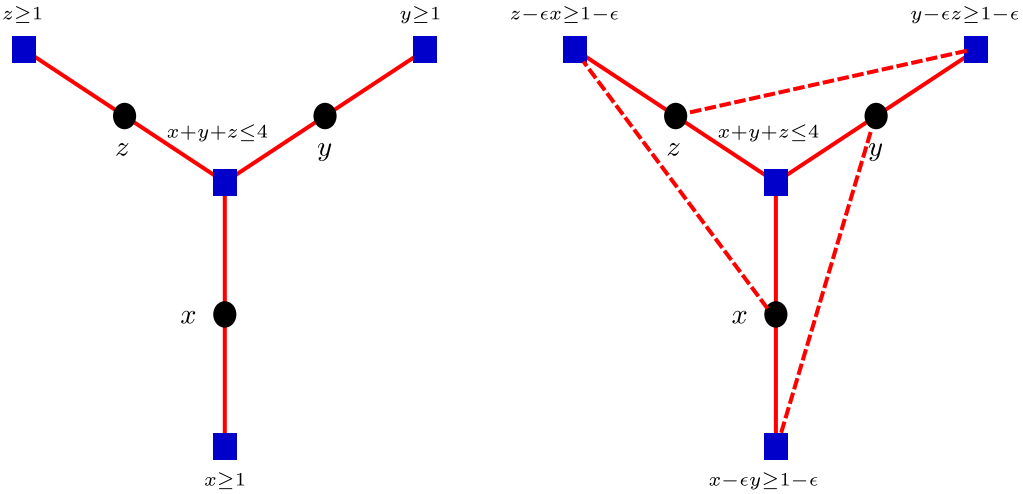

Figure 1. Bipartite graphs associated with the polyhedron defined in our example (17). The left one is exactly a tree corresponding to $\epsilon = 0$. The right one corresponds to the generic loopy case of $\epsilon \neq 0$.

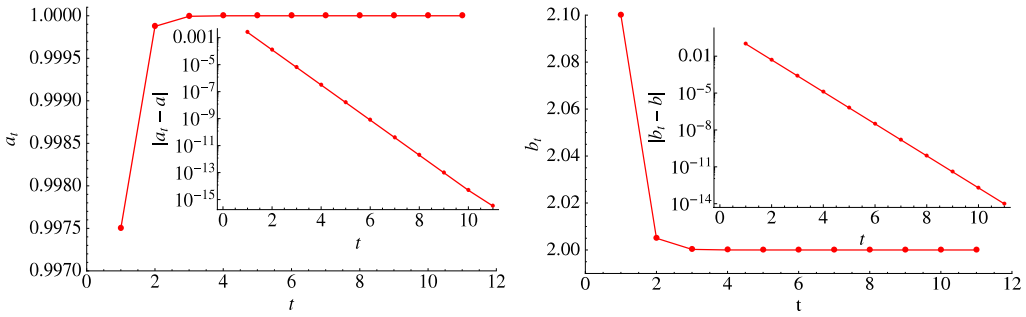

Figure 2. Results of iterating the cavity equations (11)–(13) for the supports. Due to the symmetry of the problem, the support and marginals are the same for the three variables and we parametrize the support as $[a, b]$. We have taken a value of $\epsilon = 0.05$. The left (right) figure corresponds to the evolution of the cavity equations for the lower endpoint a (b). The inset reports the absolute error between the value of the endpoint at iteration t and the exact value.

very simple example we notice that we can write $\boldsymbol{\xi}_\epsilon = \boldsymbol{\eta}_\epsilon \boldsymbol{T}_\epsilon$ with

$$\boldsymbol{\eta}_\epsilon = \begin{pmatrix} -\frac{1}{1-\epsilon} & -\frac{1}{1-\epsilon} & -\frac{1}{1-\epsilon} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad \boldsymbol{T}_\epsilon = \begin{pmatrix} 1 & -\epsilon & 0 \\ 0 & 1 & -\epsilon \\ -\epsilon & 0 & 1 \end{pmatrix}.$$

Then, the transformation $\boldsymbol{y} = \boldsymbol{T}_\epsilon \boldsymbol{x}$ takes the system into an exact tree. Of course, under which general conditions it is possible to perform these kinds of transformations is not clear to us just yet; nor it is clear which are the most appropriate transformations, whose inverse transformation can be performed efficiently.

J. Stat. Mech. (2012) P11003

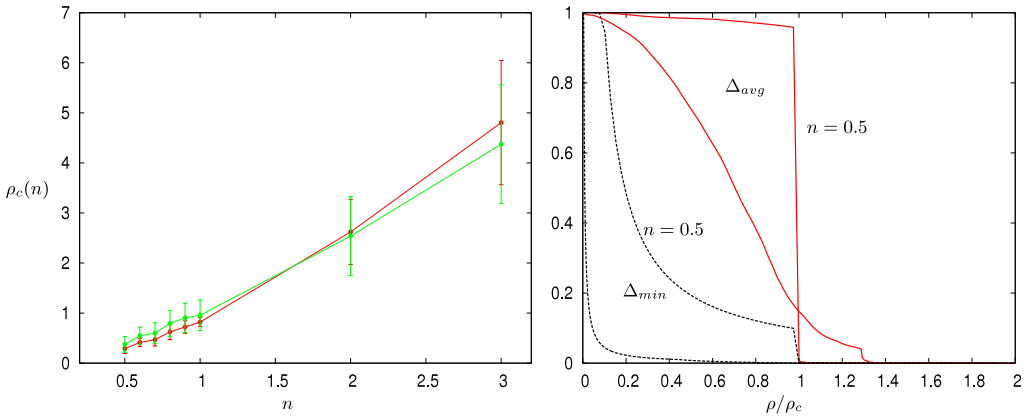

Figure 3. Left: results for the critical line $\rho_c(n)$ from MinOver$^+$ (red) and belief propagation for the supports (green). The details are explained in the text. Right: plot of the minimum support size (black dotted lines) and average support size (red solid lines) versus ρ/ρ_c for the ratios $n = 0.5$, 2.

6.2. The critical line $\rho(n)$ in von Neumann's model

As a second example we revisit the von Neumann model (3). We are interested in calculating the value of the critical line $\rho_c(n)$ (with $n = N/M$) above which it is not possible to find more solutions to the set of inequalities [6]. To find this critical line using the self-consistency equations for the supports, we assume that the critical line occurs when at least one support becomes zero. Thus for a fixed value of the ratio $n = N/M$ we increase the value of the global growth rate ρ until at least one of the supports becomes zero.

Numerical results for the critical line are reported in the left panel of figure 3 and compared with the MinOver$^+$ algorithm results. Here we have generated Poissonian graphs both for factor-nodes and for variable-nodes. Factor-nodes have an in-degree and out-degree average of 3. Thus the variable-nodes have the in-degree and out-degree average equal $3/n$. The sizes of the graphs have been generated as follows: for $n <= 1$, $M = 500$ and N is varied, while for $n > 1$, $N = 500$ and M is varied. The results are averaged over 50 graphs. Although the cavity equations for the supports can cope with much larger graphs, we keep their sizes small so as to allow comparison with the MinOver$^+$ algorithm results.

In the right panel of figure 3 we have plotted two possible "order parameters" that could be used to locate the critical line ρ_c. These are the minimum support size $\Delta_{\min}$ and the average support size Δ_{avg}. It is interesting to notice that for $n \leq 1$ both parameters become zero at the critical line (as given by the MinOver$^+$ algorithm). This implies that at the critical line the volume of the polytope collapses to a point. Remarkably, for $n \geq 1$ we have that $\Delta_{\min} = 0$ and $\Delta_{\text{avg}} > 0$ at the critical line, indicating that the polytope has collapsed in one or more dimensions, but it is still dimensionful in a subspace.

In figure 4 we also report the average running time $\bar{t}$ corresponding to figure 3 for both the belief propagation of the supports and the MinOver$^+$ algorithm. As can be clearly observed, belief propagation is faster than MinOver$^+$ by an order of magnitude. Besides, the average running time $\bar{t}$ grows with $n = N/M$ for $n < 1$, while it remains constant

J. Stat. Mech. (2012) P11003

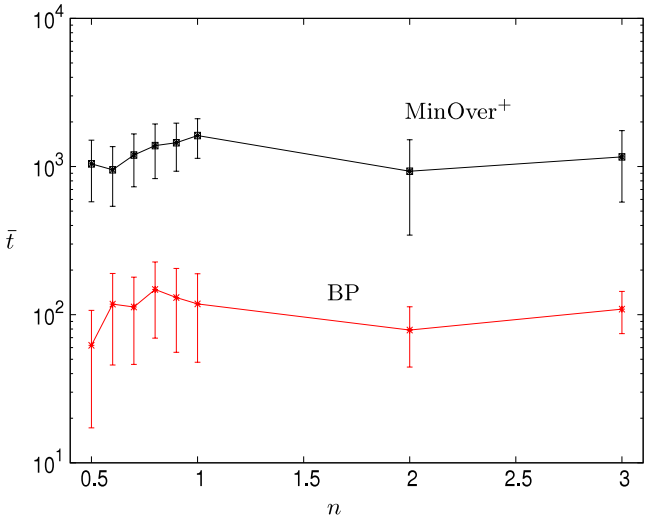

Figure 4. Average running time $\bar{t}$ versus n for the belief propagation for the supports and MinOver$^+$. As we can see, the belief propagation is an order of magnitude faster.

for $n > 1$. This is not surprising considering how the networks are generated (see the description in the text above).

6.3. A toy example for weighted population dynamics in the instance

In this section we illustrate *random assignment and variable locking* and *variable fixing* in a particular case of the example (16) discussed before. As we want to illustrate the method for estimating the integral rather than its use for finding the marginals, we consider the marginals here to be known Gaussian distributions $g_i(y; m_i, \sigma_i^2) \equiv g_i(y)$ normalized on the interval $[0, 1]$ and with mean value m_i and variance σ_i^2.

$$r(x) = \frac{1}{r} I^2(x), \qquad I(x) = \int_{[0,1]^3} \left[\prod_{i=1}^{3} \mathrm{d}y_i \, g_i(y_i) \right] \delta \left(x + \sum_{i=1}^{3} y_i - 1 \right)$$

where we also assume that $x \in [0, 1]$. Although this is a pedagogical example, it may be worth—for the sake of clarity but with the risk of being a bit repetitive—discussing it in some detail. We describe here what the method of *random assignment and variable locking* looks like in a pen-and-paper calculation. Firstly, for one of the multiple integrals $I(x)$, we note that due to the assignment $x \leftarrow 1 - \sum_{i=1}^{3} y_i$, and the fact that $x, y_1, y_2, y_3 \in [0, 1]$, we have that from the integration region $[0, 1]^3$ in the (y_1, y_2, y_3)-space, the region that actually contributes to the integral is the one below the plane $y_3 = 1 - y_1 - y_2$. Choosing an order of integration we write

$$I(x) = \int_0^1 \mathrm{d}y_1 \, g_1(y_1) \int_0^{1-y_1} \mathrm{d}y_2 \, g_2(y_2|y_1) \int_0^{1-y_1-y_2} \mathrm{d}y_3 \, g_3(y_3|y_1, y_2) w_2(y_1) w_3(y_1, y_2) \times \delta \left(x + \sum_{i=1}^{3} y_i - 1 \right)$$

J. Stat. Mech. (2012) P11003

where we have also reweighted the pdfs g_2 and g_3 so that we have normalization in the new integration intervals $[0, 1-y_1]$ and $[0, 1-y_1-y_2]$ respectively, with weights

$$w_2(y_1) = \int_0^{1-y_1} \mathrm{d}y_2\, g(y_2) = Q(y_1|m_2, \sigma_2^2), \qquad w_3(y_1, y_2) = Q(y_1 + y_2|m_3, \sigma_3^2)$$

where we have defined the function

$$Q(x|a, b) \equiv \frac{\mathrm{erf}(a/\sqrt{2b}) - \mathrm{erf}((-1+a+x)/\sqrt{2b})}{\mathrm{erf}(a/\sqrt{2b}) - \mathrm{erf}((-1+a)/\sqrt{2b})}$$

and with $g_2(y_2|y_1) = g_2(y_2)/w_2(y_1)$, $g_2(y_3|y_1, y_2) = g_3(y_3)/w_3(y_1, y_2)$. Then, to estimate the integral $I(x)$ and assign a value of x we must implement the following steps:

(1) draw a Gaussian random variable y_1 with mean m_1 and variance σ_1^2 in the interval $[0, 1]$, draw a Gaussian random variable y_2 with mean m_2 and variance σ_2^2 in the interval $[0, 1-y_1]$, and draw Gaussian random variable y_3 with mean m_3 and variance σ_3^2 in the interval $[0, 1-y_1-y_2]$;
(2) assign x the value $x \leftarrow x^\star = 1 - \sum_{i=1}^3 y_i$;
(3) calculate the weight $w_2(y_1)w_3(y_1, y_2)$.

Now for the second $I(x)$ in our example, the value of x is now locked at $x^\star$. After restricting the integral to the region with non-zero value and using the Dirac delta to integrate over z_3 we can write

$$I(x^\star) = \int_0^{1-x^\star} \mathrm{d}z_1\, g_1(z_1) \int_0^{1-z_1-x^\star} \mathrm{d}z_2\, g_2(z_2|z_1, x^\star) g_3[z_3(z_1, z_2, x^\star)] w_1(x^\star) w_2(y_1, x^\star)$$

with $z_3(z_1, z_2, x^\star) = 1 - x^\star - z_1 - z_2$, and with the weights with the same definition as before. From here we see that to estimate $I(x^\star)$ by Monte Carlo methods we can do the following:

(1) draw a Gaussian random variable z_1 with mean m_1 and variance σ_1^2 on the interval $[0, 1-x^\star]$ and draw a Gaussian random variable z_2 with mean m_2 and variance σ_2^2 on the interval $[0, 1-x^\star-z_1]$;
(2) calculate the weight $g_3[z_3(z_1, z_2, x^\star)]w_1(x^\star)w_2(y_1, x^\star)$.

From here the total weight is $\omega = w_2(y_1)w_3(y_1, y_2)g_3[z_3(z_1, z_2, x^\star)]w_1(x^\star)w_2(y_1, x^\star)$ at $x = x^\star$. Repeating the process $\mathcal{N}$ times we obtain a population of pairs $\{(x_\alpha^\star, \omega_\alpha)\}_{\alpha=1}^{\mathcal{N}}$ from which to construct $r(x)$.

To implement the numerics we have chosen $m_1 = 0$, $m_2 = 1/2$, and $m_3 = 1$ while for the standard deviations we have taken $\sigma_1 = \sqrt{0.4}$, $\sigma_2 = \sqrt{0.2}$, and $\sigma_3 = \sqrt{0.4}$ and we have reported the results in figure 5, with a more detailed explanation in its caption.

6.4. Random graphs and the comparison with the Hit-and-Run algorithm

Finally, our last example corresponds to a random network of $N = 25$ variable-nodes and $M = 10$ factor-nodes. Although small, this size can be found in actual applications in metabolic networks, for instance, human red blood cells [11, 15]. Here we have used the WPDI algorithm with variable fixing with a population size of $\mathcal{N} = 10^3$. For the averaged weights we use a time window of $T = 10^2$ weights for the cavity marginals and $T = 10^6$ for the real marginals. Besides this, for the final marginals a population is not used due

J. Stat. Mech. (2012) P11003

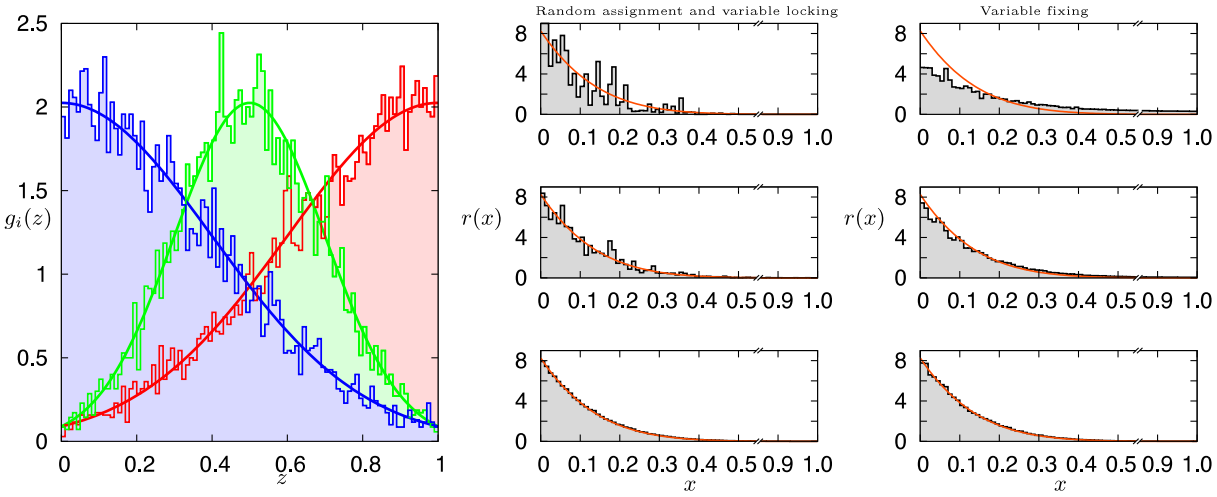

Figure 5. In this figure we report the results from using the weighted population dynamics in the instance applied to the toy example reported in section 6.3. We have taken three Gaussian distributions (left panel) with means 0, 1/2, and 1 and variances $\sigma^2 = 0.4, 0.2$, and 0.4. The centre panel corresponds to estimating $r(x)$ by means of the WPDI algorithm using the method of random assignment and variable locking using population sizes of $\mathcal{N} = 7\times10^2$, 7×10^3, and 7×10^5 (top to bottom). The right panel corresponds to WPDI using the variable fixing method for a population of $\mathcal{N} = 8\times10^2$ and using averaged weight sizes $T = 15, 60, 300$ (top to bottom). The orange solid line in the centre and right panels corresponds to the exact result for $r(x)$.

to the simple fact that then we need to plot them, so we construct the histogram directly by fixing the value of x at the mid-point for each bin. Results for each marginal $P_i(x_i)$ for $i = 1, \ldots, 25$ are presented in figure 6 and we have compared our findings with results obtained with the Hit-and-Run algorithm (for the implementation see appendix B). As one can see, the agreement is excellent considering that the network is small and therefore loopy, and our equations rely on the Bethe approximation.

7. Conclusions

As we have discussed, many problems arising from diverse research topics may be mathematically recast as properties related to the volumes of polytopes. Methods for estimating these properties can be roughly divided into mathematical and Monte Carlo approaches. In both cases, the approach used may not be efficient enough for treating polytopes of practical relevance (e.g. large metabolic networks). In this paper we have shown that for diluted random polytopes, a novel weighted belief-propagation algorithm can be used to calculate efficient single-site marginals. We have discussed several options for implementing this algorithm, devoting more lines to explaining in detail the WPDI with its two versions: *random assignment and variable locking* and *variable fixing*. Importantly, we also pointed out that it is possible to write down self-consistency equations for the supports of the marginals which, apart of the use in implementing WPDI efficiently, can be employed to obtain relevant information about the system with little effort (e.g. to

J. Stat. Mech. (2012) P11003

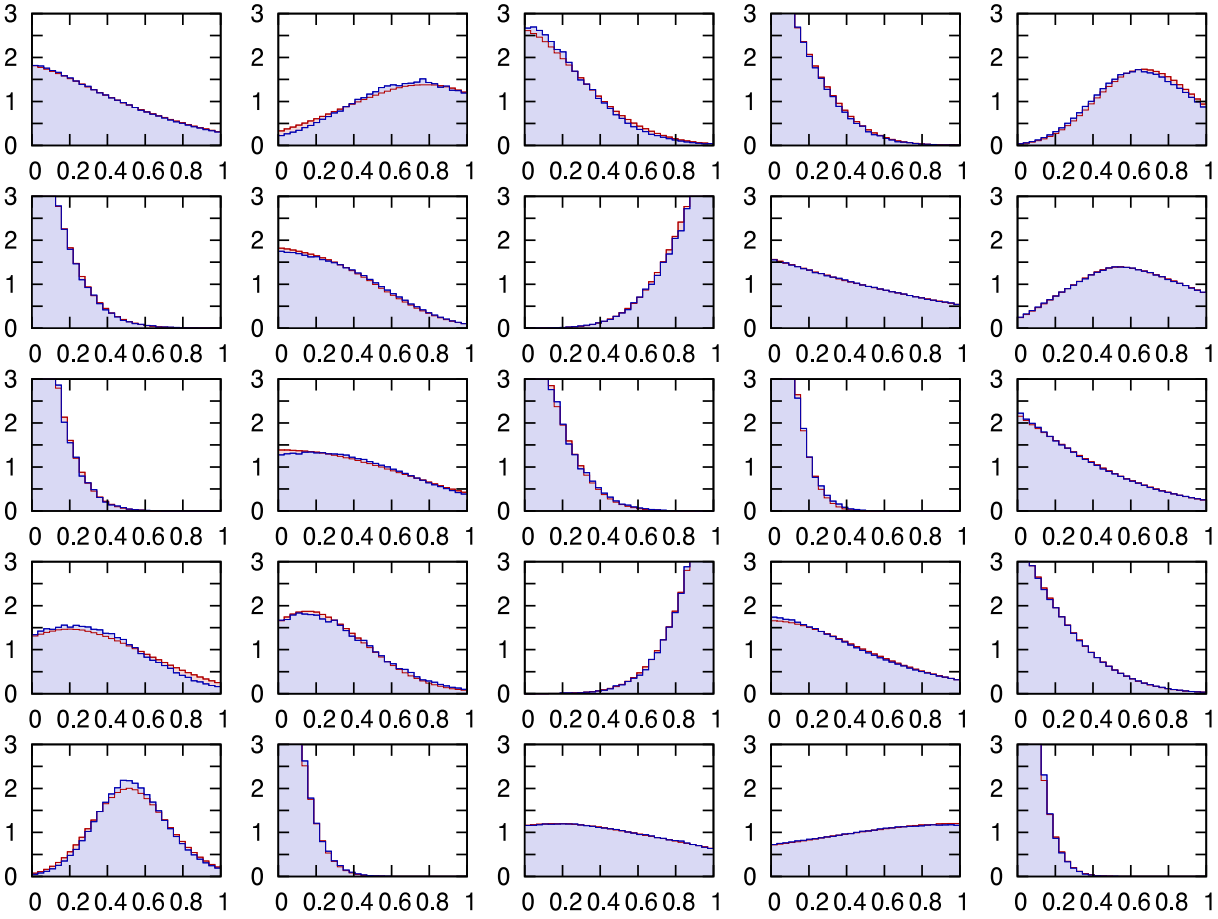

Figure 6. Results for the marginals $P_i(x_i)$ for $i = 1, \ldots, 25$ (left to right and top to bottom) of the WPDI algorithm with variable fixing (blue) compared with estimates for the marginals using the Hit-and-Run algorithm (red).

estimate the ranges of reaction rates allowed by stoichiometry; to evaluate the impact of a genetic mutation). Examples are used throughout this paper to illustrate, to support, and to benchmark our novel theoretical findings and, hence, we have kept most of them fairly simple. The attentive reader would have noticed that we have left some computational issues unattended. For instance we have not discussed in detail the fact that equation (5) is quadratic in the number of neighbours of i and how this dependence can be linearized by adapting the nice trick given in [13]. Our main goal here is to pique the reader's interest with the novel ideas presented rather than discussing ways of improving the algorithmic implementation. These issues are certainly important, but we believe that they belong in another paper.

The work presented in this paper has several applications and extensions. First of all we aim to apply weighted belief propagation to real metabolic networks, possibly starting from the human red blood cell one, which has size comparable to that of the example presented here, and extending the study to other reference metabolic networks (e.g. *E. coli*). Since we are able to tackle both FBA and the von Neumann frameworks, we may use our tools to, for instance, compare these two approaches and relate our results to

doi:10.1088/1742-5468/2012/11/P11003

J. Stat. Mech. (2012) P11003

the ones already published using these two methods. Secondly, we plan to apply weighted belief propagation to solve a metabolic network model where the direction of the reactions is not fixed and has to be determined by minimization of the Gibbs free energy. Finally, we plan to see how to use the self-consistency equations for the supports to study for instance the behaviour of metabolic networks under perturbations.

Acknowledgments

IPC would like to thank Lenka Zdeborová for pointing out some work done in the area of compressed sensing. FFC would like to thank for hospitality the Department of Mathematics at King's College London. FAM acknowledges European Union Grants PIRG-GA-2010-277166 and PIRG-GA-2010-268342. FFC acknowledges the Spanish Government grant FIS2009-09508. We would also like to thank the referees from JSTAT for helping us make this a better and more well-rounded paper.

Appendix A. Derivation of single-site marginals using the cavity method

The first step in deriving the cavity equations (5) and (6) for problem (2) is to start by calculating the joint marginal $P_\mu(x_{\partial\mu})$ of the set of variables $x_{\partial\mu}$ connected to the factor-node μ. By definition, this is given by

$$\begin{aligned} P_\mu(x_{\partial\mu}) &= \frac{1}{V}\int \mathrm{d}\boldsymbol{x}_{\setminus\partial\mu}\prod_\nu f_\nu(x_{\partial\nu}) = \frac{1}{V}f_\mu(x_{\partial\mu})\int \mathrm{d}\boldsymbol{x}_{\setminus\partial\mu}\prod_{\nu(\neq\mu)} f_\mu(x_{\partial\mu}) \\ &= \frac{1}{V_\mu}f_\mu(x_{\partial\mu})P_\mu^{(\mu)}(x_{\partial\mu}), \\ P_\mu^{(\mu)}(x_{\partial\mu}) &= \frac{1}{V^{(\mu)}}\int \mathrm{d}\boldsymbol{x}_{\setminus\partial\mu}\prod_{\nu(\neq\mu)} f_\mu(x_{\partial\mu}), \end{aligned}$$

with $f_\mu(x_{\partial\mu}) = \Theta[h_\mu(x_{\partial\mu})]$. Here, $P_\mu^{(\mu)}(x_{\partial\mu})$ is the joint marginal for the variables $x_{\partial\mu}$ in the absence of factor-node μ. If the graph is a tree or is locally tree-like, the set of variables $x_{\partial\mu}$ in the absence of μ are mostly uncorrelated, and we can confidently write $P_\mu^{(\mu)}(x_{\partial\mu}) = \prod_{\ell\in\partial\mu}P_\ell^{(\mu)}(x_\ell)$.

Let us move on to find an expression for the single-site marginals $P_i(x_i)$:

$$\begin{aligned} P_i(x_i) &= \frac{1}{V}\int \mathrm{d}\boldsymbol{x}_{\setminus i}\prod_\mu f_\mu(x_{\partial\mu}) \\ &= \frac{1}{V}\int \mathrm{d}\boldsymbol{x}_{\setminus i}\prod_{\mu\in\partial i} f_\mu(x_{\partial\mu})\prod_{\mu\notin\partial i} f_\mu(x_{\partial\mu}) \\ &= \frac{1}{V}\int \mathrm{d}x_{(\partial\mu\ni i)\setminus i}\prod_{\mu\in\partial i} f_\mu(x_{\partial\mu})\int \mathrm{d}\boldsymbol{x}_{\partial\mu\not\ni i}\prod_{\mu\notin\partial i} f_\mu(x_{\partial\mu}). \end{aligned}$$

Here the notation $\boldsymbol{x}_{(\partial\mu\ni i)\setminus i}$ stands for the collection of variable-nodes which share a factor-node with i with the exception of the node i itself. The important point to note here is that $\int \mathrm{d}\boldsymbol{x}_{\partial\mu\not\ni i}\prod_{\mu\notin\partial i}f_\mu(x_{\partial\mu}) \propto P_{\mu\in\partial i}^{(\mu\in\partial i)}(x_{(\partial\mu\ni i)\setminus i})$. As variable-node i is absent in this

J. Stat. Mech. (2012) P11003

marginalization and as we are considering locally tree-like graphs, we can confidently write $P^{(\mu\in\partial i)}_{\mu\in\partial i}(x_{(\partial\mu\ni i)\setminus i}) = \prod_{\mu\in\partial i} P^{(\mu)}_{\mu}(x_{\partial\mu\setminus i})$. All in all we have the following expression:

$$P_i(x_i) = \frac{1}{V_i}\prod_{\mu\in\partial i}\int \mathrm{d}x_{\partial\mu\setminus i}\, f_\mu(x_{\partial\mu}) P^{(\mu)}_{\mu}(x_{\partial\mu\setminus i}).$$

Recalling that the joint marginals $P^{(\mu)}_{\mu}$ themselves factorize, we finally write

$$P_i(x_i) = \frac{1}{V_i}\prod_{\mu\in\partial i}\int \mathrm{d}x_{\partial\mu\setminus i}\, f_\mu(x_{\partial\mu}) \prod_{j\in\partial\mu\setminus i} P^{(\mu)}_{j}(x_j).$$

Closed equations for the cavity marginals $P^{(\mu)}_{j}(x_j)$ can readily be written down by simply removing one of the factor-nodes in the neighbourhood of i, that is

$$P^{(\nu)}_{i}(x_i) = \frac{1}{V^{(\nu)}_{i}}\prod_{\mu\in\partial i\setminus\nu}\int \mathrm{d}x_{\partial\mu\setminus i}\, f_\mu(x_{\partial\mu}) \prod_{j\in\partial\mu\setminus i} P^{(\mu)}_{j}(x_j).$$

Notice that the functions $m^{(i)}_{\mu}(x_i)$ that we introduced in the main text are combinations of terms from of the first product in the preceding equation:

$$m^{(i)}_{\mu}(x_i) = \frac{1}{m^{(i)}_{\mu}}\int \mathrm{d}x_{\partial\mu\setminus i}\, f_\mu(x_{\partial\mu}) \prod_{j\in\partial\mu\setminus i} P^{(\mu)}_{j}(x_j).$$

Appendix B. The Hit-and-Run algorithm

As we mentioned previously, the Hit-and-Run algorithm is a numerical method for sampling directly and uniformly the volume of a polytope. This useful feature makes it particularly attractive for treating problems such as FBA [16], but the algorithm has the unfortunate drawback of being rather slow. Nevertheless, we find it useful in our case as a benchmark for our message-passing algorithm.

We explain the Hit-and-Run algorithm by following the nice work in [17], simply adapting the notation. Consider the following set of inequalities:

$$(\boldsymbol{\xi}^\mu)^T\boldsymbol{x} \geq \gamma^\mu, \qquad \mu = 1,\ldots,M$$

with $\boldsymbol{x}\in\mathbb{R}^N$ and $\|\boldsymbol{\xi}^\mu\| = 1, \forall\,\mu = 1,\ldots,M$, the unit vectors normal to the M hyperplanes encapsulating the polytope. Leaving aside some mathematical technicalities and simply assuming that the polytope is properly defined, the algorithm samples the volume by starting at a given point $\boldsymbol{X}$ inside the polytope and moving along a random direction $\boldsymbol{v}$. To know how far to move, one needs to calculate all the interactions between all hyperplanes and the straight line passing through $\boldsymbol{X}$ in the $\boldsymbol{v}$ direction. Then the closest hyperplanes would be the furthest region within the polytope that can be reached. This procedure can be iterated randomly as follows:

Step 0. Find a point $\boldsymbol{X}^{(0)}$ interior to the polytope. Set $n = 0$.

Step 1. Generate a direction vector $\boldsymbol{v}^{(n)}$ from a uniform distribution.

J. Stat. Mech. (2012) P11003

Step 2. Determine the intersections:

$$\lambda_\mu = \frac{(\boldsymbol{\xi}^\mu)^T \boldsymbol{X}^{(n)} - \gamma^\mu}{(\boldsymbol{\xi}^\mu)^T \boldsymbol{v}^{(n)}}, \qquad \mu = 1, \ldots, M$$
$$\lambda^+ = \min_{1 \le \mu \le M} \{\lambda_\mu | \lambda_\mu > 0\}, \qquad \lambda^- = \max_{1 \le \mu \le M} \{\lambda_\mu | \lambda_\mu < 0\},$$

where $\lambda^\pm$ allow us to determine the hyperplanes closest to $\boldsymbol{X}^{(n)}$.

Step 3. Generate a uniform random number $u \in [0, 1]$ and set

$$\boldsymbol{X}^{(n+1)} = \boldsymbol{X}^{(n)} + \left(\lambda^- + u(\lambda^+ - \lambda^-)\right) \boldsymbol{v}^{(n)}.$$

Step 4. Set $n \to n + 1$ and go to Step 1 or stop if satisfied with the sampling.

It was proven [4, 17] that the statistics of the sequence $\{\boldsymbol{X}^{(n)}\}_{n=0}^{\infty}$ converges to the uniform distribution on the polytope.

References

[1] Kauffman K J, Prakash P and Edwards J S, 2003 *Curr. Opin. Biotechnol.* **14** 491
[2] Gardner B E and Derrida, 1988 *J. Phys. A: Math. Gen.* **21** 271
[3] Krzakala F, Mézard M, Sausset F, Sun Y and Zdeborová L, 2012 *Phys. Rev.* X **2** 021005
[4] Smith R L, 1984 *Oper. Res.* **32** 1296
[5] von Neumann J, 1937 *Ergebn. eines Math. Kolloq.* **8** 73
von Neumann J, 1945 *Rev. Econ. Studies* **13** 1 (Engl. transl.)
[6] De Martino A and Marsili M, 2005 *J. Stat. Mech.* L09003
[7] De Martino A, Martelli C, Monasson R and Pérez Castillo I, 2007 *J. Stat. Mech.* P05012
[8] De Martino A, Martelli C and Massucci F A, 2009 *Europhys. Lett.* **85** 38007
[9] Martelli C, Martino A D, Marinari E, Marsili M and Pérez Castillo I, 2009 *Proc. Nat. Acad. Sci.* **106** 2607
[10] Martino A D and Marinari E, 2010 *J. Phys.: Conf. Ser.* **233** 012019
[11] Martino A D, Granata D, Marinari E, Martelli C and Kerrebroeck V V, 2010 *J. Biomed. Biotech.* **2010** 415148
[12] Baron D, Sarvotham S and Baraniuk R G, 2010 *IEEE Trans. Signal Process.* **58** 269
[13] Braunstein A, Mulet R and Pagnani A, 2008 *BMC Bioinform.* **9** 240
[14] Krauth W and Mezard M, 1987 *J. Phys. A: Math. Gen.* **20** L745
[15] De Martino D, Figliuzzi M, De Martino A and Marinari E, 2012 *PLoS Comput. Biol.* **8** e1002562
[16] Almaas K, Kovacs B, Vicsek T, Oltvai Z M and Barabasi A-L, 2004 *Nature* **427** 839
[17] Berbee H, Boender C, Ran A R, Scheffer C, Smith R and Telgen J, 1987 *Mathemat. Programm.* **37** 184

J. Stat. Mech. (2012) P11003