LETTER Communicated by Elena Agliari

Learning Compositional Representations of Interacting Systems with Restricted Boltzmann Machines: Comparative Study of Lattice Proteins

Jérôme Tubiana
jertubiana@gmail.com
Simona Cocco
cocco@lps.ens.fr
Rémi Monasson
monasson@lpt.ens.fr
Laboratory of Physics of the Ecole Normale Supérieure, CNRS and PSL Research, 75005 Paris, France

A restricted Boltzmann machine (RBM) is an unsupervised machine learning bipartite graphical model that jointly learns a probability distribution over data and extracts their relevant statistical features. RBMs were recently proposed for characterizing the patterns of coevolution between amino acids in protein sequences and for designing new sequences. Here, we study how the nature of the features learned by RBM changes with its defining parameters, such as the dimensionality of the representations (size of the hidden layer) and the sparsity of the features. We show that for adequate values of these parameters, RBMs operate in a so-called compositional phase in which visible configurations sampled from the RBM are obtained by recombining these features. We then compare the performance of RBM with other standard representation learning algorithms, including principal or independent component analysis (PCA, ICA), autoencoders (AE), variational autoencoders (VAE), and their sparse variants. We show that RBMs, due to the stochastic mapping between data configurations and representations, better capture the underlying interactions in the system and are significantly more robust with respect to sample size than deterministic methods such as PCA or ICA. In addition, this stochastic mapping is not prescribed a priori as in VAE, but learned from data, which allows RBMs to show good performance even with shallow architectures. All numerical results are illustrated on synthetic lattice protein data that share similar statistical features with real protein sequences and for which ground-truth interactions are known.

1 Introduction

Many complex, interacting systems have collective behaviors that cannot be understood based solely on a top-down approach. This is either because the

Neural Computation **31, 1671–1717** (2019)
doi:10.1162/neco_a_01210

underlying microscopic interactions between the constituents of the system are unknown—as in biological neural networks, where the set of synaptic connections are unique to each network—or because the complete description is so complicated that analytical or numerical resolution is intractable—as for proteins, for which physical interactions between amino acids can in principle be characterized, but accurate simulations of protein structures or functions are computationally prohibitive. In the past two decades, the increasing availability of large amounts of data collected by high-throughput experiments such as large-scale functional recordings in neuroscience (EEG, fluorescence imaging) (Schwarz et al., 2014; Wolf et al., 2015), fast sequencing technologies (Finn et al., 2015; Kolodziejczyk, Kim, Svensson, Marioni, & Teichmann, 2015; single RNA seq) or deep mutational scans (Fowler et al., 2010) has shed new light on these systems.

Given such high-dimensional data, one fundamental task is to establish a descriptive phenomenology of the system. For instance, given a recording of spontaneous neural activity in a brain region or in the whole brain (e.g., in larval zebrafish), we would like to identify stereotypes of neural activity patterns (e.g., activity bursts, synfire chains, cell-assembly activations) describing the dynamics of the system. This representation is in turn useful to link the behavior of the animal to its neural state and to understand the network architecture. Similarly, given a multiple sequence alignment (MSA) of protein sequences (i.e., a collection of protein sequences from various genes and organisms that share common evolutionary ancestry), we would like to identify amino acid motifs controlling the protein functionalities and structural features and identify, in turn, subfamilies of proteins with common functions. Important tools for this purpose are unsupervised representation-learning algorithms. For instance, principal component analysis can be used for dimensionality reduction, that is, for projecting system configurations into a low-dimensional representation, where similarities between states are better highlighted and the system evolution is tractable. Another important example is clustering, which partitions the observed data into different prototypes. Though these two approaches are very popular, they are not always appropriate: some data are intrinsically multidimensional and cannot be reduced to a low-dimensional or categorical representation. Indeed, configurations can mix multiple, weakly related features, such that using a single global distance metric would be too reductive. For instance, neural activity states are characterized by the clusters of neurons that are activated, which are themselves related to a variety of distinct sensory, motor, or cognitive tasks. Similarly, proteins have a variety of biochemical properties such as binding affinity and specificity, thermodynamic stability, or allostery, which are controlled by distinct amino acid motifs within their sequences. In such situations, other approaches such as independent component analysis or sparse dictionaries, which aim at representing the data by a (larger) set of independent

latent factors, appear to be more appropriate (McKeown et al., 1998; Rivoire, Reynolds, & Ranganathan, 2016).

A second goal is to infer the set of interactions underlying the system's collective behavior. In the case of neural recordings, we would look for functional connections that reflect the structure of the relevant synaptic connections in a given brain state. In the case of proteins, we would like to know what interactions between amino acids shape the protein structure and functions. For instance, a Van Der Waals repulsion between two amino acids is irrelevant if both are far from one another in the tridimensional structure; the set of relevant interactions is therefore linked to the structure. One popular approach for inferring interactions from observation statistics relies on graphical (e.g., Ising or Potts) models. It consists in first defining a quadratic log-probability function, then inferring the associated statistical fields (site potentials) and pairwise couplings by matching the first- and second-order moments of data. This can be done efficiently through various approaches (see Nguyen, Zecchina, & Berg, 2017, and Cocco, Feinauer, Figliuzzi, Monasson, & Weigt, 2018) for recent reviews. The inverse Potts approach, called direct-coupling-analysis (Weigt, White, Szurmant, Hoch, & Hwa, 2009; Morcos et al., 2011) in the context of protein sequence analysis, helps predict structural contact maps for a protein family from sequence data only. Moreover, such an approach defines a probability distribution that can be used for artificial sample generation and, more broadly, quantitative modeling—for example, of mutational fitness landscape prediction in proteins (Figliuzzi, Jacquier, Schug, Tenaillon, & Weigt, 2016; Hopf et al., 2017) or neural state information content (Tkacik, Prentice, Balasubramanian, & Schneidman, 2010) or brain states (Posani, Cocco, Ježek, & Monasson, 2017). The main drawback of graphical models is that, unlike representation learning algorithms, they do not provide any direct, interpretable insight over the data distribution. Indeed, the relationship between the inferred parameters (fields and couplings) and the typical configurations associated with the probability distribution of the model is mathematically well defined but intricate in practice. For instance, it is difficult to deduce the existence of data clusters or global collective modes from the knowledge of the fields and couplings.

An interesting alternative for overcoming this issue are restricted Boltzmann machines (RBM). An RBM is a graphical model that can learn both a representation and a distribution of the configuration space, naturally combining both approaches. RBM are bipartite graphical models, constituted by a visible layer carrying the data configurations and a hidden layer, where their representation is formed (see Figure 1a). Unlike Boltzmann machines, there are no couplings within the visible layer or between hidden units. RBM were introduced in Ackley, Hinton, and Sejnowski (1987) and popularized by Hinton et al. (Hinton, 2002; Hinton, Osindero, & Teh, 2006) for feature extraction and pretraining of deep networks. More recently, RBMs were shown to be efficient for modeling coevolution in protein sequences

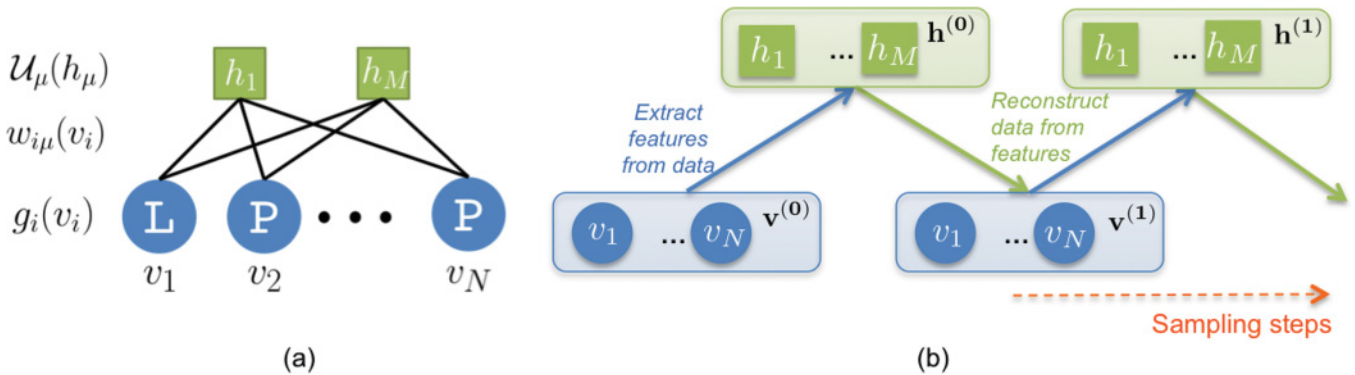

Figure 1: (a) A restricted Boltzmann machine (RBM) is a two-layer bipartite network: a visible layer (units v_i) carrying the data and a hidden layer (units h_μ) where the data are represented. The dimensions of the data and representation spaces are, respectively, N and M. Visible and hidden units are subject to local potentials denoted by, respectively, g_i and $\mathcal{U}_\mu$. Here, we consider that visible units v_i take discrete values (such as amino acids shown in the visible units' blue circles), while hidden units h_μ may take either discrete or real values. (b) The alternate Gibbs sampler for sampling from $P(\mathbf{v}, \mathbf{h})$. Sampling consists of alternating between a stochastic feature extraction from data (i.e., sample from $P(\mathbf{h}|\mathbf{v})$) and a stochastic reconstruction of data from features (i.e., sample from $P(\mathbf{v}|\mathbf{h})$).

(Tubiana, Cocco, & Monasson, 2018). The features inferred are sequence motifs related to the structure, function, and phylogeny of the protein, and they can be recombined to generate artificial sequences with putative properties. An important question is to understand the nature and statistics of the representation inferred by RBMs and how they depend on the choice of model parameters (nature and number of hidden units, sparse regularization). Indeed, unlike PCA, independent component analysis (ICA), or sparse dictionaries, no constraints on the statistics or the nature of the representations, such as decorrelation or independence, are explicitly enforced in RBM. To answer this question, we apply here RBM on synthetic alignments of lattice protein sequences, for which ground-truth interactions and fitness functions are available. We analyze and interpret the nature of the representations learned by RBM as a function of its defining parameters and in connection with theoretical results on RBMs drawn from random statistical ensembles obtained with statistical physics tools (Tubiana & Monasson, 2017). We then compare our results to the other feature extraction methods.

The letter is organized as follows. In section 2, we formally define RBM and present the main steps of the learning procedure. In section 3, we discuss the different types of representations RBM that can learn, with a reminder of recent related results on the different operation regimes, in particular, the so-called compositional phase, theoretically found in random RBM ensembles. In section 4, we introduce lattice proteins (LP) and present the main results of the applications of RBM to LP sequence data. Section 5 is dedicated to the comparison of RBM with other representation learning algorithms, including principal component analysis (PCA), independent component analysis (ICA), sparse principal component analysis (sPCA), sparse dictionaries, sparse autoencoders (sAE), and sparse variational autoencoders (sVAE).

2 Restricted Boltzmann Machines

2.1 Definition. A restricted Boltzmann machine (RBM) is a joint probability distribution over two sets of random variables: the visible layer $\mathbf{v} = (v_1, .., v_i, \ldots, v_N)$ and hidden layer $\mathbf{h} = (h_1, .., h_\mu, \ldots, h_M)$. It is formally defined on a bipartite, two-layer graph (see Figure 1a). Depending on the nature of data considered, the visible units v_i can be continuous or categorical, taking q values. Here, we use notations for protein sequence analysis in which each visible unit represents a site of the multiple sequence alignment and takes $q = 21$ values (20 amino acids + 1 alignment gap). The hidden units h_μ represent latent factors that can be either continuous or binary. Their joint probability distribution is

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp\Bigg(\underbrace{\sum_{i=1}^{N} g_i(v_i) - \sum_{\mu=1}^{M} \mathcal{U}_\mu(h_\mu) + \sum_{i,\mu} h_\mu \, w_{i\mu}(v_i)}_{-E(\mathbf{v},\mathbf{h})} \Bigg), \tag{2.1}$$

where the fields $g_i(v)$ and potentials $\mathcal{U}_\mu$ control the conditional probabilities of the v_i, h_μ and the weights $w_{i\mu}(v)$ couple the visible and hidden variables. The partition function $Z = \sum_{\mathbf{v}} \int d\mathbf{h} e^{-E(\mathbf{v},\mathbf{h})}$ is defined such that P is normalized to unity. Some choices for the hidden-unit potentials are:

- The Bernoulli potential:
  $\mathcal{U}(0) = 0, \mathcal{U}(1) = u_1$, and $\mathcal{U}(h) = +\infty$ if $h \neq 0, 1$.
- The quadratic potential,

$$\mathcal{U}(h) = \frac{1}{2}\gamma h^2 + \theta\, h, \tag{2.2}$$

  with h real-valued.
- Rectified linear unit (ReLU) potential $\mathcal{U}(h) = \frac{1}{2}\gamma h^2 + \theta\, h$, with h real-valued and positive, and $\mathcal{U}(h) = +\infty$ for negative h.
- The double rectified linear unit (dReLU) potential,

$$\mathcal{U}(h) = \frac{1}{2}\gamma^+ (h^+)^2 + \frac{1}{2}\gamma^- (h^-)^2 + \theta^+ h^+ + \theta^- h^-, \tag{2.3}$$

  where $h^+ = \max(h, 0)$, $h^- = \min(h, 0)$ and h is real-valued.

Bernoulli and quadratic potentials are standard choices in the literature. RBM with ReLU-like average activity were originally proposed in Nair and Hinton (2010), and the above potential, which can be sampled from exactly, was introduced in Tubiana and Monasson (2017). The so-called double ReLU potential, introduced in Tubiana et al. (2018), is a more general form, with an associated distribution that can be asymmetric and interpolate between bimodal, gaussian, or Laplace-like sparse distributions depending on the choice of the parameters (see Figure 2a).

Although we do not discuss them here, several extensions to the original RBM energy form were also proposed, such as covariance RBM (Dahl, Ranzato, Mohamed, & Hinton, 2010), spike-and-slab RBM (Courville, Bergstra, & Bengio, 2011), and batch-normalized RBM/DBM (Vu, Nguyen, Le, Luo, & Phung, 2018).

2.2 Sampling. Standard sampling from distribution 2.1 is achieved by the alternate Gibbs sampling Monte Carlo algorithm, which exploits the bipartite nature of the interaction graph (see Figure 1b). Given a visible layer configuration $\mathbf{v}$, the hidden unit μ receives the input

$$I_\mu(\mathbf{v}) = \sum_i w_{i\mu}(v_i), \tag{2.4}$$

and the conditional probability of the hidden-unit configuration is given by $P(\mathbf{h}|\mathbf{v}) = \prod_\mu P(\mathbf{h}_\mu|\mathbf{v})$, where

$$P(h_\mu|\mathbf{v}) \propto \exp\big(-\mathcal{U}_\mu(h_\mu) + h_\mu\, I_\mu(\mathbf{v})\big). \tag{2.5}$$

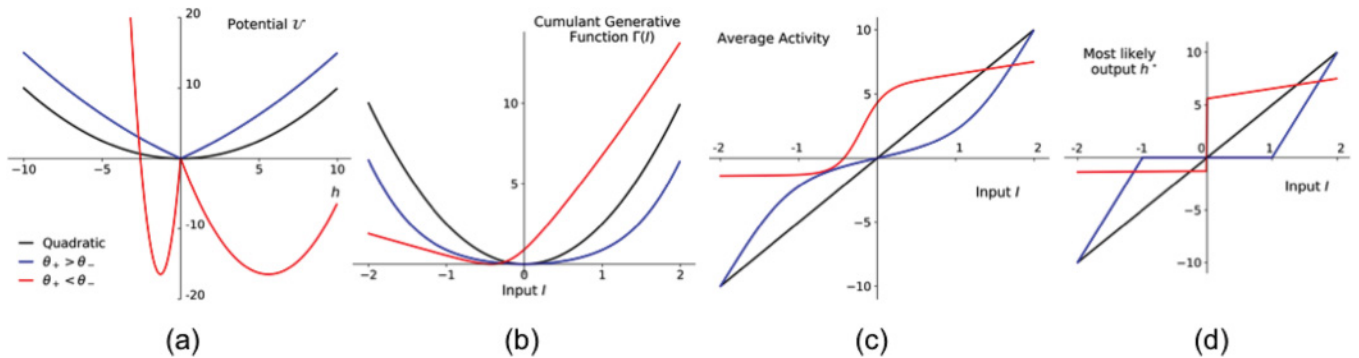

Figure 2: The double rectified linear units (dReLU) potential. (a) Three examples of potentials $\mathcal{U}$ defining the hidden-unit type in RBM: quadratic potential (black, $\gamma = 0.2$), and dReLU potential (blue: $\gamma^+ = \gamma^- = 0.1, \theta^+ = -\theta^- = 1$; red: $\gamma^+ = 1, \gamma^- = 20, \theta^+ = -6$, $\theta^- = 25$). (b) Cumulant generative function Γ as a function of the input I. (c) Average activity of hidden unit h as a function of the input I. (d) Most likely hidden unit value $H(I)$ as a function of the input I.

Similarly, given a hidden layer configuration $\mathbf{h}$, one can sample from the conditional probability distribution $P(\mathbf{v}|\mathbf{h}) = \prod_i P(v_i|\mathbf{h})$, where $P(v_i|\mathbf{h})$ is a categorical distribution,

$$P(v_i|\mathbf{h}) \propto \exp\big(g_i(v_i) + I_i^v(\mathbf{h})\big), \tag{2.6}$$

with $I_i^v(\mathbf{h}) = \sum_\mu h_\mu\, w_{i\mu}(v)$. Alternate sampling from $P(\mathbf{h}|\mathbf{v})$ and $P(\mathbf{v}|\mathbf{h})$ defines a Markov chain that eventually converges toward the equilibrium distribution. The potential $\mathcal{U}_\mu$ determines how the average conditional hidden-unit activity $\langle h_\mu\rangle(I)$ and the transfer function $H_\mu(I) = \arg\max P(h_\mu|I)$ vary with the input $I = I_\mu(\mathbf{v})$ (see Figure 2). For quadratic potential, both are linear functions of the input I, whereas for the ReLU potential, we have $H_\mu(I) = \max\big(\frac{I_\mu - \theta_\mu}{\gamma_\mu}, 0\big)$. The dReLU potential is the most general form and can effectively interpolate between quadratic ($\theta_+ = \theta_-$, $\gamma_+ = \gamma_-$), ReLU ($\gamma_- \to \infty$), and Bernoulli ($\gamma_\pm = \mp\theta_\pm \to +\infty$) potentials.

For $\theta_+ > \theta_-$ (the blue curve in Figure 2) small inputs I barely activate the hidden unit, and the average activity is a soft ReLU, whereas for $\theta_+ < \theta_-$, the average activity rises abruptly at $\frac{\theta_+\sqrt{\gamma_-} - \theta_-\sqrt{\gamma_+}}{\sqrt{\gamma_+} + \sqrt{\gamma_-}}$ (the red curve in Figure 2), similar to a soft thresholding function.

2.3 Marginal Distribution. The marginal distribution over the visible configurations, $P(\mathbf{v})$, is obtained by integration of the joint distribution $P(\mathbf{v}, \mathbf{h})$ over $\mathbf{h}$, and reads

$$\begin{aligned} P(\mathbf{v}) &= \int \prod_{\mu=1}^{M} dh_\mu P(\mathbf{v}, \mathbf{h}) \\ &= \frac{1}{Z} \exp\Bigg(\underbrace{\sum_{i=1}^{N} g_i(v_i) + \sum_{\mu=1}^{M} \Gamma_\mu\big(I_\mu(\mathbf{v})\big)}_{-E_{\text{eff}}(\mathbf{v})} \Bigg), \end{aligned} \tag{2.7}$$

where $\Gamma_\mu(I) = \log\big[\int dh\, e^{-\mathcal{U}_\mu(h) + hI}\big]$ is the cumulant generative function, or log Laplace transform, associated with the potential $\mathcal{U}_\mu$ (see Figure 2c). By construction, $\Gamma'_\mu(I) = \langle h_\mu\rangle(I)$ and $\Gamma''_\mu(I) = \langle h_\mu^2\rangle(I) - \langle h_\mu\rangle(I)^2$.

A special case of interest is obtained for the quadratic potential $\mathcal{U}(h)$ in equation 2.2. The joint distribution $P(\mathbf{v}, \mathbf{h})$ in equation 2.1 is gaussian in the hidden-unit values h_μ, and the integration can be done straightforwardly to obtain the marginal distribution over visible configurations, $P(\mathbf{v})$ in equation 2.7. The outcome is

$$P(\mathbf{v}) = \frac{1}{Z} \exp\left(\sum_i \tilde{g}_i(v_i) + \frac{1}{2} \sum_{i,j} \tilde{J}_{ij}(v_i, v_j) \right), \tag{2.8}$$

with

$$\tilde{g}_i(v) = g_i(v) - \frac{\theta}{\gamma} \sum_{\mu} w_{i\mu}(v), \tag{2.9}$$

$$\tilde{J}_{ij}(v, v') = \frac{1}{\gamma} \sum_{\mu} w_{i\mu}(v) w_{j\mu}(v'). \tag{2.10}$$

Hence, the RBM is effectively a Hopfield-Potts model with pairwise interactions between the visible units (Barra, Bernacchia, Santucci, & Contucci, 2012). The number of hidden units sets the maximal rank of the interaction matrix $\tilde{J}$, while the weights vectors attached to the hidden units play the roles of the patterns of the Hopfield-Potts model. In general, for nonquadratic potentials, higher-order interactions between visible units are present. We also note that for quadratic potential, the marginal probability is invariant under rotation of the weights $w_{i\mu}(v) \to \sum_{\mu'} U_{\mu,\mu'} w_{i\mu'}(v)$; therefore, the weights cannot be interpreted separately from each other. In general, such invariance is broken for nonquadratic potential.

2.4 Learning Algorithm. Training is achieved by maximizing the average log likelihood of the data $\langle \log P(\mathbf{v}) \rangle_{\text{data}}$ (see equation 2.7) by stochastic gradient descent (SGD). For a generic parameter θ, the gradient of the likelihood is given by

$$\frac{\partial \langle \log P(\mathbf{v}) \rangle_{\text{data}}}{\partial \theta} = -\left\langle \frac{\partial E_{\text{eff}}(\mathbf{v})}{\partial \theta} \right\rangle_{\text{data}} + \left\langle \frac{\partial E_{\text{eff}}(\mathbf{v})}{\partial \theta} \right\rangle_{\text{RBM}}, \tag{2.11}$$

where $\langle \mathcal{O} \rangle_{\text{data}}$ and $\langle \mathcal{O} \rangle_{\text{RBM}}$ indicate, respectively, the averages over the data and model distributions of an observable $\mathcal{O}$. Evaluating the model averages can be done with Monte Carlo simulations (Ackley et al., 1987; Tieleman, 2008; Desjardins, Courville, & Bengio, 2010; Desjardins, Courville, Bengio, Vincent, & Delalleau, 2010; Salakhutdinov, 2010; Cho, Raiko, & Ilin, 2010), or mean-field-like approximations (Gabrié, Tramel, & Krzakala, 2015; Tramel, Gabrié, Manoel, Caltagirone, & Krzakala, 2017). The gradients of the log-likelihood $\log P(\mathbf{v})$ with respect to the fields $g_i(v)$, couplings $w_{i\mu}(v)$, and hidden-unit potential parameters that we write generically as ξ_μ, therefore read:

$$\frac{\partial \log P}{\partial g_i(v)} = \langle \delta_{v_i,v} \rangle_{\text{data}} - \langle \delta_{v_i,v} \rangle_{\text{RBM}}, \tag{2.12}$$

$$\frac{\partial \log P}{\partial w_{i\mu}(v)} = \langle \delta_{v_i,v}\, \Gamma'_\mu \left(I_\mu(\mathbf{v}) \right) \rangle_{\text{data}} - \langle \delta_{v_i,v}\, \Gamma'_\mu \left(I_\mu(\mathbf{v}) \right) \rangle_{\text{RBM}}, \tag{2.13}$$

$$\frac{\partial \log P}{\partial \xi_\mu} = \left\langle \frac{\partial}{\partial \xi_\mu} \Gamma_\mu \left(I_\mu(\mathbf{v}) \right) \right\rangle_{\text{data}} - \left\langle \frac{\partial}{\partial \xi_\mu} \Gamma_\mu \left(I_\mu(\mathbf{v}) \right) \right\rangle_{\text{RBM}}, \tag{2.14}$$

Here, $\delta_{v_i,v} = 1$ if $v_i = v$, and 0 otherwise denotes the Kronecker function. When the likelihood is maximal, the model satisfies moment-matching equations similar to Potts models. In particular, for a quadratic potential, the average activity Γ'_μ is linear, and equation 2.13 entails that the difference between the second-order moments of the data and model distributions vanishes.

Additionally, there is a formal correspondence between equation 2.13 and other update equations in feature extraction algorithms such as ICA. For instance, in the FastICA (Hyvärinen & Oja, 2000) formulation, the weight update is given by $\Delta w \propto \langle \delta_{v_i,v} \, f\left(I_\mu(\mathbf{v})\right) \rangle_{\text{data}}$, where f is the hidden unit transfer function, followed by an application of the whitening constraint $\langle I_\mu(\mathbf{v}) I_\nu(\mathbf{v}) \rangle_{\text{data}} = \delta_{\mu,\nu}$. In both cases, the first step, which is identical for both methods, drives the weights toward nongaussian features (as long as the transfer function Γ' resp. f is nonlinear), whereas the second one prevents weights from diverging or collapsing onto one another. The gradient of the partition function, which makes the model generative, effectively behaves as a regularization of the weights. One notable difference is that using adaptive dReLU nonlinearities allows us to extract simultaneously both subgaussian (with negative kurtosis such as bimodal distribution) and supergaussian features (i.e., with positive kurtosis such as sparse Laplace distribution), as well as asymmetric ones. In contrast, in FastICA, the choice of nonlinearity biases learning toward either subgaussian or supergaussian distributions. In addition, RBM can be regularized to avoid overfitting by adding to the log-likelihood penalty terms over the fields and the weights. We use the standard L_2 regularization for the fields $\propto \sum_{i,v} g_i(v)^2$, and an L_1/L_2 penalty for the weights

$$\mathcal{R} = \frac{\lambda_1^2}{2Nq} \sum_\mu \left(\sum_{i,v} |w_{i\mu}(v)| \right)^2. \tag{2.15}$$

The latter choice can be explained by writing its gradient $\propto \left(\sum_{i,v} |w_{i\mu}(v)| \right)$ $\text{sign}(w_{i\mu}(v))$: it is similar to the L_1 regularization, with a strength increasing with the weights, hence promoting homogeneity among hidden units. These regularization terms are subtracted from the log-likelihood $\log P(\mathbf{v})$ defined above prior to maximization over the RBM parameters. Some additional practical details of the training algorithm are provided in appendix A. For more insights, interested readers are referred to the review by Fischer and Igel (2012) for an introduction, and to Tubiana et al. (2018) for the evaluation of the performance of the algorithm used in this letter.

3 Nature of the Representation Learned and Weights

3.1 Nature of Representations and Interpretation of the Weights. Taken together, the hidden units define a probability distribution over the visible layer via equation 2.7 that can be used for artificial sample generation, scoring, or Bayesian reconstruction, as well as a representation that can be used for supervised learning. However, one fundamental question is how to interpret the hidden units and their associated weights when taken individually. Moreover, what does the model tell us about the data studied?

To this end, we recall first that a data point $\mathbf{v}$ can be approximately reconstructed from its representation $\mathbf{h}$ via a so-called linear-nonlinear relationship, that is, a linear transformation $I_i(v, \mathbf{h}) = \sum_\mu w_{i\mu}(v)h_\mu$ of the hidden layer activity followed by an element-wise (stochastic) nonlinearity $P(v_i|I_i)$ (see the conditional distribution $P(v_i|\mathbf{h})$ of equation 2.6). A similar reconstruction relationship also exists in other feature extraction approaches; for instance, it is linear and deterministic for PCA/ICA and nonlinear deterministic for autoencoders (see section 5). Owing to this relationship, hidden units may be interpreted as the underlying degrees of freedom controlling the visible configurations. However, such interpretation relies on the constraints imposed on the distribution of $P(\mathbf{h})$, such as decorrelation, independence, or sparse activity. In contrast, in RBM, the marginal $P(\mathbf{h})$ is obtained by integrating over the visible layer (similarly as $P(\mathbf{v})$) and has no explicit constraints. Therefore, hidden units may be interpreted differently depending on the statistics of $P(\mathbf{h})$. We sketch in Figure 3 several possible scenarios for $P(\mathbf{h})$:

- *Prototypical representations* (see Figure 3b). For a given visible layer configuration, about one hidden unit is strongly activated, while the others are weakly activated or silent; each hidden unit responds to a localized region of the configuration state. Owing to the reconstruction relation, the weights attached to the hidden unit are prototypes of data configuration, similar to the centroids obtained with a clustering algorithm.
- *Intricate representations* (see Figure 3c). A visible layer configuration activates weakly many (of order M) hidden units. Each hidden unit is sensitive to a wide region of the configuration space, and different configurations correspond to slightly different and correlated levels of activation of the hidden units. Taken altogether, the hidden units can be very informative about the data, but the individual weights do not have a simple interpretation.[1]

[1] One extreme example of such intricate representation is the random gaussian projection for compressed sensing (Donoho, 2006). Provided that the configuration is sparse in some known basis, it can be reconstructed from a small set of linear projections onto random independent and identically distributed (i.i.d.) gaussian weights $w_{i\mu}$. Although

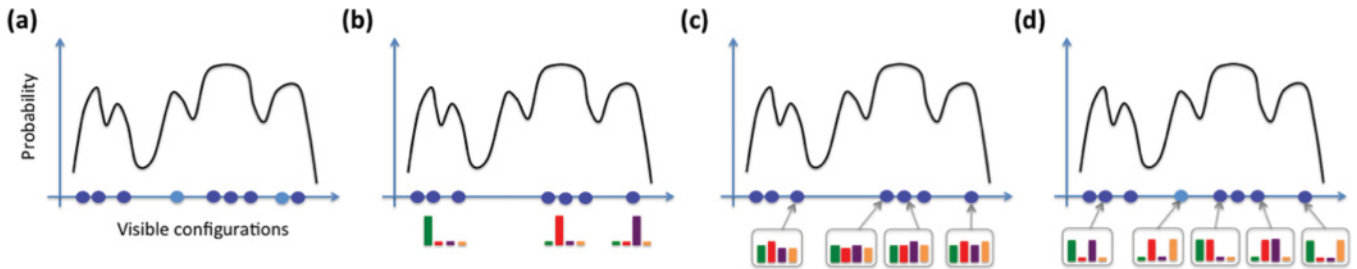

Figure 3: Nature of representations of data with RBM. (a) Data items (dark blue dots) form a very sparse sampling of the high-dimensional distribution (black curve) over visible configurations one wants to reconstruct. Many configurations having high probabilities (light blue dots) are not in the data set. (b) Prototypical representations. Each high-probability portion of the data space activates specifically one hidden unit, while the other units have very weak activations. Here, hidden units are shown by colored bars and their activation levels by the heights of the bars. (c) Intricate representations. All hidden units have similar amplitudes and play similar roles across data space. Visible configurations with high probability are obtained by slightly varying their values. (d) Compositional representations. Representations are composed of elementary features (attached to hidden units), which can be combinatorially composed to create a large diversity of data items all over the space (light blue dot). The same feature can be active in distinct portions of the configuration space.

- *Compositional representations* (see Figure 3d). A substantial number of hidden units (large compared to one but small compared to the size of the hidden layer) are activated by any visible configuration, while the other units are irrelevant. The same hidden unit can be activated in many distinct parts of the configuration space, and different configurations are obtained through combinatorial choices of the strongly activated hidden units. Conversely, the weights correspond to constitutive parts that, once combined, produce a typical configuration. Such compositional representations share similar properties with the ones obtained with sparse dictionaries (Olshausen & Field, 1996; Mairal, Bach, Ponce, & Sapiro, 2009).

Compositional representations offer several advantages with respect to the other types. First, they allow RBM to capture invariances in the underlying distribution from vastly distinct data configurations (contrary to the prototypical regime). Second, representations are sparse (as in the prototypical regime), which makes it possible to understand the relationship between weights and configurations (see the above discussion). Third, activating hidden units of interest (once their weights have been interpreted) allows one to generate configurations with desired properties (see Figure 3d). A key question is whether we can force the RBM to generate such compositional representations.

3.2 Link between Model Parameters and Nature of the Representations. As we will show, it turns out that all three behaviors can arise in RBM. Here, we argue that what determines the nature of the representation are the global statistical properties of the weights, such as their magnitude and sparsity, as well as the number of hidden-units and the nature of their potential. This can be seen from a Taylor expansion of the marginal hidden-layer distribution $P(h)$ for the generated data. For simplicity, we focus on the case where $M = 2$ and the model is purely symmetrical, with $v_i \in \pm 1$, $g_i(1) = g_i(-1) = 0$, $w_{i\mu}(\pm 1) = \pm w_{i\mu}$, $\gamma_\mu^+ = \gamma_\mu^- = 1$, $\theta_\mu^+ = -\theta_\mu^- \equiv \theta_\mu$. For dimensional analysis, we also write p as the fraction of the weights $w_{i\mu}$ that are significantly nonzero, and $\sim W/\sqrt{N}$ their corresponding amplitude. Then the marginal probability over the hidden layer $P(\mathbf{h})$ is given by

$$\log P(\mathbf{h}) = \log \sum_{\mathbf{v}} P(\mathbf{v}, \mathbf{h}) \equiv L(\mathbf{h}) - \log Z$$

$$L(\mathbf{h}) = \sum_{\mu=1,2} -\frac{1}{2} h_\mu^2 - \theta_\mu |h_\mu| + \sum_i \log \cosh \left(\sum_{\mu=1,2} w_{i\mu} h_\mu \right)$$

such representation carries all information necessary to reconstruct the signal, it is by construction unrelated to the main statistical features of the data.

$$= \sum_{\mu=1,2} \underbrace{-\frac{1}{2}h_\mu^2}_{SI} \underbrace{-\theta_\mu|h_\mu|}_{SI/SE} + \underbrace{\frac{1}{2}\left(\sum_i w_{i\mu}^2\right)h_\mu^2}_{SE} \underbrace{-\frac{1}{12}\left(\sum_i w_{i\mu}^4\right)h_\mu^4}_{SI}$$
$$+ \underbrace{\left(\sum_i w_{i1}w_{i2}\right)h_1h_2}_{CE/CI} \underbrace{-\frac{1}{2}\left(\sum_i w_{i1}^2 w_{i2}^2\right)h_1^2h_2^2}_{CI}$$
$$\underbrace{-\frac{1}{3}\left(\sum_i w_{i1}^3 w_{i2}\right)h_1^3h_2 - \frac{1}{3}\left(\sum_i w_{i1}w_{i2}^3\right)h_1^3h_2}_{CE/CI}, \tag{3.1}$$

where we have discarded higher-order terms. This expansion shows how hidden units effectively interact with one another via self-excitation (SE), self-inhibition (SI), cross-excitation (CE), or cross-inhibition (CI) terms. Some key insights are that:

- Large h values arise via self-excitation, with a coefficient in $\sum_i w_{i,\mu}^2 \sim pW^2$.
- The dReLU threshold acts either as a self-inhibition that can suppress small activities ($\theta_\mu > 0$) or as self-excitation that enhances them.
- Hidden unit interactions (hence, correlations), arise via their overlaps $\sum_i w_{i\mu}w_{i\mu} \sim p^2W^2$.
- The nongaussian nature of the visible units induces an effective inhibition term between each pair of hidden units, whose magnitude $\sum_i w_{i\mu}^2 w_{i\mu'}^2 \sim \frac{p^2W^4}{N}$ essentially depends on the overlap between the supports of the hidden units. We deduce that the larger the hidden layer, the stronger this high-order inhibition, and that RBMs with sparse weights (order p nonzero coefficients) have significantly weaker overlaps (of order p^2), and therefore have much less cross-inhibition.

Depending on the parameter values, several behaviors can therefore be observed. Intuitively, the (1) prototype, (2) intricate, and (3) compositional representation correspond to the cases where, (1) respectively, strong self-excitation and strong cross-inhibition result in a winner-take-all situation where one hidden unit is strongly activated and inhibits the others; (2) cross-inhibition dominates, and all hidden units are weakly activated; and (3) strong self-excitation and weak cross-inhibition (e.g., due to sparse weights) allow for several hidden units to be simultaneously strongly activated.

Though informative, the expansion is valid only for small W/small h, whereas hidden units may take large values in practice. A more elaborate mathematical analysis is therefore required and is presented next.

3.3 Statistical Mechanics of Random-Weights RBM. The different representational regimes shown in Figure 3 can be observed and characterized analytically in simple statistical ensembles of RBMs controlled by a few structural parameters (Tubiana & Monasson, 2017):

- The aspect ratio $\alpha = \frac{M}{N}$ of the RBM
- The threshold $\theta_\mu = \theta$ of the ReLU potential acting on hidden units
- The local field $g_i = g$ acting on visible units
- The sparsity p of the weights $w_{i\mu}$ independently and uniformly drawn at random from the distribution (Agliari, Barra, Galluzzi, Guerra, & Moauro, 2012):[2]

$$w_{i\mu} = \begin{cases} 0 & \text{with probability} \quad 1 - \dfrac{p}{2}, \\ \dfrac{W}{\sqrt{N}} & \text{with probability} \quad \dfrac{p}{2}, \\ -\dfrac{W}{\sqrt{N}} & \text{with probability} \quad \dfrac{p}{2}. \end{cases} \tag{3.2}$$

Depending on the values of these parameters, the RBM may operate, at low temperature, for $W^2 p \gg 1$, in one of the three following phases when the size N becomes very large. In the *ferromagnetic phase*, one hidden unit, say, $\mu = 1$, receives a strong input,

$$I_1 = \sum_i w_{i1} \langle v_i \rangle, \tag{3.3}$$

of the order of $\sqrt{N}$. The other hidden units receive small (on the order of ± 1) inputs and can be silenced by an appropriate finite value of the threshold θ. Hidden unit 1 is in the linear regime of its ReLU activation curve and therefore has value $h_1 = I_1$. In turn, each visible unit receives an input on the order of $h_1 \times w_{i1}$ from the strongly activated hidden unit. Due to the nonlinearity in the activation curve of ReLU and the presence of the threshold θ, most of the other hidden units are silent and do not send noisy inputs to the visible units. Hence, visible unit configurations $\{v_i\}$ are strongly magnetized

[2] Here p is assumed to be finite, such that each hidden unit receives input from a large number pN of visible units. Alternatively, one could impose finite connectivity $c = pN$, that is, p of the order of $1/N$ (Agliari, Annibale, Barra, Coolen, & Tantari, 2013). We evaluated p and pN for a set of 38 RBM trained on various protein families with different lengths and found that p was more stable (see Figure S8 in).

along the pattern $\{w_{i1}\}$. This phase is the same as the one found in the Hopfield model at sufficiently small loads α (Amit, Gutfreund, & Sompolinsky, 1985).

In the *spin-glass phase*, if the aspect ratio α of the machine is too large and the threshold θ and the visible field g are too small, the RBM enters the spin-glass phase. All inputs I_μ to the visible units are of the order of ± 1. Visible units are also subject to random inputs I_i of the order of unity, and are hence magnetized as expected in a spin glass.

The third phase is the *compositional phase*. At large enough sparsity, that is, for small enough values of p (see equation 3.2), a number $L \sim \frac{\ell}{p}$ of hidden units have strong magnetizations on the order of $m \sim p \times \sqrt{N}$, the other units being shut down by choices of the threshold $\theta \sim \sqrt{p}$. Notice that L is large compared to 1 but small compared to the hidden-layer size, M. The value of ℓ is determined through minimization of the free energy (Tubiana & Monasson, 2017). The input I_i onto visible units is on the order of $m \times w \times L \sim p$. The mean activity in the visible layer is fixed by the choice of the visible field $g \sim p$. This phase requires low temperatures, that is, weight-squared amplitudes $W^2 \sim \frac{1}{p}$ at least. Mathematically speaking, these scalings are obtained when the limit $p \to 0$ is taken after the thermodynamic limit $N, M \to \infty$ (at fixed ratio $\alpha = M/N$).

Although RBMs learned from data do not follow the same simplistic weight statistics and deviations are expected, the general idea that each RBM learns a decomposition of samples into building blocks was observed on MNIST (Hinton, 2002; Fischer & Igel, 2012), a celebrated data set of handwritten digits, and is presented hereafter on in silico protein families.

3.4 Summary. To summarize, RBMs with nonquadratic potential and sparse weights, as enforced by regularization, learn compositional representations of data. In other words, enforcing sparsity in weights results in enforcing sparsity in activity in a fashion similar to that of sparse dictionaries. The main advantages of RBM with respect to the latter are that unlike sparse dictionaries, RBM defines a probability distribution—therefore allowing sampling, scoring, and Bayesian inference—and the representation is a simple linear-nonlinear transformation instead of the outcome of an optimization.

Importantly, we note that the random-RBM ensemble analysis shows only that sparsity is a sufficient condition for building an energy landscape with a diversity of gradually related attractors; such a landscape could also be achieved in other parameter regions—for example, when the weights are correlated. Therefore, there is no guarantee that training an RBM without regularization on a compositional data set (e.g., generated by a random-RBM in the compositional phase) will result in sparse weights and a compositional representation.

Since we can enforce a compositional representation via regularization, what have we learned about the data? The answer is that enforcing sparsity does not come at the same cost in likelihood for all data sets. In some cases, for example, for data constituted by samples clustered around prototypes, weight sparsity yields a significant misfit with respect to the data. In other cases, such as lattice proteins, presented below, enforcing sparsity comes at a very weak cost, suggesting that these data are intrinsically compositional. More generally, RBM trained with fixed regularization strength on complex data sets may exhibit heterogeneous behaviors, with hidden units behaving as compositional and others as prototypes (see some examples in Tubiana et al., 2018).

4 Results

4.1 Lattice Proteins: Model and Data. Lattice protein (LP) models were introduced in the 1990s to investigate the properties of proteins (Shakhnovich & Gutin, 1990), in particular how their structure depend on their sequence. They were recently used to benchmark graphical models inferred from sequence data (Jacquin, Gilson, Shakhnovich, Cocco, & Monasson, 2016). In the version considered here, LP include 27 amino acids and fold on a $3 \times 3 \times 3$ lattice cube (Shakhnovich & Gutin, 1990). There are $\mathcal{N} = 103{,}406$ possible folds (up to global symmetries), that is, self-avoiding conformations of the 27 amino acid–long chains on the cube.

Each sequence $\mathbf{v} = (v_1, v_2, \ldots, v_{27})$ is assigned the energy

$$\mathcal{E}(\mathbf{v}; S) = \sum_{i<j} c_{ij}^{(S)} \, \epsilon(v_i, v_j), \tag{4.1}$$

when it is folded in structure S. In this equation, $c^{(S)}$ is the contact map of S. It is a 27×27 matrix, whose entries are $c_{ij}^{(S)} = 1$ if the pair of sites ij is in contact in S, that is, if i and j are nearest neighbors on the lattice and $c_{ij}^{(S)} = 0$ otherwise. The pairwise energy $\epsilon(v_i, v_j)$ represents the physicochemical interaction between the amino acids v_i and v_j when they are in contact. Its value is given by the Miyazawa-Jernigan (MJ) knowledge-based potential (Miyazawa & Jernigan, 1996).

The probability that the protein sequence $\mathbf{v}$ folds in one of the structures, say, S, is

$$P_{nat}(\mathbf{v}; S) = \frac{e^{-\mathcal{E}(\mathbf{v};S)}}{\sum_{S'=1}^{\mathcal{N}} e^{-\mathcal{E}(\mathbf{v};S')}}, \tag{4.2}$$

where the sum at the denominator runs over all $\mathcal{N}$ possible structures, including the native fold S. We consider that the sequence $\mathbf{v}$ folds in S if $P_{nat}(\mathbf{v}; S) > 0.995$.

A collection of 36,000 sequences $\mathbf{v}$ that specifically fold in structure S_A, such that $P_{nat}(\mathbf{v}; S_A) > 0.995$, were generated by Monte Carlo simulations as described in Jacquin et al. (2016). As real MSA, lattice-protein data feature nonconserved and partly conserved sites, short- and long-range correlations between amino acids on different sites, as well as high-order correlations that arise from competition between folds (Jacquin et al., 2016; see Figure 4). The correlations reflect the main modalities of amino acid interactions, such as electrostatic attraction/repulsion, hydrophobic attraction, or cysteine-cysteine disulfide bonds (see Figure 4f). Moreover, as for natural proteins, LP sequences can be partitioned into subfamilies (see Figure 4g).

4.2 Representations of Lattice Proteins by RBM: Interpretation of Weights and Generative Power. We now learn RBM from the LP sequence data, with the training procedure presented in section 2.4. The visible layer includes $N = 27$ sites, each carrying Potts variables v_i taking 20 possible values. Here, we show results with $M = 100$ dReLU hidden units, with a regularization $\lambda_1^2 = 0.025$, trained on a fairly large alignment of $B = 36{,}000$ sequences. We present in Figure 5 a selection of structural LP features inferred by the model (see Tubiana et al., 2018 for more features). For each hidden unit μ, we show in panel a the weight logo of $w_{i\mu}(v)$ and in panel b the distribution of its hidden unit input I_μ, as well as the conditional mean activity $\langle h_\mu \rangle (I_\mu)$. Weights have significant values on a limited number of sites only, which makes their interpretation easier.

As seen from Figure 5a, weight 1 focuses mostly on sites 3 and 26, which are in contact in the structure (see Figure 4a) and are not very conserved (see the sequence logo in Figure 4d). Positively charged residues (H,R,K) have a large, positive (resp. negative) component on site 3 (resp. 26), and negatively charged residues (E, D) have a large, negative (resp. positive) components on the same sites. The histogram of the input distribution in Figure 5b shows three main peaks in the data. Since $I_1(v) = \sum_i w_{i1}(v_i)$, the peaks (1) $I_1 \sim 3$, (2) $I_1 \sim -3$, and (3) $I_1 \sim 0$ correspond to sequences having, respectively, (1) positively charged amino acids on site 3 and negatively charged amino acids on site 26, (2) conversely, negatively charged amino acids on site 3 and positively charged on site 26, and (3) identical charges or noncharged amino acids. Weight 2 also focuses on sites 3 and 26. Its positive and negative components correspond respectively to charged (D, E, K, R, H) or hydrophobic amino acids (I, L, V, A, P, W, F). The bulk in the input distribution around $I_2 \sim -2$ therefore identifies sequences having hydrophobic amino acids at both sites, whereas the positive peak corresponds to electrostatic contacts as the ones shown in weight 1. The presence of hidden units 1 and 2 signals an excess of sequences having significantly high $|I_1|$ or $|I_2|$ compared to an independent-site model. Indeed, the contribution of hidden

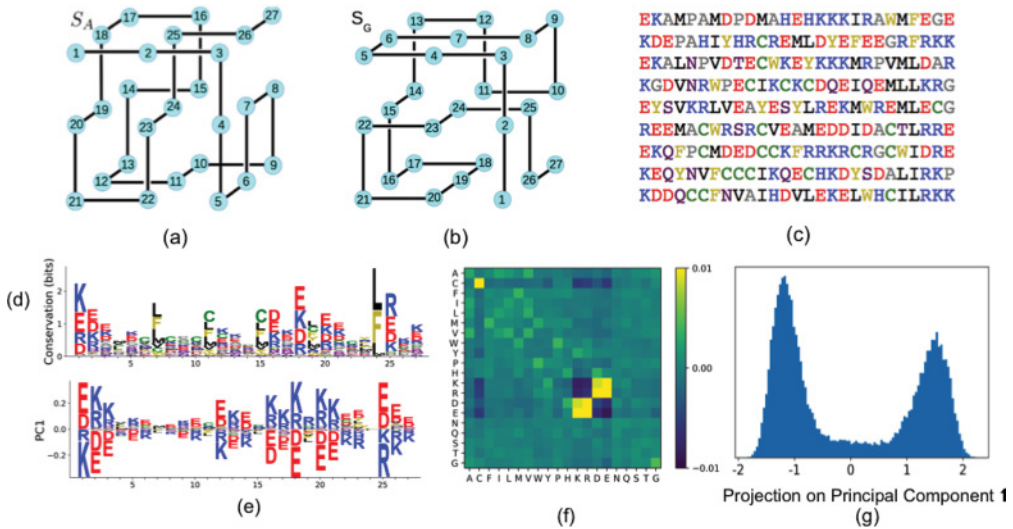

Figure 4: Main features of lattice protein (LP) sequence alignments. Two examples of structures, S_A and S_G, are shown in, respectively, panels a and b. (c) A subset of the 36,000 sequences generated by Monte Carlo that fold specifically in S_A ($p_{nat} > 0.995$), that is, with much lower energy when folded in (a) S_A than in other structures, such as (b) S_G. Color code for amino acids: red = negative charge (E, D), blue = positive charge (H, K, R), purple = noncharged polar (hydrophilic) (N, T, S, Q), yellow = aromatic (F, W, Y), black = aliphatic hydrophobic (I, L, M, V), green = cysteine (C), gray = other, small (A, G, P). (d) Position weight matrix of the MSA: for each site i with observed amino acid frequency $f_i(\upsilon)$, the total height denotes the conservation score, $\log 20 + \sum_\upsilon f_i(\upsilon) \log f_i(\upsilon)$, and the height of each letter is proportional to $f_i(\upsilon)$. (e) Weight logo of the first principal component, PC1; the heights of letters are proportional to the corresponding principal component coefficients, $w_i^{\mathrm{PC1}}(\upsilon)$. Positive and negative coefficients are, respectively, above and below the 0 axis. (f) Average covariance between pairs of sites in contact, $C(\upsilon, \upsilon') = \frac{1}{28} \sum_{i,j} c_{ij}^{(S)} \left(f_{ij}(\upsilon, \upsilon') - f_i(\upsilon) f_j(\upsilon') \right)$. (g) Histogram of the projections of sequences along PC1.

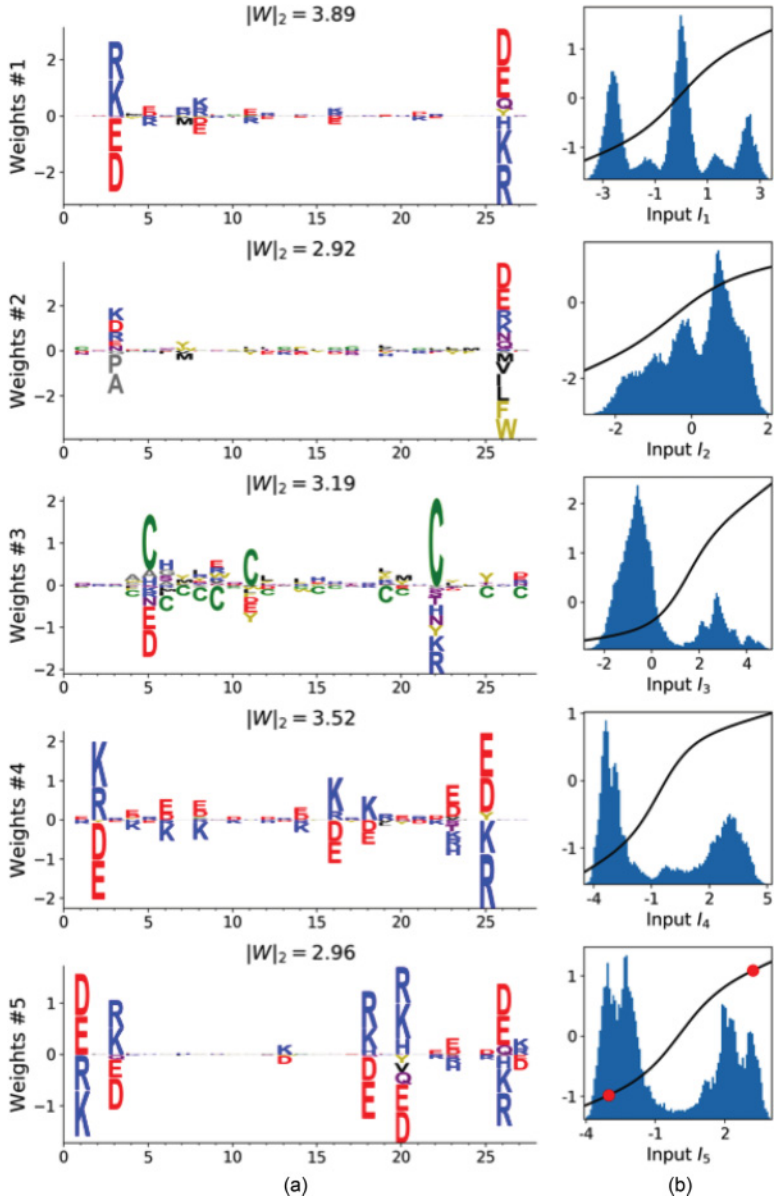

Figure 5: Modeling lattice proteins with RBM. (a) Five weight logos, visualizing the weights $w_{i\mu}(v)$ attached to five selected hidden units. (b) Distribution of inputs received by the corresponding hidden units, and conditional mean activity (full line and left scale).

unit $\mu = 1, 2$ to the log probability is $\Gamma(I_\mu) \sim I_\mu^2$, since the conditional mean $\langle h_\mu \rangle(I_\mu) = \Gamma'(I_\mu)$ is roughly linear (see Figure 5b). In other words, sequences folding into S_A are more likely to have complementary residues on sites 3 and 26 than what would be predicted from an independent site model (with the same site frequencies of amino acids).

Interestingly, RBM can extract features involving more than two sites. Weight 3 is located mainly on sites 5 and 22, with weaker weights on sites 6, 9, and 11. It codes for a cysteine-cysteine disulfide bridge located on the bottom of the structure and present in about a third of the sequences ($I_3 \sim 3$). The weak components and small peaks $I_3 \sim 4$ also highlight sequences with a triangle of cysteines 5-11-22 (see S_A). We note, however, that this is an artifact of the MJ-based interactions in lattice proteins (see equation 4.1) as a real cysteine amino acid may form only one disulfide bridge.

Weight 4 is an extended electrostatic mode. It has strong components on sites 23, 2, 25, 16, and 18 corresponding to the upper side of the protein (see Figure 4a). Again, these five sites are contiguous on the structure, and the weight logo indicates a pattern of alternating charges present in many sequences ($I_4 \gg 0$ and $I_4 \ll 0$).

The collective modes defined by RBM may not be contiguous on the native fold. Weight 5 codes for an electrostatic triangle 20-1-18 and the electrostatic contact 3-26, which is far away from the former. This indicates that despite being far away, sites 1 and 26 often have the same charge. The latter constraint is not due to the native structure S_A but impedes folding in the competing structure, S_G, in which sites 1 and 26 are neighbors (see Figure 4b). Such negative design was also reported through analysis with pairwise models (Jacquin et al., 2016).

4.3 Lattice Protein Sequence Design. The interpretability of the hidden units allows us to design new sequences with the right fold. We show in Figure 6a an example of conditional sampling, where the activity of one hidden unit (here, h_5) is fixed. This allows us to design sequences having either positive or negative I_5, depending on the sign of h_5—hence having given charges on a subset of five residues. In this example, biasing can be useful—for example, to find sequences with prescribed charge distribution. More broadly, Tubiana et al. (2018) reported that hidden units can have a structural or functional role, for example, in terms of loop size, binding specificity, or allosteric interaction. In principle, conditional sampling allows one to tune these properties at will. In Figure 6b, the sequences generated have both low sequence similarity to the sequences used in training, with about 40% sequence difference to the closest sequence in the data, and high probability P_{nat} (see equation 4.2) to fold into S_A. Interestingly, low temperature sampling for example, sampling from $P'(\mathbf{v}) \propto P(\mathbf{v})^2$, can produce sequences with higher P_{nat} than all the sequences used for training. In real protein design setups, this could correspond to higher binding affinity or better compromises between different target functions.

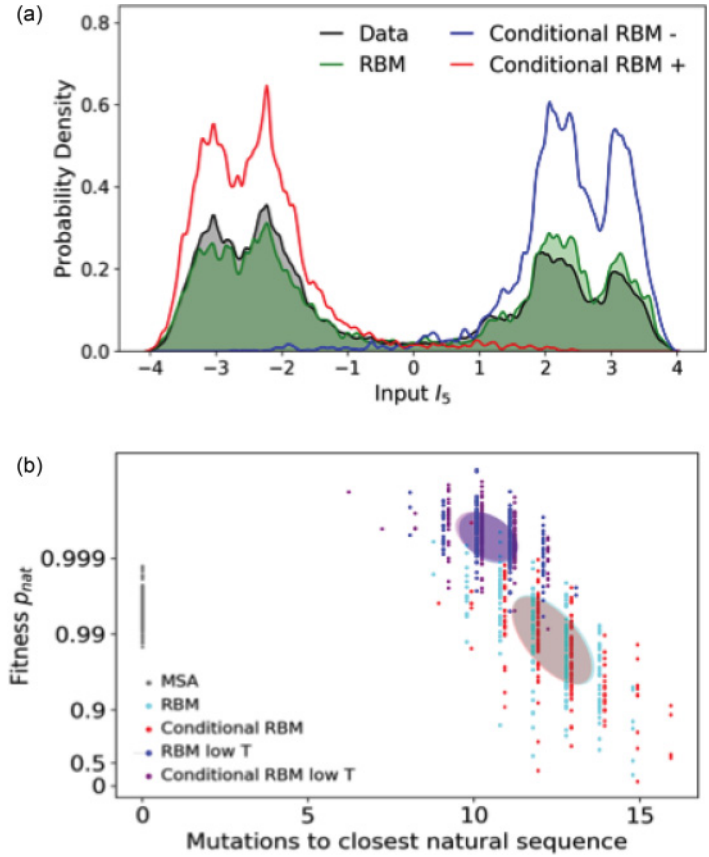

Figure 6: Sequence generation with RBM. (a) Distribution of hidden unit input I_5, for original sequences (black), RBM-generated sequences (green), and conditional RBM-generated sequences (red, blue). The last are obtained by sampling while fixing the activity of h_5 to one of the two values shown in red in Figure 5b. (b) Scatter plot of the number of mutations to the closest natural sequence versus probability of folding into S_A, for natural (gray) and artificial (colored) LP sequences. Sequences are generated using regular RBM sampling, conditional RBM sampling, or low temperature sampling (see Tubiana et al., 2018). The nonlinear scale $y = -\log(1 - P_{nat})$ was used; the ellipsis indicates the best-fitting gaussian.

4.4 Representations of Lattice Proteins by RBM: Effect of Sparse Regularization. We now repeat the previous experiment with varying regularization strength, as well as for various potentials. To see the effect of the regularization on the weights, we compute a proxy p for the weight sparsity

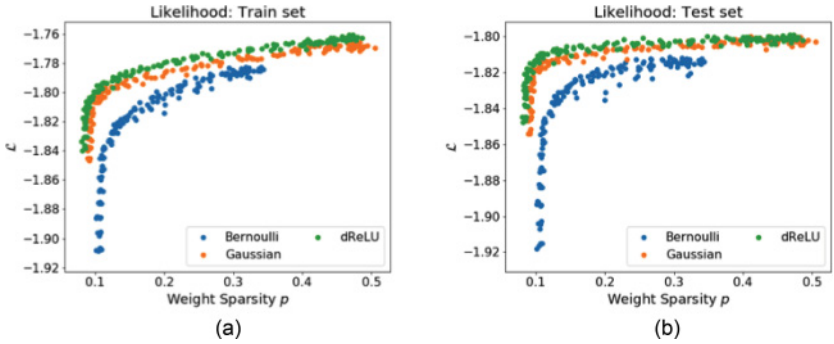

Figure 7: Sparsity-performance trade-off on lattice proteins. Scatter plots of log likelihood versus weight sparsity for the (a) training and (b) test sets. Each point corresponds to an RBM trained with a different regularization parameter.

(see the definition in appendix B). We also evaluate the likelihood $\mathcal{L}$ of the model on the train and a held-out test set. To do so, we estimate the partition function via the annealed importance sampling algorithm (Neal, 2001; Salakhutdinov & Murray, 2008). We show in Figure 7 the sparsity-likelihood scatter plot.

Without sparse regularization, the weights are not sparse and the representation is intricate. As expected, increasing the regularization strength results in fewer nonzero weights and lower likelihood on the training set. Somewhat surprisingly, the test set likelihood decays only mildly for a while, even though the data set is very large such that no overfitting is expected. This suggests a large degeneracy of maximum likelihood solutions that perform equally well on test data; sparse regularization allows us to select one for which the representation is compositional and the weights can be easily related to the underlying interactions of the model. Beyond some point, likelihood decreases abruptly because hidden units cannot encode all key interactions anymore. Choosing a regularization strength such that the model lies at the elbow of the curve, as was done for the RBM in Figure 5, is a principled way to obtain a model that is both accurate and interpretable.

4.5 Representations of Lattice Proteins by RBM: Effects of the Size of the Hidden Layer. We now study how the value of M affects the representations of the protein sequences. We repeat the training with dReLU RBM fixed regularization $\lambda_1^2 = 0.025$ for M varying between 1 and 400 and show in Figure 8 one typical weight $\mathbf{w}_\mu$ learned by dReLU RBMs for M varying between 1 and 400. We observe different behaviors as the value of M increases.

For very low M, the weight vectors are extended over the whole visible layer. For $M = 1$, the unique weight vector captures electrostatic

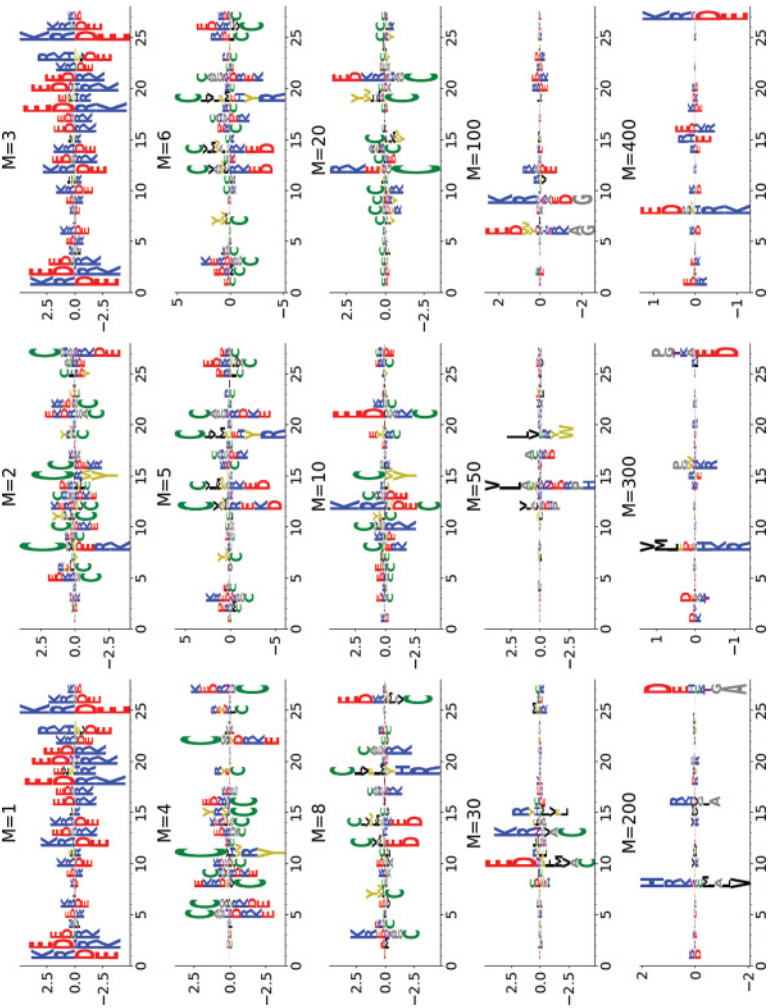

Figure 8: Typical features learned by RBMs on lattice proteins for $M \in [1, 400]$ with fixed regularization strength $\lambda_1^2 = 0.025$. For each RBM, one typical feature is shown, with sparsity p_μ closest to the median sparsity p. See the definition of p_μ in equation B.2.

features—its entries are strong on charged amino acids, such as K, R (positive charges) and E, D (negative charges)—and is very similar to the top component of the correlation matrix of the data (compare the top panel in Figure 8 and Figure 4f). Additional hidden units (see the panels on the second line of Figure 8 corresponding to $M = 2$) capture other collective modes—here patterns of correlated cysteine-cysteine bridges across the protein. Hence, RBM can be seen as a nonlinear implementation of PCA. A thorough comparison with PCA will be presented in section 5.

As M increases, the resolution of representations gets finer and finer: units focus on smaller and smaller portions of the visible layer (see Figure 8). Nonzero entries are restricted to a few sites, often in contact on the protein structure (see Figure 4, fold S_A). We introduce a proxy p for the sparsity of the weight based on inverse participation ratios of the entries $w_{i\mu}(v)$ (see appendix B). The behavior of p as a function of M is shown in Figure 9a. We observe that p decreases (weights get sparser) until M reaches 50 to 100.

For values of $M > 100$, the modes of covariation cannot be decomposed into thinner ones anymore, and p saturates to the minimal value $\sim 2/27$, corresponding to a hidden unit linked to two visible units (see Figure 9a). Then additional features are essentially duplicates of the previous ones. This can be seen from the distribution of maximum weight overlaps $O_\mu^{\max}$ shown in Figure 9a (see appendix B for a definition).

In addition, the number L of simultaneously active hidden units per sequence grows, undergoing a continuous transition from a ferromagnetic-like regime ($L \sim 1$) to a compositional one ($L \gg 1$). Note crucially that L does not scale linearly with M (see Figures 9a and 9b). If we account for duplicated hidden units, L tends to saturate at about 12, a number that arises from the data distribution rather than from M. In contrast, for an unregularized training with quadratic hidden unit potential, L grows to much larger values ($L \sim 50$) as M increases (see Figure 13). Finally, Figure 9c shows that the theoretical scaling $L \sim \frac{1}{p}$ (Tubiana & Monasson, 2017) is qualitatively correct.

5 Comparison with other Representation Learning Algorithms

5.1 Models and Definitions. We now compare the results obtained with standard representation learning approaches. Thanks to popular packages such as Scikit-learn (Buitinck et al., 2013) or Keras (Chollet, 2015), most of these approaches are easy to implement in practice.

Principal component analysis (PCA) is arguably the most commonly used tool for finding the main modes of covariation of data. It is routinely applied in the context of protein sequence analysis for finding protein subfamilies, identifying specificity-determining positions (Casari, Sander, & Valencia, 1995; Rausell, Juan, Pazos, & Valencia, 2010; De Juan, Pazos, & Valencia, 2013), and defining subsets of sites (called sectors) within proteins that control the various properties of the protein (Halabi, Rivoire,

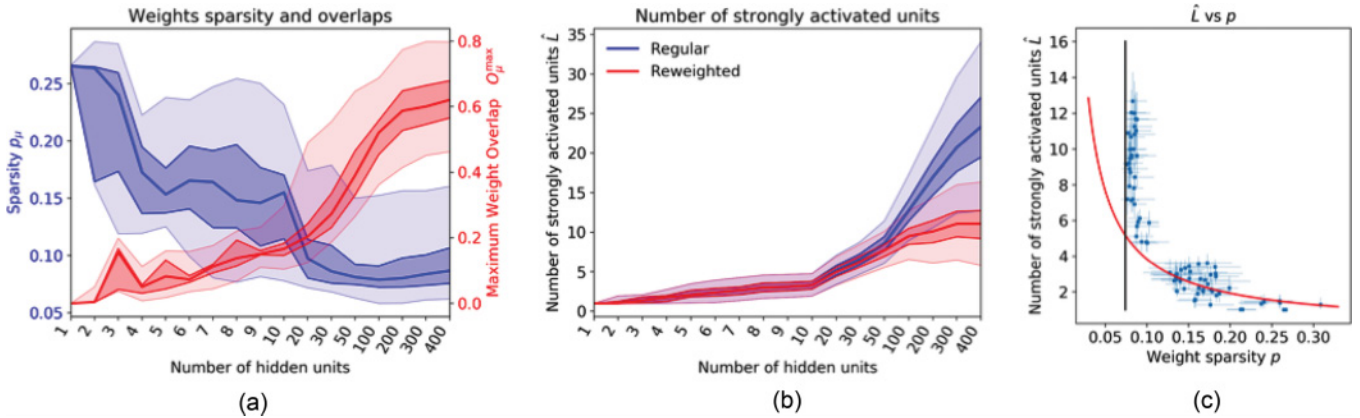

Figure 9: (a) Evolution of the distribution of weights sparsity p_μ and maximum overlaps $O_\mu^{\max}$ as functions of the number M of hidden units. (b) Evolution of the number of strongly activated hidden units as a function of the number M of hidden units. Solid lines indicate median, and colored area indicates 33%, 66% (dark) and 10%, 90% quantiles (light). (c) Scatter plot of the number of strongly activated hidden units against weight sparsity. The vertical bar locates the minimal sparsity value $p = 2/27$. See the text.

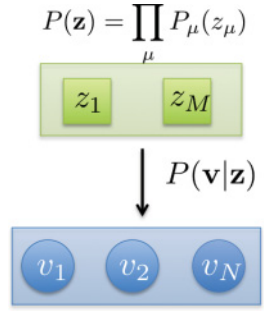

Figure 10: Directed latent variable model. Latent variables are first drawn independently from one another. Then a configuration is sampled from $P(\mathbf{v}|\mathbf{z})$. See section 5.1.

Leibler, & Ranganathan, 2009; McLaughlin, Poelwijk, Raman, Gosal, & Ranganathan, 2012). A major drawback of PCA is that it assumes that the weights are orthogonal, which is in general not true and often results in extended modes that cannot be interpreted easily. Independent component analysis (ICA; Bell & Sejnowski, 1995; Hyvärinen & Oja, 2000) is another approach that aims at alleviating this issue by incorporating high-order moments statistics into the model. ICA was applied for identifying neural areas from fMRI data (McKeown et al., 1998) and also used for protein sequence analysis (Rivoire et al., 2016). Another way to break the rotational invariance is to impose sparsity constraints on the weights or the representation via regularization. Sparse PCA (Zou, Hastie, & Tibshirani, 2006) is one such popular approach, and was considered in both neuroscience (Baden et al., 2016) and protein sequence analysis (Quadeer, Morales-Jimenez, & McKay, 2018). We will also study sparse (in weights) single-layer noisy autoencoders, which can be seen as a nonlinear variant of sparse PCA. Sparse dictionaries (Olshausen & Field, 1996; Mairal et al., 2009) are also considered. Finally, we will consider variational autoencoders (Kingma & Welling, 2013) with a linear-nonlinear decoder, which can be seen as both regularized autoencoder and a nonlinear generative PCA. VAE were recently considered for generative purpose in proteins (Sinai, Kelsic, Church, & Novak, 2017; Riesselman, Ingraham, & Marks, 2017; Greener, Moffat, & Jones, 2018), and their encoder defines a representation of the data.

All the above-mentioned models belong to the same family: the linear-nonlinear latent variable graphical models (see Figure 10). In this generative model, latent factors z_μ are drawn from an independent distribution $P(\mathbf{z}) = \prod_\mu P_\mu(z_\mu)$, and the data are obtained, as in RBM, by a linear transformation followed by an element-wise nonlinear stochastic or deterministic transformation, of the form $P(\mathbf{v}|\mathbf{z}) = \prod_i P(v_i|I_i^v(\mathbf{z}))$. Unlike RBM, a single pass, rather than an extensive back-and-forth process, is sufficient to sample

configurations. For a general choice of $P_\mu(z_\mu)$ and $P(\mathbf{v}|\mathbf{z})$, the inference of this model by maximum likelihood is intractable because the marginal $P(\mathbf{v})$ does not have a closed form. The different models correspond to different hypotheses on $P_\mu(z_\mu)$, $P\left(v_i|I_i^v(\mathbf{z})\right)$, and learning principles that simplify the inference problem (see Table 1).

5.2 Lattice Proteins: Features Inferred. We use each approach to infer features from the MSA lattice proteins. For each model, we use the same number $M = 100$ of latent factors. For PCA, ICA, and sparse dictionaries, we used one-hot-encoding for each site (i.e., converted into a 20-dimensional binary vector), and the standard implementation from Scikit-learn with default parameters. For sparse dictionaries, we adjusted the penalty such that the number L of simultaneously active latent features matches the one found in RBM—approximately 10. Sparse PCA and sparse autoencoders are special cases of autoencoders, respectively, with a mean square reconstruction error and categorical cross-entropy reconstruction error, and were implemented in Keras (Chollet, 2015). In both cases, we used the same weights for encoding and decoding. For the noisy variant, we replace 20% of the amino acids with a random one before attempting to reconstruct the original sequence. For variational autoencoders, we used the same parametric form of $P(\mathbf{v}|\mathbf{z})$ as for RBM—a linear transformation followed by a categorical distribution. The posterior probability $P(\mathbf{z}|\mathbf{v})$ is approximated by a gaussian encoder $\mathcal{N}\left(\mu(\mathbf{v}), \sigma^2(\mathbf{v})\right)$ where $\mu(\mathbf{v})$ and $\log\sigma^2(\mathbf{v})$ are computed from $\mathbf{v}$ either by linear transformation or by a single hidden layer neural network. The parameters of both $P(\mathbf{v}|\mathbf{z})$ and the encoder are learned by maximizing the variational lower bound on the likelihood of the model. The sparse regularization is enforced on the decoding weights of $P(\mathbf{v}|\mathbf{z})$ only. For all models with a sparse weight regularization, we selected the regularization strength so as to obtain similar median sparsity values p as with the RBM shown in Figure 5. We show for each method six selected features in Figure 11 (PCA, sPCA, sNAE, sVAE with linear encoder), and in Figure 14 (ICA, sparse dictionaries, sVAE with nonlinear encoder). For PCA, ICA, and sparse dictionaries, we show the most important features in terms of explained variance. For sPCA, sNAE, and sVAE, we show features with sparsity p_μ close to the median sparsity.

Most of the high-importance features inferred by PCA and ICA are completely delocalized and encode the main collective modes of the data, similar to unregularized RBM. Clearly, there is no simple way to relate these features to the underlying interactions of the system. Even if sparsity could help, the main issue is the enforced constraint of decorrelation or independence, because it impedes the model from inferring smaller coevolutionary modes such as pairs of amino acids in contact. Indeed, lattice proteins exhibits a hierarchy of correlations, with sites that are tightly correlated and also weakly correlated to others. For instance, hidden units 4 and 5 of Figure 5 are strongly but not completely correlated, with a small fraction of

Table 1: Similarities and Differences between Various Feature Extraction Approaches.

Algorithm	PCA	ICA (Infomax)	Sparse PCA	Sparse Noisy Autoencoders	Sparse Dictionaries	Variational Autoencoders	RBM
$P_\mu(z_\mu)$	Gaussian	Nongaussian	/	/	Sparse	Gaussian	Nongaussian; correlated
M	$\leq N$	$= N$	$\lesseqgtr N$	$\lesseqgtr N$	$\lesseqgtr N$	$\lesseqgtr N$	$\lesseqgtr N$
$P(v_i\|I_i^p)$	Deterministic	Deterministic	Deterministic	Deterministic	Deterministic	Stochastic	Stochastic
$P(\mathbf{z}\|\mathbf{v})$	Deterministic	Deterministic	Deterministic	Stochastic	Deterministic	Stochastic	Stochastic
$\langle v_i \rangle\ (I_i^p)$	Linear	Linear	Linear	Softmax	Linear	Softmax	Softmax
$w_{i\mu}(v)$	Orthonormal	Normalized	Sparse	Sparse	Normalized	Sparse	Sparse
Learning method	Max. likelihood $\Leftrightarrow$ Min. reconstruction error (MSE)	Max. likelihood	Min. reconstruction error (MSE)	Min. Noisy Reconstruction Error (CCE)	Min. reconstruction error (MSE)	Variational max. likelihood	Max. likelihood

Note: MSE = mean square error. CCE: categorical cross-entropy.

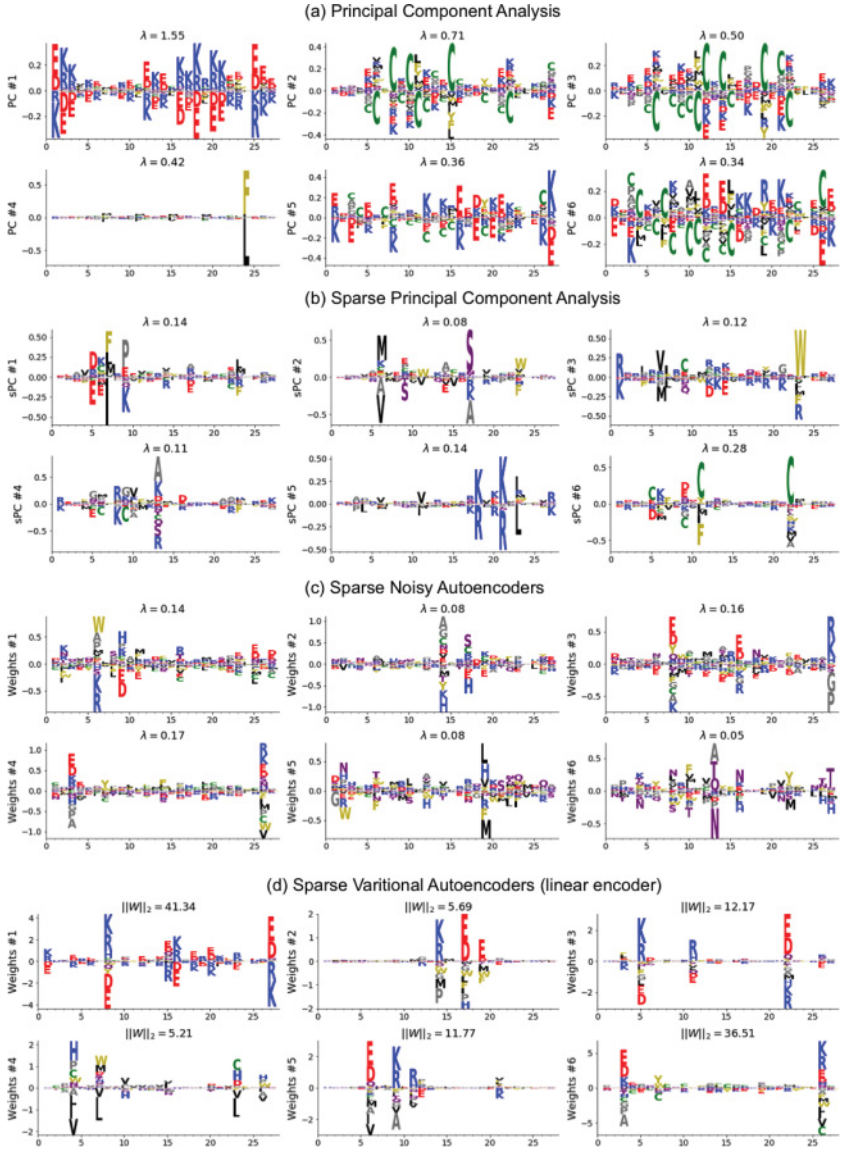

Figure 11: Six weights attached to latent factors for (a) principal component analysis, (b) sparse principal component analysis, (c) sparse noisy autoencoders, and (d) sparse variational autoencoders with linear encoding. Weights are selected as follows: (a) The first six principal components. (b–d) Selected nonzero weights with weight sparsity p_μ close to the median value. Values above the components indicate the feature importance, measured by either the latent factor variance λ (when the weights are normalized) or the weight norm $\|W\|_2$ (when the variances are normalized).

Table 2: Number of Sparse Features Extracted by the Various Approaches Localized on the Structure S_A.

Sparse Model	Pair Features	Triplet Features
RBM	47/49	16/18
sPCA	15/25	3/24
Noisy sAE	20/29	7/19
sVAE (linear encoder)	8/8	5/6
sVAE (nonlinear encoder)	6/6	2/4

sequences having $I_4 > 0$ and $I_5 < 0$ and conversely (see Figure 16a). Therefore, neither PCA nor ICA can resolve the modes; instead, the first principal component is roughly the superposition of both modes (see Figure 4e). Finally, both PCA and ICA also infer a feature that focuses only on site 24 (PC4 and IC4). As seen from the conservation profile (see Figure 4d), this site is strongly conserved, with two possible amino acids. Since mutations are uncorrelated from the remainder of the sequence and the associated variance of the mode is fairly high, both PCA and ICA encode this site. In contrast, we never find a hidden unit focusing only on a single site with RBM because its contribution $\Gamma_\mu(I_\mu(v))$ to $P(v)$ would be equivalent to a field term $g_i(v_i)$. Similar features are found with sparse dictionaries as well (see Figure 14).

As expected, all the models with sparse weights penalties (sPCA, sNAE, and sVAE) can infer localized features, provided the regularization is adjusted accordingly. However, unlike in RBM, a significant fraction of these features does not focus on sites in contact. For instance, in sparse PCA, features 2 and 5 focus respectively on pairs 6-17 and 17-21, and neither of them is in contact. To be more systematic, we identify for each method the features focusing on two and three sites (via the criterion $\sum_v |w_{i\mu}(v)| > 0.5 \max_i \sum_v |w_{i\mu}(v)|$), count them, and compare them to the original structure. Results are shown in Table 2. For RBM, 49 hidden units focus on two sites, of which 47 are in contact, and 18 focus on three sites, of which 16 are in contact (e.g., like 8-16-27). For the other methods, both the number of features and the true positive rate are significantly lower. In sparse PCA, only 15/25 pairs and 3/24 triplet features are legitimate contacts or triplets in the structure.

The main reason for this failure is that in sparse PCA (as in PCA), the emphasis is put on the complete reconstruction of the sequence from the representation because the mapping is assumed to be deterministic rather than stochastic. The sequence must be compressed entirely in the latent factors, of smaller size $M = 100 < 27 \times 20 = 540$, and this is achieved by grouping sites in a fashion that may respect some correlations but not necessarily the

underlying interactions. Therefore, even changing the reconstruction error to cross-entropy to properly take into account the categorical nature of the data does not significantly improve the results. However, we found that corrupting the sequence with noise before attempting to reconstruct it (i.e., introducing a stochastic mapping) indeed slightly improves the performance, though not to the same level as RBM for quantitative modeling.

The simplifying assumption that $P(\mathbf{v}|\mathbf{z})$ is deterministic can be lifted by adopting a variational approach for learning the model. In the case of gaussian distribution for z_μ, we obtain the variational autoencoder model. The features obtained are quite similar to the ones obtained with RBM, featuring contacts, triplets (with very few false positives), and some extended modes. Owing to this stochastic mapping, the representation need not encode individual site variability and focuses instead on the collective behavior of the system.

The major inconvenience of VAE is that we find that only a small number of latent factors (approximately 20) are effectively connected to the visible layer. Increasing the number of latent factors or changing the training algorithm (SGD versus ADAM, batch normalization) did not improve this number. This is likely because the independent gaussian assumption is not compatible with the linear-nonlinear decoder and sparse weights assumption. Indeed, the posterior distribution of the latent factors (from $P(\mathbf{z}|\mathbf{v})$) shows significant correlations and deviations from gaussianity (see Figures 16b and 16c). The KL regularization term of VAE training therefore encourages them to be disconnected. Another consequence of this deviation from i.i.d. gaussian is that sequences generated with the VAE have significantly lower fitness and diversity than with RBM (see Figures 17 and 18). Though deep mappings between the representation and the sequence as in Riesselman et al. (2017), as well as more elaborate priors, as in Mescheder, Nowozin, and Geiger (2017) can be considered for quantitative modeling, the resulting models are significantly less interpretable. In contrast, undirected graphical models such as RBM have a more flexible latent variable distribution and are therefore more expressive for a fixed linear-nonlinear architecture.

In summary, the key ingredients for learning RBM-like representations are sparse weights, stochastic mapping between configurations and representations, and flexible latent variable distributions.

5.3 Lattice Proteins: Robustness of Features. Somewhat surprisingly, we also find that allowing a stochastic mapping between representation and configurations results in features that are significantly more robust with respect to finite sampling. For all the models and parameters used above, we repeat the training five times with either the original data set and varying seed or the shuffled data set, in which each column is shuffled independently from the others so as to preserve the first-order moments and to

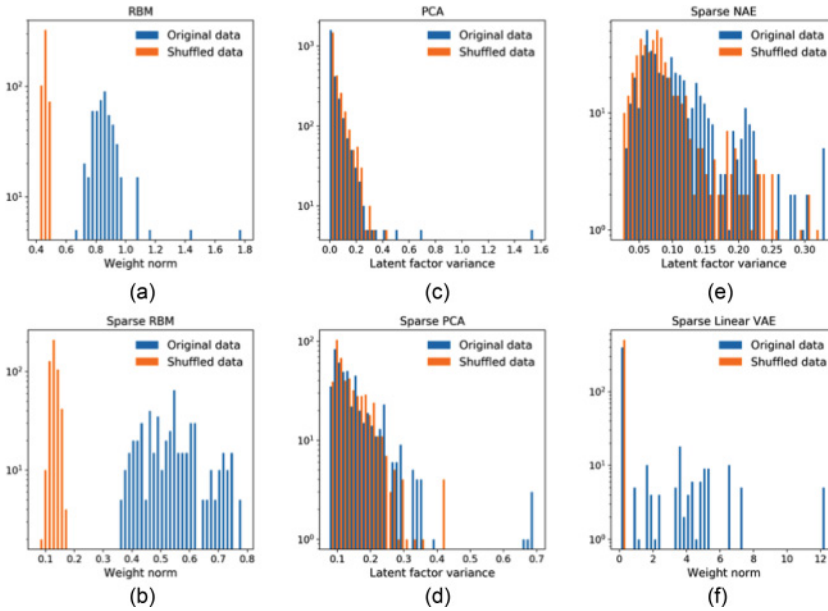

Figure 12: Comparison of feature importance distribution for real (blue) and shuffled (orange) data for (a) unregularized RBM, (b) regularized RBM, (c) PCA, (d) sparse PCA, (e) sparse noisy autoencoders, and (f) sparse variational autoencoders (with linear encoding).

suppress correlations. Since the data size is very large, the models trained on the shuffled data should have latent factors that are either disconnected or localized on a single site, with small feature importance. We show in Figures 12 and 19 the distribution of feature importance for the original and shuffled data for each method. For PCA, ICA, sparse PCA, and sparse autoencoders, the weights are normalized, and feature importance is determined by the variance of the corresponding latent factor. For RBM and VAE, the variance of the latent factors is normalized to 1, and the importance is determined by the norm of the weight vector. In PCA and ICA, we find that only a handful of features emerge from the bulk; sparse regularization slightly improves the number but not by much. In contrast, there is a clean scale separation between feature importance for regular and shuffled data, for both regularized and unregularized RBM and VAE. This notably explains why only a few principal components can be used for protein sequence modeling, whereas many features can be extracted with RBM. The number of modules that can be distinguished within protein sequences is therefore grossly underestimated by principal component analysis.

6 Conclusion

Understanding the collective behavior of a physical system from observations requires learning a phenomenology of data, as well as inferring the main interactions that drive its collective behavior. Representation learning approaches, such as principal component analysis or independent component analysis, have been frequently considered for this purpose. Comparatively, restricted Boltzmann machines, a machine learning tool originally popularized for unsupervised pretraining of deep neural networks, have received little attention, notably due to a limited understanding of the properties of the learned representations.

In this work, we have studied in detail the performances of RBM on a notoriously difficult problem: the prediction of the properties of proteins from their sequence (the so-called genotype-to-phenotype relationship). We have shown that provided that appropriate nonquadratic potentials, as well as sparsity regularization over the weights are used, RBM can learn compositional representations of data, in which the hidden units code for constitutive parts of the data configurations. Each constitutive part, due to sparsity, focuses on a small region of the data configurations (subset of visible sites) and is therefore easier to interpret than extended features. In turn, constitutive parts may be stochastically recombined to generate new data with good properties (here, probability of folding into the desired 3D structure). Regularized RBM therefore offers an appealing compromise between model interpretability and generative performance.

We stress that the behavior of RBM described in this letter is in full agreement with theoretical studies of RBM with weights drawn from random statistical ensembles, controlled by a few parameters, including the sparsity, the aspect ratio of the machine, and the thresholds of rectified linear units (Tubiana & Monasson, 2017). It is, however, a nontrivial result, due to the very nature of the lattice-protein sequence distribution, that forcing RBM trained on such data to operate in the compositional regime can be done at essentially no cost in log likelihood (see Figure 7b). This property also holds for real proteins, as shown in Tubiana et al. (2018).

In addition, RBM enjoys some useful properties with respect to the other representation learning approaches studied in this letter. First, RBM representations focus on the underlying interactions between components rather than on all the variability of the data, taken into account by site-dependent potentials acting on the visible unit. The inferred weights are therefore fully informative on the key interactions within the system. Second, the distribution of latent factors (see Figure 10) is not imposed in RBM, contrary to VAE, where it is arbitrarily supposed to be gaussian. The inference of the hidden-unit potential parameters (thresholds $\theta_{\pm}$ and curvatures $\gamma_{\pm}$ for dReLU units) confers a lot of adaptibility to RBM to fit the data distribution as closely as possible.

Altogether, beyond the protein sequence analysis application presented here, our work suggests that RBMs shed a different light on data and could be useful to model other data in an accurate and interpretable way. While we have focused on single-layer representations of data, a major challenge is to understand how we could leverage our results to learn interpretable multilayer representations of data. Indeed, many data sets, such as databases of images of objects, exhibit hierarchies of features, entailing that the low-level features extracted themselves have complex patterns of covariation. Though it is not clear whether such hierarchy exists in protein sequences, it is likely to exist in other biological data, such as in large-scale recordings of biological neural networks. In that case, an RBM may not be the optimal choice because the correlations between hidden units are not explicitly fitted from the data, as the RBM model is trained to reproduce the moments $\langle v \rangle_{\text{data}}$, $\langle f(h) \rangle_{\text{data}}$, and $\langle vh \rangle_{\text{data}}$ only. Instead, correlations between hidden units arise only through the overlaps between their attached weights (see equation 3.1). Therefore, if two nonoverlapping hidden units are correlated in the data, the only way to account for this property is to introduce an additional hidden unit, whose weight overlaps with both. In such a situation, introducing additional couplings between hidden units or additional layers of hidden units, such as in deep Boltzmann machines or deep belief networks, would result in more parameter-efficient and easier-to-interpret models. However, deep architectures raise numerous additional questions. Besides training, which is more challenging, it is not clear how to interpret and visualize the hidden-unit receptive fields, which are not linear-nonlinear anymore; how to optimally choose the potentials, architecture, and regularization as functions of the data considered; whether and when compositional representations can arise in some layer; how they interact with other types of layers, such as convolutional or pooling layers; and how the nature of representations varies from layer to layer depending on the structural parameters of the model. Another open issue is sampling. RBMs, when driven in the compositional phase, empirically show efficient mixing properties. Characterizing how fast the data distribution is dynamically sampled would be very interesting, with potential payoff for training where benefiting from efficient sampling is crucial.

Appendix A: Additional Details for the Learning Algorithm

Here we provide additional details for the learning algorithm of RBMs. Codes for training RBM, as well as the lattice protein multiple sequence alignment, are available at https://github.com/jertubiana/ProteinMotifRBM.

A.1 Gauge Choice. Since the conditional probability, equation 2.6, is normalized, the transformations $g_i(v) \to g_i(v) + \lambda_i$ and $w_{i\mu}(v) \to w_{i\mu}(v) + K_{i\mu}$ leave the conditional probability invariant. We choose the zero-sum

gauges, defined by $\sum_v g_i(v) = 0$, $\sum_v w_{i\mu}(v) = 0$. Since the regularization penalties over the fields and weight depend on the gauge choice, the gauge must be enforced throughout all training, not only at the end. The updates on the fields leave the gauge invariant, so the transformation $g_i(v) \to g_i(v) - \frac{1}{q}\sum_{v'} g_i(v')$ can be used only once, after initialization. It is not the case for the updates on the weights, so the transformation $w_{i\mu}(v) - \frac{1}{q}\sum_{v'} w_{i\mu}(v')$ must be applied after each gradient update.

A.2 Dynamic Reparameterization. For quadratic and dReLU potentials, there is a redundancy between the slope of the hidden unit average activity and the global amplitude of the weight vector. Indeed, for the gaussian potential, the model distribution is invariant under rescaling transformations $\gamma_\mu \to \lambda^2\gamma_\mu$, $w_{i\mu} \to \lambda w_{i\mu}$, $\theta_\mu \to \lambda\theta_\mu$ and offset transformation $\theta_\mu \to \theta_\mu + K_\mu$, $g_i \to g_i - \sum_\mu w_{i\mu}\frac{K_\mu}{\gamma_\mu}$. Though we can set $\gamma_\mu = 1$, $\theta_\mu = 0\ \forall\mu$ without loss of generality, it can lead to either numerical instability (at a high learning rate) or slow learning (at a low learning rate). A significantly better choice is to dynamically adjust the slope and offset so that $\langle h_\mu\rangle_{\text{data}} \sim 0$ and $\text{Var}(h_\mu) \sim 1$ at all times. This new approach, reminiscent of batch normalization for deep networks, is implemented in the training algorithm released with this work and is benchmarked in Tubiana, Cocco, and Monasson (2019).

A.3 Initialization and Stochastic Gradient Descent. The optimization is carried out by stochastic gradient descent. At each step, the gradient is evaluated using a mini-batch of the data, as well as a small number of Markov chain Monte Carlo configurations. We used the same batch size (= 100) for both sets. The model is initialized as follows:

1. The fields $g_i(v)$ are initialized with the ones of the optimal independent model, $g_i(v) = \log\langle\delta_{v_i,v}\rangle_{\text{data}}$, where δ denotes the Kronecker function. This allows starting from a distribution that is already close to the data distribution and significantly speeds up the learning.
2. The parameters of the dReLU potential are initialized to $\gamma_+ = \gamma_- = 1$, $\theta_+ = \theta_- = 0$. This way, the average activity is initially linear, with a slope of 1. This choice is the most neutral one, as it does not favor any kind of nonlinearity (symmetric, asymmetric, sigmoid-like, or ReLU like).
3. The initial weights are drawn from a gaussian ensemble with variance σ^2/N, such that the hidden unit inputs I_μ have initially zero mean and variance σ^2. Indeed, under the above initialization for the fields and dReLU potential, the initial gradient for W is $\nabla_{\mathbf{W}}\mathcal{L}_{t=0} = W \odot C_{\text{data}}$, where C_{data} is the data covariance matrix (of size $N \times 20 \times N \times 20$) and $\odot$ denotes the tensor product over the first two axes Decelle, Fissore, & Furtlehner (2017). Therefore, using zero weights

leads to a saddle point of the optimization; very small weights result in a small gradient and slow initial learning; and very large weights ($\sigma > 1$ result in a spin-glass phase in which samples mix poorly, which can result in divergence. We found that setting $\sigma = 0.1$ yields a good compromise between a fast initial learning speed and a fast mixing rate. Using orthogonal ensembles of weights instead of gaussian ones such as in Pennington, Schoenholz, and Ganguli (2017) was also briefly experimented but did not yield any significant speed-up for the lattice proteins data set.

The learning rate is initially set to 0.1 and decays exponentially after a fraction of the total training time (e.g., 50%) until it reaches a final, small value (e.g., 10^{-4}).

A.4 Explicit Expressions for Training and Sampling from RBM. Training, sampling, and computing the probability of sequences with RBM requires (1) sampling from $P(\mathbf{v}|\mathbf{h})$, (2) sampling from $P(\mathbf{h}|\mathbf{v})$, and (3) evaluating the effective energy $E_{\text{eff}}(\mathbf{v})$ and its derivatives. This is done as follows:

1. Each sequence site i is encoded as a categorical variable taking integer values $v_i \in [0, 20]$, with each integer corresponding to one of the 20 amino acids. Similarly, the fields and weights are encoded as, respectively, an $N \times 20$ matrix and an $M \times N \times 20$ tensor.
2. Given a hidden layer configuration, each visible unit is sampled from a categorical distribution with 20 states.
3. Given a visible layer distribution, each hidden unit is sampled from $P(h_\mu|\mathbf{v})$. For a quadratic potential, this conditional distribution is gaussian. For the dReLU potential, let $\Phi(x) = \exp(\frac{x^2}{2})\big[1 - \operatorname{erf}(\frac{x}{\sqrt{2}})\big]\sqrt{\frac{\pi}{2}}$. Φ satisfies the following properties:
   - $\Phi(x) \sim_{x\to-\infty} \exp(\frac{x^2}{2})\sqrt{2\pi}$
   - $\Phi(x) \sim_{x\to\infty} \frac{1}{x} - \frac{1}{x^3} + \frac{3}{x^5} + \mathcal{O}(\frac{1}{x^7})$
   - $\Phi'(x) = x\Phi(x) - 1$

   To avoid numerical issues, Φ is computed in practice with its definition for $x < 5$ and with its asymptotic expansion otherwise. We also write $\mathcal{TN}(\mu, \sigma^2, a, b)$—the truncated gaussian distribution of mode μ, width σ, and support $[a, b]$. Then, $P(h|I)$ is given by a mixture of two truncated gaussians:

$$P(h|I) = p^+\mathcal{TN}\left(\frac{I-\theta^+}{\gamma_+}, \frac{1}{\gamma_+}, 0, +\infty\right) + p^-\mathcal{TN}\left(\mu = \frac{I-\theta^-}{\gamma^-}, \sigma^2 = \frac{1}{\gamma^-}, -\infty, 0\right), \tag{A.1}$$

where $Z^\pm = \Phi\left(\frac{\mp(I-\theta^\pm)}{\sqrt{\gamma^\pm}}\right)\frac{1}{\sqrt{\gamma^\pm}}$, and $p^\pm = \frac{Z^\pm}{Z^+ + Z^-}$.

4. Evaluating E_{eff} and its derivatives requires an explicit expression for the cumulant-generating function $\Gamma(I)$. For quadratic potentials, $\Gamma(I)$ is given in main text. For dReLU potentials, $\Gamma(I) = \log(Z^+ + Z^-)$ where $Z^\pm$ are defined above.

Appendix B: Proxies for Weight Sparsity, Number of Strongly Activated Hidden Units

Since neither the hidden layer activity nor the weights are exactly zero, proxies are required for evaluating them. In order to avoid the use of arbitrary thresholds, which may not be adapted to every case, we use participation ratios.

The participation ratio (PR_e) of a vector $\mathbf{x} = \{x_i\}$ is

$$PR_e(\mathbf{x}) = \frac{(\sum_{\mathbf{i}} |\mathbf{x_i}|^{\mathbf{e}})^{\mathbf{2}}}{\sum_{\mathbf{i}} |\mathbf{x_i}|^{\mathbf{2e}}}. \tag{B.1}$$

If $\mathbf{x}$ has K nonzero and equal (in modulus) components, PR is equal to K for any a. In practice, we use the values $a = 2$ and 3: the higher a is, the more small components are discounted against strong components in $\mathbf{x}$. Note also that it is invariant to rescaling of $\mathbf{x}$.

PR can be used to estimate the weight sparsity for a given hidden unit and averaged to get a single value for a RBM:

$$\begin{aligned} w_{i\mu}^S &\equiv \sqrt{\sum_a w_{i\mu}(a)^2}, \\ p_\mu &= \frac{1}{N} PR_2(\mathbf{w}_\mu^{\mathbf{S}}), \\ p &= \frac{1}{M} \sum_\mu p_\mu. \end{aligned} \tag{B.2}$$

Similarly, the number of strongly activated hidden units for a given visible layer configuration can be computed with a participation ratio. For nonnegative hidden units such as with the ReLU potential, it is obtained via

$$\begin{aligned} h_\mu &= \langle h_\mu | I_\mu(\mathbf{v}) \rangle, \\ L &= PR_3(\mathbf{h}_\mu). \end{aligned} \tag{B.3}$$

For dReLU hidden units, which can take both positive and negative values and may have, for example, bimodal activity, their most frequent activity can be nonzero. We therefore subtract it before computing the

participation ratio:

$$h_\mu = \langle h_\mu | I_\mu(\mathbf{v}) \rangle,$$
$$h_\mu^0 = \arg\max P_{\text{data}}(h_\mu),$$
$$L = PR_3(|\mathbf{h}_\mu - \mathbf{h}_\mu^0|). \tag{B.4}$$

To measure the overlap between hidden units μ, μ', we introduce the quantities

$$O_{\mu\nu} = \frac{\sum_{i,v} w_{i\mu}(v) w_{iv}(v)}{\sqrt{\left(\sum_{i,v} w_{i\mu}(v)^2\right)\left(\sum_{i,v} w_{iv}(v)^2\right)}}, \tag{B.5}$$

which take values in the $[-1, 1]$ range. We also define $O_\mu^{\max} = \max_{\mu' \neq \mu} |O_{\mu,\mu'}|$.

To account for strongly overlapping hidden units, one can also compute the following weighted participation ratio:

$$PR_e(\mathbf{h}, \mathbf{w}) = \frac{(\sum_\mu \mathbf{w}_\mu |\mathbf{h}_\mu|^{\mathbf{e}})^2}{\sum_\mu \mathbf{w}_\mu |\mathbf{h}_\mu|^{\mathbf{2e}}}, \tag{B.6}$$

where w_μ is chosen as the inverse of the number of neighboring hidden units, defined according to the criterion $|O_{\mu\nu}| > 0.9$. This way, two hidden units with identical weights contribute to the participation ratio as much as a single isolated one does.

Appendix C: Supplementary Figures

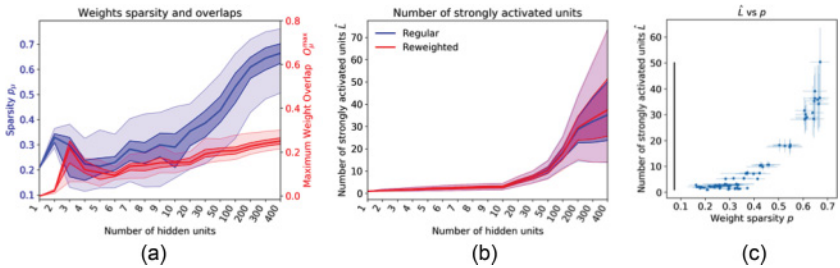

Figure 13: Evolution of the weight sparsity p as a function of the number of hidden units for quadratic potential and no regularization. Unlike in the compositional phase, the number of activated hidden units scales as M.

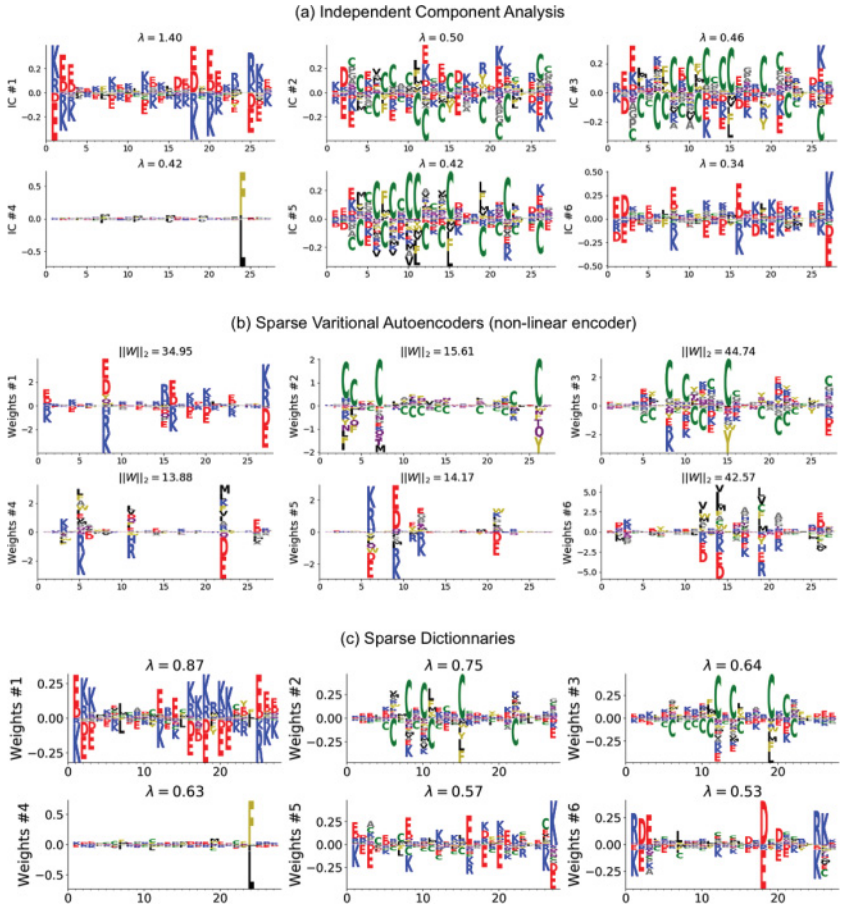

Figure 14: Six weights attached to latent factors for (a) independent component analysis, (b) sparse variational autoencoders with nonlinear encoding, and (c) sparse dictionaries. For panels a and c, the six latent factor with largest variance are shown. For panel b, we selected six latent factors whose corresponding weights have weight sparsity p_μ close to the median value.

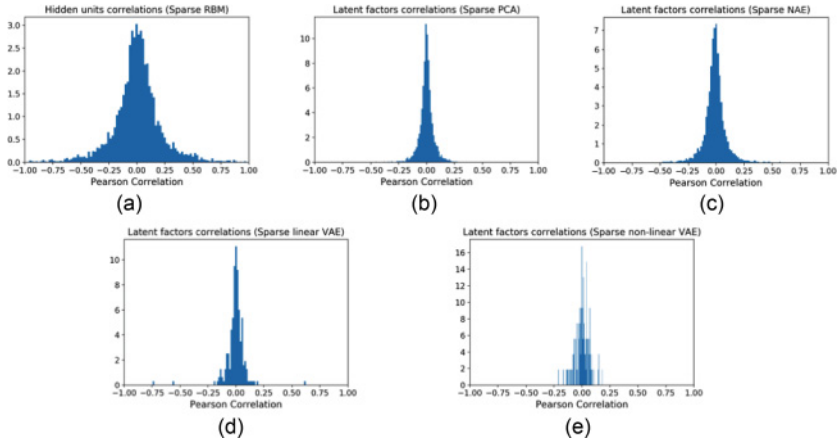

Figure 15: Distribution of Pearson correlations within the representation for the models shown in Figures 11 and 14. (a) RBM (regularized, $\lambda_1^2 = 0.025$). (b) Sparse PCA. (c) Sparse noisy autoencoders. (d) Sparse VAE with linear encoder. (e) Sparse VAE with nonlinear encoder. For models b to e, the sparse weight regularization is selected to yield the same median weight sparsity as for the regularized RBM.

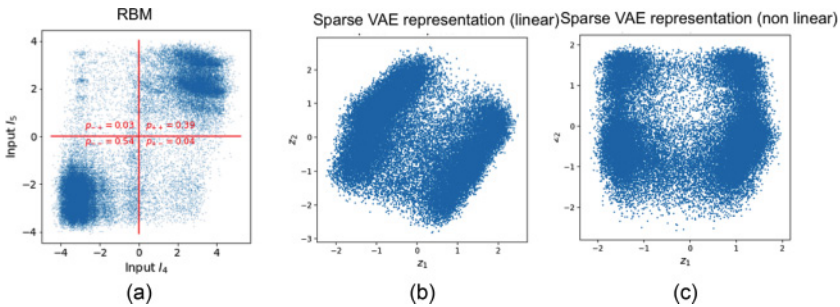

Figure 16: Example of latent factor data distributions, obtained by sampling from $P(\mathbf{h}/\mathbf{z}|\mathbf{v})$ for each sequence $\mathbf{v}$ in the data. (a) Scatter plot of h_4 versus h_5 for the RBM shown in Figure 6. Note the strong but imperfect correlation and the four distinct clusters. (b,c) Scatter plot of z_1 versus z_2, the two most important features, for the sparse VAE with (b) linear or (c) nonlinear encoding. Note the strong deviations from the i.i.d. gaussian distribution.

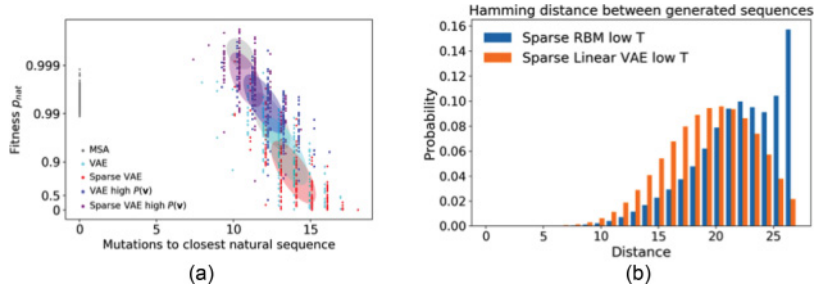

Figure 17: Sequence generation with linear VAE. (a) Scatter plot of the number of mutations to the closest natural sequence versus probability to fold into S_A, for natural and artificial LP sequences. Sequences are generated from unregularized and regularized VAE by sampling random gaussian i.i.d. $\mathbf{z}$. Then $\mathbf{v}$ are obtained either by sampling from $P(\mathbf{v}|\mathbf{z})$ or as $\arg\max_{\mathbf{v}} P(\mathbf{v}|\mathbf{z})$ (high probability). Same colors and scale as Figure 6. RBM sequences (regular and high $P(\mathbf{v})$ are shown in gray. (b) Distribution of Hamming distance between generated sequences, showing the diversity of the generated sequences.

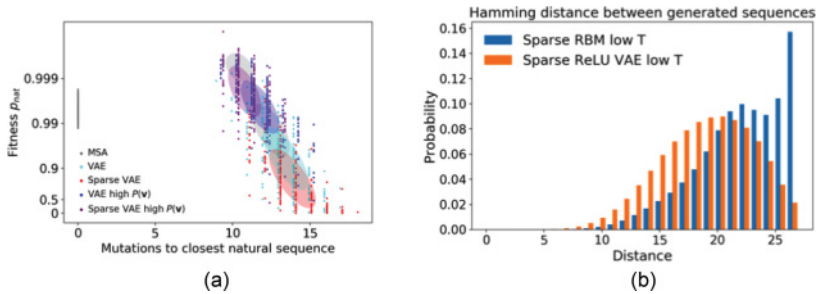

Figure 18: Same as Figure 17 for VAE with encoding.

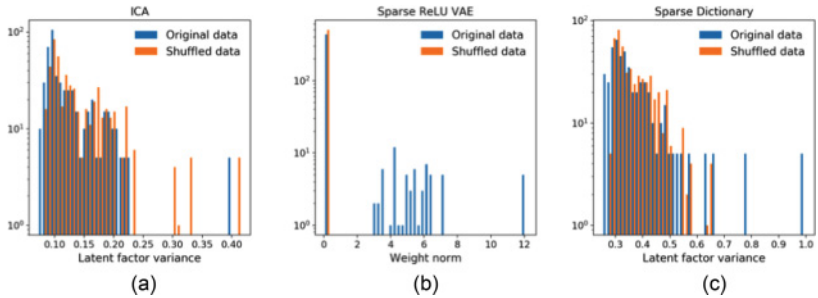

Figure 19: Comparison of feature importance distribution for real (blue) and shuffled (orange) data, for (a) ICA, (b) sparse variational autoencoders (with nonlinear encoding), (c) sparse dictionaries.

Protein Family	N	M	alpha=M/Nq	p	pN
PF00046	57	114	0.10	0.13	7.69
PF00014	53	53	0.05	0.10	5.06
PF00027	88	176	0.10	0.08	7.35
PF00028	93	93	0.05	0.08	7.66
PF00105	70	140	0.10	0.12	8.30
PF00107	129	129	0.05	0.10	13.02
PF00076	70	140	0.10	0.10	6.66
PF00028	93	186	0.10	0.07	6.25
PF00013	66	66	0.05	0.13	8.71
PF00111	78	78	0.05	0.10	7.57
PF00011	102	204	0.10	0.09	8.92
PF00084	56	112	0.10	0.10	5.62
PF00035	67	134	0.10	0.11	7.25
PF00084	56	56	0.05	0.10	5.38
PF00017	77	77	0.05	0.15	11.70
PF00081	82	164	0.10	0.08	6.67
PF00017	77	154	0.10	0.13	9.70
PF00046	57	57	0.05	0.13	7.53
PF00041	85	170	0.10	0.07	5.95
PF00107	129	258	0.10	0.07	9.65
PF00076	70	70	0.05	0.10	7.03
PF00018	48	48	0.05	0.13	6.43
PF00035	67	67	0.05	0.13	8.60
PF00111	78	156	0.10	0.09	6.78
PF00014	53	106	0.10	0.11	5.65
PF00013	66	132	0.10	0.12	8.06
PF00027	88	88	0.05	0.11	9.62
PF00043	92	92	0.05	0.11	10.49
PF00043	92	184	0.10	0.09	8.17
PF00041	85	85	0.05	0.08	6.71
PF00081	82	82	0.05	0.10	8.12
PF00018	48	96	0.10	0.12	5.80
PF00105	70	70	0.05	0.13	8.83
PF00011	102	102	0.05	0.11	11.38
Hsp70	675	200	0.01	0.04	26.64
Kunitz	53	100	0.09	0.11	5.72
Lattice Proteins	27	100	0.19	0.09	2.55
WW	31	100	0.15	0.13	3.99

std(p)/mean(p) = 0.22
std(pN)/mean(pN) = 0.45

Figure 20: Values of the average weight sparsity p and average number of connections per hidden units pN for 38 RBM trained on real protein families.

Acknowledgments

We acknowledge Clément Roussel for helpful comments. This work was partly funded by the ANR project RBMPro CE30-0021-01. JT acknowledges funding from the Safra Center for Bioinformatics, Tel Aviv University.

References

Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1987). A learning algorithm for Boltzmann machines. In M. Fischler & O. Firschein (Eds.), *Readings in computer vision* (pp. 522–533). Amsterdam: Elsevier.

Agliari, E., Annibale, A., Barra, A., Coolen, A., & Tantari, D. (2013). Immune networks: Multitasking capabilities near saturation. *Journal of Physics A: Mathematical and Theoretical*, *46*(41), 415003.

Agliari, E., Barra, A., Galluzzi, A., Guerra, F., & Moauro, F. (2012). Multitasking associative networks. *Phys. Rev. Lett.*, *109*, 268101.

Amit, D. J., Gutfreund, H., & Sompolinsky, H. (1985). Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, *55*(14), 1530.

Baden, T., Berens, P., Franke, K., Rosón, M. R., Bethge, M., & Euler, T. (2016). The functional diversity of retinal ganglion cells in the mouse. *Nature*, *529*(7586), 345.

Barra, A., Bernacchia, A., Santucci, E., & Contucci, P. (2012). On the equivalence of Hopfield networks and Boltzmann machines. *Neural Networks*, *34*, 1–9.

Bell, A. J., & Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*(6), 1129–1159.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., . . . Varoquaux, G. (2013). *API design for machine learning software: Experiences from the Scikit-learn project*. arXiv:1309.0238.

Casari, G., Sander, C., & Valencia, A. (1995). A method to predict functional residues in proteins. *Nature Structural and Molecular Biology*, *2*(2), 171.

Cho, K., Raiko, T., & Ilin, A. (2010). Parallel tempering is efficient for learning restricted Boltzmann machines. In *Proceedings of the 2010 International Joint Conference on Neural Networks* (pp. 1–8). Piscataway, NJ: IEEE.

Chollet, F. (2015). Keras. https://keras.io/

Cocco, S., Feinauer, C., Figliuzzi, M., Monasson, R., & Weigt, M. (2018). Inverse statistical physics of protein sequences: A key issues review. *Reports on Progress in Physics*, *81*(3), 032601.

Courville, A., Bergstra, J., & Bengio, Y. (2011). A spike and slab restricted Boltzmann machine. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (pp. 233–241). AISTATS.

Dahl, G., Ranzato, M., Mohamed, A.-r., & Hinton, G. E. (2010). Phone recognition with the mean-covariance restricted Boltzmann machine. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, & A. Culotta (Eds.), *Advances in neural information processing systems*, *23* (pp. 469–477). Red Hook, NY: Curran.

De Juan, D., Pazos, F., & Valencia, A. (2013). Emerging methods in protein co-evolution. *Nature Reviews Genetics*, *14*(4), 249.

Decelle, A., Fissore, G., & Furtlehner, C. (2017). Spectral dynamics of learning in restricted Boltzmann machines. *EPL (Europhysics Letters)*, *119*(6), 60001.

Desjardins, G., Courville, A., & Bengio, Y. (2010). *Adaptive parallel tempering for stochastic maximum likelihood learning of RBMS*. arXiv:1012.3476.

Desjardins, G., Courville, A., Bengio, Y., Vincent, P., & Delalleau, O. (2010). Tempered Markov chain Monte Carlo for training of restricted Boltzmann machines. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 145–152). AISTATS.

Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, *52*(4), 1289–1306.

Figliuzzi, M., Jacquier, H., Schug, A., Tenaillon, O., & Weigt, M. (2016). Coevolutionary landscape inference and the context-dependence of mutations in beta-lactamase tem-1. *Molecular Biology and Evolution*, *33*(1), 268–280.

Finn, R. D., Coggill, P., Eberhardt, R. Y., Eddy, S. R., Mistry, J., Mitchell, A. L., . . . Salazar, G. A. (2015). The PFAM protein families database: Towards a more sustainable future. *Nucleic Acids Research*, *44*(D1), D279–D285.

Fischer, A., & Igel, C. (2012). An introduction to restricted Boltzmann machines. In *Iberoamerican Congress on Pattern Recognition* (pp. 14–36). Berlin: Springer.

Fowler, D. M., Araya, C. L., Fleishman, S. J., Kellogg, E. H., Stephany, J. J., Baker, D., & Fields, S. (2010). High-resolution mapping of protein sequence-function relationships. *Nature Methods*, *7*(9), 741.

Gabrié, M., Tramel, E. W., & Krzakala, F. (2015). Training restricted Boltzmann machine via the thouless-Anderson-Palmer free energy. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems*, *28* (pp. 640–648). Red Hook, NY: Curran.

Greener, J. G., Moffat, L., & Jones, D. T. (2018). Design of metalloproteins and novel protein folds using variational autoencoders. *Scientific Reports*, *8*(1), 16189.

Halabi, N., Rivoire, O., Leibler, S., & Ranganathan, R. (2009). Protein sectors: Evolutionary units of three-dimensional structure. *Cell*, *138*(4), 774–786.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*(8), 1771–1800.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554.

Hopf, T. A., Ingraham, J. B., Poelwijk, F. J., Schärfe, C. P., Springer, M., Sander, C., & Marks, D. (2017). Mutation effects predicted from sequence co-variation. *Nature Biotechnology*, *35*(2), 128–135.

Hyvärinen, A., & Oja, E. (2000). Independent component analysis: Algorithms and applications. *Neural Networks*, *13*(4–5), 411–430.

Jacquin, H., Gilson, A., Shakhnovich, E., Cocco, S., & Monasson, R. (2016). Benchmarking inverse statistical approaches for protein structure and design with exactly solvable models. *PLoS Computational Biology*, *12*(5), e1004889.

Kingma, D. P., & Welling, M. (2013). *Auto-encoding variational Bayes*. arXiv:1312.6114.

Kolodziejczyk, A. A., Kim, J. K., Svensson, V., Marioni, J. C., & Teichmann, S. A. (2015). The technology and biology of single-cell RNA sequencing. *Molecular Cell*, *58*(4), 610–620.

Mairal, J., Bach, F., Ponce, J., & Sapiro, G. (2009). Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning* (pp. 689–696). New York: ACM.

McKeown, M. J., Makeig, S., Brown, G. G., Jung, T.-P., Kindermann, S. S., Bell, A. J., & Sejnowski, T. J. (1998). Analysis of FMRI data by blind separation into independent spatial components. *Human Brain Mapping*, *6*(3), 160–188.

McLaughlin Jr, R. N., Poelwijk, F. J., Raman, A., Gosal, W. S., & Ranganathan, R. (2012). The spatial architecture of protein function and adaptation. *Nature*, *491*(7422), 138.

Mescheder, L., Nowozin, S., & Geiger, A. (2017). *Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks*. arXiv:1701.04722.

Miyazawa, S., & Jernigan, R. L. (1996). Residue–residue potentials with a favorable contact pair term and an unfavorable high packing density term, for simulation and threading. *Journal of Molecular Biology*, *256*(3), 623–644.

Morcos, F., Pagnani, A., Lunt, B., Bertolino, A., Marks, D. S., Sander, C., . . . Weigt, M. (2011). Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *Proceedings of the National Academy of Sciences*, *108*(49), E1293–E1301.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning* (pp. 807–814). Madison, WI: Omnipress.

Neal, R. M. (2001). Annealed importance sampling. *Statistics and Computing*, *11*(2), 125–139.

Nguyen, H. C., Zecchina, R., & Berg, J. (2017). Inverse statistical problems: From the inverse Ising problem to data science. *Advances in Physics*, *66*(3), 197–261.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*(6583), 607.

Pennington, J., Schoenholz, S., & Ganguli, S. (2017). Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanatan, & R. Garnett (Eds.), *Advances in neural information processing systems*, *30* (pp. 4785–4795).

Posani, L., Cocco, S., Ježek, K., & Monasson, R. (2017). Functional connectivity models for decoding of spatial representations from hippocampal CA1 recordings. *Journal of Computational Neuroscience*, *43*(1), 17–33.

Quadeer, A. A., Morales-Jimenez, D., & McKay, M. R. (2018). *Co-evolution networks of HIV/HCV are modular with direct association to structure and function*. bioRxiv:307033.

Rausell, A., Juan, D., Pazos, F., & Valencia, A. (2010). Protein interactions and ligand binding: From protein subfamilies to functional specificity. *Proceedings of the National Academy of Sciences*, *107*(5), 1995–2000.

Riesselman, A. J., Ingraham, J. B., & Marks, D. S. (2017). *Deep generative models of genetic variation capture mutation effects*. arxiv:1712.06527.

Rivoire, O., Reynolds, K. A., & Ranganathan, R. (2016). Evolution-based functional decomposition of proteins. *PLoS Computational Biology*, *12*(6), e1004817.

Salakhutdinov, R. (2010). Learning deep Boltzmann machines using adaptive MCMC. In *Proceedings of the 27th International Conference on Machine Learning* (pp. 943–950). Madison, WI: Omnipress.

Salakhutdinov, R., & Murray, I. (2008). On the quantitative analysis of deep belief networks. In *Proceedings of the 25th International Conference on Machine Learning* (pp. 872–879). New York: ACM.

Schwarz, D. A., Lebedev, M. A., Hanson, T. L., Dimitrov, D. F., Lehew, G., Meloy, J., . . . Nicolelis, M. A. (2014). Chronic, wireless recordings of large-scale brain activity in freely moving rhesus monkeys. *Nature Methods*, *11*(6), 670.

Shakhnovich, E., & Gutin, A. (1990). Enumeration of all compact conformations of copolymers with random sequence of links. *Journal of Chemical Physics*, *93*(8), 5967–5971.

Sinai, S., Kelsic, E., Church, G. M., & Novak, M. A. (2017). *Variational autoencoding of protein sequences*. arxiv:1712.03346.

Tieleman, T. (2008). Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning* (pp. 1064–1071). New York: ACM.

Tkacik, G., Prentice, J. S., Balasubramanian, V., & Schneidman, E. (2010). Optimal population coding by noisy spiking neurons. *Proceedings of the National Academy of Sciences USA*, *107*(32), 14419–14424.

Tramel, E. W., Gabrié, M., Manoel, A., Caltagirone, F., & Krzakala, F. (2017). *A deterministic and generalized framework for unsupervised learning with restricted Boltzmann machines*. arXiv:1702.03260.

Tubiana, J., Cocco, S., & Monasson, R. (2018). *Learning protein constitutive motifs from sequence data*. arXiv:1803.08718.

Tubiana, J., Cocco, S., & Monasson, R. (2019). *Efficient sampling and parameterization improve Boltzmann machines*. Manuscript in preparation.

Tubiana, J., & Monasson, R. (2017). Emergence of compositional representations in restricted Boltzmann machines. *Physical Review Letters*, *118*(13), 138301.

Vu, H., Nguyen, T. D., Le, T., Luo, W., & Phung, D. (2018). Batch normalized deep Boltzmann machines. In *Proceedings of the Asian Conference on Machine Learning* (pp. 359–374).

Weigt, M., White, R. A., Szurmant, H., Hoch, J. A., & Hwa, T. (2009). Identification of direct residue contacts in protein–protein interaction by message passing. *Proceedings of the National Academy of Sciences*, *106*(1), 67–72.

Wolf, S., Supatto, W., Debrégeas, G., Mahou, P., Kruglik, S. G., Sintes, J.-M., . . . Candelier, R. (2015). Whole-brain functional imaging with two-photon light-sheet microscopy. *Nature Methods*, *12*(5), 379.

Zou, H., Hastie, T., & Tibshirani, R. (2006). Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, *15*(2), 265–286.

Received February 18, 2019; accepted April 6, 2019.