An Empirical Investigation of Minimum Probability Flow Learning Under Different Connectivity Patterns

Daniel Jiwoong Im(✉), Ethan Buchman, and Graham W. Taylor

School of Engineering, University of Guelph, Guelph, ON, Canada
{imj,ebuchman,gwtaylor}@uoguelph.ca

Abstract. Energy-based models are popular in machine learning due to the elegance of their formulation and their relationship to statistical physics. Among these, the Restricted Boltzmann Machine (RBM), and its staple training algorithm contrastive divergence (CD), have been the prototype for some recent advancements in the unsupervised training of deep neural networks. However, CD has limited theoretical motivation, and can in some cases produce undesirable behaviour. Here, we investigate the performance of Minimum Probability Flow (MPF) learning for training RBMs. Unlike CD, with its focus on approximating an intractable partition function via Gibbs sampling, MPF proposes a tractable, consistent, objective function defined in terms of a Taylor expansion of the KL divergence with respect to sampling dynamics. Here we propose a more general form for the sampling dynamics in MPF, and explore the consequences of different choices for these dynamics for training RBMs. Experimental results show MPF outperforming CD for various RBM configurations.

1 Introduction

A common problem in machine learning is to estimate the parameters of a high-dimensional probabilistic model using gradient descent on the model's negative log likelihood. For exponential models where $p(x)$ is proportional to the exponential of a negative potential function $F(x)$, the gradient of the data negative log-likelihood takes the form

$$\nabla_\theta = \frac{1}{|\mathcal{D}|}\left(\sum_{x\in\mathcal{D}}\frac{\partial F(x)}{\partial\theta} - \sum_{x} p(x)\frac{\partial F(x)}{\partial\theta}\right) \tag{1}$$

where the sum in the first term is over the dataset, $\mathcal{D}$, and the sum in the second term is over the entire domain of x. The first term has the effect of pushing the parameters in a direction that decreases the energy surface of the model at the training data points, while the second term increases the energy of all possible states. Since the second term is intractable for all but trivial models, we cannot, in practice, accommodate for every state of x, but rather resort to

© Springer International Publishing Switzerland 2015
A. Appice et al. (Eds.): ECML PKDD 2015, Part I, LNAI 9284, pp. 483–497, 2015.
DOI: 10.1007/978-3-319-23528-8_30

sampling. We call states in the sum in the first term *positive particles* and those in the second term *negative particles*, in accordance with their effect on the likelihood (opposite their effect on the energy). Thus, the intractability of the second term becomes a problem of *negative particle selection* (NPS).

The most famous approach to NPS is Contrastive Divergence (CD) [4], which is the centre-piece of unsupervised neural network learning in energy-based models. "CD-k" proposes to sample the negative particles by applying a Markov chain Monte Carlo (MCMC) transition operator k times to each data state. This is in contrast to taking an unbiased sample from the distribution by applying the MCMC operator a large number of times until the distribution reaches equilibrium, which is often prohibitive for practical applications. Much research has attempted to better understand this approach and the reasoning behind its success or failure [6,12], leading to many variations being proposed from the perspective of improving the MCMC chain. Here, we take a more general approach to the problem of NPS, in particular, through the lens of the Minimum Probability Flow (MPF) algorithm [11].

MPF works by introducing a continuous dynamical system over the model's distribution, such that the equilibrium state of the dynamical system is the distribution used to model the data. The objective of learning is to minimize the flow of probability from data states to non-data states after infinitesimal evolution under the model's dynamics. Intuitively, the less a data vector evolves under the dynamics, the closer it is to an equilibrium point; or from our perspective, the closer the equilibrium distribution is to the data. In MPF, NPS is replaced by a more explicit notion of *connectivity* between states. Connected states are ones between which probability can flow under the dynamical system. Thus, rather than attempting to approximate an intractable function (as in CD-k), we run a simple optimization over an explicit, continuous dynamics, and actually never have to run the dynamics themselves.

Interestingly, MPF and CD-k have gradients with remarkably similar form. In fact, the CD-k gradients can be seen as a special case of the MPF gradients - that is, MPF provides a generalized form which reduces to CD-k under a special dynamics. Moreover, MPF provides a consistent estimator for the model parameters, while CD-k as typically formalized is an update heuristic, that can sometimes do bizarre things like go in circles in parameter space [6]. Thus, in one aspect, MPF *solves* the problem of contrastive divergence by re-conceptualizing it as probability flow under an explicit dynamics, rather than the convenient but biased sampling of an intractable function. The challenge thus becomes one of how to design the dynamical system.

This paper makes the following contributions. First, we provide an explanation of MPF that begins from the familiar territory of CD-k, rather than the less familiar grounds of the master equation. While familiar to physicists, the master equation is an apparent obscurity in machine learning, due most likely to its general intractability. Part of the attractiveness of MPF is the way it circumvents that intractability. Second, we derive a generalized form for the MPF transition matrix, which defines the dynamical system. Third, we provide a Theano [1] based implementation of MPF and a number of variants of MPF that run efficiently on GPUs[1]. Finally, we compare and contrast variants of MPF with those of CD-k, and experimentally demonstrate that variants of MPF outperform CD-k for Restricted Boltzmann Machines trained on MNIST and on Caltech-101.

2 Restricted Boltzmann Machines

While the learning methods we discuss apply to undirected probabilistic graphical models in general, we will use the Restricted Boltzmann Machine (RBM) as a canonical example. An RBM is an undirected bipartite graph with visible (observed) variables $\mathbf{v} \in \{0,1\}^D$ and hidden (latent) variables $\mathbf{h} \in \{0,1\}^H$ [9]. The RBM is an energy-based model where the energy of state $\mathbf{v}, \mathbf{h}$ is given by

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\sum_i \sum_j W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j c_j h_j \tag{2}$$

where $\theta = \{W, \mathbf{b}, \mathbf{c}\}$ are the parameters of the model. The marginalized probability over visible variables is formulated from the Boltzmann distribution,

$$p(\mathbf{v}; \theta) = \frac{p^*(\mathbf{v}; \theta)}{Z(\theta)} = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} \exp\left(\frac{-1}{\tau} E(\mathbf{v}, \mathbf{h}; \theta)\right) \tag{3}$$

such that $Z(\theta) = \sum_{\mathbf{v},\mathbf{h}} \exp\left(\frac{-1}{\tau} E(\mathbf{v}, \mathbf{h}; \theta)\right)$ is a normalizing constant and τ is the thermodynamic temperature. We can marginalize over the binary hidden states in Equation 2 and re-express in terms of a new energy $F(\mathbf{v})$,

$$F(\mathbf{v}; \theta) = -\log \sum_{\mathbf{h}} \exp\left(\frac{-1}{\tau} E(\mathbf{v}, \mathbf{h})\right) \tag{4}$$

$$= \frac{1}{\tau} \sum_i^D v_i b_i - \frac{1}{\tau} \sum_{j=1}^H \log\left(1 + \exp\left(c_j + \sum_i^D v_i W_{i,j}\right)\right) \tag{5}$$

$$p(\mathbf{v}; \theta) = \frac{\exp\big(-F(\mathbf{v}; \theta)\big)}{Z(\theta)} \tag{6}$$

Following physics, this form of the energy is better known as a free energy, as it expresses the difference between the average energy and the entropy of a distribution, in this case, that of $p(\mathbf{h}|\mathbf{v})$. Defining the distribution in terms of free energy as $p(\mathbf{v}; \theta)$ is convenient since it naturally copes with the presence of latent variables.

[1] https://github.com/jiwoongim/minimum_probability_flow_learning

The key characteristic of an RBM is the simplicity of inference due to conditional independence between visible and hidden states:

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}), \qquad p(h_j = 1|\mathbf{v}) = \sigma(\sum_i W_{ij} v_i + c_j)$$
$$p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}), \qquad p(v_i = 1|\mathbf{h}) = \sigma(\sum_j W_{ij} h_j + b_i)$$

where $\sigma(z) = 1/(1 + \exp(-z))$.

This leads naturally to a block Gibbs sampling dynamics, used universally for sampling from RBMs. Hence, in an RBM trained by CD-k, the connectivity (NPS) is determined with probability given by k sequential block Gibbs sampling transitions.

We can formalize this by writing the learning updates for CD-k as follows

$$\Delta\theta_{CD-k} \propto -\sum_{j \in \mathcal{D}} \sum_{i \notin \mathcal{D}} \Big(\frac{\partial F_j(\theta)}{\partial \theta} - \frac{\partial F_i(\theta)}{\partial \theta} \Big) T_{ij} \tag{7}$$

where T_{ij} is the probability of transitioning from state j to state i in k steps of block Gibbs sampling. We can in principle replace T_{ij} by any other transition operator, so long as it preserves the equilibrium distribution. Indeed, this is what alternative methods, like Persistent CD [13], achieve.

3 Minimum Probability Flow

The key intuition behind MPF is that NPS can be reformulated in a firm theoretical context by treating the model distribution as the end point of some explicit continuous dynamics, and seeking to minimize the flow of probability away from the data under those dynamics. In this context then, NPS is no longer a sampling procedure employed to approximate an intractable function, but arises naturally out of the probability flow from data states to non-data states. That is, MPF provides a theoretical environment for the formal treatment of T_{ij} that offers a much more general perspective of that operator than CD-k can. In the same vein, it better formalizes the notion of minimizing divergence between positive and negative particles.

3.1 Dynamics of the Model

The primary mathematical apparatus for MPF is a continuous time Markov chain known as the *master equation*,

$$\dot{p}_i = \sum_{j \neq i} [\Gamma_{ij} p_j^{(t)} - \Gamma_{ji} p_i^{(t)}] \tag{8}$$

where j are the data states and i are the non-data states and Γ_{ij} is the probability flow rate from state j to state i. Note that each state is a full vector of variables, and we are theoretically enumerating all states. $\dot{p}_i$ is the rate of change of the probability of state i, that is, the difference between the probability flowing out of any state j into state i and the probability flowing out of state i to any other state j at time t. We can re-express $\dot{p}_i$ in a simple matrix form as

$$\dot{\mathbf{p}} = \mathbf{\Gamma p} \tag{9}$$

by setting $\Gamma_{ii} = -\sum_{i \neq j} \Gamma_{ji} p_i^{(t)}$. We note that if the transition matrix Γ is ergodic, then the model has a unique stationary distribution.

This is a common model for exploring statistical mechanical systems, but it is unwieldy in practice for two reasons, namely, the continuous time dynamics, and exponential size of the state space. For our purposes, we will actually find the former an advantage, and the latter irrelevant.

The objective of MPF is to minimize the KL divergence between the data distribution and the distribution after evolving an infinitesimal amount of time under the dynamics:

$$\theta_{MPF} = \text{argmin}_\theta J(\theta),\ J(\theta) = D_{KL}(p^{(0)}||p^{(\epsilon)}(\theta))$$

Approximating $J(\theta)$ up to a first order Taylor expansion with respect to time t, our objective function reduces to

$$J(\theta) = \frac{\epsilon}{|\mathcal{D}|} \sum_{j \in \mathcal{D}} \sum_{i \notin \mathcal{D}} \Gamma_{ij} \tag{10}$$

and θ can be optimized by gradient descent on $J(\theta)$. Since Γ_{ij} captures probability flow from state j to state i, this objective function has the quite elegant interpretation of minimizing the probability flow from data states to non-data states [11].

3.2 Form of the Transition Matrix

MPF does not propose to *actually* simulate these dynamics. There is, in fact, no need to, as the problem formulation reduces to a rather simple optimization problem with no intractable component. However, we must provide a means for computing the matrix coefficients Γ_{ij}. Since our target distribution is the distribution defined by the RBM, we require Γ to be a function of the energy, or more particularly, the parameters of the energy function.

A sufficient (but not necessary) means to guarantee that the distribution $\mathbf{p}^\infty(\theta)$ is a fixed point of the dynamics is to choose Γ to satisfy detailed balance, that is

$$\Gamma_{ji} p_i^{(\infty)}(\theta) = \Gamma_{ij} p_j^{(\infty)}(\theta). \tag{11}$$

The following theorem provides a general form for the transition matrix such that the equilibrium distribution is that of the RBM:

Theorem 1. *1 Suppose $p_j^{(\infty)}$ is the probability of state j and $p_i^{(\infty)}$ is the probability of state i. Let the transition matrix be*

$$\Gamma_{ij} = g_{ij} \exp\left(\frac{o(F_i - F_j) + 1}{2}(F_j - F_i)\right) \tag{12}$$

such that $o(\cdot)$ is any odd function, where g_{ij} is the symmetric connectivity between the states i and j. Then this transition matrix satisfies detailed balance in Equation 11.

The proof is provided in Appendix A.1. The transition matrix proposed by [11] is thus the simplest case of Theorem 1, found by setting $o(\cdot) = 0$ and $g_{ij} = g_{ji}$:

$$\Gamma_{ij} = g_{ij} \exp\left(\frac{1}{2}(F_j(\theta) - F_i(\theta)\right). \tag{13}$$

Given a form for the transition matrix, we can now evaluate the gradient of $J(\theta)$

$$\frac{\partial J(\theta)}{\partial \theta} = \frac{\epsilon}{|\mathcal{D}|} \sum_{j \in \mathcal{D}} \sum_{i \notin \mathcal{D}} \Big(\frac{\partial F_j(\theta)}{\partial \theta} - \frac{\partial F_i(\theta)}{\partial \theta}\Big) T_{ij}$$
$$T_{ij} = g_{ij} \exp\left(\frac{1}{2}\big(F_j(\theta) - F_i(\theta)\big)\right)$$

and observe the similarity to the formulation given for the RBM trained by CD-k (Equation 7). Unlike with CD-k, however, this expression was derived through an explicit dynamics and well-formalized minimization objective.

4 Probability Flow Rates Γ

At first glance, MPF might appear doomed, due to the size of Γ, namely $2^D \times 2^D$, and the problem of enumerating all of the states. However, the objective function in Equation 10 summing over the Γ_{ij}'s only considers transitions between data states j (limited in size by our data set) and non-data states i (limited by the sparseness of our design). By specifying Γ to be sparse, the intractability disappears, and complexity is dominated by the size of the dataset.

Using traditional methods, an RBM can be trained in two ways, either with sampled negative particles, like in CD-k or PCD (also known as stochastic maximum likelihood) [4,13], or via an inductive principle, with fixed sets of "fantasy cases", like in general score matching, ratio matching, or pseudo-likelihood [3,5,7]. In a similar manner, we can define Γ by specifying the connectivity function g_{ij} either as a distribution from which to sample or as fixed and deterministic.

In this section, we examine various kinds of connectivity functions and their consequences on the probability flow dynamics.

4.1 1-bit Flip Connections

It can be shown that score matching is a special case of MPF in continuous state spaces, where the connectivity function is set to connect all states within a small Euclidean distance r in the limit of $r \to 0$ [11]. For simplicity, in the case of a discrete state space (Bernoulli RBM), we can fix the Hamming distance to one instead, and consider that data states are connected to all other states 1-bit flip away:

$$g_{ij} = \begin{cases} 1, & \text{if state } i, j \text{ differs by single bit flip} \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

1-bit flip connectivity gives us a sparse Γ with $2^D D$ non-zero terms (rather than a full 2^{2D}), and may be seen as NPS where the only negative particles are those which are 1-bit flip away from data states. Therefore, we only ever evaluate $|\mathcal{D}|D$ terms from this matrix, making the formulation tractable. This was the only connectivity function pursued in [11] and is a natural starting point for the approach.

Algorithm 1. Minimum probability flow learning with single bit-flip connectivity. Note we leave out all g_{ij} since here we are explicit about only connecting states of Hamming distance 1.

```
– Initialize the parameters $\theta$
– **for** each training example $d \in \mathcal{D}$ **do**
    1. Compute the list of states, $L$, with Hamming distance 1 from $d$
    2. Compute the probability flow $\Gamma_{id} = \exp\left(\frac{1}{2}(F_d(\theta) - F_i(\theta)\right)$ for each $i \in L$
    3. The cost function for $d$ is $\sum_{i \in L} \Gamma_{id}$
    4. Compute the gradient of the cost function, $\frac{\partial J(\theta)}{\partial \theta} = \sum_{i \in L} \left(\frac{\partial F_d(\theta)}{\partial \theta} - \frac{\partial F_i(\theta)}{\partial \theta}\right) \Gamma_{id}$
    5. Update parameters via gradient descent with $\theta \leftarrow \theta - \lambda \nabla J(\theta)$
  **end for**
```

4.2 Factorized Minimum Probability Flow

Previously, we considered connectivity g_{ij} as a binary indicator function of both states i and j. Instead, we may wish to use a probability distribution, such that g_{ij} is the probability that state j is connected to state i (i.e. $\sum_i g_{ij} = 1$). Following [10], we simplify this approach by letting $g_{ij} = g_i$, yielding an *independence chain* [14]. This means the probability of being connected to state i is independent of j, giving us an alternative way of constructing a transition matrix such that the objective function can be factorized:

$$J(\theta) = \frac{1}{|\mathcal{D}|} \sum_{j \in \mathcal{D}} \sum_{i \notin \mathcal{D}} g_i \left(\frac{g_j}{g_i}\right)^{\frac{1}{2}} \exp\left(\frac{1}{2}\big(F_j(\mathbf{x};\theta) - F_i(\mathbf{x};\theta)\big)\right) \tag{15}$$

$$= \left(\frac{1}{|\mathcal{D}|}\sum_{j\in\mathcal{D}} \exp\left(\frac{1}{2}\big(F_j(\mathbf{x};\theta) + \log g_j\big)\right)\right)\left(\sum_{i\notin\mathcal{D}} g_i \exp\left(\frac{1}{2}\big(-F_i(\mathbf{x};\theta) + \log g_i\big)\right)\right) \tag{16}$$

where $\left(\frac{g_j}{g_i}\right)^{\frac{1}{2}}$ is a scaling term required to counterbalance the difference between g_i and g_j. The independence in the connectivity function allows us to factor all the j terms in 15 out of the inner sum, leaving us with a product of sums, something we could not achieve with 1-bit flip connectivity since the connection to state i depends on it being a neighbor of state j. Note that, intuitively, learning is facilitated by connecting data states to states that are probable under the model (i.e. to contrast the divergence). Therefore, we can use $p(v;\theta)$ to approximate g_i. In practice, for each iteration n of learning, we need the g_i and g_j terms to act as constants with respect to updating θ, and thus we sample them from $p(v;\theta^{n-1})$. We can then rewrite the objective function as $J(\theta) = J_{\mathcal{D}}(\theta)J_{\mathcal{S}}(\theta)$

$$J_{\mathcal{D}}(\theta) = \left(\frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}} \exp\left[\frac{1}{2}\big(F(\mathbf{x};\theta) - F(\mathbf{x};\theta^{n-1})\big)\right]\right) \tag{17}$$

$$J_{\mathcal{S}}(\theta) = \left(\frac{1}{|\mathcal{S}|}\sum_{\mathbf{x}'\in\mathcal{S}} \exp\left[\frac{1}{2}\big(-F(\mathbf{x}';\theta) + F(\mathbf{x}';\theta^{n-1})\big)\right]\right) \tag{18}$$

where $\mathcal{S}$ is the sampled set from $p(\mathbf{v};\theta^{n-1})$, and the normalization terms in $\log g_j$ and $\log g_i$ cancel out. Note we use the θ^{n-1} notation to refer to the parameters at the previous iteration, and simply θ for the current iteration.

4.3 Persistent Minimum Probability Flow

There are several ways of sampling "fantasy particles" from $p(\mathbf{v};\theta^{n-1})$. Notice that taking the data distribution with respect to θ^{n-1} is necessary for stable learning.

Previously, persistent contrastive divergence (PCD) was developed to improve CD-k learning [13]. Similarly, persistence can be applied to sampling in MPF connectivity functions. For each update, we pick a new sample based on a MCMC sampler which starts from previous samples. Then we update θ^n, which satisfies $J(\theta^n) \leq J(\theta^{n-1})$ [10]. The pseudo-code for persistent MPF is the same as Factored MPF except for drawing new samples, which is indicated by square brackets in Algorithm 2.

As we will show, using persistence in MPF is important for achieving faster convergence in learning. While the theoretical formulation of MPF guarantees eventual convergence, the focus on minimizing the initial probability flow will have little effect if the sampler mixes too slowly.

Algorithm 2. Factored [Persistent] MPF learning with probabilistic connectivity.

```
– for each epoch $n$ do
    1. Draw a new sample $S^n$ based on $S^0$ $\left[S^{n-1}\right]$ using an MCMC sampler.
    2. Compute $J_{\mathcal{S}}(\theta)$
    3. for each training example $d \in \mathcal{D}$ do
        (a) Compute $J_d(\theta)$. The cost function for $d$ is $J(\theta) = J_d(\theta) J_{\mathcal{S}}(\theta)$
        (b) Compute the gradient of the cost function,
            $\frac{\partial J(\theta)}{\partial \theta} = J_{\mathcal{S}}(\theta) J_d(\theta) \frac{\partial F_d(\theta)}{\partial \theta}$
                $+ \frac{1}{|\mathcal{S}|} J_d \sum_{x' \in \mathcal{S}} \left( \frac{\partial F(x')}{\partial \theta} \exp\left[\frac{1}{2}\left(F(\mathbf{x}'; \theta) - F(\mathbf{x}'; \theta^{n-1})\right)\right] \right)$
        (c) Update parameters via gradient descent with $\theta \leftarrow \theta - \lambda \nabla J(\theta)$
  end for
```

5 Experiments

We conducted the first empirical study of MPF under different types of connectivity as discussed in Section 4. We compared our results to CD-k with varying values for K. We analyzed the MPF variants based on training RBMs and assessed them quantitatively and qualitatively by comparing the log-likelihoods of the test data and samples generated from model. For the experiments, we denote the 1-bit flip, factorized, and persistent methods as MPF-1flip, FMPF, and PMPF, respectively. The goals of these experiments are to

1. Compare among MPF algorithms under different connectivities; and
2. Compare between MPF and CD-k.

In our experiments, we considered the MNIST and CalTech Silhouette datasets. MNIST consists of 60,000 training and 10,000 test images of size 28 $\times$ 28 pixels containing handwritten digits from the classes 0 to 9. The pixels in MNIST are binarized based on thresholding. From the 60,000 training examples, we set aside 10,000 as validation examples to tune the hyperparameters in our models. The CalTech Silhouette dataset contains the outlines of objects from the CalTech101 dataset, which are centred and scaled on a 28 $\times$ 28 image plane and rendered as filled black regions on a white background creating a silhouette of each object. The training set consists of 4,100 examples, with at least 20 and at most 100 examples in each category. The remaining instances were split evenly between validation and testing[2]. Hyperparameters such as learning rate, number of epochs, and batch size were selected from discrete ranges and chosen based on a held-out validation set. The learning rate for FMPF and PMPF were chosen from the range [0.001, 0.00001] and the learning rate for 1-bit flip was chosen from the range [0.2, 0.001].

[2] More details on pre-processing the CalTech Silhouettes can be found in http://people.cs.umass.edu/~marlin/data.shtml

Table 1. Experimental results on MNIST using 11 RBMs with 20 hidden units each. The average training and test log-probabilities over 10 repeated runs with random parameter initializations are reported.

Method	Average log Test	Average log Train	Time(s)	Batchsize
CD1	-145.63 ± 1.30	-146.62 ± 1.72	831	100
PCD	**-136.10 ± 1.21**	**-137.13 ± 1.21**	2620	300
MPF-1flip	-141.13 ± 2.01	-143.02 ± 3.96	2931	75
CD10	-135.40 ± 1.21	-136.46 ± 1.18	17329	100
FMPF10	-136.37 ± 0.17	-137.35 ± 0.19	12533	60
PMPF10	-141.36 ± 0.35	-142.73 ± 0.35	11445	25
FPMPF10	**-134.04 ± 0.12**	**-135.25 ± 0.11**	22201	25
CD15	-134.13 ± 0.82	-135.20 ± 0.84	26723	100
FMPF15	-135.89 ± 0.19	-136.93 ± 0.18	18951	60
PMPF15	-138.53 ± 0.23	-139.71 ± 0.23	13441	25
FPMPF15	**-133.90 ± 0.14**	**-135.13 ± 0.14**	27302	25
CD25	-133.02 ± 0.08	-134.15 ± 0.08	46711	100
FMPF25	-134.50 ± 0.08	-135.63 ± 0.07	25588	60
PMPF25	-135.95 ± 0.13	-137.29 ± 0.13	23115	25
FPMPF25	**-132.74 ± 0.13**	**-133.50 ± 0.11**	50117	25

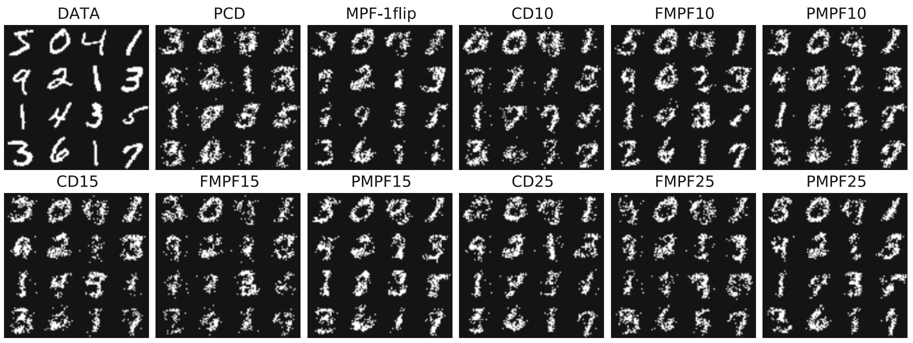

Fig. 1. Samples generated from the training set. Samples in each panel are generated by RBMs trained under different paradigms as noted above each image.

5.1 MNIST - Exact Log Likelihood

In our first experiment, we trained eleven RBMs on the MNIST digits. All RBMs consisted of 20 hidden units and 784 (28×28) visible units. Due to the small number of hidden variables, we calculated the exact value of the partition function by explicitly summing over all visible configurations. Five RBMs were learned by PCD1, CD1, CD10, CD15, and CD25. Seven RBMs were learned by 1 bit flip, FMPF, and FPMPF[3]. Block Gibbs sampling is required for FMPF-k and FPMPF-k similar to CD-k training, where the number of steps is given by k.

The average log test likelihood values of RBMs with 20 hidden units are presented in Table 1. This table gives a sense of the performance under different types of MPF dynamics when the partition function can be calculated exactly. We observed that PMPF consistently achieved a higher log-likelihood than FMPF. MPF with 1 bit flip was very fast but gave poor performance

[3] FPMPF is the composition of the FMPF and PMPF connectivities.

Table 2. Experimental results on MNIST using 11 RBMs with 200 hidden units each. The average estimated training and test log-probabilities over 10 repeated runs with random parameter initializations are reported. Likelihood estimates are made with CSL [2] and AIS [8].

	CSL		AIS			
Method	Avg. log Test	Avg. log Train	Avg. log Test	Avg. log Train	Time(s)	Batchsize
CD1	-138.63 ± 0.48	-138.70 ± 0.45	-98.75 ± 0.66	-98.61 ± 0.66	1258	100
PCD1	**-114.14 ± 0.26**	**-114.13 ± 0.28**	**-88.82 ± 0.53**	**-89.92 ± 0.54**	2614	100
MPF-1flip	-179.73 ± 0.085	-179.60 ± 0.07	-141.95 ± 0.23	-142.38 ± 0.74	4575	75
CD10	-117.74 ± 0.14	-117.76 ± 0.13	-91.94 ± 0.42	-92.46 ± 0.38	24948	100
FMPF10	-115.11 ± 0.09	-115.10 ± 0.07	-91.21 ± 0.17	-91.39 ± 0.16	24849	25
PMPF10	-114.00 ± 0.08	-113.98 ± 0.09	-89.26 ± 0.13	-89.37 ± 0.13	24179	25
FPMPF10	**-112.45 ± 0.03**	**-112.45 ± 0.03**	**-83.83 ± 0.23**	**-83.26 ± 0.23**	24354	25
CD15	-115.96 ± 0.12	-115.21 ± 0.12	-91.32 ± 0.24	-91.87 ± 0.21	39003	100
FMPF15	-114.05 ± 0.05	-114.06 ± 0.05	-90.72 ± 0.18	-90.93 ± 0.20	26059	25
PMPF15	-114.02 ± 0.11	-114.03 ± 0.09	-89.25 ± 0.17	-89.85 ± 0.19	26272	25
FPMPF15	**-112.58 ± 0.03**	**-112.60 ± 0.02**	**-83.27 ± 0.15**	**-83.84 ± 0.13**	26900	25
CD25	-114.50 ± 0.10	-114.51 ± 0.10	-91.36 ± 0.26	-91.04 ± 0.25	55688	100
FMPF25	-113.07 ± 0.06	-113.07 ± 0.07	-90.43 ± 0.28	-90.63 ± 0.27	40047	25
PMPF25	-113.70 ± 0.04	-113.69 ± 0.04	-89.21 ± 0.14	-89.79 ± 0.13	52638	25
FPMPF25	**-112.38 ± 0.02**	**-112.42 ± 0.02**	**-83.25 ± 0.27**	**-83.81 ± 0.28**	53379	25

compared to FMPF and PMPF. We also observed that MPF-1flip outperformed CD1. FMPF and PMPF always performed slightly worse than CD-k training with the same number of Gibbs steps. However, FPMPF always outperformed CD-k. The advantage of FPMPF in this case may reflect the increased effective number of entries in the transition matrix.

One advantage of FMPF is that it converges much quicker than CD-k or PMPF. This is because we used twice many samples as PMPF as mentioned in Section 4.3. Figure 1 shows initial data and the generated samples after running 100 Gibbs steps from each RBM. PMPF produces samples that are visually more appealing than the other methods.

5.2 MNIST - Estimating Log Likelihood

In our second set of experiments, we trained RBMs with 200 hidden units. We trained them exactly as described in Section 5.1. These RBMs are able to generate much higher-quality samples from the data distribution, however, the partition function can no longer be computed exactly.

In order to evaluate the model quantitatively, we estimated the test log-likelihood using the Conservative Sampling-based Likelihood estimator (CSL) [2] and annealed importance sampling (AIS) [8]. Given well-defined conditional probabilities $P(\mathbf{v}|\mathbf{h})$ of a model and a set of latent variable samples S collected from a Markov chain, CSL computes

$$\log \hat{f}(\mathbf{v}) = \log \text{mean}_{h \in S} P(\mathbf{v}|\mathbf{h}). \tag{19}$$

The advantage of CSL is that sampling latent variables $\mathbf{h}$ instead of $\mathbf{v}$ has the effect of reducing the variance of the estimator. Also, in contrast to annealed importance sampling (AIS) [8], which tends to overestimate, CSL is much more conservative in its estimates. However, most of the time, CSL is far off from the

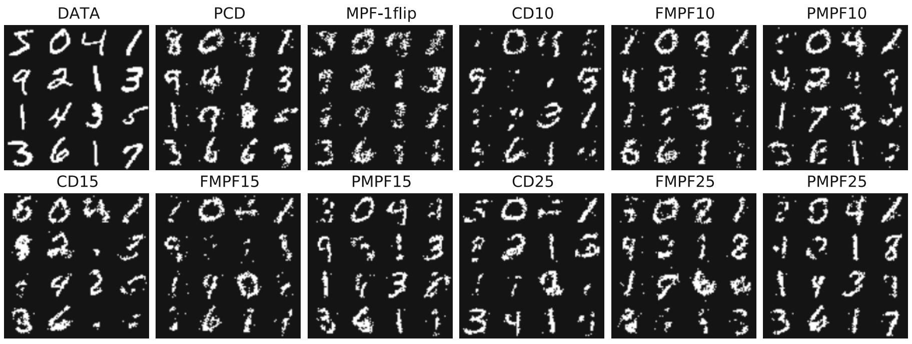

Fig. 2. Samples generated from the training set. Samples in each panel are generated by RBMs trained under different paradigms as noted above each image.

true estimator, so we bound our negative log-likelihood estimate from above and below using both AIS and CSL.

Table 2 demonstrates the test log-likelihood of various RBMs with 200 hidden units. The ranking of the different training paradigms with respect to performance was similar to what we observed in Section 5.1 with PMPF emerging as the winner. However, contrary to the first experiment, we observed that MPF with 1 bit flip did not perform well. Moreover, FMPF and PMPF both tended to give higher test log-likelihoods than CD-k training. Smaller batch sizes worked better with MPF when the number of hiddens was increased. Once again, we observed smaller variances compared to CD-k with both forms of MPF, especially with FMPF. We noted that FMPF and PMPF always have smaller variance compared to CD-k. This implies that FMPF and PMPF are less sensitive to random weight initialization. Figure 2 shows initial data and generated samples after running 100 Gibbs steps for each RBM. PMPF clearly produces samples that look more like digits.

5.3 Caltech 101 Silhouettes - Estimating Log Likelihood

Finally, we evaluated the same set of RBMs on the Caltech-101 Silhouettes dataset. Compared to MNIST, this dataset contains much more diverse structures with richer correlation among the pixels. It has 10 times more categories, contains less training data per category, and each object covers more of the image. For these reasons, we use 500 hidden units per RBM. The estimated average log-likelihood of train and test data is presented in Table 3.

The results for Caltech 101 Silhouettes are consistent with MNIST. In every case, we observed a larger margin between PMPF and CD-k when the number of sampling steps was smaller. In addition, the single bit flip technique was not particularly successful, especially as the number of latent variables grew. We speculate that the reason for this might have to do with the slow rate of convergence for the dynamic system. Moreover, PMPF works better than FMPF for similar reasons. By having persistent samples as the learning progresses, the

Table 3. Experimental results on Caltech-101 Silhouettes using 11 RBMs with 500 hidden units each. The average estimated training and test log-probabilities over 10 repeated runs with random parameter initializations are reported. Likelihood estimates are made with CSL [2] and AIS [8].

	CSL		AIS			
Method	Avg. log Test	Avg. log Train	Avg. log Test	Avg. log Train	Time(s)	Batchsize
CD1	-251.30 ± 1.80	-252.04 ± 1.56	-141.87 ± 8.80	-142.88 ± 8.85	300	100
PCD1	**-199.89 ± 1.53**	**-199.95 ± 1.31**	**-124.56 ± 0.24**	**-116.56 ± 2.40**	784	100
MPF-1flip	-281.55 ± 1.68	-283.03 ± 0.60	-164.96 ± 0.23	-170.92 ± 0.20	505	100
CD10	-207.77 ± 0.92	-207.16 ± 1.18	-128.17 ± 0.20	-120.65 ± 0.19	4223	100
FMPF10	-211.30 ± 0.84	-211.39 ± 0.90	-135.59 ± 0.16	-135.57 ± 0.18	2698	20
PMPF10	-203.13 ± 0.12	-203.14 ± 0.10	-128.85 ± 0.15	-123.06 ± 0.15	7610	20
FPMPF10	**-200.36 ± 0.16**	**-200.16 ± 0.16**	**-123.35 ± 0.16**	**-108.81 ± 0.15**	11973	20
CD15	-205.12 ± 0.87	-204.87 ± 1.13	-125.08 ± 0.24	-117.09 ± 0.21	6611	100
FMPF15	-210.66 ± 0.24	-210.19 ± 0.30	-130.28 ± 0.14	-128.57 ± 0.15	3297	20
PMPF15	-201.47 ± 0.13	-201.67 ± 0.10	-127.09 ± 0.10	-121 ± 0.12	9603	20
FPMPF15	**-198.59 ± 0.17**	**-198.66 ± 0.17**	**-122.33 ± 0.13**	**-107.88 ± 0.14**	18170	20
CD25	-201.56 ± 0.11	-201.50 ± 0.13	-124.80 ± 0.20	-117.51 ± 0.23	13745	100
FMPF25	-206.93 ± 0.13	-206.86 ± 0.11	-129.96 ± 0.07	-127.15 ± 0.07	10542	10
PMPF25	-199.53 ± 0.11	-199.51 ± 0.12	-127.81 ± 020	-122.23 ± 0.17	18550	10
FPMPF25	**-198.39 ± 0.0.16**	**-198.39 ± 0.17**	**-122.75 ± 0.13**	**-108.32 ± 0.12**	23998	10

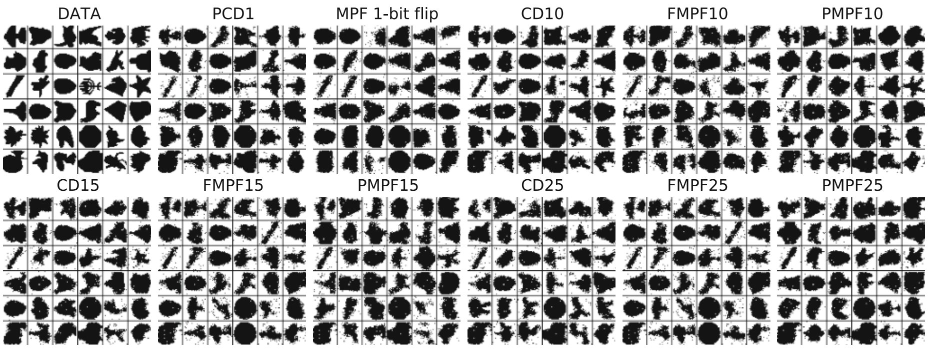

Fig. 3. Random samples generated by RBMs with different training procedures.

dynamics always begin closer to equilibrium, and hence converge more quickly. Figure 3 shows initial data and generated samples after running 100 Gibbs steps for each RBM on Caltech28 dataset.

6 Conclusion

MPF is an unsupervised learning algorithm that can be employed off-the-shelf to any energy-based model. It has a number of favourable properties but has not seen application proportional to its potential. In this paper, we first expounded on MPF and its connections to CD-k training, which allowed us to gain a better understanding and perspective to CD-k. We proved a general form for the transition matrix such that the equilibrium distribution converges to that of an RBM. This may lead to future extensions of MPF based on the choice of $o(\cdot)$ in Equation 12.

One of the merits of MPF is that the choice of designing a dynamic system by defining a connectivity function is left open as long as it satisfies the fixed point equation. Additionally, it should scale similarly to CD-k and its variants when increasing the number of visible and hidden units. We thoroughly explored three different connectivity structures, noting that connectivity can be designed inductively or by sampling. Finally, we showed empirically that MPF, and in particular, PMPF, outperforms CD-k for training generative models. Until now, RBM training was dominated by methods based on CD-k including PCD; however, our results indicate that MPF is a practical and effective alternative.

References

1. Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I.J., Bergeron, A., Bouchard, N., Bengio, Y.: Theano: new features and speed improvements. In: Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop (2012)
2. Bengio, Y., Yao, L., Cho, K.: Bounding the test log-likelihood of generative models. In: Proceedings of the International Conference on Learning Representations (ICLR) (2013)
3. Besag, J.: Statistical analysis of non-lattice data. The Statistician **24**, 179–195 (1975)
4. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation **14**, 1771–1880 (2002)
5. Hyvärinen, A.: Estimation of non-normalized statistical models by score matching. Journal of Machine Learning Research **6**, 695–709 (2005)
6. MacKay, D.J.C.: Failures of the one-step learning algorithm (2001). http://www.inference.phy.cam.ac.uk/mackay/abstracts/gbm.html, unpublished Technical Report
7. Marlin, B.M., de Freitas, N.: Asymptotic efficiency of deterministic estimators for discrete energy-based models: ratio matching and pseudolikelihood. In: Proceedings of the Uncertainty in Artificial Intelligence (UAI) (2011)
8. Salakhutdinov, R., Murray, I.: On the quantitative analysis of deep belief networks. In: Proceedings of the International Conference of Machine Learning (ICML) (2008)
9. Smolensky, P.: Information processing in dynamical systems: foundations of harmony theory. In: Parallel Distributed Processing: Volume 1: Foundations, pp. 194–281. MIT Press (1986)
10. Sohl-Dickstein, J.: Persistent minimum probability flow. Tech. rep, Redwood Centre for Theoretical Neuroscience (2011)
11. Sohl-Dickstein, J., Battaglino, P., DeWeese, M.R.: Minimum probability flow learning. In: Proceedings of the International Conference of Machine Learning (ICML) (2011)
12. Sutskever, I., Tieleman, T.: On the convergence properties of contrastive divergence. In: Proceedings of the AI & Statistics (AI STAT) (2009)
13. Tieleman, T., Hinton, G.E.: Using fast weights to improve persistent contrastive divergence. In: Proceedings of the International Conference of Machine Learning (ICML) (2009)
14. Tierney, L.: Markov chains for exploring posterior distributions. Annals of Statistics **22**, 1701–1762 (1994)

A Minimum Probability Flow

A.1 Dynamics of The Model

Theorem 2. *1 Suppose $p_j^{(\infty)}$ is the probability of state j and $p_i^{(\infty)}$ is the probability of state i. Let the transition matrix be*

$$\Gamma_{ij} = g_{ij} \exp\left(\frac{o(F_i - F_j) + 1}{2}(F_j - F_i)\right) \tag{20}$$

such that $o(\cdot)$ is any odd function, where g_{ij} is the symmetric connectivity between the states i and j. Then this transition matrix satisfies detailed balance in Equation 11.

Proof. By cancelling out the partition function, the detailed balance Equation 11 can be formulated as

$$\Gamma_{ji} \exp(-F_i) = \Gamma_{ij} \exp(-F_j) \tag{21}$$

where $F_i = F(\mathbf{v} = i; \theta)$. By substituting the transition matrix defined in Equation 12, we arrive at the following expression after some straightforward manipulation:

$$\Gamma_{ji} \exp(-F_i) / \Gamma_{ij} \exp(-F_j)) = 1$$

$$\exp\left(\frac{o(F_i - F_j) + 1}{2}(F_j - F_i) - F_i\right) / \exp\left(\frac{o(F_j - F_i) + 1}{2}(F_i - F_j) - F_j\right) = 1$$

$$\exp\left(\frac{o(F_i - F_j) + 1}{2}(F_j - F_i) - F_i - \frac{o(F_j - F_i) + 1}{2}(F_i - F_j) + F_j\right) = 1$$

$$\frac{o(F_i - F_j) + 1}{2}(F_j - F_i) - F_i - \frac{o(F_j - F_i) + 1}{2}(F_i - F_j) + F_j = 0$$

$$(F_i - F_j)\left(\frac{o(F_i - F_j) + 1}{2} + \frac{o(F_j - F_i) + 1}{2} - 1\right) = 0$$

$$(F_i - F_j)\left(\frac{o(F_i - F_j)}{2} + \frac{o(F_j - F_i)}{2}\right) = 0$$

Notice that since $o(\cdot)$ is an odd function, this makes the term $\left(\frac{o(F_i - F_j)}{2} + \frac{o(F_j - F_i)}{2}\right) = 0$. Therefore, the detailed balance criterion is satisfied.