# Optimal Perceptron Learning: an Online Bayesian Approach

*Sara A. Solla*[1,2,3] *and Ole Winther*[3,4]

[1]Physics and Astronomy, Northwestern University, Evanston, IL 60208, USA
[2]Physiology, Northwestern University Medical School, Chicago, IL 60611, USA
[3]CONNECT, The Niels Bohr Institute, 2100 Copenhagen Ø, Denmark
[4]Theoretical Physics II, Lund University, S-223 62 Lund, Sweden

**Abstract**

The recently proposed Bayesian approach to online learning is applied to learning a rule defined as a noisy single layer perceptron with either continuous or binary weights. In the Bayesian online approach the exact posterior distribution is approximated by a simpler parametric posterior that is updated online as new examples are incorporated to the dataset. In the case of continuous weights, the approximate posterior is chosen to be Gaussian. The computational complexity of the resulting online algorithm is found to be at least as high as that of the Bayesian offline approach, making the online approach less attractive. A Hebbian approximation based on casting the full covariance matrix into an isotropic diagonal form significantly reduces the computational complexity and yields a previously identified optimal Hebbian algorithm. In the case of binary weights, the approximate posterior is chosen to be a biased binary distribution. The resulting online algorithm is derived and shown to outperform several other online approaches to this problem.

## 1 Introduction

Neural networks are adaptive systems characterized by a set of parameters $\mathbf{w}$, the weights and biases that specify the connectivity among the *neuronal* computational elements. Of particular interest is the ability of these systems to learn from examples. Traditional formulations of the learning problem are based on a dynamical prescription for the adaptation of the parameters $\mathbf{w}$. The learning process thus generates a trajectory in $\mathbf{w}$ space that starts from a random initial assignment $\mathbf{w}^0$ and leads to a specific $\mathbf{w}^\star$ that is in some sense *optimal*.

Two learning modalities need to be distinguished: *offline* and *online*. In offline or batch learning, all examples in the training set are used at every time step to update the current values of the network parameters $\mathbf{w}$. In online learning, the parameters are updated after the presentation of each example.

Algorithmic approaches that result in specific values $\mathbf{w}^\star$ for the network parameters are to be contrasted to a probabilistic Bayesian formulation based on the information provided by a probability distribution over the parameter space $\mathbf{w}$. In the Bayesian approach, the learning process is described by the evolution of a prior distribution $p(\mathbf{w})$ into a posterior that incorporates the information provided by data given in the form of training examples.

The posterior, constructed as the appropriately normalized product of the prior and the likelihood of the data, quantifies the a posteriori belief in each possible setting of the parameters $\mathbf{w}$, and it plays a crucial role in the prediction of new data. The Bayesian prediction probability is computed through a weighted average over the parameter space $\mathbf{w}$, and it is the posterior that assigns a weight to every possible parameter setting. Predictions based on this approach are optimal in the sense of yielding the minimal average prediction error; the average is to be taken over all possible data sources within the family defined by the prior. The optimality of Bayesian predictors thus holds under the assumption that the prior beliefs are correct.

The procedure is intrinsically *offline*, as the computation of the posterior requires knowledge of the entire training set. A controlled approximation that leads to an online implementation of Bayesian learning has been recently proposed (Opper 1996, Winther & Solla 1998), and some of its implications are explored here. In Bayesian *online* learning, the true posterior is approximated by a simpler parametric distribution; the parameters that characterize the approximate posterior are updated online. The goal is to speed up the process by replacing averages over complicated posterior distributions by averages over simpler approximate forms. For the procedure to be meaningful, the approximate posterior needs to capture the essential features of the true posterior.

The Bayesian approach to online learning investigated here is to be contrasted to other approaches to online learning that have received considerable recent attention within the statistical physics community. These alternative procedures emphasize the dynamical adaptation of the parameters $\mathbf{w}$. One approach is based on Hebbian learning rules, for which the update vector $\Delta\mathbf{w}$ at every time step is in the direction of the input vector of the corresponding example. A modulation function that controls the instantaneous amplitude of the adaptation step is determined so as to maximize the decrease of the generalization error, either locally for the current example (Kinouchi & Caticha 1992) or globally for a set of examples presented during a specified time interval $\Delta t$ (Saad & Rattray 1997). Such methods require some degree of information about the generalization error; the Bayesian approach is more fundamental in that it makes no demands on the availability of such information.

A Bayesian algorithm only requires the specification of a prior; the algorithm then guarantees the optimal use of such a priori information. Unfortunately, the intrinsic complexity of the averaging steps involved in Bayesian learning often render such calculations intractable. Here we focus on a simple learning scenario, that of the perceptron, for which the necessary averages can be performed analytically in the limit of large system size. Our results show that the tensorial update rule that emerges naturally from the Bayesian approach outperforms the so-called optimal Hebbian rule. A Hebbian approximation to the full Bayesian update rule for the perceptron does reproduce the previously obtained optimal Hebbian algorithm (Kinouchi & Caticha 1992). Such Hebbian approximations might be necessary if the computational complexity of online Bayesian learning is to be kept noticeable smaller than that of its traditional offline version.

A clear demonstration of the generic power of the Bayesian online approach is still to come, as it requires an extension to the case of complex learning scenarios involving multilayer networks. We are currently working on the case of two layer networks, which becomes analytically tractable in the limit of a large number of hidden units. Such results need to be obtained and compared to those available for such two layer architectures when trained by simple gradient descent (Saad & Solla 1995) as well as its optimized local (Vicente & Caticha 1997) or global (Saad & Rattray 1997) versions.

The chapter is organized as follows. Section 2 contains a review of the basic principles of Bayesian inference. Section 3 develops an online implementation of Bayesian learning. The approach is applied in section 4 to the problem of learning a noisy simple perceptron with continuous weights via a Gaussian approximation to the posterior distribution. The performance, computational complexity, and storage requirements of this online Bayesian approach are compared to those for the Hebb approximation and the mean field approach to offline Bayesian learning. A biased binary distribution is proposed in section 5 as an approximate posterior for the problem of learning a noisy simple perceptron with binary weights. Update rules are obtained, and the resulting algorithm is compared to a variety of other approaches to this problem. The chapter concludes with a discussion of current and future work in section 6.

## 2 Bayesian Learning

Bayesian inference provides a framework for formulating the problem of learning from examples in purely probabilistic terms. Here we briefly review this formulation for the case of classification problems, for which each example consists of an input vector $\mathbf{s}$ and an associated classification label $\tau$. A training set of size $t$ is denoted by $D_t = \{(\tau^\mu, \mathbf{s}^\mu), 1 \le \mu \le t\}$. Training examples are assumed to be independently drawn from a distribution $p((\tau, \mathbf{s})|\mathbf{w})$, where $\mathbf{w}$ is the unknown parameter vector, to be estimated from the data $D_t$. The inference procedure consists of two steps.

The first step is to assign a probability or *likelihood* to the training examples. The statistical independence of the individual examples results in a multiplicative form for the likelihood of the training set,

$$p(D_t|\mathbf{w}) = \prod_\mu p((\tau^\mu, \mathbf{s}^\mu)|\mathbf{w}) \ . \tag{2.1}$$

We write $p((\tau, \mathbf{s})|\mathbf{w}) = p(\tau|\mathbf{w}, \mathbf{s})p(\mathbf{s})$, where $p(\tau|\mathbf{w}, \mathbf{s})$ models the input-output relation, while the input distribution $p(\mathbf{s})$ is independent of $\mathbf{w}$.

The second step in Bayesian inference is to assign a prior probability $p(\mathbf{w})$ to the unknown parameters $\mathbf{w}$. The *true values* of the parameters, i.e. the ones actually used in the generation of the data, are assumed to have nonzero prior probability. It is in this sense that the analysis is restricted to the case of *realizable* learning scenarios.

Here we focus our analysis on the case of a simple perceptron, for which classification labels $\tau = \pm 1$ are generated through $\tau = f(\mathbf{w}, \mathbf{s}) = \mathrm{sign}(\mathbf{w} \cdot \mathbf{s})$. Both $\mathbf{w}$ and $\mathbf{s}$ are $N$-dimensional vectors with components $\{w_i, 1 \le i \le N\}$ and $\{s_i, 1 \le i \le N\}$, respectively. Noise is introduced through a label flip with probability $\kappa$. The likelihood of output $\tau$ is thus given by

$$\begin{aligned} p(\tau|\mathbf{w}, \mathbf{s}) &= \kappa\ \Theta(-\tau f(\mathbf{w}, \mathbf{s})) + (1-\kappa)\ \Theta(\tau f(\mathbf{w}, \mathbf{s})) \\ &= \kappa + (1 - 2\kappa)\ \Theta(\tau\ \mathbf{w} \cdot \mathbf{s}) \ , \end{aligned} \tag{2.2}$$

where $\Theta()$ is the step-function, $\Theta(x) = 1$ for $x > 0$ and $\Theta(x) = 0$ otherwise.

We consider two possible priors: a spherical Gaussian

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{N/2}} \exp\left(-\frac{\mathbf{w} \cdot \mathbf{w}}{2}\right) \tag{2.3}$$

and a binary distribution

$$p(\mathbf{w}) = \prod_i \left[\frac{1}{2}\ \delta(w_i - 1) + \frac{1}{2}\ \delta(w_i + 1)\right] \ . \tag{2.4}$$

These prior distributions represent prior knowledge, as opposed to knowledge coming from the training data.

Bayes rule provides a prescription for writing the posterior distribution $p(\mathbf{w}|D_t)$ in terms of the prior and the likelihood of the training set:

$$p(\mathbf{w}|D_t) = \frac{\prod_\mu p(\tau^\mu|\mathbf{w}, \mathbf{s}^\mu)\ p(\mathbf{w})}{\int d\mathbf{w}\ \prod_\mu p(\tau^\mu|\mathbf{w}, \mathbf{s}^\mu)\ p(\mathbf{w})} \ . \tag{2.5}$$

The posterior distribution quantifies our knowledge about $\mathbf{w}$ after the observation of the training data $D_t$.

The posterior distribution is used to compute the *predictive probability* for each possible output $\tau = \pm 1$ given a new input $\mathbf{s}$:

$$p(\tau|\mathbf{s}, D_t) = \int d\mathbf{w} \; p(\tau|\mathbf{w}, \mathbf{s}) \; p(\mathbf{w}|D_t) \; . \tag{2.6}$$

Predictions based on the Bayes algorithm are guaranteed to minimize the average prediction error through the choice of output label $\tau$ which maximizes the above prediction probability for a given input $\mathbf{s}$. For the type of classification problem considered here, the Bayes prediction is given by

$$\tau^{\text{Bayes}}(\mathbf{s}, D_t) = \text{sign}\left( \int d\mathbf{w} \, p(\mathbf{w}|D_t) \; \text{sign}\,(\mathbf{w} \cdot \mathbf{s}) \right) \; . \tag{2.7}$$

The process discussed in this section is intrinsically offline, as information on the full data set $D_t$ is needed to compute the posterior distribution that controls the average in Eq. (2.7).

# 3 Online Bayesian Learning

The problem we now face is that of adapting the Bayesian approach summarized in the preceding section so as to obtain an online version. Learning methods based on incorporating all information provided by the data into the current values of the network parameters $\mathbf{w}$ are easily adapted onto online versions: it suffices to use the information provided by a new example to update the current values of the network parameters. In a Bayesian formulation the information provided by the data is incorporated into a distribution over $\mathbf{w}$ space, and it is this distribution that needs to be updated in an online manner when a new example becomes available.

An online Bayesian learning algorithm requires a prescription for an online update of the posterior distribution. To achieve this goal, the exact posterior $p(\mathbf{w}|D_t)$ of Eq. (2.5) is approximated by a simple parametric distribution $p(\mathbf{w}|\mathbf{A}_t)$, where $\mathbf{A}_t$ refers to the current values of a set of parameters $\mathbf{A}$ which characterize the distribution (e.g. the first two moments of $\mathbf{w}$ for a Gaussian $p(\mathbf{w}|\mathbf{A})$). The online Bayesian procedure refers to the update of the distribution parameters $\mathbf{A}$ as opposed to the network parameters $\mathbf{w}$.

The online Bayesian algorithm becomes computationally advantageous over its offline version, Eq. (2.5), to the extent that the set $\mathbf{A}$ contains a small number of parameters, so that $\mathbf{A}_t$ can be interpreted as providing a compact encoding of the information contained in the training set $D_t$. A tension arises between the need for a complex parametrization to provide a reliable approximation to the true posterior of Eq. (2.5), and a simple parametrization to gain computational speed.

The resulting online Bayesian procedure consists of two steps:

1. **Add an example** – the current posterior is updated exactly according to Bayes rule:

$$p(\mathbf{w}|\mathbf{A}_t, (\tau^{t+1}, \mathbf{s}^{t+1})) = \frac{p(\tau^{t+1}|\mathbf{w}, \mathbf{s}^{t+1})\; p(\mathbf{w}|\mathbf{A}_t)}{\int d\mathbf{w}\, p(\tau^{t+1}|\mathbf{w}, \mathbf{s}^{t+1})\; p(\mathbf{w}|\mathbf{A}_t)} \; . \tag{3.1}$$

2. **Approximate** – the updated posterior is parametrized:

$$p(\mathbf{w}|\mathbf{A}_t, (\tau^{t+1}, \mathbf{s}^{t+1})) \to p(\mathbf{w}|\mathbf{A}_{t+1}) \; . \tag{3.2}$$

Some of the information provided by the new example $(\tau^{t+1}, \mathbf{s}^{t+1})$ is discarded in the parametrization step. The parametrization is constructed so as to minimize the resulting information loss. As discussed by Opper in this volume, there is no unique way of measuring this loss; but once a metric is chosen it is possible to quantify the dissimilarity between the exact updated posterior and any proposed approximation to it. It is thus possible to select the best among several possible parametrizations.

We chose to quantify the information loss through the *relative entropy* or *Kullback-Leibler distance* between the two probability distributions:

$$D\left[p(\mathbf{w}|\mathbf{A}_t, (\tau^{t+1}, \mathbf{s}^{t+1}))||p(\mathbf{w}|\mathbf{A}_{t+1})\right]$$

$$= \int d\mathbf{w}\; p(\mathbf{w}|\mathbf{A}_t, (\tau^{t+1}, \mathbf{s}^{t+1}))\; \ln \frac{p(\mathbf{w}|\mathbf{A}_t, (\tau^{t+1}, \mathbf{s}^{t+1}))}{p(\mathbf{w}|\mathbf{A}_{t+1})} \; . \tag{3.3}$$

For a Gaussian approximate posterior distribution, the minimization of the information loss is achieved by choosing $\mathbf{A}_{t+1}$ such that $p(\mathbf{w}|\mathbf{A}_t, (\tau^{t+1}, \mathbf{s}^{t+1}))$ and $p(\mathbf{w}|\mathbf{A}_{t+1})$ have the same first two moments (see also Opper, this volume).

To make the approximation consistent it is necessary that the initial distribution $p(\mathbf{w}|\mathbf{A}_0)$ does not exclude any of the values of $\mathbf{w}$ for which $p(\mathbf{w})$ is non-zero. The natural choice $p(\mathbf{w}|\mathbf{A}_0) = p(\mathbf{w})$ fulfills this condition. The iterative process of adding examples and approximating the posterior is illustrated in figure 1. In the space of all possible distributions $p(\mathbf{w})$, the plane represents the manifold of distributions parametrized through $p(\mathbf{w}|\mathbf{A})$. At every time $t$ the approximate posterior $p(\mathbf{w}|\mathbf{A}_t)$ is in this manifold, but the update rule that incorporates the information provided by the new example $(\tau^{t+1}, \mathbf{s}^{t+1})$ takes the distribution away from the parametric manifold. The approximation step involves a projection back onto the parametric manifold; the projection is controlled by a metric provided by the Kullback-Leibler distance.

In the rest of this article we present two simple applications of the Bayesian online approach to the problem of learning a rule defined by a simple perceptron with output flip noise, for which the likelihood is given by Eq. (2.2). We show that different choices of prior $p(\mathbf{w})$ result in quite different algorithms with different average behavior.

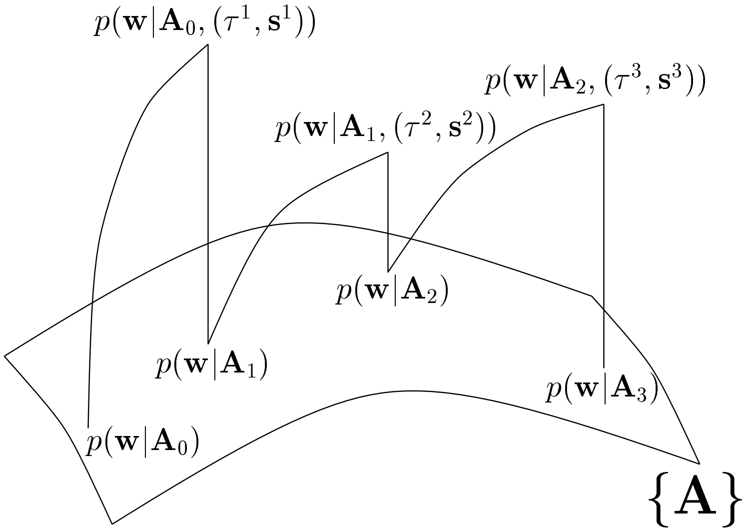


Figure 1. The update of the approximate posterior.

In section 4 we study the case of a Gaussian prior (see Eq. (2.3)) and choose a Gaussian approximation for the posterior. In section 5 the prior is binary (see Eq. (2.4)) and the approximate posterior is chosen to be a biased binary distribution.

# 4 Gaussian Approximate Posterior

The minimization of the Kullback-Leibler distance of Eq. (3.3) is achieved in the case of a Gaussian approximation to the posterior through the matching of the first two moments of the updated posterior distribution $p(\mathbf{w}|\mathbf{A}_t,(\tau^{t+1},\mathbf{s}^{t+1}))$, Eq. (3.1). The update rules can be derived through the use of partial integration (Opper 1996) to obtain:

$$\widetilde{\langle w_i \rangle} = \langle w_i \rangle + \sum_j C_{ij} \frac{\partial}{\partial \langle w_j \rangle} \ln \langle p(\tau|\mathbf{w},\mathbf{s}) \rangle \qquad (4.1)$$

$$\widetilde{C_{ij}} = C_{ij} + \sum_{kl} C_{il} C_{kj} \frac{\partial^2}{\partial \langle w_k \rangle \langle w_l \rangle} \ln \langle p(\tau|\mathbf{w},\mathbf{s}) \rangle \qquad (4.2)$$

for the update of the components $\langle w_i \rangle$ of the average weight vector and the elements $C_{ij} = \langle w_i w_j \rangle - \langle w_i \rangle \langle w_j \rangle$ of the covariance matrix. The following conventions have been adopted to simplify notation: superscripts are omitted when referring to the new example, so that $(\tau,\mathbf{s})$ refers to $(\tau^{t+1},\mathbf{s}^{t+1})$; averages $\langle \ldots \rangle$ are taken with respect to the approximate posterior $p(\mathbf{w}|\mathbf{A}_t)$ at time $t$; averages $\widetilde{\langle \ldots \rangle}$ are taken with respect to the approximate posterior $p(\mathbf{w}|\mathbf{A}_{t+1})$ at time $t+1$.

Note that $p(\tau|\mathbf{s},\mathbf{A}_t) = \langle p(\tau|\mathbf{w},\mathbf{s})\rangle = \int d\mathbf{w}\, p(\tau|\mathbf{w},\mathbf{s})\, p(\mathbf{w}|\mathbf{A}_t)$ is the online approximation to the predictive probability, Eq. (2.6).

The above update equations allow for a compact rewrite when applied to the simple perceptron, for which $p(\tau|\mathbf{w},\mathbf{s})$ of Eq. (2.2) depends on $\mathbf{w}$ only through the scalar field $h = \frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s}$. Then $p(\tau|\mathbf{w},\mathbf{s}) = p(\tau|h)$, and $\langle p(\tau|\mathbf{w},\mathbf{s})\rangle = \int d\mathbf{w}\, p(\mathbf{w}|\mathbf{A}_t)\, p(\tau|\mathbf{w},\mathbf{s}) = \int dh\, p(h|\mathbf{s},\mathbf{A}_t)\, p(\tau|h)$, with $p(h|\mathbf{s},\mathbf{A}_t) = \int d\mathbf{w}\, \delta(h - \frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s})\, p(\mathbf{w}|\mathbf{A}_t)$. Because the distribution $p(\mathbf{w}|\mathbf{A}_t)$ is chosen to be Gaussian, $p(h|\mathbf{s},\mathbf{A}_t)$ is also Gaussian,

$$p(h|\mathbf{s},\mathbf{A}_t) = \frac{1}{\sqrt{2\pi\lambda}} \exp\left(-\frac{(h-\langle h\rangle)^2}{2\lambda}\right) , \tag{4.3}$$

with mean $\langle h\rangle = \frac{1}{\sqrt{N}}\langle\mathbf{w}\rangle\cdot\mathbf{s}$ and variance $\lambda = \frac{1}{N}\mathbf{s}^T\mathbf{C}\mathbf{s}$.

We use the chain rule $\frac{\partial f(\langle h\rangle)}{\partial\langle\mathbf{w}\rangle} = \frac{1}{\sqrt{N}}\mathbf{s}\frac{\partial f(\langle h\rangle)}{\partial\langle h\rangle}$ to obtain

$$\begin{aligned} \widetilde{\langle\mathbf{w}\rangle} &= \langle\mathbf{w}\rangle + \frac{1}{\sqrt{N}}\,\mathbf{C}\mathbf{s}\,\frac{\partial}{\partial\langle h\rangle}\ln\langle p(\tau|h)\rangle && (4.4)\\ \widetilde{\mathbf{C}} &= \mathbf{C} + \frac{1}{N}\,\mathbf{C}\mathbf{s}\,\mathbf{s}^T\mathbf{C}\,\frac{\partial^2}{\partial\langle h\rangle^2}\ln\langle p(\tau|h)\rangle\ , && (4.5) \end{aligned}$$

for the update rules in matrix notation.

The predictive probability $\langle p(\tau|\mathbf{w},\mathbf{s})\rangle = \langle p(\tau|h)\rangle$ is given by:

$$\langle p(\tau|h)\rangle = \kappa + (1-2\kappa)\,\Phi\left(\tau\frac{\langle h\rangle}{\sqrt{\lambda}}\right) , \tag{4.6}$$

where

$$\Phi(x) = \int_{-\infty}^{x} \frac{dt}{\sqrt{2\pi}} e^{-(1/2)t^2}$$

is a sigmoidal error function. The Bayesian predictor of Eq. (2.7) for a new input $\mathbf{s}$ is given by

$$\tau^{\text{Bayes}}(\mathbf{s},\mathbf{A}_t) = \text{sign}(\langle\mathbf{w}\rangle\cdot\mathbf{s}) \tag{4.7}$$

for the Gaussian approximate posterior considered here.

We have thus obtained a Bayesian online algorithm for learning a perceptron rule with output noise; the corresponding Bayes predictor depends only upon the posterior mean of the weights.

It is relevant to analyze the computational complexity of the algorithm defined by Eqs. (4.4) and (4.5), and investigate how it scales with the number of input variables $N$ and the number $t$ of presented examples. The most computationally expensive operations for a given example $\mathbf{s}$ are those required to obtain $\mathbf{C}\mathbf{s}$, $\mathbf{s}\mathbf{s}^T$, and $\mathbf{s}^T\mathbf{C}\mathbf{s}$; all these operations are of order $\mathcal{O}(N^2)$. As expected, the update of all components of an $N\times N$ covariance matrix takes $\mathcal{O}(N^2)$ operations. The total computational cost is thus $\mathcal{O}(N^2t)$. The computational cost and performance of this algorithm will next be compared to those of the corresponding offline Bayesian algorithm and a further Hebbian approximation to the online Bayesian algorithm discussed above.

## Hebbian approximation to the online Bayesian algorithm

A Hebbian approximation to the full online Bayesian algorithm defined by Eqs. (4.4) and (4.5) follows from replacing the update factor $\mathbf{Cs}$ by its projection along the direction of the new input $\mathbf{s}$:

$$\mathbf{Cs} \to \mathbf{s}\frac{\mathbf{s}^T\mathbf{Cs}}{\mathbf{s}\cdot\mathbf{s}} \ .$$

The resulting update rules are those of a Hebbian algorithm with adaptable step size.

The average behavior of this algorithm can be easily analyzed in the thermodynamic limit $N \to \infty$ for spherical input distributions satisfying $\overline{s_i} = 0$ and $\overline{s_i s_j} = \delta_{ij}$ (where overline denotes an average over the input distribution). For spherical inputs in thermodynamic limit, the central limit theorem can be invoked to argue that the scalar $\mathbf{s}^T\mathbf{Cs}$ will be self averaging and can be replaced by $\overline{\mathbf{s}^T\mathbf{Cs}} = \langle \mathbf{w}\cdot\mathbf{w}\rangle - \langle\mathbf{w}\rangle\cdot\langle\mathbf{w}\rangle$, where averages are taken over the distribution for a new input $\mathbf{s}$ uncorrelated with the current $\mathbf{C}$ matrix. The update rules no longer require the computation of the full covariance matrix $\mathbf{C}$; it suffices to compute its trace $(\langle \mathbf{w}\cdot\mathbf{w}\rangle - \langle\mathbf{w}\rangle\cdot\langle\mathbf{w}\rangle)$. The computational complexity of this version of the algorithm is thus reduced to $\mathcal{O}(Nt)$.

It is of interest to note that this algorithm turns out to be equivalent to an optimal Hebb algorithm of the form $\widetilde{\mathbf{w}} = \mathbf{w} + F\mathbf{s}$, where $F$ is a scalar modulation function chosen so as to locally maximize the decrease of the generalization error for each example (Kinouchi & Caticha 1992). We shall show that the performance of this optimal Hebbian algorithm is inferior to that of the full Bayesian online algorithm presented before.

## Bayesian offline algorithm

Mean field equations for the Bayesian offline scenario have been derived for a simple perceptron (Opper & Winther 1996). These equations are expected to be valid in the thermodynamic limit, and they reduce to the form

$$\langle\mathbf{w}\rangle = \frac{1}{\sqrt{N}}\sum_\mu \mathbf{s}^\mu\langle x^\mu\rangle \tag{4.8}$$

$$\langle x^\mu\rangle = \frac{\partial}{\partial\langle h^\mu\rangle_\mu}\ln\langle p(\tau^\mu|h^\mu)\rangle_\mu \tag{4.9}$$

for spherical inputs. The sum in Eq. (4.8) is over training examples $\mu$, and $\langle\ldots\rangle_\mu$ in Eq. (4.9) refers to an average over a posterior distribution for which the $\mu$th example has been excluded; the corresponding training set is denoted as: $D_t\backslash(\tau^\mu, \mathbf{s}^\mu)$.

Consider the distribution of the *cavity field* $h^\mu$:

$$p(h^\mu|\mathbf{s}^\mu, D_t\backslash(\tau^\mu,\mathbf{s}^\mu)) = \int d\mathbf{w}\,\delta(h^\mu - \frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s}^\mu)\,p(\mathbf{w}|D_t\backslash(\tau^\mu,\mathbf{s}^\mu)) \ .$$

The mean field approximation rests on the assumption that this distribution is Gaussian in the thermodynamic limit:

$$p(h^\mu|\mathbf{s}^\mu, D_t \backslash (\tau^\mu, \mathbf{s}^\mu)) = \frac{1}{\sqrt{2\pi\lambda}} \exp\left(-\frac{(h^\mu - \langle h^\mu \rangle_\mu)^2}{2\lambda}\right) , \tag{4.10}$$

with $\lambda = \frac{1}{N}(\langle \mathbf{w} \cdot \mathbf{w} \rangle - \langle \mathbf{w} \rangle \cdot \langle \mathbf{w} \rangle)$ for uncorrelated input data. The mean $\langle h^\mu \rangle_\mu$ of the cavity field is related to the full posterior mean field $\langle h^\mu \rangle = \frac{1}{\sqrt{N}} \langle \mathbf{w} \rangle \cdot \mathbf{s}^\mu$, the *aligning field*, through

$$\langle h^\mu \rangle_\mu = \langle h^\mu \rangle - \lambda \langle x^\mu \rangle. \tag{4.11}$$

There are obvious similarities and differences between the online and offline approaches. The average $\langle p(\tau|h) \rangle$ in the online update Eqs. (4.4) and (4.5) is an average over a cavity field. In the offline approach, the Gaussianity of the cavity field follows from the central limit theorem. The mean field approximation is therefore expected to become exact in the thermodynamic limit. In online learning, the Gaussianity is a consequence of the choice of a Gaussian form for the approximate posterior. Note the similarity between Eq. (4.3) and Eq. (4.10); the correction (4.11) to the mean in Eq. (4.10) arises because in offline learning the posterior average is over all examples in the training set, whereas in online learning the example is discarded once it has been used to update the posterior. In online learning it is necessary to keep track of the covariance matrix because the inputs are discarded, whereas in offline learning the mean weight in Eq. (4.8) is spanned by the input vectors. This distinction has an important consequence for the computational complexity of the algorithm.

The set of non-linear mean field equations is usually solved by iteration. If we denote by $\hat{N}$ the number of iterations needed to obtain solutions with a desired degree of accuracy, the computational complexity of the algorithm scales like $\mathcal{O}(\hat{N}Nt)$. We expect $\hat{N}$ to scale at most with $N$. The full Bayesian online algorithm will thus have at least the same computational complexity as the Bayesian offline algorithm.

## Performance comparison

We now compare the performance of the three algorithms discussed in this section. Performance is measured through a computation of the generalization error, defined as the probability that the Bayes predictor $\tau^{\text{Bayes}}(\mathbf{s}, \mathbf{A}_t)$ of Eq. (4.7) disagrees with a teacher defined as a noisy simple perceptron with weight vector $\mathbf{v}$ and output $\tau = \eta\, \text{sign}(\mathbf{v} \cdot \mathbf{s})$, where $\eta = -1$ with probability $\kappa$, and $\eta = 1$ otherwise.

The generalization error is given by

$$\epsilon = \overline{\Theta\left(-\eta\ \text{sign}(\mathbf{v} \cdot \mathbf{s})\ \tau^{\text{Bayes}}(\mathbf{s}, \mathbf{A}_t)\right)} , \tag{4.12}$$

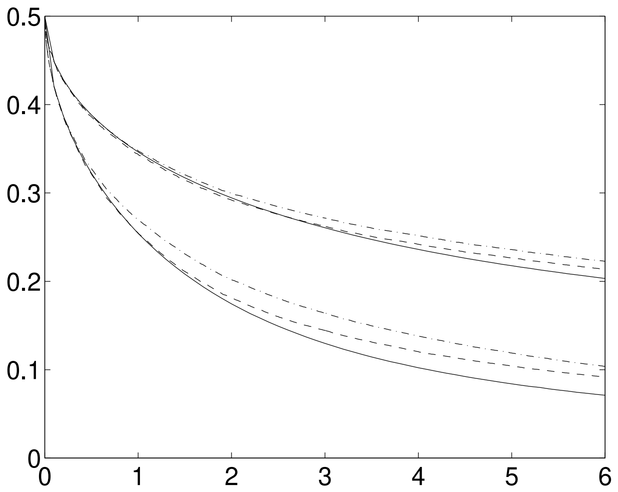


Figure 2. Learning curves ($\epsilon$ versus $\alpha$) for the online and offline algorithms for continuous weights. The three lower curves are for $\kappa = 0$ and the three upper curves are for $\kappa = 0.1$. The dash-dotted lines are for the Hebb approximation to the online algorithm, the dashed lines are for the full online algorithm, and the full lines are the theoretical predictions for the offline Bayes algorithm. Simulations were performed for $N = 100$ and averaged over 100 runs. Error bars are comparable to the line thickness.

where the average is to be taken over the input distribution from which new test examples can be drawn at random. This average is easily performed for a spherical input distribution, and leads to a simple result for the generalization error:

$$\epsilon = \kappa + (1 - 2\kappa)\frac{1}{\pi} \arccos\left(\frac{R}{\sqrt{qT}}\right) , \tag{4.13}$$

where the *order parameters* $q$, $R$, and $T$ are given by overlaps between the two weight vectors involved: $R = \frac{1}{N}\langle \mathbf{w}\rangle \cdot \mathbf{v}$, $q = \frac{1}{N}\langle \mathbf{w}\rangle \cdot \langle \mathbf{w}\rangle$, and $T = \frac{1}{N}\mathbf{v} \cdot \mathbf{v}$. Simulation results for all three algorithm are shown in figure 2; the generalization error $\epsilon$ is shown as a function of the normalized number of examples $\alpha = t/N$. No theoretical analysis is available at this time for the online Bayesian algorithm that incorporates full second order information through the update of all entries in the covariance matrix, Eqs. (4.4) and (4.5). The results of theoretical analysis available for the other two algorithms are in complete agreement within the scale of the figure with the numerical results shown in figure 2.

The simulation results for the two online algorithms allow for a performance comparison and show the significant benefit of using the full second order information. We expect this difference to be even larger for general, non-spherical input distributions.

For further comparison, we show in table 1 the asymptotic performance for $\kappa = 0$, the computational complexity, and the memory requirements for all three algorithms. Both online algorithms exhibit the same asymptotic performance for the spherical input distributions considered here, as the off-diagonal elements of the covariance matrix vanish asymptotically in this highly symmetric case. Discrepancies in favor of the full online algorithm are expected to appear for more general input distributions. The computational complexity results summarized in this table are as previously discussed in the text. As for memory requirements: the Hebb approximation requires storage of the current example and the current mean weight vector, both of $\mathcal{O}(N)$, while the full online algorithm requires the additional storage of the covariance matrix, of $\mathcal{O}(N^2)$. The offline algorithm requires storage of all training examples, of $\mathcal{O}(Nt)$, in addition to lower order contributions from the mean weight vector and auxiliary variables.

Table 1. Comparison of asymptotic performance, computational complexity, and storage requirements for the three algorithms discussed in this section.

| | Online | | Offline |
|---|---|---|---|
| | 2nd Order | Hebb Approx. | |
| $\epsilon(\alpha \to \infty)$ | $0.88/\alpha$ | | $0.44/\alpha$ |
| Comp. complexity | $\mathcal{O}(N^2 t)$ | $\mathcal{O}(Nt)$ | $\mathcal{O}(\hat{N}Nt)$ |
| Memory | $\mathcal{O}(N^2)$ | $\mathcal{O}(N)$ | $\mathcal{O}(Nt)$ |

## 5 Binary Approximate Posterior

For a binary prior of the form (2.4), a reasonable choice for the approximate posterior is that of a biased binary distribution (Winther & Solla 1998),

$$p(\mathbf{w}|\mathbf{A}_t) = \prod_i \left[ \frac{1 + \langle w_i \rangle}{2} \, \delta(w_i - 1) + \frac{1 - \langle w_i \rangle}{2} \, \delta(w_i + 1) \right] . \tag{5.1}$$

The parameters $\mathbf{A}$ of the approximate distribution are thus the components of the mean weight vector $\langle \mathbf{w} \rangle$. The Kullback-Leibler distance is minimized by setting the new values of the parameters to be equal to their mean computed with the updated posterior $p(\mathbf{w}|\mathbf{A}_t, (\tau, \mathbf{s}))$. The update rule is thus given by

$$\widetilde{\langle \mathbf{w} \rangle} = \sum_{\{w_i = \pm 1\}} \mathbf{w} \; p(\mathbf{w}|\mathbf{A}_t, (\tau, \mathbf{s})) \; . \tag{5.2}$$

No closed form for the update rule (5.2) is available for general $N$, but progress can be made in the thermodynamic limit by invoking the cavity method. We now outline the calculation.

An auxiliary field $h$ is introduced through the identity

$$1 = \int dh\ \delta(h - \frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s}) = \int \frac{dhdx}{2\pi i}\ e^{x\left(\frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s} - h\right)}\ , \tag{5.3}$$

where the integral over $x$ is along the imaginary axis. The update Eq. (5.2) can thus be rewritten as

$$\widetilde{\langle \mathbf{w} \rangle} = \frac{\sum_{\mathbf{w}=\pm 1} \int \frac{dhdx}{2\pi i}\ \mathbf{w}\ e^{x\left(\frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s} - h\right)} p(\tau|h)\, p(\mathbf{w}|\mathbf{A}_t)}{\sum_{\mathbf{w}=\pm 1} \int \frac{dhdx}{2\pi i}\ e^{x\left(\frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s} - h\right)} p(\tau|h)\, p(\mathbf{w}|\mathbf{A}_t)}\ . \tag{5.4}$$

In order to perform the trace over the weights, it is convenient to introduce a posterior average from which the $i$th weight component is excluded,

$$\langle \ldots \rangle_i = \frac{\sum_{\mathbf{w}=\pm 1} \int \frac{dhdx}{2\pi i} \ldots e^{x\left(\frac{1}{\sqrt{N}}\sum_{j\neq i} w_j s_j - h\right)} p(\tau|h)\ \prod_{j\neq i} p(w_j|\langle w_j \rangle)}{\sum_{\mathbf{w}=\pm 1} \int \frac{dhdx}{2\pi i} e^{x\left(\frac{1}{\sqrt{N}}\sum_{j\neq i} w_j s_j - h\right)} p(\tau|h)\ \prod_{j\neq i} p(w_j|\langle w_j \rangle)}\ ,$$

with

$$p(w_i|\langle w_i \rangle) = \frac{1+\langle w_i \rangle}{2}\ \delta(w_i - 1) + \frac{1-\langle w_i \rangle}{2}\ \delta(w_i + 1)\ .$$

The update rule for the $i$th weight component is then given by

$$\widetilde{\langle w_i \rangle} = \frac{\langle \frac{1+\langle w_i \rangle}{2} e^{x s_i/\sqrt{N}} - \frac{1-\langle w_i \rangle}{2} e^{x s_i/\sqrt{N}} \rangle_i}{\langle \frac{1+\langle w_i \rangle}{2} e^{x s_i/\sqrt{N}} + \frac{1-\langle w_i \rangle}{2} e^{x s_i/\sqrt{N}} \rangle_i}\ , \tag{5.5}$$

which results in

$$\widetilde{\langle w_i \rangle} = \frac{\langle \sinh(\tanh^{-1}\langle w_i \rangle + \frac{1}{\sqrt{N}} s_i\, x)\rangle_i}{\langle \cosh(\tanh^{-1}\langle w_i \rangle + \frac{1}{\sqrt{N}} s_i\, x)\rangle_i}\ . \tag{5.6}$$

The expression (5.6) is exact, but a cavity argument needs to be introduced in order to perform the required averages. We assume that in the thermodynamic limit the variable $y_i = \tanh^{-1}\langle w_i \rangle + \frac{1}{\sqrt{N}} s_i\, x$ is Gaussian, with mean $\langle y_i \rangle = \tanh^{-1}\langle w_i \rangle + \frac{1}{\sqrt{N}} s_i\, \langle x \rangle_i$ and variance $\sigma_i^2 = \frac{1}{N} s_i^2(\langle x^2 \rangle_i - \langle x \rangle_i^2)$. The variance becomes self-averaging in the thermodynamic limit, and can be written as

$$\begin{aligned} \frac{1}{N} s_i^2(\langle x^2 \rangle_i - \langle x \rangle_i^2) &\approx \frac{1}{N}(\langle x^2 \rangle_i - \langle x \rangle_i^2) \\ &\approx \frac{1}{N}(\langle x^2 \rangle - \langle x \rangle^2) \end{aligned}$$

for uncorrelated inputs, $\overline{s_i s_j} = \delta_{ij}$. The averages in Eq. (5.6) can now be performed, to obtain

$$\widetilde{\langle w_i \rangle} = \tanh\left(\tanh^{-1}\langle w_i \rangle + \frac{1}{\sqrt{N}} s_i \langle x \rangle_i\right)\ . \tag{5.7}$$

In order to complete the calculation we need to evaluate the mean value of the auxiliary variable $x$; note that the required average is not $\langle x \rangle$ but $\langle x \rangle_i$, to indicate that the average is to be performed over a posterior that excludes the $i$th weight component. The calculation of $\langle x \rangle_i$ involves two steps: a calculation of $\langle x \rangle$ followed by a cavity argument that relates $\langle x \rangle$ to $\langle x \rangle_i$.

The average $\langle x \rangle$ is given by a logarithmic derivative,

$$\langle x \rangle = \left. \frac{\partial \ln Z(u)}{\partial u} \right|_{u=0} \quad ,$$

where we have introduced a partition function

$$Z(u) = \sum_{\mathbf{w}=\pm 1} \int \frac{dhdx}{2\pi i}\; e^{x\left(u + \frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s} - h\right)} p(\tau|h)\; p(\mathbf{w}|\mathbf{A}_t) \; . \tag{5.8}$$

Note that the external field $u$ couples to $x$. Integrals over both $x$ and $h$ can be performed to obtain

$$Z(u) = \sum_{\mathbf{w}=\pm 1} p(\tau|h+u)\; p(\mathbf{w}|\mathbf{A}_t) \; , \tag{5.9}$$

where $h$ now stands for $\frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{s}$ and is no longer a variable. The quantity $Z(u)$ can be identified as an online approximation to the predictive probability $\langle p(\tau|h+u)\rangle$, with $\langle p(\tau|h)\rangle$ given by Eq. (4.6). This identification leads to

$$\langle x \rangle = \left. \frac{\partial}{\partial u} \ln\langle p(\tau|h+u)\rangle \right|_{u=0} = \frac{\partial}{\partial \langle h \rangle} \ln\langle p(\tau|h)\rangle \; . \tag{5.10}$$

The second moment of $x$ follows from the linear response theorem:

$$\langle x^2 \rangle - \langle x \rangle^2 = \frac{\partial \langle x \rangle}{\partial \langle h \rangle} \; .$$

The last step is to relate $\langle x \rangle$ and $\langle x \rangle_i$. We invoke a Gaussian assumption for the distribution of $\frac{1}{\sqrt{N}} x s_i$ and use the exact relation

$$\langle \ldots \rangle = \frac{\langle \sum_{w_i=\pm 1} \ldots p(w_i|\langle w_i \rangle)\, e^{\frac{1}{\sqrt{N}} x s_i} \rangle_i}{\langle \sum_{w_i=\pm 1} p(w_i|\langle w_i \rangle)\, e^{\frac{1}{\sqrt{N}} x s_i} \rangle_i}$$

to obtain

$$\langle x \rangle_i = \langle x \rangle - \frac{1}{\sqrt{N}} \widetilde{\langle w_i \rangle}\; s_i \, \frac{\partial \langle x \rangle}{\partial \langle h \rangle} \; . \tag{5.11}$$

The Bayesian learning algorithm for a simple perceptron with binary weights obtained above is expected to be exact in the thermodynamic limit. One further approximation is needed for the algorithm to be truly online, since $\langle x \rangle_i$

as given in Eq. (5.11) to be substituted onto Eq. (5.7) for $\widetilde{\langle w_i \rangle}$ does itself depend on $\widetilde{\langle w_i \rangle}$. The approximation is to replace $\widetilde{\langle w_i \rangle}$ by $\langle w_i \rangle$ in the right-hand side of Eq. (5.11). Since the difference between $\langle w_i \rangle$ and $\widetilde{\langle w_i \rangle}$ is $\mathcal{O}(1/\sqrt{N})$, the error arising from this approximation is $\mathcal{O}(1/N)$, and thus negligible in the thermodynamic limit.

We end this section with a cautionary note against the impulse to construct an online algorithm from a direct expansion of Eq. (5.7), based on the argument that the contribution from the new example is small, $\mathcal{O}(\frac{1}{\sqrt{N}})$. To show that this procedure leads to an incorrect result, we use the recursive relation of Eq. (5.7) to obtain:

$$\langle w_i \rangle^t = \tanh \left( \frac{1}{\sqrt{N}} \sum_{\mu=1}^{t} s_i^\mu \langle x^\mu \rangle_i^{\mu-1} \right) , \tag{5.12}$$

where

$$\langle x^\mu \rangle_i^{\mu-1} = \langle x^\mu \rangle^{\mu-1} - \frac{1}{\sqrt{N}} \langle w_i \rangle^\mu \, s_i^\mu \, \frac{\partial \langle x^\mu \rangle^{\mu-1}}{\partial \langle h^\mu \rangle^{\mu-1}} .$$

The supraindices indicate the time step and therefore label the corresponding example. The value of $\langle x^\mu \rangle^{\mu-1}$ follows from Eq. (5.10), while that of $\langle p(\tau^\mu | h^\mu) \rangle^{\mu-1}$ is given by Eq. (4.6).

In contrast to the exact result of Eq. (5.12), a second order expansion of Eq. (5.7) leads to a Hebbian approximation to the online algorithm that can be written as

$$\langle w \rangle_i^t = \frac{1}{\sqrt{N}} \sum_{\mu=1}^{t} s_i^\mu \lambda^\mu \langle x^\mu \rangle^{\mu-1} , \tag{5.13}$$

where $\lambda^\mu = \frac{1}{N}(\langle \mathbf{w} \cdot \mathbf{w} \rangle^\mu - \langle \mathbf{w} \rangle^\mu \cdot \langle \mathbf{w} \rangle^\mu)$. Discrepancies between Eqs. (5.12) and (5.13) are due to the expansion of Eq. (5.7).

## Performance comparison

In order to compare the performance of the online Bayesian algorithm for binary weights to that of several other online and offline approaches, it is useful to cast the algorithm in the form of mean field equations for the order parameters that control its average performance..

The task is that of learning a noisy simple perceptron with binary weights; the teacher's output is given by $\tau = \eta \, \mathrm{sign}(\mathbf{v} \cdot \mathbf{s})$, where $\mathbf{v}$ is a binary weight vector and $\eta = -1$ with probability $\kappa$, and $\eta = 1$ otherwise. The relevant order parameters are $R = \frac{1}{N} \mathbf{v} \cdot \langle \mathbf{w} \rangle$, $q = \frac{1}{N} \langle \mathbf{w} \rangle \cdot \langle \mathbf{w} \rangle$ and $T = \frac{1}{N} \mathbf{v} \cdot \mathbf{v}$. For the binary teacher considered here, $T = 1$. The other two parameters, $R$ and $q$, need to be determined in order to compute the generalization error of Eq. (4.13).

In a Bayesian algorithm of the type constructed here the student vector $\mathbf{w}$ and teacher vector $\mathbf{v}$ are sampled from the same space, so that $R = q$. The time evolution of $R$ is derived from the scalar product between $\mathbf{v}$ and the average student weight vector of Eq. (5.7). The overlap between $\mathbf{v}$ and the argument in the right hand side of the equation consists of two contributions: a bias towards the teacher weight vector and a fluctuating term due to the randomness of the inputs. The random contribution becomes Gaussian in the thermodynamic limit. The normalized time variable $\alpha = t/N$ becomes continuous in this limit. The time evolution of the order parameters is found to be given by (Winther & Solla 1998)

$$\begin{aligned} R = q &= \int Dz \tanh(\sqrt{A}z + A) \\ \frac{dA}{d\alpha} &= \frac{1}{\pi}\frac{1}{\sqrt{1-q}} \int Dt \frac{(1-2\kappa)^2 e^{-\frac{1}{2}qt^2}}{\kappa + (1-2\kappa)\Phi(\sqrt{q}t)} \ , \end{aligned} \tag{5.14}$$

where $Dt = dte^{-t^2/2}/\sqrt{2\pi}$.

Comparison with other algorithms is based on the calculation of the values of $R$ and $q$ for each case, in order to compute the generalization error (4.13). We consider several possibilities:

1. **Clipping – taking the sign of the binary weights algorithm**. The value of $R$ for this algorithm follows from the scalar product between an arbitrary binary teacher vector $\mathbf{v}$ and a binary vector obtained by taking the sign of $\langle \mathbf{w} \rangle$: $R_{\text{clip}} = \frac{1}{N}\sum_i v_i \text{sign}(\langle w_i \rangle) = \frac{1}{N}\sum_i \text{sign}(v_i\langle w_i \rangle)$. Since $v_i\langle w_i \rangle$ is Gaussian with mean $q$ and variance $q(1-q)$ (Schietse, Bouten & Van den Broeck 1995),

   $$R_{\text{clip}} = \int Dz \ \text{sign}(\sqrt{A}z + A) = 2\Phi(\sqrt{A}) - 1 \ ,$$

   while $q_{\text{clip}} = \frac{1}{N}\sum_i \text{sign}^2\langle w_i \rangle = 1$

2. **Hebb approximation to Bayesian online algorithm**. A binary weight vector may be considered a as specific sampling of the continuous weight prior. It is therefore possible to use a continuous weight algorithm for the binary problem; the average performance will be identical to that of the continuous case (Winther & Solla 1998).

3. **Clipping – taking the sign of the continuous weights algorithm**. The value of $R$ for this algorithm is again of the form $R_{\text{clip cont.}} = \frac{1}{N}\sum_i v_i \text{sign}(\langle w_i \rangle) = \frac{1}{N}\sum_i \text{sign}(v_i\langle w_i \rangle)$, but the average weight vector $\langle \mathbf{w} \rangle$ is now given by the continuous weights Bayesian algorithm. The Gaussian nature of $v_i\langle w_i \rangle$ is again invoked to obtain

   $$R_{\text{clip cont.}} = \int Dz \ \text{sign}(\sqrt{q(1-q)}z + q) = 2\Phi(\frac{\sqrt{q}}{\sqrt{1-q}}) - 1 \ ,$$

   with $q_{\text{clip cont.}} = 1$.

4. **Optimal binarization of continuous weights**. Functional optimization techniques have been used to obtain a transformation $f(\langle w_i \rangle)$ to be applied to average weight components $\langle w_i \rangle$ found by a continuous weights algorithm so as to maximize the overlap between the transformed weight vector and an arbitrary binary teacher vector $\mathbf{v}$ (Schietse, Bouten & Van den Broeck 1995). Their arguments have been applied to the Bayesian scenario, to obtain $f(\langle w_i \rangle) = \tanh\left(\frac{1}{1-q}\langle w_i \rangle\right)$. Since $v_i \langle w_i \rangle$ is Gaussian with mean $q$ and variance $q(1-q)$, the overlap $R$ can be computed to obtain

$$R_{\text{trans.}} = \int Dz \tanh\left(\frac{\sqrt{q}}{\sqrt{1-q}} z + \frac{q}{1-q}\right) ,$$

with $q_{\text{trans.}} = R_{\text{trans.}}$ as for the Bayesian scenario.

The performance of these various approaches in shown in figure 3 for $\kappa = 0$. Theoretical results based on Eq. (4.13) are indistinguishable at the scale of this figure from numerical results for $N = 1000$ for all online algorithms. The small $\alpha$ behavior shows two groups: the two algorithms based on clipping average weights exhibit poorer performance, while the other three online algorithms do quite well; the behavior of the latter group in this regime is likely to be described by pure Hebbian learning of the form $\mathbf{w} \propto \sum_\mu \tau^\mu \mathbf{s}^\mu$. As we follow this group into the intermediate $\alpha$ regime, notice that the performance of the continuous weights algorithm deteriorates rapidly when compared to that of the binary weights algorithm, which is optimal among the online algorithms considered here. The algorithm based on an optimal binarization of the result of the continuous weights algorithm performs quite well up to $\alpha = 2$, but its performance deteriorates as $\alpha$ increases, and it approaches that of the algorithm based on simple clipping of continuous weights. The clipped version of the binary weights algorithm, which performs poorly in the small $\alpha$ regime, approaches the performance of the binary weights algorithm with increasing $\alpha$, a result that indicates that in the large $\alpha$ regime the components of the average weight vector tend to $\pm 1$ for the binary weights algorithm.

Results for two offline algorithms are also shown in figure 3. Both exhibit a discontinuous transition to zero generalization error at $\alpha = 1.245$. The upper curve shows the result for the Gibbs algorithm, based on a random sampling of the space of solutions that are consistent with the training set (i.e. have zero learning error). The lower curve shows the result of the offline Bayesian algorithm.

# 6 Conclusion and Outlook

In this chapter we have reviewed the Bayesian approach to online learning originally proposed by (Opper 1996) and generalized by (Winther & Solla

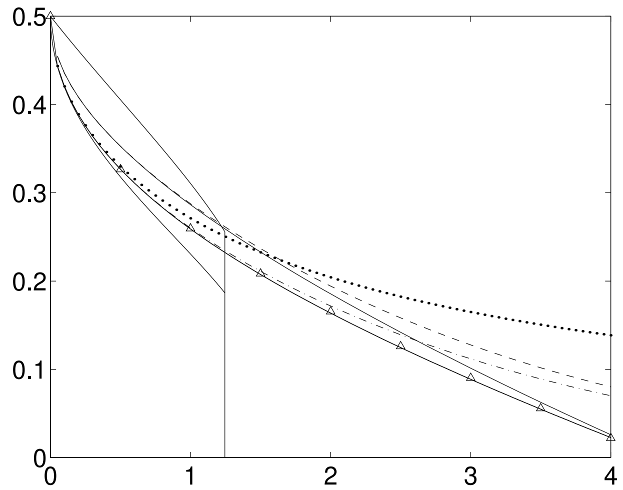


Figure 3. Learning curves ($\epsilon$ versus $\alpha$) for the binary simple preceptron at $\kappa = 0$. Full lines showing a discontinuous transition to perfect generalization correspond to offline algorithms (upper curve: Gibbs; lower curve: Bayes). Full line with triangles shows the result of the Bayesian online binary weights algorithm. Simulation results (shown as triangles) were obtained for $N = 1000$ and averaged over 100 runs. The upper full line shows the result of taking the sign of the solution to the binary weights algorithm. The dotted line is for the Bayesian online continuous weights algorithm. The dashed line shows the result of taking the sign of the solution to the continuous weights algorithm. The dashed-dotted line is for the optimal binarization of the continuous weights solution.

1998). We have applied this approach to the analysis of two learning scenarios for the noisy simple perceptron; the techniques are those of statistical physics and the results are expected to be exact in the thermodynamic limit of large system size.

In the two scenarios studied here the weights of the perceptron rule to be learned are sampled from either a spherical Gaussian distribution or a binary distribution with independent components.

The analysis of the Bayesian online algorithm for the simple perceptron with continuous weights shows that the online approach has the same computational complexity as the offline Bayesian algorithm, but inferior performance. A thermodynamic limit calculation of the asymptotic (large $\alpha$) behavior of the generalization error for the case of uncorrelated inputs shows that the generalization error of the online algorithm is twice that of the offline algorithm (Kinouchi & Caticha 1992). This discrepancy in asymptotic performance is a consequence of the non-smoothness of the simple preceptron rule, and is to be contrasted to the asymptotic equivalence between online and offline learning for smooth rules (see also Opper, this volume).

One of the usual arguments for using online learning is that it is faster than offline learning. However, our analysis of this simple learning scenario shows that the online approach is not faster unless additional approximations are introduced, such as restricting the weight update to its projection along the direction of the current example. This form of Hebbian learning exhibits the same asymptotic performance than the full online algorithm for the case of uncorrelated inputs. This asymptotic equivalence is not expected to hold for more general input distributions, where the full algorithm, which makes use of a covariance matrix that does not become asymptotically diagonal for a general input distribution, is expected to outperform an approximation whose use of second order information is restricted to the diagonal elements of the covariance matrix.

The Bayesian online approach has also been applied to a simple perceptron with binary weights. The performance of this algorithm has been compared to that of a several other online approaches: a continuous weights Bayesian algorithm, a binarization of the solution for the continuous weights algorithm by either clipping or through an optimal transformation, and a binarization by clipping of the solution for the binary weights algorithm. As expected, the Bayesian algorithm for binary weights does outperform all other online approaches.

The gradual decay of the generalization error to zero as $\alpha \to \infty$ for these online algorithms is to be contrasted with the discontinuous transition to zero generalization error at $\alpha = 1.245$ exhibited by two offline algorihtms, Bayes and Gibbs, when applied to the same learning scenario. Although the Bayesian online approach outperforms all other online approaches investigated here, it is itself outperformed for all $\alpha$ by the offline Bayesian algorithm for learning a simple perceptron with binary weights.

The current outstanding challenge is that of extending this approach to learning scenarios involving multilayer networks. Approximations will be required to render the problem analytically tractable and computationally feasible. A promising direction under investigation is that of considering a two layer network in the limit of large number of hidden units, a regime in which the activity of a linear otput unit can be assumed to converge to a Gaussian variable. Another possible extension to models of more practical interest is to consider an online approach to learning with Gaussian processes.

## Acknowledgements

The authors wish to thank Bernhard Schottky for valuable discussions. This research has been supported by the Danish Research Councils for the Natural and Technical Sciences through the Danish Computational Neural Network Center (CONNECT).

# References

Kinouchi O. and Caticha N. (1992), Optimal Generalization in Perceptrons, J. Phys. A: Math. Gen. **25** 6243.

Opper M. (1996), Online versus Offline Learning from Random Examples: General Results, Phys. Rev. Lett. **77** 4671.

Opper M. and Winther O. (1996), A Mean Field Approach to Bayes Learning in Feed-Forward Neural Networks, Phys. Rev. Lett. **76** 1964.

Schietse J., Bouten M. and Van den Broeck C. (1995), Training Binary Perceptron by Clipping, Europhys. Lett. **32** 279.

Saad D. and Rattray M. (1997), Globally Optimal Parameters for On-Line Learning in Multilayer Neural Networks, Phys. Rev. Lett. **79** 2578.

Saad D. and Solla S. A. (1995), Exact Solution for On-Line Learning in Multilayer Neural Networks, Phys. Rev. Lett. **74** 4337.

Vicente R. and Caticha N. (1997), Functional Optimization of Online Algorithms in Multilayer Neural Networks, J. Phys. Math. Gen. **30** L599.

Winther O. and Solla S. A. (1998), Optimal Bayesian Online Learning, in *Theoretical Aspects of Neural Computation (TANC-97)*, K. Y. M. Wong, I. King and D.-Y. Yeung eds., Springer Verlag, Singapore.