Dynamics of batch training in a perceptron

J. Phys. A: Math. Gen. **31** (1998) 4835–4850. Printed in the UK PII: S0305-4470(98)88016-5

Dynamics of batch training in a perceptron

Siegfried Bös[†][§] and Manfred Opper[‡][∥]

[†] Laboratory for Information Synthesis, Brain Science Institute, RIKEN, Wako-shi, Saitama 351-01, Japan

[‡] Theoretical Physics III, University of Würzburg, Am Hubland, 97074 Würzburg, Germany

Received 30 September 1997, in final form 23 December 1997

Abstract. Early stopping and weight decay are studied in a linear perceptron using a new simplified approach to the dynamics in the thermodynamical limit. The approach is directly deduced from the gradient descent weight update. It allows an exact description of the dynamics of the batch training process. The results are compared with a recent study of early stopping and weight decay based on the equilibrium statistical mechanics approach. It is shown that the equilibrium results for early stopping are good approximations and are exact for weight decay. Furthermore, in the dynamical approach it is possible to determine the necessary number of training steps to fulfil certain termination conditions. It can be shown that asymptotically, i.e. if the number of examples is large, only two batch steps are required to reach the optimal convergence, if the learning rate is optimally chosen.

1. Introduction

The ability of neural networks to learn tasks from examples makes them interesting for many applications for which direct algorithmic approaches are unknown. In *supervised* training, a set of examples, i.e. pairs of inputs and correct outputs, is used to minimize the empirical loss. The empirical loss is a certain cost function averaged over the set of examples. The performance for the whole learning task is measured by the expected loss, an average over all possible inputs. If the number of examples is large it can be expected that the minimization of the empirical loss also leads to a minimal expected loss.

However, for rather small numbers of examples empirical loss and expected loss can be quite different quantities. Then the training process which minimizes the empirical loss, will not simultaneously lead to a minimum of the expected loss. This phenomenon is called *overfitting* or *overtraining*. It is useful to distinguish clearly between overfitting and overtraining. Overfitting describes the case where a network has too many degrees of freedom and therefore fits the examples in a more complicated fashion than necessary. Overtraining on the other hand, even appears in optimally chosen network architectures if the task is nonlinear [1]. As long as the number of examples is small an overtrained network might learn the examples best, although it shows only poor generalization abilities.

To avoid overfitting, model selection strategies should be applied in order to choose the most adequate network architecture. Several approaches for model selection can be found in literature, such as the Akaike criterion [2], optimal brain damage [3], a Bayesian approach [4], the network information criterion (NIC) [5] and others.

[§] E-mail address: boes@islabbrain.riken.go.jp

[∥] E-mail address: opper@physik.uni-wuerzburg.de

The most commonly used methods to avoid overtraining are *early stopping* and *weight decay*. Early stopping simply terminates the training process at an earlier state. If the values of the two loss functions observed during the training process, overtraining implies that after an initial decrease of both loss functions, the expected loss begins to increase again while the empirical loss continues to decrease. Therefore, it would be advantageous to terminate training before the expected loss starts to rise again. However, in practice, the value of the expected loss is unknown, and one has to deal with approximations for it, which can be summarized as *validation methods* [6–8]. Here we do not want to deal with the additional errors which originate from validation methods, but rather concentrate on the abilities of early stopping alone. This is the reason why we examine what is optimally possible with early stopping, if it is guided by the actual expected loss. Some results on the effect of validation with a finite validation set, which are related to our model, can be found in [7, 9].

Weight decay introduces an additional penalty term which reduces the size of the weights in each training step. The problem of weight decay is to determine the relative strength of this penalty term [10]. Sometimes weight decay can be strong enough that a certain weight vector is effectively eliminated, thus weight decay also changes the network architecture (called *pruning*), and avoids overfitting.

Here we will introduce a model which allows a full analytical description of the training process. In this setting the effect of early stopping and also weight decay can be studied in detail.

2. The model

To make the method tractable we have to restrict ourselves to quite a simple neural network model. Nevertheless, previous works have demonstrated that this set-up displays some typical behaviour of neural nets [11, 12]. The model is a single-layer perceptron [13], which has one layer of adjustable weights W_i between the N-dimensional input layer x_i and the one-dimensional output z. The learning task can be formulated also for nonlinear outputs, however, an analytical treatment is only possible if the outputs z of the trained network are *linear*. Together, we have

$$h(\boldsymbol{x}) := \frac{1}{\sqrt{N}} \sum_{i=1}^{N} W_i x_i \qquad \text{and} \qquad z = g(h) = h. \tag{1}$$

We are interested in supervised training, where examples $x_{i\mu}$ ($\mu = 1, \ldots, P$) are given for which the correct output z^*_μ is known. To define the task more clearly and to monitor the training process, we assume that the examples are given by another network, the *teacher* network. Variables referring to the teacher are always indicated by a star '*'. The teacher is not restricted to linear outputs, it can have a different output function $z^* = g^*(h^*)$. Usual choices for $g^*(h^*)$ are $\tanh(h^*)$ or $h^* + \epsilon$, where a Gaussian noise ϵ with zero mean and variance σ is added.

Training is based on empirical loss minimization. The most common cost function is the mean squared error (MSE), i.e. $\text{loss}[\boldsymbol{x}, z^*, z] := \frac{1}{2}[z^*(\boldsymbol{x}) - z(\boldsymbol{x})]^2$. Note that only $\boldsymbol{x}$ and z^* are independent arguments of loss. By introducing the teacher, z^* is given by $\boldsymbol{x}$ and $\boldsymbol{W}^*$. The student weight vector $\boldsymbol{W}$ is determined by the training process and therefore dependent on $\boldsymbol{x}$ and $\boldsymbol{W}^*$, as

$$\text{loss}[\boldsymbol{x}, \boldsymbol{W}^*; \boldsymbol{W}] = \frac{1}{2}\left[g^*\left(\frac{1}{\sqrt{N}} \sum_{i=1}^{N} W_i^* x_i\right) - \frac{1}{\sqrt{N}} \sum_{i=1}^{N} W_i x_i\right]^2. \tag{2}$$

During training, the average error over the examples, i.e. the *training error* or empirical loss $E_T := \langle \mathrm{loss}[\boldsymbol{x}_\mu, \boldsymbol{W}^*, \boldsymbol{W}] \rangle_{\{\boldsymbol{x}_\mu, \mu=1,\ldots,P\}}$, is minimized. However, what we are interested in, is a minimal error averaged over all possible inputs $\boldsymbol{x}$, i.e. $E_G := \langle \mathrm{loss}[\boldsymbol{x}, \boldsymbol{W}^*, \boldsymbol{W}] \rangle_{\{\boldsymbol{x} \in \mathrm{Input}\}}$, called *generalization error* or expected loss.

The effect of the training can be described by the order parameters, i.e.

$$R := \frac{1}{N} \sum_{i=1}^{N} W_i^* W_i \qquad \text{and} \qquad Q := \frac{1}{N} \sum_{i=1}^{N} (W_i)^2 . \tag{3}$$

It can be shown [11] that for a fixed student and teacher and random inputs $\boldsymbol{x}$, i.e. all components x_i are independent with zero means and unit variances, the local fields h^* and h fulfil,

$$\langle (h^*)^2 \rangle_x = 1 \qquad \langle h^* h \rangle_x = \hat{R} \qquad \langle (h)^2 \rangle_x = \hat{Q} \tag{4}$$

where $\hat{R}$ and $\hat{Q}$ are averaged values of the order parameters. They can be used to describe the typical behaviour of a large neural network.

The average over the inputs $\boldsymbol{x}$ can then be transformed into an average over the local fields h^* and h. It is more convenient to use the uncorrelated local fields $\tilde{h}^*$ and $\tilde{h}$, such that $\mathrm{loss}(\tilde{h}^*, \tilde{h})$ becomes $\frac{1}{2}[g^*(\tilde{h}^*) - (\hat{R}\tilde{h}^* + \sqrt{\hat{Q} - \hat{R}^2}\tilde{h})]^2$. Then the generalization error is defined as

$$E_G = \langle \mathrm{loss}(\tilde{h}^*, \tilde{h}) \rangle_{\tilde{h}^*, \tilde{h}} \qquad \langle \cdots \rangle_{\tilde{h}} = \int_{-\infty}^{\infty} \frac{\mathrm{d}\tilde{h}}{\sqrt{2\pi}} \exp\left(-\frac{\tilde{h}^2}{2}\right) \cdots . \tag{5}$$

In the case of the linear student, it simplifies to

$$E_G(\hat{R}, \hat{Q}) = \tfrac{1}{2}[G - 2H\hat{R} + \hat{Q}] \tag{6}$$

with the two parameters,

$$G(\gamma) := \langle [g^*(\gamma \tilde{h}^*)]^2 \rangle_{\tilde{h}^*} \qquad \text{and} \qquad H(\gamma) := \langle g^*(\gamma \tilde{h}^*) \tilde{h}^* \rangle_{\tilde{h}^*} \tag{7}$$

which summarize the dependence on the teacher. Here, we normalize the teacher weights, $\|\boldsymbol{W}^*\| = 1$. The effect of a different norm is taken into account by a variable gain $\gamma \neq 1$ of the teacher transfer function. In the case, $g^*(h^*) = \gamma h^* + \epsilon$, the parameters are $G = \gamma^2 + \sigma^2$ and $H = \gamma$.

The task can be learnt exactly, which means that the final generalization error is zero, only if the teacher and the student are identical. Such tasks are called *realizable* as opposed to *unrealizable* tasks, which cannot be learnt exactly due to an inappropriately chosen student network. The solution of the realizable task, linear student learns linear teacher, is quite obvious and exactly solved after $P = N$ examples are given. The results, which will be presented in the following, can be applied to this case if we set $G = H^2 = \gamma^2$. The unrealizable task is much more interesting, since an infinite number of examples are necessary to reach the minimal generalization error.

To plot lines in the figures we have to specify the values of G and H. For compatibility reasons to other works [11, 14], we choose $G = 0.84$ and $H = 0.78$, which corresponds to $g^*(h^*) = \tanh(5h^*)$. This is no restriction of any kind.

In the next section we introduce a new method to calculate the behaviour of the order parameters during the training process. Readers, who do not wish to go into technical details in first reading, can turn directly to the discussion of the results in section 5. Later they can read how the relevant expressions are calculated. In section 4 we recapitulate briefly the equilibrium statistical mechanics approach, the detailed discussion can be found in [14]. In section 5 we illustrate the results and compare the two approaches. Finally, the paper is concluded with a summary and a perspective on further problems. Parts of this work were presented in [15].

3. Dynamical approach

In the usual equilibrium statistical mechanics approach, the typical behaviour is calculated from a thermodynamic quantity, the free energy. The free energy is derived from the averaged sum of states, which minimize the training energy and fulfil a normalization condition, see (23). The average is computed over the distribution of the inputs $\{x_{i\mu}\}$, see [16] for details.

Another approach to study learning and generalization in a linear single-layer perceptron was proposed by Krogh and Hertz [17, 18, 10]. It solves a Langevin dynamics using Green functions. With this approach it should be possible to study the full dynamical behaviour of the network.

We think, however, that our approach is more transparent, because it is directly deduced from the gradient descent update rule. In the linear model the explicit calculation of the resulting microscopic weights is possible at every timestep, see step 1. This expression can be inserted in the definitions of the order parameters (3). The order parameters must then be averaged over the distribution of the inputs. Since the dependence of the inputs is complicated, we use a trick. We can additionally average over the distribution of the teachers $\boldsymbol{W}^*$, which does by symmetry not effect our result. The average over the teacher weights $\boldsymbol{W}^*$ is computed in step 2. The average over the inputs $\boldsymbol{x}_\mu$ can then be solved in step 3.

Preliminary versions of this method were already applied to static problems in [19, 20]. It should be mentioned that the full analytic treatment is restricted to the linear student for all three approaches. We will discuss this issue in the final section.

Step 1. The linear student allows the calculation of the explicit values of the weights at every timestep. We start with the gradient descent learning rule for the linear student $g'(h) = 1$,

$$W_i(t+1) = W_i(t) - \eta \frac{\partial (PE_T)}{\partial W_i} = W_i(t) + \frac{\eta}{\sqrt{N}} \sum_{\mu=1}^{P} [z^*_\mu - z_\mu(t)] x_{i\mu} \tag{8}$$

where t is a batch training step and η denotes the *learning rate*.

We need to discriminate between the underdetermined case, where we have less examples than parameters, i.e. $P < N$, and the overdetermined case. In the underdetermined case, the students weight vector can be expanded in the space of the examples, if the initial conditions are $W_i(0) = 0$, i.e.

$$W_i(t) =: \frac{1}{\sqrt{N}} \sum_{\mu=1}^{P} \sigma_\mu(t) x_{i\mu}. \tag{9}$$

Then a recursion for the coefficients $\sigma_\mu(t)$ can be found,

$$\sigma_\mu(t+1) = \sigma_\mu(t) - \eta \sum_{\nu=1}^{P} C_{\mu\nu}\sigma_\nu(t) + \eta z^*_\mu \tag{10}$$

with the *overlap matrix* $C_{\mu\nu} := \frac{1}{N}\sum_{i=1}^{N} x_{i\mu}x_{i\nu}$. From the geometrical series, we know the solution for this recursion and therefore for the weights,

$$W_i(t) = \frac{\eta}{\sqrt{N}} \sum_{\mu,\nu=1}^{P} z^*_\mu \left[\frac{\mathbb{1} - (\mathbb{1} - \eta \mathbf{C})^t}{\mathbb{1} - (\mathbb{1} - \eta \mathbf{C})} \right]_{\mu\nu} x_{i\nu} \tag{11}$$

where $\mathbb{1}$ denotes the identity matrix. Note that since the matrices commute, their order can be arbitrary. The initial conditions are $W_i(0) = 0$ and $W_i(1) = \eta N^{-\frac{1}{2}} \sum_{\mu=1}^{P} z_\mu^* x_{i\mu}$, which resembles the Hebbian learning rule. After infinitely many timesteps the *pseudo-inverse* weights are found, i.e.

$$W_i(t \to \infty) = \frac{1}{\sqrt{N}} \sum_{\mu,\nu=1}^{P} z_\mu^* (\mathbf{C}^{-1})_{\mu\nu} x_{i\nu} \tag{12}$$

which are valid as long as the examples are linearly independent, i.e. $P < N$. The other case, $P > N$, will be explained later.

Step 2. From (11), we can calculate the order parameters (3) at every timestep, i.e. $R(t)$ and $Q(t)$. Since the order parameters depend in a complicated way on the inputs, we apply the following trick to simplify the calculations. For a spherical input distribution, the average over the inputs will by symmetry not depend on the teacher vector $\boldsymbol{W}^*$. Hence we can average, without changing the final result, also over an arbitrary distribution of the teacher vector. For convenience we use again a spherical Gaussian distribution.

Computing the average over the teacher distribution first, we receive

$$\begin{aligned}\hat{R}(t) &= \left\langle \frac{1}{N} \sum_{\mu,\nu=1}^{P} \left[\frac{\mathbb{1} - (\mathbb{1} - \eta\mathbf{C})^t}{\mathbf{C}} \right]_{\mu\nu} \langle z_\mu^* h_\nu^* \rangle_{\{W_i^*\}} \right\rangle_{\{x_{i\mu}\}} \\ &= \left\langle \frac{\alpha H}{P} \sum_{\mu=1}^{P} [\mathbb{1} - (\mathbb{1} - \eta\mathbf{C})^t]_{\mu\mu} \right\rangle_{\{x_{i\mu}\}} \end{aligned} \tag{13}$$

where we have used $P = \alpha N$, and the average $\langle z_\mu^* h_\nu^* \rangle_{\{W_i^*\}} = C_{\mu\nu} H$ from appendix A (A3).

Similarly, we can determine the other order parameter,

$$\begin{aligned}\hat{Q}(t) &= \left\langle \frac{1}{N} \sum_{\mu,\nu,\tau,\sigma=1}^{P} \left[\frac{\mathbb{1} - (\mathbb{1} - \eta\mathbf{C})^t}{\mathbf{C}} \right]_{\mu\nu} \left[\frac{\mathbb{1} - (\mathbb{1} - \eta\mathbf{C})^t}{\mathbf{C}} \right]_{\tau\sigma} \left(\frac{1}{N} \sum_{i=1}^{N} x_i^\nu x_i^\sigma \right) \langle z_\mu^* z_\tau^* \rangle_{\{W_i^*\}} \right\rangle_{\{x_{i\mu}\}} \\ &= \left\langle - \frac{\alpha(G - H^2)}{P} \sum_{\mu=1}^{P} [\mathbf{C}^{-1}(\mathbb{1} - (\mathbb{1} - \eta\mathbf{C})^t)^2]_{\mu\mu} \right. \\ &\quad \left. + \frac{\alpha H^2}{P} \sum_{\mu=1}^{P} [(\mathbb{1} - (\mathbb{1} - \eta\mathbf{C})^t)^2]_{\mu\mu} \right\rangle_{\{x_{i\mu}\}}. \end{aligned} \tag{14}$$

Again an identity (A2) from appendix A was used and some matrix algebra applied.

It is possible to calculate the behaviour of the training error in the same fashion,

$$E_T(t) = \left\langle \left\langle \frac{1}{2P} \sum_{\mu=1}^{P} \left(z_\mu^* - \frac{1}{\sqrt{N}} \sum_{i=1}^{N} W_i(t) x_{i\mu} \right)^2 \right\rangle_{\{W_i^*\}} \right\rangle_{\{x_{i\mu}\}}. \tag{15}$$

For the overdetermined case, $P > N$, a recursion analogue to (10) can be found, i.e.

$$W_i(t+1) = W_i(t) - \eta \sum_{j=1}^{N} B_{ij} W_j(t) + \frac{\eta}{\sqrt{N}} \sum_{\mu=1}^{P} z_\mu^* x_{i\mu}. \tag{16}$$

with $B_{ij} := \frac{1}{N} \sum_{\mu=1}^{P} x_{i\mu} x_{j\mu}$. From there, the calculation is quite similar to the one above, with matrix $\mathbf{B}$ playing the role of matrix $\mathbf{C}$.

Step 3. The average over the input distribution $\{x_{i\mu}\}$ can be computed by transforming the traces, containing the random matrices $\mathbf{C}$ or $\mathbf{B}$, into integrals over the eigenvalues ξ of $\mathbf{C}$ or $\mathbf{B}$. We attain integrals of the following form,

$$\left\langle \frac{1}{P}\sum_{\mu=1}^{P}[(\mathbb{1}-\eta\mathbf{C})^l\mathbf{C}^m]_{\mu\mu}\right\rangle_{\{x_{i\mu}\}} = \int_{\xi_{\min}}^{\xi_{\max}} \mathrm{d}\xi\rho(\xi)(1-\eta\xi)^l\xi^m =: I_m^l \tag{17}$$

with $l \in \{0, t, 2t\}$ and $m \in \{-1, 0, 1\}$.

The integrals $I_m^l(\alpha, \eta, t)$ can be computed, once we know the density of the eigenvalues $\rho(\xi)$. The determination of this density can be found in recent literature calculated by Opper [21] and LeCun *et al* [22] using replicas, by Krogh [17] using a perturbation theory and by Sollich [23] with matrix identities. All these authors found,

$$\rho(\xi) = \frac{1}{2\pi\alpha\xi}\sqrt{(\xi_{\max}-\xi)(\xi-\xi_{\min})} \tag{18}$$

for $\alpha < 1$. The maximal and the minimal eigenvalues are $\xi_{\max,\min} := (1 \pm \sqrt{\alpha})^2$. The density of the eigenvalues $\rho(\xi)$ for matrix $\mathbf{B}$ is (18) multiplied by α. Thus, all that remains, is a numerical integration.

3.1. Results

The time-dependent integrals converge only, if the condition $|1-\eta\xi| < 1$ is fulfilled. From this condition a maximal learning rate $\eta_{\max}$ can be deduced. It is twice the inverse of the maximal eigenvalue of the correlation matrix, i.e.

$$\eta_{\max} = \frac{2}{\xi_{\max}} = \frac{2}{(1+\sqrt{\alpha})^2}. \tag{19}$$

As long as the learning rate is smaller than the maximal learning rate, the time-dependent integrals I_m^t and I_m^{2t} vanish for $t \to \infty$.

It is a well-known fact [24], that the maximal learning rate is twice the inverse of the maximal eigenvalue of the *Hessian* matrix $\mathbf{H}$. In the case of the linear perceptron, the Hessian $\mathbf{H}$ is identical to the matrix $\mathbf{B}$, since

$$H_{ij} := \frac{\partial^2(PE_T)}{\partial W_i \partial W_j} = \frac{1}{N}\sum_{\mu=1}^{P} x_{i\mu}x_{j\mu} = B_{ij} \tag{20}$$

and both findings are therefore consistent.

The time-independent integrals I_m^0 are also independent of the learning rate and depend only on α. The dependencies are shown in table 1. The I_m^0 determine the initial values of

Table 1. The values of the time-independent integrals I_m^0.

	$\alpha < 1$	$\alpha > 1$
I_{-1}^0	$\frac{1}{1-\alpha}$	$\frac{1}{\alpha-1}$
I_0^0	1	1
I_1^0	1	α

the corresponding time-dependent integrals, i.e. $I_m^t(t=0)$ and $I_m^{2t}(t=0)$. Therefore, we find the following results in the case of $\alpha < 1$,

$$\begin{aligned} E_G(\alpha,\eta,t) &= \frac{G}{2} + \frac{G-H^2}{2}\alpha\left(\frac{1}{1-\alpha} - 2I_{-1}^t + I_{-1}^{2t}\right) - \frac{H^2}{2}\alpha(1-I_0^{2t}) \\ E_T(\alpha,\eta,t) &= \frac{G-H^2}{2}I_0^{2t} + \frac{H^2}{2}I_1^{2t} \end{aligned} \tag{21}$$

and in the case of $\alpha > 1$,

$$\begin{aligned} E_G(\alpha,\eta,t) &= \frac{G-H^2}{2}\left(1 + \frac{1}{\alpha-1} - 2I_{-1}^t + I_{-1}^{2t}\right) + \frac{H^2}{2}I_0^{2t} \\ E_T(\alpha,\eta,t) &= \frac{G-H^2}{2}\left(1 - \frac{1}{\alpha} + \frac{I_0^{2t}}{\alpha}\right) + \frac{H^2}{2}\frac{I_1^{2t}}{\alpha}. \end{aligned} \tag{22}$$

These results will be discussed in section 5. The extension of the approach to weight decay is discussed in appendix B.

4. Equilibrium approach

Early stopping and weight decay can also be described by extending the usual equilibrium statistical mechanics approach [16]. The comparison with the results from the dynamical approach gives valuable support. The statistical mechanics approach describes batch training as an equilibrium problem. The free energy f of the system, is calculated by

$$-\beta f := \frac{1}{N}\langle \log Z\rangle_{\{\boldsymbol{x}_\mu:\mu=1,\ldots,P\}} \tag{23}$$

with β being the inverse temperature and Z the sum of states.

By maximizing it with respect to the order parameters, i.e. $\partial f/\partial\hat{R} = 0$ and $\partial f/\partial\hat{Q} = 0$, the values of $\hat{R}$ and $\hat{Q}$ can be determined. Also the training error E_T can be deduced from the free energy, as $\frac{1}{\alpha}\partial(\beta f)/\partial\beta = E_T$, where β is the inverse temperature $\beta := \frac{1}{T}$, for details see [16].

Already in [11], the equilibrium has been applied to linear perceptron learning an unrealizable task and the following expression for the errors have been found,

$$\begin{aligned} E_G(\alpha,a) &= \frac{1}{2(a^2-\alpha)}[a^2G - (2a-\alpha)\alpha H^2] \\ E_T(\alpha,a) &= E_G(\alpha,a)\left(\frac{a-1}{a}\right)^2 \end{aligned} \tag{24}$$

with the parameter a defined as $(a-1)^{-1} := \beta(Q_0 - Q)$, where Q_0 is the maximal possible value for Q.

Usually, it is assumed that training is continued, until the training error E_T reaches its absolute minimum. This means that the value of a needs to be determined at temperature $T = 0$. In [11], we found that the minimal value for the parameter a is $a_0(\alpha) := \max(1,\alpha)$. If we insert a_0 in (24), we receive the values shown in table 2.

In terms of the dynamics, the absolute minimum of the training error corresponds to an infinite number of training steps, i.e. $t \to \infty$, which we call *exhaustive training*. Therefore, we can reproduce the results in table 2, if we evaluate the asymptotic limit of equations (21) and (22). If the learning rate is smaller than its maximum $\eta_{\max}$, the time-dependent integrals disappear when the number of training steps diverges, i.e. $I_m^t(t\to\infty) = 0$ and $I_m^{2t}(t\to\infty) = 0$.

Table 2. The values of the errors $E_G(\alpha)$ and $E_T(\alpha)$ under exhaustive training. Exhaustive training corresponds in the dynamical approach to infinitely many timesteps, i.e. values of (21) and (22) with $t \to \infty$ and a suitable learning rate $\eta < \eta_{\max}$. In the statistical mechanics approach, exhaustive training corresponds to a minimal training error, i.e. it follows from (24) with $a = a_0$.

	$\alpha < 1$	$\alpha > 1$
E_G	$\frac{G}{2} + \frac{\alpha}{2(1-\alpha)}[G - (2-\alpha)H^2]$	$\frac{G-H^2}{2}\left(1 + \frac{1}{\alpha-1}\right)$
E_T	0	$\frac{G-H^2}{2}\left(1 - \frac{1}{\alpha}\right)$

In table 2 the values of the two errors under exhaustive training are shown. Especially interesting is the generalization error for $\alpha > 1$, which approaches asymptotically the residual error, i.e. $E_\infty := \frac{1}{2}(G - H^2)$, with a α^{-1}-behaviour. The behaviour of the errors between the two extreme values is discussed in section 5.

The exhaustive training solution in table 2 shows, however, strong overtraining effects. In the statistical mechanics approach [14], it can be shown that values for a larger than a_0, yield results with reduced overtraining or even none at all. A higher value of a corresponds to a finite temperature T, implying a non-minimal training error $E_T > E_T(a_0)$. To terminate training at a non-minimal training error is exactly what early stopping does. The statistical mechanics approach is in itself not a dynamical description, it can, however, be used to approximate early stopping. Also weight decay can be described in the equilibrium approach.

The results of the equilibrium approach are quite attractive and in accordance with the simulations. However, it is *a priori* not clear, whether the results are exact or only approximations. That missing information follows from a comparison with the dynamical results, which are *per construction* correct. In [14], the comparison of the two approaches is discussed in detail. Here, we only want to mention the results. The description of early stopping using the equilibrium approach is not exact, but a very good approximation. The results for weight decay of both approaches are equivalent, see figure 2.

5. Results

In this section the numerical solution of the expressions for the errors are discussed. From the dynamical approach, we have the solutions without weight decay (21), (22) and with weight decay (B7).

5.1. Training process

To demonstrate how well the dynamical approach describes the training process, we compare theory with simulation. In figure 1 the results are plotted. An overdetermined case ($\alpha = 1.05$) close to the storage capacity was chosen to illustrate overtraining and the non-zero training error.

With weight decay, the effect of overtraining can be lowered ($\lambda = 0.05$) and even totally avoided ($\lambda = 0.37$). Theory and simulation always coincide.

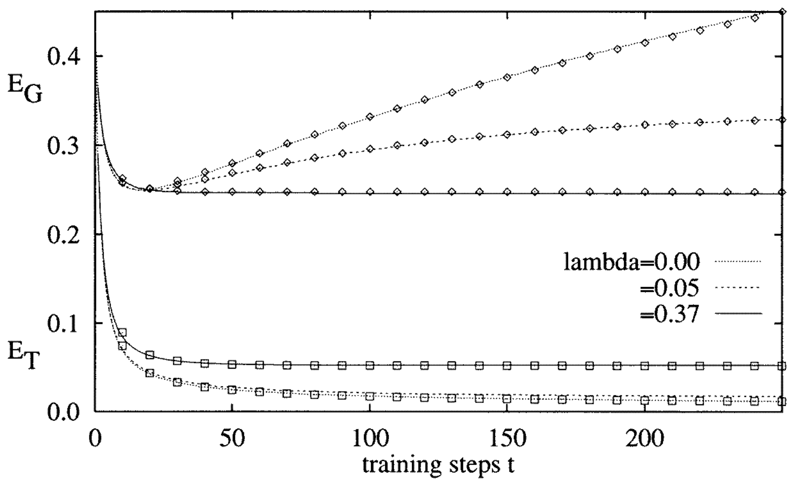

Figure 1. Training process: behaviour of the generalization error E_G (upper lines) and the training error E_T (lower lines) during training. For the loading rate $\alpha = P/N = 1.05$, i.e. near the storage capacity ($\alpha_c = 1$), different weight decay strengths $\lambda = 0.00$, 0.05 and 0.37 are tested. Theory and simulation are in very good agreement. Parameters: $G = 0.84$, $H = 0.78$; $\eta = 0.1$; and simulations with systems of size $N = 100$ averaged over 100 iterations.

5.2. Performance

The network's performance on the learning task, given by $E_G(\alpha = P/N)$, as a function of the number P of learnt examples, is shown in figure 2.

In *exhaustive training*, training is continued until the absolute minimum of the training error is reached, i.e. $t \to \infty$. Since the task is unrealizable, exhaustive training implies strong overtraining in the region around the storage capacity of the network, which is $\alpha_c = 1.0$ in the case of the continuous perceptron.

Either early stopping or weight decay can decrease overtraining. In figure 2, only the optimal results for both methods, early stopping and weight decay, and both approaches, dynamical and equilibrium, are shown. Three of the four possible optimal curves coincide. Early stopping and weight decay are equivalent in the equilibrium approach. Furthermore, the dynamical solution for weight decay, which is exact, coincides with the statistical mechanics curves. Only the exact solution for early stopping slightly deviates from the other solutions.

5.3. Training time

Here, we want to know how many training steps are necessary to fulfil certain termination conditions of the training process. The timescale will depend upon the choice of the learning rate η. To compare the necessary training steps for different numbers of examples P, it is better to choose a learning rate, which is always a fraction of the maximal learning rate $\eta_{\max}(P)$, see (19).

We will consider two termination conditions:

• $E_T(t) \leqslant E_T(\min) + \epsilon$. To obtain ϵ strict zero would imply exhaustive training, which requires an infinite number of training steps. A non-zero $\epsilon > 0$ needs only a finite number of training steps and can be illustrated.

• $E_G(t+1) > E_G(t)$. This is known as optimal early stopping, in which training is

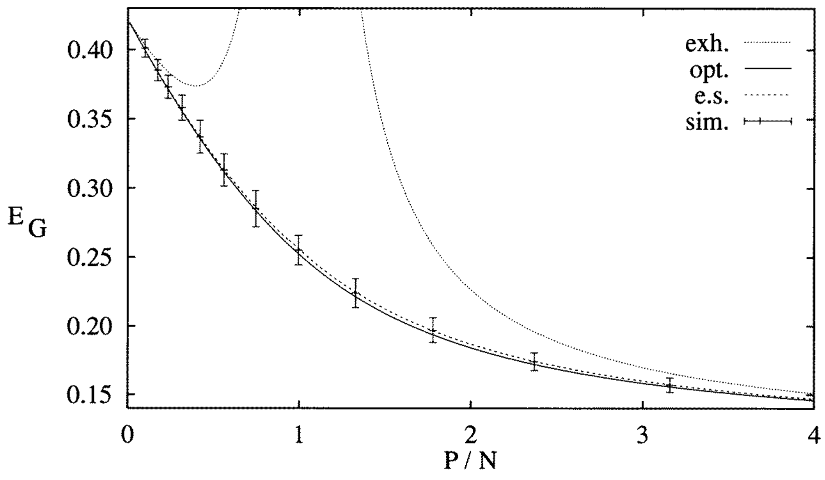

Figure 2. Performance of $E_G(\alpha)$ for early stopping and weight decay in both descriptions (equilibrium and dynamical). For exhaustive training, corresponding to unstopped training, $t \to \infty$, without weight decay, $\lambda = 0$, the results are in both approaches the same and shown as a dotted line (exh.). Of the four possible optimal curves, for early stopping and weight decay in both approaches, three are identical and shown as one full circle (opt.). In section 4 we saw that early stopping and weight decay are equivalent in the equilibrium approach. With this result coincides the exact optimal weight decay curve, which is given by the dynamical approach. The exact solution for early stopping, given by the dynamical approach, cannot fully reach the other results (broken curve). The simulations of early stopping are consistent with both descriptions, see error-bars. Parameters are as in figure 1.

terminated when the generalization error starts to increase.

In figure 3, it can be seen that the first condition always leads to diverging training times around the storage capacity $\alpha_c = 1$, resulting in overtraining. However, in early stopping the number of training steps always remains quite small.

5.4. Asymptotics

Figure 3 implies that asymptotically, very few training steps are necessary to fulfil the termination condition. This can be examined more precisely. Using the binomial series, all time-dependent integrals I_m^l, see (17), can be expressed as sums of time-independent integrals,

$$I_m^l(\alpha, \eta, t) = \sum_{i=0}^{l} \binom{l}{i} (-\eta)^i I_{m+i}^0(\alpha) \qquad \text{for } l = t, 2t. \tag{25}$$

The time-independent integrals I_m^0 depend only on α. Their leading order is $\mathcal{O}(\alpha^m)$, and the full dependence on α can be seen in tables 2 and 3.

For small numbers of training steps ($t = 1, 2, 3$), we can derive analytical expressions for the generalization error, if we use these expressions and choose the learning rate as $\eta(\alpha) = \eta_0\alpha^{-1}$. In the case $t = 1$, we obtain

$$E_G(t=1) = \frac{G - H^2}{2} + \frac{H^2}{2}(\eta_0 - 1)^2 + \frac{G}{2}\eta_0^2\frac{1}{\alpha} \tag{26}$$

which converges to the absolute minimum E_∞ only if $\eta_0 = 1$. The rate is slower than the optimal batch rate, see $E_G(\alpha > 1)$ in table 2.

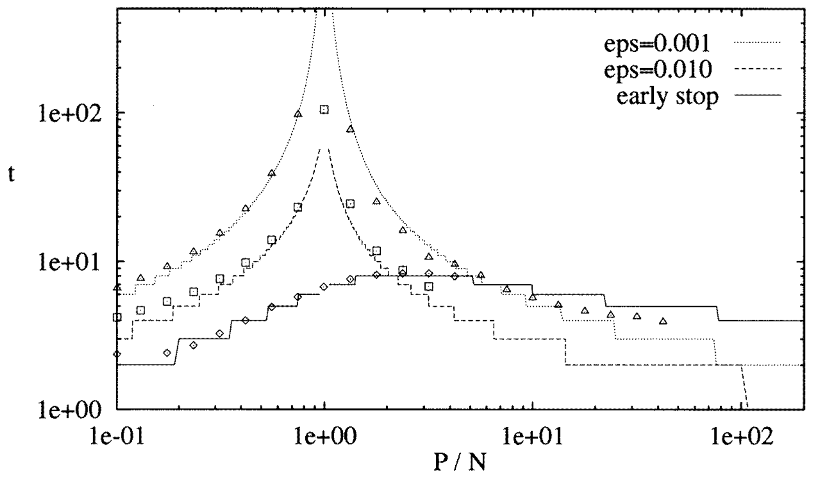

Figure 3. Necessary training steps to fulfil certain stopping criteria. Training until a certain accuracy is reached, i.e. until $E_T^{\min} + \epsilon$, is shown for $\epsilon = 0.001$ (dotted curve) and $\epsilon = 0.01$ (broken curve). In early stopping, training is terminated, when the generalization error increases, i.e. $E_G(t+1) > E_G(t)$. The result is shown by the full curve. The marks are simulation results. Parameters: $G = 0.84$, $H = 0.78$; $\eta = \alpha^{-1}$; and simulations with systems of size $N = 200$ averaged over 100 iterations.

Table 3. More values of time-independent integrals $I_m^0(\alpha)$ for $\alpha > 1$.

m	$I_m^0(\alpha)$
2	$\alpha + \alpha^2$
3	$\alpha + 3\alpha^2 + \alpha^3$
4	$\alpha + 6\alpha^2 + 6\alpha^3 + \alpha^4$
5	$\alpha + 10\alpha^2 + 20\alpha^3 + 10\alpha^4 + \alpha^5$
6	$\alpha + 15\alpha^2 + 50\alpha^3 + 50\alpha^4 + 15\alpha^5 + \alpha^6$

For $t = 2$ we obtain

$$E_G(t=2) = \frac{G - H^2}{2} + \frac{H^2}{2}(\eta_0 - 1)^4 + \frac{1}{\alpha}\left[\frac{G}{2}\eta_0^2(\eta_0 - 2)^2 + \frac{H^2}{2}\eta_0^2(5\eta_0^2 - 8\eta_0 + 2)\right] + \mathcal{O}\left(\frac{1}{\alpha^2}\right). \tag{27}$$

Again, only in the case of $\eta_0 = 1$, the absolute minimum is reached, this time with the same rate as the full batch result, i.e.

$$E_G(t=2) = \frac{G - H^2}{2}\left(1 + \frac{1}{\alpha}\right) + \mathcal{O}\left(\frac{1}{\alpha^2}\right). \tag{28}$$

The reader may use I_5^0 and I_6^0 from table 3 to calculate $E_G(t = 3)$. The result will show the same optimal convergence rate. A numerical test indicates that this rate is reached earlier.

These results are illustrated in figure 4. If terms of order α^{-2} can be neglected, then already two batch training steps are sufficient to reach the optimal convergence rate, if the learning is chosen as $\eta = \alpha^{-1}$.

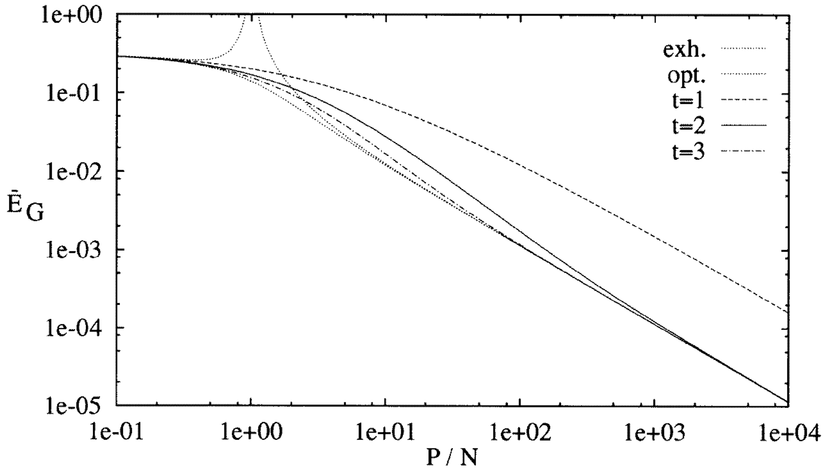

Figure 4. Asymptotical performance of the rest-error $\bar{E}_G = E_G - E_\infty$ for different numbers of training steps. The exhaustive result (upper dotted curve) corresponds to an infinite number of training steps ($t \to \infty$) and shows overtraining. Optimal early stopping uses per definition the optimal number of training steps ($t = t_{\text{opt}}$) and defines the lower bound (lower dotted curve). With only one batch training step, the same scaling as in the lower bound but with a different prefactor can be reached (broken curve). Two steps already yield the correct prefactor (full curve). With $t = 3, \ldots, 8$ the lower bound is reached earlier ($t = 3$, dotted curve). Parameters are as in figure 3.

6. Summary and discussion

In this paper, early stopping and weight decay in a linear perceptron were studied. The dynamical approach is directly related to the gradient descent update rule. Thus, the typical behaviour was found not only for a final equilibrium state, but also for each training step (see figure 1). Early stopping and weight decay can be studied in detail.

In [14], the same problem was addressed by an equilibrium approach. An approximative solution to early stopping and weight decay was found by using finite temperature solutions of the thermodynamic approach. A comparison of the two approaches (see figure 2) showed that the equilibrium solution is a very useful approximation. It is exact for weight decay and very good for early stopping. Which of the approaches is extendable from the linear machine to more complicated systems is an interesting, still open problem.

Another interesting issue was raised by observing the training times. It turned out that asymptotically, only a very small number of batch training steps is necessary to reach the optimal batch convergence, if the learning rate is chosen as $\eta = \alpha^{-1}$ (see figures 3 and 4). This result should be seen in relation to recent results on *on-line* training [25–30]. It was shown that on-line training can yield the same convergence rate as optimal batch training if many examples are available ($\alpha \to \infty$) and the learning rate is annealed during the training process as $\eta = \alpha^{-1}$. In on-line training the examples are presented only once in a sequence and after each new example all the weights are updated. Therefore, on-line training is obviously computationally much cheaper than batch training. However, if two batch training steps are already sufficient in order to reach the optimal convergence rate, then batch training is again competitive even with respect to computational costs. A better understanding of the relation between batch training with few training steps and on-line training with annealed learning rate would be desirable.

Finally, it is worth mentioning that the performance plots (figures 2 and 4) show three regimes that are characteristical for many learning tasks. From zero to the storage capacity α_c, the network is learning examples by heart or storing them, which leads to a rising generalization error if the task is unrealizable. We call this *storage regime*. Above α_c the network is generalizing, and the generalization error decreases again. This is called the *generalization regime*. We find it useful to introduce an *asymptotical subregime* starting from α_{asy}. In the asymptotical regime, the relative number of examples P/N is large, which allows some simplifying assumptions. One consequence of the simpler behaviour in the asymptotical regime is the similarity of the exhaustive and the optimal solution, which makes early stopping and weight decay unnecessary. Asymptotical results can be found in statistics and in statistical mechanics, see [31, 32] for example. It should be noted that the behaviour in the generalization regime below α_{asy} is much richer. For the perceptron, we know that $\alpha_c = 1$ and α_{asy} is of the order 10. Determining α_c and α_{asy} for more complicated networks is a necessary requirement, if one wants to understand their learning abilities better.

Acknowledgments

We would like to thank Shun-ichi Amari for stimulating discussions, Emily Helle and Adélaïde Stévenin-Barbier for careful reading and valuable comments on the manuscript.

Appendix A. Average over the teacher weights

Here we add some identities which are necessary to calculate the averages over the input distributions, see equations (13) and (14). Like in the equilibrium approach, it is based on the assumption that the distribution of the local fields h is Gaussian. The local fields h are essentially the inner product of weight vector $\boldsymbol{W}$ and input vector $\boldsymbol{x}$. Therefore, the assumption is fulfilled, if inputs x_i are random variables with mean zero and variance one. Alternatively we can keep the inputs fixed, and assume that the weights W_i are random variables with mean zero and variance one.

The Gaussian local fields h^*_μ have mean zero, i.e. $\langle h^*_\mu \rangle = 0$, and covariance

$$\langle h^*_\mu h^*_\nu \rangle_{h^*_\mu, h^*_\nu} = \left\{ \begin{array}{lll} \delta_{\mu\nu} & \text{for} & \mu = \nu \\ C_{\mu\nu} & \text{for} & \mu \neq \nu \end{array} \right\} = \delta_{\mu\nu} + (C_{\mu\nu} - \delta_{\mu\nu}). \tag{A1}$$

The two terms denote diagonal and off-diagonal part, a writing that we will always use.

To find our first identity, we have to decorrelate the local fields into uncorrelated Gaussians, $\tilde{h}_\mu$ and $\tilde{h}_\nu$, first. Then we calculate diagonal and off-diagonal term, with the assumption that $C_{\mu\nu}$ is small for $\mu \neq \nu$. We find

$$\begin{aligned} \langle z^*_\mu z^*_\nu \rangle_{\{x_{i\mu}\}} &= \langle g^* \left(\sqrt{1 - (C_{\mu\nu})^2} \tilde{h}^*_\mu + C_{\mu\nu} \tilde{h}^*_\nu \right) g^*(\tilde{h}_\nu) \rangle_{\tilde{h}^*_\mu, \tilde{h}^*_\nu} \\ &= \begin{cases} G & \text{for } \mu = \nu \\ \langle [g^*(\tilde{h}^*_\mu) + g^*(\tilde{h}^*_\mu)' C_{\mu\nu} \tilde{h}^*_\nu] g^*(\tilde{h}^*_\nu) \rangle_{\tilde{h}^*_\mu, \tilde{h}^*_\nu} & \text{for } \mu \neq \nu \end{cases} \\ &= \delta_{\mu\nu} G + (C_{\mu\nu} - \delta_{\mu\nu}) H^2. \end{aligned} \tag{A2}$$

In the same way, the following identity can be proved,

$$\langle z^*_\mu h^*_\nu \rangle_{\{x_{i\mu}\}} = \left\langle g^* \left(\sqrt{1 - (C_{\mu\nu})^2} \tilde{h}^*_\mu + C_{\mu\nu} \tilde{h}^*_\nu \right) \tilde{h}^*_\nu \right\rangle_{\tilde{h}^*_\mu, \tilde{h}^*_\nu}$$

$$= \delta_{\mu\nu} H + (C_{\mu\nu} - \delta_{\mu\nu}) H = C_{\mu\nu} H. \tag{A3}$$

Appendix B. Dynamical approach with weight decay

Weight decay can be included in the training error by adding a term that penalises a large norm of the weights, i.e.

$$\tilde{E}_T := \frac{1}{P}\left[\frac{1}{2}\sum_{\mu=1}^{P}(z_\mu^* - z_\mu)^2 + \frac{\lambda}{2}\sum_{i=1}^{N}(W_i)^2\right] \tag{B1}$$

where λ determines the relative strength of the weight decay term. Using this training error, the gradient descent learning rule becomes,

$$W_i(t+1) = (1-\eta\lambda)W_i(t) + \frac{\eta}{\sqrt{N}}\sum_{\mu=1}^{P}[z_\mu^* - z_\mu(t)]x_{i\mu}. \tag{B2}$$

Step 1. For $P < N$ and the linear student, $z_\mu = h_\mu = N^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{x}_\mu$, we can define again $\sigma_\mu(t)$ as above (9). The recursion for $\sigma_\mu(t)$ follows from equation (B2) with the *overlap matrix,* $\mathbf{C} = N^{-1}\boldsymbol{x}_\mu\boldsymbol{x}_\nu$. From the geometrical series we know the solution of this recursion,

$$W_i(t) = \frac{\eta}{\sqrt{N}}\sum_{\mu,\nu=1}^{P} z_\mu^* \left\{\frac{\mathbb{1} - [(1-\eta\lambda)\mathbb{1} - \eta\mathbf{C}]^t}{\mathbb{1} - [(1-\eta\lambda)\mathbb{1} - \eta\mathbf{C}]}\right\}_{\mu\nu} x_i^\nu. \tag{B3}$$

The initial conditions are not influenced by weight decay and remain $W_i(0) = 0$ and $W_i(1) = \eta N^{-\frac{1}{2}}\sum_{\mu=1}^{P} z_\mu^* x_{i\mu}$. However, the limit of infinitely many timesteps depends on the weight decay. We obtain the pseudoinverse including weight decay, i.e.

$$W_i(t\to\infty) = \frac{1}{\sqrt{N}}\sum_{\mu,\nu=1}^{P} z_\mu^*[(\lambda\mathbb{1} + \mathbf{C})^{-1}]_{\mu\nu}x_{i\nu}. \tag{B4}$$

Note, that for $\lambda > 0$ this solution can be extended to the case $P > N$. The introduction of the matrix $\mathbf{B}$ is then not necessary. Only the general form of the density of the eigenvalues has to be used (see references above). However, to be consistent with the solution without weight decay, $\lambda = 0$, we also use the matrix $\mathbf{B}$ here, that is as long as $P < N$.

Step 2. As an example, we calculate the behaviour of $\hat{R}(t)$ using expression (B3),

$$\hat{R}(t) = \left\langle \frac{1}{N}\sum_{\mu,\nu=1}^{P}\left[\frac{\mathbb{1} - (\mathbb{1} - \eta\lambda\mathbb{1} - \eta\mathbf{C})^t}{\lambda\mathbb{1} + \mathbf{C}}\right]_{\mu\nu}\langle z_\mu^* h_\nu^*\rangle_{\{x_{i\mu}\}}\right\rangle_{\{x_{i\mu}\}}$$
$$= \left\langle \frac{\alpha H}{P}\sum_{\mu=1}^{P}\left[\frac{\mathbb{1} - (\mathbb{1} - \eta\lambda\mathbb{1} - \eta\mathbf{C})^t}{\lambda\mathbb{1} + \mathbf{C}}\mathbf{C}\right]_{\mu\mu}\right\rangle_{\{x_{i\mu}\}}. \tag{B5}$$

The average is again expression (A3) from appendix A. With similar minor changes we receive $\hat{Q}(t)$ and the training error $E_T(t)$. The overdetermined case, i.e. $P > N$, can be determined in full analogy.

Step 3. Again we have integrals over the eigenvalues of $\mathbf{C}$ respectively $\mathbf{B}$, which are now of the form,

$$I_{lmn}(\alpha,\lambda,\eta,t) := \left\langle \frac{1}{P}\sum_{\mu=1}^{P}\left\{\frac{[\mathbb{1} - (\mathbb{1} - \eta\lambda\mathbb{1} - \eta\mathbf{C})^t]^l}{[\lambda\mathbb{1} + \mathbf{C}]^m}[\mathbf{C}]^n\right\}_{\mu\mu}\right\rangle_{\{x_{i\mu}\}}$$
$$= \int_{\xi_{\min}}^{\xi_{\max}} \mathrm{d}\xi\,\rho(\xi)\frac{[1 - (1 - \eta\lambda - \eta\xi)^t]^l}{[\lambda + \xi]^m}\xi^n \tag{B6}$$

with $l, m, n \in \{1, 2, 3\}$. These can be calculated using the same density of eigenvalues $\rho(\xi)$ as without weight decay. (18). The maximal learning rate is now twice the inverse of the sum of the weight decay strength λ and the maximal eigenvalue $\xi_{\max}$ of $\mathbf{C}$ or $\mathbf{B}$.

Without weight decay $\lambda = 0$, the integrals $I_{lmn}(\alpha, \lambda, \eta, t)$ can be expressed in terms of integrals $I_m^l(\alpha, \eta, t)$ from above (17), using the two relations, $I_{1mn} = I_{n-m}^0 - I_{n-m}^t$ and $I_{2mn} = I_{n-m}^0 - 2I_{n-m}^t + I_{n-m}^{2t}$.

B.1. Result

The results can be written in a compact form, if we introduce a constant c, which is $c = 1$ in the case of $\alpha < 1$, and $c = \alpha$ in the case of $\alpha > 1$,

$$\begin{aligned} E_G(\alpha, \lambda, \eta, t) &= \frac{G}{2a}(c + \alpha I_{221}) - \frac{H^2\alpha}{2c}(2I_{111} + I_{221} - I_{222}) \\ E_T(\alpha, \lambda, \eta, t) &= \frac{G}{2c}(c - 2I_{111} + I_{222}) + \frac{H^2}{2c}(2I_{111} - 2I_{112} - I_{222} + I_{223}). \end{aligned} \tag{B7}$$

References

[1] Bös S 1996 *Advances in Neural Information Processing Systems 8 (NIPS'95)* ed D S Touretzky, M C Mozer and M E Hasselmo (Cambridge, MA: MIT) p 218
[2] Akaike H 1974 *IEEE Trans. Autom. Control* **19** 716
[3] LeCun Y, Denker J S and Solla S 1990 *Advances in Neural Information Processing Systems 2 (NIPS'89)* ed D S Touretzky (San Mateo, CA: Morgan Kaufmann) p 598
[4] MacKay D J C 1992 *Neural Comput.* **4** 415
MacKay D J C 1992 *Neural Comput.* **4** 448
[5] Murata N, Yoshizawa S and Amari S 1993 *Advances in Neural Information Processing Systems 5 (NIPS'92)* ed S J Hanson *et al* (San Mateo, CA: Morgan Kaufmann) p 607
[6] Amari S, Murata N, Müller K-R, Finke M and Yang H 1997 *IEEE Trans. Neural Networks* **8** 985
[7] Barber D, Saad D and Sollich P 1995 *Neural Comput.* **7** 807
[8] Wang C, Venkatesh S S and Judd J S 1994 *Advances in Neural Information Processing Systems 6 (NIPS'93)* ed J D Cowan *et al* (San Mateo, CA: Morgan Kaufmann) p 305
[9] Bös S 1995 How to partition examples between cross-validation set and training set *Proc. 1995 Int. Symp. on Nonlinear Theory and its Applications (Nolta 95)* p 385
[10] Krogh A and Hertz J 1992 *Advances in Neural Information Processing Systems 4 (NIPS'91)* ed J E Moody *et al* (San Mateo, CA: Morgan Kaufmann) p 950
[11] Bös S, Kinzel W and Opper M 1993 *Phys. Rev.* E **47** 1384
[12] Bös S 1995 *Int. Conf. on Artificial Neural Networks (ICANN 1995)* EC & CIE p 111
[13] Watkin T L H, Rau A and Biehl M 1993 *Rev. Mod. Phys.* **65** 499
[14] Bös S 1998 Statistical mechanics approach to early stopping and weight decay *Phys. Rev.* E to be published
[15] Bös S and Opper M 1997 *Advances in Neural Information Processing Systems 9 (NIPS'96)* ed M C Mozer *et al* (Cambridge, MA: MIT) p 141
[16] Hertz J, Krogh A and Palmer R G 1991 *Introduction to the Theory of Neural Computation* (Redwood City, CA: Addison-Wesley)
[17] Krogh A 1992 *J. Phys. A: Math. Gen.* **25** 1119
[18] Krogh A 1992 *J. Phys. A: Math. Gen.* **25** 1135
[19] Opper M and Kinzel W 1995 *Models of Neural Networks* vol III, ed E Domany *et al* (Heidelberg: Springer) p 151
[20] Bruce A and Saad D 1994 *J. Phys. A: Math. Gen.* **27** 3355
[21] Opper M 1989 *Europhys. Lett.* **8** 389
[22] LeCun Y, Kanter I and Solla S A 1991 *Phys. Rev. Lett.* **66** 2396
[23] Sollich P 1995 *Advances in Neural Information Processing Systems 7 (NIPS'94)* ed G Tesauro *et al* (Cambridge, MA: MIT) p 207
[24] LeCun Y, Simard P and Pearlmutter B 1993 *Advances in Neural Information Processing Systems 5 (NIPS'92)* ed S J Hanson *et al* (San Mateo, CA: Morgan Kaufmann) p 156
[25] Kinouchi O and Caticha N 1992 *J. Phys. A: Math. Gen.* **25** 6243

[26] Biehl M, Riegler P and Stechert M 1995 *Phys. Rev.* E **52** 4624
[27] Kim J W and Sompolinsky H 1996 *Phys. Rev. Lett.* **76** 3031
[28] Van den Broeck C and Reimann P 1996 *Phys. Rev. Lett.* **76** 2188
[29] Opper M 1996 *Phys. Rev. Lett.* **76** 4671
[30] Bös S, Murata N, Amari S and Müller K-R 1997 How to choose the learning rate in on-line learning, submitted
[31] Seung H S, Sompolinsky H and Tishby N 1992 *Phys. Rev.* A **45** 6056
[32] Amari S, Fujita N and Shinomoto S 1992 *Neural Comput.* **4** 605