Learning dynamics on different timescales

J. Phys. A: Math. Gen. **32** (1999) 8655–8663. Printed in the UK PII: S0305-4470(99)00559-4

Learning dynamics on different timescales

Dominik Endres and Peter Riegler

Institut für Theoretische Physik, Julius–Maximilians–Universität, Am Hubland, D–97074 Würzburg, Germany

Received 23 December 1998, in final form 17 September 1999

Abstract. The special character of certain degrees of freedom in two-layered neural networks is investigated for on-line learning of realizable rules. Our analysis shows that the dynamics of these degrees of freedom can be put on a faster timescale than those remaining, with the profit of speeding up the overall adaptation process. This is shown for two groups of degrees of freedom: second-layer weights and bias weights. For the former case our analysis provides a theoretical explanation of phenomenological findings. The resulting learning algorithm is compared with natural gradient descent in order to check whether the proposed scaling can be naturally derived from that type of learning rule.

1. Introduction

Statistical mechanics has contributed deeply to the understanding of adaptive systems during the past decades. Among such systems are neural networks [1,3] which are capable of learning, i.e. of adapting themselves to a desired state by means of examples. As learning tasks can be characterized by a certain amount of inherent randomness and a number of degrees of freedom which is typically large, physics, and, in particular, statistical mechanics often provides a means to understand such phenomena. The tools used to analyse such systems, e.g. thermodynamic limit [1] and stochastic differential equations [2], allow one to describe learning processes under a variety of circumstances, such as different architectures and training algorithms [1,3]. In addition, recent contributions [4–6] have shown how to compute optimal algorithms starting from first principles.

In this paper, statistical mechanics is used to analyse learning in specific two-layered neural networks. Such networks realize an input–output relation:

$$\sigma(\boldsymbol{\xi}) = \sum_{j=1}^{K} w_j g(\boldsymbol{J}_j \cdot \boldsymbol{\xi} + \vartheta_j) \tag{1}$$

where $g(\cdot)$ is a sigmoidal function and $\mathcal{W} = \{\boldsymbol{J}_j, w_j, \vartheta_j\}_{j=1,\ldots,K}$ denotes the set of *weights* of the network. The N-dimensional vector $\boldsymbol{J}_j$ corresponds to the synaptic couplings of a first-layer branch in a two-layered neural network, while w_j denotes the second-layer weights connecting the jth input branch with the output node. The weights, ϑ_j, are usually referred to as *biases*. Given an array of N inputs $\boldsymbol{\xi}$, the network computes its output $\sigma(\boldsymbol{\xi})$ according to (1).

Two-layered networks of the form (1) can implement any continuous input–output relation $\boldsymbol{\xi} \in \mathbb{R}^N \to \tau \in \mathbb{R}$ [7] if the number of hidden units K is unrestricted. That is, given a set of *training examples*, $\mathcal{D} = \{\boldsymbol{\xi}(n), \tau(n)\}_{n=1,\ldots,\alpha N}$ the network can adjust its weights $\mathcal{W}$ in order to

0305-4470/99/498655+09$30.00

implement the function $\tau(\boldsymbol{\xi})$ as accurately as desired. In learning theory this target function $\tau(\boldsymbol{\xi})$ is usually parametrized:

$$\tau(\boldsymbol{\xi}) = \sum_{j=1}^{M} v_j g(\boldsymbol{B}_j \cdot \boldsymbol{\xi} + \varphi_j). \tag{2}$$

This function can be viewed to be represented by a so-called *teacher network* with weights $\mathcal{B} = \{\boldsymbol{B}_j, v_j, \varphi_j\}$. The learning task can then be metaphorically described as follows: a *student network* of functional form (1) is trained by means of examples $\mathcal{D}$ provided by a teacher network. The student's task is to extract the teacher weights $\mathcal{B}$ and, consequently, the functional relation $\tau(\boldsymbol{\xi})$ from these examples. This is achieved by means of a *learning algorithm* which describes how to use information contained in the training set in order to adjust the weights $\mathcal{W}$.

Recently, *on-line algorithms* have attracted considerable attention. For on-line learning the presentation of examples used in the learning process occurs in a sequential manner. At presentation of example $\boldsymbol{\xi}(n)$, each weight, $W \in \mathcal{W}$, is updated according to

$$W(n+1) = W(n) + \frac{1}{N}\eta_W f_W(\mathcal{W}(n), \boldsymbol{\xi}(n), \tau(n)). \tag{3}$$

If one views n as a (discrete) time index, equation (3) describes the time evolution of the network weights. The *weight function* f defines the on-line learning algorithm which describes how the weights $\mathcal{W}(n)$ of the student network ought to be changed in response to a given example, $\{\boldsymbol{\xi}(n), \tau(n)\}$, at time step n.

Our main focus here is not on a clever choice of the training algorithm, i.e. the functional form of f, but on its proper scaling. In (3) we have separated out this scaling into the quantity η_W which is usually referred to as the *learning rate*. The only requirement we impose on $f = \mathcal{O}(1)$ is that it vanishes at the desired solution $\mathcal{W} = \mathcal{B}$. Thus, we only consider perfectly realizable tasks ($M = K$) here, where $\mathcal{B}$ is a fixed point in the dynamics of $\mathcal{W}$. In addition, we focus on networks having a finite number of hidden units, i.e. $K = \mathcal{O}(1)$, while N is large.

The paper is organized as follows: in section 2 we will review the learning dynamics of standard backpropagation as developed in [9, 10]. This will motivate a rescaling of biases and second-layer weights discussed in section 3. Section 4 investigates the question of whether this scaling behaviour can be derived from natural gradient descent. Finally, section 5 summarizes the results.

2. Results for standard backpropagation

We restrict ourselves to on-line backpropagation [8], since for this choice of algorithm the mathematical burden reduces significantly. In particular, averages can be performed analytically [8, 9] if one chooses the network's transfer functions g to be the error function $g(z) = \mathrm{erf}\left(z/\sqrt{2}\right)$. However, the essential results of this work hold for any adaptive dynamics of type (3). See [11] for details.

For on-line backpropagation the dynamics of the weights (3) reads

$$\begin{aligned}
\boldsymbol{J}_i(n+1) &= \boldsymbol{J}_i(n) - \frac{\eta_J}{N}\nabla_{\boldsymbol{J}_i}\epsilon(\mathcal{W}, \boldsymbol{\xi}) = \boldsymbol{J}_i(n) + \frac{\eta_J}{N}\delta_i\boldsymbol{\xi}(n) \\
w_i(n+1) &= w_i(n) - \frac{\eta_w}{N}\frac{\partial}{\partial w_i}\epsilon(\mathcal{W}, \boldsymbol{\xi}) = w_i(n) + \frac{\eta_w}{N}g(x_i + \vartheta_i)(\tau - \sigma) \\
\vartheta_i(n+1) &= \vartheta_i(n) - \frac{\eta_\vartheta}{N}\frac{\partial}{\partial \vartheta_i}\epsilon(\mathcal{W}, \boldsymbol{\xi}) = \vartheta_i(n) + \frac{\eta_\vartheta}{N}\delta_i
\end{aligned} \tag{4}$$

where $\delta_i = w_i g'(x_i + \vartheta_i)(\tau - \sigma)$. The quantities $x_i = \boldsymbol{J}_i \cdot \boldsymbol{\xi}$ and $y_i = \boldsymbol{B}_i \cdot \boldsymbol{\xi}$ denote the *internal fields* of the student and teacher network, respectively. The quadratic error measure, $\epsilon(\mathcal{W}, \boldsymbol{\xi}) = \frac{1}{2}[\sigma(\boldsymbol{\xi}) - \tau(\boldsymbol{\xi})]^2$, quantifies the degree of disagreement between the student and the rule output for a particular random input $\boldsymbol{\xi}$. Denoting the average over the input distribution by $\langle \ldots \rangle_\xi$ we define the *generalization error* $\epsilon_g = \langle \epsilon(\mathcal{W}, \boldsymbol{\xi}) \rangle_\xi$. It measures the validity of the student's hypothesis for the rule $\tau(\boldsymbol{\xi})$.

The statistical mechanics analysis of on-line learning basically consists of two steps: the introduction of order parameters and the average over the randomness of the training examples. This allows one to investigate *typical* behaviour together with the reduction of an extensive number of degrees of freedom, $\mathcal{W}$, to a finite number of meaningful observables. The very property of these order parameters is to be *self-averaging*, i.e. their fluctuations vanish in the thermodynamic limit $N \to \infty$. The practical difficulty, however, consists of finding appropriate order parameters such that the resulting macroscopic equations of motion can be written in a closed form after averaging over the distribution of inputs.

We exemplify the theoretical analysis of on-line learning for the simplest two-layer network. This consists of only one hidden unit ($K = 1$) and no bias weights ($\vartheta = 0 = \varphi$): $\sigma = w\mathrm{erf}(\boldsymbol{J}\cdot\boldsymbol{\xi}/\sqrt{2})$. Following the proposal of [10] we choose $R = \boldsymbol{B}\cdot\boldsymbol{J}$, $Q = \boldsymbol{J}\cdot\boldsymbol{J}$ and w as the order parameters. The first-layer order parameters R and Q describe the overlaps between the first-layer weights of the teacher $\tau = v\mathrm{erf}(\boldsymbol{B}\cdot\boldsymbol{\xi}/\sqrt{2})$ and student network, respectively. These are the familiar order parameters of perceptron learning (see, e.g., [1]) and learning in so-called soft committee machines [8,9].

In order to motivate this work we shortly recall the analysis for the case where both η_J and η_w are $\mathcal{O}(1)$ [10,11]. Starting from the corresponding microscopic equations of motion (4) for this simple network, it is straightforward to derive recursion relations for the mean values of R, Q and w by performing the average over the latest example input [8,9]. Since these quantities become self-averaging in the thermodynamic limit $N \to \infty$, the description in terms of their mean values is sufficient. In the same limit, one can interpret $\alpha = n/N$ as a continuous time and obtain ordinary differential equations for the evolution of the learning network:

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \eta_J \langle \delta y \rangle \qquad \frac{\mathrm{d}Q}{\mathrm{d}\alpha} = 2\eta_J \langle \delta x \rangle + \eta_J^2 \langle \delta^2 \rangle \qquad \frac{\mathrm{d}w}{\mathrm{d}\alpha} = \eta_w \langle g(x)(\tau - \sigma) \rangle. \tag{5}$$

The averages are over the two-dimensional Gaussian distribution of the internal fields $\{x, y\}$ which is determined through the correlations $\langle x^2 \rangle = Q$, $\langle xy \rangle = R$ and $\langle y^2 \rangle = T$.

The macroscopic equations of motion (5) are easily integrated numerically. The asymptotic learning behaviour can be obtained analytically by a linearization of (5) around the fixed point $R = Q = T$, $w = v$. The maximum eigenvalue, $\lambda_{\max}$, of the linearization matrix determines the speed of the exponential convergence towards the fixed point. Figure 1 shows the eigenvalue spectrum as a function of the first-layer learning rate η_J.

Of particular interest is the critical learning rate $\eta_{J,c}$. Only for $\eta < \eta_{J,c}$ does the student network converge to the teacher network. A detailed analysis shows that $\eta_{J,c}$ is independent of the second-layer learning rate η_w. Consequently, the student network can learn the rule $\tau(\boldsymbol{\xi})$ perfectly for any value of η_w, as long as $\eta_J < \eta_{J,c}$.

The fact that convergence will not be destroyed for any choice of η_w leads naturally to the conclusion that one should optimize the speed of convergence with respect to η_w. One observes that the eigenvalue λ_2 which dominates the convergence for most $\eta < \eta_{J,c}$ assumes its optimal value λ_2^{opt} as $\eta_w \to \infty$. Obviously, the divergence of η_w indicates that we should have chosen a different scaling for the change of the second-layer weight w. However, from the above analysis it is not quite clear what kind of scaling this would be. Therefore, we are going to re-analyse the microscopic dynamics (4).

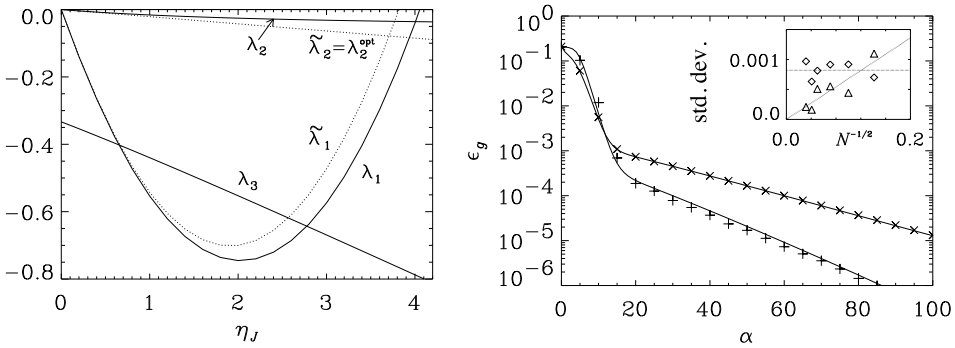

Figure 1. Left: eigenvalues of the linearization matrix governing the asymptotics of (5) (λ_i) and (7) ($\tilde{\lambda}_i$), for $T = v = \eta_w = \tilde{\eta}_w = 1$. Right: generalization error ϵ_g for two different types of scaling for the update of the second layer. For the first type ($\times$) the update of w has been chosen to scale with $1/N$ while it is of $\mathcal{O}(1)$ in the second case (+). Symbols represent simulations obtained for a system with $N = 100$ averaged over 100 runs, lines show the macroscopic equations of motion ($Q(0) = 1$, $w(0) = 0.5$, $R(0) = \mathcal{O}(1/\sqrt{N})$, $\eta_J = 2$). Errorbars would be smaller than the symbol size.

3. Rescaling the learning rates for biases and second-layer weights

3.1. Second-layer weight on a different timescale

Without loss of generality, we had chosen the component J_i of the student's weight vector to be $\mathcal{O}(1/\sqrt{N})$ and the random inputs $\xi_i = \mathcal{O}(1)$ with zero mean and unit variance. Together with the choice ϑ_i, $\varphi_i = \mathcal{O}(1)$ this assures that the arguments of the transfer function g in (1) are $\mathcal{O}(1)$. Moreover, in order to make the overall outputs σ, τ become $\mathcal{O}(1)$ the second-layer weights w should be $\mathcal{O}(1/K)$, i.e. $w_i = \mathcal{O}(1)$ for the networks considered here. Considering the scaling η_J we observe that for $\eta_J = \mathcal{O}(1)$ the change of the internal fields $x_i(n+1) - x_i(n) = \eta_J \delta_i$ is $\mathcal{O}(1)$. Hence, the change of the instantaneous error ϵ per learning step is $\mathcal{O}(1)$. The order of magnitude of this change does not alter if one chooses Δw_i, $\Delta\vartheta_i = \mathcal{O}(N^m)$ with $m \leqslant 0$. In the following we will restrict ourselves to the largest change ($m = 0$) which corresponds to η_w, $\eta_\vartheta = \mathcal{O}(N)$. This particular scaling of learning rates leads to the dynamics

$$\begin{aligned} \boldsymbol{J}_i(n+1) &= \boldsymbol{J}_i(n) + \frac{\eta_J}{N}\delta_i \boldsymbol{\xi}(n) \\ w_i(n+1) &= w_i(n) + \tilde{\eta}_w g(x_i + \vartheta_i)(\tau - \sigma) \\ \vartheta_i(n+1) &= \vartheta_i(n) + \tilde{\eta}_\vartheta \delta_i \end{aligned} \tag{6}$$

where we have defined $\eta_w = \tilde{\eta}_w N$ and $\eta_\vartheta = \tilde{\eta}_\vartheta N$.

Defining the timescale $\alpha = n/N$ as above, one immediately notices that the second-layer weights w and the biases ϑ change on a much faster timescale than the first-layer weights $\boldsymbol{J}$. For instance, typically $\mathcal{O}(N)$ many learning steps are necessary in order to achieve a change of $\boldsymbol{J}_i$ of order $\mathcal{O}(1)$, while for w_i typically only one step is required.

As before, we exemplify the analysis of the dynamical system (6) for the simple two-layered network $\sigma = wg(\boldsymbol{J}\cdot\boldsymbol{\xi})$. The profound difference in timescales becomes even more clear when we write (6) in terms of the macroscopic degrees of freedom, R and Q:

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \eta_J \overline{\langle \delta y \rangle} \qquad \frac{\mathrm{d}Q}{\mathrm{d}\alpha} = 2\eta_J \overline{\langle \delta x \rangle} + \eta_J^2 \overline{\langle \delta^2 \rangle} \tag{7}$$

$$\overline{w(n+1)} = \overline{w(n)} + \tilde{\eta}_w \overline{\langle g(x)(\tau - \sigma)\rangle} = \tilde{\eta}_w \langle v g(y) - \overline{w(n)} g(x)\rangle. \tag{8}$$

We have to study the combined dynamics of $\{R, Q\}$ and w. As the timescales of these two processes differ by a factor N we can adiabatically eliminate [12,13] the fast variable w in the thermodynamic limit. This basically means that we can act as if w has reached its stationary distribution for fixed order parameters R and Q, and use this distribution to compute the averages on the right-hand sides of (7). This additional average has been denoted by overbars while the average over the internal fields x and y is symbolized by $\langle \ldots \rangle$, as before. Note that in contrast to the dynamics (3), (5); w is no longer self-averaging for a scaling $\Delta w = \mathcal{O}(1)$.

The equilibrium value $\overline{w}(\alpha)$ is easily obtained from the equilibrium condition $v\langle g(y)\rangle - \overline{w}(\alpha)\langle g(x)\rangle = 0$ and, hence, depends on $R(\alpha)$ and $Q(\alpha)$ only. Similarly, one obtains the equilibrium value $\overline{w^2}(\alpha)$ from the corresponding mean dynamics of w^2:

$$\overline{w^2(n+1)} = \overline{w^2(n)} + 2\tilde{\eta}_w \overline{w(n)\langle g(x)(\tau - \sigma)\rangle} + \tilde{\eta}_w^2 \langle g^2(x)\overline{(\tau - \sigma)^2}\rangle. \tag{9}$$

Simulations indicate that the equilibrium distribution of w can be assumed to be Gaussian (and uncorrelated with x and y) with a good degree of accuracy. Therefore, $\overline{w}(\alpha)$ and $\overline{w^2}(\alpha)$ can be used to eliminate all moments of w on the right-hand sides of (7) which is then a coupled system of only two macroscopic degrees of freedom.

The numerical solution of the remaining equations of motion for R and Q is in good agreement with simulations, cf figure 1. As before, the asymptotic dynamics is obtained by linearizing the two-dimensional system (7) around the fixed point. The resulting matrix has the eigenvalue λ_2^{opt} which is exactly the same as the dominating eigenvalue of (5) optimized with respect to η_w. (The second eigenvalue $\tilde{\lambda}_1$ is $\tilde{\eta}_w$-dependent with $\tilde{\lambda}_1 \to \lambda_1$ as $\tilde{\eta}_w \to 0$.) Thus the divergence of η_w discussed above indicates that the change of the second-layer weight w can be as large as $\mathcal{O}(1)$ and should be larger than $\mathcal{O}(1/N)$. As already pointed out, this result can be shown to be independent of the particular choice of learning algorithm (3) [11].

In addition, the result can be easily generalized to two-layer networks with $K = \mathcal{O}(1)$ many hidden units. It provides a theoretical explanation of the phenomenological rule that the change of a weight attached to a certain node in a multi-layer network should scale with the inverse of the 'fan-in', i.e. the number of couplings projecting into that node (see, e.g., [3] and references therein), that is $\Delta \boldsymbol{J} \simeq 1/N$ and $\Delta w \simeq 1/K = \mathcal{O}(1)$ in our case.

3.2. Bias on a different timescale

Our reasoning that leads to the rescaled update rule, (6), suggests that bias weights should be put on a faster timescale as well. We are going to illustrate this for simple perceptron learning: a student network $\sigma = \mathrm{erf}((\boldsymbol{J}\cdot\boldsymbol{\xi} + \vartheta)/\sqrt{2})$ is trained by examples originating from a teacher network of the same architecture, $\tau = \mathrm{erf}((\boldsymbol{B}\cdot\boldsymbol{\xi} + \varphi)/\sqrt{2})$. As before, we compare the rescaled backpropragation dynamics of type (6) with the 'traditional' dynamics (4).

Although not all averages with respect to the internal fields x, y can be performed analytically, the macroscopic equations of motion can be easily numerically integrated for the 'traditional' scaling [17]. In this case the dynamics is described by R, Q and ϑ, all three of which have the property to be self-averaging.

In contrast, a scaling of type (6) requires an adiabatic elimination of the fast variable ϑ. The analysis follows along the same line as before: from the microscopic equations of motion for $\vartheta(n)$ and $\vartheta^2(n)$ one obtains the equilibrium values $\overline{\vartheta}(\alpha)$, $\overline{\vartheta^2}(\alpha)$ which we assume to be sufficient to describe the distribution of w at a given time α. By inserting these equilibrium values into the equations of motion for R and Q one eliminates the fast variable ϑ, adiabatically. The remaining two-dimensional system in R and Q can be solved numerically, see figure 2.

For the 'traditional' update of the bias ϑ, the analysis is completely equivalent to the one for the second-layer weights. The eigenvalues of the corresponding linearization matrix show

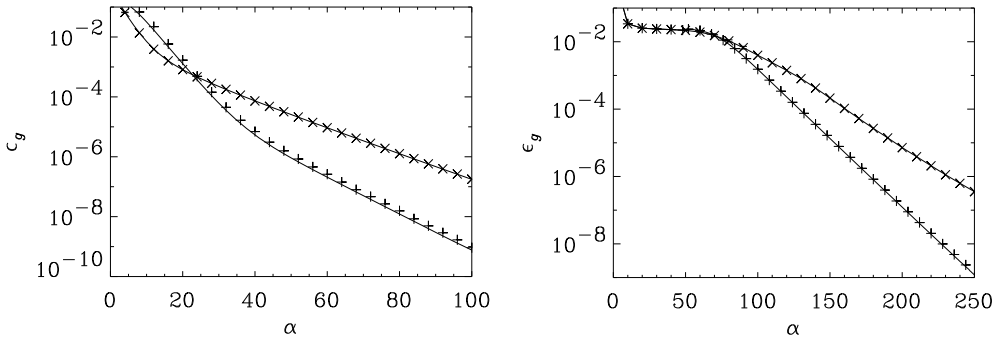

Figure 2. Left: generalization error ϵ_g for two different types of scaling for the update of the bias weight ($\varphi = 1$, $T = 1$, $\eta_J = \eta_\vartheta = \tilde{\eta}_\vartheta = 1$, initial values and symbols as in figure 1). Right: comparison of the generalization error of a two-layer network with $K = 2$ hidden units for the two different types of scaling of the bias weights ($\boldsymbol{B}_i \cdot \boldsymbol{B}_j = \delta_{ij}$, $w_i = v_i = 1$ fixed, $\varphi_i = 1$, $\eta_\vartheta = 0.5 = \tilde{\eta}_\vartheta$, $\eta_J = 0.8$).

the generic behaviour as in figure 1: there is a critical value of η_J above which the rule cannot be perfectly learned. This critical learning rate is independent of the bias's learning rate η_ϑ. Optimizing the dynamics with respect to η_ϑ (in the range of η_J where λ_2 is dominant) leads to $\eta_\vartheta^{\mathrm{opt}} \to \infty$ and the same dynamics as for the rescaled updating (6) with $\eta_\vartheta = \tilde{\eta}_\vartheta N$.

In order to indicate that our results do not just apply to the examples discussed, figure 2 shows the evolution of the generalization error for a soft-committee machine ($w_i = v_i$) [9] of type (1) with $K = 2$ hidden units. Comparison is made between backpropagation learning of type (4) and the dynamics where the change of biases weights per learning step is $\mathcal{O}(1)$, cf (6). As can be seen a dynamics of biases on a faster timescale compared with the weights $\boldsymbol{J}_i$ leads to a significantly faster decay of the generalization error.

4. Comparison with natural gradient descent

In the preceding section, we have shown that the convergence speed of a backpropagation learning process increases if one rescales the learning rates of biases and second-layer weights. One might be led to the conjecture that such a scaling behaviour might follow naturally from on-line algorithms which yield asymptotically efficient estimation. Recently, Amari [14, 15] has proposed such an algorithm known as natural gradient descent based on a differential geometric approach. In this section we will investigate whether the scaling of learning rates discussed in section 3 automatically arises from natural gradient descent algorithms. The analysis follows along the line of [15], where the special case of perceptron learning has been analysed in detail. Perceptron learning corresponds to the case $K = M = v = w = 1$ in (1), (2).

The general update rule for on-line natural gradient descent reads

$$\mathcal{W}(n+1) = \mathcal{W}(n) - \eta(n) G^{-1} \nabla_{\mathcal{W}} \epsilon(\mathcal{W}(n), \boldsymbol{\xi}(n), \tilde{\tau}(n)). \tag{10}$$

Here, $\tilde{\tau} = \tau + \nu$ denotes the teacher output which is distorted by additive noise ν of zero mean and variance Σ^2. The Fisher information metric G will be defined below. Note that in contrast to backpropagation (3), the update is determined by the gradient along the full set of weights, $\mathcal{W} = \{\boldsymbol{J}_j, w_j, \vartheta_j\}_{j=1,\ldots,K}$, with additional rotation and rescaling given by the inverse of the Fisher metric.

At this point it is important to note that in (10) the learning rate η is the same for every

component of $\mathcal{W}$. This means, in particular, that the learning rate does not scale differently with N for different components of $\mathcal{W}$. Moreover, in general, an annealing schedule $\eta(n) \propto 1/n$ is required to assure convergence of the generalization error of $\epsilon_g \propto 1/n$, see [15, 16] for details.

A different scaling of the updates in (10) can only arise if the elements of G^{-1} are of different orders of magnitude. Consequently, it is sufficient to calculate the elements of G^{-1} in order to find out whether the scaling recipe of section 3 naturally arises from natural gradient descent.

The elements of the Fisher information metric are given by

$$G_{ij} = \left\langle \frac{\partial}{\partial \mathcal{W}_i} \log p(\tilde{\tau}, \boldsymbol{\xi}; \mathcal{W}) \frac{\partial}{\partial \mathcal{W}_j} \log p(\tilde{\tau}, \boldsymbol{\xi}; \mathcal{W}) \right\rangle_{\nu,\xi} \tag{11}$$

where the average is over the joint distribution $p(\tilde{\tau}, \boldsymbol{\xi}; \mathcal{W})$ of inputs $\boldsymbol{\xi}$ and the additive noise ν. As in section 2 we restrict ourselves to the special case of teacher and student networks with zero bias and one hidden unit, i.e. $\mathcal{W} = \{\boldsymbol{J}, w\}$ and $\mathcal{B} = \{\boldsymbol{B}, v\}$. For this case, the components of G can be calculated explicitly, yielding

$$\left\langle \frac{\partial}{\partial J_i} \log p \frac{\partial}{\partial J_j} \log p \right\rangle = a\delta_{ij} + bJ_iJ_j \tag{12}$$

$$\left\langle \left(\frac{\partial}{\partial w} \log p \right)^2 \right\rangle = c \tag{13}$$

$$\left\langle \frac{\partial}{\partial w} \log p \frac{\partial}{\partial J_i} \log p \right\rangle = dJ_i \tag{14}$$

where

$$a = \frac{w^2}{\Sigma^2} \frac{2}{\pi\sqrt{1+2Q^2}} \tag{15}$$

$$b = -\frac{w^2}{\Sigma^2} \frac{4}{\pi(1+2Q^2)^{3/2}} \tag{16}$$

$$c = \frac{1}{\Sigma^2} \int \mathrm{D}\epsilon [\mathrm{erf}(Q\epsilon/\sqrt{2})]^2 \tag{17}$$

$$d = \frac{w}{\Sigma^2} \frac{2}{\pi\sqrt{1+2Q^2}(1+Q^2)}. \tag{18}$$

Given w and $Q = |\boldsymbol{J}|$, the inverse G^{-1} needed in (10) can be calculated. It is straightforward to show that

$$G^{-1} = \begin{pmatrix} \tilde{a} I_{N\times N} + \tilde{b}\boldsymbol{J}\boldsymbol{J}^\top & \tilde{d}\boldsymbol{J} \\ \tilde{d}\boldsymbol{J}^\top & \tilde{c} \end{pmatrix} \tag{19}$$

where $I_{N\times N}$ denotes the N-dimensional unity matrix and

$$u = ac + (bc - d^2)Q^2 \tag{20}$$

$$\tilde{a} = 1/a \tag{21}$$

$$\tilde{b} = (d^2 - bc)/(au) \tag{22}$$

$$\tilde{c} = (a + bQ^2)/u \tag{23}$$

$$\tilde{d} = -d/u. \tag{24}$$

From (12)–(24) it is evident that all the elements of the Fisher matrix are of the same order of magnitude, i.e. $\mathcal{O}(1)$. They do not scale with N. The same holds true for the inverse G^{-1}.

For the changes of the student's degrees of freedom one obtains the specific update rule

$$\begin{aligned} \boldsymbol{J}(n+1) - \boldsymbol{J}(n) &= \eta(n)(\tilde{\tau} - wg)\{\tilde{a}g'\boldsymbol{\xi} + [\tilde{b}(\boldsymbol{J}^\top\boldsymbol{\xi})g' + \tilde{d}g]\boldsymbol{J}\} \\ w(n+1) - w(n) &= \eta(n)(\tilde{\tau} - wg)\{\tilde{d}(\boldsymbol{J}^\top\boldsymbol{\xi})g' + \tilde{c}g\} \end{aligned} \tag{25}$$

where g and g' are to be evaluated at $x(n) = \boldsymbol{J}(n) \cdot \boldsymbol{\xi}(n)$. Consequently, the changes of every component of $\mathcal{W}$ in (10) are of the same order of magnitude, *unless* one deliberately introduces learning rates which scale differently with N for different components of $\mathcal{W}$. The latter case is the one analysed in section 3. Thus, natural gradient descent does not automatically lead to a learning dynamics where the updates of different degrees of freedom of the student network are of different orders of magnitude.

5. Summary

We have investigated the scaling of learning rates for on-line learning in simple two-layer networks. Our findings show that a specific rescaling of the learning rate of biases and second layer weights leads to a faster overall convergence of the generalization error. For the case of second-layer weights this result is in accordance with the 'fan-in' rule of thumb well known in application of neural networks.

We have shown that this rescaling effectively leads to a dynamics of the respective weights which takes place on a faster timescale as compared with the dynamics of the input-to-hidden weights. An analytic solution of the asymptotical learning dynamics has been obtained by an adiabatic elimination of the fast degrees of freedom.

In this work, we did not focus on the *optimal* choice of this timescale. For instance, a scaling Δw, $\Delta\vartheta = \mathcal{O}(1/\sqrt{N})$ might give rise to even faster convergence. Furthermore, it is presently not clear how the discussed scaling recipes can be derived from first principles. Here, we have shown that natural gradient descent can be ruled out as an explanation for these scaling recipes. An explanation starting from first principles remains a source for further research as well as a possible extension of the above analysis to systems where $K = \mathcal{O}(N)$.

Acknowledgments

This work has been supported by the Deutsche Forschungsgemeinschaft. The authors acknowledge stimulating discussions with M Biehl, M Copelli, G Reents and R Urbanczik.

References

[1] Watkin T L H, Rau A and Biehl M 1993 *Rev. Mod. Phys.* **65** 499
[2] Heskes T and Kappen B 1991 *Phys. Rev.* A **44** 2714
[3] Hertz J A, Krogh A and Palmer R G 1991 *Introduction to the Theory of Neural Computation* (Redwood City, CA: Addison-Wesley)
[4] Kinouchi O and Caticha N 1992 *J. Phys. A: Math. Gen.* **25** 6243
[5] Copelli M *et al* 1997 *Europhys. Lett.* **37** 427
[6] Reimann P and Van den Broeck C 1996 *Phys. Rev. Lett.* **76** 2188
[7] Cybenko G 1989 *Math. Control Signals Syst.* **2** 303
[8] Biehl M and Schwarze H 1995 *J. Phys. A: Math. Gen.* **28** 1995
[9] Saad D and Solla S A 1995 *Phys. Rev.* E **52** 4225
[10] Riegler P and Biehl M 1995 *J. Phys. A: Math. Gen.* **28** L507
[11] Riegler P 1997 Dynamics of on-line learning in neural networks *PhD Thesis* (Marburg: Tectum)
[12] Gardiner C W 1990 *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences* (Berlin: Springer)
[13] Heskes T and Coolen J 1997 *J. Phys. A: Math. Gen.* **30** 4983

[14] Amari S 1996 *Advances in Neural Information Processing Systems 9* ed M C Mozer *et al* (Cambridge, MA: MIT Press)
[15] Amari S 1997 *Neural Comput.* **10** 251
[16] Opper M 1996 *Phys. Rev. Lett.* **77** 4671
[17] West A H L, Saad D and Nabney I 1996 *Advances in Neural Information Processing Systems 9* ed M C Mozer *et al* (Cambridge, MA: MIT Press)