Matrix momentum for practical natural gradient learning

1999 J. Phys. A: Math. Gen. 32 4047

(http://iopscience.iop.org/0305-4470/32/22/305)

J. Phys. A: Math. Gen. **32** (1999) 4047–4059. Printed in the UK PII: S0305-4470(99)99570-7

Matrix momentum for practical natural gradient learning

Silvia Scarpetta†‖, Magnus Rattray‡¶ and David Saad§[+]

† Department of Physics 'E R Caianiello', Salerno University, Baronissi (SA), Italy and INFM, Sezione di Salerno, Italy
‡ Computer Science Department, University of Manchester, Manchester M13 9PL, UK
§ Neural Computing Research Group, Aston University, Birmingham B4 7ET, UK

Received 21 November 1998

Abstract. An on-line learning rule, based on the introduction of a matrix momentum term, is presented, aimed at alleviating the computational costs of standard natural gradient learning. The new rule, natural gradient matrix momentum, is analysed in the case of two-layer feed-forward neural network learning via methods of statistical physics. It appears to provide a practical algorithm that performs as well as standard natural gradient descent in both the transient and asymptotic regimes but with a hugely reduced complexity.

1. Introduction

Feed-forward neural networks are widely used in classification and regression applications mainly due to their ability to learn continuous and discrete input–output maps [1]. Here we concentrate on learning a *teacher* map f_B, from an input space $\boldsymbol{\xi} \in \mathbb{R}^N$ onto a scalar ζ, by modifying the parameters $\boldsymbol{J}$ of a *student* network according to some training algorithm, aimed at bringing the student map f_J as close as possible to f_B.

On-line learning is a popular method for training multi-layer feed-forward neural networks, where network parameters are updated according to only the latest in a sequence of training examples. This is contrasted to batch learning which utilizes the entire training set at each learning iteration. On-line methods can be beneficial in terms of both storage and computation time, and also allow for temporal changes in the task being learned (for an overview of on-line learning methods in neural networks see [2]).

Natural gradient (NG) learning [3] was recently proposed as a more principled alternative to standard on-line gradient descent when learning the parameters of some probabilistic model. It is based on exploiting the Riemannian structure of the likelihood manifold in order to optimize the learning dynamics. For parametric models the Riemannian metric is given by the Fisher information matrix. The idea is to re-weight the various gradient directions by pre-multiplying the standard gradient with the inverse of the Fisher information matrix, converting the covariant gradient into contravariant form.

On-line NG learning was proved to be Fisher efficient [3], implying that it has asymptotically the same performance as optimal batch parameter estimation; moreover, it was shown to provide improved transient performance in comparison with standard gradient descent (GD), with improved learning time as task complexity increases [4].

‖ E-mail address: scarpetta@na.infn.it
¶ E-mail address: magnus@cs.man.ac.uk
[+] E-mail address: D.Saad@aston.ac.uk

0305-4470/99/224047+13$19.50

NG learning has been used successfully in a variety of applications, especially in the increasingly important field of independent component analysis [5]. However, in many practical applications there will be an increased cost required in estimating and inverting the Fisher information matrix. Determining this matrix on-line is difficult as it requires an average over the input distribution in order to calculate it. Even if the Fisher information matrix can be computed on-line, inverting it will be computationally costly, especially for large networks. This is particularly undesirable as computational efficiency is one of the principal reasons for using on-line methods.

An on-line matrix momentum (MM) algorithm [6, 7] was recently introduced in order to estimate and invert the Hessian matrix efficiently on-line, obtaining asymptotically similar performance to Newton's method. We propose to use a similar method to estimate and invert the Fisher information matrix for obtaining similar performance to that of NG learning. This method is particularly efficient since the inversion is replaced by a matrix-vector multiplication which can be easily carried out. In addition, the true Fisher information matrix will, in general, be unknown and should be estimated by some method [3]; this algorithm employs a single-step estimation of the Fisher information matrix which can be determined on-line.

The algorithm is analysed within the statistical mechanics framework [4, 7–9], which is exact for large networks and enables us to examine various properties of the algorithm throughout the learning process. We formulate the problem in terms of macroscopic variables and derive a set of coupled ordinary differential equations which can then be solved numerically. This formalism is used to compare the efficiency of the proposed MM natural gradient algorithm (NGMM) with that of standard GD and standard NG in training two-layer networks. The new method turns out to provide a significant improvement over GD learning but with some sensitivity to parameter choice, due to the noisy Fisher information estimate.

In the asymptotic regime one can solve the dynamics analytically for the generic case of a noisy isotropic task. The dependence of the generalization error decay on the network architecture and parameter choice is then derived, providing the optimal and critical asymptotic training parameter values. For the appropriate parameter choice we find NGMM to provide the same asymptotic performance of NG learning (equalling the best batch algorithm), and a significant reduction in learning time when compared to GD.

The paper is organized as follows. The NG learning algorithm is defined in section 2. Section 3 briefly introduces the statistical mechanics formalism and section 4 is devoted to the derivation of the MM method. The NGMM algorithm is proposed in section 5. We then present numerical studied of transient regime (section 6) and rigorous analysis of the asymptotic phase (sections 7 and 8). Our conclusions are described in section 9.

2. Natural gradient

Consider a mapping from an input space $\boldsymbol{\xi} \in \mathbb{R}^N$ onto a scalar $\phi_{\boldsymbol{J}}(\boldsymbol{\xi}) = \sum_{i=1}^{K} g(\boldsymbol{J}_i^{\mathrm{T}}\boldsymbol{\xi})$, which defines a soft committee machine (termed the *student* network), where $\boldsymbol{J} \equiv \{\boldsymbol{J}_i\}_{1\leqslant i\leqslant K}$ is the set of input to hidden weights and the hidden to output weights are set to one. We choose $g(x) \equiv \mathrm{erf}(x/\sqrt{2})$ to be the activation function of the hidden units. We can then define a Gaussian noise model for output ζ_{m} given input $\boldsymbol{\xi}$ which is parametrized by $\boldsymbol{J}$,

$$p_{\boldsymbol{J}}(\zeta_{\mathrm{m}}|\boldsymbol{\xi}) = \frac{1}{\sqrt{2\pi\sigma_{\mathrm{m}}^2}} \exp\left(\frac{-(\zeta_{\mathrm{m}} - \phi_{\boldsymbol{J}}(\boldsymbol{\xi}))^2}{2\sigma_{\mathrm{m}}^2}\right). \tag{1}$$

Let $(\boldsymbol{\xi}^\mu, \zeta^\mu)$ be the μth input–output pair in a sequence of training examples. The activation of the student hidden node i under presentation of the input pattern $\boldsymbol{\xi}^\mu$ is denoted $x_i^\mu = \boldsymbol{J}_i^{\mathrm{T}}\boldsymbol{\xi}^\mu$. The training error at each learning iteration is taken to be proportional to the log-likelihood of

the current example under our noise model, $\epsilon_{\boldsymbol{J}}(\zeta^\mu, \boldsymbol{\xi}^\mu) \equiv \frac{1}{2}(\zeta^\mu - \sum_{i=1}^{K} g(x_i^\mu))^2$ and the most basic learning algorithm is to adapt the student weights in the negative gradient direction of this error at each iteration. But, since the probability manifold is not Euclidean, the ordinary gradient does not give the steepest direction of the error function. A more principled learning algorithm can be defined by viewing the manifold of models as a Riemannian space in which local distance is defined by the KL-divergence. The Fisher information matrix, $\boldsymbol{G}$, defines the appropriate metric in this space [10],

$$\boldsymbol{G} = \langle (\nabla_{\boldsymbol{J}} \log p_{\boldsymbol{J}}(\zeta_{\mathrm{m}}|\boldsymbol{\xi}))(\nabla_{\boldsymbol{J}} \log p_{\boldsymbol{J}}(\zeta_{\mathrm{m}}|\boldsymbol{\xi}))^{\mathrm{T}} \rangle \tag{2}$$

where the brackets denote an average over ζ_{m}, according to equation (1), followed by an average over the input distribution. Let $\boldsymbol{G}_{ik}$ be a block (i, k) of the Fisher information, which for our network is:

$$\boldsymbol{G}_{ik} = \frac{1}{\sigma_{\mathrm{m}}^2} \langle \boldsymbol{A}_{ik}(\boldsymbol{\xi}) \rangle_{\{\xi\}} \qquad \text{where} \quad \boldsymbol{A}_{ik}(\boldsymbol{\xi}) = g'(x_i^\mu) g'(x_k^\mu) \boldsymbol{\xi}\boldsymbol{\xi}^{\mathrm{T}} \tag{3}$$

where the prime indicate the derivative with respect to the argument. The natural gradient direction is found by pre-multiplying the training error gradient by the inverse of this matrix. When components of inputs $\boldsymbol{\xi}^\mu$ are selected independently at each iteration from a zero-mean Gaussian distribution with unit variance, the matrix $\boldsymbol{G}$ can be computed analytically, and for our particular choice of the activation function it turns out [4] to be:

$$\langle \boldsymbol{A}_{ik}(\boldsymbol{\xi}) \rangle_{\{\xi\}} = \frac{2}{\pi \sqrt{\Delta_{ik}}} \left[\boldsymbol{I} - \frac{(1 + Q_{kk})\boldsymbol{J}_i \boldsymbol{J}_i^{\mathrm{T}} + (1 + Q_{ii})\boldsymbol{J}_k \boldsymbol{J}_k^{\mathrm{T}} - Q_{ik}(\boldsymbol{J}_i \boldsymbol{J}_k^{\mathrm{T}} + \boldsymbol{J}_k \boldsymbol{J}_i^{\mathrm{T}})}{\Delta_{ik}} \right] \tag{4}$$

with $Q_{ik} \equiv \boldsymbol{J}_i^{\mathrm{T}} \boldsymbol{J}_k$ and $\Delta_{ik} = (1 + Q_{ii})(1 + Q_{kk}) - Q_{ik}^2$.

When the input distribution is unknown, we should estimate the average $\langle \boldsymbol{A}_{ik}(\boldsymbol{\xi}) \rangle_{\{\xi\}}$ on the basis of an empirically estimated distribution. However, it is difficult to implement the NG method as an on-line algorithm in this way. In some cases it will be possible to estimate the input distribution on-line, and Yang and Amari [10] discuss methods of preprocessing training examples to obtain a whitened Gaussian process for the inputs in this case. For $N \gg K$ this gives efficient inversion, but the method will not be applicable in general. In order to define a practical on-line algorithm with an unknown input distribution which is far from Gaussian, we need some approximation to the Fisher information matrix which can be determined on-line. The simplest approximation is to use the single training example estimation discussed in section 5.

3. General framework

We use a statistical mechanics description of the learning process [8] which is exact in the limit of large input dimension N and provides an accurate model of mean behaviour for realistic values of N.

Training examples are of the form $(\boldsymbol{\xi}^\mu, \zeta^\mu)$, as introduced in the previous section, where $\mu = 1, 2, \ldots$ labels each independently drawn example in a sequence. We consider a case where the output ζ^μ is provided by a teacher which may be corrupted by Gaussian output noise and is of a similar configuration to the student except for a possible difference in the number, M, of hidden units: $\zeta^\mu = \sum_{n=1}^{M} g(\boldsymbol{B}_n^{\mathrm{T}} \boldsymbol{\xi}^\mu) + \rho^\mu$, where $\boldsymbol{B} \equiv \{\boldsymbol{B}_n\}_{1 \leqslant n \leqslant M}$ is the set of input to hidden adaptive weights for teacher hidden nodes and ρ^μ is zero-mean Gaussian noise of variance σ^2. Due to the flexibility of this teacher mapping we can represent a variety of learning scenarios within this theoretical framework [1]. The activation of hidden node n in the teacher network under presentation of the input pattern $\boldsymbol{\xi}^\mu$ is denoted $y_n^\mu = \boldsymbol{B}_n^{\mathrm{T}} \boldsymbol{\xi}^\mu$. We will use indices $i, j, k, l = 1 \ldots K$ to refer to units in the student network and $n, m = 1 \ldots M$ for units

in the teacher network. For on-line standard GD the learning rule is: $\boldsymbol{J}_i^{\mu+1} = \boldsymbol{J}_i^{\mu} + \eta/N\delta_i^{\mu}\boldsymbol{\xi}^{\mu}$, where $\delta_i^{\mu} \equiv g'(x_i^{\mu})\big[\sum_{n=1}^{M} g(y_n^{\mu}) - \sum_{j=1}^{K} g(x_j^{\mu}) + \rho^{\mu}\big]$ and the learning rate η has been scaled with the input size N. In the NG algorithm the weight update at each iteration is given by:

$$\boldsymbol{J}_i^{\mu+1} = \boldsymbol{J}_i^{\mu} + \frac{\eta}{N}\sum_{j=1}^{K}\delta_j^{\mu}[\langle \boldsymbol{A}(\boldsymbol{\xi})\rangle_{\{\xi\}}^{-1}]_{ij}\boldsymbol{\xi}^{\mu}. \tag{5}$$

Notice that knowledge of the teacher noise variance is not required to execute this algorithm. Assuming a Gaussian input distribution it is then possible to derive equations of motion, for both gradient descent [8] and natural gradient learning [4], for a set of order parameters $\langle x_i x_j\rangle = \boldsymbol{J}_i^{\mathrm{T}}\boldsymbol{J}_k \equiv Q_{ik}$, $\langle x_i y_n\rangle = \boldsymbol{J}_i^{\mathrm{T}}\boldsymbol{B}_n \equiv R_{in}$, and $\langle y_n y_m\rangle = \boldsymbol{B}_n^{\mathrm{T}}\boldsymbol{B}_m \equiv T_{nm}$, measuring overlaps between student and teacher vectors. These order parameters are necessary and sufficient to determine the generalization error $\epsilon_g = \langle \epsilon_J(\zeta^{\mu}, \boldsymbol{\xi}^{\mu})\rangle_{\{\xi\}}$. The equations of motion are in the form of coupled first-order differential equations for the order parameters with respect to the normalized number of examples $\alpha = \mu/N$ and can be integrated numerically to determine the evolution of the generalization error. In the following sections we will show how the inclusion of an extra set of order parameters allows us to write a closed set of equations of motion for NGMM learning.

4. On-line MM

A heuristic which is sometimes useful in batch learning is to include a momentum term in the basic GD algorithm. For on-line GD learning with momentum we have

$$\boldsymbol{J}_i^{\mu+1} = \boldsymbol{J}_i^{\mu} + \frac{\eta}{N}\delta_i^{\mu}\boldsymbol{\xi}^{\mu} + \beta(\boldsymbol{J}_i^{\mu} - \boldsymbol{J}_i^{\mu-1}). \tag{6}$$

This equation defines a second-order process, in which weights from the two previous iterations are required for each update. An equivalent first-order process can be defined by introducing [11] a new set of variables $\boldsymbol{\Omega}_i^{\mu} = N(\boldsymbol{J}_i^{\mu} - \boldsymbol{J}_i^{\mu-1})$,

$$\boldsymbol{J}_i^{\mu+1} = \boldsymbol{J}_i^{\mu} + \frac{\eta}{N}\delta_i^{\mu}\boldsymbol{\xi}^{\mu} + \frac{1}{N}\beta\boldsymbol{\Omega}_i^{\mu} \qquad \boldsymbol{\Omega}_i^{\mu+1} = \beta\boldsymbol{\Omega}_i^{\mu} + \eta\delta_i^{\mu}\boldsymbol{\xi}^{\mu}. \tag{7}$$

An interesting behaviour is observed if we choose $\eta \sim \mathrm{O}(1/N)$ and $(1-\beta) \sim \mathrm{O}(1/N)$ [11]. If we define $\eta = \tilde{\eta}/N$ and $\beta = 1 - \gamma/N$, then taking $\gamma \to \infty$ and $\tilde{\eta} \to \infty$ simultaneously while keeping their ratio finite we obtain a dynamics equivalent to GD with an effective learning rate of $\eta_{\mathrm{eff}} = \tilde{\eta}/\gamma$. In fact, we find that γ does not have to be very large before this behaviour is observed.

In figure 1(*a*) we compare the standard gradient descent with gradient descent with momentum term, for a two-node network learning from noiseless examples generated by a two-node teacher network with orthonormal vectors ($T_{nm} = \delta_{nm}$). The dashed curves show results of learning with momentum term $\eta_{\mathrm{eff}} = \tilde{\eta}/\gamma = 1.666$ and $\gamma = 0.6, 1, 3, 7$ from right to left (the last dashed curve is almost obscured by the solid curve). As γ increases, the trajectory converges onto standard GD with learning rate $\eta = 1.666$ (solid curve).

Another interesting limit is $\tilde{\eta} \to 0$ and γ finite or $\gamma \to 0$ while keeping the ratio $r = \tilde{\eta}/\gamma^2 \ll \gamma$ (as may occur in some annealing schedules). In this case, consistently with the analysis in [12], the evolution of the momentum variables $\boldsymbol{\Omega}$ takes place on a much faster timescale then the evolution of the weights $\boldsymbol{J}$. Using adiabatic elimination, it can be shown that the dynamics is again equivalent to standard GD with an effective learning rate $\eta_{\mathrm{eff}} = \tilde{\eta}/\gamma$.

This effect is demonstrated in figure 1(*b*), showing the asymptotic performance of gradient descent with momentum when $\tilde{\eta}$ is annealed as α^{-1}, for a two-node network learning a noisy ($\sigma^2 = 0.01$) isotropic task of similar configuration. We use GD learning initially

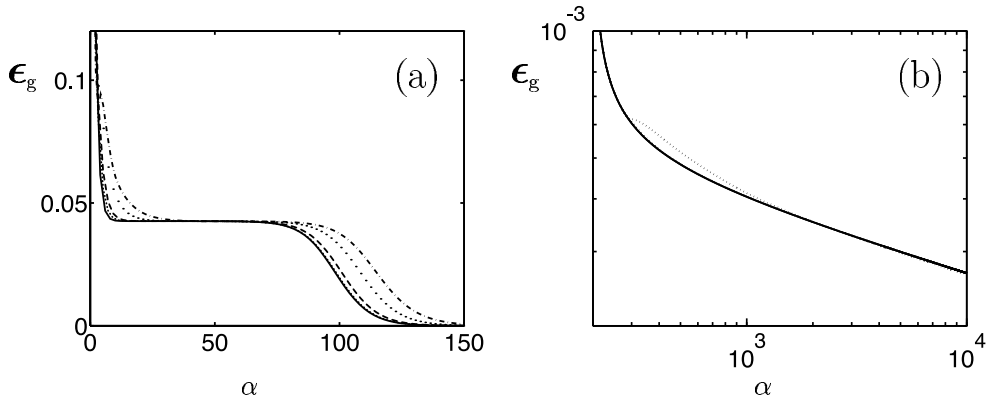

Figure 1. (*a*) The solid curve shows the generalization error for standard GD at $\eta = 1.66$, for a two-node network learning a noiseless isotropic task ($T_{nm} = \delta_{nm}$). We compare it with results for gradient descent with a momentum term, with the coefficients $\tilde{\eta} = 1.666\gamma$ and $\gamma = 0.6$ (dot-dashed curve), $\gamma = 1$ (dotted curve), $\gamma = 3$ (dashed curve), $\gamma = 7$ (double-dotted curve), the last line is almost obscured by the solid curve. (*b*) Asymptotic performance of GD with momentum γ when the learning rate $\tilde{\eta}$ is annealed as $1/\alpha$ is compared with standard GD with a learning rate $\eta = \tilde{\eta}/\gamma$, for a two-node network learning and noisy examples ($\sigma^2 = 0.01$) generated by an isotropic teacher of similar architecture. All curves were initially obtained by using standard GD until the asymptotic regime is reached. The momentum term was activated at $\alpha = 280$, with different values of $\gamma = 0.01, 0.2, 0.5, 2, 5$, and annealed learning rate $\tilde{\eta} = \gamma\ 1.666\ /(\alpha - 180)$. All curves, except the one with $\gamma = 0.01$ (dotted curve), collapse onto the GD asymptotic decay with learning rate $\eta = 1.666/(\alpha - 180)$ (solid curve).

and then, at $\alpha = 280$, we incorporate a momentum term and anneal the learning rate $\tilde{\eta} = [\gamma\ 1.666/(\alpha - 180)]$, using different values of $\gamma = 0.01, 0.2, 0.5, 2, 5$. We find that all curves collapse on the GD asymptotic decay with learning rate $\eta = 1.666$, with the exception of $\gamma = 0.01$ (dotted curve) which is not large enough with respect to the parameter $r = \tilde{\eta}/\gamma^2$.

Choosing a MM parameter of the form

$$\boldsymbol{\beta} = \boldsymbol{I} - \frac{k\boldsymbol{A}}{N} \qquad \text{with} \quad \eta = \frac{k\eta_\alpha}{N} \tag{8}$$

one expects the effective *matrix* learning rate $\eta_{\text{eff}} = \eta_\alpha \boldsymbol{A}^{-1}$, when k is large or with vanishing η_α, in correspondence to the phenomena observed for scalar momentum.

An analysis of the dynamics in the case where $\boldsymbol{A}$ is the Hessian matrix was carried out in [7], showing that matrix momentum provides an efficient inversion of the Hessian matrix. As k increase ($k = 2$ is sufficient), the trajectory converge onto the on-line Newton's method result, as desired. Similarly, setting $\boldsymbol{A}$ proportional to the Fisher information matrix and making k large or η_α very small, we expect to retrieve NG learning.

In the following section we derive a set of coupled differential equations which describe the learning process dynamics when $\boldsymbol{A}$ is proportional to a single-step estimation of the Fisher information matrix $\boldsymbol{A} = [\boldsymbol{A}_{ij}(\boldsymbol{\xi})]$.

5. Natural gradient MM

NG learning provides improved performance over standard GD during both transient [4] and asymptotic stages of learning [3]. However, in practical applications there will be an increased cost required in estimating and inverting the Fisher information matrix.

In order to define a practical on-line algorithm in the case when the input distribution is unknown, we need some approximation of the Fisher information matrix which can be determined on-line. The simplest approximation is to use a single training example estimate, i.e., we no longer average the expressions in equations (2) and (3). To achieve an efficient matrix inversion, we propose to use the MM algorithm using the Fisher information single-step estimate:

$$\boldsymbol{J}_i^{\mu+1} = \boldsymbol{J}_i^{\mu} + \frac{\eta}{N}\delta_i^{\mu}\boldsymbol{\xi}^{\mu} + \frac{1}{N}[\boldsymbol{\beta}\boldsymbol{\Omega}]_i^{\mu} \qquad \boldsymbol{\Omega}_i^{\mu+1} = [\boldsymbol{\beta}\boldsymbol{\Omega}]_i^{\mu} + \eta\,\delta_i^{\mu}\boldsymbol{\xi}^{\mu}$$
$$\text{where} \quad \boldsymbol{\beta} = \boldsymbol{I} - \frac{k\boldsymbol{A}}{N} \qquad \eta = \frac{k\eta_\alpha}{N} \qquad \boldsymbol{A} = [\boldsymbol{A}_{ik}(\boldsymbol{\xi})]$$
$$\text{and} \qquad \boldsymbol{A}_{ik}(\boldsymbol{\xi}) = g'(x_i^{\mu})g'(x_k^{\mu})\boldsymbol{\xi}\boldsymbol{\xi}^{\mathrm{T}}. \tag{9}$$

To solve the dynamics we define, as in [11], a new set of Gaussian fields related to the momentum variable $\boldsymbol{\Omega}_i$: $z_i^{\mu} = \boldsymbol{\Omega}_i^{\mathrm{T}}\boldsymbol{\xi}^{\mu}$, and a new set of order parameters relating the momentum variables: $\langle z_i z_k\rangle = \boldsymbol{\Omega}_i^{\mathrm{T}}\boldsymbol{\Omega}_k \equiv C_{ik}$, $\langle z_i y_n\rangle = \boldsymbol{\Omega}_i^{\mathrm{T}}\boldsymbol{B}_n \equiv D_{in}$, and $\langle x_i z_k\rangle = \boldsymbol{J}_i^{\mathrm{T}}\boldsymbol{\Omega}_k \equiv E_{ik}$.

In the large N limit the order parameters $\{Q, R, C, D, E\}$ provide a closed set of coupled differential equations which determine the dynamics

$$\frac{\mathrm{d}Q_{ik}}{\mathrm{d}\alpha} = E_{ik} + E_{ki} \qquad \frac{\mathrm{d}R_{in}}{\mathrm{d}\alpha} = D_{in}$$
$$\frac{\mathrm{d}C_{ik}}{\mathrm{d}\alpha} = k\langle(\eta_\alpha\delta_i - \phi_i)z_k + (\eta_\alpha\delta_k - \phi_k)z_i\rangle + k^2\langle(\eta_\alpha\delta_i - \phi_i)(\eta_\alpha\delta_k - \phi_k)\rangle \tag{10}$$
$$\frac{\mathrm{d}D_{in}}{\mathrm{d}\alpha} = k\langle(\eta_\alpha\delta_i - \phi_i)y_n\rangle \qquad \frac{\mathrm{d}E_{ik}}{d\alpha} = C_{ik} + k\langle(\eta_\alpha\delta_k - \phi_k)x_i\rangle.$$

Here, we have defined $\phi_i = g'(x_i)\sum_j z_j g'(x_j)$. The angled brackets denote averages over inputs, or equivalently averages over the field variables $\{x_i, y_n, z_k\}$ which can be carried out explicitly to provide the equations of motion (see appendix A). The generalization error can be calculated straightforwardly within this framework [8] and depends only on the order parameters Q_{ik} and R_{in}.

Note that the stochasticity comes in through the term proportional to $k^2\eta^2$ in equation (10) (for $\mathrm{d}C_{ik}/\mathrm{d}\alpha$) which is proportional to the input variance and contains an additive contribution proportional to the output noise variance.

The differential equations (10) can be integrated numerically for any number of K student and M teacher hidden units. However, for the remainder of the paper, we will focus on the realizable case ($K = M$) and uncorrelated isotropic teachers $T_{nm} = \delta_{nm}$.

For random initialization of the weights (and setting the initial momentum to zero) the resulting norms Q_{ii} of the student vector will be order O(1), while the overlaps $Q_{i\neq j}$ between different student vectors and between student and teacher vector R_{in} will be only $\mathrm{O}(1/\sqrt{N})$. So, we initialize Q_{ii} from uniform distributions in the $[0, 0.5]$ interval, $Q_{i\neq k}$ and R_{in} from $[0, 10^{-3}]$, and $C_{ik} = D_{in} = E_{ik} = 0$.

In figure 2 we show the evolution of the order parameters and the generalization error for $\eta_\alpha = 0.05$ and $k = 1.0$, for a two-node network learning a two-node isotropic task ($T_{nm} = \delta_{nm}$) in the absence of noise. As for standard GD and NG learning, the NGMM dynamics is characterized by two major phases of learning. Initially, the order parameters are trapped in an unstable fixed point characterized by a finite generalization error and a lack of differentiation between the hidden nodes of the student. The overlaps of each student node with all teacher nodes R_{in} are nearly identical, i.e., each student unit imitates all teacher nodes with similar success. Likewise, all the order parameters D_{in} are almost identical. All the student overlaps $Q_{i\neq k}$ have nearly the same value which does not differ much from the

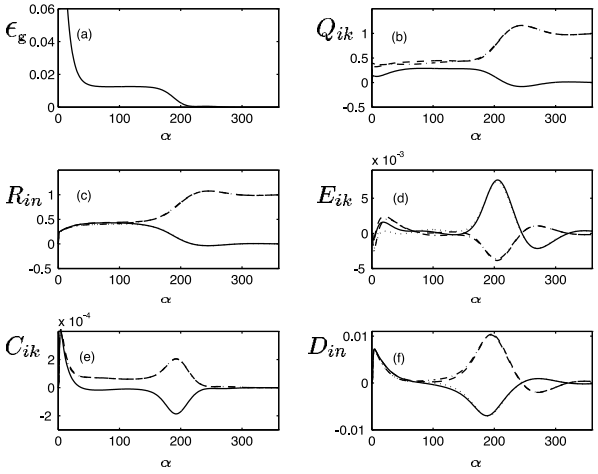

Figure 2. NGMM learning for a two-node network learning from examples generated by a two-node isotropic teacher ($T_{nm} = \delta_{nm}$) in the absence of noise. The generalization error is shown in (*a*), the student–student Q_{ik} and student–teacher R_{in} overlaps in (*b*) and (*c*), respectively, while the momentum–student E_{ik}, momentum–momentum C_{ik}, momentum–teacher D_{in} overlaps in (*d*)–(*f*) respectively. The learning rate was fixed at $\eta_\alpha = 0.05$, $k = 1$, and initial conditions were $Q_{i\neq k}$, $R_{in} = U[0, 10^{-3}]$, $Q_{ii} = U[0, 0.5]$, $C_{ik} = D_{in} = E_{ik} = 0$. A brief stage of GD is used before NGMM was activated.

value of the norms Q_{ii}, i.e. the student vectors are highly correlated with each other. Similar considerations hold for order parameters C_{ik} and E_{ik}.

Eventually, small perturbations introduced by the random initial conditions lead to an escape from this phase and convergence towards zero generalization error. The convergence phase is characterized by a specialization of each student node to a particular teacher node, which corresponds to an evolution of the order parameters Q_{ij} and R_{in} to their optimal value T_{nm}, and the remaining order parameters D_{in}, E_{ik}, C_{ik} to zero. In the following two sections we consider each phase in turn.

6. The symmetric phase

It has recently been shown that trapping time in the symmetric phase is significantly reduced by using natural gradient descent, in comparison with standard gradient descent, and exhibits a slower low power increase as task complexity grows [4].

In figure 3(*a*) we compare the performance of on-line NGMM with that of NG learning in which the averaged Fisher matrix has been explicitly inverted [4], for the noiseless isotropic case of $K = M = 2$. Ideally, we would wish for the curves to approach the true NG result (solid curve) for large k values. However, as k increases, fluctuations in the single-step Fisher information matrix estimate (due to input randomness) become significant and the weight vector norms diverge, followed by the other order parameters $\{Q, R, C, D, E\}$ and the generalization error (dashed curve, $k = 2.0$).

We see that choosing k too small ($k = 0.5$, dotted curve) may lead to long transient times, however, as we approach intermediate values ($k = 1.4$, dash-dotted curve) we obtain good performance that provides an improvement over GD learning, especially in noisy and overealizable tasks.

As the specific choice of parameters may have bearing on these results we compared the

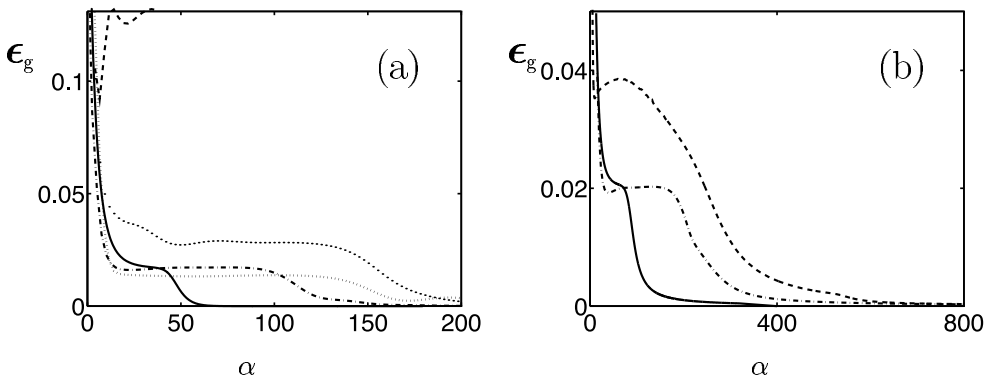

Figure 3. (*a*) The generalization error achieved by MM with single pattern estimates of the Fisher information matrix and various training parameters: $\eta_{\rm eff} = 0.15$, and $k = 0.5$ (double-dotted curve), $k = 1.4$ (dot-dashed curve), $k = 1.8$ (dotted curve), $k = 2.0$ (dashed curve), is compared with that of natural gradient descent (solid curve) in the same training scenario ($\eta = 0.15$, $K = M = 2$ and noiseless data). We see that the NGMM transient performance is worse than that of NG although it can be improved by the appropriate parameter choice. (*b*) We compare the result of NGMM (dash-dotted curve), $k = 1$ and $\eta_{\rm eff} = 0.08$ annealed from the end of the symmetric phase, $K = M = 2$ and Gaussian output noise variance ($\sigma^2 = 0.1$) with those obtained for optimal gradient descent (w.r.t. η-dashed curve) and optimal natural gradient descent (w.r.t. η-solid curve). We use $T_{mn} = \delta_{mn}$, and initial conditions $Q_{i\neq k}$, $R_{in} = U[0, 10^{-3}]$, $Q_{ii} = U[0, 0.5]$, $C_{ik} = D_{in} = E_{ik} = 0$.

results obtained for NGMM (not optimized) with those obtained with the optimal GD and optimal NG learning, using the variational method of [9]. In figure 3(*b*) we compare the NGMM method with the single pattern estimate ($k = 1$, $\eta_{\rm eff} = 0.08$, dash-dotted curve) with optimal GD (dotted curve) and optimal NG learning (solid curve), for a noisy task ($K = M = 2$ and $\sigma^2 = 0.1$), showing a reduction in learning time over optimal GD but not equalling the performance of NG learning.

7. The convergence phase

NG learning has been shown to be asymptotically optimal in the presence of output noise and with annealed learning rate $\eta = 1/\alpha$, equalling, in performance, the best possible batch algorithm.

To examine the performance of NGMM in figure 4(*a*) we show the asymptotic performance for a two-node teacher–student learning scenario and isotropic task in the presence of noise ($\sigma^2 = 0.01$) and various values of k. The NGMM performance is compared with that of NG learning [4] (taking the form $\epsilon_g \sim K\sigma^2/2\alpha$, lower dotted curve, equalling the optimal Bayes and maximum likelihood predictors [13]), and to that of optimal GD [14] (upper dotted curve). We see that NGMM asymptotic performance lies between the the optimal bound and the optimized GD asymptotic decay. We see that the prefactor of the asymptotic decay for single pattern NGMM decreases with k, converging close to the optimal bound for small values (although it takes longer to reach the asymptotic regime in this case).

In figure 4(*b*) we show that annealing k may result in a trajectory which converges rapidly to the optimal decay, providing a significant improvement over optimal GD. The training scenario is similar to the one of figure 4(*a*) except that k is annealed asymptotically ($\alpha > \alpha_o \equiv 800$) as $k = k_o/(1 + k_o(\alpha - \alpha_o)^{0.4})$. The specific choice of power law decay will be discussed in section 8. It would be interesting to find the optimal decay schedule that will bring NGMM

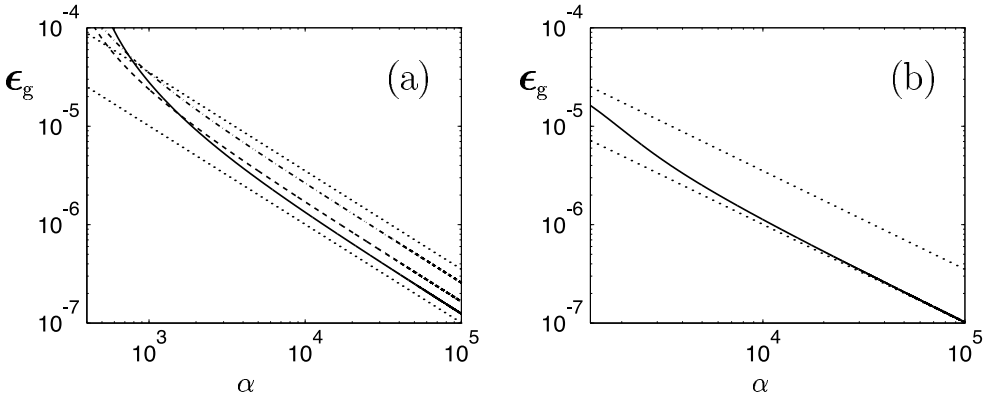

Figure 4. Generalization error decay for NGMM: the dotted curves show the optimal GD asymptotic decay (upper dotted curve), and the universal batch asymptotic bound (lower dotted curve), for a two node network learning an isotropic task $T_{mn} = \delta_{mn}$ of a similar configuration, in the presence of Gaussian output noise ($\sigma^2 = 0.01$). (*a*) Asymptotic performance of NGMM with $k = 0.3$ (solid curve), $k = 0.7$ (dashed curve) and $k = 1.4$ (dot-dashed curve). (*b*) Asymptotic decay when k is annealed from $\alpha > \alpha_o \equiv 800$ as $k = k_o/(1 + k_o(\alpha - \alpha_o)^{0.4})$.

as close as possible to the NG asymptotic performance.

8. The asymptotic dynamics—an analytical solution

To obtain an analytical solution in the asymptotic regime we now examine the asymptotic dynamics of NGMM in a noisy isotropic task with annealed learning rate $\eta_\alpha = \eta_0/\alpha$. The number of dynamical variables in equation (10) is $K(K+1) + K^2 + 2KM$, so that the analysis becomes more difficult as K and M grow. However, the symmetric architecture of the teacher network $T_{nm} = \delta_{nm}$ and the task realizability $K = M$ lead to the grouping of the dynamical variables. Hence, in the asymptotic regime the system's dynamics can be described in terms of only ten variables, via the ansatz:

$$\begin{aligned} Q_{ik} &= Q\,\delta_{ik} + \hat{Q}\,(1-\delta_{ik}) \\ R_{in} &= R\,\delta_{in} + \hat{R}\,(1-\delta_{in}) \\ E_{ik} &= E\,\delta_{ik} + \hat{E}\,(1-\delta_{ik}) \\ C_{ik} &= C\,\delta_{ik} + \hat{C}\,(1-\delta_{ik}) \\ D_{in} &= D\,\delta_{in} + \hat{D}\,(1-\delta_{in}). \end{aligned} \tag{11}$$

To solve the asymptotics of the reduced dimensional system, we expand the equations of motion (10) to first order about the asymptotic fixed point: $Q_\infty = R_\infty = 1$ and $E_\infty = C_\infty = D_\infty = \hat{E}_\infty = \hat{C}_\infty = \hat{D}_\infty = \hat{Q}_\infty = \hat{R}_\infty = 0$. Moreover, we exploit the fact that the resulting equations for the various variables evolve in different timescales ($Q, \hat{Q}, R, \hat{R}$ decay asymptotically as $1/\alpha$ while the momentum order parameters all evolve on a faster timescale, decaying as $1/\alpha^2$) to reduce the system even further.

Using adiabatic elimination for the fast variables and we find four linear coupled differential equation for the slow order parameters represented by the vector $\boldsymbol{u}$

$$\frac{\mathrm{d}}{\mathrm{d}\alpha}\boldsymbol{u} = \eta_\alpha M\boldsymbol{u} + \eta_\alpha^2\sigma^2\boldsymbol{b} \tag{12}$$

where $\boldsymbol{u} = (Q - Q_\infty, \hat{Q} - \hat{Q}_\infty, R - R_\infty, \hat{R} - \hat{R}_\infty)^{\mathrm{T}} \equiv (q, \hat{q}, r, \hat{r})^{\mathrm{T}}$, σ^2 is the noise variance, $\eta_\alpha = \eta_0/\alpha$, the vector $\boldsymbol{b}$ is a function of k and of the network size K, and $\mathcal{M}$ is

$$\mathcal{M} = \begin{bmatrix} -3 & 0 & 4 & 0 \\ 0 & -2 & 0 & 2 \\ -1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}. \tag{13}$$

The asymptotic equations of motion (12) are derived by dropping terms of $\mathrm{O}(\eta_\alpha ||\boldsymbol{u}||^2)$ and higher, and terms of $\mathrm{O}(\eta_\alpha^2 \boldsymbol{u})$. The latter are linear in the order parameters $\boldsymbol{u}$, but they are negligible in comparison to the $\eta_\alpha \boldsymbol{u}$ and $\eta_\alpha^2 \sigma^2 \boldsymbol{b}$ terms in equation (12) as $\alpha \to \infty$. These equations can be exactly solved following [14].

If λ_i are the eigenvalues of the matrix $\mathcal{M}$, and D is the matrix of the eigenvectors, such that

$$D^{-1}\, \mathcal{M}\, D = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ & \dots & \dots & \\ 0 & 0 & 0 & \lambda_4 \end{bmatrix} \tag{14}$$

then the solution of equation (12) is:

$$\boldsymbol{u}(\alpha) = D\, L(\alpha, \alpha_0)\, D^{-1}\, \boldsymbol{u}(\alpha_0) + \sigma^2\, D\, \Theta(\alpha, \alpha_0)\, D^{-1}\, \boldsymbol{b} \tag{15}$$

where $L(\alpha, \alpha_0)$ and $\Theta(\alpha, \alpha_0)$ are diagonal matrices, whose elements take the form:

$$L_{ii}(\alpha, \alpha_0) = \left(\frac{\alpha}{\alpha_0}\right)^{\lambda_i \eta_0} \qquad \text{and} \qquad \Theta_{ii}(\alpha, \alpha_0) = \frac{-\eta_0^2}{1 + \lambda_i \eta_0}[\alpha^{-1} - \alpha^{\lambda_i \eta_0} \alpha_0^{-1-\lambda_i \eta_0}]. \tag{16}$$

Since the first contribution in equation (15) depends on the actual initial conditions $\boldsymbol{u}(\alpha_0)$, and since we are interested in the asymptotic regime, we will ignore it in what follows (it decays more rapidly than the second contribution).

As we are mainly interested in the generalization error decay rate we expand the generalization error to the second order in $\boldsymbol{u}$:

$$\epsilon_g^{asy} = \frac{K}{\pi}\left[\frac{2}{\sqrt{3}}\left(\frac{1}{2}q - r\right) + (K-1)\left(\frac{1}{2}\hat{q} - \hat{r}\right) - \frac{(2r-q)^2}{12\sqrt{3}} + \frac{q(r-q)}{3\sqrt{3}} + \frac{(K-1)}{4} q(\hat{q} - \hat{r})\right]. \tag{17}$$

Using the solution equation (15) the generalization error can be rewritten as a combination of the modes Θ_{ii}, whose coefficients are functions of the system size K and of the parameter k. We find that only two modes, Θ_{22} and Θ_{44}, associated with the eigenvalues $\lambda_2 = \lambda_4 = -2$, survive in the linear terms of the generalization error. The modes Θ_{11} and Θ_{33} associated with the eigenvalues $\lambda_1 = \lambda_3 = -1$ are orthogonal to the first-order terms in the generalization error, and therefore do not contribute to their decay, but contribute only the decay of the second-order term with the corresponding eigenvalues $2\lambda_1$ and $2\lambda_3$.

The critical learning rate η_{crit}, above which the generalization ϵ_g^{asy} decays as $1/\alpha$ is therefore:

$$\eta_{crit} = \max\left\{-\frac{1}{2\lambda_1}, -\frac{1}{\lambda_2}\right\} = \frac{1}{2}.$$

For $\eta_0 > \eta_{crit}$ we can ignore the second-order terms in the generalization error (they decay as $1/\alpha^2$), finding an asymptotic error of the form:

$$\epsilon_g^{asy} = \sigma^2 f_0(k, K) \frac{\eta_0^2}{(-\lambda_2 \eta_0 - 1)} \alpha^{-1} = \sigma^2 f_0(k, K) f_1(\eta_0) \frac{1}{\alpha}. \tag{18}$$

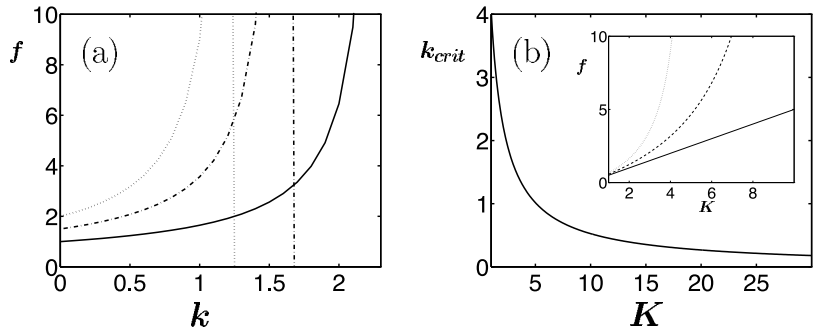

Figure 5. (*a*) The prefactor of the asymptotic generalization error decay, as a function of k, at $\eta_0 = \eta_{opt} = 1$, with $K = M = 2$ (solid curve), $K = M = 3$ (dot-dashed curve) and $K = M = 4$ (dotted curve). The prefactor diverge at $k = k_{crit}(K)$. (*b*) k_{crit} as a function of K. In the inset we show the prefactor of the asymptotic decay as a function of K when $k = 0$ (straight line), $k = 0.5$ (dashed curve) and $k = 1$ (dotted curve).

For optimal decay of the asymptotic error we minimize the prefactor of $1/\alpha$ in equation (18). Since the prefactor can be factorized we can separately optimize the learning rate η_0 and the momentum parameter k. By minimizing $f_1(\eta_0)$ we find that the optimal value of η_0 is always $\eta_0^{opt} = 1$, independently from the value of K and k; this is the same annealing schedule $\eta_\alpha = 1/\alpha$ that is optimal for NG learning [3]. In contrast, we stress that the optimal learning rate for GD is also inversely proportional to α, but the optimal prefactor α_0 depends on K in a nontrivial manner [14].

The sensitivity of the generalization error decay prefactor to the choice of k is shown in figure 5(*a*) for $K = 2, 3, 4$. We find that the prefactor is minimal at $k = 0$, independently of the value of K. The prefactor increases as k increases and diverges at $k = k_{crit}(K)$ (see figure 5(*b*)). For $k > k_{crit}$ the asymptotic regime is never reached. The asymptotic result for $\eta_0 = 1$ in the limit $k \to 0$ takes the very simple form $\epsilon_g \sim K\sigma^2/2\alpha$, shown as a straight line in the inset of figure 5(*b*), equalling the NG result and the optimal batch bound.

The picture that emerges is that k must be quite large in the transient, but should be annealed asymptotically such that $k \to 0$ when $\alpha \to \infty$. One might expect that the k annealing cannot be faster or equal to $\alpha^{-1/2}$, since one expects the ratio $r = (k\eta_\alpha)/(k)^2$ to be small with respect to k (see section 4) to obtain an effective matrix inversion from this method. Indeed, we see from numerical experiments that annealing k as $1/\alpha^x$ with $x > \frac{1}{2}$ leads to uncontrolled behaviour (the generalization error starts to increase and the order parameters diverge) while annealing schedules of the form $k \sim 1/\alpha^x$ with $x \leqslant \frac{1}{2}$ lead to very good performance and a rapid convergence of the generalization error to the optimal bound (figure 4(*b*)).

The influence of the noise variance on the generalization error can be seen directly from equation (18): the noise variance is just a prefactor in the $1/\alpha$ decay and neither the critical nor optimal values for η_0 and k are influenced by it.

9. Conclusions

The NG learning algorithm is efficient and provides a significant improvement over conventional on-line training methods; however, its complexity is generally high due to the computation required for estimating and inverting the Fisher information matrix. These can be achieved efficiently for Gaussian, or near-Gaussian input distributions [10] but may be difficult to carry out in practice, where the input distribution is significantly different.

We exploited the statistical mechanics based theoretical framework to study the efficiency of a new learning algorithm. A single pattern estimation of the Fisher information matrix is employed and the MM method is used to achieve an effective matrix inversion. The proposed algorithm provides a practical and computationally cheap approximation to NG learning.

Analytical results indicate that this approximation is efficient, especially in the asymptotic regime, but shows some sensitivity to the choice of parameters in the transient. However, if the appropriate parameters η and k are chosen in the transient, this algorithm provides a significant reduction in the length of the symmetric phase over GD learning, although not equalling the performance of the NG learning. In standard noisy tasks it provides the optimal asymptotic performance when the learning rate is $\eta_\alpha = 1/\alpha$ and k is annealed no faster than $1/\alpha^{1/2}$, equalling in performance NG learning and the best batch algorithm.

We believe that the sensitivity to the choice of k is due to noise in the Fisher information matrix estimate. It will be useful to consider more sophisticated on-line approximations to the Fisher information matrix, which may provide greater robustness to the choice of parameters. The present formalism provides a useful theoretical framework in which new approximations may be considered.

Acknowledgments

Support from EPSRC grant GR/L19232 is gratefully acknowledged. MR was further supported by EPSRC grant GR/M48123.

Appendix A. NGMM dynamical equations

The differential equations for the evolution of the order parameters $Q_{ik}, R_{in}, C_{ik}, D_{in}, E_{ik}$ are calculated by explicitly carrying out the averages that appear in equation (10) over the activation fields $\{\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}\}$, that are taken from a multivariate Gaussian distributed with zero mean and covariance matrix $\mathcal{G}$ given by

$$\mathcal{G} = \begin{bmatrix} Q & E & R \\ E^{\mathrm{T}} & C & D \\ R^{\mathrm{T}} & D^{\mathrm{T}} & T \end{bmatrix}. \tag{19}$$

Equations (10) then take the form:

$$\begin{aligned}
\frac{\mathrm{d}Q_{ik}}{\mathrm{d}\alpha} &= E_{ik} + E_{ki} \qquad \frac{\mathrm{d}R_{in}}{\mathrm{d}\alpha} = D_{in} \\
\frac{\mathrm{d}C_{ik}}{\mathrm{d}\alpha} &= 2k\eta_\alpha \Bigg[\sum_m A_3(x_i, z_k, y_m) - \sum_j A_3(x_i, z_k, x_j) \Bigg] - 2k \sum_j B_4(x_i, x_j, z_j, z_k) \\
&\quad + k^2\eta_\alpha^2 \Bigg[\sum_{mn} A_4(x_i, x_k, y_m, y_n) + \sum_{lj} A_4(x_i, x_k, x_l, x_j) \\
&\quad - 2\sum_{ml} A_4(x_i, x_k, x_l, y_m) \Bigg] + k^2 \sum_{jl} B_6(x_i, x_j, x_k, x_l, z_j, z_l) \\
&\quad - 2k^2\eta_\alpha \Bigg[\sum_{jn} B_5(x_i, x_j, x_k, z_j, y_n) - \sum_{jl} B_5(x_i, x_j, x_k, z_j, x_l) \Bigg] \\
\frac{\mathrm{d}D_{in}}{d\alpha} &= k\eta_\alpha \Bigg[\sum_m A_3(x_i, y_n, y_m) - \sum_j A_3(x_i, y_n, x_j) \Bigg] - k \sum_j B_4(x_i, x_j, z_j, y_n) \\
\frac{\mathrm{d}E_{ik}}{\mathrm{d}\alpha} &= C_{ik} + k\eta_\alpha \Bigg[\sum_m A_3(x_i, x_k, y_m) - \sum_j A_3(x_i, x_k, x_j) \Bigg] - k \sum_j B_4(x_i, x_j, z_j, x_k).
\end{aligned} \tag{20}$$

Where we have used the following notation:

$$
\begin{aligned}
A_3(u_1, u_2, u_3) &= \langle g'(u_1) u_2 g(u_3) \rangle \\
A_4(u_1, u_2, u_3, u_4) &= \langle g'(u_1) g'(u_2) g(u_3) g(u_4) \rangle \\
B_4(u_1, u_2, u_3, u_4) &= \langle g'(u_1) g'(u_2) u_3 u_4 \rangle \\
B_5(u_1, u_2, u_3, u_4, u_5) &= \langle g'(u_1) g'(u_2) g'(u_3) u_4 g(u_5) \rangle \\
B_6(u_1, u_2, u_3, u_4, u_5, u_6) &= \langle g'(u_1) g'(u_2) g'(u_3) g'(u_4) u_5 u_6) \rangle .
\end{aligned}
\tag{21}
$$

The variable u_i represent members of $\{\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}\}$ and the index d in A_d and B_d denote averages over d variables.

It is a property of multivariate Gaussian distributions [8] that integrals of reduced dimensionality like $A_3(x_1, x_1, z_2)$ are generated from the general form $A_3(u_1, u_2, u_3)$ by projection of the covariance matrix $\mathcal{G}$ onto the relevant subspace (in this case $\mathcal{G}_{u1,u2} = \mathcal{G}_{u1,u1} = \mathcal{G}_{u2,u2} = Q_{11}$, $\mathcal{G}_{u3,u3} = C_{22}$, and $\mathcal{G}_{u1,u3} = \mathcal{G}_{u2,u3} = E_{12}$). All the integrals (21) can be carried out analytically, so that equations (20) give rise to a closed set of differential equations. Additional details on the calculations and the numerical results can be found in [15].

References

[1] Cybenko G 1989 *Math. Control Signals Syst.* **2** 303
[2] Saad D (ed) 1998 *Online learning in neural networks* (London: Cambridge University Press)
[3] Amari S 1998 *Neural Comp.* **10** 251
[4] Rattray M, Saad D and Amari S 1998 *Phys. Rev. Lett.* **81** 5461
Rattray M and Saad D 1998 *Proc. Int. Conf. on Artificial Neural Networks* ed L Niklasson, M Bodòden and T Ziemke (London: Springer) p 165
[5] Amari S 1999 *IEEE Trans. Signal Process.* **47** 936
[6] Orr G B and Leen T K 1997 *Advances in Neural Information Processing Systems* vol 9, ed M C Mozer, M I Jordan and T Petsche (Cambridge, MA: MIT Press) p 606
[7] Rattray M and Saad D 1998 *Proc. Int. Conf. on Artificial Neural Networks* ed L Niklasson, M Bodòden and T Ziemke (London: Springer) p 183
[8] Saad D and Solla S A 1995 *Phys. Rev. Lett.* **74** 4337
Saad D and Solla S A 1995 *Phys. Rev.* E **52** 4225
[9] Saad D and Rattray M 1997 *Phys. Rev. Lett.* **79** 2578
Rattray M and Saad D 1998 *Phys. Rev.* E **58** 6379
[10] Yang H Y and Amari S 1997 *Advances in Neural Information Processing Systems* vol 10, ed M C Mozer, M I Jordan and T Petsche (Cambridge, MA: MIT Press) p 385
Yang H Y and Amari S 1998 *IEEE Trans. Neural Netw.* at press
[11] Prügel-Bennett A 1996 Unpublished notes
[12] Wiegerinck W, Komoda A and Heskes T 1994 *J. Phys. A: Math. Gen.* **27** 4425
[13] Amari S and Murata N 1993 *Neural Comp.* **5** 140
[14] Leen T K, Schottky B and Saad D 1998 *Advanced in Neural Information Systems* vol 10, ed M I Jordan, M J Kearns and S A Solla (Cambridge, MA: MIT Press) p 301
Leen T K, Schottky B and Saad D 1999 *Phys. Rev.* E **59** 985
[15] Scarpetta S 1998 *PhD Thesis* Salerno University, Italy