Natural Conjugate Gradient Training of Multilayer Perceptrons

Ana González and José R. Dorronsoro*

Dpto. de Ingeniería Informática and Instituto de Ingeniería del Conocimiento
Universidad Autónoma de Madrid, 28049 Madrid, Spain

Abstract. For maximum log–likelihood estimation, the Fisher matrix defines a Riemannian metric in weight space and, as shown by Amari and his coworkers, the resulting natural gradient greatly accelerates on–line multilayer perceptron (MLP) training. While its batch gradient descent counterpart also improves on standard gradient descent (as it gives a Gauss–Newton approximation to mean square error minimization), it may no longer be competitive with more advanced gradient–based function minimization procedures. In this work we shall show how to introduce natural gradients in a conjugate gradient (CG) setting, showing numerically that when applied to batch MLP learning, they lead to faster convergence to better minima than that achieved by standard euclidean CG descent. Since a drawback of full natural gradient is its larger computational cost, we also consider some cost simplifying variants and show that one of them, diagonal natural CG, also gives better minima than standard CG, with a comparable complexity.

1 Introduction

The standard approach in Multilayer Perceptron (MLP) training is to minimize the square error function

$$e(W) = \frac{1}{2}\int ||F(X,W) - Y||^2 dP(X,Y),$$

where Y denotes the target associated to a pattern X, $F(X,W)$ is the MLP transfer function and $P(X,Y)$ is the joint (X,Y) probability distribution. In practice, rather than minimizing the global error $e(W)$, one tries to do so for its sample version

$$\hat{e}(W) = \frac{1}{2N}\sum_i ||F(X_i,W) - Y_i||^2.$$

In this light MLP training can be seen as a nonlinear regression problem, but if we assume an error model $Y = F(X,W) + Z$, with Z a multivariate gaussian $g(Z)$ with density

$$g(Z) = \frac{1}{\sqrt{2\pi}\sigma}\exp^{\frac{-||Z||^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}\sigma}\exp^{\frac{-||f(X,W)-Y||^2}{2\sigma^2}}.$$

[*] With partial support of Spain's TIN 2004–07676.

S. Kollias et al. (Eds.): ICANN 2006, Part I, LNCS 4131, pp. 169–177, 2006.

we can alternatively formulate MLP training as a semiparametric maximum log likelihood estimation problem. In fact, the likelihood associated to the sample (X_i, Y_i) is

$$\prod_i g(Z_i) = \prod_i \frac{1}{\sqrt{2\pi}\sigma} \exp^{\frac{-||f(X_i,W)-Y_i||^2}{2\sigma^2}},$$

and therefore

$$-\log\left(\prod_i g(Z_i)\right) = \frac{1}{2}\sum_i ||F(X_i, W) - Y_i||^2 + C,$$

with C a suitable constant.

In this general context of likelihood estimates for parametric probability models, it has been shown by S.I. Amari [2,8] that a a Riemannian structure can be defined in weight space, for which the metric tensor is given by the matrix

$$\begin{aligned} G(W) &= E_{X,\,y}[||f - Y||^2(\nabla_W f)(\nabla_W f)^t] \\ &= \sigma^2 E_X[(\nabla_W f)(\nabla_W f)^t]. \end{aligned} \tag{1}$$

and the inner product to be used in the tangent space at a point W is $\langle u, v\rangle_W = u^t G(W) v$. It turns out [1] that the maximum descent direction of the global error $e(W)$ with respect to the $G(W)$ metric is then given by the “natural” gradient

$$\nabla_G\, e(W) = G(W)^{-1}\nabla e(W).$$

As shown by Amari and his coworkers [1,13], this can be put to advantage when on–line MLP training is considered. In fact, denoting the local error $||f(X, W) - Y||^2$ as $e(X, Y; W)$ and defining natural gradient descent as

$$W_{t+1} = W_t - \eta_t G(W_t)^{-1}\nabla e(X_t, y_t; W_t), \tag{2}$$

one obtains what probably is the fastest converging MLP on–line training method.

The main drawback of on–line natural gradient training is its complexity. For a single hidden layer MLP with input dimension D, H hidden units and C dimensional outputs, and an N pattern sample, the weight–bias dimension is then $\mathcal{D} = H(D+1)+C(H+1)$, which would imply a cost $\mathrm{O}(\mathcal{D}^3)$ for G’s inversion and an overall cost of $\mathrm{O}(\mathcal{D}^3 N)$ for each on–line full sample pass. There are ways in the on–line setting to alleviate this [3,14] and its impact is much smaller if batch natural gradient is considered. In fact, the inversion of G is done only once per batch epoch and if $N \gg \mathcal{D}$, as it happens in most settings, the main cost is then the computation of the matrix G, which is then $\mathrm{O}(N\mathcal{D}^2)$.

However, G coincides with the Gauss–Newton approximation to the Hessian of a square error function, and batch natural gradient descent can be seen [5,6] to be closely related to the Levenberg–Marquardt approach to mean square minimization. In turn, this can be used to give another explanation of the speed–up in batch MLP training with respect to standard gradient descent. But for batch MLP training there are other simpler methods such as the conjugate gradient or the variable metric methods, which also have a fast convergence without needing costly Hessian computations.

In any case, the introduction of a Riemannian structure in weight space through the Fisher metric can be done independently [12,8] of the minimization setting described above, and the resulting fast on–line convergence could also be considered as a consequence of the "naturalness" of the Fisher metric. This should also be reflected, for instance, in ways to improve on established batch minimization methods. Some of these methods rely on Hessian computations or approximations, something that may not be easy to do in a Riemannian setting. The situation should be simpler for gradient based methods. Among these, the best known is the conjugate gradient method, a good choice for instance for batch MLP training [7]. In the next section we shall briefly review conjugate gradient and show how to define a natural conjugate gradient and in section 3 we shall numerically illustrate its advantages over its standard counterpart. The paper will finish with a brief review of the paper's results and some concluding remarks.

2 Natural Conjugate Gradient

The standard conjugate gradient (CG) [10] method seeks a fast way to attain the minimum of a general function $f(W)$ by succesively performing for $i = 0, \ldots$, the following steps from an initial W_0 and $g_0 = h_0 = -\nabla\ f(W_0)$:

1. Define $g_{i+1} = -\nabla f(W_{i+1})$, where W_{i+1} is the minimum of f over the line $\{W_i + th_i : t > 0\}$;
2. Set $h_{i+1} = g_{i+1} + \gamma_{i+1} h_i$, with

$$\gamma_{i+1} = \frac{g_{i+1} \cdot g_{i+1}}{g_i \cdot g_i}.$$

The rationale for this approach comes from the fact that, for a quadratic $e(W) = c - b \cdot W + \frac{1}{2} W^t H W$, the above defined g_i, h_i verify for $j < i$

$$g_i \cdot g_j = g_i \cdot h_j = h_i^t H h_j = 0.$$

It thus follows that for such a quadratic e, a minimum W^* is achieved in at most D iterations, with D the dimension of W.

The above formulation can be easily extended when the standard gradient of the mse function $e(W)$ is replaced by its natural counterpart. More precisely, if we denote the natural gradient at W_{i+1} as $\tilde{g}_{i+1} = -G_{i+1}^{-1} \nabla e(W_{i+1})$, with G_{i+1} the natural metric at W_{i+1}, and define

$$\tilde{\gamma}_{i+1} = \frac{\langle \tilde{g}_{i+1}, \tilde{g}_{i+1} \rangle_{G_{i+1}}}{\langle \tilde{g}_i, \tilde{g}_i \rangle_{G_i}},$$

the new conjugate direction is then $\tilde{h}_{i+1} = \tilde{g}_{i+1} + \tilde{\gamma}_{i+1} \tilde{h}_i$. Under some extra assumptions, it can be shown that for the above $\tilde{g}_i$, $\tilde{h}_i$, and a quadratic $e(W)$,

$$\langle \tilde{g}_{i+1}, \tilde{g}_i \rangle_{G_{i+1}} = \langle \tilde{g}_{i+1}, \tilde{h}_i \rangle_{G_{i+1}} = \tilde{h}_{i+1}^t H \tilde{h}_i = 0. \quad (3)$$

Table 1. Training architectures used in the numerical experiments

Problem set	input dim.	hid. units	targ. dim
br. cancer	9	5	2
glass	9	6	6
heart dis.	13	7	5
ionosphere	33	3	2
iris	4	3	3
pima	7	4	2
thyroid	8	5	2
XOR4	3	10	4
abalone	7	4	1
housing	13	5	1

It easily follows from the above discussion that if we do not take into account the line minimization required to obtain the W_i, the cost of standard and natural CG is essentially that of computing the corresponding gradients. We recall that for an N pattern sample and a single hidden layer MLP with input dimension D, H hidden units and C dimensional outputs, the cost of the mse standard gradient is $\mathrm{O}(NDHC)$ per batch iteration. When natural gradient is considered and we denote the number of MLP weights as $\mathcal{D} = H(D+1) + C(H+1)$ as done before, this cost is dominated by the rather larger cost $\mathrm{O}(N\mathcal{D}^2) = \mathrm{O}(N(DH + HC)^2)$ of computing the Fisher matrix. Recall that the $\mathcal{D}^2$ term is due to the neeed to compute about $\mathcal{D}^2/2$ expectations

$$E\left[\frac{\partial e}{\partial w_{lk}}\frac{\partial e}{\partial w_{nm}}\right]. \tag{4}$$

There are several ways to lower this. We may begin by using a block–diagonal version of G, where if denote by w^O_{oh} the hidden–to–output weights and by w^H_{hi} the input–to–hidden weights, we simply assume that

$$E\left[\frac{\partial e}{\partial w^O_{oh}}\frac{\partial e}{\partial w^H_{hi}}\right] \approx 0. \tag{5}$$

The resulting cost would then be $\mathrm{O}(N(H^2D^2 + C^2H^2))$. We can further reduce the complexity assuming [5] independence between the output o_k of unit k at a given layer and the generalized error δ_l of unit l of the next layer. Since we have $\partial e(X, Y; W)/\partial w_{lk} = \delta_l o_k$ for the local gradient [4], we can therefore write

$$E\left[\frac{\partial e}{\partial w_{lk}}\frac{\partial e}{\partial w_{nm}}\right] = E\left[\delta_l o_k \delta_n o_m\right] \approx E\left[\delta_l \delta_n\right] E\left[o_k o_m\right].$$

Precomputing the matrices $E[\delta_l\delta_n]$ and $E[o_k o_m]$ for the input–to–hidden and hidden–to–output weights has a cost of $\mathrm{O}(N(C^2 + H^2))$ for the δ matrices and $\mathrm{O}(N(H^2 + D^2))$ for the o matrices. The overall cost of this "independent" natural gradient is then $\mathrm{O}(N(D^2+H^2+C^2))$, which now is dominated by the $\mathrm{O}(NDHC)$ cost of the standard

Table 2. Final mean mse values and their standard deviation for standard CG (second column), natural CG (third column), diagonal natural CG (fourth column) and line minimization natural gradient. Best final values overall when equality of means is rejected at the 5 % level are given in bold face, second place values in italics and third place values in typewriter type.

Problem set	standard CG	natural CG	diagonal NCG	line min. NCG
breastc	`0.0382 ± 0.0005`	**0.0306 ± 0.0010**	*0.0315 ± 0.0011*	0.0405 ± 0.0009
glass	`0.3499 ± 0.0107`	**0.3357 ± 0.0090**	**0.3383 ± 0.0103**	0.3997 ± 0.0112
heartdis	`0.3444 ± 0.0070`	**0.3373 ± 0.0066**	**0.3350 ± 0.0089**	0.4126 ± 0.0060
ionosphere	`0.0358 ± 0.0025`	**0.0298 ± 0.0037**	**0.0307 ± 0.0035**	0.1026 ± 0.0092
iris	`0.0453 ± 0.0012`	**0.0384 ± 0.0002**	**0.0384 ± 0.0002**	0.0504 ± 0.0028
pima	`0.2263 ± 0.0050`	**0.2189 ± 0.0058**	**0.2205 ± 0.0066**	0.2409 ± 0.0064
thyroid	`0.0488 ± 0.0007`	**0.0400 ± 0.0020**	*0.0416 ± 0.0019*	0.0492 ± 0.0020
xor405	`0.1615 ± 0.0022`	**0.1470 ± 0.0020**	**0.1477 ± 0.0018**	0.1721 ± 0.0057
abalone	0.4172 ± 0.0004	**0.4112 ± 0.0019**	*0.4140 ± 0.0017*	`0.4122 ± 0.0022`
housing	0.0789 ± 0.0029	**0.0706 ± 0.0030**	*0.0771 ± 0.0031*	`0.0735 ± 0.0025`

gradient. Finally, the simplest approach would be to consider what we may call diagonal natural gradient, where we replace the full Fisher matrix $G(W)$ by its diagonal, which results in a cost of $\mathrm{O}(N(DH + HC))$, dominated again by the cost of standard CG.

In the following section we shall compare the perfomance against standard CG of natural CG and its pure diagonal variant. Similar results are obtained in the other cases and will be published elsewhere.

3 Numerical Examples

We shall compare natural conjugate gradient MLP training against standard conjugate gradient on 10 datasets. Two of these datasets correspond to regression problems and 8 to classification problems. Nine of the datasets are taken from the UCI database [9]: we shall work with the abalone age and Boston housing regression problems, and the classification problems given by the Wisconsin breast cancer, glass, heart disease, ionosphere, iris, diabetes in Pima indians and thyroid disease datasets. In some instances the UCI repository gives separate training and test sets. Since we are interested only on square error minimization, in these cases we join both sets in a single training set.

The tenth dataset, which we denote XOR4, is a 4 class synthetic problem, an extension of bidimensional XOR to 3 dimensions, where eight 0.5 standard deviation gaussian distributions centered at the opposite corners of the unit cube are considered and four classes are defined pairing diagonally opposite distributions. That is, the gaussian centers of the first class are at $(-1, -1, -1)$ and $(1, 1, 1)$, those of the second are at $(-1, -1, 1)$ and $(1, 1, -1)$ and so on.

In all cases we have normalized input components to zero mean and one variance, and we also have done so for target values in the regression problems. Table 1 shows the training parameters used; the number of hidden units has been set heuristically, but it essentially agrees with values used in other studies.

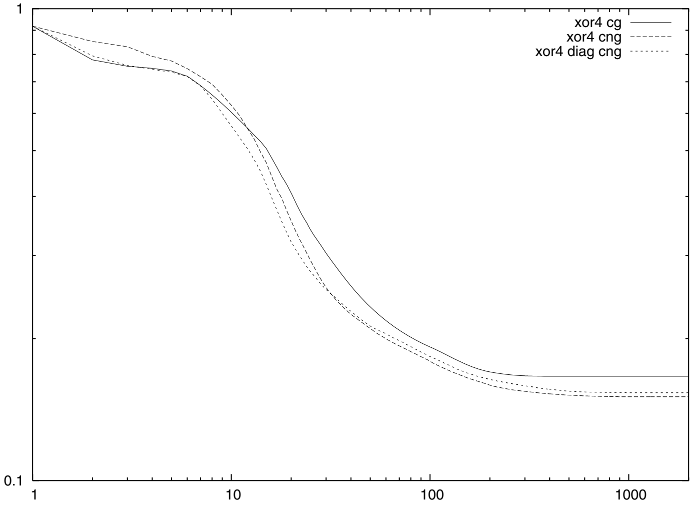

Fig. 1. Mse evolution for the XOR4 problem of standard (solid line), natural (large dash line) conjugate and diagonal natural (small dash line) conjugate gradients

We have used the Numerical Recipes implementation of standard conjugate gradient ([11], section 10.6) and adapted it for natural conjugate gradient. Instead of the Fletcher–Reeves formula for $\tilde{\gamma}_{i+1}$ given in section 2, we have used the Polak–Ribiere variant, as it seems better suited for general function minimization [10,11], namely

$$\tilde{\gamma}_{i+1} = \frac{\langle \tilde{g}_{i+1}, \tilde{g}_{i+1} - \tilde{g}_i \rangle_{G_{i+1}}}{\langle \tilde{g}_i, \tilde{g}_i \rangle_{G_i}},$$

Also, to avoid singularity problems, we invert the matrix $G + \mu I$ instead of G, with I the identity matrix and the scalar μ having an initial value of 0.05 that is decreased by a factor of 0.9 per iteration. In all cases we have run 30 independent trainings starting at different initial weights, with a maximum of 2000 gradient iterations (in many cases the Numerical Recipes implementation makes natural and standard CG descent to stop well before that limit is reached). To avoid instabilities due to training divergence, of all these, only the 20 runs with the best final mean square errors (mse) values are selected and their mean and standard deviations computed. Notice that there are more significant ways to measure MLP performance, such as computing for instance test set accuracies. However we are essentially comparing function minimization procedures, which in the MLP case means to compare final mse values.

Table 2 gives for each data set these final values for standard (second column), and full natural (third) and diagonal (fourth) CG. It also shows results for line mimization based on natural gradient (fifth column). To better compare them we have performed pairwise mean equality tests between all procedures. The table shows in bold face the smaller overall value when equality of means is rejected at the 5% confidence level. It is given in 4 cases by the natural conjugate gradient alone, and in the other 6 cases the performance of natural CG and its diagonal counterpart is similar in the sense that equality

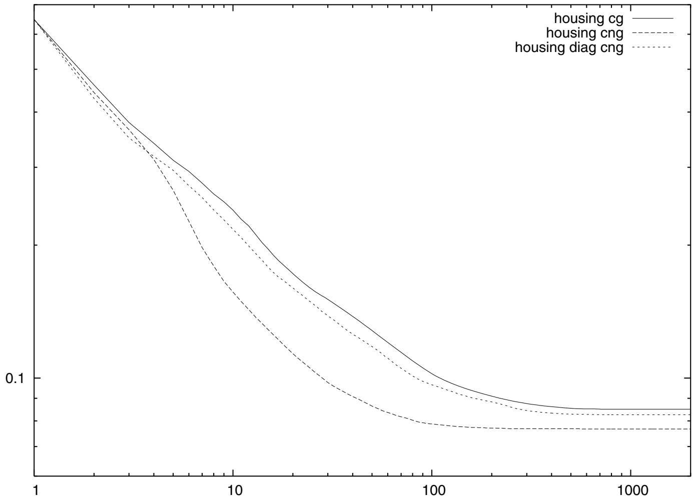

Fig. 2. Mse evolution for the housing problems of standard (solid line), natural (large dash line) conjugate and diagonal natural (small dash line) conjugate gradients

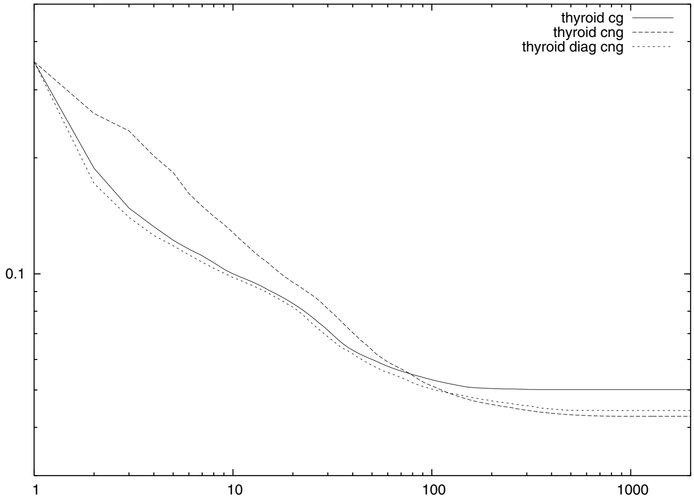

Fig. 3. Mse evolution for the thyroid problem of standard (solid line), natural (large dash line) conjugate and diagonal natural (small dash line) conjugate gradients

of means cannot be rejected. Second overall values are shown in italics and third values in typewriter type. As it can be seen from the table standard and diagonal natural CG beat standard CG in all cases. On the other hand, standard CG beats line minization based natural gradient in all problems but for the abalone and housing datasets. It can

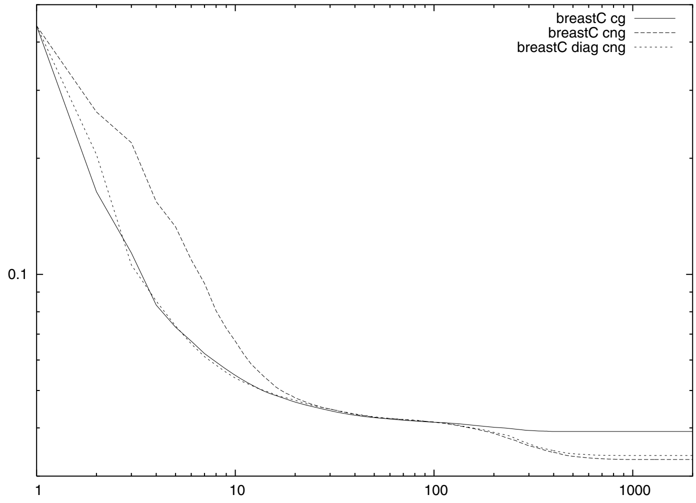

Fig. 4. Mse evolution for the breast cancer problem of standard (solid line), natural (large dash line) conjugate and diagonal natural (small dash line) conjugate gradients

be safely concluded that natural CG, either full or diagonal, yields better minima than ordinary CG for MLP training.

Besides providing better minima, natural conjugate gradient convergence can also be faster than that of ordinary conjugate gradients. This is illustrated in figures 1 and 2 for the XOR4 and housing problems, where natural CG overtakes the standard one at about the tenth iteration and does so for its diagonal variant shortly thereafter (all figures in logarithmic scale on both axes). In other cases this overtaking may happen later, but in all the datasets considered, it takes place before the 100–th iteration. This is shown, for instance, in figures 3 and 4 for the thyroid and breast cancer problems. When comparing convergence speed, one should also take into account the distinct complexity of, say, full natural CG against that of standard CG, something which we are currently studying. In any case, in all datasets diagonal natural CG does overtake standard CG at about the 10–th iteration, while both methods have essentially the same complexity.

4 Conclusions

It was shown by Rao [12] that, in a maximum log–likelihood setting, the Fisher matrix defines a Riemannian metric in weight space alternative to the standard euclidean one. Besides its theoretical advantages, Amari and his coworkers have demonstrated that for on–line MLP training, the resulting natural gradient provides minimization directions that result in a faster convergence.

If batch MLP training is considered, natural gradient descent can be seen as a variant of the Gauss–Newton method, closely related to Levenberg–Marquardt's minimization. A such it may not be competitive with other advanced batch methods, such as for instance, conjugate gradient (CG). In this paper we have shown how natural gradient can be introduced in the conjugate gradient setting and have numerically demonstrated that the performance of the resulting natural CG is consistently better than that of standard CG.

As it is the case for on–line MLP training, a drawback of natural CG is the larger complexity resulting from the required Fisher matrix computations. This can be alleviated by approximating the Fisher matrix under some simplifying assumptions, of which we have considered here the diagonal natural CG. It has essentially the same complexity of standard CG but gives better minima (although not always as good as those achieved by the full natural CG procedure) and a faster convergence.

We finally point out that there might be some interest in further research on the application of natural gradients in general function minimization. A more complete study should be made of full natural CG taking its complexity into account in a precise way. On the other hand, the definition (1) of the natural metric makes sense not only for square error problems but also for other global error functions defined as local error expectation. We are currently considering these and other similar issues.

References

1. Amari, S. (1998). Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10, 251–276.
2. Amari, S., Nagaoka, H. **Methods of information geometry**. American Mathematical Society, 2000.
3. Amari, S., Park, H., Fukumizu, K. (2000). Adaptive Method of Realizing Natural Gradient Learning for Multilayer Perceptrons. *Neural Computation*, 12, 1399–1409.
4. Duda, R., Hart, P., Stork, D. **Pattern classification**. Wiley, 2000.
5. Heskes, T. (2000). On natural Learning and pruning in multilayered perceptrons. *Neural Computation*, 12, 1037–1057.
6. Igel, Ch., Toussaint, M., Weishui, W. (2005). Rprop Using the Natural Gradient, in *Trends and Applications in Constructive Approximation*, International Series of Numerical Mathematics, Vol. 151, Birkhäuser.
7. LeCun, J., Bottou, L., Orr, G., Müller, K.R. Efficient BackProp, in *Neural Networks: tricks of the trade*, 9–50. Springer, 1998.
8. Murray, M., Rice, J.W. **Differential Geometry and Statistics**. Chapman & Hall, 1993.
9. Murphy, P., Aha, D. *UCI Repository of Machine Learning Databases*, Tech. Report, University of Califonia, Irvine, 1994.
10. Polak, F. **Computational Methods in Optimization**. Academic Press, 1971.
11. Press, W., Teukolsky, S., Vetterling, W., Flannery, B. **Numerical Recipes in C**. Cambridge U. Press, 1988.
12. Rao, C.R. (1945). Information and accuracy attainable in estimation of statistical parameters. *Bull. Cal. Math. Soc.*, 37, 81–91.
13. Rattray, M., Saad, D., Amari, S. (1998). Natural gradient descent for on–line learning. *Physical Review Letters*, 81, 5461–5464.
14. Yang, H., Amari, S. (1998). Complexity Issues in Natural Gradient Descent Method for Training Multi-Layer Perceptrons. *Neural Computation*, 10, 2137–2157.