Denker *et. al.* have already discussed how to approximate the Hessian of $E_D$ for the purpose of evaluating weight saliency and for assigning error bars to weights and network outputs [4, 10]. It can be evaluated in the same way that backpropagation evaluates $\nabla E_D$. Alternatively $\mathbf{A}$ can be evaluated by numerical methods, for example second differences. A third option: if variable metric methods are used to minimise $M$ instead of gradient descent, then the inverse Hessian is automatically generated during the search for the minimum. It is important, for the success of this Bayesian method, that the off–diagonal terms of the Hessian should be evaluated. Denker *et. al.*'s method can do this without any additional complexity. The diagonal approximation is no good because of the strong posterior correlations in the parameters.

# 4 A Demonstration

This demonstration examines the evidence for various neural net solutions to a small interpolation problem, the mapping for a two joint robot arm,

$$(\theta_1,\theta_2) \quad \rightarrow \quad (y_a, y_b) = (r_1\cos\theta_1 + r_2\cos(\theta_1+\theta_2), \\ r_1\sin\theta_1 + r_2\sin(\theta_1+\theta_2)).$$

For the training set I used $r_1 = 2.0$ and $r_2 = 1.3$, regular samples from a restricted range of $(\theta_1, \theta_2)$ were made, and gaussian noise of magnitude 0.05 was added to the outputs. The neural nets used had one hidden layer of sigmoid units and linear output units. During optimisation, the regulariser (2) was used initially, and an alternative regulariser is introduced later; $\beta$ was fixed to its true value (to enable demonstration of the properties of the quantity $\gamma$), and $\alpha$ was allowed to adapt to its locally most probable value.

Figure 1 illustrates the performance of a typical neural network trained in this way. Each output is accompanied by error bars evaluated using Denker *et. al.*'s method, *including off–diagonal Hessian terms.* If $\beta$ had not been known in advance, it could have been inferred from the data using equation (11). For the solution displayed, the model's estimate of $\beta$ in fact differed negligibly from the true value, so the displayed error bars are the same as if $\beta$ had not been fixed.

Figure 2 shows the data misfit versus the number of hidden units. Notice that, as expected, the data error tends to decrease monotonically with increasing number of parameters. The data misfit cannot serve as a criterion for choosing between solutions.

Figure 3 shows the evidence for about 100 different solutions using different numbers of hidden units. Notice how the evidence maximum has the characteristic shape of an 'Occam hill' — steep on the side with too few parameters, and shallow on the side with too many parameters. The quadratic approximations break down when the number of parameters becomes too big compared with the number of data points.

The next figures introduce the quantity $\gamma$, discussed in [14], the number of well–measured parameters. In cases where the evaluation of the evidence proves difficult, it may be that $\gamma$ will serve as a useful tool. For example, frequentist theory predicts that the addition of redundant parameters to a model should reduce $\chi^2_D$ by one unit per well–measured parameter; a stopping criterion could detect the point at which, as parameters are deleted, $\chi^2_D$ started to increase faster than with gradient 1 with decreasing $\gamma$ (figure 5). This use of $\gamma$ requires prior knowledge of the noise level $\beta$; that is why $\beta$ was fixed to its known value for these demonstrations.

Now the question is how good a predictor of network quality the evidence is. The fact that the evidence has a maximum at a reasonable number of hidden units is promising. Furthermore, figure 6 shows that the performance of the solutions on an unseen test set has similar large scale structure to the evidence. However, figure 7 shows the evidence against the performance on a test set, and it can be seen that a significant number of solutions with poor evidence actually perform well on the test set. It is time for a discussion of the relationship between the evidence and generalisation ability. We will return later to the failure in figure 7 and see that it is rectified by the development of new, more probable regularisers.

## Relation to the theory of 'generalisation'

What is the relationship between the evidence and the generalisation error (or its close relative, cross–validation)? A correlation between the two is certainly expected. But the evidence is not necessarily a good predictor of generalisation error (see discussion in [14]). First, as illustrated in figure 8, the error on a test set is a noisy quantity, and a lot of data has to be devoted to the test set to get an acceptable signal to noise ratio. Furthermore, imagine that two models have generated solutions to an interpolation problem, and that their two most probable interpolants are completely identical. In this case, the generalisation error for the two solutions must be the same, but the evidence will not in general be the same: typically, the model that was *a priori* more complex will suffer a larger Occam factor and will have smaller evidence. Also, the evidence is a measure of plausibility of the whole ensemble of networks about the optimum, not just the optimal network. Thus there is more to the evidence than there is to the generalisation error.

### What if the Bayesian method fails?

I do not want to dismiss the utility of the generalisation error: it can be important for detecting failures of the model being used. For example, if we obtain a poor correlation between the evidence and the generalisation error, such that Bayes fails to assign a strong preference to solutions which actually perform well on test data, then we are able to detect and attempt to correct such failures.

where $\mathbf{A} = \nabla\nabla M$ is the Hessian of $M$ evaluated at $\mathbf{w}_{\rm MP}$.

The maximum of $P(D|\alpha,\beta,\mathcal{A},\mathcal{R})$ has the following useful properties:

$$\chi^2_W \equiv 2\alpha E_W = \gamma \tag{10}$$

$$\chi^2_D \equiv 2\beta E_D = N - \gamma \tag{11}$$

where $\gamma$ is the effective number of parameters determined by the data,

$$\gamma = \sum_{a=1}^{k} \frac{\lambda_a}{\lambda_a + \alpha} \tag{12}$$

where $\lambda_a$ are the eigenvalues of the quadratic form $\beta E_D$ in the natural basis of $E_W$.

**Comparison of different models**

To rank alternative architectures and penalty functions $E_W$ in the light of the data, we simply evaluate the evidence, $P(D\,|\mathcal{A},\mathcal{R})$, which appeared as the normalising constant in (7). Integrating the evidence for $(\alpha,\beta)$, we have:

$$P(D|\mathcal{A},\mathcal{R}) = \int P(D|\alpha,\beta,\mathcal{A},\mathcal{R})P(\alpha,\beta)\,d\alpha\,d\beta \tag{13}$$

The evidence is the Bayesian's transportable criterion for model comparison.

# 3 Adapting the framework

For neural networks, $M(\mathbf{w})$ is not quadratic. Indeed it is well known that $M$ typically has many local minima. And if the network has a symmetry under permutation of its parameters, then we know that $M(\mathbf{w})$ must share that symmetry, so that every single minimum belongs to a family of symmetric minima of $M$. For example if there are $H$ hidden units then each non–degenerate minimum is in a family of size $g = H!\,2^H$. Now it may be the case that the significant minima of $M$ are locally quadratic, so we might be able to evaluate $Z_M$ by evaluating (9) at each significant minimum and adding up the $Z_M$s; but the number of those minima is unknown, and this approach to evaluating $Z_M$ would seem dubious. Luckily however, we do not actually want to evaluate $Z_M$. Referring to the previous section, we would need to evaluate $Z_M$ in order to assign a posterior probability over $\alpha,\beta$, and to evaluate the evidence for alternative architectures and regularisers. This is not quite what we wish to do: when we use a neural network to perform a mapping, we typically only implement one neural network at a time, and this network will have its parameters set to a *particular solution* of the learning problem. Therefore the alternatives we wish to rank are the different solutions of the learning problem, *i.e.* the different minima of $M$. We would only want a posterior probability over the number of hidden units if we were somehow able to simultaneously implement the entire posterior ensemble of networks for one number of hidden units. Similarly, we do not want the posterior over $\alpha,\beta$ for the entire posterior ensemble; rather, we wish to allow each solution (each minimum of $M$) to choose its own optimal value for these parameters.

Having adopted this slight shift in objective, it turns out that to set $\alpha$ and $\beta$ and to compare alternative solutions to a learning problem, the integral we now need to evaluate is a local version of $Z_M$. Assume that the posterior probability consists of many islands in parameter space centred on minima of $M$. We wish to evaluate how much posterior probability mass is in each of these islands. Consider a minimum located at $\mathbf{w}^*$, and define a solution $S_{\mathbf{w}^*}$ as the ensemble of networks in the neighbourhood of $\mathbf{w}^*$, and all symmetric permutations of that ensemble. Let us evaluate the posterior probability for alternative solutions $S_{\mathbf{w}^*}$, and the parameters $\alpha$ and $\beta$:

$$P(S_{\mathbf{w}^*},\alpha,\beta,\mathcal{A},\mathcal{R}|D) \propto g\frac{Z^*_M(\mathbf{w}^*,\alpha,\beta)}{Z_W(\alpha)Z_D(\beta)}P(\alpha,\beta)P(\mathcal{A},\mathcal{R}) \tag{14}$$

where $g$ is the appropriate permutation factor, and $Z^*_M(\mathbf{w}^*,\alpha,\beta) = \int_{S_{\mathbf{w}^*}} d^k\mathbf{w}\,\exp -M(\mathbf{w},\alpha,\beta)$, where the integral is performed only over the neighbourhood of the minimum at $\mathbf{w}^*$. I will refer to the quantity $g\frac{Z^*_M(\mathbf{w}^*,\alpha,\beta)}{Z_W(\alpha)Z_D(\beta)}$ as the evidence for $\alpha,\beta,S_{\mathbf{w}^*}$. $\alpha$ and $\beta$ will be chosen to maximise this evidence. Then the quantity we want to evaluate to compare alternative solutions is the evidence[2] for $S_{\mathbf{w}^*}$,

$$P(D,S_{\mathbf{w}^*}|\mathcal{A},\mathcal{R}) = \int g\frac{Z^*_M(\mathbf{w}^*,\alpha,\beta)}{Z_W(\alpha)Z_D(\beta)}P(\alpha,\beta)\,d\alpha\,d\beta. \tag{15}$$

I propose that the gaussian approximation for $Z^*_M$ may meet our needs:

$$Z^*_M \simeq e^{-M(\mathbf{w}^*)}(2\pi)^{k/2}\det{}^{-\frac{1}{2}}\mathbf{A} \tag{16}$$

where $\mathbf{A} = \nabla\nabla M$ is the Hessian of $M$ evaluated at $\mathbf{w}^*$. For most $\alpha$ and $\beta$ this approximation is probably unacceptable; however we only need it to be accurate for the small range of $\alpha$ and $\beta$ close to their most probable value. The regime in which this approximation will definitely break down is when the number of constraints, $N$, is small relative to the number of free parameters, $k$. For large $N/k$ the central limit theorem encourages us to use the gaussian approximation. It is a matter for further research to establish how large $N/k$ must be for this approximation to be reliable.

What obstacles remain to prevent us from evaluating the local $Z^*_M$? We need to evaluate or approximate the inverse Hessian of $M$, and we need to evaluate or approximate its determinant and/or trace [14].

[2] Bayesian model comparison is performed by evaluating and comparing the evidence for alternative models. Gull and Skilling defined the evidence for a model $\mathcal{H}$ to be $P(D|\mathcal{H})$. The existence of multiple minima in neural network parameter space complicates model comparison. The quantity in (15) is not $P(D|S_{\mathbf{w}^*},\mathcal{A},\mathcal{R})$ (it includes the prior for $S_{\mathbf{w}^*}|\mathcal{A},\mathcal{R}$), but I have called it the evidence because it is the quantity we should evaluate to compare alternative solutions with each other and with other models.

where $Z_m(\beta) = \int d\mathbf{t}\, \exp -\beta E$. $E$ is the error for a single datum, and $\beta$ is a measure of the presumed noise included in $\mathbf{t}$. If $E$ is the quadratic error function then this corresponds to the assumption that $\mathbf{t}$ includes additive gaussian noise with variance $\sigma_\nu^2 = 1/\beta$.

**Prior.** A prior probability is assigned to alternative network connection strengths $\mathbf{w}$, written in the form:

$$P(\mathbf{w}|\alpha, \mathcal{A}, \mathcal{R}) = \frac{\exp -\alpha E_W(\mathbf{w}|\mathcal{A})}{Z_W(\alpha)} \tag{5}$$

where $Z_W = \int d^k\mathbf{w}\, \exp -\alpha E_W$. Here $\alpha$ is a measure of the characteristic expected connection magnitude. If $E_W$ is quadratic as specified in equation (2) then weights are expected to come from a gaussian with zero mean and variance $\sigma_W^2 = 1/\alpha$. Alternative 'regularisers' $\mathcal{R}$ (each using a different energy function $E_W$) implicitly correspond to alternative hypotheses about the statistics of the environment.

The posterior probability of the network connections $\mathbf{w}$ is then:

$$P(\mathbf{w}|D, \alpha, \beta, \mathcal{A}, \mathcal{R}) = \frac{\exp -(\alpha E_W + \beta E_D)}{Z_M(\alpha, \beta)} \tag{6}$$

where $Z_M(\alpha, \beta) = \int d^k\mathbf{w}\, \exp -(\alpha E_W + \beta E_D)$. Notice that the exponent in this expression is the same as the objective function $M$ defined in (3).

So under this framework, minimisation of $M = \alpha E_W + \beta E_D$ is identical to finding the (locally) maximum a posteriori parameters $\mathbf{w}_{\rm MP}$; minimisation of $E_D$ by backpropagation is identical to finding the maximum likelihood parameters $\mathbf{w}_{\rm ML}$.

Thus an interpretation has been given to backpropagation's energy functions $E_D$ and $E_W$, and to the parameters $\alpha$ and $\beta$.

This framework offers some partial enhancements for backprop methods: The work of Levin *et. al.* [12] makes it possible to predict the average generalisation ability of neural networks trained on one of a defined class of problems. However, it is not clear whether this will lead to a practical technique for choosing between alternative network architectures for real data sets.

Le Cun *et. al.* have demonstrated how to estimate the 'saliency' of a weight, which is the change in $M$ when the weight is deleted [10]. They have used this measure successfully to simplify large neural networks. However no stopping rule for weight deletion was offered other than measuring performance on a test set.

Also Denker and Le Cun demonstrated how the Hessian of $M$ can be used to assign error bars to the parameters of a network and to its outputs [4]. However, these error bars can only be quantified once $\beta$ is quantified, and how to do this without prior knowledge or extra data has not been demonstrated. In fact $\beta$ can be estimated from the data, as has been reviewed in [14].

# 2 Review of Bayesian regularisation and model comparison

In the companion paper [14] it was demonstrated how the control parameters $\alpha$ and $\beta$ are assigned by Bayes, and how alternative interpolation models can be compared. It was noted there that it is not satisfactory to optimise $\alpha$ and $\beta$ by finding the joint maximum likelihood value of $\mathbf{w}, \alpha, \beta$; the likelihood has a skew peak whose maximum is not located at the most probable values of the control parameters. The companion paper reviewed how the Bayesian choice of $\alpha$ and $\beta$ is neatly expressed in terms of a measure of the number of well–determined parameters in a model, $\gamma$. However that paper assumed that $M(\mathbf{w})$ only has one significant minimum which was well approximated as quadratic. (All the interpolation models discussed in [14] can be interpreted as two–layer networks with a fixed non–linear first layer.) In this section I briefly review the Bayesian framework, retaining that assumption. The following section will then discuss how the framework can be modified to handle neural networks, where the landscape of $M(\mathbf{w})$ is certainly not quadratic.

### Determination of $\alpha$ and $\beta$

By Bayes rule, the posterior probability for these parameters is:

$$P(\alpha, \beta\,|D, \mathcal{A}, \mathcal{R}) = \frac{P(D|\alpha, \beta, \mathcal{A}, \mathcal{R})P(\alpha, \beta)}{P(D|\mathcal{A}, \mathcal{R})} \tag{7}$$

Now if we assign a uniform prior to $(\alpha, \beta)$, the quantity of interest for assigning preferences to $(\alpha, \beta)$ is the first term on the right hand side, the **evidence** for $\alpha, \beta$, which can be written as[1]

$$P(D|\alpha, \beta, \mathcal{A}, \mathcal{R}) = \frac{Z_M(\alpha, \beta)}{Z_W(\alpha) Z_D(\beta)} \tag{8}$$

where $Z_M$ and $Z_W$ were defined earlier and $Z_D = \int d^N D\, e^{-\beta E_D}$.

Let us use the simple quadratic energy functions defined in equations (1,2). This makes the analysis easier, but more complex cases can still in principle be handled by the same approach. Let the number of degrees of freedom in the data set, *i.e.* the number of output units times the number of data pairs, be $N$, and let the number of free parameters, *i.e.* the dimension of $\mathbf{w}$, be $k$. Then we can immediately evaluate the gaussian integrals $Z_D$ and $Z_W$: $Z_D = (2\pi/\beta)^{N/2}$, and $Z_W = (2\pi/\alpha)^{k/2}$. Now we want to find $Z_M(\alpha, \beta) = \int d^k\mathbf{w}\, \exp -M(\mathbf{w}, \alpha, \beta)$. Supposing for now that $M$ has a single minimum as a function of $\mathbf{w}$, at $\mathbf{w}_{\rm MP}$, and assuming we can locally approximate $M$ as quadratic there, the integral $Z_M$ is approximated by:

$$Z_M \simeq e^{-M(\mathbf{w}_{\rm MP})} (2\pi)^{k/2} \det{}^{-\frac{1}{2}} \mathbf{A} \tag{9}$$

[1] The same notation, and the same abuses thereof, will be used as in [14].

[2] also fall into the category of additive weight–dependent energies. A sample weight energy term is:

$$E_W(\mathbf{w}|\mathcal{A}) = \sum_i \frac{1}{2} w_i^2 \tag{2}$$

The weight energy may be implicit, for example, 'weight decay' (subtraction of a multiple of $\mathbf{w}$ in the weight change rule) corresponds to the energy in (2). Gradient–based optimisation is then used to minimise the combined function:

$$M = \alpha E_W(\mathbf{w}|\mathcal{A}) + \beta E_D(D|\mathbf{w}, \mathcal{A}) \tag{3}$$

where $\alpha$ and $\beta$ are 'black box' parameters. The constant $\alpha$ should not be confused with the 'momentum' parameter sometimes introduced into backprop; in the present context $\alpha$ is a decay rate or regularising constant.

## What is lacking

The above procedures include a host of free parameters such as the choice of neural network architecture, and of the regularising constant $\alpha$. There are not yet established ways of objectively setting these parameters, though there are many rules of thumb (see [9] for examples).

One popular way of comparing networks trained with different parameter values is to assess their performance by measuring the error on an unseen test set or by similar cross–validation techniques. However the utility of these techniques in determining values for the parameters $\alpha$ and $\beta$ or for comparing alternative network solutions, etc., is limited because a large test set may be needed to reduce the signal to noise ratio in the test energy. Furthermore if there are several parameters like $\alpha$ and $\beta$, it is out of the question to repeat the learning with all possible values of these parameters so as to find the best values by using a test set. Such parameters must be optimised on line.

It is therefore interesting to study objective criteria for setting free parameters and comparing alternative solutions which depend only on the data set used for the training. Such criteria will prove especially important in applications where the total amount of data is limited, so that one doesn't want to sacrifice good data for use as a test set.

This paper will describe practical Bayesian methods for filling the following holes in the neural network framework just described:

1. **Objective criteria for comparing alternative neural network solutions, in particular with different architectures $\mathcal{A}$.**

   Given a single architecture $\mathcal{A}$, there may be more than one minimum of the objective function $M$. If there is a large disparity in $M$ between the minima then it is plausible to choose the solution with smallest $M$. But where the difference is not so great it is desirable to be able to assign an objective preference to the alternatives.

   It is also desirable to be able to assign preferences to neural network solutions using different numbers of hidden units, and different activation functions. Here there is an 'Occam's razor' problem: the more free parameters a model has, the smaller the data error $E_D$ it can achieve. So we cannot simply choose the architecture with smallest data error.

2. **Objective criteria for setting the decay rate $\alpha$.** As in the choice of $\mathcal{A}$ above, there is an 'Occam's razor' problem: a small value of $\alpha$ in equation (3) allows the weights to become large and overfit the noise in the data. This leads to a small value of the data error $E_D$ (and a small value of $M$), so we cannot base our choice of $\alpha$ only on $E_D$ or $M$. The Bayesian solution presented here can be implemented on–line, *i.e.* it is not necessary to do multiple learning runs with different values of $\alpha$ in order to find the best.

3. **Objective choice of regularising function $E_W$.**

4. **Objective criteria for choosing between a neural network solution and a solution using a different learning or interpolation model**, for example, splines or radial basis functions.

## The probability connection

Connections between probabilistic inference and neural networks have been discussed in [3, 8, 13, 21]. Tishby *et. al.* [21] introduced a probabilistic view of learning which is an important step towards solving the problems listed above. The idea is to force a probabilistic interpretation onto the neural network technique so as to be able to make objective statements. This interpretation does not involve the addition of any new arbitrary functions or parameters, but it involves assigning a meaning to the functions and parameters that are already used.

My work is based on the same probabilistic framework, and extends it using concepts and techniques adapted from Gull and Skilling's Bayesian image reconstruction methods [5]. This paper also adopts a shift in emphasis from Tishby *et. al.*'s paper: their work concentrated on predicting the average generalisation ability of a network trained on a task drawn from a known prior ensemble of tasks. In this paper the emphasis will be on quantifying the plausibility of alternative solutions to an interpolation or classification task; that task is defined by a single data set produced by the real world, and we do not know the prior ensemble from which the task comes.

Let us now review the probabilistic interpretation of network learning.

**Likelihood.** A network with specified architecture $\mathcal{A}$ and connections $\mathbf{w}$ is viewed as making predictions about the target outputs as a function of input $\mathbf{x}$ in accordance with the probability distribution:

$$P(\mathbf{t}^m|\mathbf{x}^m, \mathbf{w}, \beta, \mathcal{A}) = \frac{\exp -\beta E(\mathbf{t}^m|\mathbf{x}^m, \mathbf{w}, \mathcal{A})}{Z_m(\beta)}, \tag{4}$$

# A Practical Bayesian Framework for Backprop Networks

David J.C. MacKay
Computation and Neural Systems
California Institute of Technology 139–74
Pasadena CA 91125
mackay@hope.caltech.edu

May 27, 1991
Submitted to *Neural Computation* as a long paper

## Abstract

A quantitative and practical Bayesian framework is described for learning of mappings in feedforward networks. The framework makes possible: (1) objective comparisons between solutions using alternative network architectures; (2) objective stopping rules for deletion of weights; (3) objective choice of magnitude and type of weight decay terms or additive regularisers (for penalising large weights, etc.); (4) a measure of the effective number of well–determined parameters in a model; (5) quantified estimates of the error bars on network parameters and on network output; (6) objective comparisons with alternative learning and interpolation models such as splines and radial basis functions. The Bayesian 'evidence' automatically embodies 'Occam's razor,' penalising over–flexible and over–complex architectures. The Bayesian approach helps detect poor underlying assumptions in learning models. For learning models well–matched to a problem, a good correlation between generalisation ability and the Bayesian evidence is obtained.

### Prerequisite

This paper makes use of the Bayesian framework for regularisation and model comparison described in the companion paper 'Bayesian interpolation' [14]. This framework is due to Gull and Skilling [5].

## 1 The gaps in backprop

There are many knobs on the black box of 'backprop' (learning by back–propagation of errors [17]). Generally these knobs are set by rules of thumb, trial and error, and the use of reserved data to test generalisation ability (or more sophisticated cross–validation). The knobs fall into two classes: (1) parameters which change the effective learning model, for example, number of hidden units, and weight decay terms; and (2) parameters concerned with function optimisation technique, for example, 'momentum' terms. This paper is concerned with making objective the choice of the parameters in the first class, and with ranking alternative solutions to a learning problem. Bayesian techniques will be described which are both theoretically well–founded and practically implementable.

Let us review the basic framework for learning in networks, emphasising the points at which objective techniques are needed. The training set for the mapping to be learned is a set of input–target pairs $D = \{\mathbf{x}^m, \mathbf{t}^m\}$, where $m$ is a label running over the pairs. A neural network architecture $\mathcal{A}$ is invented, consisting of a specification of the number of layers, the number of units in each layer, the type of activation function performed by each unit, and the available connections between the units. If a set of values $\mathbf{w}$ is assigned to the connections in the network, the network defines a mapping $\mathbf{y}(\mathbf{x}|\mathbf{w}, \mathcal{A})$ from the input activities $\mathbf{x}$ to the output activities $\mathbf{y}$. The distance of this mapping to the training set is measured by some error function; for example the error for the entire data set is commonly taken to be

$$E_D(D\,|\mathbf{w}, \mathcal{A}) = \sum_m \frac{1}{2}\left(\mathbf{y}(\mathbf{x}^m|\mathbf{w}, \mathcal{A}) - \mathbf{t}^m\right)^2 \qquad (1)$$

The task of 'learning' is to find a set of connections $\mathbf{w}$ which gives a mapping which fits the training set well, *i.e.* has small error $E_D$; it is also hoped that the learned connections will 'generalise' well to new examples. Plain backpropagation learns by performing gradient descent on $E_D$ in $\mathbf{w}$–space. Modifications include the addition of a 'momentum' term, and the inclusion of noise in the descent process. More efficient optimisation techniques may also be used, such as conjugate gradients or variable metric methods. This paper will not discuss computational modifications concerned only with speeding the optimisation. It will address however those modifications to the plain backprop algorithm which implicitly or explicitly modify the objective function, with decay terms or regularisers.

It is moderately common for extra regularising terms $E_W(\mathbf{w})$ to be added to $E$; for example terms which penalise large weights may be introduced, in the hope of achieving a 'smoother' mapping [7, 9, 18]. Some of the 'hints' in