OGI Technical Report CSE 96-003

A Simple Method for Estimating the Weight Decay Parameter

Thorsteinn S. Rögnvaldsson[*]
Oregon Graduate Institute
Dept. of Computer Science
PO Box 91000, Portland, OR 97291
denni@cse.ogi.edu

Abstract

A new method for determining the optimal weight decay parameter λ is suggested. The method uses early stopping and the estimate

$$\hat{\lambda} = \frac{\|\nabla E_0(W_{es})\|}{\|\nabla R(W_{es})\|}, \tag{1}$$

where W_{es} is the set of weights at the early stopping point, $E_0(W)$ is the training data fit error, and $R(W)$ is the weight decay cost. We demonstrate the method on a large set of data sets, both synthetic and real-life, and show that it is essentially as good an estimator for the optimal λ as the standard search method. This new method is several orders of magnitude faster than the standard method.

1 Introduction

Any practical regression problem is ill-posed (Tikhonov & Arsenin 1977), in the sense that there are infinitely many functions that can fit a finite set of training data perfectly. Furthermore, since any real-life data set consists of noisy inputs and/or outputs, a perfect fit is not even desirable, and models have to be constructed on the basis of some other criteria than just fitting the training data. Methods for doing this are, e.g., to stop training the models before they fit the training data perfectly, or impose additional constraints and thereby shrink the space of possible solutions. The former is usually called "early stopping", and has been a standard practice with neural networks for many years. The latter technique comes in many different shapes and shades, like adding a regularization term (e.g. weight decay) to the error of fit, using "hints" (Abu-Mustafa 1995), or using only specific basis functions (Girosi, Jones & Poggio 1995).

Common for all these techniques is the necessity for determining how important the additional requirements are in relation to the error of fit. When doing early stopping, it is important to determine the best early stopping point. When using weight decay, one must decide how much the weights should decay, and so on and so forth.

This paper focusses on quadratic weight decay (Plaut, Nowlan & Hinton 1986) and the problem of determining an optimal weight decay parameter. A standard technique for this is to try several different values and use a validation data set, or several validation sets, to determine the best value. This search procedure can be very time consuming, since, for each weighting factor, a network has to be trained to minimize the total error, including both the error of fit and the weight decay cost. For a realistic problem, this can take several CPU hours on a high end UNIX workstation. Furthermore, to average out noise in the experiments, one often has to train a set of neural networks for each weighting

[*] Adress after Sept. 1, 1996: Centre for Computer Architecture, Halmstad University, PO Box 823, S-301 18 Halmstad, Sweden

factor, extending the computational cost to the order of CPU days. We suggest a much simpler method for estimating the weight decay parameter, which is orders of magnitude quicker than, but essentially as precise as, the standard search method.

2 Two Simple Estimates for λ

The error functional we minimize is

$$E(W) = \frac{1}{2M}\sum_{i=1}^{M}[y^{(i)} - f(W, \boldsymbol{x}^{(i)})]^2 + \lambda\|W\|^2 = E_0(W) + \lambda R(W) \quad , \tag{2}$$

where $f(W, \boldsymbol{x})$ is the output from the neural network, $y^{(i)}$ are target values corresponding to the inputs $\boldsymbol{x}^{(i)}$, and M is the number of training patterns. W denotes the set of all weights in the network, viewed as a vector. The weight decay cost $\|W\|^2$ imposes a cost for the magnitude of each weight, and the weight decay parameter λ controls the importance of this constraint relative to the fit to the data. To simplify notation, we henceforth denote the first term in the error (the error of fit) by $E_0(W)$, and the weight decay cost (the regularization cost) by $R(W)$.

If W^* is the set of weights when $E(W)$ is minimized, then

$$\nabla E(W^*) = \nabla E_0(W^*) + \lambda\nabla R(W^*) = 0, \tag{3}$$

which implies

$$\lambda = \frac{\|\nabla E_0(W^*)\|}{\|\nabla R(W^*)\|} \tag{4}$$

for the weight decay parameter λ. Thus, if we knew W^*, then it would be a trivial task to determine λ.

Unfortunately, we do not know W^*. However, if we can find a reasonable estimate of W^*, or of $\|\nabla E_0(W^*)\|$ and $\|\nabla R(W^*)\|$, then perhaps we can estimate λ. A naïve and appealingly simple way of estimating W^* is to use early stopping. That is, monitor the error on a validation data set while training, and stop training when the validation error starts to increase. The weights at the early stopping point can then be used as estimates for W^*.

Denoting the set of weights at the early stopping point by W_{es}, we have

$$\hat{\lambda}_1 = \frac{\|\nabla E_0(W_{es})\|}{\|\nabla R(W_{es})\|}, \tag{5}$$

as a simple estimate for λ. Another possibility is to consider the whole set of linear equations defined by

$$\nabla E(W_{es}) = \nabla E_0(W_{es}) + \lambda\nabla R(W_{es}) = 0 \tag{6}$$

and minimize the total squared error, which gives

$$\hat{\lambda}_2 = \max\left[0, \frac{-\nabla E_0(W_{es})\cdot\nabla R(W_{es})}{\|\nabla R(W_{es})\|^2}\right]. \tag{7}$$

The second estimate, $\hat{\lambda}_2$, corresponds to a linear regression without intercept term on the set of points $\{\partial_i E_0(W_{es}), \partial_i R(W_{es})\}$, whereas the first estimate, $\hat{\lambda}_1$, is closer to the ratio $\max[|\partial_i E_0(W_{es})|]/\max[|\partial_i R(W_{es})|]$. It follows from the Cauchy-Schwartz inequality that

$$\hat{\lambda}_1 \geq \hat{\lambda}_2. \tag{8}$$

These estimates are sensitive to the particularities of the training and validation data sets used, and possibly also to the training algorithm. Therefore, one should at least average them over different validation and training sets. However, since early stopping training often is several orders of magnitude faster to do than a full minimization of (2), it will still be quicker to do this than to do a trace over different values of λ.

We see two reasons why this naïve approach could be believed to work. First, there is the established connection between early stopping and weight decay when learning starts from small weights (Sjöberg & Ljung 1995). Sjöberg

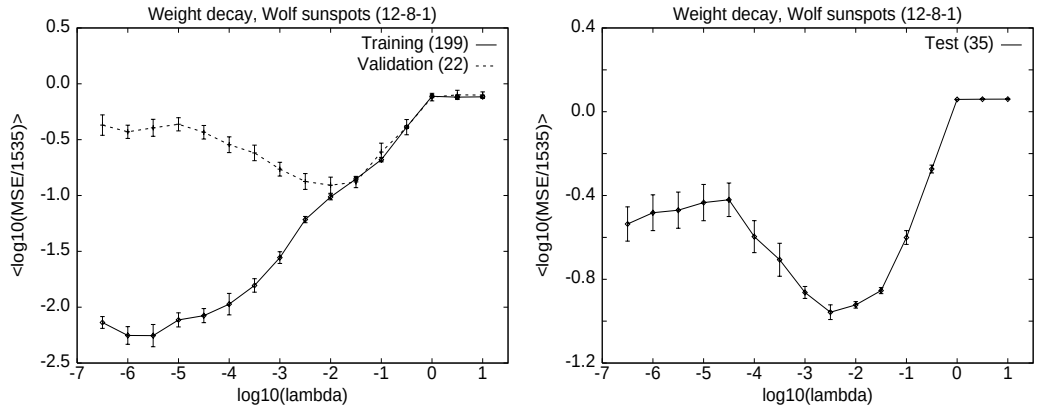

Figure 1: Left panel: The training and validation errors on the Wolf sunspot time series, plotted versus the weight decay parameter λ. Each point corresponds to an average over 10 runs with different validation and training sets. The error bars mark the 95% confidence limits for the average validation and training errors. Right panel: The corresponding plot for the error on the sunpots "test set 1". The network used had 12 inputs, 8 tanh internal units, and 1 linear output. The validation error indicates an optimal $\log\lambda = -2 \pm 0.75$, whereas the test set has an optimal $\log\lambda = -2.5 \pm 0.25$. The figure corresponds to problem B.2 in table 1.

& Ljung (1995) argue that, in the asymptotic limit and for a gradient descent scheme, the weight decay parameter λ is related to the number of iterations n roughly as

$$\lambda \sim \frac{1}{\eta n}, \tag{9}$$

where η is the learning rate. That is, stopping after a fixed number of iterations is essentially equivalent to using weight decay. A caveat, however, is that the derivation of (9) is based on a local expansion around the final and optimal stopping point, and small noise levels, which may not reflect the typical situation in practice.

Secondly, if the components of $\nabla E_0(W^*)$ and $\nabla R(W^*)$ are very different in size, then the true λ in expression (4) will be governed by the largest components of $\nabla E_0(W^*)$ and $\nabla R(W^*)$. It is a reasonable guess, for any gradient based learning scheme, that early stopping occurs after the largest gradient components of $E(W)$ have been zeroed, i.e. when the largest components of $E_0(W)$ and $R(W)$ have been traded off aginst each other. Consequently, the largest component of $\nabla E_0(W_{es})$ and $\nabla R(W_{es})$ should be fairly close to their true counterparts, and $\hat{\lambda}_1$ should be a good estimate of the true λ.

3 Experiments

We have tested the two proposed λ estimates on four different problems, both synthetic and real. For each problem, we varied the network architecture and/or the noise levels, resulting in a total of 20 different experiments.

The four problems are:

(A) Modeling a **synthetic bilinear problem** of the form

$$t(x_1, x_2) = x_1 x_2 \tag{10}$$

using three different sizes of training data sets, $M \in \{20, 40, 100\}$, but a constant validation set size of 10 patterns. The validation patterns were separate from the M training patterns. The test error, or generalization error, was computed by numerical integration over 201^2 data points on a two-dimensional lattice. The target values (but not the inputs) were contaminated with three different levels of gaussian noise with standard deviation $\sigma \in \{0.1, 0.2, 0.5\}$. This gave us a total of $3 \times 3 = 9$ different experiments on this particular problem.
This synthetic problem allowed us to do controlled studies with respect to noise levels and training set sizes, while keeping the network architecture constant (2 inputs, 8 tanh hidden, and one linear output).

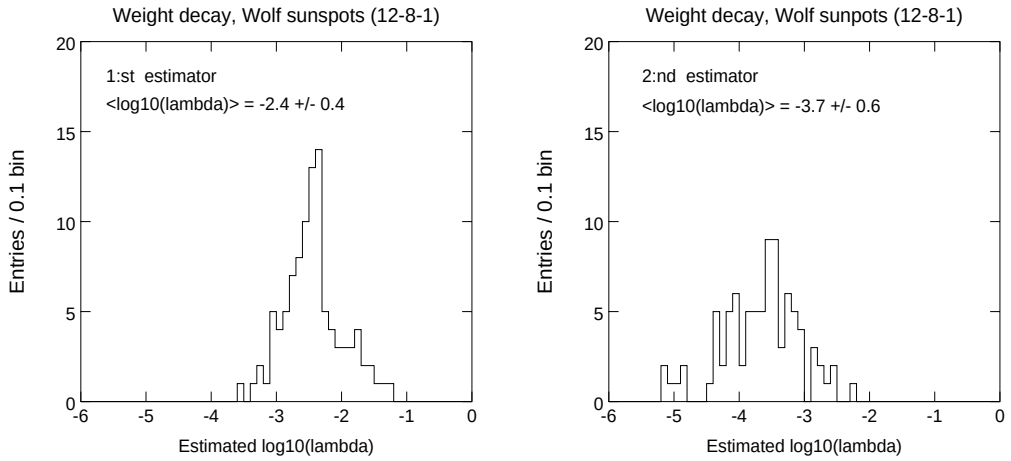

Figure 2: Left panel: Histogram showing the estimated values $\hat{\lambda}_1$ for 100 different training runs, using different training and validation sets each time. Right panel: Similar histogram for $\hat{\lambda}_2$. The problem is the same as that depicted in fig. 1. The figure corresponds to problem B.2 in table 1.

(B) Predicting the **yearly Wolf sunspots** time series. This time series has been used several times in the context of demonstrating new regularization techniques, for instance by Weigend, Rumelhart & Hubermann (1990) and Nowlan & Hinton (1992). We tried three different network architectures on this problem, always keeping 12 input units but using 4, 8, or 12 internal units in the network. The training set size was kept constant at $M = 221$ (years 1700-1920), out of which we randomly picked 22 patterns for validation. We used the standard "test set 1" with 35 data points (years 1921-1955) for testing.

(C) Predicting **daily riverflow in two Icelandic rivers**. This problem is tabulated in (Tong 1990), and the task is to model tomorrow's average flow of water (cubic meters per second) in one of two Icelandic rivers, knowing today's and previous day's waterflow, temperature, and precipitation. The training set consists of 731 data points, corresponding to the years 1972 and 1973, out of which we randomly sampled 150 datapoints for validation. The test set has 365 data points (1974).
We used two different lengths of lags, 8 or 4 days back, which corresponds to 24 or 12 inputs. The number of internal units was kept constant at 12.

(D) Estimating the **peak pressure position** in a combustion engine, using four external variables. This is a quite nonlinear and noisy task with only 49 training data points, out of which we randomly picked 9 patterns for validation. The test set consists of 35 separately data points, which were measured under slightly different conditions than the training data. We tested four different numbers of internal units on this task: 2, 4, 8, or 12.

The experimental procedure was the same for all four problems. We first estimated λ in the traditional way by doing a trace over the region $\log\lambda \in [-6.5, 1.0]$ in steps of $\Delta\log\lambda = 0.5$, measuring the average validation error for each value of λ. We then estimated λ using $\hat{\lambda}_1$ and $\hat{\lambda}_2$, and compared all three estimates to the "true" λ, defined as the λ that gave the lowest test set error (averaged over a number of test runs).

The λ trace was produced in the following way: For each λ value, a set of 10 networks were trained using the RPROP training algorithm (Riedmiller & Braun 1993). Each network was trained until the total error (2) was minimized, measured by a running average of the change in error divided by the magnitude of the weight update, or until 10^5 epochs had passed, whichever occured first (the convergence criterion was usually fulfilled within 10^5 epochs). New validation and training sets were used for each of the 10 networks, but the validation sets were allowed to overlap. Figure 1 shows an example of such a trace for the Wolf sunspot problem, using a neural network with 12 inputs, 8 internal units, and 1 linear output. Typically, the λ value that results in the lowest validation error will not be identical to the "true" optimal λ. Furthermore, the "best" λ value on the validation set is usually not unique and there will be a range of values that are likely. For instance, in figure 1 the optimal λ on the validation set was estimated to be

Problem	log[Trace λ]	log $\hat{\lambda}_1$	log $\hat{\lambda}_2$	log[True λ]
A.1 ($M = 20, \sigma = 0.1$)	-2.75 $\pm$ 0.50	-2.71 $\pm$ 0.66	-3.43 $\pm$ 0.81	-3.00 $\pm$ 0.25
A.2 ($M = 20, \sigma = 0.2$)	-2.50 $\pm$ 0.50	-2.32 $\pm$ 0.58	-3.24 $\pm$ 1.06	-2.25 $\pm$ 0.25
A.3 ($M = 20, \sigma = 0.5$)	0.00 $\pm$ 0.50	-1.93 $\pm$ 0.78	-2.93 $\pm$ 0.89	-2.25 $\pm$ 0.25
A.4 ($M = 40, \sigma = 0.1$)	-3.25 $\pm$ 0.50	-2.85 $\pm$ 0.73	-3.59 $\pm$ 0.95	-3.25 $\pm$ 0.25
A.5 ($M = 40, \sigma = 0.2$)	-2.50 $\pm$ 0.75	-2.41 $\pm$ 0.64	-3.18 $\pm$ 0.93	-2.75 $\pm$ 0.25
A.6 ($M = 40, \sigma = 0.5$)	-2.50 $\pm$ 0.75	-2.13 $\pm$ 0.74	-3.08 $\pm$ 1.11	-2.50 $\pm$ 0.25
A.7 ($M = 100, \sigma = 0.1$)	-3.50 $\pm$ 0.75	-3.01 $\pm$ 0.86	-3.77 $\pm$ 1.03	-4.00 $\pm$ 0.25
A.8 ($M = 100, \sigma = 0.2$)	-2.75 $\pm$ 0.50	-2.70 $\pm$ 0.73	-3.22 $\pm$ 0.91	-3.00 $\pm$ 0.25
A.9 ($M = 100, \sigma = 0.5$)	-3.00 $\pm$ 1.25	-2.34 $\pm$ 0.63	-3.31 $\pm$ 1.08	-2.50 $\pm$ 0.25
B.1 (12 hidden)	-2.50 $\pm$ 0.25	-2.48 $\pm$ 0.50	-3.70 $\pm$ 0.42	-2.50 $\pm$ 0.25
B.2 (8 hidden)	-2.00 $\pm$ 0.75	-2.43 $\pm$ 0.45	-3.66 $\pm$ 0.60	-2.50 $\pm$ 0.25
B.3 (4 hidden)	-2.50 $\pm$ 0.75	-2.39 $\pm$ 0.48	-3.54 $\pm$ 0.65	-2.75 $\pm$ 0.25
C.1 (river 1, 8 lags)	-2.50 $\pm$ 0.25	-2.65 $\pm$ 0.40	-3.92 $\pm$ 0.56	-2.00 $\pm$ 0.25
C.2 (river 1, 4 lags)	-2.75 $\pm$ 1.00	-2.67 $\pm$ 0.45	-3.70 $\pm$ 0.62	-2.50 $\pm$ 0.25
C.3 (river 2, 8 lags)	-2.50 $\pm$ 0.25	-3.10 $\pm$ 0.33	-4.57 $\pm$ 0.59	-2.50 $\pm$ 0.25
C.4 (river 2, 4 lags)	-2.50 $\pm$ 0.25	-3.15 $\pm$ 0.40	-4.20 $\pm$ 0.59	-2.75 $\pm$ 0.50
D.1 (12 hidden)	-3.00 $\pm$ 0.75	-3.03 $\pm$ 0.70	-4.06 $\pm$ 0.73	-3.40 $\pm$ 0.50
D.2 (8 hidden)	-3.00 $\pm$ 0.25	-3.02 $\pm$ 0.64	-3.85 $\pm$ 0.71	-3.00 $\pm$ 0.25
D.3 (4 hidden)	-3.75 $\pm$ 0.50	-3.07 $\pm$ 0.71	-3.76 $\pm$ 0.93	-3.25 $\pm$ 0.50
D.4 (2 hidden)	-3.00 $\pm$ 0.25	-3.46 $\pm$ 1.34	-4.12 $\pm$ 1.39	-2.00 $\pm$ 0.25

Table 1: Estimates of λ for the 20 different problem setups. Code A corresponds to the synthetic problem, code B to the Wolf sunspots, code C to the riverflow prediction, and code D to the maximum pressure position problem. For the log[Trace λ] column, errors indicate limits within which the difference between average validation errors is less than the 95% confidence limits (see figure 1). For the log[True λ] column, the errors indicate limits within which the difference between average test errors is less than the 95% confidence limit (see figure 1), or the quantization error $\Delta \log \lambda = 0.25$ due to the lattice search, whichever is smaller.

Problem	log[Trace λ]	log $\hat{\lambda}_1$	log $\hat{\lambda}_2$
NMSE_{est}	0.48	0.75	4.95
log [NMSE_{est}]	-0.32	-0.12	0.69
ρ_{est}	0.74	0.56	0.10

Table 2: Normalized estimation errors and linear correlations for the different estimation methods. All quantities are computed with problem setups A.3 and D.4 removed.

$\log \lambda = -2.5 \pm 0.75$, when the 95% confidence error bars were taken into consideration.

The early stopping estimates were done by training 100 networks, stopping the training when the validation error started to increase, and computing expressions (5) and (7) for the early stopping weight values. This produced 100 estimates, for which we computed means $\langle \hat{\lambda}_1 \rangle$ and $\langle \hat{\lambda}_2 \rangle$, and standard deviations. The mean value was taken as the estimate for λ and the standard deviation was used as a measure of the error. Figure 2 shows the histograms corresponding to the problem presented in fig. 1. The first estimate, $\hat{\lambda}_1$, was quite accurate in this case.

Table 1 and figure 3 summarize the results on all the 20 problem setups. We noted that experiments A.3 and D.4 are extreme cases, where at least one of the estimates are far off the target. In the D.4 case, this can be attributed to a peculiarity in the test set, whereas the A.3 case represents a case with very few datapoints and very low signal/noise ratio. We decided to mark these two cases as "outliers" and omitted them from the analysis below.

To measure of the quality of the estimators, we used the normalized estimation error

$$\text{NMSE}_{\text{est}} = \frac{\sum_{k=1}^{20} (\lambda(k) - \hat{\lambda}(k))^2}{\sum_{k=1}^{20} (\lambda(k) - \overline{\lambda})^2}, \tag{11}$$

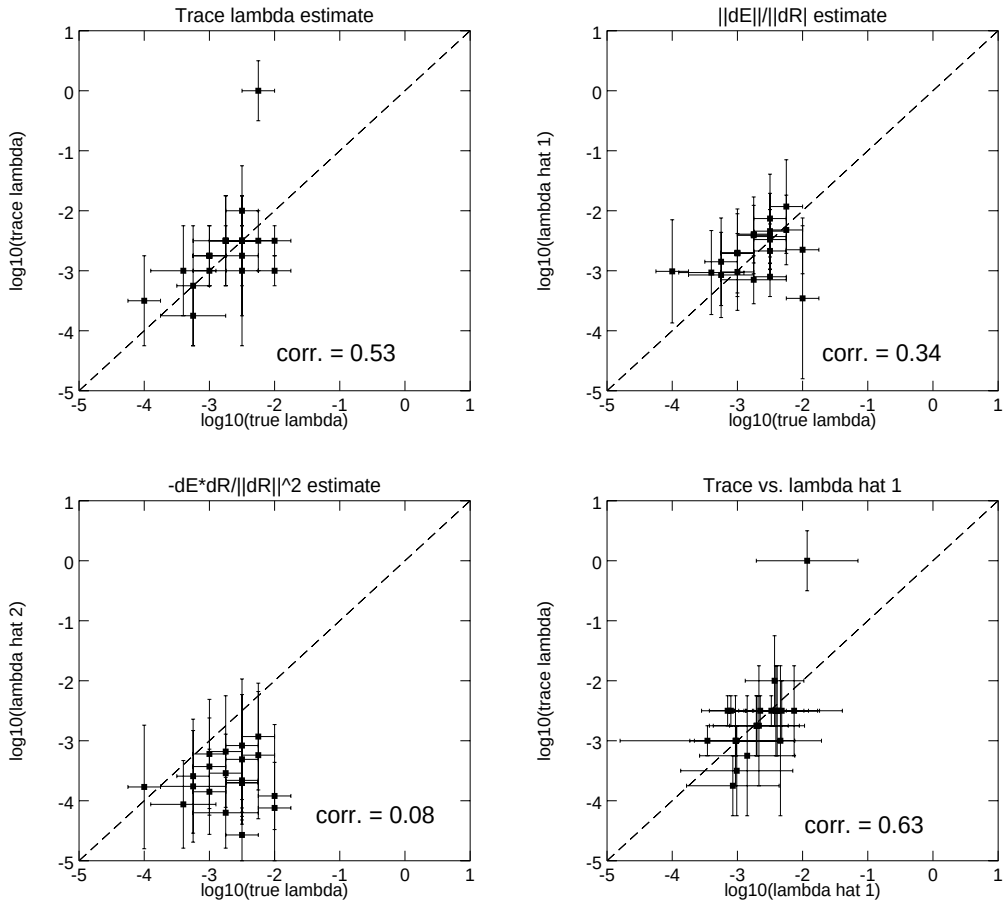

Figure 3: Plot showing the estimated λ values using the different estimators. The plot includes all 20 experimental setups. Upper left plate: The λ estimated from the trace over several values, plotted versus the "true" λ. Upper right plate: $\hat{\lambda}_1$ plotted versus the "true" λ. Lower left plate: $\hat{\lambda}_2$ plotted versus the "true" λ. Lower right plate: Plot showing the correlation between λ estimated from the trace and $\hat{\lambda}_1$.

and the coefficient of linear correlation between the estimate and the true λ

$$\rho_{\text{est}} = \frac{\sigma_{\lambda\hat{\lambda}}}{\sigma_\lambda \sigma_{\hat{\lambda}}}. \tag{12}$$

The 95% confidence intervals for these quantities, if we use 18 independent experiments, are estimated to be $\log[\text{NMSE}_{\text{est}}] \in [0, 1.5]$ and $\rho_{\text{est}} \in [-0.47, 0.47]$. Thus, if the values of these quantities are not outside of these limits, then we can not discard the zeroth hypothesis ("no relation between the estimate and the true value") with 95% confidence.

The values of NMSE_{est} and ρ_{est} for the different estimators are listed in table 2. The second early stopping estimator, $\hat{\lambda}_2$, is obviously of no use when estimating the true λ. However, both the trace estimate and $\hat{\lambda}_1$ are decent estimators. The trace estimator is somewhat better than $\hat{\lambda}_1$, the difference is small but significant, but it requires further experimentation to separate the two.

4 Discussion

We have presented a simple and fast method, based on early stopping, for estimating the weight decay parameter. The

method was initially intended for computing a "quick and dirty" estimate for λ, which could be used as an initial value for doing a full search using cross validation. The initial estimate, however, turned out to be essentially as good as the final λ, which rendered further search unneccesary!

Another finding that came out of this work, is the fact that guessing the weight decay parameter to be approximately $10^{-2.7}$ (the average of all λ values in the experiments) is a surprisingly good estimator. This is very good scaling indeed, given that the problems considered were quite different. This good scaling may explain some of the popularity of weight decay, since it implies that it one does not have to try a large range of values to find a good value for λ.

To give an example of the speed-up achieved by using an early stopping estimate instead of a full search estimate: It takes approximately 45 CPU minutes on a NeXT workstation to produce the histograms in figure 2, and it takes about 100 CPU hours on the same machine to produce the traces in figure 1. However, one would not scan the full $\log \lambda \in [-6.5, 1.0]$ region in a real application and this time can be cut by about 80%, since most of the time is spent on overfitting when λ is very small. The early stopping method is still roughly 25 times faster than the search method.

Acknowledgements

I thank David B. Rosen for a very inspiring conversation during the 1996 "Machines that Learn" Workshop in Snowbird, Utah. Financial support is gratefully acknowledged from NSF (grant CDA-9503968), and the Swedish Research Council for Engineering Sciences (contract TFR-282-95-847).

References

Abu-Mustafa, Y. S. (1995), 'Hints', *Neural Computation* **7**, 639–671.

Girosi, F., Jones, M. & Poggio, T. (1995), 'Regularization theory and neural networks architectures', *Neural Computation* **7**, 219–269.

Nowlan, S. & Hinton, G. (1992), 'Simplifying neural networks by soft weight-sharing', *Neural Computation* **4**, 473–493.

Plaut, D., Nowlan, S. & Hinton, G. (1986), Experiments on learning by backpropagation, Technical Report CMU-CS-86-126, Carnegie Mellon University, Pittsburg, PA.

Riedmiller, M. & Braun, H. (1993), A direct adaptive method for faster backpropagation learning: The RPROP algorithm, *in* H. Ruspini, ed., 'Proc. of the IEEE Intl. Conference on Neural Networks', San Fransisco, California, pp. 586–591.

Sjöberg, J. & Ljung, L. (1995), 'Overtraining, regularization, and searching for minimum with application to neural nets', *Int. J. Control* **62**(6), 1391–1407. Available by anonymous ftp `ftp.control.isy.liu.se`.

Tikhonov, A. N. & Arsenin, V. Y. (1977), *Solutions of Ill-Posed problems*, V. H. Winston & Sons, Washington D.C.

Tong, H. (1990), *Non-linear Time Series: A Dynamical System Approach*, Clarendon Press, Oxford.

Weigend, A., Rumelhart, D. & Hubermann, B. (1990), Back-propagation, weight-elimination and time series prediction, *in* T. Sejnowski, G. Hinton & D. Touretzky, eds, 'Proc. of the Connectionist Models Summer School', Morgan Kaufmann Publishers, San Mateo, California.