Progress in Neural Information Processing, ed. by S. Amari, L. Xu,
L.-W. Chan, I. King and K. S. Leung, Springer Verlag, 1996, p139-144.

Levenberg-Marquardt Learning and Regularization

Lai-Wan CHAN
Computer Science and Engineering Department
The Chinese University of Hong Kong, Shatin, HONG KONG
Email : lwchan@cs.cuhk.hk

***Abstract*— Levenberg-Marquardt Learning was first introduced to the feedforward networks to improve the speed of the training. This method is an improved Guass-Newton method which has an extra term to prevent the cases of ill-conditions. Interestingly, if we regard the learning as a constrained least square method, that extra term becomes a regularization term to deal with the additive noise in the training samples. In this paper, we look at the Levenberg-Marquardt Learning from the viewpoint of regularization. We show that the Levenberg-Marquardt learning allows other forms of regularization operators by some simple modifications. In addition, with the inclusion of test for validation error, the regularization parameter can be chosen in such a way that both the training error and validation error decrease. Thus, it prevents the occurrence of over-training.**

1 Introduction

Levenberg-Marquardt Learning had been introduced to feedforward networks for a number of years [HM94]. The primary objective of this learning algorithm is to speed up the learning process. Generalized delta-rule, the original learning algorithm of multi-layered feedforward networks, is based on gradient descent which is usually slow and the convergence is of linear order [RHW85]. In addition, the speed usually depends on some parameters, for example, the step size. Alternatively, various second order learning methods had been proposed [Bat92]. Among these second order methods, Levenberg-Marquardt method, a modification to the Gauss-Newton method, is one of the popular and effective methods. One characteristic of this method is that it incorporates an extra term to stabilize the system and to deal with the small residual problems in the learning.

In the past, the discussion of the Levenberg-Marquardt method was mainly related to the speed and convergence issue. This paper looks at the regularization aspect of the Levenberg-Marquardt Learning, on top of the convergence issue. By deriving the Levenberg-Marquardt Learning from the constrained least square method, we relate the extra term in the Levenberg-Marquardt Learning to the addition of a regularization term in the cost function. Unlike other papers on generalization and regularization, we are not trying to propose optimal or novel regularization methods. Instead, we are aiming at exploring the advantages of the Levenberg-Marquardt or related learning method.

2 The Single Layer Feedforward Network

Before going to the non-linear case of multi-layer feedforward network, we first consider the linear one where only one layer of neurons is concerned. As long as the activation function is a monotonic function, we can always rewrite the output of that layer as $y = xw$, where x is the input, and w is the weight vector. Using the ridge regression to solve the above equation gives

$$w_K = (x^T x + K)^{-1} x^T y$$

When $K = \gamma I$, I being the identity matrix, $\gamma > 0$, and $\gamma \in \mathcal{R}$. $w_K = w_\gamma = (x^T x + \gamma I)^{-1} x^T y$. This gives the ordinary ridge estimator. When K is a diagonal matrix of biasing factors γ_i (*i.e.* $K = diag(\gamma_i)$), it becomes the generalized ridge estimator. In either case, the estimators are the minima of the form

$$C(w) = (y - xw)^T (y - xw) + \frac{1}{2} w^T K w \tag{1}$$

Equation (1) is indeed a mean square error term together with a weight decaying term. Previous result shows that $E(w_k - w^*)^T (w_k - w^*) < E(w_0 - w^*)^T (w_0 - w^*)$ for some $0 < k < k_{max}$. w^* and w_0 are optimal weight vector and the solution of the ordinary least square method respectively [VU81]. k_{max} has been estimated as $\frac{2\sigma^2}{w^{*T} w^*}$ or $\frac{2}{[-\min(0,\zeta)]}$ and ζ is the minimum eigenvalue of $(x^T x)^{-1} - (w^* w^{*T} / \sigma^2)$. Thus, the ridge regression solution gives a small variance in the weight vector.

3 The Multi-layer Feedforward Network

In contrast to the single layer network, iterative updating is often used in a non-linear multi-layer feedforward networks. Let $y = f(x, w)$ be the function modeled by the network, with x and y as the input and output variables respectively. Using Taylor's series expansion of f, we have

$$f(x, w + \Delta w) = f(x, w) + J \Delta w \tag{2}$$

where J is the Jacobian matrix. Equating $f(x, w+\Delta w)$ to t, the expected output of the training samples, and let $e = t - y$, Equation 2 becomes

$$J\Delta w = e \tag{3}$$

The Levenberg-Marquardt learning gives $\Delta w = (J^T J + \gamma I)^{-1} Je$ as the solution of the above equation, with γ being a regularization parameter to prevent the ill-condition of the matrix $J^J J$. However, relating the Levenberg-Marquardt Learning to the constrained least square solution reveals that this learning method is indeed equivalent to the minimum square error of a cost function with some regularization.

Let w^* be the optimal weight vector and we assume that the training set, $\{\tilde{t}, x\}$, contains zero-mean additive noise, i.e. $\tilde{t} = f(x, w^*) + \eta$, where η has a variance of σ^2.

If $\tilde{e}$ denotes the difference between the target output of the training data and the model output, we have $\tilde{e} = \tilde{t} - y = e + \eta$. Thus,

$$E[||\tilde{e} - J\Delta w||^2] = N\sigma^2$$

with N being the dimension of $\tilde{e}$. Following the "Occam's razor", a less complex model is preferred. Thus, our object turns to the minimization of the model complexity subjected to the above constraint. Assuming that the initial weights are small in magnitude, we define the model complexity as $||Q\Delta w||^2$ and Q is a regularization operator on Δw. The expectation in the constraint term is estimated from the sum square error of the training data.

Use Lagrange multipliers,

$$C(\Delta w) = ||Q\Delta w||^2 + \mu(||\tilde{e} - J\Delta w||^2 - N\sigma^2) \tag{4}$$

Now, suppose Q is linear. The minimization of equation (4) yields

$$\Delta w = (J^T J + \gamma Q^T Q)^{-1} J^T \tilde{e} \tag{5}$$

where $\gamma = 1/\mu$.

4 The Regularizer

We see that equation (4) is in the form of a sum square error term with a regularizer. When regularization is concerned, it is not an easy task to evaluate the optimal form of the regularizers and the optimal value of the regularization parameter. The regularizer, based on some prior information, constrains the search space of the weight vector. For example, it controls the smoothness of the network output. This smoothness consequently affects the generalization performance of the network.

Different forms of regularizers have been proposed, for examples, the weight decaying [PNH86], the weight elimination [WRH91], the smoothing regularizers [WM95]. The regularizer that we are referring to in this paper differs from theirs in the way that the regularization is imposed on Δw instead of w. Instead of constraining the magnitude of the weight vector, as in the cases of the single layer network or the weight decaying, it constrains the magnitude of the changes of the weight vector. Since only the weight update is constrained, thus a prior requirement is that the magnitude of the initial weight vector has to be small. Comparing with the regularizer in the form of $||w - w^*||^2$ discussed in [SL94], this regularization uses the weight vector at the current iteration as a nominal guess of w^*.

In equation (5), which is the constrained least square solution of equation (3), we permit any form of linear regularizers, Q. A specific example is the case when $Q = I$, equation (5) becomes

$$\Delta w = (J^T J + \gamma I)^{-1} J^T \tilde{e}$$

If we choose γ according to the Marquardt strategy (we discuss it again in Section 4), it becomes the Levenberg-Marquardt Learning Algorithm [HM94].

Apart from $Q = I$, another example of Q is the finite difference matrix.

$$Q = \begin{pmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ 0 & 1 & -2 & & 0 & \\ & & \ddots & \ddots & & \\ & 0 & & 1 & -2 & 1 \\ & & & & 1 & -2 \end{pmatrix}$$

It guarantees that the estimation Δw does not oscillate wildly in the constrained solution by minimizing higher-order differences.

In addition to the regularization operator, the optimal choice of the regularization parameter is another interesting topic to study. Many research papers have shown the effects of varying the regulation parameters and have discussed the evaluation of the optimal value of the regularization factor [Wah94]. For examples, if the noise variance is known, λ can be estimated by the minimization of the Total Predicted Mean Square Error, $TPMSE(\gamma) = ||(I - K(\gamma))J\tilde{e}||^2 + \sigma^2 tr\{K(\gamma)\}^2, K(\gamma) = J(J^T J + \gamma Q^T Q)^{-1} J^T$. If the noise variance is unknown, we can estimate the parameter from the generalized cross validation function, $GCV(\gamma) = RSS(\gamma)/tr\{I - K(\gamma)\}^2$ [TBKT91]. Alternatively, γ can be estimated from the equivalent degrees of freedom method, in which γ satisfies $RRS(\gamma) = \sigma^2\{n - trK(\gamma)\}$ [TBKT91]. Weigend [WRH91] assigned different values of the regularization parameter before and after overfitting. Rögnvaldsson estimated the parameter based on the early stopping points [Rög96]. Hansen [HRSL94] adapted the regularization parameter by using a gradient descent in the estimated generalization error and showed that the regularization parameter decreases as pruning proceeds.

The above methods choose the regularization parameter mainly based on the final generalization performance. On the other hand, this parameter affects the stability and convergence of the training. As we can see from the constrained least square solution, the estimated SSE after a weight updating is

$$\begin{aligned} \widehat{e} &= J\Delta w \\ &= J(J^TJ + \gamma Q^TQ)^{-1}J^T\tilde{e} \end{aligned} \tag{6}$$

$$\widehat{e} - \tilde{e} = (I - J(J^TJ + \gamma Q^TQ)^{-1}J^T)\tilde{e}$$

If $Q = I$,

$$\begin{aligned} \widehat{e} - \tilde{e} &= (I - (I - (I + \frac{JJ^T}{\gamma})^{-1}))\tilde{e} \\ &= (I + \frac{JJ^T}{\gamma})^{-1}\tilde{e} \end{aligned}$$

The eigenvalues of $(I + \frac{JJ^T}{\gamma})^{-1}$ are $\frac{\gamma}{\gamma+\lambda_i}$, where λ_i's are the eigenvalues of JJ^T. Since all eigenvalues of $(I + \frac{J^TJ}{\gamma})^{-1}$ are smaller than 1, $||t - y||$ is decreasing, and γ is a crucial factor to determine the rate of convergence.

From the view point of convergence, there are numerous ways to adapt this parameter, for examples, the Levenberg method, the Marquardt method and the Fletcher methods [Sca85], the Kollias and Anastassiou method [KA89]. Among them, the Marquardt strategy, is a popular method used in the training of feedforward networks.

In Levenberg-Marquardt Learning [HM94], the regularization parameter is adapted as follows.

Step 1	Compute $SSE_{train}(w)$
Step 2	while $SSE_{train}(w) >$ threshold and gradient_vector $>$ min_gradient
	do
Step 3	decrement γ
Step 4	compute Δw according to equation (5).
Step 5	compute $SSE_{train}(w + \Delta w)$
Step 6	if $SSE_{train}(w + \Delta w) > SSE_{train}(w)$ then
Step 7	increment γ
	else
Step 8	$w \leftarrow w + \Delta w$
	end
	done

Note that this algorithm adapts the regularization parameter based on the criteria that there is a reduction in the training error (SSE_{train}). Considering that there are noises in the training samples, we prefer the reduction of the validation error ($SSE_{validation}$)to the reduction of the training error. In this respect, we modify the Levenberg-Marquardt Learning Algorithm by replacing the condition in Step 6 of the above with

Step 5 if $SSE_{validation}(w + \Delta w) > SSE_{validation}(w)$ then

Thus γ is adjusted to ensure the decrease in the validation error. At the same time, the training error usually decreases as long as the condition of $\gamma < \gamma_{max}$ (for some γ_{max}) is satisfied. It has to pointed out that the algorithm terminates when there is no further reduction on the validation error. In that case, γ continues to increase until the gradient vector diminishes below the threshold value.

5 Experimental Results

We illustrate the effects of the above strategy by an example. We generated 40 training data points from the equation

$$y = \frac{1}{2}sin(2x) + \frac{1}{2} + \eta \qquad -2 < x \le 2 \tag{7}$$

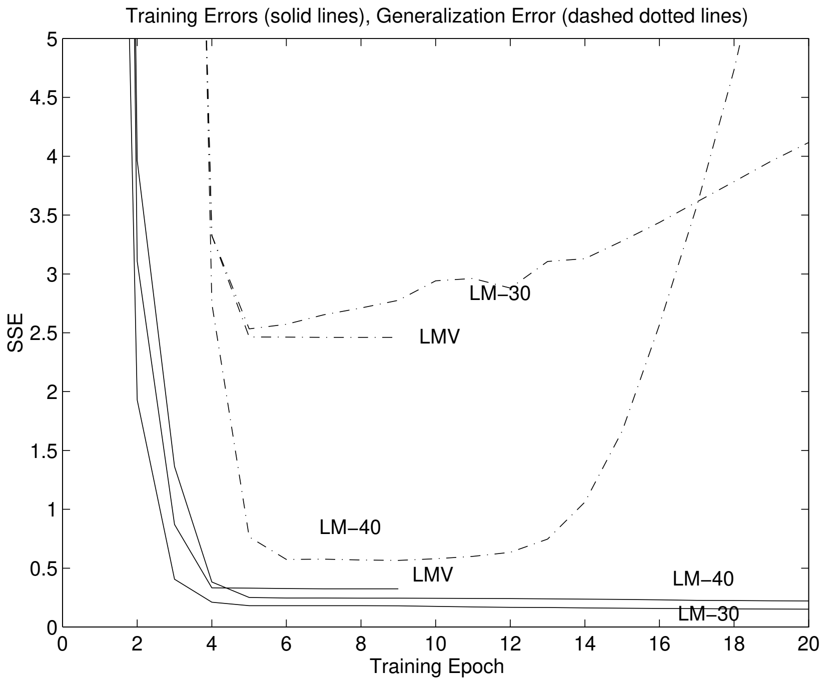

Figure 1: The training (solid lines) and generalization (dashed dotted lines) error during the training.

with $\eta = \mathcal{N}(0, 0.01)$. A one-hidden-layer feedforward network with 50 hidden nodes was trained using two different learning schemes; Levenberg-Marquardt Learning (LM) and the Levenberg-Marquardt with validation method (LMV). The weights were initialized with random numbers between [-1,+1]. LMV was trained with 30 data samples and the rest were put in the validation set. As a comparison, we trained LM with all data points (LM-40) and with only the 30 data points as in LMV (LM-30). Generalization error was measured by the sum square error (SSE) of 400 evenly distributed points in the same range. Figure 1 shows the SSE of the training set and the generalization error. The training error of LM and LMV were very close to each other. The training SSE of LMV stopped decreasing at around 0.25 but the training SSE of LM continued to decrease slowly. Figure 1 also shows that the generalization error of LM dropped to a minimum and then increased again, showing that the network is over-trained. The minimum of the generalization curve of LM-40 is quite flat, and this could be due to the effect of the regularization. When we look at the number of training iterations, about 5 iterations were required to reduce both training and generalization error to the minima and the generalization error maintained at similar level for another 8 iterations, which is about 1.5 the training time. For LMV, the generalization error converged to about 2.5 and did not increase further. Note that the generalization error of LMV and LM-30 is higher than that of LM-40. This is explained by the difference in the number of training data. LMV and LM-30 contained only 75% of the training data as in LM. The final network output is shown in Figure 2. It is obvious that the LM over-trained the system.

6 Conclusion and Discussion

Levenberg-Marquardt Learning Algorithm has a number of advantages over the gradient descent learning. Apart from the widely-known fact that it has a fast convergency, it has regularization effect. It provides regularization to stabilize the ill-condition cases during training. This regularization can also be regarded as a constraint to the amount of weight updating in each iteration. In addition, it can be generalized to cater for other linear regularizers. It provides a framework which has flexibility for the implementation of different types of regularization operators. Finally, the regularization parameter in Levenberg-Marquardt Learning was selected adaptively so as to reduce the training error. As generalization performance is concerned, this can be modified to form the Levenberg-Marquardt with validation Learning Algorithm. This method selects the regularization parameter based on the cross validation set so that it guarantees the decrease of both the training error and validation error. A major advantage of LMV is that it does not require any *a priori* knowledge of the data, such as the noise variance etc. In conclusion, Levenberg-Marquardt Learning provides an efficient and flexible learning method for feedforward network, in terms of training speed and the feasibility in the amalgamation with regularizers and cross-validation tests.

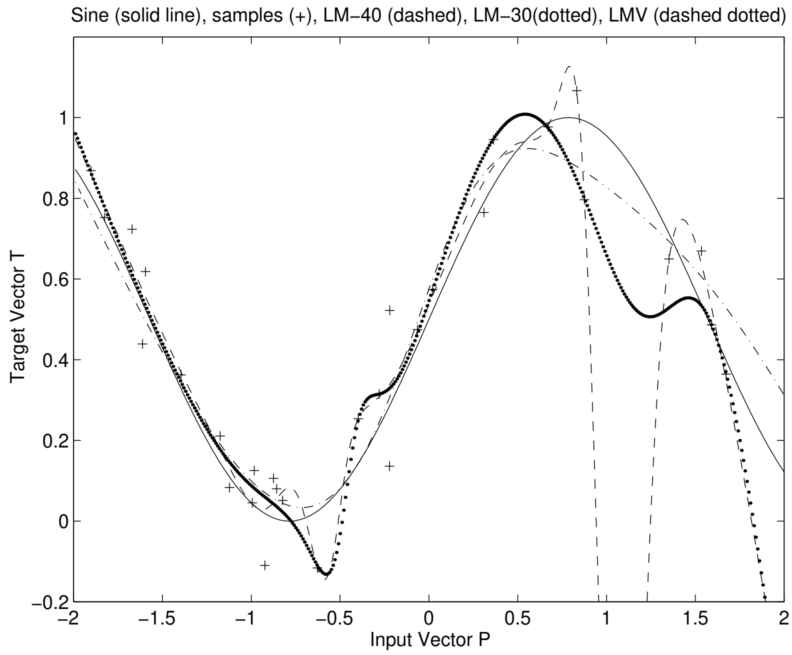

Figure 2: The training data (the crosses), the sin curve (solid line) and the prediction made by LM (dashed line) and the LMV (dashed dotted line).

Acknowledgement

The author would like to thank the RGC research grant for the financial support of this project. She would also like to thank the following people for valuable discussion during the preparation of the manuscript; C. P. Kwong, John Moody, John Sum, Z.B. Xu and members of the Neural Computing and Engineering Laboratory in the Computer Science and Engineering Department of The Chinese University of HongKong

References

[Bat92] R. Battiti. First and second-order methods for learning : Between steepest descent and newton's method. *Neural Computation*, 4(2):141–166, 1992.

[HM94] M. T. Hagan and M. B. Menhaj. Training feedforward networks with the Marquardt algorith. *IEEE Transactions on Neural Networks*, 5(6):989–993, November 1994.

[HRSL94] L. K. Hanson, C. E. Rasmussen, C. Svarer, and J. Larsen. Adaptive regularization. In *Proc. of the 4th IEEE workshop on Neural Networks for signal processing, Greece*, 1994.

[KA89] S. Kollias and D. Anastassiou. An adaptive least squares algorithm for the efficient training of artificial neural networks. *IEEE Transactions on Systems, Man and Cybernetics*, 36(8):1092–1101, 1989.

[PNH86] D. C. Plaut, S. J. Nowlan, and Geoffrey E. Hinton. Experiments on learning by back-propagation. Technical Report CMU-CS-86-126, Carnegie Mellon University, Pittsburgh PA 15213, June 1986.

[RHW85] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstruct ure of Cognition. Vol. 1: Foundations.* Cambridge, MA: Bradford Books/MIT Press, David E. Rumelhart. Institute for Cognitive Science, C-015; University of California, San Diego; La Jolla, CA 92093, September 1985.

[Rög96] T. S. Rögnvaldsson. A simple method for estimating the weight decaying parameter. Department of Computer Science and Engineering, Oregon Graduate Institute, Technical Report CSE 96-003, 1996.

[Sca85] L. E. Scales. *Introduction to Non-Linear Optimatization.* MacMillian, 1985.

[SL94] J. Sjöberg and L. Ljung. Overtraining, regularization, and searching for minimum with application to neural networks. Department of Electrical Engineering, Linköping University, LiTH-ISY-R-1567, 1994.

[TBKT91] A. M. Thompson, J. C. Brown, J. W. Kay, and D. M Titterington. A study of methods of choosing the smoothing parameter in image restoration by regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):326–338, 1991.

[VU81] H. D. Vinod and A. Ullah. *Recent Advances in Regression Methods*. Marcel Dekker, Inc, 1981.

[Wah94] G. Wahba. Generalization and regularization in nonlinear learning systems. Technical Report Technical Report : 921, Department of Statistics, University of Wisconsin, 1994.

[WM95] L. Wu and J.E. Moody. A smoothing regularizer for feedforward and recurrent neural networks. *Neural Computation*, 8(3):463–491, 1995.

[WRH91] A. S. Weigend, D. E. Rumelhart, and B. A. Huberman. Generalization by weight-elimination with application to forecasting. In Richard P. Lippmann, John E. Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing Systems 3*, pages 875–882. Morgan Kaufmann, 1991.