References

[1] Y.S. Abu-Mostafa (1990). The Vapnik–Chervonenkis dimension: information versus complexity in learning, *Neural Computation* **1** 3, 312–317.

[2] Y.S. Abu-Mostafa (1990). Learning from hints in neural networks, *J. Complexity* **6**, 192–198.

[3] J.S. Bridle (1989). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition, in *Neurocomputing: algorithms, architectures and applications,* F. Fougelman–Soulie and J. Hérault, editors, Springer–Verlag.

[4] J.S. Denker and Y. Le Cun (1990). Transforming neural-net output levels to probability distributions, *Preprint.*

[5] S.F. Gull (1989). Developments in Maximum entropy data analysis, in [20], 53–71.

[6] D. Haussler, M. Kearns and R. Schapire (1991). Bounds on the sample complexity of Bayesian learning using information theory and the VC dimension, Preprint.

[7] G.E. Hinton and S. Nowlan (1991), personal communication.

[8] G.E. Hinton and T.J. Sejnowski (1983). Optimal Perceptual Inference, *Proc. IEEE Conference on Computer Vision and Pattern Recognition,* 448–453.

[9] C. Ji, R.R. Snapp and D. Psaltis (1990). Generalizing smoothness constraints from discrete samples, *Neural Computation* **2** 2, 188-197.

[10] Y. Le Cun, J.S. Denker and S.S. Solla (1990). Optimal Brain Damage, in *Advances in neural information processing systems 2,* ed. David S. Touretzky, 598–605, Morgan Kaufmann.

[11] W.T. Lee and M.F. Tenorio (1991). On Optimal Adaptive Classifier Design Criterion — How many hidden units are necessary for an optimal neural network classifier?, *Purdue University* TR-EE-91-5.

[12] E. Levin, N. Tishby and S. Solla (1989). A statistical approach to learning and generalization in layered neural networks, *COLT '89: 2nd workshop on computational learning theory,* 245–260.

[13] S.P. Luttrell (1989). The use of Bayesian and entropic methods in neural network theory, in [20], 363–370.

[14] D.J.C. MacKay (1991). Bayesian interpolation, submitted to *Neural computation.*

[15] D.J.C. MacKay (1991). The evidence framework applied to classification networks, in preparation.

[16] Press, Flannery, Teukolsky and Vetterling (1988). *Numerical Recipes in C,* Cambridge.

[17] D.E. Rumelhart, G.E. Hinton and R.J. Williams (1986). Learning representations by back–propagating errors, *Nature* **323**, 533–536.

[18] D.E. Rumelhart (1987). Cited in [9].

[19] J. Skilling (1989). The eigenvalues of mega–dimensional matrices, in [20], 455–466.

[20] J. Skilling, editor (1989). *Maximum Entropy and Bayesian Methods, Cambridge 1988,* Kluwer.

[21] N. Tishby, E. Levin and S.A. Solla (1989). Consistent inference of probabilities in layered networks: predictions and generalization, *Proc. IJCNN, Washington.*

I thank Mike Lewicki, Nick Weir, and Haim Sompolinsky for helpful conversations, and Andreas Herz for comments on the manuscript.

This work was supported by a Caltech Fellowship and a Studentship from SERC, UK.

the evidence can be evaluated equally easily for interpolation and classification problems. V–C dimension is a worst case measure, so it yields different results from Bayesian analysis [6]. For example, V–C dimension is indifferent to the use of regularisers like (2), and to the value of α, because the use of such a regulariser does not rule out absolutely any particular network parameters. Thus V–C dimension assigns the same complexity to a model whether or not it is regularised.[3] So it cannot be used to set regularising constants α or to compare alternative regularisers. In contrast, the preceding demonstrations show that careful objective choice of regulariser and α is essential for the best solutions to be obtained.

Worst case analysis has a complementary role alongside Bayesian methods. Neither can substitute for the other.

Future tasks

The gaussian approximation used to evaluate the evidence breaks down when the number of data points is small compared to the number of parameters. For the model problems I have studied so far, the gaussian approximation seemed to break down significantly for $N/k < 3 \pm 1$. It is a matter for further research to characterise this failure and investigate techniques for improving the evaluation of Z_M^*, for example the use of random walks on M in the neighbourhood of a solution.

It will be interesting to see the results of evaluating the evidence for some larger real–world network problems.

Appendix: Numerical methods

Quick and dirty version

The three numerical tasks are automatic optimisation of α_c and β, calculation of error bars, and evaluation of the evidence. I will describe a cheap approximation for solving the first of these tasks without evaluating the Hessian. If we neglect the distinction between well–determined and poorly–determined parameters, we obtain the following update rules for α and β:

$$\begin{aligned} \alpha_c &:= k_c/2E_W^c \\ \beta &:= N/2E_D \end{aligned}$$

If you want an easy–to–program taste of what a Bayesian framework can offer, try using this procedure to update your decay terms.

Hessian evaluation

The Hessian of M, $\mathbf{A}$, is needed to evaluate γ (which relates to Trace $\mathbf{A}^{-1}$), to evaluate the evidence (which relates to Det $\mathbf{A}$), and to assign error bars to network outputs (using $\mathbf{A}^{-1}$).

I used two methods for evaluating $\mathbf{A}$: an approximate analytic method and second differences. The approximate analytic method was, following Denker *et. al.*, to use backprop to obtain the second derivatives, neglecting terms in f'', where f is the activation function of a neuron. The Hessian is built up as a sum of outer products of gradient vectors. Unlike Denker *et. al.*, I did not ignore the off–diagonal terms; the diagonal approximation is not good enough! For the evaluation of γ the two methods gave similar results, and either approach seemed satisfactory. However, for the evaluation of the evidence, the approximate analytic method failed to give satisfactory results. The 'Occam factors' are very weak, scaling only as $\log N$, and the above approximation apparently introduces systematic errrors greater than these. The reason that the evidence evaluation is more sensitive to errors than the γ evaluation is because γ is related to the sum of eigenvalues, whereas the evidence is related to the product; errors in small eigenvalues jeopardise the product more than the sum. I expect an exact analytic evaluation of the second derivatives would resolve this. To save programming effort I instead used second differences, which is computationally more demanding ($\sim kN$ backprops) than the analytic approach ($\sim N$ backprops). There were still problems with errors in small eigenvalues, but it was possible to correct these errors, by detecting eigenvalues which were smaller than theoretically permitted.

Demonstrations

The demonstrations were performed as follows:
Initial weight configuration: random weights drawn from a gaussian with $\sigma_W = 0.3$.
Optimisation algorithm for $M(\mathbf{w})$: variable metric methods, using code from [16], used several times in sequence with values of the fractional tolerance decreasing from 10^{-4} to 10^{-8}. Every other loop, the regularising constants α_c were allowed to adapt in accordance with the re–estimation:

$$\alpha_c := \gamma_c/2E_W^c \tag{17}$$

Precaution

When evaluating the evidence, care must be taken to verify that the permutation term g is appropriately set. It may be the case (probably mainly in toy problems) that the regulariser makes two or more hidden units in a network adopt identical connection values; alternatively some hidden units may switch off, with all weights set to zero; in these cases the permutation term should be smaller. Also in these cases, it is likely that the quadratic approximation will perform badly (quartic rather than quadratic minima are likely), so it is preferable to automate the deletion of such redundant units.

[3] However, E. Levine (personal communication) has mentioned that a measure of 'effective VC dimension' of a regularised model is being developed. It is speculated that this measure may turn out identical to γ, equation (12).

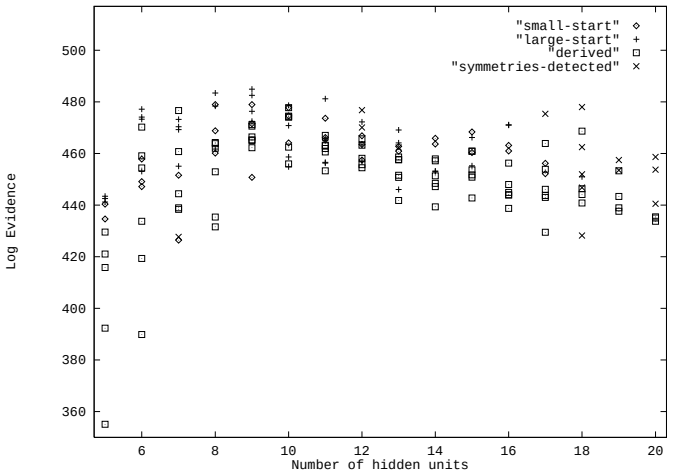

Figure 10: **Log Evidence versus number of hidden units for the second prior**
The different point styles correspond to networks with learning initialised with small and large values of σ_W; networks previously trained using the first regulariser and subsequently trained on the second regulariser; and networks in which a weight symmetry was detected (in such cases the evidence evaluation is possibly less reliable).

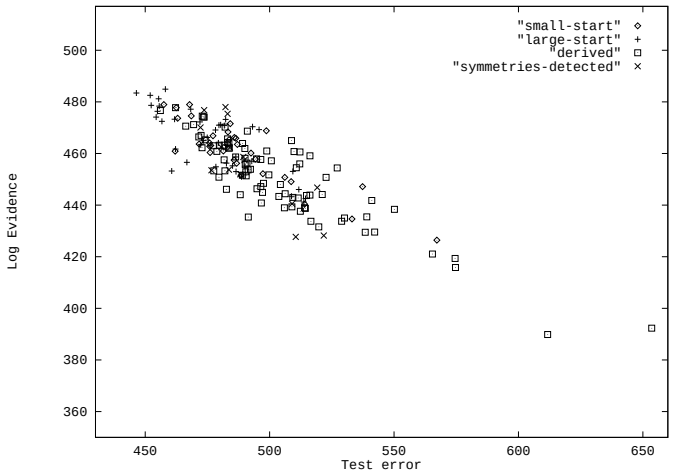

Figure 11: **Log Evidence for the second prior versus test error.**
The correlation between the evidence and the test error for the second prior is very good. Note that the largest value of evidence has increased relative to figure 7, and the smallest test error has also decreased.

that the evidence has increased compared to the evidence for the first prior. For some solutions the new prior is more probable by a factor of 10^{30}.

Now the crunch: does this more probable model make good predictions? The evidence for the second prior is shown against the test error in figure 11. The correlation between the two is greatly improved. Notice furthermore that not only is the second prior more probable, the best test error achieved by solutions found using the second prior is slightly better than any achieved using the first prior, and the number of good solutions has increased substantially. Thus the Bayesian evidence is a good predictor of generalisation ability, and the Bayesian choice of regularisers has enabled the best solutions to be found.

5 Discussion

The Bayesian method that has been presented is well–founded theoretically, and it works practically, though it remains to be seen how this approach will scale to larger problems. For a particular data set, the evaluation of the **evidence** has led us objectively from an inconsistent regulariser to a more probable one. The evidence is maximised for a sensible number of hidden units, showing that Occam's razor has been successfully embodied with no ad hoc terms. Furthermore the solutions with greatest evidence perform better on a test set than any other solutions found. I believe there is currently no other technique that could reliably find and identify better solutions using only the training set. Essential to this success was the simultaneous Bayesian optimisation of the three regularising constants (decay terms) α_c. Optimisation of these parameters by any orthodox search technique such as cross–validation would be laborious; if there were many more than three regularising constants, as could easily be the case in larger problems, it is hard to imagine any such search being possible.

This brings up the question of how these Bayesian calculations scale with problem size. In terms of the number of parameters k, calculation of the determinant and inverse of the Hessian scales as k^3. Note that this is a computation that needs to be carried out only a very small number of times compared with the immense number of derivative calculations involved in a typical learning session. However, for large problems it may be too demanding to evaluate the determinant of the Hessian. If this is the case, there are numerical methods available to approximate the determinant or trace of a matrix in k^2 time [19].

Application to classification problems

This paper has thus far discussed the evaluation of the evidence for backprop networks trained on interpolation problems. Neural networks can also be trained to perform classification tasks. A future publication [15] will demonstrate that the Bayesian framework for model comparison can be applied to these problems too.

Relation to V–C dimension

Some papers advocate the use of V–C dimension [1] as a criterion for penalising over–complex models [2, 11]. V–C dimension is most often applied to classification problems;

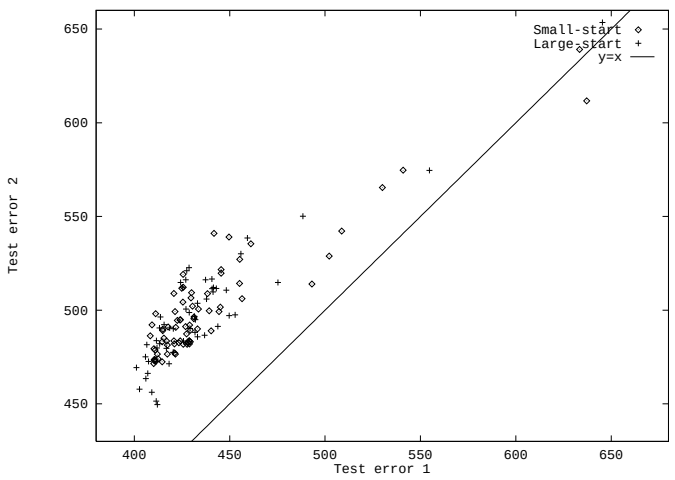

Figure 8: **Comparison of two test errors.**
This figure illustrates how noisy a performance measure the test error is. Each point compares the error of a trained network on two different test sets. Both test sets consist of 200 data points from the same distribution as the training set.

A failure indicates one of two things, and in either case we are able to learn and improve: either numerical inaccuracies in the evaluation of the probabilities caused the failure; or else the alternative hypotheses which were offered to Bayes were a poor selection, ill–matched to the real world (for example, a poor regulariser). When such a failure is detected, it prompts us to examine our models and try to discover the implicit assumptions in the model which the data didn't agree with; alternative models can be tried until a hypothesis is found that makes the data more probable.

We have just met exactly such a failure. Let us now establish what assumption in our model caused this failure and *learn* from it. Note that this mechanism for human learning is not available to those who just use the test error as their performance criterion. Going by the test error alone, there would have been no indication that there was a serious mismatch between the model and the data.

Back to the demonstration: comparing different regularisers

The demonstrations thus far used the regulariser (2). This is equivalent to a prior that expects all the weights to have the same characteristic size. This is actually an inconsistent prior: the input and output variables and hidden unit activities could all be arbitrarily rescaled; if the same mapping is to be performed (a simple consistency requirement), such transformations of the variables would imply independent rescaling of the weights to the hidden layer and to the output layer. Thus the scales of the two layers of weights are unrelated, and it is inconsistent to force the characteristic decay rates of these different classes of weights to be the same. This inconsistency is the major cause of the failure illustrated in figure 7. *All the networks deviating substantially from the desired trend have weights to the output layer far larger than the weights to the input layer;* this poor match to the model implicit in the regulariser causes the evidence for those solutions to be small.

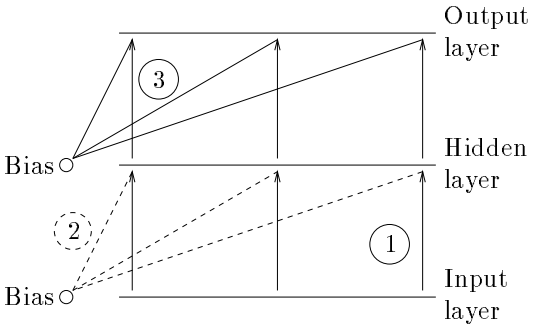

Figure 9: **The three classes of weights under the second prior**
1: Hidden unit weights. 2: Hidden unit biases. 3: Output unit weights and biases. The weights in one class c share the same decay constant α_c.

This failure enables us to progress with insight to new regularisers. The alternative that I now present is a prior which is not inconsistent in the way explained above, so there are theoretical reasons to expect it to be 'better'. However, we will allow the data to choose, by evaluating the evidence for solutions using the new prior; we will find that the new prior is indeed more probable.

The second prior has three independent regularising constants, corresponding to the characteristic magnitudes of the weights in three different classes c, namely hidden unit weights, hidden unit biases, and output weights and biases (see figure 9). The term αE_W is replaced by $\sum_c \alpha_c E_W^c$, where $E_W^c = \sum_{i \in c} w_i^2/2$. Hinton and Nowlan [7] have used a similar prior modelling weights as coming from a gaussian mixture, and using Bayesian re–estimation techniques to update the mixture parameters; they found such a model was good at discovering elegant solutions to problems with translation invariances.

Using the second prior, each regularising constant is independently adapted to its most probable value by evaluating the number of well–measured parameters γ_c associated with each regularising function, and finding the optimum where $2\alpha_c E_W^c = \gamma_c$. The increased complexity of this prior model is penalised by an Occam factor for each new parameter α_c (see [14]). Let me preempt questions along the lines of 'why didn't you use four weight classes?' Any other way of assigning weight decays is just another model, and you can try as many as you like; by evaluating the evidence you can then find out what preference the data have for the alternative decay schemes.

New solutions have been found using this second prior, and the evidence evaluated. The evidence for these new solutions with the new prior is shown in figure 10. Notice

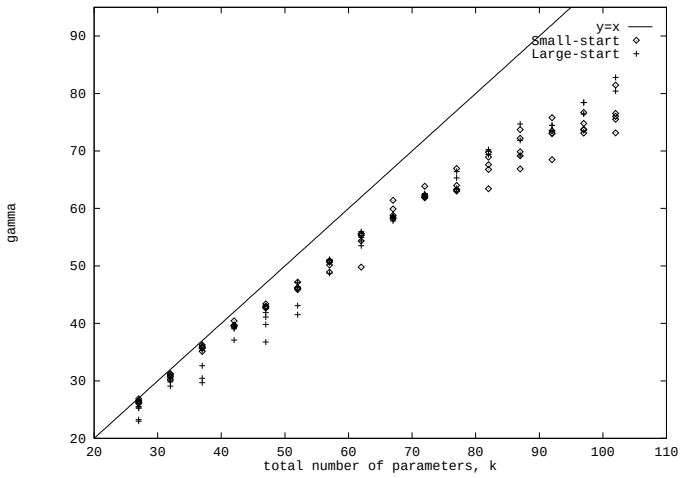

Figure 4: **The number of well–determined parameters.**
This figure displays γ as a function of k, for the same network solutions as in figure 3.

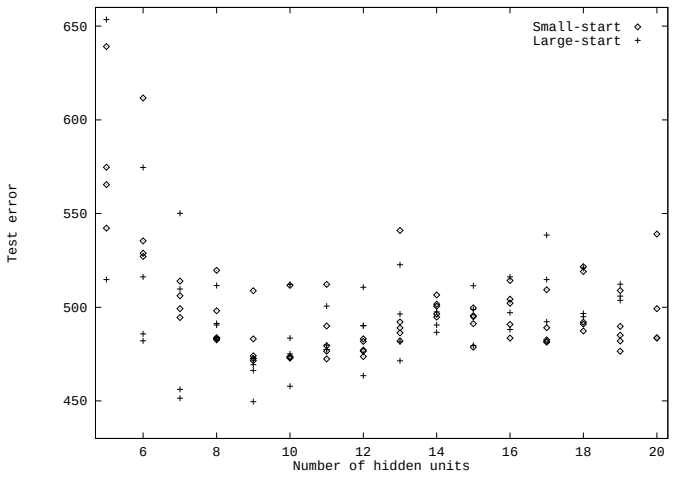

Figure 6: **Test error versus number of hidden units**
The training set and test set both had 200 data points. The test error for solutions found using the first regulariser is shown in dimensionless χ^2 units such that the expectation of error relative to the truth is 400 ± 20.

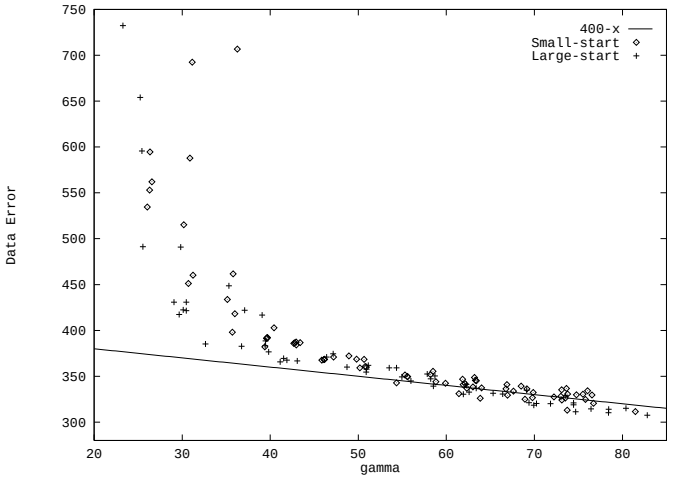

Figure 5: **Data misfit versus γ.**
This figure shows χ^2_D against γ, and a line of gradient -1. Towards the right, the data's misfit χ^2_D is reduced by 1 for every well–measured parameter. When the model has too few parameters however (towards the left), the misfit gets worse at a greater rate.

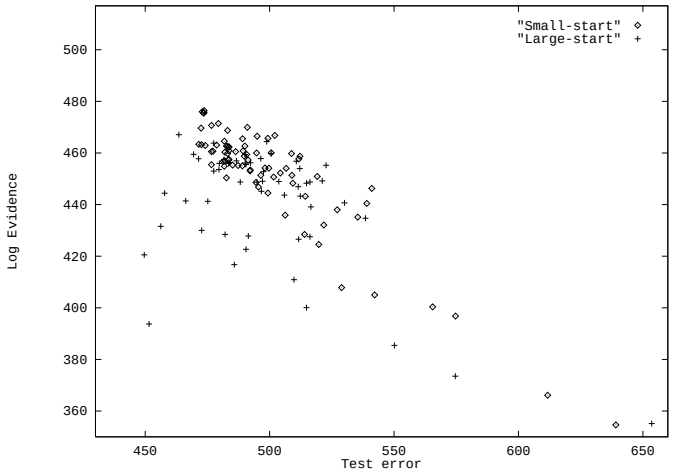

Figure 7: **Log Evidence versus Test error for the first regulariser**
The desired correlation between the evidence and the test error has negative slope. A significant number of points on the lower left violate this desired trend, so we have a failure of Bayesian prediction. The points which violate the trend are networks in which there is a significant difference in typical weight magnitude between the two layers. They are all networks whose learning was initialised with a large value of σ_W. The first regulariser is ill–matched to such networks, and the low evidence is a reflection of this poor prior hypothesis.

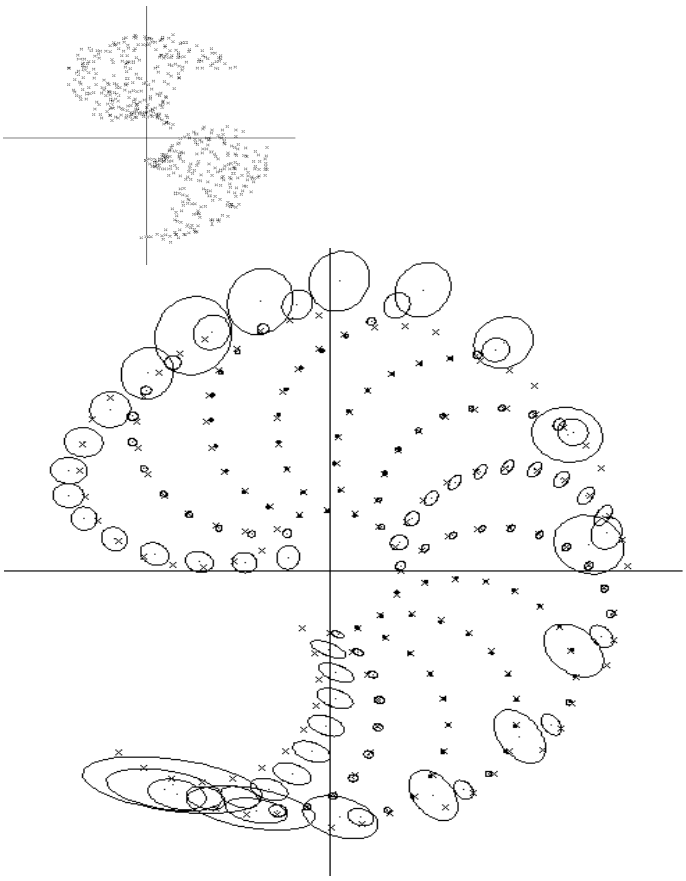

Figure 1: **Typical neural network output.** (Inset – training set)
This is the output space (y_a, y_b) of the network. The target outputs are displayed as small x's, and the output of the network with 1σ error bars is shown as a a dot surrounded by an ellipse. The network was trained on samples in two regions in the lower and upper half planes (inset). The outputs illustrated here are for inputs extending a short distance outside the training regions, and bridging the gap between them. Notice that the error bars get much larger around the perimeter. They also increase slightly in the gap between the training regions. These pleasing properties would not have been obtained had the diagonal Hessian approximation of [4] been used. The above solution was created by a three layer network with 19 hidden units.

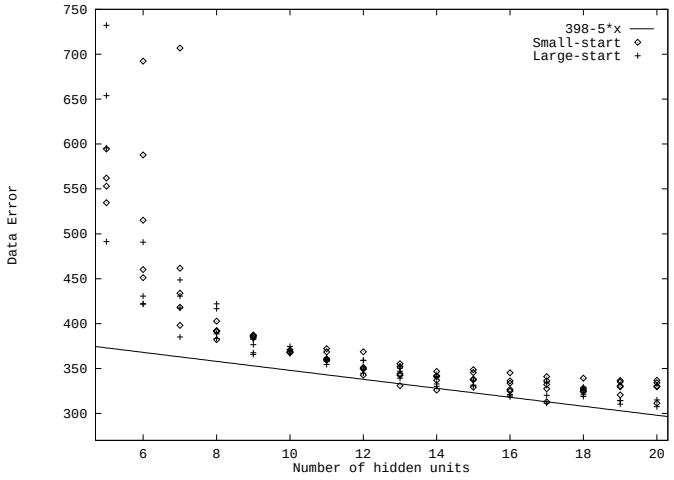

Figure 2: **Data error versus number of hidden units.** Each point represents one converged neural network, trained on a 200 i/o pair training set. Each neural net was initialised with different random weights and with different values for the initial value of $\sigma_W^2 = 1/\alpha$. The two point–styles correspond to small and large initial values for σ_W. The error is shown in dimensionless χ^2 units such that the expectation of error relative to the truth is 400 ± 20. The solid line is $400 - k$, where k is the number of free parameters.

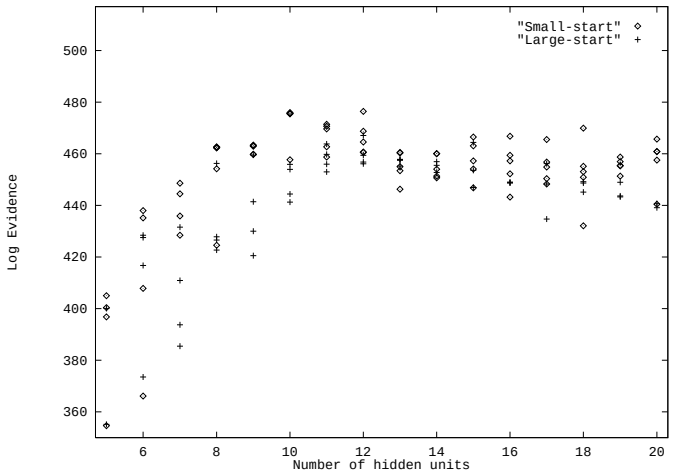

Figure 3: **Log Evidence for solutions using the first regulariser.**
For each solution, the evidence was evaluated. Notice that an evidence maximum is achieved by neural network solutions using 10, 11 and 12 hidden units. For more than ~ 19 hidden units, the quadratic approximations used to evaluate the evidence are believed to break down. The number of data points N is 400 (*i.e.* 200 i/o pairs); *c.f.* number of parameters in a net with 20 hidden units = 102.