Chapter 6

The Form of Topographic Transformations

6.1 Introduction

In many conventional applications which exploit neural networks — function approximation, classification and time-series prediction for example — there are several important design and implementation issues that must be considered, and these are equally relevant to neural networks which are to be trained to effect topographic mappings.

In terms of network design, a fundamental decision is that of *model order selection*, which is related to the trade-off between bias and variance as remarked upon in Section 4.4.2. It is necessary to determine the sufficient network *complexity* that will permit good generalisation — that is, it is naturally desired that projections of new data, drawn from the same distribution as that used to train the network, will also have low STRESS.

With respect to implementation, a key problem is one of network weight optimisation. There are numerous optimisation schemes available, and consideration of these will be deferred to the next chapter. Furthermore, from the evidence of previous research with Sammon Mappings, sub-optimal local minima may be expected to be highly problematic.

This chapter will examine the NEUROSCALE approach in terms of the form of the network transformation, with the emphasis on purely topographic, or objective, mappings (those with $\alpha = 0$). Firstly, the problem of local minima in network optimisation will be considered. While this may appear to be more relevant to Chapter 7, it in fact proves to be a fruitful starting point for this chapter as the results of that investigation lead naturally on to a discussion concerning the *smoothness* of the network transformation, which is in turn closely related to the questions of model complexity and generalisation.

6.2 Local Minima

6.2.1 The Sammon Mapping

The nonlinear optimisation procedures conventionally employed in the generation of Sammon mappings are *local* schemes (see Klein and Dubes [1989] and Hofmann and Buhmann [1995] for exceptions). When applied to the optimisation of the Sammon STRESS, the final configuration will be a local minimum of the STRESS function. In such applications, it has been a general observation that many of these minima, and respective configurations, are considerably sub-optimal. An empirical study of local minima in two-dimensional scaling was undertaken by Mathar and Zilinskas [1993]. When applied to the (10×10) cola-testing data [Schiffman, Reynolds, and Young 1981], for example, the region of attraction of the global solution was estimated at only 4.8%. Other starting configurations outwith this region gave rise to sub-optimal minima with correspondingly higher values of STRESS.

A similar experiment may be undertaken for 45 patterns[1] from the Iris data set [Fisher 1936], and the results are given in figure 6.1. (This subset of the Iris data will be referred to as the IRIS_45 set.) This presents a histogram of the number of runs of the Sammon mapping algorithm, out of a total of 1000, that gave corresponding minimum STRESS values. The configurations were initialised at random, and a conjugate-gradient minimisation routine employed.

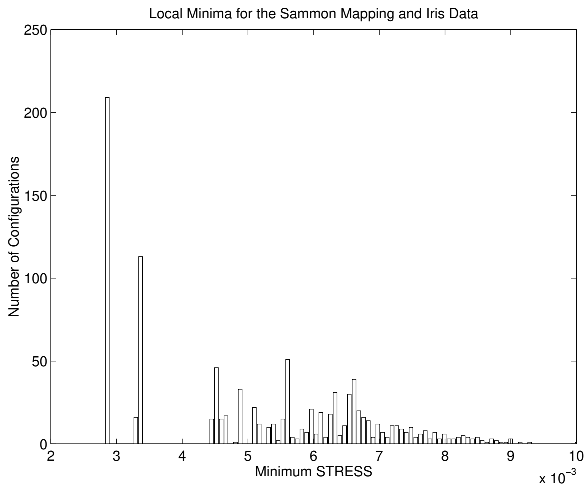

Figure 6.1: Histogram of number of final configurations with corresponding STRESS's for 1000 runs of the Sammon Mapping on 45 patterns from the Iris data set (IRIS_45).

It is clear that while the mode is at the global[2] minimum of 0.0028, there are many sub-optimal local minima. The proportion of runs which approached the global minimum was 21.3%, with the mean and standard deviation of the final STRESS being 0.0051 and 0.0017 respectively. It should be noted that the 'global' minimum found from random initialisation of the configuration was identical to that minimum obtained via PCA initialisation.

6.2.2 Parameterised Transformations

An equivalent experiment to figure 6.1 may also be conducted for NEUROSCALE (with $\alpha = 0$ to emulate the equivalent Sammon mapping). The results, for a radial basis function network with 10 Gaussian basis functions, are given in figure 6.2. The global width parameter, σ, of the basis functions was

[1] A representative subset of the full data was chosen in order to permit large numbers ($\approx$ 1000) of runs of the scaling algorithms.

[2] It is assumed, after many experiments, that this is indeed the *global* minimum, although this cannot be known for certain.

set to 3.0, where the k-th basis function is given by $\phi_k(\mathbf{x}_i) = \exp(- \parallel \mathbf{x}_i - \boldsymbol{\mu}_k \parallel^2 / \sigma^2)$, $\boldsymbol{\mu}_k$ being the respective centre.

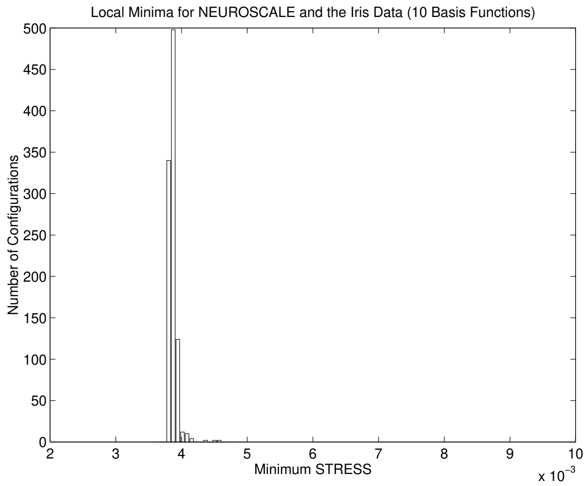

Figure 6.2: Histogram of number of final configurations with corresponding STRESS's for 1000 runs of NEUROSCALE, with $\alpha = 0$, on the `IRIS_45` dataset. The width of the 10 Gaussian basis functions was 3.0.

The contrast between the standard Sammon Mapping and the NEUROSCALE RBF approach is striking. In the case of NEUROSCALE, the minimum STRESS is 0.0038. This is notably higher than that for the Sammon Mapping, which might be expected due to the reduced flexibility of the RBF model implied by the use of only 10 basis functions. However, the maximum STRESS is only 0.0046, with the mean and standard deviation 0.0039 and 0.000074 respectively. The underlying result is that there does not appear to be a significant problem with sub-optimal local minima in the case of the NEUROSCALE approach on this dataset. This supports the observation of Webb [1995], when performing MDS by iterative majorisation with radial basis functions for the entire 150-pattern Iris data set, that "the value of the loss [STRESS] to which the procedure converged was relatively insensitive to the initial weight configuration."

It is natural to also consider a more flexible RBF, and figure 6.3 gives the minima obtained when optimising a model with 45 Gaussian basis functions, also of width 3.0.

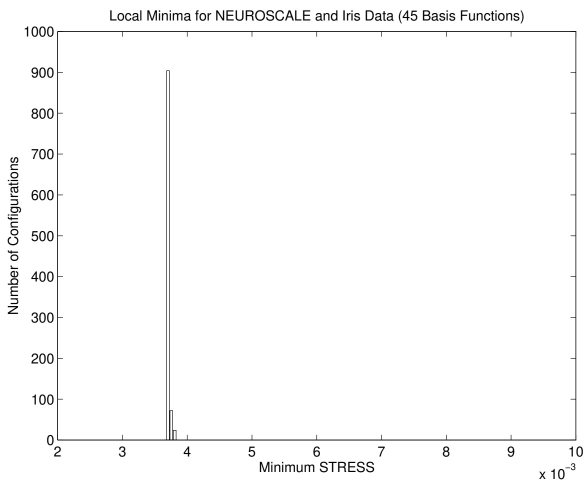

Figure 6.3: Histogram of number of final configurations with corresponding STRESS's for 1000 runs of NEUROSCALE, with $\alpha = 0$, on the `IRIS_45` data set. The width of the 45 Gaussian basis functions was 3.0.

This gives a very similar distribution of minima to the 10 basis function plot, and with a similar minimum of 0.0037. In this case, however, the maximum value is only 0.0038 and the standard deviation 2×10^{-5}. Despite the fact that there are as many basis functions in the RBF as data points, the network has still not attained the potential minimum STRESS value of 0.0028. Also, by comparison with figure 6.1, it can be seen that the apparent minimum (and that exhibited in figure 6.2 for 10 basis functions) does not correspond to one of the sub-optimal configurations generated by Sammon's algorithm. From this experiment, it is also retrospectively apparent that the relatively high minimum value of STRESS observed in figure 6.2 was not a result of employing only 10 basis functions in that network.

Further understanding of this phenomenon can be obtained by realising that there are *two* levels of minima in the neural network implementation. The first level is the set of minima as generated by Sammon's method. The second level is the set of extra minima introduced by the mechanism of the RBF model. To clarify this distinction, consider the equations for calculating the derivatives of STRESS with respect to the weights in the NEUROSCALE model. The relevant equations are reproduced here from Section 3.2:

$$\frac{\partial E}{\partial w_k} = \sum_i^N \frac{\partial E}{\partial \mathbf{y}_i} \cdot \frac{\partial \mathbf{y}_i}{\partial w_k}, \quad \text{where} \tag{6.1}$$

$$\frac{\partial E}{\partial \mathbf{y}_i} = -2 \sum_{j \neq i} \left(\frac{d^*_{ij} - d_{ij}}{d_{ij}} \right) (\mathbf{y}_i - \mathbf{y}_j). \tag{6.2}$$

The derivatives $\partial E / \partial \mathbf{y}_i$ are exactly those determined in Sammon's algorithm, and when these are all zero, clearly so are all the $\partial E / \partial w_k$. This represents the first level of minima.

However, even if these derivative terms are non-zero, all the $\partial E / \partial w_k$ may still be zero, given appropriate values of $\partial \mathbf{y}_i / \partial w_k$. To determine which such values give rise to minima, firstly the weights $\mathbf{w}_l = (w_1, w_2, \ldots, w_h)^{\mathrm{T}}$ for each output dimension, l, of the network are considered in isolation, as each such weight vector only affects a single network output (and the subscript 'l' will be dropped where it is unambiguous to do so). Then, equation (6.1) may be expressed in matrix form as:

$$\nabla E = \mathbf{J}^{\mathrm{T}} \boldsymbol{\delta}, \tag{6.3}$$

where

$$\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_h}\right)^{\mathrm{T}},$$

and $\mathbf{J}$ is the *Jacobian* matrix:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial y_{1l}}{\partial w_1} & \frac{\partial y_{1l}}{\partial w_2} & \cdots & \frac{\partial y_{1l}}{\partial w_h} \\ \frac{\partial y_{2l}}{\partial w_1} & \frac{\partial y_{2l}}{\partial w_2} & \cdots & \frac{\partial y_{2l}}{\partial w_h} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial y_{Nl}}{\partial w_1} & \frac{\partial y_{Nl}}{\partial w_2} & \cdots & \frac{\partial y_{Nl}}{\partial w_h} \end{bmatrix},$$

where y_{il} is the value of output dimension l for pattern i, and

$$\boldsymbol{\delta} = \left(\frac{\partial E}{\partial y_{1l}}, \frac{\partial E}{\partial y_{2l}}, \ldots, \frac{\partial E}{\partial y_{Nl}}\right)^{\mathrm{T}}.$$

Thus local minima occur, for non-zero $\boldsymbol{\delta}$, when $\boldsymbol{\delta}$ is orthogonal to the column-space of (or lies in the *null-space* of) $\mathbf{J}$. For an RBF with h basis functions, the Jacobian is given by

$$\mathbf{J} = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_h(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_h(\mathbf{x}_2) \\ \cdots & \cdots & \cdots & \cdots \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_h(\mathbf{x}_N) \end{bmatrix}, \tag{6.4}$$

and remains identical whichever output l of the network is considered.

With 45 basis functions, this matrix should be full rank, and no such second level minima should be present. However, for larger values of width parameter σ, it is likely to be ill-conditioned. This is the case for the example above, where the condition number (ratio of largest to smallest singular values) is 9.45×10^8. For an ill-conditioned Jacobian, if not singular and thus a true local minimum does not exist, numerical round-off error may effectively generate an artefactual one. Even if not, should $\boldsymbol{\delta}$ ever approach the subspace spanned by the eigenvectors corresponding to the very small eigenvalues, the gradient will become extremely small and training will, for all intents and purposes, terminate.

To view this more explicitly, consider a single output dimension l of the network once more, for a simple gradient-descent error minimisation scheme. The vector of output neuron l for all the patterns will be denoted $\mathbf{z}_l = (y_{1l}, y_{2l}, \ldots, y_{Nl})^{\mathrm{T}}$, and therefore represents the l^{th} column of the output data matrix $\mathbf{Y}$. The weight update equation at time step t is then $\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \boldsymbol{\nabla} E$. This implies that

$$\begin{aligned} \mathbf{z}_l^{t+1} &= \mathbf{J}\mathbf{w}^{t+1}, && (6.5) \\ &= \mathbf{J}(\mathbf{w}^t - \eta \boldsymbol{\nabla} E), && (6.6) \\ &= \mathbf{z}^t - \eta \mathbf{J}\mathbf{J}^{\mathrm{T}} \boldsymbol{\delta}^t. && (6.7) \end{aligned}$$

If there is a basis function located at each input point, then $\mathbf{J}$ is symmetric and may be decomposed as $\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\mathrm{T}}$, where $\mathbf{U}$ is the matrix of eigenvectors of $\mathbf{J}$ and $\boldsymbol{\Lambda}$ the corresponding diagonal matrix of eigenvalues. From this:

$$\mathbf{J}\mathbf{J}^{\mathrm{T}} = \mathbf{U}\boldsymbol{\Lambda}^2\mathbf{U}^{\mathrm{T}}. \quad (6.8)$$

If $\boldsymbol{\delta}$ is then expressed as a weighted sum of those eigenvectors, such that $\boldsymbol{\delta} = \sum_k \beta_k \mathbf{u}_k$, then the corresponding change in $\mathbf{z}$ is

$$\begin{aligned} \Delta \mathbf{z} &= \mathbf{z}^{t+1} - \mathbf{z}^t, && (6.9) \\ &= -\eta \sum_k \lambda_k^2 \beta_k \mathbf{u}_k. && (6.10) \end{aligned}$$

Thus, $\Delta \mathbf{z}$ will be diverted from the direction $\boldsymbol{\delta}$, such that its components in the directions of the principal eigenvectors of $\mathbf{J}$ are accentuated in comparison to those of the minor eigenvectors. The magnitude of this effect is related to the condition number of the matrix $\mathbf{J}$, and may be illustrated by figure 6.4 below. This figure provides a simple example of a two-dimensional quadratic error surface, where the minimum lies at the origin. The evolution of the vectors $\mathbf{z}^t$ are shown for an equivalent NEUROSCALE gradient-descent scheme with (2×2) $\mathbf{J}$, whose eigenvectors and squared eigenvalues are also given in the figure. For comparison, the plot for the Sammon mapping is also illustrated, which is equivalent to $\mathbf{J}\mathbf{J}^{\mathrm{T}} = \mathbf{I}$.

In figure 6.4, the trajectory for the equivalent Sammon mapping converges on the minimum directly as expected. By contrast, the effect of the λ_1^2 term in equation (6.10) above is to initially cause the gradient-descent trajectory to approach the origin more rapidly than the Sammon trace. However, $\boldsymbol{\delta}$ soon becomes near-perpendicular to $\mathbf{u}_1$, and the trajectory is dominated by the effect of the smallest eigenvalue, $\lambda_2^2 = 0.0225$, causing convergence on the origin in the direction $\mathbf{u}_2$ to be dramatically retarded.

Although the global surface of the STRESS cost function will be very different from the quadratic example given above, in the vicinity of a minimum such an approximation may be quite reasonable. (This assumption is indeed made implicitly by BFGS and other optimisation algorithms.)

By considering each output in isolation, if $\mathbf{J}$ is not full rank (for example, when there are fewer basis functions than input data points), then the implication is that there will be a genuine local minimum

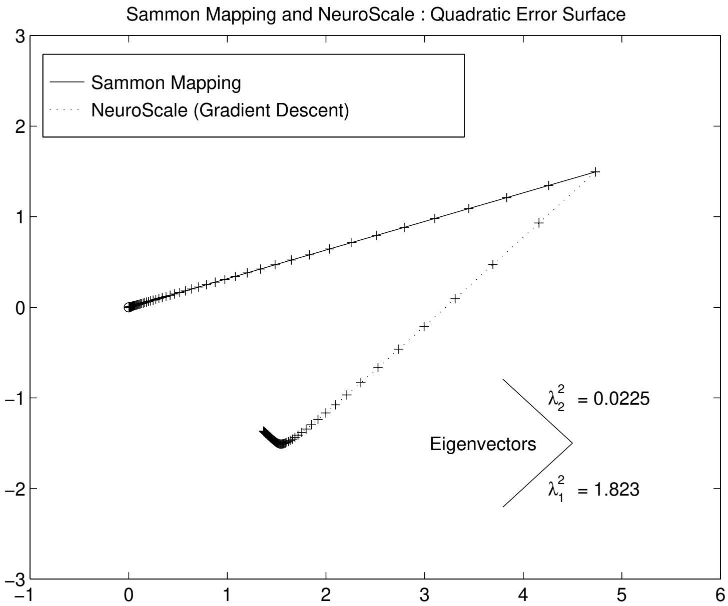

Figure 6.4: Comparison of gradient-descent and Sammon Mapping algorithms on a quadratic error surface.

due to the null-space of $\mathbf{J}$ where the corresponding eigenvalues are zero. However, for multiple outputs, the situation is more complex than this. If the error for a given output dimension l is such that $\mathbf{J}^{\mathrm{T}}\boldsymbol{\delta}_l = \mathbf{0}$, then in general, $\mathbf{J}^{\mathrm{T}}\boldsymbol{\delta}_{l'} \neq \mathbf{0}$, for $l' \neq l$. The output dimensions, l', for which the weight changes are non-zero will cause the the vector $\mathbf{z}_{l'}$, and thus the overall configuration $\mathbf{Y}$, to change, which will in turn result in modification of the inter-point distances d_{ij}. These distances are related to the $\boldsymbol{\delta}$ for *all* output dimensions through equation (6.2), which implies that $\boldsymbol{\delta}_l$ will be perturbed from the null-space and there is thus an 'escape route' from the minimum. The capability of the algorithm to avoid the minimum in such a manner is a side-effect of the existence of a set of multiple solutions which are equivalent, with respect to STRESS, and may be generated by a arbitrary rotation and/or translation of any one single solution.

The effects of the eigenvalues of $\mathbf{J}$ upon the convergence of the relative supervision algorithm suggests that better convergence behaviour may be obtained through the selection of a relatively small value of basis function width, σ, and thus ensuring that $\mathbf{J}$ is well-conditioned. An identical plot to that of figure 6.3, but with $\sigma = 0.01$, is therefore given in figure 6.5.

The distribution of final minimum STRESS values is now very similar to the Sammon mapping given in figure 6.1 earlier. The minimum value is again 0.0028 and 19.8% of runs approach this value. The mean and standard deviation were 0.0052 and 0.0018 respectively. All the local minima observed also now correspond to those observed from Sammon's procedure, with the implication that there are no longer any second level minima effects. Indeed, the condition number of the Jacobian is now 1.0. With $\sigma = 0.1$ it is 53.4, and with $\sigma = 1$, it becomes 4.4×10^7.

The two previous figures illustrate that there is an evident trade-off between levels of local minima which may be controlled by the choice of basis function width. Decreasing σ improves the conditioning of the Jacobian and can moderate the effective attenuation of learning rate that results from the squared-eigenvalue terms in the update equation. However, this in turn permits the procedure to be caught in level-1 local minima, because for a full complement of basis functions, as $\sigma \to 0$, $\mathbf{J} \to \mathbf{I}$, the identity matrix, and gradient-descent NEUROSCALE becomes an exact equivalent of Sammon's algorithm.

It is therefore clear why *decreasing* σ should improve performance of the NEUROSCALE training algorithm with respect to the level-2 minima effects. In the next section, reasons why *increasing* σ improves

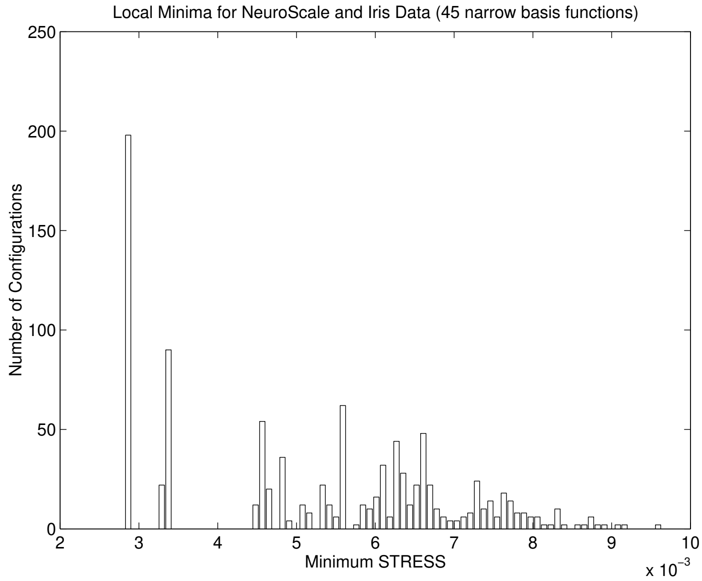

Figure 6.5: Histogram of number of final configurations with corresponding STRESS's for 1000 runs of NEUROSCALE, with $\alpha = 0$, on the `IRIS_45` data set. The width of the 45 Gaussian basis functions was 0.01.

performance with respect to level-1 minima are considered.

6.3 Smoothness of Topographic Transformations

The experiments in the previous section indicated that, for moderate values of basis function width, the NEUROSCALE technique is less plagued by sub-optimal level-1 local minima — those inherent in the Sammon Mapping alone — than is the standard Sammon procedure itself. One potential explanation for this observed behaviour is that the vast majority of such minima found by Sammon's algorithm are highly *unsmooth* transformations of the input data. The basins of attraction for these poor minima in configuration space are likely to be very distant from the configurations attained by random initialisation of the RBF weights with significant values of basis function width.[3]

Further evidence for this hypothesis may be obtained by the following experiment. The standard Sammon Mapping algorithm was run, from random initialisations, 1000 times to generate an equal number of configurations of points with an associated minimised STRESS value. An RBF, with 15 Gaussian basis functions, of width 1.0, whose fixed centres were chosen at random from the data points, was then trained, in the conventional supervised (pseudo-inverse) manner, to generate each of the previous Sammon configurations. If high-STRESS configurations are not easily realisable by the RBF network, then they should give rise to significant residual sum-of-squares error. Figure 6.6 gives this error, plotted against the STRESS minimum, for these 1000 runs. The residual error was averaged over 25 runs of the RBF training process, as the centre selection was different on each run.

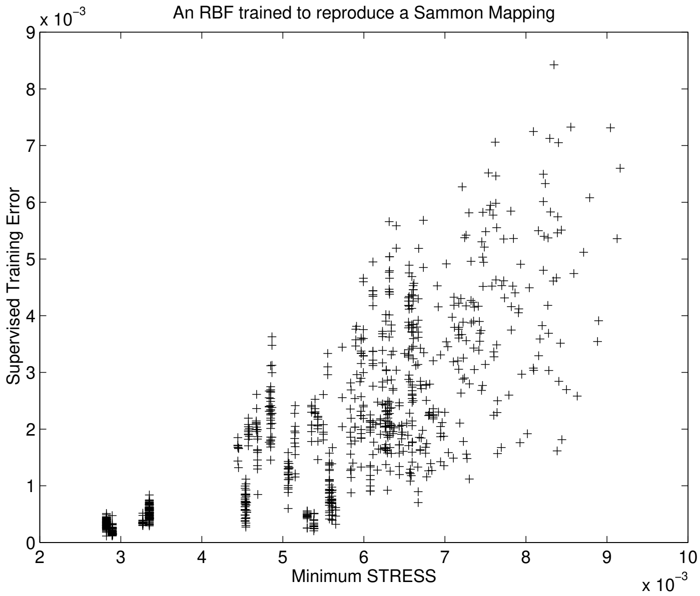

Figure 6.6: Residual error against minimum STRESS for an RBF trained to produce the final configuration of Sammon's procedure *a posteriori*. The RBF comprised 15 basis functions of width parameter 1.0.

The underlying trend in figure 6.6 supports this proposition, as, on average, higher value of minimised STRESS give higher values of residual error for an *a posteriori*[4] RBF transformation.

A more direct measure of smoothness can be obtained by calculating a measure of the *curvature* of the transformation effected by the RBF. That is, some measure of how much the gradient of $\mathbf{y}$ changes

[3] Gaussian basis functions have a width parameter, but, of course, there are other such functions utilised in radial basis function networks — cubics, thin-plate splines etc — that are not parameterised. In such cases there is an implicit global width parameter determined by the scaling of the data.

[4] This terminology will be henceforth adopted for any RBF network that effects a topographic transformation having been trained to reproduce a final configuration of a Sammon mapping, as distinct from being trained via a relative supervision procedure.

with $\mathbf{x}$. One such measure is that given by the functional

$$E_C = \sum_i^N \sum_l^q \sum_m^p \left(\frac{\partial^2 y_{il}}{\partial x_m^2} \right)^2 , \tag{6.11}$$

where i ranges over the patterns, m over the input dimensions and l over the output dimensions. Thus y_{il} is the scalar value of the l-th output node for pattern i, and the overall expression is a measure of the total magnitude of the second derivatives of the output with respect to the input, evaluated at all the input points. Such a functional was used by Bishop [1991, 1993] for the regularisation of neural networks in supervised learning problems.

This measure is evaluated in the graph of figure 6.7, which is otherwise identical to 6.6 with the exception that the vertical axis indicates the value of E_C for each *a posteriori*-trained RBF, rather than the residual error.

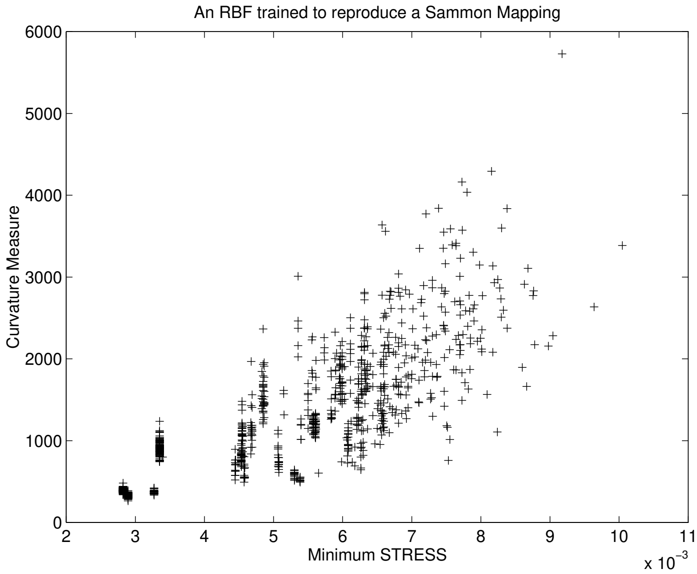

Figure 6.7: Curvature against minimum STRESS for an RBF trained to produce the final configuration of Sammon's procedure *a posteriori*. The RBF comprised 15 basis functions of width parameter 1.0.

Both these plots indicate that higher local minima represent more unsmooth transformations from the data space to the configuration space. While the spread of curvature values increases with increasing STRESS, it is nevertheless clear that low STRESS generally implies lower curvature on average, and with reduced variance in addition.

Further evidence that during the training of NEUROSCALE, curvature generally decreases with STRESS, is presented in figure 6.8. This graph plots the curvature measure E_C at each step of the training algorithm for three different RBF models with 15, 30 and 45 basis functions respectively. After an initial phase where the curvature increases over short sections of the training sequence, the measure decreases and stabilises.

Additional confirmation of this relationship is given for a single NEUROSCALE mapping of the RAE_PCB dataset from Chapter 4 in figure 6.9. In this experiment, a network with 40 Gaussian basis functions was utilised.

The results of figure 6.8 are particularly interesting, as it is evident that the final curvature is not related to the number of basis functions in the transformation. This is contrary to naive expectation, as might arise from experience with conventional supervised training problems, which would assert that the more flexible model implied by a greater number of basis functions would exhibit more curvature. To

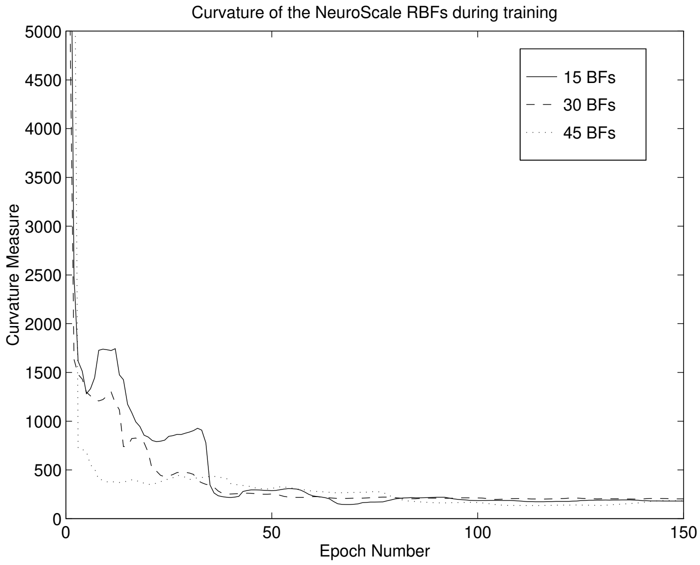

Figure 6.8: Curvature against time during the training of a NEUROSCALE mapping on the `IRIS_45` data, for networks with 15, 30 and 45 basis functions.

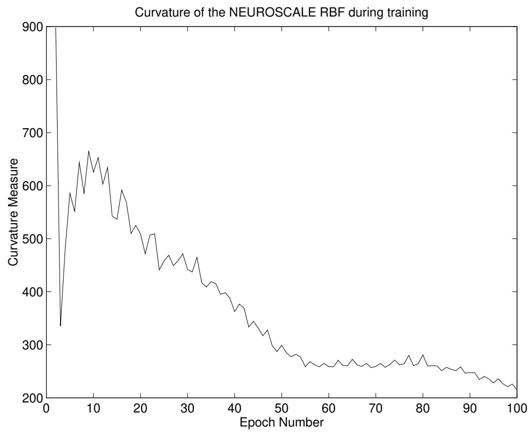

Figure 6.9: Curvature against time during the training of a NEUROSCALE mapping on the `RAE_PCB` dataset.

help understand this apparent anomaly, the relationship between STRESS and curvature is considered more formally in the next section.

6.4 The Relationship between STRESS and Curvature

The plots in figures 6.8 and 6.9 illustrate experimentally that the curvature of the RBF transformation from input space to configuration space generally decreases during the minimisation of STRESS.

To understand how the two quantities may be related, consider the diagram in figure 6.10, which illustrates a very simple example for a transformation from one-dimension to one-dimension. In practice, this is unrealistically trivial, but enables an underlying interaction between STRESS and curvature to be expressed mathematically.

The figure illustrates three input points $\{x_i, x_j, x_k\}$ mapped to three respective output points $\{y_i, y_j, y_k\}$. If any two points, for example the two extreme points $\{x_i, y_i\}$ and $\{x_k, y_k\}$, are correctly positioned, such that $y_k - y_i = x_k - x_i$, then it is intuitive that any error in the placement of the third point, for example $\{x_j, y_j\}$, must imply non-zero curvature at some point in the interval (x_i, x_k), whatever the form of the interpolating transformation.

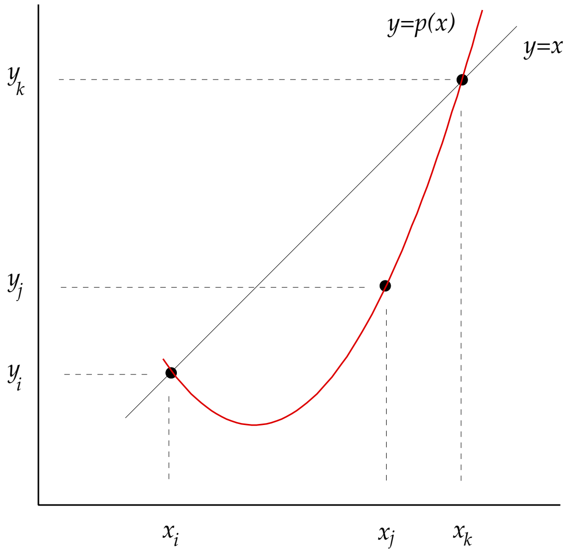

Figure 6.10: A simple example mapping of three points from one-dimension to one-dimension.

This relationship between STRESS and curvature may be demonstrated more formally. Let the mapping from x to y be effected by a RBF transformation, such that $y_i = f(x_i)$ etc. The three input points can be considered to be ordered, such that $x_k > x_j > x_i$, and are sufficiently local that the RBF function $f(x)$ can be approximated by a quadratic polynomial function $p(x) = ax^2 + bx + c$.

The Lagrange polynomial [Plybon 1992] that interpolates the three points is:

$$p(x) = y_i \frac{(x - x_j)(x - x_k)}{(x_i - x_j)(x_i - x_k)} + y_j \frac{(x - x_i)(x - x_k)}{(x_j - x_i)(x_j - x_k)} + y_k \frac{(x - x_i)(x - x_j)}{(x_k - x_i)(x_k - x_j)} \tag{6.12}$$

The curvature of the RBF transformation, as given by the measure of equation (6.11) is $[f''(x)]^2$, which for the quadratic approximation is identical evaluated at each data point and is simply $4a^2$, where a is the coefficient of the x^2 term:

$$a = \frac{y_i}{(x_i - x_j)(x_i - x_k)} + \frac{y_j}{(x_j - x_i)(x_j - x_k)} + \frac{y_k}{(x_k - x_i)(x_k - x_j)}. \tag{6.13}$$

Now, the inter-point distance relationships can be summarised by:

$$d^*_{ij} = x_j - x_i, \tag{6.14}$$
$$d^*_{jk} = x_k - x_j, \tag{6.15}$$
$$d^*_{ik} = x_k - x_i, \tag{6.16}$$

in the input space, and

$$d_{ij} = |y_j - y_i|, \tag{6.17}$$
$$d_{jk} = |y_k - y_j|, \tag{6.18}$$
$$d_{ik} = |y_k - y_i|, \tag{6.19}$$

in the output space.

Substituting the input space inter-point distances into equation (6.13) gives the simplification:

$$a = \frac{y_i(d^*_{jk} - d^*_{ik} + d^*_{ij}) - d^*_{ik}(y_j - y_i) + d^*_{ij}(y_k - y_i)}{d^*_{ij}d^*_{ik}d^*_{jk}}, \tag{6.20}$$
$$= \frac{d^*_{ij}(y_k - y_i) - d^*_{ik}(y_j - y_i)}{d^*_{ij}d^*_{ik}d^*_{jk}}, \tag{6.21}$$

since $d^*_{ij} + d^*_{jk} = d^*_{ik}$.

The above expression can be further simplified by considering that for a structure-preserving mapping, the scaling of the input points is effectively arbitrary, so d^*_{ik} may be set to 1. In addition, it is also necessary to fix two of the output points, or the curvature can be minimised by collapsing them to a single point. Therefore, consider that during optimisation of the overall configuration, it is the case that $(y_k - y_i) = d^*_{ik} = 1$, leaving y_j and thus d_{ij} (or d_{jk}) as free variables. Also, make the assumption that $y_j > y_i$, such that $d_{ij} = (y_j - y_i)$. While this assumption is unrealistic, it will be seen shortly that it is not significant. Given this:

$$a = \frac{d^*_{ij} - d_{ij}}{d^*_{ij}d^*_{jk}}, \tag{6.22}$$

and so

$$E_C = \frac{4}{(d^*_{ij}d^*_{jk})^2}(d^*_{ij} - d_{ij})^2. \tag{6.23}$$

If $y_j < y_i$, then the first expression is no longer valid as indicated previously. However, if $y_j < y_i$ then $y_j < y_k$, which gives $d_{jk} = y_k - y_j$ and so by alternative symmetric derivation:

$$E_C = \frac{4}{(d^*_{ij}d^*_{jk})^2}(d^*_{jk} - d_{jk})^2. \tag{6.24}$$

Equation (6.23) in this simplified and constrained example backs up the intuition that minimisation of STRESS implicitly reduces curvature, as it exactly corresponds to a term from the Sammon STRESS measure, scaled by some constant. As a mapping is generated, and the inter-point distances become approximately correct, the assumptions inherent in the above analysis become realistic, and it would be expected that on *local* scales, curvature would decrease. This notion is intuitively extended to higher dimensional, more realistic, mappings.

There is no such guarantee for *global* distance relationships, which may be exemplified by the `SPHERES_3` dataset. At the point in input space where the outer two spheres are 'opened out', the curvature must be high as neighbouring points one one side of the 'tear' are still nearby in output space, but neighbours on the other side of the partition become relatively highly distant.

Even in the single dimension case considered, if the distances between points in the input space are sufficiently large such that the quadratic approximation to the RBF transformation is invalid, then it is possible for there to be zero STRESS (locally), whilst still non-zero curvature. A simple such example is given for a cubic interpolant in figure 6.11.

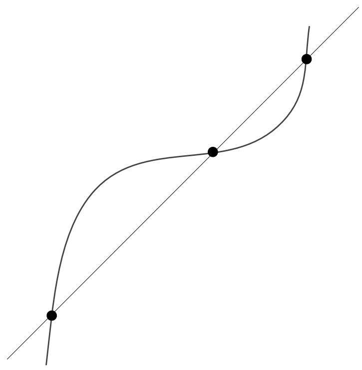

Figure 6.11: A correct mapping of three points from one-dimension to one-dimension, interpolated by a cubic.

The overall result from this analysis and discussion is that, in general, higher STRESS implies higher curvature, while lower STRESS implies lower curvature on local scales for sufficiently smooth interpolating functions. This summary has implications for the desired form of the neural network transformation, which will be considered in Section 6.6 shortly.

In fact, the reduction of curvature during training is an inherent property of networks trained by relative supervision algorithms which is not shared by their architecturally identical, but *a posteriori*-trained, counterparts. This distinction is investigated further in the following section, and its cause subsequently explained in Section 6.6.

6.5 Contrast Between NEUROSCALE and A Posteriori Fitting of an RBF Transformation

It is, of course, possible to produce a transformational Sammon Mapping by generating a configuration of points according to Sammon's standard procedure, and then fitting some model to this *a posteriori*. Indeed, this was the approach adopted by Cox and Ferry [1993] for discriminating future observations where a linear or quadratic model was fitted to a previously generated non-metric multidimensional scaling configuration. This approach could similarly be applied to the training of a neural network model, and it is this method that has been used to generate the previous plot of STRESS against curvature, for an RBF with 10 basis functions, in figure 6.7.

It is revealing to consider a similar *a posteriori* plot for an RBF with 45 basis functions, again with $\sigma = 1.0$, and superimpose a trace of the evolution of curvature of an identical RBF, during training by the relative supervision procedure of NEUROSCALE. This comparison of curvature values is given in figure 6.12. As the curvature of the transformation should be related to the generalisation performance of the networks, a plot of the test error, for the same models, is given in figure 6.13. The test dataset comprised a separate 45 patterns from the full Iris data.

The most significant feature of the two graphs in figures 6.12 and 6.13 is that configurations generated during the training of NEUROSCALE exhibit significantly lower curvature, and hence lower test error and better generalisation, than those configurations, with identical STRESS, generated by *a posteriori* fitting to Sammon mappings. It should be emphasised that that the RBF models in both cases have

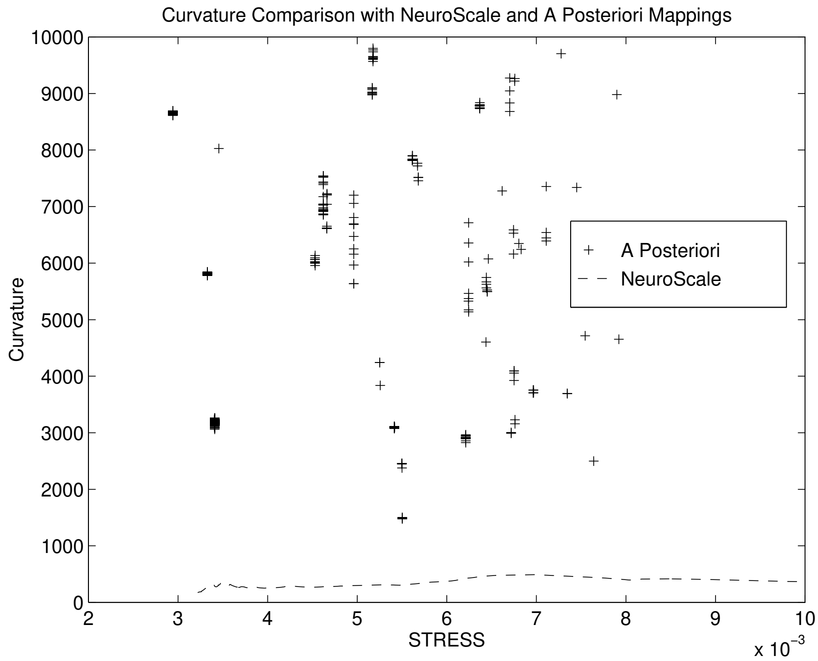

Figure 6.12: Curvature of *a posteriori*-fitted RBF networks compared to that during training of a NEUROSCALE model.

identical architectures and fixed basis functions; only the output layer weights are different.

Consider a NEUROSCALE configuration along the trace of figure 6.12, with STRESS 0.003325. The sum of the squared weights is 79.56. For a similar, *a posteriori* trained RBF, with STRESS 0.003327, this figure is 1.56×10^8 — very considerably greater. It is the smoothness constraint within the RBF model, as a result of the width parameter σ, which causes the ensuing trajectory through configuration space to be markedly different, in terms of curvature, to the corresponding trajectory of Sammon's algorithm — even though the STRESS values are comparable. The underlying cause of this alternative trajectory will be made more explicit in Section 6.6.2 later, where two apparently *identical* configurations, generated by NEUROSCALE and in *a posteriori* fashion, are shown to exhibit dramatically different generalisation errors.

The configurations that NEUROSCALE generates during training are not local minima for Sammon's algorithm, and this may be demonstrated by initialising a Sammon mapping with the final configuration from the above figures, and running that algorithm on the points alone. At each step of the mapping, the configuration is stored and an identical RBF to that of the NEUROSCALE model trained *a posteriori* to reproduce this configuration The trajectory of the networks trained during this Sammon procedure is thus superimposed on figure 6.14, and illustrates how curvature increases dramatically for minimal improvement in STRESS.

The results of this section, particularly figure 6.13, show clearly that training a neural network to perform a topographic transformation by a relative supervision method leads to significantly superior generalisation performance, on the `IRIS_45` data set studied here, than that obtained by the *a posteriori* fitting of an identical network to a Sammon mapping. The reason for this behaviour is quite subtle, and will be revealed in the discussion of regularisation in the next section.

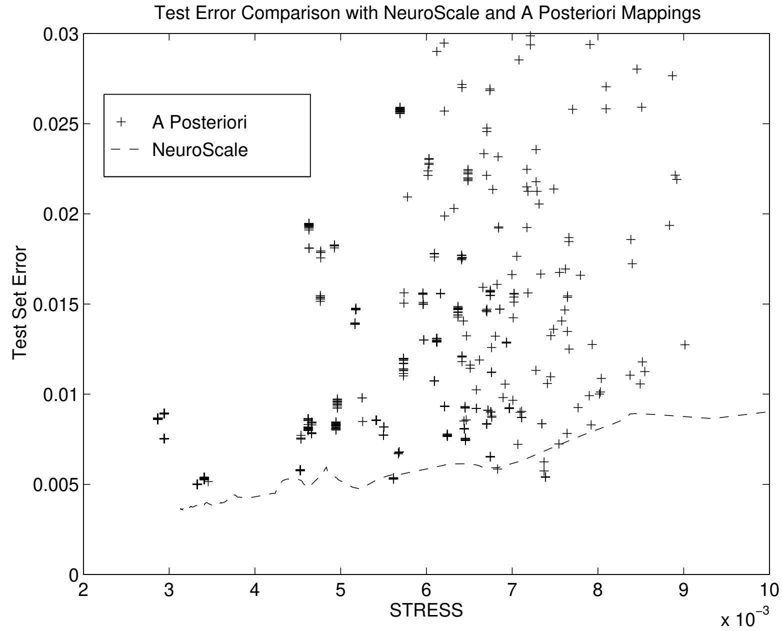

Figure 6.13: Test error of *a posteriori*-fitted RBF networks compared to that during training of a NEUROSCALE model.

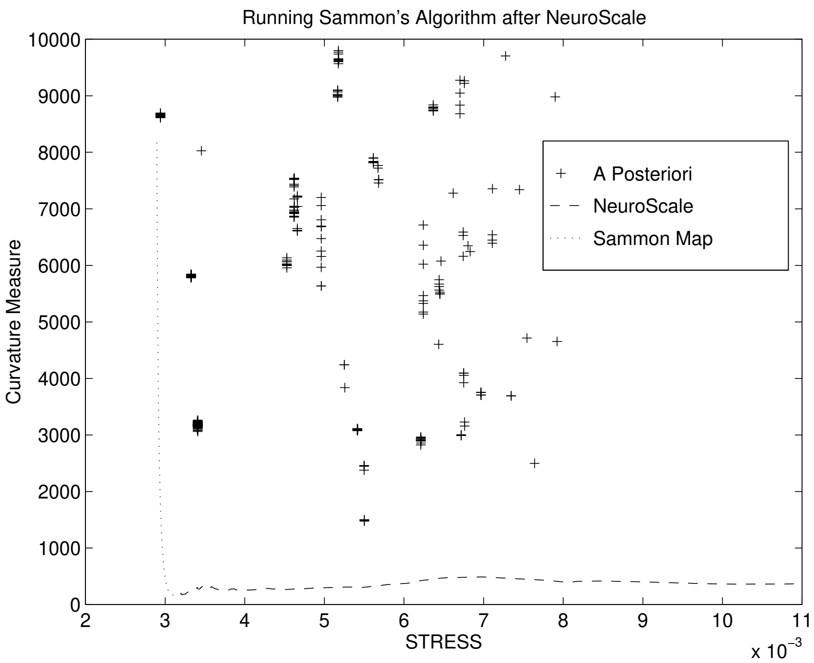

Figure 6.14: A Sammon Mapping, run after initialisation from the final configuration of the NEUROSCALE model. The dashed line illustrates the evolution of curvature and STRESS during NEUROSCALE training. At the minimum, training is stopped and the resultant configuration used as the initial step for a Sammon Mapping. For each step in the optimisation of the Sammon Map, the resulting configuration is reproduced by an RBF, and the network's curvature plotted along with the STRESS of the configuration itself.

6.6 Generalisation and Model Complexity

In many neural network applications, particularly function approximation and interpolation problems, the requirement of good *generalisation* is the salient design issue. In the general case of limited data, which has inherent additive noise, it is necessary to impose some form of *smoothness constraint* on the function. A model that fits all the training data in the presence of noise is likely to generalise, or interpolate, poorly. In the case of a model that is *over-fitted* (fits all the noise), the model's approximation of the function at a given test point will depend critically on the local noise in the training data. In this sense, the interpolation problem is *ill-posed* [Tikhonov and Arsenin 1977], due to this sensitivity to the noise.

There are two principal approaches to imposing smoothness, or limiting complexity, of a neural network model. Firstly, it is possible to reduce the complexity of the network by constraining the number of hidden nodes or basis functions, and this approach is known as *structural stabilisation*. Secondly, some form of penalty term can be introduced to the network error function, such as that given by the measure of equation (6.11) earlier, which discourages high-curvature models. This procedure is termed *regularisation*, and the degree of smoothness in this case is controlled by a hyperparameter. In either approach, the appropriate complexity of network transformation, and thus the number of basis functions or the value of the hyperparameter, will, in general, be unknown at the outset and must be determined from the data alone, often by employing some variety of cross-validation procedure.

6.6.1 Structural Stabilisation

With respect to structural stabilisation, the graph in figure 6.8, which illustrated the evolution of curvature during training for RBFs with 15, 30 and 45 basis functions, suggested that curvature was largely independent of the number of basis functions, indicating that generalisation performance might be likewise insensitive. Indeed, the generalisation error for RBFs trained on the `IRIS_45` dataset and tested on another 45-pattern subset of the full Iris data is illustrated in figure 6.15. The RBFs comprised 5 to 45 basis functions, in steps of 5, and the error values were averaged over 25 runs.

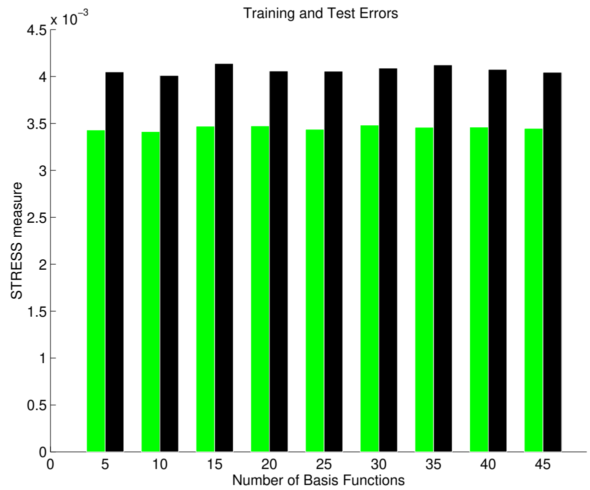

Figure 6.15: Training and test errors for NEUROSCALE RBFs with various numbers of basis functions. Training errors are on the left, test errors are on the right.

From figure 6.15 it can be seen that training and test error are roughly constant across the range of basis functions, 5 to the full complement of 45. There is no evidence of overfitting, and this is consistent with the previous evidence from figure 6.8. Reasons for this behaviour will be developed in the next subsection.

6.6.2 Regularisation

In contrast to the usual function approximation scenario where model complexity is unknown in advance, in the case of NEUROSCALE, and related models effecting such a topographic transformation, it may be reasoned that *the smoothness of the optimal network function is known a priori*.

While any particular arbitrary function with sufficient degrees of freedom may be fitted to a finite set of data and optimally[5] retain the topology (see the cubic in figure 6.11 for an example), it was shown in Section 6.4, in the one-dimensional case, that only a *perfectly smooth* function, with no curvature (or higher order terms), will exhibit optimal generalisation performance. This smoothness constraint must either persist through the entire space, or may be relaxed to only include the subspace containing the data, if this is known. This intuition may be extended to higher dimensions, where it should be apparent that non-zero curvature must imply structural distortion.

For the simple one-dimensional case given earlier in figure 6.10, it is evident that the ideal transformation is of the class:

$$y = \pm x + c, \tag{6.25}$$

where c is an arbitrary constant. The optimality property of this class of functions holds *regardless of the distribution of the data*. It is intuitive in this simple one-dimensional case that any transformation with gradient ± 1 will retain the distance relationships between all possible pairs of data points.

Given that the optimal transforming function has no second-order or higher derivative terms, it is possible to more formally generalise the result for the simple one-dimensional example to higher dimensions, still with $p = q$. This permits an expression for the gradient, or first-order derivatives, of the network function to be obtained. It can be shown that, for exact structure preservation, the following relation must hold at each and every data point:

$$\forall x_m, m \in \{1 \ldots p\} : \quad \sum_{l=1}^{q} \left(\frac{\partial y_{il}}{\partial x_m} \right)^2 = 1. \tag{6.26}$$

That is, considering a point $\mathbf{x}_i$ in input space and a point $\mathbf{x}' = \mathbf{x}_i + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is an arbitrary vector, then the distance between the corresponding image points $\mathbf{y}_i$ and $\mathbf{y}'$ is $\| \boldsymbol{\epsilon} \|$ as required. This can be seen by referring to the Taylor expansions (which contain no second or higher order terms) around the point $\mathbf{x}_i$ of each of the output functions for each dimension:

$$\forall l, l \in \{1 \ldots q\} : \quad y'_l = y_{il} + \boldsymbol{\epsilon}^{\mathrm{T}} \mathbf{g}_{il}, \tag{6.27}$$

where $\mathbf{g}_{il}$ is the gradient vector $(\partial y_{il}/\partial x_1, \ldots, \partial y_{il}/\partial x_p)^{\mathrm{T}}$ evaluated at $\mathbf{x} = \mathbf{x}_i$, and from this the inter-point distances in the output space may be calculated thus:

$$\begin{aligned}
\| \mathbf{y}' - \mathbf{y}_i \|^2 &= \sum_{l=1}^{q} (y'_l - y_{il})^2, &\quad (6.28)\\
&= \sum_{l=1}^{q} (\boldsymbol{\epsilon}^{\mathrm{T}} \mathbf{g}_{il})^2, &\quad (6.29)\\
&= \boldsymbol{\epsilon}^{\mathrm{T}} \left(\sum_{l=1}^{q} \mathbf{g}_{il} \mathbf{g}_{il}^{\mathrm{T}} \right) \boldsymbol{\epsilon}, &\quad (6.30)\\
&= \boldsymbol{\epsilon}^{\mathrm{T}} \mathbf{G}_i \boldsymbol{\epsilon}, &\quad (6.31)
\end{aligned}$$

where the matrix $\mathbf{G}_i = \sum_{l=1}^{q} \mathbf{g}_{il} \mathbf{g}_{il}^{\mathrm{T}}$ and is distinct for every data point. For the corresponding distances to be retained, $\| \mathbf{y}' - \mathbf{y}_i \|^2 = \boldsymbol{\epsilon}^{\mathrm{T}} \boldsymbol{\epsilon}$, and so $\mathbf{G}_i = \mathbf{I}$, the identity matrix, which leads directly to the equalities of equation (6.26) above.

[5]The use of the term 'optimally' here is not intended to imply that all the distances are retained perfectly, but that the optimal Sammon configuration, given the constraint imposed by reducing dimension, can be reproduced by the network.

These gradient relationships can only hold exactly in the unrealistic case where there is no reduction in dimension. In practical applications, where $q < p$, $\mathrm{rank}(\mathbf{G}_i) = q$ and therefore $\mathbf{G}_i$ can never be equal to the identity matrix. In such instances, a low generalisation error implies that the magnitude of the residual $\boldsymbol{\epsilon}^{\mathsf{T}}(\mathbf{I}-\mathbf{G}_i)\boldsymbol{\epsilon}$ should be minimised. In the Sammon mapping, and the NEUROSCALE algorithm, the vectors $\boldsymbol{\epsilon}$ of interest are the combinations of inter-point vectors $(\mathbf{x}_i - \mathbf{x}_j)$. It should therefore be the case that, for low STRESS on the training set, all potential inter-point vectors $\boldsymbol{\epsilon}$ should lie in the range of $\mathbf{G}$. A simple example can be envisaged by considering data distributed over a 2-dimensional plane in a 3-dimensional space, with the neural network transforming the data points down to 2 dimensions. Here, the range of $\mathbf{G}_i$ at all the data points should be the linear subspace spanned by the plane. In figure 3.6 in Chapter 3, excellent generalisation was observed for a single cluster of datapoints taken from a set of four in a 'linear' configuration. Even though the significant Gaussian 'noise' component implies that the data lies in four dimensions, the inter-cluster distances dominate in the STRESS minimisation. The range of the $\mathbf{G}_i$ would thus be expected to include the axial direction along which the clusters are distributed, which would explain the low value of STRESS obtained when the test cluster is incorporated and the intuitively good visualisation evident in the figure.

However, for real data, it is likely to be the case that the minimum STRESS configuration will exhibit curvature at one or more points, with significantly different first derivatives, and thus $\mathbf{G}_i$s, in different regions of the data. The nature of these matrices may be informative, and further experimental inquiry might be appropriate for future research.

The following experiment included here, however, provides some insight into the above analysis and also offers further evidence of the distinction between NEUROSCALE and *a posteriori* mappings. Figures 6.16 and 6.17 illustrate the final configurations of a topographic mapping of the `IRIS_45` dataset from the original four to two dimensions. The mappings have been aligned via a *Procrustes* rotation [Mardia, Kent, and Bibby 1979], and the goodness-of-fit (sum-of-squared distances between corresponding points) is 1.4492×10^{-5}, such that the two configurations may be considered to be identical. Superimposed on the plots is a colour scale which indicates a measure of the conformity of the mapping to the gradient constraint of equation (6.26), and is determined as follows.

For each point in the dataset, the unit vector to every other point, $(\mathbf{x}_i - \mathbf{x}_j)/d^*_{ij}$, was calculated, and the projections of the gradients of both network outputs in the direction of that vector were derived, then squared and summed. The squared deviation of this quantity from its ideal value of 1.0 was then accumulated over every such unit vector. This then gives a measure of the closeness of the gradient measure of equation (6.26) to unity along the directions of interest within the dataset. A simple interpolating surface[6] has been fitted to, and passes through, all the points. This surface is artificial (in that the interpolated values between the data points bear no real meaning) but it does permit a relatively clear visualisation of the expected generalisation accuracy in the output space.

For both figures, an identical radial basis function network comprising 45 fixed centres, located at the data points, was utilised. The basis functions were Gaussian with width $\sigma = 1$, and the results were generated by an identical segment of computer program, with the only difference between the two plots being the weight matrix loaded. Despite the fact that both configurations are similar (apart from reflection, rotation and translation), the NEUROSCALE transformation of figure 6.16 will clearly offer better generalisation performance as the gradient expression is much closer to unity (average error = 0.0027) than for the *a posteriori* mapping (average error = 0.0369). This is confirmed by the values of test error given in table 6.1.

The phenomenon that networks of identical architecture generating identical configurations (with respect to STRESS) exhibit significantly different generalisation performance is again related to the equivalence of solutions under rotation and translation. (This will be considered in more detail in Section 6.7 shortly.) The solution generated by the Sammon Mapping is generally arbitrary, in terms of rotation and translation, and the resulting RBF transformation fitted to this configuration can in turn be expected to exhibit arbitrary generalisation error. However, it can be demonstrated that the relative supervision learning algorithm should lead to better generalisation performance as it *tends to*

[6] This was automatically generated by the software package MATLAB using an inverse-distance method.

	NEUROSCALE	*A Posteriori*
Training STRESS	0.00282294	0.00282294
Test STRESS	0.0037395	0.00866962
Curvature	417.495	11620.4
$\parallel \mathbf{W} \parallel^2$	2.1315×10^7	1.0437×10^8

Table 6.1: STRESS and curvature values for the two gradient mappings.

favour solutions with smaller weights.

Firstly, consider the expression for the update of a single dimension of the network output from Section 6.2.2:

$$\mathbf{z}^{t+1} = \mathbf{z}^t - \eta \mathbf{J}\mathbf{J}^{\mathrm{T}}\delta^t, \tag{6.32}$$

$$= \mathbf{z}^t - \sum_k \lambda_k^2 \beta_k \mathbf{u}_k. \tag{6.33}$$

Now, if initially $\mathbf{z}^0$ is close to the origin, then because of the effect of the λ_k^2 term, the final vector $\hat{\mathbf{z}}$ *will have generally larger projections onto the principal eigenvectors of* $\mathbf{J}\mathbf{J}^{\mathrm{T}}$. So, if $\hat{\mathbf{z}} = \sum_k \gamma_k \mathbf{u}_k$, in general, $\gamma_{k+1} > \gamma_k$. Thus, of all the equivalent solutions, NEUROSCALE favours those $\hat{\mathbf{z}}$ which lie mainly in the principal subspace of $\mathbf{J}\mathbf{J}^{\mathrm{T}}$. A very simple illustration of this effect is given in figure 6.18 for three output points in two dimensions (for which a STRESS=0 solution exists), for full-rank, but ill-conditioned, $\mathbf{J}$. Plotted on the figure are the respective eigenvector components γ_k of $\mathbf{z}^t$ as it evolves during training. The smallest component does not visibly change, while the component in the direction of the principal eigenvector becomes dominant.

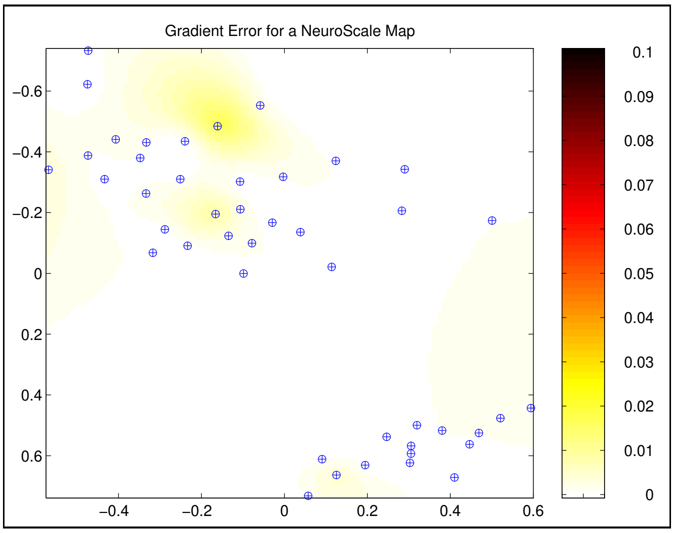

Figure 6.16: Two-dimensional topographic mapping of the IRIS_45 dataset, generated by NEUROSCALE. At each point, the squared deviation of the gradient measure from the ideal value of unity is superimposed with a colour scale.

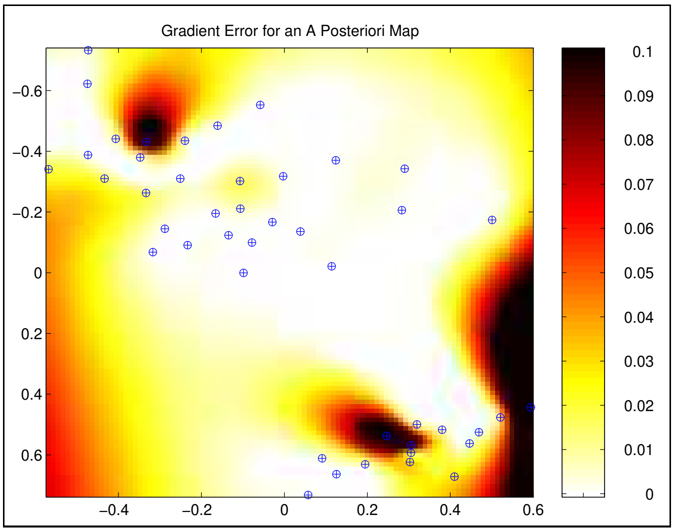

Figure 6.17: Two-dimensional topographic mapping of the IRIS_45 dataset, generated by *a posteriori* fitting of an RBF to a Sammon Mapping. At each point, the squared deviation of the gradient measure from the ideal value of unity is superimposed with a colour scale.

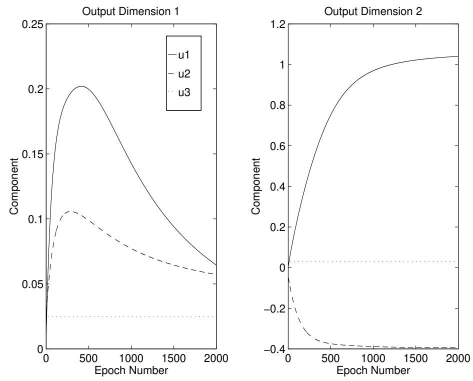

Figure 6.18: Evolution of the eigenvector components of a solution trained by NEUROSCALE.

For the two final configurations of figures 6.16 and 6.17, the values of γ_k^2 are given, for $20 \leq k \leq 45$, in figure 6.19 for both output dimensions. The relative supervision solution clearly exhibits lower values of γ_k for larger k (the minor eigenvectors).

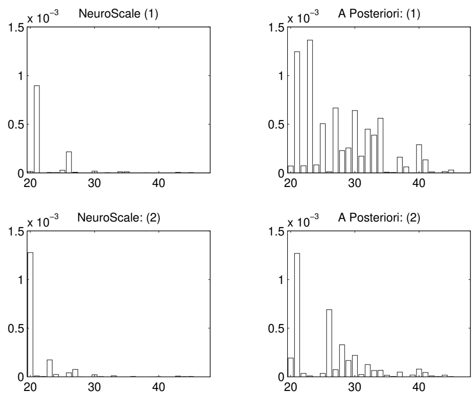

Figure 6.19: Final eigenvector components of the solutions from figures 6.17 and 6.16.

A more dynamic exposition of this effect is given in figures 6.20 and 6.21, again for the `IRIS_45` dataset with full-rank **J**. For the first of the two outputs of the network, the evolution of the (absolute) direction cosines of the vector $\mathbf{z}_1$ with the 45 eigenvectors $\mathbf{u}_k$ is shown for both a Sammon Mapping and a gradient-descent NEUROSCALE network. For the former, no discernable pattern amongst the direction cosines is visible during the 40 training cycles. For NEUROSCALE, the vector $\mathbf{z}_1$ is clearly evolving such that it lies along the direction of the principal axes $\mathbf{u}_1$ and $\mathbf{u}_2$ of **J**, and is approximately orthogonal to all other eigenvectors.

Now, for the single network output solution vector $\hat{\mathbf{z}}$ generated after training with arbitrary rank $\mathbf{J}$,

$$\mathbf{J}\mathbf{w} = \hat{\mathbf{z}}, \quad \text{so} \tag{6.34}$$
$$\mathbf{w} = \mathbf{J}^{+}\hat{\mathbf{z}}, \tag{6.35}$$
$$= \mathbf{J}^{+}\sum_{k}\gamma_k \mathbf{u}_k, \tag{6.36}$$
$$= \mathbf{V}\mathbf{S}^{-1}\mathbf{U}^{\mathrm{T}}\sum_{k}\gamma_k \mathbf{u}_k, \quad \text{by singular value decomposition,} \tag{6.37}$$
$$= \mathbf{V}\mathbf{S}^{-1}\boldsymbol{\gamma}, \tag{6.38}$$

where $\mathbf{S}$ is the diagonal matrix of singular values of $\mathbf{J}$ and $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_h)^{\mathrm{T}}$.

The sum-of-squared weights in $\mathbf{w}$ is:

$$\| \mathbf{w} \|^2 = \mathbf{w}^{\mathrm{T}}\mathbf{w}, \tag{6.39}$$
$$= \boldsymbol{\gamma}^{\mathrm{T}}\mathbf{S}^{-2}\boldsymbol{\gamma}, \tag{6.40}$$
$$= \sum_k \left(\frac{\gamma_k}{s_k}\right)^2, \tag{6.41}$$

and for the complete weight matrix $\mathbf{W}$,

$$\| \mathbf{W} \|^2 = \sum_l \| \mathbf{w}_l \|^2, \tag{6.42}$$
$$= \sum_l \sum_k \left(\frac{\gamma_{kl}}{s_k}\right)^2. \tag{6.43}$$

So, for a given $\sum_{kl}\gamma_{kl}^2$, the sum-of-squared weights is minimised by distributing the 'mass' of γ_{kl}^2 over the largest singular values s_k. However, since $\sum_{kl}\gamma_{kl}^2 = \| \mathbf{Y} \|^2$, then $\sum_{kl}\gamma_{kl}^2$ is minimised when the points $\mathbf{Y}$ are centred at the origin, and this should also serve to reduce the norm on the weights. For the *a posteriori* configuration in figure 6.17, $\| \mathbf{Y} \|^2 = 13.3$, whereas for its NEUROSCALE counterpart in figure 6.16, this measure is 59.9. This implies, from the observations in this chapter, that the tendency of NEUROSCALE to distribute the γ_{kl} in the direction of the principal eigenvectors of $\mathbf{J}$ more than compensates for the overall increase in the squared-magnitudes of those factors.

Thus, the relative supervision algorithm within NEUROSCALE must seek, amongst all candidate solutions of the STRESS optimisation procedure, those with generally lower values of $\| \mathbf{W} \|^2$.

This norm on the weights is effectively a regularisation term, and in a supervised neural network setting, is known as *weight decay* [Hinton 1989]. Such a term, when used as a penalty function in network optimisation, generally reduces the curvature of the final transformation and improves generalisation. This reduction in both sum-of-squared weights and curvature was indeed observed for the example transformations given previously in this section, and listed in table 6.1.

The implication is that topographic transformations generated by the NEUROSCALE algorithms are effectively *self-regularising*, in that they incorporate an implicit weight decay element. Because this regularisation is a side-effect of the operation of relative supervision, there is no control over its magnitude. However, given that it was reasoned previously that topographic transformations should be as smooth as possible, any regularisation component in the algorithm should be beneficial in terms of generalisation performance.

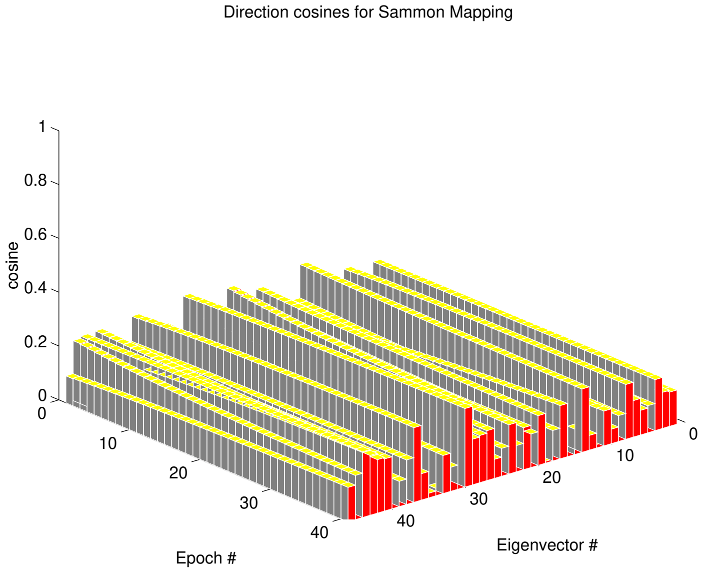

Figure 6.20: Evolution of the direction cosines of one output dimension for a Sammon mapping of the `IRIS_45` dataset.

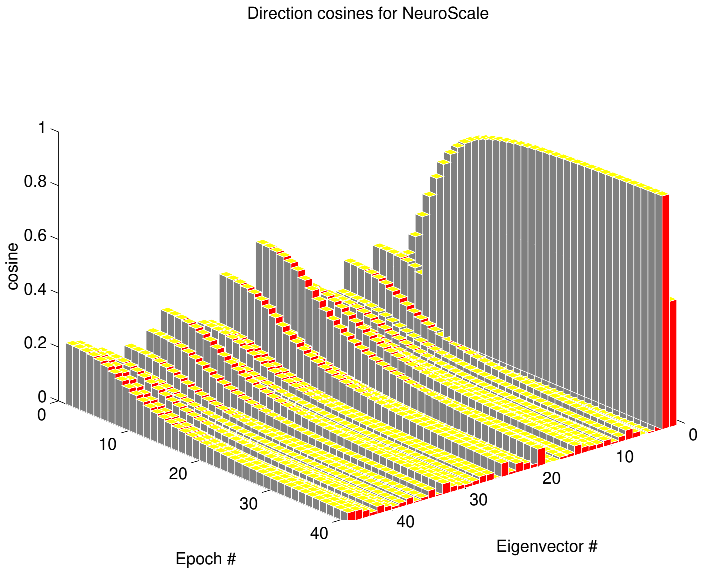

Figure 6.21: Evolution of the direction cosines of one output dimension for NEUROSCALE on the `IRIS_45` dataset.

6.6.3 Effect of Width of Gaussian Basis Functions

In the case of RBF networks as incorporated in NEUROSCALE, the global width parameter, σ, employed by Gaussian basis functions is effectively a smoothing hyperparameter. The larger the value of σ, the smoother the resultant transformation will be. On the basis of the previous discussion in this section, it would be anticipated that optimum generalisation performance should be expected for larger values of σ. There is a caveat, however, in that in the pathological case where the basis function width is infinite, there is no discrimination between any of the data points. In addition, large values of σ would be expected to imply deteriorating performance due to the problems of finite numerical accuracy in the practical computation.

Plots of training and test STRESS against basis function width, for the IRIS_45 data, are given in figures 6.22 and 6.23. The network employed comprised as many basis functions, 45, as patterns. The error values given in these figures were obtained by the 'shadow-targets' training algorithm which is presented in Section 7.3 in the next chapter, as this produces better minima. As was demonstrated in Section 6.2.2, the final *training* STRESS using the standard relative supervision algorithm is dependent on the value of σ. The shadow-targets algorithm enables the minimum realisable STRESS value to be obtained for all (reasonable) values of σ and thus permits a fairer assessment of the effects of σ on *test* STRESS. The contrasts between these two approaches to training NEUROSCALE will be developed further in the next chapter.

In the first plot, the range of σ is from 0 to 100, while the latter plot is a larger scale version of the former, covering the range of values 0 to 20, and in addition, illustrating the values of σ where successive singular values of the matrix $\mathbf{J}$ become numerically zero. [7]

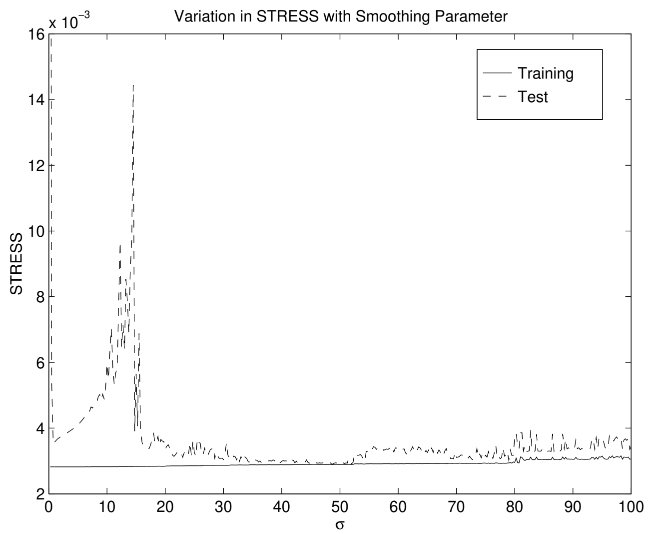

Figure 6.22: Training and test STRESS, for the IRIS_45 dataset, as a function of the basis function parameter σ.

The plot of training STRESS in figure 6.22 demonstrates that this quantity is almost insensitive to the choice of basis function width. There is a gentle increase in value with σ, due to the reduced accuracy of the computer arithmetic. In the case of test STRESS, it is apparent that after an intimal minimum at $\sigma = 1.00$, test STRESS actually *increases* and, after considerable oscillation, decreases to a value comparable with the training error. Figure 6.23 indicates that the test error is smooth up to the point where the smallest singular value of the Jacobian matrix is set to zero in the pseudo-inverse routine.

In general, the form of the test STRESS curve is consistent with the hypothesis that large width should imply better generalisation. It is also evident that test error begins to increase again, slowly, at about

[7] Machine accuracy, ϵ_m, for double precision arithmetic on a SUN Sparc10 workstation is $\approx 2.22 \times 10^{-16}$. Any singular value that is smaller than $\epsilon_m \times$ the largest singular value is discarded in the pseudo-inverse algorithm.

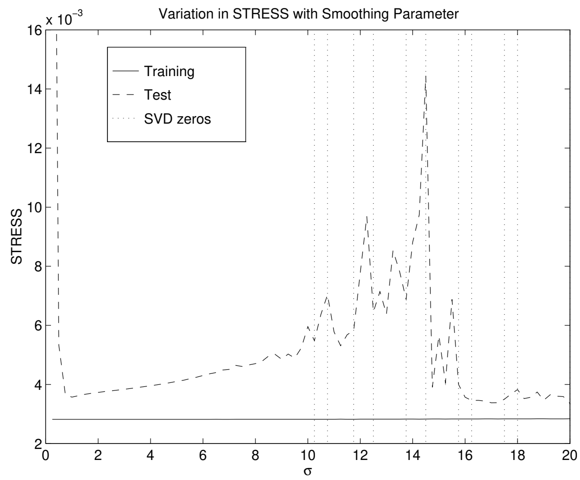

Figure 6.23: Training and test STRESS, for the IRIS_45 dataset, as a function of the basis function parameter σ. The dotted vertical lines indicate points at which successive singular values of the Jacobian become too small to be reliably computed and are thus set to zero.

$\sigma = 50$, and this is likely to be the result of degrading numerical accuracy.

The initial segment of the test error curve, however, exhibits a definite increase in generalisation error. One hypothesis to explain this shape is that this observed maximum may be due to the particular distribution of the Iris data used in the experiment. However, figure 6.24 illustrates similar curves for the topographic mapping of a simple spherical Gaussian cluster in three dimensions down to two.

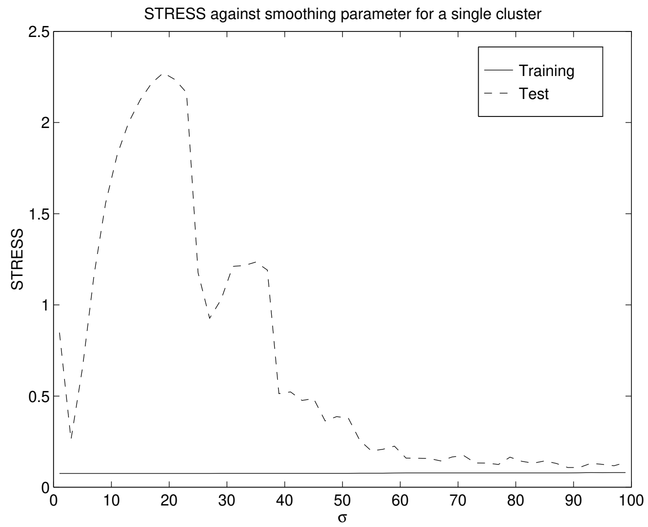

Figure 6.24: Training and test STRESS, for a spherical Gaussian cluster, as a function of the basis function parameter σ.

For the single cluster, best generalisation is again obtained with larger σ. Once more, however, there is a characteristic maximum at a relatively low value of width parameter. The cause of this apparent anomaly remains unknown and warrants further investigation.

6.7 Rotation and Translation of Solutions

It is clear that STRESS is invariant under arbitrary rotation, reflection and and translation of an output configuration $\mathbf{Y}$. What is less obvious, is how these perturbations to the solution effect the weights in the radial basis function network. As has already been noted, NEUROSCALE tends to select a solution which minimises the norm of the weight matrix, so this section will consider how this measure changes under rotation and translation. (Reflection only changes the sign of the weights.)

In both cases, let the base-line solution and associated weights be defined by the equation:

$$\mathbf{J}\hat{\mathbf{W}} = \hat{\mathbf{Y}}, \tag{6.44}$$

implying that $\hat{\mathbf{W}} = \mathbf{J}^{+}\hat{\mathbf{Y}}$.

Rotation

A rotated solution $\mathbf{Y}$ is given by $\mathbf{Y} = \hat{\mathbf{Y}}\mathbf{R}$, where $\mathbf{R}$ is an orthogonal rotation matrix. Then the new weights $\mathbf{W}$ are given by $\mathbf{W} = \mathbf{J}^{+}(\hat{\mathbf{Y}}\mathbf{R})$, and the sum-of-squared weights is:

$$\mathrm{tr}\left[\mathbf{W}\mathbf{W}^{\mathrm{T}}\right] = \mathrm{tr}\left[\mathbf{J}^{+}\hat{\mathbf{Y}}\mathbf{R}\mathbf{R}^{\mathrm{T}}\hat{\mathbf{Y}}^{\mathrm{T}}(\mathbf{J}^{+})^{\mathrm{T}}\right], \tag{6.45}$$

$$= \mathrm{tr}\left[\mathbf{J}^{+}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}(\mathbf{J}^{+})^{\mathrm{T}}\right], \tag{6.46}$$

$$= \mathrm{tr}\left[\hat{\mathbf{W}}\hat{\mathbf{W}}^{\mathrm{T}}\right]. \tag{6.47}$$

Therefore, rotation of the solution does not affect the (Frobenius) norm of the weight matrix.

Translation

For translation, each dimension of the output can be considered in isolation, as shifts along each axis can be applied independently. Then for a single dimension, a translated solution $\mathbf{z}$ is given by $\mathbf{z} = \hat{\mathbf{z}} + k\mathbf{1}$, where k is a scalar determining the amount of translation and $\mathbf{1}$ is the N-vector of 1's. Then the weights $\mathbf{w}$ for that output dimension are:

$$\mathbf{w} = \mathbf{J}^{+}(\hat{\mathbf{z}} + k\mathbf{1}), \tag{6.48}$$

$$= \mathbf{J}^{+}\hat{\mathbf{z}} + k\mathbf{J}^{+}\mathbf{1}. \tag{6.49}$$

$$= \hat{\mathbf{w}} + k\mathbf{j}, \tag{6.50}$$

where $\mathbf{j} = \mathbf{J}^{+}\mathbf{1}$.

Then the norm-squared weights are:

$$\mathbf{w}^{\mathrm{T}}\mathbf{w} = (\hat{\mathbf{w}} + k\mathbf{j})^{\mathrm{T}}(\hat{\mathbf{w}} + k\mathbf{j}), \tag{6.51}$$

$$= \hat{\mathbf{w}}^{\mathrm{T}}\hat{\mathbf{w}} + k^2\mathbf{j}^{\mathrm{T}}\mathbf{j} + 2k\hat{\mathbf{w}}^{\mathrm{T}}\mathbf{j}. \tag{6.52}$$

To minimise this measure, differentiation with respect to k gives:

$$\frac{d}{dk}(\mathbf{w}^{\mathrm{T}}\mathbf{w}) = 2k\mathbf{j}^{\mathrm{T}}\mathbf{j} + 2\hat{\mathbf{w}}^{\mathrm{T}}\mathbf{j}. \tag{6.53}$$

and setting this equal to zero allows a value of k to be determined thus:

$$k = -\frac{\hat{\mathbf{w}}^{\mathrm{T}}\mathbf{j}}{\mathbf{j}^{\mathrm{T}}\mathbf{j}}, \tag{6.54}$$

that minimises the weight-norm, as the second derivative of equation (6.53) is positive. Substituting this value back into equation (6.51) gives the minimum value of the weight norm:

$$(\mathbf{w}^{\mathrm{T}}\mathbf{w})_{min} = \hat{\mathbf{w}}^{\mathrm{T}}\hat{\mathbf{w}} - \frac{\hat{\mathbf{w}}^{\mathrm{T}}\mathbf{j}}{\mathbf{j}^{\mathrm{T}}\mathbf{j}}. \tag{6.55}$$

It is therefore possible for a particular configuration, given the network Jacobian, to determine analytically the translated solution that minimises the norm of the weight matrix. (This is only possible with a linear model where $\mathbf{J}$ is equivalent to the design matrix, and is therefore not possible with an MLP where a nonlinear optimisation approach would be required.) However, it should be noted from equation (6.55) that this minimum value depends upon $\hat{\mathbf{w}}$, which *is* dependent upon rotation. So while an arbitrary rotation does not affect the weight norm of a given configuration, it does alter the class of translated solutions that may be obtained. Furthermore, there is no analytic solution for the optimal $\hat{\mathbf{w}}$, as the weight vectors for each output dimension are mutually dependent. To select the rotation and translation parameters that produce the smallest weight-norm is therefore a difficult nonlinear optimisation problem, but one which in theory could be tackled in order to produce an *a posteriori* mapping with low weight values and, by implication, good generalisation.

6.8 Conclusions

6.8.1 Local Minima

It is known that the generation of Sammon Mappings is considerably complicated by the tendency of the algorithm to be captured in sub-optimal local minima. This is true even for simple problems, which was illustrated for a subset of the Iris data in Section 6.2.1. These poor minima tend to represent *unsmooth* transformations of the input data, as demonstrated in Section 6.3, which is why Gaussian radial basis function networks with significant values of basis function width exhibit very few local minima problems.

In fact, when exploiting such networks to minimise STRESS, *two* distinct classes of weight local minima were discerned. The first class are those weights which give rise to configurations of points that are local minima of STRESS and thus identical to those of the Sammon mapping. The theoretical second class are those introduced by the network itself as a result of the null-space of the Jacobian matrix (the matrix of basis function outputs for every pattern). However, the relative supervision algorithm is not caught by this second class of minima, as there is an escape route offered by the set of multiple rotated/translated solutions.

Nevertheless, NEUROSCALE was observed to effectively cease training before attaining the 'global' minimum. This was shown to be caused by the dynamics of the weight update equation, which dramatically slows the training process when the error vectors are in the direction of the eigenvectors of the Jacobian matrix which have small eigenvalues relative to those of the principal eigenvectors. This behaviour is therefore exaggerated as the condition number of the Jacobian increases.

The conditioning of this matrix is dependent upon the width of the basis functions. A relatively small value of width, σ, will eliminate those problems discussed in the previous paragraph, which become more problematic as σ is increased. However, increasing σ additionally reduces the likelihood of capture in the first class of sub-optimal minima, as the majority of these are unsmooth and thus unrealisable by a network with high σ. There is an evident trade-off here.

6.8.2 Model Complexity

There is a fundamental point to be made concerning the functional form of a neural network effecting a topographic transformation. In Section 6.4 it was shown that in the case of a simple one-dimensional mapping, minimisation of STRESS also implicitly reduces curvature in regions of data where the network transformation can be usefully approximated by a quadratic. This notion can be generalised to higher dimensions, where it is intuitive that for minimum-STRESS configurations, improved generalisation error will be observed when curvature is minimised.

The issue of smoothness is highly relevant to the question of controlling model complexity. A significant result from Section 6.6.1 was that generalisation error for NEUROSCALE was independent of the number of basis functions in the network. In fact, it was argued in Section 6.6.2, that in contrast to many other application domains of neural networks, the smoothness of the desired model in a topographic context is known *a priori*. It was reasoned that the network function should ideally have zero second- and higher-order derivatives, and an expression for the optimal first-order, gradient, terms was derived. The relevance of this result was demonstrated for two apparently identical training-set configurations which nevertheless exhibited considerable differences in generalisation capacity for the test set.

These observations have important implications. In a regression setting, with noise on the target (response) variables, model complexity must be constrained through, for example, regularisation. The best model, in terms of predictions for future data, will generally not fit all the data points. In the topographic context, there are no explicit targets, but the *relative* targets that are generated from the input data itself, can be considered as being *noise-free*. These targets are still precise even in the presence of noise on the input variables.

Noise on the input variables does influence the mapping, and in certain circumstances, dramatically so. A good example of this is the SPHERES_3 dataset, where to minimise STRESS, the outer sphere is 'opened out' where the data is at its least dense on the sphere. Consider, then, data drawn from a near-uniform distribution, such that there is a small region where the data is marginally less dense and where the 'tear' in the sphere naturally forms. It can be seen that the location of the tear can be highly sensitive to added input noise, and the problem of determining the transformation can be interpreted as being *unstable* [Tikhonov and Arsenin 1977]. However, in contrast to a regression problem, this instability is not a result of too complex a model, as the form (curvature) of the topographic transformation will remain similar to the ideal (based on the known input distribution) wherever the tear occurs on the sphere; what is occurring is that the function is effectively 'shifted' in the input space. There is therefore no sense in which regularisation, or other forms of limiting model complexity, can satisfactorily stabilise the problem.

That considered, the SPHERES_3 data is a notably pathological distribution, as the merit of topographically reducing the dimension of data which is near-uniformly distributed in a higher-dimensional space is questionable. It is appropriate, therefore, that the ideal topographic transformation should strictly 'interpolate' all the 'target' data points while being of minimal complexity, or alternatively, of maximal smoothness. This implies that the use of as many basis functions as data points with large values of basis function width (σ) should give optimal performance, as seen in figure 6.22. Nevertheless, the use of fewer, but data-representative, basis functions may still be appropriate for computational reasons, although no significant investigation of this issue has been included in this thesis.

Experimentally, the curvature of the transformation produced by NEUROSCALE was observed to generally decrease during the training process. This is an inherent property of the weight-updating dynamics of the relative supervision algorithm which tends to implicitly reduce the sum-of-squared network weights, and is thus effectively incorporating a *weight decay* component into the training.

However, for supervised training of a neural network, weight decay trades-off the squared-weights penalty term for the error term. The net result of this is that the error on the training set increases from its potential minimum, and the network no longer fits the data points, with the motivation that generalisation to unseen data will be improved. This is *not* the case in NEUROSCALE, where the weight decay component enacts a different rôle. Remember that there is no single unique solution of the mapping process. Any rotation or translation of a particular solution is itself a solution, as it will exhibit the same measure of STRESS. The function of weight decay in the topographic context, therefore, is to select solutions with smaller weights, and by implication, generally lower curvature.

This is of particular relevance when considering *a posteriori* mappings, where a model is fitted, in the standard supervised manner, to the output configuration previously generated by a Sammon Mapping procedure. In Section 6.5, it was illustrated that models fitted in such a manner exhibited higher curvature and poorer generalisation than trained NEUROSCALE networks. The distinction was made very clear in table 6.1, where otherwise identical, but *rotated* and *translated*, configurations exhibited dramatic differences in STRESS on the test dataset. The NEUROSCALE approach was seen to be much superior in this respect. In fact, it was shown in Section 6.7 how the sum-of-squared weights, and by implication the curvature, depends on the rotation and translation of configurations. The preferred values of these two factors could potentially be determined using a nonlinear optimisation approach in order to produce a *a posteriori*-trained RBF that exhibited comparable generalisation to the equivalent NEUROSCALE network. However, the relative supervision algorithm combined with the smoothing of the RBF implicitly achieves a similar end via a single optimisation procedure, and offers the potential to avoid local minima because of the smoothing element.

6.8.3 Objective and Subjective Mappings

It should be emphasised that the above remarks concerning the model complexity are only relevant to purely topographic, or objective mappings, as implemented by NEUROSCALE with $\alpha = 0$. Of particular note is the earlier analysis concerning the gradient and curvature of topographic mappings, as the incorporation of supervisorial influence implies regions of higher curvature at, for example, class boundaries.

Nevertheless, the weight-decay implicit within NEUROSCALE will still be present in the extraction of supervised feature spaces, and thus implies the presence of a smoothing effect even when class information is allowed to influence the mapping. This phenomenon is one hypothesis for the good generalisation observed when using an $\alpha = 0.5$ NEUROSCALE projection as a pre-processing stage in a prediction model in Chapter 4. This possible factor should be considered as part of any future investigation in that direction.