Approximator Networks and the Principle of Optimal Information Distribution

Rüdiger W. Brause

Interner Bericht 1/91

Approximator Networks and the Principle of Optimal Information Distribution

Dr. Rüdiger W. Brause,
University of Frankfurt, FB20 VSFT,
Postbox 11 19 32, D- 6000 Frankfurt 11, FRG.

Abstract

It is well known that artificial neural nets can be used as approximators of any continous functions to any desired degree. Nevertheless, for a given application and a given network architecture the non-trivial task rests to determine the necessary number of neurons and the necessary accuracy (number of bits) per weight for a satisfactory operation.

In this paper the problem is treated by an information theoretic approach. The values for the weights and thresholds in the approximator network are determined analytically. Furthermore, the accuracy of the weights and the number of neurons are seen as general system parameters which determine the the maximal output information (i.e. the approximation error) by the absolute amount and the relative distribution of information contained in the network. A new principle of *optimal information distribution* is proposed and the conditions for the optimal system parameters are derived.

For the simple, instructive example of a linear approximation of a non-linear, quadratic function, the principle of *optimal information distribution* gives the the optimal system parameters, i.e. the number of neurons and the different resolutions of the variables.

1 Introduction

One of the most common tasks of artificial neural nets is the approximation of a given function by the superposition of several functions of single neurons. Similar to the well-known theorem of Stone-Weierstraß (see e.g. [GIR90] for regularization networks) Hornik, Stinchcomb and White have shown [STIN89], [HOR89] that every function can be approximated by a two layer neural network (see figure 1) when a sufficient large number *m* of units is provided. *Sufficient large* - What does this mean? How do we select the appropriate number of neurons for a certain application ?

Fig. 1 A two-layer universal approximation network

Let us consider only the case of one-dimensional output, as it was done in the paper [HOR89]. Analogous results hold for multiple outputs for vector-valued functions.

To give an answer to the questions above, we first have to remark that our standard modelling of artificial neural nets do not reflect an important feature of reality: the descreteness of all real valued events. Contrary to the modelling of synaptic weights and neuronal activity (spike-frequency) by real numbers, there *do not exist real numbers in reality*.

Instead, there exist a kind of noise and unprecise operations which give rise to a certain amount of error in all real world systems. Especially in simulations and implementations of neural nets we replace all real numbers by more or less fine-grained physical variables, e.g. counters or other descrete variables, with a finite error. This concept is consistent with the restriction of "finite information" in our system: the information of a variable x is defined by

$$I(x_i) := - \mathrm{ld}\,(P\{x_i\}) \qquad [\text{ Bits}] \qquad \textit{Information} \qquad (1.1)$$

If all states x_i are equiprobable, the information is the logarithm of the number of possible states. For a real number, the number of different values x_i is infinite. Thus, if we have no *a priori* knowledge about the occurence of the states and we have therefore to assume an uniform, non-vanishing probability distribution for them, a real number has an infinite amount of information. This argument is also valid for the averaged information, the entropy, introduced by Shannon [SHA49]

$$H := \langle I(x)\rangle = -\Sigma_i\, P_i\, \mathrm{ld}P_i = - \int p(x)\, \mathrm{ld}\, p(x)\, dx \qquad (1.2)$$

which also becomes infinity for an uniform distribution p(x) = 1/d over the whole range of the real variable

$$\lim_{d\to\infty} H(d) = \lim_{d\to\infty} - \int_{-d/2}^{+d/2} c\ \mathrm{ld}\ c\ dx = \lim_{d\to\infty} - \mathrm{ld}\ 1/d = \infty$$

Because all systems deal with finite amounts of information, there are no "real" real numbers used in neural systems; all weights have a distinguishable number of states (at least due to quantum physics) and therefore contain a certain amount of information in the sense of the above definition (1.1).

2 Optimal information distribution

Let us now regard an approximation $\hat{f}$ for the function f: $\Re^n \ni \mathbf{x} \rightarrow f(\mathbf{x}) \in \Re$. For example, this can be done by the two-layer neural network of figure 1. Let the positive root of the maximal quadratic error of this approximation be d_f with

$$d_f^2 = (f(\mathbf{x})-\hat{f}(\mathbf{x}))^2 \qquad (2.1)$$

Then we can regard the error as a kind of discretization error $\pm d_f$. Denoting the complete value range with $V_f := |f_{max}- f_{min}|$ we can conclude that there are only V_f /d distinguishable, fixed states of the variable f which differ by an increment of $d=2d_f$. All other states are undistinguishable from deviations of the fixed states.

Thus, unless we do not know anything more about the input distribution of $\{\mathbf{x}\}$ and therefore nothing more about the error distribution, the output has minimal

$$I_{out} = \text{ld}\,(V_f /d_f) \qquad (2.2)$$

bits information. Another parameters, which determine the error of the approximation, are on the one hand the resolution of the weights or its information content

$$I_w = \text{ld}\,(V_w/d_w) \qquad (2.3)$$

with the weight increment d_w and on the other hand the number *m* of neurons.

Certainly, when we increase the number of neurons and the number $I_w = \Sigma_i I_{wi}$ of bits per neuron the approximation will become better and the error will decrease. Nevertheless, for a certain system with a finite amount of information storage capacity (such as a digital computer) the question arises:

What is the best distribution of the information, i.e. what is the best choice for *m* and I_w

1) either to get the minimum approximation error d_f, using a fixed amount of information or
2) to use the minimal amount of information for a fixed error ?

Neither one neuron with high-resolution weights nor many neurons with one bit weights will give the optimal answer; the solution is in between the range. Let us denote the parameters *m*, I_w, ... as general system parameters c_1, ..., c_k.

2.1 The principle of optimal information distribution

Let us first get the conditions for the optimal system parameters by some plausible considerations, presented in [BR89]. The rigid mathematical approach will be covered by the next section 2.2.

Assume on the one hand that we transfer a fixed, small amount of information from one parameter to another and we will find the maximal output information I_{max} increasing by decreasing the approximation error. In this case the information distribution induced by the parameter values of c_1, ..., c_k was not optimal; the new one is better. Let us assume that on the other hand we find that

the output information has decreased, then the information distribution is not optimal, too; by making the inverse transfer we can also increase I_{max}.
These considerations lead us to the following extremum principle:

> In an *optimal information distribution* a small (virtual) change in the distribution (a change in c_1, ..., c_k) neither increases nor decreases the maximal output information.

A small increment of additional information δI_{sys} in the system will produce a change δI_{out} in the minimal output information

$$\delta\, I_{out} = \delta I_{sys} \frac{\partial}{\partial I_{sys}} I_{out} = \delta I_{sys} \sum_{i=1}^{k} \frac{\partial}{\partial c_i} I_{out}(c_1, ..., c_k) \frac{\partial c_i}{\partial I_{sys}} \qquad (2.4)$$

Each term in the sum of equation (2.4) represents an information contribution of a system parameter when we increase the overall system information I_{sys}. According to the principle above, an optimal distribution is given when all terms in the sum i.e. all information contributions of the system parameters are equal.
With the definition (2.2) we get for each term of the sum of (2.4)

$$\frac{\partial}{\partial c_i} I_{out}(c_1, ..., c_k) = \frac{\partial}{\partial c_i} (\mathrm{ld}\,(V_f) - \mathrm{ld}(d)) = -\frac{1}{d} \frac{\partial d}{\partial c_i} = -\frac{1}{d_f} \frac{\partial d_f}{\partial c_i} \qquad (2.5)$$

and so the optimal distribution resides when

$$\frac{\partial d_f}{\partial c_1} \frac{\partial c_1}{\partial I_{sys}} = \;...\; = \frac{\partial d_f}{\partial c_k} \frac{\partial c_k}{\partial I_{sys}} \qquad (2.6)$$

is fulfilled. The *k* independant terms gives us (k-1) equations fo *k* variables c_1, ..., c_k, leaving us with a degree of freedom of one. So, choosing the amount of available information storage $I_{sys}(c_1, ..., c_k) := I_0$, the parameters c_1, ..., c_k are fixed and with I_{max} the smallest error d_f for the particular application will result. On the other hand, for a certain maximal error a certain amount of network information is necessary.

2.2 Optimal system parameters

Now we want to compare the principle above to a more conventional mathematical approach.

The minimal information I_{out} introduced above is a function with multiple parameters $I_{out}(c_1,...,c_k)$. If we want to get the maximal information out of the system using only a certain amount of system information we look for an optimal parameter tupel $(c_1^*,...,c_k^*)$ so that

$$I_{out}(c_1^*,...,c_k^*) = \max_{c_1,...,c_k} I_{out}(c_1,...,c_k) \qquad (2.7)$$

which is accompanied by the restriction that the whole information I_{sys} in the system should not be changed during the maximization process

$$I_{sys}(c_1,...,c_k) = I_0 = \mathrm{const} \qquad (2.8)$$

By these two conditions the relative maximum (2.7) of the function I_{out} with multiple parameters is searched under the restriction of (2.8). The standard method to solve a problem like this is the method of Lagrange multipliers. For this purpose let us define the differentiable functions

$$I(c_1,...,c_k) := I_{sys}(c_1,...,c_k) - I_0 = 0 \qquad (2.9) \qquad \text{and} L(c_1,...,c_k,\lambda) := I_{o}$$

Since the Lagrange function includes the restriction, the necessary conditions for a relative maximum of the Lagrange function gives us the optimal values for the system parameters

$$\begin{aligned} &\frac{\partial}{\partial c_1} L(c_1^*) = 0 \\ &... \\ &\frac{\partial}{\partial c_k} L(c_k^*) = 0 \\ &\frac{\partial}{\partial \lambda} L(\lambda^*) = 0 \end{aligned} \qquad (2.10)$$

The conditions above transform to the equations

$$\begin{aligned} &\frac{\partial}{\partial c_1} I_{out}(c_1^*) + \lambda \frac{\partial}{\partial c_1} I(c_1^*) = 0 \\ &... \\ &\frac{\partial}{\partial c_k} I_{out}(c_k^*) + \lambda \frac{\partial}{\partial c_k} I(c_k^*) = 0 \end{aligned} \qquad (2.11a)$$

$$I(c_1^*,...,c_k^*) = 0 \qquad (2.11b)$$

Let us assume that the function $I(c_1,...,c_k)$ is invertible for each system parameter. Then we know that

$$\frac{\partial}{\partial c_i} I(c_i) = \frac{\partial}{\partial c_i} I_{sys}(c_i) = \left[\frac{\partial c_i}{\partial I_{sys}(c_i)} \right]^{-1} \qquad (2.12)$$

and the conditions (2.11) become

$$\begin{aligned} &\frac{\partial}{\partial c_1} I_{out}(c_1^*) \frac{\partial c_1}{\partial I_{sys}} = - \lambda \\ &... \\ &\frac{\partial}{\partial c_k} I_{out}(c_k^*) \frac{\partial c_k}{\partial I_{sys}} = - \lambda \end{aligned} \qquad (2.13a)$$

$$I_{sys}(c_1^*,...,c_k^*) = I_0 \qquad (2.13b)$$

The equations (2.13a) say that for the necessary condition of an optimal information distribution all the terms on the left hand side of (2.13a) should be equal: This is the *principle of optimal information distribution* as it is stated above in section 2.1 and expressed in equation (2.6). The last condition (2.13b) is just our well-known restriction (2.8).

3 Application examples

In this section first we want to demonstrate the procedure above by a very simple example: the approximation of a quadratic form by a polygone. Throughout in this example, all design decisions (choice of value ranges etc.) are taken for demonstration purposes only.

3.1 The approximation of a simple non-linear function

Let us consider the simple non-linear function $f(x) = ax^2 + b$. The approximation of this function can be accomplished by a network with one input x shown in figure 2.

Fig. 2 The network for approximating $f(x) = ax^2 + b$ and the unit output function

Another version of the quadratic function is the *logistic function* $x(t+1)= f(x) := ax(1-x) = ax-ax^2$ which yields deterministic chaotic behavour in the interval [0,1] for some values of *a* [BAK90]. This system can be approximated by the network of figure 2, using an additional, direct input $y_{m+1}:=x$ for the second layer to model the linear term ax of the logistic function. The learning of the weights and thresholds by the Backpropagation-Algorithm was demonstrated by Lapedes and Farber [LAP87].

Let us return to our example of the quadratic function $f(x) = ax^2 + b$. Each neuron of the network of figure 2 has the output function $y_i = S(z_i)$ with the activation function z_i

$$z_i = \Sigma_j \, w_{ij} \, x_j \qquad (3.1)$$

which becomes for the first layer

$$z_i = w_i \, x + t_i \qquad \text{with the threshold } t_i \qquad (3.2)$$

and for the second layer

$$\hat{f}(x) = \Sigma_i \, W_i \, S(z_i) \quad + T \qquad (3.3)$$

Let us assume that we use simple limited linear output functions as squashing functions

$$S(z_i) = \begin{cases} 1 & 1 < z_i \\ z_i & 0 \leq z_i \leq 1 \\ 0 & z_i < 0 \end{cases} \qquad (3.4)$$

The definition (3.4) satisfy the conditions S(∞)=1, S(-∞)=0 of [HOR89] and is shown in figure 2 on the right hand side.

Let us assume that all the weights have converged by a proper algorithm for an approximation of the non-linear function by linear segments. Since the output of each neuron is only linear when x is from its intervall $[x_i-\Delta x/2,\ x_i+\Delta x/2]$ with $x_i=x_0+i\Delta x-\Delta x/2$ and otherwise it is constant 0 or 1, a sufficient condition for the linear approximation is given if the whole input intervall $[x_0,x_1]$ is divided by the *m* neurons of the first layer into *m* intervalls Δx for the approximation. The segmented normalized variable $z_i \in [0,1]$ is 1/2 for x_i. In the second layer it is then weighted by W_i. Together with an offset of the previous inetervalls it represents there the approximation function $\hat{f}(x)$ in the intervall $[x_i-\Delta x/2,\ x_i+\Delta x/2]$.

$$\hat{f}(x) = \sum_{i=1}^{m} W_i \; S(z_i) + T = \sum_{i=1}^{k-1} W_i \; + W_k \; S(z_k) + T \qquad (3.5)$$

The resulting approximation for m=5 neurons is shown in figure 3.

Fig. 3 The non-linear function and its approximation by 5 neurons

The corresponding values for w_i, t_i, W_i and T can be easily calculated.
From the conditions of (3.4) we can conclude

$$z\,|_{x_i-\Delta x/2} = 0 \qquad\qquad z\,|_{x_i+\Delta x/2} = 1$$

and by (3.2) we get

$$w_i = 1/\Delta x = m\,/\,(x_1-x_0) \qquad (3.6a)$$

and

$$t_i = -\,w_i\,(x_i-\Delta x/2) = x_0/\Delta x + 1 - i = -\,mx_i\,/\,(x_1-x_0) + 1/2 \qquad (3.6b)$$

Let us choose the weights W_i of the second layer such that in each segment the spline is the tangent of f(x) in x_i

$$\frac{\partial f(x_i)}{\partial x} = \frac{\partial (ax^2+b)}{\partial x}\Big|_{x_i} = 2ax_i := \Delta y/\Delta x$$

Since the output S(z) is normalized between 0 and 1, the weights W_i are the normalized tangent $\Delta y/1$. Therefore,

$$W_i := \Delta y/1 = 2ax_i\,\Delta x \qquad (3.6c)$$

Then the basic threshold T becomes the offset of the approximation at x_0, see figure 3. Using equation (A1.1) of appendix A1 we get

$$T = f(x_0) - d^{lin} = ax_0^2 + b - a/2\,(\Delta x/2)^2 \qquad (3.6d)$$

Example:

For a net of m:=5 neurons we get for a=1, b=0 with $\Delta x = 0.4$ five non-overlapping intervals

[-1,-.6],[-.6,-.2],[-.2,+.2],[+.2,+.6],[+.6,+1]

and x_i= {-.8,-.4, 0, +.4, +.8}, W_i={-.64, -.32, 0, +.32, +.64},

w_i = 2.5, t_i= {+2.5, +1.5, +0.5, -0.5, -1.5}, T= 0.98.

The maximal approximation error d_{lin}=0.02 has the same order as in the simulation results of Lapedes and Farber [LAP87].

In figure 4 the superposition of the approximating function by the individual neural output $S_i(x)$ is shown. Each neuron has its linear output restricted to its input interval, otherwise it remains constant.

Fig. 4 The individual neural approximations for a=1, b=0, m=5

Due to figure 3 (and figure A1.1) we might suppose that the error of the approximation do not remain constant, but has minimal and maximal values. This is confirmed in figure 5 for the example of m=5 neurons.

Fig. 5 The linear approximation error in the interval $x \in [-1,+1]$ for m=5 neurons

In real-world applications we are not interested in the mean error over the interval (which is approximately zero in the example above), but in the maximal error that can occur. Thus, we aim not to minimize the average error of the approximation, but to minimize the maximal error. As the error of the linear approximation we consider therefore the maximal linear approximation error d_{lin}^{max} which is evaluated in appendix A1 to

$$d_{lin}^{max} = a/2(\Delta x/2)^2 \tag{A.1.1}$$

This reflects the error due to the finite number of neurons. Let us now consider the other source of the approximation error, the finite information in the weights and thresholds, i.e. the error due to the finite resolutions of the system variables.

3.2 The resolution error

To calculate the information after (2.3) for w_i, t_i, W_i and T we have to define first the range V_w,V_t,V_W and V_T of the variables. For the sake of simplicity, let us assume that the value ranges and the information content of all variables are independant of the index *i*. Since the variables w and T are constant they might be implemented in read-only-memory (ROM) with $\min(w_i)=0=\min(T)$ and thus by equations (3.6a,b,c,d) we have

$$\max(w_i)-\min(w_i) = \; V_w: = w_i = m/(x_1-x_0) = (\Delta x)^{-1} \tag{3.7a}$$

$$\max(t_i)-\min(t_i) = \; V_t = (-mx_0/(x_1-x_0) + 1/2) - (-mx_1/(x_1-x_0) + 1/2) = m \tag{3.7b}$$

$$\max(W_i)-\min(W_i) = V_W = 2a(x_1-x_0)\Delta x = 2a\,(x_1-x_0)^2/m \tag{3.7c}$$

$$\max(T)-\min(T) = \; V_T : = ax_0^2 +b \; - a/2\,(\Delta x/2)^2 \tag{3.7d}$$

The maximal resolution error δ of a variable in one state is just the half of the resolution increment of equation (2.3)

$$\delta = d/2 \; = V/2 \;\; 2^{-I} \tag{3.8a}$$

and therefore

$$\delta w = (V_w/2)\, 2^{-I_w} \; = (m/(x_1-x_0)2) \;\; 2^{-I_w} \tag{3.8b}$$

$$\delta t \; = 1/2\, m\, 2^{-I_t} \tag{3.8c}$$

$$\delta W = a(x_1-x_0)^2/m \;\; 2^{-I_W} \tag{3.8d}$$

$$\delta T \; = a/2\,(x_0^2+b/a - 1/2\,[(x_1-x_0)/(2m)]^2)\, 2^{-I_T} =: a/2\, g_T(m)\, 2^{-I_T} \tag{3.8e}$$

In the present approximation function example our information distribution system parameters c_1, ...,c_k are the number of bits per variable I_w,I_t,I_W and I_T and the number *m* of neurons in the first layer. In appendix A2 the error d_{res}^{max} due to the finite resolutions I_w,I_t,I_W, I_T and *m* is evaluated to

$$d_{res}^{max}= 2ax_1\, \Delta x\, [\delta w\, x_1+\delta t] + m\delta W + \delta T \tag{A2.2}$$

3.3 The optimal information distribution

As it was already mentioned, we are not interested in minimizing the average error, but the maximal error of the approximation. Besides, since we do not assume anything about the input probability distribution p(x), we can not compute the average error.

The maximal approximation error is given by the worst case condition that the linear approximation error d_{lin} and the resolution error d_{res} do not compensate each other but adding up to

$$d_f^{max} = d_{lin}^{max} + d_{res}^{max} \tag{3.9}$$

The whole information I_{sys} contained in the network is the sum of the information $m(I_w+I_t)$ of the *m* weights and thresholds in the first layer and the information mI_W+I_T of the *m* weights and the threshold in the second layer

$$I_{sys} = m(I_w+I_t+I_W) + I_T \tag{3.10}$$

When we add some information to the system by augmenting the number *m* of neurons, the resulting approximation will be better and, naturally, the approximation error will diminuish. When we add some neurons, but reduce the information in the weights and threshold, such as to conserve the overall system information, the result is not so clear. In figure 6 the approximation error is shown for different values of *m* and constant system information I_{sys}=708 bits; the number of bits for all other variables are the same $I_w=I_t=I_W= I_T$ and can be directly computed by equation (3.10).

Fig. 6 The approximation error at constant system information (a=1, b=0)

The minimal error of $d_f^{max}=2.28\text{x}10^{-3}$ is at m*=16.2 neurons and I_T=14.2 bits, about 3% worse than with the optimal system parameters (see example ahead). To get the optimal parameters, we just have to compute the conditions for the multi-dimensional minimum of $d^{max}(m,I_w,I_t,I_W,I_T)$ which we have already solved in section 2.1 and 2.2.

The condition (2.6) for an optimal information distribution becomes

$$\frac{\partial}{\partial m}\left(d_{lin}^{max} + d_{res}^{max}\right)\left(\frac{\partial I_{sys}}{\partial m}\right)^{-1} = \;\ldots\; = \frac{\partial}{\partial I_T}\left(d_{lin}^{max} + d_{res}^{max}\right)\left(\frac{\partial I_{sys}}{\partial I_T}\right)^{-1} \tag{3.11}$$

with the derivatives

$$\frac{\partial I_{sys}}{\partial m} = I_w+I_t+I_W \qquad \frac{\partial I_{sys}}{\partial I_w} = m = \frac{\partial I_{sys}}{\partial I_t} = \frac{\partial I_{sys}}{\partial I_W} \qquad \frac{\partial I_{sys}}{\partial I_T} = 1 \tag{3.12}$$

This gives us **5 terms** which are all equal. Let us evalutate the first term.

With (A1.2) we have $\frac{\partial}{\partial m} d_{lin}^{max} = \frac{\partial}{\partial m} a/8\ (x_1-x_0)^2\, m^{-2} = -\frac{a}{4m^3}(x_1-x_0)^2$ (3.13)

and with (A2.3) we have $\frac{\partial}{\partial m} d_{res}^{max} = \frac{a\,(x_1-x_0)^2}{8\ m^3}\ 2^{-I_T}$ (3.14)

Therefore, the expressions (3.14) and (3.13) together with (3.12) yields the first term of the equations in (3.11)

1) $$\frac{\partial d_f^{max}}{\partial m}\left(\frac{\partial I_{sys}}{\partial m}\right)^{-1} = -\frac{a}{4m^3}\,(x_1-x_0)^2\,[1 - 2^{-I_T}/2\,]\,(I_w+I_t+I_W)^{-1}$$

All the other system parameters I_w,I_t,I_W,I_T do not influence the linear approximation error d_{lin}^{max}. Therefore, the derivation of the error (A1.2) is zero and we get the terms

2) $$\frac{\partial d_f^{max}}{\partial I_w} \left(\frac{\partial I}{\partial I_w^{sys}}\right)^{-1} = 2ax_1^2 \Delta x \frac{\partial \delta w}{\partial I_w} m^{-1} = -2\frac{x_1^2 a}{m^2}(x_1 - x_0) \ln(2) \delta w$$

because $$(\delta w)^{-1} \frac{\partial \delta w}{\partial I_w} = \frac{\partial}{\partial I_w} \ln(\delta w) = \frac{\partial}{\partial I_w} (\ln(V_w) - I_w \ln(2)) = -\ln(2)$$

3) $$\frac{\partial d_f^{max}}{\partial I_t} \left(\frac{\partial I}{\partial I_t^{sys}}\right)^{-1} = 2ax_1 \Delta x \frac{\partial \delta t}{\partial I_t} m^{-1} = -2\frac{x_1 a}{m^2}(x_1 - x_0) \ln(2) \delta t$$

4) $$\frac{\partial d_f^{max}}{\partial I_W} \left(\frac{\partial I}{\partial I_W^{sys}}\right)^{-1} = \frac{\partial m \delta W}{\partial I_W} m^{-1} = -\ln(2)\, \delta W$$

5) $$\frac{\partial d_f^{max}}{\partial I_T} \left(\frac{\partial I}{\partial I_T^{sys}}\right)^{-1} = \frac{\partial \delta T}{\partial I_T} = -\ln(2)\, \delta T$$

All the five terms should be equal to yield an optimal information distribution. Let us evaluate the equalities.

With **term 2) = term 3)**, we know that

$$x_1\, \delta w = \delta t \qquad (3.16)$$

The resolution errors of the weights and the threshold of the first layer should be in the same order since they produce the same final error by multiplication with W.

The equation (3.20) gives us with (3.8b) and (3.8c)

$$x_1 m/(x_1 - x_0)\, 2^{-I_w} = m\, 2^{-I_t}$$

$$\mathrm{ld}(\, 2^{I_t}) = \mathrm{ld}[(x_1 - x_0)/x_1] + \mathrm{ld}(2^{I_w})$$

$$I_t = I_w + C_1 \qquad \text{with } C_1 := \mathrm{ld}((x_1 - x_0)/x_1) \qquad (3.17)$$

The information of the threshold has a constant offset from the information of the weights. For the case of x_0=-1, x_1=+1, we have with C_1=1 just one bit offset.

term4) = term5)

The corresponding case for the threshold and weights of the second layer reveals

$$\delta W = \delta T \qquad (3.18)$$

The threshold should be as fine grained as the weights since it is always involved in the output accuracy. The equation (3.18) gives with (3.8d) and (3.8e)

$$a(x_1 - x_0)^2/m \quad 2^{-I_W} = a/2\, g_T(m)\, 2^{-I_T}$$

and therefore

$$\mathrm{ld}\,(2^{I_T}) = \mathrm{ld}(2^{I_W}) + \mathrm{ld}(g_T(m)/2) - \mathrm{ld}((x_1 - x_0)^2/m)$$

$$I_T = I_W + \mathrm{ld}(g_T(m)/2) - \mathrm{ld}((x_1 - x_0)^2/m) \qquad (3.19)$$

The threshold information of the second layer has also an offset between the weights and the threshold which depends on the number of inputs from the first layer.

term3) = term4)

The comparison between the threshold of the first layer and the weights of the second layer gives $\frac{2a}{m^2}x_1(x_1-x_0)\,\delta$

and therefore using (3.8c) and (3.8d)

$$\frac{a}{m}x_1(x_1-x_0)\,2^{-I_t} = \frac{a}{m}(x_1-x_0)^2\;2^{-I_W}$$

$$ld(2^{I_t}) = ld(2^{I_W}) + ld(x_1/(x_1-x_0))$$

$$I_t = I_W + C_2 \qquad \text{with } C_2 := ld(x_1/(x_1-x_0)) \qquad (3.21)$$

The constant offset C_2 is in the simple case of x_0=-1, x_1=+1 just -1 Bit.

term1) = term5)

The condition for the number of neurons is

$$\frac{a}{4m^3}(x_1-x_0)^2\,[1 - 2^{-I_T}/2\,]\,(I_w+I_t+I_W)^{-1} = \ln(2)\,\delta T \qquad (3.22)$$

Using (3.17),(3.19) and (3.21) the condition (3.22) becomes

$$\frac{a}{4m^3}(x_1-x_0)^2\,[1 - 2^{-I_T}/2\,]\,(I_W + C_2 - C_1 + I_W + C_2 + I_W)^{-1} = \ln(2)\,\delta T$$

With $C_2 = - C_1$ and equation (3.8e) we get

$$\frac{a}{4m^3}(x_1-x_0)^2\,[1 - 2^{-I_T}/2\,] = 3\,(I_W + C_2)\ln(2)\,a/2\;g_T(m)\,2^{-I_T}$$

$$(x_1-x_0)^2\,(2^{I_T} - 1/2\,) = 6m^3\,(I_W + C_2)\ln(2)g_T(m)$$

and finally

$$6m^3\,\mathbf{(}I_T - ld(g_T(m)/2) + ld((x_1-x_0)^2/m) + C_2\mathbf{)}\,\ln(2)g_T(m) - (x_1-x_0)^2(2^{I_T} - 1/2) = 0 \qquad (3.23)$$

The equation (3.23) is hard to solve analytically. For numerically given I_T the corresponding *m* can be found by a numerical iteration procedure. If we put equation (3.23) into the form

$$m = h(m)^{1/3} \qquad (3.24)$$

we can use it as an iteration formula at the (t+1)-th iteration for *m*:

$$m(t+1) = h(m(t))^{1/3} \qquad (3.25)$$

Since the derivative of $h(m)^{1/3}$ is lower 1, the convergence condition is satisfied and the iteration converges. The following figure 7 shows the optimal system parameter *m* when one parameter

Fig. 7 The optimal system parameters for the approximation

(the threshold information I_T) is given. The corresponding values for I_W and the overall system information I_{sys} are also given. The values for I_t and I_w differ from I_W only by a constant offset of one and two bits; in the figure they are omitted for clarity.

Example

> Let us consider an information of 16 bits in the threshold T. In the simple case of x_0=-1, x_1=+1, a=1, b=0 we have with I_T := 16 bit the optimal configuration at
>
> $$m = 16.54 \text{ neurons}, \quad I_W = 14.95 \text{ bit}, I_i=I_W+C_2 = 13.95 \text{ bit}, I_w = I_t -C_1 =12.95 \text{ bit}$$
>
> The overall information in the network is then with (3.10)
>
> $$I_{sys} = m(I_w+I_t+I_W) + I_T= 708.45 \text{ [bits]}$$
>
> and the approximation error is $d_f^{max} = 2.213\text{x}10^{-3}$. If we augment the information capacity of the system to I_T=32 Bit, the error will diminuish to $d_f^{max} =1.847\text{x}10^{-6}$ when we use the optimal system parameters.

In the following figure 8 the minimal approximation error for optimal system parameters is shown in logarithmic notation for the whole interval of I_T = 4 .. 32 bits. The nearly linear appearence is due to the fact that all terms of the resolution error contains powers of two, which transforms to linear terms in I_T.

Fig. 8 The approximation error as a function of the information in the network

The corresponding approximation error for equal resolutions $I_w=I_t=I_W=I_T$ and optimal *m* are generally slightly worse than the one for an optimal information distribution.

4 Conclusion

The principle of optimal information distribution is a criterium for the efficient use of the different information storage ressources in a given network. Furthermore, it can be used as a tool to balance the system parameters and to obtain the optimal network parameter configuration according to the minimal usable storage for a maximal error which is given.

In this paper a simple, non-linear function approximation is evaluated, the conditions for optimal system configuration are stated, their solutions are analytically computed and their nature is explained.

The example of the approximation of a simple quadratic function is quite instructive to evaluate, but has the desadvantage that it is not very common in real world applications. To show that the proposed principle of information distribution works in more realistic environment, the more complicated function of the inverse kinematic of a PUMA robot is considered in another report [BR89]. There the results for optimal system parameters are partially obtained by numerical iterative approximations.

References

[BAK90] Gregory Baker, Jerry Gollub: *Chaotic dynamics: an introduction*; Cambridge University Press, 1990

[BR89] R.Brause : *Performance and Storage Requirements of Topology-Conserving Maps for Robot Manipulator Control*; Internal Report 5/89, Fachbereich Informatik, University of Frankfurt, FRG

[BR90] R.Brause : *Optimal Information Distribution and Performance in Neighbourhood-conserving Maps for Robot Control*; IEEE Proc. Tools for Art. Int. TAI-90, Dulles 1990

[GIR90] F.Girosi, T. Poggio: *Networks and the Best Approximation Property*;
Biolog. Cybern. (1990) Vol. 63, pp. 169-176

[HOR89] K. Hornik, M. Stinchcomb, H.White: *Multilayer Feedforward Networks are Universal Approximators*
Neural Networks (1989), Vol 2, pp. 359-366

[LAP87] A. Lapedes, R. Farber: *Nonlinear Signal Processing using Neural Networks*:
Prediction and System Modelling; Los Alamos preprint LA-UR-87-2662 (1987)

[SHA49] C.E. Shannon, W.Weaver: *The Mathematical Theory of Information*;
University of Illinois Press, Urbana 1949

[STIN89] M. Stinchcomb, H.White: *Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions*
Proc. Int. Joint Conf. Neural Networks, Washington DC, June 1989,pp. I/607-611

Appendix A1: The linear approximation error

The non-linear function in the intervall $[x-\Delta x/2,\ x+\Delta x/2]$ is

$$f(x) = ax^2 + b$$

and the linear approximation by the neural network is

$$\hat{f}(x) = \alpha x + \beta \qquad \text{with } \alpha := 2ax$$

Fig. A1.1 The error of the linear approximation

The approximation error is (see figures 3 and A1.1 above)

$$d^{lin}(x) = f(x) - \hat{f}(x) = ax^2 + b - 2axx - \beta = b - \beta - ax^2 =: d$$

$$\begin{aligned} d^{lin}(x+\Delta x/2) &= f(x+\Delta x/2) - \hat{f}(x+\Delta x/2) = a(x+\Delta x/2)^2 + b - 2ax(x+\Delta x/2) - \beta \\ &= ax^2 + a\Delta xx + a(\Delta x/2)^2 + b - 2axx - ax\Delta x - \beta \\ &= -ax^2 + b - \beta + a(\Delta x/2)^2 = d + a(\Delta x/2)^2 \end{aligned}$$

$$\begin{aligned} d^{lin}(x-\Delta x/2) &= f(x-\Delta x/2) - \hat{f}(x-\Delta x/2) = a(x-\Delta x/2)^2 + b - 2ax(x-\Delta x/2) - \beta \\ &= ax^2 - a\Delta xx + a(\Delta x/2)^2 + b - 2axx + ax\Delta x - \beta \\ &= -ax^2 + b - \beta + a(\Delta x/2)^2 = d + a(\Delta x/2)^2 \end{aligned}$$

The errors at the boarders are equal. The maximal error $\max(|d^{lin}(x)|,|d^{lin}(x+\Delta x/2)|)$ is minimal when all the errors are equal

$$|d^{lin}(x)| = |d^{lin}(x+\Delta x/2)|$$

or

$$|d| = |d + a(\Delta x/2)^2|$$

This is given when

$$d := -a/2(\Delta x/2)^2$$

The maximal linear error is not dependant on the value of x, it is the same in the whole intervall

$$d_{lin}^{max} = a/2(\Delta x/2)^2 \qquad \text{(A.1.1)}$$

Since we have $\Delta x = (x_1 - x_0)/m$

$$d_{lin}^{max} = a/2((x_1-x_0)/2m)^2 = m^{-2}a(x_1-x_0)^2/8 \qquad \text{(A.1.2)}$$

and therefore

$$d_{lin}^{max} = C\, m^b \qquad \text{with } C := a(x_1-x_0)^2/8) \text{ and } b := -2 \qquad \text{(A.1.3)}$$

Appendix A2: The resolution error

For the computation of the resolution error let us assume that in all weights the maximal increment error has occurred. The approximating function becomes with (3.2) and (3.3)

$$\hat{f}(x,\delta) = \Sigma_i\, (W_i + \delta W_i)\, S(z_i+\delta z_i) \quad + T+\delta T \qquad \text{(A2.1)}$$

$$= \Sigma_i\, W_i\, S(z_i+\delta z_i) \ + T + \Sigma_i\, \delta W_i\, S(z_i+\delta z_i) \ \ +\delta T$$

Because the intervalls are exclusive, for the k-th intervall we have to regard only the influence of one neuron of the first layer; for i<k we have $S(z_i) = S(z_i+\delta z_i) = 1$ and for i>k we have $S(z_i+\delta z_i)=0$.

$$\hat{f}(x,\delta) = (\Sigma_i^{k-1} W_i) \ + W_k\, S(z_k+\delta z_k) + T + (\Sigma_i^{k-1}\delta W_i) + \delta W_k\, S(z_k+\delta z_k) + \delta T$$
$$= \hat{f}(x) \ + W_k\, \delta z_k + (\Sigma_i^{k-1}\delta W_i) + \delta W_k\, S(z_k+\delta z_k) + \delta T$$

The *maximal error* d_{res}^{max} is encountered at the boarder of the intervall $[x_0,x_1]$ with $\max(x) = x_1$
The contribution of the term $\delta W_i S(.)$ becomes maximal δW_i when $S(.) = 1$. Therefore, we have

$$\hat{f}(x_1,\delta) = \hat{f}(x_1) + (\sum_i^{m-1}\delta W_i) + W_m\, \delta z_m + \delta W_m\, S(z_m+\delta z_m) + \delta T$$

$$= \hat{f}(x_1) + (\sum_i^{m}\delta W_i) + W_m\, \delta z_m \ + \delta T$$

and so with $\delta z_m = \delta w_m x_m + \delta t_m$ we get

$$d_{res}^{max} = \hat{f}(x_1,\delta) - \hat{f}(x_1) = \ (\sum_i^{m}\delta W_i) + W_m\, (\delta w_m\, x_1 + \delta t_m) + \delta T$$

Because all the error increments are independent of their index, we get with (3.6c)

$$d_{res}^{max} = 2ax_1\, \Delta x\, [\delta w\, x_1 + \delta t] + m\delta W + \delta T \qquad \text{(A2.2)}$$

Using the definitions (3.8b,c,d,e) we get

$$d_{res}^{max}(m) = 2\frac{ax_1}{m}\, (x_0-x_1)\Big[\, \frac{m \qquad x_1}{2(x_1-x_0)2^{I_w}} + \frac{m}{2^{I_t}\, 2}\, \Big] \ + \frac{m\, a}{m\ \ 2^{I_W}}(x_1-x_0)^2 + \frac{1}{2}\Big[ax_0^{\,2} + b - \frac{a}{2}\, \frac{(x_1-x_0)^2}{4\, m^2}\Big]\, 2^{-I_T}$$

$$= \ 2ax_1\, (x_0-x_1)\Big[\, \frac{1 \qquad x_1}{2(x_1-x_0)2^{I_w}} + \frac{1}{2^{I_t}\, 2}\Big] \ + \frac{a}{2^{I_W}}(x_1-x_0)^2 + \frac{1}{2}\Big[ax_0^{\,2} + b - \frac{a}{2}\, \frac{(x_1-x_0)^2}{4\, m^2}\Big]\, 2^{-I_T}$$

(A2.3)