Noise-modulated neural networks for selectively functionalizing sub-networks by exploiting stochastic resonance

Shuhei Ikemoto*

Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology, 2-4 Hibikino, Wakamatsu, Kitakyushu, Fukuoka, 8080196, Japan

Research Center for Neuromorphic AI Hardware, Kyushu Institute of Technology, 2-4 Hibikino, Wakamatsu, Kitakyushu, Fukuoka, 8080196, Japan

Abstract

In the phenomenon of stochastic resonance, adding a certain level of nonzero noise to a nonlinear system reduces information loss. A previous study proposed a neural network consisting of thresholding functions that exploit stochastic resonance at run time and during training, with the aim of smooth mapping and backpropagation. Such a neural network can be rephrased as one that operates only when noise is added, i.e., one that is unable to smoothly map and train when noise is absent. Focusing on both explanations simultaneously, a neural network for which only a sub-network is activated selectively by adding noise locally on that sub-network is proposed in this paper. To this end, a new activation function is introduced. It exploits stochastic resonance and presents null output and derivative when no noise is added. Simple simulations confirm that the proposed neural network with the new activation function allows the sub-network to be functionalized selectively, and interpolations are investigated by imposing varying noise intensity on various regions of the network after sub-networks are trained separately.

Keywords: Stochastic resonance, Neural network, Localized noise

*Corresponding author
Email address: ikemoto@brain.kyutech.ac.jp (Shuhei Ikemoto)

1. Introduction

Neurons and receptors are always subject to strong noise effects [1, 2]. Therefore, it is assumed that they possess mechanisms for which noise is part of how their functions occur[3, 4]. In the phenomenon of stochastic resonance (SR), information lost in a transformation through a nonlinear system is recovered by adding an adequate level of nonzero noise to the input signal [5, 6]. Originally proposed as a conceptual model for periodic changes in Earth's climate [7], SR was then found to contribute widely to the activities of neurons [8, 9, 10] and receptors [11, 12, 13, 14], and it is now deemed key to understanding biological information processing [15, 16].

Because SR can only recover information that is lost in a nonlinear system (e.g., when discretizing a continuous variable, the information loss is never zero), it is not beneficial for linear systems. In other words, although SR makes a nonlinear system respond more like a linear system, the nonlinearity is not completely removed. However, a corollary is that we should focus on systems that are inherently nonlinear when seeking applications of SR. Neural networks (NNs) are such systems because the nonlinearity of the activation function plays a vital role in their universal approximation capabilities.

In a previous study, we proposed a noise-modulated artificial neural network consisting of threshold elements, where the abovementioned feature of SR was taken as an example [17]. In this model, the activation functions of the hidden units are defined as the expected values of inputs that exceed a threshold. Adding noise enables smooth mapping at run time and backpropagation during training. In the absence of noise, the mapping becomes stair-like, and backpropagation updates only the weights of the output layer. Here, we imagine replacing the threshold function with one that binarizes the input signals but always outputs zero in the absence of noise. For instance, a threshold function that always outputs zero in the absence of noise is one that is applied twice to the input and outputs 1 only if the results differ. Even in such a case, note that both smooth mapping and backpropagation are enabled by SR. However,

the situation is very different in the absence of noise; in this case, the network always outputs zero at run time, and none of the weights (including the output weights) are updated by backpropagation during training. Although it may sound trivial, this property, that if there is no noise, nothing will be output or back-propagated, is potentially important because it implies that only part of the network can be functionalized as an NN, meaning that only the sub-network given the noise is selectively enabled to train and map inputs to outputs. Therefore, this property assures that where noise is applied becomes a parameter that determines the effective network, both during training and at run time.

In this paper, an activation function with this property is proposed, and it is used to update the NN proposed in [17]. To verify only the sub-network to which noise has been added functions as an NN thanks to the proposed activation function, two independent regions in hidden layers of the same network, hereafter called coprime sub-networks, are trained to approximate two different functions in succession.

After training, interpolation with functions in the parameter space is visualized by gradually changing the parameter "where noise imposed" in order to functionalize one or the other sub-networks.

This paper is organized as follows. Section 2 presents related work to highlight the novelty and importance of this study. Section 3 begins by defining the new activation function, which ensures the aforementioned property that selectively functions sub-networks by adding noise, and the structure and mechanism of the activation function in the new noise-modulated NN are explained. Validation results are presented in detail in Section 4. A discussion is presented in Section 5 and conclusions are offered in 6.

2. Related work

Information processing in neuronal ensembles has been studied in the field of SR [3, 18]. Biological neuron models, such as the Hodgkin-Huxley and FitzHugh-Nagumo models, are often used as the elements, and the main target

of the investigation is SR occurring in the complex coupled dynamics. In particular, information throughput [19, 20, 21] and firing coherence [22, 23, 24, 25, 26] have been studied to evaluate the effects of SR. The main focus underlying those studies was to identify and observe SR, including coherence resonance [27] and chaos resonance [28, 29], and to determine their mechanisms. In addition to dynamical biological neuron models, there are well-known non-dynamical models that exhibit SR when noise is added. Although it is not necessary to use a threshold to quantize a continuous value in order to produce SR [30], the majority of prior studies to date have focused on the use of thresholding functions [31, 32, 33]. System with multiple thresholding elements were investigated [34, 35], as well as systems with an adaptive threshold function corresponding to synapses [36, 37]. As in SR studies that used biological neuron models, these studies also focused on finding, analyzing, and evaluating instances of SR.

The focus in the aforementioned SR studies was on systems that are noisy throughout. In other words, attention is yet to be paid to phenomena that occur (i) when noise is imposed on only part of a system, and (ii) where that noise is imposed. The present research is focused on a system to which noise is imposed on only in various parts of the system, and as such it has the potential to provide a new perspective in this research field.

Parallel to the aforementioned scientific studies on SR, NNs have been studied regarding functional approximation and dimensionality reduction in the fields of artificial intelligence and soft computing, as evidenced by their huge success. For distinction, this type of mathematical model is often referred to specifically as an artificial NN (ANN). Generally speaking, ANNs have two different operating modes, namely training and inference, and to date the use of noise in ANNs has been investigated mainly in the training phase. The backpropagation algorithm [38] is the foundation of training algorithms, and the role of noise was investigated to benefit the performance and the applicability. The benefits that have been studied to date fall into three categories, the first being the generalization performance. It has been shown that adding noise to input signals is equivalent to giving the loss function a regularization term

[39, 40], and adding noise for regularization is currently a popular technique for training ANNs. The second point is to enable backpropagation for binary activation functions. Because the derivatives of binary activation functions are zero almost everywhere, backpropagation cannot be applied directly, and other training algorithms have been proposed as alternatives to backpropagation [41]. Meanwhile, an alternative approach was proposed, in which stochastic noise fluctuations are imposed on weights to smooth out binary activation functions in order to facilitate gradient-descent algorithms [42, 43]. More specifically, in [44, 45], binary activation functions were replaced temporarily with continuous sigmoidal functions to train ANNs. In that approach, it was reported that noise can mitigate the discrepancy between the activation functions at run time and during training [46]. Note that those studies regarding the second point were aimed specifically at the training phase; binary activation functions without noise were used directly at run time. The third point is faster convergence. In [47, 48], sigmoidal activation functions were used, and adding noise led to faster convergence with less error in the training phase. In particular, that approach was shown to be valid for convolutional NNs [49, 50]. As is the case in the second point, the focus in the third point is clearly on injecting noise only during training. In summary, the use of noise in ANNs has been investigated for training but was not studied with regard to SR at run time.

In contrast with the aforementioned studies, noise was imposed both during training and at run time in a previous study by the present author [17], in which the idea of noise-modulated NNs was proposed. In a noise-modulated NN, noise during training enables to apply backpropagation to thresholding activation functions, while that at run time provides smooth regression with fewer hidden units. In particular, these functions were evaluated and discussed in terms of SR. Based on those results, further novelty is brought to this field by focusing on the functionalities achieved by imposing noise only on a sub-network, and by varying which sub-network receives the added noise. Consequently, the present research is worth pursuing as it is unique from scientific and engineering perspectives.

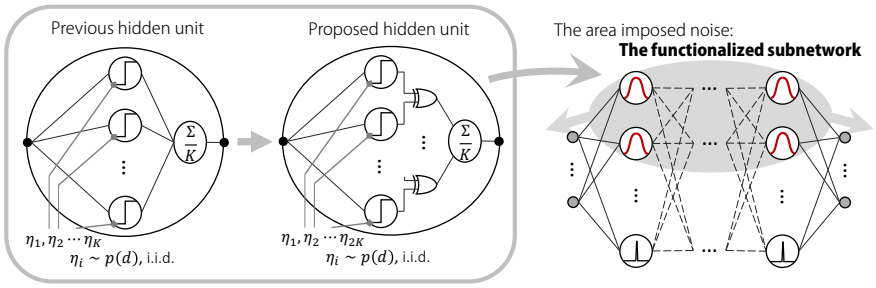

Figure 1: Schematic of a noise-modulated NN for selectively functionalizing sub-networks that exploit SR. Left: previous hidden unit proposed in [17]. Center: hidden unit proposed here, implemented simply by inserting only XOR elements into the previous element. Right: the noise-modulated NN comprising the proposed hidden units. Because of SR invoked in the selected hidden units, the only sub-network that is functionalized is the sub-network with the added noise.

3. Proposed model

Figure 1 shows a schematic of the proposed model, and its components are explained in detail in this section.

Let us focus on a step function with a threshold θ as an activation function of an ANN, namely

$$z = \phi(d) = \begin{cases} 1 & if\ d \geq \theta \\ 0 & otherwise \end{cases}, \tag{1}$$

where d and z are the input and output, respectively, of the activation function. Considering that noise η is added to this function, which specifically causes the threshold θ to fluctuate by independent and identically distributed samples from $p(\xi)$, the output z becomes stochastic as follows:

$$z = \phi(d) = \begin{cases} 1 & if\ d \geq \eta \overset{\text{iid}}{\sim} p(\xi) \\ 0 & otherwise \end{cases}. \tag{2}$$

The ensemble average $\langle z \rangle$, which is output by the previous hidden unit depicted in the left panel of Figure 1, corresponds to the following expectation value:

$$\langle z \rangle = \mathbb{E}[z] = P(d \geq \eta) = \int_{-\infty}^{d} p(\xi) d\xi. \tag{3}$$

The z is originally binary, and the information of the continuous input is lost, but the $\langle z \rangle$ obtained by adding noise can reflect the information of the continuous input. This is best known as the simplest SR. In addition, this yields the effective activation function of the hidden unit:$\bar{\phi}$ as

$$\bar{\phi}(d) = \langle z \rangle \tag{4}$$

$$\bar{\phi}(d)' = p(d). \tag{5}$$

Note that the derivative of the effective activation function $\bar{\phi}(d)'$ corresponds to the noise density function $p(d)$, despite the fact that the hidden unit comprises discrete step functions.

The core idea of the noise-modulated NN proposed in [17] is to use Eqs. 4 and 5 at run time and during training, respectively. Despite the fact that the network comprises step functions, it can provide smooth mapping and be trained by backpropagation methods owing to SR . In addition to this original idea, the aim of the present study is to disable learning merely by removing noise. Here, "disabling learning" means that the parameters are never updated, even if backpropagation is applied. Assuming there is no noise, $p(\xi) = \delta(\xi)$ holds and z is no longer stochastic. In that case, because the Dirac delta is zero almost everywhere, the derivative of Eq. 1 is deemed to be virtually zero during training. In particular, $\bar{\phi}(d) \in \{0, 1\}$ and $\bar{\phi}(d)' = 0$ hold. This means that, during backpropagation, parameter updating is not applied to weights that are connected to hidden units. However, weights connected to the output layer are updated when the hidden units to which they are connected output 1. Thus, the learning capability of the noise-modulated NN in [17] cannot be disabled merely by removing noise.

To disable learning in the absence of noise, the derivative and output of activation function must be zero. Herein, the following simple activation function is proposed:

$$z = \phi(d) = \begin{cases} 1 & if\ d \geq \eta_1\ \dot{\vee}\ d \geq \eta_2 \\ 0 & otherwise, \end{cases} \tag{6}$$

where $\dot{\vee}$ is the logic operator XOR. where $\eta_1 \overset{\text{iid}}{\sim} p(\xi)$ and $\eta_2 \overset{\text{iid}}{\sim} p(\xi)$. The center panel of Figure 1 shows how the proposed hidden unit is constructed. The corresponding effective activation function is written as follows:

$$\begin{aligned} \bar{\phi}(d) &= \alpha P(d \geq \eta)(1 - P(d \geq \eta)) & (7) \\ &\propto \langle z \rangle \\ \bar{\phi}(d)' &= \alpha(1 - 2P(d \geq \eta))p(d) & (8) \end{aligned}$$

where α is a scaling coefficient. Because the maximum of $P(d \geq \eta)(1-P(d \geq \eta))$ is 0.25, $\alpha = 4$ is typically used for normalizing the output. Note that $P(d \geq \eta)$ can also be obtained as an ensemble average of the outputs from all step functions in this architecture. Qualitatively, this activation function outputs 1 only if the added noise causes the input to cross the threshold. From Eq. 7, it is evident that $\bar{\phi}(d) = 0$ holds in the absence of noise. In addition, regarding Eq. 8, $\bar{\phi}(d)' = 0$ holds too for the same reason as that in Eq. 5. Therefore, this simple activation function fulfills the property of disabling learning merely in the absence of noise.

The noise-modulated NN proposed herein is a fully-connected feedforward NN containing the proposed hidden units. Let x and y be column vectors of the input and output, respectively; the network has N hidden layers and is recursively formalized as follows:

$$\begin{aligned} h_0 &= x & (9) \\ s_i &= W_{i-1}h_{i-1} + b_{i-1}, \quad 1 \leq i \leq N & (10) \\ h_i &= \bar{\Phi}_i(s_i) & (11) \\ y &= W_N h_N + b_N & (12) \end{aligned}$$

where W_i and b_i are weight matrices and bias vectors, respectively. In addition, $\bar{\Phi}_i(\cdot)$ indicates the element-wise application of the proposed activation function $\bar{\phi}(\cdot)$, assuming that noise $\eta \overset{\text{iid}}{\sim} p_{i,j}(\eta)$ is added to the input of unit j in hidden

layer i. The derivative of $\bar{\Phi}_i(s_i)$ is written as follows:

$$\bar{\Phi}_i(s_i)' = [\bar{\phi}_{i,1}(s_i)', \bar{\phi}_{i,2}(s_i)', \cdots \bar{\phi}_{i,M_i}(s_i)']^T \tag{13}$$

$$\bar{\phi}_{i,j}(s_{i,j})' = \alpha(1 - 2P_{i,j}(s_{i,j} \geq \eta))p_{i,j}(s_{i,j}) \tag{14}$$

$$P_{i,j}(s_{i,j} \geq \eta) = \langle \tilde{z} \rangle_{i,j} = \int_{-\infty}^{s_{i,j}} p_{i,j}(\xi) d\xi \tag{15}$$

where $\langle \tilde{z} \rangle_{i,j}$, $s_{i,j}$, and M_i indicate the ensemble average of the outputs from the $2K$ step functions constituting hidden unit j in hidden layer i, element j of s_i, and the number of hidden units in hidden layer i, respectively.

Let $\mathcal{D} = \{\mathbf{x}, \mathbf{t}\} = \{x_t, t_t\}_{t=0}^T$ be a dataset comprising D pairs of input vectors x_t and output vectors t_t. Denoting the outputs from the proposed NN corresponding to each input $\mathbf{x}$ as $\mathbf{y} = \{y_t\}_{t=0}^T$, an objective function is generally expressed as follows:

$$E(\mathbf{y}, \mathbf{t}) = \sum_{t=0}^{T} L(y_t, t_t). \tag{16}$$

To minimize the objective function, the gradients of the weights are recursively computed by backpropagation as follows:

$$e_N = \frac{\partial L}{\partial y} \tag{17}$$

$$e_{i-1} = W_i^T e_i \odot \bar{\Phi}_i(s_i)' \tag{18}$$

$$\frac{\partial E}{\partial W_i} = \sum_{t=0}^{T} \left(e_i h_i^T \right)_{\{x,y\}=\{x_t,t_t\}} \tag{19}$$

where $\odot$ indicates the Hadamard product. The bias parameters are also computed recursively using Eqs. 17-19 with $h_i = 1$.

Assuming the noise imposed on hidden unit j in hidden layer i is removed, and $p_{i,j}(\eta) = \delta(\eta)$ holds, then column j in $\frac{\partial E}{\partial W_i}$ becomes a zero vector because element j in h_i is zero. In addition, because element j in $\bar{\Phi}_i(s_i)'$ is zero, element j of e_{i-1} becomes zero and row j in $\frac{\partial E}{\partial W_{i-1}}$ becomes a zero vector. This means that backpropagation does not update the parameters connected to those units when the noise applied to the hidden units is removed. Furthermore, at run time, these hidden units do not contribute to the network's mapping because they output zero. Therefore, if noise is applied to only part of the network,

as shown in the right panel of Figure 1, then SR only functionalizes that sub-network.

The fact that the knowledge of the noise probability density function is required a priori for backpropagation appears to be an unrealistic assumption. Therefore, a kernel density estimation with a uniform kernel function can be used and naturally integrated without compromising the simplicity of the proposed approach. Details will be discussed in Section 5 and Appendix A.

In addition, it is noteworthy that the relationship between ϕ and $\bar{\phi}$ indicates the possibility of simpler expression of the proposed NN shown in Figure 1. Considering ϕ as a function that receives a stochastic input $D = d - \eta$ and outputs 1 when $D \geq 0$ holds, and assuming $p(\xi)$ has zero mean, the following relationship holds:

$$\langle \phi(D) \rangle = \bar{\phi}(\langle D \rangle) = \bar{\phi}(d). \tag{20}$$

Therefore, let Φ denote the element-wise application of the function $\phi(\cdot)$, and let Y be the output of the network using Φ instead of $\bar{\Phi}$ in Eqs. 9-12, then $\langle Y \rangle = y$ holds. This means that it is possible to construct a noise-modulated NN and train it by computing the ensemble average of the output of the network with the number of elements used in each unit K set to 1, instead of computing the output of the network while computing the ensemble mean in each unit. This feature will add further plausibility of the hardware implementation of the noise-modulated NN that will be discussed in Section 5.

4. Simulation

In this section, the noise-modulated NN explained in Section 3 is verified whether it can selectively functionalize its sub-networks as NNs by adding noise thanks to SR. The specific points to be verified are as follows.

1. The activation function in Eq. 7 exploits SR.
2. Adding noise to part of the network makes that sub-network trainable, so that two functionally coprime sub-networks approximate two different functions.

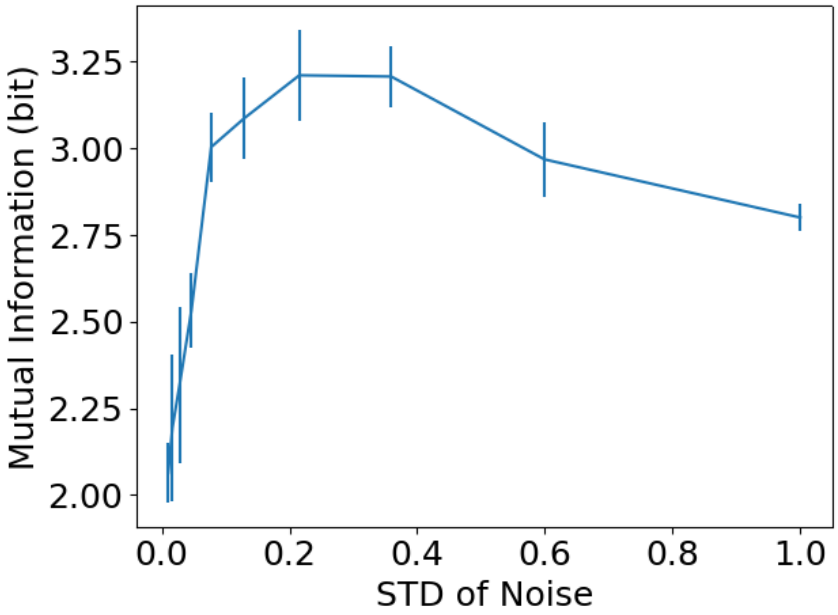

Figure 2: Mutual information with respect to changes in noise intensity. The horizontal and vertical axes represent the standard deviation of white Gaussian noise and the mutual information between an approximated function and a sinusoidal function, respectively. The occurrence of SR is confirmed by the unimodality of the curve, which is a well-known characteristic of SR.

3. Changing where and how strongly noise is added between two functionally coprime sub-networks yields an interpolation between the two functions.

The results of the verification of these points are shown in the following subsections.

4.1. Verifying the occurrence of stochastic resonance

The occurrence of SR has been confirmed by observing that a certain nonzero noise intensity would maximize some performance of the system. Mutual information between a system input and output is widely used to quantify system performance. In the present study, because the system was the noise-modulated NN explained in Section 3, data points sampled from a target function and corresponding points in the approximation of trained network were considered the

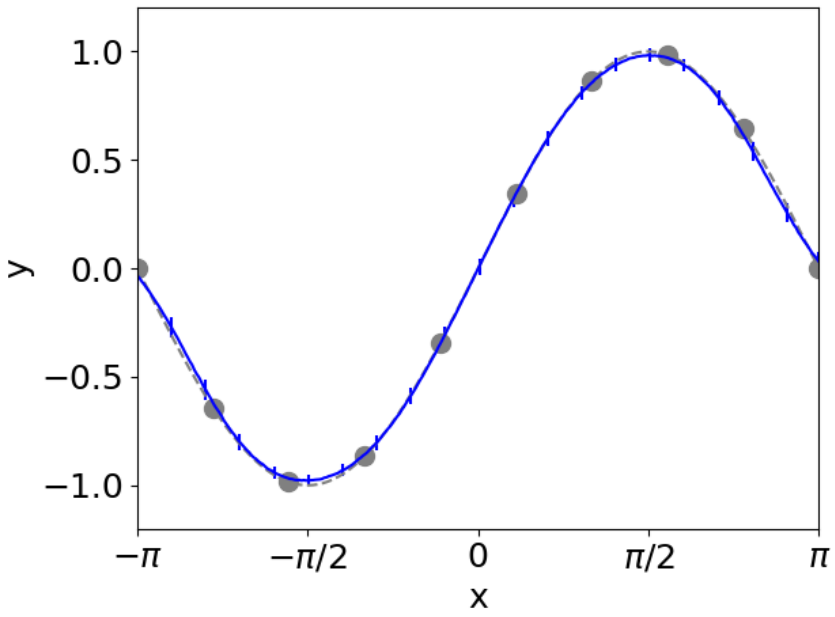

Figure 3: Function approximated by the noise-modulated NN with white Gaussian noise $\mathrm{N}(0, 0.25^2)$. The solid and dashed lines indicate the average and standard deviation, respectively, of the approximated and target functions. The markers show data that were used for training.

input and output, respectively. Considering $y(x)$ and $\hat{y}(x)$ as outputs of a target function and a trained network, the probabilities $p(y)$, $p(\hat{y})$, and $p(y, \hat{y})$ are obtained by appropriately binning their ranges and counting up those frequencies over $x \in X$. The mutual information between Y and $\hat{Y}$, i.e., sets of binned values of $y(x)$ and $\hat{y}(x)$, is calculated as follows:

$$I(Y; \hat{Y}) = \sum_{\hat{Y}} \sum_{Y} p(y, \hat{y}) \log \frac{p(y, \hat{y})}{p(y)p(\hat{y})} \tag{21}$$

where a base 2 logarithm is used throughout this study.

For evaluation, a sinusoidal function $y = \sin(x)$ was approximated by the noise-modulated NN. Therefore, the input and output from the noise-modulated NN were one-dimensional. The network had two hidden layers, each of which had 10 hidden units, where $K = 100$ was used in each hidden unit (see also Figure 1). To enable learning and mapping, white Gaussian noise was added to all hidden units $p_{i,j} = \mathrm{N}(0, \sigma^2), \forall i, j$. 10 and 314 points, respectively, with equally spaced abscissas in the range $x = [-\pi, \pi]$ were used for training and to compute the mutual information. During training, the Adam algorithm [51] was used to minimize the mean squared error of the approximation, and parameters were updated for 1000 epochs.

10 points with logarithmically spaced abscissas in the range $\sigma = [10^{-2}, 1]$ were used to investigate how the mutual information changed with the standard deviation σ. 10 networks were instantiated and trained independently for each σ value. The resulting relationship between the standard deviation σ and mutual information is shown in Figure 2. The clearly shown unimodal curve indicates that a certain nonzero noise intensity maximizes mutual information, which shows how the proposed NN exploits SR. This validates the first issue.

In addition, Figure 2 shows that the best approximation accuracy was obtained with $\sigma \approx 0.25$. Figure 3 shows the approximations obtained with 100 networks trained with $\sigma = 0.25$. As shown, the target function was approximated successfully without overfitting. In addition, the small error bars (standard deviation in the approximation) show that training was sufficiently reproducible.

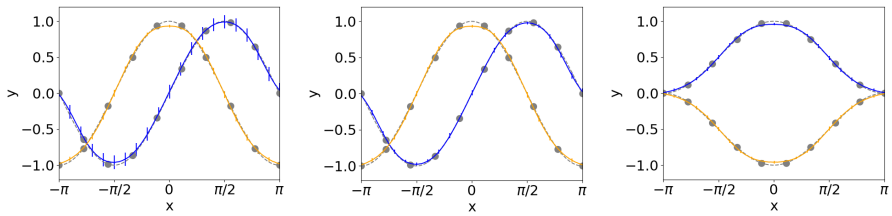

Figure 4: Approximations of two functions with the noise-modulated NN with white Gaussian noise $\mathrm{N}(0, 0.25^2)$. Training was carried out sequentially. $y = \sin(x)$ and $y = \cos(x)$ were used as target functions in the left and center graphs, and $y = (1+\cos(x))/2$ and $y = -(1+\cos(x))/2$ were used in the right graph. The solid and dashed lines show the averages and standard deviations of the approximated functions and target functions, respectively, and the markers indicate the 10 data points that were used for training. Left: Because the bias input to the output layer is not disabled by changing where the noise is imposed, the later training for $y = \cos(x)$ clearly increased the variance of earlier training for $y = \sin(x)$, and the accuracy is deteriorated. Center: If no noise is input to the output layer, then the two sequential trainings do not interfere with each other because the two sub-networks are functionally coprime. Right: Because two functions with different averages are trained sequentially, the bias to the output layer is not necessarily required.

4.2. Training two sub-networks with two different functions

Next, networks that were twice the size of those in Subsection 4.1 were used, i.e., $N = 2$ and $M_i = 20, \forall i$. To approximate two different functions in one network, noise was added to some of the hidden units. To begin, functions $y = \sin(x)$ and $y = \cos(x)$ were used as the target of approximation. For training, 10 points were sampled from each function with equally spaced abscissas in the range $x = [-\pi, \pi]$.

Let S_1 and S_2 be sets of indexes to add noise to the selected area as follows:

$$S_1 = \{i | 1 \leq i \leq 10\} \subset \mathbb{N} \tag{22}$$

$$S_2 = \{i | 11 \leq i \leq 20\} \subset \mathbb{N}. \tag{23}$$

To functionalize the first sub-network for approximating $y = \sin(x)$, noise was

imposed on half of the hidden units:

$$p_{i,j} = \mathrm{N}(0, 0.25^2), \forall i, j \in S_1 \quad (24)$$

$$p_{i,j} = \delta, \forall i, j \in S_2. \quad (25)$$

This allowed only the first sub-network to be functionalized. Conversely, to approximate $y = \cos(x)$, noise was added to the second sub-network by replacing S_1 and S_2 in Eqs. 24 and 25. Two sub-networks were trained sequentially. Regarding training of sub-networks, as in Section 4.1, the Adam algorithm was used to update the parameters in each sub-network over 1000 epochs.

Figure 4 shows the averages and standard deviations of the two approximated functions obtained over 100 instances with the proposed NN, where the three graphs show the training results performed in the different setups. The left graph shows the results obtained by simply performing training for both coprime sub-networks, showing that the standard deviation of the approximation of $y = \sin(x)$ increased compared to the results in Figure 3. This means the approximation stored in the first sub-network became less accurate during subsequent training of the second network for $y = \cos(x)$. The reason for this was deemed to be that the bias input to the output layer was not separated clearly for these sub-networks. Therefore, if no bias is used, then the two sub-networks become functionally coprime, thereby solving this problem deteriorating the achieved approximation by new training. The center graph in Figure 4 shows the results obtained when the same training process was performed without bias connected to the output layer. As expected, this graph shows that the aforementioned problem was solved and the two functions were approximated successfully by these two sub-networks. Qualitatively, the main role of the bias input to the output layer is to quickly compensate a target function with a nonzero average. Because $y = \sin(x)$ and $y = \cos(x)$ both have zero averages, removing the bias is believed to cause no problem. Therefore, functions with different nonzero averages were also tested. The right graph in Figure 4 shows the results obtained in the same manner as those in the center graph, but for two different target functions, namely $y = (1 + \cos(x)/2)$ and $y = -(1 + \cos(x)/2)$.

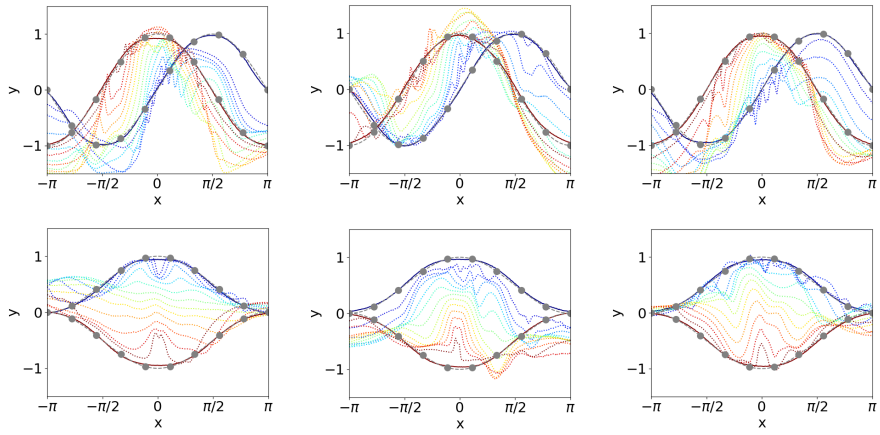

Figure 5: Transitions of approximated functions in the noise-modulated NNs by changing r in the range $[0, 1]$. Top: Transitions of approximations by three networks trained sequentially to approximate $y = \sin(x)$ and $y = \cos(x)$ by imposing white Gaussian noise $\mathrm{N}(0, 0.25^2)$ on either of the two sub-networks. Bottom: Transitions of approximations by three networks trained sequentially to approximate $y = (1 + \cos(x))/2$ and $y = -(1 + \cos(x))/2$ by imposing white Gaussian noise $\mathrm{N}(0, 0.25^2)$ on either of two sub-networks.

Clearly, these functions with different nonzero averages were also approximated successfully, even without bias to the output layer. Therefore, based on the results shown in Figure 4, the second issue was validated successfully.

4.3. Changing the noise intensity between the two sub-networks

Next, the interpolation between two functions approximated in two coprime sub-networks was investigated. Given the results in Section 4.2, no bias was applied to the output layer. Three of each of the 100 trained networks used to obtain the center and right graphs in Figure 4 were used for investigating how two approximations acquired in two coprime sub-networks are interpolated by changing where noise is added.

To smoothly change the area where noise was applied from one trained sub-network to the other one, a variable r is used to represent the noise intensity in

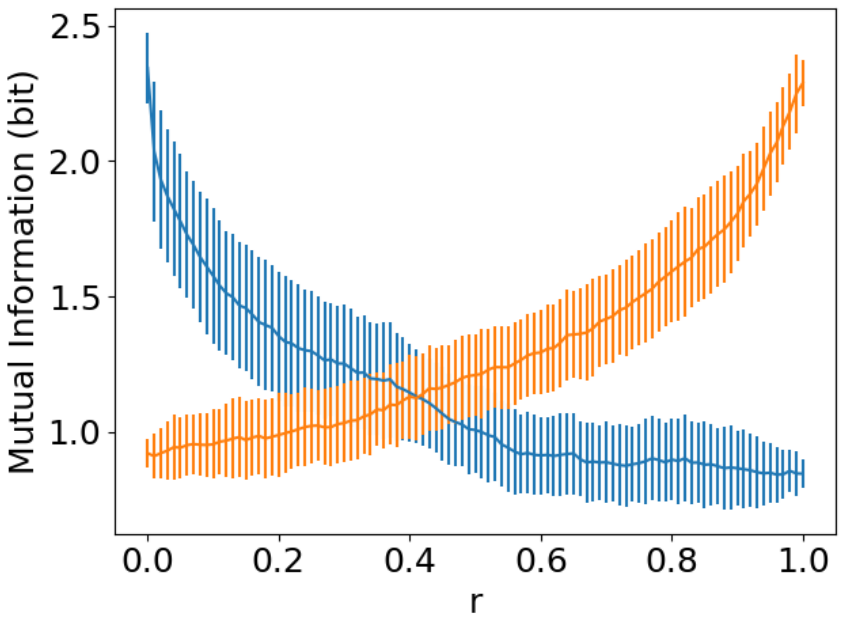

Figure 6: Mutual information between the network output and the target functions $y = \sin(x)$ and $y = \cos(x)$. The horizontal and vertical axes represent r and the mutual information, respectively. The blue and orange lines show mutual information between the outputs from the trained networks and $y = \sin(x)$ and $y = \cos(x)$, respectively. The mutual information with the two target functions increases/decreases monotonically in response to changes in r. Therefore, the functions obtained in the transition are deemed to interpolate the two functions.

the two sub-networks as follows:

$$\begin{aligned} r &\in [0,1] \subset \mathbb{R} \\ p_{i,j} &= \mathrm{N}(0, (0.25(1-r))^2), \forall i,j \in S_1 \quad (26) \\ p_{i,j} &= \mathrm{N}(0, (0.25r)^2), \forall i,j \in S_2. \quad (27) \end{aligned}$$

Note that $N(0,0^2)$ is used to denote the Dirac delta because $\delta = \lim_{\sigma \to 0} N(0,\sigma^2)$ holds. In particular, $r = 0$ or $r = 1$ corresponds to functionalizing only the first or the second sub-network, respectively.

Figure 5 shows how the approximation transitions in response to changes in r. The three graphs shown in each row were drawn using three of the 100 networks used to draw the center and right graphs in Figure 4. As shown, the transitions differ considerably even though these NNs were trained in the same manner and the sub-networks were used to approximate the same functions. This means that parametric redundancy has a strong influence on the transition characteristics. In addition, the functions that appear in the transitions have higher frequency than do those for $r = 0$ and $r = 1$. This implies that changing r induces highly nonlinear transitions, despite the relatively simple changes in noise intensity.

Figure 6 shows how mutual information with the two target functions $y = \sin(x)$ and $y = \cos(x)$ changes with r. Here, the 100 trained networks used to draw the center graph in Figure 4 were used for computing the average and the standard deviation of mutual information with the two target functions. As shown, the mutual information changed monotonically with r, therefore the transitions in the approximated functions obtained by changing r seem to be interpolations of the two target functions approximated in the two functionally coprime sub-networks.

In this section, it was confirmed that (i) the proposed NN exploits SR, namely a certain level of nonzero noise maximizes accuracy of approximation, (ii) adding noise to part of the network selectively functionalizes only that part as a trainable NN, and (iii) interpolations between target functions could be obtained by changing the noise intensities that were used to constitute func-

tionally coprime sub-networks. As such, the three points listed at the beginning of Section 4 were confirmed as functionalities of the noise-modulated NN proposed in this paper. Therefore, the core of this paper, namely the use of a noise-modulated NN to selectively functionalize sub-networks by exploiting SR, was validated.

5. Discussion

In Section 4.3, the parameter r was used to change two Gaussian distributions as shown in Eqs. 26 and 27. Although r provides simple interpolations, if we take noise distributions with zero-mean Gaussian distributions, then $\sum_i M_i$ parameters are originally needed to specify the standard deviations of the noise imposed on all hidden units. This means that there are many possibilities for changing the approximation provided by the proposed NN. Therefore, one can reason that a new target function could be approximated by changing only the noise parameters rather than the weight and bias parameters. Investigating this point is an important direction for future research. In addition, recent advances regarding training biologically plausible neural networks with multiple functions, such as those suggested in [52, 53], would suggest the possibility that the parameter "where noise is imposed in a network" of the noise-modulated NN can be discussed in terms of task representation. Based on the spatial noise distribution and its dynamics in a brain, considering biologically plausible adjustments in the parameter "where noise is imposed on the network" and deepening the discussion on this point is also an important direction for future research.

In terms of hardware implementation, the proposed NN is deemed to have suitable features. As shown in Figure 1, the proposed hidden unit is simple to implement and could be realized as an electrical circuit. However, the proposed network uses a probability density function describing the additional noise, which is generally deemed unknown when backpropagation is performed. To this issue, the proposed approach can be extended to approximate the proba-

bility density function, and to perform backpropagation by integrating kernel density estimation with a few modifications. In particular, when a uniform kernel function is used, the only modification required is to use two different threshold values in the step functions constituting a unit. The detailed theory explaining this aspect is described in Appendix A. In Section 4, $K = 100$ was used, and 200 step functions were used as a proposed hidden unit. An additional simulation was performed to validate this approach. Half of the 200 elements in the hidden unit were biased with a threshold, and half were biased with the negative of this threshold ($h = 0.18$, see Appendix A). Note that all other setting/parameters were the same. Target functions were also similarly set to $y = \sin(x)$ and $y = \cos(x)$. In addition, r was also set and varied as well in the training and the interpolating.

Figure 7 shows the result obtained using the proposed NN integrated with kernel density estimation, for the same problem whose results were shown in the top line of Figure 5. This figure shows that hidden units with two different step functions do not produce a less accurate approximation. In addition, from the comparison between Figure 5 and 7, one can see that transitions are also observed. This result seems to strongly support the validity of the above reasoning. Investigating possible hardware implementations will also be an important direction for future research.

How the proposed hidden unit reflects biological neural networks and their models is another important point of discussion. The McCulloch–Pitts model, also called the threshold logic unit, is known as the simplest mathematical model of a neuron and is widely used as a unit in ANNs. Although the proposed model consists of step functions that are equivalent to threshold logic units, the unit described in Eq.6 outputs the XOR of two units. XOR is the most popular example that requires nonlinear computation, and it seems to indicate that XOR should not be included in a hidden unit for biological plausibility. However, a very recent report [54] showed that a single neuron appears to be able to compute XOR by focusing on the measurement of dendritic action potentials of pyramidal neurons in the human cerebral cortex ex vivo. In addition to considering that

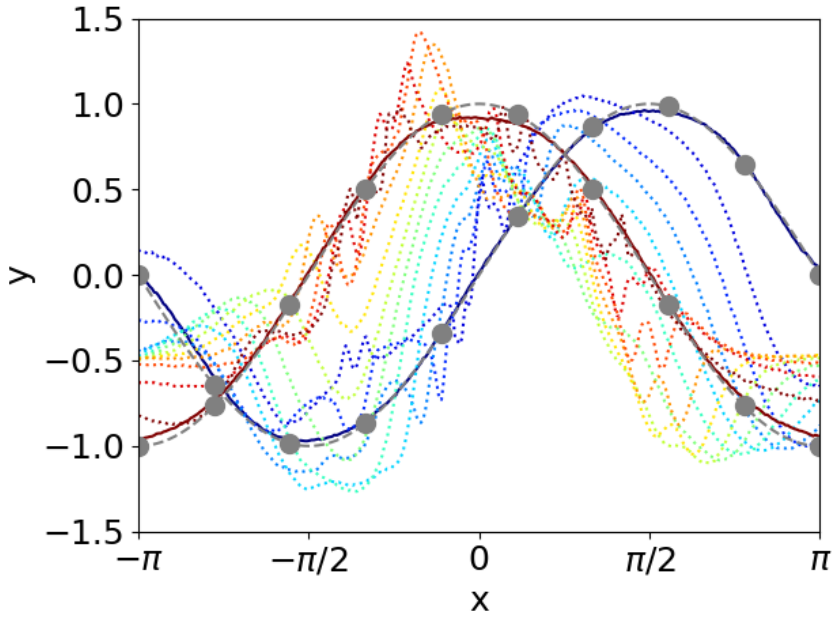

Figure 7: Transitions of approximated functions in noise-modulated NNs using two different step functions, where r is varied in the range $[0, 1]$. During training, a kernel density estimator is used instead of a probability density function for the noise distribution. Approximation of $y = \sin(x)$ and $y = \cos(x)$ is successful, and transitions similar to those in Figure 5 are observed.

XOR is computable in a single neuron, if it can be assumed that changes in an input to a threshold logic unit are sufficiently slow that the input is quasi-static and the output is ergodic, the implementation of the hidden unit becomes simpler than that shown in Figure 1. In particular, one McCulloch-Pitts neuron model and one XOR neuron model could be used to construct the proposed hidden unit by replacing the ensemble average with the time average. Further research is required, but these facts suggest that that the proposed model may be biologically plausible.

6. Conclusion

In this paper, a noise-modulated NN that exploits SR was introduced. In the proposed ANN, sub-networks can be selectively functionalized by adding noise to only part of the network. To this end, a new activation function was proposed. This activation function consists of step functions and a XOR operator. It presents null output and derivative in the absence of noise, but by adding noise, SR occurs and output and derivative can become non-zero. Simple simulations confirmed the occurrence of SR in the proposed NN by observing the distinctive unimodal curve in Figure 2, a typical feature of SR occurrence. It was then shown that the network can be partially and selectively functionalized by applying noise on part of the network, via the feature of the proposed hidden unit. It was confirmed that a sub-network in a noise-modulated NNs can be trained to approximate a target function by applying noise as is the case with normal NNs consisting of continuous activation functions. In addition, after training two coprime sub-networks with different target functions, interpolations were obtained by changing the noise intensity across sub-networks. Finally, the proposed NN has features that are worth investigating in future research, specifically (i) exploiting the parameter "where noise is imposed in a network" of the noise-modulated NN for adapting the approximation and/or expressing multiple tasks, (ii) the possibility of hardware implementation of the noise-modulated NN, and (iii) the possibility of biological plausibility of the

proposed hidden unit.

Appendix A. Integration with kernel density estimation

Let $\eta_1, \eta_2 \cdots \eta_n$ denote n independent identically distributed samples from an unknown probability density function $p(\xi)$. A kernel density estimator $\hat{p}(d) \approx p(\xi = d)$ is obtained as follows:

$$\hat{p}(d) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{d - \eta_i}{h}\right) \tag{A.1}$$

where K and h indicate a kernel function and a bandwidth. Assuming a uniform kernel function $K(u) = \frac{1}{2} 1_{\{|u| \leq 1\}}$ is used, $\hat{p}(d)$ is rewritten as follows:

$$\hat{p}(d) = \frac{1}{2h} \langle x \rangle \tag{A.2}$$

$$x = \begin{cases} 1 & if\ d - h \leq \eta\ \wedge\ d + h \geq \eta,\ \eta \overset{\text{iid}}{\sim} p(\xi) \\ 0 & otherwise \end{cases}. \tag{A.3}$$

By dividing the two conditions in Eq.A.4, $\langle x \rangle$ can be expressed as follows:

$$\langle x \rangle = \langle x_+ \rangle - \langle x_- \rangle \tag{A.4}$$

$$x_+ = \begin{cases} 1 & if\ d + h \geq \eta \overset{\text{iid}}{\sim} p(\xi) \\ 0 & otherwise \end{cases} \tag{A.5}$$

$$x_- = \begin{cases} 1 & if\ d - h \geq \eta \overset{\text{iid}}{\sim} p(\xi) \\ 0 & otherwise \end{cases}. \tag{A.6}$$

As a result, the kernel density estimator can be obtained by employing these two types of thresholding elements in the following way:

$$\hat{p}(d) = \frac{1}{2h} \left(\langle x_+ \rangle - \langle x_- \rangle \right) \approx p(\xi = d). \tag{A.7}$$

In addition, let us define $\langle \hat{z} \rangle$ as follows:

$$\begin{aligned} \langle \hat{z} \rangle &= \frac{1}{2} \left(\langle x_+ \rangle + \langle x_- \rangle \right) \\ &\approx P(d \geq \eta) = \int_{-\infty}^{d} p(\xi) d\xi. \end{aligned} \tag{A.8}$$

Therefore, Eqs.A.8 and A.7 approximate Eqs.4 and 5, respectively. In general, the bandwidth h must decrease if the number of samples n increases and $n \to \infty$, thus $h \to 0$. Therefore, both $\langle \hat{z} \rangle$ and $\hat{p}(d)$ converge to $P(d \geq \eta)$ and $p(\xi = d)$, respectively, because $n \to \infty$.

This approach can be easily extended to the case where the proposed activation function is defined in Eq.6-8. Let us consider two types of activation functions by simply biasing Eq.6 with $\pm h$ as follows:

$$z_+ = \phi_+(d) = \begin{cases} 1 & if\ d + h \geq \eta_1\ \dot{\vee}\ d + h \geq \eta_2 \\ 0 & otherwise \end{cases} \tag{A.9}$$

$$z_- = \phi_-(d) = \begin{cases} 1 & if\ d - h \geq \eta_1\ \dot{\vee}\ d - h \geq \eta_2 \\ 0 & otherwise. \end{cases} \tag{A.10}$$

The ensemble averages of z_+ and z_- are expressed by using $\langle x_+ \rangle$ and $\langle x_- \rangle$ in Eq.A.5,A.6 as follows:

$$\langle z_+ \rangle = \alpha \langle x_+ \rangle (1 - \langle x_+ \rangle) \tag{A.11}$$

$$\langle z_- \rangle = \alpha \langle x_- \rangle (1 - \langle x_- \rangle) \tag{A.12}$$

As a first approximation of Eq.7, let us define $\hat{\phi}(d)$ by

$$\hat{\phi}(d) = \frac{1}{2} (\langle z_+ \rangle + \langle z_- \rangle) \tag{A.13}$$

Eq.A.11-A.13 yields:

$$\begin{aligned}
\hat{\phi}(d) &= \frac{\alpha}{2}\left(\langle x_+\rangle\left(1-\langle x_+\rangle\right)+\langle x_-\rangle\left(1-\langle x_-\rangle\right)\right) \\
&= \frac{\alpha}{2}\left(\langle x_+\rangle+\langle x_-\rangle\right)-\frac{\alpha}{2}\left(\langle x_+\rangle^2+\langle x_-\rangle^2\right) \\
&= \alpha\hat{P}(d\geq\eta)-\frac{\alpha}{2}\left(\left(\langle x_+\rangle-\langle x_-\rangle\right)^2-2\langle x_+\rangle\langle x_-\rangle\right) \\
&= \alpha\hat{P}(d\geq\eta)-2\alpha\hat{P}(d\geq\eta)^2-2\langle x_+\rangle\langle x_-\rangle \\
&= \alpha\hat{P}(d\geq\eta)\left(1-\hat{P}(d\geq\eta)\right) \\
&\quad -\frac{\alpha}{4}\left(\langle x_+\rangle+\langle x_-\rangle\right)^2+\alpha\langle x_+\rangle\langle x_-\rangle \\
&= \alpha\hat{P}(d\geq\eta)\left(1-\hat{P}(d\geq\eta)\right) \\
&\quad -\frac{\alpha}{4}\left(\langle x_+\rangle^2-2\langle x_+\rangle\langle x_-\rangle+\langle x_+\rangle^2\right) \\
&= \alpha\hat{P}(d\geq\eta)\left(1-\hat{P}(d\geq\eta)\right)-\frac{\alpha}{4}\left(\langle x_+\rangle-\langle x_-\rangle\right)^2 \\
&= \alpha\hat{P}(d\geq\eta)\left(1-\hat{P}(d\geq\eta)\right)-\alpha h^2\hat{p}(d)^2 \qquad \text{(A.14)}
\end{aligned}$$

where $\hat{P}(d\geq\eta)$ and $\hat{p}(d)$ are an approximation of $P(d\geq\eta)$ and $p(\xi=d)$, respectively, based on the results of Eqs.A.8 and A.7. Again, because increasing the number of samples n requires decreasing h, the second term converges to zero as $\hat{P}(d\geq\eta)$ converges to $P(d\geq\eta)$ when $n\to\infty$. Therefore, $\hat{\phi}(d)$ is the appropriate approximation of $\bar{\phi}(d)$ in Eq.7.

As a first approximation of Eq.8, let us define $\hat{\phi}(d)'$ as follows:

$$\hat{\phi}(d)' = \frac{1}{2h}\left(\langle z_+\rangle-\langle z_-\rangle\right). \qquad \text{(A.15)}$$

This is transformed to:

$$\begin{aligned}
\hat{\phi}(d)' &= \frac{\alpha}{2h}\left(\langle x_+\rangle\left(1-\langle x_+\rangle\right)-\langle x_-\rangle\left(1-\langle x_-\rangle\right)\right) \\
&= \frac{\alpha}{2h}\left(\langle x_+\rangle-\langle x_-\rangle\right)-\frac{\alpha}{2h}\left(\langle x_+\rangle-\langle x_-\rangle\right)\left(\langle x_+\rangle+\langle x_-\rangle\right) \\
&= \frac{\alpha}{2h}\left(\langle x_+\rangle-\langle x_-\rangle\right)\left(1-\left(\langle x_+\rangle+\langle x_-\rangle\right)\right) \\
&= \alpha\left(1-2\hat{P}(d\geq\eta)\right)\hat{p}(d). \qquad \text{(A.16)}
\end{aligned}$$

Therefore, $\hat{\phi}(d)'$ is the appropriate approximation of $\bar{\phi}(d)'$ in Eq.8.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 18H01410 and 19K22875.

References

[1] C. Koch, Biophysics of Computation: Information Processing in Single Neurons, Computational Neuroscience Series, Oxford University Press, 2004.

[2] A. Destexhe, M. Rudolph-Lilith, Neuronal Noise, 1st Edition, Springer Series in Computational Neuroscience, Springer US, 2012. doi:10.1007/978-0-387-79020-6.

[3] A. A. Faisal, L. P. J. Selen, D. M. Wolpert, Noise in the nervous system, Nature Reviews Neuroscience 9 (4) (2008) 292–303. doi:10.1038/nrn2258.

[4] R. M. Birn, The role of physiological noise in resting-state functional connectivity, NeuroImage 62 (2) (2012) 864–870. doi:https://doi.org/10.1016/j.neuroimage.2012.01.016.

[5] L. Gammaitoni, P. Hänggi, P. Jung, F. Marchesoni, Stochastic resonance, Reviews of Modern Physics 70 (1998) 223–287. doi:10.1103/RevModPhys.70.223.

[6] M. D. McDonnell, N. G. Stocks, C. E. M. Pearce, D. Abbott, Stochastic Resonance: From Suprathreshold Stochastic Resonance to Stochastic Signal Quantization, Cambridge University Press, Cambridge, 2008. doi:DOI:10.1017/CBO9780511535239.

[7] R. Benzi, A. Sutera, A. Vulpiani, The mechanism of stochastic resonance, Journal of Physics A: Mathematical and general 14 (1981) 453–457.

[8] A. Longtin, A. Bulsara, F. Moss, Time-interval sequences in bistable systems and the noise-induced transmission of information by sensory neurons,

Phys. Rev. Lett. 67 (1991) 656–659. doi:10.1103/PhysRevLett.67.656. URL http://link.aps.org/doi/10.1103/PhysRevLett.67.656

[9] B. J. Gluckman, T. I. Netoff, E. J. Neel, W. L. Ditto, M. L. Spano, S. J. Schiff, Stochastic resonance in a neuronal network from mammalian brain, Physical Review Letters 77 (1996) 4098–4101. doi:10.1103/PhysRevLett.77.4098.

[10] E. Manjarrez, G. Rojas-Piloni, I. Méndez, A. Flores, Stochastic resonance within the somatosensory system: Effects of noise on evoked field potentials elicited by tactile stimuli, The Journal of Neuroscience 23 (6) (2003) 1997.

[11] J. Douglass, L. Wilkens, E. Pantazelou, F. Moss, Noise enhancement of information transfer in crayfish mechanoreceptors by stochastic resonance, Nature 365 (6444) (1993) 337–340.

[12] K. Wiesenfeld, D. Pierson, E. Pantazelou, C. Dames, F. Moss, Stochastic resonance on a circle, Physical Review Letters 72 (1994) 2125–2129. doi:10.1103/PhysRevLett.72.2125.

[13] P. Cordo, J. T. Inglis, S. Verschueren, J. J. Collins, D. M. Merfeld, S. Rosenblum, S. Buckley, F. Moss, Noise in human muscle spindles, Nature 383 (6603) (1996) 769–770. doi:10.1038/383769a0.

[14] J. Levin, J. Miller, Broadband neural encoding in the cricket cercal sensory system enhanced by stochastic resonance, Nature 380 (1996) 165–168. doi:10.1038/380165a0.

[15] A. R. Bulsara, F. E. Moss, Single neuron dynamics: noise-enhanced signal processing, in: IEEE International Joint Conference on Neural Networks, 1991, pp. 420–425 vol.1. doi:10.1109/IJCNN.1991.170437.

[16] M. D. McDonnell, D. Abbott, What is stochastic resonance? definitions, misconceptions, debates, and its relevance to biology, PLoS Comput Biol 5 (5) (2009) e1000348.

[17] S. Ikemoto, F. DallaLibera, K. Hosoda, Noise-modulated neural networks as an application of stochastic resonance, Neurocomputing 277 (2018) 29 – 37. doi:https://doi.org/10.1016/j.neucom.2016.12.111.

[18] M. D. McDonnell, L. M. Ward, The benefits of noise in neural systems: bridging theory and experiment, Nature Reviews Neuroscience 12 (7) (2011) 415–425. doi:10.1038/nrn3061.

[19] M. Ozer, M. Perc, M. Uzuntarla, E. Koklukaya, Weak signal propagation through noisy feedforward neuronal networks, Neuroreport 21 (5) (2010) 338–343. doi:10.1097/wnr.0b013e328336ee62.
URL http://europepmc.org/abstract/MED/20186108http://content.wkhealth.com/linkback/openurl?issn=0959-4965&volume=21&issue=5&spage=338https://doi.org/10.1097/WNR.0b013e328336ee62

[20] K. Ishimura, A. Schmid, T. Asai, M. Motomura, Stochastic resonance induced by internal noise in a unidirectional network of excitable fitzhugh-nagumo neurons, Nonlinear Theory and Its Applications, IEICE 7 (2) (2016) 164–175. doi:10.1587/nolta.7.164.

[21] B. Vázquez-Rodríguez, A. Avena-Koenigsberger, O. Sporns, A. Griffa, P. Hagmann, H. Larralde, Stochastic resonance at criticality in a network model of the human cortex, Scientific Reports 7 (1) (2017) 13020. doi:10.1038/s41598-017-13400-5.

[22] T. Shimokawa, A. Rogel, K. Pakdaman, S. Sato, Stochastic resonance and spike-timing precision in an ensemble of leaky integrate and fire neuron models, Physical Review E 59 (3) (1999) 3461–3470. doi:10.1103/PhysRevE.59.3461.
URL https://link.aps.org/doi/10.1103/PhysRevE.59.3461

[23] H. C. Tuckwell, J. Jost, B. S. Gutkin, Inhibition and modulation of rhythmic neuronal spiking by noise, Physical Review E 80 (3) (2009) 031907. doi:10.1103/PhysRevE.80.031907.

[24] M. Ozer, M. Perc, M. Uzuntarla, Stochastic resonance on newman–watts networks of hodgkin–huxley neurons with local periodic driving, Physics Letters A 373 (10) (2009) 964–968. doi:https://doi.org/10.1016/j.physleta.2009.01.034.
URL http://www.sciencedirect.com/science/article/pii/S0375960109000905

[25] D. Guo, Q. Wang, M. Perc, Complex synchronous behavior in interneuronal networks with delayed inhibitory and fast electrical synapses, Physical Review E 85 (6) (2012) 061905. doi:10.1103/PhysRevE.85.061905.

[26] M. Uzuntarla, E. Barreto, J. J. Torres, Inverse stochastic resonance in networks of spiking neurons, PLOS Computational Biology 13 (7) (2017) e1005646. doi:10.1371/journal.pcbi.1005646.

[27] A. S. Pikovsky, J. Kurths, Coherence resonance in a noise-driven excitable system, Physical Review Letters 78 (5) (1997) 775–778. doi:10.1103/PhysRevLett.78.775.

[28] G. Nicolis, C. Nicolis, D. McKernan, Stochastic resonance in chaotic dynamics, Journal of Statistical Physics 70 (1) (1993) 125–139. doi:10.1007/BF01053958.

[29] S. Nobukawa, H. Nishimura, T. Yamanishi, Chaotic resonance in typical routes to chaos in the izhikevich neuron model, Scientific Reports 7 (1) (2017) 1331. doi:10.1038/s41598-017-01511-y.

[30] S. Bezrukov, I. Vodyanoy, Stochastic resonance in non-dynamical systems without response thresholds, Nature 385 (6614) (1997) 319–321. doi:10.1038/385319a0.

[31] L. Gammaitoni, Stochastic resonance and the dithering effect in threshold physical systems, Physical Review E 52 (1995) 4691–4698. doi:10.1103/PhysRevE.52.4691.

[32] B. Kosko, S. Mitaim, Stochastic resonance in noisy threshold neurons, Neural Networks 16 (5-6) (2003) 755–761. doi:10.1016/S0893-6080(03)00128-X.

[33] P. E. Greenwood, U. U. Müller, L. M. Ward, Soft threshold stochastic resonance, Physical Review E 70 (2004) 051110. doi:10.1103/PhysRevE.70.051110.

[34] J. Collins, C. Chow, T. Imhoff, Stochastic resonance without tuning, Nature 376 (6537) (1995) 236–238.

[35] N. G. Stocks, Suprathreshold stochastic resonance in multilevel threshold systems, Physical Review Letters 84 (2000) 2310–2313. doi:10.1103/PhysRevLett.84.2310.

[36] J. F. Mejias, J. J. Torres, Emergence of resonances in neural systems: The interplay between adaptive threshold and short-term synaptic plasticity, PLOS ONE 6 (4) (2011) 10.1371/annotation/1fe001e2–eb2a–4891–a947–a0105812911b. doi:10.1371/annotation/1fe001e2-eb2a-4891-a947-a0105812911b.
URL https://doi.org/10.1371/annotation/1fe001e2-eb2a-4891-a947-a0105812911b

[37] J. J. Torres, J. Marro, J. F. Mejias, Can intrinsic noise induce various resonant peaks?, New Journal of Physics 13 (5) (2011) 053014. doi:10.1088/1367-2630/13/5/053014.
URL http://dx.doi.org/10.1088/1367-2630/13/5/053014

[38] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533–536. doi:10.1038/323533a0.

[39] L. Holmstrom, P. Koistinen, Using additive noise in back-propagation training, IEEE Transactions on Neural Networks 3 (1) (1992) 24–38. doi:10.1109/72.105415.

[40] C. M. Bishop, Training with noise is equivalent to tikhonov regularization, Neural Computation 7 (1) (1995) 108–116. doi:10.1162/neco.1995.7.1.108.

[41] H. Guang-Bin, Z. Qin-Yu, K. Z. Mao, S. Chee-Kheong, P. Saratchandran, N. Sundararajan, Can threshold networks be trained directly?, IEEE Transactions on Circuits and Systems II: Express Briefs 53 (3) (2006) 187–191. doi:10.1109/TCSII.2005.857540.

[42] T. Downs, R. Gaynier, The use of random weights for the training of multilayer networks of neurons with heaviside characteristics, Mathematical and Computer Modelling 22 (10) (1995) 53 – 61. doi:https://doi.org/10.1016/0895-7177(95)00180-A.

[43] P. Barlett, T. Downs, Using random weights to train multilayer networks of hard-limiting units, Neural Networks, IEEE Transactions on 3 (2) (1992) 202–210. doi:10.1109/72.125861.

[44] D. Toms, Training binary node feedforward neural networks by back propagation of error, Electronics Letters 26 (1990) 1745–1746(1).

[45] E. Corwin, A. Logar, W. Oldham, An iterative method for training multilayer networks with threshold functions, Neural Networks, IEEE Transactions on 5 (3) (1994) 507–508. doi:10.1109/72.286926.

[46] E. Wilson, Backpropagation learning for systems with discrete-valued functions, in: Proceedings of the World Congress on Neural Networks, Vol. 3, 1994, pp. 332–339.

[47] A. Sapkal, U. V. Kulkarni, Modified backpropagation with added white gaussian noise in weighted sum for convergence improvement, Procedia Computer Science 143 (2018) 309–316. doi:https://doi.org/10.1016/j.procs.2018.10.401.

[48] S. Kumar, A. Kumar, R. K. Jha, A novel noise-enhanced back-propagation technique for weak signal detection in neyman–pearson framework, Neural Processing Letters (Mar 2019). doi:10.1007/s11063-019-10013-z.

[49] K. Audhkhasi, O. Osoba, B. Kosko, Noise benefits in backpropagation and deep bidirectional pre-training, in: The 2013 International Joint Conference on Neural Networks (IJCNN), 2013, pp. 1–8. doi:10.1109/IJCNN.2013.6707022.

[50] K. Audhkhasi, O. Osoba, B. Kosko, Noise-enhanced convolutional neural networks, Neural Networks 78 (2016) 15–23. doi:https://doi.org/10.1016/j.neunet.2015.09.014.

[51] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: International Conference on Learning Representations (ICLR), 2015.

[52] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks, Proceedings of the National Academy of Sciences 114 (13) (2017) 3521. doi:10.1073/pnas.1611835114.
URL http://www.pnas.org/content/114/13/3521.abstract

[53] G. R. Yang, M. R. Joglekar, H. F. Song, W. T. Newsome, X.-J. Wang, Task representations in neural networks trained to perform many cognitive tasks, Nature Neuroscience 22 (2) (2019) 297–306. doi:10.1038/s41593-018-0310-2.
URL http://www.scopus.com/inward/record.url?scp=85060098347&partnerID=8YFLogxKhttp://www.scopus.com/inward/citedby.url?scp=85060098347&partnerID=8YFLogxK

[54] A. Gidon, T. A. Zolnik, P. Fidzinski, F. Bolduan, A. Papoutsi, P. Poirazi, M. Holtkamp, I. Vida, M. E. Larkum, Dendritic action potentials and computation in human layer 2/3 cortical neurons, Science 367 (6473)

(2020) 83. doi:10.1126/science.aax6239.

URL http://science.sciencemag.org/content/367/6473/83.abstract