A CLASS OF FAST COMPLEX DOMAIN NEURAL NETWORKS FOR SIGNAL PROCESSING APPLICATIONS

Aurelio Uncini and Francesco Piazza

Dipartimento di Elettronica e Automatica- Università di Ancona Italy
via Brecce Bianche, I-60131 Ancona Italy
Tel. : +39 71 220 4841; Fax: +39 71 220 4464;
e-mail: aurel@eealab.unian.it url: http://nnsp.eealab.unian.it/Uncini_a

Abstract - In this paper, we study the properties of a new kind of complex domain artificial neural networks called complex adaptive spline neural networks (CASNN), which are able to adapt their activation functions by varying the control points of a Catmull-Rom cubic spline. This new kind of neural network can be implemented as a very simple structure being able to improve the generalization capabilities using few training epochs. Due to its low architectural complexity this network can be used to cope with several nonlinear DSP problem at high throughput rate.

1. Introduction

RECENTLY in the neural network community, a new interest in adaptive activation functions has arisen. In fact, such a strategy seems to provide better fitting properties with respect to classical architectures with sigmoidal neurons.

The simplest solution we can imagine consists in involving gain a and slope b of the sigmoid $a(1-e^{-bx})/(1+e^{-bx})$ in the learning process [1]. A different approach is based on the use of polynomial functions [2], which allow to reduce the size of the network and, in particular, the connection complexity; in fact, the digital implementation of the activation function through a LUT (look-up-table) keeps the overall complexity under control. Drawbacks with this solution arise with the non-boundedness of the function (non-squashing) and with the adaptation of the coefficients in the learning phase. In [3] the direct adaptation of the LUT coefficients is proposed: this time the problems are a difficult learning process due to the large number of free parameters and the lack of smoothness of the neuron's output. These are also the main reasons for the introduction of Hermite polynomials as substitutes for the so called supersmoother in the Projection Pursuit Learning approach of Hwang et al. [4].

The solution discussed in this paper makes use of spline based activation functions whose shape can be modified through some control points. In fact, our main goal is to demonstrate that an intelligent use of the activation function can reduce hardware complexity [6-7], while, at the same time, improving generalization ability.

The main advantages of this innovative structure, very useful for nonlinear adaptive signal processing, are: 1) the training sequence may be shorter than that required by the classical MLP; 2) the architecture is general and, unlike others approaches, it does not require a specific design; 3) the low hardware complexity (low overhead with respect to a simple adaptive linear combiner) makes it suitable for high speed data transmissions using a DSP device.

So, after seeing cubic splines theory and discussing we presents the backpropagation-like learning algorithm for the CASNN. In Section 3, we'll expose the results of a simulation on complex signal processing problems.

2. The Adaptive Spline Neural Networks

Regularization theory offers a way to choose a compromise between data fitting and smoothness, through a regularizing term. According to [5], the final aspect of the approximating function is

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i \sum_{j=1}^{n} \mu_j \mathrm{G}(x_j - x_{ij}) \quad ; \tag{1}$$

where the symbol x_{ij} indicates the j-th component of the i-th component of the input vector $\mathbf{x}_i$.

An important extension of the previous function involves a change in the coordinates system for the space; as reported in [5], the choice of a proper *point of view* can be important when representing a multivariate function as the sum of a number of functions equal to the dimension of the input space. Calling $\mathbf{w}_j$, j=1,...,n, the vectors which determine the axis of the new system and α_{ij} the new centers in such a system, we can write

$$f(\mathbf{x}) = \sum_{j=1}^{n} \mu_j \sum_{i=1}^{N} c_i \mathrm{G}(\mathbf{w}_j \mathbf{x} - \alpha_{ij}) \quad ; \tag{2}$$

inverting the order of summation of equation (1). The equation (2) is the starting point of our considerations.

As developed in [18] the idea of the adaptive spline based neural network (ASNN) consists in realizing a neuron with a more complex activation function than the sigmoid, able to reproduce the shape of a whole cubic spline along the directions specified by $\mathbf{w}_j$, j=1,...,n; :

$$\varphi_j(\mathbf{w}_j \mathbf{x}) = \sum_{i=1}^{N} c_i |\mathbf{w}_j \mathbf{x} - \alpha_{ij}|^3 \quad j=1,...,n \tag{3}$$

Then $f(\mathbf{x})$ can be written as:

$$f(\mathbf{x}) = \sum_{j=1}^{n} \mu_j \varphi_j(\mathbf{w}_j \mathbf{x}) \quad , \tag{4}$$

Now μ_j, and the components of $\mathbf{w}_j$, for all the indexes j, can be found by backpropagation, thus solving the problem of the optimal set of the parameters μ_j and of the ideal system of coordinates (although we can get trapped in local minima).

As we have previously anticipated, the goal is to give a global approximation of the curve drawn by the functions φ_j, j=1,...,n, using a structure as tractable as possible. In equation (3) we find a spline with N+1 tracts: each of them is described by a different combination of the coefficients c_i, because of the change in the sign of the kernels at α_{ij}. We have chosen to represent the activation functions through the concatenation of even more local spline basis functions, controlled by only four coefficients. As we want to keep the cubic characteristic, we have used a Catmull-Rom cubic spline [8]. Using this type of spline we could exactly reproduce function (3), but, of course, this is not the cheapest solution so we'll take a

different approach. Referring to Figure 1, the *i*-th tract is expressed as

$$\mathbf{F}_i(u)=\begin{bmatrix}F_{xi}(u)\\F_{yi}(u)\end{bmatrix}=\frac{1}{2}\begin{bmatrix}u^3 & u^2 & u & 1\end{bmatrix}\begin{bmatrix}-1 & 3 & -3 & 1\\ 2 & -5 & 4 & -1\\ -1 & 0 & 1 & 0\\ 0 & 2 & 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{Q}_i\\ \mathbf{Q}_{i+1}\\ \mathbf{Q}_{i+2}\\ \mathbf{Q}_{i+3}\end{bmatrix}, \qquad (4)$$

where $u\in[0,1]$ and $\mathbf{Q}=(q_x,q_y)$. Such a spline interpolates the points $\mathbf{Q}_{i+1}$ (u=0) and $\mathbf{Q}_{i+2}$ (u=1) and has a continuous first derivative, which is useful for the backpropagation-like learning algorithm. The second derivative is not continuous only at the knots. In general, equation (4) represents a curve: to obtain a function we have ordered the x-coordinates according to the rule $q_{x,i} < q_{x,i+1} < q_{x,i+2} < q_{x,i+3}$.

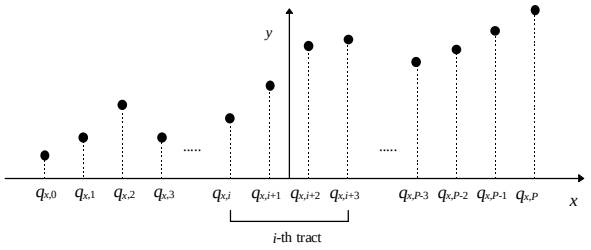

Figure 1. Control points of the Catmull-Rom spline based activation function: along the *x*-axis there is a fixed step Δx. The *i*-th tract starts with $q_{x,i}$ and ends with $q_{x,i+3}$, but the controlled interval is only between $q_{x,i+1}$ and $q_{x,i+2}$.

To find the value of the local parameter u, we have to solve the equation $F_{x,i}(u)=x_0$, where x_0 is the activation of the neuron: this is a third degree equation, whose solution can make the numerical burden of the learning algorithm heavier.

The easiest alternative consists in setting the control points uniformly spaced along the x-axis (Δx is the step): this choice allows to reduce the third degree polynomial $F_{x,i}(u)$ to a first degree polynomial and the equation for u becomes linear.

$$F_{x,i}(u)=u\Delta x+q_{x,i+1}\ . \qquad (5)$$

Now we can calculate the output of the neuron by $F_{y,i}(u)$. There is another reason not to make the x-coordinates of the control points adaptive: in fact, as we have pointed out in the introduction, a too large number of free parameters is the main cause for the overfitting of the training samples: so, if we use many tracts in building the activation function and let them move freely, the neural model will fit also the noise. Then the fixed parameter Δx can be one of the key tool for smoothness control.

As we have decided to adapt only the y-coordinates of the spline knots, they must be initialized them before starting the backpropagation-style learning: to this aim, we take, along the x-axis, P+1 uniformly spaced samples from a sigmoid or from another function assuring universal approximation capability that's why we use sometimes the acronym GS, standing for *Generalized Sigmoid*. Outside the sampling interval the neuron's output will be held constant at the values $q_{y,1}$, for the negative x-coordinate, and $q_{y,P-1}$ for the positive x. In the following, for the sake of simplicity, we'll indicate the y coordinates of the control points without the index y.

3. The Complex Domain ASNN and Learning Algorithm

The advantage of using complex-valued NNs instead of a real-valued NN counterpart fed with a pair of real values is well know [9-10]. In complex-valued neural networks one of the main problem to deal with, is the complex domain activation function, whose most suitable features have been suggested in [11]. Let $F(S)$ be the complex activation function with $S\in\mathbf{C}$ defined as the complex linear combiner output; the main constraints that $F(S)$ should satisfy are:

1) $F(S)$ should be non linear and bounded;
2) in order to derive the backpropagation algorithm the partial derivatives of $F(S)$ should exist and be bounded;
3) because of the *Liouville's* theorem $F(S)$ should not be an analytic function.

According to the previous properties, one possible choice, proposed in [12-13], consists on the superposition of real and imaginary activation functions $F(S)=f_{Re}(\mathrm{Re}[S])+jf_{Im}(\mathrm{Im}[S])$; where the functions $f_{\mathrm{Re}}(\bullet)$ and $f_{\mathrm{Im}}(\bullet)$, can be simple real-valued sigmoids or more sophisticated adaptive functions.

Using a formalism similar to the one introduced in Widrow and Lehr in [14], and following a development similar to [9], [12-13], for the synaptic weights, the learning algorithm is now extended to the spline's control points.

Considering M total layers and indicating each of them with the index l, l=1,...,M, we can find the span a_k and the local variable u_k by

$$\begin{aligned} z^{(l)}_{k\,\mathrm{Re}}&=\frac{\mathrm{Re}\left[S^{(l)}_k\right]}{\Delta x}+\frac{N-2}{2} & z^{(l)}_{k\,\mathrm{Im}}&=\frac{\mathrm{Im}\left[S^{(l)}_k\right]}{\Delta x}+\frac{N-2}{2}\\ a^{(l)}_{k\,\mathrm{Re}}&=\left\lfloor z^{(l)}_{k\,\mathrm{Re}}\right\rfloor & a^{(l)}_{k\,\mathrm{Im}}&=\left\lfloor z^{(l)}_{k\,\mathrm{Im}}\right\rfloor\ , \qquad (6)\\ u^{(l)}_{k\,\mathrm{Re}}&=z^{(l)}_{k\,\mathrm{Re}}-a^{(l)}_{k\,\mathrm{Re}} & u^{(l)}_{k\,\mathrm{Im}}&=z^{(l)}_{k\,\mathrm{Im}}-a^{(l)}_{k\,\mathrm{Im}}\end{aligned}$$

where the symbol $\lfloor\cdot\rfloor$ is the floor operator. We find for each neuron j the local tract a_j which s_j belongs to, and the local coordinate u_j. These expressions lead to neuron structure reported in Figure 2.

Complex Backpropagation Learning Algorithm

As in [9], [12-13], for the synaptic weights, the learning algorithm is now extended to the spline's control points.

$$E^{(l)}_k=\begin{cases}D_k-X^{(l)}_k & l=M\\ \sum_{m=1}^{N_{l+1}}\Delta^{(l+1)}_m W^{*(l+1)}_{mk} & l=M-1,\ldots,1\end{cases}$$

$$\Delta^{(l)}_k=\mathrm{Re}\left[E^{(l)}_k\right]\left(\left.\frac{d\,\mathrm{Re}\left[F^{(l)}_{k,a^{(l)}_k}(u)\right]}{du}\right|_{u=u^{(l)}_{k\mathrm{Re}}}\right)\frac{1}{\Delta x}+$$

$$+\,j\,\mathrm{Im}\left[E^{(l)}_k\right]\left(\left.\frac{d\,\mathrm{Im}\left[F^{(l)}_{k,a^{(l)}_k}(u)\right]}{du}\right|_{u=u^{(l)}_{k\mathrm{Im}}}\right)\frac{1}{\Delta x}\ , \qquad (7)$$

$$W^{(l)}_{kj}[p+1]=W^{(l)}_{kj}[p]+2\mu_w\Delta^{(l)}_k X^{(l-1)}_j$$

with $0\le k\le N_l$ and $0\le j\le N_{l-1}$.

The adaptation of the control points is ruled by

$$q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Re}}[p+1] = q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Re}}[p] + 2\mu_q \,\mathrm{Re}\left[E^{(l)}_k\right]\left(\frac{\partial \mathrm{Re}\left[F^{(l)}_{k,a^{(l)}_k}\right]}{\partial q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Re}}}\right) =$$

$$= q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Re}}[p] + 2\mu_q e^{(l)}_k c^{(l)}_{k,m\mathrm{Re}}(u^{(l)}_k) \quad ; \qquad (14)$$

$$q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Im}}[p+1] = q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Im}}[p] + 2\mu_q \,\mathrm{Im}\left[E^{(l)}_k\right]\left(\frac{\partial \mathrm{Im}\left[F^{(l)}_{k,a^{(l)}_k}\right]}{\partial q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Im}}}\right) =$$

$$= q^{(l)}_{k,(a^{(l)}_k+m)\mathrm{Im}}[p] + 2\mu_q e^{(l)}_k c^{(l)}_{k,m\mathrm{Im}}(u^{(l)}_k)$$

with the patch index m=0, ..., 3. The adaptation rates are μ_w for the connection weights and biases and μ_q for the control points. The control point with index 0, 1, (N-1) and N are fixed.

4. Experimental Results: *non linear QAM channel equalization*

The block diagram of the radio link used in our experiments is depicted in Figure 4. The complex input data sequence $D(k)$, represents the points of the QAM constellation. The function $g(t)$ represents the modulator filter impulse response: in the simulation we use a square-root of a raised-cosine having a *roll-off* factor α equal to 0.5, and the over-sampling factor M is chosen equal to 3. The $g(t)$ filter length is equal to $5M$ taps.

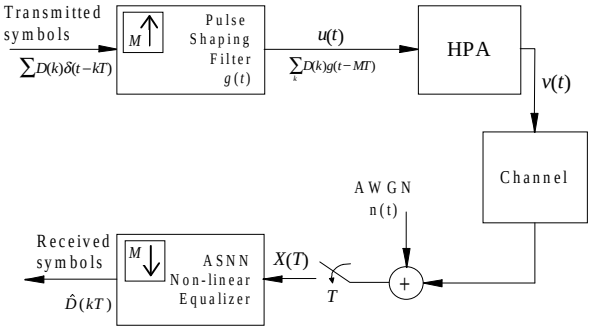

Figure 4. Block diagram of the digital radio link using a complex ASNN as nonlinear equalizer.

The model for the HPA, described in [17], is characterized by the expression:

$$v(t) = \frac{2u(t)}{1+|u(t)|^2}\exp\left[j\Phi_0 \frac{2|u(t)|^2}{1+|u(t)|^2}\right] \quad ; \qquad (8)$$

The input-output HPA (memoryless) response is described by the AM-AM response, represented by the module, and the AM-PM response represented by the phase. This description assumes, for convenience, that the maximum possible HPA input power $W_{in}=|u(t)|^2$ is equal to 1W, and the maximum shift is $\Phi_0 = \frac{\pi}{6}$, which are typical values [16].

In the simulations the transmission of 16-QAM signals are considered. The maximum input power to the HPA W_{inMAX} is very close to the saturation point, and fixed equal to -1 dB.

Three complex-valued equalizers have been tested: complex-valued linear combiner (A15_1); complex-value standard multilayer neural network with one hidden layer composed by 10 sigmoidal neurons and linear output (N15_10_1); CASNN composed by only one complex GS neuron (S15_1).

The training set consists of 6144 (2048xM) input samples, $v(t)+n(t)$ in the scheme of Figure 4, corresponding to 2048 target QAM symbols. Since the neural network output are the QAM complex constellation points, the network performs also the down-sampling conversion. For each epoch, a different realization of white zero-mean Gaussian noise $n(t)$ is added, to obtain a S/N equal to 20 dB.

During the learning phase, the S15_1 and A15_1 networks are trained for 100 epochs, while, in order to reach a suitable convergence, the standard MLP N15_10_1 is trained for 10^4 epochs. Both adaptation rates μ_q and μ_w for the S15_1 are chosen to be equal to 0.001; the same value is used for the MLP N15_10_1, while for the adaptive linear combiner A15_1 μ_w is equal to 0.0001.

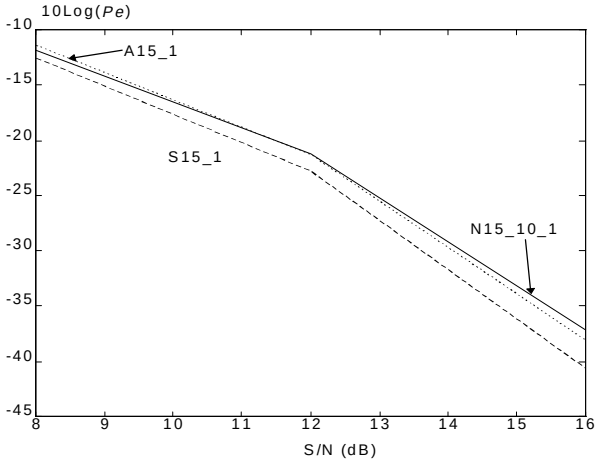

Figure 5. The symbols error probability (Pe) plotted vs. signal to noise ratio (S/N) at the input of the receiver using various equalization schemes.

Several tests using different networks initialization weights have been carried out. In order to evaluate in a realistic way the radio link performances, the symbol error probability (*Pe*) is computed. Figure 5 reports the *Pe* values vs. the S/N, both expressed in dB. From this figure we can observe that the proposed approach leads to significant improvements not only with respect to the classical linear adaptive filter, but also with respect to the already known sigmoid MLP based equalization technique.

5. Conclusions

Complex domain neural network architecture, based on adaptive Catmul-Rom splines activation function has been proposed. Derived from the standard backpropagation the learning algorithm for the new CASNNs has been derived.

Experimental results in QAM equalization problem, demonstrate the advantage in CASSN, since the network reduces to a single complex neuron. Moreover, the reduced complexity is responsible for the shorter adaptation phase in terms of training epochs, as experimentally observed. Comparing our technique with classical linear approaches, we can notice that there is a low implementation overhead with respect to the adaptive linear filter, but with a significant improvement in the performance, both in terms of MSE and symbol error probability.

References

[1] C. T. Chen, W. D. Chang, "A Feedforward Neural Network with Function Shape Autotuning", *Neural Networks*, Vol.9, No 4, pp. 627-641, June 1996.

[2] Piazza, A. Uncini, M. Zenobi, "Artificial Neural Networks with Adaptive Polynomial Activation Function", *in Proceedings of IJCNN*, Beijing, Cina, pp. II-343-349, Nov. 1992.

[3] F. Piazza, A. Uncini, M. Zenobi, "Neural Networks with Digital LUT Activation Function", in *Proceedings of IJCNN*, Nagoya, Japan, pp. II-1401-1404, 1993.

[4] Hwang, J-N, Lay, S-R, Maechler, M., Martin R. D., Schimert, J. "Regression Modeling in Back-Propagation and Projection Pursuit Learning" *IEEE Transactions on Neural Networks*, **5**(2), 342-353.

[5] Poggio, T., Girosi, F. "Networks for Approximation and Learning". *Proceedings of the IEEE*, **78(**9), 1481-1497, 1990.

[6] Guarnieri, S., Piazza, F., Uncini, A, "Multilayer Neural Networks with Adaptive Spline-based Activation Functions" In *Proceedings of the International Neural Network Society Annual Meeting WCNN 95*, Washington D.C., USA, I695-I699, 1995.

[7] Campolucci, P., Capparelli, F., Guarnieri, S., Piazza, F., Uncini, "A. Neural Networks with Adaptive Spline Activation Function" In *Proceedings of IEEE MELECON 96*, Bari Italy, 1442-1445. 1996.

[8] E. Catmull, R. Rom, "A Class of Local Interpolating Splines", in R. E. Barnhill, R. F. Riesenfeld (ed.), *Computer Aided Geometric Design*, Academic Press, NewYork, 1974, pp. 317-326.

[9] N. Benvenuto, M. Marchesi, F. Piazza and A. Uncini "A Comparison between Real and Complex valued Neural Networks in Communication Applications", *Proc. of Intern. Conference on Neural Networks*, ICANN91, Espoo, Finland, June 1991.

[10] H. Leung, S. Haykin, "The Complex Backpropagation Algorithm", *IEEE Trans Acoust. Speech and Signal Process*, Vol.ASSP-39, pp.2101-2104, Sept. 1991.

[11] S. Haykin, "Adaptive Filter Theory" Third Edition, Prentice Hall ed., 1996.

[12] N. Benvenuto, M. Marchesi, F. Piazza, A. Uncini, "Non Linear Satellite Radio Links Equalized using Blind Neural Networks", *in Proceedings of IEEE International Conference on Acoustic Speech and Signal Processing - ICASSP91*, May 14-17, Toronto, Canada, 1991.

[13] N. Benvenuto and F. Piazza "On the Complex Backpropagation Algorithm", *IEEE Trans Signal Processing*, Vol.40, pp.967-969, Apr. 1992.

[14] Widrow, M. Lehr, "30 Years of Adaptive Neural Networks: Perceptron, Adaline and Backpropagation", *in Proceedings of the IEEE*, Vol.78, No.9, Sept. 1990.

[15] Back, A. D., Tsoi, A. C. "A simplified gradient algorithm for IIR synapse Multilayer Perceptron" *Neural Networks*, **5**, 456-462, 1993.

[16] S. Pupolin, L. J. Greenstein, "Performance Analysis of Digital Radio Links with Nonlinear Transmit Amplifiers", *IEEE Journ. on Selected Areas in Communications*, Vol. SAC-5, No 3, pp 534-456, April 1987.

[17] A.A.M. Saleh, "Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers", *IEEE Trans. on Communications*., vol. COM-29, pp. 1715-1720, Nov. 1981.

[18] L. Vecci, F. Piazza and A. Uncini, "Learning and Approximation Capabilities of Adaptive Spline Activation Function Neural Networks", accepted for publication on *Neural Networks.*

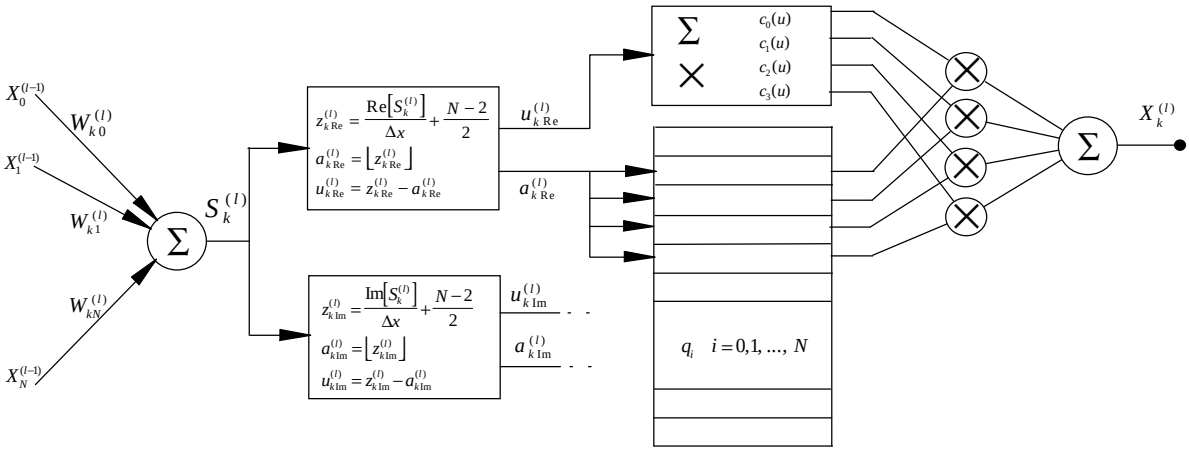

Figure 2. The complex-valued generalized sigmoid (GS) neuron structure (the structure of the imaginary part is omitted because it is identical to that of the real part).