ОБ ОДНОМ МЕТОДЕ РЕШЕНИЯ ЗАДАЧИ АППРОКСИМАЦИИ ФУНКЦИЙ С ПОМОЩЬЮ ДВУХСЛОЙНОЙ НЕЙРОННОЙ СЕТИ ПЕРЕМЕННОЙ СТРУКТУРЫ

Томашевич Н.С.

Научный центр нейрокомпьютеров
107066, Москва, ул. Новая Басманная, д. 20

Предложена методика создания двухслойной нейронной сети, выполняющей кусочно-нелинейную аппроксимацию заданной функции. Методика включает в себя определение числа нейронов сети, требуемых для аппроксимации функции с ошибкой не превышающей заданную, а также определение значений всех весов сети.

Введение

На сегодняшний день написано огромное количество работ, посвященных анализу аппроксимирующих свойств нейронных сетей [1-4]. Все авторы этих работ приходят к выводу о способности двухслойной нейронной сети аппроксимировать любую однозначно заданную функцию, однако, ни в одной из этих работ не указано, как построить такую сеть. Значительных успехов в этой области, на мой взгляд, достигли Nguyen и Widrow в своей работе [5], предложив методику выбора начальных значений весов двухслойной сети, хотя вопрос о числе нейронов скрытого слоя сети они оставили открытым.

В данной работе предложена методика синтеза двухслойной нейронной сети переменной структуры для аппроксимации одномерной функции $y^*(x)$, где x - вход сети, y^* - желаемый выход, по известным P примерам обучающей выборки. Структура сети называется «переменной», потому что число нейронов скрытого слоя сети заранее не известно, а определяется в процессе создания сети, и напрямую зависит от сложности заданной функции и от заданной пользователем максимальной ошибки аппроксимации $\Delta\varepsilon$. Выход нейрона вычисляется по формуле:

$$y_{h_k(p)} = f\left(\sum_{h_{(k-1)}=0}^{H_{(k-1)}} w_{h_k,h_{(k-1)}} \cdot y_{h_{(k-1)}(p)}\right), \qquad (1)$$

где h_k - индекс нейрона k-го слоя сети; H_k - число нейронов k-го слоя сети; p – номер примера, $p \in \{1,\ldots,P\}$; $y_{h_k(p)}$, $y_{h_{(k-1)}(p)}$ - выходы h_k-го и $h_{(k-1)}$-го нейронов, соответственно, k-го и $(k-1)$-го слоев при подаче на вход НС p-го примера; $w_{h_k,h_{(k-1)}}$ - вес к h_k-му нейрону k-го слоя от $h_{(k-1)}$-го нейрона предыдущего $(k-1)$-го слоя; $f(g) = \frac{2}{\pi}\arctan(g)$ - функция активации типа арктангенса.

Методика создания двухслойнной нейронной сети

Каждый пример обучающей выборки представляет собой пару $[x, y^*]$. Добавим к этому описанию примера еще одну характеристику: желаемое значение активации нейрона выходного слоя сети. Для p-го примера, $p \in \{1,\ldots,P\}$, оно вычисляется по формуле:

$$g^*_{(p)} = tg\left(\frac{\pi}{2} y^*_{(p)}\right). \qquad (2)$$

Создадим двухслойную нейронную сеть следующей структуры: один вход, один выход, число нейронов первого слоя равно нулю ($H_1 = 0$), число нейронов второго слоя равно единице ($H_2 = 1$). В сети есть один единственный вес – это смещение H_2-го нейрона, равное нулю ($w_{H_2,0} = 0$).

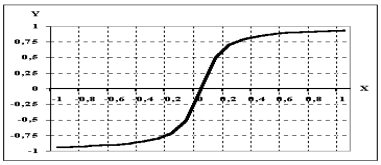

Рис. 1. Кривая типа арктангенса, реализуемая нейроном сети.

Далее нейроны добавляются только в первый слой сети. Каждый H_1-ый нейрон, добавляемый в первый слой сети представляет собой в пространстве $\{Y, X\}$ кривую типа арктангенса (рис. 1).

Будем говорить что пример обучающей выборки находится в области «нелинейности» нейрона, если значение его желаемого выхода по модулю меньше или равно некоторого заданного пользователем числа $\delta \to 1$, и в области «насыщения» если значение его желаемого выхода по модулю больше δ.

Каждый добавляемый в первый слой H_1-ый нейрон должен аппроксимировать с заданной ошибкой не превышающей $\Delta\varepsilon$ некоторое количество последовательно расположенных на оси *x* примеров. Под выражением ***« H_1 -ый нейрон аппроксимирует некоторое количество примеров»*** понимается следующее:

- ошибка аппроксимации ε_{H_1} для аппроксимируемых H_1-ым нейроном примеров не превышает $\Delta\varepsilon$, то есть $\varepsilon_{H_1} \le \Delta\varepsilon$; при этом аппроксимируемые нейроном примеры могут лежать как в области нелинейности, так и в области насыщения нейрона;
- в области нелинейности H_1-го нейрона отсутствуют примеры, ошибка аппроксимации которых данным нейроном превышает $\Delta\varepsilon$;
- примеры, ошибка аппроксимации которых данным нейроном превышает $\Delta\varepsilon$, находятся в области насыщения нейрона.

Далее на этапе добавления нейронов в скрытый слой сети нам потребуется изменять кривую типа арктангенса (рис. 1), сжимая/растягивая ее и сдвигая вверх/вниз. Для этого введем понятие полного выхода H_1-го нейрона сети:

$$\bar{y}_{H_1(p)} = y_{H_1(p)} \times w_{H_1,1(out)} + w_{H_1,0(out)}, \qquad (3)$$

где $w_{H_1,1(out)}$ – выходной вес H_1-го нейрона, позволяющий изменять высоту кривой (рис. 1), сжимая/растягивая ее;

$w_{H_1,0(out)}$ – выходной вес H_1-го нейрона, позволяющий сдвигать кривую (рис. 1) вверх/вниз.

Понятия полного выходного сигнала нейрона $\bar{y}_{h_1}$, выходных весов $w_{h_1,1(out)}$ и $w_{h_1,0(out)}$ используются только при создании сети, при ее использовании такие понятия отсутствуют.

Ошибка аппроксимации *p*-го примера H_1-ым нейроном определяется по формуле:

$$\varepsilon_{H_1(p)} = \left| \frac{2}{\pi} \arctan\left(\bar{y}_{H_1(p)}\right) - y^*_{(p)} \right|. \qquad (4)$$

Итак, будем создавать двухслойную нейронную сеть, реализующую кусочно-нелинейную аппроксимацию функции $y^*(x)$ по ее *P* известным значениям. Каждый H_1-ый нейрон, добавляемый в первый слой сети, должен аппроксимировать последний пример (обозначим его номер через P^{from}), аппроксимированный предыдущим нейроном если таковой имеется (если не имеется, то $P^{from} = 1$), и максимальное количество последующих примеров до примера P^{to} с ошибкой не превышающей $\Delta\varepsilon$.

При добавлении H_1-го нейрона сперва вычисляем значения его выходных весов: $w_{H_1,0(out)}$ и $w_{H_1,1(out)}$, по формулам:

$$y^{\max}_{H_1} = \max\left\{g^*_{(P^{from})}, g^*_{(P^{to})}\right\};\ y^{\min}_{H_1} = \min\left\{g^*_{(P^{from})}, g^*_{(P^{to})}\right\};$$

$$w_{H_1,0(out)} = \frac{y^{\max}_{H_1} - y^{\min}_{H_1}}{2} + y^{\min}_{H_1};\ w_{H_1,1(out)} = \frac{y^{\max}_{H_1} - y^{\min}_{H_1}}{2 \cdot \delta}. \qquad (5)$$

Теперь мы можем определить значение желаемого выхода H_1-го нейрона:

$$y^*_{H_1(p)} = \frac{g^*_{(p)} - w_{H_1,0(out)}}{w_{H_1,1(out)}}, \qquad (6)$$

и, решив следующую систему уравнений:

$$\begin{cases} \frac{2}{\pi} arctg\left(w_{H_1,0} + w_{H_1,1} \cdot x_{(P^{from})}\right) = y^*_{H_1(P^{from})} \\ \frac{2}{\pi} arctg\left(w_{H_1,0} + w_{H_1,1} \cdot x_{(P^{to})}\right) = y^*_{H_1(P^{to})} \end{cases},$$

получим значения входных весов H_1-го нейрона:

$$w_{H_1,1} = \frac{tg\left(\frac{\pi}{2} y^*_{H_1(P^{from})}\right) - tg\left(\frac{\pi}{2} y^*_{H_1(P^{to})}\right)}{x_{(P^{from})} - x_{(P^{to})}};\ w_{H_1,0} = tg\left(\frac{\pi}{2} y^*_{H_1(P^{to})}\right) - w_{H_1,1} \cdot x_{(P^{to})}. \qquad (7)$$

При этом вес w_{H_2,H_1} от добавленного H_1-го нейрона к нейрону выходного слоя устанавливается равным:

$$w_{H_2,H_1} = w_{H_1,1(out)}, \quad (8)$$

а смещение $w_{H_2,0}$ нейрона выходного слоя обновляется по формуле:

$$w_{H_2,0} = w_{H_2,0} + w_{H_1,0(out)} - g^*_{(P^{from})}. \quad (9)$$

Таким образом продолжаем добавлять нейроны в первый слой сети до тех пор, пока значение P^{to} очередного добавленного в первый слой нейрона не станет равно числу примеров обучающей выборки *P*.

Эксперименты и выводы

Были проведены эксперименты на двух видах функций: на низкочастотной функции (рис. 2), и на высокочастотной функции (рис. 3).

Обучающая выборка низкочастотной функции $y^*(x) = \sin(\pi x^2)\sin(2\pi x)$ состояла из 21 примера, взятых на интервале $x \in [-1;+1]$ с интервалом дискретизации равным 0.1 (эти примеры обозначены черными кружками на рис. 2). Тестовая выборка состояла из 201 примера, взятых на интервале $x \in [-1;+1]$ с интервалом дискретизации равным 0.01 (на рис. 2 это сплошная черная линия).

Обучающая выборка высокочастотной функции $y^*(x) = \sin(10\pi x^2)\sin(10\pi x)$ состояла из 201 примера, взятых на интервале $x \in [-1;+1]$ с интервалом дискретизации равным 0.01 (эти примеры обозначены черными кружками на рис. 3). Тестовая выборка состояла из 2001 примера, взятых на интервале $x \in [-1;+1]$ с интервалом дискретизации равным 0.001 (на рис. 3 это сплошная черная линия).

И обучающая и тестовая выборки для обоих видов функций были взяты с точностью до десятого знака после запятой

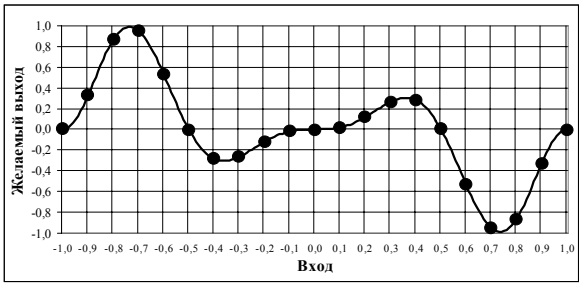

Рис. 2. Низкочастотная функция $y^*(x)=\sin(\pi x^2)\sin(\pi x)$.

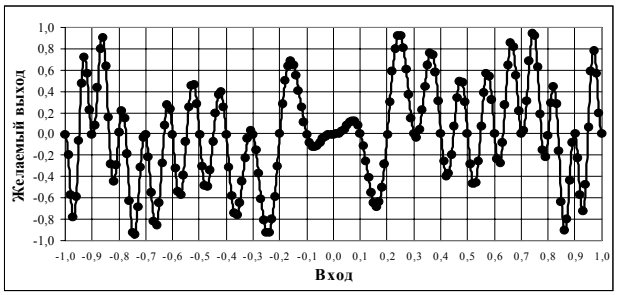

Рис. 3. Высокочастотная функция $y^*(x)=\sin(10\pi x^2)\sin(10\pi x)$.

Так как используемая функция активации (рис. 1) в области насыщения, определяемой параметром $\delta \to 1$, все-таки остается нелинейной, то достижение заданной ошибки аппроксимации $\Delta\varepsilon \to 0$ при создании сети как правило невозможно. Поэтому после создания нейронной сети, она обучалась градиентным методом. При этом использовалась методика настройки шага обучения по слоям (у каждого слоя свой шаг) [6].

Целью экспериментов было исследование следующих зависимостей:

- средней и максимальной ошибок аппроксимации (на обучающей и тестовой выборках) от величины δ на этапах создания и обучения НС;
- средней и максимальной ошибок аппроксимации (на обучающей и тестовой выборках) от величины $\Delta\varepsilon$ на этапах создания и обучения НС;
- числа нейронов получившихся в первом слое НС от величин $\Delta\varepsilon$ и δ;

а также установление предельно возможной минимальной ошибки аппроксимации функции нейронной сетью.

На основании проведенных экспериментов были сделаны следующие выводы:

- Средняя и максимальная ошибки аппроксимации (на обучающей и тестовой выборках) на этапе создания НС убывают при $\delta \to 1$ и $\Delta\varepsilon \to 0$.
- Средняя и максимальная ошибки аппроксимации (на обучающей и тестовой выборках) после обучения НС ведут себя по разному для высокочастотной и низкочастотной функций. Возможно, это связано с использованным методом обучения нейронных сетей, не приведшим к глобальным экстремумам (при обучении НС не использовалось никакой методики выхода из локальных экстремумов). Хотя, можно сделать следующие обобщения: средняя и максимальная ошибки аппроксимации на обучающей выборке

после обучения НС имели минимальные значения при $\delta \to 1$ и $\Delta\varepsilon \to 0$; а средняя и максимальная ошибки аппроксимации на тестовой выборке после обучения НС имели минимальные значения при $\Delta\varepsilon = 0.01 \div 0.1$ ($\Delta\varepsilon = 0.5 \div 5\%$ от диапазона изменения функции) и $\delta = 0.65 \div 0.75$.

- Число нейронов первого слоя сети, как и следовало ожидать, зависит только от $\Delta\varepsilon$ (чем меньше $\Delta\varepsilon$, тем больше нейронов в первом слое), и вообще не зависит от δ. Зависимость числа нейронов от $\Delta\varepsilon$ нелинейная и очень похожа по форме на кривую типа арктангенса (рис. 1). Максимальное число нейронов равно числу примеров минус два.
- Предельно возможные минимальные ошибки аппроксимации для высокочастотной и низкочастотной функций изменяются в следующий пределах (ошибка приведена в процентах от диапазона изменения функции):

Тип ошибки	**Ошибка, %**
После создания НС:	
средняя ошибка аппроксимации на обучающей выборке	$0.001 \div 0.002$
максимальная ошибка аппроксимации на обучающей выборке	$0.004 \div 0.006$
средняя ошибка аппроксимации на тестовой выборке	$1.7 \div 2.4$
максимальная ошибка аппроксимации на тестовой выборке	$8.5 \div 11.2$
После обучения НС:	
средняя ошибка аппроксимации на обучающей выборке	$0.00003 \div 0.00020$
максимальная ошибка аппроксимации на обучающей выборке	$0.0006 \div 0.0010$
средняя ошибка аппроксимации на тестовой выборке	$0.7 \div 0.9$
максимальная ошибка аппроксимации на тестовой выборке	$3.5 \div 4.8$

При этом соответствующие значения ошибок для низкочастотной функции всегда меньше чем для высокочастотной.

Литература

1. Hornik K., Stinchcombe M., White H. Multilayer feedforward networks are universal approximators. Neural Networks, 1989, vol. 2, pp. 359-366.
2. Stinchcombe M., White H. Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions. International Joint Conference on Neural Networks, San Diego: SOS Printing, 1989, pp. I: 613-618.
3. Cybenko G. Approximation by superposition of a sigmoidal functions. Mathematics of Control, Signals and Systems, 1989, vol. 2, pp. 303-314.
4. Y. Ito. Approximation of continuous function on R by linear combination of shifted rotations of a sigmoidal function with and without scaling. Neural Networks, 1992, vol. 5, pp. 105-115.
5. Derrick Nguyen and Bernard Widrow. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. IEEE Int. Conf. on Neural Networks, pp. III: 21-26.
6. Томашевич Н.С., Томашевич Д.С. О выборе шага обучения нейронной сети с полными последовательными связями. Сб. Докл. VI Всероссийской коференции "Нейрокомпьютеры и их применение", Москва, 16-18 февраля, 2000.

ABOUT ONE METHOD OF FUNCTIONS APPROXIMATION PROBLEM DECISION USING TWO-LAYER NEURAL NETWORK OF VARIABLE FRAME

Tomashevich N.S.

Scientific Center of Neurocomputers
107066, Moscow, st. Novaya Basmannaya, 20

The method of creation of two-layer neural network, which realize piecewise-nonlinear approximation of a given function, is proposed. The method involves determination of network's neurons number, which are necessary for function approximation with error not exceeding the given one, and determination of weights values for all neurons of network.

Introduction

For today it is written a lot of activities dedicated to analysis of approximation properties of neural networks [1-4]. All authors of these activities come to conclusion about capacity of two-layer neural network to approximate anyone uniquely defined function, but they don't explain how to construct such network. Considerable successes in this area, in my opinion, have been reached by Nguyen and Widrow in their activity [5], in which they propose the method of selection of initial weights values for two-layer neural network, though a question about neurons number in the hidden layer they left opened.

In this paper the method of synthesis of two-layer neural network of variable frame for approximation of one-dimensional function $y^*(x)$, where x - input of network, y^* - desired output, by *P* known examples of learning sample is proposed. The network frame is named «variable», because number of neurons of hidden layer of network is unknown beforehand, but it is determined during network creation and it directly depends on complication of a given function and from the given by user value of maximum approximation error $\Delta\varepsilon$. The output of neuron is computed by formula:

$$y_{h_k(p)} = f\left(\sum_{h_{(k-1)}=0}^{H_{(k-1)}} w_{h_k,h_{(k-1)}} \cdot y_{h_{(k-1)}(p)}\right), \qquad (1)$$

where h_k - index of neuron of *k*-th layer of network; H_k - number of neurons in *k*-th layer of network; *p* – number of example, $p \in \{1,\ldots,P\}$; $y_{h_k(p)}$, $y_{h_{(k-1)}(p)}$ - outputs of h_k -th и $h_{(k-1)}$ -th neuron of *k*-th and $(k-1)$ -th layers, correspondingly, at feeding on NN input of *p*-th example; $w_{h_k,h_{(k-1)}}$ - weight to h_k -th neuron of *k*-th layer from $h_{(k-1)}$ -th neuron of preceding $(k-1)$ -th layer; $f(g) = \frac{2}{\pi}\arctan(g)$ - activation function of arctangent type.

Method of two-layer neural network creation

Each example of learning sample represents a couple $[x, y^*]$. Let's add to this example description one more feature: desired activation of neuron of output network layer. For *p*-th example, $p \in \{1,\ldots,P\}$, it is computed by formula:

$$g^*_{(p)} = tg\left(\frac{\pi}{2} y^*_{(p)}\right). \qquad (2)$$

Let's create two-layer neural network of the following frame: one input, one output, number of neurons of first layer is equal to zero ($H_1 = 0$), number of neurons of second layer is equal to one ($H_2 = 1$). There is a single weight in network – the bias of H_2 -th neuron, which is equal to zero ($w_{H_2,0} = 0$).

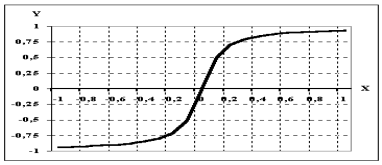

Fig. 1. Curve of arctangent type, realized by one neuron of network.

Later neurons are added only in first layer of network. Each H_1 -th neuron, which is added in first network layer, represents in space $\{Y, X\}$ the curve of arctangent type (fig. 1).

Let's talk that an example of learning sample is lying in region of «nonlinearity» of neuron, if its desired output is less or equal in absolute value to some given by user number $\delta \to 1$, and it is lying in region of neuron's «saturation» if its desired output is greater in absolute value than δ.

Each added to first layer the H_1-th neuron must approximate with error not exceeding the given value $\Delta\varepsilon$ some quantity of consecutively located on *x*-axis examples. The expression ***« H_1-th neuron approximates some quantity of examples»*** is understood as following:

- approximation error ε_{H_1} for approximated by H_1-th neuron examples doesn't exceed $\Delta\varepsilon$, that is $\varepsilon_{H_1} \le \Delta\varepsilon$; at that, approximated by neuron examples may lay and in region of nonlinearity, and in region of saturation of this neuron;
- in region of nonlinearity of H_1-th neuron there is no examples, which approximation error by this neuron exceeds $\Delta\varepsilon$;
- examples, which approximation error by this neuron exceeds $\Delta\varepsilon$, lay in region of saturation of neuron.

Then on the stage of neurons adding in the hidden layer of network we will need to change shape of curve of arctangent type (fig. 1) by its compression/distension and its shift up/down. For this let's introduce the notion of complete output of H_1-th neuron of network:

$$\bar{y}_{H_1(p)} = y_{H_1(p)} \times w_{H_1,1(out)} + w_{H_1,0(out)}, \qquad (3)$$

where $w_{H_1,1(out)}$ – output weight of H_1-th neuron, allowing to change the height of curve (fig. 1) by its compression/distension;

$w_{H_1,0(out)}$ – output weight of H_1-th neuron, allowing to shift curve (fig. 1) up/down.

Notions of complete output of neuron $\bar{y}_{h_1}$, of output weights $w_{h_1,1(out)}$ and $w_{h_1,0(out)}$ are used only at network creation, at its utilization these notions are absent.

Error of approximation of *p*-th example by H_1-th neuron is defined by formula:

$$\varepsilon_{H_1(p)} = \left| \frac{2}{\pi} \arctan\left(\bar{y}_{H_1(p)}\right) - y^*_{(p)} \right|. \qquad (4)$$

Now, let's construct two-layer neural network realizing piecewise-nonlinear approximation of function $y^*(x)$ by its *P* known values. Each H_1-th neuron, which is added to the first network layer, must approximate last example (let's note its number by P^{from}) approximated by preceding neuron if such exists (if preceding neuron doesn't exist, then $P^{from} = 1$) and maximum quantity of subsequent examples up to example P^{to} with error not exceeding $\Delta\varepsilon$.

At adding of H_1-th neuron, let's firstly compute values of its output weights: $w_{H_1,0(out)}$ and $w_{H_1,1(out)}$, by formulas:

$$y^{\max}_{H_1} = \max\left\{g^*_{(P^{from})}, g^*_{(P^{to})}\right\};\ y^{\min}_{H_1} = \min\left\{g^*_{(P^{from})}, g^*_{(P^{to})}\right\};$$

$$w_{H_1,0(out)} = \frac{y^{\max}_{H_1} - y^{\min}_{H_1}}{2} + y^{\min}_{H_1};\ w_{H_1,1(out)} = \frac{y^{\max}_{H_1} - y^{\min}_{H_1}}{2 \cdot \delta}. \qquad (5)$$

Now we can determine the value of desired output for H_1-th neuron:

$$y^*_{H_1(p)} = \frac{g^*_{(p)} - w_{H_1,0(out)}}{w_{H_1,1(out)}}, \qquad (6)$$

and, by solving the following system of equations:

$$\begin{cases} \frac{2}{\pi} arctg\left(w_{H_1,0} + w_{H_1,1} \cdot x_{(P^{from})}\right) = y^*_{H_1(P^{from})} \\ \frac{2}{\pi} arctg\left(w_{H_1,0} + w_{H_1,1} \cdot x_{(P^{to})}\right) = y^*_{H_1(P^{to})} \end{cases},$$

we will get values of input weights of H_1-th neuron:

$$w_{H_1,1} = \frac{tg\left(\frac{\pi}{2} y^*_{H_1(P^{from})}\right) - tg\left(\frac{\pi}{2} y^*_{H_1(P^{to})}\right)}{x_{(P^{from})} - x_{(P^{to})}};\ w_{H_1,0} = tg\left(\frac{\pi}{2} y^*_{H_1(P^{to})}\right) - w_{H_1,1} \cdot x_{(P^{to})}. \quad (7)$$

At that the weight w_{H_2,H_1} from added H_1-th neuron to the single neuron of output layer is established equal to:

$$w_{H_2,H_1} = w_{H_1,1(out)}, \quad (8)$$

and bias $w_{H_2,0}$ of neuron of output layer is updated by formula:

$$w_{H_2,0} = w_{H_2,0} + w_{H_1,0(out)} - g^*_{(P^{from})}. \quad (9)$$

In that way we continue to add neurons in first layer up to that moment when value P^{to} of the last added in first layer neuron will be equal to number of examples in learning sample *P*.

Experiments and conclusions

Experiments were carried out on two kinds of functions: on low-frequency function (fig. 2) and on high-frequency function (fig. 3).

Learning sample of low-frequency function $y^*(x) = \sin(\pi x^2)\sin(2\pi x)$ consisted of 21 examples, taken from interval $x \in [-1;+1]$ with quantization interval equal to 0.1 (these examples are marked by black circles in fig. 2). Test sample consisted of 201 examples, taken from interval $x \in [-1;+1]$ with quantization interval equal to 0.01 (it is a solid black line in fig. 2).

Learning sample of high-frequency function $y^*(x) = \sin(10\pi x^2)\sin(10\pi x)$ consisted of 201 examples, taken from interval $x \in [-1;+1]$ with quantization interval equal to 0.01 (these examples are marked by black circles in fig. 3). Test sample consisted of 2001 examples, taken from interval $x \in [-1;+1]$ with quantization interval equal to 0.001 (it is a solid black line in fig. 3).

Learning and test samples for both kinds of functions were taken with precision up to tenth digit after point.

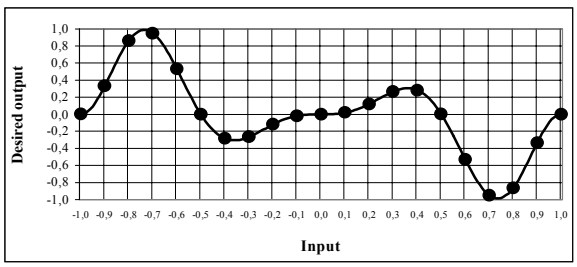

Fig. 2. Low-frequency function $y^*(x)=\sin(\pi x^2)\sin(\pi x)$.

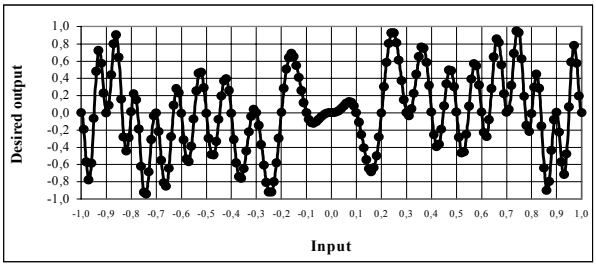

Fig. 3. High-frequency function $y^*(x)=\sin(10\pi x^2)\sin(10\pi x)$.

Since used activation function (fig. 1) remains nevertheless non-linear in region of saturation defined by parameter $\delta \to 1$, then achievement of the given approximation error $\Delta\varepsilon \to 0$ at network creation is usually impossible. Therefore after neural network creation it was taught by gradient method. At that the method of steps by layers adjusting (each layer has its own step) [6] was used.

The aim of experiments was an investigation of following dependencies:

- average and maximum approximation errors (on learning and test samples) from value of δ on stages of NN creation and NN learning;
- average and maximum approximation errors (on learning and test samples) from value of $\Delta\varepsilon$ on stages of NN creation and NN learning;
- number of neurons achieved in first layer of NN from values of $\Delta\varepsilon$ and δ;

and also determination of the possible limit of minimum function approximation error on neural network.

Based on carried out experiments the following conclusions was done:

- Average and maximum approximation errors (on learning and test samples) on stage of NN creation decrease at $\delta \to 1$ and $\Delta\varepsilon \to 0$.
- Average and maximum approximation errors (on learning and test samples) after NN learning have different behavior for low-frequency and high-frequency functions. Possibly it is because of used method of neural networks learning, which didn't lead to the global extremums (at NN learning there wasn't used any method of local extremums escaping). Though, it is possible to make following generalizations: average and maximum

approximation errors on learning sample after NN teaching have had minimal values at $\delta \to 1$ and $\Delta\varepsilon \to 0$; and average and maximum approximation errors on test sample after NN teaching have had minimal values at $\Delta\varepsilon = 0.01 \div 0.1$ ($\Delta\varepsilon = 0.5 \div 5\%$ from size of function changing) and $\delta = 0.65 \div 0.75$.

- Number of neurons of first network layer, as it was expected, depends only from $\Delta\varepsilon$ (as less $\Delta\varepsilon$, as bigger number of neurons in first layer) and doesn't depends from δ. Dependence of neurons number from $\Delta\varepsilon$ is nonlinear and is similar to curve of arctangent type (fig. 1). Maximum neurons number is equal to number of examples minus two.
- The possible limit of minimum function approximation errors for low-frequency and high-frequency functions changes in following bounds (error was taken in percents from size of function changing):

Error type	**Error, %**
After NN creation:	
average approximation error on learning sample	$0.001 \div 0.002$
maximum approximation error on learning sample	$0.004 \div 0.006$
average approximation error on test sample	$1.7 \div 2.4$
maximum approximation error on test sample	$8.5 \div 11.2$
After NN teaching:	
average approximation error on learning sample	$0.00003 \div 0.00020$
maximum approximation error on learning sample	$0.0006 \div 0.0010$
average approximation error on test sample	$0.7 \div 0.9$
maximum approximation error on test sample	$3.5 \div 4.8$

At that corresponding values of errors for low-frequency function are always less than for high-frequency.

References

1. Hornik K., Stinchcombe M., White H. Multilayer feedforward networks are universal approximators. Neural Networks, 1989, vol. 2, pp. 359-366.
2. Stinchcombe M., White H. Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions. International Joint Conference on Neural Networks, San Diego: SOS Printing, 1989, pp.613-618.
3. Cybenko G. Approximation by superposition of a sigmoidal functions. Mathematics of Control, Signals and Systems, 1989, vol. 2, pp. 303-314.
4. Y. Ito. Approximation of continuous function on R by linear combination of shifted rotations of a sigmoidal function with and without scaling. Neural Networks, 1992, vol. 5, pp. 105-115.
5. Derrick Nguyen and Bernard Widrow. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. IEEE Int. Conf. on Neural Networks, pp. III: 21-26.
6. Tomashevich N.S., Tomashevich D.S. About adjusting of learning rate for fully feedforward-connected neural network of fixed frame. Proc. of VI All-Russian Conf. "Neurocomputers and their applications", Moscow, 16-18 February, 2000.