# Adaptivna aktivacijska funkcija u skrivenom sloju neuronske mreže

**Kljajić, Zorica**




Sveučilište u Zagrebu

**Fakultet strojarstva i brodogradnje**

# Adaptivna aktivacijska funkcija u skrivenom sloju neuronske mreže

# ZAVRŠNI RAD

Voditelj rada:

Prof. dr. sc. Dubravko Majetić

Zorica Kljajić

Zagreb, 2014.

Sveučilište u Zagrebu

**Fakultet strojarstva i brodogradnje**

# Adaptivna aktivacijska funkcija u skrivenom sloju neuronske mreže

# ZAVRŠNI RAD

Zorica Kljajić

Zagreb, 2014.

*Izjavljujem da sam ovaj završni rad preddiplomskog studija radila samostalno na Fakultetu strojarstva i brodogradnje u Zagrebu znanjem stečenim tijekom dosadašnjeg studija. Zahvaljujem prof. dr. sc. Dubravku Majetiću na savjetima i pruženoj pomoći.*

*Zorica Kljajić*

# Sažetak

U radu je prikazan postupak učenja najpoznatijeg i najčešće upotrebljavanog tipa neuronskih mreža, diskretne statičke neuronske mreže s povratnim rasprostiranjem pogreške [1]. Za adaptivnu aktivacijsku funkciju u skrivenom sloju neuronske mreže je odabrana Gauss-ova radijalna bazna funkcija s dva promjenjiva parametra. To omogućuje nekorištenje Bias neurona u strukturi mreže koja je formirana s tri sloja. Neuroni izlaznog sloja su linearni što znači da je izlaze iz skrivenog sloja potrebno samo sumirati. Po uzorku se parametri mijenjaju za svaki ulazno izlazni par skupa učenja. Zbog velikog broja potrebnih iteracija kod algoritma povratnog prostiranja pogreške, uvodi se modifikacija momentumom prvog i drugog reda što omogućuje ubrzavanje procesa učenja. Programska podrška je načinjena u programskom paketu MATLAB čime je omogućena usporedba kvalitete učenja u slučajevima korištenja ili nekorištenja Bias neurona i momentuma, ili povećanja i smanjivanja koeficijenta učenja. U prvom poglavlju je opis umjetnih neuronskih mreža, neurona i procesa učenja, a u nastavku rada su prikazana detaljnija objašnjenja učenja konkretne mreže.

# Sadržaj

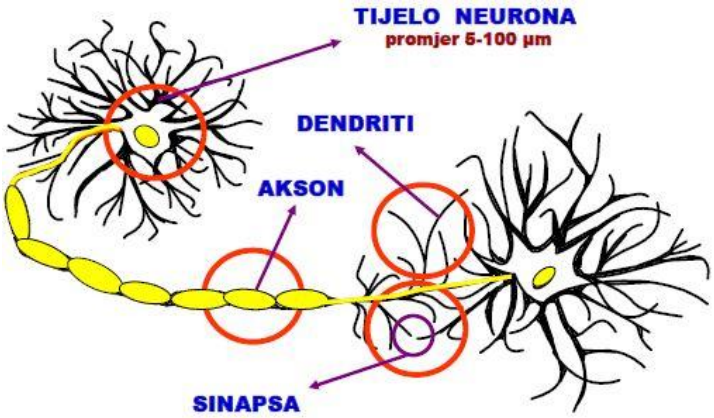
# Popis slika

# Popis tablica

# 1 Uvod

Umjetne neuronske mreže su element umjetne inteligencije. Originalna ideja za takvu neuronsku mrežu dolazi od pokušaja razumijevanja kako funkcionira mozak čovjeka. Pri tome su otkrivene velike sličnosti između umjetnog i biološkog neurona.

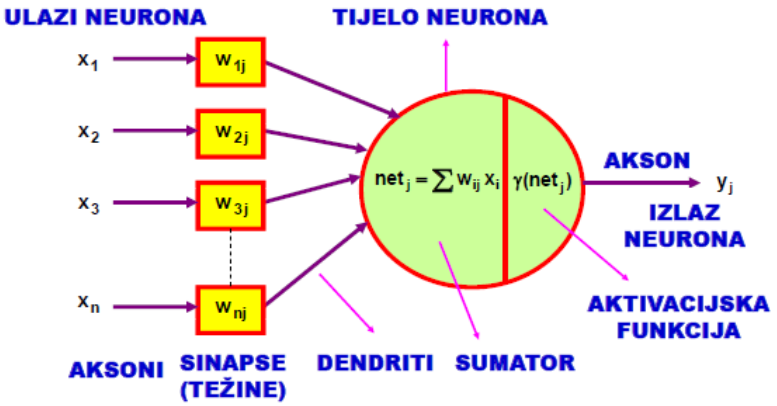
## 1.1 Biološki neuron




**Slika 1. Pojednostavljena slika biološkog neurona**

*Slika 1* prikazuje biološki neuron koji se sastoji iz tijela, aksona i mnoštva dendrita koji okružuju tijelo neurona. Akson je tanka cjevčica koje je jedan kraj povezan na tijelo neurona, a drugi se dijeli na niz grana. Razmak između završetka aksona prethodnog neurona i dendrita ili tijela sljedećeg neurona naziva se sinapsa. Izlazi neurona kroz akson putuju do sinapsi odakle se različito otežani signali šalju na tijelo drugih neurona. Neuron šalje impuls ako je njegova uzbuda veća od smirujućeg utjecaja na kritični iznos, koji predstavlja prag osjetljivosti neurona. Aktivnost neurona se može modelirati kao zbroj otežanih ulaza neurona. Ti ulazi su pomnoženi određenim faktorima koji se nazivaju težine neurona. Ako to usporedimo s umjetnim neuronom, tada on ovisi o broju ulaza (veza) iz okoline neurona, intenzitetu tih veza (iznosu težinskih faktora), te o pragu osjetljivosti. Kompleksne funkcije neuronske mreže se ostvaruju kompleksnošću veza (težinama) među neuronima, prije nego isključivo kompleksnošću svakog neurona ponaosob.

## 1.2 Umjetni neuron



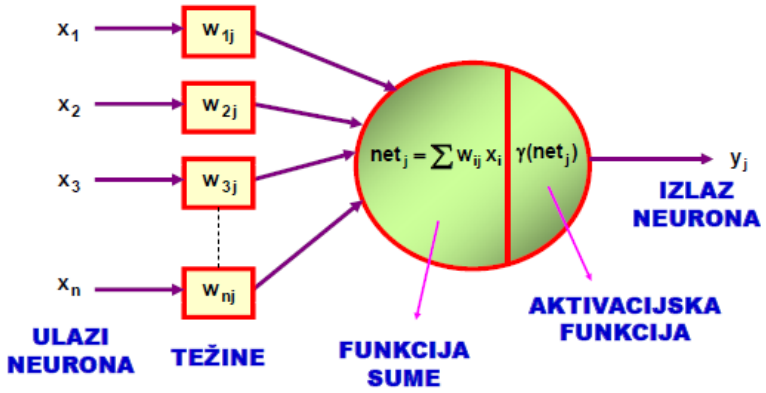

**Slika 2. Struktura umjetnog neurona**

Kod umjetnog neurona, *slika 2*, tijelo biološkog neurona se zamjenjuje sumatorom, ulogu dendrita preuzimaju ulazi u sumator, akson umjetnog neurona je izlaz sumatora, a uloga praga osjetljivosti biološkog neurona se preslikava na tzv. aktivacijske funkcije. Težinski faktori rade isto ono što i sinapse kod biološkog neurona, povezuju izlaze iz okoline neurona, tj. izlaze drugih neurona s ulazima sumatora, gdje je intenzitet veze određen iznosom. Dakle, sličnosti funkcioniranja umjetnog i biološkog neurona su u tome što se izlazi iz drugih umjetnih neurona množe s težinskim faktorima i dovode do sumatora. U sumatoru se tako dobiveni produkti sumiraju, a njihova suma se odvodi na ulaz aktivacijske funkcije, koja će na svom izlazu dati izlaz neurona.

## 1.3 Vrste i učenje umjetnih neuronskih mreža

Osnovna podjela umjetnih neuronskih mreža je na jednoslojne i višeslojne. Višeslojne mreže imaju ulazni, skriveni i izlazni sloj. Ako signali putuju samo u jednom smjeru, govorimo o unaprijednim, a ako postoji povratna petlja, o povratnim neuronskim mrežama. Učenje mreža se odvija uz ili bez nadzora, ili kombinacijom prethodnih. Kod učenja uz nadzor (supervizornog učenja), koje je ujedno i korišteno u ovom radu, skup izlaznih varijabli se uspoređuje sa željenim skupom izlaznih varijabli. Razlika željenih i ostvarenih izlaza gradi pogrešku mreže koja se koristi za računanje novih težina preko određenog algoritma, ovdje algoritma povratnim rasprostiranjem pogreške - Error Back Propagation (u daljnjem tekstu EBP). Cijeli postupak se ponavlja iteracijski dok pogreška ne bude manja od željene. Izlaz mreže je uvijek neka aproksimacija željenog izlaza i ne treba davati sto postotnu točnost. Drugim riječima, mreža je približavanje najboljem rezultatu, a ne kalkulator.

## 1.4 Osnove unaprijednih (statičkih) neuronskih mreža




**Slika 3. Model statičkog neurona**

*Slika 3* prikazuje osnovni model statičkog neurona. Vidljive su dvije temljne podfunkcije takvog neurona, funkcija sume $\sum$ i aktivacijska funckija $\gamma$. Funkcija sume predstavlja umnožak ulaza neurona i težinskih faktora, a rezultira vrijednošću *net*. Aktivacijska funkcija $\gamma$ preslikava vrijednosti *net* u izlaznu vrijednost neurona *y*. Neuron posjeduje više ulaza i samo jedan izlaz. Uobičajeno je da svaki neuron koji sudjeluje u procesu učenja posjeduje poseban ulaz jedinične vrijednosti koji je u strukturi neuronske mreže realiziran vezom sa zasebnim neuronom oznake Bias konstantnog izlaza jednakog jedinici (nema ulaz). Dodaje se ondje gdje se nalaze parametri učenja, tako da izlazni neuron nema Bias. Moguća su dva načina promjene parametara u EBP algoritmu. Jedan je učenje po skupu pri čemu se parametri mijenjaju jednom nakon prolaska čitavog ulaznog skupa za učenje kroz mrežu, a drugi, ovdje korišten, je učenje po uzorku, tzv. "Pattern" procedura, u kojoj se parametri učenja mijenjaju za svaki ulazno izlazni par skupa za učenje, pri čemu ulazno izlazni par znači jedan redak datoteke učenja.

EBP algoritam funkcionira na sljedeći način: Na početku se svi parametri (težine) postave kao mali slučajni brojevi, a onda se koristi supervizorno učenje. U unaprijednoj fazi se s vrijednostima ulaza mreže izračunavaju izlazi, a zatim se u povratnoj fazi na osnovu ostvarenog i željenog izlaza izračunava pogreška učenja i mijenjaju se vrijednosti težina izlaznog, pa skrivenog sloja.

# 2 Matematički izvod zadanog modela neuronske mreže

Na *slici 4* je prikazan poopćen model neuronske mreže zadan projektnim zadatkom i korišten u izvodu. U *tablici 1* se nalazi pojmovnik, a unutar *tablice* 2 su definirani pojmovi korišteni u izvodu. *Tablica 3* daje matematički izvod modela za primjenu neuronske mreže s Gauss-ovom radijalnom baznom funkcijom u neuronima skrivenog sloja i linearnom aktivacijskom funkcijom u neuronima izlaznog sloja. Nakon toga slijede izračuni gradijenata i konačni izrazi za promjenu svih parametara učenja u neuronskoj mreži.

**Tablica 1. Pojmovnik**

| Oznaka | Objašnjenje |
|---|---|
| $v_{kp}$ | težinski koeficijenti između ulaza $Z_k$ i neurona $p$ u prvom sloju (od ulaza) računajući s lijeve strane; $k$=1 predstavlja Bias |
| $y_{ij}$ | izlaz iz neurona u $i$-tom sloju neurona od ulaza, $j$-tog po redu s lijeve strane; $j$=1 predstavlja Bias |
| $net_{Hij}$ | ulaz u neuron u $i$-tom sloju neurona od ulaza, $j$-tog po redu računajući s lijeve strane |
| $O_f$ | $f$-ti izlaz po redu računajući s lijeve strane |
| $net_{Of}$ | ulaz u $f$-ti izlazni neuron računajući s lijeve strane |
| $u_{mn}$ | težinski koeficijent između izlaza $m$ iz prvog sloja neurona, i neurona $n$ u drugom sloju neurona; izlaz kod kojeg je $m$=1 izlaz iz Bias-a |
| $w_{os}$ | težinski koeficijent između $o$-tog izlaza u drugom sloju neurona u ulaz $s$-tog izlaznog neurona računajući s lijeve strane |
| M | broj ulaza + Bias |
| I | broj izlaza |
| $K_1$ | broj neurona u prvom sloju od ulaza u skrivenom sloju + Bias |
| $K_2$ | broj neurona u drugom sloju od ulaza u skrivenom sloju + Bias |
| N | broj elemenata u skupu za učenje |
| K | nagib linearne aktivacijske funkcije *(u jednadžbama korišteno K=1)* |

**Tablica 2. Definicija pojmova korištenih u izvodu**

| Formula | Objašnjenje |
|---|---|
| $\gamma_j(net_{Hij}) = e^{-\frac{1}{2}*\left(\frac{net_{Hij}-c_{ij}}{\sigma_{ij}}\right)^2}$ | aktivacijska funkcija neurona skrivenog sloja |
| $\gamma_k = O_f = K * net_{Of}, f = 1,2,\dots,I$ | aktivacijska funkcija neurona u izlaznom sloju |
| $net_{Of} = \sum_{i=1}^{K_2} w_{if} * y_{2i}\ , f = 1,2,\dots,I$ $y_{21} = 1$ | ulaz u neuron *f* izlaznog sloja |
| $net_{H2j} = \sum_{m=1}^{K_1} u_{mj} * y_{1m}\ , j = 1,2,\dots,(K_2-1)$ $y_{11} = 1$ | ulaz u neurone drugog sloja neurona u skrivenom sloju |
| $net_{H1o} = \sum_{l=1}^{M} v_{lo} * Z_l\ , o = 1,2,\dots,(K_1-1)$ $Z_1 = 1$ | ulaz u neurone prvog sloja neurona u skrivenom sloju |
| $E = \frac{1}{2} * \sum_{n=1}^{N} (d_n - O_n)^2$ | funkcija pogreške |
| $E = \frac{1}{2} * \sum_{n=1}^{I} (d_n - O_n)^2$ | funkcija pogreške u jednom setu ulazno izlaznih podataka |
| $A_n = (d_n - O_n)$ | pomoćna varijabla |
| $E = \frac{1}{2} * \sum_{n=1}^{I} (d_n - K * (\sum_{i=1}^{K_2} w_{in} * y_{2i})^2$ | funkcija pogreške za potrebe određivanja $\nabla E = \frac{\delta E(\vartheta)}{\delta w_{xy}}$ |
| $y_{2i} = e^{-\frac{1}{2}*\left(\frac{\sum_{m=1}^{K_1} u_{m(i-1)} * y_{1m} - c_{2(i-1)}}{\sigma_{2(i-1)}}\right)^2}$ , i = 2, 3, … , $K_2$ | funkcija $y_{2i}$ za potrebe određivanja $\nabla E = \frac{\delta E(\vartheta)}{\delta u_{xy}}$ |
| $C_{2i} = \frac{\sum_{m=1}^{K_1} u_{m(i-1)} * y_{1m} - c_{2(i-1)}}{\sigma_{2(i-1)}}$ , i = 2,3,….., $K_2$ | pomoćna varijabla |
| $y_{1m} = e^{-\frac{1}{2}*\left(\frac{\sum_{o=1}^{M} v_{o(m-1)} * Z_o - c_{1(m-1)}}{\sigma_{1(m-1)}}\right)^2}$ , m = 2,3,….., $K_1$ | funkcija $y_{1m}$ za potrebe određivanja $\nabla E = \frac{\delta E(\vartheta)}{\delta v_{xy}}$ |
| $C_{1m} = \frac{\sum_{o=1}^{M} v_{o(m-1)} * Z_o - c_{1(m-1)}}{\sigma_{1(m-1)}}$, i = 2,3,….., $K_1$ | pomoćna varijabla |

**Tablica 3. Izvod matematičkog modela za primjenu neuronske mreže**

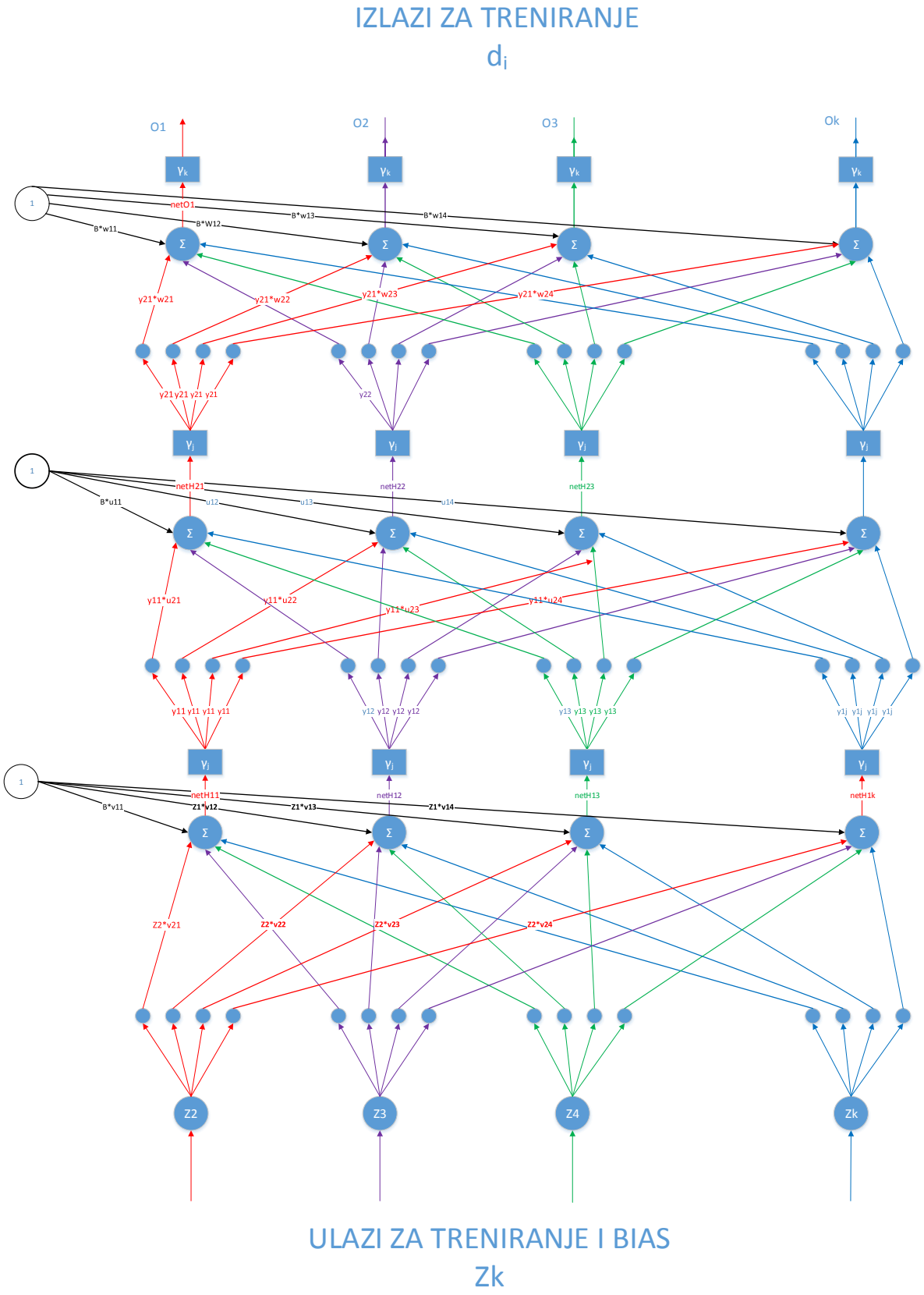
| Formula | Objašnjenje |
|---|---|
| $\nabla E(w_{xy}) = \frac{\partial E(\vartheta)}{\partial w_{xy}} = \frac{\partial E}{\partial A_y} * \frac{\partial A_y}{\partial O_y} * \frac{\partial O_y}{\partial net_{Oy}} * \frac{\partial net_{Oy}}{\partial w_{xy}}$ | gradijent pogreške $\nabla E(w_{xy})$ |
| $\nabla E(u_{xy}) = \frac{\partial E(\vartheta)}{\partial u_{xy}} = \sum_{n=1}^{I} (\frac{\partial E}{\partial A_n} * \frac{\partial A_n}{\partial O_n} * \frac{\partial O_n}{\partial net_{On}} * \frac{\partial net_{On}}{\partial y_{2y}} * \frac{\partial y_{2y}}{\partial C_{2y}} * \frac{\partial C_{2y}}{\partial u_{xy}})$ | gradijent pogreške $\nabla E(u_{xy})$ |
| $\nabla E(v_{xy}) = \frac{\partial E(\vartheta)}{\partial v_{xy}} = \sum_{n=1}^{I} (\frac{\partial E}{\partial A_n} * \frac{\partial A_n}{\partial O_n} * \frac{\partial O_n}{\partial net_{On}} * \sum_{i=2}^{K_2} (\frac{\partial net_{On}}{\partial y_{2i}} * \frac{\partial y_{2i}}{\partial C_{2i}} * \frac{\partial C_{2i}}{\partial y_{1y}} * \frac{\partial y_{1y}}{\partial C_{1y}} * \frac{\partial C_{1y}}{\partial v_{xy}}))$ | gradijent pogreške $\nabla E(v_{xy})$ |
| $\nabla E(\sigma_{2y}) = \frac{\partial E(\vartheta)}{\partial \sigma_{2y}} = \sum_{n=1}^{I} \frac{\partial E}{\partial A_n} * \frac{\partial A_n}{\partial O_n} * \frac{\partial O_n}{\partial net_{On}} * \frac{\partial net_{On}}{\partial y_{2y}} * \frac{\partial y_{2y}}{\partial C_{2y}} * \frac{\partial C_{2y}}{\partial \sigma_{2y}}$ | gradijent pogreške $\nabla E(\sigma_{2y})$ |
| $\nabla E(c_{2y}) = \frac{\partial E(\vartheta)}{\partial c_{2y}} = \sum_{n=1}^{I} \frac{\partial E}{\partial A_n} * \frac{\partial A_n}{\partial O_n} * \frac{\partial O_n}{\partial net_{On}} * \frac{\partial net_{On}}{\partial y_{2y}} * \frac{\partial y_{2y}}{\partial C_{2y}} * \frac{\partial C_{2y}}{\partial c_{2y}}$ | gradijent pogreške $\nabla E(c_{2y})$ |
| $\nabla E(\sigma_{1y}) = \frac{\partial E(\vartheta)}{\partial \sigma_{1y}} = \sum_{n=1}^{I} (\frac{\partial E}{\partial A_n} * \frac{\partial A_n}{\partial O_n} * \frac{\partial O_n}{\partial net_{On}} * \sum_{i=2}^{K_2} (\frac{\partial net_{On}}{\partial y_{2i}} * \frac{\partial y_{2i}}{\partial C_{2i}} * \frac{\partial C_{2i}}{\partial y_{1y}} * \frac{\partial y_{1y}}{\partial C_{1y}} * \frac{\partial C_{1y}}{\partial \sigma_{1y}}))$ | gradijent pogreške $\nabla E(\sigma_{1y})$ |
| $\nabla E(c_{1y}) = \frac{\partial E(\vartheta)}{\partial c_{1y}} = \sum_{n=1}^{I} (\frac{\partial E}{\partial A_n} * \frac{\partial A_n}{\partial O_n} * \frac{\partial O_n}{\partial net_{On}} * \sum_{i=2}^{K_2} (\frac{\partial net_{On}}{\partial y_{2i}} * \frac{\partial y_{2i}}{\partial C_{2i}} * \frac{\partial C_{2i}}{\partial y_{1y}} * \frac{\partial y_{1y}}{\partial C_{1y}} * \frac{\partial C_{1y}}{\partial c_{1y}}))$ | gradijent pogreške $\nabla E(c_{1y})$ |

**Tablica 4. Izračun gradijenata**

| Formula | Objašnjenje |
|---|---|
| $\nabla E(w_{xy}) = \frac{\partial E(\vartheta)}{\partial w_{xy}} = \frac{1}{2} * 2 * A_y * (-K * y_{2x})$ | izračun $\nabla E(w_{xy})$ |
| $\nabla E(u_{xy}) = \frac{\partial E(\vartheta)}{\partial u_{xy}} = \sum_{n=1}^{I} \left( \frac{1}{2} * 2 * A_n * \left( -K * w_{(y+1)n} * y_{2(y+1)} * \left( -\frac{1}{2} * 2 * C_{2(y+1)} * \frac{y_{1x}}{\sigma_{2y}} \right) \right) \right)$ | izračun $\nabla E(u_{xy})$ |
| $\nabla E(v_{xy}) = \frac{\partial E(\vartheta)}{\partial v_{xy}} = \sum_{n=1}^{I} \left( \frac{1}{2} * 2 * A_n * \left( -K * \left( \sum_{i=2}^{K_2} w_{in} * y_{2i} * \left( -\frac{1}{2} * 2 * C_{2i} * \frac{u_{(y+1)(i-1)}}{\sigma_{2(i-1)}} * y_{1(y+1)} * \left( -\frac{1}{2} * 2 * C_{1(y+1)} * \frac{z_x}{\sigma_{1y}} \right) \right) \right) \right) \right)$ | izračun $\nabla E(v_{xy})$ |
| $\nabla E(\sigma_{2y}) = \frac{\partial E(\vartheta)}{\partial \sigma_{2y}} = \sum_{n=1}^{I} \left( \frac{1}{2} * 2 * A_n * \left( -K * w_{(y+1)n} * y_{2(y+1)} * \left( -\frac{1}{2} * 2 * C_{2(y+1)} * \left( -\frac{net_{H2y} - C_{2y}}{\sigma_{2y}^2} \right) \right) \right) \right)$ | izračun $\nabla E(\sigma_{2y})$ |
| $\nabla E(c_{2y}) = \frac{\partial E(\vartheta)}{\partial c_{2y}} = \sum_{n=1}^{I} \left( \frac{1}{2} * 2 * A_n * \left( -K * w_{(y+1)n} * y_{2(y+1)} * \left( -\frac{1}{2} * 2 * C_{2(y+1)} * \left( -\frac{1}{\sigma_{2y}} \right) \right) \right) \right)$ | izračun $\nabla E(c_{2y})$ |
| $\nabla E(\sigma_{1y}) = \frac{\partial E(\vartheta)}{\partial \sigma_{1y}} = \sum_{n=1}^{I} \left( \frac{1}{2} * 2 * A_n * \left( -K * \sum_{i=2}^{K_2} w_{in} * y_{2i} * \left( -\frac{1}{2} * 2 * C_{2i} * \frac{u_{(y+1)(i-1)}}{\sigma_{2(i-1)}} * y_{1(y+1)} * \left( -\frac{1}{2} * 2 * C_{1(y+1)} * \left( -\frac{net_{H1y} - C_{1y}}{\sigma_{1y}^2} \right) \right) \right) \right) \right)$ | izračun $\nabla E(\sigma_{1y})$ |
| $\nabla E(c_{1y}) = \frac{\partial E(\vartheta)}{\partial c_{1y}} = \sum_{n=1}^{I} \left( \frac{1}{2} * 2 * A_n * \left( -K * \sum_{i=2}^{K_2} w_{in} * y_{2i} * \left( -\frac{1}{2} * 2 * C_{2i} * \frac{u_{(y+1)(i-1)}}{\sigma_{2(i-1)}} * y_{1(y+1)} * \left( -\frac{1}{2} * 2 * C_{1(y+1)} * \left( -\frac{1}{\sigma_{1y}} \right) \right) \right) \right) \right) \right)$ | izračun $\nabla E(c_{1y})$ |

**Tablica 5. Konačni izrazi za promjenu parametara neuronske mreže**

| Formula | Objašnjenje |
|---|---|
| $\Delta w_{xy}(b) = -\eta * \nabla E\left(w_{xy}(b)\right) + \alpha * \Delta w_{xy}(b-1) + \beta * \Delta w_{xy}(b-2)$ | promjena parametra $\Delta w_{xy}(b)$ |
| $\Delta u_{xy}(b) = -\eta * \nabla E\left(u_{xy}(b)\right) + \alpha * \Delta u_{xy}(b-1) + \beta * \Delta u_{xy}(b-2)$ | promjena parametra $\Delta u_{xy}(b)$ |
| $\Delta v_{xy}(b) = -\eta * \nabla E\left(v_{xy}(b)\right) + \alpha * \Delta v_{xy}(b-1) + \beta * \Delta v_{xy}(b-2)$ | promjena parametra $\Delta v_{xy}(b)$ |
| $\Delta \sigma_{2y}(b) = -\eta * \nabla E\left(\sigma_{2y}(b)\right) + \alpha * \Delta \sigma_{2y}(b-1) + \beta * \Delta \sigma_{2y}(b-2)$ | promjena parametra $\Delta \sigma_{2y}(b)$ |
| $\Delta c_{2y}(b) = -\eta * \nabla E\left(c_{2y}(b)\right) + \alpha * \Delta c_{cy}(b-1) + \beta * \Delta c_{2y}(b-2)$ | promjena parametra $\Delta c_{2y}(b)$ |
| $\Delta \sigma_{1y}(b) = -\eta * \nabla E\left(\sigma_{1y}(b)\right) + \alpha * \Delta \sigma_{1y}(b-1) + \beta * \Delta \sigma_{1y}(b-2)$ | promjena parametra $\Delta \sigma_{1y}(b)$ |
| $\Delta c_{1y}(b) = -\eta * \nabla E\left(c_{1y}(b)\right) + \alpha * \Delta c_{1y}(b-1) + \beta * \Delta c_{1y}(b-2)$ | promjena parametra $\Delta c_{1y}(b)$ |




**Slika 4. Arhitektura neuronske mreže**

Izrazi prikazani u tablicama prate prikazani model statičke unaprijedne troslojne neuronske mreže. Ulazi neurona $Z_k$ su ujedno i ulazi u mrežu uz dodatak Bias neurona jediničnog izlaza koji jedini nije povezan sa svakim neuronom prethodnog sloja. Ulazni sloj se ne računa kao pravi sloj jer u njemu nema procesiranja [2]. On samo prosljeđuje vektor ulaza u mrežu na ulaz prvog, skrivenog sloja [3]. Skriveni slojevi nisu u direktnoj interakciji s okolinom, a prvi je povezan s ulazima vezama opterećenim težinama $v_{kp}$. Težinski koeficijenti $u_{mn}$ povezuju prvi i drugi sloj. Treći, izlazni sloj neurona definira izlaze neuronske mreže označene s $O_f$, a povezan je s drugim skrivenim slojem neurona težinskim koeficijentima označenim s $w_{os}$. Kao što je prije navedeno, funkcija sume rezultira vrijednošću *net* koja predstavlja ulaze u neurone, zavisno o kojem sloju se radi, a γ preslikava *net* u izlaznu vrijednost *y*. Dakle, za dobivanje izlaza mreže u unaprijednoj fazi, proračun najprije započinje računanjem vrijednosti $net_{Hij}$ u skrivenom sloju koja je potrebna u Gauss-ovoj funkciji za dobivanje $y_{ij}$, a nakon toga se traži vrijednost $net_{Of}$ izlaznog sloja koja je potrebna za dobivanje $O_f$. Vrijednosti $net_{Of}$ i $O_f$ su u ovom slučaju jednakog iznosa zbog zadane linearne aktivacijske funkcije u izlaznom sloju $O_f = K * net_{Of}$, pri čemu je $K$=1 nagib linearne aktivacijske funkcije. Za lakše razumijevanje će biti prikazano raspisivanje sume vrijednosti *net*.

$$net_{Of} = \sum_{i=1}^{K_2} w_{if} * y_{2i}\ , f = 1, 2, \dots, I$$

$$net_{O1} = w_{11} * y_{21} + w_{21} * y_{22} + \cdots + w_{K_2 1} * y_{2K_2}$$

...

$$net_{OI} = w_{1I} * y_{21} + w_{2I} * y_{22} + \cdots + w_{K_2 I} * y_{2K_2}$$

$$net_{H2j} = \sum_{m=1}^{K_1} u_{mj} * y_{1m}\ , j = 1, 2, \dots, (K_2 - 1)$$

$$net_{H21} = u_{11} * y_{11} + u_{21} * y_{12} + \cdots + u_{K_1 1} * y_{1K_1}$$

...

$$net_{H2(K_2-1)} = u_{1(K_2-1)} * y_{11} + u_{2(K_2-1)} * y_{12} + \cdots + u_{K_1(K_2-1)} * y_{1K_1}$$

$$net_{H1o} = \sum_{l=1}^{M} v_{lo} * Z_l\ , o = 1, 2, \dots, (K_1 - 1)$$

...

$$net_{H1(K_1-1)} = v_{1(K_1-1)} * Z_1 + v_{2(K_1-1)} * Z_2 + \cdots + v_{M(K_1-1)} * Z_M$$

U povratnoj fazi učenja se na osnovu pogreške korigiraju vrijednosti parametara među slojevima pomoću sljedeće funkcije:

$$E = \frac{1}{2}\sum_{n=1}^{N}(d_n - O_n)^2 ,$$

gdje je *N* broj elemenata u skupu za učenje, $d_n$ su željeni, $O_n$ ostvareni izlazi, a $\frac{1}{2}$ pojednostavljuje potrebne derivacije funkcije pogreške.

## 2.1 Promjena parametara učenja

Za promjenu parametara Gauss-ove aktivacijske funkcije $\sigma$ i $c$ se izvodi matematika o istom principu kao i za težine. Postupak započinje sljedećim izrazom:

$$\vartheta(n+1) = \vartheta(n) + \Delta\vartheta(n),$$

$$\Delta\boldsymbol{\vartheta}(\boldsymbol{n}) = \boldsymbol{\vartheta}(\boldsymbol{n}+\mathbf{1}) - \boldsymbol{\vartheta}(\boldsymbol{n})$$

gdje je $\Delta\vartheta(n)$ veličina promjene parametara učenja, $\vartheta(n+1)$ je nova vrijednost parametra učenja, a $n$ trenutni korak učenja. Pogrešku $E(\vartheta)$ moguće je u okolišu radne točke $\vartheta$ aproksimirati s prva dva člana Taylorovog reda:

$$E(\vartheta + \Delta\vartheta) \approx E(\vartheta) + \Delta E(\vartheta),$$

$$\Delta E(\vartheta) = \Delta\vartheta^T \nabla E(\vartheta),$$

$$\nabla E(\vartheta) = \frac{\partial E(\vartheta)}{\partial \vartheta}.$$

Posljednji izraz predstavlja gradijent pogreške što je drugim riječima parcijalna derivacija funkcije pogreške.

$\Delta\vartheta$ za koji promjena pogreške učenja $\Delta E(\vartheta)$ poprima najveći negativni iznos se ostvaruje uz sljedeći uvjet:

$$\Delta\vartheta = -\eta\nabla E(\vartheta)$$

gdje je η mjera te promjene, koeficijent brzine učenja. Određuje ga učitelj, a njegova vrijednost se najčešće kreće između $10^{-3}$ i 10, danas čak i do $10^3$. Daljnjim izvodom se dobiva izraz za EBP algoritam:

$$\vartheta(n+1) = \vartheta(n) - \eta\nabla E\big(\vartheta(n)\big).$$

Budući da je njegov nedostatak velik broj potrebnih iteracija, EBP algoritam se modificira momentumom prvog, a za još veće ubrzavanje, drugog reda.

$$\Delta\vartheta(n) = -\eta\nabla E\big(\vartheta(n)\big) + \alpha\Delta\vartheta(n-1) + \beta\Delta\vartheta(n-2)$$

gdje *n* označava trenutnu promjenu parametra učenja, *(n-1)* prošlu, a *(n-2)* pretprošlu promjenu parametra učenja. Vrijednost koeficijenta $\alpha$ također određuje učitelj, a obično se bira u intervalu između 0.1 i 0.9. Na temelju koeficijenta $\alpha$, za koeficijent $\beta$ se koristi izraz:

$$\beta = \frac{1-\alpha}{3}$$

Promjena parametra učenja primjenom momentuma poprima konačni oblik:

$$\vartheta(n+1) = \vartheta(n) - \eta\nabla E\big(\vartheta(n)\big) + \alpha\Delta\vartheta(n-1) + \beta\Delta\vartheta(n-2)$$

Algoritam se uz momentum prvog reda ubrzava i do deset puta, ali ne garantira konvergenciju. Poznato je da se takav algoritam najbolje ponaša uz iznose $\alpha = 0{,}8 \div 0{,}9$.

Promjena se odvija od izlaznog prema ulaznom sloju mreže. Najprije se mijenjaju težinski koeficijenti $w_{xy}$ na sljedeći način:

$$w_{xy}(n+1) = w_{xy}(n) - \eta\nabla E(w_{xy}(n)) + \alpha\Delta w_{xy}(n-1) + \beta\Delta w_{xy}(n-2)$$

Zbog pojednostavljenog zapisivanja, a i korištenja oznake *n* u drugim izrazima, poput sume, koristit će se sljedeći izraz:

$$\Delta w_{xy}(b) = -\eta\nabla E\left(w_{xy}(b)\right) + \alpha * \Delta w_{xy}(b-1) + \beta * \Delta w_{xy}(b-2)$$

Gradijent pogreške za težine $w_{xy}$ će se izračunati prema:

$$\nabla E\big(w_{xy}\big) = \frac{\partial E(\vartheta)}{\partial w_{xy}}$$

Vidljivo je da je osnovni zadatak čitavog postupka učenja određivanje pripadajućeg gradijenta pogreške primjenom uzastopnih parcijalnih derivacija:

$$\nabla E\big(w_{xy}\big) = \frac{\partial E(\vartheta)}{\partial w_{xy}} = \frac{\partial E}{\partial A_y} * \frac{\partial A_y}{\partial O_y} * \frac{\partial O_y}{\partial net_{Oy}} * \frac{\partial net_{Oy}}{\partial w_{xy}}$$

Parcijalne derivacije su redom:

$$\frac{\partial E}{\partial A_y} \text{ iz } E = \frac{1}{2} * \sum\big(d_y - O_y\big)^2 = \frac{1}{2} * \sum A_y^2 = \frac{1}{2} * 2 * A_y$$

$$\frac{\partial A_y}{\partial O_y} \text{ iz } A_y = (d_y - O_y)^2 = -1$$

$$\frac{\partial O_y}{\partial net_{Oy}} \text{ iz } O_y = K * net_{Oy} = K$$

$$\frac{\partial net_{Oy}}{\partial w_{xy}} \text{ iz } net_{Oy} = \sum w_{xy} * y_{2x} = y_{2x}$$

Ako izračunate parcijalne derivacije vratimo u izraz za $\nabla E(w_{xy})$, dobivamo konačni izraz za promjenu težinskih koeficijenata $w_{xy}$:

$$\nabla E(w_{xy}) = \frac{\partial E(\vartheta)}{\partial w_{xy}} = \frac{1}{2} * 2 * A_y * (-1) * K * y_{2x}$$

$$\nabla E(w_{xy}) = \frac{\partial E(\vartheta)}{\partial w_{xy}} = \frac{1}{2} * 2 * A_y * (-K * y_{2x})$$

Istim postupkom se mijenjaju ostali težinski koeficijenti i parametri Gauss-ove aktivacijske funkcije $\sigma$ i $c$.

## 2.2 Primjer za izvedeni model neuronske mreže

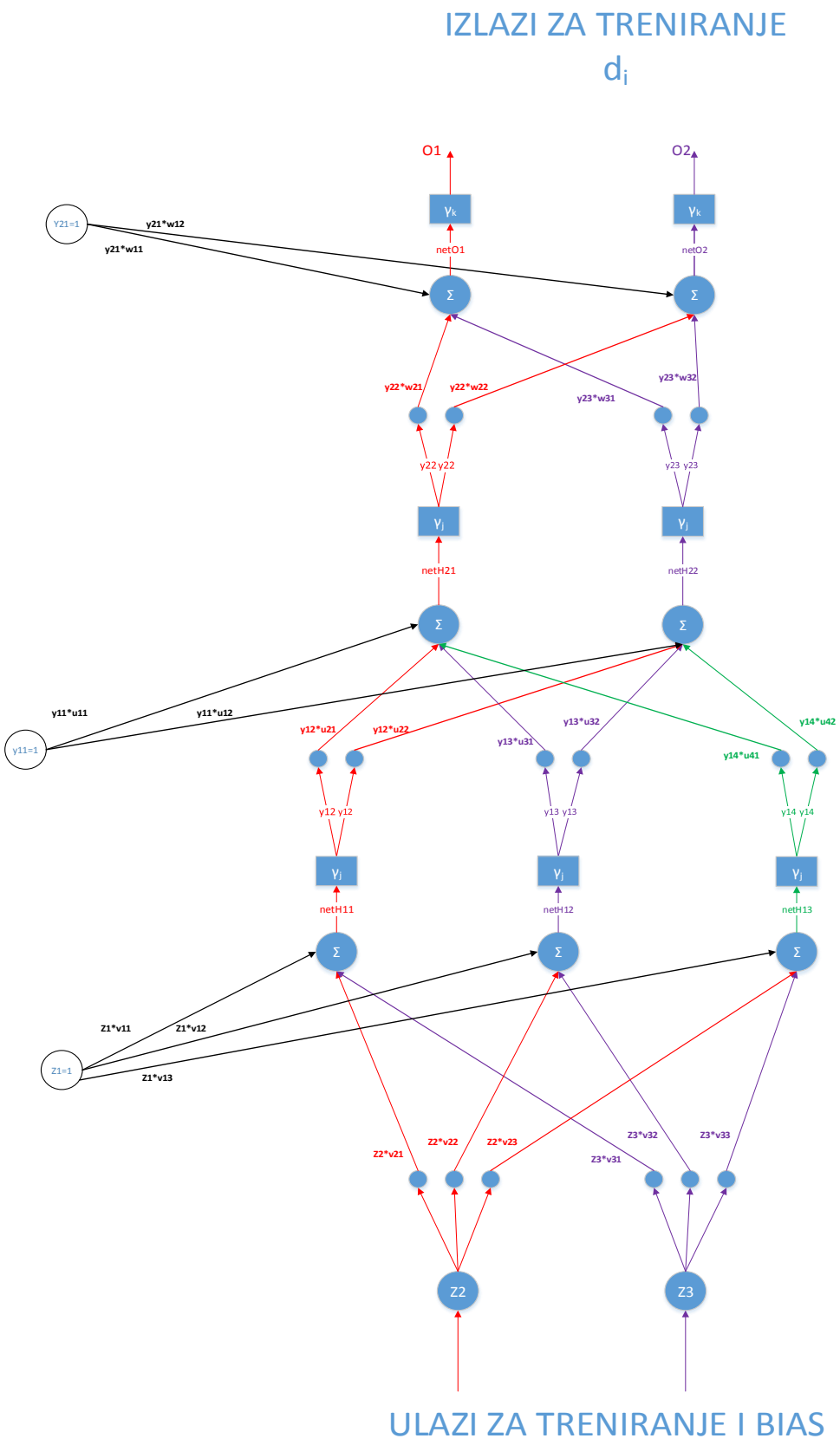
Da bi se postiglo bolje razumijevanje izvedenog modela, bit će prikazan primjer neuronske mreže s parametrima prema *tablici 6*. Unutar *tablice 7* su raspisani izrazi za model neuronske mreže prema *tablici 2* koji je vidljiv na *slici 5*.

**Tablica 6. Parametri unutar primjera**

| | |
|---|---|
| **M**<br>Broj ulaza + Bias | 2+1 |
| **I**<br>Broj izlaza | 2 |
| **$K_1$**<br>broj neurona u prvom sloju od ulaza u skrivenom sloju + Bias | 3+1 |
| **$K_2$**<br>broj neurona u drugom sloju od ulaza u skrivenom sloju + Bias | 2+1 |
| **N**<br>Broj elemenata u skupu za učenje | 2 |

**Tablica 7. Izrazi**

| |
|---|
| $E = \frac{1}{2} * (\Delta_1^2 + \Delta_2^2)$ |
| $\Delta_1 = d_1 - O_1$<br>$\Delta_2 = d_2 - O_2$ |
| $O_1 = K * (y_{21} * w_{11} + y_{22} * w_{21} + y_{23} * w_{31})$<br>$O_2 = K * (y_{21} * w_{12} + y_{22} * w_{22} + y_{23} * w_{32})$ |
| $net_{O1} = y_{21} * w_{11} + y_{22} * w_{21} + y_{23} * w_{31}$<br>$net_{O2} = y_{21} * w_{12} + y_{22} * w_{22} + y_{23} * w_{32}$ |
| $y_{21} = 1$<br>$y_{22} = e^{-\frac{1}{2}*\left(\frac{y_{11}*u_{11}+y_{12}*u_{21}+y_{13}*u_{31}+y_{14}*u_{41}-c_{21}}{\sigma_{21}}\right)^2}$<br>$y_{23} = e^{-\frac{1}{2}*\left(\frac{y_{11}*u_{12}+y_{12}*u_{22}+y_{13}*u_{32}+y_{14}*u_{42}-c_{22}}{\sigma_{22}}\right)^2}$ |
| $net_{H21} = y_{11} * u_{11} + y_{12} * u_{21} + y_{13} * u_{31} + y_{14} * u_{41}$<br>$net_{H22} = y_{11} * u_{12} + y_{12} * u_{22} + y_{13} * u_{32} + y_{14} * u_{42}$ |
| $y_{11} = 1$<br>$y_{12} = e^{-\frac{1}{2}*\left(\frac{Z_1*v_{11}+Z_2*v_{21}+Z_3*v_{31}-c_{11}}{\sigma_{11}}\right)^2}$<br>$y_{13} = e^{-\frac{1}{2}*\left(\frac{Z_1*v_{12}+Z_2*v_{22}+Z_3*v_{32}-c_{12}}{\sigma_{12}}\right)^2}$<br>$y_{14} = e^{-\frac{1}{2}*\left(\frac{Z_1*v_{13}+Z_2*v_{23}+Z_3*v_{33}-c_{13}}{\sigma_{13}}\right)^2}$ |
| $Z_1 = 1$<br>$net_{H11} = Z_1 * v_{11} + Z_2 * v_{21} + Z_3 * v_{31}$<br>$net_{H12} = Z_1 * v_{12} + Z_2 * v_{22} + Z_3 * v_{32}$<br>$net_{H13} = Z_1 * v_{13} + Z_2 * v_{23} + Z_3 * v_{33}$ |




**Slika 5. Arhitektura neuronske mreže za konkretni primjer**

# 3 Korištenje programa

## 3.1. Izbornik

Unutar tijela programa nalazi se izbornik sa *slike 6* kojim je moguće odabrati konfiguraciju neuronske mreže i parametre unutar mreže. Također, moguće je odabrati momentum prvog ili drugog reda, te parametre koji se koriste u izračunu momentuma ($\alpha$ *i* $\beta$). Varijabla *Sa_Bias* određuje hoće li se koristiti *Bias* neuron u procesu učenja, a *ISTE_TEŽINE* hoće li se generirati nove težine, koristiti spremljene ili generirati nove težine bez spremanja.

```
% ------------------------------------------------------------------------
% 3. Izbornik
% ------------------------------------------------------------------------
    K1=10;
    K2=7;
    K=1;
    iter=4000;   % zadani broj iteracija
    ni=0.5;     % inicijalizacija parametra učenja ni € [0.001, 10]
    zeljena_greska=0.5;     % staje kada je pogreška jednaka 0.5%


    Sa_bias=1;   % 0 znači da se ne koristi bias, 1 znači da se koristi bias,

    momentum=0;  % 0 znači da se ne koristi momentum, 1 znači momentum prvog reda, 2 znači momentum drugog reda
    alfa=0.8;     % koeficijent momentuma prvog reda alfa € [0.1, 0.9]
    beta=0.06;     % koeficijent momentuma za član drugog reda beta €  ((1-alfa)/3)

    % ------------------------------------------------------------------------
    % Odabir načina rada gdje se koriste iste težine
    % ------------------------------------------------------------------------
    ISTE_TEZINE=2;  % 1 - koristenje istih tezina u buducem ucenju
                    % 2 - ucitavanje spremljenih tezina
                    % 0 - koristenje razlicitih tezina
```

**Slika 6. Izbornik unutar tijela programa**

## 3.2 Učitavanje podataka

Podaci za učenje učitavaju se putem Excel dokumenta *"Input_Output.xlsx"*. Excel dokument se sastoji od pet *sheet-ova:*

- Input: unutar kojeg se upisuju ulazni podaci, bez *Bias-a*

- Output: unutar kojeg se upisuju ili računaju izlazni podaci

- Input_Test: unutar kojeg se upisuju ulazni podaci za testiranje, bez *Bias-a*

- Output_Test: unutar kojeg se upisuju ili računaju izlazni podaci za testiranje

- Rand: unutar kojeg se generiraju slučajni brojevi za potrebe simulacije

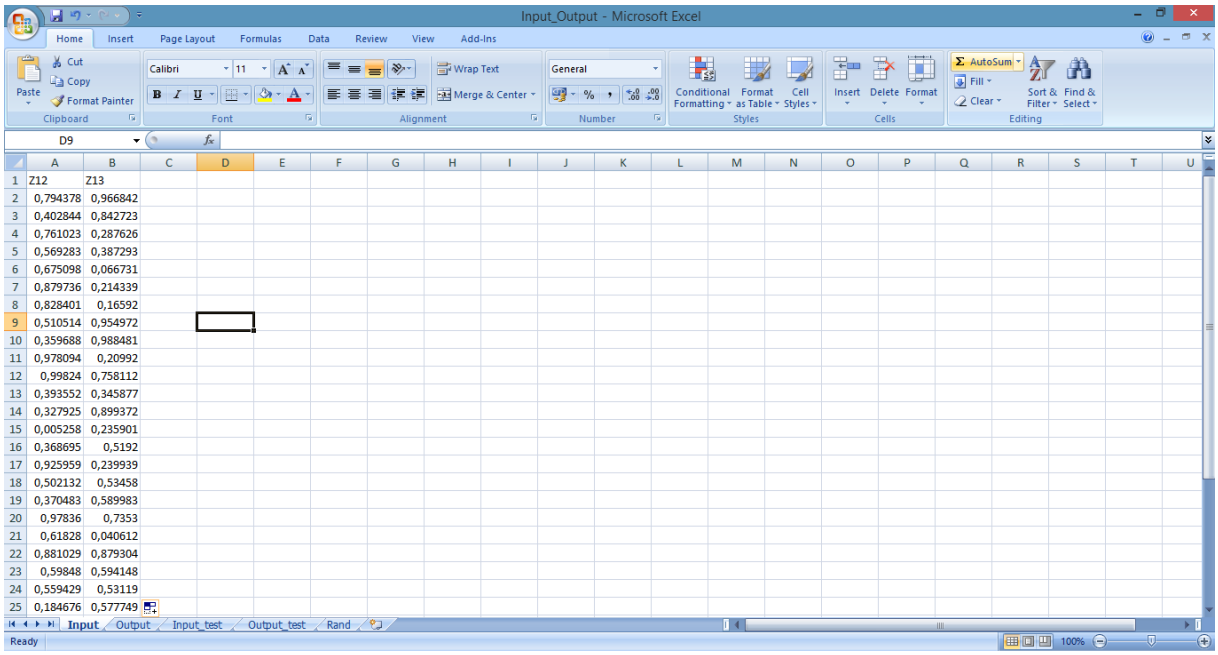
Funkcija koja u petom *sheet-u* generira slučajne brojeve u intervalu između 0 i 1 je *"=rand()"*. Vrijednosti se kopiraju i opcijom *"Paste-Values"* prenesu za Input ili Output.

Ulazni i izlazni podaci su uvijek normirani na jediničnu normalnu razdiobu prema formulama:

$$Z_{normirani} = \frac{Z_{željeni} - \mu}{\sigma}, \text{ i}$$

$$d_{normirani} = \frac{d_{željeni} - \mu}{\sigma}$$

Pri tome $\mu$ predstavlja očekivanje, tj. srednju vrijednost ulaznog ili izlaznog vektora, a σ standardnu devijaciju [4]. Rezultati učenja su prikazani za normirane podatke.




**Slika 7. Izgled Excel dokumenta za unos podataka**

## 3.3 Izvođenje programa

Tijekom izvođenja programa prikazuje se sljedeći ekran koji sadržava sljedeće mogućnosti:

- Stop: proces učenja se završava

- Reset Wts: resetiranje težinskih vrijednosti (za jedan red veličine) i parametara aktivacijskih funkcija na način da se istima pridjele nove slučajne vrijednosti

- Povećaj: povećavanje koeficijenta brzine učenja η za 2 puta

- Smanji: smanjenje koeficijenta brzine učenja η za 2 puta

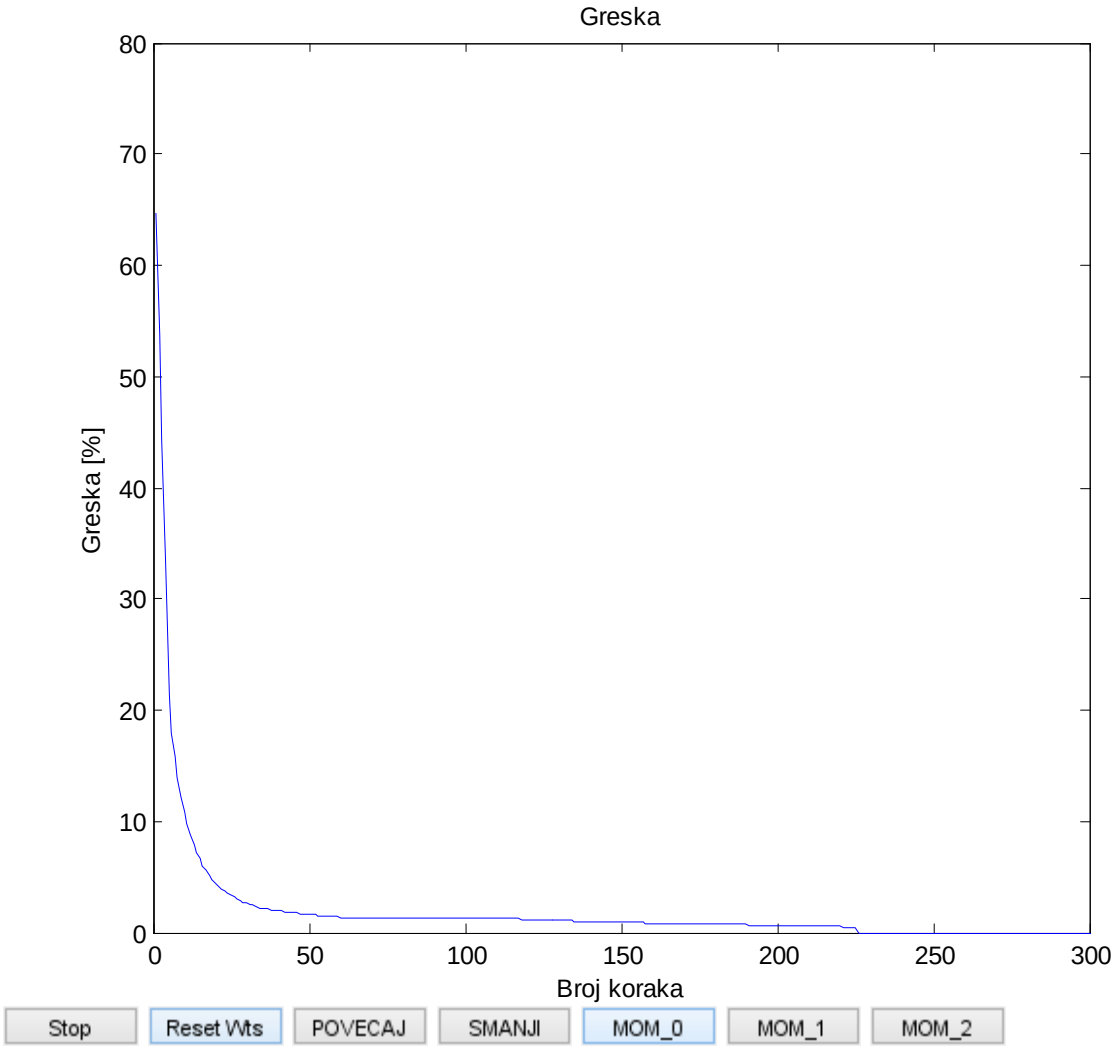
- Mom_0: učenje neuronske mreže bez momentuma

- Mom_1: učenje neuronske mreže s momentumom prvog reda

- Mom_2: učenje neuronske mreže s momentumom drugog reda

Greška se prikazuje u postotku. Program u svakom koraku pamti iznos pogreške na osnovu razlike željenog i ostvarenog izlaza te na kraju matrično računa srednju vrijednost svih apsolutnih postotnih pogrešaka. Ako je ostvarena pogreška manja od 0,5%, program staje.

$$Greška = \frac{|O - d|}{d} * 100\%$$




**Slika 8. Grafički prikaz procesa učenja u programu**

# 4 Korištenje programa za rješavanje klasifikacijskih i regresijskih problema

U ovom poglavlju će biti prikazano učenje i testiranje neuronske mreže na nekoliko različitih problema.

## 4.1 XOR klasifikacijski problem

Ekskluzivni ILI (eng. XOR), poznati je logički problem Booleove algebre. Najjednostavniji je primjer s linearno neseparabilnim uzorcima [5] i zato često primjenjivan za ispitivanje svojstava različitih modela umjetnih neuronskih mreža. Dokazano je da jednoslojna perceptronska mreža ne može riješiti ovako jednostavan problem te se upravo zato uvode skriveni slojevi. Tablica istine ove logičke operacije prikazana je *tablicom 8.*

**Tablica 8. XOR problem - tablica istine**

| Ulaz | | Izlaz |
|---|---|---|
| A | B | A XOR B |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

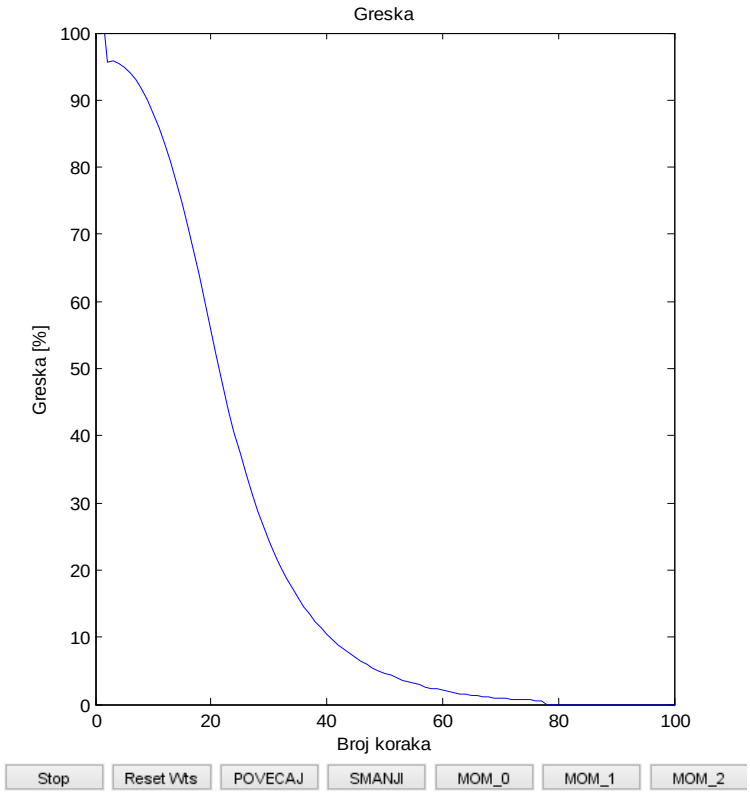
Ulaz je određen s dvije binarne varijable koje mogu poprimiti vrijednost nula ili jedan. Izlaz se sastoji od jedne binarne varijable, koja treba poprimiti vrijednost jedan ukoliko je samo jedna od binarnih ulaznih varijabli jednaka jedan, odnosno nula ukoliko obje ulazne varijable imaju istu vrijednost.

Sljedeća tablica prikazuje rezultate učenja neuronske mreže za zadane ulazno izlazne podatke.

**Tablica 9. Rezultati učenja XOR klasifikacijskog problema**

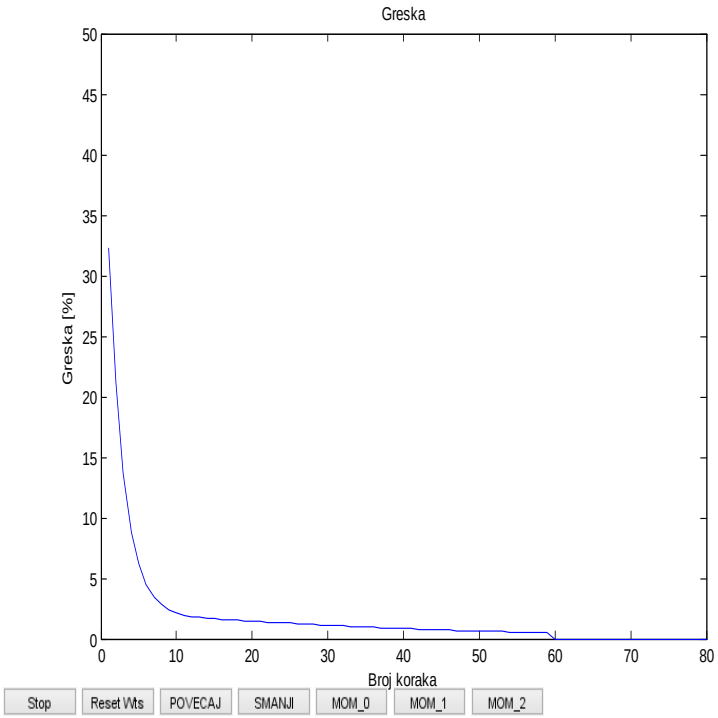
**a) K1=3, K2=2, η=0,1, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 78. iteraciji



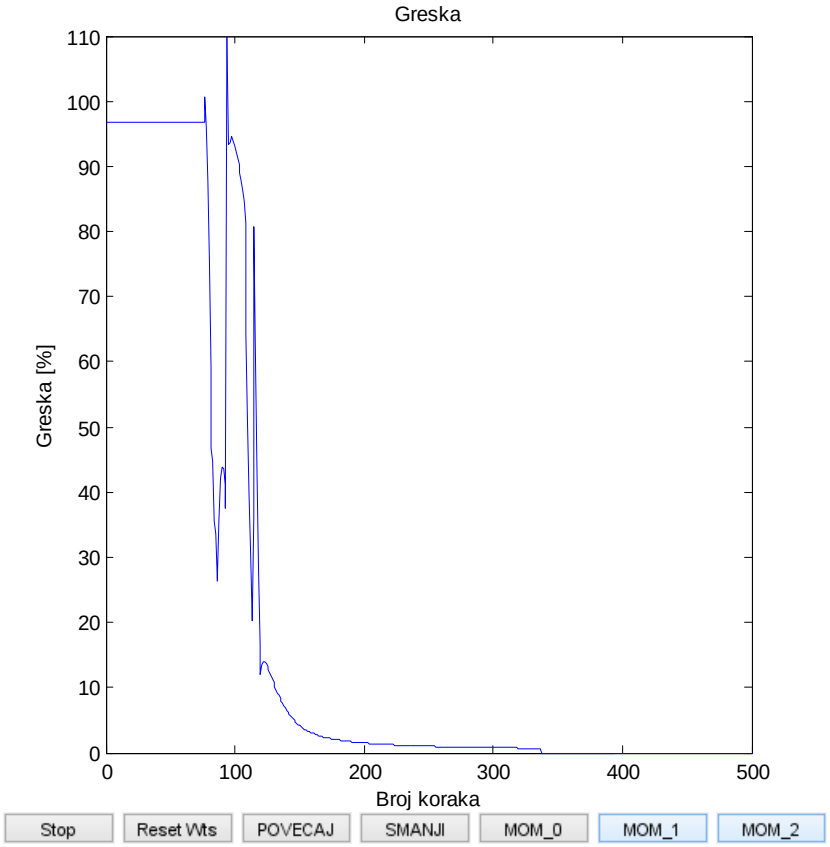

**b) K1=4, K2=4, η=0,4, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 60. iteraciji



**c) K1=6 K2=5, η=1, Bias=1, Momentum=0 (α=0, β=0)**

U ovom primjeru je početni koeficijent brzine učenja bio zadan prevelik. Smanjen je na vrijednost η=0,5 u 76. iteraciji i na vrijednost η=0,25 u 331. iteraciji. Učenje se općenito odvijalo sporije zbog većeg broja neurona u skrivenim slojevima. Konvergencija je postignuta u 338. iteraciji




Iz prikazanih primjera je vidljivo da je neuronska mreža uspješno naučila zadane ulazno izlazne podatke. Različite simulacije učenja pokazuju da brzina konvergencije najviše ovisi o načinu odabira iznosa koeficijenta brzine učenja i o slučajno generiranim vrijednostima težina i parametara aktivacijske funkcije *c* i *σ*.

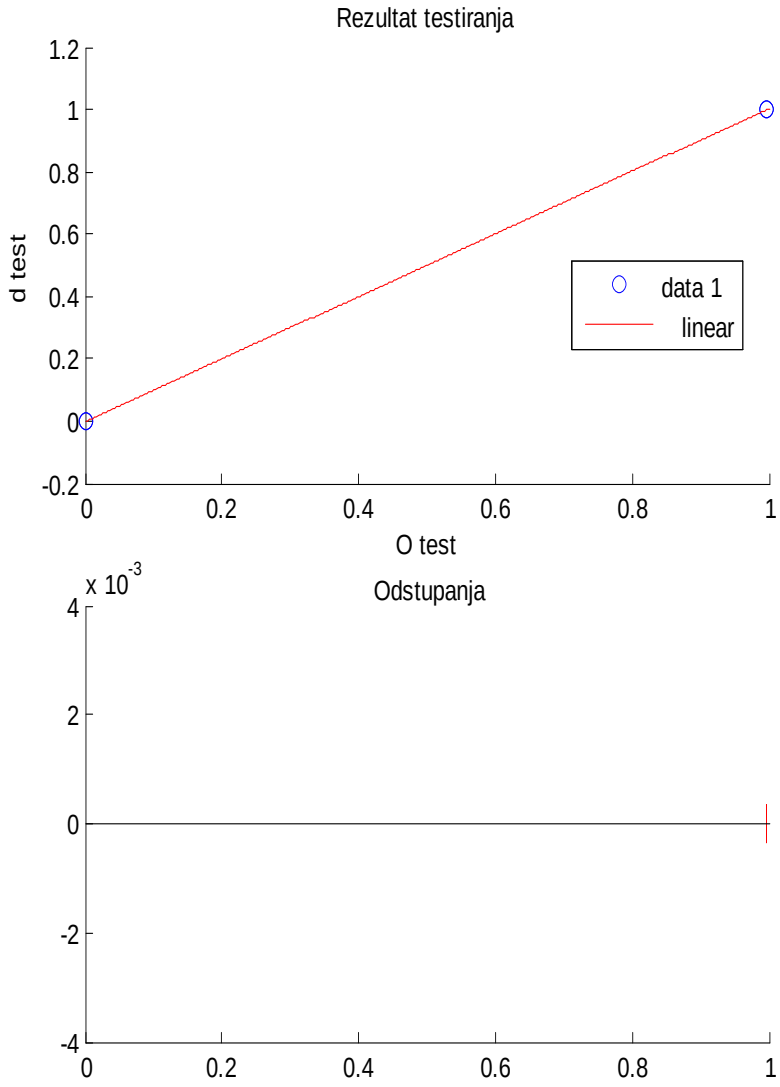
Nakon procesa učenja slijedi proces testiranja neuronske mreže. To se radi s novim skupom ulaza mreže koji nije bio sadržan u ulaznom skupu za vrijeme procesa učenja. Mreža sada producira nove izlaze koji se uspoređuju sa željenim izlazima. Pritom se ne mijenjaju parametri (težine) mreže. Iznos pogreške mreže u procesu testiranja služi za ocjenu generalizacijskih svojstava mreže, tj. sposobnosti mreže da daje zadovoljavajuće rezultate i za skup ulaza kojim nije bila učena. Za XOR klasifikacijski problem je za *slučaj a)* zamijenjen redosljed ulazno izlaznih podataka za učenje kako bi se ustanovilo hoće li neuronska mreža uspješno završiti testiranje. Rezultati testiranja su prikazani *slikom 9.*

**Tablica 10. Podaci za testiranje XOR klasifikacijskog problema**

| Ulaz | | Izlaz |
|---|---|---|
| 1 | 1 | 0 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 0 | 1 |

**K1=3, K2=2, η=0,1, Bias=1, Momentum=0 (α=0, β=0)**

**S istim težinama generiranim u *slučaju a)***




**Slika 9. Rezultati testiranja XOR klasifikacijskog problema**

Na apscisi su ostvareni izlazi koje je neuronska mreža izračunala iz ulaznih podataka za testiranje, a na ordinati željene izlazne vrijednosti za testiranje. Opcijom *plot residuals* je vidljivo da je neuronska mreža završila testiranje uz minimalna odstupanja.

## 4.2 Prošireni XOR klasifikacijski problem

Za ulazno izlazne vrijednosti iz *tablice 11* testirala se mogućnost učenja neuronske mreže. Radi se o proširenom XOR klasifikacijskom problemu s 3 ulazna vektora [6].

Unutar *tablice 12* vidljivi su rezultati učenja u ovisnosti o broju neurona i parametru učenja η.

**Tablica 11. Ulazno izlazne vrijednosti proširenog XOR klasifikacijkog problema**

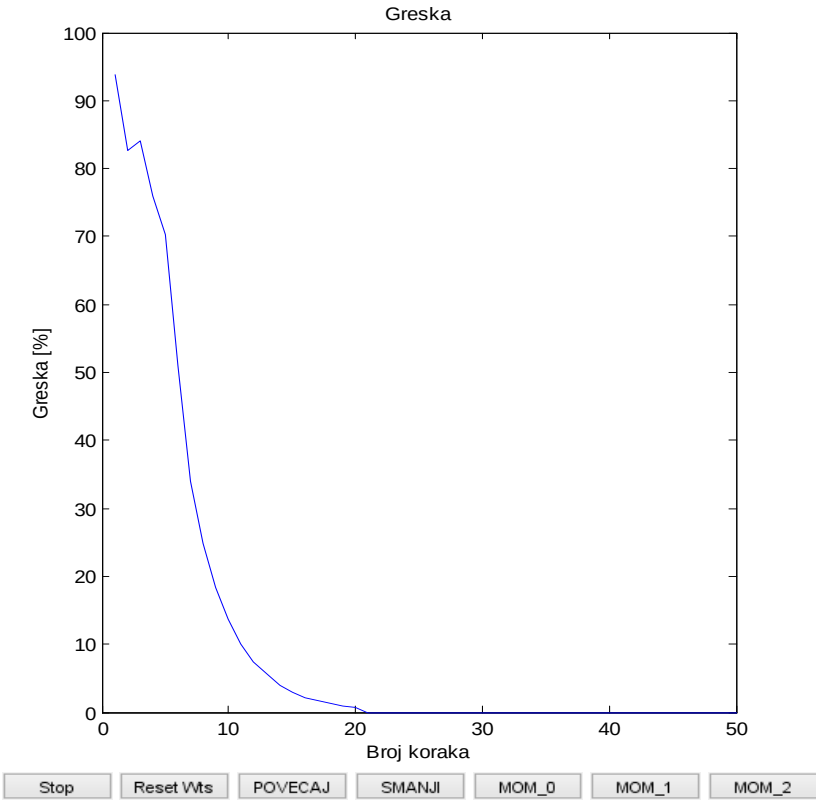
| Ulaz | | | Izlaz |
|---|---|---|---|
| A | B | C | A XOR B XOR C |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

Nakon procesa učenja provedeno je testiranje na način da su ponovno zamijenjeni neki redovi ulazno izlaznih podataka s težinama iz *slučaja a).* Na osnovu grafičkog prikaza je vidljivo da su odstupanja i u ovom slučaju minimalna.

**Tablica 12. Rezultati učenja proširenog XOR klasifikacijskog problema**

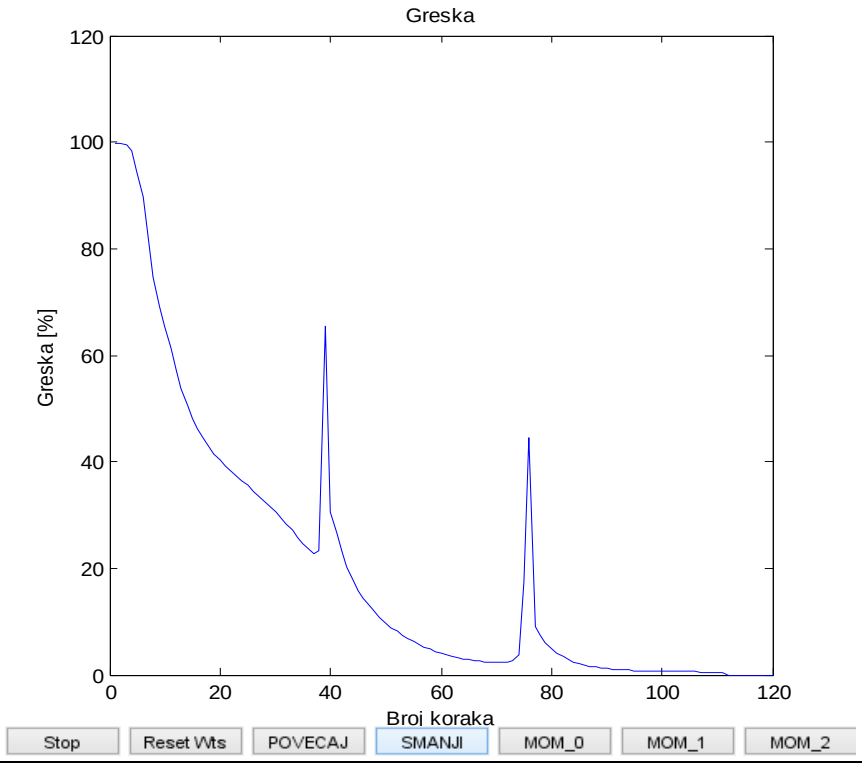
**a) K1=4, K2=3, η=0,2, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 21. iteraciji




**b) K1=5, K2=5, η=0,1, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 112. iteraciji

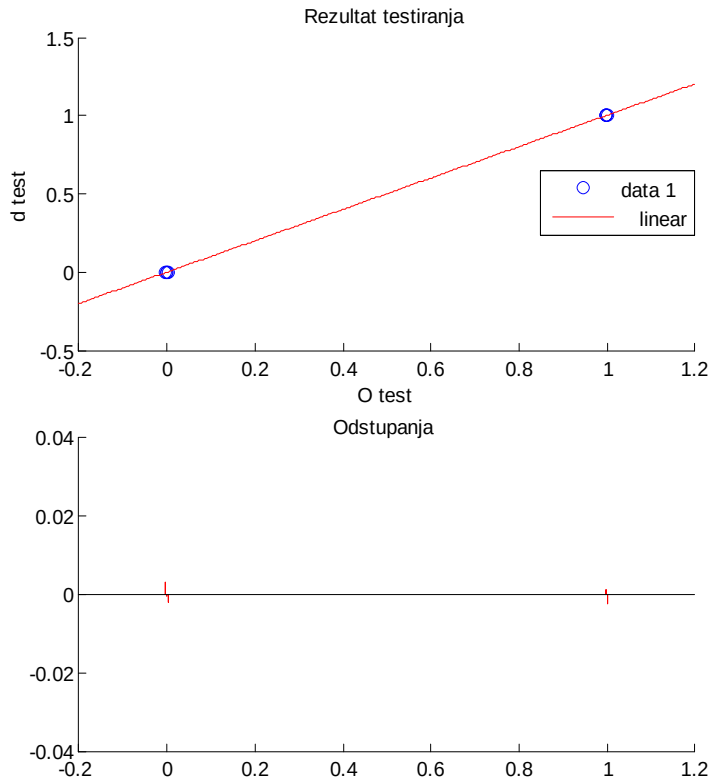


**Tablica 13. Podaci za testiranje proširenog XOR klasifikacijskog problema**

| Ulaz | | | Izlaz |
|---|---|---|---|
| A | B | C | A XOR B XOR C |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 |

**K1=4, K2=3, η=0,2, Bias=1, Momentum=0 (α=0, β=0)**

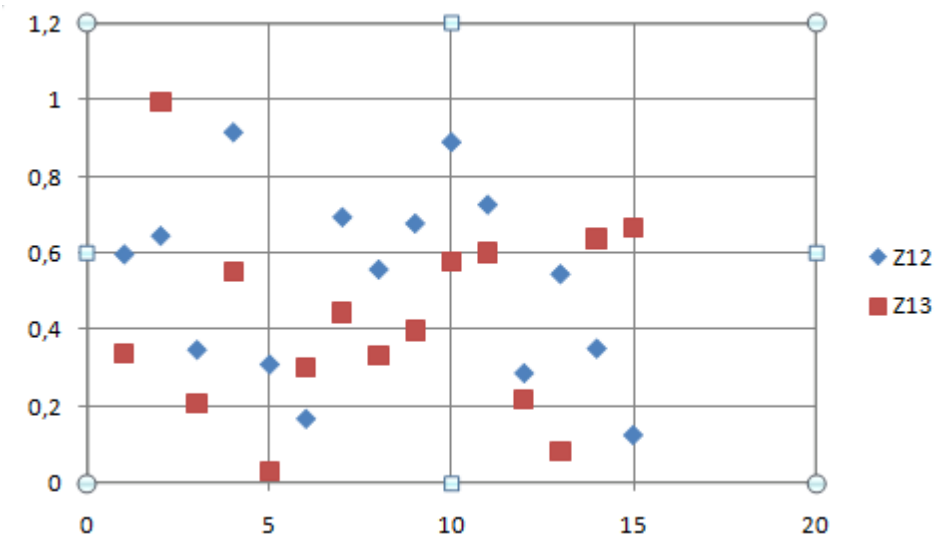
**S istim težinama generiranim u *slučaju a)***




**Slika 10. Rezultat testiranja proširenog XOR klasifikacijskog problema**

## 4.3 Regresijski problem za linearnu funkciju O1 = 3*Z12 + 2 * Z13

U ovom problemu generatorom slučajnih brojeva generirana su dva vektora (Z12 i Z13) veličine 15 redova. Iz ta dva vektora dobivena je funkcija s dvije varijable O1=3*Z12+2*Z13 te je taj problem dan na učenje neuronskoj mreži.



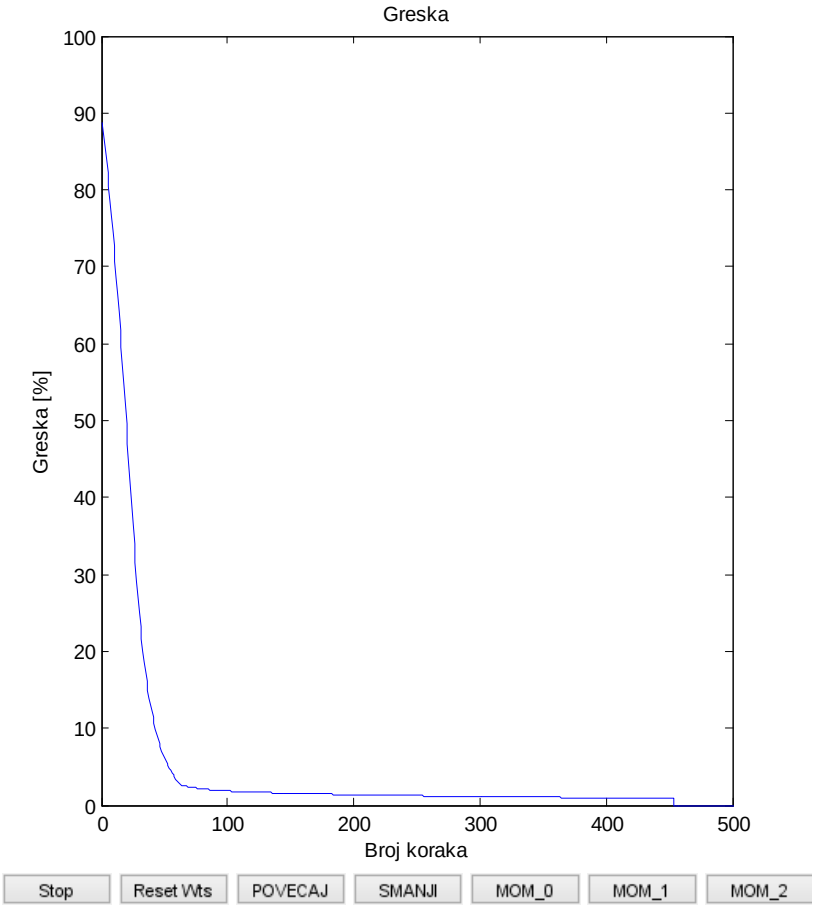

**Slika 11. Slučajno generirane vrijednosti vektora Z12 i Z13**

*Slika 11* prikazuje ulazne podatke vektora Z12 i Z13. Primjerice, prva slučajna vrijednost vektora Z12 između 0 i 1 je 0,598904325, a vektora Z13 je 0,337456849 itd.

Slijede rezultati učenja i testiranja neuronske mreže na ovom linearnom problemu.

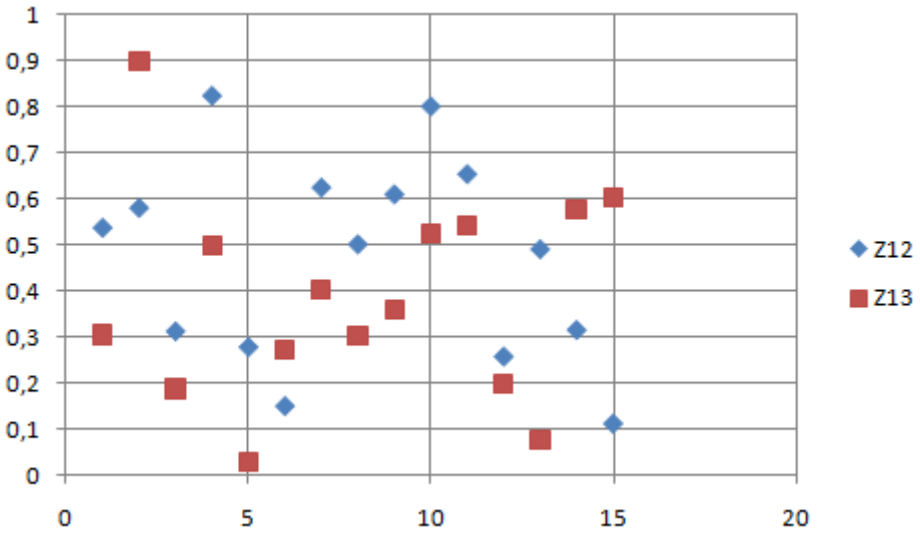
**K1=7, K2=5, η=0,04, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija je postignuta u 454. iteraciji.

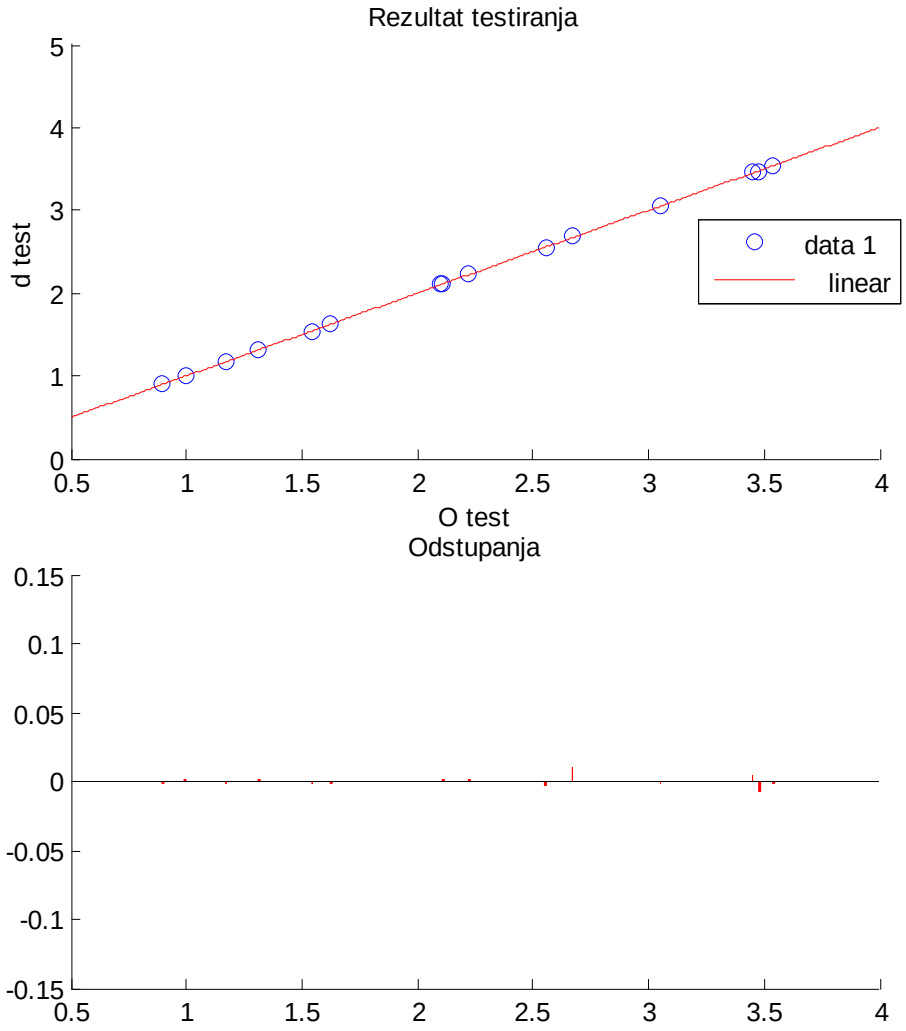



**Slika 12. Rezultat učenja linearne funkcije O1=3*Z12+2*Z13**

Za testiranje su iznosi vektora Z12 i Z13 promijenjeni na način da su umanjeni za 1%.




**Slika 13. Slučajno generirane vrijednosti vektora Z12 i Z13 umanjene za 1%**

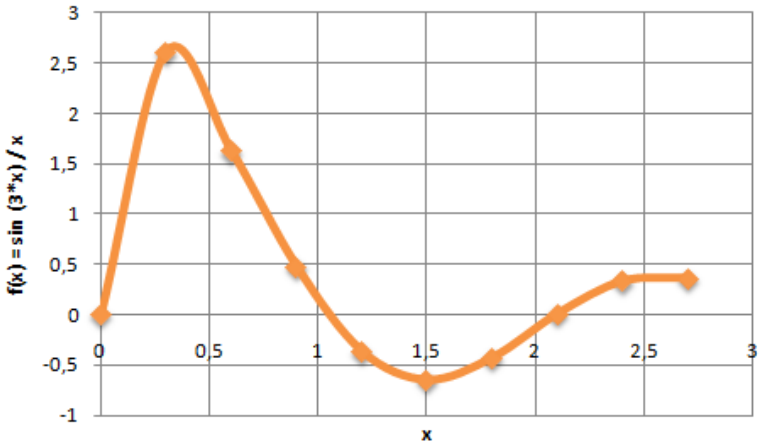



**Slika 14. Rezultat testiranja linearne funkcije O1=3*Z12+2*Z13**

Vidljivo je da je neuronska mreža uz minimalna odstupanja uspjela odraditi testiranje na ulaznim podacima na kojima nije učila.

## 4.4 Aproksimacija nelinearne dinamike prema funkciji f(x)= sin(3*x)/x

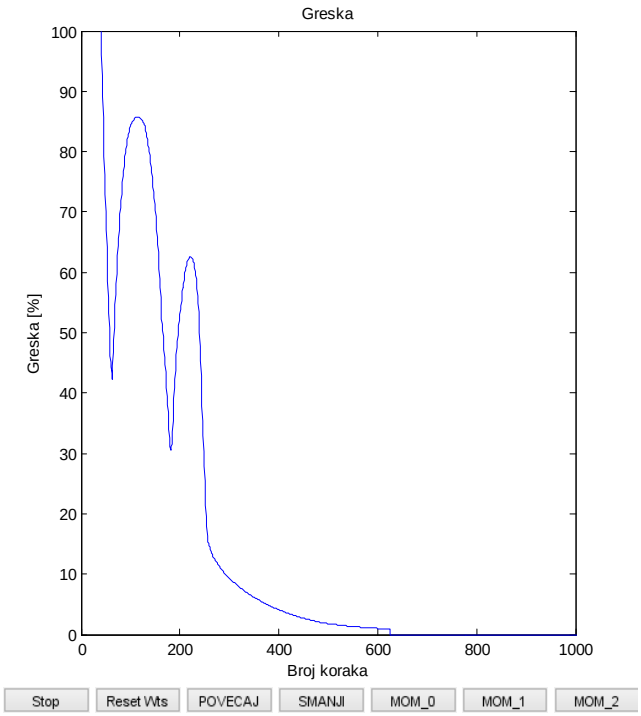
Unutar ovog potpoglavlja predstavljena je mogućnost da neuronska mreža nauči nelineranu funkciju $\frac{\sin(3*x)}{x}$ na setu ulaznih podataka sa *slike 15.*




**Slika 15. Podaci za učenje nelinearne funkcije f(x)=sin(3*x)/x**

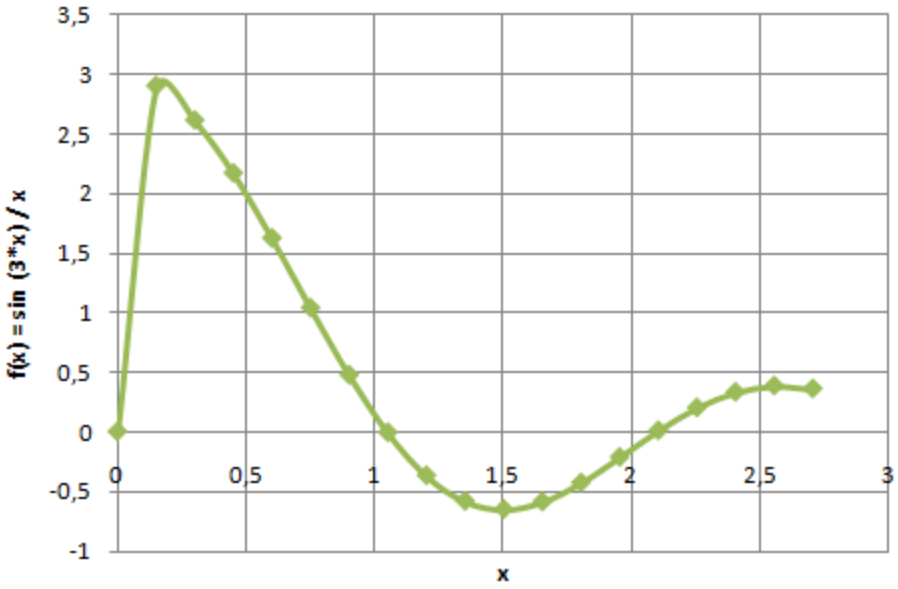
Skup za učenje sastoji se od 10 točaka, jednakomjerno raspodijeljenih unutar intervala [0, 3] s korakom 0,3. Konvergencija je postignuta u 626. iteraciji.

**K1=7, K2=5, η=0,006, Bias=1, Momentum=0 (α=0, β=0)**

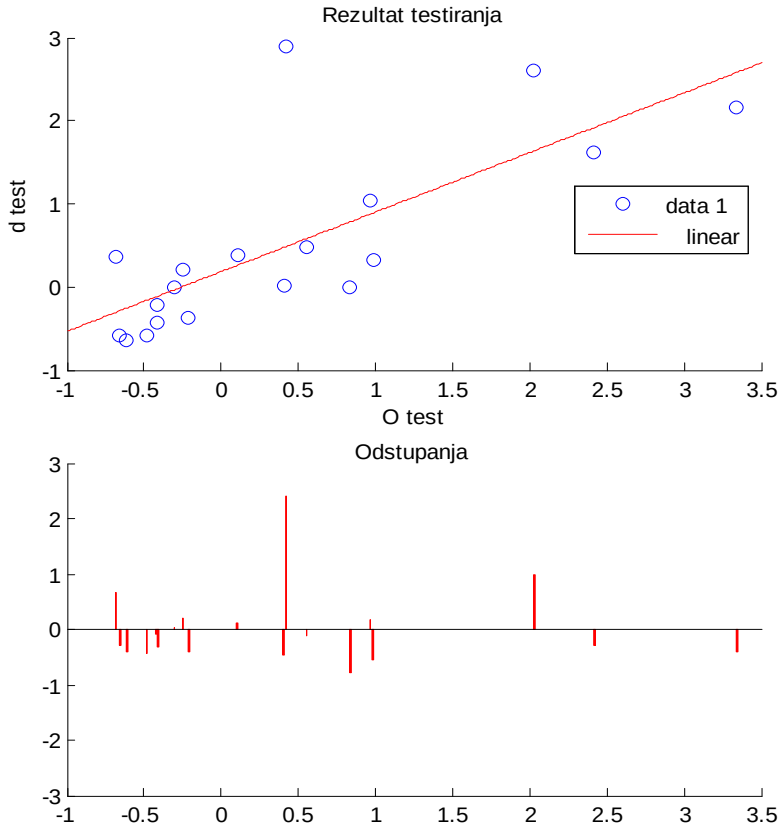



**Slika 16. Rezultat učenja nelinearne funkcije f(x)=sin(3*x)/x**

Za potrebe testiranja neuronske mreže na ovom problemu odabran je uži razmak unutar intervala [0, 3] što je vidljivo na *slici 17.* S korakom 0,15 se ulazni set podataka za testiranje sada sastoji od 19 točaka.




**Slika 17. Podaci za testiranje neuronske mreže za nelinearnu funkciju f(x)=sin(3*x)/x**




**Slika 18. Rezultat testiranja nelinearne funkcije f(x)=sin(3*x)/x**

Za razliku od dosadašnjih izvršenih testiranja, na ovom problemu su vidljiva veća odstupanja. Neuronska mreža je uspjela naučiti učene ulazno izlazne podatke, ali nije u potpunosti naučila zadanu dinamiku.

# 5 Analiza utjecaja momentuma prvog i drugog reda na proces učenja

Program ima mogućnost korištenja momentuma prvog i drugog reda. Na konkretnim problemima testiran je utjecaj momentuma koji ubrzava proces učenja. Zbog lakše usporedbe, sve tablice u ovom poglavlju najprije prikazuju tijek učenja bez momentuma. Koeficijent β za momentum drugog reda se dobiva na osnovu formule $\beta = \frac{1-\alpha}{3}$.

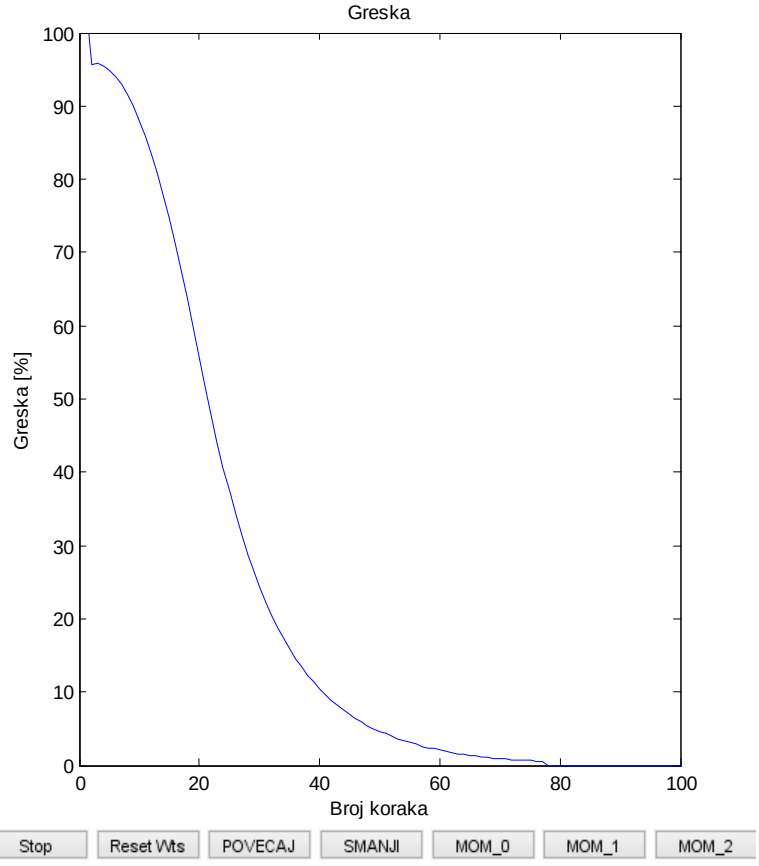
## 5.1 XOR klasifikacijski problem - utjecaj s momentumom

*Tablica 14* prikazuje tijek učenja za XOR klasifikacijski problem s istim težinama generiranim u *slučaju a)* poglavlja 4.1 ovog rada. Učenje je provedeno najprije bez momentuma, a zatim s momentumom prvog i drugog reda. Koeficijent brzine učenja η, arhitektura neuronske mreže i zadani broj iteracija se pri tome nisu mijenjali.

**Tablica 14. Utjecaj momentuma na XOR klasifikacijski problem**

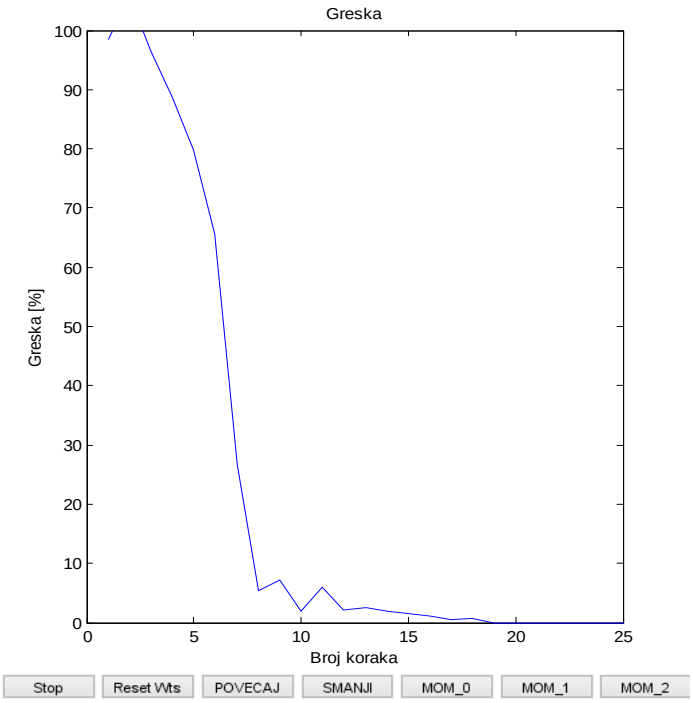
**a) K1=3, K2=2, η=0,1, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 78. iteraciji



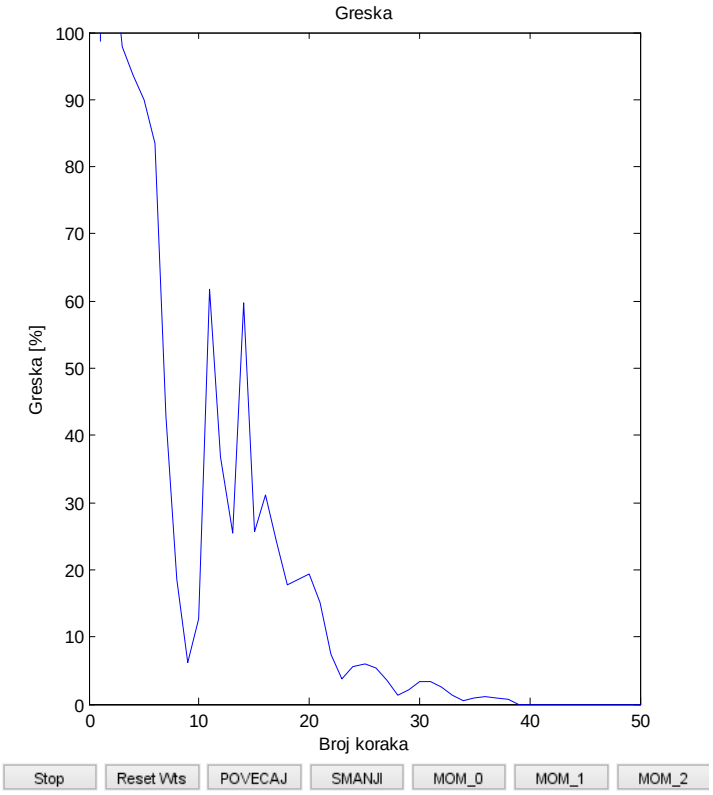
**b) K1=3, K2=2, η=0,1, Bias=1, Momentum=1 (α=0,9, β=0)**

Konvergencija postignuta u 19. iteraciji




**c) K1=3, K2=2, η=0,1, Bias=1, Momentum=2 (α=0,9, β=0,03)**

Konvergencija postignuta u 39. iteraciji



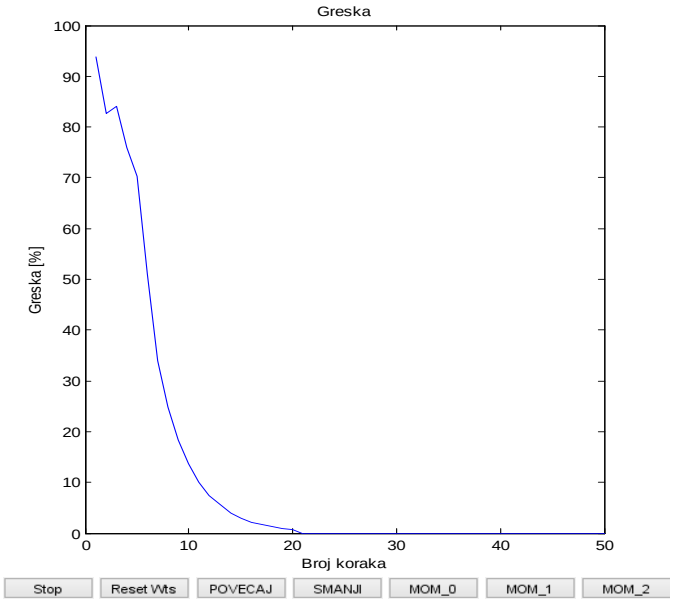
## 5.2 Prošireni XOR klasifikacijski problem - utjecaj s momentumom

Utjecaj momentuma je testiran i na proširenom XOR problemu s istim početnim težinama generiranim u *slučaju a)* iz poglavlja 4.2 bez mijenjanja ostalih parametara. Rezultati su prikazani u *tablici 15.*

**Tablica 15. Utjecaj momentuma na prošireni XOR klasifikacijski problem**

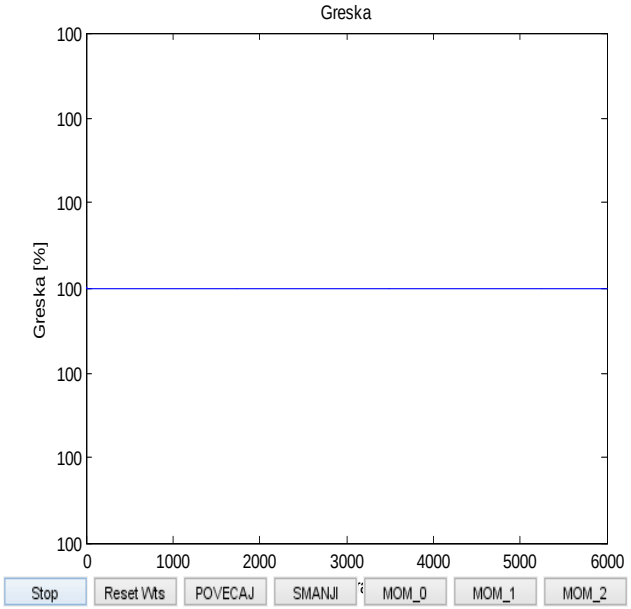
**a) K1=4, K2=3, η=0,2, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 21. iteraciji



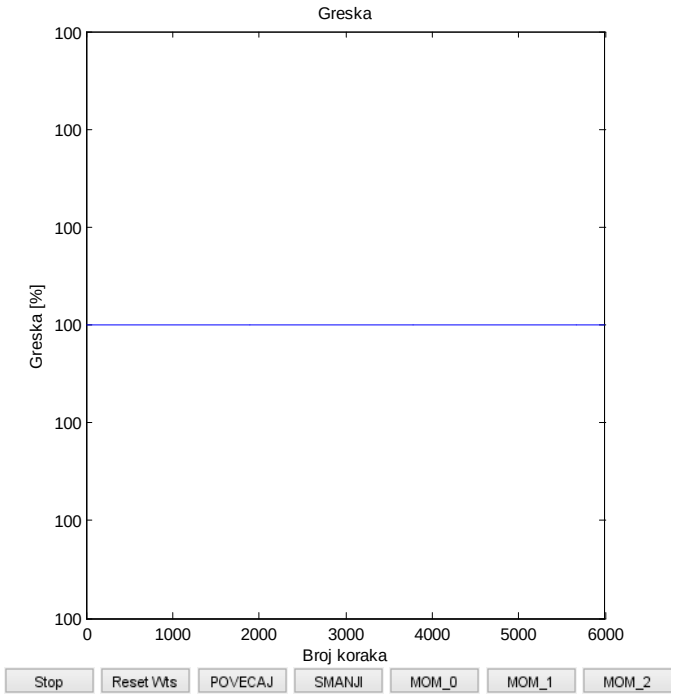

**b) K1=4, K2=3, η=0,2, Bias=1, Momentum=1 (α=0,9, β=0)**

Konvergencija nije postignuta



**b) K1=4, K2=3, η=0,2, Bias=1, Momentum=2 (α=0,9, β=0,03)**

Konvergencija nije postignuta




Vidljivo je da uvođenje momentuma kod proširenog XOR klasifikacijskog problema onemogućava postizanje kovergencije i učenje treba provoditi bez momentuma. S povoljno generiranim težinama, učenje s pogreškom od 0.5% se postiže dovoljno brzo i bez momentuma.

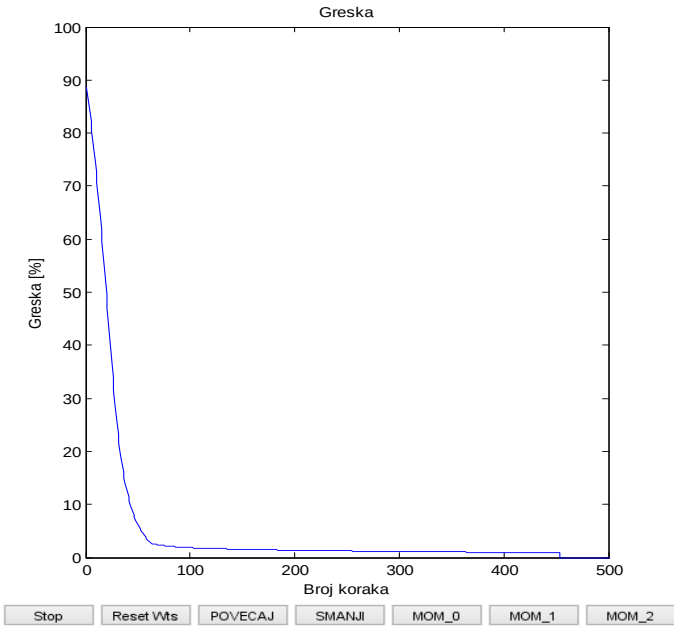
## 5.3 Regresijski problem za linearnu funkciju O1=3*Z12+2*Z13- utjecaj s momentumom

Rezultati učenja za problem prepoznavanja funkcije O1=3*Z12+2*Z13 bez momentuma, s momentumom prvog i drugog reda su prikazani u *tablici 16.*

**Tablica 16. Utjecaj momentuma na linearnu funkciju O1=3*Z12+2*Z13**

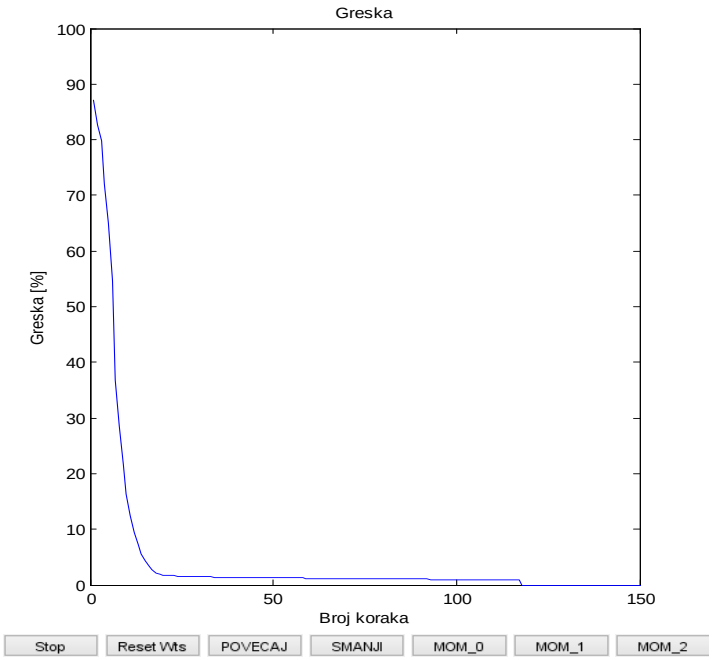
**a) K1=7, K2=5, η=0,04, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija je postignuta u 454. iteraciji



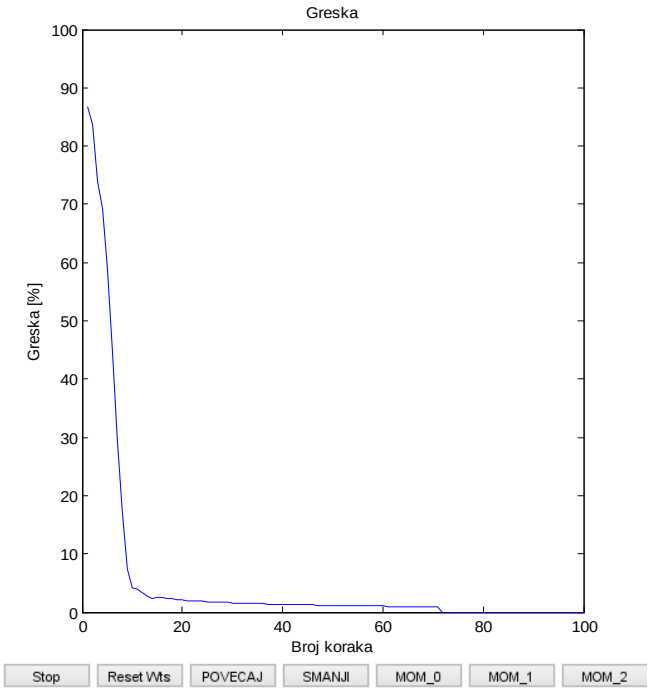

**b) K1=7, K2=5, η=0,04, Bias=1, Momentum=1 (α=0,7, β=0)**

Konvergencija postignuta u 118. iteraciji



**c) K1=7, K2=5, η=0,04, Bias=1, Momentum=2 (α=0,7, β=0,1)**

Konvergencija postignuta u 72. iteraciji




Kod ovog problema je momentum prvog, a pogotovo drugog reda ubrzao proces učenja jer je bilo potrebno puno manje iteracija za postizanje konvergencije.

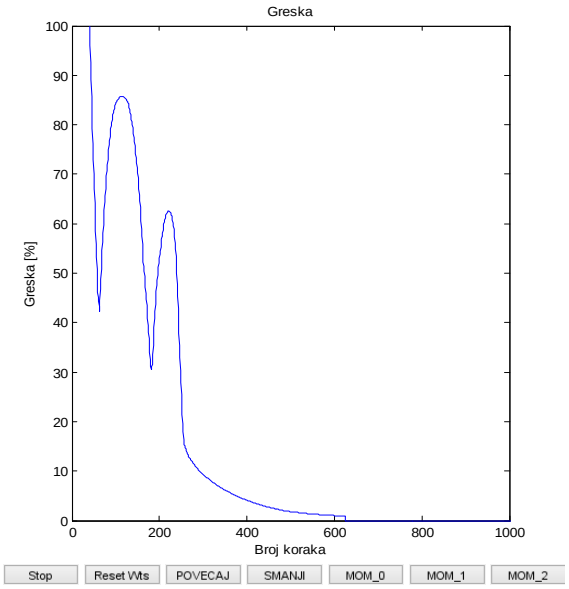
## 5.4 Aproksimacija nelinearne dinamike prema funkciji f(x)=sin(3*x)/x- utjecaj s momentumom

Rezultati učenja za problem prepoznavanja funkcije $\frac{\sin(3*x)}{x}$ bez momentuma, s momentumom prvog i drugog reda su prikazani u *tablici 17.*

**Tablica 17. Utjecaj momentuma na nelinearnu funkciju f(x)=sin(3*x)/x**

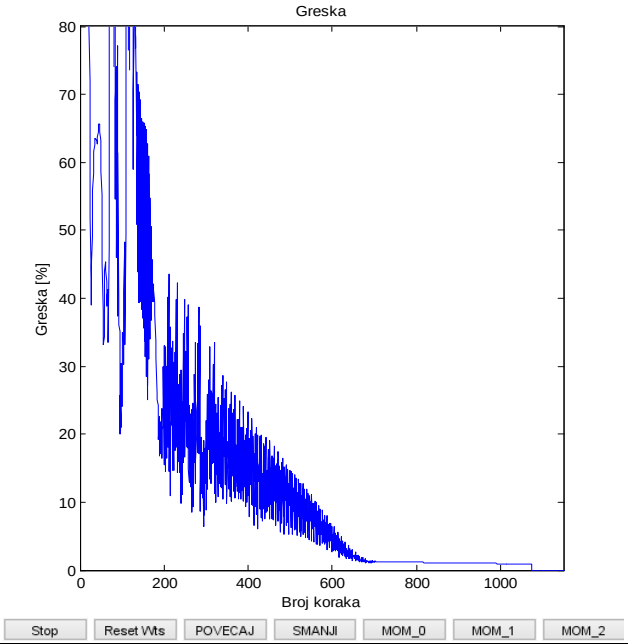
**a) K1=7, K2=5, η=0,006, Bias=1, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 626. iteraciji



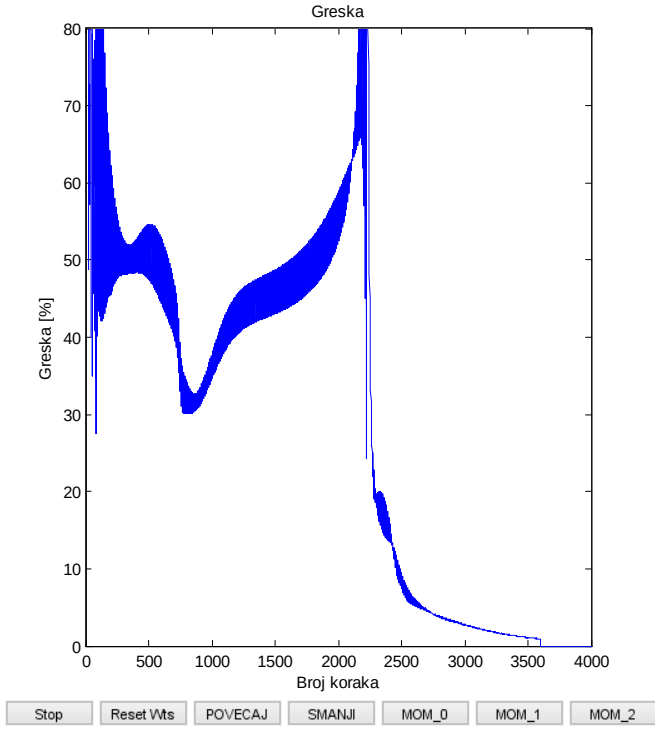

**b) K1=7, K2=5, η=0,006, Bias=1, Momentum=1 (α=0,7, β=0)**

Konvergencija postignuta u 1075. iteraciji



**c) K1=7, K2=5, η=0,006, Bias=1, Momentum=2 (α=0,7, β=0,1)**

Konvergencija postignuta u 3603. iteraciji




Kod učenja mreže na nelinearnom problemu momentumi nisu onemogućili konvergenciju, ali su dodatno produžili proces učenja. Drugim riječima i u ovom slučaju učenje je bolje provoditi bez momentuma.

# 6 Analiza mogućnosti nekorištenja Bias neurona u procesu učenja

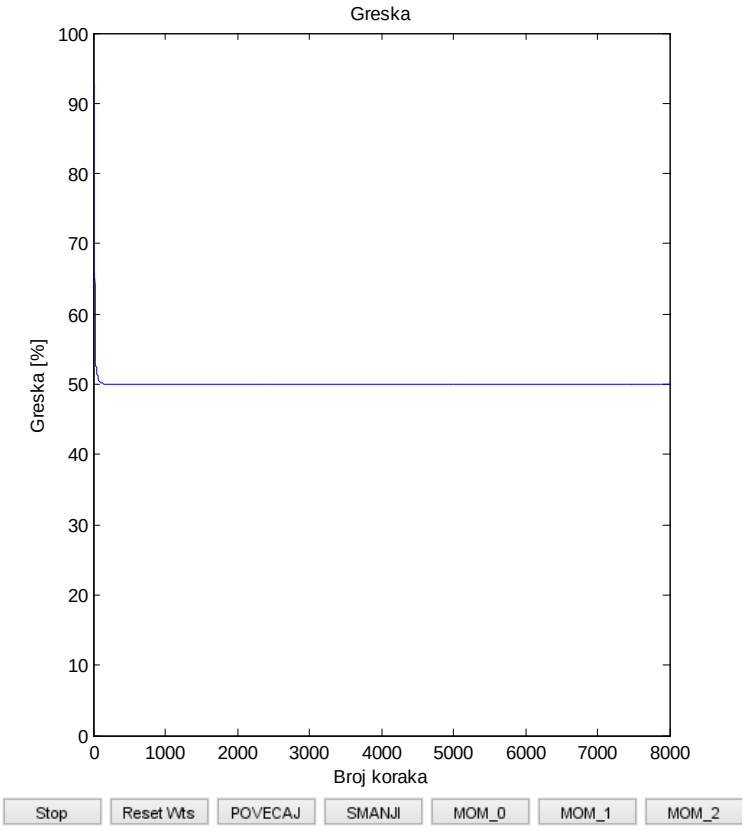
U ovom poglavlju testirana je mogućnost učenja neuronske mreže bez korištenja Bias neurona na zadanim problemima. U svim primjerima su radi usporedivosti dobivenih rezultata korištene iste početne težine i ostali parametri neuronske mreže kao oni prikazani u 4. poglavlju ovog rada.

## 6.1 XOR klasifikacijski problem - učenje bez Bias neurona

Rezultat učenja za XOR problem bez Bias neurona je prikazan na *slici 19*, iz čega je vidljivo da je neuronska mreža neuspješno završila postupak učenja.

Učenje je provedeno bez momentuma i s istim težinama i ostalim parametrima kao u *slučaju a)* poglavlja 4.1 ovog rada.

K1=3, K2=2, η=0,1, Bias=0, Momentum=0 (α=0, β=0)

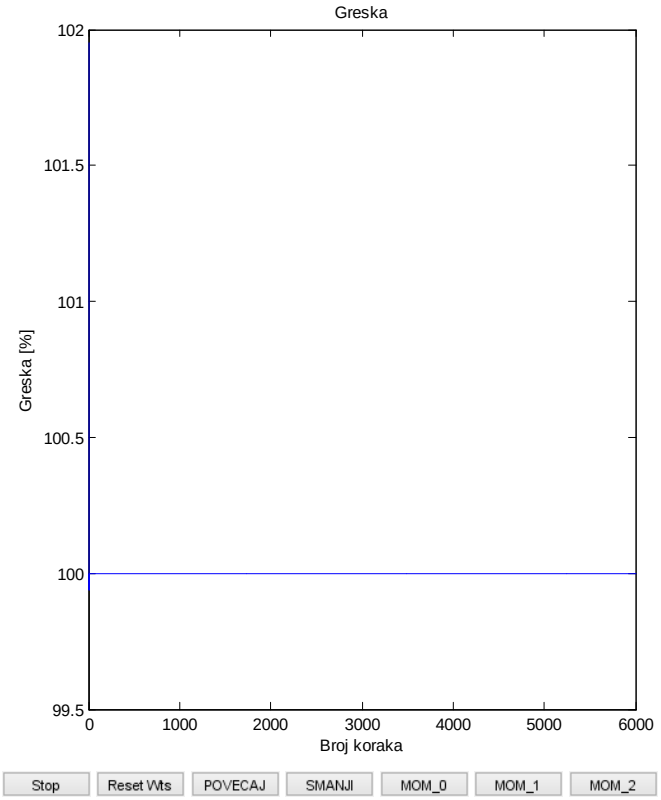



**Slika 19. XOR klasifikacijski problem-učenje bez Bias neurona**

## 6.2 Prošireni XOR klasifikacijski problem - učenje bez Bias neurona

Rezultat učenja za prošireni XOR klasifikacijski problem bez Bias neurona je prikazan *slikom 20.* Kod ovog problema, učenje bez Bias neurona s istim težinama i ostalim parametrima je također završilo neuspješno.

K1=4, K2=3, η=0,2, Bias=0, Momentum=0 (α=0, β=0)




**Slika 20. Prošireni XOR klasifikacijski problem-učenje bez Bias neurona**

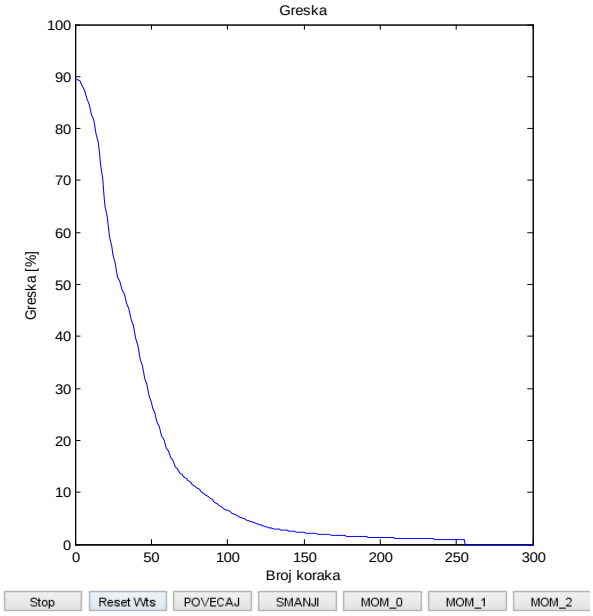
## 6.3 Učenje bez Bias neurona za regresijski problem linearne funkcije O1=3*Z12+2*Z13

Učenje bez Bias neurona za slučaj linearne funkcije O1 = 3*Z12 + 2*Z13 je završilo uspješno. Provedeno je bez momentuma, pa s momentumom prvog i drugog reda kako bi se usporedila brzina učenja. Rezultati su prikazani u *tablici 18.*

**Tablica 18. Učenje bez Bias neurona za linearnu funkciju O1=3*Z12+2*Z13**

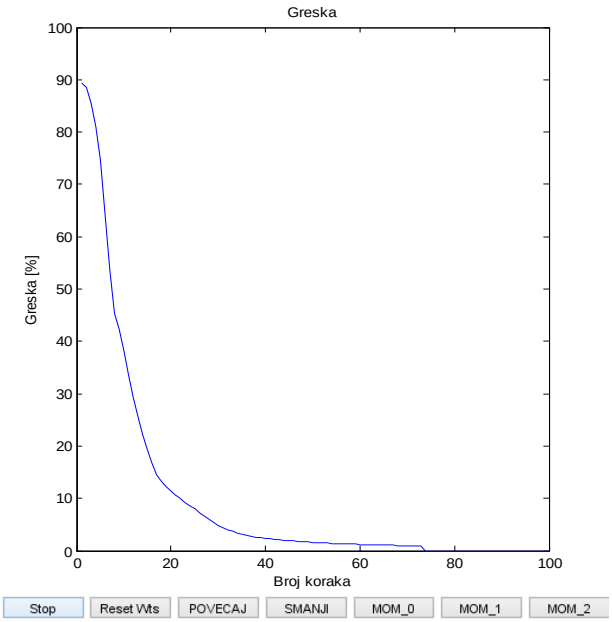
**a) K1=7, K2=5, η=0,04, Bias=0, Momentum=0 (α=0, β=0)**

Konvergencija postignuta u 256. iteraciji



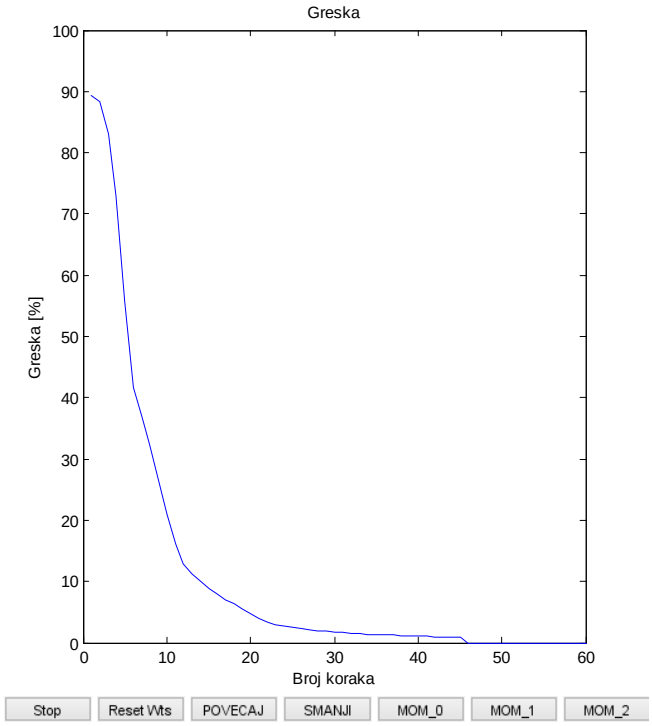

**b) K1=7, K2=5, η=0,04, Bias=0, Momentum=1 (α=0,7, β=0)**

Konvergencija postignuta u 74. iteraciji



**c) K1=7, K2=5, η=0,04, Bias=0, Momentum=2 (α=0,7, β=0,1)**

Konvergencija postignuta u 46. iteraciji

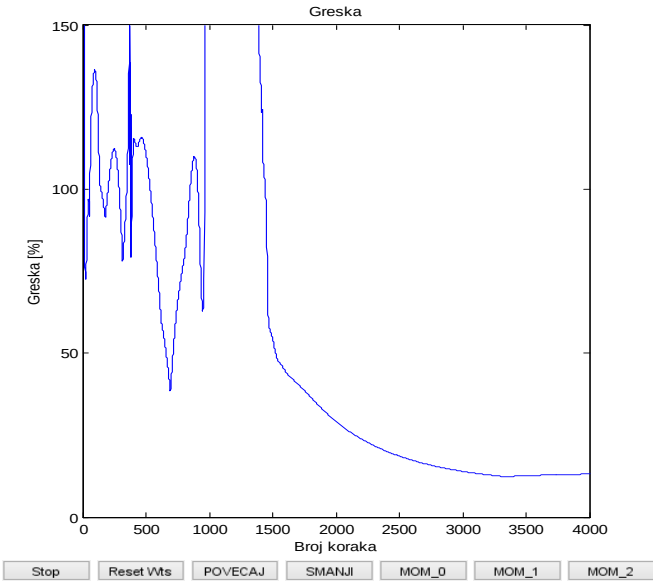


## 6.4 Učenje bez Bias neurona za regresijski problem nelinearne funkcije f(x)=sin(3*x)/x

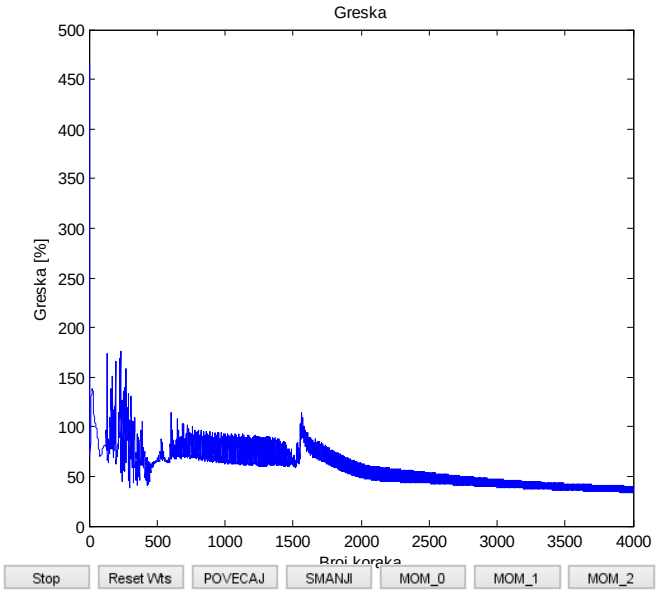
Učenje bez Bias neurona za regresijski problem nelinearne funkcije $\frac{\sin(3*x)}{x}$ nije završilo uspješno. Ni učenje s momentumom prvog reda nije dovelo do konvergencije. Rezultati su prikazani u *tablici 19.*

**Tablica 19. Učenje bez Bias neurona za problem prepoznavanja funkcije f(x)=sin(3*x)/x**

**a) K1=7, K2=5, η=0,006, Bias=0, Momentum=0 (α=0, β=0)**




**b) K1=7, K2=5, η=0,006, Bias=0, Momentum=1 (α=0,7, β=0)**



# 7 Zaključak

Iz prikazanih rezultata je vidljivo da je umjetna neuronska mreža s Gauss-ovom radijalnom baznom funkcijom u neuronima skrivenog sloja uspješno obavila postupak učenja do željene točnosti od 0.5% u svim korištenim problemima. Ključno je provoditi učenje uz povoljno odabran koeficijent brzine učenja η. Njegov iznos se za određene probleme mora promijeniti i za jedan do dva reda veličine. Osim toga su bitni iznosi slučajno generiranih početnih težina i parametara aktivacijske funkcije $c$ i $\sigma$.

Jednom pogođene povoljne težine je korisno spremiti i koristiti za daljnja testiranja. Testiranje neuronske mreže na ulazno izlaznim podacima na kojima prethodno nije učila završilo je s minimalnim odstupanjima kod svih obrađenih problema, osim kod učenja nelinearne funkcije f(x)=sin(3*x)/x. Mreža je uspjela naučiti zadane ulazno izlazne podatke sa željenom pogreškom, ali uspoređujući rezultate testa s ostalim problemima, nije potpuno naučila zadani problem.

Momentum kao metoda ubrzavanja procesa učenja, uvijek ne osigurava konvergenciju algoritma učenja na odabranim problemima, što je poznata karakterisitka ovakvog načina ubrzavanja procesa učenja. Kod učenja neuronske mreže na proširenom XOR klasifikacijskom problemu, uvođenje momentuma je onemogućilo konvergenciju jer je tijekom učenja greška konstantno iznosila 100%. Kod nelinarnog problema prepoznavanja funkcije f(x)=sin(3*x)/x, gdje su i rezultati testiranja bili najlošiji, momentumi prvog i drugog reda su dodatno produžili proces učenja uz pristutne oscilacije. Povoljni rezultati su dobiveni kod standardnog XOR problema i prepoznavanja linearne funkcije O1=3*Z12+2*Z13. Smanjili su potrebni broj iteracija za postizanje konvergencije.

Na kraju je provjereno hoće li neuronska mreža uspješno završiti postupak učenja bez korištenja Bias neurona. Rezultati pokazuju da je učenje uz Bias neuron ipak poželjno, ali ne uvijek neophodno. Kod proširenog i standardnog XOR problema se bez Bias neurona javlja greška u iznosu od 100%, a kod učenja na nelinarnom problemu su se pojavile veće oscilacije u odnosu na učenje s Bias neuronom. Pogreška se smanjuje, ali konvergencija ipak nije postignuta. Uvođenje momentuma samo dodatno pojačava nastale oscilacije. Jedino je učenje na linearnom problemu uspješno završilo bez Bias neurona i to uz manji broj iteracija. Ako se uz to još uključi momentum drugog reda, potrebno je samo 46 iteracija za postizanje konvergencije.

Istraživanje u ovom radu pokazuje da koeficijent brzine učenja η najviše utječe na kvalitetu učenja neuronske mreže. Momentumi mogu ubrzati proces učenja, ali zato uvijek ne garantiraju konvergenciju. Ako mrežu želimo naučiti jednostavnijem problemu poput standardnog XOR problema, momentumi će pozitivno utjecati na tijek učenja. No, tijekom učenja nelinarnog problema, momentumi dovode do oscilacija, što je bio slučaj i kod učenja mreže bez Bias neurona.

Bias je neuron konstantnog izlaza jednakog jedinici koji se uobičajeno dodaje svakom neuronu koji sudjeluje u procesu učenja. No, kako je prikazano u posljednjem poglavlju,

učenje se kod određenih problema može provoditi i bez navedenog neurona. Hoće li učenje bez Bias neurona završiti uspješno se uvijek može pripisati slučajno generiranim težinama u tom slučaju i odabranom iznosu koeficijenta brzine učenja η. Isto vrijedi i za ubrzavanje procesa uvođenjem momentuma koji uglavnom povoljno utječe na tijek učenja, ali u ovisnosti o složenosti zadanog problema, može dovesti do povećanja pogreške.

# 8 Literatura

[1] B. Novaković, D. Majetić, M. Široki, *Umjetne neuronske mreže*, Zagreb, 2011.

[2] S. Lončarić, *Neuronske mreže: Uvod*, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, Zagreb. [Internet] Dostupno na: http://www.fer.unizg.hr/_download/repository/01-Uvod-1s.pdf

[3] I. Petrović, N. Perić, *Inteligentno upravljanje sustavima, 2. dio: Neuronsko upravljanje*, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, Zagreb, ak. god. 2007./2008. [Internet] Dostupno na: https://www.fer.unizg.hr/_download/repository/IUS_2.dio_SKRIPTA.pdf

[4] B. Draščić Ban, T. Poganj, *Primijenjena matematika*, Sveučilište u Rijeci, Pomorski fakultet u Rijeci, Rijeka 2009. [Internet] Dostupno na: http://www.pfri.uniri.hr/~ban/primjenjena/pm.pdf

[5] D. Shiffman, *The Nature Of Code*, *Chapter 10: Neural Networks.* [Internet] Dostupno na: http://natureofcode.com/book/chapter-10-neural-networks/

[6] *Exclusive-OR Gate Tutorial*. [Internet] Dostupno na: http://www.electronics-tutorials.ws/logic/logic_7.html