KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

INTELEKTIKOS PAGRINDAI (P176B101)

Trečio laboratorinio darbo ataskaita

Atliko:

IFF-8/3 gr. studentas

Dovydas Zamas

2021 m. gegužės 23 d.

Priėmė:

Doc. Paulauskaitė-Tarasevičienė Agnė

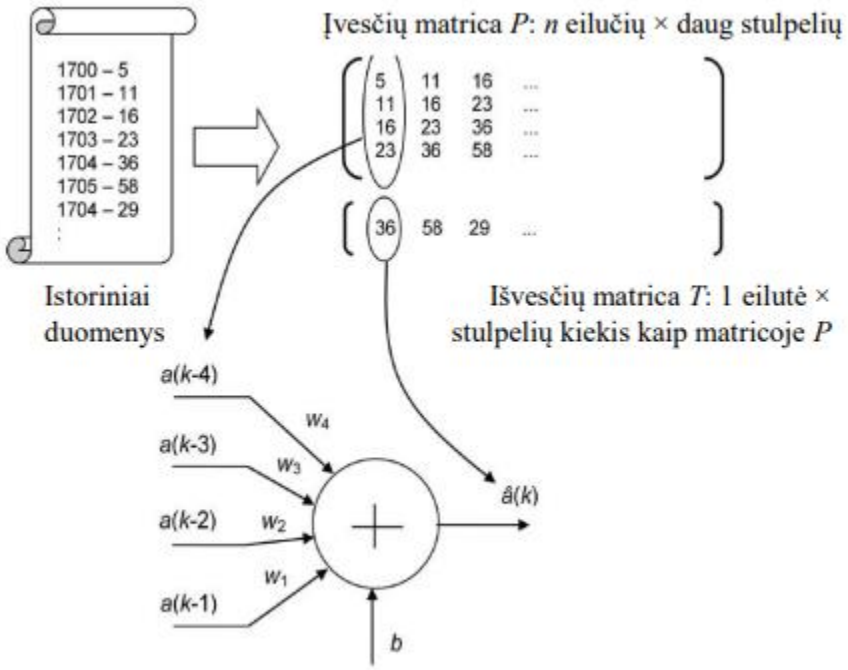
KAUNAS 2021

Turinys

1. Pirma dalis

1.1. Darbo aprašas

Darbo metu bus panaudotas paprasčiausios struktūros dirbtinis neuroninis tinklas – vienetinis neuronas su tiesine aktyvavimo funkcija (purelin(n)=purelin(Wp+b)=Wp+b). Neurono užduotimi bus laiko eilutės k-osios reikšmės a(k) prognozavimas panaudojant n ankstesnes reikšmes a(k-1), a(k-2), ..., a(k-n). Modelį, kurį realizuojame esant prielaidai, kad priklausomybė tarp prognozuojamos reikšmės ir prieš tai esančių eilės elementų gali būti aprašyta naudojant tiesinę funkciją, vadiname autoregresiniu tiesiniu modeliu n-tosios eilės.



Pav 1 Tiesinio neurono schema su n=4

1.2. Pradiniai duomenys

Pirmiausia nupiešiamas aktyvumo priklausymo nuo metų grafikas

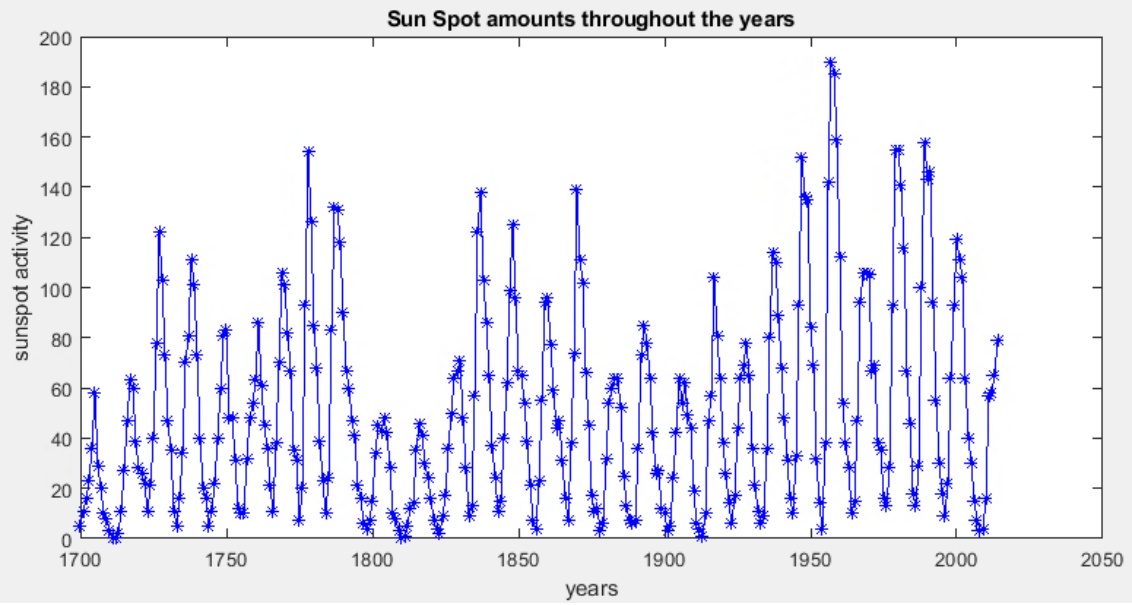


Figure 1 Saulės dėmių aktyvumo grafikas

Tada atvaizduojama 3D diagrama, parodanti, kaip aktyvumas priklauso nuo dviejų prieš tai buvusių metų aktyvumo verčių

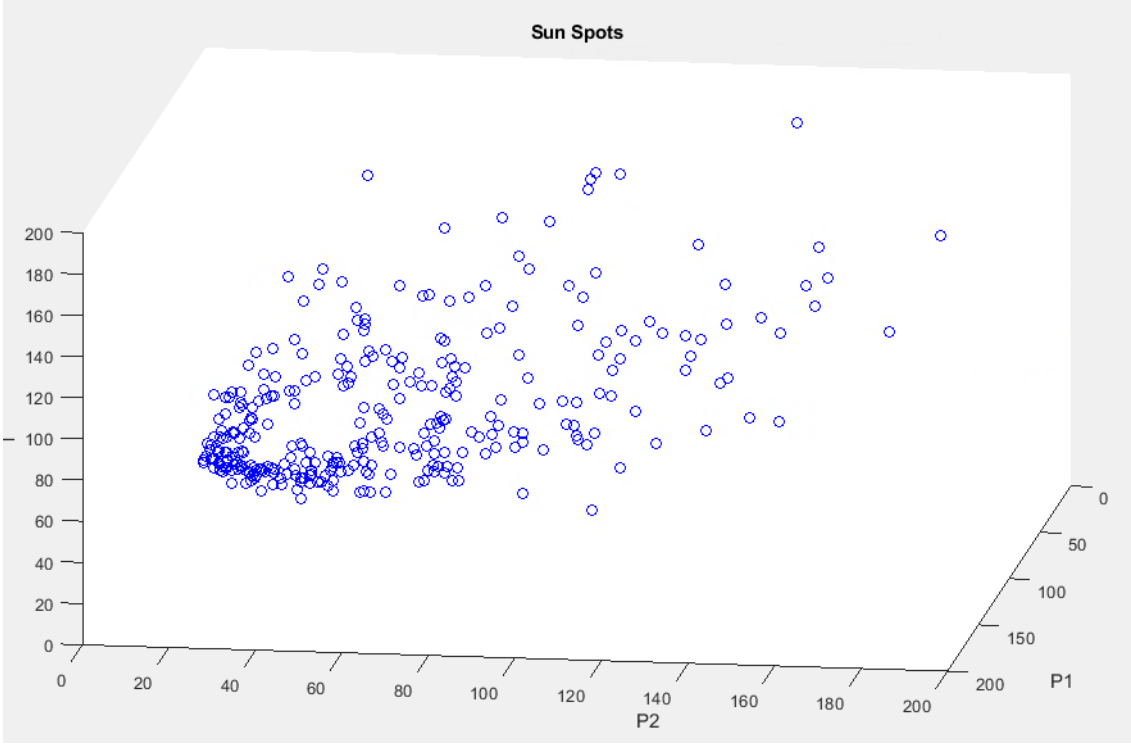


Figure 2 Saulės dėmių aktyvumo 3D grafikas

1.3. Neurono įvesties matrica ir išvesties vektorius

Sudarius P (įvesties) ir T (išvesties) matricas, patikrinamas jų dydis ir peržiūrimas turinys.

P

2x313 double

	1	2	3	4	5	6	7	8
1	5	11	16	23	36	58	29	20
2	11	16	23	36	58	29	20	10

Figure 3 Įvesties matrica

T

1x313 double

	1	2	3	4	5	6
1	16	23	36	58	29	20

Figure 4 Išvesties vektorius

1.4. Neurono sukūrimas ir apmokymas naudojant dalinį duomenų rinkinį

Pirmiausia atskiriama 200 eilučių, skirtų neurono apmokymui. Tuomet neuronas sukuriamas naudojant *newlind()*, kuri priima šiuos dalinius duomenis ir gražina neuroną, kuris buvo apmokytas naudojant duotus duomenis. Šio neurono svoriai:

```
Weight coefficients and bias:
   -0.6761    1.3715

   13.4037
```

1.5. Modelio verifikacija

Modelis verifikuojamas palyginant realias reikšmes su modelio prognozuojamomis reikšmėmis. Palyginama tiek su daliniais duomenimis, kurie buvo naudojami modelio apmokymui, tiek su pilnais duomenimis

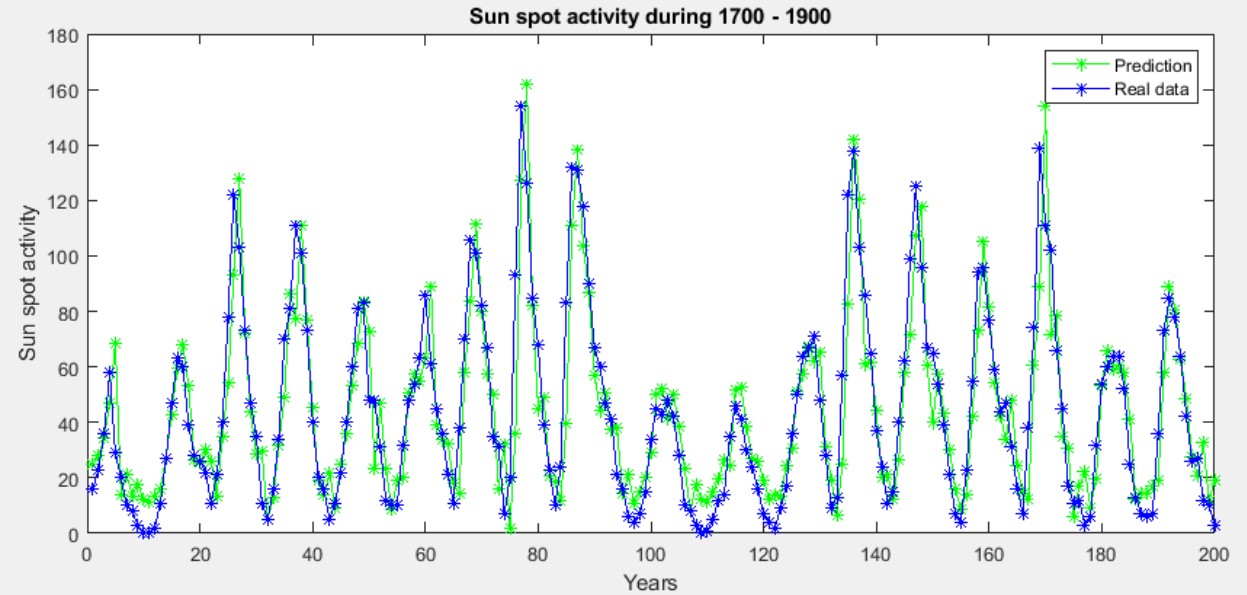


Figure 5 Prognozės ir realių duomenų palyginimas apmokymo duomenims

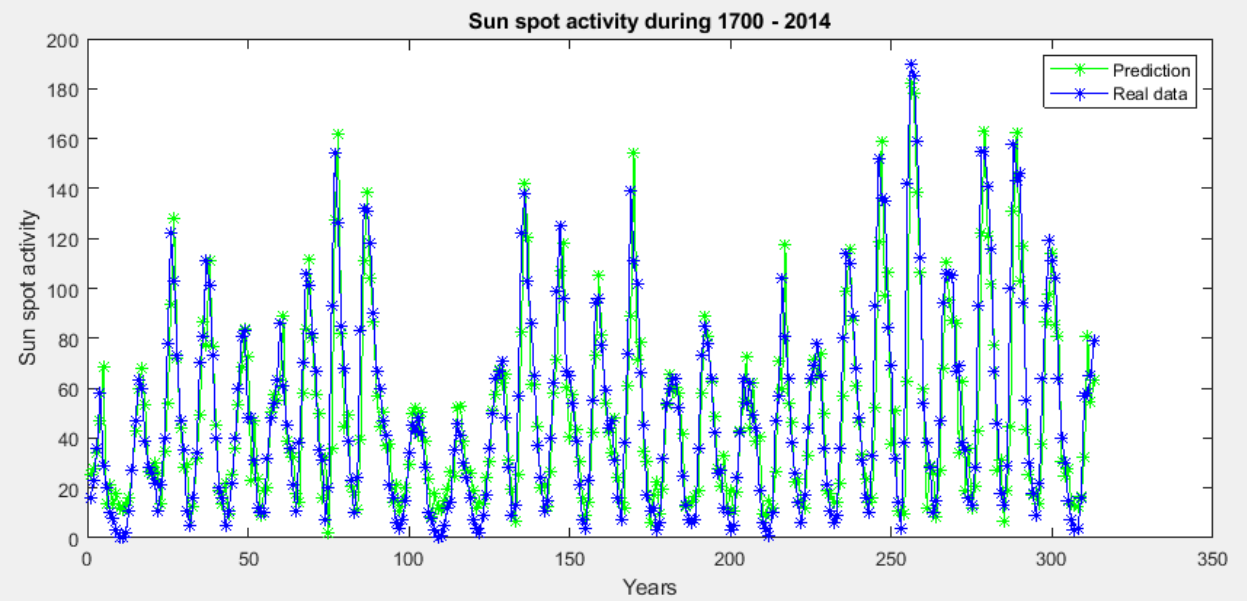


Figure 6 Prognozės ir realių duomenų palyginimas visiems duomenims

Norint patikrinti modelio prognozės tikslumą, apskaičiuojamas klaidų vektorius:

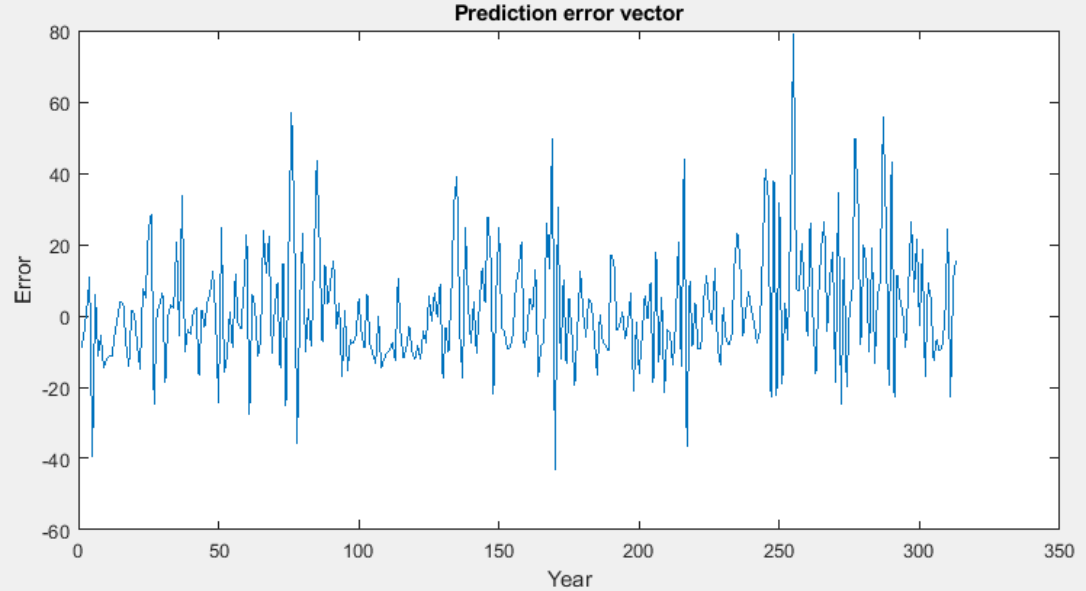


Figure 7 Klaidų vektorius

Jis taip pat atvaizduojamas histograma:

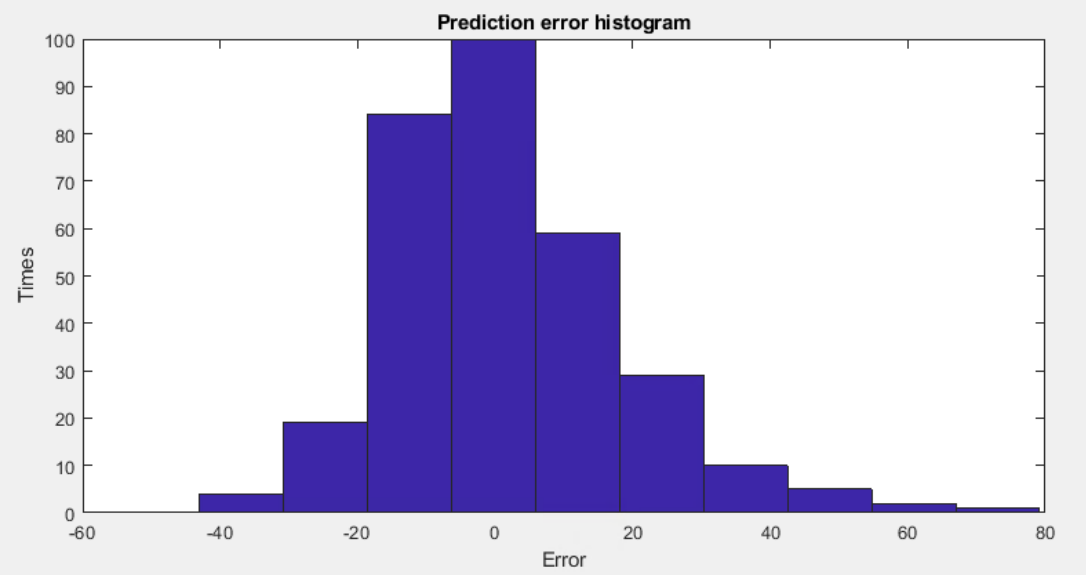


Figure 8 Klaidų vektoriaus histograma

Naudingesni klaidoms įvertinti yra MSE ir MAD įverčiai:

```
mse =          mad =

  278.2687          9.2189
```

MSE įvertis yra žymiai jautresnis nukrypimams negu MAD, ir būtent jis naudojamas apmokyti neuroną.

1.6. Rankinis neurono apmokymo parametrų pasirinkimas

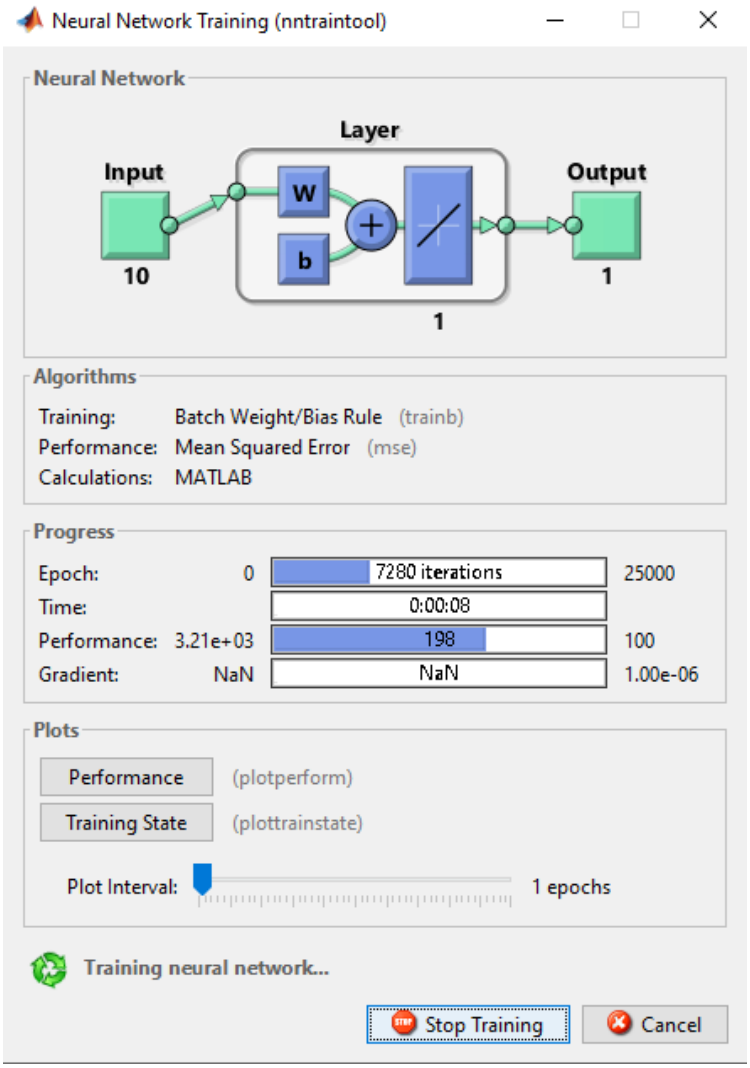
Iš naujo sukuriamas neuronas, šį kartą naudojant *newlin()* komandą. Ši komanda nevykdo automatinio apmokymo. Kuriant neuroną reikia nurodyti, kiek bus išvesčių ir koks bus apmokymo greitis.

Apmokymo greičiui apskaičiuoti naudojama funkcija *maxlinlr()*, kurį šį greitį apskaičiuoja pagal tai, kokiais duomenimis neuronas bus apmokomas.

Toliau nurodomi tinklo apmokymo limitai – apmokymas bus stabdomas, kai klaidos vektoriaus MSE pasieks nurodytą *goal* reikšmę, arba bus pasiekta *epochs* iteracijų skaičius.

Taip pat pakeičiama, kiek praeitų metų duomenų neuronas gauna kaip įvestis.

Su pakeistais parametrais neuronas apmokomas naudojant *train()* komandą. Ši komanda pateikia apmokymo langą, kuriama atvaizduojama neurono struktūra, apmokymui ir tikslumo įvertinimui naudojami algoritmai, dabartinės apmokymo reikšmės bei galimybė nubraižyti tikslumo kitimo grafiką.



Pav 2 Neurono apmokymo langas

1.7. Išvados

N	Goal	Epochs	Performance	MSE	MAD	Bias	Weights
2	100	1000	274	349.6507	9.5992	0.7731	-0.5851 1.4638
2	100	2500	265	338.812	9.1239	1.8416	-0.5928 1.456
2	100	5000	253	324.3785	8.6097	3.4252	-0.6042 1.4444
2	100	10000	237	305.2582	8.4244	5.9715	-0.6225 1.4258
6	100	1000	235	295.3619	9.3827	0.1503	0.2776 -0.2790 0.1704 -0.0251 -0.6586 1.4679
6	100	2500	234	295.5579	9.5202	0.3398	0.2808 -0.2874 0.1641 0.0036 -0.6905 1.4793
6	100	5000	233	294.3347	9.4713	0.6484	0.2774 -0.2856 0.1622 0.0038 -0.6900 1.4763
6	100	10000	231	292.0255	9.3544	1.2423	0.2712 -0.2833 0.1604 0.0021 -0.6877 1.4701
12	100	1000	195	237.7483	8.881	0.0391	-0.0440 0.1717 -0.1051 0.1436 0.0753 -0.0065 0.0239 -0.0755 0.1606 -0.1324 -0.4702 1.2420
12	100	2500	193	241.9755	8.8394	0.0892	-0.0806 0.2477 -0.1839 0.1854 0.0661 -0.0388 0.0947 -0.1501 0.1768 -0.0506 -0.5861 1.3019
12	100	5000	193	243.2175	8.6972	0.1715	-0.0889 0.2664 -0.2058 0.1983 0.0668 -0.0543 0.1164 -0.1658 0.1768 -0.0348 -0.6048 1.3106
12	100	10000	192	243.0799	8.729	0.3337	-0.0901 0.2680 -0.2077 0.1990 0.0668 -0.0562 0.1181 -0.1675 0.1766 -0.0340 -0.6059 1.3105

Lentelė 1 Įvairių parametrų keitimo resultatai apmokyme

- Visais atvejais apmokymas iš pradžių vyksta labai greitai, tačiau didėjant iteracijų skaičiui MSE mažėjimo greitis stipriai sumažėja.

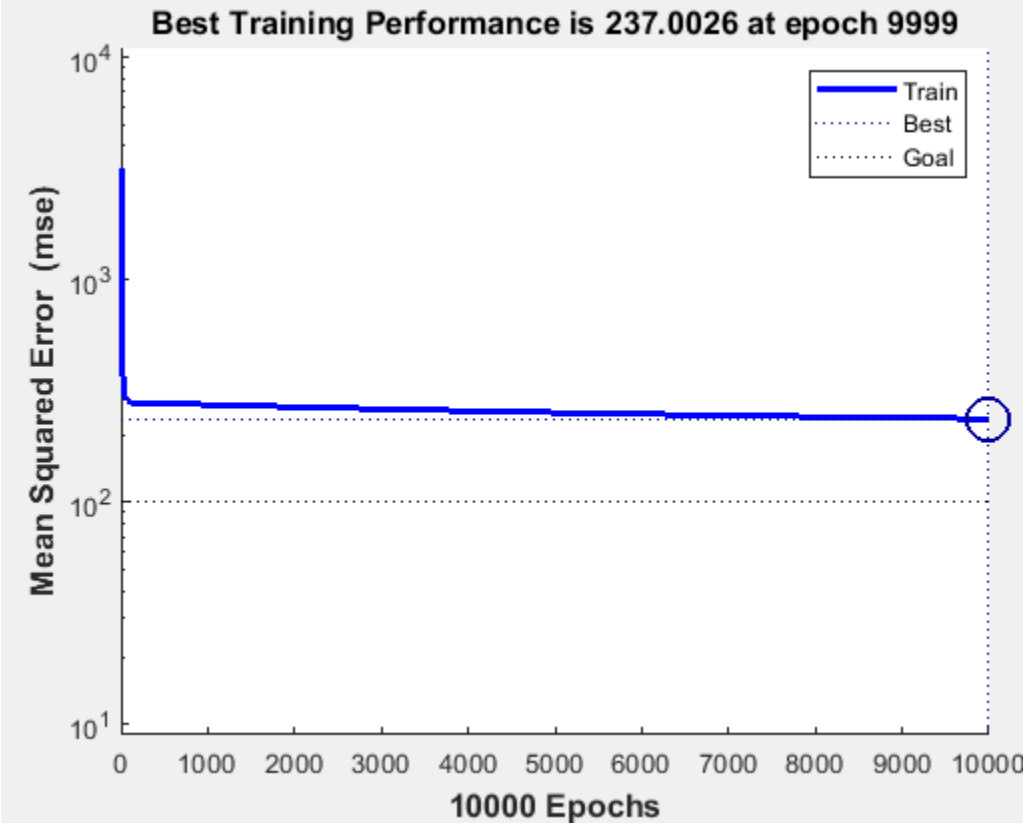


Figure 9 Apmokymo įverčio kitimas didėjant iteracijų kiekiui

- Tikslinis MSE niekada nepasiekiamas – taip yra todėl, kad neįmanoma vien iš šių duomenų tiksliai atspėti rezultato. Visada bus paklaida.
- Pateikiant 12 praeitų metų, tinklas apsimokino greitai ir jo MSE apmokinimo duomenims buvo geresnis nei pateikus 6 ar 2. Tačiau patikrinant MSE su visais duomenimis neurono rezultatas didėjant iteracijų skaičiui nekonvergavo – tinklas ‚persimokė' ant mokomųjų duomenų.

1.8. Matlab kodas

Pirma dalis:

```
%#ok<*NOPTS>

% praejusios programos isvalymas
clc
clear
close all

% duomenu uzkrovimas
load 'sunspot.txt'

% duomenu grafikas
figure(1)
plot(sunspot(:,1),sunspot(:,2),'b-*')
title('Sun Spot amounts throughout the years')
xlabel('years')
ylabel('sunspot activity')

% matricu sukurimas
L = length(sunspot);              % data size
P = [   sunspot(1:L-2,2)' ;       % input data
        sunspot(2:L-1,2)'];
T = sunspot(3:L,2)';              % output data

% trimate diagrama
figure(2)
plot3(P(1,:),P(2,:),T,'bo')
title('Sun Spots')
xlabel('P1')
ylabel('P2')
zlabel('T')

% isskirti apmokymo duomenu matricas
Pu = P(:,1:200);
Tu = T(:,1:200);

disp('Pu array size:')
disp(size(Pu))
disp('Tu array size:')
disp(size(Tu))

% sukurti neuronu tinkla
net = newlind(Pu,Tu);

% pavaizduojami neurono koefficientai
disp('Weight coefficients and bias:' )
disp( net.IW{1} )
disp( net.b{1} )

% modelio verifikacija
```

```
Tsu = sim(net,Pu);

% modelio rezultato ir tikru duomenu palyginimas 200  stulpeliu
figure(3)
plot(Tsu,'g-*')
hold on
plot(Tu,'b-*')

xlabel('Years');
ylabel('Sun spot activity');
title('Sun spot activity during 1700 - 1900');
legend('Prediction','Real data')

% modelio rezultato ir tikru duomenu palyginimas visiems stulpeliams
Ts = sim(net,P);

figure(4)
plot(Ts,'g-*')
hold on
plot(T,'b-*')

xlabel('Years');
ylabel('Sun spot activity');
title('Sun spot activity during 1700 - 2014');
legend('Prediction','Real data')

% klaidu apskaiciavimas
E = T-Ts;

% klaidu grafikas
figure(5);
plot(E);
title('Prediction error vector');
xlabel('Year');
ylabel('Error');

% klaidu histogramos
figure(6);
hist(E);
title('Prediction error histogram');
xlabel('Error');
ylabel('Times');

% klaidos apskaiciavimas
mse = mse(E)
mad = median(abs(E))
```

Antra dalis:

```
%#ok<*NOPTS>

% praejusios programos isvalymas
clc
clear
close all

% keiciamos reiksmes
inputCount = 2
trainingDataCount = 200
trainingGoal = 100
trainingEpochs = 10000

% duomenu uzkrovimas
load 'sunspot.txt'

% duomenu grafikas
figure(1)
plot(sunspot(:,1),sunspot(:,2),'b-*')
title('Sun Spot amounts throughout the years')
xlabel('years')
ylabel('sunspot activity')

% matricu sukurimas
L = length(sunspot);              % data size
N = inputCount;                   % input count
U = trainingDataCount;            % training dataset size
P=[];
for n = 1:N                       % generate input matrix
    n2 = N-n+1;
    P = [P; sunspot(n:L-n2,2)'];
end
T = sunspot(N+1:L,2)';              % output data

% trimate diagrama
figure(2)
plot3(P(1,:),P(2,:),T,'bo')
title('Sun Spots')
xlabel('P1')
ylabel('P2')
zlabel('T')

% isskirti apmokymo duomenu matricas
Pu = P(:,1:U);
Tu = T(:,1:U);

disp('Pu array size:')
disp(size(Pu))
disp('Tu array size:')
disp(size(Tu))
```

```
% sukurti neuronu tinkla
S = 1
net = newlin(Pu,S,0, maxlinlr(Pu, 'bias'));

% isdresuoti neuronu tinkla
net.trainParam.goal = trainingGoal;          % goal std
net.trainParam.epochs = trainingEpochs;      % training steps
net = train(net, Pu, Tu);                    % train the network

% pavaizduojami neurono koefficientai
disp('Weight coefs:' )
disp( net.IW{1} )
disp( net.b{1} )

% modelio verifikacija
Tsu = sim(net,Pu);

% modelio rezultato ir tikru duomenu palyginimas 200  stulpeliu
figure(3)
plot(Tsu,'g-*')
hold on
plot(Tu,'b-*')

xlabel('Years');
ylabel('Sun spot activity');
title(['Sun spot activity during 1700 - ' num2str(1700 + U)]);
legend('Prediction','Real data')

% modelio rezultato ir tikru duomenu palyginimas visiems stulpeliams
Ts = sim(net,P);

figure(4)
plot(Ts,'g-*')
hold on
plot(T,'b-*')

xlabel('Years');
ylabel('Sun spot activity');
title('Sun spot activity during 1700 - 2014');
legend('Prediction','Real data')

% klaidu apskaiciavimas
E = T-Ts;

% klaidu grafikas
figure(5);
plot(E);
title('Prediction error vector');
xlabel('Year');
ylabel('Error');

% klaidu histogramos
figure(6);
```

```
hist(E);
title('Prediction error histogram');
xlabel('Error');
ylabel('Times');

% klaidos apskaiciavimas
mse = mse(E)
mad = median(abs(E))
```