Aufgabe: Trainieren Sie ein zweistufiges neuronales Netz ohne verdeckte Schicht mit drei Eingabe- und drei Ausgabeneuronen durch Veränderung der Gewichte so, daß jedes der 3 Ausgangsneuronen durch sein spezifisches Trainingsmuster aktiviert wird. Ausgangsneuron 1 werde aktiviert durch das Trainingsmuster $(1, 0, 1)$, Ausgangsneuron 2 durch das Trainingsmuster $(1, 1, 1)$ und Ausgangsneuron 3 durch das Trainingsmuster $(0, 1, 0)$. Verwenden Sie eine sigmoide Aktivierungsfunktion.

Lösung: Seien die x_{pi} die Elemente der Input-Schicht für das Muster p, die y_{pi} entsprechend die Elemente der Output-Schicht und die w_{ij} die Gewichte der Verbindungsmatrix. Dann sind Input- und Output-Neuronen durch folgende Matrixabbildung verknüpft:

$$\begin{pmatrix} y_{p1} \\ y_{p2} \\ y_{p3} \end{pmatrix} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix} \begin{pmatrix} x_{p1} \\ x_{p2} \\ x_{p3} \end{pmatrix}.$$

In Komponentenschreibweise lautet das äquivalente lineare Gleichungssystem:

$$\begin{aligned} y_{p1} &= w_{11}x_{p1} + w_{12}x_{p2} + w_{13}x_{p3}, \\ y_{p2} &= w_{21}x_{p1} + w_{22}x_{p2} + w_{23}x_{p3}, \\ y_{p3} &= w_{31}x_{p1} + w_{32}x_{p2} + w_{33}x_{p3}. \end{aligned}$$

Wir können zur Lösung der Aufgabenstellung von einer beliebigen Verbindungsmatrix ausgehen und definieren

$$\begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix} \equiv \begin{pmatrix} 0{,}5 & -0{,}5 & 0{,}5 \\ 0{,}3 & 0{,}3 & 0{,}3 \\ -0{,}3 & 1{,}0 & -0{,}3 \end{pmatrix}.$$

Setzen wir nun als Startwerte die obigen Koeffizienten in das lineare Gleichungssystem ein, erhalten wir

$$\begin{aligned} y_{p1} &= 0{,}5x_{p1} - 0{,}5x_{p2} + 0{,}5x_{p3}, \\ y_{p2} &= 0{,}3x_{p1} + 0{,}3x_{p2} + 0{,}3x_{p3}, \\ y_{p3} &= -0{,}3x_{p1} + 1{,}0x_{p2} - 0{,}3x_{p3}. \end{aligned}$$

Für das Muster $p = 1$ ergibt sich damit der Ausgangsvektor

$$\begin{pmatrix} x_{11} \\ x_{12} \\ x_{13} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} y_{11} \\ y_{12} \\ y_{13} \end{pmatrix} = \begin{pmatrix} 0{,}5 + 0 + 0{,}5 \\ 0{,}3 + 0 + 0{,}3 \\ -0{,}3 + 0 - 0{,}3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0{,}6 \\ -0{,}6 \end{pmatrix}.$$

Mit der Sigmoidfunktion f als Aktivierungsfunktion,

$$f(y_{pi}) = \frac{1}{1+e^{-y_{pi}}},$$

im offenen Intervall $]0,1[$ erhalten wir mit dem Netzinput die Startwerte

$$\big(f(y_{11}),\, f(y_{12}),\, f(y_{13})\big) = (0.731,\, 0.646,\, 0.354).$$

Muster $p=2$ liefert den Netzinput

$$\begin{pmatrix} x_{21} \\ x_{22} \\ x_{23} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} y_{21} \\ y_{22} \\ y_{23} \end{pmatrix} = \begin{pmatrix} 0{,}5-0{,}5+0{,}5 \\ 0{,}3+0{,}3+0{,}3 \\ -0{,}3+1{,}0-0{,}3 \end{pmatrix} = \begin{pmatrix} 0{,}5 \\ 0{,}9 \\ 0{,}4 \end{pmatrix}$$

und nach Aktivierung den Startvektor

$$\big(f(y_{21}),\, f(y_{22}),\, f(y_{23})\big) = (0.622,\, 0.711,\, 0.599).$$

Für das Muster $p=3$ erhalten wir schließlich den Netzinput

$$\begin{pmatrix} x_{31} \\ x_{32} \\ x_{33} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} y_{31} \\ y_{32} \\ y_{33} \end{pmatrix} = \begin{pmatrix} 0-0{,}5+0 \\ 0+0{,}3+0 \\ 0+1{,}0+0 \end{pmatrix} = \begin{pmatrix} -0{,}5 \\ 0{,}3 \\ 1{,}0 \end{pmatrix}$$

und daraus den Startvektor

$$\big(f(y_{31}),\, f(y_{32}),\, f(y_{33})\big) = (0.378,\, 0.574,\, 0.731).$$

Der Gesamtfehler E des Netzes ergibt sich als Summe über sämtliche 3 Trainingsmuster:

$$E = \frac{1}{2}\sum_{p=1}^{3}\sum_{i=1}^{3}\left(\eta_{pi} - f\left(\sum_{j=1}^{3} w_{ij}x_{pj}\right)\right)^2.$$

Ein schrittweises Verändern der Verbindungsgewichte führt nach k Schritten zu einer signifikanten Verringerung des Gesamtfehlers. Zweck des Verfahrens ist, den Fehler iterativ zu minimieren, d.h.

$$E_k = \frac{1}{2}\sum_{p=1}^{3}\sum_{i=1}^{3}\left(\eta_{pi} - f\left(\sum_{j=1}^{3} w_{ij,k}x_{pj}\right)\right)^2 \to \text{Minimum.}$$

Die Festlegung einer oberen Grenze für den zulässigen Fehler kann zugleich als Abbruchkriterium verwendet werden. Das Verfahren startet mit den Ausgangsgewichten $w_{ij,0} = w_{ij}$, wobei mit jeder Näherung $k = 1, 2, 3, \ldots$ entsprechend

$$w_{ij,k+1} = w_{ij,k} + \Delta w_{ij,k}$$

ein Korrekturfaktor

$$\Delta w_{ij,k} = -\lambda \frac{\partial E}{\partial w_{ij,k}} = \lambda \sum_{p=1}^{3} \left(\eta_{pi} - f(y_{pi,k})\right) f(y_{pi,k}) \left(1 - f(y_{pi,k})\right) x_{pj}$$

addiert werden muß. Darin ist λ der sogenannte Lernparameter. Die Gewichtsveränderung beim Start ist $\Delta w_{ij,0} = \Delta w_{ij}$, wobei

$$\Delta w_{ij} = -\lambda \frac{\partial E}{\partial w_{ij}} = \lambda \sum_{p=1}^{3} \left(\eta_{pi} - f(y_{pi,})\right) f(y_{pi}) \left(1 - f(y_{pi})\right) x_{pj}.$$

Die Soll-Ausgangswerte müssen zu Beginn der Iteration einmalig festgelegt werden. Wir wählen dazu in Annäherung an die Aktivierungswerte folgende Sollgrößen:

$$\begin{aligned}
(\eta_{11}, \eta_{12}, \eta_{13}) &= (0.8, 0.6, 0.4) \\
(\eta_{21}, \eta_{22}, \eta_{23}) &= (0.6, 0.8, 0.6) \\
(\eta_{31}, \eta_{32}, \eta_{33}) &\equiv (0.4, 0.6, 0.8)
\end{aligned}$$

Die anfängliche Änderung der Verbindungsgewichte kann ebenfalls sofort berechnet werden.

$$\begin{aligned}
\Delta w_{ij,0} = {} & \lambda\left(\eta_{1i} - f(y_{1i})\right) f(y_{1i}) \left(1 - f(y_{1i})\right) x_{1j} \\
& + \lambda\left(\eta_{2i} - f(y_{2i})\right) f(y_{2i}) \left(1 - f(y_{2i})\right) x_{2j} \\
& + \lambda\left(\eta_{3i} - f(y_{3i})\right) f(y_{3i}) \left(1 - f(y_{3i})\right) x_{3j}
\end{aligned}$$

Damit bestimmen wir das Gewicht des nächsten Iterationsschritts $w_{ij,1} = w_{ij,0} + \Delta w_{ij,0}$ und aus diesem wiederum $w_{ij,2} = w_{ij,1} + \Delta w_{ij,1}$ usw. Allgemein können wir die Änderungen der Verbindungsgewichte gemäß folgender Beziehung berechnen:

$$\begin{aligned}
\Delta w_{ij,k} = {} & \lambda\left(\eta_{1i} - f(y_{1i,k})\right) f(y_{1i,k}) \left(1 - f(y_{1i,k})\right) x_{1j} \\
& + \lambda\left(\eta_{2i} - f(y_{2i,k})\right) f(y_{2i,k}) \left(1 - f(y_{2i,k})\right) x_{2j} \\
& + \lambda\left(\eta_{3i} - f(y_{3i,k})\right) f(y_{3i,k}) \left(1 - f(y_{3i,k})\right) x_{3j},
\end{aligned}$$

wobei sich die 9 iterativen Komponenten

$$y_{pi,k} = \sum_{j=1}^{3} w_{ij,k} x_{pj}$$

für die 3 Muster wie folgt darstellen:

$$\begin{aligned}
y_{11,k} &= w_{11,k} x_{11} + w_{12,k} x_{12} + w_{13,k} x_{13}, \\
y_{21,k} &= w_{11,k} x_{21} + w_{12,k} x_{22} + w_{13,k} x_{23}, \\
y_{31,k} &= w_{11,k} x_{31} + w_{12,k} x_{32} + w_{13,k} x_{33},
\end{aligned}$$

$$
\begin{aligned}
y_{12,k} &= w_{21,k}x_{11} + w_{22,k}x_{12} + w_{23,k}x_{13},\\
y_{22,k} &= w_{21,k}x_{21} + w_{22,k}x_{22} + w_{23,k}x_{23},\\
y_{32,k} &= w_{21,k}x_{31} + w_{22,k}x_{32} + w_{23,k}x_{33},
\end{aligned}
$$

$$
\begin{aligned}
y_{13,k} &= w_{31,k}x_{11} + w_{32,k}x_{12} + w_{33,k}x_{13},\\
y_{23,k} &= w_{31,k}x_{21} + w_{32,k}x_{22} + w_{33,k}x_{23},\\
y_{33,k} &= w_{31,k}x_{31} + w_{32,k}x_{32} + w_{33,k}x_{33}.
\end{aligned}
$$

Wenden wir auf alle diese Größen die Sigmoidfunktion an, lautet der *k*te Gesamtfehler

$$
\begin{aligned}
E_k &= \frac{1}{2}\sum_{i=1}^{3}\left[\left(\eta_{1i} - f(y_{1i,k})\right)^2 + \left(\eta_{2i} - f(y_{2i,k})\right)^2 + \left(\eta_{3i} - f(y_{3i,k})\right)^2\right]\\
&= \frac{1}{2}\left[\left(\eta_{11} - f(y_{11,k})\right)^2 + \left(\eta_{21} - f(y_{21,k})\right)^2 + \left(\eta_{31} - f(y_{31,k})\right)^2\right]\\
&\quad + \frac{1}{2}\left[\left(\eta_{12} - f(y_{12,k})\right)^2 + \left(\eta_{22} - f(y_{22,k})\right)^2 + \left(\eta_{32} - f(y_{32,k})\right)^2\right]\\
&\quad + \frac{1}{2}\left[\left(\eta_{13} - f(y_{13,k})\right)^2 + \left(\eta_{23} - f(y_{23,k})\right)^2 + \left(\eta_{33} - f(y_{33,k})\right)^2\right].
\end{aligned}
$$

Das beigefügte MATLAB-Programm zeigt sehr anschaulich, wovon die Güte neuronaler Netze abhängt: weniger von der Zahl der Iterationen und der Trainigsmuster als von der Wahl eines geeigneten Startvektors.

Anlage

```
% Fehlerberechnung für ein zweistufiges neuronales Netz ohne verdeckte
% Schicht mit je 3 Ein- und Ausgabeneuronen und einer Sigmoidkurve als
% Aktivierungsfunktion
clear all

% Der Lernparameter sei lambda
lambda = 5;

% p ist die Laufvariable des jeweiligen Trainingsmusters
% pmax ist die maximale Zahl vorliegender Trainingsmuster

% Eingangs-, Soll-Ausgangs- und Ist-Ausgangsvektor hängen jeweils von der
% Laufvariablen p des Musters ab.

% Der Soll-Ausgangsvektor eta(p,i) hat imax Komponenten, wobei i die
% Zeilennummer der Gewichtsmatrix angibt

% Die 3 Komponenten des Soll-Ausgangsvektors für die 3 Muster sind nachfol-
gend gegeben

% Sollausgabewerte für Trainingslauf 1
eta11=0.731; eta12=0.646; eta13=0.354;
eta21=0.622; eta22=0.711; eta23=0.599;
eta31=0.378; eta32=0.574; eta33=0.731;

% Sollausgabewerte für Trainingslauf 2

% eta11=0.8; eta12=0.6; eta13=0.4;
% eta21=0.6; eta22=0.8; eta23=0.6;
% eta31=0.4; eta32=0.6; eta33=0.8;

% Sollausgabewerte für Trainingslauf 3

% eta11=1; eta12=0; eta13=0;
% eta21=0; eta22=1; eta23=0;
% eta31=0; eta32=0; eta33=1;

% Die 3 Komponenten der 3 Trainingsmuster x(p,j) des Eingangsvektors sind
gegeben durch
x11 = 1; x12 = 0; x13 = 1;
x21 = 1; x22 = 1; x23 = 1;
x31 = 0; x32 = 1; x33 = 0;

% Die Laufvariable der kmax Iterationsschritte sei k
kmax = 5;

% Die imax mal jmax Startgewichte w(i,j,1)seien gegeben durch
w11(1)=0.5;  w12(1)=-0.5; w13(1)=0.5;
w21(1)=0.3;  w22(1)=0.3;  w23(1)=0.3;
w31(1)=-0.3; w32(1)=1.0;  w33(1)=-0.3;

% Für Trainingslauf 3 verwenden wir als Startgewichte eine Matrix
% aus lauter Einsen:
% w11(1)=1; w12(1)=1; w13(1)=1;
% w21(1)=1; w22(1)=1; w23(1)=1;
% w31(1)=1; w32(1)=1; w33(1)=1;
```



```
for k=1:kmax
    % Erste Zeile
    y11(k) = w11(k)*x11 + w12(k)*x12 + w13(k)*x13;
    y21(k) = w11(k)*x21 + w12(k)*x22 + w13(k)*x23;
    y31(k) = w11(k)*x31 + w12(k)*x32 + w13(k)*x33;
    f(1,1,k)=1/(1+exp(-y11(k)));
    f(2,1,k)=1/(1+exp(-y21(k)));
    f(3,1,k)=1/(1+exp(-y31(k)));
    Deltaw11(k) = lambda*((eta11-f(1,1,k))*f(1,1,k)*(1-
f(1,1,k))*x11+(eta21-f(2,1,k))*f(2,1,k)*(1-f(2,1,k))*x21+(eta31-
f(3,1,k))*f(3,1,k)*(1-f(3,1,k))*x31);
    w11(k+1) = w11(k)+Deltaw11(k);
    Deltaw12(k) = lambda*((eta11-f(1,1,k))*f(1,1,k)*(1-
f(1,1,k))*x12+(eta21-f(2,1,k))*f(2,1,k)*(1-f(2,1,k))*x22+(eta31-
f(3,1,k))*f(3,1,k)*(1-f(3,1,k))*x32);
    w12(k+1) = w12(k)+Deltaw12(k);
    Deltaw13(k) = lambda*((eta11-f(1,1,k))*f(1,1,k)*(1-
f(1,1,k))*x13+(eta21-f(2,1,k))*f(2,1,k)*(1-f(2,1,k))*x23+(eta31-
f(3,1,k))*f(3,1,k)*(1-f(3,1,k))*x33);
    w13(k+1) = w13(k)+Deltaw13(k);
    % Zweite Zeile
    y12(k) = w21(k)*x11 + w22(k)*x12 + w23(k)*x13;
    y22(k) = w21(k)*x21 + w22(k)*x22 + w23(k)*x23;
    y32(k) = w21(k)*x31 + w22(k)*x32 + w23(k)*x33;
    f(1,2,k)=1/(1+exp(-y12(k)));
    f(2,2,k)=1/(1+exp(-y22(k)));
    f(3,2,k)=1/(1+exp(-y32(k)));
    Deltaw21(k) = lambda*((eta12-f(1,2,k))*f(1,2,k)*(1-
f(1,2,k))*x11+(eta22-f(2,2,k))*f(2,2,k)*(1-f(2,2,k))*x21+(eta32-
f(3,2,k))*f(3,2,k)*(1-f(3,2,k))*x31);
    w21(k+1) = w21(k)+Deltaw21(k);
    Deltaw22(k) = lambda*((eta12-f(1,2,k))*f(1,2,k)*(1-
f(1,2,k))*x12+(eta22-f(2,2,k))*f(2,2,k)*(1-f(2,2,k))*x22+(eta32-
f(3,2,k))*f(3,2,k)*(1-f(3,2,k))*x32);
    w22(k+1) = w22(k)+Deltaw22(k);
    Deltaw23(k) = lambda*((eta12-f(1,2,k))*f(1,2,k)*(1-
f(1,2,k))*x13+(eta22-f(2,2,k))*f(2,2,k)*(1-f(2,2,k))*x23+(eta32-
f(3,2,k))*f(3,2,k)*(1-f(3,2,k))*x33);
    w23(k+1) = w23(k)+Deltaw23(k);
    % Dritte Zeile
    y13(k) = w31(k)*x11 + w32(k)*x12 + w33(k)*x13;
    y23(k) = w31(k)*x21 + w32(k)*x22 + w33(k)*x23;
    y33(k) = w31(k)*x31 + w32(k)*x32 + w33(k)*x33;
    f(1,3,k)=1/(1+exp(-y13(k)));
    f(2,3,k)=1/(1+exp(-y23(k)));
    f(3,3,k)=1/(1+exp(-y33(k)));
    Deltaw31(k) = lambda*((eta13-f(1,3,k))*f(1,3,k)*(1-
f(1,3,k))*x11+(eta23-f(2,3,k))*f(2,3,k)*(1-f(2,3,k))*x21+(eta33-
f(3,3,k))*f(3,3,k)*(1-f(3,3,k))*x31);
    w31(k+1) = w31(k)+Deltaw31(k);
    Deltaw32(k) = lambda*((eta13-f(1,3,k))*f(1,3,k)*(1-
f(1,3,k))*x12+(eta23-f(2,3,k))*f(2,3,k)*(1-f(2,3,k))*x22+(eta33-
f(3,3,k))*f(3,3,k)*(1-f(3,3,k))*x32);
    w32(k+1) = w32(k)+Deltaw32(k);
    Deltaw33(k) = lambda*((eta13-f(1,3,k))*f(1,3,k)*(1-
f(1,3,k))*x13+(eta23-f(2,3,k))*f(2,3,k)*(1-f(2,3,k))*x23+(eta33-
f(3,3,k))*f(3,3,k)*(1-f(3,3,k))*x33);
    w33(k+1) = w33(k)+Deltaw33(k);
    % Fehlerberechnung
```

```
    E(k) = 0.5*((eta11-f(1,1,k))^2+(eta21-f(2,1,k))^2+(eta31-
f(3,1,k))^2+(eta12-f(1,2,k))^2+(eta22-f(2,2,k))^2+(eta32-
f(3,2,k))^2+(eta13-f(1,3,k))^2+(eta23-f(2,3,k))^2+(eta33-f(3,3,k))^2);
end

disp(['w11 = ' num2str(w11)])
disp(['w12 = ' num2str(w12)])
disp(['w13 = ' num2str(w13)])
disp(['w21 = ' num2str(w21)])
disp(['w22 = ' num2str(w22)])
disp(['w23 = ' num2str(w23)])
disp(['w31 = ' num2str(w31)])
disp(['w32 = ' num2str(w32)])
disp(['w33 = ' num2str(w33)])
disp(' ');
y11 = f(1,1,kmax);
y12 = f(1,2,kmax);
y13 = f(1,3,kmax);
y21 = f(2,1,kmax);
y22 = f(2,2,kmax);
y23 = f(2,3,kmax);
y31 = f(3,1,kmax);
y32 = f(3,2,kmax);
y33 = f(3,3,kmax);
disp(' ');
disp(['f(y11) = ' num2str(y11)])
disp(['f(y12) = ' num2str(y12)])
disp(['f(y13) = ' num2str(y13)])
disp(['f(y21) = ' num2str(y21)])
disp(['f(y22) = ' num2str(y22)])
disp(['f(y23) = ' num2str(y23)])
disp(['f(y31) = ' num2str(y31)])
disp(['f(y32) = ' num2str(y32)])
disp(['f(y33) = ' num2str(y33)])
disp(' ');
disp(['E = ' num2str(E)])
```

<u>Trainingslauf 1</u>

Wir sehen hier, daß sich die Gewichte auch nach 5 Iterationen kaum geändert haben.

```
w11 = 0.5       0.4994     0.49937     0.49927     0.49922     0.49918
w12 = -0.5         -0.5    -0.49967     -0.4995    -0.49938    -0.49929
w13 = 0.5       0.4994     0.49937     0.49927     0.49922     0.49918
w21 = 0.3      0.30045     0.30057     0.30067     0.30073     0.30077
w22 = 0.3      0.29951     0.29908     0.29882     0.29865     0.29854
w23 = 0.3      0.30045     0.30057     0.30067     0.30073     0.30077
w31 = -0.3    -0.30002    -0.30011    -0.30015    -0.30019    -0.30021
w32 = 1       1.0003      1.0005      1.0006      1.0007      1.0008
w33 = -0.3    -0.30002    -0.30011    -0.30015    -0.30019    -0.30021
```

Der Vergleich zwischen Soll- und Ist-Ausgabeneuronen weist im Rahmen der Meßgenauigkeit kaum Abweichungen auf.

```
eta11=0.731;      f(y11) = 0.73075
eta12=0.646;      f(y12) = 0.64599
eta13=0.354;      f(y13) = 0.35426
```



```
eta21=0.622;      f(y21) = 0.62224
eta22=0.711;      f(y22) = 0.71097
eta23=0.599;      f(y23) = 0.59877
eta31=0.378;      f(y31) = 0.37769
eta32=0.574;      f(y32) = 0.57411
eta33=0.731;      f(y33) = 0.7312
```

Die Fehlergenauigkeit liegt unterhalb der dritten Dezimale.

```
E = 4.8051e-07  2.9299e-07  2.3244e-07  2.0623e-07  1.9452e-07
```

Trainingslauf 2

Dieser Trainingslauf wird mit deutlich größeren Abweichungen zum Sollwert durchgeführt.

```
w11 = 0.5      0.54138      0.5435     0.54914     0.55255       0.555
w12 = -0.5         -0.5    -0.52155    -0.53297    -0.54134    -0.54712
w13 = 0.5      0.54138      0.5435     0.54914     0.55255       0.555
w21 = 0.3      0.33927     0.31482     0.30324     0.29565     0.29064
w22 = 0.3      0.42274     0.46413     0.49539     0.51648     0.53073
w23 = 0.3      0.33927     0.31482     0.30324     0.29565     0.29064
w31 = -0.3      -0.2462    -0.26835    -0.26895    -0.27275    -0.27474
w32 = 1       1.0694      1.0757      1.0905      1.0987      1.1051
w33 = -0.3      -0.2462    -0.26835    -0.26895    -0.27275    -0.27474
```

Die Neuronen der Diagonalelemente feuern, wenn wir die Aktivierungsschwelle > 0,75 setzen.

```
eta11=0.8;  f(y11) = 0.75122
eta12=0.6;  f(y12) = 0.64366
eta13=0.4;  f(y13) = 0.36691
eta21=0.6;  f(y21) = 0.63732
eta22=0.8;  f(y22) = 0.75171
eta23=0.6;  f(y23) = 0.63488
eta31=0.4;  f(y31) = 0.36788
eta32=0.6;  f(y32) = 0.62632
eta33=0.8;  f(y33) = 0.75002
```

Der Fehler des Verfahrens ist immer noch verschwindend gering.

```
E = 0.011634   0.0083397    0.007674   0.0073988   0.0072728
```

Trainingslauf 3

Für Trainingslauf 3 kann die Aktivierungsschwelle > 0,5 gesetzt werden, weil der zweitgrößte vorkommende Wert bei 0,5 liegt. Die nachfolgenden Gewichte ergeben sich nach der 100. Iteration.

```
w11 = 1.4285
w12 = -5.8358
w13 = 1.4285
w21 = -1.3878e-17
w22 = -2.3592e-16
w23 = -1.3878e-17
```



```
w31 = -3.1705
w32 = 2.9836
w33 = -3.1705
```

Bei diesem Lauf sehen wir, daß das mittlere Ausgangsneuron nicht mehr feuert, wenn die Aktivierungsschwelle auf über 0,51 angehoben wird. Das liegt jedoch nicht am Programm, sondern daran, daß die Soll-Ausgangswerte zu weit von den Ist-Ausgabewerten entfernt sind. Während die Ausgangsneuronen 1 und 3 bei 100 Iterationen immer noch größer werden als 0,9, ändert sich am Wert von Ausgangsneuron 2 nichts mehr, die Iteration läuft plateaumäßig aus. Das ist eine bekannte Schwäche neuronaler Netze, die nur durch eine geeignete Wahl der Startwerte vermieden werden kann.

```
eta11=1;    f(y11) = 0.946
eta12=0;    f(y12) = 0.5
eta13=0;    f(y13) = 0.002
eta21=0;    f(y21) = 0.048
eta22=1;    f(y22) = 0.5
eta23=0;    f(y23) = 0.034
eta31=0;    f(y31) = 0.003
eta32=0;    f(y32) = 0.5
eta33=1;    f(y33) = 0.952
```

Der anfängliche Fehler ist ziemlich groß, auch wenn er sich im Laufe von 5 Iterationen auf ein Viertel verringert. Viel kleiner wird er allerdings auch bei noch so vielen Iterationsschritten nicht.

```
E = 2.262      1.5798     0.63431     0.54088     0.49485
```

