Aufgabe: Zeigen Sie, daß ein neuronales Netzwerk in der Lage ist, jede beliebige kinematische Gesetzmäßigkeit nachzubilden.

Beweis: Gegeben sei eine Menge von Funktionen $f(x_i; a_1, a_2, \ldots, a_n)$ und eine Menge von Trainingswertepaaren (x_i, y_i), $i = 1, 2, \ldots, N$, mit $N > n$. Gesucht ist diejenige Funktion $\hat{f}(x_i; a_1, a_2, \ldots, a_n)$, für welche $\rho(\hat{f}(x_i) - y_i) \leq \rho(f(x_i) - y_i)$ für alle f, wobei ρ ein Maß für die Abweichungen $|f(x_i) - y_i|$ ist. Das von C. F. Gauß ausgearbeitete Ausgleichsprinzip besagt, daß die Funktion

$$\rho(f(x_i) - y_i) \equiv \sum_{i=1}^{N} (f(x_i) - y_i)^2$$

einem Minimum zustreben muß. Sind die Fehler normalverteilt, entspricht dieses Prinzip der Maximum-Likelihood-Methode. Bei einem neuronalen Netzwerk müssen wir voraussetzen, daß die Parameter a_i $(i = 1, 2, \ldots, n)$ in f linear vorkommen, d.h.

$$f(x_i; a_1, a_2, \ldots, a_n) = \sum_{k=1}^{n} a_k f_k(x_i).$$

Dann lautet die Ableitung der Ausgleichsfunktion

$$\frac{\partial \rho}{\partial a_j} = 2\sum_{i=1}^{N} (f(x_i) - y_i) \frac{\partial f(x_i)}{\partial a_j} = 2\sum_{i=1}^{N} (f(x_i) - y_i) f_j(x_i),$$

und die notwendige Bedingung für ein Extremum ist gegeben durch

$$\sum_{i=1}^{N} f(x_i) f_j(x_i) = \sum_{i=1}^{N} \sum_{k=1}^{n} a_k f_k(x_i) f_j(x_i) = \sum_{k=1}^{n} a_k \sum_{i=1}^{N} f_k(x_i) f_j(x_i) = \sum_{i=1}^{N} y_i f_j(x_i)$$

für $j = 1, 2, \ldots, n$. Im Spezialfall

$$f_k(x_i) = x_i^{k-1}$$

für $k = 1, 2, \ldots, n$ führt die Methode auf Ausgleichspolynome. Da jede analytische Funktion in eine Potenzreihe entwickelt werden kann, läßt sie sich folglich durch ein neuronales Netzwerk beschreiben. Die Parameterdarstellung des Netzwerks ergibt sich somit aus den n Normalgleichungen für $j = 1, 2, \ldots, n$:

$$\sum_{k=1}^{n} a_k \sum_{i=1}^{N} x_i^{k+j-2} = \sum_{i=1}^{N} y_i x_i^{j-1},$$

die wir für jedes j explizit schreiben können als

$$\begin{aligned}
\sum_{k=1}^{n} a_k \sum_{i=1}^{N} x_i^{k-1} &= \sum_{i=1}^{N} y_i, \\
\sum_{k=1}^{n} a_k \sum_{i=1}^{N} x_i^{k} &= \sum_{i=1}^{N} y_i x_i, \\
\sum_{k=1}^{n} a_k \sum_{i=1}^{N} x_i^{k+1} &= \sum_{i=1}^{N} y_i x_i^2, \\
&\vdots \\
\sum_{k=1}^{n} a_k \sum_{i=1}^{N} x_i^{k+n-2} &= \sum_{i=1}^{N} y_i x_i^{n-1}.
\end{aligned}$$

Lösen wir ferner die Summation über k auf, ergibt sich

$$\begin{aligned}
a_1 N + a_2 \sum_{i=1}^{N} x_i + a_3 \sum_{i=1}^{N} x_i^2 + \ldots + a_n \sum_{i=1}^{N} x_i^{n-1} &= \sum_{i=1}^{N} y_i, \\
a_1 \sum_{i=1}^{N} x_i + a_2 \sum_{i=1}^{N} x_i^2 + a_3 \sum_{i=1}^{N} x_i^3 + \ldots + a_n \sum_{i=1}^{N} x_i^{n} &= \sum_{i=1}^{N} y_i x_i, \\
a_1 \sum_{i=1}^{N} x_i^2 + a_2 \sum_{i=1}^{N} x_i^3 + a_3 \sum_{i=1}^{N} x_i^4 + \ldots + a_n \sum_{i=1}^{N} x_i^{n+1} &= \sum_{i=1}^{N} y_i x_i^2, \\
&\vdots \\
a_1 \sum_{i=1}^{N} x_i^{n-1} + a_2 \sum_{i=1}^{N} x_i^{n} + a_3 \sum_{i=1}^{N} x_i^{n+1} + \ldots + a_n \sum_{i=1}^{N} x_i^{2n-2} &= \sum_{i=1}^{N} y_i x_i^{n-1}.
\end{aligned}$$

In übersichtlicher Matrixdarstellung lautet dieses Gleichungssystem

$$\begin{pmatrix}
N & \sum_{i=1}^{N} x_i & \sum_{i=1}^{N} x_i^2 & \cdots & \sum_{i=1}^{N} x_i^{n-1} \\
\sum_{i=1}^{N} x_i & \sum_{i=1}^{N} x_i^2 & \sum_{i=1}^{N} x_i^3 & \cdots & \sum_{i=1}^{N} x_i^{n} \\
\sum_{i=1}^{N} x_i^2 & \sum_{i=1}^{N} x_i^3 & \sum_{i=1}^{N} x_i^4 & \cdots & \sum_{i=1}^{N} x_i^{n+1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\sum_{i=1}^{N} x_i^{n-1} & \sum_{i=1}^{N} x_i^{n} & \sum_{i=1}^{N} x_i^{n+1} & \cdots & \sum_{i=1}^{N} x_i^{2n-2}
\end{pmatrix}
\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{pmatrix} =
\begin{pmatrix}
\sum_{i=1}^{N} y_i \\
\sum_{i=1}^{N} y_i x_i \\
\sum_{i=1}^{N} y_i x_i^2 \\
\vdots \\
\sum_{i=1}^{N} y_i x_i^{n-1}
\end{pmatrix}.$$

Mit den Definitionen

$$x_{jk} \equiv \sum_{i=1}^{N} x_i^{k+j-2}, \quad b_j \equiv \sum_{i=1}^{N} y_i x_i^{j-1}$$

folgt ein inhomogenes lineares Gleichungssystem mit konstanten Koeffizienten:

$$\begin{pmatrix}
x_{11} & x_{12} & x_{13} & \cdots & x_{1n} \\
x_{21} & x_{22} & x_{23} & \cdots & x_{2n} \\
x_{31} & x_{32} & x_{33} & \cdots & x_{3n} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
x_{n1} & x_{n2} & x_{n3} & \cdots & x_{nn}
\end{pmatrix}
\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{pmatrix} =
\begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}.$$

Dieses kann vektoriell auch geschrieben werden als $\mathbf{Xa} = \mathbf{b}$. Wegen der Regularität von $\mathbf{X}$ kann die Lösung $\mathbf{a}$ in der Form $\mathbf{a} = \mathbf{X}^{-1}\mathbf{b}$ angegeben werden, d.h.

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{pmatrix} = \frac{1}{\det \mathbf{X}} \begin{pmatrix} X_{11} & X_{21} & X_{31} & \cdots & X_{n1} \\ X_{12} & X_{22} & X_{32} & \cdots & X_{n2} \\ X_{13} & X_{23} & X_{33} & \cdots & X_{n3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ X_{1n} & X_{2n} & X_{3n} & \cdots & X_{nn} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}.$$

Mit den Adjunkten

$$X_{jk} = (-1)^{j+k} \begin{vmatrix} x_{11} & x_{12} & \cdots & x_{1,k-1} & x_{1,k+1} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2,k-1} & x_{2,k+1} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_{j-1,1} & x_{j-1,2} & \cdots & x_{j-1,k-1} & x_{j-1,k+1} & \cdots & x_{j-1,n} \\ x_{j+1,1} & x_{j+1,2} & \cdots & x_{j+1,k-1} & x_{j+1,k+1} & \cdots & x_{j+1,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{n,k-1} & x_{n,k+1} & \cdots & x_{nn} \end{vmatrix}$$

ergibt sich die Determinante des homogenen Systems nach dem Laplaceschen Entwicklungssatz zu

$$D = \sum_{j=1}^{n} x_{jk} X_{jk}.$$

Das *k*-te Gewicht a_k erhalten wir gemäß der Formel

$$a_k = \frac{1}{\det \mathbf{X}} \left(X_{1k} b_1 + X_{2k} b_2 + X_{3k} b_3 + \ldots + X_{nk} b_n \right) = \frac{1}{\det \mathbf{X}} \begin{pmatrix} x_{11} & \cdots & b_1 & \cdots & x_{1n} \\ x_{21} & \cdots & b_2 & \cdots & x_{2n} \\ x_{31} & \cdots & b_3 & \cdots & x_{3n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{n1} & \cdots & b_n & \cdots & x_{nn} \end{pmatrix}$$

bzw. $a_k = D_k / D$. Für das praktische Rechnen ist es besser, die Gaußsche Summenschreibweise zu verwenden, d.h. mit den Definitionen für *k*fache Potenzen von *x*,

$$\left[x^k\right] \equiv \sum_{i=1}^{N} x_i^k \quad \text{und} \quad \left[yx^k\right] \equiv \sum_{i=1}^{N} y_i x_i^k,$$

läßt sich der obige Ausdruck übersichtlicher schreiben als

$$\begin{pmatrix} N & [x] & [x^2] & \cdots & [x^{n-1}] \\ [x] & [x^2] & [x^3] & \cdots & [x^n] \\ [x^2] & [x^3] & [x^4] & \cdots & [x^{n+1}] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ [x^{n-1}] & [x^n] & [x^{n+1}] & \cdots & [x^{2n-2}] \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} [y] \\ [yx] \\ [yx^2] \\ \vdots \\ [yx^{n-1}] \end{pmatrix},$$

wobei wir noch die Abkürzungen $N = [x^0]$, $[x] = [x^1]$, $[y] = [yx^0]$ und $[yx] = [yx^1]$ verwendet haben. Im Falle, daß die *x*-Koordinate einer gewöhnlichen Zeitabhängigkeit entspricht, zeigen wir nun an einem einfachen Beispiel, daß das neuronale Netz die exakte naturgesetzliche Abhängigkeit herausfindet. Das setzt allerdings voraus, daß unsere Trainingswerte möglichst genau gemessen wurden.

Betrachten wir z.B. den Fall eines schiefen Wurfs, für den das Weg-Zeit-Gesetz

$$y(t_i) = y_0 + v_0 \sin\alpha \cdot t_i - \frac{1}{2} g t_i^2$$

gilt. Mit $n = 3$ und $x \equiv t$ lautet die Ausgleichsfunktion

$$f(t_i; a_1, a_2, a_3) = \sum_{k=1}^{3} a_k t_i^{k-1} = a_1 + a_2 t_i + a_3 t_i^2,$$

und die konstanten Koeffizienten sind gegeben durch

$$a_1 = y_0, \quad a_2 = v_0 \sin\alpha, \quad a_3 = -\frac{1}{2} g.$$

Das neuronale Netzwerk hierzu sieht wie folgt aus:

$$\begin{pmatrix} N & [t] & [t^2] \\ [t] & [t^2] & [t^3] \\ [t^2] & [t^3] & [t^4] \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} [y] \\ [yt] \\ [yt^2] \end{pmatrix},$$

wobei

$$[t^k] = \sum_{i=1}^{N} t_i^k \quad \text{und} \quad [yt^k] = \sum_{i=1}^{N} y_i t_i^k.$$

Die Abbildung besitzt die Determinante

$$D = \begin{vmatrix} N & [t] & [t^2] \\ [t] & [t^2] & [t^3] \\ [t^2] & [t^3] & [t^4] \end{vmatrix} = N\left([t^2][t^4] - [t^3]^2\right) - [t]\left([t][t^4] - [t^2][t^3]\right) + [t^2]\left([t][t^3] - [t^2]^2\right),$$

und die drei Koeffizientendeterminanten sind gegeben durch

$$D_1 = \begin{vmatrix} [y] & [t] & [t^2] \\ [yt] & [t^2] & [t^3] \\ [yt^2] & [t^3] & [t^4] \end{vmatrix} = [y]\left([t^2][t^4]-[t^3]^2\right)-[yt]\left([t][t^4]-[t^2][t^3]\right)+[yt^2]\left([t][t^3]-[t^2]^2\right),$$

$$D_2 = \begin{vmatrix} N & [y] & [t^2] \\ [t] & [yt] & [t^3] \\ [t^2] & [yt^2] & [t^4] \end{vmatrix} = -[y]\left([t][t^4]-[t^2][t^3]\right)+[yt]\left(N[t^4]-[t^2]^2\right)-[yt^2]\left(N[t^3]-[t][t^2]\right),$$

$$D_3 = \begin{vmatrix} N & [t] & [y] \\ [t] & [t^2] & [yt] \\ [t^2] & [t^3] & [yt^2] \end{vmatrix} = [y]\left([t][t^3]-[t^2]^2\right)-[yt]\left(N[t^3]-[t][t^2]\right)+[yt^2]\left(N[t^2]-[t]^2\right).$$

Mit diesen Hilfsgrößen können wir sofort die Gewichte des neuronalen Netzwerks angeben:

$$a_1 = \frac{D_1}{D}, \quad a_2 = \frac{D_2}{D}, \quad a_3 = \frac{D_3}{D}.$$

Seien etwa $\alpha = \pi/4$, $v_0 = 10$ m/s, $y_0 = 2$ m und $g = 10$ m/s^2. Für diese Konfiguration trainieren wir nun das Netzwerk in Abständen von 1/10 s im Intervall $[0, 2]$ mit $N = 21$ Meßwerten. Das ergibt folgende Trainingsgleichung:

$$\left[\frac{y_i}{\text{m}}\right] = 2 + 5\sqrt{2}\left[\frac{t_i}{\text{s}}\right] - 5\left[\frac{t_i}{\text{s}}\right]^2.$$

Die Koeffizienten werden mit dem im Anhang gelisteten Programm bestimmt. Sie lauten

$$a_1 = 2, \quad a_2 = 7{,}071, \quad a_3 = -5$$

qed

Anmerkung: In einem unbekannten, zeitlich veränderlichen System müssen also, um das System zu trainieren, aus dem gegebenen Szenario hinreichend viele Meßwerte entnommen werden, damit die Gewichte hinreichend gut bestimmt werden können. Sind die Gewichte anschließend bekannt, kann die zugrunde liegende physikalische Gesetzmäßigkeit extrapoliert werden. Man könnte damit z.B. herausfinden, wohin ein ungelenkter ballistischer Körper fliegt, und diesen abfangen. Ein aktiv gelenkter Flugkörper muß nach jedem größeren Lenkausschlag erneut in Echtzeit trainiert werden, da die obige Minimalbedingung einem völlig anderen Weg-Zeit-Gesetz gehorcht.[1] Ein intelligentes Netzwerk sollte aber in der Lage sein, solche Änderungen zu durchschauen und sich davon nicht beirren lassen.

[1] Es hat dem Hasen in der Evolution zum Überleben verholfen, daß er Haken schlagen kann.

Anhang

```
% Determinanten
clear all

N = 21;
% Trainingswerte
for j=1:N
    for i=1:N
        t(i) = (i-1)*2/(N-1);
        y(i) = 2+5*sqrt(2)*t(i)-5*t(i)^2;
        yt(i) = y(i)*t(i);
        yt2(i) = y(i)*t(i)^2;
        t2(i) = t(i)^2;
        t3(i) = t(i)^3;
        t4(i) = t(i)^4;
    end
end

% Summen
t_0 = N
t_1 = sum(t)
t_2 = sum(t2)
t_3 = sum(t3)
t_4 = sum(t4)
yt_0 = sum(y)
yt_1 = sum(yt)
yt_2 = sum(yt2)

% Determinanten
D = t_0*(t_2*t_4-t_3^2)-t_1*(t_1*t_4-t_2*t_3)+t_2*(t_1*t_3-t_2^2)
D1 = yt_0*(t_2*t_4-t_3^2)-yt_1*(t_1*t_4-t_2*t_3)+yt_2*(t_1*t_3-t_2^2)
D2 = -yt_0*(t_1*t_4-t_2*t_3)+yt_1*(t_0*t_4-t_2^2)-yt_2*(t_0*t_3-t_1*t_2)
D3 = yt_0*(t_1*t_3-t_2^2)-yt_1*(t_0*t_3-t_1*t_2)+yt_2*(t_0*t_2-t_1^2)

% Gewichte
a1 = D1/D
a2 = D2/D
a3 = D3/D

>> determinanten
t_0 = 21
t_1 = 21.0000
t_2 = 28.7000
t_3 = 44.1000
t_4 = 72.2666
yt_0 = 46.9924
yt_1 = 24.4396
yt_2 = 7.9011
D = 362.7362
D1 = 725.4724
D2 = 2.5649e+03
D3 = -1.8137e+03
a1 = 2.0000
a2 = 7.0711
a3 = -5.0000
```

