UNIVERSITÄT FRANKFURT AM MAIN

www.uni-frankfurt.de

Adaptive Systeme

Prof. Rüdiger Brause

WS 2013

Organisation

UNIVERSITÄT FRANKFURT AM MAIN

- **„Einführung in adaptive Systeme“ B-AS-1, M-AS-1**
  - Vorlesung Dienstags 10-12 Uhr, SR11
  - Übungen Donnerstags 12-13 Uhr, SR 9
- **„Adaptive Systeme“ M-AS-2** (Theorie)
  - Vorlesung Donnerstags 10-12 Uhr, SR 9
  - Übungen Donnerstags 13-14 Uhr, SR 9
- **Tutor:** Markus Hildebrand *MarkHild@stud.uni-frankfurt.de*
- **Gemeinsames Übungsblatt,** unterteilt in 2 Teile

Ausgabe: Dienstags, **Abgabe**: Dienstags per email

Besprechung: Donnerstags

UNIVERSITÄT FRANKFURT AM MAIN

Einführung

Grundlagen

Modellierung

Vorschau Themen

1. Einführung und Grundlagen
2. Lernen und Klassifizieren
3. Merkmale und lineare Transformationen
4. Lokale Wechselwirkungen: *Konkurrentes Lernen*
5. Netze mit RBF-Elementen
6. Fuzzy-Systeme
7. Evolutionäre und genetische Algorithmen
8. Schwarmalgorithmen

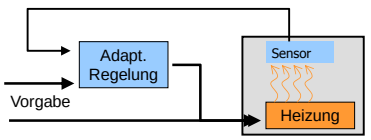
Arten von Adaptiven Systeme

- **Vorbild Gehirn**
  - Neuronale Netze
  - Psychologisch-kognitive Modelle
- **Biologische Systeme**
  - Evolutionäre Systeme
  - Schwarm-Intelligenz: Ameisen-Algorithmen,...
  - Molekular-genetische Selbstorganisation
- **Soziale Systeme**
  - Gruppenprozesse
  - Soziale Selbstordnung
- **Physikalische Systeme**
  - Synergieeffekte
- **Technische Systeme**

Einleitung: Was sind Adaptive Systeme?

- **Selbst-anpassende Systeme – statt programmieren**

  Beispiel: Temperaturregler statt fester Heizeinstellung

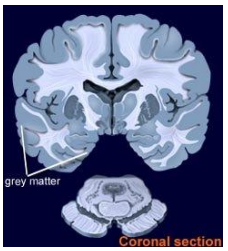


- **„Lernende“ Systeme**
  - Trainierte Systeme (Trainingsphase-Testphase)
  - Selbstlernende Systeme (Orientierung an Daten)

Einsatzgebiete Adaptiver Systeme

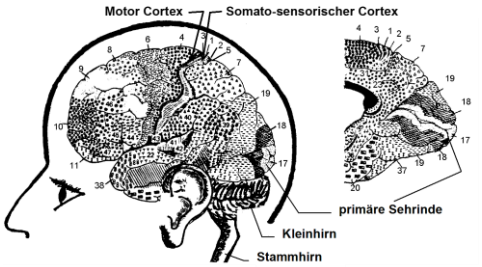
- **Echtzeitreaktionen**
  Stahlwalzstraßen, Flugzeugsteuerung,..
- **Analytisch unbekannte Abhängigkeiten**
  Polymerchemie, DNA-Schätzungen, ..
- **Analytisch nicht zugängige Abhängigkeiten**
  psychische Faktoren, ergebnisverändernde Messungen,..
- **Analytisch nur unter großem Aufwand bearbeitbare, hochdimensionale Gesetzmäßigkeiten**
  Wechselkursabhängigkeiten,..
- **Statistische Analysen durch untrainierte Benutzer (?!)**

Vorbild Gehirn



Das Vorbild: Gehirnfunktionen

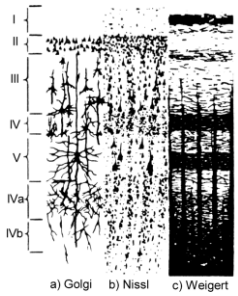
- Unterteilung des Gehirns in funktionale Bereiche



Gehirn = 2-dim „Tuch“

Das Vorbild: Gehirnfunktionen

- Unterteilung der Neuronenschicht: *Darstellungsarten*



Das Vorbild: Gehirnfunktionen

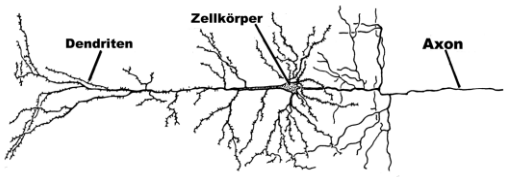
- Neuronentypen

a)-c) Pyramidenzellen

f,h) Stern/Glia zellen

Das Vorbild: Gehirnfunktionen

- Pyramidalzellen

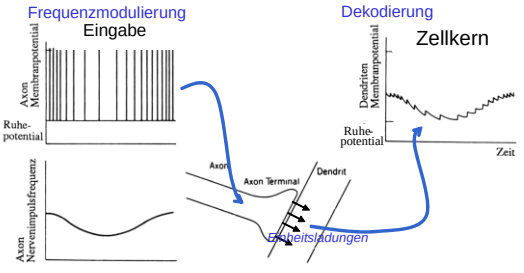


Das Vorbild: Gehirnfunktionen

- Signalverarbeitung Output

Das Vorbild: Gehirnfunktionen

- Signalverarbeitung Input

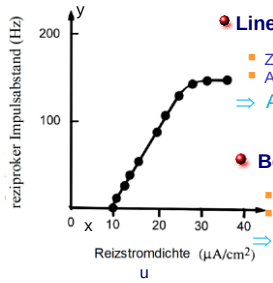


Zell-Potential ~ Eingabe-Spikefrequenz

Das Vorbild: Gehirnfunktionen

- **Signalverarbeitung Input-Output**

Tintenfisch-Riesenneuron: Ausgabe-Spikefrequenz ~ Zellstrom



- **Lineares Modell** z = kx
  - Zell-Potential ~ Eingabe-Spikefrequenz
  - Ausgabe-Spikefrequenz ~ Zellstrom

  ⇒ Ausgabe-Freq.z ~ Eingabe-Freq.x

- **Begrenzt-Lineares Modell** y = S(z)
  - Sättigung wenn zu große Aktivität
  - Keine Ausgabe wenn keine Aktivität

  ⇒ Ausgabe-Freq. S(z) limitiert

Frage

- **Wie lautet die math. Schreibweise für die begrenzt-lineare Aktivität y = S(z)?**

$$y = \begin{cases} z_{max} & z > s \\ z & -s < z < s \\ -z_{max} & z < -s \end{cases}$$

Einführung

Grundlagen

Modellierung

Modellierung

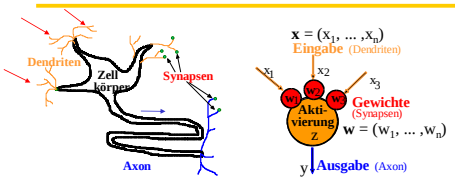
UNIVERSITÄT FRANKFURT AM MAIN

Informatik: Granularität Paralleler Aktivität

- **Grob:** Computer, Jobs (Lastverteilung) *wenig Komm.*
- **Fein:** Multi-CPU, Threads *viel Komm.*
- **Sehr fein:** formale Neuronen, Funktionen *sehr viel Komm*

Modellierung formaler Neuronen

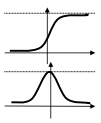
UNIVERSITÄT FRANKFURT AM MAIN



Ausgabefunktionen

squashing function

radial basis function



$y = S(z)$ $\qquad z = \sum_{i=1}^{n} w_i x_i = \mathbf{w}^T\mathbf{x}$

Formale Neuronen

UNIVERSITÄT FRANKFURT AM MAIN

- **Code-Beispiel „formales Neuron“**

Aktivität

```
float   z (float w[ ],float x[ ]) {
/* Sigma-Neuron: Aufsummieren aller Eingaben */
float sum;

   sum = 0;          /* Skalarprodukt bilden */
   for (int i = 0; i<x.length; i++) {
         sum= sum  + w[i]*x[i];
   }
   return sum;
}
```

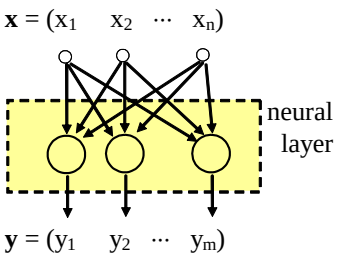
Ausgabe

```
float S (float z) {
(* begrenzt-lineare Ausgabefunktion*)
float s=1.0;   float Zmax=1;   float k=Zmax/s;
         z = z*k;
         if (z> s)  z = Zmax;
         if (z<-s)  z =-Zmax;
         return z;
}
```

Schichten

UNIVERSITÄT FRANKFURT AM MAIN

- **DEF Schicht**



Frage

UNIVERSITÄT FRANKFURT AM MAIN

- **Wie lautet der Code für eine ganze Schicht?**

```
float   [ ] Y (float w[ ],float x[ ]) {
/* Für jedes Sigma-Neuron extra Aktivität bilden */

float [ ] y = new float[m];          float sum;

for (int k = 0; k < m; k++) {
   sum = 0;        /* Skalarprodukt bilden */
   for (int i = 0; i < x.length; i++) {
         sum = sum  + w[i]*x[i];
   }
  y[k] = S(sum);
}
return y;
```

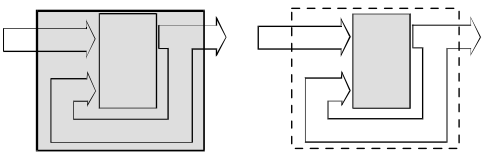

Modellierung der Netze

UNIVERSITÄT FRANKFURT AM MAIN

- **Feed-forward vs. Feedback-Netze**

  DEF *feedforward* Netz ⇔ der gerichtete Graph ist *zyklenfrei*

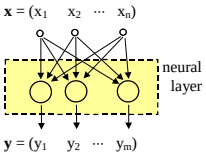
  Problem: *feedforward* oder nicht?



Lineare Transformation mit NN

UNIVERSITÄT FRANKFURT AM MAIN

- **lineare Schicht**



$$y_1 = (w_{11},\ldots,w_{1n})\,\mathbf{x}$$
$$\vdots$$
$$y_m = (w_{m1},\ldots,w_{mn})\,\mathbf{x}$$

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{m1} & \cdots & w_{mn} \end{bmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \mathbf{W}\cdot\mathbf{x}$$ *Matrix-Multiplikation*