Systems Neuroscience &
Neurotechnology Unit

BMT 823 Neural & Cognitive Modeling

Artificial Neural Networks (ANN)

Dipl.W.-Ing. Michael Busse, M.Sc.
Saarland University, Saarland University Hospital,
and Saarland University of Applied Sciences

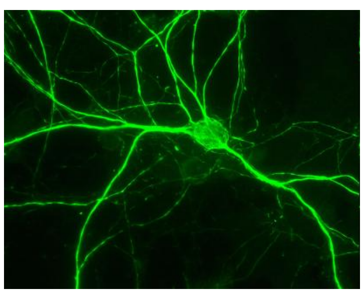
Basic Motivation

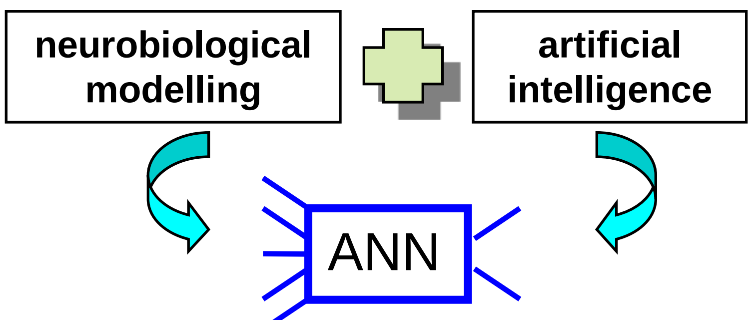
- Inspired by how the brain processes information - NN architectures are inspired by the architecture of biological nervous systems, which use many simple processing elements operating in parallel to obtain high computation rates.
- Composed of a large set of highly interconnected processing elements (neurons) working in parallel to solve specific problems
- ANN, like humans, learn by example
- pattern recognition, data classification of an ANN is configured through a learning process

Definition of Neural Networks

Definition of Neural Networks (given by DARPA[1] 1998):

> a system composed of many simple processing elements operating in parallel whose functions are determined by network structures, connection strengths and the processing performed at the computing elements or nodes[2].



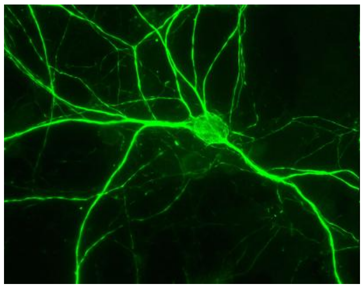


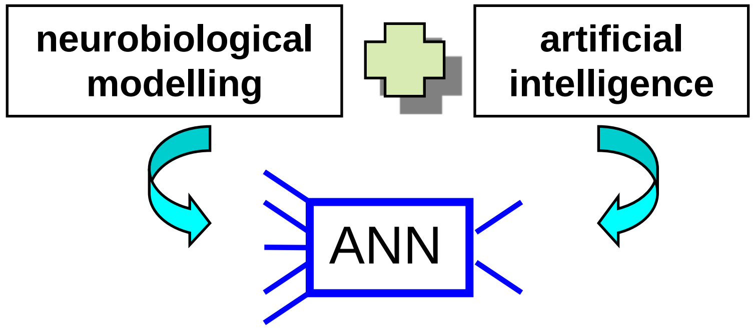
[1] DARPA = Defense Advanced Research Projects Agency

[2] nodes are functional units in NN also referred to as “units“, “cells“ and “populations“.

Definition of Neural Networks

- **Reunification of**

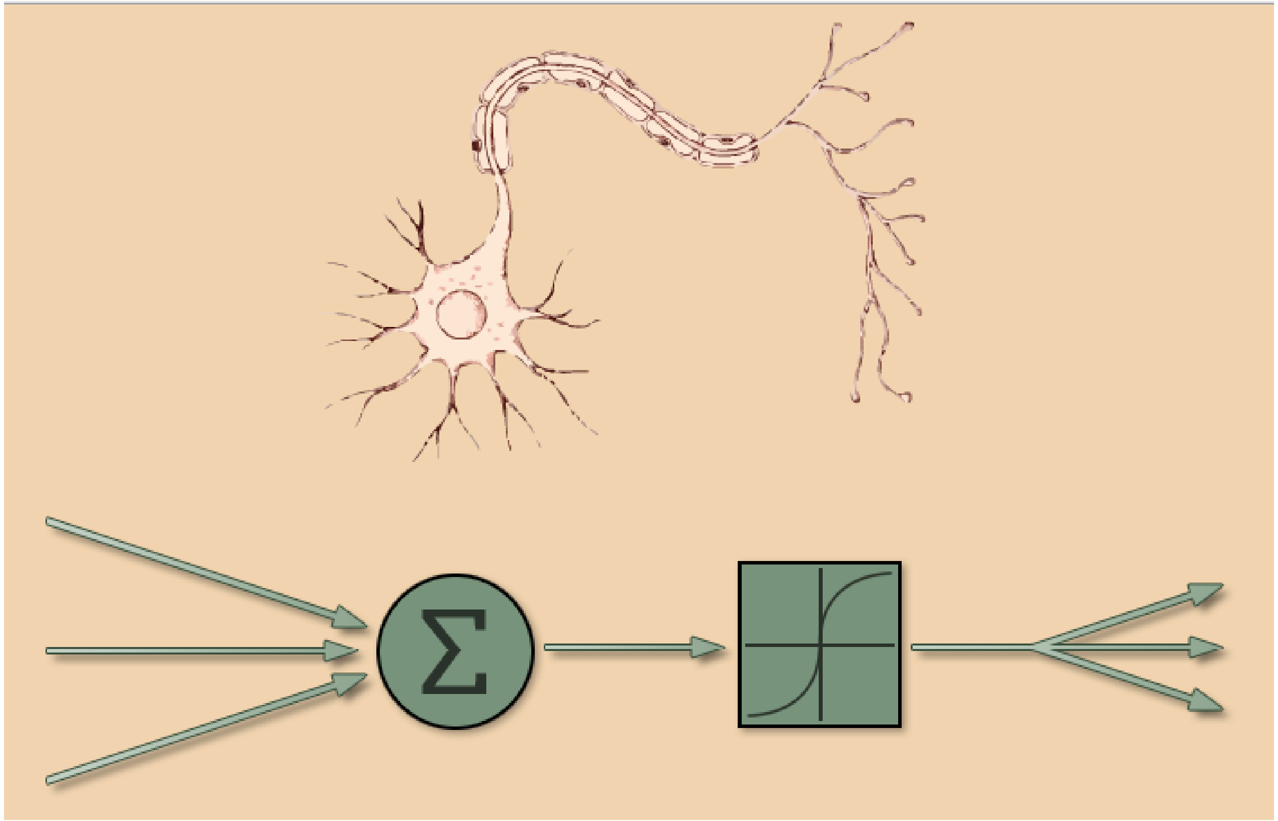




Machines performing cognitive functions are now built according to HOW brains are performing cognitive functions (simulated brain regions)

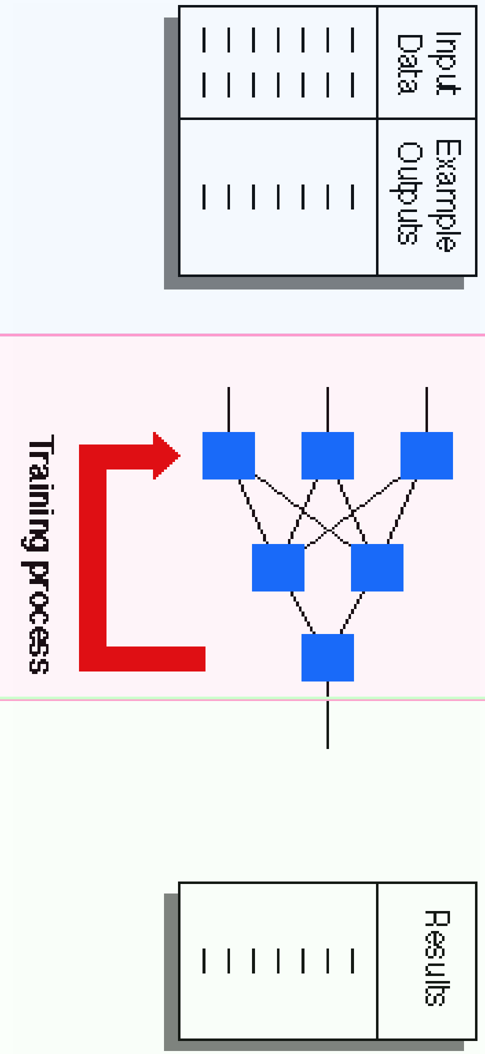
- **This development is coined as connectionism or more famously as Artificial Neural Network (ANN)**

brief overview:



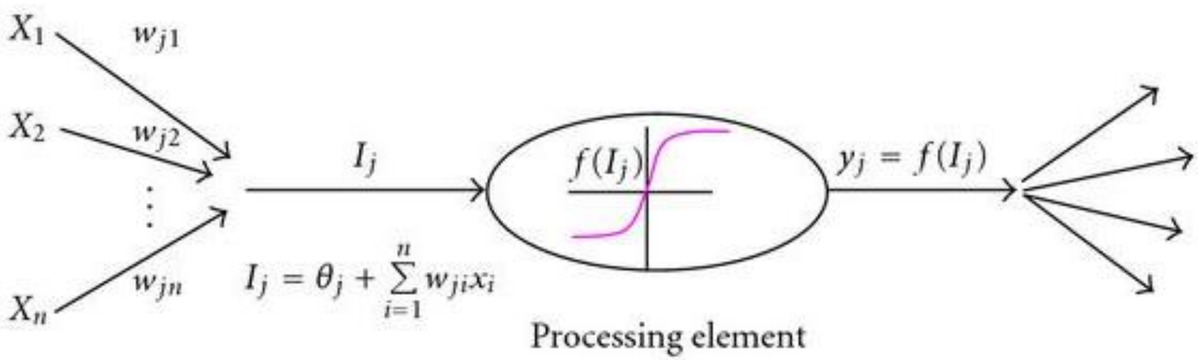
brief overview:

- An Artificial Neural Network is an adaptive, most often nonlinear system that learns to perform a function (an input/output map) from data.
- Adaptive means that the system parameters are changed during operation, normally called the training phase.
- After the training phase the Artificial Neural Network parameters are fixed and the system is deployed to solve the problem at hand (the testing phase).



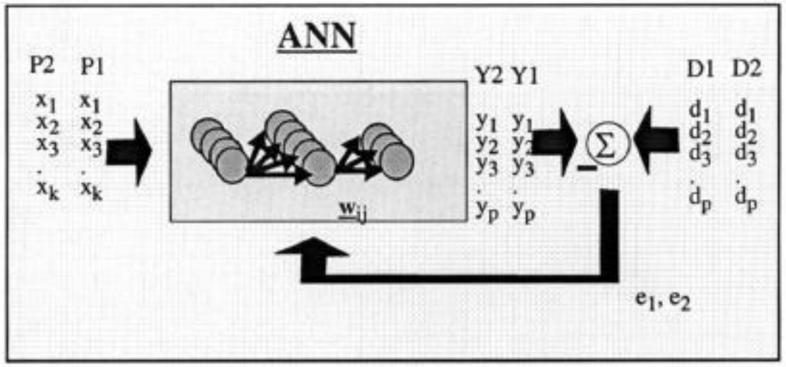
brief overview:

- The input/output training data are fundamental in ANN technology, because they convey the necessary information to "discover" the optimal operating point.
- The nonlinear nature of the neural network processing elements (PEs) provides the system with lots of flexibility to achieve practically any desired input/output map



brief overview:

- An input is presented to the NN & a corresponding target response set at the output (called supervised).
- An error is composed from the difference between the desired response and the system output. This error information is fed back to the system and adjusts the system parameters in a systematic fashion (the learning rule).

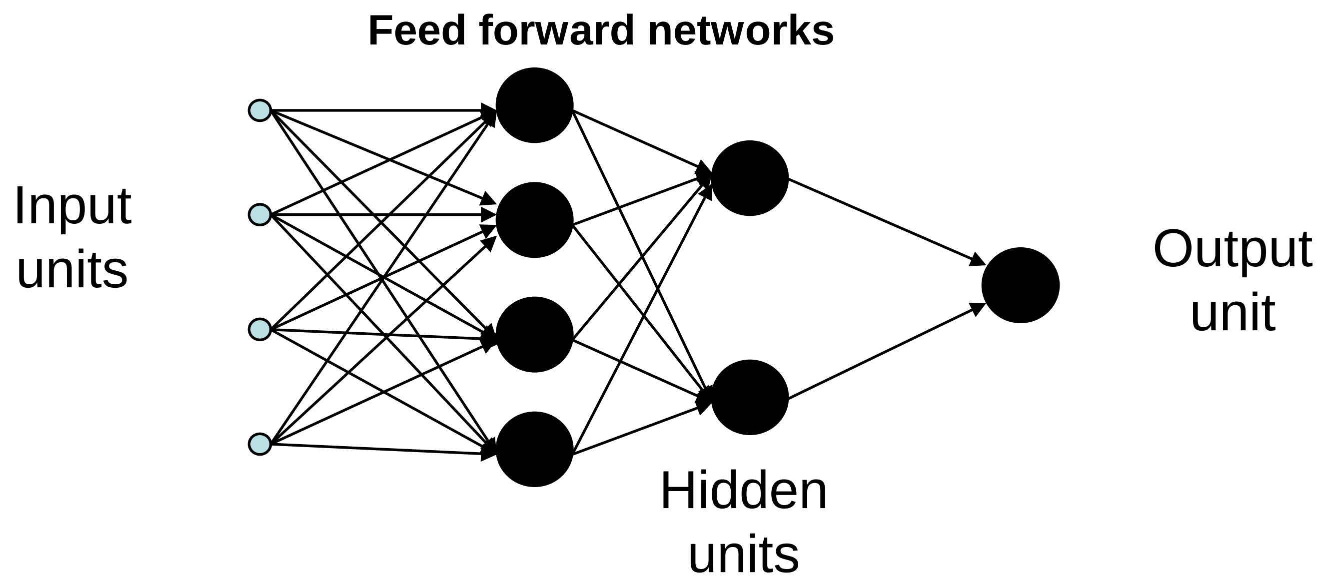


The style of neural computation.

- The process is repeated until the performance is acceptable.

brief overview:

- the performance hinges heavily on the data.
  - If one does not have data that cover a significant portion of the operating conditions or if they are noisy, then **NN** technology is probably not the right solution.
  - If there is plenty of data and the problem is poorly understood to derive an approximate model, then neural network technology is a choice.



Classification of ANNs

- ANN can be classified in a number of ways:
  - Network architecture
    1. Feed forward networks
    2. Recurrent networks
    3. Competitive networks
  - Mode of learning
    1. Supervised learning
    2. Unsupervised learning
    3. Reinforcement learning

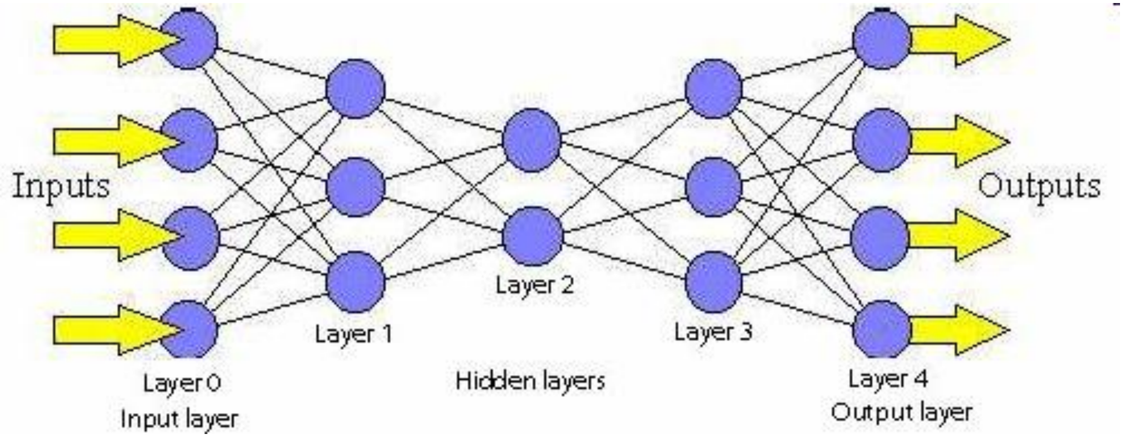
Neural Network topologies

focus on the pattern of connections between the units and the propagation of data. As for this pattern of connections, the main distinction that can be made is between:

- **Feed-forward neural networks**, where the data flow from input to output units is strictly feedforward. The data processing can extend over multiple (layers of) units, but no feedback connections are present, that is, connections extending from outputs of units to inputs of units in the same layer or previous layers.
- **Recurrent neural networks** that do contain feedback connections. Contrary to feed-forward networks, the dynamical properties of the network are important. In some cases, the activation values of the units undergo a relaxation process such that the neural network will evolve to a stable state in which these activations do not change anymore. In other applications, the change of the activation values of the output neurons are significant, such that the dynamical behaviour constitutes the output of the neural network (Pearlmutter, 1990).
- Classical examples of feed-forward neural networks are the Perceptron and Adaline. Examples of recurrent networks have been presented by Anderson (Anderson, 1977), Kohonen (Kohonen, 1977), and Hopfield (Hopfield, 1982) .

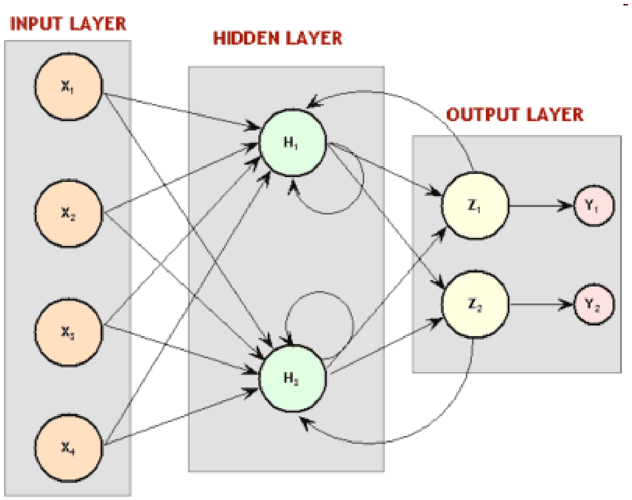
Feed forward networks

- Perceptrons are arranged in layers, with the first layer taking in inputs and the last layer producing outputs. The middle layers have no connection with the external world, and hence are called hidden layers.
- Each perceptron in one layer is connected to every perceptron on the next layer. Hence information is constantly "fed forward" from one layer to the next (so called feed-forward networks).
- There is no connection among perceptrons in the same layer.



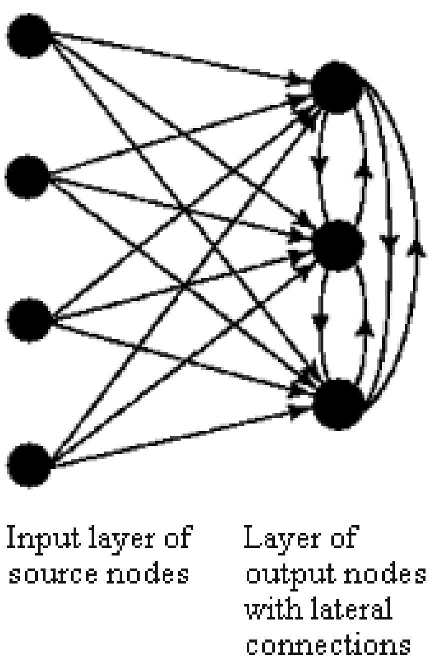
Recurrent networks

- Recurrent network has input nodes, hidden neurons, output neurons, just as feedforward networks, what distinguish themselves from feedforward networks is that they have at least one feedback loop, i.e. when a neurons output is fed back into the network as input.
- so they not only operate just on an input space but also on an internal *state*
- connections between units form a directed cycle. This internal state of the network allows it to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. (This makes them applicable to tasks such as unsegmented connected handwriting recognition, where they have achieved the best known results.)



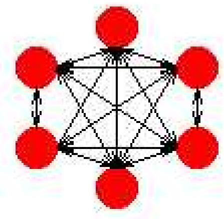
Competitive networks

- **Competitive learning** is a form of unsupervised learning, in which nodes compete for the right to respond to a subset of the input data. (A variant of Hebbian learning, competitive learning works by increasing the specialization of each node in the network.)
- It is well suited to finding clusters within data.

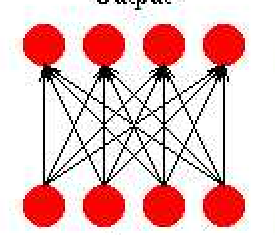


Competitive networks

- Simple competitive networks are composed of two networks: the Hemming net and the Maxnet. Each of them specializes in a different function:
- 1. The Hemming net measures how much the input vector resembles the weight vector of each perceptron.
- 2. The maxnet finds the perceptron with the maximum value.



A Maxnet



A Hemming net

Learning Rules

- A learning algorithm refers to a procedure in which *learning rules* are used for adjusting the weights, which formally can be defined in the context of neural networks as follows [haykin94] :
- "Learning is a process by which the free parameters (the weights) of a neural network are adapted through a continuing process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameter changes take place.
- Which implies the following sequence of events:
- The neural network is stimulated by an environment.
- The neural network undergoes changes as a result of this stimulation.
- The neural network responds in a new way to the environment, because of the changes that have occurred in its internal structure."

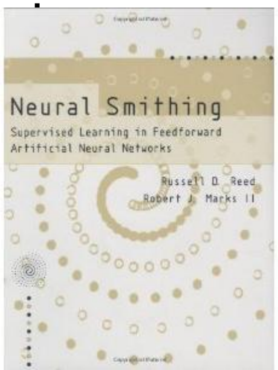
Supervised learning

- Learning in feed-forward networks belongs to the realm of supervised learning, in which pairs of input and output values are fed into the network for many cycles, so that the network 'learns' the relationship between the input and output.
- In backpropagation learning, every time an input vector of a training sample is presented, the output vector *o* is compared to the desired value *d*.
- The comparison is done by calculating the squared difference of the two:

$$Err = (d-o)^2$$

- The value of *Err* tells us how far away we are from the desired value for a particular input. The goal of backpropagation is to minimize the sum of *Err* for all the training samples, so th network behaves in the most "desirable" way.

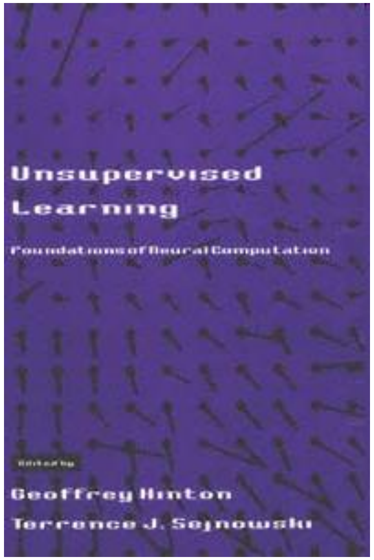
$$\textit{minimize} \quad \sum Err = (d-o)^2$$



Unsupervised learning

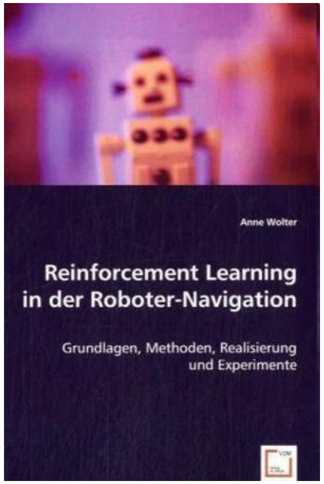
- Unsupervised learning techniques use only input data and attempt through self organization to divide the examples presented to the network inputs up into categories or groups with similar charkteristics.

- Unsupervised learning can act as type of discovery process identifying significant features in the input patterns presented to it.



Reinforcement learning

- Similar to supervised learning
- Instead being provided with the correct output for each network input, the algorithm is given a grade, which is a measure of the network performance over some sequence of inputs.



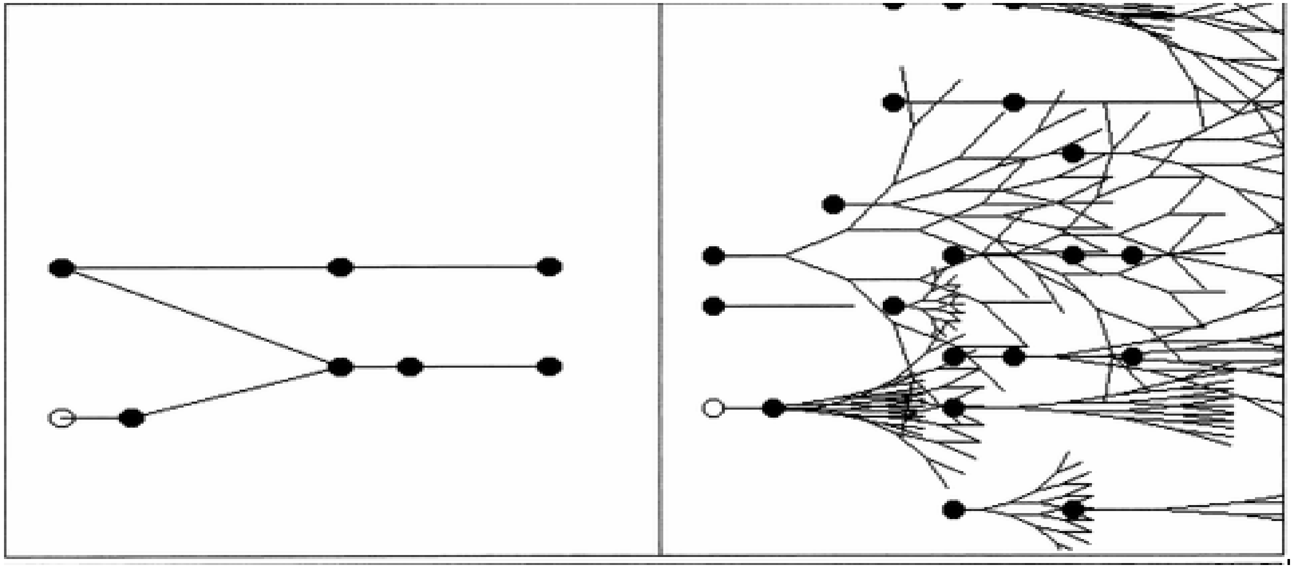
Recapitulation - Training of artifcial neural networks

We can categorise the learning situations in three distinct sorts. These are:

- **Supervised learning** or Associative learning in which the network is trained by providing it with input and matching output patterns. These input-output pairs can be provided by an external teacher, or by the system which contains the neural network (self-supervised).
- **Unsupervised learning** or Self-organisation in which an (output) unit is trained to respond to clusters of pattern within the input. In this paradigm the system is supposed to discover statistically salient features of the input population. Unlike the supervised learning paradigm, there is no a priori set of categories into which the patterns are to be classified; rather the system must develop its own representation of the input stimuli.
- **Reinforcement Learning** This type of learning may be considered as an intermediate form of the above two types of learning. Here the learning machine does some action on the environment and gets a feedback response from the environment. The learning system grades its action good (rewarding) or bad (punishable) based on the environmental response and accordingly adjusts its parameters. Generally, parameter adjustment is continued until an equilibrium state occurs, following which there will be no more changes in its parameters. The selforganizing neural learning may be categorized under this type of learning.

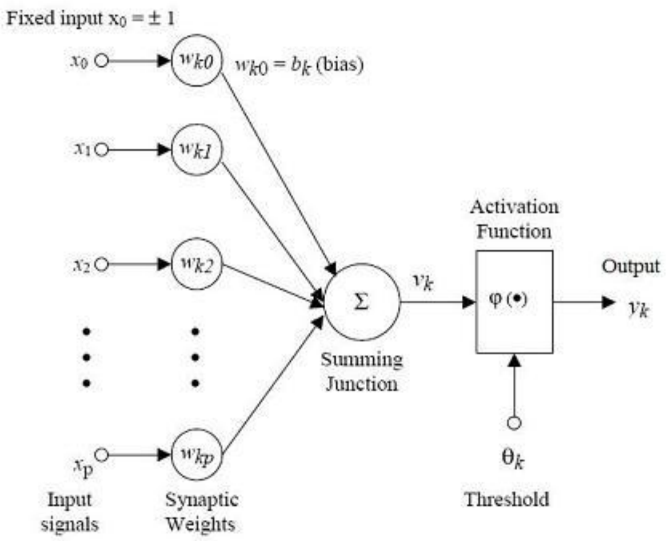
Subnetworks

- Subnetworks or larger networks with ANN principles can be re-use in other models, the connections can be added or reduced.
- NN theory addresses complex problems, no single model has cracked the problem of categorization/attention/memory.



Mathematical background

- When creating a functional model of the biological neuron, there are 3 basic components of importance:
- 1st: synapses of the neuron are modeled as weights. The strength of the connection is noted by the value of the weight. Negative weight values reflect inhibitory connections, while positive values designate excitatory connections.
- 2nd: An adder sums up all the inputs modified by their respective weights (linear combination).
- 3rd: Finally, an activation function controls the amplitude of the output.

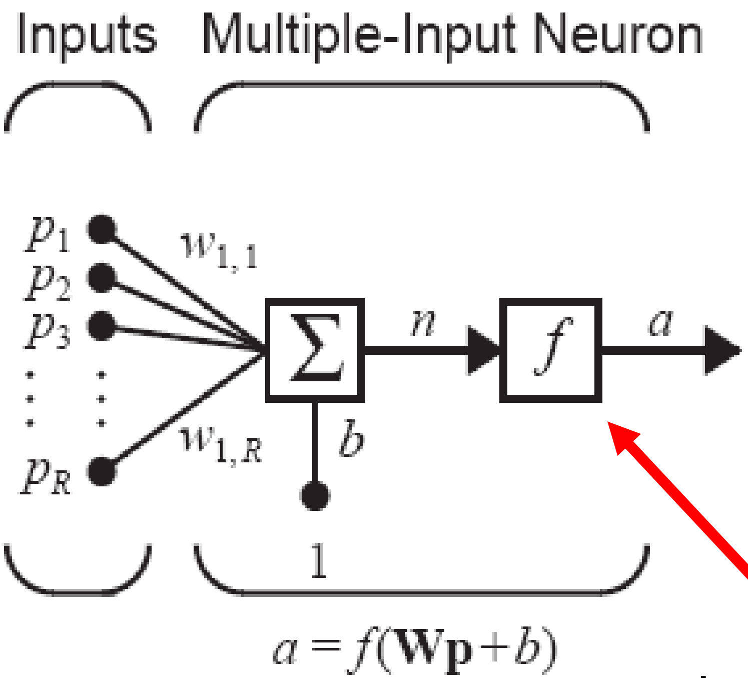


- From this model the interval activity of the neuron can be shown to be:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j$$

- The output of the neuron, yk, would therefore be the outcome of some activation function on the value of vk.

Activation functions



the activation function acts such that the output of a neuron in a neural network is between certain values (usually 0 and 1, or -1 and 1). In general, there are three types of activation functions:

Activation functions

- 1st: the Threshold Function which takes on a value of 0 if the summed input is less than a certain threshold value (v), and the value 1 if the summed input is greater than or equal to the threshold value.

$$\varphi(v)=\begin{cases}1 & \text{if } v\geq 0\\ 0 & \text{if } v<0\end{cases}$$

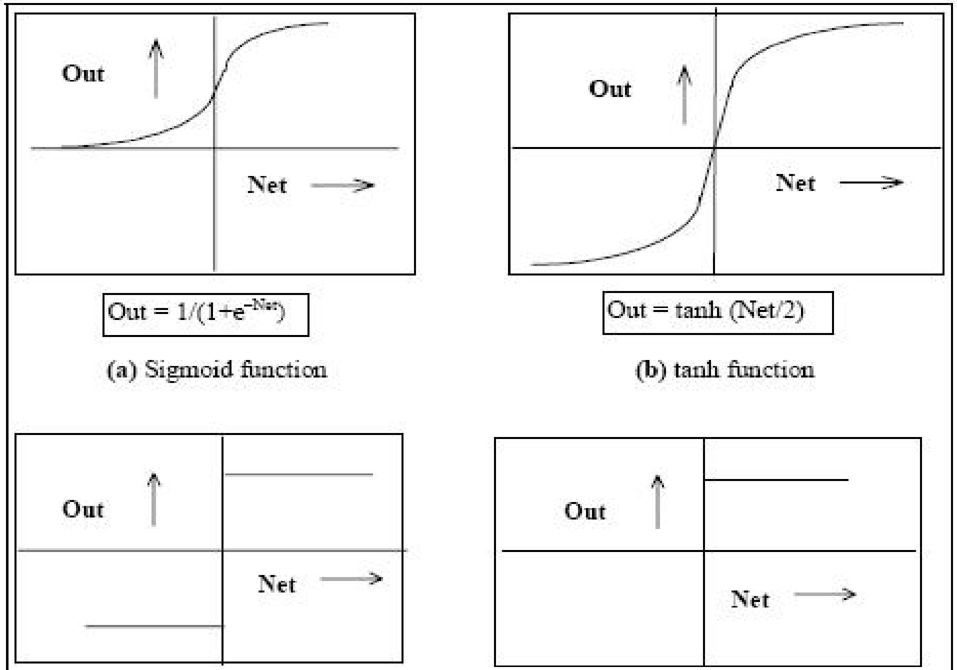
- 2nd: the Piecewise-Linear function. This function again can take on the values of 0 or 1, but can also take on values between that depending on the amplification factor in a certain region of linear operation.

$$\varphi(v)=\begin{cases}1 & v\geq \frac{1}{2}\\ v & -\frac{1}{2}>v>\frac{1}{2}\\ 0 & v\leq -\frac{1}{2}\end{cases}$$

- 3rd: the sigmoid function. This function can range between 0 and 1, but it is also sometimes useful to use the -1 to 1 range. An example of the sigmoid function is the hyperbolic tangent function.

$$\varphi(v)=\tanh\left(\frac{v}{2}\right)=\frac{1-\exp(-v)}{1+\exp(-v)}$$

Activation functions



$\text{Out} = 1/(1+e^{-\text{Net}})$

(a) Sigmoid function

$\text{Out} = \tanh(\text{Net}/2)$

(b) tanh function

Out = +1, Net > 0
= -1, Net < 0
= undefined, Net =0.

(c) Signum function

Out = 1, Net > 0
= 0, Net = 0
= undefined, Net < 0.

(d) Step function

Common non-linear functions used for synaptic inhibition. Soft non-linearity: (a) Sigmoid and (b) tanh; Hard non-linearity: (c) Signum and (d) Step.

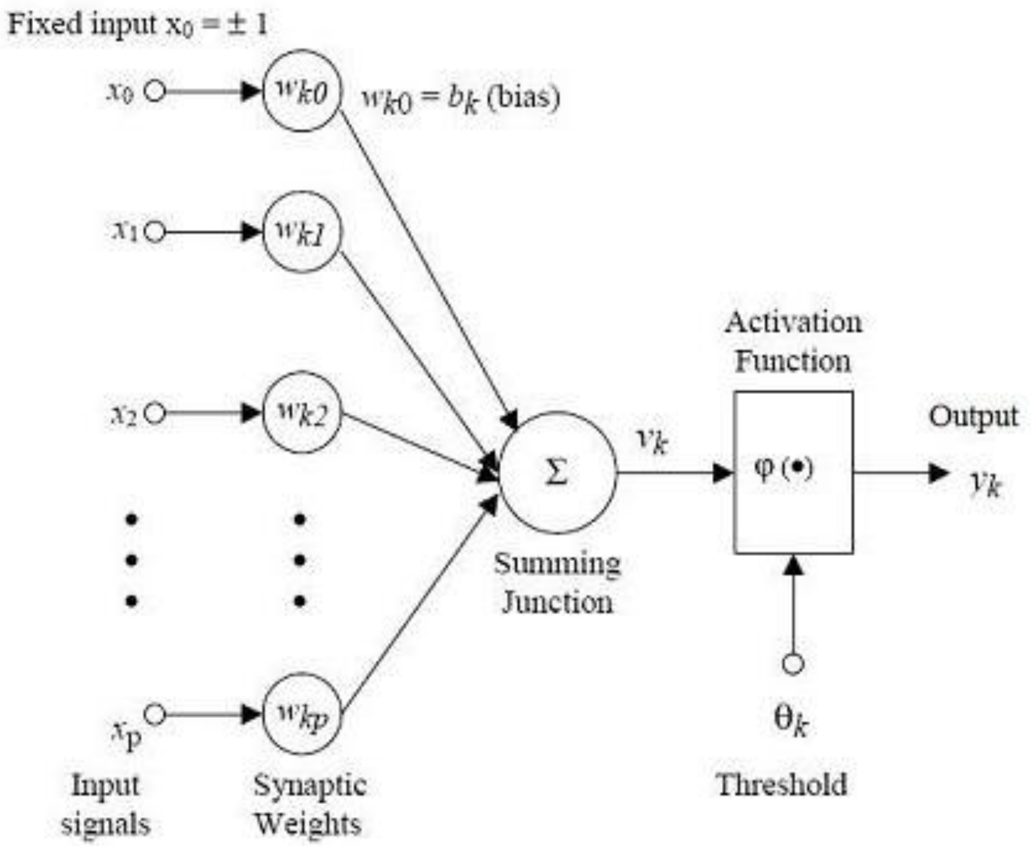
Net = threshold value (v)
Out = output value

The common artifcial neural networks are all variations on the parallel distributed processing (PDP) idea. The architecture of each neural network is based on very similar building blocks which perform the processing.

framework for distributed representation

- An ANN consists of a pool of simple processing units which communicate by sending signals to each other over a large number of weighted connections. A set of major aspects of a parallel distributed model can be distinguished:
- a set of processing units ('neurons,' 'cells');
- a state of activation for every unit, which equivalent to the output of the unit;
- connections between the units. Generally each connection is defined by a weight wjk which determines the effect which the signal of unit j has on unit k;
- an activation function, which determines the new level of activation based on the input and the current activation (i.e., the update);
- a method for information gathering (the learning rule);
- an environment within which the system must operate, providing input signals and|if necessary|error signals.

framework for distributed representation

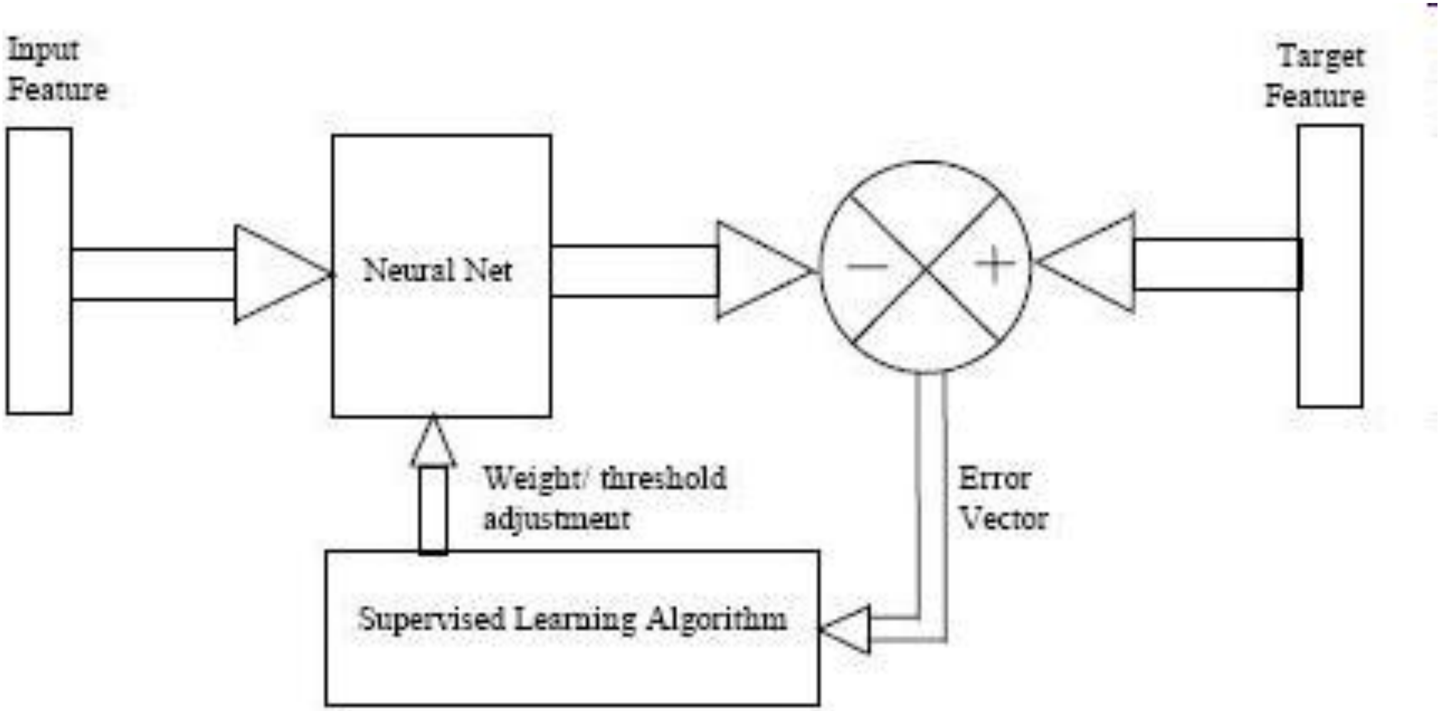


Processing units

- Each unit performs a relatively simple job: **receive input** from neighbours or external sources and use this to **compute an output** signal which is propagated to other units.
- Apart from this processing, a **second task is the adjustment of the weights**. The system is inherently parallel in the sense that many units can carry out their computations at the same time.
- Within neural systems it is useful to distinguish three types of units:
  - input units which receive data from outside the neural network,
  - output units which send data out of the neural network,
  - and hidden units whose input and output signals remain within the neural network.
- During operation, units can be updated either synchronously or asynchronously. With synchronous updating, all units update their activation simultaneously; with asynchronous updating, each unit has a (usually fixed) probability of updating its activation at a time t

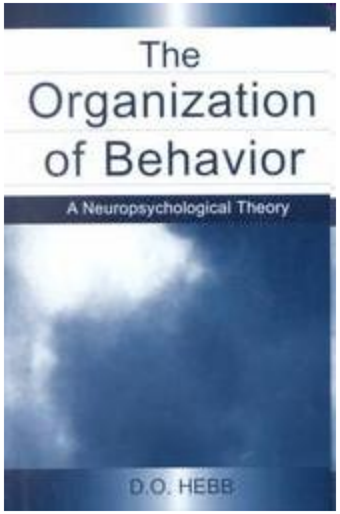
Training of artifcial neural networks

- A **neural network** has to be configured such that the application of a set of inputs produces (either 'direct' or via a relaxation process) the desired set of outputs. Various methods to set the strengths of the connections exist. One way is to set the weights explicitly, using a priori knowledge. Another way is to **'train' the neural network** by feeding it teaching patterns and letting it change its weights according to some learning rule.



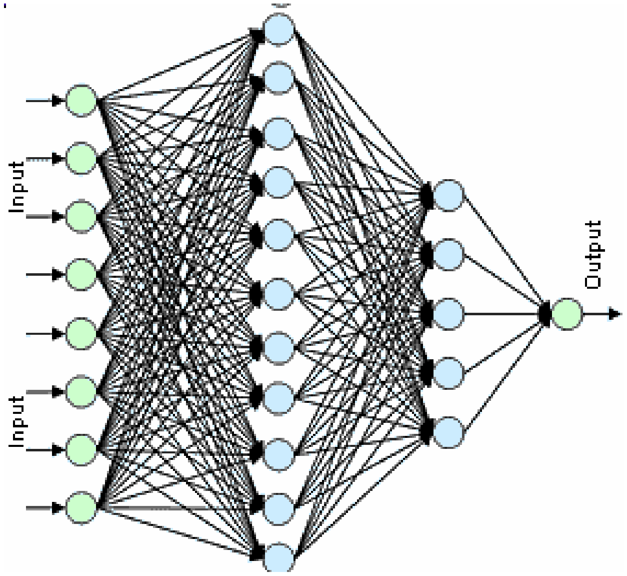
Modifying patterns of connectivity of Neural Networks

- Both learning paradigms **supervised learning and unsupervised learning result** in an adjustment of the weights of the connections between units, according to some modification rule. Virtually all learning rules for models of this type can be considered as a variant of the Hebbian learning rule suggested by Hebb in his classic book Organization of Behaviour (1949) (Hebb, 1949). The basic idea is that if two units j and k are active simultaneously, their interconnection must be strengthened.



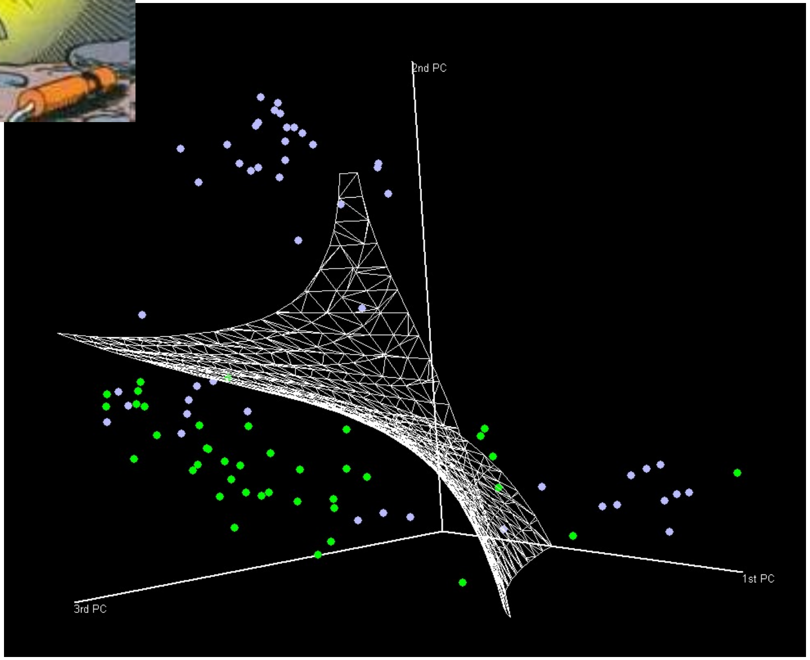
ANN drawbacks

- Designer chooses the network topology, the performance function, the learning rule, and the criterion to stop the training phase, but the system automatically adjusts the parameters.
- So, it is difficult to bring a priori information into the design, and when the system does not work properly it is also hard to incrementally refine the solution.
- Massive parallelism
- Minmal tolerance required
- Lack of data
- **Can get caught in local minima →**



Improvement

- → **Better apply support vector machines**



First applied ANN

- Feed forward backpropagation NN
- trainLM (training function) trainlm is a network training function that updates weight and bias values according to Levenberg-Marquardt optimization. Validation vectors are used to stop training early if the network performance on the validation vectors fails to improve or remains the same for max_fail epochs in a row. Test vectors are used as a further check that the network is generalizing well, but do not have any effect on training.
- LEARNGDM (learning function) learngdm calculates the weight change dW for a given neuron from the neuron's input P and error E, the weight (or bias) W, learning rate LR, and momentum constant MC, according to gradient descent with momentum
- MSE performance fct. It measures the network's performance according to the mean of squared errors.
- Tansig (transfer fct) Hyperbolic tangent sigmoid transfer function

First applied ANN

Mat. No.	INPUT p MPa	INPUT v mm/s	OUTPUT cof	St.Dev 1	ws 1 mm3/mN	St.Dev mm3/mN
1	1	1	0.46	0.04	0.0000715	0.0000171
2	1	1	0.55	0.02	0.0000542	0.00000972
3	1	1	43	0.09	0.00036	0.000144
4	1	1	0.4	0.02	0.000295	0.000109
5	1	1	0.4	0.01	0.000376	0.0000198
6	1	1	0.59	0.02	0.000000749	0.0000000676
6	1	3	0.5	0.02	0.00000114	0.0000000997
6	2	1	0.27	0.07	0.00000102	0.000000122
6	2	3	0.32	0.04	0.00000103	0.000000891
6	3	1	0.22	0.05	0.000000776	0.0000000838
6	3	3	0.2	0.01	0.00000137	0.000000102
6	4	1	0.2	0.02	0.000000618	0.0000000812
7	1	1	0.37	0.02	0.0000007	0.0000000806
7	1	3	0.15	0.01	0.00000059	0.0000000513
7	2	1	0.24	0.02	0.000000586	0.0000000386
7	2	3	0.12	0.02	0.000000819	0.00000000951
7	3	1	0.15	0.03	0.000000488	0.000000184
7	3	3	0.14	0.03	0.000000694	0.0000000249
7	4	1	0.13	0.01	0.00000219	0.000000209

A1 + A2

Mat. No.	Density g/cm³	Microhardness Mpa	St.Dev. 1	Impact stren kJ/m²	St.Dev. 1
1	1,354	206,99	3,63	3,38	1,2
2	1,373	212,88	0,98	3,36	0,85
3	1,418	225,63	4,22	2,53	0,9
4	1,464	218,76	3,73	2,6	0,79
5	1,492	216,8	5,69	2,66	0,89
6	1,386	239,36	10,1	3,47	0,29
6	1,386	239,36	10,1	3,47	0,29
6	1,386	239,36	10,1	3,47	0,29
6	1,386	239,36	10,1	3,47	0,29
6	1,386	239,36	10,1	3,47	0,29
6	1,386	239,36	10,1	3,47	0,29
6	1,386	239,36	10,1	3,47	0,29
7	1,414	251,14	13,64	3,11	0,51
7	1,414	251,14	13,64	3,11	0,51
7	1,414	251,14	13,64	3,11	0,51
7	1,414	251,14	13,64	3,11	0,51
7	1,414	251,14	13,64	3,11	0,51
7	1,414	251,14	13,64	3,11	0,51
7	1,414	251,14	13,64	3,11	0,51

A3 INPUT

Mat. No.	E_t Mpa	St.Dev. 1	σ_M MPa	St.Dev. 1	ε_M %	St.Dev. 1	Mat. No.	E_c MPa
1	3741,57	103,12	86,8	0,63	2,37	0,13	1	
2	3631,04	133,99	74,8	8,04	1,98	0,28	2	
3	3852,16	129,73	64,34	4,14	1,59	0,12	3	
4	3744,17	130,83	72,89	6,17	1,87	0,19	4	
5	3795,29	277,46	68,04	4,64	1,72	0,18	5	
6	4777	340,68	65,97	3,43	1,61	0,12	6	
6	4777	340,68	65,97	3,43	1,61	0,12	6	
6	4777	340,68	65,97	3,43	1,61	0,12	6	
6	4777	340,68	65,97	3,43	1,61	0,12	6	
6	4777	340,68	65,97	3,43	1,61	0,12	6	
6	4777	340,68	65,97	3,43	1,61	0,12	6	
6	4777	340,68	65,97	3,43	1,61	0,12	6	
7	5401,91	379,61	65,12	3,53	1,48	0,08	7	
7	5401,91	379,61	65,12	3,53	1,48	0,08	7	
7	5401,91	379,61	65,12	3,53	1,48	0,08	7	
7	5401,91	379,61	65,12	3,53	1,48	0,08	7	
7	5401,91	379,61	65,12	3,53	1,48	0,08	7	
7	5401,91	379,61	65,12	3,53	1,48	0,08	7	
7	5401,91	379,61	65,12	3,53	1,48	0,08	7	

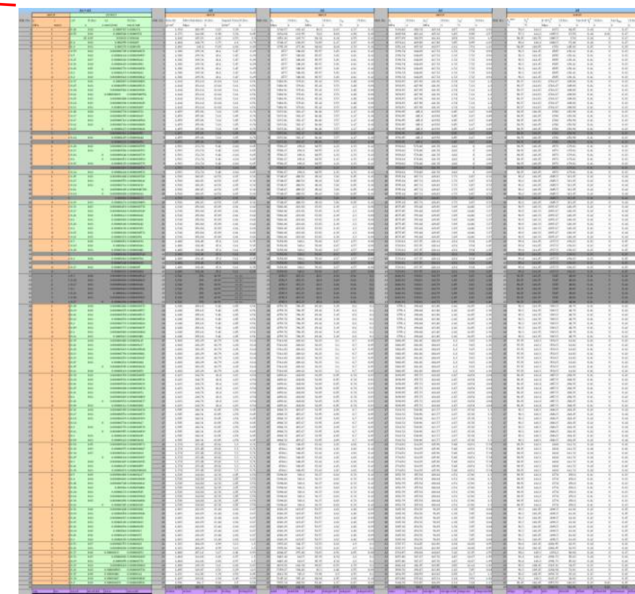
A4 INPUT

	A	B	C	D	E	F
	[Vol.%]	[Vol.%]	[Vol.%]	[Vol.%]	[Vol.%]	[Vol.%]
1	100	0	0	0	0	0
2	99	0	1	0	0	0
3	97	0	3	0	0	0
4	95	0	5	0	0	0
5	93	0	7	0	0	0
6	85	15	0	0	0	0
7	84	15	1	0	0	0
8	82	15	3	0	0	0

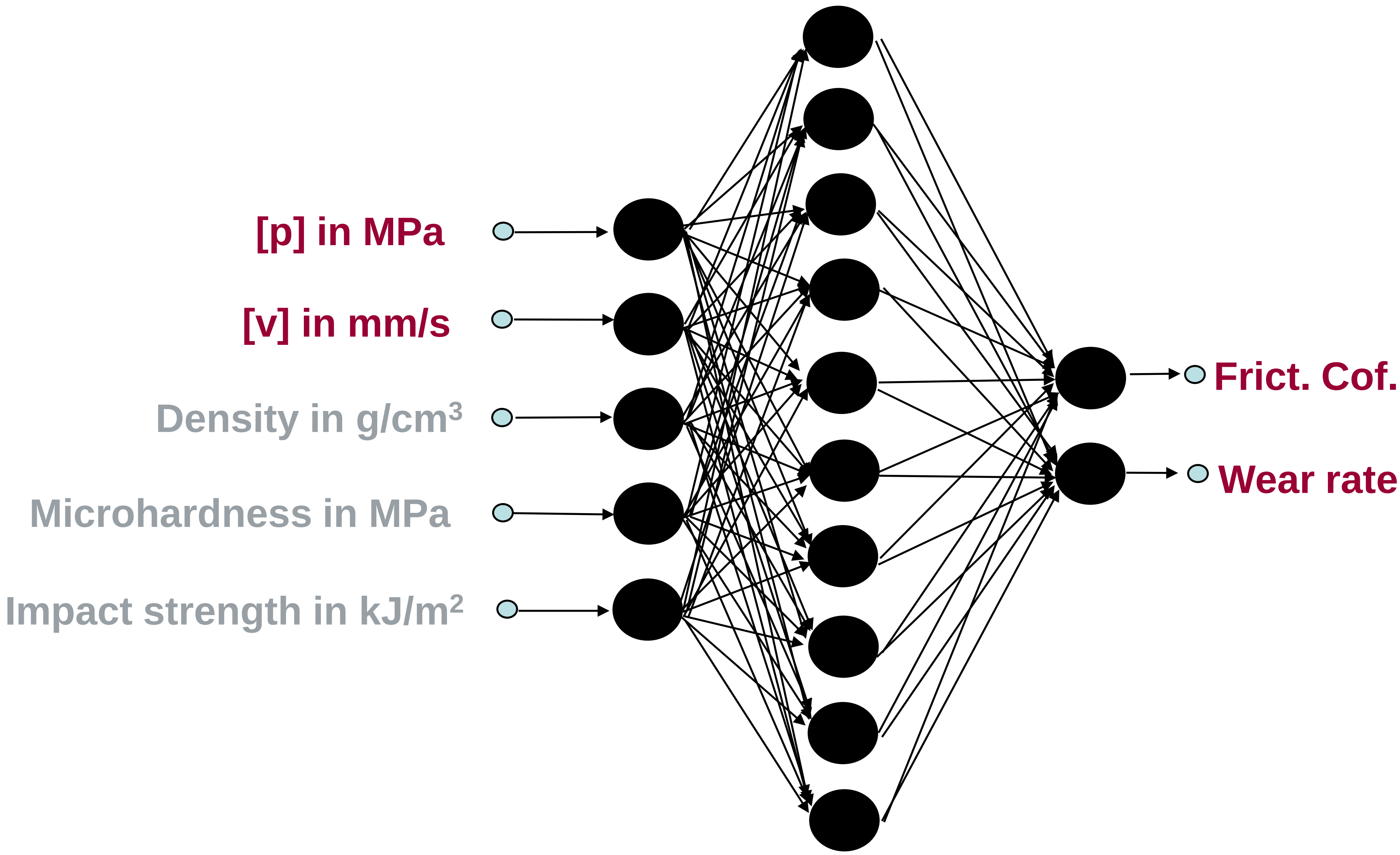
Ingredients in %.

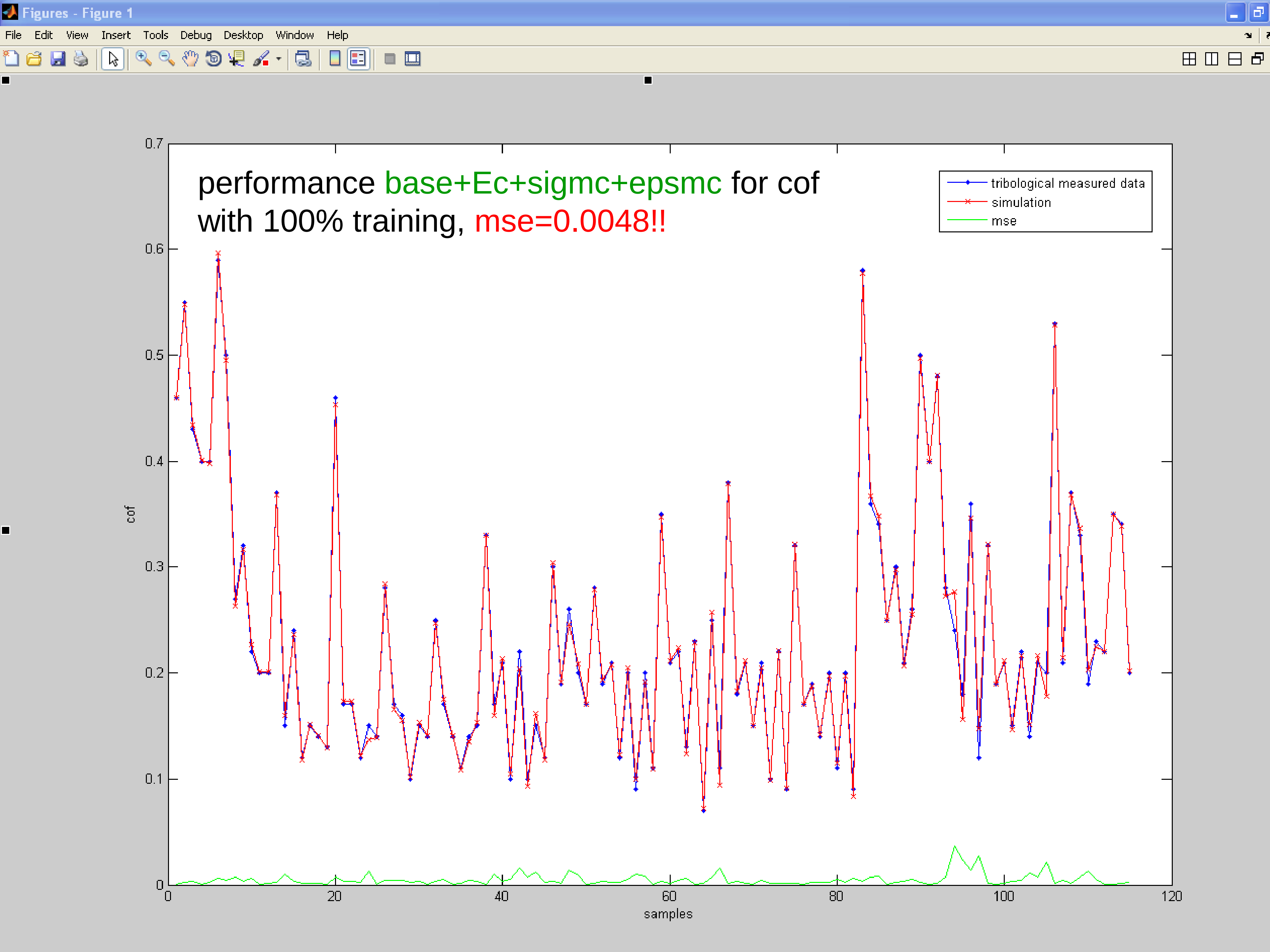
⋮

Material-no.

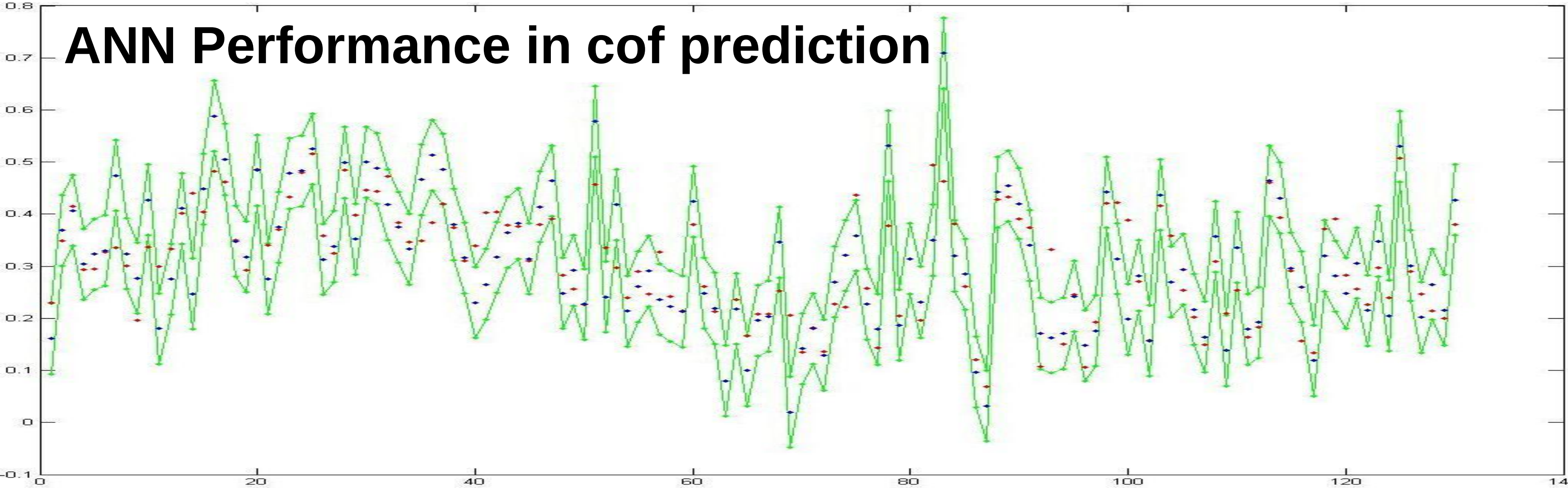


First applied ANN




Figures - Figure 1
File Edit View Insert Tools Debug Desktop Window Help
performance base+Ec+sigmc+epsmc for cof
with 100% training, mse=0.0048!!
tribological measured data
simulation
mse
cof
samples
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
20
40
60
80
100
120

Sim new PEEK cof data from KL (L.Josch)

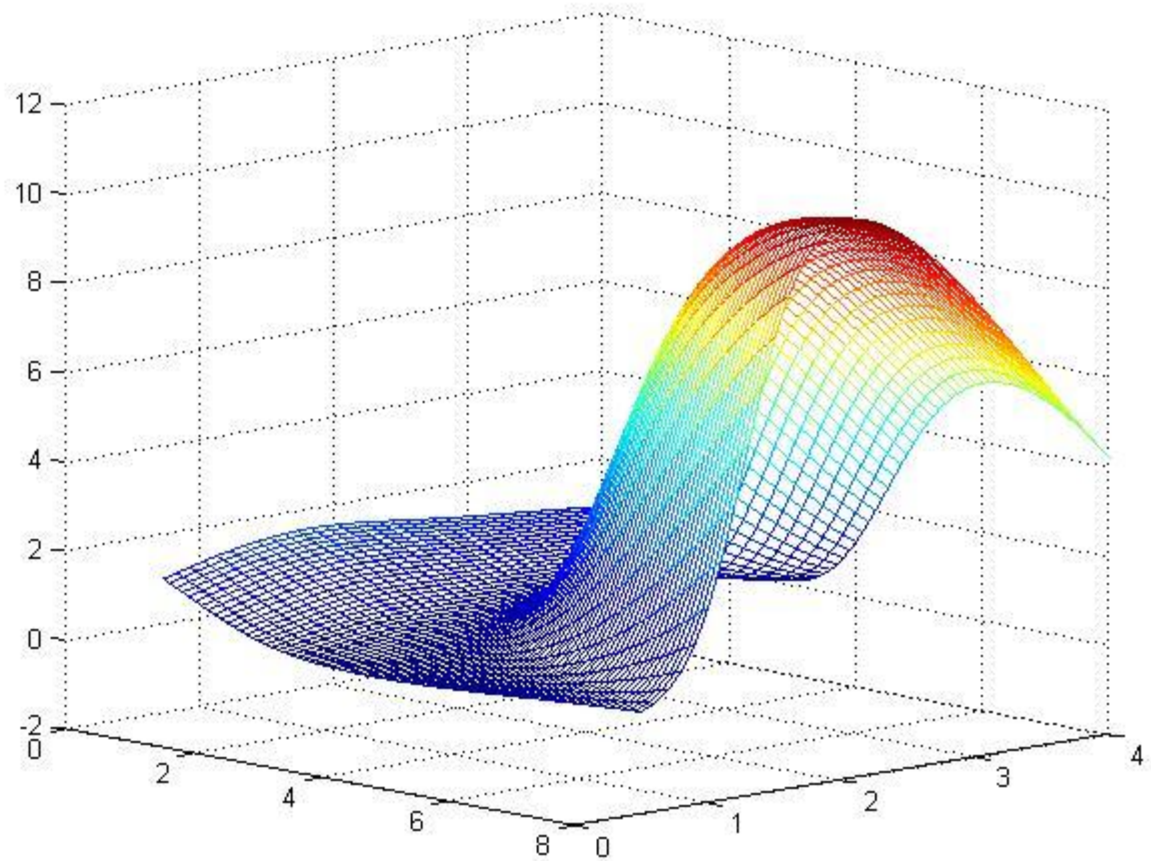


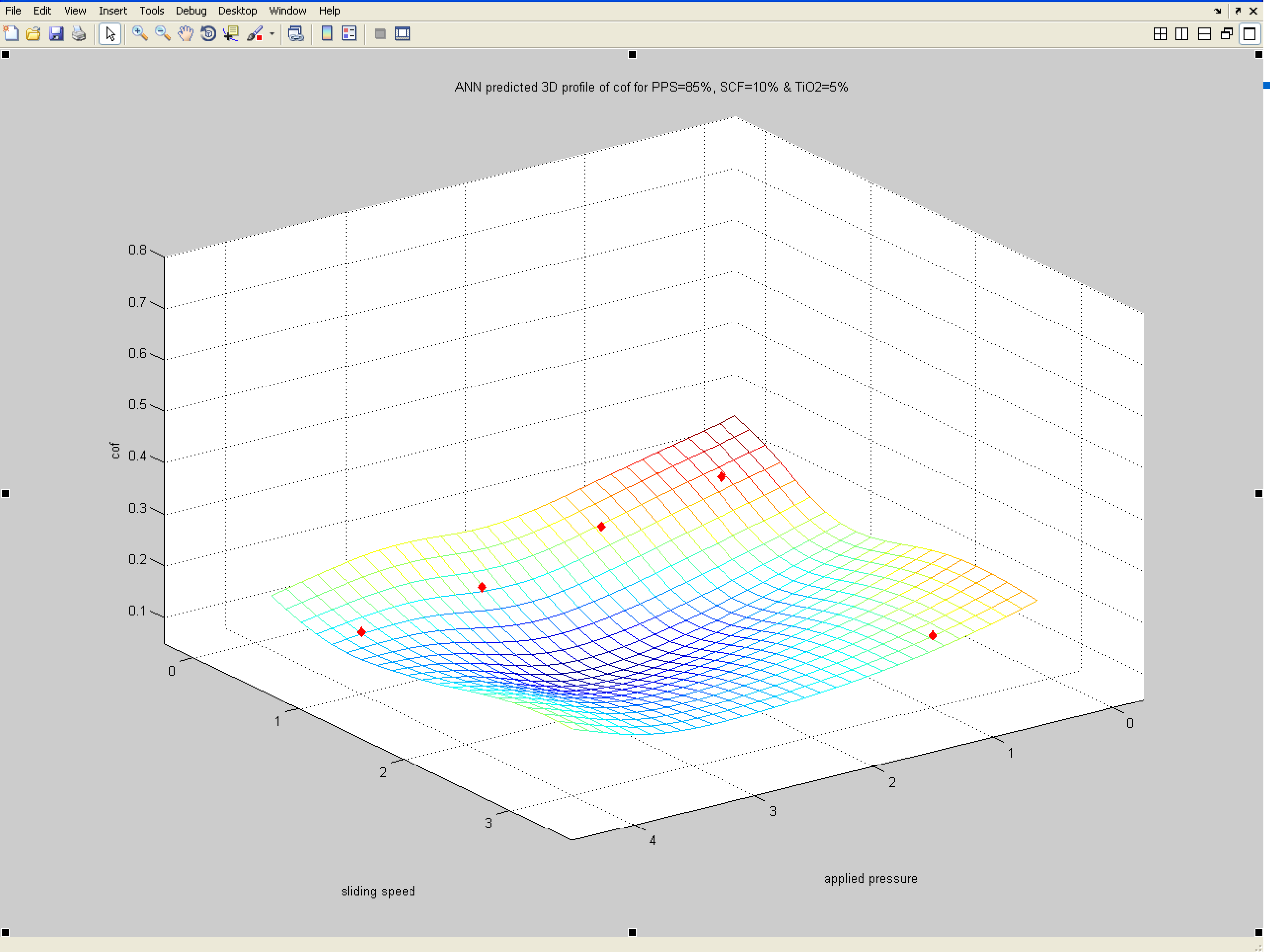
Blue= measured cof data

Green = MAE range (+/-0.0678)

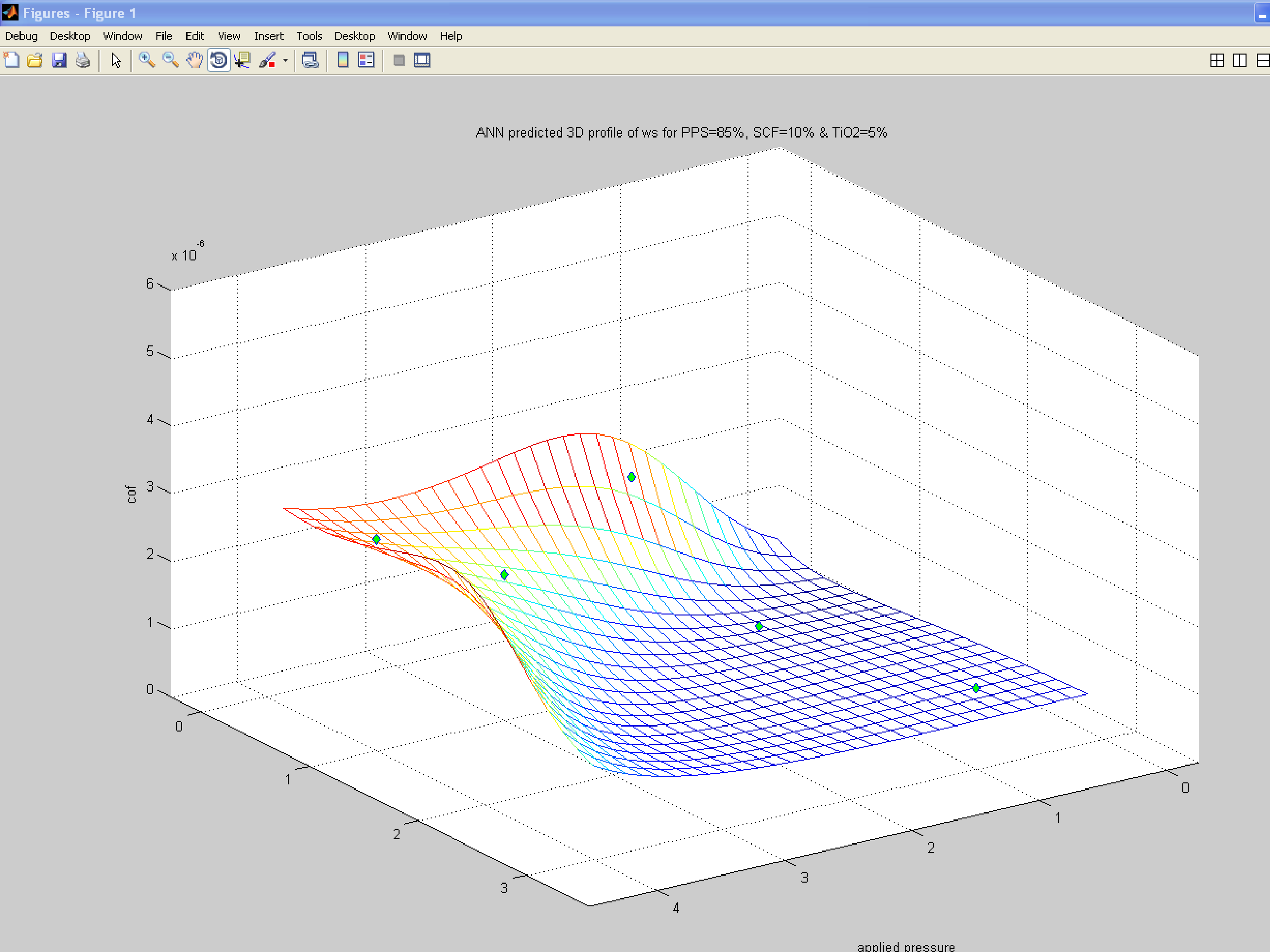
Red = simulated cof

Error higher due to just 5x 8MPa pressures in database of 130

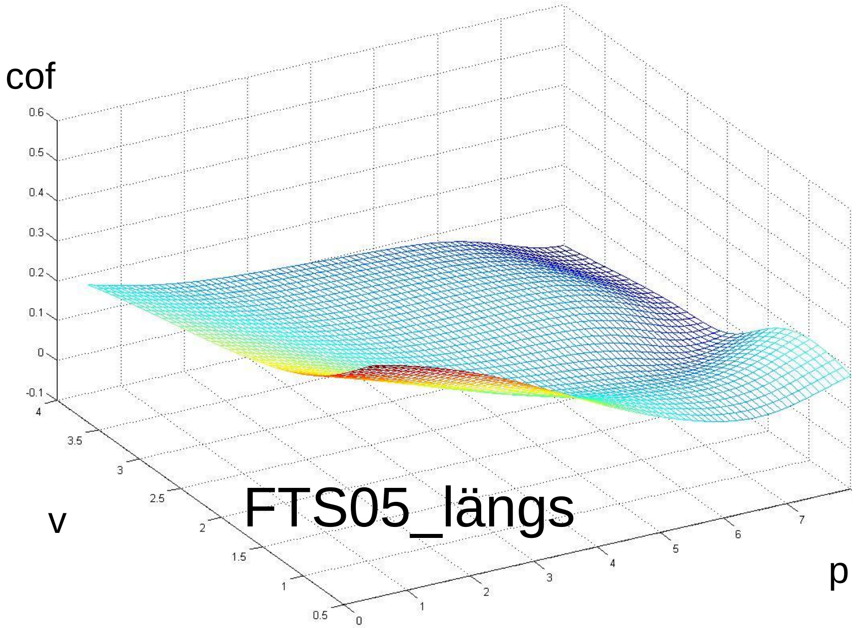

12
10
8
6
4
2
0
-2
0
2
4
6
8
0
1
2
3
4

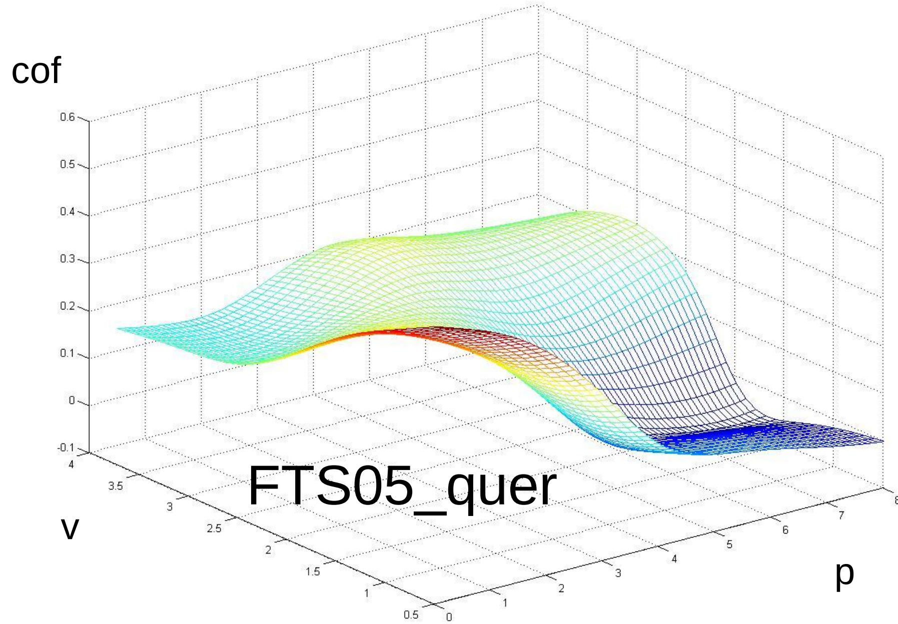


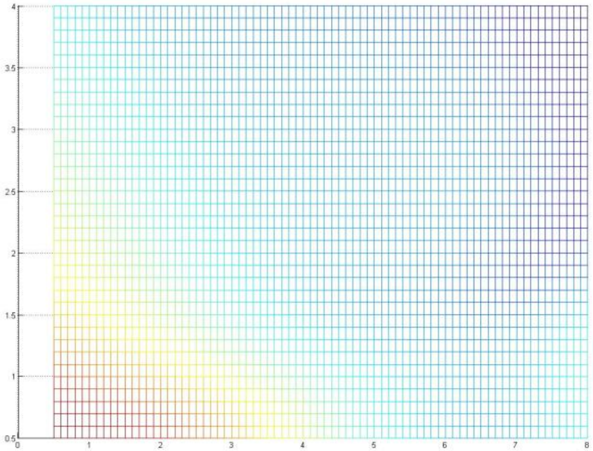
File Edit View Insert Tools Debug Desktop Window Help
ANN predicted 3D profile of cof for PPS=85%, SCF=10% & TiO2=5%
cof
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
sliding speed
0
1
2
3
4
applied pressure

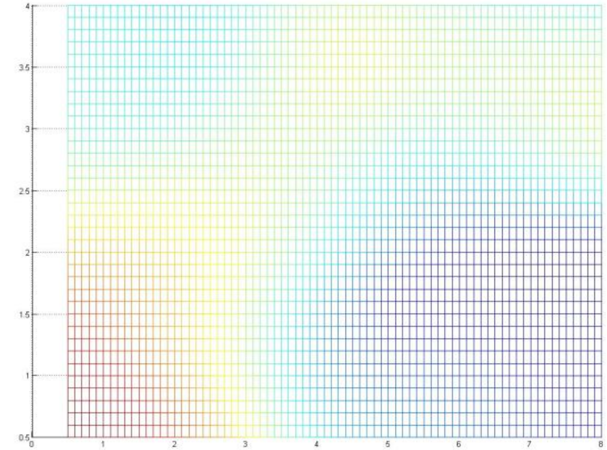

Figures - Figure 1
Debug Desktop Window File Edit View Insert Tools Desktop Window Help
ANN predicted 3D profile of ws for PPS=85%, SCF=10% & TiO2=5%
x 10^-6
6
5
4
3
2
1
0
cof
0
1
2
3
4
3
2
1
0
applied pressure

Sim new PEEK cof FTS05

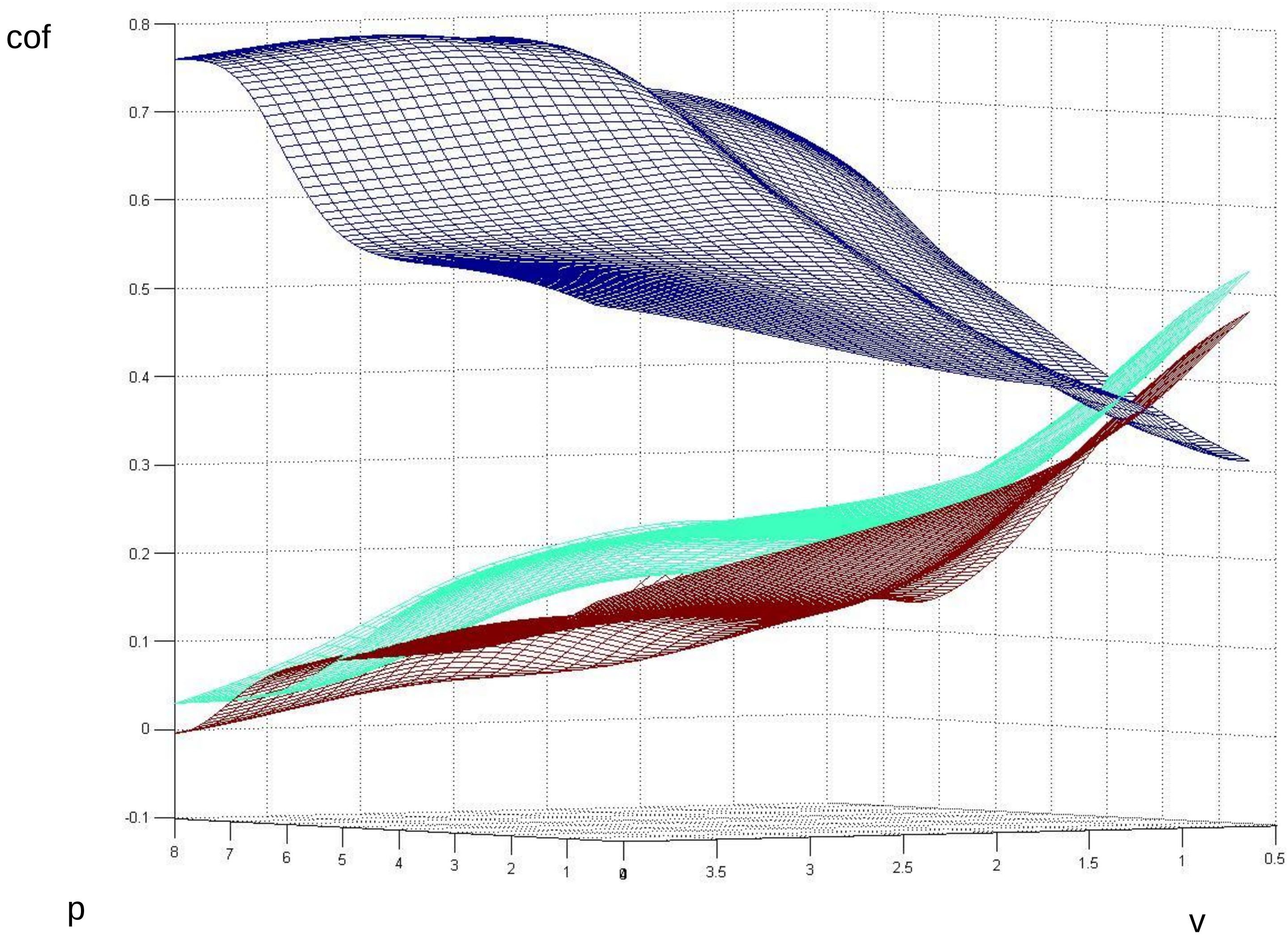








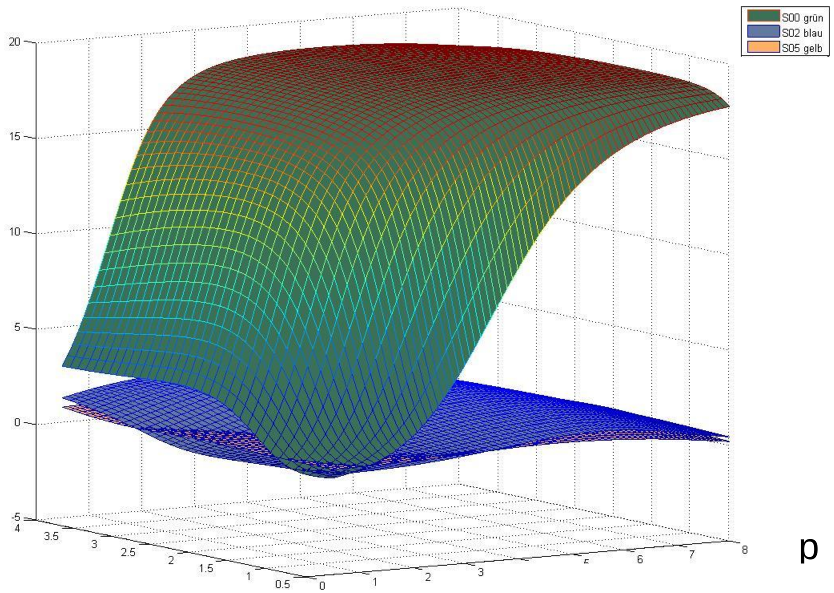
Sim cof S00 vs. S02 vs. S05, längs



Blue = S00
Red = S02
Green = S05

Sim ws S00 vs. S02 vs. S05, längs

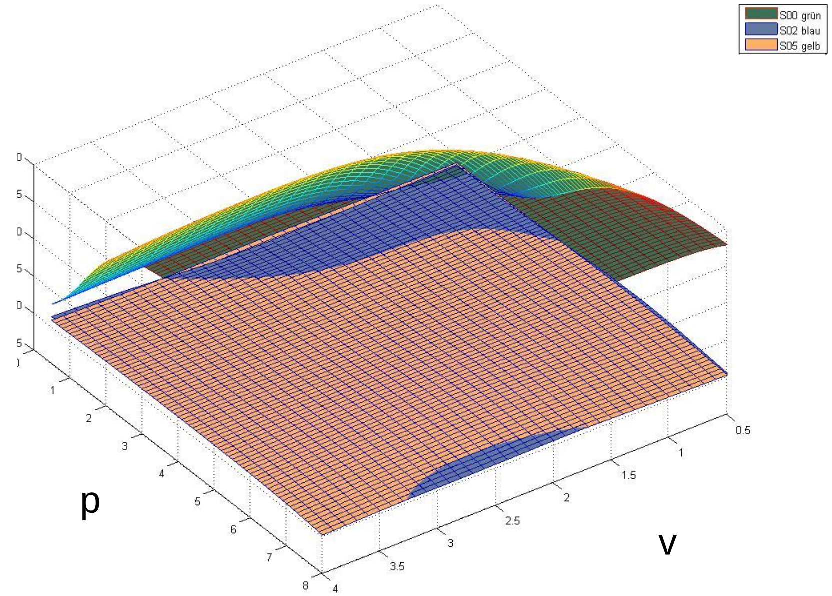
ws



v

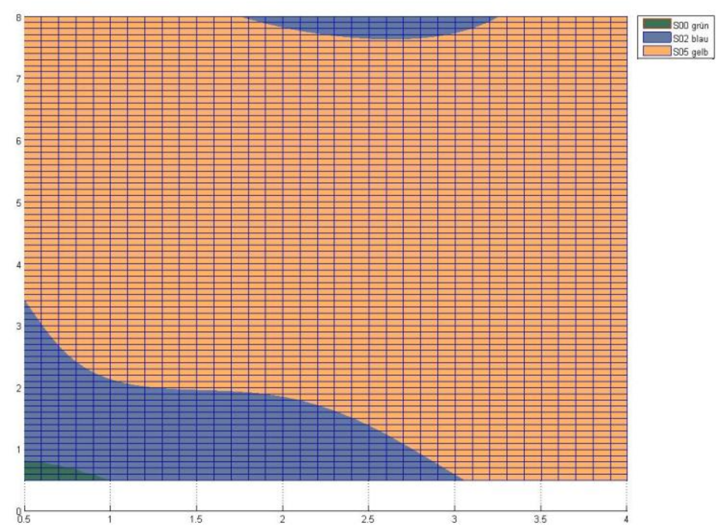
p

ws

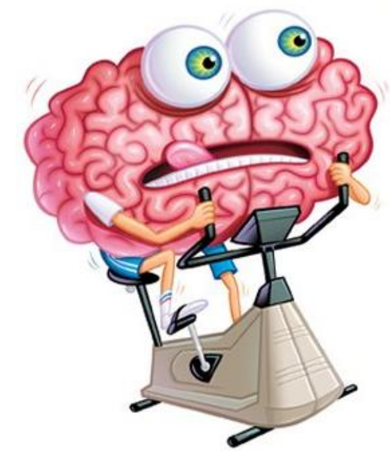


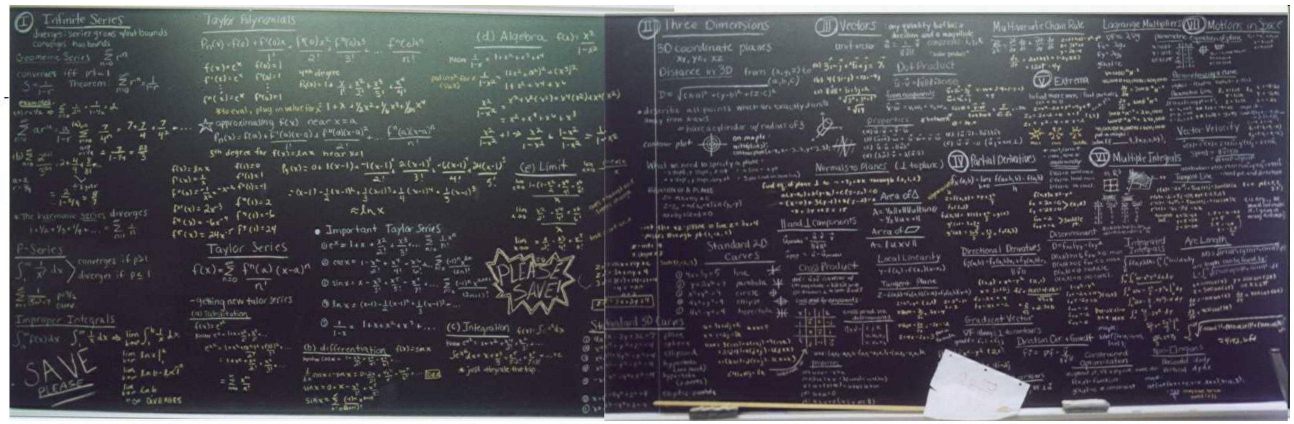
p

v



Thank you for your attention





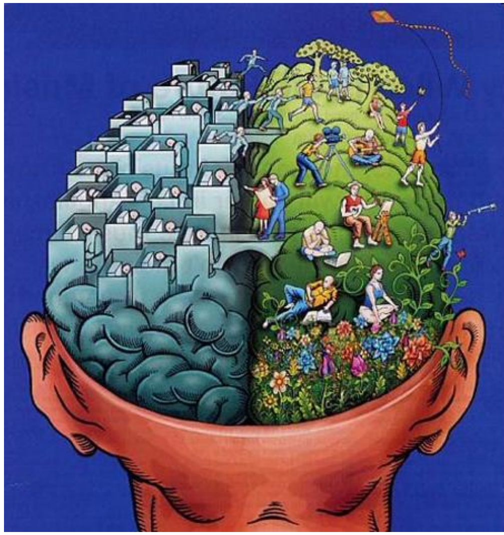
Questions ?

Fragen ?

Preguntas ?

Domande ?

?



References

- **Ben Kröse & Patrick van der Smagt(1996)
An introduction to Neural Networks
URL:http://www.fwi.uva.nl/research/neuro/
URL:http://www.op.dlr.de/FF-DR-RS/**
- **Principe, J.C. "Artificial Neural Networks"(extract)
The Electrical Engineering Handbook
Ed. Richard C. Dorf
Boca Raton: CRC Press LLC, 2000**
- **Konar, Amit.
Artificial intelligence and soft computing : behavioral and cognitive modeling of the human brain / Amit Konar.**
- **Joe Tebelskis (1995)
Speech Recognition using Neural Networks
School of Computer Science
Carnegie Mellon University**
- **Efstathios Kalyvas (2001)
USING NEURAL NETWORKS AND GENETIC ALGORITHMS TO PREDICT STOCK MARKET RETURNS**
- **James A. Freeman, David M. Skapura
Neural Networks Algorithms, Applications, and Programming Techniques
Addison-Wesley**
- **A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, J. Schmidhuber. A Novel Connectionist System for Improved Unconstrained Handwriting Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 5, 2009.**
- **S. Haykin. Neural Networks : A Comprehensive Foundation. Macmillan, New York, 1994.**
- Parts taken from **Neuro AI - Intelligent systems and Neural Networks**

Übungen

- Matlab: NN toolbox erforderlich
- Command window: nftool,
- Netzwerkarchitektur festlegen,
- Daten einlesen,
- ANN trainieren,
- Ergebnisse überprüfen/ Plots anfertigen
- Parameter ändern, Ergebnisse untereinander vergleichen