

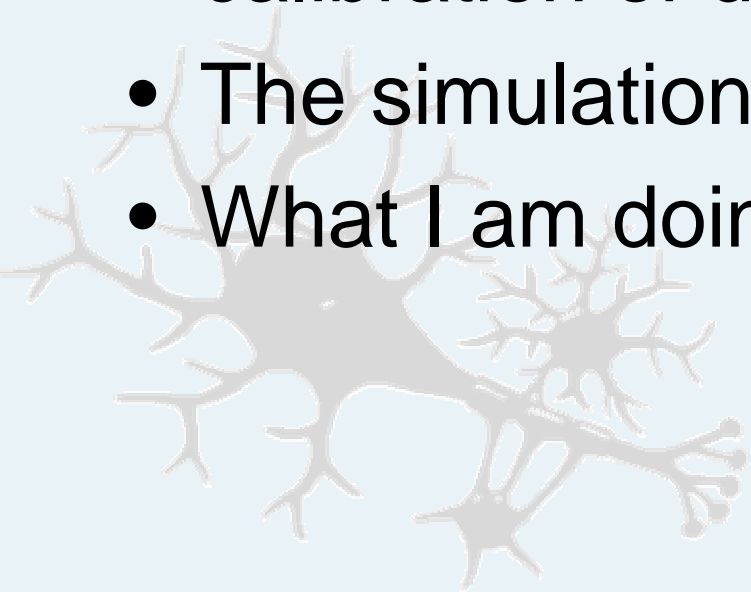


Development of a new calibration method based on neural networks for high energy detectors

A brief introduction to the thesis work

Summary

- The aims of the thesis
- How the neural networks can perform the calibration of detectors
- The simulation tool
- What I am doing





The aims of the thesis

- By leveraging the adaptive capabilities of the neural networks, to develop a new method for the calibration of a detector.
- To define the guidelines to carry out the new method in the future experiments:
 - Network topology
 - Training algorithms
 - Performance improving techniques
 - Simulation tool optimization
 - Integration with other applications

How the neural networks can perform the calibration of detectors

- The “classic” method: the energy of the particle shower is determined by the weighted sum of the equalized signals.
- There are different limitations. In fact by the classic method we can not consider the dependence on:
 - Particle energy
 - Interaction vertex position

How the neural networks can perform the calibration of detectors

- The neural networks are adaptable systems that can learn relationships through repeated presentation of data;
- After a suitable training, the NN can generalize to new, previously unseen data.
- A NN can measure the energy of the particles starting from the signals of the calorimeter.



The simulation tool

- I carried out a research to identify the best NN simulation tool for our purposes. The research was based on following sources:
 - “Neural Computation” and “Neural Networks”, the most important technical papers in this field
 - The newsgroup named *comp.ai.neural-nets*
 - The articles available on web sites
 - The customer list and the references of the simulation tool

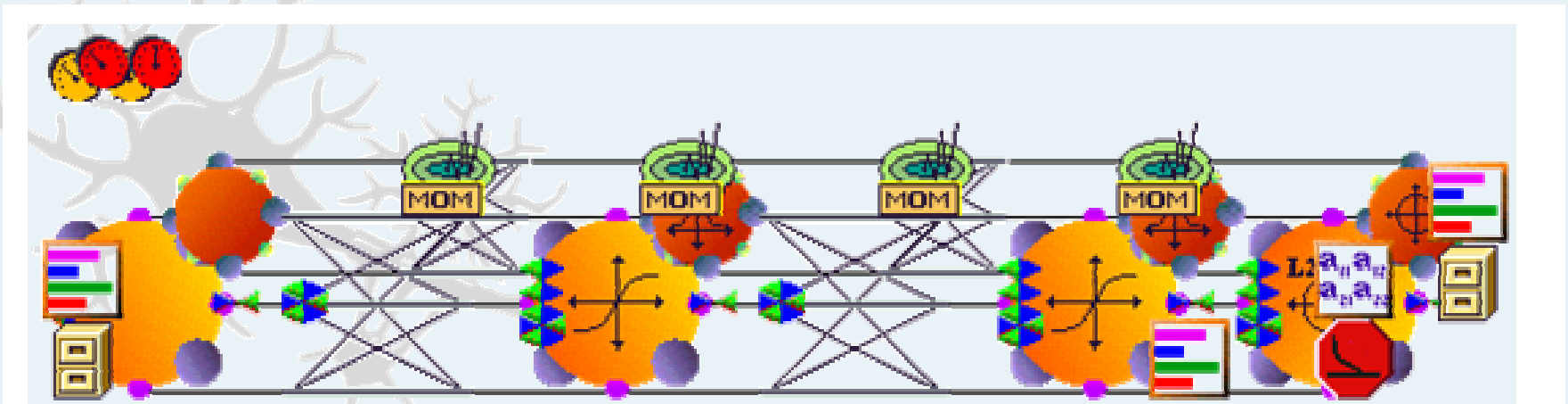
I picked out this software:

NeuroSolutions produced by NeuroDimension Inc.

NeuroSolutions features

GUI

- NeuroSolutions adheres to the so-called local additive model. Under this model, each component can activate and learn using only its own weights and the activations of its neighbors.
- This lends itself very well to the object-oriented modeling
- This in turn allows for a graphical user interface (GUI) with icon-based construction of networks.
- By allowing you to arbitrarily interconnect these components, a virtually infinite number of network architectures are possible





NeuroSolutions features

- **Wide set of topologies supported**

Below are the most common architectures NeuroSolutions supports.

- Multilayer Perceptron (MLP)
- Modular Neural Network
- Principal Component Analysis Network (PCA)
- General Regression Neural Network (GRNN)
- Self-Organizing Map Network (SOM)
- Generalized Recurrent Network
- Support Vector Machine
- Generalized Feed Forward
- Jordan/Elman Network
- Radial Basis Function Network (RBF)
- Probabilistic Neural Network (PNN)
- Time-Lag Recurrent Network (TLRN)
- CANFIS Network (Fuzzy Logic)
- Learning Vector Quantization (LVQ)

User-defined Neural Topologies

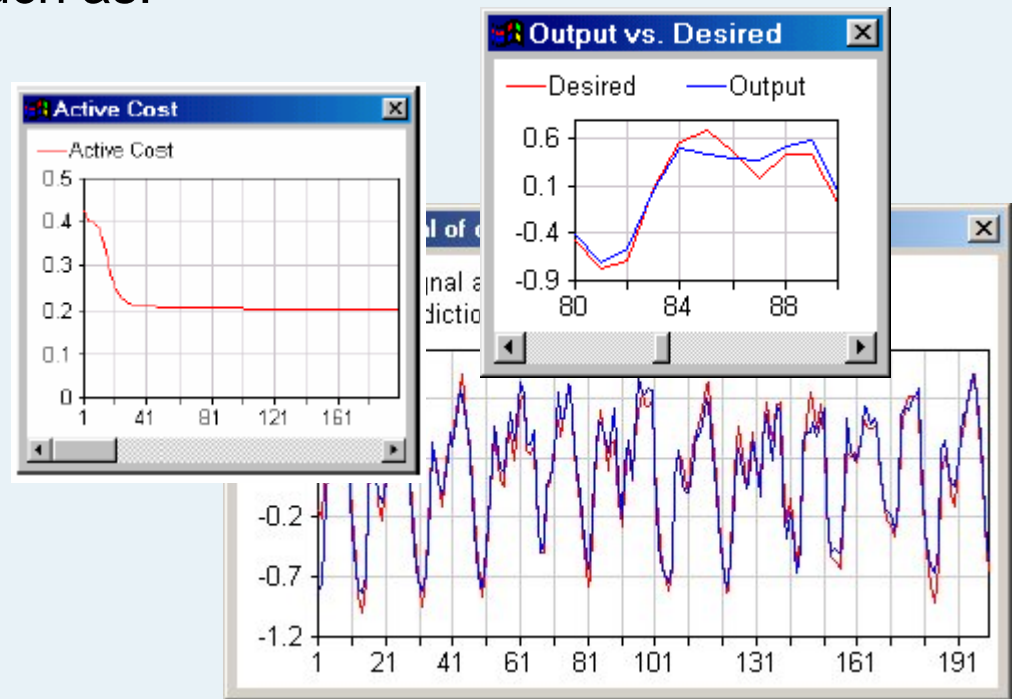
NeuroSolutions is based on the concept that neural networks can be broken down into a fundamental set of neural components that are relatively simplistic, but several components connected together can result in networks capable of solving very complex problems. ***A virtually infinite number of neural models are possible!***

NeuroSolutions features

Extensive Probing Capabilities

Neural networks are often criticized as being a "black box" technology. With NeuroSolutions' extensive and versatile set of probing tools. Probes provide you with real-time access to all internal network variables, such as:

- **Inputs/Outputs**
- **Weights**
- **Errors**
- **Hidden States**
- **Gradients**
- **Sensitivities**



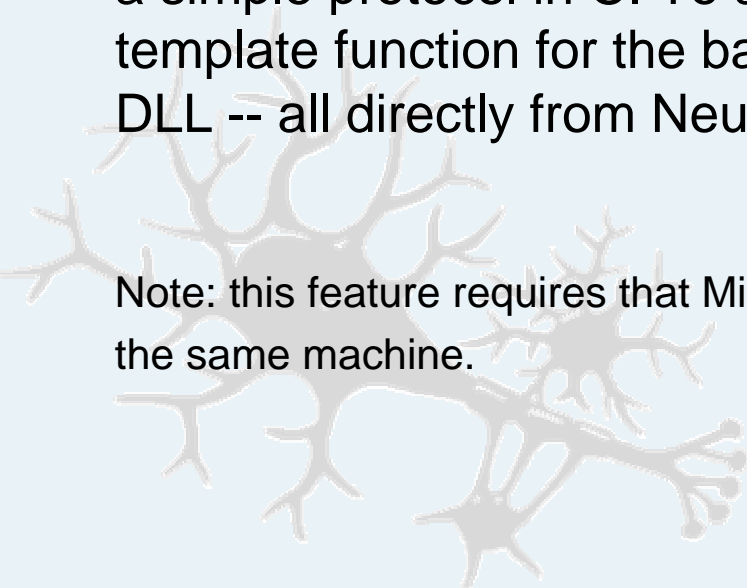


NeuroSolutions features

User-defined Neural Components

- NeuroSolutions allows you to integrate your own algorithms into the simulation environment through dynamic link libraries (DLLs).
- Every NeuroSolutions component implements a function conforming to a simple protocol in C. To add a new component you simply modify the template function for the base component and compile the code into a DLL -- all directly from NeuroSolutions!

Note: this feature requires that Microsoft Visual C++ (version 5.0 or higher) be installed on the same machine.





NeuroSolutions features

Deploying a Neural Network

Code Generation

NeuroSolutions allows you to automatically generate C++ source code for your neural network. This gives you the flexibility to customize the neural network code for your particular application (if necessary). Since the generated code is ANSI-compliant, you can deploy your neural network solution to other platforms such as UNIX (requires the Source Code License product).

DLL Generation

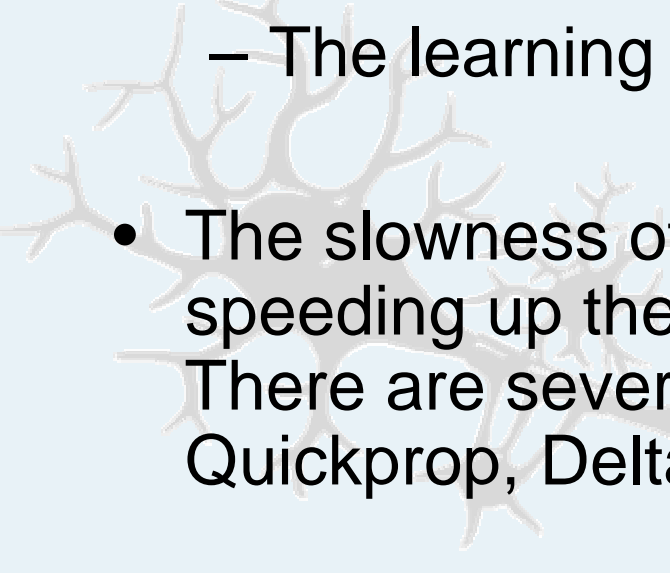
The Custom Solution Wizard is an add-in that will take a neural network designed within NeuroSolutions and encapsulate it into a dynamic link library (DLL) that conforms to a simple protocol. This DLL can then be embedded into your own C++, Visual Basic, Excel, Access or Internet (ASP) application. The key advantage to this approach is that you don't need to be an advanced programmer to use it.

What am I doing?

- **Hybrid network topology.** Our network must carry out a real measure. So that it must provide a linear response. We have a single output node (linear). I am investigating the number and the disposition of the other linear nodes.
- I am using the same training and validation data produced for the *Chorus* experiment (MonteCarlo events for pions with energies of 5, 10, 15, 20, 25, 30 e 40 GeV)
- I am investigating the solutions for the improvement of the NN performances



What am I doing?

- I would like to optimize the BackPropagation algorithm. Particularly I want to resolve the following problems:
 - The search for the optimal weight values can get caught in local minima
 - The BP algorithm is slow to converge
 - The learning rates must be set heuristically
 - The slowness of convergence can be improved by speeding up the original gradient descent learning. There are several faster search algorithms such as Quickprop, Delta, Momentum.
- 



The performance surface

