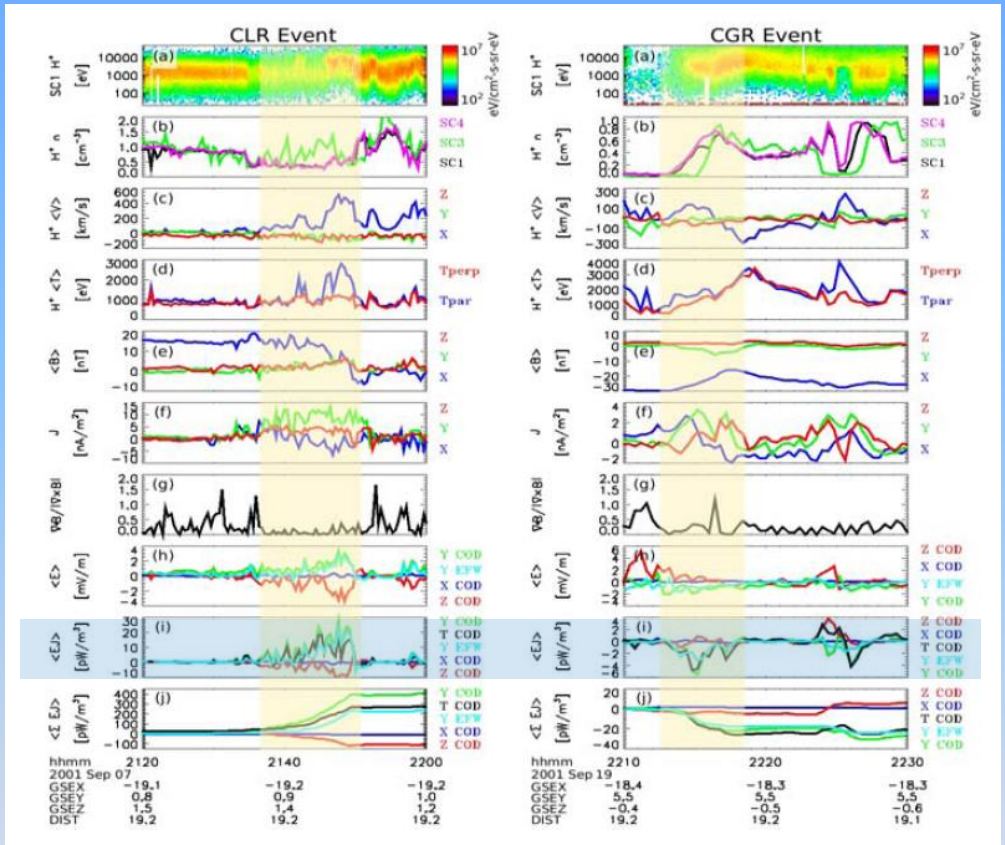
NEURAL NETWORK IDENTIFICATION OF ENERGY CONVERSION EVENTS

Vlad Constantinescu, Octav Marghitu
Institute for Space Sciences, Bucharest, Romania
contact : vlad@gpsm.spacescience.ro

Outline

- Introduction
  - Motivation : Energy Conversion Events
  - Feed-forward neural networks
  - Events encoding
- First approach (input divided in fixed-size intervals)
  - Network configuration
  - Results used real and synthetic data
- Second approach (sliding window)
  - Principle
  - Network configuration
  - Results using synthetic data
  - Results using real E·J data
  - Results on ion velocity data

Motivation – CLR and CGR events

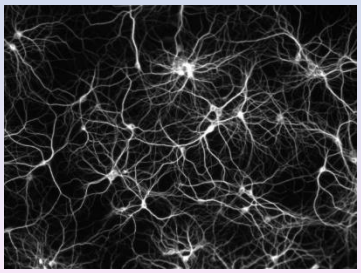


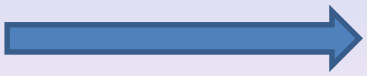
- ECR – energy conversion regions : $E \cdot J \neq 0$
- $E \cdot J < 0$ – generator regions (CGRs)
- $E \cdot J > 0$ – load regions (CLRs)

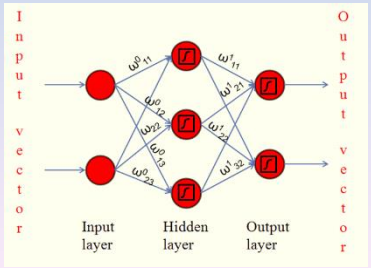
Hamrin, et al., 2009

Artificial Neural network

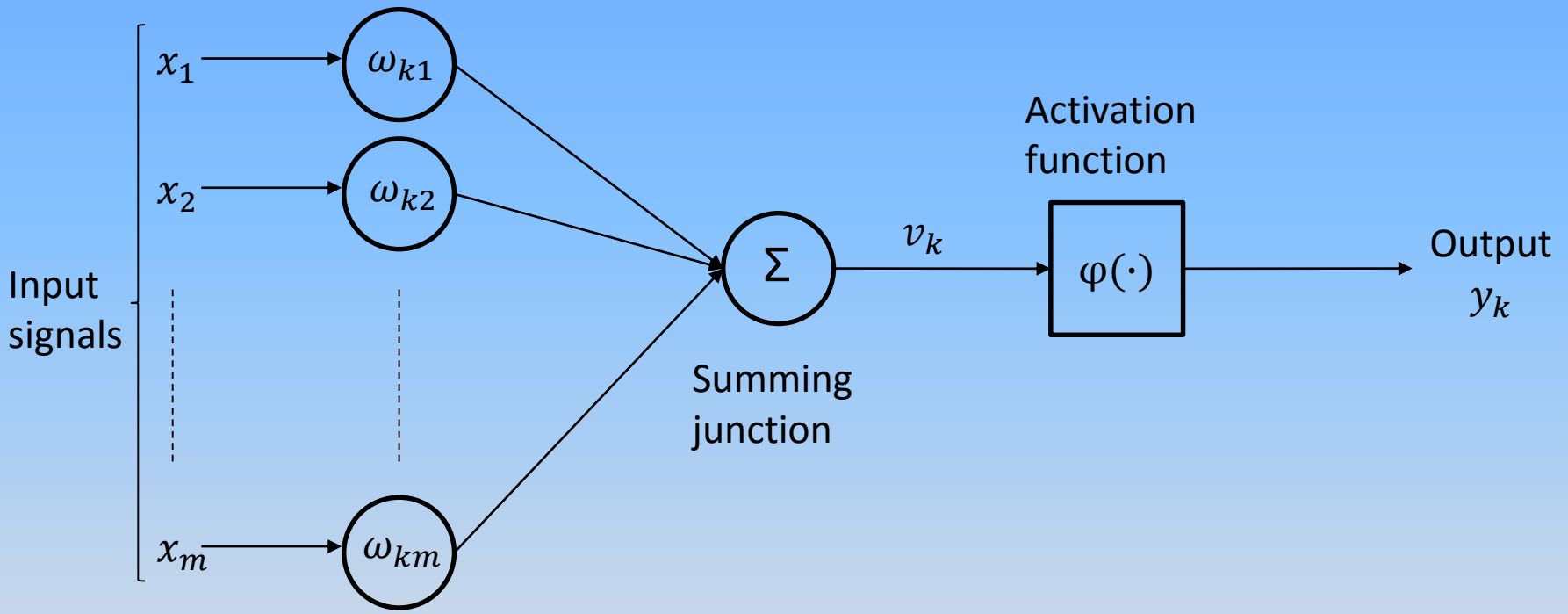
- A massively parallel distributed processor made up of simple processing units.
- It resembles the brain: knowledge is acquired through a learning process and it is stored in the interneuron connection strength
- Benefits : generalization, nonlinearity, input-output mapping, fault tolerance





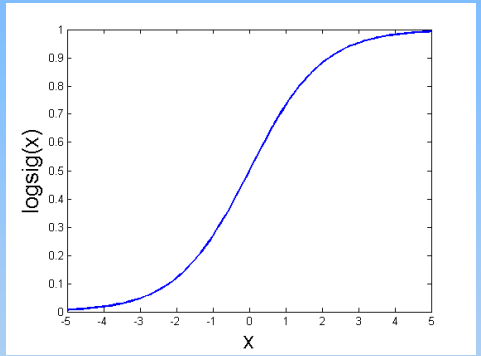


Artificial neuron model

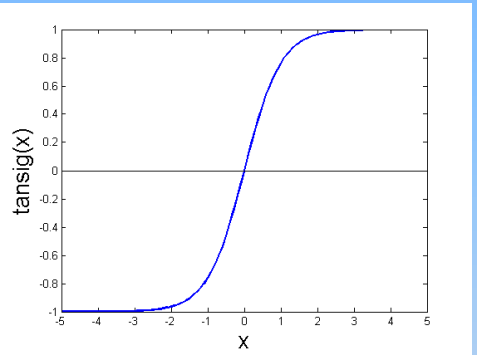


$$v_k = \sum_{j=1}^{m} \omega_{kj} \cdot x_j$$

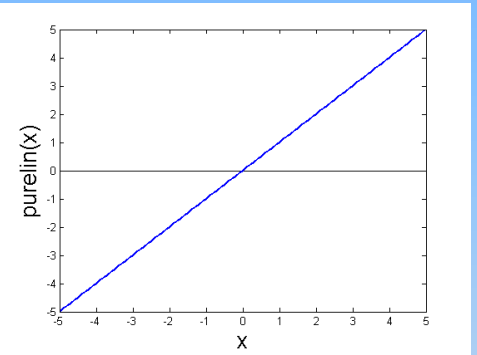
Activation functions



Logarithmic sigmoid

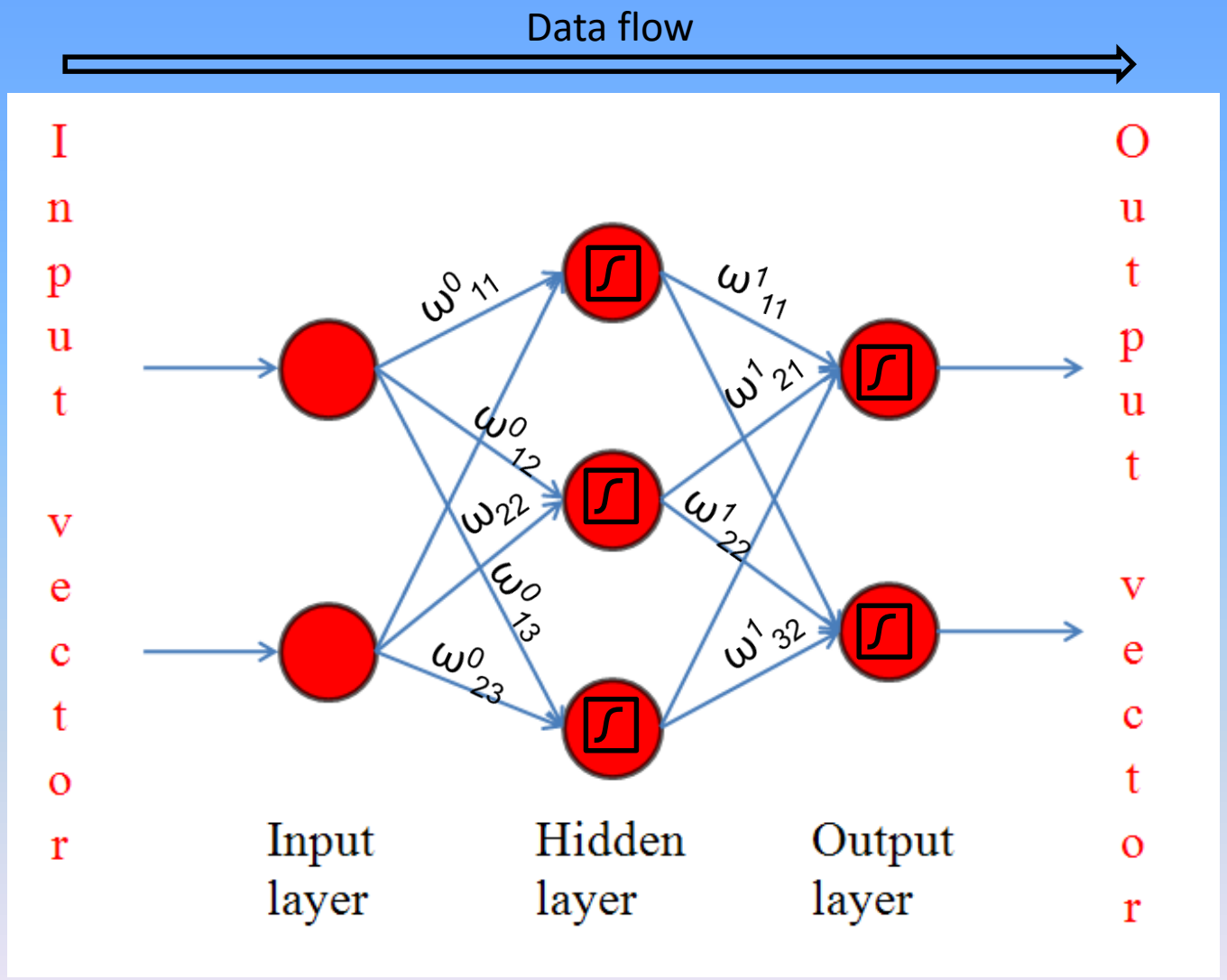


Hyperbolic tangent sigmoid



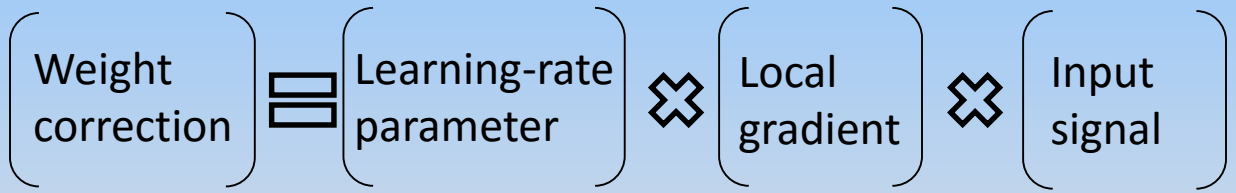
Linear transfer function

Feed-forward back-propagation neural networks



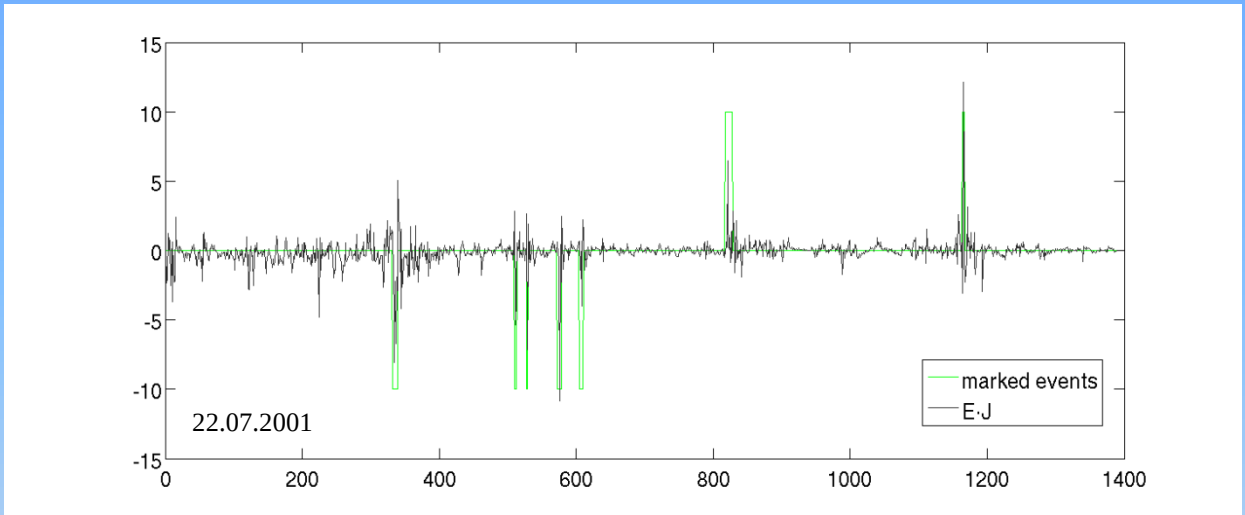
Learning mechanism – back-propagation algorithm

- Training data : known input-output pairs
- Input data is run through the network and the weights are adjusted in order to minimize the error:



- If the neuron is an output neuron, local gradient takes into consideration the error signal and the derivative of the activation function
- If the neuron is a hidden node, the local gradient takes into consideration the derivative of the activation function and the weighted sums of the local gradients of the neurons in the next layer

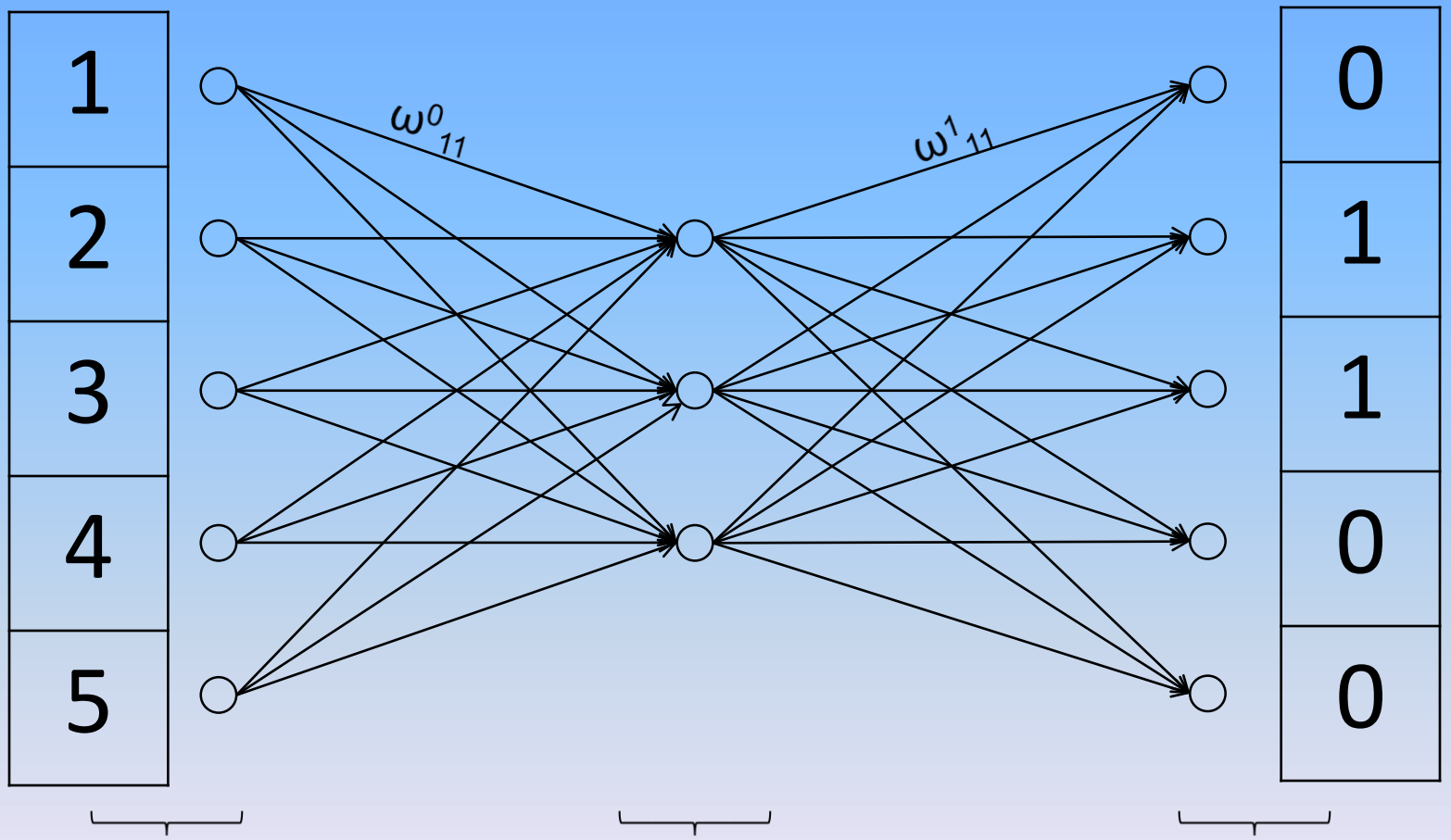
Encoding events



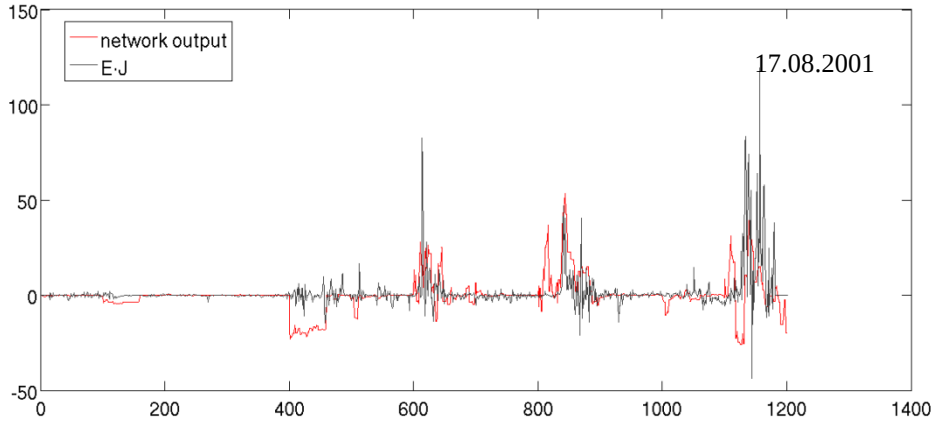
- Encoding used :
  - +1 for CLRs (E·J>0)
  - -1 for CGRs (E·J<0)
  - 0 – in rest

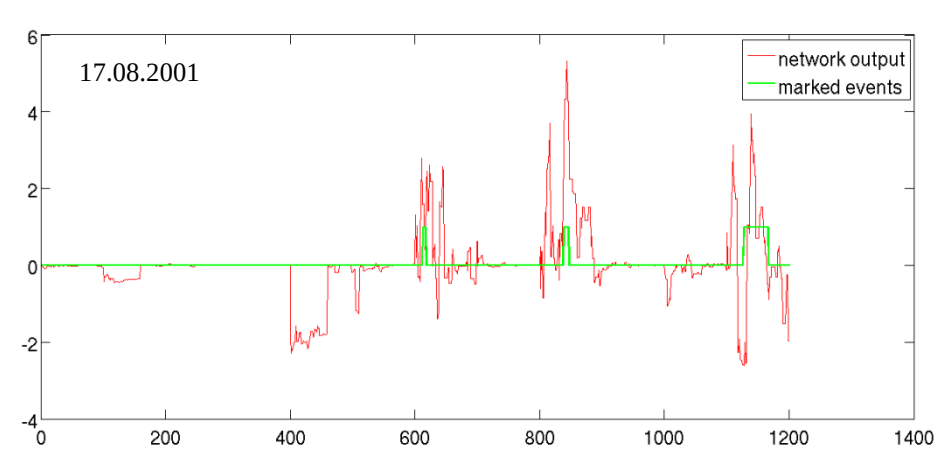
Fixed interval– network configuration

Interval size = 5; hidden layer size = 3



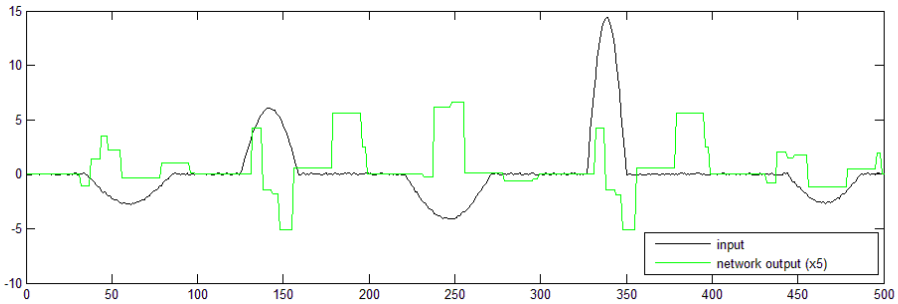
Results based on real data – 100 input intervals



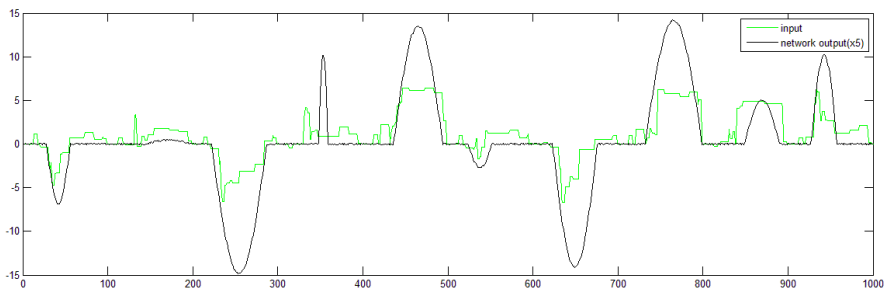


The trained network identifies pretty well the ECR events, including one event not included in the manual database (around x=400). However, the network response does not consist only of -1, 0, and 1, as required during the training stage (in the top plot the response is scaled by 10 for better visibility).

Results based on synthetic data – 100 input intervals



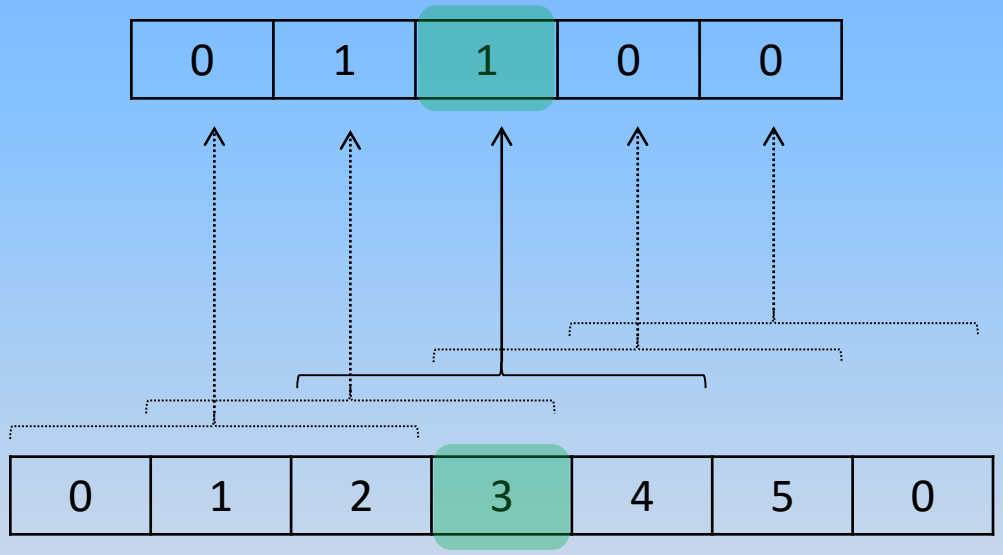
In the top panel we observe the reaction of the NN for all of the events. The identification of the events is not clear, and most of the time, the output of the network has the opposite sign to the desired result. We also notice a reaction to regions that do not contain any events



The bottom plot presents the output of a NN with less neurons on the hidden layers, but trained with more input points. We observe an improved reaction of the network and the better identification of source and generator events .

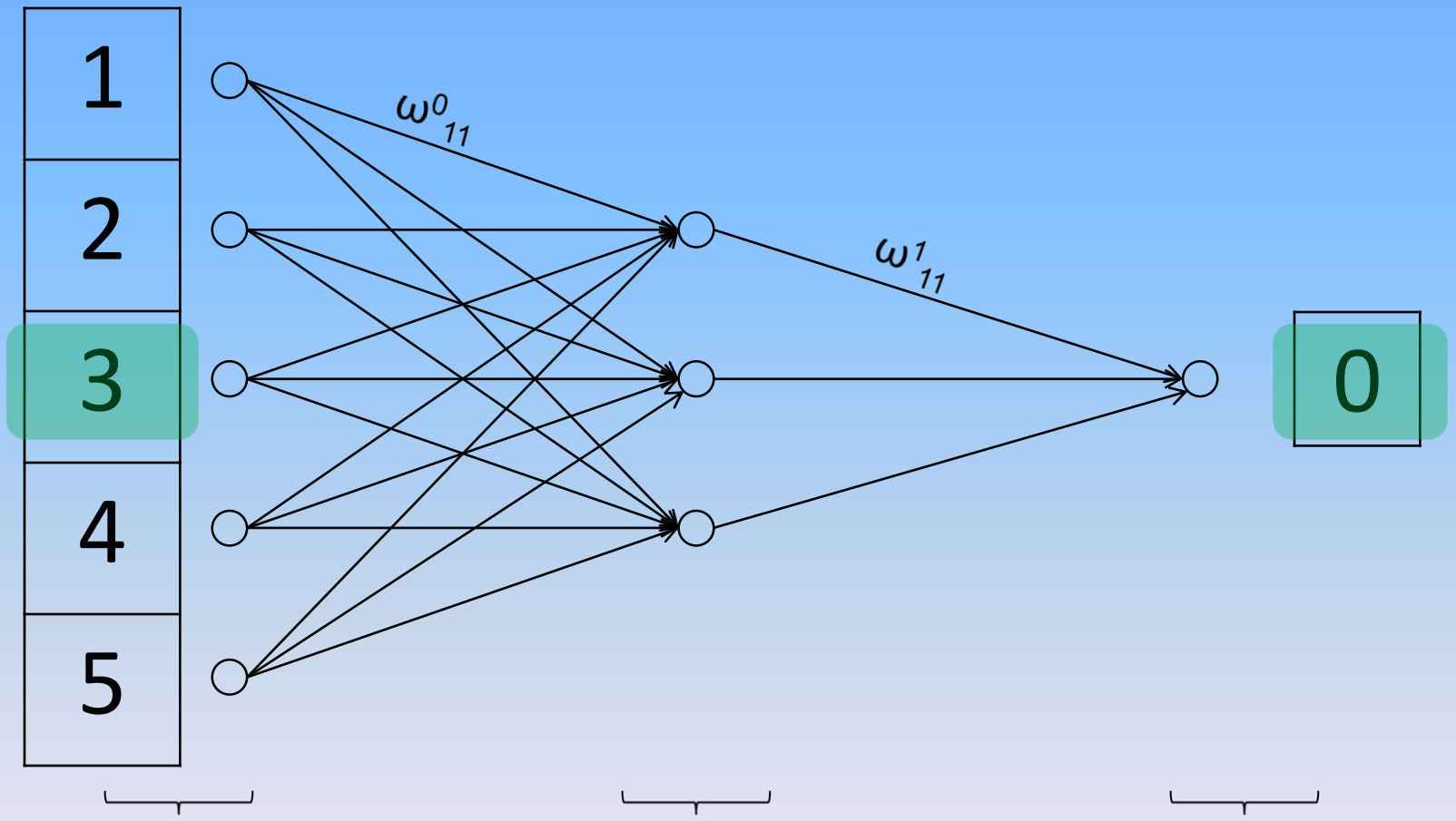
Sliding window - principle

Window = 3

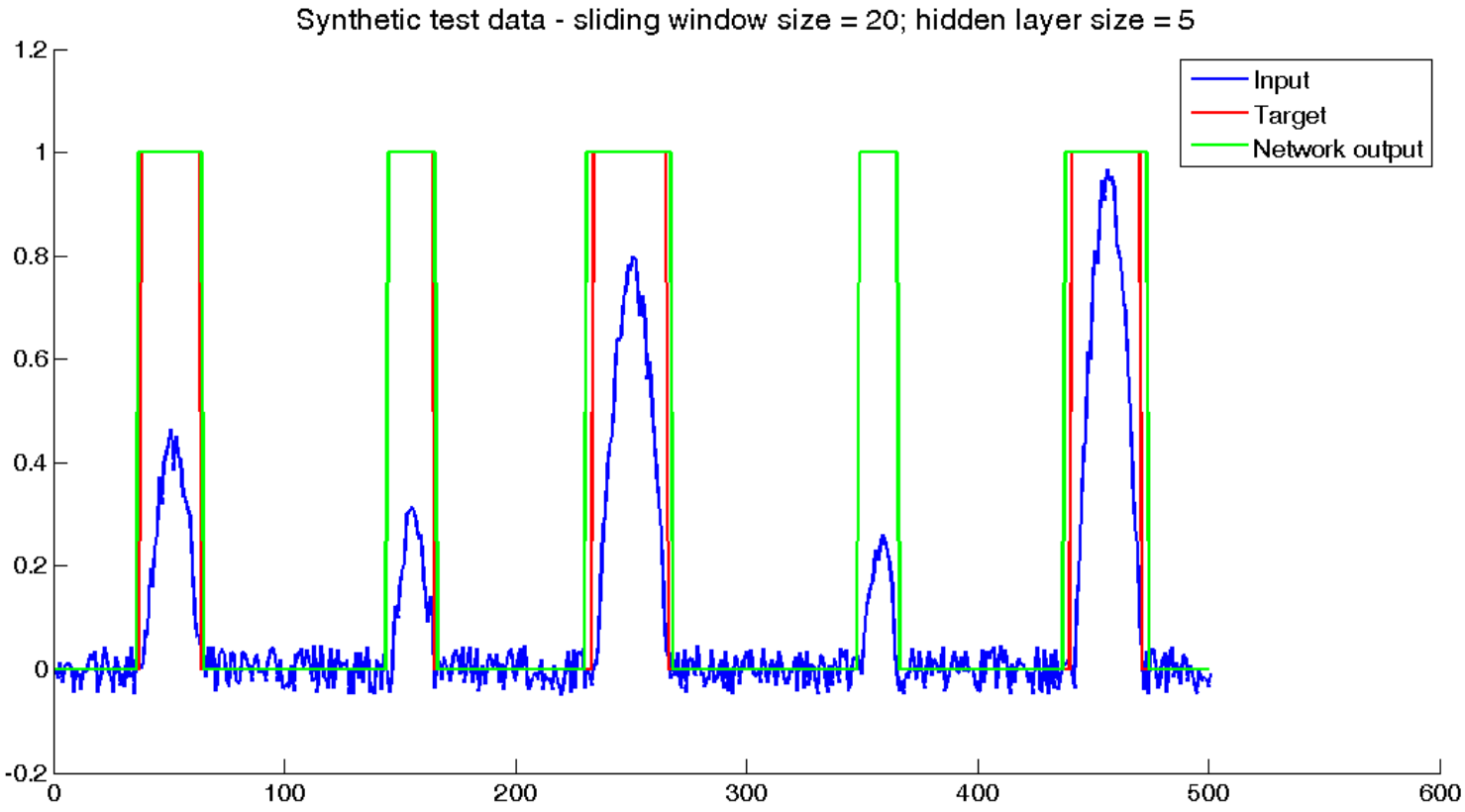


Sliding window – network configuration

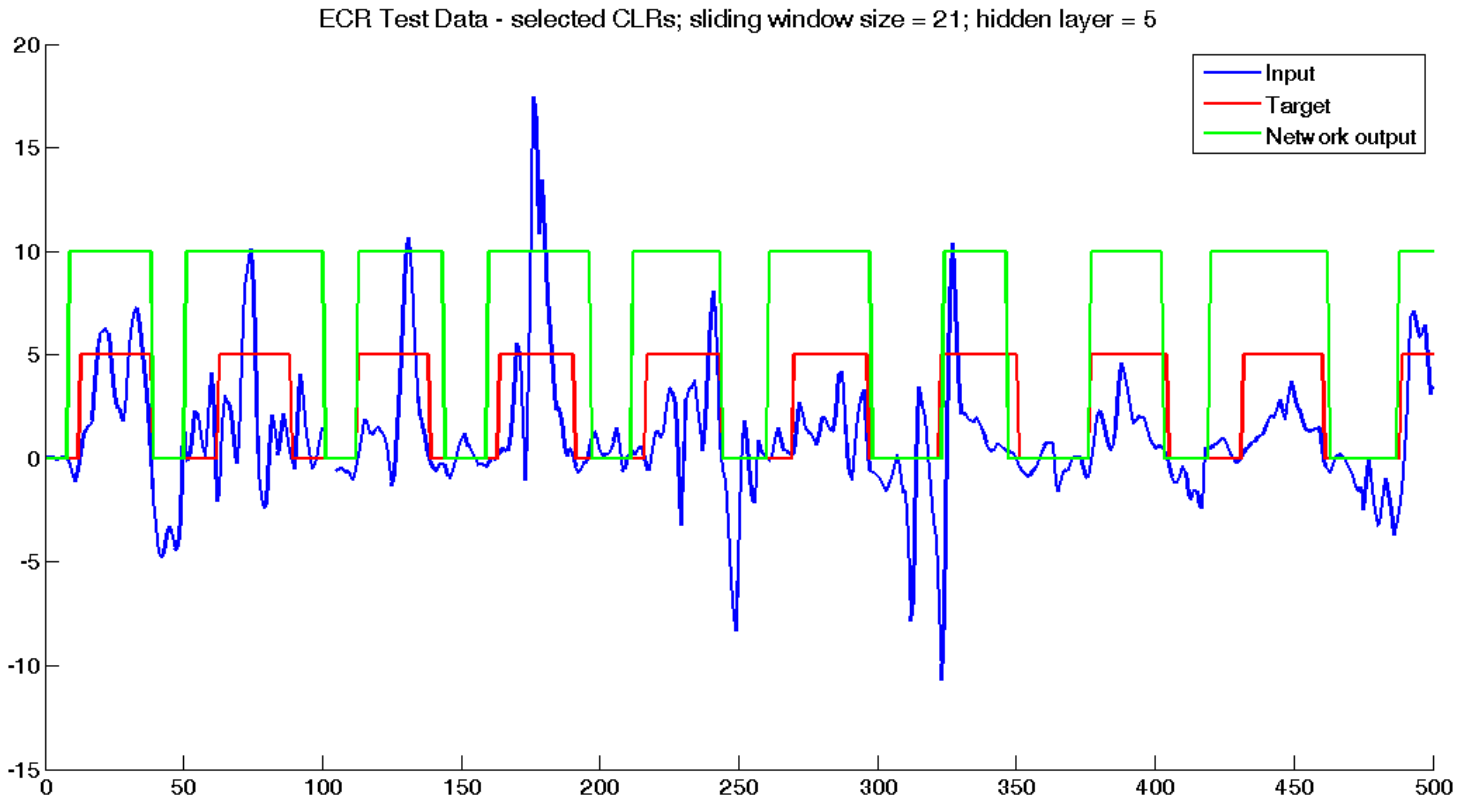
Window size = 5; hidden layer size = 3



Results using synthetic data



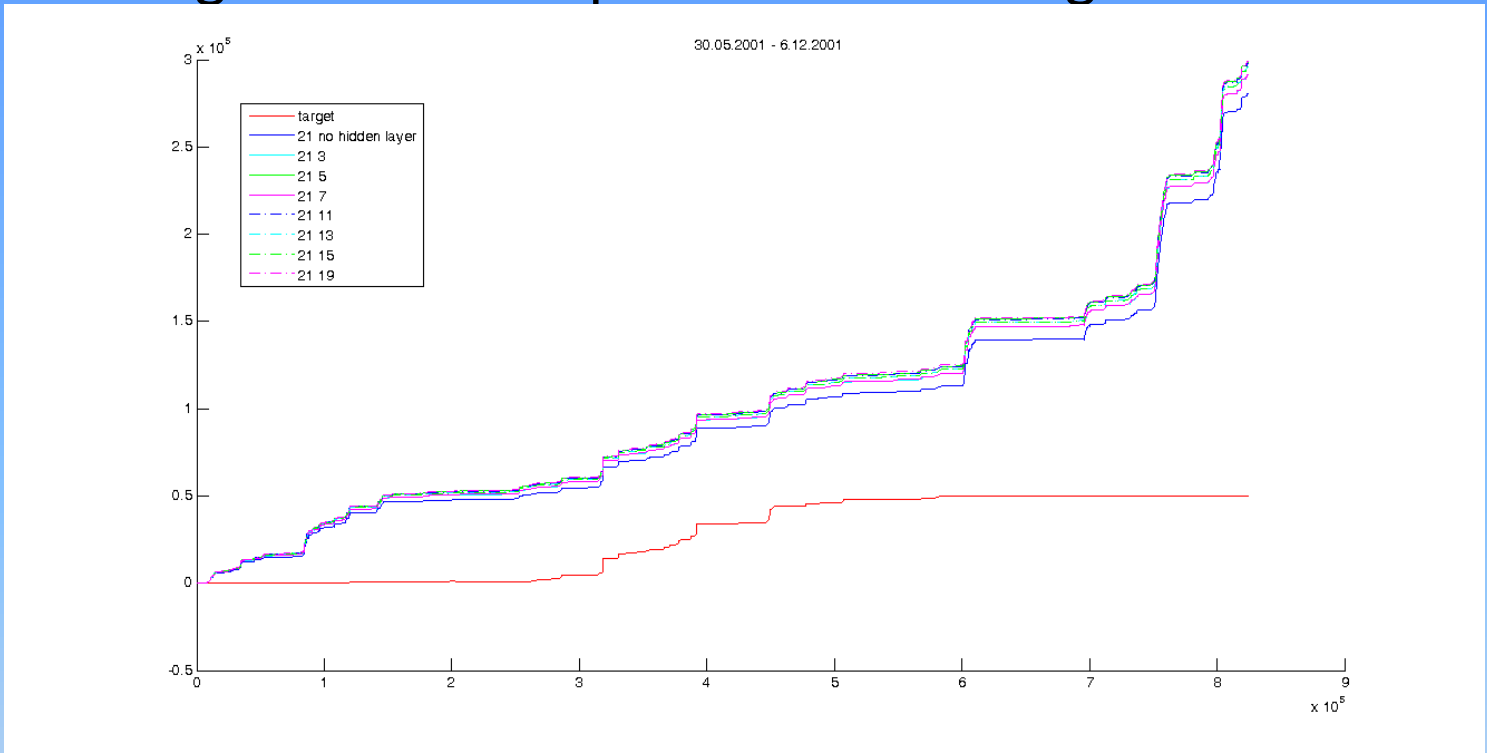
Results using a selection of real CLRs data



Test run using a selection of Cluster data

- The target ECR events were identified by a semi-automatic procedure, developed at the Umea University (Hamrin et al., Ann. Geophys., 27, 4131, 2009)
- The selection of data used for both the training and the test run was obtained by cutting out of the original data most of the 'non-event' time intervals between two events. In the remaining data sets, the 'event' and 'non-event' time intervals ('on' and 'off' in the red line) have about the same weight.

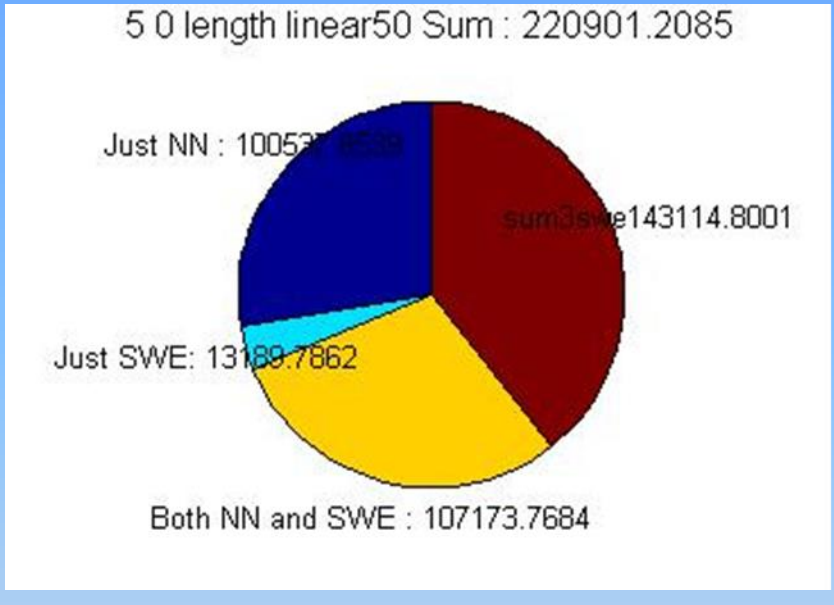
Evaluating the network performance using cumulative sum

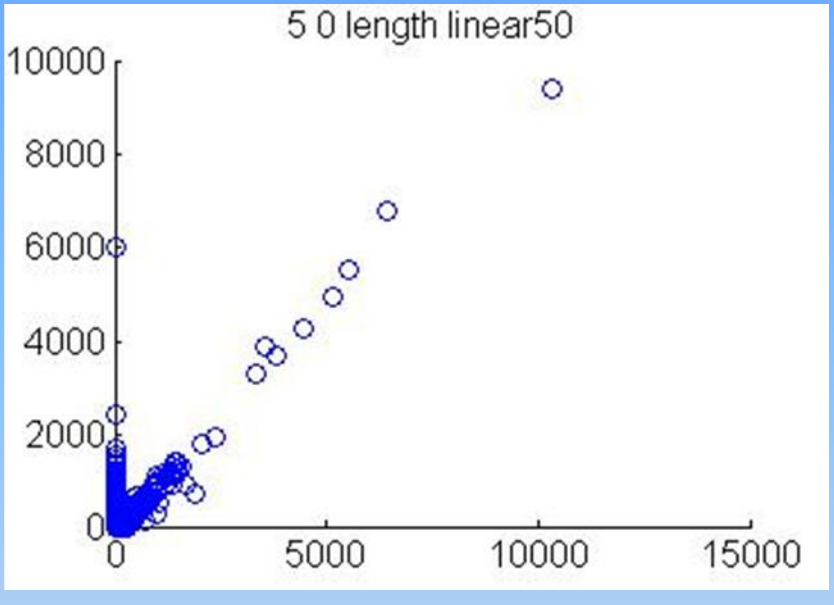


Cumulative sum of E.J over load regions – Cluster data from 2001

- The detailed examination of the results shows that the NN identifies neighboring spikes as separate events, while the semi-automatic procedure includes merging such spikes together. By integration (cumulative sum, CS) this difference becomes irrelevant and one can better compare the results.
- The CS can also provide a global perspective on the results and is appropriate for statistical investigations.
- The steps in the CS are remarkably similar over certain time intervals (e.g. between 3 and 5 on the abscissa - August/September).
- Some of the steps, in particular the very big ones, are only seen in the NN results. A preliminary exploration suggests that such big steps are related to real but noisy events, discarded by the semi-automatic procedure (which is rather conservative - better skip an event than count a non-event - and sensitive to noise). An event oriented study will focus on such big steps.

Evaluating network performance II



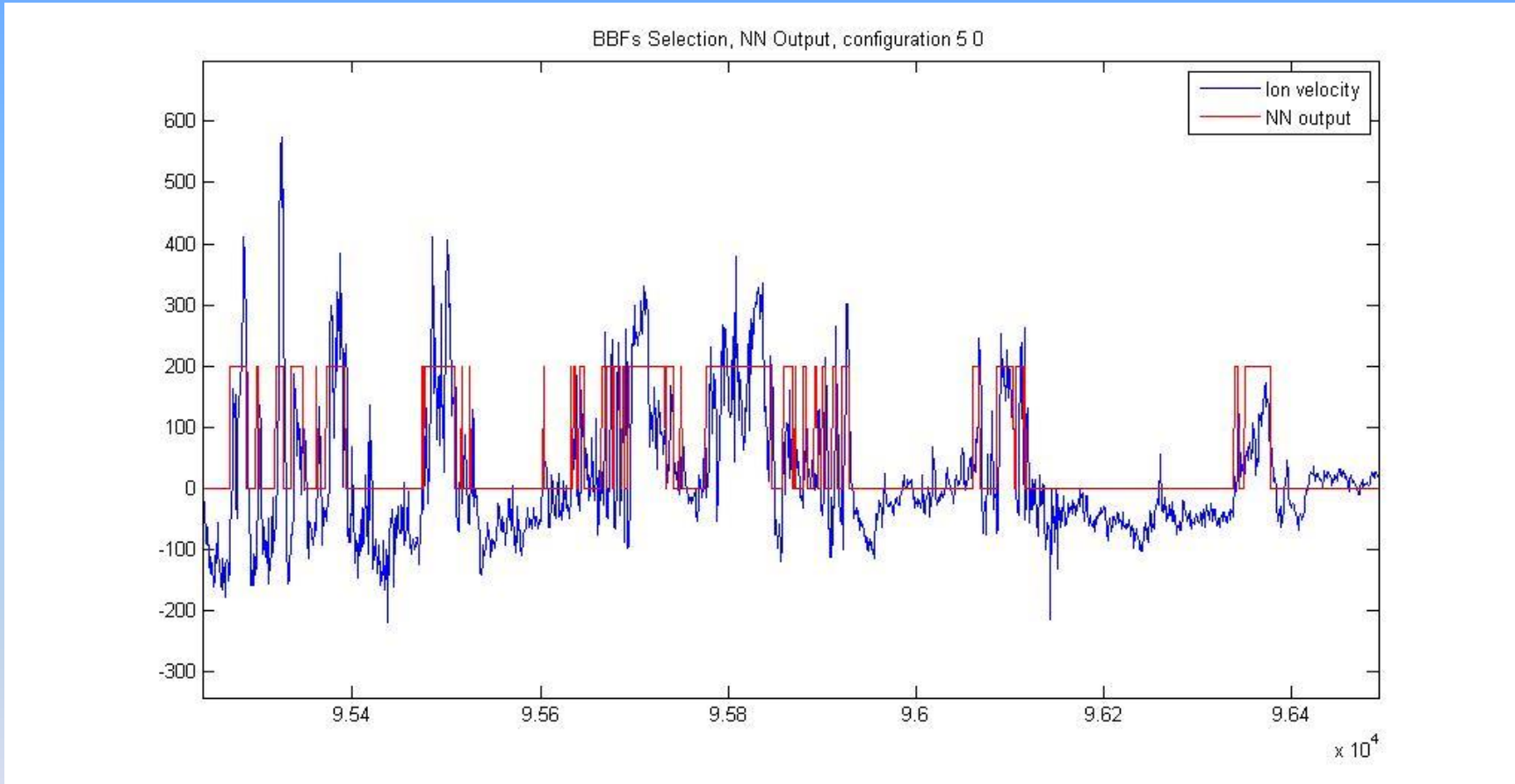


scatter plots of the cumulative sum of E•J, this time computed individually for each event.

The training set consisted of 50 events, linearly distributed in duration. The leftmost plot shows the cumulative sum of E•J over the selected events:

- with blue the events found only by the neural network;
- with brown the common events, cumulative sum computed over the semi-automated selection;
- with yellow the common events, cumulative sum computed over the network selection;
- with cyan the events found only by the semi-automated procedure

Ion velocity data



- Tested on ion velocity data for locating Bursty Bulk Flows (BBFs; Angelopoulos, et al., 1992) . Data from the 3D plasma instrument and from the fluxgate magnetometer on board the AMPTE/IRM satellite is considered.
- The figure above shows the rounded output (multiplied by 200 for better visibility) of a configuration with 5 input neurons and no hidden layer.

Open questions, future work, method applications

- When using Cluster data we encountered two main problems: the training set was limited in size and the regions marked with 0 in the training set could not be explored later – in a consistent manner – for the presence of CLRs / CGRs .
- By using synthetic generated data the training set size limitation is removed and we can clearly mark the interesting regions in the data. We can clearly notice the improved results when using a larger training set.
- When using the sliding window method, we must find the balance between the two main parameters, the size of the window and the number of neurons on the hidden layer as well as the training parameters of the network
- Can we use on real data a network previously trained on synthetic data? If so, how can we best adjust the synthetic data parameters (Eg. : noise, event amplitude and duration) for best event detection on real data ?
- Can the detection be stable enough ? Starting from the same training data and using the same network configurations can we get similar if not identical results ?
- Further running on real Cluster data and fine-tuning network parameters for optimum results
- When perfected, the method can be used to identify various signatures in data

ISS GPSM computing cluster

- >10 TB storage; RAID 1,5 or 6
- Dell Blade server, completely populated – 16 computing nodes
- Node configuration :
  - Dual Quadcore 2.2 Ghz CPU;
  - 96 or 128 GB RAM
- Interconnect : Gigabit Ethernet and Infiniband QDR
- UPS : 8kVA
- Interface : KVM switch

Thank you for your attention