Learning at the Speed of Light: A New Type of Optical Neural Network

A. Steven Younger [1] and Emmett Redd [2]

[1] Jordan Valley Innovation Center
[2] Department of Physics, Astronomy and Materials Science
Missouri State University
Springfield, Missouri USA
{SteveYounger,EmmettRedd}@MissouriState.edu

This material is based on work supported by the United States National Science Foundation under Grant No. 0725867.

Learning at the Speed of Light

- Focus on the Fixed-Weight Learning Nets than details of opto-electronic hardware (in paper)
- The problem that we are addressing
- Overview of Optical Test Apparatus
- Fixed-Weight Learning Neural Networks
  - Theory
  - Creating and Training
- Experimental Results
  - uMULT
  - PlanTran
  - BooLean
- Future Work

Optical Neural Networks

- Two main activities "Think" and "Learn"
- Forward Propagate: "Think"
  - Think →Matrix Multiply followed by nonlinear "squash"

$$y_j = \sigma_j\left(\sum_i W_{ji} \cdot x_i\right), \quad \text{where } \sigma_j \text{ is}$$

$$\sigma_j(s) = \text{logsig}(\text{s}) = \frac{1}{1+e^{-s}}, \quad \text{... or perhaps....}$$

$$\sigma_j(s) = \text{linsig}(\text{s}) = \begin{Bmatrix} 0, & \text{if } x \le 0 \\ \text{x}, & \text{if } 0 < x \le 1 \\ 1, & \text{if } 1 < x \end{Bmatrix}$$

Optical Neural Networks

"Think" executes very fast on simple optical hardware.

Parallel attenuation of light signals passing through synaptic medium

Intensity-level or pulse-coded analog optical signals

$$y_j = \sigma_j \left(\sum_i W_{ji} g x_i \right)$$

Op-Amps or nonlinear materials (VO_2)

Synaptic Media (35 mm film, SLM)

Optical Intensity Parallel Summation (Cylindrical Lenses, Holographic Media)

Optical Neural Networks

"Learn" is slower and requires more complex and costly hardware.

- Iteratively adjust synaptic weights toward values that minimize errors.
- Performed by a Learning Algorithm, such as the well-known Backpropagation of Errors
- Analogous to long-term memory in biology.

$$\Delta W = -\eta x \times y \times (1 - y) \times (y - T)$$

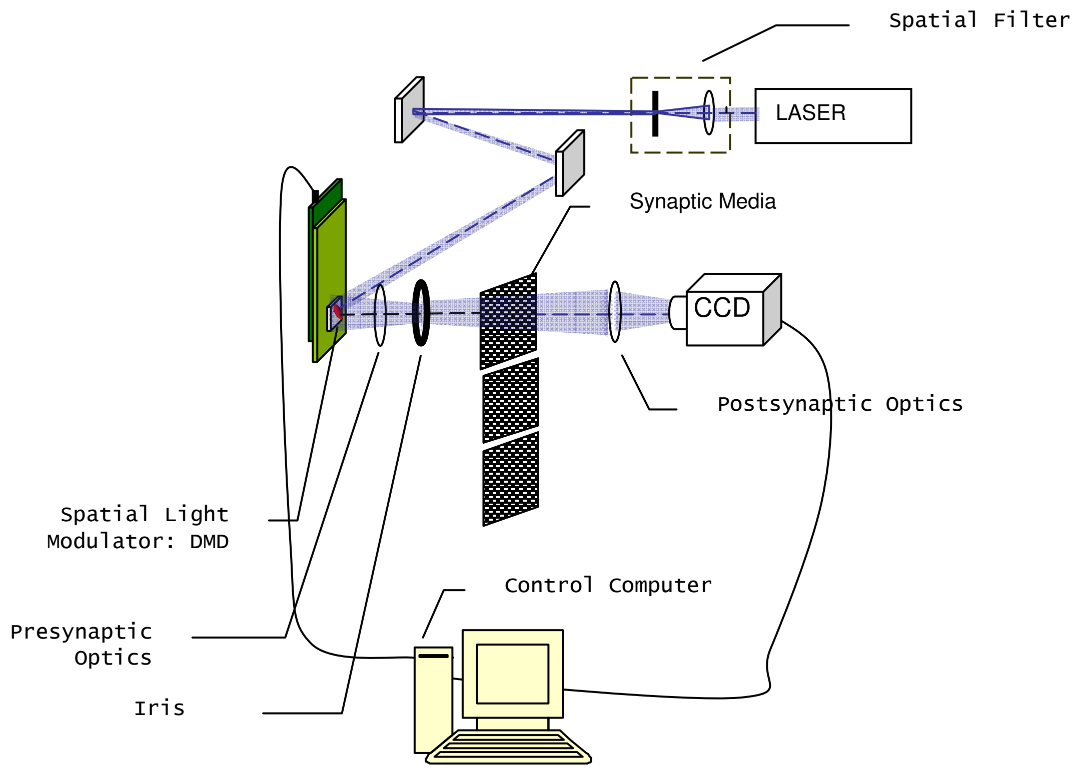
Multiply a signal by a another signal: higher-order synapses, more hardware

Changing Synaptic Weights is a slow process: milliseconds (SLM) to hours (film)

Fixed-Weight Learning Neural Networks FWL-NN

- Most Optical NNs use standard "von Neumann" CPU-based computations to perform learning.
- We believe the learning speed issue is major reason for the lack of common use of Optical NNs.
- FWL-NNs are our solution to this problem.
  - "Learning" takes place at "thinking" speeds.
  - First order synapses
  - Don't ever need to change weights
  - Analogous to short-term *working memory* in biology.
  - Adaptive

Optical Neural Network
Laboratory Test Apparatus



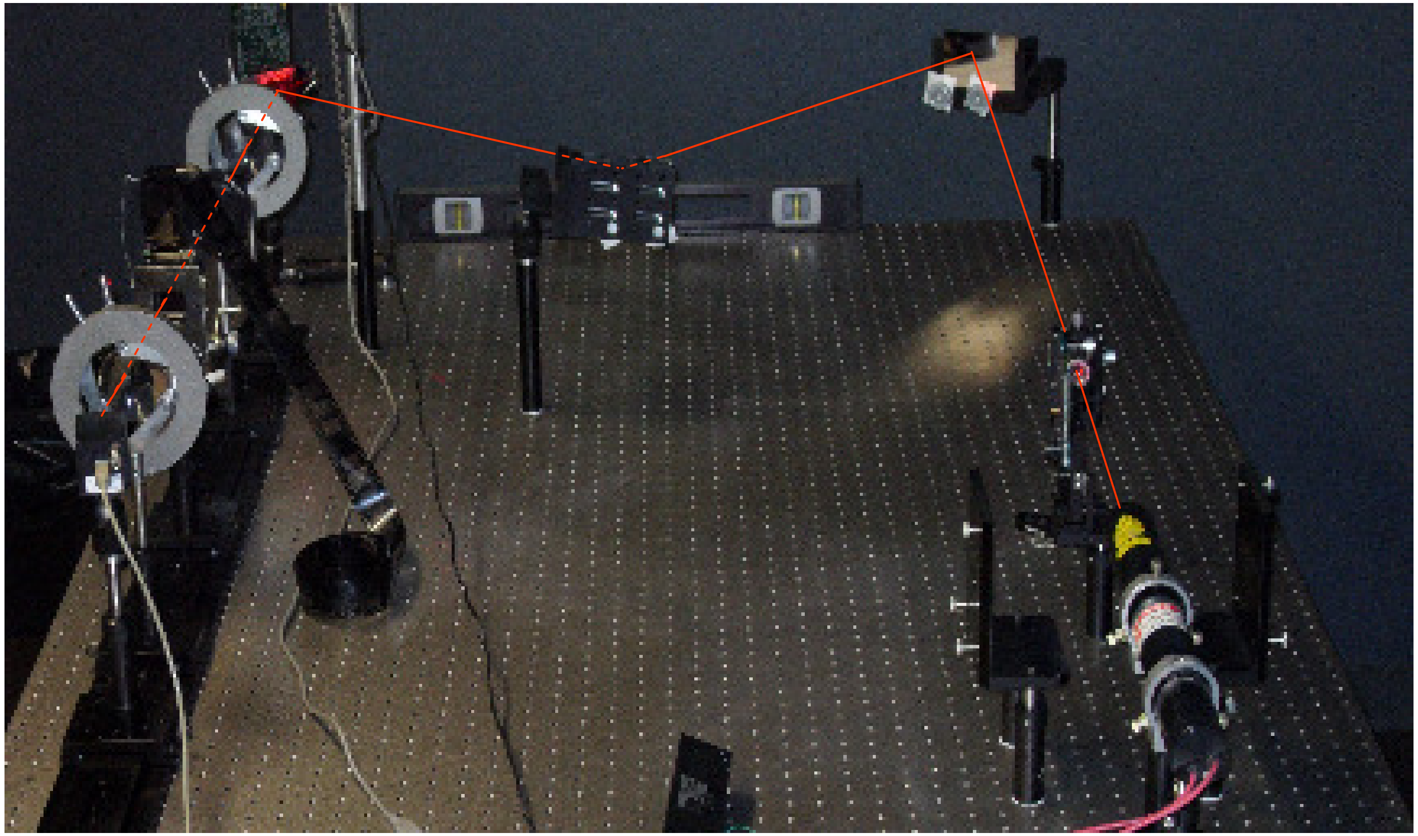
Optical Neural Network

Laboratory Test Apparatus

- Designed for flexibility, not speed.
- Digital Micromirror Device (DMD) for electronic-to-optical signal
- Pulse based - stochastic (SP) or duty cycle (PWM)
- Typical timing: 1 exemplar cycle → 4 phases → 4×256 pulses
- 35mm film for Synaptic Media.
  - Opaque/Clear pixel density encoding of synaptic medium.
- CCD Camera for optical-to-electronic signal conversion
- Software-based summation and squashing.
- Synchronous operation of neurons.
- Intensity Calibration every cycle or every phase.
- Software-based distortion corrections.

Optical Neural Network

Laboratory Test Apparatus



Fixed-Weight Learning Neural Networks

- Standard neural networks learn new function mappings by the changing of their synaptic weights. However, the FWL-NNs learn new mappings by dynamically changing recurrent neural signals.

- The (fixed) synaptic weights of the FWL-NN implement learning "algorithm" which adjusts the recurrent signals toward their proper values.

- That is, instead of encoding a particular mapping, the synaptic weights of a FWL-NN encode how to learn any mapping.

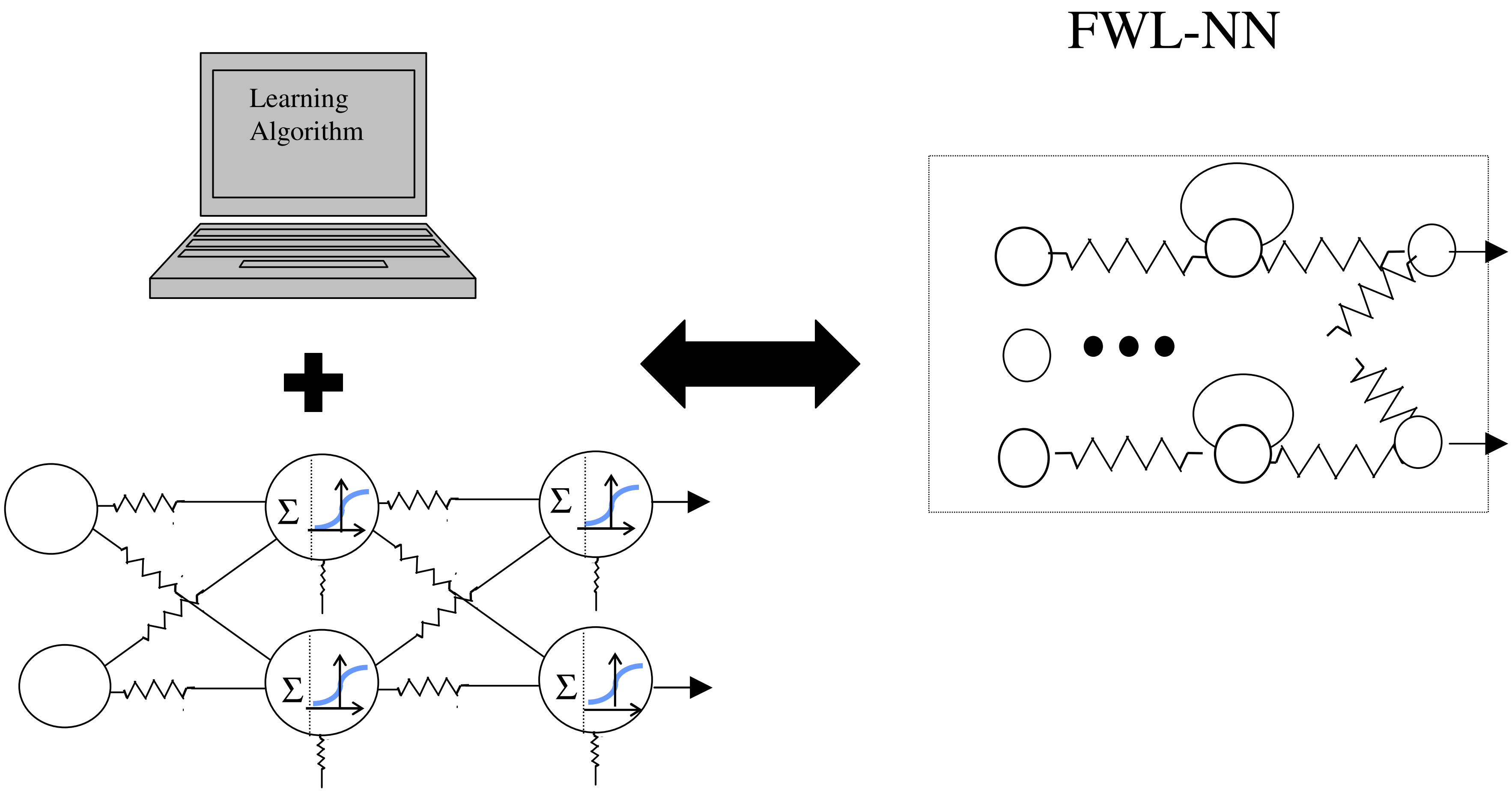
FWL-NN:
Some History

- Not a new idea, application to Optical NN is new.
- Also called "Adaptive Behavior with Fixed Weights" and/or "Accommodative Neural Networks"
- Cotter, Conwell, Prokhorov, Feldkamp, Hochreiter, Younger, Redd, Lo

Fixed-Weight Learning Theorem

- Cotter and Conwell 1990 – 1991
- For any (changing weight) neural network and its attendant learning algorithm, there exists a FWL-NN that can learn the same functional mappings without changing any of its synaptic weights.
- Existence theorem
- Based on Universal Approximation.
- Must be recurrent. Usually larger than equivalent non-fixed-weight NN.
- Analogous to short-term *working memory* in biology.

FWL NN is Equivalent to a standard Neural Network and Learning Algorithm



Generating FWL-NNs: The sub-network method.

1. Decide the equivalent FFNN topology

2. Determine the function mapping for the learning algorithm (*planapse* or the teacher equation).

3. Train a planapse sub-network to learn the planapse equation.

4. Determine the *tranapse* (sometimes called the model) formula.

5. Train a tranapse sub-network to learn the tranapse formula.

6. Replace each FFNN synapse with the appropriate planapse-tranapse pair. Provide new connections as necessary.

7. Test/Validate the FWL-NN.

Method for Generating FWL-NNs: sub-network training.

- Scaling considerations.
  - Unipolar signals.
  - Limited range synaptic weights
  - Limited range signals.

- Signal propagation timing: cycles/phases/pulses

- Train on random inputs over wide range.

- Alternate squashing can simplify!

- Large training set (>25,000) many epochs (>100,000) on MATLAB **`traingdx.m`**

Generating FWL-NNs: Alternatives

- **Analytical:** same as sub-network method, but generate the sub-networks by analytically design instead of training.
  - We used it on the BooLean network.

- **Meta-learning:** optimize the (initially random) synaptic weights of a FWL-NN to be an efficient learner of function mappings from a given set of mappings.
  - Requires optimizing over many examples of many mappings from the given set of mappings.

- Can combine methods.

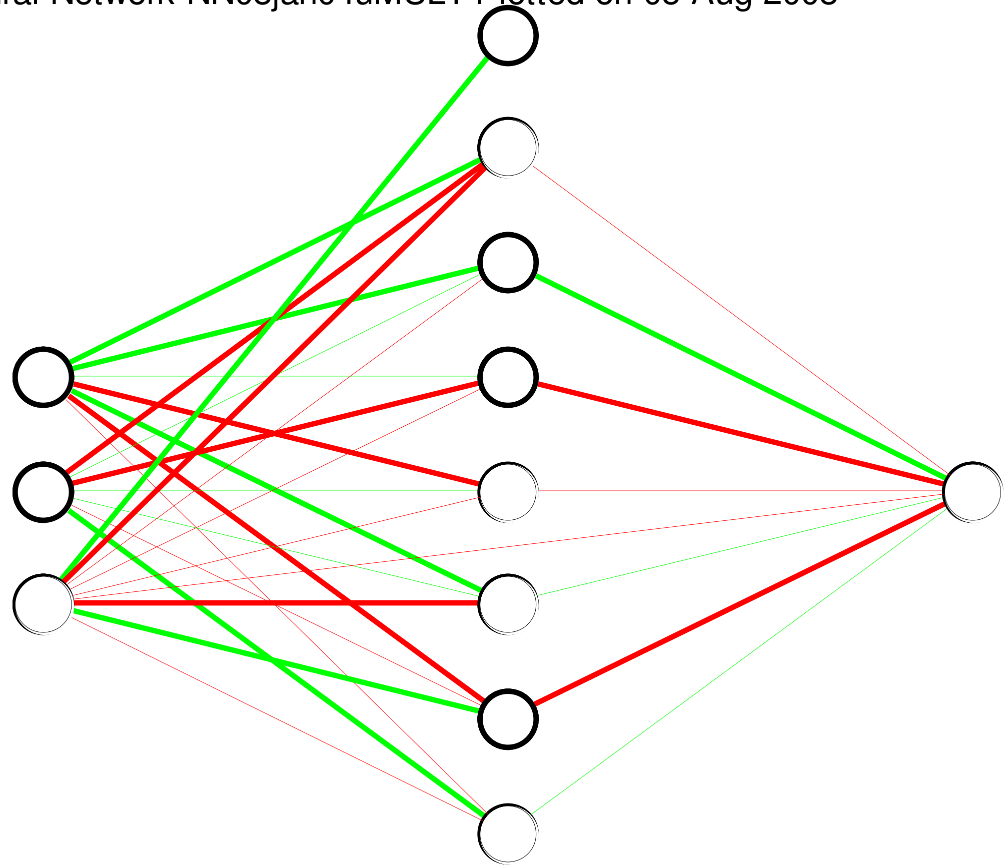
Experimental Results

- Several networks were created. Results from 3 reported here.
- uMULT – Unsigned Multiplication. Building block.

- PlanTran – A single planapse/tranapse pair.

- BooLean – Can learn linearly separable Boolean functions

Neural Network NN08jan04uMULT Plotted on 05-Aug-2008



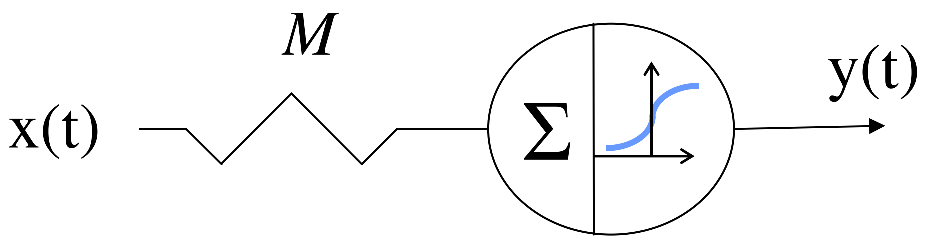
uMULT: Unsigned Multiplication
Training and Simulation Results

Large training set (25,000) many epochs (100,000) on MATLAB **traingdx.m**

Hidden Layer Size	MSE	Sig Bits
3	6.5003×10^{-4}	5.3
4	3.6876×10^{-4}	5.7
5	3.0794×10^{-4}	5.8
6	3.1636×10^{-5}	7.5
7	2.1617×10^{-5}	7.7
8	4.0069×10^{-5}	7.3
9	5.4367×10^{-5}	7.1

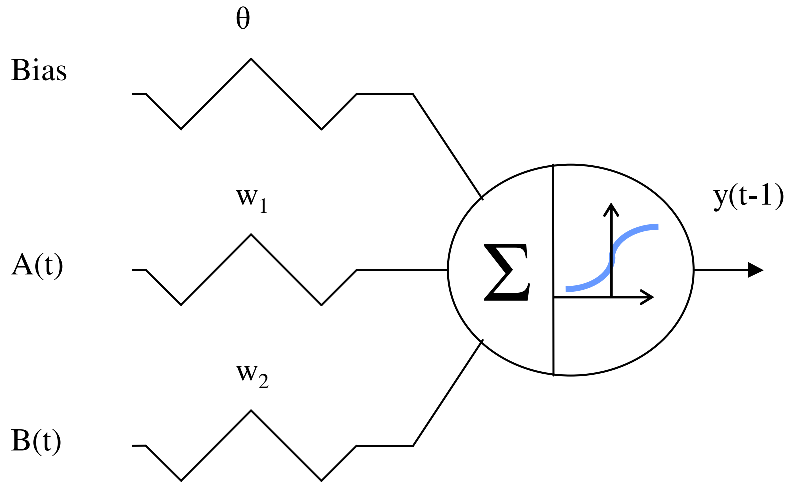
PlanTran FWL-NN:

Equivalent FFNN



Feed-Forward Network that can learn Linearly Separable Boolean Functions

(a single neuron)



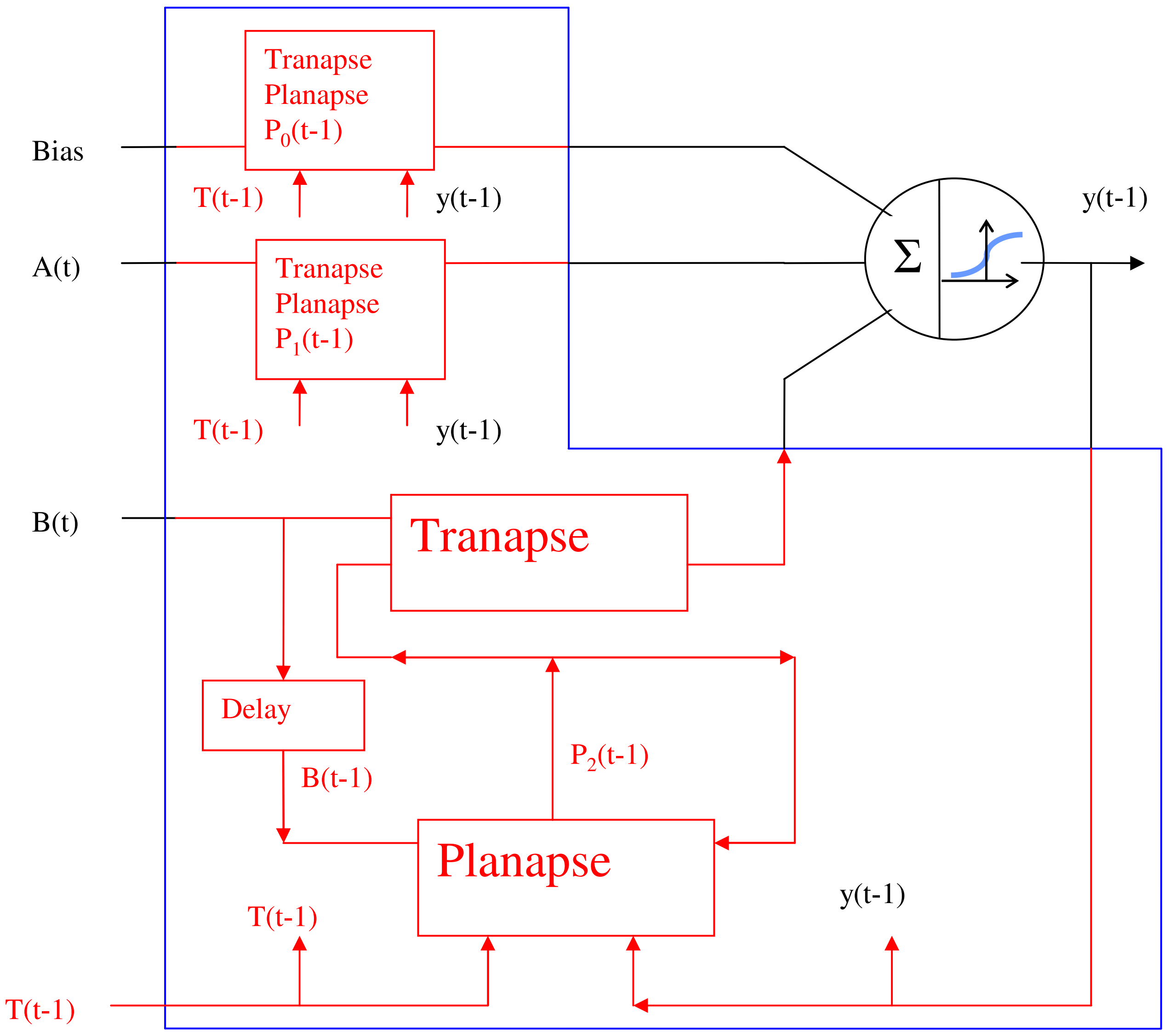
Weights for Various Functions

Function	θ	w_1	w_2
AND	-1	2/3	2/3
OR	-1/2	1	1
NAND	1	-2/3	-2/3
TRUE	1	0	0

. . .

Recurrent Fixed-Weight-Learning Separable Boolean Neural Network

One synapse of the above network has been expanded into its sub-network(s); weights replaced with potencies.



Generating Test Data for FWL-NNs.

Algorithm to generate training/test data for a FWL-NN:

```
repeat Number-of-Mappings times
  Randomly select a mapping M from a set S.
  repeat Number-of-Exemplars-per-Mapping times
      Generate a random input vector x
      Use x with mapping M generate target vector T
      Output training pair (x,T)
    end repeat
end repeat
```

Generating Test Data for FWL-NNs

Data Sets

- For *PlanTran*, set ***S*** is all function mappings, $T = \mathrm{logsig}\left(M \cdot x\right),\ -4 \le M \le +4$ where the real index *M* specifies the particular mapping.
- For *BooLean*, ***S*** is the set of all 14 of the Linearly Separable Boolean functions with two logical arguments and one logical result. The integer index *M* specifies the particular mapping.

Boolean Problem Set Truth Table

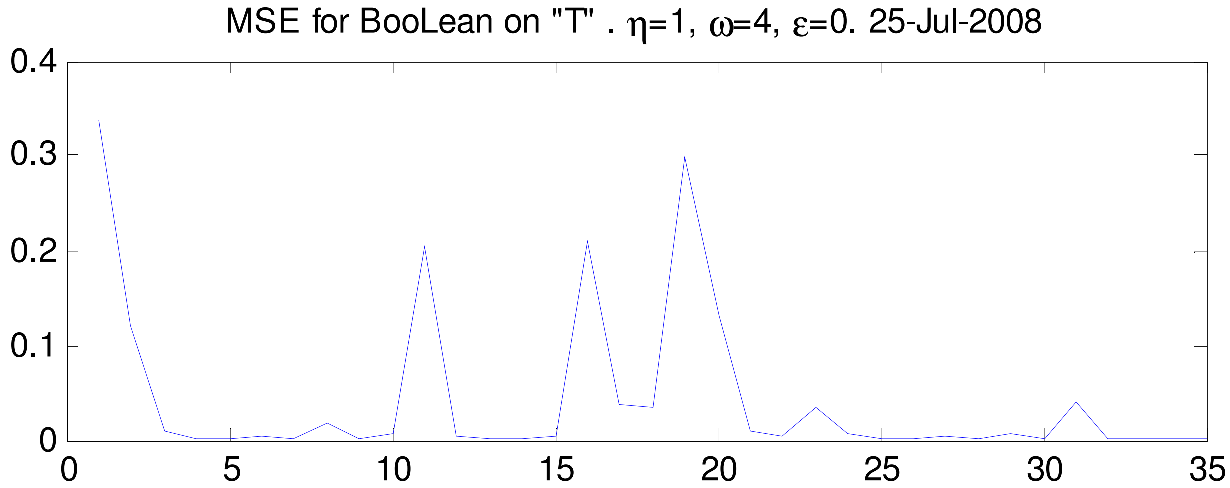
Index *M*	AB 00	AB 01	AB 10	AB 11	Function Mapping Name	Notes
1	0	0	0	0	**ALWAYS FALSE**	
2	0	0	0	1	**AND**	
3	0	0	1	0	**A AND NOT(B)**	
4	0	0	1	1	**A**	**ignore B**
5	0	1	0	0	**NOT(A) AND B**	
6	0	1	0	1	**B**	**ignore A**
7	0	1	1	0	**XOR**	**Disallowed**
8	0	1	1	1	**OR**	
9	1	0	0	0	**NOR**	
10	1	0	0	1	**NOT XOR**	**Disallowed**
11	1	0	1	0	**NOT(B)**	**ignore A**
12	1	0	1	1	**A OR NOT(B)**	
13	1	1	0	0	**NOT(A)**	**ignore B**
14	1	1	0	1	**NOT(A) OR B**	
15	1	1	1	0	**NAND**	
16	1	1	1	1	**ALWAYS TRUE**	

Experimental Results on Optical Hardware.

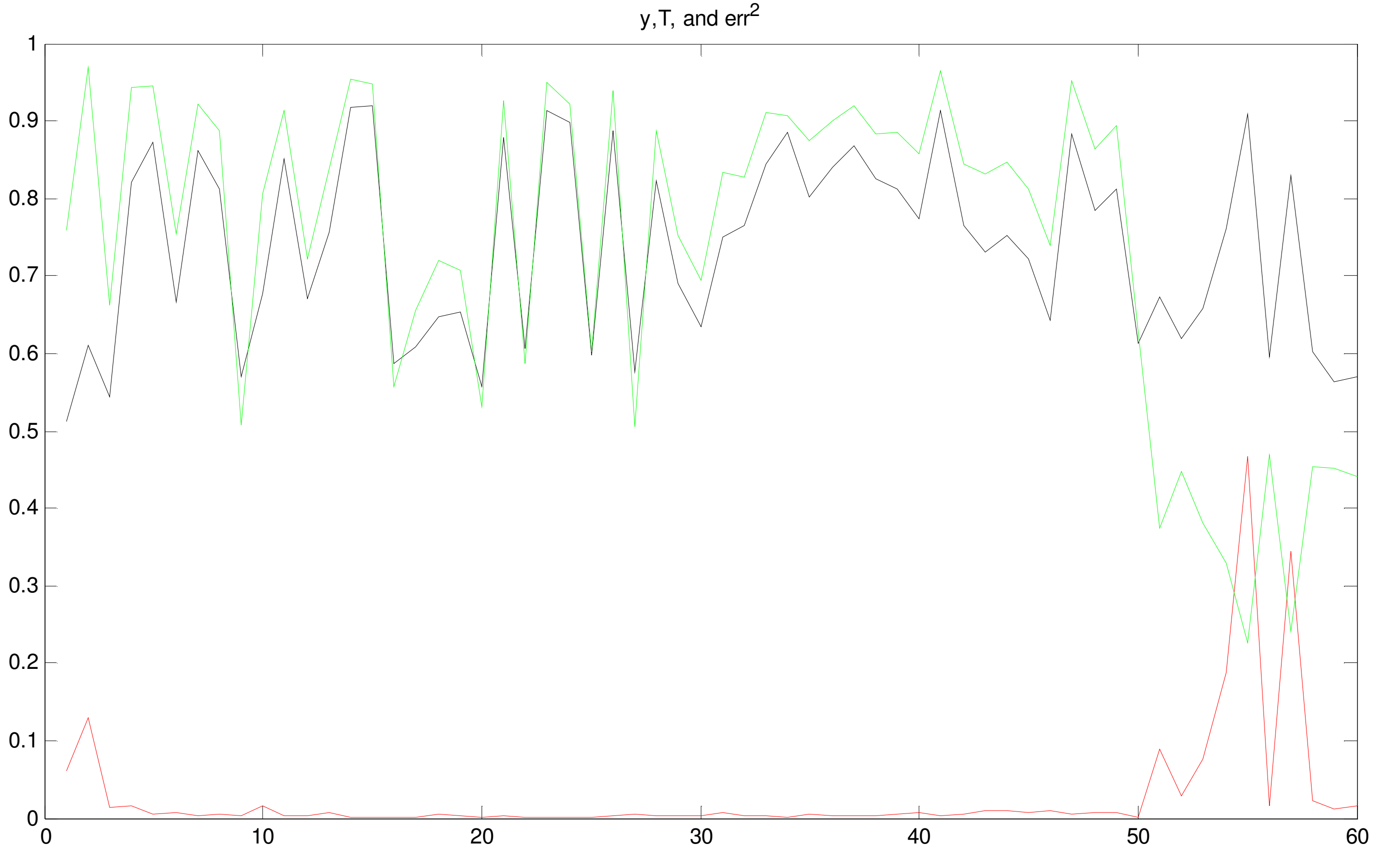
L- Number of Layers, N–number of neurons, W-––number of synapses, ϕ **-** Phases per Exemplar, Pulses – Number of pulse timeslots in one Phase. Learn – Number of Exemplars required to learn mapping (for FWL-NN) , MSE – mean squared error (after learning), SigBits – Number of Significant Bits

NN	L	N	W	ϕ	Pulses	Learn	MSE	SigBits	Notes
uMULT	3	13	30	2	128	n/a	0.0013	~6	
PlanTran	4	29	100	6	256	11	0.0083	~4	
BooLean	5	33	56	4	256	21	0.0076	~4	M=16

FWL-NN BooLean
Hardware-Based Learning Curve



FWL-NN PlanTran
Hardware-Based Learning Curve



Lessons Learned: FWL-NN

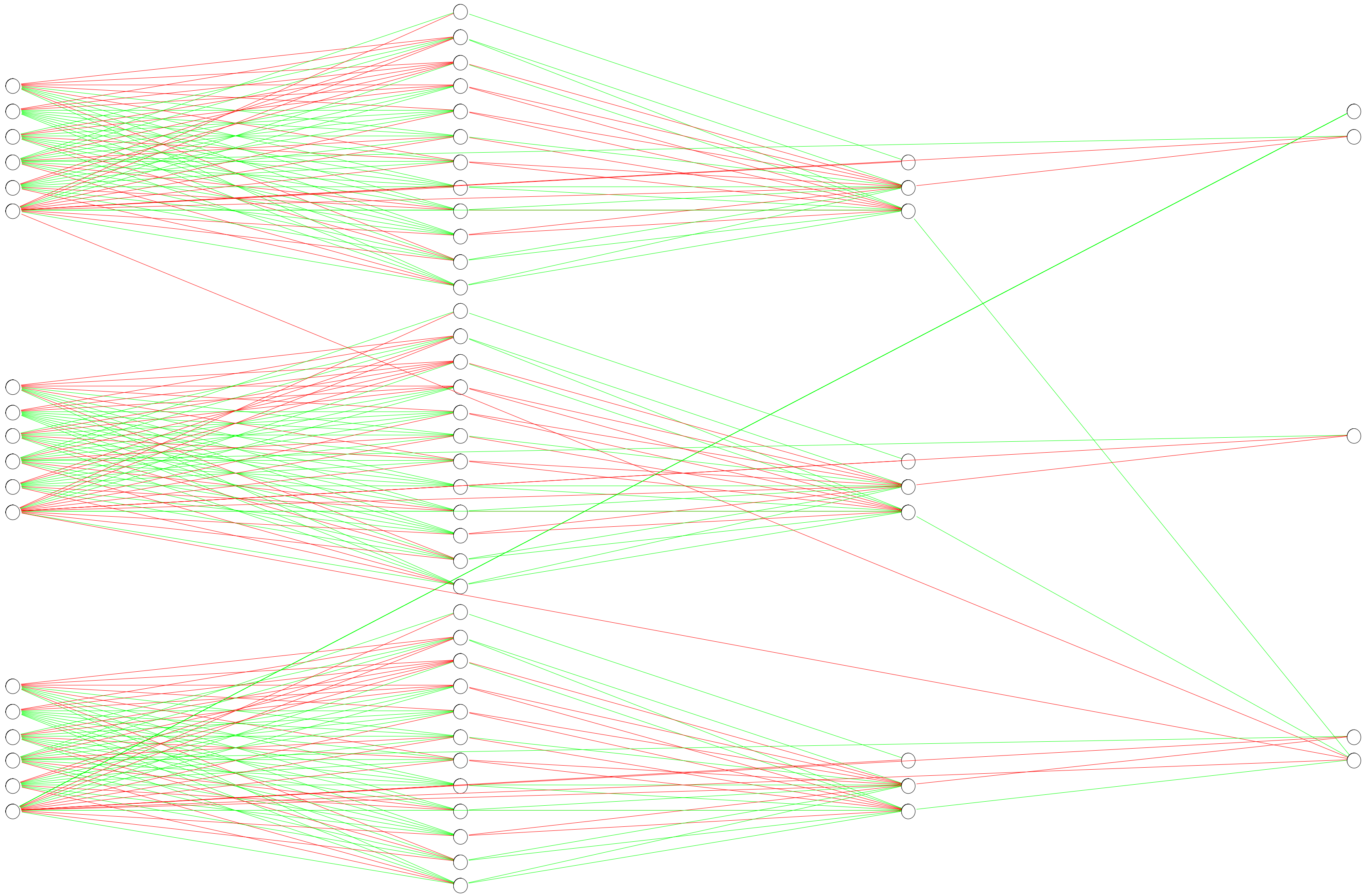
- It works!
  - Sub-network and analytic approaches are valid.
- Sufficient signal resolution (significant bits) is was hard to achieve.
- Synaptic weight resolution was sufficient with 35mm film ~16 bits. (However, repeatability was a problem.)
- “Opaque” areas on a slide aren’t.
- Both pulse encoding schemes have methods of trading-off other resources for more signal resolution.
  - Stochastic pulse: More bits are slower.
  - Intensity: More bits require more neurons.

Future Work

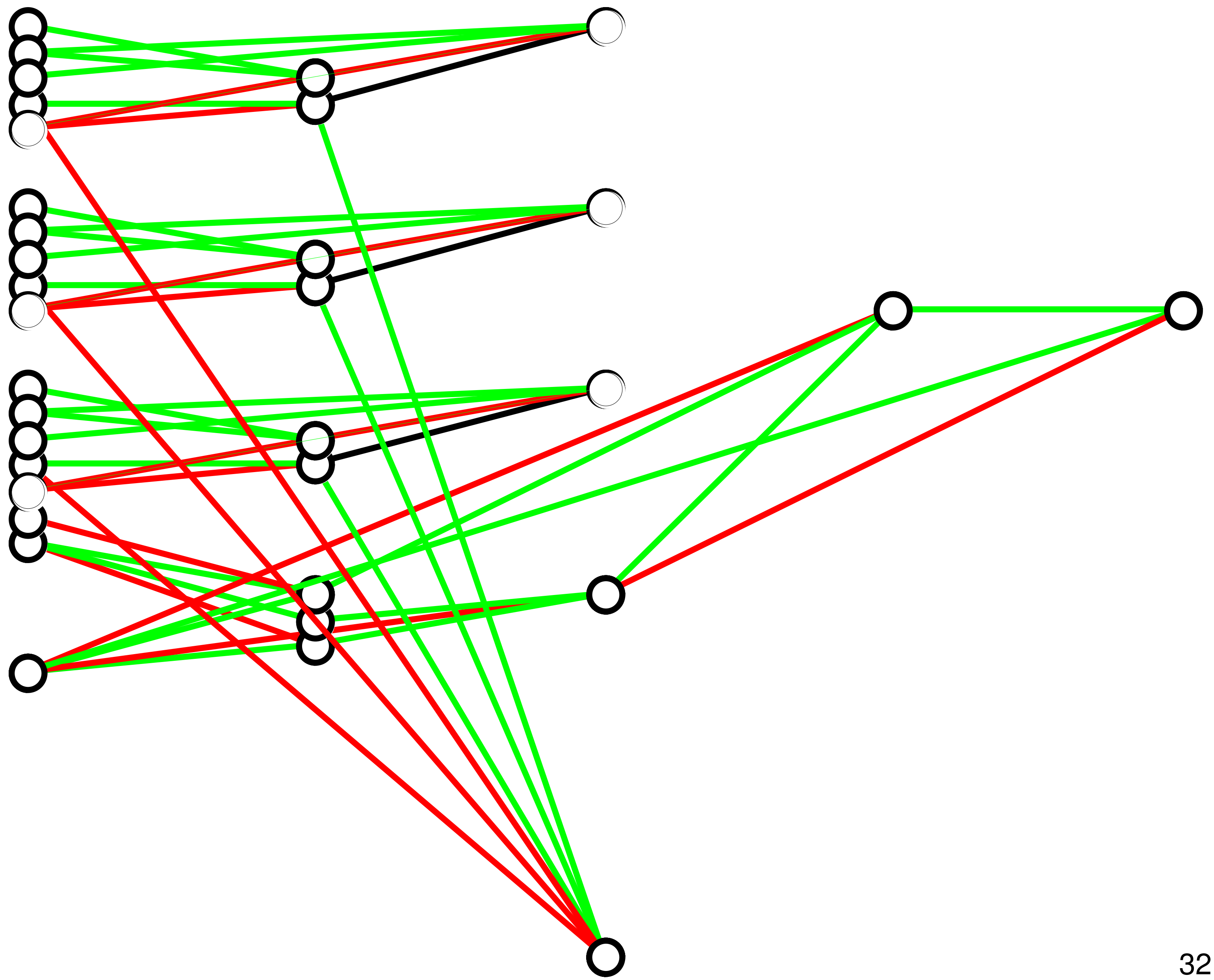
- Faster Hardware
  - More functions moved to optical path.

- Expanded Neuralware
  - Larger Networks
  - FWL-NNs that are equivalent to 3 layer FFNNs (Universal Approximation)
  - Improved Learning
    - Reduce overhead
    - Off-line or Batch learning.
- Applications
  - Real-World problems: Speech, Vision, Data Mining, Adaptive Robotics
  - APIs
  - Promotion

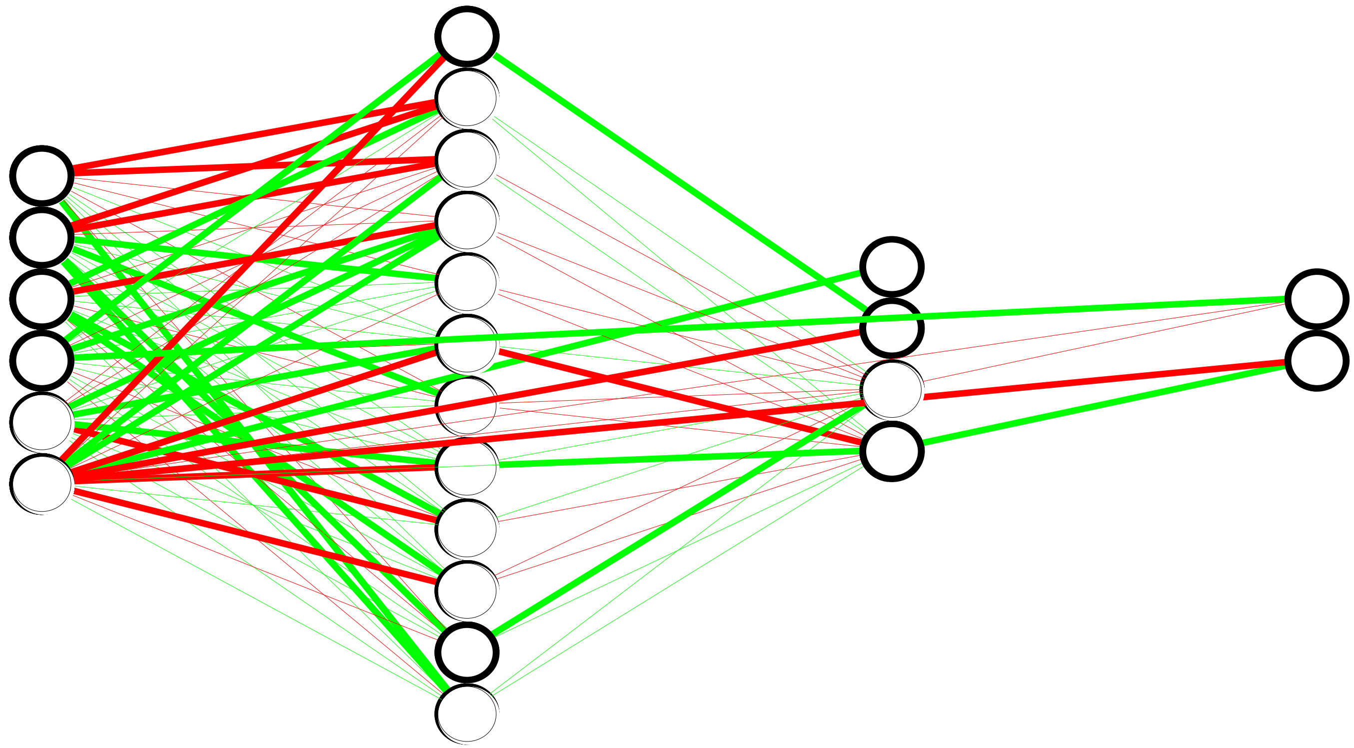
Neural Network NN07feb12Recurron Plotted on 05-Aug-2008

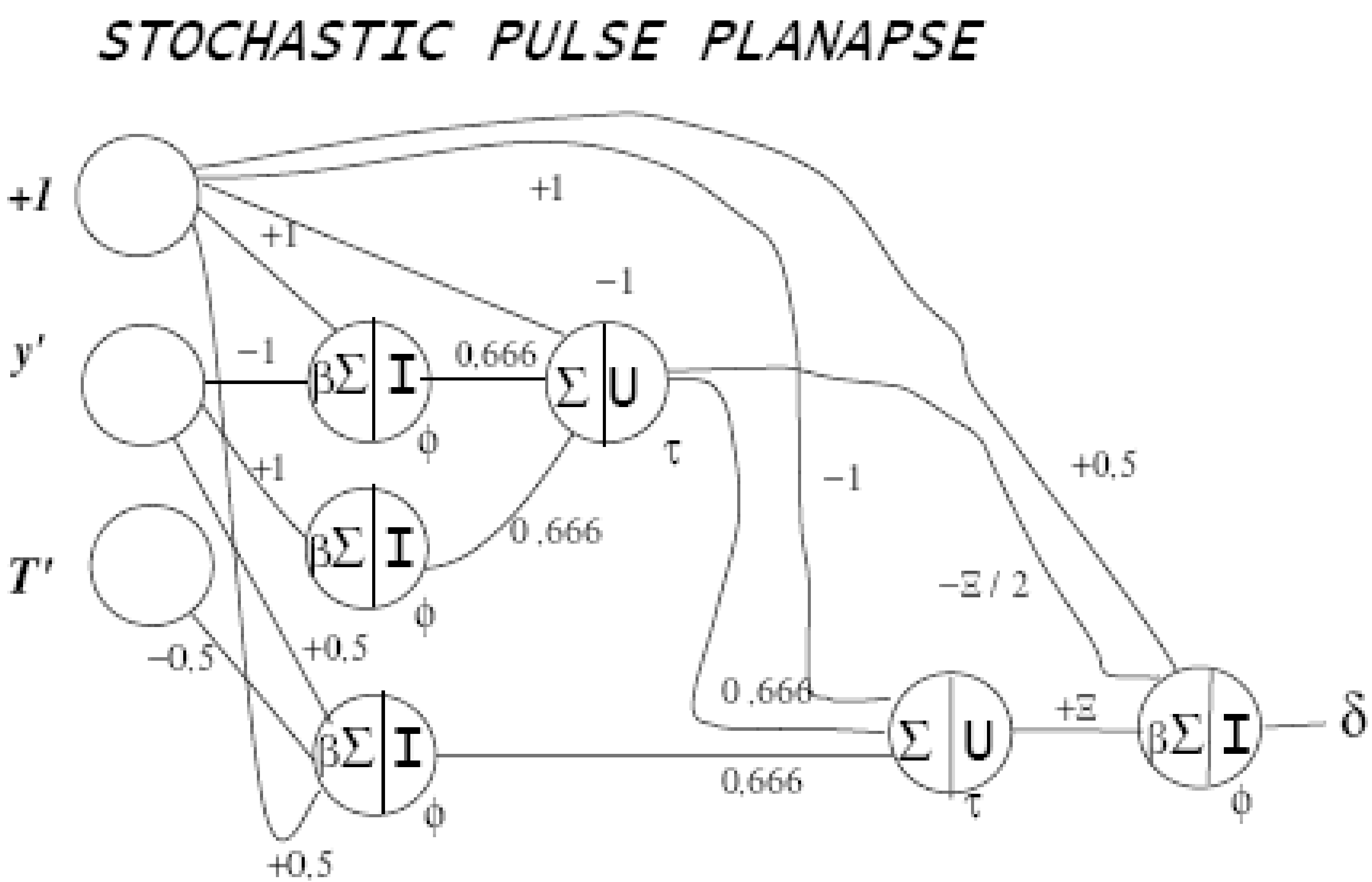


Neural Network NN07aug31Boolean Plotted on 05-Aug-2008



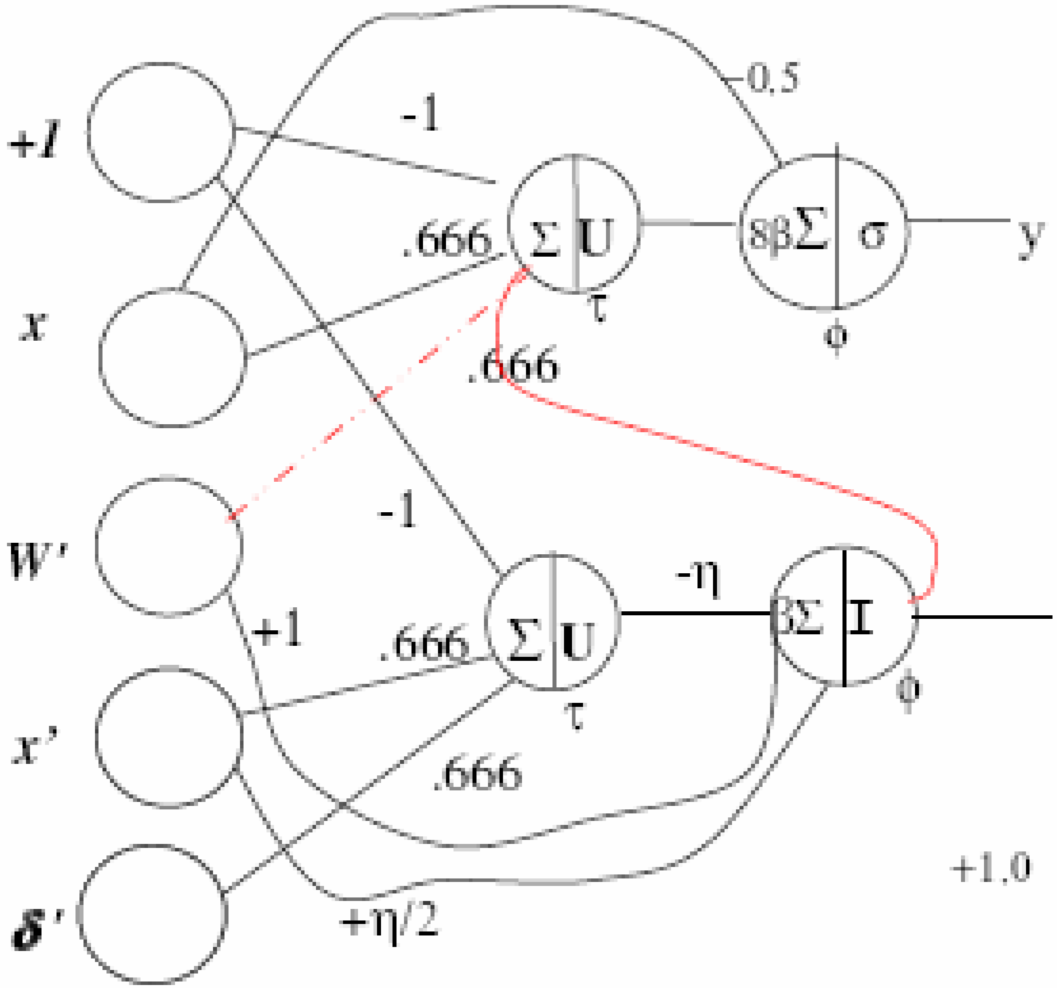
Neural Network NN07feb21TranPlan Plotted on 08-Aug-2008





STOCHASTIC PULSE PLANAPSE
+1
y'
T'
+1
+1
−1
−1
0,666
+1
0 ,666
+0,5
−0,5
+0,5
−1
+0,5
−Ξ / 2
0 ,666
+Ξ
0,666
δ

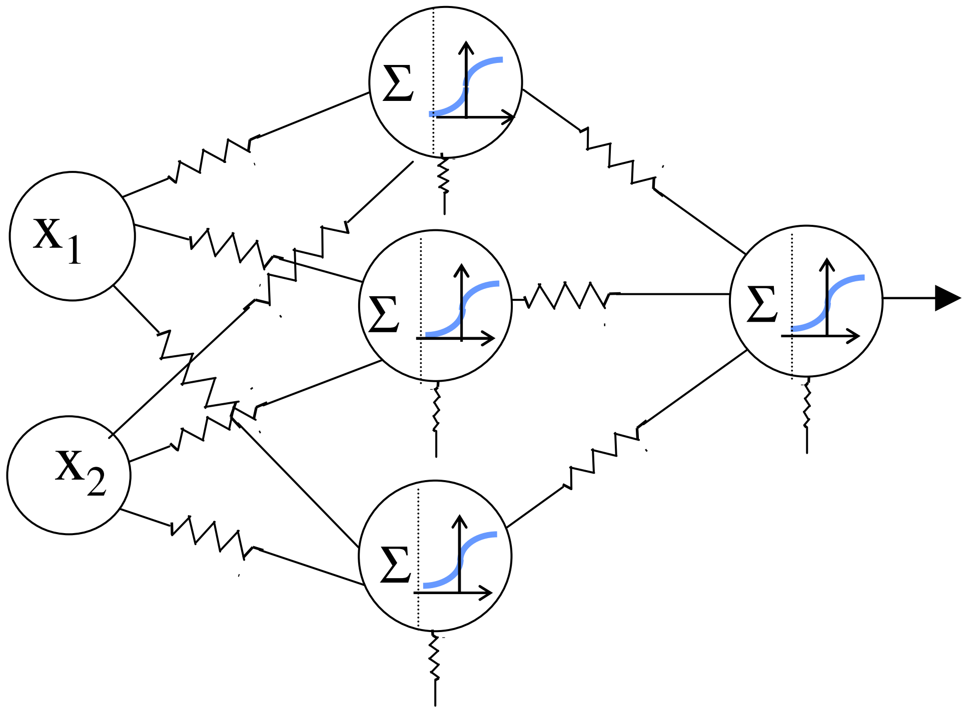
STOCHASTIC PULSE TRANAPSE



Meta-Learning

- Alternative method of deriving FWL-NNs.
- Optimize the (initially randomized) synaptic weights of a FWL-NN to be an efficient learner of function mappings from a given set of mappings.
- Requires optimizing over many examples of many mappings from the given set of mappings.
- Slow to converge, but has derived FWL-NNs that are very efficient learners.

uMULT: Unsigned Multiplication



$$y = x_1 \times x_2$$

Stochastic Pulse Neural Networks

- A stochastic bit-stream is a sequence of equally weighted bits where the probability of each being set is proportional to the value that the bit-stream represents.

Maps rational number x into random variable A with generating probability:

$p = P(A=1) = x.$

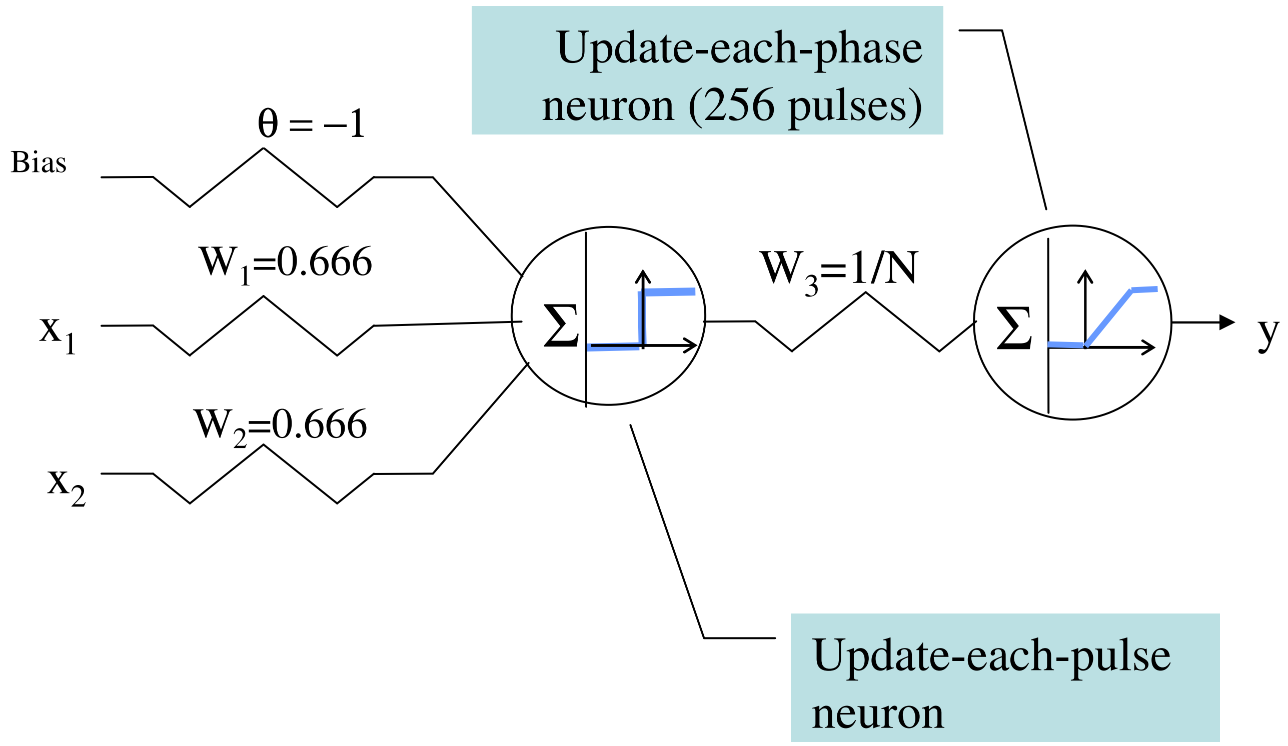
For a signal bitstream of length N, the expected number of bits that are *on* is : $p \times N$

For two such streams, A and B. The probability of a given bit poistion being set for both streams is : $P(A=1) \wedge P(B=1) \leftrightarrow p_A \times p_B = x_A \times x_B$

- That is, multiplication can be done by an AND operation!
- We do the AND with a small squash-each-pulse neural network.

Stochastic Pulse Multiplication NN

$$y = x_1 \times x_2$$



Lessons Learned: Opto-Electronics

- Practical:
  - Windows is not real time. This causes problems. S-l-o-w
  - MATLAB Rules!
  - Intensity must be tightly controlled (drift).
- "Opaque" areas on a slide aren't.
- Slide gray areas are not repeatable enough, even when on/off pixel encoding is used.
  - LCD-based Spatial Light Modulators
  - Laser-cut masks
- Stochastic Pulse Coding works well. It greatly reduces FWL-NN sizes.
- DMD works well, except slow compared to Laser Diodes.
- Software-based alignment and distortion correction works well.
- Diffraction may limit spatial resolution in free-space optics.
- Random noise was not much of a problem.
  - Windows is not real time. This causes problems. S-l-o-w

$$M(t) = M(t-1) - \eta x(t-1) \times y(t-1) \times \left[1 - y(t-1)\right] \times \left[y(t-1) - T(t-1)\right]$$