Net1: (Last week)

- **Macroscopic continuous concentration rates** (rbc)
  - **Cooperativity & Hill coefficients**
  - **Bistability** (oocyte cell division)
- **Mesoscopic discrete molecular numbers**
  - **Approximate & exact stochastic** (low variance feedback)
- **Chromosome Copy Number Control**
- **Flux balance optimization**
  - **Universal stoichiometric matrix**
  - **Genomic sequence comparisons** *(E.coli & H.pylori)*

Net2: Bio-algorithms

- **Biology to aid algorithms to aid biology**
- **Molecular & nano-computing**
- **Self-assembly**
- **Cellular network computing**
- **Genetic algorithms**
- **Neural nets**

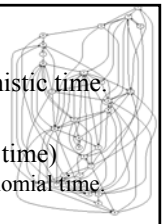
Algorithm Running Time

Given a size *n* problem, an algorithm runs ***O(f(n))*** time:

O(f(n)): upper bound. (Ω :lower θ: equal)

	Time	$n=1$	$n=10$	$n=100$	$n=1000$
Polynomial	n	1	10	10^2	10^3
	n^2	1	10^2	10^4	10^6
	n^{10}	1	10^{10}	10^{20}	10^{30}
Exponential	2^n	2	$>10^3$	$>10^{30}$	$>10^{300}$
	$n!$	1	$>10^6$	$>10^{150}$	$>10^{2500}$

Algorithm Complexity



- P = solutions in polynomial deterministic time:
  - e.g. dynamic programming
- NP = (non-deterministic polynomial time) solutions checkable in deterministic polynomial time.
  - e.g. RS**A** code breaking by factoring
- NP-complete = most complex subset of NP
  - e.g. traveling all vertices with mileage < x
- NP-hard = optimization versions of above
  - e.g. Minimum mileage for traveling all vertices
- Undecidable = no way even with unlimited time & space
  - e.g. program halting problem

NIST UCI

How to deal with NP-complete and NP-hard Problems

- Redefine the problem into Class P:
  - RNA structure Tertiary => Secondary
  - Alignment with arbitrary function=>constant
- Worst-case exponential time:
  - Devise exhaustive search algorithms.
  - Exhaustive searching + Pruning.
- Polynomial-time close-to-optimal solution:
  - Exhaustive searching + Heuristics (Chess)
  - Polynomial time approximation algorithms

What can biology do for difficult computation problems

- DNA computing
  - A molecule is a small processor,
  - Parallel computing for exhaustive searching.
- Genetic algorithms
  - Heuristics for finding optimal solution, adaptation
- Neural networks
  - Heuristics for finding optimal solution, learning,...

Net2: Bio-algorithms

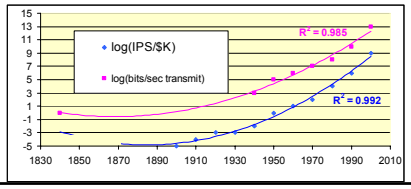
- **Biology to aid algorithms to aid biology**
- **Molecular nano-computing**
- **Self-assembly**
- **Cellular network computing**
- **Genetic algorithms**
- **Neural nets**

Electronic, optical & molecular nano-computing

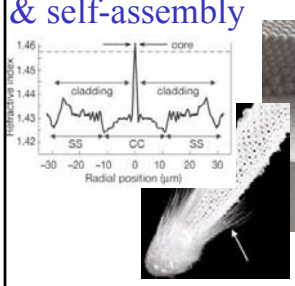
Steps: assembly > Input > memory > processor/math > output

Potential biological sources: harvest design evolve

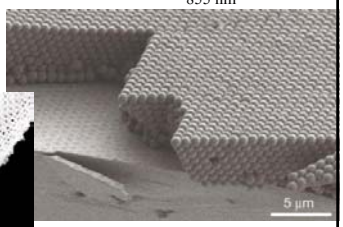
A 30-fold improvement = 8 years of Moore's law



Optical nano-computing & self-assembly



855 nm

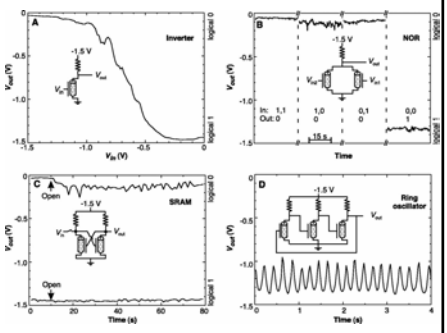


Sundar et al.. Fibre-optical features of a glass sponge. 2003 Nature. 424:899-900.

Vlasov et al. (2001) On-chip natural assembly of silicon photonic bandgap crystals.

Low heat, 10X faster interconnections,

Electronic-nanocomputing

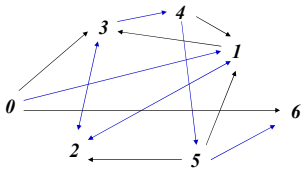


Bachtold et al. & Huang et al. (2001) Science 294: 1317, 1313.

Molecular nano-computing

- R. P. Feynman (1959) American Physical Society, "There's Plenty of Room at the Bottom" (Pub)
- K. E. Drexler (1992) Nanosystems: molecular machinery, manufacturing, and computation. (Pub)
- L. M. Adleman, *Science* 266, 1021 (1994) Molecular computation of solutions to combinatorial problems.
- 727 references (Nov 2002)

DNA computing: Is there a Hamiltonian path through all nodes once?

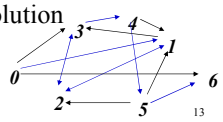


A Hamiltonian path is (0,1,2,3,4,5,6).

L. M. Adleman, *Science* 266, 1021 (1994) Molecular computation of solutions to combinatorial problems.

DNA Computing for a Hamiltonian Path

- Encode graph (nodes and edges) into ss-DNA sequences.
- Create all possible paths (overlapping sequences) using DNA hybridization.
- Determine whether the solution (or the sequence) exists.



13

Encode Graph into DNA Sequences

Nodes => Sequences:

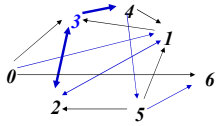
...
2: 5′ TATCGGATCG GTATATCCGA 3′
3: 5′ GCTATTCGAG CTTAAAGCTA 3′
4: 5′ GGCTAGGTAC CAGCATGCTT 3′
...

Edges => Sequences:

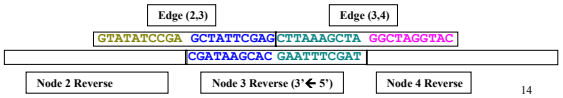
...
(2,3): 5′ GTATATCCGA GCTATTCGAG 3′
(3,4): 5′ CTTAAAGCTA GGCTAGGTAC 3′
...

Reverse-Complement Node:

...
3: 5′ CGATAAGCAC GAATTTCGAT 3′



Edges + Nodes => Path (2,3,4):

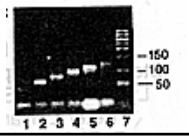


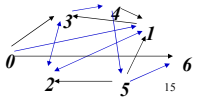
14

DNA Computing Process

- •Encode graph into DNA sequences.
- •Create all paths from ***0*** to ***6***.
- •Extract paths that visit every node.
- •Extract all paths of ***n*** nodes.
- •Report Yes if any path remains

- •Oligonucleotide synthesis
- •PCR
- •Serial hybridization
- •Electrophoretic size
- •Graduated PCR electrophoretic fluorescence





15

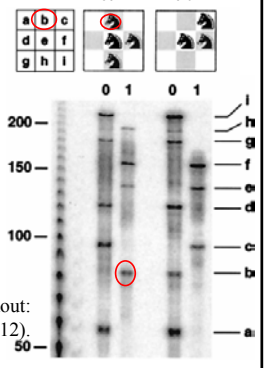
Molecular computation: RNA solutions to chess problems.

Faulhammer, et al. 2000 PNAS 97, 1385-1389. (Pub)

split & pool oligonuc. synthesis
split & pool RNase H elimination

Multiplex colony graduated PCR readout:
42/43 correct solutions (random = 94/512).

two clone solutions: 010011010
= befh efc



$((.h \wedge .f) \vee .a) \wedge ((.g \wedge .i) \vee .b) \wedge ((.d \wedge .h) \vee .c) \wedge ((.c \wedge .i) \vee .d) \wedge ((.a \wedge .g) \vee .f).$

Problems of DNA Computing

- Polynomial time but exponential volumes
- A 100 node graph needs $>10^{30}$ molecules.
- Far slower than a PC.
- Experimental errors:
  - mismatch hybridization
  - incomplete cleavage
- (Some are non-reusable.)

17

Promises of DNA Computing

- High parallelism
- Operation costs near thermodynamic limit
  - 2 vs 34×10^{19} ops/J (10^9 for conventional computers)
- Solving one NP-complete problem implies solving many.
- Possible improvement
  - Faster readout techniques (eg. DNA chips).
  - Natural selection.

18

A sticker-based model for DNA computation.

Roweis et al. J Comput Biol 1998; 5:615-29 (Pub, JCB)

Unlike previous models, the stickers model has a random access memory that requires no strand extension and uses no enzymes.

In theory, ...reusable. [We] propose a specific machine architecture for implementing the stickers model as a microprocessor-controlled parallel robotic workstation…

Concerns about molecular computation (Smith, 1996; Hartmanis, 1995; Linial et al., 1995) are addressed:

1) General-purpose algorithms can be implemented by DNA-based computers
2) Only modest volumes of DNA suffice.
3) [Altering] covalent bonds is not intrinsic to DNA-based computation.
4) Means to reduce errors in the separation operation are addressed in Karp et al., 1995; Roweis and Winfree, 1999).

3SAT

Given n boolean (0/1) variables $x = (x_1, x_2, \ldots, x_n)$, and m 3-variable clauses $c = (c_1, c_2, \ldots, c_m)$, is $c_1 \wedge c_2 \wedge \ldots \wedge c_m$ satisfiable for some x?

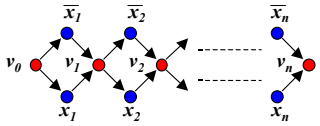
$c_1 = x_1 \vee \bar{x}_3 \vee \bar{x}_7$

$c_2 = \bar{x}_1 \vee x_2 \vee x_4$

…

$c_m = x_1 \vee x_{m-1} \vee \bar{x}_m$

DNA Computing for 3SAT



ALGORITHMS:

1. Encode Graph G into DNA sequences.
2. Create all paths from v_0 to v_n.
3. For every clause
4. Select sequences that satisfy this clause.
5. Report Yes or No.

DNA computing on surfaces

Liu Q, et al. Nature 2000;403:175-9 A set of DNA molecules encoding all candidate solutions to the computational problem of interest is synthesized on a surface. Cycles of hybridization operations and exonuclease digestion identify & eliminate non-solutions.

The solution is identified by PCR and hybridization to an addressed array. The advantages are scalability and potential to be automated (solid-phase formats simplify repetitive chemical processes, as in DNA & protein synthesis). Here we solve a NP-complete problem (SAT) (Pub)

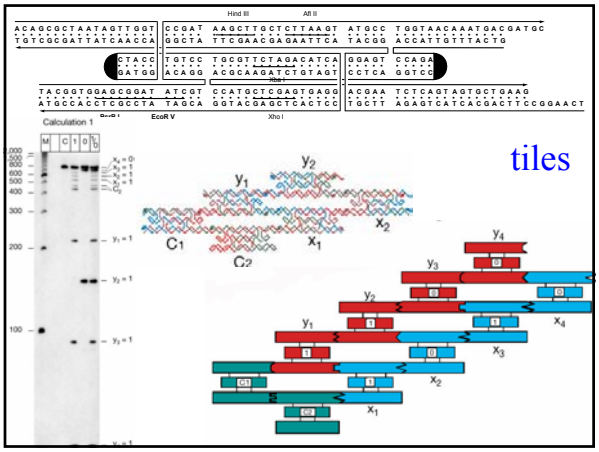
Braich RS, Chelyapov N, Johnson C, Rothemund PW, Adleman L. Solution of a 20-variable 3-SAT problem on a DNA computer. Science. 2002 Apr 19;296(5567):499-502.

Net2: Bio-algorithms

- **Biology to aid algorithms to aid biology**
- **Molecular nano-computing**
- **Self-assembly**
- **Cellular network computing**
- **Genetic algorithms**
- **Neural nets**

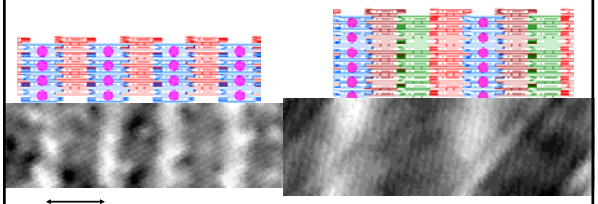
Logical computation using algorithmic self-assembly of DNA triple-crossover molecules.

Aperiodic mosaics form by the self-assembly of 'Wang' tiles, emulating the operation of a Turing machine … a logical equivalence between DNA sticky ends and Wang tile edges. Algorithmic aperiodic self-assembly requires greater fidelity than periodic, because correct tiles must compete with partially correct tiles. Triple-crossover molecules that can be used to execute four steps of a logical (cumulative XOR) operation on a string of binary bits. (a XOR b is TRUE only if a and b have different values) Mao et al. Nature 2000 Sep 28;407(6803):493-6 (Pub)



Nanoarray microscopy readout

(vs gel assays)



~33 nm AFM,
Atomic Force Microscopy

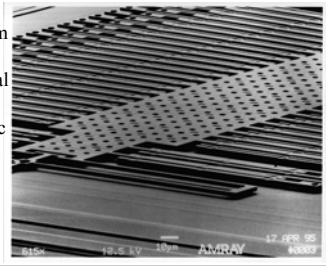
~65 nm

Winfree et al, 1998; Nature 394, 539 - 544 (Pub)

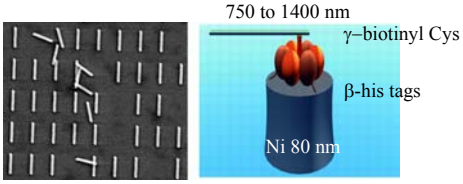
Micro-ElectroMechanical Systems (MEMS)

"Ford Taurus models feature Analog Devices' advanced airbag sensors"

"A unit gravity signal will move the beam 1% of the beam gap and result in a 100fF change in capacitance. Minimal detectable deflections are 0.2 Angstroms; less than an atomic diameter. " (tech specs)

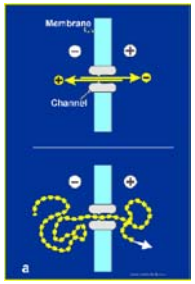


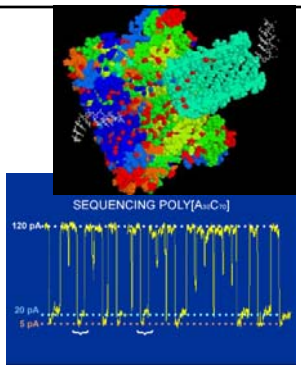
Nano-ElectroMechanical Systems (NEMS)



Soong et al. Science 2000; 290: 1555-1558.Powering an Inorganic Nanodevice with a Biomolecular Motor. (Pub)

Nanosensors

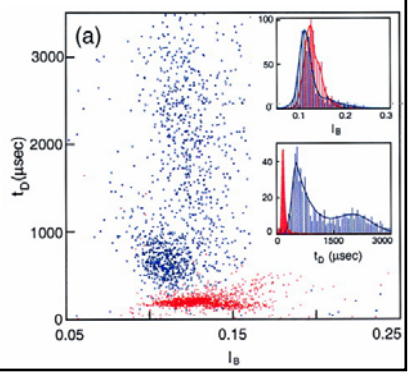




Meller, et al. (2000) "Rapid nanopore discrimination between single polynucleotide molecules." PNAS 1079-84. Akeson et al. Microsecond time-scale discrimination among polyC, polyA, and polyU as homopolymers or as segments within single RNA molecules. Biophys J 1999;77:3227-33

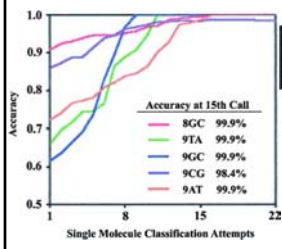
$poly(dA)_{100}$ & $poly(dC)_{100}$ at 15°C

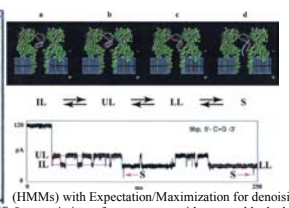
Vercoutere M., et al, Rapid discrimination among individual DNA hairpin molecules at single-nucleotide resolution using an ion channel. Nat Biotechnol. 2001 Mar;19(3):248-52.



Accurate classification of basepairs on termini of single DNA molecules.

- Winters-Hilt et al. 2003 Biophys J. 84:967-76.





(HMMs) with Expectation/Maximization for denoising & associating a feature vector with current blockade of the DNA. Discriminators were multiclass SVM.

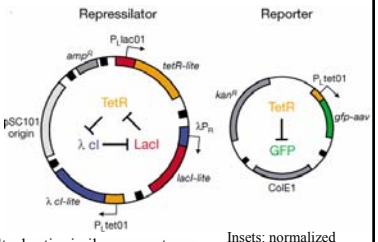
When a 9bp DNA hairpin enters the pore, the loop is perched in the vestibule mouth and the stem terminus binds to amino acid residues near the limiting aperture = *IL* conductance. *b*) When the terminal basepair desorbs from the pore wall, the stem and loop may realign, increase to *UL*. *LL* state corresponds to binding of the stem terminus to amino acids near the limiting aperture but in a different manner from *IL*. *d*) From the *LL* bound state, the duplex terminus may fray, resulting in extension and capture of one strand in the pore constriction (S).

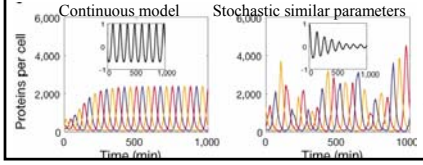
Net2: Bio-algorithms

- **Biology to aid algorithms to aid biology**
- **Molecular nano-computing**
- **Self-assembly**
- **Cellular network computing**
- **Genetic algorithms**
- **Neural nets**

A synthetic oscillatory network of transcriptional regulators

SsrA 11-aa 'lite' tags reduce repressor half-life from > 60 min to ~4 min.



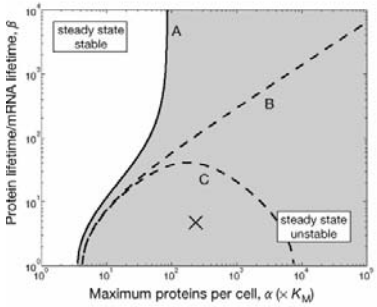


Insets: normalized autocorrelation of the first repressor

Elowitz &Leibler, (Pub), Nature 2000;403:335-8

Synthetic oscillator network

Curves A, B and C mark the boundaries between the two regions for different parameter values: A, $n = 2.1$, $0 = 0$; B, $n = 2$, $0 = 0$; C, $n = 2$, $0/ = 10$-3. The unstable region (A), which includes (B) and (C). A set of typical parameter values, marked by the 'X' in were used to solve the continuous (& stochastic) model in the previous slide.



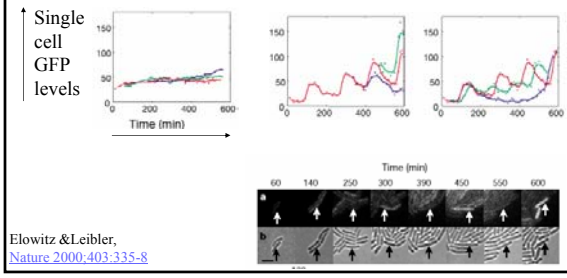
Elowitz &Leibler, Nature 2000;403:335-8

Synthetic oscillator network

Controls with IPTG

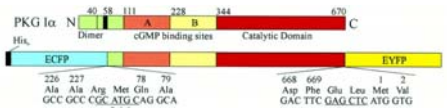
Variable amplitude & period in sib cells

Single cell GFP levels



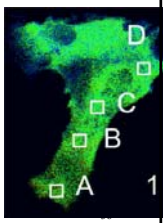
Elowitz &Leibler, Nature 2000;403:335-8

Internal state sensors



Honda et al (2001) PNAS 98:2437-42
Spatiotemporal dynamics of **cGMP** revealed by a genetically encoded, fluorescent indicator.

Ting et al. protein kinase/phosphatase activities



Net2: Bio-algorithms

- **Biology to aid algorithms to aid biology**
- **Molecular nano-computing**
- **Self-assembly**
- **Cellular network computing**
- **Genetic algorithms**
- **Neural nets**

Genetic Algorithms (GA)

1. Initialize a random population of individuals (strings)
2. Select a sub population for offspring production
3. Generate new individuals through genetic operations (mutation, variation, and crossover)
4. Evaluate individuals with a fitness function.
5. If solutions are not found, Go to step 2
6. Report solution.

Genetic Operations

Mutation

...ACCGGTTACGTTGGA...

...ACCGGTTGCGTTGGA...

Crossover

...ACCGGTTTTCGTTGGA...
...CGTACGCCGTTTACCC...

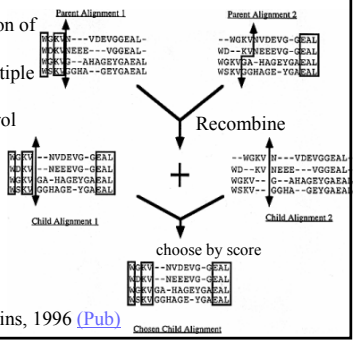
...ACCGGTTTGTTTACCC...
...CGTACGCCTCGTTGGA...

SAGA: Sequence Alignment by Genetic Algorithm

[DP: $O(2^N L^N)$ N sequences length L]

A one point crossover

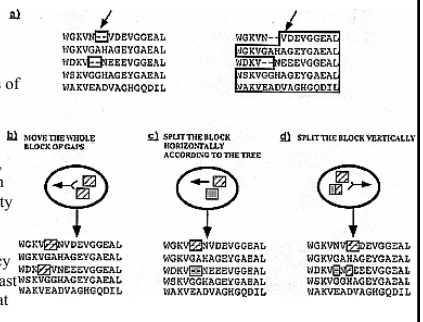
Improve fitness of a population of alignments by an objective function which measures multiple alignment quality, [using] automatic scheduling to control 22 different operators for combining alignments or mutating them between generations.



C. Notredame & D. G. Higgins, 1996 (Pub)

SAGA continues

The 16 block shuffling operators, the two types of crossover, the block searching, the gap insertion and the local rearrangement operator, make a total of 22. Each operator has a probability of being used that is a function of the efficiency it has recently (e.g. 10 last generations) displayed at improving alignments.



Comparison of ClustalW & SAGA

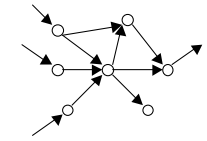
Test case	Nseq	CLUSTAL W versus structure (%)	CPU-time	SAGA versus structure (%)	CPU-time
Igb	32	55.86	60	55.97	41 135
Ac Protease2	10	41.02	16	43.50	12 236
S Protease2	12	64.37	21	66.18	20 537
Globin2	12	94.90	18	94.01	2538

Net2: Bio-algorithms

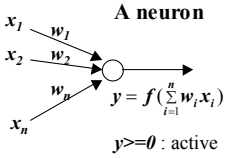
- **Biology to aid algorithms to aid biology**
- **Molecular nano-computing**
- **Self-assembly**
- **Cellular network computing**
- **Genetic algorithms**
- **Neural nets**

Artificial Neural Networks

A neural network:



A neuron



$y>=0$: active
$y<0$: inactive

Neural Networks

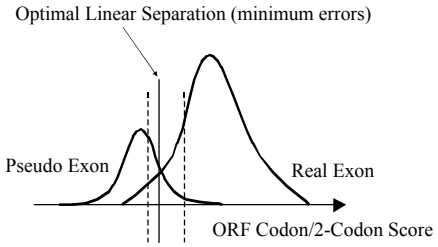
McCulloch and Pitts (1943) Neurology inspired "& /OR"operations

Werbos 1974 back-propagation learning method

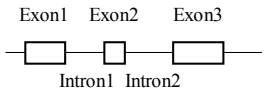
Hopfield 1984, PNAS 81:3088-92 Neurons with graded response have collective computational properties like those of two-state neurons. (Pub)

(ANN)

An ORF Classification Example



Measuring Exons



Exon Features {
Donor Site Score,
Acceptor Site Score,
In-frame 2-Codon Score,
Exon Length (log),
Intron Scores,
…… }

Linear Discriminate Function and Single Layer Neural Network

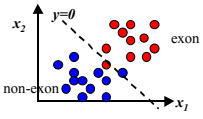
Exon: e=($x_1 x_2 ... x_d$)

A linear separator :

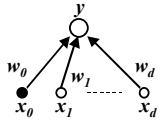
$y = \sum_{i=1}^{d} (w_i x_i) + w_0$

$y > 0$ **: Exon** $y < 0$ **: Non - Exon**

A 2-feature linear separation



Output



Inputs

An activation function :

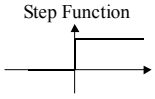
$y = f(\sum_{i=0}^{d} w_i x_i)$

Activation Function

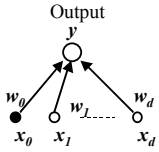
$$f(a) = a$$

$$\begin{cases} f(a) = 0 & a < 0 \\ f(a) = 1 & a \geq 0 \end{cases}$$

Step Function



Output

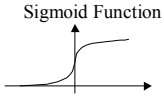


Inputs

$$y = f(\sum_{i=0}^{d} w_i x_i)$$

$$f(a) = \frac{1}{1+e^{a}}$$

Sigmoid Function



Determining Edge Weights from Training Sets

Given a set of *n* known exons/nonexons :

$(\bar{e}_1, t_1), (\bar{e}_2, t_2), \ldots, (\bar{e}_n, t_n)$

Step1 **Initialize *w***

Step2 **Sum of squares error function :**

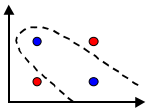
$$E(\bar{w}) = \frac{1}{2} \sum_{k=1}^{n} \{ f(\bar{e}_k, \bar{w}) - t_k \}^2$$

Step3 **Updating w_j**

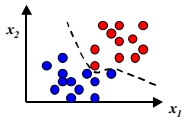
$$w_j^{\tau+1} = w_j^{\tau} - \lambda \frac{\partial E(w)}{\partial w_j}\Big|_{\bar{w}^{\tau}}$$

Non-linear Discrimination

Exclusive-OR Problem



A 2-feature non-linear separation

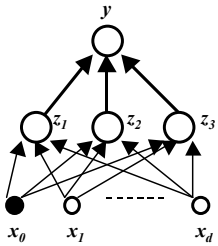


The Multi-Layer Perceptron

Output

Hidden Layer

Inputs



$$y = g(\sum_{i=1}^{3} w_i^{(2)} z_i)$$

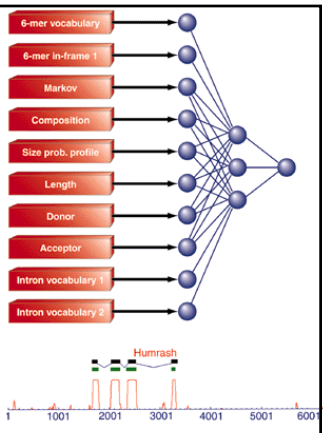
$$z_j = f(\sum_{i=0}^{d} w_{ji}^{(1)} x_i)$$

Training: Error Back Propagation

GRAIL

Located 93% of all exons regardless of size with a false positive rate of 12%. Among true positives, 62% match actual exons exactly (to the base), 93% match at least one edge exactly.

Xu et al, Genet Eng 1994;16:241-53 Recognizing exons in genomic sequence using GRAIL II.
(Pub)



Net2: Bio-algorithms

- **Biology to aid algorithms to aid biology**
- **Molecular nano-computing**
- **Self-assembly**
- **Cellular network computing**
- **Genetic algorithms**
- **Neural nets**