

- **Preprocessing and genetic algorithms**

Dr J.W. MacInnes

June 8th, 2017

Data pre processing

- Neural nets of sufficient complexity can find any nonlinear pattern in our data
- Data preprocessing
 - Can make that job easier (simpler network)
 - Help explain which pattern the neural net found (its related to the aspects you processed)
 - Prevent you from learning ‘unexpected’ associations

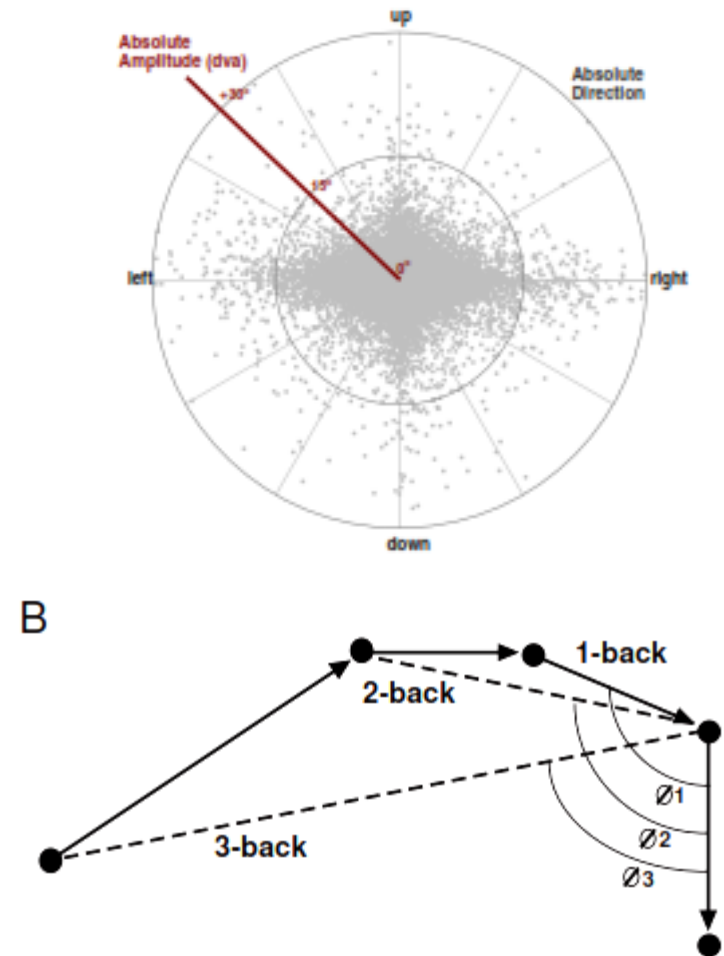
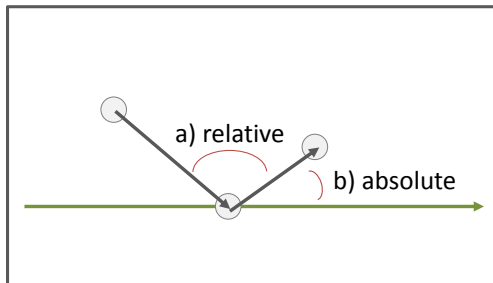
Eye tracking examples

- (see... “Driving forces in free visual search: An ethology”. 2014. WJ MacInnes, AR Hunt, MD Hilchey, RM Klein)
- Eye movement data is a sequence of X-Y coordinates plus saccadic and fixation features (saccadic **velocity**, fixation **duration**...)
- But X and Y locations are artificial measures of screen pixels, not something that’s important to the visual system
 - Although it is loosely related to retinal coordinates
 - And even then only in combination with image saliency properties and other relative locations



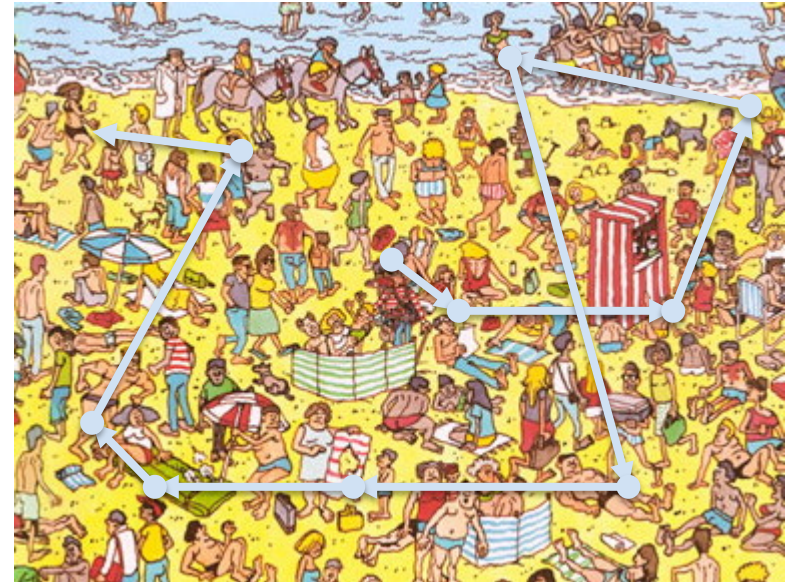
Simple processing

- **Amplitude** and **direction** of saccades in absolute screen coordinates
- **Angle** of saccade
 - **Absolute** and **relative** coordinates
 - Compared to one back, two back, ...
- A time sensitive algorithm could learn these, but why not include them directly if we have theoretical reasons to consider them important (like IOR)



pre-processing

- How do you shoot a purple elephant?
- Search as a sequence of *fixations* and *saccades* over time
- Question: Are there patterns in the sequences?
 - How do we quantify those patterns
- Genetic algorithms are used in Data mining (business) and **genetics** for detecting temporal patterns in sequences
- But these sequences consist of discrete values from a finite set (GATC)



GATCATCGAA.....



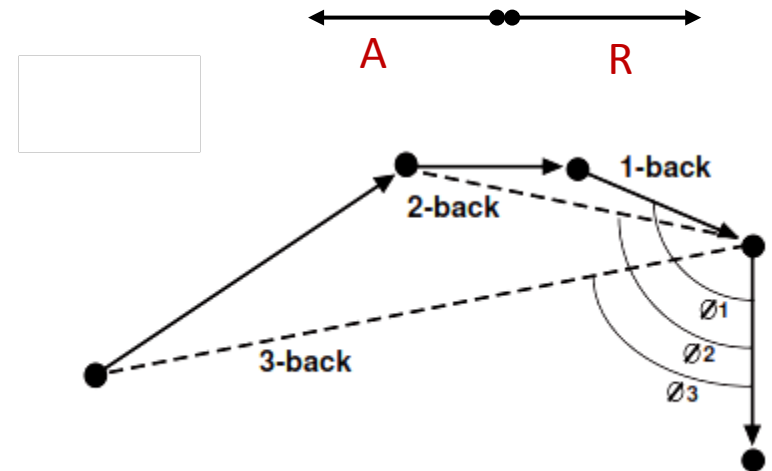
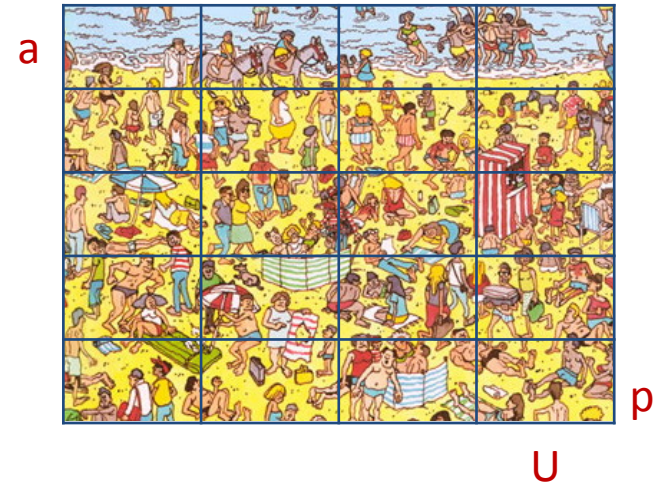
{ATC}; {GA}

Processing Steps

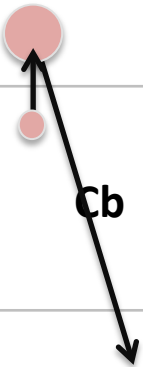
1. Convert key continuous saccade and fixation information to discrete items
2. For large strings, get collective of smaller substrings
3. Establish similarity/distance measure between substrings
4. Search for common substrings
5. Groups of similar substrings

1: Sequences to discrete Strings

- **Absolute Fixation Grid**
 - 50 pixel squares, No FD
 - Global patterns
- **Absolute Saccade Angle**
 - Saccade angle + FD dyads
 - 18 bins of angle, and 20 bins of FD (50ms)
- **Relative Saccade Angle 1B**
 - as above, but relative to 1-back location
- **Relative Saccade Angle 2B**
 - as above, but relative to 2-back location



Aa	Ba	Ca	Da
Ab	Bb	Cb	Db
Ac	Bc	Cc	Dc



Abs Grid (**CbCaCc**)

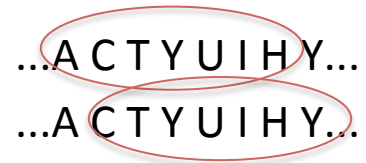
Abs Ang (**HbJe**)

Rel Ang1B (**Be**)

Grid	Dir	FD
Left = A	0°-10° = A	0-100 = a
Right = U	90°-100° = I	900-1000 = j
Top = a	170 – 180° = R	
Bottom = p		

2: Substrings

- Each trial is now strings of 100's of dyads
- Overlapping sliding window?
 - Leads to non meaningful comparison because of 'overlap' similarity (Keogh, 2003)
- Use non-overlapping window or random selection of a subset of possible strings
- Subsequences from non-overlapping sliding window of size 1 – 5 dyads



...ACTYUIHY...

...ACTYUIHY...

The diagram shows two identical strings, "...ACTYUIHY...", one above the other. Each string has a red oval drawn around the substring "ACTYUIHY". The ovals are positioned such that they overlap, with the second oval shifted slightly to the right relative to the first, visually demonstrating the concept of an overlapping sliding window.

Find Similar : Heuristic function

- Many AI problems require a way to measure the quality of different choices
 - We have used Euclidean distance, and city block distance as two examples for spatial distance
 - Cosine similarity measures the angle between two vectors and was used in LSA for semantic similarity
- For this dataset, we need a way to compare strings

3: Distance Measure

- **Needleman/Wunsch** (NW)(1970):
String similarity measurement used
in genetics
- **Cost**
 - insertions, deletions and gaps
- **Count**
 - Final number of matching characters
- **Score**
 - $(\text{Count} - \text{cost}) / \text{string length}$
 - Range -1 +1

1 : AGTY

2 : ASY

1 : AGTY_{Cost 1}

2': A_SY

1 : AGTY

2': A_TY_{Cost 1}
0.1

Count : 1011

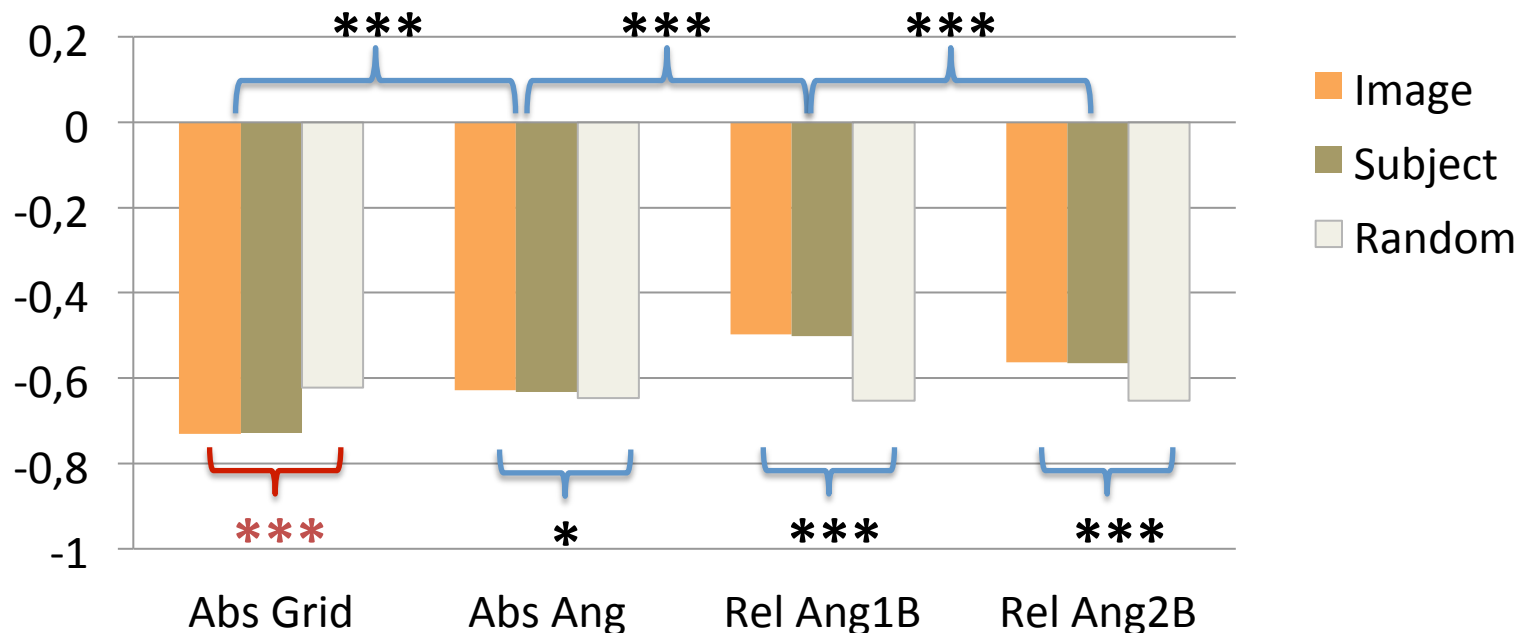
NW SCORE:

= $(3 - 1.1) / 4$

= +0.475

Results: (For anyone interested)

1. Scores are strongly negative, rejecting simple scanning strategies
2. Search data different from random walk
 - Random walk shows **stronger** patterns for absolute grid
3. Local patterns (Angles) are stronger than global patterns (Grid)
4. Relative patterns are stronger than absolute coordinate patterns
5. These patterns are equal combination subject (strategy) and image (salience)



Search for common : Genetic algorithms

- Primarily a search algorithm
- Search for best combination of parameters when there are too many combinations for 'brute force' search
- Requires some measure 'best'
 - Fitness function/heuristic

[http://
www.youtube.com/
watch?v=HgWQ-gPlvt4](http://www.youtube.com/watch?v=HgWQ-gPlvt4)

[http://rednuht.org/
genetic_walkers/](http://rednuht.org/genetic_walkers/)

[https://
www.youtube.com/
watch?v=KjOtNMQxxXo](https://www.youtube.com/watch?v=KjOtNMQxxXo)

Generation x

- What are the key parameters and variables of your model?
 - Define the possible range of values they could take
 - Begin with 50(?) random models, each one possible combination of values
 - This is generation one
1. Test each model of the current generation against some fitness function
 2. Selection: Keep the top 20(?) winning models
 3. Mutate or crossover an additional 20
 4. Add a new set of 10 random models
 5. Return to step 1 as new generation

Generation	N	N+1
Mutate	ARTY	ARKY
Splice	ARTY, GLGS	ARGS, GLTY
Merge	ARTY, GLGS	ARTYGLGS
Random	ARTY, GLGS	EWCF, JYTE

Notes:

Merge is only possible if you allow new generations of different length
Elitism is simply keeping some of the best in a generation 'as is'

4: Search for Common Substrings

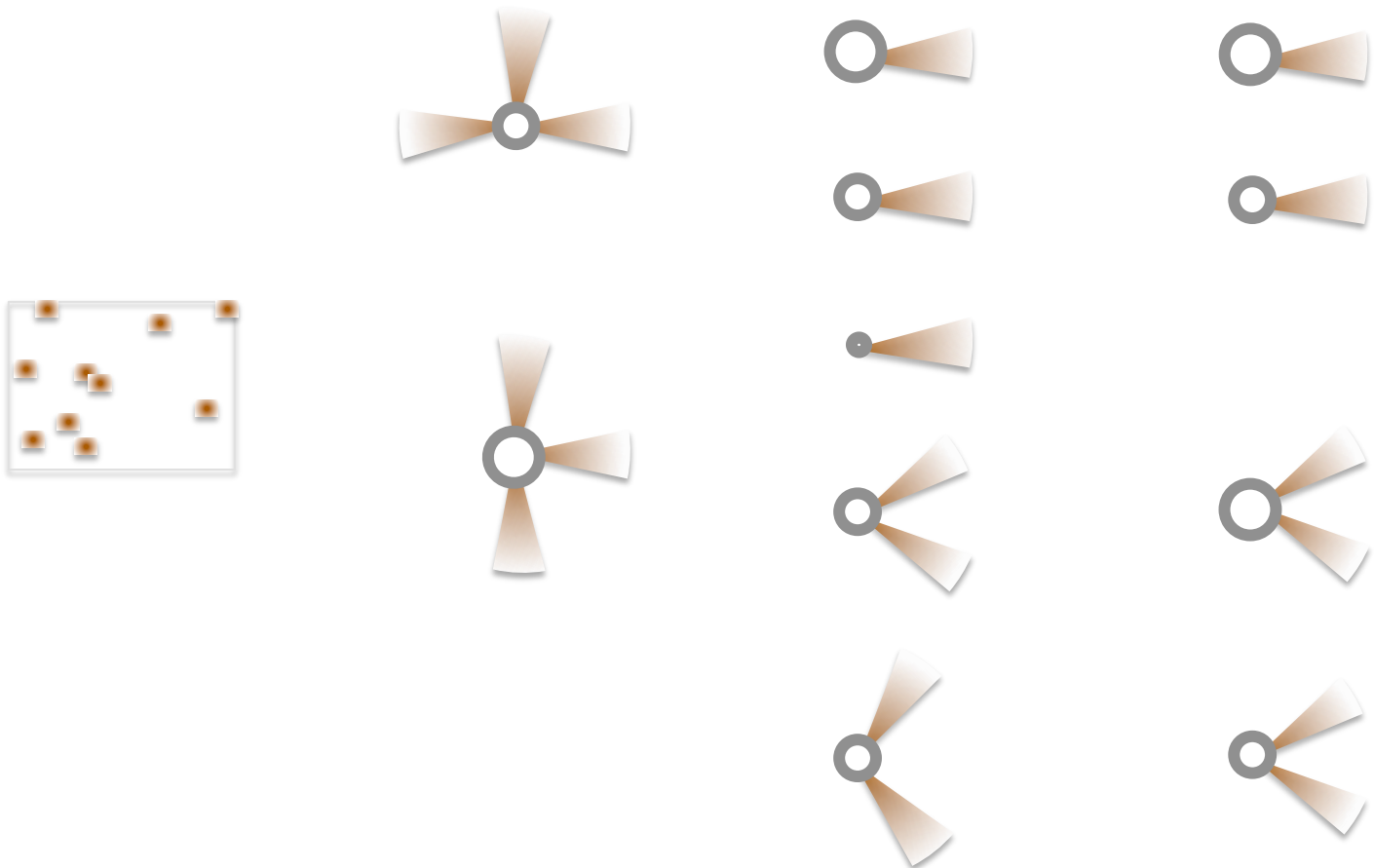
- What do these patterns look like?
- Brute force search is not feasible for strings of this length
- Genetic algorithm
 - a. Select 'good' candidate strings from a probability weight matrix (PWM)**
 - b. Search for candidates in data, and weight/sort by **Needleman Wuncsh**
 - c. Keep top 20 of 50 and fill remaining 30 with mutate, splice, merge and random
 - d. Repeat at b

	Position 1	Position 2	Position 3.
Element a	P()		
Element b			
Element c			

Probability weight matrix lists the likelihood of each element falling in each position. Its useful first step in many algorithms where order matters

Patterns were all single or double saccade

AbsGrid	AbsAng	RelAng1B	RelAng2B
---------	--------	----------	----------



Uses

- Optimize parameters of model
- Search through 'space' of computational models
 - What are the best settings/hidden nodes for your neural net?
- Eye tracking and EEG data, find best input parameters, or sequences of parameters
 - MacInnes, Hilchey, Klein and Hunt, 2014, APP
- Search through expt parameters?
 - Proposed as alternative to Latin squares design

Unsupervised learning

- What if we don't have input/output pairs needed for supervised learning
 - Like back propagation and Bayesian Nets
- Unsupervised learning look for patterns or groups given our data parameters
 - Also called 'clustering'
 - Loosely related to principle component analysis which tries to match variance of original data with reduced dimensions

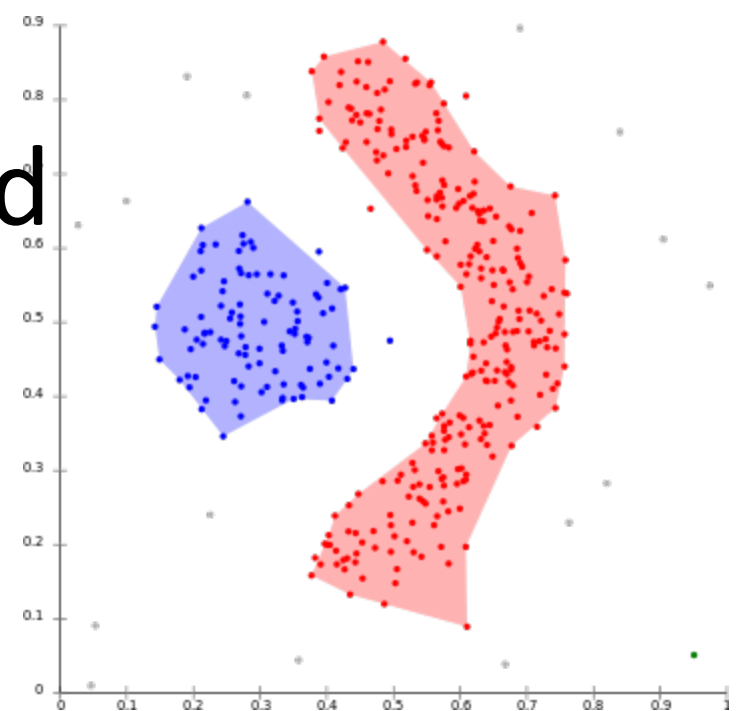
K Means clustering

1. Choose the number of clusters
 2. Choose a distance measure between points
 1. Is data point A closer/more similar to B or C?
 3. Randomly pick point as 'centroid' of each cluster
 4. Assign every data point to its nearest centroid
 5. Each cluster gets a new centroid that is the centre of all assigned points for that cluster
 6. Repeat at 4, until some minimum error is reached
- Fuzzy clustering
 - Points assigned to all centroids, but to varying degree



Variant: density based

- DBScan
 - Density based or spatial path grouping
- Advantages
 - Allows arbitrary shaped clusters
 - No need to specify #clusters
 - Outliers and noise do not affect clusters
- Free toolkit for all clustering algorithms
 - <http://www.mathworks.com/matlabcentral/fileexchange/7486-clustering-toolbox>



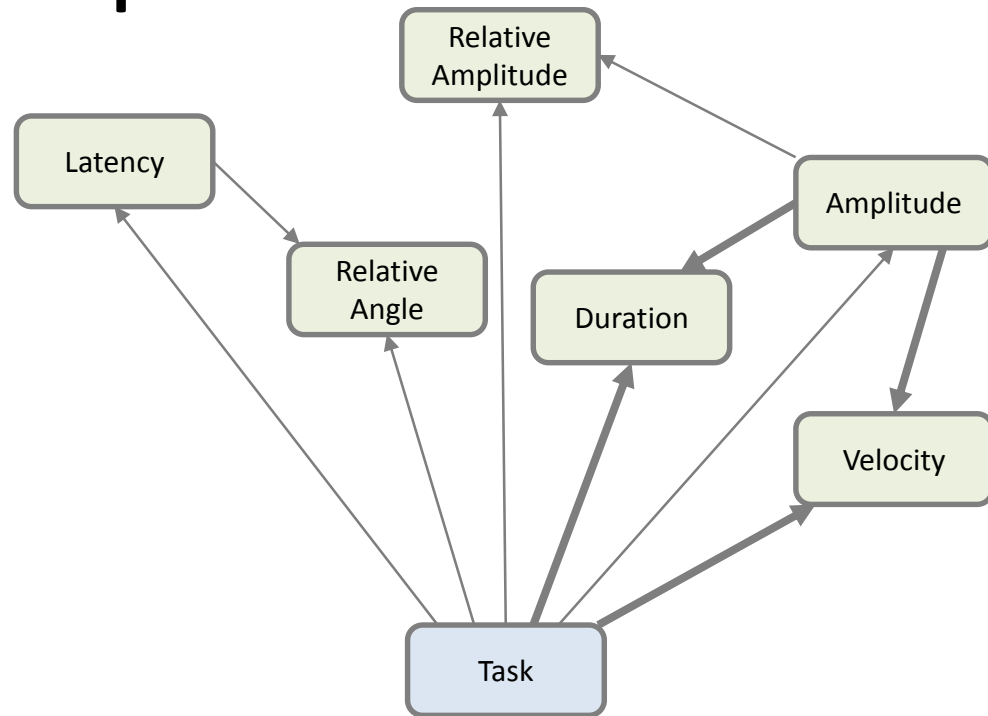
1. Choose a starting point for a new cluster
2. Any point within X distance is added to the cluster
3. While points are near, add them to the cluster and repeat 2 with new points in cluster until no new points are added
4. Once no new points can be added
 1. If there are a minimum number of points, it is a cluster
 2. if not, label them as noise/outliers
5. Go to step one with a new, unvisited random point

Uses

- Levels for a hidden state in a Bayesian network

example

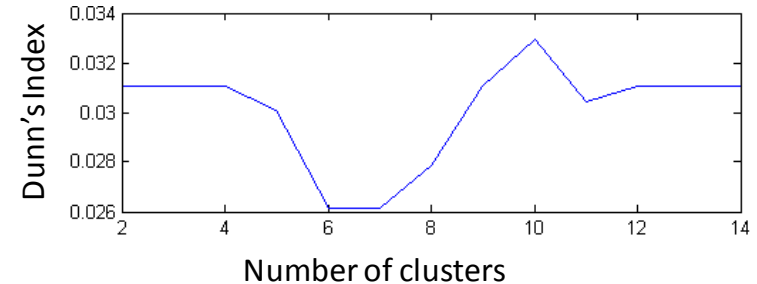
- How does instructed task influence saccades
- Black box model only includes observable input/output
- Clustering of saccadic and pupil properties can help us make an attempt at modelling hidden attentional state



Step 2: find clusters

- Dunn's index helps us choose best number of clusters
- Its a measure of compactness and separateness of clusters
 - Smaller index is better
- Test multiple number of clusters, and choose the one with the smallest Dunn's index
 - In this case, 6

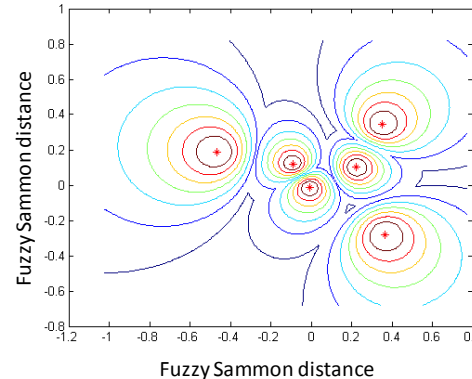
Comparison of Dunn's index for various number of clusters



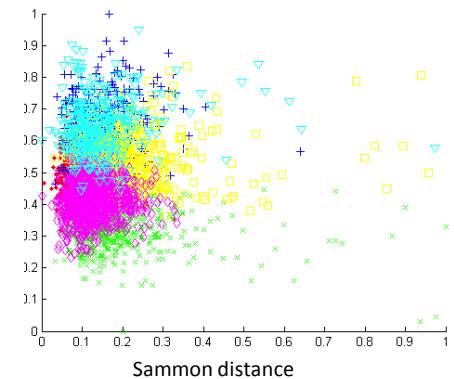
Step 3: sanity check

- Can we visualize the clusters to make sure they 'look' like real clusters
 - Easy for 2D data
- For X-D?
 - How do you shoot a purple elephant?
 - multidimensional scaling like Sammon Mapping
 - Then visualize in two dimensions

c) 2D Sammon projection of Fuzzy cluster distance



b) 2D Sammon projection of cluster distance



$$E = \frac{1}{\sum_{i < j} d_{ij}^*} \sum_{i < j} \frac{(d_{ij}^* - d_{ij})^2}{d_{ij}^*}.$$

d_{ij}^* is distance in high dimension

d_{ij} is distance in low dimension

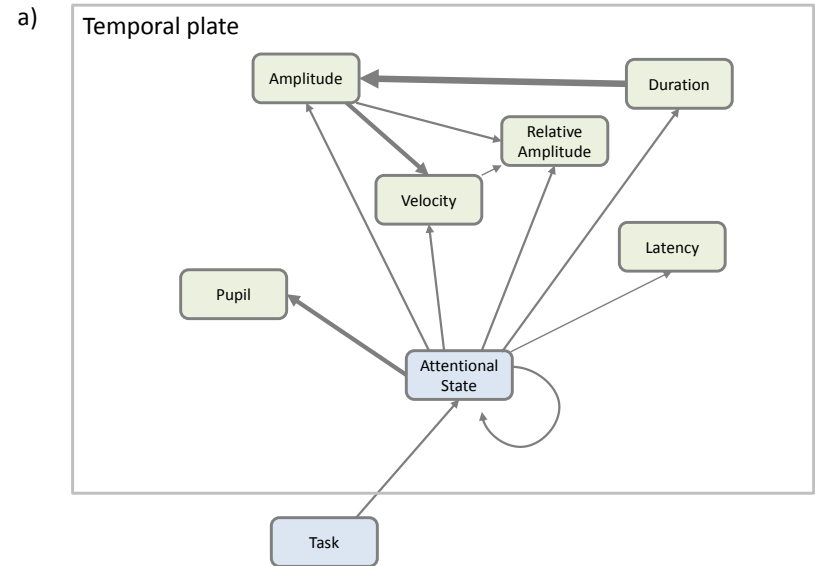
E is the calculated error for a given projection

How can we minimize this error?

Gradient descent!

Result

- New model with hidden attentional state
- Can still be tested using existing data
- Can these attentional state parameters be tested on special populations?



This week

- Advice for those that want to go deeper
 - Code one of these algorithms yourself from scratch (Java, Matlab, C++)
 - AI is learned by DOING
 - These slides provide 'pseudocode'
- More advice on projects and data sets
 - Questions? Examples?

Pseudocode

1. Choose a starting point for a new cluster
2. Any point within X distance is added to the cluster
3. While points are near, add them to the cluster and repeat 2 with new points in cluster until no new points are added
4. Once no new points can be added
 1. If there are a minimum number of points, it is a cluster
 2. if not, label them as noise/outliers
5. Go to step one with a new, unvisited random point

Mixture models/mixture of Gaussians

- If you suspect multiple underlying distributions for your data
- How we determine the originating distribution of an observation given they likely overlap?
- How do we solve it?
 - Expectation maximization

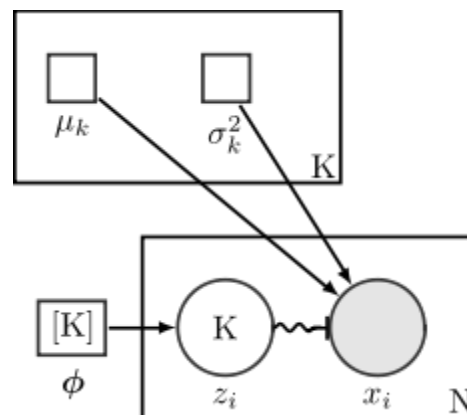
More plate notation!!!

N data points

K number of assumed sub-distributions with some mean and sd

Φ probability of each K distribution (some are more likely than others, and can be thought of as prior probability of that distribution. Must sum to 1)

z_i which of the k gaussians do we assign to (known, shaded) observation **x_i** ?



EM

- Any time you want maximize parameter likelihood
 - Works for statistical models only (those that contain probability distributions, assume sample from larger population, generative models)
 - Dempster, Laird and Ruben, 1977
- Very common algorithm for solving unknown assignment
 - Bayesian networks, Mixtures of Gaussians, temporal algorithms, ...
- ‘Gentle introduction’ for those who want to dig deeper
 - <http://crow.ee.washington.edu/people/bulyko/papers/em.pdf>

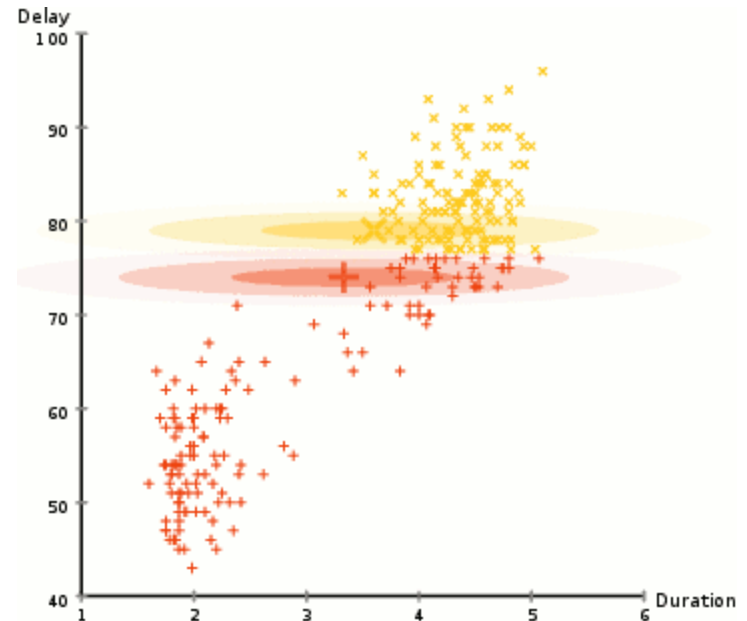
EM algorithm

- Randomize latent/hidden/unknown variables \emptyset
 - For mixture of Gaussians, the latent variable is the distribution to which each data point belongs
- Expectation
 - Compute the best values of all \mathbf{Z} given \emptyset
- Maximization
 - Use the new \mathbf{Z} to compute better estimate for \emptyset
 - Only parameters for a given \mathbf{Z} are used to estimate its \emptyset
- Check log likelihood of solution, stop when it doesn't change much from each iteration
- Improved: Instead of the hard version where a single \mathbf{Z} is chosen for each X , calculate the weighted $p(\mathbf{Z} | \emptyset)$ over entire set of \mathbf{X}

\mathbf{X} – set of observed data

\mathbf{Z} – set of hidden variables

\emptyset – vector of unknown parameters



EM of old faithful data

More on EM?

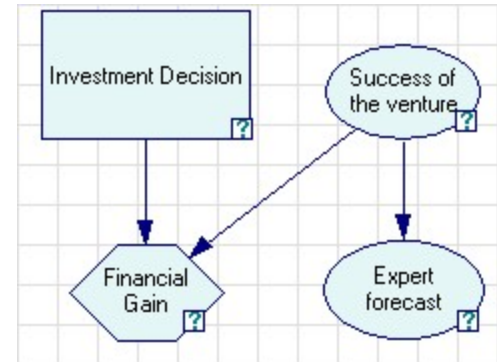
- Bonus reading for those interested (not enough time to discuss in class)
 - This is a powerful algorithm used in a lot of machine learning
- Tutorial on maximum likelihood estimation, 2003, Jae Myung
 - The M step

Utility and decision making

- Decision making needs more than just a representation of the world state
- It also needs an understanding of the quality of choices and actions
- This is often modelled as 'Utility'
 - The expected benefit of a possible result
 - Different from the likelihood of that result
- Influence diagram = Utility + Bayesian probability
 - (AKA decision network)
- Influence diagrams are often about 'optimal decision policy'
 - Its our job to make sure optimal means 'closely tied to the way people do it'

Genie

- 2 extra nodes required
- Decision definition node
 - Simply defines the possible decisions to make
 - Yes, no, maybe
- Value (Utility)
 - Receives connections from decision and a chance node
- Must define value of each decision under each possible chance node result
- Can, of course, be combined with more complex networks
- Note: decision nodes can also be direct parents of chance nodes
 - This is used to reflect ‘sensitivity’ or best and worst case range of probabilities



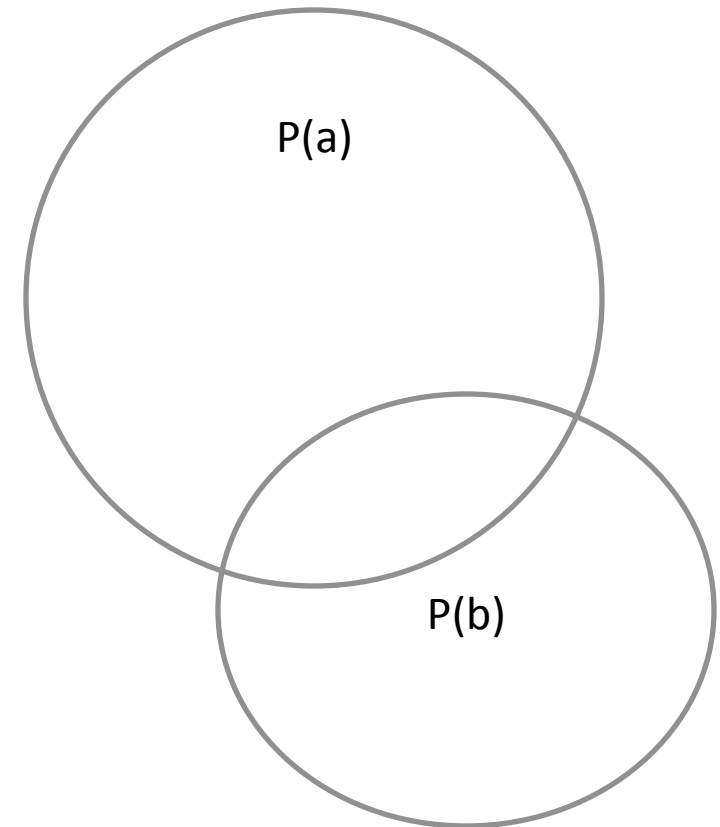
Node properties: Financial Gain

General Definition Format User properties

	Success		Failure	
Investment De...	Invest	DonotInvest	Invest	DonotInvest
Value	10000	500	-5000	500

Utility for fallacies

- Remember your model must reflect actual choices and probabilities, not optimal
- Conjunction fallacy
 - Linda is 31 years old, single, outspoken, and very bright. She majored in philosophy. As a student, she was deeply concerned with issues of discrimination and social justice, and also participated in anti-nuclear demonstrations.
- Which is more probable?
 - Linda is a bank teller.
 - Linda is a bank teller and is active in the feminist movement.
- People consistently estimate the second as more likely
 - Even though $p(a \wedge b) \leq p(a)$
- Other probability theories may inherently work better for
- Pothos, E. M., & Busemeyer, J. R. (2013). Can quantum probability provide a new direction for cognitive modeling?.

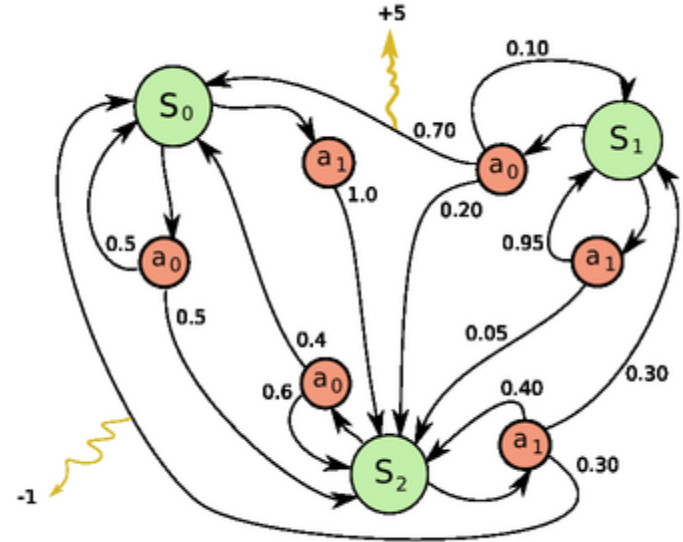


Learning Utility

- Not every problem has nice, labelled 'correct' answers for supervised learning
- Even when feedback exists, it is often delayed
 - Chess is the typical AI example: how does each move contribute to the win?
- Language acquisition might be a better example for us
 - What contributes to long term acquisition and comprehension?
 - Short term repetition and testing may or may not reflect this long term goal
- Reinforcement learning is a branch of learning algorithms to achieve long term reward
- Reinforcement is related to the concept from animal behaviourism
- From an AI agent perspective, reward is a sensory input, but the agent must have an internal mechanism for recognizing this signal as separate from other input

What to learn

- All reinforcement learning is about maximizing reward
- **Utility** based agent learns utility function
 - Stateless. Simply which is best action?
- **Q-learning** finds an action-utility function
 - maximizing utility for each action-state combination
- **Reflex** agents are deterministic
 - Policy for direct mapping from state to action



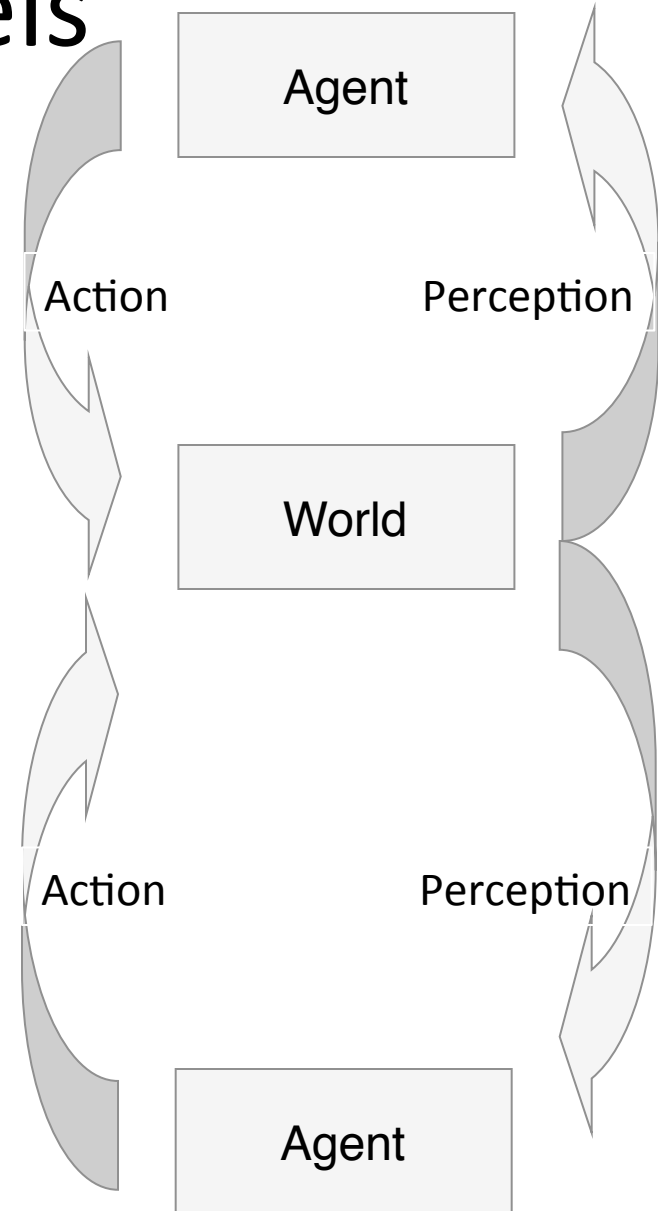
Markov decision process with three states. Similar to DFA, but actions are probabilistic and modelled explicitly. After every state transition, some response from the environment is interpreted as reward.

Q-learning

- No previous knowledge of the world is assumed and all action-utility must be learned
- Exploration and exploitation are both valuable in different situations
- We will cover this in more detail in the temporal algorithm class

Social psychology models

- Autonomous agents are a very useful paradigm here
- Each agent observes the environment (sensors) and can only understand other agents through observing their actions in that environment



Modelling others and TOM

- Theory of mind – ability to see others as rational actors
- Perspective taking – understanding a situation from another's point of view (likely not possible without TOM)
- Recursive modelling
 - Level 0 : model opponent as non-agent; simple object in environment
 - Level 1 : understand opponent's model of world
 - Level 2: model opponent's model of your model of the world
- Modelling deception (Thagard, 1992)
 - Second order modelling is required for deception
 - I need to understand my opponent's likely model of me in order to manipulate it

Mixture of experts (MOE)

- Neural networks (Experts) trained on sub population of the data
- Gating mechanism which chooses between experts
 - Also learned algorithm, perhaps another neural net or classifier
- Not necessarily more accurate than single neural net
 - But experts might have fewer hidden nodes and train faster
 - Could also reflect the what you are trying to model better than black box network
- Hierarchical option exists for different depths of representation
- A similar idea can be implemented in probabilistic models using 'Boosting' or 'Bagging'

Constraint satisfaction

(By popular demand)

- Another search algorithm when brute force is not feasible
- Also useful when its easier to define parameters you *don't* want
- Does not search for optimal values of parameters, just a set of values that satisfy all of the constraints

X - set of parameters or variables

D - the domain for each variable

C - set of constraints

Eg:

$X_1 \neq X_2$

$X_1 == \text{'Fast'}$

Between(X,Y,Z)

'Alldiff'

Examples

- Any efficient search algorithm can be used to find parameters of a statistical model
 - Like Hidden Markov Models
- Weighted constraints could support connectionist learning of symbols
 - Symbolic functions from neural computation (Smolensky, 2012)
 - Thanks to Jon for this one, perhaps more on class with models of language
- As an ‘optimal’ problem solver to compare with human performance
 - Eg ask subject to assemble Ikea furniture, CSP can suggest possible orders
 - $\text{Shelves} \geq \text{Frame} + 10$
 - (shelves must start after the frame is in place and the frame takes 10 minutes)

Algorithm

- At each iteration, the solver can do one of two things
 - Assign a value to a parameter from its domain (this is traditional search)
 - Reduce the domain size of unassigned variables (this is what results in a huge time savings)
- How do we choose when to assign a value to a parameter, which parameter and which value?
 - There are simple path based search algorithms like A* that can be useful here
- *Option*: preferences instead of, or in addition to, constraints
 - These will determine the order that you try parameter values
- *Option*: instead of doing constraint propagation with the search, do it as a pre-processing step
- *Option*: parameters have continuous domains
 - These are usually solved with linear programming

Pseudocode : this is done recursively (not iteratively with loops) or with a *queue*

1. Create a network with edges for each constrained pair
2. Assigning a value for the most connected parameter from its current domain
3. Constraint propagation: Spread through the network reducing the domain of each parameter based on set of constraints
4. If any parameters reduce its domain to zero, back up and try a different previous parameter value
 1. (this step is why you need recursion)
5. If any parameters still have more than one value in domain, go to step 2

Data structures

- This is a small taste of a computer science course, and may help you understand these algorithms a little more
 - Multiple classes are taught on data structures, their uses and the algorithms that use them
 - I will mention only a few basic ones
- The data structure is how you organize the data to make it easier for the algorithm to manipulate and search
 - Some algorithms require specific datasets
- The simplest data structure you already know - the array/vector/matrix
 - Its very efficient to look at value of any element
 - Very inefficient to remove elements
- Some of these things we will talk about can be implemented with vectors and a lot of extra code, but...
 - Having a specialized data structure allows you to focus on problem, and not the low level implementation
 - Tools like this help you think at a more abstract level which makes **mistakes and bugs less likely**

Data structures - Queue

- First in, first out (FIFO)
 - Think of a nicely organized British bus stop
- Add() a new item
- Remove() the current front of the queue
 - Sometimes you'll hear these called push and pop
- Uses:
 - CSP: you no longer have to keep track of which parameters have been visited for each stage of the algorithm
 - Very efficient adding and removing
 - Inefficient reading or finding random element
- Variant: the Stack
 - First in Last out (FILO)
 - Think a stack of pancakes

Data structures - Tree

- Root element is always the first accessed
 - Every other element is accessed by traversing the tree
- Every element, including the root, contains its parameter/value plus links to one or more leaves/children
 - Traversing the tree is done recursively
- Complicated, but very efficient for Add(), Remove()
- If you build the tree with any relationship between children, the any search on that relationship is efficient
 - Child1 <= currentnode <= child2

Recursion consists of a function calling itself.

This can immensely simplify code compared to iterative looping

```
Function TraverseTree(node)
    (do what you want with
     the current node here)
    If HasChildren(node)
        TraverseTree(child1)
        TraverseTree(child2)
    end
```

Call this function with the root node first, it will then traverse the entire tree

This works because *every sub-tree is also a tree*

More psuedocode

- For those that want to try themselves, this is the style of pseudocode you can find in books

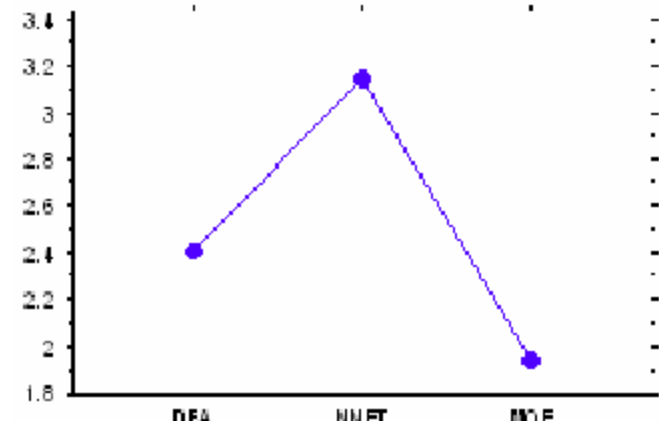
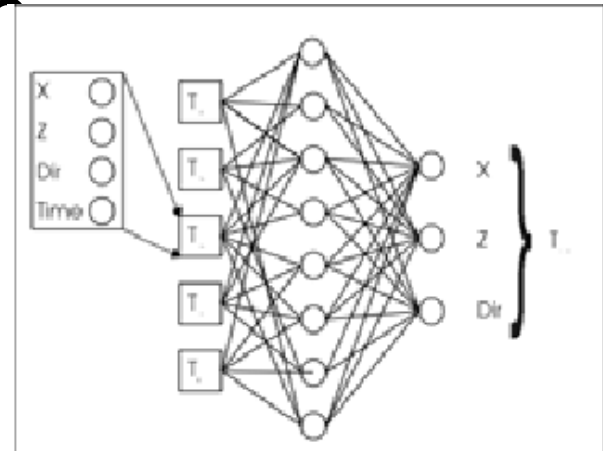
```
function AC-3(esp) returns false if an inconsistency is found and true otherwise
inputs: esp, a binary CSP with components ( $X$ ,  $D$ ,  $C$ )
local variables: queue, a queue of arcs, initially all the arcs in esp

while queue is not empty do
    ( $X_i$ ,  $X_j$ )  $\leftarrow$  REMOVE-FIRST(queue)
    if REVISE(esp,  $X_i$ ,  $X_j$ ) then
        if size of  $D_i$  = 0 then return false
        for each  $X_k$  in  $X_i$ .NEIGHBORS - { $X_j$ } do
            add ( $X_k$ ,  $X_i$ ) to queue
return true
```

```
function REVISE(esp,  $X_i$ ,  $X_j$ ) returns true iff we revise the domain of  $X_i$ 
    revised  $\leftarrow$  false
    for each  $x$  in  $D_i$  do
        if no value  $y$  in  $D_j$  allows ( $x, y$ ) to satisfy the constraint between  $X_i$  and  $X_j$  then
            delete  $x$  from  $D_i$ 
            revised  $\leftarrow$  true
    return revised
```

Test recursive modelling and deception

- Agent/automaton in 3D shooter environment
- Must model opponent to win match and predict location
- Tested three models
 - Neural network
 - Mixture of experts
 - DFA
- Mixture of experts was split by finding clusters of behaviour, then training expert networks
- Four depths of recursion
 - 0..3
- MacInnes, Banyasad and Upal, 1999; MacInnes, 2004; 2006



MOE performed best (lower) for believability and performance. Recursion did not impact either measure

Comparing and testing models

- What do we want from models
 - Classification? Maybe
 - Improved understanding of cognition!
 - Exploration of theories

Too few parameters?

- We know that too many parameters and our models may over-fit
- But what about too few parameters?
- Data from university admissions, California
 - Case was brought up for discrimination

	Men	Women
Applicants	8,400	4,300
admitted	44%	35%

Bias?

Simpson's Paradox

- But no bias for any of the 4 faculties????
- Women tended to apply for departments with very low acceptance rates
- Men applied for departments with much higher acceptance
- Leaving key parameters out of your model will be as problematic as leaving them out of your analysis

	Men		Women	
Dept	Applied	Admitted	Applied	Admitted
A	825	511 (62%)	108	89 (82%)
B	560	353 (63%)	25	17 (68%)
C	325	120 (37%)	593	225 (38%)
D	191	53 (28%)	393	114 (29%)

Individuals as parameters

- New statistical methods treat participants as random variables (LME), why not models?
- Parameters in statistical models (like Bayesian networks) can be tested like any other distribution
 - Including the degree that individual data (subjects) fit that distribution
 - The simplest practical method would be have you model generate a sample distribution and run a t-test
- Be careful of reading individual differences with minimal or single observations per individual
 - Another reminder that subgroups can be an alternative

Comparing and testing models

- Testability /falsifiability
 - Can we identify the ‘correct model’
 - Does it even exist
 - Data fitting merely show a model is *sufficient* to explain the data, not *necessary*
 - Maybe not, and this is not a unique problem to psychology... remember there are an infinite number of models that could describe our solar system
- So why do we try?
 - Because goodness of fit is not the only criteria
 - Because goodness of fit can **rule out** many models
- So some models can be falsified, can all?
 - See Roberts and Pashler (2000)
 - It depends on the predictive scope of the model
- Number of free parameters, in general, increase predictive scope
 - But also increase flexibility/generalizability
 - Trade off?
- Eliminate process models altogether and focus on descriptive models?
 - ‘Rational analysis of behaviour’ – Anderson
- Or use other complementary measures in addition to fit
 - Imaging, theoretical overlap, lesion, special populations, individual differences

Wrong vs useless

- If we can't prove a model correct, does that make it useless?
- How wrong is it?
 - Verisimilitude – a measure of partial truth (popper, 1963)
- ***Does the model make accurate predictions that would have been difficult to predict without the model**
- Does the model **explain** anything
 - Unify desperate processes?
 - Suggest **How** observations arise from the data
 - Its not a very good explanation if we don't understand it
 - Reductionism increases understanding just as it decreases correctness (*cough* model *cough* cat)
- Are there any data that could prove the model *incorrect*?

Neuroscience and correctness

- Imaging data could provide constraints or verify correctness of a model's sequence of cognitive processes
 - 'Where' and 'when' can constrain, but not explain, models of 'how'
 - And remember, for modelling, our ultimate goal is to improve our understanding of 'how'
- To be a proper constraint, imaging data would need to be reliable for subjects across similar trials
 - Good: fMRI BOLD variance is less for within subject than between subject (Bennet and Miller, 2010)
 - Bad: average overlap of significant signal is 29% of voxels (B&M, 2010)

Neuroscience and falsifiability

- For imaging data to constrain a model, the model must consider the neural basis
- Smith and Jonides (1997) suggested their fMRI data supported the Baddeley (1986, ++) model of memory
 - Coltheart (2006) disagreed since the Baddeley model makes no claims about neural correlates
 - There is no data that could have falsified Baddeley's model
 - In this case, its not the imaging data or the model that are necessarily faulty, but the ***interpretation*** of support
- For any data to be said to support a model, there has to be some data of that type that could falsify it

Neuroscience and modelling

- So what is the correct use of neuroscience data?
- Testing the diffusion model

- These are all machine learning algorithms
 - AI vs machine learning
- Agent based models (model perception and action and time)
- Reinforcement learning