Introduction to Learning and Memory

## Introduction

### Last time

Matrix algebra review

Finished up with "outer product", "eigenvectors" of a matrix.  Useful for neural networks and natural computation.

### Today

Linear systems--as an abstraction of matrix operations in linear neural networks. Will provide insight into how to think       about experimental "black box" tests for whether it makes sense to model a neural or behavioral process as a       linear system.

Brief overview of learning and memory

Modeling associative memory

# Linear systems, matrices & neural networks

## Introduction

Consider the generic 2-layer network. Each neuron takes a weighted sum of the inputs (stage 1), followed by a point-nonlinearity (the squash function of stage 2), and added noise (stage 3). Although later we will see how the non-linearity enables computations that are not possible without it, useful functions can be realized with just the linear or stage 1 part of the network. We've seen one application already with the model of the limulus eye. In the next section, we will see how linear networks can be used to model associative memory recall. But first, let us take what we've learned so far about modeling linear networks and look at them in the context of linear systems theory. T

We've seen that the basic linear network is represented by a matrix of weights that operates on a vector of input activities. One property of such a system is that it satisfies the fundamental definition of a "linear system".

The world of input/output systems can be divided up into linear and non-linear systems. Linear systems are nice because the mathematics that describes them is not only well-known, but also has a mature elegance. On the other hand, it is a fair statement to say that most real-world systems are not linear, and thus hard to analyze...but fascinating if for that reason alone. Scientists were lucky with the limulus eye. That nature is usually non-linear doesn't mean one shouldn't familiarize oneself with the basics of linear system theory. Many times a non-linear system has a sufficiently smooth mapping that it can be approximated by a linear one over restricted ranges of parameter values.

So precisely what is a "linear system"?

The notion of a "linear system" can be thought of as a generalization of the input/output properties of a straight line passing through zero. The matrix equation $\mathbf{W}.\mathbf{x} == \mathbf{y}$ is a linear system. There are two basic requirements of a linear system that can be induced from matrix/vector multiplication. Suppose that $\mathbf{W}$ is a matrix, $\mathbf{x1}$ and $\mathbf{x2}$ are vectors, and a and b are scalars. Then we know that:

$$\mathbf{W}.(a\ \mathbf{x}1) = a\mathbf{W}.(\ \mathbf{x}1)$$

This the *scaling* or *homogeneity* requirement for a linear system.

**We noted earlier that neural (and behavioral) responses are often described in terms of small signal suppression, and large signal saturation, leading to the squashing function. Does the squashing function satisfy the homogeneity requirement?**

Another basic property of a linear system is *additivity*:

$$\mathbf{W}.(\mathbf{x}1 + \mathbf{x}2) = \mathbf{W}.\mathbf{x1} + \mathbf{W}.\mathbf{x2}$$

Together these specify the *superposition principle*:

$$\mathbf{W}.(a\ \mathbf{x}1 + b\ \mathbf{x2}) = a\ \mathbf{W}.\mathbf{x1} + b\ \mathbf{W}.\mathbf{x2}$$

This so far is just a consequence of the laws of matrix algebra.

## Generalization

The idea of a linear system has been generalized beyond matrix algebra. This is useful mathematically, but also provides insight into basic experimental tests that psychologists and neuroscientists use to test linearity in a system. Linear systems analysis provides a set of tools for the experimental study of a system. Linearity tests have been applied to behavioral responses, neuron responses (recall the experimental graphs showing linearity and departures from linearity in the slow potentials of a neuron), and neuroimaging responses (e.g. functional magnetic resonance imaging of BOLD signals).

Imagine we have a box that takes inputs such as f, and outputs g = T[f], where f and g could be vectors or continuous functions.

$$f \longrightarrow \boxed{T[\bullet]} \longrightarrow g = T[f]$$
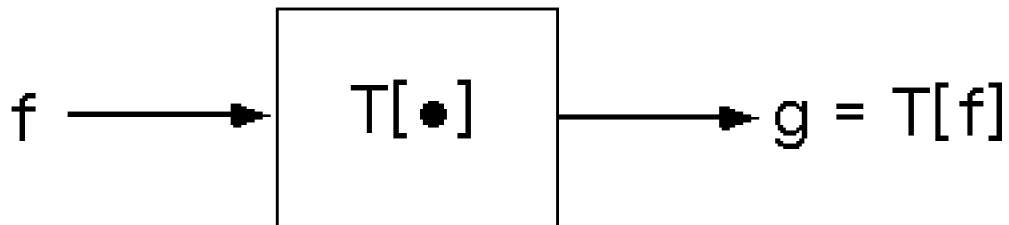
The abstract definition of a linear system is that it satsifies:

$$T[a\, f_1 + b\, f_2] = a\, T[f_1] + b\, T[f_2]$$

where T is the transformation that takes the sum of scaled inputs $f_1$, $f_2$ (which can be functions or vectors) to the sum of the scaled transformation of $f_1$ and $f_2$. The property, that the output of a sum is the sum of the outputs, is the *superposition principle* for linear systems described above. The fact that linear systems show superposition is good for doing theory, is often a good approximation, but as we will see later, it limits the kind of computations that can be done with linear systems, and thus with linear neural network models.

Let's get a graphical view.

■ **Homogeneity**

Suppose we have an input **f1** representing an impulse at time i=33 (i.e. 0s everywhere but at i=33, where i=1). And suppose the measured response is **g1** as shown in the first figure below. This is sometimes called an impulse response (e.g. given an amplifier, a click on a microphone yields a response on the speaker, which isn't exactly the same as the input).

If the system is linear, and in particular respects homogeneity, then if we double the input 2 x f1, the response should be 2 x **g1**.

```
f1 = Table[If[i == 33, 1, 0], {i, 1, 100}];
g1 = Table[If[i ≤ 33, 0, -e^(-1/20 Abs[i-33])] Sin[i/10], {i, 1, 100}];
GraphicsRow[{ListPlot[{f1, g1}, PlotRange → {-0.5, 2}],
ListPlot[{2 f1, 2 g1}, PlotRange → {-0.5, 2}]}]
```
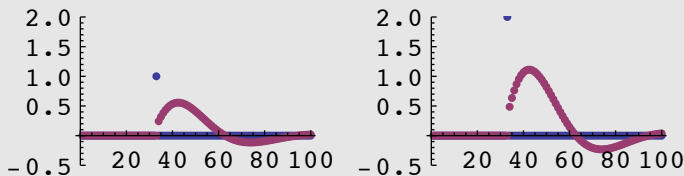


## ■ Additivity

Now imagine an input impulse **f1** at time i=33, followed by another **f2** at time i=45. These lead to responses **g1** and **g2**.

```
f1 = Table[If[i == 33, 1, 0], {i, 1, 100}];
f2 = Table[If[i == 45, 1.5`, 0], {i, 1, 100}];
g1 = Table[If[i ≤ 33, 0, -e^(-1/20 Abs[i-33])] Sin[i/10], {i, 1, 100}];

g2 = Table[If[i ≤ 45, 0, -e^(-1/30 Abs[i-45])] Sin[(i-14)/10], {i, 1, 100}];

GraphicsRow[{ListPlot[{f1, g1}, Filling → Axis, PlotRange → {-0.5`, 2}],
ListPlot[{f2, g2}, Filling → Axis, Filling → Axis, PlotRange → {-0.5`, 2}],
ListPlot[{f1, f2, g1 + g2}, Filling → Axis, PlotRange → {-0.5`, 2}]}]
```
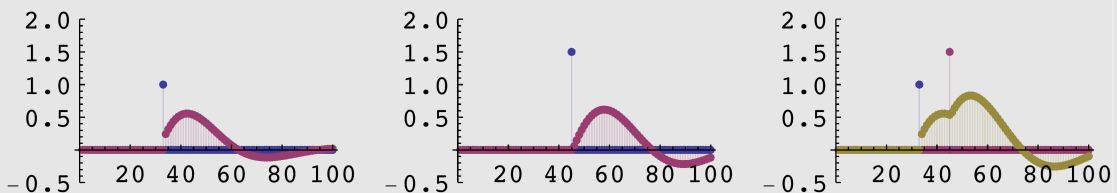


Here's an example of a response that has a non-linear component. The best least squares fit linear prediction is below, in green.

```
ListPlot[{f1, f2, g1 + g2 + 2 (g1 g1) + 2 (g2 g2) + 2 (g1 g2),
   2.087417506355265 * g1 + 2.0162072415081367 * g2}, Filling → Axis,
  PlotRange → {-0.5`, 2}]
```
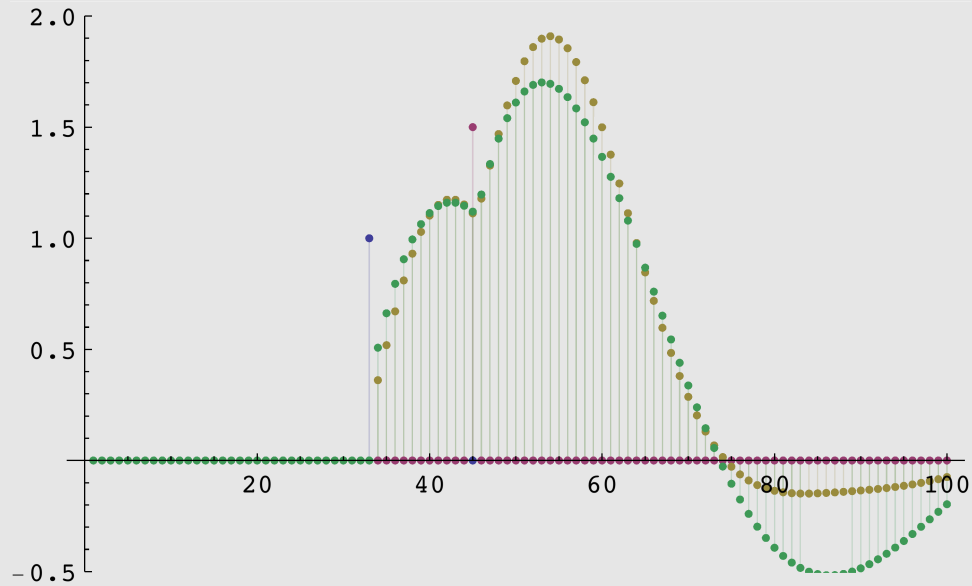


Note that we used the following syntax to multiply two n-dimensional vectors element by element to produce an n-dimensional output:

```
{a, b} {x, y}
```

$\{a\,x,\,b\,y\}$

■ **Note on getting the least squares fit**

```
Clear[a1, a2];
nonlinearoutput = N[g1 + g2 + 2 (g1 g1) + 2 (g2 g2) + 2 (g1 g2)];
bestleastsquaresfit = a1 * N[g1] + a2 * N[g2];
sumsquares = Total[(nonlinearoutput - bestleastsquaresfit)^2];
FindMinimum[sumsquares, {a1, a2}]
```

```
{2.46813, {a1 → 2.08742, a2 → 2.01621}}
```

## Characterizing a linear system by its response to an orthonormal basis set

Suppose we have an unknown physical or biological system, which we model as a linear system, represented by a (square) matrix **T**:

```
T = Table[RandomReal[], {i, 1, 8}, {j, 1, 8}];
```

We would like to make a simple set of measurements that could characterize **T** in such a way that we could predict the output of **T** to any input. This is the sort of task that engineers face when wanting to characterize, say a stereo amplifier (as a model linear system), so that the output sound can be predicted for any input sound. It is also the kind of task that neuroscientists face when describing a neural subsystem, such as hearing, touch or sight.

What kind of measurements would tell us what **T** is? Well, we could make a huge look-up table that enumerates the response to each possible input. That's impractical, and also unnecessary if the system is linear (or is approximately so).

Another thing we could do is "stimulate" the system with cartesian vectors {1,0,0,0,0,0,0,0},{0,1,0,0,0,0,0,0}, and so forth and collect the responses which would be the columns of **T**. Once we have those, we can predict the response to any input. This is in fact the kind of thing illustrated above with pulses at just two of 100 possible time points, i.e. $i = 33$ and $i = 45$.

This is theoretically fine, but can lead to two practical problems:

1) for a real physical system, such as your stereo, or a neuron in the limulus eye, this would require stimulating it with a high-intensity audio or light intensity spike, which could damage what you are trying to study. For example, if you have 1 million receiving elements, getting a response by stimulating only 1 part in a million may require concentrating lots of energy on just one element to get a measurement above noise levels. In the theoretical limit, this corresponds to stimulating it with a "delta" function, a spike which is infinitely narrow and infinitely high.

2) Characterizing the linear system by a matrix **T**, requires $n^2$ numbers, where **n** is the input signal vector length (because each of the n inputs generates an output vector consisting of n numbers)--and **n** can be pretty big for both audio and visual systems.

Problem 2) has a nice solution when **T** is symmetric, and even nicer solution if the rows are shifted versions of each other, as with the model for lateral inhibition (this is addressed later). Problem 1) can be addressed by showing that we can characterize **T** with any basis set--so we can pick one that won't blow out the physical system being tested.

The set of Walsh functions we looked at earlier is just one of many possible basis sets with the advantage that the elements that contribute to the "energy", i.e. (the square of the vector length) are distributed across the vector.

```
v1 = {1., 1, 1, 1, 1, 1, 1, 1}; w1 = v1/Norm[v1];
v2 = {1.,-1,-1, 1, 1,-1,-1, 1}; w2 = v2/Norm[v2];
v3 = {1., 1,-1,-1,-1,-1, 1, 1}; w3 = v3/Norm[v3];
v4 = {1.,-1, 1,-1,-1, 1,-1, 1}; w4 = v4/Norm[v4];
v5 = {1., 1, 1, 1,-1,-1,-1,-1}; w5 = v5/Norm[v5];
v6 = {1.,-1,-1, 1,-1, 1, 1,-1}; w6 = v6/Norm[v6];
v7 = {1., 1,-1,-1, 1, 1,-1,-1}; w7 = v7/Norm[v7];
v8 = {1.,-1, 1,-1, 1,-1, 1,-1}; w8 = v8/Norm[v8];
```

We have already seen that the vector basis set {$w_i$} spans 8-space in such a way that we can express any vector as a linear sum of these basis vectors.

$$g = \sum (g.w_i)\, w_i \qquad\qquad (1)$$

So as you've seen, an arbitrary vector, **g**

```
g = {2,6,1,7,11,4,13,29};
```

is the sum of its own projections onto the basis set:

```
(g.w1) w1  + (g.w2) w2  +(g.w3) w3  +(g.w4) w4  +
(g.w5) w5  +(g.w6) w6  +(g.w7) w7  +(g.w8) w8
```

{2., 6., 1., 7., 11., 4., 13., 29.}

Suppose we now do an "experiment" to find out how **T** transforms the vectors of our basis set, and we put all of these transformed basis elements into a new set of vectors **newW[[i]].**

**newW** is a matrix for which each row is the response of **T** to a basis vector.

```
newW = {T.w1,T.w2,T.w3,T.w4,T.w5,T.w6,T.w7,T.w8};
```

Note that **newW** is an 8x8 matrix.

## Are the rows of newW orthogonal?

So how can we calculate the output of **T**, given **g** without actually running the input through **T**? Note that if we do run the input through **T** we get:

```
T.g
```

{40.7548, 26.3793, 53.13, 37.6783, 49.4476, 58.9351, 33.1965, 42.0629}

We'll use this "answer" as a check for comparison with the "prediction" below using the **newW** matrix.

By the principle of linearity, we can also calculate the output by finding the "spectrum" of **g** as in Problem Set 1, and then scaling each of the transformed basis elements by the spectrum and adding them up:

$$T.g = T.\left\{\sum (g.w_i)\, w_i\right\} = \sum (g.w_i)\, T.w_i \qquad\qquad (2)$$

```
(g.w1) T.w1 + (g.w2) T.w2 + (g.w3) T.w3 + (g.w4) T.w4 +
(g.w5) T.w5 + (g.w6) T.w6 + (g.w7) T.w7 + (g.w8) T.w8
```

{40.7548, 26.3793, 53.13, 37.6783, 49.4476, 58.9351, 33.1965, 42.0629}

Now here's the interesting point. Of course, we have already done our "experiment", so we know what the transformed

basis vectors are, we stored them as rows of the matrix **newW**. We can calculate what the spectrum (**g.w**i) is, so the output of **T** is:

```
(g.w1) newW[[1]] + (g.w2) newW[[2]] + (g.w3) newW[[3]] +
(g.w4) newW[[4]] + (g.w5) newW[[5]] + (g.w6) newW[[6]] +
(g.w7) newW[[7]] + (g.w8) newW[[8]]
```

{40.7548, 26.3793, 53.13, 37.6783, 49.4476, 58.9351, 33.1965, 42.0629}

## Same thing in more concise notation

Let the basis vectors be the rows of a matrix **W**:

```
W = {w1, w2, w3, w4, w5, w6, w7, w8};
```

So again, we can project **g** onto the rows of **W**, and then reconstitute it in terms of **W** to get **g** back again:

```
(W.g).W
```

{2., 6., 1., 7., 11., 4., 13., 29.}

But as before, but more concisely, by calculating the spectrum, and multiplying it by our stored representation of what the system does to the basis vectors-- **newW**, we can predict the output:

```
g.Transpose[W].newW
```

{40.7548, 26.3793, 53.13, 37.6783, 49.4476, 58.9351, 33.1965, 42.0629}

**Show that T==Transpose[newW].W,  and that the system output is thus: Transpose[newW].W.g**

**The main idea is:  *An elegant way to characterize an unknown system T is by  its response to orthonormal input vectors.***

**...even more elegant: *If T is symmetric, then measure the response with respect to the eigenvectors of a symmetric matrix (e.g. T itself).***

These new basis vectors (rows of newW) do span 8-space, but they are not necessarily orthonormal. Under what conditions would they be orthogonal? What if the matrix **T** was symmetric, and we deliberately chose the basis set to describe our input to be the eigenvectors of T?

## What if the choice of basis set is the set of eigenvectors of T?

We've seen how linearity provides us with a method for characterizing a linear system in terms of the responses of the system to a set of basis vectors. The problem is that if the input signals are long vectors, say with dimension 40,000, then this set of basis vector responses is really big--$1.6 \times 10^9$ numbers.

Suppose though that we have a symmetric matrix transformation, **T**. One can show that if vectors of the basis set are the eigenvectors of **T**, $\{e_i\}$, then the transformation of any arbitrary input vector **x** is given by:

$$T[x] = \sum (x.e_i) \, \lambda_i \, e_i = \sum \alpha_i \, \lambda_i \, e_i \qquad (3)$$

Where the $\alpha_i$ are the projections of **x** onto each eigenvector ($\alpha_i = \mathbf{x}.\mathbf{e}_i$). Having the eigenvectors of **T** enables us to express the input and outputs of **T** in terms of the *same basis set--the eigenvectors*. All **T** does to the input is to scale its projection onto each eigenvector by the eigenvalue for that eigenvector.

The set of these eigenvalues is sometimes called the *modulation transfer function* because it describes how the amplitude of the eigenvectors change as they pass through **T**.

Linear systems analysis is the foundation for *Fourier analysis*, and is why it makes sense to characterize your stereo amplifier in terms of frequency response. But your stereo isn't just any linear system--it has the special property that if you input a sound at time t and measure the response, and then you input the same sound again at a later time, you get the same response, except of course that is is shifted in time. It is said to be a *shift-invariant* system. The eigenvectors of a shift-invariant linear system are sinusoids. Further, a linear shift-invariant system can modeled as a convolution, as in the lateral inhibition network.

(Recall the eigenvectors of the symmetric matrix in the optional exercises of Lecture 6? They were sinusoids, not just because the matrix was symmetric, but also because each row of the matrix was a shifted version of the previous row--the elements along any given diagonal are identical. This is called a symmetric circulant--special case of Toeplitz--matrix.)

Sinewave inputs are the eigenvectors of your stereo system. The dimensionality, of course, is much higher--if you are interested in frequencies up to 20,000 Hz, your eigenvector for this highest frequency would have least 40,000 elements-- not just 8!

As mentioned in the introduction, this kind of analysis has been applied not only to physical systems, but to a wide range of neural sensory systems. For the visual system, linear systems analysis has been used to study the cat retina (Enroth-Cugell and Robson, 1964), the monkey visual cortex, and human contrast sensivity system as a whole (Campbell and Robson, 1968).

Much empirical analysis has been done using linear systems theory to characterize neural sensory systems in audition and touch, and other neural systems such as those for eye movements. It works wonderfully as long as the linear system approximation holds. And it does do quite well for the lateral eye of the limulus, X-cells and P-cells of the mammalian visual system, over restricted ranges for so-called "simple" cells in the visual cortex, among others. The optics of the simple eye is another example of an approximately linear system--one can predict the image distortions on your fovea using linear systems theory. Many non-linear systems can be approximated as linear systems over smooth subdomains.

■  **In summary:**

If **T** has n distinct orthogonal eigenvectors, $\mathbf{e}_i$, and known eigenvalues, $\lambda_i$, then to calculate the response to an arbitrary input **x**, do the following:

Step 1: Project **x** onto eigenvectors of **T** to get: $\mathbf{x} \cdot \mathbf{e}_i$

Step 2: Scale each $\mathbf{x} \cdot \mathbf{e}_i$ by the eigenvalue of $\mathbf{e}_i$: $\lambda_i(\mathbf{x} \cdot \mathbf{e}_i)$

Step 3: Scale each $\mathbf{e}_i$ by $\lambda_i \mathbf{x} \cdot \mathbf{e}_i$: $(\lambda_i \, \mathbf{x} \cdot \mathbf{e}_i)\mathbf{e}_i$

Step 4: Sum these vectors up. That's the response of T to **x**! : $\sum_i (\lambda_i \, \mathbf{x} \cdot \mathbf{e}_i)\mathbf{e}_i$

So the eigenvectors are presumed fixed (i.e. sinusoidal input vectors are just part of our standard toolbox we can apply as inputs to any system). We test our unknown, say n x n, system **T** to measure the eigenvalues, which may be different from one system to the next. To predict the response to **x,** we get another tool out of our toolbox (e.g. "spectrum analyzer") that gives us the spectrum of **x**, another set of n numbers. The product of the eigenvalues and the spectral values tell us the length of each eigenvector for the output.


# Learning and memory: Brief overview

It is curious to note that historically, the simple linear model that we will discuss came after much research had been devoted to non-linear learning models, based in particular on a special case of McCulloch-Pitts neurons, used in a system called the Perceptron.  Linear models have limitations,  in particular a consequence of the superposition principle discussed above. Nevertheless,  many of the interesting properties of non-linear systems can be understood in terms of small signal linearity properties.  So with that preamble, let's take a look at memory and see where linearity applies, and how far we can get with a linear model. But first some definitions, and a general overview.


## Definition of learning and memory

Memory is the adaptive change in the behavior of an organism as a conseqence of past experience. More precisely one can distinguish learning and memory:

Learning has to do with aquisition, and memory with storage and retrieval of information. Memory recall is the retrieval of information.

# Psychology and biology of learning and memory

## ■ Associative and non-associative memory

We are going to be talking about *associative memory*, so it is useful to bear in mind that not all memory is considered associative.

**Associative memory:** animal learns about the relationship of one stimulus to another or about the relationship between a stimulus and the organism's response, or the association of an input stimulus with a reward.

**Nonassociative memory:** exposure to a single stimulus either once, or repeated offers opportunity for learning about it

## ■ Examples of nonassociative learning

**Habituation** (Ivan Pavlov)

decrease in behavioral reflex response to repeated non-noxious stimulus

**Sensitization** (pseudo-conditioning)

exposure to noxious stimuli increases sensitivity

**More complex examples** of nonassociative learning

memory for sensory record...although what one calls a stimulus vs. response is problematic

imitation learning

## ■ Associative learning

**Classical conditioning** (Ivan Pavlov)

>           (tone) -- food -- salivation
>
>           (conditioned stimulus--unconditioned stimulus--response)
>
>           CS -- US -- R
>
>           tone -- air puff -  eye blink
>
>           Temporal contiguity of CS-US, and frequency important, but not the only factors

**Operant conditioning** (Thorndike of Columbia U.)

>      also called instrumental or trial-and-error learning
>
>       e.g. hungry rat in a cage with a lever

        occasionally the rat may press the lever out of curiosity, accident or whatever, but if it does and promptly receives some food (a "consequence", which in this case is a reward or a positive reinforcement), the lever pressing (called the "operant") will increase above a spontaneous rate (the rate in the control state where there is no reward).

        In general, consequences can involve positive or negative reinforcement (take away an ongoing unpleasant stimulus), or positive or negative punishment (take away a pleasant stimulus).

## ■ Animal learning

Conditioning is dependent on the degree to which the US (e.g. food) is expected (e.g. Kamin's 2 part experiment)

>           1. light - shock
>
>                 measure strength of conditioning by how the light affects on-going behavior
>
>           2. (light-tone pair) - shock
>
>           3. test with tone alone, not effective (i.e. no suppression of ongoing behavior)
>
>           "blocking phenomenon"
>
>           (Rescorla-Wagner is an elegant model that can account for blocking and other learning behavior. Later

we'll look at its neural network equivalents in the so-called ``delta-rule'' or Widrow-Hoff rule for learning. But below, we first describe the simple Hebbian learning rule).

Ecological constraints on learning

■ **Human memory**

■ **Stages of memory**

Classical view from cognitive psychology -- stages of memory

**Iconic** -- e.g. visual afterimages (1 sec)

**Working memory** (short-term memory, minutes to hours)

small capacity, disrupted by being knocked unconscious
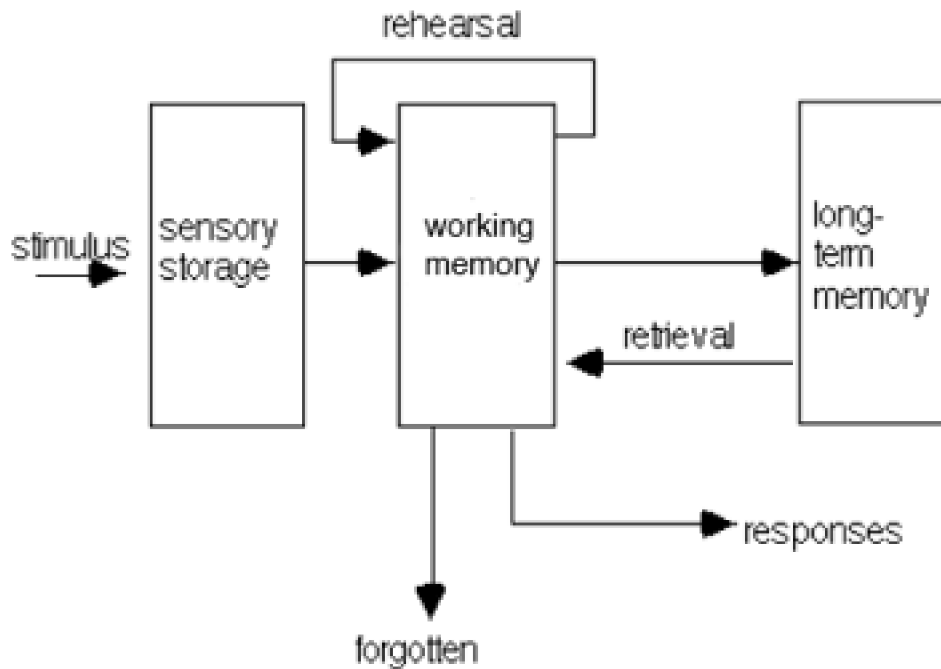
short-term neural plastic events

possible mechanisms?

rapid synaptic modification?

reverberating circuits?

**Long-term memory**

large capacity, relatively permanent (years)

■ **Implicit (reflexive) vs. explicit (declarative) memory**

   **reflexive** -- automatic readout not dependent on awareness, or cognitive processes of comparison, evaluation

   **declarative** -- "I saw a yellow canary yesterday" relies on inference, comparison, and evaluation

Learning to driving a car -- memory moves from declarative towards reflexive with experience.

In our consideration of models, we will be primarily concerned with simple reflexive associative memory in which an neural input event leads to the reconstruction of a predictive response based on experience.

■ **Theoretical/machine learning perspective**

Types of learning

*Unsupervised*

   discover regularities in an ensemble of inputs

      Example: correlations between image pixels or sound samples, efficient coding, PCA in statistics, autoassociation in neural networks

*Supervised Learning*

   learning with a teacher, where the desired outputs are known.

      "Try to do this....ok, now this is the right answer, see how close you were?"

   weights are adjusted to minimise the difference between desired & actual output

      - Example: regression in statistics, multi-layer networks with error backpropogation learning rule, heteroassociation in neural networks

*Reinforcement Learning:* sensation, action, and goal

   "Reinforcement learning is learning what to do--how to map situations to actions--so as to maximize a numerical reward signal."-- Sutton, R. S., & Barto, A. G. (1998)

   Learning with a "critic". e.g. "You did really well" or "You really messed up on that one"

   Could be considered a form of supervised learning, but more representative of how biological learning occurs, and has distinct challenges and characteristics: "Trial and error search", exploration/exploitation trade-off, delayed reward

# Linear model of associative memory

Knowledge is represented in a neural network in the connections of the neurons. Knowledge is acquired/learned through experience that changes in the weights of the connections.

## Hebbian rule for synaptic modification

### ■ Introduction: Modeling associative learning and memory

The brain has developed mechanisms that allow the animal to distinguish events that reliably and predictably occur together from those that do not. What kinds of neural models could be used to capture and retrieve associations? How can one pattern of neural activity come to be associated with another?

**Assumptions:**

• physical basis of memory is in synaptic modification which alters the mapping of inputs to outputs

• the strength of the modification is determined by how often input and output activity occurs together, and by the strengths of the input and output activities.

The idea of learning as association goes back to William James (1890) (See Anderson).

"When two elementary brain processes have been active together or in immediate succession, one of them on recurring, tends to propagate its excitement into the other (Psychology: Briefer Course)."

James also has an "activitation equation" that sounds a lot like our stage 1 of the 2-layer feedforward network:

"The amount of activity at any given point in the brain-cortex is the sum of the tendencies of all other points to discharge into it"

Replace "point" by "neuron", and we have our limulus equation.

There are similar statements elsewhere (e.g. Kenneth Craik of Cambridge in the 1940's).

But the clearest and most explicit statement of an associative learning rule is credited to Canadian Psychologist Donald Hebb:

"When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells, such that A's efficiency as one of the cells firing B, is increased" (Hebb in the "Organization of behavior", 1949).

Until recently, there was little evidence to support this notion of synaptic modification. But direct tests of Hebbian conjecture have now been made in rat hippocampus. (See Anderson, chapter 6 for a review.)

### ■ Hebbian synaptic modification: Modeling use the outer product

The fundamental Hebbian assumption is that synaptic strength grows with co-activity of pre- and post-synaptic activity. How can we quantify this? We will use a simple model of synaptic modification that assumes that a change in connection strength between neuron i and j is proportional to the product of the input and output activities.

$$\Delta W_{ij} = \alpha \, f_i g_j$$

If you remember the definition of the outer product of two vectors, you can see that the above rule is just that--an outer product of the input and output activities.

## The linear model

### ■ Heteroassociation vs. autoassociation

We will look at two types of associative memory models. In heteroassociation, an input **f** is associated with **g**. So at some later time, when the system is stimulated with **f**, it should produce **g**. In autoassociation, an input **f** is associated with itself.

On the face of it, autoassociation sounds a bit silly, but in the next lecture, we will see the use of autoassociation.

### ■ Summary of linear association model

1. **Learning.** Let $\{\mathbf{f}_n, \mathbf{g}_n\}$ be a set of input/output activity pairs. Memories are stored by superimposing new weight changes on old ones. Information from many associations is present in *each* connection strength.

$$W_{n+1} = W_n + \mathbf{g}_n \; \mathbf{f}_n^T \tag{4}$$

(For simplicity, we've dropped the learning rate coefficient $\alpha$.)

2. **Recall.** Let **f** be an input possibly associated with output pattern **g**. For recall, the neuron acts as a linear summer:

$$\mathbf{g} = W\mathbf{f} \tag{5}$$

$$g_i = \sum_j w_{ij} \; f_j \tag{6}$$

3. **Condition for perfect recall**

If $\{\mathbf{f}_n\}$ are orthonormal, the system shows perfect recall:

$$
\begin{aligned}
W_n \; \mathbf{f}_m &= \left( \mathbf{g}_1 \; \mathbf{f}_1^T + \mathbf{g}_2 \; \mathbf{f}_2^T + \ldots + \mathbf{g}_n \; \mathbf{f}_n^T \right) \; \mathbf{f}_m \\
&= \mathbf{g}_1 \; \mathbf{f}_1^T \; \mathbf{f}_m + \mathbf{g}_2 \; \mathbf{f}_2^T \; \mathbf{f}_m + \ldots + \mathbf{g}_m \; \mathbf{f}_m^T \; \mathbf{f}_m + \ldots + \mathbf{g}_n \; \mathbf{f}_n^T \; \mathbf{f}_m \\
&= \mathbf{g}_m
\end{aligned}
\tag{7}
$$

since,

$$\mathbf{f}_n^T \; \mathbf{f}_m = \begin{cases} 1, & n = m \\ 0, & n \neq m \end{cases} \tag{8}$$

So an **nxn** matrix has a memory capacity of **n** for orthogonal inputs. For random vectors, it is about 10-20% of **n**.

But linear association is also useful as a mapping, in which one wants to generalize in a smooth (actually linear) way to novel inputs. For example, linear regression in 2D can be done with a 2x2 matrix. It can be "trained" with a few input/output pairs of points {x,y}. Once trained, if the data are well-modeled by a straight line, the network will do a nice job of "predicting" the output value (y) given a novel input value (x).

## Verify that linear memory model shows perfect recall for orthonormal

# inputs

**Define an 8x8 matrix G whose rows are 8 dimensional random output vectors**

---

**Print out row 3: G[[3]]**

---

**Define an 8x8 matrix Wmemory whose elements is the sum of the outer products of the rows of G with the rows of W (the orthonormal Walsh set defined earlier). This corresponds to Step 1 in the previous section**

---

**Show that G[[3]]=Wmemory.W[[3]]**

---

**Discussion question: What if the input vectors are not orthonormal?**

---

**Exercise: Pencil and paper**

---

Let f1 and f2 be two orthogonal, normalized input patterns:

$$f_1 = \frac{1}{\sqrt{3}}\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$$

$$f_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

and $g_1$, $g_2$ two output patterns:

$$g_1 = \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix}$$

$$g_2 = \begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix}$$

Form the outer product:

$$W = g_1 f_1^T$$

Test for recall by recalling $f_1$ as input to $W$. Stimulate $W$ with $f_1$: What happens? Add the outer product:

```
Solve[-2 lambda + lambda^2 == 3, lambda]
```

```
{{lambda → -1}, {lambda → 3}}
```

So our eigenvalues are -1 and 3. We can plug these values of lambda into our equations to solve for the eigenvectors:

```
Solve[{-x == x + 2 y, - y == 2 x + y}, {x,y}]
Solve[{3 x == x + 2 y, 3 y == 2 x + y}, {x,y}]
```

Solve::svars : Equations may not give solutions for all "solve" variables. ≫

```
{{x → -y}}
```

Solve::svars : Equations may not give solutions for all "solve" variables. ≫

```
{{x → y}}
```

### ■ Reduce

*Mathematica* is smart enough that we can use **Reduce[]** to do it all in one line:

```
Reduce[{lambda xv == W1.xv}, {x,y,lambda}]
```

$$(x == 0 \,\&\&\, y == 0) \,||\, \left( (y == -x \,||\, y == x) \,\&\&\, x \neq 0 \,\&\&\, \text{lambda} == \frac{x + 2\,y}{x} \right)$$

The eigenvectors are unique only up to a scale factor, so one can choose how to normalize them. For example, we could arbitrarily set x to 1, and then the eigenvectors are: {1,1}, and {1,-1}. Alternatively, we could normalize them to {1/Sqrt[2], 1/Sqrt[2]} and {1/Sqrt[2], -1/Sqrt[2]}.

```
Eigenvectors[W1]
```

```
{{1, 1}, {-1, 1}}
```

### ■ Side note: Solve[] vs. Reduce[]

**Solve[]** makes assumptions about constraints left unspecified, so the following returns the solution true for any lambda:

```
Solve[{lambda x == x + 2 y, lambda y == 2 x + y},
    {x,y,lambda}]
```

Solve::svars : Equations may not give solutions for all "solve" variables. ≫

$\{\{x \to -y, \text{lambda} \to -1\}, \{x \to y, \text{lambda} \to 3\}, \{x \to 0, y \to 0\}\}$

```
Solve[{lambda x == x + 2 y, lambda y == 2 x + y,
lambda !=0, x != 0, y != 0}, {x,y,lambda}]
```

Solve::svars : Equations may not give solutions for all "solve" variables. ≫

$\{\{x \to -y, \text{lambda} \to -1\}, \{x \to y, \text{lambda} \to 3\}\}$

**Reduce[]** gives all the possibilities without making specific assumptions about the parameters:

Either of the following forms will work too:

```
Reduce[lambda xv == W1.xv, {xv[[1]],xv[[2]],lambda}]
Reduce[{{lambda x, lambda y} == {x + 2 y, 2 x + y}},
    {x,y,lambda}]
```

$(x == 0 \;\&\&\; y == 0) \;||\; \left( (y == -x \;||\; y == x) \;\&\&\; x \neq 0 \;\&\&\; \text{lambda} == \frac{x + 2y}{x} \right)$

$(x == 0 \;\&\&\; y == 0) \;||\; \left( (y == -x \;||\; y == x) \;\&\&\; x \neq 0 \;\&\&\; \text{lambda} == \frac{x + 2y}{x} \right)$

#### ■ Determinant solution

$\mathbf{Ax} = \lambda\mathbf{x}$ can be written: $(\mathbf{A} - \mathbf{I}\lambda).\mathbf{x}$, where **I** is the identity matrix. The interesting values of **x** that satisfy this equation are the ones that aren't zero. For this to be true, $(\mathbf{A} - \mathbf{I}\lambda)$ must be singular (i.e. no inverse). And this is true if the determinant of $(\mathbf{A} - \mathbf{I}\lambda)$ is zero

## Answers

```
G = RandomReal[{0, 1}, {8, 8}];
```

```
W[[1]]
```

```
{0.353553, 0.353553, 0.353553,
  0.353553, 0.353553, 0.353553, 0.353553, 0.353553}
```

```
G[[1]]
```

```
{0.143815, 0.893796, 0.610998,
  0.943808, 0.934189, 0.352163, 0.306604, 0.176464}
```

```
Wmemory = Table[0, {8}, {8}];
For[i = 1, i ≤ 8, i++,
   Wmemory = Wmemory + Outer[Times, G[[i]], W[[i]]]];
```

```
Wmemory.W[[3]]
G[[3]]
```

```
{0.453989, 0.577835, 0.882825, 0.558917,
  0.936185, 0.553627, 0.328085, 0.0249444}
```

```
{0.453989, 0.577835, 0.882825, 0.558917,
  0.936185, 0.553627, 0.328085, 0.0249444}
```

# References

Boynton, G. M., Engel, S. A., Glover, G. H., & Heeger, D. J. (1996). Linear systems analysis of functional magnetic resonance imaging in human V1. J Neurosci, 16(13), 4207-4221.

Campbell, F. W., & Robson, J. R. (1968). Application of Fourier Analysis to the Visibility of Gratings. Journal of Physiology , 197, 551-566.

Carandini, M., Heeger, D. J., & Movshon, J. A. (1997). Linearity and normalization in simple cells of the macaque primary visual cortex. J Neurosci, 17(21), 8621-44.

Enroth-Cugell, C., & Robson, J. G. (1966). The contrast sensitivity of retinal ganglion cells of the cat, Journal of Physiology, London, 187, 517-552.

Gaskill, J. D. (1978). Linear Systems, Fourier Transforms, and Optics . New York: John Wiley & Sons.

Mackintosh, N. J. (1994). Animal learning and cognition. San Diego: Academic Press.

Nykamp, D. Q., & Ringach, D. L. (2002). Full identification of a linear-nonlinear system via cross-correlation analysis. J Vis, 2(1), 1-11.

Silverman, M.S., Grosof, D.H., DeValois, R.L., & Elfar, S.D. (1989). Spatial-frequency organization in primate striate cortex. *Proceedings of the National Academy of Science USA, 86*, 711-715.

Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning : an introduction. Cambridge, Mass.: MIT Press.

http://onlinebooks.library.upenn.edu/webbin/book/lookupid?key=olbp29573