STATISTICAL INFERENCE AND LEARNING

Learning

- The NEF gives analytic weights, so why learn?
  - fine-tuning (NEF weights are based on rate model approximations; NEF weights are "first guesses" at desired weights; see later)
  - we can compare learned with analytically found weights giving us insight into learning
  - highlight both new challenges, and the inherent strengths of this approach

Learning a comm channel

- Simplest circuit to consider, just reproduction
- Maximizing the variance of the responses in the receiving pop. allows it to carry the most info it can about incoming signals; e.g. all of it
- I.e. maximize:

$$E = \frac{1}{2} \sum_j \left\langle \left(b_j(x) - \bar{b}_j(x) \right)^2 \right\rangle_x$$

- We know : $b_j(x) = G_j\left[\sum_i \omega_{ji} a_i(x) + J_j^{bias}\right]$
- Take the derivative of E with respect to weights

$$\frac{dE}{\delta\omega_{ji}} = \frac{dG_j\left[\xi\right]}{d\xi}\frac{d\xi}{d\omega_{ij}}\left(b_j(x) - \bar{b}_j(x)\right)$$

$$\frac{dE}{\delta\omega_{ji}} = \frac{dG_j\left[\xi\right]}{d\xi} a_i(x)\left(b_j(x) - \bar{b}_j(x)\right)$$

- Where $\xi = \sum_i \omega_{ji} a_i(x) + J_j^{bias}$

Learning a comm channel

- The G_j term means that the rate of change of the activity of the neuron for input x matters
- We can replace it with the simpler term $b_j(x) > 0$ as an approximation. This evaluates to 1 when the neuron is active and 0 when it is not
- Finally, the mean can be found as

$$\bar{b}_j(x; t + dt) = (1 - \epsilon)\bar{b}_j(x; t) + \epsilon b_j(x; t)$$

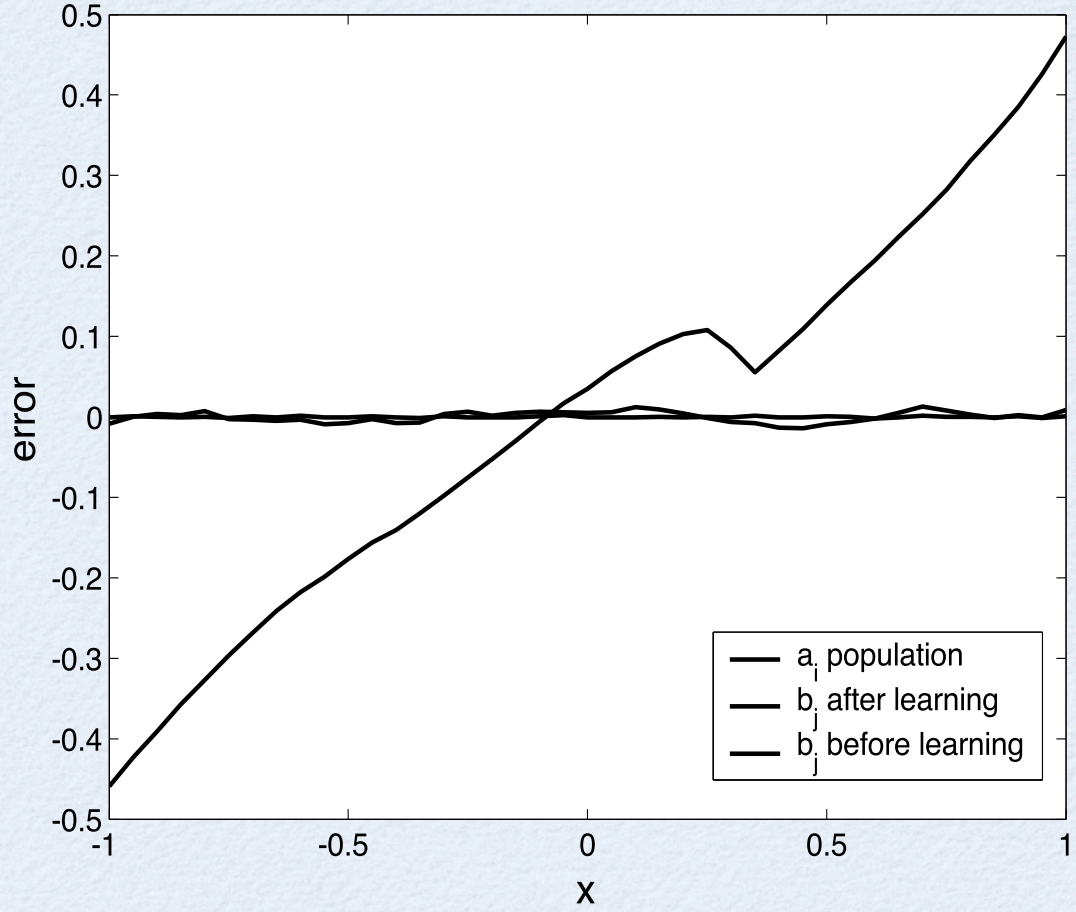
Learning a comm channel

- Now use the standard 'delta rule' approach:

$$\begin{aligned}\Delta\omega_{ij} &= -\kappa\frac{dE}{\delta\omega_{ji}} \\ &= -\kappa\left(b_j(x) > 0\right) a_i(x) \left(b_j(x) - \bar{b}_j(x)\right)\end{aligned}$$

- Where κ is the learning rate
- Note this is a *local* Hebbian rule

Error before/after learning

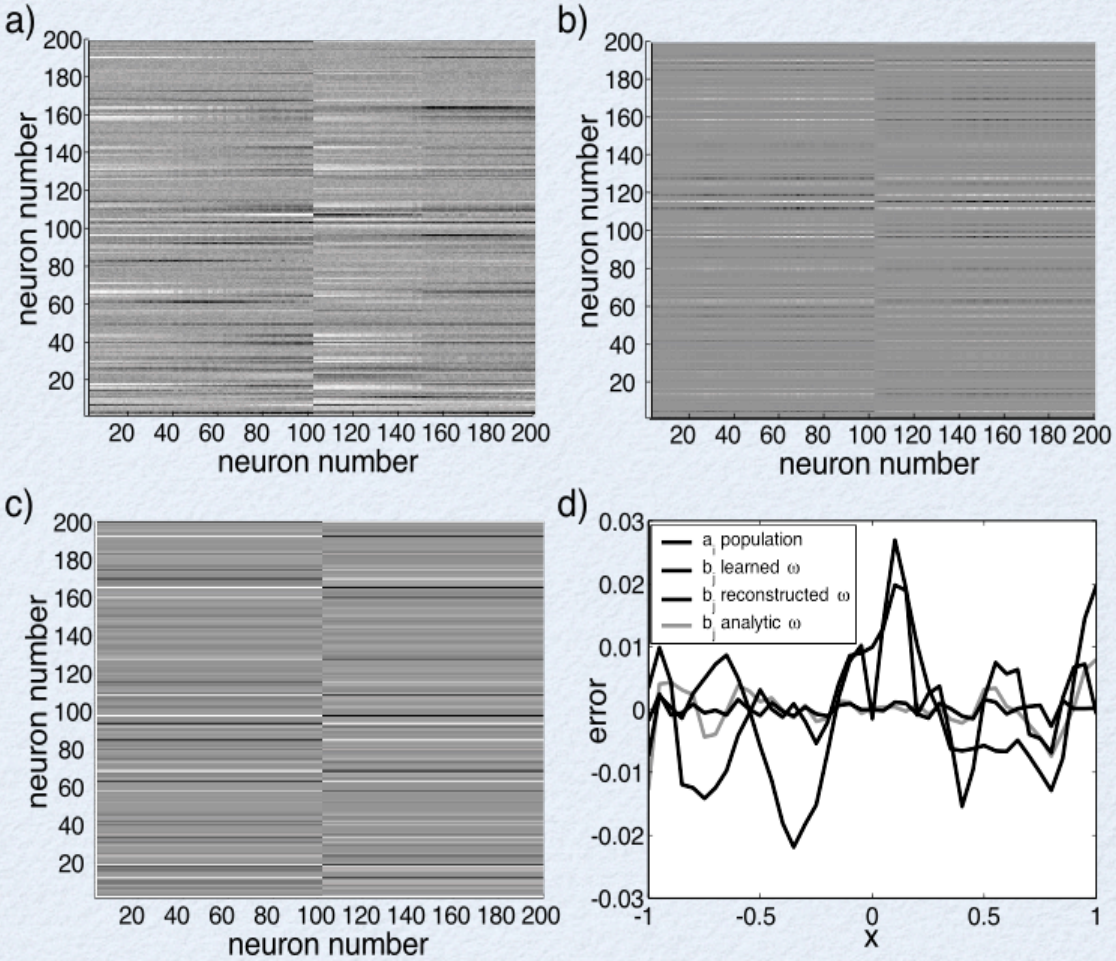


Learning a comm channel

- Natural to ask if we can decompose this learned matrix into its components
- Recall the general form of weights is $\omega_{ji} = \alpha_j \tilde{\phi}_j \phi_i$
- The encoders are ±1 and we know the gains, so we can use SVD to 'recover' the decoders
- Given this approximation, we then reconstruct the original weight matrix to compare

Weight reconstruction

- a) learned b) analytic c) reconstructed



Weight reconstruction

- Similar performance emphasizes that there are many solutions to such problems given the high-dimensionality of weight space
- Here, the pattern of connectivity is similar: oppositely tuned neurons tend to have negative weights; the strength of weights depends on the similarity of the neurons' tuning curves.

Weight reconstruction

- Note that even the reconstructed matrix is good at preserving the information in the signal
- Decomposition may be useful for analyzing the results of employing a learning rule
- Derived decoders can be used to determine what function those weights compute (given the encoding assumptions) by using them to decode the incoming signal

Associative memory

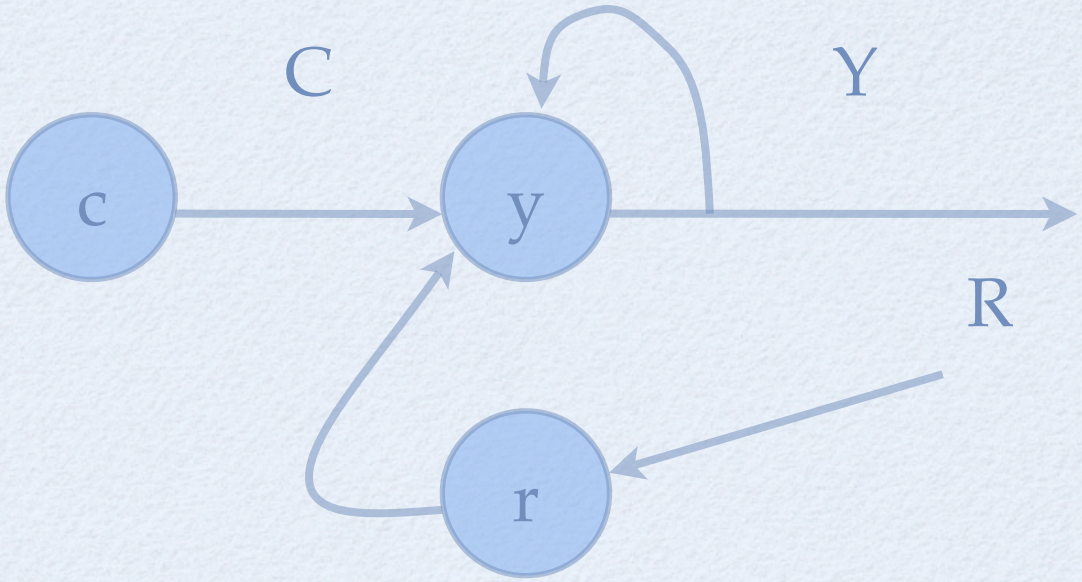
- Demonstrates a means of relating 'high-level' learning to neural connection weights
- Associate C with Y given the 'answer' R (next slide)

$$E = \frac{1}{2}[R - Y]^2$$

- Let the representations be as usual, e.g.

$$\begin{aligned} Y &= \sum_j y_j \phi_j \\ y_j &= G_j \left[\alpha_j \left\langle \tilde{\phi}_j \left[\hat{C} + \hat{R} \right] + J_j \right\rangle \right] \end{aligned}$$

Learning transformations



Associative memory

- Substitute into E and find learning for C

$$E = \frac{1}{2}\left[\sum_k \phi_k r_k - \sum_j \phi_j G_j \left[\sum_l \omega_{jl} c_l + \sum_k \omega_{jk} r_k + J_j\right]\right]^2$$

$$\frac{dE}{d\omega_{jl}} = -\left(\sum_k \phi_k r_k - \sum_j \phi_j y_j\right)\left[\frac{d}{d\omega_{jl}} \sum_i \phi_i G_i \left[\sum_l \omega_{il} c_l + \sum_k \omega_{ik} r_k + J_j\right]\right]$$

- As before, use (y_j>0) for the derivative of G_j:

$$\frac{dE}{d\omega_{jl}} = -\left(\sum_k \phi_k r_k - \sum_i \phi_i y_i\right)(y_j > 0)\phi_j c_l$$

Associative memory

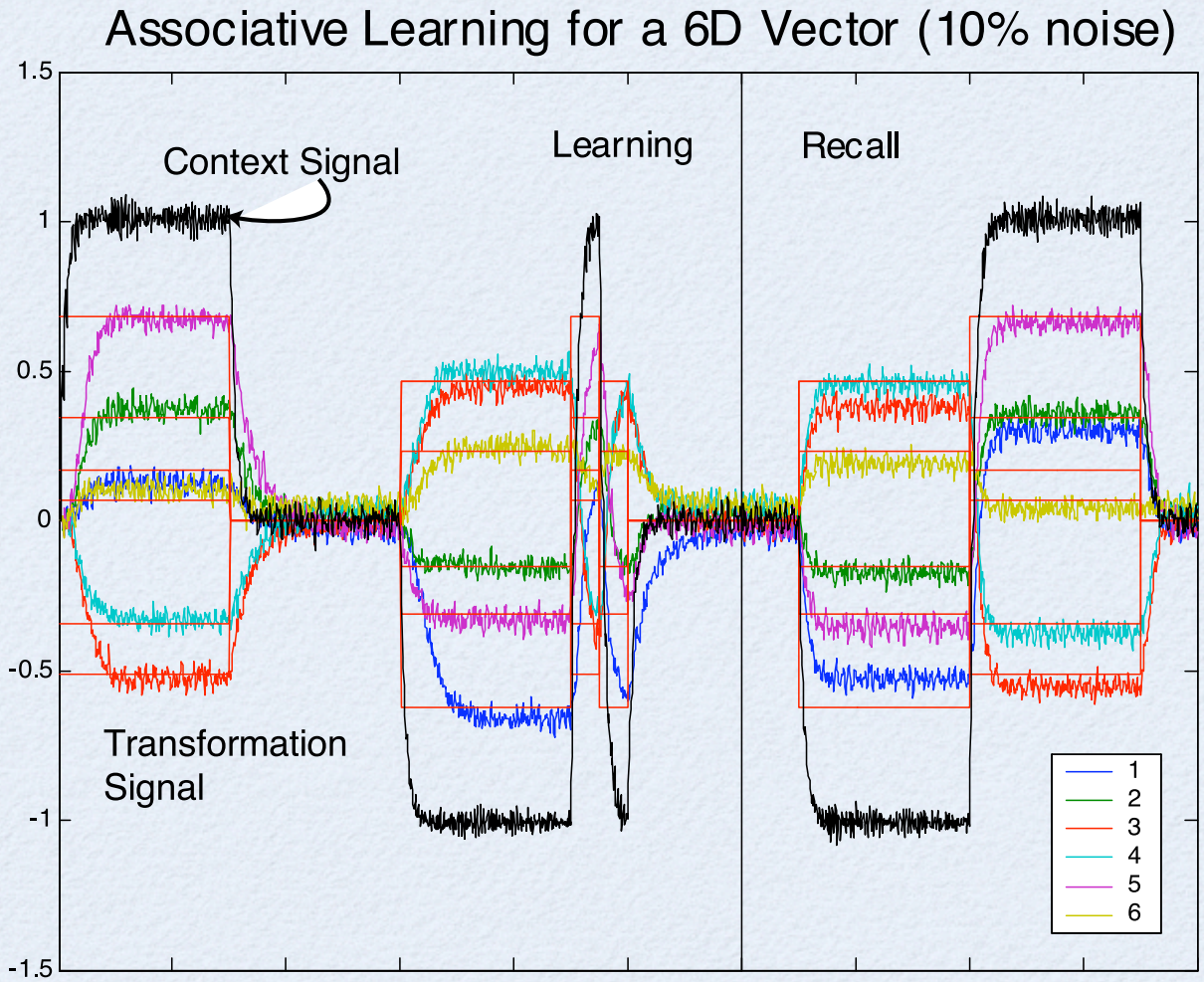
- Standard delta rule: $\Delta\omega_{jl} = -\kappa \frac{dE}{d\omega_{jl}}$
- Gives a biologically implausible rule, so multiply by the encoders and gains:

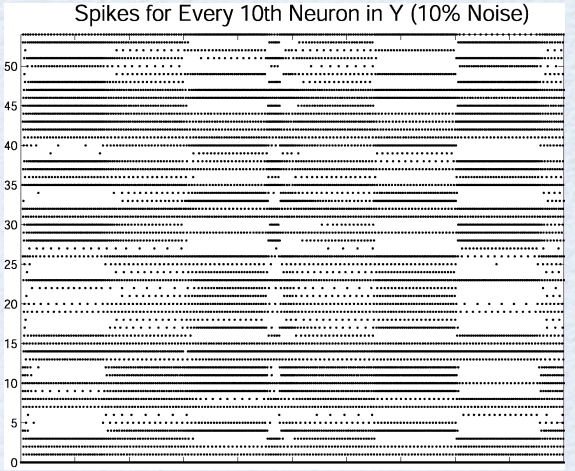
$$\Delta\omega_{jl}\alpha_j\tilde{\phi}_j = \kappa\left(\sum_k \omega_{jk}r_k - \sum_i \omega_{ij}y_i\right)(y_j > 0)\phi_j c_l$$

- Do it again (notice norm = 1; enc*dec ≈ K)

$$\begin{aligned}\Delta\omega_{jl}\alpha_j\left\|\tilde{\phi}_j\right\| &= \kappa\left(\sum_k \omega_{jk}r_k - \sum_i \omega_{ij}y_i\right)(y_j > 0)c_l \\ \Delta\omega_{jl} &= \frac{\kappa}{\alpha_j}\left(\sum_k \omega_{jk}r_k - \sum_i \omega_{ij}y_i\right)(y_j > 0)c_l\end{aligned}$$

Learning results





Thoughts

- Learning literature is enormous, we've barely scratched the surface, nevertheless:
  - The NEF can help us gain new insights regarding standard learning rules, because of the analysis (e.g., reconstruction)
  - The representational hierarchy lets us generalize rules and analyses to systems trafficking in more complex representations

Thoughts

- Because an attractor is a 'self-communication channel' & the hierarchy, we can find an attractor learning rule (see text)
- Since we have a general means of incorporating input signals and transformations, accounting for learning in systems with explicit error signals, like the neural integrator (retinal slip), is straightforward

Thoughts

- Importantly, the NEF gives us a means of constructing complex, functional systems, *independent* of learning
- This is essential for modeling complex systems because, at the moment, it is not clear how to learn many of the complex behaviors exhibited by neurobiological systems
- Suggests NEF and learning approaches are powerfully *complimentary*

Challenges

- Can we generate learning rules that give rise to the systems we construct analytically?
- Kinda, but for complex transformations and complex control structures, it is not clear what the answer will be.
- Notice that maximizing the variance between two populations, only constrains one degree of freedom in the network's representations.
- For arbitrary transformations, we will need to constrain many degrees of freedom

Challenges

- Suppose that we have a population from which we can encode the lower-order polynomials, x^n
- We can find decoders
- Now suppose we want to compute some function, $f(x)$, that is a sum of these polynomials
- The weights will be $\omega_{ji} = \tilde{\phi}_j \sum_m A_m \phi_i^{(x^n)}$
- So a learning rule will have to constraint M degrees of freedom independently

Challenges

- Worse, there needs to be a mechanism such that the desired values for the coefficients are applied consistently across the population.
- But, to remain biologically plausible we have to impose these constraints using *local* rules
- Such rules will need to be highly sophisticated (they need to be controlled by a set of *M*-dimensional signals in a neurally plausible way)

Challenges

- Learning also highlights the ambiguity of connection weights: there is no isolated `right' decomposition of a weight matrix
- Any ambiguity can only be overcome in the context of a larger understanding of the system
- For the comm. chan., finding decoders depends on our assumptions about the encoders.
- This is like the representational ambiguity we discussed earlier

Challenges

- Finally, how can we use the framework itself to generate novel learning rules?
- It would be very useful to be able to have a means of determining plausible learning rules that would give rise to the kinds of high-level structure that the NEF can build into models
- These challenges can be summarized by noting that we have not provided a systematic way to relate learning to the NEF

One solution

- Exploit error signals
- Error signals are common:
  - dopamine/serotonin
  - ERN (error related negativity)
  - ACC (anterior cingulate cortex)
- Need biological rules that exploit error information

Fine-tuning

- Stability can only be achieved by tuning weights to within 1% of the theoretical ideal
  - But there's lots of noise!
- Try some other approaches (e.g., Koulakov et al, 2002; Goldman 2009)
  - But, no strong evidence for these mechanisms

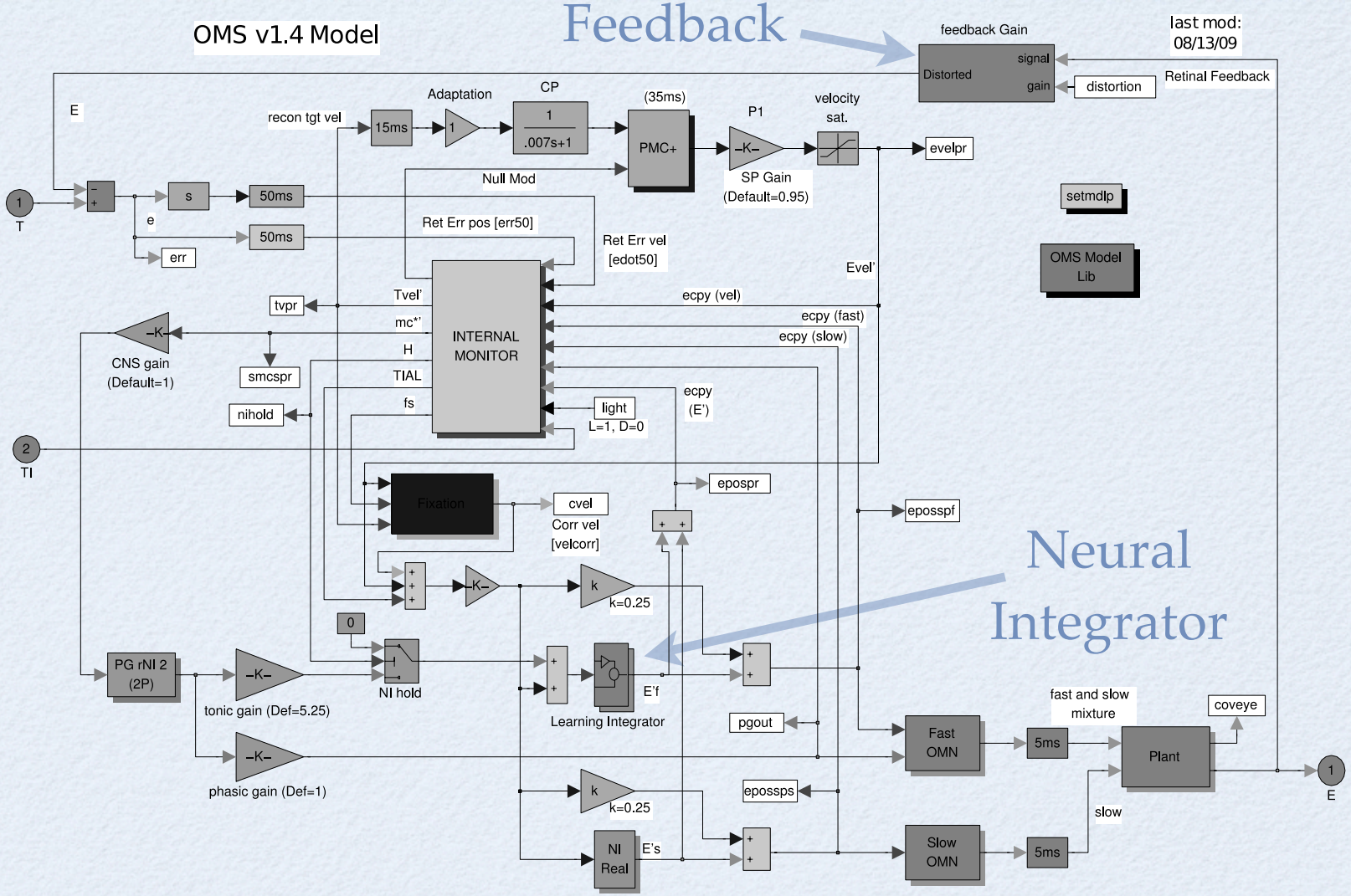
Fine-tuning

- How about *in vivo* fine-tuning?
  - Past rules (e.g. Arnold and Robinson, 1992, 1997; Turaga et al., 2006; Renart et al, 2003)
    - aren't local
    - assume retinal slip is available (it's not (Collewijn, 1977))
    - can't learn in the dark (Harris and Cynader, 1981)
    - can't detune the integrator (Major et al, 2004)

Let's try again

- Build a model with the appropriate input
  - Corrective saccades drive adaptation (Park & Shimojo, 2007)
  - Take the input from Dell'Osso and Wang's OMS model
- Derive a learning rule to exploit that input

OMS Model



"k" in gain block and in "IM (SP)" are set in the "Initialization" tab of model's mask.

Dell'Osso and Wang (2008)

Learning

- Minimize

$$E = \frac{1}{2}(x - \hat{x})^2$$

$$= \frac{1}{2}(x - \sum_i a_i d_i)^2$$

- Gives

$$\frac{dE}{dd_i} = v_c a_i$$

- Delta rule

$$\Delta d_i = \kappa v_c a_i$$

- Multiply both sides by encoder and gain

Current from velocity command

$$\Delta d_i e_j \alpha_j = \kappa \alpha_j e_j v_c a_i = \Delta \omega_{ij}$$

Presynaptic activity

Experiments

- Benchmarks
  1. Optimal
  2. Noised (30% Gaussian on weights)
  3. Learned after noised
  4. Learned after continuous noise (10%)
  5. Learned after continuous (5%) and noised (30%)

Results

- Goldfish range between 29 and 95s (Mensh et al., 2004)

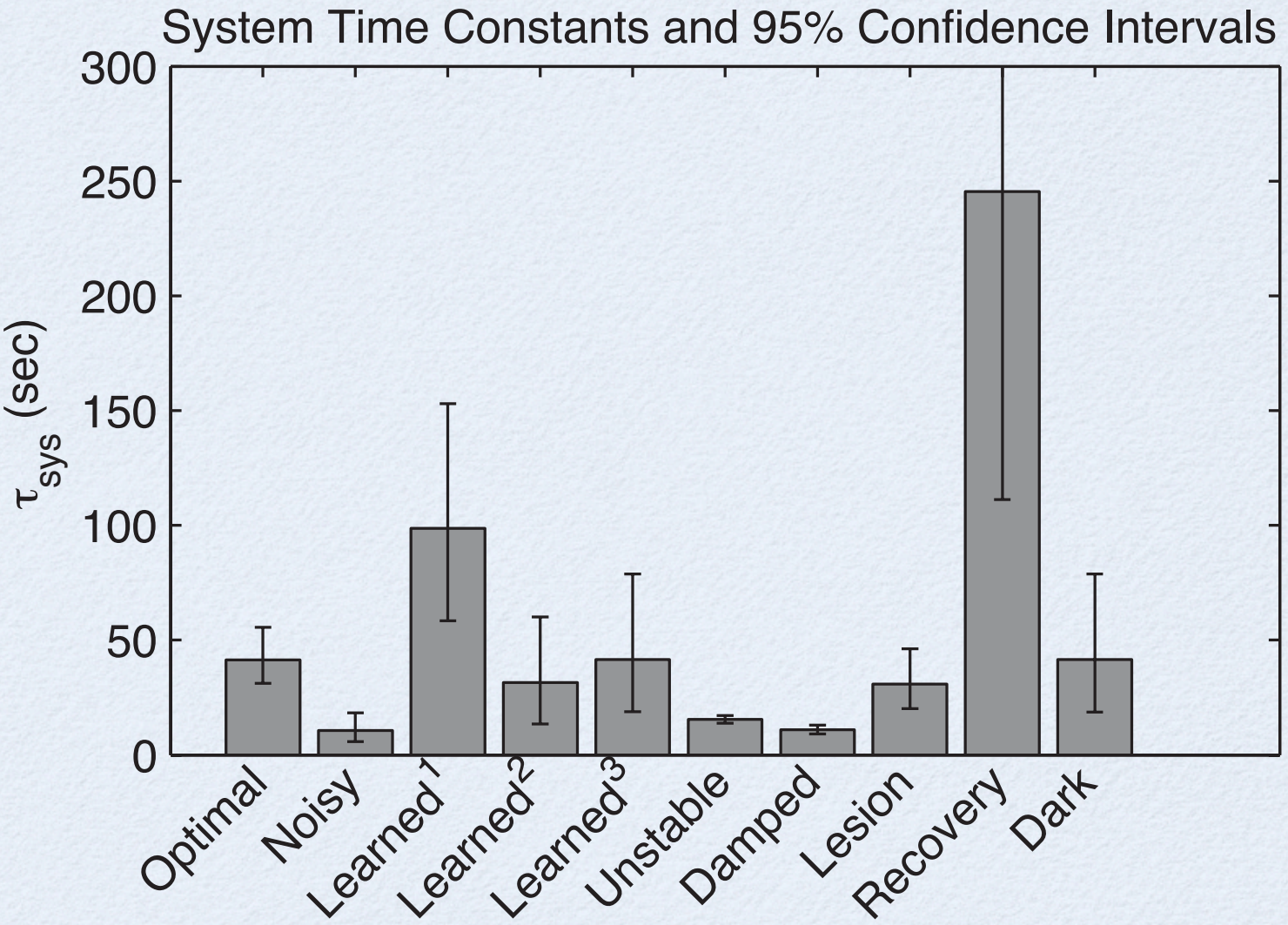
	Case	Mean	CI
1	Optimal	(+) 41.4	31.2 - 55.6
2	Noisy	(+) 10.6	5.85 - 18.2
3	Learned[1]	(+) 98.7	58.5 - 153
4	Learned[2]	(-) 31.6	13.5 - 60.1
5	Learned[3]	(+) 41.4	18.9 - 78.8

[1] After an initial disturbance (30%) to connection weights.
[2] With continuous noise (10%) added to connection weights.
[3] After an initial disturbance (30%) and continuous noise (5%).

Experiments

- Oculomotor integrator manipulations
  1. Unstable (Major et al. 2004)
  2. Damped (Major et al. 2004)
  3. Lesioned (Arnold & Robinson 1991)
  4. Recovery (Arnold & Robinson 1991)
  5. Dark (Harris & Cynander 1981)

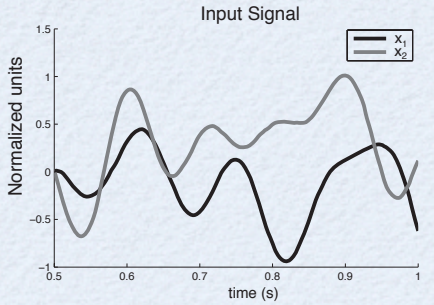
Results summary

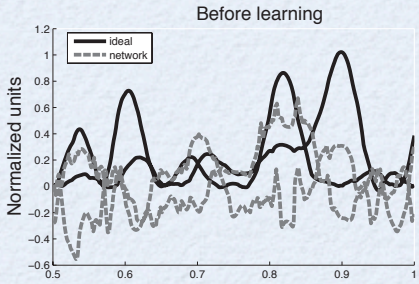


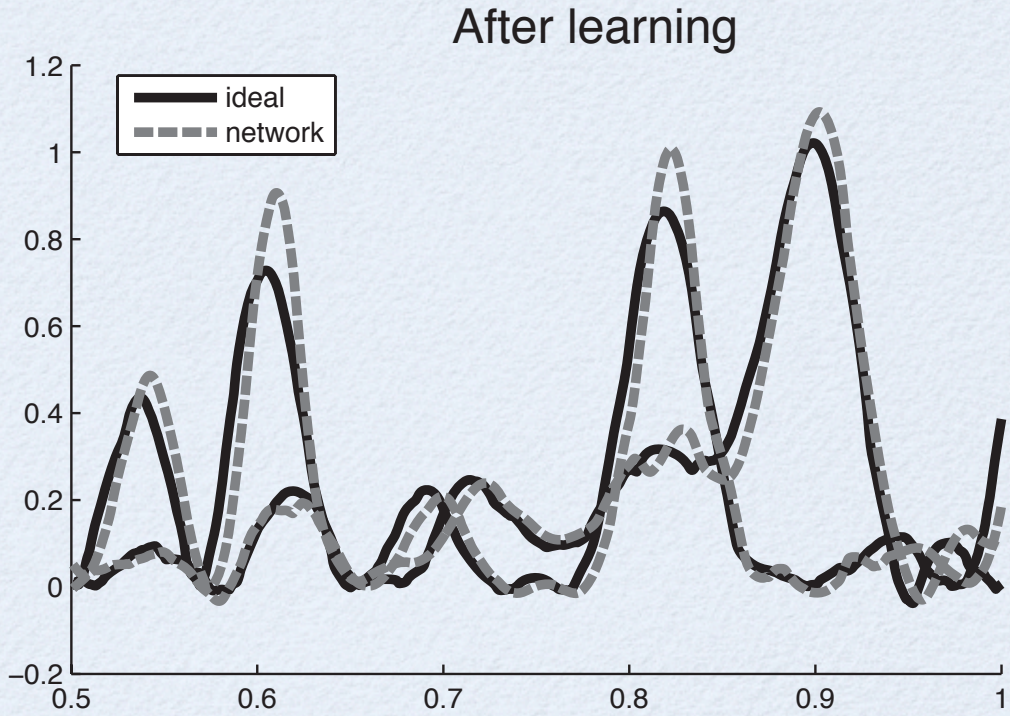
Generalizing

- Introduce preferred vectors and generalized 'error'

$$\Delta\omega_{ij} = \kappa\alpha_j \mathbf{e}_j \mathbf{E} a_i$$

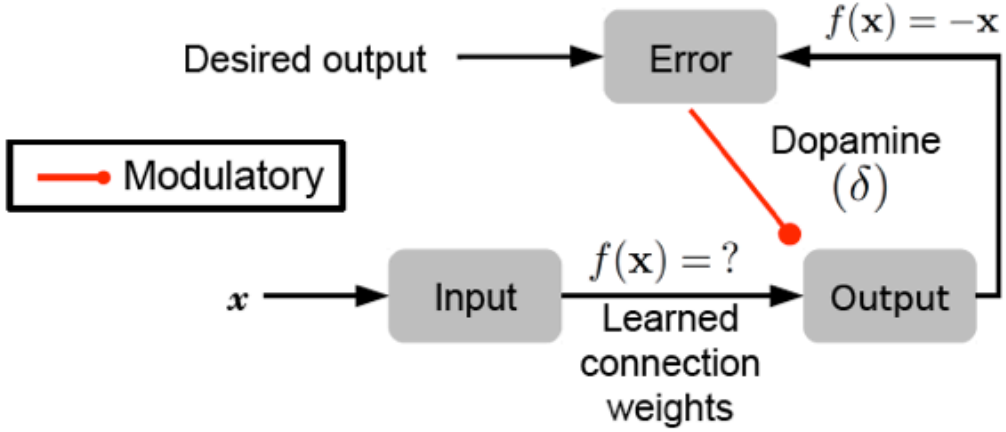


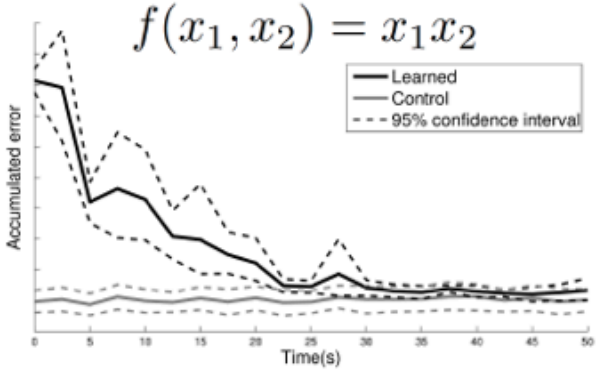


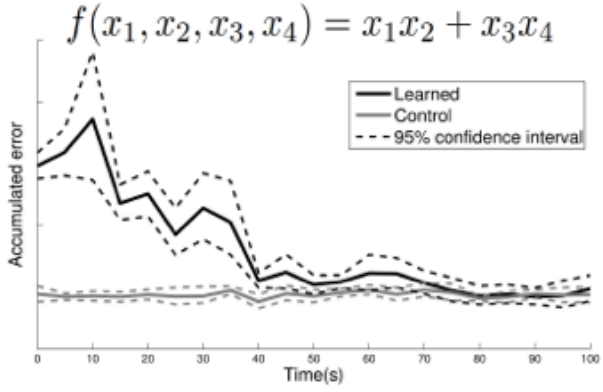


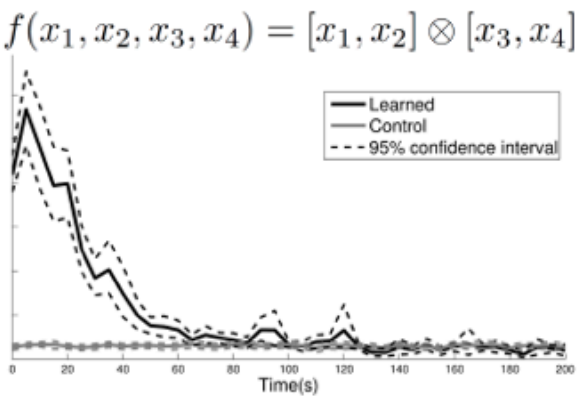
Learning 'anything'

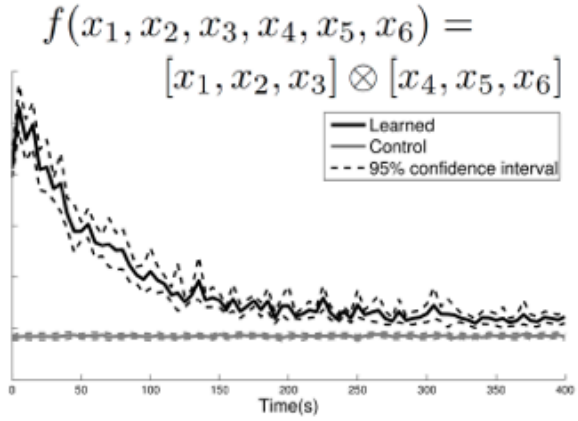
$$\Delta\omega_{ij} = \kappa\alpha_j \mathbf{e}_j \mathbf{E} a_i$$











Learning 'anything'

- Obviously, you need to have the error signal.
  - How are these generated?
  - How are they transmitted (dopamine? other neuromodulators? directly?)
  - How are they encoded? (must multiply the neuron's encoder by the E)

Generalizing

- Past work has shown how to model any attractor network (ring, plane, cyclic, and chaotic attractors; Eliasmith, 2005)
- Are the stabilizing errors available?
  - Head direction: yes
  - Working mem? others?

Thoughts...

- Learning is often considered supervised or unsupervised
- Unsupervised is often called self-organization
  - Captures statistics of the input stream
- This seems an intermediate case:
  - 'Supervision' is internally generated
  - Self-directed organization(?) (a kind of RL?)

Statistical inference

- What is statistical inference?
  - Rather than consider the truth and falsity of sentences (logic), probability theory considers the likelihood of events or states of the world.
  - Statistical inference uses such representations to determine an appropriate behaviour

Statistical inference

- Why statistical inference?
  - Our focus is neurobiological systems in general (most with no language) all of which must reason about states of a noisy, uncertain physical world.
  - Probability theory (PT) is an appropriate tool for characterizing such systems
- PT describes how to use multiple uncertain sources of information to increase certainty; and, how to update a current `take' on the world

Statistical inference

- PT is the best available quantitative tool for describing the relevant kinds of reasoning
- There are already sophisticated probabilistic formalisms for modeling 'reasoning' as SI
- E.g. pattern theory (Grenander):
  - Bayesian inference defines transformations that work with the complex representations defined in pattern theory

Statistical inference

- Why for neurobiology?
- Consider the joint distribution of two variables, $p(\mathbf{x},\mathbf{y})$. We can write this joint in terms of the individual distributions $p(\mathbf{x})$ and $p(\mathbf{y})$:

$$\begin{aligned} \rho(\mathbf{x},\mathbf{y}) &= \rho(\mathbf{y}|\mathbf{x})\rho(\mathbf{x}) \\ &= \rho(\mathbf{x}|\mathbf{y})\rho(\mathbf{y}) \end{aligned}$$

- (Bayes' rule consists of equating the two right hand sides ... but we don't need that yet)

$$\rho(\mathbf{x}|\mathbf{y}) = \frac{\rho(\mathbf{y}|\mathbf{x})\rho(\mathbf{x})}{\rho(\mathbf{y})}$$

Statistical inference

- To determine the probability density function (PDF) for either parameter alone, we `marginalize' (i.e., integrate) the joint:

$$\rho(\mathbf{y}) = \int \rho(\mathbf{x}, \mathbf{y})\, d\mathbf{x}$$

$$\rho(\mathbf{x}) = \int \rho(\mathbf{x}, \mathbf{y})\, d\mathbf{y}$$

- In the NEF, these are represented in a neural population

Statistical inference

- Consider the example of vision
  - Take $p(\mathbf{x})$ as how likely each image is
  - This estimate is made in the context of some data provided by the environment (the image falling on the retina, corrupted by noise).
  - Thus we need the PDF $p(\mathbf{x} \mid \mathbf{d})$, the true image given the measured image

Statistical inference

- We want properties of the image that are not directly captured by the retinal image (e.g. objecthood). Let $\mathbf{y}$ be such properties.
- The PDF for $\mathbf{y}$ given the image is thus:

$$\rho(\mathbf{y}|\mathbf{d}) = \int \rho(\mathbf{y}|\mathbf{x})\rho(\mathbf{x}|\mathbf{d})\, d\mathbf{x}$$

- Tells us how to determine the probability of all possible objects given image (over-simplified)
- This is simply a linear transform into the new space $p(\mathbf{y})$ & includes $\mathbf{y} = f(\mathbf{x})$ as a subset

Statistical inference

- There is no assumption of Gaussian statistics (multi-modal distributions in either space, or the conditional (so unimodal inputs support multiple hypotheses))
- Being able to implement these transformations results in computationally powerful systems (e.g. ANN models)
- Real neurobiological networks are also ideally suited to implementing these transformations

Statistical inference

- To see why, notice that $p(\mathbf{x} \mid \mathbf{d})$ is a function parameterized by the variables $\mathbf{d}$.
- Need neurons to encode these (volunteer):

$$a_i(\mathbf{d}) = G_i \left[\alpha_i \left\langle \tilde{\phi}_i(\mathbf{x}) \rho(\mathbf{x}|\mathbf{d}) \right\rangle_{\mathbf{x}} + J_i^{bias} \right]$$

$$b_j(\mathbf{d}) = G_j \left[\alpha_j \left\langle \tilde{\phi}_j(\mathbf{y}) \rho(\mathbf{y}|\mathbf{d}) \right\rangle_{\mathbf{y}} + J_j^{bias} \right]$$

- With decoders:

$$\hat{\rho}(\mathbf{x}|\mathbf{d}) = \sum_i \phi_i(\mathbf{x}) a_i(\mathbf{d})$$

$$\hat{\rho}(\mathbf{y}|\mathbf{d}) = \sum_j \phi_j(\mathbf{y}) b_j(\mathbf{d})$$

Statistical inference

- Given the need transformation, we have

$$
\begin{aligned}
b_j(\mathbf{d}) &= G_j\left[\alpha_j\left\langle\tilde{\phi}_j(\mathbf{y})\rho(\mathbf{y}|\mathbf{x})\rho(\mathbf{x}|\mathbf{d})\right\rangle_{\mathbf{x},\mathbf{y}} + J_j^{bias}\right] \\
&= G_j\left[\alpha_j\left\langle\tilde{\phi}_j(\mathbf{y})\rho(\mathbf{y}|\mathbf{x})\sum_i\phi_i(\mathbf{x})a_i(\mathbf{d})\right\rangle_{\mathbf{x},\mathbf{y}} + J_j^{bias}\right] \\
&= G_j\left[\sum_i\omega_{ji}a_i(\mathbf{d}) + J_j^{bias}\right]
\end{aligned}
$$

- where $\omega_{ji} = \alpha_j\left\langle\tilde{\phi}_j(\mathbf{y})\rho(\mathbf{y}|\mathbf{x})\phi_i(\mathbf{x})\right\rangle_{\mathbf{x},\mathbf{y}}$

Statistical inference

- So feedforward inference in the NEF gives connection weights between neurons that are the projection of the encoding functions of the output neurons, on the conditional, weighted by the decoders of each input neuron
- Some general consequences of this example:
  - SI in high-D spaces requires estimating high-D integrals. The high degree of convergence in neural networks, makes them ideally suited for performing these kinds of transformations

Statistical inference

- The form of these equations is identical to those for a simple feed-forward ANN, and there is no clear advantage gained by introducing the nonlinearity G_i (implementation? dimensionality?)
- The conditional is a `look-up table' that specifies the value of **y** for each **x**. Clearly a limited approach (all possibilities pre-computed; curse of dimensionality (CoD))

Statistical inference

- To address the CoD we divide high-D spaces into independent subspaces: i.e., $p(\mathbf{x},\mathbf{z}) = p(\mathbf{x})p(\mathbf{z})$

$$\rho(\mathbf{y}|\mathbf{d_x},\mathbf{d_z}) = \int \rho(\mathbf{y}|\mathbf{x},\mathbf{z})\rho(\mathbf{x}|\mathbf{d_x})\rho(\mathbf{z}|\mathbf{d_z})\,d\mathbf{x}\,d\mathbf{z}$$

- Implementation would be

$$b_j(\mathbf{y}) = G_j\left[\sum_{ik}\omega_{jik}a_i(\mathbf{d_x})c_k(\mathbf{d_z}) + J_j^{bias}\right]$$

- where

$$\omega_{jik} = \left\langle \tilde{\phi}_j(\mathbf{y})\rho(\mathbf{y}|\mathbf{x},\mathbf{z})\phi_i(\mathbf{x})\phi_k(\mathbf{z})\right\rangle_{\mathbf{x},\mathbf{y},\mathbf{z}}$$

Statistical inference

- Realize that the number of dimensions that must be stored when **x** and a are independent is $D_1 = D_{\mathbf{x}} + D_{\mathbf{z}}$. When they aren't $D_{\mathbf{x}} \times D_{\mathbf{z}} >> D_1$
- Rich set of computations:
  - Take **z** to change the connection weights between **y** and **x** (i.e. provide a context), dynamically
  - Take **x** and **z** could represent evidence from two-different modes (vision and audition; feedforward)
  - Or, **z** could represent variables in a higher order (`top-down') model and **x** could provide the feed-forward (`bottom-up') evidence.

Statistical inference

- The cost of these computations is multiplicative interactions between the activities of the **z** and **x**
- The ubiquitous need for performing this kind of context dependent statistical inference is one more reason that we might expect to find such nonlinearities in dendrites.
- Others have made similar suggestions

Ambiguous input

- Inference depends critically on the system's assumptions (`top-down' information) about the structure of the environment
- Functionally and anatomically evident
  - Functional: concave faces look convex from a meter or more away, unless upside-down
  - Anatomical: massive reciprocal projections

Ambiguous input

- Toy problem: object location
- Animal's `guess' as to where the object is: $p(y)$
- Based on info from sensory system: $p(x \mid \mathbf{d})$ (sensory system's assignment of the probability that the object is at x, given noisy measurements, $\mathbf{d}$; suppose it's bimodal)
- Suppose there is an 'expectation' or 'model' of location: $p(z \mid \mathbf{m})$
- $\mathbf{m}$ summarizes past experience with object positions

Ambiguous input

- To determine a best guess for some particular $p(x \mid \mathbf{d})$ and $p(z \mid \mathbf{m})$, construct the joint $p(x,y,z \mid \mathbf{d},\mathbf{m})$, which captures all of the relations between $p(x \mid \mathbf{d})$, $p(y \mid \mathbf{d},\mathbf{m})$, and $p(z \mid \mathbf{m})$
- Assume that the input $\mathbf{d}$ and model $\mathbf{m}$ are independent, then marginalize to find $p(y \mid \mathbf{d},\mathbf{m})$

$$\rho(y|\mathbf{d},\mathbf{m}) = \iint \rho(y|x,z)\rho(x|\mathbf{d})\rho(z|\mathbf{m})\,dx\,dz$$

Ambiguous input

- We need to know $p(y \mid x,z)$, let's choose

$$\rho(y|x,z) = \frac{1}{\sqrt{2\pi(\alpha(x-z)^2+\beta)}} e^{-(y-\frac{1}{2}(x+z))^2/\left(\alpha(x-z)^2+\beta\right)}$$

- This emphasizes where data and model agree, and de-emphasizes where they don't

- It takes the average value of x and z and constructs a distribution around that mean whose variance depends on how similar x and z are (β gives non-zero variance, α controls how much the emphasis relies on this difference)

Recall the implementation

- Suppose events are independent as before, i.e., $p(\mathbf{x},\mathbf{z}) = p(\mathbf{x})p(\mathbf{z})$

$$\rho(\mathbf{y}|\mathbf{d_x},\mathbf{d_z}) = \int \rho(\mathbf{y}|\mathbf{x},\mathbf{z})\rho(\mathbf{x}|\mathbf{d_x})\rho(\mathbf{z}|\mathbf{d_z})\,d\mathbf{x}\,d\mathbf{z}$$

- Implementation would again be

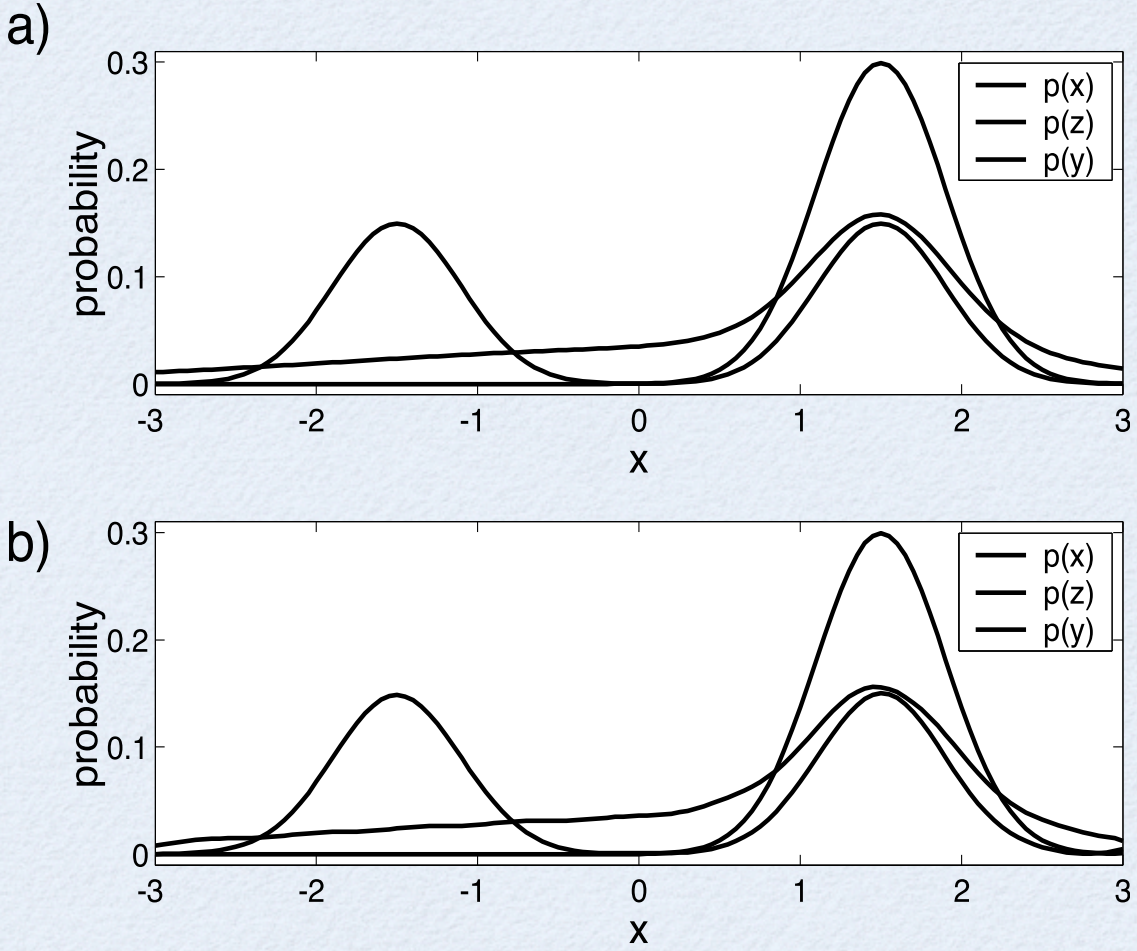
$$b_j(\mathbf{y}) = G_j\left[\sum_{ik}\omega_{jik}a_i(\mathbf{d_x})c_k(\mathbf{d_z}) + J_j^{bias}\right]$$

- where

$$\omega_{jik} = \left\langle \tilde{\phi}_j(\mathbf{y})\rho(\mathbf{y}|\mathbf{x},\mathbf{z})\phi_i(\mathbf{x})\phi_k(\mathbf{z})\right\rangle_{\mathbf{x},\mathbf{y},\mathbf{z}}$$

Top-down inference

- a) ideal, b) neural implementation



Parameter estimation

- Toy problem of finding **m**, under constraint
- Suppose the system gets a stream of values, x_n, `measurements' of a statistical process.
- Assume that the values are generated by a Gaussian process, mean $\underline{x}$ and variance σ^2.
- The system must estimate the mean and variance from the measurements

Parameter estimation

- Define the conditional probability of obtaining the value of x given $\underline{x}$ and σ as a Gaussian

$$\rho(x|\bar{x},\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\bar{x})^2/2\sigma^2}$$

- Assume that $\underline{x}$ and σ are drawn from a broad distribution, $p_0(\underline{x}, \sigma)$, the marginal for x is

$$\rho(x) = \iint \rho(x|\bar{x},\sigma)\rho_0(\bar{x},\sigma)\, d\bar{x}\, d\sigma$$

- So, $\underline{x}$ and σ have been drawn from $p_0(\underline{x}, \sigma)$ to generate a signal we want to characterize

Parameter estimation

- Goal is have a network that represents a PDF over $\underline{x}$ and σ that starts with the prior, and is updated appropriately as the signal arrives
- Representation of input

$$b_j(x_1, \ldots, x_n) = G_j \left[\alpha_j \left\langle \tilde{\phi}_j(\bar{x}, \sigma) \rho(\bar{x}, \sigma | x_1, \ldots, x_n) \right\rangle_{\bar{x}, \sigma} \right]$$

$$\hat{\rho}(\bar{x}, \sigma | x_1, \ldots x_n) = \sum_j b_j(x_1, \ldots, x_n) \phi_j(\bar{x}, \sigma)$$

Parameter estimation

- The measurements themselves are uncertain, so we will take them to be PDFs as well

$$a_i(x_n) = G_i \left[\alpha_i \left\langle \tilde{\phi}_i(x) \rho(x|x_n) \right\rangle_x + J_i^{bias} \right]$$

$$\hat{\rho}(x|x_n) = \sum_i \phi_i(x) a_i(x_n)$$

- Each measurement x_n is thus taken as the mean of some Gaussian (width=noise).
- This 1D function (of x) is represented as usual

Parameter estimation

- As before, we need an update rule to relate these, for each new x_{n+1} measurement

$$\begin{aligned}
\rho(\bar{x}, \sigma | x_1, \ldots x_{n+1}) &= \int \rho(\bar{x}, \sigma | x) \rho(x | x_{n+1}) \, dx \\
&= \int \frac{\rho(x | \bar{x}, \sigma)}{\rho(x)} \rho(x | x_{n+1}) \, dx \, \rho(\bar{x}, \sigma | x_n) \\
&\approx \int \rho(x | \bar{x}, \sigma) \rho(x | x_{n+1}) \, dx \, \rho(\bar{x}, \sigma | x_n)
\end{aligned}$$

- Step 1 is Bayes rule
- Step 2 is the assumption that the prior is broad

Parameter estimation

- We can thus substitute our b repn. of $p(\underline{x},\sigma \mid x_n)$ and a repn of $p(x \mid x_{n+1})$ into this repn, to get

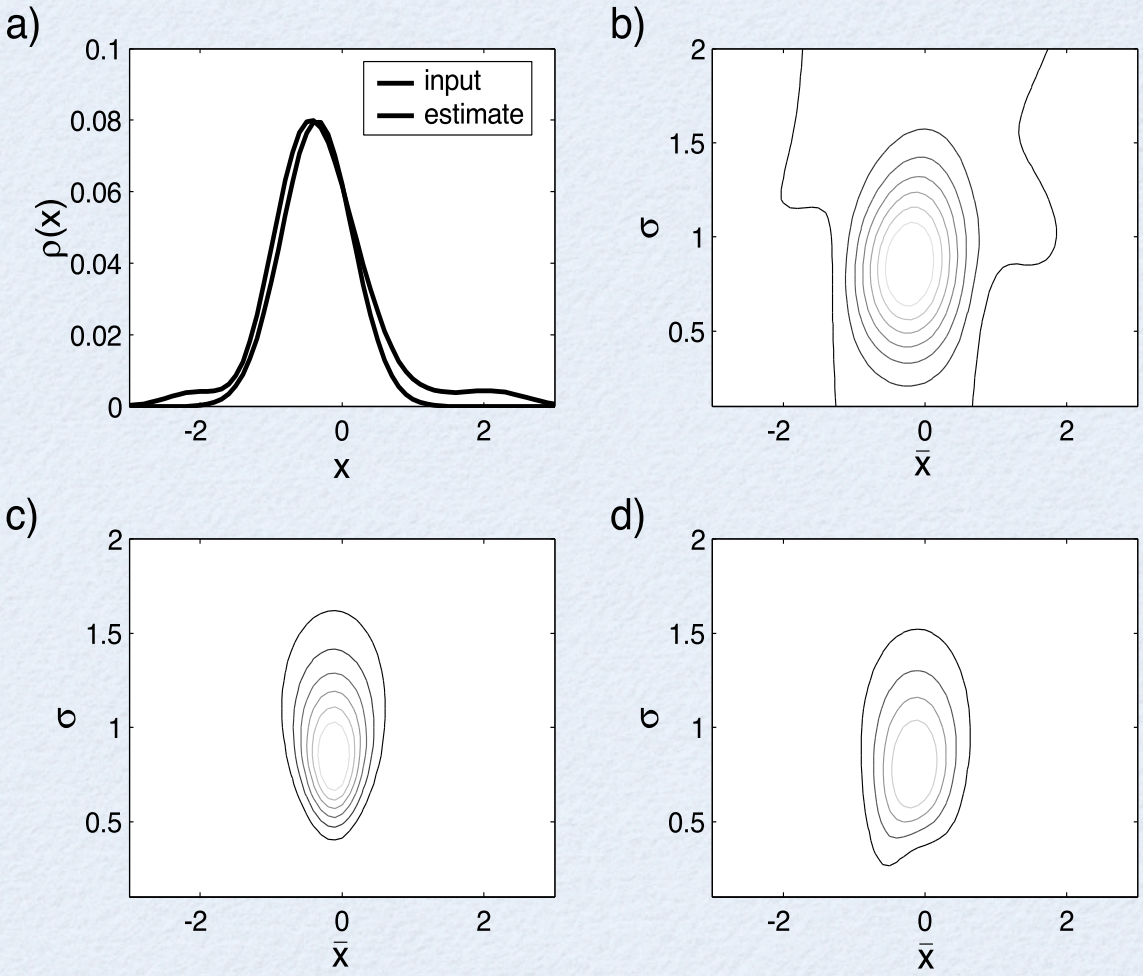
$$b_j(x_{n+1}) = G_j \left[\sum_{il} \omega_{jil} a_i(x_{n+1}) b_l(x_n) + J_j^{bias} \right]$$

- where

$$\omega_{jil} = \alpha_j \left\langle \tilde{\phi}_j(\bar{x},\sigma) \rho(x|\bar{x},\sigma) \phi_i(x) \phi_l(\bar{x},\sigma) \right\rangle_{x,\bar{x},\sigma}$$

Parameter estimation

- Run mplayer (parmest, parmest2.avi)



Parameter estimation

- This derivation holds regardless of our explicit restrictions on the shape of $p(\underline{x},\sigma \mid x_n)$ or $p(x \mid x_n)$
- Just one of many ways of doing this kind of estimation
- Notice that there is no explicit learning/change in weights, yet the system is adaptive

Summary

- Learning and statistical inference are both about 'modeling' the (messy, uncertain) world
- There are a variety of ways of constructing such models, each with different resource constraints/ assumptions, etc.
- A huge challenge to figure out what the 'best' assumptions are -- looking to the brain should help