=================================================================

SECTION: SCADS Replication and NN Extension

We obtained the original 1990 SCADs lisp code from JS's backup files. That code has three main modules: The "ADD" module is similar to the Siegler and Shrager 1984 model. It uses an associate table to try to retrieve an answer to an addition problem. If an answer problem cannot be retrieved above a confidence threshold, it chooses an explicit strategy to use, executes it and then updates an association table with the results. One advance over the 1984 model in this area is that there are multiple strategies, so, as in Siegler and Shipley, statistics are kept on which strategies perform better in which problems, and this is used to choose which strategy to execute. The "STRATEGY LEARNER" module changes strategies. This is the major advance between 1984 and 1990 models. The "EXPERIMENT" module drives simulation runs, collects data, and analyzes the results. This module is highly specific to the particular experiments reported in the SCADS 90 paper. We broke the code up into the above three parts and then rewrite only the "ADD" module in python. The strategy learner is not necessary for the present experiments, and the experiment module was rewritten from scratch for the present experiments. The major change to the ADD module was to replace the association matrix with a simple off-the-shelf python three-layer neural network. (I think that this came from: http://www.bogotobogo.com/python/python\_Neural\_Networks\_Backpropagation\_for\_XOR\_using\_one\_hidden\_layer.php)

We started with two "basic sanity" predictions that should hold before we believe that should be stably replicable in order for us to believe that the model works at the basic level, before we address more interesting questions. These are:

(1) U-Shaped Retrieval Usage: We predict a u-shaped retrieval curve where retrieval is heavily used early in the simulation's "development", but these retrievals report a preponderance of incorrect answers, analogous to a child's babbling (or, more mathematically relevant, believing that she knows the answer, but getting it wrong). This early high-retrieval/low-correctness period will rapidly extinguish, leading to an extended period of explicit strategy use, which will slowly tail off to retrieval again, but this time with much more correct sums.

(2) S-Shaped (or Tangential) Correctness: As above, retrieval will be used early on, but will give incorrect results. As this early (incorrect) retrieval extinguishes, and explicitly strategies come to be employed, we will see more correct answers. Not very far into the run answers will come to be much more correct, mostly (in this middle part) as a result of the correctness of strategies (controlled by the PERR parameter), followed, eventually, by the end, in more "adult" phase where retrieval is used, but this time is mostly correct.

Both of these results were obtained. (Insert figures)

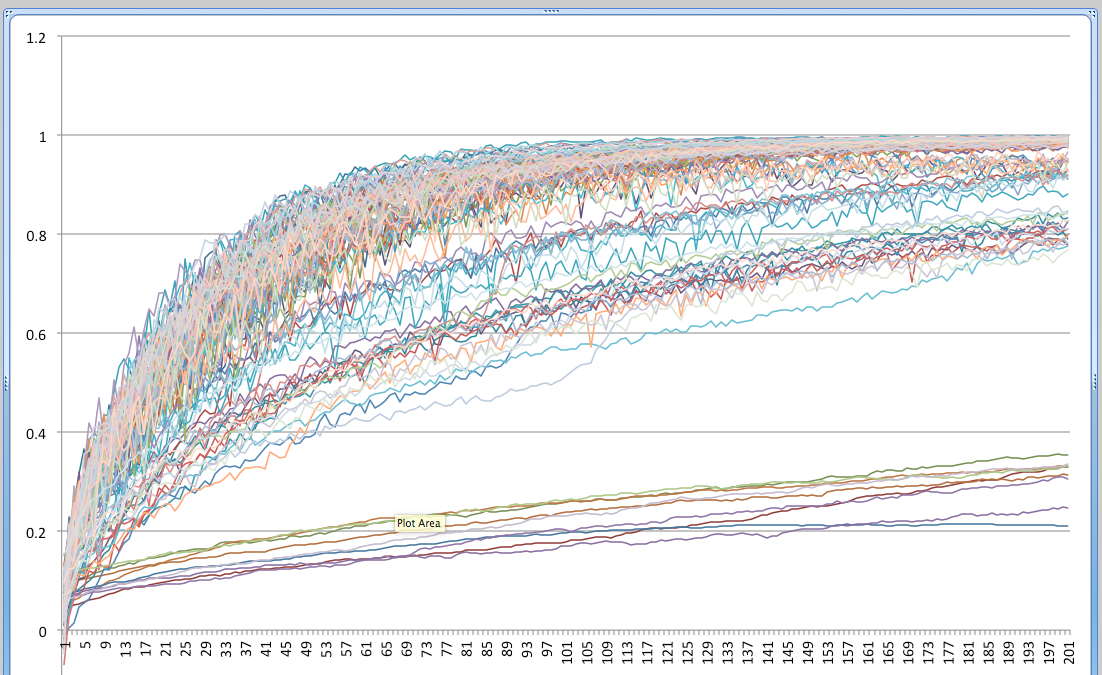
Another important prediction is the prediction from Siegler and Shipley and later Shrager and Siegler, that: (3) Final Primary Retrieval: Excluding the very early period, retrieval will come to dominate as the primary strategy (over other explicit strategies). This is analogous to the adult, who can accurately recall the answer to any positive single digit addition problem whose sum is under 10. This prediction also obtained.

It is easy to see how these results arise from the model: Early on the NN weights are set randomly -- or are burned into the counting facts [we need to test this] [note params that control this] --

=================================================================

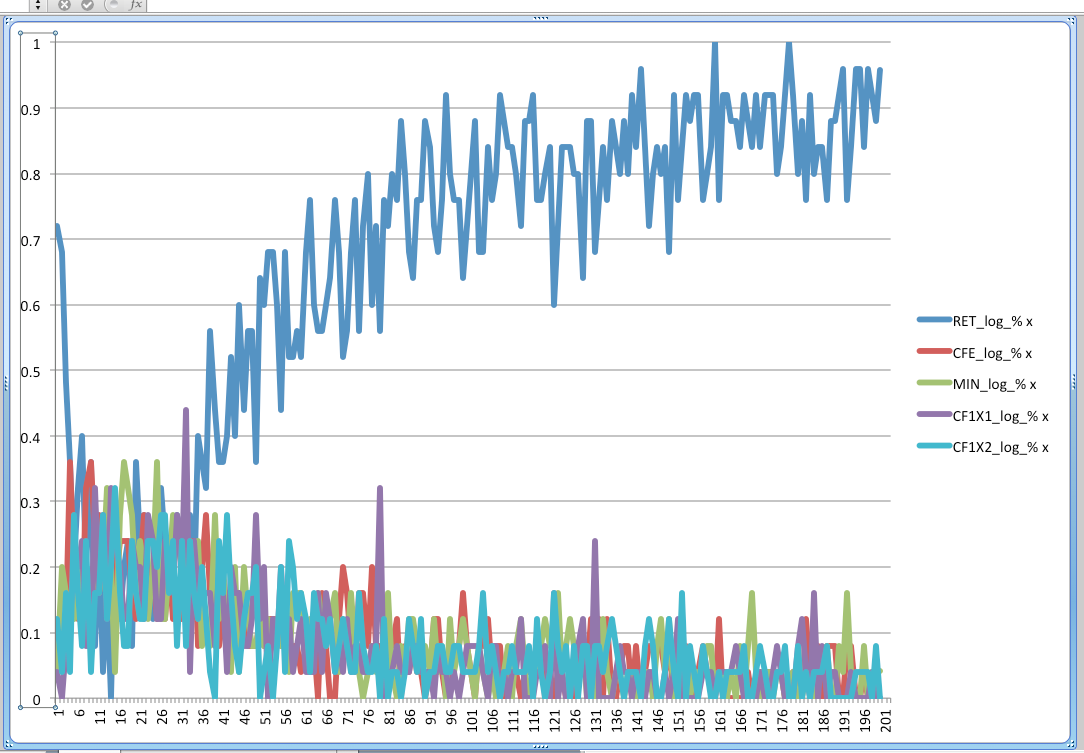
Experiment: Basic Sanity -- Improving correctness toward adult responses

Adult cc trajectories from: 201509010826: scanning perr, (internal)epochs, and learning rate)

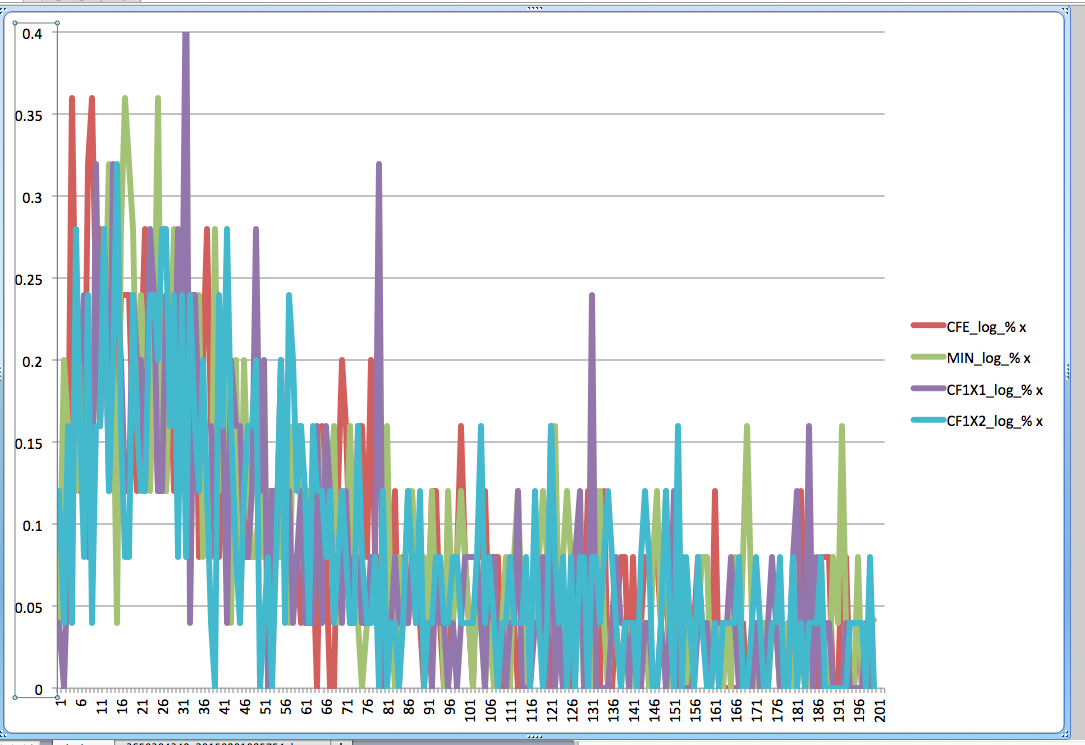


=================================================================

Experiment: Strategy usage

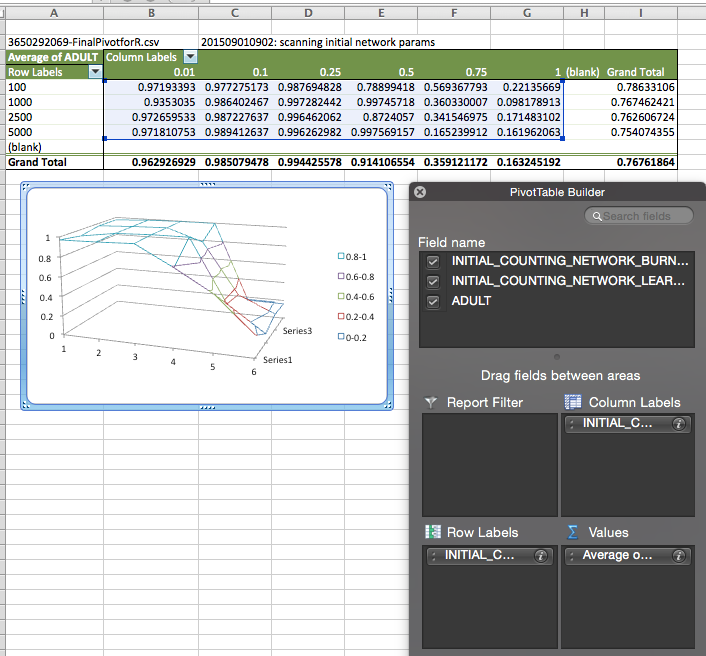


Excluding retrieval:



=================================================================

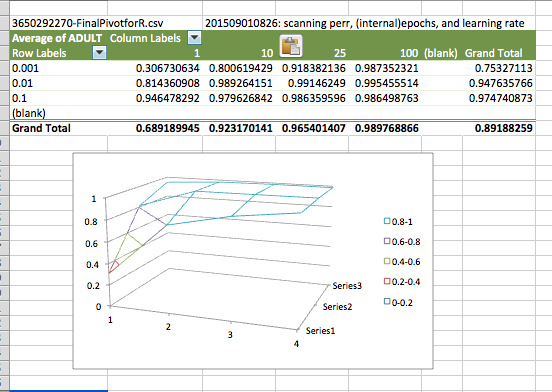
Experiment modulating initial counting network burn in.



=================================================================

Experiment modulating learning rate and in-process-training-epochs:

(201509010826: scanning perr, (internal)epochs, and learning rate)



Note that EITHER learning rate OR number of training epochs will improve the learning.

=================================================================

experiment: 201509031445: Long run with high prob. err and tight ret thresh to expose strategy usage separations

settings.DECR\_on\_WRONG=-1.0

settings.non\_result\_y\_filler=0.0

settings.initial\_counting\_network\_burn\_in\_epochs=1000

settings.DR\_threshold=1.0

settings.initial\_counting\_network\_learning\_rate=0.25

settings.RETRIEVAL\_HIGH\_CC=1.0

settings.INCR\_the\_right\_answer\_on\_WRONG=0.0

settings.STRATEGY\_LOW\_CC=0.6

settings.STRATEGY\_HIGH\_CC=1.0

settings.addend\_matrix\_offby1\_delta=1.0

settings.RETRIEVAL\_LOW\_CC=0.95

settings.PERR=0.5

settings.learning\_rate=0.01

settings.in\_process\_training\_epochs=10

settings.INCR\_on\_RIGHT=1.0

settings.n\_problems=50000

This failed. Stay tuned.

=================================================================

SECTION: Problematizing "simple" in "simple addition strategies"

Aside from isolated numbers, and counting (which, in some cases, can be thought of as merely a song or sentence [citation?]), single digit addition is the most common place that children are introduced to any sort of complex number manipulation [citation; maybe piaget?]. Yet, even this seemingly simple skill, when one considers it from the standpoint of the cognitive manipulations required, is extremely complex.

Let's look in detail at what seems at first an unproblematic simple finger-counting strategy for adding 3+4. At the surface, it looks quite simple:

1A. Choose a hand, let's say the left hand.

1B. Count up from 1 to 3 while raising three fingers on that (left) hand.

1C. Shift your attention to the other (right) hand.

1D. Count up from 1 to 4 while raising four fingers on that (right) hand.

1E. No go back to the first (left) hand.

1F. Start counting from 1 all the fingers that are up on both hands.

This is sometimes called the "sum" strategy, and we represent the activity as 123+1234/1234567=>7, where the + separate the hand-based finger counting, and at the / the child recounts the fingers and then => is reporting the sum.

Although this seems simple enough, looking at the process in more detail reveals some interesting complexities. One of the most intriguing is this: How does the child get the addends straight? Although the second addend, 4, was said last in the problem presentation, and so should have priming support from recency, the child must choose the earlier, smaller, 3, to count first. The child must then go back, now much farther in memory, and pick up the second addend, 4, and count that up, whereas she has just counted up to 3 on the first addend. Why doesn't she often accidentally pick up 3 twice? Why doesn't she pick up 5, which is the natural and heavily overlearned (by this point, assuming that she is a good counter) follow-on to "3...4..." Where does the second addend (4) live while the counting up of the first addend (3) is taking place? And notice that counting up to 4 (in step 1D) goes right through 3, which was set as the stop point in step 1B, yet in step 1D, the child has to remember that the second addend was 4, and not accidentally stop at 3. One interesting question is whether, in fact, some of the possible errors predicted by the complex overlapping of memory traces just described are observed.

Here's another strategy that gets to the same end:

2A. Choose a hand, let's say the left hand.

2B. Count up from 1 to 3 while raising three fingers on that (left) hand.

2C. Shift your attention to the other (right) hand.

2D. Count up from 4 to 7 while raising four fingers on that (right) hand.

This is sometimes called the Shortcut Sum Strategy, and we represent the action as 123+4567=>7, note that there is no recounting (/) step here.

In addition to the problem of recovering the second addend in the Sum strategy, the counting up part of the Shortcut Sum strategy requires that the child somehow recognize when she has reached 4 additional fingers on the right hand without actually saying the number 4, or, rather, stopping on 7 instead of 4 (when 4 is said aloud at the beginning of the counting in step 2D).

A possible clue come from the difficulty people (even adults) have in not raising their fingers if they are forced to execute finger counting strategies (even though they can usually simply retrieve the answer without any effort at all). One way of thinking of what is going on here is to posit two new cognitive activities. First, we posit that when the child hears that the problem is one of addition, the activated strategy grabs the two addends out of very short term number identity memory and "hangs them", so to speak, someplace (in working memory, it is usually said), until the are needed to play their respective roles. The strategy can pick these up off their "hangers" (aka. out of working memory), when needed. Second, we posit that the child can recognize (subitize) the number of fingers on one hand (that is, up to 5). Indeed, this ability to subitize small numbers may play one in the same role as the addend "hangers" just previously posited; that is, what is mean by "hanging" these numbers in working memory is exactly to deploy "recognizers" waiting for these numbers to show up on one's hands, under the assumption that simple recognizers (for things like small numbers in the subitizing range) are like attentional spotlights that can be set up to "lay in wait", as a sort of "stop limit", for finger sets with the correct numerosity to appear, and trigger the strategy to go on to the next phase (report out the sum). So, in the respective steps D (1D and 2D), the child needn't recover the second addend at all, but merely start counting from 1 again (in 1D), or from where she was (in 2D), and having the subitizing "stop limit" trigger when she gets the right number of fingers up on the second hand.

This theory would predict an interesting pattern of errors where the "stop limits" fire incorrectly, resulting in various complex confusions. For example, when solving 3+4 the correct performance is: Sum:123+1234/1234567=>7 or Shortcut Sum:123+4567=>7, but with various misfirings of the stop limits, could produce any of these:

123+1234/1234567=>7 CORRECT Sum

123+4567=>7 CORRECT Shortcut Sum

123=>3 Early termination on first trigger

1234=>4 Early termination on wrong first trigger

123+123=>3 Wrong second trigger

123+1234=>4 Early termination with correct triggers

1234+1234=>4 Wrong first trigger and early termination

1234+1234/12345678=>8 Wrong first trigger

123+123/123456=>6 Wrong second trigger

123/123=>3 Early continuation

1234/1234=>4 Early continuation on wrong first trigger

123+1234/123=>3 Early termination of counting on first trigger

123+1234/1234=>4 Early termination of counting on second trigger

1234/123=>3 Early continuation on wrong first trigger and

early termination of counting on first trigger

1234+123/1234567=>7 Wrong first and second triggers in Sum

\*\*\* = CORRECT Reverse Sum Strategy

567=>7 CORRECT Min Strategy

Note that we assume in the above that counting of fingers is always accurate (when it takes place at all; that is, no fingers are skipped or double counted), and that finger rasing is similarly accurate. Although these are not reasonable assumptions -- these things take place all the time, esp. in young children -- but we cannot list every possible such error here, and, more importantly, such errors are trivial regarding the details of the cognitive strategies, at least as concerns us here.

The point: What one calls a "strategy" is important and complex. The reason that it is important is that people teach what they think of as strategies to others, for example, math teachers teach mathematical strategies to students, and when people "think strategically" they are often thinking about deploying strategies at this level of analysis. Also, when we talk about "multi-tasking", what we mean to indicate by this is the interweaving of stratagies, again taken at that level of analysis. We tell ourselves that strategies are algorithms for getting things done. But none of the foregoing is as simple as it may at first seem. The goal of this paper is to problematize the concepts of "strategy", of "task", and thereby of "multi-task", as well as the relationship between "algorithms" qua the things that humans program into computers, v. cognitive strategies qua the things that humans actually do in order to get things done. (This should not be confused with the philosophical question of brains v. computers. Rather, this is a much narrower technical point re: human strategies v. computer algorithms.)

In this paper we study the fine detail of very simple addition strategies carried out by children in order to reveal the necessary and likely complexities of such activities. We will study two strategies, the "sum" strategy, and the "count from first" strategy. Note that these strategies are not actually perfectly well specified in the literature, and there are many versions that could possibly be categorized into this. This is part of the reason for the present exploration is to expose complexities such as these. But we will begin with one specific version of each strategy.

Shrager and Siegler describe The Sum Strategy for children's addition of small numbers (5 or less) this way: For the problem "3+4": Put up three fingers, usually saying "1, 2, 3" then put up four fingers on the other hand, usually saying "1, 2, 3, 4" then count the fingers, saying "1, 2, 3, 4, 5, 6, 7". But let's really dig into what has to happen in order for the child to execute this strategy. Here we describe the "correct" path, although along the way we will point out some places where errors or shortcuts are likely (and probably observed in the data). Moreover, we will try to identify brain regions that are likely to be associated with some of the machinery that we will describe, and how these regions may interact with one another.

What skills does the child being into the task setting? Let us assume that the problem is presented verbally; i.e., the interrogator asks the child aloud: "What is three plus four?" Let us further assume that the child is expert enough in language to recognize "What is ... plus ...?" as being asked to add two numbers, and also that the child has enough control of their echoic short term memory to be able to recall the two numbers is asked immediately after what the two numbers were.

The layering to a time-serial "cascade" of NN inputs of the sequential expresion of the problem: ["what" "is"] "three" "plus" "one" "?" -- Christian's point that even in adults 2\*3=5 tends to get a "true" response, and the distinction between "+" and "\*" has to be learned and so should go into a distributed semantic representation.

The use of the echoic buffer and what sort of attentional focused and/or recurrent nn can model this...

MIX IN MULTI-TASKING!!