The Memory Tesseract: Distributed MINERVA and the Unification of Memory

Matthew A. Kelly (matthew_kelly2@carleton.ca) [1]
Douglas J. K. Mewhort (mewhortd@queensu.ca) [2]
Robert L. West (robert_west@carleton.ca) [1]

[1] Institute of Cognitive Science, Carleton University
1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6 Canada

[2] Department of Psychology, Queen's University
62 Arch Street, Kingston, Ontario, K7L 3N6 Canada

Abstract

We prove that MINERVA 2, a widely-used model of biological long-term memory, is mathematically equivalent to an auto-associative memory implemented as a fourth order tensor. We further propose an alternative implementation of MINERVA 2 as a holographic lateral inhibition network. Our work clarifies the relationship between MINERVA 2 and other memory models, and shows that MINERVA 2 and derivative models can be neurally implemented and scaled-up to long-term learning tasks.

Keywords: memory; cognitive modelling; MINERVA 2; vectors; tensors; holographic reduced representations; HRRs.

Introduction

Most memory phenomena can be explained by unified, computational memory models (e.g., Franklin & Mewhort, 2013; Hintzman, 1984; Humphreys, Bain, & Pike, 1989; Jamieson & Mewhort, 2011). Simulation has led to parsimonious theories of memory, but at a cost of a profusion of competing models. As different models focus on different phenomena, there is no best model.

Simulation models share many characteristics indicating wide agreement about the mathematics of how memory works. Here, we argue that memory models, including the MINERVA 2 (Hintzman, 1984) model and variants (e.g., Jamieson, Crump, & Hannah, 2012; Jamieson & Mewhort, 2011; Kwantes, 2005; Thomas et al., 2008), as well as holographic models of short/long term memory (e.g., Eich, 1982; Franklin & Mewhort, 2013; Murdock, 1993), and the DSHM model of declarative memory (Rutledge-Taylor & West, 2008) which uses the BEAGLE learning algorithm (Jones & Mewhort, 2007), can be understood in terms of a single neurally plausible memory framework.

This effort at unification is based on the MINERVA 2 model (Hintzman, 1984), a computational model of biological memory, intended by Hintzman to describe long-term memory (both episodic and semantic). It has been applied to several experimental paradigms, including judgement of frequency tasks (Hintzman 1984), recognition tasks (Hintzman, 1984), "schema-abstraction" or category learning (Hintzman, 1984; 1986), implicit learning tasks such as artificial grammar learning (Jamieson & Mewhort, 2011), as well as speech perception (Goldinger, 1998), and naming words from print (Kwantes & Mewhort, 1999).

Variations on the MINERVA 2 model address an even broader range of phenomena. MINERVA-AL makes and corrects predictions to capture numerous associative learning phenomena from both the animal and human learning literature (Jamieson, Crump, & Hannah, 2012). Kwantes (2005) used MINERVA to study how semantic similarity can be learned from word co-occurrence in the language. Thomas et al. (2008) used MINERVA to study hypothesis generation and probability judgement in humans.

In this paper, we use the term MINERVA 2 to refer specifically to Hintzman's model, and MINERVA to refer collectively to MINERVA 2 and any model based on it.

Our central contribution is to prove that MINERVA is mathematically equivalent to an auto-associative fourth order tensor memory, or *memory tesseract*. We further demonstrate that MINERVA is approximately equivalent to a holographic lateral inhibition network (Levy & Gayler, 2009). These demonstrations have three implications for memory modelling: (1) the relationship between MINERVA and other memory models is clarified, suggesting that MINERVA may be suitable as a basis for all memory modelling, (2) MINERVA is scalable to arbitrarily long-term learning, and (3) MINERVA is neurally plausible.

How does MINERVA work?

Hintzman (1986) describes MINERVA's key assumptions:
(1) only episode traces are stored in memory,
(2) repetition produces multiple traces of an item,
(3) a retrieval cue contacts all traces simultaneously,
(4) each trace is activated according to similarity to the cue,
(5) all traces respond in parallel, the retrieved information reflecting their summed output.

Each individual experience, or episode, is represented by a high dimensional vector. Memory is a table, where each row is a vector representing an 'episode trace' corresponding to a stored experience. Each new experience is stored as a new row in the memory table. New experiences do not need to be novel. A repeated experience is also stored as a new row, separate from previous instances of that experience.

In MINERVA, memory retrieval is not a look-up process, it is a reconstruction process. In the words of Tulving and Watkins (1973), "a probe combines or interacts with the stored information to create the memory of a previously experienced event". When a retrieval cue is presented, each

vector in the table "resonates" with the cue in proportion to its similarity to the cue. Similarity is computed as a cosine (i.e., normalized dot-product) of the cue with the stored vector. Each vector is activated by its cubed similarity to the cue. Information is retrieved from memory in the form of a new vector, called an echo. The echo is a weighted sum of the vectors in the table, each vector weighted by its activation. By computing activation as the cube of similarity, the contribution of the most similar vectors (or experiences) is emphasized and that of the least similar (and least relevant) is minimized. The echo is used by the model to respond to the cue as appropriate for the given task.

Abstract, conceptual, and categorical information reflect aggregate retrieval over many episode traces. The blending of experiences in the echo is one source of our ability to use abstractions (e.g., Goldinger, 1998).

According to Hintzman (1990), MINERVA 2 can be understood as an artificial neural network. A layer of input nodes represent the cue. A layer of output nodes represent the echo. Between the two is a hidden layer of nodes. In the hidden layer, each node corresponds to an episode trace. It follows that MINERVA's hidden layer is a localist network: specific nodes represent specific pieces of information.

Modellers using MINERVA are generally agnostic as to how the model is related to the brain. No one claims that for each new experience one grows a new neuron that is forever singly dedicated to that particular experience. But no other interpretation of how MINERVA is related to the brain has been previously proposed, leaving open the question of MINERVA's neural and, hence, theoretical plausibility.

A comparison of memory models

The memory models discussed here use vector and tensor representations to simulate the processes of storage and retrieval. Tensors are a generalization of matrices. A vector is a first order tensor. A matrix is second order tensor. A third order tensor is a "3D matrix" or a stack of matrices.

Memory models can use either localist or distributed representations and have either localist or distributed memory stores. Vector-based memory models use *distributed representations*, that is to say, an item to be remembered is represented as a high dimensional vector, which can be thought of as a pattern of activation across nodes in a network. Conversely, in a network that uses *localist representation*, an item is represented by the activation of a single node, as opposed to a pattern of activation across a group of nodes.

In vector-based memory models, the memory store can be either *localist* or *distributed*. By a *localist store*, we mean that a model stores different data in different places. By a *distributed store*, we mean a model that stores all data in a single place or all data in all places. For our purposes, vector-based memory models can be divided into four categories: *vector memory*, where all memories are stored in a single vector, *matrix memory*, where all memories are stored in a single matrix, *tensor memory*, where all memories are stored in a single, higher-order tensor (e.g., a "3D" or "4D" matrix), and *multi-vector memory*, where multiple vectors are used to store memories. *Multi-vector* memory models, such as MINERVA, and DSHM (Rutledge-Taylor & West, 2008) use localist stores because different vectors are used to store different memories, whereas *vector* (e.g., Eich, 1982; Franklin & Mewhort, 2013; Murdock, 1993), *matrix* (e.g., Humphreys, Pike, Bain, & Tehan, 1989; Howard & Kahana, 2002), and *tensor* memory models (e.g., Humphreys, Bain, & Pike, 1989; Smolensky, 1990) use distributed stores. Across these four categories, there are strong similarities between models, as we will now discuss.

Matrix and tensor models

Humphreys et al. (1989) note that their matrix memory, MINERVA 2, and TODAM (Murdock, 1993) retrieve information as a sum of all traces in memory, each trace weighted by its similarity to the cue. As we will show, these models are not different in kind, and so we elect to use the term *echo*, normally reserved for MINERVA, to refer to the retrieved vector in all of these memory models.

A key point of comparison is how vector-based models represent associations between items. Smolensky (1990) notes that the tensor product, a generalization of the matrix product, can be used to form associations between an arbitrary number of items, though at the cost of producing progressively larger and more unwieldy tensors.

The order of the tensor used to store memories indicates the number of vectors the memory model associates together. An association between a pair of items is represented as the tensor product of the items' vectors, which is a second order tensor, or matrix. A matrix memory is the sum of those associations. Howard and Kahana's (2002) matrix memory is **context** x **item**, that is, it associates a vector representing an item with a vector representing a context. Humphreys et al. (1989) matrix memory is **item a** x **item b**, that is, it associates two different items together.

If three items are being associated, the result is a "3D matrix", or third order tensor. Humphreys, Bain, and Pike's (1989) third order tensor memory is **context** x **item a** x **item b**, that is, it associates two different items together with a representation of context.

There is a problem here. Across these models, the architecture of memory is being modified to suit the particulars of the tasks being modelled. If we just need one cue (be it an item or context), we use a matrix memory. If we use two cues (be it an item and context, or two items) we use a third order tensor. But what if we need to use three cues? Do we then need to use a fourth order tensor? What about four cues? Using this approach, not only does the architecture of memory need to be changed depending on the particulars of the task, but the architecture becomes increasingly unwieldy as the task becomes more complex.

Vector models

Holographic vectors can represent arbitrarily complex associations of items and context, making it unnecessary to use matrices or higher order tensors to represent association, allowing modellers to adopt a memory architecture that is invariant with respect to the complexity of the associations.

In a vector memory, an association between a set of items is the convolution of the vectors representing those items. Convolution is a noisy compression of the tensor-product (Plate, 1995; for discussion see Kelly, Blostein, & Mewhort, 2013), thus vector models differ from matrix and tensor

models only in that the highly lossy nature of holographic vector storage adds noise to the echo, and that individual items and associations between sets of any size can all be stored together as a sum in a single vector memory.

Holographic vectors do have one weakness: they are highly lossy. This is the only reason one might prefer the aforementioned matrix or tensor memories over a holographic one. However, combining holographic vectors with MINERVA creates a system that can store arbitrarily complex associations between items and contexts, and retrieve them with fidelity (Jamieson & Mewhort, 2011).

MINERVA

MINERVA 2 differs from vector or matrix models in that:

(1) MINERVA 2 associates items by either adding together or concatenating the vectors representing those items, rather than using the tensor-product or convolution.

(2) All traces in the echo of MINERVA 2 are weighted by the cubed similarity to the cue. In a vector or matrix model, the similarity is not raised to an exponent.

The first difference is not essential. The Holographic Exemplar Model (Jamieson & Mewhort, 2011) is a MINERVA that uses convolution rather than concatenation, gaining the ability to represent arbitrarily complex associations.

The second difference is critical. Raising similarity to an exponent of 3 sets the MINERVA models apart from the vector and matrix models. The purpose of this exponent is to make MINERVA nonlinear, as Hintzman explains (1990):

> This model escapes being just a less efficient version of the vector model by using nonlinearity. In particular, the activation of each hidden unit is a positively accelerated function of its match to the input vector, limiting the number of units that will respond significantly to any input, and thereby reducing noise.

By weighting each episode trace by the cube of its similarity, the traces that are most similar to the cue contribute much more to the echo than traces that have only partial similarity to the cue, or traces that have tiny, incidental similarity to the cue. Cubing similarity weights retrieval in favour of an exact match to a single item in memory over partial matches to several items in memory or slight matches with a very large number of items in memory.

This characteristic allows MINERVA to "clean-up" the echo using iterative retrieval. MINERVA can "clean up" an echo by using the echo as a cue to produce a new echo. With each pass through the memory system, the contribution of the most similar episode traces grows. This process can be repeated until the echo reaches a steady-state where it no longer changes, at which point the echo will closely resemble the trace in memory most similar to the cue.

The clean-up process also serves as a possible explanation for why we are faster to remember some things than others: echoes formed from frequently occurring and distinctive episode traces reach a steady-state more quickly.

Linear systems cannot perform a clean-up process and so require an external clean-up memory. The echo produced by a holographic vector memory (e.g., TODAM; Murdock, 1993) or matrix memory (e.g., Humphreys et al., 1989) is noisy and requires a secondary (or long-term) memory in order to identify which memory the echo most resembles (Murdock, 1993). This is acceptable for TODAM, as it is intended as a model of primary (or working) memory. MINERVA, however, is a model of long-term memory and thus it would "violate the spirit of MINERVA 2" (Hintzman, 1986) to rely on an external memory to clean-up the echo.

A disadvantage of MINERVA is that it is a localist memory store, which raises the question of neural plausibility, as well as scalability. How is MINERVA neurally implemented, if not by growing new neurons for each new experience? Is MINERVA practical for very long-term learning given that each experience adds another row to the memory table? While MINERVA has been applied to large problems (Kwantes, 2005; Kwantes & Mewhort, 1999), doing so requires abandoning key assumptions of the model for the sake of computational feasibility.

Given the usefulness of using the cube of the similarity in MINERVA 2, we want to keep this feature in a variant of the model that uses a distributed memory store.

MINERVA as a fourth order tensor

In what follows, we prove equivalence between the MINERVA 2 model and an auto-associative fourth order tensor memory. To do so, we first prove that a variant of MINERVA that raises similarity to an exponent of 1 is equivalent to an auto-associative second order tensor (i.e., a matrix) memory. Then we prove that a MINERVA that uses an exponent of 2 is equivalent to a third order tensor. Finally, we prove that the MINERVA 2 model, which uses an exponent of 3, is equivalent to a fourth order tensor.

Consider a variant on the MINERVA model that uses dot product (denoted by •) to measure similarity and weights each trace by its similarity raised to the exponent of 1. Each episode trace in memory is represented by a vector $\mathbf{v}_i$ where $i = 1 \ldots m$ and m is the number of traces in memory. When the model is presented with a cue $\mathbf{x}$, the echo $\mathbf{y}$ is:

$$\mathbf{y} = (\mathbf{x}\bullet\mathbf{v}_1)\mathbf{v}_1 + \ldots + (\mathbf{x}\bullet\mathbf{v}_m)\mathbf{v}_m$$

This is equivalent to an auto-associative matrix memory.

In an auto-associative matrix memory, each episode trace is represented by a vector $\mathbf{v}_i$. To store a trace in memory, the trace is associated with itself (hence *auto-associative*) by taking the outer-product of the vector with itself, $\mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}$, then taking the sum of all the outer-product matrices to create the memory matrix, $\mathbf{M}$:

$$\mathbf{M} = \mathbf{v}_1 \mathbf{v}_1^{\mathrm{T}} + \ldots + \mathbf{v}_m \mathbf{v}_m^{\mathrm{T}}$$

The echo, $\mathbf{y}$, is the inner-product of the cue and the matrix:

$$\mathbf{y} = \mathbf{M}\mathbf{x}$$
$$\mathbf{y} = (\mathbf{v}_1 \mathbf{v}_1^{\mathrm{T}} + \ldots + \mathbf{v}_m \mathbf{v}_m^{\mathrm{T}})\,\mathbf{x}$$
$$\mathbf{y} = \mathbf{v}_1 \mathbf{v}_1^{\mathrm{T}}\mathbf{x} + \ldots + \mathbf{v}_m \mathbf{v}_m^{\mathrm{T}}\mathbf{x}$$

Because $\mathbf{v}_i^{\mathrm{T}}\mathbf{x}$ is the dot-product of $\mathbf{v}_i$ and $\mathbf{x}$:

$$\mathbf{y} = (\mathbf{x}\bullet\mathbf{v}_1)\mathbf{v}_1 + \ldots + (\mathbf{x}\bullet\mathbf{v}_m)\mathbf{v}_m$$

which is identical to MINERVA with an exponent of 1 or to a matrix memory (e.g., Humphreys et al., 1989). Consider a MINERVA that raises similarity to the exponent of 2:

$$\mathbf{y} = (\mathbf{x}\bullet\mathbf{v}_1)^2\,\mathbf{v}_1 + \ldots + (\mathbf{x}\bullet\mathbf{v}_m)^2\,\mathbf{v}_m$$

This variant of MINERVA, as we shall demonstrate, is mathematically equivalent to an auto-associative third order tensor memory. Using the tensor product, denoted by ⊗, we

can store each trace as $\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i$, which is a third order tensor. The memory tensor **M** is the sum of the third order tensor outer-products of each episode trace:

$$\mathbf{M} = \mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1 + ... + \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m$$

The echo, **y**, can be computed from the cue, **x**, by taking the inner product *twice*:

$$\mathbf{y} = (\mathbf{M}\ \mathbf{x})\ \mathbf{x}$$

If each $\mathbf{v}_i$ is a vector of n dimensions, then **M** is an n x n x n tensor. **M** can be thought of as n matrices of n x n dimensions. When we compute the inner-product of **M** with the cue **x**, we compute the inner product of **x** with each of those n matrices. This results in n vectors that can be rearranged into a new n x n matrix. The second inner product with **x** then produces a vector, the echo **y**.

To illustrate, let us break **M** into its components:

$$\mathbf{y} = (\mathbf{M}\ \mathbf{x})\ \mathbf{x}$$
$$\mathbf{y} = ((\mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1 + ... + \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m)\ \mathbf{x})\ \mathbf{x}$$
$$\mathbf{y} = (\mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1\ \mathbf{x} + ... + \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m\ \mathbf{x}\ )\ \mathbf{x}$$

The tensor product $\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i$ can be understood as n matrices, where each matrix is the outer-product $\mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}$ weighted by a different element j of $\mathbf{v}_i$, for all $j = 1 ... n$.

$$\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i = \{v_{i\,1}\ \mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}\ , ...\ , v_{i\,n}\ \mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}\}$$

Taking the inner-product of the cue **x** with $\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i$, we get n vectors, each weighted by the dot-product of **x** with $\mathbf{v}_i$:

$$\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i\ \mathbf{x} = \{v_{i\,1}\ \mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}\ \mathbf{x},\ ...\ ,\ v_{i\,n}\ \mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}\ \mathbf{x}\}$$
$$\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i\ \mathbf{x} = \{v_{i\,1}\ (\mathbf{x} \bullet \mathbf{v}_i)\ \mathbf{v}_i,\ ...\ ,\ v_{i\,n}\ (\mathbf{x} \bullet \mathbf{v}_i)\ \mathbf{v}_i\}$$

If we factor out the dot-product of **x** and $\mathbf{v}_i$, the result is n vectors, or rather, the outer-product matrix of $\mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}$:

$$\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i\ \mathbf{x} = (\mathbf{x} \bullet \mathbf{v}_i)\ \{v_{i\,1}\ \mathbf{v}_i,\ ...\ ,\ v_{i\,n}\ \mathbf{v}_i\}$$
$$\mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i\ \mathbf{x} = (\mathbf{x} \bullet \mathbf{v}_i)\ \mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}$$

Thus, when we take the outer-product of **x** with **M**, the result is a sum of m matrices $\mathbf{v}_i \mathbf{v}_i^{\mathrm{T}}$, each matrix weighted by the dot-product of **x** with $\mathbf{v}_i$.

$$\mathbf{y} = (\mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1\ \mathbf{x} + ... + \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m\ \mathbf{x}\ )\ \mathbf{x}$$
$$\mathbf{y} = ((\mathbf{x} \bullet \mathbf{v}_1)\ \mathbf{v}_1 \mathbf{v}_1^{\mathrm{T}} + ... + (\mathbf{x} \bullet \mathbf{v}_m)\ \mathbf{v}_m \mathbf{v}_m^{\mathrm{T}}\ )\ \mathbf{x}$$

By then taking the second inner-product with **x**, we reduce each matrix to a vector weighted by the squared similarity to **x**, producing an echo like MINERVA with an exponent of 2:

$$\mathbf{y} = (\mathbf{x} \bullet \mathbf{v}_1)\ \mathbf{v}_1 \mathbf{v}_1^{\mathrm{T}}\ \mathbf{x} + ... + (\mathbf{x} \bullet \mathbf{v}_m)\ \mathbf{v}_m \mathbf{v}_m^{\mathrm{T}}\ \mathbf{x}$$
$$\mathbf{y} = (\mathbf{x} \bullet \mathbf{v}_1)\ (\mathbf{x} \bullet \mathbf{v}_1)\ \mathbf{v}_1 + ... + (\mathbf{x} \bullet \mathbf{v}_m)\ (\mathbf{x} \bullet \mathbf{v}_m)\ \mathbf{v}_m$$
$$\mathbf{y} = (\mathbf{x} \bullet \mathbf{v}_1)^2\ \mathbf{v}_1 + ... + (\mathbf{x} \bullet \mathbf{v}_m)^2\ \mathbf{v}_m$$

The dot product of two vectors, where each vector has a magnitude of one, produces a value in the range of one, if the vectors are identical, to zero, if the vectors are orthogonal, to negative one, if one vector is the negation of the other. Thus it is important to preserve the sign of the dot product. By taking the square of the dot product, the sign is lost. For this reason, MINERVA 2 uses an exponent of 3.

MINERVA 2 is equivalent to an auto-associative memory implemented as a fourth order tensor. Memory is constructed as a sum of fourth order tensors:

$$\mathbf{M} = \mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1 + ... + \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m$$

Given a cue **x**, an echo **y** is computed by taking the inner product three times:

$$\mathbf{y} = ((\mathbf{M}\ \mathbf{x})\ \mathbf{x})\ \mathbf{x}$$
$$\mathbf{y} = (((\mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1 + ... + \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m)\ \mathbf{x})\mathbf{x})\mathbf{x}$$
$$\mathbf{y} = (((\mathbf{x} \bullet \mathbf{v}_1)\ \mathbf{v}_1 \otimes \mathbf{v}_1 \otimes \mathbf{v}_1 + ... + (\mathbf{x} \bullet \mathbf{v}_1)\ \mathbf{v}_m \otimes \mathbf{v}_m \otimes \mathbf{v}_m)\mathbf{x})\mathbf{x}$$
$$\mathbf{y} = (((\mathbf{x} \bullet \mathbf{v}_1)^2\ \mathbf{v}_1 \mathbf{v}_1^{\mathrm{T}} + ... + (\mathbf{x} \bullet \mathbf{v}_m)^2\ \mathbf{v}_m \mathbf{v}_m^{\mathrm{T}}\ )\ \mathbf{x}$$
$$\mathbf{y} = (\mathbf{x} \bullet \mathbf{v}_1)^3\ \mathbf{v}_1 + ... + (\mathbf{x} \bullet \mathbf{v}_m)^3\ \mathbf{v}_m$$

Is the memory tesseract practical?

MINERVA is equivalent to a distributed memory system implemented as an auto-associative fourth order tensor, or *memory tesseract*. Unfortunately, fourth order tensors are very large. For most applications of MINERVA, the dimensionality n of a vector will be larger than the number of memories m stored in the model. MINERVA, as standardly implemented, is an m x n table, whereas a memory tesseract is an n^4 data structure. A typical MINERVA 2 has $10 \leq n \leq 200$. In general, the number of memories stored is smaller than n and *much* smaller than n^3. For applications where $m < n^3$, the implementation of MINERVA as a table is more efficient. However, for large scale applications where $m \geq n^3$, the fourth order tensor is more efficient.

Alternatively, a holographic approximation to the memory tesseract can be implemented as an n x p data structure for the p of your choice, as is discussed in the next section.

Using holographic vectors rather than tensors

In a holographic vector system, trying to clean-up the echo by iteratively using the echo as a cue to retrieve a new echo is like trying to clean a pair of glasses with an oily cloth: the more you try to clean it, the worse it becomes. Yet Levy and Gayler (2009) have demonstrated that this is possible using a lateral inhibition network implemented as a fully distributed vector architecture. To store a trace v_i in Levy and Gayler's model, the trace is associated with itself twice, then each trace is added to the memory vector m:

$$\mathbf{m} = \mathbf{v}_1 * \mathbf{v}_1 * \mathbf{v}_1 + ... + \mathbf{v}_m * \mathbf{v}_m * \mathbf{v}_m$$

where * is a binding operation. In holographic reduced representations (Plate, 1995), binding uses circular convolution and unbinding uses circular correlation. Given a cue, **x**, we can unbind, denoted by #, to recover an echo:

$$\mathbf{y} = \mathbf{x}\ \#\ (\mathbf{x}\ \#\ \mathbf{m})$$
$$\mathbf{y} = \mathbf{x}\ \#\ (\mathbf{x}\ \#\ (\mathbf{v}_1 * \mathbf{v}_1 * \mathbf{v}_1 + ... + \mathbf{v}_m * \mathbf{v}_m * \mathbf{v}_m))$$

Unbinding is such that given a bound pair, $\mathbf{v}_i * \mathbf{v}_j$,

$$\mathbf{x}\ \#\ \mathbf{v}_i * \mathbf{v}_j = (\mathbf{x} \bullet \mathbf{v}_i)\ \mathbf{v}_j + \mathit{noise}$$

Thus:

$$\mathbf{y} = \mathbf{x}\ \#\ (\mathbf{x}\ \#\ (\mathbf{v}_1 * \mathbf{v}_1 * \mathbf{v}_1 + ... + \mathbf{v}_m * \mathbf{v}_m * \mathbf{v}_m))$$
$$\mathbf{y} = \mathbf{x}\ \#\ ((\mathbf{x} \bullet \mathbf{v}_1)\mathbf{v}_1 * \mathbf{v}_1 + ... + (\mathbf{x} \bullet \mathbf{v}_m)\mathbf{v}_m * \mathbf{v}_m + \mathit{noise})$$
$$\mathbf{y} = (\mathbf{x} \bullet \mathbf{v}_1)^2\ \mathbf{v}_1 + ... + (\mathbf{x} \bullet \mathbf{v}_m)^2\ \mathbf{v}_m + \mathit{noise}$$

However, if we wish to imitate MINERVA 2 as closely as possible (and preserve the sign of the similarity) we need to add another association to Levy and Gayler's model. We compute the memory vector, **m**, and unbind the echo, **y**, as:

$$\mathbf{m} = \mathbf{v}_1 * \mathbf{v}_1 * \mathbf{v}_1 * \mathbf{v}_1 + ... + \mathbf{v}_m * \mathbf{v}_m * \mathbf{v}_m * \mathbf{v}_m$$
$$\mathbf{y} = \mathbf{x}\ \#\ (\mathbf{x}\ \#\ (\mathbf{x}\ \#\ \mathbf{m}))$$
$$\mathbf{y} = \mathbf{x}\ \#\ (\mathbf{x}\ \#\ (\mathbf{x}\ \#\ (\mathbf{v}_1 * \mathbf{v}_1 * \mathbf{v}_1 * \mathbf{v}_1 + ... + \mathbf{v}_m * \mathbf{v}_m * \mathbf{v}_m * \mathbf{v}_m)))$$
$$\mathbf{y} = (\mathbf{x} \bullet \mathbf{v}_1)^3\ \mathbf{v}_1 + ... + (\mathbf{x} \bullet \mathbf{v}_m)^3\ \mathbf{v}_m + \mathit{noise}$$

While this allows the most similar traces to the cue to dominate the echo, the noise term threatens to overwhelm the signal. Because holographic vectors use lossy compression, by iterating, the noise will only grow.

Levy and Gayler solve this problem by using random permutations. Given three random permutation matrices $\mathbf{P}_1$, $\mathbf{P}_2$, $\mathbf{P}_3$, we can permute the vectors as follows:

$$\mathbf{m} = (\mathbf{P}_1\ \mathbf{v}_1) * (\mathbf{P}_2\ \mathbf{v}_1) * (\mathbf{P}_3\ \mathbf{v}_1) * \mathbf{v}_1 + ... + (\mathbf{P}_1\ \mathbf{v}_m) * (\mathbf{P}_2\ \mathbf{v}_m) * (\mathbf{P}_3\ \mathbf{v}_m) * \mathbf{v}_m$$

To recover an echo, we can use inverse permutations:

$$\mathbf{y} = ((\mathbf{P}_1{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_2{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_3{}^{\mathrm{T}}\,\mathbf{x})) \# \mathbf{m}$$

To eliminate the noise, Levy and Gayler use hundreds of such vector memories in parallel, each of which uses its own pair, or in our variant, triple, of permutations. Let p be the number of vector memories being used, then the echo $\mathbf{y}$ is the sum of echoes for each p:

$$\mathbf{y} = \mathbf{y}_1 + \ldots + \mathbf{y}_p$$

where for each $\mathbf{y}_j$, $j = 1 \ldots p$,

$$\mathbf{y}_j = ((\mathbf{P}_{j,1}{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_{j,2}{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_{j,3}{}^{\mathrm{T}}\,\mathbf{x})) \# \mathbf{m}_j$$

such that:

$$\mathbf{y} = ((\mathbf{P}_{1,1}{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_{1,2}{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_{1,3}{}^{\mathrm{T}}\,\mathbf{x})) \# \mathbf{m}_1 + \ldots + ((\mathbf{P}_{p,1}{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_{p,2}{}^{\mathrm{T}}\,\mathbf{x}) * (\mathbf{P}_{p,3}{}^{\mathrm{T}}\,\mathbf{x})) \# \mathbf{m}_p$$

Because each echo $\mathbf{y}_j$ is produced using a different triple of permutations, each echo's noise term will be different. Because the noise in each echo is different, a different part of the signal is preserved in each echo. By taking the sum of all these echoes, we average across them to get a close approximation to the true signal. With large enough p, we minimize the noise sufficiently that we can clean-up the echo by iterating. We can divide the sum of all the echoes, $\mathbf{y}$, by its magnitude to normalize, then use it as the new cue. By varying the dimensionality of the vectors n and the number of memory vectors operating in parallel p, we can manipulate how fast the echo is cleaned up by iterating.

The number of iterations to clean-up the echo is a measure of the time the model takes to perform a memory retrieval. For this reason, the number of iterations is an ideal candidate as a predictor for human reaction time. Being able to map a measure of the model's processing time onto human reaction time would help to provide evidence for *strong equivalence* (Pylyshyn, 1989) between what the model is doing and what the brain is doing.

The number of iterations to clean-up the echo in MINERVA 2 is typically too few to provide sufficient granularity to map well onto human reaction times. If we instead use Levy and Gayler's model, by varying p we can control the average number of iterations that the system takes to clean-up the echo. Using smaller p, the network takes, on average, longer to converge to a steady-state than an iterated MINERVA 2 model, such that the iterations map onto smaller units of human reaction time. Thus, the lateral inhibition network, when coupled with a suitable decision mechanism to end retrieval, can provide more fine-grained reaction time predictions than an iterated MINERVA 2.

The lateral inhibition network is also more tractable than a fourth order tensor for large n (i.e., more space efficient for $p < n^3$ and more time efficient for $p \log n < n^3$).

The semantic tesseract: Scaling MINERVA up

Applying MINERVA to large semantic memory tasks, such as learning the meaning of words (Kwantes, 2005) or learning how to sound-out written words (Kwantes & Mewhort, 1999), requires abandoning a key assumption of MINERVA for purely pragmatic reasons, namely, that repetition of an item produces multiple traces of that item. But if these models were re-implemented using the memory tesseract (or its holographic approximation), we would not have to abandon that key assumption because the tesseract does not grow with the addition of new memories.

BEAGLE (Jones & Mewhort, 2007) is a learning algorithm that models how people abstract the meaning of words from their lifetime language experience. DHSM (Rutledge-Taylor & West, 2008) uses a similar approach to BEAGLE but re-purposes the algorithm as a general memory model. DSHM is a multi-vector memory system, like the standard implementation of MINERVA, but unlike MINERVA, each vector stands for a concept rather than an individual experience. In DSHM, each experience updates the vector for each concept that comprises the experience. An experience is stored in the associations between concept vectors.

In MINERVA, concepts are echoes, artifacts of aggregate retrieval across experiences. We speculate that the vectors in BEAGLE and DSHM are akin to echoes. The memory tesseract now provides a means of re-implementing BEAGLE or DSHM as MINERVA models.

In BEAGLE, an experience is a sentence. To re-implement BEAGLE in the memory tesseract, each time a word is encountered in a sentence, memory is updated with a new episode trace by summing a new fourth order tensor with the memory tensor:

$$\mathbf{M}_t = \mathbf{M}_{t\text{-}1} + \mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i \otimes \mathbf{v}_i$$

Each episode trace is the sum of a word's orthographic information as calculated in Cox et al. (2011) with the word's semantic information gleaned from that particular sentence, as calculated in Jones and Mewhort (2007):

$$\mathbf{v}_i = \mathbf{v}_{i\ orthographic} + \mathbf{v}_{i\ semantic}$$

However, holographic vectors typically have $n \geq 512$. 512^4 is about 1678 times larger than the BEAGLE model. If we assume that BEAGLE and MINERVA use vectors of the same size, given a literate adult with a vocabulary of 80 000 words, the memory tesseract will be larger than BEAGLE for vectors with more than 43 dimensions.

If we were to run a memory tesseract model that learns word meaning from context by processing a large corpus, as BEAGLE does, we would need to keep n as small as possible without sacrificing too much fidelity. We could also use the holographic approximation to the memory tesseract, which is, again, scalable depending on the desired fidelity.

The proposed model escapes being just a less efficient version of BEAGLE because it raises similarity to the exponent of 3. BEAGLE stores all experiences of a particular word in a single vector. This storage is highly lossy and the individual experiences are difficult to recover. MINERVA retains more of each experience, information that can be recovered by iterating to "clean up" the echo.

A memory tesseract implementation of BEAGLE would have another benefit. Individual vectors in BEAGLE are sensitive to first-order (word co-occurrence) and second order (synonymy) associations. Kwantes (2005) found that aggregating across vectors with first-order associations produces an echo with second order associations. Likewise, we suspect that aggregating across BEAGLE's vectors would produce an echo with third order associations. Third-order associations (and higher) may be useful for identifying part of speech. This change to BEAGLE is a benefit of adopting the MINERVA memory retrieval mechanism.

The preceding discussion is intended only as an extended example of how MINERVA, when implemented as a tensor, can be applied to larger scale problems.

Conclusion

We demonstrate that the influential MINERVA 2 (Hintzman, 1984) model is mathematically equivalent to an auto-associative fourth order tensor memory system, or *memory tesseract*. We further show that this is approximately equivalent to a variant of the holographic lateral inhibition network proposed by Levy and Gayler (2009). These demonstrations have three theoretical implications:

(1) Viewing MINERVA 2 and its variants (collectively, MINERVA models) as a fourth order tensor clarifies the relationship between MINERVA and third order, second order (i.e., matrix), and compressed tensor (i.e., holographic vector) memory models, allowing us to move toward a unified understanding of memory.

(2) MINERVA can be scaled up to model long term learning, broadening the scope of tasks to which MINERVA can be applied and allowing for unification with models of semantic learning, such as BEAGLE.

(3) A naïve neural interpretation of MINERVA might suggest that a new neuron is grown for each new experience, corresponding to the addition of another row to MINERVA's memory table. Understood as a memory tesseract, MINERVA is fully distributed across neural connectivity, and memories can be added by changing the connectivity without requiring additional neural resources. Eliasmith's (2013) neural engineering framework provides a system for translating linear algebra computations (e.g., convolution, permutation) into spiking neuron models. The memory tesseract, or its holographic approximation, could be easily implemented as a realistic neural model using Eliasmith's framework.

References

Cox, G. E., Kachergis, G., Recchia, G., & Jones, M. N. (2011). Towards a Scalable Holographic Representation of Word Form. *Behavior Research Methods, 43*, 602-615.

Eich, J. M. (1982). A composite holographic associative recall model. *Psychological Review, 89*, 627–661.

Eliasmith, C. (2013). *How to build a brain: A neural architecture for biological cognition*. Oxford University Press.

Franklin, D. R. J., & Mewhort, D. J. K. (2013). Control Processes in Free Recall. In R. West & T. Stewart (eds.), *12th International Conference on Cognitive Modeling* (pp. 47-52), Ottawa, CA.

Goldinger, S. D. (1998). Echoes of echoes? An episodic theory of lexical access. *Psychological Review, 105*, 251-279.

Hintzman, D. L. (1984). MINERVA 2: A simulation model of human memory. *Behavior Research Methods, Instruments, and Computers, 16*, 96-101.

Hintzman, D. L. (1986). "Schema abstraction" in multiple-trace memory models. *Psychological Review, 93*, 441-428.

Hintzman, D. L. (1990). Human learning and memory: Connections and dissociations. *Annual Review of Psychology, 41*, 109-139.

Howard, M. W., & Kahana, M. J. (2002). A distributed representation of temporal context. *Journal of Mathematical Psychology, 46*, 269-299.

Humphreys, M. S., Bain, J. D., & Pike, R. (1989). Different ways to cue a coherent memory system: A theory for episodic, semantic, and procedural tasks. *Psychological Review, 96*, 208-233.

Humphreys, M. S., Pike, R., Bain, J. D., & Tehan, G. (1989). Global matching: A comparison of the SAM, Minerva II, Matrix, and TODAM models. *Journal of Mathematical Psychology, 33,* 36-67.

Jamieson, R. K., Crump, M. J. C., & Hannah, S. D. (2012). An instance theory of associative learning. *Learning & Behavior, 40*, 61-82.

Jamieson, R. K., & Mewhort, D. J. K. (2011). Grammaticality is inferred from global similarity: A reply to Kinder (2010). *The Quarterly Journal of Experimental Psychology, 64*, 209-216.

Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and order information in a composite holographic lexicon. *Psychological Review, 114*, 1-37.

Kelly, M. A., Blostein, D., & Mewhort, D. J. K. (2013). Encoding structure in holographic reduced representations. *Canadian Journal of Experimental Psychology, 67*, 79-93.

Kwantes, P. J. (2005). Using context to build semantics. *Psychonomic Bulletin & Review, 12,* 703-710.

Kwantes, P. J., & Mewhort, D. J. K. (1999). Modeling lexical decision and word naming as a retrieval process. *Canadian Journal of Experimental Psychology, 53*, 306-315.

Levy, S. D., & Gayler, R. W. (2009). "Lateral inhibition" in a fully distributed connectionist architecture. In Howes, A., Peebles, D. & Cooper, R. (Eds.), *9th International Conference on Cognitive Modeling* (pp.318-323). Manchester, UK.

Murdock, B. B. (1993). TODAM2: a model for the storage and retrieval of item, associative and serial-order information. *Psychological Review, 100*, 183–203.

Rutledge-Taylor, M. F., & West R. L. (2008). Modeling the fan-effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *30th Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC.

Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks, 6*, 623–641.

Pylyshyn, Z. (1989). Computing in Cognitive Science. In Posner, M. (Ed.) *Foundations of Cognitive Science*. Cambridge: MIT Press.

Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence, 46*, 159-216.

Thomas, R. P., Dougherty, M. R., Sprenger, A. M., & Harbison, J. I. (2008). Diagnostic hypothesis generation and human judgment. *Psychological Review, 115*, 155-185.

Tulving, E., & Watkins, M.J. (1973). Continuity between recall and recognition. *American Journal of Psychology, 86*, 739−748.