A Neural Architecture Based on Hadamard Designs

Alan Herbert*

Department of Pharmacology and Experimental Therapeutics School of Medicine Boston University 715 Albany Street, Boston MA 02118, USA

Abstract: We describe a simple Hadamard design for neural architecture with an equal number of input and output elements that is both error-tolerant and robust to missing information. The design provides a basis for calculation using a classification scheme based on the Chinese remainder theorem, producing an abstract representation of the physical world. The underlying co-prime arrays can be generated in a simple manner biologically and can evolve into more complex designs. The approach differs from previously described neural network constructions in that all connectivity is specified by design, with each correctly wired array producing a single output for each subset of inputs. The wiring is consistent with the "On-Off" schema observed for different senses because only about half the inputs can be active at any one time. The arrays can be tuned through by varying the number of simultaneous inputs required for activation within a range specified by the array size. The architecture is scalable.

Keywords: Hadamard, On-Off array, Abstraction, Brain, Neural Computation.

INTRODUCTION

Understanding how the brain works is challenging since not enough information exists in the genome to fully specify all the neural connections present [1]. Even if there were, a wiring diagram of a circuit by itself does not reveal how the circuitry functions. Approaches to reverse engineering the brain have been so far unsuccessful; huge assemblies of micro-processors performing huge numbers of calculations per second are unable to pass a general Turing test of equivalence [2]. Amazingly, the brain has evolved to perform complex calculation, classification, abstraction and adaptation using only a simple set of parts arranged with monotonous regularity [3,4]. It self assembles, forming most of its connections with itself with as little as 3-5% neurons receiving information directly from the environment [1,5]. Even before development is complete and within hours of formation, the brains of some vertebrates are capable of coordinating complex behavioral responses to external stimuli. Currently we do not understand much about these remarkable properties.

Here we propose an architecture that uses Hadamard designs as the basic building block for building neural arrays in the brain [6]. These designs are self-generating; connections are created and pruned during development to ensure each array has a single output regardless of how many inputs it receives. A major feature of these designs is that the architecture is fixed prior to training, with a particular set of inputs always producing only a single output from the array (Fig. **1**). As described below, such arrays are and error-tolerant and robust to missing information.

Since Hadamard designs are based on prime numbers, it is possible to select sets of Hadamard designs where members of the set have a co-prime number of elements. Such sets provide a basis for a residue number scheme useful both for indexing inputs and for computation [8]. The neural architecture that results can be viewed as the Cartesian product of two different types of graph: one type that specifies each array in the set and the other that connects them in space. Each of these graphs can change independently of the other, generating many different configurations for selection and providing the basis for high level abstractions. The approach taken here has both similarities and differences to previous proposals (recently reviewed in [9]). It is highly structured and produces a single output by design without the need for the training that is required by unsupervised models to generate self-organizing maps [10]. As with supervised learning, the input can be reconstructed from the output. However, with Hadamard arrays, all possible inputs capable of generating that output are mapped, rather than those inputs that were actually present [11].

DIFFERENCES TO PREVIOUS MODELS USING HADAMARD ARRAYS

Hadamard arrays have found wide application in signal transmission, coding and image analysis in many fields [7]. These schemes differ from the one proposed here in the way mapping of inputs to outputs is performed. In one previously described method, each input signal is transformed to produce an output orthogonal to all other outputs from that matrix. The outputs are thus uncorrelated [7]. This outcome is achieved by using a different row from a Hadamard array to encode each output. This approach maintains a one to one mapping between inputs and outputs and allows the outputs to be combined without loss of information. Each input vector can be recovered from the combined output by employing the same row of the Hadamard matrix used for encoding. This method is used for communicating multiple unrelated signals over a single channel, such as in CDMA phones and has been applied for pattern matching as a model for mem-

*Address correspondence to these authors at the Department of Pharmacology and Experimental Therapeutics School of Medicine Boston University 715 Albany Street, Boston MA 02118, USA; Tel: 617.414.1254; Fax: 617.638.5254; E-mail: aherbert@bu.edu

1874-0820/12

ory in the brain [18]. In another previously described method, all rows of the Hadamard matrix are used in a scheme where the length of input and output vectors is the same but the mapping is many to many [12]. By discarding outputs falling below a particular threshold, data can be compressed and the effects of noise reduced. The input vector can be regenerated with the noise removed by using an inverse Hadamard transform. The approach described here is different. For each Hadamard array we use, the length of the output vector is one regardless of the array size. Thus the mapping of inputs to output is many to one. In this scheme, it is not always possible to reconstruct the exact input signal from the output as different input combinations will yield the same output (Fig. **1**). Nevertheless, as we describe below, the scheme underlies a robust classification system and enables functions such as calculation and abstraction.

ERROR-TOLERANT ARRAYS

The simple one-to-one mapping of inputs to outputs in a primitive array is vulnerable to loss of signal and to signals generated by noise. Error-tolerant arrays avoid these problems. An example of one is shown in Fig. (**1a**). Here, "v"=11 inputs to the upper edge of an array are mapped to 11 outputs from the right-hand edge to produce a square array [13]. Only one output is active at a time, so the mapping is many to one. This property is essential to the error-tolerant nature of these arrays: while each row of the array at most receives "k"=6 of the inputs (as indicated by a "1" in the connectivity matrix), only four of these need be active to specify output uniquely from that row (giving the minimum number "t" of connections necessary to produce a unique output from the array). When two of the 6 inputs are missing, there are thus 15 input combinations ($^6C_4$) that specify the correct output. The system not only tolerates the lost information but also allows its recovery through imputation of the missing data. The array design is also resistant to noise. A spurious output from an otherwise quiescent array requires four false inputs while two errors are needed to generate an incorrect output from an active array.

Inherent in the operation of these arrays is the delivery of inputs to an output neuron when it is capable of firing. Whether an active input is temporally encoded or rate encoded, it will need to arrive in the appropriate time-window. Tuning of an output neuron to a particular time-scale, or to a specific response frequency, is possible using feed-forward or top-down mechanisms to alter the responsiveness of the output neuron [14], even in the presence of noise [15].

The C(v=11,k=6,t=4) system in Fig. (**1a**) is an example of cyclic 2-Hadamard difference design. Here ν=number of outputs from the array, k=total number of inputs per row of the array and t=minimum number of inputs to produce a unique output from a particular row. The layout of the first row is sufficient to specify the full array. The pattern of connections shifts rightwards by one position at the start of each subsequent row, until the design repeats after the 11^{th} permutation. More examples of sets in this series are listed in Table **1** and displayed in Supplementary Fig. (**1**), along with their mathematical properties. The modular nature of these sets provides a basis for the self-generating, self-organizing number systems discussed here.

THE PROPERTIES OF SQUARE ARRAYS

Square arrays are a plausible mechanism for error-tolerant operation of neuronal arrays within the brain due to their inherent robustness and tolerance of noise. They have other very important properties. First, only those arrays with the proper number of connections will provide *a single output for each set of inputs*; this provides a test to check that an array is wired correctly. When fully configured, about half the inputs (i.e. k=(v±1)/2) will be connected to an output and

Table 1. Square Sets C (v,k,t)

a						**b**				
v	*k*	*t*	*b*	*r*		*v*	*k'*	*t'*	*b*	*r'*
*+ 7	3	2	7	3		++ 7	4	3	7	4
+11	5	3	11	10		++11	6	4	11	15
*13	4	2	13	6						
+15	7	4	15	35		++15	8	5	15	56
+19	9	5	19	126		++19	10	6	19	210
*21	5	2	21	10						
+23	11	6	23	462		++23	12	7	23	792
*31	6	2	31	15						
+31	15	7	31	6435		++31	16	8	31	12870

Square sets in which the number of points *v* equals the number of blocks *b of size k,* with *t* points contained in exactly one block and with *r* sets of *t* points in each block (redundancy). Sets C(13,9,7) and C(21,16,13) are not shown or discussed in the text. Steiner sets are indicated by an asterisk. Square sets corresponding to cyclic set difference Hadamard designs*t,* k=(2*t* -1) or as k=(v-1)/2, v=(4*t* -1) are marked with a '+' in panel a and those corresponding to its complement (*t* +1), k=2*t* or ask=(v+1)/2, v=(4*t* -1) are shown with a '++' in panel b. These sets can also be expressed as v=4n+3, n∈ {1,2,3…}, subsets of which includes the Gaussian primes, the product of twin primes and the series 2^m-1.

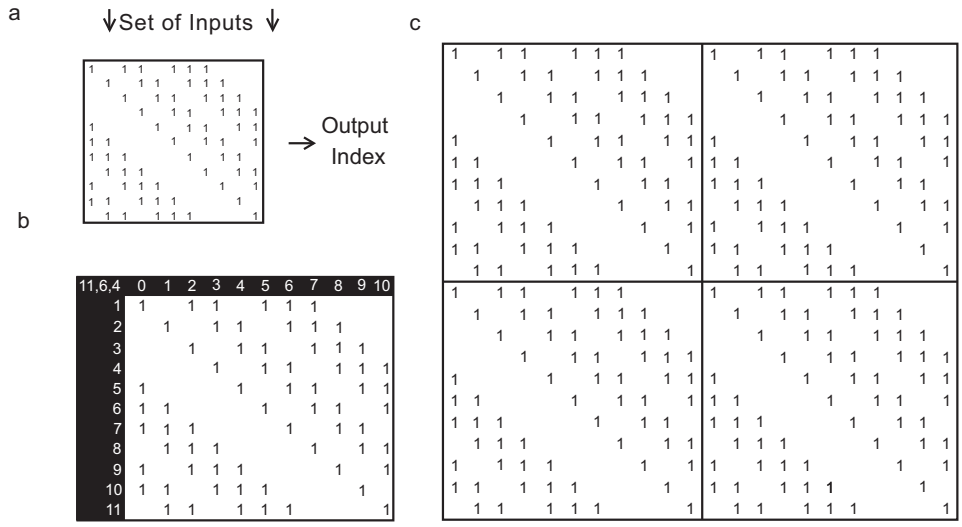

Fig. (1). Square Arrays with tolerance to noise and missing information. Biologically, this array is interpreted as a pattern of connections (synapses or gap junctions) between input and output neurons. (**a**) In the designs used here, multiple inputs to the upper edge of an array are mapped to a single output from the right-hand edge i.e. the mapping is many to one. This property is essential to the error-tolerant nature of these arrays as shown in the rest of the figure. Only half of the inputs at most can be active at any one time and is consistent with the observed "On-Off" arrangements observed for sensory inputs. (**b**) The C(11,6,4) array is constructed using by cyclic permutation of the vector (1,0,1,1,0,1,1,1,0,0,0) in the binary representation of the array and (1,-1,1,1,-1,1,1,1,-1,-1,-1) in the Hadamard and bipolar representations (i.e. the matrix is orthogonal) or (1,3,4,6,7,8) to index the connections in the first row of the array. The number of possible inputs (given by the column labels) is equal to the number of possible outputs (given by the row labels). Connections between an input and an output are indicated by a "1". Here eleven input elements are physically mapped onto eleven output elements in an error-tolerant fashion. Output from a particular rowonly occurs when at least four of the possible six inputs are active. When two of the six inputs are missing, there remain fifteen ways of using any four inputs to uniquely specify the output from that row, allowing the array to identify objects even when complete information is missing. The array is also resistant to noise as discussed in the text. Other mappings of inputs to outputs are possible using this array design. For example, 11 inputs can be mapped to just 5 outputs as shown by rows 1 to 5. By connecting the vertical edges, arrays could be formed as a cylinder [3]. (**c**)The 11 by 11 array is replicated to produce a sheet of like arrays, each of which can act in parallel, permitting scaling of the design.

about a quarter (either t=(v+1)/4 or t=(v+5)/4 depending on the design) will be sufficient to specify that output uniquely when active. Second, square arrays do not require complete wiring of all inputs for them to be functional. They are very tolerant of wiring variations that arise during development or to disruption by degenerative disease. Partial wiring only reduces the number of input combinations that elicit a particular output. With minimal wiring, a single input set of size t is sufficient to specify a particular response. Third, the arrays can be tuned by varying the number of inputs necessary for the output neuron to fire from k to t. This change in this threshold may be temporary or made permanent through learning. A higher threshold (i.e. more active input neurons required to produce a single output) increases the specificity of a response as more information is used. This reduces sensitivity of the array to noise but renders it less tolerant to missing data. At one extreme, only the full set of k inputs to a row would produce an output from a row while at the other extreme, t inputs would be sufficient to ensure a unique response from the array. The value d=k-t specifies the dynamic range of the array and increases with the array size v. Fourth, if an object is identified based on partial information, then the object's presence can be confirmed by scanning inputs for other subsets specifying the same output. This process could work on the local scale by polling of the silent synapses on the output neuron for input (e.g. by altering the local conductance properties) and thereby increase the firing probability of the output neuron. It could involve higher level functions, such as anticipation, that scan the environment for the missing information [14]. Fifth, the smallest 7 sets in the Hadamard series have a pairwise co-prime number of elements (i.e. the only common factor for of the set v= {7, 11, 13, 15, 19, 23, 31} is 1). Non-prime members of this series such as the set v= {15, 35, 143, 323 and 899} can be constructed using twin primes (3,5), (5,7), (11,13), (17,19) and (29, 31) (Supplementary Methods). This arrangement provides a scheme that creates an indexer for an input based on the output from a set of co-prime arrays, each of different size v (Fig. **2**). This approach is justified by the Chinese remainder theorem (CRT)(Supplementary Material). The CRT states that the maximum number of unique indexes generated by such an arrangement is simply the joint product of the array sizes (i.e. $v_1 \times v_2 \times v_3 \ldots$). Sixth, this classification scheme can be scaled as organisms evolve either by increasing the number of co-prime arrays used for indexing or by selecting a set that has larger sized arrays in it. Seventh, error-tolerant arrays of any size can be easily generated using a strategy based on primitive arrays as described below.

INDEXING LARGE SETS USING SQUARE ARRAYS

Applying the CRT to the v= {7, 11, 13, 15, 19, 23, 31} set of square arrays provides over 15 million unique indexers (7×11×13×15×19× 23×31), each specified by a vector length

6. Use of twin-prime arrays produces many more (~1.5 billion for 35×143×323×899) twin arrays with a vector length 4 based on (3,5), (5,7), (11,13), (17,19) and (29,31) twin prime constructions. We propose that such error-tolerant schemes are used to index inputs to the brain. We refer to these input arrays as primary mod arrays. In Fig. **2**, input is passed through a set of arrays that differ in size. Only the final array for each modulus is shown, with the number of intermediary arrays between an input array and the final mod array representing a multiple of the modulus v. For proper indexing, the inputs for each attribute must eventually be passed through single array of each modulus. The output of the last array then provides the index for that modulus. A set of indices from the different sized arrays is referred to as an indexer. The indexer can be matched with known objects or combined with output from other arrays, generating new set of indices. For example, the outputs of 7 mod C (7,4,3) arrays could be used as inputs for a higher level, error-tolerant C (7,4,3) array. The system can be scaled easily as the number of inputs increase by adding more primary and secondary arrays [3]. The result is a hierarchical system [16,17] based on co-prime arrays for processing information. This scheme remains robust to missing information since it does not require a full set of inputs at any level to correctly classify an object. It allows for the accurate identification of a particular object or its class attributes under a variety of circumstances. It is also possible that the indexer generated will be novel as it has not been previously experienced as part of the physical world. Even then, a subset of its indices may overlap with known objects. The partial match may be sufficient to elicit an appropriate response or provide the correct classification of an object type.

CALCULATING WITH MAP ARRAYS

The organization of square arrays provides a representation of inputs suitable for modular arithmetic. For example, dual arrays based on the joint use of mod 7 and mod 11 wiring yield a simple 7x11 representation of inputs with 77 unique values. Simple addition is then possible: the residuals of each number mod 7 are added, as are the residuals of each number mod 11 [8]. The new values mod 7 and mod 11 uniquely specify the sum of the numbers when this is less than 77. Unlike the base arithmetic we are all familiar with, no carry operation is required to perform the calculation [8]. The calculations can be performed simply using the output of one calculation to index the output of the next. For example, in Fig. **3**, three different representations of this process are given of the process. In Fig. (**3a**), the indexing of a set of arrays is shown in grey boxes. The output of a calculation resets the index of the next array. This recalibration is illustrated by the open-face arrows pointing to the new alignment shown in the white text box. The calculation is performed by aligning the new index with the old index to give the result that is highlighted in red, as shown by the solid back arrow. The index is then reset and the process repeated to give the next result. In Fig. (**3b**), the process is illustrated with a sheet of arrays similar to that drawn in Fig. (**1b**) and using the appropriate Hadamard coding to represent the numbers. In Fig. (**3c**), the process is presented using black boxes. For multiplication, the addition would be repeated the appropriate number of times as specified by the multiplier, for each of the mod arrays used to index an input. The operations can be performed in parallel since each mod array acts independently of the others [18]. The limitations of such residue number systems are well known: the magnitude of a number

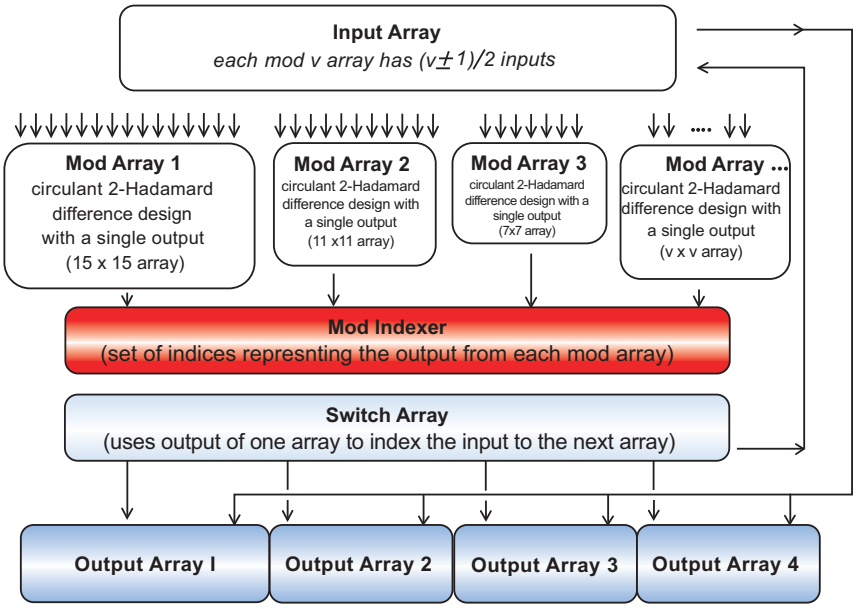

Fig. (2). Processing through different mod arrays produces an indexer for the input. The switch array uses the set of indices generated by the Mod arrays to index the next input set to the next array in the graph. The output array may be a many to one Mod array, such as those used in calculation as described in the text, or a one to many Mod array used in recall or in initiating an action. Some of the indexers generated by Mod Arrays will be abstract as they have not been previously experienced as part of the physical word. Even when the indexer is novel, a subset of its indices may overlap with indices from a previously experienced object leading to an approximate match for the novel input set and an appropriate response. In some cases, the match will lead to a misidentification and the wrong response. The feed forward and feedback loops illustrated with the long arrows provide mechanisms to check input against expectation and to anticipate possible outputs.

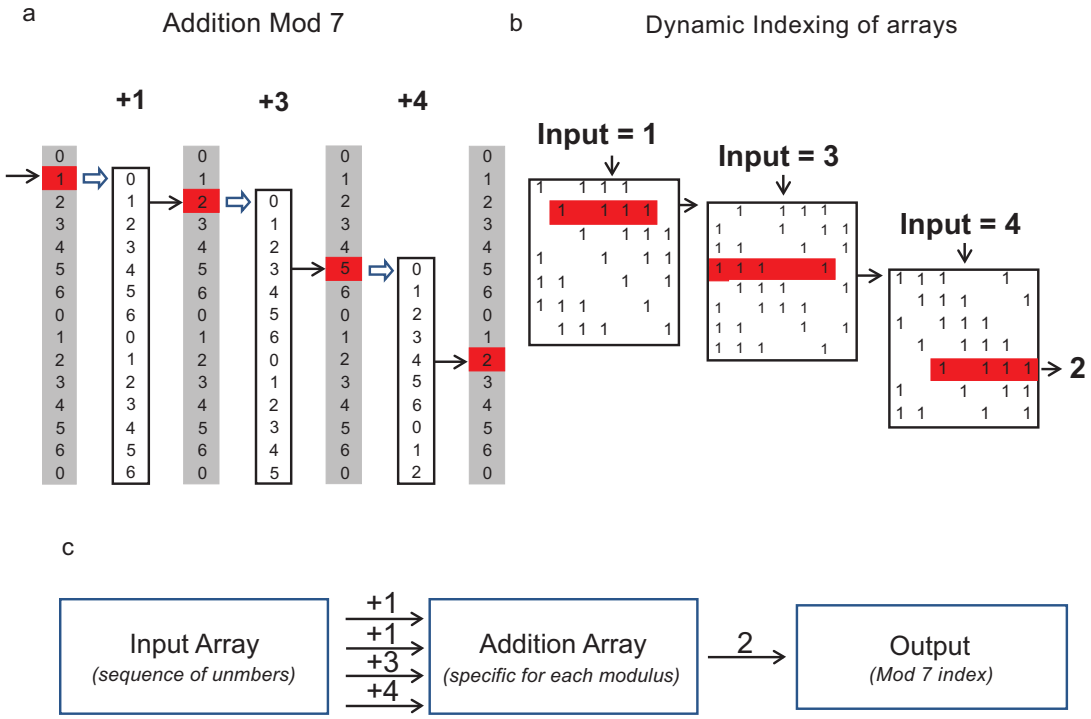

Fig. (3). Three representations of addition using Mod 7 Arrays. (**a**). The figure shows addition of 1+1+3+4 = 2 mod 7 from left to right. The grey boxes contain the index for the arrays used in the calculation. The result from one calculation (in red box) resets the indexing of the next array as shown with the open-face arrows; the new alignment is given in the white boxes. The next addition is performed by aligning the new index with the old index at the position specified by the input. This is shown by the solid back arrow. The new result is highlighted in red. The index is then reset and the process repeated to give the next result. (**b**) The process is illustrated with sheets of arrays similar to that drawn in Fig 1b and using the appropriate Hadamard coding to represent the numbers. (**c**) The process is presented using black boxes with the sequence of numbers to be added specified by the input array.

is difficult to estimate from the vector of indices and the product of multiplication must be less than the maximum specified by the CRT. In biological systems, where both scale and magnitude of the result are constrained by physical realities, the number of primary mod arrays needed to fully index the initial input may be sufficient to give estimates of size. Non-linear scales may also be used e. g., a logarithmic scale or one based on primitive roots where exponents are added rather than whole numbers [8].

ABSTRACTION

Calculations based on mod arrays can generate output vectors with values not experienced as part of the physical world. This property naturally leads to the concept of abstract indices that can be classified based on their similarities to and differences with known indices [10]. Each subset of identical indices would identify a class of objects. Use of different sets of shared indices would allow an object to be classified in different ways, or one object to be mistaken for another. Categorization is facilitated by the evolution of higher order arrays that combine multiple primary sensory inputs, providing additional dimensions for comparison. While organisms could evolve using arrays based on just one modulus, the number of different indexers would be limited by array size; additional moduli would by the CRT greatly increase the complexity of representations with each array type offering a different way to index attributes.

TIMING

For sequential calculation, events require isolation in time and space to prevent overwriting of steps initiated before or after the current one. Synchronization at a local level can be achieved by using the output of one step to regulate the output of the next step. Local oscillatory neurons can be used to gatearray output with more distant connections introducing bias through feed-forward and top-down signaling mechanisms.

RECALL, LEARNING AND SWITCH ARRAYS

The generation of a brain from reiterated design elements produces array fields (Fig. **1b**). The boundaries between arrays may be static and fully specified during development, or dynamic and changeable. With static arrays, there is precise matching of inputs to outputs. With dynamic arrays, calibration is necessary to correctly define each array boundary, index outputs and set the number of inputs necessary to cause the output neuron to fire (Figs. **1b** and **3b**). For example, the 7 × 7 set of inputs and outputs selected may arise anywhere in a field of C (7,4,3) array elements shown in Fig. (**1c**). The location of this array then limits the placement of other arrays in that field. The layout could be quite variable with the optimal mapping of inputs to outputs dependent on stimuli from external sources, network processes such as anticipation, or sensory scanning mechanisms such as eye microsaccades. Over time, boundaries and thresholds may

become fixed, and the arrays static through experiential learning and the fine-tuning of overall performance [10].

So far, the discussion has focused only on arrays that map many inputs to one output. It is also possible to reverse the wiring of these arrays so a single input produces multiple outputs. With a one-to-many array, an index derived from a particular modulus could be expanded to recall the set of attributes that the index encodes. In other cases, the index could be used to generate a set of outputs that prime a sensory field for particular inputs or others that initiate a set of actions. The outcome will depend entirely on how the difference Hadamard designs are connected in space. We can thus define two types of two graphs: one that specifies the set of Hadamard designs and the other that specifies their connections. The Cartesian product of the two types of graph defines the architecture of the brain. While the graph of Hadamard designs is relatively fixed, the graph describing the set of connections is likely to change with time as a result of learning.

We refer to the building blocks for the graph that specifies connections between Hadamard arrays as switch arrays (Fig. **2**). One type of switch array is described in the section on calculation. The addition array works by changing the indexing of a Hadamard array to ensure the result is mapped correctly to the next Hadamard array (Fig. **3** and Fig. **4**). More elaborate switch arrays for direct multiplication would activate only those rows in a field that are separated by a count equal to the multiplicand. For example, only every third row would be used when multiplication is by 3. The input would specify which of these selected rows would be used to calculate the result. Even more complex switch arrays would be needed for arithmetic that involves carry operations because multiple switch arrays are necessary to index each exponent of the base to correctly map the result of the calculation. Similarly, switch arrays would also underlie the recall of associations between objects by using the output of one array to index the elements of another. The schema presented here involving the Cartesian product of two different graphs is similar to approaches based on Reduced Boltzmann Machines where weights for a lower level (corresponding here to the Hadamard designs) are relatively fixed and those to an upper level (corresponding here to the switch arrays) are varied [9].

VARIATIONS IN ARRAY DESIGN

The complete square arrays discussed here are by way of example. Variations of these designs may be used to construct the switching arrays discussed in the previous section. For example, the 19 inputs of the C(19,10,6) system (Supplementary Fig. **1**) can be mapped to only 10 outputs to produce a mod 10 indexer. The focus here has been on rows and columns. Wiring along the diagonal of a square array is also

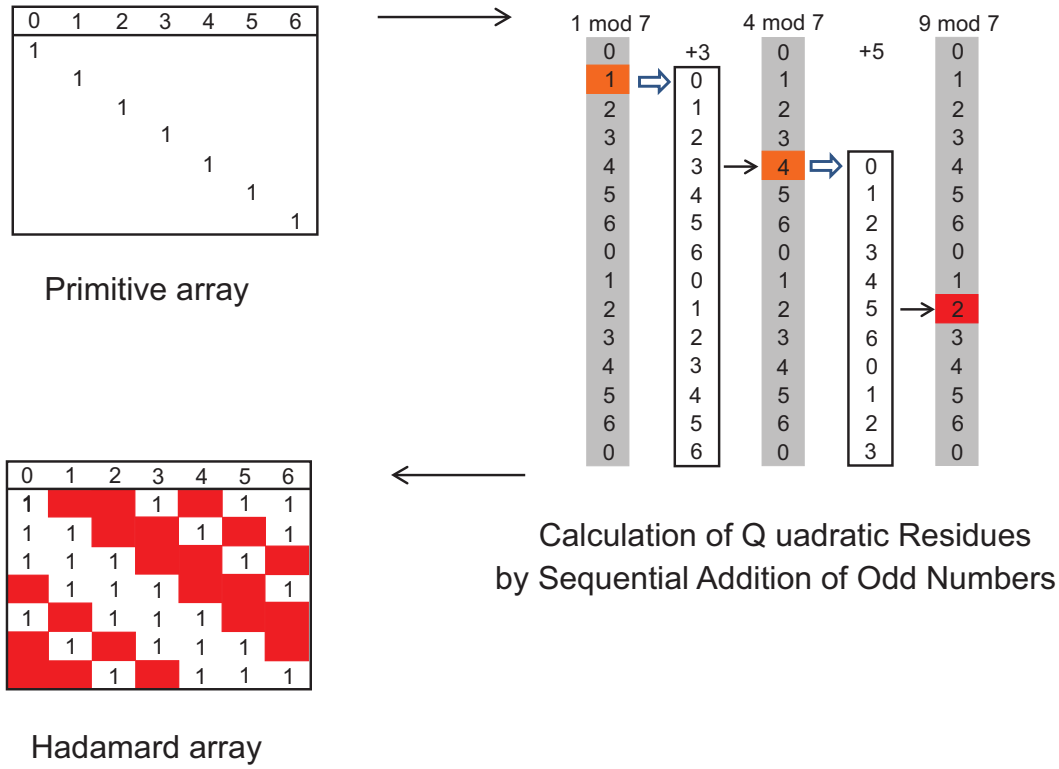

Fig. (4). Generation of a Mod 7 Hadamard Array from a Mod 7 Primitive array. (**a**) A primitive array with a single connection between an input and an output per row. (**b**) The array is used to sequentially add odd numbers to generate quadratic residues to specify the wiring of a Hadamard array. The panel is labeled as in Fig. 3a with the result of each addition mod 7 shown in red. (**c**) The residues obtained from the calculation (specifying positions 1,2 and 4 in the vector for the first row) are used to prune connections in a sheet of cells at those positions to obtain a correctly wired Hadamard matrix. For each row, the pruning shifts one place to the right to generate the complete array. In general, this approach does not require addition of any number greater than v, nor the use of any array greater than v. The first row is specified in (v±1)/2 steps and will have (v±1)/2 elements when v is prime.

possible so that all output elements are connected to the same input. Connections from a single neuron could then gate the output from an array. Other designs for one to one mapping of inputs to outputs by cyclic permutation exist; they do not generate square arrays [19]. The primary advantage of square sets is their robust performance and their utility in calculation.

EVOLUTION OF SQUARE ARRAYS

How could these systems evolve? The ability of these arrays to tolerate noise and missing information would have offered a selective advantage. Larger arrays would involve fewer steps to process a particular set of inputs, allowing faster responses that would favor their selection over time. Lower order sets may have initially arisen by chance (e.g. $^{11}C_6$ = 462 possible combinations). This is less likely for higher order arrays where the search space is large ($^{23}C_{12}$ =1,352,078 possibilities). An alternative solution is to directly construct the arrays. A simple process for doing so draws on the fact that this class of Hadamard designs is fully specified by the quadratic residues (i.e. the residue remaining when the square of a number is taken mod v). These residues can becalculated by the sequential addition of all odd numbers less than v using a mod v array to do the addition (Fig. **4**). This requires k=(v-1)/2 steps to enumerate all the quadratic residues using any array that will perform mod v addition, even a primitive one where each row has only one element (Fig **4**). The quadratic residues could specify either the connections to be pruned (as in the Hadamard Array in Fig. **4**) or the connections to be made, generating either of the two types of array shown in Table **1** for each modulus. Once made, a Hadamard array can be checked for correct wiring by testing the output from the array; for a newly formed row with k elements, only the output neuron from that row, but not from previously generated rows, should be active when the k inputs are active. Connections to the new row can be edited as necessary until these conditions are met. The method of construction ensures correct wiring of the array in the absence of external stimuli. Once the array is fully wired, the minimum size input set t that produces only a single output from the array can be determined. This process establishes the minimum wiring necessary for the array to be functional. This process can be extended to form very large square arrays. Even bigger, giant arrays could also be generated using twin primes where it is only necessary to calculate quadratic residues for each prime rather than for their product. Thus the results for v=71 and v+2=73 can be combined to yield the connectivity matrix rather than it being necessary to add all the odd numbers less than v×(v+2)=5183. Giant arrays, regardless of how they are formed, offer a selective advantage as they allow data to be processed in larger chunks and thus faster, favoring their rapid evolution.

CODING OF INPUTS INTO SQUARE ARRAYS

An interesting feature of the proposed arrays is that by their design, no more than about half of the inputs can be active at any one time. This is true even in the absence of learning. This outcome could be accomplished in biological systems by the encoding of primary sensory input using "On" and "Off" cells to encode the presence or absence of an input. The use of "On" and "Off" cells has also been described in visual [20] and auditory sensory systems [21]. There are also an equal number of positive and negatively correlated afferents in tactile responses to curvature [22].

The "On-Off" coding scheme is easily applied to square arrays. An "On" cell would be paired with an "Off" cell, with each pair being either (1,0), meaning the "On" cell is active and the "Off" cell quiescentand as (0,1) for the reverse case when there is no stimulus. The $2^{v/2}$ possible input combinations to the array each will be mapped to one of the v possible output indices. A simple algorithm to do this finds the minimal Hamming distance between the input vector and an output row i.e. the row to which the input vector is most similar. An index for the input is assigned when there is a best match to only one row. When the best match is with multiple rows, a single element in the input vector is flipped to either "On" or "Off" in a reiterative fashion until a best match to a single output row is found. The algorithm flips elements in a pre-specified order so that it is deterministic, ensuring that a particular input vector always receives the same index, something that could not be guaranteed if flipping was randomly performed. With this approach, it is not necessary to adjust connection weights as is done with other neural network models. This scheme has the interesting mathematical property of producing the same index for the "negative" image of the input when the requirement is changed to maximize rather than minimize the Hamming distance.

PREDICTIVE FEATURES OF THE MODEL

By their very nature, error-correcting arrays create challenges for the experimenter analyzing brain function; mapping of inputs to outputs may be complicated since a particular output may be triggered by many different combinations of active inputs. This result may create the impression that the system is degenerate rather than precisely wired [23]. Further, a particular set of inputs need not be spatially contiguous in the mature brain; the inputs could be sourced from different parts of the brain during development [24, 25]. This may make localization of an adult function to just one area of the brain difficult , especially when inputs from other regions compensate for the loss of another [26]. There may also be a multitude of connections to an output neuron, especially in the case of giant arrays, with apparent complexity obscuring their underlying mode of operation.

The square array format is also consistent with the sparse coding of sensory input because there is only one output per array despite a complex pattern of inputs [14, 27]. Classification using multiple co-prime arrays provides each primary sensory input an inherent numerical representation regardless of modality. A prediction is that the average separation between adjacent active output neurons in an array field is equal to the array size, while the number of active input neurons per array would be about half the array size (since 2k=v±1). Another is that the number of external inputs could be used to estimate the array sizes. For example it is estimated that ~24 thalamic neurons map to a layer IVc neuron of the cat visual cortex suggest an array size of twice that number is required for this mapping [5, 28]. A region where array size could also be determined is the medial entorhinal

cortex where grid cell firing rate provides a positional map ofan animal's location in space [29]. Giant arrays would be predicted to facilitate classification; performance would be enhanced as fewer steps are required for indexing large numbers of inputs. The regional architecture would reflect the number of connections between input and output neurons. Variations in array size may partly explain known differences in connectivity between rostral and caudal regions of the human brain [30]. Indexing uses a series of hierarchical arrays of size v, with v input and v output neurons. The number of neurons required imposes an upper limit on array size v since there are only so many neurons in the brain: for v=1019, at least 10^6 neurons of the 10^{11} in the human brain would be necessary for a single array, each connecting with about 500 other neurons in that array. In general, the number of connections in arrays of dimension v x v would scale by $(v^2 \pm v)/2$. The total number of neurons required to index a particular set of inputs also would vary depending on the way mapping was performed; a greater number would be needed if all arrays are statically defined as opposed to designs in which array boundaries are set dynamically using switch arrays. Connection of input neurons to multiple arrays corresponding to moduli of different sizes is implicit in the CRT design and is consistent with studies reporting that numeration is both modular in nature and spread over a number of brain regions [24, 25, 31].

The following questions arise. Is there a class of genes that determine the preferred size of arrays, or do these vary by individual? Are there mutations in these genes that reduce the variety of arrays possible, impairing cognition and learning by reducing the total number of indices available for categorization? Do learning difficulties arise because array boundaries are either poorly or wrongly set? How is the number of inputs required to produce a unique output specified, given that it may vary in the range of t to k? If the number is always equal to k, does that lock in responses to a highly specific set of inputs resulting in stereotypic behaviors such as those present in autism? If too low and equal to t, does that alter attention span such as in hyperactivity disorders due to increased sensitivity to neural noise? Do partial arrays exist, either as part of normal development or due to disease-associated neuronal death and are they characterized by partial sensory deficits where some, but not all, input combinations elicit a response?

SUMMARY

A set of reiterated, easily generated structured arrays provides a means through which error-tolerant classification systems can evolve biologically, enabling simple arithmetic with abstraction as an emergent property. Given the ease with which these arrays can be constructed from primitive arrays, it may not be surprising that many organisms may have evolved a number sense and the ability to perform simple quantification [24].

ACKNOWLEDGEMENTS

Thanks to Elizabeth B. Riley for a critical reading and Simon Kasif AND Richard Beigel for helpful discussions. Support for this work comes from a Boston University Center for Neuroscience grant and from NIDDK grant DK077120.

CONFLICT OF INTEREST

None declared.

SUPPLEMENTARY MATERIAL

Supplementary material is available on the publishers Web site along with the published article.

REFERENCES

[1] Binzegger T, Douglas RJ, Martin KA. Topology and dynamics of the canonical circuit of cat V1. Neural Netw 2009; 22(8): 1071-8.
[2] Markram H. The blue brain project. Nat Rev Neurosci 2006; 7(2): 153-60.
[3] Rakic P. The radial edifice of cortical architecture: from neuronal silhouettes to genetic engineering. Brain Res Rev 2007; 55(2): 204-19.
[4] Swanson LW. Quest for the basic plan of nervous system circuitry. Brain Res Rev 2007; 55(2): 356-72.
[5] Peters A. Examining neocortical circuits: some background and facts. J Neurocytol 2002; 31(3-5): 183-93.
[6] Sima J, Orponen P. General-purpose computation with neural networks: a survey of complexity theoretic results. Neural Comput 2003; 15(12): 2727-78.
[7] Horadam KJ. Hadamard matrices and their applications. Princeton NJ: Princeton University Press 2007.
[8] Omondi A, Premkumar B. Residue Number Systems: Theory and Implementation. London: Imperial College Press 2007.
[9] Hinton GE. Learning to represent visual input. Philos Trans R Soc Lond Ser B Biol Sci [Review] 2010; 365(1537): 177-84.
[10] Kohonen T. Self-organized formation of topologically correct feature maps. Biol Cybern 1982; 43: 59-69.
[11] Hinton GE, McClelland JL, Rumelhart DE, Eds. Distributed representations. Cambridge, MA: MIT Press 1986.
[12] Castleman KR. Digital image processing. Englewood Cliffs NJ: Prentice Hall 1996.
[13] Ball WWR, Coxeter HSM. Mathematical Recreations and Essays. 13 ed. New York: Dover Publications 1987; pp. 50-2.
[14] Wolfe J, Houweling AR, Brecht M. Sparse and powerful cortical spikes. Curr Opin Neurobiol 2010; 20: 1-7.
[15] Destexhe A, Contreras D. Neuronal computations with stochastic network states. Science 2006; 314(5796): 85-90.
[16] Simon HA. The Architecture of Complexity. Proceed Am Philosophic Soc 1962; 106: 467-82.
[17] Bullmore E, Sporns O. Complex brain networks: graph theoretical analysis of structural and functional systems. Nat Rev Neurosci 2009; 10(3): 186-98.
[18] Garner HL, Ed. The Residue Number System. Proceeding of the western joint computer conference; 1959; San Francisco, USA.
[19] Fossorier M, Jezek J, Nation J, Pogel A. Ordinary graphs and subplane partitions. Discr Math 2004; 282: 137-48.
[20] Schiller PH. Inaugural Article: Parallel information processing channels created in the retina. Proc Natl Acad Sci U S A 2010; 107(40): 17087-94.
[21] Scholl B, Gao X, Wehr M. Nonoverlapping sets of synapses drive on responses and off responses in auditory cortex. Neuron; 65(3): 412-21.
[22] Jenmalm P, Birznieks I, Goodwin AW, Johansson RS. Influence of object shape on responses of human tactile afferents under conditions characteristic of manipulation. Eur J Neurosci 2003; 18(1): 164-76.
[23] Horton JC, Adams DL. The cortical column: a structure without a function. Philos Trans R Soc Lond B Biol Sci 2005; 360(1456): 837-62.
[24] Nieder A, Dehaene S. Representation of number in the brain. Annu Rev Neurosci 2009; 32: 185-208.
[25] Chen Q, Verguts T. Beyond the mental number line: a neural network model of number-space interactions. Cogn Psychol 2010; 60(3): 218-40.
[26] Park DC, Reuter-Lorenz P. The adaptive brain: aging and neurocognitive scaffolding. Annu Rev Psychol 2009; 60: 173-96.

[27] Olshausen BA, Field DJ. Sparse coding of sensory inputs. Curr Opin Neurobiol 2004; 14(4): 481-7.

[28] Wang HP, Spencer D, Fellous JM, Sejnowski TJ. Synchrony of thalamocortical inputs maximizes cortical reliability. Science 2010; 328(5974): 106-9.

[29] Moser EI, Kropff E, Moser MB. Place cells, grid cells, and the brain's spatial representation system. Ann Rev Neurosci 2008; 31: 69-89.

[30] Spruston N. Pyramidal neurons: dendritic structure and synaptic integration. Nat Rev Neurosci 2008; 9(3): 206-21.

[31] Kaiser M. Brain architecture: a design for natural computation. Philos Transact A Math Phys Eng Sci 2007; 365(1861): 3033-45.

Received: September 16, 2011 Revised: November 22, 2011 Accepted: November 23, 2011