A large-scale model of the functioning brain - Supplemental material

1 Materials and methods

1.1 Semantic pointers

We adopt the methods of the Semantic Pointer Architecture (SPA) to construct Spaun (*21*). The central kinds of representation employed in the SPA are "semantic pointers." Semantic pointers are neurally realized representations of a vector space generated through a compression method. In the main text, we refer to these representations only as firing patterns, to avoid the introduction of new theoretical terminology. However, introducing this terminology is helpful for describing the methods used to construct the Spaun model.

Semantic pointers are constructed by compressing information from one or more other high-dimensional neural vector representations, which can themselves be semantic pointers. The newly generated semantic pointer is of a similar or lower dimensionality than its constituents. Any semantic pointer can be subsequently decompressed (or "dereferenced") to recover (much of) the original information.

Semantic pointers are compact ways of referencing large amounts of data; consequently they function similarly to "pointers" as understood in computer science. Typically, in computer science a "pointer" is the address of some large amount of data stored in memory. Pointers are easy to transmit, manipulate and store, because they are a few bytes. Hence they can act as an efficient proxy for the data they point to. Semantic pointers provide the same kind of efficiency benefits in a neural setting.

Unlike pointers in computer science, however, semantic pointers are *semantic*. That is, they are systematically related to the information that they are used to reference, because they were generated via compression from that information. This means that semantic pointers carry similarity information that is derived from their source, in contrast to an arbitrary index that does not contain semantic information.

Compression operations used in the construction of semantic pointers can be learned or defined explicitly. In Spaun, the visual hierarchy employs learned compression, whereas the working memory and motor hierarchies employ defined compression (see section 1.3). The decoded conceptual representations shown in similarity graphs (e.g. Figures 2 and 3) are the dot product between the semantic pointer representation decoded from the neural activity (see section 1.2) and a labelled, ideal semantic pointers. This is purely for visualization purposes, as the model itself only processes the underlying neural activity.

1.2 Representing and transforming semantic pointers in spiking neurons

Semantic pointers are "neurally realized," "compressed," and "decompressed" (or "dereferenced") representations in a vector space. To characterize both neural representation and neural computation, the SPA employs the Neural Engineering Framework (NEF) (*22*). The NEF provides a set of methods for building biologically plausible models based on a functional specification of a neural system. These methods have been broadly employed to generate detailed spiking neural models of a wide variety of neural systems and behaviors, including the barn owl auditory system (*23*), parts of the rodent navigation system (*24*), escape and swimming control in zebrafish (*25*), tactile working memory in monkeys (*26*), decision making in humans (*27*), and central functions of the basal ganglia (*28,14*).

The central idea behind the NEF is that a group of spiking neurons can represent a vector space over time, and that connections between groups of neurons can compute functions on those vectors. The NEF provides a set of methods for determining what the connections need to be to compute a given function on the vector space represented by a group of neurons.

Suppose we wish to compute the function $\mathbf{y} = f(\mathbf{x})$, where vector space $\mathbf{x}$ is represented in population A, and vector space $\mathbf{y}$ is represented in population B. To do so, the NEF assumes that each neuron in A and B has a "preferred direction vector" (*29*) The preferred direction vector is the vector (i.e. direction in the vector space) for which that neuron will fire most strongly. This is a well-established way to characterize the behavior of motor neurons (*30*), because the direction of motion – hence the vector represented in the neural group in motor cortex – is directly observable. This kind of characterization of neural response has also been used in the head direction system (*31*), visual system (*32*), and auditory system (*23*). The NEF generalizes this notion to all neural representation.

The spiking activity of every neuron in population A can be written as

$$a_i(\mathbf{x}) = G_i[\alpha_i \mathbf{e}_i \mathbf{x} + J_i^{bias}] \tag{1}$$

where a_i is the spike train of the ith neuron in the population, G is the spiking neural nonlinearity, α is the gain of the neuron, $\mathbf{e}$ is the preferred direction (or "encoding") vector, and J^{bias} is a bias current to account for background activity of the neuron. Notably, the elements in the square brackets determine the current flowing into the cell, which then drives the spiking of the chosen single cell model G. For computational efficiency, we employ a leaky-integrate-and-fire (LIF) model of neurons in Spaun, but the NEF is defined more generally. Equation (1) describes how a vector space is encoded into neural spikes. This equation is depicted for a 2-dimensional vector space in Figure S1.

The NEF proposes that linear decoders can be found to provide an appropriate estimate of any vector $\mathbf{x}$ given the neural activities from the encoding equation. We can write this as a decoding equation:

$$\hat{\mathbf{x}} = \sum_i^N a_i(\mathbf{x}) \mathbf{d}_i \tag{2}$$

where N is the number of neurons in the group, $\mathbf{d}_i$ are the linear decoders, and $\hat{\mathbf{x}}$ is the estimate

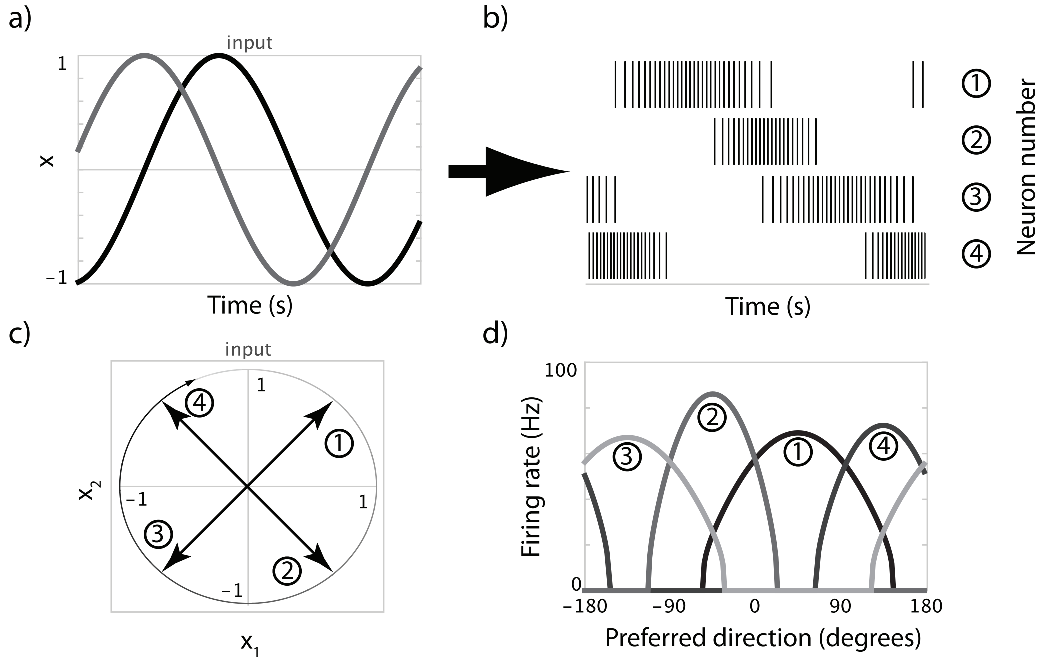


Figure S1: NEF encoding in two dimensions with four neurons. a) Both dimensions of the input plotted on the same graph, over 1.2s. The input to the two dimensions is $x_1 = \sin(6t)$ (black) and $x_2 = \cos(6t)$ (gray). b) The spikes generated by the neurons in the group driven by the input in a). c) The same input shown in the vector space. The path of the input is a unit circle, where the arrowhead indicates the vector at the end of the run, and the direction of movement. Older inputs are in progressively lighter gray. The preferred direction vectors of all four neurons is also shown. d) The firing rate tuning curves of all four neurons. Gains and biases are randomly chosen. The script for generating elements of this figure is in the supplementary material (four_neurons.py).

of the input driving the neurons.

The NEF determines this complementary decoding for any given encoding. Specifically, this decoding found using a least-squares optimization:

$$E = \frac{1}{2}\int [\mathbf{x} - \sum_i a_i(\mathbf{x})\mathbf{d}_i]^2 d\mathbf{x} \tag{3}$$

where $\mathbf{d}_i$ are the decoding vectors over which this error is minimized.

In effect, this optimization process replaces learning in most other approaches to constructing neural networks. This optimization is not biologically plausible on its own, although networks generated in this manner can also be learned with a spike-based rule (*15*). In the Spaun model, all optimizations of this type use 5000 or fewer sample points to find the decoders. This results in a significant computational savings (several orders of magnitude) over trying to learn the same function in a spiking network setting.

The decoding process is depicted in Figure S2, where the optimal linear decoders have been found and used for twenty neurons. Notably, this kind of temporal decoding requires an assumption about the nature of the temporal filter being used. Here we assume that post-synaptic currents are such filters, and set the time constants to reflect the kind of neurotransmitter receptors in the connection (e.g., AMPA receptors have short time constants (˜10ms) and NMDA receptors have longer time constants (˜50ms)).

Such temporal filters map to biophysical processes once we connect groups of neurons together. Defining the encoding and decoding for groups A and B using equations (1) and (2) provides a means of connecting groups. For example, we can substitute the decoding of A into the encoding of B, thereby deriving connection weights

$$\omega_{ij} = \mathbf{d}_i \alpha_j \mathbf{e}_j \tag{4}$$

where i indexes the neurons in group A and j indexes the neurons in B. These weights will compute the function $\mathbf{y} = \mathbf{x}$ (where $\mathbf{y}$ is the vector space represented in B and $\mathbf{x}$ is the vector

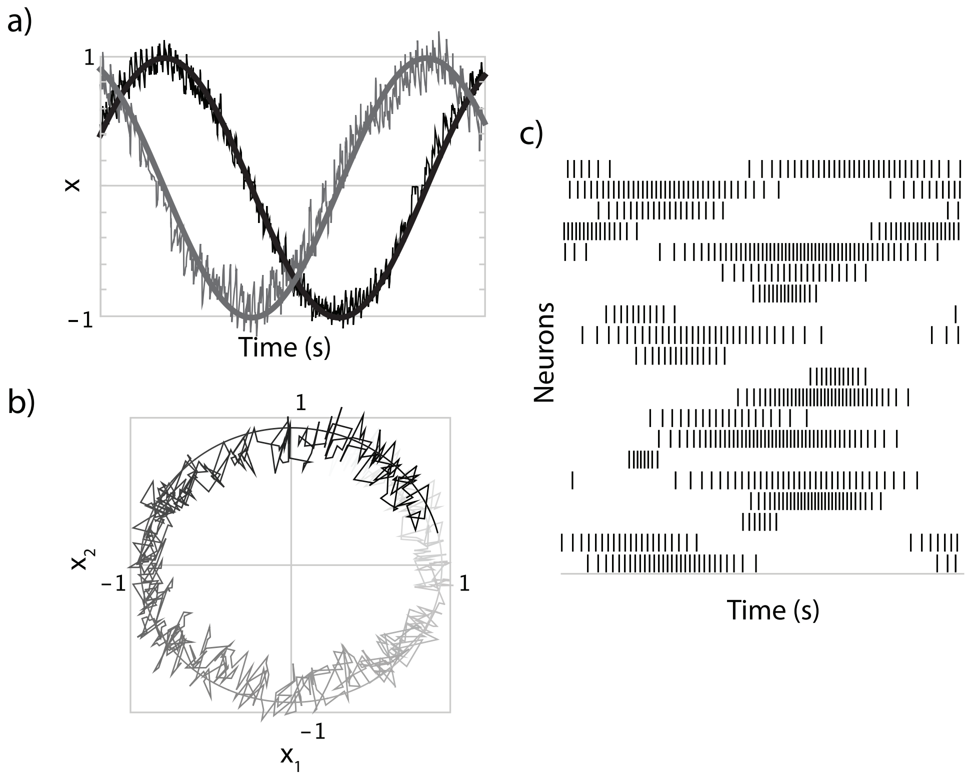


Figure S2: NEF decoding in two dimensions with 20 neurons. The inputs in the vector space are the same as in Figure S1. a) The original input and neuron estimate over 1.2s, with both dimensions plotted on the same graph over time (black is x1, gray is x2). b) The same simulation shown in the vector space. Older states are lighter gray. For both a and b, smooth lines represent the ideal $\mathbf{x}$ values, while noisy lines represent the estimate $\hat{\mathbf{x}}$. c) The spikes generated by the 20 neurons during the simulation, and used to generate the decodings shown in a) and b). Encoding vectors are randomly chosen from an uniform distribution around the unit circle, and gains and biases are also randomly chosen, as in Spaun. The script for generating elements of this figure is in the supplementary material (twenty_neurons.py).

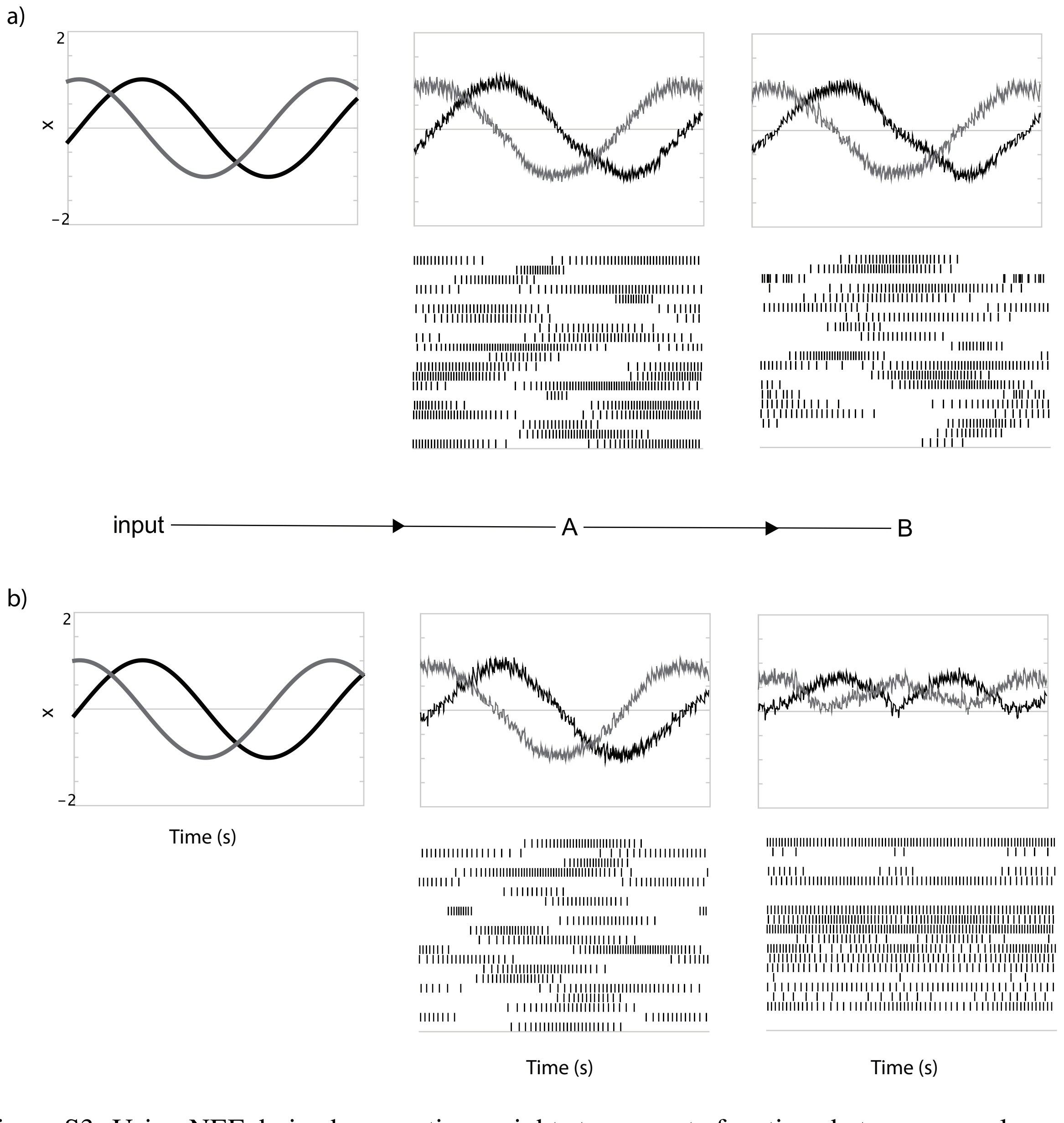


Figure S3: Using NEF derived connection weights to compute functions between neural populations representing 2-dimensional vectors. a) Computing the identity function between A and B. b) Computing the element-wise square between A and B. These simulations are 1.2s long. Both populations have 20 neurons, with randomly chosen encoders, gains and biases. The script for generating elements of this figure is in the supplementary material (vector_square.py).

space represented in A), as shown in Figure S3a. For the more general case, it is possible to solve for decoders $\mathbf{d}_i^f$ for any function by substituting $f(\mathbf{x})$ for $\mathbf{x}$ in equation (3), i.e., solving

$$E = \frac{1}{2} \int [f(\mathbf{x}) - \sum_i a_i(\mathbf{x}) \mathbf{d}_i^f]^2 d\mathbf{x}. \tag{5}$$

In addition, if the function to be computed is linear, the relevant linear operator can be introduced into Equation (4). The resulting general weight equation for computing any combination of linear and nonlinear functions becomes:

$$\omega_{ij} = \alpha_j \mathbf{d}_i^f \mathbf{L} \mathbf{e}_j \tag{6}$$

for any nonlinear function f and $N_B \times N_A$ linear operator $\mathbf{L}$. Computing the nonlinear function which is the element-wise square of the vector $\mathbf{x}$ (i.e., $f(\mathbf{x}) = [x_1^2, x_2^2, ..., x_n^2]$ and $\mathbf{L} = \mathbf{I}$) is shown in Figure S3b.

This brief discussion is insufficient to fully introduce the generality of the NEF. However, showing how to compute linear and nonlinear functions of vector spaces is sufficient for many neural computations. As these same methods can be used to compute connection weights for recurrent connections, the NEF also allows for the neural implementation of a wide variety of linear and nonlinear dynamical systems in recurrent networks (*22*). In the context of the Spaun model, these methods are used to realize spiking implementations consistent with the functions employed in Spaun's functional architecture.

1.3 Details of Spaun's architecture

In Figure 1 of the main text, we have diagrammed the anatomical and functional architecture that underlies the Spaun model. Table S1 characterizes the mapping between these two views of the model in more detail, defining abbreviations used, the functions of major brain areas employed, and citations to relevant work. In the remainder of this section we provide more in-depth descriptions of each of the functional elements from Figure 1b.

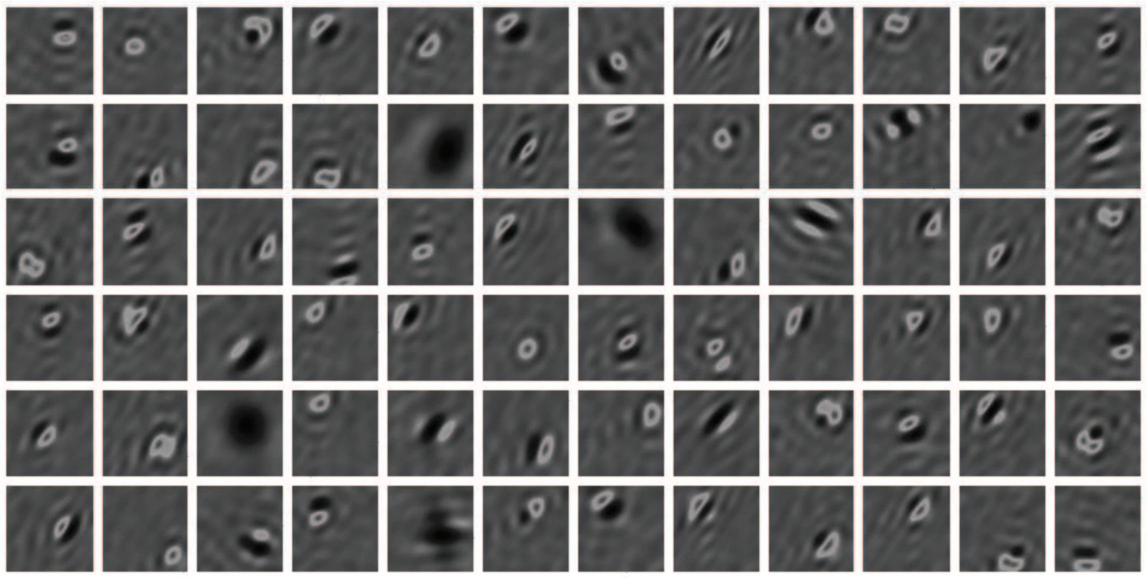
Table S1: The function to anatomical mapping used in Spaun, including anatomical abbreviations used in the main text.

Functional element	Acronym	Full name and description
visual input	V1	primary visual cortex: the first level of the visual hierarchy, tuned to small oriented patches of different spatial frequencies (*63*)
	V2	secondary visual cortex: pools responses from V1, representing larger spatial patterns (*63*)
	V4	extrastriate visual cortex: combines input from V2 to recognize simple geometric shapes (*63*)
	IT	inferior temporal cortex: the highest level of the visual hierarchy, representing complex objects (*63*)
information encoding	AIT	anterior inferior temporal cortex: implicated in representing visual features for classification and conceptualization (*64*)
transform calculation	VLPFC	ventrolateral prefrontal cortex: area involved in rule learning for pattern matching in cognitive tasks (*41*)
reward evaluation	OFC	orbitofrontal cortex: areas involved in the representation of received reward (*43*)
information decoding	PFC	prefrontal cortex: implicated in a wide variety of functions, including executive functions and manipulation of working memory (*44*)
working memory	PPC	posterior parietal cortex: involved in the temporary storage and manipulation of information, particularly visual data (*65,37*)
	DLPFC	dorsolateral prefrontal cortex: temporary storage and manipulation of higher level data related to cognitive control (*66,40*)
action selection	Str (D1)	striatum (D1 dopamine neurons): input to the “direct pathway” of the basal ganglia (*67*)
	Str (D2)	striatum (D2 dopamine neurons): input to the “indirect pathway” of the basal ganglia (*67*)
	STN	subthalamic nucleus: input to the “hyperdirect pathway” of the basal ganglia (*68*)
	VStr	ventral striatum: involved in the representation of expected reward in order to generate reward prediction error (*69*)

Functional element	Acronym	Full name and description
	GPe	globus pallidus externus: part of the "indirect pathway", projects to other components of the basal ganglia in order to modulate their activity (*70*)
	GPi/SNr	globus pallidus internus and substantia nigra pars reticulata: the output from the basal ganglia (*67*)
	SNc/VTA	substantia nigra pars compacta and ventral tegmental area: relay signal from ventral striatum as dopamine modulation to control learning in basal ganglia connections (*71*)
routing	thalamus	thalamus: receives output from the basal ganglia, sensory input, and coordinates/monitors interactions between cortical areas (*72*)
motor processing	PM	premotor cortex: involved in the planning and guidance of complex movement (*45,73*)
motor output	M1	primary motor cortex: generates muscle based control signals that realize a given internal movement command (*74*)
	SMA	supplementary motor area: involved in the generation of complex movements (*73*)

Visual hierarchy The visual hierarchy is a model of the ventral visual stream, including areas V1, V2, V4, and IT. The model is constructed by training Restricted Boltzmann Machine (RBM) based auto-encoders (*6*) on natural images and the Modified National Institute of Standards and Technology (MNIST) handwriting database[1]. The methods used result in a noise tolerant and sparse RBM (*33*). The network has a 28×28 dimensional visible layer (the 728-dimensional input image) and consecutive hidden layers of 1000, 500, 300 and 50 nodes. Like the visual system, the first layer of the cortical visual system (V1) is higher-dimensional than the actual input image. Hence we refer to it as an image-based space to distinguish it from the original image space.

[1]The database can be downloaded at http://yann.lecun.com/exdb/mnist/.

a)


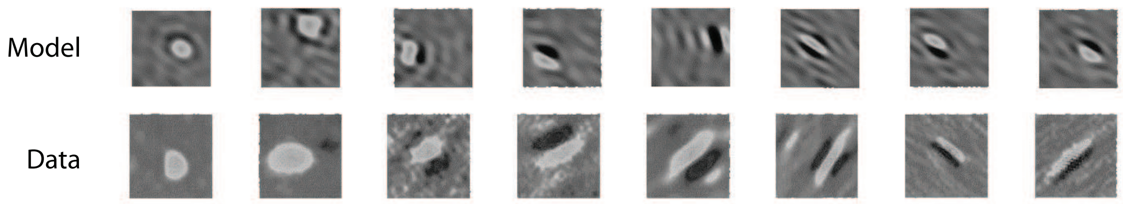
b)


Figure S4: A sample of tuning curves of neurons in the model. a) These learned model representations have a variety of orientations, spatial frequencies, and positions, like those found in V1. b) A more direct comparison of data and model tuning curves from V1. The analytical methods used to generate these images are identical for the model and monkey data (monkey data adapted from (*61*).)

This network learns the compression needed to reduce the original image to a 50-dimensional semantic pointer. These layers define vector spaces that can be embedded into spiking neurons using the NEF methods. The connections between these layers define the vector space transformations that can be implemented in the NEF connection weights. Example tuning curves from layer V1 of an RBM embedded into spiking neurons using the NEF are shown in Figure S4. To enable running the full Spaun model on available hardware, Spaun only implements IT in spiking neurons. The other layers compute the transformations between vector spaces in the original RBM, although neural dynamics are included in this computation. Implementing these transformations in spiking neurons would require approximately an additional 250,000 neurons in Spaun. This made it infeasible to simulate the model on available hardware (see section 1.4).

Motor hierarchy The methods used to construct the motor hierarchy are described in detail in (*34*). In brief, the hierarchy consists of an optimal controller in the workspace that determines control commands that are then projected to joint angle space. This model of the motor hierarchy has been shown to account for the behavioral effects of a variety of motor system disturbances, including Huntington's, Parkinson's, and cerebellar damage. In addition, it has been shown to qualitatively reproduce the dynamics of spiking neurons observed in monkey cortex during similar reaching tasks (*34*).

Working memory The working memory hierarchy is a central element of Spaun. It consists of several distinct memory systems, each of which can store semantic pointers. Some employ compression and some do not, but they have been grouped into a single subsystem for simplicity. All are recurrent attractor neural networks, based on the multidimensional working memory circuit of (*26*). This circuit has been shown to reproduce the variety of single cell dynamics observed in somatosensory working memory, and recent extensions of this work account for over 95% of the variance in a population of 800 cells (*35*). Anatomically, the functions of the working memory hierarchy in Spaun cover large portions of prefrontal and parietal cortex (*36,37*).

The networks that employ compression use circular convolution to perform compression (*29*). This is an example of a defined compression operator for generating semantic pointers (this operator can also be learned in a spiking network (*29*)). This operator can be thought of binding two vectors together (*38*). Consequently, serial memories are constructed by binding the semantic pointer of the current input with its appropriate position, e.g.,

$$MemoryTrace = Position1 \otimes Item1 + Position2 \otimes Item2 + ... \tag{7}$$

where $Item$ semantic pointers are the semantic pointers to be stored (numbers in Spaun), and $Position$ semantic pointers are internally generated position index semantic pointers. Position indices are generated using random unitary base semantic pointers, $Base$, where the next position index is generated by successive convolution, e.g.,

$$\begin{aligned} Position1 &= Base \\ Position2 &= Position1 \otimes Base \\ Position3 &= Position2 \otimes Base \\ &\vdots \quad . \end{aligned}$$

This allows for the generation of as many positions as needed to encode a given list. A unitary vector is one which does not change length when it is convolved with itself. In the figures in the main text $Position1$ is written as $P1$, and so on.

The overall memory trace is a sum of this encoding through two memory pathways, which have different decay dynamics. In our past work, this approach to working memory has been shown to reproduce human results on 7 different working memory experiments (including free recall, confusable item recall, delayed recall, backward recall, etc.), while being based on an independent fit of only two parameters (*39*).

Anatomically, working memory responses are found in large areas of parietal and frontal cortices, including PPC and DLPFC (*37,40*).

Information encoding The information encoding subsystem maps semantic pointers generated from images to conceptual semantic pointers that encode progression relations. This information encoding is used for most tasks except the copy drawing task, where it is the visual features of the input, not its conceptual properties, that matter for successful completion of the task.

The conceptual semantic pointers for numbers are constructed in a similar manner to those for position. That is,

$$\begin{aligned} One &= Base \\ Two &= One \otimes AddOne \\ Three &= Two \otimes AddOne \\ &\vdots \quad . \end{aligned}$$

where $Base$ and $AddOne$ are random unitary semantic pointers.

Anatomically, this mapping is likely implemented in later parts of the ventral visual stream such as AIT.

Transformation calculation The transformation calculation subsystem is a recurrent attractor network similar to the working memory elements, with an input transformation. Its purpose is to compute the transformation between its inputs and store a running average of the result. As a result, this running average is the inferred relation that exists between all of the inputs it is shown. This subsystem is most critical for the RPM and rapid variable creation tasks. fMRI studies have suggested that rule learning of this kind takes place in VLPFC (*41*).

We are not aware of any past work outside of our group that has employed this method for solving such tasks (*42*).

Reward evaluation This subsystem determines if the current input in the current context has an associated reward. In Spaun this means that during the reinforcement learning task, if a '1' is shown after a guess, a positive reward signal is generated and sent to the ventral striatum and subsequently if the reward is unpredicted, to the dopamine system in the basal ganglia. All further reward processing is done in the basal ganglia. This mapping from visual input to reward is consistent with the function of OFC involvement in corticostriatal loops (*43*).

Information decoding The information decoding subsystem is largely used to extract information that is useful for motor control from the working memory hierarchy. That is, when information is encoded in working memory in the form shown in Equation 7 specific items must be extracted so that they can be used to drive behavior. This sort of function is assumed to be intermediate between working memory and motor areas, and hence is expected to be anatomically located in medial PFC (*44*).

Motor processing The motor processing subsystem typically takes the results from the information decoding subsystem and helps to present it to the motor hierarchy for driving the arm. Consequently it is important for timing the presentation of decoded output to ensure that recently sent motor commands are completed before new ones are presented. Premotor cortex is involved in this kind of motor planning (*45*).

Table S2: Example input/output mappings for each task. Randomly selected handwritten digits not used during training are used in the first two tasks. If Spaun does not know the proper response, it will write a dash (see, e.g., serial working memory).

Task Name	Input Examples	Typical Output
Image recognition	A1 [3 ?	3
Copy drawing	A0 [4 ?	4
Reinforcement learning	A2 ? 1 ? 0 ? 0...	0 0 1
Serial working memory	A3 [3 2 4 3 5] ? A3 [4 2 5 3 4 2] ?	3 2 4 3 5 4 2 5 – 4 2
Counting	A4 [3] [2] ? A4 [0] [7] ?	5 7
Question answering	A5 [4 3 8 6] [K] [4] ? A5 [4 3 8 6] [P] [4] ?	1 6
Rapid variable creation	A6 [3312] [12][3392] [92] [3362] [62] [3342] ? A6 [342] [2][345] [5] [347] [7] [340] ?	4 2 0
Fluid reasoning	A7 [5] [55] [555] [2] [22] [222] [1] [11] ? A7 [4] [3] [2] [8] [7] [6] [4] [3] ?	1 1 1 2

1.4 Simulation details

Table S2 shows examples of the kinds of inputs that Spaun receives for each task, and the kinds of responses that it provides.

All neurons used in the simulations are leaky-integrate-and-fire (LIF) model neurons. The membrane time constant was set to 20ms, the absolute refractory period to 2ms, and maximum firing rates were chosen from a uniform distribution between 100-200Hz. Encoding vectors are chosen from an uniform distribution around the unit hyper-sphere. AMPA glutamate receptors were taken to have a time constant of 10ms, and were used for most projections in the model, except recurrent projections, which were taken to employ NMDA receptors with a time constant of 50ms. Projections in the basal ganglia employed inhibitory GABA receptors as anatomically appropriate, with a time constant of 8ms. Dopamine projections used a time constant of 10ms.

All simulations were run using the freely available Nengo simulation software package (`http://www.nengo.ca`). The full model can be downloaded from the Nengo model archive (`http://models.nengo.ca/spaun`).

The simulations were run on the clusters Orca and Kraken of the SharcNet High Performance Computing Consortium (`http://sharcnet.ca/`). These were the only available computers that could load the model, as it occupies approximately 24GB of RAM (to view the structure of the model, the network can be constructed with very few neurons, requiring approximately 2GB of RAM; see the readme file for details). The simulations take approximately 2.5h of processing time for 1s of simulated time.

2 Supplemental results

2.1 Videos

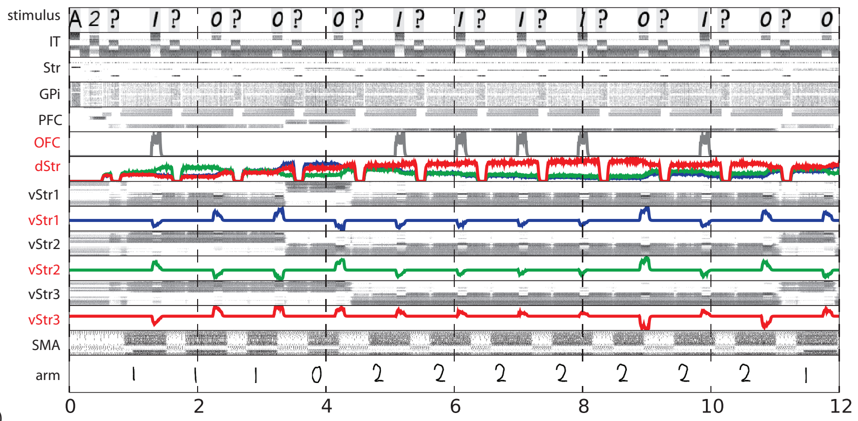
Videos of all tasks can be viewed at `http://nengo.ca/build-a-brain/spaunvideos`, and are included in the Supplemental Online Material.

2.2 Supporting figures

Figure S5a shows a single run on a reinforcement learning task. This demonstrates the behavioral flexibility of the model, as it solves a three-armed bandit task by adjusting its behavior given changing contingent rewards. In other work, we have shown that the detailed spike patterns found in the striatum of this basal ganglia model matches those found in rats performing this same task (*14*). As well, figure S5b shows many more trials than in Figure S5a. This helps to demonstrate that changing contingencies are indeed learned by the model. Over 60 trials, each of the arms becomes the most highly rewarded for a period of time, and the model's choice probability tracks those changes.

Figure S6 provides another example pattern that Spaun can discover in the RPM task. In general, any pattern that exploits either numerical ordering relations, or object number relations

a)



b)

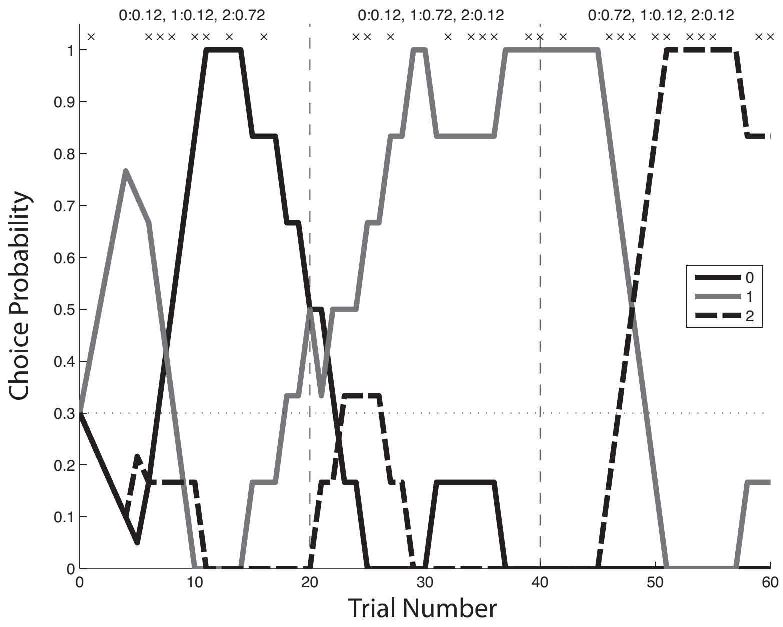


Figure S5: Choice probability in a three-armed bandit task in Spaun over 60 trials. Every 20 trials, the probability of reward for the three choices changes, as indicated at the top of the graph (e.g. 0:0.12 indicates that choice 0 has a 12% chance of being rewarded). The probability of choosing each action is indicated by the continuous lines. These probabilities are generated by averaging over a 5-trial window. Reward delivery during the run is indicated by the ‘x’ marks along the top of the graph. Note that Spaun learns to vary its selected choice as appropriate for the changing environmental reward contingencies.

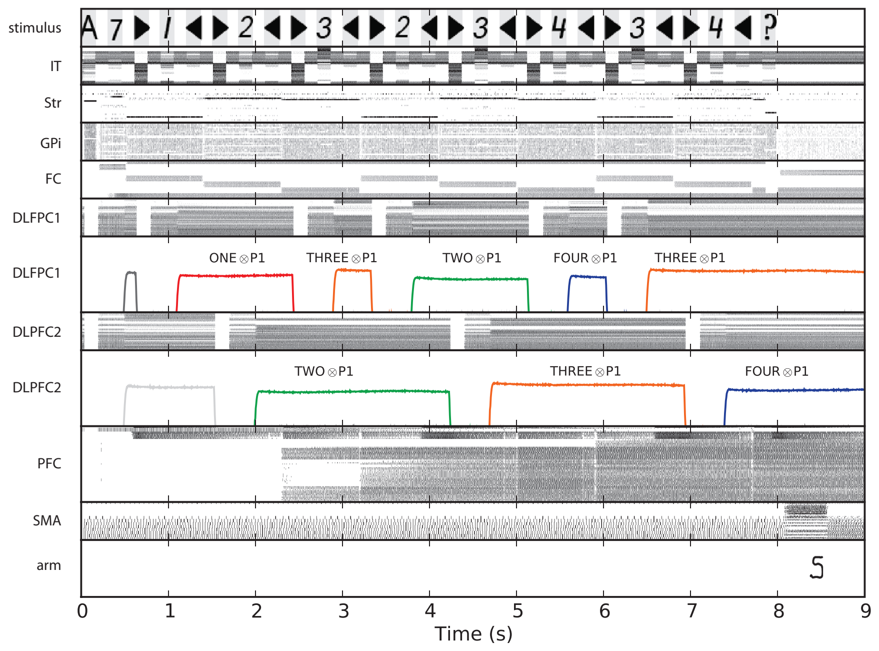


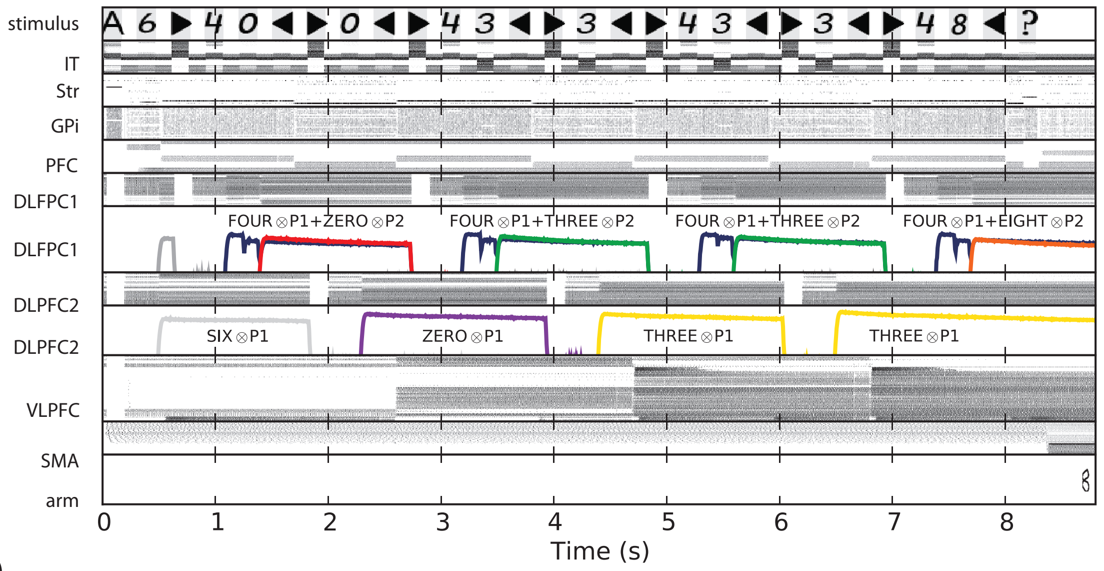
Figure S6: An additional example of fluid reasoning in Spaun. The time course of Spaun's activity while inferring the pattern provided in the input images. This figure is plotted using the same methods as in Figure 2 in the main text. Color is used to distinguish conceptual decodings (also labelled). Spaun is able to complete the final set of three inputs by having learned the appropriate transformation from the first two sets. It learns this transformation by comparing the appropriate elements of DLPFC1 and DLPFC2.

will be discovered by the model, whether they be increasing or decreasing.

Figure S7 shows the rapid variable creation task for two example runs. This task was identified as one which no contemporary neural model could perform as quickly as humans (i.e., within 2 seconds) (*20*). Spaun provides an answer after 150ms of simulated time, for a variety of patterns.

Figure S8a shows results from the counting task. Specifically, it shows the length of time required by the model to produce a response, as a function of the number of positions counted. The model reproduces the expected linear relationship between subvocal counting and response times (*46*). Spaun's count time per item (419 ± 10 ms) lies within the human measured range of 344 ± 135 ms for subvocal counting (*47*), although the variance is lower. This is likely a result

a)



b)

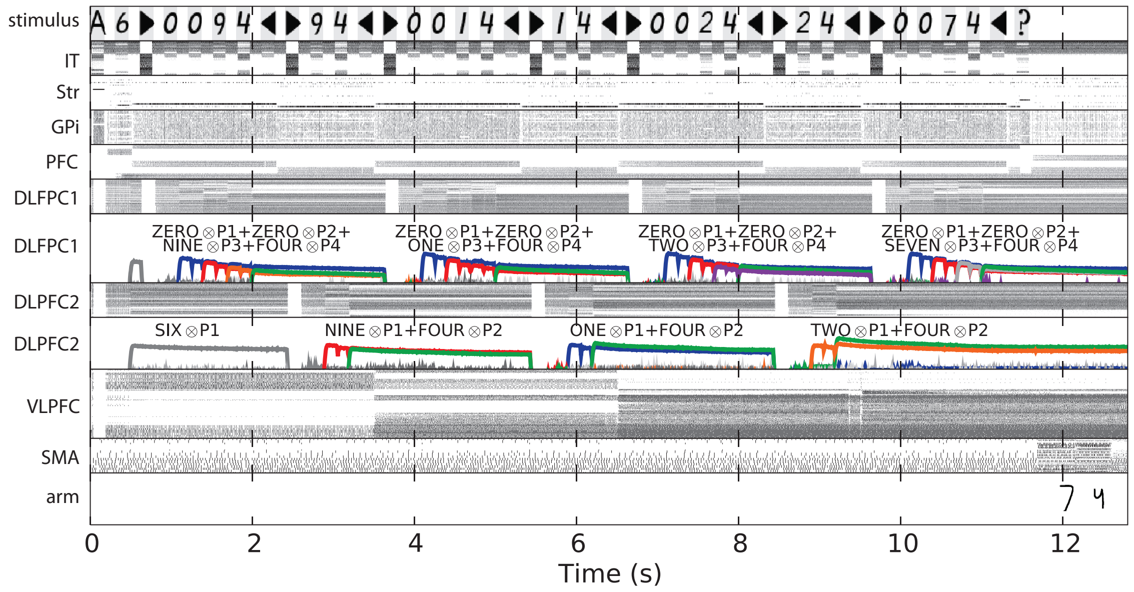


Figure S7: Time course plots for rapid variable creation. All graphs are plotted as described in figure 2. Color is used to distinguish conceptual decodings (also labelled). Both graphs provide examples of Spaun learning how to complete a syntactic pattern given input/output examples. The examples are stored across DLPFC1 and DLPFC2, and the pattern is learned by comparing these in VLPFC. a) A simple syntactic pattern where the last of two items is the variable item. b) A more complex pattern, where the variable item is second last in a string of four items.

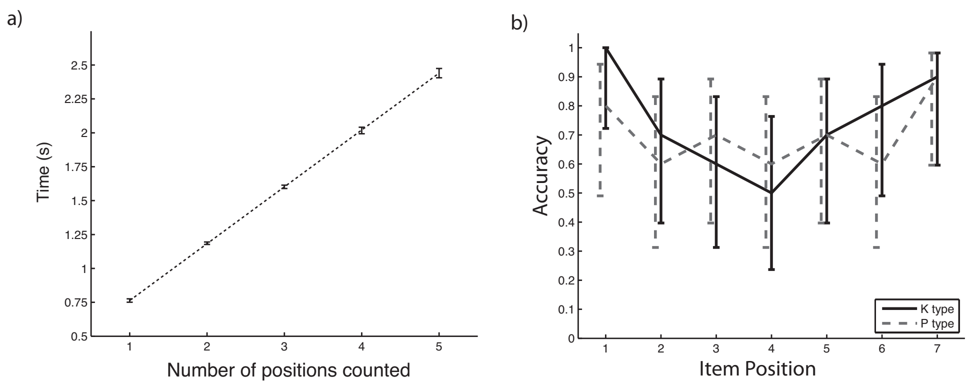


Figure S8: Population level behavioral data for two Spaun tasks. a) Reaction time as a function of the number of positions counted in the counting task. Error bars indicate one standard deviation over 5 simulated individuals, each doing all five count lengths. These reaction times are consistent with human reaction times, and correspondingly show increasing variability with list length. b) Predicted accuracy as a function of queried item position for both query types in the question answering task. Error bars indicate 95% confidence intervals for 10 simulated individuals, each queried at every position and for each type of question. Spaun predicts that humans will show no effect of question type, but show primacy and recency effects on the question answering task.

of the relative simplicity of the model. Interestingly, the model reproduces the well-known psychophysical regularity called Weber's law (i.e., that the variance in response time increases with the mean response time (*48*)), typically evident in such tasks. We suspect that this feature of the model is present because, despite not adding noise to the simulation, it is highly stochastic because of the many nonlinearities present. A stochastic system will tend to perform a random walk, which diffuses over time, generating a higher variance after longer temporal delays.

Figure S8b shows the accuracy rates of the model on the question answering task for lists of length 7. Cognitive modelers have long used the paradigm of question answering to evaluate knowledge representation – a model's ability to flexibly represent and access structured information (e.g., (*49*)). Spaun is able to perform this task, but human data for this specific task is not available. Consequently, Spaun produces the behavioral prediction that, while primacy and recency effects will be evident, the type of question asked will not affect accuracy.

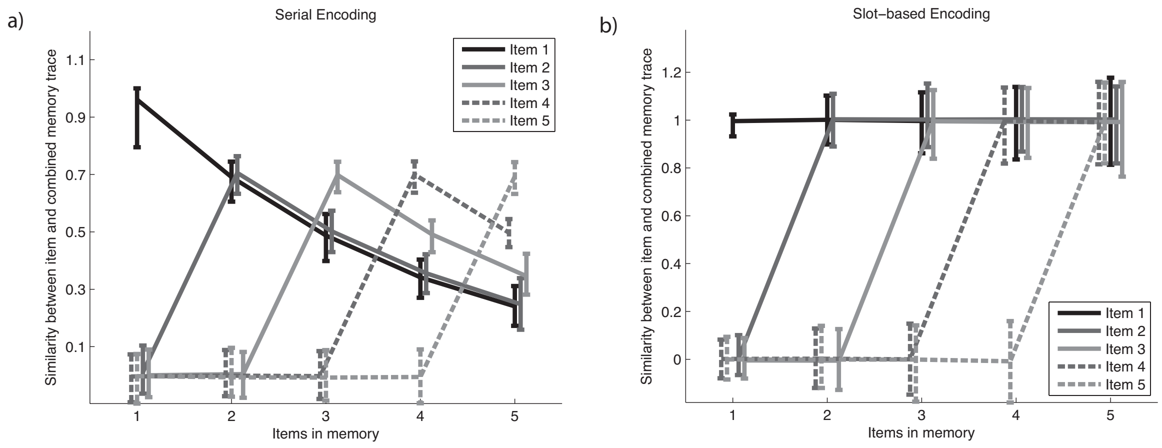


Figure S9: The expected change of similarity in neural firing during a serial working memory task. Each line indicates the expected similarity between a lone encoding of that item, and an encoding of that item in a list of length one to five. a) Serial encoding as used in Spaun. b) Slot-based encoding proposed by other models. Error bars are 95% confidence intervals.

A second prediction of the Spaun model comes from its means of implementing working memory. Spaun's implementation presumes a single memory into which items are added after being tagged with position information (section S1.3). This contrasts with models of working memory that assume a set number of independent 'slots' used to store subsequent items (*50*). Figure S9 compares the expected change in neural activity patterns in DLPFC during a serial working memory task. The figure shows the expected similarity between neural population activity while remembering a list of one to five items, and each of the items remembered alone. This figure was generated by directly comparing vector implementations of semantic pointers. Spiking activity can also be extracted from Spaun for direct comparison to electrophysiological experiments (not shown).

Figure S10 provides three examples of Spaun's responses to invalid or unusual input. This figure helps demonstrate the robustness of the model.

Figure S11 shows the results of an analysis on the working memory model spike trains that is identical to the analysis performed on spike trains recorded from monkeys during a

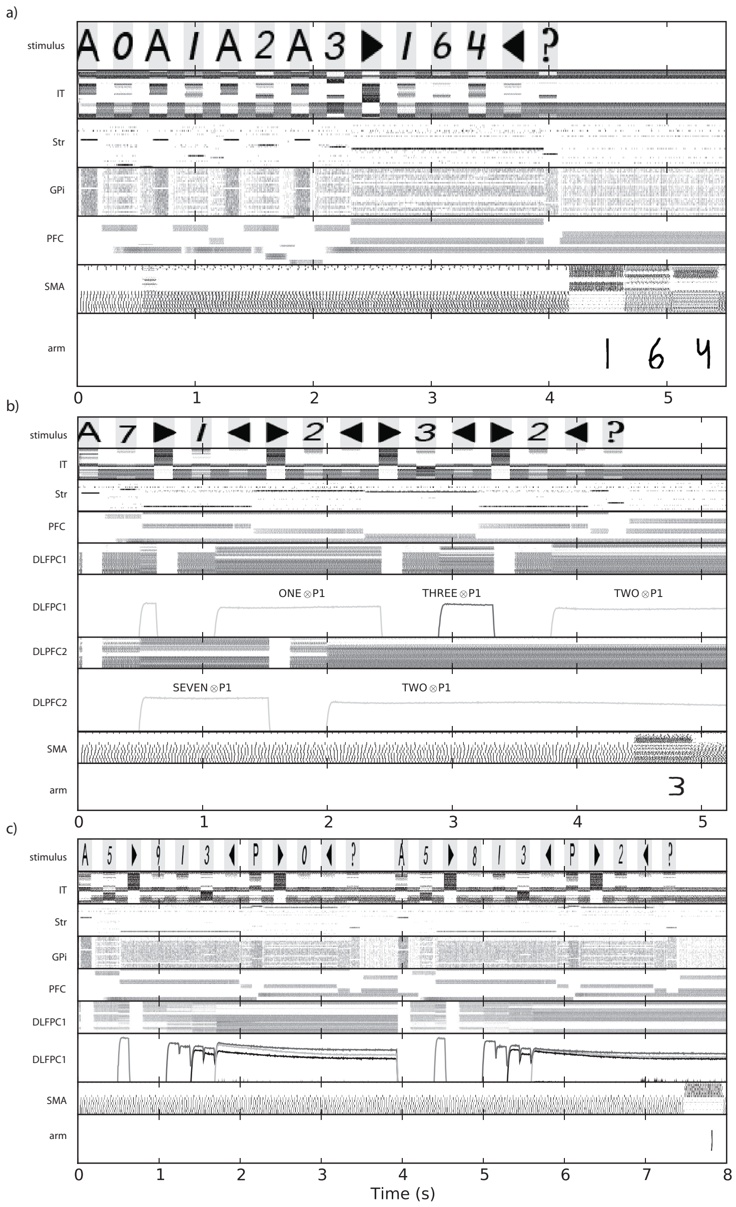


Figure S10: Invalid input in Spaun. a) A series of four tasks is identified before any valid input is provided. Nevertheless, Spaun performs the last identified task (serial working memory) correctly. b) Spaun is asked to complete an RPM matrix half way through the second row. It uses the information provided to give a best guess, which in this case is correct. c) A completely invalid question is asked in the question answering task (there is no “zero” position). Spaun does not respond, but proceeds normally to the subsequent task.

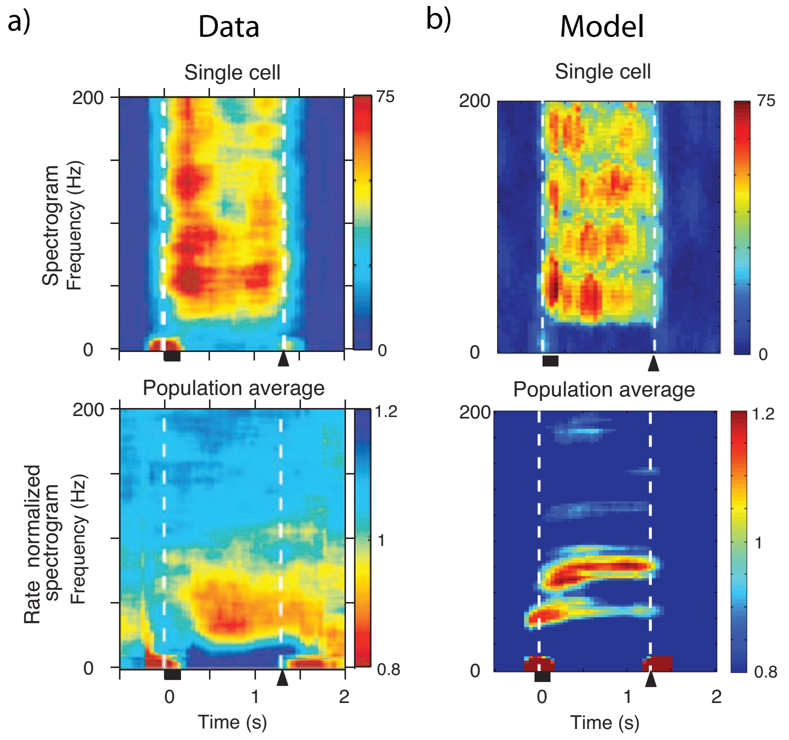


Figure S11: Oscillation frequency profiles during a working memory task. Spike spectrograms for a single neuron (top) and the population average (bottom) during a simple working memory task. White dashed lines indicate the stimulus presentation and recall cues. A) Data from macaque (adapted from (*62*)). b) Model spike trains are processed using the same algorithms as for the animal data. Both single cell responses show broadband power increase during the delay period. Both population averages show broad gamma band increases during delay as well as low frequency pulses during the start and end of the delay period.

working memory task. Both show a similar shift in the single cell and population frequency analyses between pre-delay and delay periods. Specifically, both the model and the data show a shift from minimal frequency response before the stimulus to stimulus-driven low frequency information (at the first white line) to significantly higher frequency content during the delay period, with a final reduction after response (at the second white line). Both population averages show frequency content mainly in the gamma range.

Table S3 shows the coefficient of variation (C.V.) of spikes recorded from neurons while Spaun performed the rapid variable creation task. In general, C.V.s of cortical and subcortical

Table S3: Mean coefficient of variation (C.V.) and 95% confidence intervals (C.I.) across all areas recorded during the rapid variable creation task. Abbreviations as in table S1.

Area	Mean C.V.	95% C.I.
IT	1.26	(1.11-1.41)
PFC	0.25	(0.23, 0.27)
DLPFC/PPC	0.98	(0.86, 1.10)
Str	2.83	(1.85, 4.12)
GPi	1.48	(1.00, 2.12)
VLPFC	0.40	(0.30, 0.52)
SMA	0.59	(0.50, 0.71)

neurons are between 0.5 and 2 (*51,52*). See section S2.4 for further discussion.

2.3 Statistics

For the RPM task, it is somewhat difficult to compare the model to human data because the task itself is slightly different. Spaun must *generate* the correct answer, but in the RPM, only a *match* to one of 8 possible answers must be indicated. To enable a comparison, we compute a match-adjusted success rate that assumes that Spaun narrows the response to two of the eight possibilities and randomly chooses between them. Human subjects average 89% correct (chance is 13%) on the Raven's matrices that include only an induction rule (5 of 36 matrices; (*16*)). Spaun is similarly good at these tasks, achieving a match-adjusted success rate of 88%. The raw accuracy rate of Spaun is 75% (95% CI of 60% to 88%) over 40 runs. Assume that Spaun's answer on its error trials would pick the correct match 50% of the time, the match-adjusted rate is $75\%+25\%\times50\% = 88\%$.

All reported confidence intervals were generated using 3000 bootstrapped samples.

2.4 Discussion

Limitations of Spaun

In the main text, we list various limitations of the model, but note that many of these limitations can be overcome using the same methods as were to used to construct Spaun. For instance, missing functions can be included in a larger model that encompasses more brain areas and includes more of the connectivity found in real brains. In the case of attention, we have developed a recent model that can be directly integrated into Spaun (*53*). Similarly, the limitation of the representations to the space of digits is a result keeping the complexity of the model manageable. However, the methods used for training the visual system and representing concepts can scale to more complex cases (*21*). In general, increasing the complexity of the model (adding neurons, anatomical areas, etc.) is consistent with the general methods used for Spaun. However, we do not mean to suggest that these changes will be simple, but rather that we are unaware of in principle limitations to addressing these kinds of challenges.

One of the limitations we mention in the main text that requires additional discussion is the variability of neural spiking in the model. As shown in Table S3, the C.V.s of neurons in 5 of 7 areas are consistent with the expected C.V.s of cortical and subcortical neurons. However, C.V.s are blunt metrics of variability, and closer inspection of spike rasters (figure S12) make it clear that the neurons are distinguishable from neural recordings by their regularity, and higher than expected firing rates. Consequently, Spaun does not always display the appropriate degree of single cell variability and spike rates. But, past work has demonstrated that the observed variety of neural variability can be fully reproduced with appropriate spike rates in NEF models (*54*). However, doing so requires more neurons, which currently presents too high a computational cost to allow us to run Spaun on available computers.

We also note that the sharp transitions in the raster plots in most figures are present for three reasons: 1) we are showing very long run times; and 2) we have sorted neurons so similar

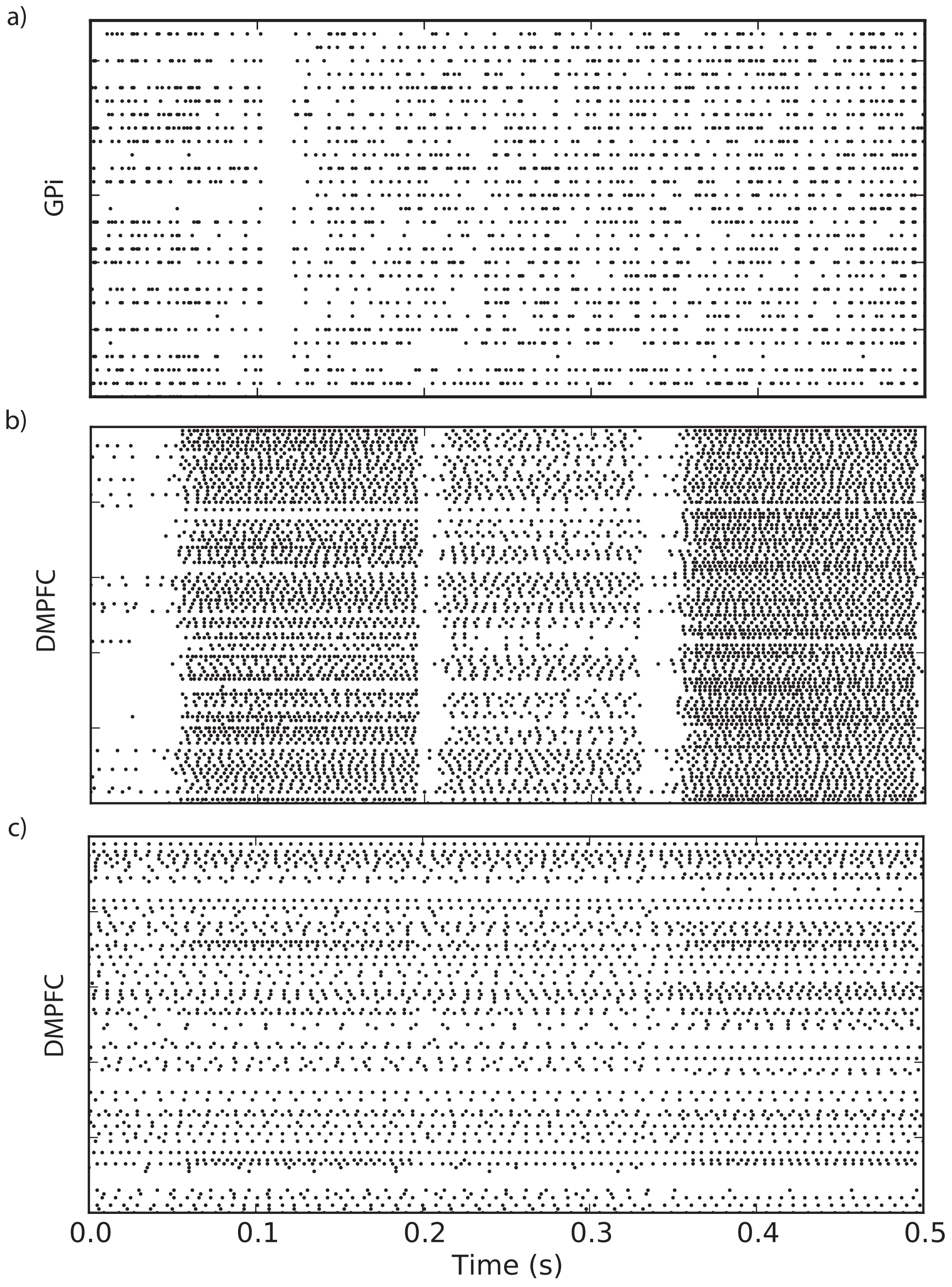


Figure S12: Sample spike trains from Spaun. These graphs show a close up view of the spike trains during a run of the rapid variable creation task. Coefficients of variation for these same areas are reported in table S3. In a) and b), neurons were not sorted and are shown for 500ms. However, only neurons with greater than 10% variability during the task were randomly selected from. In c) neurons were also completely randomly chosen from the area.

responses are near one another. Figure S12 does not make these assumptions, resulting in graphs more typical of neural data plots. More detailed comparisons between the mechanisms used in Spaun experimental raster data can be found in (*14,26*).

Learning

While Spaun is running, learning (i.e., updating future behavior based on past experience) occurs using two main mechanism. One is changing connection weights (on the RL task), the other is by updating neural activities that are sustained over time (on the RPM and rapid variable creation tasks). These are only two of the many neural mechanisms that support learning in the brain, and the model stands to be improved by including more.

In neural modeling, learning is also often used as a method of constructing the model. In the main text we mention that all elements of Spaun can be learned using a spike-based rule. The learning rule and other details are reported in (*15*). More specifically, that paper demonstrates learning of attractor networks such as Spaun's working memory, the learning methods used to generate the visual system is in (*33*), and the learning of the compression operator for binding is shown in (*29*). Additional examples of using this rule can be found in (*55*). While the rule proposed in (*15*) is spike-based, general, and biologically plausible, it also relies on an error signal. While many error signals have been identified in the brain, it remains a significant challenge to determine how they could be generated and targeted so as to learn a structure as complex as Spaun.

We generally do not use learning to generate model elements because the optimization method described in section S1.2 is significantly more computationally efficient. However, learning with the spike-based rule has been shown to be marginally more effective (*15*). Notably, learnability may also come at a very high cost as the dimensionality of the spaces being learned in increases. We have not fully explored the resource scaling with complexity of the learned space. As discussed in the main text, we believe that ultimately learning the model one element at a time does not reflect the challenge of learning faced by biological systems, which must learn in the context of the whole system. This challenge is addressed by research on neural development, for which Spaun has little to contribute as it stands, and by demonstrating robust learning in a complex, largely developed brain, which Spaun does minimally with the RL task.

In sum, Spaun does not learn in nearly as a sophisticated manner as people do. However, it has elements of robust weight-based learning (in the RL task), short-term adaptation (in the RPM and rapid variable creation tasks), and all elements of Spaun are learnable by biologically plausible mechanisms. Most models in computational neuroscience focus only on this last feature (i.e., learning of relatively local mechanisms). Consequently, we believe that Spaun can provide a platform to greatly improve our understanding of the breadth of learning in neurobiological systems.

The relation between NEF and Spaun

The Neural Engineering Framework (NEF) is described in more detail in S1.2. This set of methods allows the implementation of arbitrary, dynamical vector functions in spiking networks. However, the NEF does not uniquely determine how functions are to be mapped to a neural architecture, or which functions are relevant for understanding how the brain functions.

Consequently, the structure of Spaun is an independent hypothesis about neural organization. That is, Spaun identifes a specific set of functions, and their organization, that exemplify a detailed hypothesis about what functions the brain computes. The NEF is silent on such matters. The more general architecture on which Spaun is based is described in (*21*).

Errors in Spaun

One of the interesting aspects of Spaun is its ability to reproduce the statistics of errors produced in human behaviour. We have not systematically analyzed all sources of error, but it is clear that errors arise from several sources, as in the real brain. These include noise (i.e., unpredicatable fluctuations in neuarl activity), imperfect optimization of various mappings (i.e., the optimization depends on the specific, randomly chosen, tuning curves in any instantiation of the model), interference between concurrent representations (e.g., adding more, similar memories to working memory), the particular tiling of the represented space by neuron tuning curves, and differences in the quality of the input (i.e., variability in the input can change error rates in tasks like classification).

These error rates can be systematically changed by changing the number of dimensions represented at various points in the model, by adding or removing neurons (*42*), or by changing the optimization in various ways (e.g., weighting the optimization). All NEF optimizations are done with an equal weighting across the optimized domain. In addition, we have not characterized the relationship between these variables and behavioral error rates in Spaun in detail. However, (*22*) discusses the relationships between the number of neurons and representation dimensionality and representational quality, computable functions, and dynamic stability in some detail.

Predictions and scaling

Two as yet untested predictions we have derived from the Spaun model are exemplified in Figures S8 and S9 and the surrounding discussion. In addition to these task specific predictions, the architecture of the model includes general expectations about how the system will change as additional tasks are added to the model's repertoire. For instance, as long as the same *cortical* elements are used, only the basal ganglia must be changed to add a new task. As a specific example, the model with and without the RPM task only changes in size by 900 neurons in the basal ganglia and thalamus (which are used to identify the task and help re-structure the information flow to perfom the additional task). In contrast, adding a new cortical component would be much more demanding. For instance, an additional working memory subnetwork would require about 200,000 additional neurons.

Dynamics in Spaun

One of the surprising features of the model is that its dynamics are largely determined by four synaptic parameters (plus the network architecture). Specifically, throughout the model we have set neurotransmitter time constants to be 50ms for NMDA (*56*), 8ms for GABA (*57*), and 10ms for both AMPA (*58*) and dopamine (*59*). These are set by considering empirical data, not parameter fitting. Nevertheless, Spaun is consistent with a wide variety of behavioural (figure S8), population (figure S11), and single cell (table S3 and figure S12) dynamics data. In past work, we have shown that shifting these parameters outside of their empirically consistent ranges will remove this feature of the model (*28*). As a consequence, the dynamics of the full model are highly constrained by these dynamical properties of synapses.

This has been a consistent feature of predecessor models, and has been shown to allow predictions of fMRI dynamics (*60*), and account for cortical (*26*) and subcortical (*14*) single cell dynamics.

This wide variety of detailed timing data available from the model is ripe for comparison to experimental results. Unlike with smaller scale neural models, Spaun predicts such data across many stages of reasonably complex tasks. Only recently has detailed single cell timing data across several areas become available. This kind information provides excellent tests of Spaun's architectural assumptions.

In a more practical vein, the dynamical constraints in Spaun have mandated that stimuli be shown to the model for at least 150ms to be encoded into working memory, which is consistent with a lower limit on human working memory tasks (blanks in between stimuli presentation are an assumption of the model that is important in order to know when the same digit is presented twice in a row). Because we attempt to minimize run times for tasks due to computational costs, we have generally shown the model dynamically regular input. This rhythm can be broken with no negative consequences – other than expected consequences such as working memory decay – for model performance.

This kind of temporal invariance of the model arises from the fact that much of the processing depends on stable, controlled dynamics of various attractor networks. Consequently, the limitations on delay are coupled to the stability of elements such as the working memory. In Spaun working memory decays with a time constant of about 5s. Importantly, no "re-learning" is needed as the presentation dynamics change.

References and Notes

1. H. de Garis, C. Shuo, B. Goertzel, L. Ruiting, A world survey of artificial brain projects, Part I: Large-scale brain simulations. *Neurocomputing* **74**, 3 (2010). doi:10.1016/j.neucom.2010.08.004
2. H. Markram, The blue brain project. *Nat. Rev. Neurosci.* **7**, 153 (2006). doi:10.1038/nrn1848 Medline
3. R. Ananthanarayanan, D. S. Modha, in *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing - SC '07* (Association for Computing Machinery Press, New York, 2007), p. 1.
4. E. M. Izhikevich, G. M. Edelman, Large-scale model of mammalian thalamocortical systems. *Proc. Natl. Acad. Sci. U.S.A.* **105**, 3593 (2008). doi:10.1073/pnas.0712231105 Medline
5. M. Ranzato, Y. Boureau, Y. LeCun, Sparse feature learning for deep belief networks. *Adv. Neural Inf. Process. Syst.* **20**, 1 (2007).
6. G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks. *Science* **313**, 504 (2006). doi:10.1126/science.1127647 Medline
7. T. Orlov, V. Yakovlev, D. Amit, S. Hochstein, E. Zohary, Serial memory strategies in macaque monkeys: Behavioral and theoretical aspects. *Cereb. Cortex* **12**, 306 (2002). doi:10.1093/cercor/12.3.306 Medline
8. B. B. Murdock, TODAM2: A model for the storage and retrieval of item, associative, and serial-order information. *Psychol. Rev.* **100**, 183 (1993). doi:10.1037/0033-295X.100.2.183 Medline
9. W. Schultz, Multiple reward signals in the brain. *Nat. Rev. Neurosci.* **1**, 199 (2000). doi:10.1038/35044563 Medline
10. E. Vasilaki, N. Frémaux, R. Urbanczik, W. Senn, W. Gerstner, Spike-based reinforcement learning in continuous state and action space: when policy gradient methods fail. *PLOS Comput. Biol.* **5**, e1000586 (2009). doi:10.1371/journal.pcbi.1000586 Medline
11. J. Raven, J. Court, *Manual for Raven's Progressive Matrices and Vocabulary Scales* (Harcourt Assessment, San Antonio, TX, 2004).
12. T. Pasternak, J. Bisley, D. Calkins, in *Handbook of Psychology, Biological Psychology*, M. Gallagher, R. J. Nelson, Eds. (Wiley, Hoboken, NJ, 2003), vol. 3, pp. 139–185.
13. E. Todorov, *The Cognitive Neurosciences*, M. S. Gazzaniga, Ed. (MIT Press, Cambridge, MA, 2009).
14. T. Stewart, T. Bekolay, C. Eliasmith, Learning to select actions with spiking neurons in the basal ganglia. *Front. Decis. Neurosci.* **6**, article no. 00002 (2012); 10.3389/fnins.2012.00002.
15. D. MacNeil, C. Eliasmith, Fine-tuning and the stability of recurrent neural networks. *PLoS ONE* **6**, e22885 (2011). doi:10.1371/journal.pone.0022885 Medline
16. A. R. Forbes, An item analyis of the advance matrices. *Br. J. Educ. Psychol.* **34**, 223 (1964). doi:10.1111/j.2044-8279.1964.tb00632.x

17. J. C. Jahnke, Delayed recall and the serial-position effect of short-term memory. *J. Exp. Psychol.* **76**, 618 (1968). doi:10.1037/h0025692 Medline

18. B. A. Dosher, Item interference and time delays in working memory: Immediate serial recall. *Int. J. Psychol.* **34**, 276 (1999). doi:10.1080/002075999399576

19. I. Chaaban, M. R. Scheessele, "Human performance on the USPS database" (Technical Report, Indiana Univ., South Bend, IN, 2007).

20. R. F. Hadley, The problem of rapid variable creation. *Neural Comput.* **21**, 510 (2009). doi:10.1162/neco.2008.07-07-572 Medline

21. C. Eliasmith, *How to Build a Brain: A Neural Architecture for Biological Cognition* (Oxford Univ. Press, New York, 2012).

22. C. Eliasmith, C. H. Anderson, *Neural Engineering: Computation, Representation and Dynamics in Neurobiological Systems* (MIT Press, Cambridge, MA, 2003).

23. B. J. Fischer, J. L. Peña, M. Konishi, Emergence of multiplicative auditory responses in the midbrain of the barn owl. *J. Neurophysiol.* **98**, 1181 (2007). doi:10.1152/jn.00370.2007 Medline

24. J. Conklin, C. Eliasmith, A controlled attractor network model of path integration in the rat. *J. Comput. Neurosci.* **18**, 183 (2005). doi:10.1007/s10827-005-6558-z Medline

25. P. D. Kuo, C. Eliasmith, Integrating behavioral and neural data in a model of zebrafish network interaction. *Biol. Cybern.* **93**, 178 (2005). doi:10.1007/s00422-005-0576-9 Medline

26. R. Singh, C. Eliasmith, Higher-dimensional neurons explain the tuning and dynamics of working memory cells. *J. Neurosci.* **26**, 3667 (2006). doi:10.1523/JNEUROSCI.4864-05.2006 Medline

27. A. Litt, C. Eliasmith, P. Thagard, Neural affective decision theory: Choices, brains, and emotions. *Cogn. Syst. Res.* **9**, 252 (2008). doi:10.1016/j.cogsys.2007.11.001

28. T. C. Stewart, X. Choo, C. Eliasmith, in *Proceedings of the 10th International Conference on Cognitive Modeling*, D. D. Salvucci, G. Gunzelmann, Eds. (Drexel Univ., Philadelphia, 2010), pp. 235–240.

29. T. C. Stewart, T. Bekolay, C. Eliasmith, Neural representations of compositional structures: Representing and manipulating vector spaces with spiking neurons. *Connection Sci.* **23**, 145 (2011).

30. A. P. Georgopoulos, J. T. Lurito, M. Petrides, A. B. Schwartz, J. T. Massey, Mental rotation of the neuronal population vector. *Science* **243**, 234 (1989). doi:10.1126/science.2911737 Medline

31. J. S. Taube, The head direction signal: Origins and sensory-motor integration. *Annu. Rev. Neurosci.* **30**, 181 (2007). doi:10.1146/annurev.neuro.29.051605.112854 Medline

32. N. C. Rust, V. Mante, E. P. Simoncelli, J. A. Movshon, How MT cells analyze the motion of visual patterns. *Nat. Neurosci.* **9**, 1421 (2006). doi:10.1038/nn1786 Medline

33. Y. Tang, C. Eliasmith, in *Proceedings of the 27th International Conference on Machine Learning*, J. Fürnkranz, T. Joachims, Eds. (Omnipress, Madison, WI, 2010), pp. 1055–1062.

34. T. Dewolf, C. Eliasmith, The neural optimal control hierarchy for motor control. *J. Neural Eng.* **8**, 065009 (2011); 10.1088/1741-2560/8/6/065009.

35. C. K. Machens, R. Romo, C. D. Brody, Functional, but not anatomical, separation of "what" and "when" in prefrontal cortex. *J. Neurosci.* **30**, 350 (2010). doi:10.1523/JNEUROSCI.3276-09.2010 Medline

36. M. J. Kane, R. W. Engle, The role of prefrontal cortex in working-memory capacity, executive attention, and general fluid intelligence: an individual-differences perspective. *Psychon. Bull. Rev.* **9**, 637 (2002). doi:10.3758/BF03196323 Medline

37. X. L. Qi *et al*., Comparison of neural activity related to working memory in primate dorsolateral prefrontal and posterior parietal cortex. *Front. Syst. Neurosci.* **4**, 12 (2010). Medline

38. T. A. Plate, *Holographic Reduced Representations* (Center for the Study of Language and Information Publication, Stanford, CA, 2003).

39. X. Choo, thesis, University of Waterloo (2010).

40. L. Ma *et al*., Working memory load modulation of parieto-frontal connections: Evidence from dynamic causal modeling. *Hum. Brain Mapp.* **33**, 1850 (2011).

41. C. A. Seger, C. M. Cincotta, Dynamics of frontal, striatal, and hippocampal systems during rule learning. *Cereb. Cortex* **16**, 1546 (2006). doi:10.1093/cercor/bhj092 Medline

42. D. Rasmussen, C. Eliasmith, A Neural model of rule generation in inductive reasoning. *Top. Cognit. Sci.* **3**, 140 (2011). doi:10.1111/j.1756-8765.2010.01127.x

43. M. L. Kringelbach, The human orbitofrontal cortex: Linking reward to hedonic experience. *Nat. Rev. Neurosci.* **6**, 691 (2005). doi:10.1038/nrn1747 Medline

44. M. D'Esposito, B. R. Postle, D. Ballard, J. Lease, Maintenance versus manipulation of information held in working memory: An event-related fMRI study. *Brain Cogn.* **41**, 66 (1999). doi:10.1006/brcg.1999.1096 Medline

45. M. Weinrich, S. P. Wise, K. H. Mauritz, A neurophysiological study of the premotor cortex in the rhesus monkey. *Brain* **107**, 385 (1984). doi:10.1093/brain/107.2.385 Medline

46. S. Cordes, R. Gelman, C. R. Gallistel, J. Whalen, Variability signatures distinguish verbal from nonverbal counting for both large and small numbers. *Psychon. Bull. Rev.* **8**, 698 (2001). doi:10.3758/BF03196206 Medline

47. T. K. Landauer, Rate of implicit speech. *Percept. Mot. Skills* **15**, 646 (1962). doi:10.2466/pms.1962.15.3.646

48. L. E. Krueger, Reconciling Fechner and Stevens: Toward a unified psychophysical law. *Behav. Brain Sci.* **12**, 251 (1989). doi:10.1017/S0140525X0004855X

49. J. F. Sowa, *Conceptual Structures: Information Processing in Mind and Machine* (Addison-Wesley, Reading, MA, 1984).

50. W. Zhang, S. J. Luck, Discrete fixed-resolution representations in visual working memory. *Nature* **453**, 233 (2008). doi:10.1038/nature06860 Medline

51. S. Shinomoto, K. Shima, J. Tanji, Differences in spiking patterns among cortical neurons. *Neural Comput.* **15**, 2823 (2003). doi:10.1162/089976603322518759 Medline

52. J. K. H. Tang *et al.*, Neuronal firing rates and patterns in the globus pallidus internus of patients with cervical dystonia differ from those with Parkinson's disease. *J. Neurophysiol.* **98**, 720 (2007). doi:10.1152/jn.01107.2006 Medline

53. B. Bobier, thesis, University of Waterloo (2011).

54. B. P. Tripp, C. Eliasmith, Neural populations can induce reliable postsynaptic currents without observable spike rate changes or precise spike timing. *Cereb. Cortex* **17**, 1830 (2007). doi:10.1093/cercor/bhl092 Medline

55. T. Bekolay, thesis, University of Waterloo (2011).

56. S. Hestrin, P. Sah, R. A. Nicoll, Mechanisms generating the time course of dual component excitatory synaptic currents recorded in hippocampal slices. *Neuron* **5**, 247 (1990). doi:10.1016/0896-6273(90)90162-9 Medline

57. A. Gupta, Y. Wang, H. Markram, Organizing principles for a diversity of GABAergic interneurons and synapses in the neocortex. *Science* **287**, 273 (2000). doi:10.1126/science.287.5451.273 Medline

58. N. Spruston, P. Jonas, B. Sakmann, Dendritic glutamate receptor channels in rat hippocampal CA3 and CA1 pyramidal neurons. *J. Physiol.* **482**, 325 (1995). Medline

59. D. C. Rotaru, D. A. Lewis, G. Gonzalez-Burgos, Dopamine D1 receptor activation regulates sodium channel-dependent EPSP amplification in rat prefrontal cortex pyramidal neurons. *J. Physiol.* **581**, 981 (2007). doi:10.1113/jphysiol.2007.130864 Medline

60. T. Stewart, C. Eliasmith, in *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, L. Carlson, C. Hölscher, T. Shipley, Eds. (Cognitive Science Society, Austin, TX, 2011), pp. 656–661.

61. D. L. Ringach, Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *J. Neurophysiol.* **88**, 455 (2002). Medline

62. B. Pesaran, J. S. Pezaris, M. Sahani, P. P. Mitra, R. A. Andersen, Temporal structure in neuronal activity during working memory in macaque parietal cortex. *Nat. Neurosci.* **5**, 805 (2002). doi:10.1038/nn890 Medline

63. D. J. Felleman, D. C. Van Essen, Distributed hierarchical processing in the primate cerebral cortex. *Cereb. Cortex* **1**, 1 (1991). doi:10.1093/cercor/1.1.1-a Medline

64. Y. Liu, B. Jagadeesh, Neural selectivity in anterior inferotemporal cortex for morphed photographic images during behavioral classification or fixation. *J. Neurophysiol.* **100**, 966 (2008). doi:10.1152/jn.01354.2007 Medline

65. A. M. Owen, Working memory: Imaging the magic number four. *Curr. Biol.* **14**, R573 (2004). doi:10.1016/j.cub.2004.07.016 Medline

66. E. Hoshi, Functional specialization within the dorsolateral prefrontal cortex: A review of anatomical and physiological studies of non-human primates. *Neurosci. Res.* **54**, 73 (2006). doi:10.1016/j.neures.2005.10.013 Medline

67. R. L. Albin, A. B. Young, J. B. Penney, The functional anatomy of basal ganglia disorders. *Trends Neurosci.* **12**, 366 (1989). doi:10.1016/0166-2236(89)90074-X Medline

68. A. Nambu, H. Tokuno, M. Takada, Functional significance of the cortico-subthalamo-pallidal "hyperdirect" pathway. *Neurosci. Res.* **43**, 111 (2002). doi:10.1016/S0168-0102(02)00027-5 Medline

69. M. A. A. van der Meer, A. Johnson, N. C. Schmitzer-Torbert, A. D. Redish, Triple dissociation of information processing in dorsal striatum, ventral striatum, and hippocampus on a learned spatial decision task. *Neuron* **67**, 25 (2010). doi:10.1016/j.neuron.2010.06.023 Medline

70. K. Gurney, T. J. Prescott, P. Redgrave, A computational model of action selection in the basal ganglia. I. A new functional anatomy. *Biol. Cybern.* **84**, 401 (2001). doi:10.1007/PL00007984 Medline

71. J. R. Hollerman, W. Schultz, Dopamine neurons report an error in the temporal prediction of reward during learning. *Nat. Neurosci.* **1**, 304 (1998). doi:10.1038/1124 Medline

72. T. Gisiger, M. Boukadoum, Mechanisms gating the flow of information in the cortex: What they might look like and what their uses may be. *Frontiers in Computational Neuroscience* **5**, 1 (2011). doi:10.3389/fncom.2011.00001 Medline

73. U. Halsband, N. Ito, J. Tanji, H.-J. Freund, The role of premotor cortex and the supplementary motor area in the temporal control of movement in man. *Brain* **116**, 243 (1993). doi:10.1093/brain/116.1.243 Medline

74. E. Todorov, in *Progress in Motor Control III*, M. Latash, M. Levin, Eds. (Human Kinetics, Chicago, 2004), chap. 6, pp. 125–166.