Supplementary Information: Synaptic turnover promotes efficient learning in bio-realistic spiking neural networks

1st Nikos Malakasis*
School of Medicine, *UoC* and *IMBB*, *FoRTH*
Heraklion, Greece
nikos.malakasis@tum.de

2nd Spyridon Chavlis
IMBB, *FoRTH*
Heraklion, Greece
schavlis@imbb.forth.gr

3rd Panayiota Poirazi†
IMBB, *FoRTH*
Heraklion, Greece
poirazi@imbb.forth.gr

I. Supplementary methods

A. Datasets

Our model was evaluated on two datasets. Initial experiments were conducted on a custom shape dataset that we had control over, and then, we solidified our claims by utilizing MNIST [3], a well-known machine learning benchmark dataset.

1) Shape Dataset: The shape dataset consists of 20x20 pixel images that contain a rectangle at the top left denoting class 1 and a disk at the bottom right for class 2 (Fig. 3B). The disks and the rectangles consist of bright pixels at maximum intensity on their perimeter and pixels with random intensity inside. The total amount of images generated per class was 1,000 for training and 200 for testing. The task's difficulty (i.e., easy, medium, hard) was determined by two variables, the intraclass stability and the interclass overlap. Stability corresponds to how much the mass center of a shape could move from image to image within the same class, negatively correlated with the task difficulty level, i.e., making the task easier as it increases since it makes the data per class less variable. Overlap, on the other hand, is a metric corresponding to the amount of potential overlap between the two classes, making the task harder as it increases since it increases the similarity between the two classes. In this work, we used three different combinations of stability and overlap to generate shape datasets. The easy task corresponds to overlap equals zero and stability equals one, the medium task to slight overlap and stability, and the hard task to adequate overlap and low stability (Fig. 3B), for the means per class depicted as images). For every simulation, a new shape dataset is generated to increase the robustness of our results.

This work was supported by NIH (1R01MH124867-01) the European Commission (H2020-FETOPEN-2018-2019-2020-01, FET-Open Challenging Current Thinking, NEUREKA GA-863245), and the Einstein Foundation Berlin, Germany (visiting fellowship, EVF-2019-508)
*: Now in School of Life Sciences Weihenstephan, Technical University of Munich, Munich, Germany
†: corressponding author

2) MNIST handwritten digits: The MNIST (Modified National Institute of Standards and Technology database) is a well-known handwritten digit dataset widely used to benchmark machine learning models. It comprises 70,000 28x28 pixel images of handwritten digits from 0 to 9. The dataset is already split into a training set containing 60,000 images and a test set containing 10,000 images.

B. Neuronal Model

The network consists of excitatory and inhibitory neuronal populations, modeled as two-stage integrators8, based on experimental data primarily from the hippocampal CA1 area. Neurons consist of a somatic spiking unit and independent dendritic subunits capable of nonlinear synaptic integration, dendritic spike initiation, and compartmentalized plasticity [4]. For model parameters see **Table I**.

Synaptic input in each dendritic subunit is integrated and then passed to the somatic compartment (1).

$$\tau_b \frac{dV_b}{dt} = \sum_{i,j} w_j E_{syn} \delta(t - t_{i,j}) - V_b \tag{1}$$

where $t_{i,j}$ denotes the time i of incoming spikes in synapse j, w_j the weight of j-th synapse, τ_b the dendritic membrane time constant, E_{syn} the unitary excitatory postsynaptic potential (EPSP), V_b the b-th dendritic branch voltage, and $\delta(\cdot)$ corresponds to the Dirac delta function.

A depolarization component models the backpropagating action potential V_b^{BP}, which is added to all dendritic subunits connected to the somatic compartment. V_b^{BP} is modeled by an exponential decay (2),

$$V_b^{BP}(t) = E_b^{BP} \exp\left(-\frac{t}{\tau_b^{BP}}\right) \tag{2}$$

where E_b^{BP} is the peak of the backpropagating depolarization and τ_b^{BP} is the time constant of the backpropagating action potential.

The back-propagating action potential V_b^{BP} is summed with V_b to determine the total depolarization of the dendrite, V_d. When the sum of $V_b + V_b^{BP}$, exceeds the dendritic spike generation threshold, θ_{dspike}, a dendritic spike is generated,

which causes the voltage of the corresponding subunit, V_b, to rise instantaneously to V_{dspike} (3).

$$\begin{aligned} &V_d = V_b + V_b^{BP} \\ &\text{if } V_d > \theta_{dspike} \text{ then} V_d := V_{dspike} \end{aligned} \tag{3}$$

The voltage response of the somatic subunit and its spiking output is modeled by an integrate-and-fire point unit with adaptation [5] (4).

$$\begin{aligned} C\frac{dV}{dt} &= -g_L(V - E_L) - g_w(V - E_K) + I_{syn} \\ \tau_w\frac{dg_w}{dt} &= \alpha_w\delta(t - t_{spike}) - g_w \end{aligned} \tag{4}$$

where C denotes the somatic membrane capacitance, g_L the leak conductance, E_L the somatic membrane resting potential, g_w the conductance of the afterhyperpolarization (AHP) current, E_K the potassium reversal potential, as AHP is a potassium-specific channel, and I_{syn} denotes the total synaptic current soma receives. Somatic spiking and the following reset occur when the somatic voltage crosses a threshold V_T. The τ_w is the adaptation time constant, α_w is the quantal increase of g_w after a somatic spike occurs at time t_{spike}. The time constant τ_w can take two values corresponding to the neuron's high and low excitability levels.

The synaptic input I_{syn} to the somatic compartment is calculated based on the total dendritic synaptic current and the inhibition at the soma (5)

$$I_{syn}(t) = g_{syn}\sum_n V_{b,n}(t) - IPSC(t) \tag{5}$$

where g_{syn} corresponds to the dendritic coupling constant, $V_{b,n}$ to the voltage of the n-th branch, and $IPSC$ to the inhibitory input that the neuron receives.

C. Synaptic Plasticity

1) Functional Plasticity: Learning in the model occurs according to the synaptic tag and capture plasticity model, which requires both synaptic tagging and the availability of protein-related proteins (PRPs) for stable strengthening and weakening of synapses [6]. When a postsynaptic branch receives presynaptic inputs, it leads to calcium influx. A certain amount of calcium ions accumulated in the respective branches at the end of a pattern presentation event. A synaptic tag is generated based on the individual calcium ion levels, leading to long-term depression (LTD) of the corresponding synapse if the levels are low or long-term potentiation (LTP) if the levels are high. Once the synapses are tagged, plasticity-related proteins are produced that are consolidated together with the synaptic tags into the existing synaptic weights in the timespan of a couple of hours, leading to the synaptic weight update.

2) Calcium modeling: Calcium is critical to this plasticity mechanism, as it induces synaptic tags and PRP synthesis. The total calcium influx during the presentation of a pattern to a synapse determines the calcium Ca_{syn}, which models the calcium concentration of each synapse. Each incoming synaptic spike causes a step increase of calcium, which depends nonlinearly on the local depolarization of the dendritic branch where the synapse resides. We assume that calcium influx upon arrival of a presynaptic spike ΔCa_{syn} is primarily through NMDA receptors [7], and is, thus, dependent sigmoidally on the dendritic membrane voltage (6).

$$\Delta\text{Ca}_{syn} = \alpha_{\text{Ca}}\frac{1}{1 + \exp\left(-\frac{V_d - V_{1/2}}{q_V}\right)} \tag{6}$$

where α_{Ca} denotes the maximum calcium ion influx, $V_{1/2}$ and q_V denote the midpoint and the steepness of the sigmoid function, and V_d corresponds to the total dendritic membrane voltage. $V_{1/2} = 30$ mV and $q_V = 5$ mV are held constant for all simulations.

3) Plasticity Related Proteins Production: It has been shown in multiple plasticity studies that both dendritic [8], [9] and somatic [10], [11] protein synthesis is essential for synaptic weight consolidation, separately or even both at the same time [12]. The previously published model incorporated three different strategies for plasticity-related protein synthesis, where either somatic solely, dendritic solely, or both were taking place. In the current model, only the somatic PRP synthesis model was used. PRP synthesis initiation was modeled as an all-or-none phenomenon. When the calcium level of a dendritic branch, which is equal to the sum of all of its dendritic calcium levels, passes a threshold P_{soma}, a PRP transient was generated as an alpha function (7).

$$PRP_{soma} = H(t - k_P)\left(\frac{t - k_P}{q_P}\right)\exp\left(1 - \frac{t - k_P}{q_P}\right) \tag{7}$$

where q_P denotes the steepness of the function, k_P the shift in time, i.e., delay, and $H(\cdot)$ denotes the Heaviside step function. $k_P = 20$ minutes and $q_P = 30$ minutes for all simulations in the study.

4) Synaptic Tag Generation and Consolidation: As previously mentioned, the synaptic tag's strength and sign are determined according to the Calcium Control Model [13], thus lower levels of Ca^{2+} cause LTD, while higher levels cause LTP (8). The synaptic tag does not immediately alter the synapse's weight but only after capturing PRPs, which are required for consolidation [14].

$$\text{sTag}(x) = \sigma(100x - 35)) - \sigma(190x - 38) \tag{8}$$

where x denotes the Ca^{2+} concentration and $\sigma(\cdot)$ the sigmoid function, i.e., $\sigma(z) = 1/(1 + \exp(-z))$. Synaptic tags in the model decay exponentially with a time-constant $\tau_{\text{sTag}} = 1$ hour (9).

$$\frac{d\text{sTag}}{dt} = -\frac{\text{sTag(x)}}{\tau_{\text{sTag}}} \tag{9}$$

The synaptic tags are consolidated into synaptic weights over time (10).

$$\Delta w = \eta_s \cdot \alpha_s \cdot \text{sTag}(t) \cdot \text{PRP}(t) \tag{10}$$

where η_s is the learning rate of the synapse, α_s the rate of synaptic tag consolidation, sTag the value of the synaptic tag at time t, either positive or negative, and PRP to the PRP levels at time t. The weights are clipped in the range $[0, 1]$.

5) Dynamic Excitability: Learning has been shown to increase the excitability of neurons participating in forming a given memor [15]–[20]. On the other hand, neurons with increased excitability are more likely to participate in forming a new memory engram [17], [21], [22]. The activation of the transcription factor CREB (cAMP Response Element-Binding Protein) has also been found to modulate the excitability of neurons [23], [24] through the reduction of the AHP current [17], [22]. Therefore, it has been suggested that learning makes cells more amenable to be recruited in future learning events by activating CREB [19], [25]–[27]. Finally, it has been proposed that CREB may also induce the downstream activation of its repressors [17], [19], which would reduce excitability after a certain period, thus creating a time window of increased neuronal activity excitability. Increased excitability is simulated through the transient reduction of the AHP current in the neurons in which PRP synthesis is triggered for approximately 12 hours after a learning event [2].

6) Local Adaptive Learning Rate: A feature of biological synapses is that as they grow and become more stable, their growth rate drops [28]–[32]. This is incorporated into the network in the form of an adaptive and synapse-specific learning rate that is dependent on the synaptic weight value w_s. $\eta_s(w_s)$ corresponds to the learning rate of synapse s with weight w_s and follows a reversed sigmoidal function (11).

$$\eta_s(w_s) = \frac{\eta_{max} - \eta_{min}}{1 + \exp\left(\frac{w_s - w_{1/2}}{q_\eta}\right)} \tag{11}$$

where η_{min} and η_{max} are the minimum and maximum learning rate values, respectively, q_η and $w_{1/2}$ denote the steepness and midpoint of the reversed sigmoidal, respectively.

7) Biologically Inspired Stopping Criterion: Larger spines have been observed to be more stable than smaller ones, leading to the hypothesis that the large ones should play a role in memory formation [1], [28], [29], [31]. A stopping criterion was implemented that terminates training when a percentage of synaptic spines becomes larger than a threshold, assuming that memory has already been formed under such circumstances. A synapse s is characterized by its weight w_s or its learning rate η_s.

D. Structural Plasticity

Synaptic turnover is a structural plasticity mechanism that involves synaptic pruning and synaptogenesis. After a given number of iterations N_{turn}, plastic synapses with a weight smaller than the threshold θ_{turn}, are pruned. Then, synaptogenesis takes place, where synapses equal to the number of pruned synapses are born between random presynaptic input neurons and postsynaptic neuron dendrites, with weight values pulled from a uniform distribution in the range $[w_{min}, w_{max}]$.

E. The Network

The network consists of one input layer and a hidden/output layer. The input layer is connected to the hidden one sparsely, and learning occurs with the help of a class-specific learning signal during training. The learning signal is silenced during the testing phase, and the output evaluation occurs based on a majority rule.

1) Input Translation and Transmission: To use the datasets mentioned above as inputs for our spiking neural network, they are first flattened into a $1D$ vector. Then, they have to be translated into spike trains. This was done by translating pixel intensity into firing rates (12) and then generating a periodic spike train per pixel with the corresponding firing frequency. The stimulus duration was set to 4 seconds. For all network related parameters see **Table II**.

$$f_n = f_{max} \cdot \frac{PI_n}{PI_{max}} \tag{12}$$

where f_n is the firing rate of the n-th input neuron, PI_n the pixel intensity of the n-th pixel, and PI_{max} denotes the maximum pixel intensity.

The translated input is then transmitted to the hidden layer in spikes through synaptic connections. After the pattern has been presented, a consolidation period of 138 minutes follows. Synaptic tags and plasticity-related proteins are produced and consolidated into synaptic weights, leading to weight updates. After this period, the weights are updated, and the next input pattern is presented.

2) Network Connectivity and Computations: The network consists of an input and a hidden layer. The hidden layer possesses both excitatory pyramidal neurons and inhibitory interneurons. Input neurons are sparsely and randomly connected to the dendrites of the pyramidal neurons of the hidden layer, with initial weights pulled from a uniform distribution in the range $[w_{min}, w_{max}]$ and those are the only plastic weights of the network. For learning to occur, the network is equipped with a supervised mechanism as a teaching signal specific to each class. Each teaching signal targets a distinct subpopulation of pyramidal neurons in the hidden layer and fires with a firing rate f_{tn}. This ensures that those neurons in the hidden layer that receive inputs both from the input layer and the learning signal will participate in the learning procedure of the presented pattern, leading to the formation of a memory engram. In this project, each task is a binary classification task. Thus, the teaching signal neurons and the corresponding pyramidal subpopulations were two. Besides pyramidal neurons, interneurons are also represented in the hidden layer. The network possesses both dendritic and somatic targeting interneurons. Dendritic targeting interneurons have supralinear dendrites, while somatic targeting ones have sublinear dendrites. In addition to the interneuron classification presented above, interneurons are also split into two functional groups. One group suppresses the network's activity, as the network itself is quite excitable, and the other performs feedback inhibition to further assist with separating the two classes. The former consists of both dendritic and somatic targeting interneurons that randomly receive input from pyramidal neurons in the hidden layer, and, in return, they randomly project back to them. The latter consists of exclusively dendritic targeting interneurons that are grouped in the same manner as pyramidal subpopulations while they

receive input from one of the subpopulations and inhibit the other one.

3) Network Training and Testing: The network's training procedure has already been described more or less. Patterns are presented for 4 seconds each, with consolidation periods in between, until enough weights grow large enough for the stopping criterion to kick in and to stop the training procedure. After training is finished, the model is evaluated in its ability to classify the patterns that it was presented with into its respective classes. To do that, the plastic weights are frozen, and the hidden layer receives input only from the input layer. For every testing pattern presented, the network's decision is determined by a majority rule that compares the number of active neurons in each subpopulation and selects the digit corresponding to the subpopulation with the most active ones. An active neuron in our framework corresponds to a pyramidal neuron that fires with a firing rate higher than 5 Hz.

F. Analysis and Visualization Techniques

1) Representation Maps: We generated representation maps to visualize the representations formed in each subpopulation during learning. We used the weights of the synapses projecting to each subpopulation to generate them. Specifically, for each input node, we summed the outgoing weights towards each subpopulation separately. Then, we rearranged the sums of the weights into $2D$ matrices of the same size as the original input images, resulting in two different representation maps per run, one for every subpopulation.

2) Similarity score: To quantify the quality of our representation maps, we defined a similarity score inspired by the silhouette coefficient used to quantify clustering quality [33]. The similarity score was calculated as a function of the within-class and the across-class distance metrics (13).

$$\text{Similarity Score} = \frac{b-a}{\max(a,b)} \tag{13}$$

where a and b denote the average within-class and across-class distances, respectively.

Firstly, we measure the normalized Euclidean distance of both representation maps to every image in the test set. Then, we calculated the average distances ($\bar{d}_{i,j}$) of the representation maps from the test images they encoded and test images belong to the opposite class (14).

$$\bar{d}_{i,j} = \frac{1}{n_j} \sum_{x \in \mathfrak{P}_j} d(r_i, x) \tag{14}$$

where $d(r_i, x)$ denotes the Euclidian distance between the representation map of class i and a test image x belongs to class j, and n_j is the total number of test images of class j. $\mathfrak{P}_j$ denotes the class j.

The within-class and the across-class distances are calculated as averages across classes, i.e., two classes in the case of the binary classification task (15).

$$a = \frac{\bar{d}_{i,i} + \bar{d}_{j,j}}{2} \text{ and } b = \frac{\bar{d}_{i,j} + \bar{d}_{j,i}}{2} \tag{15}$$

where $\bar{d}_{i,j}$ is the average Euclidean distance of the representation map of subpopulation i to every image in the test set belonging to class j.

G. Computational Resources

The simulations for this project were run on the high-performance computing (HPC) cluster [34] (version 7.0) of the Poirazi Lab, which consists of 624 cores and 3.25 TB shared RAM operating under CentOS Linux distribution. Data analysis was performed in Python (version 3.9) using the NumPy [35], SciPy [36], and Pandas [37] libraries. The plots were generated using Seaborn [38] and Matplotlib [39] libraries, and the statistical analysis was performed using Pingouin library [40].

H. Statistical Analysis

For all standard statistical tests (detailed in figure legends), the significance level α was 0.05. To correct for multiple comparisons, α was divided by the number of tests according to the Bonferroni procedure. Throughout the figures, p values are denoted by *, **, *** for $p < 0.05$, $p < 0.01$, and $p < 0.001$, respectively. To compare performance among different groups (difficulty level $\times$ synaptic turnover), accuracies were analyzed using two-way analysis of variance (ANOVA) followed by an unpaired t-test (two-tailed) with Bonferroni's correction whenever statistical difference was observed for posthoc comparisons.

II. Supplementary Tables

TABLE I
NEURONAL MODELS PARAMETERS

Parameter	Description	value
τ_b	Passive dendritic integration time constant	20 ms
E_{syn}^{pyr}	Maximum unitary PSP of pyramidal cells	4 mV
E_{syn}^{int}	Maximum unitary PSP of interneurons	3 mV
θ_{spike}^{d}	Voltage threshold for dendritic spike	25 mV
V_{spike}^{d}	Dendritic spike max depolarization	50 mV
E_L	Somatic leakage reversal potential	0 mV
θ_{spike}^{s}	Voltage threshold for somatic spike	20 mV
g_{syn}^{pyr}	Dendritic coupling conductance of pyramidal cells	180 nS
g_{syn}^{int}	Dendritic coupling conductance of interneurons	120 nS
τ_{soma}^{pyr}	Somatic time constant for pyramidal cells	30 ms
τ_{soma}^{int}	Somatic time constant for interneurons	10 ms
τ_w	Adaptation time constant[a]	110 ms
α_w	Spike-triggered adaptation conductance	0.18 nS
E_K	Adaptation reversal potential	-10 mV
τ_{bAP}	Back propagating action potential time constant	17 ms
E_{bAP}	Back propagating action potential max amplitude	30 mV

[a] 120 ms for slow adapting neuronal models.

References

[1] Trachtenberg, J. T. *et al.* Long-term in vivo imaging of experience-dependent synaptic plasticity in adult cortex. Nature 420, 788–794 (2002).

TABLE II
NETWORK PARAMETERS

Parameter	Description	value
N_n	Number of neurons in the hidden layer	120
N_b	Number of dendrites per soma	10
N_{pyr}	Number of pyramidal cells	80[a]
N_{int}^{ctrl}	Number of control intereneurons	[b]
N_{int}^{fb}	Number of feedback intereneurons	20
N_{input}	Number of input cells	[c]
$N_{pyr \to int^{ctrl}}$	Number of synapses from pyramidal cells on control interneurons	[d]
$N_{int^{ctrl} \to pyr}$	Number of synapses from control interneurons on pyramidals	[e]
$N_{pyr \to int^{fb}}$	Number of synapses from pyramidal cells on feedback interneurons	160*
$N_{int^{fb} \to pyr}$	Number of synapses from feedback interneurons on pyramidals	160*
$N_{tn \to pyr}$	Number of synapses from teaching neurons on pyramidals	80*
$N_{input \to pyr}$	Number of synapses from input neurons on pyramidals	[f]
f_{input}^{max}	Maximum firing rate of input neurons	[g] Hz
f_{tn}	Firing rate of teaching neurons	[h] Hz

[a] 40 pyramidals per subpopulation.
[b] 10 somatic and 10 dendritic targeting.
[c] 400 for Shapes and 784 for MNIST.
[d] 100 on somatic and 500 on dendritic targeting.
[e] 400 from somatic and $5,000$ from dendritic targeting.
[f] $1,000$ from Shapes and $1,750$ for MNIST.
[g] 30 Hz from Shapes and 25 Hz for MNIST.
[h] 35 Hz from Shapes and 40 Hz for MNIST.
*Per subpopulation.

[2] Kastellakis, G., Silva, A. J., Poirazi, P. Linking memories across time via neuronal and dendritic overlaps in model neurons with active dendrites. Cell Rep. 17, 1491–1504 (2016).
[3] Li Deng. The MNIST database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Process Mag. 29, 141–142 (2012).
[4] Poirazi, P., Brannon, T., Mel, B. W. Pyramidal neuron as two-layer Neural Network. Neuron 37, 989–999 (2003).
[5] Jolivet, R., Rauch, A., Lüscher, H.R., Gerstner, W. Predicting spike timing of neocortical pyramidal neurons by simple threshold models. J. Comput. Neurosci. 21, 35–49 (2006).
[6] Redondo, R. L., Morris, R. G. M. Making memories last: the synaptic tagging and capture hypothesis. Nat. Rev. Neurosci. 12, 17–30 (2011).
[7] Higley, M. J., Sabatini, B. L. Calcium signaling in dendritic spines. Cold Spring Harb. Perspect. Biol. 4, a005686–a005686 (2012).
[8] Huber, K. M., Kayser, M. S., Bear, M. F. Role for rapid dendritic protein synthesis in hippocampal mGluR-dependent Long-term depression. Science (1979) 288, 1254–1256 (2000).
[9] Kang, H., Schuman, E. M. A Requirement for local protein synthesis in neurotrophin-induced hippocampal synaptic plasticity. Science (1979) 273, 1402–1406 (1996).
[10] Nguyen, P. V., Abel, T., Kandel, E. R. Requirement of a critical period of transcription for induction of a late phase of LTP. Science (1979) 265, 1104–1107 (1994).
[11] Frey, U., Krug, M., Brödemann, R., Reymann, K., Matthies, H. Long-term potentiation induced in dendrites separated from rat's CA1 pyramidal somata does not establish a late phase. Neurosci. Lett. 97, 135–139 (1989).
[12] Frey, U., Morris, R. G. M. Synaptic tagging and long-term potentiation. Nature 385, 533–536 (1997).
[13] Shouval, H. Z., Bear, M. F., Cooper, L. N. A unified model of NMDA receptor-dependent bidirectional synaptic plasticity. PNAS 99, 10831–10836 (2002).
[14] Shouval, H. Z., Castellani, G. C., Blais, B. S., Yeung, L. C., Cooper, L. N. Converging evidence for a simplified biophysical model of synaptic plasticity. Biol. Cybern. 87, 383–391 (2002).
[15] Sehgal, M., Ehlers, V. L., Moyer, J. R. Learning enhances intrinsic excitability in a subset of lateral amygdala neurons. Learning Mem. 21, 161–170 (2014).
[16] Oh, M. M., Kuo, A. G., Wu, W. W., Sametsky, E. A., Disterhoft, J. F. Watermaze learning enhances excitability of CA1 pyramidal neurons. J. Neurophysiol. 90, 2171–2179 (2003).
[17] Zhou, Y. *et al.* CREB regulates excitability and the allocation of memory to subsets of neurons in the amygdala. Nat. Neurosci. 12, 1438–1443 (2009).
[18] Frick, A., Magee, J., Johnston, D. LTP is accompanied by an enhanced local excitability of pyramidal neuron dendrites. Nat. Neurosci. 7, 126–135 (2004).
[19] Silva, A. J., Zhou, Y., Rogerson, T., Shobe, J., Balaji, J. Molecular and cellular approaches to memory allocation in neural circuits. Science (1979) 326, 391–395 (2009).
[20] Disterhoft, J. F., Oh, M. M. Learning, aging and intrinsic neuronal plasticity. Trends Neurosci. 29, 587–599 (2006).
[21] Kim, J., Kwon, J.T., Kim, H.S., Han, J.H. CREB and neuronal selection for memory trace. Front. Neural Circuits 7, (2013).
[22] Huang, Y. H. *et al.* CREB modulates the functional output of nucleus accumbens neurons. J. Biol. Chem. 283, 2751–2760 (2008).
[23] Han, M.H. et al. Role of cAMP response element-binding protein in the rat locus ceruleus: regulation of neuronal activity and opiate withdrawal behaviors. J. Neurosci. 26, 4624–4629 (2006).
[24] Dong, Y. *et al.* CREB modulates excitability of nucleus accumbens neurons. Nat. Neurosci. 9, 475–477 (2006).
[25] Rogerson, T. *et al.* Synaptic tagging during memory allocation. Nat. Rev. Neurosci. 15, 157–169 (2014).
[26] Benito, E., Barco, A. CREB's control of intrinsic and synaptic plasticity: implications for CREB-dependent memory models. Trends Neurosci. 33, 230–240 (2010).
[27] Silva, A. J., Kogan, J. H., Frankland, P. W., Kida, S. CREB and Memory. Annu. Rev. Neurosci. 21, 127–148 (1998).
[28] Akbergenova, Y., Cunningham, K. L., Zhang, Y. V, Weiss, S., Littleton, J. T. Characterization of developmental and molecular factors underlying release heterogeneity at Drosophila synapses. Elife 7, (2018).
[29] Matsuzaki, M., Honkura, N., Ellis-Davies, G. C. R., Kasai, H. Structural basis of long-term potentiation in single dendritic spines. Nature 429, 761–766 (2004).
[30] Grutzendler, J., Kasthuri, N., Gan, W.B. Long-term dendritic spine stability in the adult cortex. Nature 420, 812–816 (2002).
[31] Kasai, H., Matsuzaki, M., Noguchi, J., Yasumatsu, N., Nakahara, H. Structure–stability–function relationships of dendritic spines. Trends Neurosci. 26, 360–368 (2003).
[32] Matus, A. Growth of dendritic spines: a continuing story. Curr. Opin. Neurobiol. 15, 67–72 (2005).
[33] Rousseeuw, P. J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 20, 53–65 (1987).
[34] Papadopoulos, P. M., Katz, M. J., Bruno, G. NPACI Rocks: tools and techniques for easily deploying manageable Linux clusters. Concurr. Comput. 15, 707–725 (2003).
[35] Harris, C. R.*et al.* Array programming with NumPy. Nature 585, 357–362 (2020).
[36] Virtanen, P. *et al.* SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat. Methods 17, 261–272 (2020).
[37] McKinney, W. Data structures for statistical computing in Python. In Proceedings of the 9th python in science conference (SCIPY 2010) (2010).
[38] Waskom, M. seaborn: statistical data visualization. J. Open Source Softw. 6, 3021 (2021).
[39] Hunter, J. D. Matplotlib: A 2D graphics environment. Comput. Sci. Eng. 9, 90–95 (2007).
[40] Vallat, R. Pingouin: statistics in Python. J. Open. Source. Softw. 3, 1026 (2018).