BRAIN INSIGHTS IMPROVE RNNS' ACCURACY AND ROBUSTNESS FOR HIERARCHICAL CONTROL OF CONTINUALLY LEARNED AUTONOMOUS MOTOR MOTIFS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study the problem of learning dynamics that can produce hierarchically organized continuous outputs consisting of the flexible chaining of re-usable motor motifs from which complex behavior is generated. Can a motif library be efficiently and extendably learned without interference between motifs, and can these motifs be chained in arbitrary orders without first learning the corresponding motif transitions during training? This requires (i) parameter updates while learning a new motif that do not interfere with the parameters used for the previously acquired ones; and (ii) successful motif generation when starting from the network states reached at the end of any of the other motifs, even if these states were not present during training (a case of out-of-distribution generalization). We meet the first requirement by designing recurrent neural networks (RNNs) with specific architectures that segregate motif-dependent parameters (as customary in continual learning works), and try a standard method to address the second by training with random initial states. We find that these standard RNNs are very unreliable during zero-shot transfer to motif chaining. We then use insights from the motor thalamocortical circuit, featuring a specific module that shapes motif transitions. We develop a method to constrain the RNNs to function similarly to the thalamocortical circuit during motif transitions, while preserving the large expressivity afforded by gradient-based training of non-analytically tractable RNNs. We then show that this thalamocortical inductive bias not only acts in synergy with gradient-descent RNN training to improve accuracy during in-training-distribution motif production, but also leads to zero-shot transfer to new motif chains with no performance cost. Besides proposing an efficient, robust and flexible RNN architecture, our results shed new light on the function of motor preparation in the brain.

1 INTRODUCTION AND RELATION TO OTHER WORKS

Animals have the remarkable ability to efficiently learn and compose elaborate continuous behaviors, which often relies on the flexible chaining of 'motifs' - reproducible bouts of behavior - in response to hierarchical commands (Zimnik & Churchland, 2021; Geddes et al., 2018; Merel et al., 2019b). The mechanisms behind this are however not fully understood, and engineering controllers for dynamical systems that could perform such complex and structured continuous behaviors in the real world has been a long-standing challenge of robotics (Brooks, 1986; Prescott et al., 1999; Merel et al., 2019b). Such tasks involve two different computational operations: first, at coarse temporal intervals, the flexible selection of discrete and abstract action commands by a 'high-level controller'; and, second, the transmission of each abstract command to a 'lower-level controller' that continuously produces a corresponding command - a 'motif' - for the motor effector.

Many lines of work have used artificial neural networks (ANNs) to solve the first - discrete - computational operation (notably using tools from deep reinforcement learning, e.g. (Merel et al., 2019a; Frans et al., 2018; Dennis et al., 2020)) and have leveraged the ability of ANNs to perform well when using a rich training set that includes many motif sequences (Frans et al., 2018; Dennis et al., 2020; OpenAI et al., 2021; Xu et al., 2020). However, to the best of our knowledge, in the context of the second computational operation, there is a gap in the literature about how to design ANN mechanisms that enable zero-shot transfer to performing new continuous motif sequences from a library of

independently learned motifs. Here we will focus on this latter task, which notably highlights a remarkable human skill, as we can - for instance - learn to pronounce a new word and then immediately include it in arbitrary sentences. While this biological relevance has led the neuroscience community to start investigating related questions, this literature has focused on a hand-designed 'bottom-up' approach. This approach studies how networks can function in spite of strong constraints introduced e.g. to mimic the brain activity patterns or connectivity, and/or to ensure analytical tractability (Kao et al., 2020; Logiaco et al., 2019; Sussillo et al., 2015; Zimnik & Churchland, 2021; Ijspeert et al., 2013; Kulvicius et al., 2012). Importantly, this approach cannot determine whether specific network features are more generally advantageous for solving a task - instead, with this bottom-up approach, these chosen features could reflect constraints that arose through random evolutionary idiosyncracies, or could become irrelevant for networks that cannot be designed through analytical insights but that can still be successfully trained with modern machine learning algorithms. Overcoming these limitations has implications for engineering and can provide new neuroscientific insight.

To address this knowledge gap, here, we use Recurrent Neural Networks (RNNs) whose computational power is not arbitrarily constrained. An RNN is a type of ANN which is a generic dynamical system, and which therefore naturally fits the desired characteristics of motor outputs. Consequently, ANNs are indeed regularly used for tasks requiring the production of continuous outputs, including in the context of robotics (Wyffels & Schrauwen, 2009; Sussillo & Abbott, 2009; Tani, 2003; Liu et al., 2019; Merel et al., 2019a; Maheswaranathan et al., 2019). We will examine the ability of RNNs to *(i)* independently learn motor motifs in order to to build a continuously expandable motif library; and *(ii)* flexibly chain motifs in arbitrary orders (Fig.. 1a and see Appendix A.1 for a formal definition). In order to better dissect the mechanisms by which RNNs can fail or succeed at this task, we focus on the purest form of motor control through dynamics: the production of trajectories without needing the external anchor of a time-dependent sensory input. We refer to this as autonomous control, which is especially relevant when sensory input is too unreliable (Yeo et al., 2016; Shenoy et al., 2013; Brembs, 2021), but also in other cases such as the above-mentioned speech production. We will show that while it is possible to engineer RNNs to independently, extendably and efficiently learn to produce single motifs in response to discrete input commands, these RNNs are limited in their generalization ability during improvisation of motif sequences. We will then use insights from the mammalian thalamocortical motor system - notably the presence of a motor preparation phase before each motif (Zimnik & Churchland, 2021; Nashef et al., 2021; Logiaco et al., 2019). We will show that weaving in these insights into performance-optimized RNNs leads to both improved motif production accuracy (through a positive synergy with single motif gradient-decent training) and excellent robustness during generalization to motif sequencing.

2 TASK AND ARCHITECTURE DESIGN

Here, we study the ability of RNNs to fulfill the first requirement of our task: the ability to acquire an extendable library of autonomous motifs. However, before describing these analyses, we want to clarify how we chose the motifs that the RNNs have to learn. First, we chose motifs of long durations – on the order of a thousand timesteps – so that they strongly leverage the above-mentioned autonomous capabilities of RNNs. Second, because we are interested in assessing the relative expressive power of the RNNs we study, we have designed two different types of motor motifs so that they constitute two 'difficulty' levels for the RNNs. Following recent analytical approaches for studying RNN dynamics (Schuessler et al., 2020b;a; Logiaco et al., 2019), we characterize the difficulty of motifs as the number of certain basis functions - complex exponentials that act similarly as different frequencies of a Fourier transform - needed to approximate a motif well through linear combination. Therefore, we define a set of oscillatory motifs that are relatively easy to produce, and a set of more difficult 'step' motifs (Fig. 1d,f-g, see Appendix A.2 for the full list of motifs used in this paper). We train motifs using gradient descent – specifically, ADAM (Kingma & Ba, 2015)) – with a time discretization of $dt = 0.1\tau$ and $\tau = 1$. Our objective function is the mean square error between desired and actual output. We tested various parameters of ADAM, we identified the following as yielding successful training in our setting: learning rate $= 10^{-3} - 10^{-4}$, $\beta_1 = \beta_2 = 0.5$, and $\epsilon = 10^{-8}$ (which performed better than default values in the most difficult cases).

We will now study how RNNs can meet the requirement to learn a new motif without 'catastrophic interference' with the memory of previously learned motifs - which relates to the literature on continual learning (Kirkpatrick et al., 2017; Parisi et al., 2019; Yoon et al., 2020; Farajtabar et al., 2019).

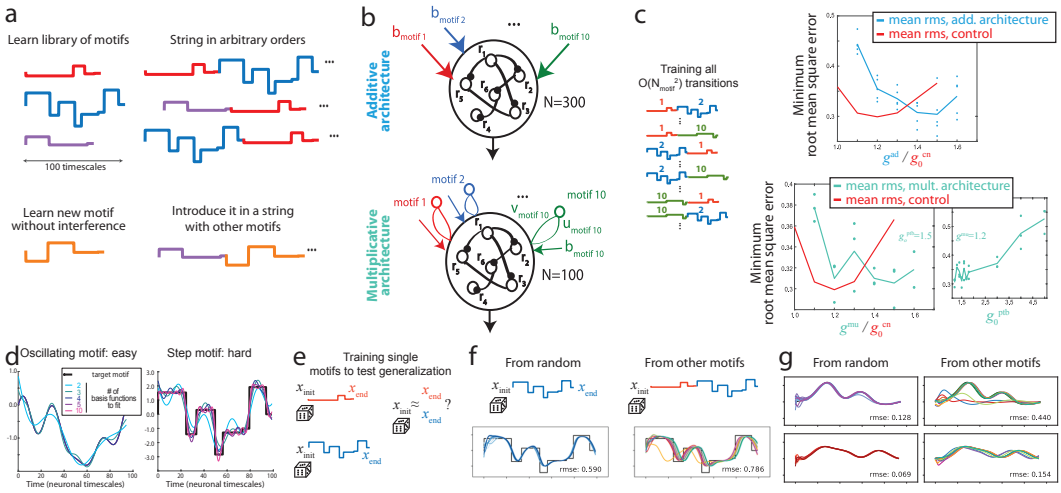

Figure 1: **Task and candidate networks.** **a**) Task. Left: learning an extendable library of motifs without interference. Right: without additional training, stringing the motifs into chains with arbitrary orders. **b**) Additive and multiplicative architectures, who may succeed in the task in **a** because they segregate parameters into motif-specific sets (schematized in colors) while benefiting from fixed shared recurrent and readout weights (schematized in black). **c**) Minimum root mean square error over training with overcomplete training set, depending on the gain hyperparameters g^{ad}, g^{mu}, and g_o^{ptb}. Dots are individual networks, the line is the average. In red, we show the mean minimum root mean square error in the control architecture (averaged over five individually trained networks for each g_0^{cn}). **d**) Examples easy oscillatory motif and hard step motif; fitting the latter requires a larger number of basis functions (complex exponentials) of varied oscillation frequencies. **e**) 'Classical' strategy to promote zero-shot transfer in order to produce a chain of motifs: train ANNs to produce the motifs starting from random initial network states $\mathbf{x}_{\text{init}}$ that emulate the variability of network states at the end of motifs $\mathbf{x}_{\text{end}}^{\mu}$ (where μ indexes the motif). **f**) Increased inaccuracy during zero-shot transfer to chains of motifs compared to when starting from random $\mathbf{x}_{\text{init}}$ drawn from the same distribution as during training (additive architecture). **g**) As in **f** but for oscillatory motifs. Note that the target trajectory (black) is buried below the output of the network (colored lines), because the latter is almost perfectly accurate with random $\mathbf{x}_{\text{init}}$.

This line of work emphasizes avoiding interference between gradient updates used to train a network on many sub-tasks, while promoting the re-use of neural resources across sub-tasks (so that the network makes efficient use of its parameters). These will therefore be desired features for the RNNs we will use.

We test whether catastrophic forgetting is an issue in our setup by considering a 'standard' RNN ('control architecture') which *(i)* is fully-tuned, *(ii)* receives a static input to instruct the motif (Maheswaranathan et al., 2019; Sussillo & Abbott, 2009; Tani, 2003), and *(iii)* is trained in a plausible noise-robust regime imposed by random initialization of the internal state. Consistent with the continual learning literature (Kirkpatrick et al., 2017), we indeed find that even though it is possible to learn motifs by simultaneously including all of them in each training batch (Fig. 1c, and see below), learning motifs sequentially instead leads previously learned motifs to be immediately forgotten when learning a new one (not shown). This 'control architecture' – which we will take as a baseline moving forward that we will compare to our modified networks – obeys the following dynamics:

$$\tau\, \dot{\mathbf{x}} = -\mathbf{x} + g_0^{\text{cn}}\, \mathbf{J} \tanh(\mathbf{x}) + \mathbf{b}_\mu .$$

The output $y = \mathbf{w}^\intercal \tanh(\mathbf{x}) = \mathbf{w}^\intercal \mathbf{r}$ is produced through the vector $\mathbf{w}$ that is initialized from a centered Gaussian distribution with std $1/\sqrt{N^{\text{cn}}}$ – where here $N^{\text{cn}} = 50$ is the number of recurrent units of this control network, so that its outputs have appropriate maximal positive or negative magnitude scaling as $\sqrt{N^{\text{cn}}}$. Also, $\mathbf{J}$ is initialized with iid elements taken from a standard Gaussian because previous work has shown that in this case choosing a gain $g_0^{\text{cn}} > 1$ leads to rich dynamical regimes appropriate for complex computations (Sompolinsky et al., 1988; Sussillo & Abbott, 2009; Sussillo et al., 2015; Schuessler et al., 2020b). We performed hyperparameter tuning on g_0.

Table 1: Number of neurons and number of parameters in the different architectures

	Additive	**Multiplicative**	**Control**
# of recurrent units N	300	100	50
# of learned parameters for 10 motifs	3000	3000	3050
# of motif-specific parameters per motif	300	300 (input: 100, loop: 200)	50

We now consider modifications of this control RNN that address this issue of catastrophic forgetting, and focus on possible architectural strategies (Parisi et al., 2019; Merel et al., 2019a;b) as they will more readily allow us to weave in insights from the brain. These strategies rely on segregating the parameters involved in learning different motifs, while favoring efficiency by sharing some weights across motifs. Here, we will share *(i)* fixed readout weights (with the above-mentioned centered Gaussian distribution with std $1/\sqrt{N}$ as this is sufficient to ensure the successful production of our motifs); and *(ii)* the recurrent weights as they can set rich 'baseline dynamics' that can be modulated by some motif-specific weights. We globally adjust the shared recurrent weights through tuning their above-mentioned gain hyperparameter g (a similar alternative could be to pre-tune these recurrent weights to an original set of motifs and to then freeze them, but - as we will see - our chosen approach leads to good results).

First, we consider an 'additive' architecture (Fig. 1b, top). Here, each motif μ is produced in response to a learned input vector $\mathbf{b}_\mu$, leading to the following dynamics for the network activities:

$$\tau\, \dot{\mathbf{x}} = -\mathbf{x} + g^{\text{ad}}\, \mathbf{J} \tanh(\mathbf{x}) + \mathbf{b}_\mu,$$

where the input acts as a motif-specific controller of the dynamics. Note that this occurs because the gradient of the loss with respect to the input weights propagates through the recurrent connections - whereas if different outputs weights would be learned for different motifs (Jaeger, 2007), the dynamics would not be affected by learning.

Second, we consider a 'multiplicative' architecture (Fig. 1b, bottom), that is inspired by both previous machine learning literature (Sutskever et al., 2011; Schuessler et al., 2020b) and the anatomy of the brain's motor system (Guo et al., 2017; Logiaco et al., 2019). Here, each motif μ is produced in response to both a learned input vector $\mathbf{b}_\mu$ and a learned rank-one perturbation of the connectivity $\mathbf{u}_\mu \mathbf{v}_\mu^\intercal$. The latter is equivalent to a loop through an instantaneous 'unit' receiving input from the recurrent network through the weights $\mathbf{v}_\mu$ and feeding back through the weights $\mathbf{u}_\mu$. Here, this motif-specific loop participates to modulating the dynamics (Logiaco et al., 2019), in a way that yields more computational flexibility compared to networks with random feedback weights (Susman et al., 2021). Interestingly, learning the full recurrent weights in randomly initialized RNNs can yield a low-rank weight update (Schuessler et al., 2020b). Therefore, by imposing that the motif-specific learning is restricted to a low-rank weight perturbation, we expect to get close to full-weight learning while enabling segregation of the learned weights per motif. The dynamics are:

$$\tau\, \dot{\mathbf{x}} = -\mathbf{x} + \left(g^{\text{mu}}\, \mathbf{J} + \mathbf{u}_\mu \mathbf{v}_\mu^\intercal\right) \tanh(\mathbf{x}) + \mathbf{b}_\mu,$$

where g^{mu} and $\mathbf{J}$ are defined as for the additive network, and $\mathbf{u}_\mu$ and $\mathbf{v}_\mu$ are each learned and initialized iid from a centered Gaussian with std $g_o^{\text{ptb}}/\sqrt{N}$ (i.e. expected norm g_o^{ptb}).

To test the baseline relative accuracies of the additive, multiplicative and control networks, we trained them on all the possible chains of motifs of length two (excluding repetitions of the same motif) for the ten step motifs. Consistent with the continual learning aspect of our task – which values limiting the number of parameters that need to be tuned and stored per motif – we equalized the number of tunable parameters across architectures (Collins et al., 2017). Interestingly, after optimizing over hyperparameters (i.e., g^{ad}, g^{mu}, g_o^{ptb}, and g_0^{cn}), we found that all networks had similar accuracy (Fig. 1c, reminiscent of (Collins et al., 2017)). This suggests that our strategy of segregating tunable parameters in the additive and multiplicative networks do not lead to drastic decrease in per-tuned-parameter expressivity, while by construction preventing interference when learning motifs sequentially (while the control network suffers from forgetting of previously learned motifs when learning new ones as expected, not shown). On the other hand, we note that the multiplicative architecture, which is closer to models constrained to mimic brain dynamics and architecture (Logiaco et al., 2019), appears to have similar accuracy as the additive network while requiring fewer

neurons. This echoes recent results suggesting that more biologically-plausible object-recognition ANNs tend to have architectures that require less neurons (Nayebi et al., 2021). We now turn to investigate the robustness of these different architectures when improvising motif sequences.

3 BRITTLENESS OF STANDARD RNNS DURING GENERALIZATION

Here, we ask whether RNNs trained on single motifs can produce arbitrary chains of motifs – a form of zero-shot transfer where the RNN's internal state differs between training and testing due to the RNN's memory over several timesteps (a similar network mechanism as in (Lake & Baroni, 2017)).

We evaluate a standard technique used to promote both generalization and noise-robustness in ANNs, that relies on leveraging randomness (e.g. (Vezhnevets et al., 2017; Liu & Hodgins, 2017; Merel et al., 2019a)) during training. Each motif is trained in isolation but with initial network activities selected randomly according to a distribution that approximates the activities at the ends of all other motifs (Fig. 1e). In this scenario, if we can approximate the end-of-motifs distribution well enough and if training succeeds, then all transitions should work with no transition-specific training. Unfortunately, the end-of-motifs distribution is unknown and is training-dependent. However, since we are using networks with large N, Gaussian weights, and a $\tanh$ nonlinearity, a Gaussian should well-approximate the marginal statistics of this unknown distribution (Landau & Sompolinsky, 2018; Rivkind & Barak, 2017). Indeed, we observed that a standard normal is a good choice (Fig. 2a and b, panels (ii)). Hence, we tried training the RNNs to generate the step motifs one at a time when starting from random initial network activities $\mathbf{x}_{\text{init}}$ drawn from a standard normal distribution. We set our hyperparameters to the optima from Fig. 1c ($g^{\text{ad}} = 1.4$, $g^{\text{mu}} = 1.4$ and $g_o^{\text{ptb}} = 1.5$) and otherwise trained as above. For all architectures, training was successful (Fig. 2a and b panel (i), left; Fig. 2c left).

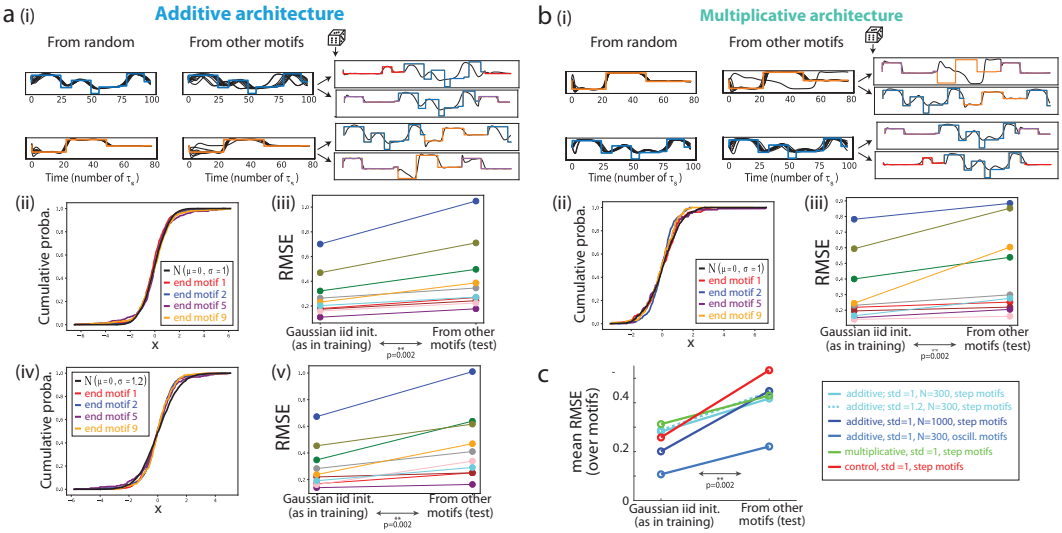

Figure 2: **Motif sequencing by RNNs trained on individual motifs from random initializations**. Step motifs are always used except in panel **c**. **a**) Additive architecture; **b**) multiplicative architecture. Panels (i): Network output (black traces, compared to the desired motif shown in color) when initial conditions $\mathbf{x}_{\text{init}}$ are randomly drawn from a standard Gaussian as during training vs. when transitioning from other motifs (with example full motif chains shown on the right). Panels (ii): After training with $\mathbf{x}_{\text{init}}$ drawn from a standard Gaussian, 'marginal' cumulative distribution of network states $\mathbf{x}_{\text{end}}^{\mu}$ at the end of example motifs. Panels (iii): for each motif, root mean square error taken over 9 random initializations (left) vs. when transitioning from 9 other motifs. P-values from two-sided Wilcoxon signed-rank test. Panels (iv) and (v): same as (ii) and (iii) but when training with $\mathbf{x}_{\text{init}}$ drawn from a Gaussian with standard deviation 1.2. **c**) Average root mean square error over motifs during training conditions vs. during zero-shot transfer, for different architectures, network sizes, motif types, and variance of the random initialization. All comparisons are significant.

We then asked our networks to chain two motifs (the first one starting from random initial activities as in training), and found that the performance of the second motif was substantially degraded compared to the same motif produced when starting from random (Fig. 2a,b panels (i) and (iii); see also Fig. 1g right). Each transition from a given first motif leads to a relatively reproducible output for the second motif because the network variability is low at the end of the first motif. However the second motif's output can vary substantially from the target when following certain first motifs. Moreover, the inaccuracies are not restricted to the start of the second motif; rather, the whole output is either clearly shifted or even more grossly wrong. Note that these issues are also present in the control architectures and are therefore not caused by our strategy of segregating the tuned parameters per motif to tackle catastrophic interference (Fig. 2c, Appendix A.3.5).

Interestingly, the shape of the marginal distribution of the elements of $\mathbf{x}$ at the end of a given motif μ ($\mathbf{x}^{\mu}_{\text{end}}$) was not a good predictor of whether transitions from this motif would lead to worse performance. For instance, in the case of the additive network, the transition from motif 1 to motif 2 leads to poor performance for motif 2 (Fig. 2a (i), top right), even though the marginal distribution of $\mathbf{x}^{1}_{\text{end}}$ at the end of motif 1 appears extremely similar to a standard Gaussian (Fig. 2a (ii), red curve). Conversely the distribution of $\mathbf{x}^{5}_{\text{end}}$ at the end of motif 5 is wider than the standard Gaussian (Fig. 2a (ii), purple curve), but the transition from motif 5 to motif 2 leads to a more accurate output (Fig. 2a (i), second chain from top right). This strongly suggests that the correlations between the activities of different units matter to determine the success of the transitions.

Figs. 2a,b panels (ii) show that the standard deviation of the marginal distribution of the values of $\mathbf{x}^{\mu}_{\text{end}}$ slightly exceeds 1 for some motifs. To ensure that the transition failures we observed when initializing with the standard normal were not due to this mismatch in scale, we retrained our additive network with initial $\mathbf{x}$ values sampled from a Gaussian with std of 1.2 (Fig. 2a panels (iv) and (v)). We observe that the end-of-motif distributions still have standard deviations around 1 for all motifs (Fig. 2a panel (iv)). Despite training with a wider distribution, performance impairment after transitioning still occurred with even slightly greater errors compared to training with a standard normal (Fig. 2a panel (iv) Fig. 2c dashed cyan line).

In addition, we verified that the decreased performance during zero-shot transfer was not a consequence of using our relatively difficult step motifs for which accuracy is limited even when starting from random. Using our oscillatory motifs instead (as in Fig. 1g), we see that performance when starting from random improves significantly, but zero-shot transfer is again unsuccessful with relatively large second motif errors during motif chaining (the light blue line in Fig. 2c, notice that the increase of error during zero-shot transfer is of similar amount as for the step motifs).

Finally, we also investigated whether larger networks would somehow be more regularized in a way that would favor zero-shot transfer performance. However, increasing the network size from $N = 300$ to $N = 1000$ did not improve the performance during zero-shot transfer; instead, as expected, it only improved accuracy when the network was initialized from random as in training (Fig. 2c, cyan line vs. dark blue line).

The failure to robustly transition between motifs when training on single motifs that are randomly initialized is a particular instance of an out-of-distribution generalization limitation, a general feature of ANNs (Russin et al., 2019; D'Amour et al., 2020). Not only is the shape of the true marginal distribution of the values of $\mathbf{x}^{\mu}_{\text{end}}$ at the end of a given motif not exactly matched to the statistics of the Gaussian distribution used during training, but we also neglect the correlations and other higher order interactions between the values of $\mathbf{x}^{\mu}_{\text{end}}$, that are notably dependent on the learned inputs received by the network.

We stress that trivial attempts to solve this issue do not constitute satisfactory solutions. First, explicitly including all transitions between motifs in the training set (Fig. 1c, left) would scale quadratically with the number of motifs and is thus ultimately prohibitive for real-world applications. Further, we have shown that *(i)* subtle variations in the state of the network at the end of different first motifs $\mathbf{x}^{\mu}_{\text{end}}$ can lead to very different outcomes for the performance of a given second motif (Fig. 2a,b); and *(ii)* when transitioning from a given first motif, the performance can be almost flawless for a particular second motif but very poor for another second motif (Appendix A.3.4). This demonstrates subtle-but-potent differences across motifs in both $\mathbf{x}_{\text{end}}$ and in initial dynamics' convergence properties, so that including just a few motif transitions in the training set would not safe-proof the RNNs against large inaccuracies during new sequences. Second, another naive attempt to solve the transition issue

would be to train all motifs from a fixed arbitrary network state, and to implement a hard reset to this state before each motif. However, this strategy *(i)* would not be noise-robust; *(ii)* would introduce sharp discontinuities at transitions in case of discrepancy between two motif's end and start points – and would more generally prevent the RNN's dynamics to use the implicit information about the previous motif's shape encoded into its internal state in order to implement a smooth transition to the next motif; *(iii)* would lead to suboptimal single motif learning because RNN performance depends on identifying, and converging towards, a motif-specific state to start motif production (more on this below (Mohajerin & Waslander, 2019; Kemeth et al., 2021)).

To satisfactorily solve the RNNs' brittleness during performance of new sequences, we will now leverage insights from the mammalian motor thalamocortical circuit. This type of domain-knowledge is particularly promising because human behavior clearly demonstrates that the brain has acquired the necessary inductive biases to perform zero-shot transfer (Zador, 2019).

4 BRAIN-INSPIRED INDUCTIVE BIASES IMPROVE ACCURACY & ROBUSTNESS

Recordings from the brain's motor cortex during flexible sequencing indicate that a period of motor preparation precedes every motif in a sequence (Zimnik & Churchland, 2021). This preparation leads the activity to quickly converge towards a motif-specific pattern (Lara et al., 2018). This motor preparation process involves interactions between the recurrent motor cortex and thalamus (Nashef et al., 2021) – a brain region whose neurons quickly respond to cortical input and then projects back to cortex, forming feedback loops that effectively act as perturbations of the effective cortical connectivity (Kao et al., 2020; Logiaco et al., 2019). In a computationally limited linear regime, previous work has shown that this preparatory perturbation can be linked to the optimal control solution to bring the cortical activity towards the unique motif-specific stable state instructed by an external input provided to cortex (Kao et al., 2020; Logiaco et al., 2019). However, the dynamics of the continuously nonlinear RNNs that we and modern ML practitioners consider (Maheswaranathan et al., 2019; Wyffels & Schrauwen, 2009; Sussillo & Abbott, 2009; Tani, 2003; Liu et al., 2019; Merel et al., 2019a) are richer and qualitatively different from this linear regime, and do not allow setting the parameters through analytics as done in the former neuroscience works (Kao et al., 2020; Logiaco et al., 2019). Indeed, in this limited linear regime, *(i)* single motifs could not be produced in response to a simple static input (Logiaco et al., 2019) – whereas the RNNs can (Fig. 2); *(ii)* spontaneous activity decays to zero, whereas units are spontaneously active in our networks (Fig. 3b, 'prior learning' curves, (Sompolinsky et al., 1988)); and *(iii)* a single stable fixed point exists, while continuously nonlinear RNNs can exhibit multistability (Maheswaranathan et al., 2019). Therefore, it is unclear whether such a preparatory connectivity perturbation can be successfully trained with gradient descent to shape a convergence phase in the more powerful performance-optimized continuously nonlinear RNNs. Also, to draw conclusions about the functionality and applicability of the preparatory connectivity perturbation, it is critical to determine whether it can act in synergy with gradient descent training of the motif-specific parameters in the continuously nonlinear RNNs. To answer these questions, we first design a training protocol for a preparatory connectivity perturbation (that we call 'preparatory module', Fig.. 3a left). This training is done once and for all before learning any motif. Specifically, we want to train the weights $\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal}$ that can shape fast convergence to zero activity in the absence of input in nonlinear RNNs. To do so, we initialized the network activities with standard random initial $\mathbf{x}$ values and trained using the cost function $\sum_t \|\mathbf{r}(t)\|^2$ (Fig. 3b), while the dynamics of our networks obey:

$$\tau\dot{\mathbf{x}} = -\mathbf{x} + (g^x\mathbf{J} + \mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal})\tanh(\mathbf{x})$$

where g^x is g^{ad} or g^{mu} for the additive and multiplicative networks respectively. We initialized the weights $\mathbf{U}_{\text{prep}}$ and $\mathbf{V}_{\text{prep}}$ with centered Gaussian with std $\sqrt{0.05/\sqrt{P*N}}$, with $P = 50$ (though smaller values of P give similar results as long as P/N is more than a few percent).

Then, with the thalamic preparatory modules in hand (whose weights are then fixed), we trained additive and multiplicative networks on our ten individual step motifs with the network activities $\mathbf{x}$ randomly initialized with a standard Gaussian, but starting each motif with a motif preparation period (Fig. 3b). For the additive network, our only modification from before is that we included the thalamic preparatory module in the network dynamics for the first 5τ:

$$\tau\dot{\mathbf{x}} = -\mathbf{x} + (g^{\text{ad}}\mathbf{J} + \mathbb{1}_{t\leq 5\tau}\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal})\tanh(\mathbf{x}) + \mathbf{b}_{\mu}.$$

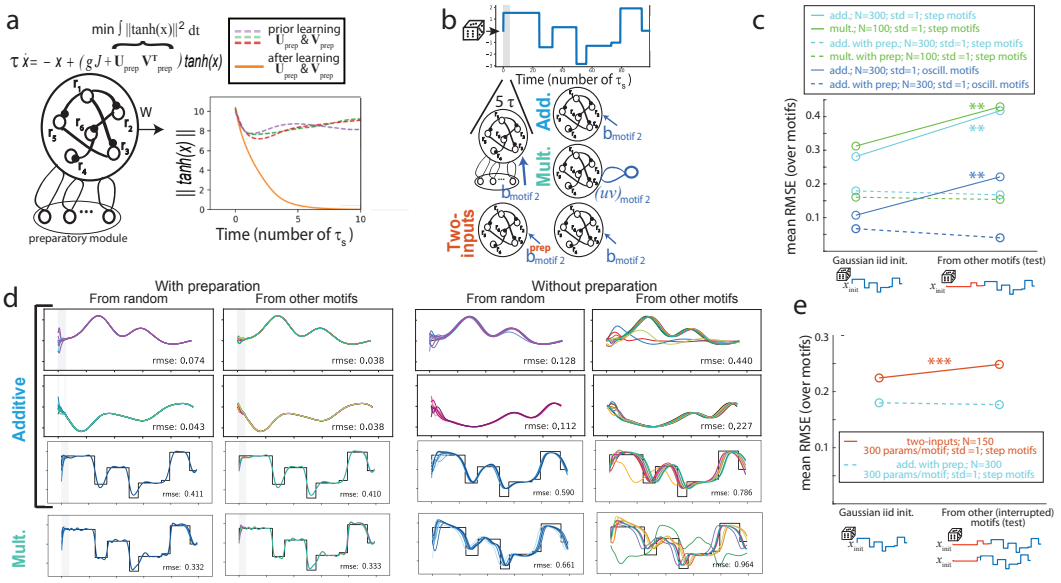

Figure 3: **Improvisation of motif sequences by RNNs with thalamocortical insights**.**a**) Training a preparatory module consisting of P loops weights forming a perturbation $\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal}$ of the connectivity, for a strongly nonlinear network. **b**) Training the motif-specific parameters in the additive, multiplicative and two-input architectures. **c**) Average root mean square error over motifs during training conditions vs. during transitioning, in the additive and multiplicative architectures; and with or without preparatory module. Ten step motifs are used. **d**) Example network performance when initializing the network state with the same random distribution as during training, as opposed to when transitioning from another motif. **e**) Same as e but comparing additive and two-inputs architectures using more motif transitions. Stars indicate a significant error increase (signed rank test).

For the multiplicative network, to make a more direct comparison with the thalamocortical model (Appendix A.4), we set the dynamics such that the input $\mathbf{b}_\mu$ and loop $\mathbf{u}_\mu\mathbf{v}_\mu^{\intercal}$ were only active during the transition and post-transition periods respectively:

$$\tau\dot{\mathbf{x}} = -\mathbf{x} + (g^{\text{mu}}\mathbf{J} + \mathbb{1}_{t\leq 5\tau}\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal} + \mathbb{1}_{t>5\tau}\mathbf{u}_\mu\mathbf{v}_\mu^{\intercal})\tanh(\mathbf{x}) + \mathbb{1}_{t\leq 5\tau}\mathbf{b}_\mu.$$

First, we observe that the RNNs trained with a preparatory module show a large improvement in accuracy even during the production of single motifs initialized with the same distribution used during training, especially for the more difficult step motifs ($\approx 50\%$ improvement, Fig. 3c and d, left). This is a strong indication that, during noise-robust training through random initialization, the imposition of fast network dynamics by the preparatory module allows the the tuned preparatory input to more efficiently steer the dynamics to explore and find a good network state that maximizes the accuracy of motif production. After training, the preparatory module also contributes to increasing the speed of convergence to this motif-specific network state. Conversely, without the preparatory module, the motif-specific parameters appear to struggle to fulfill all their functions to ensure accurate motif production (create fast dynamics at motif start, shape the pattern towards which activity must converge, and modulate the shared RNN dynamics during motif production).

Second, the RNNs trained with a preparatory module are now able to tackle zero-shot transfer to chains of motifs at no performance cost (Fig. 3c and d, right). This strongly argues that the preparatory module shapes a wide attractive landscape at motif start that includes the complex correlated network states reached at the end of motifs.

Third, through nonlinear interactions between the preparatory input and the rich dynamics of the RNNs, our networks can match the motifs' shapes during preparation (Fig. 3d, first two rows). This is another advantage of our RNNs over the above-mentioned dynamically limited models (which can only implement a motif-independent interpolation during preparation, besides having more limited expressivity (Kao et al. (2020); Logiaco et al. (2019) and Appendix A.4). This enables robust, accurate and plausible motif transitions without any tuning of the duration of the preparatory period.

Finally, we want to stress the advantages of the preparatory connectivity perturbation $\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\mathsf{T}}$ compared to possible alternative approaches that would only try to match the apparent timecourse of the neural activity during preparation in the brain (Sussillo et al., 2015; Zimnik & Churchland, 2021). First, the preparatory loops can modulate the input in a unit-specific way to create efficient convergence with a subtle modulation of the RNN connectivity. This is the reason why the preparatory input can still interact with the rich RNN's recurrent connectivity, which allows this input to shape the output into the desired motif even during preparation. Second, because these loops are not tuned for any particular motif, they can be optimized once and for all, and do not impose any additional cost when learning new motifs. Instead, when considering naive ways of setting specific dynamics at the beginnings of motifs such as using a second motif-specific input at the start of each motif ('two-inputs' approach, Fig. 3c), any parameter used for preparation scales with the number of motifs. We made this 'two-inputs' approach comparable to our preparatory module approach by applying the first input the network for the first 5τ of the motifs and then a second input during the remainder of the motif, and by equalizing the number of parameters involved (as we had done between the additive and multiplicative network) by setting $N = 150$. We found that this strategy led to both decreased accuracy and robustness compared to using our preparatory module, with both effects combining to a large ($\approx 40\%$) increase in RMSE during transitioning. We could precisely assess the transitions robustness in these networks this by using an extended motif transition data set (Fig. 3e). More precisely, we considered transitions between pairs of step motifs where the transition occurs at fractions of 0.5, 0.6, 0.7, 0.8, 0.9, 1.0 of the total duration of the first motif (e.g. Fig. 3e right, bottom). This is a more challenging task – that involves a larger variety of network states from which to transition than before (then, we only considered the 100% scenario where all motifs ended at the same value: zero). For these transitions, the 'two-inputs' network showed a significant increase in error – including occasional very large errors for some transitions, see Appendix A.5.2 – compared to initializing motifs with the randomness used during training. In contrast, the additive network had equally robust performance in both conditions (Fig. 3f). Thus, our results argue that using brain insights to constrain the architectures of networks has advantages, even beyond taking inspiration from the biological activity patterns (Li et al., 2019; Sinz et al., 2019; Zador, 2019).

5 Discussion

We found that when trained on many randomly initialized single motifs, gradient-trained nonlinear RNNs struggle during zero-shot transfer to new sequence orders. This is relatively surprising given that such RNNs are known to be able to learn attractive trajectories (Laje & Buonomano, 2013; Sussillo & Abbott, 2009; Pollock & Jazayeri, 2020). This suggests that training mostly constrains the readout dimension in our setting. This is both a blessing - as it is the ultimate reason why different motifs can efficiently share parameters and thus benefit from a common dynamical baseline that is only partially modulated, without being hurt by irrelevant aspects of the dynamics (Logiaco et al., 2019; Russo et al., 2018) - and a curse as it makes flexible motif chaining difficult. In this context, we show that using a preparatory module whose architecture is brain-inspired and who imposes a dynamics that mimics motor cortical activity during hierarchical motor sequencing (Zimnik & Churchland, 2021) can efficiently enable zero-shot transfer to new sequence orders. Even more surprisingly, introducing the preparatory module in the nonlinear RNNs actually sizably improves single motif training such that the motifs are also more accurate during training conditions. These results reveal a general function for motor preparation specifically for flexible chaining of motor motifs, and show that it is not a byproduct of arbitrary biological hardware constraints or of the restricting assumptions made by previous models to allow for analytical tractability (Logiaco et al., 2019; Kao et al., 2020). Therefore, ultimately, for the application to robotics control - at least in a context where intrinsic dynamics matter (Liu et al., 2019; Yeo et al., 2016) - this preparatory module could be very useful (i) to increase accuracy, (ii) to enable smooth transitions governed by dynamics that can be tuned to avoid passing through undesirable states, and (iii) to offer a mechanism for learning appropriate transition times by gradient descent - which are all current challenges for state-of-the-art networks (Merel et al., 2019a;b). In addition, our approach is extendable to combining recurrent dynamics - useful in contexts where sensory feedback is not fully reliable (Yeo et al., 2016; Liu et al., 2019) - with corrections driven by sensory inputs (Guo et al., 2019). Our work therefore joins several recent calls outlining the need to infuse ANNs with more expert domain knowledge in order to achieve good generalization in real-world scenarios involving out-of-distribution generalization and zero-shot transfer (Russin et al., 2019; D'Amour et al., 2020; Zador, 2019).

REFERENCES

Björn Brembs. The brain as a dynamically active organ. *Biochemical and Biophysical Research Communications*, 564:55–69, July 2021. doi: 10.1016/j.bbrc.2020.12.011. URL https://doi.org/10.1016/j.bbrc.2020.12.011.

R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, 2(1):14–23, 1986. doi: 10.1109/jra.1986.1087032. URL https://doi.org/10.1109/jra.1986.1087032.

Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. Capacity and trainability in recurrent neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=BydARw9ex.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification presents challenges for credibility in modern machine learning, 2020.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre M. Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/985e9a46e10005356bbaf194249f6856-Abstract.html.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. *CoRR*, abs/1910.07104, 2019. URL http://arxiv.org/abs/1910.07104.

Kevin Frans, Jonathan Ho, Xi Chen, Pieter Abbeel, and John Schulman. Meta learning shared hierarchies. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=SyX0IeWAW.

Claire E. Geddes, Hao Li, and Xin Jin. Optogenetic editing reveals the hierarchical organization of learned action sequences. *Cell*, 174(1):32–43.e15, June 2018. doi: 10.1016/j.cell.2018.06.012. URL https://doi.org/10.1016/j.cell.2018.06.012.

Jian-Zhong Guo, Britton Sauerbrei, Jeremy D. Cohen, Matteo Mischiati, Austin Graves, Ferruccio Pisanello, Kristin Branson, and Adam W. Hantman. Dynamics of the cortico-cerebellar loop fine-tune dexterous movement. *bioRXiv*, May 2019. doi: 10.1101/637447. URL https://doi.org/10.1101/637447.

Zengcai V. Guo, Hidehiko K. Inagaki, Kayvon Daie, Shaul Druckmann, Charles R. Gerfen, and Karel Svoboda. Maintenance of persistent activity in a frontal thalamocortical loop. *Nature*, 545(7653):181–186, May 2017. doi: 10.1038/nature22324. URL https://doi.org/10.1038/nature22324.

Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors. *Neural Computation*, 25 (2):328–373, 02 2013. ISSN 0899-7667. doi: 10.1162/NECO_a_00393. URL https://doi.org/10.1162/NECO_a_00393.

Herbert Jaeger. Echo state network. *Scholarpedia*, 2(9):2330, 2007. doi: 10.4249/scholarpedia.2330. URL https://doi.org/10.4249/scholarpedia.2330.

Ta-Chu Kao, Mahdieh S. Sadabadi, and Guillaume Hennequin. Optimal anticipatory control as a theory of motor preparation: a thalamo-cortical circuit model. *bioRxiv*, 2020. doi: 10.1101/2020.02.02.931246. URL https://www.biorxiv.org/content/early/2020/02/06/2020.02.02.931246.

Matthew T Kaufman, Mark M Churchland, Stephen I Ryu, and Krishna V Shenoy. Cortical activity in the null space: permitting preparation without movement. *Nature Neuroscience*, 17(3):440–448, February 2014. doi: 10.1038/nn.3643. URL https://doi.org/10.1038/nn.3643.

Felix P. Kemeth, Tom Bertalan, Nikolaos Evangelou, Tianqi Cui, Saurabh Malani, and Ioannis G. Kevrekidis. Initializing lstm internal states via manifold learning. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9):093111, Sep 2021. ISSN 1089-7682. doi: 10.1063/5.0055371. URL http://dx.doi.org/10.1063/5.0055371.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. ISSN 0027-8424. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/content/114/13/3521.

Tomas Kulvicius, KeJun Ning, Minija Tamosiunaite, and Florentin Worgtter. Joining movement sequences: Modified dynamic movement primitives for robotics applications exemplified on handwriting. *IEEE Transactions on Robotics*, 28(1):145–157, 2012. doi: 10.1109/TRO.2011.2163863.

Rodrigo Laje and Dean V Buonomano. Robust timing and motor patterns by taming chaos in recurrent neural networks. *Nature Neuroscience*, 16(7):925–933, May 2013. doi: 10.1038/nn.3405. URL https://doi.org/10.1038/nn.3405.

Brenden M. Lake and Marco Baroni. Still not systematic after all these years: On the compositional skills of sequence-to-sequence recurrent networks. *CoRR*, abs/1711.00350, 2017. URL http://arxiv.org/abs/1711.00350.

Itamar Daniel Landau and Haim Sompolinsky. Coherent chaos in a recurrent neural network with structured connectivity. *PLOS Computational Biology*, 14(12):e1006309, December 2018. doi: 10.1371/journal.pcbi.1006309. URL https://doi.org/10.1371/journal.pcbi.1006309.

Antonio H Lara, Gamaleldin F Elsayed, Andrew J Zimnik, John P Cunningham, and Mark M Churchland. Conservation of preparatory neural events in monkey motor cortex regardless of how movement is initiated. *eLife*, 7, August 2018. doi: 10.7554/elife.31826. URL https://doi.org/10.7554/elife.31826.

Zhe Li, Wieland Brendel, Edgar Y. Walker, Erick Cobos, Taliah Muhammad, Jacob Reimer, Matthias Bethge, Fabian H. Sinz, Xaq Pitkow, and Andreas S. Tolias. Learning from brains how to regularize machines. *CoRR*, abs/1911.05072, 2019. URL http://arxiv.org/abs/1911.05072.

Libin Liu and Jessica Hodgins. Learning to schedule control fragments for physics-based characters using deep q-learning. *ACM Transactions on Graphics*, 36(3):1–14, July 2017. doi: 10.1145/3083723. URL https://doi.org/10.1145/3083723.

Vincent Liu, Ademi Adeniji, Nathaniel Lee, Jason Zhao, and Mario Srouji. Recurrent control nets for deep reinforcement learning. *CoRR*, abs/1901.01994, 2019. URL http://arxiv.org/abs/1901.01994.

Laureline Logiaco, L.F. Abbott, and Sean Escola. A model of flexible motor sequencing through thalamic control of cortical dynamics. *bioRxiv*, 2019. doi: 10.1101/2019.12.17.880153. URL https://www.biorxiv.org/content/early/2019/12/18/2019.12.17.880153.

Niru Maheswaranathan, Alex H. Williams, Matthew D. Golub, Surya Ganguli, and David Sussillo. Universality and individuality in neural dynamics across large populations of recurrent networks, 2019.

Josh Merel, Arun Ahuja, Vu Pham, Saran Tunyasuvunakool, Siqi Liu, Dhruva Tirumala, Nicolas Heess, and Greg Wayne. Hierarchical visuomotor control of humanoids. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019a. URL https://openreview.net/forum?id=BJfYvo09Y7.

Josh Merel, Matthew Botvinick, and Greg Wayne. Hierarchical motor control in mammals and machines. *Nature Communications*, 10(1), December 2019b. doi: 10.1038/s41467-019-13239-6. URL https://doi.org/10.1038/s41467-019-13239-6.

Nima Mohajerin and Steven L. Waslander. Multistep prediction of dynamic systems with recurrent neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3370–3383, 2019. doi: 10.1109/TNNLS.2019.2891257.

Abdulraheem Nashef, Rea Mitelman, Ran Harel, Mati Joshua, and Yifat Prut. Area-specific thalamocortical synchronization underlies the transition from motor planning to execution. *Proceedings of the National Academy of Sciences*, 118(6):e2012658118, February 2021. doi: 10.1073/pnas.2012658118. URL https://doi.org/10.1073/pnas.2012658118.

Aran Nayebi, Javier Sagastuy-Brena, Daniel M. Bear, Kohitij Kar, Jonas Kubilius, Surya Ganguli, David Sussillo, James J. DiCarlo, and Daniel L. K. Yamins. Goal-driven recurrent neural network models of the ventral visual stream. *bioRXiv*, February 2021. doi: 10.1101/2021.02.17.431717. URL https://doi.org/10.1101/2021.02.17.431717.

OpenAI OpenAI, Matthias Plappert, Raul Sampedro, Tao Xu, Ilge Akkaya, Vineet Kosaraju, Peter Welinder, Ruben D'Sa, Arthur Petron, Henrique Ponde de Oliveira Pinto, Alex Paino, Hyeonwoo Noh, Lilian Weng, Qiming Yuan, Casey Chu, and Wojciech Zaremba. Asymmetric self-play for automatic goal discovery in robotic manipulation, 2021.

German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, May 2019. doi: 10.1016/j.neunet.2019.01.012. URL https://doi.org/10.1016/j.neunet.2019.01.012.

Eli Pollock and Mehrdad Jazayeri. Engineering recurrent neural networks from task-relevant manifolds and dynamics. *PLOS Computational Biology*, 16(8):e1008128, August 2020. doi: 10.1371/journal.pcbi.1008128. URL https://doi.org/10.1371/journal.pcbi.1008128.

Tony J. Prescott, Peter Redgrave, and Kevin Gurney. Layered control architectures in robots and vertebrates. *Adaptive Behavior*, 7(1):99–127, 1999. doi: 10.1177/105971239900700105. URL https://doi.org/10.1177/105971239900700105.

Alexander Rivkind and Omri Barak. Local dynamics in trained recurrent neural networks. *Physical Review Letters*, 118(25), June 2017. doi: 10.1103/physrevlett.118.258101. URL https://doi.org/10.1103/physrevlett.118.258101.

Jacob Russin, Randall C. OReilly, and Yoshua Bengio. Deep learning needs a prefrontal cortex. In Yoshua Bengio and Yann LeCun (eds.), *8th International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2020, Workshop on Bridging AI and Cognitive Science*, 2019.

Abigail A. Russo, Sean R. Bittner, Sean M. Perkins, Jeffrey S. Seely, Brian M. London, Antonio H. Lara, Andrew Miri, Najja J. Marshall, Adam Kohn, Thomas M. Jessell, Laurence F. Abbott, John P. Cunningham, and Mark M. Churchland. Motor cortex embeds muscle-like commands in an untangled population response. *Neuron*, 97(4):953–966.e8, February 2018. doi: 10.1016/j.neuron.2018.01.004. URL https://doi.org/10.1016/j.neuron.2018.01.004.

Friedrich Schuessler, Alexis Dubreuil, Francesca Mastrogiuseppe, Srdjan Ostojic, and Omri Barak. Dynamics of random recurrent networks with correlated low-rank structure. *Physical Review Research*, 2(1), February 2020a. doi: 10.1103/physrevresearch.2.013111. URL https://doi.org/10.1103/physrevresearch.2.013111.

Friedrich Schuessler, Francesca Mastrogiuseppe, Alexis Dubreuil, Srdjan Ostojic, and Omri Barak. The interplay between randomness and structure during learning in rnns. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 13352–13362. Curran Associates, Inc., 2020b. URL https://proceedings.neurips.cc/paper/2020/file/9ac1382fd8fc4b631594aa135d16ad75-Paper.pdf.

K. V. Shenoy, M. Sahani, and M. M. Churchland. Cortical control of arm movements: a dynamical systems perspective. *Annu. Rev. Neurosci.*, 36:337–359, Jul 2013.

Fabian H. Sinz, Xaq Pitkow, Jacob Reimer, Matthias Bethge, and Andreas S. Tolias. Engineering a less artificial intelligence. *Neuron*, 103(6):967–979, September 2019. doi: 10.1016/j.neuron.2019.08.034. URL https://doi.org/10.1016/j.neuron.2019.08.034.

H. Sompolinsky, A. Crisanti, and H. J. Sommers. Chaos in random neural networks. *Phys. Rev. Lett.*, 61:259–262, Jul 1988. doi: 10.1103/PhysRevLett.61.259. URL https://link.aps.org/doi/10.1103/PhysRevLett.61.259.

Lee Susman, Francesca Mastrogiuseppe, Naama Brenner, and Omri Barak. Quality of internal representation shapes learning performance in feedback neural networks. *Physical Review Research*, 3(1), February 2021. doi: 10.1103/physrevresearch.3.013176. URL https://doi.org/10.1103/physrevresearch.3.013176.

D. Sussillo, M. M. Churchland, M. T. Kaufman, and K. V. Shenoy. A neural network that finds a naturalistic solution for the production of muscle activity. *Nat. Neurosci.*, 18(7):1025–1033, Jul 2015.

David Sussillo and L.F. Abbott. Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63(4):544–557, August 2009. doi: 10.1016/j.neuron.2009.07.018. URL https://doi.org/10.1016/j.neuron.2009.07.018.

Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating text with recurrent neural networks. In Lise Getoor and Tobias Scheffer (eds.), *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pp. 1017–1024, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5.

Jun Tani. Learning to generate articulated behavior through the bottom-up and the top-down interaction processes. *Neural Networks*, 16(1):11–23, January 2003. doi: 10.1016/s0893-6080(02)00214-9. URL https://doi.org/10.1016/s0893-6080(02)00214-9.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3540–3549. PMLR, 2017. URL http://proceedings.mlr.press/v70/vezhnevets17a.html.

Francis Wyffels and Benjamin Schrauwen. Design of a central pattern generator using reservoir computing for learning human motion. In *2009 Advanced Technologies for Enhanced Quality of Life*, pp. 118–122, 2009. doi: 10.1109/AT-EQUAL.2009.32.

Kelvin Xu, Siddharth Verma, Chelsea Finn, and Sergey Levine. Continual learning of control primitives : Skill discovery via reset-games. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 4999–5010. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/3472ab80b6dff70c54758fd6dfc800c2-Paper.pdf.

Sang-Hoon Yeo, David W. Franklin, and Daniel M. Wolpert. When optimal feedback control is not enough: Feedforward strategies are required for optimal control with active sensing. *PLOS Computational Biology*, 12(12):e1005190, December 2016. doi: 10.1371/journal.pcbi.1005190. URL https://doi.org/10.1371/journal.pcbi.1005190.

Jaehong Yoon, Saehoon Kim, Eunho Yang, and Sung Ju Hwang. Scalable and order-robust continual learning with additive parameter decomposition. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1gdj2EKPB.

Anthony M. Zador. A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature Communications*, 10(1), August 2019. doi: 10.1038/s41467-019-11786-6. URL https://doi.org/10.1038/s41467-019-11786-6.

Andrew J. Zimnik and Mark M. Churchland. Independent generation of sequence elements by motor cortex. *Nature Neuroscience*, 24(3):412–424, February 2021. doi: 10.1038/s41593-021-00798-5. URL https://doi.org/10.1038/s41593-021-00798-5.

A Appendix

A.1 Formal definition of our problem setting

Here is how we formally define our task:

-Given a dynamical system $\mathcal{D}$ with (i) internal variables $\mathbf{h}$ (i.e., $\mathbf{h}(t)$ is defined through its derivative with respect to the continuous time variable t) and (ii) the property that $\mathcal{D}$ can be in one of K different states indexed by k (with $k \in 1 \dots K$, and K is large and a priori unknown), each lasting a fixed duration t_k, such that whenever $\mathcal{D}$ is in state k, a function of the internal states $F(\mathbf{h})$ exactly follows a fixed continuous function g_k (of length t_k) that we term a 'motif'. (Concretely, if a particular trajectory of states $[s_2, s_1, s_3]$ is visited, $F(\mathbf{h}(t))$ is formed of the concatenation of three continuous 'motifs' indexed by $[s_2, s_1, s_3]$.)

- Given a sequence of indices $I = [p_1, ...p_M]$ that label a sequence of states undergone by $\mathcal{D}$, where $\forall i, p_i \in \{1, \dots, K\}$

- Given labeled training examples that only contain a finite number T_k of *discrete* samples from each continuous function g_k, and given that the learning from different g_k must each occur during a single separate continuous bout of learning

- Find a function f parameterized by descriptor vectors V_k of size $N_V < T_k$ (where N_V is as small as possible) such that $f(I; , V_1, \dots, V_K, dt)$ produces a time series corresponding to $F(\mathbf{h})$ when $\mathcal{D}$ undergoes the sequence of states ordered by I, with an adjustable sampling interval given by dt (such that a smaller dt leads to a smoother time series).

Finally, we note that is also desirable that f be noise-robust – for instance, it is desirable that the output of f could be accurate even if f receives noisy version of the descriptor vectors.

A.2 Motif generation method and full list of motifs used

For this study, we use two different types of motifs that challenge the dynamics of continuous time RNNs: oscillatory motifs, that are easier to produce; and step motifs that are harder.

A.2.1 Oscillatory motifs

To generate oscillatory motifs that would be relatively easy to produce by RNNs, we sampled the output from a random Gaussian network with $\tanh$ nonlinearity set in the chaotic regime (Sompolinsky et al., 1988; Sussillo & Abbott, 2009; Sussillo et al., 2015). To make these motifs very easy and possible to connect in 'seamless' chains, we also low-pass filtered the resulting trajectories.

Fig. 4 lists the oscillation motifs used in this paper.

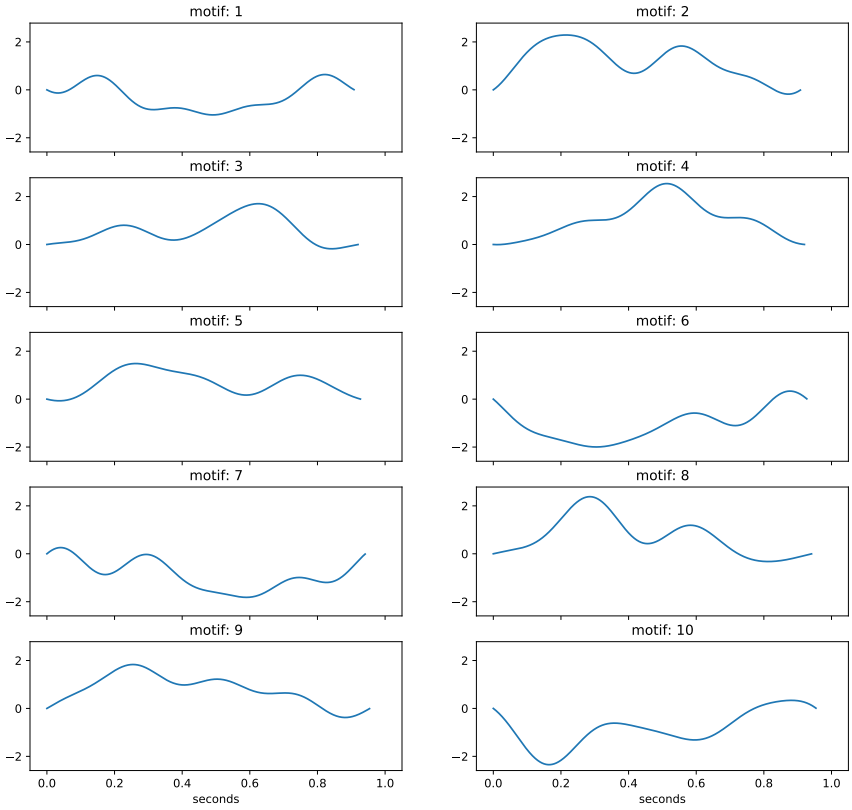

Figure 4: **Oscillatory motifs used in this paper**.

A.2.2 STEP MOTIFS

These motifs consist of a series of positive and negative discrete jumps with intervening constant periods. To generate a particular motif, we follow the following steps. First, we generate a centered Ornstein–Uhlenbeck process z by running the dynamics $dz/dt = -z + \left(\gamma/\sqrt{dt}\right)\chi(t)$, where $\chi(t)$ is taken from a standard Gaussian independently at each timestep, dt=0.1 and $\gamma = 3\sqrt{2}$ such that the steady-state standard deviation of this process is 3. After a 'warmup' period of $200\,dt$ that we discard, we produce an additional $1000\,dt$ of z-values that will be the basis for generating the motif. Second, we draw random time intervals T_k with a uniform probability between $50\,dt$ and $500\,dt$ and select the first few in a list $[T_1, T_2, ..., T_j]$ such that $\sum_{k=1}^{j} T_k < 1000\,dt$. Third, we set the value of the motif between the start and time T_1 to the average of the z-values over the same time interval; and similarly for the subsequent intervals, the value of the motif between the times $[T_{i-1} + dt, T_i]_{i\in[2,...,k]}$ is set to the average of the z-values over the corresponding interval. Finally, we pad the end of the motif with zeros for $50dt$ and also reset the first value of the motif to 0.

Fig. 5 lists the step motifs used in this paper.

A.3 PERFORMANCE OF 'VANILLA' RNNS

Here we show examples for the motif-by-motif performance of different 'vanilla' RNNs (i.e., without a preparatory module) that relate to section 3 of the main text.

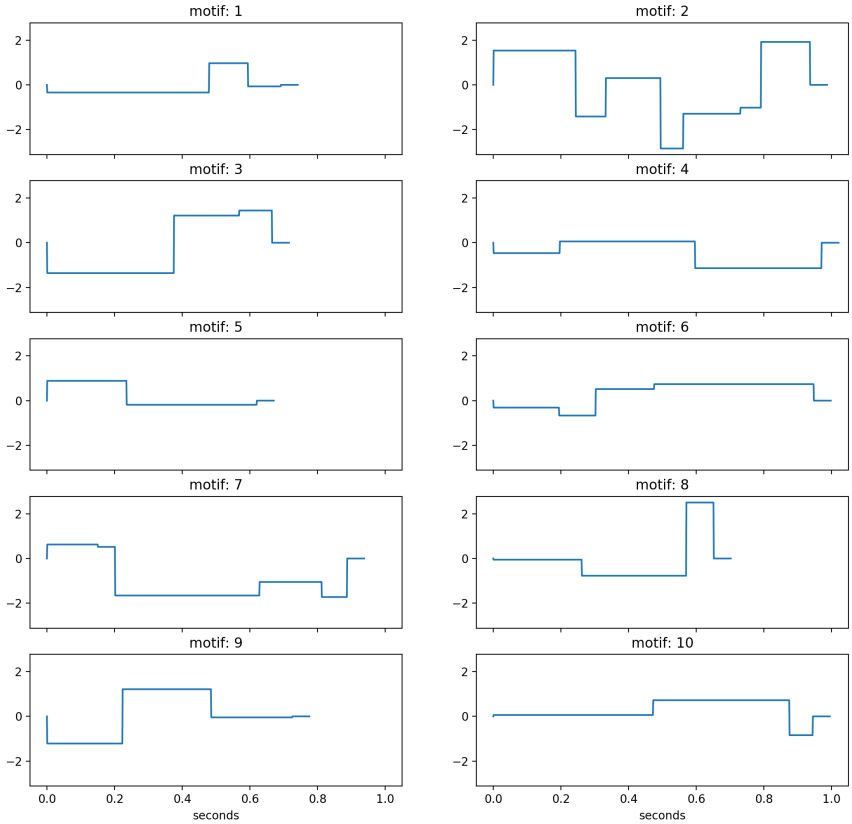

Figure 5: **Step motifs used in this paper**.

A.3.1 ADDITIVE NETWORK, N=300, STEP MOTIFS

In Fig. 6, we show an additive network of size 300 trained on the step motifs. This is a different network than the one we show in the main text Fig. 2a (i)-(iii) but the conclusions are identical – despite the presence of a small variability between different networks trained from different initial seeds (see e.g. main text Fig. 1c).

A.3.2 MULTIPLICATIVE NETWORK, N=100, STEP MOTIFS

In Fig. 7, we show a multiplicative network of size 100 trained on the step motifs. This is a different network than the one we show in the main text Fig. 2b but the conclusions are identical – despite the presence of a small variability between different networks trained from different initial seeds (see e.g. main text Fig. 1c). Note that the number of parameters tuned for motifs is adjusted relative to the additive network of size 300 shown previously (main text table 1).

A.3.3 ADDITIVE NETWORK, N=1000, STEP MOTIFS

In Fig. 8, we show an additive network of size 1000 trained on the step motifs. Notice that while increasing the network size improved the performance when starting from the random initialization used during training, there are still large errors when transitioning from other motifs.

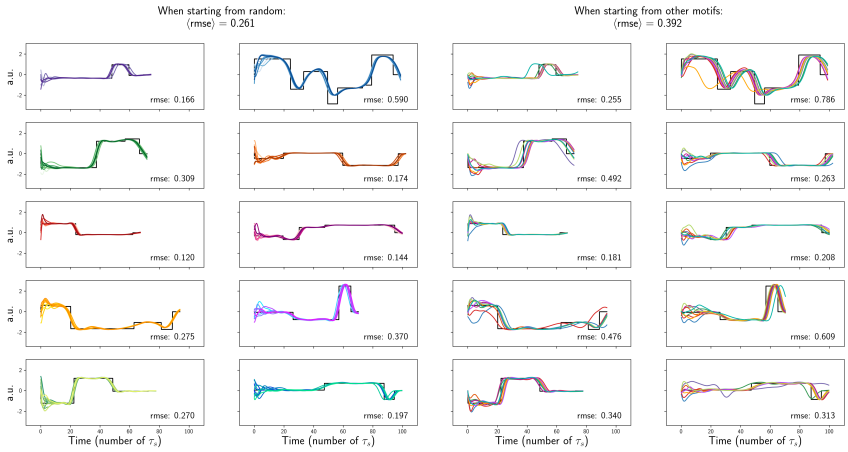

Figure 6: **Performance of an additive network, N=300, step motifs**. Left: the network outputs for each motif when starting from 9 different random $\mathbf{x}$ values. The saturations (light to dark) indicate different trials. Right: network outputs for each motif when starting from the $\mathbf{x}$ values taken at the end of the other 9 motifs. Colors indicate the identity of the prior motif (as labeled in the left panel). Numbers indicate the root mean square errors (rmse) for each motif (and their average over motifs).

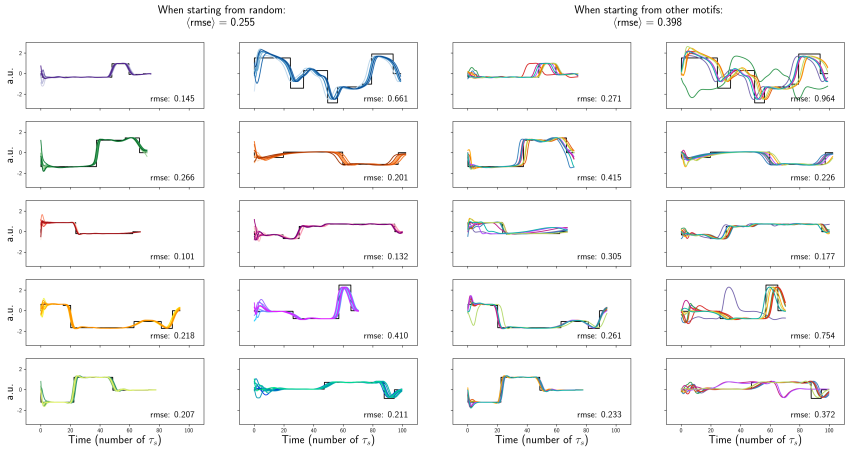

Figure 7: **Performance of a multiplicative network, N=100, step motifs**. Conventions as in Figure 6.

A.3.4 ADDITIVE NETWORK, N=300, OSCILLATORY MOTIFS

In Fig. 9, we show an additive network of size 300 trained on the oscillatory motifs. Notice that while, with these easier motifs, the performance is really good when starting from the random initialization used during training (as compared to the performance for the step motifs), there are still large errors when transitioning from other motifs. Also, we want to stress that, during sequencing, even when focusing on a particular first motif, the network performance can be very different depending on the identity of the second motif. For instance, let's focus on the light green motif in the panel 'When starting from random', bottom row on the left - the sequences where this motif is the first motif are shown as green traces in the panel 'When starting from other motifs'. When this green light motif is the first motif in a sequence, the performance is almost flawless for following the motif in the panel 'When starting from other motifs', second line on the left. In contrast, the performance

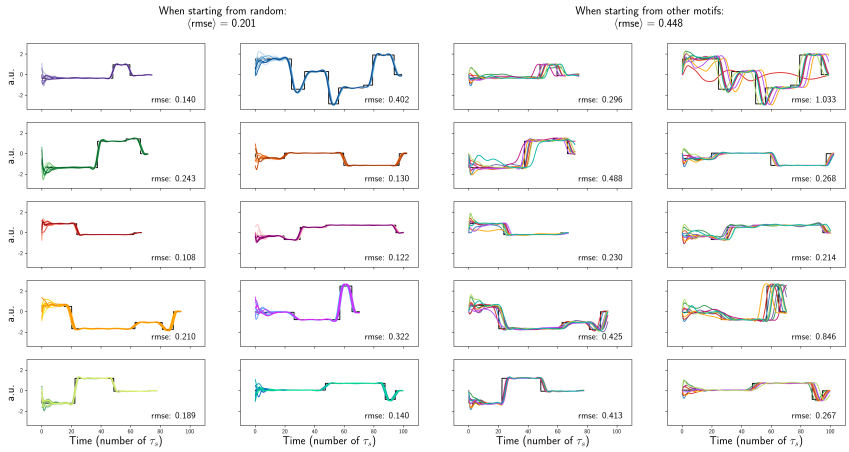

Figure 8: **Performance of an additive network, N=1000, step motifs**. Conventions as in Figure 6.

is very poor for another following motif situated in the panel 'When starting from other motifs', fourth line on the right.

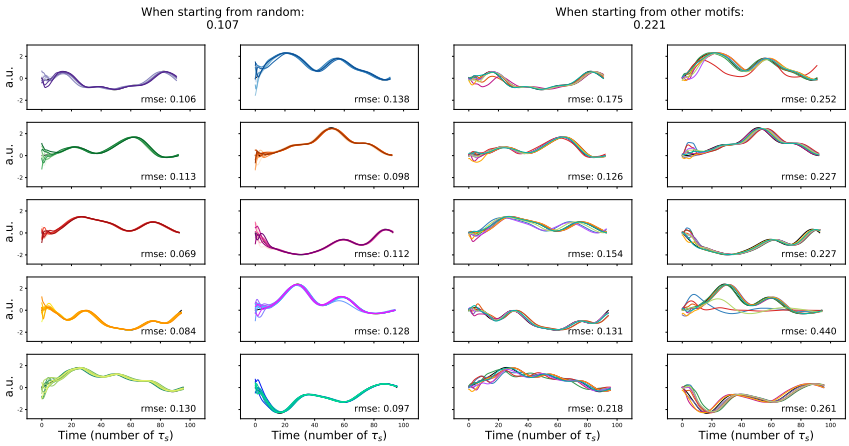

Figure 9: **Performance of an additive network, N=300, oscillatory motifs**. Conventions as in Figure 6.

A.3.5 Control network, N=50, step motifs

In Fig. 10, we show a control network of size 50 trained on the step motifs. Like the additive and multiplicative networks, this control network struggles when generalizing from single motif training to motif sequencing.

A.4 Analytically tractable thalamocortical model

Here, we summarize the insights that can be gained from an analytically tractable thalamocortical model whose switching linear dynamics are constrained to stay in the linear regime during individual motif preparation and execution ((Logiaco et al., 2019), see also (Kao et al., 2020)). Importantly, the constraints on the dynamics of this model largely limit its computational capacity, and no online learning method has been suggested for this model so that the approach is strongly limited to

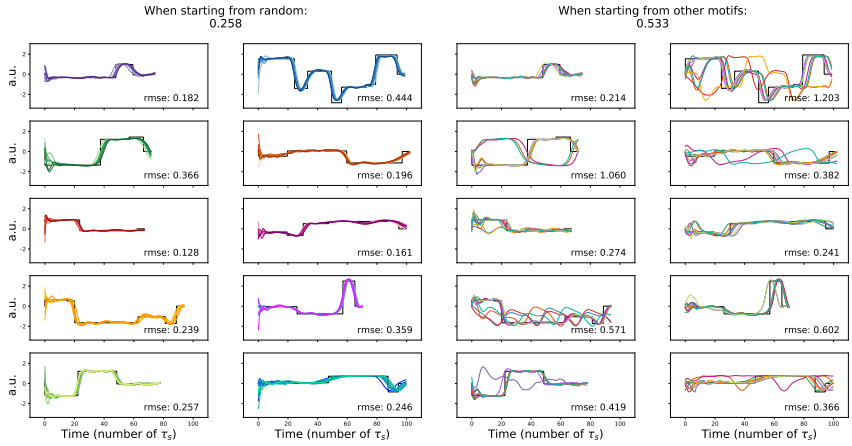

Figure 10: **Performance of a control network, N=50, step motifs**. Conventions as in Figure 6.

simple scenarios where analytical tractability is possible. The work set forth in the main text is therefore critical to test and expand the ideas of this model in a setting that is amenable to solving useful tasks in an engineering sense, and for determining whether the biological features included in this model might have been selected as generally useful rather than reflecting random evolutionary idiosyncracies.

The model consists of a recurrent cortical module with connectivity $\mathbf{J}^{\text{cc}}$ and activities $\mathbf{x}$ whose projection through the readout weights $\mathbf{w}$ constitutes the output. This cortical module interacts with a non-recurrent thalamic module through instantaneous loops consisting of corticothalamic and thalamocortical weights. The basal ganglia, which provides inhibitory input into thalamus, are modeled as selectively disinhibiting specific thalamic loops in order to cause execution of the associated motif.

Motif execution: During motif μ, a single thalamic loop is disinhibited leading to the dynamics:

$$\tau\dot{\mathbf{x}} = \tilde{\mathbf{J}}_\mu \mathbf{x}, \quad \text{where} \quad \tilde{\mathbf{J}}_\mu \equiv g\left(\mathbf{J}^{\text{cc}} - \mathbf{I}\right) + \mathbf{u}_\mu \mathbf{v}_\mu^\intercal, \tag{1}$$

with motif-specific loop vectors $\mathbf{u}_\mu$ and $\mathbf{v}_\mu^\intercal$. We now consider how these dynamics can approximate a desired output y_μ, knowing that in general a good approximation for y_μ can be reached through a (preliminarily numerically identified) linear combination of a small number K of complex exponentials: $y_\mu(t) \approx \hat{y}_\mu(t) = \sum_{k=1}^{K} [\hat{\boldsymbol{\alpha}}_\mu]_k e^{[\hat{\boldsymbol{\lambda}}_\mu]_k t}$ (see main text Fig. 1d and main text Fig. 3a, right, where the parameters of the basis functions A_k and ϕ_k relate to the real and imaginary parts of $[\hat{\boldsymbol{\alpha}}_\mu]_k$, and their timescales T_k and τ_k relate to the real and imaginary parts of $[\hat{\boldsymbol{\lambda}}_\mu]_k$, (Logiaco et al., 2019)). The cortical readout can exactly match $\hat{y}_\mu$ if the eigenvalues of $\tilde{\mathbf{J}}_\mu$ contain the entries of the vector $\hat{\boldsymbol{\lambda}}_\mu$ and if the initial network activities $\mathbf{x}_\mu^{\text{init}}$ are set correctly. We accomplish the former (Fig. 11a) by setting $\mathbf{v}_\mu = \mathbf{L}^\intercal \operatorname{diag}(\mathbf{L}\mathbf{u}_\mu)^{-1}\mathbf{Q}^+\mathbf{1}$, where $\mathbf{L}$ is the left eigenvector matrix of $\mathbf{J}^{\text{cc}}$, and $Q_{kj} = 1/([\hat{\boldsymbol{\lambda}}_\mu]_k - \lambda_j)$ where λ_j is an eigenvalue of $g\left(\mathbf{J}^{\text{cc}} - \mathbf{I}\right)$. Next, we set the initial activities at the beginning of motif μ to $\mathbf{x}_\mu^{\text{init}} = \tilde{\mathbf{R}}_\mu \operatorname{diag}(\tilde{\mathbf{R}}_\mu^\intercal \mathbf{w})^{-1}\boldsymbol{\alpha}_\mu$ where $\tilde{\mathbf{R}}_\mu$ contains right eigenvectors of $\tilde{\mathbf{J}}_\mu$ (with the first K columns corresponding to the eigenvalues in $\hat{\boldsymbol{\lambda}}_\mu$), and $[\boldsymbol{\alpha}_\mu]_{k\leq K} = [\hat{\boldsymbol{\alpha}}_\mu]_k$ and $[\boldsymbol{\alpha}_\mu]_{k>K} = 0$. This setting of the initial activities is equivalent to setting the initial amplitudes A and phases ϕ of the basis functions (main text Fig. 3a, right)

The preceding two steps do not specify $\mathbf{u}_\mu$, and with random $\mathbf{u}_\mu$ the readout will be highly sensitive to noise in $\mathbf{x}_\mu^{\text{init}}$ (pink trace in Fig. 11a right; see (Logiaco et al., 2019)). However, if $\mathbf{u}_\mu$ is set to minimize the analytically-computed expected readout deviation due to noise in $\mathbf{x}_\mu^{\text{init}}$ by modifying the eigenvectors of the dynamics, then robust readout is possible (cyan trace in Fig. 11a right; minimization of the cost $C(\mathbf{u})$ in (Logiaco et al., 2019)).

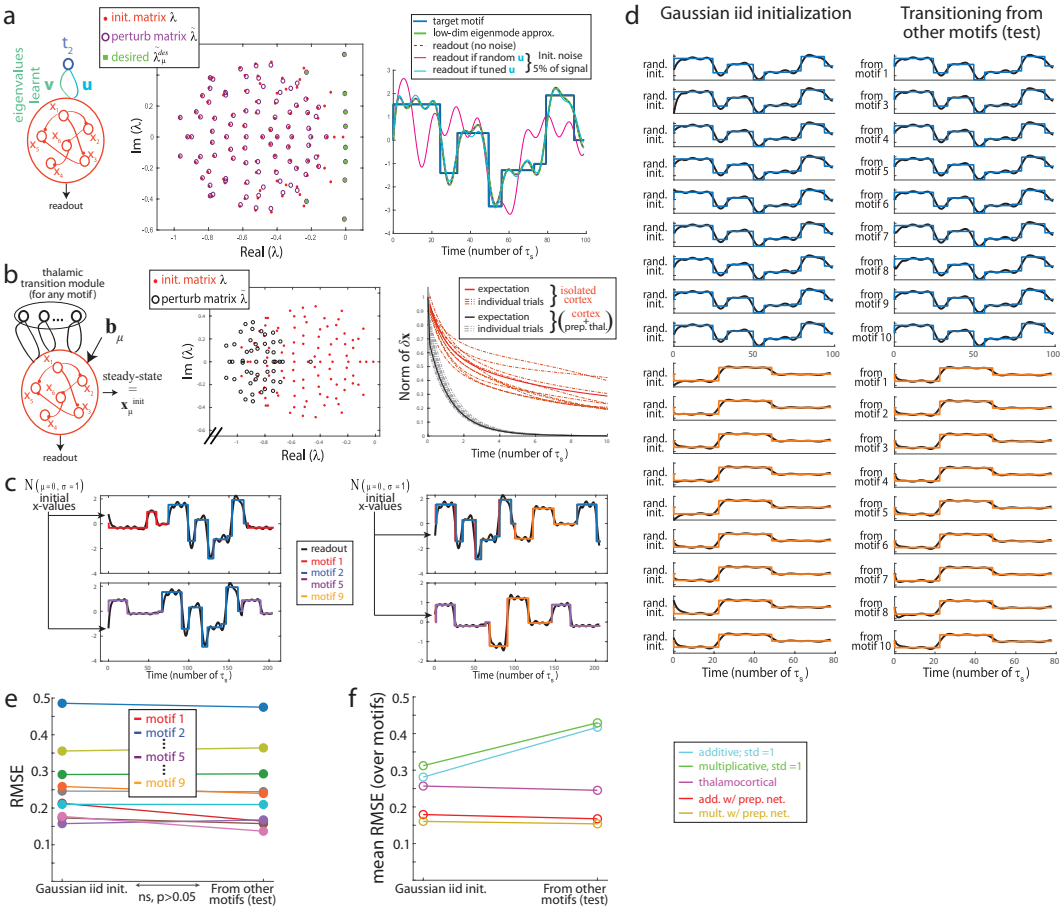

Figure 11: **Robust transitions in thalamocortical model**. **a**) Adjusting a motif-specific loop through the thalamic unit $\mathbf{t}_2$ (i.e. motor preparation, left), leading to the control of both eigenvalues (middle) and eigenvectors of the dynamics $\mathbf{x}(t)$ such that the readout robustly follows motif 2 (right). **b**) Thalamic module used for all motif transitions (which involves preparation of the cortical state to execute the next motif, left). After optimization, the thalamic module sets the eigenvalues of the dynamics to be more negative (middle) which results in a fast decrease of the distance to steady-state $|\delta\mathbf{x}|$ (right). **c**) Example sequences. **d**) Example trials starting from standard Gaussian random initial conditions vs. when transitioning from other motifs. **e**) Change of root mean square error for each motif between random initialization and zero-shot transfer to motifs transitions (not significant as per signed rank test). **f**) Comparing the performance between the thalamocortical model and the models presented in the main text, for both random initialization and zero-shot transfer to motifs transitions.

Motif transitions: To successfully transition to motif μ, it is sufficient to implement a mechanism by which $\mathbf{x}$ approaches $\mathbf{x}_\mu^{\text{init}}$, which will be the case if the dynamics during a so-called "preparatory period" has $\mathbf{x}_\mu^{\text{init}}$ as its steady-state. Additionally, it is desirable that the transition dynamics are fast and that they do not cause large transients values on the readout while relaxing to steady-state (Kaufman et al., 2014). To achieve this, it is possible (Logiaco et al., 2019) to employ a specific thalamic subpopulation of size P which is disinhibited during all motif transitions, as well as a constant input $\mathbf{c}_\mu$ specific to the upcoming motif μ, leading to the dynamics:

$$\tau\dot{\mathbf{x}} = \mathbf{J}_{\text{prep}}\mathbf{x} + \mathbf{c}_\mu, \quad \text{where} \quad \mathbf{J}_{\text{prep}} \equiv g\left(\mathbf{J}^{\text{cc}} - \mathbf{I}\right) + \mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal}, \tag{2}$$

with $N \times P$ loop weights $\mathbf{U}_{\text{prep}}$ and $\mathbf{V}_{\text{prep}}$. With these dynamics, the activity at steady-state will match $\mathbf{x}_\mu^{\text{init}}$ if $\mathbf{c}_\mu = -\mathbf{J}_{\text{prep}}\mathbf{x}_\mu^{\text{init}}$ (Fig. 11b). Note that the difference $\delta\mathbf{x} = \mathbf{x} - \mathbf{x}_\mu^{\text{init}}$ between the cortical activities and their steady state decays at a rate that is independent of $\mathbf{c}_\mu$ and therefore of the upcoming motif: $\tau\dot{\delta\mathbf{x}} = \mathbf{J}_{\text{prep}}\delta\mathbf{x}$ for all μ. This allows us to optimize the same weights $\mathbf{U}_{\text{prep}}$

and $\mathbf{V}_{\text{prep}}$ to favor rapid and smooth transitions between all pairs of motifs. Following ref. (Logiaco et al., 2019), we achieve fast transitions by minimizing the time-integral of the expected square norm of $\delta\mathbf{x}$, with rates $\delta\mathbf{x}_0$ at the beginning of the transition period sampled iid. Here, we also augment our cost function with the time-integral of the expected squared derivative of the readout to ensure smooth transitions. Our total cost function is therefore:

$$C(\mathbf{U}_{\text{prep}}, \mathbf{V}_{\text{prep}}) = E_{\delta\mathbf{x}_0}\left[\int_0^\infty dt\, \|\delta\mathbf{x}\|^2\right] + \beta\, N\, E_{\delta\mathbf{x}_0}\left[\int_0^\infty dt \left(\frac{d}{dt}\mathbf{w}^\intercal \delta\mathbf{x}\right)^2\right] \tag{3}$$
$$\propto \operatorname{Tr}\left(\mathbf{R}_{\text{prep}}\left(\left(\mathbf{L}_{\text{prep}}\,\mathbf{L}_{\text{prep}}^\intercal\right)\odot\mathbf{\Lambda}\right)\mathbf{R}_{\text{prep}}^\intercal\right) + \beta\, N\, \mathbf{w}^\intercal\mathbf{R}_{\text{prep}}\left(\left(\mathbf{L}_{\text{prep}}\,\mathbf{L}_{\text{prep}}^\intercal\right)\odot\mathbf{\Gamma}\right)\mathbf{R}_{\text{prep}}^\intercal\mathbf{w}$$

where $\mathbf{R}_{\text{prep}}$ and $\mathbf{L}_{\text{prep}}$ are the right and left eigenvectors of $\mathbf{J}_{\text{prep}}$, and its eigenvalues $\boldsymbol{\lambda}^{\text{prep}}$ are used to compute $\Lambda_{ij} = -1/(\lambda_i^{\text{prep}} + \lambda_j^{\text{prep}})$ and $\Gamma_{ij} = \lambda_i^{\text{prep}}\lambda_j^{\text{prep}}\Lambda_{ij}$. Finally, N is the number of cortical units and β is a hyperparameter which trades off the relative importance of transition speed and readout smoothness.

Note that the model parameters can be adjusted through analytical and semi-analytical techniques which do not require stochastic gradient descent on the simulated dynamics. This is an advantage when strictly imposing linear dynamical regimes during each motor preparation and motor execution for the fixed type of autonomous computation that it was designed for, but these restrictions can be limiting to optimize the network for more complex objectives (for instance, including correcting responses to sensory feedback).

We simulated the model on the same task as the gradient-trained RNNs, using our step motifs. The motif-specific parameters scale as in the multiplicative architecture (main text table 1), but the thalamocortical model does have a few more hyperparameters. To make sure that these did not induce an inability to compare between approaches, we reduced the cortical size to $N = 99$. Further, after exploring a few values, we set $g/\tau = 0.5$, $K = 10$, $P = 50$, $\beta = 1/20$, and the motif transition duration to 5τ. The readout weights $\mathbf{w}$ and recurrent weights $\mathbf{J}^{\text{cc}}$ were sampled from a centered Gaussian distribution with std $1/\sqrt{N}$. The approximations $\hat{y}_\mu$ were fit to the target motifs under the constraints that $\hat{y}_\mu(0) = 0$, that the elements of $\hat{\boldsymbol{\lambda}}_\mu$ had negative real part and were at least $\epsilon = 0.05$ apart from each other, and that the magnitudes of the elements of $\hat{\boldsymbol{\alpha}}_\mu$ were not exceedingly large (no larger than 3). The resulting $\hat{\boldsymbol{\lambda}}_\mu$ and $\hat{\boldsymbol{\alpha}}_\mu$ were then used to optimize $\mathbf{u}_\mu$, $\mathbf{v}_\mu$, and $\mathbf{c}_\mu$ as described above.

We generated sequences from the thalamocortical model after initializing with iid standard Gaussian samples for the elements of $\mathbf{x}$ (this choice of a unit standard deviation indeed leads to readout values within the same range as the the target motifs, Fig. 11c). Interestingly, the motifs were produced with the same reliability when preceded by another motif in a sequence or when starting from random initial conditions (Fig. 11d,e). Also, choosing $K = 10$ pushed the network close to its limit in terms of noise robustness, without allowing the constrained thalamocortical network to reach the performance of nonlinear RNNs with the preparatory module (Fig. 11g).

In conclusion, the constrained thalamocortical model is therefore limited in its expressivity, in its capacity to shape the readout during motif transitions, as well as by the rigid and unpractical procedure by which its weights are adjusted.

We will now present additional results demonstrating that insights from this constrained thalamocortical model can be weaved into RNNs to combine zero-shot transfer ability with large expressivity and the flexibility afforded by gradient-based training.

A.5 Weaving in insights from the thalamocortical model in gradient-trained ANNs

In Section 4 of the main text, we presented the results of augmenting each of the additive and multiplicative networks with a 50-unit thalamic transition module that is active for the first 5τ of every motif. Here we present more detail about these results.

A.5.1 DYNAMICS OF THE RNNS INTERACTING WITH A THALAMIC TRANSITION MODULE

The dynamics of our nonlinear recurrent networks with the transition module but without any input are:

$$\tau\dot{\mathbf{x}} = -\mathbf{x} + (g^x\mathbf{J} + \mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal})\tanh(\mathbf{x}),$$

where g^x is g^{ad} or g^{mu} for the additive and multiplicative networks respectively. The weights in $\mathbf{U}_{\text{prep}}$ and $\mathbf{V}_{\text{prep}}$ were initialized with centered Gaussian with std $\sqrt{0.05/\sqrt{P*N}}$. We trained the weights of the transition modules with ADAM under the cost function $\sum_t |\mathbf{r}(t)|^2$ where $\mathbf{r}(t) = \tanh(\mathbf{x}(t))$. Minibatches consisted of 64 trials of length 20τ, each starting with random $\mathbf{x}$ values sampled iid from the standard normal distribution. After 1,000 minibatches, both $N = 300$ and $N = 100$ networks were seen to have converged.

Figs. 12a,d show the eigenvalue distributions of $g^x\mathbf{J}+\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal}$ after training. Importantly, though the thalamic module is low rank ($P < N$), all eigenvalues have real part significantly less than 1 which causes decay of the network rates towards a $\mathbf{0}$ fixed-point in the vicinity of this fixed-point. For comparison, we show the eigenspectrum of $g^x\mathbf{J}_{\text{rnd}} + \mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal}$ for random matrix $\mathbf{J}_{\text{rnd}}$ which has the same statistics as $\mathbf{J}$. In this case, there continues to be large eigenvalues that will prevent fast rate decay. These results demonstrate that the solutions $\mathbf{U}_{\text{prep}}$ and $\mathbf{V}_{\text{prep}}$ needed to negate the amplifying dynamics of $\mathbf{J}$ are specific to that particular $\mathbf{J}$.

In Figs. 12b,e, we show the time evolution of the norm of the rate vector $\mathbf{r}$ during three sample trajectories prior to learning and one post-learning (all post-learning samples are nearly identical). On the scale plotted, the norms after learning are indistinguishable from zero after approximately 7τ.

With the thalamic transition modules in hand, we retrained the additive and multiplicative networks as described in the main text Sections 2 and 3. For the additive network, our only modification from before is that we included the thalamic transition module in the network dynamics for the first 5τ:

$$\tau\dot{\mathbf{x}} = -\mathbf{x} + (g^{\text{ad}}\mathbf{J} + \mathbb{1}_{t\leq 5\tau}\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal})\tanh(\mathbf{x}) + \mathbf{b}_\mu.$$

For the multiplicative network, to make a more direct comparison with the thalamocortical model presented in the previous section of the Appendix, we set the dynamics such that the input $\mathbf{b}_\mu$ and loop $\mathbf{u}_\mu\mathbf{v}_\mu^{\intercal}$ were only active during the transition and post-transition periods respectively:

$$\tau\dot{\mathbf{x}} = -\mathbf{x} + (g^{\text{mu}}\mathbf{J} + \mathbb{1}_{t\leq 5\tau}\mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal} + \mathbb{1}_{t>5\tau}\mathbf{u}_\mu\mathbf{v}_\mu^{\intercal})\tanh(\mathbf{x}) + \mathbb{1}_{t\leq 5\tau}\mathbf{b}_\mu.$$

Figs. 12c,f show that both networks learn to perform the task and show no degradation in their performance when tested on sequence generation (i.e., having initial $\mathbf{x}$ values given by the ends of other motifs) rather than when starting them with standard Gaussian $\mathbf{x}$ values (which was their training regime).

Similarly, Fig. 13 shows that training on smooth motifs using a preparatory module leads to robust performance during zero-shot transfer to motif chaining in an example additive network (compare to Fig. 9). Notice how the shape of the motif is also matched during the preparatory period, which is enabled by the interaction between the motif-specific input $\mathbf{b}_\mu$ and the nonlinear $\tanh$ dynamics.

A.5.2 COMPARING THE THALAMIC TRANSITION NETWORK TO OTHER WAYS OF IMPLEMENTING A DIFFERENT DYNAMICAL REGIME AT TRANSITIONS

In the main text, at the end of section 4, we gave statistics about a comparison between using transition module described above and an alternative approach for setting the RNN in a different dynamical state at the beginning of a motif. More specifically, we considered a 'two-inputs' approach where we use a first motif-specific input during the first 5τ of each motif to try to facilitate transitions, and a second motif-specific input during the rest of the motif to facilitate the production of the motif by the network.

Following the general approach in our article which adjusts the number of parameters tuned per motif (main text table 1), we set the size of the recurrent network to $N = 150$ so that the two inputs together lead to 300 parameters tuned per motif (and therefore 3000 parameters tuned in total when learning the ten motifs).

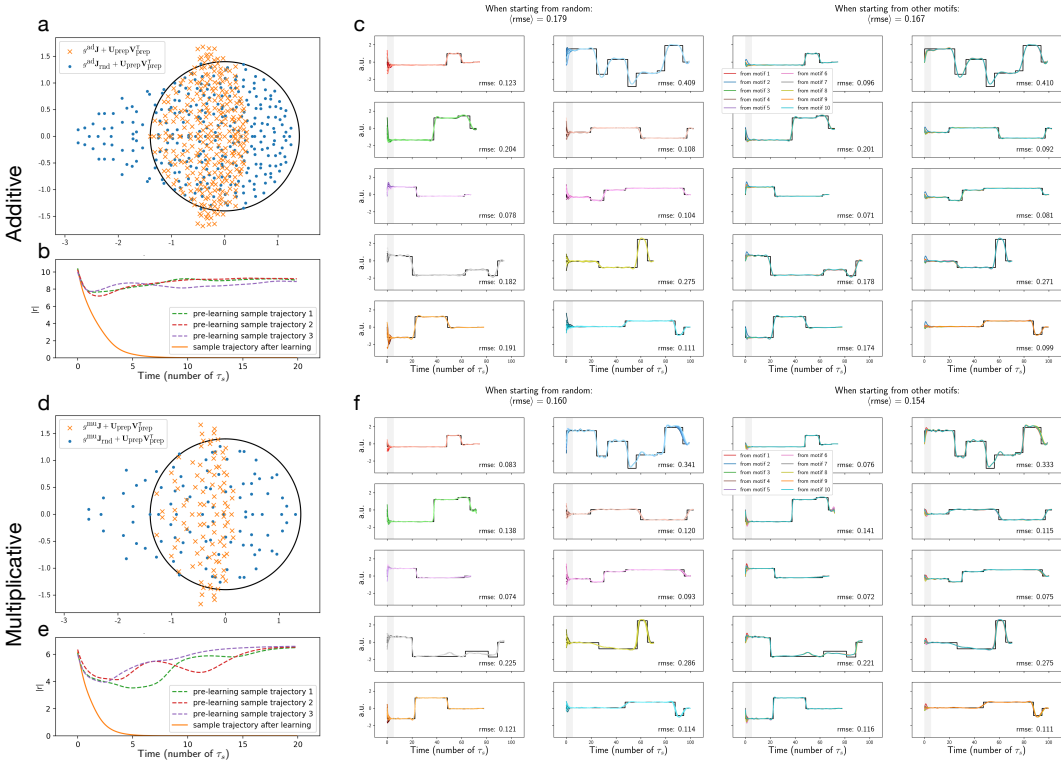

Figure 12: **Using a thalamic transition module rescues transitioning for nonlinear RNNs trained with SGD**. **a,b,c** Additive model. **a**. Eigenspectrum of $g^{\text{ad}}\mathbf{J} + \mathbf{U}_{\text{prep}}\mathbf{V}_{\text{prep}}^{\intercal}$ after training of $\mathbf{U}_{\text{prep}}$ and $\mathbf{V}_{\text{prep}}$ (orange crosses) and when replacing $\mathbf{J}$ with a random matrix not used during training (blue dots). Black circle has radius g^{ad}. **b**. $|\mathbf{r}|$ versus time before (dotted lines) and after (solid line) training of $\mathbf{U}_{\text{prep}}$ and $\mathbf{V}_{\text{prep}}$. **c**. The grey bars indicate the time during which the transition module was active. Other conventions as in Figure 6. **d,e,f**. As in **a,b,c** for the multiplicative model.

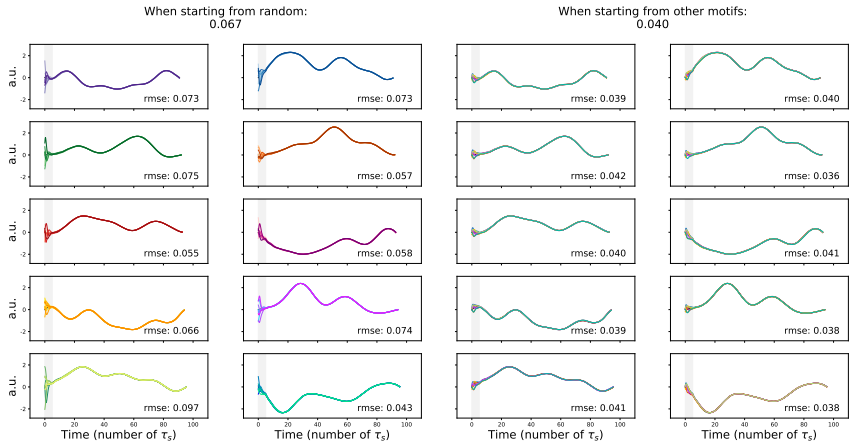

Figure 13: **Using a thalamic transition module rescues transitioning between oscillatory motifs in an additive ANN**. Conventions as in Figure 6.

We used an extended data set consisting of transitions between pairs of step motifs where the transition occurs at 50%, 60%, 70%, 80%, 90%, or 100% of the total duration of the first motif. This is a more challenging task than before when we only considered the 100% scenario because all motifs

end at the same value (zero). As stated in the main text, we found that the 'two-input' network did not perform as well as the matched additive network with a transition module (as described in the previous subsection of the Appendix, this module is not tuned to particular motifs, so the size of the recurrent network is here simply $N = 300$ to match the per-motif parameter count with the 'two-input' network).

In Fig. 14, we show examples for the single-motif performance of the 'two-input' network and the matched additive network, when transitioning from 70% vs. 100% of the duration of the previous motif (the latter one being the same situation as considered elsewhere in this work).

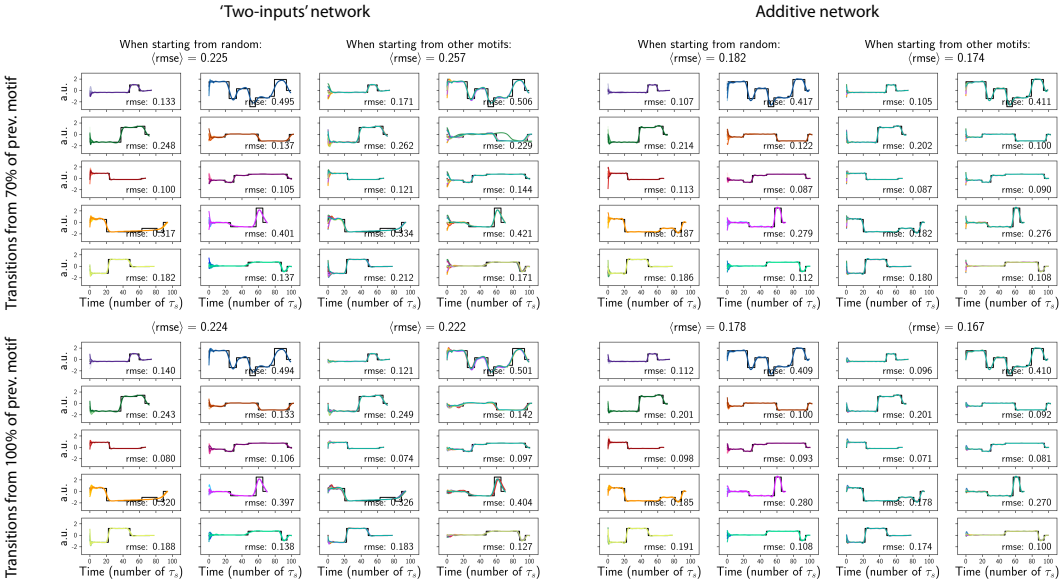

Figure 14: **Comparing a 'two-inputs' vs. additive network**. Left half: 'two-inputs' network; right half: additive network, adjusted for the number of parameters tuned per motif. Top half: comparing random initialization as experienced during training, and transitioning from 70% of the duration of other motifs. Bottom half: comparing random initialization as experienced during training, and transitioning from 100% of the duration of other motifs. For each quadrant, the two left vs. two right columns differ in the initial state of the motif with conventions as in Figure 6.

Examining Fig. 14 reveals that the 'two-inputs' network behaves differently from the additive network in two major ways. First, its overall motif accuracy is lower; and second, when transitioning from 70% of the duration of the previous motif, their accuracy consistently decreased compared to when initializing the motif at random as during training. This decrease in accuracy was seen consistently when transitioning in all scenarios when the previous motifs interrupted before their ending (which is zero-padded). This is probably because the larger variability of network outputs when interrupting motifs is reflected in a larger variability of network states. Importantly, the additive network with a transition module could handle this variability to allow the network to produce a more accurate output when chaining an interrupted motif with a subsequent one.

These results strongly suggest that 'engineering' a fixed point in the dynamics at the start of motifs is a particularly potent way of ensuring accurate and robust motif performance. In addition, close to the fixed point at the start of motifs, the dynamics is quasi-linear, in which case classical results from control theory indicate that feedback loops such as those of the transition module are the optimal strategy to steer the network towards its target. Therefore, these results further strengthen our conclusion that the transition module we propose is a particularly potent strategy for zero-shot transfer during motif chaining.

A.6 A NOTE ON THE SHAPE OF THE LEARNING CURVES OF THE MULTIPLICATIVE AND ADDITIVE ARCHITECTURES

Our investigation through hyperparameter search of the performance of additive and mutliplicative networks matched in number of tuned parameters (by fixing the size of the additive network to $N = 300$ and the size of the multiplicative network to $N = 100$) suggests that they lead to similar errors on average when trained to produce the step motifs (main text Fig.1c). Interestingly, we observed that these similar performances were typically reached through different types of learning curves (Fig. 15): the multiplicative networks tend to show discontinuous steps in the learning curve, whereas the additive networks typically have smooth learning curves. This suggests that the performance of the multiplicative networks may be limited by the ability of gradient descent to fully optimize these networks to their best possible performance. In the future, it would be interesting to investigate ways of improving the optimization of the multiplicative networks - for instance using motif-specific learning rate schedules, or by increasing its size to $N = 300$ while only training one-third of the weights of the loop and input vectors (so that it has as many untuned parameters as the additive network).

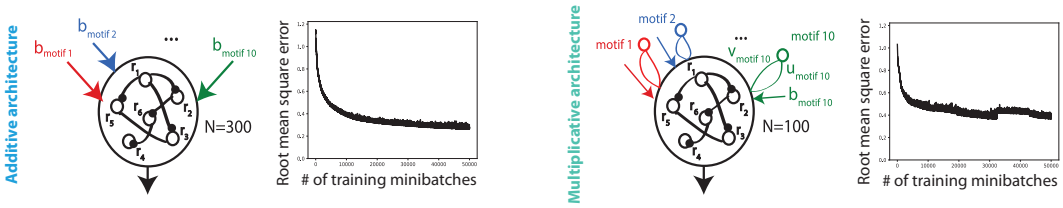

Figure 15: **Representative example learning curves for the additive and multiplicative networks**. These are chosen examples from the networks shown in the main text Fig.1c.