nature portfolio

https://doi.org/10.1038/s41586-022-04507-5

Supplementary information

Cortical ensembles orchestrate social competition through hypothalamic outputs

Methods

1. Subjects

Adult (8-10 week old at the time of first experimental procedure), wild-type male C57/BL6J mice from the Jackson Laboratory (RRID: IMSR_JAX:000664) were used for all experiments described. Mice were housed with randomly assigned littermates in cages of 3-4, maintained under a 12-hour reverse light/dark cycle, and given access to food and water *ad libitum* except where indicated in experimental proceedings. Experiments were conducted during the light-off light cycle phase. Experiments conducted at MIT were approved by the MIT IACUC while experiments conducted at the Salk Institute were approved by the Salk Institute IACUC. Sample size was not predetermined and instead estimated based on similar studies[31,32].

2. Surgery

Surgeries were performed aseptically using a digital small animal stereotaxic instrument (David Kopf Instruments). Anesthesia was induced with 5% isoflurane and mice maintained at 1%-3% isoflurane in the stereotactic frame, with a heating pad stabilizing body temperature. Immediately post-surgery, all animals were injected subcutaneously with 1.0mL saline for rehydration and moved to a clean cage. Animals were allowed a recovery period of ≥7 days prior to experimental procedures and resumption of tube test ranking.

A beveled 33-gauge microinjection needle was used for all injections, and virus was delivered at a rate of 100nL/min using a 1mL microsyringe (nanofil; WPI) with both a microsyringe pump (UMP3; WPI) and controller (Micro4; WPI). For all injections, coordinates were measured relative to bregma. For the optogenetic stimulation experiment mPFC->LH experimental mice received two 200nL injections of AAV5-ef1-DIO-hCHR2-eYFP into the mPFC (AP: +1.8 ML: +0.3 DV: -2 and -2.5), an optical fiber in mPFC (AP +1.8 ML +0.3 DV: -2.25) and CAV2-Cre mixed AAV5-CAMKIIa-mCherry for visualization (3:1) into the LH (AP: -0.7 & -1.0 ML: 2.86 DV: -5.0) 20° vertical angle. Identical parameters were used for non-opsin control animals except the virus used for mPFC injections was AAV5-ef1-DIO-eYFP. For the phototagging experiments, we used a red-shifted excitatory opsin ChrimsonR for better light penetrance with the wireless devices[3]. Mice received two injections of 200 nL of AAV8-hSyn-flex-ChrimsonR-tdtomato into the mPFC (AP: +1.8 ML: +0.3 DV: -2 and -2.5) and CAV2-Cre into the LH (AP: -0.7 & -1.0 ML: 2.86 DV: -5.0 at 20° vertical angle) or into the BLA (AP:-1.4 ML: 3.28 DV:-5.0). The optical fiber for phototagging experiments was placed at an angle to allow tethering the laser while the logger was connected. The mPFC fiber was implanted in AP: 1.8, ML-1.5 DV -1.3 with a 45° vertical angle. In addition, a ground wire was implanted in the left hemisphere touching the brain surface. Electrode implants and fiber implantation for the phototagging experiment were done in a second surgery 4 weeks after viral injection. The electrode consisted of a bundle of 32 single nichrome wires of 25 um diameter that was implanted into mPFC (AP: 1.8, ML: +0.5, DV: -2.25). All fibers and electrodes were secured using C&B-Metabond® Quick Adhesive Luting Cement (Parkell) and several additional layers of dental cement (Ortho-Jet powder, Lang Dental) were also applied.

3. Food Restriction

Subjects were fed *ad libitum* until initiation of experiments requiring food restriction (reward training). Prior to these experiments, subjects were weighed to the nearest tenth of a gram to attain a baseline for free-feeding body weight. During experiments requiring restriction, subjects were weighed daily and food-restricted to a limit of 85% of this baseline weight. Restriction was attained through limiting daily access to food pellets, providing approximately 2.5-4g/animal per day or 3 hrs of unlimited access to food. Animals had free access to water throughout.

4. Tube testing

After at least 1-week post-surgery recovery, mice were handled for 15-min/day for two days to gain familiarity with experimenters and reduce stress during experiments. Tube training began after handling. The mouse tube test assay was adapted from Lindzey and colleagues[34] and validated by Wang et al.[31] for social dominance in mice. Mice were individually trained to walk through the tube for at least two days, until they were comfortably walking across without resistance. For testing to determine rank, all mice in each cage were tested in a round-robin design in a randomized order. For each pair, mice were released at opposite ends of the tube simultaneously. The mouse that either itself backed out or was pushed out from the end where it was released was designated as “loser/subordinate” whereas the other mouse was designated as “winner/dominant.” Social ranks obtained with the tube test were considered stable when obtaining the same results for 3 or more days in a row.

5. Reward training and competition

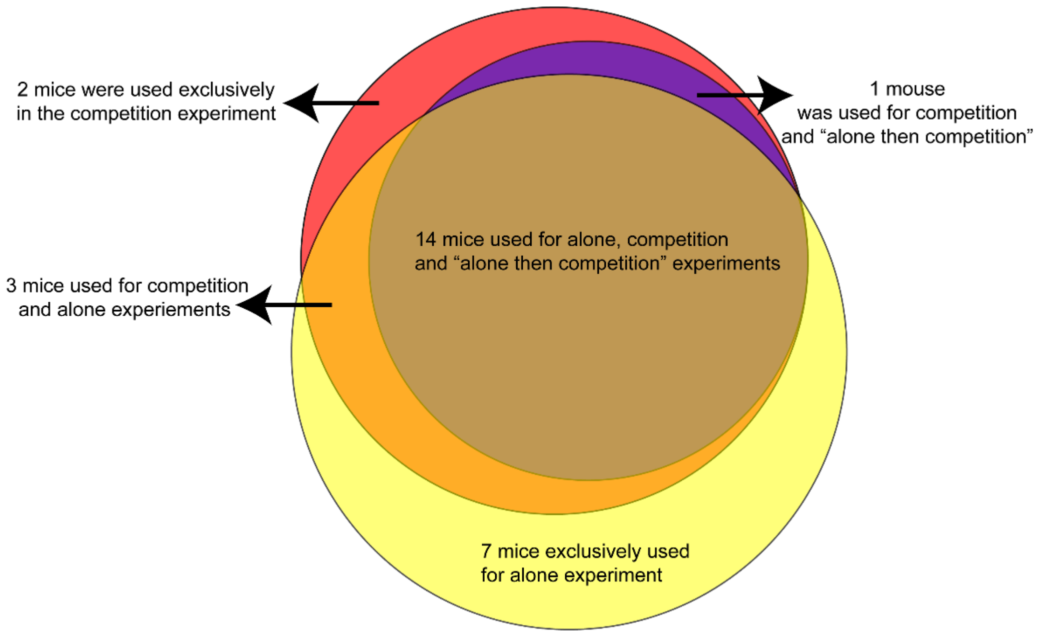
The reward competition assay was validated in unimplanted male mice (data presented in Fig. 1; n=8). Food-restricted mice were trained individually in preparation for the reward competition. During training sessions, single mice were placed in a MED-PC chamber with an adapted 3D-printed reward port. Trials began with presentation of a red LED located above the reward port and a 60dB, 5kHz tone, which both lasted for 10-seconds. A 10uL reward of vanilla Ensure® was delivered 2 seconds after tone onset. Inter-trial intervals (ITIs) lasted 60-100 seconds. A house light and white noise were on throughout the session. Training sessions consisted of at least 20 trials. Training consisted of 5-10 daily sessions for unimplanted mice, and 12-16 days for mice with recording devices. Training continued until all mice within a cage reached learning criteria of collecting the rewards with a mean latency of 5 seconds or less from tone onset. Reward port entries were detected through an electrical lickometer or infrared beam. After all mice in a cage attained learning criteria, we initiated reward competition. During all competition sessions, two mice were placed in one chamber. Mice competed against one cagemate per session. Competition sessions lasted 30-minutes (20 rewards) or 45-minutes (30 rewards). During the reward competition behavior, was recorded via webcams (at 25 or 30 fps). A custom-made MATLAB GUI was used to score pushing, resisting and displacement by a trained observer. The mouse that occupied the port for 3 seconds surrounding the reward delivery, or the first mouse to the port after reward delivery was considered the winner of the trial. Trials where these criteria were not met and thus who got the reward was unclear were excluded from analysis. AlphaTracker was used to track the nose, ears and tail base of both mice during the competition. For AlphaTracker analysis only videos with the same camera angle and zoom were pooled.

For optogenetic reward competition experiments, 11 cages of 4 mice each were injected and implanted with fibers in two separate experiments (data presented in Figure 4 and ED Fig 10). Cages were randomly assigned to experimental or control group. Scoring and analysis of behavior was done blinded to experimental group and rank when possible. All mice were reward trained but only intermediate or subordinate mice were stimulated during competition. After all animals in each cage reached criteria, competition sessions began. After 3 days of robin round competition, pairs of mice were selected for optogenetic reward competition experiments that met the following criteria: stable tube ranks and stable winning/losing relationship in the reward competition for two consecutive days or more, and winner being the higher ranking animal. These select pairs performed the reward competition two more additional consecutive days: one day for a light OFF competition and the other for a light ON competition for the relative subordinate animal only. For the light ON competition, the subordinate animal received laser stimulation in alternating epochs of 5 minutes of bursting stimulation (four 5 ms pulses at 100 Hz every 200 ms) and 10 minutes of light off. Both animals were tethered to the patch cords during training prior to the competition. To prevent the two patch cords from tangling we designed a commutator to split the laser into two with a dual commutator and then into single commutators for each individual mouse.

6. Reward competition and “alone” recording experiments

Wireless devices consisted in MiniLogger (SpikeGadgets) that recorded locally and digitized mPFC neural activity. During the reward competitions we recorded continuous signals from the 32 channels filtered for spikes (600-6,000 Hz) and a common average reference was applied using Trodes Software (SpikeGadgets). Tone times, port entries and laser pulses were detected with a main control unit that also triggered the miniloggers to record (Main Control Unit, SpikeGadgets). After recording sessions, logger data was extracted and merged with the Main Control Unit event related data. Spike sorting was done with Klusta[35] and subsequently verified by a trained observer for merging, deleting or splitting spike clusters.

For the mPFC recordings during the reward competition 24 new mice were implanted with electrodes in three batches of recording experiments. Out of those 24 mice only 20 mice had cells but the remaining 4 were still used for behavioral purposes as competitors. Additional animals were implanted (7 mice) and together with a subset of the mice used for competition (17 mice), had recording sessions after reaching training criteria while performing the task alone before ever having a social competition session (24 mice in total). Finally, in 15 out of the 20 mice used for competition recording we performed an additional experiment consisting in alone trials followed by competition trials (Extended Data Fig. 8i-j) in the same recording session. For those recording sessions, the mouse started the reward task alone and the experimenter opened the chamber after 10 to 15 trials and added a competitor who had simultaneously received the same number of trials alone in a different chamber. For all experiments, several times recording sessions were excluded because of battery failures. Competition sessions during recording experiments consistent of 30 trials to maximize the amount of data obtained, both mice always wore loggers to match conditions across competitors. However, in 6 mice we utilized first generation loggers that had shorter battery life and therefore sessions for those animals consisted in 20 trials. Electrode-implanted mice were trained to wear a weight matched dummy headstage (mimicking the wireless logger weight) during their individual reward training. Individual reward training continued until mice reached the mean 5 sec latency criteria.


2 mice were used exclusively in the competition experiment
1 mouse was used for competition and "alone then competition"
14 mice used for alone, competition and "alone then competition" experiments
3 mice used for competition and alone experiements
7 mice exclusively used for alone experiment

Venn Diagram indicating overlap of mice recorded across experiments. Most of the mice were used in all experiments, however, to consider absolute rank differences we added additional animals to the alone experiments.

Reward competition behavioral analysis

If the port was occupied during reward delivery, winning was assigned based on who was at the reward port for seconds 1.5-3 post tone onset. If there was complete switch of who was at the reward port during the period of reward delivery the trial was considered complicated and not assigned as a win nor lose. If the port was unoccupied during the reward delivery winning and losing was assigned based on latency to enter port post-delivery of reward. Reward competition data was scored via two methods: semi-automated based on distance to the port tracked using AlphaTracker and confirmed by trained observer or using a custom made Matlab GUI to score videos by a trained observer. Using the GUI we also scored pushing, resisting and displacing behavior, while relative location and distance to port were scored automatically via post processing of AlphaTracker tracking data. Pushing success was calculated as percent of pushing that resulted in displacing competitor. For the latency to port logger vs no logger comparison (Extended Data Fig. 5a), we used 12 animals that met the criteria of having reward alone sessions without logger within one week of reward alone sessions wearing loggers. Behavioral data across groups was compared using ANOVAs and t-test except when data was shown to be not normally distributed. For non-normal data we used Wilcoxon tests. Analyses in Extended Data Figs. 1c-d and 7 only include animals with the same weight recording device (logger), excluding a subset of mice that were recorded with an older version logger, while those in Extended Data Fig. 1e-g are paired analyses that include all mice.

Effort T-maze

The effort T-maze task was adapted from Walton et al. (2002)[36]. Mice were trained on a T-maze (main arm: 40x8x15cm, side arms 30x8x15cm) to associate one side with a high reward (HR; 20µL droplet of 100% vanilla Ensure®) and the other side with a low reward (LR; 20µL droplet of 25% vanilla Ensure mixed with water). The side with the high arm and low arm were counterbalanced across subjects. Training lasted nine days. Mice were first habituated to the T-maze apparatus with the cagemates where droplets of Ensure® were placed through the arms. The next four days, mice were individually given forced trials to sample the LR and HR arms and free choice trials to determine arm preference. Each training day always started with a forced sampling of both arms. After mice consistently showed preference for the high arm side, on day 5, a 7 cm wall was introduced in the HR arm. Mice were habituated to performing the task (including climbing the walls) while tethered to the patch cord. They were trained with forced and free choice trials with the 7 cm wall and on day 8 the HR arm wall was replaced with a 15cm wall. After two days of training with the 15 cm wall, on day 10 mice received stimulation on 4 out of 8 trials, in blocks of two trials.

8. Three-chamber Sociability Test

This assay was adapted from Brodkin, et al. (2014)[37]. For each trial, the subject was placed in a test chamber (57.15 x 22.5 x 30.5 cm), with the empty central compartment connected to one additional compartment of equal size on both sides. The compartments on both sides contained a single perforated cup. One cup was empty, whereas the other held a conspecific (novel juvenile) underneath. The subject was allowed to explore the chamber for a period of 10-min. Each subject underwent two social interaction tests separated by 24-hrs, with one test paired with optical stimulation and one without. All groups were counterbalanced for both order of light stimulation and a different novel juvenile was used on both days. EthoVision XT video tracking system (Noldus, Netherlands) was used to record and quantify time spent within each chamber.

9. Open Field Test

Subjects were tested individually, and placed in the center of a transparent, plastic chamber (53 x 53 cm) divided into a central and a peripheral field. The test consisted of a 10-min period divided into five 2-min epochs (OFF-ON-OFF-ON-OFF) in which the mouse freely investigated the chamber. Animals were recorded with a video camera and tracked for the duration of the test using EthoVision XT video tracking system.

10. Conditioned Place Preference (CPP)

A test chamber (57.15 x 22.5 x 30.5 cm) with a monochromatic wallpaper of different patterns on each side was used. On day 1, each mouse was allowed 15-min free exploration of the apparatus. Mice with strong side preference (≥60%) were excluded from days 2-3. On day 2, mice received two (≥2-hr separation) 30-min conditioning sessions while confined to one side of the chamber. Mice received optogenetic stimulation in one side only, counterbalancing for order and side across animals. On day 3, mice were allowed to explore freely for 45-min, the first 10-min used for analysis. All sessions were EthoVision-tracked and video-recorded.

11. Food Assay

Each animal was individually placed in a clean cage along with a single food pellet for 30 min, and stimulated in an epoch manner OFF-ON-OFF with 10 min epochs. Both animal and pellet were weighed before and after assay initiation. Sessions were EthoVision-tracked and recorded via side-webcam. All food assay experiments were done in the afternoon under dim lighting conditions (40 lux).

12. Optogenetic stimulation

Mice were stimulated with bursting 473nm light delivered in pulses of four 5 ms pulses at 100Hz in 5Hz intervals using a DPSS laser. Intensity was set to 10-11 mW at the end of each 200 nm patch cord (Thorlabs). For all experiments involving stimulation, all animals were tethered during training or habituation as well as testing.

For reward competition or alone sessions with phototagging, after the reward session, we tethered the patch cord to the fiber in mPFC and delivered 5 ms pulses at 10 or 20 Hz of red light (635 nm) at 8 mW (measured from patch cord). Neurons were identified as phototagged if they significantly increased firing rate with the photolatency threshold established during the ex-vivo experiment. For the in vivo data included in the manuscript, 3 out of 8 mice had mPFC-BLA phototagged cells and 3 out of 12 mice had mPFC-LH phototagged cells.

To calculate if cells were phototagged, mean firing rate in the 10 ms pre to vs fist 10 ms post light pulse presentation was calculated. Next, the latency of excitation was considered the first post light onset millisecond bin in which the mean firing rate was 4 standard deviations higher than the mean firing rate of the 10 ms baseline period. A given cell was considered phototagged if the mean firing rate in the 10ms pre vs post was significantly excited based on a Wilcoxon sign rank test with alpha=0.001 and latency of excitation was 7 ms or shorter.

13. Histology

After experiments, mice were anesthetized deeply with sodium pentobarbital (200 mg/kg; intraperitoneal injection) and transcardially perfused with 10 mL of Ringer's solution followed by 10 mL of cold 4% PFA in 1xPBS. Mice were decapitated and brains were extracted, fixed in 4% PFA for ≥24-hr at 4°C, then transferred to 30% sucrose in 1xPBS until they sank to the bottom of the solution. Coronal sections were then cut at 50 um and stored in 1xPBS at 4°C until processing. Sections were added to 25mL of 1xPBS and 2uL of DAPI and placed on a rocker at room temperature for 10-min. Sections were then mounted on glass microscope slides, cover-slipped with PVA-DABCO, and stored at room temperature.

14. Electrophysiology data analysis

Neural trajectories

Principal component analysis (PCA) was conducted to evaluate the population-level firing rate dynamics[38–41]. PCA is a commonly used dimensionality reduction tool that preserves the direction of largest variance in the data[38]. A single global PCA was done on a matrix containing all the data (concatenated mean firing rate per event type: win, lose, self port entry during tone, other port entry during tone, self port entry during ITI and other port entry during ITI) for all animals such that we could compare neural trajectories across groups[40]. This matrix had neurons in rows, and in the columns had mean firing rates during -5 to 5 seconds post task-relevant event using 100 ms bins. The neural trajectories for each task-relevant event were created per group by multiplying the coefficients obtained in the PCA by the mean firing rates across trials or port entries. For each trajectory, the geodesic length was calculated as the sum of Euclidean distances between adjacent 100 ms bins[42]. Distance between trajectories was calculated as the Euclidean distance between the two trajectories bin-by-bin. To allow for statistical comparisons, the neural trajectory metrics were calculated using the leave one out (LOO) method, leaving out all the neurons from a single animal per group, thus the number of iterations is the number of mice in that group. The n reported in trajectory quantifications corresponds to the number of mice utilized for the LOO. Importantly, in every iteration the same PCA coefficients per cell were used for the neural trajectory, since the PCA was done once prior to this step, but the neurons included varied. For visualization purposes we plotted the first 2 or 3 PC subspaces, but for quantification of trajectory lengths and distance between trajectories the first 148 PCs were used to capture 90% of the variance. For all trajectory visualizations and quantifications, we matched the number of neurons for each group to control for number of neurons being different across groups. For just the PC space visualizations, we smoothed the trajectories using MATLAB's smoothdata function. To compute the alignment, we used the following formulation as described by Miri et al.[43] say D1 and D2 are data matrices for the neural activity of neurons (rows being neurons and columns being time bins) under two different conditions (such as win vs lose trials), and P1 and P2 are the corresponding principal component vectors, then

$$alignment\ index = \frac{trace(P_1^T cov(D_2) P_1)}{trace(P_2^T cov(D_2) P_2)}$$

SVM classifier

To test if rank or competitive success could be decoded from single trial mPFC population activity, we used a support vector machine (SVM) with a Gaussian kernel. To obtain single trial mPFC population activity we used the coefficients obtained for each neuron in the global PCA and created a single trial neural trajectory using the firing rate for that trial. We trained the SVM using the first 8 PCs per trial and timepoints (-30 to 30 sec post cue or -5 to 5 sec post cue) as features and either rank or the competitive outcome of the trial as the label. We did a 10-fold cross-validation (CV), briefly, the data was split into 10 subsets and in each iteration the training consisted of a different 90% subset of the data and the testing was done with the remaining 10% of the data. We utilized the default hyperparameters as defined in MATLAB 2019b. Since the standard SVM is not probabilistic, we used an appropriate score-to-posterior-probability transformation function as proposed by Platt[44]. For the 10-fold CV, we computed the area under the receiver operating characteristic curve (AUC score) for the test data, since it's sensitive to class imbalance. Further, we tried to maintain a class balance wherever possible (e.g. win vs lose trials used). We used this model for a variety of decoding tests to decode competitive success, relative rank and absolute rank. For the decoding of absolute social rank in Extended Data Fig. 8, we used a one-vs-all strategy, where we generate one performance curve for each absolute rank by converting the multi-class problem into four binary problems (to test whether we can decode absolute rank 1 vs the rest, absolute rank 2 vs the rest and so on). Since we're using neural activity recorded from two mice competing in the task at a time to decode absolute social rank defined on a cage of four mice, we cannot be certain whether the decoding algorithms are predicting absolute social rank directly or are instead predicting a variable correlated to the absolute social rank, such as competition specific patterns in neural activity.

Firing rate analysis and hierarchical clustering for neural activity

To determine if a cell was significantly responding to an event, we compared the firing rate in a baseline period vs the event onset (5 second window for baseline and event) using a Wilcoxon sign rank test. If firing rate change was significant, excitation or inhibition was determined based on the average Z-score during the 5 second response window (if it was positive the cell was considered excited, while if it was negative the cell was considered inhibited). Note that the mean firing rate per trial type was smoothed using MATLAB's smoothdata function prior to z-score calculations for this analysis. Mean firing rate across events for the cue winning and losing trials, and port entries for self and other (tone and ITI) were concatenated per cell and arranged across animals in a matrix where column was time and row was neuron number. Using custom made MATLAB code, this matrix was clustered using agglomerative hierarchical clustering based on Euclidean distance and a threshold of 0.4 was used to determined functional clusters. For the hierarchical clustering cells that had a firing rate lower than 0.05 Hz during the baseline period were excluded from the analysis given that cells with zero spikes in baseline could not be z-scored such that we only used 913/998 cells. In Fig. 3 we only display neurons with strong responses to events (firing rate during event larger or equal to 2 Z-scores or smaller or equal to -1 Z-scores).

15. AlphaTracker

Architecture of AlphaTracker

Inspired by similar tools like DLC and SLEAP we turned to deep learning for multi-animal tracking[45–47], however, we took a top-down approach based on AlphaPose[48]. First, a convolutional neural network is trained to predict the whole mouse location as a bounding box. For this, YOLOv3[49] was adapted to achieve high detection speed. Then the bounding box becomes the input of an additional convolutional neural network Squeeze-and-Excitation Networks (SENet)[50] that tracks the body points of interest. Finally, to track animals across frames we use intersection overlap union (IOU). First, each animal's position and body parts are represented using a set of hierarchical bounding boxes. Then for two adjacent frames, we calculate the similarity of all descriptor pairs between these two frames according to formula 1.

$$Sim\left(D_i^t, D_i^{t+1}\right) = IOU\left(Box_i^t, Box_i^{t+1}\right) + \frac{1}{n}\sum_{k=1}^{n} IOU\left(P_{ik}^t, P_{jk}^{t+1}\right)$$

Formula 1

Where D_i^t is the descriptor of animal *i* at frame *t*, Box_i^t is the bounding box of animal *i* at frame *t* predicted by the convolutional neural network. P_{ik} is the box that wraps the kth body point of animal *i* at frame *t*. Intersection Overlap Union (IOU) (*,*) is defined as formula 2.

$$IOU\left(Box_{1,}Box_{2}\right) = \frac{AreaOfOverlap\left(Box_{1,}Box_{2}\right)}{AreaOfOverlap\left(Box_{1,}Box_{2}\right)}$$

Formula 2

The descriptor similarities are then sorted in descending order. The animal in the current frame and the animal in the previous frame that has the highest similarity are first matched and assigned with the same tracking ID. Then, for the remaining mouse the ones with has the second highest similarity are matched and so on until all animals in the current frame are assigned with a tracking ID or until no animals are left unmatched in the previous frame. Finally, a Kalman filter[51] is used to model the velocity and acceleration to predict the expected location of body points to try to overcome errors in cases with occlusion or tracking errors.

Evaluation of AlphaTracker

We randomly extract and labeled 800 frames from 4 different videos of two unmarked mice interacting in a clean cage, 200 of those frames were annotated by two people to determine human level accuracy. We split the data multiple times into non-overlapping sequences using different training-testing ratios (1:7, 1:3, 1:1 and 3:1) and trained 12 different networks with either 100, 200, 400 or 600 frames (3 different splits for each ratio), and tested the tracking quality using the remaining frames. Furthermore, we extracted 600 frames from a single video of four unmarked mice interacting in a clean cage, 200 of which were human-annotated to determine ground-truth values, and a subset of 100 were annotated by two different humans to calculate error across humans. We then trained 2 different networks with either 200 or 400 frames.

We quantified average precision[52,53], the root mean square error (RSME) of the different networks, and how they compared to the precision and error across two humans. RMSE, in this case, was defined as the Euclidean distance of pixels between a ground truth point and a network-predicted point. Importantly, RMSE was done after identity error correction in a subset of the frames coming from the same video to quantify accuracy of keypoints and percent identity error in the same dataset. We excluded 3 frames in which AlphaTracker failed to find one of the mice. Specifically, for each instance (animal) in a frame of video, the Euclidean distance (pixels) was calculated for each target point (head, left ear, right ear, and tail). Then, the distances for each body part were averaged across all animals in the current frame (either two or four mice). Lastly, this was performed across all evaluation frames coming from the same video and a final average was calculated as the average RMSE per frame across all body parts.

Training and data processing

Reward competition videos were annotated using the open source annotation tool Sloth (https://github.com/cvhciKIT/sloth). AlphaTracker was trained with at least 200 annotated images for a given camera angle and chamber. After training, AlphaTracker, tracked competition videos and generated x,y coordinates for both mice that were imported into MATLAB for further analysis. A human annotated the x,y location of the reward port for each video and that location was used to quantify distance to port and to normalize mouse locations for the 2D histogram of locations of the reward chamber.

Unsupervised clustering for behavioral motifs

AlphaTracker output was used to calculate features for behavioral motif clustering inspired by other tools[54]. We implemented two popular techniques; agglomerative hierarchical clustering to generate a dendrogram, and Uniform Manifold Approximation & Projection (UMAP)[55] to generate a manifold. The hierarchical clustering is a bottom-up technique of building clusters based on similarity between data points without having to specify the number of clusters and for which results are presented in a dendogram. In our case the clustering was done based on the Euclidean distance of a combination of behavioral features (see below) for every 500 ms clip. UMAP is a dimensionality reduction technique which preserves local as well as global structure in the data and defines an embedding that can be easily visualized in two dimensions. The data processing for these techniques was as follows: First, the data was split into 500 ms clips. For each video clip, we extract the features of the individual mouse and for frames where mice were close to each other we also extract features of the interaction between mice from the pose estimation and tracking result as well as the original image. For each clip we normalized the data by setting the origin of coordinates of all the poses (left ear, right ear, nose and tailbase) to the nose position of the target mouse in the middle frame of the clip. For the features shown in Extended Fig. 3 z-score normalization was applied using the mean across the whole clip. In addition, the direction of x axis of coordinates of all the poses was set to be the orientation of the target mouse's body vector in the middle frame of the clip. Fourier Transform was applied to the contour images of the mice. For each mouse the following features were used: length of head, length of body, angle between head and body, ear to nose length for each ear, ear-body angle for each ear, displacement of the mouse, angle of the body across frames, Fourier transform of the displacement and Fourier transform of the contour of the mouse. For social clustering, the following additional features were used: distance between noses, angles between mice, distance between tail of one mouse and nose of the other. All angles, distances and displacement are reported using polar coordinates. All these features were then used to cluster the AlphaTracker output using hierarchical clustering or UMAP.

16. Ex vivo electrophysiological recordings and analysis:

A separate cohort of mice (n= 6) was injected using the same viral strategy described above for unilateral ChrimsonR expression in mPFC-LH or mPFC-BLA neurons. Following 6 weeks of postoperative recovery (time-matched for behavioral recordings), mice were anesthetized with sodium pentobarbital (200 mg/kg-1; intraperitoneal injection), decapitated and brains were quickly extracted in oxygenated (95% O2 and 5% CO2) high sucrose cutting solution containing (in mM): 87 NaCl, 2.5 KCl, 1.3 NaH2PO4, 7 MgCl2, 25 NHCO3, 75 sucrose, 5 ascorbate, and 0.5 CaCl2 (~320-300 mOsm; pH = 7.3). Coronal sections (300 µm) containing the mPFC and LH or BLA were collected using a vibrating blade microtome (VT120; Leica, Buffalo Grove, IL). Slices were then incubated in oxygenated, modified artificial cerebrospinal fluid (ACSF) containing (in mM): 75 sucrose, 87 NaCL, 2.5 KCl, 1.3 NaH2PO4, 7 MgCl2, 0.5 CaCl2, 25 NaHCO3 and 5 ascorbic acid (~300 mOsm; pH =7.3) in a 31°C water bath for 1 hour before recordings were obtained.

Viral injections sites in the mPFC and LH or BLA for mPFC-LH and mPFC-BLA neurons, respectively, were confirmed. Whole-cell patch-clamp recordings were obtained from visually identified neurons expressing ChrimsonR (ChrimsonR+) and adjacent non-expressing neighboring neurons. Recordings of mPFC-LH and mPFC-BLA neurons were obtained using glass pipettes with a pipette resistance of 3-5 MΩ fabricated with a horizontal puller (P-1000, Sutter, CA) and filled with internal solution containing (in mM): 125 potassium gluconate, 20 HEPES, 10 NaCl, 3 MgATP, and 8 biocytin (~290mOsm; pH = 7.3). Recorded signals were acquired, amplified, and digitized at 10kHz using a Multiclamp 700B amplifier, Digidata144, and pClamp10 software (Molecular Devices, Sunnyvale, CA). Neurons were voltage-clamped at -70mV and given a 1 second constant red (589nm, 10mW light power at tip of fiber) light pulse to confirm ChrimsonR-mediated inward photocurrents. Neurons were then given a 1 Hz train of 5 ms red light pulses for 10 pulses in current-clamp under resting membrane conditions to assess photo-response latencies. Data were analyzed using Clampfit software (Molecular Devices, Sunnyvale, CA. The average photo-response latency of action potentials (APs) and excitatory postsynaptic potentials (EPSP) were measured as the time from light onset to the peak AP or EPSP for the 10 pulses for each cell.

17. Decoding of behavior

Dataset

The dataset used for modeling consisted of mPFC multiunit firing rate (X) and behavioral states (Y) determined from AlphaTracker output along with annotation from trained experts. Multiunit firing rate activity was determined by applying a common reference average and determining spikes using 4 standard deviations from the mean. Since behavioral conditions needed to be consistent across animals for the AlphaTracker tracking, we selected 13 animals with equivalent video conditions. Data from 13 mice was pooled and arranged as:

X = #trials x #channels x #time_stamps

Y = #trials x #time_stamps

The dataset consists of 965 trials, with spiking information from 32 channels across 70 time stamps of 200 ms intervals. For each time stamp, we have assigned one of 9 behavior labels, which served as the 'classes' for the classification problem (see Fig 1f). These classes were determined based on distance thresholds applied using AlphaTracker tracking output, and pushing, resisting and displaced were annotated by humans. The classification problem was to input the multi-unit activity sequence and obtain the corresponding behavior label sequence for the same trial.

We utilized five decoding models with increasing complexity, with the HMM-GLM being the most complex. The goal was to build a decoder by iteratively adding a richer feature space and complex relationships with the output behaviors. In each case, we have used a 10-fold cross-validation method, training with 90% of the trials and testing on the held-out 10% of trials, and computed the area under the receiver operating characteristic curve (AUC scores).

For the HMM-GLM, in addition to the 10-fold cross-validation, we also performed leave-one-animal-out (LOO) cross-validation to test the model's performance in a more stringent situation, the results from which are summarized in Extended Fig. 5j & 5k. In the LOO the model was trained with all mice but one held out mouse, and the model is trained on the held-out mouse, this was done 13 times, one per mouse.

Frequency Table (FT)

We constructed a table with the frequencies of the 9 behavioral labels normalized by the total occurrence of all of them. This helps generate fixed probability values for each of the behaviors, which is used as the likelihood for all time points, independent of the neural activity. This is the simplest possible classifier, solely based on the distribution of the behavior labels in the training set. The AUC scores for the FT model were close to chance (Fig. 1i).

Generalized Linear Model (GLM)

We next wanted to incorporate neural activity using a simple decoder. We used a Multinomial GLM with the logit as link function and with a 9-output softmax. The GLM takes the neural activity at each time point as input and produces a likelihood for the 9 behaviors as output. The AUC scores for the GLM model were significantly higher than chance, describing a relationship between the behavior labels and the neural activity (Fig 1i).

Support Vector Machine (SVM)

Our next strategy was to employ an SVM. Because the SVM with a Gaussian kernel was successful in decoding rank type as well as trial outcome, we utilized that. Since SVMs are traditionally binary classifiers, we utilized a one-vs-all strategy wherein for each of the 9 behavioral labels, we re-labeled the whole dataset as zero (not-that-label) or one (that-label) and then ran the model on the dataset with 10-fold cross-validation. The SVM takes the neural activity at each time point as input and produces a likelihood for the 9 behaviors as output. The AUC scores for the SVM model were close to chance (Fig 1i).

Ensemble of GLMs (9 GLMs)

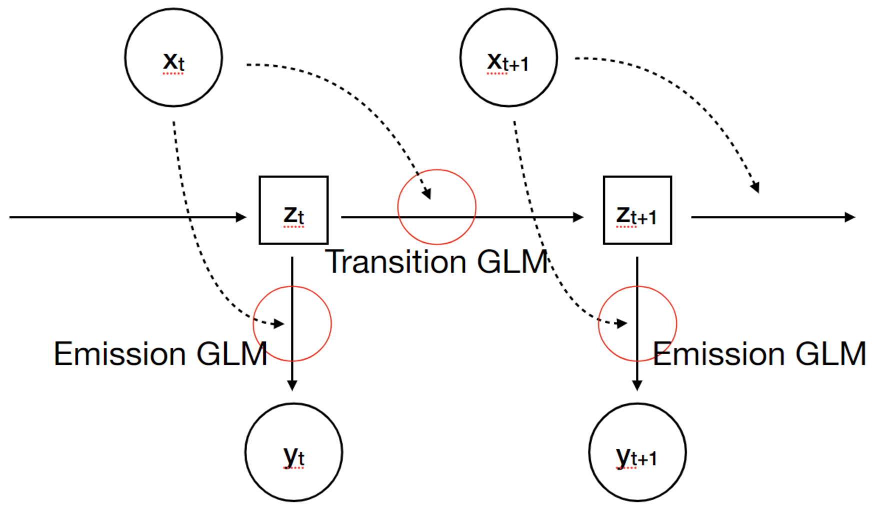
We then utilized an ensemble of 9 Multinomial GLMs, each with the logit as the link function, for the 9 behaviors. Since this meant that each GLM is trained independently to solve a binary class problem instead of one GLM to solve a multi-class problem, it seemed like a natural extension of the standard GLM. The AUC scores for this model were comparable to that of the single multi-class GLM, and the additional complexity was not useful (Fig 1i).

Hidden Markov Model – Generalized Linear Model (HMM-GLM)

We hypothesized that while the Multinomial GLM is able to partially capture mPFC dynamics, it still had two shortcomings. The first is the 'static' nature of the model and its inability to store past information when making predictions, which is key to common inference problems involving time series. The second is the simplistic nature of the model —which does not resemble the neural computation of the PFC —and hence is unable to utilize the statistical power of the novel trial structure used in this study. Moreover, the 9-class problem is a far more complex problem than a 2-class problem (rank or win/lose) with a much noisier dataset. This also explains the lack of improvement with 9-Multinomial-GLMs since it's still a linear and static model.

To incorporate temporal context into the model, we looked at the models proposed by Calhoun et al.[56] as well as Escola et al.[57] Calhoun et al. first introduced the idea of using a Hidden Markov Model (HMM) for modeling the change in the internal state of the organism as transitions of 'hidden states' and the neural spikes as emissions obtained from the hidden states. Escola and colleagues used Poisson GLMs for computing probabilities of each transition between hidden states as well as for determining emissions of neural spikes. The input (x_t) for the transition and emission steps were the time-varying stimuli and the emitted observations (y_t) was the firing rate of the neurons. This approach provided a mechanism where both the state transition probabilities as well as neural spike emission probabilities varied with time. Calhoun et al. proposed a modification of the HMM-GLM in which both the input and output were behaviors, with a logit GLM instead of Poisson. While trying to decode the behavior of a male *Drosophila* during courtship rituals, they fed certain physical feedback cues that the male observed in the female into the GLM layers. These feedback cues from the past served as the input (x_t) and the current behavior was predicted via the emissions (y_t). For our study, we modified the model further by having mPFC neural activity be the input to the GLM layers. The model formulation and training algorithm are similar to Escola et al. Note that the GLMs used in this model are binomial GLMs instead of multinomial. This is simply for the purpose of a simple interpretation for them. The primary differences are: 1) the input to the transition and state GLMs for the current time point is neural activity from the immediately previous time point only and not further back, 2) the GLM link function being a sigmoid/softmax has different gradients and dynamics as compared to the Poisson, 3) The outputs are categorical behaviors in our case instead of the binned spike counts and 4) we utilize an iterative optimization technique since the second order derivatives in our case are highly complex and computationally inefficient to use.

Model Architecture & Assumptions



Both the state transitions and output emissions follow the one-step Markov assumption:

$$P(z_{t+1}|z_1, z_2, \dots, z_t, y_1, y_2, \dots, y_t;\ x_1, x_2, \dots, x_t, \theta) = P(z_{t+1}|z_t; x_t, F_{tr})$$

$$P(y_t|z_1, z_2, \dots, z_t, y_1, y_2, \dots, y_{t-1}\ ;\ x_1, x_2, \dots, x_t, \theta) = P(y_t|z_t; x_t, F_{em})$$

where,

$z_t \epsilon \{1, 2, \dots, K\}$,

$y_t \epsilon \{1, 2, \dots, J\}$,

$\theta = set\ of\ all\ parameters$,

$x_t \epsilon \mathbb{R}^D$,

$F_{tr} =\ transition\ parameters$ and

$F_{em} = emission\ parameters$

(K = number of possible hidden states, J = number of behavioral labels, T = number of time stamps in a trial and D = number of dimensions/channels of the input).

The first assumption states that given the current hidden state, the future state is independent of both the past states as well as behavior outputs. The second assumption states that given the current hidden state, the current emission is independent of both the past states as well as behavior outputs. Furthermore, the neural activity is treated as a deterministic variable since it is fixed for any given trial.

Training

Let the data pertaining to one trial be concatenated into large matrices as:

$$X = [x_1, x_2, x_3, \dots, x_T]\ \epsilon\ \mathbb{R}^{D \times T}$$

$$Y = [y_1, y_2, y_3, \dots, y_T] \epsilon [1, 2, \dots, J]^{1 \times T}$$

$$Z = [z_1, z_2, z_3, \dots, z_T] \epsilon [1, 2, \dots, K]^{1 \times T}$$

where each array is either a single dimensional or multidimensional discrete time series or sequence.

Now, let us look at the parametrization of the model:

1) For state initialization:

$$P(z_1 = k; \Pi) =\ \pi_k$$

where: $\Pi = [\pi_1, \pi_2, \pi_3, \dots, \pi_K]$ represents the set of prior probabilities for all possible hidden states for the first time point in a sequence.

2) For state transitions:

$$\alpha^t_{z_t, z_{t+1}} \coloneqq P(z_t = k|z_{t-1} = k'; F_{tr})\ = \frac{\exp\left(F^T_{tr_{k',k}} \begin{bmatrix} x_t \\ 1 \end{bmatrix}\right)}{\sum_{k"=1}^{K} \exp\left(F^T_{tr_{k',k''}} \begin{bmatrix} x_t \\ 1 \end{bmatrix}\right)}$$

where: $F_{tr} \epsilon \mathbb{R}^{(D+1) \times K \times K}$ represents the array of filters (or weights) used for state transitions. For a transition from state m to state n, we use the weight vector $F_{tr_{m,n}}$. The data vector x_t is augmented with a ‘1’ at the end to incorporate a bias.

3) For emissions:

$$\eta^t_{z_t, y_t} \coloneqq P(y_t = j|z_t = k;\ F_{em})\ = \frac{\exp\left(F^T_{em_{k,j}} \begin{bmatrix} x_t \\ 1 \end{bmatrix}\right)}{\sum_{j'=1}^{J} \exp\left(F^T_{em_{k,j'}} \begin{bmatrix} x_t \\ 1 \end{bmatrix}\right)}$$

where: $F_{em} \epsilon \mathbb{R}^{(D+1)\times K\times J}$ represents the array of filters (or weights) used for output emissions. For an emission of output p from state q, we use the weight vector $F_{em_{q,p}}$

Thus, $\theta = \{\Pi, F_{tr}, F_{em}\}$ is the set of all parameters such that $|\theta| =$ total number of parameters $= K + (D+1)K^2 + (D+1)K$ scales linearly with the number of inputs and quadratically with the number of hidden states.

The introduction of the hidden states makes the estimation of the parameters an unsupervised learning process. We use an adapted version of the Baum-Welch implementation of the Expectation Maximization (E.M.) algorithm for this.

For one sample sequence, the complete data likelihood is described by:

$$P(Y, Z; \theta) = P(z_1; \theta) \prod_{t=2}^{T} P(z_t | z_{t-1}; \theta) \prod_{t=1}^{T} P(y_t | z_t; \theta)$$

$$= \pi_{z_1} \prod_{t=2}^{T} \alpha_{z_{t-1}, z_t}^{t-1} \prod_{t=1}^{T} \eta_{z_t, y_t}^{t}$$

While $P(Y, Z; \theta)$ represents the "complete" likelihood, it is a misnomer since the hidden state sequence is not known. This then becomes the *marginal* likelihood instead. The E.M. algorithm helps us maximize the true likelihood, i.e., using the law of total probability, $P(Y; \theta) = \sum_{Z \epsilon Z'} P(Y, Z; \theta)$ via iteratively maximizing the marginal likelihood in a roundabout manner. Here, $Z' = [1, 2, \dots, K]^{1\times T}$ is the set of all possible hidden state sequences.

E-Step:

Define: $Q(\theta; \theta^{old}) \coloneqq \mathbb{E}_{Z|Y;\theta^{old}}[\log L(\theta; Y, Z)]$ where $L(\theta; Y, Z) \coloneqq P(Y, Z; \theta)$ is the likelihood function.

In order to compute the objective function Q, known as the Baum Auxiliary Function, we utilize a pair of algorithms called the forward and backward algorithms. Note that we are referring to Q as the 'objective function' since it will be optimized in the M-Step of the E.M. algorithm.

<u>Forward Algorithm:</u>

Define: $a_k(t) \coloneqq P(y_1, y_2, \dots, y_t, z_t = k; \theta)$

Thus:

$$\begin{aligned} a_k(1) &= P(y_1, z_1 = k; \theta) \\ &= P(z_1 = k; \theta) P(y_1 | z_1 = k; \theta) \\ &= \pi_k \eta_{k, y_1}^{t} \end{aligned}$$

And:

$$a_k(t) = \eta_{k, y_t}^{t} \sum_{k'=1}^{K} a_{k'}(t-1) \alpha_{k',k}^{t-1}$$

<u>Backward Algorithm:</u>

Define: $b_k(t) \coloneqq P(y_{t+1}, y_{t+2}, \dots, y_T | z_t = k; \theta)$

Thus:

$b_k(T) = 1$, since it is deterministic

And:

$$b_k(t) = \sum_{k'=1}^{K} \alpha_{k,k'}^{t} \eta_{k',y_{t+1}}^{t+1} b_{k'}(t+1)$$

Using these relationships, we can compute the single and consecutive pairwise marginal probabilities of the posterior distribution of the state sequence given the emission sequence as:

$$\begin{aligned}\gamma(z_t = k) &\coloneqq P(z_t = k|Y;\theta) \\ &= \frac{a_k(t)b_k(t)}{P(Y;\theta)} \\ &= \frac{a_k(t)b_k(t)}{\sum_{k'=1}^{K} a_{k'}(t)b_{k'}(t)}\end{aligned}$$

$$\begin{aligned}\zeta(z_t = k, z_{t+1} = k') &\coloneqq P(z_t = k, z_{t+1} = k'|Y;\theta) \\ &= \frac{a_k(t)\alpha_{k,k'}^{t}\eta_{k',y_{t+1}}^{t+1} b_{k'}(t+1)}{P(Y;\theta)} \\ &= \frac{a_k(t)\alpha_{k,k'}^{t}\eta_{k',y_{t+1}}^{t+1} b_{k'}(t+1)}{\sum_{k''=1}^{K}\sum_{k'''=1}^{K} a_{k''}(t)\alpha_{k'',k'''}^{t}\eta_{k''',y_{t+1}}^{t+1} b_{k'''}(t+1)}\end{aligned}$$

We can now describe the whole posterior, $P(Z|Y;\theta)$, using these new definitions.

Returning to the likelihood function $L(\theta; Y, Z)$:

$$\begin{aligned}\log L(\theta; Y, Z) &= \log\big(P(Y, Z; \theta)\big) \\ &= \log \pi_{z1} + \sum_{t=2}^{T} \log \alpha_{z_{t-1},z_t}^{t-1} + \sum_{t=1}^{T} \log \eta_{z_t,y_t}^{t}\end{aligned}$$

Plugging all the expressions into the objective function $Q(\theta; \theta^{old})$:

$$\begin{aligned}Q\big(\theta;\theta^{old}\big) &= \mathbb{E}_{Z|Y;\theta^{old}}[\log L(\theta; Y, Z)] \\ &= \sum_{Z\epsilon Z'} P\big(Z\big|Y;\ \theta^{old}\big) \log L(\theta; Y, Z) \\ &= \sum_{k=1}^{K} \gamma^{old}(z_1 = k) \log \pi_k + \sum_{t=2}^{T}\sum_{k=1}^{K}\sum_{k'=1}^{K} \zeta^{old}(z_{t-1} = k, z_t = k') \log \alpha_{k,k'}^{t-1} + \sum_{t=1}^{T}\sum_{k=1}^{K} \gamma^{old}(z_t = k) \log \eta_{k,y_t}^{t}\end{aligned}$$

Up until this point, only one sample trial sequence has been considered. Say we have M number of trials. Then, under the i.i.d. (independent & identically distributed) assumption, the overall objective function becomes:

$$\begin{aligned}Q\big(\theta;\ \theta^{old}\big) = \sum_{m=1}^{M} \Bigg(&\sum_{k=1}^{K} \gamma^{m^{old}}(z_1^m = k) \log \pi_k + \sum_{t=2}^{T}\sum_{k=1}^{K}\sum_{k'=1}^{K} \zeta^{m^{old}}(z_{t-1}^m = k, z_t^m = k') \log \alpha_{k,k'}^{m^{t-1}} \\ &+ \sum_{t=1}^{T}\sum_{k=1}^{K} \gamma^{m^{old}}(z_t^m = k) \log \eta_{k,y_t}^{m^t} \Bigg)\end{aligned}$$

M-Step:

This step involves the optimization of the objective function using an iterative scheme. While several methods can be used to achieve this, we resorted to gradient ascent. We initialize Π (the prior) to chance and other elements of θ with a Gaussian distribution of zero mean and small standard deviation ($\mu = 0, \sigma = 0.001$), and then iteratively update θ as:

$\theta^{new} = \theta^{old} + \rho \nabla_{\theta} Q|_{\theta=\theta^{old}}$ where ρ is the learning rate.

In order to set a dynamic learning rate, we utilize a technique called Learning Rate Annealing, which is expressed as:

$\rho = \frac{\rho_0}{1+\frac{ep}{T_0}}$, where ρ_0 is the initial learning rate, ep is the number of epochs that have passed and T_0 is the annealing temperature. Thus, as more training epochs pass, the learning rate becomes smaller.

Before we can compute the gradients, we need to incorporate the Gaussian prior into our objective function. This would help regularize the expression for the assumption of the Gaussian distribution for our parameters.

$$Q(\theta;\, \theta^{old}) = \sum_{m=1}^{M}\left(\sum_{k=1}^{K} \gamma^{m^{old}}(z_1^m = k)\log \pi_k + \sum_{t=2}^{T}\sum_{k=1}^{K}\sum_{k'=1}^{K} \zeta^{m^{old}}(z_{t-1}^m = k, z_t^m = k')\log \alpha_{k,k'}^{m^{t-1}} + \sum_{t=1}^{T}\sum_{k=1}^{K}\gamma^{m^{old}}(z_t^m = k)\log \eta_{k,y_t}^{m^t}\right) - \frac{1}{2\sigma^2}\left(\sum_{k=1}^{K}\pi_k^2 + \sum_{k=1}^{K}\sum_{k'=1}^{K} F_{tr_{k,k'}}^{T} F_{tr_{k,k'}} + \sum_{j=1}^{J}\sum_{k=1}^{K} F_{em_{k,j}}^{T} F_{em_{k,j}}\right)$$

Finally, let us look at the gradients:

$\forall r \epsilon \{1, 2, \dots, K\}$, $\forall s \epsilon \{1, 2, \dots, K\}$ and $\forall l \epsilon \{1, 2, \dots, J\}$

$$\nabla_{\pi_r} Q(\theta;\, \theta^{old})$$
$$= \sum_{m=1}^{M}\sum_{k=1}^{K}\left(\frac{\delta_{k,r}}{\pi_k}\gamma^{m^{old}}(z_1^m = k)\right) - \frac{1}{\sigma^2}\sum_{k=1}^{K}\delta_{r,k}\pi_k$$
$$= \sum_{m=1}^{M}\frac{1}{\pi_r}\gamma^{m^{old}}(z_1^m = r) - \frac{1}{\sigma^2}\pi_r$$

$$\nabla_{F_{tr_{r,s}}} Q(\theta;\, \theta^{old})$$
$$= \sum_{m=1}^{M}\sum_{t=2}^{T}\sum_{k=1}^{K}\sum_{k'=1}^{K}\left(\delta_{k,r}\left(\delta_{k',s} - \frac{\sum_{k''=1}^{K}\exp\left(F_{tr_{k,k''}}^{T}\begin{bmatrix}x_t^{\,m}\\1\end{bmatrix}\right)\delta_{k'',s}}{\sum_{k''=1}^{K}\exp\left(F_{tr_{k,k''}}^{T}\begin{bmatrix}x_t^{\,m}\\1\end{bmatrix}\right)}\right)\zeta^{m^{old}}(z_{t-1}^m = k, z_t^m = k')\begin{bmatrix}x_t^{\,m}\\1\end{bmatrix}\right) - \frac{1}{\sigma^2}\sum_{k=1}^{K}\sum_{k'=1}^{K}\delta_{k,r}\delta_{k',s}F_{tr_{k,k'}}$$
$$= \sum_{m=1}^{M}\sum_{t=2}^{T}\left(\zeta^{m^{old}}(z_{t-1}^m = r, z_t^m = s) - \alpha_{r,s}^{m^{t-1}}\gamma^{m^{old}}(z_{t-1}^m = r)\right)\begin{bmatrix}x_t^{\,m}\\1\end{bmatrix} - \frac{1}{\sigma^2}F_{tr_{r,s}}$$

Note that in the previous step, we have used the following algebraic manipulation:
$\sum_{k'=1}^{K} P(z_{t-1}^m = r, z_t^m = k'|\theta) = P(z_{t-1}^m = r|\theta) \coloneqq \gamma(z_t^m = r)$.

$$\nabla_{F_{em_{r,l}}} Q(\theta; \theta^{old})$$

$$= \sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{k=1}^{K}\left(\delta_{k,r}\left(\delta_{y_t^m,l} - \frac{\sum_{j=1}^{J}\exp\left(F_{em_{k,j}}^T\begin{bmatrix}x_t^m\\1\end{bmatrix}\right)\delta_{j,l}}{\sum_{j=1}^{J}\exp\left(F_{em_{k,j}}^T\begin{bmatrix}x_t^m\\1\end{bmatrix}\right)}\right)\gamma^{m^{old}}(z_t^m = k)\begin{bmatrix}x_t{}^m\\1\end{bmatrix}\right) - \frac{1}{\sigma^2}\sum_{j=1}^{J}\sum_{k=1}^{K}\delta_{k,r}\delta_{j,l}F_{em_{k,j}}$$

$$= \sum_{m=1}^{M}\sum_{t=1}^{T}\left(\left(\delta_{y_t^m,l} - \eta_{r,l}^{m^t}\right)\gamma^{m^{old}}(z_t^m = r)\begin{bmatrix}x_t{}^m\\1\end{bmatrix}\right) - \frac{1}{\sigma^2}F_{em_{r,l}}$$

Where $\delta_{i,j} = 1\; if\; i = j\; and\; 0\; otherwise$ is the Kronecker delta.

Since a closed-form solution for these updates would be very complicated to obtain, we relied on numerical computing to optimize the objective function and train the model.

We followed the tips laid out by Rabiner[58] to deal with problems of numerical overflow/underflow while computing the forward and backward steps of the training algorithm. We also binned all probability distributions to at least 10^{-300}, to prevent numerical errors that MATLAB's precision levels could not handle.

Cross-validation & Model Selection

Since we had to manually select the number of hidden states to be used, we resorted to a 10-fold cross validation mechanism to remove any biases in the process. In each fold, we trained the model on 9/10ths of the data, and then computed the likelihood on the 10th held-out part. We then plotted the log-likelihoods (trial average) of the held-out data from all the folds as a function of number of hidden states with standard error of the mean as error bars (Extended Data Fig. 5j). We found that as you increase the number of hidden states, the log likelihood increases, reaching a plateau at the 6-state model.

Testing:

Identifying the most probable hidden state sequence.

Since $P(z_t = k|Y;\theta) = \gamma(z_t = k)$, then for a given trial, $z_t^* = \arg\max_k \gamma(z_t = k)$ is the most probable hidden state for time t.

Predicting the behavior of mice.

We utilized the one-step-prediction method to decode the behavior of mice from neural activity. In short, the information from prior time point was indirectly used together with neural activity of the current time point to predict the behavioral label. For the time points of a given trial the most probable hidden state of the previous time point was stored. This helped compute the hidden state distribution for each time point.

For $t = 1$, there is no hidden state prior to this time. However, the distribution of hidden state is given by Π.

$\forall\ t \in \{2, 3, \ldots, T\}$, say the most probable hidden state for the previous time point is given by z_{t-1}^*. Now:

$$P(y_t|y_1, y_2, \ldots, y_{t-1};\ x_1, x_2, \ldots, x_t, \theta\ )$$
$$= \sum_{k=1}^{K} P(y_t, z_t = k|y_1, y_2, \ldots, y_{t-1}; x_1, x_2, \ldots, x_t, \theta)$$
$$= \sum_{k=1}^{K} P(y_t|y_1, y_2, \ldots, y_{t-1}, z_t = k; x_1, x_2, \ldots, x_t, \theta)\, P(z_t = k|y_1, y_2, \ldots, y_{t-1}; x_1, x_2, \ldots, x_t, \theta)$$
$$= \sum_{k=1}^{K} P(y_t|z_t = k; x_t, \theta) \sum_{k'=1}^{K} P(z_t = k, z_{t-1} = k'|y_1, y_2, \ldots, y_{t-1}; x_1, x_2, \ldots, x_t, \theta)$$

$$= \sum_{k=1}^{K} P(y_t|z_t = k; x_t, \theta) \sum_{k'=1}^{K} P(z_t = k|y_1, y_2, \dots, y_{t-1}, z_{t-1} = k'; x_1, x_2, \dots, x_t, \theta)P(z_{t-1} = k'|y_1, y_2, \dots, y_{t-1}; x_1, x_2, \dots, x_t, \theta)$$

$$= \sum_{k=1}^{K} P(y_t|z_t = k; x_t, \theta) \sum_{k'=1}^{K} P(z_t = k|z_{t-1} = k'; x_{t-1}, \theta)P(z_{t-1} = k'|y_1, y_2, \dots, y_{t-1}; x_1, x_2, \dots, x_t, \theta)$$

Since z_{t-1}^* is the most probable value for z_{t-1}, the probabilities $P(z_{t-1} \neq z_{t-1}^*|y_1, y_2, \dots, y_{t-1}; x_1, x_2, \dots, x_t, \theta) = 0$ while that for $z_{t-1} = z_{t-1}^*$ would be unity.

Thus:

$$P(y_t = j|y_1, y_2, \dots, y_{t-1}; x_1, x_2, \dots, x_t, \theta) = \sum_{k=1}^{K} P(y_t = j|z_t = k; x_t, \theta)P(z_t = k|z_{t-1} = z_{t-1}^*; x_{t-1}, \theta)$$

This is how we generated the likelihoods for all the behavior labels for time t. The behavior label with the highest likelihood was selected as the most probable behavior for time t, i.e.,

$$y_t^* = \arg\max_j P(y_t = j|y_1, y_2, \dots, y_{t-1}; x_1, x_2, \dots, x_t, \theta).$$

Further we selected the state that leads to the highest likelihood for the correct behavior as compared with the ground truth as the most probable hidden state for time t, i.e.,

$$z_t^* = \arg\max_k P(y_t = y_t^{true}|z_t = k; x_t, \theta)P(z_t = k|z_{t-1} = z_{t-1}^*; x_{t-1}, \theta)$$

In this way we obtain the likelihoods for the behavior label for all the time points of a trial in an inductive manner.

Lastly, we observed a time lag of one time bin (200 ms) in the model output predictions. Since this was present uniformly across the entire dataset, we simply shifted the model output backwards in time by one bin.

Data Availability Statement:

Data will be made upon reasonable request to corresponding authors.

References:

31. Wang, F. *et al.* Bidirectional Control of Social Hierarchy by Synaptic Efficacy in Medial Prefrontal Cortex. *Science* **334**, 693–697 (2011).

32. Zhou, T. *et al.* History of winning remodels thalamo-PFC circuit to reinforce social dominance. *Science* **357**, 162–168 (2017).

33. Klapoetke, N. C. *et al.* Independent optical excitation of distinct neural populations. *Nat. Methods* **11**, 338–346 (2014).

34. Lindzey, G., Winston, H. & Manosevitz, M. Social Dominance in Inbred Mouse Strains. *Nature* **191**, 474 (1961).

35. Rossant, C. *et al.* Spike sorting for large, dense electrode arrays. *Nat. Neurosci.* **19**, 634–641 (2016).

36. Walton, M. E., Bannerman, D. M. & Rushworth, M. F. The role of rat medial frontal cortex in effort-based decision making. *J. Neurosci.* **22**, 10996–11003 (2002).

37. Brodkin, E. S., Hagemann, A., Nemetski, S. M. & Silver, L. M. Social approach-avoidance behavior of inbred mouse strains towards DBA/2 mice. *Brain Res.* **1002**, 151–157 (2004).

38. Cunningham, J. P. & Yu, B. M. Dimensionality reduction for large-scale neural recordings. *Nat. Neurosci.* **17**, 1500–1509 (2014).

39. Gao, P. & Ganguli, S. On simplicity and complexity in the brave new world of large-scale neuroscience. *Curr. Opin. Neurobiol.* **32**, 148–155 (2015).

40. Linsenbardt, D. N., Timme, N. M. & Lapish, C. C. Encoding of the Intent to Drink Alcohol by the Prefrontal Cortex is blunted in Rats with a Family History of Excessive Drinking. *eNeuro* **6**, (2019).

41. Williamson, R. C., Doiron, B., Smith, M. A. & Yu, B. M. Bridging large-scale neuronal recordings and large-scale network models using dimensionality reduction. *Curr. Opin. Neurobiol.* **55**, 40–47 (2019).

42. Shifrin, T. Differential geometry: a first course in curves and surfaces. *Univ. Ga.* (2015).

43. Miri, A. *et al.* Behaviorally selective engagement of short-latency effector pathways by motor cortex. *Neuron* **95**, 683–696 (2017).

44. Platt, J. C. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. in *Advances in Large Margin Classifiers* 61–74 (MIT Press, 1999).

45. Pereira, T. D. *et al.* SLEAP: Multi-animal pose tracking. *bioRxiv* 2020.08.31.276246 (2020) doi:10.1101/2020.08.31.276246.

46. Lauer, J. *et al. Multi-animal pose estimation and tracking with DeepLabCut*. 2021.04.30.442096 https://www.biorxiv.org/content/10.1101/2021.04.30.442096v1 (2021) doi:10.1101/2021.04.30.442096.

47. Segalin, C. *et al. The Mouse Action Recognition System (MARS): a software pipeline for automated analysis of social behaviors in mice*. 2020.07.26.222299 https://www.biorxiv.org/content/10.1101/2020.07.26.222299v1 (2020) doi:10.1101/2020.07.26.222299.

48. Fang, H.-S., Xie, S., Tai, Y.-W. & Lu, C. RMPE: Regional Multi-person Pose Estimation. in *2017 IEEE International Conference on Computer Vision (ICCV)* 2353–2362 (IEEE, 2017). doi:10.1109/ICCV.2017.256.

49. Redmon, J. & Farhadi, A. Yolov3: An incremental improvement. *ArXiv Prepr. ArXiv180402767* (2018).

50. Hu, J., Shen, L. & Sun, G. Squeeze-and-excitation networks. in *Proceedings of the IEEE conference on computer vision and pattern recognition* 7132–7141 (2018).

51. Kalman, R. E. A new approach to linear filtering and prediction problems. (1960).

52. Pishchulin, L. *et al.* DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation. *ArXiv151106645 Cs* (2016).

53. Andriluka, M. *et al.* PoseTrack: A Benchmark for Human Pose Estimation and Tracking. (2017).

54. Wiltschko, A. B. *et al.* Mapping Sub-Second Structure in Mouse Behavior. *Neuron* **88**, 1121–1135 (2015).

55. McInnes, L., Healy, J., Saul, N. & Großberger, L. UMAP: Uniform Manifold Approximation and Projection. *J. Open Source Softw.* **3**, 861 (2018).

56. Calhoun, A. J., Pillow, J. W. & Murthy, M. Unsupervised identification of the internal states that shape natural behavior. *Nat. Neurosci.* **22**, 2040–2049 (2019).

57. Escola, S., Fontanini, A., Katz, D. & Paninski, L. Hidden Markov models for the stimulus-response relationships of multistate neural systems. *Neural Comput.* **23**, 1071–1132 (2011).

58. Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* **77**, 257–286 (1989).