Video Article

Automated, Quantitative Cognitive/Behavioral Screening of Mice: For Genetics, Pharmacology, Animal Cognition and Undergraduate Instruction

Randy Gallistel[1], Fuat Balci[1,2], David Freestone[1,3], Aaron Kheifets[1], Adam King[4]

[1]Department of Psychology, Rutgers University

[2]College of Social Science and Humanities, Koc University

[3]Center for Neural Science, New York University

[4]Department of Mathematics & Computer Science, Fairfield University

Correspondence to: Randy Gallistel at galliste@ruccs.rutgers.edu

URL: http://www.jove.com/video/51047
DOI: doi:10.3791/51047

Keywords: Empty Value, Issue, Behavior genetics, cognitive mechanisms, behavioral screening, learning, memory, timing

Date Published: 9/16/2013

Citation: Gallistel, R., Balci, F., Freestone, D., Kheifets, A., King, A. Automated, Quantitative Cognitive/Behavioral Screening of Mice: For Genetics, Pharmacology, Animal Cognition and Undergraduate Instruction. *J. Vis. Exp.* (), e51047, doi:10.3791/51047 (2013).

Abstract

We describe a high-throughput, high-volume, fully automated, live-in 24/7 behavioral testing system for assessing the effects of genetic and pharmacological manipulations on basic mechanisms of cognition and learning in mice. A standard polypropylene mouse housing tub is connected through an acrylic tube to a standard commercial mouse test box. The test box has 3 hoppers, 2 of which are connected to pellet feeders. All are internally illuminable with an LED and monitored for head entries by IR beams. Mice live in the environment, which eliminates handling during screening. They obtain their food during two or more daily feeding periods by performing in operant (instrumental) and Pavlovian (classical) protocols, for which we have written protocol-control software and quasi-real-time data analysis and graphing software. The data analysis and graphing routines are written in a Matlab-based language created to simplify greatly the analysis of large time-stamped behavioral and physiological event records and to preserve a full data trail from raw data through all intermediate analyses to the published graphs and statistics within a single data structure. The data-analysis code harvests the data several times a day and subjects it to statistical and graphical analyses, which are automatically stored in the "cloud" and on in-lab computers. Thus, the progress of individual mice is visualized and quantified daily. The data-analysis code talks to the protocol-control code, permitting the automated advance from protocol to protocol of individual subjects. The behavioral protocols implemented are matching, autoshaping, timed hopper-switching, risk assessment in timed hopper-switching, impulsivity measurement, and the circadian anticipation of food availability. Open-source protocol-control and data-analysis code makes the addition of new protocols simple. Eight test environments fit in a 48x24x78" cabinet; two such cabinets (16 environments) may be controlled by one computer.

Introduction

To bring the powerful techniques of genetics, molecular genetics, molecular biology and neuropharmacology to bear on elucidating the cellular and molecular mechanisms that mediate basic mechanisms of cognition, we need high-volume, high-through-put psychophysical screening methods that quantify physiologically meaningful properties of cognitive mechanisms. A psychophysically measurable, physiological meaningful quantitative property of a mechanism is a property that can be measured by behavioral means and also by electrophysiological or biochemical means. Examples are the absorption spectrum of rhodopsin, the free-running period of the circadian clock, and the refractory period of reward axons in the medial forebrain bundle[1,2]. Psychophysical measurements that can be compared to cellular and molecular measurements lay a foundation for linking cellular and molecular mechanisms to psychological mechanisms through quantitative correspondence. For example, the fact that the in situ absorption spectrum of the rhodopsin in the outer segments of rods superimposes on the human scotopic spectral sensitivity function is strong evidence that the photon-triggered isomerization of rhodopsin is the first step in scotopic vision. The quantitative aspects of complex behavior patterns are also central to the use of QTL methods in behavioral genetics[3,4]

The performance of mice (and rats) on well-established instrumental and Pavlovian learning protocols depends on brain mechanisms that measure abstract quantities like time, number, duration, rate, probability, risk and spatial location. For example, the speed of acquisition of Pavlovian conditioned responses depends on the ratio between the average interval between the reinforcing events (typically, food deliveries) and the average latency to reinforcement following the onset of the signal for impending reinforcement[5-7]. For a second example, the ratio of the average duration of the visits to two feeding hoppers in a matching protocol approximately equals the ratio of the rates of reinforcement at those two hoppers[8-10].

The behavioral testing methods currently in wide use by neuroscientists interested in underlying mechanisms are, for the most part, low volume, low through-put, and labor intensive[26]. Moreover, they do not measure quantities that can be compared with quantities measured by electrophysiological and biochemical methods, as, for example, the behaviorally measured periods and phases of circadian oscillators may be compared to electrophysiological and biochemical measures of circadian period and phase. Current behavioral testing methods focus on categories of learning, such as spatial learning, temporal learning, or fear learning, rather than on underlying mechanisms. The widely used water maze test of spatial learning[11-15] is an example of these shortcomings. Spatial learning is a category. Learning in that category depends

on many mechanisms, one of which is the mechanism of dead reckoning[16,17]. Dead reckoning depends in turn on the odometer, the mechanism that measures distance run[18]. Similarly, temporal learning is a category. A circadian clock is among the mechanisms on which learning in that category depends, because an oscillator with an approximately 24-hr period is required for animals to learn the time of day at which events occur[17,19]. The clock that enables food anticipation has yet to be discovered[19].

A clock is a time-measuring mechanism. Endogenous oscillators with a wide range of periods allow the brain to locate events in time by recording the phases of those clocks[16,17]. The ability to record locations in time enables the measurement of durations, that is, distances between locations in time. Associative learning depends on the brain's measurements of durations[5,6,20,21]. Counters are number-measuring mechanisms. Number measuring enables probability estimation, because a probability is the proportion between the numerosity of a subset and the numerosity of the superset. Number measuring and duration measuring enable rate estimation, because a rate is the number of events divided by the duration of the interval over which that number was measured. Measurements of duration, number, rate and probability enable behavioral adjustments to changing risks.[22,23] Our method focuses on measuring the accuracy and precision of these foundational mechanisms. Accuracy is the extent to which the brain's measure corresponds to an objective measure. Precision is the variation or uncertainty in the brain's measure of a fixed objective value, for example, a fixed duration. Weber's Law is the oldest and most securely established result in psychophysics. It asserts that the precision of the brain's measure of a quantity is a fixed fraction of that quantity. The Weber Fraction, which is the statistician's coefficient of variation in a distribution, measures precision. The ratio of the psychophysical mean (e.g., mean judged duration) to the objective mean (mean objective duration) is the measure of accuracy.

The method presented here maximizes volume (number of animals being screened at any one time in a given amount of lab space) and throughput (amount of information obtained divided by the average duration of the screening of a single animal) while minimizing the amount of human labor required to make the measurements and maximizing the immediacy with which the results of the screening become known.

The data-analysis software architecture presented here automatically puts the raw data and all the summary results and statistics derived from the data together in a single data structure, with field headings that render intelligible the vast seas of numbers therein contained. The analytic software only operates on data in that structure, and always stores the results of its operations in fields within that same structure. This insures an intact trail from raw data to published summaries and graphs.

The software automatically writes into the structure the experiment-control programs that governed the fully automated testing, and it automatically indicates which raw data came from which program. Thus, it preserves an impeccable data trail, with no doubt as to which experimental conditions were in force for each animal at each point in the testing and no doubt about how the summary statistics derive from the raw data. This method of data preservation greatly facilitates the development of standardized behavioral screening data bases, making it possible for other laboratories to further analyze these rich data sets.

This method minimizes the risk of loss of support for the firmware and software on which it depends. The testing apparatus is trivially modified from a long-established commercial source. The programming languages are the custom language provided by the hardware manufacturer, for protocol control, and, for data analysis and graphing, a purpose-built, non-commercial, open-source toolbox (TSsystem) written in a very widely supported commercial scientific programming, data analysis and graphing language. The toolbox contains high-level commands for extracting structural information and summary statistics from lengthy time-stamped event records. The protocol-implementing programs and the data-analyzing programs are open source and thoroughly documented.

The screening system is schematized in **Figure 1**. Ten cabinets, each containing 8 test environments may be set up in a 10' x 15' laboratory room, enabling 80 mice to be run at one time. Cables passing through a port in a party wall should connect the environments to the electronic/electrical interface cards and PCs in another room. The PCs run the protocol-control programs. One computer is required for every 2 cabinets (16 test environments). The PCs must be connected via a Local Area Network to a server running the data-analysis and graphing software.

Protocol

The three fully automated protocols in the TSsystem (matching, appetitive instrumental and classical conditioning) and the switch protocol have been approved by the Animal Care and Facilities Committee at Rutgers New Brunswick.

Setting up the physical system

1. Set up the test environments in the cabinets (see **Figure 1**)
2. Install the experiment-control software provided with the test environments on the protocol-control computers.

   Notes: Do not use these computers for any other purpose!

Setting up the software system

1. Set up the LAN (local area network) so that the server on which data-analysis software is installed can access the hard disks of the computer(s) controlling the test environments (see **Figure 1**).
2. Establish a file-synchronizing account for data storage in the "cloud".
3. Put the TSsystem folder and its subfolders on the search path of the commercial programming language in a cloud-synchronized folder.

   Notes: The TSsystem is a software toolbox, that is, a library of over 30 high-level functions that facilitate the creation of complex data-analysis and data-graphing code that automatically processes the data whenever it is harvested from the output files generated by the experiment-control program. All of the commands operate on data in fields of the Experiment structure and put the results in other fields in the

same structure (see **Figure 2**). These open-source commands are written in one of the most widely used commercial scientific programming and graphing languages. It has many other "toolboxes", including most usefully a statistics toolbox.

Starting an Experiment

1. Call TSbegin (see **Figure 3**)

   Notes: TSbegin is an interactive GUI (Graphic User Interface) in the TSsystem toolbox. It leads the user through the process of creating the hierarchical data structure into which the raw data and all results derived from it will be placed by the other functions in the TSsystem toolbox.
2. Call TSaddprotocol (see **Figure 4**)

   Notes: TSaddprotocol is a GUI in the TSsystem toolbox. It leads the user through the process of specifying control parameters for an experimental protocol, specifying decision code that will automate the decision to terminate the protocol and go on to the next one, and specifying the decision criteria to be used.
3. Place mice in the 24/7 live-in test environments, one mouse per environment.

   Notes: Take care to note the ID number of the mouse that goes into each of the numbered experimental environments (Box 1, Box 2, etc). Also, note the letter that identifies the experiment-control computer on the Local Area Network (LAN) and its IP address.
4. Call TSstartsession (**Figure 5**)

   Notes: TSstartsession is a GUI in the TSsystem toolbox . It leads the user through the process of starting an experimental session. Experimental sessions last one or two weeks, during which several different behavioral testing protocols are run. TSstartsession stores the information that goes into the macro that the protocol-control software reads when a session is started. Included is the path to and name of the code file that the protocol-control software reads. TSsystem's analytic software reads this code into the hierarchical data structure, so there is never doubt as to the exact protocol in force at any time.
5. Go to the control computers and call the ~~macro for each box controlled by a given computer, thereby activating the control protocol for that box~~.

Data Analysis

1. If you have created a new protocol, write appropriate data-analysis and graphing code using the commands in the TSsystem toolbox, which greatly simplify the creation of complex data analyses

   Notes: Data-analysis and graphing code for the three protocols whose results are described below are included in the TSsystem toolbox. Because, they are open source, they may be modified at will. The code for these analyses is extensively commented, which makes it easier to create code for analyzing the results from user-specified protocols.
2. For the duration of the experiment (24 hrs to many weeks), monitor email for alerts from the server indicating possible equipment malfunctions (power failures, spontaneous, control-computer reboots, pellet feeder malfunctions, etc), which the TSsystem data-analyzing program detects.
3. Study the plots of performance that the data-analyzing code written in TSsystem produces every time it is called by the analysis timer (typically 2 - 4 times per day).

   Notes: The analysis timer calls the data-analysis and graphing program at user-specified intervals. The called program is written with functions in TSsystem. It reads into the hierarchical data structure the raw data harvested from the file to which the protocol-control software writes. Then, it analyzes the data and graphs the results of the analyses. The file containing the hierarchical data structure is stored in a file-synchronization folder in the cloud. This provides automatic off-site backup. The automatic file-synchronization stores copies of the structure file on the computers of all the personnel and collaborators who have been granted access. Specified graphs are automatically emailed to specified personnel and collaborators. A principal investigator can monitor the progress of the testing from anywhere in the world at any time, and, if necessary, revise the experimental protocol, on line, remote from the site where the mice are being tested.
4. Use TSbrowser to study the data and summary statistics in the hierarchical data structure as they become available, in quasi real time (see **Figure 2**).

Representative Results

The system can and should be used to run protocols tailored to the aims of the individual investigator or classroom teacher. However, we have developed a suite of 3 protocols that should prove useful in large-scale screening of genetically manipulated mice and large-scale pharmacological testing: the matching protocol, the 2-hopper autoshaping protocol, and the switch protocol. The matching protocol measures the mouse's capacity to estimate incomes (food pellets per unit time) at two different locations, to remember which location yields which income, to match the ratio of its average visit durations to the ratio of the incomes, and the rapidity with which it detects and adjusts to changes in the income ratio. The 2-hopper autoshaping protocol measures rates of instrumental and classical conditioning (associative learning rates). The switch protocol measures interval timing accuracy and precision, the ability to estimate probabilities (relative frequencies) and to adjust to changes in probabilities (risk assessment). It also gives a measure of impulsivity.

Matching. The feeders in each of the two lateral hoppers (the feeding hoppers) are independently armed at an exponentially varying latency following each pellet delivery. A pellet is delivered whenever the feeder is armed and the mouse triggers pellet release by poking into the hopper, thereby interrupting the infrared (IR) beam. The intervals that elapse before the next arming are drawn from exponential distributions, whose expectations are specified in the parameter file (Table 1). In this protocol, mice rapidly begin to cycle between the two feeding hoppers, poking for a while into one, then moving to the other, then back to the first one. The ratio of the average durations of their stays at the two hoppers approximately matches the ratio of the pellets per minute that they are obtaining from the two hoppers (N.B. Pellets per minute in the environment, that is the "incomes" from hoppers, not pellets per minute of poking into the hopper, that is the "returns" from their investments in

the two hoppers). Thus, if they get on average 3 times as many pellets from Hopper 1 as from Hopper 2, then the average duration of a visit to Hopper 1 is approximately 3 times as long as the average duration of a visit to Hopper 2.We run this protocol first because of the rapidity with which the matching behavior appears[8]. It is usually measurable within the first 24 hours of testing, except, of course, in the occasional mouse that is tootimid to visit the hoppers within the first 24 hours.

Matching behavior implies that the mechanisms for measuring duration and number and computing relative rates are intact, as are the spatial localizing mechanisms that enable a mouse to associate appropriately the two different rates with the two different hopper locations. Thus, it is an excellent quick test of basic cognitive function. The precision with which a mouse matches the ratio of its expected visit durations to the ratio of the experienced incomes is an indication of the precision with which the mouse can represent quantities.

The precision with which a mouse matches is graphically visualized by the extent to which two concurrently plotted cumulative records have the same slope (**Figure 6**). A cumulative record is the cumulating sum of a sequence of measurements. In this case, one plot (the red ones in **Figure 6**) is the cumulative record of the pellet-by-pellet income-imbalance measure; the other (the black ones in **Figure 6**) is the cumulative record of the pellet-by-pellet visit-imbalance measure. For a single inter-feeding interval, the income imbalance is either -1 or +1 depending on whether the pellet obtained at the conclusion of the interval came from Hopper 1 (imbalance = -1) or 2 (imbalance = +1). The slope of the cumulative record of these imbalances is the average income imbalance (the imbalance per feeding). If in 10 intervals the mouse gets 5 pellets from Hopper 1 and 5 from Hopper 2, then the average income imbalance over that those 10 inter-feed intervals is 0. If 10 pellets came from Hopper 2, then it is 1, if all 10 came from Hopper 1, then it is -1. Thus, the average income imbalance can range from -1 to 1, and it is 0 when there is on average no imbalance. The visit imbalance during an inter-feeding interval is the time spent at Hopper 2 (*T*2) minus the time spent at Hopper 1 (*T*1), divided by the total of the two times. For a single inter-feeding interval, this measure can take on any value between -1 and 1, unlike the income imbalance, which must be either -1 or 1. However, the slope of the cumulative record of this measure reflects its average value, just as is the case for the income measure. When the two slopes of the two cumulative records (the red and the black records in **Figure 6)** are equal, the average visit imbalance equals the average income imbalance, which means that the mouse is matching the ratio of its average visit durations to the ratio of its incomes from the two hoppers.

For three of the mice in **Figure 6** (5027, 5015 and 5024), the slope of the cumulative visit imbalance plot (black) closely tracks the slope of the cumulative income imbalance (red), indicating nearly perfect matching of the ratio of the average visit durations to the ratio of the incomes. Note how quickly the ratio of the visit durations adjusts to the change in the ratio of the incomes, at the points of downward inflection. On the other hand, Mouse 5034 (bottom left plot) failed to match during the second feeding session; the slope of the cumulative visit-duration imbalance plot (black) is 0 whereas the slope of the income imbalance (red) is substantially positive. However, during the third feeding phase (after the black vertical), the slopes are parallel, which means that this mouse began abruptly at the beginning of this phase to match its visit ratio to the income ratio. Clearly, therefore, it was not incapable of doing so, but did not do so for some reason during part of the testing. This illustrates the importance of visualizing the entire course of performance rather than relying entirely on a few summary statistics. Mouse 5028 matched precisely during the first income ratio, but when it was reversed, it did not fully reverse. Mouse 5025 "overmatched" during the 3rd feeding phase (after green vertical), that is, its average visit imbalance was greater than its average income imbalance and it did not fully adjust to the reversal of the income imbalance. Notice, however, that all 6 mice, both the 3 wild types and the 3 heterozygotes matched with precision during the very first 4-hour feeding phase. Notice also that these records cover only 36 - 48 hours (1.5-2 day/night cycles with 2 feeding phases per cycle), during which one observes not only initial matching but an abrupt response to the reversal of the income ratio. As previously explained, this demonstrates the intact functioning in the heterozygotes of many basic mechanisms of cognition, to wit, duration estimation, number estimation, rate estimation, spatial localization (of income rates to hoppers) and the ability to remember simple abstract quantities. The quantitative results, which are typical, are obtained from automated testing in a live-in environment, with no handling of the mice beyond their initial placement in the environment.

As is evident in **Figure 6**, mice match the ratio of their visit durations to the ratio of the incomes as soon as they begin to cycle between the hoppers. The rapidity with which the mouse begins to cycle between the hoppers is a measure of what might be called its boldness or tendency to explore. This is visualized by plotting the cumulative record of cycles, that is, the number of cycles completed as a function of session time. **Figure 7** shows these plots for the same mice as in **Figure 6**. Five of the six showed some exploration before the onset of the first feeding phase and began abruptly to cycle between the hoppers as soon as it began. One mouse, however (bottom left), did not show any exploratory behavior at all (not a single cycle) until midway through the 2nd feeding phase, at which point, it began abruptly to cycle between the hoppers.

Instrumental and Classical Conditioning. Traditional learning theory distinguishes between two kinds of associative learning: classical conditioning (also called Pavlovian conditioning) and instrumental conditioning (also called operant conditioning). The physiologically meaningful parameter in both cases is the learning rate. This is thought to measure the rapidity which synaptic strengths are adjusted in response to the pairing of two sensory events in classical conditioning (e.g., hopper illumination and food delivery) or in response to the pairing of an S-R event and a reinforcing event in instrumental conditioning. The learning rate is measured by the reciprocal of trials-to-acquisition, that is, the number of pairings prior to the appearance of the conditioned behavior. Our second protocol measures these rates. It also teaches the subjects the feeding latencies to be expected at the two pellet-delivering hoppers, which knowledge is tested in the third protocol.

A pretrial in this protocol begins with the illumination of the middle hopper, the one where pellets are never delivered, because there is no pellet feeder for it (see diagram of Test Box, upper right in **Figure 1**). When the mouse pokes into the illuminated middle hopper, the light there goes out, and a randomly chosen one of the two flanking hoppers lights up. If it is the left hopper that lights, then a pellet is delivered there 4 s after illumination, regardless of what the mouse does. If it is the right hopper that lights, then a pellet is delivered there 12s after illumination, regardless again of what the mouse does. Thus, the poking into the illuminated middle hopper is instrumental in initiating a trial; when this hopper is illuminated, nothing further happens until the mouse pokes into it; the mouse must poke into this hopper in order to get a pellet. Once the mouse has made this instrumental response, a pellet will be delivered into whichever hopper then lights up, at the delivery-latency peculiar to that hopper, regardless of whether the mouse goes to the hopper or not during the interval that elapses before pellet delivery.

The software automatically makes three plots of performance in this task: 1) the cumulative record of trial initiation speed (**Figure 8**), the cumulative distribution of trial-initiation latencies (**Figure 9**) and the cumulative records of the CS-ITI poke-rate differences (**Figure 10**). And, it derives from these records three sets of summary statistics: 1) trials to acquisition of the instrumental response (instrumental learning rate);

2) the median trial-initiation latency; and 3) trials to acquisition of anticipatory poking into both the short-latency and the long-latency feeding hoppers (autoshaped, that is, classical learning rates).

Timed decisions. The third protocol is the "switch" protocol first used with pigeons by Fetterman and Killeen[24] and adapted for the mouse by Balci et al[22]. As in the previous protocol, trials are initiated when the mouse pokes into the illuminated middle hopper. However, in this protocol, this trial-initiating poke illuminates both flanking hoppers, one of which is silently and randomly chosen by the computer as the "hot" hopper for that trial. The feeding latencies associated with the two hoppers are the same as in the previous protocol (4s & 12s). However, only the hot hopper delivers, and it delivers only in response to a poke made there at or after "its" feeding latency. If the short-latency hopper is hot, then if the mouse is at that hopper at the end of the short latency, the first poke made there at or after that latency delivers a pellet into that hopper. If the long latency hopper is the hot hopper, then the first poke made there at or after the long latency delivers a pellet into that hopper. Mice rapidly learn to go first to the short-latency hopper on most trials and then, on "long" trials, switch to the long-latency hopper when poking in the short-latency hopper fails to deliver a pellet at the conclusion of the short latency of 4 s. The mouse obtains its pellets on the great majority of trials. However, if it stays too long at the short-latency hopper on a long trial, with the result that the first poke at or after the long latency is in the wrong hopper, the trial ends without a pellet delivery. Similarly, if it switches prematurely on a short-latency trial, so that the first poke at or after the short latency is in the wrong (long-latency) hopper, the trial ends without pellet delivery. The data of interest are the latencies of the switches on the long trials, defined as the termination-latency of the last poke into the short hopper prior to the first poke into the long hopper.

In this paradigm, we vary the long-latency. We start with a long latency of 12s, which is 3 times the 4s short latency, because mice learn to switch ~~must~~ faster when the task is made that easy. We then shorten the long latency to 8s or even to 7s or6s to test how tightly the mice can time their switching (see **Figure 11**). The measure of timing precision is the coefficient of variation in the Gaussian component of the distribution of switch latencies. The Gaussian coefficient of variation is the ratio between the standard deviation and the mean (σ/μ in **Figure 11**).

We also vary the probability of a long trial, thereby varying the relative risk of switching prematurely versus switching too late. When the probability of a long trial is high, the risk incurred by a premature departure from the short-hopper is reduced, because on a long trial, it does not matter how soon the mouse switches. Moreover, the risk incurred by a too-late departure is correspondingly elevated. Conversely, when the probability of a long trial is low, the relative risk of a premature departure versus a too-late departure shifts in the opposite direction. A rational decision maker should shift the distribution of its switches so as to approximately equate the risks, which is what mice in fact do[22,23] (**Figure 12**). Measuring the shifts induced by changing the probability of a long trial assesses the mouse's ability to estimate probability (of a long versus a short trial), to estimate the variability in the distribution of their switches, and to compute the relative risks implied by these two sources of uncertainty.

Impulsivity. The distribution of switch latencies is a mixture distribution. In the vast majority of mice under most conditions, the distribution is dominated by switch latencies from a Gaussian distribution, as in **Figure 11**. However, there is often a small component of "impulsive switches," which occur much too soon, appear not to be carefully timed, and are clearly not optimal (unlike the Gaussian component, which is approximately optimally positioned between the temporal goal posts, as may be seen in **Figure 11**). The frequency of these impulsive switches varies strikingly between mice and with conditions. Making the long trial highly probable increases their frequency in most mice, as does making the task more difficult by moving the long latency closer to the short latency (see **Figure 11**). The distribution of switch latencies can almost always be fit with impressive accuracy by a "weibgauss" function, which is the mixture of a Weibull distribution and a Gaussian distribution. The measure of impulsivity is the fraction of the switches imputed to the Weibull distribution.

Circadian Memory. Animals record in memory the time of day (phase of their circadian clock) at which motivationally important events happen. When the same event routinely recurs at approximately the same circadian phase, their behavior begins to anticipate its recurrence. A commonly studied indication of this ability to remember and thereby anticipate a daily recurrence is feeding-anticipatory behavior[25]. We measure this while running the above described (or almost any other) protocols, simply by restricting the intervals during which the mice can obtain pellets to two 4-hour intervals surrounding dusk (house light out) and dawn (house light on). During other intervals, the mice are still in the test environment, but no protocol is operative. Thus, they obtain pellets only during the scheduled feeding intervals. In somewhere between 6 and 15 days after the beginning of testing, anticipatory poking activity appears in the hour or so before the dusk feeding interval (**Figure 13**). The circa-dusk feeding interval begins without any signal one hour prior to extinguishing of the house light (dusk).

On those days when 3/4 of the behavioral events occur during the final 1/3 of the 9-hour phase of no-food availability preceding the onset of the dusk feeding phase, we count that as an instance of food-anticipatory activity. Anticipatory activity thus scored does not occur during the first several days of testing ~~(blue asterisks in **Figure 13**)~~. Our measure of "trials" (days)-to-acquisition of circadian food-anticipatory activity is the day after which the frequency of anticipatory activity shows the first statistically significant increase.

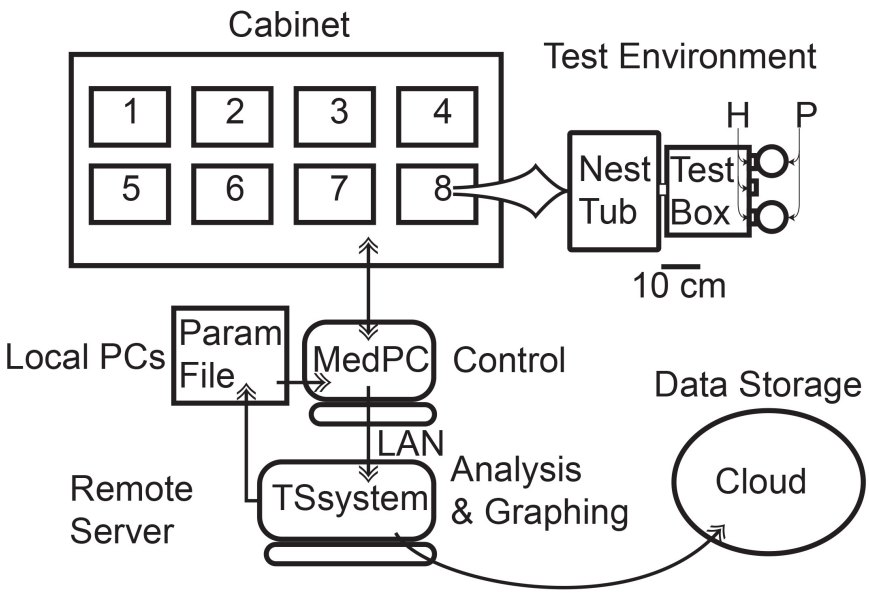


Figure 1. Schematic of screening system. Each cabinet contains 8 test environments ~~Each~~ consisting of a polypropylene mouse tub connected by an acrylic tube to a Mouse Test box with 3 hoppers (H) monitored by IR beams, two of which hoppers have attached pellet feeders (P). A single PC (In-lab computers) running a protocol-control program that controls the environments in two such cabinets (16 environments) and logs time-stamped records of stimulus and response events. A single protocol-control program implements all the different protocols. Which protocol is operative and with which parameters is determined by the contents of a text file (Param File), which is read regularly by the protocol-control program. Changing the contents of this text file changes the operative protocol. The protocol-control program writes the event record every 10 minutes to another text file. The TSsystem data-analysis software, running on a remote server, harvests the data every 15 minutes. At regular intervals (typically every 8 or 12 hours), it performs an extensive statistical and graphical analysis of the data obtained so far and puts the results into files on a cloud-synch storage site, where they may be accessed by anyone in the lab at anytime from anywhere. The analysis software automatically moves individual mice on to the next protocol in a specified series of protocols by writing new parameters to the parameter file (Param File) read by the protocol-control program. It does so when the data from that mouse meet user-specified decision criteria. LAN = local area network.Click here to view larger image.

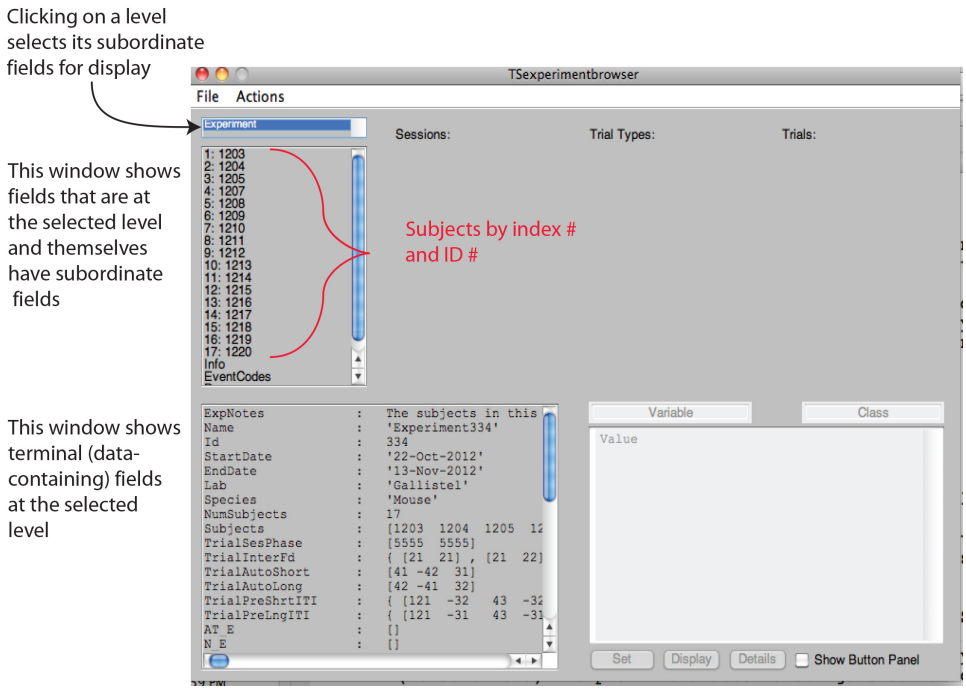
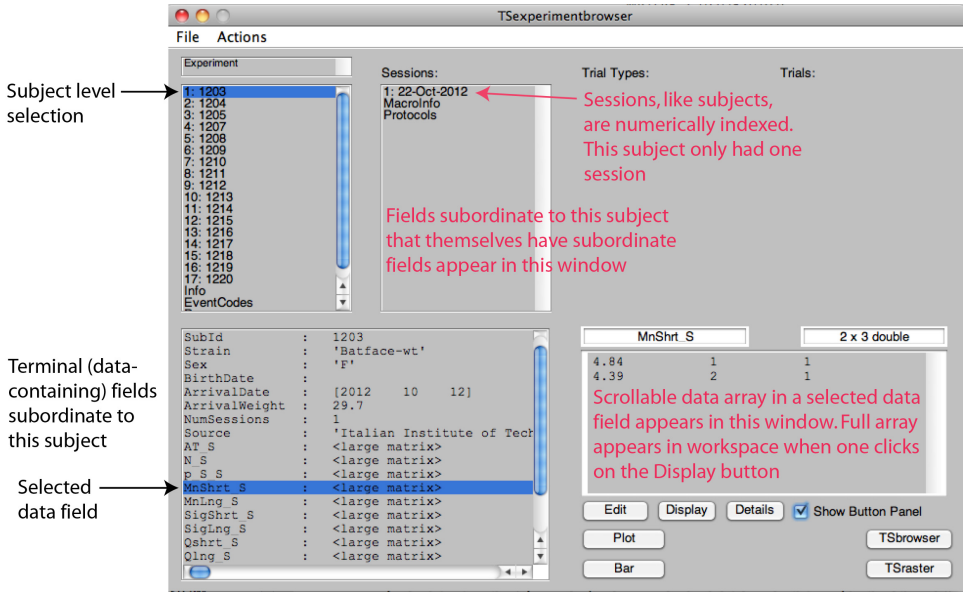
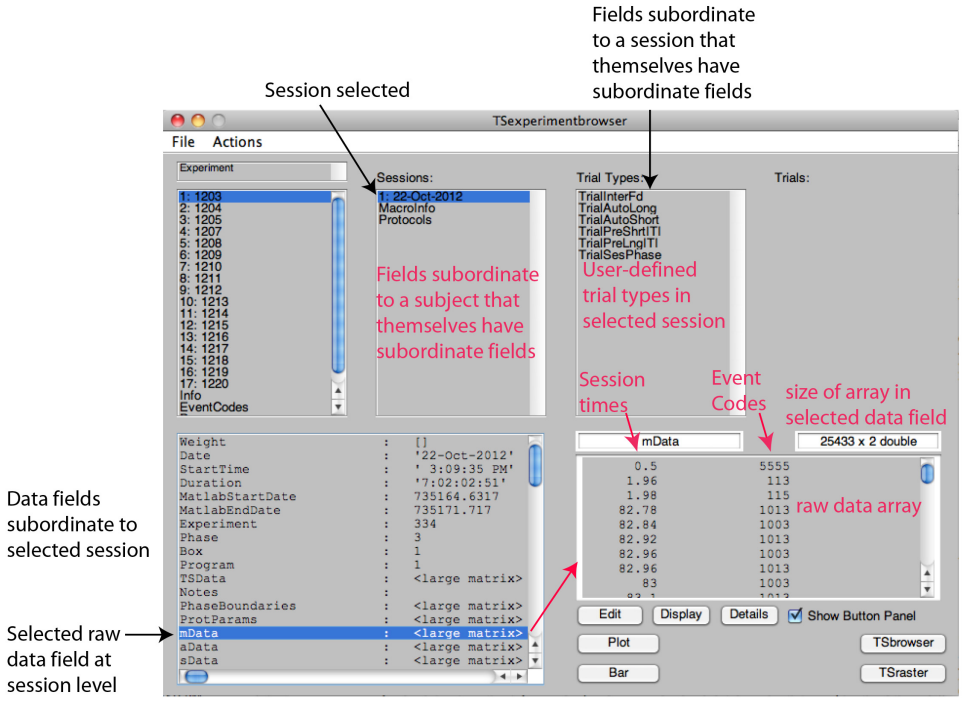


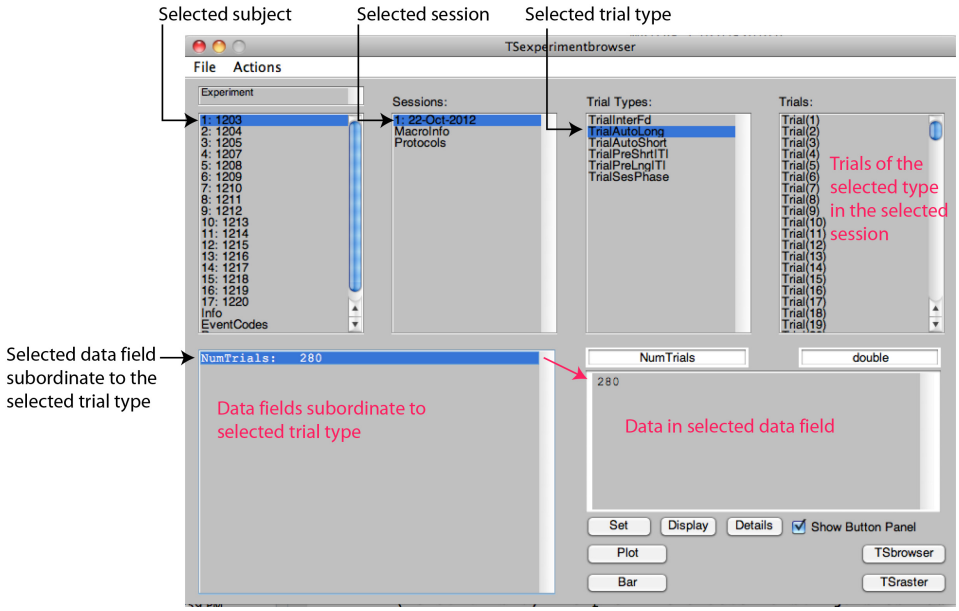
Figure 2.A. Screen shot of the TSbroswer GUI. The browser enables browsing through the complex hierarchical data file that contains the raw data and all of the results derived from it. The functions in the TSsystem operate on data in fields of this structure and store the results in other fields within the same structure. Subordinate to the fields at all levels but the lowest are terminal data fields containing summary statistics, notes and numerically indexed lower levels: The principal numerically indexed lower levels are: The Subject Level, the Session Level, and the Trial Level (see subsequent panels in this figure). Click here to view larger image.



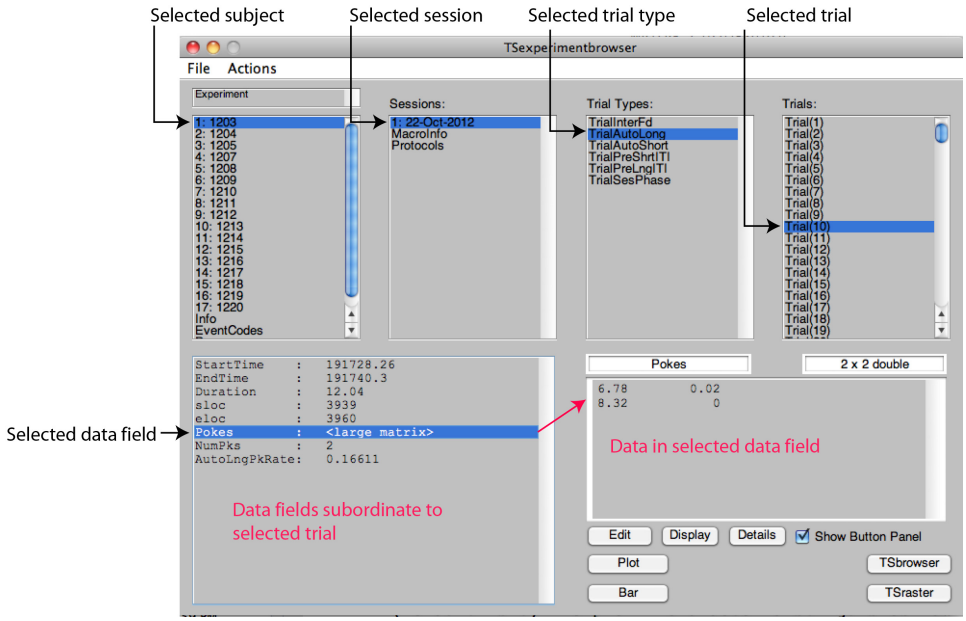
B. Screen shot of the browser after selection of a subject. In the lower left window are the data fields directly subordinate to that subject. They contain summary statistics for that subject. In the Sessions window are the numerically indexed sessions for this subject and other fields that, like the Session field, have subordinate fields of their own. The MacroInfo field has subfields giving information about the protocol-control macro, including the path to and file name of the code file read by the process-control software at the start of a session-.The Protocols field. The Protocols field has subfields giving the parameters of the different protocols run in the selected session, the decision fields containing the data on which the decision to terminate a protocol is based, the code that makes that decision, and the decision criteria. Selecting a data field in the lower left window, displays the data in that field in the lower right window. Click here to view larger image.



C. ~~When a session is selected, the data fields immediately subordinate to it~~ are displayed in the lower left window. Among those fields are those containing the raw data and subdivisions thereof and those containing summary statistics extracted from those raw data. When a data field in this window is selected, its contents are displayed in the window at lower right. In this screen shot, the field containing the raw data from the matching protocol has been selected. In the Trial Type window to the right of the session window are displayed the trial types run during the selected session. A "trial" is any user-specified sequence of not-necessarily contiguous events that define multiple stretches of data that are to be analyzed for their content. The user may define arbitrarily many different types of trials. Click here to view larger image.



D. ~~When a trial type is selected, the~~ numerically indexed trials of that type are displayed in the Trials window. As always, the data fields immediately subordinate to the selected field are displayed in the lower left window, and, if one of those fields is selected, its contents are displayed in the lower right window. Click here to view larger image.



E. ~~When a Trial is selected, the~~ data fields subordinate to it are displayed in the lower left window and, if one of those fields is selected, its contents are displayed in the lower right window. Among the fields automatically included in every trial are fields specifying the session time at which the trial began, the session time at which it ended, its duration, and the row numbers in the raw data file that compass the trials. The other fields (here 'Pokes,' 'NumPks', and 'AutoLngPkRate' are user defined. The contents of all the fields are computed from the raw data stored in a field at the session level. Click here to view larger image.

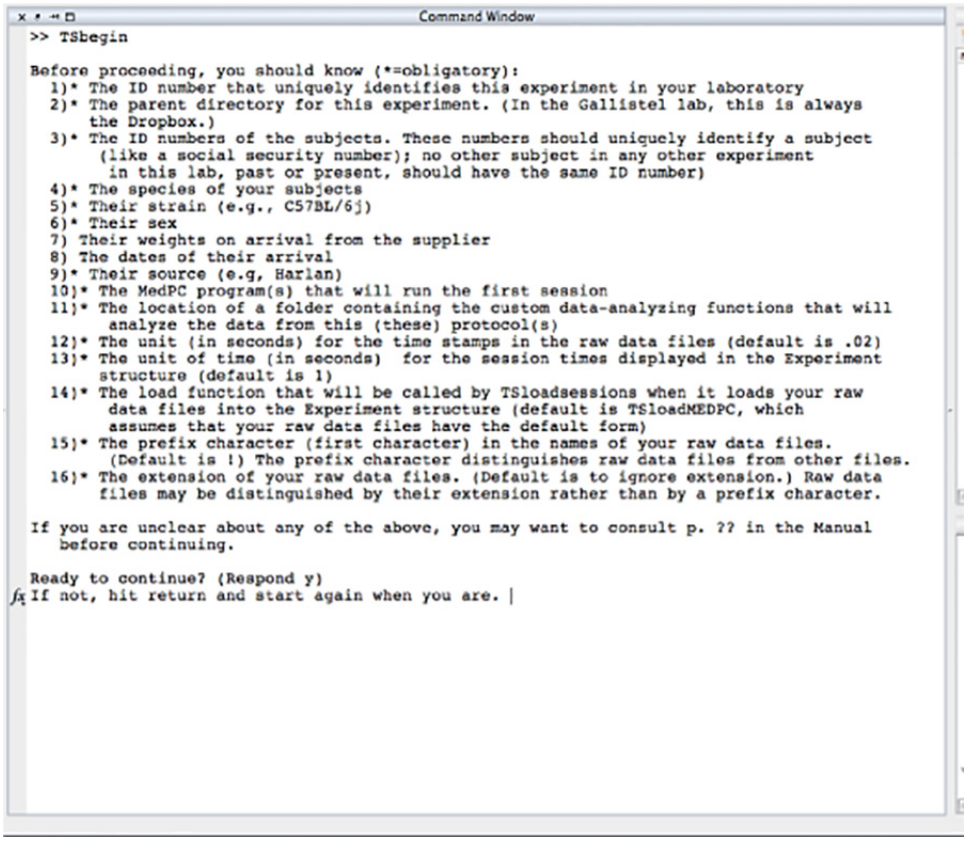


Figure 3. Screen shot of the initial prompt from TSbegin. TSbegin uses prompts to lead the user through the process of creating the hierarchical data structure into which the raw data and all the summary statistics and other results derived from those data will go. Click here to view larger image.

```
TSaddprotocol adds a new column of protocol parameters, specifying a new
protocol in the Protocol structure in the Protocols field at the Subject
level of an Experiment structure. To be called once for each new protocol
that is to be added to a given set of subjects. Also adds corresponding
DecisionField, DecisionCode and DecisionCriteria

Syntax: [Params ProtocolStructure] = TSaddprotocol(Exp,Subjects,Subject-Value pairs),

where Exp is the ID# of the experiment and Subjects is a row(1) vector of
Subject indices specifying the subjects to whose sequence of protocols
the new protocol is to be added.
ProtocolStructure is  the 1-column structure showing the
column that was added to the structure in the Protocols field of each
subject specified in Subjects. It has 4 fields: .Parameters;
.DecisionFields; .DecisionCode; & .DecisionCriteria. Params is the column
vector of new parameter values; it also appears in the Parameters field
of ProtocolStructure. The structure in the Protocols field of Experiment
has 2 more fields: .Current and .Info. This function does not alter the
contents (if any) of those fields

The subject-
value pairs are:
  'Protocol', ['M' 'A' or 'S'];
  'Dawn' [time of lights on in HH:MM format];
  'Dusk'[time of lights off];
  'ErlStrt' [time when early feeding phase starts in HH:MM format];
  'ErlStp' [time when it stops];
  'LtStrt'[time when late feeding phase starts];
  'LtStp'[time when it stops];
  'ITI' [expected intertrial interval in autoshape & switch protocols in
     seconds];
  'p(S)' [probability of a short trial in chances out of 10000];
  'p(O)' [probability of an operant trial in chances out of 10000;
  'FdLat1' [feeding latency for Hopper 1 in seconds];
  'FdLat2' [feeding latency for Hopper 2];
  'VI1 [variable interval for Hopper 1 in seconds]';
  'VI2' [variable interval for Hopper 2;
  'Flag' [false for matching protocol, true for autoshape and switch]
```

Figure 4. Screen shot of help for TSaddprotocol. This interactive GUI leads the user through the process of creating a new version of one experimental protocol. To do this, the user must specify new values for the parameters of that protocol (for example, new values for the concurrent variable interval schedules in the matching protocol). The user must also specify (or choose previously specified) code for making the decision to terminate the protocol and go on to the next protocol. The user must also specify (or choose previously specified) decision fields and decision criteria. Click here to view larger image.

```
>> TSstartsession

Before proceeding, you should know:
  1) The Id number of the experiment [#]
  2) The location of the file containing the Experiment structure
      (This should be the Experiment[#].mat file in the Experiment[#] folder)
  3) The ID numbers of the currently active subjects
       (NB, not their index numbers in the Experiment structure!)
  4) The information needed to create the MedPC macro: For each subject
       i) The letter that identifies on the LAN the computer that controls
            that subject s test environment
      ii) The location on that computer s hard drive of the .MPC file
            for its protocol (aka the condition, phase, or group for that subject)
     iii) The box number (ID # for the test environment for that subject)
      iv) The number that identifies that protocol (Protocol 1, 2, etc)
  5) Whether each protocol restricts feeding periods to certain intervals
       during each 24 hr day and, if so, the event codes that indicate
       the starts and ends of those intervals (e.g, StartEarly  StopEarly)
  6) The custom data-analysis Matlab function for each protocol in this
       session (e.g., MatchingAnalysis.m)
  7) The arguments that must be passed to that function
     (e.g., TdyPlts,Rows,Format). If unsure, you can use defaults
  8) The names of the plots that you want emailed (if unsure,
     default is CmPksFds
  9) The email addresses of the recipients


Are you ready? [y]
If not, hit return & start again when you are.
```

Figure 5. Initial prompt from TSstartsession. TSstartsession is an interactive GUI that leads the user through the process of starting an experimental session. One session typically runs from a day or two to several weeks, with several different protocols (tasks) performed in the course of that one session. Click here to view larger image.

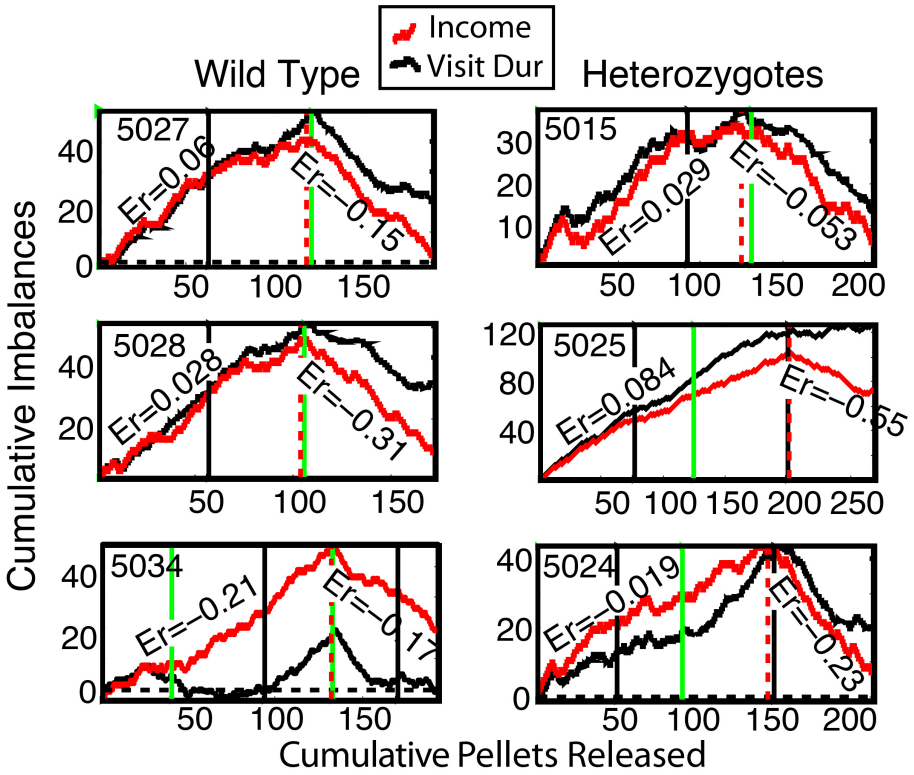


Figure 6. Cumulative imbalance plots for income (pellets obtained, in red) and visit duration (in black) for 3 wild type mice and 3 L1 heterozygotes. The relative richness of the two concurrent VI schedules was reversed at the point where the slopes turn downward (marked by vertical dashed red line). Mouse ID numbers are on each plot. The green verticals mark the onset of the dusk feeding phase; the ~~blue~~ verticals mark the onset of the dawn feeding phase. The Er (for error) values are the differences in slope between the income plot and the visit-duration plot. These imbalance errors are twice the error in the Herrnstein fraction. The Herrnstein income fraction is the fraction of total income obtained from a hopper; the visit-duration fraction is the fraction of total time spent at a hopper (relative to combined time spent at both hoppers). A positive sign indicates "overmatching", that is, the time fraction is more extreme than the income fraction; a negative sign indicates "undermatching", that is, the time fraction is less extreme than the income fraction. For details on these experiments see Gallistel et al[28]. Click here to view larger image.

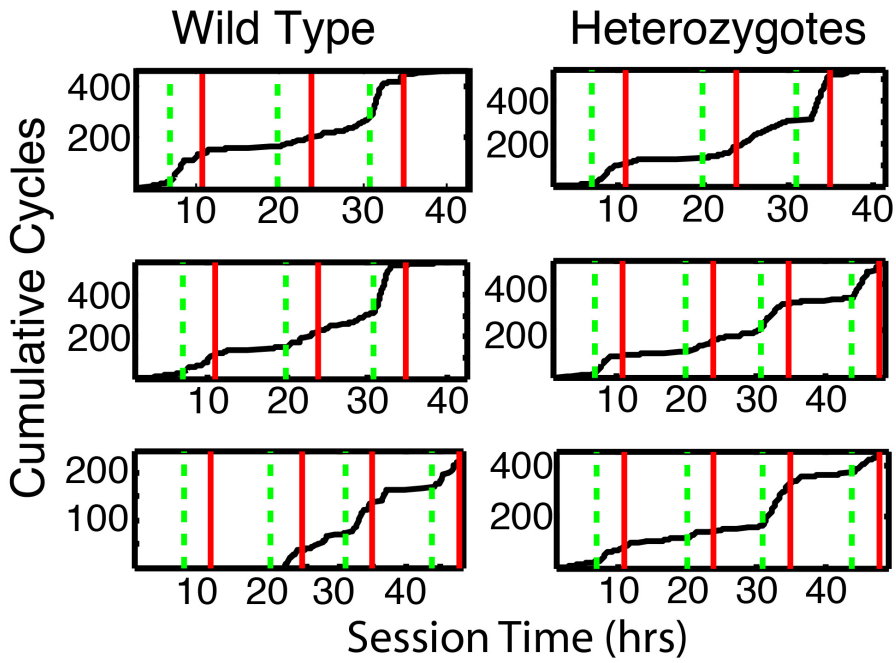


Figure 7. Cumulative cycles (Hopper 1 visit, followed by Hopper 2 visit, followed by a return to Hopper 1) versus testing time for the same 6 mice as in Figure 2. Dashed green verticals mark the onsets of feeding phases; solid red verticals, the offsets. Click here to view larger image.

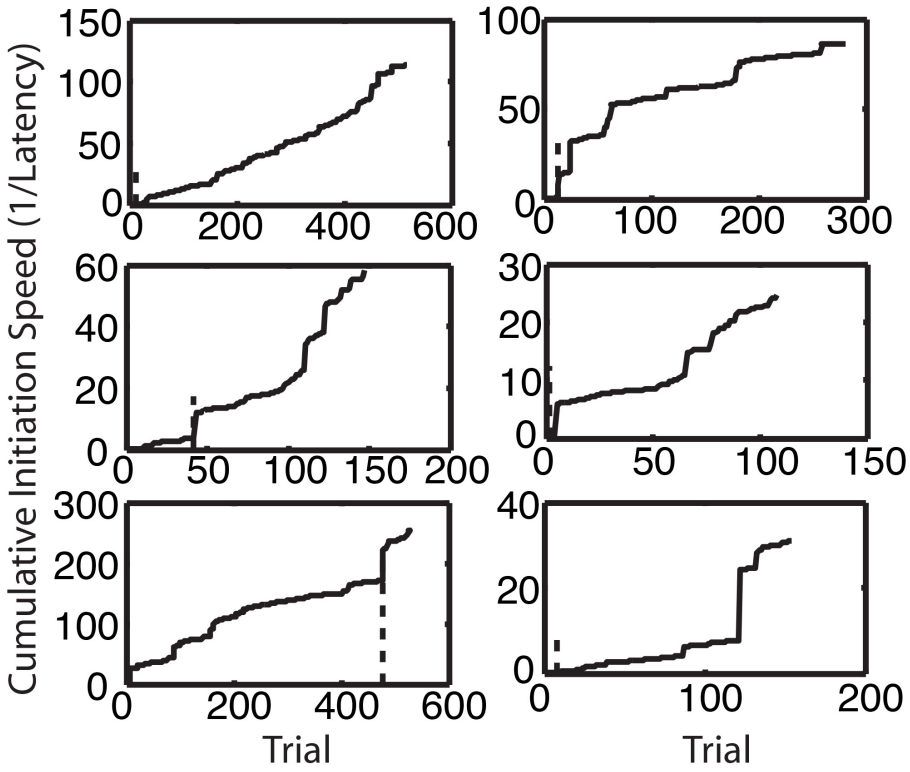


Figure 8. Cumulative records of trial-initiation speeds during the instrumental/classical conditioning protocol, same 6 mice as in Figures 2 & 3. The short heavy vertical dashed lines mark the acquisition trial in each record, defined as the first trial at which there was a significant increment in average speed to a value greater than the average speed in the first segment of the record. Click here to view larger image.

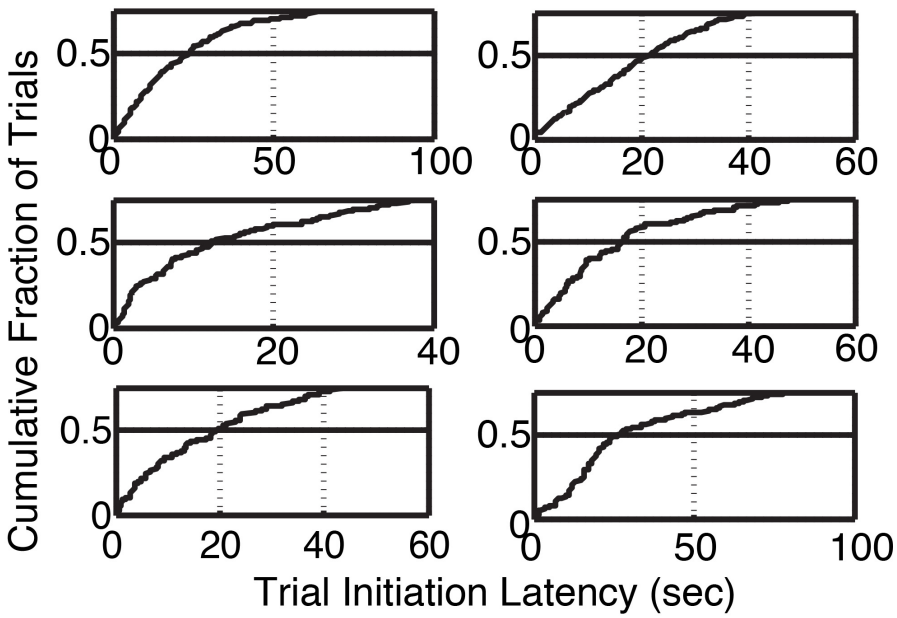


Figure 9. Cumulative distributions of trial initiation latencies, same mice as in previous figures. The distributions have been truncated at .7, because there are extremely long initiation latencies from trials when the mouse was not present in the test box when the trial-initiation hopper lit up, or the mouse was satiated. The point above the x-axis at which the horizontal line at .5 intersects the cumulative distribution is the median latency to initiate a trial. Click here to view larger image.

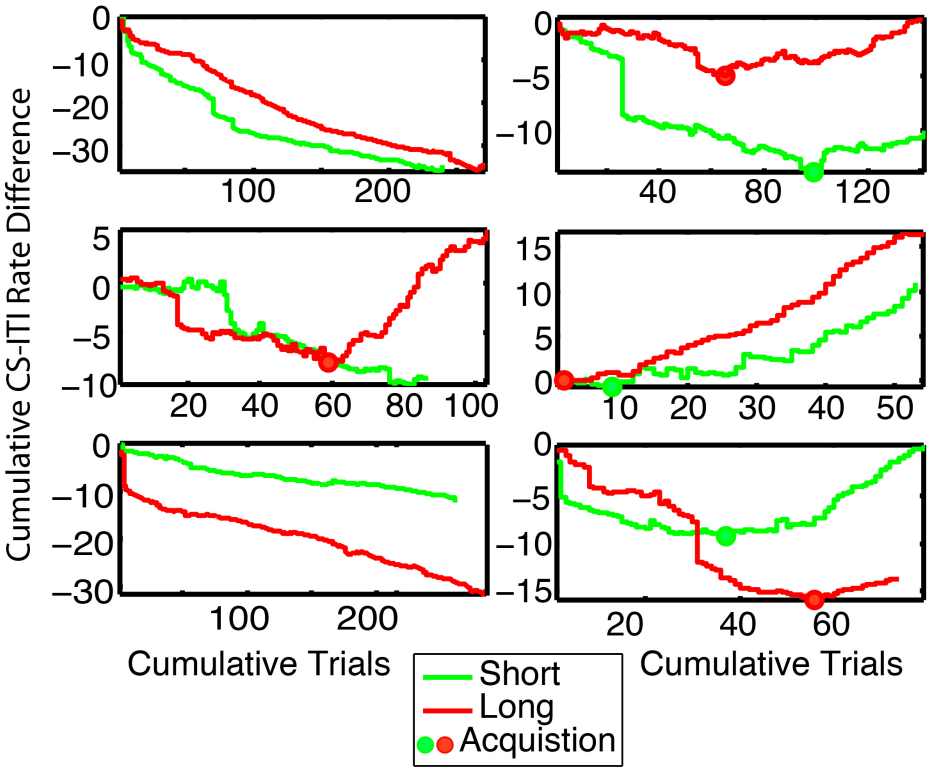


Figure 10. Cumulative records of the differences between the poking rate during the illumination of a feeding hopper and the poking rate during the preceding intertrial interval. Same mice as in previous figures. Initially, the slope of this record is 0 or negative, because the mouse pokes less during hopper illumination than during the intertrial interval. The slope turns positive when the mouse begins to poke into the illuminated hopper in anticipation of the impending delivery of a pellet. This anticipatory poking is classically conditioned because the pellet is delivered at the end of the hopper-specific delivery latency whether the mouse pokes into the hopper or not. The solid dots mark the point at which the slope becomes positive, which is the trial at which the conditioned response first reliably appears (trial-to-acquisition). This is an extremely ~~noisy (~~variable~~)~~ measure, as is evident in this sample. Click here to view larger image.

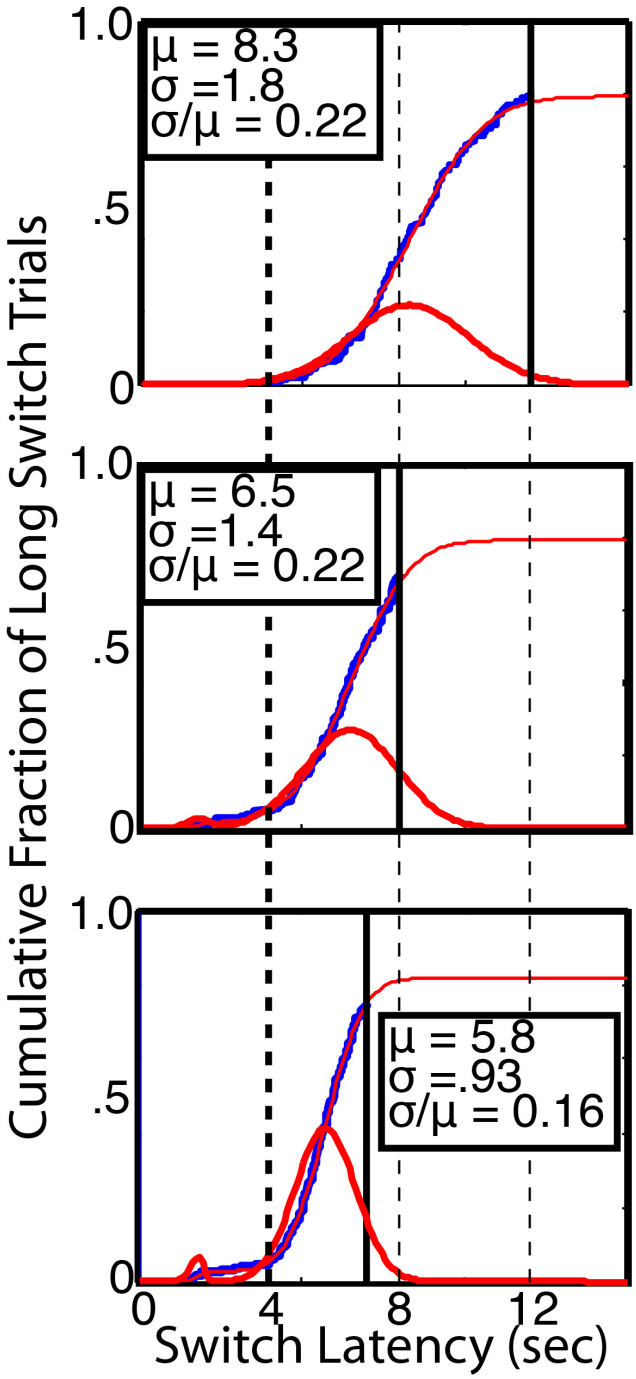


Figure 11. The slightly ragged heavy blue lines are empirical cumulative distribution functions, showing the cumulative distribution of switch latencies for Mouse #5024 at three different settings of the feed latency at the long hopper (heavy black verticals at 12, 8 and 7 s from top plot to bottom plot, respectively). The short feed latency was always 4 s (heavy dashed vertical). The thin red curves that superimpose almost exactly on the empirical functions are the 6-parameter weibullgauss functions fit to these data. The heavy red distributions are the corresponding probability density functions for the weibull-gauss mixture distributions. (They are the derivatives of the cumulative distribution functions.) The parameters of the Gaussian component of these mixture distributions are shown on the plots. Shortening the long feed latency caused the mouse to shorten the mean of its switch latencies (μ) and increase their precision (σ/μ). It also caused the appearance of some impulsive switches (the bump in the left tail of the probability density function in the bottom plot). The measure of this impulsivity is the fraction of the mixture attributed to the weibull component by the best-fitting version of the weibullgauss mixture distribution. Click here to view larger image.

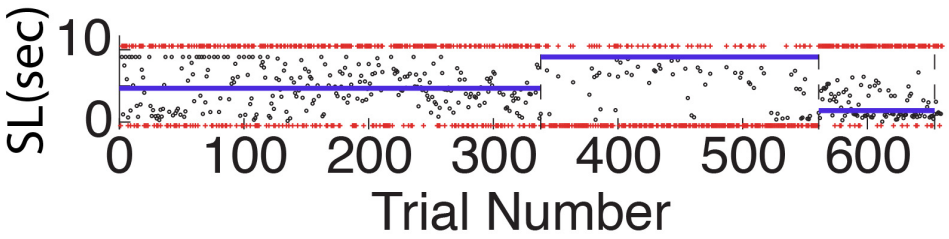


Figure 12. Switch latencies (small open circles) shift abruptly soon after changes in the relative frequency of short and long trials. The short trials are marked with tiny red pluses at the bottom of the plot, while the long trials are marked with tiny red pluses at the top of the plot. The change in the density of the resulting red streaks at bottom and top indicate the change in the relative frequency (probability) of the short and long trials. Note that when the red at bottom becomes more dense (when the short trials become more probable), the small circles (switch latencies) shift upward and when the red at the top becomes denser (an increase in long-trial probability), the circles shift downward. The blue lines are the medians of the distributions. Excerpted from **Figure 2** in[23]. Click here to view larger image.

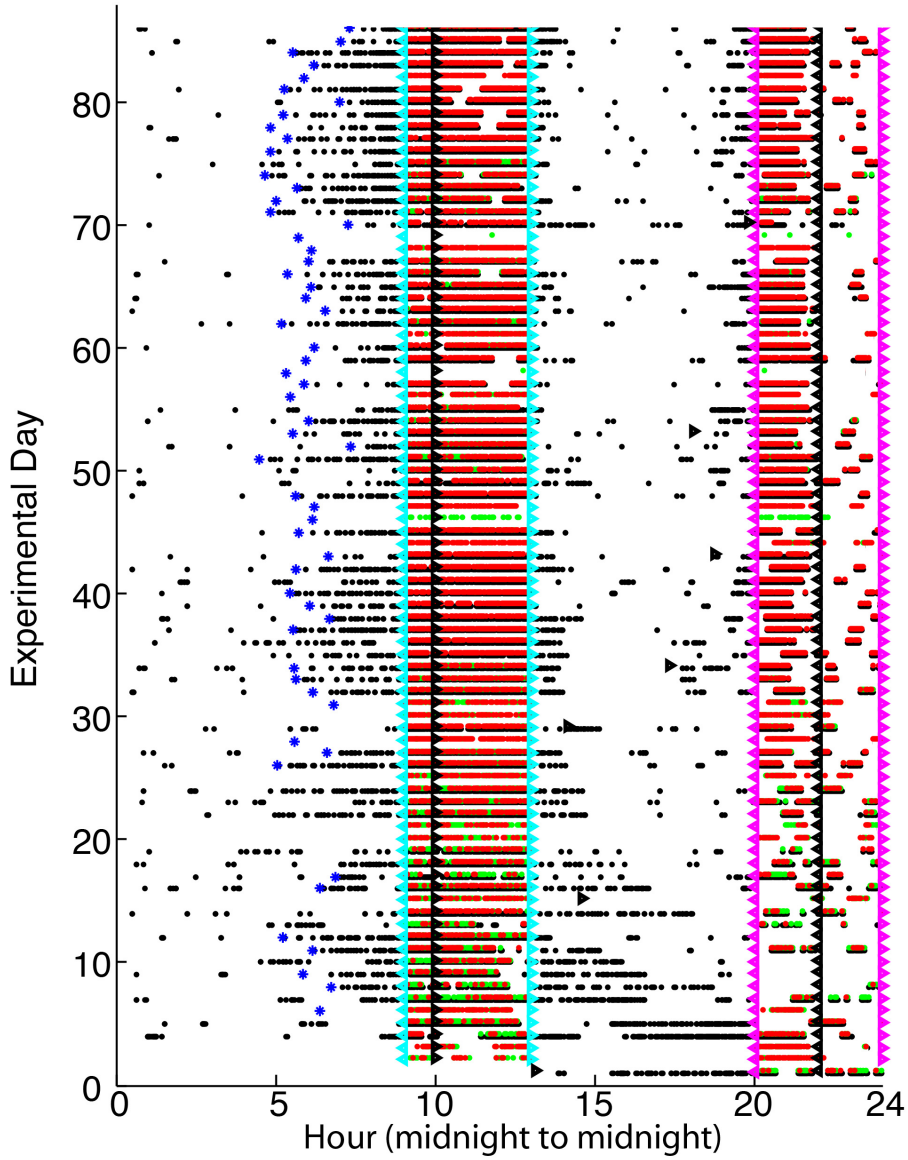


Figure 13. 24-hour raster plot of behavioral and environmental events over a 90-day test period. Black dots record pokes; red and green dots record pellet deliveries at the two feeding hoppers. Black right-pointing triangles record house-light offs; black left pointing triangles record house-light ons; cyan left-pointing triangles record the (unsignaled) onsets of the dusk feeding interval, which occur one hour prior to dusk itself; cyan right-pointing triangles record the offsets of the dusk feeding interval (3 hours after house-light off); left-pointing magenta triangles record the (unsignaled) onsets of the dawn feeding intervals (3 hours before house-light on); right-pointing magenta triangles record the offsets of the dawn feedings. The 24-hour time axis is conventional, midnight to our midnight, not the reverse-cycle time of the test environment in which the mouse lived. The onsets of feeding-anticipatory activity prior to the dusk feeding are marked with blue asterisks. Click here to view larger image.

Discussion

Our method yields a wide range of physiological meaningful, quantitative results on the functioning of several different mechanisms of cognition, learning and memory, for many mice at once, in a minimum amount of time, with a minimum of human labor, and with no handling of the experimental subjects during days, weeks, or months of testing. These attributes suit it for genetic and pharmacological screening programs. It uses minimally modified off-the-shelf hardware (test boxes and nest tubs). It produces more data from more mice on more aspects of basic cognition in less time than other currently used methods for testing the cognitive capacities of genetically and/or pharmacologically manipulated mice.

It depends on two kinds of open-source software, written in commercially available, well-supported languages: 1) A language supplied by the maker of the test equipment (see Materials) for the implementation of experimental protocols and the recording of data; 2) an open-source toolbox, TSsystem, for on-line, quasi-real-time data analysis, graphing and decision making. The toolbox is written in one of the most widely used and thoroughly supported proprietary scientific data analysis and graphing languages (see Materials).

The TSsystem toolbox ensures a clear and secure data trail, because the process-control code that governed the experimental protocols, together with the raw data, and all results derived from the raw data, are kept in a single data structure. This hierarchical data structure has field names, which function like the headers in a spreadsheet in that they indicate the contents of the field. Unlike the headers in a spreadsheet, the field names in this data structure are indexible. Data summarizing and graphing commands can access data anywhere in the structure by specifying a path through the headings, for example:

"Experiment.Subject(3).Session(2).TrialTypeCS.Trial(5).NumPokes."

The commands in the custom toolbox, TSsystem, operate on data in these fields and store the results in another field or fields within the same structure. Usually the field or fields in which the results of an operation are stored are created by the TSsystem command. The data-analysis code may itself be stored in this same structure. The different levels of the structure contain Notes fields, where explanatory notes of arbitrary length may be put. The user can make additional notes fields, explaining and interpreting the contents of data fields. Thus, archiving the structure or putting it online in a publically accessible database stores every relevant aspect of the experiment and the data analysis in a single file. While the experiment is running, the file containing the data structure is in a folder that is synchronized in "the cloud." This arrangement insures off-site storage and state-of-the art backup. It gives access to the latest version of the data analysis and graphics whenever and wherever an investigator has access to the internet.

Critical steps. The critical steps in the construction of a test protocol are:

1. The writing and debugging of the process-control code in the language provided by the commercial supplier of the computer-controlled mouse-testing apparatus (see Equipment and Supplies). A user that sticks with the protocols whose representative results were reviewed above can use the code provided in the Supplementary Material. This single code file implements any or all of the above-described test protocols. The operative protocol for a given subject at a given point in the testing depends on which column of protocol parameter values are in force at that time. The user determines which of these protocols are run in which sequence using TSaddprotocol before the testing begins. Included in the Supplementary Material are protocol-specific decision codes and decision fields. The user need only specify the decision criteria. These decision criteria govern how much data is collected from a given protocol for a given subject. Because the results from each subject are graphically portrayed several times in each 24-hour period, the user can adjust the criteria upward or downward on a mouse-by-mouse basis in the light of the results so far obtained from each mouse.
2. The writing and debugging of the quasi-real-time data analysis and graphing code using the library of commands in the TSsystem toolbox. A user that sticks with the protocols whose representative results were reviewed above can use the code provided in the Supplementary Materials. There is one master analysis function for each test protocol (matching, instrumental and classical conditioning, and timed hopper switching). Another master function (DailyAnalysis.m), which does protocol--independent trouble-shooting analyses and calls the protocol-specific analysis functions when it has completed its trouble-shooting.
3. The daily monitoring of graphs produced by the data analysis code to be sure that the protocol is operating properly and that all mice are obtaining enough pellets to keep them in good health.

Troubleshooting. The DailyAnalysis.m function is called 2 or more times in every 24 hours by a timer function whenever one or more experiments are running. Whenever it is called, it loads the file that contains the hierarchical data structure and reads into that structure the raw data files for all active subjects. These raw data files are written to every 10 minutes by the process-control software. The DailyAnalysis function runs a series of basic checks. It produces graphs that enable the user to verify that the pellet dispensers are operating properly, that each mouse is active, and that each mouse has obtained a sufficient number of pellets within the last 24 hours. The absence of data from a mouse for some user-specified number of hours, a number of pellets obtained that falls below a user-specified critical value, or numerical results suggesting that a pellet dispenser is not operating properly, trigger email alerts.

Modifications. Because the process-control code and the TSsystem code are open source and extensively commented, users can devise their own protocols. The protocols for which code is provided in the Supplementary Materials are only appetitive protocols. They use food reward to elicit behavior the quantitative aspects of which depend on the functioning of the target cognitive mechanisms. However, the commercial mouse test chambers may be ordered in a configuration that enables the fear conditioning by means of foot shock. Thus, the system can implement both contextual, cued, and time-of-day fear conditioning for users able to write the requisite computer code for both process control and quasi real-time data analysis. The heavily commented code files provided in the Supplementary Material should facilitate the creation of the requisite new code.

Limitations. The basic mechanisms of cognition are those that enable animals to locate themselves in space and time and to estimate event probabilities and attendant risks. The fully automated testing system here described provides extensive testing of the mechanisms that enable mice to locate themselves in time, and to estimate probabilities and attendant risks. However, it provides very limited information about the mechanisms that enable them to locate themselves in space. Both the matching protocol and the timed switching protocol test whether the mouse can distinguish otherwise identical hoppers on the basis of their location within the test box. However, none of these protocols tests, for example, the mechanism that enables animals to orient themselves using the geometry of a familiar space [26] nor the odometer that measures how far a mouse has run and that plays a central role in dead reckoning [18].

Significance of technique. The behavioral testing technique here described differs from most other techniques in common use in behavioral pharmacology and behavioral genetics in five important ways:

First, it makes physiologically meaningful measurements. These are measurements, such as the accuracy and precision of the representation of an interval duration, that can be repeated at the electrophysiological and biochemical levels of analysis with comparable results. Physiologically meaningful behavioral measurements play a crucial role in establishing secure linkage hypotheses of the form this molecular or cellular or circuit level mechanism is the neurobiological realization of that behaviorally manifest mechanism. Examples are the behavioral measurement of spectral sensitivity functions and the periods and phases of oscillators, such as the circadian clock. Diverse examples of behavioral measurements that are not physiologically meaningful are the amount of time a mouse can hang on a rotorod, the mean latency to find a

submerged platform in a water maze, the percentage of a swim trajectory in the quadrant where the platform was previously found, and the fraction of time that a mouse freezes in a test chamber where it has previously been shocked.

Second, it is an automated, high volume, high throughput procedure. The mice are not handled during the course of testing. Many mice can be tested simultaneously in a limited amount of laboratory space, with minimal time investment, during the running of several different protocols giving quantitative information about basic cognitive mechanisms. Most behavioral testing requires daily handling of the mice to place them in and remove them from the test apparatus. Some of the most popular procedures (the water maze and contextual fear conditioning, for example) require handling the mouse immediately before and immediately after each measure is taken and time-consuming scoring of video records.

Third, the results are analyzed and graphed in quasi real-time, and the progress from test to test occurs automatically for each mouse individually. This enables tuning the procedure (by, for example, changing decision criteria) mouse by mouse in the light of up-to-the-hour data from each mouse. The data from commonly used tests are often analyzed long after the test has been run. This requires running mice collectively for the same amount of time in each procedure. This is wasteful because different mice master different procedures after widely differing amounts of time. If the collective test time is made too short, several mice do not master the task; if it is made long enough so that almost all mice master the task, some mice are run much longer than necessary.

Fourth, the design of the TSsystem toolbox ensures an intact, readily traceable trail from the published summary statistics and graphs back to the raw data, with no uncertainty about which protocol and which parameters were operative at any point in the testing of any mouse. Many current tests obtain very little data for each mouse, in which case, the data trail is short and easily followed. When tests obtain a high volume of data (thousands of time-stamped events) from each mouse, then the data trail may become long, badly fragmented and hard to follow. Over time, slightly different process-control code files are generated; many different data analyzing code files are written; the different code files produce many different results files by operating on different earlier results files. It becomes difficult to keep track of the complex relationships between the many different electronic files and difficult to make sure they are all saved together. In the TSsystem, the raw data and all that derives from them are stored in the same hierarchical data structure. Thus, published results cannot become separated from the underlying raw data. The process-control code for each session is automatically read by the TSsystem software and compared character by character to all previous process-control code in the same experiment. If there is any difference, the code for the new session is stored in the structure along with the data and a number identifying that process-control version is automatically entered into a field for that session in that mouse. Thus, every version of the process-control code is automatically stored in the same structure as the data obtained from the execution of that code. With this system, one is never unsure as to which data came from which version of a multi-version protocol. The data-analysis code files, which are usually few in number, may also be stored in this structure upon completion of the analysis. This puts everything in a single data structure, which indicates both by its branching structure (what is subordinate to what) and its field names the relations between the many different kinds of information.

Fifth, the hierarchical data structure at the core of the TSsystem, in which all relevant information is stored, together with the very high level data-analysis commands used to extract summaries and construct graphs, makes feasible the sharing of complex behavioral phenotyping data between laboratories. A single electronic file gives other researchers intelligible, exploitable access to all levels of the data processing, and to the computer code that digested, summarized and graphed the data. It makes the later re-analysis of rich older data sets possible. It also makes possible the creation of large useable public behavioral and cognitive phenotyping databases *for future applications.* The fully-automated system may prove useful in laboratory courses in animal behavior. It finesses the increasingly troublesome issue of instructing large numbers of students in the proper handling of mice and attendant concerns about possible threats to their health from doing so. It enables large numbers of students to design and execute experiments with real mice without ever handling them. For instructional use, one might want to put small inexpensive infrared video cameras in the test boxes, so that students could observe the mouse performing in the protocol they have devised. The placing of the raw data in a single hierarchical data structure stored in the cloud makes it possible for many different students to analyze the same data as the data come in.

Disclosures

There is nothing to disclose.

Acknowledgements

The creation of this system was supported by 5RO1MH77027.

References

1. Gallistel, C. R., Shizgal, P. & Yeomans, J. S. A portrait of the substrate for self-stimulation. *Psychological Review*, **88**, 228-273 (1981).
2. Takahashi, J. S. Molecular neurobiology and genetics of circadian rhythms in mammals. *Annual Review of Neuroscience*, **18**, 531-553 (1995).
3. Mackay, T. F. C., Stone, E. A. & Ayroles, J. F. The genetics of quantitative traits: challenges and prospects. *Nature Reviews Genetics*, **10**, 565-577 (2009).
4. Weber, J. N., Peterson, B. K. & Hoekstra, H. E. Discrete genetic modules are responsible for complex burrow evolution in *Peromyscus* mice. *Nature*, 402-405 (2013).
5. Balsam, P. D., Drew, M. R. & Gallistel, C. R. Time and Associative Learning. *Comparative Cognition & Behavior Reviews*, **5**, 1-22 (2010).
6. Gallistel, C. R. & Gibbon, J. Time, rate, and conditioning. *Psychological Review*, **107**, 289-344 (2000).
7. Ward, R. D. *et al.* Conditional Stimulus Informativeness Governs Conditioned Stimulus—Unconditioned Stimulus Associability. *Journal of Experimental Psychology: Animal Behavior Processes*, doi:10.1037/a0027621 (2012).
8. Gallistel, C. R. *et al.* Is matching innate? *Journal of the Experimental Analysis of Behavior*, **87**, 161-199 (2007).
9. Herrnstein, R. J. Derivatives of matching. *Psychological Review*, **86**, 486-495 (1979).
10. Mark, T. A. & Gallistel, C. R. Kinetics of matching. *Journal of Experimental Psychology: Animal Behavior Processes*, **20**, 79-95 (1994).

11. Brandeis, R., Brandys, Y. & Yehuda, S. The use of the Morris water maze in the study of memory and learning. *International Journal of Neuroscience*, **48**, 29-69 (1989).
12. Foucaud, J., Burns, J. G. & Mery, F. Use of spatial information and search strategies in a water maze analog in *Drosophila melanogaster*. *PLoS ONE*, **5** (2010).
13. Logue, S. F., Paylor, R. & Wehner, J. M. Hippocampal lesions cause learning deficits in inbred mice in the Morris water maze and conditioned-fear task. *Behavioral Neuroscience*, **111**, 104-113 (1997).
14. Upchurch, M. & Wehner, J. M. Differences between inbred strains of mice in Morris water maze performance. *Behavior Genetics.*, **18**, 55-68 (1988).
15. Zilles, K., Wu, J., Crusio, W. E. & Schwegler, H. Water maze and radial maze learning and the density of binding sites of glutamate, GABA, and serotonin receptors in the hippocampus of inbred mouse strains. *Hippocampus.*, **10**, 213-225 (2000).
16. Chen, G., King, J. A., Burgess, N. & O'Keefe, J. How vision and movement combine in the hippocampal place code. *Proceedings of the National Academy of Sciences*, **110**, 378-383 (2013).
17. Gallistel, C. R. *The organization of learning*. (Bradford Books/MIT Press, 1990).
18. Wittlinger, M., Wehner, R. & Wolf, H. The desert ant odometer: a stride integrator that accounts for stride length and walking speed. *Journal of Experimental Biology*, **210**, 198-207 (2007).
19. Challet, E., Mendoza, J., Dardente, H. & Pevet, P. Neurogenetics of food anticipation. *European Journal of Neuroscience*, **30**, 1676-1687 (2009).
20. Arcediano, F. & Miller, R. R. Some constraints for models of timing: A temporal coding hypothesis perspective. *Learning and Motivation*, **33**, 105-123 (2002).
21. Denniston, J. C., Blaisdell, A. P. & Miller, R. R. Temporal Coding in Conditioned Inhibition: Analysis of Associative Structure of Inhibition. *Journal of Experimental Psychology: Animal Behavior Processes*, **30**, 190-202 (2004).
22. Balci, F., Freestone, D. & Gallistel, C. R. Risk assessment in man and mouse. *Proceedings of the National Academy of Science (USA)*, **106**, 2459-2463 (2009).
23. Kheifets, A. & Gallistel, C. R. Mice take *calculated* risks. *Proceedings of the National Academy of Sciences (USA)*, **109** 8776-8779, doi:10.1073/pnas.1205131109 (2012).
24. Fetterman, J. G. & Killeen, P. R. Categorical scaling of time: Implications for clock-counter models. *Journal of Experimental Psychology: Animal Behavior Processes*, **21**, 43-63 (1995).
25. Luby, M. *et al.* Food anticipatory activity behavior of mice across a wide range of circadian and non-circadian intervals. *PLoS One*, **7**, doi:10.1371/journal.pone.0037992. Epub 2012 May 25 (2012).
26. Lee, S. A., Vallortigara, G., Ruga, V. & Sovrano, V. A. Independent effects of geometry and landmark in a spontaneous reorientation task: a study of two species of fish *Animal Cognition*, **15**, 861-870, doi:10.1007/s10071-012-0512-z (2012).
27. Rodriguiz, R. & Wetsel, W. C. in *Animal Models of Cognitive Impairment*.eds ED Levin & JJ Buccafusco) Ch. 12, (CRC Press, 2006).
28. Gallistel, C. R. *et al.* Fully Automated Cognitive Assessment of Mice Strains Homozygous for Cell--Adhesion Genes Reveals Strain--Specific Alterations in Timing Precision. *Philosophical Transactions of the Royal Society. B.* (in press 2013).

