Implicit value updating explains transitive inference performance: The betasort model

Greg Jensen[1], **Fabian Muñoz**[1], **Yelda Alkan**[1], **Vincent P. Ferrera**[1], **and Herbert S. Terrace**[1]

[1]**Columbia University**

ABSTRACT

Transitive inference (the ability to infer that "$B > D$" given that "$B > C$" and "$C > D$") is a widespread characteristic of serial learning, observed in dozens of species. Despite these robust behavioral effects, reinforcement learning models reliant on reward prediction error or associative strength routinely fail to perform these inferences. We propose an algorithm called *betasort*, inspired by cognitive processes, which performs transitive inference at low computational cost. This is accomplished by (1) representing stimulus positions along a unit span using beta distributions, (2) treating positive and negative feedback asymmetrically, and (3) updating the position of every stimulus during every trial, whether that stimulus was visible or not. Performance was compared for rhesus macaques, humans, the betasort algorithm, and *Q*-learning (an established RPE model). Of these, only *Q*-learning failed to respond above chance during critical test trials. Implications for cognitive/associative rivalries, as well as for the model-based/model-free dichotomy, are discussed.

Keywords: transitive inference, cognition, reinforcement learning, rhesus macaques

INTRODUCTION

Tests of *transitive inference* (TI) are among the oldest tools for assessing abstract thinking. First introduced by Piaget (1921) to demonstrate the emergence of logic in child development, TI has since been studied in many species. The cognitive faculties that permit transitive inference are very general: To date, TI has been observed in every vertebrate species tested, including primates (MacLean et al., 2008), rodents (Takahashi et al., 2008), birds (Bond et al., 2010), and even fish (Grosenick et al., 2007). The widespread occurrence of this phenomenon suggests that TI procedures tap into deep and enduring learning systems.

In the classical TI task, an ordered list is assembled from otherwise arbitrary stimuli. For example, seven photographic stimuli are given the ordered labels "A" through "G." During training, subjects are only shown randomly selected adjacent pairs (AB, BC, CD, DE, EF, and FG), and are required to select one stimulus in every trial. The only feedback provided is a reward (if the earlier list item was selected) or no reward (if the later item was selected). No other cues indicate that stimuli have an ordering, and no more than two items are ever simultaneously presented. Following training, preference is assessed for non-adjacent pairs (e.g. BD). If subjects select earlier items in novel pairs at above-chance levels, they are said to have performed a "transitive inference" because doing so exploits the transitive relationship that "$B > C$" and "$C > D$" implies "$B > D$." Figure 1 depicts sample stimuli, trial structure, and stimulus pairings for a 7-item TI task.

In the above example, only the stimuli A and G are differentially rewarded, and can therefore be identified on the basis of reward prediction error. Accordingly, these stimuli are correctly identified more often, (the *terminal item effect*). Correct choices among stimuli B, C, D, E, and F are more difficult to explain, however, because their expected value during training is 0.5. The pair BD is a *critical pair*

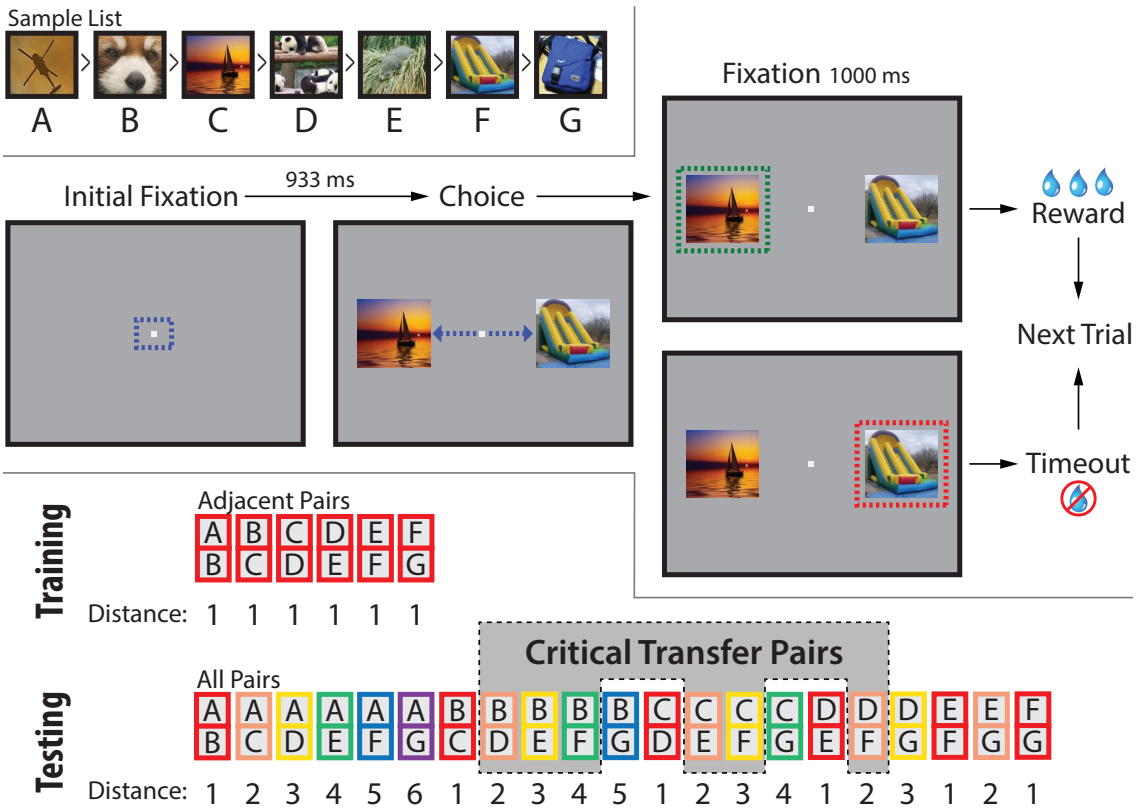

Figure 1. The transitive inference procedure, as implemented for rhesus macaques responding using eye tracking. **(Top)** Each session used a novel seven-item list, like the one depicted here. However, subjects were never presented with the entire list. **(Middle)** Each trial began with a fixation point. Following fixation, two stimuli appeared, and subjects received feedback upon a saccade to either stimulus. If the stimulus appearing earlier in the list was selected, a reward was delivered; if the other stimulus was selected, the animal was subjected to a timeout. Either outcome constituted the completion of a trial. In the event of an incomplete trial (e.g. subjects fixating but failing to saccade to a stimulus) was deemed incomplete and did not count toward the set of trials completed. All dashed lines and arrows represent eye movements and fixation areas, and did not appear on the screen. **(Bottom)** Subjects were initially trained only on the six adjacent pairs. Following adjacent pair training, subjects were then tested on all twenty-one pairs. These varied in their ordinal distance (with the pair AG being the largest). Additionally, six pairs were considered the *critical transfer pairs* (shaded in gray) because they were neither adjacent nor did they include the terminal items. Consequently, these are the pairs that provide the strongest test of transitive inference and symbolic distance effects.

during testing because that pair is novel and contains no terminal items. Learning models that rely on only the expected values of stimuli fail to make the inference and respond at chance levels (Lazareva and Wasserman, 2012).

Despite decades of research, controversy remains over what exactly is learned during TI tasks. The *cognitive learning* school of thought holds that inferring the order of BD based only on BC and CD implies internal representation of the ordered list (Jensen et al., 2013). The strongest evidence in favor of this view are the *symbolic distance effects* (D'Amato and Colombo, 1990). Pairs of items with greater symbolic distance (i.e. the number of steps needed to traverse the list from one stimulus to the other) reliably yield more accurate discriminations. Thus, contrary to predictions based on expected value, subjects should not only favor stimulus B when presented with BE, but should also do so more than during BD trials. The reliability of distance effects provides evidence that subjects (human or otherwise) make inferences about list items using serial representations, rather than the frequency with which stimuli are paired with rewards (Gazes et al., 2012).

On the other hand, the *associative learning* school maintains that TI can be explained by stimulus-response-outcome associations alone (Wynne, 1995). Although associative models of TI struggle to accommodate the full range of empirical findings, their mathematical formalism at least permits specific predictions (Vasconcelos, 2008). Cognitive models, by contrast, have historically been too vague to permit the simulation of behavior (Lazareva and Wasserman, 2012).

Here, we attempt to resolve this difficulty by comparing the ability of computational models to explain aspects of TI performance observed in humans and monkeys. These include the transfer of knowledge from adjacent to non-adjacent pairs, and symbolic distance effects. One model, drawn from the machine learning literature, can only learn from *experienced* outcomes. Another is a new model that can *infer* the relative list positions of the stimuli. We argue that these models are representative of associative learning on the one hand, and cognitive learning on the other.

When situating TI in the current literature, it is important to define terms. The overarching topic of *reinforcement learning* (RL) pertains to how subjects learn by trial and error, whether through associative or cognitive processes. This approach is informed by the machine learning literature (popularized by Sutton & Barto, 1998), which specifies a different distinction: "model-free" RL vs. "model-based" RL (Maia, 2009). "Model-based" algorithms use contingency tables to relate states and actions, and are more precisely called Markov decision processes (MDPs). "Model-free" algorithms in turn have the following characteristics:

1. Each action is represented by an expected value of reward.
2. Values are updated as a function of discrepancy between the expectation and outcome, called *reward-prediction error* (RPE).
3. Predictions are made about available actions, so only values associated with available actions are updated.

Such algorithms can solve certain problems without contingency tables, instead using minimalistic models based on RPE. We will henceforth refer to model-based and model-free algorithms as MDP models and RPE models, respectively.

Q-learning (Watkins, 1989) is a widely-studied RPE model that estimates each action's expected value, making it an *updating policy* (modifying memory as a function of feedback). However, Q-learning is only unbiased if, during training, it performs every action in every context uniformly (Watkins and Dayan, 1989). Favoring successful actions (and avoiding harmful ones) can catastrophically bias its estimates. Accurate convergence is only guaranteed when Q-learning is paired with a counterproductive "try everything" *choice policy* (selecting the next behavior), a recognized shortcoming of RPE models generally (Thrun, 1992). Despite this, Q-learning is often paired with the *softmax function* (Luce, 1959), a choice policy that selects actions stochastically as a function of expected values. We subsequently refer to this pairing as Q/softmax, which takes two parameters (α and β, described in the methods).

RPE models like Q/softmax systematically fail transitivity tests, because non-terminal stimuli are rewarded equally during training. Although models have been proposed that overcome this difficulty in specific cases (e.g. von Fersen et al., 1991; Siemann and Delius, 1996), they remain vulnerable to bias. For example, Lazareva and Wasserman (2012) trained pigeons and RPE models on adjacent pairs from the 5-item list ABCDE. They then presented massed trials of only the pair DE. Pigeons correctly identified "$B > D$," but all RPE models either concluded that the value of stimulus D was larger than that of every other stimulus, or responded at chance levels. RPE models fail to learn correctly from massed

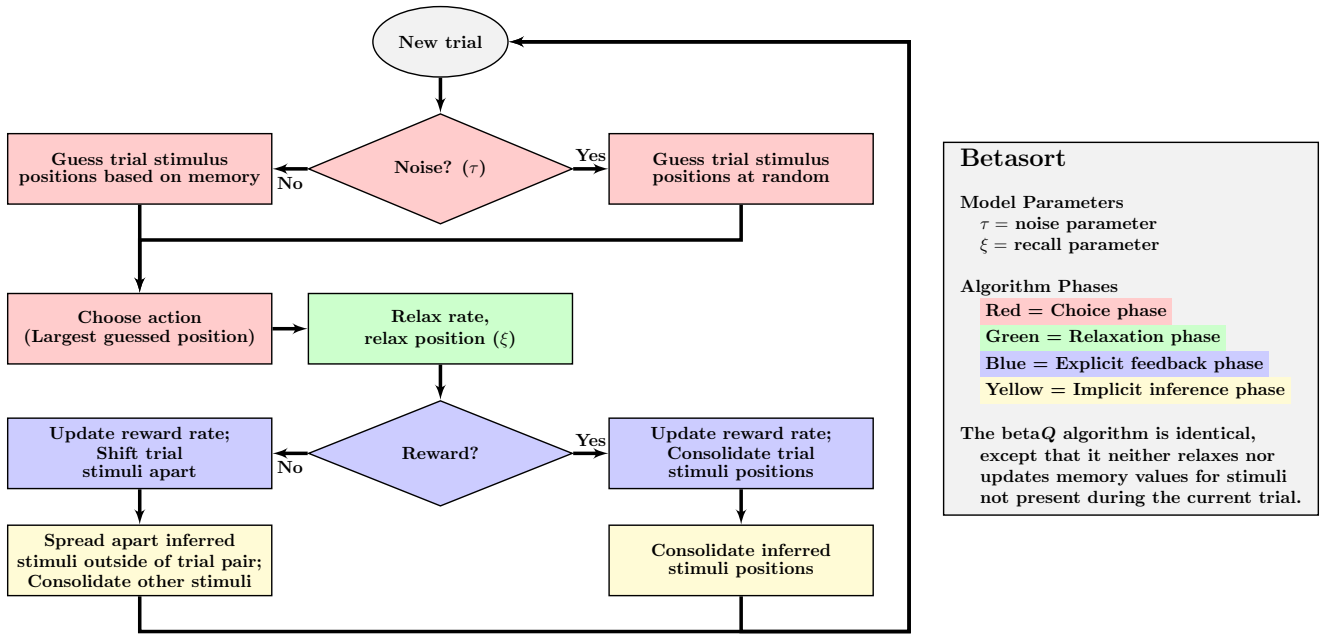

Figure 2. A schematic overview of the betasort algorithm over the course of one trial. Rectangles refer to operations and diamonds to logical branches. Four phase are depicted: the choice policy (red), the relaxation of the contents of memory (green), the processing of explicit feedback (blue), and implicit inference (yellow). A more detailed version of this diagram is provided in Figure 8.

trials because they are only adaptive in cases where expected value is a suitable proxy for behavior.

The Betasort Algorithm

We propose a new model, called betasort, which successfully performs the transitive inference task for lists of arbitrary length at low computational cost. Betasort qualifies as neither an RPE model nor an MDP model. Figure 2 outlines betasort's operation (including both its choice and updating policies); a detailed description is provided in the methods.

Betasort is based on three principles. The first is the use of beta distributions. Although commonly used as sampling distributions for probabilities, we instead use them here to represent stimulus position on a unit scale. Betasort selects behaviors using these distributions, and then updates stimulus positions and their uncertainty. The second principle is that feedback should be used to update the position of a stimulus, rather than its expected value. Consequently, when the outcome of an action is satisfactory, one should *consolidate* the current position, rather than shift it. The third principle is that every stimulus representation should be updated during *every trial*, regardless of which stimuli are presented. Collectively, these principles provide a plausible mechanism for transitive inference.

The position of a stimulus i is represented by two parameters: An "upper" parameter U_i and a "lower" parameter L_i, both positive. If $U_i > L_i$, then the stimulus position is closer to the top of the scale; if $L_i > U_i$, then it is closer to the bottom. As U_i and L_i both get larger, the uncertainty associated with the stimulus position decreases. The density function over a sample space from 0.0 to 1.0 is given by:

$$Beta(x; U_i, L_i) = \frac{\Gamma(U_i + L_i)}{\Gamma(U_i)\,\Gamma(L_i)} x^{U_i - 1} (1 - x)^{L_i - 1} \tag{1}$$

Here, $\Gamma()$ represents the gamma function. When $U_i = L_i = 1.0$, the probability density is uniform; it grows increasingly normal as these parameters increase. In order to consolidate a stimulus position, rather

than shift it, these parameters are increased by a proportion of their current value (i.e. $U_i \leftarrow U_i + \frac{U_i}{U_i+L_i}$ and $L_i \leftarrow L_i + \frac{L_i}{U_i+L_i}$). This "distributes" a single reward across both parameters, leaving the position intact while reducing its uncertainty. Thus, incrementing values of U_i and L_i is effectively Bayesian updating. Betasort also tracks the reward (R_i) and non-rewards (N_i) associated with trials that include each stimulus. Importantly, if a trial is rewarded, the value of R_i is increased for *both* stimuli. This is because R_i and N_i control the algorithm's explore/exploit tradeoff, increasing the variability of behavior when the current representation is not functioning effectively.

Betasort's choice policy (red in Figure 2) draws random values from each position distribution, and selects the largest from among the available actions. The policy uses one free parameter: *noise* ($0.0 < \tau < 1.0$), which is the probability that betasort ignores its memory and selects an action at random. When $\tau = 1.0$, the algorithm is entirely stochastic; when $\tau = 0.0$, all choices are governed by memory.

Betasort's updating policy begins with the *relaxation phase*, (green in Figure 2), which makes use of another free parameter: *recall* ($0.0 < \xi < 1.0$), which scales the contents of memory downward during every trial prior to processing the feedback for that trial. For example, if U_i = 20 and $L_i = 10$, then given $\xi = 0.9$, these values will be updated to $(\xi \times U_i) = 18$ and $(\xi \times L_i) = 9$, respectively. These representations are further relaxed as a function of R_i and N_i: As the algorithm makes more mistakes, it discounts its representation more rapidly (and thus explores more); given fewer mistakes, it discounts more slowly (and thus exploits more).

Following trial feedback, betasort applies *explicit feedback* (blue in Figure 2) to those stimuli present in the current trial. If the choice was rewarded, both have their current positions consolidated. If the choice was not rewarded, their positions are shifted to improve performance during later trials. Next, betasort applies *implicit inference* (yellow in Figure 2) to the values of all stimuli not presented during the trial. If the choice was rewarded, all inferred positions are consolidated; if not, those stimuli that fall *between* the trial stimuli are consolidated, but those that fall *outside* the trial pair are shifted toward the edge of the sample space. Figure 3 presents relaxation, explicit feedback, and implicit inference during a single incorrect trial. Additionally, a detailed description of betasort is provided in the methods.

To emphasize the importance of the implicit inference stage, we also present the *betaQ* algorithm, which uses beta distributions but only updates the values of stimuli present during the trial (omitting all implicit inference). A detailed description of beta*Q* is provided in the methods.

RESULTS

Here, we present an analysis of TI performance by rhesus macaques and college undergraduates. Additionally, these empirical results are modeled using three algorithms: betasort, beta*Q*, and *Q*/softmax.

Behavioral Results from Rhesus Macaques

Three rhesus macaques completed sessions of TI training, learning novel 7-item lists during every session. Choices were made using eye movements. The six adjacent pairs (AB, BC, . . . , FG) were presented in randomized blocks of twelve pairs each to counterbalance for stimulus position. After 20 blocks of training, subjects were presented with all 21 pairs of stimuli in a similarly counterbalanced fashion.

Figure 4A shows response accuracy (averaged across monkeys) for the non-terminal adjacent pairs (BC, CD, DE, and EF, in red), as well as the critical pairs with ordinal distance 2 (BD, CE, and DF, in orange), 3 (BE and CF, in green), and 4 (BF, in blue). Adjacent-pair performance is close to chance during training, but performance is above chance on non-adjacent pairs at transfer, showing a symbolic distance effect (with highest accuracy for distance 4 pairs, followed by distance 3, etc.). Over the next 400 trials of

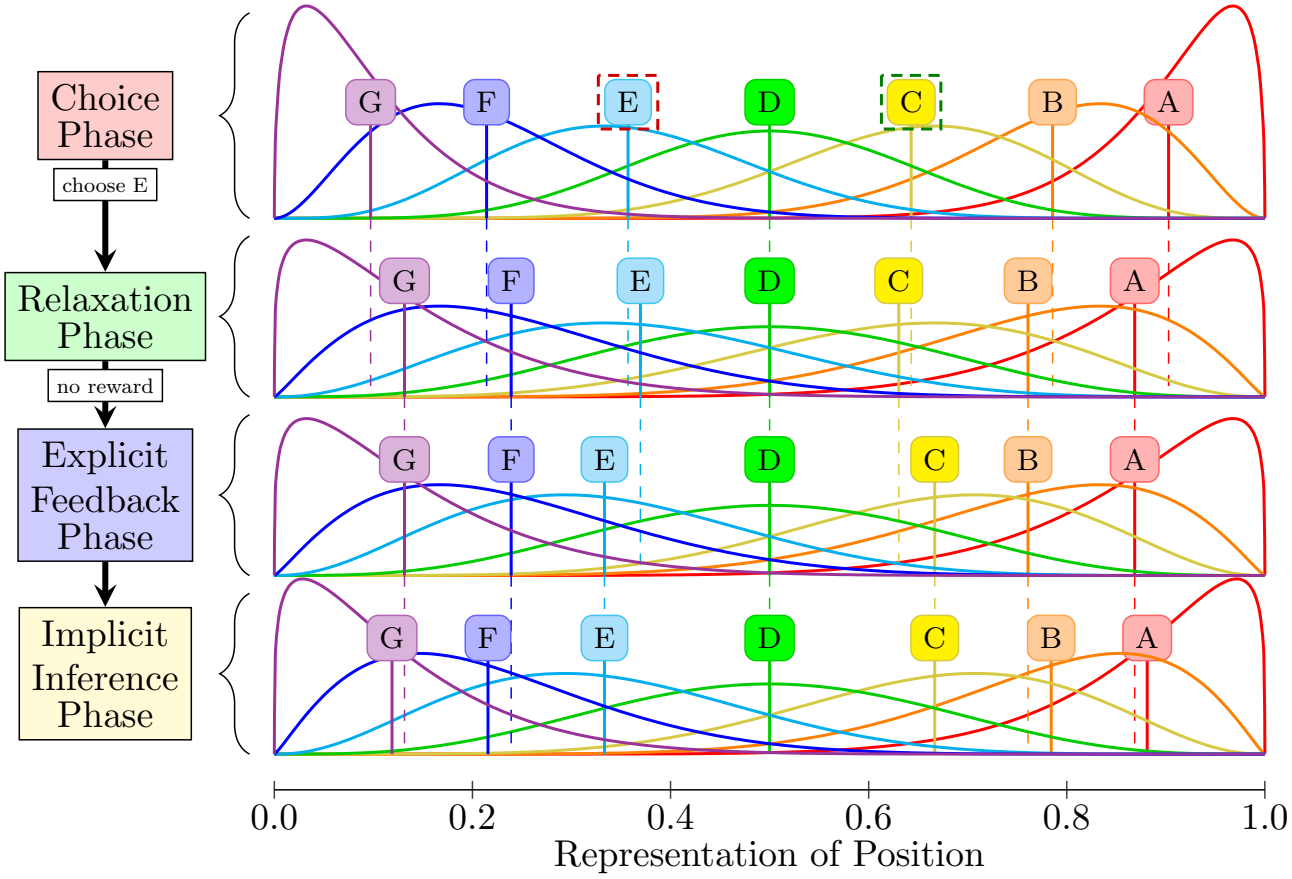

Figure 3. Visualization of Betasort's adjustment of the beta distributions during a single trial in which an incorrect response is given. For this example, the trial stimuli are the pair CE. The initial conditions show the beta distributions of a well-learned list, with means labeled with a vertical line. During the *choice phase*, a value is drawn randomly from the beta distributions of the trial stimuli, and the stimulus with the larger random value is chosen. In this example, the algorithm incorrectly selects stimulus E, an unlikely but possible event. Immediately following the choice, but before feedback is taken into account, the positions of all stimuli are *relaxed* (using $\xi = 0.6$ for this example). This has the effect of making all density functions slightly more uniform, and reduces the influence of older trials in favor of more recent ones. During *explicit feedback*, the increases L_E by one, while also increasing U_C by one. This increases the odds of future selections of stimulus C, while decreasing the odds of future selections of stimulus E. Next, the algorithm makes *implicit inferences* about the positions of all known stimuli that did not appear during the trial. Because stimulus D falls between C and E, its count of successes and failures is consolidated and its position does not change. Meanwhile, because stimuli A and B were greater than the two stimuli considered, their corresponding values for U_A and U_B are increased by one, shifting their position toward the top of the scale. The values L_F and L_G are also increased by one, shifting stimuli F and G respectively toward the bottom of the scale. Once these steps are completed, the algorithm awaits the start of the next trial.

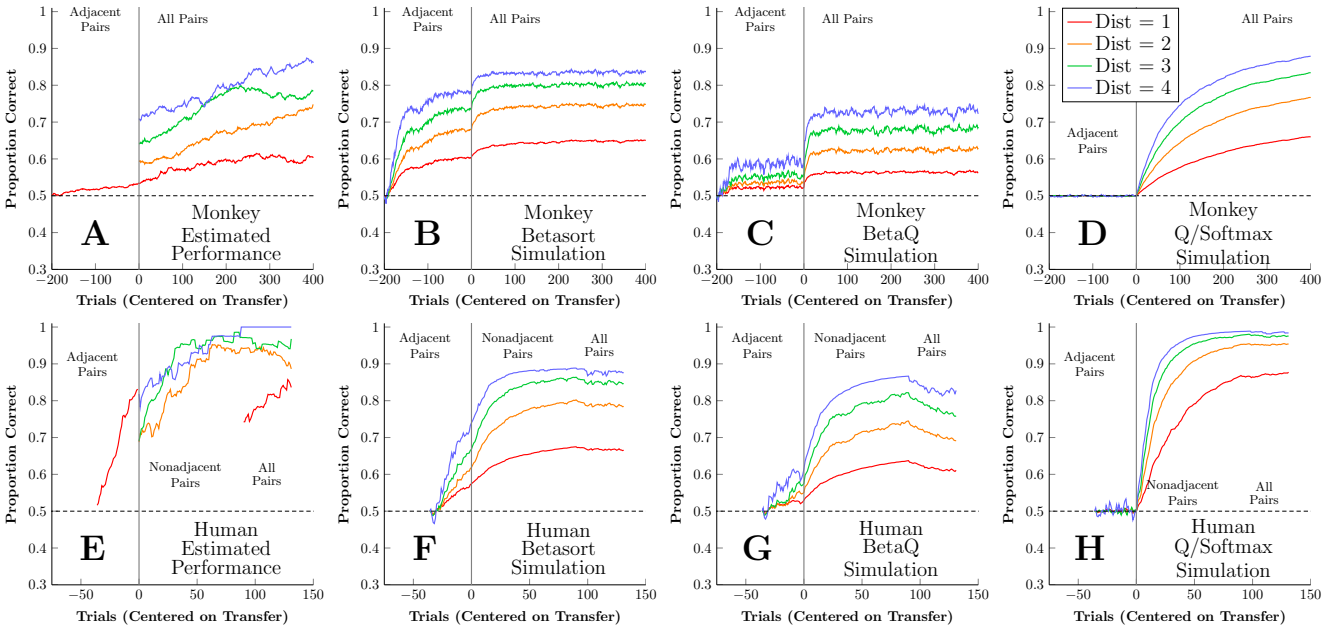

Figure 4. Performance on non-terminal stimulus pairs (i.e. those excluding stimuli A and G) for subjects and algorithms. Trial number is set to zero at the point of transfer from adjacent-pair-only training to testing with other pairs. **(A)** Smoothed response accuracy for three rhesus macaques over 200 trials of adjacent pair training, followed by the first 400 trials of responding to all pairs. Performance is divided into the adjacent pairs (BC, CD, DE, and EF) in red, the pairs with an ordinal distance of two (BD, DE, and CF) in orange, those with a distance of three (BE and DF) in green, and the pair with a distance of four (BF) in blue. Subjects show an immediate distance effect (i.e. increased accuracy as a function of ordinal distance between stimuli) from the first transfer trial. **(B)** Simulated performance using the betasort algorithm, using each monkey's maximum-likelihood model parameters for each session. Since these results are simulated, lines are plotted for all distances at all times, to show how the algorithm would respond had it been presented with trials of each type. Like the monkeys, the algorithm displays an immediate distance effect. **(C)** Simulated performance using the beta$Q$ algorithm, with maximum-likelihood parameters. Although a small distance effect is observed, performance remains close to chance throughout training. **(D)** Simulated performance using the Q/softmax algorithm. Performance remains strictly at chance throughout adjacent-pair training, and only begins to display a distance effect after the onset of the all-pairs trials. **(E)** Performance of human participants given 36 trials of adjacent-pair training, followed by 90 trials of non-adjacent pairs only, and finally 42 trials of all pairs. Unlike the monkeys, participants rapidly acquire the adjacent pairs, and show only a mild distance effect at transfer. **(F-H)** Simulations based on human performance using the three algorithms, analogous to panels B through D. As in the monkey case, Q/softmax displays no distance effect at all until non-adjacent pairs are presented.

training, performance continued to improve.

Each algorithm had two free parameters (noise τ and recall ξ for betasort and beta$Q$; α and β for Q/softmax) that were fit to the monkey data using a maximum likelihood method, as described in the methods below. The sequence of stimulus pairs shown during each session was then presented to each algorithm using that session's best-fitting parameters. Simulated performance was then averaged to compare the algorithms to the monkeys.

Figure 4B shows the average simulated performance of the betasort algorithm, based on the sessions and best-fitting parameters derived from the monkey data. Because simulation permits undisruptive probe trials, putative accuracy for non-adjacent pairs is also plotted during adjacent-pair training. Although monkeys and betasort differ in their particulars, several important hallmarks of the TI behavior are displayed. A distance effect is observed with the non-adjacent critical pairs, which persists over the course of the subsequent training. Contrastingly, Figure 4C shows the average simulated performance of the beta$Q$ algorithm, which displays less resemblance to the monkey data. Although a weak symbolic distance effect is observed, it does not exceed an accuracy of 60% at transfer. Figure 4D, which displays the average simulated performance for the Q/softmax algorithm, resembles the monkey data the least: Its performance on the non-terminal adjacent pairs is precisely 50% throughout training, and no transitive inference is displayed at transfer. Instead, the algorithm begins the all-pairs phase of the experiment at chance on all critical pairs, and only gradually determines their ordering once it has received all-pair training.

Figures 5A-5C presents the contrast between model predictions and animal behavior during the first block of trials following transfer. The betasort algorithm, in red, (Figure 5A) largely aligns with observed response accuracy, in green, for all 21 possible pairs. This includes a distance effect for the critical pairs, which are indicated by a gray backdrop. Confidence intervals were calculated using bootstrapping. Those pairs whose means differ significantly are denoted in Figure 5 with an asterisk below the pair's axis label; 5 such pairs differed significantly in Figure 5A. Contrastingly, the beta$Q$ algorithm (Figure 5B, blue) did less well in approximating performance. While the betasort algorithm tended to overperform on the critical transfer pairs, beta$Q$ tended to underperform. Of the 21 pairs, beta$Q$ differs significantly from observed performance for 12. The Q/softmax algorithm (Figure 5C, brown) transfered poorly: Its goodness of fit was driven by terminal item effects, and all non-terminal pairs displayed chance performance. Of the 21 pairs, 11 differed significantly from observed.

Two omnibus model comparisons of algorithm performance at transfer were calculated for each model, relative to subject accuracy: the Schwarz-Bayes Information Criterion ("SBIC" Kass and Wasserman, 1995) and the log-likelihood ratio ("$\mathcal{G}$" Woolf, 1957). Betasort received the lowest (i.e. most favorable) score in both cases (SBIC = 9248.7, $\mathcal{G} = 186.5$) compared to beta$Q$ (SBIC = 9264.8, $\mathcal{G} = 200.7$) and Q/softmax (SBIC = 9338.6, $\mathcal{G} = 276.4$). This constitutes strong evidence favoring betasort over the two competing models.

Behavioral Results from Humans

19 college undergraduates completed sessions of TI training, using a touchscreen. Training consisted of 36 trials consisting of adjacent pairs, which was then followed by 90 trials consisting of only the non-adjacent pairs. The session then concluded with 42 trials using all pairs. Average response accuracy for the non-terminal pairs is shown in Figure 4E, for four symbolic distances.

As with the monkeys, best-fitting algorithm parameters were identified based on maximum likelihood, and these were used to simulate the behavior of each participant. Figures 4F-4H depict average performance of the simulations at transfer. Although betasort shows the closest approximation of the symbolic distance effect at transfer, several discrepancies are evident. In particular, performance on the non-terminal adjacent pairs is rapidly learned by humans, but is not learned by the algorithms. This suggests that, in addition to

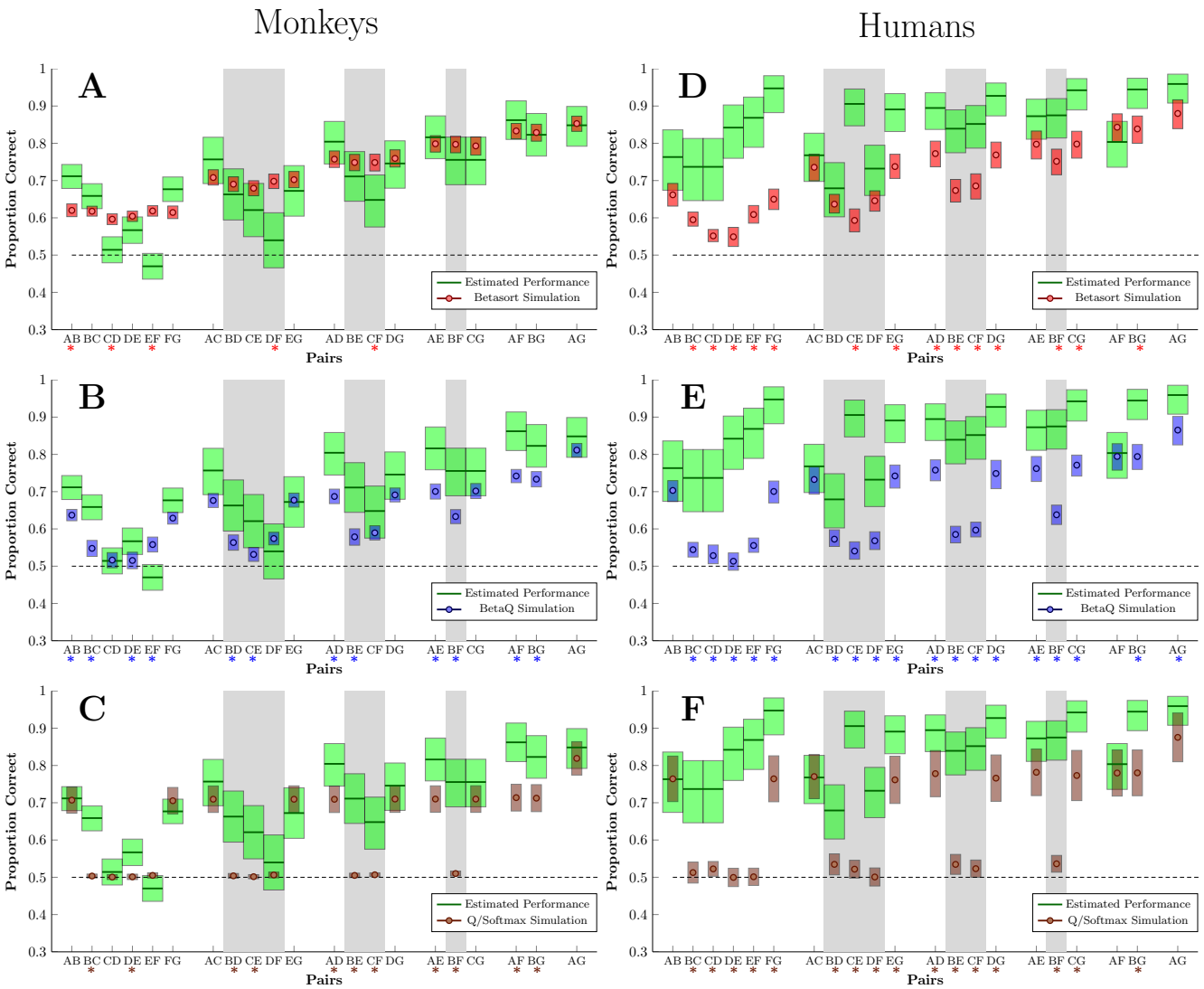

Figure 5. Estimated response accuracy on the first transfer trial for each of the 21 possible pairs, comparing estimates of performance by subjects (blue lines) to those generated by simulations using each algorithm. Those pairs with a gray backdrop are the critical transfer pairs that are not expected to be subject to the terminal item effect. Shaded regions around each point/line represent the 95% confidence interval for the mean, determined using bootstrapping. **(A)** Monkey performance (green) compared to the performance of the betasort algorithm (red), given each session's maximum likelihood parameter estimates. An overall distance effect is reliably observed from the simulation. **(B)** Monkey performance (green) compared to the beta$Q$ algorithm (blue), given maximum likelihood parameters. Although a distance effect is evident among critical pairs, beta$Q$ fails to perform appropriate levels of accuracy. **(C)** Monkey performance (green) compared to the Q/softmax algorithm (brown), given maximum likelihood parameters. Apart from a robust terminal item effect, the algorithm's responding is strictly at chance, including all critical transfer pairs. **(D-F)** Human performance compared to the three algorithms, analogous to figures A through C. Although none of the algorithms precisely resemble the participants, the betasort algorithm comes closest, yielding a distance effect on critical transfer pairs.

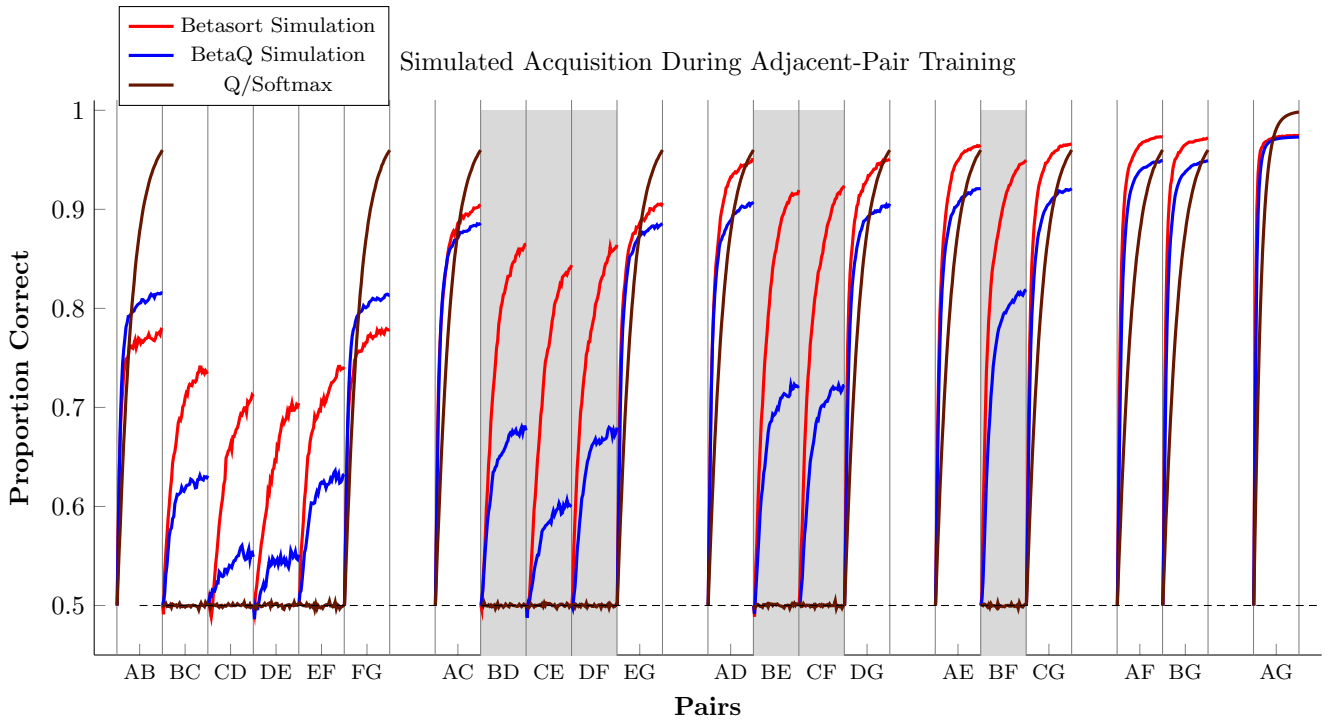

Figure 6. Simulated response accuracy for all stimulus pairs of a seven-item list using betasort (red), beta$Q$ (blue), and Q/softmax (brown) over the course of 200 trials of adjacent-pair training. Critical transfer pairs are indicated with a gray shaded background. Both betasort and beta$Q$ used the same parameters ($\tau = 0.05$, $\xi = 0.95$), while Q/softmax used the parameters ($\alpha = 0.03$, $\beta = 10$). Betasort shows more pronounced transfer in the critical pairs, whereas beta$Q$ shows a more pronounced terminal item effect. Q/softmax rapidly acquires the terminal items, but remains strictly at chance for all non-terminal pairs.

an inferential procedure, participants also used rote memorization.

Figures 5D-5F show performance at transfer for each of the 21 pairs by humans (green) vs. each of the algorithms. Betasort (red) provided the best fit: 14 pairs differed significantly at transfer, based on bootstrapped comparisons of means, compared to 19 and 16 pairs for beta$Q$ (blue) and Q/softmax (brown) respectively. As in the monkey case, the Q/softmax algorithm shows no symbolic distance effect at transfer. Omnibus model comparisons also favor betasort (SBIC =1001.1, $\mathcal{G} = 152.1$) over beta$Q$ (SBIC = 1062.3, $\mathcal{G} = 213.4$) or Q/softmax (SBIC = 1100.1, $\mathcal{G} = 251.3$).

Simulation

Betasort provided a better fit to the human data than beta$Q$ or Q/softmax, but all three displayed poor transfer on critical pairs. This poor fit might reflect the model's inability to do well *in general*, or 36 trials may not be sufficient training. To assess this, each algorithm was presented with extended adjacent-pair training in order to determine how rapidly transitive inference effects were expected to emerge.

Figure 6 displays response accuracy, for each pair, over 200 trials of adjacent-pair training. Using parameters similar to those obtained from the highest-performing human participants ($\tau = 0.05$, $\xi = 0.95$), the betasort algorithm (red) rapidly improved accuracy for all non-adjacent items, exceeding 80% accuracy for critical transfer pairs after 200 trials. The beta$Q$ algorithm (blue), working with the same parameters, fared worse, but nevertheless showed a symbolic distance effect. Q/softmax (brown; $\alpha = 0.03$, $\beta = 10$) remained at precisely chance levels for all non-terminal pairs.

To showcase the trial-by-trial behavior of each algorithm, another simulation was performed, consisting of three phases: (1) 200 trials of adjacent pairs only, (2) 200 trials of all pairs, and then (3) 200 pairs of

only the pair FG. This third phase was included to test the prediction that inferential updating should make betasort's representation of stimulus positions robust against massed trials (unlike RPE models, which are expected to fail; Lazareva and Wasserman, 2012). Rather than response accuracy, Figure 7 depicts the contents of memory for each of the algorithms ($\frac{U_i}{U_i+L_i}$ in the case of betasort and beta*Q*, and Q_i in the case of *Q*/softmax; full density functions are omitted for clarity).

Figure 7A shows expected values for the betasort algorithm. By the end of adjacent pair training, betasort has inferred that items should be spaced evenly over the unit span. Subsequent massed trials do not disrupt the stimulus ordering. Although the value of stimulus F rises, implicit inference ensures that stimuli A through E are modified accordingly.

Figure 7B shows the expected values for the beta*Q* algorithm. Although a mild symbolic distance effect is observed at transfer, non-terminal items remain clustered near the center. During the third phase, massed pairings of FG causes the expected value of stimulus F to move out of order. Because the stimuli A through E are not presented during this period, their values remain static.

Figure 7C depicts the stored memory for the *Q*/softmax algorithm. During training, the values of all non-terminal Q_i remain at 0.5, which is why the algorithm fails at transitive inference. The ordering begins to emerge when all pairs are presented in the second phase, but is disrupted by massed FG trials.

DISCUSSION

We present two new models, betasort and beta*Q*, alongside data from humans and monkeys performing a transitive inference task. Betasort and beta*Q* are neither RPE models nor MDP models, and as such both fall outside the model-based/model-free formalism introduced by Sutton and Barto (1998). Although beta*Q* displays mild inference at transfer, betasort yields a much more pronounced distance effect that better matches the empirical behavior of subjects, particularly the monkeys (Figure 5). This is achieved through active modification of memory for implicit stimuli. Although these algorithms incur low computational cost, they are unambiguously cognitive models: They place each stimulus along a putative number line and track position uncertainty using Bayesian updating (Figures 2 and 3).

Both betasort and beta*Q* demonstrate transitive inference (Figures 4B, 4C, 4F, and 4G). This is accomplished because they track putative position of each stimulus as a serial representation. Positive feedback thus consolidates current position values, even if those values are low. Even without implicit inference, beta*Q* is able to gradually ratchet its way towards reasonable values. However, implicit inference allows betasort to substantially outperform beta*Q*. Another virtue of betasort is that it protects against bias introduced by massed presentation of specific pairs (Figure 7A).

The inability of RPE models to demonstrate TI is a major limitation, because such effects are ubiquitous across vertebrates (Vasconcelos, 2008). It is nevertheless not our aim to argue that betasort is the learning model used by subjects to solve transitive inference problems. Rather, published TI results strongly suggest that organisms make use of representations that have a pseudo-spatial character, and that represented list members benefit from implicit updating. Betasort accomplishes this at low computational expense, and does so outside of the RPE/MDP framework. The failure of *Q*/softmax, on the other hand, reflects the more general conclusion that TI performance cannot be explained by associative strength alone.

Explaining learning using only RPE is an old ambition of the behaviorist tradition, from its original specification (Bush and Mosteller, 1951) to modern associative theory (Siegel and Allan, 1996). This desire to explain complex behavior using mathematical laws (assumed to apply across species despite radically differing brain architecture) contrasts with the cognitive tradition, which emphasizes analogous representation and information processing (Acuna et al., 2002). On the surface, model-free and model-based RL algorithms appear to map onto associative and cognitive mechanisms, respectively (e.g. Gläscher

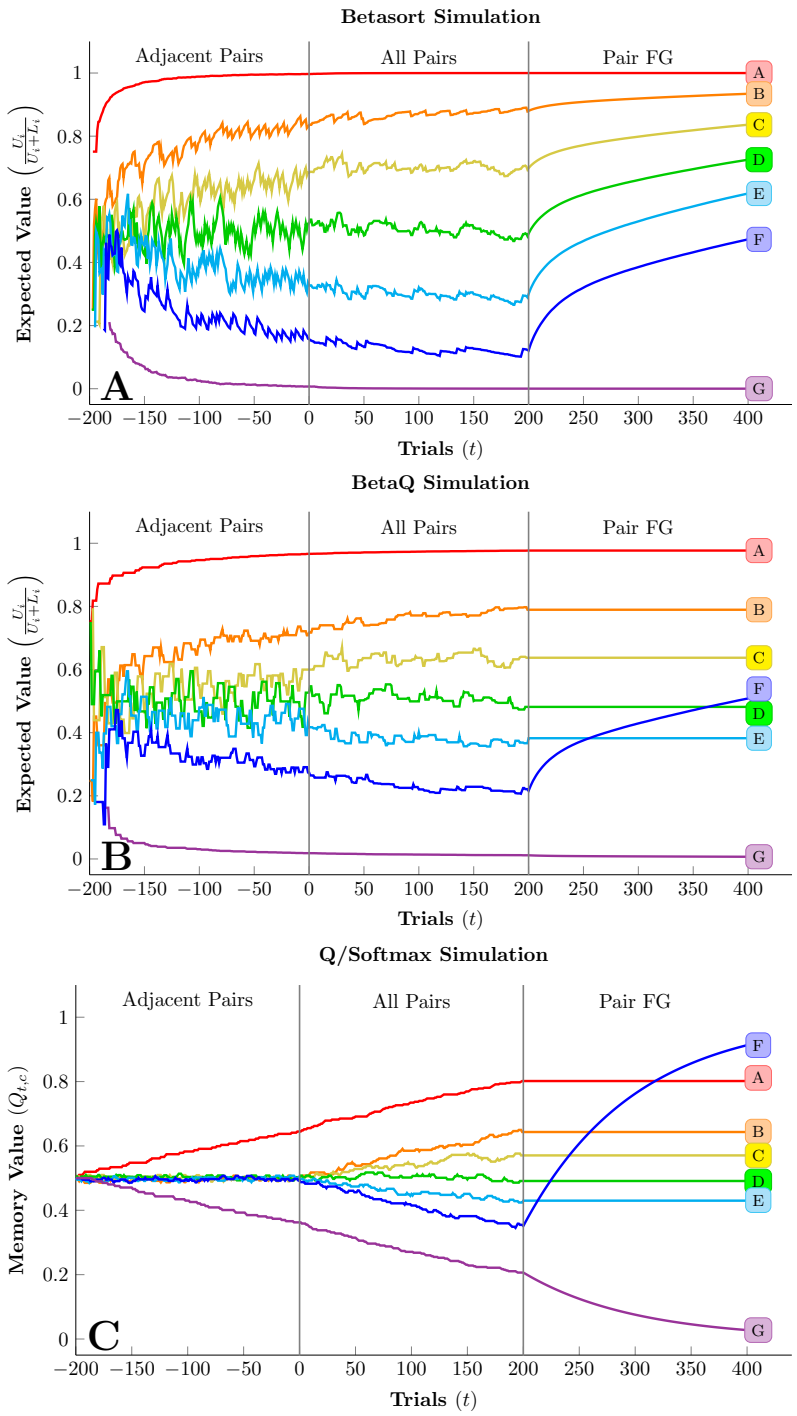

Figure 7. Visualization of the contents of memory for the three algorithms under simulated conditions: 200 trials of adjacent pairs only, followed by 200 trials of all pairs, and then followed by 200 massed trials of only the pair FG. **(A)** Expected value for each stimulus under the betasort algorithm, given parameters of $\tau = 0.05$ and $\xi = 0.95$. Not only is learning during adjacent pair training faster, but mass trials of FG do not disrupt the algorithm's representation of the order, because occasional erroneous selection of stimulus G increases the value of all stimuli, not just stimulus F. **(B)** Expected value for each stimulus under the beta$Q$ algorithm, given parameters of $\tau = 0.05$ and $\xi = 0.95$. Although the algorithm derives an ordered inference by the time the procedure switches to all pairs, that order is not preserved during the massed trials of FG, as a result of the lack of inferential updating. **(C)** Expected value **Q** for each stimulus under the Q/softmax algorithm, given parameters of $\alpha = 0.03$ and $\beta = 10$. Values for non-terminal items remain fixed at 50% throughout adjacent pair training, and only begin to diverge when all pairs are presented in a uniformly intermixed fashion. Subsequent massed training on the pair FG disrupts the ordered representation because rewards drive the value of stimulus F (and the value of stimulus G down) while the other stimuli remain static.

et al., 2008; McDannald et al., 2009). This informal analogy is at odds, however, with machine learning's formalism. Careful comparison reveals that these distinctions are orthogonal; for example, associative theories of Pavlovian learning do not cleanly map onto "model-free" RL (Dayan and Berridge, 2014). While machine learning's rigorous formalism is both admirable and useful, phenomena such as TI suggest that associative/cognitive distinctions are more fruitful for future theorizing about the brain.

Despite this, a very considerable literature has accumulated arguing from neural evidence that the brain makes use of model-free learning mechanisms. Some studies, particularly those involving midbrain dopamine neurons (Schultz et al., 1997; Waelti et al., 2001; Fiorillo et al., 2003) have been cited extensively. These report graded dopamine firing as a function of prediction error, which is taken to be consistent with RPE learning. Problematically, such results could instead reflect other models that monitor expectation-outcome discrepancy, such as information-theoretic models (Ward et al., 2013), Bayesian models (Behrens et al., 2007), or incentive salience models (Berridge, 2007). While it is unsurprising that signals correlated with prediction error are observed in the brain, these are not sufficient to justify the conclusion that a "model-free learning system" is at work.

A growing literature reports equally correlational evidence of model-based brain activity (Doll et al., 2015). Such interpretations must be taken with a grain of salt, however, because MDP models necessarily specify and maintain a state-action transition matrix. In practice, models sensitive to contextual effects should correlate with MDP predictions, including associative models of second-order conditioning. It should therefore come as no surprise that ostensibly model-based signals are ubiquitous in brain imaging (Doll et al., 2012). Furthermore, insofar as MDPs can be represented as recurrent neural networks, their specification is somewhat arbitrary, and the biological validity of any particular network configuration is unclear (Green, 2001). It is time for neuroscience to wean itself of the habit of "confirming" models in the brain on the basis of neural activity correlated with some aspect of those models.

That said, the quest to decipher the brain's cognitive machinery faces substantial obstacles. In the context of transitive inference and models like betasort, the clearest difficulty is implicit updating. As described, betasort updates every stimulus during every trial. This implies that, on any given trial, neural signals will be observed that relate to stimuli not presently visible. It is unclear how such implicit signals could be extracted from either single-unit recordings or from fMRI data. Nevertheless, evidence from behavior suggests that such updating is likely taking place. The difficulty of detecting these implicit mechanisms is a challenge for neuroscience techniques, not a weakness of cognitive theories. Until our understanding of brain networks advances to a stage that permits more comprehensive examination of the contents of memory, making theoretical commitments to specific mathematical formalisms hinders the discovery of other plausible accounts.

MATERIALS & METHODS

Notation

Let c_t denote the index associated with a subject's choice at time t. Let r_t indicate the delivery of a reward (or lack thereof), indicated by a value of 1.0 or 0.0 respectively. Let $\aleph$ denote the set of all stimuli presently employed in the experiment. $\aleph+_t$, denotes only those stimuli that are presented during the current trial, while $\aleph-_t$, denotes those stimuli whose presence is implied by past experience but are not currently visible.

In several cases below, Hadamard (i.e. entry-wise) matrix multiplication and division are used in the interest of concision. In these cases, multiplication is denoted by the operator $\otimes$ and division is denoted by the operator $\oslash$ (not to be confused with the empty set).

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.954v1 | CC-BY 4.0 Open Access | rec: 2 Apr 2015, publ: 2 Apr 2015

Q/softmax Model Specification

Memory Structure: Information about the stimuli was stored in a 1×7 vector denoted by $\mathbf{Q}$, with each column indicating the expected value of a given stimulus on a scale from 0.0 to 1.0. By convention, the value of stimulus i as time t is denoted by $Q_t(i)$ to clearly delineate time and stimulus index. Each value of $Q_t(i)$ is initialized to a value of 0.5 when $(t = 0)$. Although no *Q*-learning method is ever truly "model-free" in the cognitive sense, this constitutes the simplest model of memory that accommodates RPE-based updating.

It is important to note that, in theoretical discussions of reinforcement learning, the value stored in memory is routinely denoted by Qt(st, at), where st refers to a current state, whereas at refers to a particular action in that state. This formalism is particularly ill-suited to the transitive inference procedure, however, because the states about which we are curious during testing (i.e. non-adjacent pairs) have never before been seen. The manner in which the model extrapolates to these hitherto-unknown states must be formally specified. Rather than smuggle inference in at the extrapolation stage, the present model is limited to ascribing an expected value to each stimulus and extrapolating on the basis of the relative values in any stimulus pairing.

Choice Policy: Stimuli are selected by this algorithm using the softmax function, which has one free parameter β, such that $\beta \geq 0$:

$$p\left(c_t = i | \aleph+_t, \mathbf{Q}_t\right) = \frac{\exp\left(\beta \cdot Q_t\left(i\right)\right)}{\sum\limits_{j \in \aleph+_t} \exp\left(\beta \cdot Q_t\left(i\right)\right)} \tag{2}$$

Updating Policy: This algorithm uses the most basic form of temporal difference reward-prediction error, which has a single free parameter α, such that $0 \leq \alpha \leq 1$:

$$\begin{aligned} &Q_{t+1}\left(i\right) = Q_t\left(i\right) + \alpha\delta_t\left(i\right), \text{ for all } i, \text{ given that} \\ &\delta_t\left(i\right) = \begin{cases} 1 - Q_t\left(i\right) & i \in \aleph+_t \wedge \left(\left(r_t = 1 \wedge c_t = i\right) \vee \left(r_t = 0 \wedge c_t \neq i\right)\right) \\ -Q_t\left(i\right) & i \in \aleph+_t \wedge \left(\left(r_t = 0 \wedge c_t = i\right) \vee \left(r_t = 1 \wedge c_t \neq i\right)\right) \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{3}$$

Thus, the value of $Q_t(i)$ for every stimulus i is updated at time t, but only those present during the current trial (i.e. $i \in \aleph+_t$) are updated as a consequence of the feedback r_t. If the choice was rewarded, the value of the chosen stimulus is increased by some factor of the discrepancy between the reward and the value, while the unchosen stimulus has its value correspondingly decreased. If, on the other hand, the choice was not rewarded, the opposite occurred: The selected stimulus was decreased and the unchosen alternative was increased. This updating process is also described by the pseudocode in Algorithm 1

Note that many RPE implementations do not use this symmetrical structure, and instead only update the stimulus that was selected. We implemented both the version above and a chosen-stimulus-only updating procedure, and these yielded nearly indistinguishable results.

Parameter Estimation: Since softmax gives the probability of selecting an outcome directly (once β is specified), and since the value of $Q_t(i)$ is straightforwardly defined for every trial (once α is specified), it is therefore easy to calculate the log-likelihood associated with a set of parameters (α, β), given an observed history of choices and responses (as described by Daw, 2011). Because *Q*/softmax is an iterative algorithm, no closed-form solution exists for finding the parameters that maximum the likelihood. Consequently, these were identified using the `fminsearch()` optimizer packaged with Matlab 2014b (The MathWorks, Inc.). In general, optimal parameters of β were large (i.e., greater than 4), in order to guarantee the stimulus

Algorithm 1: The Q-learning updating policy.

Data: memory array **Q**, chosen stimulus ch, unchosen stimulus nc, outcome r, modifier α
Result: updated model **Q**
begin
  if $r = 1$ **then**
    $Q_{ch} \leftarrow Q_{ch} + \alpha \cdot (1 - Q_{ch})$ /* shift ch up */
    $Q_{nc} \leftarrow Q_{nc} - \alpha \cdot Q_{nc}$ /* shift nc down */
  else if $r = 0$ **then**
    $Q_{ch} \leftarrow Q_{ch} - \alpha \cdot Q_{ch}$ /* shift ch down */
    $Q_{nc} \leftarrow Q_{nc} + \alpha \cdot (1 - Q_{nc})$ /* shift nc up */
  return **Q**

with the greater value was selected almost exclusively. This, in turn guaranteed values of α that were very small, to prevent preferences becoming too extreme too quickly.

Betasort Model Specification

The procedure for betasort is depicted in Figure 8.

Memory Structure: The betasort algorithm makes use of four 7×1 matrices to track feedback concerning the available stimuli: **U**, **L**, **R**, and **N**. The vector **U** tracks cumulative successes, such that U_i indicates the degree to which stimulus i is close to the top of the unit scale. The vector **L** plays a similar role the bottom end of the scale. Jointly, **U** and **L** provide the parameters to the beta distributions that represent the estimated position of each stimulus on the unit span. Meanwhile, **R** and **N** store rewarded and unrewarded trials for each stimulus, respectively. Thus, if $R_i = 10.5$ and $N_i = 4.5$, then the algorithm estimates 70% probability of reward during trials in which i was present, based on the last 15 trials. Although all four vectors conceptually represent sums of discrete events, they support fractional values, resulting from the relaxing phase of the updating policy.

Choice Policy: At stimulus onset for every trial, each stimulus in the set $\aleph+_t$ had a number X_i drawn at random either from a beta distribution, parameters governed by past learning U_i and L_i, or else draws these values from a uniform distribution (in which case behavior is entirely random). The odds of choosing entirely randomly is governed by the "noise" parameter τ, such that $0 \leq \tau \leq 1$:

$$X_i = \begin{cases} Beta\,(1,1) & \text{if } Rnd < \tau \\ Beta\,(U_i + 1, L_i + 1) & \text{otherwise} \end{cases} \tag{4}$$

A value of 1 is added to U_i and L_i in order to act as a prior on the probability distribution. This also prevents the distribution from approaching a singularity as a consequence of some edge conditions during updating. The betasort choice policy is to select the alternative whose random value is largest:

$$c_t = i \text{ such that } X_i = \max\left(X_{i \in \aleph+_t}\right) \tag{5}$$

This choice policy is only marginally more expensive than softmax, in that it involves only drawing random values from a handful of closed-form probability density functions. It also has the added benefit that, because the absolute values of U_i and L_i are preserved, they govern the narrowness of the beta distributions and thus model a subject's growing accuracy as a function of increased experience. Updating

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.954v1 | CC-BY 4.0 Open Access | rec: 2 Apr 2015, publ: 2 Apr 2015

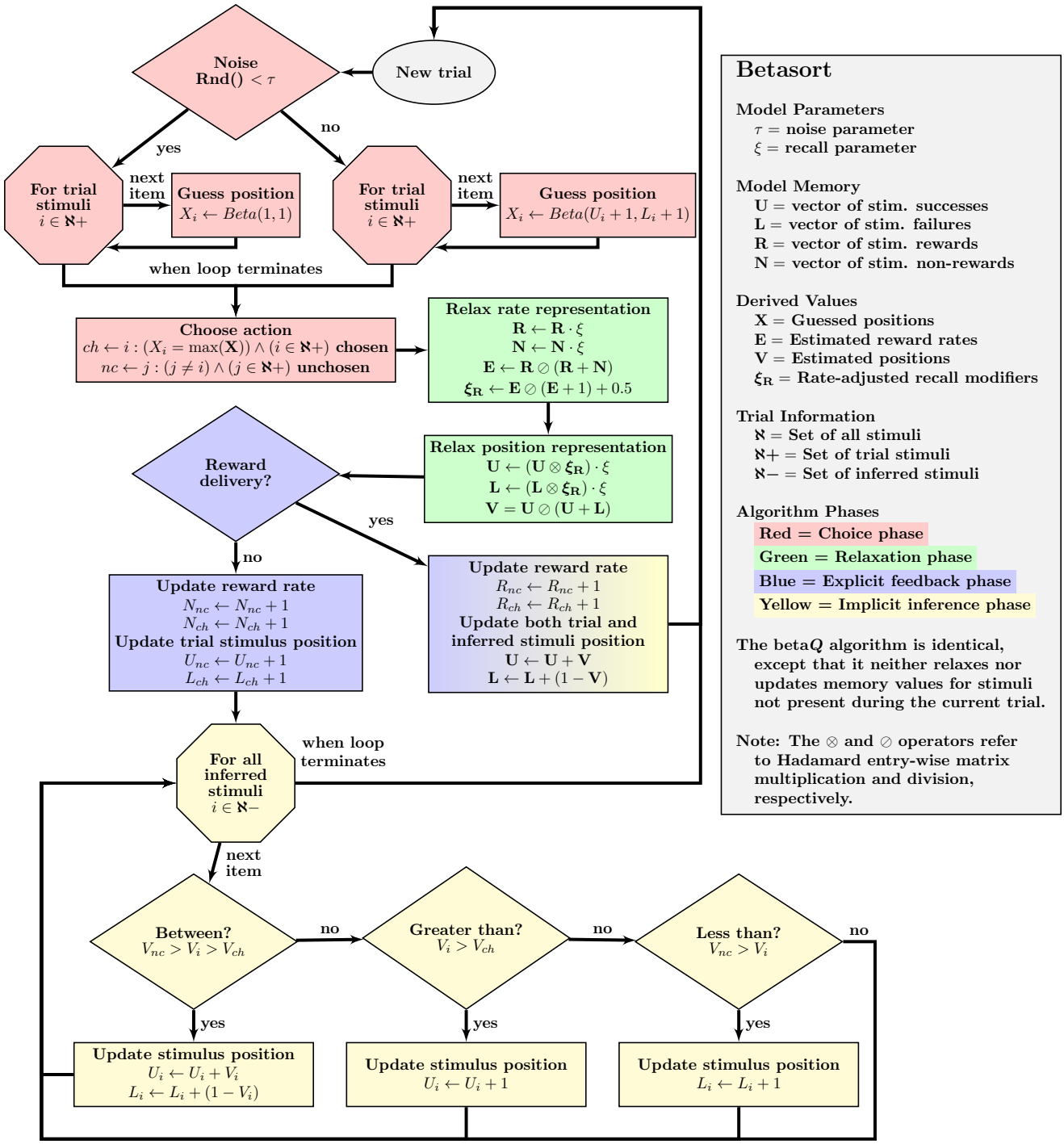

Figure 8. A schematic specification of the betasort algorithm over the course of one trial. Rectangles refer to operations, diamonds to logical branches, and octagons to loops that iterate over sets of items. Four phase are depicted: the choice policy (red), the relaxation of the contents of memory (green), the processing of explicit feedback (blue), and implicit inference (yellow).

Policy: Betasort's updating policy involves three stages: Relaxation, explicit updating, and implicit inference.

Updating Policy: Betasort's updating policy involves several steps. These are presented in the pseudocode for Algorithm 2

The first step in updating is relaxation, which weakens the influence of old information in favor of more recent feedback. This is governed by the "recall" parameter ξ, such that $0 \leq \xi \leq 1$. All four vectors ($\mathbf{U}$, $\mathbf{L}$, $\mathbf{R}$, and $\mathbf{N}$) are multiplied by ξ, so all qualify as "leaky accumulators" (Usher and McClelland, 2001), steadily decreasing in absolute value. These loses are then counteracted by subsequent updating. In addition to ξ, the values of S and F are further relaxed by a factor $\xi_{\mathbf{R}}$ based on the reward rate accrued during trials in which a given stimulus was present. When accuracy is high, this additional relaxation is minimal; however, when accuracy is low, more aggressive relaxation yields greater variability in behavior, helping to keep the algorithm from being trapped in local minima. Collectively, relaxation makes the following modifications:

$$\begin{aligned}
\mathbf{R} &\leftarrow \mathbf{R} \cdot \xi \\
\mathbf{N} &\leftarrow \mathbf{N} \cdot \xi \\
\boldsymbol{\xi}_{\mathbf{R}} &\leftarrow (\mathbf{R} \oslash (\mathbf{R} + \mathbf{N})) \oslash (\mathbf{R} \oslash (\mathbf{R} + \mathbf{N}) + 1) + 0.5 \\
\mathbf{U} &\leftarrow (\mathbf{U} \otimes \boldsymbol{\xi}_{\mathbf{R}}) \cdot \xi \\
\mathbf{L} &\leftarrow (\mathbf{L} \otimes \boldsymbol{\xi}_{\mathbf{R}}) \cdot \xi
\end{aligned} \tag{6}$$

Subsequent updating depends on a vector of "expected values" $\mathbf{V}$ of each stimulus. These are not expected values in the econometric sense, but instead represent the best estimate of the position of each stimulus along the unit span:

$$\mathbf{V} = \mathbf{U} \oslash (\mathbf{U} + \mathbf{L}) \tag{7}$$

In the event that $(U_i = L_i = 0.0)$, V_i is set to 0.5. Subsequent updating depends on (1) whether the trial resulted in a reward or not, (2) whether each stimulus i was part of the set present during the trial or not, and (3) the relative values of $\mathbf{V}$.

If the response is rewarded, then the algorithm consolidates its current estimates. This is done by increasing every U_i by an amount equal to V_i, whereas every L_i is increased by an amount equal to $(1?V_i)$:

$$\begin{array}{c}\text{Choice was} \\ \text{correct}\end{array} = \left\{ \begin{array}{c} \mathbf{U} \leftarrow \mathbf{U} + \mathbf{V} \\ \mathbf{L} \leftarrow \mathbf{L} + 1 - \mathbf{V} \end{array} \right. \tag{8}$$

This is done regardless of whether the stimulus was present on the current trial. If, on the other hand, the response was not rewarded, then L_{chosen} is increased by one, as is $U_{notchosen}$. Then, for all other stimuli not present during the trial, their values are updated as a function of their V_i relative to the stimuli presented:

$$\begin{array}{c}\text{Choice was} \\ \text{incorrect}\end{array} = \left\{ \begin{array}{l} U_{i \in \aleph_{-t}} \leftarrow \left\{ \begin{array}{ll} U_i + V_i & \text{if } V_{notchosen} > V_i > V_{chosen} \\ U_i + 1 & \text{if } V_i > V_{notchosen} \end{array} \right. \\ L_{i \in \aleph_{-t}} \leftarrow \left\{ \begin{array}{ll} L_i + 1 - V_i & \text{if } V_{notchosen} > V_i > V_{chosen} \\ L_i + 1 & \text{if } V_{chosen} > V_I \end{array} \right. \end{array} \right. \tag{9}$$

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.954v1 | CC-BY 4.0 Open Access | rec: 2 Apr 2015, publ: 2 Apr 2015

Algorithm 2: The betasort updating policy.

Data: memory arrays $\mathbf{U}$, $\mathbf{L}$, $\mathbf{R}$, $\mathbf{N}$, chosen stimulus ch, unchosen stimulus nc, outcome r, recall ξ
Result: updated model $\mathbf{U}$, $\mathbf{L}$, $\mathbf{R}$, $\mathbf{N}$

```
begin
    R ← R · ξ; N ← N · ξ                                  /* Relax R and N */
    E ← R ⊘ (R + N)                                       /* Estimate trial reward rates */
    ξ_R ← E ⊘ (E + 1) + 0.5
    U ← U · ξ_R · ξ; L ← L · ξ_R · ξ                      /* Relax U and L */
    V ← U ⊘ (U + L)                                       /* Estimate stimulus positions */
    if r = 1 then
        U ← U + V; L ← L + (1 − V)                        /* consolidate all stimuli */
    else
        U_nc ← U_nc + 1                                   /* shift nc up */
        L_ch ← L_ch + 1                                   /* shift ch down */
        for j = 1 to 7 do
            if j ≠ ch AND j ≠ nc then
                if V_j > V_ch AND V_j < V_nc then
                    U_j ← U_j + V_j; L_j ← L_j + (1 − V_j)    /* consolidate j */
                else if V_j < V_nc then
                    L_j ← L_j + 1                         /* shift j down */
                else if V_j > V_ch then
                    U_j ← U_j + 1                         /* shift j up */
    return U, L, R, N
```

Thus, in the cases where the response was incorrect, the algorithm consolidates the representation of those stimuli falling between the pair, and pushes those lying outside the pair outward toward the margins.

Although this procedure involves a number of logical comparisons, its adjustments are otherwise strictly arithmetic, and can be computed rapidly without recourse to bootstrapping.

Parameter Estimation: Although generating choices and updating memory can both be accomplished rapidly, computing the likelihood of an observed response is more computationally costly. Doing so requires computing the incomplete beta distribution:

$$I\left(x|U_i, L_i\right) = \int_0^x Beta\left(x|U_i, L_i\right) dx \tag{10}$$

In the case of a two-stimulus trial, the odds of stimulus A being chosen over stimulus B, the odds depend both on the noise parameter τ and integrating over two convolved distributions (Raineri et al., 2014):

$$p\left(c_t = A|\aleph+_t = \{A, B\}, \tau\right) = \frac{\tau}{2} + (1-\tau)\int_0^1 Beta\left(x|U_A, L_A\right) I\left(x|U_B, L_B\right) dx \tag{11}$$

Given this formula, computing log-likelihoods associated with the parameters (τ, ξ) for a set of observed data can be performed in much the same manner as with the Q/softmax algorithm.

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.954v1 | CC-BY 4.0 Open Access | rec: 2 Apr 2015, publ: 2 Apr 2015

Algorithm 3: The beta$Q$ updating policy.

Data: memory arrays $\mathbf{U}$, $\mathbf{L}$, $\mathbf{R}$, $\mathbf{N}$, chosen stimulus ch, unchosen stimulus nc, outcome r, recall ξ
Result: updated model $\mathbf{U}$, $\mathbf{L}$, $\mathbf{R}$, $\mathbf{N}$
begin
  $R_{ch} \leftarrow R_{ch} \cdot \xi$; $R_{nc} \leftarrow R_{nc} \cdot \xi$ /* Relax R_{ch} and R_{nc} */
  $N_{ch} \leftarrow N_{ch} \cdot \xi$; $N_{nc} \leftarrow N_{nc} \cdot \xi$ /* Relax N_{ch} and N_{nc} */
  $\mathbf{E} \leftarrow \mathbf{R} \oslash (\mathbf{R} + \mathbf{N})$ /* Estimate trial reward rates */
  $\boldsymbol{\xi}_{\mathbf{R}} \leftarrow \mathbf{E} \oslash (\mathbf{E} + 1) + 0.5$
  $U_{ch} \leftarrow U_{ch} \cdot \boldsymbol{\xi}_{\mathbf{R}_{ch}} \cdot \xi$; $U_{nc} \leftarrow U_{nc} \cdot \boldsymbol{\xi}_{\mathbf{R}_{nc}} \cdot \xi$ /* Relax U_{ch} and U_{nc} */
  $L_{ch} \leftarrow L_{ch} \cdot \boldsymbol{\xi}_{\mathbf{R}_{ch}} \cdot \xi$; $L_{nc} \leftarrow L_{nc} \cdot \boldsymbol{\xi}_{\mathbf{R}_{nc}} \cdot \xi$ /* Relax L_{ch} and L_{nc} */
  $\mathbf{V} \leftarrow \mathbf{U} \oslash (\mathbf{U} + \mathbf{L})$ /* Estimate stimulus positions */
  if $r = 1$ **then**
    $U_{ch} \leftarrow U_{ch} + V_{ch}$; $L_{ch} \leftarrow L_{ch} + (1 - V_{ch})$ /* consolidate ch */
    $U_{nc} \leftarrow U_{nc} + V_{nc}$; $L_{nc} \leftarrow L_{nc} + (1 - V_{nc})$ /* consolidate nc */
  else
    $U_{nc} \leftarrow U_{nc} + 1$ /* shift nc up */
    $L_{ch} \leftarrow L_{ch} + 1$ /* shift ch down */
  return $\mathbf{U}$, $\mathbf{L}$, $\mathbf{R}$, $\mathbf{N}$

Unfortunately, τ and ξ are not strictly orthogonal: Performance near chance can alternatively be explained by high values for τ or low values for ξ. To avoid unstable parameter estimates, values of τ were estimated heuristically, based on the observation that subjects reliably showed near-ceiling performance on the pairs AF, BG, and AG. Under the assumption that the integral in Equation S10 equals 1.0 for those pairs, a bit of arithmetic yields the following estimate:

$$\tau \approx 2 - 2p\left(correct \middle| \{A, F\} \vee \{B, G\} \vee \{A, G\}\right) \tag{12}$$

Having set this parameter, we then used the `fminsearch()` optimizer packaged with Matlab 2014b (The MathWorks, Inc.) to identify the maximum likelihood parameter estimate for ξ. Parameters were obtained for each session.

Beta*Q* Model Specification

Memory Structure: Identical to betasort: four 7×1 arrays, meant to represent successes $\mathbf{U}$, failures $\mathbf{L}$, rewards $\mathbf{R}$, and nonrewards $\mathbf{N}$ for all stimuli.

Choice Policy: Identical to betasort.

Updating Policy: Incorporates only those aspects of betasort's updating policy that relate to the presently-visible stimuli. This yields the pseudocode presented in Algorithm 3.

In this respect, beta$Q$ preserves the longstanding associative assumption that only the stimuli present during the current trial can have their corresponding values updated.

Parameter Estimation: Identical to betasort, differing only in its use of the beta$Q$ updating policy.

Monkey Experimental Procedures

Subjects: Subjects were three male rhesus macaques (*Macaca mulatta*). Subject treatment conformed with the guidelines set by the U.S. Department of Health and Human Services (National Institute of

Health) for the care and use of laboratory animals. All protocols were approved by the Institutional Animal Care and Use Committee at Columbia University, and at the New York State Psychiatric Institute. Monkeys were prepared for experiments by surgical implantation of a post used for head restraint, and a scleral search coil (Judge et al., 1980). All surgery was performed under general anesthesia (isoflurane 1-4%) and aseptic conditions. Monkeys were then trained using positive reinforcement to sit in a primate chair for the duration of the experiment with their heads restrained and to perform visual discrimination and eye movement tasks for liquid rewards while eye movements were recorded. Although subjects had extensive experience ($\geq$ 6 months) with a version of the task during which all pairs were presented in a counterbalanced fashion (a procedure identical to that reported by Jensen et al., 2013), they were naïve with respect to the adjacent-pair training procedure at the beginning of the experiment.

Apparatus: Subjects were seated in an upright primate chair while head movements were restrained by head post. Visual stimuli were generated by a VSG2/5 video controller (CRS, Cambridge, UK). The output from the video controller was displayed on a calibrated color monitor with a 60 Hz non-interlaced refresh rate. The spatial resolution of the display was 1280 pixels by 1024 lines. The video controller was programmed to send out digital pulses (frame sync) for timing purposes at the beginning of each video frame in which a stimulus was turned on or off. These pulses were recorded by the computer using a hardware timer and stored together with the eye movement data. Unless otherwise noted, the apparatus was identical to that described by Teichert et al. (2014).

Eye position was recorded using a monocular scleral search coil system (Robinson, 1963; Judge et al., 1980, CNC Engineering, Seattle, WA). Horizontal and vertical eye position signals were digitized with 12-bit resolution at a sampling rate of 1 KHz per channel. Eye velocity was computed offline by convolving eye position with a digital filter, constructed by taking the first derivative of a temporal Gaussian, $G(t)$:

$$\frac{dG}{dt} = -k \cdot t \cdot \exp\left(\frac{-t^2}{\theta^2}\right) \tag{13}$$

Here, $\theta = 8$ msec, and k is a constant that sets the filter gain to 1.0. This filter does not introduce a time shift between the position input and velocity output, but adds temporal uncertainty to the velocity estimates. Horizontal eye velocities $h'(t)$ and vertical eye velocities $v'(t)$ were combined to estimate radial eye speed $r'(t)$, where speed is the magnitude of the two-dimensional velocity vector:

$$r'(t) = \sqrt{h'(t)^2 + v'(t)^2} \tag{14}$$

Eye speed was used to estimate the onset of saccadic eye movements.

Procedure: A "session" is here defined as a continuous string of trials using a particular ordered list of seven photographic stimuli. In some cases, subjects performed more than one session during a day. These photographic stimuli were randomly selected in advance from a bank of 2500 stock photographs. The only criteria for assembling such a list was that stimuli were visually checked to ensure that they did not appear easy to confuse for one another. Although the list ordering was stored in the computer, it was never displayed or otherwise explicitly communicated to the monkey. By convention, we will refer to the seven items presented during a session using the letters A through G. Here, stimulus A is considered the "first" (i.e. "best") list item and was always rewarded when selected, whereas stimulus G is considered the "last" (i.e. "worst") list item and its selection was never rewarded. For the remaining stimuli (B through F), its selection was only rewarded when it held an earlier list position than any other visible stimulus.

Individual trials began with a central fixation point (0.5 deg red square) for 933 ms. Following fixation, two stimuli were presented on opposite sides of the central point. The fixation target disappeared at the

same time the pictures appeared. Subjects responded by making a saccade to one stimulus or the other, and then fixating on that stimulus for 0.5 s, at which time feedback was provided. Failure to perform the initial fixation, or to saccade to a stimulus within 4 s of stimulus presentation, led to the trial being deemed "incomplete." Once a saccade was made to the chosen stimulus, subjects were required to fixate for 1 s on the stimulus in order to receive a reward of 3 to 5 drops of juice. Number of drops varied day to day as a function of subject motivation, but was always held constant within a session.

Each session was divided into "blocks" of trials. A block consisted of randomly permuted presentations of a set of stimulus pairs, counterbalanced for screen position. For example, the six adjacent pairs in a session were the pairs AB, BC, CD, DE, EF, and FG. A block of adjacent pair trials would thus consist of twelve trials, with each pair presented twice (e.g. once with the on-screen arrangement AB, and once as BA). Subjects did not begin a new block until they had completed all trials in a previous block. If the monkey made an incorrect response, the trial was not repeated within the same block. In the event that a trial was deemed incomplete, another pair was randomly selected from the list of pairings not yet completed in that block. These steps ensured that there was equal information provided about all stimuli.

Each session began with an adjacent-pair training phase, subjects completed 20 blocks consisting of only the six adjacent pairs (240 trials total). Subjects then completed an additional ten blocks (counterbalanced for position and presented without replacement) of all twenty-one pairs (420 trials total). Throughout the session, the only information subjects received about the stimuli were the rewards (or lack thereof) during completed trials; at no point were they shown all stimuli at once, nor were there position cues suggesting stimulus ordering. In general, responding resembled that seen when macaques perform TI using a touchscreen apparatus (Jensen et al., 2013), suggesting that performance was modality-independent.

Human Experimental Procedures

Participants: Nineteen college undergraduate volunteers gave written consent to participate in the experiment for course credit. The study was approved under protocol IRB-AAAA7861 by the Institutional Review Board in the Human Research Protection Office of Columbia University.

Apparatus: Participants selected stimuli by touching them using a touchscreen (Keytec, Inc.) mounted on a 15 in desktop computer monitor. Unless otherwise noted, the apparatus was identical to that described by Merritt and Terrace (2011).

Procedure: To introduce them to the task, participants first completed a session with a list of seven novel images, consisting of 4 blocks (168 trials) of all pairs. Positive feedback was indicated with a magenta screen and a bell sound, and negative feedback was indicated with a black screen and a whoosh sound. Feedback was always provided immediately following each touch. Apart from being told how to distinguish positive vs. negative feedback, and an instruction to "do as well as you can," participants were given no further instruction regarding the objectives or content of the task.

After this practice session, participants immediately completed a second session with a new list of seven items, also lasting 168 trials. They first completed 3 blocks (36 trials) of only the adjacent pairs, then completed 3 blocks (90 trials) of only the non-adjacent pairs. Finally, they completed a single block (42 trials) of all pairs.

ACKNOWLEDGMENTS

The authors with to thank Drew Altschul and Jessica Joiner for assistance training the monkeys and their administrative support, Benjamin Kennedy for his assistance collecting monkey data, and Sheila Krishnan for assistance in collecting the human data. This work was supported by US National Institute of Health, grant 5R01MH081153-06 awarded to V.F. and H.T.

AUTHOR CONTRIBUTIONS

G.J., H.S., and V.F., conceived the experiments. F.M. and Y.A. acquired data. G.J. designed the model. G.J. and F.M. performed analyses. G.J., F. M., Y. A., V. F., and H. S. wrote the paper.

REFERENCES

Acuna, B. D., Sanes, J. N., and Donoghue, J. D. (2002). Cognitive mechanisms of transitive inference. *Experimental Brain Research*, 146:1–10.

Behrens, T. E. J., Woolrich, M. W., Walton, M. E., and Rushmore, M. F. S. (2007). Learning the value of information in an uncertain world. *Nature Neuroscience*, 10:1214–1221.

Berridge, K. C. (2007). The debate over dopamine?s role in reward: The case for incentive salience. *Psychopharmacology*, 191:391–431.

Bond, A. B., Wei, C. A., and Kamil, A. C. (2010). Cognitive representation in transitive inference: A comparison of four corvid species. *Behavioural Processes*, 85:283–292.

Bush, R. R. and Mosteller, F. (1951). A mathematical model for simple learning. *Psychological Review*, 58:313–323.

D'Amato, M. R. and Colombo, M. (1990). The symbolic distance effect in monkeys. *Animal Learning and Behavior*, 18:133–140.

Daw, N. D. (2011). Trial-by-trial data analysis using computational models. In Delgado, M. R., hpelps, E. A., and Robbins, T. W., editors, *Decision Making, Affect, and Learning: Attention and Performance XXIII*, pages 3–38. Oxford University Press, Oxford, UK.

Dayan, P. and Berridge, K. C. (2014). Model-based and model-free pavlovian reward learning: Revaluation, revision, and revelation. *Cognitive, Affective, & Behavioral Neuroscience*, 14:473–492.

Doll, B. B., Shohamy, D., and Daw, N. D. (2015). Multiple memory systems as substrates for multiple decision systems. *Neurobiology of Learning and Memory*, 117:4–13.

Doll, B. B., Simon, D. A., and Daw, N. D. (2012). The ubiquity of model-based reinforcement learning. *Current Opinion in Neurobiology*, 22:1075–1081.

Fiorillo, C. D., Tobler, P. N., and Schultz, W. (2003). Discrete cording of reward probability and uncertainty by dopamine neurons. *Science*, 299:1898–1902.

Gazes, R. P., Chee, N. W., and Hampton, R. R. (2012). Cognitive mechanisms for transitive inference performance in rhesus monkeys: Measuring the influence of associative strength and inferred order. *Journal of Experimental Psychology: Animal Behavior Processes*, 38:331–345.

Gläscher, J., Daw, N., Dayan, P., and O'Doherty, J. P. (2008). States versus rewards: Dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron*, 66:585–595.

Green, C. D. (2001). Scientific models, connectionist networks, and cognitive science. *Theory & Psychology*, 11:97–117.

Grosenick, L., Clement, T. S., and Fernald, R. D. (2007). Fish can infer social rank by observation alone. *Nature*, 445:429–432.

Jensen, G., Altschul, D., Danly, E., and Terrace, H. S. (2013). Transfer of a serial representation between two distinct tasks by rhesus macaques. *PLOS ONE*, 8:e70285.

Judge, S. J., Richmond, B. J., and Chu, F. C. (1980). Implantation of magnetic search coils for measurement of eye position: An improved method. *Vision Research*, 125:535–538.

Kass, R. E. and Wasserman, L. (1995). A reference bayesian test for nested hypotheses and its relationship to the schwarz criterion. *Journal of the American Statistical Association*, 90:928–934.

Lazareva, O. F. and Wasserman, E. A. (2012). Transitive inference in pigeons: Measuring the associative values of stimuli b and d. *Behavioural Processes*, 89:244–255.

Luce, D. (1959). *Individual Choice Behavior*. Wiley, New York, NY.

MacLean, E. L., Merritt, D. J., and Brannon, E. M. (2008). Social complexity predicts transitive reasoning in prosimian primates. *Animal Behaviour*, 76:479–486.

Maia, T. V. (2009). Reinforcement learning, conditioning, and the brain: Successes and challenges. *Cognitive, Affective, & Behavioral Neuroscience*, 9:343–364.

McDannald, M. A., Lucantonio, F., Burke, K. A., Niv, Y., and Schoenbaum, G. (2009). Ventral striatum and orbitofrontal cortex are both required for model-based, but not model-free, reinforcement learning. *Journal of Neuroscience*, 31:2700–2705.

Merritt, D. J. and Terrace, H. S. (2011). Mechanisms of inferential order judgments in humans (*Homo sapiens*) and rhesus monkeys (*Macaca mulatta*). *Journal of Comparative Psychology*, 125:227–238.

Piaget, J. (1921). Une forme verbale de la comparison chez l'enfant. *Archive de Psychologie*, 18:141–172.

Raineri, E., Dabad, M., and Heath, S. (2014). A note on exact differences between beta distributions in genomic (methylation) studies. *PLOS ONE*, 9:e97349.

Robinson, D. A. (1963). A method of measuring eye movement using a scleral search coil in a magnetic field. *IEEE Transactions on Biomedical Electronics*, 10:137–145.

Schultz, W., Dayan, P., and Montague, P. R. (1997). A neural substrate of prediction and reward. *Science*, 275:1593–1599.

Siegel, S. and Allan, L. G. (1996). The widespread influence of the rescorla-wagner model. *Psychonomic Bulletin & Review*, 3:314–321.

Siemann, M. and Delius, J. D. (1996). Algebraic learning and neural network models for transitive and non-transitive responding. *European Journal of Cognitive Psychology*, 10:307–334.

Takahashi, M., Ushitani, T., and Fujita, K. (2008). Inference based on transitive relation in three shrews (*Tupaia belangeri*) and rats (*Ratus norvegicus*) on a spatial discrimination task. *Psychological Record*, 58:215–227.

Teichert, T., Yu, D., and Ferrera, V. P. (2014). Performance monitoring in monkey frontal eye field. *Journal of Neuroscience*, 34:1657–1671.

Thrun, S. B. (1992). The role of exploration in learning control. In White, D. A. and Sofge, D. A., editors, *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, pages 527–559. Van Nostrand Reinhold, Florence, KY.

Usher, M. and McClelland, J. L. (2001). The time course of perceptual choice: The leaky, competing accumulator model. *Psychological Review*, 108:550–592.

Vasconcelos, M. (2008). Transitive inference in non-human animals: An empirical and theoretical analysis. *Behavioural Processes*, 78:313–334.

von Fersen, L., Wynne, C. D. L., and Delius, J. D. (1991). Transitive inference formation in pigeons. *Journal of Experimental Psychology: Animal Behavior Processes*, 17:334–341.

Waelti, P., Dickinson, A., and Schultz, W. (2001). Dopamine responses comply with basic assumptions of formal learning theory. *Nature*, 412:43–48.

Ward, R. D., Gallistel, C. R., and Balsam, P. D. (2013). It?s the information! *Behavioural Processes*, 95:3–7.

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. PhD thesis, University of Cambridge.

Watkins, C. J. C. H. and Dayan, P. (1989). Q-learning. *Machine Learning*, 8:279–292.

Woolf, B. (1957). The log likelihood ratio test (the G-test). *Annals of Human Genetics*, 21:397–409.

Wynne, C. D. L. (1995). Reinforcement accounts for transitive inference performance. *Animal Learning & Behavior*, 23:207–217.