Reinforcement Learning constrained State Space Modeling of Decisions

Yusi Chen

August 19, 2025

1 ONLY decision modeling

1.1 Forward Q learning

From Bari et al. [1], we could write down the classical forward Q learning updates with forgetting for each trial k:

$$\begin{aligned} Q^l_{k+1} &= C^l_k[\zeta Q^l_k + \alpha(R_k - Q^l_k)] + (1 - C^l_k)(\zeta Q^l_k) \\ Q^r_{k+1} &= C^r_k[\zeta Q^r_k + \alpha(R_k - Q^r_k)] + (1 - C^r_k)(\zeta Q^r_k) \\ P(C^l_k = 1) &= \text{Sigmoid}(Q^l_k - Q^r_k) \end{aligned} \tag{1}$$

where C^l is binary and indicates whether chooses left or not. Sigmoid function has a β parameter determining the slope. Note that for most cases, the update of Q happens for each trial. We could also write down similar update equations for each time step:

$$\begin{aligned} Q^l_{t_2+1} &= C^l_{t_2}[\zeta Q^l_{t_2} + \alpha(R_{t_2} - Q^l_{t_2})] + (1 - C^l_{t_2})(\zeta Q^l_{t_2}) \\ Q^r_{t_2+1} &= C^r_{t_2}[\zeta Q^r_{t_2} + \alpha(R_{t_2} - Q^r_{t_2})] + (1 - C^r_{t_2})(\zeta Q^r_{t_2}) \\ P(C^l_{t_1} = 1) &= \text{Sigmoid}(Q^l_{t_1} - Q^r_{t_1})g(t_1) \end{aligned} \tag{2}$$

where $g(t_1)$ is a causal filter after a cue happened at t_0. In this set of equations, t_1 and t_2 don't need to be close to each other and Q gets updated at $t_1 + 1$ where $C_{t_1} = 1$. Otherwise, Q is decaying. Q_{t+1} is linear in Q_t given C_t and R_t. So let's abbreviate the update equations for Q:

$$Q_{t_2+1} = A(C_{t_2})Q_{t_2} + B(C_{t_2}, R_{t_2}) \tag{3}$$

In summary, we got a Markovian computation graph shown in Fig. 1.1.

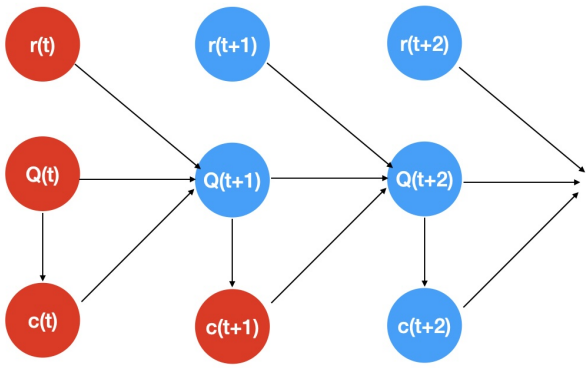

Figure 1: Computation graph at $t+1$. Read nodes are information available for inference of Q_{t+1}

1.2 Q learning constrained state space modeling (SSM)

Based on dynamics provided in Eq. 2, we could write down a hidden markov (HMM) chain with Q as latent variables, R as input and C as observations. **The subtle difference from classical HMM is that C_t is used to determine the dynamics of Q_{t+1}.** The formal problem formulation is: given reward $R_{0:t}$, choices $C^l_{0:t+1}$ and $C^r_{0:t+1}$, infer latent Q^l_{t+1} and Q^r_{t+1} such that $P(Q_{t+1}|R_{0:t}, C_{0:t+1})$ is maximized.

Based on Bayes rule and Markov assumptions, we could write down this posterior as:

$$\begin{aligned}
&P(Q_{t+1}|R_{0:t}, C_{0:t+1}) \\
&= \frac{1}{\eta} P(Q_{t+1}, C_{t+1}|R_{0:t}, C_{0:t}) \\
&= \frac{1}{\eta} P(C_{t+1}|Q_{t+1}, R_{0:t}, C_{0:t}) P(Q_{t+1}|R_{0:t}, C_{0:t}) \\
&= \frac{1}{\eta} P(C_{t+1}|Q_{t+1}, R_{0:t}, C_{0:t}) \int P(Q_{t+1}|Q_t, R_{0:t}, C_{0:t}) P(Q_t|R_{0:t}, C_{0:t}) dQ_t \\
&= \frac{1}{\eta} P_h(C_{t+1}|Q_{t+1}) \int P_f(Q_{t+1}|Q_t, C_t, R_t) P(Q_t|R_{0:t-1}, C_{0:t}) dQ_t
\end{aligned} \tag{4}$$

This is a recursive evaluation so we only need to worry about the observation model P_h and the motion model P_f.

For the observation model P_h:

$$P_h(C^l_{t+1}|Q_{t+1}) \sim Bernoulli(\text{Sigmoid}(Q^T_{t+1} D) g_t) \tag{5}$$

If we denote $D = [1, -1]^T$ and drop g_t for now, then

$$P(C^l_{t+1}|Q_{t+1}) = C^l_{t+1}[1 - \text{Sigmoid}(Q^T_{t+1} D)] + (1 - C^l_{t+1})\text{Sigmoid}(Q^T_{t+1} D) \tag{6}$$

For the motion model P_f, let's add uncorrelated Gaussian noise with diagonal variance V to Eq.3. We could have a probabilistic formulation:

$$P_f(Q_{t+1}|Q_t, C_t, R_t) \sim Normal(A(C_t)Q_t + B(C_t, R_t), V) \tag{7}$$

The closed form expression of posterior might be very messy as Gaussian and Bernoulli are not closed under multiplications. We may need sampling methods to approximate these distributions. The upshot is that we could calculate the Q posterior recursively.

1.3 Joint inference of latent and parameters

So in this specfic Q learning paradigm, we have latent variable Q and parameters ζ, α, β left for inference from data. The steps are summarized as below:

- For fixed values of ζ, α, β, the posterior distribution of Q_{t+1} can be found via the Bayesian filtering through the recursive calculation of Eq.4 **(Forward filtering)**.
- For fixed values of ζ, α, β, the marginal likelihood of the observed data C_{t+1} can be efficiently computed from Eq.4 by integrating out the Q_{t+1}. **(Data likelihood computation)**
- For fixed posterior over Q_{t+1}, we can infer ζ, α, β that maximize the likelihood of observing C_{t+1}. **(MLE parameter estimation)**

1.4 Inference of Latents

1.4.1 Forward filtering

Denote $\theta = \{\alpha, \beta, \zeta\}$, forward filtering computes the probabilities of latent states given information up to current time t as detailed in Eq.4.

$$P(Q_t|R_{0:t-1}, C_{0:t}, \theta) \propto P_h(C_t|Q_t) \int P_f(Q_t|Q_{t-1}, C_{t-1}, R_{t-1}) P(Q_{t-1}|R_{0:t-2}, C_{0:t-1}) dQ_{t-1} \tag{8}$$

1.4.2 Backward smoothing?

Backward smoothing compute latent state probability in an offline manner given all available information up till T, i.e. $P(Q_t|R_{0:T}, C_{0:T}, \theta)$. Calculation could be derived in by induction. Assume we know $A_{t+1} = P(Q_{t+1}|R_{0:T}, C_{0:T}, \theta)$, we want to compute $A_t = P(Q_t|R_{0:T}, C_{0:T}, \theta)$.

$$\begin{aligned} A_t &= \int P(Q_t|Q_{t+1}, R_{0:T}, C_{0:T}) A_{t+1} dQ_{t+1} \\ &= \int P(Q_t|Q_{t+1}, R_{0:t}, C_{0:t+1}) A_{t+1} dQ_{t+1} \\ &= \int \frac{P(Q_t, Q_{t+1}|R_{0:t}, C_{0:t+1})}{P(Q_{t+1}|R_{0:t}, C_{0:t+1})} A_{t+1} dQ_{t+1} \end{aligned} \tag{9}$$

The second line uses the Markovian assumption. The make it more obvious, one could reverse $P(Q_t|Q_{t+1})$ using the Bayes rule and check the conditional dependence of the three involved terms one by one.

We could compute $P(Q_{t+1}|R_{0:t}, C_{0:t+1})$ and $P(Q_t, Q_{t+1}|R_{0:t}, C_{0:t+1})$ using the forward filtering process. Then induction could be performed to backward smooth the probability.

1.4.3 General Gaussian filtering

1.5 Parameter Estimation

There are multiple different algorithms to infer the set of parameters that maximize the likelihood of observing the data. In the special scenario of making decisions, assuming independent observations per time step, data likelihood could be expressed as product of multiple Poisson or Bernoulli process. Most algorithms would suffer from the problem of being stuck in local minimum because the convexity of the loss function is hard to prove (?). Maybe in some special functional, we could assume convexity.

1.5.1 Stochastic Gradient Descent (SGD) of a Loss Function

.

1.5.2 Expectation Maximization (EM)

1.5.3 State space identification (SSID) initialized EM

SSID for linear dynamical systems (LDS): Ho-Kalman method [2].
Generative model:

$$\begin{aligned} x_1 &\sim \mathcal{N}(0, \Pi) \\ x_{t+1}|x_t &\sim \mathcal{N}(Ax_t, Q) \\ z_t &= Cx_t + d \\ y_t|z_t &\sim \mathcal{N}(z_t, R) \end{aligned} \tag{10}$$

Assume stationary, $\Pi = \lim_{t\to\infty} Cov[x_t]$. Define a 'future-past Hankel matrix' H of observations as:

$$H := \text{Cov}[y_t^+, y_t^-], \quad y_t^+ := \begin{pmatrix} y_t \\ \vdots \\ y_{t+k-1} \end{pmatrix}, \quad y_t^- := \begin{pmatrix} y_{t-1} \\ \vdots \\ y_{t-k} \end{pmatrix} \tag{11}$$

Under stationary assumptions, the Hankel matrix could be expressed as:

$$H = \begin{pmatrix} C^\top & (CA)^\top & \cdots & (CA^{k-1})^\top \end{pmatrix}^\top \cdot \begin{pmatrix} A\Pi C^\top & \cdots & A^k \Pi C^\top \end{pmatrix} \tag{12}$$

Performing SVD on the Hankel matrix, we could estimate values for A and C.

SSID for general LDS [3].

- For general LDS, the Gaussian observation model is replaced by more general observation models such as a Bernoulli or Poisson observation. Each observation entry is independent conditioned on z_t.
- Assume the moments of z_t and y_t are linked as: $\mathbb{E}(y_t|z_t) = f(z_t)$ and $V(y_t|z_t) = g(z_t)$. The basic is that we could apply SSID to z_t, which has normal distribution and link y to z though their moments.
- The key is to compute future-past Hankel matrix of z from observations of y.

SSID for Bernoulli LDS.

1.6 Relationship to other models

1.6.1 Drift Diffusion models (DDM)and RL-DDM

Main reference: mathematical analysis of DDM [4] and Reinforcement learning DDM [5].

In this basic RLDD model, the non-decision time T_{er}, starting point z, and boundary separation a are trial independent free parameters, as in the ordinary DDM. The drift rate $v(t)$ varies from trial to trial as a function of the difference in the expected rewards, multiplied by a scaling parameter m, which can capture differences in the ability to use knowledge of the reward probabilities:

$$v(t) = [V_{upper}(t) - V_{lower}(t)] \times m \tag{13}$$

1.6.2 Poisson GLM for spike trains

Main reference: Poisson GLM used by Pillow et al. [6] and likelihood of point process [7].

Probability description of exact event times in a point process. Let $N(t)$ represents the number of events up until t and their spike time is described by $s(t) = \{s_0, s_1, .., s_{N(t)}\}$, the probability density of the spike train during $[0, T]$ is given by combining independent Poisson process with infinitesimal time intervals and inhomogeneous rate λ:

$$p(s_1, s_2, ..., s_n) = \prod_{k=1}^{n} \lambda(s_k|s_1, .., s_{k-1}) \exp - \int_0^T \lambda(u|s(u))du \tag{14}$$

where $N(T) = n$ and

$$\lambda(t|s(t)) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} P(N(t + \Delta t) - N(t) = 1|s(t)) \tag{15}$$

The general form of event rate given above is dependent on the entire event history from the beginning. A tractable simplification is to only consider the influence of the most recent event s_*. It could take the form of:

$$\lambda(t|s(t)) = \lambda(t, t - s_*(t)) \tag{16}$$

- If $\lambda(t, t - s_*(t)) = \lambda(t)$, this is inhomogenous Poisson process;
- If $\lambda(t, t - s_*(t)) = \lambda(t - s_*(t))$, this is a renewal process where inter-event intervals follow the same probability distribution;
- If $\lambda(t, t - s_*(t)) = \lambda_1(t)\lambda_2(t - s_*(t))$, we could use this to model refractory period. This is the basis for discarding time points in refractory period.

Poisson GLM. Conditioned intensity (i.e. spike rate) is given by $\lambda(t|s(t)) = \exp(k * x + h * s + \mu)$ where x is stimulus and k, h are filters to be estimated. Temporal filter could be consisted of a basis of raised cosine bumps of the form $b_j(t) = 0.5\cos(a\log[t + c] - \phi_j) + 0.5$.

1.6.3 Loss function for Poisson RL.

In the simple case of stop task with Q learning, the loss function could be written as a parameter of α and ζ given that we know the lick choice and rewards.

In the discrete time case, lick time up to T was denoted by $s(T) = \{s_1, s_2, .., s_N\}$, assume that:

$$\begin{aligned} Q_{t+1} &= \zeta Q_t + \alpha C_t(R_t - Q_t) \\ C_t &= \sum_{i=1}^{N} \delta_{s_i}(t) \end{aligned} \tag{17}$$

The loss function, similar to Eq. 14, could be written as:

$$L = \sum_{i=1}^{N} \log Q(s_i) - \sum_{t=1}^{T} Q_t \tag{18}$$

Value of Q could be written in a recursive way.

$$\begin{aligned} Q_0 &= 1 \\ Q_{s_k+1} &= Q_{s_k}(\zeta - \alpha) + \alpha R_{s_k} \\ Q_{s_k+t} &= Q_{s_k+1}\zeta^{t-1}, t \leq s_{k+1} - s_k \end{aligned} \tag{19}$$

If we plug in the expression into the loss function, we could obtain L parameterized by α and ζ. With simplifications on the expression of R_t, we might be able to obtain a closed form solution.

1.7 Case study: when to stop

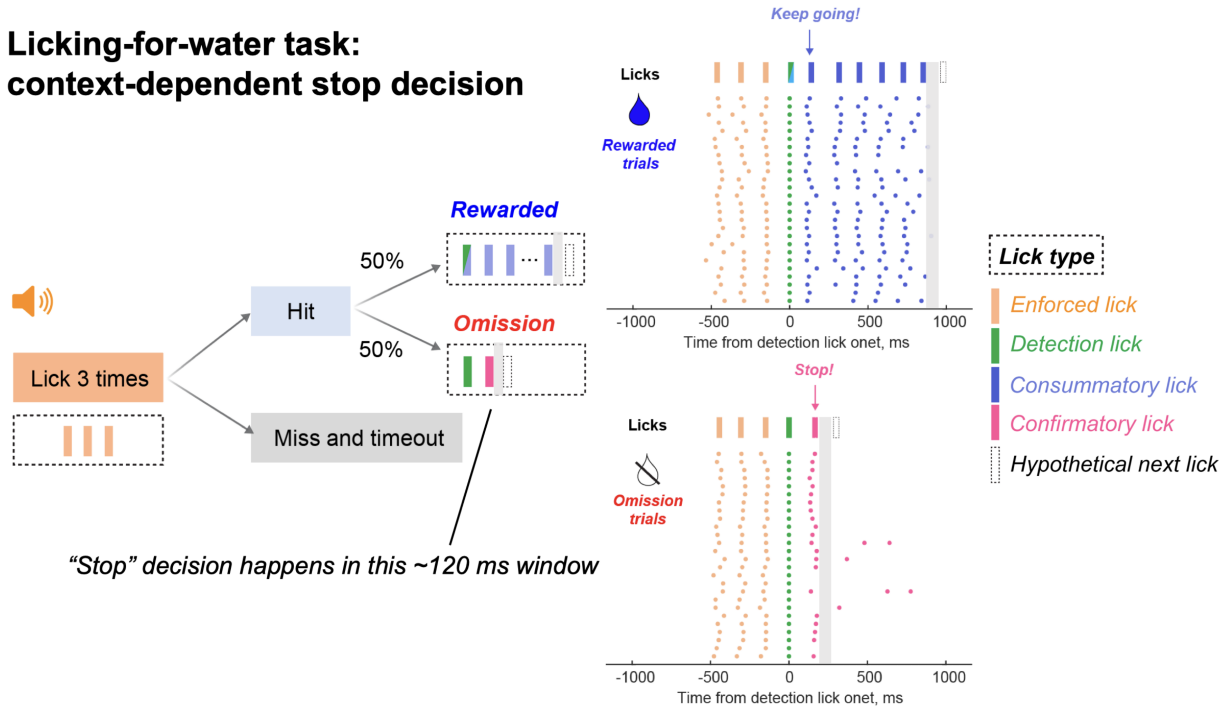

Figure 2: When to stop task setup. Credit to Dr. Shijia Liu (Sabatini lab).

Given cue g_t, reward R_t, binary lick decision C_t, there're two different ways to estimate animal's belief of licking Q_t: 1) Forward value iteration given a specific RL model (e.g. TD, choice kernel etc [8]). *The important thing is that the chosen RL model needs to be applied within each trial*; 2) Simultaneous latent variable and model parameter inference in constrained SSM. The forward approach is simpler but results depend on model choice and hyper-parameter choice while the second approach allows model comparison and hyper-parameter estimation.

1.7.1 Forward Q learning

The continuous Q update in this context could be written as (dropping the influence of g for simplicity):

$$\begin{aligned} Q_{t_2+1} &= C_{t_2}[\zeta Q_{t_2} + \alpha(R_{t_2} - Q_{t_2})] + (1 - C_{t_2})(\zeta Q_{t_2}) \\ P(C_{t_1} = 1) &= \text{Sigmoid}(\beta Q_{t_1}) \end{aligned} \tag{20}$$

Q is initialized to be big due to enforced licking. If the fourth detection lick is rewarded, reward prediction error (RPE) is small, Q remain unchanged, except for slow decaying. If the detection lick is not rewarded, $R - Q$ is very negative, Q value will be drastically decreased at the next time step.

1.7.2 Constrained SSM

Input includes cue g_t, reward R_t; Observation includes the binary lick choice C_t; Latent variable Q_t represents the subjective value of licking. System dynamics could be described as below:

$$\begin{aligned} Q_{t+1}|C_t, Q_t, R_t &\sim Normal(A(C_t)Q_t + B(C_t, R_t), V) \\ C_t|Q_t &\sim Bernoulli(\text{Sigmoid}(\beta Q_t)) \end{aligned} \tag{21}$$

1.7.3 With refractory period

Since it is not possible to lick at each time step, an artificial refractory period (τ) is enforced in the generation of licks. R_t is generated from C_t and is supplied to the algorithm as another control signal.

$$\begin{aligned} C_t|Q_t &\sim Bernoulli(\text{Sigmoid}(\beta Q_t)R_t) \\ R_t &= 1 - sgn(\sum_{i=t-\tau}^{t-1} C_i) \end{aligned} \tag{22}$$

1.8 Case study: dynamic foraging

2 References

1. Bari, B. A. *et al.* Stable representations of decision variables for flexible behavior. *Neuron* **103,** 922–933 (2019).
2. Ho, B. *et al.* Effective construction of linear state-variable models from input/output functions: Die konstruktion von linearen modeilen in der darstellung durch zustandsvariable aus den beziehungen für ein-und ausgangsgrößen. *at-Automatisierungstechnik* **14,** 545–548 (1966).
3. Buesing, L. *et al.* Spectral learning of linear dynamics from generalised-linear observations with application to neural population data. *Advances in neural information processing systems* **25** (2012).
4. Bogacz, R. *et al.* The physics of optimal decision making: a formal analysis of models of performance in two-alternative forced-choice tasks. *Psychological review* **113,** 700 (2006).
5. Pedersen, M. L. *et al.* The drift diffusion model as the choice rule in reinforcement learning. *Psychonomic bulletin & review* **24,** 1234–1251 (2017).
6. Pillow, J. W. *et al.* Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature* **454,** 995–999 (2008).
7. Kass, R. E. *et al.* A spike-train probability model. *Neural computation* **13,** 1713–1720 (2001).
8. Wilson, R. C. *et al.* Ten simple rules for the computational modeling of behavioral data. *Elife* **8,** e49547 (2019).