A Approach Details

A.1 Information Gain Derivation

Information gain can be computed by calculating the change in entropy in a distribution X after receiving the datapoint y:

$$
\begin{aligned}
&\mathrm{IG}(X, y) && (14)\\
&= H(X) - H(X|y) && (15)\\
&= -\mathbb{E}_{x|X} \log P(x) + \mathbb{E}_{x|X,y} \log P(x|y) && (16)\\
&= \sum_{x \in X} P(x|y) \cdot \log P(x|y) - \sum_{x \in X} P(x) \cdot \log P(x) && (17)
\end{aligned}
$$

We adapt this generic formulation to use our definitions of interaction types, query space, and choice space, and aim to solve for the optimal query:

$$
\max_{q \in Q} \mathbb{E}_{c|C_i(q)} \left[\mathrm{IG}(\mathcal{W}, c) \right] \tag{18}
$$

We now solve for the expected information gain according to Eqs. 14- 17 and following the derivation presented in [3]:

$$
\begin{aligned}
&\mathbb{E}_{c|C_i(q)} \left[\mathrm{IG}(\mathcal{W}, c) \right] && (19)\\
&= H(\mathcal{W}) - \mathbb{E}_{c|C_i(q)} \left[H(\mathcal{W}|c) \right] && (20)\\
&= -\mathbb{E}_{\mathcal{W}} \left[\log P(\mathcal{W}) \right] + \mathbb{E}_{\mathcal{W}, c|C_i(q)} \left[\log P(\mathcal{W}|c) \right] && (21)\\
&= \mathbb{E}_{\mathcal{W}, c|C_i(q)} \left[\log P(\mathcal{W}|c) - \log P(\mathcal{W}) \right] \qquad \text{(see proof in Sec. A.1.1)} && (22)\\
&= \mathbb{E}_{\mathcal{W}, c|C_i(q)} \left[\log \frac{P(\mathcal{W}|c)}{P(\mathcal{W})} \right] && (23)\\
&= \mathbb{E}_{\mathcal{W}, c|C_i(q)} \left[\log \frac{P(c|\mathcal{W})}{P(c)} \right] \qquad \text{(by Bayes' rule)} && (24)\\
&= \sum_{c \in C_i(q)} \left[P(c) \sum_{w \in \mathcal{W}} \left[P(w|c) \cdot \log \frac{P(c|w)}{P(c)} \right] \right] && (25)\\
&= \sum_{c \in C_i(q)} \left[P(c) \sum_{w \in \mathcal{W}} \left[\frac{P(w)P(c|w)}{P(c)} \cdot \log \frac{P(c|w)}{P(c)} \right] \right] && (26)\\
&= \sum_{c \in C_i(q)} \sum_{w \in \mathcal{W}} \left[P(w) \cdot P(c|w) \cdot \log \frac{P(c|w)}{P(c)} \right] && (27)\\
&= \sum_{c \in C_i(q)} \sum_{w \in \mathcal{W}} \left[P(w) \cdot P(c|w) \cdot \log \frac{P(c|w)}{\sum_{w' \in \mathcal{W}} P(w') \cdot P(c|w')} \right] && (28)\\
&\approx \frac{1}{M} \sum_{c \in C_i(q)} \sum_{w \in \Omega} \left[P(c|w) \cdot \log \frac{M \cdot P(c|w)}{\sum_{w' \in \Omega} P(c|w')} \right] && (29)
\end{aligned}
$$

Where Ω contains M samples of the distribution $\mathcal{W}$.

A.1.1 Proof of Eq. 22

$$
\begin{aligned}
&-\mathbb{E}_{\mathcal{W}}\left[\log P(\mathcal{W})\right] + \mathbb{E}_{\mathcal{W},c|C_i(q)}\left[\log P(\mathcal{W}|c)\right] && (30)\\
&= \mathbb{E}_{\mathcal{W},c|C_i(q)}\left[\log P(\mathcal{W}|c)\right] - \mathbb{E}_{\mathcal{W}}\left[\log P(\mathcal{W})\right] && (31)\\
&= \left[\sum_{w\in\mathcal{W}} P(w) \sum_{c\in C_i(q)} P(c|w)\cdot \log P(w|c)\right] - \left[\sum_{w\in\mathcal{W}} P(w)\cdot \log P(w)\right] && (32)\\
&= \sum_{w\in\mathcal{W}} P(w)\cdot \left[\left(\sum_{c\in C_i(q)} P(c|w)\cdot \log P(w|c)\right) - \log P(w)\right] && (33)\\
&= \sum_{w\in\mathcal{W}} P(w)\cdot \left[\sum_{c\in C_i(q)} P(c|w)\cdot \log P(w|c) - \sum_{c\in C_i(q)} P(c|w)\cdot \log P(w)\right] && (34)\\
&= \sum_{w\in\mathcal{W}} P(w)\cdot \sum_{c\in C_i(q)} P(c|w)\cdot \left[\log P(w|c) - \log P(w)\right] && (35)\\
&= \mathbb{E}_{\mathcal{W},c|C_i(q)}\left[\log P(\mathcal{W}|c) - \log P(\mathcal{W})\right] && (36)
\end{aligned}
$$

A.2 KL Divergence Formulation

We now show that we can alternatively derive Eq. 29 from the standard KL divergence equation:

$$
\text{KL}(P||Q) = \sum_{x\in X}\left[P(x)\cdot \log\frac{P(x)}{Q(x)}\right] \tag{37}
$$

Where P and Q represent the data distribution before and after receiving feedback, respectively. We convert this formulation to our terminology as follows:

$$
\max_{q\in Q} \mathbb{E}_{c|C_i(q)}\left[\text{KL}(P(\mathcal{W}|c)||P(\mathcal{W}))\right] \tag{38}
$$

We now solve for the optimal query:

$$
\begin{aligned}
&\mathbb{E}_{c|C_i(q)}\left[\text{KL}(P(\mathcal{W}|c)||P(\mathcal{W}))\right] && (39)\\
&= \mathbb{E}_{c|C_i(q)}\left[\sum_{w\in\mathcal{W}}\left[P(w|c)\cdot \log\frac{P(w|c)}{P(w)}\right]\right] && (40)\\
&= \mathbb{E}_{c|C_i(q)}\left[\sum_{w\in\mathcal{W}}\left[P(w|c)\cdot \log\frac{P(c|w)}{P(c)}\right]\right] && (41)\\
&= \sum_{c\in C_i(q)}\left[P(c)\sum_{w\in\mathcal{W}}\left[P(w|c)\cdot \log\frac{P(c|w)}{P(c)}\right]\right] && (42)
\end{aligned}
$$

Which is equivalent to Eq. 25, and thus results in Eq. 29.

A.3 Probability Tensor Derivations

See Table 1 for all definitions of q, c, c^+, c^- for each interaction type. In the demonstration case, we define $\mathbf{P}$ as follows:

$$\begin{aligned}
\mathbf{P}^{(\text{demo})}_{q,c,\omega} &= \frac{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}} && (43)\\
&= \frac{e^{\beta\cdot\phi(c_0^+)\cdot\omega}}{\sum_{t\in T} e^{\beta\cdot\phi(t)\cdot\omega}} \qquad \text{(since } |c^+| = 1 \text{ and } c^+ \cup c^- = T \text{ for demonstrations)} && (44)\\
&= \frac{\mathbf{E^T}_{0,c_0^+,\omega}}{\sum_{t\in T} \mathbf{E^T}_{0,t,\omega}} && (45)\\
&= \left[\mathbf{E^T}_0 \oslash \sum_{t\in T} \mathbf{E}_t\right]_{c,\omega} \qquad \text{(since there is a 1-1 correlation between } c \text{ and } c^+ \text{ in demos)} && (46)
\end{aligned}$$

where $\oslash$ represents an element-wise division of two matrices (i.e., $(\mathbf{A}\oslash\mathbf{B})_{ij} = \mathbf{A}_{ij}/\mathbf{B}_{ij}$).

In the preference case:

$$\begin{aligned}
\mathbf{P}^{(\text{pref})}_{q,c,\omega} &= \frac{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}} && (47)\\
&= \frac{e^{\beta\cdot\phi(c_0^+)\cdot\omega}}{e^{\beta\cdot\phi(q_0)\cdot\omega} + e^{\beta\cdot\phi(q_1)\cdot\omega}} \qquad \text{(since } |c^+| = 1 \text{ and } c^- = q \setminus c^+ \text{ in preferences)} && (48)\\
&\text{Since } c_0 \implies c^+ = \{q_0\} \text{ and } c_1 \implies c^+ = \{q_1\}\text{:} && (49)\\
&= \left[\frac{e^{\beta\cdot\phi(q_0)\cdot\omega}}{e^{\beta\cdot\phi(q_0)\cdot\omega} + e^{\beta\cdot\phi(q_1)\cdot\omega}}, \frac{e^{\beta\cdot\phi(q_1)\cdot\omega}}{e^{\beta\cdot\phi(q_0)\cdot\omega} + e^{\beta\cdot\phi(q_1)\cdot\omega}}\right]_c \qquad \text{(where } c \in \{0,1\}\text{)} && (50)\\
&= \left[\frac{\mathbf{E^T}_{q_0,q_1,\omega}}{[\mathbf{E}+\mathbf{E^T}]_{q_0,q_1,\omega}}, \frac{\mathbf{E}_{q_0,q_1,\omega}}{[\mathbf{E}+\mathbf{E^T}]_{q_0,q_1,\omega}}\right]_c && (51)\\
&= \left[\left(\mathbf{E}\oslash(\mathbf{E}+\mathbf{E^T})\right)^{\mathbf{T}}, \mathbf{E}\oslash(\mathbf{E}+\mathbf{E^T})\right]_{c,q_0,q_1,\omega} && (52)
\end{aligned}$$

In the corrections case:

$$\begin{aligned}
\mathbf{P}^{(\text{corr})}_{q,c,\omega} &= \frac{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}} && (53)\\
&= \frac{e^{\beta\cdot\phi(c_0^+)\cdot\omega}}{e^{\beta\cdot\phi(c_0^+)\cdot\omega} + e^{\beta\cdot\phi(c_0^-)\cdot\omega}} \qquad \text{(since } |c^+| = 1 \text{ and } |c^-| = 1\text{)} && (54)\\
&= \frac{\mathbf{E^T}_{q,c,\omega}}{[\mathbf{E}+\mathbf{E^T}]_{q,c,\omega}} \qquad \text{(due to 1-1 correlation between } q \text{ and } c^- \text{ and between } c \text{ and } c^+ \text{ in corrections)} && (55)\\
&= \left[\mathbf{E^T}\oslash(\mathbf{E}+\mathbf{E^T})\right]_{q,c,\omega} && (56)
\end{aligned}$$

In the binary reward case, we compare the likelihood of the teacher demonstrating q to the average likelihood of demonstrating any other trajectory in T:

$$\mathbf{P}^{\text{(bnry)}}_{q,c,\omega} = \frac{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}} \tag{57}$$

$$\text{Since } c_0 \implies c^+ = T\setminus q, \quad c^- = q \tag{58}$$

$$\text{and } c_1 \implies c^+ = q, \quad c^- = T\setminus q\text{:} \tag{59}$$

$$= \frac{1}{\alpha}\left[\frac{1}{|T\setminus q|}\cdot\frac{\sum_{t\in T\setminus q} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in T} e^{\beta\cdot\phi(t)\cdot\omega}}, \frac{e^{\beta\cdot\phi(q)\cdot\omega}}{\sum_{t\in T} e^{\beta\cdot\phi(t)\cdot\omega}}\right]_c \qquad \text{(since } c\in\{0,1\} \text{ in binary rewards)} \tag{60}$$

$$= \left[\frac{1-\mathbf{P}^{\text{(demo)}}_{0,q,\omega}}{\alpha\left(|T|-1\right)}, \frac{\mathbf{P}^{\text{(demo)}}_{0,q,\omega}}{\alpha}\right]_c \qquad \text{(where } \alpha \text{ is a normalization factor s.t. } \sum_c \mathbf{P}^{\text{(bnry)}}_{q,c,\omega} = 1\text{)} \tag{61}$$

$$= \left[1-\frac{\mathbf{P}^{\text{(demo)}}_{0,q,\omega}}{\alpha}, \frac{\mathbf{P}^{\text{(demo)}}_{0,q,\omega}}{\alpha}\right]_c \qquad \text{(since } \sum_c \mathbf{P}^{\text{(bnry)}}_{q,c,\omega} = 1\text{)} \tag{62}$$

$$= \left[1-\left(\mathbf{E^T}_0 \oslash \alpha\sum_{t\in T}\mathbf{E}_t\right), \mathbf{E^T}_0 \oslash \alpha\sum_{t\in T}\mathbf{E}_t\right]_{c,q,\omega} \tag{63}$$

where $\alpha = \frac{1-\mathbf{P}^{\text{(demo)}}_{0,q,\omega}}{|T|-1} + \mathbf{P}^{\text{(demo)}}_{0,q,\omega}$

A.4 Gradient Derivation

Our goal is to update the weight estimate such that it maximizes the likelihood of all feedback in $\mathbf{F}$:

$$\omega^* = \arg\max_{\omega} \prod_{c\in\mathbf{F}} P(c|\omega) \tag{64}$$

$$= \arg\max_{\omega} \prod_{c\in\mathbf{F}} \frac{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}} \tag{65}$$

We calculate the gradient over ω by differentiating over its log-likelihood given $\mathbf{F}$:

$$\ell(\omega) = \log\prod_{c\in\mathbf{F}} \frac{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}} \tag{66}$$

$$= \sum_{c\in\mathbf{F}}\left[\log\frac{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}}\right] \tag{67}$$

$$= \sum_{c\in\mathbf{F}}\left[\log\left(\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}\right) - \log\left(\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}\right)\right] \tag{68}$$

$$\frac{\partial\ell(\omega)}{\partial\omega_j} = \sum_{c\in\mathbf{F}}\left[\frac{\sum_{t\in c^+}\beta\cdot\phi_j(t)\cdot e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+} e^{\beta\cdot\phi(t)\cdot\omega}} - \frac{\sum_{t\in c^+\cup c^-}\beta\cdot\phi_j(t)\cdot e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}}\right] \tag{69}$$

Note that when c^+ contains a single trajectory (i.e., in all interaction types except for binary reward), this gradient simplifies to:

$$\frac{\partial\ell(\omega)}{\partial\omega_j} = \sum_{c\in\mathbf{F}}\left[\beta\cdot\phi_j(c^+_0) - \frac{\sum_{t\in c^+\cup c^-}\beta\cdot\phi_j(t)\cdot e^{\beta\cdot\phi(t)\cdot\omega}}{\sum_{t\in c^+\cup c^-} e^{\beta\cdot\phi(t)\cdot\omega}}\right] \tag{70}$$

A.5 Training Parameters

We enforce $\forall \omega \in \mathcal{W},\ ||\omega|| = 1$. We set a high convergence threshold (10^{-3}) when updating each weight sample in order to maintain sparsity within Ω (which becomes less sparse as $\mathbf{F}$ grows with more queries), and then fully converge (convergence threshold of 10^{-6}) for reporting the distance between ω^* and the weight estimate $\tilde{\omega}$ after each query. During gradient descent, we use a step size of 5x10^{-4} for all tasks except for the Pizza domain, where we use a step size of 10^{-4}.

B Evaluation Details

B.1 Domain Implementations

Domain #1: Parameter Estimation This task involves directly estimating a randomly-initialized, ground truth weight vector ω^* containing 8 parameters. This formulation represents a generic learning problem relevant to many robotics tasks, such as learning the relative importance between task outcomes according to a user's preference. There is no "state" in this domain, and each "trajectory" consists of a single sample of the weight vector. As a result, we do not enable demonstration queries in this domain since the resulting feedback would be akin to directly providing ω^* to the algorithm. The feature representation ϕ of a sample returns the sample itself. Since $||\omega|| = ||\omega^*|| = 1$, the reward of any sampled weight vector directly reflects the cosine similarity between it and the ground truth vector ($r(\omega) = \omega \cdot \omega^* = \cos(\theta)$).

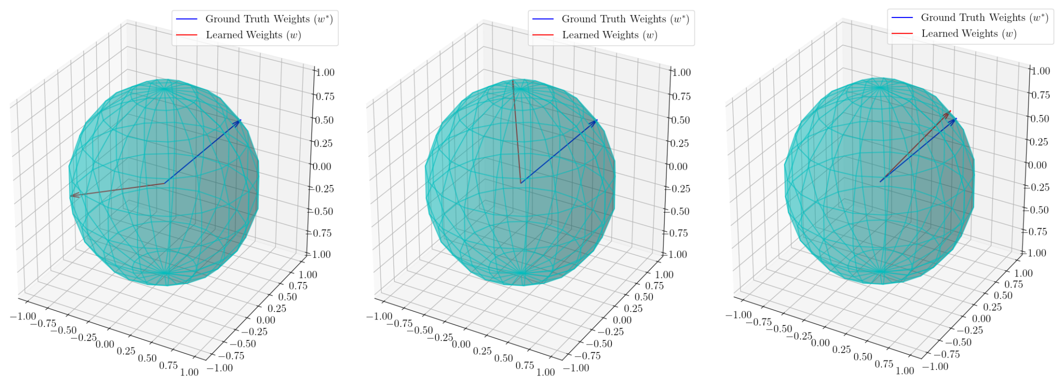


Figure 4: In the Parameter Estimation domain, the robot is tasked with estimating a high-dimensional ground truth weight vector w^* with its own set of learned weights w. To visualize this concept, a simpler case is illustrated above in three dimensions. All weight vectors (ground truth and learned weights) are unit vectors, and therefore lie on a unit sphere. Over time, the robot updates w by interacting with a teacher to gain a better estimate of w^*.

Domain #2: Linear Dynamical System We consider a simple Linear Dynamical System representing a robot that optimizes its controls according to a learned task objective. We represent the dynamics of the robot's state $\mathbf{s}$ as $d\mathbf{s}/dt = \mathbf{A}\mathbf{s}(t) + \mathbf{B}\mathbf{u}(t)$ by using dynamics matrix $\mathbf{A}$, input matrix $\mathbf{B}$, and random controls $\mathbf{u}$. An optimal control vector is one that results in a trajectory of states maximizing $\frac{1}{|T|}\sum_{s\in T}\phi(\mathbf{s})\cdot\omega^*$. We define the feature representation $\phi(\mathbf{s})$ of a state $\mathbf{s}$ as the concatenation of the element-wise, absolute difference between the robot's pose at time t and the goal pose, and the controls $\mathbf{u}(t)$. We experiment with an 8-dimensional feature-space (4 pose elements and 4 corresponding controls).

In a demonstration query, the oracle provides a trajectory (produced by simulating a series of controls) from the initial state that maximizes the total reward. In a preference query, the algorithm proposes two trajectories and the oracle selects the option which yields higher reward. In a corrections query, the algorithm proposes a trajectory and the oracle returns a trajectory that maximizes the reward-to-similarity ratio. In a binary reward query, the algorithm proposes a trajectory and the oracle indicates whether that trajectory results in reward that exceeds the agent's internal threshold.

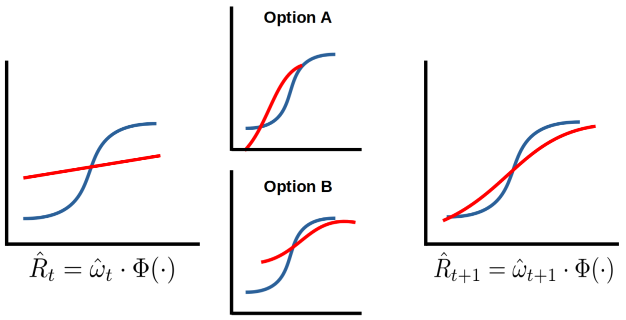


Figure 5: An example of a preference query in the Linear Dynamical System domain. At time $= t$, the learned reward function yields the red "trajectory." After posing a preference query (which consists of options A and B), the corresponding belief update yields the approximated reward function at time $= t + 1$.

Domain #3: Lunar Lander We define a ω^* that results in the agent efficiently moving from its start state to an upright pose on the landing pad. We use the same feature representation as in [9], consisting of four features: the lander's angle, velocity, distance from the landing pad, and final position with respect to the landing pad. We implement each query type in the same manner as in the Linear Dynamical System.

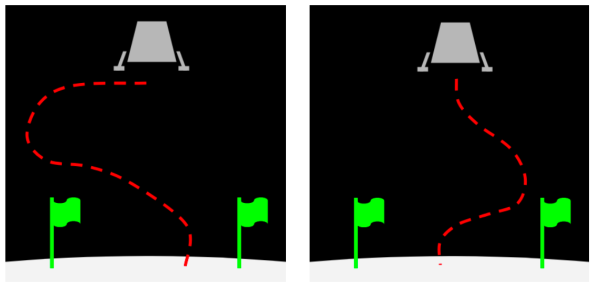


Figure 6: The Lunar Lander domain involves having the robot pilot a lunar lander to safely descend and arrive at a landing pad. The depicted preference query illustrates two different trajectories that may be taken by the lander to reach the destination.

Domain #4: Pizza Arrangement We approximate a preference-learning task in which the robot learns to place toppings on only the left side of a pizza and with uniform spacing between them. We define each "trajectory" as the *next* action the robot should take from the current pizza state; thus, the trajectory is defined as the (x, y)-coordinate of the next topping to be placed. The feature representation consists of four features: the x and y position of the topping, its distance to its nearest-neighboring topping, and the difference between that distance and 4cm.

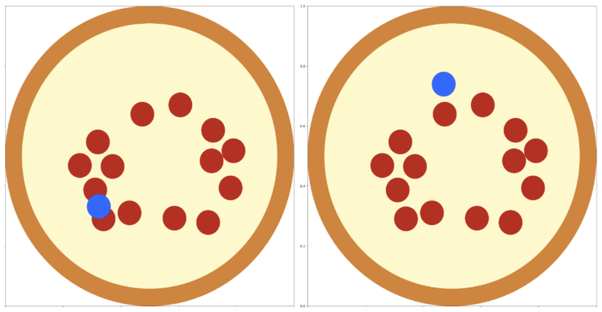


Figure 7: The task in the Pizza Arrangement domain is to learn how to place toppings according to a human's reward over topping positions. In the depicted preference query, a human's choice indicates their preference for the "next" topping's position (choices represented in blue).

B.2 Oracle Implementation

When responding to a query, the oracle requires its own set of trajectory samples. Similar to INQUIRE, we derive this set by uniformly sampling N trajectories; however, the two sample sets are kept separate, and so we distinguish the oracle's trajectory set as T' (resampled for each query state).

Demonstration/Preferences The oracle returns the highest-reward trajectory (according to ω^*) from a uniformly-sampled trajectory set T' (for demonstrations) or from the pair of queried trajectories $C(q)$ (for preferences):

$$\text{Oracle}_{\text{demo}}(q) = \underset{t \in T'}{\arg\max}\,(\phi(t) \cdot \omega^*) \qquad \text{Oracle}_{\text{pref}}(q) = \underset{t \in C(q)}{\arg\max}\,(\phi(t) \cdot \omega^*) \tag{71}$$

Corrections The oracle produces T' by performing rejection sampling; it uniformly samples trajectories and accepts only those with a reward greater than or equal to the queried trajectory q until T' contains N trajectories:

$$\forall t \in T', \phi(t) \cdot \omega^* \geq \phi(q) \cdot \omega^* \tag{72}$$

After producing this trajectory set, the oracle selects the trajectory with the highest ratio of reward-to-distance from the queried trajectory:

$$\text{Oracle}_{\text{corr}}(q) = \underset{t \in T'}{\arg\max}\,\frac{\Delta_r(q,t)}{\Delta_d(q,t)} \tag{73}$$

$$\Delta_r(q,t) = \min_{t' \in T'} \frac{\phi(t) \cdot \omega^* - \phi(q) \cdot \omega^*}{\phi(t') \cdot \omega^* - \phi(q) \cdot \omega^*} \qquad \Delta_d(q,t) = \min_{t' \in T'} \frac{e^{\delta(t,q)}}{e^{\delta(t',q)}} \tag{74}$$

The distance metric δ between two trajectories is domain-specific. In the Parameter Estimation domain, we define this as the angular distance between the two parameter vectors. In the Linear Dynamical System and Lunar Lander domains, we define δ as the normalized distance between the two trajectories' aligned x and y poses over time. We use the DTW-Python package [24] to align trajectories via Dynamic Time Warping and return their normalized distances. In the Pizza Arrangement domain, we define δ as the Euclidean distance between two toppings.

Binary Reward The oracle produces T' by uniformly sampling N trajectories and produces a cumulative distribution R over ground-truth rewards for T'. It then selects a positive or negative reward indicating whether the agent's query q meets or exceeds a threshold percentile α:

$$R = \{\omega^* \cdot \phi(t), \forall t \in T'\} \qquad \text{Oracle}_{\text{bnry}}(q) = \begin{cases} + & R(\omega^* \cdot \phi(q)) \geq \alpha \\ - & \text{otherwise} \end{cases} \tag{75}$$

We set $\alpha = 0.75$ in our experiments.

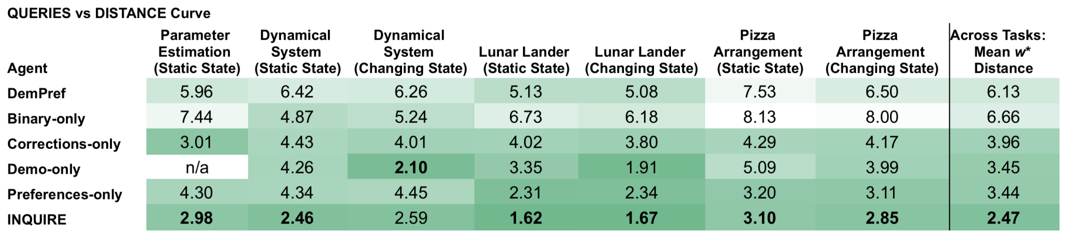
B.3 Evaluation Procedure

Algorithm 3 Evaluation Procedure

```
Input: generate_query and update_weights methods according to algorithm being tested

1:  Generate ground truth reward function $\omega^*$
2:  Generate 10 test states
3:  Compute optimal trajectory $t_{\max}$ for each test case using $\omega^*$
4:  Compute least-optimal trajectory $t_{\min}$ for each test case using $\omega^*$
5:  for each of 10 runs do
6:      Generate 20 query states (if testing in the static condition, repeat the same state 20 times)
7:      for each of 20 queries do
8:          $s \leftarrow$ next query state
9:          $q^* \leftarrow$ generate_query($s, \mathcal{I}, \mathbf{\Omega}$)
10:         $\mathbf{F} \leftarrow \mathbf{F}+$ query_oracle($q^*$)
11:         $\mathbf{\Omega} \leftarrow$ update_weights($\mathbf{F}$)
12:         $\tilde{\omega} \leftarrow$ mean($\Omega$)
13:         Record distance: $\frac{\text{arcccos}(\tilde{\omega} \cdot \omega^*)}{\pi}$
14:         for each of 10 test states do
15:             Compute optimal trajectory $t$ from the test state according to $\tilde{\omega}$
16:             Record performance: $\frac{\phi(t) \cdot \omega^* - \phi(t_{\min}) \cdot \omega^*}{\phi(t_{\max}) \cdot \omega^* - \phi(t_{\min}) \cdot \omega^*}$
```

C AUC Figures



QUERIES vs DISTANCE Curve

Agent	Parameter Estimation (Static State)	Dynamical System (Static State)	Dynamical System (Changing State)	Lunar Lander (Static State)	Lunar Lander (Changing State)	Pizza Arrangement (Static State)	Pizza Arrangement (Changing State)	Across Tasks: Mean *w** Distance
DemPref	5.96	6.42	6.26	5.13	5.08	7.53	6.50	6.13
Binary-only	7.44	4.87	5.24	6.73	6.18	8.13	8.00	6.66
Corrections-only	3.01	4.43	4.01	4.02	3.80	4.29	4.17	3.96
Demo-only	n/a	4.26	**2.10**	3.35	1.91	5.09	3.99	3.45
Preferences-only	4.30	4.34	4.45	2.31	2.34	3.20	3.11	3.44
INQUIRE	**2.98**	**2.46**	2.59	**1.62**	**1.67**	**3.10**	**2.85**	**2.47**

Figure 8: AUC values for the distance plots in Figs 2-3. Darker cells indicate lower (better) values.

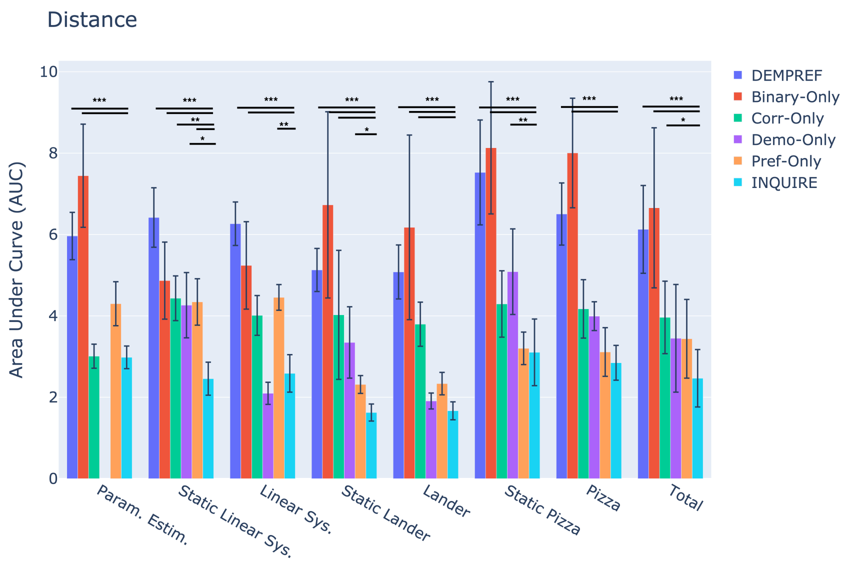
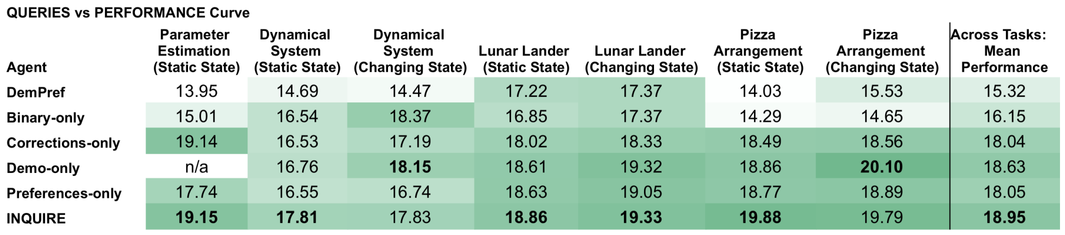


Figure 9: Visualizing Fig. 8, with statistical significance noted. (*: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$)



QUERIES vs PERFORMANCE Curve

Agent	Parameter Estimation (Static State)	Dynamical System (Static State)	Dynamical System (Changing State)	Lunar Lander (Static State)	Lunar Lander (Changing State)	Pizza Arrangement (Static State)	Pizza Arrangement (Changing State)	Across Tasks: Mean Performance
DemPref	13.95	14.69	14.47	17.22	17.37	14.03	15.53	15.32
Binary-only	15.01	16.54	18.37	16.85	17.37	14.29	14.65	16.15
Corrections-only	19.14	16.53	17.19	18.02	18.33	18.49	18.56	18.04
Demo-only	n/a	16.76	**18.15**	18.61	19.32	18.86	**20.10**	18.63
Preferences-only	17.74	16.55	16.74	18.63	19.05	18.77	18.89	18.05
INQUIRE	**19.15**	**17.81**	17.83	**18.86**	**19.33**	**19.88**	19.79	**18.95**

Figure 10: AUC values for the performance plots in Figs 2-3. Darker cells indicate higher (better) values.

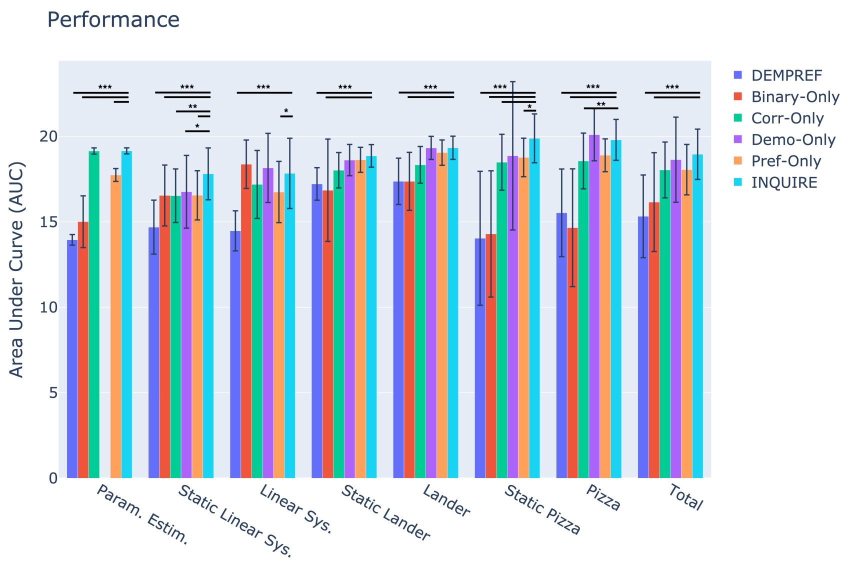
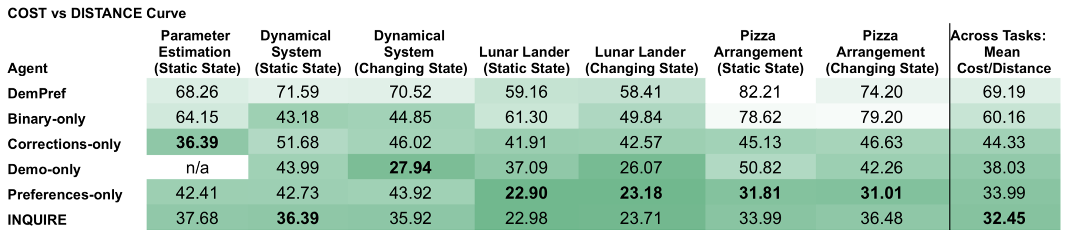


Figure 11: Visualizing Fig. 10, with statistical significance noted. (*: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$)


COST vs DISTANCE Curve

Agent	Parameter Estimation (Static State)	Dynamical System (Static State)	Dynamical System (Changing State)	Lunar Lander (Static State)	Lunar Lander (Changing State)	Pizza Arrangement (Static State)	Pizza Arrangement (Changing State)	Across Tasks: Mean Cost/Distance
DemPref	68.26	71.59	70.52	59.16	58.41	82.21	74.20	69.19
Binary-only	64.15	43.18	44.85	61.30	49.84	78.62	79.20	60.16
Corrections-only	**36.39**	51.68	46.02	41.91	42.57	45.13	46.63	44.33
Demo-only	n/a	43.99	**27.94**	37.09	26.07	50.82	42.26	38.03
Preferences-only	42.41	42.73	43.92	**22.90**	**23.18**	**31.81**	**31.01**	33.99
INQUIRE	37.68	**36.39**	35.92	22.98	23.71	33.99	36.48	**32.45**

Figure 12: AUC values for the cost plots in Figs 2-3. Darker cells indicate lower (better) values.

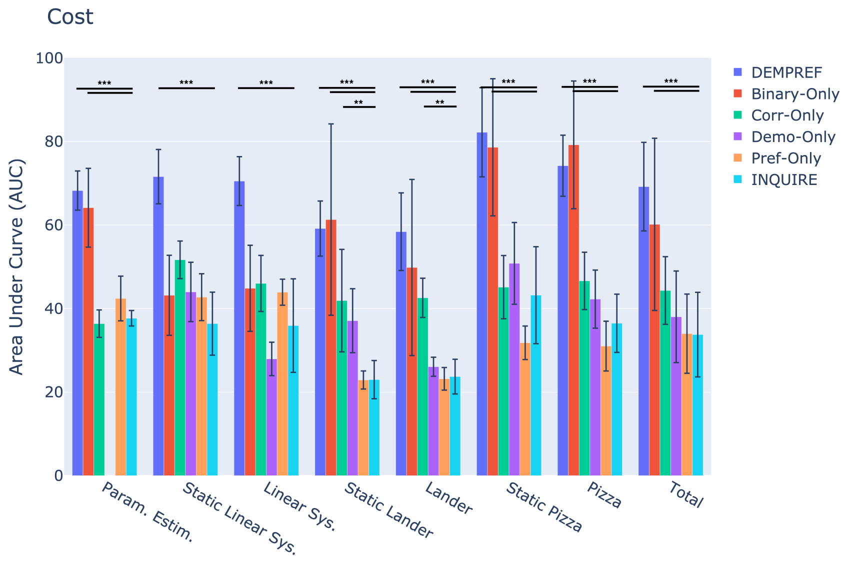


Figure 13: Visualizing Fig 12, with statistical significance noted. (*: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$)