

The City College of New York

CSC 59867-E: Senior Project II

AI Agents for Decision Making in the Real World

Counterfactual Design, Active Learning, and Autonomous Exploration

By Saptarashmi Bandyopadhyay
Email: sbandyopadhyay@ccny.cuny.edu
Assistant Professor of Computer Science
City College of New York and Graduate Center at the City University of New York

Saptarashmi Bandyopadhyay

September 23, 2026 CSC 59867 - E

Today's Agenda

Directed Exploration, Counterfactual Baselines, and Batch Active Learning.

~30/45 mins



In-Class Classwork 1 (*Sizing Processor Clusters*).

30 minutes

Sabyasachi Bandyopadhyay

Motivation of Today's Lecture

Rewards are often both sparse and composite (attributable to multiple factors), especially for real world environments

Random perturbation (Recall ϵ-greedy policies from last semester) fails exponentially as state and action spaces grow.

We'll be discussing three main topics related to this challenge:

Counterfactual Reasoning
Isolating an individual agent's contribution from a joint outcome.

Active Learning
Optimal batch experiment selection under uncertainty.

Autonomous Curiosity
Self-supervised intrinsic exploration without external rewards.

Saptarashmi Bandyopadhyay

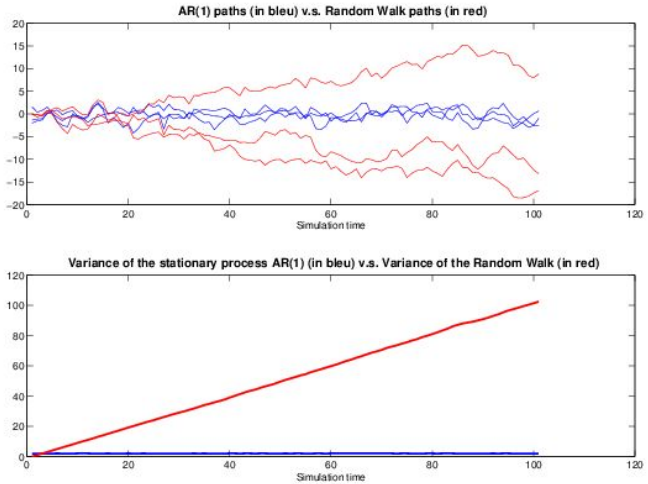
When does näive search fail?

If our action spaces have a high dimension d or we need to coordinate a large number of agents n, our state space coverage grows exponentially for a random walk: $\mathcal{O}(e^d)$ / $\mathcal{O}(|A|^n)$

Real-world costs of randomly searching:

- *Laboratory Automation*: Wasting expensive biological reagents and risking hardware collisions.
- *Power Grids*: Tripping transmission breakers, triggering cascading blackouts.

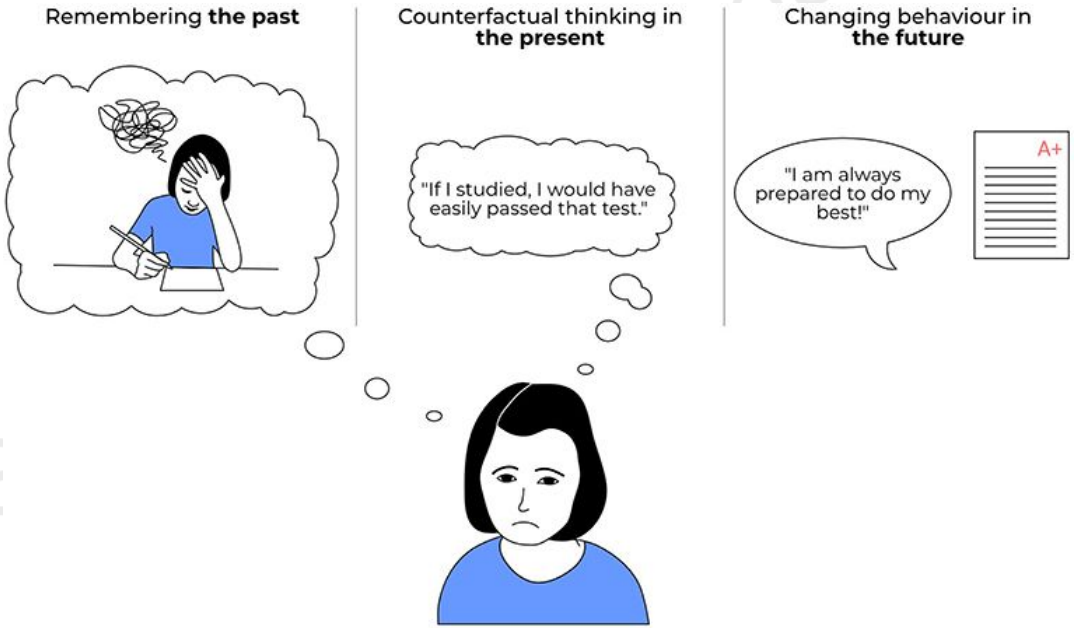
These problems have sparse external rewards that make gradient-based learning very difficult.



Example of random vs directed search

Sapthrashmi Bandyopadhyay

Counterfactual Thinking as a Concept



r

Multi-Agent Credit Assignment

How do we know which agents caused a success or a failure when their actions affect one another and the environment? Consider we're dealing with n agents that have a joint action $\mathbf{a} = \langle a_1, \dots, a_n \rangle$

Difference Rewards: Counterfactually compare the global reward against a system where agent i executed a neutral default action $\tilde{a}_i$:

$$d_i = \mathcal{R}(s, \langle a_i, \mathbf{a}_{-i} \rangle) - \mathcal{R}(s, \langle \tilde{a}_i, \mathbf{a}_{-i} \rangle)$$

The Aristocrat Utility: Eliminates the arbitrary choice of a "default action" by subtracting agent i's expected reward under its current policy π_i:

$$Adv_i(s, \mathbf{a}) = Q(s, \langle a_i, \mathbf{a}_{-i} \rangle) - \sum_{a'_i \in \mathcal{A}_i} \pi_i(a'_i | h_i) Q(s, \langle a'_i, \mathbf{a}_{-i} \rangle)$$

One method is the *Counterfactual Multi-Agent Policy Gradient (COMA)*. COMA takes a critic $Q(s, \mathbf{a})$ and creates a counterfactual adv. for each agent: $A_i(s, \mathbf{a}) = \mathcal{R}(s, \langle a_i, \mathbf{a}_{-i} \rangle) - \sum_{a'_i \in \mathcal{A}_i} \pi_i(a'_i | h_i) \mathcal{R}(s, \langle a'_i, \mathbf{a}_{-i} \rangle)$

Saptarashmi Bandyopadhyay

Active Learning via Gaussian Process Surrogates

Maximizing an expensive black-box objective $f(x)$ (e.g., peptide binding affinity, cluster runtime tuning) with minimal evaluations. A ***Gaussian Prior*** is the assumption that any finite collection of evaluation points is jointly Gaussian:

$$f(x) \sim \mathcal{GP}\Big(\mu(x), k(x, x')\Big)$$

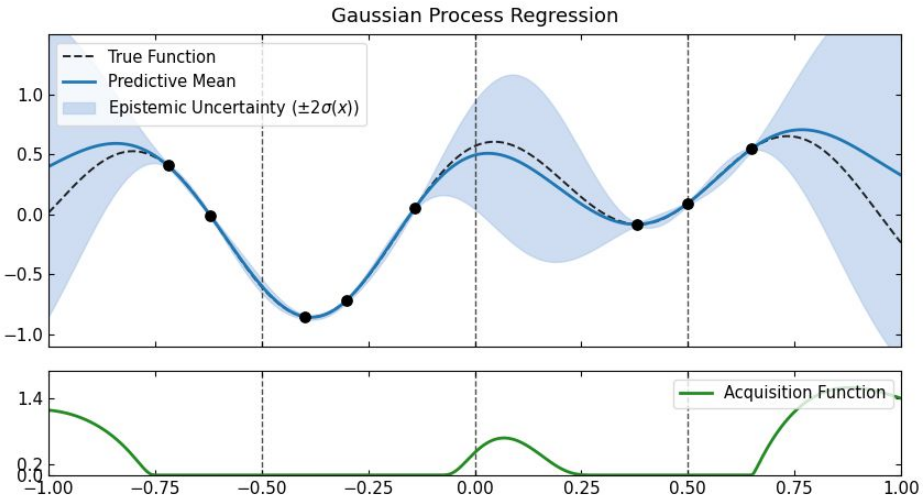
Posterior Inference Given Data $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^n$:

- Predictive Mean: $\mu_n(x) = \mathbf{k}(x)^\top (K + \sigma^2 I)^{-1} \mathbf{y}$
- Predictive Variance: $\sigma_n^2(x) = k(x, x) - \mathbf{k}(x)^\top (K + \sigma^2 I)^{-1} \mathbf{k}(x)$

Acquisition Strategy: Use posterior mean $\mu_n(x)$ (exploitation) and posterior uncertainty $\sigma_n(x)$ (exploration) to guide the next sampling point.

Soptarshi Bandyopadhyay

Example: Gaussian Process Regression



Batch Active Learning - Expected Improvement (EI)

We need to eval batches of candidates in parallel ($q > 1$), not one at a time in a serial thread.

Multi-Point Acquisition Formulation: $$\alpha_{q\text{EI}}(X) = \mathbb{E}\left[\max\left(0, \max_{j=1,\dots,q} f(x_j) - y^*\right) \,\middle|\, \mathcal{D}\right]$$

where $X = \{x_1, \dots, x_q\}$ is a candidate batch and y^* is the current best observed value.

Analytic Intractability: For $q > 1$, joint Gaussian order statistics cannot be integrated analytically.
Monte Carlo Acquisition: Approximate the expectation over N posterior samples:

$$\xi(X) = \mu_{\mathcal{D}}(X) + L_{\mathcal{D}}(X)\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$$

The Reparameterization Trick: Generate differentiable posterior samples via Cholesky decomposition of the predictive covariance ($K_{\mathcal{D}}(X) = LL^\top$): $$\hat{\alpha}_{q\text{EI}}(X) \approx \frac{1}{N}\sum_{i=1}^{N} \max\left(0, \max_{j=1,\dots,q} \xi^{(i)}(x_j) - y^*\right)$$

Saptarashmi Bandyopadhyay

Autonomous Exploration

Exploration via Self-Supervised Prediction: Train an auxiliary model to predict environment transitions; reward the agent for prediction errors.

Using intrinsic rewards, we can reward both external signals and internal, novel behavior:

$$r_t = r_t^{\text{ext}} + \eta \cdot r_t^{\text{int}}$$

Forward Dynamics Prediction Error:

$$r_t^{\text{int}} = \|\hat{\phi}(s_{t+1}) - g_\theta(\phi(s_t), a_t)\|_2^2$$

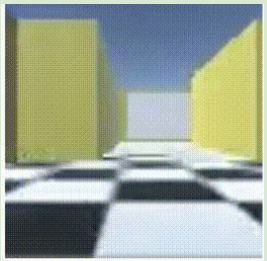
The "Noisy-TV" Failure Mode (Aleatoric Uncertainty):

- In stochastic environments, true transition entropy cannot be compressed (e.g., static noise on a screen, ambient lighting changes). An agent maximizing forward prediction error becomes addicted to random white noise; prediction error remains high without yielding task progress.

Saptarashmi Bandyopadhyay

Example: AI Agent watching too much TV

No Noisy TV



With Noisy TV



Random Network Distillation (RND)

- **Target Network** $f(s)$: A randomly initialized neural network, **permanently frozen**. Maps states to a fixed feature embedding space $\mathbb{R}^k$.
- **Predictor Network** $\hat{f}_\theta(s)$: Trained online on the agent's collected states to minimize error (MSE).

The RND Intrinsic Reward: $r_t^{\text{int}} = \|\hat{f}_\theta(s_{t+1}) - f(s_{t+1})\|_2^2$

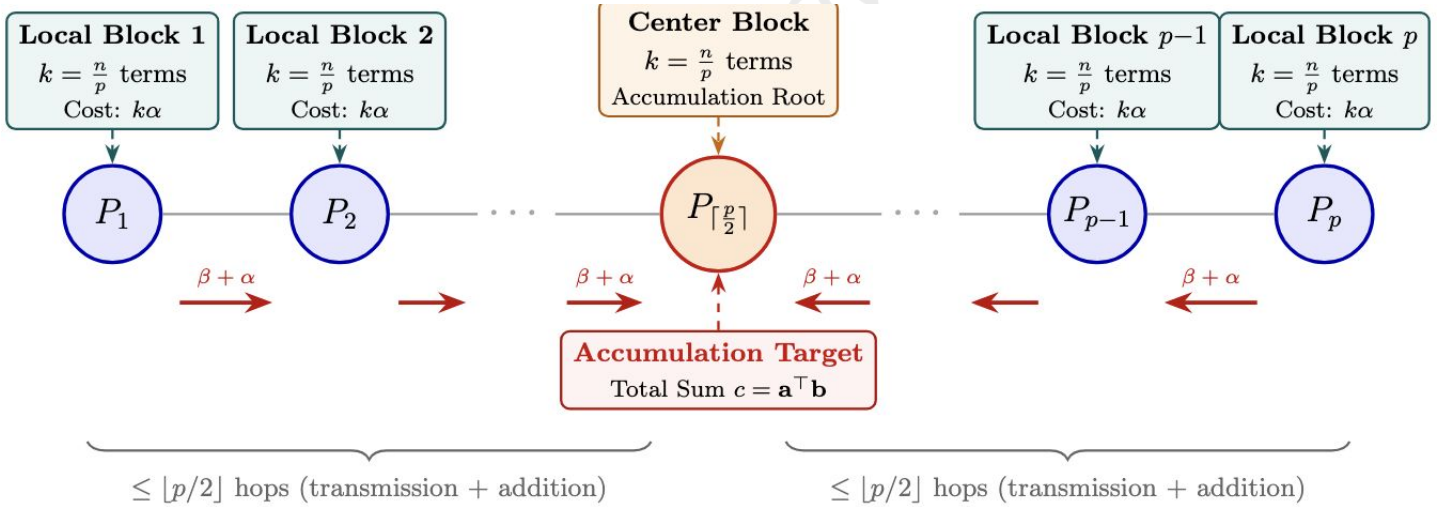
Why RND Solves the Noisy-TV Problem:

- The target function $f(s)$ is strictly deterministic.
- Prediction error stems purely from **epistemic uncertainty** (novelty), not environment stochasticity. Frequently visited states converge to zero error.

Dual Value Heads: Maintain separate value critics V_E (discounted by $\gamma_E = 0.999$, episodic) and V_I (discounted by $\gamma_I = 0.99$, non-episodic) to balance exploration against task survival.

Sanjanashini Bandyopadhyay

In-Class Problem Visualization



Questions?

Saptarashmi Bandyopadhyay