

Offline Reinforcement Learning

You cannot try. Learn from what was already done.

Jinkyoo Park · KAIST

The last thing taken away is the right to act at all

Where we are — the last thing taken away

PREVIOUS STEP

Lectures 8–11 could collect new transitions during learning.

THIS CHAPTER'S QUESTION

Now the training log is fixed.

Lecture 5 already showed how optimizing a learned score can exploit error. Here unsupported action values also enter repeated Bellman targets.

Estimate a policy's return from a log and identify an unsupported action.

Learning route — improve a policy using only the log

Bring: TD3, Bellman fitting and a basic probability ratio.

FIRST PASS	WHAT TO DO
Follow the idea	Diagnose missing support → TD3+BC → CQL/IQL → evaluate
Work without the solution	Estimate a policy's return from a log and identify an unsupported action.
Return later	Full CQL guarantees, model/sequence alternatives and advanced OPE estimators are references.

For the temperature thread: **predict** → **calculate** → **reveal and check** → **change one condition**. Complete the core calculation before reading the research extensions.

— ACT 1

What breaks: distributional shift

Q1. Nothing in the update changes. Only the guarantee that someone will check.

The setting, and the sentence that breaks

We are handed $D = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^N$, collected by some **behaviour policy** β that may be a human operator, an old controller, or a mixture of several. No more data will arrive. Run Lecture 8's update on it, unchanged:

$$y = r + \gamma \max_{a'} Q(s', a')$$

The `max` ranges over **every** action, including actions β never took at s' . There is no datum anywhere in D that contradicts whatever Q happens to say there; a function approximator will extrapolate cheerfully; and the `max` is not a passive reader of that extrapolation — it **actively seeks out whichever one is highest**. The result is written into the next target, and bootstrapped again.

Every algorithm in Part IV assumed it could try something. Take that away and an estimated Bellman target becomes unreliable — because the one term it needs is evaluated **at actions nobody ever took**.

The one term nobody measured

Q1 What breaks?

Q2 Constrain the policy?

Q3 Constrain the value?

Q4 How would you know?

$$Q(s, a) \leftarrow r + \gamma \max_{a'} Q(s', a') \quad \text{with } (s, a, r, s') \sim D, \quad a' \text{ chosen by the max, not by } \beta$$

The transition is real: s, a, r, s' were all measured. The highlighted term was not — it is the model's opinion about an action at a state, and for most (s', a') that opinion rests on **no evidence whatsoever**.

Call the gap $\epsilon(s', a') = Q(s', a') - Q^*(s', a')$. Three properties make it lethal together:

- **It is unbounded off-support.** A neural network's error away from its training distribution has no bound; only its architecture decides what it says there.
- **The max selects for it.** $\max_{a'} Q = \max_{a'} (Q^* + \epsilon)$ favors a large combination of true value and error. It need not pick the largest error, but optimization can select optimistic mistakes.
- **The bootstrap compounds it.** That target becomes a label; the fit spreads it to neighbouring (s, a) ; the next sweep takes a max over the raised surface.

(Fujimoto, Meger & Precup, 2019) name the first of these **extrapolation error**. Their experiment: a DDPG agent trained on the replay buffer of *another, concurrently running* DDPG agent — the same task, the same algorithm, near-identical data — fails, while the agent that generated the buffer succeeds. The only difference is who chose the actions.

Online, the environment answers back. Offline, nothing does.

Off-policy learning has always been hard. What follows is not that difficulty; it is the removal of the thing that made it survivable.

ONLINE — A CLOSED LOOP

Q over-rates $(s, a) \Rightarrow$ the policy tries a at $s \Rightarrow$ the environment returns a disappointing $r, s' \Rightarrow$ the target drops $\Rightarrow Q$ falls.

New visits **can provide corrective evidence**; online learning still needs exploration and a stable update method.

OFFLINE — THE LOOP IS CUT

Q over-rates $(s, a) \Rightarrow$ the policy would try a at $s \Rightarrow$ **but it may not** $\Rightarrow$ no transition from (s, a) is ever added $\Rightarrow$ the target never drops.

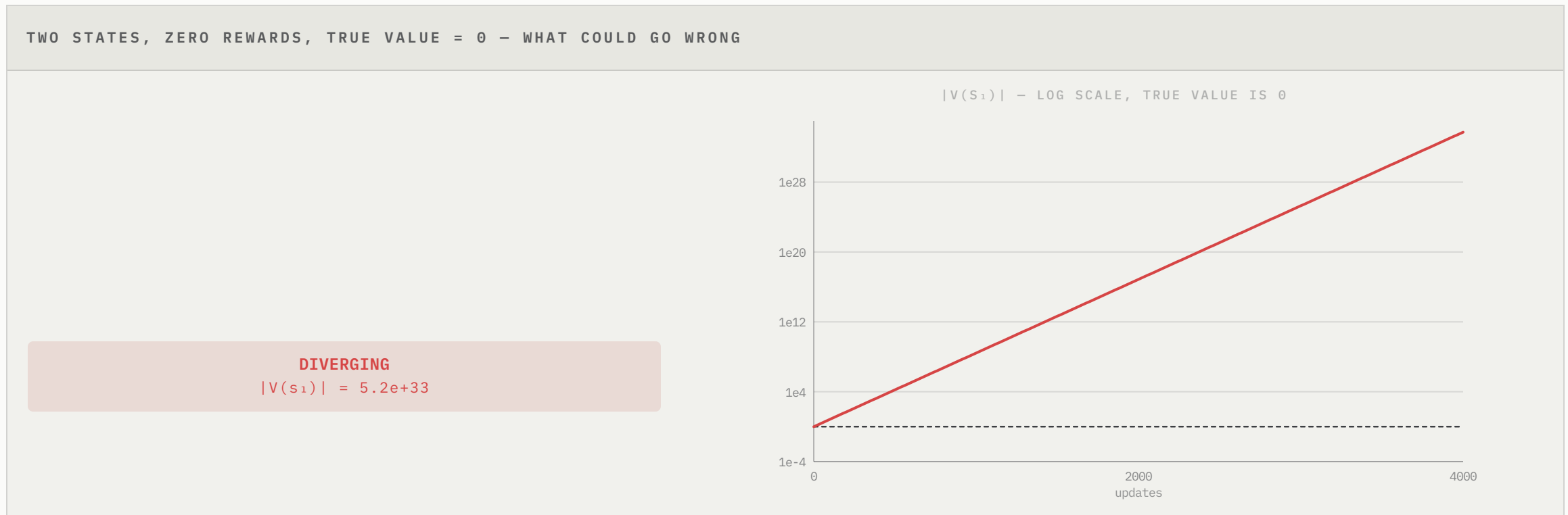
The log supplies no direct corrective observation at that pair. Without other structure or regularization, the error may persist and grow.

Optimism is cheap when it will be tested. Offline it is never tested, so **optimism becomes the answer you ship**.

This is Lecture 4 against Lecture 5 again, in the dynamic world. There, uncertainty was an *opportunity* while the oracle remained and a *hazard* once it was gone. Here, exploration was an opportunity while the environment remained, and the same width in Q is now a hazard.

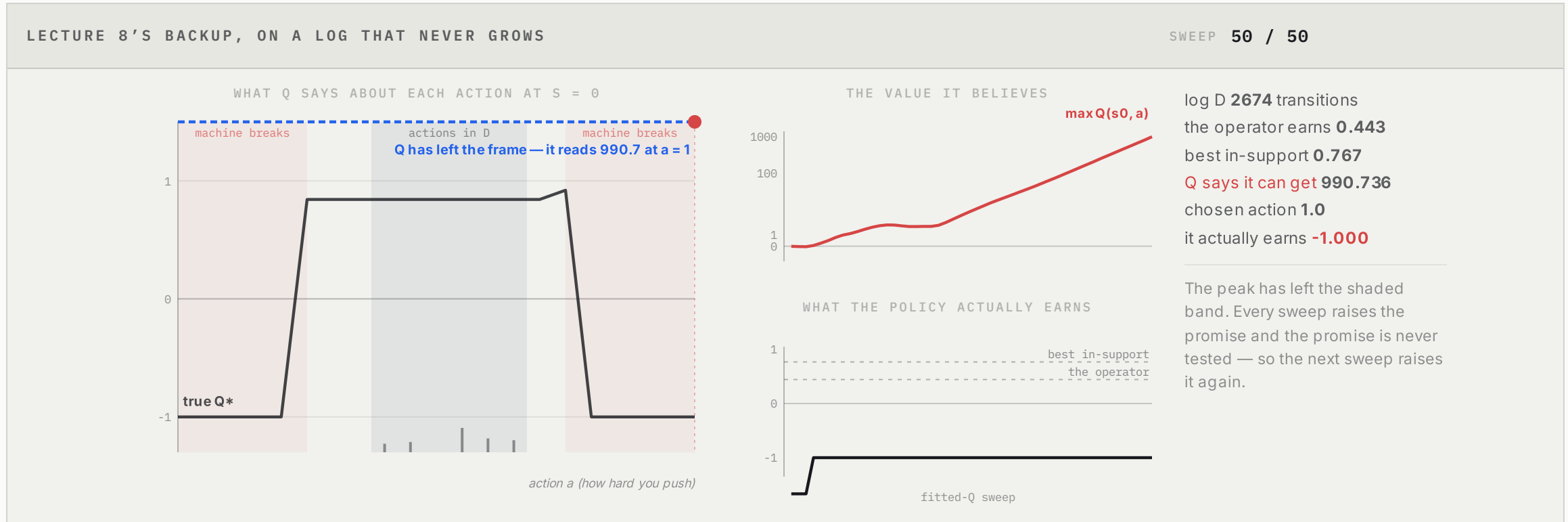
The deadly triad, with the escape hatch removed

Lecture 8's Act 4 named three ingredients — function approximation, bootstrapping, off-policy data — whose combination can cause divergence. Standard bootstrapped offline value learning has all three: the data is not merely off-policy, it comes from a policy we did not choose and cannot re-run. And what online RL had, the ability to go and visit the state-action pair it is wrong about, is precisely what has been withdrawn.



Lecture 8's counterexample, unchanged. Read the third switch again — *update* $s_1 \rightarrow s_2$ *only*. Online that switch is a modelling choice; **offline it is the dataset**, and there is no switching it off. (Tsitsiklis & Van Roy, 1997)

Watch it happen



A machine on a ten-step track. The action $a \in [-1, 1]$ is how hard you push: bigger jumps further, and within the data bigger is genuinely better — until $|a| > 0.5$, where the machine breaks. The operator who logged D never pushed past 0.3. Run Lecture 8's backup unchanged and $\hat{V}(s_0) = \max_a Q$ climbs from 0.21 to **990.7** in fifty sweeps — a factor of 1.21 per sweep, without bound — while the greedy policy it implies scores **-1.00** in the real machine, every single sweep. Then restrict the $\max$ to the five actions in D : the same code settles at 0.788 and returns 0.767, the best any in-support policy can do.

A target can grow without one new observation

The log contains a transition with reward **1** to state B. At B, action **stay** was observed and has estimated value 2; action **jump** was never observed but the network predicts 20. Let $\gamma = 0.9$.

BACKUP	CONTINUATION USED	TARGET
unrestricted maximum	20, from the unsupported action	$1 + 0.9(20) = 19$
maximum over observed actions	2	$1 + 0.9(2) = 2.8$

Both targets use the same real reward. The difference comes entirely from an **unverified continuation estimate**. Restricting the action set reduces this particular extrapolation risk; it is not a proof that all remaining estimates are correct.

The same disease, one rung up the course

Lecture 5, Act 3 — the slide that said it would be quoted here

COMS — A CONSERVATIVE OBJECTIVE

$$\theta^* = \arg \min_{\theta} \left\{ \frac{1}{2} \mathbb{E}_D[(f_{\theta}(x) - y)^2] + \alpha (\mathbb{E}_{\mu}[f_{\theta}] - \mathbb{E}_D[f_{\theta}]) \right\}.$$

The optimiser exploits f_{θ} at **out-of-distribution inputs**.

CQL — A CONSERVATIVE VALUE

$$Q^* = \arg \min_Q \left\{ \frac{1}{2} \mathbb{E}_D[(Q - \mathcal{B}^{\pi} \hat{Q})^2] + \alpha \mathbb{E}_{s \sim D} \left[\log \sum_a e^{Q(s,a)} - \mathbb{E}_{a \sim \hat{\pi}_{\beta}(\cdot|s)}[Q(s,a)] \right] \right\}.$$

The policy exploits Q at **out-of-distribution actions**.

Fit the data · push down what the optimiser would chase · hold up what the data actually contains. **One shape, twice.**

Lecture 5 promised this slide would return with a policy in place of an optimiser and a Q -function in place of a surrogate. It has. The analogy replaces a design x by a state-action pair (s, a) . CQL also includes a Bellman target and a specific action-distribution penalty, so it is not simply COMs with renamed variables. Act 3 develops the loss.

Check — where offline RL breaks

You run ordinary Q-learning on a fixed dataset, changing nothing. The learned Q climbs steadily while the policy's true return falls. What is happening?

- ☒ A The max in the target queries Q at actions absent from the data; those values are overestimated, and with no way to try them the error is never corrected
- ☐ B The learning rate is too high
- ☐ C The dataset is too small to fit Q
- ☐ D The discount factor is too close to 1

Online interaction can supply corrective data for visited actions. Offline, that feedback loop is cut. Maximization can select optimistic unsupported values and feed them back as targets, causing estimates to inflate. It is Lecture 5's adversarial optimiser, wearing a policy.

— ACT 2

Constrain the policy

Q2. If the trouble is that π asks about actions β never took, forbid the question.

Why not simply copy the data?

Q1 What breaks?

Q2 Constrain the policy?

Q3 Constrain the value?

Q4 How would you know?

The trivial offline algorithm is **behaviour cloning** — supervised learning of $\hat{\pi}(a \mid s)$ from D , no Bellman equation anywhere:

$$\hat{\pi} = \arg \max_{\pi} \mathbb{E}_{(s,a) \sim D} [\log \pi(a \mid s)]$$

It directly fits demonstrated actions. Exact replication has the same expected performance as the behavior policy, but approximation, generalization, and action selection can make it better or worse. It has **no reward-based mechanism for selecting better actions** and no automatic safety guarantee.

WHAT OFFLINE RL IS FOR

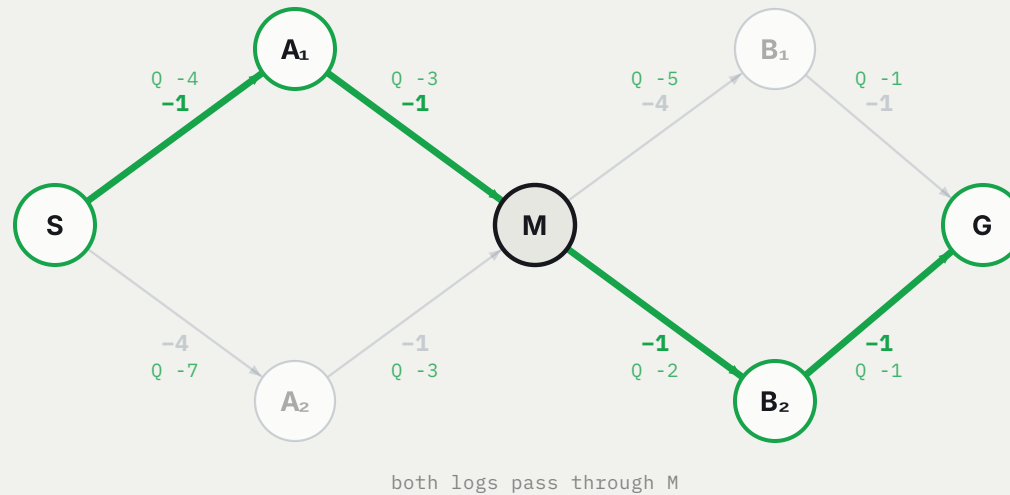
the one capability cloning does not have

Reward-guided stitching. A Markov state can connect the good first half of one run to the good second half of another. Bellman backups use rewards to identify those continuations. A state-based clone can also produce unseen combinations, but does not explicitly select them by return.

So the offline problem is not "imitate the data" but **seek better expected return using supported actions and transitions**. Hold those two clauses together: support reduces extrapolation risk; it does not by itself establish safety or improvement.

Stitching, run

TWO MEDIOCRE LOGS, AND THE ROUTE NEITHER TOOK



20 logged transitions

best trajectory in D **cost 7**

behaviour cloning **cost 7**

offline Q-learning **cost 4**

sampling the clone instead of taking its mode: 7.0

The backup at **M** compares two continuations that came from **different trajectories**. Bellman does not know or care which; it takes the max. Cost **4**, on a route no one drove.

Two logged routes from **S** to **G**, crossing at **M**. One is cheap early and expensive late; the other is expensive early and cheap late. Both cost **7**. Behaviour cloning reproduces them and costs **7** too, whether it copies the modal action or samples the whole distribution. Tabular *Q*-learning on the *same twenty transitions* returns $S \rightarrow A_1 \rightarrow M \rightarrow B_2 \rightarrow G$ at a cost of **4** — a route no one ever drove, assembled entirely from steps that were.

TD3+BC — one line, and the price of the family

Fujimoto & Gu, NeurIPS 2021

Take TD3 — Lecture 10's overestimation-hardened DDPG — exactly as it stands. Add one term to the actor loss:

$$\pi \leftarrow \arg \max_{\pi} \mathbb{E}_{(s,a) \sim D} \left[\lambda Q(s, \pi(s)) - (\pi(s) - a)^2 \right], \quad \lambda = \frac{\alpha}{\frac{1}{N} \sum_i |Q(s_i, a_i)|}$$

A behaviour-cloning term, and a normaliser λ that reduces sensitivity to the scale of Q ; it does not eliminate task dependence. No generative model, no divergence estimate, no extra network.

Fujimoto & Gu report competitive performance against more elaborate offline methods on their D4RL evaluation, with substantially lower training overhead. This supports using a simple baseline before adding machinery; it is not a claim of dominance on every offline dataset. [A Minimalist Approach to Offline Reinforcement Learning \(NeurIPS 2021\)](#)

THE COST THE WHOLE FAMILY PAYS

and it cannot be paid down

A tight distribution constraint can limit improvement. A support constraint can still select better actions than the behavior policy by changing their probabilities. The right restriction depends on coverage and estimation error.

Three ways to say "stay close"

METHOD	WHAT IT CONSTRAINS	THE MECHANISM
BCQ (Fujimoto, Meger & Precup, 2019)	the support of π	fit a generative model $G_\omega(s) \approx \beta$, sample n candidate actions, allow a small learned perturbation ξ_ϕ , and take the max only over those
BEAR (Kumar et al., 2019)	the support , softly	an MMD penalty between $\pi(\cdot s)$ and $\hat{\beta}(\cdot s)$ — encourages similarity using finite-sample MMD; MMD is not literally a support-only distance
BRAC (Wu, Tucker & Nachum, 2019)	the distribution	an explicit divergence, usually KL, penalised in the actor loss or subtracted from the reward

The distinction in column two is the one that matters. A **support** constraint says *only propose actions β might plausibly have taken*; a **distribution** constraint says *propose them about as often as β did*. A support restriction allows reweighting of observed choices; a tight distribution penalty limits it — at M , both continuations are in β 's support, so BCQ may take the better one, while a tight KL to β drags the choice back toward the mixture.

Written as a modified backup, BCQ is one edit to the target: $y = r + \gamma \max_{a' \in \mathcal{A}_G(s')} Q(s', a')$ with $\mathcal{A}_G(s')$ the sampled candidate set. That is the **escape hatch from Act 1's widget, made into an algorithm**.

Check — what offline RL is for

If the policy must stay close to the data-collecting policy, why not simply clone it? What does a reward-based offline RL objective explicitly try to achieve?

- ☐ A Learn from fewer trajectories
- ☒ B Use rewards to choose good segments across trajectories, potentially improving on the logged behavior
- ☐ C Handle continuous action spaces
- ☐ D Guarantee it will never perform worse than the behaviour policy

Cloning maximizes action likelihood rather than return. Because the Bellman equation propagates value **across** trajectories through shared states, offline RL can discover that the first half of one poor trajectory joins the second half of another to make a good one — a route nobody in the dataset ever drove. That is the whole reason to accept the difficulty.

— ACT 3

Constrain the value

Q3. Do not fence the policy in. Make the ground outside the fence look as unattractive as our ignorance deserves.

Fence the value, not the policy

Q1 What breaks?

Q2 Constrain the policy?

Q3 Constrain the value?

Q4 How would you know?

$$\min_Q \alpha \left(\underbrace{\mathbb{E}_{s \sim D, a \sim \mu} [Q(s, a)]}_{\text{(i) push down what } \pi \text{ likes}} - \underbrace{\mathbb{E}_{(s, a) \sim D} [Q(s, a)]}_{\text{(ii) hold up what the data has}} \right) + \underbrace{\frac{1}{2} \mathbb{E}_D [(Q - \mathcal{B}\hat{Q})^2]}_{\text{(iii) the usual Bellman error}}$$

- **(i)** μ is the distribution the learner is drawn toward — in CQL(H), the softmax of Q itself, which makes the term a soft max. Whatever Q currently loves, push it down.
- **(ii)** the counter-term offsets pessimism on data actions. With a fixed target, the squared Bellman error already anchors observed predictions; removing the counter-term does not imply every value tends to minus infinity.
- **(iii)** unchanged from Lecture 8.

The policy may still maximise freely. It will simply find that **the peaks it used to climb are no longer there**. (Kumar, Zhou, Tucker & Levine, 2020)

Turning the dial



The same machine, the same logs, the same fifty sweeps — only α changes. At $\alpha = 0$ the value reaches 990.7 against a truth of -1.00 : **over-promised by 991.7**. At $\alpha = 0.01$ it reports 0.653 and the policy actually earns **0.767** — an honest under-promise, and the best return any in-support policy can reach, against the operator's own 0.443. Keep turning: by $\alpha = 0.3$ the argmax has been squeezed back onto β 's single most common action and earns 0.361 — **worse than the logs it was learned from**.

IQL — never ask the question at all

Kostrikov, Nair & Levine, ICLR 2022

CQL still evaluates Q at out-of-distribution actions in order to push them down. IQL declines to evaluate them at all. Fit a **state** value V by expectile regression toward Q on in-data actions, then bootstrap through V :

$$L_V = \mathbb{E}_{(s,a) \sim D} \left[L_2^\tau(Q(s, a) - V(s)) \right], \quad L_Q = \mathbb{E}_{(s,a,r,s') \sim D} \left[(r + \gamma V(s') - Q(s, a))^2 \right]$$

with the asymmetric loss $L_2^\tau(u) = |\tau - \mathbf{1}\{u < 0\}| u^2$.

$\tau = \frac{1}{2}$ is ordinary least squares and V becomes $\mathbb{E}_{a \sim \beta}[Q]$ — policy evaluation of β . Raise τ and under-predictions are penalised more heavily, so V is pulled toward the **upper** expectile of Q over the actions β took. As $\tau \rightarrow 1$ it approaches $\max_{a \in \text{supp } \beta} Q(s, a)$.

The improvement operator becomes an **in-support maximum, obtained without ever writing a max**. Every Q the algorithm touches sits on a state–action pair that was measured.

The policy is then extracted separately by advantage-weighted regression, $\pi = \arg \max_\pi \mathbb{E}_D [\exp(\kappa(Q(s, a) - V(s))) \log \pi(a | s)]$ — a weighted behaviour clone, which is why IQL avoids one important source of training extrapolation. The price is that the in-support maximum is estimated from however many actions β happened to try at s ; where β was nearly deterministic, there is nothing for the expectile to climb.

Expectile regression — two observed actions are enough to see it

At one state, suppose the log contains the two actions equally often, with Q values **2 and 6**. For an expectile m between them:

$$\tau(6 - m) = (1 - \tau)(m - 2).$$

EXPECTILE LEVEL	FITTED VALUE	INTERPRETATION
$\tau = 0.5$	4	ordinary mean
$\tau = 0.8$	5.2	pulled toward the higher observed value
$\tau \rightarrow 1$	approaches 6	approaches the observed maximum

IQ-L can favor better observed actions without asking Q to score a newly invented action. At finite τ , an expectile is **not an exact maximum**; the learned actor can still generalize outside the data.

One dial, and both ends are bad

this is Lecture 5's α , again

	TOO LITTLE	TOO MUCH
COMs , α	the optimiser escapes the data and returns a hallucinated design	the surface flattens; ascent cannot move at all
CQL , α	the max escapes the support and Q diverges	$\hat{Q}$ tracks $\log \hat{\beta}$; the argmax becomes β 's modal action — behaviour cloning, arrived at by accident
IQL , τ	$V \rightarrow \mathbb{E}_{\beta}[Q]$; mean-value fitting; actor weighting can still favor better actions	the expectile chases the largest <i>sampled</i> Q and re-imports the over-estimation it was designed to avoid

It is the same dial in all three rows — **how far may we trust a model beyond its evidence** — and in all three, extreme settings can be unhelpful; the three parameters are not mathematically identical.

CQL's Lagrangian variant turns the penalty into a constraint with a budget. TD3+BC instead normalizes the Q term to reduce sensitivity to its numerical scale. Both address balancing terms, but **normalization is not a constrained optimization derivation**; cross-task transfer still needs checking.

One offline problem, three different interventions

All three start with a fixed log and face the same risk: the learned policy may prefer actions whose value is poorly supported by that log.

MAIN METHOD	WHAT IS CHANGED?	QUESTION TO ASK WHEN READING ITS LOSS
TD3+BC	The actor objective	How strongly does copying logged actions oppose maximizing the critic?
CQL	The critic objective	Which candidate actions have their values pushed down relative to the data?
IQL	Value fitting and policy extraction	How do expectile fitting and weighted cloning use logged actions without maximizing over new actions during value training?

These are different responses to **unsupported improvement**. None creates missing evidence about an unseen action. Judge the resulting policy using the coverage and evaluation checks in Act 4.

Check — the quotation from Lecture 5

CQL adds a term that pushes Q *down* on actions not in the data. Which earlier method has a closely related motivation?

- ☐ A The trust region of Lecture 1 and Lecture 10
- ☐ B The ε -greedy exploration tax of Lecture 8
- ☒ C The conservative objective model of Lecture 5 — penalize unsupported high predictions while fitting the data, reducing the incentive to exploit estimation error
- ☐ D The target network of Lecture 8

Both chapters study optimization with imperfect models and limited data. COMs regularizes a design surrogate; CQL regularizes action values inside Bellman learning. Their conservative objectives are related, but their guarantees and assumptions differ. Excessive conservatism can suppress useful improvements.

— ACT 4

Evaluate a policy using the fixed log

Q4. What evidence in the log supports a claim of improvement?

Off-policy evaluation — the question that comes first

You have a candidate policy π and a fixed log D . If new evaluation trials are unavailable, estimate V^π using the log and explicit coverage assumptions. Offline RL restricts interaction during training; it does not by definition prohibit every later deployment.

IMPORTANCE SAMPLING

$$\hat{V}_{\text{IS}} = \frac{1}{n} \sum_{j=1}^n \left(\prod_{t=1}^H \frac{\pi(a_t^j | s_t^j)}{\beta(a_t^j | s_t^j)} \right) G^j$$

Unbiased with correct behavior probabilities, matching environment dynamics, and target-policy support covered by the data policy. The weight is a **product of ratios**.

PER-DECISION IS (PRECUP, SUTTON & SINGH, 2000)

$$\hat{V}_{\text{PDIS}} = \frac{1}{n} \sum_j \sum_{t=1}^H \gamma^{t-1} \left(\prod_{t' \leq t} \rho_{t'}^j \right) r_t^j$$

A reward at step t cannot depend on later actions, so it should not be reweighted by them. This avoids unnecessary future-action ratios; it often reduces variance, without a universal strict ranking.

In the independent, identical-step example below, let $\mathbb{E}_\beta[\rho^2] = q$. Then $\mathbb{E}[W^2] = q^H$ and $\text{Var}(W) = q^H - 1$, so the estimator's standard error grows like $q^{H/2}$ — geometric in the horizon, with a base fixed by how far π has moved from β . For a fixed dataset, this can make the estimate too uncertain to compare policies reliably.

Importance sampling — change frequencies, not rewards

One decision, two actions. The behavior policy uses probabilities **(0.5, 0.5)**; the target policy uses **(0.8, 0.2)**. Rewards are **(1, 0)**.

LOGGED ACTION	TARGET / BEHAVIOR WEIGHT	WEIGHTED REWARD
A	$0.8/0.5 = 1.6$	1.6
B	$0.2/0.5 = 0.4$	0

With one observation of each, IS gives $(1.6 + 0)/2 = 0.8$, matching the target's true value. This exact agreement is specific to this balanced sample.

If behavior never chooses A, its ratio is undefined and the log cannot identify A's reward without additional assumptions. **No estimator repairs missing support by arithmetic alone.**

Temperature thread — evaluate the heater policy from a log

shared teaching example · predict before revealing the calculation

Use only this one-step log; the learner has no simulator formula. Behavior chooses off/heat with probabilities (0.5, 0.5). The log contains one off reward -4 and one heat reward -2 . The proposed policy uses probabilities (0.2, 0.8).

Predict: Should giving the lower-cost action more probability improve the estimated return?

Calculate and check. Weights are $0.2/0.5 = 0.4$ and $0.8/0.5 = 1.6$. The IS estimate is $[0.4(-4) + 1.6(-2)]/2 = -\mathbf{2.4}$, compared with logged mean -3 . This exact calculation uses the balanced sample shown.

Weight observed outcomes by how their probability changes. Do not substitute the critic score for a measured reward.

Try it — the log contains no heating actions

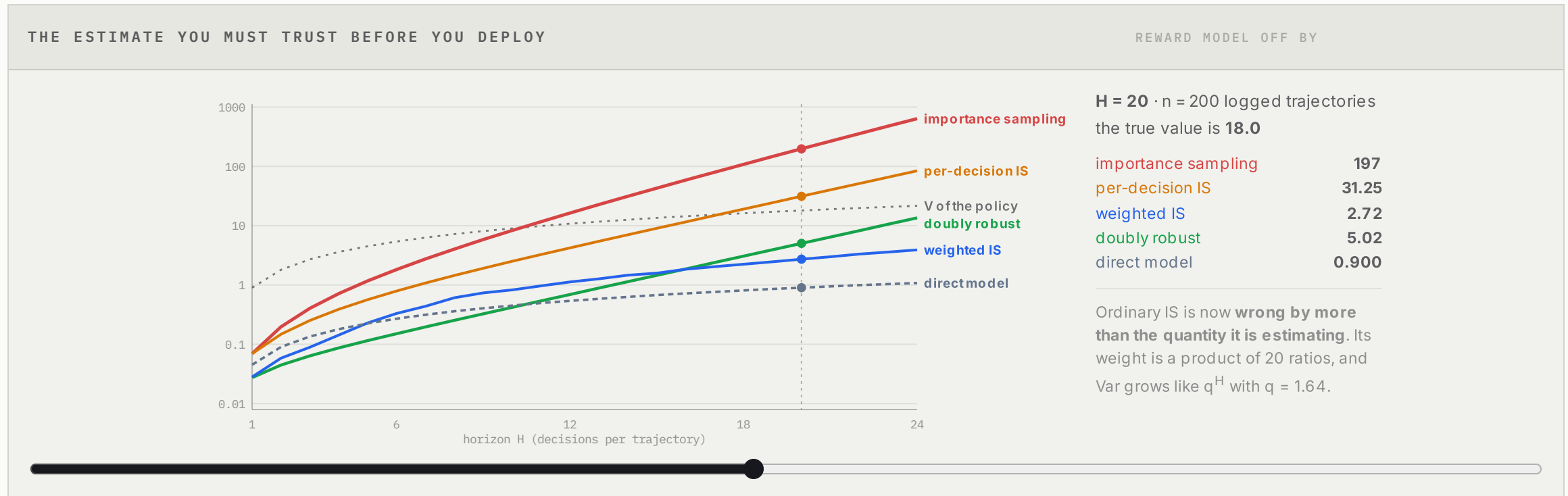
work independently · reveal only after writing an answer

Suppose behavior always chose off, but the proposed policy still heats with probability 0.8. Can the existing log identify its value without extra assumptions about heating outcomes?

Check your answer. No. The heating probability ratio has zero behavior probability in its denominator, and no heating rewards were observed. Many different heating outcomes are compatible with the same all-off log.

Model predictions can add assumptions, but arithmetic cannot create missing evidence. Coverage is part of the evaluation question.

The variance, measured



H decisions, two actions, β a coin flip and π choosing the good action nine times in ten, so $q = 1.64$ and $V^\pi = 0.9H$. With $n = 200$ logged trajectories, ordinary IS has a root-mean-square error of 0.069 at $H = 1$ and **636 at $H = 24$** — where the quantity being estimated is 21.6. Doubly robust with a 5 % reward-model error runs a decade and a half below it, **and parallel to it**: it scales the exponential down, it does not remove it. Self-normalized and fitted estimators look steadier in this example; that does not guarantee accurate evaluation at long horizons in general.

What evidence supports the recommendation?

CHECK	WHAT TO REPORT
Coverage	Does the log include actions the proposed policy could choose?
Estimated return	State the estimator, behavior probabilities and assumptions.
Stability	Inspect large importance weights and sensitivity to the evaluation data.
Comparison	Compare with the logged policy and a simple baseline under the same protocol.

A high fitted Q value is a prediction. A defensible improvement claim needs **evaluation evidence and its limits**. FQE, doubly robust estimators and model-based alternatives are retained in the appendix.

Check — the axis left uncrossed

The course toured the decision cube along the stages and model axes. Which axis is deliberately never crossed, and where does it go?

- ☐ A The stages axis — static to dynamic, left to a later course
- ☐ B The model axis, since offline RL is neither model-based nor data-driven
- ☐ C None — all three are crossed by Lecture 12
- ☒ D The agents axis — single to multi-agent, which is where IE579 begins

Our main optimization and RL methods use one decision maker's objective. With several decision makers, outcomes also depend on their interaction: objectives may align, conflict, or partly overlap. Equilibrium concepts become relevant, but adding an agent does not automatically turn every optimization problem into a Nash-equilibrium problem. IE579 develops these strategic models.

— CLOSING

Closing

One failure, three answers — and then the whole map, stood back up.

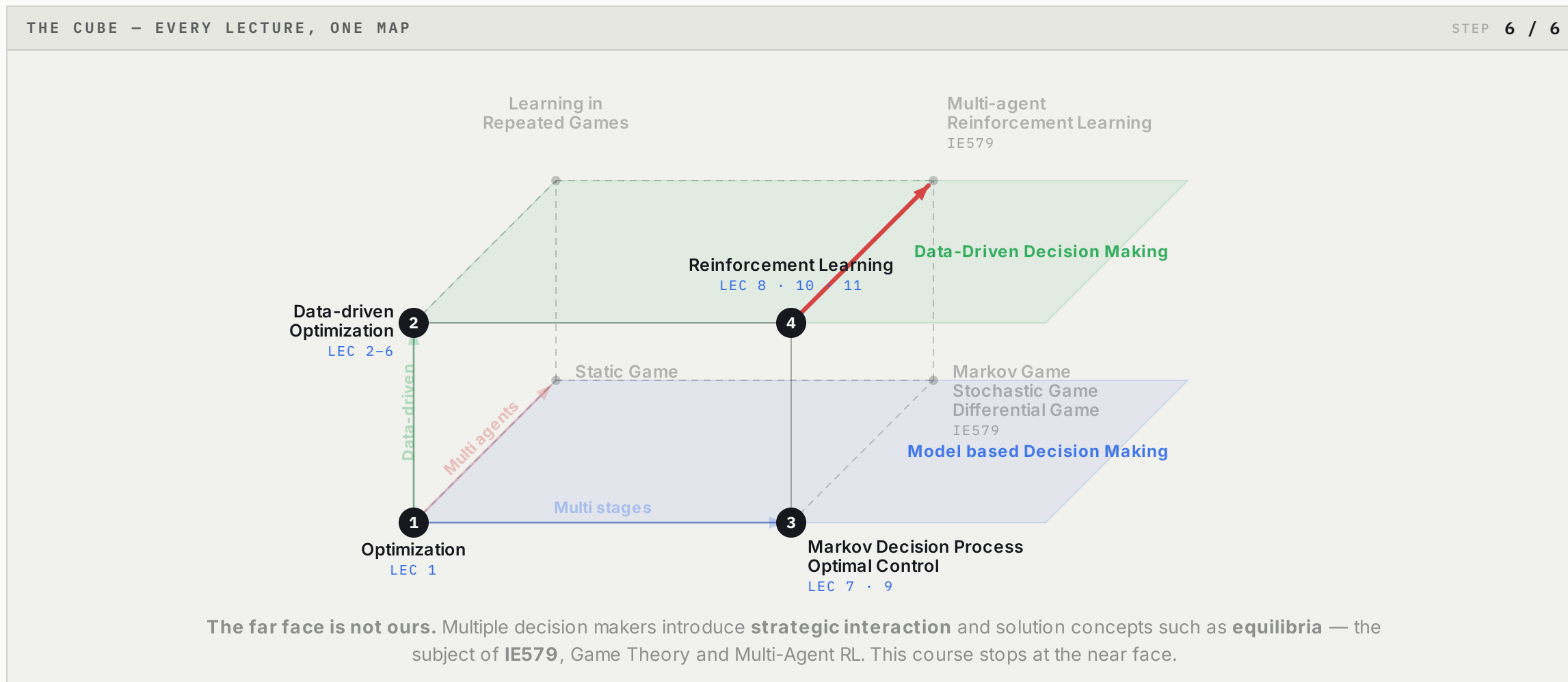
Where we are — policy, value and evaluation

QUESTION	CORE TOOLS	WHAT STILL NEEDS CHECKING
Which actions should the policy prefer?	TD3+BC and behavior restrictions	The trade-off between staying near the data and improving on it.
How should values guide improvement?	CQL and IQL	Their different objectives and the effect of conservatism.
Does the new policy perform better?	Off-policy evaluation	Coverage, behavior probabilities and estimation uncertainty.

Learning and evaluation answer different questions. A high predicted value alone does not establish an improvement over the behavior policy.

Model-based alternatives and sequence models remain in the appendix. Each adds its own modeling assumptions; none creates missing observations.

The tour, complete — and the face we did not visit



Lecture 0's cube, walked to its last cell: ⑤ [the interaction withdrawn](#). Step once more and no badge lights — **multi agents** is the crossing this course never makes.

Offline training learns from the actions already logged.

Use the fixed data carefully: distinguish supported actions from extrapolation, explain the learning objective, and evaluate the policy under stated assumptions. Additional model predictions are not additional measurements.

Questions?

The map has three axes. We studied static versus dynamic decisions and given versus learned models while keeping one decision maker. The remaining axis is **agents**: multiple decision makers introduce strategic interaction and solution concepts such as equilibria. That face belongs to **IE579 Game Theory and Multi-Agent Reinforcement Learning**. Everything you carry there was assembled here — a belief that data sharpens, a value you can only sample, a policy you can only nudge, and the discipline to distrust all three exactly where the evidence runs out.

— APPENDIX

Backup slides

Complete statements, kept out of the narrative.

Reading guide — three main responses to a fixed dataset

one main idea to explain, one comparison, one application

ROLE	READ OR REVISIT	QUESTION TO ANSWER
Core	Kumar et al., <i>Conservative Q-Learning for Offline Reinforcement Learning</i> (NeurIPS 2020)	How can the critic discourage actions unsupported by the data?
Compare	Fujimoto & Gu, <i>A Minimalist Approach to Offline Reinforcement Learning</i> (NeurIPS 2021) and Kostrikov et al., <i>IQL</i> (ICLR 2022)	What is constrained: the actor update, the value, or the action queries?
Apply	The logged-route stitching example and off-policy evaluation calculations	Can measured evidence support the claimed improvement?

TD3+BC, CQL and IQL form the main method comparison. BCQ/BEAR/BRAC are a compact family map; sequence models and learned-model methods extend the picture. Evaluation is part of the main lesson.

Backup 1— three regimes, one algorithm

	ON-POLICY	OFF-POLICY	OFFLINE
who collects the data	the policy being learned	some other policy, but we still act	a fixed log; we never act
an over-estimate is	corrected next episode	corrected once visited	never corrected
exploration	required	required	impossible
the design principle	on-policy correction	replay and stability	conservatism
representative method	SARSA, PPO	Q-learning, DQN, TD3	BCQ, CQL, IQL, MOPO

The middle column is where the confusion usually sits. Q-learning has always been off-policy, and Lecture 8 showed it converging happily while behaving ε -greedily. What made that work was not the algorithm but the clause "with every (s, a) visited infinitely often" in Watkins & Dayan's theorem. A finite log generally cannot supply fresh samples from every true transition law. Replaying every covered pair infinitely often can solve an empirical MDP; it does not remove uncertainty about the real MDP. A fully covered deterministic finite problem, such as the stitching example, can still be solved from a finite log. (*Levine, Kumar, Tucker & Fu, 2020*)

Backup 2 — the CQL objective, term by term

$$\min_Q \max_{\alpha \geq 0} \alpha \left(\mathbb{E}_{s \sim D, a \sim \mu(\cdot|s)}[Q(s, a)] - \mathbb{E}_{(s,a) \sim D}[Q(s, a)] - \tau \right) + \frac{1}{2} \mathbb{E}_{(s,a,s') \sim D} \left[(Q(s, a) - \hat{\mathcal{B}}^\pi \hat{Q}(s, a))^2 \right]$$

Choosing μ . Leaving μ free and adding an entropy regulariser $\mathcal{R}(\mu) = \mathcal{H}(\mu)$ gives $\mu^*(a | s) \propto \exp Q(s, a)$, and the penalty collapses to the closed form used in practice — **CQL(H)**:

$$\alpha \mathbb{E}_{s \sim D} \left[\log \sum_a \exp Q(s, a) - \mathbb{E}_{a \sim \hat{\beta}}[Q(s, a)] \right]$$

The first term is a soft max over all actions; the second is the empirical behaviour average. For discrete actions, the difference is $H(\hat{\beta}) + D_{\text{KL}}(\hat{\beta} \parallel \text{softmax}(Q))$. Its minimum is $H(\hat{\beta})$, not generally zero. A dominant penalty encourages softmax scores to match behavior frequencies; greedy extraction then favors the mode.

The Lagrangian form. The $\max_\alpha$ with a budget τ is Lecture 5's move again: τ is read in the units of the value function and is interpretable on the value scale, but is not automatically transferable across tasks.

Why the counter-term is not optional. The counter-term protects supported actions from excessive pessimism. With fixed Bellman targets, squared error anchors observed values; a linear pessimism term alone does not imply that every Q value diverges downward.

Backup 3 — expectile regression, and why $\tau \rightarrow 1$ is an in-support max

For a random variable X , the τ -expectile m_τ is the minimiser of $\mathbb{E}[L_2^\tau(X - m)]$ with $L_2^\tau(u) = |\tau - \mathbf{1}\{u < 0\}|u^2$. Setting the derivative to zero,

$$\tau \mathbb{E}[(X - m_\tau)_+] = (1 - \tau) \mathbb{E}[(m_\tau - X)_+]$$

so $m_{1/2} = \mathbb{E}[X]$, and for bounded X , as $\tau \rightarrow 1$ the expectile approaches $\text{ess sup } X$. IQL applies this with $X = Q(s, a)$, $a \sim \hat{\beta}(\cdot | s)$ — hence the supremum is over the **support of the behaviour policy at s** , never over the whole action set.

The two losses, in order.

repeat:

```
V ← argmin_V E_D[ L_2^τ ( Q(s,a) - V(s) ) ]           in-data actions only
Q ← argmin_Q E_D[ ( r + γ V(s') - Q(s,a) )^2 ]         target uses V(s'), not max_a
π ← argmax_π E_D[ exp( κ (Q(s,a) - V(s)) ) log π(a|s) ]
```

Note what is absent from every line: an action that is not in D . IQL is the only method in this lecture whose Q -network is never evaluated at an out-of-distribution input during training, which is why it needs no explicit conservatism term at all — and why its ceiling is set by how much β varied its actions.

Backup 4 — the OPE estimators, and where each one breaks

Let $\rho_t = \pi(a_t | s_t) / \beta(a_t | s_t)$ and $W = \prod_t \rho_t$. Assume target support is covered and environment dynamics are shared.

ESTIMATOR	MAIN CALCULATION	PRACTICAL LIMITATION
IS	average WG	unbiased with correct ratios; variance can be very large
PDIS	weight each reward by its prefix ratios	avoids unnecessary future ratios; can still have high variance
WIS	$\sum_j W^j G^j / \sum_j W^j$	finite-sample bias; direction varies; few weights can dominate
DR	model estimate + weighted TD residuals	correct ratios or exact Q can preserve unbiasedness with appropriate fitting; no universal variance cure
FQE	regress on $r + \gamma \mathbb{E}_{a' \sim \pi} Q(s', a')$	depends on coverage, horizon, function class, and fitting

Estimated behavior probabilities add error. Rare large weights may be absent from a small test sample, making the apparent error misleadingly small. Estimator disagreement is a diagnostic, **not a confidence interval**.

WIS stays within the observed return range when its denominator is positive; that does not imply closeness to the true policy value. Detailed IS, PDIS, and DR formulas are in Act 4.

— EXTENSION

Extensions — guarantees, alternative models and OPE estimators

Read after completing the main route.

What the CQL lower-bound statement actually covers

The theoretical result concerns a **policy value**, averaged over that policy's actions:

$$\hat{V}^{\pi}(s) = \mathbb{E}_{a \sim \pi}[\hat{Q}(s, a)] \leq V^{\pi}(s).$$

It requires the theorem's coverage, estimation-error, update, and sufficiently large penalty assumptions. The practical neural-network loss does not automatically certify those assumptions.

WHAT IT CAN SAY

Under the stated assumptions, the estimated value of the evaluated policy is conservative.

WHAT IT DOES NOT SAY

Every action value is a lower bound; any extracted policy is safe; or every trained network has the guarantee.

Keep theoretical scope, training objective, and measured performance distinct, just as we did for COMs in Lecture 5. (*Kumar et al., 2020, Theorem 3.2*)

Policy, value — and model

Lecture 11 handed us a learned dynamics model $\hat{P}$. Offline, its bias becomes acute for the same reason everything else does: a rollout that leaves the data is never contradicted. So penalise the reward by the model's own uncertainty and plan in the penalised MDP:

$$\tilde{r}(s, a) = r(s, a) - \lambda u(s, a)$$

MOPO (YU ET AL., 2020)

$u(s, a)$ is the maximum standard deviation across a bootstrapped ensemble of dynamics models — Lecture 5's `ensemble-alarm`, now measuring disagreement about *where you will end up* rather than about *how good it is*. A theoretical lower bound requires a valid model-error bound; empirical ensemble uncertainty is a proxy, not an automatic certificate.

MOREL (KIDAMBI ET AL., 2020)

Harder-edged: an *unknown state-action detector* partitions the space, and every pair it flags is routed to an absorbing state with the worst possible reward. Planning then avoids the unknown region because the model says it is a cliff.

Constrain the policy · constrain the value · constrain the model. Three places to put the same instinct — **do not trust a model where the data is thin** — and the set is now complete.

Or drop the Bellman equation entirely

Every method so far has kept Bellman and defended it. The last family does not keep it.

DECISION TRANSFORMER (CHEN ET AL., 2021)

Model the trajectory as a sequence: $\hat{R}_1, s_1, a_1, \hat{R}_2, s_2, a_2, \dots$, where $\hat{R}_t = \sum_{t' \geq t} r_{t'}$ is the **return-to-go**. Train a causal transformer to predict a_t . At test time, *condition* on the return you want and let it autoregress.

It avoids Bellman bootstrapping and its particular feedback loop; training and generalization can still fail. The failure mode moves instead: ask for a return the data never achieved and it will confabulate.

DIFFUSER (JANNER ET AL., 2022)

Go further: learn a diffusion model over **whole trajectories** and generate one, guided by a reward gradient. Planning becomes sampling; the learned distribution guides the proposal, but feasibility and dynamics consistency are not guaranteed for free.

This is Lecture 6 arriving in the dynamic world. There, *search a forward model* (Lec 5) sat opposite *sample an inverse model* (Lec 6), and the spine promised the pair would return as value against policy. Here it returns a third time, in its sharpest form: **learn Q and search it, or learn $p(\tau \mid \text{return})$ and draw from it**. Conditioning on a desired outcome and sampling a design is exactly what Lecture 6's **condition-shift** did — including the way the samples thin out as the condition leaves the data.

What a practitioner actually runs

DOUBLY ROBUST (JIANG & LI, 2016; THOMAS & BRUNSKILL, 2016)

$$\begin{aligned} W_t &= \prod_{t' \leq t} \rho_{t'}, \\ \delta_t &= r_t + \gamma \hat{V}(s_{t+1}) - \hat{Q}(s_t, a_t), \\ \hat{V}_{\text{DR}} &= \hat{V}(s_1) + \sum_t \gamma^{t-1} W_t \delta_t. \end{aligned}$$

With support and appropriate independent fitting, DR is unbiased if the ratios are correct or the relevant Q model is exact. Weights multiply **Bellman residuals**; an accurate model can reduce their variance.

FITTED Q EVALUATION (LE, VOLOSHIN & YUE, 2019)

Regress Q^π directly: $Q \leftarrow r + \gamma \mathbb{E}_{a' \sim \pi}[Q(s', a')]$, fitted on D . FQE avoids products of trajectory ratios, but its error still depends on horizon, coverage, function approximation, and fitting. More data can reduce estimation error; misspecification or missing support can leave irreducible error.

THE HONEST VERDICT

and the reason to report more than one number

Compare estimators and inspect coverage and importance weights. Their **disagreement is a diagnostic, not a confidence interval**. Any reported uncertainty interval needs an explicit construction and its assumptions; agreement alone cannot establish accuracy.

In an industrial deployment this act comes *first*. Before anyone asks whether to use CQL or IQL, someone asks: if we hand you six months of logs and a proposed controller, **can you tell us whether it is better than the one we are running?** The course has not addressed that question until now, and it is the question that gates the rest.