

Optimal Control & Planning

The control-theoretic origin of sequential decision making

Jinkyoo Park · KAIST

Lecture 7 solved the dynamic optimum with operations research. *This is the other tradition.*

Where we are — the other column of the map

PREVIOUS STEP	THIS CHAPTER'S QUESTION
Lecture 7 introduced Bellman reasoning for sequential decisions.	Now use continuous state/action models to compute feedback.

Control and dynamic programming overlap. We compare mathematical tools and assumptions, not mutually exclusive academic families.

Compute a feedback gain and compare the guarantees of the three mathematical views.

Learning route — solve a small control problem first

Bring: Derivatives, matrix multiplication and the Bellman principle.

FIRST PASS	WHAT TO DO
Follow the idea	One-step control → LQR feedback → interpret HJB → interpret Pontryagin
Work without the solution	Compute a feedback gain and compare the guarantees of the three mathematical views.
Return later	Full HJB, Riccati and Pontryagin derivations are after the main route.

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

— ACT 1

It is the same problem

Q1. Control theory writes different symbols. Is it solving anything different?

Discrete-time optimal control — the problem, stated

Q1 Same problem?

Q2 LQR feedback?

Q3 HJB meaning?

Q4 Two views?

Choose a control to minimise an accumulated cost subject to deterministic dynamics:

$$L(u) = \sum_{k=0}^{K-1} g_k(x_k, u_k) + g_K(x_K), \quad x_{k+1} = f_k(x_k, u_k), \quad u_k \in U_k$$

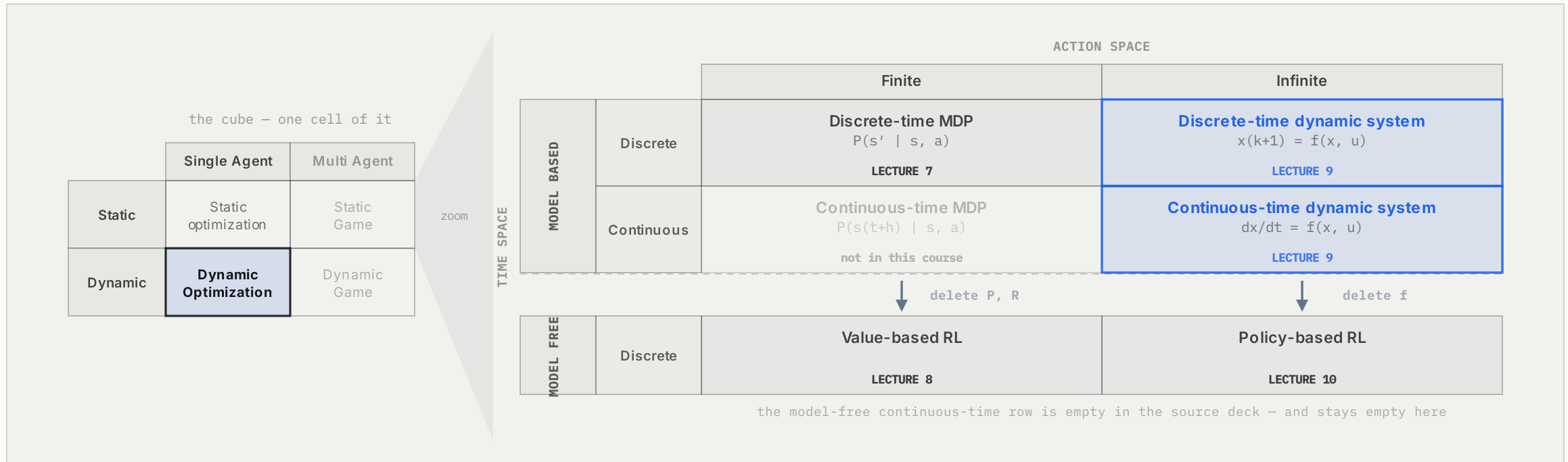
Read the third clause carefully, because it is the whole lecture in one symbol:

$$u_k = \gamma_k(x_k), \quad \gamma_k \text{ a permissible control strategy at stage } k$$

What we seek is not a sequence of numbers $u_0, \dots, u_{K-1}$ but a **rule** γ_k from state to control. Control theory insists on this from its very first slide, for the same reason Lecture 7 wanted a policy rather than a plan: a rule is still valid from wherever the system actually lands.

The same 2×2, one level down

Lecture 7 set this out as two tables — here it is drawn whole



The course cube puts Lectures 7 and 9 in **one cell**: single agent, dynamic, model-based. Zoom into that cell and two finer coordinates appear — how large the action set is, and whether time ticks or flows. **Lineage A is the finite column; lineage B is the infinite column**, and deleting the model is a single vertical arrow that acts on both at once.

Lecture 7 read this grid across the *columns*, to say what separates it from today. Read it down the *rows* instead and it says something Lecture 7 could not yet: the model axis is one move, made twice — and the empty cells are outside our syllabus, not empty research fields.

Dynamic programming, in control's notation

Define the cost-to-go from *any* state x at *any* time k :

$$V(k, x) = \min_{\gamma_k, \dots, \gamma_{K-1}} \left[\sum_{i=k}^{K-1} g_i(x_i, u_i) + g_K(x_K) \right], \quad x_k = x$$

BELLMAN'S PRINCIPLE OF OPTIMALITY

quoted as the source deck quotes it

An optimal policy has the property that whatever the initial state and initial decision are, **the remaining decisions must constitute an optimal policy** with regard to the state resulting from the first decision.

Applied directly, it gives the recursion — and it is Lecture 7's, letter for letter:

$$V(k, x) = \min_{u_k \in U_k} \left\{ g_k(x, u_k) + V(k+1, f_k(x, u_k)) \right\}$$

Each u_k^* is read off as the argument attaining the right-hand side — what the source deck calls a **single-shot optimisation**. That phrase is worth keeping: a problem that unfolds over time has been reduced to one static optimisation performed once per instant. We meet the move again in Act 2 and again in Act 4.

Where the expectation went

LECTURE 7 — THE BELLMAN BACKUP

$$V(s) = \max_a \sum_{s'} T(s, a, s') [R + \gamma V(s')]$$

- a **distribution** over successors,
- a **finite** set of actions to enumerate,
- **reward**, and max.

LECTURE 9 — THE DP RECURSION

$$V(k, x) = \min_u \{g_k(x, u) + V(k+1, f_k(x, u))\}$$

- **one** deterministic successor,
- a **continuum** of controls,
- **cost**, and min.

Two of those three differences are cosmetic. The expectation $\sum_{s'} P[\dots]$ collapses to a single point because the dynamics are deterministic — put the noise back and it returns unchanged. Cost versus reward is a sign.

Solve backward in time; optimal sub-plans compose into an optimal plan. **The logic is identical.**

One-step control — spend effort now to reduce the final error

Let $x_1 = x_0 + u$, $x_0 = 2$, and minimize $J(u) = u^2 + x_1^2$. There are no input constraints.

$$J(u) = u^2 + (2 + u)^2 = 2(u + 1)^2 + 2.$$

CONTROL	FINAL STATE	EFFORT COST	FINAL-STATE COST	TOTAL
0	2	0	4	4
-1	1	1	1	2
-2	0	4	0	4

The optimum is $u^* = -1$. Reaching zero immediately costs too much effort. For any initial state, the same calculation gives the **feedback rule** $u^*(x) = -x/2$.

Temperature thread — derive a feedback command

shared teaching example · predict before revealing the calculation

For a measured temperature error x , minimize the known one-step cost $c(u) = (x + u)^2 + u^2$. Allow a signed command for heating or cooling in this unconstrained version.

Predict: As the room gets colder, should the heating command increase? Find a rule valid for any x .

Calculate and check. $dc/du = 2(x + u) + 2u = 0$ gives $u^*(x) = -x/2$. At $x = -2$, the command is **1**. The rule is feedback: it recomputes the action from the current state.

This is a finite, one-step LQ problem. Its gain 1/2 need not equal the gain of an infinite-horizon problem.

Try it — the measured error differs from the planned error

work independently · reveal only after writing an answer

A previous prediction suggested the next error would be -1 , so a precomputed command was 0.5 . The next measurement instead gives $x = -1.5$. Using the same one-step objective, calculate the feedback command.

Check your answer. $u^* = -(-1.5)/2 = \mathbf{0.75}$. Reusing 0.5 ignores the new state measurement. Feedback revises the decision; it does not require the previous prediction to have been correct.

Distinguish an action sequence computed once from a state-to-action rule used after each measurement.

The key computational difference — optimize over a continuum

In Lecture 7 the inner $\max_a$ was a *loop*: try each of the finitely many actions, keep the best. Here U_k is a continuum, so the inner minimisation

$$\min_{u \in U_k} \underbrace{g_k(x, u) + V(k+1, f_k(x, u))}_{\text{a function of } u}$$

is not a loop at all. It is an optimisation problem — [Lecture 1's template, nested inside every backup](#), once per state and once per time step.

So the continuous action space is not a nuisance detail; it is the structural fact that makes this a different tradition. Everything that follows is a way of coping with it: first solve a *quadratic* problem explicitly (Act 2), then interpret the *smooth* Bellman condition (Act 3), or characterise its solution *along one trajectory* rather than everywhere (Act 4).

Same problem, same principle — but the backup now contains [an optimisation, not a loop](#).

Check — the same problem, other clothes

Optimal control minimises a cost $\int c(x, u) dt$ subject to dynamics $\dot{x} = f(x, u)$. Read as an MDP, what plays the role of the reward?

- ☒ A The negative of the cost, $-c(x, u)$ — the two traditions differ by a sign and a vocabulary
- ☐ B The dynamics f
- ☐ C The control u
- ☐ D The terminal constraint on $x(T)$

State is state, control is action, dynamics are the transition, and cost is negative reward. Operations research maximised a return; control theory minimised a cost; both are solving the same object. Keeping the translation table in view is what makes the second half of this course two parallel stories rather than two unrelated ones.

— ACT 2

Compute linear-quadratic feedback

Q2. Linear dynamics, quadratic cost: a useful case whose value function can be represented by a matrix.

LQR — quadratic cost, linear feedback

Use a finite horizon, linear dynamics and unconstrained controls. Assume $Q, Q_f \succeq 0$ and $R \succ 0$.

$$\dot{x} = Ax + Bu, \quad J = \int_0^T (x^\top Qx + u^\top Ru) dt + x(T)^\top Q_f x(T).$$

The optimal value and feedback law take the form

$$V(t, x) = x^\top P(t)x, \quad u^*(t, x) = -K(t)x, \quad K(t) = R^{-1}B^\top P(t).$$

OBJECT	HOW TO READ IT
$P(t)$	Future cost attached to the current state; compute backward from $P(T) = Q_f$.
$K(t)$	Feedback gain: turn the measured state into a control command.

The Riccati equation computes P . First verify a scalar feedback law by ordinary calculus; the full matrix derivation follows the HJB discussion in the appendix.

Read the gain with numbers — a scalar infinite-horizon example

Let $\dot{x} = u$ and $J = \int_0^\infty (x^2 + u^2) dt$. Consider stabilizing controls with $x(t) \rightarrow 0$.

Complete a square: since $d(x^2)/dt = 2xu$,

$$x^2 + u^2 = (u + x)^2 - \frac{d(x^2)}{dt}.$$

Integrating gives a lower bound that we can attain:

$$J = x(0)^2 + \int_0^\infty (u + x)^2 dt \geq x(0)^2.$$

Choose $u = -x$. The square is zero, so this control attains the bound. Starting from $x(0) = 2$, the state is $x(t) = 2e^{-t}$ and the optimal cost is **4**.

The gain is 1: error 2 gives command -2 , and error 0.5 gives command -0.5 . This infinite-horizon continuous-time problem differs from the earlier one-step heater problem.

Conditions behind the tidy answer

CAN WE CONTROL UNSTABLE MOTION?

For $x_{t+1} = 2x_t + 0u_t$, $x_0 = 1$, and state cost x_t^2 , every control has infinite cost.

Stabilizable means every unstable mode can be controlled. Full controllability is stronger than necessary.

DOES THE COST SEE UNSTABLE MOTION?

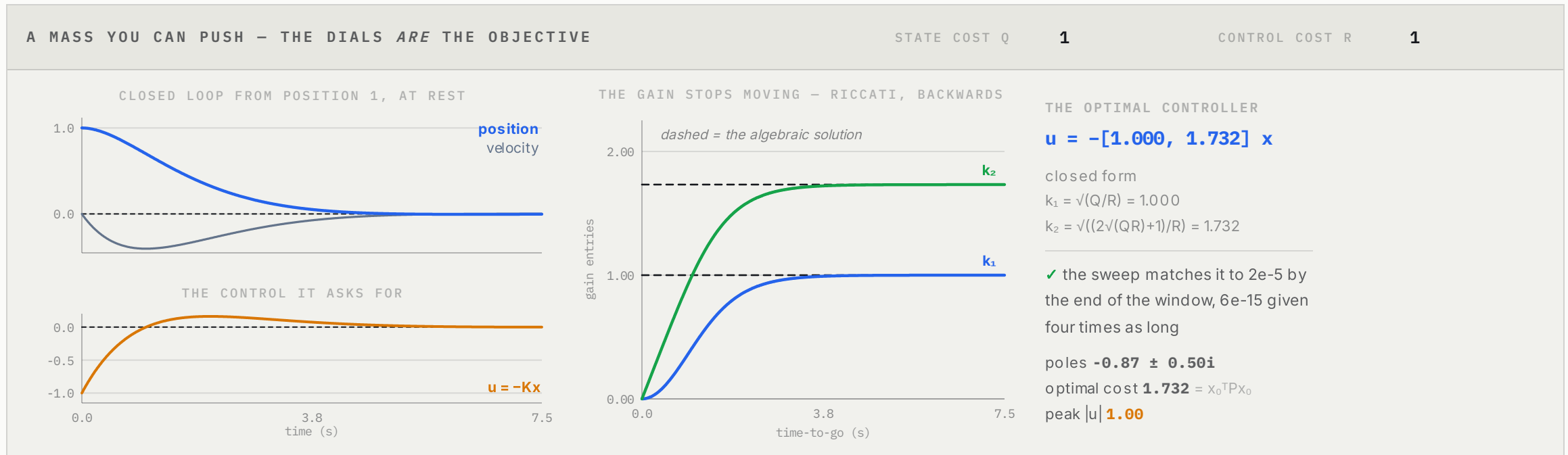
Detectable means an unstable mode cannot remain invisible to the state penalty Q .

Together with $Q \succeq 0$, $R \succ 0$, this gives the standard stabilizing infinite-horizon LQR result.

Positive R makes the unconstrained minimization strictly convex. If R is singular, a minimum may still exist, but the displayed inverse formula does not apply. Input constraints also change the solution.

LQR, run

a double integrator — position and velocity as the state, force as the input



$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ — a mass you can push. Turn the dials and read the gain. The Riccati integration on the right **flattens onto the algebraic solution**, which is drawn as a dashed line from the closed form $k_1 = \sqrt{q/r}$, $k_2 = \sqrt{(2\sqrt{qr} + 1)/r}$. Cheap control (R small) buys speed with a violent input; expensive control is gentle and slow. There is no tuning here — the dials *are* the objective.

Why LQR is the cornerstone

- **Globally optimal, and linear.** For the stated unconstrained LQ problem, the Riccati solution is globally optimal. Computing that matrix usually requires numerical integration or an algebraic solver.
- **A useful local approximation.** Linearise any smooth problem about a trajectory, take a quadratic approximation of the cost, solve the resulting LQR, re-linearise, repeat: that is **iLQR / DDP**, and it is the backbone of trajectory optimisation and of model-based planning in Lecture 11. (*Jacobson & Mayne, 1970*)
- **The target the data-driven world reaches for.** In Lecture 10, DDPG's learned actor $\mu_\theta(s)$ occupies exactly the place of this gain K — for a system whose A and B are **unknown**.

Hold $u = -Kx$ in view: it is a known-model feedback law to compare with a learned actor. Policy-gradient methods optimize a chosen policy class from experience; they do not automatically recover this LQR optimum.

Check — why LQR is the exception that matters

For linear dynamics and a quadratic cost, the optimal controller is $u = -Kx$, with K from the Riccati equation. Why does this single special case deserve a whole act?

- ☐ A Because most real systems are exactly linear
- ☐ B Because it is the only case where the HJB equation has a solution
- ☒ C Because in this case the value function collapses to a finite matrix representation — and it is the local model that non-linear methods iterate on
- ☐ D Because quadratic costs are the only ones that are convex

Almost nothing is truly linear-quadratic. What LQR gives is a **solvable local approximation**: linearise the dynamics and take a quadratic expansion of the cost around the current trajectory, solve exactly, step, and repeat. That is iLQR, and it is how the closed form earns its place — as the inner loop of methods for problems that are not LQ at all.

— ACT 3

Interpret continuous-time Bellman reasoning

Q3. Let the time step shrink to nothing. What does the Bellman equation become?

Continuous time — the problem, and the cost-to-go

Q1 Same problem?

Q2 LQR feedback?

Q3 HJB meaning?

Q4 Two views?

Time no longer ticks; it flows.

$$L(u) = \int_0^T g(t, x(t), u(t)) dt + q(T, x(T)), \quad \dot{x}(t) = f(t, x(t), u(t)), \quad x(0) = x_0$$

with $u(t) = \gamma(t, x(t)) \in U$ and $\gamma \in \Gamma$, the class of **admissible feedback strategies**.

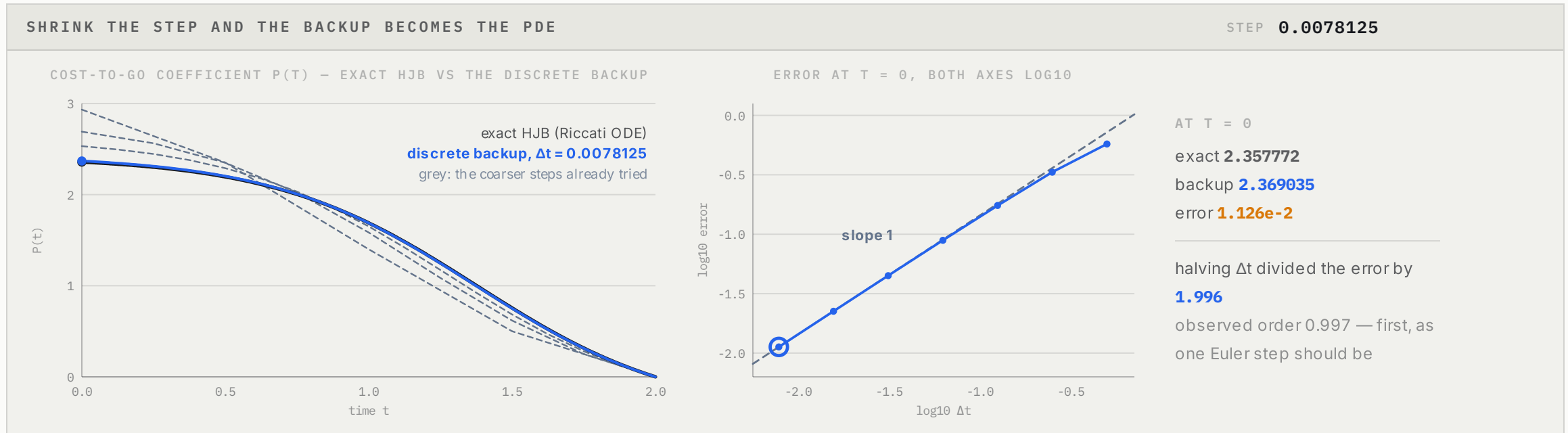
The cost-to-go is defined exactly as before, over the remaining interval:

$$V(t, x) = \min_{u(s), t \leq s \leq T} \left[\int_t^T g(s, x(s), u(s)) ds + q(T, x(T)) \right], \quad V(T, x) = q(T, x)$$

The boundary condition is the terminal cost. In Lecture 7 the recursion was seeded at $V(K, \cdot) = g_K$; here it is seeded at $t = T$. Same seed, same backward direction.

The claim, checked

shrink Δt on a discrete backup and watch it land on the PDE



A scalar problem whose HJB solution is known exactly: $\dot{x} = x + u$, cost $\int_0^2 (x^2 + u^2) dt$, so $V(t, x) = P(t)x^2$ with P solving a Riccati ODE. The *discrete* backup of Act 1 is run on the Euler-discretised system at step Δt and its P_k plotted against the exact curve. Halve Δt and **the error halves** — first order, exactly as an Euler step should be.

What HJB buys — a certificate, not just a formula

SUFFICIENT, AND GLOBAL

Find a continuously differentiable V satisfying **the PDE and terminal condition**, and an admissible control attaining its minimum: the verification argument proves optimality — and the optimal control falls out of a **static, pointwise** minimisation:

$$u^*(t) = \arg \min_{u \in U} \left\{ \partial_x V f(t, x, u) + g(t, x, u) \right\}$$

FEEDBACK, EVERYWHERE

V is a field over the *whole* state space, so the rule it generates is credible from **any state the system actually reaches** — not only the one you predicted.

And it extends to stochastic dynamics with one extra term, $\frac{1}{2} \text{tr}(\sigma \sigma^\top \partial_{xx} V)$.

"Sufficient" is a strong word and it is earned: a solution of the PDE is a *certificate* that every other admissible trajectory costs at least as much. The four-line verification argument is Backup 2, and it is the only place in this lecture where something is proved optimal rather than merely derived.

And what it costs

HJB is a **nonlinear PDE over the entire state space**.

Three prices, and each is a limitation we will spend the rest of the course working around:

- **It needs the model.** f and g appear explicitly inside the minimisation. Delete them and there is no equation left to solve — which is Lecture 10.
- **It assumes V is differentiable.** Often it is not; the honest fix is a weaker notion of solution (viscosity solutions), which buys existence at the cost of the clean formula. (*Crandall & Lions, 1983*)
- **It does not survive dimension.** A value *field* over $x \in \mathbb{R}^n$ is exponentially large — the same curse Lecture 7's table faced, now with the table replaced by a continuum.

So HJB is the **characterisation**, rarely the computation. LQR is a particularly useful solvable case: a quadratic value function closes the equation. Other special problems also admit explicit solutions.

Check — what happens to Bellman in continuous time

Shrink the time step $\Delta t \rightarrow 0$ in the discrete Bellman equation. What do you get?

- ☐ A The Riccati equation
- ☒ B The Hamilton–Jacobi–Bellman equation — a partial differential equation in the value function
- ☐ C Pontryagin's maximum principle
- ☐ D The Euler–Lagrange equation

The recursion becomes a PDE. HJB is the Bellman equation with the discrete step taken to its limit, and it inherits both the strength — a sufficient condition for global optimality — and the weakness, since solving a PDE over a high-dimensional state space is exactly as hard as the curse of dimensionality suggests. The earlier LQR example avoids a state-space grid because its value has a quadratic form. The next section studies conditions along one trajectory instead.

— ACT 4

The other view

Q4. HJB carries a value field over every state. Is there a way in that follows only one trajectory?

Adjoin the dynamics — the Hamiltonian and the costate

Q1 Same problem?

→ Q2 LQR feedback?

→ Q3 HJB meaning?

→ Q4 Two views?

HJB is the *Eulerian* view: stand still and watch a field $V(t, x)$ over all states. There is a *Lagrangian* view: ride the optimal trajectory and enforce the dynamics as a constraint along it.

This is Lecture 1's Lagrangian move, applied once per instant. Adjoin $\dot{x} = f$ with a multiplier $\lambda(t)$ — the **costate** — and collect the terms into the **Hamiltonian**:

$$H(t, x, u, \lambda) = g(t, x, u) + \lambda^\top f(t, x, u)$$

One multiplier per instant, and a whole function $\lambda(\cdot)$ instead of a vector. The constraint set of Lecture 1 has become the dynamics themselves.

The necessary conditions

For a smooth, fixed-horizon problem with free terminal state and no state constraints, the **normal** minimum principle gives these necessary conditions:

$$\dot{x}^* = \frac{\partial H}{\partial \lambda} = f, \quad \dot{\lambda}^* = -\frac{\partial H}{\partial x}, \quad \lambda(T) = \frac{\partial q}{\partial x} \Big|_{x^*(T)}, \quad u^* = \arg \min_{u \in U} H(t, x^*, u, \lambda^*)$$

Read the shape of it. The state runs **forward** from $x(0) = x_0$; the costate runs **backward** from $\lambda(T)$; and they are coupled. This is a **two-point boundary-value problem**, not an initial-value problem — you cannot simply integrate.

And the last condition is the familiar move again: dynamic optimisation reduced to **a static minimisation of H , freshly at each instant.**

What the costate actually is

Under the smooth optimal-control assumptions, the costate equals the value gradient along an optimal trajectory:

$$\lambda(t) = \frac{\partial V}{\partial x}(t, x^*(t)).$$

VIEW	WHAT IT CARRIES
HJB	The value function over time and state; its gradient is available wherever the smooth solution is known.
Pontryagin	State and costate along one candidate trajectory, with endpoint conditions.

For the scalar quadratic value $V(t, x) = P(t)x^2$, the costate is $\lambda = 2P(t)x$. At the infinite-horizon solution $P = 1$, a small state change Δx changes the value by approximately $2x\Delta x$.

The costate measures sensitivity to the **state**. It is not automatically the gradient with respect to a policy's parameters. A Pontryagin candidate still needs the stated optimality checks.

What does a costate measure? Use the same scalar example

For $\dot{x} = u$ and $V(x) = x^2$, the costate is $\lambda = V_x = 2x$.

At $x = 2$, $\lambda = 4$: increasing the state by 0.01 raises optimal future cost by approximately $4(0.01) = 0.04$. The exact change is $2.01^2 - 2^2 = 0.0401$.

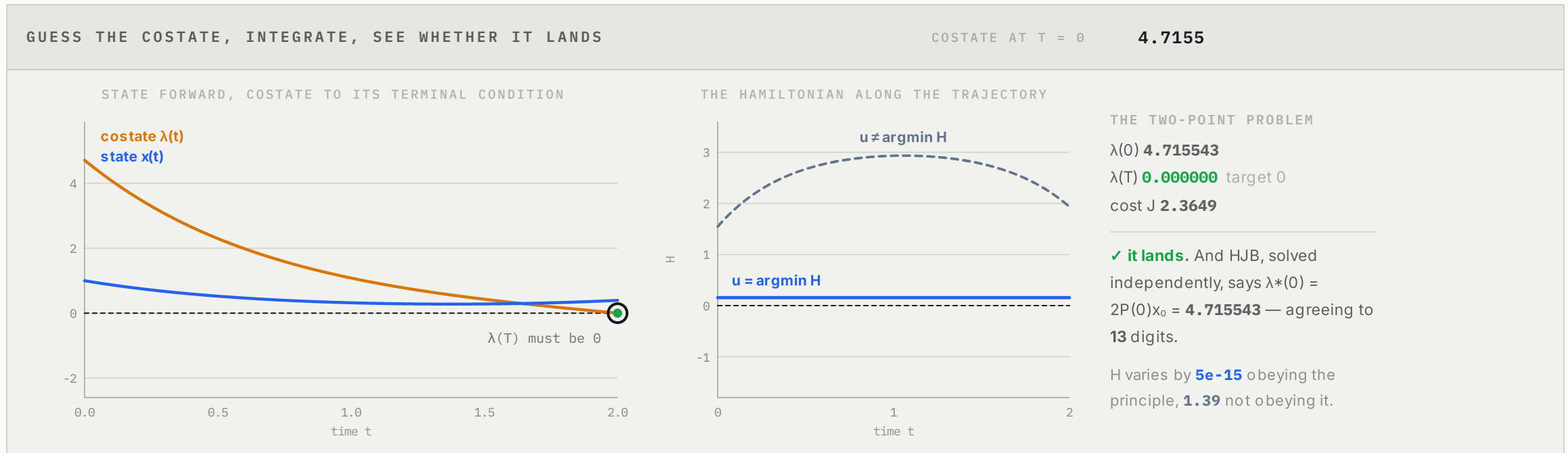
$$H = x^2 + u^2 + \lambda u, \quad H_u = 2u + \lambda = 0.$$

Thus $u = -\lambda/2 = -x$, the same feedback law as before.

A costate is a **marginal future cost**. Pontryagin carries this sensitivity along a trajectory instead of solving for the whole value field.

Shooting — two views, one optimum

guess the costate, integrate, and see whether it lands



The same problem the HJB widget solved: $\dot{x} = x + u$, $\int_0^2 (x^2 + u^2) dt$. Guess $\lambda(0)$, integrate the coupled equations forward, and check whether $\lambda(2) = 0$ as the transversality condition demands. Exactly one guess lands — and it is $\lambda^*(0) = 2P(0)x_0$, the value gradient from the *other* method. The dashed trace is a control that does *not* minimise H pointwise — and its Hamiltonian, conserved to fourteen digits when the principle is obeyed, drifts.

One scalar system, three views of the same answer

teaching comparison · continuous-time infinite-horizon LQR

Use $\dot{x} = u$, cost $\int_0^\infty (x^2 + u^2) dt$, and the stabilizing infinite-horizon solution.

VIEW	CALCULATION	ANSWER
HJB	Try $V(x) = Px^2$: $0 = \min_u (x^2 + u^2 + 2Pxu)$	$u = -Px$, then $1 - P^2 = 0$
LQR / Riccati	$A = 0, B = 1, Q = R = 1$ gives $1 - P^2 = 0$	choose stabilizing $P = 1$
Pontryagin	$H = x^2 + u^2 + \lambda u, H_u = 0$	$u = -\lambda/2$; along this solution $\lambda = 2x$

Thus all three are consistent with $u = -x$, $x(t) = x_0 e^{-t}$ and $V(x_0) = x_0^2$. At $x_0 = 2$, the total cost is **4**.

Agreement here comes from the LQ structure and the stabilizing solution. Pontryagin's conditions alone do not certify an arbitrary nonlinear candidate as globally optimal. Compare the mathematical guarantees as well as the answer.

HJB versus Pontryagin — distinguish what each certifies

	HJB (DYNAMIC PROGRAMMING)	PONTRYAGIN (MINIMUM PRINCIPLE)
view	Eulerian — a value field $V(t, x)$	Lagrangian — one trajectory $x^*(t)$
logic	sufficient , over all (t, x)	necessary , along the optimum
output	a feedback law $\gamma(t, x)$	an open-loop $u^*(t)$, from a costate ODE
mathematics	a nonlinear PDE	a two-point boundary-value ODE
stochastic	second-order HJB	stochastic maximum principles exist; beyond this lecture
cost	the curse of dimensionality	one trajectory; necessity alone does not certify a minimum

Feedback beats an open-loop plan for the reason Lecture 7 already gave: $\gamma(t, x)$ is valid from *any* state you actually reach, while $u^*(t)$ is valid only along the trajectory you predicted. But Pontryagin scales where HJB cannot, which motivates trajectory methods, alongside direct optimization approaches. Its machinery returns in Lecture 10 — REINFORCE is a trajectory view — and in Lecture 11, where a learned policy imitates an iLQR teacher.

Check — the parallel about to be drawn

Lecture 8 deleted P and R from the MDP and got value-based RL. Lecture 10 deletes what from optimal control?

- ☐ A The cost function c
- ☐ B The state x
- ☐ C The horizon T
- ☒ D The dynamics f — leaving a controller that must be improved from experience alone

The two lineages lose their model in exactly the same way, one lecture apart. Delete P from dynamic programming and you must sample the expectation; delete f from optimal control and you can no longer differentiate through the rollout, so you must estimate the gradient from trajectories instead. Same deletion, two different repairs — and Lecture 11 gets the model back.

— CLOSING

Closing

Both model-based origins are now on the table. Each one leaned entirely on knowing f .

Where we are — three tools, different guarantees

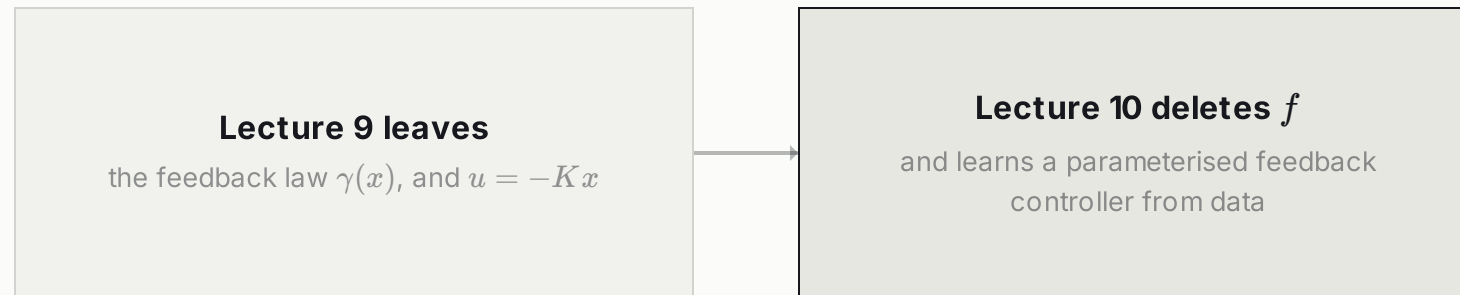
	LINEAGE A • OR / DYNAMIC PROGRAMMING	LINEAGE B • CONTROL THEORY
MODEL-BASED (ORIGIN)	MDP & Dynamic Programming LECTURE 7	Optimal Control / Planning LECTURE 9
DATA-DRIVEN (EXTENSION)	Value-Based RL LECTURE 8	Policy-Based RL LECTURE 10

Three mathematical tools with different roles; numerical solutions still need checking:

	WHAT IT GIVES	WHAT IT NEEDS
LQR	linear feedback $u = -Kx$; compute the Riccati matrix	the stated unconstrained LQ model and horizon assumptions
HJB	a value over (t, x) and an optimal feedback law	dynamics, cost, terminal condition and a smooth admissible solution
Pontryagin	necessary conditions for candidate trajectories	f , g , and $\partial_x f$

HJB can verify an admissible solution; LQR solves its special case; Pontryagin provides necessary conditions. **All use the dynamics.**

What we hand on



Lecture 10 follows the same teaching move as Lecture 8: keep the decision objective, but learn from experience when a complete model is unavailable. The linear gain K is one feedback policy. A network $\mu_\theta(s)$ is a more general policy class trained using sampled performance; it is not guaranteed to recover the LQR solution or a global optimum.

A learned actor can avoid a fresh continuous action search at every step. The difficulty moves into training and evaluating that actor.

Bellman and optimal control study the same sequential decision structure.

HJB can verify a value function and feedback law; LQR solves a structured special case; Pontryagin supplies necessary conditions for candidate trajectories. Their guarantees and computational requirements differ.

Questions?

Two traditions, one destination. Bellman wrote the same equation twice — once as a sum over a table, once as a partial differential equation — and LQR gives a central case where the value field reduces to a matrix.

Everything after this is what happens when you are no longer told f .

— APPENDIX

Backup slides

Complete arguments, kept out of the narrative.

Reading guide — solve the same system using different mathematical tools

one main idea to explain, one comparison, one application

ROLE	READ OR REVISIT	QUESTION TO ANSWER
Core	Tedrake, <i>Underactuated Robotics</i>: selected LQR sections and this lecture's Bellman/HJB derivation	Why does the linear-quadratic special case admit a tractable controller?
Compare	HJB verification, Riccati/LQR and Pontryagin necessary conditions	Which method gives a value function, a gain, or candidate trajectory conditions?
Apply	One scalar system solved three ways, then the original control examples	Do the methods agree under the stated LQ assumptions?

The same small problem is the main comparison. HJB and Pontryagin derivations are advanced reading; their historical importance does not make every technical step a prerequisite for computing feedback.

Backup 1—the HJB equation, derived in one step

Start from Bellman's optimality principle over a short interval δ :

$$V(t, x) = \min_{u \in U} \left\{ g(t, x, u) \delta + V(t + \delta, x + f(t, x, u) \delta) \right\} + o(\delta)$$

Taylor-expand the second term about (t, x) :

$$V(t + \delta, x + f\delta) = V(t, x) + \frac{\partial V}{\partial t} \delta + \frac{\partial V}{\partial x} f(t, x, u) \delta + o(\delta)$$

Substitute, cancel $V(t, x)$ from both sides — it is free of u , so it passes through the minimum — divide by δ and let $\delta \rightarrow 0$:

$$0 = \min_{u \in U} \left\{ g(t, x, u) + \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(t, x, u) \right\} \implies -\frac{\partial V}{\partial t} = \min_{u \in U} \left\{ \frac{\partial V}{\partial x} f + g \right\} \quad \blacksquare$$

Compare Lecture 7's derivation of the Bellman optimality equation: identical, with $\delta \rightarrow 0$ in place of a one-step sum. The minimising u at each (t, x) is the optimal feedback — a static pointwise minimisation embedded inside a PDE for V .

Backup 2 — why HJB is **sufficient**: the verification argument

Suppose V is continuously differentiable and satisfies HJB. Take any admissible $\gamma \in \Gamma$, with trajectory x , alongside an admissible minimizing candidate γ^* with x^* , both at the same fixed terminal time T . Also require $V(T, x) = q(T, x)$. Because the HJB right-hand side is a *minimum* over u , the arbitrary control can only do worse, while the minimising control attains it exactly:

$$g(t, x, u) + \partial_x V f(t, x, u) + \partial_t V \geq 0, \quad g(t, x^*, u^*) + \partial_x V f(t, x^*, u^*) + \partial_t V \equiv 0$$

Along a trajectory the last two terms of each are exactly $\frac{d}{dt} V(t, x(t))$, so integrating the first over $[0, T]$ and the second over $[0, T]$ gives

$$\int_0^T g \, dt + V(T, x(T)) - V(0, x_0) \geq 0, \quad \int_0^T g^* \, dt + V(T, x^*(T)) - V(0, x_0) = 0$$

Eliminate $V(0, x_0)$ between them and apply the boundary condition $V(T, x) = q(T, x)$:

$$L(u) = \int_0^T g \, dt + q(T, x(T)) \geq \int_0^T g^* \, dt + q(T, x^*(T)) = L(u^*) \quad \blacksquare$$

A solution of the PDE is therefore a **certificate**: no admissible strategy beats γ^* , from any initial state. This is exactly what Pontryagin's conditions do *not* give — they are stationarity conditions, satisfied by every extremal, optimal or not.

Backup 3 — LQR: the Riccati derivation, both times

Continuous time. $\dot{x} = Ax + Bu$, running cost $g = x^\top Qx + u^\top Ru$. Guess $V(t, x) = x^\top P_t x$, so $\partial_x V = 2P_t x$ and $\partial_t V = x^\top \dot{P}_t x$. HJB reads

$$-x^\top \dot{P}_t x = \min_u \left\{ 2x^\top P_t (Ax + Bu) + x^\top Qx + u^\top Ru \right\}$$

Minimise over u : $2B^\top P_t x + 2Ru = 0 \Rightarrow u^* = -R^{-1}B^\top P_t x$. Substitute back and match $x^\top(\cdot)x$ for all x :

$$-\dot{P}_t = A^\top P_t + P_t A - P_t B R^{-1} B^\top P_t + Q$$

Infinite horizon: $\dot{P} \rightarrow 0$, giving the algebraic Riccati equation $A^\top P + PA - PBR^{-1}B^\top P + Q = 0$ under the stabilizability and detectability assumptions above, with a unique PSD stabilizing solution, hence the constant gain $K = R^{-1}B^\top P$ and $u^* = -Kx$.

Discrete time. With $x_{t+1} = Ax_t + Bu_t$ and $V(z) = z^\top Pz$, the backup is $z^\top Pz = \min_w \{z^\top Qz + w^\top Rw + (Az + Bw)^\top P(Az + Bw)\}$, whose minimiser is $w^* = -(R + B^\top PB)^{-1}B^\top PAz$. Substituting and matching for all z :

$$P = Q + A^\top PA - A^\top PB(R + B^\top PB)^{-1}B^\top PA, \quad K = (R + B^\top PB)^{-1}B^\top PA$$

Under stabilizability and detectability, the ARE has a unique positive-semidefinite stabilizing solution. Started from $P_1 = Q$, the recursion converges to it — value iteration, on a matrix. This discrete form is the one a learned critic implicitly targets in Lecture 10.

Backup 4 — Euler–Lagrange to the minimum principle

Adjoin the dynamics with a costate $\lambda(t)$ and form $H = g + \lambda^\top f$:

$$\bar{J} = \int_0^T [H(t, x, u, \lambda) - \lambda^\top \dot{x}] dt + q(x(T))$$

Perturb the optimal control, $u^* \rightarrow u^* + \epsilon p(t)$, and require $\frac{d\bar{J}}{d\epsilon} \big|_{\epsilon=0} = 0$ for every admissible p . Integrating by parts and collecting terms gives

$$\dot{x}^* = \frac{\partial H}{\partial \lambda} = f, \quad \dot{\lambda}^* = -\frac{\partial H}{\partial x}, \quad \lambda(T) = \frac{\partial q}{\partial x}, \quad \frac{\partial H}{\partial u} = 0$$

From Euler–Lagrange to Pontryagin. Stationarity, $\partial_u H = 0$, presumes u is interior to U . Pontryagin replaces it with the global condition $u^* = \arg \min_{u \in U} H(t, x^*, u, \lambda^*)$, which remains valid on the *boundary* of U — so it covers saturated actuators and bang-bang controls, where the derivative never vanishes.

A conserved quantity. Along a trajectory $\frac{dH}{dt} = \frac{\partial H}{\partial u} \dot{u} + \frac{\partial H}{\partial t}$; for a time-invariant problem the second term vanishes and along a smooth interior extremal the first vanishes because $H_u = 0$. Thus H is constant there; constancy alone does not prove optimality or that the minimum principle holds — the invariant the Act 4 widget plots. The minimum principle is also the standard way to *state* equilibrium conditions for dynamic games, which is how [IE579](#) uses it.

Same problem, different dialect

The fixed point of the whole lecture, in one line:

Control theory and dynamic programming are **two dialects for one dynamic optimum**.

Where Lecture 7 had a stochastic transition over a finite set of states, control theory studies a system — usually continuous, often deterministic — driven by a control:

$$\dot{x}(t) = f(t, x(t), u(t)) \quad \text{or} \quad x_{k+1} = f_k(x_k, u_k), \quad \min_u \int_0^T g \, dt + q(T, x(T))$$

The goal is identical to Lecture 7's: not a *plan*, but a **feedback law** $u = \gamma(t, x)$ — the best control at every state and every time. The differences are of tradition, not of substance:

- finite action set $\rightarrow$ **continuous** control $u \in U \subseteq \mathbb{R}^m$;
- probabilistic transition $T(s, a, s') \rightarrow$ **differential** dynamics f ;
- sum $\rightarrow$ integral; reward and max $\rightarrow$ cost and **min**; a backup $\rightarrow$ a **PDE**.

One wall Lecture 8 left, and where it actually lands

the continuous-argmax problem — a meeting point, not the premise

Lecture 8 ended at a wall: with a continuum of actions, $\max_{a'} Q(s', a')$ cannot be found by enumerating a finite action list. A generic action maximum may require a costly numerical optimization.

That wall is not this lecture's starting point — we begin from Lecture 7, not Lecture 8. But it *is* one of the seams where the two lineages meet, and it is worth naming now because it recurs three times today:

WHERE	THE SAME INNER PROBLEM	WHAT HAPPENS TO IT
Act 1 • the DP backup	$\min_{u \in U} \{g + V(f(x, u))\}$ over a continuum	it becomes a nested Lecture-1 problem
Act 2 • LQR	the identical min, with g quadratic and f linear	solved in closed form , one line of algebra
Act 4 • Pontryagin	$\min_{u \in U} H$ at each instant	reduced to a <i>static</i> minimisation

Control theory has lived with the continuous $\arg \min$ since 1956, and its three answers are today's acts. Lecture 10's answer — *learn a network that outputs the minimiser* — is the fourth, and it is a descendant of the third.

The translation table — Lecture 7 in continuous time

Every object of dynamic programming has a control-theoretic twin. Keep this in sight all lecture.

LECTURE 7 (OR / DYNAMIC PROGRAMMING)	LECTURE 9 (CONTROL)	THE SHIFT
transition $T(s, a, s')$	dynamics $\dot{x} = f(t, x, u)$	probabilistic $\rightarrow$ differential
reward R , maximise	cost g , minimise	a sign, and a culture
value $V(s), Q(s, a)$	cost-to-go $V(t, x)$	state $\rightarrow$ state <i>and</i> time
policy $\pi(s)$	feedback law $u = \gamma(t, x), u = -Kx$	a rule, now continuous-valued
Bellman optimality (sum, max)	HJB equation (PDE, min)	a backup $\rightarrow$ a PDE
value iteration	the Riccati recursion	the same sweep, on a matrix
policy iteration / LP	Riccati / Pontryagin	a solver, not a sampler

Read it as one claim: **HJB is the Bellman optimality equation made infinitesimal**. Lecture 7's intuition transfers wholesale; only the calculus changes. And keep the last two rows in view — that is the seam where, in Lecture 10, samples will replace the solver.

— EXTENSION

Derivations — HJB and Riccati in detail

Read after completing the main route.

Shrink the step — and a backup becomes a PDE

Apply the principle of optimality over a short interval δ , then let $\delta \rightarrow 0$:

$$V(t, x) = \min_{u \in U} \left\{ g(t, x, u) \delta + V(t + \delta, x + f(t, x, u) \delta) \right\} + o(\delta)$$

Taylor-expand the second term, cancel $V(t, x)$ from both sides, divide by δ and take the limit. Assuming V is continuously differentiable, what survives is the **Hamilton–Jacobi–Bellman equation**:

$$-\frac{\partial V(t, x)}{\partial t} = \min_{u \in U} \left\{ \frac{\partial V(t, x)}{\partial x} f(t, x, u) + g(t, x, u) \right\}, \quad V(T, x) = q(T, x)$$

This *is* the Bellman optimality equation. It has become **a nonlinear PDE for V** .

The full three-line derivation is Backup 1. Compare it with Lecture 7's: identical, with $\delta \rightarrow 0$ where Lecture 7 had a one-step sum.

The Riccati equation, and a gain that stops moving

Substitute u^* back and match $x^\top(\cdot)x$ for all x . What is left is an ODE in the matrix P_t alone:

$$-\dot{P}_t = A^\top P_t + P_t A - P_t B R^{-1} B^\top P_t + Q, \quad P_T = Q_f$$

the [Riccati differential equation](#), integrated backwards in time — a backward sweep, exactly like Lecture 7's.

For infinite-horizon LQR, standard sufficient assumptions are **stabilizability of** (A, B) and **detectability of** $(Q^{1/2}, A)$, with $R \succ 0$.

The stabilizing Riccati solution gives a stationary value and gain:

$$A^\top P + P A - P B R^{-1} B^\top P + Q = 0 \quad (\text{the } \text{algebraic Riccati equation})$$

And so the optimal controller is a matrix: $u^* = -Kx$, $K = R^{-1}B^\top P$.

The discrete-time problem $x_{t+1} = Ax_t + Bu_t$, $J = \sum (x^\top Q x + u^\top R u)$ runs the identical argument and lands on $P = Q + A^\top P A - A^\top P B (R + B^\top P B)^{-1} B^\top P A$ with $K = (R + B^\top P B)^{-1} B^\top P A$. Backup 3 carries both derivations in full.

The Riccati recursion **is** value iteration

The single sharpest bridge between Lecture 7 and Lecture 9, and it comes free with the discrete-time form.

$$P_{k+1} = Q + A^\top P_k A - A^\top P_k B (R + B^\top P_k B)^{-1} B^\top P_k A, \quad P_1 = Q$$

LECTURE 7	LECTURE 9
$V_{k+1} \leftarrow \max_a \mathbb{E}[R + \gamma V_k]$	$P_{k+1} \leftarrow \text{the Riccati map}$
V_k is a table , one entry per state	P_k is an $n \times n$ matrix
converges to the unique fixed point V^*	converges to the unique PSD solution of the ARE
a sweep touches every state	a matrix recursion represents values over a <i>continuum</i> of states

Value iteration, with the value function carried in closed form instead of enumerated. The uniqueness of the positive-semidefinite P plays the role of the uniqueness of the Bellman fixed point, and — the source deck's own remark — **infinite-horizon LQR is just steady-state finite-horizon LQR**. One backward sweep, run to convergence.

Derivation — LQR from the HJB equation

$$\dot{x} = Ax + Bu, \quad J = \int_0^T (x^\top Qx + u^\top Ru) dt + x(T)^\top Q_f x(T), \quad Q, Q_f \succeq 0, R \succ 0$$

Try $V(t, x) = x^\top P_t x$. Then $\partial_x V = 2P_t x$ and $\partial_t V = x^\top \dot{P}_t x$, and the HJB equation reads

$$-x^\top \dot{P}_t x = \min_u \left\{ 2x^\top P_t (Ax + Bu) + x^\top Qx + u^\top Ru \right\}$$

The inner minimisation — the continuum that Act 1 warned about — is now a **convex quadratic in u** . Set the gradient to zero:

$$2B^\top P_t x + 2Ru = 0 \quad \implies \quad u^*(t) = \underbrace{-R^{-1}B^\top P_t}_{-K_t} x(t)$$

The wall of the continuous arg min falls to **one line of linear algebra**.

Derivation — the scalar LQR Bellman equation

Let $\dot{x} = u$ and $J = \int_0^\infty (x^2 + u^2) dt$. Try $V(x) = Px^2$.

$$0 = \min_u \{x^2 + u^2 + 2Pxu\}, \quad u^* = -Px.$$

Substitution gives $0 = (1 - P^2)x^2$. The nonnegative stabilizing solution is $P = 1$, so $u^* = -x$.

Starting at $x(0) = 2$: $x(t) = 2e^{-t}$ and

$$J^* = \int_0^\infty 8e^{-2t} dt = 4 = V(2).$$

The gain is 1: a displacement of 2 calls for control -2 ; a displacement of 0.5 calls for -0.5 . Feedback automatically weakens as the error shrinks.