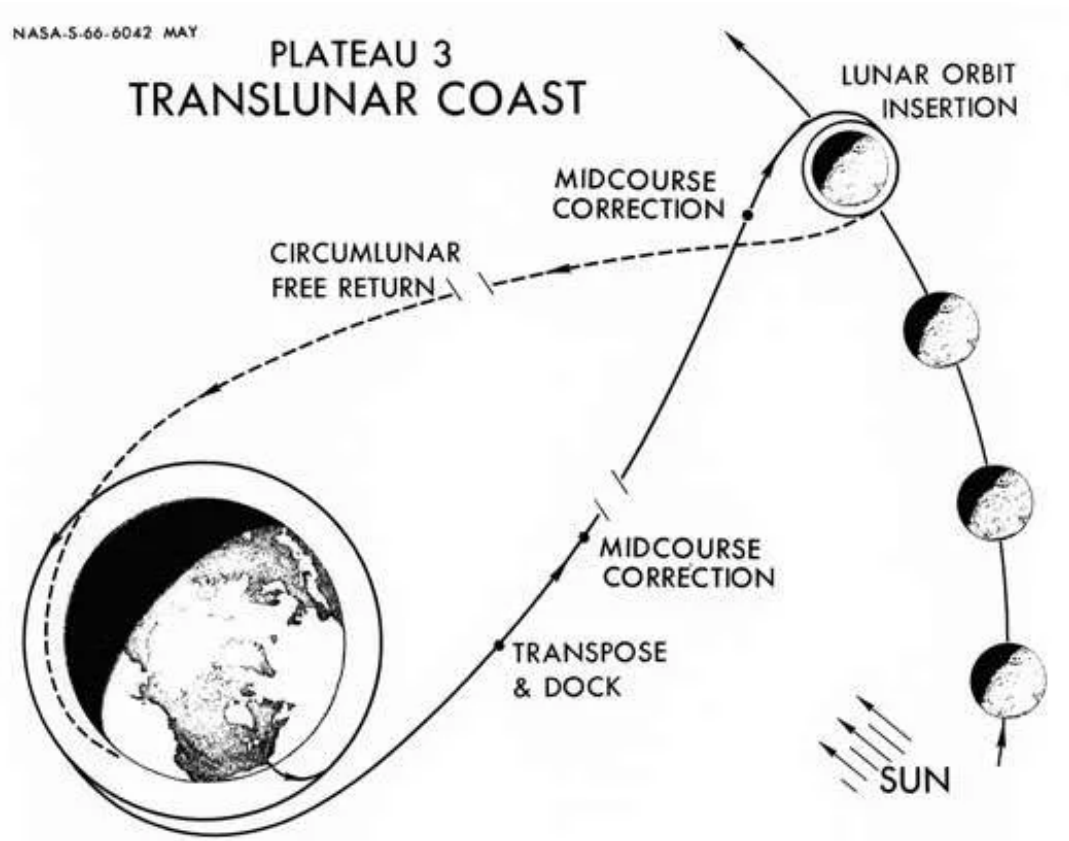
Computational Control

Dynamic Programming and LQR

Saverio Bolognani

Automatic Control Laboratory (IfA)
ETH Zurich



- **Control task:** decide sequence of control "pulses" $\{u_t\}_{t=0,1,2,\ldots,T}$
- **Performance metric:**
  - minimize fuel use
  - circumnavigate the Moon
  - get back to Earth by time T

Optimal control input

In absence of disturbance/uncertainty, the entire behavior of the system is determined by the sequence $\mathbf{u} = u_0, u_1, u_2, \ldots$

Optimization problem

$$\min_{\mathbf{u} \in \mathcal{U}} J(\mathbf{u})$$

- $\mathcal{U} = \tilde{\mathcal{U}} \times \{0, 1, \ldots, T\}$ represents the set of feasible inputs
- $J(\mathbf{u})$ is the sum of
  - a fuel cost term $\|\mathbf{u}\|_1$
  - a terminal constraint term $\delta(\mathbf{u})$

$$\delta(\mathbf{u}) = \begin{cases} 0 & \text{if the probe has reached the Moon and is back on Earth at time } T \\ +\infty & \text{otherwise} \end{cases}$$

$$\min_{\mathbf{u} \in \mathcal{U}} J(u)$$

An **intractable** optimization problem:

- the dimension of **u** make the **problem size** formidable
- the cost function requires an accurate **model of the system**
  - first-principle model
  - numerical simulator
  - black box/oracle for the terminal constraint
- the optimization problem is **non-convex**
- the **open-loop** nature of the optimal control sequence makes it useless in presence of **disturbances** and **uncertainty**

Problem decomposition

Key assumptions: Markovian representation

1. There exists a **Markovian state** that evolves according to

$$x_{t+1} = f_t(x_t, u_t) \qquad \text{(discrete-time dynamical system)}$$

2. The **initial condition** x_0 is known.
3. The **cost** is additive over time

$$J = \sum_t g_t(x_t, u_t)$$

- What does Markovian mean?
- What would be the a valid state for the lunar trajectory planning?
- Would the proposed cost satisfy this assumption?

Bellman's principle

$$\begin{aligned} \min_{\mathbf{u},\mathbf{x}} \quad & \sum_{t=0}^{T} g_t(x_t, u_t) \\ \text{subject to} \quad & x_{t+1} = f_t(x_t, u_t), \quad x_0 = X_0 \\ & x_t \in \mathcal{X}_t \quad \forall t \\ & u_t \in \mathcal{U}_t \quad \forall t. \end{aligned}$$

Bellman's principle of optimality

"Any tail of an optimal trajectory is optimal too."

In particular:

$$\min_{u_0} \left\{ g_0(X_0, u_0) \; + \; \begin{aligned} \min_{\substack{u_1,\dots u_T \\ x_1,\dots,x_T}} \quad & \sum_{t=1}^{T} g_t(x_t, u_t) \\ \text{subject to} \quad & x_{t+1} = f_t(x_t, u_t) \\ & x_1 = f_0(x_0, u_0) \end{aligned} \right\}$$

$$V_0(X_0) = \min_{u_0} \left\{ g_0(X_0, u_0) + \underbrace{\begin{array}{ll} \min\limits_{\substack{u_1,\dots u_T \\ x_1,\dots,x_T}} & \sum_{t=1}^{T} g_t(x_t, u_t) \\ \text{subject to} & x_{t+1} = f_t(x_t, u_t) \\ & x_1 = f_0(x_0, u_0) \end{array}}_{= V_1(x_1) = V_1(f_0(x_0, u_0))} \right\}$$

- The problems are nested, therefore we need to
  - solve the subproblem **parametrized in** x_1
  - compute the **value function** $V_1(x_1)$
  - solve the "outer" problem

$$\min_{u_0} \left\{ g(X_0, u_0) + V_1(f_0(X_0, u_0)) \right\}$$

- The "inner" problem is a smaller optimal control problem (shorter horizon).

Dynamic programming

At the **last stage** (T), the subproblem is trivial.

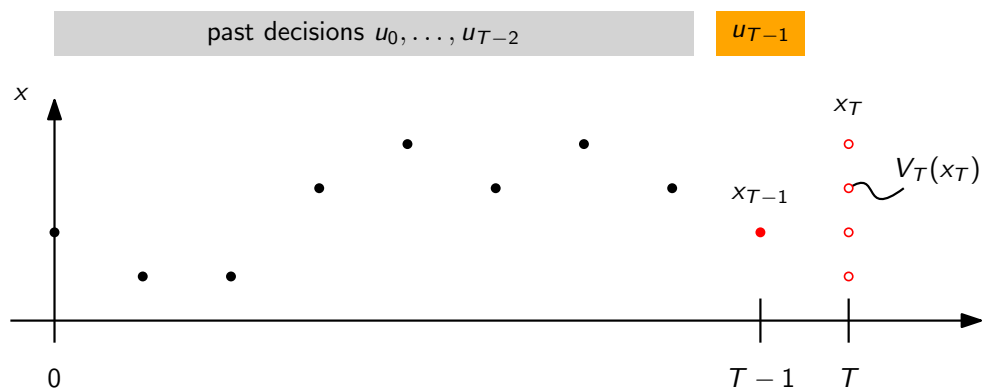
$$V_T(x_T) = g_T(x_T)$$

At **time** $T-1$, compute $V_{T-1}(x_{T-1})$ as

$$\begin{aligned} \min_{u_{T-1}} \quad & g_{T-1}(x_{T-1}, u_{T-1}) + V_T(x_T) \\ \text{subject to} \quad & x_T = f_{T-1}(x_{T-1}, u_{T-1}) \end{aligned}$$

that is

$$\min_{u_{T-1}} \quad g_{T-1}(x_{T-1}, u_{T-1}) + V_T(f_{T-1}(x_{T-1}, u_{T-1}))$$



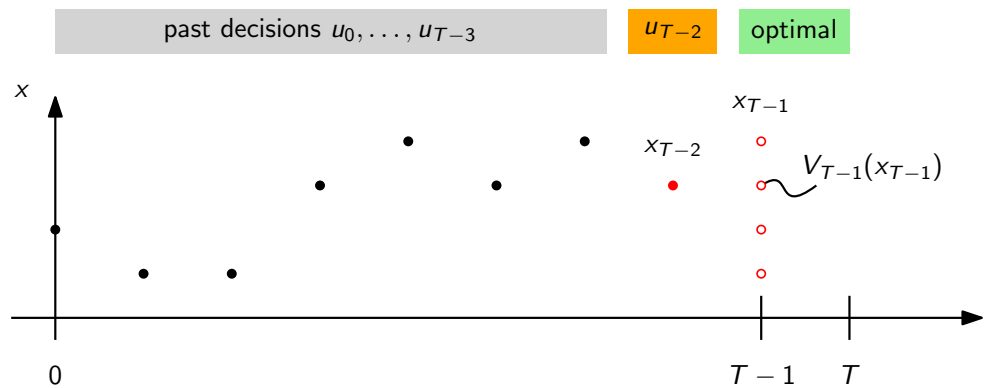
Dynamic programming

At **time** $T-2$, compute $V_{T-2}(x_{T-2})$ as

$$\begin{aligned} \min_{u_{T-2}} \quad & g_{T-2}(x_{T-2}, u_{T-2}) + V_{T-1}(x_{T-1}) \\ \text{subject to} \quad & x_{T-1} = f_{T-2}(x_{T-2}, u_{T-2}) \end{aligned}$$

that is

$$\min_{u_{T-2}} \quad g_{T-2}(x_{T-2}, u_{T-2}) + V_{T-1}(f_{T-2}(x_{T-2}, u_{T-2}))$$



Problem decomposition

The optimal control problem is decomposed into **stage problems** that we can solve via **backward induction**.

$$V_t(x_t) = \min_{u_t} \left\{ g_t(x_t, u_t) + V_{t+1}(f_t(x_t, u_t)) \right\}, \qquad V_T(x_T) = g_T(x_T).$$

When is this stage problem simple to solve?

- Decision space $\tilde{\mathcal{U}}$ convex.
- g_t convex in u_t.
- $V_{t+1} \circ f_t$ convex in u_t

In practice: **Quadratic cost** g and **linear dynamics** f

Unless we use approximations of the value function

Optimal Linear-Quadratic Regulation (LQR)

Markovian update

$$x_{t+1} = Ax_t + Bu_t$$

Cost function

$$\sum_{t=0}^{T-1} \left(x_t^\top Q x_t + u_t^\top R u_t \right) + x_T^\top S x_T, \qquad Q, S \succeq 0,\ R \succ 0$$

Value function (minimum cost to go, starting from x at time t)

$$\begin{aligned} V_t(x) \quad = \quad & \min_{u_t, \dots u_{T-1}} \quad \sum_{s=t}^{T-1} \left(x_s^\top Q x_s + u_s^\top R u_s \right) + x_T^\top S x_T \\ & \text{subject to} \quad x_{s+1} = Ax_s + Bu_s \quad s = t, \dots, T-1 \\ & \qquad\qquad\ \ x_t = x \end{aligned}$$

$V_0(x_0)$ is the optimal cost.

LQR

Key ideas

1. $V_T(x) = x^\top S x$ (convex quadratic function)
2. Show that $V_t(x)$ is also quadratic: $V_t(x) = x^\top P_t x$
3. Compute P_t recursively, working backward from T
4. The optimal u_t can be computed as a solution of a convex optimization problem.

We proceed by induction at stage t, where the value function is

$$V_t(x) \quad = \quad \min_u \quad \left(\underbrace{x^\top Q x + u^\top R u}_{\text{stage cost}} + V_{t+1}(Ax + Bu) \right)$$

Induction step: optimal control at stage t

We proceed by induction, **assuming** that $V_{t+1}(x) = x^\top P_{t+1} x$.
Then $V_t(x)$ becomes

$$V_t(x) = x^\top Q x + \min_u \left(u^\top R u + (Ax + Bu)^\top P_{t+1} (Ax + Bu) \right) =$$

$$x^\top Q x + \min_u \left(u^\top R u + x^\top A^\top P_{t+1} A x + 2u^\top B^\top P_{t+1} A x + u^\top B^\top P_{t+1} B u \right)$$

$$= x^\top Q x + \min_u \left(u^\top (R + B^\top P_{t+1} B) u + x^\top A^\top P_{t+1} A x + 2u^\top B^\top P_{t+1} A x \right)$$

Unconstrained optimization problem:
$\rightarrow$ to find the minimizer, we set the gradient with respect to u to zero, i.e.

$$(R + B^\top P_{t+1} B) u^* + B^\top P_{t+1} A x = 0$$

that is

$$u^* = -(R + B^\top P_{t+1} B)^{-1} B^\top P_{t+1} A x$$

Induction step: quadratic value function

Now we need to check that $V_t(x)$ is also quadratic in x, i.e. $V_t(x) = x^\top P_t x$, and to find how to compute P_t.
We plug the minimizer u^* into $V_t(x)$, and obtain the ugly expression

$$\begin{aligned} V_t(x) = x^\top Q x + x^\top A^\top P_{t+1} B (R + B^\top P_{t+1} B)^{-1} (R + B^\top P_{t+1} B) \cdot \\ \cdot (R + B^\top P_{t+1} B)^{-1} B^\top P_{t+1} A x + x^\top A^\top P_{t+1} A x \\ - 2 x^\top A^\top P_{t+1} B (R + B^\top P_{t+1} B)^{-1} B^\top P_{t+1} A x. \end{aligned}$$

Just collecting the different terms we get $V_t(x) = x^\top P_t x$ where P_t is defined as

$$P_t = Q + A^\top P_{t+1} A - A^\top P_{t+1} B (R + B^\top P_{t+1} B)^{-1} B^\top P_{t+1} A$$

which completes the proof.

Anything missing?

Optimal control sequence (open loop)

Backward induction:

$$u_t = -(R + B^\top P_{t+1} B)^{-1} B^\top P_{t+1} A x_t$$

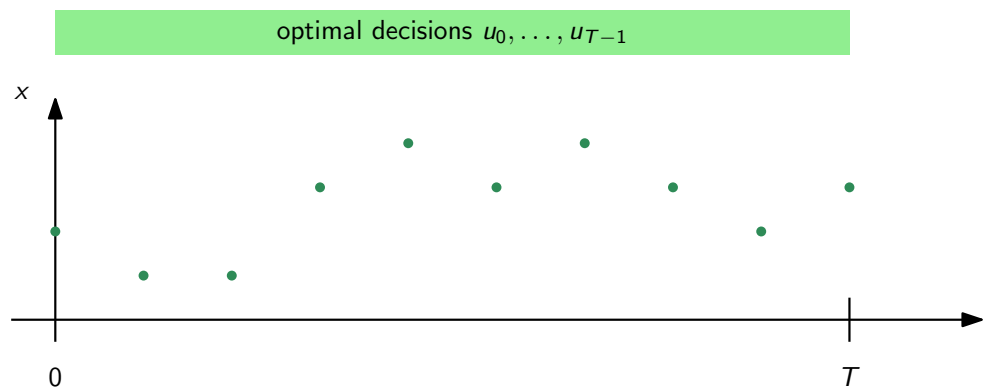
where

$$P_{t-1} = Q + A^\top P_t A - A^\top P_t B (R + B^\top P_t B)^{-1} B^\top P_t A, \qquad P_T = S.$$

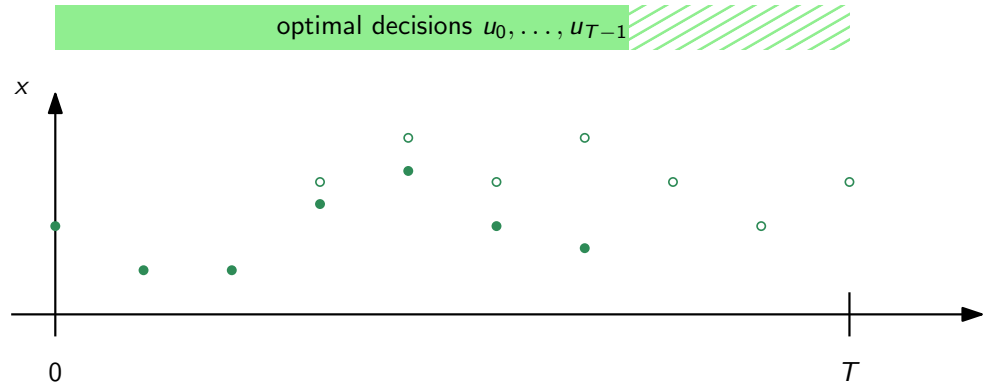
Forward integration from x_0:

$$x_{t+1} = A x_t + B u_t.$$

Result from **offline computation**: open-loop sequence $u_0, u_1, \ldots, u_{T-1}$



What if the optimal control input **u** takes us to
a trajectory different from the one we have computed?



Optimal feedback control

$$u_t = \underbrace{-(R + B^\top P_{t+1} B)^{-1} B^\top P_{t+1} A}_{:= \Gamma_t} x_t$$

where

$$P_{t-1} = Q + A^\top P_t A - A^\top P_t B (R + B^\top P_t B)^{-1} B^\top P_t A, \qquad P_T = S.$$

Computational complexity of optimal feedback control

Part of the computation can still be performed offline, and only a small part needs to happen online.

Optimal feedback control

$$u_t = \underbrace{-(R + B^\top P_{t+1} B)^{-1} B^\top P_{t+1} A}_{:= \Gamma_t} x_t$$

where

$$P_{t-1} = Q + A^\top P_t A - A^\top P_t B (R + B^\top P_t B)^{-1} B^\top P_t A, \qquad P_T = S.$$

Offline computation

- $n \times n$ matrix multiplications
- $m \times m$ matrix inversion
- T iterations

Online computation

- storage of T $m \times n$ matrices
- $m \times n$ matrix multiplications

Infinite horizon LQR



Extension of the LQR optimal control to $T \to \infty$.

- "persistent" tracking and regulation problems
- time-invariant feedback control law
  - requires less memory in online implementation
  - allows analysis tools (stability, etc.) from LTI system theory

Infinite-horizon cost function

$$\begin{aligned} \min \quad & \sum_{t=0}^{\infty} \left(x_t^\top Q x_t + u_t^\top R u_t \right), \qquad Q \succeq 0,\ R \succ 0 \\ \text{subject to} \quad & x_{t+1} = A x_t + B u_t, \quad x_0 = X_0 \end{aligned}$$

Feasibility

If the system is stabilizable, then there is an input sequence that yields a finite cost.

Proof:
Let K be a linear feedback such that $A + BK$ has eigenvalues inside the unit circle. Consider the input $u(t) = Kx(t)$, which yields

$$V(X_0) = \sum_{t=0}^{\infty} x_t^\top Q x_t + x_t^\top K^\top R K x_t$$

As $x_t = (A + BK)^t X_0$, we have

$$V(X_0) = X_0^\top \sum_{t=0}^{\infty} ((A + BK)^\top)^t (Q + K^\top R K)(A + BK)^t X_0$$

which is finite ($\approx$ geometric series).

How would you compute the optimal control for the infinite-horizon LQR?

Algebraic Riccati Equation

The iteration

$$P_{t-1} = Q + A^\top P_t A - A^\top P_t B(R + B^\top P_t B)^{-1} B^\top P_t A, \qquad P_0 = 0$$

converges, for $t \to -\infty$, to a solution $P_\infty \succ 0$ of the **algebraic Riccati equation**

$$P_\infty = Q + A^\top P_\infty A - A^\top P_\infty B(R + B^\top P_\infty B)^{-1} B^\top P_\infty A.$$

Sketch of the proof:

- The sequence $P_0, P_{-1}, P_{-2}, \ldots$ is non decreasing ($P_t \succeq P_{t+1}$). Why?
- The sequence $P_0, P_{-1}, P_{-2}, \ldots$ is upper-bounded. Why?

Optimal feedback control (infinite horizon)

The input

$$u_t = \underbrace{-(R + B^\top P_\infty B)^{-1} B^\top P_\infty A}_{:= \Gamma_\infty} x_t$$

minimizes the infinite-horizon control cost and yields $V(X_0) = X_0^\top P_\infty X_0$.

Offline computation

- $n \times n$ matrix multiplications
- $m \times m$ matrix inversion
- "Infinite" iterations*

* P_∞ can be computed by solving the Algebraic Riccati Equation.

Online computation

- storage of **one** $m \times n$ matrix
- $m \times n$ matrix multiplication

Infinite horizon LQR



Extension of the LQR optimal control to $T \to \infty$.

- "persistent" tracking and regulation problems
- time-invariant feedback control law
  - requires less memory in online implementation
  - allows analysis tools (stability, etc.) from LTI system theory

Stability of the optimal controller

Is it a sensible question to ask?

Example

$$x_{t+1} = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} x_t + u_t, \qquad Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad u_t = \underbrace{\begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix}}_{\Gamma_\infty} x_t$$

Theorem

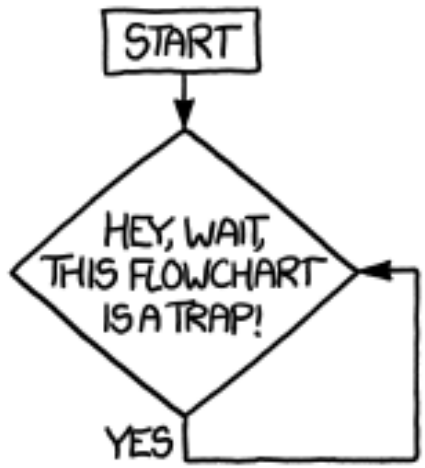
Let $Q = C^\top C$. The optimal infinite-horizon feedback Γ_∞ stabilizes the system **if and only if** it is stabilizable and

$$x_{t+1} = Ax_t, \quad y_t = Cx_t$$

does **not** have **unobservable** unstable modes.

In practice: if the system has unstable dynamics, those states need to be weighted in the cost function.

The control engineer flowchart





This work is licensed under a
Creative Commons Attribution-NonCommercial 4.0 International License

https://bsaver.io/COCO