

DYNAMIC PROGRAMMING IN DISCRETE-TIME

5c

- Dynamic programming in discrete state space
- Linear quadratic problems
- Infinite horizon problems
- The linear quadratic regulator
- (Robust and stochastic Dynamic programming)
- (Properties of the DP operator)
- Gradient of the value function
- Discrete time minimum principle
- Iterative dynamic programming
- Differential dynamic programming.

GENERALITIES

DYNAMIC PROGRAMMING REFERS TO A SET OF TECHNIQUES USED TO SOLVE OPTIMAL CONTROL PROBLEMS

→ IT IS EASIER TO IMPLEMENT AND APPLY TO SYSTEMS WITH DISCRETE STATE AND CONTROL SPACES, IN DISCRETE TIME

→ WHEN APPLIED TO DISCRETE-TIME SYSTEMS WITH CONTINUOUS STATE SPACES, IT IS OFTEN NEEDED THE USE OF SOME APPROXIMATION (usually discretization)

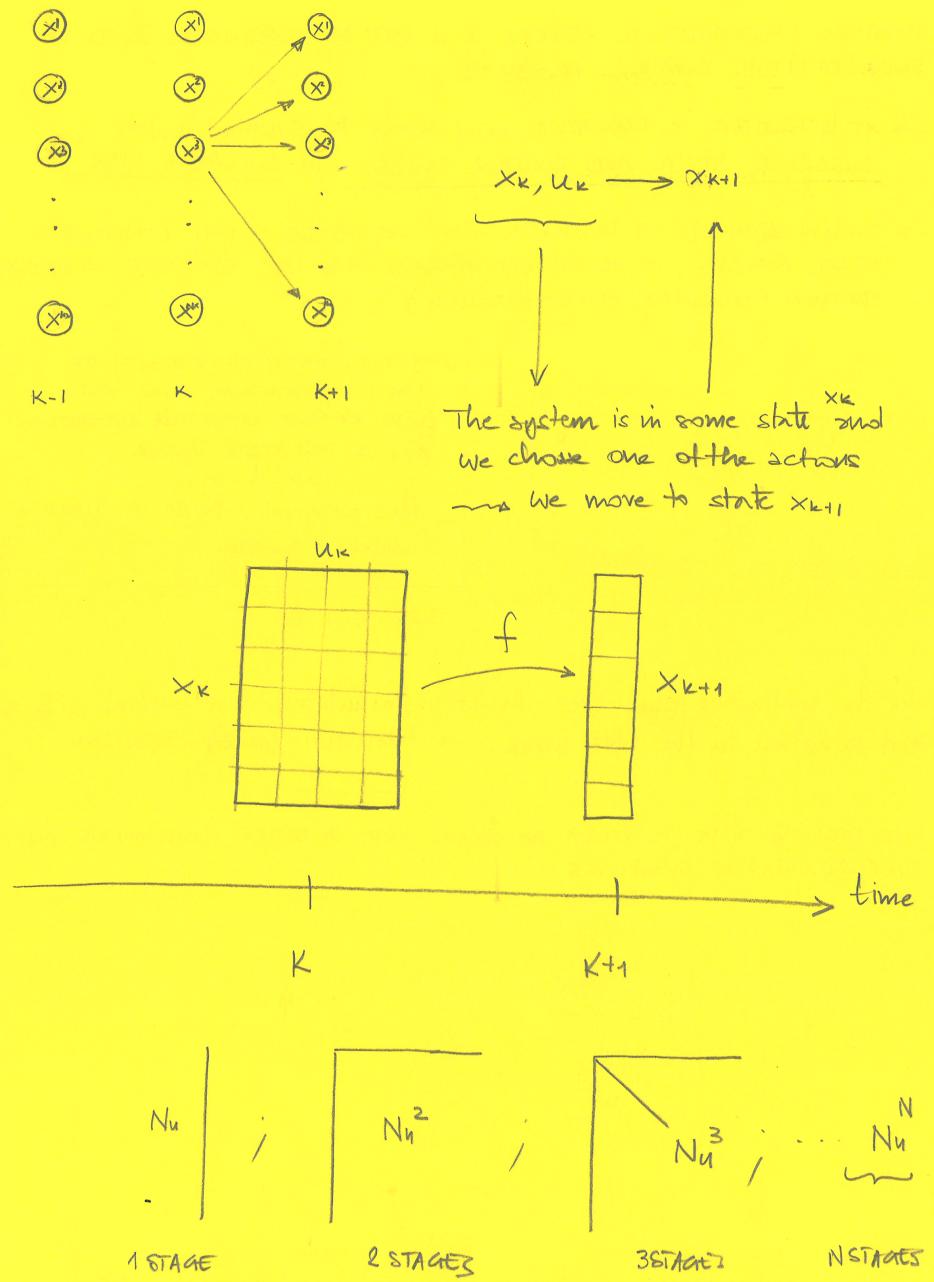
With exponential growth of the computational time / cost with respect with the dimension N_x of the state vector
 " " "
 The original " curse of dimensionality" problem

$$\text{MAX } N_x = 6$$

In the continuous time case, the DP is formulated as a partial differential equation in the state space → HAMILTON-JACOBI-BELTRAMI

THE POSITIVE SIDE IS THAT DP DOES NOT REQUIRE CONTINUOUS AND DIFFERENTIABLE DYNAMICS

DP1



DYNAMIC PROGRAMMING w/ DISCRETE STATE SPACE

Consider the dynamic system $x_{k+1} = f(x_k, u_k)$ with
 $f: \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{X}$

$$\rightarrow x_k \in \mathbb{X}$$

$$\rightarrow u_k \in \mathbb{U}$$

THE SETS ARE ASSUMED TO BE FINITE AND THUS COUNTABLE, WITH

$$|\mathbb{X}| = N_x$$

$$|\mathbb{U}| = N_u$$

We now define the STAGE COST $L(x, u)$ and the TERMINAL COST $\bar{L}(x)$
AND THEY ARE BOTH ASSUMED TO TAKE VALUES IN $\mathbb{R}_{\geq 0} = \{0, 1, 2, \dots\}$
AND ∞ , INFINITY, DENOTES THE INFEASIBLE PAIR (x, u) OR $x = \infty$

THE OPTIMAL CONTROL PROBLEM (SIMPLIFIED)

$$\begin{array}{ll} \text{minimize} & \sum_{k=0}^{N-1} L(x_k, u_k) + \bar{L}(x_N) \\ x_0, x_1, \dots, x_{N-1}, x_N & \text{STAGE COST} \\ u_0, u_1, \dots, u_{N-1} & \text{TERMINAL COST} \end{array}$$

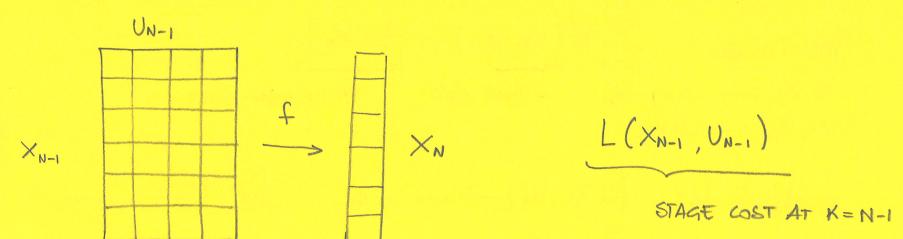
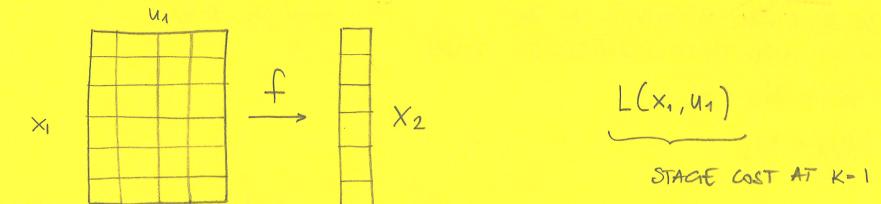
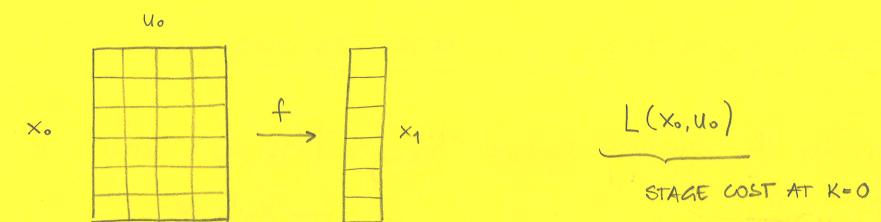
$$\begin{array}{ll} \text{subject to} & f(x_k, u_k) - x_{k+1} = 0, \quad \text{for } k=0, 1, \dots, N-1 \\ & x_0 = x_0 \end{array}$$

\uparrow INITIAL STATE IS FIXED

* The initial state is fixed

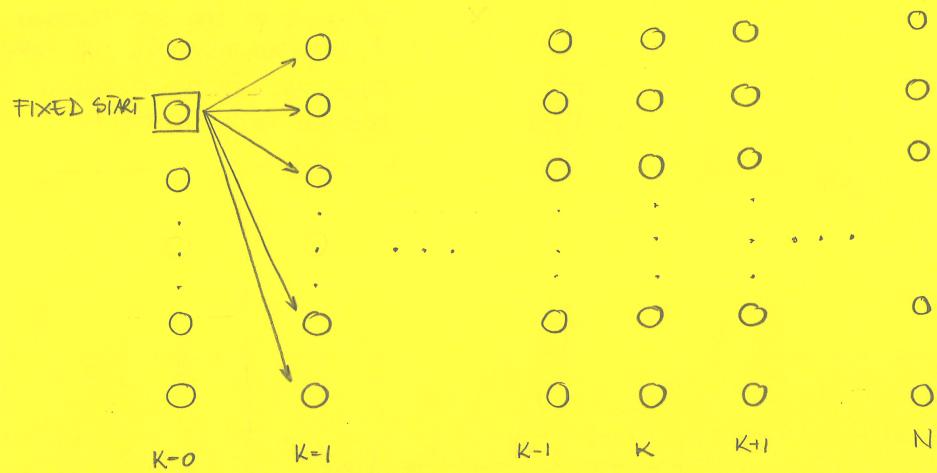
* The control are the only true degrees of freedom
 $\{u_k\}_{k=0}^{N-1}$ with $u_k \in \mathbb{U}$

* There exist exactly N^{N-1} different trajectories
 \rightarrow EACH ASSOCIATES TO A SPECIFIC VALUE OF THE OBJECTIVE
 \rightarrow INFINITY INDICATES AN UNPLASIBUE TRAJECTORY DPL



$L(x_N)$
TERMINAL COST

Graphically,



ASSUMING THAT THE COMPUTATION OF f AND L TAKES ONE COMPUTATIONAL UNIT AND NOTING THAT EACH TRAJECTORY REQUIRES N SUCH EVALUATIONS

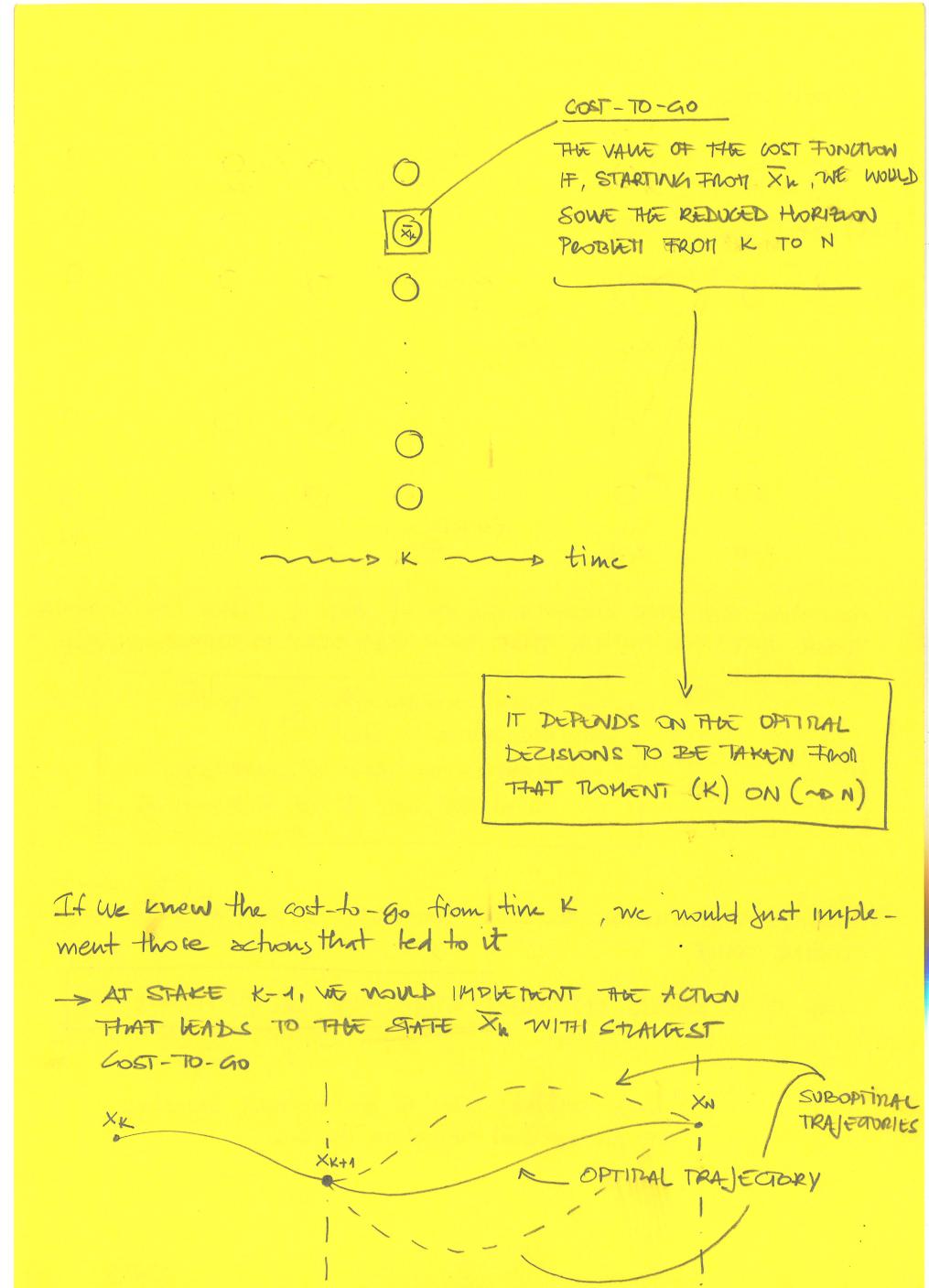
The overall complexity of simple enumeration is $O(N^N)$
→ Complexity grows exponentially with the size of the horizon N

DYNAMIC PROGRAMMING IS ABOUT ENUMERATING ALL VALUABLE TRAJECTORIES ONLY

→ IT IS BASED ON THE **PRINCIPLE OF OPTIMALITY**

"Each subtrajectory of an optimal trajectory is an optimal trajectory itself."

DP3



WE DEFINE THE VALUE FUNCTION or COST-TO-GO FUNCTION

\Rightarrow It is the optimal cost that would be achieved if at time $k \in \{0, 1, \dots, N\}$ and at state \bar{x}_k we would solve the optimal control problem on the shortened horizon

$$J_k(\bar{x}_k) = \underset{\substack{x_k, x_{k+1}, \dots, x_N \\ u_k, u_{k+1}, \dots, u_{N-1}}}{\text{minimize}} \sum_{i=k}^{N-1} \mathcal{L}(x_i, u_i) + E(x_N)$$

subject to

$$\begin{cases} f(x_i, u_i) - x_{i+1} = 0 & i = k, \dots, N-1 \\ \bar{x}_k - x_k = 0 \end{cases}$$

↓

**SHRINKED HORIZON
OPTIMAL CONTROL PROBLEM**

RETAINING DECISION VARIABLES

RETAINING CONSTRAINTS

ALL \bar{x}_k AT K

Each function $J_k : \mathbb{X} \rightarrow \mathbb{R}_{\geq 0}$ summarizes the cost-to-go to the end, when we start from some given state

$\forall k \neq N$, we trivially have $J_N(\bar{x}_N) = E(\bar{x}_N)$

Based on the principle of optimality, we can state that, for any k with $k \in \{0, 1, \dots, N-1\}$, we have

$$J_k(\bar{x}_k) = \underset{u}{\text{minimize}} \mathcal{L}(\bar{x}_k, u) + J_{k+1}(f(\bar{x}_k, u))$$

Thus the optimal trajectory can be reconstructed by forward simulation at $x_0 = \bar{x}_0$ and then proceeds as

$$x_{k+1} = f(x_k, u^*_k(x_k)), \quad \text{for } k = 0, 1, \dots, N-1$$

COST-TO-GO AT THE NEXT STEP

$$\text{GIVEN } \bar{J}_k(\bar{x}_k) = \underset{u}{\text{minimize}} \bar{\mathcal{L}}(\bar{x}_k, u) + \bar{J}_{k+1}(f(\bar{x}_k, u))$$

WE CAN DEFINE A RECURSION

- COMPUTE ALL FUNCTIONS \bar{J}_k SEQUENTIALLY FOR ALL $x_k \in \mathbb{X}$
- BACKWARDS, FOR $k = N-1, N-2, \dots, 0$
(TERMINAL COST-TO-GO IS THE TERMINAL COST \bar{e}_N)

WITH ALL THE \bar{J}_k COMPUTED, THE OPTIMAL FEEDBACK CONTROL
FOR A GIVEN STATE x_k AT TIME k IS THEN

$$u_k^*(x_k) \in \underbrace{\arg \min_u}_{\mathcal{U}} \bar{\mathcal{L}}(x_k, u_k) + \bar{J}_{k+1}(f(x_k, u)) \}$$

→ THIS COMPUTES THE OPTIMAL TRAJECTORY BY A FORWARD
SIMULATION THAT STARTS AT $x_0 = \bar{x}_0$ AND THEN PROCEEDS

$$x_{k+1} = f(x_k, u_k^*(x_k)), \quad k = 0, 1, \dots, N-1$$

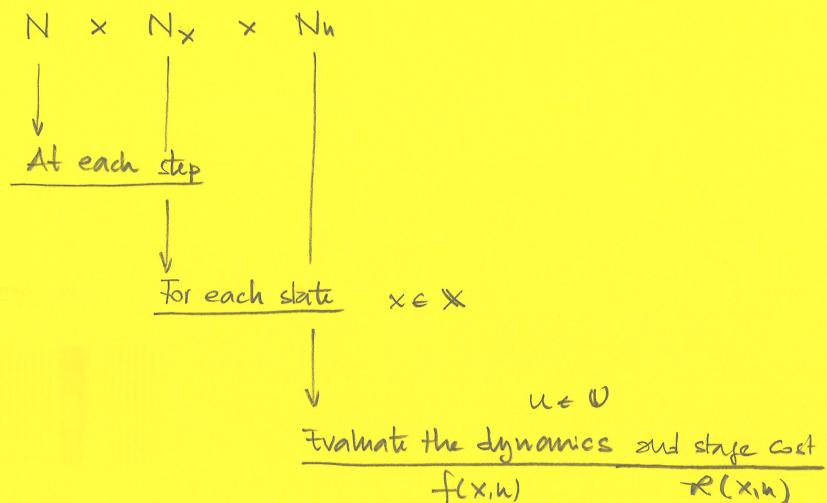
THIS SOLVES THE OPTIMAL CONTROL PROBLEM GLOBALLY

↓
THIS IS THE OPTIMAL ACTION
(A CONTROL LAW $u = \pi(x)$)

THE COMPLEXITY OF THE DP SOLUTION

- THE COST OF THE BACKWARD RECURSION

~ N STEPS, AT EACH STEP WE NEED TO EVALUATE THE SYSTEM DYNAMICS FOR EACH CONTROLS, FOR EACH STATE VALUE



- THE COST OF THE FORWARD RECURSION

~ N STEPS, AT EACH STATE EVALUATE THE SYSTEM DYNAMICS WITH OPTIMAL CONTROLS

~NEGLIGIBLE, IF COMPARED WITH THE BACKWARD RECURSION

Say $N=100$, $N_x=1K$, $N_u=10$ ~ 10^6 COMPUTATIONAL UNITS (WHICH IS SMALLER THAN 10^{100} FROM THE NATIVE ENVIRONMENT)

The solution of an optimal control using DP is that there is no need to assume the differentiability or the convexity of function f , L and E .

→ THIS IS ALSO TRUE FOR CONTINUOUS STATE SPACES

In the case of a continuous state space, we still need to represent functions J_K on the computer

via TABULATION OVER A GRID IN THE STATE SPACE

IF THE CONTINUOUS STATE SPACE IS N_x -DIMENSIONAL BOX AND WE USE A EQUAL GRID WITH m VALUES ALONG EACH DIMENSION
THEN THE TOTAL NUMBER OF GRID POINTS IS m^{N_x}

DP on this grid, then the complexity estimate is with $N_x = m^{N_x}$

When DP is applied to system with continuous state space, the computational complexity grows with the dimension of the state space

THERE EXIST MANY WAYS OF APPROXIMATING THE VALUE FUNCTION (NN)

EXPONENTIALLY

DPO

Proof: NOTE THAT THE OPTIMIZATION PROBLEM FOR A SPECIFIC x IS

$$J_{\text{NEW}}(x) = \min_u \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} Q + A^T P A & S^T A^T P A \\ S + B^T P A & R + B^T P A \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}$$

THEN, THIS PROBLEM CAN BE SOLVED EXPLICITLY FOR $\bar{R} = R + B^T P A$ NON SINGULAR, GIVING THE P_{NEW}

(APPLY SCHUR'S COMPLEMENT METHOD)

SCHUR'S COMPLEMENT METHOD

IF \bar{R} IS POSITIVE DEFINITE, THEN

$$\min_u \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} \bar{Q} & \bar{S}^T \\ \bar{S} & \bar{R} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} = x^T (\bar{Q} - \bar{S}^T \bar{R}^{-1} \bar{S}) x$$

AND THE MINIMIZER $u^*(x)$ IS GIVEN BY $u^*(x) = -\bar{R}^{-1} \bar{S} x$

Th (QUADRATIC VALUE FUNCTION). let $R + B^T P B$ be positive definite, then the minimum $J_{\text{NEW}}(x)$ in one step of the dynamic programming recursion

$$J_{\text{NEW}}(x) = \min_u \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + \begin{bmatrix} x \\ u \end{bmatrix}^T [A|B]^T P [A|B] \begin{bmatrix} x \\ u \end{bmatrix}$$

(NEW INDEX)

IT IS A QUADRATIC FUNCTION

IT IS GIVEN BY $J_{\text{NEW}}(x) = x^T P_{\text{NEW}} x$ (QUADRATIC)
WITH $P_{\text{NEW}} = Q + A^T P A - (S^T A^T P A)(R + B^T P B)^{-1}(S + B^T P A)$

ONE STEP FORWARD

(QUADRATIC)

LINEAR QUADRATIC CASE

PSD matrices

Consider the linear quadratic optimal control problem

$$\underset{x,u}{\text{minimize}} \sum_{k=0}^{N-1} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \underbrace{\begin{bmatrix} Q_k & S_k^T \\ S_k & R_k \end{bmatrix}}_{\text{PSD matrices}} \begin{bmatrix} x_k \\ u_k \end{bmatrix} + x_N^T P_N x_N$$

'all states' 'all controls'

$$\text{subject to } \begin{cases} x_{k+1} - A_k x_k - B_k u_k = 0, & k = 0, 1, \dots, N-1 \\ x_0 = \bar{x}_0 \end{cases}$$

HOW DO WE SOLVE THIS PROBLEM USING DYNAMIC PROGRAMMING?

* AT EACH STAGE, WE MUST COMPUTE THE COST

$$l_k(x, u) = \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} Q_k & S_k^T \\ S_k & R_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix}$$

↑
it is, in general, time-varying.

* AT EACH STAGE, WE MUST COMPUTE THE DYNAMICAL SYSTEM

$$f_k(x, u) = A_k x + B_k u$$

* THE RECURSION STEP

$$J_k(x) = \min_u l_k(x, u) + J_{k+1}(f_k(x, u))$$

↓
QUADRATIC
THESE COST-TO-GO FUNCTIONS
NEED TO BE REPRESENTED

→ We can start with $J_N(x) = x^T P_N x$

→ UNDER THESE CONDITIONS, WE HAVE THAT EACH J_k IS ITSELF QUADRATIC, THAT IS, IT HAS THE FORM

$$J_k(x) = x^T P_k x \quad \forall k$$

↙

• THIS IS ALSO QUADRATIC
IN X

IT IS QUADRATIC IN X AND U

IT IS ALSO QUADRATIC
IN X AND U

DP 7

THANK TO THE QUADRATIC FORMULATION OF THE VALUE FUNCTION, WE CAN SOLVE THE OPTIMAL CONTROL PROBLEM EXPLICITLY

1 FIRST COMPUTER EXPLICITLY ALL MATRICES P_k

2 THEN, PERFORM A FORWARD CLOSED LOOP SIMULATION

Starting with P_N , we iterate for $k = N-1, N-2, \dots, 0$
(backward)

$$P_k = Q_k + A_k^T P_{k+1} A_k - \left(S_k^T + A_k^T P_{k+1} B_k \right) \left(R_k + B_k^T P_{k+1} B_k \right)^{-1} \left(S_k + B_k^T P_{k+1} A_k \right)$$

DIFFERENCE OR DISCRETE RICCATI EQUATION

THE BACKWARD RECURSION

The optimal feedback $u_k^*(x_k) = -\underbrace{\left(R_k + B_k^T P_{k+1} B_k \right)^{-1} \left(S_k + B_k^T P_{k+1} A_k \right)}_{K_k \text{ SOME MATRIX}} x_k$

Starting with $x_0 = \bar{x}_0$, we perform the forward recursion

$$x_{k+1} = A_k x_k + B_k u_k^*(x_k) \quad \text{THE COMPLETE OPTIMAL TRAJECTORY FOR THE LQR}$$

A more general case are problems with linear quadratic costs and affine linear systems

$$\underset{x_{1:N}}{\text{minimize}} \sum_{i=0}^{N-1} \begin{bmatrix} 1 \\ x_i \\ u_i \end{bmatrix} \begin{bmatrix} * & q_i^T & s_i^T \\ Q_i & R_i & S_i^T \\ s_i & R_i & P_i \end{bmatrix} \begin{bmatrix} 1 \\ x_i \\ u_i \end{bmatrix} + \begin{bmatrix} 1 \\ x_N \\ u_N \end{bmatrix} \begin{bmatrix} * & P_N^T & \\ P_N & P_N & \end{bmatrix} \begin{bmatrix} 1 \\ x_N \\ u_N \end{bmatrix}$$

subject to $x_0 - x_0^{\text{fixed}} = 0$

$$x_{k+1} = A_i x_i + B_i u_i + c_i, \quad i=0, 1, \dots, N-1$$

Typical in linearized nonlinear systems for which an optimal control solution to the reference tracking problem is sought

$$\rightsquigarrow L_i(x; u_i) = \|x_i - x_i^{\text{ref}}\|_Q^2 + \|u_i\|_R^2$$

They must be treated using the same recursion, but

1. MUST AUGMENT THE STATE x_k

$$\rightsquigarrow \begin{bmatrix} 1 \\ x_k \end{bmatrix} = \bar{x}_k$$

2. REPLACE THE DYNAMICS

$$\rightsquigarrow \bar{x}_{k+1} = \begin{bmatrix} 1 & 0 \\ C_k & A_k \end{bmatrix} \tilde{x}_k + \begin{bmatrix} 0 \\ B_k \end{bmatrix} u_k$$

3. CORRECT THE INITIAL VALUE

$$\rightsquigarrow \bar{x}_0^{\text{fix}} = \begin{bmatrix} 1 \\ x_0^{\text{fix}} \end{bmatrix}$$

→ THE REFORMULATED PROBLEM IS THEN SOLVED BY USING THE DISCRETE RICCATI EQUATION

INFINITE HORIZON PROBLEMS ARE LOGICAL CHOICES IF THE INTEREST IS TO REGULATE A SYSTEM TOWARDS STEADY STATE

→ HOW TO APPLY DYNAMIC PROGRAMMING?

Suppose that the cost penalizes deviations from the steady state

→ OTHERWISE THE COST WILL INCREASE WITH TIME

IF WE ARE AT STEP k AND THE COST-TO-GO IS $J_k(\bar{x}_k)$, AFTER ONE STEP THE COST-TO-GO IS STILL THE SAME

$$J_{k+1}(\bar{x}_{k+1}) = J_k(\bar{x}_k), \quad \forall k$$

WE ARE STILL AS FAR AS WE WERE EVEN AFTER ONE STEP FORWARD

THE COST TO GO FROM STATE \bar{x}_{k+1} AT TIME k TO INFINITY

WE CAN REMOVE TIME INDEXES ALTOGETHER

$$J(x) = \min_u \underline{L}(x, u) + J(f(x, u))$$

↳ Q-FACTOR IN RL

↓ NO INDEX BECAUSE OF TIME INDEPENDENCE

THE COST CAN ALSO BE DISCOUNTED

$$\underset{x_{1:N}}{\text{minimize}} \sum_{k=0}^{\infty} \underline{L}(x, u) \alpha^k$$

DISCOUNT FACTOR WITH $\alpha \in (0, 1)$

INFINITE HORIZON CASES

Dynamic programming can be generalized to infinite-horizon problems

$$\begin{aligned} & \text{minimize}_{x,u} \sum_{k=0}^{\infty} \mathcal{L}(x_k, u_k) \quad \text{WE ASSUME IT TO BE TIME INDEPENDENT} \\ & \text{subject to} \quad x_0 - \bar{x}_0 = 0 \quad (\text{FIXED INITIAL VALUE}) \\ & \quad x_{k+1} - f(x_k, u_k) = 0, \quad k = 0, 1, \dots, \infty \quad (\text{SYSTEM DYNAMICS}) \end{aligned}$$

IN THIS CASE THE COST-TO-GO FUNCTION $J_k(x_k)$ BECOMES INDEPENDENT OF THE INDEX k

$$J_k(\bar{x}_k) = \begin{aligned} & \text{minimize}_{x_0, x_1, \dots, x_N, u_0, u_1, \dots, u_{N-1}} \sum_{i=k}^{N-1} \mathcal{L}(x_i, u_i) + E(x_N) \\ & \text{subject to} \quad f(x_i, u_i) - x_{i+1} = 0, \quad i = 0, 1, \dots, N-1 \\ & \quad \bar{x}_k - x_k = 0 \end{aligned}$$

Definition of Value function / Cost-to-go (GENERAL)

THE COST-TO-GO FUNCTION IS SUCH THAT $\boxed{J_k = J_{k+1}}$, FOR ALL k

THIS LEADS TO THE BELLMAN EQUATION

$$\tilde{J}(x) = \min_u \mathcal{L}(x, u) + \tilde{J}(f(x, u))$$

THE OPTIMAL CONTROLS ARE OBTAINED BY THE FUNCTION

$$\underline{u^*(x)} = \arg \min_u \tilde{J}(x, u)$$

→ MAY NOT BE UNIQUE

LQR IS THE MOST COMMON APPLICATION OF DYNAMIC PROGRAMMING IN ENGINEERING

→ LINEAR DYNAMICS, TIME INVARIANT

→ QUADRATIC COST (PENALIZES ERRORS FROM STEADY STATE, OR ZERO)

THE STAGE COST $\begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}$ LEADS TO A FINITE COST

- WE CAN THUS FIND A SOLUTION

→ WE TAKE THAT $\tilde{J}(x) = x^T P x$ WHICH IS NOT DEPENDENT ON TIME

→ WE GET A P MATRIX WHICH IS ALSO NOT DEPENDENT ON TIME

$u(x) = -K x$ WITH K WHICH IS NOT DEPENDENT ON TIME

THE LINEAR QUADRATIC REGULATOR

A special case: LINEAR SYSTEM with quadratic cost

IT IS THE SOLUTION TO THE INFINITE HORIZON PROBLEM WITH
A LINEAR TIME INVARIANT SYSTEM $\dot{x} = Ax + Bu = f(x, u)$ AND
THE QUADRATIC COST

$$J(x, u) = \begin{bmatrix} x \\ u \end{bmatrix}^\top \begin{bmatrix} Q & S^\top \\ S & R \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}$$

WE REQUIRE THE STATIONARY SOLUTION TO THE RICCATI RECURSION

AND WE SET $P_k = P_{k+1}$

AND WE OBTAIN THE ALGEBRAIC RICCATI EQUATION IN D-T

$$P = Q + A^\top PA - (S + B^\top PA)^{-1}(S + B^\top PA)$$

ALGEBRAIC
RICCATI EQUATION IN DISCRETE TIME

NONLINEAR EQUATION IN P
* P IS SYMMETRIC
* $N \times (N_x + 1)/2$ UNKNOWN'S

GIVEN THE SOLUTION P,

$$u^*(x) = -\underbrace{(R + B^\top PB)^{-1}(S + B^\top PA)}_K x$$

THE LQR
GAIN

IT CAN BE SOLVED BY ITERATIVE
APPLICATION OF THE DIFFERENCE
RICCATI EQUATION
* START WITH A ZERO MATRIX
 $P=0$
IT CAN BE SOLVED BY NEWTON-
TYPE METHODS (SOLUTION MUST
BE POSITIVE DEFINITE, THOUGH)

GRADIENT OF THE VALUE FUNCTION

The meaning of the cost-to-go, or the value function, J_k , is that it is the cost incurred on the remainder of the horizon for the best possible strategy.

→ THERE EXIST INTERESTING CONNECTIONS BETWEEN THE VALUE FUNCTION AND THE LAGRANGE MULTIPLIERS

We can consider a discrete-time optimal control problem w/o coupled constraints (which cannot be directly handled by DP)

→ WE ALSO ASSUME THAT THE INITIAL STATE IS FIXED AND THAT ALL INEQUALITY AND TERMINAL CONSTRAINTS ARE DIRECTLY IMPLEMENTED IN THE STAGE COST $L(x_{ik})$ AND TERMINAL COST $E(x_N)$ BY LETTING THEM TAKE ON INFINITE VALUES OUTSIDE FEASIBLE REGIONS

$$\text{minimise}_{x_0, u_0, x_1, u_1, \dots, x_N} \sum_{k=0}^N L(x_k, u_k) + E(x_N)$$

$$\text{subject to } f(x_k, u_k) - x_{k+1} = 0 \quad , \quad k=0, \dots, N-1 \\ x_0 - \bar{x}_0 = 0$$

The dynamic programming recursion associated to it

$$- J_N(x) = E(x) \\ - J_k(x) = \min_u \underbrace{L(x_k, u)}_{\mathcal{R}(x_k, u_k)} + J_{k+1}(f(x_k, u)) \quad , \quad k=N-1, \dots, 0$$

$$\text{And then, } x_0 = \bar{x}_0, \quad x_{k+1} = \underbrace{f(x_k, u_k)}_{\mathcal{R}(x_k, u_k)}, \quad k=0, \dots, N-1 \\ \text{with } u_k = \arg \min_u \mathcal{R}(x_k, u_k) + J_{k+1}(f(x_k, u_k)) \quad \text{DP12}$$

WE NOW CONSIDER THE SOLUTION OF

$$u_k = \underset{u}{\operatorname{argmin}} \mathcal{L}(x_k, u) + J_{k+1}(f(x_k, u))$$

→ IT MUST SATISFY THE FIRST ORDER NECESSARY CONDITIONS

$$\nabla_u \mathcal{L}(x_k, u_k) + \frac{\partial f}{\partial u} \Big|_{x_k, u_k}^+ \nabla J_{k+1}(f(x_k, u_k)) = 0$$

WHICH DEFINES u_k LOCALLY

FROM THE DYNAMIC PROGRAMMING RECURSION WE CAN FORWARDED CERTAIN CONDITIONS ON x_k AND u_k

- ON THE OPTIMAL TRAJECTORY WE HAVE $x_{k+1} = f(x_k, u_k)$

- ON THE OPTIMAL TRAJECTORY WE ALSO HAVE

$$J_N(x_N) = E(x_N)$$

$$J_k(x_k) = \mathcal{L}(x_k, u_k) + J_{k+1}(x_{k+1}), \quad k = N-1, \dots, 0$$

These imply that the value function remains constant on the entire trajectories for problems with zero stage costs

NOW CONSIDER THE GRADIENT $\nabla J_k(x_k)$ ALONG THE OPTIMAL STATE TRAJECTORY

- BY DIFFERENTIATING THE DP RECURSION AT POINT x_k WITH RESPECT TO x , WE GET

$$\nabla J_N(x_N) = \nabla E(x_N)$$

$$\nabla J_k(x_k)^T = \frac{d}{dx} \underbrace{[\mathcal{L}(x_k, u_k) + J_{k+1}(f(x_k, u_k))]}_{\text{denoted } \overline{J}_k(x_k, u_k)}$$

for $k = N-1, \dots, 0$

The evaluation of the total derivative requires to observe that the optimal u_k is obtained from

$$\nabla_{u_k} \mathcal{L}(x_k, u_k) + \frac{\partial f}{\partial u} \Big|_{x_k, u_k}^T \nabla J_{k+1}(f(x_k, u_k)) = 0$$

an implicit function of x_k

However, we have that the derivative does not depend on

$$\frac{du}{dx_k}$$
 because of $\frac{d}{dx} \bar{J}_k(x_k, u_k) = \frac{\partial \bar{J}}{\partial x} \Big|_{x_k, u_k} + \frac{\partial \bar{J}_k}{\partial u} \Big|_{x_k, u_k} \cancel{\frac{du_k}{dx_k}} = 0$
 BECAUSE $\nabla_{u_k} \mathcal{L}(x_k, u_k) + \frac{\partial f}{\partial u} \Big|_{x_k, u_k}^T \nabla J_{k+1}(f(x_k, u_k)) = 0$

thus, the gradients of the value function along the optimal trajectory have to satisfy the recursion

$$\nabla J_k(x_k) = \nabla_x \mathcal{L}(x_k, u_k) + \frac{\partial f}{\partial x} \Big|_{x_k, u_k}^T \nabla J_{k+1}(x_k, u_k), k \in \{N-1, \dots, 0\}$$

This recursion is equivalent to first order necessary conditions as we obtained for differentiable optimal control problems

WE CAN IDENTIFY $\lambda_k = \nabla J_k(x_k)$

→ The Lagrange multipliers are the gradients of the value function along the optimal solution.

DISCRETE-TIME MINIMUM PRINCIPLE

BY COLLECTING ALL THE NECESSARY CONDITIONS FOR OPTIMALITY THAT WE DERIVED, AND BY REPLACING $\nabla J_k(x_k)$ WITH λ_k , WE GET

$$\left\{ \begin{array}{l} x_0 = \bar{x}_0 \quad (\text{fixed initial state}) \\ x_{k+1} = f(x_k, u_k) \quad (\text{dynamics}) \\ \lambda_N = \nabla_{x_N} E(x_N) \\ \lambda_k = \nabla_x \mathcal{L}(x_k, u_k) + \frac{\partial f}{\partial x} \Big|_{x_k, u_k}^T \lambda_{k+1}, \quad \text{for } k=N-1, \dots, 1 \\ 0 = \nabla_u \mathcal{L}(x_k, u_k) + \frac{\partial f}{\partial u} \Big|_{x_k, u_k}^T \lambda_{k+1}, \quad \text{for } k=N-1, \dots, 1 \end{array} \right.$$

The recursion for λ becomes a differential equation that can be integrated in time