Static and Dynamic Optimization (42111)

Build. 303b, room 048
Section for Dynamical Systems
Dept. of Applied Mathematics and Computer Science
The Technical University of Denmark

Email: nkpo@dtu.dk
phone: +45 4525 3356
mobile: +45 2890 3797

2019-11-17 17:48

Lecture 11: Dynamic Programming

Outline of lecture

Dynamic Programming

Anecdote: In 1960+ buzz words like 'dynamic' and 'programming' was a **must** (like AI and 'learning' now, 'adaptive' and ANN some years ago.) to get funding.

- Recap F7-F10
- Dynamic Programming (D)
- Unconstrained
- Constrained
- Reading guidance (DO: chapter 6, page 73-83)

Dynamic Optimization (D)

Find a sequence $u_i \in \mathcal{U}_i$, $i = 0, \ldots, N-1$ which takes the system

$$x_{i+1} = f_i(x_i, u_i) \qquad x_0 = \underline{x}_0 \qquad \psi(x_N) = 0$$

from its initial state $\underline{x}_0$ along a trajectory such that the performance index

$$J = \phi_N[x_N] + \sum_{i=0}^{N-1} L_i(x_i, u_i)$$

is minimized. Define the Hamiltonian function as:

$$H_i = L_i(x_i, u_i) + \lambda_{i+1}^T f_i(x_i, u_i)$$

The necessary conditions are:

$$x_{i+1} = f_i(x_i, u_i) \qquad \lambda_i^T = \frac{\partial}{\partial x_i} H_i$$

$$u_i = arg \min_{u_i \in \mathcal{U}_i} H_i \qquad \left(0^T = \frac{\partial}{\partial u_i} H_i\right)$$

with boundary conditions:

$$x_0 = \underline{x}_0 \qquad \lambda_T^N = \nu^T \frac{\partial}{\partial x_N} \psi + \frac{\partial}{\partial x_N} \phi$$

Dynamic Optimization (C)

Find a function $u_t \in \mathcal{U}_t$ $t \in [0;\ T] \in \mathbb{R}$ which takes the system system

$$\dot{x}_t = f_t(x_t, u_t) \qquad x_0 = \underline{x}_0 \qquad \psi(x_T) = 0$$

from its initial state $\underline{x}_0$ along a trajectory such that the performance index

$$J = \phi_T[x_T] + \int_0^T L_t(x_t, u_t)\ dt$$

is minimized. Define the Hamilton function as:

$$H_t = L_t(x_t, u_t) + \lambda_t^T f_t(x_t, u_t)$$

The necessary conditions are:

$$\dot{x}_t = f_t(x_t, u_t) \qquad -\dot{\lambda}_t^T = \frac{\partial}{\partial x_t} H_t$$

$$u_t = arg \min_{u_t \in \mathcal{U}_t} H_t \qquad \left(0^T = \frac{\partial}{\partial u_t} H_t\right)$$

with boundary conditions:

$$x_0 = \underline{x}_0 \qquad \lambda_T^T = \nu^T \frac{\partial}{\partial x_T} \psi + \frac{\partial}{\partial x_T} \phi$$

Dynamic Programming - Deterministic

Found in Bertsekas (probably also an anecdote).

Kierkegaard: Life can only be understood going backwards, but it must be lived going forwards.

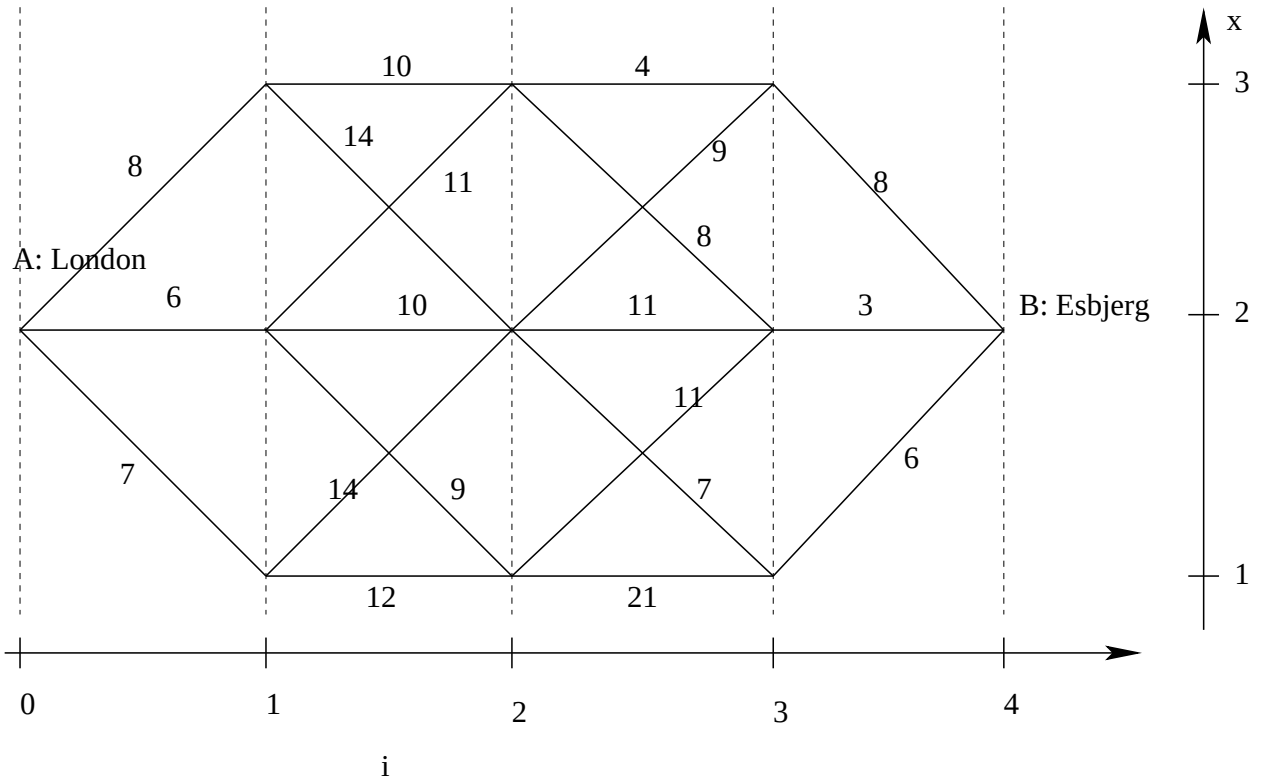
Let us (for a start) now focus on:

Discrete independent variable (normally the time, $i \in \mathbb{N}$ or $t = iT_s$).
Discrete and finite state space and decision space ($\mathcal{X}_i$ and $\mathcal{U}_i$)

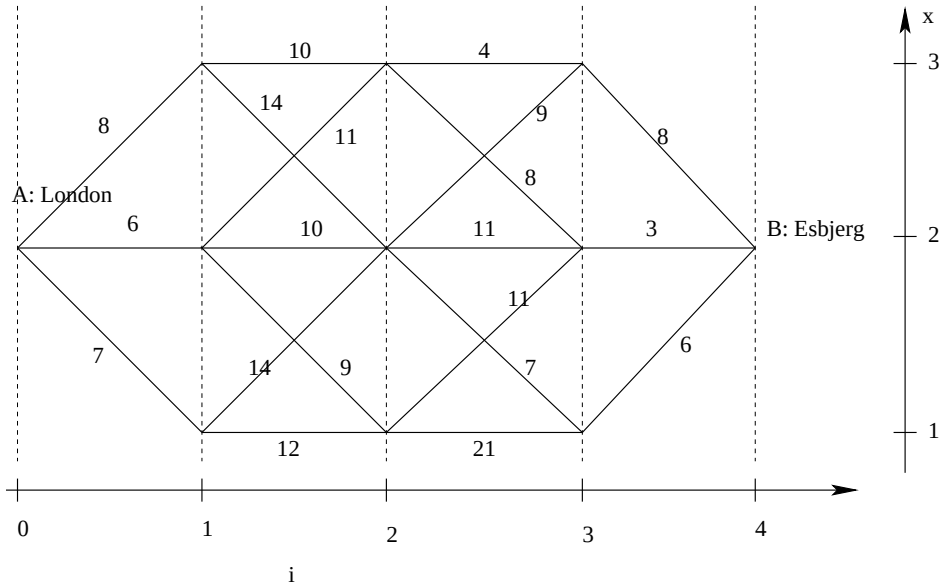
Stagecoach problem - Shortest path.

Example (Hans Ravn: Statisk og Dynamisk Optimering)

Go from A to B with minimal cost.



Dynamic Programming



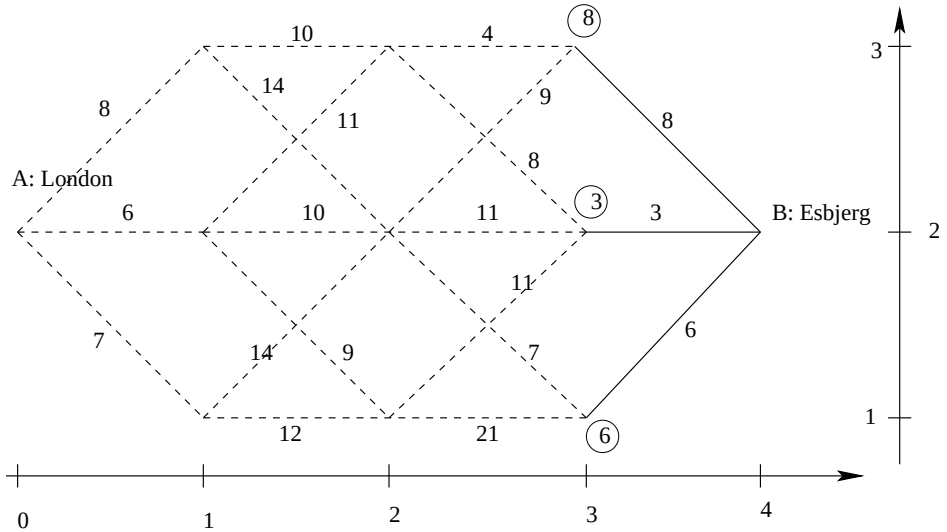
$$J = \phi_N(x_N) + \sum_{i=0}^{N-1} L_i(x_i, u_i) \qquad \text{Objective}$$

$$x_{i+1} = f_i(x_i, u_i) \qquad x_0 = \underline{x}_0 \qquad \text{Dynamics}$$

$$(x_i, u_i) \in V_i \qquad \text{Constraints}$$

$$x_N \in V_N$$

Let us start with the easy part - next to the end (i.e. $i = 3$).



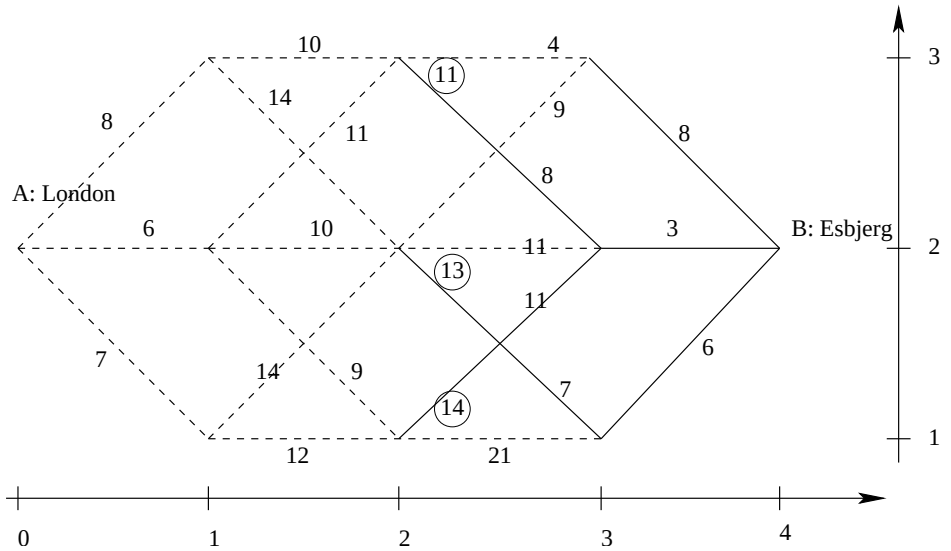
Here we easily find the optimal decisions:

$$u_3^*(x_3) = \begin{cases} -1 & \text{for } x_3 = 3 \\ 0 & \text{for } x_3 = 2 \\ 1 & \text{for } x_3 = 1 \end{cases}$$

and the optimal costs to go:

$$V_3(x_3) = \begin{cases} 8 & \text{for } x_3 = 3 \\ 3 & \text{for } x_3 = 2 \\ 6 & \text{for } x_3 = 1 \end{cases}$$

For $i = 2$ we have:



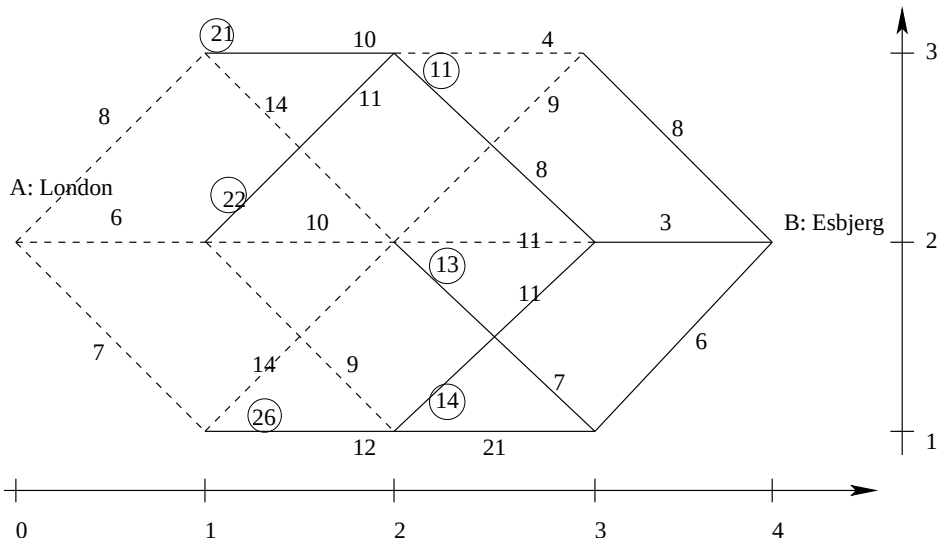
the optimal decisions:

$$u_2^*(x_2) = \begin{cases} -1 & \text{for } x_2 = 3 \\ -1 & \text{for } x_2 = 2 \\ 1 & \text{for } x_2 = 1 \end{cases}$$

and the optimal costs:

$$V_2(x_2) = \begin{cases} 11 & \text{for } x_2 = 3 \\ 13 & \text{for } x_2 = 2 \\ 14 & \text{for } x_2 = 1 \end{cases}$$

For $i = 1$ we have:



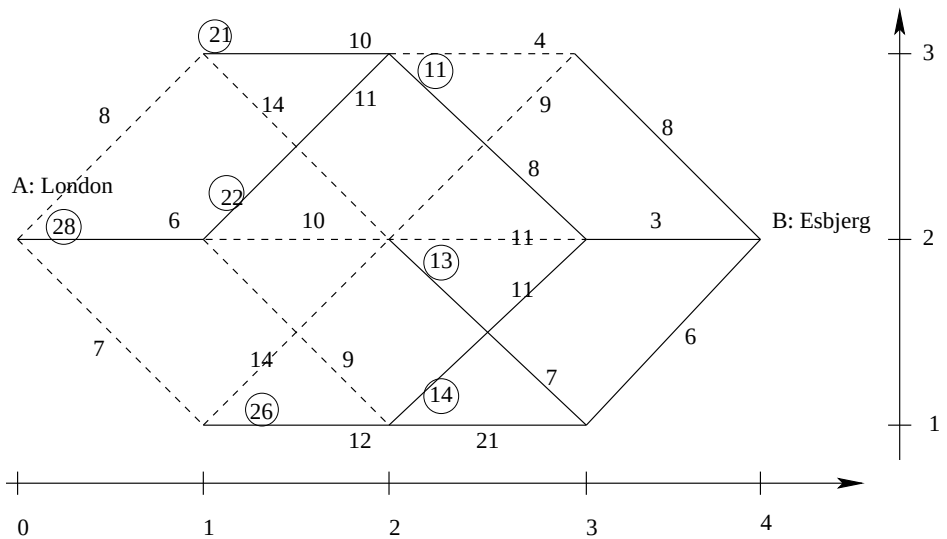
the optimal decisions:

$$u_1^*(x_1) = \begin{cases} 0 & \text{for } x_1 = 3 \\ 1 & \text{for } x_1 = 2 \\ 0 & \text{for } x_1 = 1 \end{cases}$$

and the optimal costs:

$$V_1(x_1) = \begin{cases} 21 & \text{for } x_1 = 3 \\ 22 & \text{for } x_1 = 2 \\ 26 & \text{for } x_1 = 1 \end{cases}$$

Finally, for $i = 0$ we have:



the optimal decisions:

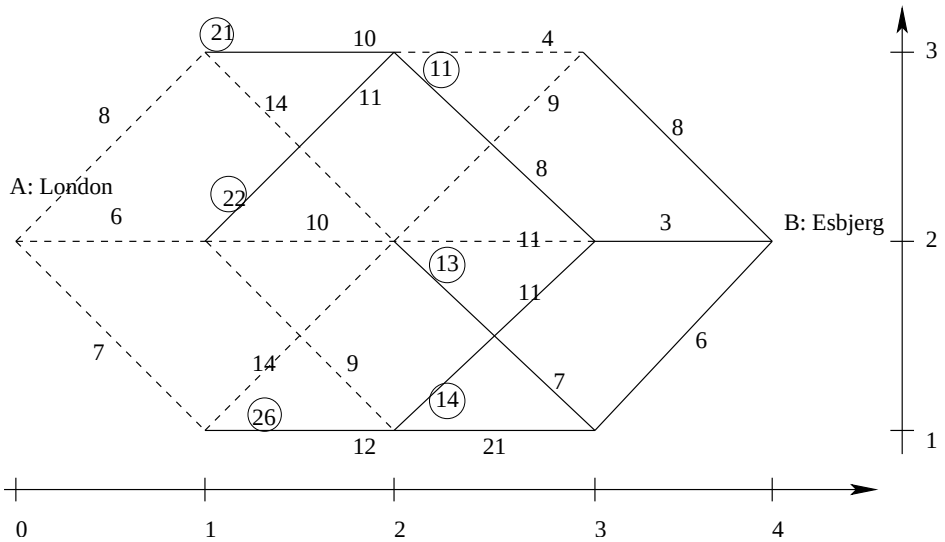
$$u_0^*(x_0) = 0 \quad \text{since} \quad x_0 = 2$$

and the optimal cost:

$$V_0(x_0) = 28 \quad \text{since} \quad x_0 = 2$$

Dynamic Programming

Let us return to $i = 1$ and see what we actually did.



We had the optimal cost to go, $V_2(x_2)$, for the next stage stored ie.

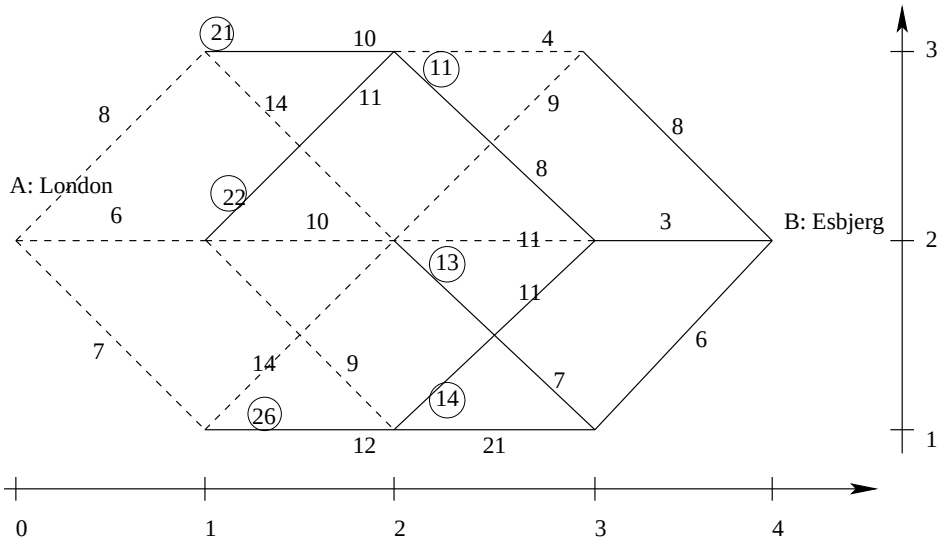
$$V_2(x_2) = \begin{cases} 11 & \text{for } x_2 = 3 \\ 13 & \text{for } x_2 = 2 \\ 14 & \text{for } x_2 = 1 \end{cases}$$

as well as the optimal decision sequence from there on .

For each possible value of x_1 ($x_1 = 1, 2, 3$), we evaluated the loss for each possible decision (e.g. $u_1 = -1, 0, 1$ for $x_1 = 2$), and minimized wrt. u_1:

$$L_1(x_1, u_1) + V_2(f_1(x_1, u_1)) \quad \text{because} \quad x_2 = f_1(x_1, u_1)$$

That minimization results in $V_1(x_1)$.



For each possible value of x_1 $(1, 2, 3)$:

$$V_1(x_1) = \min_{u_1}\Big\{ L_1(x_1, u_1) + V_2(f_1(x_1, u_1)) \Big\} \quad (x_1, u_1) \in \mathcal{V}_1$$

Or in general (for $i = 3, 2, 1, 0$):

$$V_i(x_i) = \min_{u_i}\Big\{ L(x_i, u_i) + V_{i+1}(f_i(x_i, u_i)) \Big\} \quad (x_i, u_i) \in \mathcal{V}_i$$

Free Dynamic Programming

Richard Ernest Bellman (1920-1984) was an American applied mathematician who was central to the rise of "modern control theory", i.e. state space methods in systems and control. His most celebrated contribution is dynamic programming and the principle of optimality, which both concern dividing complex decision-making problems into more, but simpler, sub-problems. A Ph.D. from Princeton, he spend the majority of his career at RAND corporation.

Free Dynamic Programming

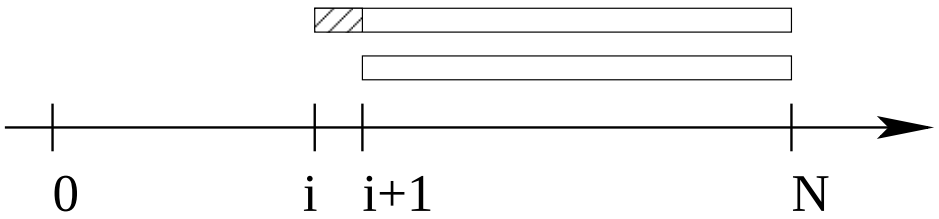
Let us now focus on finding a sequence of decisions $u_i \quad i = 0, \ , 1, \ldots \ N$ which takes the system

$$x_{i+1} = f_i(x_i, u_i) \qquad x_0 = \underline{x}_0$$

along a trajectory, such that the cost function

$$J = \phi(x_N) + \sum_{i=0}^{N-1} L_i(x_i, u_i) = J_0(x_0, u_0^{N-1})$$

is minimized.



Def.: u_i^k the sequence of decision from i to k.

The truncated performance index (the cost to go)

$$J_i(x_i, u_i^{N-1}) = \phi(x_N) + \sum_{k=i}^{N-1} L_k(x_k, u_k)$$

It is quite easy to see that

$$J_i(x_i, u_i^{N-1}) = L_i(x_i, u_i) + J_{i+1}(x_{i+1}, u_{i+1}^{N-1})$$

and that in particular

$$J = J_0(x_0, u_0^{N-1}) \qquad J_N = \phi(x_N)$$

Free Dynamic Programming

The Bellman function (the optimal cost to go) is defined as:

$$V_i(x_i) = \min_{u_i^{N-1}} J_i(x_i, u_i^{N-1})$$

and is a function of the present state, x_i, and index, i.
In particular

$$V_N(x_N) = \phi_N(x_N)$$

Theorem

The Bellman function V_i, is given by the backwards recursion

$$V_i(x_i) = \min_{u_i} \Big[\; L_i(x_i, u_i) + V_{i+1}(x_{i+1}) \; \Big]$$

with the boundary condition

$$V_N(x_N) = \phi_N(x_N)$$

Bellman equation is a **functional equation**, *gives a* **sufficient condition** *and the overall optimimum is given as $V_0(x_0) = J^*$.*

□

Free Dynamic Programming

What about the decisions?

$$u_i = arg \min_{u_i} \Big[\; L_i(x_i, u_i) + V_{i+1}(\underbrace{f_i(x_i, u_i)}_{x_{i+1}}) \; \Big] \quad = \quad u_i(x_i)$$

If a maximization problem: min $\rightarrow$ max.

Proof.

By definition we have:

$$\begin{aligned} V_i(x_i) &= \min_{u_i^{N-1}} J_i(x_i, u_i^{N-1}) \\ &= \min_{u_i^{N-1}} \left[L_i(x_i, u_i) + J_{i+1}(x_{i+1}, u_{i+1}^{N-1}) \right] \end{aligned}$$

Since u_{i+1}^{N-1} do not affect L_i we can write

$$V_i(x_i) = \min_{u_i} \left[L_i(x_i, u_i) + \min_{u_{i+1}^{N-1}} J_{i+1}(x_{i+1}, u_{i+1}^{N-1}) \right]$$

The last term is nothing but V_{i+1}, due to the definition of the Bellman function. The boundary condition is also given by definition of the Bellman function $(V_N(x_N) = \phi_N(x_N))$ that do not depend on u_N .

□

Pause

LQ problem

Simple LQ problem: The problem is bring the system

$$x_{i+1} = ax_i + bu_i \qquad\qquad x_0 = \underline{x}_0$$

from the initial state along a trajectory such the performance index

$$J = px_N^2 + \sum_{i=0}^{N-1} qx_i^2 + ru_i^2$$

is minimized.

In the boundary we have

$$V_N(x_N) = \phi(x_N) = px_N^2$$

Inspired of this, we will try the candidate function

$$V_i = s_i x_i^2$$

The Bellman equation

$$V_i(x_i) = \min_{u_i}\Big[\ \ L_i(x_i, u_i) + V_{i+1}(x_{i+1})\ \ \Big] \qquad \text{(cut an paste)}$$

gives in this special (scalar LQ) case:

$$s_i x_i^2 = \min_{u_i}\Big[\underbrace{q x_i^2 + r u_i^2}_{L_i(x_i,u_i)} + \underbrace{s_{i+1} x_{i+1}^2}_{V_{i+1}(x_{i+1})}\Big]$$

or with the state equation inserted

$$s_i x_i^2 = \min_{u_i}\Big[q x_i^2 + r u_i^2 + s_{i+1}(\underbrace{a x_i + b u_i}_{f_i(x_i,u_i)})^2\Big]$$

The minimum is obtained for

$$u_i = -\frac{a b s_{i+1}}{r + b^2 s_{i+1}} x_i$$

which inserted in the recursion results in:

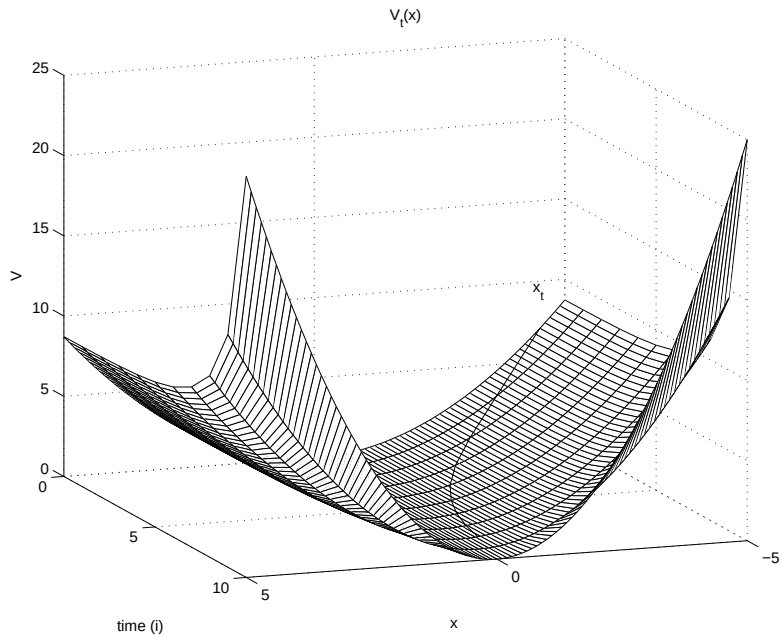
$$s_i x_i^2 = \Big[q + a^2 s_{i+1} - \frac{a^2 b^2 s_{i+1}^2}{r + b^2 s_{i+1}}\Big] x_i^2$$

The candidate function satisfies the Bellman equation if

$$s_i = q + a^2 s_{i+1} - \frac{a^2 b^2 s_{i+1}^2}{r + b^2 s_{i+1}} \qquad s_N = p$$

which is the (scalar version of the) Riccati equation. The solution: s_{i+1} and

and



Method: Guess the type of functionallity in $V_i(x)$ i.e. up to a number of parameter. Check if it satisfy the Bellman equation. This results in a (number of) recursion(s) for the parameter(s).

Euler-Lagrange and Bellman

$$V_i(x_i) = \min_{u_i} \Big[\; L_i(x_i, u_i) + V_{i+1}(x_{i+1}) \; \Big]$$

Necessary condition (stationarity):

$$0^T = \frac{\partial L_i}{\partial u_i} + \frac{\partial V_{i+1}}{\partial x_{i+1}} \frac{\partial x_{i+1}}{\partial u_i} \qquad\qquad x_{i+1} = f_i(x_i, u_i)$$

If the **costate** (or the adjoint state) is defined as the sensitivity, i.e. as:

$$\lambda_i^T \triangleq \frac{\partial V_i(x_i)}{\partial x_i} \qquad\qquad \lambda_{i+1}^T = \frac{\partial V_{i+1}}{\partial x_{i+1}}$$

then

$$0^T = \frac{\partial L_i}{\partial u_i} + \lambda_{i+1}^T \frac{\partial f_i}{\partial u_i}$$

or:

$$0^T = \frac{\partial}{\partial u_i} H_i \qquad where \qquad H_i = L_i(x_i, u_i) + \lambda_{i+1}^T f_i(x_i, u_i)$$

i.e. the stationarity condition in the the Euler-Lagrange equation.

Euler-Lagrange and Bellman

On the optimal trajectory

$$V_i(x_i) = L_i(x_i, u_i^*) + V_{i+1}(f_i(x_i, u_i^*))$$

or if we apply the chain rule

$$\lambda_i^T \triangleq \frac{\partial V_i(x_i)}{\partial x_i} = \frac{\partial L_i}{\partial x_i} + \frac{\partial V_{i+1}}{\partial x_{i+1}} \frac{\partial f_i}{\partial x_i} = \frac{\partial L_i}{\partial x} + \lambda_{i+1}^T \frac{\partial f_i}{\partial x_i}$$

or

$$\lambda_i^T = \frac{\partial}{\partial x_i} H_i \qquad \lambda_N^T = \frac{\partial}{\partial x} \phi_N(x_N)$$

i.e. the costate equation.

Pause

Constrained Dynamic Programming

Constraints:

$$u_i \in \mathcal{U}_i \qquad x_i \in \mathcal{X}_i$$

System dynamics:

$$x_{i+1} = f_i(x_i, u_i) \qquad x_0 = \underline{x}_0$$

Performance index

$$J = \phi_N(x_N) + \sum_{i=0}^{N-1} L_i(x_i, u_i)$$

Feasible state set:

$$\mathscr{D}_i = \{\ x_i \in \mathcal{X}_i \mid \exists\ u_i \in \mathcal{U}_i : \quad f_i(x_i, u_i) \in \mathscr{D}_{i+1}\ \}$$

$$\mathscr{D}_N = \mathcal{X}_N$$

Feasible decision set:

$$\mathcal{U}_i^*(x_i) = \{\ u_i \in \mathcal{U}_i : \quad f_i(x_i, u_i) \in \mathscr{D}_{i+1}\ \}$$

Theorem

Bellman equation:

$$V_i(x_i) = \min_{u_i \in \mathcal{U}_i^*} \Big[\; L_i(x_i, u_i) + V_{i+1}(x_{i+1}) \; \Big]$$

$$V_N(x_N) = \phi_N(x_N)$$

Feasible state set:

$$\mathscr{D}_i = \{ \; x_i \in \mathcal{X}_i \mid \exists \; u_i \in \mathcal{U}_i : \quad f_i(x_i, u_i) \in \mathscr{D}_{i+1} \; \}$$

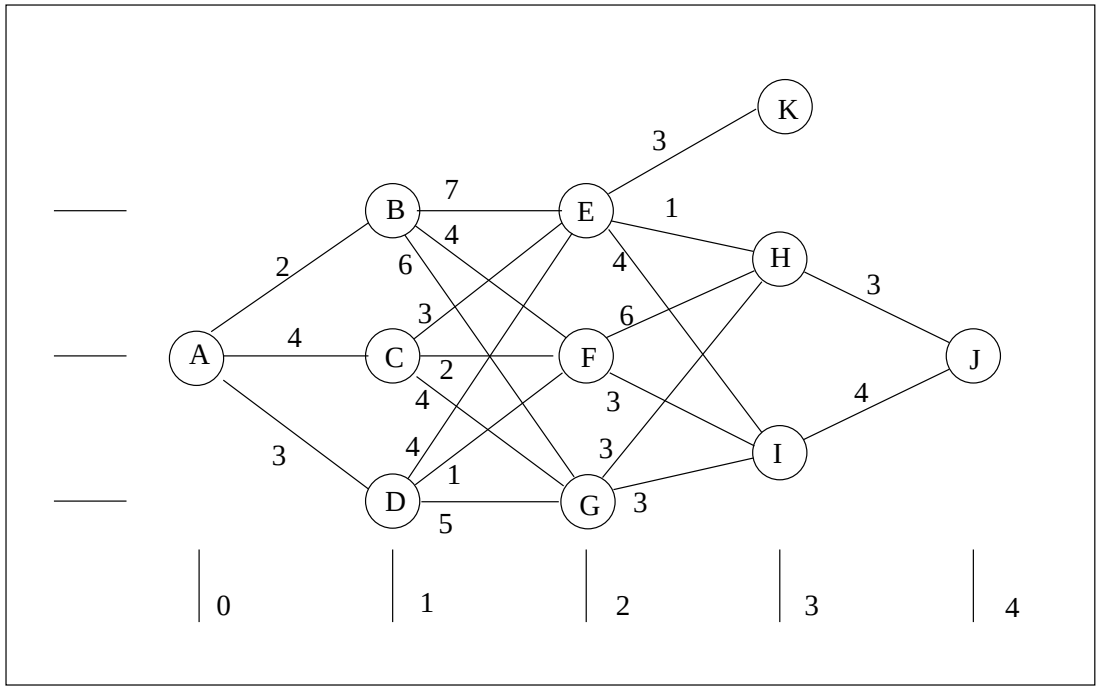
$$\mathscr{D}_N = \mathcal{X}_N$$

Feasible decision set:

$$\mathcal{U}_i^*(x_i) = \{ \; u_i \in \mathcal{U}_i : \quad f_i(x_i, u_i) \in \mathscr{D}_{i+1} \; \}$$

□

Stagecoach problem II



It is easy to realize that

$$\mathcal{X}_4 = \{J\} \qquad \mathcal{X}_3 = \{K, H, I\} \qquad \mathcal{X}_2 = \{E, F, G\}$$
$$\mathcal{X}_1 = \{B, C, D\} \qquad \mathcal{X}_0 = \{A\}$$

However, since there is no path from K to J

$$\mathcal{D}_4 = \{J\} \qquad \mathcal{D}_3 = \{H, I\} \qquad \mathcal{D}_2 = \{E, F, G\}$$
$$\mathcal{D}_1 = \{B, C, D\} \qquad \mathcal{D}_0 = \{A\}$$

Optimal stepping (DD)

. Consider the system

$$x_{i+1} = x_i + u_i \qquad x_0 = 2$$

the performance index

$$J = x_N^2 + \sum_{i=0}^{N-1} x_i^2 + u_i^2 \qquad \text{with} \qquad N = 4$$

and the constraints

$$u_i \in \{-1,\ 0,\ 1\} \qquad x_i \in \{-2,\ -1,\ 0,\ 1,\ 2\}$$

Firstly, we establish $V_4(x_4)$ as in the following table

x_4	V_4
-2	4
-1	1
0	0
1	1
2	4

$$V_4(x_4) = \phi_4(x_4) = x_4^2$$

The combination of x_3 and u_3 determines the following state

$$x_4 = x_3 + u_3$$

and consequently the $V_4(x_4)$ contribution.

x_4	u_3		
x_3	-1	0	1
-2	-3	-2	-1
-1	-2	-1	0
0	-1	0	1
1	0	1	2
2	1	2	3

$V_4(x_4)$	u_3		
x_3	-1	0	1
-2	∞	4	1
-1	4	1	0
0	1	0	1
1	0	1	4
2	1	4	∞

The combination of x_3 and u_3 also determines the instantaneous loss $L_3 = x_3^2 + u_3^2$.

L_3	u_3		
x_3	-1	0	1
-2	5	4	5
-1	2	1	2
0	1	0	1
1	2	1	2
2	5	4	5

If we add up the instantaneous loss and V_4 we have a tableau in which we for each possible value of x_3 can perform the minimization in

$$V_3(x_3) = \min_{u_3} \left[L_3(x_3, u_3) + V_4(x_3 + u_3) \right]$$

and determine the optimal value for the decision and the Bellman function (as function of x_3).

$L_3 + V_4$	u_3			V_3	u_3^*
x_3	-1	0	1		
-2	∞	8	6	6	1
-1	6	2	2	2	0,1
0	2	0	2	0	0
1	2	2	6	2	-1,0
2	6	8	∞	6	-1

Knowing $V_3(x_3)$ we have one of the components for $i = 2$. In this manner we can iterate backwards and finds:

$L_2 + V_3$	u_2			V_2	u_2^*
x_2	-1	0	1		
-2	∞	10	7	7	1
-1	8	3	2	2	1
0	3	0	3	0	0
1	2	3	8	2	-1
2	7	10	∞	7	-1

$L_1 + V_2$	u_1			V_1	u_1^*
x_1	-1	0	1		
-2	∞	11	7	7	1
-1	9	3	2	2	1
0	3	0	3	0	0
1	2	3	9	2	-1
2	7	11	∞	7	-1

$L_0 + V_1$	u_0			V_0	u_0^*
x_0	-1	0	1		
-2	∞	11	7	7	1
-1	9	3	2	2	1
0	3	0	3	0	0
1	2	3	9	2	-1
2	7	11	∞	7	-1

With $x_0 = 2$ we can trace forward and find the input sequence $-1,\ -1,\ 0,\ 0$ which give (an optimal) performance equal 7. **Sensitivity. Steering vs. feedback.**

Optimal stepping (DD) II - EPC

Consider the system from previous example, but with the constraints that $x_4 = 1$.
System

$$x_{i+1} = x_i + u_i \qquad x_0 = 2 \qquad x_4 = 1$$

Performance index

$$J = x_4^2 + \sum_{i=0}^{3} x_i^2 + u_i^2$$

Constraints:

$$u_i \in \{-1,\ 0,\ 1\} \qquad x_i \in \{-2,\ -1,\ 0,\ 1,\ 2\}$$

x_4	V_4
-2	∞
-1	∞
0	∞
1	1
2	∞

$L_3 + V_4$	u_3			V_3	u_3^*
x_3	-1	0	1		
-2	∞	∞	∞	∞	
-1	∞	∞	∞	∞	
0	∞	∞	2	2	1
1	∞	2	∞	2	0
2	6	∞	∞	6	-1

$L_2 + V_3$	u_2			V_2	u_2^*
x_2	-1	0	1		
-2	∞	∞	∞	∞	
-1	∞	∞	4	4	1
0	∞	2	3	2	0
1	4	3	8	3	0
2	7	10	∞	7	-1

$L_1 + V_2$	u_1			V_1	u_1^*
x_1	-1	0	1		
-2	∞	∞	9	9	1
-1	∞	5	4	4	1
0	5	2	4	2	0
1	4	4	9	4	-1
2	8	11	∞	8	-1

$L_0 + V_1$	u_0			V_0	u_0^*
x_0	-1	0	1		
-2	∞	13	9	9	1
-1	11	5	4	4	1
0	5	2	5	2	0
1	4	5	10	4	-1
2	9	12	∞	9	-1

With $x_0 = 2$ we can iterate forward and find the optimal input sequence $-1,\ -1,\ 0,\ 1$ which is connected to a performance index equal 9.

Reading guidance

DO: chapter 6, page 73-83