Optimal Control and Dynamic Programming

TU/e

4SC000 Q2 2017-2018

Duarte Antunes
Introduction
Outline

• Course info
• Introduction to optimal control and applications
• Dynamic programming algorithm
Course information

Teaching staff

- Lecturer: Duarte Antunes (d.antunes@tue.nl)
- Assistants: Eelco van Horssen (e.p.v.horssen@tue.nl)
  Ruben di Filippo (r.di.filippo@student.tue.nl)

Grading

- 1 Exam and 3 homework assignments to be submitted via Matlab cody coursework*
- Assignments can be solved individually or in a group (max 4 people); select group via canvas.
- If solved in a group, presence at BZ 7 & 14 is mandatory to discuss individual grade of assignments 1, 2 respectively. Peer assessment of homework 3 should be sent by Feb 4th.
- Contributions to the final grade: each of the 3 homework assignments, 40/3%; exam, 60%.

Question hours

<table>
<thead>
<tr>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>12h30-13h30</td>
<td>17h30-18h30</td>
<td>15h45-17h30</td>
<td>12h45-13h45</td>
<td>10h45-12h30</td>
</tr>
<tr>
<td>Eelco</td>
<td>Ruben</td>
<td>BZ</td>
<td>Duarte</td>
<td>BZ</td>
</tr>
<tr>
<td>GEM-Z 0.55</td>
<td>GEM-Z 0.55</td>
<td>Paviljoen U46</td>
<td>GEM-Z -1.139</td>
<td>Paviljoen U46</td>
</tr>
</tbody>
</table>

*since there are questions hours every working day, no further appointments will be scheduled and we try to avoid answering questions via e-mail.

*https://coursework.mathworks.com/, students will receive an email to register
# Course schedule

**Lectures (L):** Wednesdays 13h45-15h30 (LUNA 1.050), Fridays 8h45-10h30 (Paviljoen B2).

**Guided self-study (BZ):** Wednesdays 15h45-17h30 (Paviljoen U46)
Fridays 10h45-12h30 (Paviljoen U46)

**Deadlines:** PSI: Dec. 5th, 23h45; PSII: Jan 11rd 23h45; PSIII: Feb 4th, 23h45.

**Exam:** February 1st, 13h30-16h30, retake: April 12, 18h00-21h00.

<table>
<thead>
<tr>
<th></th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>November</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>14</td>
<td>L1 BZ1</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>21</td>
<td>L3 BZ3</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>28</td>
<td>L5 BZ5</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td><strong>December</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>L7 BZ7*</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>12</td>
<td>L9 BZ9</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>19</td>
<td>L11 BZ11</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td><strong>January &amp; February</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>9</td>
<td>L13 BZ13</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>16</td>
<td>L15 BZ15</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>29</td>
<td>30</td>
<td>31</td>
<td>Exam</td>
<td>1</td>
</tr>
</tbody>
</table>

*In BZ7 and BZ14 the grade of each member of a group pertaining to homework 1 and 2, respectively, will be discussed.*
Course material

Main course material

• Slides and Problem sets

Further reading


• Other reference books [1]-[10]

*Slides and video lectures available at http://www.athenasc.com/dpbook.html*
# Outline of the course

<table>
<thead>
<tr>
<th>Topic</th>
<th>Lectures</th>
</tr>
</thead>
</table>
| I Discrete optimization problems | 1. Introduction and the dynamic programming algorithm  
2. Stochastic dynamic programming  
3. Shortest path problems in graphs  
4. Bayes filter and partially observable Markov decision processes |
| II Stage decision problems | 5. State-feedback controller design for linear systems -LQR  
6. Optimal estimation and output feedback- Kalman filter and LQG  
7. Discretization  
8. Discrete-time Pontryagin’s maximum principle  
9. Approximate dynamic programming |
| III Continuous-time optimal control problems | 10. Hamilton-Jacobi-Bellman equation and deterministic LQR in continuous-time  
11. Linear quadratic control in continuous-time - LQR/LQG  
12. Frequency-domain properties of LQR/LQG  
13. Pontryagin’s maximum principle I  
14. Pontryagin’s maximum principle II |
| | 15 & 16. Revision/sample exam |
Position in the MSc programs

Systems and control oriented programs

- Systems and control, Mechanical and Electrical Engineering with control specialisation
- Clear track
  Q1 System theory for control, Q2 Optimal control, Q3 Model predictive control
- Optimal control is one of the cornerstones of control systems theory

Other programs

- Optimal control and dynamic programming is very broad and may be useful for you.
- For example for the Automotive students: optimal control appears in many automotive applications, such as optimization of powertrains, optimal power management in hybrid vehicles, etc.
Background

Matlab

• Nice intro: https://matlabacademy.mathworks.com/.
• Best way to learn: read Matlab documentation and gain experience.

System theory

• Basic knowledge of concepts such as state-space representation, observability, controllability is useful
• Course ‘System Theory for Control’ taught at TU/e is enough.
• If you have not taken the course, suggestion for a book: ‘Linear systems theory’, 2009, João Hespanha.

Optimization

• Notions of gradient, convex functions, constrains, see Appendix B of Bertsekas’ book.

Probability theory

• Basic notions, see Appendix C of Bertsekas’ book.
Outline

• Course info
• Introduction to optimal control and applications
• Dynamic programming algorithm
Optimal control

Optimality

• Useful design principle in many engineering contexts (optimize efficiency of a refrigerator, minimize the fuel consumption of a car, etc.).
• Nature is described by laws derived from optimality principles.
• We optimize every day to make decisions (true?).

Optimal control

• Deals with problems in which optimal decisions or control actions are pursued over a time period in order to reach final and intermediate goals.
• Arises in the control of physical systems (e.g. mechanical, electrical, biological) and in many other contexts (e.g., economics, computer science, and game theory).
Optimal control vs static optimization

Static optimization

- Determine one optimal decision.
- Examples: decide on the price of a product, determine the slope of a straight line which best fits data, etc.
Optimal control

- Determine several optimal decisions over time.
- Decisions are functions of state, i.e., a control law to cope with disturbances.
- Examples: driving a car/bike in a race, positioning the tip of a robot arm in the presence of disturbances, playing chess, etc.

Disturbance at time $t = 2$

$\theta(t) \neq \tilde{\theta}(t)$  $t \geq 2$
Optimal control formulation

Dynamic model

• Specifies the rules of the problem or the equations of the physical system.
• **State**: summarizes relevant information to make future decisions.
  • **Control actions**: influence the evolution of the state over time.
  • State evolution may be deterministic or stochastic (driven by disturbances).

Cost function

• Encapsulates the goals to be achieved in the problem.
• Typically additive over time and by convention should be minimized.

Goal: find a control policy which minimizes the cost

• **Policy**: set of functions mapping the state at each instant of time to an action.
• Related problem: compute an optimal path/trajectory consisting of optimal decisions over time for a given initial state.
Optimal control problems

Three classes of problems will be considered in the course

<table>
<thead>
<tr>
<th>Type of Problem</th>
<th>Time</th>
<th>State Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete optimization problems*</td>
<td>discrete</td>
<td>discrete</td>
</tr>
<tr>
<td>Stage decision problems</td>
<td>discrete</td>
<td>general</td>
</tr>
<tr>
<td>Continuous-time optimal control problems</td>
<td>continuous</td>
<td>general</td>
</tr>
</tbody>
</table>

Some applications are discussed next and more applications later. However there are many others - see Appendix B.
Applications

Traditional process control
- controlling an inverted pendulum, mass-spring damper, double integrator, quadcopter, etc.

Aerospace
- minimum-fuel launch of a satellite, etc.

Operational research, management, finance
- inventory control, control of a queue, control of networks (data, traffic, etc.), etc.

Computer Science
- shortest path in graphs, scheduling, selection problems, among others.

Other fields
- Computational biology, automotive, games, many others.

Next slides address some applications treated in the course, where we will consider also cases where uncertainty is present.
Specified by a transition diagram with $h - 1$ decision stages

- **Dynamic model**: circles indicate states at each of $h$ stages; arrows indicate actions for each state which lead to states at next stages.
- **Costs** $c_{i,j}^k$ are associated with actions $j$ for each state $i$ at each stage $k$; for the terminal stage $h$ the costs $c_{i}^h$ depend only on the state $i$. 

---

Discrete Optimization Problems
Discrete Optimization Problems

Challenges

(i) Determine an optimal path for a given initial state which minimizes the sum of costs incurred at every stage (including the terminal stage).

(ii) Determine an optimal policy specifying for each state the first decision of the optimal path from that state to the terminal stage.
Inventory control

How to manage the supply of products in a shop? Overstock is prejudicial (physical space limitations, technological obsolescence, etc.) and under stock undermines sales.
Shortest paths in graphs

What is the shortest distance from Bucharest to Lugoj?

*Source: Artificial intelligence: a modern approach, Stuart J. Russel, Peter Novig, 3rd edition, 2016*
What is the shortest path for a robot to go from point A to B?
Games

How to make profit in expectation in a game such as blackjack?

As portrayed in the movie ‘21’ the MIT blackjack team had an answer to this problem (using optimal control?). The same movie unveils the principle to achieve this, using a famous game show problem

https://www.youtube.com/watch?v=Zr_xWfThjJ0
Stage decision problems

Dynamic model

\[ x_{k+1} = f_k(x_k, u_k) \quad k \in \{0, \ldots, h - 1\} \]

Cost function

\[ \sum_{k=0}^{h-1} g_k(x_k, u_k) + g_h(x_h) \]

Goals

(i) find policy \( \pi = \{\mu_0, \ldots, \mu_{h-1}\} \)

that leads to the minimum cost for every initial condition.

(ii) find path \( \{(x_0, u_0), (x_1, u_1), \ldots, (x_{h-1}, u_{h-1})\} \)

that leads to the minimum cost for a given initial condition.
Stage decision problems

Generalization of discrete optimization problem considering general state and input spaces, e.g., $\mathbb{R}^n$
Digital control

Prime application: how to design a digital controller for a physical system?

Several variants: full state is available or only an output, system can have disturbances or not, etc.
Mixing

How to mix two fluids in minimum time?

Actuation: 4 possible rotations decided once every $h$ seconds

$\theta_{in} = 3\pi \, \text{rad}$

$\theta_{in} = -3\pi \, \text{rad}$

$\theta_{out} = \pi \, \text{rad}$

$\theta_{out} = -\pi \, \text{rad}$

Control law

Camera images
Digital control of a unicycle robot

Given a unicycle robot with constraints on speed and rotation rate and controlled digitally, how to do a curve maneuver in minimum time?
Continuous-time optimal control problems

Dynamic model

\[ \dot{x}(t) = f(t, x(t), u(t)), \quad x(0) = x_0, \quad t \in [0, h] \]

Cost function

\[ \int_{0}^{h} g(t, x(t), u(t)) \, dt + g_h(x(h)) \]

Goals

- Find a feedback policy \( u(t) = \mu(t, x(t)) \) minimizing the cost function for every initial condition.
- Find a control input \( u(t), \quad t \in [0, h] \), minimizing the cost function for a given initial condition.
Continuous-time optimal control problems

Most applications in control systems: motion control, aerospace, etc.
Minimum energy control

How to move a motion system described by a linear equation from point A to point B with minimum energy?

\[
\min_u \int_0^T g(u(t))dt
\]

\[
\dot{x}(t) = Ax(t) + Bu(t)
\]

\[
x(0) = x_0
\]

\[
x(T) = x_{desired}
\]
Minimum time control

How to move a (linearized model of a) quadcopter from one hovering position to another one in minimum time?

\[ \dot{x}(t) = Ax(t) + Bu(t) \]

subject to

\[ x(0) = x_0 \]

\[ x(T) = x_{\text{desired}} \]

\[ \min T \]
Energy management of Hybrid Electric Vehicles

Hybrid electric vehicles have a battery where energy can be stored (e.g. during braking). Given a drive cycle, how to design the power slip between the battery and the internal combustion engine to minimize fuel consumption?
Outline

• Course info
• Introduction to optimal control and applications
• Dynamic programming algorithm
Dynamic programing

• Dynamic programming is an approach to solve optimal control problems.
• It allows to find functions mapping states into actions. These functions we call policies or control laws.
• One can use these functions to control a system in the presence of disturbances.
• It also allows to compute optimal paths/trajectories, although, as we shall see, other methods might be more efficient.
The principle of optimality

Example: shortest route from Eindhoven to Paris passes through Antwerp. Then the piece of the route from Antwerp to Paris is the shortest route between the two cities.
The principle of optimality

The tail of an optimal path is also optimal

- Given an optimal path for a discrete optimization problem from stage $0$ to stage $h$ consider the state $x_j$ at stage $j$ belonging to the optimal path.

- Then the decisions along the optimal path from stages $j$ to $h$ are also optimal for the discrete optimization problem with initial stage $j$, initial state $x_j$ and final stage $h$.

- The principle of optimality also holds for stage decision problems and continuous-time optimal control problems and is the basis of the dynamic programming algorithm.
Proof of the principle of optimality

1. Consider an optimal path from stage 0 to stage $h$.

![Diagram 1](image1)

$$\text{cost} = \text{cost}_{[0,j]} + \text{cost}_{[j,h]}$$

2. Suppose that the piece of the path from stage $j$ to stage $h$ is not optimal.

![Diagram 2](image2)

$$\text{cost}_{[j,h]} < \text{cost}_{[j,h]}$$

3. Then there exists a path with smaller cost from stage 0 to stage $h$ - contradiction!

![Diagram 3](image3)

$$\text{cost}_{[0,j]} + \text{cost}_{[j,h]} < \text{cost}_{[0,j]} + \text{cost}_{[j,h]}$$
The dynamic programming algorithm

Main idea
- Find first the optimal policy and paths from stage \( j \) to \( h \), and then use these to compute the optimal policy and paths from stage \( j - 1 \) to stage \( h \) (principle of optimality).

The dynamic programming algorithm for discrete optimization problems:

1. Start at the final decision stage and denote the terminal cost by cost-to-go at stage \( h \),
\[
J_h(i) = c_h^i
\]
2. For every state \( i \) at stage \( k = h - 1 \) compute the optimal action \( j \) as follows
\[
\min_{j \in \text{actions/arrows}} c_{ij}^k + J_{k+1}(\text{state at stage } k+1 \text{ when } j \text{ is picked})
\]
Denote the minimum by cost-to-go at stage \( k \), \( J_k(i) \).
3. Repeat (2) for stages \( k \in \{h - 2, h - 3, \ldots, 1, 0\} \) moving backwards.

Then, the function which maps each state to the action obtained in (2) is an optimal policy.
Example

### Iteration 1 - Stage 3

<table>
<thead>
<tr>
<th>State</th>
<th>Cost-to-go</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(\min{0 + 4, 5 + 0} = 4)</td>
</tr>
<tr>
<td>2</td>
<td>(\min{4 + 0, 2 + 4} = 4)</td>
</tr>
<tr>
<td>3</td>
<td>(\min{5 + 0, 2 + 4} = 5)</td>
</tr>
</tbody>
</table>
Example

Iteration 2 - Stage 2

<table>
<thead>
<tr>
<th>State</th>
<th>Cost-to-go</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 + 4 = 5</td>
</tr>
<tr>
<td>2</td>
<td>min{2 + 4, 5 + 4} = 6</td>
</tr>
<tr>
<td>3</td>
<td>min{1 + 5, 3 + 4} = 6</td>
</tr>
<tr>
<td>4</td>
<td>min{4 + 5, 3 + 4} = 7</td>
</tr>
</tbody>
</table>
Example

Iteration 3 - Stage 1

<table>
<thead>
<tr>
<th>State</th>
<th>Cost-to-go</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\min{4 + 6, 3 + 5} = 8$</td>
</tr>
<tr>
<td>2</td>
<td>$\min{0 + 6, 1 + 6} = 6$</td>
</tr>
<tr>
<td>3</td>
<td>$\min{1 + 7, 3 + 6, 1 + 6} = 7$</td>
</tr>
</tbody>
</table>
Example

Iteration 4 - Stage 0

<table>
<thead>
<tr>
<th>State</th>
<th>Cost-to-go</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\min{1 + 8, 4 + 6} = 9$</td>
</tr>
<tr>
<td>2</td>
<td>$\min{2 + 7, 1 + 6} = 7$</td>
</tr>
</tbody>
</table>
Optimal policy and optimal paths

Initial transition diagram

Optimal policy

- While running the DP algorithm, for each state at each stage a decision is made to compute the cost-to-go. That decision is precisely the decision specified by the optimal policy.

Optimal path

- For a given initial state, follow the arrows leading to the final stage. This is the optimal path.
- The cost-to-go at stage 0 of that initial state coincides with the cost of the optimal path.
Non-uniqueness

The optimal policy and the optimal paths may not be unique

If more than one option has the same cost while running the dynamic programming algorithm, simply pick one of the options. At the end one optimal policy is obtained (while several may be optimal).
Inventory control

Controlling the supply of one product

- Dynamic model
  
  \[ x_{k+1} = \max\{x_k + u_k - d_k, 0\} \]
  
  \[ u_k \in \{0, 1, \ldots, N - x_k\} \]

- Cost
  
  \[ \sum_{k=0}^{h-1} g_k(x_k, u_k) + g_h(x_h) \]

\[ g_k(x_k, u_k) = (c_1(x_k) + cu_k + c_{tr}\|u_k\|_0) - p \min\{d_k, x_k + u_k\} \]

\[ x_k \quad \text{number of items} \]
\[ N \quad \text{capacity} \]
\[ u_k \quad \text{supply} \]
\[ d_k \quad \text{demand} \]
\[ g_h \quad \text{terminal cost} \]
\[ c_1 \quad \text{storage cost} \]
\[ p \quad \text{selling price} \]
\[ c \quad \text{purchase price} \]
\[ c_{tr} \quad \text{transportation price} \]

\[ \|u_k\|_0 = \begin{cases} 0 & \text{if } u_k = 0 \\ 1 & \text{if } u_k \neq 0 \end{cases} \]
Formulation as discrete optimization problem

Transition diagram

- circles at each stage indicate number of items, supplies determine transitions

Stage 0

Stage 1

Stage h

\[ u_k = N - i \]

\[ u_k = d_k \]

\[ u_k = 0 \]

Stage k

\[ i \]

\[ i - d_k \]

\[ N - d_k \]
Inventory control

\[ x_{k+1} = \max\{x_k + u_k - d_k, 0\} \]

\[ u_k \in \{0, 1, \ldots, N - x_k\} \]

\[ \sum_{k=0}^{h-1} g_k(x_k, u_k) + g_h(x_h) \]

\[ g_k(x_k, u_k) = c_1(x_k) + cu_k + c_{tr} \| u_k \|_0 - p \min\{d_k, x_k + u_k\} \]

- number of stages: \( h = 4 \)
- capacity: \( N = 4 \)
- demand: \( d_0 = d_1 = 2, d_2 = d_3 = 1 \)
- selling price: \( p = 10 \)
- purchase price: \( c = 5 \)
- transportation price: \( c_{tr} = 0.5 \)
- storage cost: \( c_1(i) = 0.2i, i \in \{0, \ldots, N\} \)
- terminal cost: \( g_4(i) = -r_{i+1}, i \in \{0, \ldots, 4\}, \quad r = [0 4.8 9.6 14.4 19.2] \)

What are the optimal supplies for a zero initial inventory?
Some iterations

Stage 3

Stage 4

Stage 2

Stage 3

Stage 3

Stage 2

Stage 3

Stage 4
Final policy and optimal path

$\begin{align*}
    d_0 &= 2 & u_0 &= -48.1 \\
    d_1 &= 2 & u_1 &= -38.6 \\
    d_2 &= 1 & u_2 &= -28.2 \\
    d_3 &= 1 & u_3 &= -23.6 \\
\end{align*}$

Stage 0

Stage 1

Stage 2

Stage 3

Stage 4

Cost for a zero initial inventory $-28.4$
Historical note

- Dynamic programming was proposed in the 1940s by Richard E. Bellman

‘I was intrigued by dynamic programming. It was clear to me that there was a good deal of good analysis there. Furthermore, I could see many applications. It was a clear choice. I could either be a traditional intellectual, or a modern intellectual using the results of my research for the problems of contemporary society.’

R. E. Bellman (1920-1984)
Concluding remarks

Summary

• Optimal control: Determine several optimal decisions over time and as a function of the state.
• Optimal control problems: three classes (discrete optimization, stage-decision, continuous-time control), many applications.
• Dynamic programming: Optimal decisions computed from the end to the initial stage.

After this lecture, you should be able to:

• Apply the dynamic programming algorithm.
• Solve deterministic inventory control problems.
Appendix A

Inventory control with uncertainty
Coping with disturbances

In the context of the example of slides 40-44

What if the demand is \( d_2 = 2 \) instead of the expected \( d_2 = 1 \)?

- The state at stage 3 is then \( x_3 = 0 \) instead of \( x_3 = 1 \).
- Open loop: blindly pick \( u_3 = 0 \) as initially planned.
- Closed-loop (using DP policy): pick \( u_3 = 1 \).
Open loop vs closed loop

Costs

- **Open loop**
  \[ g_0(0, 4) + g_1(2, 0) + g_2(0, 2) + g_3(0,0) + g_4(0) \]
  
  \[
  0.5 - 19.6 + (-9.5) + 0 + 0 = -28.6
  \]

- **Closed loop**
  \[ g_0(0, 4) + g_1(2, 0) + g_2(0, 2) + g_3(0,1) + g_4(0) \]
  
  \[
  0.5 - 19.6 + (-9.5) + (-4.5) + 0 = -33.1
  \]

Expected cost if

\[ \text{Prob}[d_2 = 1] = 0.5, \text{Prob}[d_2 = 2] = 0.5 \]

- **Open loop**
  \[ 0.5 \times (-28.4) + 0.5 \times (-28.6) = -28.5 \]

- **Closed loop**
  \[ 0.5 \times (-28.4) + 0.5 \times (-33.1) = -30.75 \]
Appendix B

References and applications
References

Textbooks


Numerical methods

References

Applications

Aerospace


Biomedicine and sequential alignment of DNA


Power Systems

References

Applications

Operational research and inventory control


Finance and economics


Computer science and scheduling problems


Automotive

References

History


Seminal papers

[S1] Anonymous, "Letter sent to Charles Montague, President of the Royal Society, where two mathematical problems proposed by the celebrated Johann Bernoulli are solved". *Acta Eruditorum Lipsi*, (1697) 223.


