

Systems and Control Theory Part 2: Optimal Control

Lecture notes of lectures 7-12
L.G. van Willigenburg

See also: Black Board

7. Optimal control and optimal feedback of non-linear systems.	2
7.1 Introduction	2
7.2 Optimal control.....	2
7.3 Optimal control and optimal feedback	4
7.4 Summary and implications	5
8. Formulation and classification of optimal control problems	8
8.1 Formulation of optimal control problems: an example.....	8
8.2. Formulation and classification of optimal control problems.....	12
8.3 Examples of how to translate verbal control objectives and constraints into mathematical cost functions and constraints.	16
8.4 Summary.....	18
9. Solution of optimal control problems by the calculus of variations and Lagrange theory.	19
9.1 Constrained and unconstrained function minimisation: Lagrange theory.	19
9.2 First derivatives of functions of vectors and vector functions.	24
9.3 Solution of the general optimal control problem.....	28
9.4 Summary & Pontryagin's Minimum Principle	35
10. Gradient algorithms to solve optimal control problems. LQ and LQG feedback design.	37
10.1 Gradient algorithms to solve optimal control problems	37
10.2 The Mayer formulation of optimal control problems.	40
10.3 LQ and LQG feedback design.....	41
10.4 Selection and interpretation of the LQ and LQG design matrices	45
10.5 Time-invariant infinite horizon LQ problems and stability.....	48
11. Transformation of optimal control problems into constrained function minimization problems.....	50
11.1 Constrained function minimization.....	50
11.2 Transformation of optimal control problems into constrained function minimization problems by control parameterization.	55
11.3 Digital control systems and control parameterization.....	60
12. Control and state parameterization by Chebyshev polynomials. Actively and passively adaptive control systems.	63
12. 1 Control and state parameterization by polynomials and other time functions.	64
12.2 Control and state parameterization by Chebyshev polynomials.....	66
12.3 Substitution of possible solutions into differential equations	68
12.4 Interpretation and computation of co-states.....	69
12.5 When models are of insufficient quality: Passively and actively adaptive control	70
References.....	71

7. Optimal control and optimal feedback of non-linear systems.

7.1 Introduction

The first landing on the moon should have been performed by an optimal controller but finally it was performed manually by the astronauts! So what does the word optimal mean in terms of control?

After reading this chapter the reader should be able to:

- 1) Explain the difference between control computation and feedback
- 2) Explain why feedback is necessary only if the system description is imperfect
- 3) Recognise the importance of accurate system knowledge for control system design
- 4) Separate the design of a control system into a control computation and a computation of how the feedback should take place (feedback design)
- 5) Know what optimal means in terms of control and feedback

7.2 Optimal control

To answer what optimal means in terms of control, first consider a control problem where based on a systems model we have to determine the control inputs $u(t)$ such that the system behaviour $x(t)$ meets our requirements, see figure 7.1.

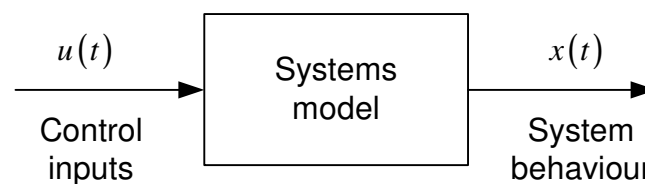


Figure 7.1: Control and behaviour related by a systems model

In the case of optimal control we have a quantitative mathematical systems model in state-space form.

$$\dot{x} = f(x, u, d, p, t), \quad x \in R^{n_x}, \quad u \in R^{n_u}, \quad d \in R^{n_d}, \quad p \in R^{n_p} \quad (7.1)$$

The four column vectors $x \in R^{n_x}$, $u \in R^{n_u}$, $d \in R^{n_d}$, $p \in R^{n_p}$ contain respectively the state variables the control inputs or control variables and the external inputs or external variables and the parameters or constants of the system. The behaviour of

the system is fully determined by $x(t)$. The control objectives are translated in a quantitative cost function the outcome of which depends on the behavior $x(t)$ and the control $u(t)$. Since the behavior $x(t)$ depends on the control $u(t)$ the cost function depends totally on the control $u(t)$: $J(u(t))$. Based on the state-space model and the cost function an optimal control $u^*(t)$ can be computed which minimizes this cost function. So $J(u^*(t))$ is the minimum of $J(u(t))$. So optimal relates to the systems model and the cost function. Associated to the optimal control $u^*(t)$ is the optimal system behaviour $x^*(t)$, see figure 7.2.

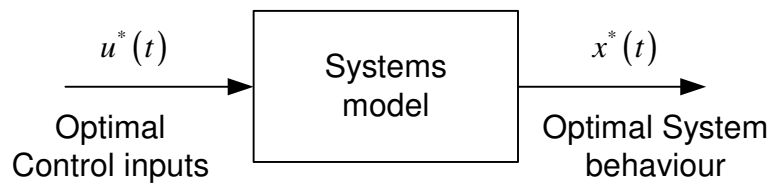


Figure 7.2: Optimal control and optimal system behaviour.

Now if we supply the computed optimal control $u^*(t)$ to the real system will the system behave optimally, see figure 7.3?

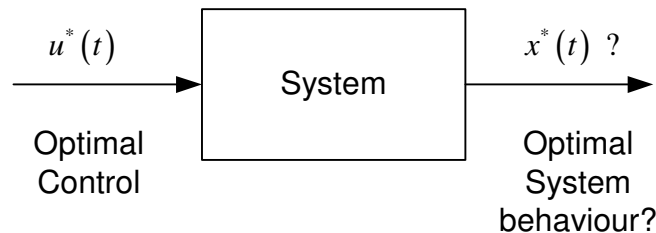


Figure 7.3: Optimal system behaviour?

The most simple and straightforward answer is: yes! And this is true of course but only if the model of the system is entirely correct (and also the initial state which we have not considered so far). If not, in general the system behaviour will deviate from the desired behaviour, see figure 7.4.

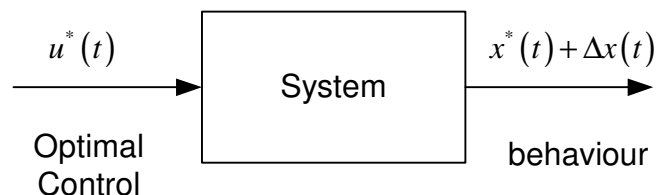


Figure 7.4: Deviations $\Delta x(t)$ if the systems model (and/or the initial state) is not entirely correct.

Then the obvious next question is: can we do something to prevent or limit these deviations? The answer is: we cannot prevent deviations but through feedback we can limit these deviations. Feedback implies that we have to perform measurements.

7.3 Optimal control and optimal feedback

We will consider two types of feedback: state feedback and output feedback, see figure 7.5 and figure 7.6.

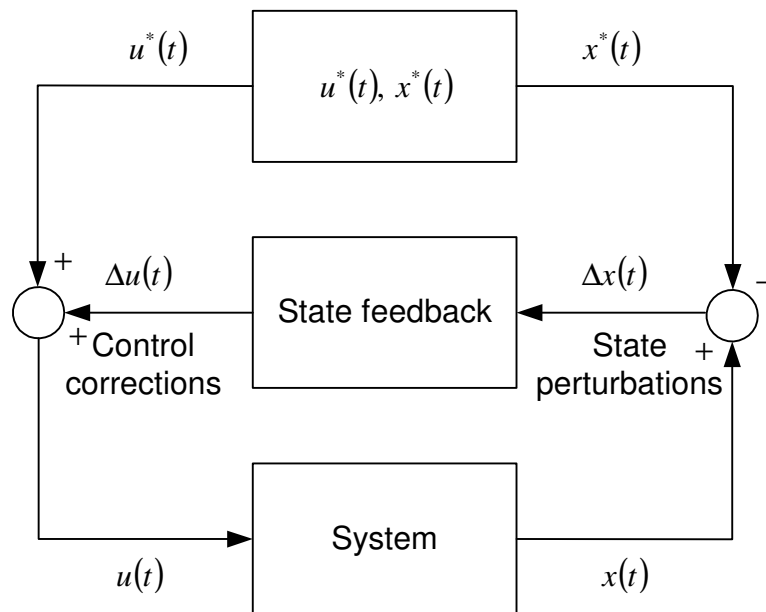


Figure 7.5: State feedback

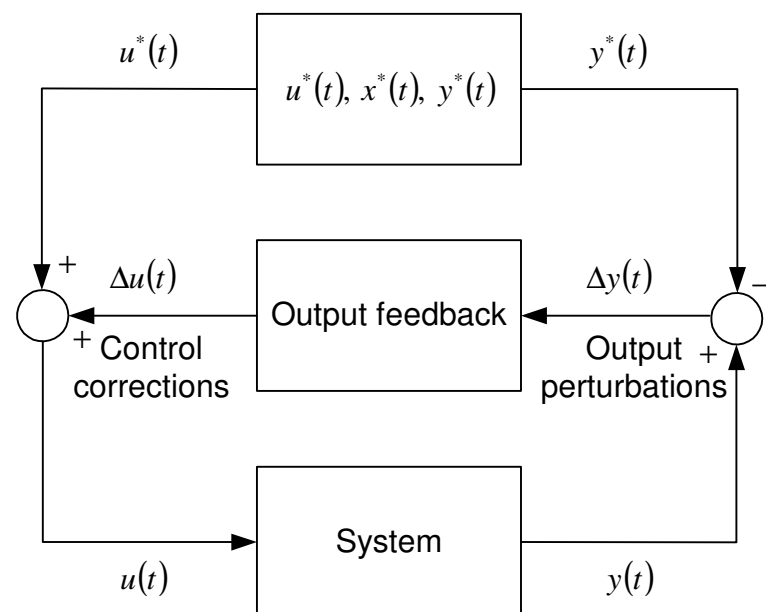


Figure 7.6: Output feedback

In the case of state feedback the problem is how to compute control corrections $\Delta u(t)$ from the measured state deviations (perturbations) $\Delta x(t)$. State feedback requires us to measure all the state variables. In practice this is very often impossible or too expensive. Output feedback allows us to use any available measurements. On the other hand, in general, output feedback is less successful in reducing the deviations (perturbations) than state feedback. In this lecture we will consider both optimal state and optimal output feedback. This means that we will compute the control corrections from the measured perturbations using again the (linearised) dynamic systems model together with a different (quadratic) cost function.

7.4 Summary and implications

- 1) Accurate quantitative system knowledge (the model and the initial state) is vital to accurately control a system.
- 2) Feedback is needed if the system knowledge is imperfect to limit deviations from the desired behaviour. Feedback implies that we have to perform on-line measurements.
- 3) Optimal feedback relies on the system knowledge that we do have (the quantitative model and the initial state). The better this knowledge, the better the feedback and the smaller the deviations from the desired behaviour.
- 4) A controller design may be separated into two parts. One part where we calculate the control assuming our system knowledge (the quantitative model and the initial state) is perfect. The calculation of the control can be done before hand (off-line) and is a mathematical, computational problem. The second part deals with the determination of how exactly the feedback should take place. Optimal feedback relies on the system knowledge that we do have (the quantitative model, the initial state and also the desired system behaviour). The optimal feedback design (how the feedback should take place) can also be done before hand (off-line) and is also a mathematical, computational problem. The feedback itself must be performed during the actual control (on-line) because it uses measurements and these only become available during the actual control.

- 5) Since both the computation of the optimal control and the optimal feedback rely on the system knowledge (the quantitative model and the initial state), which in general is not perfect, the controller design described under 4) may 'fall apart' if the system knowledge (the quantitative model and the initial state) is too poor. On the other hand it is clear that without proper system knowledge we will never be able to come up with a controller of high quality (see also 1)
- 6) Based on points 1-5 in this course we will adopt the following approach to controller design. Our design will be along the lines described under point 4 and we will compute both the control and the feedback in an optimal manner. Optimal means that we will use all available system knowledge (the quantitative model and the initial state) on the one hand and on the other specify exactly, via quantitative cost functions, our control objectives. This is approximately the best we can do, if our system knowledge is of sufficient quality. Furthermore the optimal controller that we obtain needs to perform only very few on-line computations namely those related to the feedback. Figure 7.7 illustrates our approach in the case of output feedback.

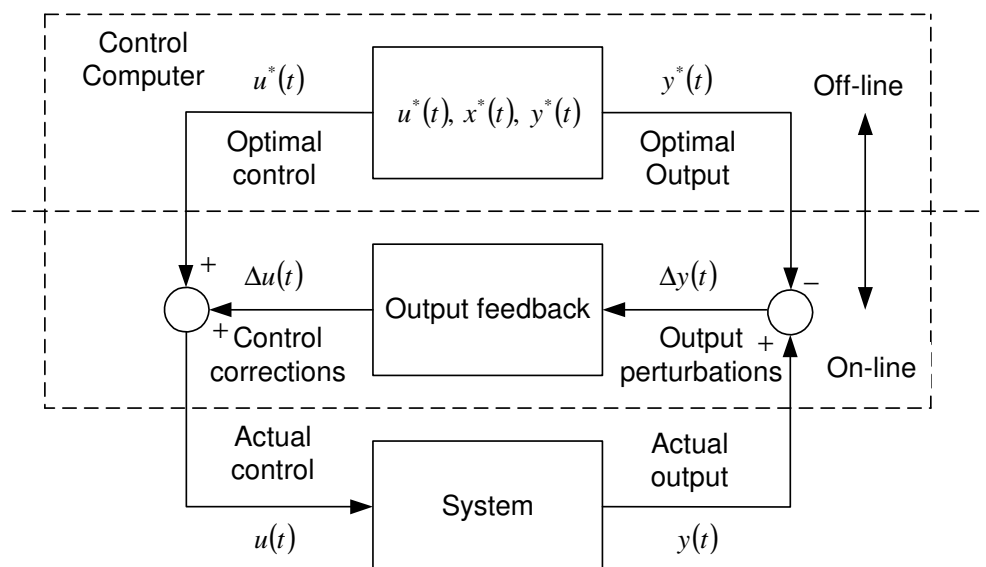


Figure 7.7: Optimal control system based on output-feedback.

- 7) Compared to other control methodologies optimal control and optimal feedback has the following advantages:
- a) All quantitative (scientific) system knowledge can be exploited through the use of a quantitative systems model.
 - b) The use of a cost function allows and forces the designer to specify exactly and quantitatively the control objectives.
 - c) The control objectives will be optimally met.
 - d) Most of the control computations are performed off-line. Only very few on-line control computations are required.
 - e) The approach works for general non-linear systems and arbitrary measurements.
 - f) If the quantitative systems model and the knowledge about the initial state are of sufficient quality optimal control and optimal feedback is (approximately) the best we can do!

8. Formulation and classification of optimal control problems

8.1 Formulation of optimal control problems: an example

After reading this chapter the reader should be able to:

- 1) Formulate simple optimal control problems in mathematical terms.
- 2) Use standard notation to describe optimal control problems and their solution.

Consider a very simple example: the temperature control in a room. To keep the example simple assume that the inside and outside temperature are homogeneous. The temperature in the room can be controlled through a heater the power of which we can adjust. With these assumptions we obtain the following very simple dynamic model,

$$\frac{dT_r}{dt} = c_1(T_o - T_r) + c_2 P_h \quad (8.1)$$

where T_r, T_o are the room and outside temperature respectively P_h is the power of the heater and c_1, c_2 are two known positive constants. The first step in obtaining an optimal control problem formulation is to write the dynamic model of the system (the differential equations and possibly the associated algebraic equations) in state-space form. To do this we first have to determine state variables, next parameters and finally inputs. In (8.1) T_r is the only state variable (state variables are variables of which time derivatives appear in (8.1)). Also all time derivatives of T_r are state variables up to but not including the highest time derivative that appears in (8.1)). The positive constants c_1, c_2 are parameters (parameters are all the constants that appear in (8.1)). Finally the inputs are the variables that are left over i.e. P_h and T_o . P_h is a control input (control variable) since it can be manipulated by us at each time. T_o is an external input (external variable) since it is determined externally. In a state-space model state variables are collected in the column vector $x \in R^{n_x}$, control inputs in the column vector $u \in R^{n_u}$, external inputs in the column vector $d \in R^{n_d}$ and parameters in the column vector $p \in R^{n_p}$. Therefore,

$$x_1 = T_r, u_1 = P_h, d_1 = T_o, p_1 = c_1, p_2 = c_2 \quad (8.2)$$

and we obtain the following state-space model of the system,

$$\frac{dx_1}{dt} = p_1 (d_1 - x_1) + p_2 u_1 \quad (8.3)$$

In the previous section we talked about the desired system behaviour. The system behaviour is $x(t)$, $t_0 \leq t \leq t_f$ i.e. the time response of the state variables where t_0, t_f are respectively the initial and final time. In our example: $x(t) = x_1(t)$, $t_0 \leq t \leq t_f$. The desired system behaviour could be to keep the temperature at a known constant reference value x_1^{ref} . Suppose that the initial state of the system, i.e. the initial temperature $x_1(t_0) \neq x_1^{ref}$. Then at the initial time t_0 the system behaviour cannot be equal to the desired system behaviour (the attentive reader will observe that this contradicts the thought experiment in the previous section, but as with most thought experiments it contained a simplification as to this matter). Also the power that can be supplied by the heater is limited i.e. $u_1 \leq u_1^{\max}$ where $u_1^{\max} > 0$ is the known positive maximum power that can be supplied by the heater. This implies that also shortly after the initial time t_0 the temperature x_1 in the room, in general, cannot be made equal to the desired value x_1^{ref} . Furthermore since cooling is not possible $u_1 \geq 0$. From all this we now obtain the following problem formulation.

Optimal control problem formulation 1

Given the system (8.3) and its initial state $x_1(t_0)$ calculate the control $0 \leq u_1(t) \leq u_1^{\max}$, $t_0 \leq t \leq t_f$ that keeps the temperature $x_1(t)$, $t_0 \leq t \leq t_f$ closest to the desired value x_1^{ref} •

Everything in the optimal control problem formulation 1 is described in mathematical terms except for the last part: closest to the desired value x_1^{ref} . This can be represented in mathematical terms as follows. Minimise the following integral,

$$\int_{t_0}^{t_f} |x_1(t) - x_1^{ref}| dt \quad (8.4)$$

By minimising the integral (8.4) one minimises the average difference $|x_1(t) - x_1^{ref}|$ over the time interval $t_0 \leq t \leq t_f$, see figure 8.1. From this figure observe that,

$$\int_{t_0}^{t_f} |x_1(t) - x_1^{ref}| dt = \text{Area A} + \text{Area B} \quad (8.5)$$

while

$$\int_{t_0}^{t_f} (x_1(t) - x_1^{ref}) dt = \text{Area A} - \text{Area B} \quad (8.6)$$

So as a cost function the integral (8.6) is not suitable.

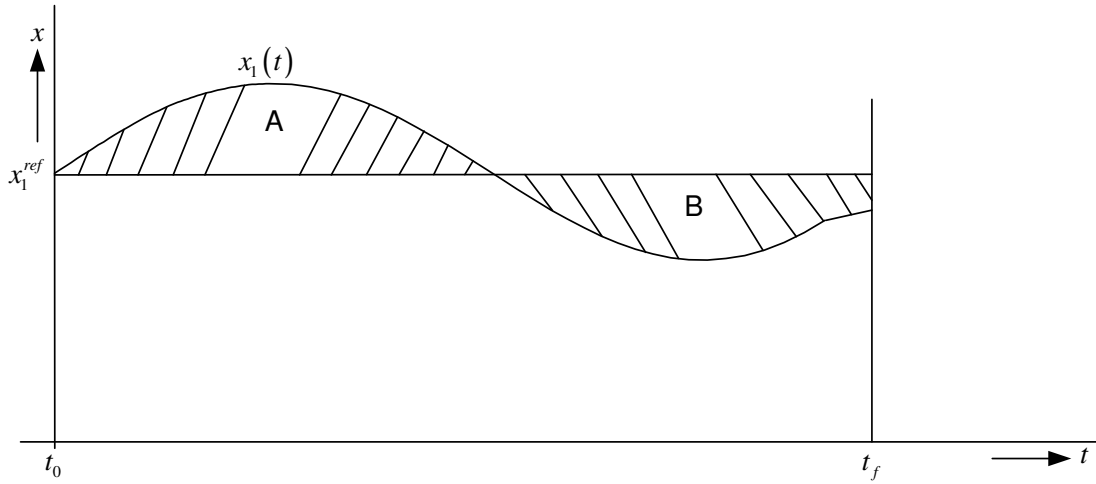


Figure 8.1: Interpretation of the integrals $\int_{t_0}^{t_f} |x_1(t) - x_1^{ref}| dt$ and $\int_{t_0}^{t_f} (x_1(t) - x_1^{ref}) dt$

Optimal control problem formulation 2

Given the system (8.3) and its initial state $x_1(t_0)$ calculate the control

$$0 \leq u_1(t) \leq u_1^{\max}, \quad t_0 \leq t \leq t_f \quad \text{that minimises} \quad \int_{t_0}^{t_f} |x_1(t) - x_1^{ref}| dt \quad \bullet$$

This optimal control problem formulation is stated completely in mathematical terms. It is a so called constraint minimization problem because we have to minimise the integral (8.4) while at the same time satisfying the system equation (8.3) with the known initial condition $x_1(t_0)$. So the constraints are the initial condition and the system equation. The integral (8.4) is called a cost function. There are several alternative names for the cost function such as cost criterion, penalty function, performance measure, performance index.

An alternative for the cost function (8.4) is the following integral,

$$\int_{t_0}^{t_f} (x_1(t) - x_1^{ref})^2 dt \quad (8.7)$$

This integral, instead of the average difference, calculates the average squared difference. There are two reasons why we often prefer to minimise (8.7) instead of (8.4). The first reason is that we are often more concerned with large errors than with small ones. Now because (8.7) is quadratic while (8.4) is proportional large errors will be better prevented by minimizing (8.7). Secondly quadratic functions and integrals of quadratic functions are smooth and very often have a single global minimum which makes the numerical computation of this minimum relatively fast and easy. Observe that (8.4) is an integral of a non-smooth function (why is this function non-smooth?).

Optimal control problem formulation 3

Given the system (8.3) and its initial state $x_1(t_0)$ calculate the control

$$0 \leq u_1(t) \leq u_1^{\max}, t_0 \leq t \leq t_f \text{ that minimises } \int_{t_0}^{t_f} (x_1(t) - x_1^{ref})^2 dt \quad \bullet$$

The calculated control that minimises the cost function $\int_{t_0}^{t_f} (x_1(t) - x_1^{ref})^2 dt$ is, for obvious reasons, called the optimal control and is denoted by

$$u_1^*(t), t_0 \leq t \leq t_f \quad (8.8)$$

where * indicates optimality. Associated with the optimal control (8.8) is the so called optimal state trajectory or optimal system behaviour,

$$x_1^*(t), t_0 \leq t \leq t_f \quad (8.9)$$

Note that, in general, $x_1^*(t) \neq x_1^{ref}$, $t_0 \leq t \leq t_f$ (Why? When does the equality hold?). Finally the minimal value of the cost function (minimal costs), which is a function of the optimal control, is denoted by,

$$J^*(u^*(t)) = \int_{t_0}^{t_f} (x_1^*(t) - x_1^{ref})^2 dt \quad (8.10)$$

Summary

- 1) Optimal control problems are described in mathematical terms.
- 2) Optimal control problems are formulated in terms of the state, control and external variables of the system. The system description therefore has to be in state-space form.
- 3) The 'desired system behaviour' is formulated in terms of a cost function that is minimised. Often the cost function is an integral which runs from the initial time t_0 , where the control starts, to the final time t_f , where the control ends.
- 4) The solution of an optimal control problem consists of the optimal control $u^*(t)$, $t_0 \leq t \leq t_f$, the associated optimal state trajectory (the associated optimal system behaviour) $x^*(t)$, $t_0 \leq t \leq t_f$ and associated minimum costs $J^*(u^*(t))$.

8.2. Formulation and classification of optimal control problems

After reading this chapter the reader should be able to:

- 1) Recognise the notation and general formulation of optimal control problems
- 2) Recognise different types of optimal control problems: with or without a) running costs b) terminal costs c) control constraints d) state constraints e) terminal constraints f) free final time
- 3) Formulate specific optimal control problems in terms of the general formulation

Consider again the system (8.3). The goal is now to bring the temperature as closely as possible towards x_1^{ref} within 1 minute. Opposite to the example in section 2 we do not care about what happens during this minute. Because the system is time-invariant (time does not explicitly appear in the differential equations) we select,

$$t_0 = 0 \text{ sec.}, t_f = 60 \text{ sec.} \quad (8.11)$$

Because we only care about what happens with the temperature x_1 at the final time $t_f = 60 \text{ sec.}$ our cost function is not an integral in this case. Two choices of the cost function are,

$$J(u(t)) = |x_1(t_f) - x_1^{ref}(t_f)| \quad (8.12)$$

$$J(u(t)) = (x_1(t_f) - x_1^{ref}(t_f))^2 \quad (8.13)$$

If we do not care about what happens during the time-interval $[t_0, t_f]$ over which we control the system, but are only interested in the state of the system at the final time t_f then the cost function is a function of only the final state,

$$J(u(t)) = \phi(x(t_f)) \quad (8.14)$$

For obvious reasons $\phi(x(t_f))$ in equation (8.14) are called the terminal costs. As we have seen in the previous section, if we do care about the system behaviour $x(t)$, $t_0 \leq t \leq t_f$ during the time-interval $[t_0, t_f]$ over which we control the system, the cost function is an integral, the integrand of which is a function of the state of the system,

$$J(u(t)) = \int_{t_0}^{t_f} L(x) dt \quad (8.15)$$

$\int_{t_0}^{t_f} L(x) dt$ in equation (8.15) are called the running costs. It may happen that we care both about what happens with the state during the time-interval $[t_0, t_f]$ and at the final time t_f . In this case the cost function contains both running and terminal costs respectively,

$$J(u(t)) = \phi(x(t_f)) + \int_{t_0}^{t_f} L(x) dt \quad (8.16)$$

So far our cost function (8.16) contains terminal costs and running costs which are only explicit functions of the state (only the state does appear explicitly in the function description). Because, through the system equation (8.3), the state depends on the control our costs depend implicitly on the control. Very often however, we want our running costs to depend explicitly on the control, for instance if we do not want to use too much power $u(t)$ for heating the room while at the same time we want the temperature $x_1(t)$ to stay close to x_1^{ref} . In that case our cost function may be chosen as follows,

$$J(u(t)) = \int_{t_0}^{t_f} \alpha_1 (x_1(t) - x_1^{ref})^2 + \alpha_2 u_1(t) dt \quad (8.17)$$

In equation (8.17) α_1, α_2 are positive constants which we select. By α_1 we indicate the importance of keeping the temperature $x_1(t)$ close to x_1^{ref} and by α_2 we indicate the importance of keeping the heating power $u_1(t)$ small. Clearly the solution of the optimal control problem is a compromise between these two. This compromise is influenced by the selection of α_1, α_2 which, for obvious reasons, are called weighting factors. Finally we remark that we may also want our running costs and terminal costs to depend explicitly on the external variables. Then we obtain the following general form of the cost function,

$$J(u(t)) = \phi(x(t_f), d(t_f)) + \int_{t_0}^{t_f} L(x, u, d) dt \quad (8.18)$$

Together with the state-space representation of systems, which is a general form to represent systems, this results in a general formulation of optimal control problems. Because, without complicating the optimal control analysis, time-varying systems may also be considered, in this general formulation time appears explicitly in the system equation, the cost function and the constraints.

General optimal control problem formulation

Given the system,

$$\dot{x} = f(x, u, d, t), \quad x \in R^{n_x}, \quad u \in R^{n_u}, \quad d \in R^{n_d} \quad (8.19)$$

with the known initial condition

$$x(t_0) = x_0 \quad (8.20)$$

and given the known time evolution of the external variables,

$$d(t), \quad t_0 \leq t \leq t_f \quad (8.21)$$

find the optimal control $u^*(t), t_0 \leq t \leq t_f$ that minimizes the cost function

$$J(u(t)) = \phi(x(t_f), d(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, d, t) dt \quad (8.22)$$

and find the minimum cost $J(u^*(t))$. •

In the optimal control problem formulations 1-3 in the previous section,

$$0 \leq u_1(t) \leq u_1^{\max}, t_0 \leq t \leq t_f \quad (8.23)$$

had to be satisfied. If, as in these problem formulations, some of the control variables (or a combination of them) may only belong to a restricted set, these restrictions are also part of the control problem formulation. In this case the optimal control problem is said to contain control constraints. Equation (8.23) is an example of a control inequality constraint. Likewise there exist optimal control problems with state constraints. As an example consider the temperature control in the room where the temperature has to stay within a certain range,

$$x_1^{\min} \leq x_1(t) \leq x_1^{\max}, t_0 \leq t \leq t_f \quad (8.24)$$

Equation (8.24) represents a state inequality constraint. Finally there also exist optimal control problems with so called terminal constraints. In this case the final state of the system $x(t_f)$ has to satisfy exactly one or more constraints. These constraints are described by,

$$\psi(x(t_f), d(t_f), t_f) = 0, \psi \in R^{n_\psi}, n_\psi \leq n_x \quad (8.25)$$

where ψ is a vector function in which the terminal constraints are collected. Equation (8.25) represents terminal state equality constraints. As an example consider again the temperature control in a room. Suppose that we want the temperature at the final time $x_1(t_f)$ to be exactly equal to x_1^{ref} . Then,

$$\psi(x(t_f)) = x_1(t_f) - x_1^{ref} \quad (8.26)$$

Note the subtle difference between the use of $\phi(x(t_f))$, which is part of the cost function (8.22), and the use of terminal constraints $\psi(x(t_f))$. Again consider the temperature control in a room. If we want to bring the temperature as closely as possible towards the desired value x_1^{ref} at the final time, taking into account that it may not be possible to exactly reach this temperature, we specify,

$$\phi(x(t_f)) = (x_1(t_f) - x_1^{ref})^2 \quad (8.27)$$

If we want to bring the temperature at exactly the desired value x_1^{ref} we specify,

$$\psi(x(t_f)) = x_1(t_f) - x_1^{ref} \quad (8.28)$$

An interesting question is now under which circumstances these two optimal control problems have the same solution. The answer is left to the reader.

Usually the initial time t_0 and final time t_f of an optimal control problem are fixed. However, consider the problem of controlling the temperature in a room as quickly as possible from the initial temperature $x_1(t_0)$ to the desired value x_1^{ref} . In this case the time t_f is not fixed but has to be found (optimised) as well. This is an optimal control problem with a so called free final time. In the case of optimal control problems with a free final time the solution consists of the optimal control, the associated optimal system behaviour, the minimum costs and also the optimal (minimal in our example) free final time. The formulation, in general terms, of the problem of controlling the temperature in a room as quickly as possible from the initial temperature $x_1(t_0)$ to the desired value x_1^{ref} is also left as an exercise to the reader.

8.3 Examples of how to translate verbal control objectives and constraints into mathematical cost functions and constraints.

The general notation of a to be minimized cost function (performance measure) is,

$$J(u(t)) = \underbrace{\phi(x(t_f), t_f)}_{\text{Terminal cost}} + \underbrace{\int_{t_0}^{t_f} L(x, u, t) dt}_{\text{Running cost}}, \quad J, \phi, L \in \mathbb{R}^1 \text{ (scalars)}. \quad (8.29)$$

By reversing the sign of the terminal cost $\phi(x(t_f), t_f)$ and running cost $L(x, u, t)$ minimization turns into maximization of the cost function (performance measure).

The general notation of terminal constraints is,

$$\psi(x(t_f), t_f) = 0, \quad \psi \in \mathbb{R}^{n_\psi} \text{ (column vector)}. \quad (8.30)$$

Examples:

a) $x_1(t_f)$ close to 2 : $\phi(x(t_f), t_f) = (x_1(t_f) - 2)^2$: terminal costs

b) $x_1(t)$ close to $\sin(t)$, $t_0 \leq t \leq t_f$: $L(x, u) = (x_1(t) - \sin(t))^2$: running costs

c) Both 1) and 2): $J = \underbrace{\alpha_1}_{\text{weighing factor}} \underbrace{(x_1(t_f) - 2)^2}_{\phi(x(t_f), t_f)} + \int_{t_0}^{t_f} \underbrace{(x_1(t) - \sin(t))^2}_{L(x, u, t)} dt$

d) $x_1(t_f)$ exactly 2 : $\psi(x(t_f), t_f) = x_1(t_f) - 2$; terminal state equality constraint

e) $x_1^{\min} \leq x_1(t) \leq x_1^{\max}$, $t_0 \leq t \leq t_f$: lower and upper bound $x_1(t)$; state inequality constraint

f) $u_1^{\min} \leq u_1(t) \leq u_1^{\max}$, $t_0 \leq t \leq t_f$: lower and upper bound $u_1(t)$; control inequality constraint

g) Suppose $u_1(t) = P(t) = \text{power}$ $[W = Js^{-1}]$. Power when integrated equals energy. Minimum energy consumption:

$$J = \int_{t_0}^{t_f} P(t) dt = \int_{t_0}^{t_f} \underbrace{u_1(t)}_{L(x, u)} dt$$

h) Terminal quadratic cost represented using a diagonal positive square matrix:

$$h_{11}x_1^2(t_f) + h_{22}x_2^2(t_f) = x^T(t_f)Hx(t_f), \quad H = \begin{bmatrix} h_{11} & 0 \\ 0 & h_{22} \end{bmatrix}, \quad h_{11}, h_{22} > 0, \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

8.4 Summary

The general form of an optimal control problem reads as follows. Given the system,

$$\dot{x} = f(x, u, d, t), \quad x \in R^{n_x}, \quad u \in R^{n_u}, \quad d \in R^{n_d} \quad (8.31)$$

with the known initial condition

$$x(t_0) = x_0 \quad (8.32)$$

and given the known time evolution of the external variables,

$$d(t), \quad t_0 \leq t \leq t_f \quad (8.33)$$

find the optimal control $u^*(t)$, $t_0 \leq t \leq t_f$ that minimizes the cost function

$$J(u(t)) = \phi(x(t_f), d(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, d, t) dt \quad (8.34)$$

and find the minimum cost $J(u^*(t))$, where possibly the following terminal state constraints have to be satisfied,

$$\psi(x(t_f), d(t_f), t_f) = 0, \quad \psi \in R^{n_\psi}, \quad n_\psi \leq n_x \quad (8.35)$$

In addition there may be control constraints and state constraints.

The solution of an optimal control problem consists of the optimal control $u^*(t)$, $t_0 \leq t \leq t_f$, the associated optimal state trajectory (optimal system behaviour) $x^*(t)$, $t_0 \leq t \leq t_f$ and the associated minimum costs $J(u^*(t))$. If the optimal control problem has a free final time then also the optimal value of t_f is part of the solution.

Optimal control problems can be classified as follows:

- 1) With fixed or free final time.
- 2) With or without control (equality/inequality) constraints.
- 3) With or without state (equality/inequality) constraints.
- 4) With or without terminal state (equality/inequality) constraints.

As we see this function has several (in fact infinitely many) local minima and local maximums. Furthermore these local minima and maximums are not unique, they come in pairs. These properties, which are rather special, may also be properties of optimal control problems.

We are interested in how to calculate local and global minima. Throughout this section we will only consider smooth functions meaning functions all the derivatives of which are continuous functions. A necessary condition to be at a local minimum is that the first derivative of the function equals zero. The same however holds for a local maximum. Local minima and maximums can be distinguished by considering the second derivative. In the case of a local minimum this second derivative is positive, since from decreasing the function starts to increase, while at a local maximum it is negative because from increasing the function starts to decrease. Summarising,

$$\left. \frac{df}{dx} \right|_{x=a} = 0, \left. \frac{d^2 f}{dx^2} \right|_{x=a} > 0 \Leftrightarrow x = a \text{ is a local minimum of } f(x) \quad (9.2)$$

$$\left. \frac{df}{dx} \right|_{x=a} = 0, \left. \frac{d^2 f}{dx^2} \right|_{x=a} < 0 \Leftrightarrow x = a \text{ is a local maximum of } f(x) \quad (9.3)$$

Equations (9.2) and (9.3) are called necessary and sufficient conditions for respectively a local minimum and maximum. Now that we can distinguish between local minima and local maximums the next important question is whether we can distinguish between local and global minima. Unfortunately, in general, no conditions are known which distinguish global from local minima. In general, the only thing that we can do is compare the function values of local minima and from them pick the one with the smallest function value. Therefore in general, only if we know the number of local minima and are able to find them all, we can guarantee to have found the global minimum. So the problem of finding the global minimum comes down to finding local minima. So this is what we will be concerned with.

Well known methods to find numerically, on a computer, a local minimum of a function, are so called gradient methods. These methods only compute $f(x)$ and

$\frac{df}{dx}$ for several values of x to find a local minimum. Although gradient methods do not compute second derivatives the way the algorithms search for a solution

guarantees that it does not find a local maximum. In other words computing only function values and first derivatives is sufficient to find a local minimum.

Next consider the problem of finding local minima of constrained function minimisation problems. An example is the determination of the minimum distance between a parabola and a line. The situation is shown in figure 9.2.

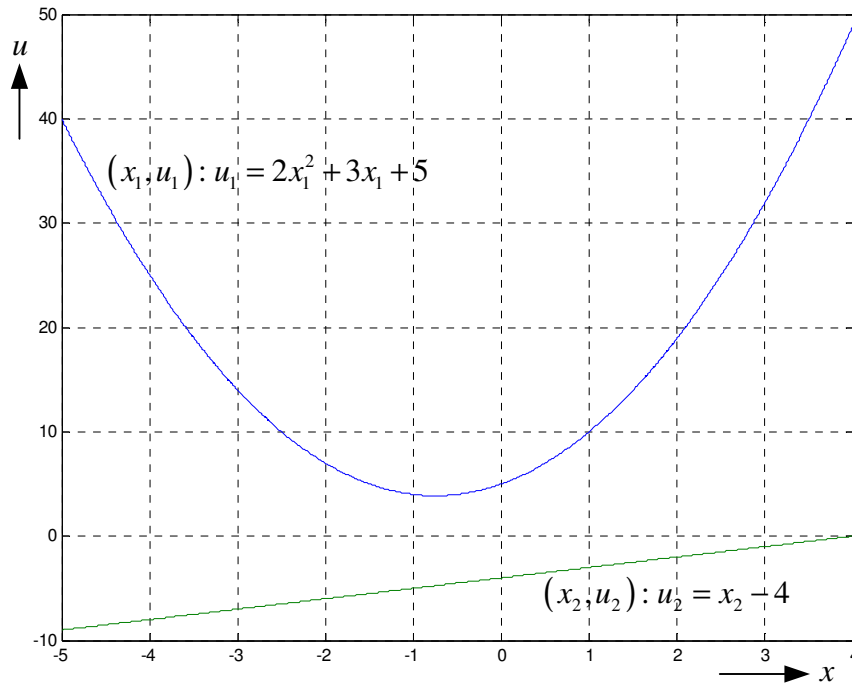


Figure 9.2: Minimum distance between the parabola and the line

Although this is not an optimal control problem yet, we use mathematical symbols that also appear in the general optimal control problem formulation. The reason is that, later on, we can more easily observe similarities. The parabola is described by the following equation,

$$u_1 = 2x_1^2 + 3x_1 + 5 \text{ or } u_1 - 2x_1^2 - 3x_1 - 5 = 0 \quad (9.4)$$

where (u_1, x_1) is a point on the parabola. The line is described by,

$$u_2 = x_2 - 4 \text{ or } u_2 - x_2 + 4 = 0 \quad (9.5)$$

where (u_2, x_2) is a point on the line. The distance D between a point (u_1, x_1) on the parabola and a point (u_2, x_2) on the line is given by,

$$D(x_1, x_2, u_1, u_2) = \sqrt{(x_1 - x_2)^2 + (u_1 - u_2)^2} \quad (9.6)$$

Finding the minimum of D is identical to finding the minimum of,

$$J(x_1, x_2, u_1, u_2) = \frac{1}{2}(x_1 - x_2)^2 + \frac{1}{2}(u_1 - u_2)^2 \quad (9.7)$$

which is more easy computationally. So the problem of finding the minimum distance between the parabola and the line is the same as finding the points (u_1, x_1) and (u_2, x_2) , on respectively the parabola and the line, which minimise the function (9.7).

This is a constrained function minimisation problem because equations (9.4) and (9.5) have to be satisfied.

To solve constrained minimisation problems we will now perform a kind of mathematical 'trick'. This 'trick' is obtained from Lagrange theory. After performing this 'trick' we can solve the constrained function minimisation problem as if it were a function minimisation problem without constraints, which we know how to solve. In mathematical terms: we will transform the constrained function minimisation problem into an unconstrained function minimisation problem which has the same solution. The unconstrained function minimisation problem associated to the above example reads as follows. Minimise,

$$J'(x_1, x_2, u_1, u_2, \lambda_1, \lambda_2) = J(x_1, x_2, u_1, u_2) + \lambda_1(u_1 - 2x_1^2 - 3x_1 - 5) + \lambda_2(u_2 - x_2 + 4) \quad (9.8)$$

In the new function J' that we are going to minimise, two new variables λ_1, λ_2 appear. These are multiplied by respectively the two constraints that have to be satisfied. Now let us minimise J' as if there are no constraints. Because J' is a function of several variables at a local minimum of the function J' ,

$$\frac{\partial J'}{\partial x_1} = \frac{\partial J'}{\partial x_2} = \frac{\partial J'}{\partial u_1} = \frac{\partial J'}{\partial u_2} = \frac{\partial J'}{\partial \lambda_1} = \frac{\partial J'}{\partial \lambda_2} = 0 \quad (9.9)$$

must be satisfied. Note that (9.9) are six equations with six unknowns. Furthermore,

$$\begin{aligned} \frac{\partial J'}{\partial \lambda_1} = 0 &\Leftrightarrow u_1 - 2x_1^2 - 3x_1 - 5 = 0 \\ \frac{\partial J'}{\partial \lambda_2} = 0 &\Leftrightarrow u_2 - x_2 + 4 = 0 \end{aligned} \quad (9.10)$$

so at a local minimum of J' the original constraints (9.4), (9.5) are satisfied! If these constraints are satisfied observe from equation (9.8) that

$$J' = J \quad (9.11)$$

holds at a local minimum of J' . From (9.11) and (9.9),

$$\frac{\partial J}{\partial x_1} = \frac{\partial J}{\partial x_2} = \frac{\partial J}{\partial u_1} = \frac{\partial J}{\partial u_2} = 0 \quad (9.12)$$

also holds at a local minimum of J' . From (9.12), (9.11) and (9.10) observe that a local minimum of the unconstrained function, J' is also a minimum of the function J with the constraints f !

To summarise and generalise our result the following vector notation is introduced,

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}, f = \begin{bmatrix} u_1 - 2x_1^2 - 3x_1 - 5 \\ u_2 - x_2 + 4 \end{bmatrix} \quad (9.13)$$

Note that f is now a vector containing the constraints. Applying the ordinary rules for vector and matrix multiplication observe from (9.6) and (9.8) that,

$$J' = J + \lambda^T f \quad (9.14)$$

where λ^T denotes the transpose of λ .

Summary

1. Gradient methods can be used to compute numerically, on a computer, local minima of functions of several variables. They only compute function values and first derivatives.
2. Global minima cannot be distinguished from local minima. To try to find global minima one can only compute local minima and among them search for the one with the smallest function value.
3. Constrained function minimisation problems can be solved by transforming them into unconstrained function minimisation problems and solving these using gradient methods. This transformation has to be performed as follows. The original function J that has to be minimised, must be extended with as many variables as there are constraints, into the function J' . The function J'

is equal to the function J plus the additional variables collected in the vector λ , each one multiplied with one of the constraints, collected in the vector f ,

$$J' = J + \lambda^T f \quad (9.15)$$

According to Lagrange theory the minima of the unconstrained function J' are identical to minima of the function J with constraints f . The 'price' that we have to pay for transforming a constrained into an unconstrained function minimisation problem is that we have to solve λ , which did not appear in the original constrained problem. The elements of the vector λ (sometimes the vector itself) are called Lagrange multipliers.

9.2 *First derivatives of functions of vectors and vector functions.*

After reading this section the reader should be able to:

1. Differentiate vector functions
2. Apply rules of differentiation to vectors and vector functions

In the last part of the previous section, which considered function minimisation, it turned out to be very convenient to use vector notations. The same applies to optimal control problems. Furthermore from the previous section we learned that to compute local minima, it is sufficient to compute function values and first derivatives only. Therefore in this section we will introduce derivatives of functions of vectors and vector functions. These derivatives are more or less straightforward generalisations of the scalar derivatives that we all know.

Consider again the function,

$$J(x_1, x_2, u_1, u_2) = \frac{1}{2}(x_1 - x_2)^2 + \frac{1}{2}(u_1 - u_2)^2 \quad (9.16)$$

and the vectors

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad (9.17)$$

used in the previous section. Using (9.17) we may rewrite (9.16) as,

$$J(x, u), J \in R^1, u \in R^2, x \in R^2 \quad (9.18)$$

Define,

$$\frac{\partial J}{\partial x} = \begin{pmatrix} \frac{\partial J}{\partial x_1} & \frac{\partial J}{\partial x_2} \end{pmatrix} \in R^{1 \times 2} \quad (9.19)$$

and similarly

$$\frac{\partial J}{\partial u} = \begin{pmatrix} \frac{\partial J}{\partial u_1} & \frac{\partial J}{\partial u_2} \end{pmatrix} \in R^{1 \times 2} \quad (9.20)$$

so the partial derivative of the scalar function $J \in R^1$ of the two column vectors $u, x \in R^{2 \times 1}$ with respect to one of these column vectors is a row vector with a length equal to the length of the column vector. In general,

$$J(x), J \in R^1, x \in R^{n_x} \Rightarrow \frac{dJ}{dx} = \begin{pmatrix} \frac{\partial J}{\partial x_1} & \frac{\partial J}{\partial x_2} & \dots & \frac{\partial J}{\partial x_{n_x}} \end{pmatrix} \in R^{1 \times n_x} \quad (9.21)$$

The first derivative of a function of a vector is called a gradient which is a row vector.

The result of the previous section concerning J , J' and f can now be summarised very shortly. A local minimum of the function,

$$J'(x, u) = J(x, u) + \lambda^T f(x, u) \quad (9.22)$$

satisfies,

$$\frac{\partial J'}{\partial x} = 0 \in R^{1 \times 2}, \frac{\partial J'}{\partial u} = 0 \in R^{1 \times 2}, \frac{\partial J'}{\partial \lambda} = f^T = 0 \in R^{1 \times 2} \quad (9.23)$$

Equation (9.23) are three vector equations in three unknown vectors $x, u, \lambda \in R^2$.

They are identical to the six scalar equations (9.9) in section 9.1. From (9.22), (9.23) and (9.21) it follows that,

$$\frac{\partial J'}{\partial x} = \frac{\partial J}{\partial x} = 0 \in R^{1 \times 2}, \frac{\partial J'}{\partial u} = \frac{\partial J}{\partial u} = 0 \in R^{1 \times 2} \quad (9.24)$$

holds for a local minimum of J' . Therefore this local minimum is also a local minimum of the function J with constraints f .

A vector function $f(x, u)$ is a function the outcome of which is a vector,

$$f(x, u) = \begin{bmatrix} f_1(x, u) \\ f_2(x, u) \\ \vdots \\ f_{n_f}(x, u) \end{bmatrix}, \quad f \in R^{n_f}, \quad x \in R^{n_x}, \quad u \in R^{n_u} \quad (9.25)$$

The n_f elements of the vector function (9.25) are the scalar functions (the outcome of which is a scalar) $f_1(x, u), \dots, f_{n_f}(x, u)$. Define,

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x} \\ \frac{\partial f_2}{\partial x} \\ \vdots \\ \frac{\partial f_{n_f}}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_{n_x}} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_{n_x}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_{n_f}}{\partial x_1} & \frac{\partial f_{n_f}}{\partial x_2} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \in R^{n_f \times n_x}, \quad f \in R^{n_f}, \quad x \in R^{n_x} \quad (9.26)$$

In general,

$$f(x), \quad f \in R^{n_f}, \quad x \in R^{n_x} \Rightarrow \frac{df}{dx} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_{n_x}} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_{n_x}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_{n_f}}{\partial x_1} & \frac{\partial f_{n_f}}{\partial x_2} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \in R^{n_f \times n_x} \quad (9.27)$$

Another way to write (9.27) is,

$$f(x), \quad f \in R^{n_f}, \quad x \in R^{n_x} \Rightarrow \left[\frac{df}{dx} \right]_{i,j} = \frac{\partial f_i}{\partial x_j}, \quad i = 1, 2, \dots, n_f, \quad j = 1, 2, \dots, n_x \quad (9.28)$$

where i, j denotes the associated element of the matrix. The first derivative of a vector function is called the Jacobian which is a matrix.

Having introduced the gradient and the Jacobian we can now establish rules of differentiation for vector functions (note that a scalar function is simply a special case of a vector function). These rules can be verified by working out simple examples where the matrices, vectors and vector functions have just two components. If $t \in R, \quad x, \lambda, f, g \in R^n, \quad A, S \in R^{n \times n}$ then,

$$\text{Product rule 1: } \frac{d}{dx} \left(f^T(x) g(x) \right) = f^T \frac{dg}{dx} + g^T \frac{df}{dx} \quad (9.29)$$

$$\text{Product rule 2: } \frac{d}{dt} \left(S(t) x(t) \right) = \frac{dS}{dt} x + S \frac{dx}{dt} \quad (9.30)$$

$$\frac{\partial}{\partial x} (\lambda^T x) = \frac{\partial}{\partial x} (x^T \lambda) = \lambda^T \quad (9.31)$$

$$\frac{\partial}{\partial x} (\lambda^T A x) = \frac{\partial}{\partial x} (x^T A^T \lambda) = \lambda^T A \quad (9.32)$$

$$\frac{\partial}{\partial x} (\lambda^T f(x)) = \frac{\partial}{\partial x} (f^T(x) \lambda) = \lambda^T \frac{df}{dx} \quad (9.33)$$

$$\frac{\partial}{\partial x} (x^T A x) = x^T (A + A^T) \quad (9.34)$$

Equations (9.31)-(9.34) are all gradients. The following is a Jacobian,

$$\frac{\partial}{\partial x} (A x) = A \quad (9.35)$$

Summary

The derivative of a vector function $f(x) \in R^{n_f}$ of the vector $x \in R^{n_x}$ is defined by,

$$\frac{df}{dx} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_{n_x}} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_{n_x}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_{n_f}}{\partial x_1} & \frac{\partial f_{n_f}}{\partial x_2} & \cdots & \frac{\partial f_{n_f}}{\partial x_{n_x}} \end{bmatrix} \in R^{n_f \times n_x} \quad (9.36)$$

and is called the Jacobian. Alternatively (9.36) can be written as,

$$\left[\frac{df}{dx} \right]_{i,j} = \frac{\partial f_i}{\partial x_j}, \quad i=1,2,\dots,n_f, \quad j=1,2,\dots,n_x \quad (9.37)$$

where i, j points to a matrix element. If we have a scalar function $f(x) \in R^1$, which is a special case of a vector function, then the derivative is called the gradient, which is a row vector. From the definition above rules of differentiation can be derived. The

important ones are (9.29)-(9.35). They can be verified by writing down simple examples where the vectors and vector functions have two components.

9.3 *Solution of the general optimal control problem*

After reading this section the reader should be able to:

1. Reproduce the general formulation of an optimal control problem
2. Reproduce the necessary optimality conditions for this problem
3. Recognise that the necessary optimality conditions constitute a two point boundary value problem.

In the previous section we have learned how to transform constrained function minimisation problems into unconstrained function minimisation problems using Lagrange multipliers. Optimal control problems contain both static and dynamic constraints. Like function minimisation problems optimal control problems can also be transformed into problems without constraints by using Lagrange multipliers. Let us recall the general formulation of an optimal control problem, presented in section 8.2, but without control and state constraints.

Optimal control problem formulation

Given the system,

$$\dot{x} = f(x, u, d, t) \text{ or } f(x, u, d, t) - \dot{x} = 0, \quad x \in R^{n_x}, \quad u \in R^{n_u}, \quad d \in R^{n_d} \quad (9.38)$$

with the known initial condition

$$x(t_0) = x_0 \quad (9.39)$$

and given the known time evolution of the external variables,

$$d(t), \quad t_0 \leq t \leq t_f \quad (9.40)$$

find the optimal control $u^*(t), t_0 \leq t \leq t_f$ that minimizes the cost function

$$J(u(t)) = \phi(x(t_f), d(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, d, t) dt \quad (9.41)$$

and find the minimum costs $J(u^*(t))$, where possibly the following terminal constraints have to be satisfied,

$$\psi(x(t_f), d(t_f), t_f) = 0, \quad \psi \in R^{n_\psi}, \quad n_\psi \leq n_x \quad (9.42)$$

•

The constraints in the optimal control formulation are equations (9.38) and (9.42). Using Lagrange multipliers we can minimise the following new cost function without constraints,

$$J'(u(t)) = \phi(x(t_f), d(t_f), t_f) + v^T \psi(x(t_f), d(t_f), t_f) + \int_{t_0}^{t_f} (L(x, u, d, t) + \lambda^T (f(x, u, d, t) - \dot{x})) dt \quad (9.43)$$

In equation (9.43) we have introduced a column vector $v \in R^{n_\psi}$ of Lagrange multipliers associated to the terminal state constraint (9.42). Furthermore for each time $t_0 \leq t \leq t_f$ we have introduced a vector of Lagrange multiplier $\lambda \in R^{n_x}$ associated to the constraint (9.38), being the state equation which must hold for each time $t_0 \leq t \leq t_f$.

To minimise our new cost function without constraints we have to solve for,

$$u(t), x(t), \lambda(t), 0 \leq t \leq t_f, v, t_f \quad (9.44)$$

as if these are independent. If the final time is fixed we do not have to solve for t_f .

The function (9.43) is not an ordinary function but a functional. A functional is not (only) a function of variables but of functions (as well). The functional (9.43) is a function of the three time functions and two variables mentioned in (9.44). To find a local minimum of the functional (9.43), instead of considering partial first derivatives which all have to be zero, according to the calculus of variations, we have to consider very small changes dJ' in the functional J' caused by very small changes $du(t), dx(t), d\lambda(t), dv, dt_f$. At a local minimum of J' , dJ' is zero for arbitrary $du(t), dx(t), d\lambda(t), dv, dt_f$. Similarly for an ordinary function $J(x)$ at a local minimum $dJ = \frac{dJ}{dx} dx = 0 \bullet dx = 0$ for arbitrary very small changes dx .

According to the calculus of variations we have to distinguish between variations $\delta x(t)$ and differentials $dx(t)$ of time functions $x(t)$. A variation $\delta x(T)$ is a small change in $x(T)$ where the time T is held fixed. A differential $dx(T)$ is a small

change in $x(T)$ when there is also a small time change dT in T . The relation between the two is illustrated by figure 9.3 and described by the following equation,

$$dx(T) = \delta x(T) + \dot{x}(T) dT \quad (9.45)$$

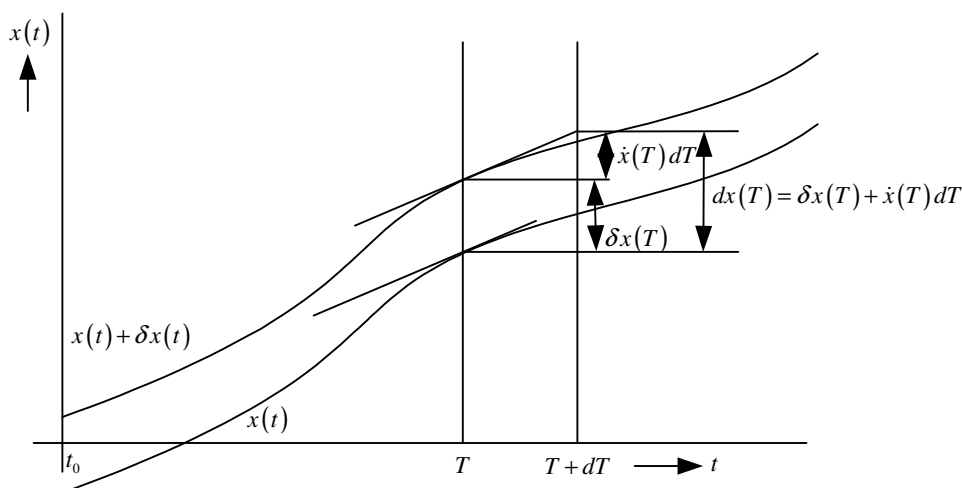


Figure 9.3: Relation between differentials and variations

The first term in (9.45) represents the increment in $x(T)$ caused by the variation $\delta x(T)$ while the second term represents the additional small change in $x(T)$ caused by the time increment dT . Different notations of differentials and variations are being used,

$$dx(T) = dx(t)|_{t=T} = dx(t)|_T = dx|_{t=T} = dx|_T \quad (9.46)$$

$$\delta x(T) = \delta x(t)|_{t=T} = \delta x(t)|_T = \delta x|_{t=T} = \delta x|_T \quad (9.47)$$

The following notations of time increments are used,

$$dt|_{t=T} = dt|_T = dT \quad (9.48)$$

We will use the last and shortest notation in (9.46)-(9.48). In Figure 9.3 the following assumption is made,

$$\left. \frac{d}{dt} \delta x \right|_T = 0 \Rightarrow \dot{x}(T) = \left. \frac{d}{dt} (x + \delta x) \right|_T \quad (9.49)$$

Since $\delta x(t)$ are increments that are infinitely small in principle, this assumption holds. To determine the increments dJ' of the functional (9.43) we need Leibniz's rule which reads as follows. If,

$$J(x(t)) = \int_{t_0}^{t_f} h(x(t), t) dt, \quad J, h \in R^1, \quad x \in R^{n_x} \quad (9.50)$$

then

$$dJ = h(x(t_f), t_f) dt_f + \int_{t_0}^{t_f} \frac{\partial h}{\partial x} \delta x dt \quad (9.51)$$

Figure 9.4 illustrates Leibniz rule. To interpret this figure we have to remember that

$\int_{t_0}^{t_f} h(x(t), t) dt$ is the surface under the function $h(x(t))$ in between t_0 and t_f and

also that $\frac{\partial h}{\partial x} \delta x = \delta h$. The change of the surface under the function $h(t)$ equals,

$$\text{Area B} - \text{Area A} + \text{Area C} \quad (9.52)$$

Furthermore,

$$\text{Area B} - \text{Area A} = \int_{t_0}^{t_f} \delta h(t) dt = \int_{t_0}^{t_f} \frac{\partial h}{\partial x} \delta x(t) dt \quad (9.53)$$

$$\text{Area C} = h(x(t_f), t_f) dt_f \quad (9.54)$$

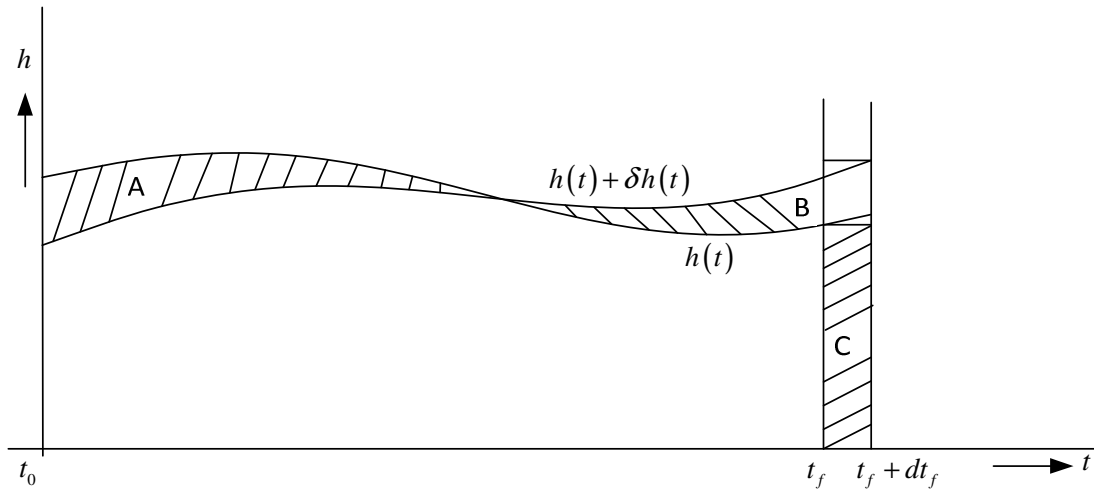


Figure 9.4: Leibniz rule visualised

Consider again the new cost function J' in (9.43). For convenience we introduce the so called Hamiltonian,

$$H(x, u, d, t) = L(x, u, d, t) + \lambda^T f(x, u, d, t) \quad (9.55)$$

Using (9.55) we can rewrite (9.43) as follows,

$$J'(u(t)) = \phi(x(t_f), d(t_f), t_f) + v^T \psi(x(t_f), d(t_f), t_f) + \underbrace{\int_{t_0}^{t_f} (H(x, u, d, t) - \lambda^T \dot{x}) dt}_{J'_r} \quad (9.56)$$

To find the minimum of J' in (9.56), the increment dJ' has to be written as a function of $\delta x, \delta u, \delta \lambda, dv, dx|_{t_f}, dt_f$. For the increments $d\phi$ and $d(v^T \psi)$ of the first two terms on the right of (9.56) we have,

$$d\phi = \frac{\partial \phi}{\partial x} dx|_{t_f} + \frac{\partial \phi}{\partial t} dt|_{t_f} \quad (9.57)$$

$$d(v^T \psi) = v^T \frac{\partial \psi}{\partial x} dx|_{t_f} + v^T \frac{\partial \psi}{\partial t} dt|_{t_f} + \psi^T|_{t_f} dv \quad (9.58)$$

Applying Leibniz's rule (9.51) to the integral J'_r in (9.56) gives for the increment dJ'_r ,

$$dJ'_r = \int_{t_0}^{t_f} \left(\frac{\partial H}{\partial x} \delta x + \frac{\partial H}{\partial u} \delta u + \frac{\partial H}{\partial \lambda} \delta \lambda - \dot{x}^T \delta \lambda - \lambda^T \delta \dot{x} \right) dt + (H - \lambda^T \dot{x}) dt|_{t_f} \quad (9.59)$$

To eliminate the variation $\delta \dot{x}$ in (9.59) use,

$$\int_{t_0}^{t_f} -\lambda^T \delta \dot{x} dt = -\lambda^T \delta x|_{t_f} + \int_{t_0}^{t_f} \dot{\lambda}^T \delta x dt \quad (9.60)$$

Equation (9.60) follows from integration by parts. Using (9.45) equation (9.60) becomes,

$$\int_{t_0}^{t_f} -\lambda^T \delta \dot{x} dt = -\lambda^T \delta x|_{t_f} + \lambda^T \dot{x} dt|_{t_f} + \int_{t_0}^{t_f} \dot{\lambda}^T \delta x dt \quad (9.61)$$

Substitution of (9.61) in (9.59) and adding the terms (9.57), (9.58) finally gives,

$$\begin{aligned}
dJ' = & \left(\frac{\partial \phi}{\partial x} + v^T \frac{\partial \psi}{\partial x} - \lambda^T \right) dx \Big|_{t_f} + \psi^T \Big|_{t_f} dv + \\
& \left(\frac{\partial \phi}{\partial t} + v^T \frac{\partial \psi}{\partial t} + H - \underbrace{\lambda^T \dot{x} + \lambda^T \dot{x}}_{=0} \right) dt \Big|_{t_f} \\
& + \int_{t_0}^{t_f} \left(\left(\frac{\partial H}{\partial x} + \dot{\lambda}^T \right) \delta x + \frac{\partial H}{\partial u} \delta u + \left(\frac{\partial H}{\partial \lambda} - \dot{x}^T \right) \delta \lambda \right) dt
\end{aligned} \tag{9.62}$$

Now at a local minimum of J' the value dJ' in (9.62) should be zero for all possible variations δu , δx , $\delta \lambda$ and also differentials dv , $dx \Big|_{t_f}$ and, if the final time is free, $dt \Big|_{t_f}$. Therefore necessary conditions for a local minimum of J' are,

$$\text{State equation: } \dot{x} = \left(\frac{\partial H}{\partial \lambda} \right)^T = f, \quad t_0 \leq t \leq t_f \tag{9.63}$$

$$\text{Costate equation: } -\dot{\lambda} = \left(\frac{\partial H}{\partial x} \right)^T = \left(\frac{\partial f}{\partial x} \right)^T \lambda + \left(\frac{\partial L}{\partial x} \right)^T, \quad t_0 \leq t \leq t_f \tag{9.64}$$

$$\text{Stationarity condition: } 0 = \left(\frac{\partial H}{\partial u} \right)^T = \left(\frac{\partial f}{\partial u} \right)^T \lambda + \left(\frac{\partial L}{\partial u} \right)^T, \quad t_0 \leq t \leq t_f \tag{9.65}$$

$$\text{Boundary conditions: } x(t_0)$$

$$\begin{aligned}
& \psi(x(t_f), t_f) = 0 \\
& \left(\frac{\partial \phi}{\partial x} + v^T \frac{\partial \psi}{\partial x} - \lambda^T \right) \Big|_{t_f} = 0 \\
& \left(\frac{\partial \phi}{\partial t} + v^T \frac{\partial \psi}{\partial t} + H \right) \Big|_{t_f} = 0 \text{ if } t_f \text{ is free}
\end{aligned} \tag{9.66}$$

For some optimal control problems the final state $x(t_f)$ is entirely fixed. This can be described by,

$$\psi(x(t_f)) = x(t_f) - x^{ref} \tag{9.67}$$

where x^{ref} is the prescribed fixed final state. However, the fixed final state may be described more easily and shortly, without the use of the function ψ ,

$$x(t_f) = x^{ref} \tag{9.68}$$

Because (9.68) implies $dx|_{t_f} = 0$ from equation (9.62) observe that the third boundary condition in (9.66) no longer applies. In summary if the final state is entirely fixed this is described by (9.68) that replaces the 2nd second and 3rd boundary condition in (9.66). Equation (9.68) describes n_x boundary conditions.

The necessary conditions (9.63)-(9.66) consist of two dynamic equations, the state and costate equation, which are coupled through the stationarity condition. Together these are $2n_x$ differential equations where n_x is the dimension of x (and also λ). There are also precisely $2n_x$ boundary conditions specified by equation (9.66) if the final time is fixed and if there are no terminal constraints (ψ is not present). Half of the initial conditions, namely $x(t_0)$, is specified and half of the terminal conditions. Such problems are called two point boundary value problems because we have to find the missing n_x boundary conditions at the initial time, namely $\lambda(t_0)$, and also the missing n_x at the final time, such that the coupled dynamic equations and the other boundary conditions are satisfied. If terminal equality constraints, represented by $\psi \in R^{n_\psi}$, are present, we also have to solve for $v \in R^{n_\psi}$. This is possible because $\psi \in R^{n_\psi}$ are precisely n_ψ additional equations that have to be satisfied. When the final time is free we also have to solve for t_f . Then from equation (9.66) observe that

$$\left(\frac{\partial \phi}{\partial t} + v^T \frac{\partial \psi}{\partial t} + H \right) \Big|_{t_f} = 0 \text{ is the additional equation that has to be satisfied. The}$$

complete problem is still called a two point boundary value problem.

Two point boundary value problems, in general, are rather difficult to solve numerically. Using the special structure of the two point boundary value problem associated with optimal control problems enables us to develop suitable algorithms to solve the optimal control problems numerically.

9.4 Summary & Pontryagin's Minimum Principle

1. General optimal control problem formulation (no control and state constraints)

Given the system,

$$\dot{x} = f(x, u, d, t) \text{ or } f(x, u, d, t) - \dot{x} = 0, \quad x \in R^{n_x}, \quad u \in R^{n_u}, \quad d \in R^{n_d} \quad (9.69)$$

with the known initial condition

$$x(t_0) = x_0 \quad (9.70)$$

and given the known time evolution of the external variables,

$$d(t), \quad t_0 \leq t \leq t_f \quad (9.71)$$

find the optimal control $u^*(t)$, $t_0 \leq t \leq t_f$ that minimizes the cost function

$$J(u(t)) = \phi(x(t_f), d(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, d, t) dt \quad (9.72)$$

where possibly the following terminal constraints have to be satisfied,

$$\psi(x(t_f), d(t_f), t_f) = 0, \quad \psi \in R^{n_\psi}, \quad n_\psi \leq n_x \quad (9.73)$$

2. Necessary optimality conditions

$$\text{State equation: } \dot{x} = \left(\frac{\partial H}{\partial \lambda} \right)^T = f, \quad t_0 \leq t \leq t_f \quad (9.74)$$

$$\text{Costate equation: } -\dot{\lambda} = \left(\frac{\partial H}{\partial x} \right)^T = \left(\frac{\partial f}{\partial x} \right)^T \lambda + \left(\frac{\partial L}{\partial x} \right)^T, \quad t_0 \leq t \leq t_f \quad (9.75)$$

$$\text{Stationarity condition: } 0 = \left(\frac{\partial H}{\partial u} \right)^T = \left(\frac{\partial f}{\partial u} \right)^T \lambda + \left(\frac{\partial L}{\partial u} \right)^T, \quad t_0 \leq t \leq t_f \quad (9.76)$$

Boundary conditions: $x(t_0)$

$x(t_f) = x^{ref}$ if the final state is entirely fixed

$$\text{If the final state } x(t_f) \text{ is not entirely fixed: } \left(\frac{\partial \phi}{\partial x} + v^T \frac{\partial \psi}{\partial x} - \lambda^T \right) \Big|_{t_f} = 0 \quad (9.77)$$

$$\text{If } t_f \text{ is free: } \left(\frac{\partial \phi}{\partial t} + v^T \frac{\partial \psi}{\partial t} + H \right) \Big|_{t_f} = 0$$

3. A two point boundary value problem

Equation (9.74) are n_x differential equations running forward in time. Equation (9.75) are n_x differential equations running backward in time. These equations are coupled by the stationarity condition (9.76). $x(t_0)$ represents n_x initial conditions. The second boundary condition in (9.77) represents n_x terminal conditions. So for each differential equation there is one boundary condition. Because these boundary conditions are specified at two time points, namely the initial and final time, this is called a two point boundary value problem because we have to find the missing n_x boundary conditions at both points. If the final time is free the third boundary condition in (9.77) represents one additional equation that has to be satisfied. If there are terminal constraints $\psi \in R^{n_\psi}$, $n_\psi \leq n_x$ these act as n_ψ equations that have to be satisfied. Using these the associated $v \in R^{n_\psi}$ Lagrange multipliers can be solved as well. This complete problem is also called a two point boundary value problem.

4. Pontryagin's Minimum Principle

When the controls are upper and/or lower bounded, or in any other way restricted, the necessary optimality conditions of the optimal control problem are unchanged except for the stationarity condition (9.76). According to Pontryagin's Minimum Principle the optimal controls $u^*(t)$ at each time $t_0 \leq t \leq t_f$ are admissible, meaning they satisfy the restrictions, and minimize the Hamiltonian,

$$\begin{aligned} H(x^*(t), u^*(t), \lambda^*(t), t) &\leq H(x^*(t), u(t), \lambda^*(t), t), \\ u^*(t), u(t) \text{ admissible, } t_0 &\leq t \leq t_f \end{aligned} \quad (9.78)$$

Observe that the stationarity condition (9.76) is a special case of (9.78) because if there are no restrictions all controls are admissible and the one that minimizes the Hamiltonian satisfies (9.76).

10. Gradient algorithms to solve optimal control problems. LQ and LQG feedback design.

10.1 Gradient algorithms to solve optimal control problems

After reading this chapter the student is able to

1. Recognise four types of optimal control algorithms
2. Recall the basic steps of the gradient algorithm
3. Know the advantages and disadvantages of gradient algorithms

The necessary optimality conditions for a solution of an optimal control problem are:

1. The state equation (9.74)
2. The co state equation (9.75)
3. The stationarity condition (9.76) or Pontryagin's Minimum Principle (9.78)
4. The boundary conditions (9.77)

Usually 3 of these 4 conditions can be satisfied easily and a-priori. Algorithms that find solutions of optimal control problems iterate towards satisfying the 4th condition. So basically there are four types of optimal control algorithms. In this section we will consider only one type called gradient algorithms. Gradient algorithms satisfy a-priori conditions 1,2 and 4 and iterate towards satisfying the 3rd. Gradient algorithms are especially suitable for problems with fixed final time, no terminal constraints and no state constraints. Free final time and terminal constraints can be handled but they seriously complicate the algorithm and furthermore the computation time often increases dramatically. Here we present the gradient algorithm for problems without terminal constraints and with fixed final time.

Gradient algorithm

- 1 Guess $u(t)$, $t_0 \leq t \leq t_f$ the optimal control
- 2 Starting from $x(t_0)$ numerically integrate the system equation (9.74) using the control $u(t)$, $t_0 \leq t \leq t_f$. Store $x(t)$, $t_0 \leq t \leq t_f$.
- 3 From $x(t_f)$ and ϕ compute $\lambda(t_f) = \left(\frac{\partial \phi}{\partial x} \right)^T \Big|_{t_f}$.
- 4 Starting at $\lambda(t_f)$ integrate the co-state equation (9.75) backward in time using $x(t)$, $t_0 \leq t \leq t_f$ to compute $\left(\frac{\partial f}{\partial x} \right)^T \left(\frac{\partial L}{\partial x} \right)^T$. Store $\lambda(t)$, $t_0 \leq t \leq t_f$.
- 5 Using $u(t)$, $x(t)$, $\lambda(t)$ compute $\left(\frac{\partial H}{\partial u} \right)^T = \left(\frac{\partial f}{\partial u} \right)^T \lambda + \left(\frac{\partial L}{\partial u} \right)^T$, $t_0 \leq t \leq t_f$
- 6 Select $\alpha > 0 \in R^1$ such that $u'(t) = u(t) - \alpha \left(\frac{\partial H}{\partial u} \right)^T$ minimizes the cost function. This is called a line search. $u(t) := u'(t)$
- 7 If the cost function improvement is below a tolerance $u(t) = u^*(t)$, goto 8 else goto 2.
- 8 Verify if $\frac{\partial H}{\partial u_i}$ is "small" if u_i is not at a bound $i = 1, 2, \dots, n_u$

Step 1 is an important step because if the guess of the optimal control is poor the algorithm needs more iterations (computation time) to find the solution or it may not find a solution at all. Also depending on the control guess the algorithm may find a local solution.

In performing **step 2** we satisfy a-priori the state equation (9.74) and the initial boundary condition $x(t_0)$.

In performing **step 3 and 4** we satisfy a-priori the terminal boundary condition

$\lambda(t_f) = \left(\frac{\partial \phi}{\partial x} \right)^T \Big|_{t_f}$ and the co state equation (9.75). So only the stationarity condition

(Pontryagin's Minimum Principle in the case of control constraints) is not satisfied a-priori.

Step 5 and 6 improve the satisfaction of the stationarity condition. This also holds for the minimum principle in the case of control constraints. Step 5 uses the gradient $\frac{\partial H}{\partial u}$ to improve the control: it searches for a control improvement in the direction $\frac{\partial H}{\partial u}$. Therefore this type of algorithm is called a gradient algorithm.

Searching in the gradient direction is not very efficient, especially not if one is near the optimal solution. We will use a so called conjugate gradient algorithm to solve optimal control problems in Matlab. In this case the search direction during each iteration is not equal to $\frac{\partial H}{\partial u}$ but is constructed from current and passed values of

$\frac{\partial H}{\partial u}$ obtained in previous iterations. This improves the speed of convergence of the algorithm. Simple gradient algorithms do not perform a line search but use a fixed small value for α . Although the line search itself takes additional computation time, it generally improves the efficiency of the algorithm. By evaluating the cost function $J(u'(t))$ we can search for a global minimum in the search direction. Implementing such a so called global line search increases the chances of finding a global solution of the optimal control problem. In the Matlab algorithm that we will use a global line search is implemented.

Step 7 checks whether the improvement of the costs is sufficient to justify continuation of the algorithm. It should be noted that the improvement of the costs can be very small during some iterations. So in practice one only stops iterating when several consecutive iterations have failed to provide significant improvement of the costs.

Step 8 is a step which verifies whether the stationarity condition (minimum principle in the case of control constraints) is approximately satisfied.

The advantageous of gradient algorithms are:

1. The inaccuracy of intermediate solutions is roughly equal over the time-interval $t_0 \leq t \leq t_f$.
2. The algorithm does not require accurate initial guesses of the control which may save dynamic analysis of the optimal control problem (partly thanks to 1).

3. The stationarity condition (minimum principle in the case of control constraints) is satisfied numerically which may save analysis of the optimal control problem (for instance if the optimal control problem is singular) and programming effort.

The disadvantages of the gradient algorithm are:

1. The algorithm is not very efficient computationally.
2. The algorithm is not very well suited to solve optimal control problems with terminal constraints and/or free final time.
3. The algorithm is not suitable to solve problems with state constraints (most algorithms based on the necessary optimality conditions (9.74)-(9.77) are not suitable to solve problems with state constraints).

10.2 The Mayer formulation of optimal control problems.

After reading this section the reader should be able to:

1. Mention two reasons for application of the Mayer formulation
2. Describe optimal control problems using the Mayer formulation

The reason why the Mayer formulation of optimal control problems is important are twofold.

1. The formulation offers a way to compute the running costs numerically
2. The formulation transforms optimal control problems with running costs into problems with only terminal costs

1. clearly is important for any algorithm that computes the costs $J(u(t))$ during some stage of the algorithm. Note that the gradient algorithm with a line search does so during the line search. 2. is important because it may simplify the programming of optimal control problems. Furthermore it is important for the transformation of optimal control problems into constrained function minimization problems which we shall do in chapters 11 and 12. This is only possible for optimal control problems with terminal costs. Only by using the Mayer formulation we can still include running costs.

Consider $J_r(u(t))$ the running costs of an optimal control problem:

$$J_r(u(t)) = \int_{t_0}^{t_f} L(x, u, d, t) dt \quad (10.1)$$

Note the following equivalence,

$$J_r(t) = \int_{t_0}^t L(x, u, d, t) dt \Leftrightarrow \frac{\partial J_r}{\partial t} = L(x, u, d, t), J_r(t_0) = 0 \quad (10.2)$$

According to the right hand part of (10.2) the running costs can be introduced as an additional state variable as follows,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n_x} \\ \dot{J}_r \end{bmatrix} = \begin{bmatrix} f_1(x, u, d, t) \\ f_2(x, u, d, t) \\ \vdots \\ f_{n_x}(x, u, d, t) \\ L(x, u, d, t) \end{bmatrix}, \quad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_x} \\ J_r \end{bmatrix} = \begin{bmatrix} x \\ J_r \end{bmatrix} = x', \quad \begin{bmatrix} f_1(x, u, d, t) \\ f_2(x, u, d, t) \\ \vdots \\ f_{n_x}(x, u, d, t) \\ L(x, u, d, t) \end{bmatrix} = \begin{bmatrix} f \\ L \end{bmatrix} = f' \quad (10.3)$$

Then if,

$$J_r(t_0) = x'_{n_x+1}(t_0) = 0 \quad (10.4)$$

we obtain,

$$J_r(t_f) = x'_{n_x+1}(t_f) = \int_{t_0}^{t_f} L(x, u, d, t) dt \quad (10.5)$$

From equations (10.1) and (10.5) observe that,

$$J_r(u(t)) = J_r(t_f) \quad (10.6)$$

Furthermore,

$$J(u(t)) = \phi(x(t_f)) + J_r(t_f) = \phi(x(t_f)) + x'_{n_x+1}(t_f) = \phi'(x'(t_f)) \quad (10.7)$$

so we can reformulate the original optimal control problem entirely using x' , f' and ϕ' . Doing so we have obtained an optimal control problem formulation without running costs.

10.3 LQ and LQG feedback design

After reading this section the reader should be able to:

1. Recall and explain 3 reasons to apply optimal LQ and LQG feedback design
2. Draw block diagrams of the feedback and mention the equations which relate the inputs and outputs of these blocks

Reconsider the state feedback part of the optimal control system in figure 7.5 on page 4. The associated control problem is visualized by figure 10.1 and can be stated as follows,

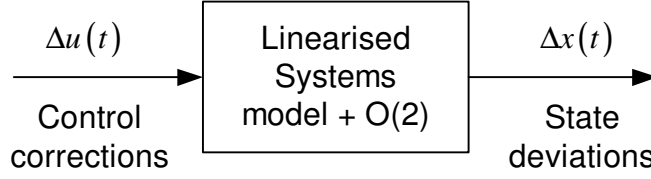


Figure 10.1: Relation between control corrections and state deviations

Find small control corrections $\Delta u(t)$ which keep the state perturbations $\Delta x(t)$ small. The systems model that relates the control corrections $\Delta u(t)$ and the state perturbations $\Delta x(t)$ is the original non-linear systems model $\dot{x} = f(x, u, d, t)$. Because we are considering small control and state perturbations the linearized model about the optimal trajectory $u^*(t), x^*(t)$,

$$\Delta \dot{x}(t) = A(t) \Delta x + B(t) \Delta u, \quad A(t) = \left. \frac{\partial f}{\partial x} \right|_{x(t)=x^*(t), u(t)=u^*(t)}, \quad B(t) = \left. \frac{\partial f}{\partial u} \right|_{x(t)=x^*(t), u(t)=u^*(t)} \quad (10.8)$$

actually is a good approximation. And it will turn out very convenient to use it. According to a Taylor series expansion the errors caused by using this linearised model consist of quadratic and higher order terms in Δx and Δu . If Δx and Δu are small the second order terms dominate these errors.

Consider the following quadratic cost function,

$$J(\Delta u(t)) = \Delta x^T(t_f) F \Delta x(t_f) + \int_{t_0}^{t_f} [\Delta x^T(t) Q(t) \Delta x(t) + \Delta u^T(t) R(t) \Delta u(t)] dt$$

$$F \geq 0, Q \geq 0, R \geq 0 \quad (10.9)$$

1. The quadratic cost function (10.9) minimizes second order terms in Δx and Δu and therefore the errors that we make by using a linearized model.
2. The quadratic criterion (10.9) tries to keep the control and state perturbations $\Delta u(t)$ and $\Delta x(t)$ small.

3. The solution of the optimal control problem described by the model (10.8) and the cost function (10.9) is in feedback form as we know from the compulsory exercise of lecture 9: $\Delta u(t) = -L(t)\Delta x(t)$. The optimal feedback gain $L(t)$ can be computed in advance (off-line). From figure 7.5 on page 4 observe that this is precisely what we need because the only on-line computation that has to be performed is the multiplication of the stored values $L(t)$ with the measured state perturbations $\Delta x(t)$. This can be done on-line very quickly.

The above three arguments justify the LQ (Linear Quadratic) state feedback design.

In the case of dynamic LQG output feedback the feedback consists of a state estimator in combination with state feedback, see figure 10.2

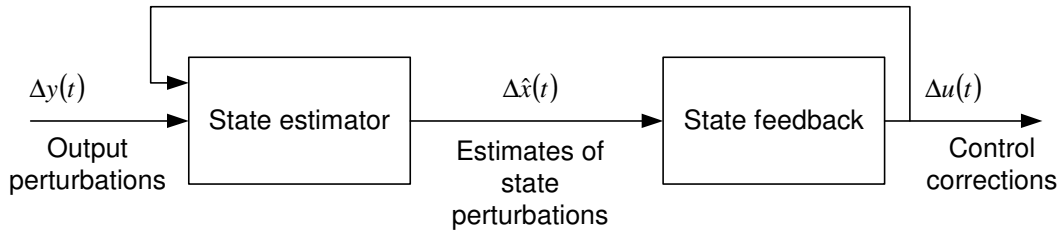


Figure 10.2: Dynamic LQG output feedback

The state feedback part is described by,

$$\Delta u(t) = -L(t)\Delta \hat{x}(t) \quad (10.10)$$

where $\Delta \hat{x}(t)$ are estimates of the state perturbations computed by the state estimator. The feedback gain $L(t)$ is identical to the state feedback gain obtained from the LQ problem. The state estimator is the Kalman filter, a dynamical system described by the following differential equation,

$$\dot{\Delta \hat{x}}(t) = A(t)\Delta \hat{x}(t) + B(t)\Delta u(t) + K(t)(\Delta y(t) - C(t)\Delta \hat{x}(t)) \quad (10.11)$$

with initial condition

$$\Delta \hat{x}(t_0) = 0 \quad (10.12)$$

As inputs the Kalman filter has the output perturbations $\Delta y(t)$ and the control corrections $\Delta u(t)$. The matrices $A(t), B(t)$ are those of the linearised system about the optimal trajectory given by equation (10.8). The matrix $C(t)$ is obtained from a linearization of the output equation,

$$y = g(x, d, t) \quad (10.13)$$

about the optimal trajectory:

$$C(t) = \left. \frac{\partial g}{\partial x} \right|_{x(t)=x^*(t), u=u^*(t)} \quad (10.14)$$

The only part of the state estimator that has to be determined is the optimal estimator (Kalman) gain $K(t), t_0 \leq t \leq t_f$. This gain follows from an optimal estimation problem that is dual (mathematically identical) to the LQ regulator problem. The optimal estimator (Kalman) gain $K(t), t_0 \leq t \leq t_f$ can be computed in advance (off-line) according to,

$$K(t) = P(t) C^T(t) W^{-1}(t) \quad (10.15)$$

where $P(t)$ is the solution of the Riccati matrix differential equation,

$$\dot{P} = AP + PA^T - PC^T W^{-1} CP + V, \quad t_0 \leq t \leq t_f \quad (10.16)$$

with initial condition,

$$P(t_0) = G \quad (10.17)$$

In equations (10.15)-(10.17) $V(t) \in R^{n_x \times n_x}, W(t) \in R^{n_y \times n_y}$ are matrices which represent model and measurement errors (uncertainty) respectively. Finally $G \in R^{n_x \times n_x}$ is a matrix that represents initial state errors (uncertainties). Actually the optimal estimator (Kalman filter) gain $K(t), t_0 \leq t \leq t_f$ at each time t is an optimal compromise between the modeling, initial state and output errors. It can be computed in advance (off-line). Note that the first two terms,

$$A(t) \Delta \hat{x}(t) + B(t) \Delta u(t) \quad (10.18)$$

on the right in equation (10.11) describe the propagation of $\Delta \hat{x}(t)$ according to the linearised model. Consider the final term on the right in equation (10.11),

$$K(t)(\Delta y(t) - C(t)\Delta \hat{x}(t)) = K(t)(\Delta y(t) - \Delta \hat{y}(t)) \quad (10.19)$$

where $\Delta \hat{y}(t) = C(t)\Delta \hat{x}(t)$ is the estimated output according to the linearised model and the current estimate $\Delta \hat{x}(t)$ of the state perturbation. Therefore the final term (10.19) in equation (10.11) equals the Kalman gain multiplied by the difference between the measured and estimated output perturbations. This term “corrects” the propagation of $\Delta \hat{x}(t)$ according to the model.

To implement the dynamic feedback laws (10.10), (10.11) note that (10.11) can be written as,

$$\Delta \dot{\hat{x}}(t) = \underbrace{(A(t) - K(t)C(t) - B(t)L(t))}_{F(t)} \Delta \hat{x}(t) + K(t)\Delta y(t) \quad (10.20)$$

Therefore we need to store,

$$F(t), K(t), L(t) \quad (10.21)$$

Then on-line we have to numerically integrate equation (10.20) and compute equation (10.10).

10.4 Selection and interpretation of the LQ and LQG design matrices

After reading this section the reader should:

1. Know how to select the LQ and LQG design matrices

The errors made by using a linearised model to describe the behavior of the perturbations $\Delta u(t)$, $\Delta x(t)$, $\Delta y(t)$ depend mainly on the second order terms in a Taylor series expansion. These second order terms are given by,

$$\Delta x^T(t) \frac{\partial^2 f_i}{\partial x^2} \bigg|_{o(t)} \Delta x(t), i = 1, 2, \dots, n_x, \Delta u^T(t) \frac{\partial^2 f_i}{\partial u^2} \bigg|_{o(t)} \Delta u(t), i = 1, 2, \dots, n_x \quad (10.22)$$

$$\Delta x^T(t) \frac{\partial^2 g_i}{\partial x^2} \bigg|_{o(t)} \Delta x(t), i = 1, 2, \dots, n_y, \Delta u^T(t) \frac{\partial^2 g_i}{\partial u^2} \bigg|_{o(t)} \Delta u(t), i = 1, 2, \dots, n_y \quad (10.23)$$

where $o(t)$ indicates evaluation along the optimal control and state trajectory $x(t) = x^*(t)$, $u(t) = u^*(t)$. Besides minimizing errors due to the use of a linearized

model minimizing deviations from the optimal cost is also important. These deviations are dominated by the second order terms,

$$\Delta x^T(t) \frac{\partial^2 H}{\partial x^2} \Big|_{o(t)} \Delta x(t), \Delta u^T(t) \frac{\partial^2 H}{\partial u^2} \Big|_{o(t)} \Delta u(t), \Delta x^T(t_f) \frac{\partial^2 \phi}{\partial x^2} \Big|_{o(t_f)} \Delta x(t_f) \quad (10.24)$$

To minimize the errors (10.22), (10.23) and the cost deviations (10.24) we should select the square matrices $Q(t) \geq 0$ in (10.9) to be a kind of upperbound of the so

called Hessian matrices $\frac{\partial^2 f_i}{\partial x^2} \Big|_{o(t)}$, $i = 1, 2, \dots, n_x$, $\frac{\partial^2 g_i}{\partial x^2} \Big|_{o(t)}$, $i = 1, 2, \dots, n_y$, $\frac{\partial^2 H}{\partial x^2} \Big|_{o(t)}$ in

(10.22)-(10.24) [1]. Hessian matrices are defined for scalar functions of vectors. For the scalar function H of the vector x the Hessian matrix is defined by,

$$\left[\frac{\partial^2 H}{\partial x^2} \right]_{j,k} = \frac{\partial^2 H}{\partial x_j \partial x_k}, \quad j = 1, 2, \dots, n_x, \quad k = 1, 2, \dots, n_x \quad (10.25)$$

Similarly $R(t) > 0$ in (10.9) should be selected a kind of upperbound of $\frac{\partial^2 f_i}{\partial u^2} \Big|_{o(t)}$,

$i = 1, 2, \dots, n_x$, $\frac{\partial^2 g_i}{\partial u^2} \Big|_{o(t)}$, $i = 1, 2, \dots, n_y$, $\frac{\partial^2 H}{\partial u^2} \Big|_{o(t)}$. Finally $F \geq 0$ in (10.9) should be

selected a kind of upperbound of $\frac{\partial^2 \phi}{\partial x^2} \Big|_{o(t_f)}$. These upperbounds may be chosen to be

symmetric matrices because,

$$\Delta x^T Q \Delta x = \Delta x^T (Q_s + Q_a) \Delta x = \Delta x^T Q_s \Delta x + \Delta x^T Q_a \Delta x = \Delta x^T Q_s \Delta x. \quad (10.26)$$

In (10.26) $Q_s = (Q + Q^T)/2$ is a symmetric matrix for which $q_{ij} = q_{ji}$ and $Q_a = (Q - Q^T)/2$ an anti-symmetric matrix for which $q_{ij} = -q_{ji}$. Now (10.26) holds because $\Delta x^T Q_a \Delta x = 0$ for any Δx . Therefore without changing the cost function we may replace $Q(t)$ with $Q_s(t)$ and $R(t)$ with $R_s(t)$. Often we choose $Q(t)$ and $R(t)$ to be diagonal matrices. Then,

$$\Delta x^T Q \Delta x = \sum_{i=1}^{n_x} Q_{i,i} \Delta x_i^2 \quad (10.27)$$

and the i^{th} diagonal element $Q_{i,i}$ of Q determines the penalty on the squared i^{th} component of the vector Δx . So the larger $Q_{i,i}$ the larger the penalty on Δx_i^2 .

Based on the ideas outlined above, reference [7] presents numerical algorithms to compute the diagonal matrices $F \geq 0, Q(t) \geq 0, R(t) \geq 0$ in (10.9):

$$Q(t) = W_f \frac{1}{n_x} \sum_{i=1}^{n_x} D^0 \left(\left. \frac{\partial^2 f_i}{\partial x^2} \right|_{o(t)} \right) + W_g \frac{1}{n_y} \sum_{i=1}^{n_y} D^0 \left(\left. \frac{\partial^2 g_i}{\partial x^2} \right|_{o(t)} \right) + W_I D^0 \left(\left. \frac{\partial^2 H}{\partial x^2} \right|_{o(t)} \right) \quad (10.28)$$

$$R(t) = W_f \frac{1}{n_x} \sum_{i=1}^{n_x} D^0 \left(\left. \frac{\partial^2 f_i}{\partial u^2} \right|_{o(t)} \right) + W_g \frac{1}{n_y} \sum_{i=1}^{n_y} D^0 \left(\left. \frac{\partial^2 g_i}{\partial u^2} \right|_{o(t)} \right) + W_I D^0 \left(\left. \frac{\partial^2 H}{\partial u^2} \right|_{o(t)} \right) \quad (10.29)$$

$$F = W_I D^0 \left(\left. \frac{\partial^2 \phi}{\partial x^2} \right|_{o(t_f)} \right) \quad (10.30)$$

In equation (10.28) $D^0 \left(\left. \frac{\partial^2 f_i}{\partial x^2} \right|_{o(t)} \right)$ is a nonnegative diagonal matrix upperbounding

the matrix $\left. \frac{\partial^2 f_i}{\partial x^2} \right|_{o(t)}$ that has the same dimensions. To compute

$F \geq 0, Q(t) \geq 0, R(t) \geq 0$ in (10.28)-(10.30) three weighting factors W_f, W_g, W_I have to be specified. W_f weighs the relative importance of minimizing the magnitudes of the second order terms (10.22) relating to the error due to linearization of the state equation. W_g weighs the relative importance of minimizing the magnitudes of the second order terms (10.23) relating to the error due to linearization of the output equation. W_I weighs the importance of minimizing the magnitude of the deviations (10.24) from the optimal costs. So using the numerical algorithms from reference [7] the selection of the matrices $F \geq 0, Q(t) \geq 0, R(t) \geq 0$ reduces to the selection of the three scalar design parameters W_f, W_g, W_I in equations (10.28)-(10.30)!

The square matrices $V(t) \in R^{n_x \times n_x} \geq 0, W(t) \geq 0 \in R^{n_y \times n_y}$ represent the model and measurement uncertainty respectively at each time t . Very often also these matrices are taken to be diagonal. In this case the i^{th} diagonal element of $V(t)$ represents the

average squared error in the i^{th} state derivative of the model at time t . Similarly the i^{th} diagonal element of $W(t)$ represents the average squared error in the i^{th} output at time t . The matrix $G \in R^{n_x \times n_x}$ represents the errors in the initial state $x(t_0)$. If this matrix is diagonal the i^{th} diagonal element represents the average squared error in the i^{th} component $x_i(t_0)$ of $x(t_0)$.

Finally for the selection of the LQG design matrices it is important to realize that only if,

$$F \geq 0, Q(t) \geq 0, R(t) > 0, G \geq 0, V(t) \geq 0, W(t) > 0 \quad (10.31)$$

the two dual LQ regulator problems are guaranteed to have a solution, which moreover is unique.

10.5 Time-invariant infinite horizon LQ problems and stability.

After reading this section the reader should:

1. Recall and know when to apply the infinite horizon time-invariant LQ optimal feedback design

Assume we have an LQ regulator problem where the system is time-invariant and also the matrices of the cost function. Furthermore assume that the time-invariant system is controllable and that the final time is infinity,

$$\begin{aligned} A(t) = A, B(t) = B, Q(t) = Q \geq 0, R(t) = R > 0, t_f \rightarrow \infty \\ (A, B) \text{ controllable} \end{aligned} \quad (10.32)$$

then the solution $S(t)$ of the Riccati differential equation and the associated optimal feedback gain $L(t)$ have the following properties illustrated by figure 10.3,

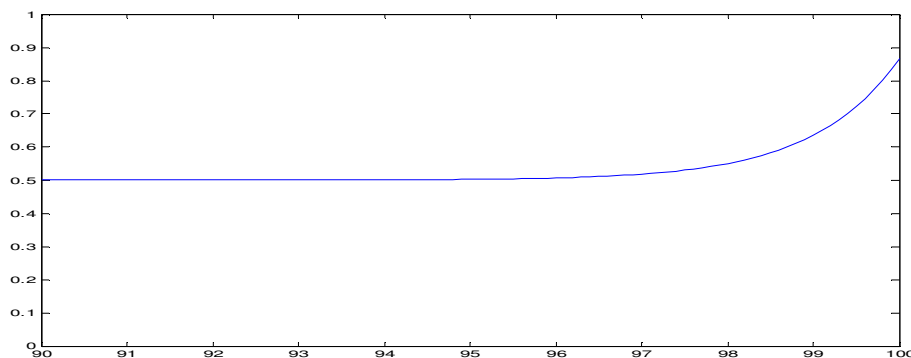


Figure 10.3: Typical behavior of each component of $S(t)$

$$S(t) \rightarrow S \Rightarrow L(t) \rightarrow L = R^{-1}B^T S, \dot{S} \rightarrow 0 \quad (10.33)$$

where S is the unique solution of the so called Algebraic Riccati equation,

$$0 = A^T S + SA - SBR^{-1}B^T S + Q \quad (10.34)$$

Furthermore the optimal control is

$$u^*(t) = -Lx(t), t_0 \leq t \leq \infty \quad (10.35)$$

and the optimal closed loop system

$$\dot{x} = (A - BL)x$$

is (asymptotically) stable. In other words the eigenvalues of the matrix,

$$A - BL \quad (10.36)$$

all have a negative real part. We remark here that the condition (A, B) controllable in (10.32) may be replaced with the condition (A, B) stabilizable which is a weaker condition. The solution of the algebraic Riccati equation (10.33) can very easily and quickly be computed with the Matlab function `lqr.m`. This function requires (A, B) to be controllable (and also $Q \geq 0, R > 0$) and checks for this.

In practice this result is especially important if we want to control a non-linear system about a steady state for a long period of time. Then the linearized system is time-invariant and time tends to infinity and we want all small deviations to tend to zero (stability).

11. Transformation of optimal control problems into constrained function minimization problems.

The reason to perform this transformation is to be able to solve any type of optimal control problem by means of function minimization. By any type we mean optimal control problems with state equality/inequality constraints and/or control equality/inequality constraints and/or terminal equality/inequality constraints and/or free final time. The disadvantage of solving optimal control problems by means of constrained function minimization is that when the state and/or the control vector has a large dimension the algorithms become inefficient (long computation times) or even infeasible (solutions are not obtained).

11.1 Constrained function minimization

After reading this section the reader should:

1. Know how to transform function minimization problems with equality and inequality constraints into function minimization problems without constraints.
2. State the necessary conditions for a solution of an unconstrained function minimization problem.

Consider two straight lines described by,

$$u_1 = x_1 + 1, u_2 = -x_2 + 2 \quad (11.1)$$

where (u_1, x_1) is a point on the first line and (u_2, x_2) is a point on the second line, see figure 11.1.

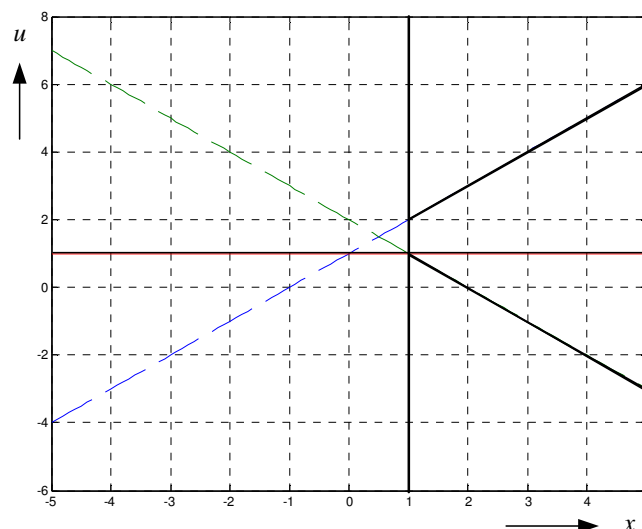


Figure 11.1: Visualization of the function minimization problem

Suppose that we want to find the points on these lines which have the smallest distance. Since these lines intersect these two point are equal so,

$$x_1 = x_2, u_1 = u_2, \quad (11.2)$$

and the point of intersection is computed as follows,

$$x_1 + 1 = -x_1 + 2 \Rightarrow 2x_1 = 1 \Rightarrow x_1 = \frac{1}{2} \Rightarrow u_1 = 1\frac{1}{2} \quad (11.3)$$

As we have seen in section 9.1 this problem can be formulated as a constrained function minimization problem. Minimize,

$$J(u, x) = \frac{1}{2}(u_1 - u_2)^2 + \frac{1}{2}(x_1 - x_2)^2 \quad (11.4)$$

with constraints

$$u_1 - x_1 - 1 = 0, u_2 + x_1 - 2 = 0 \quad (11.5)$$

Now we change this function minimization problem with equality constraints by introducing two additional inequality constraints.

$$x_1 \geq 1, u_2 \leq 1 \quad (11.6)$$

These inequality constraints are visualized in figure 11.1 by the horizontal and vertical straight line which represent the boundary. Furthermore the non dashed parts of the other two lines are the parts of the lines that satisfy the inequality constraints.

We may write the inequality constraints (11.6) as follows,

$$x_1 - 1 \geq 0, -u_2 + 1 \geq 0 \quad (11.7)$$

It is always possible to transform any inequality constraint into one which is larger and/or equal to zero. From figure 11.1 it is clear that the solution is,

$$(u_1, x_1) = (2, 1), (u_2, x_2) = (1, 1) \quad (11.8)$$

From section 9.1 we know that function minimization with equality constraints can be handled by introducing Lagrange multipliers. Now inequality constraints like those in (11.7) can be handled by introducing so called slack variables y by which we transform the inequality constraints (11.7) into the following equality constraints,

$$x_1 - 1 = y_1^2, -u_2 + 1 = y_2^2 \quad (11.9)$$

or equivalently,

$$x_1 - 1 - y_1^2 = 0, -u_2 + 1 - y_2^2 = 0 \quad (11.10)$$

If we can find y_1, y_2 such that the equality constraints (11.10) are satisfied, because the squares y_1^2, y_2^2 are always non-negative, by definition the inequality constraints (11.9) are satisfied. The 'price' we have to pay to transform inequality constraints into equality constraints is the introduction of an additional slack variable for each inequality constraint.

Summarising by introducing a slack variable for each inequality constraint we can transform function minimization problems with inequality constraints into functions minimization problems with equality constraints which from chapter 9 can be solved using Lagrange multipliers. Doing so our example becomes as follows. Minimize

$$J(u, x) = \frac{1}{2}(u_1 - u_2)^2 + \frac{1}{2}(x_1 - x_2)^2 \quad (11.11)$$

with the equality constraints

$$f = \begin{bmatrix} u_1 - x_1 - 1 \\ u_2 + x_1 - 2 \\ x_1 - 1 - y_1^2 \\ -u_2 + 1 - y_2^2 \end{bmatrix} = 0 \quad (11.12)$$

Introducing Lagrange multipliers the problem becomes as follows. Minimize

$$\begin{aligned} J'(u, x, y, \lambda) = & \frac{1}{2}(u_1 - u_2)^2 + \frac{1}{2}(x_1 - x_2)^2 + \lambda_1(u_1 - x_1 - 1) + \lambda_2(u_2 + x_1 - 2) \\ & + \lambda_3(x_1 - 1 - y_1^2) + \lambda_4(-u_2 + 1 - y_2^2) \end{aligned} \quad (11.13)$$

The necessary condition for a solution are,

$$\frac{\partial J'}{\partial u} = (u_1 - u_2 + \lambda_1, -u_1 + u_2 + \lambda_2 - \lambda_4) = 0 \quad (11.14)$$

$$\frac{\partial J'}{\partial x} = (x_1 - x_2 - \lambda_1 + \lambda_3, -x_1 + x_2 + \lambda_2) = 0 \quad (11.15)$$

$$\frac{\partial J'}{\partial \lambda} = f = 0 \quad (11.16)$$

$$\frac{\partial J'}{\partial y} = (-2\lambda_3 y_1, -2\lambda_4 y_2) = 0 \quad (11.17)$$

Now knowing the solution (11.8) we can verify whether this is a solution of these necessary conditions. From (11.8), (11.16)=(11.12) we obtain,

$$y_1 = y_2 = 0 \quad (11.18)$$

From (11.8) and (11.14) we obtain,

$$\lambda_1 = -1, \lambda_2 - \lambda_4 - 1 = 0 \quad (11.19)$$

From (11.8), (11.15) and (11.19) we obtain

$$\lambda_3 = 1, \lambda_2 = 0, \lambda_4 = -1 \quad (11.20)$$

Indeed $u = (2, 1)^T$, $x = (1, 1)^T$, $\lambda = (-1, 0, 1, -1)^T$, $y = (0, 0)^T$ satisfy all the necessary conditions (11.14)-(11.17). Finally we make two remarks. Firstly observe from (11.7) and (11.9) that $y = (0, 0)^T$ indicates that the solution is precisely at the edge allowed by the inequality constraints (11.7). At these edges the equalities

$$x_1 - 1 = 0, -u_2 + 1 = 0 \quad (11.21)$$

hold. Secondly $\lambda_2 = 0$ indicates that the constraint associated to this Lagrange multiplier namely $u_2 + x_1 - 2 = 0$ does not influence the solution, or in other words, at the minimum the sensitivity of the cost function for this constraint is zero. This can also be seen from figure 11.1 Without the constraint that (u_2, x_2) should lie on the second line the solution would have been the same.

The Matlab code to solve the constrained function minimization problem presented and solved in this section is presented at the end of this section.

Summary

Function minimization problems with equality and inequality constraints can be solved as ordinary function minimization problems without constraints by introducing Lagrange multipliers and slack variables.

Matlab code.

% Exlec11: Example script lecture 11

```
% Set options vector: opt(1) : 1=display intermediate results  
%           opt(13): number of equality constraints  
opt(1)=1; opt(13)=2;  
% Initial guess for solution x(1:2)=u, x(3:4)=x  
xguess=[1;1;1;1];  
% Perform the constraint function minimization and return the solution x  
[x,options,lambda]=constr('flec11',xguess,opt);  
% Split x in u,x  
u=x(1:2); x=x(3:4);  
% Display the solution;  
disp(['(u1,x1)=(' num2str(u(1)) ',' num2str(x(1)) ')']);  
disp(['(u2,x2)=(' num2str(u(2)) ',' num2str(x(2)) ')']);  
% Display Lambda  
disp('Lambda:'); lambda
```

% flec11: Constrained function minimization example lecture 11

```
function [f,g]=flec11(x);  
% Split x in u and x  
u=x(1:2); x=x(3:4);  
% f: function to be minimized  
f=0.5*(u(1)-u(2))*(u(1)-u(2))+0.5*(x(1)-x(2))*(x(1)-x(2));  
% g: constraints, first the 2 equality constr.  
%   then the 2 inequality constr <=0!!  
g=[u(1)-x(1)-1; u(2)+x(2)-2; -x(1)+1; u(2)-1];
```

11.2 Transformation of optimal control problems into constrained function minimization problems by control parameterization.

After reading this section the reader should:

1. Know how to transform any type of optimal control problem into a function minimization problems with equality and inequality constraints

Suppose we have an optimal control problem. Then given a certain control $u(t)$, $t_0 \leq t \leq t_f$ using the augmented system description,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n_x} \\ \dot{J}_r \end{bmatrix} = \begin{bmatrix} f_1(x, u, d, t) \\ f_2(x, u, d, t) \\ \vdots \\ f_{n_x}(x, u, d, t) \\ L(x, u, d, t) \end{bmatrix}, \quad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_x} \\ J_r \end{bmatrix} = \begin{bmatrix} x \\ J_r \end{bmatrix} = x', \quad \begin{bmatrix} f_1(x, u, d, t) \\ f_2(x, u, d, t) \\ \vdots \\ f_{n_x}(x, u, d, t) \\ L(x, u, d, t) \end{bmatrix} = \begin{bmatrix} f \\ L \end{bmatrix} = f' \quad (11.22)$$

used in the Mayer formulation we can compute the costs

$$J(u(t)) = \phi(x(t_f)) + J_r(t_f) = \phi(x(t_f)) + x'_{n_x+1}(t_f) = \phi'(x'(t_f)) \quad (11.23)$$

associated to this control. To compute these costs we numerically integrate the augmented system equation (11.22) and so we automatically satisfy the system equation,

$$\dot{x} = f(x, u, d, t) \quad (11.24)$$

To represent the control $u(t)$, $t_0 \leq t \leq t_f$, $u \in R^{n_u}$ we need n_u numbers at each time instant $t_0 \leq t \leq t_f$. Since there are in principle infinitely many time instants in between t_0 and t_f we need infinitely many numbers. When applying gradient algorithms this problem is solved by time discretisation and interpolation which is performed automatically during numerical integration.

Suppose we are able to represent the control by finitely many vectors,

$$u_k \in R^{n_u}, \quad k = 0, 1, \dots, N-1 \quad (11.25)$$

If this is possible we call the vectors (11.25) a parameterization of the control $u(t)$, $t_0 \leq t \leq t_f$. Figure 11.2 shows one solution to this problem where we keep the control constant over small time-intervals,

$$u(t) = u(t_k) = u_k, \quad t_k \leq t < t_{k+1}, \quad k = 0, 1, \dots, N-1 \quad (11.26)$$

The time intervals $t_{k+1} - t_k, k = 0, 1, \dots, N-1$ over which the control is considered constant in (11.26) are denoted by $T_k, k = 0, 1, \dots, N-1$. In Figure 11.2,

$$T_k = t_{k+1} - t_k = 0.5, \quad k = 0, 1, \dots, N-1 \quad (11.27)$$

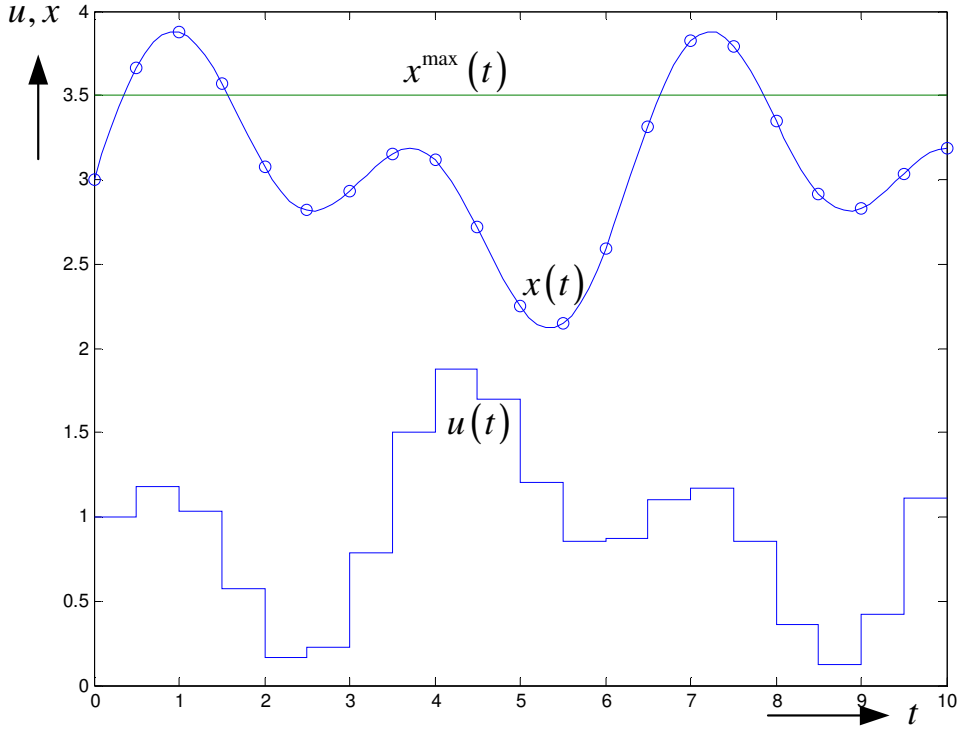


Figure 11.2: A form of control parameterization and checks of state constraints

Figure 11.2 reveals two things. Firstly control parameterization restricts the possible choices of the control $u(t), t_0 \leq t \leq t_f$. Secondly if N is large enough any $u(t), t_0 \leq t \leq t_f$ can be very well approximated.

Now having $u_k \in R^{n_u}, k = 0, 1, \dots, N-1$ we obtain $u(t), t_0 \leq t \leq t_f, u \in R^{n_u}$ and after numerical integration of (11.24) we obtain the associated costs (11.23) while automatically satisfying the system equation (11.24). Then the optimal control problem has become a function minimization problem:

$$\text{Find } u_k \in R^{n_u}, k = 0, 1, \dots, N-1 \text{ such that } J(u(t)) = J(u_k) \text{ is minimal} \quad (11.28)$$

Algorithms for function minimization are well developed.

Now suppose we have an optimal control problem with state inequality constraints. Note that the algorithms for solving optimal control problems we have presented so far are in general unable to deal with state inequality constraints. Suppose the state constraint is of the following form

$$x_i(t) \leq x_i^{\max}, i = 1, 2, \dots, n_x \quad (11.29)$$

where $x_i^{\max}, i = 1, 2, \dots, n_x$ are known bounds. Suppose furthermore that we evaluate these constraints (bounds) only at a limited number of time instants,

$$t_{k+1} \geq t_k, k = 0, 1, \dots, N-1 \quad (11.30)$$

where $t_N = t_f$. This results in the following finite number of inequality constraints,

$$x_i(t_k) \leq x_i^{\max}, i = 1, 2, \dots, n_x, k = 0, 1, \dots, N \quad (11.31)$$

Finally note that if the control is represented by the vectors $u_k \in R^{n_u}, k = 0, 1, \dots, N-1$ then $x_i(t_k), k = 0, 1, \dots, N$ are known functions of $u_k \in R^{n_u}, k = 0, 1, \dots, N-1$ which are actually computed during the numerical integration of the augmented system (11.22). So by numerically integrating the augmented system we are not only able to compute the costs as a function of $u_k \in R^{n_u}, k = 0, 1, \dots, N-1$ but we are also able to compute (evaluate) the state constraints (11.29) at a finite number of time instants as described by (11.30), (11.31). This is represented in figure 11.2 by the dots. In this way we have turned the optimal control problem into a function minimization problem with inequality constraints. It is clear that in the manner described above we can also evaluate state equality constraints as well as control equality and inequality constraints as well as terminal constraints.

Finally a free final time can be handled by time scaling. In this case we introduce a new time-variable,

$$\tau = \frac{t}{t_f}, 0 \leq \tau \leq 1 \quad (11.32)$$

From (11.32) we obtain,

$$\frac{d}{d\tau} = \frac{d}{d\left(\frac{t}{t_f}\right)} = t_f \frac{d}{dt} \quad (11.33)$$

Therefore the original system equation,

$$\begin{bmatrix} dx_1 / dt \\ dx_2 / dt \\ \vdots \\ dx_{n_x} / dt \\ dJ_r / dt \end{bmatrix} = \begin{bmatrix} f_1(x, u, d) \\ f_2(x, u, d) \\ \vdots \\ f_{n_x}(x, u, d) \\ L(x, u, d) \end{bmatrix}, 0 \leq t \leq t_f \quad (11.34)$$

changes into,

$$\begin{bmatrix} dx_1 / d\tau \\ dx_2 / d\tau \\ \vdots \\ dx_{n_x} / d\tau \\ dJ_r / d\tau \end{bmatrix} = \begin{bmatrix} t_f * f_1(x, u, d) \\ t_f * f_2(x, u, d) \\ \vdots \\ t_f * f_{n_x}(x, u, d) \\ t_f * L(x, u, d) \end{bmatrix}, 0 \leq \tau \leq 1 \quad (11.35)$$

and t_f is considered as an additional variable of the to be minimized cost function which now becomes,

$$J(u_k, t_f) \quad (11.36)$$

The fact that we have to multiply the right hand side of (11.34) by t_f to obtain (11.36) can also be seen from $x_1(t)$ in figure 11.2. If we scale the time axis from zero to one and if we do not change the plot of x_1 then the derivative of $x_1(\tau)$ at each point $0 \leq \tau \leq 1$ has become $t_f = 10$ times larger due to the fact that we have reduced the time axis with a factor $t_f = 10$.

The Matlab code to solve the optimal control problem of fishing anchovies and salmon by means of control parameterization is presented at the end of this section.

Summary

If we parameterize the control and when we evaluate the state and control equality/inequality constraints at a limited number of time points only we can transform an optimal control problem of any type into a function minimization problem with equality and inequality constraints. If the final time is free this also requires time-scaling.

Matlab code.

```
% Exslec11: Script 2 lecture 11, Optimal control with state constraints
% Set options vector: opt(1) : 1=display intermediate results
%           opt(13): number of equality constraints
%           opt(14): 500, maximum number of function evaluations
opt=[]; opt(1)=1; opt(14)=500;
uguess=5e-3*ones(2,10); % initial guess for control
if cont==1 % cont=1: continue with current values
    uguess=u;
end
lb=zeros(2,10); ub=5e-2*ones(2,10); % control lower and upper bounds
% Perform the constraint function minimization and return the solution x
u=constr('fsys11',uguess,opt,lb,ub);
% Obtain the optimal control and optimal state
[f,g,tt,xt]=fsys11(u);
% Plot the optimal state
figure(1); plot(tt,xt);
xlabel('Time [days]'); ylabel('states [tons, euros]'); title('Optimal x(t)');
% Plot the optimal control
figure(2); stair(0:365/10:9/10*365,u);
xlabel('Time [days]'); ylabel('control [per day]'); title('Optimal u(t)');
```

```
% fsys11: Optimal control by function minimization example lecture 11
function [f,g,tt,xt]=fsys11(u);
tt=[]; % vector to store time instants obtained from numerical integration
xt=[]; % matrix to store states obtained from numerical integration
xc=[]; % matrix to store states for evaluation of constraints
tk=0; K=10; x0=[4;2;0]; % initial time, number of time points, initial state
for k=1:K
    tk1=tk+365/10; % next time point
    [t,x]=ode45('anchsal',[tk tk1],x0,[],u(:,k)); % numerical integration
    tt=[tt; t]; xt=[xt; x]; % store time points and states
    tk=tk1; x0=x(end,:); % update time and the initial state
    xc=[xc x0]; % update state matrix for constraint evaluation
end;
f=x0(3); %Compute J
%g=[]; % no constraints or
g=[xc(1,:)-10]; %Compute/evaluate constraints: g(i,j)<=0 should hold
```

```
% Anchsal: Dynamic model and cost of the Anchovis Salmon system
function [dx]=anchsal(t,x,flag,u);
p=[5e-2; 2e-2; 2e-2; 1e-2; 1e-1; 1.5e-1; 1e-3; 2.5e-3];
dx(1,1)=p(1)*x(1)-p(2)*x(1)*x(2)-u(1)*x(1);
dx(2,1)=-p(3)*x(2)+p(4)*x(1)*x(2)-u(2)*x(2);
dx(3,1)=-p(5)*u(1)*x(1)-p(6)*u(2)*x(2)+p(7)*u(1)+p(8)*u(2);
```

11.3 Digital control systems and control parameterization

After reading this section the student is able to:

1. Explain the difference between continuous-time and digital control
2. Draw the block diagram of a conventionally sampled digital control system
3. Recognize that digital controls do not require control parameterization
4. Recall the definition of conventional sampling and the associated sampling period
5. Recall under which condition our continuous-time control system design is an accurate approximation when implemented in a digital computer
6. Recall the associated advantage of applying digital control theory

All our control computations so far have been made assuming the control changes continuously in time. Furthermore we have assumed that also measurements are performed continuously in time. Nowadays most control systems are digital control system where the controller is a digital computer. A digital computer however cannot control and measure continuously in time and is connected to the system through a digital to analog converter (D/A converter) and an analog to digital converter (A/D converter), see figure 11.3. The dots in the graphs represent $u(t_i)$, $y(t_i)$ the digital values of the control and output at the time instants t_i . Figure 11.4 is another representation of a digital control system. Finally figure 11.5 represents the task sequence of the digital controller (control computer).

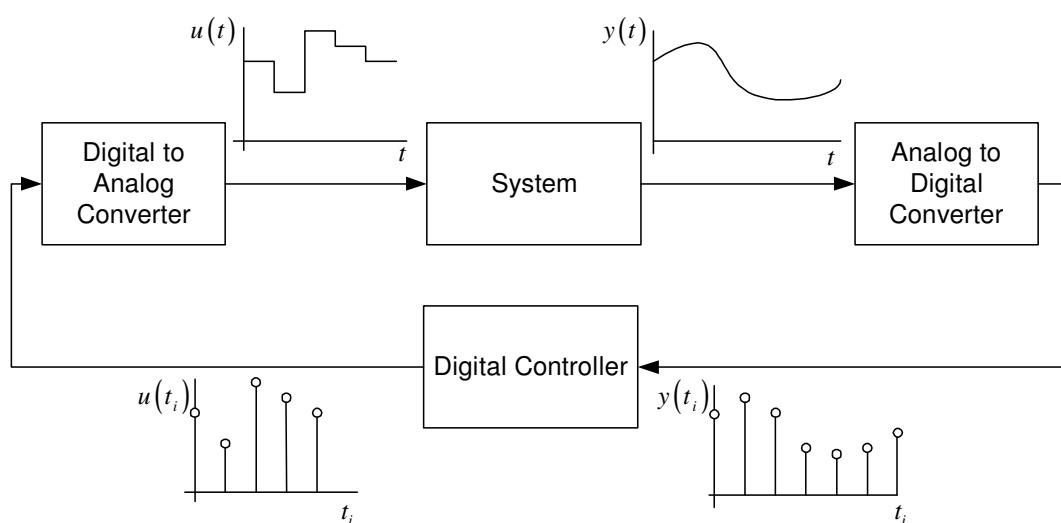


Figure 11.3: A digital control system

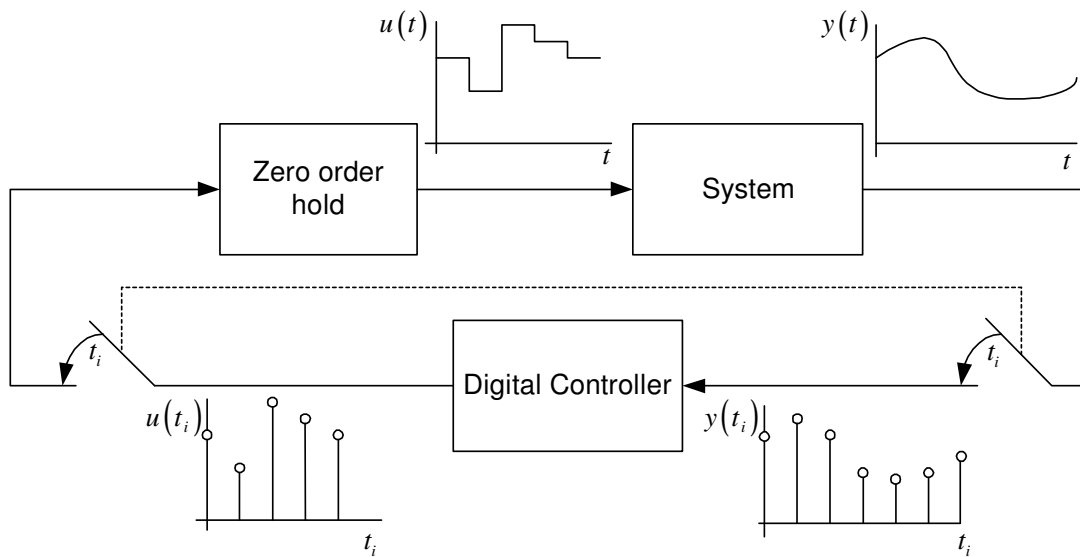


Figure 11.4: Another representation of a digital control system

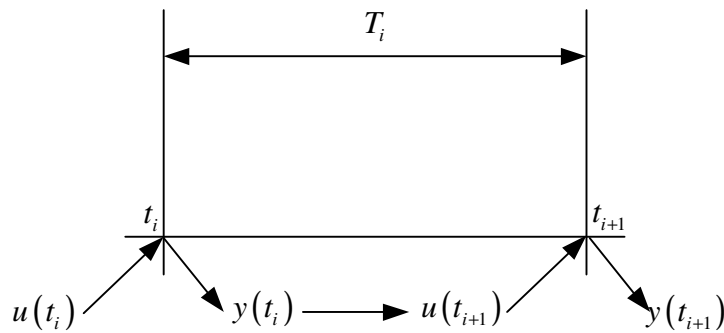


Figure 11.4: Task sequence of the digital controller

In figures 11.3-11.5 it is assumed that all control inputs and outputs of the system are updated and read by the digital computer simultaneously at the time instants

$$t_i, i = 0, 1, \dots \quad (11.37)$$

which are called the sampling instants of the digital control system. In figure 11.3 this is represented by the simultaneous closing of the two switches. At time t_i the computer sends the control $u(t_i)$ it has computed previously and reads the output $y(t_i)$. From $y(t_i)$ and possibly other data, within the sampling interval T_i , it has to compute $u(t_{i+1})$ which is supplied to the system at t_{i+1} . After having been updated the values of the control inputs are held constant by a so called zero-order hold (usually a digital to analog converter which converts the last received digital number into a constant voltage at its output until it receives a new number). This is mathematically described as follows,

$$u(t) = u(t_i), \quad t_i \leq t < t_{i+1}, \quad i = 0, 1, \dots \quad (11.38)$$

The control (11.38) is called a piecewise constant control which results in a staircase plot as you can see in figure 11.3. In general digital controllers use zero-order holds so they generate piecewise constant controls. Now note that this type of control corresponds precisely with the control parameterization (11.26) in the previous section if we replace k with i . So if the controller is digital and if we select

$$t_i = t_k \quad (11.39)$$

then what was called a control parameterization in the previous section is now precisely the true digital control! Clearly digital controls, when considered over a finite horizon, are fully described by finitely many numbers.

In practice and within digital control theory one often assumes that the sampling intervals

$$T_i = t_{i+1} - t_i, \quad i = 0, 1, \dots \quad (11.40)$$

are constant,

$$T_i = T, \quad i = 0, 1, \dots \quad (11.41)$$

Then T is called the sampling period or sampling interval and $1/T$ the sampling frequency. If all control variables and measurements are updated simultaneously and regularly, as described by (11.38), (11.40), (11.41), this is called conventional sampling (digital control theory and other types of sampling, which quite often occur in Wageningen, are and have been thoroughly researched by your lecturer). Now roughly speaking, if the sampling interval T is small compared to the time needed for the state variables to change significantly then a continuous control approximates a piecewise constant digital control quite well. But note that to keep T small the digital controller has to perform its computations fast enough. Applying digital control theory overcomes the demand to take the sampling time sufficiently small.

12. Control and state parameterization by Chebyshev polynomials. Actively and passively adaptive control systems.

After reading this chapter the student should be able to:

- 1) Explain why Chebyshev polynomials are especially suitable for control and state parameterization
- 2) Explain why it is advantageous to also parameterize the state
- 3) Explain the possible disadvantage of state parameterization
- 4) Substitute possible solutions into differential equations to obtain information on their validity and the associated initial state and controls
- 5) Explain the difference between an optimal control system and an adaptive control system
- 6) Explain the difference between an actively and passively adaptive control system

If the controller is a digital controller the piecewise constant control parameterization is not an approximation but a true description of the control. However, if we are interested in computing continuous-time optimal controls by function minimization piecewise constant control parameterizations are not very efficient. This means that we need a relatively large number of parameters to approximate the control. Parameterization of the control by Chebyshev polynomials is, in a certain sense, the most efficient way to parameterize the control. Since like the control the state is a time function, we can also parameterize the state. If we also parameterize the state we can transform the constraint that we have to satisfy the state equation $\dot{x} = f(x, u, d, t)$ into algebraic equality constraints. Doing so we no longer require numerical integration to compute the function values during the function minimization. Numerical integration is a very time consuming computation. The price we pay is that additional constraints have to be satisfied which usually however is (much) less time-consuming computationally.

12.1 Control and state parameterization by polynomials and other time functions

From Fourier analysis any time function $s(t)$, $0 \leq t \leq t_f$ can be approximated by adding a limited number of sine and cosine functions as follows,

$$s(t) \approx \sum_{n=0}^{N-1} a_n \cos(n\omega_f t) + b_n \sin(n\omega_f t), \quad \omega_f = \frac{2\pi}{t_f} \quad (12.1)$$

Equation (12.1) is called a Fourier series approximation of $s(t)$, $0 \leq t \leq t_f$. Everything in equation (12.1) on the right is known except for the number N and the values of the $2N$ numbers $a_0, a_1, \dots, a_{N-1}$ and $b_0, b_1, \dots, b_{N-1}$ called the Fourier coefficients. Fourier analysis presents formulas how to compute the Fourier coefficients $a_0, a_1, \dots, a_{N-1}$, $b_0, b_1, \dots, b_{N-1}$ from $s(t)$, $0 \leq t \leq t_f$. In the case of control and state parameterization these formulas are not important. What is important is to observe that by varying the numbers $a_0, a_1, \dots, a_{N-1}$, $b_0, b_1, \dots, b_{N-1}$ we can approximate any time function $s(t)$, $0 \leq t \leq t_f$. This means equation (12.1) is a parameterization of the time function $s(t)$, $0 \leq t \leq t_f$ by sine and cosine functions and the Fourier coefficients $a_0, a_1, \dots, a_{N-1}$, $b_0, b_1, \dots, b_{N-1}$ are the $2N$ parameters. It will not surprise you that from Fourier analysis it follows that the larger we select N the more accurate the approximation (12.1) will be. This is illustrated by figure 12.1 in which a block function is approximated using several different values for N .

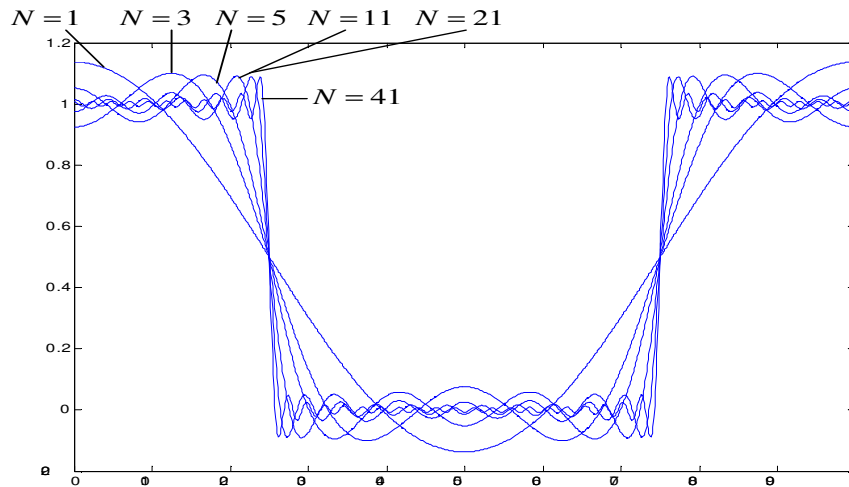


Figure 12.1: Approximation of a block function by sine and cosine functions

As to control parameterization we may raise the following interesting question. Suppose a time function $s(t)$, $0 \leq t \leq t_f$ is parameterized by N numbers. This parameterization we denote by,

$$p_N(t), 0 \leq t \leq t_f \quad (12.2)$$

Consider the largest error $e_{\max}$ between the time function $s(t)$ and its parameterization $p_N(t)$,

$$e_{\max} = \max_t (|s(t) - p_N(t)|) \quad (12.3)$$

Now which parameterization $p_N(t)$ has the smallest $e_{\max}$? The answer to this question is: a parameterization by Chebyshev polynomials. That is the reason why we select Chebyshev polynomials for control and state parameterization. So instead of using piecewise constant time functions or sine and cosine functions to parameterize time functions we will use Chebyshev polynomials. Figure 12.2 illustrates for several values of N that the approximation by Chebyshev polynomials is more accurate than the one obtained from the other two parameterizations.

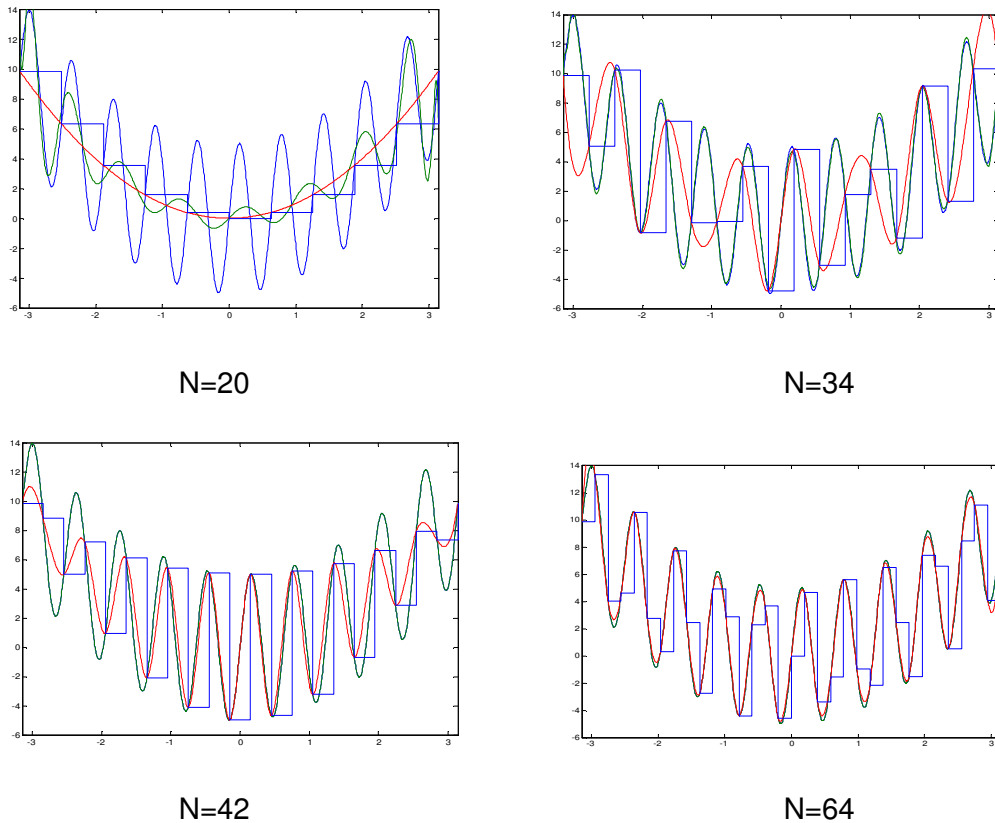


Figure 12.2: Chebyshev, Fourier and piecewise constant control parameterizations

12.2 Control and state parameterization by Chebyshev polynomials

A Chebyshev polynomial of order n is denoted by,

$$T_n(t), \quad n = 0, 1, 2, \dots \quad (12.4)$$

where the time t is restricted to the interval,

$$t \in [-1, 1] \quad (12.5)$$

Chebyshev polynomials can be defined recursively as follows,

$$T_0(t) = 1, \quad T_1(t) = t, \quad T_n(t) = 2tT_{n-1}(t) - T_{n-2}(t), \quad n = 2, 3, \dots \quad (12.6)$$

A plot of the Chebyshev polynomials from order 2 to 5 are presented in figure 12.3.

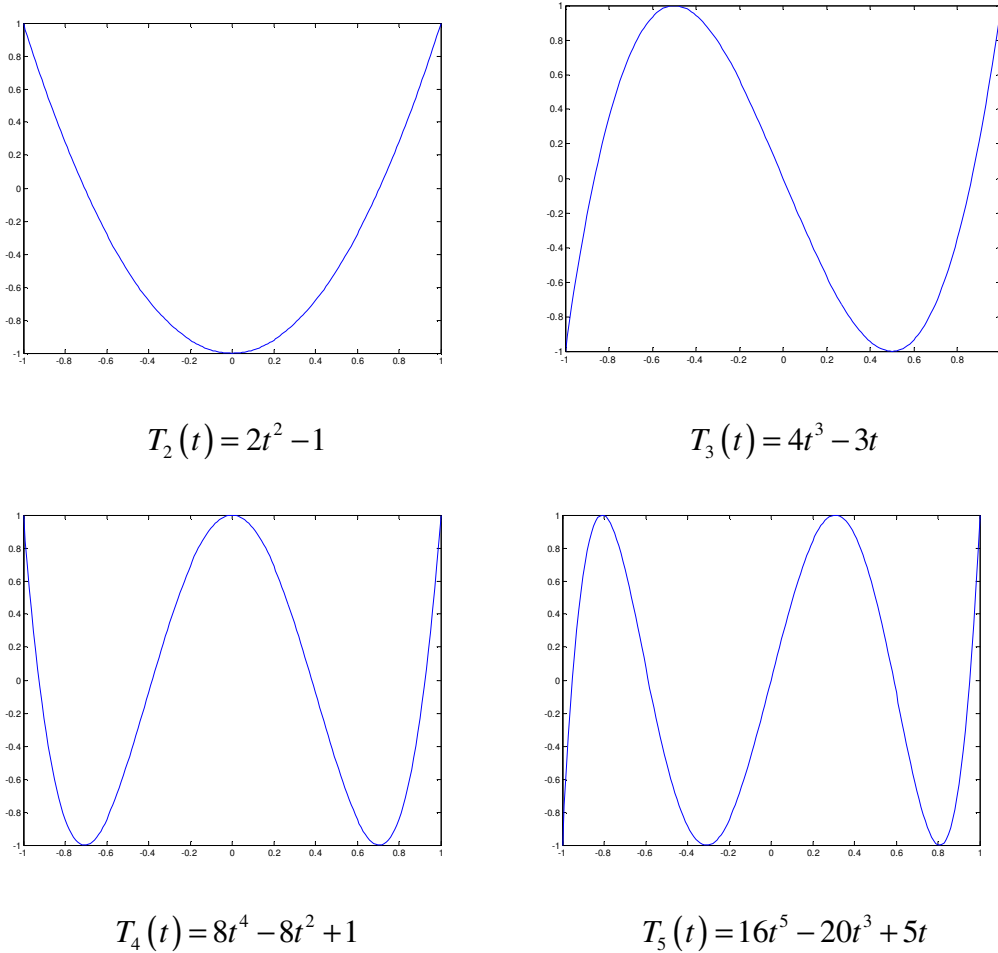


Figure 12.3: Chebyshev polynomials

To use the polynomials for control and state parameterization of optimal control problems the time interval $t \in [t_0, t_f]$ is shifted and scaled onto $t \in [-1, 1]$. Now similar to Fourier theory any time function $s(t)$, $-1 \leq t \leq 1$ can be approximated by adding a number of Chebyshev polynomials as follows,

$$s(t) \approx \frac{1}{2}a_0T_0(t) + \sum_{n=1}^N a_nT_n(t) \quad (12.7)$$

So equation (12.7) is a parameterization of the function $s(t)$, $-1 \leq t \leq 1$ by Chebyshev polynomials and $a_0, a_1, \dots, a_{N-1}$ are the N parameters called Chebyshev coefficients. Therefore,

$$p_N(t) = \frac{1}{2}a_0T_0(t) + \sum_{n=1}^N a_nT_n(t) \quad (12.8)$$

is called a Chebyshev series approximation of $s(t)$, $-1 \leq t \leq 1$. The following two properties of Chebyshev series will be especially useful to solve optimal control problems by means of control and state parameterization,

$$\int_{-1}^1 p_N(t) dt = a_0 - \sum_{n=1}^N \frac{1+(-1)^n}{n^2-1} a_n \quad (12.9)$$

In the case of state parameterization equation (12.9) is important for the computation of the final state of the system and the running costs which can be observed from the augmented system equation (11.22) on page 51. Due to equation (12.9) we do not need numerical integration to compute these but instead calculate algebraic expressions in the Chebyshev coefficients.

Let a'_n , $n=0,1,\dots,N$ denote the parameters of the Chebyshev series approximation of,

$$\dot{p}_N(t) = \frac{dp_N}{dt}, \quad -1 \leq t \leq 1 \quad (12.10)$$

Then the following property of Chebyshev series applies,

$$a'_{n-1} - a'_{n+1} - 2na_n = 0, \quad n=1,2,\dots,N \quad (12.11)$$

Equation (12.11) is important because it relates Chebyshev series approximations of time functions to Chebyshev series approximations of the time derivatives of these time functions. Actually equation (12.11) is the one that enables us to replace the

differential equation constraint $\dot{x} = f(x, u, d, t)$ by equality constraints of the type (12.11) on the Chebyshev coefficients of the Chebyshev series approximations of $x(t)$, $u(t)$, $d(t)$. Thereby we avoid the need for numerical integration which is time consuming. Using higher order Chebyshev approximations by increasing N improves the accuracy at the expense of computation time since the number of equality constraints described by (12.11), that have to be satisfied, increase with N .

12.3 Substitution of possible solutions into differential equations

Suppose we have a dynamic system described by the following state equation in state space form,

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_2 + u_1 x_4 \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= -x_4 + u_2 x_2\end{aligned}\tag{12.12}$$

Suppose we want $x_1(t)$ to follow as closely as possible $\cos(t)$, $0 \leq t \leq t_f$. Suppose furthermore that we want $x_3(t)$ to follow as closely as possible $\sin(t)$, $0 \leq t \leq t_f$. These control objectives are implemented by the following cost function,

$$J(u(t)) = \int_0^{t_f} \left[(x_1(t) - \cos(t))^2 + (x_3(t) - \sin(t))^2 \right] dt\tag{12.13}$$

Now we may wonder whether it is possible to find a control such that this cost function is zero. Or, in other words, whether there is a control such that,

$$x_1(t) = \cos(t), \quad x_3(t) = \sin(t)\tag{12.14}$$

First note that equation (12.14) implies,

$$\dot{x}_1(t) = -\sin(t), \quad \dot{x}_3(t) = \cos(t)\tag{12.15}$$

An interesting and simple way to find the answer to our question is by substitution of equations (12.14) and (12.15) in the differential equations (12.12),

$$\begin{aligned}
-\sin(t) &= x_2 \\
-\cos(t) &= \sin(t) + u_1 \cos(t) \\
\cos(t) &= x_4 \\
-\sin(t) &= -\cos(t) + u_2 \sin(t)
\end{aligned} \tag{12.16}$$

From equation (12.16) we obtain,

$$\begin{aligned}
u_1(t) &= \frac{-\cos(t) - \sin(t)}{\cos(t)} = -1 - \frac{\sin(t)}{\cos(t)} \\
u_2(t) &= \frac{-\sin(t) + \cos(t)}{\sin(t)} = -1 + \frac{\cos(t)}{\sin(t)}
\end{aligned} \tag{12.17}$$

Finally note that equations (12.14), (12.15) demand that,

$$x_1(0) = 1, x_2(0) = 0, x_3(0) = 0, x_4(0) = 1 \tag{12.18}$$

So only if,

$$x(0) = [1 \ 0 \ 0 \ 1]^T \tag{12.19}$$

equations (12.14), (12.15) are satisfied when the control is given by (12.17). In other words only if $x(0) = [1 \ 0 \ 0 \ 1]^T$ the optimal control is given by equation (12.17) and the optimal minimal costs are zero.

If we approximate the control and state by a Chebyshev series and we substitute these polynomials in the differential equations we obtain precisely the equality constraints in terms of the Chebyshev coefficients mentioned in the previous section.

12.4 Interpretation and computation of co-states

When using polynomial methods for optimal control computations co-states $\lambda^*(t), t_0 \leq t \leq t_f$ are not computed since they are not needed to obtain the solution, as opposed to gradient algorithms. The optimal co-state $\lambda^*(t), t_0 \leq t \leq t_f$ has an important interpretation however,

$$-\lambda^{*T}(t) = \frac{\partial J^*(t, t_f)}{\partial x^*(t)} \Leftrightarrow -\lambda_i^*(t) = \frac{\partial J^*(t, t_f)}{\partial x_i^*(t)}, \quad i = 1, 2, \dots, n_x, \quad t_0 \leq t \leq t_f \tag{12.20}$$

In equation (12.20) $J^*(t, t_f)$ represents the minimum costs made from time t to t_f when the state at time t is $x^*(t)$. Therefore at each time $t_0 \leq t \leq t_f$, $-\lambda_i^*(t)$ equals the sensitivity of the minimum costs $J^*(t, t_f)$ to a change in the state variable $x_i^*(t)$. In other words $-\lambda_i^*(t)$ represents the *marginal costs* of the corresponding state variable $x_i^*(t)$, $t_0 \leq t \leq t_f$ while $\lambda_i^*(t)$ represents the *marginal value* of the corresponding state variable $x_i^*(t)$. This interpretation may be important for analysis, control or design. In that case optimal controls computed from polynomial methods may be submitted as “initial guesses” to gradient algorithms that need to iterate only once to obtain the corresponding co-state of the problem. A highly interesting further economic interpretation of optimal control can be found in Dorfman (1969).

12.5 When models are of insufficient quality: Passively and actively adaptive control

The optimal control systems presented in figures 7.5, 7.6 and 7.7 are approximately optimal only if the model is of sufficient quality such that the errors are kept small by the optimal LQ state feedback or the optimal LQG dynamic output feedback. Therefore an important question is what to do if the quality of the systems model is insufficient. A very simple yet crucial answer to this question is: improve the model!

There are basically two ways to improve the model. One is to improve the model off-line by gathering additional knowledge based on first principles (scientific knowledge). If such knowledge is lacking this may call for (scientific) experimentation and research. The other is to use measurement data to improve the model. This may be either done off-line or on-line. In our optimal control system measurements are only used to generate control corrections, they are not used to improve the model. In optimal adaptive control systems they are also used to improve the model on-line. A block diagram of an adaptive control systems is presented in figure 12.4. Compared to our original optimal control system the number of on-line computations to be performed by an adaptive control system increases dramatically. This is because every time the model is adapted the optimal control must be recomputed on-line. In our original optimal control system the computation of the optimal control could be performed off-line.

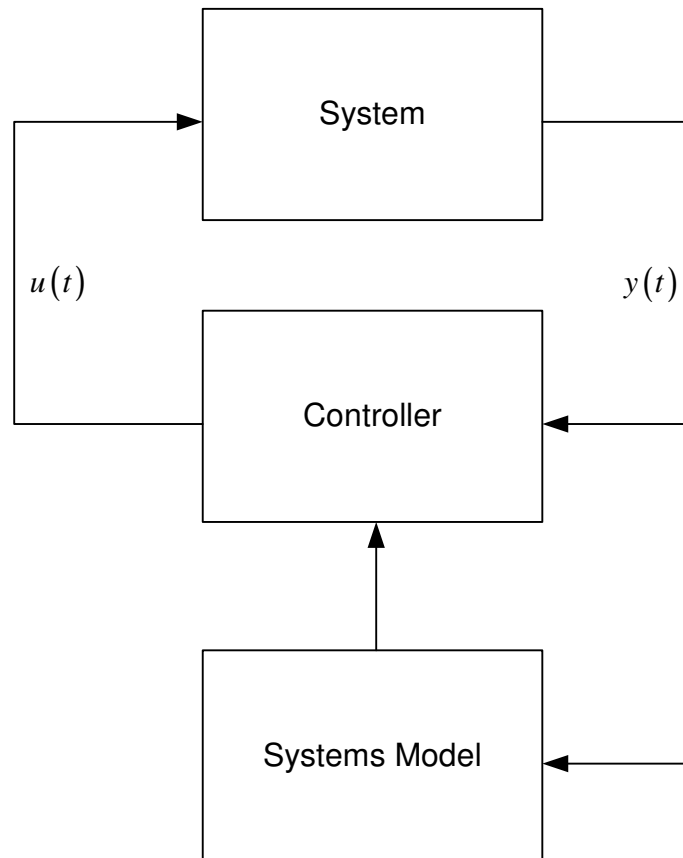


Figure 12.4: An adaptive control system

Compared to an optimal control system the additional objective of an optimal adaptive control system is to improve the quality of the model. The quality improvement of the model, such as the improvement of model parameters, depends on the control of the system. So apart from considering the original control objectives it is wise to also include into the cost function the objective to improve the model. If this is done the adaptive control system is called an actively adaptive optimal control system, because the optimal control is selected to also promote the improvement of the model. If only the original control objectives are described by the cost function then the control system is called a passively adaptive optimal control system. Optimal active adaptive control system design is an important new area of research within the systems and control group. Be welcome to join us!

References

1. The Role and Use of the Stochastic Linear-Quadratic-Gaussian Problem in Control System Design, Michael Athans, IEEE Transaction on Automatic Control, 1971, Vol. AC-16, no 6., pp. 529-552.
2. Optimal Control, 1986, F.L. Lewis, Wiley.

3. Dynamic Optimization, A.E. Bryson Jr., 1999, Addison and Wesley.
4. Towards a Theory of Difficulty of Computation in Optimal Control, R.E. Kalman, Proceedings 1964 IBM Symposium on Computation and Control.
5. J. Vlassenbroeck, R. van Dooren, A Chebyshev technique for solving Nonlinear optimal control problems, IEEE Transaction on Automatic Control, 1988, Vol. 33, no 4, pp. 333-340.
6. Y. Bar-Shalom, 1981, Stochastic dynamic programming: Caution and probing, IEEE Transactions on Automatic Control, AC-26, no. 5, 1184-1195
7. L.G. van Willigenburg, W.L. De Koning, 2006, On the synthesis of time-varying LQG weights and noises along optimal control and state trajectories, Optimal Control Applications and Methods, 27, 137-160
8. L.G. Van Willigenburg, W.L. De Koning, Z.S. Chalabi, M. Tchamitchian, G. Van Straten, 2004, On the selection of appropriate control system design methodologies, Proceedings of the IFAC World Congress, 3-8 July, 2005, Prague, Tsjechië, 04296.pdf
9. R. Dorfman, 1969, "An Economic Interpretation of Optimal Control Theory", The American Economic Review, 817-831.

Systems and Control Theory Part 2: Optimal Control

**Lecture 7.
Optimal control and optimal feedback of non-
linear systems**

L.G. van Willigenburg

**Wageningen University
Mathematical & Statistical Methods Group
(Biometris)**





Material

1. Lecture notes
2. Sheets of the lectures
3. Computer assignments
4. Software

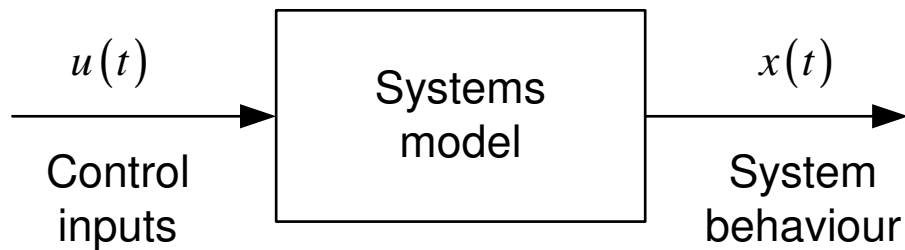
To be downloaded from blackboard

Contents

1. Optimal control problems
2. Optimal control and optimal feedback
3. Optimal LQ and LQG feedback
4. Summary
5. Exercise
6. Requirements for the test



1. Optimal control problems



Control problem: Find the control $u(t)$ such that the system behaviour meets our requirements.

Translate the requirements into a cost function.

Optimal control problem: Find the control $u(t)$ that minimizes the cost function.

More specific:

Given the system model

in state-space form $\dot{x} = f(x, u, d, p, t)$,

the initial state $x(t_0) = x_0$,

and the external inputs $d(t), t_0 \leq t \leq t_f$,

find the control $u(t), t_0 \leq t \leq t_f$

that minimizes the cost function.



Example: A simple control problem without a cost function.

Consider the Anchovis / Salmon system from sheet 1-16 written in state-space form:

$$\frac{dx_1}{dt} = p_1 x_1 - p_2 x_1 x_2 - u_1 x_1$$

$$\frac{dx_2}{dt} = -p_3 x_2 + p_4 x_1 x_2 - u_2 x_2$$

$$p_1 = 5, p_2 = 2, p_3 = 2, p_4 = 1$$

a) Suppose $x(t_0) = x_0 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$. Compute the control $u(t)$ that keeps the system at the initial state in other words:
$$x(t) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}, \quad t \geq t_0.$$

b) Suppose the model is exact and we apply the control computed under a) to the system. In reality $x(t_0) = \begin{bmatrix} 4 \\ 2 \end{bmatrix} + \Delta x(t_0)$ where $\Delta x(t_0)$ is a vector with small perturbations (deviations). Does $x(t) \rightarrow \begin{bmatrix} 4 \\ 2 \end{bmatrix}$ as $t \rightarrow \infty$ for any small $\Delta x(t_0)$? Answer this question by making a computation.

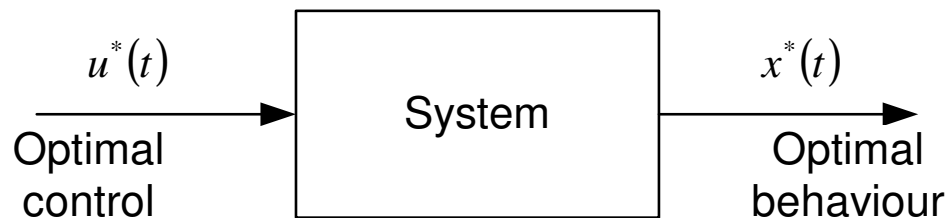


2. Optimal control and optimal feedback

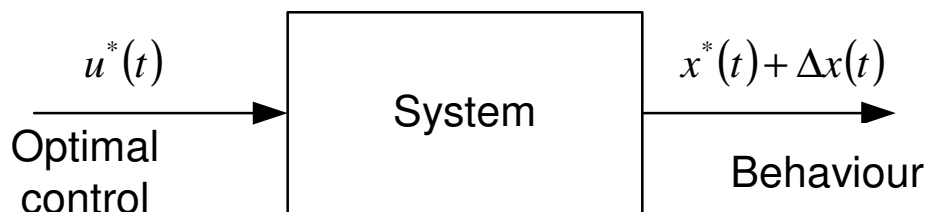
Solution of optimal control problem:

$$u^*(t), x^*(t), t_0 \leq t \leq t_f$$

If the model, initial state and external inputs are exact:



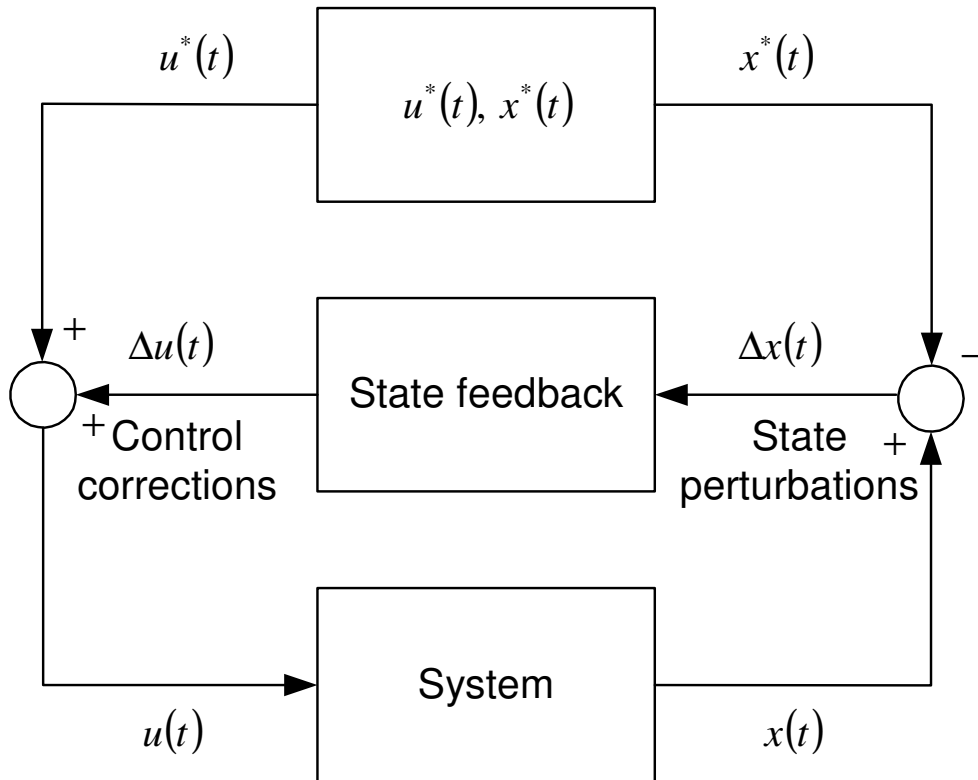
If the model, the initial state and/or the external inputs are not exact:



Problem: how to limit $\Delta x(t)$? **Solution:** by feedback.



State Feedback:



$$\Delta x(t) = x(t) - x^*(t), \Delta u(t) = u(t) - u^*(t)$$

Example:

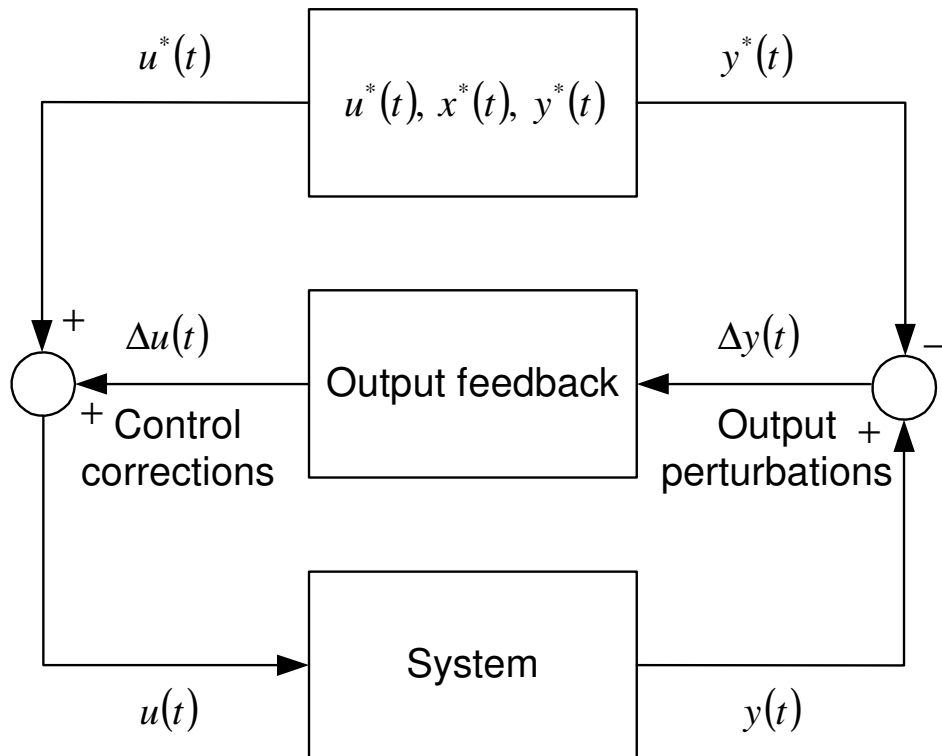
Static state feedback : $\Delta u(t) = -L(t)\Delta x(t)$

State feedback is usually static meaning that there is an algebraic relation between $\Delta u(t)$ and $\Delta x(t)$.

Problem: The complete state must be measured. In practice this is often impossible or too expensive.

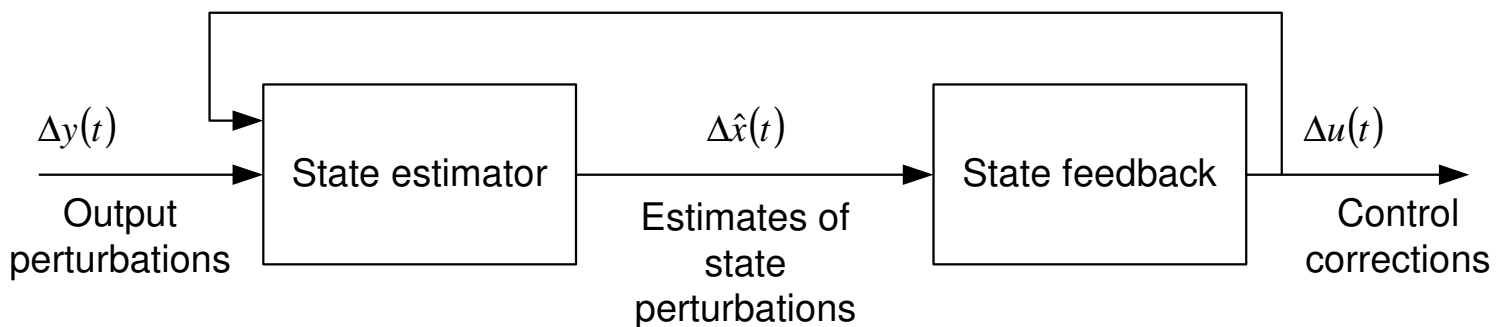


Output feedback:



$$\Delta y(t) = y(t) - y^*(t), \Delta u(t) = u(t) - u^*(t)$$

Example: Dynamic output feedback



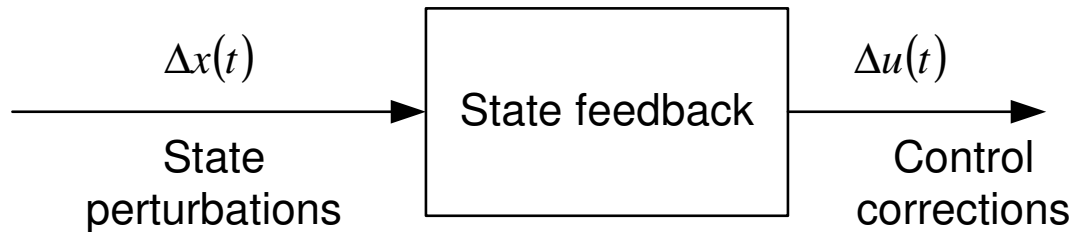
The state estimator is a dynamic system. Therefore this type of output feedback is called dynamic.

Remark: To limit perturbations state feedback is usually more successful than output feedback.



3. Optimal LQ and LQG feedback

Optimal Static State feedback: LQ state feedback

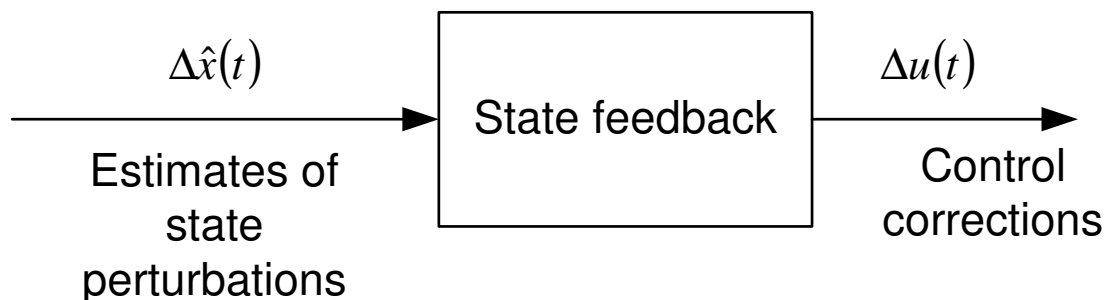


$$\Delta u(t) = -L(t)\Delta x(t).$$

$L(t)$: Feedback gain computed from an optimal Linear Quadratic (LQ) control problem in which the system model is linear and the cost function quadratic.

Optimal dynamic output feedback: LQG output feedback

The state feedback part is identical to LQ state feedback with the state $x(t)$ replaced by its estimate $\Delta \hat{x}(t)$.

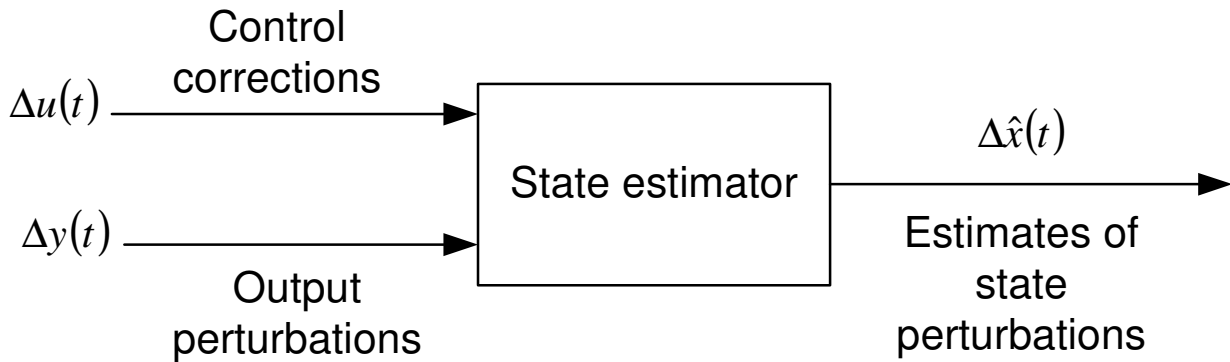


$$\Delta u(t) = -L(t)\Delta \hat{x}(t).$$

$L(t)$ computed from the LQ problem.



State estimator: The Kalman filter.



$$\dot{\Delta \hat{x}}(t) = A(t)\Delta \hat{x}(t) + B(t)\Delta u(t) + K(t)(\Delta y(t) - C(t)\Delta \hat{x}(t))$$

$$\Delta \hat{x}(t_0) = 0$$

$A(t), B(t), C(t)$: Linearized system model about the optimal solution $u^*(t), x^*(t)$

$K(t)$: Kalman gain computed from an optimal Linear Gaussian (LG) estimation problem.

The optimal Linear Gaussian estimation problem and the optimal Linear Quadratic control problem are dual problems. This means that the problems are mathematically identical.



4. Summary

To design and compute the optimal control and optimal feedback we have to determine and solve:

- 1) An optimal control problem involving the non-linear system and a cost function reflecting our requirements.
- 2) A linear quadratic optimal control problem to compute the optimal LQ state feedback.
- 3) In the case of optimal LQG output feedback another linear quadratic optimal control problem which is the dual version of the linear Gaussian estimation problem.



5. Exercise: A simple control problem without a cost function (a modification of the example on sheet 7-4)

Consider an Anchovis / Salmon system where the terms involving u_1, u_2 have been modified (changed)

$$\frac{dx_1}{dt} = p_1 x_1 - p_2 x_1 x_2 - u_1 x_1 x_1$$

$$\frac{dx_2}{dt} = -p_3 x_2 + p_4 x_1 x_2 - u_2 x_2 x_2$$

$$p_1 = 5, p_2 = 2, p_3 = 2, p_4 = 1$$

a) Suppose $x(t_0) = x_0 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$. Compute the control $u^*(t)$ that

keeps the system at the initial state in other words:

$$x^*(t) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}, \quad t \geq t_0.$$

b) Suppose the model is exact and we apply the control computed under a) to the system. In reality

$$x(t_0) = \begin{bmatrix} 4 \\ 2 \end{bmatrix} + \Delta x(t_0) \quad \text{where } \Delta x(t_0) \text{ is a vector with small}$$

perturbations (deviations). By making a computation

$$\text{show that } x(t) \rightarrow \begin{bmatrix} 4 \\ 2 \end{bmatrix} \text{ as } t \rightarrow \infty \text{ for any small } \Delta x(t_0).$$

c) Although $x(t) \rightarrow \begin{bmatrix} 4 \\ 2 \end{bmatrix}$ as $t \rightarrow \infty$ there may still be reasons

to apply feedback in practice. Mention as many different reasons as you can think of.



6. Requirements for the test

Within the test you have to,

- 1) Specify a general optimal control problem without constraints. See sheet 7.3.
- 2) Draw the block diagram of an optimal control system with static state feedback and with dynamic output feedback, including all the symbols and their meaning. See sheet 7-6, 7-7, figure 7.5, 7.6, page 4.
- 3) Draw the sub-block diagram of static state feedback including all the symbols, their meaning and their relation. See sheet 7-8.
- 4) Draw the sub-block diagram of dynamic output feedback including all the symbols, their meaning and their relation. See sheet 7-8, 7-9, figure 10.2, page 42. Know how we obtain the matrices $A(t)$, $B(t)$, $C(t)$ and why these are time-varying.
- 5) Perform a comparable exercise as the one on sheet 7-11.

Systems and Control Theory Part 2: Optimal Control

Skills required to apply optimal control

1. Modelling dynamic systems, transformation of dynamic models into state-space form required for optimal control
2. Formulation of quantitative control objectives by means of the performance measure (cost function)
3. Programming skills (Matlab/Tomlab in this course) to apply standard software for optimal control.
4. Interpretation of optimal control and state behaviour to possibly improve 1. or 2.

Lecture 8.

Formulation and classification of optimal control problems.

L.G. van Willigenburg



**Wageningen University
Mathematical & Statistical Methods Group
(Biometris)**

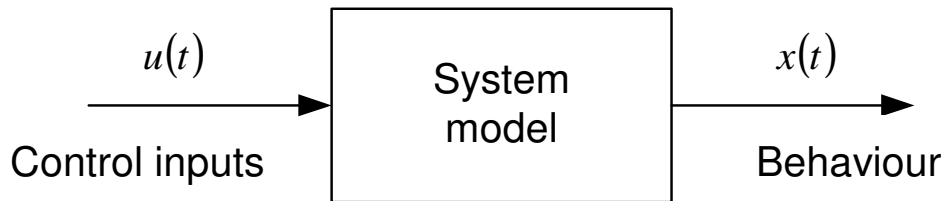


Contents

1. Recall optimal control problems
2. Examples of performance measures
3. A quadratic performance measure
4. The general form of a performance measure
5. Control constraints
6. State constraints
7. Terminal constraints
8. Free or fixed final time
9. Summary
10. Exercise
11. Requirements for the test



1. Recall optimal control problems



- Control problem: Find the control $u(t)$ such that the system behaviour meets our requirements.
- Translate requirements into a performance measure^{*}.
- Optimal control problem: Given the system model in state-space form, the initial state and the external inputs find the control that minimizes the performance measure.
- More specific:
Given the system model
in state-space form $\dot{x} = f(x, u, d, p, t)$,
the initial state $x(t_0) = x_0$,
and the external inputs $d(t), t_0 \leq t \leq t_f$,
find the control $u(t), t_0 \leq t \leq t_f$
that minimizes the performance measure.

*** Alternative names for the performance measure**
Objective function, Cost function, Cost criterion,
Criterion, Penalty function.



2. Examples of performance measures

- Requirement compulsory exercise 7:

keep the state at $\begin{bmatrix} 4 \\ 2 \end{bmatrix}$ in other words

$$x^*(t) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}, t_0 \leq t \leq t_f$$

- Suppose $x(t_0) = \begin{bmatrix} 5 \\ 1 \end{bmatrix}$. Then $x^*(t) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}, t_0 \leq t \leq t_f$ cannot be achieved (Why?).

- What we can achieve:

Keep the difference between $x(t)$ and $\begin{bmatrix} 4 \\ 2 \end{bmatrix}$ as small as possible over $t_0 \leq t \leq t_f$.

- Which cost function $J(u(t))$, when it is minimised, achieves this goal?

- $$J(u(t)) = \int_{t_0}^{t_f} |x_1(t) - 4| + |x_2(t) - 2| dt$$

- Why does $J(u(t)) = \int_{t_0}^{t_f} x_1(t) - 4 + x_2(t) - 2 dt$ not work?



- Suppose that now we want the state to follow, as closely as possible, the following so called state reference vector function:

$$x^{ref}(t) = \begin{bmatrix} x_1^{ref}(t) \\ x_2^{ref}(t) \end{bmatrix} = \begin{bmatrix} 4 + \sin(t) \\ 2 + \cos(t) \end{bmatrix}$$

- How should we select the cost function?

- $$J(u(t)) = \int_{t_0}^{t_f} |x_1(t) - x_1^{ref}(t)| + |x_2(t) - x_2^{ref}(t)| dt$$

- What is the difference between the previous and the following cost function:

$$\begin{aligned} J(u(t)) &= \int_{t_0}^{t_f} \left(x_1(t) - x_1^{ref}(t) \right)^2 + \left(x_2(t) - x_2^{ref}(t) \right)^2 dt \\ &= \int_{t_0}^{t_f} \left(x(t) - x^{ref}(t) \right)^T \left(x(t) - x^{ref}(t) \right) dt \end{aligned}$$

- A quadratic cost function is generally preferred because:
 - 1) It penalizes large errors more strongly
 - 2) It is more easily minimized computationally



- Suppose we want to bring the state from $x(t_0) = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$ as closely as possible to $x(t_f) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$ while we do not care about how this is done. Determine an appropriate cost function.

- $$J(u(t)) = \left(x_1(t_f) - 4 \right)^2 + \left(x_2(t_f) - 2 \right)^2 = \left(x(t_f) - \begin{bmatrix} 4 \\ 2 \end{bmatrix} \right)^T \left(x(t_f) - \begin{bmatrix} 4 \\ 2 \end{bmatrix} \right)$$

- We may also select:

$$J(u(t)) = F_{1,1} \left(x_1(t_f) - 4 \right)^2 + F_{2,2} \left(x_2(t_f) - 2 \right)^2$$

with

$F_{1,1} > 0$, $F_{2,2} > 0$ being weighing factors.

Take $F = \begin{bmatrix} F_{1,1} & 0 \\ 0 & F_{2,2} \end{bmatrix} > 0$ then

$$J(u(t)) = \left(x(t_f) - \begin{bmatrix} 4 \\ 2 \end{bmatrix} \right)^T F \left(x(t_f) - \begin{bmatrix} 4 \\ 2 \end{bmatrix} \right)$$



3. A quadratic performance measure

- Let $F = \begin{bmatrix} F_{1,1} & 0 \\ 0 & F_{2,2} \end{bmatrix}$, $Q = \begin{bmatrix} Q_{1,1} & 0 \\ 0 & Q_{2,2} \end{bmatrix}$, $R = \begin{bmatrix} R_{1,1} & 0 \\ 0 & R_{2,2} \end{bmatrix}$
 $F_{1,1}, F_{2,2}, Q_{1,1}, Q_{2,2}, R_{1,1}, R_{2,2} \geq 0$ and consider the cost function

$$J(u(t)) = x^T(t_f) F x(t_f) + \int_{t_0}^{t_f} x^T(t) Q x(t) + u^T(t) R u(t) dt$$

Explain in words what this criterion tries to do with the state $x(t)$ and the control $u(t)$ and how this is influenced by the choices of $F_{1,1}, F_{2,2}, Q_{1,1}, Q_{2,2}, R_{1,1}, R_{2,2}$.

4. The general form of a performance measure

$$J(u(t)) = \phi\left(x(t_f), d(t_f), t_f\right) + \int_{t_0}^{t_f} L(x, u, d, t) dt$$

$\phi \in R^1, L \in R^1$ are scalar functions (meaning that their outcome is a scalar)

- What is ϕ and what is L in the case of a quadratic cost function?
- $\phi\left(x(t_f), d(t_f), t_f\right)$ are called the terminal costs and $\int_{t_0}^{t_f} L(x, u, d, t) dt$ are called the running costs. Why?



Example:

Consider the Anchovis / Salmon system from sheet 1-16 written in state-space form:

$$\frac{dx_1}{dt} = p_1 x_1 - p_2 x_1 x_2 - u_1 x_1$$

$$\frac{dx_2}{dt} = -p_3 x_2 + p_4 x_1 x_2 - u_2 x_2$$

$$p_1 = 5, p_2 = 2, p_3 = 2, p_4 = 1$$

The last terms $u_1 x_1$ and $u_2 x_2$ in each differential equation represents tons of Anchovis and Salmon that is fished per day respectively.

Suppose 1 ton of Anchovis (x_1) is worth 10000 Euro.

Suppose 1 ton of Salmon (x_2) is worth 15000 Euro.

Suppose 1 unit of fishing Anchovis (u_1) costs 1000 Euro

Suppose 1 unit of fishing Salmon (u_2) costs 2500 Euro

- a) If you are going to fish both Anchovis and Salmon for one year and you want maximum profit how should you select the cost function? Or, in other words, how should you select the scalar functions $\phi\left(x(t_f)\right)$ and $L(x, u)$ and the initial and final time t_0, t_f ? Hint: first determine the units of the variables.
- b) How can you modify the cost function if, after the one year that you fished, you want to leave over some Anchovis and Salmon to be able to fish next year as well? With this modification, what will happen to the profit?



5. Control constraints

- In the prey-predator model the controls $u_1(t), u_2(t)$ cannot be negative:

$$u_1(t) \geq 0, u_2(t) \geq 0, t_0 \leq t \leq t_f$$

- In practice very often all control variables have both an upper and lower bound:

$$u_i^{\min} \leq u_i(t) \leq u_i^{\max}, t_0 \leq t \leq t_f, i=1,2,\dots,n_u$$

where $u_i^{\min}, u_i^{\max}, i=1,2,\dots,n_u$ are a-priori known bounds

- **Definition**

If the choice of the control variables $u_i, i=1,2,\dots,n_u$ is in any way restricted the optimal control problem is said to have control constraints.

- There are two types of control constraints:
 - 1) Control inequality constraints
 - 2) Control equality constraints

Example of 2) :

$$u \in R^2, \|u(t)\| = \sqrt{u_1^2(t) + u_2^2(t)} = 1, t_0 \leq t \leq t_f$$



6. State constraints

- In practice one sometimes wants certain state variables (such as the temperature in a furnace) not to exceed certain bounds:

$$x_i(t) \leq x_i^{\max}, t_0 \leq t \leq t_f \text{ for one or several } 1 \leq i \leq n_x \text{ and/or}$$

$$x_i(t) \geq x_i^{\min}, t_0 \leq t \leq t_f \text{ for one or several } 1 \leq i \leq n_x \text{ and/or}$$

$$x_i^{\min} \leq x_i(t) \leq x_i^{\max}, t_0 \leq t \leq t_f \text{ for one or several } 1 \leq i \leq n_x.$$

where $x_i^{\min}, x_i^{\max}$ are a-priori known bounds.

- **Definition**

If the values of the state variables are in any way restricted the optimal control problem is said to have state constraints.

- There are two types of state constraints:
 - 1) State inequality constraints
 - 2) State equality constraints

Example of 2) :

$$x \in R^2, \|x(t)\| = \sqrt{x_1^2(t) + x_2^2(t)} = 1, t_0 \leq t \leq t_f$$



7. Terminal state constraints

- Sometimes one wants $x(t_f)$, the terminal (final) state, to satisfy,

$$x_i(t_f) \leq x_i^{\max}(t_f) \text{ for one or several } 1 \leq i \leq n_x \text{ and/or}$$

$$x_i(t_f) \geq x_i^{\min}(t_f) \text{ for one or several } 1 \leq i \leq n_x \text{ and/or}$$

$$x_i^{\min}(t_f) \leq x_i(t_f) \leq x_i^{\max}(t_f) \text{ for one or several } 1 \leq i \leq n_x.$$

where $x_i^{\min}(t_f)$, $x_i^{\max}(t_f)$ are known bounds.

- **Definition**

If the values of the terminal (final) state variables are in any way restricted the optimal control problem is said to have terminal state constraints.

- There are two types of terminal state constraints:
 - 1) Terminal inequality constraints
 - 2) Terminal equality constraints

2) is always described by $\psi(x(t_f)) = 0, \psi \in R^{n_\psi}$

Example: $x(t_f) = \begin{bmatrix} 4 \\ 2 \end{bmatrix} \Leftrightarrow \psi(x(t_f)) = x(t_f) - \begin{bmatrix} 4 \\ 2 \end{bmatrix} = 0$



8. Free or fixed final time

- So far the final time of an optimal control problem was fixed a-priori. Sometimes, together with the control, one wants to optimize the final time, instead of fixing it.
- **Definition**
If the final time is not a-priori fixed but optimized the optimal control problem is said to be one with free final time.
- Example: Time-optimal control problems where the objective is to control the system as quickly as possible from the initial state $x(t_0)$ to a prescribed terminal state $x(t_f)$ in the presence of control constraints.
- Optimal control problems without free final time are optimal control problems with fixed final time.



9. Summary

- Optimal control problems can be classified as follows:
 - 1) With fixed or free final time.
 - 2) With or without control (equality/inequality) constraints.
 - 3) With or without state (equality/inequality) constraints.
 - 4) With or without terminal (equality/inequality) constraints.
- The general form of the cost function reads

$$J(u(t)) = \phi\left(x(t_f), d(t_f), t_f\right) + \int_{t_0}^{t_f} L(x, u, d, t) dt$$

$$\phi\left(x(t_f), d(t_f), t_f\right): \text{terminal costs,}$$

$$\int_{t_0}^{t_f} L(x, u, d, t) dt: \text{running costs.}$$

- The quadratic cost function for LQ and LQG feedback design reads:

$$J(u(t)) = \Delta x^T(t_f) F \Delta x(t_f) + \int_{t_0}^{t_f} \Delta x^T(t) Q \Delta x(t) + \Delta u^T(t) R \Delta u(t) dt$$

with $F \geq 0, Q \geq 0, R > 0$. It tries to keep the perturbations $\Delta x(t), \Delta u(t), t_0 \leq t \leq t_f$ and $\Delta x(t_f)$ small.



Examples of how to translate verbal control objectives and constraints into mathematical performance measures (cost functions) and constraints:

- General form of to be minimized performance measure (cost function):

$$J(u(t)) = \underbrace{\phi(x(t_f), t_f)}_{\text{Terminal cost}} + \underbrace{\int_{t_0}^{t_f} L(x, u, t) dt}_{\text{Running cost}}$$

$$J, \phi, L \in \mathbb{R}^1 \text{ (scalars)}$$

By reversing the sign of the terminal cost $\phi(x(t_f), t_f)$ and running cost $L(x, u, t)$ minimization turns into maximization of the performance measure.

- General form of terminal constraints:

$$\psi(x(t_f), t_f) = 0$$

$$\psi \in \mathbb{R}^{n_\psi} \text{ (column vector)}$$

Examples:

- $x_1(t_f)$ close to 2 : $\phi(x(t_f), t_f) = (x_1(t_f) - 2)^2$: terminal costs
- $x_1(t)$ close to $\sin(t)$, $t_0 \leq t \leq t_f$: $L(x, u) = (x_1(t) - \sin(t))^2$: running costs

Both 1) and 2): $J = \underbrace{\alpha_1 (x_1(t_f) - 2)^2}_{\phi(x(t_f), t_f)} + \underbrace{\int_{t_0}^{t_f} (x_1(t) - \sin(t))^2 dt}_{L(x, u, t)}$

- $x_1(t_f)$ exactly 2 : $\psi(x(t_f), t_f) = x_1(t_f) - 2$; terminal state equality constraint
- $x_1^{\min} \leq x_1(t) \leq x_1^{\max}$, $t_0 \leq t \leq t_f$: lower and upper bound $x_1(t)$; state inequality constraint
- $u_1^{\min} \leq u_1(t) \leq u_1^{\max}$, $t_0 \leq t \leq t_f$: lower and upper bound $u_1(t)$; control inequality constraint
- Suppose $u_1(t) = P(t) = \text{power}$ $[W = Js^{-1}]$. Power when integrated equals energy.

Minimum energy consumption: $J = \int_{t_0}^{t_f} P(t) dt = \int_{t_0}^{t_f} \underbrace{u_1(t)}_{L(x, u, t)} dt$

- Terminal quadratic cost represented by a diagonal positive square matrix:

$$h_{11}x_1^2(t_f) + h_{22}x_2^2(t_f) = x^T(t_f) H x(t_f), H = \begin{bmatrix} h_{11} & 0 \\ 0 & h_{22} \end{bmatrix}, h_{11}, h_{22} > 0, x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

- Minimum-time/time-optimal control problem:

$$J = t_f \Rightarrow \phi(x(t_f), t_f) = t_f, \quad J = \int_{t_0}^{t_f} 1 dt = t_f - t_0 \Rightarrow L(x, u, t) = 1$$



10. Exercise

Consider a car moving on a straight road in a straight line from start to finish. The finish is 1 km. from the start. The dynamic behaviour of the car is represented by Newton's law: $F = ma$ where F is the force [$\text{N} = \text{kgms}^{-2}$] applied by the motor which is a control input, m is the mass of the car being 1000 kg and a is the acceleration of the car [ms^{-2}]. Both acceleration and braking is performed by the motor. The motor force is limited to 10.000 N.

- Newton's law actually is a second-order differential equation. Explain this and write down this second-order differential equation which represents a simple dynamic model of the car.
- Represent the model meant under a) in state-space form (determine and use the appropriate symbols for control inputs state variables and parameters). Mention the meaning and unit of each variable and parameter.
- The car stands still at the start. What then is the initial state?
- The goal is to reach the finish as quickly as possible. What type of optimal control problem is this?
- Write down an appropriate cost function for the optimal control problem described above (specify $t_0, t_f, \phi(x(t_f), d(t_f), t_f), L(x, u, d)$).
- Suppose that the car has to stand still at the finish line. Does this change the cost function?
- Does this change the type of optimal control problem? Does it change the mathematical description of the optimal control problem? If yes: how?
- Suppose that we want to drive the car from start to finish spending as little work (energy) as possible. Is the final time free in this case?

- The work done by the force generated by the motor equals: $\int_{t_0}^{t_f} F(t)v(t)dt$

where v represents the velocity of the car [ms^{-1}]. Furthermore suppose that the energy efficiency of the motor is 100%. Write down the appropriate cost function (specify $t_0, t_f, \phi(x(t_f), d(t_f), t_f), L(x, u, d)$).

- Suppose during driving the car may not exceed a speed limit of 40 [ms^{-1}]. How does this change the type and mathematical description of the optimal control problem?



11. Requirements for the test

Within the test you have to,

1. Write down the mathematical description of a general optimal control problem including terminal equality constraints, control inequality constraints (upper and lower bounds) and state inequality constraints (upper and lower bounds). Specify exactly what is given (known a-priori), what needs to be found (solved), what needs to be minimized and what needs to be satisfied.
2. Explain the meaning of all the symbols within this description and specify their dimensions.
3. Perform an exercise comparable to the one on Sheet 8-15.

Systems and Control Theory Part 2: Optimal Control

Lecture 9.

**Solution of optimal control problems by the
calculus of variations and Lagrange theory.**

L.G. van Willigenburg



**Wageningen University
Mathematical & Statistical Methods Group
(Biometris)**



Contents

1. Optimal control problems to be solved by the calculus of variations and Lagrange theory.
2. Static optimisation (function minimisation) with and without equality constraints: Lagrange theory.
3. Application of Lagrange theory to the optimal control problems.
4. The calculus of variations.
5. Necessary optimality conditions.
6. A two point boundary value problem.
7. Necessary optimality conditions in the case of control constraints: The minimum principle of Pontryagin.
8. Bang-bang control.
9. The difference between ϕ and ψ .
10. Exercise
11. Requirements for the test



1. Optimal control problems to be solved by the calculus of variations and Lagrange theory.

Given

- the system model in state-space form $\dot{x} = f(x, u, t)$,
- the initial state $x(t_0) = x_0$,

find the control $u(t)$, $t_0 \leq t \leq t_f$, and t_f if it is free,

that minimizes the cost function

- $J(u(t)) = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, t) dt$
- while satisfying the terminal state equality constraints $\psi(x(t_f), t_f) = 0$
- This is a dynamic optimisation problem with equality constraints (the dynamic system equations $\dot{x} = f(x, u, t)$ and the terminal constraints $\psi(x(t_f), t_f) = 0$)
- Known: $t_0, x_0, f, \phi, L, \psi$ and t_f if it is not free.



2. Static optimisation (function minimisation) with and without equality constraints: Lagrange theory.

Function minimization without constraints:

- Scalar cost function J of several variables collected in the vector u : $J(u)$, $J \in R^1$, $u \in R^{n_u}$. Example $J = u_1^2 + u_2^2$.
- Find the vector u such that $J(u)$ is minimal.

Necessary condition: $\frac{dJ}{du} = 0$.

- Example:

$$\frac{\partial J}{\partial u} = \begin{bmatrix} \frac{\partial J}{\partial u_1} & \frac{\partial J}{\partial u_2} \end{bmatrix} = \begin{bmatrix} 2u_1 & 2u_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \end{bmatrix} \Rightarrow u_1 = 0, u_2 = 0$$

- Rules for differentiation $t \in R$, $x, \lambda, f, g \in R^n$, $A, S \in R^{n \times n}$

Product rule 1: $\frac{d}{dx} \left(f^T(x) g(x) \right) = f^T \frac{dg}{dx} + g^T \frac{df}{dx}$

Product rule 2: $\frac{d}{dt} \left(S(t) x(t) \right) = \frac{dS}{dt} x + S \frac{dx}{dt}$

$$\frac{\partial}{\partial x} (\lambda^T x) = \frac{\partial}{\partial x} (x^T \lambda) = \lambda^T$$

$$\frac{\partial}{\partial x} (\lambda^T A x) = \frac{\partial}{\partial x} (x^T A^T \lambda) = \lambda^T A$$

$$\frac{\partial}{\partial x} (\lambda^T f(x)) = \frac{\partial}{\partial x} (f^T(x) \lambda) = \lambda^T \frac{df}{dx}$$

$$\frac{\partial}{\partial x} (x^T A x) = x^T (A + A^T)$$

$$\frac{\partial}{\partial x} (A x) = A$$



Function minimization with constraints:

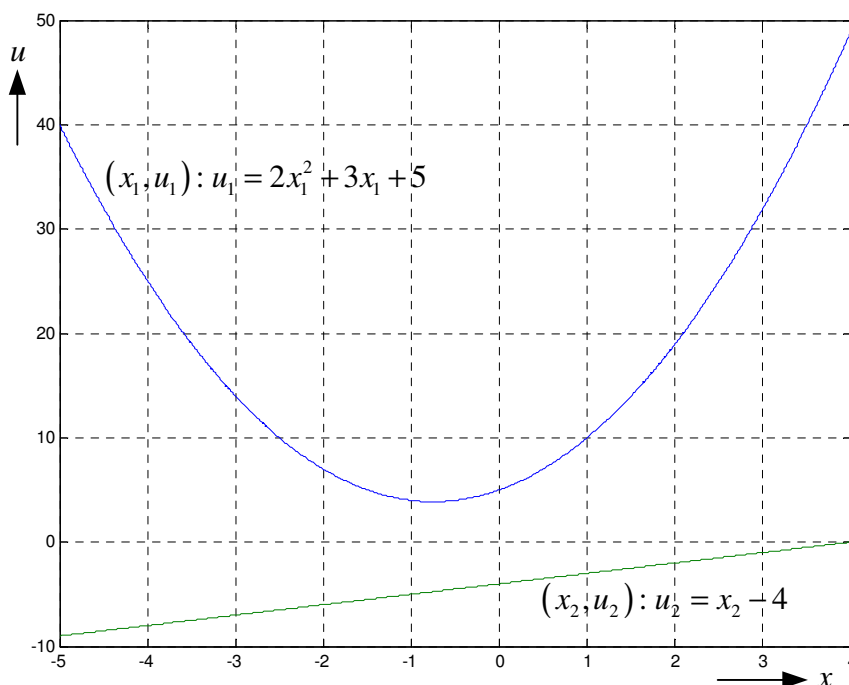
Scalar cost function J of several variables collected in the vectors u, x : $J(u, x)$, $J \in \mathbb{R}^1$, $u \in \mathbb{R}^{n_u}$, $x \in \mathbb{R}^{n_x}$.

Example $J = \frac{1}{2}(u_1 - u_2)^2 + \frac{1}{2}(x_1 - x_2)^2$.

Constraints: $f(x, u) = 0$, $f \in \mathbb{R}^{n_f}$

Example $f = \begin{bmatrix} u_1 - 2x_1^2 - 3x_1 - 5 \\ u_2 - x_2 + 4 \end{bmatrix}$

Interpretation of this example: Minimum distance between a parabola and a line:





Define the extended cost function: $J' = J + \lambda^T f, \lambda \in R^{n_f}$

Example:

$$J' = \frac{1}{2}(u_1 - u_2)^2 + \frac{1}{2}(x_1 - x_2)^2 + \begin{bmatrix} \lambda_1 & \lambda_2 \end{bmatrix} \begin{bmatrix} u_1 - 2x_1^2 - 3x_1 - 5 \\ u_2 - x_2 + 4 \end{bmatrix} =$$

$$\frac{1}{2}(u_1 - u_2)^2 + \frac{1}{2}(x_1 - x_2)^2 + \lambda_1(u_1 - 2x_1^2 - 3x_1 - 5) + \lambda_2(u_2 - x_2 + 4)$$

Suppose that x, u, λ are independent variables and compute the minimum of J' :

$$\frac{\partial J'}{\partial \lambda} = 0 \Rightarrow f^T(x, u) = 0 \Rightarrow J' = J$$

then

$$\frac{\partial J'}{\partial x} = \frac{\partial J}{\partial x} = 0$$

$$\frac{\partial J'}{\partial u} = \frac{\partial J}{\partial u} = 0$$

The function minimization of J with constraints $f=0$ can be performed by the minimization of the function J' without constraints!

The price we have to pay is the introduction of extra variables collected in the vector λ called Lagrange multipliers.

Lagrange theory: By multiplying each constraint with a Lagrange multiplier we can convert a constrained function minimisation problem into an unconstrained function minimisation problem!



3. Application of Lagrange theory to the optimal control problems

$$J'(u(t)) = \phi\left(x(t_f), t_f\right) + v^T \psi\left(x(t_f)\right) + \int_{t_0}^{t_f} L(x, u, t) + \lambda^T(t) \left(f(x, u, t) - \dot{x}\right) dt$$

Lagrangre multipliers: v and $\lambda(t), t_0 \leq t \leq t_f$ (so one vector of Lagrange multipliers λ for each time $t_0 \leq t \leq t_f$.)

- $\lambda(t) \in R^{n_x}$: co-state
- Minimise $J'(u(t))$ assuming $u(t), x(t), \lambda(t), v, t_f$ are independent.
- Running costs: $\int_{t_0}^{t_f} (H - \lambda^T \dot{x}) dt$ with H the Hamiltonian:

$$H(x, u, t) = L(x, u, t) + \lambda^T f(x, u, t)$$
- $J'(u(t))$ is a functional and not a function. (a function depends only on variables a functional depends on functions and variables)
- To minimise a functional we have to apply the calculus of variations.



4. The calculus of variations

- Function $J(x)$:

Differential dx : small change in x .

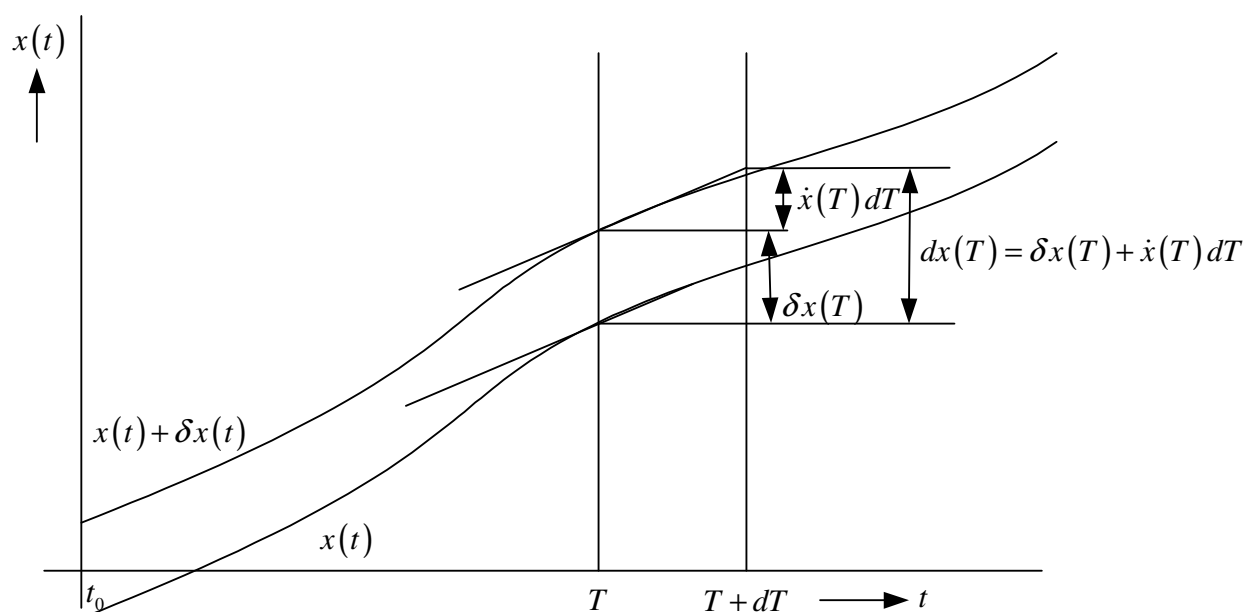
$$dJ(x) = \frac{\partial J}{\partial x} dx.$$

At a local minimum: $\frac{\partial J}{\partial x} = 0 \Leftrightarrow dJ(x) = 0$ for arbitrary dx

- Functional $J(x(t)), t_0 \leq t \leq t_f$:

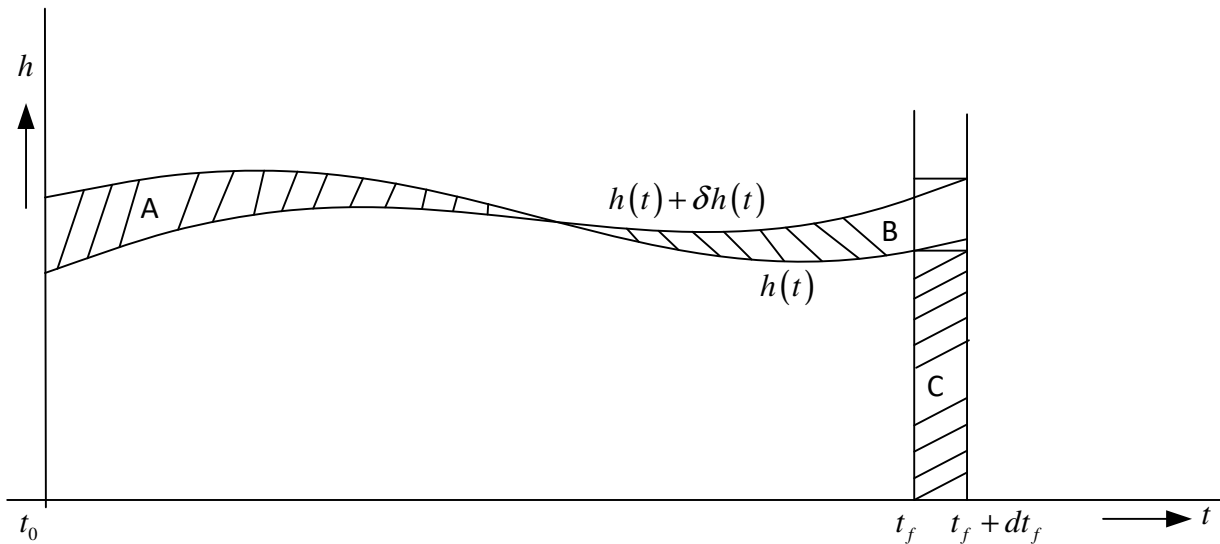
At a local minimum: $dJ(x(t)) = 0$ for arbitrary $dx(t)$

- Variation $\delta x(T)$: small change in $x(T)$ where T is fixed
- Differential $dx(T)$: small change in $x(T)$ where T is free
- $dx(T) = \delta x(T) + \dot{x}(T)dT \Leftrightarrow \delta x(T) = dx(T) - \dot{x}(T)dT$





$$dJ(x(t)) = h\left(x(t_f), t_f\right) dt_f + \int_{t_0}^{t_f} \delta h(t) dt$$



$$\text{Area B} - \text{Area A} = \int_{t_0}^{t_f} \delta h(t) dt$$

$$\text{Area C} = h\left(x(t_f), t_f\right) dt_f$$

- Leibniz rule: $J(x(t)) = \int_{t_0}^{t_f} h(x(t), t) dt$ then

$$dJ(x(t)) = h\left(x(t_f), t_f\right) dt_f + \int_{t_0}^{t_f} \underbrace{\left[\frac{\partial h}{\partial x} \delta x(t) \right]}_{\delta h(t)} dt$$

- Integration by parts

$$\int_{t_0}^{t_f} -\lambda^T \delta \dot{x} dt = -\lambda^T \delta x(t_f) + \int_{t_0}^{t_f} \dot{\lambda}^T \delta x dt$$



Extended cost function in which $H = L + \lambda^T f$:

Eq. in Lect. notes

$$J'(u(t)) = \phi(x(t_f), t_f) + v^T \psi(x(t_f), t_f) + \overbrace{\int_{t_0}^{t_f} \left(\underbrace{H(x, u, d, t)}_a - \underbrace{\lambda^T \dot{x}}_b \right) dt}_{J'_r} \quad (9.56)$$

$$\text{Split: } dJ' = d\phi + d(v^T \psi) + dJ'_r \quad (..)$$

$$d\phi = \frac{\partial \phi}{\partial x} dx \Big|_{t_f} + \frac{\partial \phi}{\partial t} dt \Big|_{t_f} \quad (9.57)$$

$$d(v^T \psi) = v^T \frac{\partial \psi}{\partial x} dx \Big|_{t_f} + v^T \frac{\partial \psi}{\partial t} dt \Big|_{t_f} + \psi^T \Big|_{t_f} dv \quad (9.58)$$

Leibniz rule:

$$dJ'_r = \int_{t_0}^{t_f} \left(\underbrace{\frac{\partial H}{\partial x} \delta x + \frac{\partial H}{\partial u} \delta u + \frac{\partial H}{\partial \lambda} \delta \lambda}_a - \underbrace{\lambda^T \delta \dot{x} - \dot{x}^T \delta \lambda}_b \right) dt + (H - \lambda^T \dot{x}) dt \Big|_{t_f} \quad (..)$$

Integration by parts (9.60) together with the relation (9.45):

$$\int_{t_0}^{t_f} \underbrace{-\lambda^T \delta \dot{x}}_{b_1} dt = \underbrace{-\lambda^T dx \Big|_{t_f}}_{b_{1,1}} + \underbrace{\lambda^T \dot{x} dt \Big|_{t_f}}_{b_{1,2}} + \underbrace{\int_{t_0}^{t_f} \dot{\lambda}^T \delta x dt}_{b_{1,3}} \quad (9.61)$$

Final expression:

$$\begin{aligned} dJ' = & \left(\frac{\partial \phi}{\partial x} + v^T \frac{\partial \psi}{\partial x} - \underbrace{\lambda^T}_{b_{1,1}} \right) dx \Big|_{t_f} + \psi^T \Big|_{t_f} dv + \left(\underbrace{\frac{\partial \phi}{\partial t} + v^T \frac{\partial \psi}{\partial t} + H - \lambda^T \dot{x} + \underbrace{\lambda^T \dot{x}}_{b_{1,2}}}_{=0} \right) dt \Big|_{t_f} \\ & + \int_{t_0}^{t_f} \left(\left(\underbrace{\frac{\partial H}{\partial x}}_{a_1} + \underbrace{\dot{\lambda}^T}_{b_{1,3}} \right) \delta x + \underbrace{\frac{\partial H}{\partial u}}_{a_2} \delta u + \left(\underbrace{\frac{\partial H}{\partial \lambda}}_{a_3} - \underbrace{\dot{x}^T}_{b_2} \right) \delta \lambda \right) dt \end{aligned} \quad (9.62)$$



5. Necessary optimality conditions: $H = L + \lambda^T f$

State equation:

$$\dot{x} = \left(\frac{\partial H}{\partial \lambda} \right)^T = f, \quad t_0 \leq t \leq t_f$$

Co-state equation:

$$-\dot{\lambda} = \left(\frac{\partial H}{\partial x} \right)^T = \left(\frac{\partial f}{\partial x} \right)^T \lambda + \left(\frac{\partial L}{\partial x} \right)^T, \quad t_0 \leq t \leq t_f$$

Stationarity condition:

$$0 = \left(\frac{\partial H}{\partial u} \right)^T = \left(\frac{\partial f}{\partial u} \right)^T \lambda + \left(\frac{\partial L}{\partial u} \right)^T, \quad t_0 \leq t \leq t_f$$

Boundary conditions:

Initial state: $x(t_0)$.

If terminal state entirely fixed: $x(t_f) = x^{ref}$

otherwise terminal co-state: $\left(\frac{\partial \phi}{\partial x} + \nu^T \frac{\partial \psi}{\partial x} - \lambda^T \right) \Big|_{t_f} = 0$

If t_f free terminal Hamiltonian: $\left(\frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \psi}{\partial t} + H \right) \Big|_{t_f} = 0$



6. A two point boundary value problem

- Stationarity condition $\Rightarrow u(x, \lambda)$. This eliminates u .
- State plus co-state equation: $2n_x$ differential equations in x and λ .

- $x(t_0)$ and $x(t_f)$ or $\left(\frac{\partial \phi}{\partial x} + v^T \frac{\partial \psi}{\partial x} - \lambda^T \right) \bigg|_{t_f} = 0$:

$2n_x$ boundary conditions.

- $\psi=0$: n_ψ equations then v : n_v unknowns
 - If t_f free then $\left(\frac{\partial \phi}{\partial t} + v^T \frac{\partial \psi}{\partial t} + H \right) \bigg|_{t_f} = 0$: one equation.
 - For each differential equation there is a boundary condition and for each unknown an equation:
- A two point boundary value problem (boundary conditions at two time points: t_0, t_f).
- No terminal state constraints: ψ, v drop (not present).
 - No terminal costs: ϕ drops (not present).



7. Necessary optimality conditions in the case of control constraints: The minimum principle of Pontryagin.

- In the case of control constraints only the stationarity condition changes into:

$H(x^*, u^*, \lambda^*, t) \leq H(x^*, u, \lambda^*, t)$ for all $t \in [t_0, t_f]$ and all admissible control values u . This rule is called Pontryagin's minimum principle.

- Both the stationarity condition and Pontryagin's minimum principle can be stated as: Minimize H with respect to admissible u (considering x, λ constant).

8. Bang-bang control and constant Hamiltonian

- If $H(x, u, \lambda)$ is linear in u the control is bang-bang meaning that at each time $t_0 \leq t \leq t_f$ each control input is either maximal or minimal (except when the problem is singular).
- $H(x, u, \lambda, t)$ linear in $u \Leftrightarrow f(x, u, t)$ and $L(x, u, t)$ linear in u .
- If $f = f(x, u)$ and $L = L(x, u)$ so f, L are time-invariant then $H(x^*, u^*, \lambda^*)$ is constant.



9. The difference between ϕ and ψ

- Terminal constraints $\psi\left(x(t_f), t_f\right)=0$ have to be satisfied

exactly. Example: $\psi\left(x(t_f), t_f\right)=x(t_f)-\begin{bmatrix} 4 \\ 2 \end{bmatrix}=0$

- The terminal costs $\phi\left(x(t_f), t_f\right)$ is part of $J(u(t))$ and is therefore minimised together with the running costs.

Example: $\phi\left(x(t_f), t_f\right)=\left(x(t_f)-\begin{bmatrix} 4 \\ 2 \end{bmatrix}\right)^T \left(x(t_f)-\begin{bmatrix} 4 \\ 2 \end{bmatrix}\right)$

In this case $x(t_f)-\begin{bmatrix} 4 \\ 2 \end{bmatrix}=0$ does not necessarily hold

because:

- The system cannot reach the state $x(t_f)=\begin{bmatrix} 4 \\ 2 \end{bmatrix}$
- The running costs are too high to reach $x(t_f)=\begin{bmatrix} 4 \\ 2 \end{bmatrix}$ so an optimal compromise is obtained for which $x(t_f) \neq \begin{bmatrix} 4 \\ 2 \end{bmatrix}$



Exercise: Derivation of optimal LQ state feedback.

Consider the following Linear Quadratic (LQ) optimal control problem called the LQ optimal regulator problem.

System model: $\dot{x} = A(t)x + B(t)u$

Cost function: $\frac{1}{2}x^T(t_f)Fx(t_f) + \frac{1}{2}\int_{t_0}^{t_f} (x^T Q(t)x + u^T R(t)u) dt$

where $F \geq 0, Q(t) \geq 0, R(t) \geq 0, t_0 \leq t \leq t_f$. t_f fixed.

- To answer the following questions use the rules of differentiation on sheet 9-4. Note that: $(AB)^T = B^T A^T$
- a) Show that if $R = R^T$ from the stationarity condition it follows that $u = -R^{-1}B^T \lambda$ by which we can eliminate u .
- b) Use the expression $u = -R^{-1}B^T \lambda$ to show that the state equation is $\dot{x} = Ax - BR^{-1}B^T \lambda$. If $Q = Q^T$ show that the co-state equation is $\dot{\lambda} = -Qx - A^T \lambda$.
- c) Show that if $F = F^T$ according to the terminal boundary condition $\lambda(t_f) = Fx(t_f)$.
- d) Suppose $\lambda(t) = S(t)x(t), t_0 \leq t \leq t_f$ where $S(t) = S^T(t)$ is a symmetric time-varying $n_x \times n_x$ matrix.

This is true for $t = t_f$: $S(t_f) = F$

Apply the product rule to show that, with the above assumption: $\dot{\lambda} = \dot{S}x + S\dot{x} = \dot{S}x + S(Ax - BR^{-1}B^T Sx)$.

- e) Now use the co-state equation to show that $-\dot{S}x = (A^T S + SA - SBR^{-1}B^T S + Q)x$



- The expression above must hold for different states x . Therefore: $-\dot{S} = A^T S + SA - SBR^{-1}B^T S + Q$ must hold.
- This is a matrix differential equation where the element i,j of the matrix on the right is minus the time-derivative of element i,j of S .
- From d) the terminal boundary condition of this matrix differential equation is $S(t_f) = F$. From this terminal boundary condition we can integrate numerically backward in time to obtain $S(t), t_0 \leq t \leq t_f$. Note that this computation can be performed off-line since it does not depend on $x(t)$.
- Having $S(t), t_0 \leq t \leq t_f$ from the answers to a) and d) the optimal control is: $u^*(t) = -R^{-1}(t)B^T(t)S(t)x(t) = -L(t)x(t)$. This is the optimal LQ state feedback law mentioned on sheet 7-8.
- $L(t) = R^{-1}(t)B^T(t)S(t)$ is called the optimal LQ state feedback gain which can be computed off-line and stored in a control computer. As an input the control computer receives the measurements $x(t)$ and as an output it generates the optimal control $u^*(t) = -L(t)x(t)$.

Finally one can prove that: $J^*(u^*(t)) = \frac{1}{2}x^T(t_0)S(t_0)x(t_0)$.



11. Requirements for the test

Within the test you have to,

1. Formulate a general optimal control problem with terminal equality constraints, free final time, but without state or control constraints.
2. Write the necessary optimality conditions for the solution of the optimal control problem mentioned under 1.
3. Know how to modify 1. and 2. when there are control inequality constraints.
4. Repeat the exercise on sheet 9-15, 9-16 and explain why $S(t), L(t), J^*$ can be computed in advance (off-line).

Systems and Control Theory Part 2: Optimal Control

Lecture 10.

Gradient algorithms to solve optimal control problems.

Optimal LQ and LQG feedback design.

L.G. van Willigenburg

Contents

1. Necessary optimality conditions for optimal control problems with fixed final time and no terminal constraints
2. Gradient algorithms
3. Advantages of gradient algorithms
4. Disadvantages of gradient methods
5. Mayer formulation of optimal control problems
6. Justification of LQ and LQG feedback design
7. Selection of the quadratic cost function matrices
8. Dynamic optimal LQG output feedback
9. Time invariant infinite horizon LQ problems and stability
10. The LQG problem and the separation property
11. Compulsory computer exercise
12. Requirement for the test





1. Necessary optimality conditions for optimal control problems with fixed final time and no terminal state constraints

State equation:

$$\dot{x} = \left(\frac{\partial H}{\partial \lambda} \right)^T = f, \quad t_0 \leq t \leq t_f$$

Co-state equation:

$$-\dot{\lambda} = \left(\frac{\partial H}{\partial x} \right)^T = \left(\frac{\partial f}{\partial x} \right)^T \lambda + \left(\frac{\partial L}{\partial x} \right)^T, \quad t_0 \leq t \leq t_f$$

Stationarity condition / minimum principle:

$$0 = \left(\frac{\partial H}{\partial u} \right)^T = \left(\frac{\partial f}{\partial u} \right)^T \lambda + \left(\frac{\partial L}{\partial u} \right)^T, \quad t_0 \leq t \leq t_f$$

$$H(x^*, u^*, \lambda^*, t) \leq H(x^*, u, \lambda^*, t), \quad t_0 \leq t \leq t_f$$

Boundary conditions:

Initial state: $x(t_0)$

$$\left. \left(\left(\frac{\partial \phi}{\partial x} \right)^T - \lambda \right) \right|_{t=t_f} = 0 \Leftrightarrow \lambda(t_f) = \left. \left(\frac{\partial \phi}{\partial x} \right)^T \right|_{t=t_f}$$

Of these 4 conditions 3 can usually be met easily.

Algorithms **iterate** to satisfy the 4th.



2. Gradient algorithms

The algorithm iterates towards satisfying the station. cond. / minimum principle

Algorithm:

- 1 Start with an initial guess $u(t)$, $t_0 \leq t \leq t_f$ for the control
- 2 Starting from $x(t_0)$ numerically integrate the system equation forward in time. Store the result $x(t)$, $t_0 \leq t \leq t_f$.
- 3 From $x(t_f)$ and ϕ compute $\lambda(t_f) = \left(\frac{\partial \phi}{\partial x} \right)^T \Big|_{t=t_f}$.
- 4 Starting at $\lambda(t_f)$ integrate the co-state equation backward in time using $x(t)$, $t_0 \leq t \leq t_f$ to compute $\left(\frac{\partial f}{\partial x} \right)^T, \left(\frac{\partial L}{\partial x} \right)^T$. Store $\lambda(t)$, $t_0 \leq t \leq t_f$.
- 5 Using $u(t)$, $x(t)$, $\lambda(t)$ compute $\left(\frac{\partial H}{\partial u} \right)^T = \left(\frac{\partial f}{\partial u} \right)^T \lambda + \left(\frac{\partial L}{\partial u} \right)^T$, $t_0 \leq t \leq t_f$
- 6 Select $\alpha > 0 \in R^1$ such that $u'(t) = u(t) - \alpha \left(\frac{\partial H}{\partial u} \right)^T$ minimizes the cost function : a line search. $u(t) := u'(t)$
- 7 If the cost function improvement is below a tolerance stop: $u(t) = u^*(t)$ else goto 2
- 8 Verify if $\frac{\partial H}{\partial u_i}$ is "small" if u_i is not at a bound $i=1,2,\dots,n_u$



3. Advantages of gradient algorithms

- 1 The inaccuracy of the “intermediate solutions” is roughly equal over the time-interval $t_0 \leq t \leq t_f$.
- 2 The stationarity condition / minimum principle is satisfied numerically which may save programming effort and analysis of the problem (for instance if the problem is singular).
- 3 The algorithm requires less accurate initial guesses which may save dynamic analysis of the problem.

4. Disadvantages of gradient methods

- 1 Not very efficient computationally.
- 2 Not very well suited to solve problems with terminal constraints and free final time.
- 3 Not suitable to solve problems with state constraints. (State constraints can be handled approximately through penalty functions).

Important reference:

Dynamic Optimization

Bryson Jr.

Addison & Wesley, 1999



5. Mayer formulation of optimal control problems

- Running costs until time t :

$$J_r(t) = \int_{t_0}^t L(x, u, t) dt \Leftrightarrow \frac{\partial J_r}{\partial t} = L(x, u, t), J_r(t_0) = 0$$

- Running costs as an additional state variable:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n_x} \\ \dot{J}_r \end{bmatrix} = \begin{bmatrix} f_1(x, u, t) \\ f_2(x, u, t) \\ \vdots \\ f_{n_x}(x, u, t) \\ L(x, u, t) \end{bmatrix}, \quad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_x} \\ J_r \end{bmatrix} = \begin{bmatrix} x \\ J_r \end{bmatrix} = x', \quad \begin{bmatrix} f_1(x, u, t) \\ f_2(x, u, t) \\ \vdots \\ f_{n_x}(x, u, t) \\ L(x, u, t) \end{bmatrix} = \begin{bmatrix} f \\ L \end{bmatrix} = f'$$

$$\text{If } J_r(t_0) = x'_{n_x+1}(t_0) = 0 \Rightarrow J_r(t_f) = x'_{n_x+1}(t_f) = \int_{t_0}^{t_f} L(x, u, t) dt$$

- After numerical integration of $\dot{x}' = f'$ from t_0 to t_f :

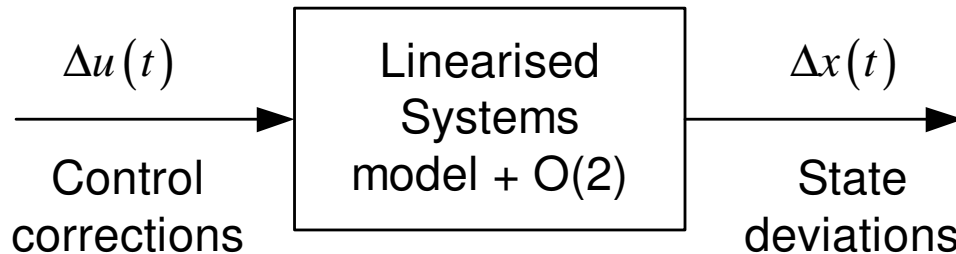
$$J(u(t)) = \phi(x(t_f)) + J_r(t_f) = \phi(x(t_f)) + x'_{n_x+1}(t_f) = \phi'(x'(t_f))$$

- 1) Equivalent problem $\dot{x}' = f'$, $J = \phi'$ without running costs.
- 2) A way to numerically compute the running costs



6. Justification of LQ and LQG feedback design

- Optimal feedback control problem:



How to compute control corrections such that both $\Delta x(t)$ and $\Delta u(t)$ remain small

- Taylor series expansion:

$$\Delta \dot{x}(t) = \left. \frac{\partial f}{\partial x} \right|_{x(t)=x^*(t), u(t)=u^*(t)} \Delta x(t) + \left. \frac{\partial f}{\partial u} \right|_{x(t)=x^*(t), u(t)=u^*(t)} \Delta u(t) + O(2)$$

$O(2)$: second and higher order terms.

- If $\Delta x(t)$ and $\Delta u(t)$ remain small the linearized model

$$\Delta \dot{x}(t) = \left. \frac{\partial f}{\partial x} \right|_{x(t)=x^*(t), u(t)=u^*(t)} \Delta x(t) + \left. \frac{\partial f}{\partial u} \right|_{x(t)=x^*(t), u(t)=u^*(t)} \Delta u(t) = A(t)\Delta x(t) + B(t)\Delta u(t)$$

approximately describes the behaviour of $\Delta u(t)$, $\Delta x(t)$.



- If $\Delta x(t)$ and $\Delta u(t)$ remain small second order terms in Δx and Δu dominate $O(2)$ the error due to linearization.

- A quadratic cost function

$$J(\Delta u(t)) = \Delta x^T(t_f) F \Delta x(t_f) + \int_{t_0}^{t_f} [\Delta x^T(t) Q(t) \Delta x(t) + \Delta u^T(t) R(t) \Delta u(t)] dt$$

minimizes such quadratic terms. Therefore:

- A quadratic cost function minimizes errors due to linearization.
- The solution of the LQ regulator problem is in state feedback form: $\Delta u^*(t) = -L(t) \Delta x(t)$. This is exactly what we need!

7. Selection of the quadratic cost function matrices

- To keep $\Delta x(t)$ and $\Delta u(t)$ small $Q(t) \geq 0, R(t) \geq 0$ is required. In addition $F \geq 0$ further reduces $\Delta x(t_f)$.
- $Q(t) \geq 0, R(t) \geq 0$ must be a kind of upperbound of the second-order terms and in addition must be sufficiently positive to keep the perturbations $\Delta x(t)$ and $\Delta u(t)$ small.
- $F \geq 0$ must be a kind of upperbound of the second-order terms and must be sufficiently positive to further reduce $\Delta x(t_f)$.



8. Dynamic optimal LQG output feedback

Now we also require a solution of the dual LQ problem:
 This problem is fully determined by the matrices

$$A(t) = \left. \frac{\partial f}{\partial x} \right|_{x(t)=x^*(t), u=u^*(t)}, C(t) = \left. \frac{\partial g}{\partial x} \right|_{x(t)=x^*(t), u=u^*(t)},$$

$$V(t) \in R^{n_x \times n_x}, W(t) \in R^{n_y \times n_y}, G \in R^{n_x \times n_x}$$

The optimal estimator gain matrices $K(t)$ equal

$$K(t) = P(t)C^T(t)W^{-1}(t)$$

where $P(t)$ is the solution of the matrix differential equation

$$\dot{P} = AP + PA^T - PC^T W^{-1} CP + V, \quad t_0 \leq t \leq t_f$$

with initial condition

$$P(t_0) = G$$

- $V(t), W(t)$: intensity matrices describing small modelling and measurement errors (uncertainties).
- G : covariance matrix describing the small initial state errors (uncertainties).
- If $V(t), W(t), G$ are taken to be diagonal matrices the diagonal elements are roughly equal to the average squared error of the associated state derivative or output.



9. Time invariant infinite horizon LQ problems and stability

- If $A(t)=A, B(t)=B, Q(t)=Q \geq 0, R(t)=R \geq 0, t_f \rightarrow \infty$, and A, B controllable then

$S(t) \rightarrow S \Rightarrow L(t) \rightarrow R^{-1}B^T S, \dot{S} \rightarrow 0$: where S is now the unique solution of the algebraic Riccati equation:

$$0 = A^T S + SA - SBR^{-1}B^T S + Q$$

and $\Delta x(t) \rightarrow 0$ for any small initial state perturbation $\Delta x(t_0)$. In other words: The feedback control system is stable: All small disturbances are controlled to zero.

- If an LQ regulator problem is time-invariant and the linear system is controllable the optimal closed loop control system:

$$\dot{x} = (A - BL)x$$

is asymptotically stable $\Leftrightarrow$ Eigenvalues matrix $(A - BL)$ all have a negative real part.

Remark: Controllable may be replaced with stabilizable which is a weaker property.



10. The LQG problem and the separation property

- In the LQG optimal control problem modelling and measurement errors are considered and described by white noise (stochastic processes). Apart from this the linear system and quadratic cost function remain unchanged.
- The optimal solution to this problem consists of the dynamic output feedback scheme on sheets 7-7, 7-8.
- The optimal Kalman filter gain $K(t)$ and the optimal feedback gain $L(t)$ are computed from the two dual LQ problems just presented.
- The fact that $L(t)$ and $K(t)$ can be computed independently from one another is called the separation property of the LQG problem.
- Separability of an optimal control problem means that the optimal controller can be separated in a state estimator and a state feedback part which can be computed independently from one another.
- Most optimal control problems are not separable. To simplify the computation and design one sometimes assumes that they are. The solution is then clearly sub optimal.



11. Compulsory computer exercise 2: The thermal sterilization of canned food.

- Assumptions:

A homogeneous temperature T_s of the liquid in the sterilizer

A homogeneous temperature T_f of the food in the cans

- Differential equations:

$$\frac{dT_f}{dt} = c_1(T_s - T_f), \quad \frac{dT_s}{dt} = c_2(T_a - T_s) + c_3H$$

$$\frac{dF}{dt} = \exp\left\{\frac{\ln(10)}{\delta_{refm}}(T_f - T_{refm})\right\}, \quad \frac{dN}{dt} = \exp\left\{\frac{\ln(10)}{\delta_{refn}}(T_f - T_{refn})\right\}$$

$H(t)$: Heating power, $0 \leq H(t) \leq 50$.

$F(t)$: Sterilization value at time t , $F(t_0) = 0$.

$N(t)$: Value at time t associated with nutrient retention,

$N(t_0) = 0$. ($N(t) = 1/G(t)$ in the paper by Chalabi *et.al*)

$$T_f(t_0) = T_s(t_0) = T_a = 20, \quad \delta_{refm} = 10, \quad \delta_{refn} = 25.56,$$

$$T_{refm} = T_{refn} = 121.11, \quad c_1 = 0.01, \quad c_2 = 0.005, \quad c_3 = 0.01,$$

$$t_0 = 0, \quad t_f = 1500.$$

- Control objectives:

$F(t_f)$ close to 100. $T_f(t_f)$ close to 40. Minimize $N(t_f)$.



- a) **Preparation:** Formulate on paper the optimal control problem. In other words determine $t_0, t_f, x(t_0), f, \phi, L$.
- b) **Computer exercise:** You will solve this optimal control problem using Tomlab. Further instructions you find in the ce2.pdf file in the ce2 folder.

12. Requirement for the test:

Mention 3 reasons why solving an LQ problem is highly suitable for state-feedback and output-feedback design.

Systems and Control Theory Part 2: Optimal Control

Lecture 11.

Transformation of optimal control problems into constrained function minimization problems.

L.G. van Willigenburg

Contents

1. Objectives of the transformation
2. Function minimization with inequality constraints using slack variables
3. Control parametrization and state constraints
4. Free final time and time scaling
5. Advantage and disadvantage of function minimization
6. Digital control systems and control parametrization
7. Practical exercise



**Wageningen University
Mathematical & Statistical Methods Group
(Biometris)**



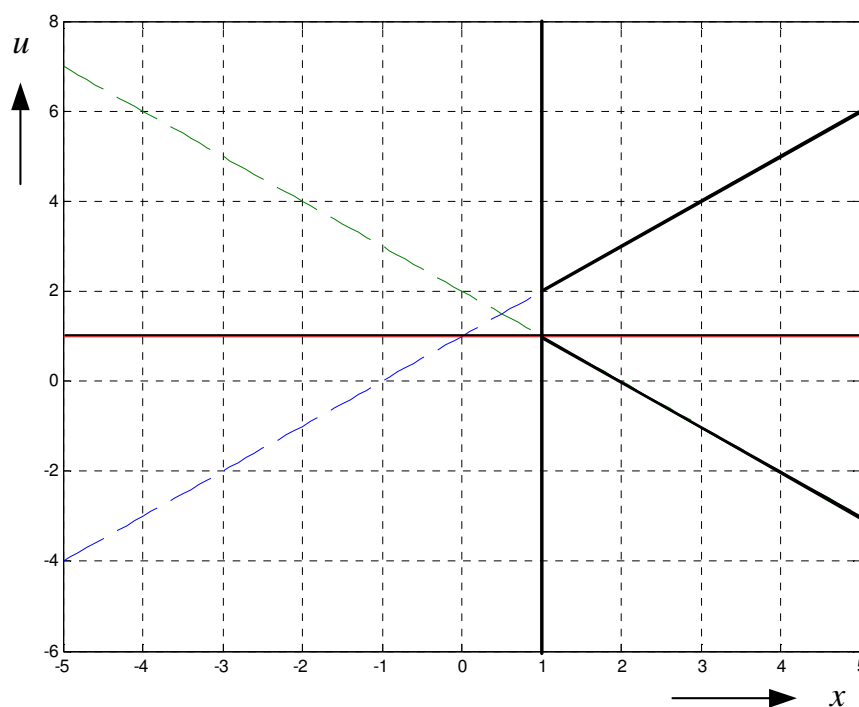
1. Objectives of the transformation:

Solve optimal control problems with state, control and terminal equality and/or inequality constraints and possibly free final time.

2. Function minimization with inequality constraints using slack variables

An example:

Find two points (x_1, u_1) and (x_2, u_2) the first one on the line $u_1 = x_1 + 1$ and the second on the line $u_2 = -x_2 + 2$ which have the smallest distance while satisfying $x_1 \geq 1$ and $u_2 \leq 1$.





- Equality constraints: $u_1 = x_1 + 1, u_2 = -x_2 + 2$
- Inequality constraints: $x_1 \geq 1, u_2 \leq 1$
- Function to be minimized: $J = \sqrt{(x_1 - x_2)^2 + (u_1 - u_2)^2}$ or
alternatively $\frac{1}{2}(x_1 - x_2)^2 + \frac{1}{2}(u_1 - u_2)^2$

Extended cost function:

$$J' = J + \lambda_1(u_1 - x_1 - 1) + \lambda_2(u_2 + x_2 - 2) + \lambda_3(x_1 - 1 - y_1^2) + \lambda_4(-u_2 + 1 - y_2^2)$$

y_1, y_2 : slack variables turn inequalities into equalities!

$$J' = J + \lambda^T f = J + (\lambda_1, \lambda_2, \lambda_3, \lambda_4) \begin{pmatrix} u_1 - x_1 - 1 \\ u_2 + x_2 - 2 \\ x_1 - 1 - y_1^2 \\ -u_2 + 1 - y_2^2 \end{pmatrix}.$$

Minimize J' with respect to $u_1, u_2, x_1, x_2, \lambda_1, \lambda_2, \lambda_3, \lambda_4, y_1, y_2$

$$\frac{\partial J'}{\partial u_1} = 0, \frac{\partial J'}{\partial u_2} = 0, \dots, \frac{\partial J'}{\partial y_2} = 0: 10 \text{ eq. with 10 unknowns}$$

- Show Matlab solution



- Inequality constraints in $x \in R^{n_x}$ can always be written in the form $g(x) \geq 0$ or $g(x) \leq 0$, $g \in R^{n_g}$
- Inequality constraints are transformed into equality constraints by associating one slack variable to each inequality constraint.
- The “price” we have to pay for this transformation is the introduction of the additional slack variables.
- Function minimization problems with equality constraints can be transformed into function minimization problems without constraints by introducing Lagrange multipliers.
- Function minimization problems with equality and/or inequality constraints can be solved as if there are no constraints.



```
% Exslec11: Script 1 lecture 11, Optimal control with state constraints
%
% Input:
% Options vector opt for the function constr that performs the constrained
% function minimization
%         opt(1) : 1=display intermediate results
%         opt(13): number of equality constraints
%         opt(14): 500, maximum number of function evaluations
% GvW 21-2-2006
%

% Set options vector: opt(1) : 1=display intermediate results
%                   opt(13) : 2 equality constraints
opt(1)=1; opt(13)=2;
% Initial guess for solution x(1:2)=u, x(3:4)=x
xguess=[1;1;1;1];
% Perform the constraint function minimization and return the solution x
[x,options,lambd]=constr('flec11',xguess,opt);
% Split x in u,x
u=x(1:2); x=x(3:4);
% Display the solution;
disp(['(u1,x1)=(' num2str(u(1)) ',' num2str(x(1)) ')']);
disp(['(u2,x2)=(' num2str(u(2)) ',' num2str(x(2)) ')']);
% Display Lambda
disp('Lambda:'); lambda

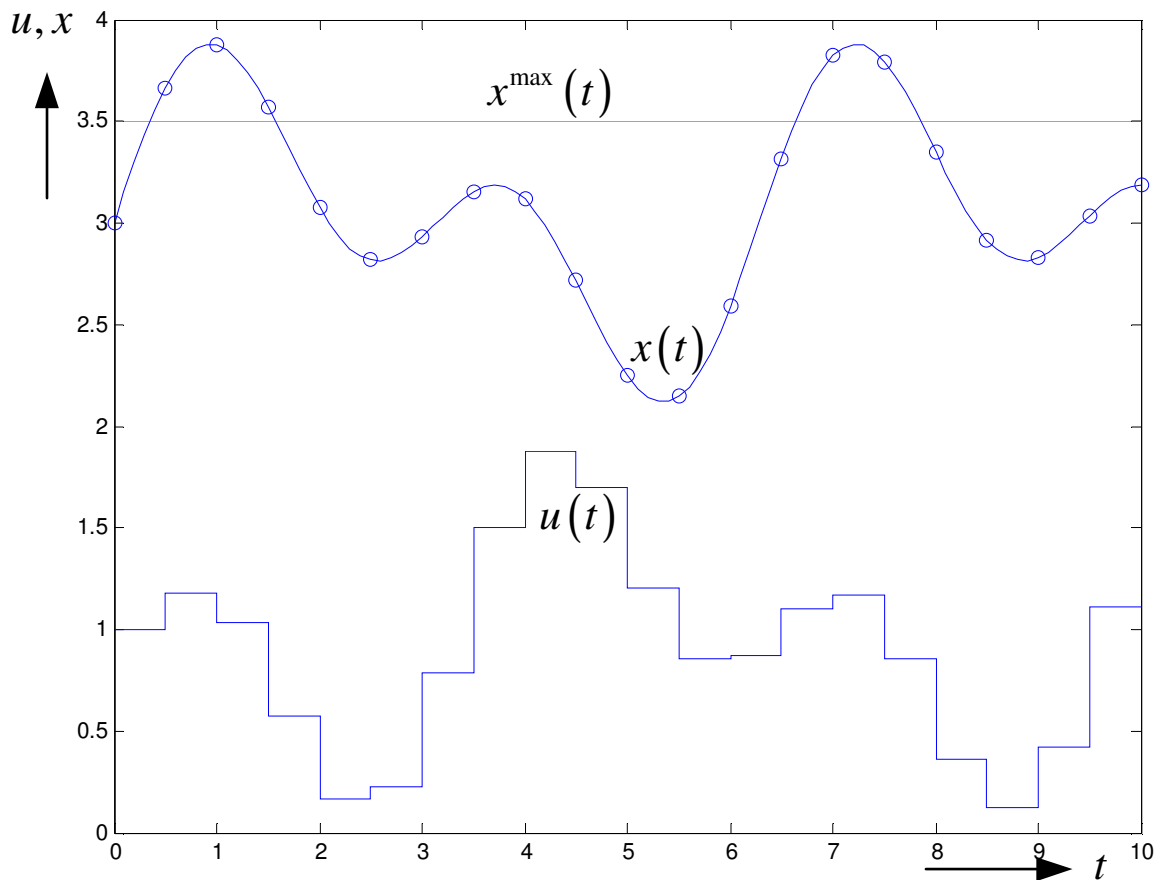
% flec11: Constrained function minimization example lecture 11
function [f,g]=flec11(x);

% Split x in u and x
u=x(1:2); x=x(3:4);
% f: function to be minimized
f=0.5*(u(1)-u(2))*(u(1)-u(2))+0.5*(x(1)-x(2))*(x(1)-x(2));
% g: constraints, first the 2 equality constr.
% then the 2 inequality constr <=0!!
g=[u(1)-x(1)-1; u(2)+x(2)-2; -x(1)+1; u(2)-1];
```



3. Control parametrization and state constraints

- $u(t) \in R^{n_u}, t_0 \leq t \leq t_f : n_u$ numbers at each time $t_0 \leq t \leq t_f$.
- There are infinitely many time instants $t_0 \leq t \leq t_f$
- Can we find a way to make the number finite?
- Control parametrization: approximate description of $u(t) \in R^{n_u}, t_0 \leq t \leq t_f$ by finitely many vectors $u_k \in R^{n_u}, k = 0, 1, \dots, N-1$
- Example of control parametrization: A piecewise constant control



- $u(t) = u(t_k) = u_k \in R^{n_u}, t_k \leq t \leq t_{k+1}, k=0,1,\dots,N-1, t_N = t_f.$

Figure:

$$t_0=0, t_1=0.5, \dots, t_{20}=10 : t_k = k*0.5, k=0,1,\dots,N, N=20$$

- Then $J(u(t)) \rightarrow J(u_k)$. Now $J(u_k)$ is a function of finitely many vectors: $u_k \in R^{n_u}, k=0,1,\dots,N-1$.
- In general $J(u(t)) \rightarrow J(u_k), u_k = u(t_k) \in R^{n_u}, k=0,1,\dots,N-1$



- Given the control parametrization $u_k \in R^{n_u}, k=0,1,...,N-1$
 can we compute $J(u_k)$?

- $u_k \in R^{n_u}, k=0,1,...,N-1 \rightarrow u(t), t_0 \leq t \leq t_f$

- Numerically integrate $\dot{x}' = f'$ from t_0 to t_f :

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n_x} \\ \dot{J}_r \end{bmatrix} = \begin{bmatrix} f_1(x,u,t) \\ f_2(x,u,t) \\ \vdots \\ f_{n_x}(x,u,t) \\ L(x,u,t) \end{bmatrix}, \quad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_x} \\ J_r \end{bmatrix} = \begin{bmatrix} x \\ J_r \end{bmatrix} = x', \quad \begin{bmatrix} f_1(x,u,t) \\ f_2(x,u,t) \\ \vdots \\ f_{n_x}(x,u,t) \\ L(x,u,t) \end{bmatrix} = \begin{bmatrix} f \\ L \end{bmatrix} = f'$$

$$J(u(t)) = J(u_k) = \phi(x(t_f)) + J_r(t_f) = \phi(x(t_f)) + x'_{n_{x+1}}(t_f) = \phi'(x'(t_f))$$

- Another result of this computation is $x(t_k), i=0,1,2,...,N$:
 the values of the state variables at finitely many time
 points $t_0 \leq t_k \leq t_f$ which depend (in a complicated but
computable way) on $u_k = u(t_k), i=0,1,...,N-1$
- So we can evaluate/compute state constraints at finitely
 many time-points $t_k, k=0,1,...,N$



```
% Exslec11: Script 2 lecture 11, Optimal control with state constraints
%
% Input:
%         cont : 0 : compute new solution
%              1 : continue with previous values
% Options vector opt for the function constr that performs the constrained
% function minimization
%         opt(1) : 1=display intermediate results
%         opt(13): number of equality constraints
%         opt(14): 500, maximum number of function evaluations
% GvW 21-2-2006
%
opt=[]; opt(1)=1; opt(13)=0; opt(14)=700;
uguess=0e-3*ones(2,10); % initial guess for control
if cont==1 % cont=1: continue with current values
    uguess=u;
end
lb=zeros(2,10); ub=5e-2*ones(2,10); % control lower and upper bounds
% Perform the constraint function minimization and return the solution x
u=constr('fsysl11',uguess,opt,lb,ub);
% Obtain the optimal control and optimal state
[f,g,tt,xt]=fsysl11(u);
% Plot the optimal state
figure(1); plot(tt,xt');
xlabel('Time [days]'); ylabel('states [tons, euros]'); title('Optimal x(t)');
% Plot the optimal control
figure(2); stair(0:365/10:9/10*365,u');
xlabel('Time [days]'); ylabel('control [per day]'); title('Optimal u(t)');

% fsysl11: Optimal control by function minimization example lecture 11
function [f,g,tt,xt]=fsysl11(u);

tt=[]; % vector to store time instants obtained from numerical integration
xt=[]; % matrix to store states obtained from numerical integration
xc=[]; % matrix to store states for evaluation of constraints

tf=365; K=10; x0=[4;2;0]; % Final time, number of time points, initial state
tk=0; % Initial time
for k=1:K
    tk1=tk+tf/K; % next time point
    [t,x]=ode45('anchsal',[tk tk1],x0,[],u(:,k)); % numerical integration
    tt=[tt; t]; xt=[xt; x]; % store time points and states
    tk=tk1; x0=x(end,:); % update time and the initial state
    xc=[xc x0]; % update state matrix for constraint evaluation
end;
f=x0(3); %Compute J
%g=[]; % no constraints or
g=[xc(1,:)-15]; %Compute/evaluate constraints: g(i,j)<=0 should hold

% Anchsal: Dynamic model and cost function of the Anchovis Salmon system
function [dx]=anchsal(t,x,flag,u);
p=[5e-2; 2e-2; 2e-2; 1e-2; 1e-1; 1.5e-1; 1e-3; 2.5e-3];
dx(1,1)=p(1)*x(1)-p(2)*x(1)*x(2)-u(1)*x(1);
dx(2,1)=-p(3)*x(2)+p(4)*x(1)*x(2)-u(2)*x(2);
dx(3,1)=-p(5)*u(1)*x(1)-p(6)*u(2)*x(2)+p(7)*u(1)+p(8)*u(2);
```



4. Free final time problems and time scaling

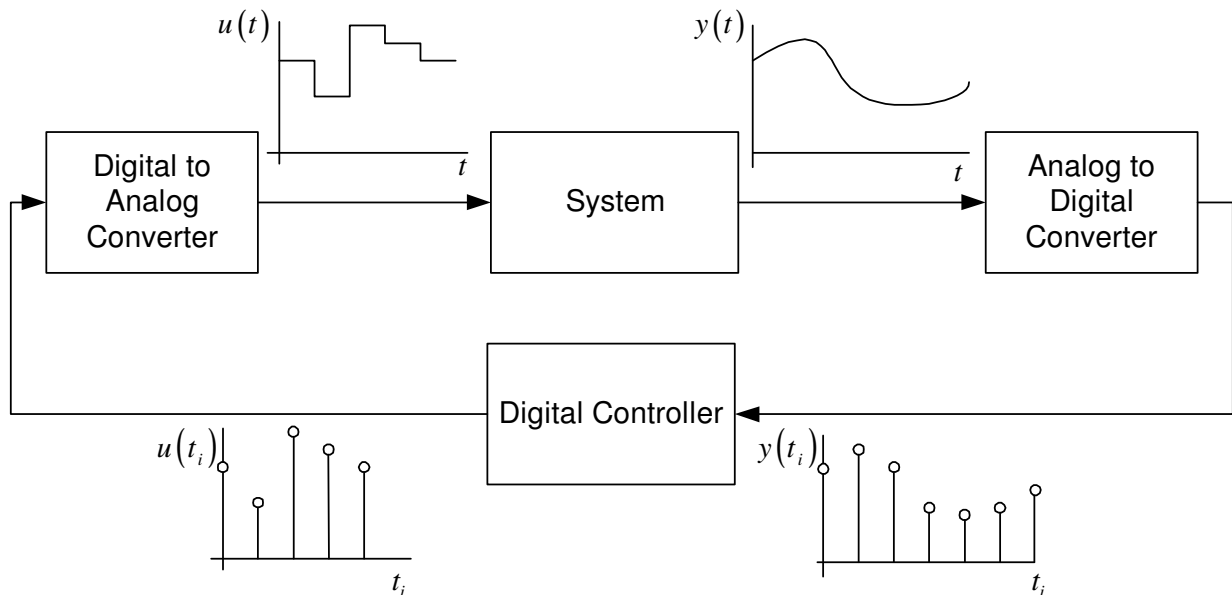
- New time variable $\tau = \frac{t}{t_f}$
- $0 \leq t \leq t_f \rightarrow 0 \leq \tau \leq 1$: fixed final time for τ .
- $\frac{d}{d\tau} = \frac{d}{d\frac{t}{t_f}} = t_f \frac{d}{dt}$
- $$\begin{bmatrix} dx_1/dt \\ dx_2/dt \\ \vdots \\ dx_{n_x}/dt \\ dJ_r/dt \end{bmatrix} = \begin{bmatrix} f_1(x,u) \\ f_2(x,u) \\ \vdots \\ f_{n_x}(x,u) \\ L(x,u) \end{bmatrix} \rightarrow \begin{bmatrix} dx_1/d\tau \\ dx_2/d\tau \\ \vdots \\ dx_{n_x}/d\tau \\ dJ_r/d\tau \end{bmatrix} = \begin{bmatrix} t_f * f_1(x,u) \\ t_f * f_2(x,u) \\ \vdots \\ t_f * f_{n_x}(x,u) \\ t_f * L(x,u) \end{bmatrix}$$
- $J(u(t), t_f) \rightarrow J(u(\tau), t_f) = J(u_k, t_f)$ with
 u_k a control parametrization of $u(\tau)$, $0 \leq \tau \leq 1$.
- Besides $u_k, i=1,2,...,N$ the final time t_f becomes a variable of the function $J(u_k, t_f)$ which is minimized.

5. Advantage and disadvantage of function minimization.

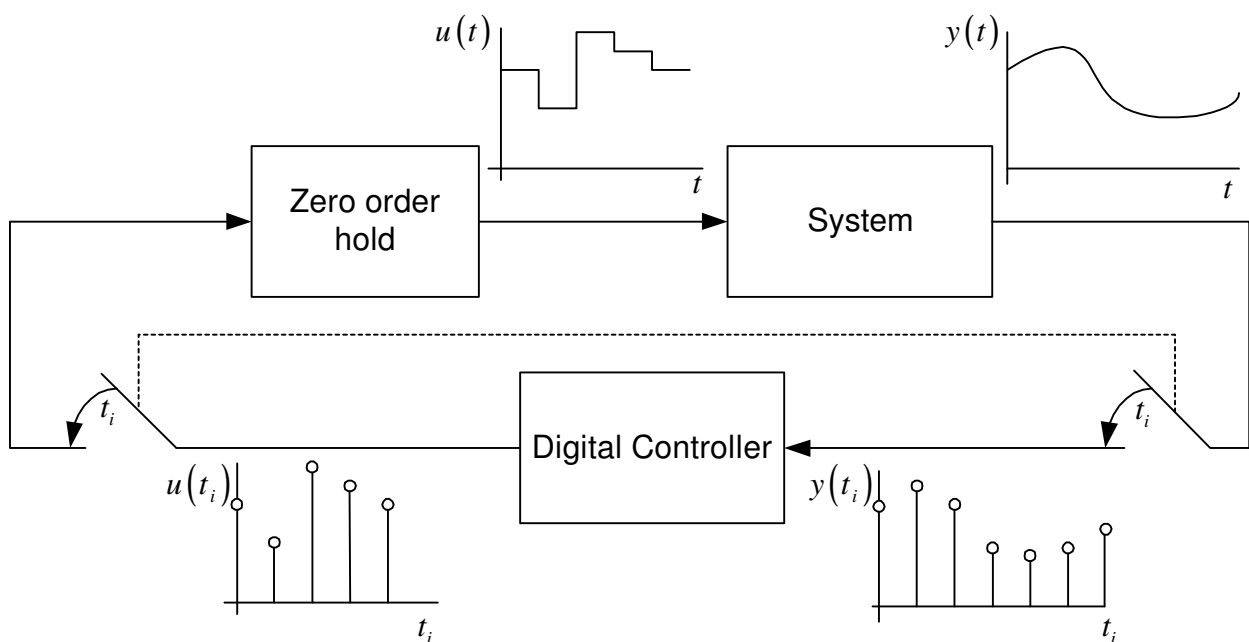
- Advantage: any type of optimal control problem can be handled
- Disadvantage: computationally less efficient if the dimensions of the state and/or the control and/or the control parameterization are large.



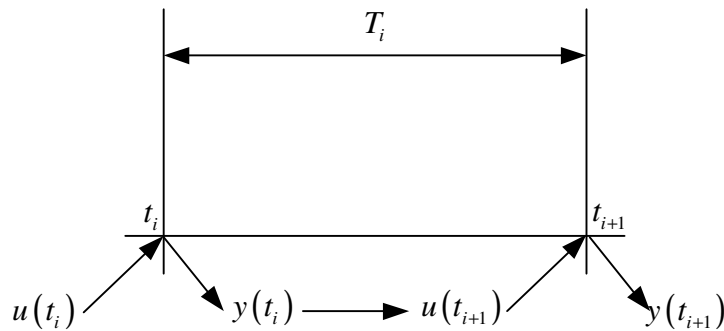
6. Digital control systems and control parametrization



A digital control system



Another representation of a digital control system



Task sequence of the digital controller

- $t_i, i=0,1,\dots,N$: Sampling instants
- $T_i = t_{i+1} - t_i, i=0,1,\dots,N-1$: Sampling intervals
- Digital control: $u(t) = u(t_i), t_i \leq t \leq t_{i+1}$, piecewise constant!
- Conventional sampling: $T_i = T, i=0,1,\dots,N-1$
- T : sampling interval or sampling time
- $1/T$: sampling frequency
- T small: continuous optim. contr. $\approx$ digital optim. contr.
- Advantage of applying digital control theory:
No small sampling time required which relaxes the computational demands on the digital controller



7. Practical exercise (non compulsory part computer exercise 2)

Program one Matlab script and two functions, similar to the example presented in this lecture, to compute the time-optimal bang-bang control for the car when it has to stand still at the finish line. The car should reach the finish as quickly as possible.

All the other data of this optimal control problem are identical to those in the compulsory exercise of lecture 8. There are no state constraints.

Note that the bang-bang control is completely parameterized by the initial value of the control (maximal or minimal) the switch times and the final time.

For the parameterization of the control assume:

- 1) There is only one switch time
- 2) The initial value for the control is maximal

Note that 1) and 2) follow from simple physical reasoning.

- a) With the assumptions 1) and 2) which two parameters must be optimized (found) to obtain the optimal bang-bang control?
- b) Do these two parameters have upper and lower bounds? If so determine these bounds.
- c) Using the answer to b) program and run the code and check the solution against the analytical one.

Systems and Control Theory Part 2: Optimal Control

Lecture 12.

Control and state parameterization by Chebyshev polynomials. Passively and actively adaptive optimal control.

L.G. van Willigenburg

Contents

1. Control and state parameterization by polynomials and other time functions
2. Control and/or state parameterization by Chebyshev polynomials
3. Substitution of possible solutions into differential equations
4. When models are not of sufficient quality: Passively and actively adaptive optimal controllers



**Wageningen University
Mathematical & Statistical Methods Group
(Biometris)**



1. Control and state parameterization by polynomials and other time functions

- Fourier analysis:

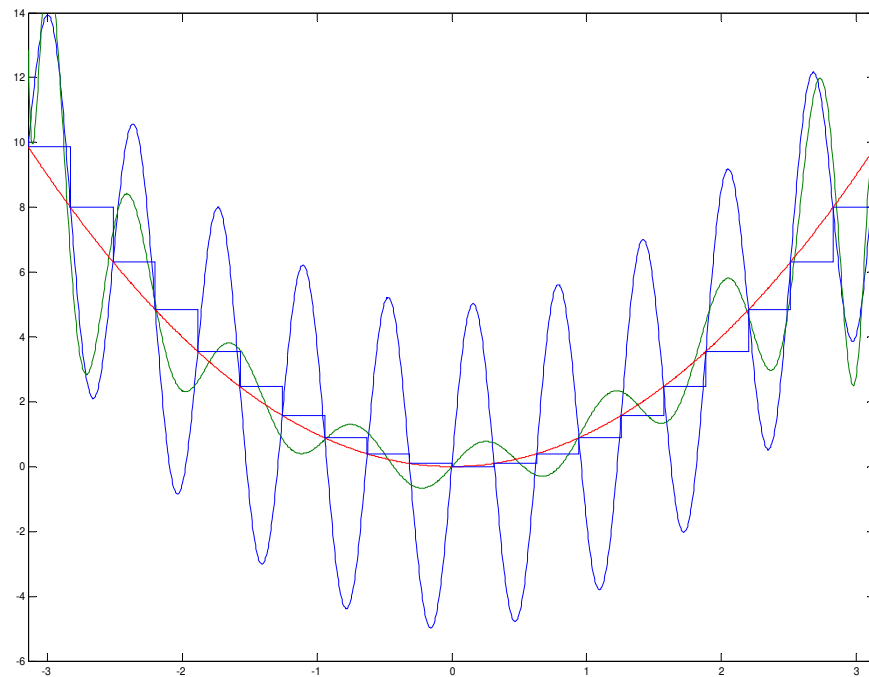
$$s(t) \approx \frac{1}{2}a_0 + \sum_{n=1}^N a_n \cos(n\omega_f t) + b_n \sin(n\omega_f t), \quad \omega_f = \frac{2\pi}{t_f}$$

A Fourier series approximation of $s(t)$, $0 \leq t \leq t_f$ by the $2N+1$ Fourier coefficients $a_0, a_1, \dots, a_N$ and $b_1, \dots, b_N$.

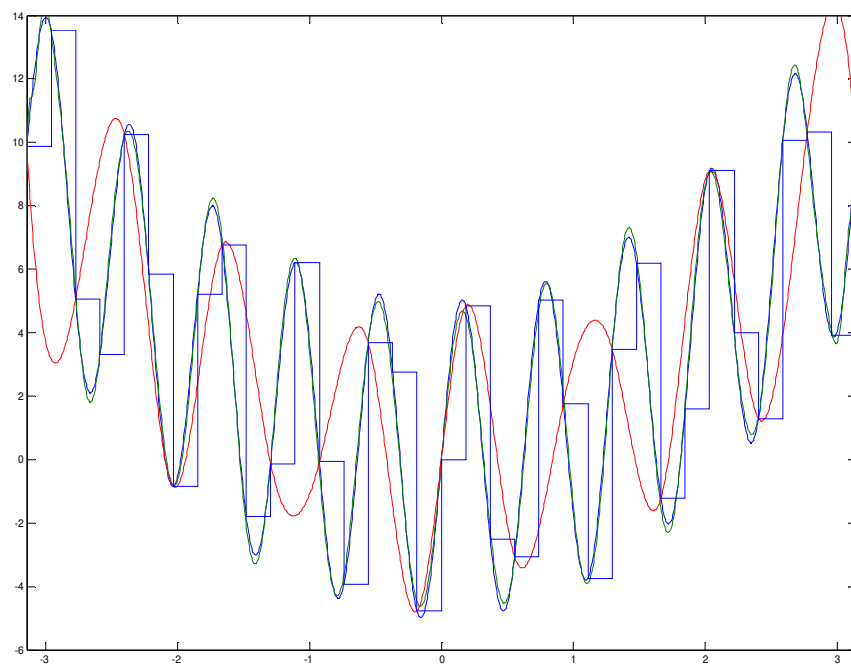
- Let $p_N(t), 0 \leq t \leq t_f$ denote a parameterization of $s(t)$, $0 \leq t \leq t_f$ by $N+1$ parameters called a parametrization of order N .
- Maximum error of $p_N(t)$: $e_{\max} = \max_t \left| s(t) - p_N(t) \right|$
- Which parameterization has the smallest maximum error?
- Answer : Chebyshev polynomial parameterizations.



N=20

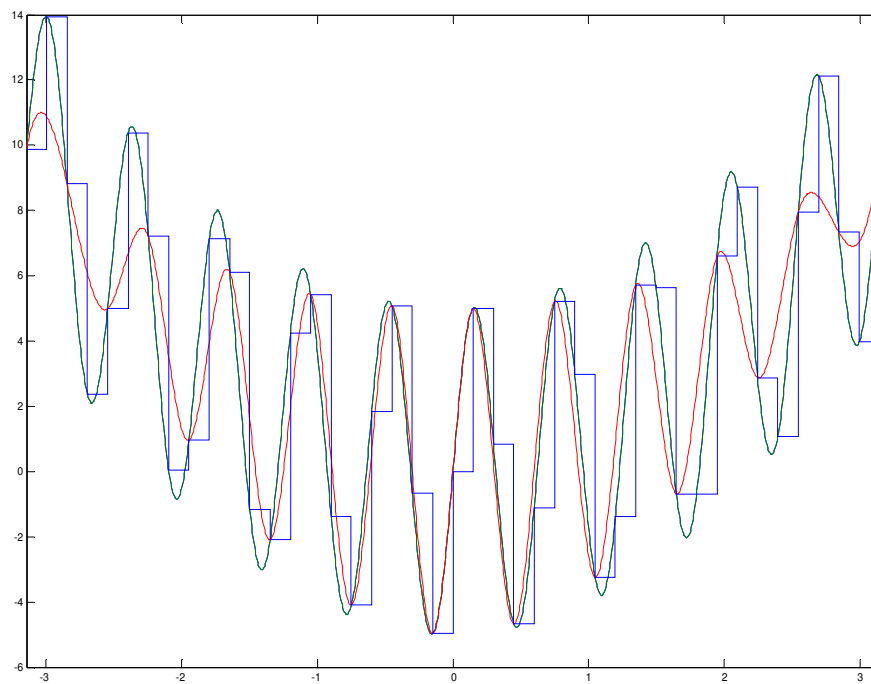


N=34

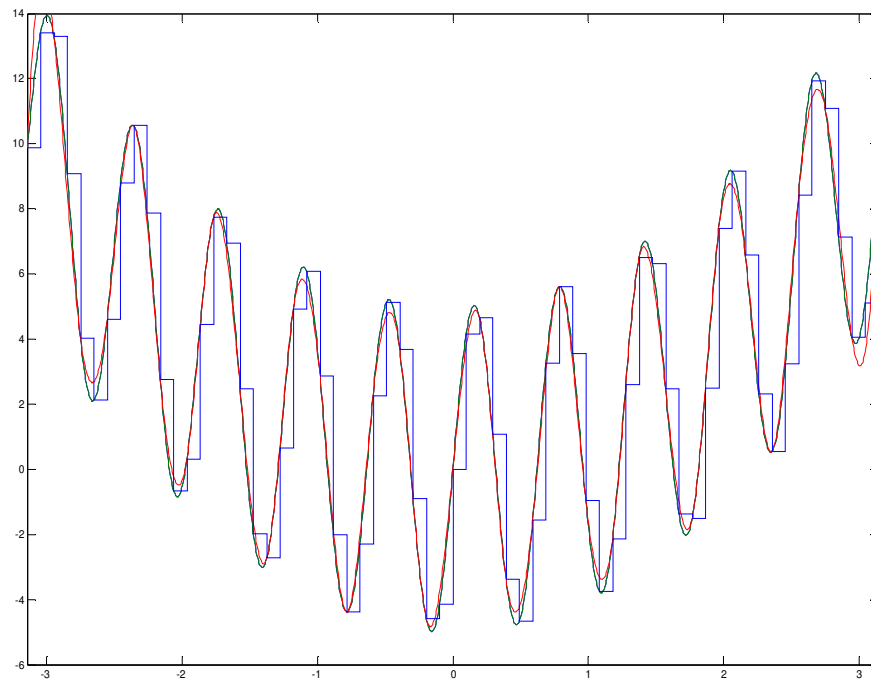




N=42



N=64





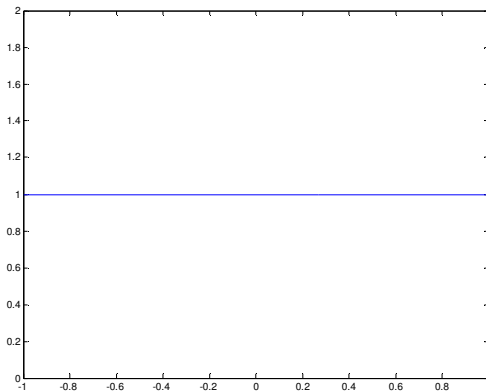
2. Control and/or state parameterization by Chebyshev polynomials

Denote a Chebyshev polynomial of order n by,

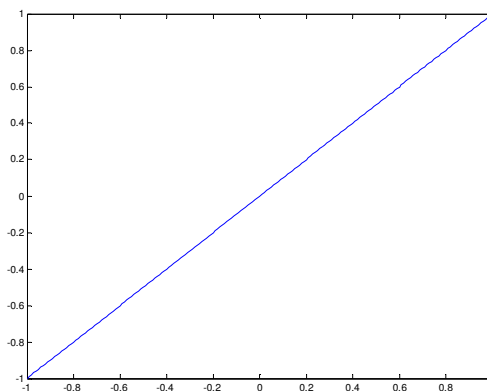
$$T_n(t), n=0,1,2,\dots, t \in [-1,1]$$

- Recursive definition of Chebyshev polynomials

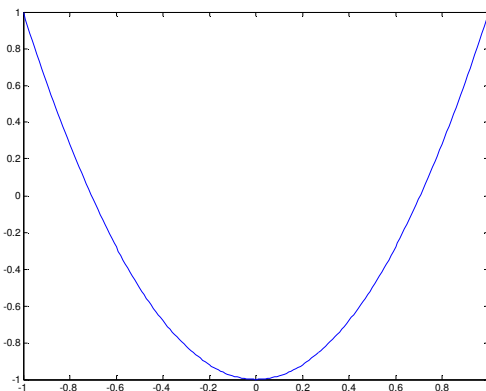
$$T_0(t)=1, T_1(t)=t, T_n(t)=2tT_{n-1}(t)-T_{n-2}(t), n=2,3,\dots$$



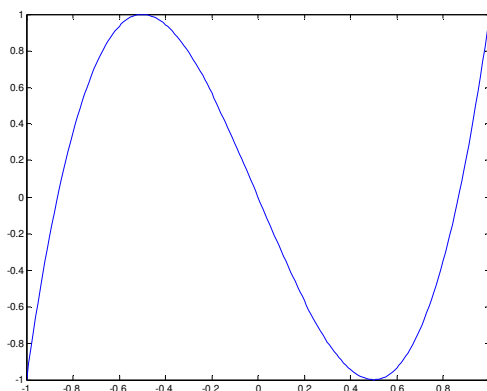
$$T_0(t)=1$$



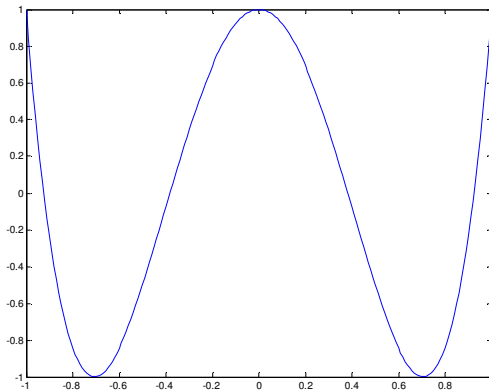
$$T_1(t)=t$$



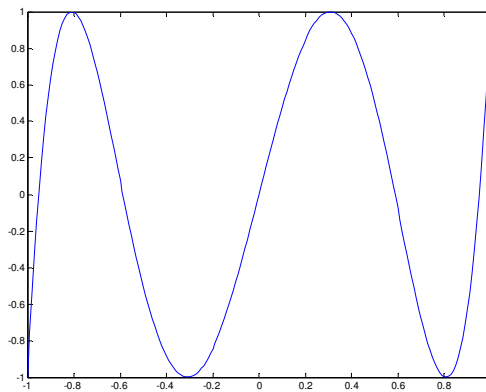
$$T_2(t)=2t^2-1$$



$$T_3(t)=4t^3-3t$$



$$T_4(t) = 8t^4 - 8t^2 + 1$$



$$T_5(t) = 16t^5 - 20t^3 + 5t$$

- A Chebyshev series approximation of order N:

$$p_N(t) = \frac{1}{2}a_0T_0(t) + \sum_{n=1}^N a_nT_n(t) = \frac{1}{2}a_0 + \sum_{n=1}^N a_nT_n(t) \approx s(t)$$

$a_0, a_1, \dots, a_N$: $N+1$ Chebyshev coefficients.

- Useful properties of Chebyshev polynomials

- $$\int_{-1}^1 p_N(t) dt = a_0 - \sum_{n=1}^N \frac{1 + (-1)^n}{n^2 - 1} a_n$$

Replaces numerical integration with an algebraic computation.

- If $\dot{p}_N(t) = \frac{dp_N}{dt}$, $-1 \leq t \leq 1$ has Chebyshev coefficients a'_n , $n=0, 1, \dots, N-1$ then

$$a'_{n-1} - a'_{n+1} - 2na_n = 0, n=1, 2, \dots$$

Turns first order differential equation constraints into algebraic constraints.



3. Substitution of possible solutions into differential equations

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_2 + u_1 x_4 \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= -x_4 + u_2 x_2\end{aligned}$$

$$J(u(t)) = \int_0^{t_f} \left[\left(x_1(t) - \cos(t) \right)^2 + \left(x_3(t) - \sin(t) \right)^2 \right] dt$$

- Can we achieve $J^*(u(t)) = 0$?

$$x_1(t) = \cos(t), x_3(t) = \sin(t),$$

$$\dot{x}_1(t) = -\sin(t) = x_2, \dot{x}_3(t) = \cos(t) = x_4$$

$$x_1(0) = 1, x_2(0) = 0, x_3(0) = 0, x_4(0) = 1$$

- Compute $u^*(t)$ by substitution in second and fourth differential equation:

$$-\cos(t) = \sin(t) + u_1^*(t) \cos(t) \xRightarrow{:\cos(t)} -1 = \frac{\sin(t)}{\cos(t)} + u_1^*(t) \Rightarrow u_1^*(t) = -1 - \frac{\sin(t)}{\cos(t)}$$

$$-\sin(t) = -\cos(t) + u_2^*(t) \cdot (-\sin(t)) \xRightarrow{:-\sin(t)} 1 = \frac{-\cos(t)}{-\sin(t)} + u_2^* \Rightarrow u_2^*(t) = 1 - \frac{\cos(t)}{\sin(t)}$$



4. When models are of insufficient quality: Passively and actively adaptive optimal controllers

- Our control system design is approximately optimal if the quality of the model is "sufficient".
- If the quality is insufficient what to do?
- Improve the quality of the model!
- By scientific research (first principles, off-line)
- Using the on-line measurements to improve the model (parameters) on-line: adaptive control
- Passively adaptive optimal control: The cost function only describes the control objectives not the need to identify the model parameters
- Actively adaptive optimal control: The cost function describes both the control objectives and the need to identify the model parameters.
- Actively adaptive optimal control: a new area of research

Be welcome to join us!

MAT-32306, Part 2, Computer exercise 1

Optimal heating of your living room at night

An interesting question concerns the best policy in terms of energy consumption to control the temperature in your living room at night when you are asleep. Suppose you go to bed at 0.00 hours and you rise at 7.00 hours. Also suppose the room temperature is at the desired level of $20\text{ }^{\circ}\text{C}$ when you go to sleep and you want it to be at exactly $20\text{ }^{\circ}\text{C}$ again at 7.00 hours in the morning. Suppose you want to find the heating policy that minimizes energy consumption by means of an optimal control computation. The maximal heating power of the heater equals $1000\text{ }[W] = 3600\text{ }[kJ\text{ }h^{-1}]$. A differential equation and the associated state-space model of the room temperature are represented by equations (8.1)-(8.3) in the lecture notes.

Part 1 (on paper)

- Use the mathematical model in state-space form derived during the lecture.
- Does this model have any external inputs? Explain.
- What are the initial conditions? Explain.
- Formulate a cost function. Explain.
- Is this a problem with a free final time? Explain.
- Is this a problem with terminal state constraints? If so specify these.
- Is this a problem with control constraints? If so specify these.
- Is this a problem with state constraints? If so specify these.

Part 2 (on the computer)

- From blackboard obtain the zip file scoc.zip that contains all the material for the practicals of part 2 of the course. On the computer create a folder m:\scoc. Unzip the file scoc.zip in this folder with preservation of the folder structure in the zip file. In the folder m:\scoc\ce1 you find the material needed for computer exercise 1.
- Startup Matlab and type: run m:\scoc\startoc. This adjusts the Matlab path and initializes some additional toolboxes required for optimal control, among them Tomlab. When startoc is finished give the Matlab command cd m:\scoc\ce1. In this folder you find everything regarding computer exercise 1.
- Use the Tomlab/PROPT optimal control toolbox to solve the optimal control problem described above numerically. To do this use the example file otcroom.m that you obtained. Open it and save it under the name otcnight.m so that you keep the original. Next modify the file to obtain the numerical solution. Assume a constant outside temperature of $10\text{ }^{\circ}\text{C}$.
- Play around with the 2 parameters in the model to see what their influence is on the solution. Try to give a physical explanation of these influences. Illustrate the results with Matlab plots and discuss these.

Part 3 (on the computer)

- Next implement a variable outside temperature $15 - 10\sin(\pi t / t_f)$, $0 \leq t \leq t_f$ in otcnight.m and compute the optimal control again.
- Change the variable outside temperature pattern and compute and try to physically explain the influence on the optimal control policy.

Part 4 (on the computer)

- o) Suppose you do not want the room temperature to fall below $15 \text{ } [^{\circ}\text{C}]$. How should this be incorporated in the optimal control problem formulation? Explain.
- p) Implement the condition mentioned under o) in `otcnight.m`. Compute and try to physically explain the changes to the optimal control policy. Illustrate the results with Matlab plots and discuss these.
- q) Play around with the minimum acceptable room temperature and compute and try to physically explain the influence. Illustrate the results with Matlab plots and discuss these.

Part 6) Before you leave notify the assistant. He will decide whether you completed the exercise successfully and make a note of this.

MAT-32306, Part 2, Computer exercise 2

Optimal sterilization / optimal bang-bang control

Compulsory part

- a) Using the same software and starting from the same example used in computer exercise 1 solve the sterilization problem described on sheet 10-11 of lecture 10.
- b) Play around with the optimal control problem data and see how they affect the optimal solution. Comment on the changes and try to explain them.
- c) Solve the optimal control problem concerning the car described on sheet 8-14 of lecture 8. Take the case where the car has to stand still at the finish. Remember that the optimal control should be bang-bang.
- d) Play around with the optimal control problem data and see how they affect the optimal solution. Comment on the changes and try to explain them.

Not compulsory

- e) Knowing the control is bang-bang we can more accurately solve this optimal control problem by treating it as a **function minimization problem** for which standard software is available. One such Matlab function is **constr**. Type **help constr** to find out how to use it. To solve the optimal control problem using **constr** note the following:
 - I. Only two **variables** need to be found: The time t_s at which the control switches from maximum to minimum and the final time t_f .
 - II. Given certain values of these two variables, by means of **numerical integration** we can compute the value of the cost function that needs to be minimized.
 - III. To guarantee that the final time $t_f \geq t_s$ optimize t_s and $t_f - t_s$ instead and specify the lower bounds $0 \leq t_s$, $0 \leq t_f - t_s$.
 - IV. Use the Matlab function **ode45** for numerical integration. Type **help ode45** for more information.

Part 2, computer exercise 3: Optimal control of a laboratory helicopter set up.

1. Introduction	1
2. Part A: Optimal control computation and simulation (4 hours)	5
2.1 Preparation part A	5
2.2 Information to be supplied by the control engineer.....	5
2.3 The optimal control software.....	6
2.4 The dynamic model and bounded control of the helicopter.....	6
2.5 The control objectives (requirements) and the initial state.	7
2.6 Solution of the optimal control problem using Tomlab.....	7
2.7 Programming exercise part A (4 hours)	8
3. Part B: Optimal output feedback computation and optimal control experiments (4 hours).....	9
3.1 Preparation part B	9
3.2 Modified control objectives (requirements) and initial state.	9
3.3 Optimal LQ state feedback design.....	9
3.4 Optimal state estimator design	11
3.5 Computation and implementation of the optimal control and the dynamic optimal LQG output feedback controller.....	12
4. Practical report.....	13

1. Introduction

Using optimal control software, in this exercise you will design compute and implement an optimal control system that uses optimal dynamic LQG output feedback. The to be controlled system is a laboratory helicopter set up represented by Fig. 1 and Fig. 2 on the next page. The helicopter has two rotational degrees of freedom represented by the angles θ_1 and θ_2 . Two propellers generate forces F_1 and F_2 that actuate the helicopter. You will perform experiments with the optimal control system to verify and improve its performance. Part A of this exercise is performed at morning (afternoon) one Part B at morning (afternoon) two. A short report has to be written which focuses on the choice of the design parameters within the optimal controller. Their expected and actual effect on the control system performance has to be discussed.

You have to prepare both parts of the computer exercise. If you have not prepared you are not allowed to start the programming exercise.

Questions that have to be answered on paper during the preparation are typed in this current letter format

Actions that have to be performed during the programming exercises are typed in this current letter format.

Fig. 3 and Fig. 4 show schematic representations of Part A and Part B of this computer exercise.

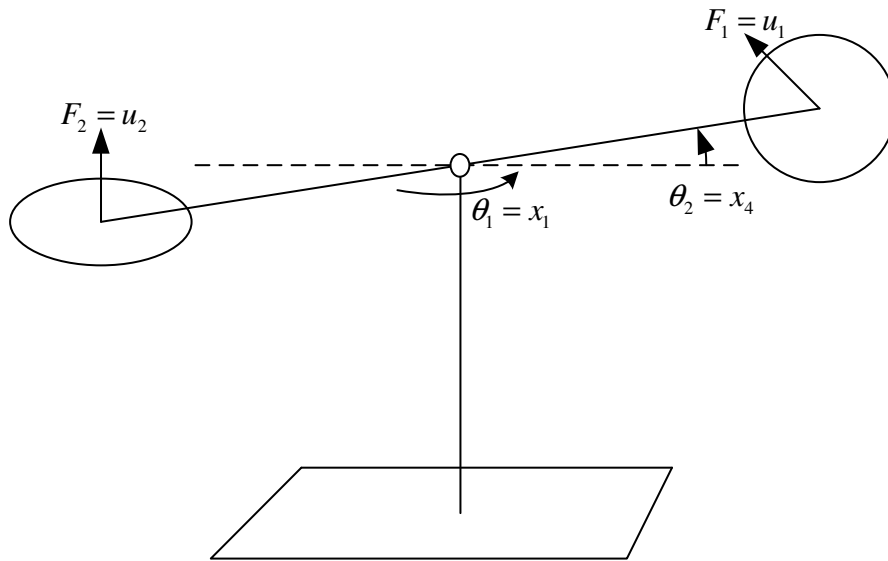


Fig. 1: Schematic representation of the helicopter



Fig. 2: Helicopter setup

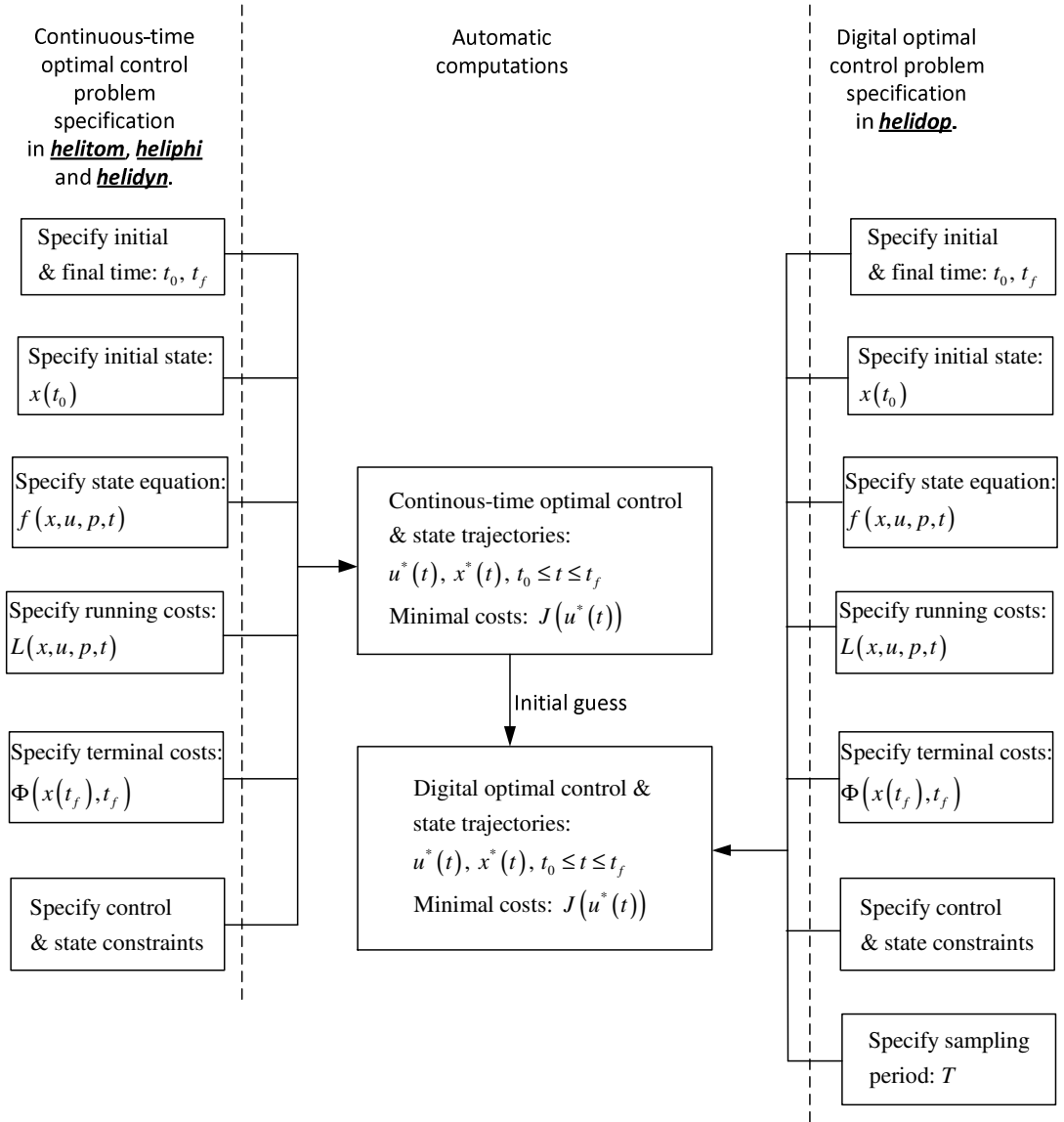


Fig. 3: Schematic representation Part A.

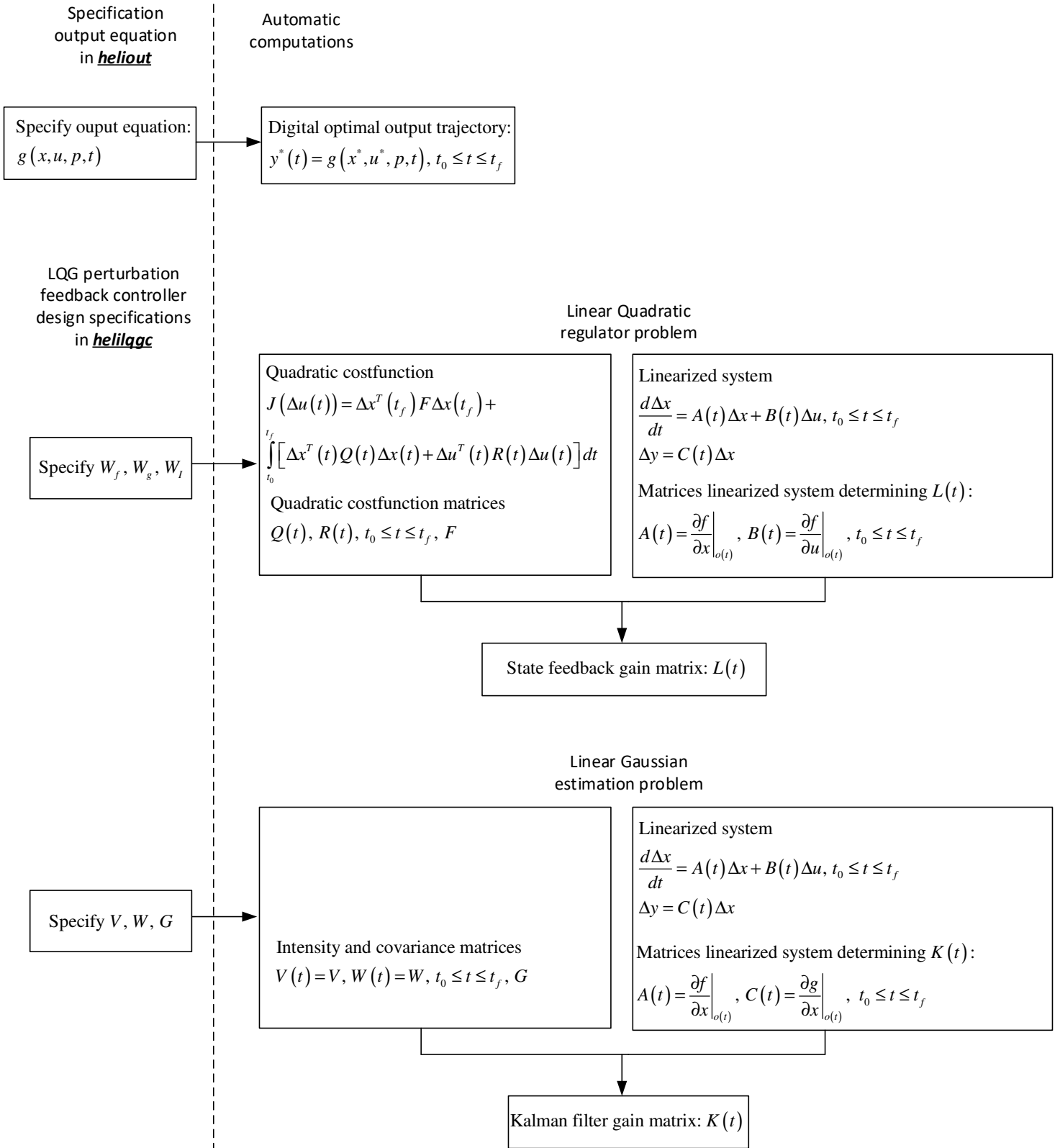


Fig. 4: Schematic representation Part B.

2. Part A: Optimal control computation and simulation (4 hours)

2.1 Preparation part A

Before you start the programming exercise of part A, described in section 2.7, you should read sections 2.1-2.7 of this guide and answer on paper any question. The questions relate to the translation of the control objectives, mentioned in section 2.5, into a cost function and an initial state. Section 2.7 contains the actual programming exercise of part A.

2.2 Information to be supplied by the control engineer.

The optimal control system which uses output feedback is represented by figure 1.

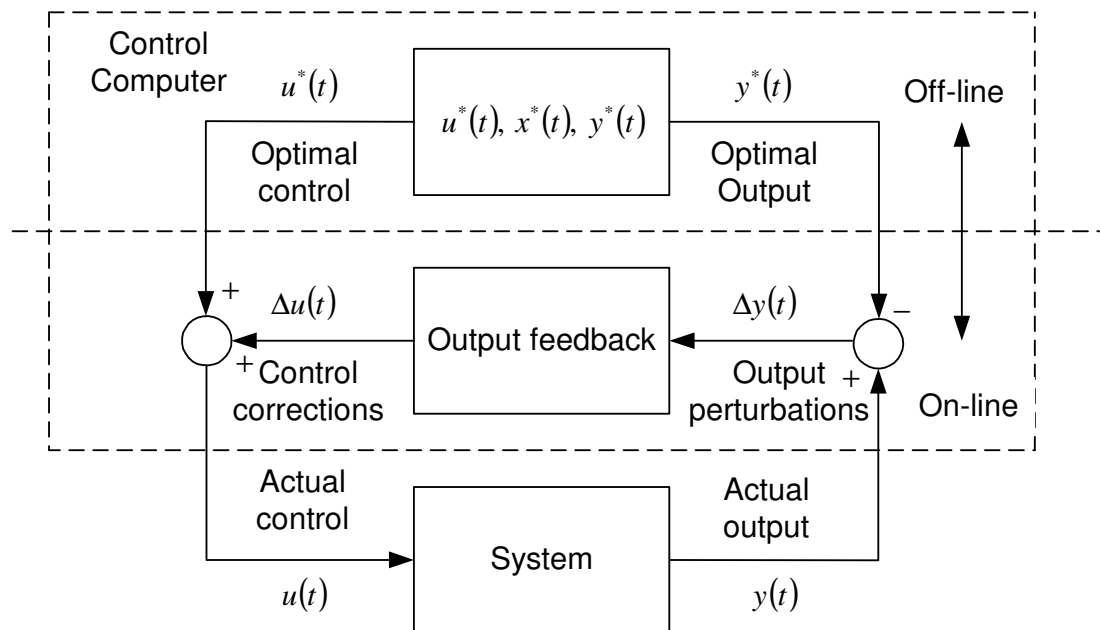


Figure 1: The optimal control system

From figure 1 and the lectures it should be clear that the control engineer has to supply the following information to design, compute and implement an optimal control system that uses optimal dynamic LQG output feedback.

- 1 A state-space model of the helicopter.
- 2 The initial state of the helicopter
- 3 A cost function that reflects the control objectives (requirements).
- 4 A quadratic cost function to design the optimal LQ state-feedback part.
- 5 A description of model and measurement uncertainty through intensity and covariance matrices of the measurement and system noise to design the state perturbation estimator (observer).
- 6 The sampling time (interval).

When the information 1-6 is supplied properly to the optimal control software all computations required for the implementation of the optimal control system are

performed automatically! So the emphasis of this exercise is on selecting and improving this information. The state-space model and the initial state of the helicopter will be given. So in this exercise we focus mainly on the selection and improvement of 3-6.

2.3 The optimal control software.

The software consists of the following four parts.

- a) Matlab software that employs state and control parametrization by means of Legendre polynomials to solve continuous-time optimal control problems with general constraints. It is part of the commercial Tomlab/Propt optimal control toolbox.
- b) A Matlab function developed by GvW to solve digital optimal control problems with general constraints. It uses an NLP solver together with a computation of the gradient by means of the costate and automatic differentiation.
- c) A Matlab function developed by GvW that:
 - 1) computes (simulates using numerical integration) the optimal system behavior $x^*(t)$ and the optimal output $y^*(t)$ $t_0 \leq t \leq t_f$ knowing the digital optimal control $u^*(t)$ obtained from b). The Matlab program uses the state equation and output equation specified to solve both a) and b).
 - 2) computes the dynamic LQG output feedback that is $L(t)$ and $K(t)$, $t_0 \leq t \leq t_f$. $L(t)$ is computed based on the matrices $Q(t), R(t), F$ which determine the quadratic cost function of the LQ problem and the linearized model about the optimal solution which is computed automatically. $K(t)$ is computed based on $V(t), W(t), G$ the matrices which describe the model, measurement and initial state errors (uncertainties) respectively and again the linearized model about the optimal solution which is computed automatically.
 - 3) Performs a Monte Carlo simulation (that is several simulations with different random errors in the measurements and state equation) to determine the average deviation of the costs and the possible instability of the control system.
- d) A Matlab control program developed by GvW that supplies the stored optimal control $u^*(t)$ to the real system and also computes on-line the optimal dynamic output feedback based on the stored values $y^*(t), L(t), K(t)$ and the on-line measurements $y(t)$, $t_0 \leq t \leq t_f$.

2.4 The dynamic model and bounded control of the helicopter.

The following dynamic model in state-space form describes approximately the helicopter behavior.

$$\dot{x}_1 = x_2 \quad (1)$$

$$\dot{x}_2 = -0.2368 x_2 + 2.8665 (x_3)^2, \quad x_3 \geq 0 \quad (2)$$

$$\dot{x}_2 = -0.2368 x_2 - 3.7639 (x_3)^2, \quad x_3 < 0 \quad (3)$$

$$\dot{x}_3 = 5.67(u_1 - x_3) \quad (4)$$

$$\dot{x}_4 = x_5 \quad (5)$$

$$\dot{x}_5 = -0.19 x_5 - 1.6818 \sin(x_4) + 2.2355 (x_6)^2, \quad x_6 \geq 0 \quad (6)$$

$$\dot{x}_5 = -0.19 x_5 - 1.6818 \sin(x_4) - 3.1297 (x_6)^2, \quad x_6 < 0 \quad (7)$$

$$\dot{x}_6 = 5.67(u_2 - x_6) \quad (8)$$

$$y_1 = x_1 \quad (9)$$

$$y_2 = x_4 \quad (10)$$

In equation (1) x_1 represents the horizontal helicopter rotation in radians. Similarly x_4 represents the vertical helicopter rotation in radians. Furthermore x_2, x_5 represent respectively the horizontal and vertical angular velocity in radians per second. Finally x_3, x_6 are measures of the propellor speeds of respectively the one that steers the horizontal and vertical rotation. u_1, u_2 are the control variables which are setpoints for these propellor speeds respectively. Equations (4) and (8) describe that changes in the setpoint for each propellor take a little time to be realized. Finally equations (9), (10) describe that the horizontal and vertical angular rotation are measured.

The control inputs u_1, u_2 of the helicopter are bounded from above and below,

$$-1 \leq u_1 \leq 1, -1 \leq u_2 \leq 1 \quad (11)$$

2.5 The control objectives (requirements) and the initial state.

Starting at time zero, during 10 seconds, the horizontal helicopter rotation should follow as closely as possible a cosine function with a frequency of 0.1 Hz and an amplitude of 0.8. The vertical helicopter rotation should follow as closely as possible a cosine function with a frequency of 0.1 Hz and amplitude of 0.5.

Choose the initial state to match these cosine functions and translate these control objectives into a cost function. Write both the initial state and cost function on paper with an explanation.

Not compulsory

Suppose the initial state matches the cosine functions. Also suppose that the differential equations (4) and (8) are permanently in equilibrium: $dx_3/dt=0$, $dx_6/dt=0$. Then you can compute by hand the optimal control using the remaining four differential equations in a similar manner as described in section 12.3 (pages 66-67) of the lecture notes. Perform this computation on paper.

2.6 Solution of the optimal control problem using Tomlab.

Consider the following optimal control problem. Given the system,

$$\dot{x}_1 = 0.1x_1 + x_2 + 0.1u_1 \quad (12)$$

$$\dot{x}_2 = -x_2 + u_2 \quad (13)$$

with initial state,

$$x(0) = \begin{bmatrix} 2 \\ -0.5 \end{bmatrix} \quad (14)$$

find the bounded control,

$$-2 \leq u_i(t) \leq 2, \quad i = 1, 2 \quad (15)$$

that minimizes the cost function,

$$J(u(t)) = x_1^2(t_f) + x_2^2(t_f) + \int_{t_0}^{t_f} (0.2x_1^2 + 0.2x_2^2 + 0.1u_1^2 + 0.1u_2^2) dt \quad (16)$$

where $t_f = 5$. Furthermore the following terminal state inequality constraints apply,

$$x_1(t_f) \leq -0.3, x_2(t_f) \geq -0.3 \quad (17)$$

Finally the following state inequality constraints apply,

$$x_1 \leq 0.5, 2.5 \leq t \leq t_f, x_2 \geq -0.65, 0 \leq t \leq t_f \quad (18)$$

In the folder `ce3` you find the matlab script ***lqtom*** that solves the above LQ problem with constraints in continuous-time. Read it to see if you understand the code. Note that the state-space model and the terminal costs of the constraint LQ problem are described in separate files. Which files? Check these files to see if they match the LQ problem data.

The associated **digital optimal control** problem with a constant sampling interval of 0.5 is solved by means of function minimization. The code that has to be supplied by the user you find in the matlab script ***lqdop***. The state-space model, the terminal cost and the output of the system are described in separate files. Check these files to see if they match the LQ problem data.

2.7 Programming exercise part A (4 hours)

- 1) Install the optimal control software as indicated in the ***readme.doc*** file that is in the root of the zipped file ***scoc.zip*** that you find on blackboard.
- 2) Startup Matlab as indicated and then type `cd m:\scoc\ce3` to enter this folder. Modify the example files ***lq*.m*** so that they deal with the helicopter control problem described in sections 2.4, 2.5. To do this it is convenient to first copy the ***lq*.m*** files to ***heli*.m***. Then edit ***helidyn.m*** (use ***signsc.m***), ***heliphi.m*** and ***helitom.m*** to compute and save in a datafile the continuous-time optimal control.
- 3) With ***helitom***, under 2), you solved the continuous-time version of the optimal control problem. The helicopter however is controlled by a digital controller. Therefore we will now solve the digital version of the optimal control problem using a piecewise constant (digital) control and function minimization. The digital version of the LQ problem is solved by ***lqex.m*** that calls ***optconstr.m*** to compute the digital optimal control. ***Opconstr.m*** calls ***lqdop.m*** that contains all the problem data. Modify ***heliex.m*** (was ***lqex.m***) and ***helidop.m*** (was ***lqdop.m***) to solve the digital version of the optimal control problem of the helicopter. A very good initial guess of the solution can be obtained by averaging the continuous-time optimal control over each sampling period. Use the matlab function ***pwaver.m*** to do this and speed up the digital optimal control computation. Take the sampling interval to be $0.5[\text{sec}]$.

3. Part B: Optimal output feedback computation and optimal control experiments (4 hours)

3.1 Preparation part B

Read and study sections 10.3, 10.4 of the lecture notes. Read sections 3.2-3.5 below and answer on paper all the questions. The programming exercises are spread over sections 3.2-3.5.

3.2 Modified control objectives (requirements) and initial state.

Because it is difficult to realize initial states other than,

$$x(t_0) = x(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (19)$$

we change the initial state into this value. Furthermore in the horizontal direction instead of a cosine function with amplitude 0.8 and frequency 0.1 Hz. we want to move as closely as possible according to a sine function with the same amplitude and frequency. Finally the sampling interval is changed into 0.1 [sec.]

- 1) Modify `helidyn`, `heliphi`, `helitom`, `helidop` and `heliex` according to these changes and rerun the matlab code to compute the continuous-time and digital optimal control.

3.3 Optimal LQ state feedback design

The computation of the optimal dynamic LQG output feedback controller comes down to the computation of $L(t)$ the optimal static state feedback gain matrices and the computation of $K(t)$ the optimal state estimator gain matrices.

The optimal static state feedback gain $L(t)$ is computed from the following LQ regulator problem. The system is the linearized system about the optimal trajectory $u^*(t)$, $x^*(t)$ that will be denoted by the shorthand $o(t)$.

$$\Delta \dot{x}(t) = A(t) \Delta x(t) + B(t) \Delta u(t), \quad \Delta y(t) = C(t) \Delta x(t) \quad (20)$$

where $A(t)$, $B(t)$, $C(t)$ are computed by automatic differentiation,

$$A(t) = \left. \frac{\partial f}{\partial x} \right|_{o(t)}, \quad B(t) = \left. \frac{\partial f}{\partial u} \right|_{o(t)}, \quad C(t) = \left. \frac{\partial g}{\partial x} \right|_{o(t)} \quad (21)$$

The linearized system (20), (21) about $u^*(t)$, $x^*(t)$, denoted by $o(t)$ in (21), describes approximately the dynamic behavior of the state perturbations $\Delta x(t)$, the output perturbations $\Delta y(t)$ and the control corrections $\Delta u(t)$. The quadratic cost function is,

$$J(\Delta u(t)) = \Delta x^T(t_f) F \Delta x(t_f) + \int_{t_0}^{t_f} (\Delta x^T(t) Q(t) \Delta x(t) + \Delta u^T(t) R(t) \Delta u(t)) dt \quad (22)$$

In the lecture notes, section 10.4, and in reference [1] of the lecture notes, it is argued that $F \geq 0, Q(t) \geq 0, R(t) > 0$ should be chosen such that (22) is a kind of upper bound of 1) the magnitude of the second order terms determining the errors due to linearization and 2) the deviations of the optimal costs. The second order terms that determine largely the error due to linearization are,

$$\Delta x^T(t) \frac{\partial^2 f_i}{\partial x^2} \bigg|_{o(t)} \Delta x(t), i = 1, 2, \dots, n_x, \Delta u^T(t) \frac{\partial^2 f_i}{\partial u^2} \bigg|_{o(t)} \Delta u(t), i = 1, 2, \dots, n_x \quad (23)$$

$$\Delta x^T(t) \frac{\partial^2 g_i}{\partial x^2} \bigg|_{o(t)} \Delta x(t), i = 1, 2, \dots, n_y, \Delta u^T(t) \frac{\partial^2 g_i}{\partial u^2} \bigg|_{o(t)} \Delta u(t), i = 1, 2, \dots, n_y \quad (24)$$

The second order term determining largely deviations from the optimal cost are,

$$\Delta x^T(t) \frac{\partial^2 H}{\partial x^2} \bigg|_{o(t)} \Delta x(t), \Delta u^T(t) \frac{\partial^2 H}{\partial u^2} \bigg|_{o(t)} \Delta u(t), \Delta x^T(t_f) \frac{\partial^2 \phi}{\partial x^2} \bigg|_{o(t_f)} \Delta x(t_f) \quad (25)$$

An upper bound (22) of (23)-(25) is computed by our software using a method described in reference [7] of the lecture notes. This method results in nonnegative diagonal matrices $F \geq 0, Q(t) \geq 0, R(t) \geq 0$:

$$Q(t) = W_f \frac{1}{n_x} \sum_{i=1}^{n_x} D^0 \left(\frac{\partial^2 f_i}{\partial x^2} \bigg|_{o(t)} \right) + W_g \frac{1}{n_y} \sum_{i=1}^{n_y} D^0 \left(\frac{\partial^2 g_i}{\partial x^2} \bigg|_{o(t)} \right) + W_l D^0 \left(\frac{\partial^2 H}{\partial x^2} \bigg|_{o(t)} \right) \quad (26)$$

$$R(t) = W_f \frac{1}{n_x} \sum_{i=1}^{n_x} D^0 \left(\frac{\partial^2 f_i}{\partial u^2} \bigg|_{o(t)} \right) + W_g \frac{1}{n_y} \sum_{i=1}^{n_y} D^0 \left(\frac{\partial^2 g_i}{\partial u^2} \bigg|_{o(t)} \right) + W_l D^0 \left(\frac{\partial^2 H}{\partial u^2} \bigg|_{o(t)} \right) \quad (27)$$

$$F = W_l D^0 \left(\frac{\partial^2 \phi}{\partial x^2} \bigg|_{o(t_f)} \right) \quad (28)$$

In equation (26) $D^0 \left(\frac{\partial^2 f_i}{\partial x^2} \bigg|_{o(t)} \right)$ is a nonnegative diagonal matrix upperbounding the

matrix $\frac{\partial^2 f_i}{\partial x^2} \bigg|_{o(t)}$ that has the same dimensions. To compute

$F \geq 0, Q(t) \geq 0, R(t) \geq 0$ in (26)-(28) three weighting factors W_f, W_g, W_l have to be specified. W_f weighs the relative importance of minimizing the magnitudes of the second order terms (23) relating to the error due to linearization of the state equation. W_g weighs the relative importance of minimizing the magnitudes of the second order terms (24) relating to the error due to linearization of the output equation. Since the output equation is linear all terms in (24) are zero. Therefore the choice of W_g is irrelevant and W_g is taken to be zero. W_l weighs the importance of minimizing the magnitude of the deviations (25) from the optimal costs.

Explain why equation (27) results in $R(t) = 0$.

$R(t) = 0$ implies that there is no penalty on $\Delta u(t)$. As a result $\Delta u(t)$ may grow very large. For two reasons this is undesirable. The control $u(t)$ is bounded and therefore $\Delta u(t)$ cannot grow large in practice, although the LQG design expects this. Secondly and more importantly, a too large $\Delta u(t)$ may result in unstable behavior because of modeling errors that are present. The LQG output feedback only performs properly if there is also some penalty on $\Delta u(t)$. Therefore we have to add to $R(t) = 0$ a diagonal matrix with positive diagonal elements, to prevent $\Delta u(t)$ from becoming too large. For the moment we select $R(t)$ to be the identity matrix.

Note that the LQG perturbation output feedback controller gain $L(t)$ depends only on the *ratios* of the diagonal elements of $Q(t), R(t), F$. Similarly the Kalman filter gain $K(t)$ depends only on the *ratios* of the diagonal elements of V, W, G . Explain why.

3.4 Optimal state estimator design

The optimal estimator gain $K(t)$ is computed from a linear Gaussian estimation problem where the system is again the linearized system described by equation (21). However in addition to (21) we also require the linearized output equation,

$$\Delta y(t) = C(t) \Delta x(t), \quad C(t) = \left. \frac{\partial g}{\partial x} \right|_{o(t)} \quad (29)$$

where $C(t)$ is also computed automatically by automatic differentiation of the output equation (9), (10). The estimation problem is dual to the LQ regulator problem and requires specification of the matrices $V \geq 0, W > 0, G \geq 0$ which are the dual versions of Q, R, F in the LQ regulator problem. V, W are respectively the intensity matrices associated with the errors $\Delta \dot{x}(t)$ (observe the dot!) and $\Delta y(t)$.

G is the covariance matrix associated with the error $\Delta x(0)$. If V, W, G are diagonal matrices their diagonal elements represent the average squared errors in the associated component of $\Delta \dot{x}(t)$, $\Delta y(t)$ and $\Delta x(0)$. Assume that the measured rotations of the helicopter have an average absolute error of 0.01 and that the average absolute error of the angular velocity computed by the model is 0.1. Also assume that the average absolute error of the angular acceleration computed by the model is 0.33 while the absolute error of the propeller acceleration computed by the model is 0.033. Finally assume that the initial state is errorless.

Based on these assumptions select the matrices V, W, G . Write the result on paper together with a motivation.

Having selected the optimal LQG output feedback design matrices Q, R, F (through W_f, W_l) and V, W, G the optimal static state feedback and estimator gains $L(t), K(t)$ can be computed by solving the two associated LQ regulator problems.

3.5 Computation and implementation of the optimal control and the dynamic optimal LQG output feedback controller

- 2) Edit the matlab function **heliout** so that it describes the output equations (9), (10).
- 3) Edit the matlab function **helilqgc** which contains the information required for the LQG output feedback controller computation as well as the Monte Carlo simulations. As explained earlier take $W_g=0$, $R=I$ by specifying $Wg=0$; $Ra=eye(m)$ in **helilqgc**. Remember that without Ra , $R(t)=0$. Furthermore specify the matrices V, W, G (V , W , G in **helilqgc**). Set $W_f=1$, $W_l=5$ in **helilqgc**.
- 4) Use the matlab function **lqgcomp** to compute the LQG output feedback controller and perform the Monte Carlo simulations.
- 5) From the Monte Carlo simulations the average deviations of the optimal costs (ΔJ) and the number of times the control system becomes unstable (control system failure) are computed. To obtain *reliable* averages set nmc (number of runs Monte Carlo simulation) in between 20 and 50 in **helilqgc**. Recall that the LQG perturbation output feedback controller gain $L(t)$ depends only on the *ratios* of the diagonal elements of $Q(t), R(t), F$. Therefore it depends only on the *ratio* of W_f and W_l . Change W_l into a vector with different values around 5. Then for all W_f , W_l combinations an LQG output feedback controller is computed and associated Monte Carlo simulations are performed. Based on the outcome select the best single value for W_f and W_l .
- 6) Set W_f and W_l equal to their best single value and set nmc back to 1 in **helilqgc**. Run **lqgcomp** once more to obtain "the best" LQG perturbation output feedback controller.

From sheet 7-9 the state estimator differential equation can also be written as,

$$\Delta \dot{\hat{x}}(t) = (A(t) - B(t)L(t) - K(t)C(t))\Delta \hat{x}(t) + K(t)\Delta y(t) = A_e(t)\Delta \hat{x}(t) + K(t)\Delta y(t) \quad (30)$$

So to implement the optimal control system we need storage of,

- The optimal control trajectory $u^*(t)$
- The optimal output $y^*(t)$
- The optimal state feedback gain matrices $L(t)$
- The optimal estimator gain matrices $K(t)$
- The estimator matrices $A_e(t) = (A(t) - B(t)L(t) - K(t)C(t))$

- 7) Having executed steps 1)-6) above we now have a data file `comp.mat` that contains the above vectors and matrices associated with "the best LQG perturbation output feedback controller". Copy this data file on a USB stick and sign in the assistant. He will go with you to the laboratory helicopter set up. The assistant will show and explain to you the use of Matlab to execute the optimal control experiment. Relevant data of the experiment are stored in the file `compout.mat`. **Copy this file back onto the USB stick.** Executing the Matlab script `heliplot` enables you to plot relevant results of the experiment based on the data in files `comp.mat` and `compout.mat`.

Observe and discuss with the assistant the behavior of the optimal control system and next try to improve the performance by modifying several design variables. To that end you may use the Matlab function `plmarr`. It allows you to plot elements of the matrices $Q(t)$ and $R(t)$ against time. To use them you must first load into Matlab `edorp.mat` and `digcomp.mat`. These are files containing data used during computation of the output feedback.

4. Practical report

Since this seems no longer common knowledge to many students let me state here explicitly that a scientific report should motivate all choices made during the (control system) design by linking them to the associated theory presented during this course. Furthermore the report concerning the helicopter control assignment should answer all the questions that were part of the preparation. Also the report should explain, discuss and illustrate by appropriate figures:

- 1) The continuous time optimal open loop control (costfunction & optimal input & state trajectories & minimum costs) from both part A and part B.
The digital optimal open loop control (costfunction & optimal input & state trajectories & minimum costs) from both part A and part B.
- 2) The digital optimal LQG feedback controller design of part B; specifically the selection of the feedback design parameters and their expected and actual influence on
 - a. The Monte Carlo simulations
 - b. The actual control of the helicopter.
- 3) The expected and actual effect of modifying one or more optimal control system design parameters.

Non compulsory: At the end of the report you may provide me with (positive and negative) feedback on part 2 of the course together with suggestions for improvement. This part will (of course) not be judged.