# Static and Dynamic Optimization (42111)

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

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

2019-10-21 21:59

Lecture 7: Free dynamic optimization (D)

# Outline of lecture

- What is Dynamic Optimization
- Dynamics - examples
- Cost functions - examples
- Free dynamics optimization
- Euler-Lagrange equations
- Example: optimal stepping
- Practicalities
- Exercise DO.1
- Reading: DO: p. 5-11, 15-27, Appendix B

DTU

# Dynamic Optimization

What is Dynamic Optimization?

Dynamic Optimization has 3 ingredients:

- Some dynamics. Here (in this course) a state space model.
- A performance index (cost function, objective function, rewards, index function) to be optimized (minimized or maximized). In our case it is a summation (or integral) of contribution over a period of time of fixed or free length (part of the optimization).
- Eventually some constraints (on the decisions or on the states)

# Problem statement (Free dynamic (D) optimization)

**Task:** Find a sequence of optimal decisions $(u_i \; i = 0, \; \ldots \; N-1)$.

**Dynamics** (described by a state space model):

$$x_{i+1} = f_i(x_i, u_i) \qquad i = 0, \; 1, \; \ldots \; N-1 \qquad\qquad x_0 = \underline{x}_0$$

**Objective:** (to minimize or maximize the index):

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

- Here $N$ and $\underline{x}_0$ are fixed (given),
- $J$, $\phi$ and $L$ are scalars.
- $x_i$ and $f_i$ are $n$-dimensional vector and vector function
- and $u_i$ is a $m$-dimensional vector of decisions.

Notice: no constraints (except given by the dynamics) i.e. Free Dynamic Optimization.

# Ex1: Optimal Pricing (simplified)

We are producing a product (brand A) and have to determine its price in order to maximize our income.

There is a competitor product B - and a problem.

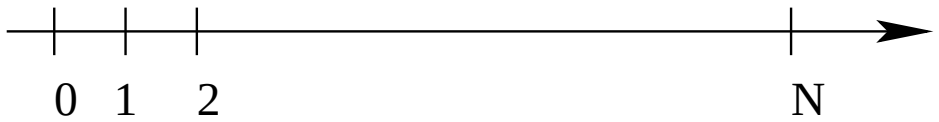
If we are to modest we might have almost all the costumers - but we will not earn that much.

If we are to greedy then the bulk majority of the costumers will bye the other brand B - and we will lose money.

We devide the period into $N$ intervals and going for a price profile $(u_i,\ i = 0,\ \ldots\ N-1)$.

# Optimal Pricing - the performance index

We have to decide the price of the product $u_i \sim \underline{u}$ ($\underline{u}$ being the production cost) in each interval.



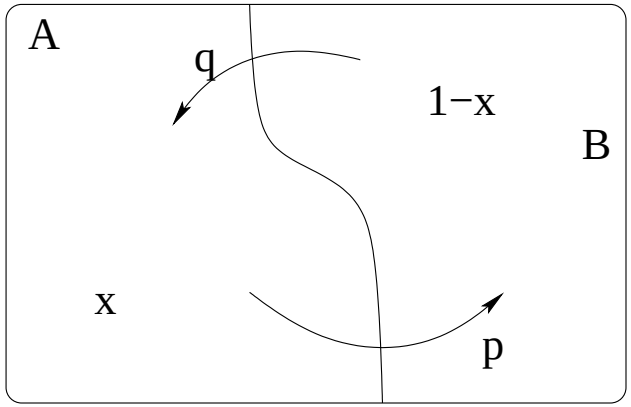

Let $M$ be the size of the market and let $x_i$ $(0 \leq x_i \leq 1)$ be the (A) share of the marked in the i'th interval.

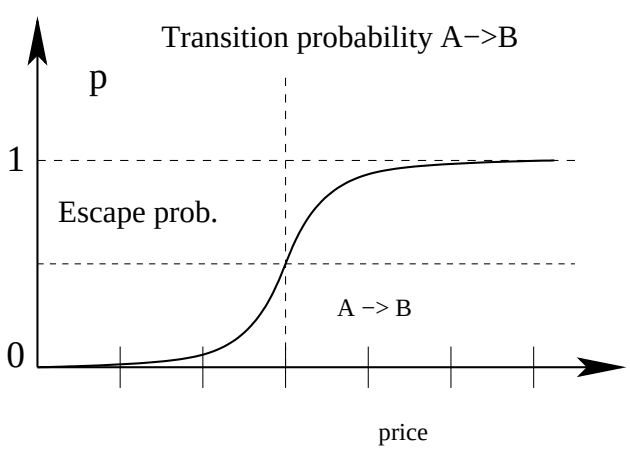
Objective: to make some money - i.e. to maximize

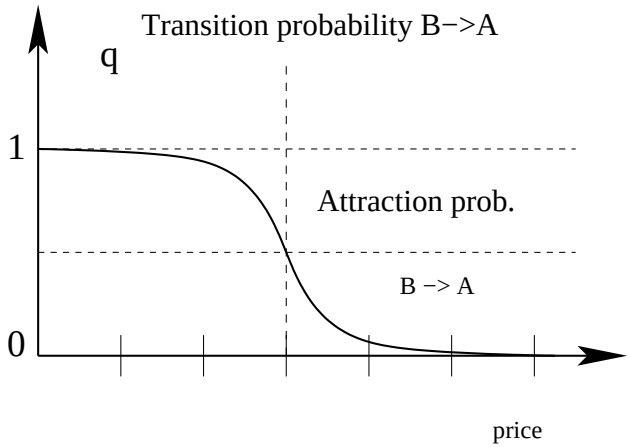
$$Max\ J \quad \text{where} \quad J = \sum_{i=0}^{N-1} M\bar{x}_i(u_i - \underline{u}) \qquad \bar{x}_i = \frac{1}{2}(x_i + x_{i+1})$$

More precisely, $x_i$ is the marked share at the beginning of interval $i$ and $\bar{x}_i$ is the average share of the marked in interval $i$ $(i = 0,\ \ldots\ N-1)$.

# Optimal Pricing - the dynamics




Dynamics: A→A B→A

$$x_{i+1} = \big(1 - p[u_i]\big)x_i + q[u_i]\big(1 - x_i\big) \qquad x_0 = \underline{x}_0$$

# Recap Optimal Pricing

Dynamics:

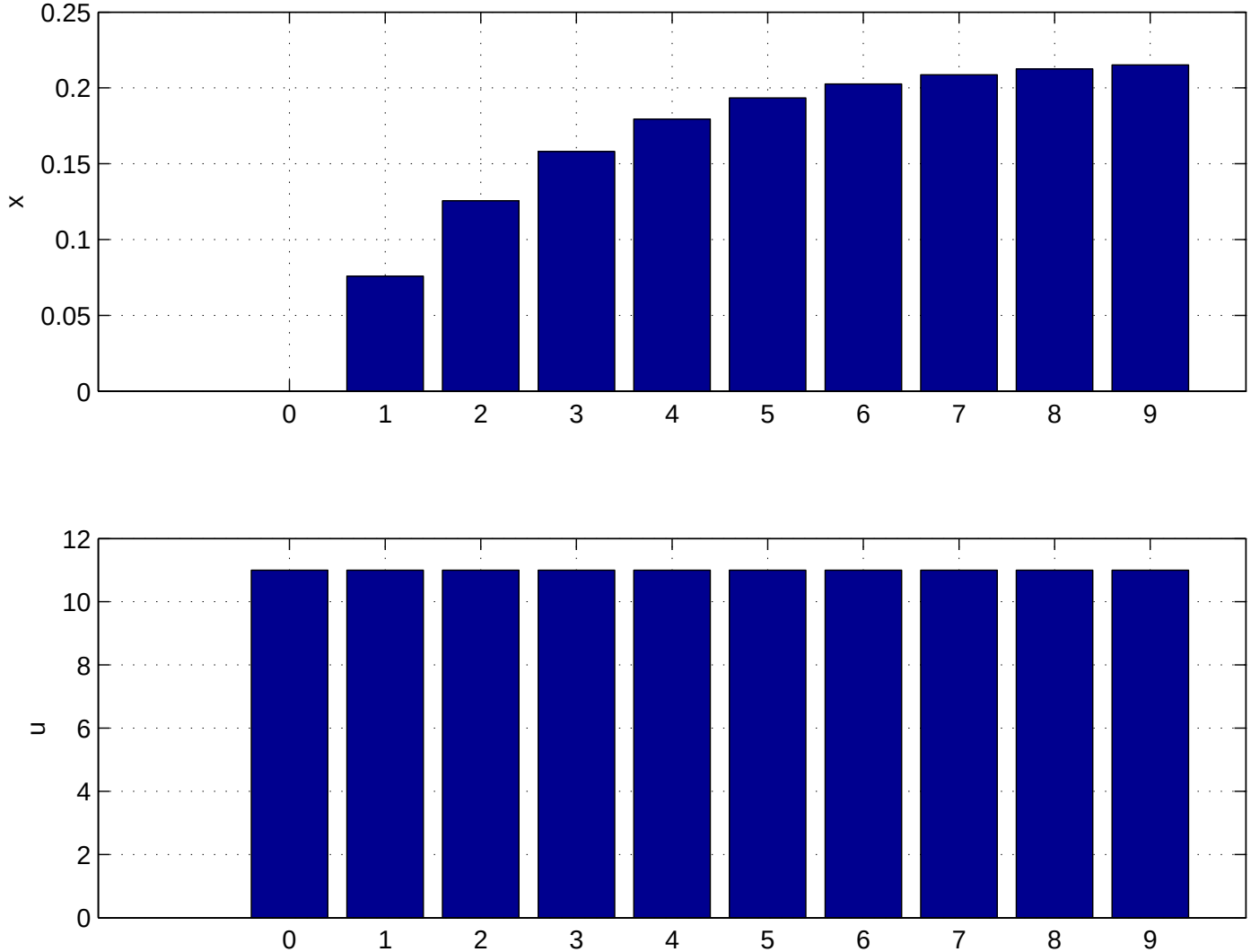
$$x_{i+1} = \big(1 - p(u_i)\big)x_i + q(u_i)\big(1 - x_i\big) \qquad\qquad x_0 = \underline{x}_0$$

Objective:

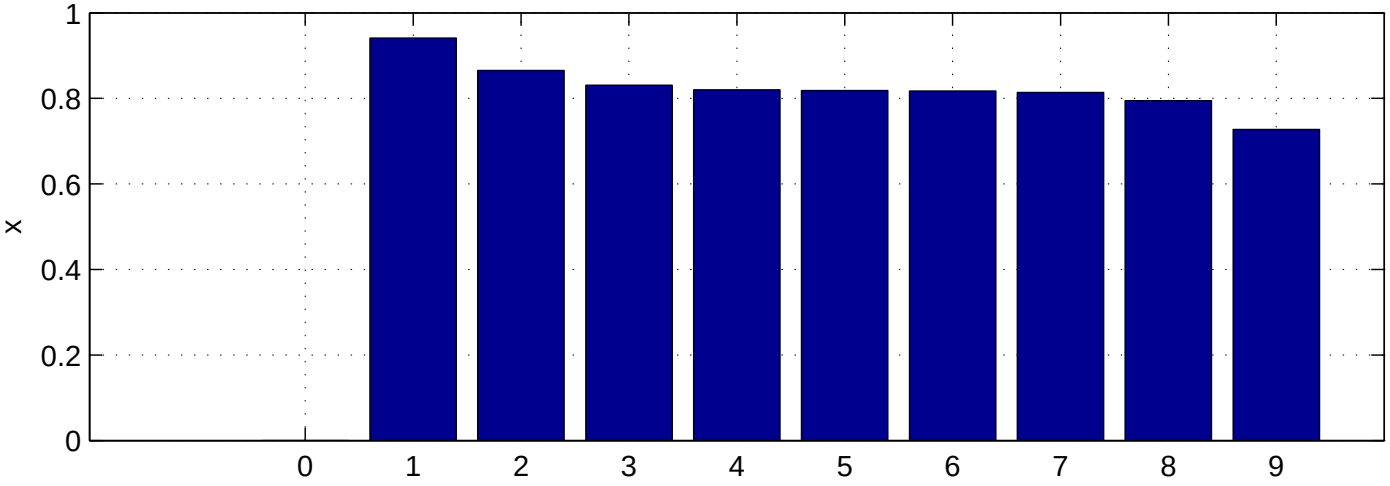
$$Max\ J \quad \text{where} \quad J = \sum_{i=0}^{N-1} M\bar{x}_i\big(u_i - \underline{u}\big)$$

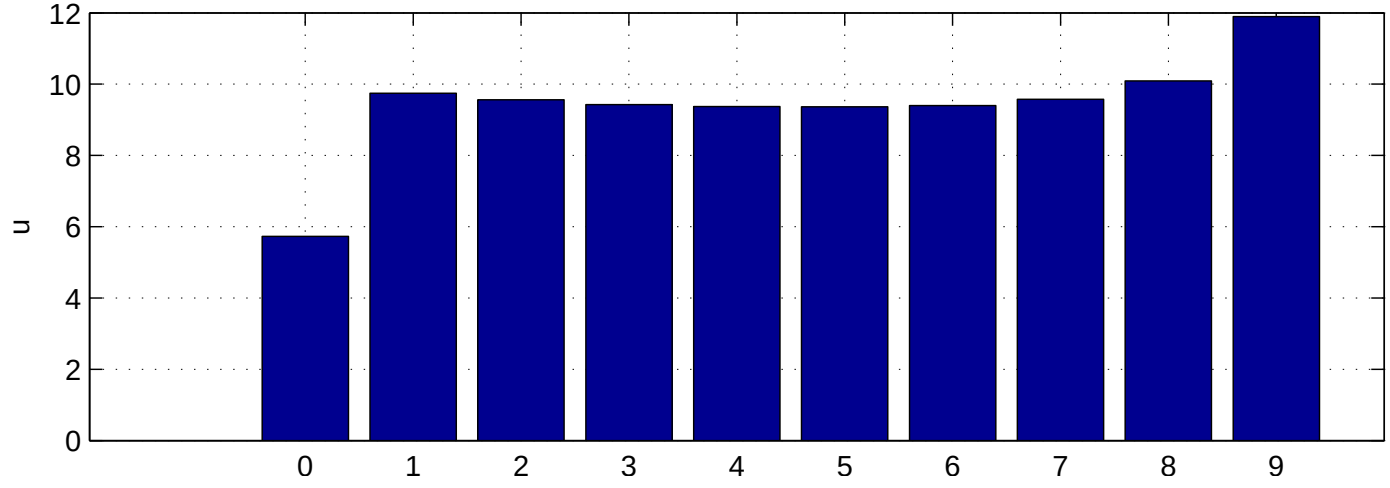
Notice: This is a discrete time model. No constraints. The length of the period (the horizon, $N$) is fixed.

If $u_i = \underline{u} + 5$ ($\underline{u} = 6$, $N = 10$) we get $J = 8$ (rounded to integer).



Optimal pricing (given correct model): $J = 27$ (rounded to integer).




Notice different axis for x.

# Free Dynamic Optimization

Dynamics (described by a state space model):

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

Objective (to optimize the index):

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

Here $N$ and $\underline{x}_0$ are fixed (given), $J$, $\phi$ and $L$ are scalars. $x_i$ and $f_i$ are $n$-dimensional vector and vector function and $u_i$ is a vector of decisions.

Notice: no constraints (except given by the dynamics).

# Example: Optimal pricing

In this case:

The dynamic is given by:

$$f_i(x_i, u_i) = \big(1 - p(u_i)\big)x_i + q(u_i)\big(1 - x_i\big)$$

where $\underline{x}_0 = 0$ and $N = 10$.

---

The objective function (performance index) is characterized by:

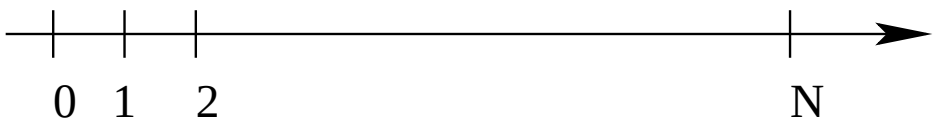
$$L_i(x_i, u_i) = M\bar{x}_i\big(u_i - \underline{u}\big) \qquad \phi_N(x_N) = 0$$

$$\bar{x}_i = \frac{1}{2}\big[x_i + x_{i+1}\big] \qquad \text{where} \qquad x_{i+1} = \big(1 - p[u_i]\big)x_i + q[u_i]\big(1 - x_i\big)$$

# Dynamics

Dynamics (described by a state space model):

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



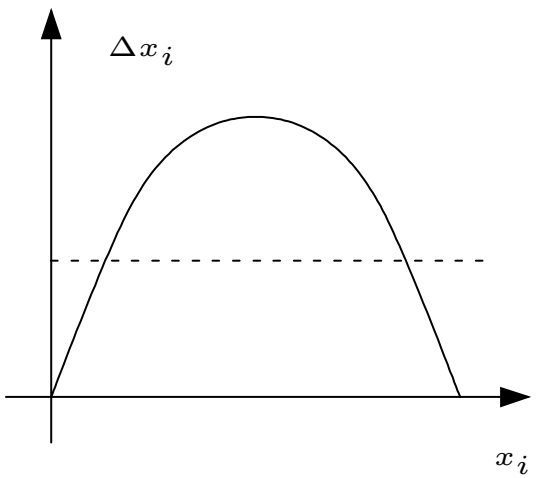

Let us have a look at some other examples:

# Example: The Schaefer model

Common growth model in connection to biological systems. E.g. (re)production of fish.

| | | |
|---|---|---|
| State | $x_i$ | Biomass (mass of fish) in a given area and at the beginning of interval $i$. |
| Decision | $u_i$ | Mass of fish to be catch in interval $i$. |
| Parameters | $r$ and $\alpha$ | |

$$x_{i+1} = x_i + r x_i \left(1 - \alpha x_i\right) - u_i \qquad x_0 = \underline{x}_0 \qquad i = 0,\ 1,\ \ldots\ N$$



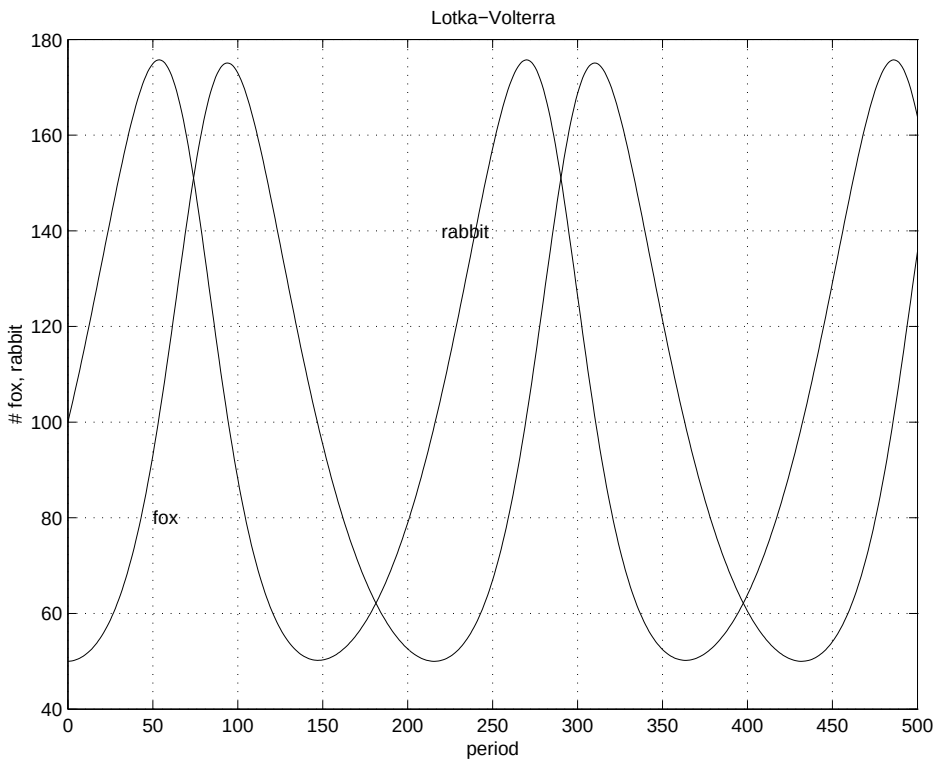

$$\Delta x_i = x_{i+1} - x_i$$

This is a first order (one state), but an non-linear model. Stationarity and Stability.

# Example: The Lotka-Volterra model

A Prey-Predator (the Lotka-Volterra (1925) model or the rabbit[r]-fox[F]) model).

Dynamics:
$$\begin{bmatrix} r_{i+1} \\ F_{i+1} \end{bmatrix} = \begin{bmatrix} r_i \\ F_i \end{bmatrix} + \underbrace{\begin{bmatrix} \alpha_1 r_i \\ \beta_2 r_i F_i \end{bmatrix}}_{\text{birth}} - \underbrace{\begin{bmatrix} \beta_1 r_i F_i \\ \alpha_2 F_i \end{bmatrix}}_{\text{death}} - \underbrace{\begin{bmatrix} u_i \\ v_i \end{bmatrix}}_{\text{hunt}}$$




A nonlinear, **second** order model with oscillations.

# Dynamics

Dynamic (discrete time) state space model:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{i+1} = f_i\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_i, \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix}_i\right); \qquad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_0 = \begin{bmatrix} \underline{x}_1 \\ \underline{x}_2 \\ \vdots \\ \underline{x}_n \end{bmatrix}_0$$

or in short:

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

$$f : \mathbb{R}^{n+m+1} \to \mathbb{R}^n$$

---

The fox-rabbit example:

$$\begin{bmatrix} \mathbf{r} \\ \mathbf{F} \end{bmatrix}_{i+1} = \begin{bmatrix} \mathbf{r_i} + \alpha_1 \mathbf{r_i} - \beta_1 \mathbf{r_i F_i} - u_i \\ \mathbf{F_i} + \beta_2 \mathbf{r_i F_i} - \alpha_2 \mathbf{F_i} - v_i \end{bmatrix}$$

# The State Space concept

**The state space model:**

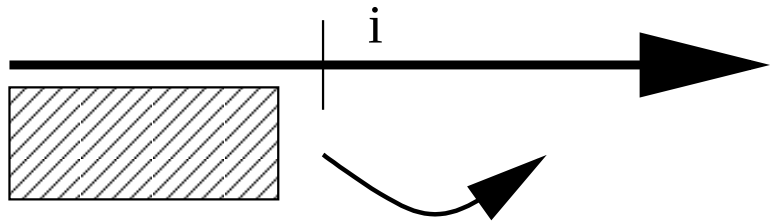
$$x_{i+1} = f_i(x_i, u_i) \qquad x_0 = \underline{x}_0$$

- State $x_i \in \mathbb{R}^n$: vector containing quantities describing the situation (the state) of the system.

  They contain information of the system history (is a sufficient statistics of the history). No historical data is needed for determining future state values ($x_\tau$ where $\tau > i$) if present state vector $x_i$ is known. Notice, this is a Markov property.

- Input $u_i \in \mathbb{R}^m$: vector containing the inputs.

  Might be control inputs or decision variables. It might be the inputs from an opponent or simply a disturbance. Here (for the time being) , we are going to determine a sequence of optimal inputs.



**Linear Time invariant (LTI) systems**

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{i+1} = A \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_i + B \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix}_i$$

or in short

$$x_{i+1} = Ax_i + Bu_i$$

---

An example:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{i+1} = \begin{bmatrix} 1 & 0 \\ 0.5 & 0.8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_i + \begin{bmatrix} 0.3 \\ 0.2 \end{bmatrix} u_i \qquad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

## A glimpse of stability

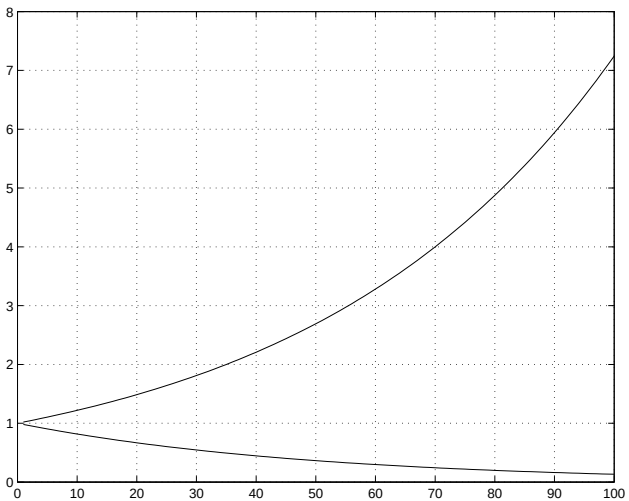
Common sense definition: States do not grow unlimited.

**Example 1:** Unattended bank loan (r=0.02) :

$$x_{i+1} = [1 + r]x_i \quad x_0 = \underline{x}_0$$

**Example 2:** Controlled loan with a down payment equals $u_i = 0.04x_i$

$$x_{i+1} = [1 + r]x_i - u_i = 0.98x_i \quad x_0 = \underline{x}_0$$



# Stability - the LTI case

First order (LTI) system (ie. $x_i$, $a$ are scalars)

$$x_{i+1} = ax_i$$

is (asymptotic) stable if $|a| < 1$.

---

A LTI system

$$x_{i+1} = Ax_i$$

is (asymptotically) stable if all eigenvalues of $A$ is (strictly) inside the unit circle (length of eigenvalue less than one).

# The Objective function


$$J = \phi[x_N] + \sum_{i=0}^{N-1} L_i(x_i, u_i) \qquad \in \mathbb{R}$$

- scalar function(s)
- consists of additive local terms (additve seperable terms)
- terminal contribution $\phi[x_N]$
- running cost, stage cost, interval cost $L_i(x_i, u_i)$

# Quadratic cost

**Example: Minimum Energy/Power**

Often the square of the state or control variable is related to energy:

$$E_p = Ri^2 \quad (current) \qquad E_p = \frac{u^2}{R} \quad (voltage)$$

$$E = \frac{1}{2}mv^2 \quad (moving\ mass) \qquad E = \frac{1}{2}kx^2 \quad (spring)$$

---

Control system such that

$$J = \sum_{i=0}^{N-1} x_i^2$$

is minimized.

---

Often it is compromise between objectives.

$$J = \sum_{i=0}^{N-1} x_i^2 + \rho u_i^2 \qquad \rho \geq 0$$

# Quadratic cost

For a multivariate system ($x \in \mathbb{R}^n, \quad u \in \mathbb{R}^m$):

$$J = \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i \qquad R \geq 0 \;\; Q \geq 0$$

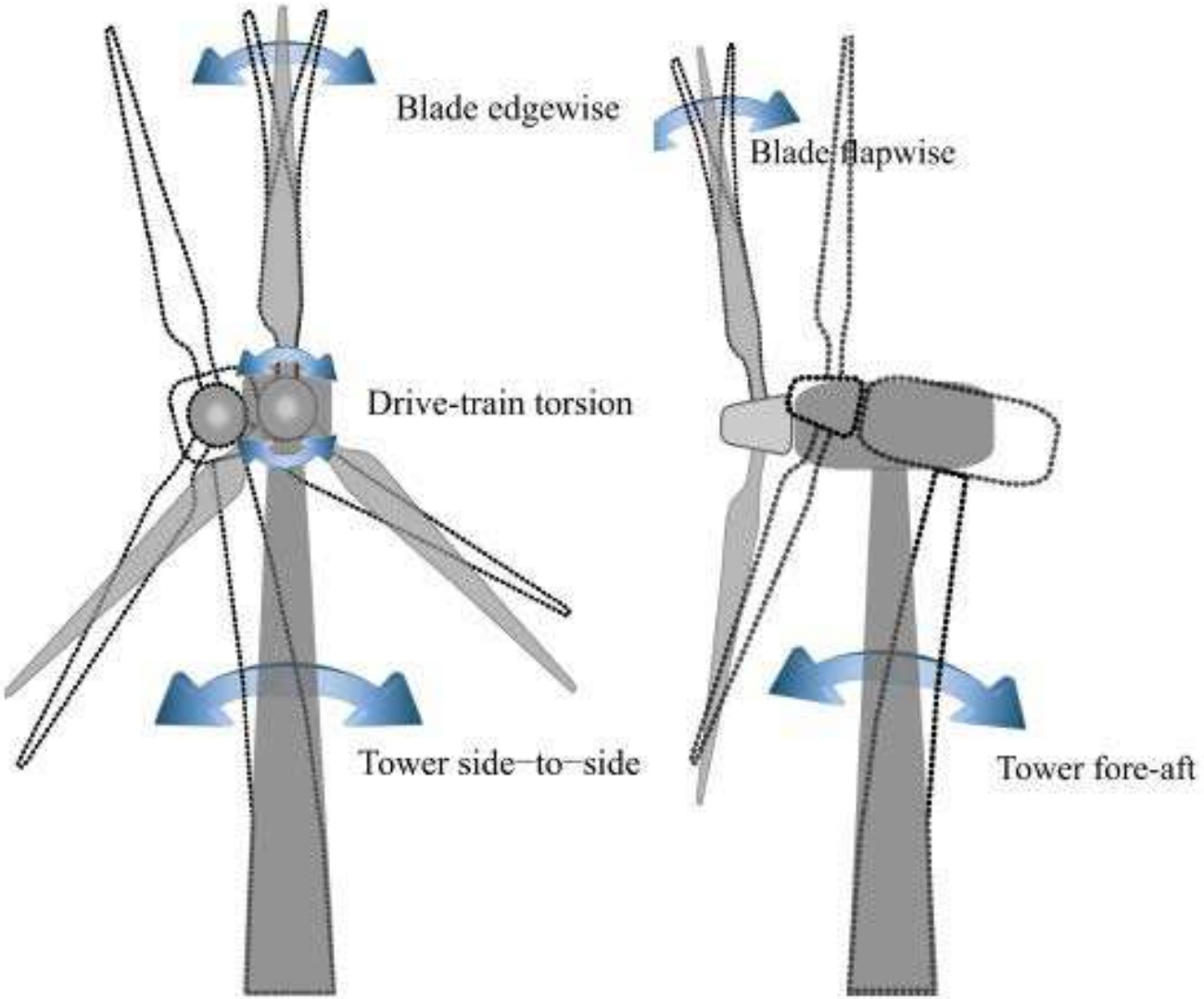
For example:

$$x^T Q x = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 1 & 0.1 \\ 0.1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x_1^2 + 2\, x_2^2 + 0.2\, x_1 x_2$$

---

The quadratic cost is often used in connection to minimizing the (square of) the distance (often to the origin - as in the example above). **Tuning.**

**Fatigue alleviation**




Figures courtesy Mahmood Mirzaei, DTU Wind energy

**Example: Max harvesting**

Consider a system described by the Schaefer model:

$$x_{i+1} = x_i + rx_i\Big(1 - \alpha x_i\Big) - u_i \qquad x_0 = \underline{x}_0 \qquad i = 0,\ 1,\ \ldots\ N$$

Here we might be in a situation in which we want to maximize the total harvest (over a longer peirod of time).

First idea: For fixed $N$ drive the system from $\underline{x}_0$ along a trajectory such that

$$J = u_N + \sum_{i=0}^{N-1} u_i$$

is maximized.

Pitch fall in optimization: Be careful in what you ask for - you might actually get it.

**Example: Minimum consumption**

Consider a situation in which $u_i$ is related to consumption e.g. of some material or cost (price).

For fixed $N$ drive the system from $\underline{x}_0$ to $\underline{x}_N$ such that

$$J = \sum_{i=0}^{N-1} p_i u_i$$

is minimized. $p_i$ is a price. E.g. consumption of electric energy.

# Example: Minimum time

Drive the system from $\underline{x}_0$ to $\underline{x}_N$ such that

$$J = \sum_{i=0}^{N-1} 1 = N$$

is minimized. The length, $N$, of the period is then a part of the optimization.

---

Often some constraints on the states or the decisions are involved.

# Free Dynamic Optimization (D)

Minimize $J$ (ie. determine the sequence $u_i$, $i = 0, \ldots, N-1$) where:

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

subject to

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

Here $N$, $x_0$ (and $f$, $L$ and $\phi$) are given.

This is the **Bolza** formulation. The alternative is a **Mayer** formulation.

# The Mayer form

The problem can also be formulated as a minimization of:

$$J = \tilde{\phi}[\tilde{x}_N] \qquad \tilde{x} = \begin{bmatrix} x \\ z \end{bmatrix}$$

subject to

$$\tilde{x}_{i+1} = \tilde{f}_i(\tilde{x}_i, u_i) \qquad i = 0,\ 1,\ \ldots\ N-1 \qquad\qquad \tilde{x}_0 = \underline{\tilde{x}}_0$$

---

This is an equivalent formation of the Dynamic Optimization problem.

---

The link from the Bolza to the Mayer formulation is given by:
Minimize

$$J = \tilde{\phi}[\tilde{x}_N] = \phi(x_N) + z_N$$

subject to:

$$\begin{bmatrix} x \\ z \end{bmatrix}_{i+1} = \begin{bmatrix} f_i(x_i, u_i) \\ z_i + L_i(x_i, u_i) \end{bmatrix} \qquad i = 0,\ 1,\ \ldots\ N-1 \qquad\qquad \begin{bmatrix} x \\ z \end{bmatrix}_0 = \begin{bmatrix} \underline{x}_0 \\ 0 \end{bmatrix}$$

## Opgave 7.1

Try to minimize

$$J = \|u - \begin{bmatrix} 1 \\ 3 \end{bmatrix}\|^2 + 1$$

wrt. the two dimensional vector $u$ using eg. fminsearch. Write a short m-file (e.g. with the name: fopt.m) with the following contents

```
function loss=fopt(u)
loss=sum((u-[1;3]).^2)+1;
```

and execute the matlab commands (or put them into a script)

```
opt=optimset;
opt=optimset(opt,'Display','iter');
fminsearch('fopt',[1; 1],opt)
```

The 2 first lines are just included for controlling the output (to display) from fminsearch.

## Opgave 7.2

Try to use the matlab command fsolve to find the solution to the problem ($u \in \mathbb{R}^2$):

$$u - \begin{bmatrix} 1 \\ 3 \end{bmatrix} = 0$$

Write a short m-file (e.g. with the name: fz.m) with the following contents

```
function f=fz(u)
f=u-[1;3];
```

and execute the matlab commands

```
opt=optimset;
opt=optimset(opt,'Display','off');
fsolve('fz',[1; 1],opt)
```

# Pause

# Simple optimization

## Unconstrained optimization

Let

$$u = \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix} \qquad u^* = \arg min\ J(u)$$

$$\frac{\partial}{\partial u} J(u) = 0 \qquad \frac{\partial^2}{\partial u^2}\ J(u) > 0$$

## Constrained optimization II

$$\frac{\partial}{\partial \lambda} J_L(u, \lambda) = 0 \qquad g(u) = 0$$

On $g(u) = 0$ we have:

$$\frac{\partial}{\partial u} J_L(u, \lambda) \qquad = 0 = \qquad \frac{\partial}{\partial u} J(u)$$

## Constrained optimization I

Consider the problem of minimizing (find $u \in \mathbb{R}^m$)

$$L(u) \qquad \in \mathbb{R}$$

with respect to the $(p)$ constraints

$$g(u) = 0 \qquad \in \mathbb{R}^p$$

Introduce the Lagrange multipliers ($\lambda \in \mathbb{R}^p$) and the Lagrange relaxation:

$$J_L(u, \lambda) = J(u) + \lambda^T g(u)$$

$$\frac{\partial}{\partial \lambda} J_L(u, \lambda) = 0 \qquad \frac{\partial}{\partial u} J_L(u, \lambda) = 0$$

## Constrained optimization III

Consider the problem of minimizing (find $u \in \mathbb{R}^m$)

$$L(u) \qquad \in \mathbb{R}$$

with respect to the $(p)$ constraints

$$g(u) = h(u) - c = 0 \qquad \in \mathbb{R}^p$$

---

Introduce the Lagrange multipliers ($\lambda \in \mathbb{R}^p$) and the Lagrange relaxation:

$$J_L(u, \lambda) = J(u) + \lambda^T (h(u) - c)$$

---

$$\frac{\partial}{\partial \lambda} J_L(u, \lambda) = 0 \qquad h(u) = c$$

$$\frac{\partial}{\partial u} J_L(u, \lambda) = \frac{\partial}{\partial u} J(u, \lambda) = 0$$

$$\frac{\partial}{\partial c} J_L(u, \lambda) = \frac{\partial}{\partial c} J(u) = -\lambda^T$$

## State dependency

Consider the problem of minimizing (find $u \in \mathbb{R}^n$)

$$L(x, u) \qquad \in \mathbb{R}$$

with respect to the $(n)$ constraints

$$g(x, u) = 0 \qquad \in \mathbb{R}^n$$

---

Introduce the Lagrange multipliers ($\lambda \in \mathbb{R}^n$) and the Lagrange relaxation:

$$J_L(x, u, \lambda) = J(x, u) + \lambda^T g(x, u)$$

---

$$\frac{\partial}{\partial \lambda} J_L(x, u, \lambda) = 0$$

$$\frac{\partial}{\partial x} J_L(x, u, \lambda) = 0$$

$$\frac{\partial}{\partial u} J_L(x, \lambda) = 0$$

# Euler-Lagranges equations

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

subject to

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

---

Define a Lagrange multiplier vector, $\lambda_{i+1}$, (Costate or Adjoin state) for each equality constraints

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

and form the Lagrangian relaxation:

$$J_L = \phi[x_N] + \sum_{i=0}^{N-1} L_i(x_i, u_i) + \sum_{j=0}^{N-1} \lambda_{j+1}^T \left[f_j(x_j, u_j) - x_{j+1}\right] + \lambda_0^T \left[\underline{x}_0 - x_0\right]$$

Necessarily condition: stationarity wrt. to $x_i$, $\lambda_i$ and $u_i$.

# Stationarity wrt. $\lambda$

$$J_L = \phi[x_N] + \sum_{i=0}^{N-1} L_i(x_i, u_i) + \sum_{j=0}^{N-1} \lambda_{j+1}^T \left[ f_j(x_j, u_j) - x_{j+1} \right] + \lambda_0^T \left[ \underline{x}_0 - x_0 \right]$$

Stationarity wrt. $\lambda_{j+1}$ (i.e. wrt. $\lambda_1$ ... $\lambda_N$) gives

$$f_j(x_j, u_j) - x_{j+1} = 0 \qquad\qquad j = 0,\ 1,\ \ldots\ N-1$$

---

or simply the state equation:

$$x_{i+1} = f_i(x_i, u_i) \qquad\qquad j = 0,\ 1,\ \ldots\ N-1$$

Stationarity wrt. $\lambda_0$ gives:

$$x_0 = \underline{x}_0$$

# Stationarity wrt. $x_i$

$$J_L = \phi[x_N] + \sum_{i=0}^{N-1} L_i(x_i, u_i) + \sum_{j=0}^{N-1} \lambda_{j+1}^T \left[ f_j(x_j, u_j) - x_{j+1} \right] + \lambda_0^T \left[ \underline{x}_0 - x_0 \right]$$

Stationarity wrt. $x_i$ $(i = 1, \ \ldots \ N-1)$ gives

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

Notice results for $j = i$ and $j + 1 = i$ . Same result for $i = 0$. Result for $i = N$ stated below.

---

or the costate equation:

$$\lambda_i^T = \frac{\partial}{\partial x_i} L_i(x_i, u_i) + \lambda_{i+1}^T \frac{\partial}{\partial x_i} f_i(x_i, u_i) \qquad\qquad i = 0, \ 1, \ \ldots \ N-1$$

Stationarity wrt. $x_N$ gives

$$\lambda_N^T = \frac{\partial}{\partial x_N} \phi$$

# Stationarity wrt. $u_i$

Cut and paste - and use some colors:

$$J_L = \phi[x_N] + \sum_{i=0}^{N-1} L_i(x_i, u_i) + \sum_{j=0}^{N-1} \lambda_{j+1}^T \left[ f_j(x_j, u_j) - x_{j+1} \right] + \lambda_0^T \left[ \underline{x}_0 - x_0 \right]$$

---

Stationarity wrt. $u_i$ gives the optimality condition or the stationarity condition:

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

# Euler-Lagrange equations

Actually, the discrete Euler-Lagrange (EL) equations. Consider the (the Bolza formulation of the) problem of minimizing $J$, where

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

subject to

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

The EL equations (KKT conditions) are (for $i = 0,\ 1,\ \ldots\ N-1$):

$$\begin{aligned} x_{i+1} &= f_i(x_i, u_i) \\ \lambda_i^T &= \frac{\partial}{\partial x_i} L_i(x_i, u_i) + \lambda_{i+1}^T \frac{\partial}{\partial x_i} f_i(x_i, u_i) \\ 0 &= \frac{\partial}{\partial u_i} L_i(x_i, u_i) + \lambda_{i+1}^T \frac{\partial}{\partial u_i} f_i(x_i, u_i) \end{aligned}$$

with boundary conditions

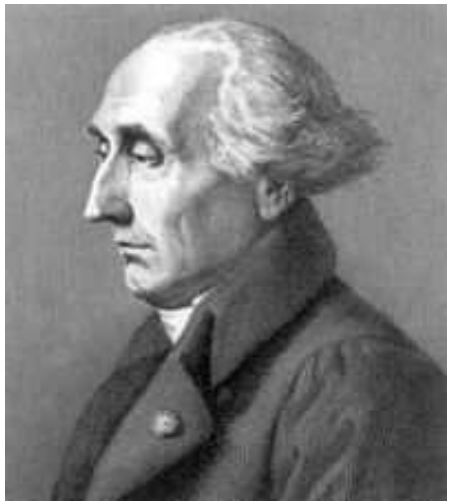
$$x_0 = \underline{x}_0 \qquad \lambda_N^T = \frac{\partial}{\partial x_N} \phi$$

This is a two-point boundary value problem (TPBVP) with $N(2n+m)$ unknowns and equations.

# Euler

Born 15 April 1707 Basel, Switzerland. Died: 18 September 1783 (aged 76) Saint Petersburg, Russian Empire. Residence: Kingdom of Prussia, Russian Empire, Switzerland. Nationality: Swiss. Alma mater: University of Basel. Doctoral advisor: Johann Bernoulli. Doctoral students: Nicolas Fuss, Johann Hennert, Joseph Louis Lagrange.

# Lagrange



Born: Giuseppe Luigi Lagrancia, 25 January 1736 Turin, Piedmont-Sardinia. Died: 10 April 1813 (aged 77) Paris, France. Residence: Piedmont, France, Prussia. Citizenship: Kingdom of Sardinia, France. Nationality: Italian, French. Institutions: Ecole Polytechnique. Doctoral advisor: Leonhard Euler. Doctoral students: Joseph Fourier, Giovanni Plana, Simeon Poisson.

# Example: Optimal stepping (D-time version of motion control)

Consider the toy (or black board) problem of stepping the system

$$x_{i+1} = x_i + u_i \qquad i = 0,\ 1,\ \ldots\ N-1$$

from an original position $\underline{x}_0$ towards origin in $N$ steps. This is related to minimizing

$$J_1 = \frac{1}{2}x_N^2$$

If the original objective and the step power has same weight (for the sake of simplicity) then the steps should be found as the minimizing strategy of

$$J = \frac{1}{2}x_N^2 + \sum_{i=0}^{N-1} \frac{1}{2}u_i^2$$

In this case the EL equations are:

$$x_{i+1} = x_i + u_i \qquad x_0 = \underline{x}_0$$
$$\lambda_i = \lambda_{i+1} \qquad \lambda_N = x_N$$
$$0 = u_i + \lambda_{i+1}$$

# Example: Optimal stepping

The last two are easily solved

$$\lambda_i = x_N \qquad u_i = -x_N$$

which in the state equation give

$$x_{i+1} = x_i - x_N$$

or

$$x_i = x_0 - ix_N$$

For $i = N$ this results in

$$x_N = \frac{1}{N+1} x_0$$

and

$$u_i = -\frac{1}{N+1} x_0 \qquad x_i = x_0 - i\frac{1}{N+1} x_0 \qquad \lambda_i = \frac{1}{N+1} x_0$$

This problem has a degree of simplicity that allow us to find an analytical solution.

# Types of solutions

Examples:

- Analytical solutions (for very simple problems)
- Semi analytical solutions (for semi simple problems eg. the LQ problem)
- Numerical solutions

# The Hamiltonian function

Introduce the Hamiltonian function:

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

Then the EL can be written in a condensed form:

$$x_{i+1}^T = \frac{\partial}{\partial \lambda_{i+1}} H_i \qquad \lambda_i^T = \frac{\partial}{\partial x_i} H_i \qquad 0 = \frac{\partial}{\partial u_i} H_i$$

$$x_0 = \underline{x}_0 \qquad \lambda_N^T = \frac{\partial}{\partial x} \phi$$

- $\frac{\partial}{\partial u} H_i$ is the gradient of $J$ wrt. $u_i$ and is (also) called the pulse response sequence.
- $\lambda_0^T$ is the gradient of $J$ wrt. $x_0$.

Notice the EL equations are necessary conditions.

# Lesson learned (7)

Minimize $J$ (ie. determine the sequence $u_i$, $i = 0, \ldots, N-1$) where:

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

subject to

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

Defining the Hamiltonian function

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

The Euler-Lagrange equations can be written as:

$$x_{i+1} = f_i \qquad \lambda_i^T = \frac{\partial}{\partial x_i} H_i \qquad 0 = \frac{\partial}{\partial u_i} H_i$$

$$x_0 = \underline{x}_0 \qquad \lambda_N^T = \frac{\partial}{\partial x} \phi$$

# Reading guidance

Slides, Exercises, solutions and m-files on:

http://www.imm.dtu.dk/courses/42111/

Reading:
DO: p. 5-11, 15-27 (Ch. 2 except continuous time problems).
Appendix B.

# Notation

Let $s$ be a scalar, $x$ a (column) vector

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

and $f$ a vector function (of dim m). Then

$$\frac{\partial}{\partial x} s = \left(\frac{\partial}{\partial x_1} s \;\; \frac{\partial}{\partial x_2} s \;\; \ldots \;\; \frac{\partial}{\partial x_n} s\right)$$

ie. a row vector.

Furthermore, the Jacobian is defined as:

$$\frac{\partial}{\partial x} f = \begin{bmatrix} \frac{\partial}{\partial x_1} f_1 & \frac{\partial}{\partial x_2} f_1 & \cdots & \frac{\partial}{\partial x_n} f_1 \\ \frac{\partial}{\partial x_1} f_2 & \frac{\partial}{\partial x_2} f_2 & \cdots & \frac{\partial}{\partial x_n} f_2 \\ \vdots & \vdots & & \vdots \\ \frac{\partial}{\partial x_1} f_m & \frac{\partial}{\partial x_2} f_m & \cdots & \frac{\partial}{\partial x_n} f_m \end{bmatrix}$$

# Notation

The most common examples are:

$$\frac{\partial}{\partial x} Ax = A$$

$$\frac{\partial}{\partial x} y^T x = y^T$$

$$\frac{\partial}{\partial x} x^T y = y^T$$

$$\frac{\partial}{\partial x} x^T Q x = 2x^T Q$$

Rene Victor Valqui Vidal

# Glimpse of History

## International

- Norbert Wiener

1969 Bryson and Ho

- Lewis
- Bertsekas
- R.E. Kalman
- Atkins

## National

1967 VVV

- Hans Ravn
- Svend Clausen
- Jens Ryberg

1995 Jens Clausen - nkp

- Richard Lusby
- Rune Larsen
- Evelien van der Hurk

# Notation

## A comment on notation

We use

$$x_i \qquad \text{and} \qquad f_i(x_i, u_i)$$

rather than

$$x(i) \qquad \text{and} \qquad f(i, x(i), u(i))$$

---

## The curse of $x$

- vector containing the state variable
- first coordinate (in a position)
- decision variable in static optimization

Hack(s):

- use $s$ as symbol for the state
- use $z$ as symbol for first coordinate
- use $u$ as decision variable.

# Other practicalities

- Campus net (cn.dtu.dk, now inside.dtu.dk).
- Course home page (www.imm.dtu.dk/courses/42111).
- Gbar download (Matlab) (http://downloads.cc.dtu.dk/).
- Report errors and fishy elements in slides and lecture notes.
- Email suggestions for improvements.
- Include course id (42111) and your study id in emails (subject field).

# Task 7.3

Consider the problem of payment of a (study) loan which at the start of the period is 50.000 Dkr. Let us focus on the problem for a period of 10 years. We are going to determine the optimal pay back strategy for this loan, i.e. to determine how much we have to pay each year. Assume that the rate of interests is 5 % per year ($\alpha = 0.05$) (and that the loan is credited each year), then the dynamics of the problem can be described by:

$$x_{i+1} = ax_i + bu_i \qquad a = 1 + \alpha \qquad b = -1$$

where $x_i$ is the actual size of the loan (including interests) and $u_i$ is the annual payment.
On one hand, we are interested in minimizing the amount we have to pay to the bank. On the other hand, we are not interested in paying to much each year. The objective function, which we might use in the minimization could be

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

where $q = \alpha^2$. The weights $r$ and $p$ are at our disposal. Let us for a start choose $r = q$ and $p = q$ (but let the parameters be variable in your program in order to change them easily).

## General EL

Minimize:

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

subject to

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

---

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

---

$$x_{i+1} = f_i \qquad \lambda_i^T = \frac{\partial}{\partial x_i} H_i \qquad 0 = \frac{\partial}{\partial u_i} H_i$$

$$x_0 = \underline{x}_0 \qquad \lambda_N^T = \frac{\partial}{\partial x} \phi$$

## Specific EL

$$J = \frac{1}{2} p x_N^2 + \sum_{i=0}^{N-1} \frac{1}{2} q x_i^2 + \frac{1}{2} r u_i^2$$

$$x_{i+1} = a x_i + b u_i$$

---

$$H_i = \frac{1}{2} q x_i^2 + \frac{1}{2} r u_i^2 + \lambda_{i+1}(a x_i + b u_i)$$

---

$$x_{i+1} = a x_i + b u_i \qquad x_0 = 50000$$
$$\lambda_i = q x_i + a \lambda_{i+1} \qquad \lambda_N = p x_N$$
$$0 = r u_i + b \lambda_{i+1}$$

---

Guess $\lambda_0$ and iterate:

$$\lambda_{i+1} = \frac{\lambda_i - q x_i}{a}$$

$$u_i = \frac{b \lambda_{i+1}}{r}$$

$$x_{i+1} = a x_i + b u_i$$

until $\lambda_N = p x_N$.