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

2019-10-26 20:28

Lecture 8: Free dynamic optimization (D+C)

Outline of lecture

- D time DO
  - Why (Theory)
- C time DO
  - Why (Theory)
  - How (algorithms)

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:

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

Constrained optimization I

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

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

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:

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

KKT:

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

Constrained optimization III

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

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

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 $u \in \mathbb{R}^n$)

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

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:

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

KKT

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

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

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

Euler-Lagrange equations

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

subject to

$$x_{i+1} = f_i(x_i, u_i) \quad \in \mathbb{R}^n \qquad i = 0,\ 1,\ \dots\ N-1 \qquad\qquad x_0 = \underline{x}_0$$

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

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

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, λ_i and u_i.

Stationarity wrt. λ

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

Stationarity wrt. λ_{j+1} (i.e. wrt. λ_1 ... λ_N) gives

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

---

or simply the state equation:

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

Stationarity wrt. λ_0 gives:

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

Stationarity wrt. x_i

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

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

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

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:

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

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

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

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 $i = 0,\ 1,\ \dots\ N-1$):

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

with boundary conditions

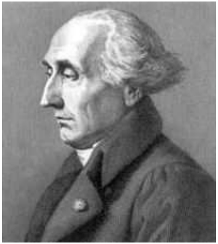
$$x_0 = \underline{x}_0 \qquad \lambda_N^T = \frac{\partial}{\partial x_N} \phi$$

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.

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:

$$\boxed{\begin{array}{c} x_{i+1}^T = \frac{\partial}{\partial \lambda_{i+1}} H_i \qquad \lambda_i^T = \frac{\partial}{\partial x_i} H_i \qquad 0 = \frac{\partial}{\partial u_i} H_i \\ x_0 = \underline{x}_0 \qquad \lambda_N^T = \frac{\partial}{\partial x} \phi \end{array}}$$

Notice the EL equations are necessary conditions.

Euler Lagrange II

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

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

subject to

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

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:

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

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

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

$$\frac{\partial}{\partial x} s = (\frac{\partial}{\partial x_1} s \; \frac{\partial}{\partial x_2} s \; ... \; \frac{\partial}{\partial x_n} s)$$

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

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

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

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

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

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

Guess λ_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 $\lambda_N = p x_N$.

Type of solutions:

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

Contents of a file (parms.m) setting the parameters.

```
% Constants etc.

alf=0.05;
a=1+alf; b=-1;
x0=50000;
N=10;
q=alf^2; r=q; p=q;
```

The following code (fejlf.m) solves these recursions.

```
function err=fejlf(la0)

parms % set parameters a,b,p,q,r,x0

la=la0; x=x0;
for i=0:N-1,
 la=(la-q*x)/a;
 u=-b*la/r;
 x=a*x+b*u;
end
err=la-p*x;
```

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

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

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

Extented version of fejlf.m (for plotting).

```
function [err,xt,ut,lat]=fejlf(la0)

parms % set parameters a,b,p,q,r,x0

la=la0; x=x0;
ut=[]; lat=la; xt=x;
for i=0:N-1,
 la=(la-q*x)/a;
 u=-b*la/r;
 x=a*x+b*u;
 xt=[xt;x]; lat=[lat;la]; ut=[ut;u];
end
err=la-p*x;
```

Master program (script).

```
% The search for la0
la0g=10; % a wild guess%
la0=fsolve('fejlf',la0g)

% The simulation with the correct la0
[err,xt,ut,lat]=fejlf(la0);

subplot(211); bar(ut); grid; title('Input sequence');
subplot(212); bar(xt); grid; title('Saldo');
```

Forward shooting method - Simplified

If separation possible: reverse the costate equation and find u_i from the stationarity condition.

The Euler-Lagrange equations

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

Guess λ_0 (<u>or another parameterization</u>) and use $\underline{x}_0$.

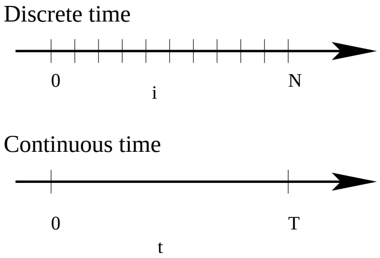
Iterate for $i = 0\ ,1\ \ldots\ N-1$:

1. Knowing x_i and λ_i, determine u_i and λ_{i+1} from the stationarity and the costate equation.
2. Update the state equation i.e. find x_{i+1} from x_i and u_i.

At the end $(i = N)$ check if

$$
\lambda_N^T = \frac{\partial}{\partial x_N} \phi(x_N) \qquad \rightarrow \qquad \varepsilon = \lambda_N^T - \frac{\partial}{\partial x_N} \phi(x_N) = 0^T
$$

Continuous time Optimization



The **Schaefer model** (Biomass i a biotop, Fish in the Baltics)

$$x_{i+1} = x_i + rhx_i(1 - \alpha x_i) \qquad x_0 = \underline{u}_0$$

h is the length of the intervals. The model can in continuous time be given as:

$$\boxed{\dot{x}_t = \frac{dx_t}{dt} = rx_t(1 - \alpha x_t) \qquad x_0 = \underline{x}_0}$$

The **fox(F) and rabbit(r)** example.

$$\begin{bmatrix} \dot{r} \\ \dot{F} \end{bmatrix} = \begin{bmatrix} \alpha_1 r - \beta_1 r F \\ -\alpha_2 F + \beta_2 r F \end{bmatrix} \qquad \begin{bmatrix} r \\ F \end{bmatrix}_0 = \begin{bmatrix} \underline{r}_0 \\ \underline{F}_0 \end{bmatrix}$$

In general Dynamic (continuous time) state space model:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \end{bmatrix} = f_t\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_t, \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix}_t \right) \qquad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_0 = \begin{bmatrix} \underline{x}_1 \\ \underline{x}_2 \\ \vdots \\ \underline{x}_n \end{bmatrix}_0$$

or in short:

$$\dot{x}_t = f_t(x_t, u_t) \qquad x_0 = \underline{x}_0 \qquad f : \mathbb{R}^{n+m+1} \to \mathbb{R}^n$$

The function, f, should be sufficiently smooth (existence and uniqueness).

Solution to ODE

- Analytical methods
- Numerical methods

$$\dot{x} = f_t(x_t) \qquad x_0 = \underline{x}_0$$

Euler integration (the most simple method)

$$x_{t+h} = x_t + h f_t(x_t)$$

is the most simple method. Others and more efficient numerical methods do exists.

Analytical solutions

Discrete time

$$x_{i+1} = x_i \qquad x_i = C$$

$$x_{i+1} = x_i + \alpha \qquad x_i = C + \alpha i$$

$$x_{i+1} = a x_i \qquad x_i = C a^i$$

$$x_{i+1} = A x_i + B u_i \qquad x_0 = \underline{x}_0$$

$$x_i = A^i \underline{x}_0 + \sum_{j=0}^{i} A^{n-j-1} B u_j$$

Continuous time

$$\dot{x}_t = 0 \qquad x_t = C$$

$$\dot{x}_t = \alpha \qquad x_t = C + \alpha t$$

$$\dot{x}_t = a x_t \qquad x_t = C exp(at)$$

$$\dot{x}_t = A x_t + B u_t \qquad x_0 = \underline{x}_0$$

$$x_t = e^{At} \underline{x}_0 + \int_0^t e^{A(t-s)} B u_s ds$$

Constant as C can be determined from boundary conditions. Examples are

$$x_0 = \underline{x}_0 \qquad \text{or} \qquad x_N = \underline{x}_N$$

Opgave 8.1

Consider the Van der Pool oscillator represented by the ODE:

$$\ddot{z} - a(1 - z^2)\dot{z} + z = 0$$

where $a = 1$ is a parameter. This is an example on a second order, non-linear differential equation. If we choose the two (since it is a second order system) as:

$$x_1 = z \qquad x_2 = \dot{z}$$

we have

$$\dot{x}_1 = \dot{z} = x_2 \qquad \dot{x}_2 = \ddot{z} = a(1 - z^2)\dot{z} - z = a(1 - x_1^2)x_2 - x_1$$

This is a state space description of the Van der Pool oscillator and can be written in a compact form as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ a(1 - x_1^2)x_2 - x_1 \end{bmatrix}$$

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

```
function dx=vdp(t,x,a)
x1=x(1); x2=x(2);
dx=[x2; a*(1-x1^2)*x2-x1];
```

Assume the initial state is characterized by the fact $z = 1$ and $\dot{z} = 0$ (i.e. $\underline{x}_0 = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$) and we want se the solution in the interval $[0;\ 30]$ with a resolution equals 0.1.
Execute the followin matlab commands:

```
Tspan=0:0.1:30;
x0=[1;0];
a=1;

[t,x]=ode45(@vdp,Tspan,x0,[],a);

z=x(:,1);
plot(t,z); grid; title('Van der Pool oscillator');
xlabel('time'); ylabel('position');
```

Change the value of a in the interval $[1;\ 10]$ and study (i.e. plot) the effect on the solution.

Dynamic Optimization and Euler-Lagrange Equations

Free dynamic optimization: Minimize J (ie. determine the function u_t, $0 \leq t \leq T$) where:

$$J = \phi_T(x_T) + \int_0^T L_t(x_t, u_t)\, dt \qquad \text{Objective}$$

subject to

$$\dot{x} = f_t(x_t, u_t) \qquad x_0 = \underline{x}_0 \qquad \text{Dynamics}$$

$$\begin{aligned}
\dot{x}_t &= f_t(x_t, u_t) \\
-\dot{\lambda}_t^T &= \frac{\partial}{\partial x_t} L_t(x_t, u_t) + \lambda_t^T \frac{\partial}{\partial x_t} f_t(x_t, u_t) \\
0^T &= \frac{\partial}{\partial u_t} L_t(x_t, u_t) + \lambda_t^T \frac{\partial}{\partial u_t} f_t(x_t, u_t)
\end{aligned}$$

with boundary conditions:

$$x_0 = \underline{x}_0 \qquad \lambda_T^T = \frac{\partial}{\partial x} \phi_T(x_T)$$

Hamilton function

Define the Hamilton function as:

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

Then the Euler-Lagrange equations (KKT conditions) for this problem can be written as:

$$\boxed{\dot{x}_t^T = \frac{\partial}{\partial \lambda_t} H_t \qquad -\dot{\lambda}_t^T = \frac{\partial}{\partial x_t} H_t \qquad 0^T = \frac{\partial}{\partial u_t} H_t}$$

The first equation is just the state equation

$$\dot{x} = f_t(x_t, u_t)$$

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

$$\begin{aligned}
\dot{H} &= \frac{\partial}{\partial t}H + \frac{\partial}{\partial u}H \;\; \dot{u} + \frac{\partial}{\partial x}H \;\; \dot{x} + \frac{\partial}{\partial \lambda}H \;\; \dot{\lambda} \\
&= \frac{\partial}{\partial t}H + \frac{\partial}{\partial u}H \;\; \dot{u} + \frac{\partial}{\partial x}H \;\; f + f^T \dot{\lambda} \\
&= \frac{\partial}{\partial t}H + \frac{\partial}{\partial u}H \;\; \dot{u} + \left[\frac{\partial}{\partial x}H + \dot{\lambda}^T\right] f \\
&= \frac{\partial}{\partial t}H \qquad = 0 \qquad \text{for time invariant problems}
\end{aligned}$$

along the optimal trajectories for x, u and λ.

Reading guidance

DO: Section 2, Appendix B

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).