Chapter 1: Feedback Linearisation Control

Chapter 1.1 Lie Derivative

```
% The plant is considered bellow
close all;
figure(1)
imshow(imread("plant_siso_system.png"));
```

```
syms x1 x2 u % symbolic presentation
fx=[x2;-sin(x1)];
g=[0;1];
x=[x1 x2];
h=x1;
LH=LieDerivative(h,x)
```

LH = $\begin{pmatrix} 1 & 0 \end{pmatrix}$

```
[lhf lhg]=solvelieder(LH,fx,g)
```

lhf = x_2

lhg = 0

Chapter 1.2 Feedback Linearisation Controller Examples

```
clear all;clc
disp('------------------------------------');
```

```
------------------------------------
```

```
disp('The Nonlinear systems should ');
```

```
The Nonlinear systems should
```

```
disp('be written in the following form ');
```

```
be written in the following form
```

```
disp('State space equations x=f(x)+g(x)u');
```

```
State space equations x=f(x)+g(x)u
```

```
disp('------------------------------------');
```

```
------------------------------------
```

```
% The your system contains
% Input the extra parameters
```

```
par=input('Parameters ','s');
eval(sprintf('syms %s',par));
parameters=sprintf('%s',par)
```

```
parameters =

  1×0 empty char array
```

```
%% Declare how many states and inputs
n=input('Number of states:=');
nin=input('Number of inputs:=');
x=sym(zeros(1,n));
u=sym(zeros(1,nin));
for j=1:n
    eval(sprintf('syms x%d',j))
    x(:,j)=sprintf('x%d',j);
end
for k=1:nin
    eval(sprintf('syms u%d',k));
    u(:,k)=sprintf('u%d',k);
end
% Enter the functions from the keyboard
f=input('The vector f(x):=','s');
g=input('The vector g(x):=','s');
Hc=input('The output variables:=','s');
%Represent all the functions
%f(x), g(x) and h(x) on a symbolic format
fx=str2sym(f);
g=str2sym(g);
Hc=str2sym(Hc); %
% Use the inoutfeedbacklinearization.m
% programm to generate the desired functions
[Lhf Lhg]=inoutfeedbacklinearization(fx,g,Hc,x)
```

```
The relative degree of h1
equal:=1
The relative degree of h2
equal:=1
Lhf =
```

$$\begin{pmatrix} x_1 + x_1 x_2 \\ -\sin(x_1) \end{pmatrix}$$

```
Lhg =
```

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Chapter 1.3 Illustrative examples

1.3.1 Aircraft altitude dynamics

```
% The plant is considered bellow
close all;
```

```
figure(1)
imshow(imread("aircraft_motion_equations.png"));
```

```
clear all;clc
disp('------------------------------------');
```

```
------------------------------------
```

```
disp('The Nonlinear systems should ');
```

```
The Nonlinear systems should
```

```
disp('be written in the following form ');
```

```
be written in the following form
```

```
disp('State space equations x=f(x)+g(x)u');
```

```
State space equations x=f(x)+g(x)u
```

```
disp('------------------------------------');
```

```
------------------------------------
```

```
% The your system contains
% Input the extra parameters
par=input('Parameters ','s');
eval(sprintf('syms %s',par));
parameters=sprintf('%s',par)
```

```
parameters =

  1×0 empty char array
```

```
%% Declare how many states and inputs
n=input('Number of states:=');
nin=input('Number of inputs:=');
x=sym(zeros(1,n));
u=sym(zeros(1,nin));
for j=1:n
    eval(sprintf('syms x%d',j))
    x(:,j)=sprintf('x%d',j);
end
for k=1:nin
    eval(sprintf('syms u%d',k));
    u(:,k)=sprintf('u%d',k);
end
% Enter the functions from the keyboard
f=input('The vector f(x):=','s');
```

```
g=input('The vector g(x):=','s');
Hc=input('The output variables:=','s');
%Represent all the functions
%f(x), g(x) and h(x) on a symbolic format
fx=str2sym(f);
g=str2sym(g);
Hc=str2sym(Hc); %
% Use the inoutfeedbacklinearization.m
% programm to generate the desired functions
[Lhf Lhg]=inoutfeedbacklinearization(fx,g,Hc,x)
```

```
The relative degree of h1
equal:=2
```

Lhf = $6\,x_1$

Lhg = -1

1.3.2 Asynchronous motor speed control

```
% The plant is considered bellow
close all;
figure(1)
imshow(imread("asynchronous_motor_dynamical_model.png"));
```

```
clear all;clc
disp('------------------------------------');
```

```
------------------------------------
```

```
disp('The Nonlinear systems should ');
```

```
The Nonlinear systems should
```

```
disp('be written in the following form ');
```

```
be written in the following form
```

```
disp('State space equations x=f(x)+g(x)u');
```

```
State space equations x=f(x)+g(x)u
```

```
disp('------------------------------------');
```

```
------------------------------------
```

```
% The your system contains
% Input the extra parameters
par=input('Parameters ','s');
eval(sprintf('syms %s',par));
parameters=sprintf('%s',par)
```

```
parameters =
'T1 gamma K Tr p fm Jm Lr M Ls sigma'
```

```
%% Declare how many states and inputs
n=input('Number of states:=');
nin=input('Number of inputs:=');
x=sym(zeros(1,n));
u=sym(zeros(1,nin));
for j=1:n
    eval(sprintf('syms x%d',j))
    x(:,j)=sprintf('x%d',j);
end
for k=1:nin
    eval(sprintf('syms u%d',k));
    u(:,k)=sprintf('u%d',k);
end
% Enter the functions from the keyboard
f=input('The vector f(x):=','s');
g=input('The vector g(x):=','s');
Hc=input('The output variables:=','s');
%Represent all the functions
%f(x), g(x) and h(x) on a symbolic format
fx=str2sym(f);
g=str2sym(g);
Hc=str2sym(Hc); %
% Use the inoutfeedbacklinearization.m
% programm to generate the desired functions
[Lhf Lhg]=inoutfeedbacklinearization(fx,g,Hc,x)
```

```
The relative degree of h1
equal:=2
The relative degree of h2
equal:=2
Lhf =
```

$$\left(\begin{array}{c}\left(\frac{4\,x_3}{\mathrm{Tr}}-\frac{2\,M\,x_1}{\mathrm{Tr}}\right)\sigma_4-\left(\frac{4\,x_4}{\mathrm{Tr}}-\frac{2\,M\,x_2}{\mathrm{Tr}}\right)\sigma_3+\frac{2\,M\,x_3\,\sigma_2}{\mathrm{Tr}}-\frac{2\,M\,x_4\,\sigma_1}{\mathrm{Tr}}\\ \frac{\mathrm{fm}\,\left(\frac{T_1}{\mathrm{Jm}}+\frac{\mathrm{fm}\,x_5}{\mathrm{Jm}}+\frac{M\,p\,\left(x_1\,x_4-x_2\,x_3\right)}{\mathrm{Jm}\,\mathrm{Lr}}\right)}{\mathrm{Jm}}-\frac{M\,p\,x_3\,\sigma_1}{\mathrm{Jm}\,\mathrm{Lr}}-\frac{M\,p\,x_4\,\sigma_2}{\mathrm{Jm}\,\mathrm{Lr}}-\frac{M\,p\,x_1\,\sigma_3}{\mathrm{Jm}\,\mathrm{Lr}}-\frac{M\,p\,x_2\,\sigma_4}{\mathrm{Jm}\,\mathrm{Lr}}\end{array}\right)$$

where

$$\sigma_1=\gamma\,x_2-\frac{K\,x_4}{\mathrm{Tr}}+K\,p\,x_3\,x_5$$

$$\sigma_2=\frac{K\,x_3}{\mathrm{Tr}}-\gamma\,x_1+K\,p\,x_4\,x_5$$

$$\sigma_3=p\,x_3\,x_5-\frac{x_4}{\mathrm{Tr}}+\frac{M\,x_2}{\mathrm{Tr}}$$

$$\sigma_4=\frac{x_3}{\mathrm{Tr}}+p\,x_4\,x_5-\frac{M\,x_1}{\mathrm{Tr}}$$

```
Lhg =
```

$$\left(\begin{array}{cc}\frac{2\,M\,x_3}{\mathrm{Ls}\,\mathrm{Tr}\,\sigma} & \frac{2\,M\,x_4}{\mathrm{Ls}\,\mathrm{Tr}\,\sigma}\\ -\frac{M\,p\,x_4}{\mathrm{Jm}\,\mathrm{Lr}\,\mathrm{Ls}\,\sigma} & \frac{M\,p\,x_3}{\mathrm{Jm}\,\mathrm{Lr}\,\mathrm{Ls}\,\sigma}\end{array}\right)$$

Chapter 2 Sliding Mode Control

2.1 Sliding mode control examples for SISO Systems

2.1.1 Van der Pol system

```
% The plant is considered bellow
close all;
figure(1)
imshow(imread("van_der_pol_dynamical_equations.png"));
```

```
clear all;clc
disp('--------------------------------------------------');
```

```
--------------------------------------------------
```

```
disp('The Nonlinear systems should be written in the following form');
```

The Nonlinear systems should be written in the following form

```
disp('--Sliding Mode Controller for a class of Nonlinear systems--');
```

```
--Sliding Mode Controller for a class of Nonlinear systems--
```

```
disp(' State space equations x=f(x)+gu ');
```

```
 State space equations x=f(x)+gu
```

```
disp('---------------------------------------------------');
```

```
---------------------------------------------------
```

```
n=input('Number of states:=');
nin=input('Number of inputs:=');
x=sym(zeros(1,n));
u=sym(zeros(1,nin));
par=input('Parameters ','s');
eval(sprintf('syms %s',par));
parameters=sprintf('%s',par)
```

```
parameters =
'e'
```

```
for j=1:n
    eval(sprintf('syms x%d',j))
    x(:,j)=sprintf('x%d',j);
end
for k=1:nin
    eval(sprintf('syms u%d',k));
    u(:,k)=sprintf('u%d',k);
end
syms u1
f=input('The vector f(x):=','s');
g=input('The vector g(x):=','s');
Hc=input('The output variables:=','s');
f=str2sym(f);Hc=str2sym(Hc);g=str2sym(g);
[Lhf,Lhg,dh,L,u,r]=NonContSidFed(f,g,Hc,x,u1);
[Surf,dSurf,dd,K,Uc]=SlidingModeTerms(Hc,L,r,Lhg);
```

The sliding mode control law for SISO systems=:

Uc = $\mathrm{d3yr} + x_1 + \mathrm{kp}\,\mathrm{sgnS} + k_1\,(\mathrm{d2yr} - x_2) + e\,x_2\,\left(x_1^2 - 1\right)$

The Sliding mode surface:=

Surf = $\mathrm{d2yr} - x_2 + k_1\,(\mathrm{d1yr} - x_1)$

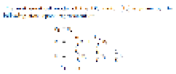
The derivative of Sliding mode surface:=

dSurf = $\mathrm{d3yr} + x_1 + k_1\,(\mathrm{d2yr} - x_2) + e\,x_2\,\left(x_1^2 - 1\right)$

2.1.1 DC motor angular position control

```
% The plant is considered bellow
close all;
```

```
figure(1)
imshow(imread("DC_motor_mathematical_model.png"));
```



```
clear all;clc
disp('---------------------------------------------------');
```

```
---------------------------------------------------
```

```
disp('The Nonlinear systems should be written in the following form');
```

```
The Nonlinear systems should be written in the following form
```

```
disp('--Sliding Mode Controller for a class of Nonlinear systems--');
```

```
--Sliding Mode Controller for a class of Nonlinear systems--
```

```
disp(' State space equations x=f(x)+gu ');
```

```
 State space equations x=f(x)+gu
```

```
disp('---------------------------------------------------');
```

```
---------------------------------------------------
```

```
n=input('Number of states:=');
nin=input('Number of inputs:=');
x=sym(zeros(1,n));
u=sym(zeros(1,nin));
par=input('Parameters ','s');
eval(sprintf('syms %s',par));
parameters=sprintf('%s',par)
```

```
parameters =
'b J km ke L R'
```

```
for j=1:n
    eval(sprintf('syms x%d',j))
    x(:,j)=sprintf('x%d',j);
end
for k=1:nin
    eval(sprintf('syms u%d',k));
    u(:,k)=sprintf('u%d',k);
end
syms u1
f=input('The vector f(x):=','s');
g=input('The vector g(x):=','s');
Hc=input('The output variables:=','s');
f=str2sym(f);Hc=str2sym(Hc);g=str2sym(g);
[Lhf,Lhg,dh,L,u,r]=NonContSidFed(f,g,Hc,x,u1);
```

```
[Surf,dSurf,dd,K,Uc]=SlidingModeTerms(Hc,L,r,Lhg);
```

```
The sliding mode control law for SISO systems=:
Uc =
```

$$\mathrm{d4yr} + \mathrm{kp}\,\mathrm{sgnS} + k_1\left(\mathrm{d3yr} + \frac{b\,x_2}{J} - \frac{\mathrm{km}\,x_3}{J}\right) + k_2\,(\mathrm{d2yr} - x_2) + \frac{\mathrm{km}\left(\frac{R\,x_3}{L} + \frac{\mathrm{ke}\,x_2}{L}\right)}{J} - \frac{b\left(\frac{b\,x_2}{J} - \frac{\mathrm{km}\,x_3}{J}\right)}{J}$$

```
The Sliding mode surface:=
Surf =
```

$$\mathrm{d3yr} + k_2\,(\mathrm{d1yr} - x_1) + k_1\,(\mathrm{d2yr} - x_2) + \frac{b\,x_2}{J} - \frac{\mathrm{km}\,x_3}{J}$$

```
The derivative of Sliding mode surface:=
dSurf =
```

$$\mathrm{d4yr} + k_1\left(\mathrm{d3yr} + \frac{b\,x_2}{J} - \frac{\mathrm{km}\,x_3}{J}\right) + k_2\,(\mathrm{d2yr} - x_2) + \frac{\mathrm{km}\left(\frac{R\,x_3}{L} + \frac{\mathrm{ke}\,x_2}{L}\right)}{J} - \frac{b\left(\frac{b\,x_2}{J} - \frac{\mathrm{km}\,x_3}{J}\right)}{J}$$

2.2 Sliding mode control examples for MIMO Systems

2.2.1 Permanent magnet synchronous motor speed control

```
% The plant is considered bellow
close all;
figure(1)
imshow(imread("PMSM_dynamical_model.png"));
```

The dynamical model of PMSM [15] describing the electrical and mechanical part is given by:

$$\dot{x}_1 = -\frac{R}{L_d}x_1 + p\frac{L_q}{L_d}x_2x_3 + \frac{1}{L_d}u_d$$
$$\dot{x}_2 = -\frac{R}{L_q}x_2 - p\frac{L_q}{L_d}x_1x_3 - p\frac{\Phi}{L_q} + \frac{1}{L_q}u_q$$
$$\dot{x}_3 = p\frac{\Phi_f}{J}x_2 - p\frac{L_q - L_d}{J}x_1x_2 - \frac{f}{J}x_3 - \frac{1}{J}\tau$$

Where x_1 and x_2 are the d and q axis stator currents respectively; x_3 is the mechanical speed of the motor; u_d and u_q are the d axis and q axis stator voltages respectively; R and $L_d = L_d$ are the winding resistance and inductance on the d and q axis. J is mechanical inertia of the motor; τ is the electrical torque.

The objective is to control the mechanical velocity x_3 and the x_1 current.

$$h(x) = \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_3 \end{bmatrix}$$

```
clear all;clc
disp('---------------------------------------------------');
```

```
---------------------------------------------------
```

```
disp('The Nonlinear systems should be written in the following form');
```

```
The Nonlinear systems should be written in the following form
```

```
disp('--Sliding Mode Controller for MIMO Nonlinear systems--');
```

```
--Sliding Mode Controller for MIMO Nonlinear systems--
```

```
disp(' State space equations x=f(x)+gu');
```

```
 State space equations x=f(x)+gu
```

```
disp('---------------------------------------------------');
```

```
---------------------------------------------------
```

```
n=input('Number of states:=');
nin=input('Number of inputs:=');
x=sym(zeros(1,n));
u=sym(zeros(1,nin));
par=input('Parameters ','s');
eval(sprintf('syms %s',par));
parameters=sprintf('%s',par);
for j=1:n
eval(sprintf('syms x%d',j))
x(:,j)=sprintf('x%d',j);
end
for k=1:nin
eval(sprintf('syms u%d',k));
u(:,k)=sprintf('u%d',k);
end
f=input('The vector f(x):=','s');
g=input('The vector g(x):=','s');
h=input('The output variables:=','s');
f=str2sym(f);h=str2sym(h);g=str2sym(g);
[Lhf Lhg L r]=MIMOSlidingModeLieDer(f,g,h,x);
```

```
The relative degree of h1
equal:=1
The relative degree of h2
equal:=2
```

```
[e,der,Surf,dSurf,Uc]=MIMOSlidingModeController(h,L,r,Lhg);
```

```
The sliding mode control law for MIMO systems
------Is given by Uc=:inv(Lhg)*(S)-------------
------The function S:=----------
S =
```

$$\left(\begin{array}{c} \mathrm{d2yr}_1 + \mathrm{kp}_1\,\mathrm{sgnS}_1 + \frac{R\,x_1}{\mathrm{Ld}} - \sigma_2 \\ \mathrm{d2yr}_2 + \mathrm{kp}_2\,\mathrm{sgnS}_2 + \frac{T_1}{J} + k_1 \left(\mathrm{d3yr}_2 + \left(\frac{p\,\phi}{J} + \frac{p\,x_1\,(\mathrm{Ld}-\mathrm{Lq})}{J} \right) \left(\frac{R\,x_2}{\mathrm{Lq}} + \frac{p\,\phi}{\mathrm{Ld}} + \frac{\mathrm{Lq}\,p\,x_1\,x_3}{\mathrm{Ld}} \right) - \frac{f\,\left(\frac{T_1}{J} + \cdots \right.}{\cdots} \right. \end{array} \right.$$

where

$$\sigma_1 = \frac{p\,x_1\,x_2\,(\mathrm{Ld}-\mathrm{Lq})}{J}$$

$$\sigma_2 = \frac{\mathrm{Ld}\,p\,x_2\,x_3}{\mathrm{Lq}}$$

$$\sigma_3 = \frac{p\,\phi\,x_2}{J}$$

```
------The Matrix Lhg:=----------
Lhg =
```

$$\begin{pmatrix} \frac{1}{\mathrm{Ld}} & 0 \\ \frac{p\,x_2\,(\mathrm{Ld}-\mathrm{Lq})}{J\,\mathrm{Ld}} & \frac{\frac{p\,\phi}{J} + \frac{p\,x_1\,(\mathrm{Ld}-\mathrm{Lq})}{J}}{\mathrm{Lq}} \end{pmatrix}$$

```
The sliding mode control law for MIMO systems=:
Uc =
```

$$\left(\begin{array}{c} \mathrm{Ld}\,\sigma_3 \\ \frac{J\,\mathrm{Lq} \left(\mathrm{d2yr}_2 + \mathrm{kp}_2\,\mathrm{sgnS}_2 + \frac{T_1}{J} + k_1 \left(\mathrm{d3yr}_2 + \left(\frac{p\,\phi}{J} + \frac{p\,x_1\,(\mathrm{Ld}-\mathrm{Lq})}{J} \right) \left(\frac{R\,x_2}{\mathrm{Lq}} + \frac{p\,\phi}{\mathrm{Ld}} + \frac{\mathrm{Lq}\,p\,x_1\,x_3}{\mathrm{Ld}} \right) - \frac{f\,(\cdots}{\cdots} \right. \right.}{p\,\phi + \mathrm{Ld}\,p\,x_1 - \mathrm{Lq}\,p\,x_1} \end{array} \right.$$

where

$$\sigma_1 = \frac{p\,x_1\,x_2\,(\mathrm{Ld}-\mathrm{Lq})}{J}$$

$$\sigma_2 = \frac{p\,\phi\,x_2}{J}$$

$$\sigma_3 = \mathrm{d2yr}_1 + \mathrm{kp}_1\,\mathrm{sgnS}_1 + \frac{R\,x_1}{\mathrm{Ld}} - \sigma_4$$

$$\sigma_4 = \frac{\mathrm{Ld}\,p\,x_2\,x_3}{\mathrm{Lq}}$$