Solution1:

$$\begin{cases}\dot{x}_1 & x_2{}^2 + u \\ \dot{x}_2 = & x_3{}^2 + u \\ \dot{x}_3 & p(x_1) + u\end{cases} \quad f(x) = \begin{bmatrix} x_2{}^2 \\ x_3{}^2 \\ p(x_1) \end{bmatrix} \quad g(x) = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

The condition of the existence of a linearizer of all states

$$ad_{f,g} = [f, g] = -\frac{\partial f}{\partial x} g = \begin{bmatrix} 2x_2 \\ 2x_3 \\ \frac{\partial p}{\partial x_1} \end{bmatrix}$$

$$ad^2{}_{f,g} = [f, ad_{f,g}] = -\frac{\partial f}{\partial x} ad_{f,g} = \begin{bmatrix} 4x_3x_2 \\ 2x_3 \frac{\partial p}{\partial x_1} \\ 2x_2 \frac{\partial p}{\partial x_1} \end{bmatrix} \quad G = [g \;\; ad_{f,g} \; ad^2{}_{f,g}]$$

$$\det(G) \neq 0 \quad \text{else } (x_1, x_2, x_3) = (0,0,0)$$

$$H = \text{span}\{g, ad_{f,g}\} \text{ is involutive}$$

$$\text{then } [g, ad_{f,g}] = \frac{\partial ad_{f,g}}{\partial x} g = \begin{bmatrix} 2 \\ 2 \\ \frac{\partial^2 p}{\partial x_1{}^2} \end{bmatrix} \in \text{span}(g, ad_{f,g})$$

$$\text{for } p(x_1) = x_1{}^2 + cx_1 + b \quad [g, ad_{f,g}] = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} = 2 * g$$

$$\text{with } p = \text{polynomial form we can do fullstate Linearization}$$

The condition that all states become linear for a 3rd order system like the above is that

$$Z = T(x) \text{ and } (I) \begin{cases} \frac{\partial T_1}{\partial x} g = 0 \\ \frac{\partial T_2}{\partial x} g = 0 \\ \frac{\partial T_3}{\partial x} g \neq 0 \end{cases} \text{ and } (II) \begin{cases} \frac{\partial T_1}{\partial x} f = T_2 \\ \frac{\partial T_2}{\partial x} f = T_3 \end{cases}$$

The condition that all states: exist $T_1 = h$ only, and according to condition (II) we have:

$$\frac{\partial h}{\partial x} f = T_2 = L_f h \quad \text{and} \quad \frac{\partial L_f h}{\partial x} f = T_3 = {L_f}^2 h$$

$$L_f h = \begin{bmatrix} \frac{\partial h}{\partial x_1} & \frac{\partial h}{\partial x_2} & \frac{\partial h}{\partial x_3} \end{bmatrix} * \begin{bmatrix} {x_2}^2 \\ {x_3}^2 \\ p(x_1) \end{bmatrix} = \frac{\partial h}{\partial x_1} {x_2}^2 + \frac{\partial h}{\partial x_2} {x_3}^2 + \frac{\partial h}{\partial x_3} p(x_1)$$

$$\text{then}: \ (I) ==> \frac{\partial T_1}{\partial x} g = 0 \text{ then } L_g h = 0$$

$$\text{then}: \ \frac{\partial h}{\partial x} g = \begin{bmatrix} \frac{\partial h}{\partial x_1} & \frac{\partial h}{\partial x_2} & \frac{\partial h}{\partial x_3} \end{bmatrix} * \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \frac{\partial h}{\partial x_1} + \frac{\partial h}{\partial x_2} + \frac{\partial h}{\partial x_3} = 0$$

$$\text{then}: \ \frac{\partial T_2}{\partial x} g = \text{then } L_g L_f h = 0$$

$$\frac{\partial L_f h}{\partial x} g = \begin{bmatrix} \frac{\partial L_f h}{\partial x_1} & \frac{\partial L_f h}{\partial x_2} & \frac{\partial L_f h}{\partial x_3} \end{bmatrix} * \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \frac{\partial L_f h}{\partial x_1} + \frac{\partial L_f h}{\partial x_2} + \frac{\partial L_f h}{\partial x_3}$$

$$\frac{\partial L_f h}{\partial x} g = \frac{\partial^2 h}{\partial^2 x_1} {x_2}^2 + \frac{\partial^2 h}{\partial x_1 \partial x_2} {x_3}^2 + \frac{\partial^2 h}{\partial x_1 \partial x_3} p(x_1) + \frac{\partial h}{\partial x_3} \frac{\partial p(x_1)}{\partial x_1}$$

$$+ \frac{\partial^2 h}{\partial x_1 \partial x_2} {x_2}^2 + \frac{\partial^2 h}{\partial^2 x_2} {x_3}^2 + \frac{\partial^2 h}{\partial x_2 \partial x_3} p(x_1) + 2 \frac{\partial h}{\partial x_1} x_2 + \frac{\partial^2 h}{\partial x_1 \partial x_3} {x_2}^2$$

$$+ \frac{\partial^2 h}{\partial x_2 \partial x_3} {x_3}^2 + \frac{\partial^2 h}{\partial^2 x_3} p(x_1) + 2 \frac{\partial h}{\partial x_2} x_3$$

Due to the continuous assumption of the function h, the differential can be moved.

$$\frac{\partial L_f h}{\partial x} g = \frac{\partial}{\partial x_1} \left(\frac{\partial h}{\partial x_1} + \frac{\partial h}{\partial x_2} + \frac{\partial h}{\partial x_3} \right) {x_2}^2 + \frac{\partial}{\partial x_2} \left(\frac{\partial h}{\partial x_1} + \frac{\partial h}{\partial x_2} + \frac{\partial h}{\partial x_3} \right) {x_3}^2$$

$$+ \frac{\partial}{\partial x_3} \left(\frac{\partial h}{\partial x_1} + \frac{\partial h}{\partial x_2} + \frac{\partial h}{\partial x_3} \right) p(x_1) + \frac{\partial h}{\partial x_3} \frac{\partial p(x_1)}{\partial x_1} + 2 \frac{\partial h}{\partial x_1} + x_2 2 \frac{\partial h}{\partial x_2} x_3$$

$$\frac{\partial L_f h}{\partial x} g = 0$$

$$\frac{\partial h}{\partial x_1} + \frac{\partial h}{\partial x_2} + \frac{\partial h}{\partial x_3} = 0 \ \text{ then}: \ \frac{\partial h}{\partial x_3} \frac{\partial p(x_1)}{\partial x_1} + 2 \frac{\partial h}{\partial x_1} x_2 + 2 \frac{\partial h}{\partial x_2} x_3 = 0$$

$$\frac{\partial h}{\partial x_1}+\frac{\partial h}{\partial x_2}+\frac{\partial h}{\partial x_3}=0 \text{ then } : \frac{\partial h}{\partial x_3}=-\left(\frac{\partial h}{\partial x_1}+\frac{\partial h}{\partial x_2}\right)$$

$$\text{then : (eq1) : } \frac{\partial h}{\partial x_1}\left(2x_2-\frac{\partial p(x_1)}{\partial x_1}\right)+\frac{\partial h}{\partial x_2}\left(2x_3-\frac{\partial p(x_1)}{\partial x_1}\right)=0$$

solve this equation with "MAPEL15" and suppose that $p(x_1)$ is a polynomial function:

$$p(x_1)={x_1}^2+cx_1+b$$

$$h(x1,x2,x3)=\_F1\Bigg(x3, -\frac{1}{6}\bigg(2\arctan\left(\frac{1}{3}\frac{(4x2-2x3-2x1-c)\sqrt{3}}{2x3-2x1-c}\right) + \sqrt{3}\ln\left(4x2^2-4x3x2-4x2x1-2x2c+4x3^2-4x3x1-2x3c+4x1^2+4x1c+c^2\right)\bigg)\sqrt{3}\Bigg)$$

$$\text{Then : } \quad T1=h= -2\arctan\left(\frac{1}{3}\frac{(4x2-2x3-2x1-c)\sqrt{3}}{2x3-2x1-c}\right)-\sqrt{3}\ln\left(4x2^2-4x3x2-4x2x1-2x2c+4x3^2-4x3x1-2x3c+4x1^2+4x1c+c^2\right)$$

$$T2=L_fh \text{ and } T3={L_f}^2h$$

With T1, T2, T3 and $Z=T(x)$ We can have full state linearization:

Then $z_1=h\,,\ z_2=L_fh\,,\ z_3={L_f}^2h$

with this $z=T(x)$ $L_gh=0$ and $L_gL_fh=0$

and $L_g{L_f}^2h\neq 0$ then $r-1=2$ then : $r=3$

$$\begin{cases}\dot z_1 = z_2\\ \dot z_2 = z_3\\ \dot z_3 = L^3{}_fh+L_gL^2{}_fhu\end{cases} \ \& \ x=T^{-1}(z) \quad \text{and} \quad u=-\frac{L^3{}_fh}{L_gL^2{}_f}+\frac{1}{L_gL^2{}_f}v$$

$$\begin{cases}\dot z_1 = z_2\\ \dot z_2 = z_3\\ \dot z_3 = v\end{cases} \quad v=-z_1-3z_2-3z_3 \quad \begin{cases}\dot z_1 = z_2\\ \dot z_2 = z_3\\ \dot z_3 = -z_1-3z_2-3z_3\end{cases} \quad \text{And system is e.s}$$

Solution2:

1. We notice that all state equation but the last one are linear. The last state equation reads:

$$\dot{x}_n = f(x) + g(x)u$$

If we assume that $g(x) \neq 0$ for all x, we can apply the control:

$$u = h(x,v) = \frac{1}{g(x)}(-f(x) + Lx + v)$$

renders the last state equation linear

$$\dot{x}_n = Lx + u$$

The response from v to x is linear, and the closed loop dynamics is given by

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots \\ 0 & 0 & 0 & \ddots & 0 \\ l_1 & l_2 & l_3 & \dots & l_n \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} v$$

(You may recognize this as the controller form from the basic control course). For the control to be well defined, we must require that $g(x) \neq 0$ for all x.

2. The above procedure suggest the control

$$u = h(x,v) = \frac{1}{bcos(x_1)}(-\operatorname{asin}(x_1) + l_1x_1 + l_2x_2 + v)$$

which results in the closed loop system

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ l_1 & l_2 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v$$

The system matrix has a double eigenvalue in $s = -1$ if we let:

$$l_1 = -1, l_2 = -2$$

The control law is well defined for $x_1 \neq \pi/2$. This corresponds to the pendulum being horizontal. For $x_1 = \pi/2$, u has no influence on the system. Notice how the control "blows up" nearby this singularity. Extra. You may want to verify by simulations the behavior of the modified control:

$$u = sat(h(x,v))$$

for different values of the saturation level.

3. The above procedure suggest the control:

$$u = -x^2 - x + v$$

Letting $v = 0$, we apply the control to the perturbed system:

$$\dot{x} = (1+\epsilon)x^2 - x^2 - x = \epsilon x^2 - x$$

and note that for $x > 1/\epsilon$, we have $\dot{x} > 0$, which implies that the trajectories tend to infinity. Thus, global cancellation is non-robust in the sense that it may require a very precise mathematical model.

Solution3:

1. The aim of feedback linearization is to produce a transformation system whose states are the output y and its derivatives. Therefore, the lie derivative is then used, and since it's an important tool for analysis and synthesis, a MATLAB program has been developed for any class of nonlinear system that could be written in the form of an equation (1). The two programs work as follows:

1. The *Lie Derivative*(h; x) program finds the partial derivative of $h(x)$ along x; as described by the following equation: $\dot{f} = \frac{\partial h(x)}{\partial x}\dot{x}$; thus, the user should provide the program with the vector field
2. The *solvelieder*(Lh; fx; g) gives the lied derivatives of the vector fields $f(x)$ and $h(x)$ along the vector fields $f(x)$ and $g(x)$ respectively. Where the user has to provide the program with the function $f(x)$ and $g(x)$ as well as the output derivative from the function *LieDerivative*(h; x)

```
function df=Lie_Derivative(h,x)
% The LieDerivaive MATLAB function is used
% to find the jacobian vector of a given output
% h(x) : Is the output function
% x : The state vector
% df : The jacobian of h along x

if nargin<2 & nargin==0
error('not enough input argument');
end
df=[];
n=length(x);
```

```
for ii=1:n
xx=x(ii);
dff=diff(h,xx);
df=[df,dff];
end
df;
end


function [lhf lhg]=solvelieder(Lh,fx,g)
% The solvelieder MATLAB function is used to find
% the lie derivatives of the functions f(x) and g(x)
% along the vector field h(x)
% Lh : The jacobian vector of h along x
% fx : The function f(x)
% g : The input function g(x)
LHg=[];
lhf=Lh*fx;
[n,b]=size(g);
for ii=1:b
Lgh=Lh*g(:,ii);
LHg=[LHg,Lgh];
end
lhg=LHg;
end
```

Hand Calculation:

The lie derivative of any nonlinear dynamical systems that has the form of Equation 1 is obtained by differentiating the output of the system; hence, the first lie derivative is given by:

$$\dot{y} = \frac{\partial h(x)}{\partial x}\,\dot{x} = [1\ \ 0](f(x) + gu)$$

Therefore, the first lie derivative is given by:

$$\dot{y} = \frac{\partial h(x)}{\partial x}\,\dot{x} = [1\ \ 0]\begin{pmatrix} x_2 \\ -\sin(x_1) + u \end{pmatrix}$$

$$\dot{y} = L_f^1\, h(x) = x_2$$

MATLAB *Computation:*

The first lie derivative of equation is then obtained using MATLAB program described on the previous section as follows:

```
clear;
clc;
```

```
syms x1 x2 u % symbolic presentation
fx=[x2;-sin(x1)];
g=[0;1];
x=[x1 x2];H=x1;
LH=Lie_Derivative(H,x)
[lhf lhg]=solvelieder(LH,fx,g)
```

```
Answer:
LH =[ 1, 0]
lhf =x2
lhg =0
```

The first lie derivative of the dynamical system in Equation 1.13 is displayed using MATLAB as $L_f h(x) = \text{lhf}$ and $L_g h = \text{lhg}$.

2. Feedback linearization MATLAB-based function; and for better readability of the problems are then solved using programmes illustrated on the previous chapters, we will write them on another manner that simplify the use for users; hence, the program that should be used by the user will be written as:

```
clear all;
clc
disp('_____________________________________________');
disp('The Nonlinear systems should ');
disp('be written in the following form ');
disp('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)
%% 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=sym(f);
g=sym(g);
Hc=sym(Hc); %
%% Use the inoutfeedbacklinearization.m programm to generate the desired
functions
disp(['_________ANSWER_________']);
[Lhf Lhg]=inoutfeedbacklinearization(fx,g,Hc,x)
disp(['The feedbacklinearization controller Uc:']);
disp(inv(Lhg)*(-Lhf+u))
disp(['_________END_PROGRAM_________']);
```

Thus the inoutfeedbacklinearization.m function is then written using a nested MATLAB functions instead of using the programs separately as shown on the previous chapters; therefore the whole code is given by:

```
function [Lhf Lhg]=inoutfeedbacklinearization(fx,g,h,x)
% The function inoutfeedbacklinearization is used to find the
% the feedbacklinearization control law for SISO and MIMO
% nonlinear systems using symbolic MATLAB library;
% The user should provide the program with the following
% inputs
% fx : The system function f(x)
% g : The system output function g(x)
% h(x) : The vector of outputs h(x)=[x1;x2, ;xn]
% x : The state vector x=[x1,x2, ,xn]
% After having provided the program the necessary input
% functions
% The program will output the following variables
% Lhf : The lie derivative of h(x) along the function f(x)
% Lhg : The lie derivative of h(x) along the function g(x)
% which is called the decoupling matrix
% u : The vector of inputs u=[u1;u2;...;un]
% The control law will be given by the following formula
% u= inv(Lhg)*(?Lhg+v)
if nargin <4
error('Not enough input argument');
end
k=1;
```

```
Lhg=[];Lhf=[];
nb=length(h);
while k<length(h)+1
h1=h(k);
for i=1:nb+1
% this Lie derivative function
df=Lie_Derivative(h1,x);
% solve for the g
[lhf lhg]=solvelieder(df,fx,g);
[n b]=size(lhg);
for ii=1:n
d=any(lhg(ii,:)~=0);
end
if d==1;
disp(['The relative degree of h',num2str(k)]),
disp(['equal:=',num2str(i)]);
break;
else
h1=lhf;
end
if i==nb+1 && d==0
disp(['The system dose not admit NFL']);
return;
end
end
Lhg=[Lhg;lhg];
Lhf=[Lhf;lhf];
k=k+1;
end

end
```

Hand Calculation:

Following the steps mentioned in the flow chart of Figure 1; then the control law will be obtained by differentiating the output h till u appears as:

$$L_f^1 h_1(x) = x_1 + x_1 x_2$$

$$L_f^1 h_2(x) = -\sin(x_2)$$

$$L_g^1 h_1(x) = 1$$

$$L_g^1 h_2(x) = 1$$

the relative degree of the system which is 1 and the control law u is obtained using as:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -L_f^1 h_1(x) + v_1 \\ -L_f^1 h_2(x) + v_1 \end{bmatrix}$$

MATLAB Calculation:

```
____________________________________________
The Nonlinear systems should
be written in the following form
State space equations x=f(x)+g(x)u
____________________________________________
Parameters
parameters =
  1×0 empty char array

Number of states:=2
Number of inputs:=2
The vector f(x):=[x1+x1*x2;-sin(x1)]
The vector g(x):=[1,0;0,1]
The output variables:=[x1;x2]
_________ANSWER_________
The relative degree of h1
equal:=1
The relative degree of h2
equal:=1
Lhf =
 x1 + x1*x2
    -sin(x1)
```

```
Lhg =
[ 1, 0]
[ 0, 1]
```

3. MATLAB inputs:

```
parameters =  1×0 empty char array
Number of states:=4
Number of inputs:=1
The vector f(x):=[x2;-4*x2-4*x1;x4;6*x1]
The vector g(x):=[0;3;0;-1]
The output variables:=x3
_________ANSWER_________
The relative degree of h1
equal:=2
Lhf =
6*x1
Lhg =
-1
The feedbacklinearization controller Uc:
6*x1 - u1
_________END_PROGRAM_________
```