Backstepping control

BASIC CONCEPTS

A picture containing diagram

Description automatically generated

The problem: design , s.t.

Step 1: [choose PHI fcn]

Let , [st ] [it has only one zero and it is at the origin]

What is the condition on for the dynamics to be stable [A.S.] ?

Let’s use a Lyapunov fcn for that,

[lyap fcn must be pos-def-fcn], [time-der of lyap fcn must be neg-def-fcn]

So must satisfy these two constraints:

Step 2: [change of variables]

Define

and

What is z-dyns?

In the new coordinate system, the dyns,

Define a new input signal , therefore the new dyns,

A picture containing text

Description automatically generated

Now the new problem is as the one given in the block-diagram. the second subsystem is A.S., when zero “input” [z-signal] is applied.

All of the transformations and new variable definitions were introduced to make the second subsystem A.S. And This has consequences, now we can find u using lyap-fcn in this coordinate frame. [the reason why we were not able to find u in the original coordinates by using lyap fcn approach is that if we did we would have terms in our ctrl law and as xi goes to zero, these terms becomes problematic]

Continuing, the new problem, design st

Choose a lyap-fcn as

must be pos-def-fcn, and must be pos-def-fcn. must be neg-def-fcn.

Take the time-der

So, must satisfy these two constraints: [ is a pos-def-fcn]

Let us recall the fcn that we have constructed to determine fcn,

Therefore, fcn can be selected as

Let us go back to lyap-fcn-constraints to design ,

So the cons-2 can be changed by another cons,

Solving for yields,

Therefore,

Which indicates the stability of the system.

CTRL ALGO: [backstepping]

This is the general problem definition, but easier version can be obtained by assuming,

Again, by assuming,

PROBLEM 1

Hassan Khalil – nonlinear systems 3rd Ed Example 14.9

|  |  |  |
| --- | --- | --- |
|  | Subs |  |
|  | Subs |  |

By choosing,

[NOTE: more sophisticated functions can also be chosen]

If there is any term in first parenthesis, carry them to their corresponding parenthesis by arranging the terms.

Arrange LHS, s.t. there is only and terms in TERM-1.

Arrange LHS, s.t. there is not any terms in TERM-2.

Find s.t.

Now let us focus on the TERM-2

Find s.t.

Solve for

Using

Using

Now let us focus on the TERM-3

Find s.t.

Using

Substitute explicit forms of by using

[NOTE: This computation is brought to you by MATLAB sym toolbox!]

THE ANSWER:

Text, letter

Description automatically generated

Check the result by simulating the closed loop system in Simulink[MATLAB].

|  |  |
| --- | --- |
|  |  |

Result: [initial condition]

Chart, line chart

Description automatically generated

|  |
| --- |
| Matlab code for problem 1 |
| clear all,close all,clc;  %%  syms x1 x2 x3 u real  syms phi1(x1) phi2(x1,x2) real  syms z2 z3 real  x1dot\_1=(x1)^2-(x1)^3+x2  x2dot\_1=x3  x3dot\_1=u  % substitute "phi1+z2" for "x2" in x1dot Eq.  x1dot=subs(x1dot\_1,x2,phi1+z2)  % methods(x1dot)  symvar(x1dot) % after subs'n, x1dot should be composed of "[phi1, x1, z2]"  % delete all z2 terms in x1dot and find phi1 s.t. "x1-dyns" is stable  x1dot\_wo\_z2=subs(x1dot,z2,0)  % at the end the lyap fcn will look like  % V=x1\*[x1,phi1,z2]+z2\*[phi2,z3,phi1dot]+z3\*[u,phi2dot]  % arrange this expr in ...  % V=x1\*[x1,phi1]+z2\*[phi2,phi1dot]+z3\*[u,phi2dot]  % so that  % V=x1\*[TERM\_1]+z2\*[TERM\_2]+z3\*[TERM\_3]  % TERM\_1 should not have z2,z3... terms  % TERM\_2 should not have z3,z4... terms  z2\_terms\_to\_add\_to\_z2\_paranthesis=(x1dot-x1dot\_wo\_z2)\*x1/z2  symvar(x1dot\_wo\_z2)  % x1dot=fcn(x1,phi1(x1)), choose "phi1(x1)" s.t. "x1 is stable"  phi1=rhs(isolate(x1dot\_wo\_z2==-x1-x1^3,phi1))  symvar(phi1) % "phi1" must be composed of [x1]  % find "phi1\_dot"  phi1\_dot=diff(phi1,x1)\*x1dot\_1  symvar(phi1\_dot) % "phi1\_dot" must be composed of [x1,x2]  % x2=z2+phi1 --> x2dot=z2dot+phi1dot --> z2dot=x2dot-phi1dot  % find z2dot expression  z2dot=x2dot\_1-phi1\_dot  % substitute "phi2+z3" for "x3" in z2dot Eq.  z2dot=subs(z2dot,x3,phi2+z3)  % delete all z3 terms in z2dot and find phi2 s.t. "z2-dyns" is stableXXX  z2dot\_wo\_z3=subs(z2dot,z3,0)  z3\_terms\_to\_add\_to\_z3\_paranthesis=(z2dot-z2dot\_wo\_z3)\*z2/z3  %  phi2=rhs(isolate(z2dot\_wo\_z3+z2\_terms\_to\_add\_to\_z2\_paranthesis==-z2,phi2))  symvar(phi2) % "phi2" must be composed of [x1,x2]  phi2=subs(phi2,z2,x2-phi1)  symvar(phi2) % "phi2" must be composed of [x1,x2]  %%% isAlways(subs(phi2,z2,x2-phi1)==-x1-(1+2\*x1)\*(x1^2-x1^3+x2)-(x2+x1+x1^2))  phi2\_dot=diff(phi2,x1)\*x1dot\_1+diff(phi2,x2)\*x2dot\_1  phi2\_dot=simplify(phi2\_dot)  symvar(phi2\_dot) % "phi2\_dot" must be composed of [x1,x2,x3]  % x3=z3+phi2 --> x3dot=z3dot+phi2dot --> z3dot=x3dot-phi2dot  % find z3dot expression  z3dot=x3dot\_1-phi2\_dot  symvar(z3dot) % "z3dot" must be composed of [x1,x2,x3,u]  u=rhs(isolate(z3dot+z3\_terms\_to\_add\_to\_z3\_paranthesis==-z3,u))  symvar(u) % "u" must be composed of [x1,x2,x3]  u=subs(u,z2,x2-phi1)  u=subs(u,z3,x3-phi2)  symvar(u) % "u" must be composed of [x1,x2,x3]  u=simplify(u)  symvar(u)  [C,T]=coeffs(u,[x1,x2,x3]) |
| Matlab code output for problem 1 |
|  |

PROBLEM 2

CONTRIVED EXAMPLE

|  |  |  |
| --- | --- | --- |
|  | Subs |  |
|  | Subs |  |

By choosing,

[NOTE: more sophisticated functions can also be chosen]

If there is any term in first parenthesis, carry them to their corresponding parenthesis by arranging the terms.

Arrange LHS, s.t. there is only and terms in TERM-1.

Arrange LHS, s.t. there is not any terms in TERM-2.

Find s.t.

Now let us focus on the TERM-2

Find s.t.

Solve for

Using

Using

Now let us focus on the TERM-3

Find s.t.

Using

Substitute explicit forms of by using

|  |  |
| --- | --- |
|  |  |
|  |  |

[NOTE: This computation is brought to you by MATLAB sym toolbox!]

THE ANSWER:

Text, letter

Description automatically generated

Check the result by simulating the closed loop system in Simulink[MATLAB].

|  |  |
| --- | --- |
|  |  |

Result: [initial condition]

Chart, line chart

Description automatically generated

|  |
| --- |
| Matlab code for problem 2 |
| clear all,close all,clc;  %%  syms x1 x2 x3 u real  syms phi1(x1) phi2(x1,x2) real  syms z2 z3 real  x1dot\_1=(x1)^2+(x1+1)\*x2  x2dot\_1=(x1)^2+x3  x3dot\_1=u  % substitute "phi1+z2" for "x2" in x1dot Eq.  x1dot=subs(x1dot\_1,x2,phi1+z2)  % methods(x1dot)  symvar(x1dot) % after subs'n, x1dot should be composed of "[phi1, x1, z2]"  % delete all z2 terms in x1dot and find phi1 s.t. "x1-dyns" is stable  x1dot\_wo\_z2=subs(x1dot,z2,0)  % at the end the lyap fcn will look like  % V=x1\*[x1,phi1,z2]+z2\*[phi2,z3,phi1dot]+z3\*[u,phi2dot]  % arrange this expr in ...  % V=x1\*[x1,phi1]+z2\*[phi2,phi1dot]+z3\*[u,phi2dot]  % so that  % V=x1\*[TERM\_1]+z2\*[TERM\_2]+z3\*[TERM\_3]  % TERM\_1 should not have z2,z3... terms  % TERM\_2 should not have z3,z4... terms  z2\_terms\_to\_add\_to\_z2\_paranthesis=(x1dot-x1dot\_wo\_z2)\*x1/z2  z2\_terms\_to\_add\_to\_z2\_paranthesis=expand(z2\_terms\_to\_add\_to\_z2\_paranthesis)  symvar(x1dot\_wo\_z2)  % x1dot=fcn(x1,phi1(x1)), choose "phi1(x1)" s.t. "x1 is stable"  expand(x1dot\_wo\_z2/x1) % analyze TERM\_1  phi1=rhs(isolate(x1dot\_wo\_z2==-x1,phi1))  % phi1=rhs(isolate(x1dot\_wo\_z2==-x1-x1^3,phi1))  symvar(phi1) % "phi1" must be composed of [x1]  % find "phi1\_dot"  phi1\_dot=diff(phi1,x1)\*x1dot\_1  symvar(phi1\_dot) % "phi1\_dot" must be composed of [x1,x2]  % x2=z2+phi1 --> x2dot=z2dot+phi1dot --> z2dot=x2dot-phi1dot  % find z2dot expression  z2dot=x2dot\_1-phi1\_dot  % substitute "phi2+z3" for "x3" in z2dot Eq.  z2dot=subs(z2dot,x3,phi2+z3)  % delete all z3 terms in z2dot and find phi2 s.t. "z2-dyns" is stableXXX  z2dot\_wo\_z3=subs(z2dot,z3,0)  z3\_terms\_to\_add\_to\_z3\_paranthesis=(z2dot-z2dot\_wo\_z3)\*z2/z3  z3\_terms\_to\_add\_to\_z3\_paranthesis=expand(z3\_terms\_to\_add\_to\_z3\_paranthesis)  %  phi2=rhs(isolate(z2dot\_wo\_z3+z2\_terms\_to\_add\_to\_z2\_paranthesis==-z2,phi2))  phi2=expand(phi2)  symvar(phi2) % "phi2" must be composed of [x1,x2]  phi2=subs(phi2,z2,x2-phi1)  symvar(phi2) % "phi2" must be composed of [x1,x2]  %%% isAlways(subs(phi2,z2,x2-phi1)==-x1-(1+2\*x1)\*(x1^2-x1^3+x2)-(x2+x1+x1^2))  phi2\_dot=diff(phi2,x1)\*x1dot\_1+diff(phi2,x2)\*x2dot\_1  pretty(simplify(phi2\_dot))  phi2\_dot=simplify(phi2\_dot)  symvar(phi2\_dot) % "phi2\_dot" must be composed of [x1,x2,x3]  % x3=z3+phi2 --> x3dot=z3dot+phi2dot --> z3dot=x3dot-phi2dot  % find z3dot expression  z3dot=x3dot\_1-phi2\_dot  symvar(z3dot) % "z3dot" must be composed of [x1,x2,x3,u]  u=rhs(isolate(z3dot+z3\_terms\_to\_add\_to\_z3\_paranthesis==-z3,u))  symvar(u) % "u" must be composed of [x1,x2,x3]  u=subs(u,z2,x2-phi1)  u=subs(u,z3,x3-phi2)  symvar(u) % "u" must be composed of [x1,x2,x3]  u=simplify(u)  symvar(u)  subs(u,[x1,x2,x3],[0,0,0])  [C,T]=coeffs(u,[x1,x2,x3]) |
| Matlab code output for problem 2 |
|  |

SUMMARY - PROBLEM 3

|  |  |  |
| --- | --- | --- |
| Variables: | Subs | Variables: |
|  | Subs | Variables: |

Design process

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
| Sps are pos-def-fcns. |  |

Therefore, the ctrl law is computed through the algebraic operations. Which is significantly simpler than dealing with PDEs.

PROBLEM 4 Backstepping Control

|  |  |  |
| --- | --- | --- |
| Find s.t. |  |  |
|  |  |  |

aaa

aaa

aaa

As a result of those operations the control law is computed to be

PROBLEM 4 sliding mode control

|  |  |  |
| --- | --- | --- |
| Find s.t. |  |  |
|  |  |  |

aaa

aaa

By using this, the sliding manifold is computed as

Explicitly it can be expressed as

Recall, is the general s-dyns that is used to obtain the control law.

Let us look at the s-dyns

Let us gather the “known terms” together and rename them,

Now we have form that is easy to manipulate.

Let us gather the “unknown terms” together and rename them,

Recall, the objective is to have a resulting s-dyns of

The uncertain term is also known to be bounded since , this info can be used to determine the upper bound for , this can be represented as

As a result of this, the s-dyns can be represented as

To stabilize this dyns, the proposed control law is

|  |
| --- |
| Question |
| Find such that s-dyns are stable [finite time stability].  Answer: |
| Let us define  Therefore,  The uncertainty term can be expressed as  [idk how to prove than other than appealing to intuition, but there must be “uncertain term algebra” type of course to make these operations more systematic, [maybe there is and idk, but if there is no such a course, there should be]]  The TERM should be negative, therefore  Therefore, the suggested control law has the ability to drive the states onto the sliding manifold in finite time. |

Let us determine term explicitly,

Let us determine the condition on

Finally, the control law is written as

Ctrl law can be represented as

PROBLEM 4 systematizing the problem

|  |  |  |
| --- | --- | --- |
| Find s.t. |  |  |
|  |  |  |

aaa

aaa

|  |  |
| --- | --- |
|  | Choose |
|  | Distribute over the parenthesis |
|  | Isolate the term with |
|  | Arrange the terms st |
|  | Find a known upper bound for LHS |
|  | “upper” is a made-up operator: |
|  | If there is no unc-term that is left in the LHS, turn the ineq to eq |
|  | Solve for |
|  | Define for simplicity |
|  |  |

Focus on time-der-phi1

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Focus on u-term

|  |  |
| --- | --- |
|  | Choose |
|  | Subs explicit form of |
|  | Gather the known-terms in parenthesis and rename them |
|  | distribute over the parenthesis |
|  | Isolate v-term |
|  | Put it in the  v-termother terms |
|  | Gather the unc-terms |
|  | Use “upper-operator” for LHS |
|  | |
|  | Fcn of upper-operator: |
|  | Use to isolate v |
|  | Use to isolate v |
|  | Turn the ineq to eq to find v |

Finally,

By using

By using the explicit form of

This control law can be checked by simulating the following dynamical system,

|  |
| --- |
| Matlab code [to simulate the CL-system] |
| clear all,close all,clc;  %%  a=1;b=2;delta\_1=0.1;delta\_2=-0.2;  %% single run  % % % tspan = [0 5]; x0 = [1 1];  % % % [t,x] = ode45(@(t,x) odefcn(t,x,a,b,delta\_1,delta\_2), tspan, x0);  % % % plot(x)  %% multiple runs  tspan = [0 5];  x0 = [1 2];  fig1=figure(1);fig1.Color=[1,1,1];  ax1=axes('Parent',fig1);  fig2=figure(2);fig2.Color=[1,1,1];  ax2=axes('Parent',fig2);  %%  for ii=1:1:10  % plot(randn(100,1))  sigma=0.2; mu=0;  delta\_1=mu+randn(1)\*sigma; if(abs(delta\_1)>1) delta\_1=0.97; end  delta\_2=mu+randn(1)\*sigma; if(abs(delta\_2)>1) delta\_2=0.97; end  [t,x] = ode45(@(t,x) odefcn(t,x,a,b,delta\_1,delta\_2), tspan, x0);  x1\_vec=x(:,1); x2\_vec=x(:,2);  set(0, 'currentfigure', fig1);  set(fig1, 'currentaxes', ax1);  plot(t,x1\_vec,"Color",[1,0,0],"LineStyle","--","LineWidth",[1],"Parent",ax1); hold on;  plot(t,x2\_vec,"Color",[0,0,1],"LineStyle","-","LineWidth",[1],"Parent",ax1); hold on;  set(0, 'currentfigure', fig2);  set(fig2, 'currentaxes', ax2);  plot(x1\_vec,x2\_vec,"Color",[0,0,1],"LineStyle","-","LineWidth",[1],"Parent",ax2); hold on;  end  % ax1 properties  ax1.TickLabelInterpreter='latex';  ax1.XLabel.Interpreter='latex';  ax1.YLabel.Interpreter='latex';  ax1.XLabel.String='time';  ax1.YLabel.String='x(t)';  ax1.YLabel.FontSize=[20];  ax1.FontSize=[20];  ax1.XGrid='on';  ax1.YGrid='on';  % ax2 properties  ax2.TickLabelInterpreter='latex';  ax2.XLabel.Interpreter='latex';  ax2.YLabel.Interpreter='latex';  ax2.XLabel.String='$x\_{1}(t)$';  ax2.YLabel.String='$x\_{2}(t)$';  ax2.YLabel.FontSize=[20];  ax2.FontSize=[20];  ax2.XGrid='on';  ax2.YGrid='on';  legend(ax1,{'$x\_{1}(t)$','$x\_{2}(t)$'},"Interpreter","latex");  %%  function xdot = odefcn(t,x,a,b,delta\_1,delta\_2)  xdot = zeros(2,1);  x\_1=x(1); x\_2=x(2);  k=a+1;  u=[-[sign([x\_2+k\*x\_1])]\*[abs([x\_2+k\*x\_1])+b\*(x\_2)^2+a\*k\*abs(x\_1)]]-[2\*x\_1+k\*x\_2];  xdot(1) = x\_2+a\*delta\_1\*x\_1;  xdot(2) = b\*delta\_2\*(x\_2)^2+x\_1+u;  end |

|  |  |
| --- | --- |
| Matlab code output[CL-system simulation results] | |
|  |  |