現代控制理論報告

104303206黃筱晴

[新增內容]

1. Q2 Lyap.控制器

重新設計v(x)=0.5\*(x12+x22+x32)，恆≧0。

則v'=x1 x1'+x2 x2'+x3 x3'≡α(x)+β(x)\*u

設計u=(α(x)+1)/β(x)使v'恆等於-1。

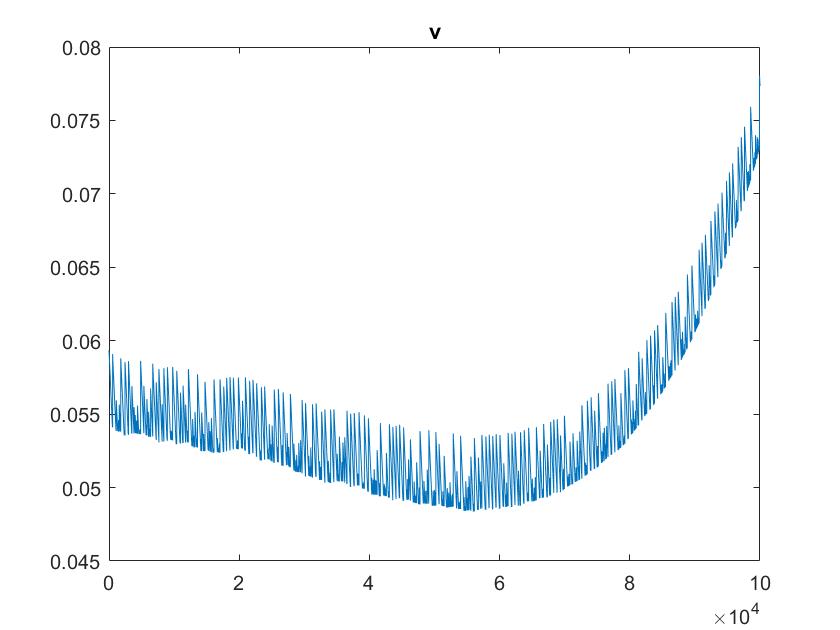
因為當u(x)分母趨近零時會導致發散，所以加入判斷式讓u達飽和，也因此 造成圖中的不連續的部分。

if abs(u(i))>10000

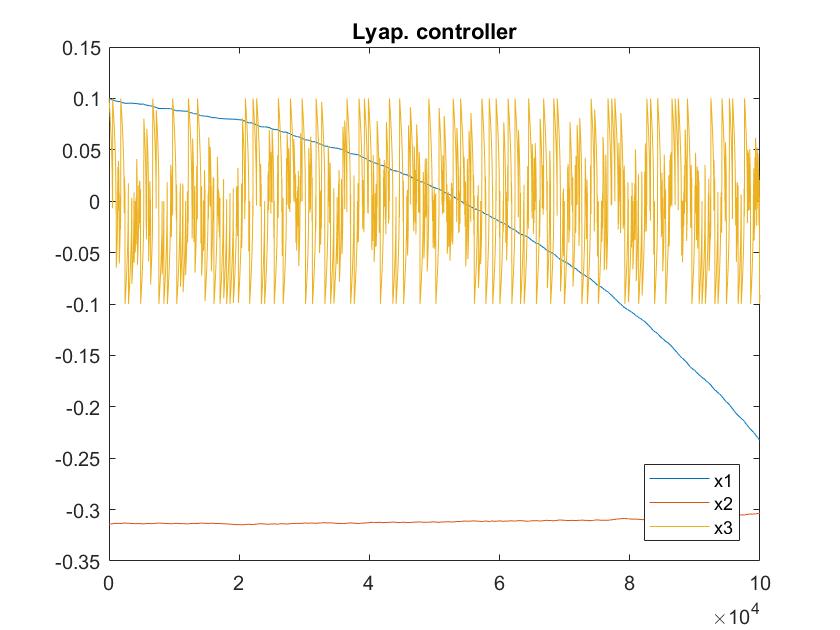
u(i)=10000\*sign(u(i));

fprintf('DANGER !!! i=%f\n',i);

end



前半段在鬆開u後還勉強控的回來，v大致以固定斜率遞減，但是後來鬆開 太多次後就控不下來了，v越來越大，最後系統還是發散。



Q2 FB linearization 控制器

設計Z'=[]Z

pole1=[1 8];pole2=[1 6];pole3=[1 7];

char\_poly=conv(pole1,conv(pole2,pole3));%(s+p1)\*(s+p2)\*(s+p3)=0

設計pole位置:-6,-7,-8 ，

由程式算出a1=- char\_poly (4)、a2= - char\_poly (3)、a3=- char\_poly (2)

令z1=x1-x3；z2=z1'；z3=z2'

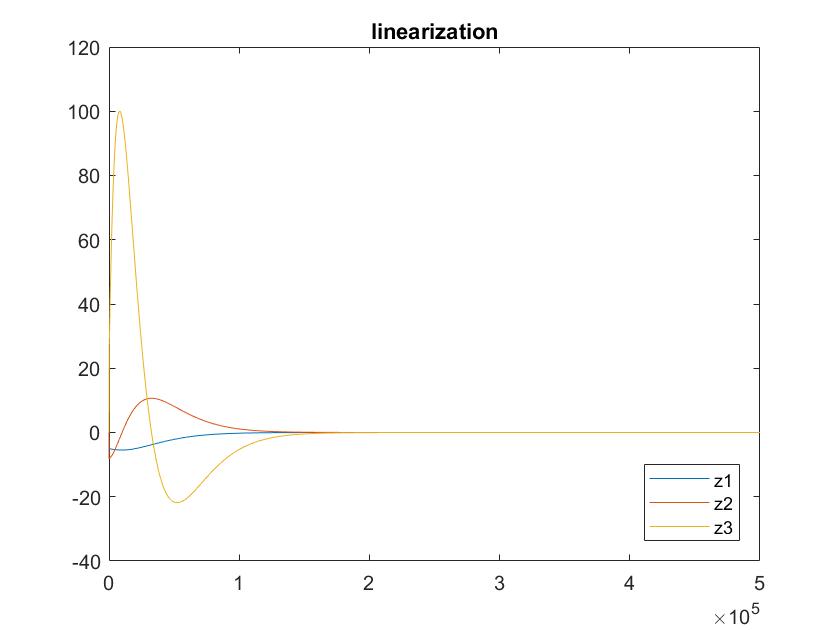
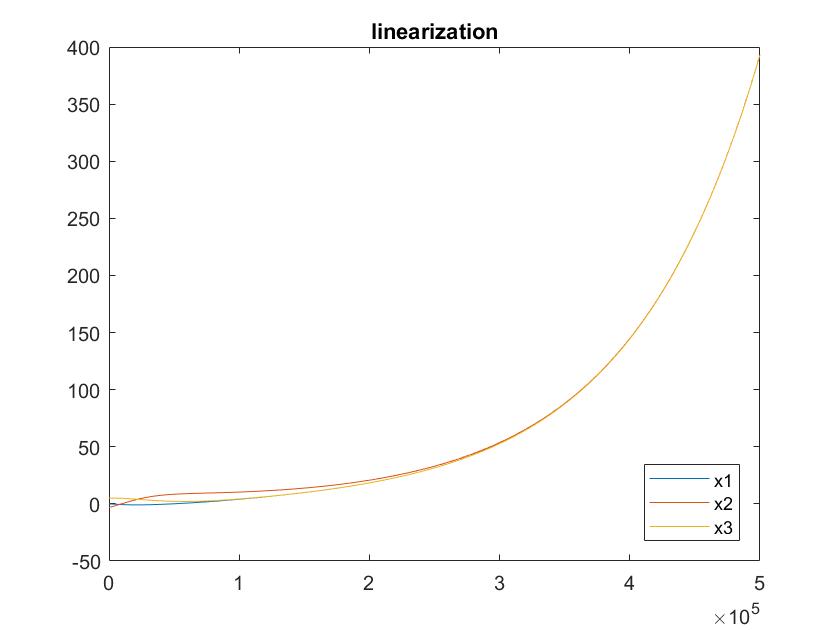
使用z3'來設計控制器，將題目給的系統帶入計算後得

z3'=sin(x1-x3)+u+2z1z2+z3-u''，設計u''使z3'=a1z1+a2z2+a3z3

tmp=-char\_poly(4)\*z1(i)-char\_poly(3)\*z2(i)-char\_poly(2)\*z3(i);

u\_dot\_dot(i)=sin(x1-x3)+u(i-2)+2\*z1(i)\*z2(i)+z3(i)-tmp;

這裡步數好像會有點問題(u\_dot\_dot(i), u\_dot(i-1), u(i-2)?)，所以把數值逼近的 delta調到很小(delta=0.00001)讓這個問題不要造成太大影響。



z成功以線性系統的方式收斂下來。但是因為z設計的關係，最後x是跑到

(x1-x3)=0的平面上，然後x2沒有控到...

程式碼

%linearization

clc;clear;

delta=0.00001;

totalTime=5;

totalStep=totalTime/delta;

pole1=[1 8];pole2=[1 6];pole3=[1 7];

char\_poly=conv(pole1,conv(pole2,pole3));%(s+p1)\*(s+p2)\*(s+p3)=0

x1array=[1:totalStep]\*0;x2array=x1array;x3array=x1array;

u=x1array;u\_dot=x1array;u\_dot\_dot=x1array;

z1=x1array;z2=x1array;z3=x1array;

x1\_dot=x1array;x2\_dot=x1array;x2\_dot=x1array;

x1array(1)=0;x2array(1)=-pi;x3array(1)=5;%init condition

x1array(2)=0;x2array(2)=-pi;x3array(2)=5;%init condition

x1array(3)=0;x2array(3)=-pi;x3array(3)=5;%init condition

for i=3:totalStep

x1=x1array(i);x2=x2array(i);x3=x3array(i);

x1\_dot(i)=x2+x1-x3+sin(x1-x3);

x2\_dot(i)=x3+(x1-x3)^2;

x3\_dot(i)=sin(x1-x3)+u(i-2);

z1(i)=x1-x3;

z2(i)=x1\_dot(i)-x3\_dot(i);

z3(i)=x3+z1(i)^2+z2(i)-u\_dot(i-1);

tmp=-char\_poly(4)\*z1(i)-char\_poly(3)\*z2(i)-char\_poly(2)\*z3(i);

u\_dot\_dot(i)=sin(x1-x3)+u(i-2)+2\*z1(i)\*z2(i)+z3(i)-tmp;

x1array(i+1)=x1+x1\_dot(i)\*delta;

x2array(i+1)=x2+x2\_dot(i)\*delta;

x3array(i+1)=x3+x3\_dot(i)\*delta;

u\_dot(i)=u\_dot(i-1)+u\_dot\_dot(i)\*delta;

u(i-1)=u(i-2)+u\_dot(i-1)\*delta;

end

figure(1);

plot(x1array);hold on;

plot(x2array);hold on;

plot(x3array);legend('x1','x2','x3','location','southeast');

title('linearization');

figure(2);

plot(z1);hold on;

plot(z2);hold on;

plot(z3);hold on;legend('z1','z2','z3','location','southeast');

title('linearization');

figure(3);

plot(x1array-x3array);hold on;legend('x1-x3','location','southeast');

title('linearization');

figure(4);

time=1:totalStep-2;

xt=z1(time);yt=z2(time);zt=z3(time);

plot3(xt,yt,zt);hold on;

grid on;title('z1 z2 z3 phase portrait');

Q3 MRAC

因為MRAC控制器快追到modle之後過久了還是會發散，所以嘗試：

1. 500筆後就不再重新設計u。結果它很快就發散了。

2.Offline操作，第500步資料後，每30000步中只有前500步更新控制器。

if k<=500

u(k)=theta0(k)\*r(k)+theta1(k)\*x1(k)+theta2(k)\*x2(k)+theta3(k)\*x3(k);

end

if k>500

if mod(k,30000)<500

u(k)=theta0(k)\*r(k)+theta1(k)\*x1(k)+theta2(k)\*x2(k)+theta3(k)\*x3(k);

else

u(k)=u(k-1);

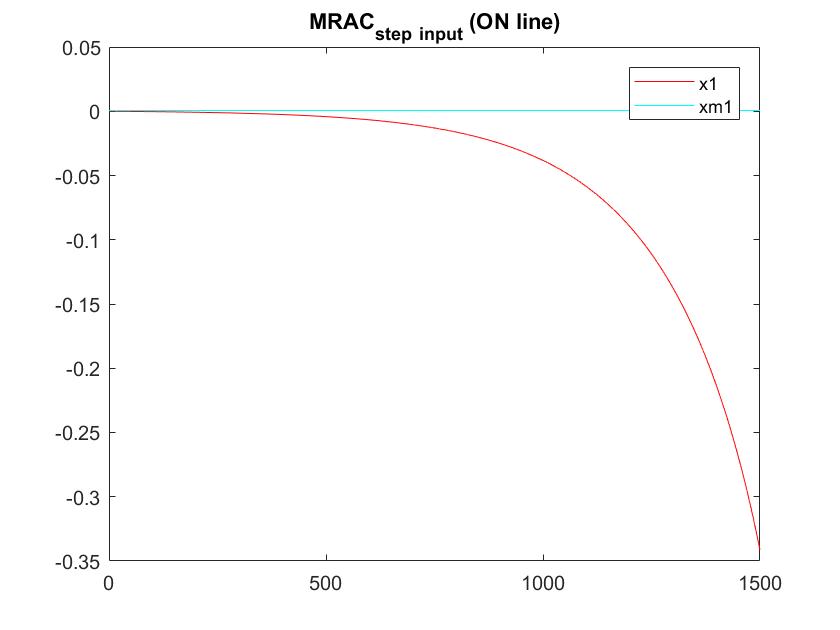
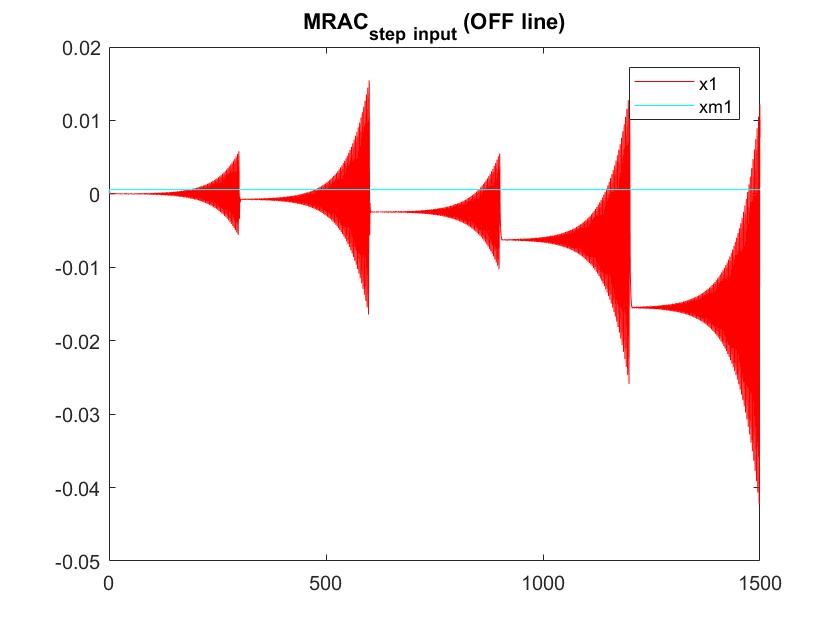
end

end

totaltime=1500;delta=0.01;

一直Online的MRAC控制器，1500秒時，發散程度大概 |0.03|。

Offline的大概|0.004|。性能有比較好一點。



1.

考慮三階系統T(s)==(s3+a1s2+a2s+a3)-1，極點為使(s3+a1s2+a2s+a3)=0的s。

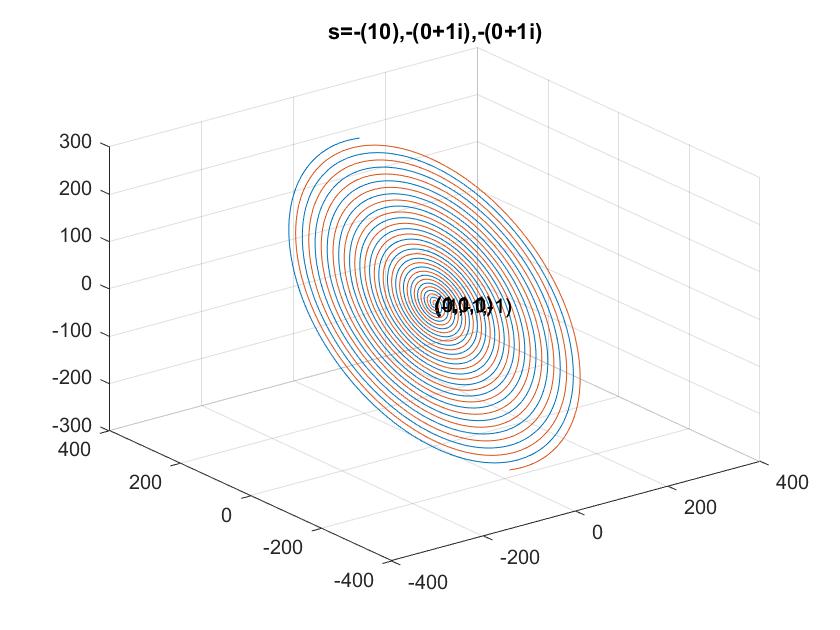
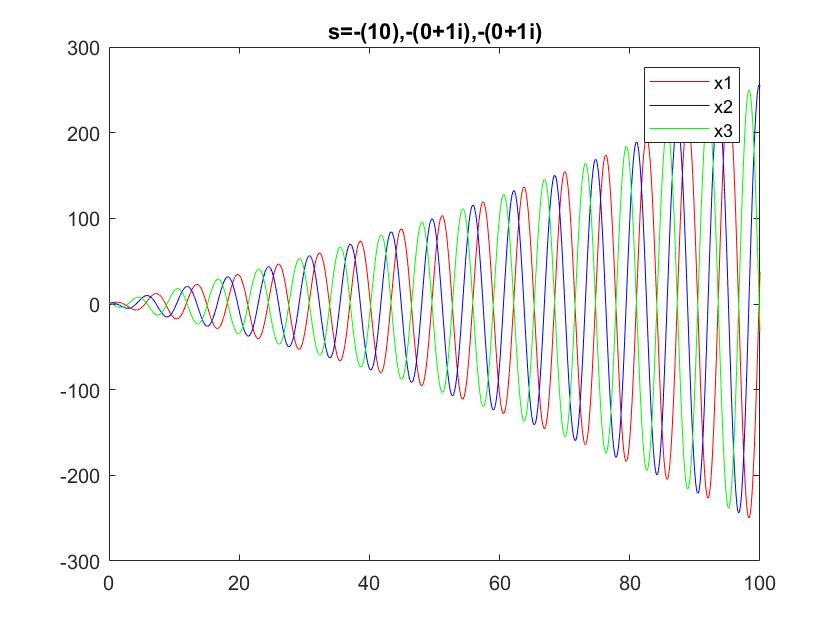
則此系統的微分方程式為：y'''+a1y''+a2y'+a3y=u

令x1=y ; x2=x1'=y' ; x3=x2'=y'' ;

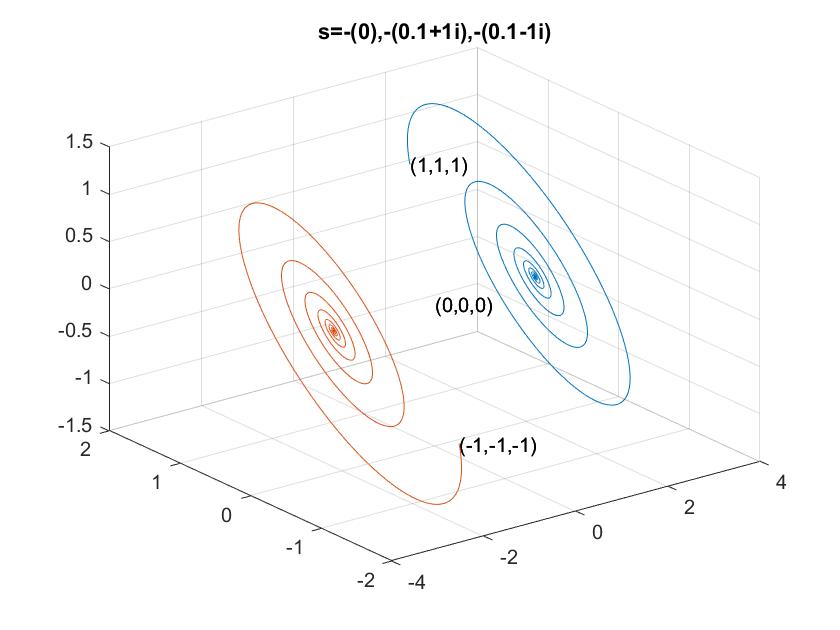
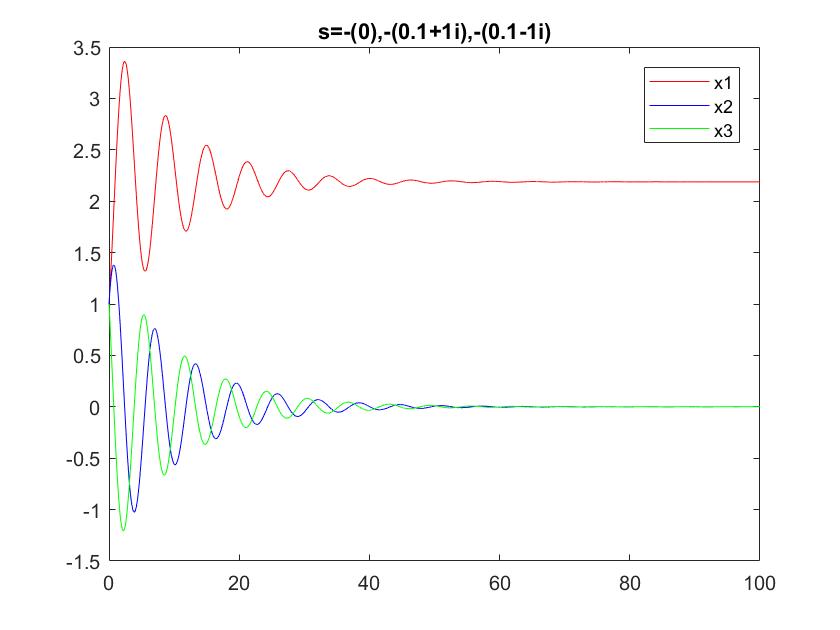
則x3'=y'''=-a1y''-a2y'-a3y+u=-a3x1-a2x2-a1x3+u

u=0則有 []=[][]

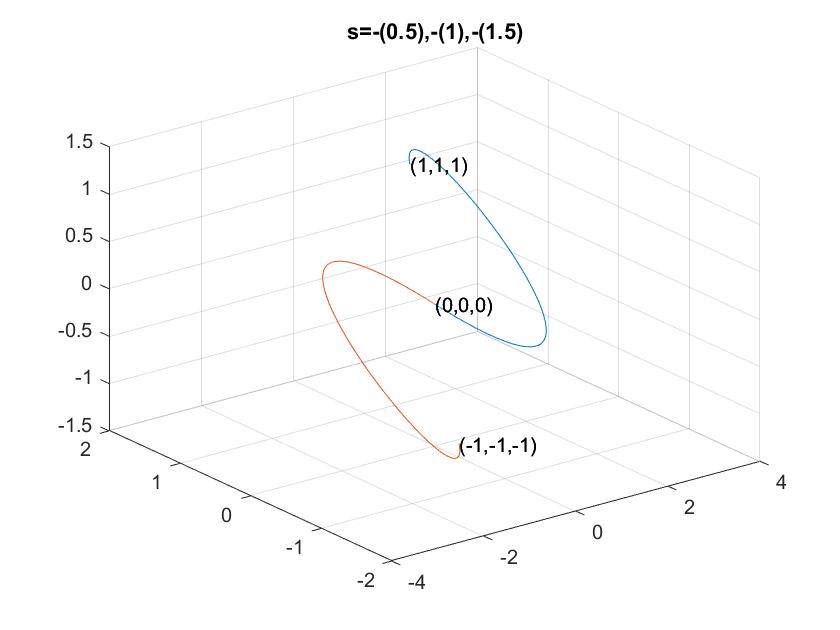
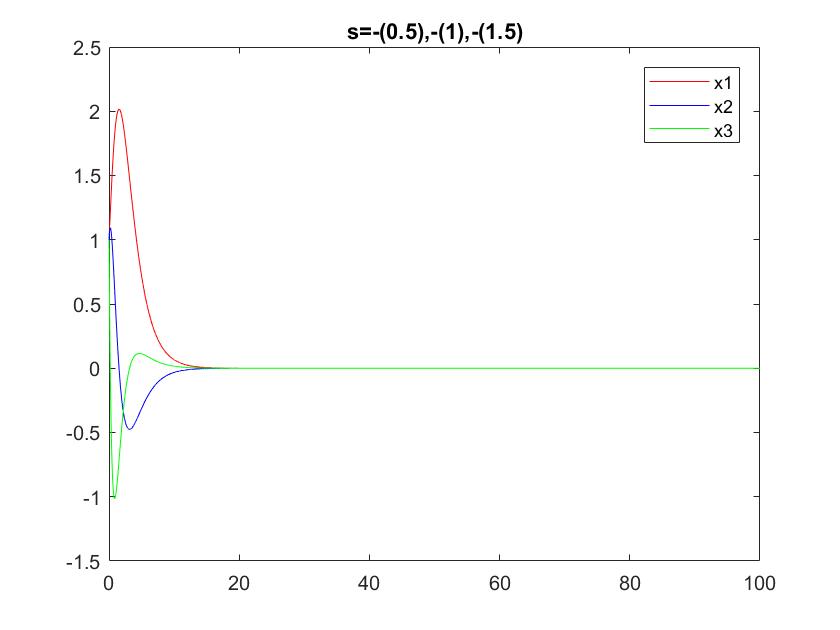
(1)虛軸上有≧2極點：系統發散



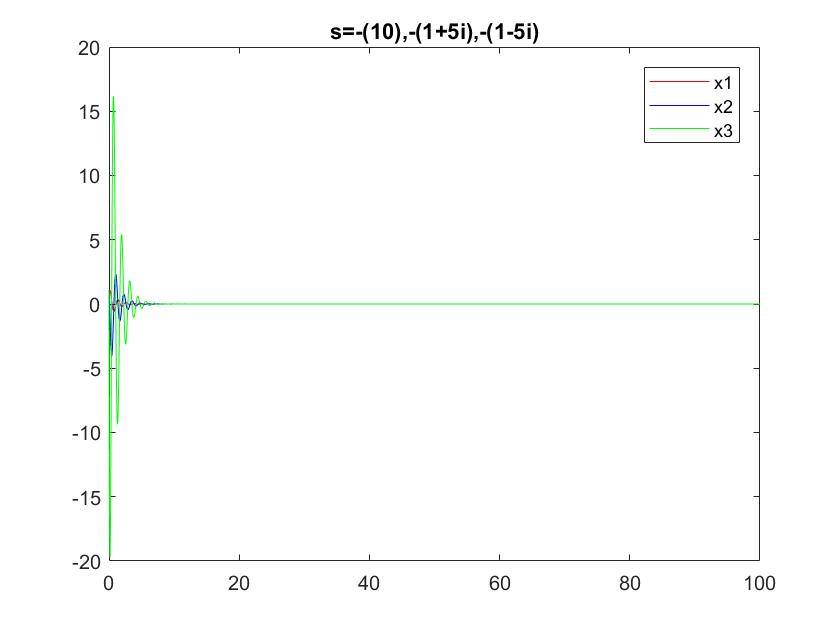
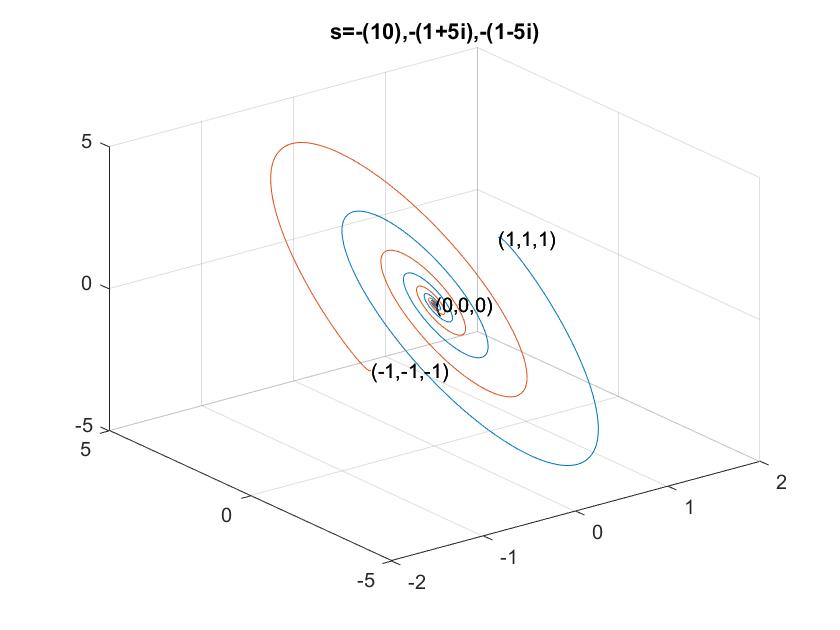
(2)虛軸上有1極點



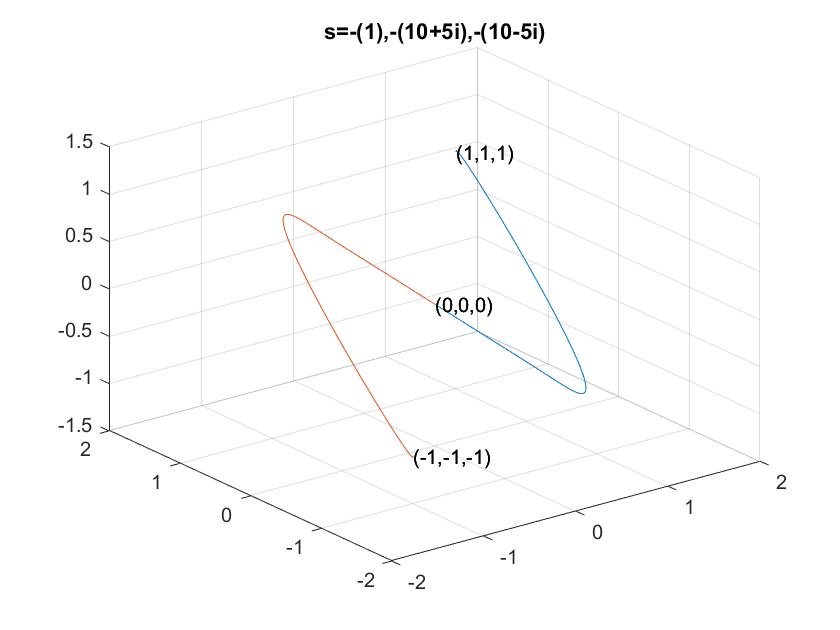
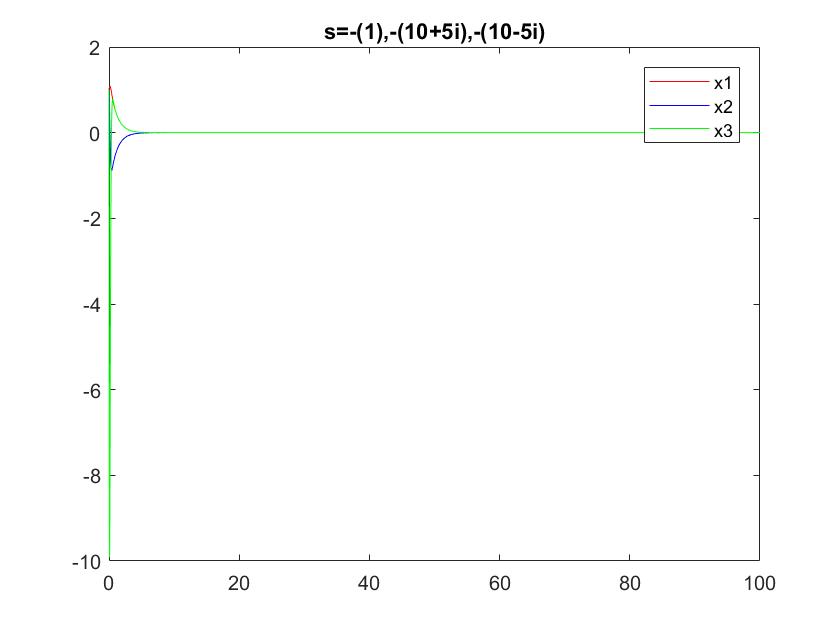
(3)3實根



(4)共軛虛根為主極點



(5)共軛虛根不是主極點



程式碼

clear;clc;

totaltime=100;

delta=0.01;

totalstep=totaltime/delta;

pole1=[1 0];pole2=[1 0.1-2i];pole3=[1 0.1+2i];

char\_poly=conv(pole1,conv(pole2,pole3));%(s+p1)\*(s+p2)\*(s+p3)=0

A=[0 1 0;0 0 1;-char\_poly(4) -char\_poly(3) -char\_poly(2)];%x\_dot=A\*x

IC=[1,1,1;-1,-1,-1];%initial condition

for i=1:2

x1=[1:totalstep]\*0;x2=x1;x3=x1;

x1\_dot=x1;x2\_dot=x1;x3\_dot=x1;

x1(1)=IC(i,1);

x2(1)=IC(i,2);

x3(1)=IC(i,3);

fprintf('init. condidtion:(%d,%d,%d)\n',x1(1),x2(1),x3(1));

for k=1:totalstep

x1\_dot(k)=x2(k);

x2\_dot(k)=x3(k);

x3\_dot(k)=A(3,1)\*x1(k)+A(3,2)\*x2(k)+A(3,3)\*x3(k);

x1(k+1)=x1(k)+x1\_dot(k)\*delta;

x2(k+1)=x2(k)+x2\_dot(k)\*delta;

x3(k+1)=x3(k)+x3\_dot(k)\*delta;

end

if mod(i,2)==1

figure(1);

plot([0:1:totalstep]\*delta,x1,'r');hold on;

plot([0:1:totalstep]\*delta,x2,'b');hold on;

plot([0:1:totalstep]\*delta,x3,'g');hold on;

legend('x1','x2','x3');

str1=num2str(pole1(2));str2=num2str(pole2(2));str3=num2str(pole3(2));

str=['s=-(' str1 '),-(' str2 '),-(' str3 ')'];

title(str);

end

figure(2);

time=1:totalstep-1;

xt=x1(time);yt=x2(time);zt=x2(time);

plot3(xt,yt,zt);hold on;

text(0,0,0,'(0,0,0)');text(1,1,1,'(1,1,1)');text(-1,-1,-1,'(-1,-1,-1)');

grid on;

end

title(str);

fprintf(str);

2.

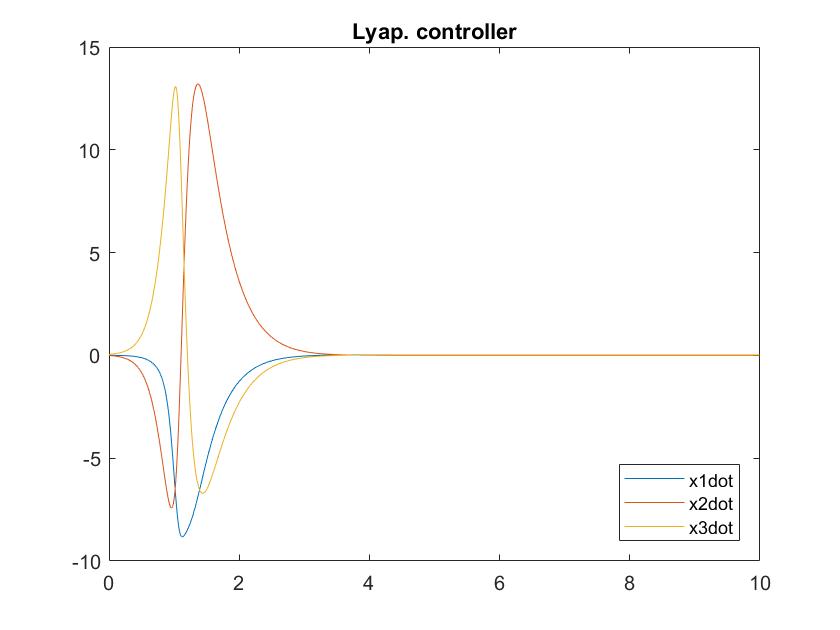
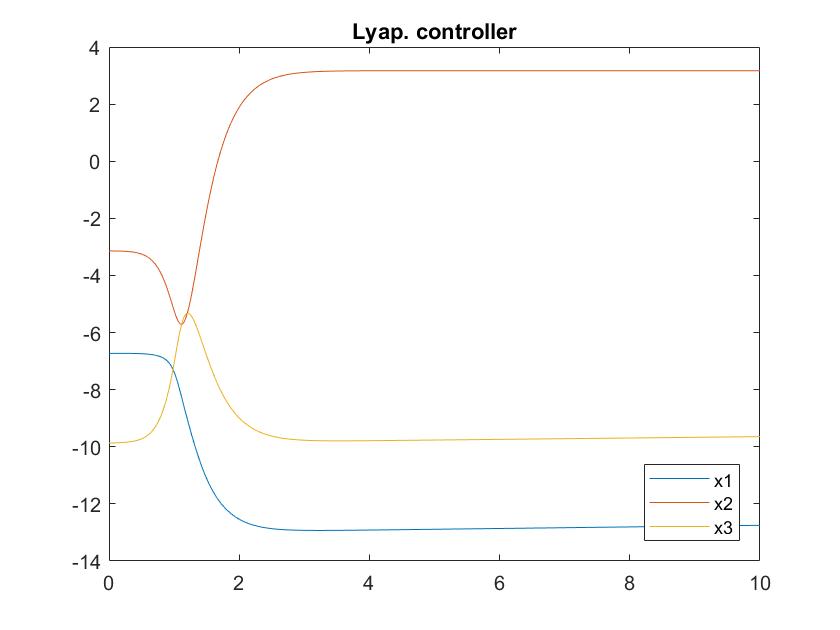
(1)Lyap. controller

令v(x)=0.5\*(x1+x2+x3)2，恆≧0。

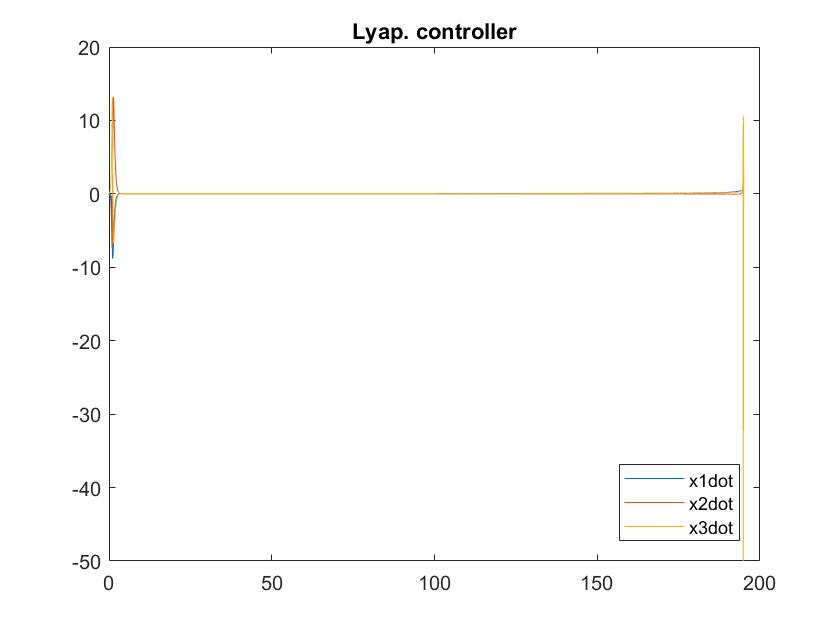
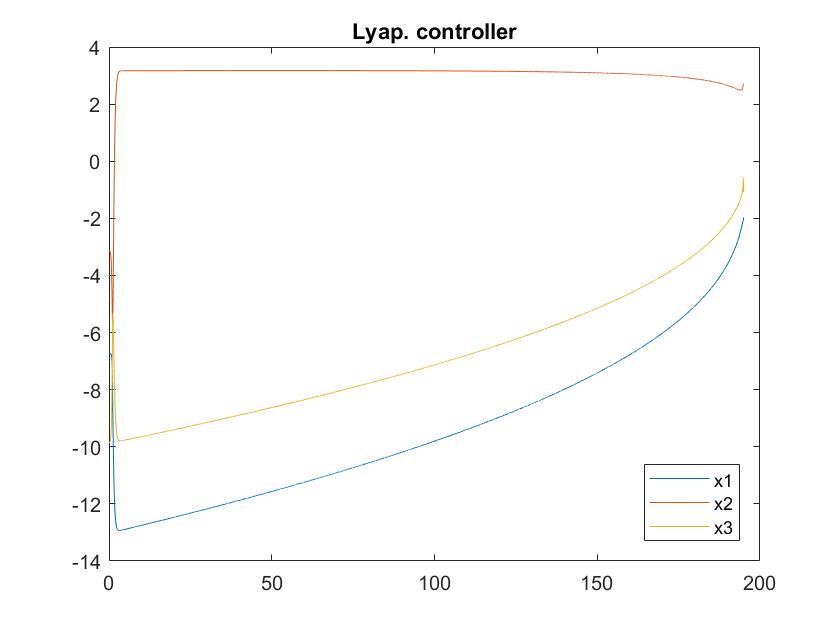
則v'=(x1+x2+x3)( x1'+x2'+x3')≡α(x)+β(x)\*u

設計u=(α(x)+1)/β(x)使v'恆等於-1。

這題超困難的，怎麼做都發散。後來嘗試先算出系統的其中一個平衡點 (x1,x2,x3)=(π-π2,-π,-π2)，直接將初值設在這裡。原本預期它會一直停 在這裡，但過了一陣子它又跑向另一個平衡點了。



它在下一個平衡過了一陣子後，開始衝向原點。但是因為u的設計在原 點會發散，所以加入了一些判斷式適時將u鬆開，結果鬆開幾次系統就 再也控不下來了(大概在196秒發散)



程式碼

%lyap.

clc;clear;

delta=0.01;

totalTime=200;

totalStep=totalTime/delta;

x1array=[1:totalStep]\*0;x2array=x1array;x3array=x1array;

x1\_dot=x1array;x2\_dot=x1array;x2\_dot=x1array;

x1array(1)=pi-(pi)^2;x2array(1)=-pi;x3array(1)=-pi^2;%init condition

for i=1:totalStep

x1=x1array(i);x2=x2array(i);x3=x3array(i);

v(i)=0.5\*(x1+x2+x3)^2;

u(i)=(-(x2+x1+2\*sin(x1-x3)+(x1-x3)^2)-1/(x1+x2+x3));

v\_dot(i)=(x1+x2+x3)\*(x2+x1+2\*(sin(x1-x3))+(x1-x3)^2+u(i));

fprintf('i=%d v=%f v\_dot=%f u=%f\n',i,v(i),v\_dot(i),u(i));

if v(i)<0.001

u(i)=0;

fprintf('DANGER1 !!!\n');

end

if abs(u(i))>10^10

u(i)=10^10\*sign(u(i));

fprintf('DANGER2 !!!\n');

end

x1\_dot(i)=x2+x1-x3+sin(x1-x3);

x2\_dot(i)=x3+(x1-x3)^2;

x3\_dot(i)=sin(x1-x3)+u(i);

x1array(i+1)=x1+x1\_dot(i)\*delta;

x2array(i+1)=x2+x2\_dot(i)\*delta;

x3array(i+1)=x3+x3\_dot(i)\*delta;

end

figure(1);

plot([0:1:totalStep]\*delta,x1array);hold on;

plot([0:1:totalStep]\*delta,x2array);hold on;

plot([0:1:totalStep]\*delta,x3array);legend('x1','x2','x3','location','southeast');

title('Lyap. controller');

figure(2);

plot([0:1:totalStep-1]\*delta,x1\_dot);hold on;

plot([0:1:totalStep-1]\*delta,x2\_dot);hold on;

plot([0:1:totalStep-1]\*delta,x3\_dot);legend('x1dot','x2dot','x3dot','location','southeast');

title('Lyap. controller');

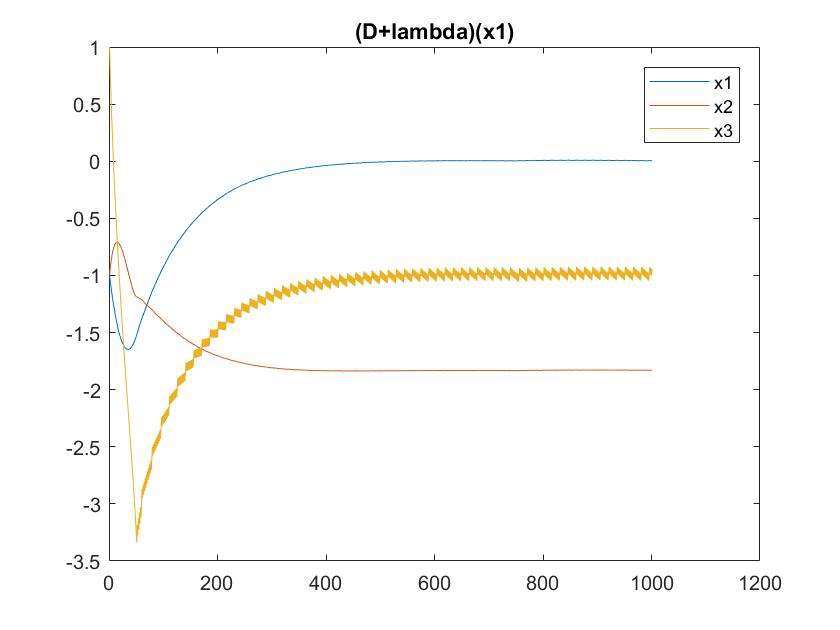
(2)sliding control

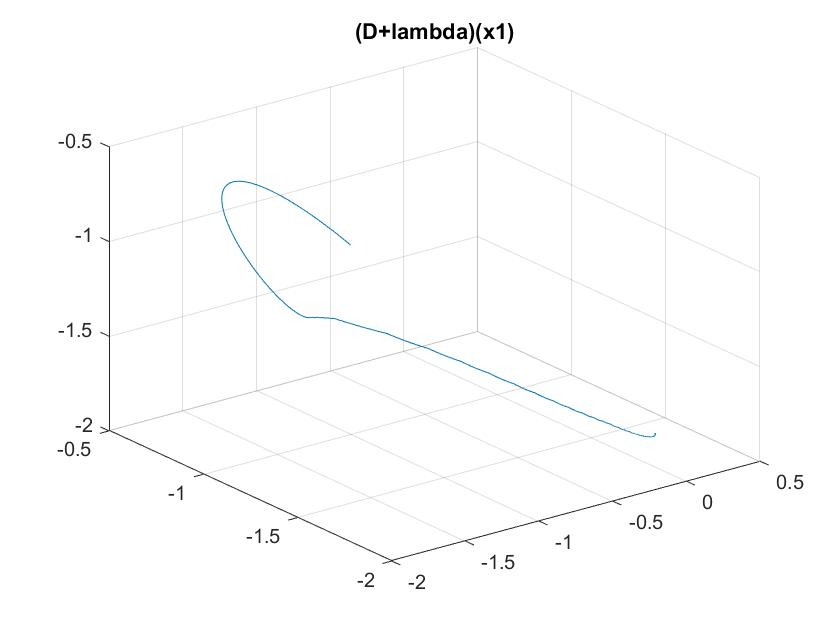
<1>

令f= (D+λ)(x1)= x1'+λx1=x2+x1-x3+sin(x1-x3)+λx1;

則f'= x2'+x1'-x3'+(sin(x1-x3))'+λx1'≡α(x)+β(x)\*u;

設計u使f'=-K\*sign(f)，讓f以e-λt收斂。最後收斂到x1=0。



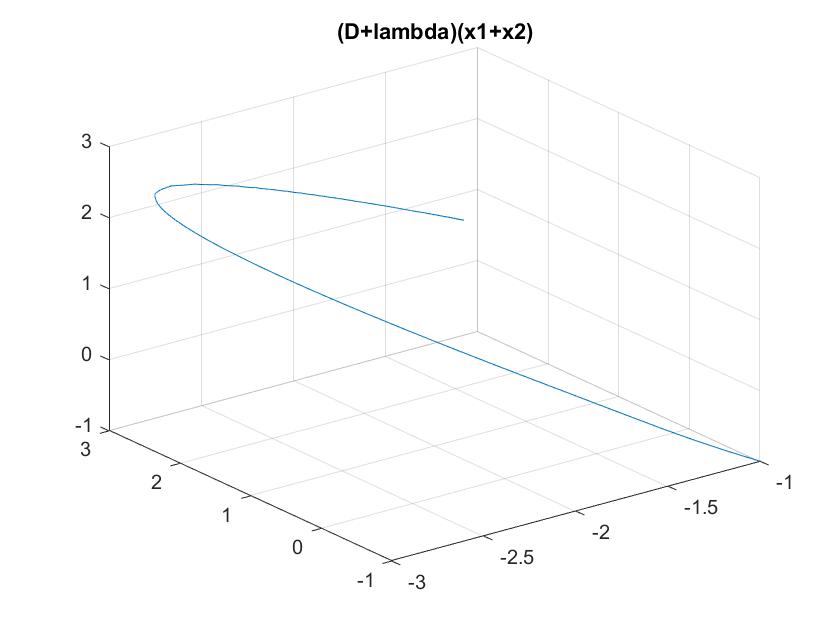
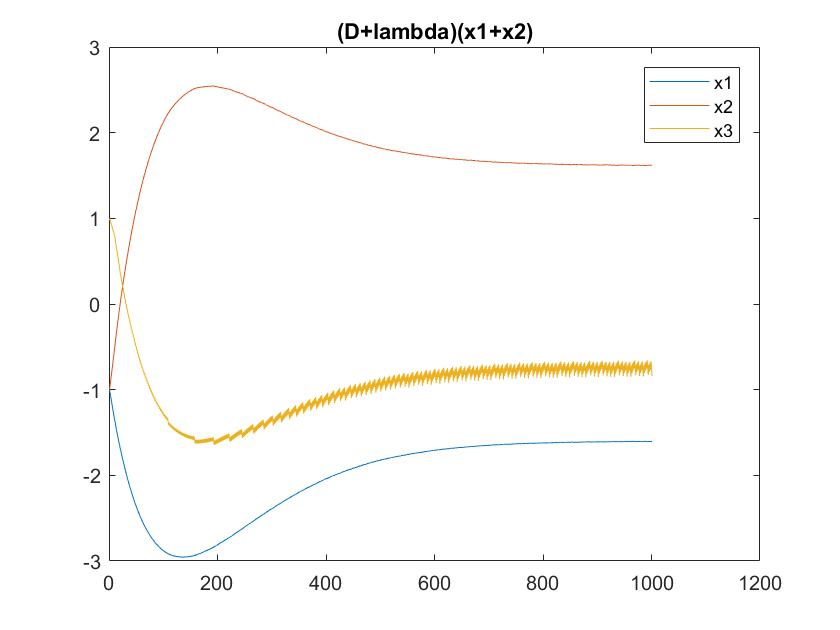


<2>

令f=(D+λ)(x1+x2)=x1+x2+sin(x1-x3)+(x1-x3)2+λ(x1+x2);

則f'= x1'+x2'+(sin(x1-x3))'+2(x1-x3)(x1'-x3')+λ(x1'+x2');≡α(x)+β(x)\*u;

設計u使f'=-K\*sign(f)，讓f以e-λt收斂。最後收斂到x1+x2=0。



程式碼

%f=(D+lambda)(x1+x2)

clc;clear;

lambda=1;

K=10;

delta=0.01;

totalTime=10;

totalStep=totalTime/delta;

x1array=[1:totalStep]\*0;x2array=x1array;x3array=x1array;

x1\_dot=x1array;x2\_dot=x1array;x2\_dot=x1array;

x1array(1)=-1;x2array(1)=-1;x3array(1)=1;%init condition

for i=1:totalStep

x1=x1array(i);x2=x2array(i);x3=x3array(i);

f(i)=x1+x2+sin(x1-x3)+(x1-x3)^2+lambda\*(x1+x2);

u(i)=(K\*sign(f(i))+(1+lambda)\*((x2+x1-x3+sin(x1-x3))+(x3+(x1-x3)^2))+(cos(x1-x3)+2\*(x1-x3))\*(x2+x1-x3))/(cos(x1-x3)+2\*(x1-x3));

x1\_dot(i)=x2+x1-x3+sin(x1-x3);

x2\_dot(i)=x3+(x1-x3)^2;

x3\_dot(i)=sin(x1-x3)+u(i);

x1array(i+1)=x1+x1\_dot(i)\*delta;

x2array(i+1)=x2+x2\_dot(i)\*delta;

x3array(i+1)=x3+x3\_dot(i)\*delta;

end

figure(1);

plot(x1array);hold on;

plot(x2array);hold on;

plot(x3array);legend('x1','x2','x3');

title('(D+lambda)(x1+x2)')

figure(2);

time=1:totalStep-1;

xt=x1array(time);

yt=x2array(time);

zt=x2array(time);

plot3(xt,yt,zt);

grid on;title('(D+lambda)(x1+x2)')

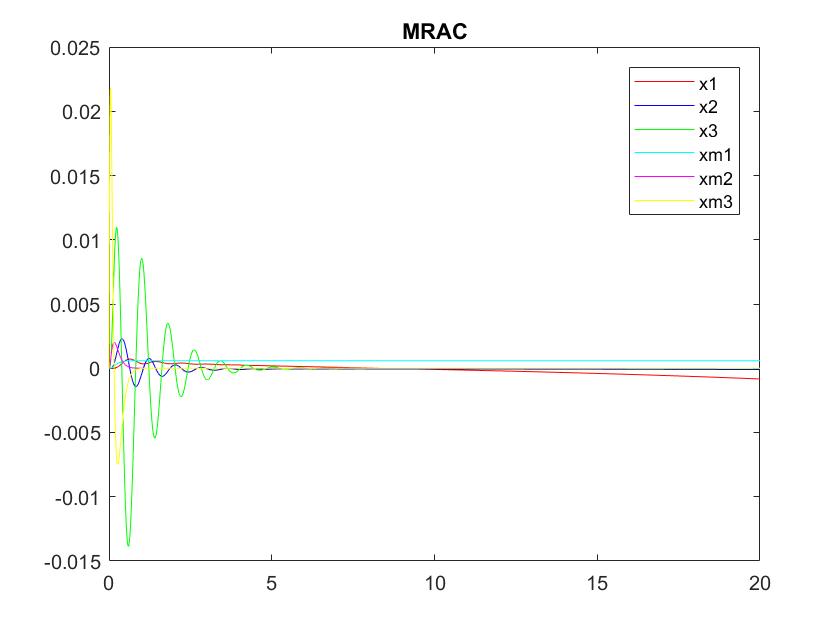
3.

設計modle的極點為s=-11,s=-12,s=-13，要離虛軸夠遠系統才不會發散。

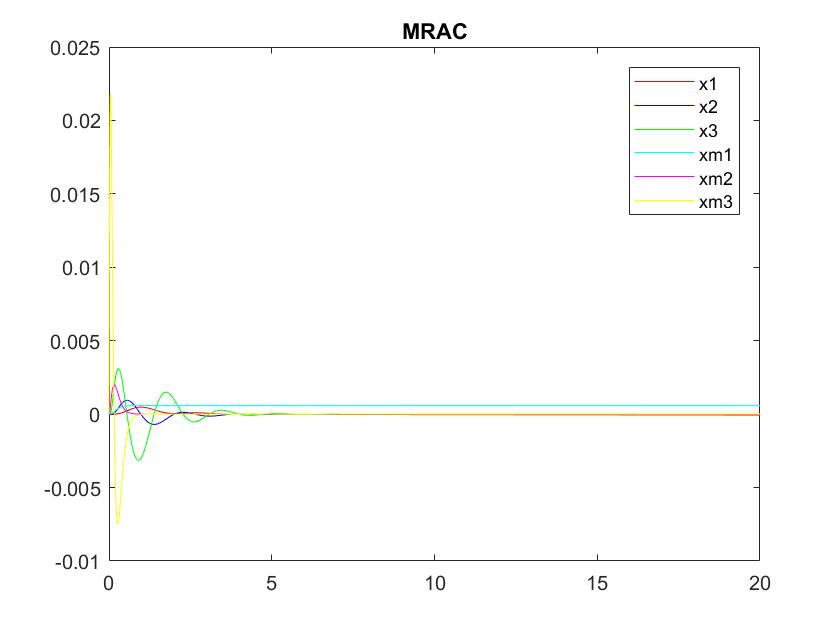
經計算得Am=[]

(1)r=unit step

一開始先預設Q為單位矩陣，γ皆為1。

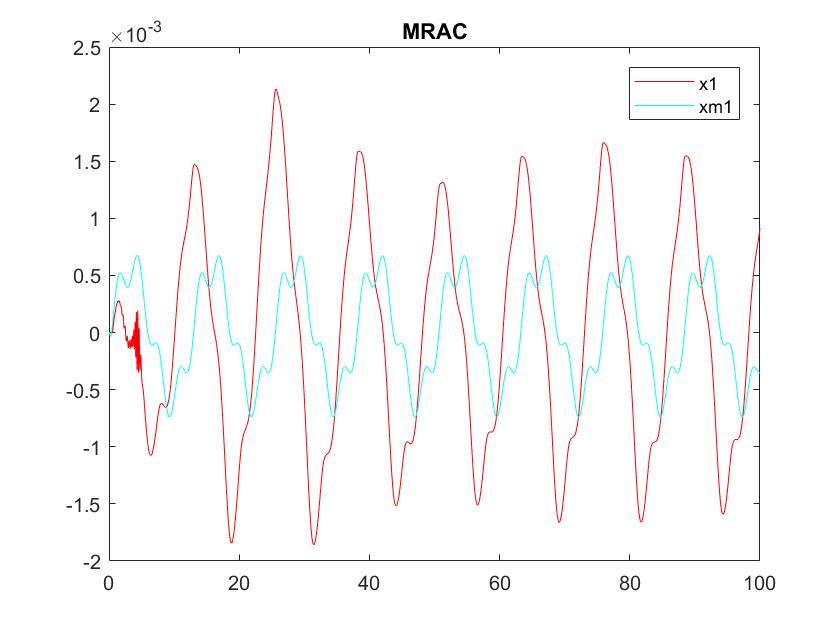


發現它快追上之後又會垂下來，其中x1的誤差最嚴重而且越來越大，最 後導致發散。故調整權重，改成Q=[]，也試過調γ 但是效果不彰。



(2)r=弦波的合成

再改成Q=[]，γ仍為1。



程式碼

%MRAC

clear;clc;

totaltime=100;

delta=0.01;

totalstep=totaltime/delta;

%select para.

Q=[0.00001 0 0;0 0.00001 0;0 0 100000];

pole=conv([1 13],conv([1 12],[1 11]));

Am=[0 1 0;0 0 1;-pole(4) -pole(3) -pole(2)];bm=1;%model

A=[0 1 0;0 0 1;-12 -4 -3];b=1;%real sys.

P=lyap(Am,Q);

gamma0=1;gamma1=1;gamma2=1;gamma3=1;

%model

xm1(1)=0;xm2(1)=0;xm3(1)=0;

for k=1:totalstep

% r(k)=1;

r(k)=sin(0.5\*k\*delta)+0.3\*cos(2\*k\*delta+4);

xm1\_dot(k)=xm2(k);

xm2\_dot(k)=xm3(k);

xm3\_dot(k)=Am(3,1)\*xm1(k)+Am(3,2)\*xm2(k)+Am(3,3)\*xm3(k)+bm\*r(k);

xm1(k+1)=xm1(k)+xm1\_dot(k)\*delta;

xm2(k+1)=xm2(k)+xm2\_dot(k)\*delta;

xm3(k+1)=xm3(k)+xm3\_dot(k)\*delta;

end

%real sys.

theta0(1)=0;theta1(1)=0;theta2(1)=0;theta3(1)=0;

x1(1)=0;x2(1)=0;x3(1)=0;

for k=1:totalstep

u(k)=theta0(k)\*r(k)+theta1(k)\*x1(k)+theta2(k)\*x2(k)+theta3(k)\*x3(k);

x1\_dot(k)=x2(k);

x2\_dot(k)=x3(k);

x3\_dot(k)=A(3,1)\*x1(k)+A(3,2)\*x2(k)+A(3,3)\*x3(k)+1\*u(k);

x1(k+1)=x1(k)+x1\_dot(k)\*delta;

x2(k+1)=x2(k)+x2\_dot(k)\*delta;

x3(k+1)=x3(k)+x3\_dot(k)\*delta;

e1(k)=xm1(k)-x1(k);

e2(k)=xm2(k)-x2(k);

e3(k)=xm3(k)-x3(k);

zeta(k)=0.5\*(P(1,3)\*e1(k)+P(2,3)\*e2(k)+P(3,3)\*e3(k));

theta0\_dot(k)=zeta(k)\*r(k)/(b\*gamma0);

theta1\_dot(k)=zeta(k)\*x1(k)/(b\*gamma1);

theta2\_dot(k)=zeta(k)\*x2(k)/(b\*gamma2);

theta3\_dot(k)=zeta(k)\*x3(k)/(b\*gamma3);

theta0(k+1)=theta0(k)+theta0\_dot(k)\*delta;

theta1(k+1)=theta1(k)+theta1\_dot(k)\*delta;

theta2(k+1)=theta2(k)+theta2\_dot(k)\*delta;

theta3(k+1)=theta3(k)+theta3\_dot(k)\*delta;

end

plot([0:1:totalstep]\*delta,x1,'r');hold on;

% plot([0:1:totalstep]\*delta,x2,'b');hold on;

% plot([0:1:totalstep]\*delta,x3,'g');hold on;

plot([0:1:totalstep]\*delta,xm1,'c');hold on;

% plot([0:1:totalstep]\*delta,xm2,'m');hold on;

% plot([0:1:totalstep]\*delta,xm3,'y');hold on;

legend('x1','xm1');

title('MRAC');

4.

C1(s)=、C2(s)=。

看起來C2抗雜訊功能較佳，只是前期震盪也較嚴重。

