Introduktion till
Control System Toolbox 5.0

This version: October 14, 2020

1 Inledning

Denna skrift är en kort inledning till hur MATLAB och Control System Toolbox (CST) används i kurserna i Reglerteknik.

2 System

I Control System Toolbox finns datastrukturer för att hantera s k *LTI-objects*, dvs linjära tidsinvarianta system, på ett bekvämt sätt. Vi kommer inledningsvis främst att arbeta med system på överföringsfunktionsform, men senare även med system på tillståndsform. Ett objekt som representerar ett system på överföringsfunktionsform skapas med funktionen `tf`. Detta kan göras på två olika sätt, och det första alternativet visas i exemplet nedan.

Betrakta överföringsfunktionen

$$G(s) = \frac{4}{s(s^2 + 2s + 4)}$$

Mata in systemet och ge objektet namnet `G`. Argumenten till funktionen `tf` utgörs av radvektorer innehållande täljarens respektive nämnarens koefficienter.

```
>> G = tf( 4, [ 1 2 4 0 ] )

Transfer function:
        4
-----------------
s^3 + 2 s^2 + 4 s
```

Med det andra alternativet kan man mata in överföringsfunktionen på symbolisk form genom att först skapa ett objekt bestående av symbolen `s`. Därefter kan man t ex addera och multiplicera med denna symbol på samma sätt som görs med Laplace-variabeln s vid handräkning.

Skapa ett objekt bestående av symbolen `s`. Bilda överföringsfunktionen genom att använda vanliga räkneoperationer.

```
>> s = tf( 's' );
>> G = 4 / ( s * ( s^2 + 2*s + 4 ) )

Transfer function:
        4
-----------------
s^3 + 2 s^2 + 4 s
```

En finess med överföringsfunktioner representerade som LTI-objekt är att man kan multiplicera och addera överföringsfunktioner på ett rättframt sätt.

Skapa en ny överföringsfunktion `G2` genom att seriekoppla $G(s)$ och överföringsfunktionen

$$\frac{1}{s+1}$$

```
>> G2 = G * 1 / ( s + 1 )

Transfer function:
            4
-------------------------
s^4 + 3 s^3 + 6 s^2 + 4 s
```

---

3 Poler och nollställen

Poler och nollställen till överföringsfunktioner beräknas med funktionerna `pole` respektive `tzero`. Poler och nollställen kan även ritas med funktionen `pzmap`.

Beräkna polerna till $G(s)$. Systemet har en reell pol i origo och två komplexa poler.

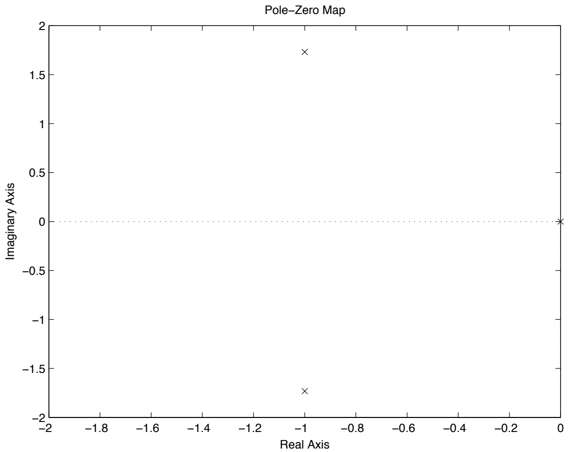
```
>> pole( G )

ans =

        0
  -1.0000 + 1.7321i
  -1.0000 - 1.7321i
```

Beräkna nollställena till $G(s)$. Eftersom täljaren i överföringsfunktionen är konstant saknar systemet nollställen.

```
>> tzero( G )

ans =

   Empty matrix: 0-by-1
```

Rita in systemets poler och nollställen i det komplexa talplanet. Poler markeras med kryss och nollställen, i de fall de förekommer, markeras med ringar.

```
>> pzmap( G )
>> axis([ -2 0 -2 2 ])
```



4 Återkoppling

I kursen behandlas återkopplade reglersystem enligt figur 1.

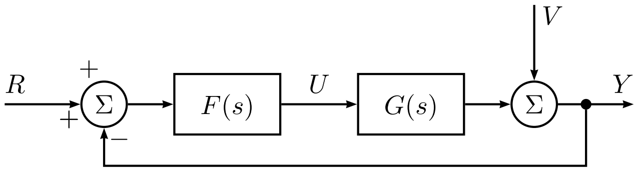


Figure 1. Reglersystem

Med systembeskrivningen

$$Y(s) = G(s)U(s) + V(s)$$

och återkopplingen

$$U(s) = F(s)(R(s) - Y(s))$$

ges det återkopplade systemet av

$$Y(s) = G_c(s)R(s) + S(s)V(s)$$

där

$$G_c(s) = \frac{F(s)G(s)}{1 + F(s)G(s)}$$

och

$$S(s) = \frac{1}{1 + F(s)G(s)}$$

Överföringsfunktionerna för det återkopplade systemet kan beräknas med funktionen `feedback`.

Generera överföringsfunktionen för en proportionell regulator med förstärkning $K_p = 0.7$.

```
>> F = tf( 0.7 )

Transfer function:
0.7
```

Beräkna överföringsfunktionen för det återkopplade systemet.

```
>> Gc = feedback( F * G, 1 )

Transfer function:
          2.8
-----------------------
s^3 + 2 s^2 + 4 s + 2.8
```

Beräkna känslighetsfunktionen.

```
>> S = 1 / ( 1 + F * G )

Transfer function:
   s^3 + 2 s^2 + 4 s
-----------------------
s^3 + 2 s^2 + 4 s + 2.8
```

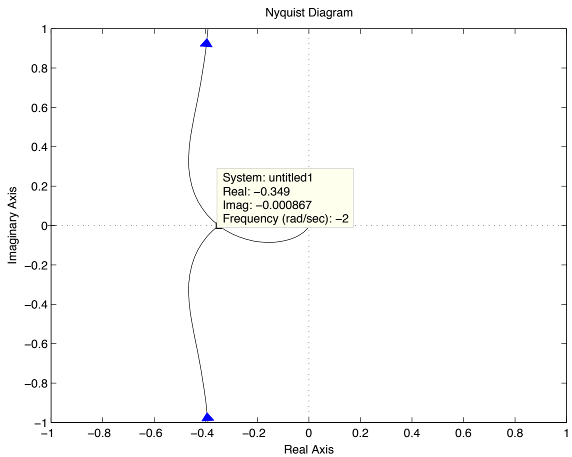
I exemplet ovan hade vi kunnat beräkna `Gc` på motsvarande sätt som `S` beräknades, dvs `Gc=F*G/(1+F*G)`. Med denna metod får dock täljaren och nämnaren i `Gc` ett antal gemensamma faktorer som kan förkortas bort. Genom att använda funktionen `feedback` undviks detta. De gemensamma faktorerna i det första alternativet kan elimineras genom att använda funktionen `minreal(Gc)` Testa själv och jämför.

5 Nyquistdiagram

Nyquistkurvor för en eller flera överföringsfunktioner ritas med funktionen `nyquist`. Eftersom funktionen `nyquist` graderar axlarna automatiskt kan diagrammet ibland bli svårläst. Läsbarheten kan förbättras genom att man själv väljer axlarnas gradering med funktionen `axis`. Man kan få ut mycket information ur figuren genom att använda vänster respektive höger musknapp. Med vänster musknapp kan man t ex markera en punkt på kurvan och få ut motsvarande värde på ω samt nyquistkurvans värde i denna frekvens. Med höger musknapp får man en meny med olika operationer som kan göras med figuren.

Rita nyquistkurvan för det öppna systemet då systemet $G(s)$ styrs med en proportionell återkoppling med förstärkning $K_P = 0.7$. Justera axlarnas gradering och markera punkten där nyquistkurvan passerar negativa delen av reella axeln.

```
>> nyquist( F * G )
>> axis([ -1 1 -1 1 ])
```



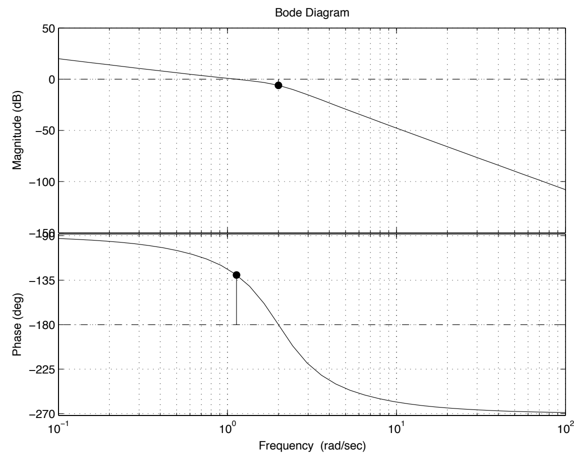
---

6 Bodediagram

Bodediagram för en eller flera överföringsfunktioner ritas med funktionen `bode`. Även i detta fall kan man läsa av punkter i figuren genom att markera med vänster musknapp. Med höger knapp får man en meny där man t ex kan välja att markera frekvenserna där stabilitetsmarginalerna läses av.

Beräkna frekvensfunktionen för systemet `G` och rita upp den i ett bodediagram. Notera att amplitudkurvan graderas i decibel. Använd höger musknapp och lägg in rutnät i figuren samt markera var fas- och amplitudskärfrekvenserna ligger.

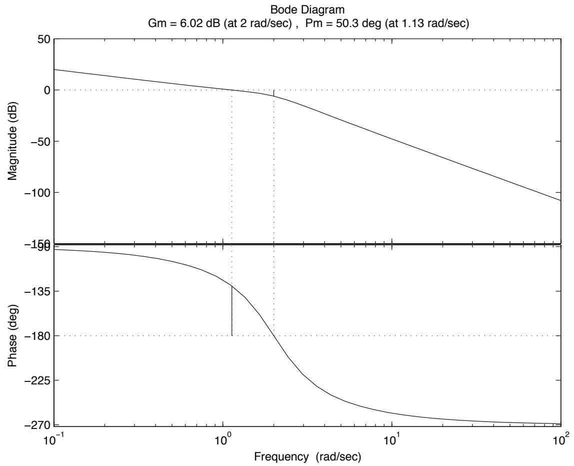
```
>> bode( G )
```



För att bestämma skärfrekvenser samt fas- och amplitudmarginal kan man även använda funktionen `margin`, vilken förutom att rita upp amplitud- och faskurvorna även skriver ut dessa värden. `Gm` och `Pm` betecknar amplitud- respektive fasmarginal.

Beräkna frekvensfunktionen för systemet `G` och rita upp den i ett bodediagram.

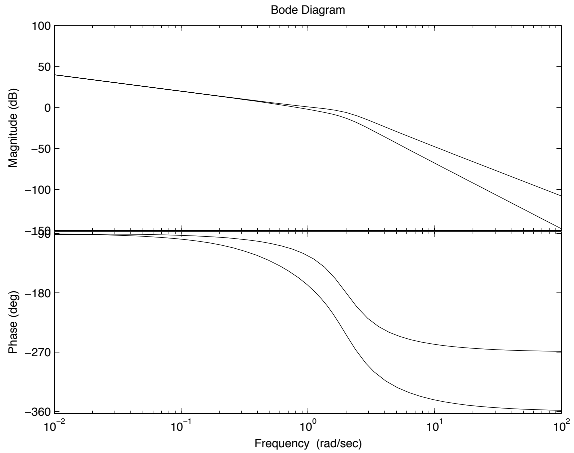
```
>> margin( G )
```



För att t ex kunna göra jämförelser mellan två frekvensfunktioner kan dessa ritas i samma diagram.

Beräkna frekvensfunktionerna för systemen `G` och `G2`.

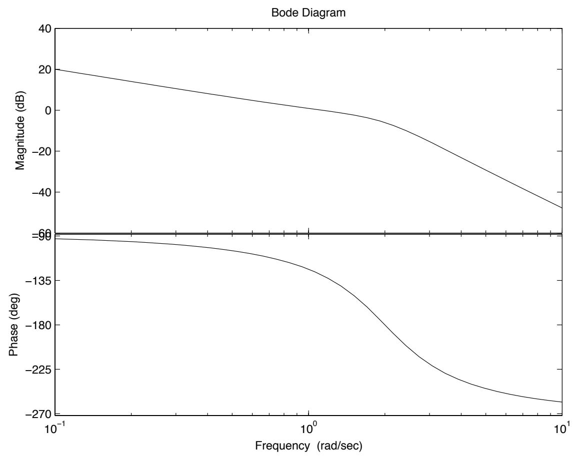
```
>> bode( G, G2 )
```



Skalan på frekvensaxeln kan väljas genom att som sista argument i funktionsanropet ange största och minsta frekvensvärdet mellan krullparenteser .

Beräkna frekvensfunktionen för systemet `G` från 0.1 till 10 rad/s och rita upp den i ett bodediagram.

```
>> bode( G, { 0.1, 10 } )
```



7 Simulering

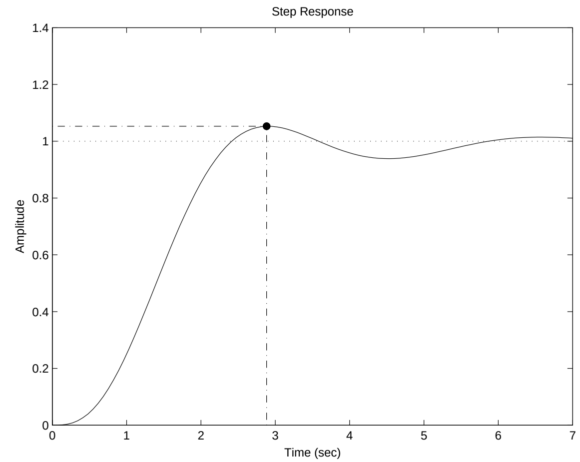
7.1 Stegsvar

Den vanligaste typen av simulering är att beräkna ett systems stegsvar. Detta kan utföras med funktionen `step`, med vilken man både simulerar systemet och ritar dess stegsvar. I likhet med tidigare kan man läsa av enskilda värden i figuren med vänster musknapp och få en meny med olika val med höger knapp. Genom att t ex välja `Peak Response` från `Characteristics` markeras tidpunkt och värde för överslängen. Placera markören över punkten i diagrammet visas tillhörande numeriska värden.

Antag att systemet `G` styrs med proportionell återkoppling med förstärkning $K_p = 0.7$.

Beräkna och rita upp det återkopplade systemets stegsvar. Markera stegsvarets översläng.

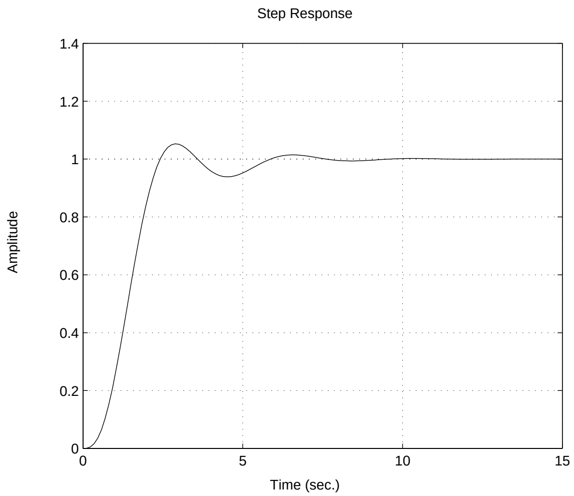
```
>> step( Gc )
```



I normalfallet väljs simuleringstiden automatiskt, men genom att ange ett extra argument kan man välja simuleringstiden själv.

Beräkna det återkopplade systemets stegsvar under femton sekunder och rita upp resultatet.

```
>> step( Gc, 15 )
```



7.2 Allmän insignal

För att simulera linjära system med allmänna insignaler kan man använda funktionen `lsim(G,u,t)`. Indata till denna funktion är ett (eller flera) system `G`, en insignalvektor `u` och en tidsvektor `t`.

Antag exempelvis att vi vill studera reglerfelet för det återkopplade systemet ovan då referenssignalen är en ramp. Vi vet att sambandet mellan referenssignal och reglerfel ges av känslighetsfunktionen

$$E(s) = S(s)R(s)$$

där

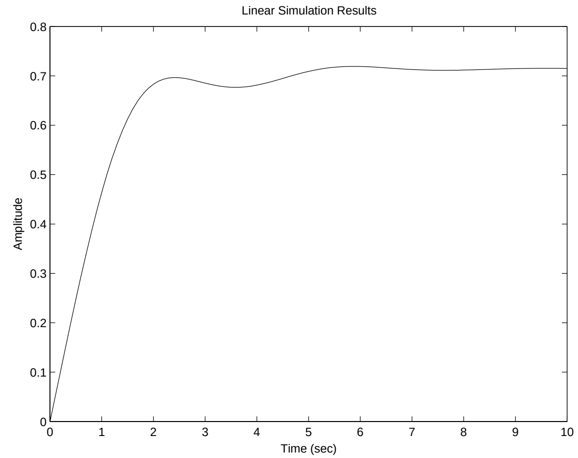
$$S(s) = \frac{1}{1 + F(s)G(s)}$$

Skapa en tidsvektor mellan 0 och 10 med steget 0.1.

```
>> t = ( 0 : 0.1 : 10 ).';
```

Simulera det återkopplade systemet då referenssignalen är en ramp med lutning 0.5. Reglerfelet går i detta fall mot 0.71. Funktionen ritar även insignalen, men den kan välja bort på menyn som nås via höger musknapp.

```
>> lsim( S, 0.5*t, t )
```



För att skapa sinus- och fyrkantsignaler kan funktionen `gensig` användas.

8 Rotort

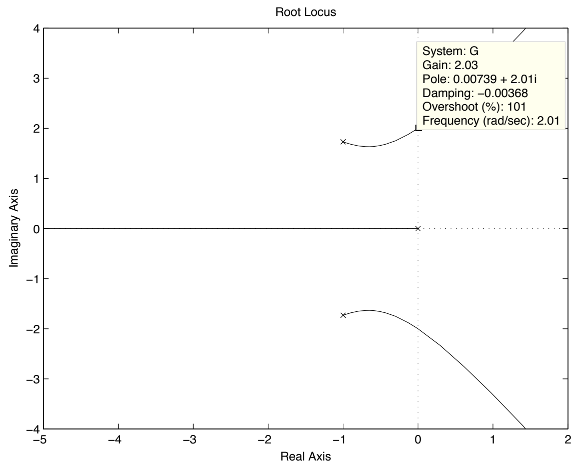
För att avgöra hur rötterna till ekvationen

$$P(s) + KQ(s) = 0$$

rör sig i komplexa talplanet då K går från noll och mot oändligheten kan man rita ekvationens rotort med funktionen `rlocus`. Indata till funktionen är en överföringsfunktion med polynomet $Q(s)$ som täljare och polynomet $P(s)$ som nämnare. Med höger musknapp kan man markera relevanta punkter i figuren, såsom t ex då rotorten passerar imaginäraxeln.

Rita upp rotorten för det återkopplade systemets karakteristiska funktion då systemet `G` styrs med en proportionell återkoppling. Markera där en av rötterna passerar imaginäraxeln.

```
>> rlocus( G )
```



För att t ex kontrollera för vilken förstärkning polerna har viss dämpning kan man med höger musknapp lägga in ett nät vilket markerar polplaceringar med samma avstånd till origo respektive samma dämpning.

9 SISO Design Tool

Ett ytterligare användbart verktyg är *SISO Design Tool*, vilket är ett användargränssnitt med vilket man enkelt kan studera ett system ur olika aspekter såsom stegsvar, bodediagram, poler och nollställen, etc. Verktyget *SISO Design Tool* startas genom att skriva `sisotool`. Automatiskt kommer de skapade LTI-objekten att finnas tillgängliga för analys. I figuren nedan visas ett exempel på vilka figurer som kan visas samtidigt. Testa dig fram!

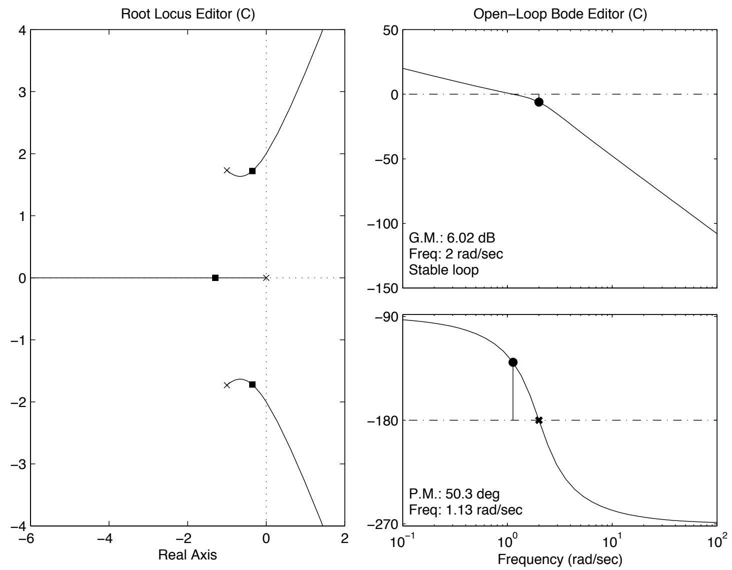


Figure 1. SISO Design Tool.

10 Tillståndsbeskrivning

I Control System Toolbox finns även en datastruktur för att hantera system på tillståndsform

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t)$$

För att skapa ett system på denna form används funktionen `ss`, med vilken man kan skapa ett system på tillståndsform från början eller konvertera ett system från överföringsfunktionsform.

Överför systemet `G` till tillståndsform.

```
>> G = ss( G )

a =
        x1  x2  x3
   x1   -2  -2   0
   x2    2   0   0
   x3    0   1   0

b =
        u1
   x1    2
   x2    0
   x3    0

c =
        x1  x2  x3
   y1    0   0   1

d =
        u1
   y1    0

Continuous-time model.
```

Matriserna A, B, C och D i tillståndsbeskrivningen ingår nu i datastrukturen `G`. För att komma åt matriserna kan man referera till dem direkt genom att skriva `G.a`, `G.b` etc.

Beräkna egenvärdena till matrisen A i tillståndsmodellen

```
>> eig( G.a )

ans =

         0
  -1.0000 + 1.7321i
  -1.0000 - 1.7321i
```

Denna möjlighet är användbar t ex när man skall beräkna polplacerande tillståndsåterkoppling på formen

$$u(t) = -Lx(t) + r(t)$$

vilket kan göras med funktionen `place` (och i special-fall med `acker`).

Bestäm en tillståndsåterkoppling som placerar det återkopplade systemets poler i närheten av -2. Lägger man alla polerna exakt i -2 kan återkopplingen inte beräknas med hjälp av `place`, så polerna sprids ut en aning. För att få alla polerna exakt i -2 kan `acker` användas, men den funktionen har så dåliga numeriska egenskaper att den alltid bör undvikas till förmån för `place`.

```
>> L = place( G.a, G.b, ...
              -2 * ( 1 + [ -0.01 0 0.01 ] ) )

L =

    2.0000    1.9999    1.9998
```

Det återkopplade systemet

$$\dot{x}(t) = (A - BL)x(t) + Br(t)$$

$$y(t) = Cx(t)$$

kan nu skapas t ex med funktionen `ss`.

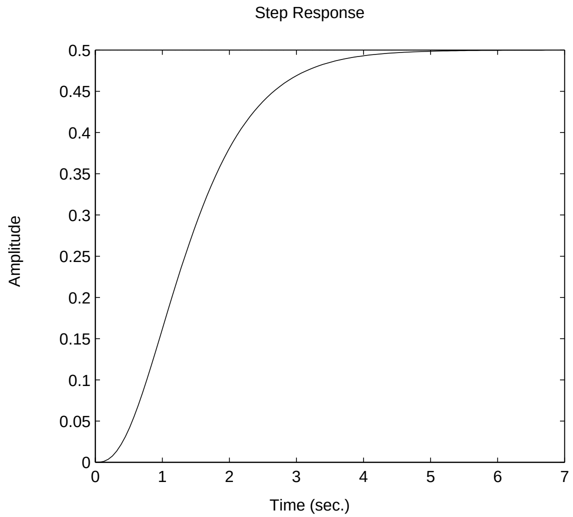
Generera tillståndsbeskrivningen för det återkopplade systemet. Kontrollera att polerna placerats på önskat sätt.

```
>> Gc = ss( G.a - G.b * L, G.b, G.c, 0 );
>> eig( Gc.a )

ans =

   -2.0200
   -2.0000
   -1.9800
```

Det återkopplade systemets stegsvar kan nu beräknas och ritas upp med funktionen `step`.

Beräkna och rita upp det återkopplade systemets stegsvar.

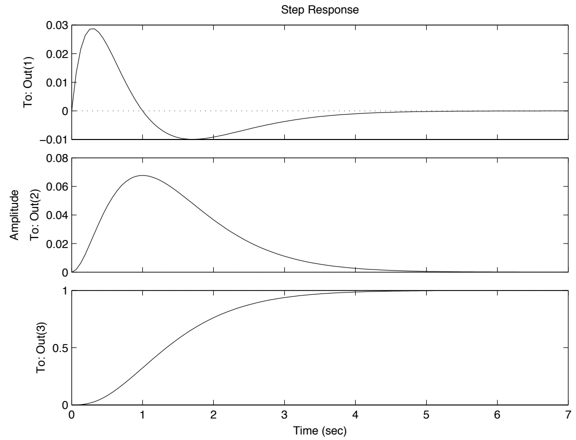
```
>> step( Gc )
```



På detta sätt ser vi endast den utsignal som definieras av vektorn C. Vill vi studera samtliga tillstånd kan detta göras genom att låta C vara en enhetsmatris med dimension lika med systemets ordningstal.

Skapa det återkopplade systemet på nytt, men med samtliga tre tillstånd som utsignaler.

```
>> Gc = ss( G.a - G.b * L, G.b, eye(3), 0 );
>> step( Gc )
```



För att beräkna linjärkvadratisk tillståndsåterkoppling kan funktionen `lqr` användas.

11 Sammanfattning av kommandon

11.1 Användbara kommandon i Control System Toolbox

`tf`	System på överföringsfunktionsform
`ss`	System på tillståndsform
`pole`	Poler
`step`	Stegsvar
`tzero`	Nollställen
`feedback`	Återkoppling
`nyquist`	Nyquistdiagram
`bode`	Bodediagram
`bodemag`	Bodediagrammets amplitudkurva
`sigma`	Generalisering av bodemag
`margin`	Bodediagram och stabilitetsmarginaler
`rlocus`	Rotort
`lsim`	Simulering med godtycklig insignal
`place`	Polplacerande tillståndsåterkoppling
`lqr`	Linjärkvadratisk tillståndsåterkoppling
`ctrb`	Styrbarhetsmatris
`obsv`	Observerbarhetsmatris
`ltiview`	Startar LTI Viewer
`pzmap`	Pol-nollställediagram
`minreal`	Förkortning av gemensamma faktorer
`sisotool`	Grafiskt gränssnitt

11.2 Användbara MATLAB-kommandon

`abs`	Absolutbelopp
`eig`	Egenvärden
`conv`	Polynommultiplikation
`det`	Determinant
`diag`	Diagonalmatris
`imag`	Imaginärdel
`inv`	Matrisinvers
`real`	Realdel
`roots`	Rötter till polynom
`grid`	Nät i figurer
`hold`	Frysning av figur
`loglog`	Diagram i log-log skala
`plot`	Diagram i linjär skala
`cd`	Byte av bibliotek
`dir`	Listning av bibliotek
`clear`	Radering av variabler och funktioner i arbetsminnet
`load`	Inläsning av variabler från fil
`save`	Lagring av variabler på fil
`who`	Listning av variabler i arbetsminnet
`helpdesk`	Startar HTML-baserad hjälpfunktion