ISSN 0280–5316
ISRN LUTFD2/TFRT--5549--SE

Adaptiv reglering av system med glapp

Mattias Grundelius

Institutionen för Reglerteknik
Lunds Tekniska Högskola
Januari 1996

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
January 1996

Document Number
ISRN LUTFD2/TFRT--5549--SE

Author(s)
Mattias Grundelius

Supervisor
Karl Johan Åström

Sponsoring organisation

Title and subtitle

Adaptiv reglering av system med glapp (Adaptive control of systems with backlash)

Abstract

Backlash is common in mechanical and hydraulic systems. It severly limits the overall system performance. Many mechanical solutions have been developed to overcome backlash, for example by using spring-loaded split gear assemblies or dual motor systems. These mechanical solutions can satisfactorily handle the backlash problem but they give rise to other problems like decreased accuracy and reduced control speed. They are also more expensive, they consume more energy and they increase the weight of the system. Therefore it would be very nice to find ways to do backlash compensation that is not based on mechanical devices. In this thesis it is attempted to approach the problem by adaptive control. Two different types of backlash compensations are examined; one linear, phase lead compensation, and one nonlinear, backlash inverse. The nonlinear backlash compensation gives better performance and is used in the controller. The parameter values in the backlash inverse are crucial to the control performance and they need to be estimated if the backlash is unknown or varying. Two different schemes for estimating the backlash parameters has been considerd, a model reference adaptive system and a recursive least squares estimator. Both methods require that the state of the backlash (i.e. if it is in the deadzone or not) is known. The recursive least squares estimator is used in the adaptive controller because it converge faster. The recursive least square estimator is expanded so that the plant parameters can be estimated too. The adaptive controller is tested by simulation on a second order plant with known or unknown parameters and a unknown backlash acting on the input of the system. The simulations show that both the backlash and the plant parameters can be estimated if you know the state of the backlash.

Key words
Backlash, Limit Cycle, Phase Lead, Describing Function, Backlash Inverse, Model Reference Adaptive System, Recursive Least Square, Adaptive Control, Indirect Self Tuning Regulator, Object-Oriented Modeling

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280–5316

ISBN

Language
Swedish

Number of pages
62

Recipient's notes

Security classification

Adaptiv Reglering av System med Glapp

Mattias Grundelius

Institutionen för Reglerteknik
Lunds Tekniska Högskola
Lund, Januari 1996

Innehåll

1. Inledning

Glapp förekommer i så gott som alla mekaniska och hydrauliska system och är en förutsättning för att dessa skall fungera. Om t.ex. en växellåda konstruerades så att det inte fanns något glapp skulle kugghjulen låsa sig i varandra om temperaturen steg och glapp skulle uppstå om temperaturen sjönk. I mekaniska system är de olika komponenterna i kontakt med varandra och nöter således på varandra. På så sätt kan glappets storlek växa med tiden.

Om glapp förekommer i reglerkretsen försämras prestanda kraftigt och självsvängningar kan uppstå (eng. limit cycles). Ett sätt att undvika självsvängningar är att reglera systemet försiktigt och ha en liten förstärkning. Det har även lagts ned mycket konstruktionsarbete på att göra mekaniska system utan glapp huvudsakligen genom att använda fjäderförspända kugghjul och fler drivmotorer. Dessa konstruktioner har med framgång lyckats eliminera eller minska glappets inverkan, men har medfört andra begränsningar på systemet såsom försämrad noggrannhet och långsammare reglering. De har också varit dyrare att producera, förbrukat mer energi och ökat vikten hos systemet. Ett enkelt och billigt sätt vora att reglertekniskt kompensera för glappet på ett effektivt sätt.

Det här examensarbetet har gått ut på att finna reglertekniska metoder för att förbättra regleringen av system med glapp om systemets parametrar är okända och/eller tidsvariabla. För glappkompensering har två metoder undersökts; dels en linjär metod, fasavancering, dels en olinjär metod, glappinvers. Om man effektivt skall kunna kompensera glapp krävs det att man vet glappets storlek. Om glappet är okänt eller varierar i tiden behöver man skatta glappets storlek. Skattning har gjorts med både modellreferenssystem och minstakvadratskattning. För att undersöka de olika kompenserings- och skattningsmetoderna har de olika modellerna implementerats i Omola och simuleringar gjorts i OmSim.

I kapitel 2 presenteras en glappmodell och simuleringar görs för att se vilken inverkan glappet har på regleringen. För att förbättra regleringen introduceras en glappkompensering i reglerkretsen. Kapitel 3 behandlar olika metoder för att skatta glappets storlek. Skattningen görs enligt två olika principer; modellanpassning och parameterskattning. En adaptiv regulator med adaptiv glappkompensering beskrivs i kapitel 4. Kapitlet innehåller även simuleringar gjorda på kända och okända system med glapp och varierande parametrar. I kapitel 5 beskrivs de simuleringsmodeller som har använts vid simuleringarna i Omsim. Kapitel 6 innehåller slutsatser och förslag på framtida utökningar. Appendix A innehåller listningar av de implementerade Omola modellerna.

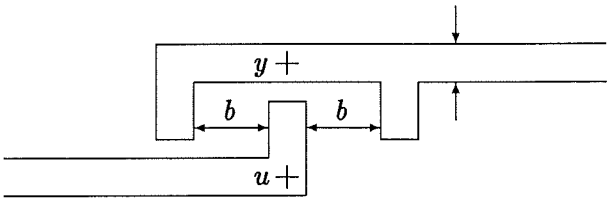
2. Egenskaper hos glapp och system med glapp

I detta kapitel behandlas de olika egenskaper som system med glapp har. Först tas en modell för glappet fram, avsnitt 2.1, och sedan studeras vilka konsekvenser glapp har i ett reglersystem, avsnitt 2.2. Slutligen undersöks två reglertekniska möjligheter att förbättra prestandan hos system med glapp; en linjär metod, fasavancering, och en olinjär metod, glappinvers, avsnitt 2.3.

2.1. Modell av glapp

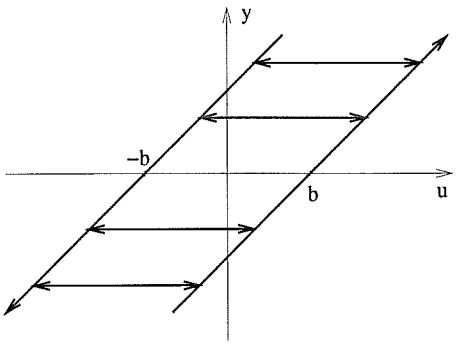
En enkel mekanisk modell av glapp visas i figur 2.1. En matematisk modell härleds utgående från figurens mekaniska egenskaper. I härledningen tas ingen hänsyn till Newtons lagar.

Om u rör sig åt höger kommer y vara oförändrad ($\dot{y} = 0$) ända tills u når den högra tappen ($u = y + b$). Om u fortsätter åt höger kommer y att ändras med samma hastighet som u ($\dot{y} = \dot{u}$) och ha värdet $u - b$. Om u ändrar riktning kommer y att ha samma värde som när u ändrade riktning ($\dot{y} = 0$) tills u når den vänstra tappen ($u = y - b$). Om u fortsätter åt vänster kommer y att ändras med samma hastighet som u ($\dot{y} = \dot{u}$) och ha värdet $u + b$.



Figur 2.1 Enkel mekanisk modell av glapp.

Matematisk kan glappet beskrivas av en figur bestående av två parallella linjer med lutningen ett och mellan dem horisontella förbindelser som används vid riktningsändringar då utsignalen är konstant, se figur 2.2. De parallella linjerna används för rörelse i positiv respektive negativ riktning.



Figur 2.2 Grafisk representation av glapp.

Eftersom y skall minnas sitt gamla värde när u befinner sig mellan tapparna måste det finnas ett tillstånd i modellen. I det kontinuerliga fallet representeras minnet av $y(t^-)$ som är värdet av y omedelbart före tiden t. I diskret tid representeras minnet av $y(t-h)$, där h är samplingstiden. Ekvationerna (2.1a), (2.1b) och (2.2) beskriver glappet i både diskret och kontinuerlig tid, där glappets storlek är tidsvariabelt.

$$y(t) = f_b(u(t), y(t^-), b(t)) \qquad \text{kontinuerlig tid} \qquad (2.1a)$$

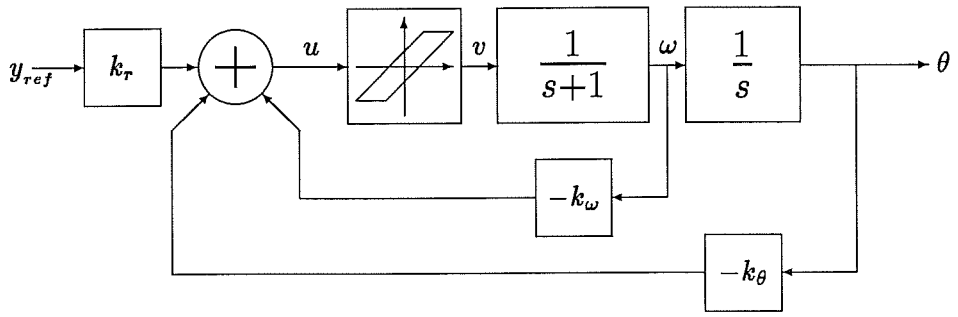
$$y(t) = f_b(u(t), y(t-h), b(t)) \qquad \text{diskret tid} \qquad (2.1b)$$

Glappfunktionen f_b ges av ekvation (2.2).

$$f_b(u,v,b) = \begin{cases} u-b & \text{om } u-v \geq b \\ u+b & \text{om } u-v \leq -b \\ v & \text{annars} \end{cases} \qquad (2.2)$$

2.2. Konsekvenser av glapp

För att visa vad som kan inträffa om det förekommer glapp i reglerkretsen kommer ett enkelt servosystem med glapp att simuleras, se figur 2.3. Undersökningen kommer att begränsas till fallet när glappet verkar på ingången till processen. I allmänhet sitter det dynamiska block både före och efter glappet.



Figur 2.3 Blockschema för servosystemet.

Servot regleras med tillståndsåterkoppling av positionen θ och rotationshastigheten ω. Koefficienterna i tillståndsåterkopplingen beräknas så att det slutna systemet får skärfrekvensen $\omega_c = 2$ rad/s, dämpningen $\zeta = 0.9$ och den statiska förstärkningen ett.

Servot beskrivs på tillståndsform av ekvation (2.3) och styrsignalen ges av ekvation (2.4).

$$\dot{x}(t) = \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} v(t)\,, \qquad x = \begin{bmatrix} \omega \\ \theta \end{bmatrix} \qquad (2.3)$$

$$u(t) = k_r y_{ref}(t) - \begin{bmatrix} k_\omega & k_\theta \end{bmatrix} x(t) \qquad (2.4)$$

Om glappets inverkan försummas ger ekvation (2.4) och ekvation (2.3) överföringsfunktionen från y_{ref} till θ i ekvation (2.5).

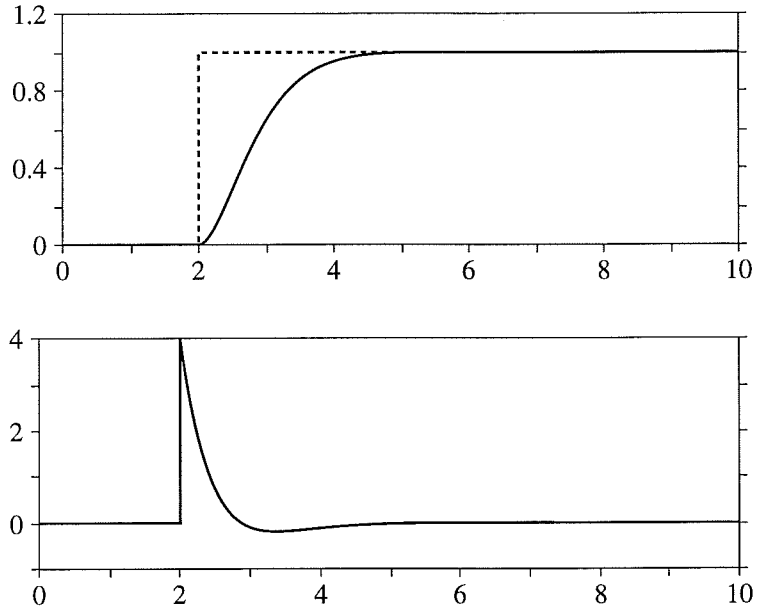
$$\Theta(s) = \frac{k_r}{s^2 + (1 + k_\omega)s + k_\theta} Y_{ref}(s) \tag{2.5}$$

Det önskade slutna systemets överföringsfunktion från y_{ref} till θ ges av ekvation (2.6).

$$\Theta(s) = \frac{\omega_c^2}{s^2 + 2\zeta\omega_c s + \omega_c^2} Y_{ref}(s) = \frac{4}{s^2 + 3.6s + 4} Y_{ref}(s) \tag{2.6}$$

Identifiering av parametrar i ekvation (2.5) och ekvation (2.6) ger

$$k_\omega = 2.6 \qquad k_\theta = 4 \qquad k_r = 4$$



Figur 2.4 Stegsvar för servosystemet utan glapp. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen.

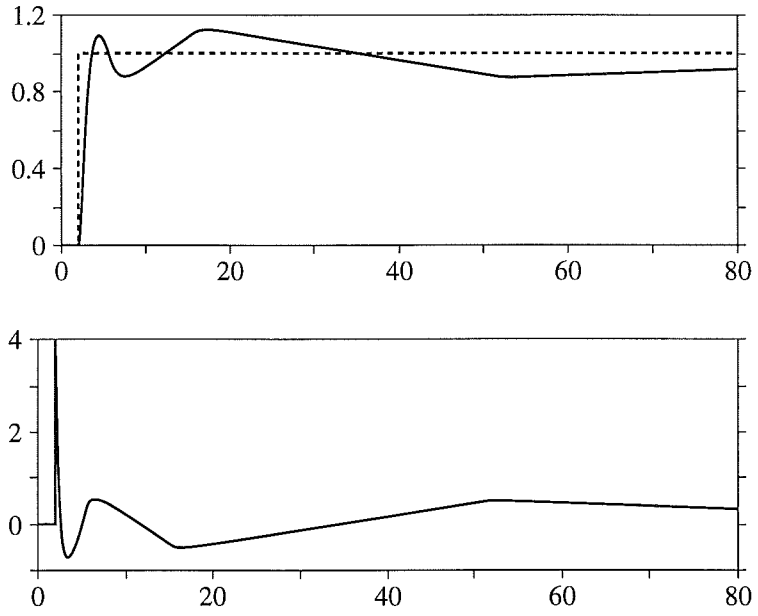
Figur 2.4 visar stegsvaret för servosystemet utan glapp. Om glappet införs i reglerkretsen försämras regleringen avsevärt, se figur 2.5. Detta beror på att glappet försummades när regulatordesignen gjordes. Glappet har storleken 0.5.

Detta visar att glapp inte kan försummas i regulatordesignen och att det är önskvärt att finna metoder för glappkompensering.

2.3. Glappkompensering

I det här avsnittet kommer två olika reglertekniska metoder för glappkompensering att presenteras, en linjär och en olinjär metod.

För att inte begränsa sig till någon speciell reglerstratergi kommer glappkompenseringen att vara separerad från regulatorn.



Figur 2.5 Stegsvar för servosystemet med glapp. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen.

Linjär kompensering

Ett enkelt sätt att kompensera för glapp är fasavancering. Överföringsfunktionen för den fasavancerande länken ges av

$$G(s) = N\frac{s+b}{s+bN}$$

För att bestämma parametrarna i fasavanceringen används begreppet beskrivande funktion (eng. describing function), se [1] och [3]. En beskrivande funktion för glapp ges av ekvation (2.7).

$$\begin{aligned} \mathrm{Re}\{N(C)\} &= \frac{1}{\pi}\left(\frac{\pi}{2} + \arcsin(1-2\frac{D}{C}) + 2(1-2\frac{D}{C})\sqrt{\frac{D}{C}(1-\frac{D}{C})}\right) \\ \mathrm{Im}\{N(C)\} &= -\frac{4D}{\pi C}(1-\frac{D}{C}) \qquad C \geq D \end{aligned} \tag{2.7}$$

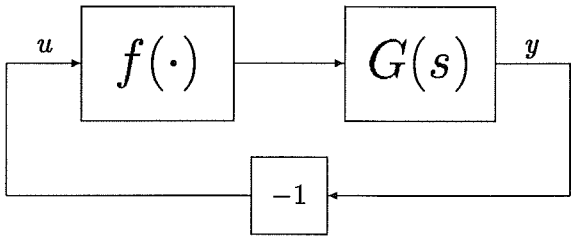
I ekvation (2.7) är D glappets storlek och C är insignalens amplitud. Den beskrivande funktionen $N(C)$ kan ses som glappets överföringsfunktionen för en ren sinussignal med avseende på amplituden och oberoende av frekvensen.

Systemet delas upp i en linjär del och en olinjär del, se figur 2.6. I fallet med servosystemet ges den linjära delen av regulatorn, fasavanceringen och processdynamiken, se ekvation (2.8). Den olinjära delen ges av glappet som beskrivs av den beskrivande funktionen i ekvation (2.7).

$$G(s) = (sk_\omega + k_\theta)N\frac{s+b}{s+bN}\frac{1}{s(s+1)} \tag{2.8}$$

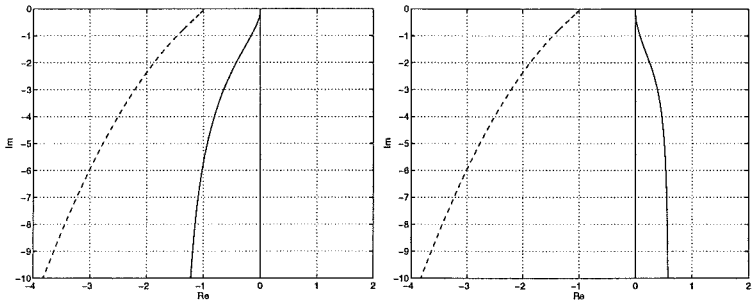
Om alla övertoner försummas dvs. $y = C\sin\omega t$ ges det slutna systemet av

$$\frac{G(i\omega)N(C)}{1+G(i\omega)N(C)}$$



Figur 2.6 System uppdelat i en olinjär och en linjär del.

Om $1 + G(i\omega)N(C) = 0$ kommer systemet att oscillera. För högre ordningens system är det omständigt att analytiskt lösa denna ekvationen men genom att rita Nyquistdiagrammet för $G(i\omega)$ och $-1/N(C)$ eller för $1/G(i\omega)$ och $-N(C)$. I diagrammet ges lösningen av skärningen mellan kurvorna. Figur 2.7 visar Nyquistdiagrammet för servosystemet med och utan fasavancering. Fasavanceringen har parametrarna $b = 1$ och $N = 2$.



Figur 2.7 Nyquist diagram för $G(i\omega)$ (heldragen) och $-1/N(C)$ (streckad) utan fasavancering (vänster) och med fasavancering (höger).

I figur 2.7 finns det ingen skärning mellan kurvorna men empiriska undersökningar har visat att om $G(i\omega)$ är positivt reell, dvs. $\mathrm{Re}\{G(i\omega)\} \geq 0$ för alla ω, uppstår inga oscillationer. Realdelen av $G(i\omega)$ ges av utrycket nedan.

$$\mathrm{Re}\{G(i\omega)\} = \frac{\omega^2(k_\omega(1 + b(N-1)) - k_\theta) + k_\omega N b^2 - k_\theta b(1 + N(b-1))}{\omega^4 + \omega^2(1 + N^2b^2) + N^2b^2}$$

$\mathrm{Re}\{G(i\omega)\} \geq 0$ ger olikheterna (2.9a) och (2.9b).

$$k_\omega(1 + b(N-1)) - k_\theta \geq 0 \tag{2.9a}$$
$$k_\omega N b - k_\theta(1 + N(b-1)) \geq 0 \tag{2.9b}$$

Addition av olikheterna (2.9a) och (2.9b) ger olikheten (2.10).

$$(2k_\omega - k_\theta)Nb - k_\omega b + k_\theta N + k_\omega - 2k_\theta \geq 0 \tag{2.10}$$

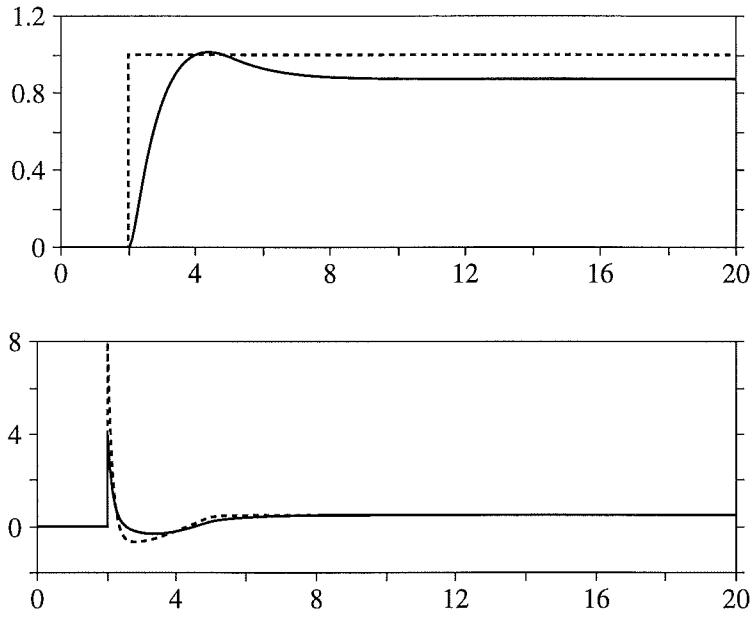
Om k_ω och k_θ är bestämda ger olikheten (2.10) ett sambandet mellan b och N.

$$N \geq \frac{k_\omega(b-1) + 2k_\theta}{k_\theta(b-1) + 2k_\omega}$$

Om $b = 1$ och $N = 1$ är systemet ekvivalent med servosystemet utan fasavancering och olikheten (2.10) kan skrivas som

$$k_\omega - k_\theta \geq 0$$

Detta visar att regulatordesignen kan göras så att ingen oscillation uppstår.



Figur 2.8 Stegsvar för servosystemet med glapp och linjär kompensering. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen före fasavanceringen (streckad) och efter fasavanceringen (heldragen).

I figur 2.8 visas en simulering av servosystemet med fasavancering där $b = 1$ och $N = 2$. Oscillationen har försvunnit men ett stationärt fel kvarstår. Om systemet varit linjärt skulle detta gå att åtgärda genom att införa integralverkan i regulatorn. Införandet av en integrator i regulatorn kommer att försämra fasen och motverka fasavanceringen, därför kommer oscillationerna att bli kraftigare än utan fasavancering.

Olinjär kompensering

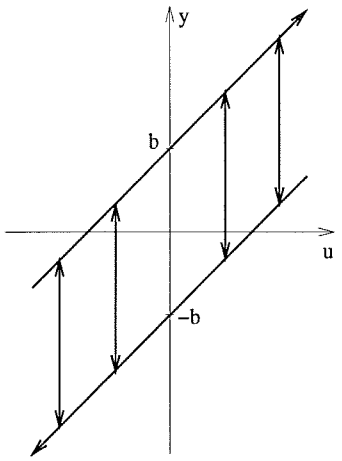
Ett annat sätt att kompensera för en olinjäritet är att införa en invers till olinjäriteten i reglerkretsen. Om det finns en funktion som är invers till glappet kan glappets inverkan elimineras totalt.

Inversen till en funktion fås genom att spegla funktionens graf i linjen $y = x$. Spegling av grafen i figur 2.2 ger grafen som visas i figur 2.9.

Glappinversen som visas i figur 2.9 beskrivs matematiskt av ekvationerna (2.11a), (2.11b) och (2.12) i både kontinuerlig och diskret tid.

$$y(t) = f_b^{-1}(u(t), u(t^-), y(t^-), b(t)) \qquad \text{kontinuerlig tid} \qquad (2.11a)$$

$$y(t) = f_b^{-1}(u(t), u(t-h), y(t-h), b(t)) \qquad \text{diskret tid} \qquad (2.11b)$$

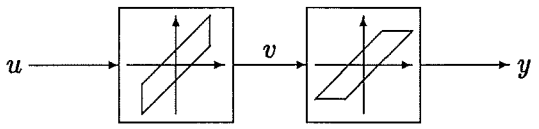


Figur 2.9 Grafisk representation av glappinvers.

Glappinversfunktionen f_b^{-1} ges av ekvation (2.12).

$$f_b^{-1}(u,v,w,b) = \begin{cases} u+b & \text{om } u-v>0 \\ u-b & \text{om } u-v<0 \\ w & \text{annars} \end{cases} \tag{2.12}$$

För att få klarhet i vad som händer om en glappinvers införs i reglerkretsen kommer systemet som visas i figur 2.10 att undersökas. I glappet är $b(t) = b_g$ och i glappinversen är $b(t) = b_i$.



Figur 2.10 Glappinvers kopplat i serie med glapp.

Systemet som visas i figur 2.10 beskrivs matematiskt av följande uttryck.

$$\begin{aligned} y(t) &= f_b(v(t), y(t^-), b_g) \\ v(t) &= f_b^{-1}(u(t), u(t^-), v(t^-), b_i) \\ y(t) &= f_b(f_b^{-1}(u(t), u(t^-), v(t^-), b_i), y(t^-), b_g) \end{aligned}$$

Antag att $y(0) = 0$ och att $u(t)$ och $v(t)$ ges av

t	$u(t)$	$v(t)$	$\dot{u}(t) = \dot{v}(t)$
$0 \to t_1$	t	$t + b_i$	1
$t_1 \to t_2$	t_1	$t_1 + b_i$	0
$t_2 \to t_3$	$t_1 + t_2 - t$	$t_1 + t_2 - t - b_i$	-1
$t_3 \to \infty$	$t_1 + t_2 - t_3$	$t_1 + t_2 - t_3 - b_i$	0

där $t_1 > |b_g - b_i|$ och $t_3 - t_2 > |b_g - b_i|$.

Undersökningen delas in i tre olika fall beroende på om b_g är större än, mindre än eller lika med b_i. Differensen mellan glappets storlek och glappinversens storlek defineras som $b_d = b_g - b_i$.

Om glappet är större än glappinversen dvs. $b_g > b_i$ blir $y(t)$ enligt nedan.

$$
\begin{array}{lll}
0 \le t < b_d & : y(t) = 0 & \text{ty } |v(t) - y(t^-)| = t + b_i < b_g \\
b_d \le t < t_1 & : y(t) = u(t) - b_d & \text{ty } v(t) - y(t^-) = b_i + b_d \ge b_g \\
t_1 \le t < t_2 & : y(t) = u(t) - b_d & \text{ty } v(t) - y(t^-) = b_i + b_d \ge b_g \\
t_2 \le t < t_2 + 2b_d & : y(t) = t_1 - b_d & \text{ty } |v(t) - y(t^-)| = |t_2 - t + b_g| < \\
 & & < |2b_i - b_g| < b_g \\
t_2 + 2b_d \le t < t_3 & : y(t) = u(t) + b_d & \text{ty } v(t) - y(t^-) = -b_i - b_d \le -b_g \\
t_3 \le t < \infty & : y(t) = t_1 + t_2 - t_3 + b_d & \text{ty } v(t) - y(t^-) = -b_i - b_d \le -b_g
\end{array}
$$

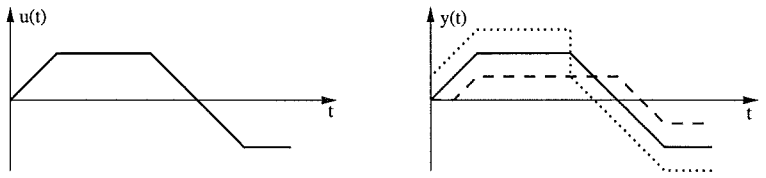
Detta är ekvivalent med $y(t) = f_b(u(t), y(t^-), b_d)$ dvs. glappinversen har tillsammans med glappet samma inverkan som ett glapp med storleken b_d.

Om glappet är mindre än glappinversen dvs. $b_g < b_i$ blir $y(t)$ enligt nedan.

$$
\begin{array}{lll}
0 \le t < t_1 & : y(t) = u(t) - b_d & \text{ty } v(t) - y(t^-) = b_i + b_d \ge b_g \\
t_1 \le t < t_2 & : y(t) = u(t) - b_d & \text{ty } v(t) - y(t^-) = b_i + b_d \ge b_g \\
t_2 \le t < t_3 & : y(t) = u(t) + b_d & \text{ty } v(t) - y(t^-) = -b_i - b_d \le -b_g \\
t_3 \le t < \infty & : y(t) = u(t) + b_d & \text{ty } v(t) - y(t^-) = -b_i - b_d \le -b_g
\end{array}
$$

Detta är ekvivalent med $y(t) = f_b^{-1}(u(t), u(t^-), y(t^-), -b_d)$ dvs. glappinversen har tillsammans med glappet samma inverkan som en glappinvers med storleken $-b_d$.

Om glappet är lika med glappinversen dvs. $b_g = b_i$ blir $y(t) = u(t)$ alltså har glappets inverkan eliminerats. Detta fås genom att sätta in $b_d = 0$ i endera av ovanstående fall. I figur 2.11 åskådliggörs hur $y(t)$ ser ut i de olika fallen.



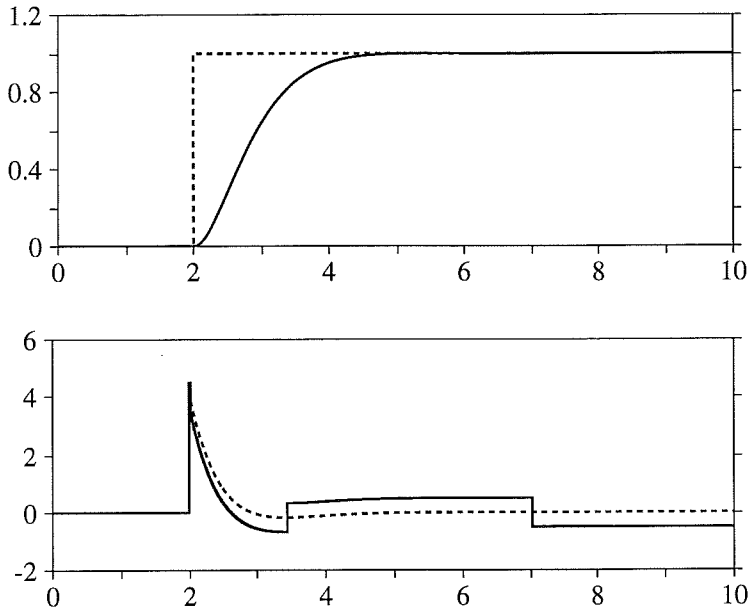
Figur 2.11 Insignal (vänster) och utsignal (höger) från systemet. Utsignalen är för fallen $b_g = b_i$ (heldragen), $b_g > b_i$ (streckad) och $b_g < b_i$ (prickad).

För att verifiera ovanstående resonemang har ett antal simuleringar gjorts på servosystemet med glapp och glappinvers. Storleken på glappet b har i alla simuleringar varit 0.5.

I figur 2.12 visas en simulering med glappinversens storlek $b = 0.5$, dvs. glappet är lika med glappinversen. Stegsvaret får samma utseende som i figur 2.4, där servosystemet simulerades utan glapp. Alltså har glappets inverkan eliminerats.

Figur 2.13 visar en simulering med glappinversens storlek $b = 0.25$, dvs. glappet är större än glappinversen. Detta ger ett stegsvar med samma karaktär som stegsvaret i figur 2.5, där servosystemet med glapp utan kompensering simulerades, fast med mindre amplitud. Detta visar på att glappets storlek har reducerats.

Avslutningsvis visas i figur 2.14 en simulering med glappinversens storlek $b = 0.75$, dvs. glappet är mindre än glappinversen. Stegsvaret ser likadant



Figur 2.12 Stegsvar för servosystemet med glapp och olinjär kompensering när glapp och glappinvers är lika stora. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen före glappinversen (streckad) och efter glappinversen (heldragen).

ut som när glappinversens storlek är lika med glappet. Styrsignalen oscillerar dock kraftigt när reglerfelet är noll.

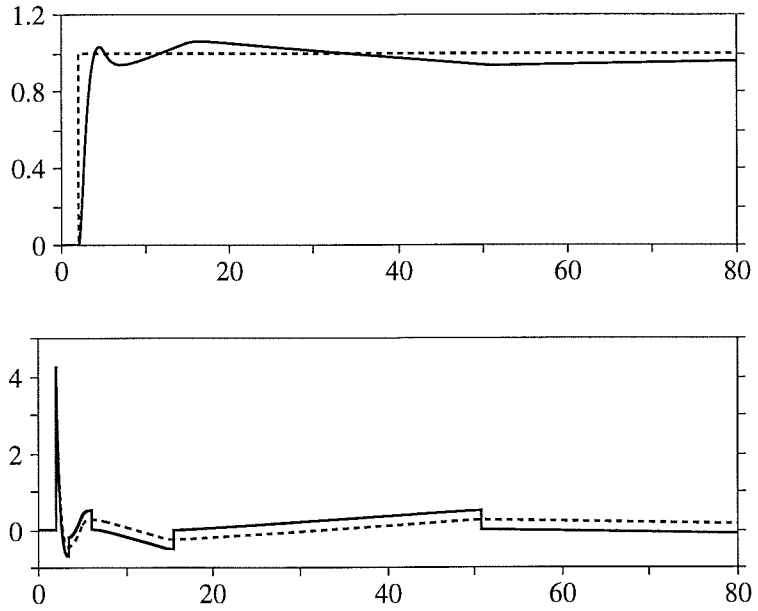
Fasavancering eller glappinvers?

Slutsatsen man kan dra av simuleringarna med glappinvers och fasavancering är att glappinvers ger bättre kompensering än fasavanceringen.

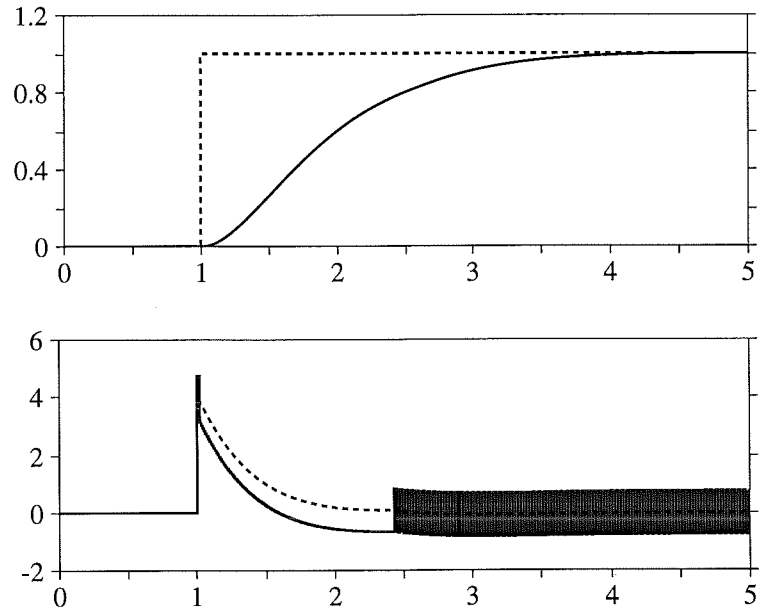
En nackdel med glappinversen är att om glappinversen är för liten så minskas bara glappets storlek och oscillationerna kvarstår. Om å andra sidan glappinversen är för stor kommer det att bli oscillationer i styrsignalen. Därför går det inte att använda en konstant glappinvers om glappets storlek varierar i tiden. Det är därför önskvärt att ha någon form av självinställande glappinvers.

Detta problem får man inte om man använder fasavancering för att kompensera för glapp, därför att glappets beskrivande funktion har samma graf oavsett vilken storlek glappet har. Däremot är fasavanceringen beroende av process och regulatorparametrar.

En annan fördel med fasavacering är att den ger samma kompensering oberoende av var glappet sitter i systemet. Glappinversen eliminerar bara glappet om glappet verkar direkt på ingången till systemet. Om det förekommer ett dynamiskt block mellan glappet och glappinversen måste man även använda en invers till det dynamiska blocket för att eliminera glappet totalt.



Figur 2.13 Stegsvar för servosystemet med glapp och olinjär kompensering när glappet är större än glappinversen. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen före glappinversen (streckad) och efter glappinversen (heldragen).



Figur 2.14 Stegsvar för servosystemet med glapp och olinjär kompensering när glappet är mindre än glappinversen. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen före glappinversen (streckad) och efter glappinversen (heldragen).

3. Skattning av glapp och okända system med glapp

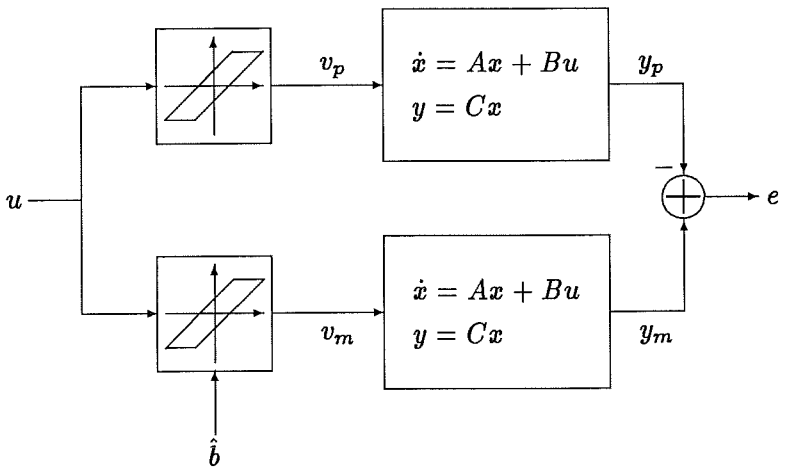
Detta kapitel behandlar olika metoder för att skatta glappets storlek. Det är nödvändigt för att kunnna använda någon metod av glappkompensering om glappets storlek är okänt eller tidsvariabelt.

För skattningen av glappets storlek om processen är känd tas två olika skattare fram; ett tidskontinuerligt modellreferenssystem baserat på stabilitetsteori, avsnitt 3.1, och en tidsdiskret minstakvadratskattare, avsnitt 3.2. För skattning av både processparametrar och glappets storlek tas en rekursiv minstakavdratskattare i diskret tid fram, avsnitt 3.2.

I avsnitt 3.3 görs en jämförelse mellan modellreferenssystemet och en rekursiv minstakvadratskattare för skattning av glapp i en känd process.

3.1. Modellanpassning

För att skatta glappets storlek om man känner processens överföringsfunktion kan man använda ett modellreferenssystem som visas i figur 3.1. Lämplig uppdateringslag för $\hat{b}$ kommer att härledas med hjälp av stabilitetsteori.



Figur 3.1 Blockschema för modellreferenssystemet.

Systemet består av den verkliga processen med okänt glapp och en observerare vars glappparameter $\hat{b}$ skall anpassas så att felet, e, blir noll. Processen beskrivs av ekvation (3.1a) och observeraren beskrivs av ekvation (3.1b).

$$\begin{aligned} \dot{x}_p &= Ax_p + Bv_p = Ax_p + B\underbrace{f_b(u(t), v_p(t^-), b(t))}_{f_p} \\ y_p &= Cx_p \end{aligned} \tag{3.1a}$$

$$\dot{x}_m = Ax_m + Bv_m = Ax_m + B\underbrace{f_b(u(t), v_m(t^-), \hat{b}(t))}_{f_m} \tag{3.1b}$$
$$y_m = Cx_m$$

Felet ges av ekvation (3.2) och felets tidsderivata ges av ekvation (3.3).

$$e = y_m - y_p = C(\underbrace{x_m - x_p}_{e_x}) \tag{3.2}$$

$$\dot{e} = \dot{y}_m - \dot{y}_p = CAe_x + CB(f_m - f_p) \tag{3.3}$$

En kandidat till Lyapunovfunktion ges i ekvation (3.4) och dess tidsderivata i ekvation (3.5) där P är en reell symmetrisk positivt definit matris.

$$V = e_x^T P e_x + \frac{1}{\gamma}(f_m - f_p)^2 \tag{3.4}$$

$$\begin{aligned} \dot{V} &= e_x^T P \dot{e}_x + \dot{e}_x^T P e_x + \frac{2}{\gamma}(f_m - f_p)(\dot{f}_m - \dot{f}_p) \\ &= e_x^T (PA + A^T P) e_x + \frac{2}{\gamma}(f_m - f_p)(\dot{f}_m - \dot{f}_p + \gamma B^T P e_x) \\ &= -e_x^T Q e_x + \frac{2}{\gamma}(f_m - f_p)(\dot{f}_m - \dot{f}_p + \gamma B^T P e_x) \end{aligned} \tag{3.5}$$

Om P väljs så att $B^T P = C$ kan $\dot{V}$ skrivas som

$$\dot{V} = -e_x^T Q e_x + \frac{2}{\gamma}(f_m - f_p)(\dot{f}_m - \dot{f}_p + \gamma e) \tag{3.6}$$

Q är en symmetrisk matris definerad av

$$PA + A^T P = -Q \tag{3.7}$$

Systemet är stabilt om V är positivt definit och $\dot{V}$ är negativt semidefinit. V är positivt definit om P är positivt definit och $\gamma > 0$. $\dot{V}$ är negativt semidefinit om Q är positivt definit och $(2/\gamma)(f_m - f_p)(\dot{f}_m - \dot{f}_p + \gamma e) \leq 0$. Om A är stabil, dvs. alla egenvärden till A har negativ realdel, finns det för varje positivt definit symmetrisk matris Q, en positivt definit symmetrisk matris P som uppfyller ekvation (3.7). Mer om Lyapunovfunktioner finns att läsa i [1] och [2].

Tidsderivatan av f_b ges av ekvation (3.8a) och (3.8b).

$$\dot{f}_b = \frac{\partial f}{\partial u}\frac{du}{dt} + \frac{\partial f}{\partial v}\frac{dv}{dt} + \frac{\partial f}{\partial b}\frac{db}{dt} \tag{3.8a}$$

$$\begin{aligned} \frac{\partial f}{\partial u} &= \begin{cases} 0 & \text{om } |u - v| < b \\ 1 & \text{annars} \end{cases} \\ \frac{\partial f}{\partial v} &= \begin{cases} 1 & \text{om } |u - v| < b \\ 0 & \text{annars} \end{cases} \\ \frac{\partial f}{\partial b} &= \begin{cases} -1 & \text{om } u - v \geq b \\ 1 & \text{om } u - v \leq -b \\ 0 & \text{annars} \end{cases} \end{aligned} \tag{3.8b}$$

Av ekvation (3.8b) kan man dra slutsatsen att $\dot{f}_b$ är olika beroende på vilket tillstånd som glappet befinner sig i, därför kommer analysen att delas in i olika fall. Glappet kan antingen driva åt höger respektive vänster eller befinna sig i dödzonen. Det är två olika glapp som har tre tillstånd vardera, detta ger nio möjliga kombinationer. Två av de nio möjliga kombinationerna kan inte inträffa eftersom glappen inte kan driva åt olika håll då de ha samma insignal och kommer därför inte att undersökas.

1. Båda glappen driver åt vänster.

$$\begin{array}{l} u - v_p \leq -b \\ u - v_m \leq -\hat{b} \end{array} \quad \Longrightarrow \quad \begin{array}{l} f_p = u + b \\ f_m = u + \hat{b} \end{array}$$

$$\begin{aligned} \dot{V} &= -e_x^T Q e_x + \frac{2}{\gamma}(u + \hat{b} - u - b) \\ &\left(\frac{\partial f_m}{\partial u}\frac{du}{dt} + \frac{\partial f_m}{\partial v}\frac{dv_m}{dt} + \frac{\partial f_m}{\partial b}\frac{d\hat{b}}{dt} - \frac{\partial f_p}{\partial u}\frac{du}{dt} - \frac{\partial f_p}{\partial v}\frac{dv_p}{dt} - \frac{\partial f_p}{\partial b}\frac{db}{dt} + \gamma e \right) \\ &= -e_x^T Q e_x + \frac{2}{\gamma}(\hat{b} - b)\left(\frac{du}{dt} + \frac{d\hat{b}}{dt} - \frac{du}{dt} - \frac{db}{dt} + \gamma e \right) \\ &= -e_x^T Q e_x + \frac{2}{\gamma}(\hat{b} - b)\left(\frac{d\hat{b}}{dt} + \gamma e \right) \quad \Longrightarrow \quad \frac{d\hat{b}}{dt} = -\gamma e \end{aligned}$$

   Genom att välja uppdateringen av $\hat{b}$ enligt ovan blir $\dot{V}$ negativt definit.

2. Det verkliga glappet driver åt vänster och observerar glappet befinner sig i dödzonen, dvs. $\hat{b} > b$.

$$\begin{array}{l} u - v_p \leq -b \\ |u - v_m| < \hat{b} \end{array} \quad \Longrightarrow \quad \begin{array}{l} f_p = u + b \\ f_m = v_m(t^-) \end{array}$$

$$\dot{V} = -e_x^T Q e_x + \frac{2}{\gamma}(v_m(t^-) - u - b)\left(-\frac{du}{dt} + \gamma e \right)$$

   Eftersom en ändring av $\hat{b}$ inte påverkar e väljs uppdateringen av $\hat{b}$ som

$$\frac{d\hat{b}}{dt} = 0$$

3. Det verkliga glappet befinner sig i dödzonen och observerarglappet driver åt vänster, dvs. $\hat{b} < b$.

$$\begin{array}{l} |u - v_p| < b \\ u - v_m \leq -\hat{b} \end{array} \quad \Longrightarrow \quad \begin{array}{l} f_p = v_p(t - \epsilon) \\ f_m = u + \hat{b} \end{array}$$

$$\dot{V} = -e_x^T Q e_x + \frac{2}{\gamma}(u + \hat{b} - v_p(t^-))\left(\frac{du}{dt} + \frac{d\hat{b}}{dt} + \gamma e \right)$$

   Efttersom $d\hat{b}/dt$ förekommer i $\dot{V}$ är det möjligt att välja uppdateringslagen för $\hat{b}$ så att $\dot{V}$ blir negativt definit. Empiriska försök har dock visat att det blir bäst resultat om uppdaterings lagen väljs som

$$\frac{d\hat{b}}{dt} = 0$$

4. Det verkliga glappet och observerar glappet befinner sig i dödzonen.

$$\begin{array}{l} |u - v_p| < b \\ |u - v_m| < \hat{b} \end{array} \quad \Longrightarrow \quad \begin{array}{l} f_p = v_p(t^-) \\ f_m = v_m(t^-) \end{array}$$

$$\dot{V} = -e_x^T Q e_x + 2(v_m(t^-) - v_p(t^-))e$$

Updateringen av $\hat{b}$ väljs på samma sätt som i fall 2.

5. Det verkliga glappet befinner sig i dödzonen och observerar glappet driver åt höger, dvs. $\hat{b} < b$.

$$\begin{array}{l} |u - v_p| < b \\ u - v_m \geq \hat{b} \end{array} \quad \Longrightarrow \quad \begin{array}{l} f_p = v_p(t^-) \\ f_m = u - \hat{b} \end{array}$$

$$\dot{V} = -e_x^T Q e_x + \frac{2}{\gamma}(u - \hat{b} - v_p(t^-)) \left(\frac{du}{dt} - \frac{d\hat{b}}{dt} + \gamma e \right)$$

Updateringen av $\hat{b}$ väljs på samma sätt som i fall 3.

6. Det verkliga glappet driver åt vänster och observerar glappet befinner sig i dödzonen, dvs. $\hat{b} > b$.

$$\begin{array}{l} u - v_p \geq b \\ |u - v_m| < \hat{b} \end{array} \quad \Longrightarrow \quad \begin{array}{l} f_p = u - b \\ f_m = v_m(t^-) \end{array}$$

$$\dot{V} = -e_x^T Q e_x + \frac{2}{\gamma}(v_m(t^-) - u + b) \left(-\frac{du}{dt} + \gamma e \right)$$

Updateringen av $\hat{b}$ väljs på samma sätt som i fall 2.

7. Båda glappen driver åt vänster.

$$\begin{array}{l} u - v_p \geq b \\ u - v_m \geq \hat{b} \end{array} \quad \Longrightarrow \quad \begin{array}{l} f_p = u - b \\ f_m = u - \hat{b} \end{array}$$

$$\dot{V} = -e_x^T Q e_x + \frac{2}{\gamma}(b - \hat{b}) \left(-\frac{d\hat{b}}{dt} + \gamma e \right) \quad \Longrightarrow \quad \frac{d\hat{b}}{dt} = \gamma e$$

Genom att välja uppdateringen av $\hat{b}$ enligt ovan blir $\dot{V}$ negativt definit.

Genom att införa indikatorfunktionerna χ_g och χ_o definerade som

$$\chi_g = \begin{cases} 1 & \text{om det verkliga glappet driver åt något håll} \\ 0 & \text{om det verkliga glappet befiner sig i dödzonen} \end{cases}$$

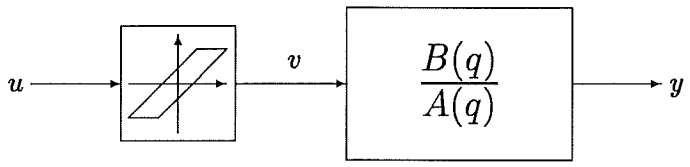
$$\chi_o = \begin{cases} 1 & \text{om observerar glappet driver åt höger} \\ -1 & \text{om observerar glappet driver åt vänster} \\ 0 & \text{om observerar glappet befiner sig i dödzonen} \end{cases}$$

kan uppdateringslagarna från de olika fallen sammanfattas som

$$\frac{d\hat{b}}{dt} = -\gamma \chi_g \chi_o e \tag{3.9}$$

3.2. Parameterskattning

Skattning av glapp och process kan även göras med en rekursiv minstakvadratskattare. Processen som skattas är i diskret tid och visas i figur 3.2 och beskrivs av ekvationerna (3.10a) och (3.10b).



Figur 3.2 Blockschema för processmodellen.

$$A(q)y(t) = B(q)v(t) \tag{3.10a}$$

$$v(t) = f_b(u(t), q^{-1}v(t), b(t)) \tag{3.10b}$$

I följande resonemang är h samplingstiden, n_a respektive n_b gradtalen för A respektive B polynomen, $b(t) = b$ är glappets storlek och är konstant.

Eftersom glappet kan befinna sig i tre olika tillstånd delas analysen upp i tre fall.

1. Glappet driver åt vänster och har drivit åt vänster i minst $n_b h$ sekunder.

   $$u(t) - q^{-1}v(t) \leq -b \quad \Longrightarrow \quad f_b(u(t), q^{-1}v(t), b(t)) = u(t) + b$$

   Insättning av f_b i ekvation (3.10b) ger tillsammans med ekvation (3.10a)

   $$A(q)y(t) = B(q)(u(t) + b) = B(q)u(t) + B(q)b = B(q)u(t) + b\sum_{i=0}^{n_b} b_i$$

2. Glappet befinner sig i dödzonen och har varit i dödzonen i minst $n_b h$ sekunder.

   $$|u(t) - q^{-1}v(t)| < b \quad \Longrightarrow \quad f_b(u(t), q^{-1}v(t), b(t)) = q^{-1}v(t)$$

   Eftersom $v(t) = q^{-1}v(t)$ kommer $v(t)$ vara konstant och ha samma värde som när glappet gick in i dödzonen. Om glappet gick in i dödzonen vid tiden t_d ger insättning av f_b i ekvation (3.10b) tillsammans med ekvation (3.10a)

   $$A(q)y(t) = B(q)v(t) = B(q)v(t_d) = B(q)(u(t_d) \pm b) = (u(t_d) \pm b)\sum_{i=0}^{n_b} b_i$$

   Där tecknet på b beror på åt vilket håll som glappet drev åt senast.

3. Glappet driver åt höger och har drivit åt höger i minst $n_b h$ sekunder.

   $$u(t) - q^{-1}v(t) \geq b \quad \Longrightarrow \quad f_b(u(t), q^{-1}v(t), b(t)) = u(t) - b$$

   Insättning av f_b i ekvation (3.10b) ger tillsammans med ekvation (3.10a)

   $$A(q)y(t) = B(q)(u(t) - b) = B(q)u(t) - B(q)b = B(q)u(t) - b\sum_{i=0}^{n_b} b_i$$

Ur ovanstående resonemang dras slutsatsen att det endast finns information för att skatta parametrar om glappet driver åt något håll. Genom att införa indikatorfunktionerna χ_k och χ_r definerande som

$$\chi_k = \begin{cases} 1 & \text{om glappet är i kontakt och} \\ & \text{har varit i kontakt i minst } n_b h \text{ sekunder} \\ 0 & \text{annars} \end{cases}$$

$$\chi_r = \begin{cases} 1 & \text{om glappet driver åt höger} \\ -1 & \text{om glappet driver åt vänster} \end{cases}$$

kan systemet sammanfattas med ekvation (3.11) om $\chi_k = 1$.

$$A(q)y(t) = B(q)u(t) - \chi_r b \sum_{i=0}^{n_b} b_i = B(q)u(t) - \chi_r C \tag{3.11}$$

Om $\chi_k = 0$ kommer ingen skattning att ske och det behövs ingen beskrivning av systemet.

Minstakvadratskattning

För skattning av parametrar kommer en vanlig rekursiv minstakvadratskattare att används. Genom att multiplicera regressionsvektorn med indikatorfunktionen χ_k kan man få uppdatering att ske endast när $\chi_k = 1$, ty

$$\chi_k = 0 \rightarrow \varphi(t) = 0 \rightarrow K(t) = 0 \rightarrow \hat{\theta}(t) = \hat{\theta}(t-1)$$

Nedan följer algoritmer för att skatta glappets storlek om processen är känd och skattning av glappets storlek och processparametrarna om processen är okänd.

Skattning av glapp om processen är känd

Om processen är känd är det bara C i ekvation (3.11) som behöver skattas. En rekursiv minstakvadratskattare för C ges av algoritm (3.12).

$$\begin{aligned} \varphi^T(t) =& \chi_k(t-1)[-\chi_r(t-1)] \\ \hat{\theta}^T =& [\hat{C}] \\ \hat{\theta}(t) =& \hat{\theta}(t-1) + K(t)\big(y(t) + \ldots + a_{n_a} y(t-n_a) - \\ & b_0 u(t-1) - \ldots - b_{n_b} u(t-n_b-1) - \varphi(t)\hat{\theta}(t-1)\big) \\ K(t) =& P(t-1)\varphi(t)/(\lambda + \varphi^T(t)P(t-1)\varphi(t)) \\ P(t) =& (I - K(t)\varphi^T(t))P(t-1)/\lambda \end{aligned} \tag{3.12}$$

Glappets storlek ges av

$$b = \hat{C}\left(\sum_{i=0}^{n_b} b_i\right)^{-1}$$

Skattning av glapp och processparametrar
Om processen är okänd måste $a_1 \dots a_{n_a}$, $b_0 \dots b_{nb}$ och C i ekvation (3.11) skattas. En rekursiv minstakvadratskattare för $a_1 \dots a_{n_a}$, $b_0 \dots b_{nb}$ och C ges av algoritm (3.13).

$$\begin{aligned}
\varphi^T(t) =& \chi_k(t-1)[-\chi_r(t-1) \quad -y(t-1)\dots -y(t-n_a) \\
& u(t-1)\dots u(t-n_b-1)] \\
\hat{\theta}^T =& [\hat{C} \quad \hat{a}_1 \dots \hat{a}_{n_a} \quad \hat{b}_0 \dots \hat{b}_{n_b}] \\
\hat{\theta}(t) =& \hat{\theta}(t-1) + K(t)\big(y(t) - \varphi^T(t)\hat{\theta}(t-1)\big) \\
K(t) =& P(t-1)\varphi(t)/(\lambda + \varphi^T(t)P(t-1)\varphi(t)) \\
P(t) =& (I - K(t)\varphi^T(t))P(t-1)/\lambda
\end{aligned} \tag{3.13}$$

Glappets storlek ges av

$$b = \hat{C}\left(\sum_{i=0}^{n_b} \hat{b}_i\right)^{-1}$$

Konvergens hos skattningen

För att undersöka om skattningen konvergerar mot det rätta värdet kommer ett första ordningens system med kända parametrar att analyseras.

Systemet kommer först att analyseras utan störningar för att härleda ett excitationsvillkor, och sedan med en konstant laststörning för att se vilken inverkan störningen har på de skattade parametrarna. Fallet när processparametrarna har fel värden i skattaren kommer också att undersökas

Algoritm (3.12) kommer att användas med glömskefaktorn $\lambda = 1$ och initialvärdet till kovarians matrisen $P(0)$ är stort.

Inga störningar
Om inga störningar förekommer beskrivs processen av nedanstående ekvation när $\chi_k = 1$

$$y(t) + a_1 y(t-1) = b_0 u(t-1) + \chi_r(t-1)C$$

där a_1 och b_0 är kända samt y, u, χ_k och χ_r är mätbara. Prediktionsfelet ges av

$$e(t) = v(t) - \varphi^T(t)\hat{\theta}$$

där

$$v(t) = y(t) + a_1 y(t-1) - b_0 u(t-1) = \varphi^T(t)\theta^0$$

Den rekursiva minstakvadratskattningen av $\hat{\theta}$ efter n steg ges av

$$\begin{aligned}
\hat{\theta}(n) &= \left(P^{-1}(0) + \sum_{t=1}^{n} \varphi(t)\varphi^T(t)\right)^{-1}\left(P^{-1}(0)\hat{\theta}(0) + \sum_{t=1}^{n}\varphi(t)v(t)\right) \\
&= \frac{P^{-1}(0)\hat{\theta}(0) + \sum \varphi(t)\varphi^T(t)\theta^0}{P^{-1}(0) + \sum \varphi(t)\varphi^T(t)} = \frac{P^{-1}(0)\hat{\theta}(0) + C\sum \chi_k^2(t-1)\chi_r^2(t-1)}{P^{-1}(0) + \sum \chi_k^2(t-1)\chi_r^2(t-1)} \\
&= \frac{P^{-1}(0)\hat{\theta}(0) + C\sum \chi_k(t-1)}{P^{-1}(0) + \sum \chi_k(t-1)} \qquad \text{ty } \chi_r^2(t) = 1\ ,\ \chi_k^2 = \chi_k \quad \forall t
\end{aligned}$$

Asymptotiskt erhålles

$$\lim_{n\to\infty}\hat{\theta}(n) = \lim_{n\to\infty}\frac{\frac{1}{n}P^{-1}(0)\hat{\theta}(0) + C\frac{1}{n}\sum\chi_k(t-1)}{\frac{1}{n}P^{-1}(0) + \frac{1}{n}\sum\chi_k(t-1)}$$

$$= \begin{cases} C & \text{om } \lim\frac{1}{n}\sum\chi_k(t-1) > 0 \\ \hat{\theta}(0) & \text{annars} \end{cases}$$

Alltså konvergerar $\hat{\theta}$ mot C om excitationsvillkoret i ekvation (3.14) är uppfyllt.

$$\lim_{n\to\infty}\frac{1}{n}\sum_{t=0}^{n}\chi_k(t) > 0 \tag{3.14}$$

Det är svårt att hitta enkla villkor på insignalen u så att villkor (3.14) blir uppfyllt. Om man antar att $u = A\sin\omega t$ så kommer villkor (3.14) att bli uppfyllt om $A > b$, där b är glappets storlek.

Med konstant laststörning
Om det förekommer en konstant laststörning i processen beskrivs processen av nedanstående ekvation när $\chi_k = 1$

$$y(t) + a_1y(t-1) = b_0(u(t-1) + d) + \chi_r(t-1)C$$

där a_1 och b_0 är kända samt y, u, χ_k och χ_r är mätbara. Prediktionsfelet ges av

$$e(t) = v(t) - \varphi^T(t)\hat{\theta}$$

där

$$v(t) = y(t) + a_1y(t-1) - b_0u(t-1) = \varphi^T(t)\theta^0 + b_0d$$

Den rekursiva minstakvadratskattningen av $\hat{\theta}$ efter n steg ges av

$$\hat{\theta}(n) = \frac{P^{-1}(0)\hat{\theta}(0) + C\sum\chi_k(t-1) + b_0d\sum\chi_k(t-1)\chi_r(t-1)}{P^{-1}(0) + \sum\chi_k(t-1)}$$

Om excitationsvillkoret i ekvation (3.14) är uppfyllt går skattningan av $\hat{\theta}$ när asymptotiskt mot

$$\lim_{n\to\infty}\hat{\theta}(n) = C + b_0d\underbrace{\lim_{n\to\infty}\frac{\sum\chi_k(t-1)\chi_r(t-1)}{\sum\chi_k(t-1)}}_{k}$$

Där k ligger mellan ett och minus ett, k antar värdet ett respektive minus ett om glappet endast driver åt höger respektive vänster. Om glappet driver lika lång tid åt höger som åt vänster antar k värdet noll. Alltså blir det ett fel i skattnigen som beror på laststörningens storlek och insignalens utseende.

Med felaktiga processparametrar i skattaren
Om parametrarna i processmodellen är felaktiga beskrivs processen av

$$y(t) + a_1 y(t-1) = b_0 u(t-1) + \chi_r(t-1)C$$

och processmodellen av

$$y(t) + a_1^* y(t-1) = b_0^* u(t-1) + \chi_r(t-1)C$$

där a_1 och b_0 är okända, a_1^* och b_0^* är kända samt y, u, χ_k och χ_r är mätbara. Prediktions felet ges av

$$e(t) = v(t) - \varphi^T(t)\hat{\theta}$$

där

$$\begin{aligned} v(t) =& y(t) + a_1^* y(t-1) - b_0^* u(t-1) \\ =& \varphi^T\theta^0 + (a_1^* - a_1)y(t-1) - (b_0^* - b_0)u(t-1) \end{aligned}$$

Den rekursiva minstakvadratskattningen av $\hat{\theta}$ efter n steg ges av

$$\begin{aligned} \hat{\theta}(n) =& \frac{P^{-1}(0)\hat{\theta}(0) + C\sum\chi_k(t-1)}{P^{-1}(0) + \sum\chi_k(t-1)} \\ &+ \frac{(a_1^* - a_1)\sum\chi_k(t-1)\chi_r(t-1)y(t-1)}{P^{-1}(0) + \sum\chi_k(t-1)} \\ &- \frac{(b_0^* - b_0)\sum\chi_k(t-1)\chi_r(t-1)u(t-1)}{P^{-1}(0) + \sum\chi_k(t-1)} \end{aligned}$$

Om excitationsvillkoret i ekvation (3.14) är uppfyllt går skattningen av $\hat{\theta}$ asymptotiskt mot

$$\begin{aligned} \lim_{n\to\infty}\hat{\theta}(n) =& C + (a_1^* - a_1)\lim_{n\to\infty}\frac{\sum\chi_k(t-1)\chi_r(t-1)y(t-1)}{\sum\chi_k(t-1)} \\ &-(b_0^* - b_0)\lim_{n\to\infty}\frac{\sum\chi_k(t-1)\chi_r(t-1)u(t-1)}{\sum\chi_k(t-1)} \end{aligned}$$

Alltså blir det ett fel i skattningen som beror på felet i parametrarna och insignalen till processen. Felet kan bli både positivt, negativt och noll beroende på insignalens utseende.

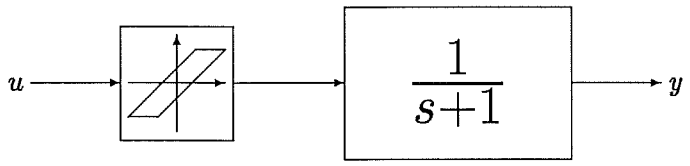
Slutsatser och praktiska aspekter

De slutsatser man kan dra av konvergensanalysen är att skattningen är känslig för laststörningar och fel i processparametrar. Det sistnämnda går att undvika genom att även skatta processparametrarna med algoritm (3.13), då även ändringar i processparametrar kan följas.

Om glappet befinner sig i dödzonen kommer regressionsvektorn φ att vara noll och kovariansmatrisen P kommer att öka exponentiellt om glömskefaktorn λ är mindre än ett. För att undvika detta bör man utforma skattaren så att ingen uppdatering av kovariansmatrisen sker när glappet befinner sig i dödzonen, dvs. när $\chi_k = 0$.

3.3. Modellanpassning eller parameterskattning?

För att jämföra de olika metoderna för skattning av glapp kommer modellreferenssystemet och minstakvadratskattaren att användas för skattning av glappet i systemet som visas i figur 3.3. Processparametrarna är kända och glappets storlek är 0.5. Insignalen som används är $u(t) = \sin 2\pi t/10$.

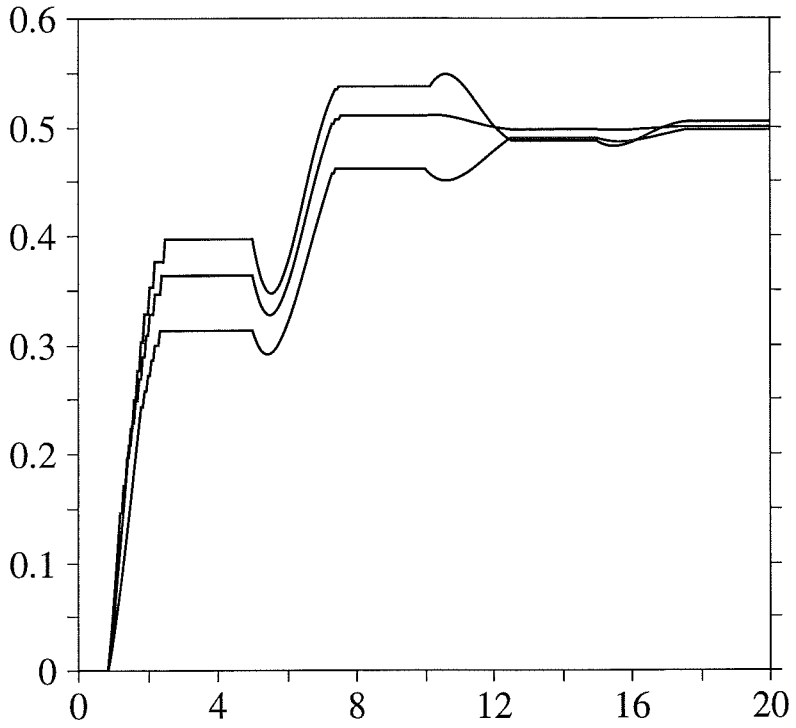


Figur 3.3 Linjärt system med glapp verkande på ingången.

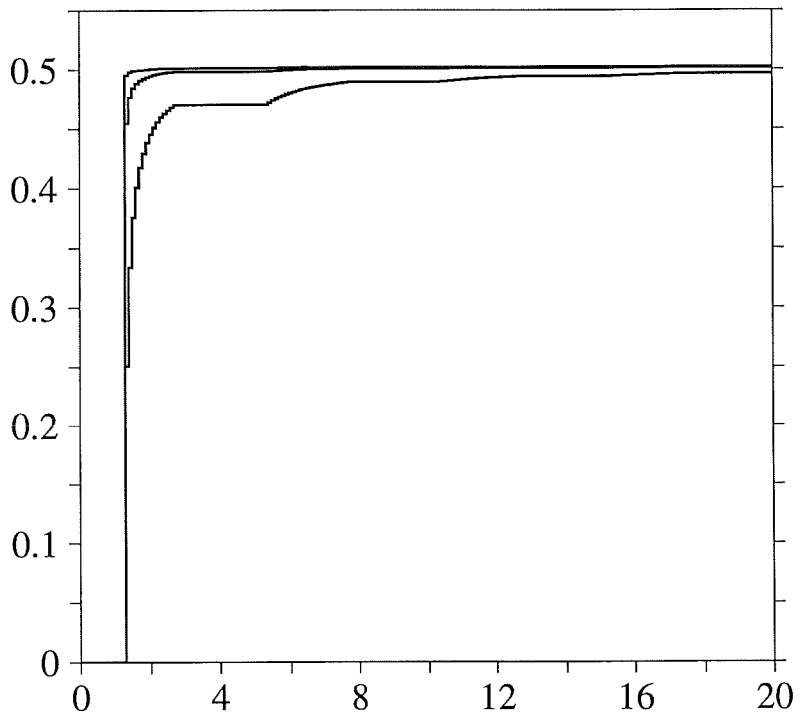
Figur 3.4 visar en simulering med modellreferenssystemet där uppdateringslagen i ekvation (3.9) används för $\gamma = [1,\ 1.5,\ 2]$. Skattningen konvergerar mot det korrekta värdet men konvergensen är långsam och har otrevliga under- och överslängar. Dessa under- och överslängar uppstår därför att man inte tar hänsyn till processdynamiken i uppdateringslagen för skattningen. Det tar ungefär lika lång tid för skattningen att konvergera oberoende av γ men oscillationerna blir mindre med ett litet värde på γ.

I figur 3.5 visas en simulering med minstakvadratskattaren där skattningsalgoritm (3.12) har använts för $P(0) = [1,\ 10,\ 100]$ och $\lambda = 1$ med samplingstiden $h = 0.1$. Skattningen konvergerar mot det korrekta värdet väldigt snabbt och utan några oscillationer. Tiden det tar för skattningen att konvergera minskar om initialvärdet på kovariansmatrisen $P(0)$ ökar.

Genom att studera figurerna 3.4 och 3.5 kan man dra slutsatsen att minstakvadratskattning är att föredra framför modellreferenssystemet, därför att minstakvadratskattningen konvergerar fortare och inte är oscillativ.



Figur 3.4 Skattning av glapp med modellreferenssystemet för $\gamma = [1,\ 1.5,\ 2]$.



Figur 3.5 Skattning av glapp med minstakvadratskattaren för $P(0) = [1,\ 10,\ 100]$ och $\lambda = 1$.

4. Reglering av kända och okända system med glapp

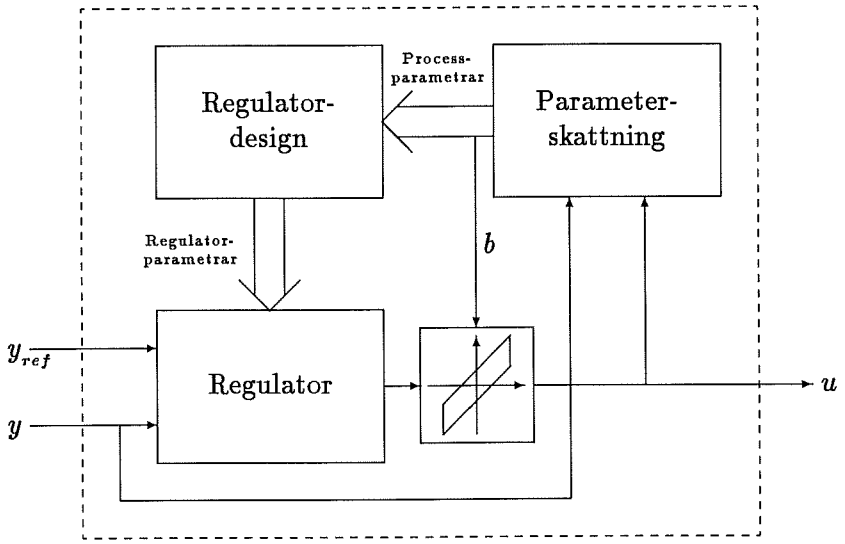
Detta kapitel kommer att behandla hur man sätter samman ett adaptivt reglersystem för system med glapp om processen är känd eller okänd.

Glappkompenseringen kommer att bestå av en glappinvers och skattningen av glapp och processparametrar kommer att göras med en rekursiv minstakvadratskattare.

I avsnitt 4.1 presenteras den regulatorstruktur som kommer att användas. Avsnitt 4.2 beskriver regulatorn och regulatordesignen. I avsnitt 4.3 ges en utökning av minstakvadratskattaren. Simuleringar av den adaptiva regulatorn visas i avsnitt 4.4.

4.1. Blockstruktur för en adaptiv regulator

Den regulatorstruktur som kommer att användas för den adaptiva regulatorn visas i figur 4.1. I tabell 4.1 beskrivs funktionsblocken som den adaptiva regulatorn är uppbyggd av.



Figur 4.1 Struktur för adaptiv regulator med adaptiv glappkompensering.

Med blockstrukturen som visas i figur 4.1 kan en adaptiv regulator för många olika typer av system konstrueras genom att ha olika parameterskattare och regulatorer.

Genom att använda någon av de i avsnitt 3.2 presenterade skattningsalgoritmerna kan en en regulator med konstanta parametrar och adaptiv glappkompensering eller en adaptiv regulator med adaptiv glappkompensering konstrueras.

Det är också enkelt att använda olika reglerstatergier genom att byta ut algoritmen för beräkning av styrsignalen. Det är även lätt att ändra designmetod genom att byta algoritmen för beräkning av regulatorparametrarna.

Funktionsblock	Funktion	Beskrivning
Regulator	Beräknar styrsignalen	Avsnitt 4.2
Glappinvers	Utför glappkompenseringen	Avsnitt 2.3
Parameterskattning	Beräknar skattningar av process-parametrar och glappets storlek	Avsnitt 3.2 och 4.3
Regulatordesign	Beräknar regulatorparametrar utifrån de skattade processparametrarna	Avsnitt 4.2

Tabell 4.1 De olika funktionsblocken i den adaptiva regulatorn.

4.2. Regulator och regulatordesign

Den regulator som används är en generell RST-regulator i diskret tid där parametrarna ges i form av polynomen $R(q)$, $S(q)$ och $T(q)$. Styrsignalen ges av ekvation (4.1).

$$u(t) = -\frac{S(q)}{R(q)}y(t) + \frac{T(q)}{R(q)}y_{ref}(t) \tag{4.1}$$

Polynomen R, S och T beräknas med polplacering, se [4], så att det slutna systemet får önskad överföringsfunktion. Regulatordesignen görs så att det slutna systemet får överföringsfunktionen $B(q)/A_m(q)$, dvs. inga nollställen förkortas bort, observerar polynomet $A_o(q)$ och den statiska förstärkningen ett. Regulatordesignen görs både med och utan integralverkan. Om $(q-1)$ är en faktor i R så har regulatorn integralverkan.

Om processen beskrivs av ekvation (4.2) fås det slutna systemet av ekvation (4.3) genom insättning av ekvation (4.1) i ekvation (4.2).

$$y(t) = \frac{B(q)}{A(q)}u(t) \tag{4.2}$$

$$y(t) = \frac{B(q)T(q)}{A(q)R(q) + B(q)S(q)}y_{ref}(t) = \frac{B(q)T(q)}{A_m(q)A_o(q)}y_{ref}(t) \tag{4.3}$$

Oavsett om regulatorn har integralverkan eller ej så ges T polynomet av ekvation (4.4).

$$T(q) = \frac{A_m(1)}{B(1)}A_o(q) \tag{4.4}$$

Utan integralverkan

Om regulatorn inte har integralverkan så ges R och S polynomen av lösningen till ekvation (4.5).

$$AR + BS = A_m A_o \tag{4.5}$$

Med integralverkan

Om regulatorn skall ha integralverkan måste $(q-1)$ vara en faktor i R. R' och S polynomen ges av lösningen till ekvation (4.6).

$$A(q-1)R' + BS = A_m A_o \tag{4.6}$$

R polynomet beräknas sedan som $R = (q-1)R'$.

4.3. Utökning av minstakvadratskattaren

Minstakvadratskattaren som presenteras i avsnitt 3.2 fungerar bäst om de skattade parametrarna är konstanta eller varierar långsamt. Om stegvisa ändringar förekommer i parametrarna kommer skattningen konvergera långsamt mot de nya parametervärdena.

För att snabbare följa stegvisa ändringar införs en periodisk återställning av kovariansmatrisen P (eng. covariance resetting), se [2]. Återställningen fungerar på så sätt att kovariansmatrisen med periodtiden T_{reset} sätts till αP_0, där α ligger mellan noll och ett och P_0 är kovariansmatrisens initialvärde.

Återställningen hindrar också att kovariansmatrisen växer obegränsat när prediktionsfelet är litet om man har en glömskefaktor λ som är mindre än ett.

4.4. Simulering av adaptiv regulator med glappkompensering

För att undersöka regulatorns egenskaper kommer servosystemet i figur 2.3 att simuleras där tillståndsåterkopplingen är utbytt med den adaptiva regulatorn som visas i figur 4.1.

Eftersom den adaptiva regulatorn är tidsdiskret måste det kontinuerliga systemet samplas. Om samplingstiden väljs till 0.1 s ger sampling av det kontinuerliga systemet $G(s)$ det diskreta systemet $H(q)$ enligt (4.7).

$$G(s) = \frac{k}{s(s+1)} \quad \Longrightarrow \quad H(q) = k\frac{0.0048q + 0.0047}{q^2 - 1.9048q + 0.9048} \tag{4.7}$$

Regulatordesignen görs så att det slutna systemets karakteristiska polynom och observerarpolynomet efter sampling blir enligt (4.8) samt att regulatorn har integralverkan. Eftersom glappkompenseringen görs med glappinvers behöver man inte ta hänsyn till glappet när man gör regulatordesingnen.

$$\begin{aligned} A_m(s) &= s^2 + 3.6s + 4 \quad &\Longrightarrow \quad A_m(q) &= q^2 - 1.6642q + 0.6977 \\ A_o(s) &= (s+3)^2 \quad &\Longrightarrow \quad A_o(q) &= q^2 - 1.4816q + 0.5488 \end{aligned} \tag{4.8}$$

Specifikationen i (4.8) ger regulatorparametrarna i (4.9) om processförstärkningen k är lika med ett.

$$\begin{aligned} R(q) &= q^2 - 1.3386q + 0.3386 \\ S(q) &= 20.1843q^2 - 36.2945q + 16.3466 \\ T(q) &= 3.5182q^2 - 5.2127q + 1.9308 \end{aligned} \tag{4.9}$$

Två olika simuleringar har gjorts med servosystemet för att studera den adaptiva regulatorns och den adaptiva glappkompenseringens egenskaper. Simuleringar har gjorts i följande två fall:

1. Kända konstanta processparametrar och okänt varierande glapp.
2. Okända varierande processparametrar och okänt konstant glapp.

I simuleringarna har börvärdet y_{ref} varit en fyrkantsvåg med amplituden ett och periodtiden 10 s. Eftersom parametrarna har varierat i simuleringen har återställning av kovariansmatrisen P använts för att snabbt följa parameterändringarna.

Konstanta processparametrar och varierande glapp

I simuleringen med konstanta processparametrar och varierande glapp har glappets storlek $b(t)$ varierat stegvist enligt ekvation (4.10) och processförstärkningen k har varit konstant lika med ett. Eftersom glappet är den enda parameter som varierar räcker det att skatta glappets storlek. I tabell 4.2 visas de parametrar och initialvärden som har använts i simuleringen.

$$b(t) = \begin{cases} 0.5 & 0 \leq t < 14 \\ 0.25 & 14 \leq t < 29 \\ 0.5 & 29 \leq t < 40 \end{cases} \tag{4.10}$$

R, S & T	λ	T_{reset}	α	P_0	b_0
(4.9)	1	10	1	10	0

Tabell 4.2 Initialvärden och parametrar till parameterskattaren vid skattning av glapp.

I figur 4.2 visas processvärdet, börvärdet och styrsignalen vid simuleringen. Figur 4.3 visar det skattade glappets storlek och det verkliga glappets storlek.

Simuleringen visar att det går att eliminera glappets inverkan även om glappet är okänt genom att använda adaptiv glappkompensering. De enda problemet som uppstår är när glappets storlek minskar och det skattade glappets storlek är för stort.

Eftersom glappinversen då är större än glappet kommer styrsignalen att oscillera kraftigt när felet är litet och detta leder till att ingen skattning sker. Detta syns tydligt i figur 4.2 där styrsignalen oscillerar mellan $t = 14$ och $t = 20$ förutom vid börvärdesändringen. Denna oscillation i styrsignalen påverkar dock inte processvärdet. Om man studerar skattningen av glappets storlek i figur 4.3 för samma tid ser man att ingen skattning sker då styrsignalen oscillerar. Anledningen till att skattningen går så snabbt till det rätta värdet vid $t = 20$ är att kovariansmatrisen P just har återställts till sitt initialvärde.

Förutom oscillationerna i styrsignalen när glappinversen är större än glappet får man även en översläng i processvärdet. Detta ser man vid $t = 18$ i figur 4.2.

Varierande processparametrar och konstant glapp

Om processparametrarna varierar måste både glappets storlek och processparametrarna skattas. I simuleringen har glappet storleken b konstant lika med 0.5 och processförstärkningen $k(t)$ har varierat enligt ekvation (4.11). I tabell 4.3 visas de initialvärden och parametrar som har använts vid simuleringen.

$$k(t) = \begin{cases} 1 & 0 \leq t < 14 \\ 2 & 14 \leq t < 64 \\ 1 & 64 \leq t < 100 \end{cases} \tag{4.11}$$

A_m & A_o	λ	T_{reset}	α	P_0	A_0 & B_0	b_0
(4.8)	1	10	0.25	(4.12)	(4.7)	0

Tabell 4.3 Initialvärden och parametrar till parameterskattaren vid skattning av processparametrar och glapp.

$$P_0 = \begin{bmatrix} 1000 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 5 \end{bmatrix} \tag{4.12}$$

I figur 4.4 visas processvärdet, börvärdet och styrsignalen vid simuleringen. Figur 4.6 innehåller det skattade glappets storlek och det verkliga glappets storlek samt den skattade processförstärkningen och den verkliga processförstärkningen. Figur 4.6 visar de skattade parametrarna i A och B polynomen samt den skattade glapparametern C.

Den skattade processförstärkningen beräknas enligt ekvation (4.13) där samplingstiden $h = 0.1$ s och där $\hat{b}_0$, $\hat{b}_1$ och $\hat{a}_2$ är de skattade parametrarna i den diskreta överföringsfunktionen.

$$\hat{k} = \frac{1}{h}\frac{\hat{b}_0 + \hat{b}_1}{1 - \hat{a}_2} \tag{4.13}$$

Nedan följer en härledning av ekvation (4.13). Parametrarna i den diskreta överföringsfunktionen $H(q)$ i ekvation (4.7) ges av ekvation 4.14, se [4].

$$\begin{aligned} b_0 &= \frac{k}{a}(ah - 1 + e^{-ah}) & b_1 &= \frac{k}{a}(1 - e^{-ah} - ahe^{-ah}) \\ a_1 &= -(1 + e^{-ah}) & a_2 &= e^{-ah} \end{aligned} \tag{4.14}$$

Insättning av $a = 1$ och $e^{-ah} = a_2$ i uttrycken för b_0 och b_1 i ekvation (4.14) ger uttrycken nedan.

$$b_0 = k(h - 1 + a_2) \qquad b_1 = k(1 - a_2 - ha_2)$$

Summering av b_0 och b_1 ger följande uttryck.

$$b_0 + b_1 = kh - k + ka_2 + k - ka_2 - kha_2 = kh(1 - a_2)$$

Eftersom $a_2 = -1 - a_1$ kan processförstärkningen k beräknas på två sätt enligt nedan.

$$k = \frac{b_0 + b_1}{h(1 - a_2)} \qquad \text{eller} \qquad k = \frac{b_0 + b_1}{h(2 + a_1)}$$

I figur 4.4 ser man att processvärdet följer börvärdet bra och att det inte blir några svängningar förutom en transient i början av simuleringen. Styrsignalen ser bra ut förutom oscillationerna vid $t = 20$ och $t = 70$, dessa oscillationer uppstår därför att glappinversen är större än det verkliga glappet.

Anledningen till att glappinversen är för stor vid dessa tidpunkter är att skattningen av glappet är känsligare än skattningen av processförstärkningen. Så när den verkliga processförstärkningen ändras blir det en transient i skattningen av glappet som man ser i figur 4.5. I figuren ser man även att det blir ett stationärt fel i skattningen av processförstärkningen. Detta fel beror på att excitationen är för dålig, eftersom styrsignalen är konstant stora delar av tiden. Den dåliga excitationen visar sig också genom att skattningen av processförstärkningen bara ändrar sig när det är en transient i styrsignalen.

Figur 4.6 visar att skattningen av A polynomets koefficienter inte påverkas av förändringen i processförstärkning. I figuren ser man också att skattningen av B polynomets koefficienter aviker från de rätta värdena. Summan av koefficienterna i B polynomet ligger däremot nära det rätta värdet och därför är den skattade processförstärkningen nära den rätta. Skattningen av glapparametern C är dock nära det rätta värdet. I tabell 4.4 visas de rätta värdena på b_0, b_1 och C för de olika tidpunkerna.

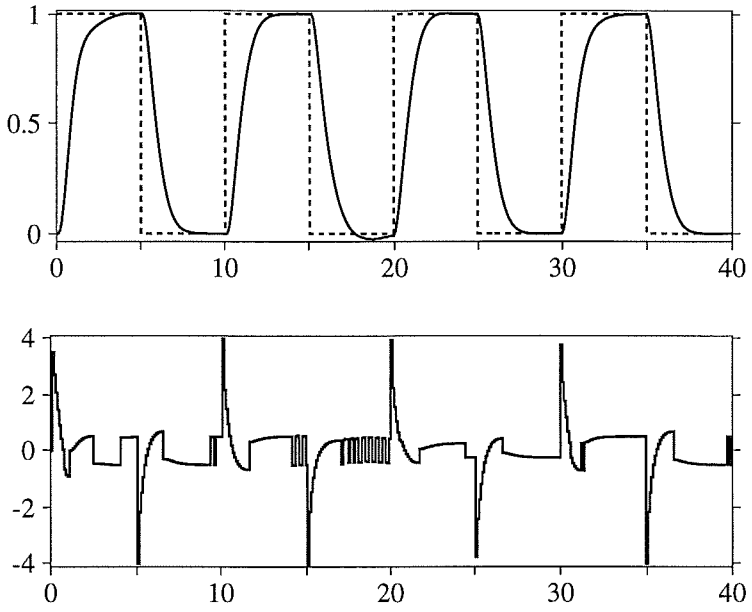
	b_0	b_1	C
$0 \leq t < 14$	0.00484	0.00468	0.00476
$14 \leq t < 64$	0.00967	0.00936	0.00952
$64 \leq t < 100$	0.00484	0.00468	0.00476

Tabell 4.4 De rätta väredena för koefficienterna i B polynomet och glappparametern C.

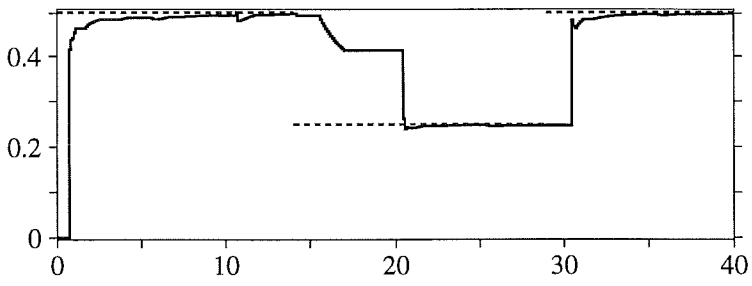
Även om det blir ett fel i skattningen av processförstärkningen så skattas glappets storlek rätt. Eftersom skattningen av glappet är rätt kommer glappinversen att har rätt storlek och således eliminera glappets inverkan.

Slutsatser av simuleringen

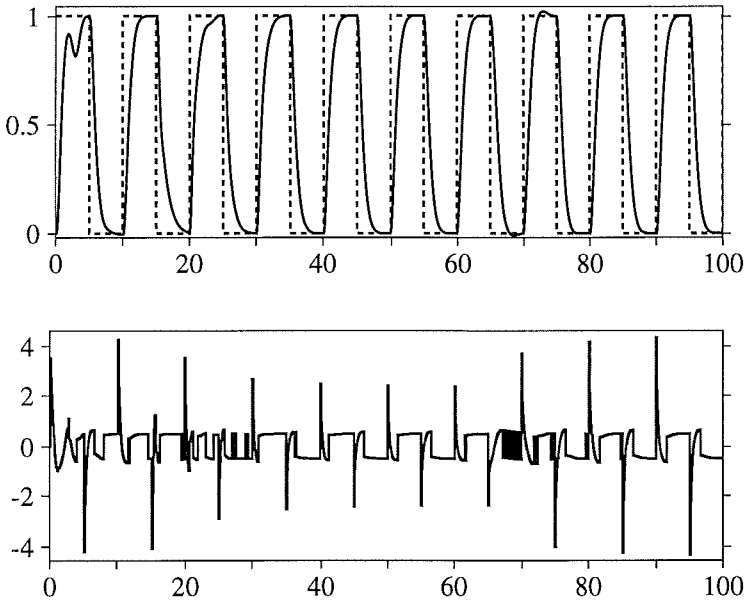
Av simuleringarna kan man dra slutsatsen att det går bra att skatta glappets storlek om processen är känd och glappets storlek varierar. Det har även visat sig att det går att skatta processen och glappet samtidigt när processens förstärkning varierar och glappet är konstant men okänt.



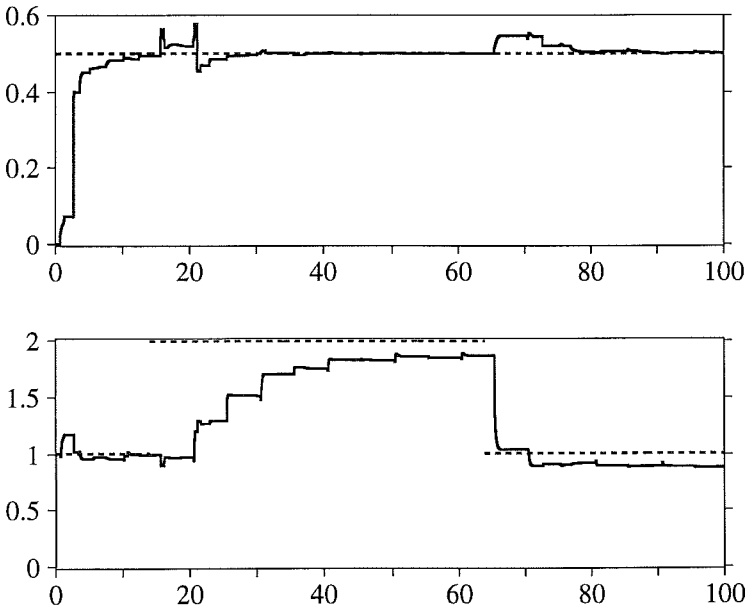
Figur 4.2 Simulering med konstanta processparametrar och varierande glapp. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen.



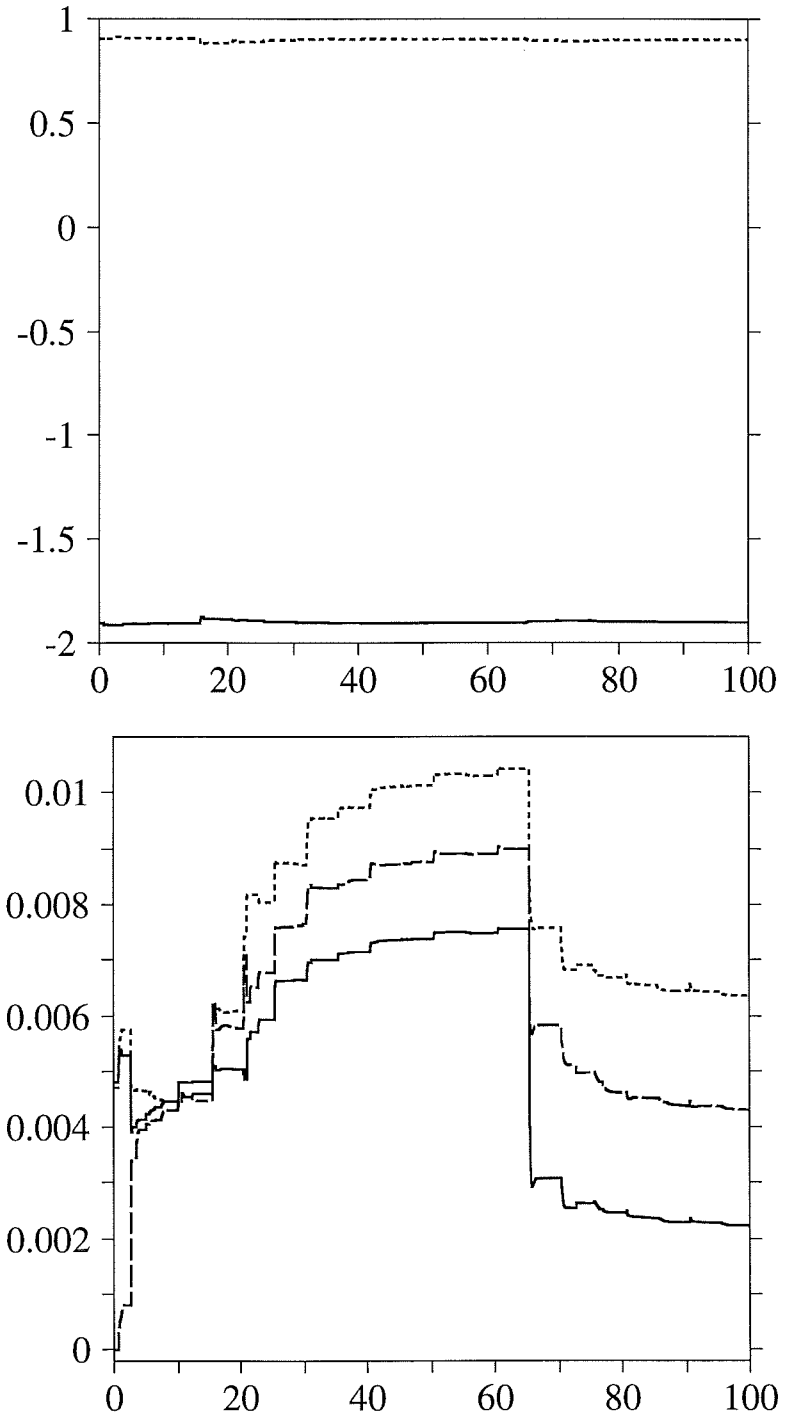
Figur 4.3 Simulering med konstanta processparametrar och varierande glapp. Bilden visar det skattade glappets storlek (heldragen) och det verkliga glappets storlek (streckad).



Figur 4.4 Simulering med varierande processparametrar och konstant glapp. Övre bilden visar processvärde (heldragen) och börvärde (streckad). Nedre bilden visar styrsignalen.



Figur 4.5 Simulering med varierande processparametrar och konstant glapp. Övre bilden visar det skattade glappets storlek (heldragen) och det verkliga glappets storlek (streckad). Nedre bilden visar den skattade processförstärkningen (heldragen) och den verkliga processförstärkningen (streckad).



Figur 4.6 Simulering med varierande processparametrar och konstant glapp. Övre bilden visar de skattade koefficienterna i A polynomet, $\hat{a}_1$ (heldragen) och $\hat{a}_2$ (streckad). Nedre bilden visar de skattade koefficienterna i B polynomet samt den skattade glapparametern C, $\hat{b}_0$ (heldragen), $\hat{b}_1$ (streckad) och $\hat{C}$ (streckad med långa streck).

5. Modellering och simulering med OmSim/Omola

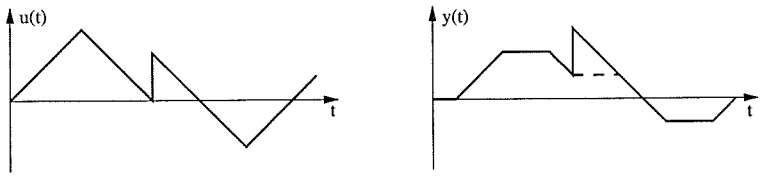
Detta kapitel behandlar de simuleringsmodeller som har använts i arbetet. Alla simuleringar har gjorts i OmSim och all modellering och implementering har gjorts i det objekt-orienterad modellerings språket Omola. En grundlig beskrivning av OmSim och Omola ges i [5].

I avsnitt 5.1 behandlas modellering av glapp och i avsnitt 5.2 ges en implementering av den adaptiva regulatorn. Avsnitt 5.3 beskriver hur en komplett simuleringsmodell sätts samman för att simulera servosystemet med den adaptiva regulatorn. Slutligen ges en kortfattad beskrivning av samtliga implementerade klasser i avsnitt 5.4.

5.1. Modellering och simulering av glapp

Utsignalen från glappet beror på insignalen och i vilket tillstånd glappet befinner sig i. Glappet har tre diskreta tillstånd, det kan antingen befinna sig i dödzonen, driva åt höger eller driva åt vänster. Tillståndsändring sker om insignalens derivata byter tecken och glappet driver åt något håll eller om glappet befinner sig i dödzonen och absolutbeloppet av differensen mellan insignalen och utsignalen blir större än eller lika med glappets storlek.

Om insignalen är diskontinuerlig blir det problem med insignalens derivata eftersom derivatan av ett steg inte är definierad i OmSim. Problemen som orsakas av detta illustreras i figur 5.1.



Figur 5.1 Insignal till glappet (vänster) och utsignal från glappet (höger). Den heldragna kurvan visar utsignalen när derivatan för steget är noll, den streckade kurvan visar den rätta utsignalen.

Med anledning av detta har två olika glappmodeller implementerats, en för helt kontinuerliga signaler och en som klarar alla typer av signaler. Den senare arbetar i diskret tid och ger en tidsdiskret utsignal. Den tidsdiskreta glappmodellen ger således endast en helt korrekt utsignal om insignalen är tidsdiskret och samplingen är synkroniserad med insignalen.

Om samplingstiden är kort ger den senare en god approximation av den korrekta utsignalen om insignalen är kontinuerlig. Om samplingstiden inte är kort detekteras inte tillståndsändringarna lika snabbt och glappet verkar vara större än vad det är.

Den tidsdiskreta varianten av glappet har använts i alla simuleringar eftersom om man använder en glappinvers så blir insignalen till glappet diskontinuerlig.

Båda glappmodellerna har tre terminaler, insignal u, utsignal y och tillståndssignal k. Tillståndssignalen som är en utsignal från glappet anger i vilket tilstånd som glappet befinner sig, $k = -1$ om glappet driver åt vänster, $k = 0$ om glappet befinner sig i dödzonen och $k = 1$ om glappet driver åt höger.

För att kunna styra glappets storlek har ytterligare en glappmodell gjorts. Denna modell är en utökning av den diskreta med ytterligare en insignal b som bestämmer glappets storlek. Den utökade diskreta modellen används som observerare i simuleringarna med modellreferenssystemet.

De tre olika glappmodellerna är definierade i klasserna `ContBacklash`, `DiscBacklash` och `ObsDiscBacklash`, se appendix A.

5.2. Implementering av en adaptiv regulator i Omola

Den adaptiva regulatorn som presenteras i kapitel 4 har implementerats i Omola. Regulatorns struktur har använts för att organisera implementeringen av regulatorn och det har resulterat i sex olika klasser, två regulatorer, tre parameterskattare och en för regulatordesign. De sex olika klasserna beskrivs i tabell 5.1.

Klass	Beskrivning
RST	Vanlig RST-regulator
RSTBC	RST-regulator med glappinvers
Design	Design av regulator
PE	Skattning av processparametrar
BE	Skattning av glappets storlek
PBE	Skattning av processparametrar och glappets storlek

Tabell 5.1 De olika klasserna som används för att bygga upp en adaptiv regulator.

De två regulatorklasserna har en gemensam superklass `Controller` som innehåller gemensamma attribut. De gemensamma attributen i `Controller` är terminalerna u, y och `yref`. Glappinversen är sammanslagen med regulatorn i `RSTBC` för att man inte skall få någon tidsfördröjning av styrsignalen pga. att samplingen inte är synkroniserad i regluatorn och glappinversen. Detta skulle även gått att lösa genom att låta regulatorn styra samplingen i glappinversen.

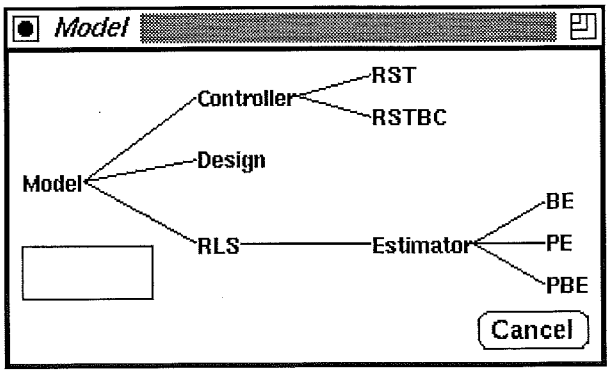
På samma sätt har de tre parameterskattarna en gemensam superklass `Estimator` som i sin tur har superklassen `RLS`. Klassen `RLS` innehåller en generell minstakvadratalgoritm som används av alla parameterskattare. Skattningen startas när händelsen `Estimate` avfyras och när skattningen är utförd avfyras händelsen `NewEst`. Klassen `Estiamtor` innehåller övergripande funktioner för parameterskattningen såsom initiering, återställning av kovariansmatrisen och möjlighet att stänga av och sätta på skattaren. Skattningen kan startas och stoppas genom att avfyra händelseterminalerna `StopEst`, `ContEst` eller `StartEst` i skattaren. Detta gör det enkelt att ansluta en övervakningsfunktion till skattaren som bestämmer när skattning skall ske. `Estimator` innehåller även terminalerna u och y som är insignaler till alla parameter-

skattare samt händelseterminalen `Sample` som bestämmer när uppdatering av skattningarna sker.

De tre parameterskattarna `PE`, `BE` och `PBE` är alla uppbyggda på samma sätt och har klassen `Estimator` som superklass. Det som skiljer dem åt är att regressionsvektorn `fi` är olika beroende på vad som skattas, se avsnitt 3.2. De har också olika ingångar och utgångar. Processparameter skattaren `PE` har två nya utgångar `ModParOut`, som innehåller de skattade processparametrarna, och `NewModPar`, en händelse som avfyras när en ny skattning av processparametrarna är tillgänglig. Båda dessa utgångar ansluts till motsvarande ingångar i `Design`. Glappskattaren `BE` har en ny ingång, `BackPos`, som är glappets tillstånd, och en ny utgång, `bl` som är det skattade glappets storlek. Utgången `bl` ansluts till regulatorn `RSTBC` och ingången `BackPos` kopplas till glappet. Process- och glapparameterskattaren `PBE` har samtliga nya in- och utgångar som `PE` och `BE` har.

Klassen `Design` använder Omolafunktionen `diophant` för att lösa DAB-ekvationerna (4.5), utan integralverkan, respektive (4.6), med integralverkan, och på så sätt beräkna regulatorparametrarna. För kommunikationen med parameterskatteren har `Design` ingångarna `EstModParIn` som ger de skattade processparametrarna, och `NewModPar`, som anger när de skattade processparametrarna har blivit uppdaterade. För att kommunicera med regulatorn har `Design` utgångarna `RegParOut`, som innehåller de beräknade regulatorparametrarna,och `NewRegPar`, som avfyras när regulatorparametrarna har blivit uppdaterade.

I figur 5.2 visas vilka superklasser de olika klasserna har.



Figur 5.2 Släktskap mellan de olika klasserna som har implementerats för att bygga upp de adaptiva regulatorerna.

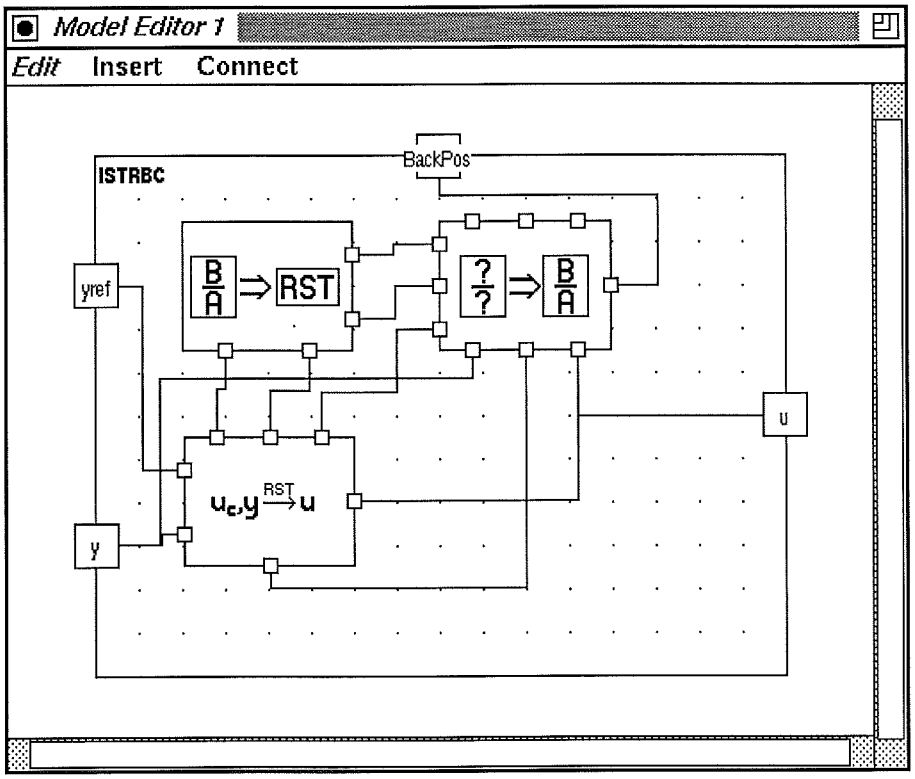
För att regulatorn och parameterskattaren skall vara synkroniserade så är det regulatorn som styr samplingen i parameterskattaren via händelseterminalerna `SampleNext` i regulatorn och `Sample` i parameterskattaren.

Dessa sex klasser gör det möjligt att sätta samman tre olika adaptiva regulatorer med olika egenskaper genom att kombinera olika skattare och regulatorer med varandra. De tre olika adaptiva regulatorerna beskrivs i tabell 5.2 där det också framgår vilka komponenter de olika regulatorerna är uppbyggda av.

Figur 5.3 visar en bild av hur de olika objekten är sammankopplade i den indirekta självinställande regulatorn med adaptiv glappkompensering. Bilden är tagen från modelleditorn som finns i Omsim. Linjerna mellan blocken är grafiska kopplingar mellan de olika objektens terminaler.

Klass	Beskrivning	Komponenter
ISTR	Indirekt självinställande adaptiv regulator	RST, PE, Design, Filter
ISTRBC	Indirekt självinställande adaptiv regulator med adaptiv glappkompensering	RSTBC, PBE, Design
STBC	Regulator med konstanta parametrar men adaptiv glappkompensering	RSTBC, BE

Tabell 5.2 De tre olika regulator klasserna.



Figur 5.3 Visar hur olika moduler kopplas samman till en indirekt självinställande regulator med adaptiv glappkompensering.

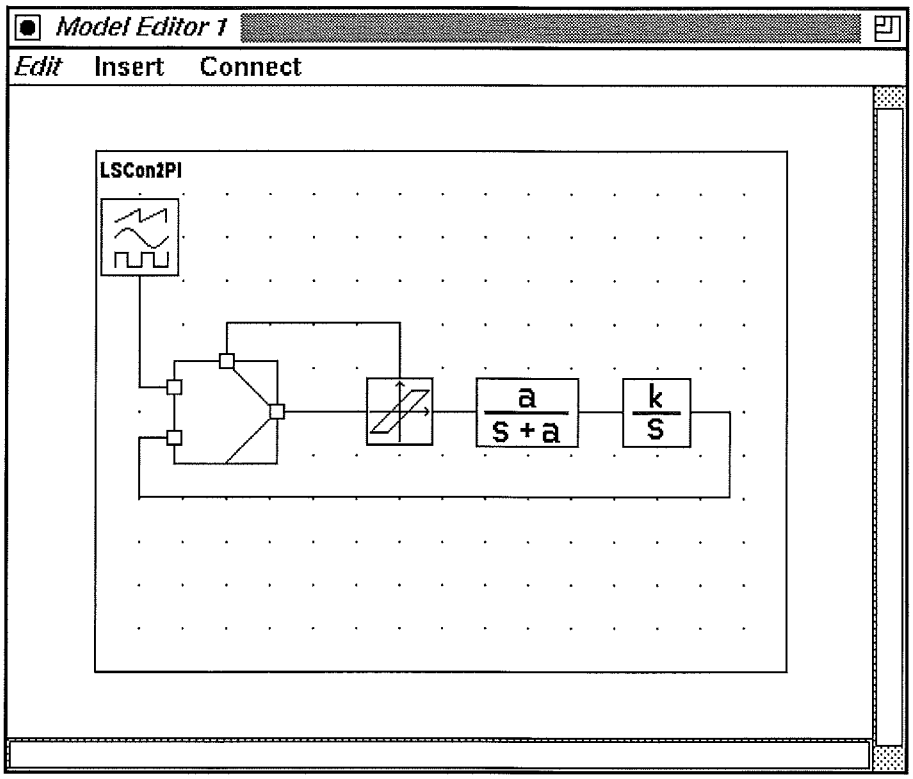
Förutom de grafiska kopplingarna som åskådliggörs i figur 5.3 finns det även variabel kopplingar mellan objekten, för att de skattade processens gradtal skall vara samma i parameterskattningen och regulatordesignen samt så att regulatorpolynomen har samma gradtal i regulatorn och regulatordesignen.

För överföring av process- och regulatorparametrar har två terminalklasser implementerats `ModParTerm`, för modellparametrar, och `RegParTerm`, för regulatorparametrar till en RST-regulator.

För mer detaljer om implementeringen av de olika klasserna för adaptiv reglering, se appendix A.

5.3. Simuleringsmodell för servosystem med adaptiv regulator

Figur 5.4 visar en bild från modelleditorn med en simuleringsmodell av servosystemet med den adaptiva regulatorn med adaptiv glappkompensering. Modellen är uppbyggd av objekt ur klasserna `FunctionGenerator`, `ISTRBC`, `DiscBacklash`, `FirstOrder` och `Integrator`.



Figur 5.4 Sammankoppling av de olika modellerna för att simulera servosystemet med den adaptiva regulatorn.

Gradtalen på processen som skall skattas och om regulatorn skall designas med integralverkan eller ej måste specificeras innan modellen instansieras och är konstanta under simuleringen. Det görs genom att i `ISTRBC` tilldela variablerna `degA` och `degB` respektive gradtal och `Integral` antingen ett för integralverkan eller noll utan integralverkan, se klassen `LSCon2PI` i appendix A. `LSCon2PI` innehåller även en beräkning av processförstärkningen utifrån de skattade processparametrarna.

5.4. Modell klasser

Tabell 5.3 innehåller alla implementerade klasser sorterade efter vilket bibliotek de tillhör och med en kort beskrivning av funktionen. Omolaprogrammen redovisas i appendix A.

Bibliotek	Klass	Beskrivning
Adaptive	BacklashMRAS	Modellreferensskattare
	BE	Skattning av glappets storlek
	Design	Regulatordesign
	Estimator	Generell parameterskattare
	ModParTerm	Terminal för processparametrar
	PBE	Skattning av processparametrar och glappets storlek
	PE	Skattning av processparametrar
	RegParTerm	Terminal för regulatorparametrar
	RLS	Generell minstakvadratalgoritm
Backlash	ContBacklash	Glappmodell i kontinuerlig tid
	DiscBacklash	Glappmodell i diskret tid
	ObsDiscBacklash	Glappmodell i diskret tid
Con	LSCon2I	Simulering av STBC-regulatorn
	LSCon2PI	Simulering av ISTRBC-regulatorn
	MRASCon	Simulering av modellrefernssystemet
	StateFB	Simulering med tillståndsåterkoppling
	StateFBComp	Simulering med tillståndsåterkoppling och fasavancering
	StateFBComp2	Simulering med tillståndsåterkoppling och glappinvers
Control	BacklashInverse	Glappinvers
	Controller	Grundklass för regulatorer
	Filter	Diskret filter
	ISTR	Indirekt självinställande regulator
	ISTRBC	Indirekt självinställande regulator med adaptiv glappinvers
	PhaseLead	Fasavancering
	RST	RST-regulator
	RSTBC	RST-regulator med glappinvers
	State	Tillståndsåterkoppling
	STBC	Fix regulator med adaptiv glappinvers
Process	FirstOrder	Första ordningens process
	Integrator	Integrator
	SecondOrder	Andra ordningens process
Signal	FunctionGenerator	Funktionsgenerator

Tabell 5.3 Samtliga implementerade klasser med kortfattad beskrivning.

6. Avslutning

Detta kapitel innehåller en sammanställning av resultaten i detta arbete. Avsnitt 6.2 ger förslag till fortsättning.

6.1. Slutsatser

I detta arbete har reglering av system med glapp studerats. Det har visat sig att om glapp förekommer i reglerkretsen försämras regleringen av systemet avsevärt om man inte tar hänsyn till glappet när regulatorn dimensioneras. Två olika metoder för glappkompensering har undersökts, fasavancering respektive glappinvers. Resultaten av undersökningen sammanfattas i tabell 6.1. Eftersom glappinversen eliminerar glappet fullständigt och är oberoende av

Fasavancering	Glappinvers
linjär kompensering	olinjär kompensering
oberoende av glappets storlek	beroende av glappets storlek
beroende av process & regulator	oberoende av process & regulator
oberoende av glappets placering	glappet måste verka på ingången
hindrar självsvängning	eliminerar glappet totalt
okänslig för felaktiga parametrar	känslig för felaktiga parametrar
inga krav på styrsignalen	styrsignalen måste ändras stegvis

Tabell 6.1 Jämförelse mellan fasavancering och glappinvers.

process- och regulatorparametrar har glappinvers använts i de regulatorer som undersökts.

Eftersom kompenseringen med glappinvers är känslig för felaktiga parametrar måste glappets parametrar skattas. Skattning av glappet har gjorts med två metoder, modellanpassning och parameterskattning. Modellanpassningen bygger på ett modellreferenssystem där en observerare anpassas så att felet blir litet. Parameterskattaren bygger på en minstakvadratskattning av det samplade tidsdiskreta systemet. Båda metoderna kräver att glappets tillstånd är känt, man måste alltså veta om glappet befinner sig i dödzonen eller ej. Modellanpassningen kräver också att processen är känd. Parameterskattaren går att använda dels för att skatta glappets storlek och dels för att både skatta glappets storlek och processparametrarna. Simuleringar av de båda metoderna enbart för skattning av glapp visade att parameterskattaren konvergerade mycket snabbare än modellanpassningen. Med anledning av detta valdes parameterskattaren att användas för skattning av glapp.

Minstakvadratskattaren för skattning av glapp då processen är känd undersöktes närmare med avsende på konvergensen hos skattningen. Ett resultat av undersökningen är följande excitationsvillkor

$$\lim_{n\to\infty} \frac{1}{n} \sum_{t=0}^{n} \chi_k(t) > 0$$

där

$$\chi_k = \begin{cases} 1 & \text{om glappet är i kontakt och} \\ & \text{har varit i kontakt i minst } n_b h \text{ sekunder} \\ 0 & \text{annars} \end{cases}$$

och h är skattarens samplingstid och n_b är gradtalet hos den tidsdiskreta överföringsfunktiones täljarpolynom $B(q)$. Om excitationsvillkoret är uppfyllt och det inte förekommer några störningar eller felaktigheter hos processparametrarna i skattaren konvergerar skattningen mot det korrekta värdet. Om det förekommer en konstant laststörning beror felet i skattningen på laststörningenstorlek och insignalens utseende. På samma sätt beror felet i skattningen på felaktigheter i skattarens processparametrar och insignalens utseende när skattarens processparametrar är felaktiga.

Glappinversen och minstakvadratskattaren kan kombineras med en RST-regulator till en konstant regulator eller en regulator med adaptiv glappkompensering. RST-regulatorns parametrar beräknas med polplacering. I fallet när processparametrarna skattas uppdateras regulatorparametrarna när en ny skattning av processparametrarna är tillgänglig.

För att verifiera att regulatorn fungerar utfördes simuleringar på ett servosystem med glapp verkande på ingången till systemet. Två fall undersöks dels när glappet varierar och processparametrarna är kända och konstanta och dels när glappet är konstant men okänt och processparametrarna varierar. Simuleringen av regulatorn med konstanta parametrar visar att det går att skatta glappets storlek och använda skattningen till glappkompensering. Simuleringen av den adaptiva regulatorn visar att det går att skatta både processparametrar och glappets storlek och använda processparametrarna till regulatordesign.

6.2. Förslag på fortsättning

Några tänkbara fortsättningar av arbetet är:

- Undersök om det går att använda någon form av glappinvers när det finns dynamik mellan styrsignalen och glappet.
- Ta fram en metod för att bestämma om glappet befinner sig i dödzonen eller ej.
- Undersök om det går att skatta glappets storlek utan att veta vilket tillstånd glappet befinner sig i.
- Undersök hur mätbrus påverkar glappinversen och skattningen av glappets storlek, och modifiera glappinversen så att brusets påverkan minimeras.
- Finn en metod för att skatta glappets storlek när det förekommer laststörningar i systemet.
- Ta fram skydd som undviker oscillationer i styrsignalen när glappinversen är större än glappet.

7. Litteraturförteckning

[1] H. K. KHALIL, "Nonlinear Systems", Macmillan Publishing Company, 1992.

[2] K. J. ÅSTRÖM och B. WITTENMARK, "Adaptive Control", Addison-Wesley Publishing Company, 1989.

[3] R. JOHANSSON , "Synthesis of servo system subject to backlash", Labhandledning i kursen Olinjär reglering & Servosystem vid Inst. för Reglerteknik LTH, 1995.

[4] K. J. ÅSTRÖM och B. WITTENMARK, "Computer-Controlled Systems: Theory and Design", Prentice-Hall International, 1990.

[5] M. ANDERSSON, "Object-Oriented Modeling and Simulation of Hybrid Systems", Doktorsavhandling vid Inst. för Reglerteknik LTH , 1994.

[6] G. TAO och P. KOKOTOVIĆ, "Adaptive Control of Systems with Sensor and Actuator Nonlinearities", Under utgivning.

A. Klassdefinitioner i Omola

Detta appendix innehåller samtliga modeller som har använts i arbetet. För beskrivning av modellerna se kapitel 5.

För att minska volymen har alla delar av modellerna som hör samman med grafiken och layouten i modelleditorn tagits bort. Således innehåller modellerna bara ekvationer och deklarationer av objekt, parametrar och variabler.

Modellerna är uppdelade i bibliotek och är sorterade i bokstavsordning efter först biblioteksnamn och sedan klassnamn.

A.1. Adaptive

BacklashMRAS

```
BacklashMRAS ISA Base::Model WITH
  gamma ISA Base::Parameter WITH default := 1; END;

  y ISA Base::SimpleInput;
  ym ISA Base::SimpleInput;
  u ISA Base::SimpleInput;
  kObs ISA Base::DiscreteTerminal;
  kBack ISA Base::DiscreteTerminal;
  b ISA Base::SimpleOutput;

  x TYPE Real;
  e TYPE Real;

  b := if x < 0 then 0 else x;
  e := ym - y;
  x' = kBack*abs(kObs)*gamma*e;

  Graphic ISA super::Graphic;
END;
```

BE - Backlash Estimator

```
BE ISA Adaptive::Estimator WITH
  InitA ISA Std::VectorPar WITH n := degA + 1; END;
  InitB ISA Std::VectorPar WITH n := degB + 1; END;
  Initbl ISA Base::Parameter WITH default := 0; END;

  degA TYPE Integer := 1;
  degB TYPE Integer := 1;
  NbrOfPar := 1;

  BackPos ISA Base::SimpleInput;
  bl ISA Base::DiscreteTerminal;

  State TYPE DISCRETE (Est, NoEst);
```

```
  fitmp TYPE DISCRETE Column[NbrOfPar + 1];
  VBackPos TYPE DISCRETE Column[degB + 3];
  yv TYPE DISCRETE Column[degA + 1];
  yvtmp TYPE DISCRETE Column[degA + 2];
  uv TYPE DISCRETE Column[degB + 1];
  uvtmp TYPE DISCRETE Column[degB + 2];

  WHEN Init or StartEst DO
    new(theta) := ones(1, degB + 1)*InitB*Initbl;
    new(fitmp) := zeros(NbrOfPar + 1, 1);
    new(VBackPos) := zeros(degB + 3, 1);
    new(yvtmp) := zeros(degA + 2, 1);
    new(uvtmp) := zeros(degB + 2, 1);
    new(State) := 'NoEst;
  END;

  WHEN Sample DO
    new(yvtmp) := [y; yvtmp[1..degA + 1]];
    new(yv) := new(yvtmp[1..degA + 1]);
    new(uvtmp) := [u; uvtmp[1..degB + 1]];
    new(uv) := new(uvtmp[1..degB + 1]);
    new(fitmp) := [-BackPos; fitmp[1]];
    new(fi) := fitmp[1];
    new(VBackPos) := [BackPos; VBackPos[1..degB + 2]];
    new(val) := trans(InitA)*new(yv) - trans(InitB)*uv;
    new(State) := if abs(ones(1, degB + 3)*new(VBackPos))
                              < degB + 3 then 'NoEst
                  else 'Est;
    if Mode == 'On and State == 'Est then schedule(Estimate, 0);
  END;

  WHEN NewEst DO
    new(bl) := theta/(ones(1, degB + 1)*InitB);
  END;
END;
```

Design

```
Design ISA Base::Model WITH
  Init ISA Base::Event;

  Ao ISA Std::VectorPar WITH n := degAo + 1; END;
  Am ISA Std::VectorPar WITH n := degAm + 1; END;

  Integral TYPE Integer := 0;
  degA TYPE Integer := 1;
  degB TYPE Integer := 0;
  degAm TYPE Integer := degA;
  degAo TYPE Integer := degA + Integral - 1;
  degR TYPE Integer := degA + Integral - 1;
  degS TYPE Integer := degA + Integral - 1;
  degT TYPE Integer := degA + Integral - 1;
```

```
  NewModPar ISA Base::EventInput;
  NewRegPar ISA Base::EventOutput;
  EstModParIn ISA Adaptive::ModParTerm WITH
    na := degA + 1;
    nb := degB + 1;
    A.causality := 'input;
    B.causality := 'input;
  END;
  RegParOut ISA Adaptive::RegParTerm WITH
    nr := degR + 1;
    ns := degS + 1;
    nt := degT + 1;
    R.causality := 'output;
    S.causality := 'output;
    T.causality := 'output;
  END;

  AmAo TYPE DISCRETE Row[degAm + degAo + 1];
  RStmp TYPE DISCRETE Row[degR + degS + 2 - Integral];
  Atmp TYPE DISCRETE Row[degA + 2];

  WHEN NewModPar DO
    new(AmAo) := conv(trans(Am),trans(Ao));
    new(Atmp) := if Integral == 0 then [trans(EstModParIn.A), 0]
                 else conv(trans(EstModParIn.A),[1, -1]);
    new(RStmp) := diophant(new(Atmp[1..degA + Integral + 1]),
                           trans(EstModParIn.B), new(AmAo));
    new(RegParOut.R) := if Integral == 0 then
                          trans(new(RStmp[1..degR + 1]))
                        else
                          trans(conv(new(RStmp[1..degR]),[ 1, -1]));
    new(RegParOut.S) := trans(new(RStmp[degR + 2 - Integral..
                                  degR + degS + 2 - Integral]));
    new(RegParOut.T) := ones(1, degAm + 1)*Am/(ones(1, degB + 1)
                                        *EstModParIn.B)*Ao;
    schedule(NewRegPar, 0);
  END;
END;
```

Estimator

```
Estimator ISA Adaptive::RLS WITH
  Init ISA Base::Event;
  ResetP ISA Base::Event;

  P0 ISA Std::SqMatrixPar WITH
    n := NbrOfPar;
    default := eye(NbrOfPar);
  END;
  ResetPeriod ISA Base::Parameter WITH default := 100; END;
  alfa ISA Base::Parameter WITH default := 1; END;
```

```
  PReseting ISA Base::Parameter WITH
    value TYPE (On, Off);
    default TYPE (On, Off) := 'Off;
  END;

  Sample ISA Base::EventInput;
  StopEst ISA Base::EventInput;
  StartEst ISA Base::EventInput;
  ContEst ISA Base::EventInput;
  u ISA Base::SimpleInput;
  y ISA Base::SimpleInput;

  Mode TYPE DISCRETE (On, Off);

  WHEN Init or StartEst DO
    new(P) := P0;
    new(K) := zeros(NbrOfPar, 1);
    new(Mode) := 'On;
    schedule(ResetP, 0);
  END;

  WHEN StopEst DO
    new(Mode) := 'Off;
  END;

  WHEN ContEst DO
    new(Mode) := 'On;
  END;

  WHEN ResetP DO
    new(P) := if PReseting == 'On then P0*alfa else P;
    schedule(ResetP, ResetPeriod);
  END;
END;
```

ModParTerm

```
ModParTerm ISA Base::RecordTerminal WITH
  na TYPE Integer := 1;
  nb TYPE Integer := 1;
  A ISA Std::DiscrVectorTerm WITH n := na; END;
  B ISA Std::DiscrVectorTerm WITH n := nb; END;
END;
```

PBE - Plant and Backlash Estimator

```
PBE ISA Adaptive::Estimator WITH
  InitA ISA Std::VectorPar WITH n := degA + 1; END;
  InitB ISA Std::VectorPar WITH n := degB + 1; END;
  Initbl ISA Base::Parameter WITH default := 0; END;

  degA TYPE Integer := 1;
  degB TYPE Integer := 1;
```

```
  NbrOfPar := degA + degB + 2;

  BackPos ISA Base::SimpleInput;
  bl ISA Base::DiscreteTerminal;
  ModParOut ISA Adaptive::ModParTerm WITH
    na := degA + 1;
    nb := degB + 1;
    a.causality := 'output;
    b.causality := 'output;
  NewModPar ISA Base::EventOutput WITH

  State TYPE DISCRETE (Est, NoEst);
  fitmp TYPE DISCRETE Column[NbrOfPar + 2];
  VBackPos TYPE DISCRETE Column[degB + 3];

  WHEN Init or StartEst DO
    new(theta) := [InitA[2..degA + 1]; InitB;
                   ones(1, degB + 1)*InitB*Initbl];
    new(fitmp) := zeros(NbrOfPar + 2, 1);
    new(VBackPos) := zeros(degB + 3, 1);
    new(State) := 'NoEst;
    schedule(ResetP, 0);
    schedule(NewEst, 0);
  END;

  WHEN Sample DO
    new(fitmp) := [-y; fitmp[1..degA]; u;
                   fitmp[degA + 2..NbrOfPar]; -BackPos];
    new(fi) := [fitmp[1..degA]; fitmp[degA + 2..NbrOfPar];
                                        fitmp[NbrOfPar + 2]];
    new(VBackPos) := [BackPos; VBackPos[1..degB + 2]];
    new(State) := if abs(ones(1, degB + 3)*new(VBackPos))
                                       < degB + 3 then  'NoEst
                  else 'Est;
    new(val) := y;
    if Mode == 'On and State == 'Est then schedule(Estimate, 0);
  END;

  WHEN NewEst DO
    new(ModParOut.a) := [1; new(theta[1..degA])];
    new(ModParOut.b) := [new(theta[degA + 1..degA + degB + 1])];
    new(bl) := new(theta[NbrOfPar])/(ones(1, degB + 1)*
                                          new(ModParOut.b));
    schedule(NewModPar, 0);
  END;
END;
```

PE - Plant Estimator

```
PE ISA Adaptive::Estimator WITH
  InitA ISA Std::VectorPar WITH n := degA + 1; END;
  InitB ISA Std::VectorPar WITH n := degB + 1; END;
```

```
  degA TYPE Integer := 1;
  degB TYPE Integer := 1;

  ModParOut ISA Adaptive::ModParTerm WITH
    na := degA + 1;
    nb := degB + 1;
    A.causality := 'output;
    B.causality := 'output;
  END;
  NewModPar ISA Base::EventOutput;

  NbrOfPar := degA + degB + 1;
  fitmp TYPE DISCRETE Column[NbrOfPar + 2];

  WHEN Init or StartEst DO
    new(theta) := [InitA[2..degA + 1]; InitB];
    new(fitmp) := [zeros(NbrOfPar + 2, 1)];
    schedule(NewEst, 0);
  END;

  WHEN Sample DO
    new(fitmp) := [-y; fitmp[1..degA]; u;
                          fitmp[degA + 2..NbrOfPar + 1]];
    new(fi) := [fitmp[1..degA]; fitmp[degA + 2..NbrOfPar + 1]];
    new(val) := y;
    IF Mode == 'On THEN schedule(Estimate, 0);
  END;

  WHEN NewEst DO
    new(ModParOut.A) := [1; new(theta[1..degA])];
    new(ModParOut.B) := [new(theta[degA + 1..NbrOfPar])];
    schedule(NewModPar, 0);
  END;
END;
```

RegParTerm

```
RegParTerm ISA Base::RecordTerminal WITH
  nr TYPE Integer := 1;
  ns TYPE Integer := 1;
  nt TYPE Integer := 1;
  R ISA Std::DiscrVectorTerm WITH n := nr; END;
  S ISA Std::DiscrVectorTerm WITH n := ns; END;
  T ISA Std::DiscrVectorTerm WITH n := nt; END;
END;
```

RLS - Recursive Least Square

```
RLS ISA Base::Model WITH
  Estimate ISA Base::Event;
  NewEst ISA Base::Event;
```

```
  lambda ISA Base::Parameter WITH default := 1; END;

  NbrOfPar TYPE Integer := 1;
  P TYPE DISCRETE Matrix[NbrOfPar, NbrOfPar];
  K TYPE DISCRETE Column[NbrOfPar];
  fi TYPE DISCRETE Column[NbrOfPar];
  theta TYPE DISCRETE Column[NbrOfPar];
  val TYPE DISCRETE Real;

  WHEN Estimate DO
    new(K) := P*fi/(lambda + trans(fi)*P*fi);
    new(theta) := theta + new(K)*(val - trans(fi)*theta);
    new(P) := (P - new(K)*trans(fi)*P)/lambda;
    schedule(NewEst, 0);
  END;
END;
```

A.2. Backlash

ContBacklash

```
ContBacklash ISA Base::Model WITH
  Init ISA Event;

  b ISA Parameter WITH default := 0; END;

  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;
  k ISA Base::DiscreteTerminal;

  mode TYPE DISCRETE Integer;
  yold TYPE DISCRETE Real;
  du TYPE Real;

  du := u';

  WHEN Init DO
    new(mode) := 0;
    new(k) := 0;
    new(yold) := 0;
  END;

  WHEN mode == 0 and u - yold < -b DO
    new(mode) := -1;
    new(k) := 1;
  END;

  WHEN mode == 0 and u - yold > b DO
    new(mode) := 1;
    new(k) := 1;
```

```
  END;

  WHEN mode == 1 and du < 0 or mode == -1 and du > 0 DO
    new(mode) := 0;
    new(k) := 0;
    new(yold) := y;
  END;

  y := if mode == 1 then u - b
       else if mode == -1 then u + b
       else yold;
END;
```

DiscBacklash

```
DiscBacklash ISA Base::Model WITH
  Sample ISA Event;
  Init ISA Event;

  h ISA Parameter WITH default := 0.01; END;
  b ISA Parameter WITH default := 0.0; END;

  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;
  k ISA Base::DiscreteTerminal;

  ytmp TYPE DISCRETE Real;
  yold TYPE DISCRETE Real;

  WHEN Sample or Init DO
    new(k) := if u - yold > b then 1
              else if u - yold < -b then -1
              else 0;
    new(ytmp) := if new(k) == 1 then u - b
                 else if new(k) == -1 then u +
                 else yold;
    new(yold) := new(ytmp);
    schedule(Sample, h);
  END;

  y := ytmp;
END;
```

ObsDiscBacklash

```
ObsDiscBacklash ISA Base::Model WITH
  Sample ISA Event;
  Init ISA Event;

  h ISA Parameter WITH default := 0.005; END;

  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;
```

```
  k ISA Base::DiscreteTerminal;
  b ISA Base::SimpleInput;

  ytmp TYPE DISCRETE Real;
  yold TYPE DISCRETE Real;

  WHEN Sample or Init DO
    new(k) := if u - yold > b - eps then 1
              else if u - yold < -b + eps then -1
              else 0;
    new(ytmp) := if new(k) == 1 then u - b
                 else if new(k) == -1 then u + b
                 else yold;
    new(yold) := new(ytmp);
    schedule(Sample, h);
  END;

  y := ytmp;
END;
```

A.3. Con

LSCon2I

```
LSCon2I ISA Base::Model WITH
  FuncGen ISA Signal::FunctionGenerator;
  Cont ISA Control::STBC WITH
    degA := 2;
    degB := 1;
    degR := 2;
    degS := 2;
    degT := 2;
  END;
  Backlash ISA Backlash::DiscBacklash;
  Proc ISA Process::FirstOrder;
  Int ISA Process::Integrator;

  FuncGen.x AT Cont.yref;
  Cont.u AT Backlash.u;
  Backlash.y AT Proc.u;
  Proc.y AT Int.u;
  Int.y AT Cont.y;
  Cont.BackPos AT Backlash.k;
END;
```

LSCon2PI

```
LSCon2PI ISA Base::Model WITH
  FuncGen ISA Signal::FunctionGenerator;
  Cont ISA Control::ISTRBC WITH
```

```
    degA := 2;
    degB := 1;
    Integral := 1;
  END;
  Backlash ISA Backlash::DiscBacklash;
  Proc ISA Process::FirstOrder;
  Int ISA Process::Integrator;

  k TYPE DISCRETE Real;

  WHEN Cont.Estimator.NewModPar DO
    new(k) := (Cont.Estimator.ModParOut.b[1]+
               Cont.Estimator.ModParOut.b[2])/
               (1-Cont.Estimator.ModParOut.a[3])/
               Cont.Controller.h;
  END;

  FuncGen.x AT Cont.yref;
  Cont.u AT Backlash.u;
  Backlash.y AT Proc.u;
  Proc.y AT Int.u;
  Cont.BackPos AT Backlash.k;
  Int.y AT Cont.y;
END;
```

MRASCon

```
MRASCon ISA Base::Model WITH
  FuncGen ISA Signal::FunctionGenerator;
  Backlash ISA Backlash::DiscBacklash;
  Process ISA Process::FirstOrder;
  ObsBacklash ISA Backlash::ObsDiscBacklash;
  Observer ISA Process::FirstOrder;
  Adapt ISA Adaptive::BacklashMRAS;

  FuncGen.x AT Backlash.u;
  FuncGen.x AT ObsBacklash.u;
  Backlash.y AT Process.u;
  ObsBacklash.y AT Observer.u;
  Process.y AT Adapt.y;
  Observer.y AT Adapt.ym;
  Backlash.k AT Adapt.kBack;
  ObsBacklash.k AT Adapt.kObs;
  Adapt.b AT ObsBacklash.b;
END;
```

StateFB

```
StateFB ISA Base::Model WITH
  FuncGen ISA Signal::FunctionGenerator;
  Cont ISA Control::State;
  Backlash ISA Backlash::DiscBacklash;
  Proc ISA Process::FirstOrder;
```

```
  Int ISA Process::Integrator;

  FuncGen.x AT Cont.yref;
  Cont.u AT Backlash.u;
  Backlash.y AT Proc.u;
  Proc.y AT Int.u;
  Int.y AT Cont.pos;
  Proc.y AT Cont.vel;
END;
```

StateFBComp

```
StateFBComp ISA Base::Model WITH
  FuncGen ISA Signal::FunctionGenerator;
  Cont ISA Control::State;
  Backlash ISA Backlash::DiscBacklash;
  Proc ISA Process::FirstOrder;
  Int ISA Process::Integrator;
  Comp ISA Control::PhaseLead;

  FuncGen.x AT Cont.yref;
  Backlash.y AT Proc.u;
  Proc.y AT Int.u;
  Int.y AT Cont.pos;
  Proc.y AT Cont.vel;
  Cont.u AT Comp.u;
  Comp.y AT Backlash.u;
END;
```

StateFBComp2

```
StateFBComp2 ISA Base::Model WITH
  b ISA Base::Parameter WITH default := 0; END;

  BacklashInverse1.b := b;

  FuncGen ISA Signal::FunctionGenerator;
  Cont ISA Control::State;
  Backlash ISA Backlash::DiscBacklash;
  Proc ISA Process::FirstOrder;
  Int ISA Process::Integrator;
  Comp ISA Control::BacklashInverse;

  FuncGen.x AT Cont.yref;
  Backlash.y AT Proc.u;
  Proc.y AT Int.u;
  Int.y AT Cont.pos;
  Proc.y AT Cont.vel;
  Cont.u AT Comp.u;
  Comp.y AT Backlash.u;
END;
```

A.4. Control

BacklashInverse

```
BacklashInverse ISA Base::Model WITH
  Init ISA Event;
  Sample ISA Event;

  h ISA Parameter WITH default := 0.005; END;

  b ISA Base::SimpleInput;
  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;

  x TYPE DISCRETE Real;
  uold TYPE DISCRETE Real;

  WHEN Init or Sample DO
    new(uold):=u;
    new(x) := if u - uold > 0 then u + b
              else if u - uold < 0 then u - b
              else x;
    schedule(Sample, h);
  END;

  y := x;
END;
```

Controller

```
Controller ISA Base::Model WITH
  yref ISA Base::SimpleInput;
  y ISA Base::SimpleInput;
  u ISA Base::SimpleOutput;
END;
```

Filter

```
Filter ISA Base::Model WITH
  Init ISA Base::Event;
  Sample ISA Base::EventInput;

  State ISA Base::Parameter WITH
    value TYPE (On, Off);
    default TYPE (On, Off) := 'On;
  END;
  A ISA Std::VectorPar WITH n := degA + 1; END;
  B ISA Std::VectorPar WITH n := degB + 1; END;

  u ISA Base::SimpleInput;
  y ISA Base::DiscreteTerminal;
```

```
  degB TYPE Integer := 1;
  degA TYPE Integer := 1;
  uv TYPE DISCRETE Column[degB + 1];
  yv TYPE DISCRETE Column[degA];
  uvtmp TYPE DISCRETE Column[degB + 2];
  yvtmp TYPE DISCRETE Column[degA + 1];

  WHEN Sample or Init DO
    new(uvtmp) := [u; uv];
    new(uv) := new(uvtmp[1..degB + 1]);
    new(y) := if State == 'On then trans(B)*new(uv) -
                                        trans(A[2..degA + 1])*yv
              else u;
    new(yvtmp) := [new(y); yv];
    new(yv) := new(yvtmp[1..degA]);
  END;
END;
```

ISTR - Indirect Self Tuning Regulator

```
ISTR ISA Control::Controller WITH
  Filters ISA Base::Parameter WITH
    value TYPE (On, Off);
    default TYPE (On, Off) := 'Off;
  END;
  FilterA ISA Std::VectorPar WITH n := FdegA + 1; END;
  FilterB ISA Std::VectorPar WITH n := FdegB + 1; END;

  degA TYPE Integer := 2;
  degB TYPE Integer := 1;
  Integral TYPE Integer := 0;
  FdegA TYPE Integer := 2;
  FdegB TYPE Integer := 1;

  Controller ISA Control::RST WITH
    degR := Design.degR;
    degS := Design.degS;
    degT := Design.degT;
  END;
  Design ISA Adaptive::Design WITH
    Integral := outer::Integral;
    degA := outer::degA;
    degB := outer::degB;
  END;
  Estimator ISA Adaptive::PE WITH
    degA := outer::degA;
    degB := outer::degB;
  END;
  Filter1 ISA Control::Filter WITH
    State := Filters;
    degA := FdegA;
    degB := FdegB;
```

```
    A := FilterA;
    B := FilterB;
  END;
  Filter2 ISA Control::Filter WITH
    State := Filters;
    degA := FdegA;
    degB := FdegB;
    A := FilterA;
    B := FilterB;
  END;

  y AT Controller.y;
  yref AT Controller.yref;
  Design.RegParOut AT Controller.RegParIn;
  Estimator.ModParOut AT Design.EstModParIn;
  Estimator.NewModPar AT Design.NewModPar;
  Design.NewRegPar AT Controller.NewRegPar;
  y AT Filter1.u;
  Filter1.y AT Estimator.y;
  Controller.u AT u;
  Controller.u AT Filter2.u;
  Filter2.y AT Estimator.u;
  Controller.SampleNext AT Estimator.Sample;
  Controller.SampleNext AT Filter2.Sample;
  Controller.SampleNext AT Filter1.Sample;
END;
```

ISTRBC - Indirect Self Tuning Regulator with Backlash Compensation

```
ISTRBC ISA Control::Controller WITH
  degA TYPE Integer := 1;
  degB TYPE Integer := 1;
  Integral TYPE Integer := 1;

  Estimator ISA Adaptive::PBE WITH
    degA := outer::degA;
    degB := outer::degB;
  END;
  Design ISA Adaptive::Design WITH
    Integral := outer::Integral;
    degA := outer::degA;
    degB := outer::degB;
  END;
  Controller ISA Control::RSTBC WITH
    degR := Design.degR;
    degS := Design.degS;
    degT := Design.degT;
  END;
  BackPos ISA Base::SimpleInput;

  y AT Estimator.y;
```

```
  BackPos AT Estimator.BackPos;
  Estimator.NewModPar AT Design.NewModPar;
  Estimator.ModParOut AT Design.EstModParIn;
  Estimator.bl AT Controller.b;
  Design.RegParOut AT Controller.RegParIn;
  Design.NewRegPar AT Controller.NewRegPar;
  y AT Controller.y;
  yref AT Controller.yref;
  Controller.SampleNext AT Estimator.Sample;
  Controller.u AT Estimator.u;
  Controller.u AT u;
END;
```

PhaseLead

```
PhaseLead ISA Base::Model WITH
  N ISA Parameter WITH default := 2; END;
  b ISA Parameter WITH default := 1; END;

  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;

  tmp TYPE Real;
  tmp'=b*N*(u-tmp);
  y = N*(u-tmp)+tmp;
END;
```

RST - RST-controller

```
RST ISA Control::Controller WITH
  Init ISA Base::Event;
  Sample ISA Base::Event;

  h ISA Base::Parameter WITH default := 1; END;

  degR TYPE Integer := 1;
  degS TYPE Integer := 1;
  degT TYPE Integer := 1;

  SampleNext ISA Base::EventOutput;
  NewRegPar ISA Base::EventInput;
  RegParIn ISA Adaptive::RegParTerm WITH
    nr := degR + 1;
    ns := degS + 1;
    nt := degT + 1;
    R.causality := 'input;
    s.causality := 'input;
    T.causality := 'input;
  END;

  yv TYPE DISCRETE Column[degS + 1];
  yrv TYPE DISCRETE Column[degT + 1];
  uv TYPE DISCRETE Column[degR + 1];
```

```
  yvtmp TYPE DISCRETE Column[degS + 2];
  yrvtmp TYPE DISCRETE Column[degT + 2];
  uvtmp TYPE DISCRETE Column[degR + 2];
  utmp TYPE DISCRETE Real;
  R TYPE DISCRETE Column[degR + 2];
  s TYPE DISCRETE Column[degS + 1];
  T TYPE DISCRETE Column[degT + 1];

  WHEN NewRegPar DO
    new(R) := [RegParIn.R; 0];
    new(s) := RegParIn.s;
    new(T) := RegParIn.T;
  END;

  WHEN Sample or Init DO
    new(yrvtmp[1..degT + 2]) := [yref; yrv];
    new(yvtmp[1..degS + 2]) := [y; yv];
    new(yrv) := new(yrvtmp[1..degT + 1]);
    new(yv) := new(yvtmp[1..degS + 1]);
    new(utmp) := - trans(s)*new(yv) + trans(T)*new(yrv) -
                        trans(R[2..degR + 2])*uv;
    new(uvtmp) := [new(utmp); uv];
    new(uv) := new(uvtmp[1..degR + 1]);
    schedule(SampleNext, 0);
    schedule(Sample, h);
  END;

  u := utmp;
END;
```

RSTBC - RST-controller with Backlash Compensation

```
RSTBC ISA Control::Controller WITH
  Init ISA Base::Event;
  Sample ISA Base::Event;

  h ISA Base::Parameter WITH default := 1; END;
  Comp ISA Base::Parameter WITH
    value TYPE (On, Off);
    default TYPE (On, Off) := 'On;
  END;

  degR TYPE Integer := 1;
  degS TYPE Integer := 1;
  degT TYPE Integer := 1;

  SampleNext ISA Base::EventOutput;
  NewRegPar ISA Base::EventInput;
  b ISA Base::SimpleInput;
  RegParIn ISA Adaptive::RegParTerm WITH
    nr := degR + 1;
    ns := degS + 1;
```

```
    nt := degT + 1;
    R.causality := 'input;
    s.causality := 'input;
    T.causality := 'input;
  END;

  yv TYPE DISCRETE Column[degS + 1];
  yrv TYPE DISCRETE Column[degT + 1];
  uv TYPE DISCRETE Column[degR + 1];
  yvtmp TYPE DISCRETE Column[degS + 2];
  yrvtmp TYPE DISCRETE Column[degT + 2];
  uvtmp TYPE DISCRETE Column[degR + 2];
  utmp TYPE DISCRETE Real;
  utmp2 TYPE DISCRETE Real;
  R TYPE DISCRETE Column[degR + 2];
  s TYPE DISCRETE Column[degS + 1];
  T TYPE DISCRETE Column[degT + 1];

  WHEN NewRegPar DO
    new(R) := [RegParIn.R; 0];
    new(s) := RegParIn.s;
    new(T) := RegParIn.T;
  END;

  WHEN Sample or Init DO
    new(yrvtmp[1..degT + 2]) := [yref; yrv];
    new(yvtmp[1..degS + 2]) := [y; yv];
    new(yrv) := new(yrvtmp[1..degT + 1]);
    new(yv) := new(yvtmp[1..degS + 1]);
    new(utmp) := - trans(s)*new(yv) + trans(T)*new(yrv) -
                   trans(R[2..degR + 2])*uv;
    new(uvtmp) := [new(utmp); uv];
    new(uv) := new(uvtmp[1..degR + 1]);
    new(utmp2) := if Comp == 'Off then utmp
                  else if new(utmp)-utmp > 0 then new(utmp) + b
                  else if new(utmp)-utmp < 0 then new(utmp) - b
                  else utmp2;
    schedule(SampleNext, 0);
    schedule(Sample, h);
  END;

  u := utmp2;
END;
```

State - State feedback controller

```
State ISA Base::Model WITH
  Kr ISA Base::Parameter WITH default := 1; END;
  Kw ISA Base::Parameter WITH default := 1; END;
  Kt ISA Base::Parameter WITH default := 1; END;

  yref ISA SimpleInput;
```

```
  vel ISA SimpleInput;
  pos ISA SimpleInput;
  u ISA SimpleOutput;

  u := Kr*yref - Kw*vel - Kt*pos;
END;
```

STBC - Self Tuning Backlash Compensation

```
STBC ISA Control::Controller WITH
  R ISA Std::VectorPar WITH n := degR + 1; END;
  S ISA Std::VectorPar WITH n := degS + 1; END;
  T ISA Std::VectorPar WITH n := degT + 1; END;

  degA TYPE Integer := 1;
  degB TYPE Integer := 0;
  degR TYPE Integer := 1;
  degS TYPE Integer := 1;
  degT TYPE Integer := 1;

  Estimator ISA Adaptive::BE WITH
    degA := outer::degA;
    degB := outer::degB;
  END;
  Controller ISA Control::RSTBC WITH
    degR := outer::degR;
    degS := outer::degS;
    degT := outer::degT;
    WHEN Init DO
      new(RegParIn.R) := outer::R;
      new(RegParIn.S) := outer::S;
      new(RegParIN.T) := outer::T;
      schedule(NewRegPar, 0);
    END;
  END;
  BackPos ISA Base::SimpleInput;

  y AT Estimator.y;
  y AT Controller.y;
  yref AT Controller.yref;
  Controller.SampleNext AT Estimator.Sample;
  Controller.u AT u;
  Controller.u AT Estimator.u;
  BackPos AT Estimator.BackPos;
  Estimator.bl AT Controller.b;
END;
```

A.5. Process

FirstOrder

```
FirstOrder ISA Base::Model WITH
  a ISA Parameter WITH default := 1; END;
  k ISA Parameter WITH default := 1; END;

  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;

  y' = a*(k*u - y);
END;
```

Integrator

```
Integrator ISA Base::Model WITH
  k ISA Parameter WITH default := 1; END;

  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;

  y' = k*u;
END;
```

SecondOrder

```
SecondOrder ISA Base::Model WITH
  w ISA Parameter WITH default := 1; END;
  z ISA Parameter WITH default := 0.7; END;
  k ISA Parameter WITH default := 1; END;

  u ISA Base::SimpleInput;
  y ISA Base::SimpleOutput;

  y'' + 2*w*z*y' + w*w*y = k*w*w*u;
END;
```

A.6. Signal

FunctionGenerator

```
FunctionGenerator ISA Model WITH
  Init ISA Event;
  SwitchOn ISA Event;
  SwitchOff ISA Event;

  Func ISA Parameter WITH
    value TYPE (sawtooth, sine, square, triangle);
    default TYPE (sawtooth, sine, square, triangle) := 'sine;
```

```
  END;
  Freq ISA Parameter WITH default := 1.0; END;
  Ampl ISA Parameter WITH default := 1.0; END;
  DutyCycle ISA Parameter WITH default := 0.5; END;
  Offset ISA Parameter;

  x ISA SimpleOutput;

  square TYPE DISCRETE Real;
  triangleDer TYPE DISCRETE Real;
  triangle TYPE Real;
  switchOffTime TYPE Real := DutyCycle/Freq;
  triangle' = triangleDer;

  WHEN Init CAUSE SwitchOn;

  WHEN SwitchOn DO
    schedule(SwitchOff, switchOffTime);
    schedule(SwitchOn, 1/Freq);
    new(square) := Ampl + Offset;
    new(triangle) := 0.0;
    new(triangleDer) := if Func == 'triangle then Ampl*Freq/
                                                    DutyCycle
                        else Ampl*Freq;
  END;

  WHEN SwitchOff DO
    new(square) := Offset;
    new(triangleDer) := if Func == 'triangle then -Ampl*Freq/
                                              (1.0 - DutyCycle)
                        else triangleDer;
  END;

  x := if Func == 'sine then Ampl*sin(Freq*6.28319*Base::time) +
                                                        Offset
       else if Func == 'square then square
       else triangle + Offset;
END;
```