CODEN:LUTFD2/(TFRT-5252)/0-075/(1981)

OPTIMAL INSTÄLLNING AV PID-REGULATORER MED HJÄLP AV FÖRLUSTFUNKTIONSOPTIMERING

CLAS EMANUELSSON

GÖRAN THEANDER

INSTITUTIONEN FÖR REGLERTEKNIK

LUNDS TEKNISKA HÖGSKOLA

AUGUSTI 1981

LUND INSTITUTE OF TECHNOLOGY
DEPARTMENT OF AUTOMATIC CONTROL
Box 725
S 220 07 Lund 7 Sweden

Document name
MASTER THESIS

Date of issue
August 1981

Document number
CODEN:LUTFD2/(TFRT-5252)/0-075/(1981)

Author(s)
Clas Emanuelsson
Göran Theander

Supervisor
Björn Wittenmark

Sponsoring organization

Title and subtitle

Optimal inställning av PID-regulatorer med hjälp av förlustfunktionsoptimering
(Optimal tuning of PID-controllers using lossfunction minimization)

Abstract

This report treats the adjustment of the parameters of an analogous PID-controller using a search method which optimizes a loss-function. A process model is updated periodically by means of a suitable identification algorithm. The loss-function is minimized by using a search algorithm. This is done by comparing the values of the loss-function for different setting of the parameters forming a vertex of a geometric structure in the function space and replacing the vertex with the highest value with a new calculated vertex. When acceptable controller parameters are obtained these can be automatically or manually transmitted to the real controller.

The system has been simulated and has been found to work satisfactorily especially when using reference model to calculate the loss-function.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
Swedish

Number of pages
75

Recipient's notes

Security classification

OPTIMAL INSTÄLLNING AV PID-REGULATORER MED HJÄLP AV FÖRLUSTFUNKTIONSOPTIMERING.

Clas Emanuelsson
Göran Theander

Examensarbete vid
Institutionen för Reglerteknik
Lunds Tekniska Högskola

1981

Handledare :
Björn Wittenmark
Sten Bergman

INNEHÅLLSFÖRTECKNING

1.PROBLEMSTÄLLNING OCH SAMMANFATTNING.

1.Problemställning.

2.Sammanfattning.

2.INSTÄLLNING AV PID-REGULATORER.

1.PID-regulatorn.

2.Konventionella inställningsmetoder.

3.Förlustfunktionsmetoder.

4.Sammanfattning och jämförelser.

3.OPTIMERING.

1.Optimering-allmänt.

2.Optimeringsmetoder.

3.Nelder-Mead metoden.

4.SIMULERINGSSYSTEMET.

1.Allmänt om SIMNON.

2.Systemets utformning.

3.Söksystemets SIMNONkod.

5.SIMULERINGAR.

1.Testprocesser.

2.Simuleringskommando.

3.Funktionsminimering.

4.Simuleringsresultat.

6.SLUTSATSER.

1. PROBLEMSTÄLLNING OCH SAMMANFATTNING

1.1 PROBLEMSTÄLLNINGEN

Denna rapport behandlar inställningen av en analog PID-regulator genom användning av ett söksystem som optimerar en förlustfunktion som bildas ur stegsvaret. Processmodellen i söksystemet kan tänkas uppdaterad med jämna tidsintervall m.h.a. någon lämplig identifieringsalgoritm. Förlustfunktionen minimeras med hjälp av en optimeringsalgoritm. När godtagbara regulatorparameterar erhållits, kan dessa överföras automatiskt eller manuellt till den verkliga regulatorn.

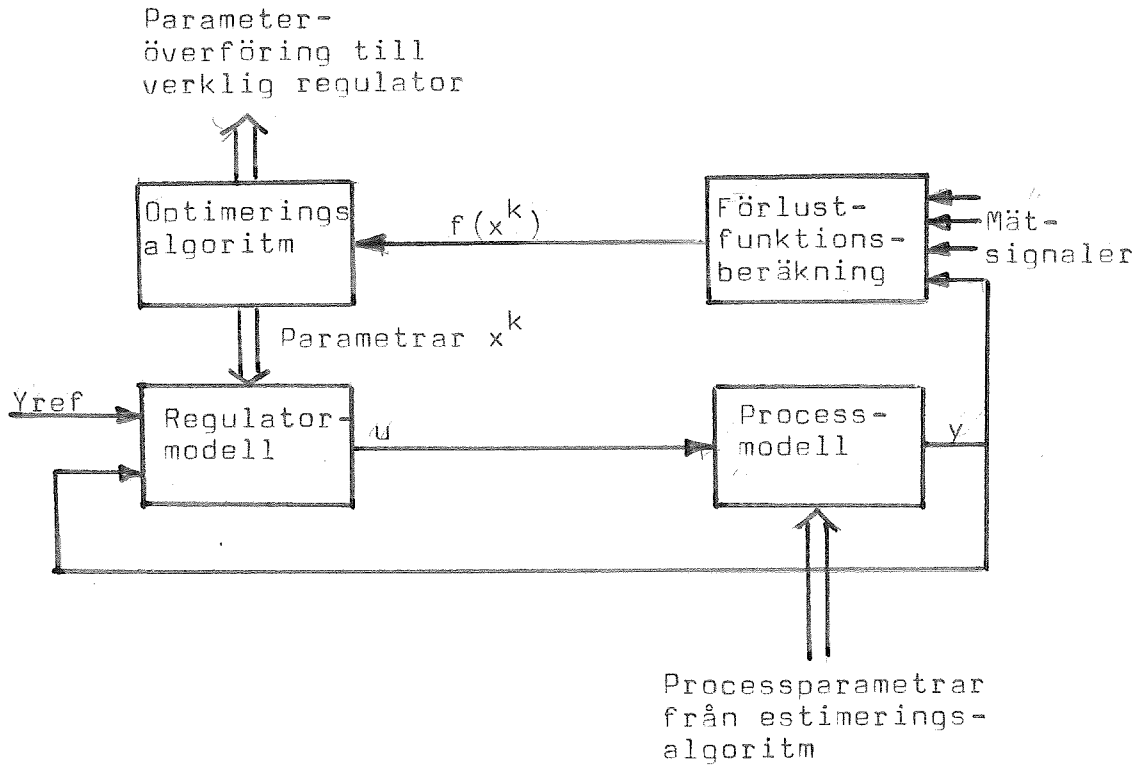

Fig. 1.1 Söksystemet

Systemet har simulerats på det vid reglerinstitutionen vid LTH framtagna simuleringsspråket SIMNON. Rapporten är en fortsättning och vidareutveckling av ett tidigare examensarbete (Paul Kongstad: Automatisk inställning av PID-regulatorer baserad på optimering och identifiering). Huvuddelen av rapporten har lagts vid optimeringen av regulatorparametrarna. Processerna som används får förutsättas vara styr- och observerbara samt kunna approximeras med en högst 5:e ordningens linjär process, som kan innehålla en ren tidsfördröjning. Dessutom antas att systemet har en insignal och en utsignal.

1.2 SAMMANFATTNING

Problemet med inställning av PID-regulatorer har studerats med hjälp av en simuleringsmodell. En jämförelse mellan olika inställningsmetoder visar, att förlustfunktionsmetoder är de bästa vid inställning med hjälp av dator.

Vid optimering av förlustfunktioner har det framkommit, att Fletchers metod i programpaketet OPTA (utvecklat på Reglerinstitutionen, LTH) ej fungerar tillfredsställande i detta sammanhang. Därför har en annan optimeringsalgoritm, Nelder-Meadmetoden, testats och funnits fungera betydligt bättre.

2. INSTÄLLNING AV PID-REGULATORN

2.1 PID-REGULATORN

Så gott som alla industrier kräver idag någon form av automatisering. Detta har medfört att reglering kommit att spela en allt större roll inom olika industriella processer. I princip ser ett slutet regulatorsystem ut som figur 2.1 visar.

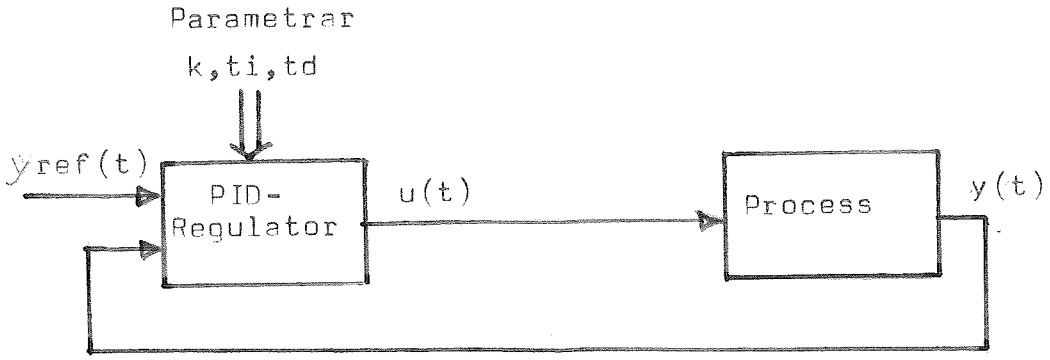

Fig. 2.1 Slutet reglersystem

Beteckningen yref är referenssignalen, y motsvarar utsignalen från processen och u är styrsignalen in till processen. Mellan dessa parametrar gäller följande samband:

$$(1)\quad u(t)=k*(yref(t)-y(t))+td*dy/dt+1/ti*\int_0^t (yref(s)-y(s))ds$$

Yref(t)-y(t) kallas reglerfelet och brukar betecknas med e(t). Styrsignalen består således av tre termer:

1.Proportionaltermen (P) som är proportionell mot felet.

2.Derivatatermen (D) som är proportionell mot utsignalens derivata.

3.Integraltermen (I) som är proportionell mot felets tidsintegral.

Vidare kallas k förstärkningsfaktorn och faktorerna ti och td integraltid resp. derivatatid, och är inställbara parametrar i regulatorn. Om man tar bort olika termer ur (1) får man olika specialfall av PID-regulatorn. Om

exempelvis td sätts lika med 0 erhålles en PI-regulator, vilken är den mest förekommande ute i indusrin. Vidare finns PD-, P- och I-regulatorer.

Genom att variera k, ti och td kan man ändra det slutna systemets beteende. Förstärkningsparametern k påverkar stationära fel, stabilitet och bandbredd. En ökning av förstärkningen medför en minskning av de stationära felen, försämring av stabiliteten, ökning av bandbredden och därmed snabbare system, men också större känslighet för mätbrus. Enbart P-verkan kan anses tillräckligt om processens dynamik kan beskrivas som en integrator.

Genom införande av en D-parameter kan regleringen baseras på en prediktion av framtida reglerfel. Derivatatiden bör motsvara den tid det tar för en ändring av styrsignalen att bli märkbar i utsignalen. En ökning av derivatatiden förbättrar stabiliteten, ökar bandbredden och ger systemet större snabbhet, men överskrides ett visst kritiskt värde uppnås motsatta ändringar. Detta medför att man tvingats bygga in en automatisk begränsning av derivatatermen i många PID-regulatorer. Ytterligare en nackdel med derivatatermen är att processen blir känsligare för högfrekventa mätfel.

Om störningar av olika slag påverkar processen, kommer kvarstående fel att påverka utsignalen då regulatorn är av PD-typ. Införandet av I-termen motverkar detta genom att bilda reglerfelets medelvärde och kompensera styrsignalens nivå. Integraltiden bör vara av samma storleksordning som det slutna systemets periodtid. En minskning av integraltiden (d.v.s. ökning av integraltermen) medför att systemet blir långsammare och stabiliteten försämras. För att motverka en oscillativ utsignal vid kraftiga ändringar i referensvärdet begränsar man ofta integraldelens belopp. Integralverkan kopplas in först då reglerfelets absolutbelopp är mindre än ett visst värde. Detta kallas för "reset windup".

2.2 KONVENTIONELLA INSTÄLLNINGSMETODER

Det har under årens lopp utvecklats många olika inställningsmetoder för PID-regulatorer. I detta kapitel skall de viktigaste beskrivas. Det skall påpekas att dessa metoder endast ger approximativa värden på parametrarna. Man får sedan för hand själv optimera sin regulator.

2.2.1 Inställningskartor

Inställningskartor demonstrerar hur stegsvar eller reglerfel ser ut för en viss typ av process, när olika regulatorparametrar användes. Se figur 2.2. Ofta kan godtagbara parametrar erhållas genom att gå efter inställningskartor.

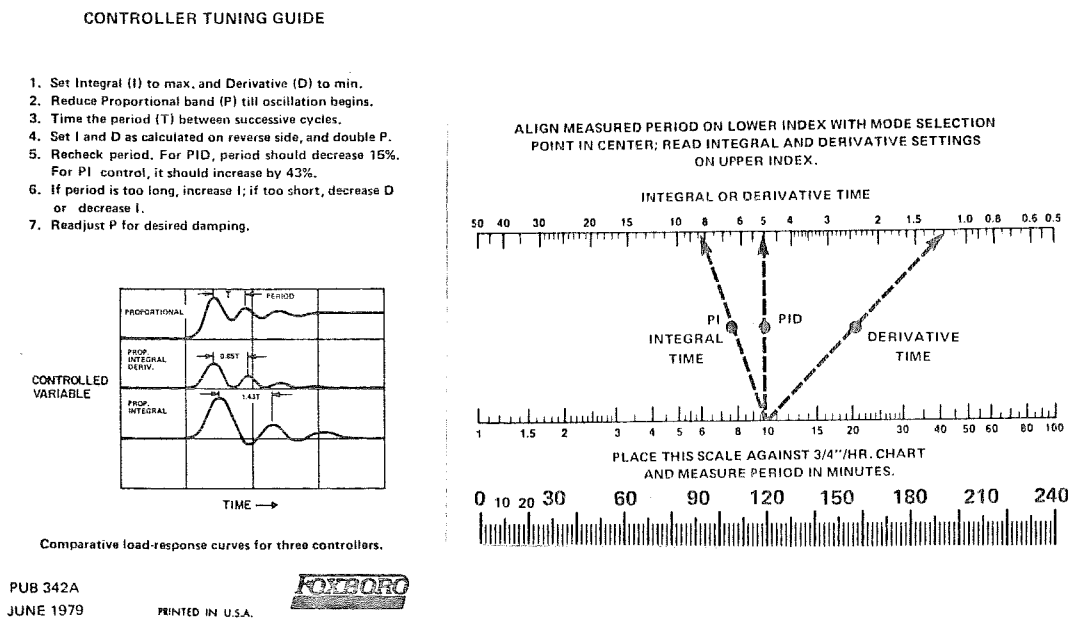

Fig. 2.2 Inställningskarta för PID-regulator

2.2.2 Ziegler-Nichols stegsvarsmetod

Ziegler-Nichols stegsvarsmetod baseras på vissa parametrar i den aktuella processens stegsvar. Se figur 2.3.

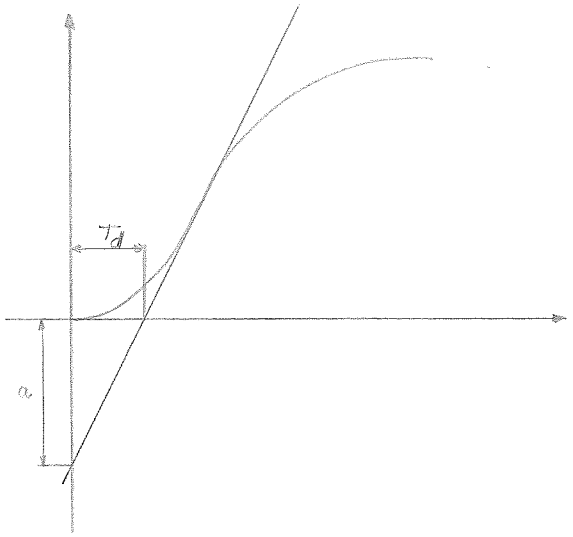

Fig. 2.3 Stegsvar för en viss process. Tangenten till stegsvarets maxlutning dras. a och T är sträckan från origo till tangentens skärning med x- resp. y-axeln.

Om man mäter upp a och T_d i figuren kan man sedan genom att gå in i nedanstående tabell ställa in k, ti och td.

Regulator	K	T_I	T_D
P	1/a		
PI	0.9/a	3 T_d	
PID	1.2/a	2 T_d	$T_d/2$

Störningar på systemet kan göra denna metod något osäker. Det skall också påpekas att stegsvarsmetoden bygger på det öppna systemets stegsvar, vilket är en allvarlig nackdel hos metoden då man sällan kan öppna ett system i drift.

2.2.3 Ziegler-Nichols självsvängningsmetod

Om processen utan risk kan försättas i självsvängning kan självsvängningsmetoden användas. Metoden går ut på att regulatorn ställs in som en proportionell regulator (ti=∞,td=0).

Genom att öka förstärkningen så att gränsen för instabilitet uppnås, avläsa förstärkning och svängningstid, kan lämpliga regulatorparametrar fås ur nedanstående tabell.

Regulator	K	T_I	T_D
P	0.5 K_c		
PI	0.45 K_c	0.83 T_c	
PID	0.6 K_c	0.5 T_c	0.12 T_c

K_c =förstärkningen vid självsvängning. T_c =svängningstiden vid självsvängning. Reglerna kan leda till dåligt dämpade system. Vid system av låg ordning är resultaten helt oanvändbara.

2.2.4 Cohen, Coons metod

Ziegler-Nichols metoder är mycket enkla, vilket medför att de ej passar i alla fall. Cohen och Coon har därför utvecklat en mer komplicerad inställningsprocedur i vilken flera parametrar ingår. Inställningen görs sedan m.h.a. en tabell på samma sätt som vid Ziegler-Nichols-metoden.

2.2.5 Grundtonens dämpning (QAM)

Alla signaler i ett linjärt system är av formen:

$$(2) \quad x(t)=\sum_{k=0}^{n} A_k * \exp(-\delta_k * t + i * \omega_k * t)$$

QAM-metoden förutsätter att det i reglertekniska sammanhang ofta är en svängningsmod som dominerar ($x_0(t)=A_0 * \exp(-\delta_0 * t + i * \omega_0 * t)$). Genom att mäta dämpningsfaktorn $d=\exp(-2*\pi*\delta_0/\omega_0)$ för den dominerande svängningen och justera regulatorparametrarna så att d=0.25 (se figur 2.4) fås en motsvarande amplitudmarginal Am på ungefär 2 och en relativ dämpning ζ=0.2.

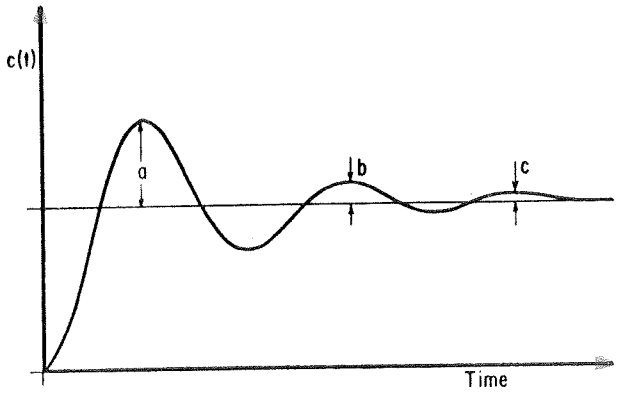

Fig. 2.4 Här gäller d=b/a=c/b=0.25

2.3 FÖRLUSTFUNKTIONSMETODER

Den stora fördelen med förlustfunktionsmetoder är att med ett enda tal kunna avgöra om regulatorn är optimal i någon mening. Förlustfunktionens form varierar från metod till metod, och därför blir jämförelser ofta subjektiva beroende på vad för kriterium man sätter i främsta hand. Förlustfunktionsmetoder går ut på att arean mellan ett önskat modellstegsvar och processens stegsvar beräknas på något bestämt sätt och minimeras med avseende på regulatorparametrarna som variabler. På samma sätt kan regulatorinställningen optimeras m.a.p. insignalreferens, "set point tuning" och störningsundertryckning, "disturbance tuning" i servo- resp. regulatorproblemet.

1. IE(integrated error).

Det integrerade felet definieras av

$$IE=\int_0^\infty e(t)*dt \tag{3}$$

Nackdelen med detta förhållandevis enkla kriterium är att om felet är periodiskt blir IE=0.

2. IAE(integrated absolute error).

Genom att ta absolutbeloppet på felet elimineras IE:s nackdel

$$IAE=\int_0^\infty abs(e(t))*dt \tag{4}$$

3. ISE(integrated squared error).

I detta kriterium viktas stora fel kraftigt och små fel obetydligt. Resultatet blir ofta ett snabbt svarande system med lång inställningstid. Överslängen blir måttlig.

$$ISE=\int_0^\infty e(t)^2*dt \tag{5}$$

4. ITAE(integrated timemultiplied absolute error).

Här tvingas processen uppnå stationaritet snabbt eftersom kriteriet straffar hårdare ju längre tid som går.

Överslängen i början blir något större än för ISE, men å andra sidan blir stegsvaret bättre dämpat.

(6) $$ITAE=\int_0^\infty t*abs(e(t))*dt$$

5. Det finns gott om varianter på de ovan uppräknade kriterierna. En variant är

(7) $$\int_0^\infty t*(abs(e(t))+abs(de/dt))*dt$$

En annan ser ut som

(8) $$\int_0^\infty f(t)*abs(e(t))*dt$$

där f(t) blir stor för vissa kritiska tidpunkter. En tredje variant tar även hänsyn till kraftiga ändringar i styrsignalen:

(9) $$F+\lambda*\int_0^\infty (du/dt)^2*dt$$

där F är någon av de tidigare nämnda kriterierna.

6. Genom att specificera ett visst område i y-t planet inom vilket man vill att stegsvaret skall hålla sig kan man åstadkomma en någorlunda bra regulator. Utanför detta område ökar förlustfunktionen kraftigt. Se figur 2.5.

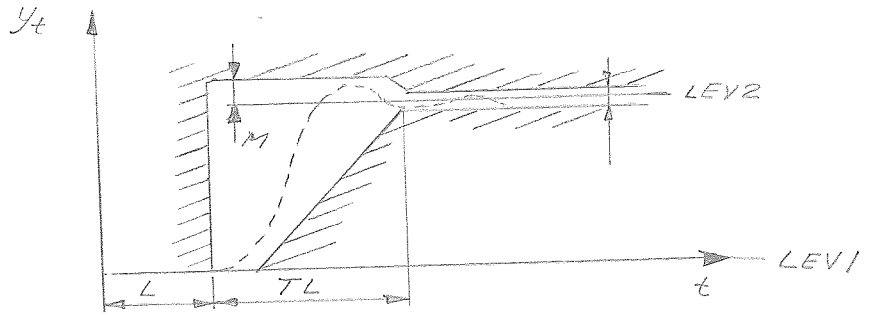

Fig. 2.5 Optimeringsområde i y-t planet

2.4 SAMMANFATTNING OCH JÄMFÖRELSER

Jämförelser har gjorts av de olika inställningsmetoderna av bl.a. Miller, Lopez, Smith och Murrill (se ref.). I sina försök använder de sig av en modell av första ordningen med dödtid, vilket är en god approximation till många industriella processer. Modellen har ekvationen:

$$(10)\quad \text{utsignal} = \text{insignal} * \frac{K * \exp(-\theta * s)}{(\tau * s + 1)}$$

där θ kallas dödtiden och τ är en tidskonstant. Figurerna 2.6, 2.7 och 2.8 indikerar att integralkriterierna identifierar de bästa metoderna. Man ser också att den bästa integralmetoden motsvarar sitt eget kriterium vilket understryker att val av kriterium måste ske helt subjektivt beroende på vad man vill uppnå. Vidare framgår att då θ/τ är mindre än 0.4 ger PI och PID-reglering i stort sett samma resultat.

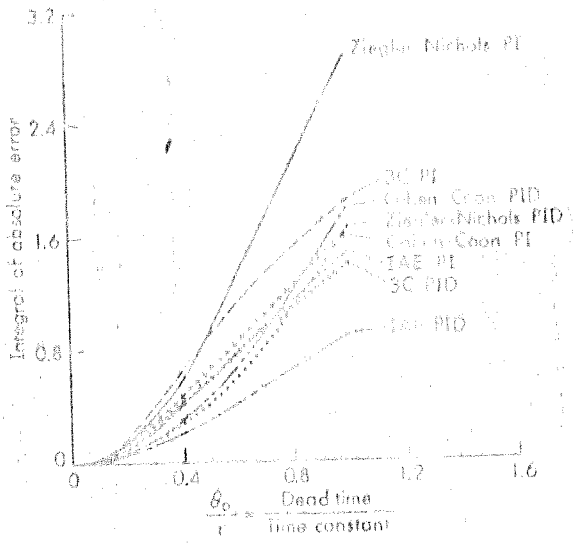

Fig. 2.6 Jämförelser av olika inställningsmetoder baserat på IAE-kriteriet. (Kurvan 3C står för QAM.)

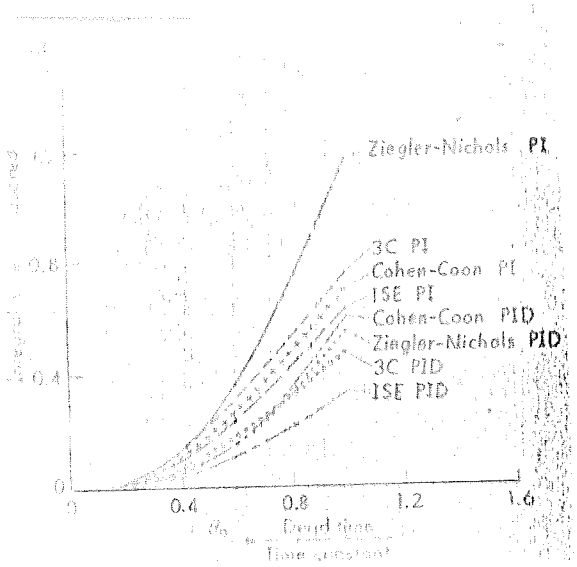

Fig. 2.7 Jämförelser av olika inställningsmetoder baserat på ISE-kriteriet

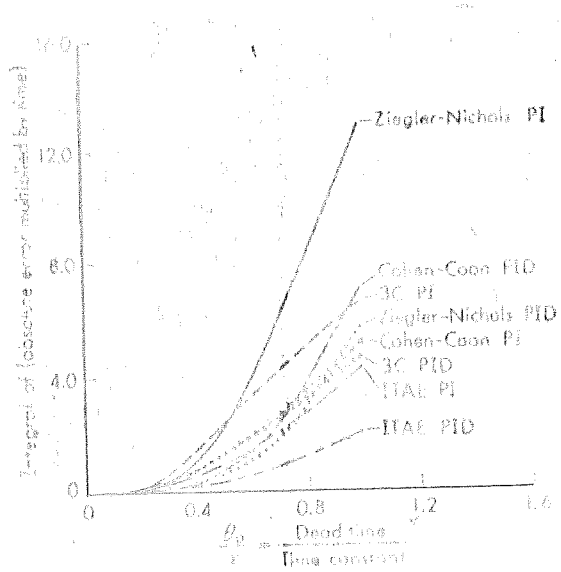

Fig. 2.8 Jämförelser av olika inställningsmetoder baserat på ITAE-kriteriet

3. OPTIMERING

3.1 OPTIMERING-ALLMÄNT

Optimering är ett matematiskt hjälpmedel som användes för att bilda underlag för att kunna fatta beslut. Datorn har gjort det möjligt att angripa problem med mycket stor komplexitet med matematisk optimeringsteknik. Typiska tillämpningsområden är prediktion, tidsplanering inom produktion, kvalitetskontroll, underhåll och reparation, processdesign och kapitalbudget.

De flesta problem av ovanstående karaktär har ett flertal lösningar. Syftet med optimering är att finna den "bästa" av många potentiella lösningar till ett givet problem med avseende på effektivitet eller utförandekriterium.

De olika allmänna optimeringsmetoderna kan klassificeras enligt:

1 Analytiska metoder

2 Numeriska metoder

3 Grafiska metoder

4 Experimentella metoder

Numeriska metoder använder tidigare information för att iterera sig fram till bästa lösningen. Dessa metoder tillgripes då de analytiska ej räcker till p.g.a. t.ex. ökande komplexitet. I de flesta fall kan problemet formuleras i en matematisk modell, vilken måste efterlikna de betydelsefulla dragen hos den verkliga processen för att sökandet efter en optimal lösning skall vara meningsfull.

Hos den matematiska modellen kan följande svårigheter uppstå:

1 Okänslighet mot förändringar av beslutsvariabel.

2 Optimeringskriteriet såväl som optimeringsvillkoren kan bli obegränsade i någon del av sökområdet. Detta gäller även gradientuttryck, om modellen innehåller polynom i nämnaren.

3 Om modellen har ojämn skalning av variablerna kan problem av typ 1 uppstå. Ojämnheter i skalningen kan vara svåra att upptäcka om produkter av variabler förekommer. En transformering till nya koordinater kan avhjälpa problemet.

Problem som kan relateras till den numeriska lösningstekniken är följande:

1 Om modellen innehåller olinjära uttryck existerar i regel mer än en extrempunkt. Detta problem förekommer ej hos linjära modeller. Om initialgissningen ligger alltför långt från extrempunkten kan därför optimeringen terminera vid en lokal extrempunkt istället för vid den globala.

2 Vid numeriska beräkningar introduceras fel. Avrundningsfel kan orsaka problem speciellt då metoder användes som approximerar derivator med differensekvationer. Stabilitetsproblem kan uppstå så att lösningen ej konvergerar med rimlig noggrannhet.

Ofta kan optimeringen av en verklig process ej uttryckas med en enkel matematisk formel (objektfunktion), utan sambanden mellan parametrar och/eller variabler måste uttryckas med likhets- eller olikhetsvillkor. Sambanden kan utgöras av fysikaliska lagar, gränser för fysikalisk realiserbarhet eller empiriska samband. Se figur 3.1.

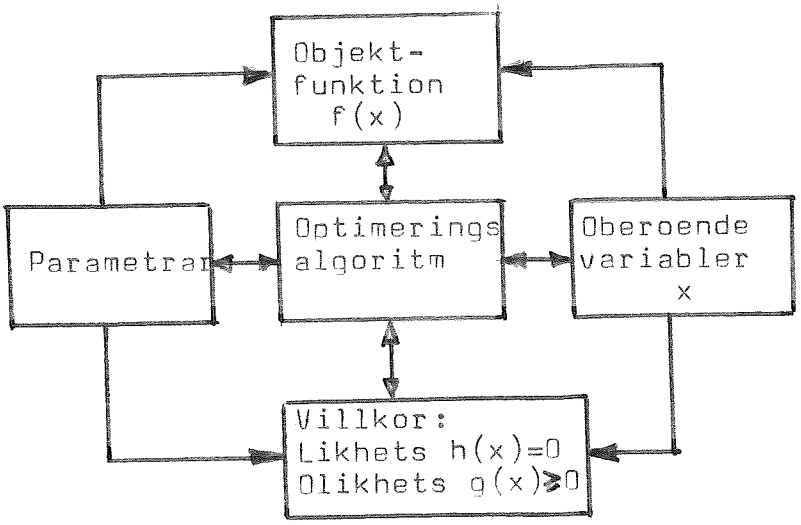

Fig. 3.1

Skillnaden mellan antalet oberoende beslutsvariabler n och antalet likhetsvillkor m definierar antalet frihetsgrader. Om n=m övergår optimeringen i lösning av

ett ekvationssystem $h_j(x)=0$ $j=1,..,n$. Om $n-m)0$ kan endast $n-m$ variabler justeras så objektfunktionen når sitt optimum. I vissa fall kan en variabel (ej påverkbar) elimineras genom substitution ur objektfunktionen.

I vissa fall kan ett villkorligt optimeringsproblem överföras till ett ovillkorligt genom införande av lämpligt vald straffunktion. Fördelen härmed är att en enklare optimeringsalgoritm kan användas. Ett allmänt uttryck för objektfunktionen med tillägg av straffunktion ger följande s.k. "penalty function":

$$(11)\quad P(x^k,\varrho^k)=f(x^k)+\sum_{i=1}^{m}\varrho_i^k H(h_i(x^k))+\sum_{i=m+1}^{p}\varrho_i^k G(g_i(x^k))$$

där $\varrho_i^k)0$ är viktfaktorer samt $G(g_i(x^k))$ och $H(h_i(x^k))$ funktionaler i $g_i(x^k)$ och $h_i(x^k)$ med vissa väldefinierade egenskaper så att effekten av villkoren minskas gradvis då sökningen pågår. Detta medför att värdet av $P(x^k,\varrho^k)$ konvergerar mot värdet av objektfunktionen $f(x^k)$ då sökningen närmar sig extrempunkten.

Möjliga felkällor i den matematiska optimeringsmodellen är t.ex. brist på giltiga parametrar i energi och massbalans vilka former villkor i modellen. Vidare kan förutsättningar om statisk modell vara dåligt uppfyllda p.g.a. dynamik i processen. Dessutom kan processvaiabler och parametrar ha ett stokastiskt uppförande snarare än ett deterministiskt.

Olika optimeringsalgoritmer kan klassificeras enligt följande:

1 Klassificering genom problemspecifikationer.

1.1 Villkorlig eller ovillkorlig objektfunktion
- a Ovillkorlig
- b Likhetsvillkor
- c Olikhetsvillkor
- d Såväl likhets- som olikhetsvillkor

1.2 Diskreta eller kontinuerliga variabler

2 Klassificering genom egenskaper i lösningstekniken.

2.1 Derivata eller derivatafri sökning

2.2 Analytiska eller numeriska derivator

2.3 Första eller andra ordningens derivator

2.4 Gradient eller gradientfri

2.5 Kortstegs- eller långstegsgradient

2.6 Samtidig iteration på alla variabler eller en i taget vid sökning

2.7 Deterministisk eller slumpmässig sökning

3 Klassificering genom använt programmeringsspråk.

3.2 OPTIMERINGSMETODER

3.2.1 Gradientmetoder

När man vill bestämma ett minimum är det rimligt att man av en användbar numerisk iterativ metod kräver att $f(x^{k+1}) < f(x^k)$. Geometriskt kan detta åskådliggöras i två dimensioner med hjälp av nivåkurvor där i varje punkt x på en kurva gäller att f(x)=konstant.

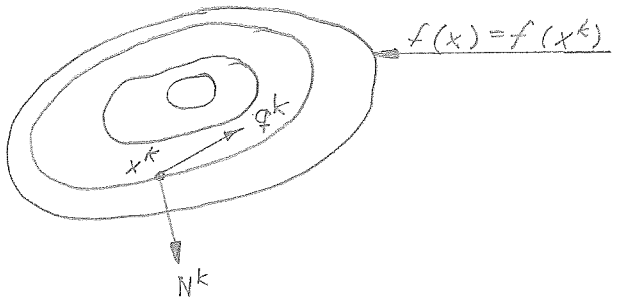

Fig. 3.2

Om vi sätter

(12) $x^{k+1}=x^k+q^k$

så visar figur 3.2 i exemplet att det finns vissa tillåtna riktningar $q^k/\|q^k\|$ och att för varje riktning steglängden $\|q^k\|$ är begränsad. Generellt består varje metod av två delar, nämligen:

1 Val av riktning

2 Val av steglängd

Man brukar allmänt skriva

(13) $x^{k+1}=x^k+\lambda_k q^k$

där q^k är riktningen och λ_k är ett tal som anger

steglängden. Varje metod för lösning av minimeringsproblemet kan karakteriseras av den maximala ordningen på de derivator av f som utnyttjas:

1 Metoder som inte utnyttjar derivator

2 Metoder som utnyttjar första-derivator

3 Metoder som utnyttjar andra-derivator

I regel fordrar metoder av typ 3 färre iterationer än metoder av typ 1 och 2, men de kräver ett betydligt större programmeringsarbete och exekveringstiden för varje iteration blir längre. Metoder av typ 2 och 3 kan endast användas om f är deriverbar.

3.2.2 Steepest-descent-metoden

En enkel och ofta använd men ej särskilt bra metod är steepest-descent-metoden. Den utnyttjar första-derivator. Riktningen q^k väljes som negativa normalriktningen $-N^k$ till nivåkurvan. N^k ges (icke normerad) av

$$(14)\quad N^k=\nabla f(x^k))=(\partial f/\partial x_1,\ldots,\partial f/\partial x_n)^T_{x=x^k}$$

Iterationsformeln kan då skrivas

$$(15)\quad x^{k+1}=x^k-\lambda_k N^k$$

där talet λ_k bestämmer steglängden. Ett sätt att välja λ_k är så att

$$(16)\quad f(x^{k+1})=f(x^k-\lambda_k N^k)$$

antar minimum som funktion av λ_k.

Steepest-descent-metoden ger ett minimum i den "gryta" startvektorn x^0 befinner sig i. Man kan således ha oturen att till slut hamna i ett lokalt minimum. Metoden fungerar bra om nivåkurvorna är nära cirkulära men ger annars långsam konvergens. Konvergenshastigheten kan vara mycket långsam om relativa skalningen hos variablerna är dålig, ty extrempunkterna ligger då i mycket utsträckta

dalar varmed stegriktningen är nästan ortogonal. Detta kan förbättras med hjälp av information om andraderivatan eller bättre skalning genom transformation av variabelrummet.

3.2.3 Newtons metod

Denna metod använder andraderivator för att beräkna sökriktningen

(17) $q^k = -H^{-1}(x^k)\nabla f(x^k)$ (*)

där H^{-1} är inversen av objektfunktionens hessianmatris. En stor nackdel är att den eller dess numeriska approximation måste vara positivt definit för konvergens. Fördelen med metoden är dess snabbhet i närheten av minimum p.g.a. att sökriktningen pekar mot minimum för objektfunktionen (i alla fall en stor komponent i sökriktningen). Långt ifrån minimum är steepest-descent klart bättre, om inte lokala utdragna dalar finnes. Metoder finns som m.h.a. transformationer tvingar H^{-1} att bli positivt definit. Svårigheter kan uppstå vid beräkning av objektfunktionens analytiska första- och andraderivator. Vidare kan de numeriska fel som introduceras då analytiska derivator ersättes med differensschema förstöra uppförandet av metoden i extrempunktens närhet.

3.2.4 Kvasi-Newton-metoder

Kvasi-Newton-metoder bygger på den ursprungliga Newtons metod modifierad på olika sätt. De utnyttjar formeln

(18) $x^{k+1} = x^k - J^{-1}(x^k)N(x^k)$

där J(x) är funktionens Jacobi-matris:

(*) H är en matris med objektfunktionen deriverad två gånger m.a.p. beslutsvariablerna.

$$(19)\quad J(x)=\begin{bmatrix} \partial f_1/\partial x_1 & \cdots & \partial f_1/\partial x_n \\ \vdots & & \vdots \\ \partial f_n/\partial x_1 & \cdots & \partial f_n/\partial x_n \end{bmatrix}$$

Den enklaste modifieringen är att utnyttja Kvasi-Newtons metod för att bestämma riktningen q^k och sedan en annan teknik för att bestämma steglängden. Man får då följande:

$$(20)\quad J(x^k)q^k=-N(x^k)$$

$$(21)\quad x^{k+1}=x^k+\lambda_k q^k$$

där talet λ_k kan väljas så att $\varphi(x^{k+1})$ minimeras som funktion av λ_k. Modifieringen förutsätter att $\det(J)\neq 0$.

3.2.5 Marquardts metod

Marquardts metod fungerar även om $\det(J)=0$ ty dess formler lyder:

$$(22)\quad (J(x^k)+\delta_k I)q^k=-N(x^k)$$

$$(23)\quad x^{k+1}=x^k+q^k$$

där δ_k är ett tal valt så att $J(x^k)+\delta_k I$ ej är singulär. Om δ_k är stort blir q_k nästan parallell med $-N(x^k)$ och man får samma riktning som i steepest-descent-metoden. Genom att starta med ett stort δ_0 och därefter utnyttja allt mindre δ_k har man en metod som har samma egenskaper som Newtons metod i närheten av minimipunkten.

Allmänt kan sägas att Marquardts metod är mera användbar än den stelbenta steepest-descent-metoden och Kvasi-Newtons metod.

3.2.6 Metoder för val av riktning

Generellt kan kravet för val av riktning formuleras som $(N^k)^T q^k < 0$ d.v.s. vinkeln mellan normalriktningen N^k och stegriktningen q^k skall vara större än 90 grader. Man kan bilda

(24) $q^k = -B^k N^k$

där B^k är en nxn-matris. Om man kräver att B^k är positivt definit(*) så gäller:

(25) $(N^k)^T q^k = -(N^k)^T B^k N^k < 0$

d.v.s. q^k är en tillåten riktning. Davidon-Fletcher-Powells metod beräknar B^K som uppfyller ovanstående villkor. Dess formel lyder:

(26) $$B^{k+1} = B^k + \frac{\lambda_k q^k (q^k)^T}{(q^k)^T p^k} - \frac{B^k p^k (q^k)^T B^k}{(q^k)^T B^k p^k}$$

där $p^k = N^{k+1} - N^k$. Fördelarna med Davidon-Fletcher-Powells metod är att den konvergerar bättre än steepest-descent-metoden, att den klarar sämre startvärden än Newtons metod och att beräkningarna är avsevärt enklare.

(*)Att en matris A är positivt definit innebär att $x^T Ax > 0$.

3.2.7 Metoder för val av steglängd

Oberoende av vilken metod som används för valet av riktning måste steglängden bestämmas så att

$$(27)\quad f(x^{k+1})=f(x^{k}+\lambda_k q^{k}) < f(x^{k}).$$

Två vanliga metoder att bestämma λ skall här nämnas:

1 λ väljs så att $f(x+\lambda q)$ antar minimum. Normalt används en teknik att beräkna $f(x+\lambda q)$ för en följd av λ-värden. Beräkningarna fortsätter tills φ åter börjar växa.

2 Kravet på att λ skall väljas så att $f(x+\lambda q)$ blir minimum släpps. Istället krävs att λ är så stort som möjligt och att f minskar så mycket som möjligt.

3.2.8 Sökmetoder

Dessa metoder skiljer sig från tidigare genom att de ej använder sig av objektfunktionens derivator eller approximationer för dessa. Dessa metoder konvergerar därför långsammare. För vissa funktioner kan det vara arbetsamt eller omöjligt att ta fram analytiska uttryck för derivator. I vissa fall kan derivatorna ersättas med differensekvationer men de numeriska fel som introduceras särskilt i närheten av extrempunkten kan försämra resultatet väsentligt. Sökmetoder kräver ej heller regularitet, kontinuitet eller existens av derivator hos objektfunktionen. Gradientmetoder och andraderivata-metoder kräver dessutom mer förberedande arbete innan problemet kan introduceras för datoralgoritmen.

3.3 NELDER-MEAD METODEN

Metoden konvergerar mot minimum för objektfunktionen genom att beräkna funktionsvärde i olika hörn av geometriska strukturer och succesivt ersätta det hörn med störst funktionsvärde med ett nytt hörn. Storleken och utseende av strukturen anpassas efterhand till funktionens nivåkurvor och minskas efterhand som metoden konvergerar mot minimum. Detta medför att steglängden minskas succesivt.

Då optimeringsalgoritmen NELME som användes i vårt simuleringssystem bygger på denna metod, beskrives denna mer i detalj i det följande.

Beskrivning av flexibel polyhedronmetoden:

Vi antar att antal oberoende variabler är två (n=2), men metoden har ingen begränsning av antalet obekanta. Den geometriska strukturens hörn definieras av tre vektorer med koordinater enligt:

$x_1=(0,0)$, $x_2=(d_1,d_2)$ samt $x_3=(d_2,d_1)$ där

$d_1=\frac{t}{n\sqrt{2}}(\sqrt{n+1}+n-1)$ samt $d_2=\frac{t}{n\sqrt{2}}(\sqrt{n+1}-1)$ där t=avståndet mellan två hörn.

Algoritmsteg:

Initialsteg: Beräkna startstrukturen x_1^0, x_2^0, x_3^0 enligt ovan samt funktionens värde

$f(x_1^0), f(x_2^0), f(x_3^0)$.

Steg 1: Beräkna värdena $f(x_h^k)=\max(f(x_1^k), f(x_2^k), f(x_3^k))$ och

$f(x_1^k)=\min(f(x_1^k), f(x_2^k), f(x_3^k))$

samt centrumpunkt definierad som $x_4=(x_{41}^k, x_{42}^k)$ med koordinater

$$x_{4j}^k=\frac{1}{n}\left(\sum_{i=1}^{n+1} x_{ij}^k - x_{hj}^k\right) \quad \text{för } j=1,2. \tag{28}$$

Steg 2 : Spegling

Spegla punkten x_h^k genom centrumpunkten med $\alpha>0$:

(29) $$x_5^k = x_4^k + \alpha * (x_4^k - x_h^k)$$

Steg 3 : Expansion

Om $f(x_5^k) \leq f(x_1^k)$, expandera vektorn $(x_5^k - x_4^k)$ genom att beräkna

(30) $$x_6^k = x_4^k + \gamma * (x_5^k - x_4^k) \qquad \text{där } \gamma > 1.$$

Om $f(x_6^k) < f(x_1^k)$ så ersätt x_h^k med x_6^k och k=k+1 samt fortsätt med steg 1. I annat fall ersätt x_h^k med x_5^k och k=k+1 samt fortsätt med steg 1.

Steg 4 : Kontraktion

Om $f(x_5^k) > f(x_i^k)$ för i=1,2,3 utom då i=h så förminska vektorn $(x_h^k - x_4^k)$ genom

(31) $$x_7^k = x_4^k + \beta * (x_h^k - x_4^k) \text{ med } 0 < \beta < 1.$$

Ersätt x_h^k med x_7^k och k=k+1 samt fortsätt med steg 1.

Steg 5 : Förminskning

Om $f(x_5^k) > f(x_h^k)$, minska alla vektorer $(x_i^k - x_1^k)$ för i=1,2,3 genom att beräkna $x_i^k = x_1^k + 0.5 * (x_i^k - x_1^k)$.

Ersätt k=k+1 och fortsätt med steg 1.

Ett lämpligt termineringsvillkor kan vara

(32) $$\left[\frac{1}{n+1} \sum_{i=1}^{n+1} \left[f(x_i^k) - f(x_4^k) \right]^2 \right]^{1/2} \leq \text{eps}$$

där eps = ett litet tal som motsvarar vald noggrannhet.

Parametrar: Genom lämpligt val av parametrarna α, β och γ kan man få den flexibla polyhedronstrukturen att anpassa sig effektivt till objektsfunktionens topologi t.ex.

expandera α vid långa sluttande plan och förminska β i närheten av extrempunkten. Expansionskoefficienten γ används för att förlänga steglängden om en spegling har hittat ett hörn med ett värde f(x) mindre än det minsta uppnått före speglingen. Ett lämpligt värde på parametern α är 1 som en kompromiss mellan snabb konvergens (=litet α-värde) och effektivare anpassning till en topologi med många speglingar i trånga kurviga dalar (= litet α-värde) å ena sidan och mindre antal funktionsberäkningar (= stort α-värde) å andra sidan. Några regler för val av parametrarna β och γ är svåra att ge. Parametern β har större inflytande på effektiviteten i sökningarna än parametern γ. Rekommenderade värden är $0.4 \leq \beta \leq 0.6$ och $2.8 \leq \gamma \leq 3.0$. Se vidare figur 3.3.

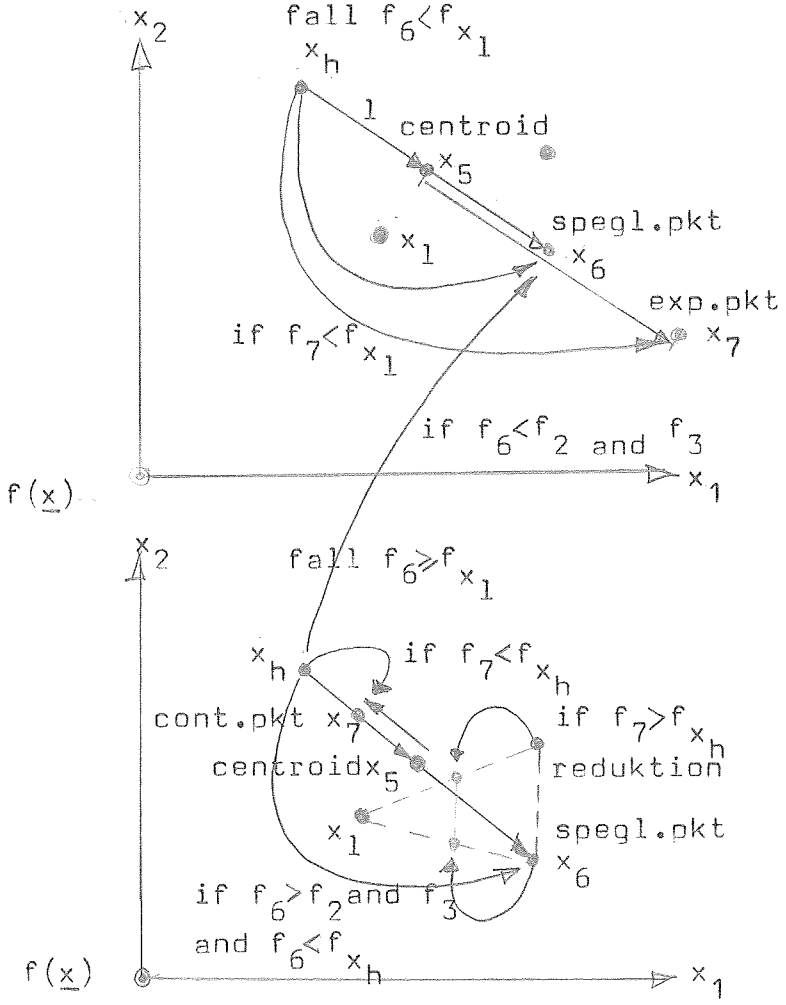

Fig. 3.3 Illustration av algoritmen

Storleken och orienteringen av startpolyhedronen har en ganska begränsad effekt på exekveringstiden. Dock bör flera startpunkter prövas för att få en uppfattning om extrempunkten är global eller lokal.

4. SIMULERINGSSYSTEMET

4.1 ALLMÄNT OM SIMNON

SIMNON är ett interaktivt simuleringsprogram. Programspråkets enkelhet och interaktiva kommando gör det lämpligt för snabba simuleringsstudier. Möjligheten att inkludera subsystem i FORTRAN gör att man även kan använda komplicerade modeller i simuleringssystemet. Användaren kan styra program m.h.a. kommando vilka även kan slås samman till en macrokropp för att minska terminalarbetet. Kompilatorn arbetar parallellt med en editor och möjliggör omedelbar ändring av felaktig rad. För att kunna simulera datorstyrda processer finns möjlighet att beskriva den fysikaliska processen m.h.a. differentialekvationer, och datorn m.h.a. differensekvationer. Notationen för detta är continuous time subsystem resp. discrete time subsystem. Beskrivningen av ett subsystem består av två delar:

1. Beräkning av tidsderivator med uppdatering av tillstånd.

2. Beräkning av utsignaler.

Signalutbytet mellan olika system beskrivs i ett s.k. connecting system. Systemekvationerna löses efter automatisk sekventiell sortering. Resultatet av en simulering kan plottas på grafisk skärm. Modellbeskrivningen matas normalt in m.h.a. SIMNONs editor och lagras i massminne.

4.2 SYSTEMETS UTFORMNING

Systemet som vi har simulerat på visas i figur 4.1. Jämför med figur 1.1

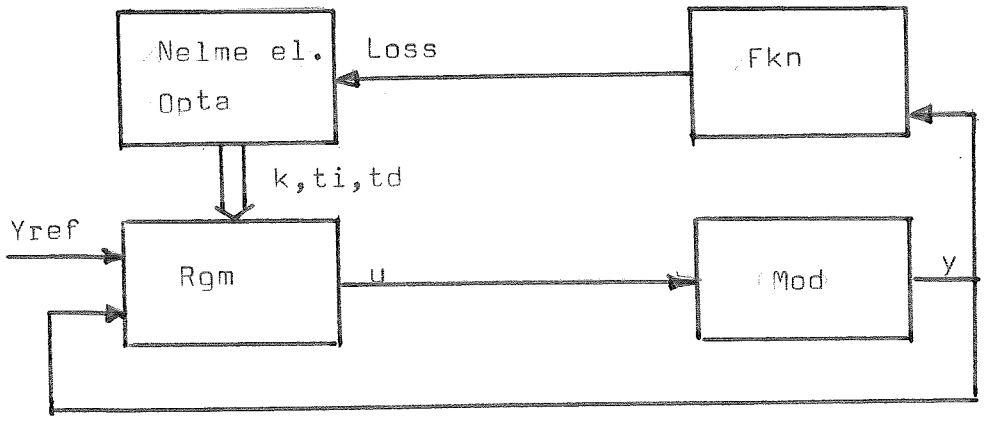

Fig. 4.1 Blockschema över subsystemen i modellen

Från början ges vissa initialvärden på regulatorparametrarna p, ti och td. Ett steg ges som värde på Yref. Subsystemet FKN jämför sedan detta steg med modellens stegsvar och beräknar m.h.a. en vald förlustfunktion ut ett specifikt värde på LOSS som mått på godheten av regulatorparametrarna. LOSS skickas sedan till optimeringsalgoritmen NELME. Denna beräknar nya och bättre värden på parametrarna som sedan ges till regulatorn RGM. På detta sätt fortsätter optimeringen till godtagbara värden på p, ti och td har hittats.

Genom att modifiera systemet kan man få felet e(t) att vara skillnaden mellan utsignalen från ett känt system med önskvärd dynamik, och utsignalen från modellen om insignalen är gemensam. Se figur 4.2.

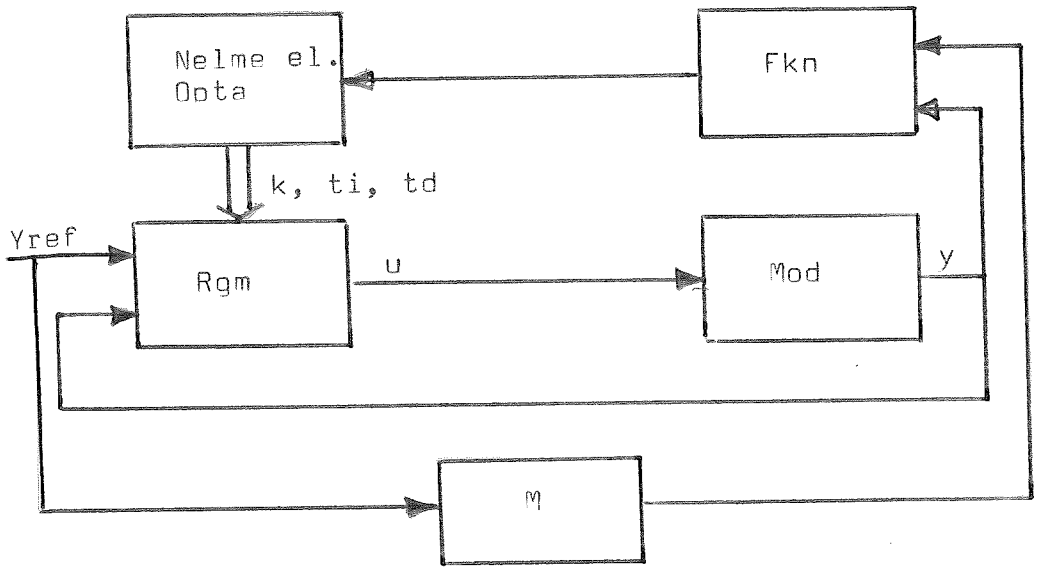

Fig. 4.2 Blockschema vid s.k. modellreferens

4.3 SÖKSYSTEMETS SIMNONKOD

Subsystem RGM

Söksystemets modell av en PID-regulator är uppbyggd som ett discrete system RGM, vilket approximerar den analoga regulatorn väl, om samplingstiden väljes kort t.ex. 0.04s. Modellen är en samplad form

$$(34) \qquad u_t = k*(e_t + DTM/ti*\sum_{k=0}^{t} e_k - td*(y_t - y_{t-1})/DTM)$$

av en idealiserad PID-regulator med formeln:

$$(35) \qquad u(t) = k*(e(t) + 1/ti*\int_0^t e(s)ds - td*dy/dt)$$

Då processignalen är brusig kan derivatatermens belopp begränsas till DMAX genom att sätta parametrarna SYSL och SN. SYSL är processens stegsvarslutning och SN en faktor som anger hur mycket snabbare derivationen får vara. Förslag SN:1-5. Eftersom parametrarna kan varieras utifrån är de ingångar till RGM.

```
DISCRETE SYSTEM RGM
INPUT YREF YP K TI TD
OUTPUT U
STATE I X
NEW NI NX
TIME T
TSAMP TS
INITIAL
DMAX=SYSL*SN*DTRG
OUTPUT
E=YREF-YP
DSL=-TD*(YP-X)/DTRG
D=SIGN(DSL)*MIN(ABS(DSL),DMAX)
U=IF MOD(T/DTM,PROVL))PROVL-.999 THEN 0 ELSE K*(E+I+D)
DYNAMICS
NI=IF MOD(T/DTM,PROVL))PROVL-.999 THEN 0 ELSE I+E/TI*DTRG
NX=IF MOD(T/DTM,PROVL))PROVL-.999 THEN 0 ELSE YP
TS=T+DTM
SYSL:3
SN:5
DTM:1
PROVL:50
DTRG:1
END
```

Subsystem MOD

Söksystemets modell av processen är uppbyggd som ett discrete system MOD. Processen är beskriven i extern tidsdiskret form. Den tänkes representerad genom sin styrbara kanoniska form och ett ordningstal av maximalt fem. Koefficienterna hålles konstanta i simuleringarna men kan i verkligheten uppdateras från en estimeringsalgoritm med jämna mellanrum. För närmare beskrivning av själva processerna hänvisas till kapitel 5.1. Då ett stegsvar simulerats nollställs tillståndet av systemet MT.

```
DISCRETE SYSTEM MOD
INPUT U MT
OUTPUT Y
STATE X1 X2 X3 X4 X5
NEW NX1 NX2 NX3 NX4 NX5
TIME T
TSAMP TS
OUTPUT
Y=B1*X1+B2*X2+B3*X3+B4*X4+B5*X5
DYNAMICS
NN=-A1*X1-A2*X2-A3*X3-A4*X4-A5*X5+U
NX1=IF MT<47 THEN NN ELSE 0
NX2=IF MT<47 THEN X1 ELSE 0
NX3=IF MT<47 THEN X2 ELSE 0
NX4=IF MT<47 THEN X3 ELSE 0
NX5=X4
TS=T+DTM
A1:-3.2
A2:3.84
A3:-2.048
A4:.4096
A5:0
B1:.03
B2:.06
B3:.005
B4:.03
B5:0
DTM:1
PROVL:50
END
```

Subsystem FKN

För beräkning av förlustfunktionen i ett antal punkter (PROVL) på ett simulerat stegsvar användes ett discrete system FKN. Med parametern NOM väljer man om inställningen skall baseras på modellreferens eller insignalreferens.

I första fallet beräknas felsignal e som:

(36) $e_t = y_t^M - y_t^{MOD}$ för NOM=1,

och i andra fallet som:

(37) $e_t = y_{ref} - y_t^{MOD}$ för NOM=0.

Med parametern SQ väljer man om förlustfunktionen skall beräknas enligt ISE-kriteiet eller ITAE-kriteriet.

I första fallet beräknas förlustfunktionen enligt:

(38) $\sum_{t=2}^{PROVL-1} e_t^2$ för SQ=1,

och i andra fallet som:

(39) $\sum_{t=2}^{PROVL-1} t*abs(e_t)$ för SQ=0.

Önskar man dessutom att lägga till straff på ändringar i styrsignalen till systemet MOD, sättes parametern LA)O (motsvarar λ i formel 9). Variabeln STRAF ger tillägg till förlustfunktionen för negativa värden på regulatorparametrarna och adderas automatiskt. FKN innehåller en variabel R, som räknar hur många samplingsintervall simuleringsstegsvaret har pågått. Själva förlustfunktionen summeras i variablen SUM, och just innan stegsvaret skall sluta laddas det uppnådda värdet över till varibeln LOSS, som håller värdet tillgängligt under nästa simuleringsstegsvar. Nollställning skötes dels av FKN själv och dels av systemet MT.

```
 DISCRETE SYSTEM FKN
INPUT U Y YNOM YREF MT
OUTPUT RR
TSAMP TS
TIME T
STATE SUM UOLD R LOSS
NEW NS NOLD NR NL
OUTPUT
RR=R
DYNAMICS
E=(IF NOM THEN YNOM ELSE YREF)-Y
DU=U-UOLD
STR=IF TI(.01 THEN 10*(.01-TI) ELSE 0
STRA=(IF K(.01 THEN 10*(.01-K) ELSE 0)+STR
STRAF=STRA+(IF TD(0 THEN -10*TD ELSE 0)
SNS=SUM+LA*DU*DU+(IF SQ THEN E*E ELSE (R-1)*ABS(E))+STRAF
NR=IF MT(47 THEN R+1 ELSE 0
NL=IF R)PROVL-2.999 AND R(PROVL THEN SNS ELSE LOSS
NS=IF R(.99 THEN 0 ELSE SNS
NOLD=IF R(.99 THEN 0 ELSE U
TS=T+DTM
LA:0
K:1
TI:1
TD:1
SQ:1
NOM:0
PROVL:50
DTM:1
END
```

Subsystem M

Vid modellreferens användes ett samplat andra ordningens system uppbyggt av ett discrete system M. Modellens överföringsfunktion definieras genom

$$G(s)=\frac{\omega^2}{s^2+2*\zeta*\omega*s+\omega^2} \qquad (40)$$

där ζ anger relativa dämpningen och ω systemets snabbhet.

Systemet M's insignal är samma som referenssignalen till det slutna systemets PID-regulator. Parametrarna Z och ONOLL representerar dämpningen ζ resp. systemets snabbhet ω. Genom val av dessa kan man få det slutna systemets stegsvar att konvergera mot olika önskade stegsvar. Samplingstiden DT är samma som för systemet MOD och nollställning vid nytt simuleringsstegsvar åstadkommes av systemet MT.

```
DISCRETE SYSTEM M
INPUT U MT
OUTPUT Y
TIME T
STATE S1 S2
NEW NS1 NS2
TSAMP TS
INITIAL
OM=ONOLL*SQRT(ABS(1-Z*Z))
A=EXP(-Z*ONOLL*DT)
B=IF Z)1 THEN .5*(EXP(OM*DT)+EXP(-OM*DT)) ELSE COS (OM*DT)
E=SIN (OM*DT)
D=Z*ONOLL/OM*(IF Z)1 THEN .5*(EXP(OM*DT)-EXP(-OM*DT)) ELSE E)
B1=1-A*(B+D)
B2=A*(A+D-B)
A1=-2*A*B
A2=A*A
OUTPUT
Y=S1
DYNAMICS
NS1=IF MT(47 THEN -A1*S1+S2+B1*U ELSE 0
NS2=IF MT(47 THEN -A2*S1+B2*U ELSE 0
TS=T+DT
Z:.2672
ONOLL:10
DT:.04
PROVL:50
END
```

Subsystem MT

För att nollställning av tillstånd i olika subsystem skall ske samtidigt utförs denna i en speciell nollställningsrutin uppbyggd som ett discrete system MT.

```
DISCRETE SYSTEM MT
OUTPUT MT
TIME T
TSAMP TS
OUTPUT
MT=MOD(T/DTN,PROVL)-1
DYNAMICS
TS=T+DTN
PROVL:50
DTN:.04
END
```

Connecting systems

För att åstadkomma sammankoppling mellan olika subsystem användes connecting system MKON2 eller MKON3 för regulatorn RGM kopplad som PI- resp. PID-regulator. Se vidare figur 4.1 och 4.2. För användning tillsammans med optimeringsrutinen NELME:

```
CONNECTING SYSTEM MKON2
TIME T
MT[FKN]=MT[MT]
MT[M]=MT[MT]
MT[MOD]=MT[MT]
REFER=IF RR[FKN]<1.999 OR RR[FKN]>PROVL-.999 THEN 0 ELSE NIV
YREF[RGM]=REFER
YP[RGM]=Y[MOD]
U[MOD]=U[RGM]
U[M]=REFER
YNOM[FKN]=Y[M]
YREF[FKN]=REFER
U[FKN]=U[RGM]
Y[FKN]=Y[MOD]
K[RGM]=PN1[NELME]
TI[RGM]=PN2[NELME]
LOSS[NELME]=LOSS[FKN]
TD[RGM]=0
NIV:    1
PROVL :50
END
```

```
 CONNECTING SYSTEM MKON3
TIME T
MT[FKN]=MT[MT]
MT[M]=MT[MT]
MT[MOD]=MT[MT]
REFER=IF RR[FKN]<1.999 OR RR[FKN]>PROVL-.999 THEN 0 ELSE NIV
YREF[RGM]=REFER
YP[RGM]=Y[MOD]
K[RGM]=PN1[NELME]
TI[RGM]=PN2[NELME]
TD[RGM]=PN3[NELME]
LOSS[NELME]=LOSS[FKN]
U[MOD]=U[RGM]
U[M]=REFER
YNOM[FKN]=Y[M]
YREF[FKN]=REFER
U[FKN]=U[RGM]
Y[FKN]=Y[MOD]
NIV:    1
PROVL :50
END
```

Motsvarande connecting systems MKON20 och MKON30 för användning tillsammans med optimeringsrutinen OPTA:

```
CONNECTING SYSTEM MKON20
TIME T
MT[FKN]=MT[MT]
MT[M]=MT[MT]
MT[MOD]=MT[MT]
REFER=IF RR[FKN]<1.999 OR RR[FKN]>PROVL-.999 THEN 0 ELSE NIV
YREF[RGM]=REFER
LOSS[OPTA]=LOSS[FKN]
YP[RGM]=Y[MOD]
U[MOD]=U[RGM]
U[M]=REFER
YNOM[FKN]=Y[M]
YREF[FKN]=REFER
U[FKN]=U[RGM]
Y[FKN]=Y[MOD]
K[RGM]=P1[OPTA]
TI[RGM]=P2[OPTA]
TD[RGM]=0
NIV:    1
PROVL :50
END
```

```
CONNECTING SYSTEM MKON30
TIME T
MT[FKN]=MT[MT]
MT[M]=MT[MT]
MT[MOD]=MT[MT]
REFER=IF RR[FKN]<1.999 OR RR[FKN]>PROVL-.999 THEN 0 ELSE NIV
YREF[RGM]=REFER
LOSS[OPTA]=LOSS[FKN]
YP[RGM]=Y[MOD]
K[RGM]=P1[OPTA]
TI[RGM]=P2[OPTA]
TD[RGM]=P3[OPTA]
U[MOD]=U[RGM]
U[M]=REFER
YNOM[FKN]=Y[M]
YREF[FKN]=REFER
U[FKN]=U[RGM]
Y[FKN]=Y[MOD]
NIV:    1
PROVL :50
END
```

Parameterfiler

Parameterfilerna MPAR2 och MPAR3 innehåller startvärden för tillstånd och olika parametrar för simulering av ett andra resp. tredje ordningens system. Dessa filer kan skapas, editeras och återkallas enligt tidigare beskrivning. För användning tillsammans med optimeringsrutinen OPTA:

```
[RGM] I:0.  X:0. SYSL:3. SN:5. DTRG:1.  DTM:.04 PROVL:50.
[MOD] X1:0. X2:0. X3:0. X4:0. X5:0. B1:.399577 B2:.146996
B3:0 B4:0  B5:0. A1:-.503215  A2:.049787 A3:0  A4:0 A5:0.
DTM:.04 PROVL:50. [M] S1:0.  S2:0. ONOLL:10. Z:.8 DT:0.04
PROVL:50. [FKN]  SUM:0. UOLD:0. R:-1. LOSS:0.  NOM:0 K:1.
TI:1. TD:1.  LA:0. SQ:1. PROVL:50. DTM:.04  [MT] DTN:0.04
PROVL:50.  [OPTA]  PI1:.5 PI2:5  XM1:1.  XM2:1.  DFN:-0.5
TINC:2  HH:.1  EPS:0  PRIN:0  EVMAX:10000.  CEQ:0.  C:1.
DELTA:10.E-3  RESET:0  DARK:0  MODE:1.  LPLOT:0  [MKON20]
PROVL:50. NIV:1.
```

```
[RGM] I:0.  X:0. SYSL:3. SN:5. DTRG:1.  DTM:.04 PROVL:50.
[MOD] X1:0. X2:0.  X3:0. X4:0. X5:0. B1:1  B2:0 B3:0 B4:0
B5:0.  A1:-.8  A2:.64  A3:-.512  A4:.4096  A5:0.  DTM:.04
PROVL:50. [M] S1:0. S2:0. ONOLL:10 Z:.8 DT:0.04 PROVL:50.
[FKN] SUM:0. UOLD:0. R:-1 LOSS:0.  NOM:1 K:1. TI:1. TD:1.
LA:0.  SQ:0  PROVL:50. DTM:.04  [MT]  DTN:0.04  PROVL:50.
[OPTA] PI1:.5  PI2:5 PI3:2.5  XM1:1 XM2:1  XM3:1 DFN:-0.5
TINC:2.  HH:.1  EPS:0  PRIN:0  EVMAX:10000.  CEQ:0.  C:1.
DELTA:10.E-3  RESET:0  DARK:0  MODE:1.  LPLOT:0  [MKON30]
PROVL:50. NIV:1.
```

Motsvarande parameterfiler MPAR20 och MPAR30 för användning tillsammans med optimeringsrutinen NELME:

```
[RGM] I:0.  X:0. SYSL:3. SN:5. DTRG:1.  DTM:.04 PROVL:50.
[MOD] X1:0  X2:0 X3:0. X4:0.  X5:0. B1:.399  B2:.147 B3:0
B4:0  B5:0  A1:-.503  A2:.0498 A3:0  A4:0  A5:0.  DTM:.04
PROVL:50.  [M]  S1:0.  S2:0.  ONOLL:10.  Z:.7  DT:0.04
PROVL:50. [FKN] SUM:0. UOLD:0.  R:-1 LOSS:0.  NOM:1 K:1.
TI:1. TD:0.  LA:0. SQ:0  PROVL:50. DTM:.04  [MT] DTN:0.04
PROVL:50.  [NELME]  X1:.5 X2:5  WX11:0.  WX12:0.  WX21:1.
WX22:1. WX31:10. WX32:3. WX41:3.  WX42:1. MODE:0. TINC:2.
VDIST:.1 BETA:-.5 GAMMA:-2. [MKON2] PROVL:50. NIV:1.
```

```
[RGM] I:0.  X:0. SYSL:3. SN:5. DTRG:1.  DTM:.04 PROVL:50.
[MOD]  X1:0 X2:0  X3:0 X4:0.  X5:0. B1:1  B2:0 B3:0  B4:0
B5:0.  A1:-.8  A2:.64  A3:-.512  A4:.4096  A5:0.  DTM:.04
PROVL:50.  [M]  S1:0.  S2:0.  ONOLL:10.  Z:.8  DT:0.04
PROVL:50. [FKN] SUM:0. UOLD:0.  R:-1 LOSS:0.  NOM:1 K:.5
TI:5 TD:2.5  LA:0. SQ:0  PROVL:50. DTM:.04  [MT] DTN:0.04
PROVL:50.  [NELME]  X1:.5  X2:5  X3:2.5  WX11:0.  WX12:0.
WX13:0. WX21:0.  WX22:0. WX23:0. WX31:0.  WX32:0. WX33:0.
WX41:0. WX42:0. WX43:0. MODE:0. TINC:2. VDIST:.1 BETA:-.5
GAMMA:-2.0 [MKON3] PROVL:50. NIV:1.
```

Subsystem NELME

För att åstadkomma en minimering av en förlustfunktion av två eller tre variabler (k, ti och td) m.h.a. Nelder-Mead-metoden användes ett discrete system NELME. Programmet är skrivet i FORTRAN och anpassat till simuleringsspråket SIMNON. För anpassningsregler hänvisas till SIMNON-manualen. Kommentarer är införda i koden för att förtydliga rutinens funktion. Nedan följer en förklaring till olika parametrars inverkan på rutinens uppförande (se även beskrivning av Nelder Mead-metoden i tidigare avsnitt). Defaultvärden inom parantes.

VDIST - avstånd mellan två hörn i startpolygonen då MODE=0. (0.1)

MODE - kontollerar beräkningsgången av startpolygonen. Värde antingen 0 eller 1. (0)

MODE=0 : Rutinen beräknar en startpolygon med vektorn X (given i parameterfilen) som koordinater för centroid och med VDIST som avstånd mellan två hörn. Dessa koordinater lagras i NPAR*(NPAR+1) första elementen av vektorn WX.

MODE=1 : Startpolygonens koordinater är givna i parameterfilen som de NPAR*(NPAR+1) första elementen i vektorn WX. Denna MODE kan användas för återstart.

ALFA, BETA och GAMMA - inverkan är beskriven tidigare i samband med beskrivning av Nelder Mead-metoden. Här motsvaras ALFA av α, BETA av β och GAMMA av γ. (1.0 -0.5 resp. -2.0)

TINC - samplingsintervall i NELME. Detta skall vara lika med tiden för ett simuleringsstegsvar i söksystemet. (2.0)

```
              SUBROUTINE NELME
C
      DIMENSION PI(3),P(1),PD(3),PDN(3),HPI(1),HXHIGH(2),HL(1),
     *HP(1),HPD(1),SYSID(2),FN1(2),X(3),WF(10),HBETA(1),HGAMMA(2),
     *HPDN(1),HLOSS(1),HN(1),HVDIST(2),HMAXFN(2),SYS1(2),HWF(1),
     *HX(1),HWX1(1),HWX2(1),HWX3(1),HWX4(1),HMODE(1),HTS(1),HTINC(1),
     *HXLOW(1)
C
      LOGICAL ISTOP
C
      COMMON /DESTIN /IDUM,IPART
C
      COMMON /USER/ISTOP
      COMMON /TIME /T
C
      COMMON/NAME/ N,EPS,IPRINT,TS,TINC,NPAR,PN(3),NPLUS1,NPLUS2,
     *WX1(3),WX2(3),WX3(3),WX4(3),WX5(3),WX6(3),WX7(3),
     *WXLOW(3),WXHIGH(3),ALOSS,BETA,GAMMA
C
      COMMON/DEVICE/LTO
C
      DATA HN,HVDIST,HMAXFN,HP,HLOSS,HPD,HPDN,HPI,HPN,HFMIN,HWF/4HN    ,
     *4HVDIS,4HT    ,4HMAXF,4HN    ,4HP    ,4HLOSS,4HPD   ,4HPDN ,4HPI   ,
     *4HPN   ,4HFMIN,4HWF   /
C
      DATA SYS1,HDISC/4HNELM,4HE    ,4HDISC/
C
      DATA HX,HWX1,HWX2,HWX3,HWX4,HMODE,HTS,HTINC,HBETA,HGAMMA,HXHIGH/
     *4HX    ,4HWX1 ,4HWX2 ,4HWX3 ,4HWX4 ,4HMODE,4HTS   ,4HTINC,4HBETA,
     *4HGAMM,4HA    ,4HXHIG,4HH    /
      DATA SYSID,FN1,HL,HXLOW/4HNELM,4HE    ,4HNPAR,4H      ,4HL    ,
     *4HXLOW/
C
      GOTO (1,2,3,4,5,6,7,8),IPART
C
    1 CALL IDENT2(HDISC,SYS1)
      RETURN
C
    2 CALL FINT(FN1,SYSID,NPAR,IND)
      N=NPAR
      IF(N.LE.1.OR.N.GT.3)THEN
        CALL IWRITE(LTO,1)
        WRITE(LTO,2170)
 2170   FORMAT('N MASTE VARA 2 ELLER 3')
        ISTOP=.TRUE.
      ENDIF
      CALL INITV(PI,N,HPI)
      CALL PARV(X,N,HX)
      CALL STATEV(PD,N,HPD)
      CALL NEWV(PDN,N,HPDN)
      CALL INPUT(ALOSS,HLOSS)
      CALL PARV(WX1,N,HWX1)
      CALL PARV(WX2,N,HWX2)
      CALL PARV(WX3,N,HWX3)
```

```
      CALL PARV(WX4,N,HWX4)
      CALL PAR(AMODE,HMODE)
      CALL PAR(BETA,HBETA)
      CALL PAR2(GAMMA,HGAMMA)
      CALL PAR(TINC,HTINC)
      CALL PAR2(VDIST,HVDIST)
      CALL TSAMP(TS,HTS)
      CALL OUTPUV(WF,10,HWF)
      CALL OUTPV2(WXHIGH,N,HXHIGH)
      CALL OUTPUV(WXLOW,N,HXLOW)
      CALL OUTPUT(AL,HL)
      CALL OUTPUV(PN,N,HPN)
      RETURN
C
C     INITIALVÄRDEN
C
    3 DO 20 I=1,N
   20 X(I)=0
      TINC=2.
      AMODE=0.0
      VDIST=0.1
      ALFA=1.
      BETA=-.5
      GAMMA=-2.0
      RETURN
C
C     UTRÄKNING AV INITIALVÄRDEN
C
    4 NPLUS1=N+1
      FLN=FLOAT(N)
      MODE=IFIX(AMODE)
      FLNPL1=FLOAT(NPLUS1)
      IF(N.LT.2) RETURN
      IF(MODE.EQ.1) GOTO 60
      IF(MODE.NE.0) RETURN
C
C     UTRÄKNING AV INITIALVÄRDENA I POLYHEDRONEN
C
      VDIST1=VDIST
      A=SQRT(2.0)
      B=SQRT(FLNPL1)
      D3=-VDIST1/(A*B)
      C=VDIST1/(A*FLN)
      B=B-1
      D2=C*B+D3
      D1=C*(B+FLN)+D3
      DO 40 I=1,N
      A=X(I)
      B=A+D2
      IF(N.EQ.2) WX3(I)=A+D3
      IF(N.EQ.3) WX4(I)=A+D3
      WX1(I)=B
      WX2(I)=B
      IF(N.EQ.3) WX3(I)=B
      IF(I.EQ.1)WX1(I)=A+D1
```

```
      IF(I.EQ.2) WX2(I)=A+D1
      IF(I.EQ.3) WX3(I)=A+D1
   40 CONTINUE
   60 DO 1000 K=1,N
 1000 PN(K)=WX1(K)
      L=1
      IVAL=1
      GOTO 300
C
C     OUTPUT-SEKTION
C
    5 WF(L)=ALOSS
C
      GOTO(101,102,103,104,105,106,107,108,109,110,111),IVAL
  101 DO 1001 K=1,N
 1001 PN(K)=WX2(K)
      L=2
      IVAL=2
      GOTO 300
C
  102 DO 1002 K=1,N
 1002 PN(K)=WX3(K)
      L=3
      IVAL=3
      IF(N.EQ.2) IVAL=4
      GOTO 300
C
  103 DO 1003 K=1,N
 1003 PN(K)=WX4(K)
      L=4
      IVAL=4
      GOTO 300
C
C     ITERATIONS-LOOP
C
  104 CONTINUE
   70 CONTINUE
      IHIGH=NPLUS1
      ILOW=NPLUS1
      DO 80 I=1,N
      IF(WF(I).GT.WF(IHIGH)) IHIGH=I
      IF(WF(I).LT.WF(ILOW)) ILOW=I
   80 CONTINUE
      DO 100 I=1,N
      IF(IHIGH.EQ.1) WXHIGH(I)=WX1(I)
      IF(IHIGH.EQ.2) WXHIGH(I)=WX2(I)
      IF(IHIGH.EQ.3) WXHIGH(I)=WX3(I)
      IF(IHIGH.EQ.4) WXHIGH(I)=WX4(I)
      A=-WXHIGH(I)
      IF(ILOW.EQ.1) WXLOW(I)=WX1(I)
      IF(ILOW.EQ.2) WXLOW(I)=WX2(I)
      IF(ILOW.EQ.3) WXLOW(I)=WX3(I)
      IF(ILOW.EQ.4) WXLOW(I)=WX4(I)
      A=A+WX1(I)+WX2(I)+WX3(I)
      IF(N.EQ.3) A=A+WX4(I)
```

```
      WX5(I)=A/FLN
  100 CONTINUE
      DO 116 K=1,N
  116 PN(K)=WX5(K)
      L=5
      IVAL=5
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(5)   I CENTROID
C
C     UTRÄKNING AV REFLEKTIONSPUNKTEN N+3
C
  105 DO 1004 K=1,N
 1004 WX6(K)=WX5(K)+ALFA*(WX5(K)-WXHIGH(K))
      DO 1005 K=1,N
 1005 PN(K)=WX6(K)
      L=6
      IVAL=6
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(6) I
C     REFLEKTIONSPUNKTEN.
C
C     UTRÄKNING AV EXPANSIONSPUNKTEN N+4
C
  106 IF(WF(6).GE.WF(ILOW)) GOTO 120
      DO 115 K=1,N
  115 WX7(K)=WX5(K)+GAMMA*(WX5(K)-WX6(K))
      DO 1006 K=1,N
 1006 PN(K)=WX7(K)
      L=7
      IVAL=7
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(7) I
C     EXPANSIONSPUNKTEN.
C
  107 IF(WF(7).GE.WF(ILOW)) GOTO 1100
      DO 1007 K=1,N
      IF(IHIGH.EQ.1) WX1(K)=WX7(K)
      IF(IHIGH.EQ.2) WX2(K)=WX7(K)
      IF(IHIGH.EQ.3) WX3(K)=WX7(K)
 1007 IF(IHIGH.EQ.4) WX4(K)=WX7(K)
      WF(IHIGH)=WF(7)
      GOTO 70
 1100 CONTINUE
      DO 1008 K=1,N
      IF(IHIGH.EQ.1) WX1(K)=WX6(K)
      IF(IHIGH.EQ.2) WX2(K)=WX6(K)
      IF(IHIGH.EQ.3) WX3(K)=WX6(K)
 1008 IF(IHIGH.EQ.4) WX4(K)=WX6(K)
      WF(IHIGH)=WF(6)
      GOTO 70
  120 CONTINUE
      DO 130 I=1,NPLUS1
```

```
      IF(WF(6).GT.WF(I)) GOTO 130
      IF(I.EQ.IHIGH) GOTO 130
      GOTO 1100
  130 CONTINUE
      IF(WF(6).GE.WF(IHIGH)) GOTO 140
      DO 1009 K=1,N
      IF(IHIGH.EQ.1) WX1(K)=WX6(K)
      IF(IHIGH.EQ.2) WX2(K)=WX6(K)
      IF(IHIGH.EQ.3) WX3(K)=WX6(K)
 1009 IF(IHIGH.EQ.4) WX4(K)=WX6(K)
      WF(IHIGH)=WF(6)
      IHIGH=NPLUS1
      DO 140 I=1,N
      IF(WF(I).GT.WF(IHIGH)) IHIGH=I
  140 CONTINUE
      DO 141 I=1,N
      IF(IHIGH.EQ.1)WXHIGH(I)=WX1(I)
      IF(IHIGH.EQ.2)WXHIGH(I)=WX2(I)
      IF(IHIGH.EQ.3)WXHIGH(I)=WX3(I)
      IF(IHIGH.EQ.4)WXHIGH(I)=WX4(I)
  141 CONTINUE
C
C     UTRÄKNING AV KONTRAKTIONSPUNKTEN N+4
C
      DO 1010 K=1,N
 1010 WX7(K)=WX5(K)+BETA*(WX5(K)-WXHIGH(K))
      DO 1011 K=1,N
 1011 PN(K)=WX7(K)
      L=7
      IVAL=8
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(7) I
C     KONTRAKTIONSPUNKTEN.
C
  108 IF(WF(7).GT.WF(IHIGH)) GOTO 150
      DO 1012 K=1,N
      IF(IHIGH.EQ.1) WX1(K)=WX7(K)
      IF(IHIGH.EQ.2) WX2(K)=WX7(K)
      IF(IHIGH.EQ.3) WX3(K)=WX7(K)
 1012 IF(IHIGH.EQ.4) WX4(K)=WX7(K)
      WF(IHIGH)=WF(7)
      GOTO 70
  150 CONTINUE
      NFN=NFN+N
C
C     REDUKTION AV POLYHEDRONEN
C
      IF(ILOW.EQ.1) GOTO 109
      DO 1013 K=1,N
 1013 WX1(K)=WXLOW(K)-0.5*(WXLOW(K)-WX1(K))
      DO 1014 K=1,N
 1014 PN(K)=WX1(K)
      L=1
      IVAL=9
```

```
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(1) I VERTEX 1
C
  109 IF(ILOW.EQ.2) GOTO 110
      DO 1015 K=1,N
 1015 WX2(K)=WXLOW(K)-0.5*(WXLOW(K)-WX2(K))
      DO 1016 K=1,N
 1016 PN(K)=WX2(K)
      L=2
      IVAL=10
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(2) I VERTEX 2
C
  110 IF(ILOW.EQ.3) GOTO 111
      DO 1017 K=1,N
 1017 WX3(K)=WXLOW(K)-0.5*(WXLOW(K)-WX3(K))
      DO 1018 K=1,N
 1018 PN(K)=WX3(K)
      L=3
      IVAL=11
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(3) I VERTEX 3
C
  111 IF((ILOW.EQ.4).OR.(N.EQ.2))  GOTO 70
  112 DO 1019 K=1,N
 1019 WX4(K)=WXLOW(K)-0.5*(WXLOW(K)-WX4(K))
      DO 1020 K=1,N
 1020 PN(K)=WX4(K)
      L=4
      IVAL=4
      GOTO 300
C
C     SIMULERA OCH BERÄKNA FUNKTIONSVÄRDE WF(4) I VERTEX 4
C
  300 DO 290 I=1,N
  290 PD(I)=PN(I)
      AL=L
      TS=TS+TINC
C
C
      RETURN
C
    6 RETURN
    7 RETURN
    8 RETURN
C
C
C
      END
```

5. SIMULERINGAR

5.1 TESTPROCESSER

I simuleringarna har använts processer, vars överföringsfunktioner har följande utseende:

1. En andra ordningens process:

$$(33)\quad G(s)=\frac{1}{(1+0.5s)(1+s)}$$

2. En tredje ordningens process:

$$(34)\quad G(s)=\frac{1}{(1+0.5s)(1+s)(1+5s)}$$

3. En fjärde ordningens process (uttryckt i z-transform):

$$(35)\quad G(z)=\frac{1}{(1-0.8z^{-1})^4}$$

5.2 SIMULERINGSKOMMANDO

För att underlätta simuleringen har de olika SIMNON-kommandona sammanförts i macron. För betydelsen av resp. kommando hänvisas till SIMNON-manualen.

För initieringen av simuleringssystemet kan följande macron användas:

```
MACRO GONELME X Y
LET NPAR.NELME=X
"NPAR=ANTAL VARIABLER
SYST RGM MOD M FKN MT NELME Y
"Y=MKON2 FÖR PI-REGULATOR
"Y=MKON3 FÖR PID-REGULATOR
END

MACRO GOOPTA X Y
LET NPAR.OPTA=X
LET NCONS.OPTA=0
"NCONS=ANTAL VILLKOR
SYST RGM MOD M FKN MT OPTA Y
END
```

Om parameterfil ej lagrats tidigare göres detta med kommandot SAVE (filnamn). I annat fall hämtas startvärden från parameter-filen med kommandot GET (filnamn).

För lagring av olika intressanta variabler och plottning av dessa finns följande macron:

```
MACRO SIMNE X E Z A B C D
"X=SIMULERINGSTIDEN
"E=10 OCH Z=12.2 GER
"PLOTTNING MELLAN DESSA TIDER
"A OCH B RESP C OCH D GÄLLER
"SAMMA SOM FÖR E OCH Z
STORE LOSS(FKN) K(RGM) TI(RGM) TD(RGM)
SPLIT 3 1
AREA 1 1
SIMU 0 X-MARK
ASHOW E Z Y(M) Y(MOD)-MARK
TEXT'1=YM 2=YMOD I START'
ASHOW A B Y(M) Y(MOD)-MARK
TEXT'1=YM 2=YMOD I MITT'
ASHOW C D Y(M) Y(MOD)-MARK
TEXT'1=YM 2=YMOD I SLUT'
END
```

```
MACRO SIMNE2 X
SPLIT 2 1
AXES H 0 X V 0 5
SHOW LOSS(FKN)
TEXT'LOSS'
ASHOW K(RGM) TI(RGM) TD(RGM)-MARK
TEXT'K=1 TI=2 TD=3'
DISP LOSS(FKN) K(RGM) TI(RGM) TD(RGM)
END
```

För att ändra vissa parametrars startvärde kan följande macron användas:

```
MACRO SETNELME X Y Z A
"STARTKOORDINATER FÖR CENTROID
PAR X1(NELME):X
PAR X2(NELME):Y
PAR X3(NELME):Z
"VDIST=AVSTAND MELLAN
"HÖRN I POLYHEDRONEN
PAR VDIST:A
END

MACRO SETOPTA X Y A B D
"STARTVÄRDE FÖR K OCH TI
INIT PI1:X
INIT PI2:Y
"STEGLÄNGD FÖR ALGORITM=XM*HH
PAR XM1(OPTA):A
PAR XM2(OPTA):B
PAR HH:D
END
```

Följande sekvens av kommandon simulerar ett söksystem med 3:e ordningens modellprocess i 150s, och ger en figur på grafisk skärm motsvarande figur 5.5.

```
GONELME 3 MKON3
GET MPAR30
SETNELME 2 2 3 0.2
SIMNE 150 10 12 76 78 146 148
SIMNE2 150
```

5.3 FUNKTIONSMINIMERING

Minimering av en känd funktion med två eller tre variabler (begränsning hos vår programkod) kan göras m.h.a. NELME enligt följande uppbyggnad av systemet:

Fig. 5.1 Blockschema

Exempel på kod för minimering av Rosenbrocks funktion.

Initiering:

```
MACRO GOFUNC X
LET NPAR.NELME=X
SYST FUNC NELME FKON
END

DISCRETE SYSTEM FUNC
INPUT X1 X2
TSAMP TS
TIME T
STATE F
NEW NF
NF=100*(X2-X1*X1)*(X2-X1*X1)+(1-X1)*(1-X1)
TS=T+DTM
DTM:2
END

CONNECTING SYSTEM FKON
TIME T
X1(FUNC)=PN1(NELME)
X2(FUNC)=PN2(NELME)
LOSS(NELME)=F(FUNC)
END
```

Parameterfil kan skapas med SAVE FUNCPAR, och ändrade startvärde kan föras in m.h.a. SIMNONs editor och återkallas med GET FUNCPAR.

5.4 SIMULERINGSRESULTAT

Simuleringar har gjorts med två olika optimeringsmetoder: Fletcher-Reeves-metoden och Nelder-Mead-metoden. (förkortas F.R. resp. N.M.) Inledningsvis visas när en känd funktion använts för optimering. De till denna funktion körda simuleringarna är tvådimensionella (två varierbara parametrar) för att kunna demonstreras på papper, men inget hinder föreligger för att öka antalet dimensioner till tre.

Funktionen "ROSENBROCKS VALLEY" lyder:

$$(36)\quad f(x)=100*(x_2-x_1^2)^2+(1-x_1)^2$$

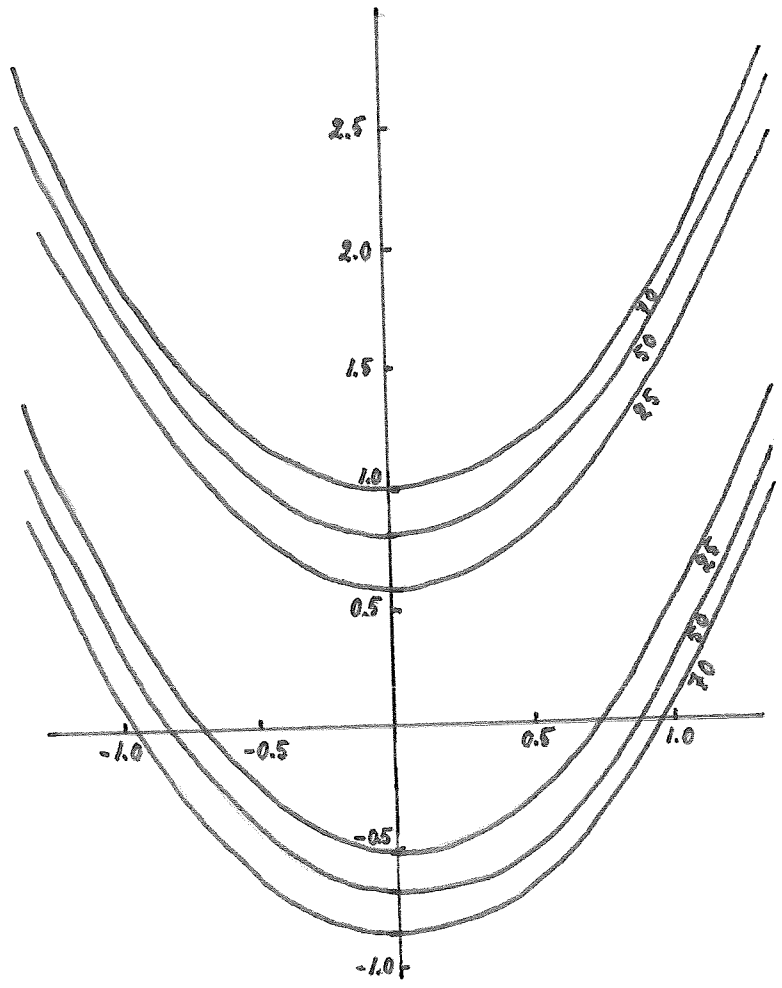

Fig. 5.2 Rosenbrocksfunktionens nivåkurvor

I följande figurer är den övre kurvan LOSS i varje simulering förlustfunktionens värde vid olika tidpunkter, medan den undre kurvan demonstrerar hur metoden söker sig fram till bättre funktionsvärden. Minimum ligger i punkten (1,1). I de två första simuleringarna har givits ganska snälla startvärden, (-1,1).

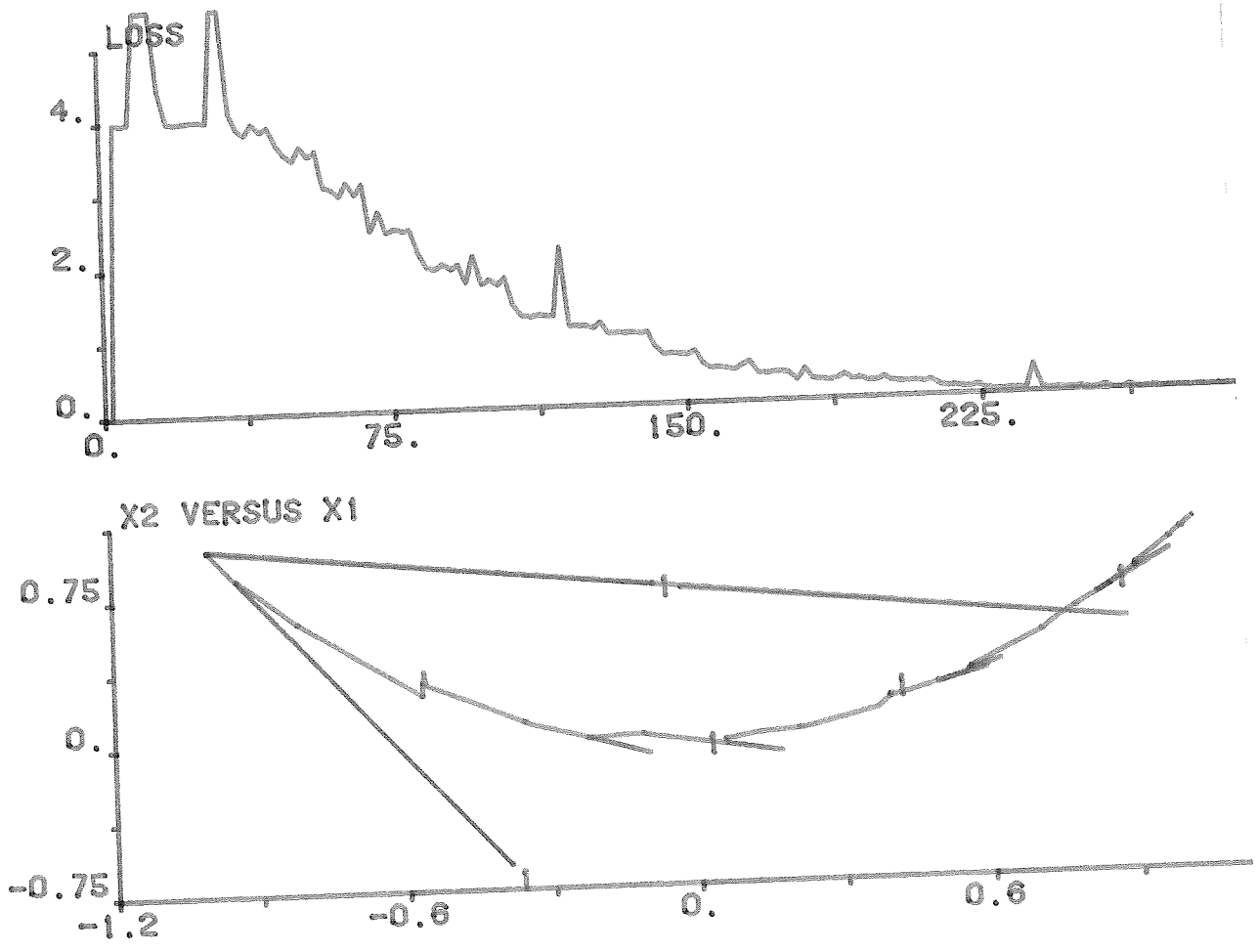

Fig. 5.3 F.R.-metoden i Rosenbrocks Valley

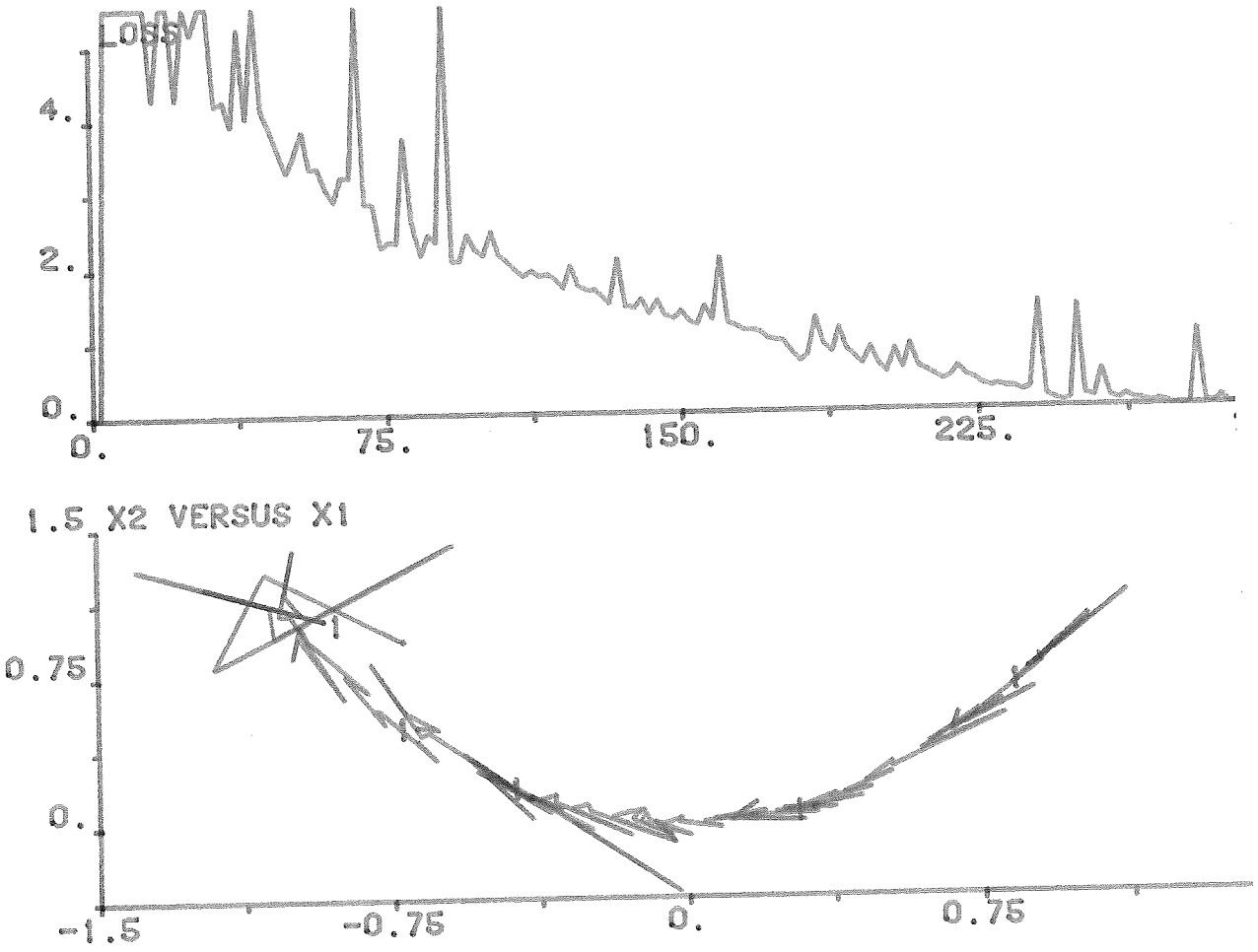

Fig. 5.4 N.M.-metoden i Rosenbrocks Valley

Ur dessa kurvor kan utläsas att F.R. är snabbare än N.M. bl.a. beroende på att den använder längre steg och därför snabbare kommer fram till minimum.

I de nästa två simuleringarna har givits sämre startvärden, (10,20).

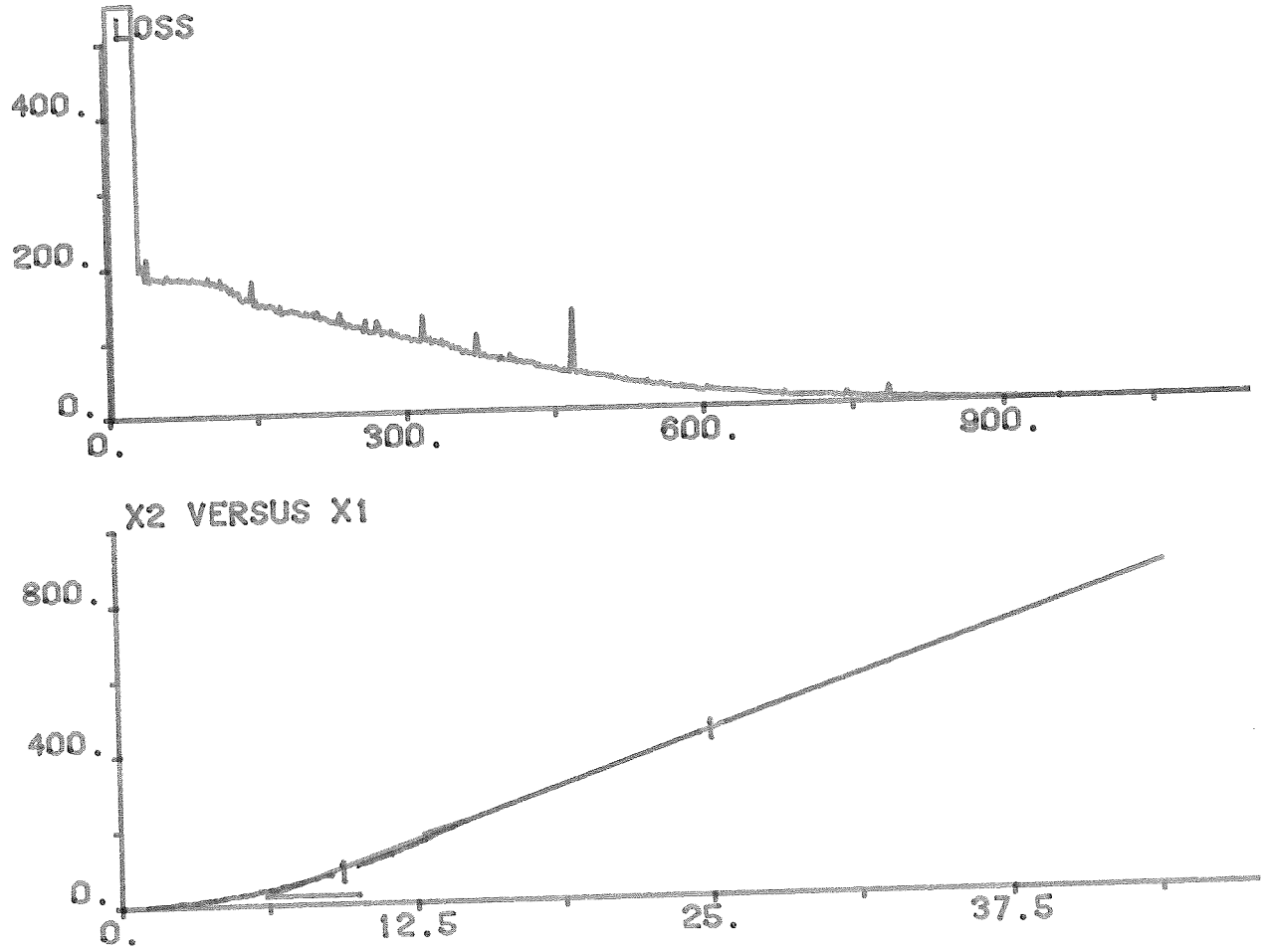

Fig. 5.5 F.R.-metoden i Rosenbrocks Valley

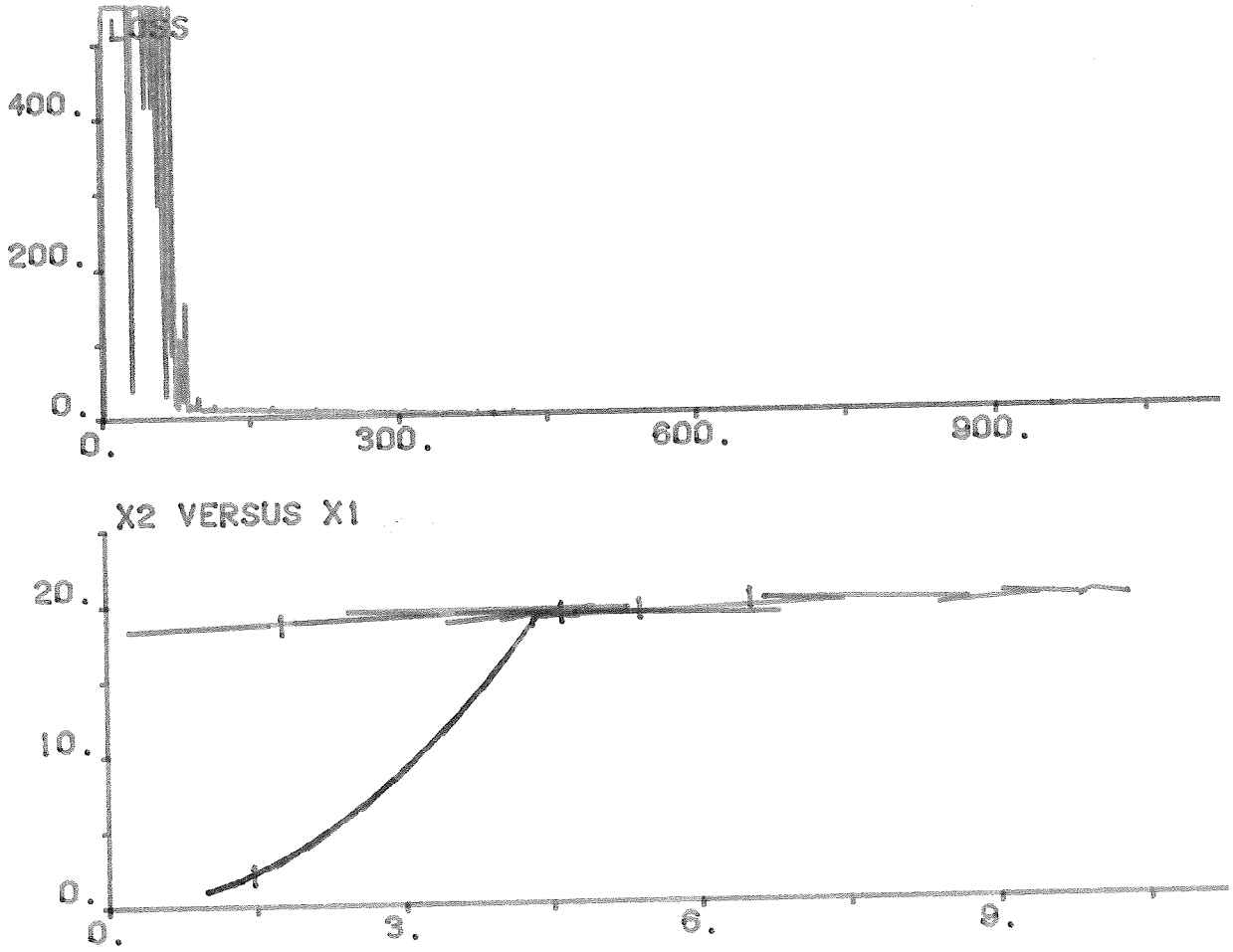

Fig. 5.6 N.M.-metoden i Rosenbrocks Valley

Som synes kommer N.M. ganska snabbt fram till dalsänkan och söker sig i denna fram till minimum, medan F.R. har större problem och ett tag är helt ute på fel spår.

Av dessa simuleringarna och andra liknande som vi kört framgår det klart att med hyggliga startvärden är F.R. snabbare, medan N.M. är en säkrare metod.

Omfattande simuleringsförsök har gjorts med huvudsakligen modellreferens. Här nedan följer en sammanfattning av dessa.

Startvärdenas inflytande.

Från våra körningar med N.M.-metoden visade det sig att startvärdena hade betydelse för hur snabbt förlustfunktionen gick under ett visst värde. Däremot konvergerade parametrarna alltid mot ungefär samma slutvärde oberoende av startvärdena.

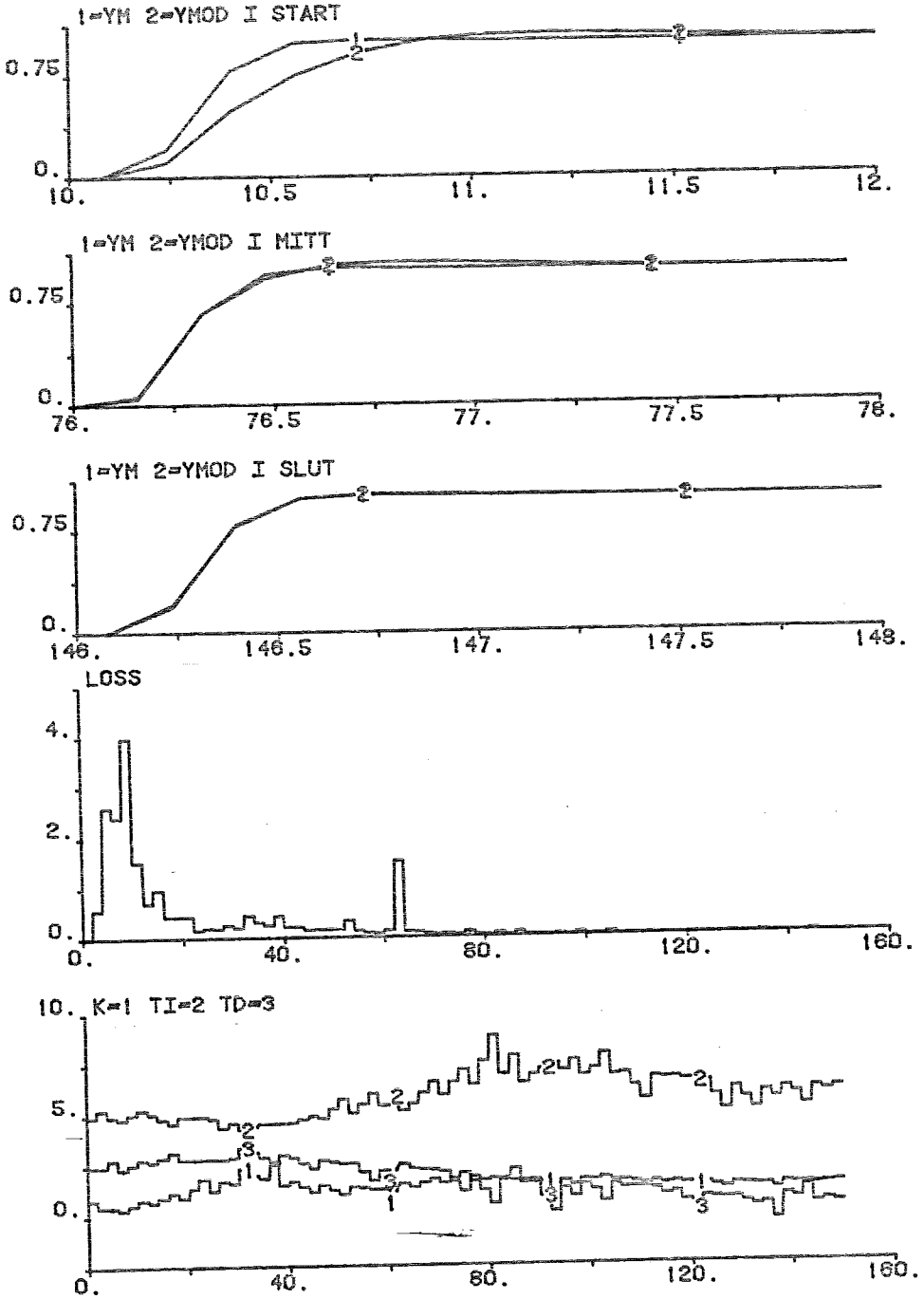

Fig.5.5. Modellreferens, 3:e ordningen med ISE-kriterium. Startvärden: K=0.5 TI=5 TD=2.5 Slutvärden efter 150s: K=1.33 TI=5.85 TD=0.61 LOSS=0.011

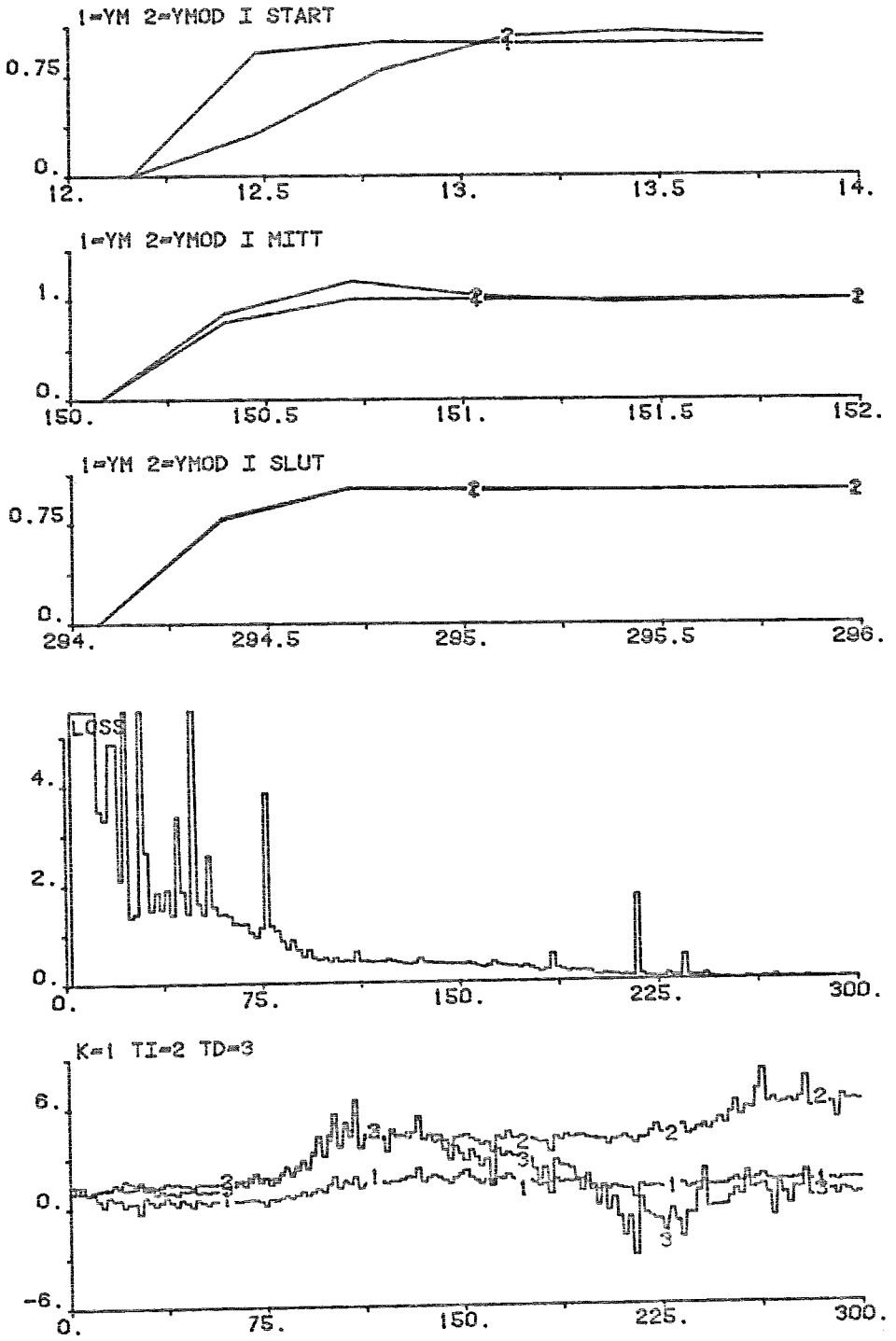

Fig.5.6. Modellreferens, 3:e ordningen med ISE-kriterium. Startvärden: K=1 TI=1 TD=1 Slutvärden efter 300s: K=1.32 TI=6.08 TD=0.33 LOSS=0.003

Anm. Som synes varierar td kraftigt, förmodligen beroende på ett flackt minimum för denna variabel.

Vdists inflytande

Vdist är en parameter i N.M.-metoden som anger avståndet mellan punkterna i den till metoden använda polyhedronen. Med stort vdist tas stora steg i sökningen efter bättre funktionsvärde, och då kommer man i början ofta snett vilket leder till högt förlustfunktionstal. Detta brukar dock snabbt korrigeras. Stort vdist innebär också att regulatorparametrarna blir oroliga i början. De konvergerar dock till ungefär samma värden som parametrarna med litet vdist.

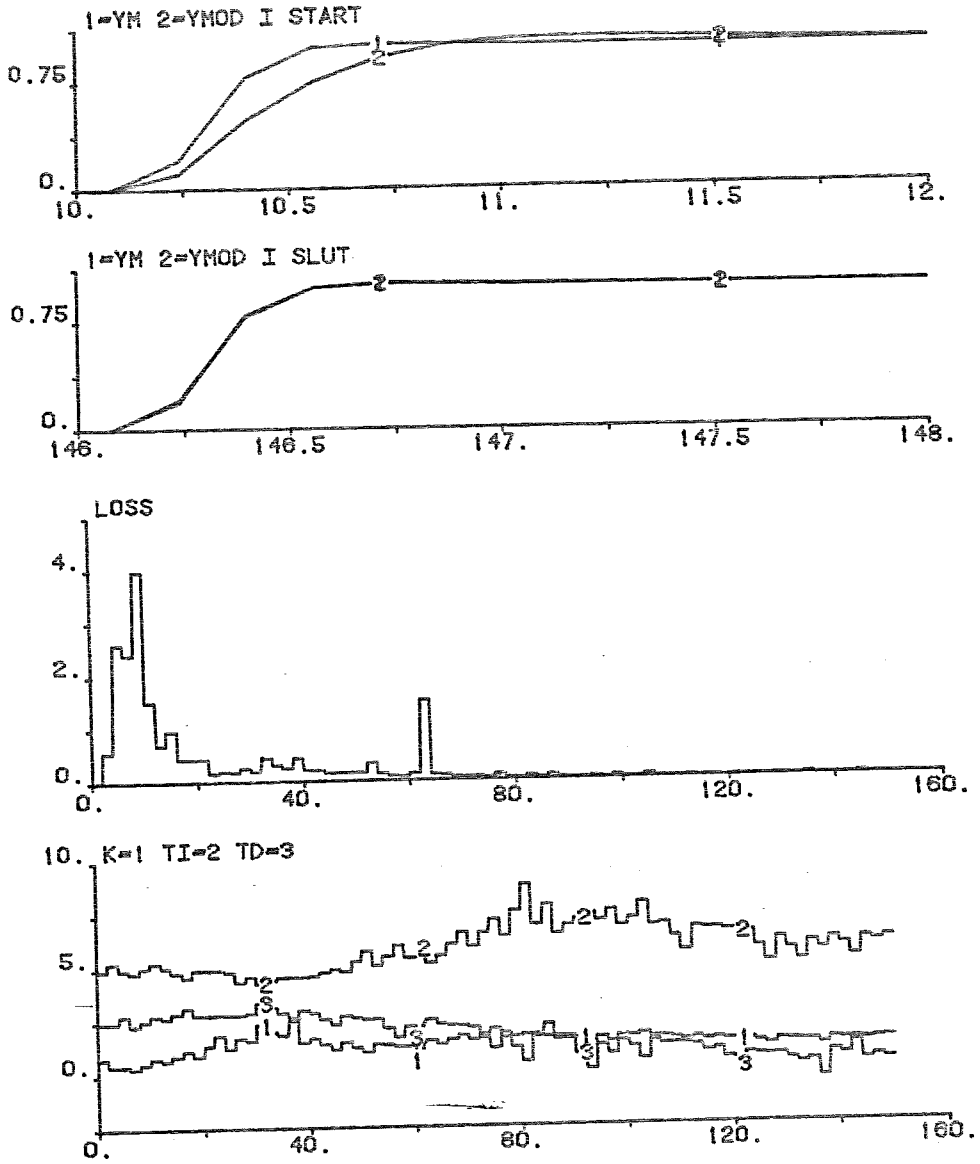

Fig.5.7. Modellreferens, 3:e ordningen med ISE-kriterium. VDIST=0.5 Startvärden: K=0.5 TI=5 TD=2.5 Slutvärden efter 150s: K=1.33 TI=5.85 TD=0.61 LOSS=0.011

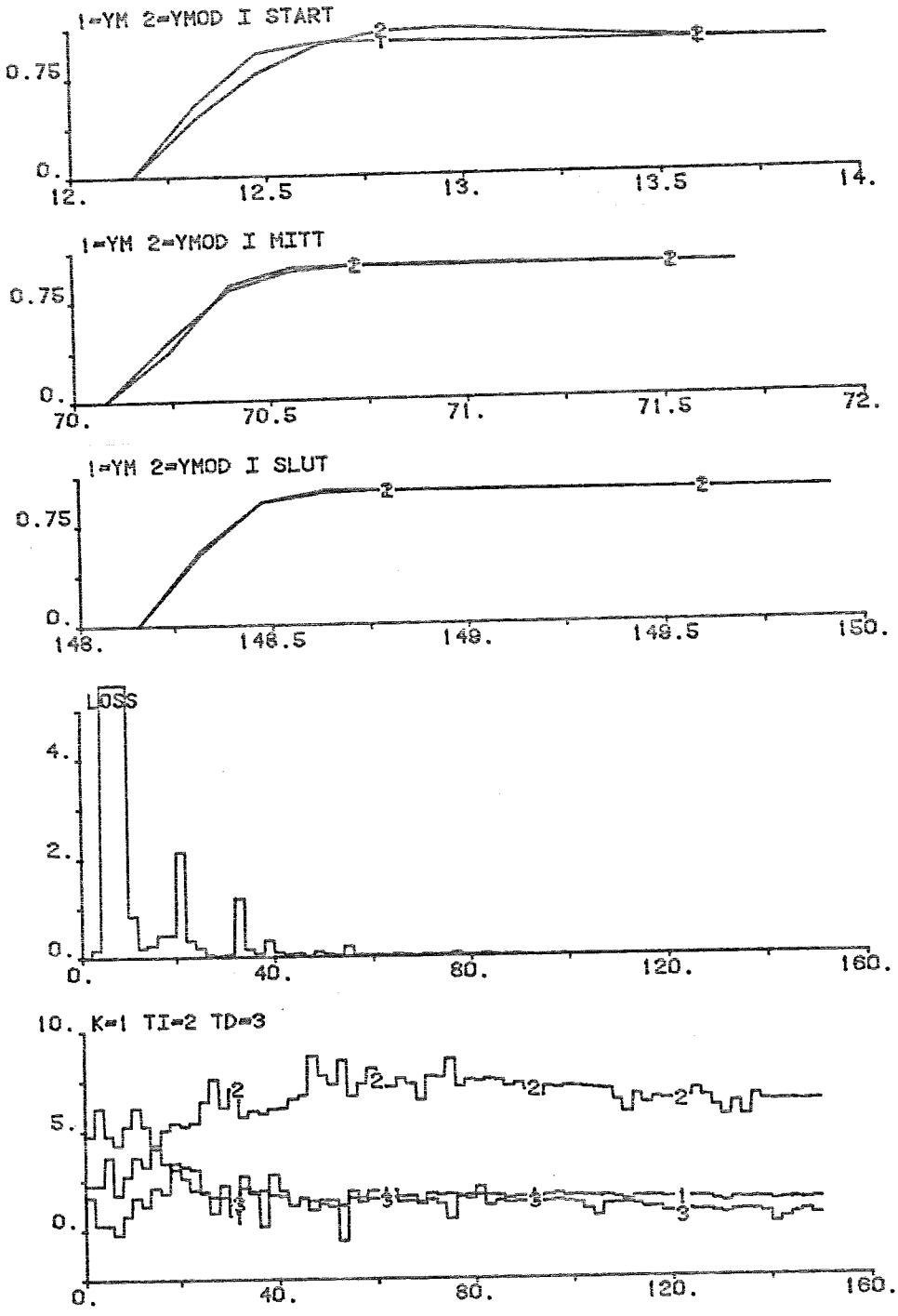

Fig.5.8. Modellreferens, 3:e ordningen med ISE-kriterium. VDIST=2 Startvärden: K=0.5 TI=5 TD=2.5 Slutvärden efter 150s: K=1.38 TI=6.27 TD=0.58 LOSS=0.004

Anm. N.M.-metoden korrigerar efterhand avståndet mellan punkterna i polyhedronen. Initialavståndet vdist kommer således att spela en allt mindre roll ju längre simuleringen pågår.

Olika kriterier

En jämförelse mellan ISE- och ITAE-kriterium visar vad som redan är bekant, nämligen att med ITAE sammanfaller modellens och processmodellens stegsvar tidigare än med ISE. Detta är naturligt eftersom ITAE straffar hårdare ju längre tid som går. En nackdel till följd av detta är dock att man får en kraftigare översläng i början.

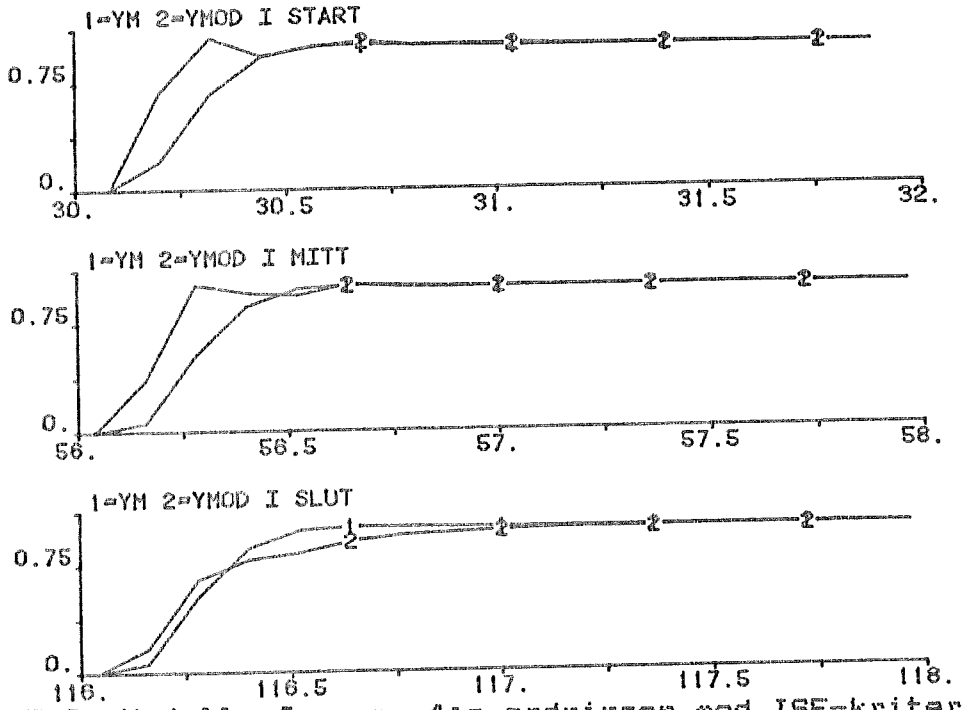

Fig.5.9. Modellreferens, 4:e ordningen med ISE-kriterium.

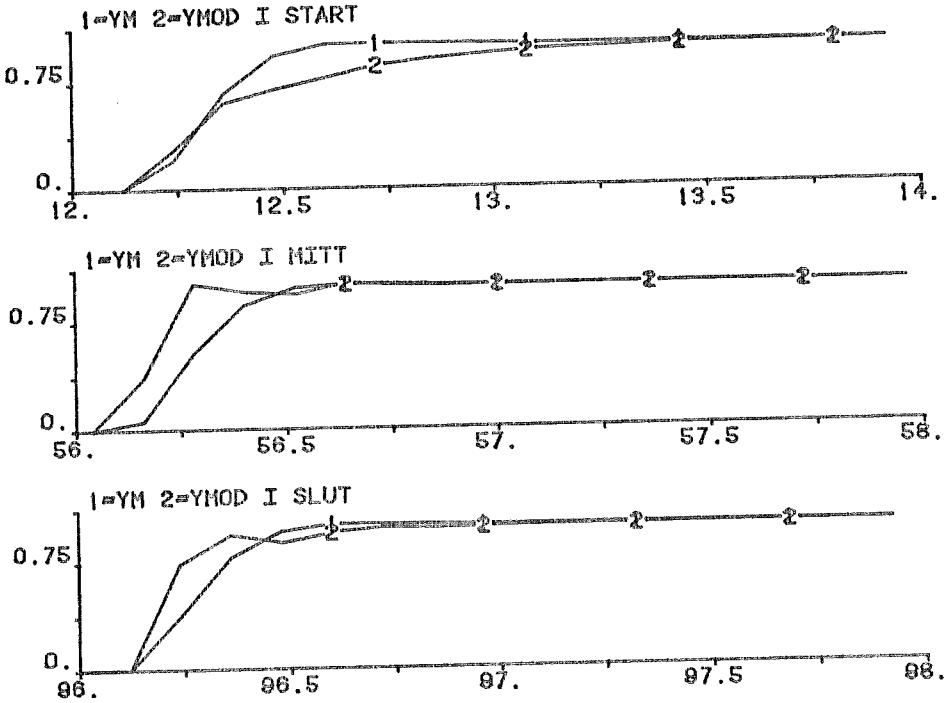

Fig.5.10. Modellreferens, 4:e ordningen, ITAE-kriterium.

Ifråga om snabbhet kan man i de nästa figurerna se att ISE är något bättre än ITAE.

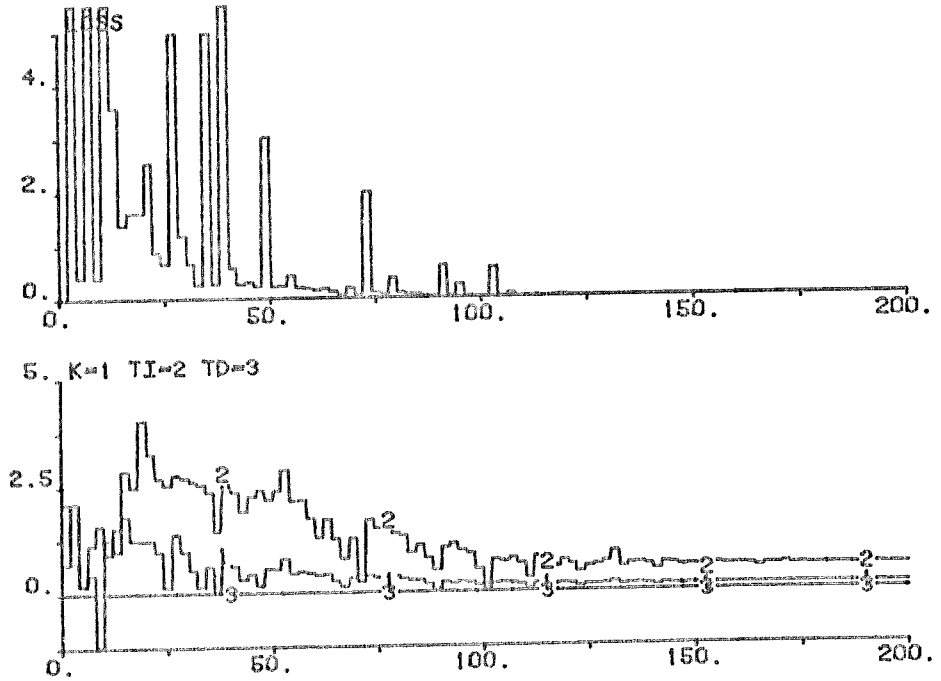

Fig.5.11. Modellreferens, 2:a ordningen, ISE-kriterium.

Efter litet mer än 100s är LOSS≈0 även om regulatorparametrarna varierar fram till 150s.

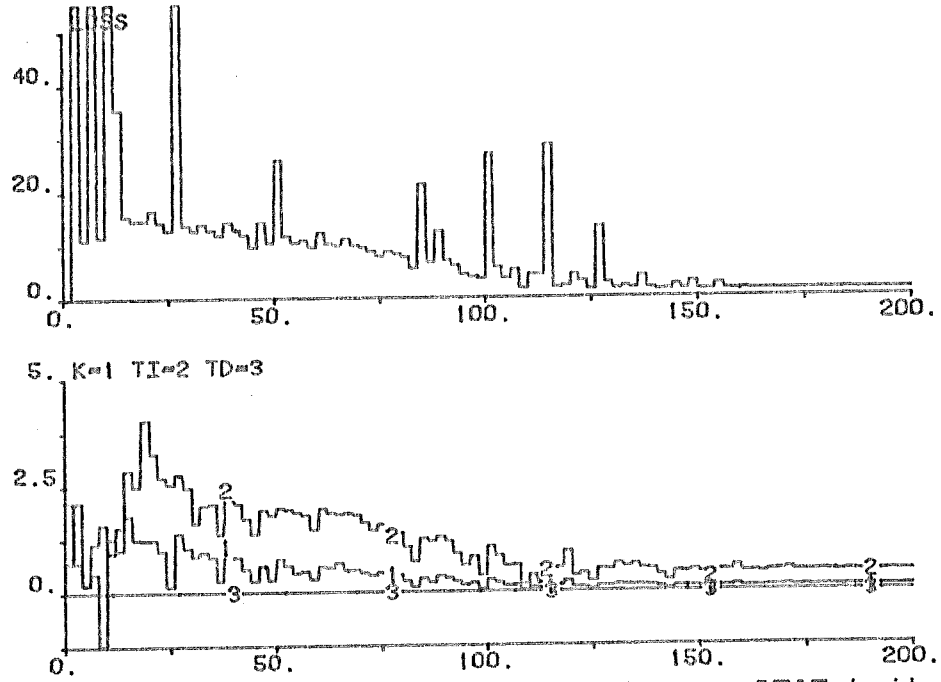

Fig.5.12. Modellreferens, 2:a ordningen, ITAE-kriterium.

Efter litet mer än 150s har LOSS stabiliserats. (Detta LOSS går ej att jämföra med LOSS för ISE eftersom de erhållits på olika sätt.)

Jämförelser, Fletcher-Reeves- och Nelder-Mead-metoden.

Jämförelser har gjorts mellan F.R.- och N.M.-metoderna för 2, 3 och 4:e ordningens system med modellreferens och får 4:e ordningens system med insignalreferens.

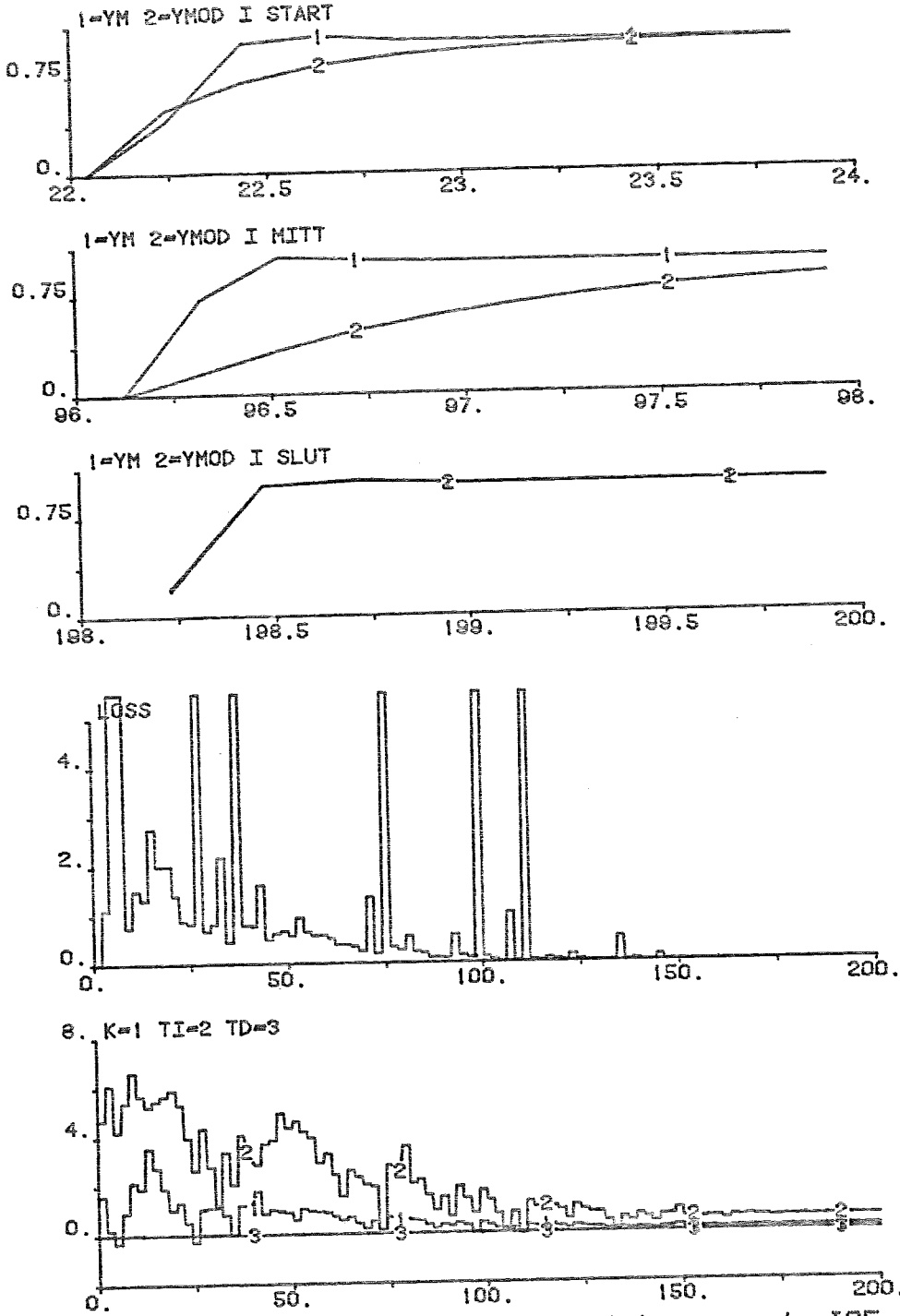

Fig. 5.13. Modellreferens, 2:a ordningen med ISE enligt N.M.-metoden. Startvärden: K=0.5 TI=5 Slutvärden: K=0.14 TI=0.59 LOSS=0.001 (Stegsvaret i mitten är dåligt beroende på att rutinen just tagit ett steg mot högre funktionsvärde.)

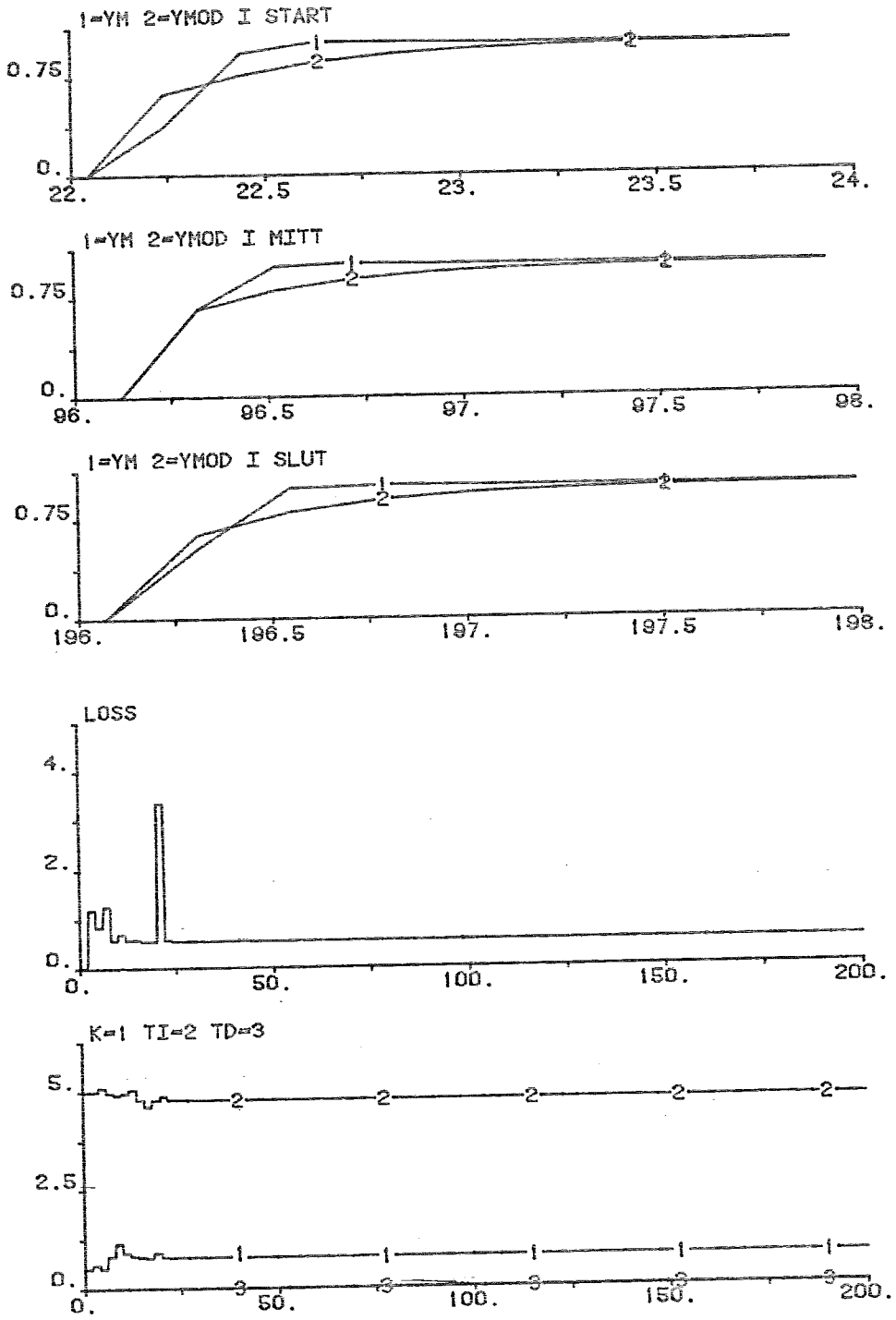

Fig. 5.14. Modellreferens, 2:a ordningen med ISE enligt F.R.-metoden. Startvärden: K=0.5 TI=5 Slutvärden: K=0.80 TI=4.78 LOSS=0.54

De två figurerna ovan demonstrerar F.R.-metodens osäkerhet. Trots att startvärdena är i någorlunda rätt storleksordning klarar inte F.R.-metoden av att hitta minimum, utan hamnar i en återvändsgränd efter 25 s.

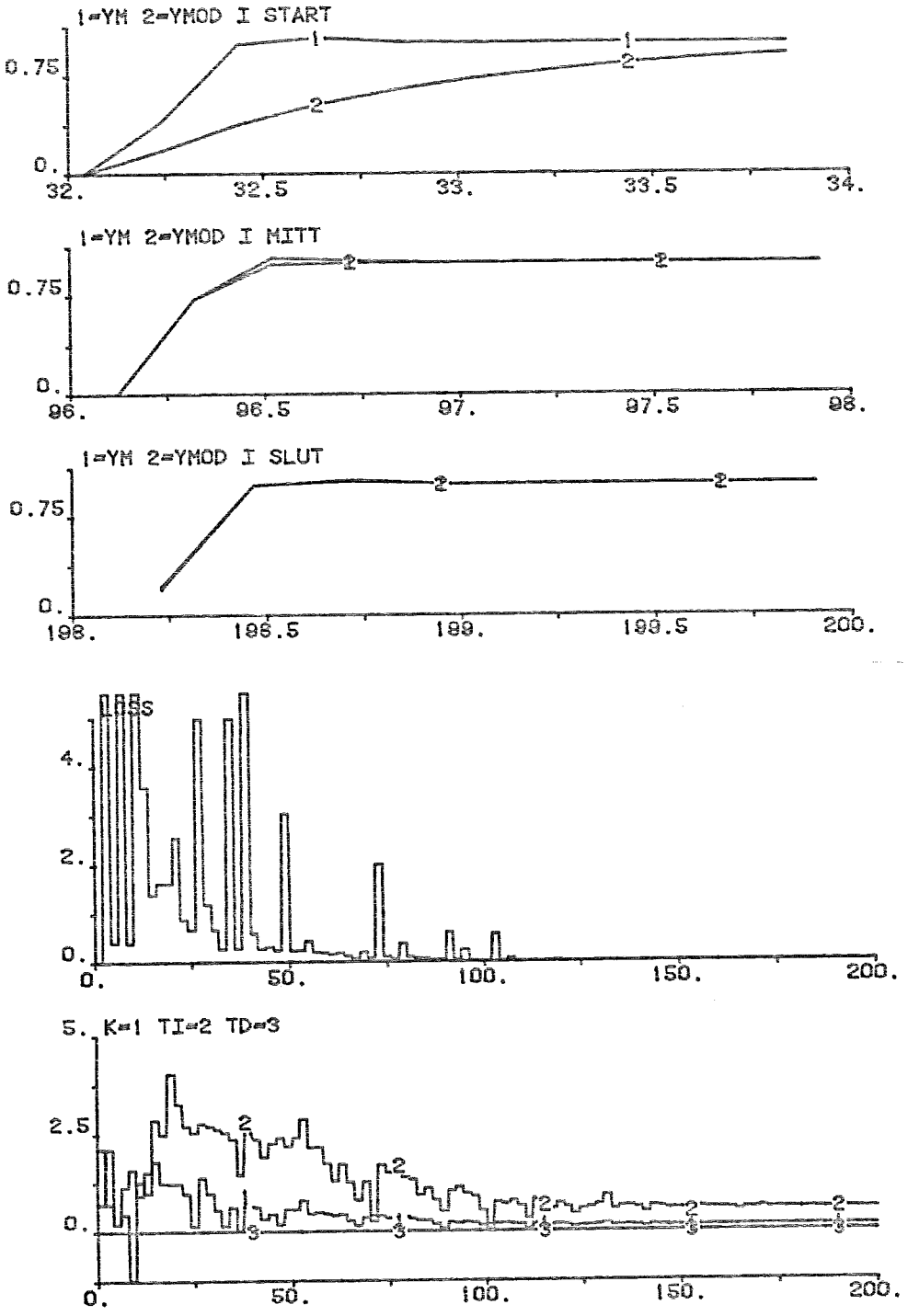

Fig. 5.15. Modellreferens, 2:a ordningen med ISE enligt N.M.-metoden. Startvärden: K=TI=1 Slutvärden: K=0.14 TI=0.59 LOSS=0.001

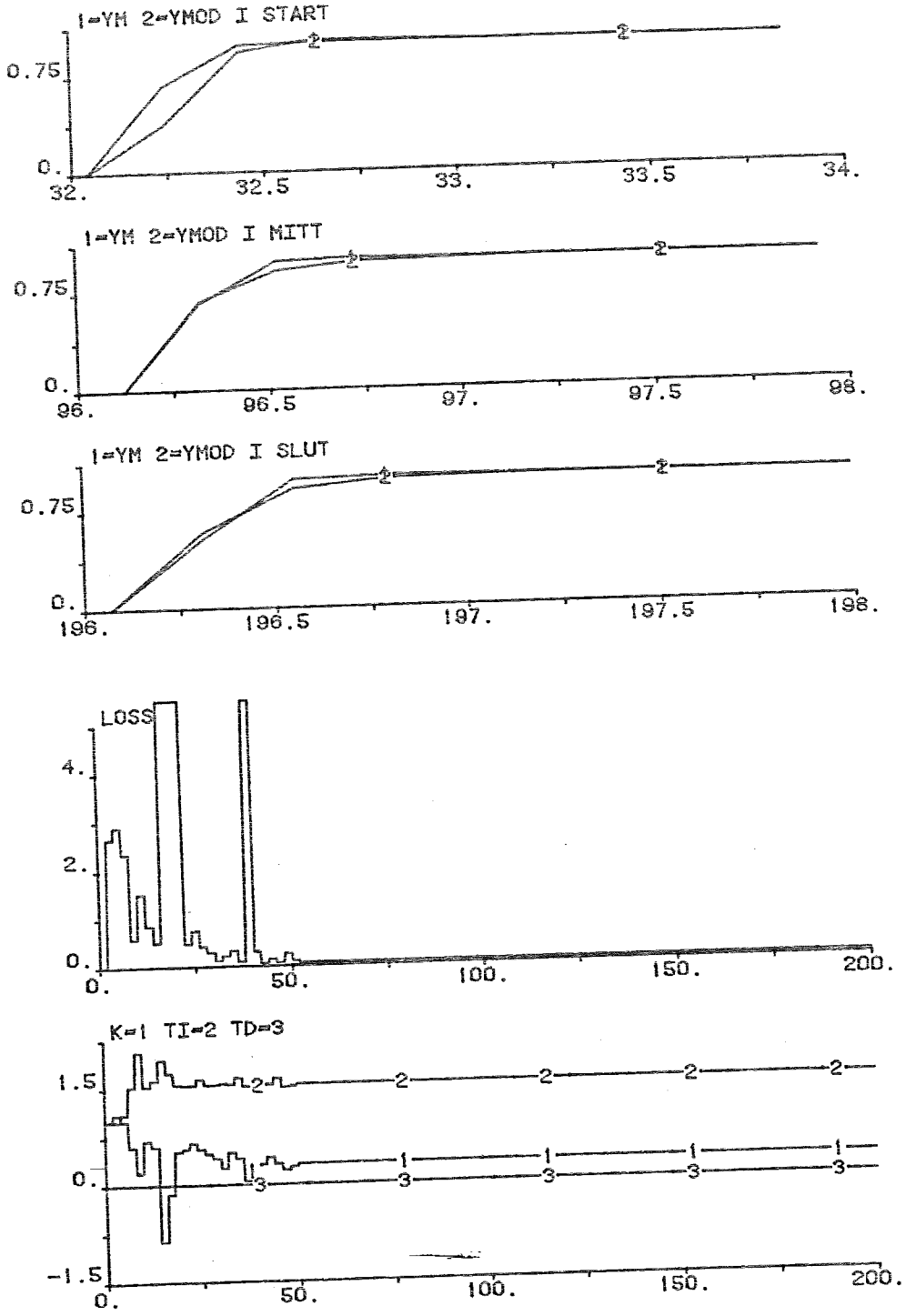

Fig. 5.16. Modellreferens, 2:a ordningen med ISE enligt F.R.-metoden. Startvärden: K=TI=1 Slutvärden: K=0.32 TI=1.55 LOSS=0.06

Figurerna 5.15 och 5.16 visar att även med dessa startvärden är N.M.-metoden överlägsen.

N.M.-metoden är bättre än F.R. även för 3:e ordningens system. Se figur 5.17 och 5.18.

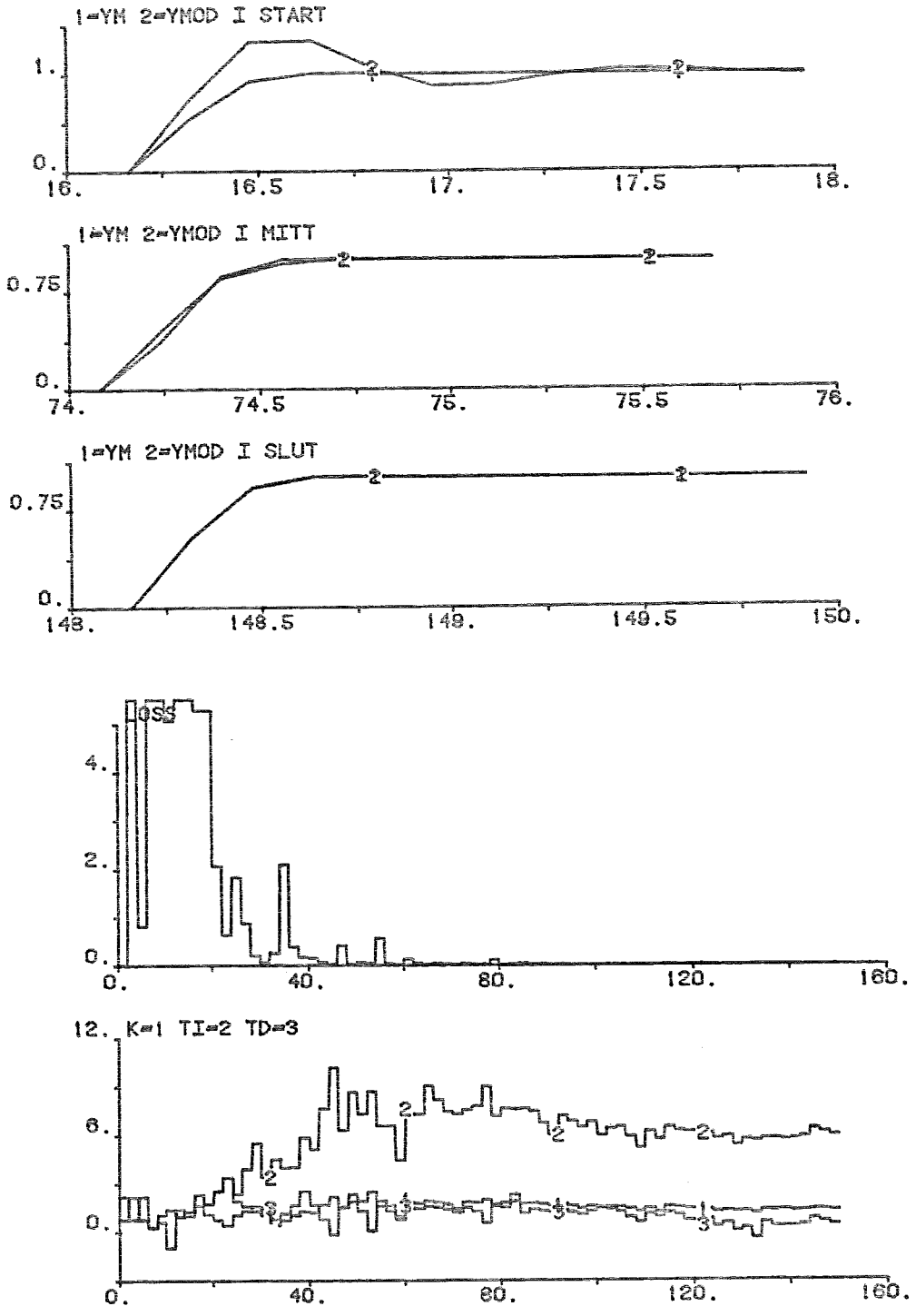

Fig. 5.17. Modellreferens, 3.e ordningen med ISE enligt N.M.-metoden. Startvärden: K=TI=TD=1 Slutvärden: K=1.31 TI=6.19 TD=0.29 LOSS=0.002

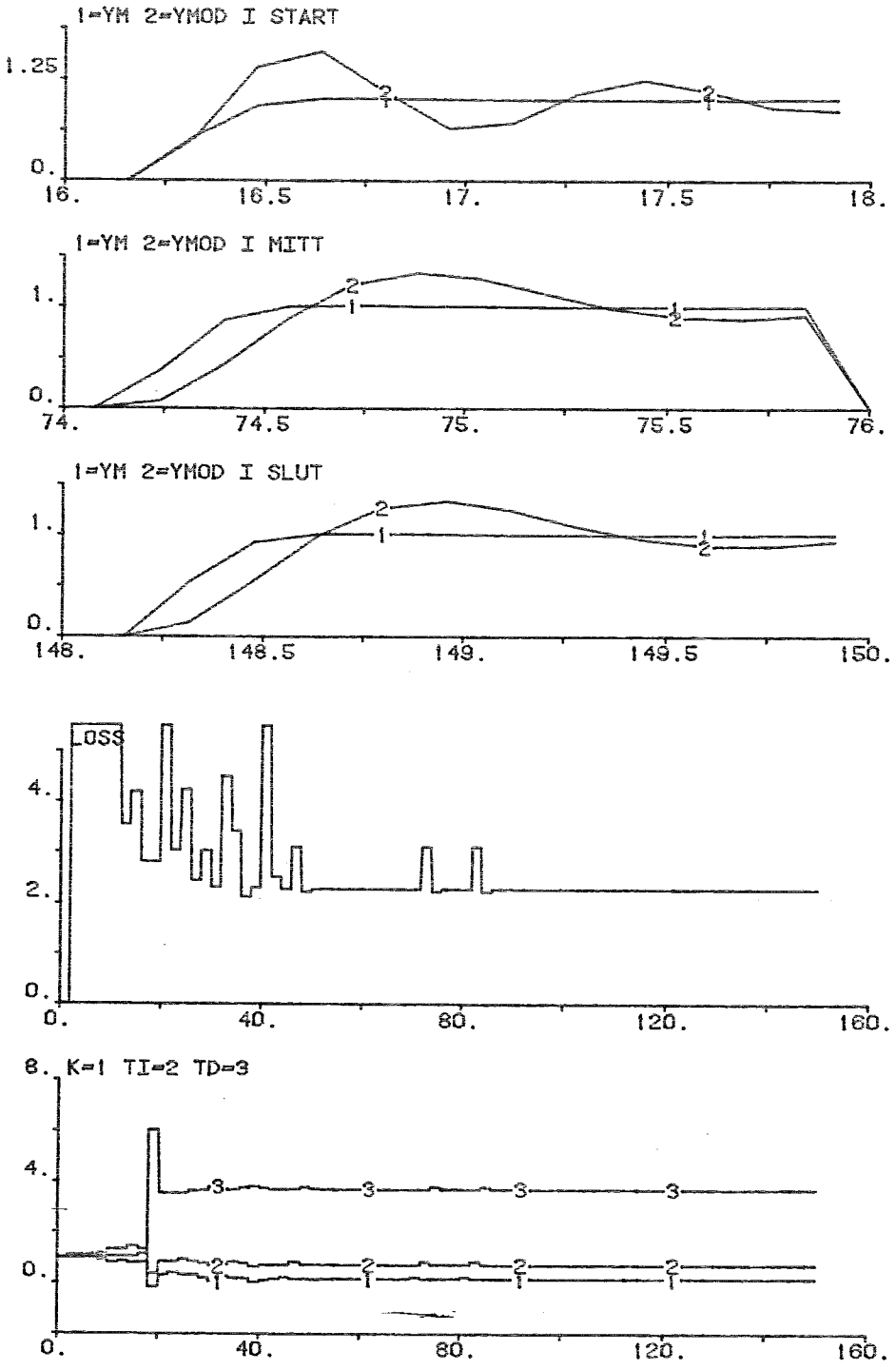

Fig. 5.18. Modellreferens, 3:e ordningen med ISE enligt F.R.-metoden. Startvärden: K=TI=TD=1 Slutvärden: K=0.15 TI=0.73 TD=3.71 LOSS=2.28

Den största skillnaden mellan F.R. och N.M. uppkommer dock för högre ordningens system.

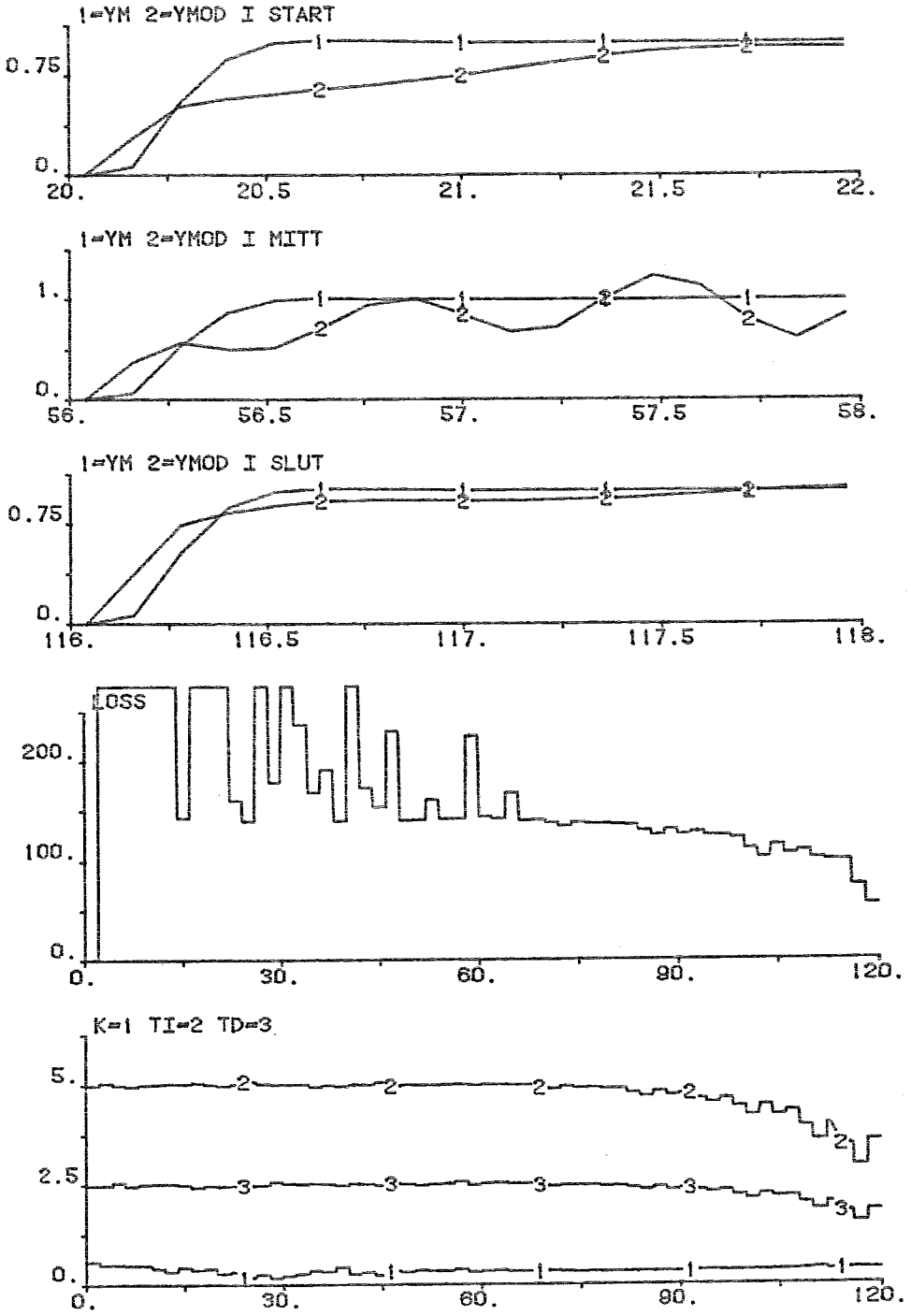

Fig. 5.19. Modellreferens, 4:e ordningen med ITAE enligt N.M.-metoden. Startvärden: K=0.5 TI=5 TD=2.5 Slutvärden: K=0.39 TI=2.94 TD=1.51 LOSS=80.7

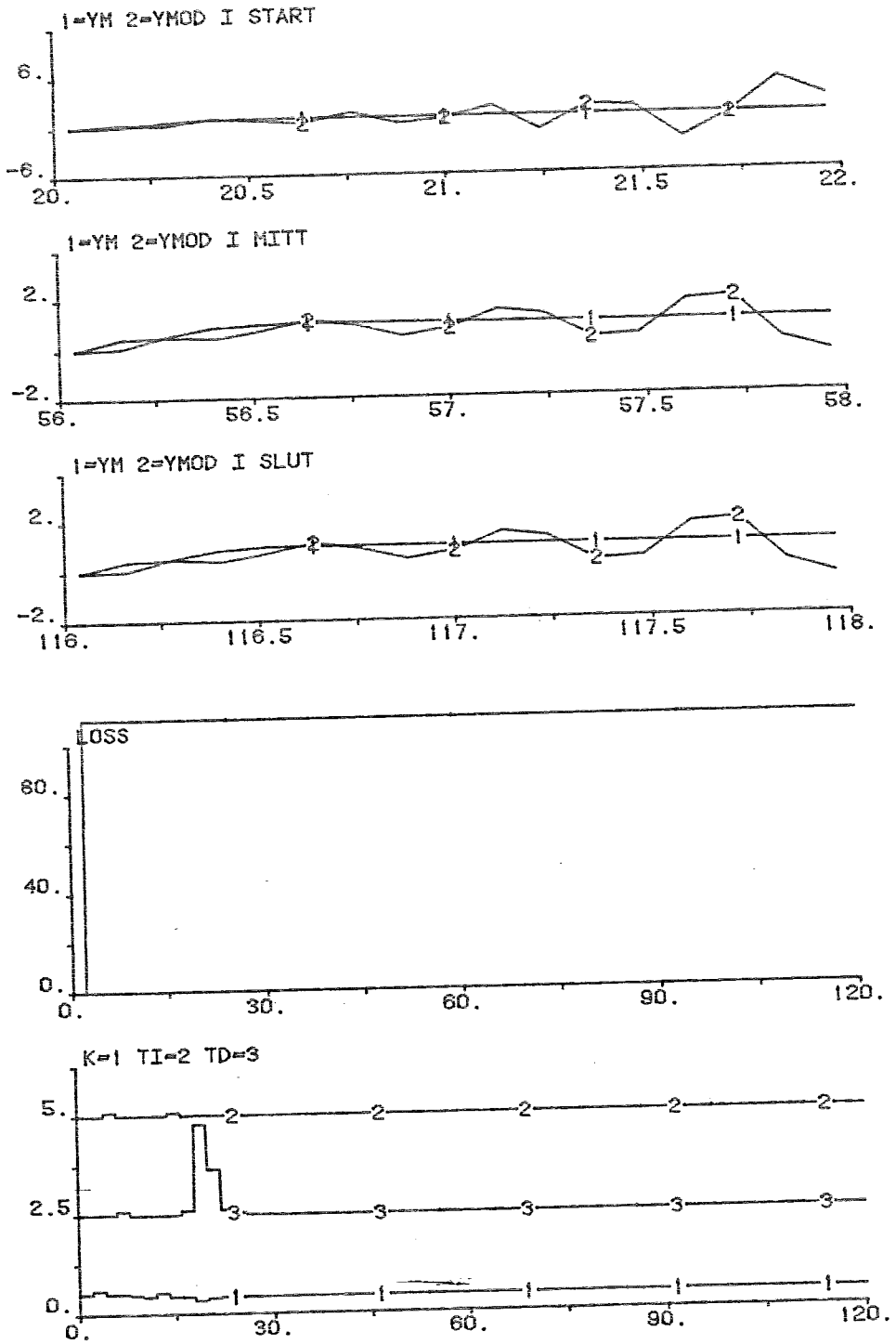

Fig. 5.20. Modellreferens, 4:e ordningen med ITAE enligt F.R.-metoden. Startvärden: K=0.5 TI=5 TD=2.5 Slutvärden: K=0.44 TI=4.99 TD=2.49 LOSS=626.9

I figurerna 5.19 och 5.20 ser man tydligt att fast N.M. har problem fortsätter den fram till bättre och bättre funktionsvärden. F.R. däremot ger upp redan efter 30 s. Den klarar ej av att leta fram minimum och LOSS blir mycket stort.

Vid insignalreferens blir skillnaderna lika stora mellan de bägge metoderna. Nedanstående figurer får tala för sig själva.

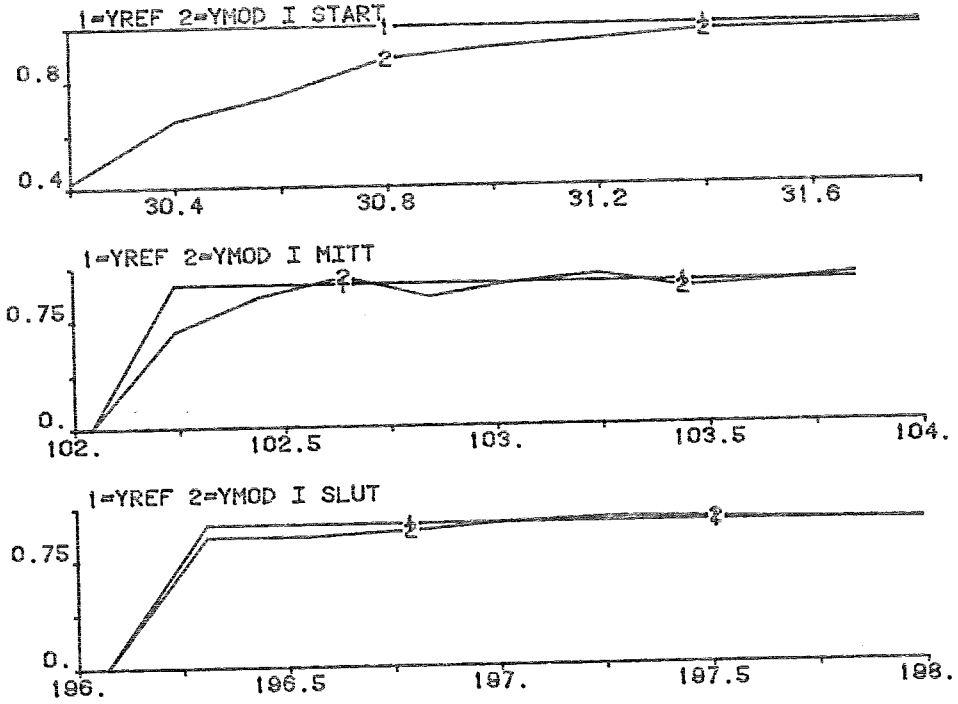

Fig. 5.21 Insignalreferens, 4:e ordningen med ISE enligt N.M.-metoden.

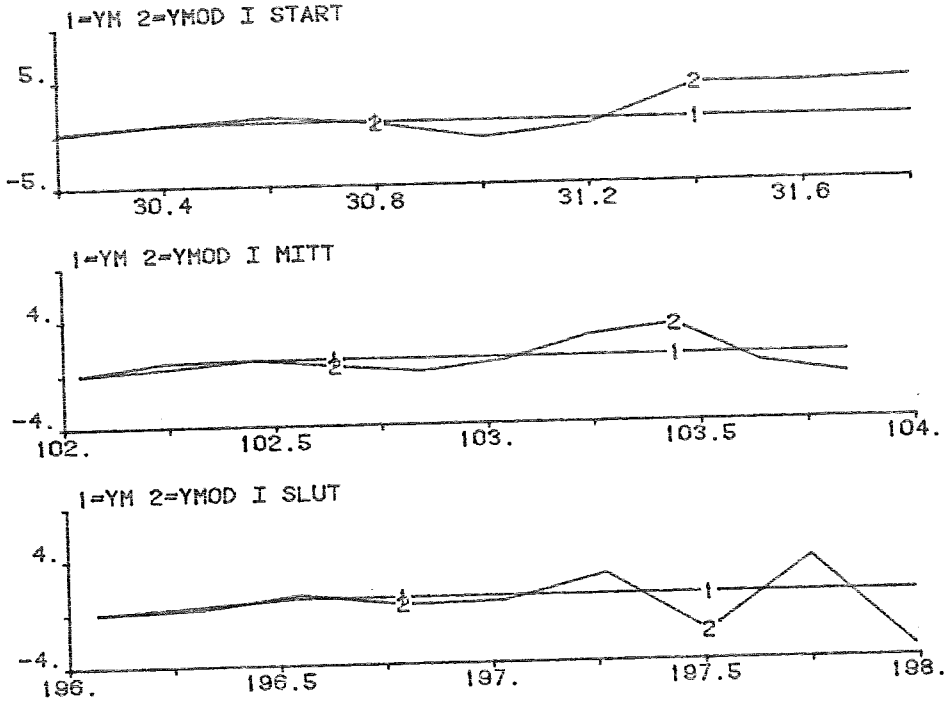

Fig. 5.22 Insignalreferens, 4:e ordningen med ISE enligt F.R.-metoden.

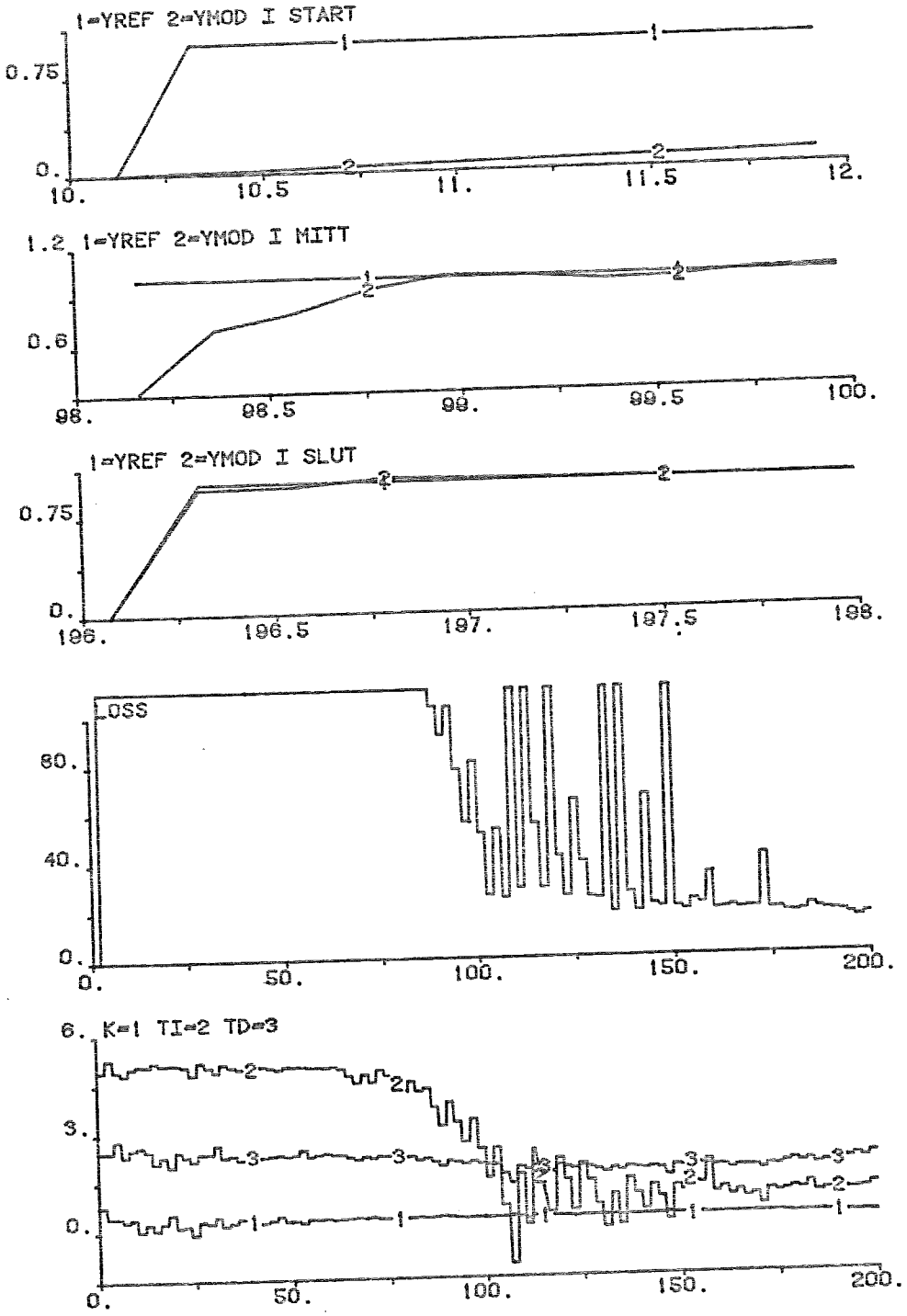

Fig. 5.23 Insignalreferens, 4:e ordningen med ITAE enligt N.M.-metoden. Slutvärde: LOSS=20

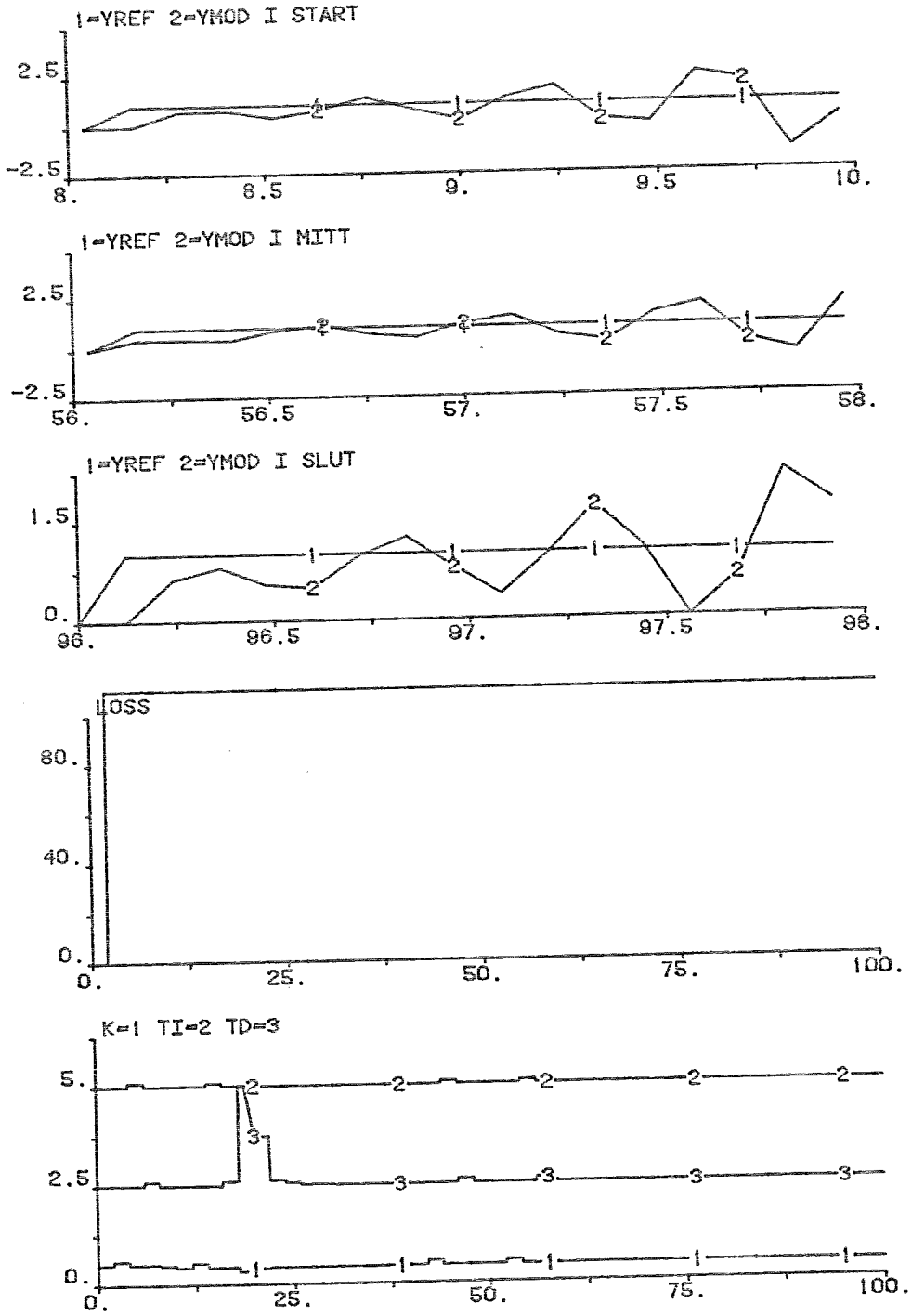

Fig. 5.24 Insignalreferens, 4:e ordningen med ITAE enligt F.R.-metoden. Slutvärde: LOSS=635

6. SLUTSATSER

1. Optimeringsmetoden enligt Nelder-Mead fungerar tillfredsställande. Rutinen är tämligen okänslig för variation i parametrarnas startvärde och är därför lätt att starta. Metoden har visat sig vara klart bättre än Fletcher-Reeves-metoden.

2. Oberoende av vilket kriterium som valts har metoden visat sig pålitlig vid modellreferens med litet värde på ω. För signalreferenssimulering uppstår instabilitetsproblem, vilket förmodligen beror på svagheter i SIMNONkoden.

3. Högre ordningens system medför svårare optimering p.g.a. för få varierbara parametrar i PID-regulatorn.

4. Om söksystemet enligt fig. 1.1 skulle implementeras på mikrodator (typ APPLE) krävs det enligt vår uppskattning ungefär 1000 FORTRAN-rader.

5. Metoden kan vara praktiskt användbar för inställning av industriella regulatorer.

REFERENSER

H.Elmqvist — SIMNON, an interactive simulation program for nonlinear systems, user's manual. Report TRFT-3091, Dept of Autom Control, LTH, Lund 1975.

G.Eriksson — Numerisk analys, FK, Inst. för inf.behandling, Lund 1979.

T.Glad — A program for the interactive solution of parametric optimization problems in dynamic systems. Report TRFT-3084, Dept of Autom Control, LTH, Lund 1974.

D.Himmelblau — Applied nonlinear programming, 1972.

P.Kongstad — Automatisk inställning av PID-regulatorer baserad på optimering och identifiering. Report TFRT-5222, Dept of Autom Control, LTH, Lund 1979.

S.E.Mattsson — Evaluation of a subroutine for Nelder and Mead search. Report TFRT-7150, Dept of Autom Control, LTH, Lund 1978.

J.Miller et al — A comparision of controller. Control Engineering, dec 1967.

J.A.Nelder R.Mead — A simplex method for function minimization. Computer Journal, vol.7 1964.

A.Rovira et al — Tuning controllers for setpoint changes. Instruments & Control Systems, dec 1969.

K.J.Åström — Reglerteori, 1976.