CODEN: LUTFD2/(TFRT-5438)/1–53/(1991)

Transientanalys

Jörgen Bengtsson

Institutionen för Reglerteknik
Tekniska Högskolan i Lund
Juni 1991

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

Document name
Master Thesis

Date of issue
June 1991

Document Number
CODEN:LUTFD2/(TFRT-5438)/1-53/(1991)

Author(s)
Jörgen Bengtsson

Supervisor
Rolf Johansson, Wilhelm Csenki, Stefan Ljung

Sponsoring organisation
Saab Scania AB

Title and subtitle
Transientanalys (Transient response analysis)

Abstract

When measuring transient behavior in discrete time, limitations in memory size impose truncation errors on the computations performed in the subsequent analysis of data. In this paper different approaches to minmize or avoid these errors are investigated.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
Swedish

Number of pages
53

Recipient's notes

Security classification

Förord

Arbetet har utförts på SAAB Flygdivisionens elmiljöavdelning och där vill jag först och främst tacka Wilhelm Csenki och Stefan Ljung för utmärkt handledning.

Jag vill också tacka Rolf Johansson, institutionen för reglerteknik LTH, för idéer, synpunkter och ett värdefullt stöd.

Dessutom ett tack till hela elmiljöavdelningen för hjälpsamhet och ett trevligt bemötande, speciellt då till TUSTP-sektionen vars sällskap jag gärna gjorde.

Linköping i juni 1991

Jörgen Bengtsson

INNEHÅLLSFÖRTECKNING

1. INLEDNING

1.1 Elmiljöavdelningen

Flygplan i atmosfären utsätts för en mängd elektromagnetisk påverkan som kan äventyra flygsäkerheten. På sjuttiotalet förlorades exempelvis ett Viggen-plan efter ett blixtnedslag och på kontinenten förlorades ett militärt flygplan, då det flög över en stark radiosändare som interfererade så starkt med flygplanets elektronik, att piloten inte kunde kontrollera planet. Faktum är också att civila flygplan träffas av en blixt så ofta som ca en gång per 2000 flygtimmar.

För att säkerställa flygsäkerheten har därför varje flygplanstillverkare idag resurser för att kontrollera och förbättra robustheten mot krävande elektromagnetiska miljöer. Hos SAAB Flygdivisionen i Linköping är det elmiljöavdelningen som står för den delen av verksamheten.

Elmiljöavdelningen sysselsätter ca 20 personer. Målet för verksamheten är att i samarbete med konstruktionsavdelningar och underleverantörer säkerställa flygplanens funktion med hänsyn till de elektromagnetiska miljöer de kan utsättas för. Exempel på sådana hot är starka radio- och radarsändare, blixtnedslag, interferens i egen utrustning, elektromagnetisk puls från kärnvapensprängning eller statisk uppladdning.

Mått på robustheten erhålls genom en kombination av provningar och beräkningar. Ofta interpoleras resultatet av en provning i mindre skala med hjälp av beräkningar till en situation som motsvarar en större belastning.

1.2 Kortfattad problemformulering

Vid mätningar i diskret tid när data lagras på ett givet minnesutrymme kan endast ett begränsat antal sampel lagras. Då måste en kompromiss göras mellan samplingsintervall och mättid:

$$\text{mättid} = \text{samplingsintervall} \cdot \text{antal sampel}$$

Detta är ekvivalent med en kompromiss mellan höga och låga frekvenser. Många av elmiljöavdelningens provningar är av transient typ vilket betyder att såväl insignalens som utsignalens amplituder har klingat av till noll(eller under brusnivån) efter en begränsad tid. En begränsad mättid är alltså i princip tillräcklig. I rapporten behandlas endast transienta signaler.

Om den signal man vill mäta på innehåller frekvenser som kräver en hög samplingshastighet kan det inträffa att signalens amplitud ej hunnit minska till brusnivån innan minnesutrymmet tagit slut. Man får en abrupt avklippt signal och erhåller alltså inte information om hela förloppet. Detta innebär också problem, när man vill gå över till frekvensplanet. Den gängse metoden för denna övergång,

Fouriertransformen(FFT), kräver just att signalen ska vara känd i hela det intervall där den inte är noll, eller om den är periodisk, känd under en hel period.

Målsättningen med examensarbetet var att komma runt denna svårighet genom att med hjälp av informationen i den avklippta signalen extrapolera hur den kommer att klinga av till noll. Därigenom skulle ett mera sant frekvensinnehåll erhållas.

1.3 Hur arbetet utförts

Examensarbetet utförde jag på avdelningen i Linköping under perioden mars-juni 1991. I specifikationen av examensarbetet angavs tre möjliga lösningsvägar som alla prövats, dessutom har några andra idéer som dök upp under arbetets gång testats.

De olika idéerna har testats med hjälp av programpaketet MATLAB [1], vilket gjorde att det krävdes endast ett minimum av programmeringsinsats. MATLAB var installerat i VAX-stordatormiljö varför kapaciteten i minne och snabbhet inte innebar någon egentlig begränsning.

1.4 Rapportens utformning

Rapporten redovisar de lösningsansatser som prövats och de resultat som uppnåtts med respektive metod. För läsbarhetens skull finns korta teoretiska avsnitt med, i övrigt hänvisas till referenslitteratur.

I kapitel två utreds närmare själva problemet och förutsättningarna.

I kapitel tre återfinns de olika lösningsvägarna som prövats. De presenteras var för sig med en kort redogörelse för den bakomliggande teorin följt av något eller några exempel med resultat och avslutas med en sammanfattning med kommentarer.

Kapitel fyra innehåller en sammanfattande diskussion av de olika lösningsansatserna vilket leder till de slutsatser och rekommendationer som också finns redovisade här.

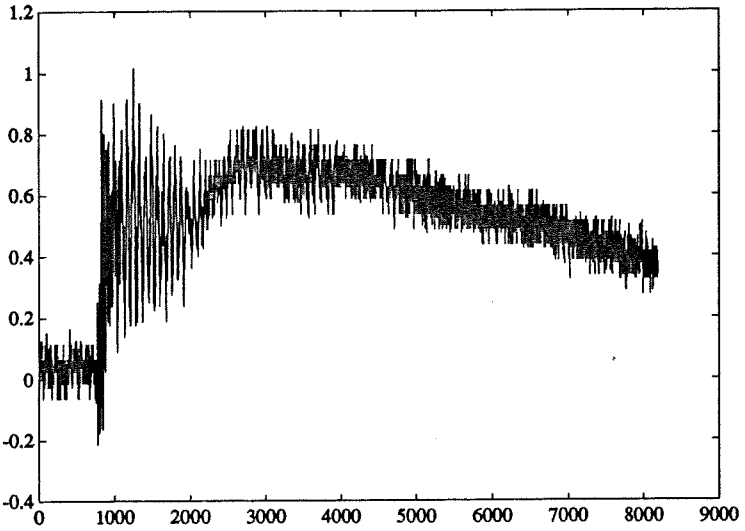
I avsnitt fem återfinns referenslistan. Appendix innehåller journalfiler från MATLAB som redovisar körningar som gjorts till några av de exempel som finns med i rapporten. Dessutom finns listningar av några egna MATLAB-macros som använts.

På grund av att en del av mätresultaten är klassade som hemlig information har alla amplitudskalor från mätningar normerats eller skalats om på något sätt.

2. FÖRUTSÄTTNINGAR

2.1 En uppmätt signal

I figur1 ser vi ett exempel på hur en uppmätt signal kan se ut. Signalen består av 8192 sampel och samplingsfrekvensen är 1GHz. Då blir mättiden 8,192μs. Det är tydligt att mättiden inte är tillräcklig för att fånga hela förloppet. I tidsplanet är det lätt att se att "svansen" fattas men vad händer i frekvensplanet?



Figur 1 En uppmätt signal.

Det egentliga förloppet börjar inte förrän kring sampel nr 800, detta beror på att mätutrustningen triggas något tidigare än själva förloppet. Detta för att början säkert ska fångas. I fortsättningen av rapporten har den delen av signalerna klippts bort. AD-omvandlaren som tillhör mätutrustningen har åtta bitars upplösning.

2.2 Fouriertransform av trunkerade signaler

Betrakta en signal:

$$\begin{aligned} f(t) &= 0 && -\infty < t < 0 \\ f(t) &= g(t) && 0 \le t < \infty \end{aligned} \qquad (1)$$

Det är nu klart att om vi mäter signalen under en begränsad tid, $t_{mät}$, så kan vi inte få den sanna Fouriertransformen som ges av

$$\mathcal{F}(f(t)) = F_{sig}(\omega) = \int_{-\infty}^{\infty} e^{-i\omega t} f(t)\,dt = \int_{0}^{\infty} e^{-i\omega t} g(t)\,dt \qquad (2)$$

utan vi får istället:

$$F_{mät}(\omega) = F_{sig}(\omega) - \int_{t_{mät}}^{\infty} e^{-i\omega t} g(t)\,dt \qquad (3)$$

där den andra termen representerar trunkeringsfelet.

Om funktionen antar tillräckligt små värden utanför mättiden blir felet försumbart, men om signalen ser ut som i figuren ovan blir det avsevärt.

Ett annat resonemang visar tydligare på effekten av trunkeringen. Se den uppmätta signalen som den verkliga multiplicerad med ett rektangulärt fönster:

$$f_{mät}(t) = g(t)h(t) \qquad \text{där } h(t) = 1 \;\; 0 \le t < t_{mät} \qquad h(t) = 0 \;\; \text{annars} \tag{4}$$

vilket i frekvensplanet innebär:

$$F_{mät}(\omega) = G(\omega) * H(\omega) \tag{5}$$

Ur (4) följer också att :

$$| H(\omega) | = | t_{mät} sinc(\omega t_{mät}) | \tag{6}$$

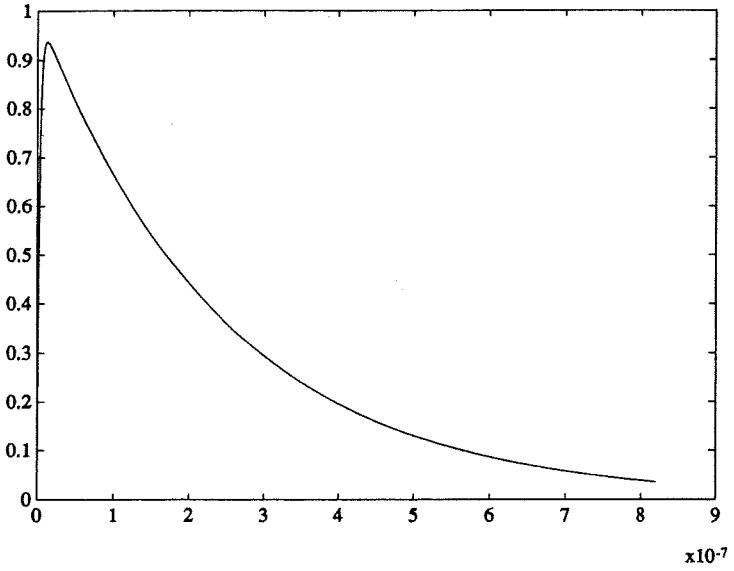
Det betyder att den sanna frekvensgången faltas i frekvensplanet med något som ser ut som en sinc-funktion, det vill säga att vi får en distorsion, en utsmetning av karaktäristikan. Ju längre mättid som används desto mer kommer sinc-funktionen att likna en Dirac-spik och därmed vid faltningen allt mindre distordera den äkta signalen.

Som exempel visar jag en analytisk signal som Fourier-transformerats diskret för några olika mättider. Signalen syns i figur 2 för 0<t<0.8μs och i figur 3 ser man resultatet av FFT tagen på olika långa tidssekvenser av signalen. Även den exakta frekvensgången är inritad. Här syns att frekvensgången liksom planas ut. Lägg också märke till att frekvensupplösningen minskar med minskande mättid. Detta syns i figuren genom att den första frekvenspunkten(efter f=0) flyttar sig längre åt höger.
Exempel 2.1:

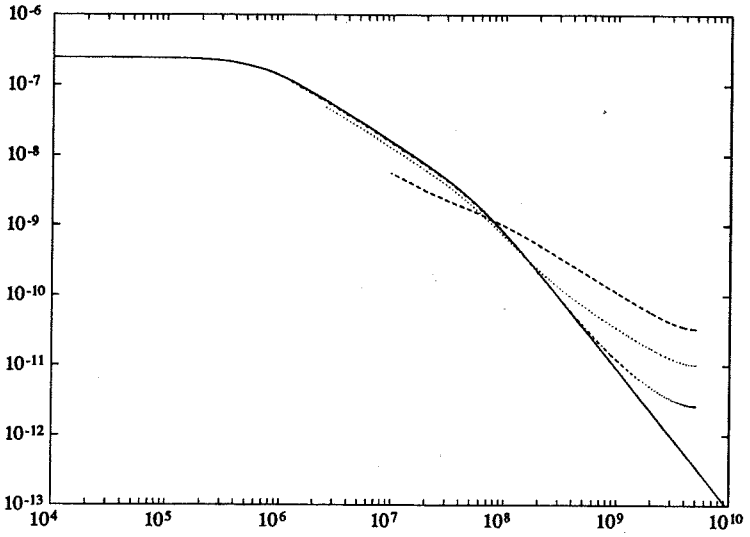
$$f(t) = e^{-\alpha t} - e^{-\beta t} \qquad t \ge 0 \tag{7}$$
$$\alpha = 4.08\ 10^6$$
$$\beta = 3.50\ 10^8$$

Fouriertransform ger enligt (2):

$$F(\omega) = \frac{1}{(i\omega - \alpha)} - \frac{1}{(i\omega - \beta)} \tag{8}$$



Figur 2 Signalen från exempel 2.1



Figur 3 Analytisk frekvensgång(heldragen) tillsammans med FFT för olika långa sekvenser($t_{mät}$=0.1μs streckad, $t_{mät}$=0.4μs punkter, $t_{mät}$=0.8μs punkt-streckad).

2.3 Enkelsidigt fönster som en första metod.

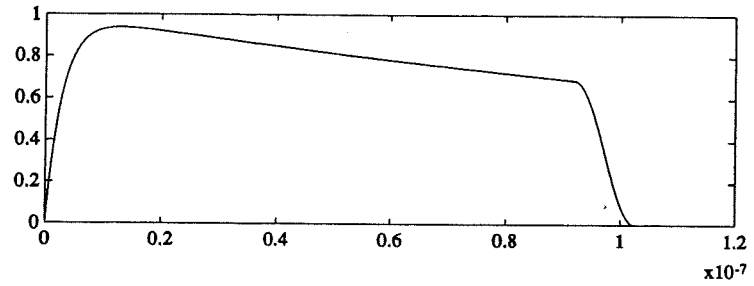
En enkel metod för att mildra effekterna av trunkeringen är att göra signalens avslutning mjukare. Detta kan man åstadkomma genom att multiplicera slutet av signalen med en så kallad fönsterfunktion som på ett mjukt sätt går ner till noll. Naturligtvis får man då inte veta hur signalen hade klingat av, men man introducerar inte så stora fel som man gör om man applicerar en FFT på en abrupt avklippt signal. Det finns en mängd olika fönsterfunktioner som används inom signalbehandlingsområdet, de skiljer sig inte åt kvalitativt och jag kommer här att använda ett enkelsidigt Hanningfönster [2]:

$$w(k) = 0.5(1-\cos(\pi+\pi\frac{k}{N-1})) \qquad k=0..N-1 \qquad (9)$$

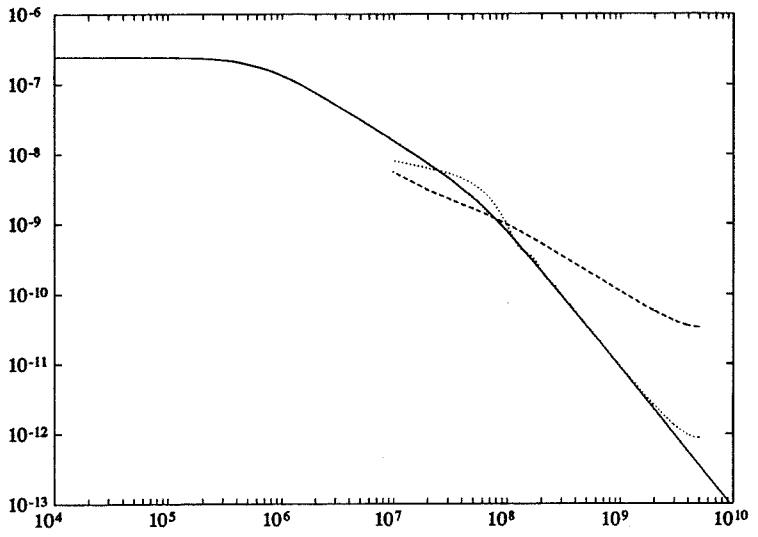
Signalen blir nu:

$$f_{wind}(k)=f(k) \qquad 0\le k<k_{bryt}$$
$$f_{wind}(k)=f(k)w(k-k_{bryt}) \qquad k_{bryt}\le k<antal\ sampel \qquad (10)$$

Vi tittar på signalen från figur 2 igen. I figur 4 syns signalen för $t_{mät}$=0.1μs med ett Hanningfönster på de sista 10% av samplen. I figur 5 syns sedan den analytiska frekvensgången(heldragen) tillsammans med FFT av den fönsteravslutade signalen(prickad) och FFT av den abrupt klippta kurvan med samma längd(streckad). Vi ser att den fönsteravslutade signalens FFT mer liknar den analytiska frekvensgången.



Figur 4 En fönsteravslutad signal



Figur 5 Effekten av mjuk avslutning.

2.4 Målsättning

Vi har sett att den begränsade mättiden klipper vår signal i tidsplanet och distorderar karaktäristikan i frekvensplanet. Vi såg också att en första metod kan vara att tvinga signalen ner till noll på ett mjukt sätt. Denna metod kan vara tillräcklig om den avklippta svansen ej är för stor. Har signalen klippts vid en tidig tidpunkt i förloppet är dock metoden behäftad med stora brister. Signalen kanske i själva verket fortsätter med ett oscillerande förlopp.

Målsättningen är att utröna, om vi kan använda den information vi har från den korta mättiden effektivare eller om vi på något annat sätt kan mildra effekterna av trunkeringen.

Om vi tittar på signalen i figur 1 igen, så ser vi, att de högfrekventa komponenterna klingat av innan mättidens slut. Om det är fallet att vi bara är intresserade av dessa så bör vi separera låg- och högfrekventa delar. Detta utvecklas i 3.1 .

Kanske vore det möjligt att ur den begränsade datamängden extrapolera det fortsatta förloppet och på detta sätt skaffa sig en konstlad längre mättid. Detta utreds i 3.2 och 3.3 .

Extrapoleringen av det fortsatta förloppet underlättas om insignalen är känd över ett längre intervall och en överföringsfunktion kan skattas. Se 3.4 .

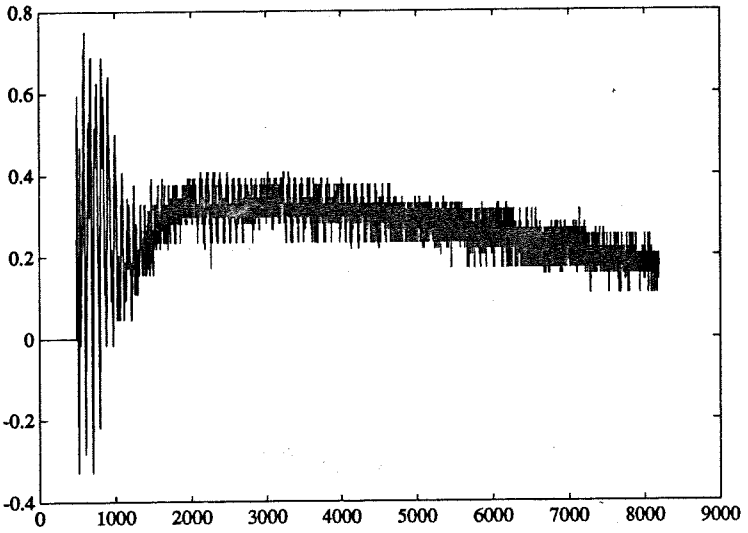
Slutligen diskuteras i 3.5 hur man skulle kunna ändra mätsituationen.

3. LÖSNINGSANSATSER

3.1 Separationsfiltrering

3.1.1 Principen

Vi tittar på en uppmätt signal, figur 6 nedan. Om vi nu tittar lite närmare, så ser vi att ett högfrekvent förlopp ligger överlagrat på en betydligt långsammare svängning. Vi kan också se att de högfrekventa komponenternas amplitud försvinner i brusnivån gott och väl innan mättidens slut. I det fall då vi endast är intresserade av de högfrekventa delarna är alltså mättiden tillräcklig. Det förefaller då vettigt att separera signalen i en hög- och en lågfrekvent del. Detta låter sig göras med filtrering. Genom att lågpassfiltrera signalen och sedan punktvis subtrahera resultatet ifrån den ursprungliga signalen får man två komponenter.



Figur 6 En uppmätt signal, horisontella axeln anger sampelnummer.

Filtreringen innebär dock vissa svårigheter jämfört med traditionell stationär filtrering. Vi kan inte tillåta någon fasförskjutning, som i så fall skulle distordera kurvformen. Transienter från filtret får ej heller tillåtas ha någon inverkan.

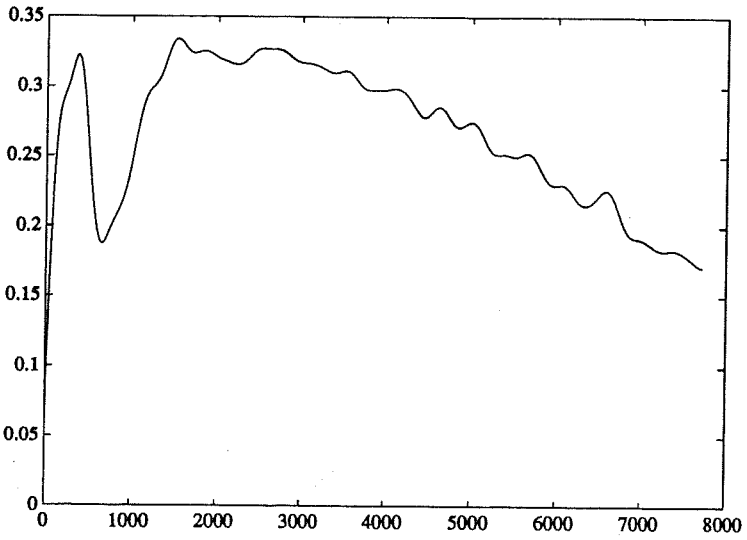
3.1.2 Filterdimensionering

Ett digitalt filter implementeras lätt i MATLAB, jag hänvisar till MATLABs manual [3]. Jag valde ett filter av Chebyshev-typ med brytfrekvens ca 1MHz och 20dB dämpning i spärrbandet.

3.1.3 Filtrering av transienta signaler

För att åstadkomma fasrenheten filtreras data först fram- och sedan baklänges. För att undvika transienter i filtret speglas en bit av data i båda ändar före filtrering. Dessa åtgärder finns redan implementerade som ett macro i MATLAB, **filtfilt** [3].Macrot ligger i en s k m-fil som man som användare har tillgång till. Jag modifierade filen och gjorde ett eget macro. Den enda ändringen som behövdes var att spegla längre sekvenser i ändarna för att minimera transienter.

I figur 7 ser vi resultatet efter lågpass-filtrering.

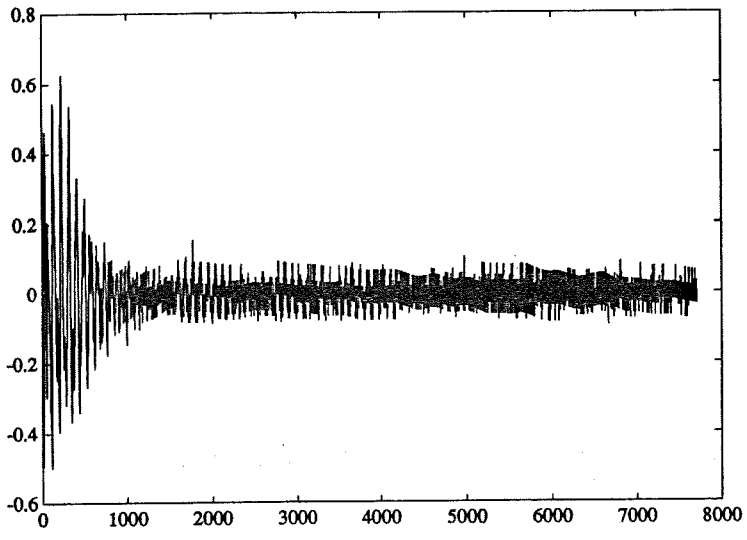


Figur 7 Den lågfrekventa delen av signalen.

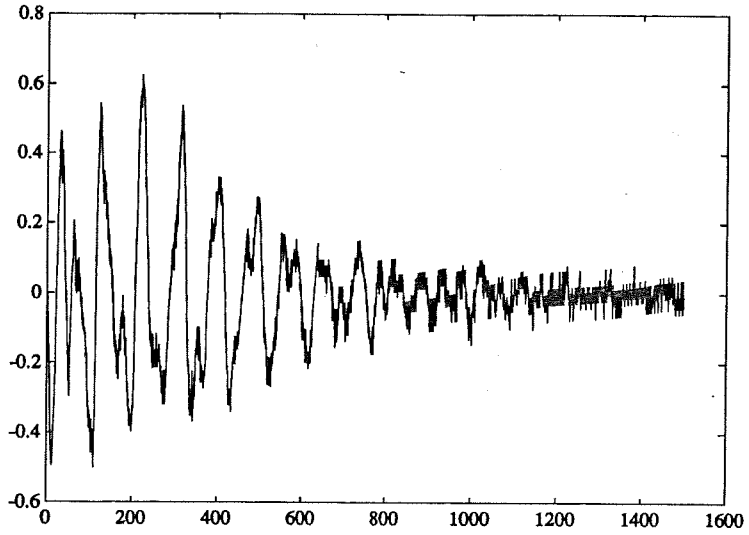
3.1.4 Separering

Vi kan nu subtrahera den lågfrekventa delen från mätsignalen och erhåller då den högfrekventa delen. Som vi ser i figur 8 klingar denna av till under brusnivån under mättiden och FFT kan appliceras utan nämnvärt trunkeringsfel. För den lågfrekventa delen kvarstår dock problemet som tidigare.

I figur 9 visas endast de första 1500 samplen av signalen i figur 8, här syns tydligare att signalen består av en summa av dämpade svängningar.



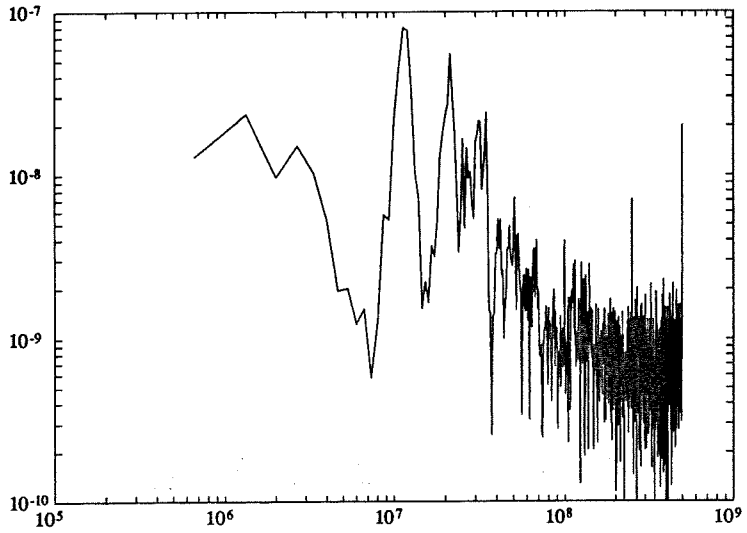
Figur 8 Den högfrekventa delen av signalen.



Figur 9 De första 1500 samplen av den högfrekventa delen.

3.1.5 Signalform - förväntat frekvensinnehåll

Här följer en liten diskussion om vilka kurvformer man kan vänta sig. Om vi tar FFT på signalen i figur 9 får vi resultatet i figur 10.



Figur 10 Den högfrekventa delens frekvensinnehåll. Den horisontella axeln är graderad i Hz.

Vi ser resonanstoppar i området 10MHz och uppåt. Den största av dessa ligger strax över 10MHz vilket motsvarar $\lambda/2 \approx 15$m som är flygplanets längd. De andra topparna svarar mot andra resonanser t ex vingspets till vingspets eller vissa kabellängder. Vi kan egentligen inte vänta oss några resonanser under 10MHz på grund av flygplanets fysiska dimensioner.

Den avsevärt långsammare svängningen (figur 7) härrör från en mer direkt koppling till insignalformen som vi kommer att se i avsnitt 3.4

3.1.6 Sammanfattning

För den typ av provning varifrån signalen i figur 6 härrör löser separationsfiltrering vårt problem i det fallet då vi endast är intresserade av de högfrekventa resonanserna. Naturligtvis är det också en förutsättning att dessa klingat av under mättiden, men detta var fallet i alla de tiotalet uppmätta signaler från just denna typ av provning som jag hade tillgång till.

Om intresset även omfattar de lågfrekventa svängningarna kvarstår problemet, men separationsfiltreringen är lämplig att använda som ett första steg varefter man kan jobba vidare endast med LF-delen.

3.2 Tidsserie som signalmodell

I det här avsnittet använder jag tidsseriemodeller för att identifiera en trunkerad signal och extrapolera en fortsättning. Jag gör antagandet att signalen kan beskrivas som en reell summa av komplexa svängningar.

3.2.1 Sambandet tidsseriemodell - differentialekvation

Jag ansätter följande kontinuerliga modell för min signal:

$$f(t) = \sum_{n=1}^{m} r_n e^{-s_n t} \qquad t \geq 0 \tag{1}$$

där s_n och r_n kan vara komplexa. Jag kommer att kalla s_n för kontinuerliga poler och r_n för residyer. Signalen antas alltså bestå av en summa av komplexa svängningar, vilket är en tillräckligt generell modell för många tillämpningar. Vi kan också se att signalmodellen (1) har precis samma form som de transienta lösningarna till en homogen linjär differentialekvation.

Eftersom mätningarna sker vid diskreta tidpunkter behöver vi en diskret modell. Antag att samplingsintervallet är h och låt k betyda numrering av enskilda sampel. Då kan vi beskriva vår signal i sampeltidpunkterna som

$$\begin{aligned} t_k &= kh \qquad k=0,1,\ldots,N \\ f(t_k) &= \sum_{n=1}^{m} r_n e^{-s_n kh} \end{aligned} \tag{2}$$

Om vi sedan inför

$$p_n = e^{-s_n h} \tag{3}$$

får vi i diskret tid:

$$f_d(k) = \sum_{n=1}^{m} r_n p_n^k \tag{4}$$

Analogt får alltså signalmodellen i diskret tid samma form som lösningen till en homogen differensekvation. De diskreta polerna representeras av p_n och residyerna blir desamma. Observera att de diskreta polerna beror av samplingshastigheten.Differensekvationen som (4) löser ser ut som :

$$y(k) = -a_1 y(k-1) - a_2 y(k-2) \ldots - a_m y(k-m) \tag{5}$$

och det är klart att de diskreta polerna p_n är lösningar till (5):s karaktäristiska ekvation:

$$\begin{aligned} &A(z^{-1}) = 0 \\ &\text{med} \quad A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_m z^{-m} \end{aligned} \tag{6}$$

Signaler enligt (1) kan alltså beskrivas, i samplingstidpunkterna, med en differensekvation enligt (5). Observera att (5) kan skrivas som:

$$A(z^{-1})y(k)=0 \tag{7}$$

vilket leder oss över till tidsseriemodeller.

3.2.2 Tidsseriemodeller

Utan mätbrus skulle problemet att identifiera en signal vara trivialt det skulle endast behövas lika många mätpunkter som parametrar för att exakt lösa problemet. I närvaro av mätbrus måste jag göra ett antagande om brusets egenskaper för att sedan kunna göra en bästa skattning enligt något kriterium. Ett vanligt antagande är att mätningarna störs av vitt brus vilket leder till modellen :

$$A(z^{-1})y(k)=e(k) \qquad E(e(k)e(k))=\sigma^2 \tag{8}$$

Denna modellstruktur brukar kallas AR-modeller(AutoRegressive). Här vill jag betona att denna beteckning främst förekommer i sammanhang med stationära signaler d v s signaler på vilka begynnelsevärdenas inverkan har upphört. Detta är INTE fallet för de uppmätta signalerna i denna undersökning, utan här är tvärtom det stationära förloppet helt ointressant.

Ofta stämmer inte antagandet om vitt brus utan en bättre modell är:

$$A(z^{-1})y(k)=C(z^{-1})e(k) \tag{9}$$

som brukar kallas ARMA-modell(AutoRegressive Moving Average).

3.2.3 Skattning av parametrar

Vi börjar med att beskriva hur A-polynomets koefficienter i (8) kan skattas, detta ger oss sedan r_n och s_n i (1).
Om vi hade $n=2m$ stycken brusfria($\sigma=0$) observationer från ett system som i (8), vore det trivialt att skatta parametervektorn θ ur relationerna:

$$\begin{bmatrix} y(m) & \dots & y(1) \\ y(m+1) & \dots & y(2) \\ \dots & \dots & \dots \\ y(n-1) & \dots & y(n-m) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \dots \\ a_m \end{bmatrix} = \begin{bmatrix} y(m+1) \\ y(m+2) \\ \dots \\ y(n) \end{bmatrix} \tag{10}$$

$$\text{eller} \qquad \Phi\theta = Y$$

$$\Rightarrow \hat{\theta} \equiv [\hat{a}_1 \ \hat{a}_2 \ \dots \hat{a}_m]^T = \Phi^{-1}Y = \theta$$

I närvaro av mätbrus är det lämpligt att göra fler observationer, $n>2m$, och det traditionella tillvägagångssättet att skatta parametrarna är sedan minstakvadrat-metoden som minimerar förlustfunktionen:

$$V(\hat{\theta}) = \sum_{k=1}^{antal\ sampel} (Data(k) - Modell(\hat{\theta},k))^2 \qquad (11)$$

där Data innehåller våra observationer. För AR-modellen i (8) blir minstakvadrat-skattningen med beteckningar enligt (10),[4]:

$$\hat{\theta} = (\Phi^T \Phi)^{-1} \Phi^T Y \qquad (12)$$

Vi har nu en skattning av θ, d v s av A-polynomets koefficienter. Polerna p fås som lösningarna till (6) och kan räknas om till kontinuerlig tid enligt (3). Slutligen kan vi få residyerna genom att lösa ett begynnelsevärdesproblem:

$$\begin{bmatrix} 1 & \dots & 1 \\ p_1^2 & \dots & p_m^2 \\ \dots & \dots & \dots \\ p_1^n & \dots & p_m^n \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ \dots \\ r_m \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \dots \\ y_n \end{bmatrix} \qquad (13)$$

Detta överbestämda ekvationssystem kan också lösas i minstakvadrat-mening och i AR-fallet är vi därmed klara.

I EMC-sammanhang nämns ofta Pronys (första) metod, det är emellertid ekvivalent med att använda metoden ovan, se [5].

Om vi istället ansätter en ARMA-modell kan vi göra på samma sätt, utom vid skattningen av A-polynomet då en annan algoritm måste användas för att undvika systematiska fel i skattningarna.

I MATLAB finns kommandon implementerade för att skatta parametrar i en rad olika modellstrukturer. För AR-fallet finns t ex en operator, \, för matrisdivision som löser ekvationssystem som (10) i minstakvadrat-mening, d v s som i (12). Det finns även andra varianter, t ex Yule-Walker, Burg. För skattning av ARMA-parametrar använder MATLAB en iterativ Gauss-Newton algoritm, se [6].

3.2.4 Validering av skattning

För att kunna bedöma hur en skattning lyckats kan man titta på förlustfunktionens storlek. En bättre uppfattning om riktigheten och rimligheten kan man få genom att helt enkelt rita upp den signal som beskrivs av de erhållna parametrarna och modellen. Signalen kan erhållas i ett godtyckligt tidsintervall t>0, så att man förutom likheten med originalet under mättiden även kan bedöma om fortsättningen är rimlig.

3.2.5 Från parametrar till frekvensplanet

Det är klart att när poler och residyer bestämts kan en analytisk Fouriertransform genomföras enligt:

$$Y(\omega)=\sum_{n=1}^{m}\frac{r_n}{i\omega-s_n} \qquad (14)$$

Detta innebär att om skattningen av parametrarna gått bra under mättiden, så har signalens egenskaper fångats och ett frekvensinnehåll som inte störs av den korta mättiden kan erhållas. Det innebär också en avsevärd datakompression, eftersom signalen kan representeras fullständigt med hjälp av m poler och lika många begynnelsevärden, d v s $2m$ tal.

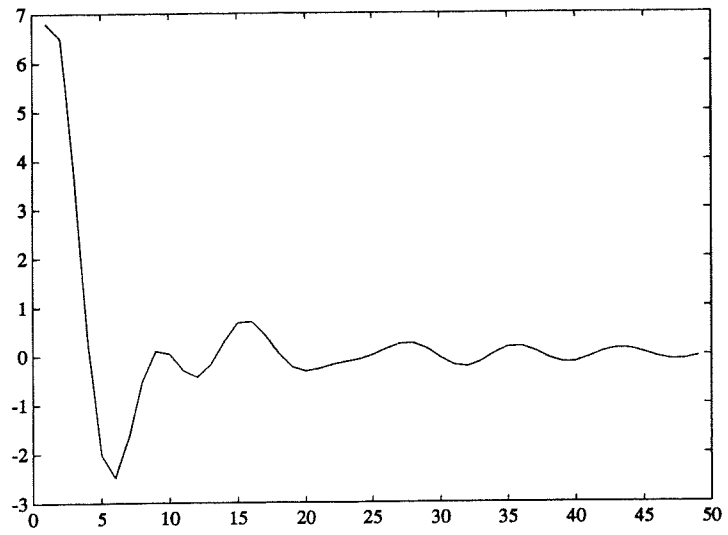
3.2.6 Bruskänslighet

Ovan har beskrivits metoder att skatta parametrar från brusiga mätningar i [4] finns utredningar om konvergensegenskaper etc under brus. Här skall vi se konkret hur brus påverkar möjligheten att identifiera. För att belysa bruskänsligheten använder jag ett exempel som återfinns i[5]. Till tre komplexa dubbelpoler adderas vitt brus med förhållandevis mycket låg amplitud.

Ex 3.1
Beteckningar enligt modellen i (1).

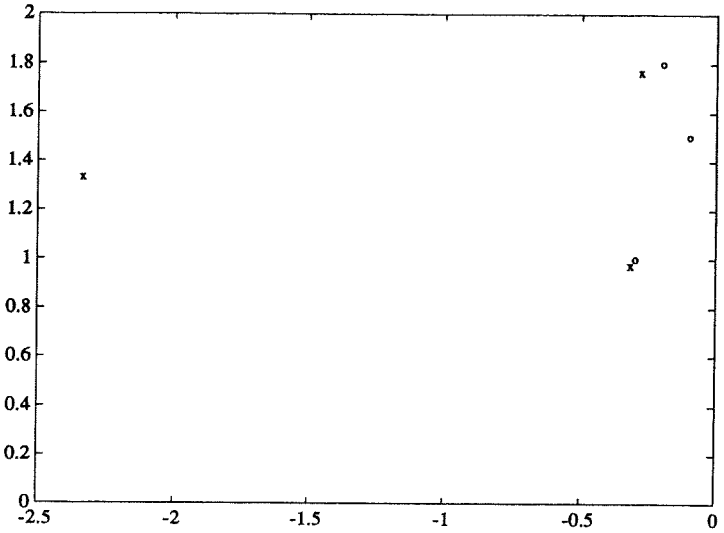
$$\begin{aligned} s_{1,2}&=-0.1\pm i1.5 & r_{1,2}&=0.4\pm i0.2\\ s_{3,4}&=-0.2\pm i1.8 & r_{3,4}&=1.0\pm i0.5\\ s_{5,6}&=-0.3\pm i1.0 & r_{5,6}&=2.0\pm i1.0\\ h&=0.50 & & \\ \sigma&=10^{-3} & & \end{aligned} \qquad (15)$$

Signalen syns i figur 11 med brus.



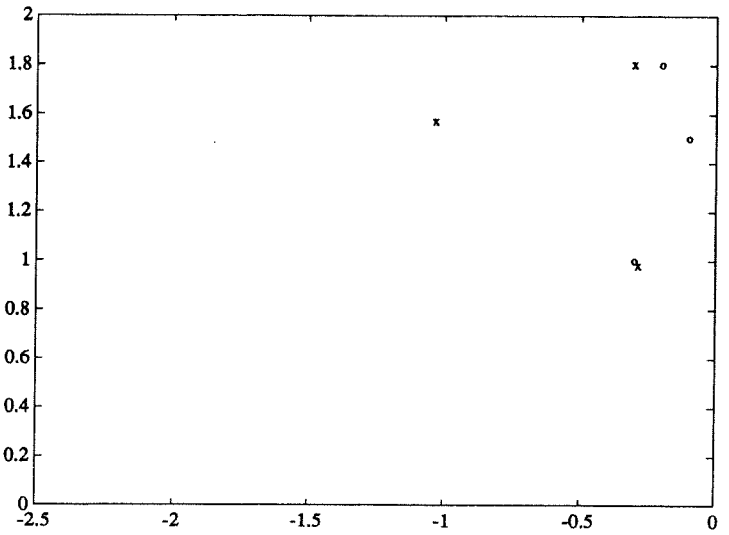
Figur 11 Signalen i exempel 3.1.

Vi använder AR-modellen och en minstakvadrat-skattning på signalen. Resultatet av skattningen syns i fig12. Som synes misslyckas metoden kapitalt trots den mycket låga brusnivån.



Figur 12 Poldiagram som visar resultatet av identifiering med en AR-modell av ordning sex. Ringarna anger signalens egentliga poler och kryssen de poler som är resultatet av skattningen.

Kanske är det AR-modellen som inte räcker till, vi prövar med en ARMA-modell,resultat i figur 13:

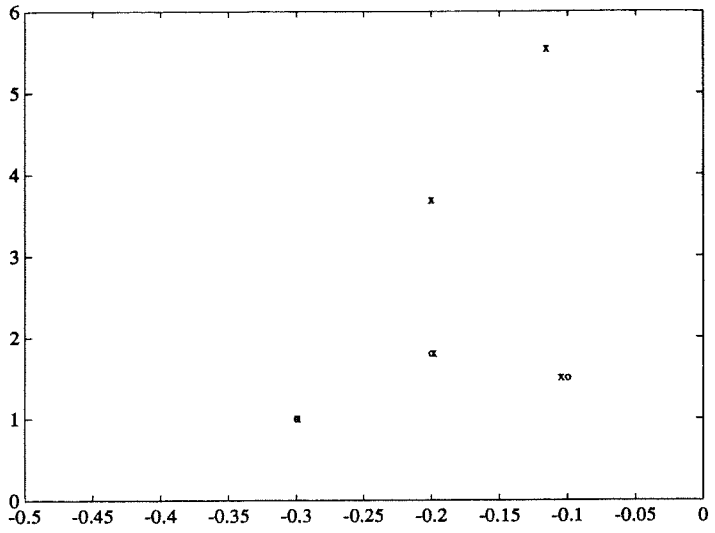


Figur 13 Som figur 12, men en ARMA-modell har använts.

Identifieringen lyckas tydligen inte så bra trots att modellen och förutsättningarna är korrekta.

3.2.7 Ordningstal

I föregående avsnitt var resultatet nedslående. Om vi istället prövar att hitta tio poler till en AR-modell får vi ett bättre resultat, i den meningen att de sex "riktiga" polerna hittas med mycket större noggrannhet. Se figur 14.



Figur 14 Resultat av skattning med AR-modell, ordning tio.

Det ser alltså ut att vara lämpligt att använda fler parametrar i skattningen än vad som egentligen finns. Då får man en beskrivning av en signal som ligger mycket nära den verkliga men observera att detta inte betyder att polerna som hittats "existerar".

3.2.8 Modellreduktion

Föregående avsnitt visade att en överidentifikation förbättrade möjligheten att hitta de verkliga polerna med större noggrannhet. Dock kvarstår svårigheten att veta hur många och vilka poler som är de releventa samt att reducera ner modellen på ett "bra" sätt till den ordning man anser trolig.En metod för att reducera högre ordningens **tillstånds**-modeller till lägre finns beskriven i [7]. För att kunna utnyttja den metoden behöver jag alltså en beskrivning på tillståndsform. Antag att min signal är ett ändligt långt impulssvar från ett linjärt system. Eftersom ett linjärt systems impulssvar fullständigt beskriver systemet kan jag då direkt sätta upp en tillståndsbeskrivning:

$$h(k)=h_k=f_d(k)$$

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ \dots \\ x_N(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & & \dots & & \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ \dots \\ x_N(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ \dots \\ 0 \end{bmatrix} u(k) \qquad \textbf{(16)}$$

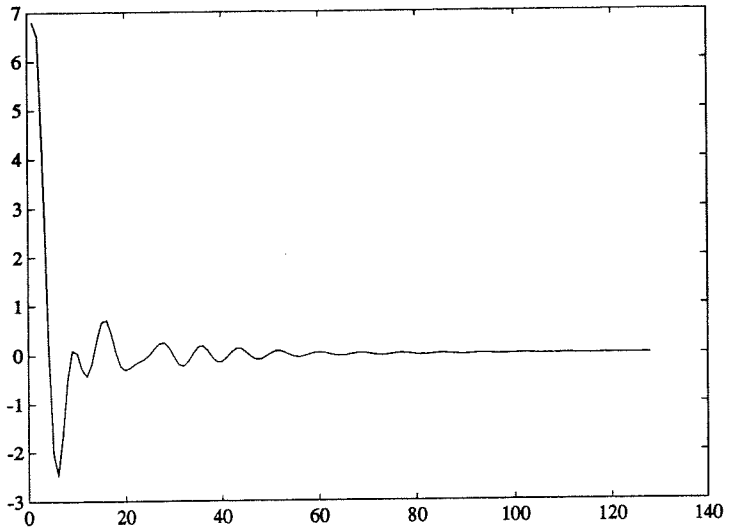
$$y(k) = \begin{bmatrix} h_0 & h_1 & h_2 & \dots & h_{N-1} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ \dots \\ x_N \end{bmatrix}$$

Nu är de signaler som är aktuella inga impulssvar, men både insignalen och utsignalen är kontinuerliga och jag kan därför använda mig av metoden.

Jag tar alltså min utsignal, ser den som impulssvaret från ett linjärt system och sätter upp en N:te ordningens tillståndsrepresentation enligt(16). Med hjälp av metoden i[7] transformeras denna representation till en balanserad realisation. I denna representation kan man få ett mått på vilka tillstånd(poler) som har störst inflytande över in-utsignalsambandet. Med hjälp av detta mått avgör jag hur många tillstånd jag vill behålla och reducerar ner beskrivningen till denna ordning. Polerna erhålls sedan som egenvärdena till systemmatrisen.

I MATLABs Control System Toolbox [8] finns macron för detta förfarande; **dbalreal** för att få den balanserade realiseringen och en vektor med måttet på tillståndens "vikt" samt **dmodred** för att reducera systemet till en önskad lägre ordning.

Jag demonstrerar metoden på exempel 3.1 ovan. Jag utgår ifrån att sytemets ordning är mindre än sexton, identifierar därför med en AR -modell med ordning sexton. Därefter simulerar jag fram signalen för k=0,1,..,127 med hjälp av min modell, figur 15:

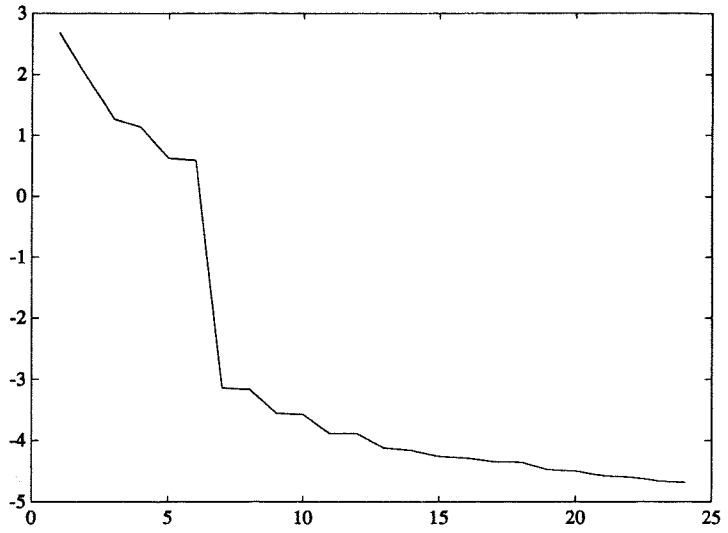


Figur 15 Den simulerade signalen från framidentifierade poler och nollställen.

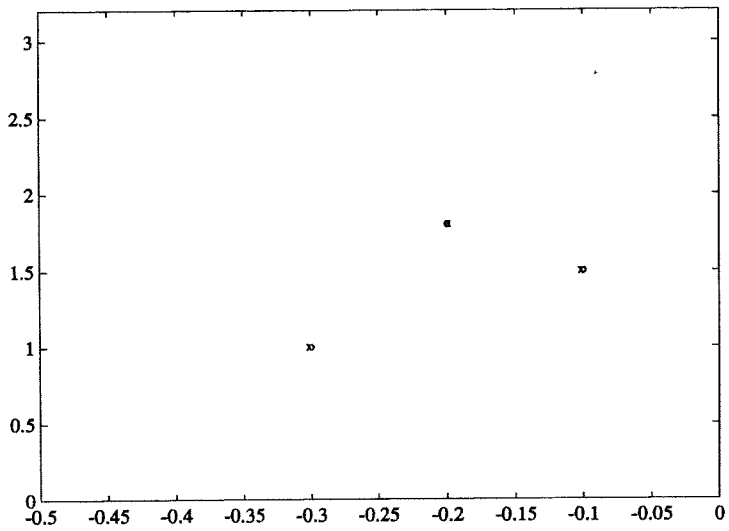
Denna sekvens använder jag nu som impulssvar och går över till en tillståndsform med 128 tillstånd. I figur 16 är vektorn med vikten hos tillstånden illustrerad med logaritmisk skala. Utseendet indikerar klart att en modell med ordning sex tycks vara den underliggande strukturen.

Nu kan jag reducera till sjätte ordningens system och vi ser i figur 17 att det lyckas bra i detta exempel att hitta de sex polerna.

Fördelen med denna metod är att jag även får hjälp med att avgöra hur många poler som är relevanta vilket sällan är a priori kunskap.



Figur 16 Vikten för de 24 första tillstånden. Logaritmisk skala.



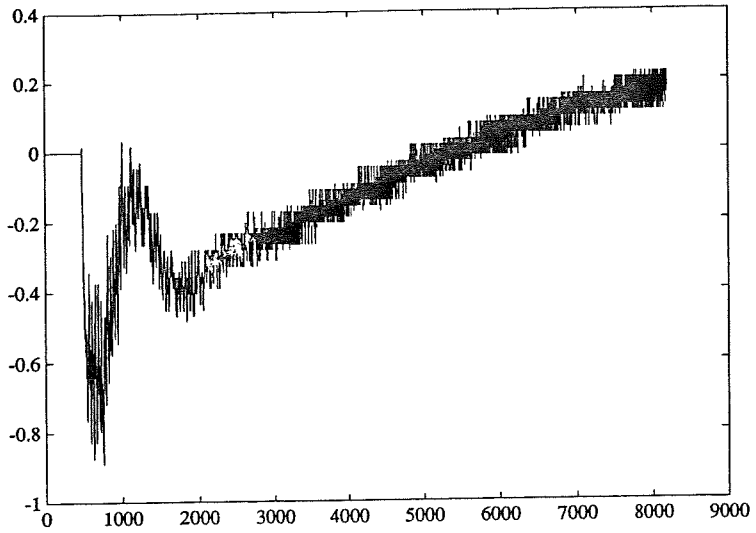
Figur 17 Originalets poler(ringar) och skattade(kryss) efter identifikation och modellreduktion.

3.2.9 Test på uppmätta signaler

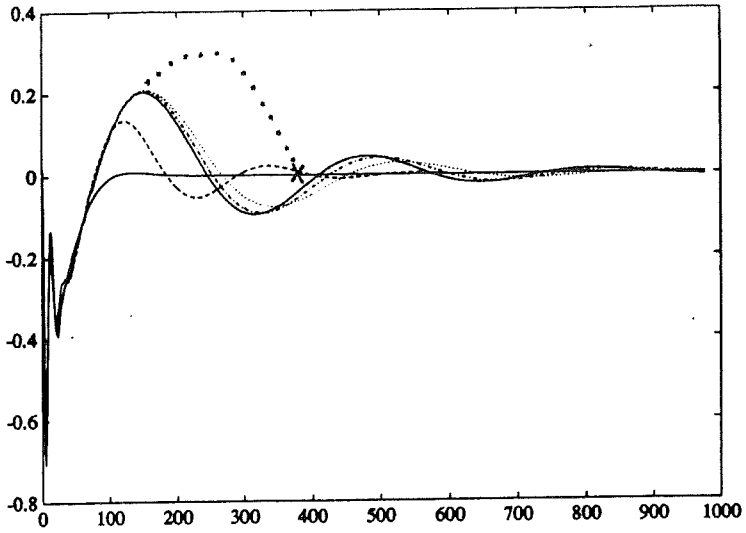
I 3.2.8 har vi sett att det är möjligt åtminstone för simulerade exempel att identifiera fram parametrar för den generella modellen i (1). Detta är möjligt även då mätningarna störts av brus och då ordningstalet är okänt. Nu är tiden mogen att pröva på uppmätta signaler.

Enligt 3.1 separerar jag signalen och arbetar här endast vidare med den lågfrekventa delen. Dessutom måste jag för identifierbarhetens skull decimera data[9]. I figur 18 syns signalen före filtrering och decimering.

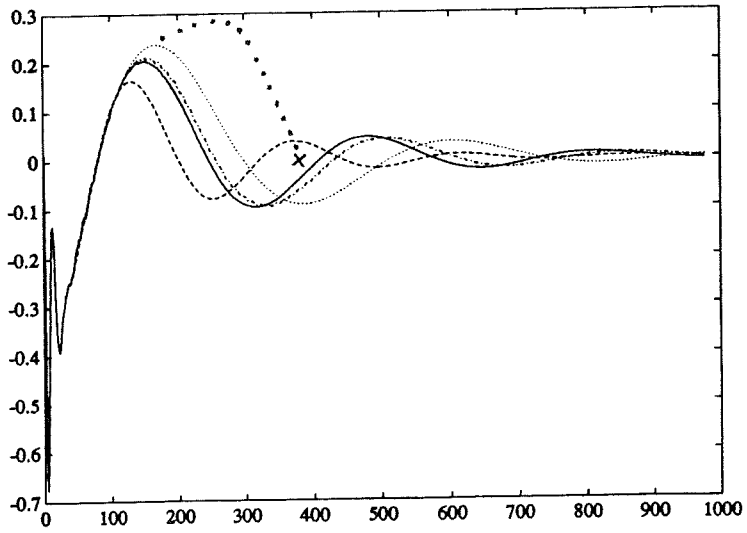
Denna signal har identifierats med AR- och ARMA-modeller. Mätdata fanns även tillgänglig från samma experiment då det upprepats med en lägre samplingshastighet och längre mättid. Det finns ingen anledning att tro att det längre förloppet skulle skilja sig åt i de båda fallen, följaktligen sitter vi med facit i hand och kan jämföra skattningen med en sann fortsättning. Se figur 19 och 20.



Figur 18 Signalen som använts, före filtrering och decimering.



Figur 19 Resultatet för olika AR-modeller, (ordning 12-heldragen, 24-streckad, 48-heldragen, 46-punktstreckad, 36-prickad), jämförda med originalet som förts in för hand.



Figur 20 Resultatet för några ARMA-modeller(ordning [24,5]-streckad, (AR)46-heldragen, [46,5]-punktstreckad, [36,5]-prickad), och originalet som förts in för hand.

Som synes finns inga tecken på att ett högre ordningstal skulle komma närmare den verkliga signalen. Snarare verkar det råda en viss slumpmässighet vad gäller resultatet och ingen av de "gissade" kurvorna är egentligen riktigt nära, varför det är troligt att metoden ej är lämplig att använda.

3.2.10 Sammanfattning

Med detta kan vi konstatera att metoderna som diskuterats i detta avsnitt tycks fungera bra på simulerade exempel, men ej för verkliga signaler. Vi kan inledningsvis urskilja två svårigheter, nämligen bruskänslighet och osäkerhet om modellordning. En tredje faktor som ännu inte berörts är kvantiseringseffekter. En ytterligare invändning är att verkliga system kan ha ett stort antal poler.

Bruskänsligheten och det okända ordningstalet har kunnat lösas åtminstone i simulerade exempel genom överidentifikation och modellreduktion. Det tycks nu vara lämpligt att undersöka huruvida kvantiseringen påverkar resultatet. I nästa avsnitt använder jag en annan metod för att skatta parametrarna i (1) där kvantiserings-effekterna tydliggörs.

3.3 Numerisk minimering av förlustfunktion

I det här avsnittet ansätter jag liksom i 3.2 en signalmodell och försöker skatta parametrar under det att jag minimerarar en förlustfunktion. Här gör jag minimeringen med en iterativ optimeringsmetod.

3.3.1 Förlustfunktionen

Funktionen vi vill minimera är som tidigare summan av kvadraterna på avvikelserna mellan modell och data:

$$V(\hat{\theta}) = \sum_{k=1}^{\text{antal sampel}} (\mathit{Data}(k) - \mathit{Modell}(\hat{\theta},k))^2 \qquad (1)$$

och vi vill minimera funktionen med avseende på parametrarna i parametervektorn θ. Modellen vi kommer att använda är som tidigare en summa av komplexa svängningar, som vi kan skriva som:

$$f(t_k) = \sum_{n=1}^{N} r_n e^{-s_n t_k} = \sum_{p=1}^{L} a_p e^{b_p t_k} \sin(\omega_p t_k + \phi_p) + \sum_{p=L+1}^{M} a_p e^{b_p t_k} \qquad (2)$$

med $\theta = [a_1,..,a_M,b_1,..,b_M,\phi_1,..,\phi_L,\omega_1,...,\omega_L]^T$

3.3.2 Nelder-Meades algoritm

Algoritmen finns implementerad som ett macro i MATLAB, **fmins**, [1], [10]. Man anger vilken funktion man vill minimera och sedan utförs minimeringen till en angiven toleransnivå eller avbryts vid ett givet antal itereringar. Macrot **fmins** tar två parametrar. Den första är namnet på den funktion, som man skrivit själv, som ska minimeras. Den har strukturen enligt (1). För det andra anger man initialvärden för parametervektorn θ.

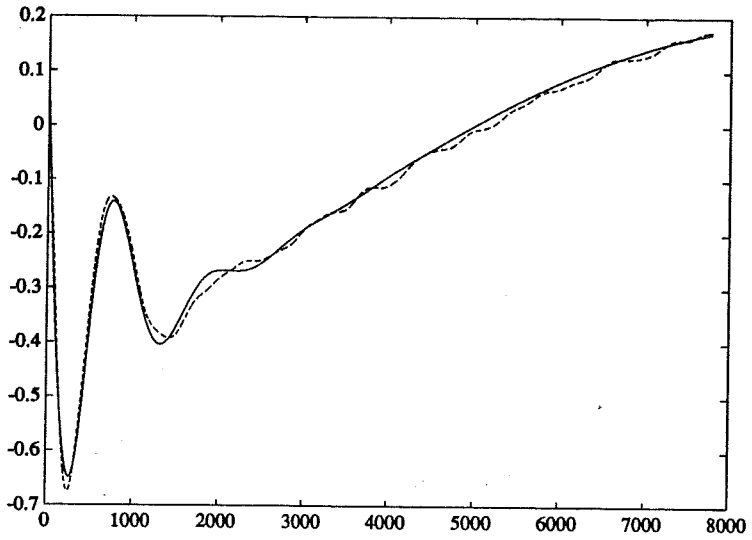
I beskrivningen anges att algoritmen presterar bäst då funktionen arbetar på fem variabler eller färre. Dessutom kan det finnas lokala minima som algoritmen så att säga fastnar i utan att hitta det eventuella globala minmat. Detta beror på vilka begynnelsevärden man ger algoritmen. Dessa begränsningar gör att metoden främst är lämplig när man redan vet en hel del om vad man letar efter.

3.3.3 Två försök

För att få en jämförelse med metoden i 3.2 arbetar jag med samma signal som i avsnitt 3.2.9, figur 18. Jag började med att skatta några få parametrar och efterhand som jag minimerat förlustfunktionen för dessa lade jag till flera komponenter i modellen , se (2) och letade efter nytt minima. Jag fortsatte tills jag tyckte att jag fått en god likhet med LF-delen av signalen. Observera dock att som Data användes den ofiltrerade varianten av signalen. Detta bl a för att undersöka om filtreringen i 3.2 trots de åtgärder som vidtagits haft en ödesdiger distorderande effekt som omöjliggjort identifiering.

På detta sätt gick jag vidare tills jag fått en god likhet, se figur 21 med LF-delen. Detta krävde 14 parametrar, funktionen i formelform blev:

$$f_1 = 3.9e^{-8.2\,10^5 t} + 0.57e^{-1.2\,10^5 t}\sin\left(2\pi 50\,10^3 t + \frac{3\pi}{2}\right) \qquad (3)$$
$$-0.72e^{-2.1\,10^6 t}\sin(2\pi 9.9\,10^5 t)$$

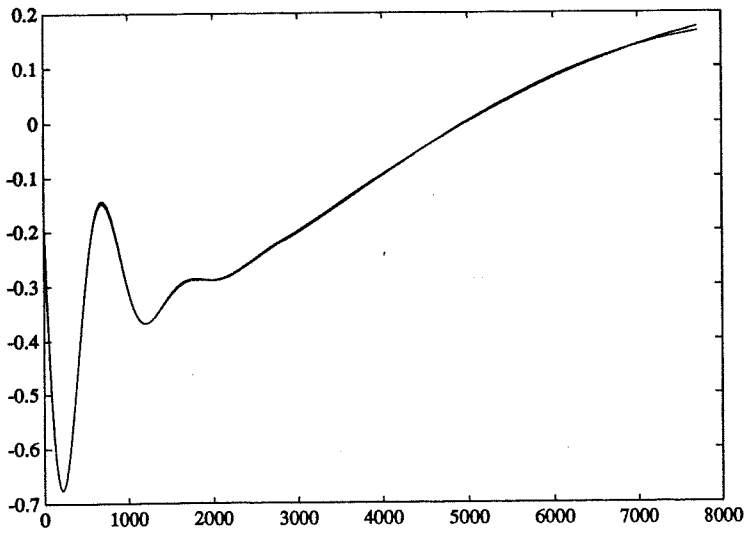


Figur 21 Funktionen f_1(heldragen) tillsammans med signalen från 3.2.9 efter filtrering(streckad).

Likheten är god och man kan förledas att lita på resultatet, men jag gjorde om proceduren med andra begynnelsevärden och fick då med samma modellstruktur en annan parametervektor:

$$f_2 = 8.6e^{-7.0\,10^5 t} + 1.7e^{-1.4\,10^5 t}\sin\left(2\pi 20\,10^3 t + \frac{43\pi}{25}\right) \qquad (4)$$
$$-0.74e^{-2.2\,10^6 t}\sin(2\pi 9.9\,10^5 t)$$

Signalerna f1 och f2 tillsammans syns i figur 22:

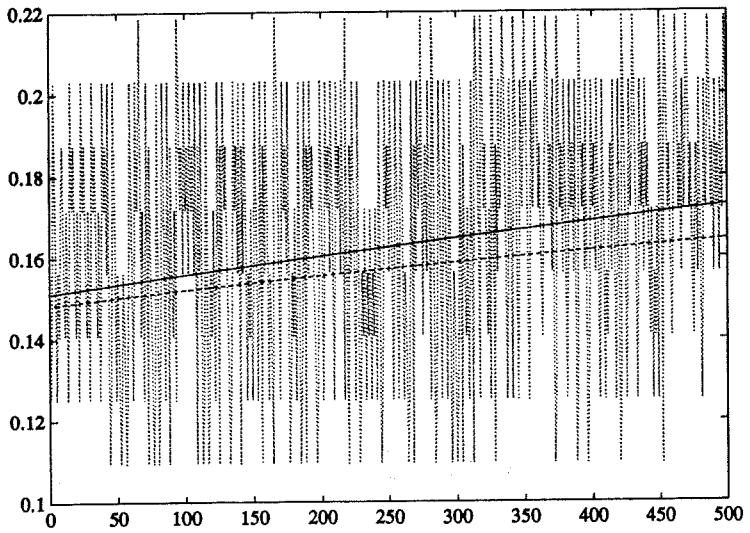


Figur 22 Funktionerna f_1 och f_2 tillsammans.

De båda funktionerna är mycket lika i det intervall där vi minimerat förlustfunktionen. Om man däremot låter dem fortsätta ser man att de skiljer sig åt avsevärt. Den ena kommer att fortsätta som en dämpad sinus med frekvensen 50kHz, medan den andra fortsätter som 20 kHz. Detta påminner om resultatet i avsnitt 4.2.

3.3.4 Kvantiseringseffekter

Om vi zoomar in de 500 sista samplen för data, respktive de båda funktionerna f1 och f2 ser vi att kvantiseringssteget i data är större än den största skillnaden mellan de båda "gissningarna", figur 23. Detta visar att kvaliteten på data är för låg för att kunna prediktera det fortsatta förloppet.



Figur 23 De femhundra sista samplen av den uppmätta signalen, f_1 och f_2.

Upplösningen i y-led tillsammans med den korta mättiden begränsar helt enkelt den tillgängliga informationen.

3.3.5 Sammanfattning

Efter att ha prövat en andra metod för att modellera signalen och skatta parametrar står det klart att kvaliteten på befintliga data är otillräcklig för att utan någon ytterligare information prediktera det fortsatta förloppet för de lågfrekventa svängningarna.

För att metoderna i 3.2 och 3.3 ska ge något resultat måste mättiden förlängas och/eller upplösningen i y-led förbättras.

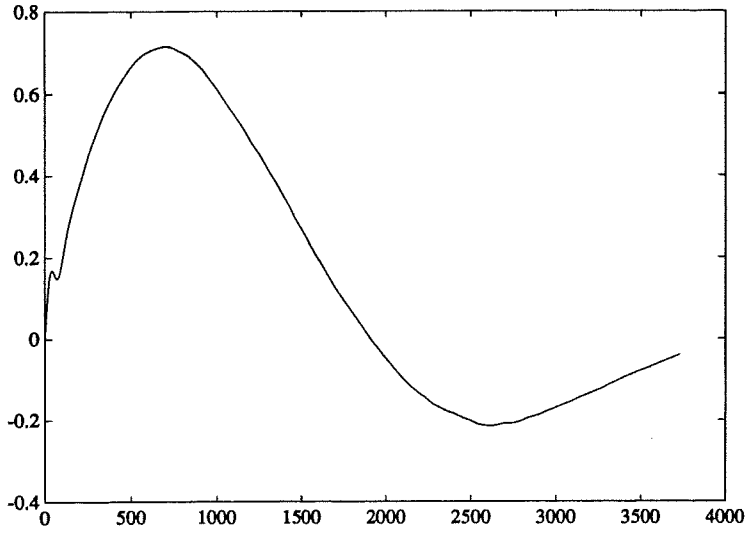
3.4 Identifikation av överföringsfunktion

Här prövar jag om det går bättre att prediktera signalens fortsatta förlopp då insignalen är känd. I stället för att modellera en signal försöker jag här få fram en modell för överföringsfunktionen från insignal till utsignal.

3.4.1 Insignalen

Vid den aktuella provningen har insignal och utsignal inte mätts samtidigt. Insignalen mättes istället stickprovsvis för att kontrollera att den behållit sitt utseende. I det här avsnittet har jag använt samma insignal, d v s ett av de här stickproven, till alla utsignalerna.

Insignalen har mätts med en deriverande prob, varför jag integrerat fram det slutliga utseendet i figur 24. Samplingsintervallet är 10 ns och antalet sampel ca 4000. Vi känner alltså insignalen för $0 \leq t \leq 40\mu s$.



Figur 24 Insignalen som använts.

3.4.2 Modell

Jag har använt modellen:

$$y(k) = \frac{B(z^{-1})}{F(z^{-1})}u(k) + e(k) \qquad (1)$$

som kallas Output Error(OE), därför att vitt brus adderas direkt till utsignalen.

Jag prövade även med en ARMAX-modell men fick sämre resultat.

3.4.3 Identifikation

Som tidigare nämnts finns i MATLABs System Identification Toolbox algoritmer implementerade för att skatta parametrar till olika modeller, också till OE-modellen. Jag hänvisar till MATLABs manual,[6].

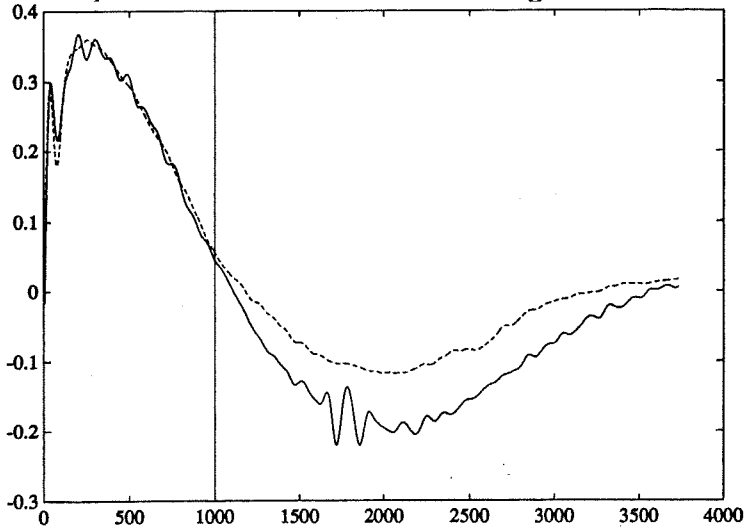
3.4.4 Test på uppmätta signaler

Jag prövar nu att skatta en överföringsfunktion utifrån ett antal uppmätta (ut)signaler och min insignal. Vissa experiment har repeterats med olika samplingshastighet. Nu tar jag några signaler som samplats med ett samplingsintervall på 10ns, ca 4000 sampel, vilket ger en mättid på 40μs. Precis som i avsnitt 4.2 filtrerar jag bort den högfrekventa delen. Jag har även insignalen uppmätt för samma intervall.

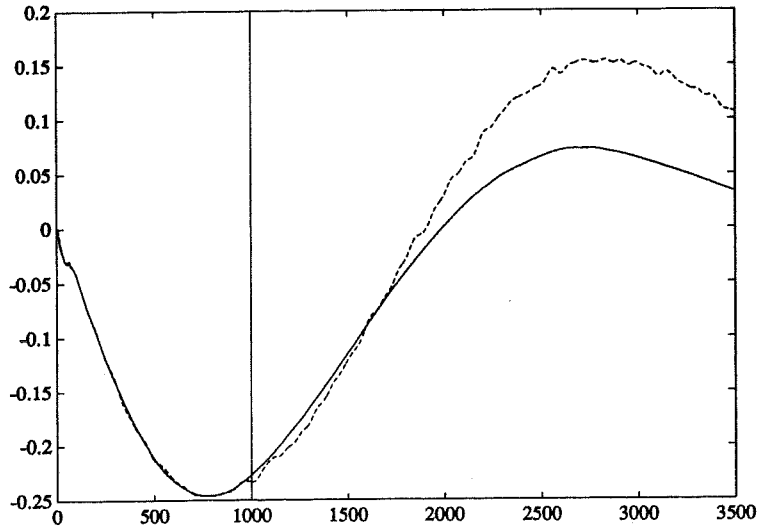
Jag använder sedan den första femtedelen av signalerna för att skatta överföringsfunktionen. Den första femtedelen svarar således emot de 8μs mättid vi haft tidigare, se avsnitt 3.1. Överföringsfunktionen matas sedan med hela insignalen och den då erhållna utsignalen jämförs med originalet.

Jag har skattat överföringsfunktioner med en pol och ett nollställe, eftersom jag inte kunde se någon förbättring genom ökning av ordningstalet.

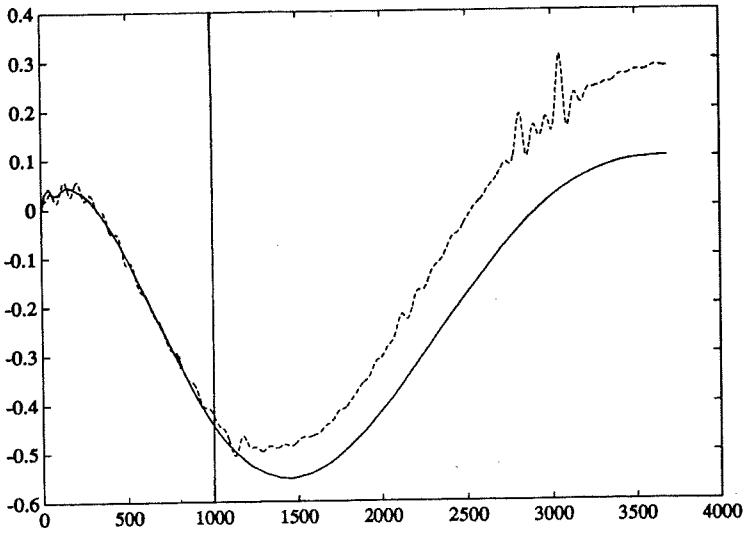
Figurerna nedan, (25-27), redovisar resultatet. Original och simulerad utsignal är ritade tillsammans. Data till vänster om den lodräta linjen har använts vid identifieringen.



Figur 25 Simulerad utsignal(streckad) och original(heldragen).

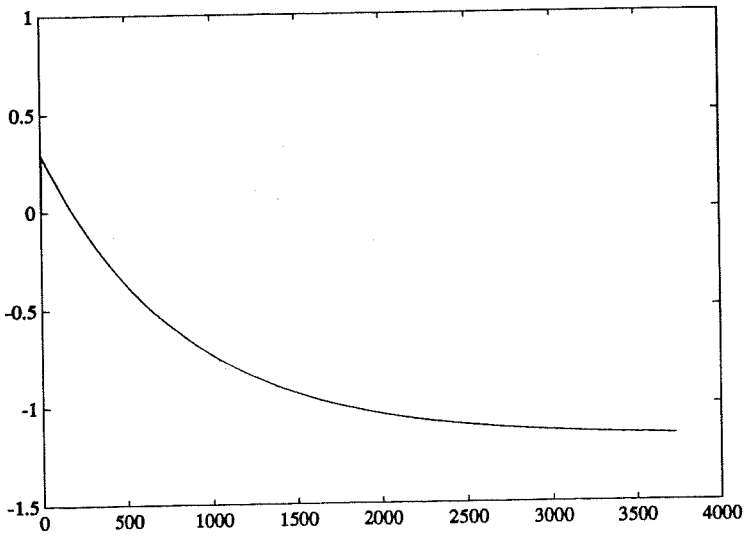


Figur 26 Simulerad utsignal(heldragen) och original(streckad).



Figur 27 Simulerad utsignal(heldragen) och original(streckad).

Med hjälp av den skattade överföringsfunktionen kan man t ex få en uppfattning om systemets stegsvar, figur 28:



Figur 28 Stegsvaret för överföringsfunktionen som skattats från signalen i figur 27.

3.4.5 Sammanfattning

Om vi har insignalen kan vi skatta en överföringsfunktion. Vi kan sedan med hjälp av denna prediktera utsignalen i det tidsintervall där insignalen är känd.

Troligen skulle resultatet bli bättre, om metoden tillämpades på äkta par av in- och utsignaler.

3.5 Förändra mätsituationen

I avsnitten 3.1-3.4 har olika metoder att utvinna information ur en kort mättid prövats och här följer en liten diskussion om hur man eventuellt skulle kunna ändra mätsituationen och därmed förutsättningarna.

3.5.1 Två parallella mätningar

För att fånga det långsamma förloppet räcker en långsam samplingshastighet. Ett sätt att fånga detta skulle vara att parallellt med den befintliga utrustningen också mäta med en betydligt lägre samplingshastighet under längre tid. Denna andra utrustning skulle inte behöva bli så dyr just på grund av att den inte behöver vara så snabb.

3.5.2 Ändra samplingshastighet under mätning

En metod som redan använts för att fånga både en snabb stigtid och ett långsammare urladdningsförlopp är att lägga in bryttidpunkter då samplingshastigheten ändras. Detta är möjligt på den befintliga utrustningen. Genom interpolering kan sedan datan vidarebehandlas med FFT. Naturligtvis kräver detta förfarande kunskap om den förväntade signalen, och den begränsade minnesarean kommer troligen även då att innebära vissa kompromisser.

3.5.3 Mät in- och utsignaler

En annat sätt kan vara att mäta par av in- och utsignaler och sedan använda sig av metodiken i avsnitt 3.4 för att skatta en överföringsfunktion. Man skulle då troligen få ett bättre resultat än i 3.4 eftersom insignal och utsignal där ej mätts vid samma tillfälle.

En fördel med denna metod är att man kan använda den skattade överföringsfunktionen även för att bedöma hur resultatet hade blivit för andra insignaler.

4. SAMMANFATTNING OCH SLUTSATSER

Separationsfiltreringen i avsnitt 3.1 löser problemet om endast de högfrekventa resonanserna är intressanta för en vidare behandling.

Om det långsammare förloppets fortsättning är intressant blir resultatet mer nedslående. Avsnitt 3.2 och 3.3 visar att kvantiseringen av mätdata tillsammans med den korta mättiden i princip omöjliggör en tillförlitlig extrapolation av fortsättningen. För att lyckas med detta krävs mer information.

I 3.4 används insignalen vid experimentet för att skatta överföringsfunktioner. Resultatet blir med tvekan godkänt, troligen beror detta på att insignal och utsignal ej kommer från samma experiment.

För att få mer information vid experiment av det slag som behandlats här bör nog övervägas om det inte är möjligt att förändra mätsituationen enligt något av förslagen i 3.5 eller på något annat sätt.

5. REFERENSER

[1] The Math Works Inc, *MATLAB™ - User's Guide*, 1989.

[2] F. J. Harris, *On the use of Windows for Harmonic Analysis with the Discrete Fourier Transform*, Proc. IEEE, vol. 66, no. 1, pp. 51-83.

[3] The Math Works Inc., *Signal Processing Toolbox for use with MATLAB™ - User's Guide*, 1988.

[4] A.J. Mackay, A. McGowen, *Improved Pencil-of-Functions Method and Comparisons with Traditional Methods of Pole Extraction* , IEEE Trans. on Ant. and Prop. , vol. AP-35, no. 4, April 1987.

[5] T. Söderström, P. Stoica, *System Identification*, Prentice-Hall, 1989.

[6] The Math Works Inc., *System Identification Toolbox for use with MATLAB™ - User's Guide*, 1988.

[7] A. Laub, *On computing balancing transformations*, Proc. 1980, Joint Autom. Control Conf., San Fransisco, Aug. 1980.

[8] The Math Works Inc., *Control System Toolbox - for use with MATLAB™ - User's Guide*, 1988.

[9] J.V. Candy, G.A. Clark, D. A. Goodman, *Transient Electromagnetic Signal Processing: An Overview of Techniques*, edited by Edmund K. Miller, van Nostrand Reinhold, 1986, ch. 12, pp. 422-423.

[10] J.E. Dennis Jr, P.J. Woods, *New Computing Environments: Microcomputers in Large-Scale Computing*, edited by A. Wouk, SIAM, 1987, pp. 116-122.

APPENDIX A

MATLAB

MATLAB är ett programpaket som lämpar sig väl för bl a matrisoperationer, signalbehandling och numerisk analys. Programmet finns i versioner för såväl persondatorer som arbetsstationer och stordatorer.

Programpaketet är uppbyggt utifrån ett interaktivt arbetssätt och funktionsorienterat. En mångfald funktioner finns redan i MATLABs grundutförande, men till olika områden finns möjlighet att skaffa sig en "toolbox"; t ex Signal Processing Toolbox eller System Identification Toolbox.

Dessa innehåller då flera faciliteter och behöver man ytterligare funktioner är det enkelt att skriva sina egna. Detta går också så mycket enklare därför att MATLAB byggts kring en generell matris som datatyp, vilket passar bra för problem från en mängd olika områden.

Resultatet från beräkningar kan sedan enkelt åskådliggöras med de grafikmöjligheter som finns i MATLAB.

APPENDIX B

Journalfiler

Det följer här tre journalfiler som får visa i grova drag hur jag jobbat med MATLAB. Några av de funktioner som anropas har jag skrivit själv, de återfinns listade i APPENDIX C. Journalfilerna har en koppling till texten i rapporten, kommandot ">>meta figurx.met" anger således att figur x i rapporten skapats på detta sätt. Journalfilerna är dock på intet sätt fullständiga.

1. **Separationsfiltrering**, jfr avsnitt 3.1.

2. **Identifikation av poler och modellreduktion**,jfr 3.2.

3. **Identifikation av överföringsfunktion**, jfr 3.4.

1. Separationsfiltrering

```
   Journalfil för figur 6 till 9 i avsnitt 3.1.
>>who

  Your variables are:

            signal1

>> plot(signal1);
>> meta figur6.met
>> signal=signal1(480:8192);
>>Wp=0.005; Ws=0.01; Rp=0.05; Rs=20;
>> Ån,WnÅ=cheblord(Wp,Ws,Rp,Rs);
>> Åb,aÅ=cheby1(n,Rp,Wn);
>>LF=myfiltfilt(b,a,signal);
>>plot(LF);
>>meta figur7.met
>>HF=signal-LF;
>>plot(HF);
>>meta figur8.met
>>plot(HF(1:1500));
>>meta figur9.met

>> Åf1,Y1Å=myfft(HF(1:1500),1e-9,1);

  parseval =

     1.0e-07 *

      0.3650     0.3650


>> meta figur10.met
>>diary off
```

2. Identifikation av poler och modellreduktion

Journalfil som visar principiellt tillvägagångssätt för identifikation med efterföljande modellreduktion

```
>>help ex41

    ett script som genererar signalen i exempel 4.1

    factor som anger brusets st. avv. måste vara definierad vid anrop.

    signalen återfinns sedan i variabeln x

>>factor=1e-3;
>>ex41
>>meta figur11.met
>>help id

    function Ä A,P,C,simdEt,contisimdEtÅ =id(dEt,ord,h,method,plott)

    Funktionen identifierar fram ett AR-polynom ur datan dEt,
    och därur beräknas sedan diskreta poler och begynnelsevärden.

    Polynomkoefficienterna återfinns i A, Polerna i P och
    de skattade begynnelsevärdena i C. Slutligen kan den
    återskapade sekvensen simdEt jämföras med dEt, och även en
    längre sekvens, contisimdEt, kan inspekteras.

    Önskad ordning anges med ord, h är sampelintervallet och med method
    anges önskad id-algoritm. Möjliga val är(jfr MATLAB):
                                              'arx'
                                              'armax'
                                              'ivar'
                                              'fb'
                                              'ls'
    Plott slutligen anger om man vill ha plottar, om plott=1 får man plottar.

>>ÄA,P,C,s,scÅ=id(x',Ä16Å,0.5,'arx',1);
>>impuls=sc(1:128);
>>plot(impuls);
>>meta figur15.met
>>help impulse2ss

   function     ÄAss,Bss,CssÅ=impulse2ss(impulssvar);

    Skapar en tillståndsrepresentation för det linjära system som har
    impulssvaret impulssvar.

    Tillståndsrepresentationen blir av en ordning som är lika med
    längden av impulssvaret.

>> ÄAss,Bss,CssÅ=impulse2ss(impuls);
>> help dbalreal

   DBALREAL  Discrete balanced state-space realization and model reduction.
   ÄAb,Bb,CbÅ = DBALREAL(A,B,C) returns a balanced state-space realization
   of the system (A,B,C).
   ÄAb,Bb,Cb,m,TÅ = DBALREAL(A,B,C) also returns a vector M containing the
   diagonal of the gramian of the balanced realization, and matrix T, the
   similarity transformation used to convert (A,B,C) to (Ab,Bb,Cb).  If
```

```
   the system (A,B,C) is normalized properly, small elements in gramian M
   indicate states that can be removed to reduce the model to lower order.
   See also DMODRED, BALREAL and MODRED.



>> ÄAb,Bb,Cb,mÅ=dbalreal(Ass,Bss,Css);



>> semilogy(m);
>>meta figur16.met
>>help dmodred

   DMODRED Discrete-time model state reduction.
   ÄAb,Bb,Cb,DbÅ = DMODRED(A,B,C,D,ELIM) reduces the order of a model
   by eliminating the states specified in vector ELIM.  The state
   vector is partioned into X1, to be kept, and X2, to be eliminated,

   A = öA11  A12öB = öB1öC = öC1 C2ö
       öA21  A22ö    öB2ö

   xÄn+1Å = AxÄnÅ + BuÄnÅ,    yÄnÅ = CxÄnÅ + DuÄnÅ

   X2Än+1Å is set to X2ÄnÅ, and the resulting equations solved for X1.
   The resulting system has LENGTH(ELIM) fewer states and can be
   envisioned as having set the ELIM states to be infinitely fast.

   See also DBALREAL, BALREAL and MODRED

>> ÄAbb,Bbb,Cbb,DbbÅ=dmodred(Ab,Bb,Cb,0,Ä7:128Å);



>> newP=eig(Abb);



  'Nu finns polerna i newP';
>>help pol2beg

    script pol2beg

    från givna diskreta poler P och måldata dEt räknas begynnelse värden C
    fram och en jämförelse mellan resultat och måldata ritas

    samplingsintervallet måste vara samma för P och dEt

>> dEt=x';
>> P=newP;



>>pol2beg
```

```
>>help z2s

   function ÄcontpolesÅ=z2s(discpoles,sampint);

    Beräknar kontinuerliga poler från diskreta och samplingsintervall.

>>cP=z2s(P,0.5);


   De skattade (kontinuerliga) polerna finns i cP och
  residyerna i C .
>>diary off
```

3. Identifikation av överföringsfunktion

Journalfil som visar principiell arbetsgång för att
skatta en överföringsfunktion

```
>> plot(insig);
>> meta figur24.met
>> theta=oe(Ä utsig(1:1000) insig(1:1000)Å,Ä 2 1 4Å,-1,-1,-1-1,1e-8);

>> present(theta)

 This matrix was created by the command OE     on 5/2 1991 at 16:15
 Loss fcn: 7.8105e-05   Akaikeés FPE: 7.8693e-05 Sampling interval 1.0000e-08
 The polynomial coefficients and their standard deviations are

 B =

         0         0         0         0    0.2879   -0.2893
         0         0         0         0    0.0041    0.0041

 F =

    1.0000   -0.9988
         0    0.0001

>> simut=idsim(insig,theta);

>> plot(Äsimut utsigÅ);

>> meta figur25.met
>> diary off
```

APPENDIX C

Egna MATLAB-macros

Här följer en listning av några egna MATLAB-macros, de flesta både korta och enkla:

deci
expowsp
id
impulse2ss
myfft
myfiltfilt (MATLABs filtfilt modifierad)
pol2beg
window
z2s,s2z

```
function Ä dEt,dsampint Å=deci(Et,h,dec,plott)
% function ÄdEt,dsampintÅ=deci(Et,h,dec,plott)
%
% Decimerar en datamängd Et, tar vart dec:te värde. Ange
% samplingsintervall med h, nya samplingsintervallet fås fås i
% dsampint.
%
% plott=1 ger plottning
%
% Observera att decimering bör kombineras med filtrering

k=1:length(Et);
dEt=Et(rem(k-1,dec)==0);
dsampint=h*dec;

% ev. plot
if plott==1
subplot(211),plot(Et)
subplot(212),plot(dEt)
pause;
subplot;
end;
return;
```

```
function Äf1,Y1Å= expowsp(P,C,f,dh,plott);

% function Äf1,Y1Å= expowsp(P,C,f,dh,plott);
%
% Räknar ut exakt frekvenskaraktäristik för frekvenserna i  vektorn f (Hz)
% om P är diskreta poler
%    C är residyer
%    dh är samplingsintervall.
%
% om plott=1 så plottas frekvensgången

Y=ÄÅ;
fw=ÄÅ;
for index=1:length(f),
  for pol=1:length(P),
    fw(pol)=1/(2*pi*f(index)*i-log(P(pol))*(1/dh));
  end;
 Y(index)=C'*fw';
end;
Pyy=Y.*conj(Y);

if plott==1,
  loglog(f(2:length(f)),abs(Pyy(2:length(f))));
end;

pause;
return;
```

```
function Ä A,P,C,simdEt,contisimdEt Å = id(dEt,ord,h,method,plott)

% function Ä A,P,C,simdEt,contisimdEtÅ =id(dEt,ord,h,method,plott)
%
% Funktionen identifierar fram ett AR-polynom ur datan dEt,
% och därur beräknas sedan diskreta poler och begynnelsevärden.
%
% Polynomkoefficienterna återfinns i A, Polerna i P och
% de skattade begynnelsevärdena i C. Slutligen kan den
% återskapade sekvensen simdEt jämföras med dEt, och även en
% längre sekvens, contisimdEt, kan inspekteras.
%
% Önskad ordning anges med ord, h är sampelintervallet och med method
% anges önskad id-algoritm. Möjliga val är(jfr MATLAB):
%                                          'arx'
%                                          'armax'
%                                          'ivar'
%                                          'fb'
%                                          'ls'
% Plott slutligen anger om man vill ha plottar, om plott=1 får man plottar.


nrofdata=length(dEt);
sampint=h;

% metod
if length(method)<5 ,
  d=length(method);
  for i=(d+1):5,
    method(i)=' ';
  end;
end;

% identifiera
if method=='arx  ' ,
  Etheta=arx(dEt,ord(1),-1,sampint);
elseif method=='armax' ,
  Etheta=armax(dEt,ord,-1,-1,-1,-1,sampint);
elseif method=='ivar ' ,
  Etheta =ivar(dEt,ord(1),ord(2),-1,sampint);
else
  method=method(1:2);
  Etheta=ar(dEt,ord(1),method,'now',-1,sampint);
end;

% extrahera A-polynomet
Ä A,BÅ=polyform(Etheta);

%räkna ut poler
P=roots(A);

% kontrollera stabilitet, om instabil avbryt.
if any(abs(P)>1),
  'instabil'
  C=zeros(P);
  simdEt=zeros(dEt)';
  return;
end;

% plotta poldiagram
if plott==1 ,
  zpplot(zp(Etheta));
  pause;
end;
```

```
Pt=P';
% lös begynnelsevärden
grad=ord(1);
for rad =1:nrofdata,
  for kol =1:grad,
    phi(rad,kol)=Pt(kol).Ü(rad-1);
  end
end
C=phiÖdEt;


% simulering
k=1:nrofdata;
  for k= 1:nrofdata,
    simdEt(k)=C'*((P).Ü(k-1));
  end



%simulering av fortsättning
  Pk=ones(P);
  contisimdEt=zeros(nrofdata*8,1);
  for k= 1:nrofdata*8,
    contisimdEt(k)=C'*Pk;
    Pk=Pk.*P;
  end



%jämförande plottar
if plott==1 ,
  %tidsaxel
  dt=linspace(0,(nrofdata-1)*sampint,nrofdata);
  plot(dt,Ä dEt simdEt' Å);
  xlabel('Original: heldragen, simulerad: streckad');
  pause;
  longt=linspace(0,(8*nrofdata-1)*sampint,8*nrofdata);
  plot(longt,Ä   real(contisimdEt') Å);
  xlabel('En längre sekvens av den simulerade signalen');
  pause;
end;
return;
```

```
function      ÄAss,Bss,CssÅ=impulse2ss(impulssvar);

%function      ÄAss,Bss,CssÅ=impulse2ss(impulssvar);
%
% Skapar en tillståndsrepresentation för det linjära system som har
% impulssvaret impulssvar.
%
% Tillståndsrepresentationen blir av en ordning som är lika med
% längden av impulssvaret.

dim=size(impulssvar);
if dim(1) > dim(2),
  Css=impulssvar';
else
  Css=impulssvar;
end;

N=length(Css);

r=zeros(1,N);
c=zeros(1,N);
c(2)=1;
Ass=toeplitz(c,r);

Bss=zeros(N,1);
Bss(1)=1;
return;
```

```
function Ä f1,Y1Å=myfft(signal1,h,plott)

% function Äf1,Y1Å=myfft(signal1,h,plott)
%
% Funktionen beräknar FFT av signal1, sampelintervallet ska anges
% i h.
% Utparametrar är f1:frekvensvektor och Y1:resultatvektor från FFT
%
% med plott=1 fås en plot

fs05=1/(h*2);

N1=length(signal1);
Y1=fft(signal1,N1)*h;

Pyy1=Y1.*conj(Y1);

N105=N1/2;
f1=fs05*(0:N105)/N105;



% FFT, positiva frekvenser
if plott ==1,
loglog(f1(2:length(f1)),abs(Y1(2:(N105+1))));
title('FFT ');
xlabel('frekvens(Hz)');
pause;
end;

% Effekt neg+pos frekv.
Pyy1((N105+2):N1)=Ä Å;
Pyy1(2:N105)=2*Pyy1(2:N105);


%test, jämför energier för signal och signalens fouriertransform
%  ( Parseval)
parseval=Ä sum(signal1.Ü2)*h sum(Pyy1)/(N1*h)Å
```

```
function y = myfiltfilt(b,a,x)
% myFILTFILT Zero-phase forward and reverse digital filtering.
%Y = myFILTFILT(B, A, X) filters the data in vector X with the
%filter described by vectors A and B to create the filtered
%data Y.  The filter is described by the difference equation:
%
%  y(n) = b(1)*x(n) + b(2)*x(n-1) + ... + b(nb+1)*x(n-nb)
%                   - a(2)*y(n-1) - ... - a(na+1)*y(n-na)
%
%After filtering in the forward direction, the filtered
%sequence is then reversed and run back through the filter.
%The resulting sequence has precisely zero-phase distortion
%and double the filter order.  Care is taken to minimize
%startup and ending transients by matching initial conditions.
%See also FILTER.

%L. Shure 5-17-88
%(c) Copyright 1988, by The MathWorks, Inc.

Äm,nÅ = size(x);
b = b(:).';
a = a(:).';
nb = length(b);
na = length(a);
nfilt = max(nb,na);
% set up initial conditions to remove dc offset problems at the beginning of
% the sequence
if nb == na
zi = cumsum(b(nfilt:-1:1)-a(nfilt:-1:1)); zi = zi(nfilt-1:-1:1);

else
if na < nfilt
bb = b;
aa = a;
aa(nfilt) = 0;
else% nb < nfilt
aa = a;
bb = b;
bb(nfilt) = 0;
end
zi = cumsum(bb(nfilt:-1:1)-aa(nfilt:-1:1)); zi = zi(nfilt-1:-1:1);
clear bb aa
end
% reflect beginning sequence of data so slopes match and filter this
% precursory signal so end effects are reduced

% jag har ändrat nfact
nfact =400*(nfilt-1);
%

if m == 1% row data
y = Ä2*x(1)-x((nfact+1):-1:2) xÅ;
%        plot(y);
 %       pause;
   %     meta ywithpend.met;
else % column data
y = Ä2*x(1)-x((nfact+1):-1:2);xÅ;
%        plot(y);
 %       pause;
  %      meta ywithpend.met;
end
y = filter(b,a,y,Äzi*y(1)Å);
%plot(y);
%pause;
```

```
%meta yafter1filt.met;
% reverse data, prepend and filter again
y(:) = y(length(y):-1:1);
if m == 1% row data
y = Ä2*y(1)-y((nfact+1):-1:2) yÅ;
 %       plot(y);
  %      pause;
   %     meta yrevwithpend.met;
else % column temp
y = Ä2*y(1)-y((nfact+1):-1:2);yÅ;
    %    plot(y);
     %   pause;
      %  meta yrevwithpend.met;
end
y = filter(b,a,y,Äzi*y(1)Å);
%plot(y);
%pause;
%meta yafter2filt.met;

% remove pieces of y that were tacked on to eliminate end effects
y(Ä1:nfact max(m,n)+nfact+(1:nfact)Å) = ÄÅ;
% reverse the final sequence to be in original direction
y = y(length(y):-1:1);
%plot(y);
%pause;
%meta finaly.met;
```

```
% script pol2beg
%
% från givna diskreta poler P och måldata dEt räknas begynnelse värden C
% fram och en jämförelse mellan resultat och måldata ritas
%
% samplingsintervallet måste vara samma för P och dEt

N=length(dEt);
clear C;

% kontrollera stabilitet
if any(abs(P)>1),
  'instabil'
   return;
end;


Pt=P';
%begynnelsevärden
grad=length(P);
for rad =1:N,
  for kol =1:grad,
    phi(rad,kol)=Pt(kol).Ü(rad-1);
  end
end
C=phiÖdEt;



%simulering av fortsättning
  Pk=ones(P);
  contisimdEt=zeros(N,1);
  for k= 1:N,
       contisimdEt(k)=C'*Pk;
    Pk=Pk.*P;
  end



%jämförande plottar
if 1==1 ,
  %tidsaxel
  %dt=linspace(0,(nrofdata-1)*sampint,nrofdata);
  plot(Ä dEt real(contisimdEt) Å);
  pause;
end;
return;
```

```
function Ä windEtÅ=window(Et,plott);

% function Ä windEtÅ=window(Et,plott);
%
% lägger på ett Hanning-fönster på de
% sista 10 procenten av kurvan Et
%
% Om plott=1 fås plot
%

windEt=Et;
N=length(Et);
nc=fix(0.9*N);
nd=N-nc;
wind=hanning(2*nd+1);
windEt(nc+1:N)=windEt(nc+1:N).*wind(nd+2:2*nd+1);

if plott==1,
  plot(windEt);
  title('Den fönsteravslutade signalen.');
end;
return;
```

```
function ÄcontpolesÅ=z2s(discpoles,sampint);

%function ÄcontpolesÅ=z2s(discpoles,sampint);
%
% Beräknar kontinuerliga poler från diskreta och samplingsintervall.

contpoles=log(discpoles)/sampint;
```

```
function ÄdiscpolesÅ=s2z(contpoles,sampint);

%function ÄdiscpolesÅ=s2z(contpoles,sampint);
%
% Beräknar diskreta poler från kontinuerliga och samplingsintervall.

discpoles=exp(sampint*contpoles);
```