CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

# Digital flickermeter

Håkan Svensson

Institutionen för Reglerteknik
Lunds Tekniska Högskola
April 1988

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

# Digital flickermeter

Håkan Svensson

Institutionen för Reglerteknik
Lunds Tekniska Högskola
April 1988

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

*Document name*
Master Thesis

*Date of issue*
April 1988

*Document Number*
CODEN: LUTFD2/(TFRT-5378)/1-51/(1988)

*Author(s)*
Håkan Svensson

*Supervisor*
Rolf Johansson, Lars Messing

*Sponsoring organisation*
Sydkraft AB

*Title and subtitle*
Digital Flickermeter

*Abstract*

The following report deals with the construction of a digital flickermeter. This is an instrument which measures fluctuations on the mains voltage. In the conditions there was given a measure signal sampled with 100 Hz. Out of this, according to the specifications of UIE (The International Union for Electroheat), flicker severity should be measured. To achieve this purpose I have constructed a program, which simulates the eye's sensitivity for illumination changes. This is done by filtering of the mains voltage through a bandpass filter, and afterwards statistical computations according to UIE:s methods.

*Key words*

*Classification system and/or index terms (if any)*

*Supplementary bibliographical information*

*ISSN and key title*

*ISBN*

*Language*
Swedish

*Number of pages*
51

*Recipient's notes*

*Security classification*

# EXAMENSRAPPORT - DIGITAL FLICKERMETER

AV HÅKAN SVENSSON 1988-03-01

Innehåll

# 1 Sammanfattning

Följande rapport behandlar konstruktion av en digital flickermeter. Förutsättningarna var att en mätsignal samplad med 100 Hz fanns given. Ur denna skulle flicker mätas enligt UIE:s (The International Union for Electroheat) specifikationer. För att uppnå dessa mål har jag konstruerat ett program, som simulerar ögats och hjärnans känslighet för belysningsvariationer genom att bandpassfiltrera inom lämpliga frekvenser. Slutligen utföres statistiska beräkningar enligt UIE:s metoder.

Programmet är kodat i språket Fortran och finns implementerat på VAX-datorn. Dessutom finns i avdelningens IBM PC en Modula-2-version som kan utföra mätningarna i fält.

# 2 Inledning

Denna examensrapport är uppdelad på en sammanfattning som ni förhoppningsvis redan stiftat bekantskap med, ett kapitel, där en grundlig analys av den digitala flickermetern genomförs, ett resultatkapitel samt ett kort kapitel som redogör högst sammanfattat hur man allmänt konstruerar digitala filter. Innan vi kastar oss över nämnda kapitel presenteras här en inledning, som kortfattat beskriver flicker, berättar om bakgrunden till mitt arbete samt dess syfte.

## 2.1 Flicker

Vad är flicker? Denna fråga ställde jag mig för drygt ett halvår sedan, då jag studerade Sydkrafts exjobbskatalog. Jag var dock inte först, utan under ett par års tid har UIE samlat sina krafter för att på ett vetenskapligt och objektivt sätt kunna besvara frågan.

Rent fysikaliskt kan vi beskriva flicker som spänningsfluktuationer orsakade av varierande last på nätet. Den enskilde abonnenten upplever det som irriterande blinkningar i sina lampor. Ett vanligt exempel är belysningen i järnvägsvagnar som matas med en spänning vars frekvens (16 2/3 Hz) endast är en tredjedel av det normala. För övrigt är det främst industrin som bidrar med flickerstörningar, varvid ljusbågsugnar är ett av de mest typiska exemplen. Även laster såsom värmepumpar och svetsmaskiner kan orsaka besvärande flicker.

## 2.2 Bakgrund

Problemet med tidigare flickermetrar har varit att de alla byggt på olika specifikationer och följaktligen har jämförelser länder emellan varit av föga intresse. Denna brist på överensstämmelse i mätresultat beror på att själva flickret egentligen är en subjektiv variabel. Olika personer reagerar för ljusvariationer vid olika frekvenser. UIE har därför sedan 1979 arbetat fram en specifikation som så objektivt som möjligt bedömer flickersignalen. Genom att bland annat samla stora grupper av människor har man genom upprepade försök kunnat definiera ögats känslighet vid olika frekvenser för nämnda fluktuationer. Resultatet av dessa ansträngningar är den kurva som redovisas i bilaga 1. Den tämligen olinjära kurvformen indikerar att ögat är som mest känsligt vid frekvenser mellan 8-10 Hz.

## 2.3 Syfte

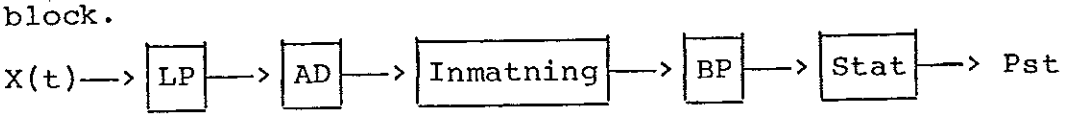
Problemet är därför att arbeta fram en funktionsbeskrivning för den framtida flickermetern. Målet är att så noggrant som möjligt simulera ögats känslighet med hjälp av modern teknik. Man utvecklade även statistiska metoder för att på ett objektivt sätt kunna analysera den erhållna flickersignalen. Detta ledde så småningom fram till en funktionsbeskrivning av ett analogt instrument, men man nämner samtidigt att en digital motsvarighet skulle kunna vara en alternativ lösning. Det är på den här punkten mitt examensarbete kommer in i bilden. Syftet är helt enkelt att ta fram programvara som enligt gällande specifikationer utför flickermätning samt önskade statistiska bearbetningar.

# 3 Digital flickermeter

I detta kapitel behandlas själva kärnan till problemet. Först en inledande sammanfattning, varefter en grundlig analys av de två centrala momenten digital filtrering och statistisk behandling presenteras.

## 3.1 Sammanfattning

Den färdiga flickermetern kan delas upp i fem separata block.

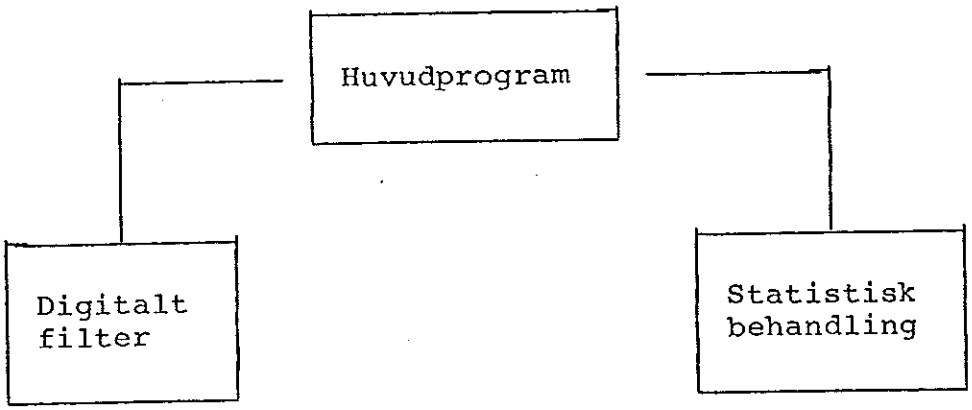



Utifrån nätspänningen hämtas den sexpulslikriktade mätsignalen X(t), som passerar ett analogt LP-filter

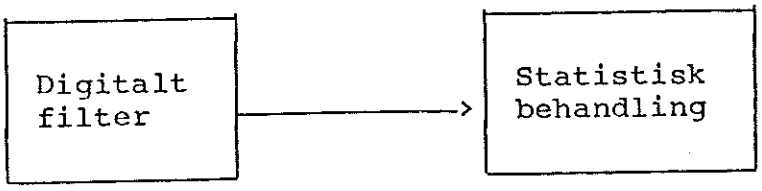
med gränsfrekvensen 50 Hz. Denna filtrering utföres delvis för att bli av med högfrekventa störningar, men framförallt för att signaler med frekvenser över 50 Hz inte ska kunna påverka resultatet genom viknings-distorsion. I block två AD-omvandlas signalen, varefter den är färdig för datorbehandling. Inmatningen är tänkt att ske antingen i fält eller genom att flickermetern läser data i efterhand från en diskett. De avlästa och samplade värdena samlas i en vektor, varefter signalen bandpassfiltreras i ett digitalt IIR-filter med mitt-frekvensen vid 8,8 Hz. Ut ur filtret erhåller vi den samplade flickersignalen, som behandlas i ett statistiskt program med metoder angivna av UIE. Som resultat får man ett mått, Pst (st = short time), som anger graden av flicker. Då detta värde är mindre än ett kan flickret anses ligga på en acceptabel nivå.

Examensarbetet har omfattat framtagning och kodning av en algoritm för flickermätning, programvara för statistisk bearbetning av flickersignalen och att undersöka om det är rimligt att utföra signalbehand-lingen kontinuerligt i en mätdator. Detta har resul-terat i två versioner, en skriven i Fortran och den andra kodad i Modula-2, som lämpar sig bättre för real-tidsprogrammering.

Modula-2-variantens programstruktur består enligt dess speciella filosofi av ett huvudprogram med tillhörande definitions-/implementationsmodul-par.




Fortranflickermetern består av två separata huvud-program som kommunicerar med varandra genom en datafil där signalens värde lagras.



## 3.2 Digitalt filter

I detta kapitel behandlas framtagningen av den matematiska algoritmen samt ett blockschema över mjukvarurealiseringen.

### 3.2.1 Matematisk teori

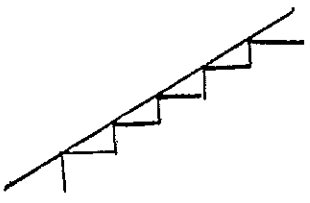
I förutsättningarna till denna uppgift fanns redan överföringsfunktionen för signalbehandlingen given i Laplaceform.

$$F(s) = \frac{kw_1 s}{(s^2 + 2Ls + w_1^2)} \quad \frac{1 + s/w_2}{(1 + s/w_3)(1 + s/w_4)}$$

$w_1$ = 2pi 9,15494
$w_2$ = 2pi 2,27979
$w_3$ = 2pi 1,22535
$w_4$ = 2pi 21,9
L = 2pi 4,05981
k = 1,74802

Uppgiften blev därför att konvertera detta kontinuerliga uttryck till en tidsdiskret pulsöverföringsfunktion, det vill säga ett digitalt filter.

Som det antyds i Appendix A.2 utgår man ofta från en analog systemfunktion vid konstruktion av IIR-filter och det är precis vad vi har här. Enligt gällande teorier, se Appendix A.2, är vårt mål att avbilda s-planet på z-planet. Här kan man välja mellan olika metoder beroende på önskad noggrannhet. Den enklaste, som jag använder här, beskrivs i figur 1.



Figur 1

Denna kallas "sample and hold-metoden" därför att den direkt utnyttjar sig av AD-omvandlarens utsignal. Detta kommer att ge upphov till ett fel i filtret, eftersom den trappstegsformiga insignalen förlorar information mellan varje sampel. Felet antas dock vara så litet att det inte ska påverka resultatet i slutändan.

För att återgå till beräkningarna underlättas dessa om en partialbråksuppdelning genomförs.

$$F(s) = \frac{As + B}{s^2 + 2Ls + w_1^2} + \frac{C}{1 + s/w_3} + \frac{D}{1 + s/w_4}$$

Efter en stunds räknande får man följande resultat.

$$A = -(w_3C + w_4D) = 64,77228$$

$$B = -w_1(C + D) = 1955,73343$$

$$C = \frac{kw_1 - (2L - w_4 - w_1/v_4)D}{2L - w_3 - w_1/w_2} = -0,12748$$

$$D = \frac{kw_1w_4/w_2 - kw_1}{2Lw_4/w_3 - w_4/w_3 - w_1/w_3 - 2L + w_4 + w_1/w_4}$$

$$= -0,46359$$

Nästa steg i våra ambitioner att nå fram till det färdiga filtret är transformationen till z-planet. Detta utförs enklast med hjälp av färdiga tabeller, se referens 4, men innan denna tämligen enkla operation måste vi redigera våra tre bråk, så att tabellen går att använda. Det är då främst det första bråket, som måste anta en vänligare form.

$$\frac{As + B}{s^2 + 2Ls + w_1^2} = A \frac{s}{s^2 + 2Ls + w_1^2}$$

$$+ \frac{B}{w_1^2} \frac{w_1^2}{s^2 + 2Ls + w_1^2}$$

Nu är det enkelt att med tabellens hjälp (referens 4) beräkna H(z).

$$H(z) = \frac{b_1 + b_2 z}{1-2z^{-1}e^{-LT}\cos(aT) + z^{-2}e^{-2LT}}$$

$$+ \frac{b_3 + b_4 z}{1-2z^{-1}e^{-LT}\cos(aT) + z^{-2}e^{-2LT}}$$

$$+ \frac{C\ (1-e^{-w_3 T})}{1-e^{-w_3 T}z^{-1}} + \frac{D\ (1-e^{-w_4 T})}{1-e^{-w_4 T}z^{-1}}$$

$$a = w_1^{\,2} - L^2$$

$$b_1 = \frac{e^{-LT}\sin(aT)}{a}$$

$$b_2 = -b_1$$

$$b_3 = 1 - e^{-LT}\left(\cos(aT) + \frac{L*\sin(aT)}{a}\right)$$

$$b_4 = e^{-2LT} + e^{-LT}\left(\frac{L*\sin(AT)}{a} - \cos(aT)\right)$$

Enligt gällande regler för överföring av signal i linjära system vet vi att utsignalen är lika med insignalen faltat med pulssvaret. I z-planet motsvarar detta den betydligt mer angenäma operationen multiplikation, det vill säga $Y(z) = H(z)W(z)$.

Som tidigare nämnts, så har vi på ingången till filtret en AD-omvandlad signal w(k). Uppgiften är nu att på lämpligaste sätt kombinera vårt H(z) med den samplade signalen. För att lösa detta problem tar vi till följande trick.

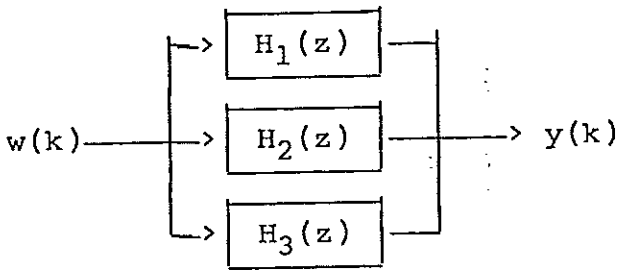
$z^{-n}\ W(z)$ - >w(k-n)

Vårt slutliga uttryck kommer då att se ut såsom följer.

$$y(k) - a_1y(k - 1) - .. a_ny(k - n) = b_1w(k) + .. + b_nw(k - n)$$

Detta är en form som ett datorprogram med lätthet kan behandla. Koefficienterna matas till exempel in i en vektor varefter man i en lämplig loop utför filtreringen.

För att öka överskådligheten och hanterbarheten delas filtret upp på tre parallellkopplade systemfunktioner. Vid olika simuleringar har det också visat sig att denna form är klart att föredra framför en seriekopplad variant. Beroende på de stora tidskonstantskillnaderna kan den seriella lösningen ge ett kraftigt instabilt filter.




$H_1(z)$ är en sammanslagning av de två första bråken varefter det tredje och fjärde bråket motsvarar, i tur och ordning, $H_2(z)$ och $H_3(z)$.

$$H_1(z) = \frac{(Ab_1 + \frac{B}{w_1}^2b_3) + (Ab_2 + \frac{B}{w_1}^2b_4)z}{1 - 2z^{-1}e^{-LT}\cos(aT) + z^{-2}e^{-2LT}}$$

Nu återstår endast identifiering av filterkoefficienterna, så att vi kan beskriva filterna enligt följande ekvation.

$$y(k) = b(n)w(k - n) + a(n)y(k - n)$$

Den fullständiga utsignalen y(k) beräknas sedan helt enkelt genom att addera de tre delfilternas utsignaler.

$$y(k) = y_1(k) + y_2(k) + y_3(k)$$

Filterkoefficienterna:

Filter 1

$b_1 = Ab_1 + \frac{B}{w_1}^2 b_3$ $\qquad A = 2e^{-LT}\cos(aT)$

$b_2 = Ab_2 + \frac{B}{w_1}^2 b_4$ $\qquad A = -e^{-2LT}$

Filter 2

$B = C(1 - e^{-w_3 T})$ $\qquad A = e^{-LT}$

$B = 0,0$ $\qquad A = 0,0$

Filter 3

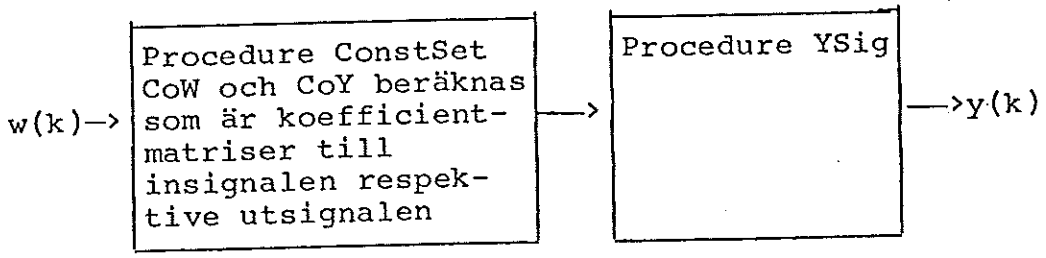
$a = D(1 - e^{-w_4 T})$ $\qquad A = e^{-LT}$

$a = 0,0$ $\qquad A = 0,0$

### 3.2.2 Programvara

Modula-2

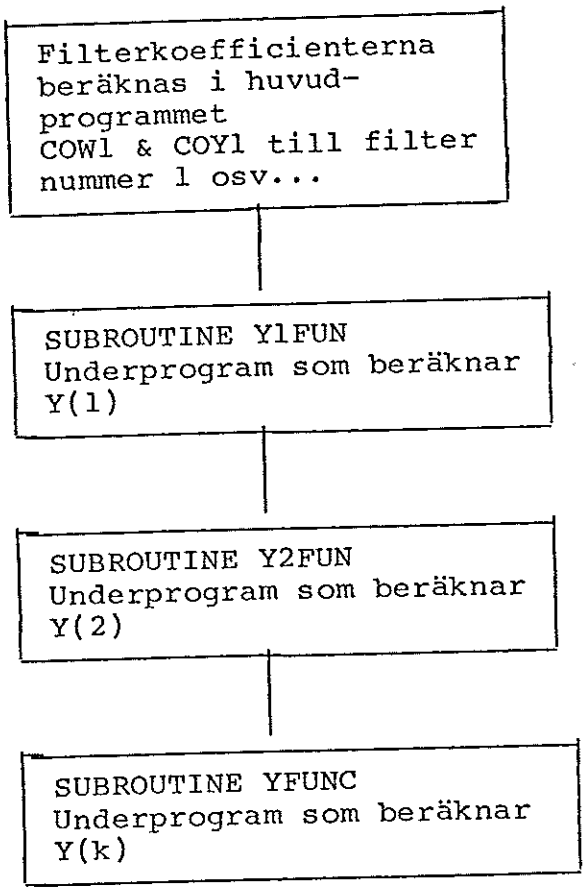
Vårt digitala filter är det ena definitions- och implementationsparet i vår realisering. I definitionsmodulen definieras de konstanter, typer, variabler och procedurer, som sänds på export till huvudprogrammet. Implementationsmodulen redovisar den totala programkoden. I ett blockschema kan programmet ses som två boxar, där den första initierar filtrets koefficienter och den andra beräknar utsignalen.




Insignalen till filtret hämtas från den vektor som bildats i huvudprogrammets inmatningsprocedur. Den kompletta programkoden finns presenterad i bilaga A. 3

Fortran

Fortranversionen har förstås samma mål, men block-schemat ser lite annorlunda ut.




Den fullständiga programkoden finns att beskåda i bilaga 5.

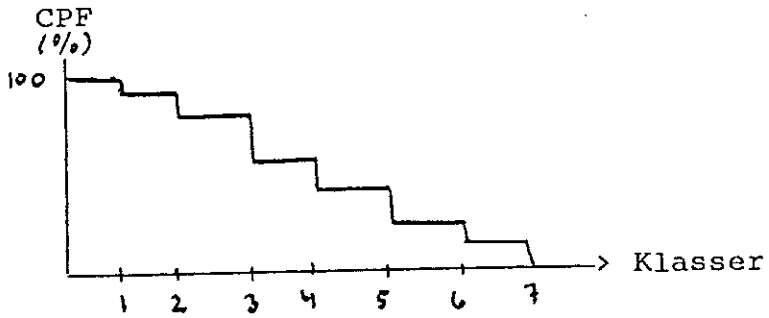
## 3.3 Statistisk behandling

### 3.3.1 Teoretisk bakgrund

Avsikten med den statistiska behandlingen är att oberoende av störningskällan ge en objektiv indikation på om flickret kommer att uppfattas som störande eller inte. Metoden ska därför kunna fungera på såväl ljusbågsugnar som vanliga hushållsmaskiner. Den enda möjliga skillnaden skulle kunna vara vid vilken nivå, som flickret inte anses tolerabelt. Dessutom blir resultatet mer överskådligt och betydligt enklare än om man endast studerade signalen från bandpassfiltret.

Flickret är en tämligen stokastisk signal, den kan uppnå mycket höga värden för ett ögonblick för att sedan återgå till en mer normal nivå. Det är därför

nödvändigt att utveckla en statistisk metod som inte bara kontrollerar den maximala nivån, utan också hur stor del av mättiden, som varje flickervärde motsvarar. För att detta ska uppnås krävs det dels en tabell som delar in flickernivåerna i olika klasser, dels en tabell som anger klassens procentuella varaktighet. Dessa två tabeller kommer nu att läggas till grund för upprättandet av den kumultativa sannolikhets- funktionen, på engelska CPF (Cumultitative Probability Function). Ur denna funktion som beskrivs i figur nedan finner vi sedan våra statistiska mått.




Vid experiment med olika laststörningar visar det sig ganska snart att CPF-kurvornas utseende är tämligen olika. Därför måste ett medel finnas för att kunna beskriva dem så objektivt som möjligt. Om detta uppnås kunde flickersignalen analyseras på ett meningsfullt vis. Sålunda utvecklade man en algoritm som motsvarade önskemålen. Detta ledde fram till uttrycket nedan.

$$P_{st} = K_1P_1 + K_2P_2 + \cdots K_nP_n$$

$P_{st}$ = Värdet av graden på flickret, Pst >1 oacceptabelt

$K_n$ = Vägningskoefficienter

$P_n$ = CPF-nivåer med en bestämd sannolikhet att inträffa

De slutliga nivåerna, som man bestämde sig för, blev

$P_{0,1}$ =nivån som överskridits vid 0,1 % av observationstiden.

$P_1$ =nivån som överskridits vid 1,0 % av observationstiden.

$P_3$ =nivån som överskridits vid 3,0 % av observationstiden.

$P_{10}$= nivån som överskridits vid 10,0 % av observationstiden.
$P_{50}$= nivån som överskridits vid 50,0 % av observationstiden.

Punkten för 50 % anger den flickernivå, som överskridits under halva observationstiden. De övriga punkterna är tagna vid nivåer, som motsvarar högre värden av flickret, vilka också är troligare att utöva bidrag till oönskade störningar.

Nästa steg är nu att definiera observationstidens längd. Här önskas också ett mått som är oberoende av störkällan. Utgående från tidigare försök på människor har man funnit att tio minuter är en god kompromiss. Tiden är tillräckligt lång, för att flickret ska märkas och samtidigt så kort, att detaljerade beräkningar med tidskrävande procedurer är meningsfulla.

För att uttrycket ska bli komplett, återstår nu endast att definiera vägningskoefficienternas värden. Återigen utfördes experiment med hjälp av människor, vilket gav följande resultat.

$K_1 = 0,0314$
$K_2 = 0,0525$
$K_3 = 0,0657$
$K_4 = 0,28$
$K_5 = 0,08$

Nu kan man på ett meningsfullt och objektivt sätt ange flickret. Analysen har mynnat ut i ett värde, P, som vid värden större än ett är ett mått på att flickret nått en oacceptabel nivå.

Långtidsmätning

Tiominutersperioden som $P_{st}$-beräkningarna baseras på är lämplig vid mätningar på individuella källor typ värmepumpar och hushållsapparater. I de fall då flera källor samverkar stokastiskt, måste man därför utföra mätningar under längre tidsperioder. För att uppnå detta krav genomförs helt enkelt flera korta $P_{st}$-mätningar för att sedan med en lämplig metod beräkna $P_{LT}$ (LT = long time). Här presenteras en av de bättre.

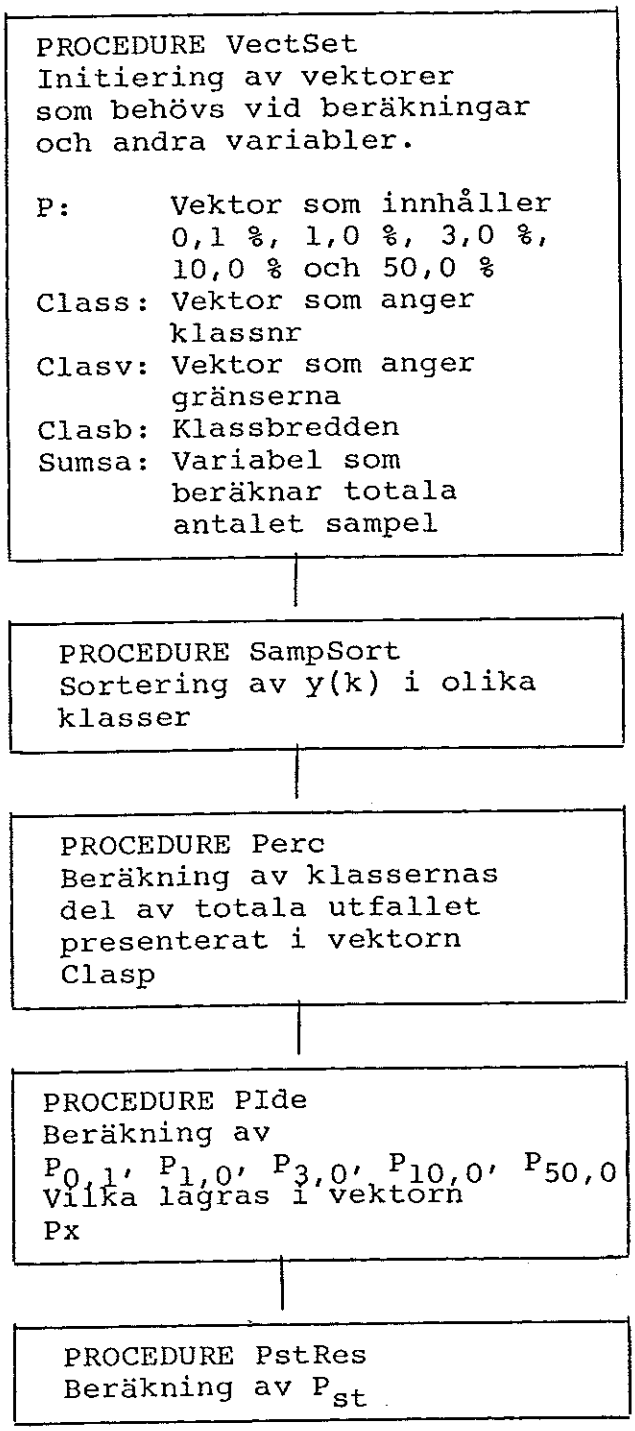
$$P_{LT} = \sqrt{\frac{\sum_{i}^{N} P_{st\ i}^3}{N}}$$

### 3.3.2 Blockschema för programvaran

Modula-2

Det andra definitions- och implementationsmodulparet är den statistiska behandlingen. Här definieras bland annat antalet klasser, som valts till 256 för att inte beräkningarna ska bli för tidsödande med bibehållen god upplösning.

Blockschema:




Vad beträffar Fortran-versionen använder den sig av samma variabler och blockstruktur som ovan, vilket föranleder mig att direkt hänvisa intresserade läsare till bilaga 3 (Modula-2) och bilaga 4 för djupare studium av den fullständiga koden.

# 4 Resultat

## 4.1 Sammanfattning

Sommarens aktiviteter har resulterat i ett digitalt bandpassfilter, som enligt UIE:s specifikationer filtrerar fram flickersignalen ur en given mätsignal. Dessutom har jag utifrån ovan nämnda specifikationer konstruerat ett statistiskt program, som analyserar graden av flicker i systemet. Samtliga nämnda komponenter finns representerade i två varianter med likvärdiga resultat.

I avdelningens IBM-PC finns flickermetern realiserad i språket Modula-2. Tanken var att PC:n skulle fungera som en fältmässig apparat, för att på plats undersöka flickret. Det visar sig dock att denna idé stöter på praktiska problem, nämligen tidsfaktorn. En insamlingsperiod motsvarande en minut ger en datorkörning på 1,5 minut. Ofta är man dock intresserad av betydligt längre tidsintervall, vilket vid tio minuters mättid motsvarar 1,5 timmes väntan på resultatet.

Den andra versionen finns implementerad i Fortran på VAX-datorn. Vi har nu lämnat kravet att mätningen ska ske ute i fält. Istället genomförs analysen på hemmaplan. Detta medför bland annat att datorkörningen minskas med en faktor fem. Mätsignalen läses nu istället från en diskett, på vilken man tidigare lagrat den intressanta signalen. Detta ställer vissa volymkrav på disketten, som måste uppfyllas. Enligt förutsättningen är det givet en signal med sampelfrekvensen 100 Hz. Detta innebär att en tiominutersmätning producerar 60 000 sampel. Om vi räknar med, att varje sampel upptar två byte behövs det en diskett med kapaciteten 720 kbyte för en timmes mätning. Det stora antalet sampel skapar problem i Modula-2 programmet då vektorerna ganska snart fylls. Modula-2 kan endast erbjuda plats för 8 000 sampel av typ "real". För att lösa detta problem måste man först t ex lagra värdena på PC:ns hårdvara. Därefter låter man programmet läsa från denna.

## 4.2 Simuleringar

UIE:s specifikationer innehöll förutom Laplaceformen för bandpassfiltret även en grafisk beskrivning över dess utseende. Detta gav mig utmärkta möjligheter att

kontrollera mitt eget resultat. Med hjälp av programpaketet IDPAC kunde jag grafiskt beskriva H(z) som en funktion av frekvensen. Genom att sedan studera kurvornas amplitudsvar fann jag, att det råder god överensstämmelse dem emellan. Läsaren kan själv övertyga sig om detta i bilaga 3.

För att kontrollera flickermeterns funktion tog jag till hjälp den av IEC specificerade flickerkurvan, som för fyrkantvågor anger maximalt tillåten flickerstyrka vid given frekvens (bilaga 1).

Detta innebär att samtliga värden ovanför kurvan ska ge ett $P_{st}$ >1. Om man istället väljer ett värde på kurvan ska $P_{st}$ bli lika med ett. För att testa om så var fallet genererade jag en fyrkantvåg, där amplituden valdes till 1 % av nätspänningen. Detta motsvarade frekvensen 0,221 Hz. Denna analys gav ett vilket får anses vara acceptabelt. Delresultaten från bandpassfiltret och det statistiska programmet lagrades i filer för att sedan med Tispacs hjälp illustreras grafiskt.

I bilaga 5 presenteras således insignalen, dess bandpassfiltrering och den kumultativa sannolikhetskurvan för mätsignalen.

Simuleringen omfattade även två andra mätsignaler med samma frekvens och amplitud men med annorlunda utseende. Mätsignal två gav ett $P_{st}$= 0,72 och mätsignal tre gav $P_{st}$ = 0,24. Samtliga signaler, deras bandpassfiltreringar och de kumultativa sannolikhetsfunktionerna finns även de grafiskt beskrivna i bilaga 5.

## Digitala filter

Vid konstruktion av digitala filter kan man välja mellan två vägar, nämligen FIR-filter (Finite-duration Impulse Response) eller IIR-filter (Infinite-duration Impulse Response). FIR-filtret karakteriseras av dess ändliga pulssvar samt att det är ett icke rekursivt filter, så kallat transversalfilter.

$$y(k) = b_0\, w(k) + b_1\, w(k-1) + \ldots + b_n$$

IIR-filtret är däremot ett rekursivt filter, som kan beskrivas med följande ekvation.

$$a_0 y(k) + a_1 y(k-1) + .. + a_m = b_0 w(k) + b_1 w(k-1) +.. + b_n w(k-n)$$

## FIR-filter

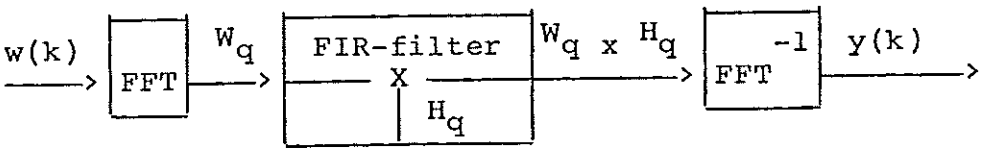
FIR-filtret har några speciella egenskaper, vilket gör det attraktivt för filtersyntes.

1 Det ändliga pulssvaret medför att de alltid är stabila.

2 Genom lämplig fördröjning kan de alltid göras kausala.

3 Icke rekursiva

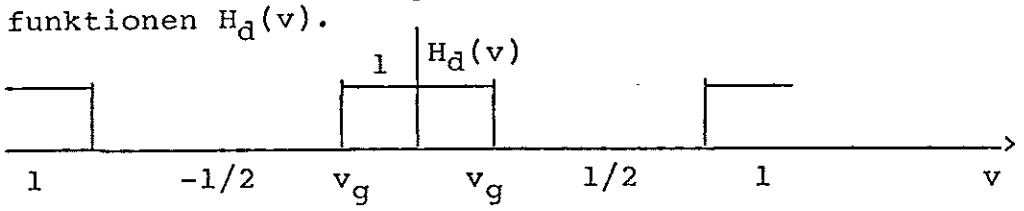
4 Realisering kan ske genom faltningssumma, beräknad med DFT, se figur.




Det finns förstås också nackdelar med FIR-filter-realiseringar. Till exempel krävs ett stort L (långt pulssvar) för att erhålla selektiva filter. Detta medför i sin tur att filtren inför en lång fördröjning. Vidare saknas enkla syntesmetoder, med vilka man direkt kan uppfylla godtyckliga toleranskrav. En metod som dock visat sig vara tämligen enkel, är den så kallade fönstermetoden.

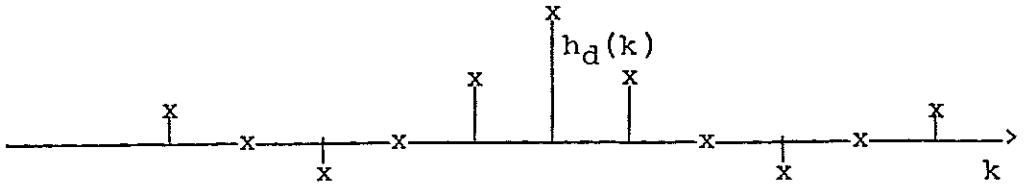
## Syntes av FIR-filter med fönstermetoden

Vid fönstermetoden utgår vi från den önskade frekvensfunktionen $H_d(v)$.




$$H_d(v) = \sum_{l=-\infty}^{\infty} h_d(l) e^{-j2\pi v l}$$

Vi bildar den inversa fourierttransformen och erhåller då pulssvaret $h_d(k)$.



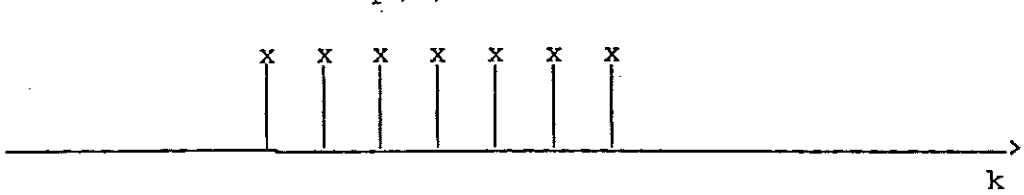

$$h_d(k) = \int_{-1/2}^{1/2} H_d(v) e^{j2\pi v k} dv$$

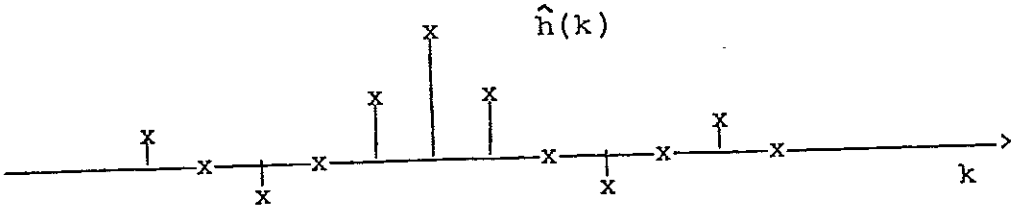
$h_d(k)$ är nu det önskade pulssvaret, men som regel svarar $h_d(k)$ mot ett icke-kausalt IIR-filter. För att en FIR approximation till $H_d(v)$ bildas därför sekvensen:

$$\hat{h}(k) = h_d(k)p(k)$$

där fönsterfunktionen $p(k)$ är en lämplig vald ändlig sekvens.

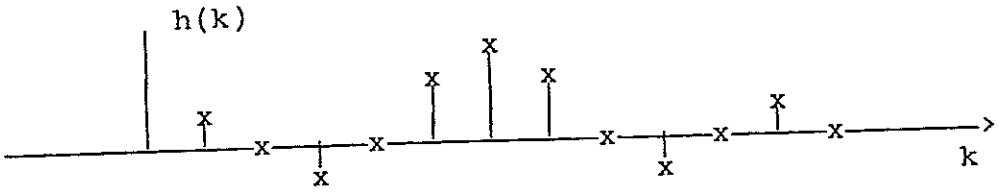
$$p(k) \begin{cases} \neq 0 & k \leq (L-1)/2 \\ = 0 & k > (L-1)/2 \end{cases}$$





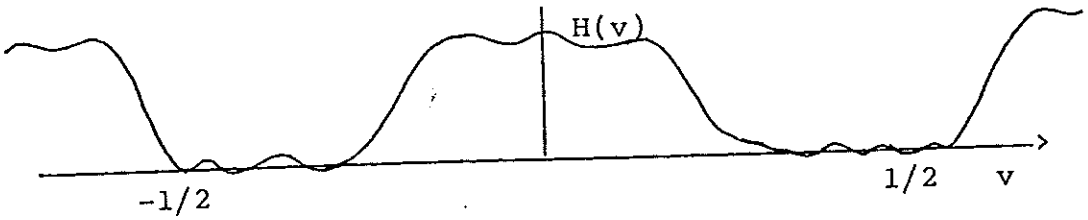

Fördröjs pulssvaret (L-1)/2 sampelintervall blir det kausalt

$$h(k) = \hat{h}(k-(L-1)/2)$$




Slutligen beräknar vi

$$H(v) = \sum_{-(L-1)/2}^{(L-1)/2} h(n)e^{-j2\pi vn}$$




Vad som nu återstår att bestämma är vilken typ av fönsterfunktion vi önskar. Till vårt förfogande har vi tre olika sorter, nämligen rektangulärt fönster, Hammingfönster samt Blackmanfönster.

1 Det rektangulära fönstret

$$P_r(k) = \begin{cases} 1 & -\frac{L-1}{2} \leq k \leq \frac{L-1}{2} \quad (L \text{ udda}) \\ 0 & \text{f.ö.} \end{cases}$$

Det rektangulära fönstrets huvudlob har bredden 2/L och sidolobernas maximinivå ligger endast 13 dB under huvudlobens. Detta innebär, att då man önskar dämpa en godtycklig signal med mer än 13 dB, får man använda sig av något fönster med bättre egenskaper, t ex Hamming-fönstret.

2 Hammingfönstret

$$P_H(k) = \begin{cases} 0,54 + 0.46 \cos \frac{2\pi k}{L} & -\frac{(L-1)}{L} \leq k \leq \frac{L-1}{2} \\ 0 \quad \text{f.ö.} \end{cases}$$

Huvudloben har bredden 4/L och dess sidolobsnivå ligger ca 45 dB under huvudlobens.

3 Blackmanfönstret

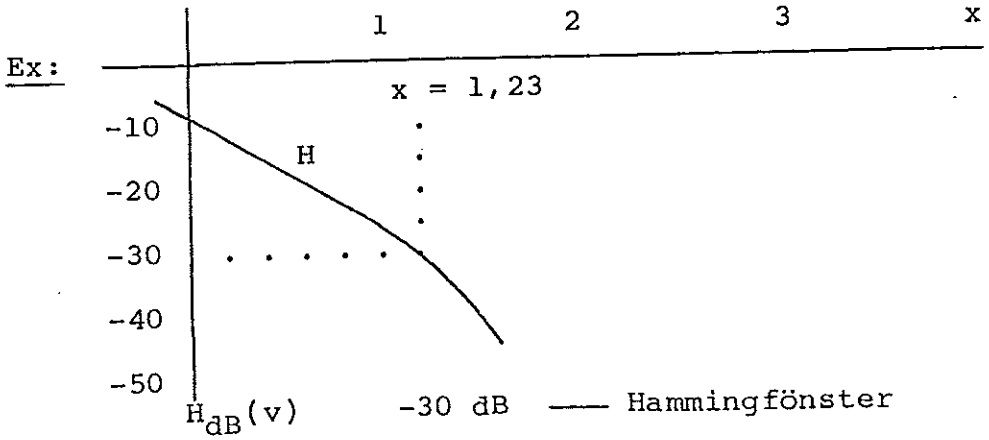
$$P_B(k) = \begin{cases} 0,42+0,5 \cos(\frac{2\pi k}{L})+0,08 \cos(\frac{4\pi k}{L}) & -\frac{(L-1)}{2} \leq k \leq \frac{L-1}{2} \\ 0 \quad \text{f.ö.} \end{cases}$$

För Blackmanfönstret är huvudlobens bredd 6/L, medan sidolobernas nivåer är ca 60 dB under huvudlobens.

- Sammanfattningsvis kan vi konstatera, att en bättre dämpning medför ett bredare fönster. Denna negativa effekt kan dock avhjälpas genom att välja ett stort L, vilket helt osökt leder oss in på spåret mot den sista pusselbiten i vår jakt på ett färdigt FIR-filter.

Beräkning av pulssvarets längd L

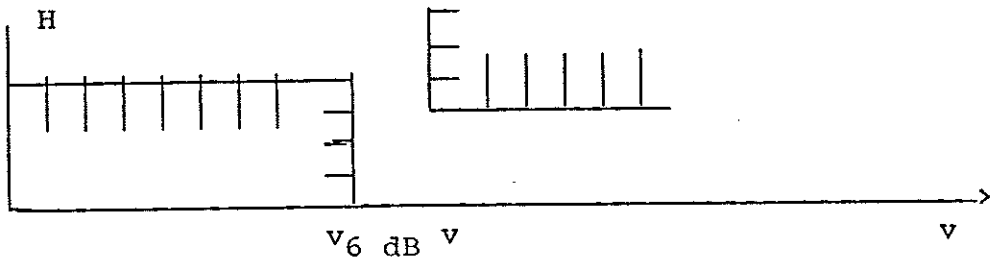
Till vår hjälp har vi diagrammet i referens 7. Där finner vi tre kurvor, en för varje fönster. När vi beräknar vårt L söker vi upp den önskade dämpningen på $H_{dB}(v)$-axeln och väljer därefter ett fönster, som motsvarar våra krav. På x-axeln finner vi då ett värde, genom att studera var fönsterkurvan skär den önskade dämpningen.



Ur ekvationen

$x = (v-v_{6\ dB})L$ (LP-filter)

kan vi sedan beräka vårt L, där v och $v_{6\ dB}$ är angivna enligt figur nedan.




## IIR-filter

IIR-filter karakteriseras av att de är rekursiva. De har dessutom oändligt långt pulssvar. Vi ska också anta att de är kausala och stabila.

Arbetsgången vid konstruktion av IIR-filter är oftast att man först definierar en analog systemfunktion,

$$H^c(s) = \frac{d_0 + d_1 s + \ldots + d_m s^m}{1 + C_1 S + \ldots + C_n S^n}$$

vilken uppfyller de ställda kraven.

Därefter gör man en enkel transformation som avbildar s-planet på z-planet. Vi ska här behandla en metod som kallas impulsinvariantmetoden, men först några ord om hur man definierar den analoga systemfunktionen.

## Konstruktion av analoga filter

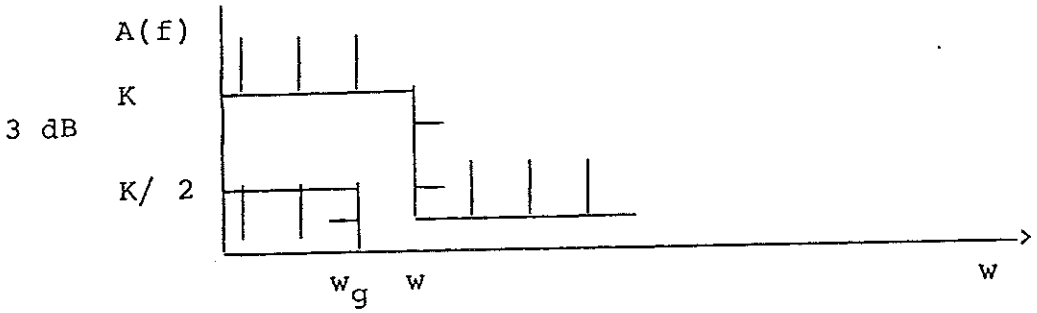
Det ideala LP-filtret är inte fysikaliskt realiserbart, därför uppstår problemet hur man på bästa sätt ska approximera det med en rationell överföringsfunktion, som enkelt kan realiseras av en krets. Vi ska betrakta två approximationer, nämligen Butterworth och Chebyshev.

## Butterworthfilter

Vid konstruktion av Butterworthfilter utgår vi från dess amplitudfunktion.

$$A(f) = \frac{K}{\sqrt{1 + \left(\frac{w}{w_g}\right)^{2n}}}$$

K = amplitudfunktionens maximivärde (w = 0)




I amplitudfunktionen finner vi endast en okänd parameter, filtrets ordning n. Denna bestäms med hänsyn till dämpningskravet. För det erhållna värdet, närmast övre heltal, går man in i tabell (referens 7) och finner sina koefficienter till överföringsfunktionen H(s).

$$H(s) = \frac{K}{\left(\frac{s}{w_g}\right)^{n} + a_{n-1}\left(\frac{s}{w_g}\right)^{n-1} + \ldots + a_1\left(\frac{s}{w_g}\right) + 1}$$

## Chebyshevfilter

Även vid konstruktion av Chesbyshevfilter utgår vi från amplitudfunktionen.

$$A(f) = \frac{K}{\sqrt{1 + e^2 C_n^{\,2}\left(\frac{w}{w_g}\right)}}$$

e = ripplet i passbandet
Cn = Chesbyshev-polynom

Innan vi kan bestämma filtrets ordning löser vi först ut Chesbyshevpolynomet $C_n(w)$, därefter kan vi ur följande ekvationer lösa ut n.

$$C_n(w) = \begin{cases} \text{Cos}\ (n \arccos(w)) & |w| \leq 1 \\ \text{Cosh}\ (n\ \text{arcosh}(w)) & |w| \geq 1 \end{cases}$$

$$\text{arcosh}(x) = \text{In}(x + x^2-1)$$

Med parametrarna och n kända kan vi liksom tidigare erhålla koefficienternas värde ur tabell (bilaga 3) och vår systemfunktion är åter känd.

$$H(s) = \frac{K}{\left(\frac{s}{wg}\right)^n - a_{n-1}\left(\frac{s}{wg}\right)^{n-1} + \ldots a_o}$$

IIR-filter

Vi har nu tillverkat oss en analog systemfunktion, nästa steg är att översätta den till den tidsdiskreta systemfunktionen H(z). Detta gör vi med impulsinvariant transformation.

Impulsvariant transformation

Kort sagt innebär denna metod att man vandrar från s-planet till tidsplanet,

$$h(t) = \frac{1}{j2\pi} \int H^c(s)\ e^{st} ds$$

bildar det tidsdiskreta pulssvaret h(k)

$$h(k) = Th^c(kT)$$ (T = sampelintervallet)

varefter vi ur tabell (referens 4) räknar fram H(z). Denna metod lämpar sig tyvärr endast för begränsade filter, LP- och BP-filter.

## Realisering av filter i datorer

Slutligen återstår det bara att översätta våra filter till ett språk som datorn förstår. Låt oss som exempel realisera ett FIR-filter med L + 1 st koefficienter. B(0), B(1), ...B(L). Insignalen är given för M sampel, X(1), X(2) ...X(M). Vi söker utsignalens motsvarande M sampel, Y(1), Y(2), ...Y(M) och förutsätter att systemet från början var i vila. Sambandet

$$Y(K) = \sum_{n=0}^{L} B(n)X(k-n)$$

ska realiseras för k = 1,2 .... M. I Fortran får filtret följande utseende.

```
DO 10 K = 1, M
DO 20 N = 1, min (L, K-1)
Y(K) = Y(K) + B(N)X(K-N)
20 CONTINUE
10 CONTINUE
STOP
END
```

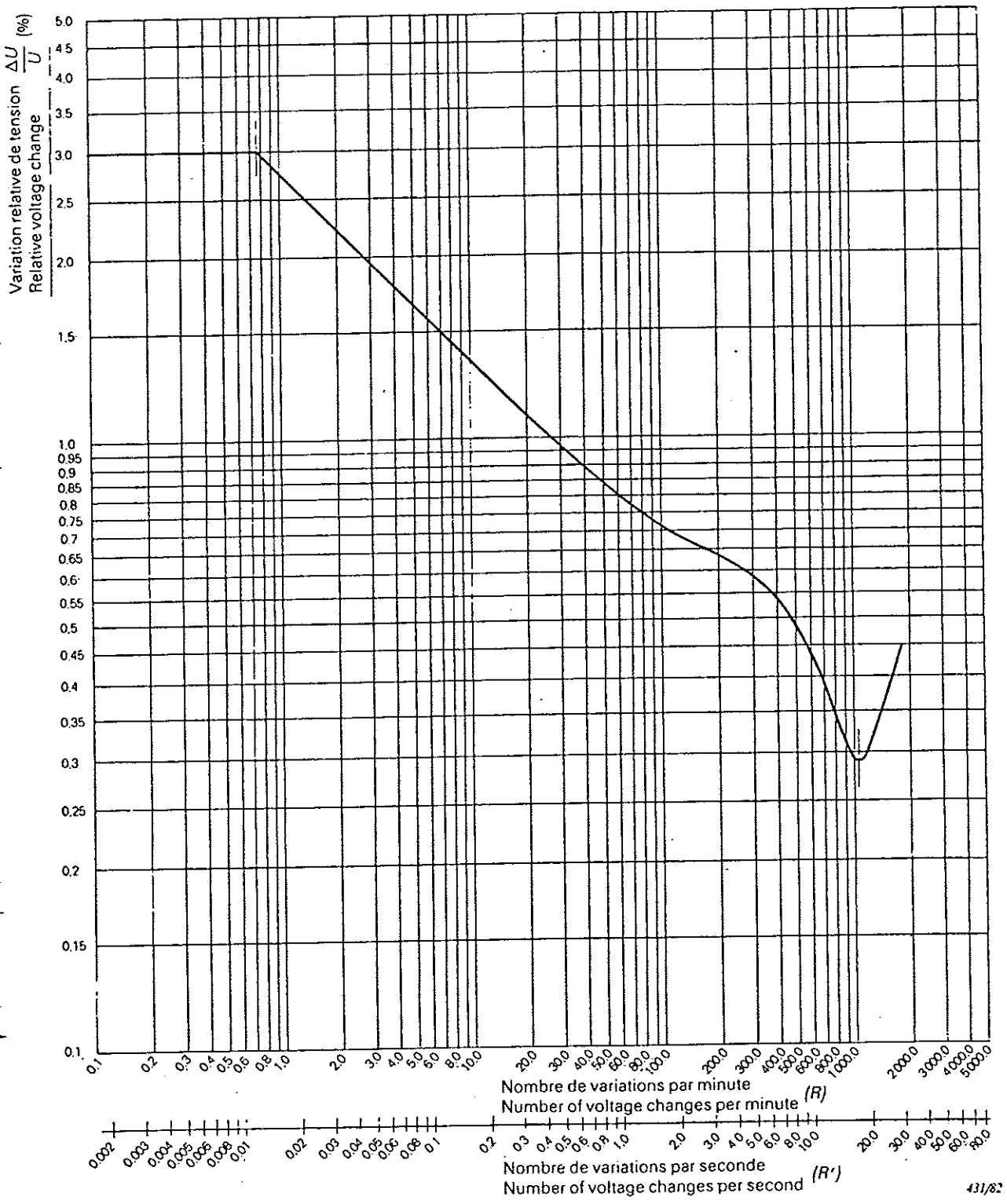



FIG. 4a. — Amplitude des variations maximales admissibles de tension relatives $\Delta U/U$(%) rapportées au nombre de variations de tension par seconde ou par minute.

Magnitude of maximum permissible percentage voltage changes $\Delta U/U$(%) with respect to number of voltage changes per second or per minute.

*Note.* — Pour les coordonnées de points donnés, voir tableau II, page 22. La limite de 3% de la variation de tension est valable à partir d'une variation par heure (0,0167 variation par minute) jusqu'à 0,76 variation par minute.

See Table II, page 23, for the co-ordinates of particular points. The 3% limit of voltage change applies from one change per hour (0.0167 change per minute) to 0.76 change per minute.

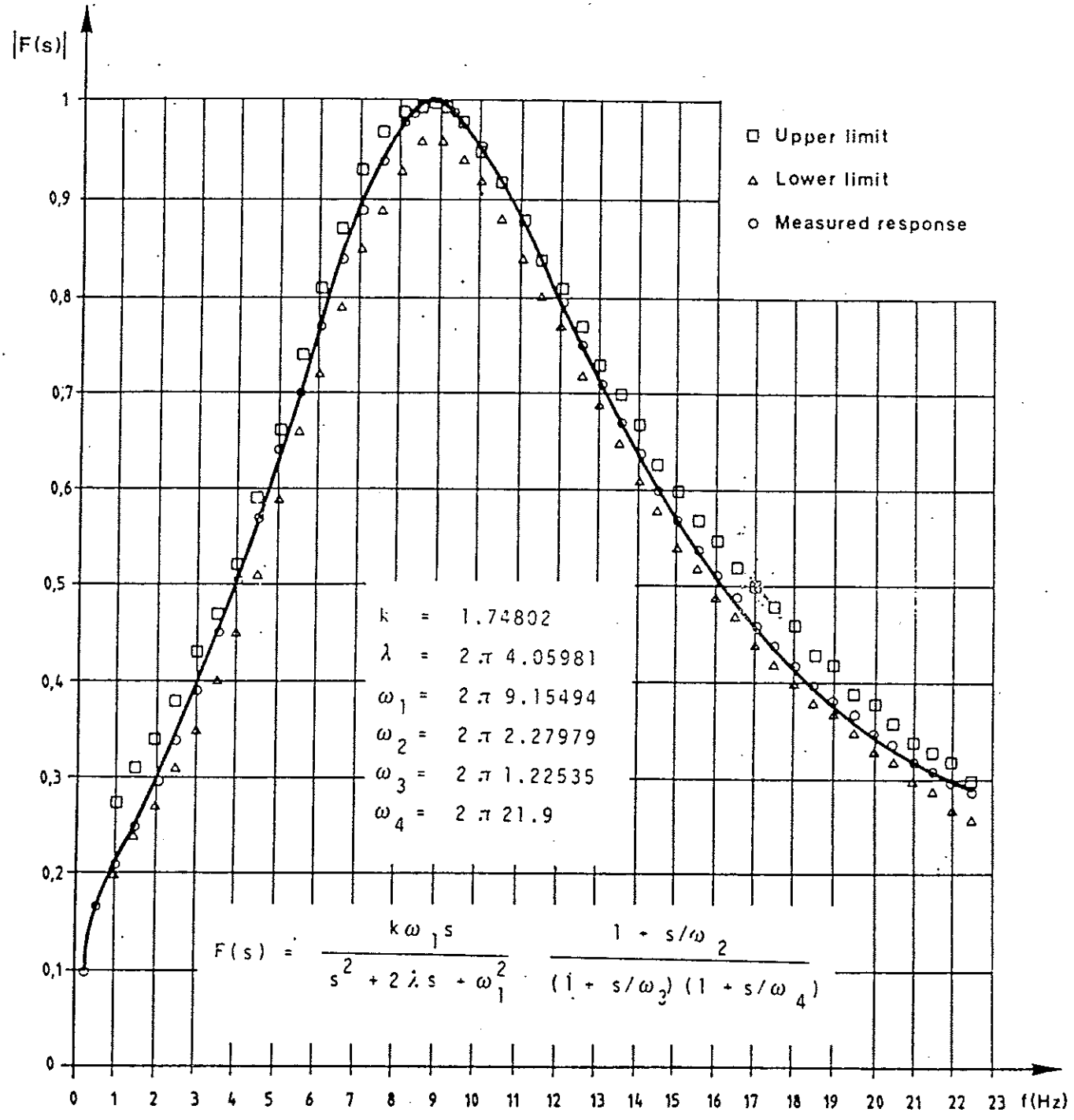
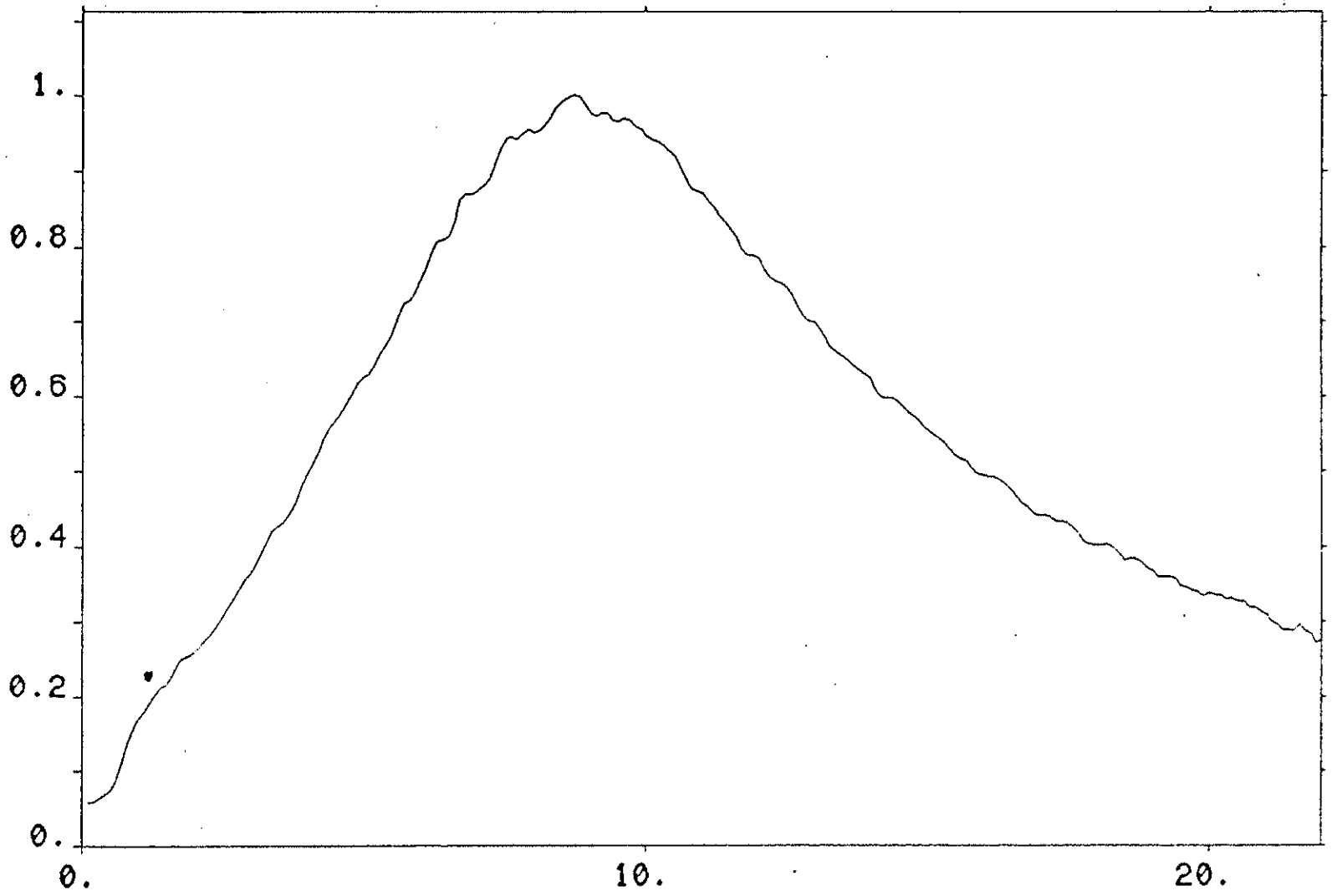



Fig. 3.2 - Normalised amplitude response curve of the flickermeter filter fitted to the specified band.

>




PLOT FRE < H87.09.17 - 08:45:06

```
MODULE Flickermeter;
 IMPORT FileSystem;
 FROM Kernel IMPORT
      MaxPriority,Time,InitKernel,
      CreateProcess,SetPriority,ThisProcess,
      IncTime,WaitUntil,GetTime,Priority;
 FROM Terminal IMPORT WriteString,WriteLn,ReadString;
 FROM NumberConversion IMPORT StringToCard,IntToString;
 FROM ConvReal IMPORT RealToString,StringToReal;
 FROM MathLib IMPORT sin,float,round;
 FROM DigFilt IMPORT
      ConstSet,YSig,CoW,CoY,Y,SigTyp;
 FROM Stat IMPORT
      VectSet,ClasI,P,Class,Clasv,Clasb,Sumsa,
      SampSort,Perc,PIde,PstRes,Clasp,Px,ClassTyp;
 CONST
  SamFre=100;
  SamTid=0.01;     (*Insignalen samplingstid*)
 TYPE
  MinTyp=ARRAY [1..5] OF CHAR;
 VAR
  W:SigTyp;         (*Insignalen*)
  sig:ARRAY [1..10] OF CHAR;
  pma:ARRAY [1..5] OF CHAR;
  minuter,short:MinTyp;
  AntSam,AntPst,tid,tsh,pos:CARDINAL;
  done:BOOLEAN;
  I,k,min,max,i,j:INTEGER;
  Yk,Amp:REAL;
 PROCEDURE MesSig(VAR W:SigTyp);
 (*Insamling av mtsignalen frn fil*)
 VAR  f:FileSystem.File;
      w,po:CARDINAL;
 BEGIN
   w:=6;
   FileSystem.Lookup(f,'c:data.mat',TRUE);
   FOR i:=1 TO AntSam DO
     po:=0;
     FOR j:=1 TO 10 DO FileSystem.ReadChar(f,sig[j]) END;
     StringToReal(sig,po,W[i]);
     RealToString(W[i],sig,w);
     WriteString(sig);
    END;
   FileSystem.Close(f);
 END MesSig;
 PROCEDURE FiltOSort(VAR Sumsa:INTEGER;VAR Class:ClassTyp);
 BEGIN
  YSig(CoW,CoY,W,Y,AntSam);   (*Berkning av utsignalen*)
  pos:=9;
  FOR I:=1 TO AntSam DO
   RealToString(Y[I],sig,pos);
   WriteString(sig);
  END;
  WriteString("Filtrering avslutad");
  pos:=0;
  WriteString("amplitud?");
  ReadString(pma);
  StringToReal(pma,pos,Amp);
  VectSet(P,Class,Clasv,Clasb,Sumsa,Amp);  (*Vektor implementetion i Stat*)
```

```
  WriteString("Klassbredd=");
  pos:=6;
  RealToString(Clasb,sig,pos);
  WriteString(sig);
  FOR I:=1 TO Clasl DO
   RealToString(Clasv[I],sig,pos);
   WriteString(sig);
  END;
  WriteString("Nu brjar sorteringen");
  FOR I:=1 TO AntSam DO
   Yk:=Y[I];
   IF (Yk<=Amp) THEN
    SampSort(Class,Clasv,Amp,Yk,Sumsa,min,max); (*Sortering av utsig i klasse
   END;
  END;
END FiltOSort;
BEGIN
  ConstSet(CoW,CoY);                    (*Berkning av filtrets koefficienter*)
  WriteString("Vlj antal sampel");
  ReadString(short);
  StringToCard(short,AntSam,done);
  InitKernel();          (*Pst fr att uppfylla kraven*)
  MesSig(W);
  WriteString("Filtrering brjar");
  FiltOSort(Sumsa,Class);
  WriteString("Antalet sampel i klasserna");
  FOR I:=1 TO Clasl DO
   IntToString(Class[I],sig,pos);
   WriteString(sig);
  END;
  WriteString(",Sortering avslutad");
  WriteLn;
  Perc(Class,Clasp,Sumsa);
  PIde(Clasp,Clasv,P,Px,Clasb);         (*Statistiska berkningar*)
  PstRes(Px);
END Flickermeter.
```

```
IMPLEMENTATION MODULE DigFilt;
 FROM MathLib IMPORT sqrt,exp,sin,cos,float,round;
 FROM Terminal IMPORT WriteString,Write;
 PROCEDURE ConstSet(VAR CoW,CoY:ConTyp);
 (*Berkning av koefficienterna till filtret*)
 CONST
  X=1000;
  T=0.01;      (*Systemets samplingstid*)
  K=1.74802;   (*Konstant specificerad av IEC*)
  PI=3.14159;
 VAR
  B1,B2,B3,B4,Alf,Rot,W1,W2,W3,W4,L,A,B,C,D,N,U:REAL;
 BEGIN
  W1:=2.0*PI*9.15494;
  W2:=2.0*PI*2.27979;    (*Konstanter specificerade i IEC:s laplace*)
  W3:=2.0*PI*1.22535;    (*verfringsfunktion*)
  W4:=2.0*PI*21.9;
  L:=2.0*PI*4.05981;
  U:=(K*W1*W4/W2)-K*W1;
  N:=(2.0*L*W4/W3)-(W4*W4/W3)-(W1*W1/W3)-2.0*L+W4+(W1*W1/W4);
  D:=U/N;
  C:=(K*W1-(2.0*L-W4-W1*W1/W4)*D)/(2.0*L-W3-W1*W1/W3);
  B:=W1*W1*(-C-D);
  A:=-(W3*C+W4*D);
  Rot:=W1*W1-L*L;
  Alf:=sqrt(Rot);
  B1:=exp(-L*T)*sin(Alf*T)/Alf;
  B2:=-B1;
  B3:=1.0-exp(-L*T)*(cos(Alf*T)+L*sin(Alf*T)/Alf);
  B4:=exp(-2.0*L*T)+exp(-L*T)*(L*sin(Alf*T)/Alf-cos(Alf*T));
  CoW[1,1]:=A*B1+B*B3/(W1*W1);
  CoW[1,2]:=A*B2+B*B4/(W1*W1);
  CoW[2,1]:=C*(1.0-exp(-W3*T));
  CoW[2,2]:=0.0;
  CoW[3,1]:=D*(1.0-exp(-W4*T));
  CoW[3,2]:=0.0;
  CoY[1,1]:=2.0*cos(Alf*T)*exp(-L*T);
  CoY[1,2]:=-exp(-2.0*L*T);
  CoY[2,1]:=exp(-W3*T);
  CoY[2,2]:=0.0;
  CoY[3,1]:=exp(-W4*T);
  CoY[3,2]:=0.0;
 END ConstSet;
 PROCEDURE YSig(VAR CoW,CoY:ConTyp;VAR W,Y:SigTyp;VAR AntSam:CARDINAL);
 (*procedur som berknar den filtrerade signalen*)
 TYPE
  YxTyp=ARRAY [1..2] OF REAL;
 VAR
  in1,in2:REAL;
  rad,sam:INTEGER;
  Y1,Y2,Y3:YxTyp;          (*Utsignalen frn vart och ett av de*)
```

```
PROCEDURE YxSet(VAR Yx:YxTyp; rad,sam:INTEGER);
VAR                              (*parallellkopplade filtrena*)
 YW,YY:REAL;
 kol:INTEGER;
BEGIN
 YW:=0.0;
 YY:=0.0;
 FOR kol:=0 TO 1 DO
  YW:=YW+CoW[rad,kol+1]*W[sam-kol];
 END;
 FOR kol:=1 TO 2 DO
  YY:=YY+CoY[rad,kol]*Yx[3-kol];
 END;
 Yx[1]:=Yx[2];
 Yx[2]:=YW+YY;
END YxSet;
BEGIN
 in1:=W[1];
 in2:=W[2];
 Y1[1]:=CoW[1,1]*in1;
 Y2[1]:=CoW[2,1]*in1;
 Y3[1]:=CoW[3,1]*in1;
 Y[1]:=Y1[1]+Y2[1]+Y3[1];
 Y1[2]:=CoW[1,1]*in2+CoW[1,2]*in1+CoY[1,1]*Y1[1];
 Y2[2]:=CoW[2,1]*in2+CoW[2,2]*in1+CoY[2,1]*Y2[1];
 Y3[2]:=CoW[3,1]*in2+CoW[3,2]*in1+CoY[3,1]*Y3[1];
 Y[2]:=Y1[2]+Y2[2]+Y3[2];
 FOR sam:=3 TO AntSam DO
  rad:=1;
  YxSet(Y1,rad,sam);     (*Berkning av utsignalen genom filter nr rad och*)
  rad:=2;                (*fr sampel nr sam*)
  YxSet(Y2,rad,sam);
  rad:=3;
  YxSet(Y3,rad,sam);
  Y[sam]:=Y1[2]+Y2[2]+Y3[2];  (*Utsignal vid sampel nr sam*)
 END;
 WriteString("Filtrering avslutad");
 END YSig;
END DigFilt.
```

```
DEFINITION MODULE DigFilt;
 EXPORT QUALIFIED ConstSet,YSig,CoW,CoY,Y,SigTyp;
 CONST
  MaxInt=6000;
 TYPE
  ConTyp=ARRAY [1..3],[1..2] OF REAL;
  SigTyp=ARRAY [1..MaxInt] OF REAL;
 VAR
  Y:SigTyp;       (*Utsignalen frn filtret*)
  CoW,CoY:ConTyp; (*Koefficientmatriser till filtret*)
 PROCEDURE ConstSet(VAR CoW,CoY:ConTyp);
 (*Berkning av koefficienterna till filtret*)
 PROCEDURE YSig(VAR CoW,CoY:ConTyp;VAR W,Y:SigTyp;VAR AntSam:CARDINAL);
 (*Berkning av den fullstndiga utsignalen Y*)
END DigFilt.
```

```
IMPLEMENTATION MODULE Stat;
 FROM MathLib IMPORT sqrt,float,round,entier;
 FROM Terminal IMPORT ReadString,WriteString,WriteLn;
 FROM ConvReal IMPORT RealToString,StringToReal;
 CONST
  X=1000;
 PROCEDURE VectSet(VAR P:PTyp;VAR Class:ClassTyp;VAR Clasv:ClasvTyp;
                   VAR Clasb:REAL;VAR Sumsa:INTEGER;Amp:REAL);
 (*Initiering av vektorerna samt Clasb och Sumsa*)
 VAR
  I:INTEGER;
  pos:CARDINAL;
  pma:ARRAY [1..5] OF CHAR;
 BEGIN
  Sumsa:=0;
  Clasb:=2.0*Amp/float(Clasl);  (*Berkning av klassbredden*)
  Class[1]:=0;
  Clasv[1]:=Clasb;
  FOR I:=2 TO Clasl DO
   Class[I]:=0;
   Clasv[I]:=Clasv[I-1]+Clasb;
  END;
  P[1]:=0.01;
  P[2]:=1.0;
  P[3]:=3.0;
  P[4]:=10.0;
  P[5]:=50.0;
 END VectSet;
 PROCEDURE SampSort(VAR Class:ClassTyp;VAR Clasv:ClasvTyp;Amp,Yk:REAL;
                    VAR Sumsa:INTEGER;min,max:INTEGER);
 (*Sortering av utsignalen i klasser utgende frn dess amplitud*)
 VAR
  I:INTEGER;
 BEGIN
  I:=entier((Yk+Amp)/Clasb)+1;
  Class[I]:=Class[I]+1;
  Sumsa:=Sumsa+1;
 END SampSort;
 PROCEDURE Perc(VAR Class:ClassTyp;VAR Clasp:ClasvTyp; Sumsa:INTEGER);
 (*Berkning av klassernas del av totala utfallet*)
 VAR
  I:INTEGER;
 BEGIN
  FOR I:=1 TO Clasl DO
   Clasp[I]:=100.0*float(Class[I])/float(Sumsa);
  END;
 END Perc;
 PROCEDURE Plde(VAR Clasp,Clasv:ClasvTyp;VAR P,Px:PTyp; Clasb:REAL);
 (*Berkning av Px*)
 VAR
  Z,Cl:INTEGER;
  Pk,Pro:REAL;
 BEGIN
  Cl:=256;
  Pro:=0.0;
  FOR Z:=1 TO 5 DO
   WHILE Pro<P[Z] DO     (*Algoritm som sker upp den klass dr den*)
    Cl:=Cl-1;
    Pk:=Pro;             (*nskade procenten hr hemma*)
    Pro:=Pro+Clasp[Cl];
   END;
   Px[Z]:=(Clasv[Cl]+Clasv[Cl-1])/2.0;
  END;
```

```
  END;
 END PIde;
 PROCEDURE PstRes(VAR Px:PTyp);
 (*Brkning av Pst*)
 VAR
  w:CARDINAL;
  s:ARRAY [1..100] OF CHAR;
 BEGIN
  Pst:=sqrt(K1*Px[1]+K2*Px[2]+K3*Px[3]+K4*Px[4]+K5*Px[5]);  (*Berkning av*
                                                              (*Pst*)
  w:=6;
  WriteString('Pst=');
  RealToString(Pst,s,w);
  WriteString(s);
 END PstRes;
END Stat.
```

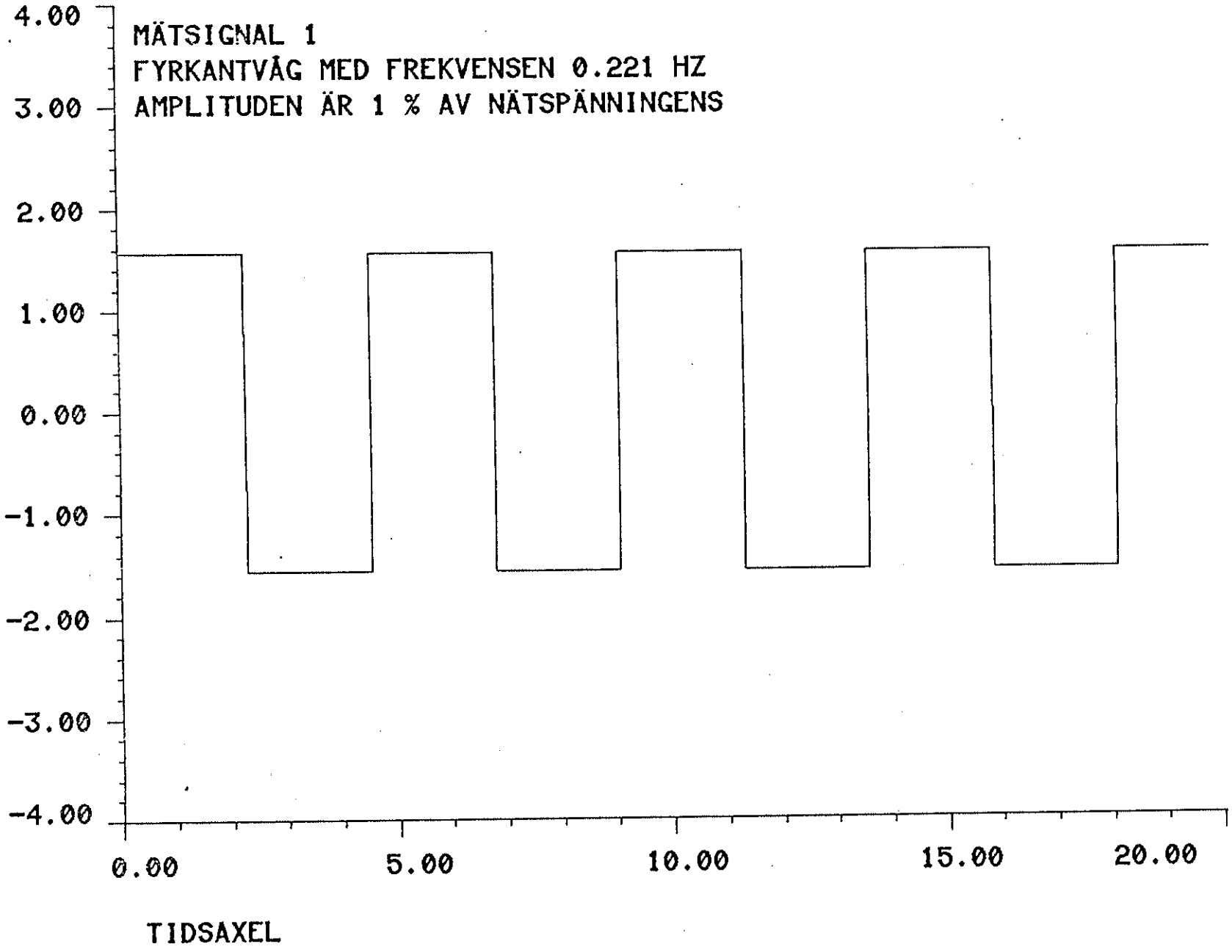
```
DEFINITION MODULE Stat;
 EXPORT QUALIFIED
    VectSet,Clasl,P,Class,Clasv,Clasb,Sumsa,
    SampSort,Perc,PIde,PstRes,Pst,Clasp,Px,ClassTyp;
 CONST
  Clasl=256;     (*Antalet klasser*)
  K1=0.0314;     (*Konstantewr specificerade av IEC*)
  K2=0.0525;
  K3=0.0657;
  K4=0.28;
  K5=0.08;
 TYPE
  ClassTyp=ARRAY [1..Clasl] OF INTEGER;
  ClasvTyp=ARRAY [1..Clasl] OF REAL;
  PTyp=ARRAY [1..5] OF REAL;
 VAR
  Class:ClassTyp;          (*Vektor som berknar antalet sampel inom varje klas
  Clasv,Clasp:ClasvTyp;    (*Clasv=vektor som anger klassgrnserna*)
  P,Px:PTyp;               (*Clasp=vektor som visar den % av varje klass*)
  Sumsa:INTEGER;           (*Sumsa=Totala antalet sampel*)
  Clasb,Pst:REAL;          (*Clasb=Klasbredden*)
 PROCEDURE VectSet(VAR P:PTyp;VAR Class:ClassTyp;VAR Clasv:ClasvTyp;
                   VAR Clasb:REAL;VAR Sumsa:INTEGER;Amp:REAL);
 (*Initiering av vektorerna samt Clasb och Sumsa*)
 PROCEDURE SampSort(VAR Class:ClassTyp;VAR Clasv:ClasvTyp;Amp,Yk:REAL;
                    VAR Sumsa:INTEGER;min,max:INTEGER);
(*Sortering av sampelvrdena*)
 PROCEDURE Perc(VAR Class:ClassTyp;VAR Clasp:ClasvTyp; Sumsa:INTEGER);
 (*Berknig av klassernas del av totala utfallet*)
 PROCEDURE PIde(VAR Clasp,Clasv:ClasvTyp;VAR P,Px:PTyp; Clasb:REAL);
 (*Berkning av Px*)
 PROCEDURE PstRes(VAR Px:PTyp)
 (*Berkning av Pst*)
END Stat.
```

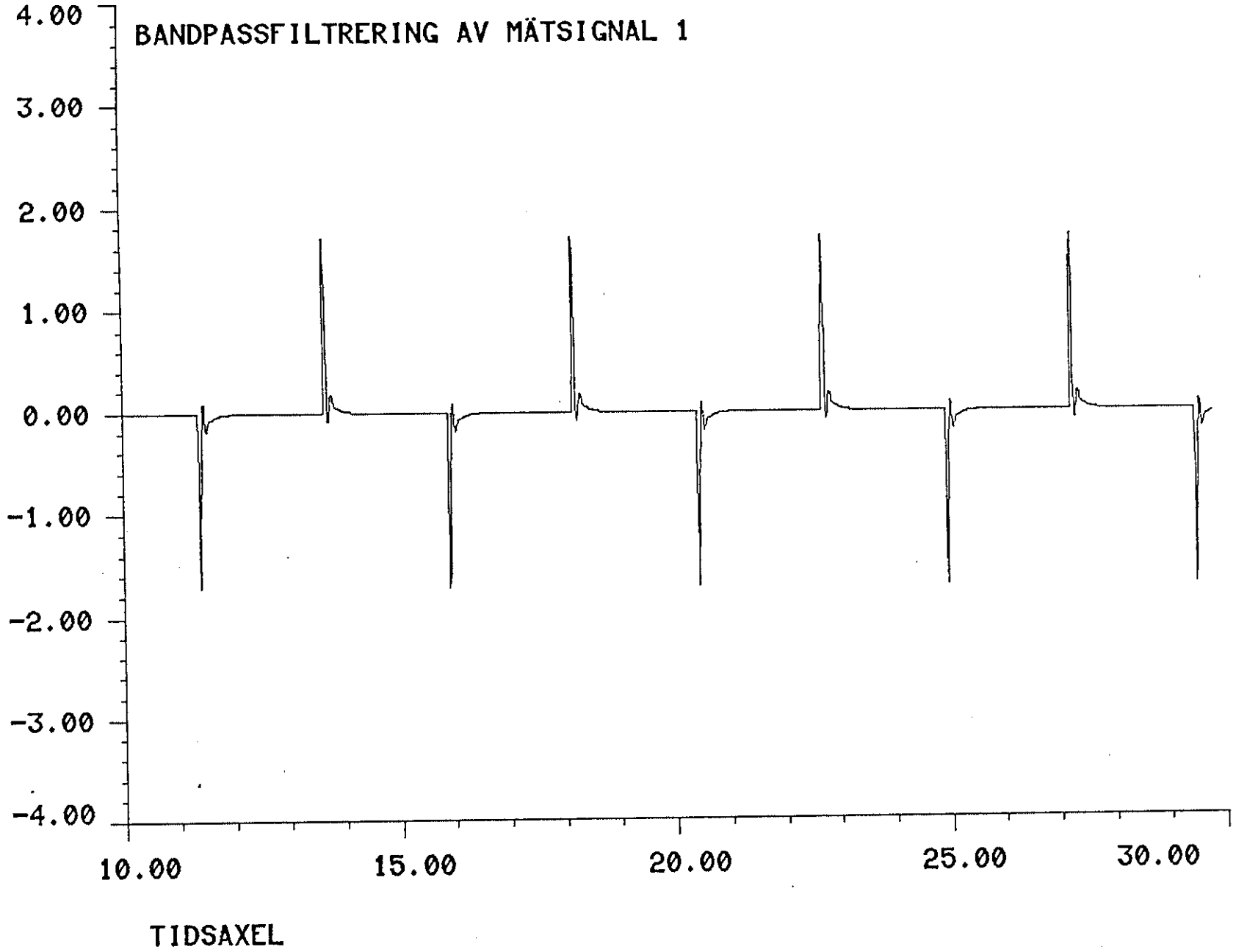
```
***       PROGRAM  FÖR FLICKERMETER    ***
      PROGRAM FLICK
      REAL K,W1,W2,W3,W4,L,PI
      PARAMETER (PI=3.14159,
     1           K=1.748Ø2,
     1           W1=2*PI*9.15494,
     1           W2=2*PI*2.27979,
     1           W3=2*PI*1.22535,
     1           W4=2*PI*21.9,
     1           L=2*PI*4.Ø5981)
      REAL A,B,C,D,N,U
      PARAMETER (U=(K*W1*W4/W2)-K*W1,
     1           N=(2*L*W4/W3)-(W4*W4/W3)-(W1*W1/W3)-2*L+W4+(W1*W1/W4),
     1           D=U/N,
     1           C=(K*W1-(2*L-W4-W1*W1/W4)*D)/(2*L-W3-W1*W1/W3),
     1           B=W1*W1*(-C-D),
     1           A=-(W3*C+W4*D))
      INTEGER X,I,R,NR1,G,NR2,NR3,IT(1:1Ø)
      PARAMETER (X=6ØØØØ,NR1=1,NR2=2,NR3=3)
      REAL W(1:X),Y(1:X),Y1(1:X),Y2(1:X),Y3(1:X)
      REAL T,ALF,TID(1:X),B1,B2,B3,B4
      REAL COW1(1:2),COW2(1:2),COW3(1:2),COY1(1:2),COY2(1:2),COY3(1:2)
      CHARACTER*8 NAMN1,NAMN2,NAMN3
      CHARACTER*6Ø NAMN4
***           HUVUDPROGRAM      ***
      T=Ø.Ø1
      ALF=SQRT(W1*W1-L*L)
      B1=EXP(-L*T)*SIN(ALF*T)/ALF
      B2=-B1
      B3=1-EXP(-L*T)*(COS(ALF*T)+L*SIN(ALF*T)/ALF)
      B4=EXP(-2*L*T)+EXP(-L*T)*(L*SIN(ALF*T)/ALF-COS(ALF*T))
      COW1(1)=A*B1+B*B3/(W1*W1)
      COW1(2)=A*B2+B*B4/(W1*W1)
      COW2(1)=C*(1-EXP(-W3*T))
      COW2(2)=Ø.Ø
      COW3(1)=D*(1-EXP(-W4*T))
      COW3(2)=Ø.Ø
      COY1(1)=2*COS(ALF*T)*EXP(-L*T)
      COY1(2)=-EXP(-2*L*T)
      COY2(1)=EXP(-W3*T)
      COY2(2)=Ø.Ø
      COY3(1)=EXP(-W4*T)
      COY3(2)=Ø.Ø
      OPEN(UNIT=NR1,FILE='INSIG.BIN',FORM='UNFORMATTED')
      DO 5 I=1,X
       READ(NR1) W(I)
    5 CONTINUE
      CLOSE(UNIT=NR1)
      CALL Y1FUN(COW1,COW2,COW3,Y1,Y2,Y3,W,X,Y)
      CALL Y2FUN(COW1,COW2,COW3,COY1,COY2,COY3,Y1,Y2,Y3,W,X,Y)
      TID(1)=T
      TID(2)=2.Ø*T
      DO 1Ø I=3,X
        CALL YFUNC(I,Y1,COW1,COY1,W)
        CALL YFUNC(I,Y2,COW2,COY2,W)
        CALL YFUNC(I,Y3,COW3,COY3,W)
        Y(I)=Y1(I)+Y2(I)+Y3(I)
        TID(I)=REAL(I)*T
      WRITE(*,*) Y(I),TID(I)
   1Ø CONTINUE
      OPEN(UNIT=NR2,FILE='YSIG.BIN',FORM='UNFORMATTED')
      IT(1)=X
```

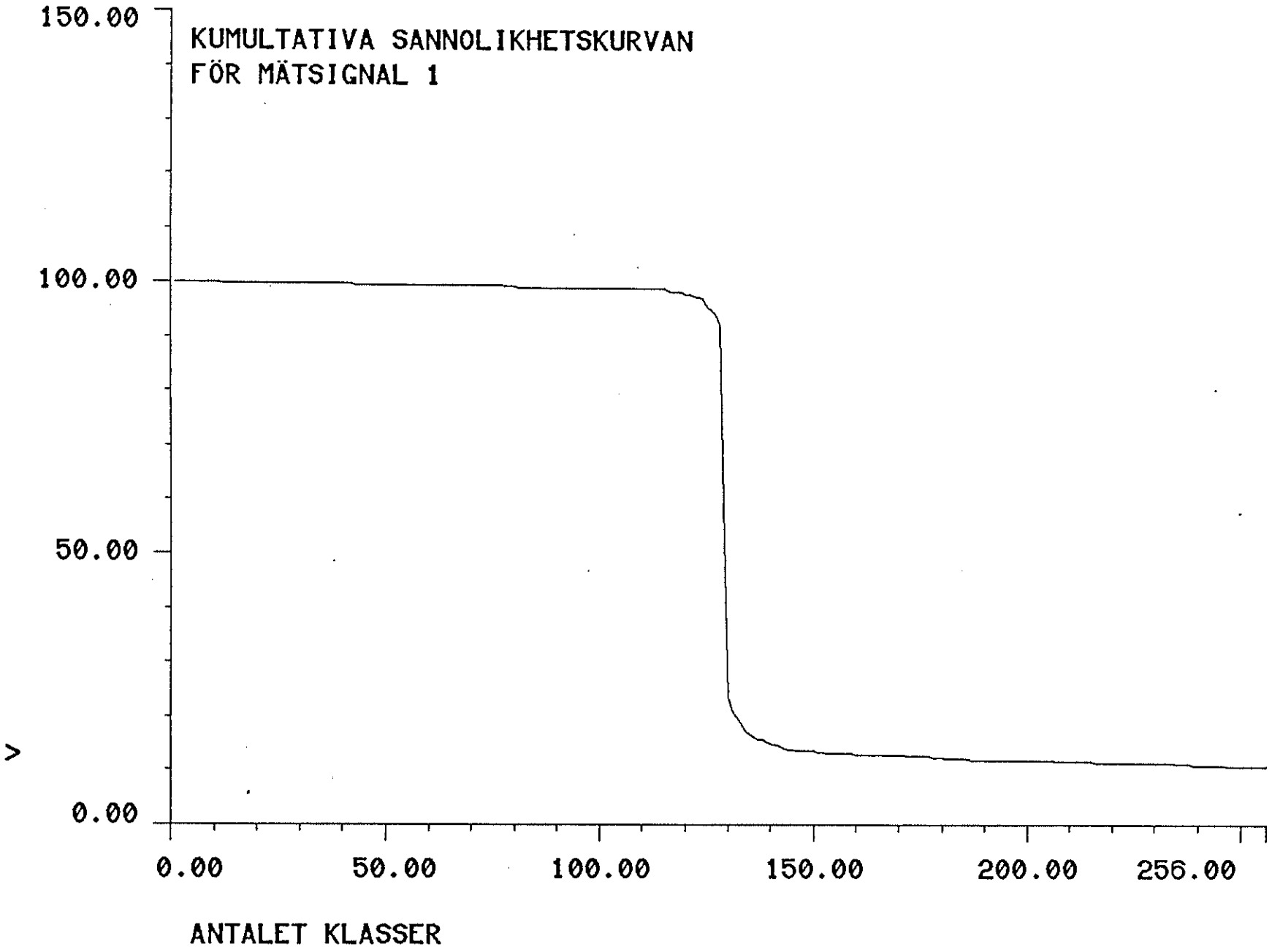
```
      IT(2)=2
      IT(10)=3
      DO 20 I=3,9
       IT(I)=0
   20 CONTINUE
      NAMN1='FILN3'
      NAMN2='BPSIG'
      NAMN3='TID'
      NAMN4='BANDPASSFILTRRING'
      WRITE(NR2) (IT(I),I=1,10)
      WRITE(NR2) NAMN1,NR2
      WRITE(NR2) NAMN2,NAMN2,NAMN4
      WRITE(NR2) NAMN3,NAMN3,NAMN4
      DO 30 I=1,X
      WRITE(NR2) Y(I),TID(I)
   30 CONTINUE
      CLOSE(UNIT=NR2)
      STOP
      END
***       UNERPROGRAM FÖR BERÄKNING AV Y-PARAMETRARNA  ***
      SUBROUTINE YFUNC(I,YX,COW,COY,W)
      INTEGER I,Z
      REAL COW(1:2),COY(1:2),YX(1:I),W(1:I)
      REAL YW,YY
      YW=0.0
      YY=0.0
      DO 10 Z=0,1
        YW=YW+COW(Z+1)*W(I-Z)
   10 CONTINUE
      DO 20 Z=1,2
        YY=YY+COY(Z)*YX(I-Z)
   20 CONTINUE
      YX(I)=YW+YY
      RETURN
      END
***   UNDERPROGRAM FÖR BERÄKNING AV Y(1)    ***
      SUBROUTINE Y1FUN(COW1,COW2,COW3,Y1,Y2,Y3,W,X,Y)
      INTEGER X
      REAL COW1(1:2),COW2(1:2),COW3(1:2),Y1(1:X),Y2(1:X),Y3(1:X),W(1:X)
      REAL Y(1:X)
      Y1(1)=COW1(1)*W(1)
      Y2(1)=COW2(1)*W(1)
      Y3(1)=COW3(1)*W(1)
      Y(1)=Y1(1)+Y2(1)+Y3(1)
      WRITE(*,*) Y(1),Y1(1),Y2(1),Y3(1)
      RETURN
      END
***       UNDERPROGRAM FÖR BERÄKNING AV Y(2)    ***
      SUBROUTINE Y2FUN(COW1,COW2,COW3,COY1,COY2,COY3,Y1,Y2,Y3,W,X,Y)
      INTEGER X
      REAL COW1(1:2),COW2(1:2),COW3(1:2),COY1(1:2),COY2(1:2),COY3(1:2)
      REAL Y1(1:X),Y2(1:X),Y3(1:X),W(1:X),Y(1:X)
      Y1(2)=COW1(1)*W(2)+COW1(2)*W(1)+COY1(1)*Y1(1)
      Y2(2)=COW2(1)*W(2)+COY2(1)*Y2(1)
      Y3(2)=COW3(1)*W(2)+COY3(1)*Y3(1)
      Y(2)=Y1(2)+Y2(2)+Y3(2)
      RETURN
      END
```

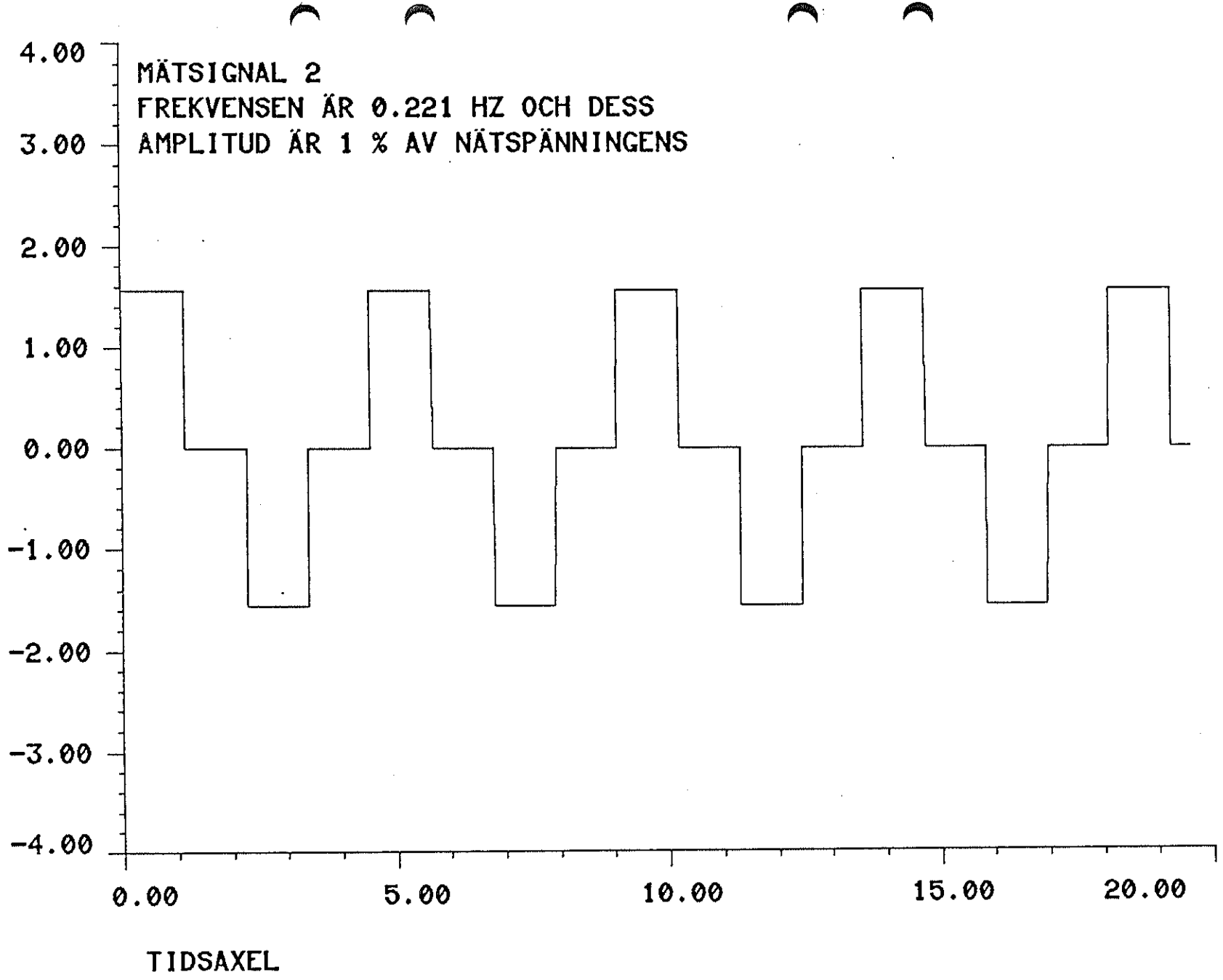
```
***       PROGRAM FÖR STATISTISK BEHANDLING AV FLICKERSIGNALEN     ***
      PROGRAM STAT
      CHARACTER*8 NAMN1,NAMN2,NAMN3
      CHARACTER*60 NAMN4
      INTEGER I,X,CLASL,Z,NR2,NR4
      PARAMETER (X=60000,CLASL=250,NR2=2,NR4=4)
      INTEGER IT(1:10)
      REAL CLASS(1:CLASL),SUMSA
      REAL CLASP(1:CLASL),CLASV(1:CLASL),Y(1:X),PST,PX(1:5)
      REAL P(1:5)
      REAL K1,K2,K3,K4,K5,CLASB
      PARAMETER (K1=0.0314,
     1           K2=0.0525,
     1           K3=0.0657,
     1           K4=0.28,
     1           K5=0.08)
***         HUVUDPROGRAM      ***
      AMP=0
      OPEN(UNIT=NR2,FILE='YSIG.BIN',FORM='UNFORMATTED')
      DO 5 I=1,X
      READ(NR2) Y(I)
    5 CONTINUE
      CLOSE(NR2)
      WRITE(*,*) 'AMP'
      READ(*,*) AMP
      WRITE(*,*) (Y(I),I=300,350)
      CALL VECT(AMP,P,CLASS,CLASV,CLASL,CLASB)
      WRITE(*,*) (CLASV(I),I=1,CLASL)
      SUMSA=0.0
      CALL SORT(AMP,CLASS,SUMSA,CLASV,CLASL,Y,X,CLASB)
      WRITE(*,*) (CLASS(I),I=1,CLASL),SUMSA
      CALL PERC(CLASS,SUMSA,CLASL,CLASP)
      WRITE(*,*) (CLASP(I),I=1,CLASL)
      CALL PIDE(CLASP,CLASV,P,CLASL,PX,CLASB)
      PST=SQRT(K1*PX(1)+K2*PX(2)+K3*PX(3)+K4*PX(4)+K5*PX(5))
      WRITE(*,*) PST
      CLASP(1)=100-CLASP(1)
      DO 20 I=2,CLASL
       CLASP(I)=CLASP(I-1)-CLASP(I)
   20 CONTINUE
      OPEN(UNIT=NR4,FILE='STAT.BIN',FORM='UNFORMATTED')
      NAMN1='FILN4'
      NAMN2='PROC'
      NAMN3='KLASS'
      IT(1)=CLASL
      IT(2)=1
      IT(10)=2
      WRITE(NR4) (IT(I),I=1,10)
      WRITE(NR4) NAMN1,NR4
      WRITE(NR4) NAMN2,NAMN2,NAMN4
      DO 30 I=1,CLASL
       WRITE(NR4) CLASP(I)
   30 CONTINUE
      CLOSE(UNIT=NR4)
      STOP
      END
***       UNDERPROGRAM SOM LAGRAR BEGYNNELSEVÄRDEN I VEKTORER     ***
      SUBROUTINE VECT(AMP,P,CLASS,CLASV,CLASL,CLASB)
      INTEGER CLASL,I
      REAL CLASV(1:CLASL),P(1:5),CLASS(1:CLASL),CLASB,AMP
      CLASB=2*AMP/CLASL
      CLASS(1)=0.0
```

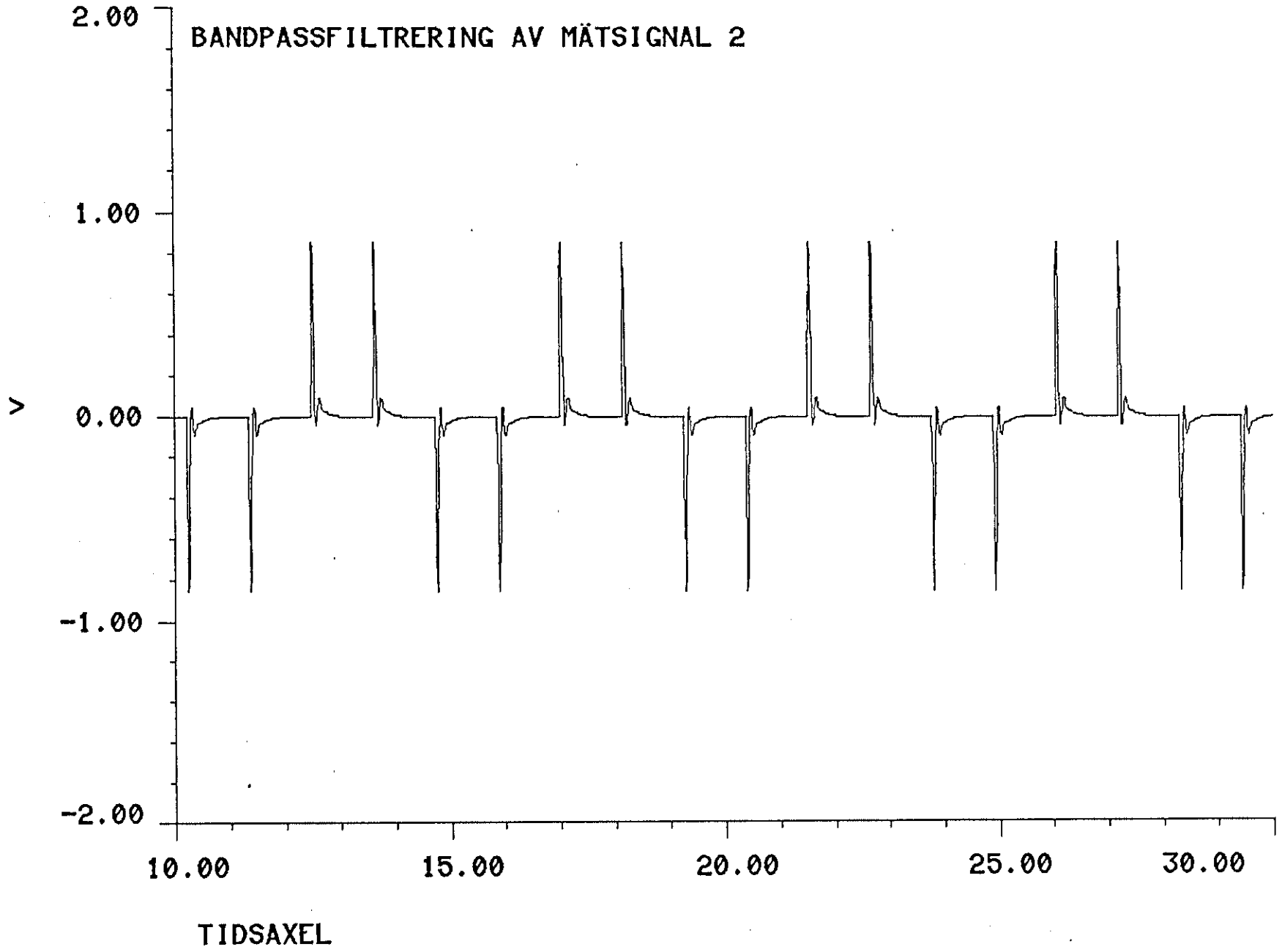
```
      CLASV(1)=CLASB
      DO 1Ø I=2,CLASL
        CLASS(I)=Ø.Ø
        CLASV(I)=CLASV(I-1)+CLASB
   1Ø CONTINUE
      P(1)=Ø.1
      P(2)=1.Ø
      P(3)=3.Ø
      P(4)=1Ø.Ø
      P(5)=5Ø.Ø
      RETURN
      END
***     UNDERPROGRAM FÖR SORTERING AV SAMPELVÄRDENA    ***
      SUBROUTINE SORT(AMP,CLASS,SUMSA,CLASV,CLASL,Y,X,CLASB)
      INTEGER Z,X,I,CLASL
      REAL CLASV(1:CLASL),Y(1:X),CLASS(1:CLASL),SUMSA,AMP,CLASB
      DO 1Ø Z=1,X
       IF (ABS(Y(Z)).LT.AMP) THEN
        I=INT((Y(Z)+AMP)/CLASB)+1
        CLASS(I)=CLASS(I)+1.Ø
        SUMSA=SUMSA+1.Ø
       ENDIF
   1Ø CONTINUE
      RETURN
      END
*** UNDERPROGRAM FÖR BERÄKNING AV KLASSERNAS DEL AV TOTALA UTFALLET ***
      SUBROUTINE PERC(CLASS,SUMSA,CLASL,CLASP)
      INTEGER CLASL,I
      REAL CLASP(1:CLASL),CLASS(1:CLASL),SUMSA
      DO 1Ø I=1,CLASL
        CLASP(I)=1ØØ.Ø*CLASS(I)/SUMSA
   1Ø CONTINUE
      RETURN
      END
***     UNDERPROGRAM FÖR BERÄKNING AV PX    ***
      SUBROUTINE PIDE(CLASP,CLASV,P,CLASL,PX,CLASB)
      INTEGER I,Z,CLASL,CL
      REAL H1,H2,H3
      REAL CLASV(1:CLASL),PRO,P(1:5),PX(1:5),CLASP(1:CLASL),CLASB,PK
      B=CLASB
      I=CLASL
      PRO=Ø
      DO 2Ø Z=1,5
   1Ø   IF (PRO.LT.P(Z)) THEN
          I=I-1
          PK=PRO
          PRO=PRO+CLASP(I)
        GOTO 1Ø
        ENDIF
        CL=I
        PX(Z)=(CLASV(CL)+CLASV(CL-1))/2
        WRITE(*,*) 'LIN'
        WRITE(*,*) PX(Z)
   2Ø CONTINUE
      RETURN
      END
```

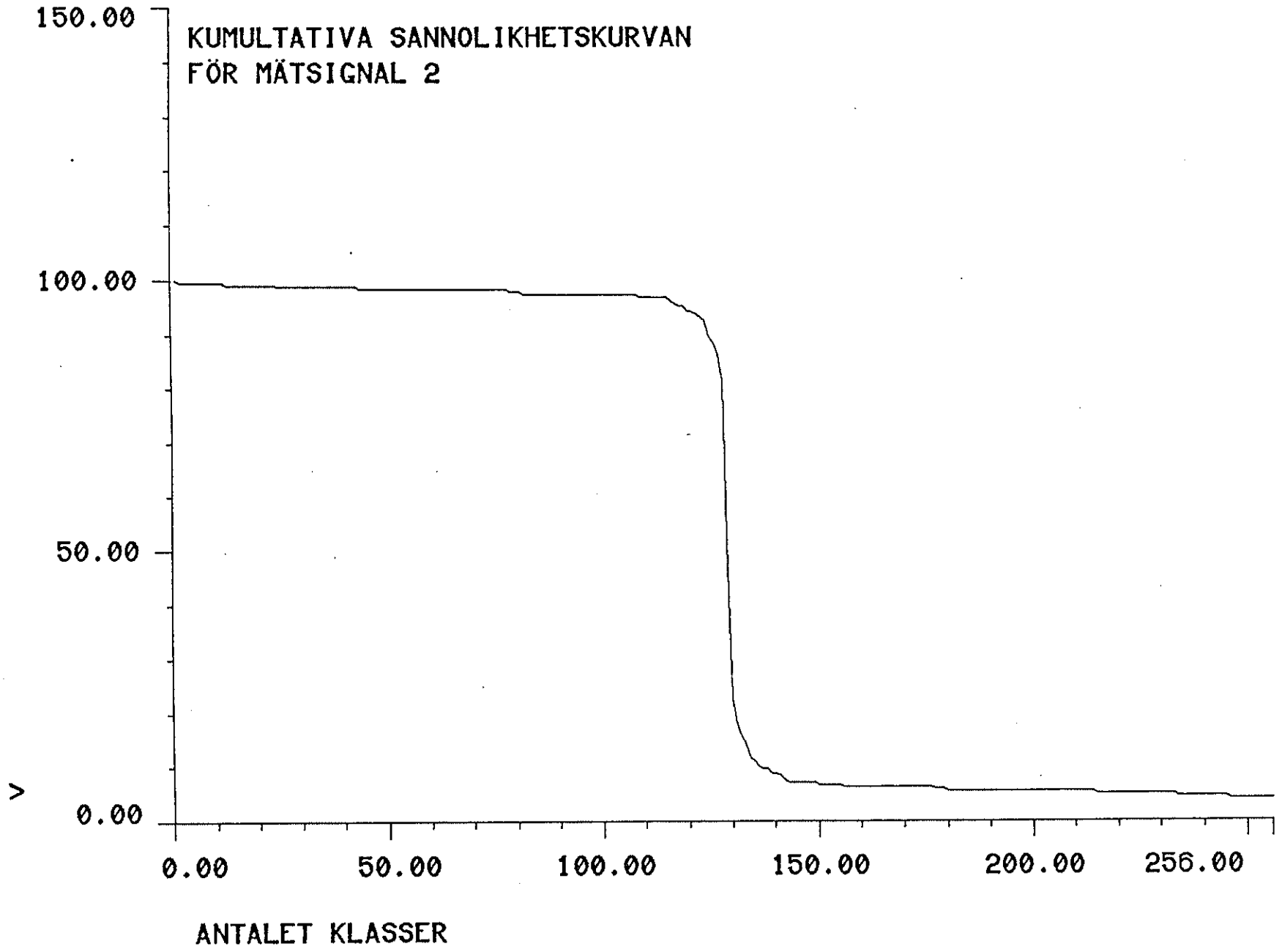

MÄTSIGNAL 1
FYRKANTVÅG MED FREKVENSEN 0.221 HZ
AMPLITUDEN ÄR 1 % AV NÄTSPÄNNINGENS
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
0.00
5.00
10.00
15.00
20.00
TIDSAXEL

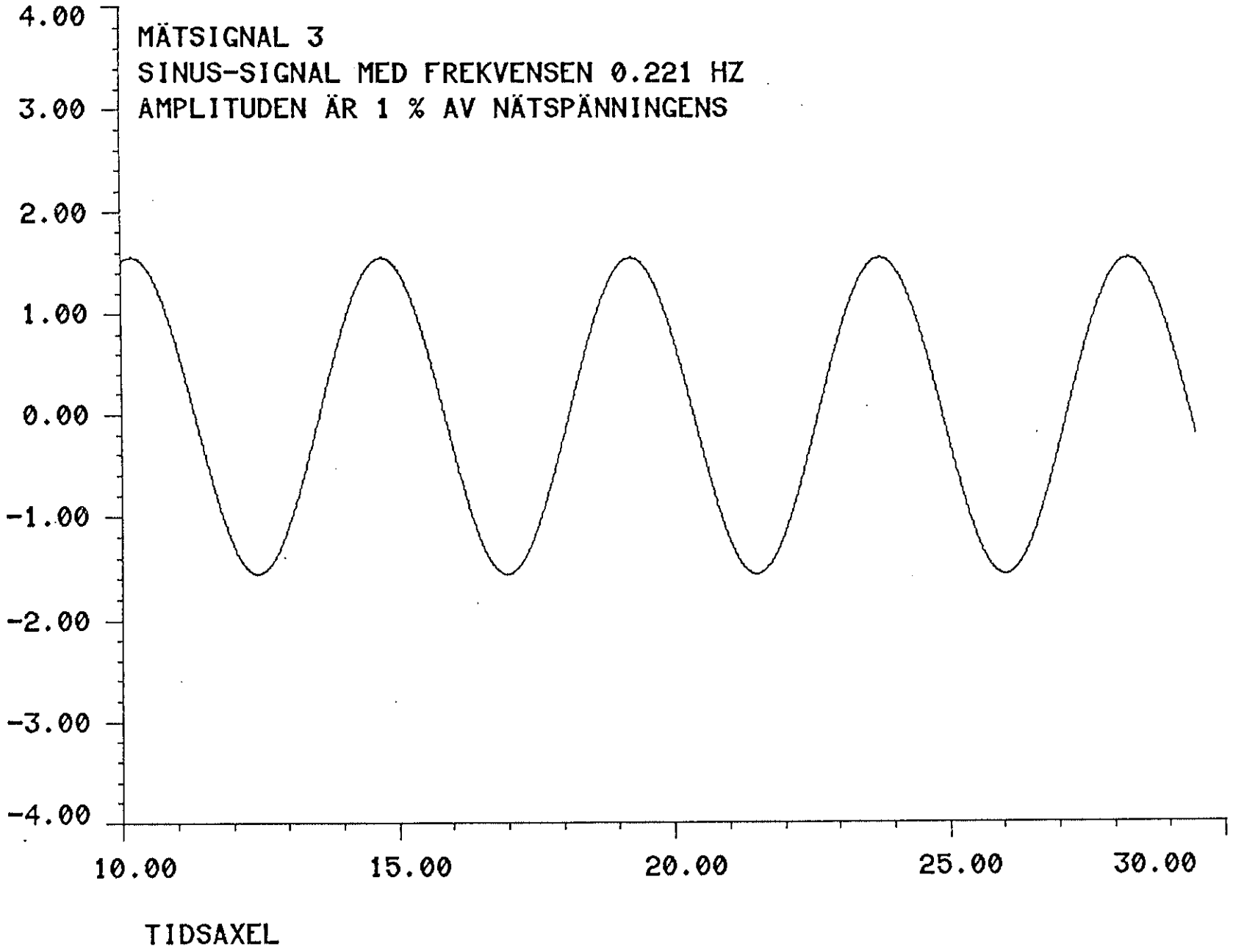

BANDPASSFILTRERING AV MÄTSIGNAL 1
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
10.00
15.00
20.00
25.00
30.00
TIDSAXEL

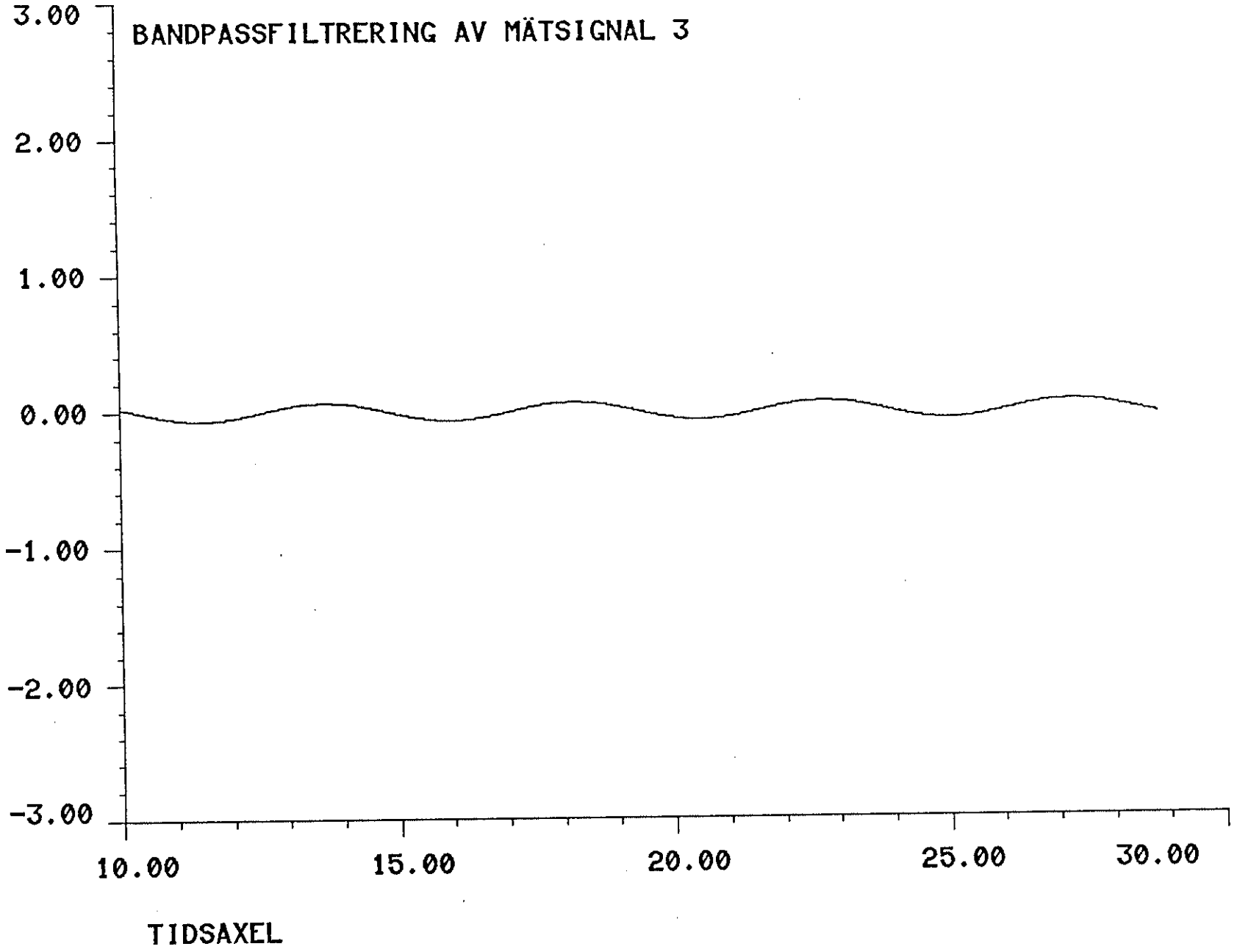

KUMULTATIVA SANNOLIKHETSKURVAN
FÖR MÄTSIGNAL 1
150.00
100.00
50.00
0.00
>
0.00
50.00
100.00
150.00
200.00
256.00
ANTALET KLASSER

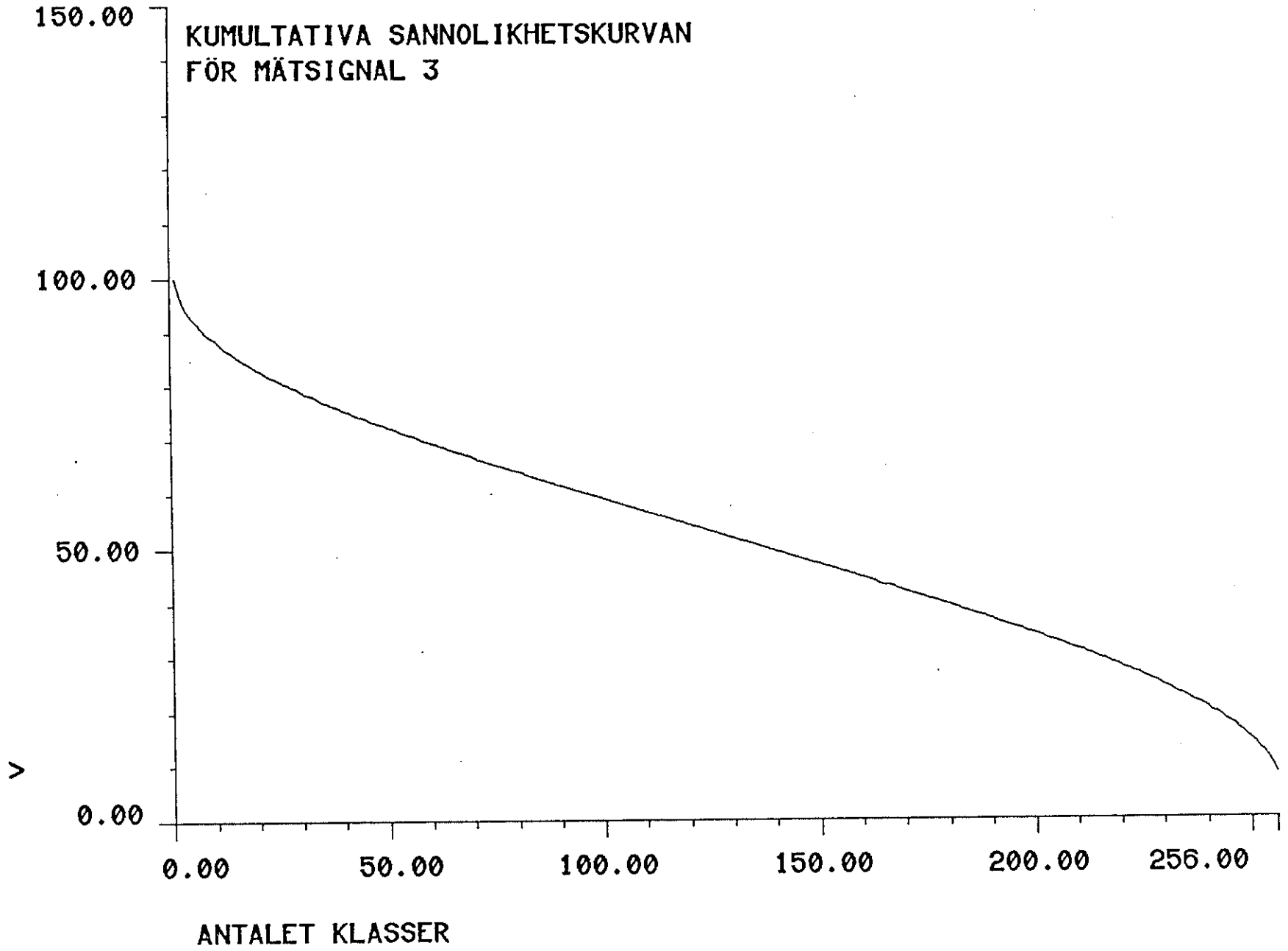

MÄTSIGNAL 2
FREKVENSEN ÄR 0.221 HZ OCH DESS
AMPLITUD ÄR 1 % AV NÄTSPÄNNINGENS
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
0.00
5.00
10.00
15.00
20.00
TIDSAXEL


BANDPASSFILTRERING AV MÄTSIGNAL 2
2.00
1.00
0.00
-1.00
-2.00
>
10.00
15.00
20.00
25.00
30.00
TIDSAXEL


KUMULTATIVA SANNOLIKHETSKURVAN
FÖR MÄTSIGNAL 2
150.00
100.00
50.00
0.00
0.00
50.00
100.00
150.00
200.00
256.00
ANTALET KLASSER


MÄTSIGNAL 3
SINUS-SIGNAL MED FREKVENSEN 0.221 HZ
AMPLITUDEN ÄR 1 % AV NÄTSPÄNNINGENS
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
-4.00
10.00
15.00
20.00
25.00
30.00
TIDSAXEL


BANDPASSFILTRERING AV MÄTSIGNAL 3
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
10.00
15.00
20.00
25.00
30.00
TIDSAXEL


KUMULTATIVA SANNOLIKHETSKURVAN
FÖR MÄTSIGNAL 3
150.00
100.00
50.00
0.00
>
0.00
50.00
100.00
150.00
200.00
256.00
ANTALET KLASSER

Referenser

1. Disturbance Working Group, UIE, "Flicker Measurement and Evaluation", Paris 1986

2. G Salomonsson, P-O Börjesson, O Pahlm och G Eriksson, "Tidsdiskreta kretsar och signaler", Lund 1985

3. P Eriksson och G Salomonsson, "Kompendium i digital signalbehandling", Lund 1985

4. K-J Åström och B Wittenmark, "Computer Controlled Systems", Prentice-Hall, Inc., Englewood Cliffs 1984

5. Tore Mårtensson, "Tispac", Umeå 1984

6. Johan Wieslander, "Idpac", Lund 1980

7. Teletransmissionsteori, Formler och tabeller, Lund 1983