Module 20 Filtering van signalen

Onderwerp	Butterworth-filters, optimale filters, discretisatie, Matlab
Voorkennis	Fourier- en Laplace-transformatie, overdrachtsfunctie
Expressies	`laplace, fourier, invlaplace, invfourier, setvar, getvar, evalM`
Bibliotheken	`inttrans, Matlab`
Zie ook	Module 19, 21.

20.1  Filters

Filters zijn systemen die worden gebruikt om na te gaan of een signaal aan een bepaalde – gewenste of ongewenste – eigenschap voldoet. Men construeert het filter zodanig dat – ongeacht de aard of complexiteit van het ingangssignaal – aan het uitgangssignaal eenvoudig – en dus automatiseerbaar – de aanwezigheid van de betreffende eigenschap is vast te stellen. Wanneer het filter deel uitmaakt van een groter systeem, kan – weer automatisch – worden besloten tot een adequate reactie van dit systeem.
Het ontwerp van een filter hangt zeer af van de toepassing, en is meestal niet triviaal. In de loop der tijd zijn hiervoor tal van strategieën ontwikkeld. Wij zullen twee van deze strategieën eruit lichten. De eerste, het filterontwerp volgens Butterworth, omdat hieraan een heel elegante afleiding ten grondslag ligt. Deze filters kunnen worden gerealiseerd door analoge elektronica. De tweede strategie richt zich op de constructie van filters die in zekere zin geoptimaliseerd zijn voor het doorlaten van gewenste signalen en blokkeren van ruis. Deze strategie is vooral relevant in praktische toepassingen, daar alle in de vrije natuur voorkomende signalen vervormd zijn door enige vorm van ruis.
Wij beschouwen hier alleen signalen die kunnen worden voorgesteld als reële of complexe functies van één variabele, de tijd. Bovendien kijken we alleen naar lineaire, tijdsinvariante systemen. Signalen beschreven in het tijddomein zullen we aangeven met kleine letters, en signalen beschreven in het frequentiedomein met hoofdletters. Het verband tussen ingangssignaal en uitgangssignaal kan voor lineaire, tijdsinvariante systemen eenvoudig worden beschreven met behulp

van Laplace- of Fourier-transformaties. Hierbij geldt:

$$Y_L(s) = H_L(s)F_L(s), \quad \text{respectievelijk} \quad Y(\omega) = H(\omega)F(\omega).$$

Hierbij stellen $Y_L(s)$, $H_L(s)$, en $F_L(s)$ de *Laplace-getransformeerden* voor van respectievelijk $y(t)$, $h(t)$ en $f(t)$:

$$Y_L(s) = \int_0^\infty e^{-st} y(t)\, dt,$$

voor die waarden van $s \in \mathbb{C}$ waarvoor de integraal bestaat. De functie $H_L(s)$ heet de *overdrachtsfunctie* van het systeem, en $h(t)$ is de *impulsrespons*. De functies $Y(\omega)$, $H(\omega)$ en $F(\omega)$ noteren de Fourier-getransformeerden van $y(t)$, $h(t)$ en $f(t)$:

$$Y(\omega) = \int_{-\infty}^\infty y(t) e^{j\omega t}\, dt.$$

Systemen met een rationale overdrachtsfunctie $H_L(s) = P_L(s)/Q_L(s)$, waarbij de graad van P lager is dan die van Q kunnen – in het tijddomein – ook worden beschreven met een lineaire differentiaalvergelijking:

$$y^{(n)}(t) + q_{n-1}y^{(n-1)}(t) + \cdots + q_0 y(t) = p_n u^{(n)}(t) + \cdots + p_0 u(t),$$

waarbij $y(0) = y^{(1)}(0) = \cdots = y^{(n-1)}(0) = 0$ als het systeem in rust is – dus geen uitvoersignaal levert – op het moment dat het ingangssignaal 'begint'.
Wanneer voor s de variabele $j\omega$ wordt ingevuld in $|H_L(s)|$ vinden we de *frequentiekarakteristiek* van het systeem, gegeven door $|H_L(j\omega)|$. Losjes zou je kunnen zeggen dat $|H_L(j\omega)|$ aangeeft in hoeverre een signaal met frequentie ω door het systeem wordt 'doorgegeven'. Wanneer $|H_L(j\omega)| = 1$ voor $|\omega| \leq \omega_b$ en 0 elders spreken we van een ideaal laagdoorlaatfilter. Merk op dat de Laplace-transformatie algemener toepasbaar is dan de Fourier-transformatie. Bijvoorbeeld, de Laplace-getransformeerde van een signaal $\sin(t)$ bestaat wel, maar de Fourier-getransformeerde niet. Daarom beschouwen we hier $H_L(j\omega)$ als frequentiekarakteristiek.

20.2  Laplace- en Fourier-transformatie

`laplace`

De Laplace- en Fourier-getransformeerde van een functie $f(t)$ kunnen we berekenen middels de opdracht `laplace` respectievelijk `fourier`

uit de bibliotheek `inttrans`. Ondanks het feit dat deze procedures behoorlijk krachtig zijn, vormen ze geen 'silver bullet': eigen inventiviteit blijft onmisbaar zodra de te transformeren functies wat moeilijker zijn.

Voorbeeldopgave

Bepaal de Laplace- en Fourier-getransformeerden van de functie f, met $f(t) = e^{-3t}$ voor $t > 0$ en $f(t) = 0$ voor $t \leq 0$.

Voorbeeldsessie

```
> restart; with(inttrans);
```

[*addtable*, *fourier*, *fouriercos*, *fouriersin*, *hankel*, *hilbert*, *invfourier*, *invhilbert*, *invlaplace*, *invmellin*, *laplace*, *mellin*, *savetable*]

```
> f:=x->exp(-3*x)*Heaviside(x);
```

$$f := x \rightarrow e^{(-3\,x)}\,\mathrm{Heaviside}(x)$$

```
> laplace(f(x),x,s):  F := unapply(%,s);
```

$$F := s \rightarrow \frac{1}{s+3}$$

```
> F(I*omega);
```

$$\frac{1}{\omega\,I+3}$$

```
> fourier(f(x),x,omega);
```

$$\frac{1}{\omega\,I+3}$$

Toelichting

In deze opdracht is de tweede parameter de integratievariabele, en de derde is de – vrije – parameter van de getransformeerde functie. Middels `unapply` kan er een Maple-procedure van worden gemaakt. Merk op dat in dit geval de Laplace-getransformeerde $F(s)$ voor $s = j\omega$ gelijk is aan de Fourier-getransformeerde van f. Dit is in het algemeen niet zo! (Zie opgave 20.1.) ◇

20.3  Het Butterworth-filter

Het *Butterworth-filter* van orde n is het systeem met overdrachtsfunctie

$$H_L(s) = \prod_{k=1}^{n} \frac{1}{s - s_k},$$

waarbij $\{s_k\}$ de set oplossingen is van de vergelijking

$$1 + (-s)^{2n} = 0$$

met negatief reëel deel.
Het Butterworth-filter is een benadering van een ideaal laagdoorlaatfilter.

Voorbeeldopgave

Gegeven signalen $f_1(t) = \sin(t)$ en $f_2(t) = \sin(10t)$, beide voor $0 \leq t < 10$, en een tweede orde Butterworth-filter. Teken de frequentiekarakteristiek van het filter, alsmede het spectrum van f_1 en f_2. Hoe zien – als functie van de tijd – de uitgangssignalen $y_1(t)$ en $y_2(t)$ eruit?

Voorbeeldsessie

```
> restart; with(inttrans):
> rect := t -> Heaviside(t)-Heaviside(t-1):
> f1 := t -> sin(t)*rect(t/10):
  f2 := t-> sin(10*t)*rect(t/10):
> F_L1 := unapply( laplace(f1(t),t,s), s );
```

$$F\_L1 := s \rightarrow \frac{1 + e^{(-10\,s)}\,(-\cos(10) - \sin(10)\,s)}{s^2 + 1}$$

```
> F_L2 := unapply( laplace(f2(t),t,s), s ):
> sol := solve(1+(-s)^4=0);
```

$$sol := \frac{\sqrt{2}}{2} + \frac{1}{2}\,I\,\sqrt{2},\ -\frac{\sqrt{2}}{2} + \frac{1}{2}\,I\,\sqrt{2},\ -\frac{\sqrt{2}}{2} - \frac{1}{2}\,I\,\sqrt{2},\ \frac{\sqrt{2}}{2} - \frac{1}{2}\,I\,\sqrt{2}$$

```
> S := select( z->evalf(Re(z))<0, [sol] );
```

$$S := [-\frac{\sqrt{2}}{2} + \frac{1}{2}\,I\,\sqrt{2},\ -\frac{\sqrt{2}}{2} - \frac{1}{2}\,I\,\sqrt{2}]$$

```
> product( 1/(s-S[i]), i=1..2 ):
  H_L:=unapply( %, s );
```

$$H\_L := s \rightarrow \frac{1}{(s + \frac{\sqrt{2}}{2} - \frac{1}{2}\,I\,\sqrt{2})\,(s + \frac{\sqrt{2}}{2} + \frac{1}{2}\,I\,\sqrt{2})}$$

```
> Y_L1 := unapply( F_L1(s)*H_L(s), s );
```

$$Y\_L1 := s \rightarrow \frac{1 + e^{(-10\,s)}\,(-\cos(10) - \sin(10)\,s)}{(s^2 + 1)\,(s + \frac{\sqrt{2}}{2} + \frac{1}{2}\,I\,\sqrt{2})\,(s + \frac{\sqrt{2}}{2} - \frac{1}{2}\,I\,\sqrt{2})}$$

```
> Y_L2 := unapply( F_L2(s)*H_L(s), s ):
> plot( abs(H_L(I*omega)), omega=-15..15 );
```

(zie figuur 33, links)

```
> p1:=plot( abs(F_L1(I*omega)), omega=-15..15,
        linestyle=DASH, color=black ):
  p2:=plot( abs(F_L2(I*omega)), omega=-15..15,
        thickness=2, color=black ):
  plots:-display({p1,p2});
```

(zie figuur 33, rechts)

```
> y1 := simplify( invlaplace(Y_L1(s),s,t) ):
  y2 := simplify( invlaplace(Y_L2(s),s,t) ):
```

```
> py1 := plot( y1, t=0..20, linestyle=DASH, color=black ):
  py2 := plot( y2, t=0..20, thickness=3, color=black ):
  plots:-display({py1,py2});
```

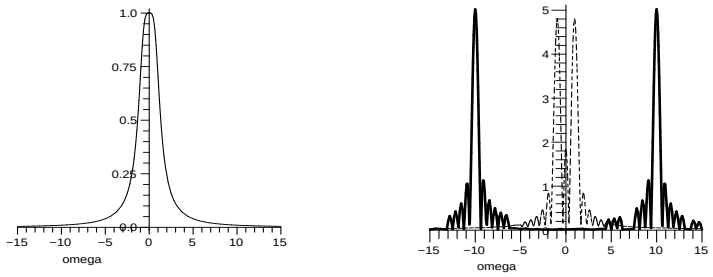
(zie figuur 34, links)

```
> ht := invlaplace(H_L(s),s,t);
```

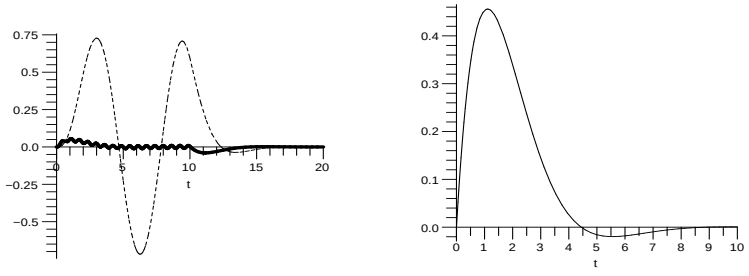
$$ht := e^{(-\frac{\sqrt{2}\,t}{2})}\,\sqrt{2}\sin(\frac{\sqrt{2}\,t}{2})$$

```
> plot(ht,t=0..10);
```

(zie figuur 34, rechts)



FIGUUR 33. Frequentiekarakteristiek van het filter (links) en spectra van f_1 en f_2 (rechts)



FIGUUR 34. Uitgangssignalen y_1 en y_2 (links) en impulsrespons (rechts)

Toelichting

Op grond van de frequentiekarakteristiek van het filter verwachten we dat f_1 wel, en f_2 niet wordt doorgelaten. De spectra van f_1 en f_2 ondersteunen deze gedachte. Merk overigens op dat de spectra geen lijnspectra zijn. Dit komt doordat de signalen een eindige tijd duren (hetgeen is aangegeven door de rechthoekfunctie). De spectra vertonen het kenmerkende spectrum van die rechthoekfunctie geconcentreerd rond $\omega = 1$ respectievelijk $\omega = 10$. Zie ook Module 19. Het tijdsgedrag van de uitgangssignalen laat zien dat het signaal f_2 inderdaad veel minder goed wordt doorgegeven dan het signaal f_1. De duur van het uitgangssignaal is langer dan de duur van het ingangssignaal. Dit komt omdat het systeem nog uitslingert met functie $h(t)$, de impulsrespons. Deze is in figuur 34 zichtbaar gemaakt. ◇

20.4 Het optimale filter

Stel dat $f(t)$ een signaal is met eindige energie-inhoud E,

$$E = \int_{-\infty}^{\infty} |f(t)|^2 \, dt < \infty \,,$$

met Fourier-getransformeerde

$$F(\omega) = \int_{-\infty}^{\infty} f(t)\, e^{-j\omega t} \, dt \,.$$

We gaan proberen een filter te ontwerpen zodanig dat dit optimaal onderscheid kan maken tussen enerzijds een ingangssignaal bestaande uit een verwacht of gewenst signaal f en anderzijds een ingangssignaal dat alleen uit *witte ruis* bestaat. Ruwweg kan witte ruis gekarakteriseerd worden als een stochastisch signaal $n(t)$, waarbij, gemiddeld over een heleboel realisaties van $n(t)$, in het frequentiedomein elke frequentie 'even vaak voorkomt'.
De impulsrespons van het systeem noteren we met $h(t)$, en Fourier-getransformeerde hiervan als $H(\omega)$. Het instantane vermogen van het uitgangssignaal $y(t)$ is te schrijven als

$$|y(t)|^2 = \left| \int_{-\infty}^{\infty} F(\omega) H(\omega) e^{j\omega t} \, d\omega \right|^2 . \qquad (20.1)$$

De witte ruis heeft een uitgangssignaal tot gevolg dat geschreven kan worden als

$$N = \frac{N_0}{2} \int_{-\infty}^{\infty} |H(\omega)|^2 \, d\omega \,, \tag{20.2}$$

waarin N_0 de spectrale vermogensdichtheid weergeeft. We zoeken een overdrachtsfunctie H waarbij de verhouding

$$R = \frac{\max_t |y(t)|^2}{N} \tag{20.3}$$

maximaal wordt. Het (onbekende) tijdstip t waarop dit gebeurt noteren we met $T_{\max}$. Het getal R noemen we hier de signaal-ruis verhouding.[48]
Met behulp van de ongelijkheid van Cauchy-Schwartz vinden we, wanneer we (20.1) en (20.2) invullen in (20.3), na wat rekenwerk,

$$R \leq \frac{\int_{-\infty}^{\infty} |H(\omega)|^2 \, d\omega \int_{-\infty}^{\infty} |F(\omega)|^2 \, d\omega}{N_0/2 \int_{-\infty}^{\infty} |H(\omega)|^2 \, d\omega} = \frac{\int_{-\infty}^{\infty} |F(\omega)|^2 \, d\omega}{N_0/2}, \tag{20.4}$$

met gelijkheid alleen wanneer

$$H(\omega) = cF^*(\omega)e^{-j\omega T_{\max}},$$

voor een willekeurige constante c. Hierin noteert F^* de complex geconjugeerde van F. In het tijddomein vinden we voor de impulsrespons $h(t)$, via de inverse Fourier-transformatie,

$$h(t) = cf^*(T_{\max} - t),$$

en voor het uitgangssignaal

$$\begin{aligned} y(t) = f(t) * h(t) &= \int_{-\infty}^{\infty} h(\tau) f(t-\tau) \, d\tau \\ &= c \int_{-\infty}^{\infty} f^*(T_{\max} - \tau) f(t-\tau) \, d\tau. \end{aligned} \tag{20.5}$$

Het maximale instantane signaalvermogen aan de uitgang van het filter is E en treedt op als $t = T_{\max}$. Het ruisvermogen aan de uitgang is gelijk aan

$$N = \frac{N_0}{2} \int_{-\infty}^{\infty} |h(\tau)|^2 \, d\tau = \frac{N_0}{2} E,$$

zodat de maximale instantane signaal-ruis verhouding gelijk is aan $\frac{2E}{N_0}$, onafhankelijk van de precieze *vorm* van het signaal.

[48] Deze definitie wijkt af van een andere veelgebruikte definitie waarin de *gemiddelde* signaalsterkte wordt gebruikt.

20.5 Berekening van optimale filters

Aan de hand van een voorbeeld laten we zien hoe Maple gebruikt kan worden bij de berekening van een optimaal filter.

Voorbeeldopgave

Gegeven het signaal $f(t) = 1$ voor $0 \leq t \leq 1$. Bepaal de overdrachtsfunctie van het optimale filter voor f. Bepaal het uitgangssignaal op twee manieren: eerst via de convolutie in het tijddomein (zie vergelijking (20.5)), en ook via het frequentiedomein door $Y(\omega)$ te bepalen en terug te transformeren. Wat is de rol van $T_{\max}$?

Voorbeeldsessie

```
> restart; with(inttrans):  with(plots):
> f := t -> Heaviside(t)-Heaviside(t-1):
> F := unapply(
  simplify(fourier((Heaviside(t)-Heaviside(t-1)),t,omega)), omega );
```

$$F := \omega \rightarrow \frac{(e^{(-I\,\omega)} - 1)\,I}{\omega}$$

We berekenen de tijdsevolutie van het uitgangssignaal:

```
> ykern := f(Tmax-tau)*f(t-tau); #(f is reeel)
```

$$ykern := (\text{Heaviside}(Tmax - \tau) - \text{Heaviside}(Tmax - \tau - 1))$$
$$(\text{Heaviside}(t - \tau) - \text{Heaviside}(t - \tau - 1))$$

```
> y := int(ykern,tau=-infinity..infinity):
> p1 := plot( subs(Tmax=1,y), t=-5..5, color=black ):
  t1 := textplot( [1,1.03,typeset(T[max]=1)], align=ABOVE ):
  p2 := plot( subs(Tmax=3,y), t=-5..5, color=black ):
  t2 := textplot( [3,1.03,typeset(T[max]=3)], align=ABOVE ):
  display({p1,t1,p2,t2});
```

(zie figuur 35)

Nu via de Fouriergetransformeerde Y:

```
> H := (omega,Tmax) -> exp(-I*omega*Tmax)*evalc(conjugate(F(omega))):
  H(omega,Tmax);
```

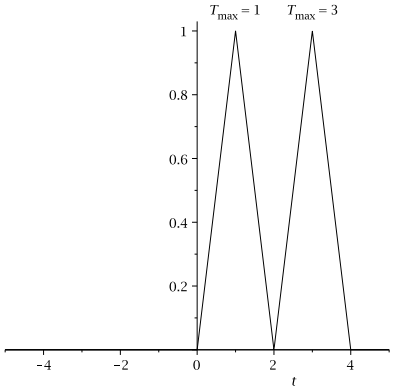
$$e^{(-I\,\omega\ Tmax)} \left(\frac{\sin(\omega)}{\omega} - \frac{(\cos(\omega) - 1)\,I}{\omega} \right)$$

```
> Y := H(omega,Tmax)*F(omega);
```

$$Y := \frac{e^{(-I\,\omega\ Tmax)} \left(\frac{\sin(\omega)}{\omega} - \frac{(\cos(\omega) - 1)\,I}{\omega} \right) \left(e^{(-I\,\omega)} - 1\right)\ I}{\omega}$$

```
> yt := simplify( subs(Tmax=1,invfourier(Y,omega,t)) ):
> plot(yt,t=-5..5);
```

(plaatje niet getoond)



FIGUUR 35. Uitgangssignaal bij verschillende waarden van $T_{\max}$

Toelichting

`invfourier`

Het is aan te bevelen om bij het gebruik van `fourier` en `invfourier` een `simplify` of `evalc` op te geven. We zien dat $T_{\max}$ het moment bepaalt waarop het signaal maximaal wordt. De vorm van het uitgangssignaal verandert er niet door. In theoretische beschouwingen is daarom de grootte van $T_{\max}$ niet relevant; in de praktijk hangt ze af van de tijdvertraging in de realisatie van het filter. Ook in de representatie in het frequentiedomein is dit te zien: het effect van $T_{\max}$ is een faseverschuiving in $H(\omega)$, hetgeen in overeenstemming is met de tijdvertraging in het tijddomein.
Verder zien we dat de weg via het frequentiedomein leidt tot een expliciete uitdrukking voor $y(t)$. Dit geldt ook voor de representatie in het tijddomein. Het reultaat is echter nogal onoverzichtelijk, wegens de vele gevallen die moeten worden onderscheiden. Dat hebben we daarom maar niet afgedrukt. Ten slotte merken we op dat de *hoogte* van de piek in het uitgangssignaal (namelijk 1) gelijk is aan de energie-inhoud E van $f(t)$; dit komt omdat we in (20.5) de constante $c = 1$ hebben genomen. In de praktijk hangt c onder meer af van de versterking die in het filter moet/kan worden gerealiseerd. ◇

20.6 Het discrete geval

In het discrete geval wordt het ingangssignaal bemonsterd, met tijdsinterval T_s. We noteren $f_s[n]$ voor $f(nT)$, met $n \in \mathbb{Z}$. Zonder op

de details in te gaan, kan men zich voorstellen dat de impulsrespons van het systeem gegeven wordt door de discretisatie van de continue impulsrespons $h(t)$. We gaan ervan uit dat de impulsrespons een eindige drager heeft. Het uitgangssignaal $y(t)$ is dan te schrijven als

$$y_s[m] = T_s \sum_{i=1}^{n} h_s[i] f_s[m+i]. \tag{20.6}$$

Denk hierbij aan de factor T_s die erin komt omdat we als het ware de integraal (20.5) numeriek benaderen. Voor een filter geoptimaliseerd voor het doorlaten van een signaal f is dan dus $h_s[i] = (f_s[i])^*$.

Voorbeeldopgave

Beschouw het ingangssignaal $f_s[n]$, met $n > 0$ gegeven door

$$f_s[n] = \begin{cases} 0, & n < 30; \\ nT_s, & 30 \leq n < 50; \\ 0 & n \geq 70. \end{cases}$$

waarbij $T_s = 1/20$. Beschouw verder het systeem uit de vorige voorbeeldopgave, maar nu met discrete impulsrespons $h_s[n]$, een discretisatie van $h(t)$. Maak een plaatje van het uitgangssignaal. Doe hetzelfde met een verruist signaal.

Voorbeeldsessie

```
> restart; with(plots):
> T_s := 1/20:
```

We definieren eerst welk signaal we zouden moeten herkennen na filtering.

```
> expected_signal_length := 20:
  expected_signal :=
  Array( [seq((n-1)*0.05, n=1..expected_signal_length)] ):
```

Nu construeren we het testsignaal:

```
> signal_length := 70:
  f := Array(1..signal_length):
  f[31..50] := expected_signal:
> listplot( f, thickness=2 );
```

(zie figuur 36, links)

Nu maken we de impulsrespons:

```
> h := map( conjugate, expected_signal ):
```

Vervolgens het uitgangssignaal

```
> output_length := signal_length - expected_signal_length:
> ys := m ->
    T_s*add( h[i]*f[m+i], i=1..expected_signal_length):
> y := Array(1..signal_length):
  y[1..output_length] :=
     Array( [seq( ys(n), n=1..output_length )] ):
```

```
> listplot( y, thickness=2 );
```

(zie figuur 36, rechts)

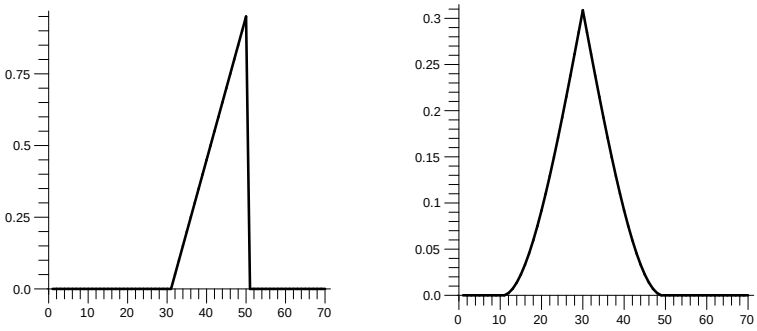
Nu voegen we wat ruis toe:

```
> with(stats):
> map( x->x+0.4*random[normald](), f ):  f := %:
> listplot(f, thickness=2);
```

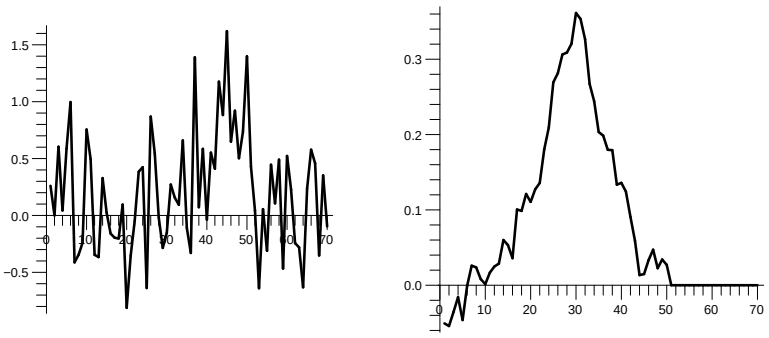
(zie figuur 37, links)

```
> y_noise := Array(1..signal_length):
  y_noise[1..output_length] :=
     Array( [seq( ys(n), n=1..output_length )] ):
> listplot(y_noise, thickness=2);
```

(zie figuur 37, rechts)



FIGUUR 36. Ingangssignaal (links) en uitgangssignaal (rechts)



FIGUUR 37. Ingangssignaal met ruis (links); gefilterd signaal (rechts)

Toelichting

Voor het uitgangssignaal $y_s[m]$, zie formule (20.6) hebben we een 'hulpfunctie' `ys(m)` gemaakt, waarmee we de `Array y` gemakkelijk kunnen opbouwen.
We kunnen in de grafiek zien dat de maximale waarde van het uitgangssignaal wordt bereikt 20 stappen na het moment waarop het verwachte ingangssignaal het filter binnenkomt. Op dat moment vallen de waarden van $h_s[n]$ en $f_s[n]$ samen. Het is ook het eerste moment waarop het volledige signaal 'binnen is'. Wanneer we naar de top van de grafiek van $y(t)$ kijken, zien we dat deze ongeveer 1/3 hoog is; dit correspondeert weer met de energie-inhoud van het signaal (namelijk $\int_0^1 t^2\,dt = 1/3$). Te zien is dat er zelfs bij behoorlijke hoeveelheden ruis nog een detectie mogelijk is. Uiteraard hangt dit af van de toevallige realisatie van de ruis. Echter, zolang we te maken hebben met normaal verdeelde witte ruis is het gebruikte filter – gemiddeld genomen – optimaal.
De laatste 20 punten $y_s[51] \ldots y_s[70]$ in het uitgangssignaal hebben we niet kunnen berekenen; hiervoor zou het ingangssignaal langer moeten zijn om de discrete convolutie uit te voeren. ◇

20.7 Maple en Matlab

Naast Maple is het pakket Matlab ook zeer geschikt voor signaalbewerking. Standaard heeft Matlab een groot aantal bibliotheken waarin onder meer een enorm scala aan – voorgedefineerde – filters te vinden is. Welk van beide pakketten het meest geschikt is voor een bepaalde toepassing is voor een deel een kwestie van smaak en is vaak het onderwerp van levendige discussie. Het meest in het oog lopende verschil is dat Matlab alleen werkt op gediscretiseerde data, en dus niet zo geschikt is voor symbolische manipulaties. Daartegenover staat dat Matlab eenvoudiger te begrijpen is (omdat vrijwel alles numeriek is, zijn er geen ingewikkelde evaluatieregels), goede interfaces heeft naar andere programmeertalen, over meer en uitgebreidere bibliotheken beschikt voor numeriek werk, en, last but not least, zeer veel gebruikt wordt in de industrie.
Wanneer men beschikt over het pakket Matlab is het mogelijk vanuit een Maple-sessie een Matlab-sessie op te starten. Deze sessie is dan gekoppeld aan de Maple-sessie. Door variabelen vanuit Maple

over te dragen naar Matlab, kan men dan in deze Matlab-sessie allerlei Matlab-procedures op deze variabelen toepassen. De resultaten kunnen dan weer worden overgedragen aan de Maple-sessie.
De Maple-bibliotheek om Matlab-sessies op te starten, heeft de naam `Matlab`. Door deze bibliotheek te laden wordt, als men in een Windows-omgeving werkt, automatisch een Matlab command window geopend; in een Linux-omgeving is dat niet het geval. Het overhevelen van een variabele naar Matlab gaat met de opdracht

`setvar`

```
setvar("Matlab_naam", Maple_var)
```

Het eerste argument – een string – is de naam die de variabele krijgt in de gekoppelde Matlab-sessie. Het tweede argument is de variabele uit de Maple-sessie. Raadpleeg zelf `?setvar` voor nadere informatie. Het commando `setvar` werkt alleen op vector- en matrix-achtigen. Men kan dus niet een *willekeurige* variabele overdragen. Via

`getvar`

```
getvar("Matlab_naam")
```

haalt men het resultaat terug naar Maple.

`evalM` Het commando `evalM` kan worden gebruikt om in een Maple-sessie een Matlab-commando te geven. Dit is vooral handig voor Linux-gebruikers.
We illustreren dit met een eenvoudige voorbeeldopgave.

Voorbeeldopgave

(Zie ook de eerste voorbeeldopgave van deze Module.) Gegeven signalen $f_1(t) = \sin(t)$ en $f_2(t) = \sin(10t)$, beide voor $0 \leq t < 10$. Gebruik de Matlab-procedure `remez` om een laagdoorlaatfilter te construeren zodanig dat f_1 wel, en f_2 niet wordt doorgegeven. Maak gebruik van een discretisatie met sampletijd $T_s = 0.25$ (Ga na dat dit voldoende is!) De (Matlab-) aanroep

```
b=remez(11,[0 0.1 0.2 1],[1 1 0 0]);
```

maakt een filter met frequentiekarakteristiek

$$H(j\omega) = \begin{cases} 1, & 0 \leq \omega \leq 0.1 f_{Ny} \\ \text{onbepaald}, & 0.1 f_{Ny} < \omega < 0.2 f_{Ny} \\ 0, & 0.2 f_{Ny} < \omega \leq 1 \end{cases}$$

Hierin is f_{Ny} de Nyquist-frequentie; deze is in ons geval 8π, namelijk $\frac{2\pi}{T_s}$. Met de Matlab-procedure `freqz` is de frequentiekarakteristiek van het filter te bekijken. Het getal 11 zegt iets over de orde van het filter, en daarmee ook iets over de lengte (in aantallen samples) van het filter. Probeer orde 11 en 21.

Voorbeeldsessie

```
> restart; with(plots):
> with(Matlab);
```

[*AddTranslator* , *FromMFile* , *FromMatlab* , *chol*, *closelink*, *defined*, *det*, *dimensions*, *eig*, *evalM*, *fft*, *getvar*, *inv*, *lu*, *ode15s*, *ode45* , *openlink*, *qr*, *setvar*, *size*, *square*, *transpose*]

```
> rect := t -> piecewise( t<0, 0, t<=1, 1, 0 ):
> f1 := t -> sin(t)*rect(t/10):
  f2 := t-> sin(10*t)*rect(t/10):
> T_s := 0.25:  end_time := 20:
> Nsamples := floor(end_time/T_s):
> t_list := Vector(Nsamples, i->i*T_s):
  f1_list := Vector(Nsamples, i->evalf(f1(i*T_s))):
  f2_list := Vector(Nsamples, i->evalf(f2(i*T_s))):
```

exporteren naar Matlab

```
> setvar("t_list",t_list);
  setvar("f1_list",f1_list):  setvar("f2_list",f2_list):
```

Nu geven we in de Matlab-sessie de volgende commando's:

```
b=remez(11,[0 0.1 0.2 1],[1 1 0 0]);
y2_11=filter(b,1,f2\_list);
y1_11=filter(b,1,f1\_list);
b=remez(21,[0 0.1 0.2 1],[1 1 0 0]);
y1_21=filter(b,1,f1\_list);
y2_21=filter(b,1,f2\_list);
```

Als er geen Matlab-venster is (of als u dat niet wilt gebruiken) dan kan het ook direct in de Maplesessie:

```
> evalM("b=remez(11,[0 0.1 0.2 1],[1 1 0 0])");
  evalM("y2_11=filter(b,1,f2_list)");
  evalM("y1_11=filter(b,1,f1_list)");
  evalM("b=remez(21,[0 0.1 0.2 1],[1 1 0 0])");
  evalM("y1_21=filter(b,1,f1_list)");
  evalM("y2_21=filter(b,1,f2_list)");
```

Importeren uit Matlab:

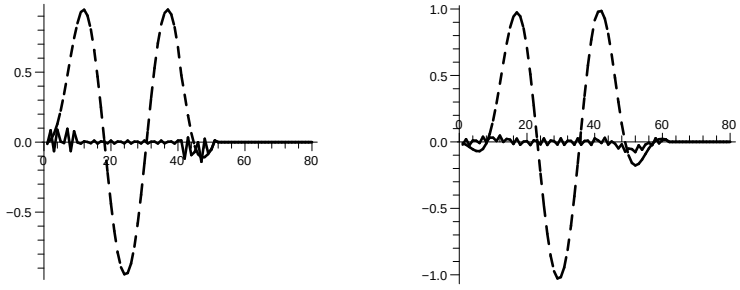
```
> y2_11 := getvar("y2_11");
```

$$y2\_11 := \begin{bmatrix} \text{80 Element Column Vector} \\ \text{Data Type : float[8]} \\ \text{Storage : rectangular} \\ \text{Order : Fortran\_order} \end{bmatrix}$$

```
> y1_11 := getvar("y1_11"):
  y1_21 := getvar("y1_21"):
  y2_21 := getvar("y2_21"):
> p1 := listplot(y1_11,linestyle='DASH',thickness=2):
  p2 := listplot(y2_11,linestyle='SOLID',thickness=2):
> p3 := listplot(y1_21,linestyle='DASH',thickness=2):
  p4 := listplot(y2_21,linestyle='SOLID',thickness=2):
> display({p1,p2});
```

(zie figuur 38, links)

```
> display({p3,p4});
```
(zie figuur 38, rechts)



FIGUUR 38. Filtering van $\sin(t)$ (streepjeslijn) en $\sin(10t)$ (dikke lijn) met een laagdoorlaatfilter verkregen via de Matlab-procedure `remez` . De orde van dit filter is 11 (links) en 21 (rechts)

Toelichting

Om te voorkomen dat er niet gedefinieerde waarden in de samples terecht komen hebben we de functie `rect`, met behulp van `piecewise`, voor *alle* waarden van t gedefinieerd, dus óók in de sprongpunten. De uit Matlab teruggehaalde resultaten blijken in Maple terecht te komen in een kolomvector. Vergelijking van de resultaten met die uit de eerste voorbeeldsessie van deze Module leert dat globaal gezegd het gedrag van het uitgangssignaal gelijk is. Wat ook kan worden opgemerkt is dat een filter van hogere orde het niet per se beter hoeft te doen dan een filter van lagere orde. Wel zijn de inslinger- en uitslingertijd langer. ◇

Opgave 20.1

(a) Bereken de Laplace-getransformeerde $F(s)$ van de functie $t \mapsto \cos(t)$.

(b) Bereken de Fourier-getransformeerde van f en vergelijk dit met $F(j\omega)$.

(In het algemeen geldt dat $F(j\omega)$ gelijk is aan de Fourier-getransformeerde van f wanneer de integraal $\int_0^\infty f(t)e^{-st}\,dt$ bestaat voor $s = j\omega$.)

Opgave 20.2

(a) Schrijf een procedure die de overdrachtsfunctie $H_n(s)$ van het Butterworth-filter van orde n berekent.
(b) Teken in één figuur de grafieken van $|H_n(j\omega)|$ voor $\omega \in [-10..10]$, voor $n = 1, 2, 4, 8$.
(c) Maak gebruik van een lineaire transformatie in het s-domein om met behulp van het Butterworth laagdoorlaatfilter een zogenaamd banddoorlaatfilter te benaderen. Dit is een filter met $|H(j\omega)| = 1$ voor $\omega \in [\omega_-, \omega_+]$ (de doorlaatband) en $|H(j\omega)| = 0$ elders. Teken de frequentiekarakteristiek van dit filter.

Opgave 20.3

(a) Teken in één figuur de frequentiekarakteristiek van het Butterworth laagdoorlaatfilter van orde 1 en 3.
(b) Bereken de respons van het filter op het ingangssignaal $u(t) = \sin(t/2) + \cos(2t)$. Gebruik de Laplace-transformatie en de inverse Laplace-transformatie, voor de ordes 1 en 3. Teken in één figuur de uitgangssignalen voor $t \in [0 \ldots 10]$, alsmede díe term uit de functie u (dus $\sin(t/2)$ of $\sin(2t)$) die het meest lijkt op het uitgangssignaal van het derde orde filter.

Voor hogere ordes van het filter werkt de route via Laplace niet fijn meer: Maple heeft veel moeite met de inverse Laplace-transformatie. Om toch het tijdsgedrag van het uitgangssignaal te kunnen bestuderen, kunnen we onze toevlucht nemen tot de representatie van het filter in de vorm van een gewone differentiaalvergelijking. De oplossing hiervan kunnen we dan numeriek berekenen.

(c) Bereken het Butterworth laagdoorlaatfilter van de orde 8, en teken de grafiek van $y(t)$ door gebruik te maken van de Mapleprocedure `odeplot`. Hierbij is $y(t)$ het uitgangssignaal behorende bij het ingangssignaal $u(t)$ uit onderdeel (a). Natuurlijk moet u eerst de bijbehorende differentiaalvergelijking bepalen.

Opgave 20.4

Gegeven het verwachte ingangssignaal $s(t) = t$ voor $3 \leq t \leq 4$, en 0 elders. Bepaal het optimale filter voor detectie van dit signaal in ruis. Geef de overdrachtsfunctie, de impulsrespons, alsmede het uitgangssignaal van dit filter.

Opgave 20.5

Gegeven het verwachte ingangssignaal $s(t) = \cos(10t)$ voor $0 \leq t \leq 1$, en 0 elders. Bepaal het optimale filter voor detectie van dit signaal in ruis. Geef de overdrachtsfunctie, de impulsrespons, alsmede het uitgangssignaal van dit filter.
Verstoor het ingangssignaal met ruis, en probeer in hoeverre aan de uitgang van het filter te achterhalen is of er al dan niet het signaal $s(t)$ is gepasseerd.

Opgave 20.6

Gegeven het signaal $s(t) = e^{jt^2}$ voor $T \leq t \leq T+1$, en 0 elders. Neem $T = 3$. Neem een bemonstering van 50 Hz. Bepaal het optimale filter voor s. Bekijk het signaal aan de uitgang van het filter.
Als het goed is, ziet u een signaal waarin de meeste energie te vinden is in een smalle band rond het maximum. Deze band is veel smaller dan de lengte van het oorspronkelijke signaal s.
Dit type signalen wordt veel gebruikt om nauwkeurig afstanden te bepalen van objecten die ver verwijderd zijn. Hierbij wordt een (elektromagnetische of akoestische) puls gegenereerd, en vervolgens wordt geluisterd of, en zo ja, op welk tijdstip een echo van dit signaal wordt waargenomen. Om objecten op grote afstand waar te nemen is het van belang om voldoende energie uit te zenden: gemakkelijk is af te leiden dat de hoeveelheid ontvangen energie in de echo-puls met R^{-4} afneemt. Deze hoeveelheid energie is evenredig met de lengte van de puls. Het uitzenden van lange pulsen verkleint echter de nauwkeurigheid van de afstandsschatting: die is ruwweg van de ordegrootte van de lengte van de uitgezonden puls. Door pulsen met een zekere spectrale breedte te gebruiken vindt men een betere afstandsschatting met toch voldoende uitgezonden energie. De nauwkeurigheid is ongeveer omgekeerd evenredig met de gebruikte bandbreedte. Deze techniek draagt de naam *pulscompressie*.

Opgave 20.7

Probeer eens het effect zichtbaar te maken wanneer niet precies het verwachte signaal f het filter binnenkomt. Men kan hierbij denken aan ruis opgeteld bij het signaal, een kleine verandering in frequentie, of een verstoorde fase in f. Neem voor het filter h een van de filters uit de voorgaande drie opgaven.