Transformer, Signaler och System
SSY080

Erik Thorsell
Robert Gustafsson

2015-10-16

3.1 Fourierserie Uppbyggnad

a)

En fyrkantssignal enligt Figur 3 i lab-pm kan betecknas på följande vis:

$$x(t) = \begin{cases} 1 & 0 < t < \frac{T}{2} \\ -1 & \frac{T}{2} < t < T \end{cases}$$

För att beräkna Fourierkoefficienterna för signalen använde vi oss av följande kända uttryck från kurslitteraturen:
$2C_k = A_k - jB_k$ (A, B $\in \mathbb{R}$)
$C_k = \frac{1}{T}\int_T x(t)e^{-jkw_0t}dt$

Beräkningar följer nedan:

$$C_k = \frac{1}{T}\int_0^T x(t)e^{-jkw_0t}dt = \frac{1}{T}\left(\int_0^{\frac{T}{2}} e^{-jkw_0t}dt - \int_{\frac{T}{2}}^T e^{-jkw_0t}dt\right) =$$

$$\frac{1}{T}\left(\left[\frac{e^{-jkw_0t}}{-jkw_0}\right]_0^{\frac{T}{2}} - \left[\frac{e^{-jkw_0t}}{-jkw_0}\right]_{\frac{T}{2}}^T\right) =$$

$$\frac{1}{T}\left(\frac{e^{\frac{-jkw_0T}{2}}}{-jkw_0} - \frac{1}{-jkw_0} - \left(\frac{e^{-jkw_0T}}{-jkw_0} - \frac{e^{\frac{-jkw_0T}{2}}}{-jkw_0}\right)\right) =$$

$$\frac{2e^{\frac{-jkw_0T}{2}} - e^{-jkw_0T} - 1}{-jkw_0T} =$$

$$\frac{1}{-jkw_0T}\left(2e^{\frac{-jkw_0T}{2}} - e^{-jkw_0T} - 1\right) =$$

$$\frac{1}{-jkw_0T}\left(2cos\left(\frac{kw_0T}{2}\right) - 2jsin\left(\frac{kw_0T}{2}\right) - cos\left(kw_0T\right) + jsin\left(kw_0T\right) - 1\right) =$$

$$\frac{1}{-jk2\pi}(2cos(k\pi) - 2jsin(k\pi) - cos(k2\pi) + jsin(k2\pi)) =$$

$$\begin{cases} \frac{1}{-jk2\pi}(2-1-1) = 0 & \text{Om k jämn.} \\ \frac{1}{-jk2\pi}(-2-1-1) = \frac{-2j}{k\pi} & \text{Om k udda.} \end{cases}$$

$$2C_k = A_k - jB_k$$

Eftersom $C_k = 0$ för jämna k $\Rightarrow A_k = 0$

$$\Rightarrow 2C_k = -jB_k \Rightarrow B_k = -2C_k/j = -2j/k\pi \Leftrightarrow B_k = \frac{4}{k\pi}$$

b)

Givet i labpm finns kod för att rita ut en sinusvåg. Genom att modifiera den koden kan vi genom våra erhållna Fourierkoefficienter rita upp en bättre och bättre approximation av en fyrkantsvåg. Koden för att beskriva fyrkantsvågsapproximationen finns nedan och figur 1 visar vår approximerade fyrkantsvåg efter summationsindex satt till 100.

```
T=1;
w=2*pi/T;
M=200;
x=0;
t=T*(0:M-1)/M;

for n=1:100
    Ak = 0;
    Bk = 4*mod(n,2)/(n*pi);
    x = x + Ak*cos(n*w*t) + Bk*sin(n*w*t);
end
plot(t,x)
```

3.2 Linjära System och Sinusar

a)

Givet i labpm finns ekvation 12 enligt: $G(s) = \frac{(s+0.1)(s+10)}{(s+1)(s^2+s+9)} = \frac{s^2+10.1s+1}{s^3+2s^2+10s+9}$ detta polynom kan tecknas som täljare och nämnare i Matlab - samt ovandlas till en överföringsfunktion - med följande kod:

```
num = [1 10.1 1];
den = [1 2 10 9];
Gs=tf(num, den);
```

Bodediagrammet samt systemets pol- och nollställen kan ses i figur 2 och figur 3. I bodediagrammet kan vi se hur systemet G(s) förändras med avseende på $s = j\omega$. Den övre delen av diagrammet visar $|G(s)|$ och den undre delen visar $arg\{G(s)\}$. Som vi kan se i den övre delen så dämpas amplituden ungefär för $\omega < 1$ och $5 < \omega$. För $1 < \omega < 5$ ser vi dock att amplituden ökar. I den nedre delen av diagrammet ser vi att en markant fasförskjutning sker $\omega \simeq 3$.

Som väntat finner vi, i det nedre diagrammet, kryss vilka markerar systemets nämnares nollställen (poler) och cirklar vilka markerar systemets täljares nollställen (noder). I det nedre diagrammet ser vi särskilt polerna vid $s = \alpha \pm j3$ vilka förtäljer att amplituden kommer nå sin kulmen vid $\omega = j3$ varefter amplituden sjunker med 20dB per dekad.

b)

De tre sinussignalerna ($x1 = sin(t), x2 = sin(3t), x3 = sin(5t)$) finns att beskåda i figur 4.

c)

Efter att ha låtit signalerna passera genom vårt givna system ($G(s)$) erhöll vi tre nya signaler ($y1, y2, y3$) vilka hade förändrad amplitud och fas gentemot vår insignal. Detta var givetvis väntat. Den svarta kurvan i figur 5 visar insignalen och den blå visar signalen vilken returnerades av lsim. Vidare beräknade vi även amplitud och fas var och en för sig varefter vi lät den signalen gå genom systemet för att erhålla en signal lika med den som erhölls av lsim. Detta för att bekräfta ekvation 2 i labpm.

Koden för att bekräfta ekv 2 kan ses nedan:

```
phi1 = angle(evalfr(Gs,1j));
x1p = sin(t+phi1);
y1p = abs(evalfr(Gs,1j))*x1p;

phi2 = angle(evalfr(Gs,3j));
x2p = sin(3*t+phi2);
y2p = abs(evalfr(Gs,3j))*x2p;

phi3 = angle(evalfr(Gs,5j));
x3p = sin(5*t+phi3);
y3p = abs(evalfr(Gs,5j))*x3p;
```

Här kommer signalerna $y1p, y2p$ samt $y3p$ vara lika med $lsim1, lsim2$ och $lsim3$ vilka erhölls från lsim och vi ser alltså att ekv 2 stämmer. Ekvation 2 lyder som bekant: $y(t) = |G(j\omega)|sin(\omega t + arg\{G(j\omega)\})$.

3.3 Periodiska Insignaler och DFT

a)

För att generera och rita upp en fyrkantsvåg i Matlab använde vi oss av följande kod:

```
N = 2.^13;
F = 100;
Ts = 1/F;
t = 0:Ts:40*pi;
x = square(t);
plot(t, x)
```

Den resulterande grafen, samt dess fourierkoefficienter (vilka ska räknas ut i senare uppgifter) finns att finna i figur 6. Eftersom $x(t) = -x(-t)$ ser vi att signalen är udda.

b)

De tre första - nollskilda - fourierkoefficienterna för fyrkantsvågen är: 1.2732, 0.4244, 0.2546 och beräknades med hjälp av $B_k = \frac{4}{k\pi}$ där $k = 1,3,5$.

c)

Genom att använda den snabba fouriertransformen (den inbyggda matlabfunktionen *fft*) kunde vi beräkna den diskreta fouriertransformen av fyrkantsvågen. Grafen till uppgiften återfinns i figur 6 vilket är samma figur som i uppgift a).

Spektrumet består av $\omega_k = 1, 3, 5$ och det går även att skymta $\omega_k = 7$.

d)

Genom att använda sambandet i ekvation 10 ($B = \frac{2|X[k_0]|}{N}$) i labpm kunde vi beräkna de praktiska värdena för fourierkoefficienterna vi tog fram i uppgift b. Resultatet för de båda ges i tabellen nedan.

	1	3	5
Teoretiska	1.2732	0.4244	0.2546
Praktiska	1.2702	0.4154	0.2398

e)

Fyrkantssignalen applicerades på $G(s)$ varefter de tre första fourierkoefficienterna bestämdes teoretiskt med hjälp av ekvation 8 i labpm. Ekvation 8 lyder som bekant: $y(t) = A_0 g(0) + \sum_{k=1}^{\infty} B_k g(k\omega_0) sin(k\omega_0 t + \Phi(k\omega_0))$. Då $g(w) = |G(jw)|$ och B_k är fourierkoefficienten för insignalen kan vi beräkna fourierkoefficienten för utsignalen vid ett givet k med hjälp av:

```
abs(evalfr(H,kj))*(4/k*pi)
```

Därefter användes återigen den snabba fouriertransformen för att beräkna koefficienterna i praktiken. En tabell med värdena ges nedan och i figur 7 ges återigen spektrumet för fyrkantsvågen i frekvensdomänen samt spektrumet för utsignalen när fyrkantsvågen applicerats på systemet $G(s)$.

Eftersom ett system bara kan förändra en signals amplitud och fas är spektrum oförändrat före och efter det att systemet har applicerats på signalen.

	1	3	5
Teoretiska	1.1279	1.4020	0.1666
Praktiska	1.023	1.3378	0.1513

3.4 Notch-filter

a)

Polynomet vilket söks i uppgiften är $Ts = s*(s-1j)*(s+1j)*(s-5j)*(s+5j)*(s-7j)*(s+7j)*(s-9j)*(s+9j)$ där komplexkonjugerade nollställen används för att få ett polynom med reella koefficienter. Om man låter nämnaren vara lika med 1 och använder Matlabs inbyggda funktion för att generera en överföringsfunktion kan det resulterande systemet visas med hjälp av ett Bodediagramm. Detta diagramm finns att beskåda i figur 8. Som väntat ligger det notchar i $\omega = 1, 5, 7, 9$. Anledningen till att det är olämpligt att implementera ett filter med fler nollställen än poler är att vi då inte får den eftersträvansvärda dämpningen för alla frekvenser utan istället kommer förstärka vissa frekvenser och feedbacken kommer ge en oändligt förstärkt signal.

b)

Vi lade till lika många poler som vi har nollställen för att få en acceptabel dämpning. Bodediagrammet finns att se i figur 9. $p = -4$ är ett bra val eftersom vi strävar efter att släppa igenom frekvenser vid $\omega = 3$ och dämpa de övriga, särskilt de högre, frekvenserna.

c)

Genom att beräkna mag2db(abs(evalfr(sys,1000j))) - mag2db(abs(evalfr(sys,100j))) fann vi att 12 poler erfordrades för att en dämpning på 60 dB per dekad för vinkelfrekvenser $\omega \gg 9\frac{\text{rad}}{s}$. Detta återspeglas även i vår slutliga uppritning av utsignalen då färre poler ger disskontinuiteter i signalen medan 12 poler ger en fin sinusformad kurva. Bodediagrammet som visar detta finns att beskåda i figur 10.

För att filtret ska bli realiserbart krävs det att det finns minst lika många poler som det finns nollställen. Detta för att ge den eftersträvade dämpningen och inte råka ut för feedback som resulterar i en oändlig förstärkning. I vårt fall erfodras 10 stycken poler då vi har 10 stycken nollställen.

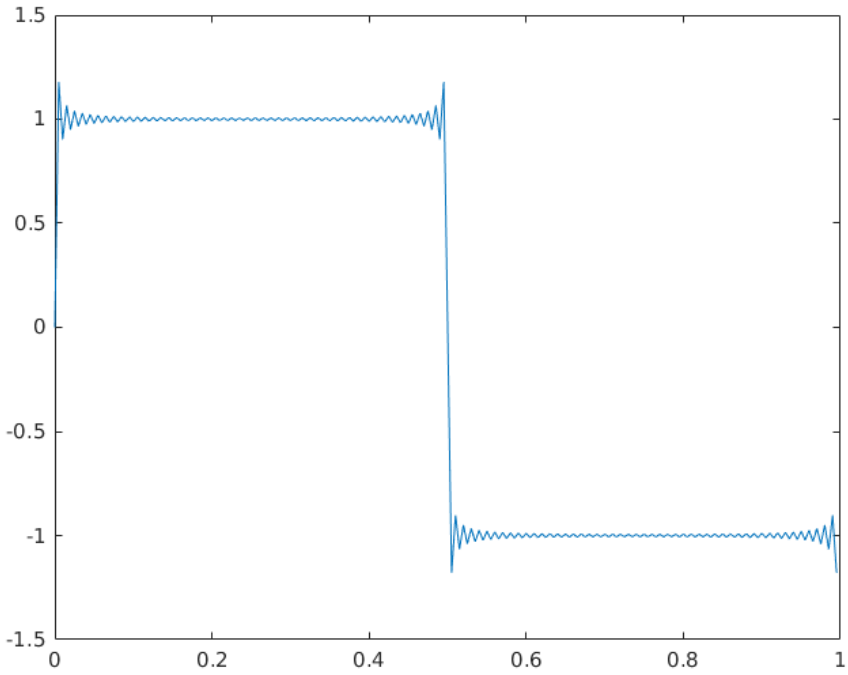
d)

För att förstärka vårat system skalade vi täljarpolynomet med $|H(j3)|$. Filtrets bodediagramm finns återgivet i figur 11.

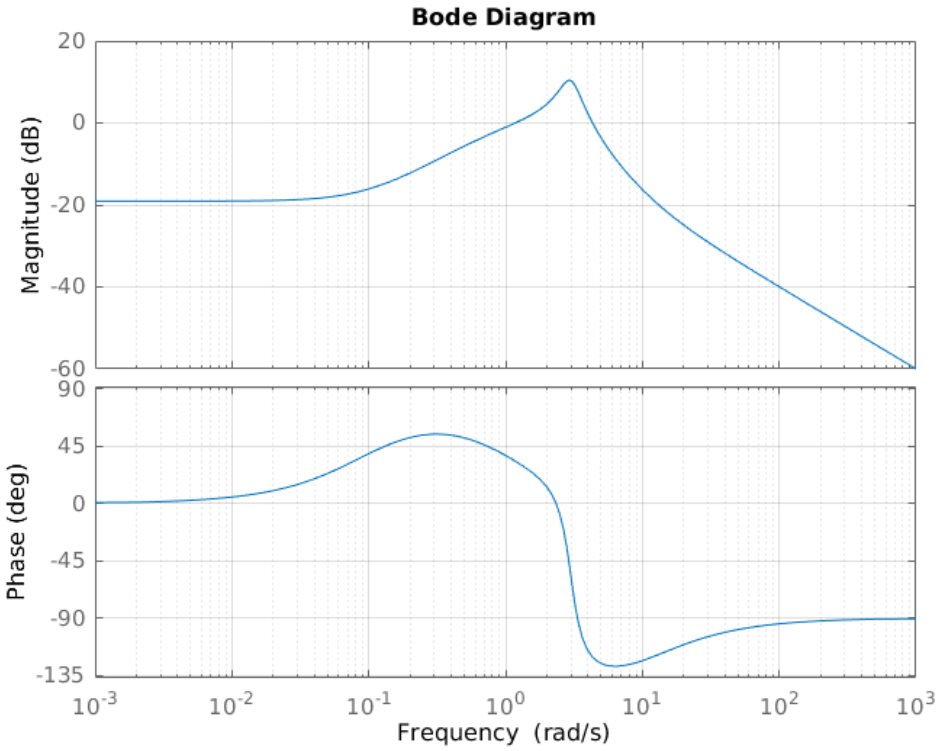
e)

I figur 12 och figur 13 syns respektive insignal $x(t)$ i svart och utsignal $y(t)$ i blått. Man kan tydligt se att den stationära utsignalen liknar en sinus med frekvensen $\omega = 3$. Som brukligt kan vi även avläsa från graferna i figur 14 och figur 15 att amplituden är den förväntade vilket även stämmer överrens med tabellerna från uppgift 3.3 d respektive e.

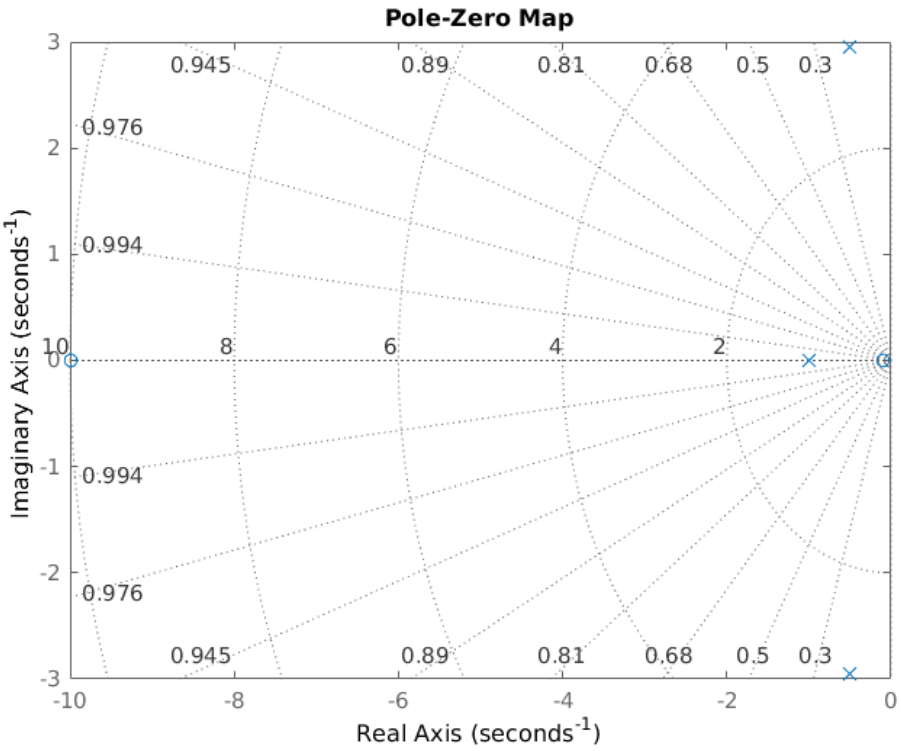
Figur 1: Fyrkantsvåg för uppgift 3.1 b).



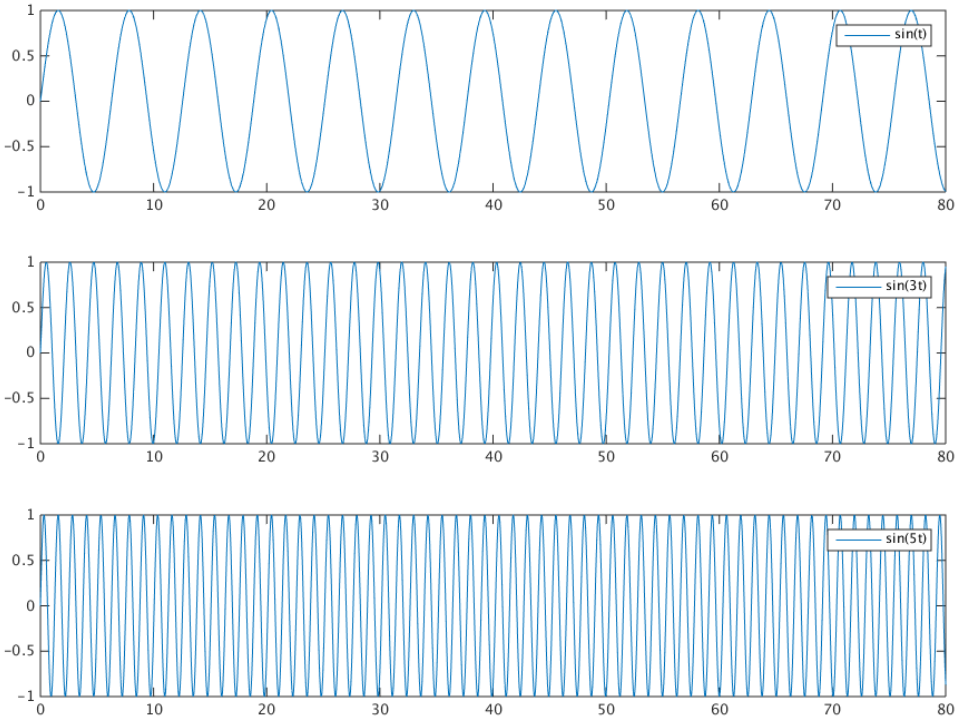
Figur 2: Bodediagram för G(s).



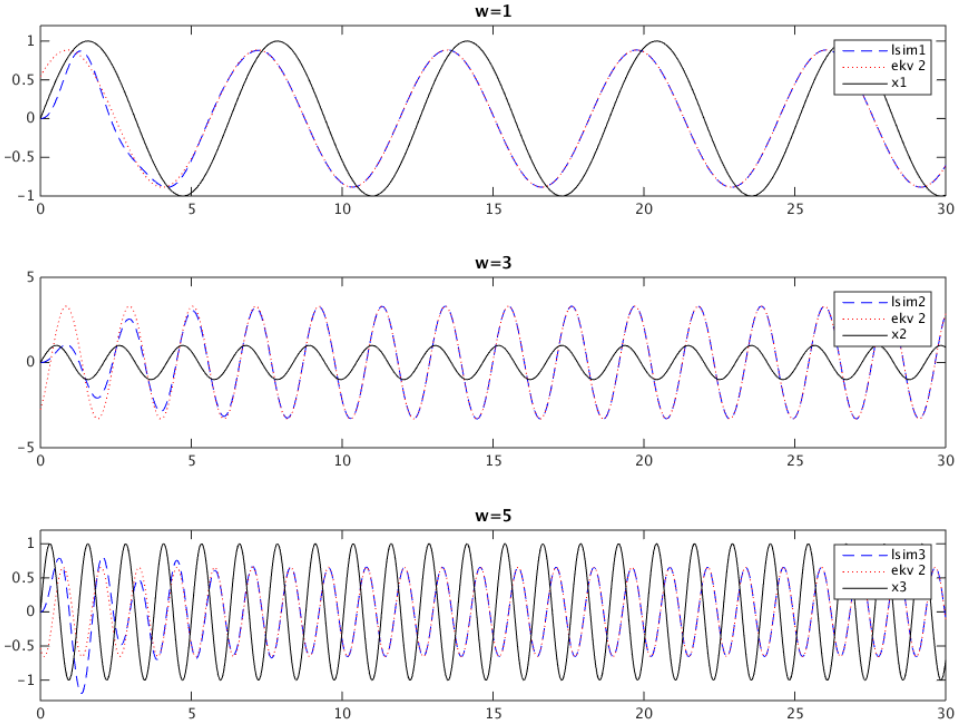
Figur 3: Nod- och polgraf för G(s).



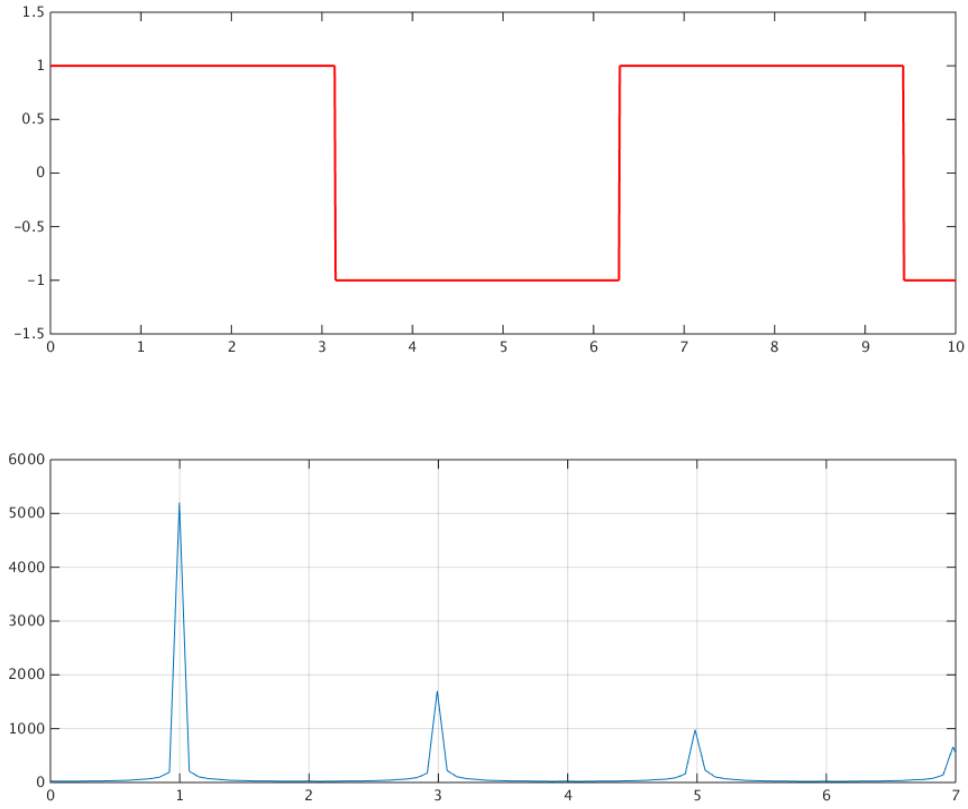
Figur 4: De tre insignalerna givna i uppgift 3.2 b).



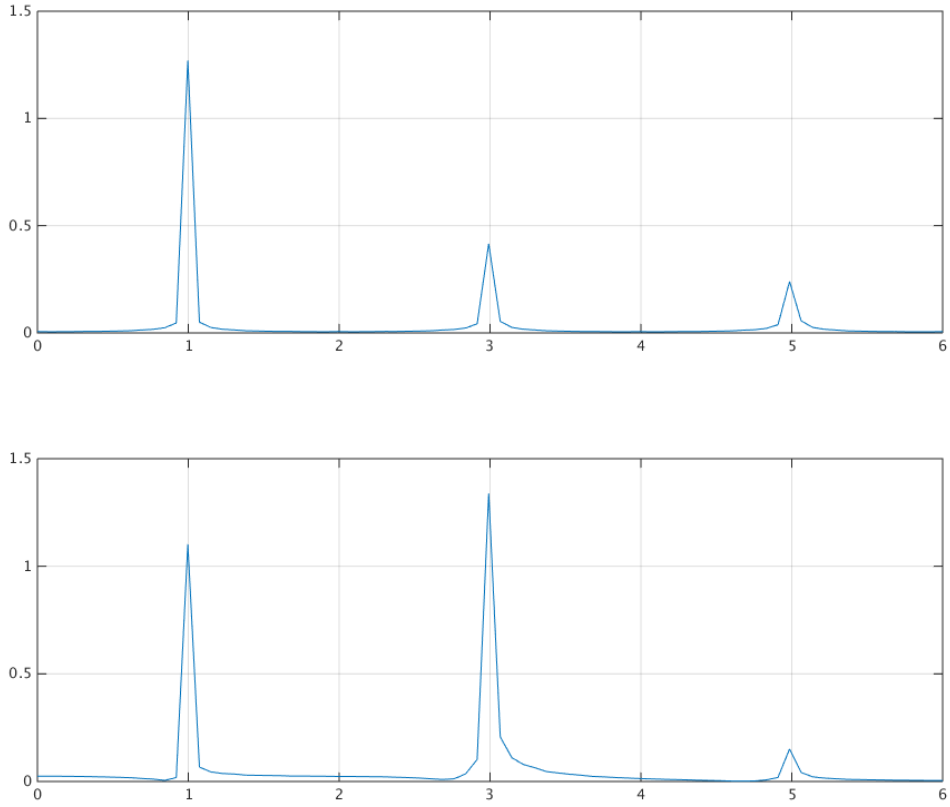
Figur 5: Utsignaler samt insignal för uppgift 3.2 c).



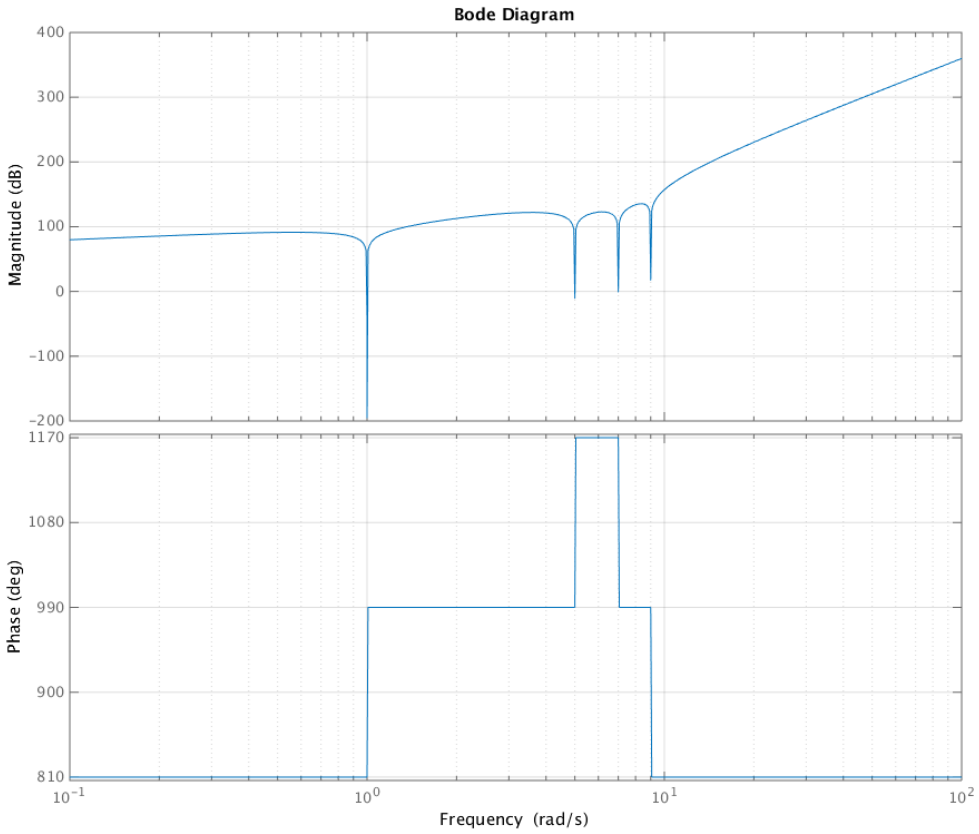
Figur 6: En, av matlab genererad, fyrkantsvåg samt spektrumet för dess snabba fouriertransform med avseende på ω_k.



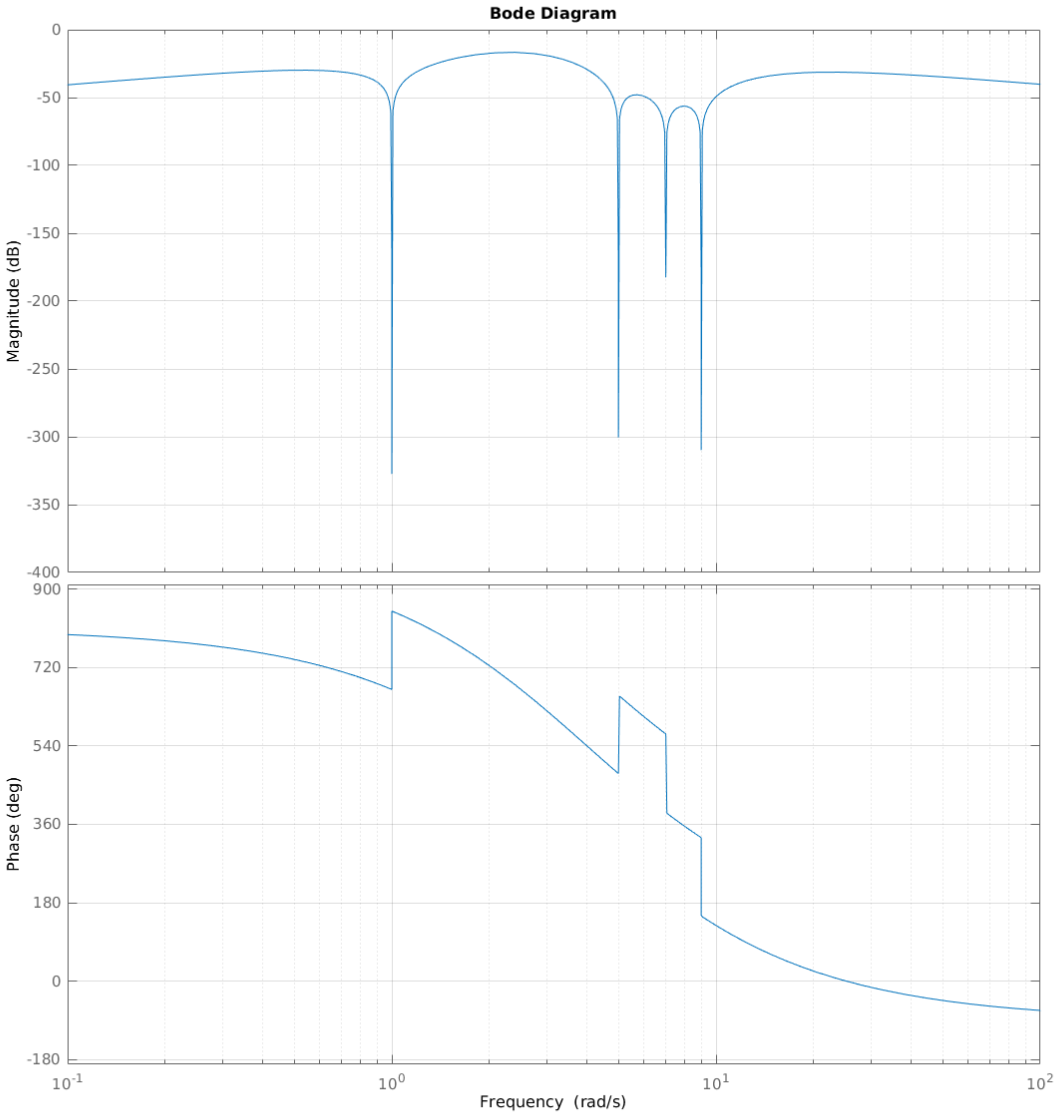
Figur 7: De tre första - nollskiljda - fourierkoefficienterna för fyrkantsvågen samt för utsignalen som ges när fyrkantsvågen passerar systemet G(s).



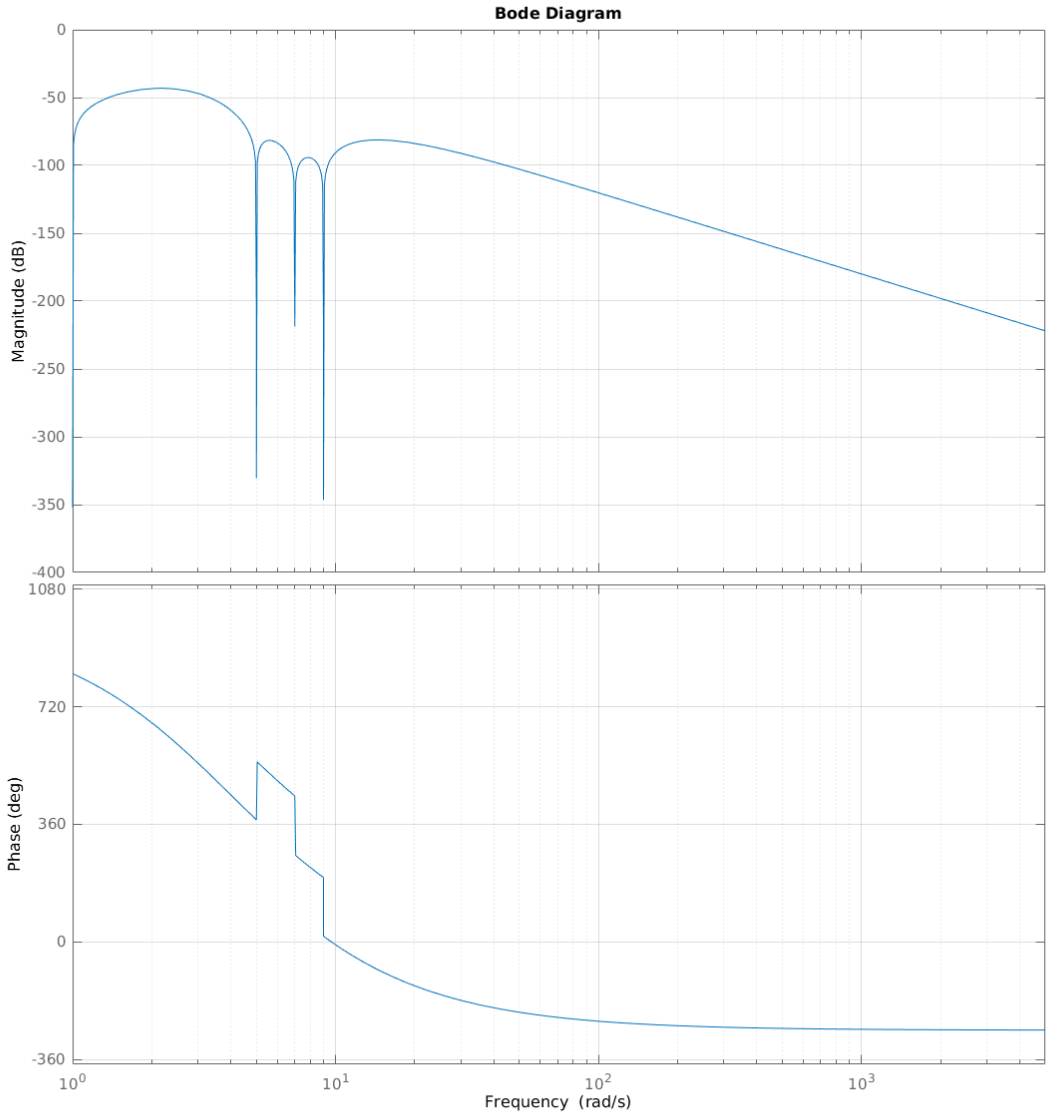
Figur 8: Bodediagram för vårt egentillverkade system enligt uppgift 3.4 a).



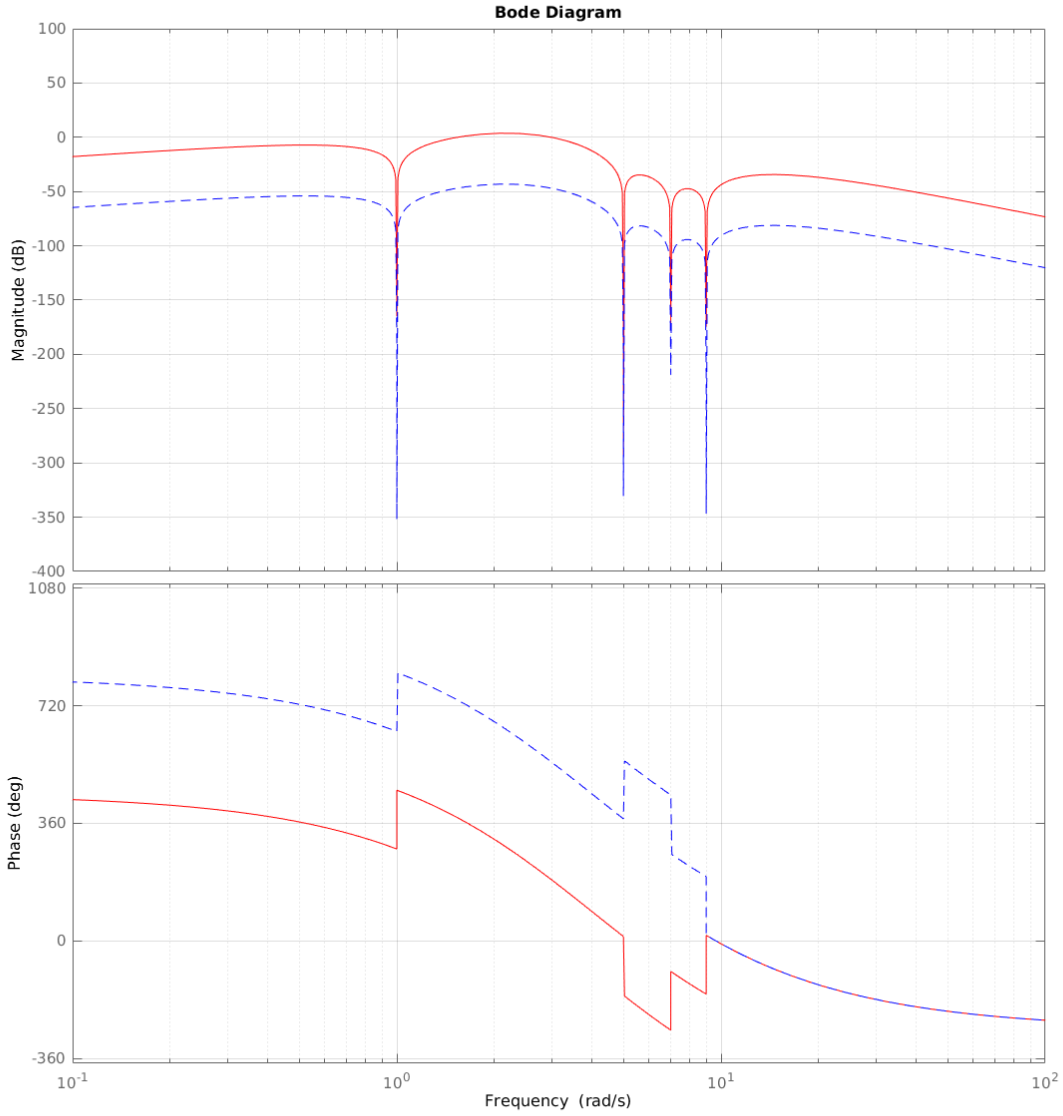
Figur 9: Bodediagram som visar systemet efter att $\omega > 9$ dämpats.



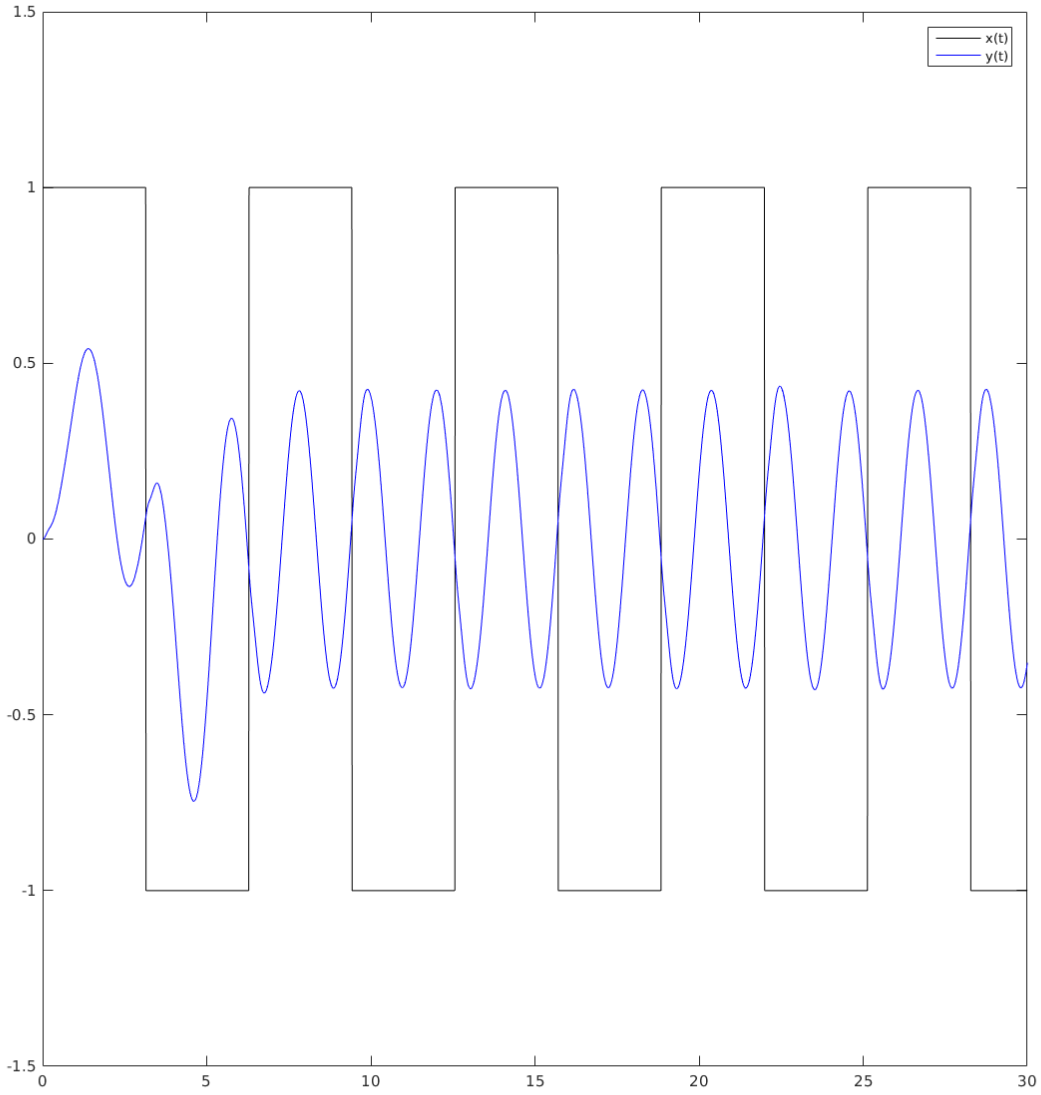
Figur 10: Bodediagram som visar systemet efter att $\omega \gg 9$ dämpats.



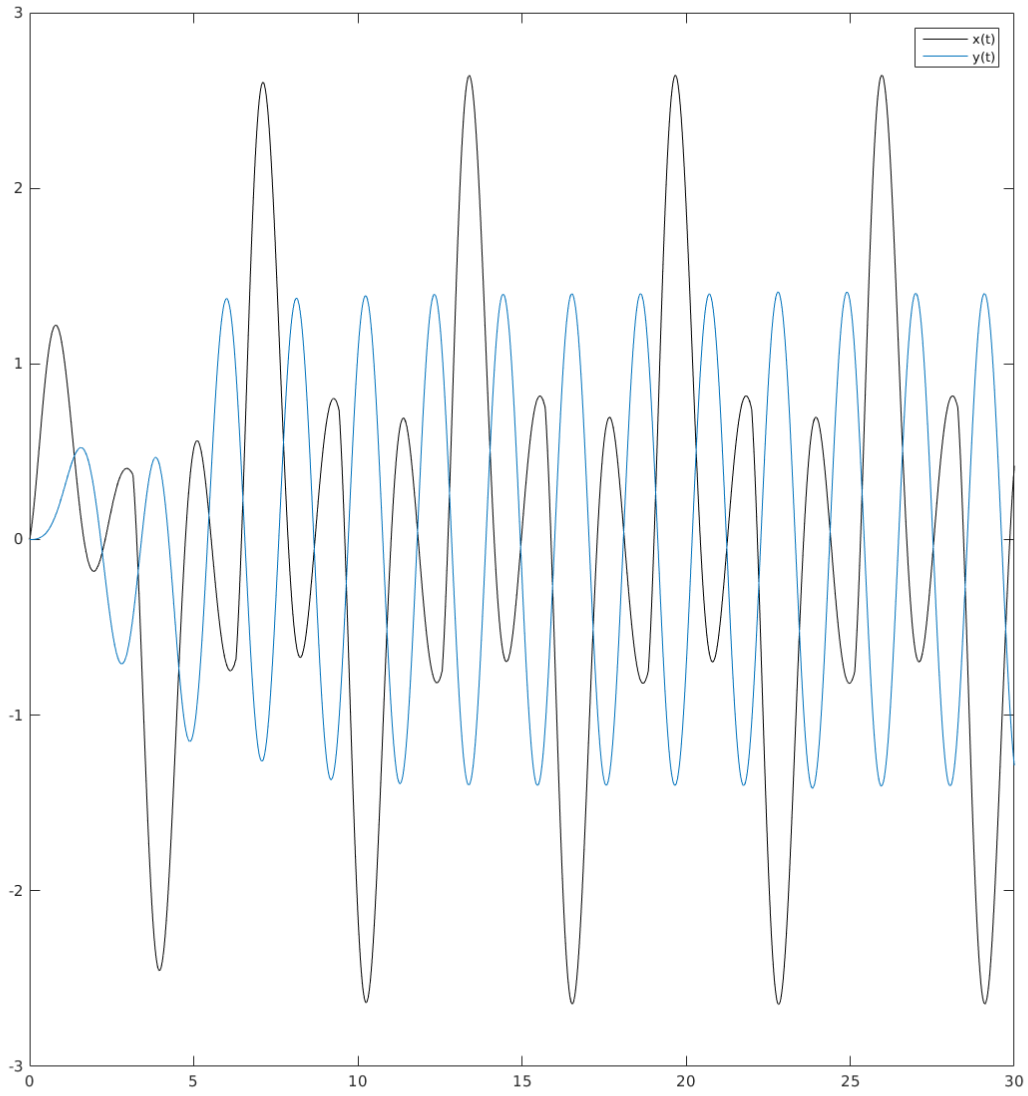
Figur 11: Bodediagram som visar vårt system innan amplitudkorrigering i blått och det slutliga systemet i rött.



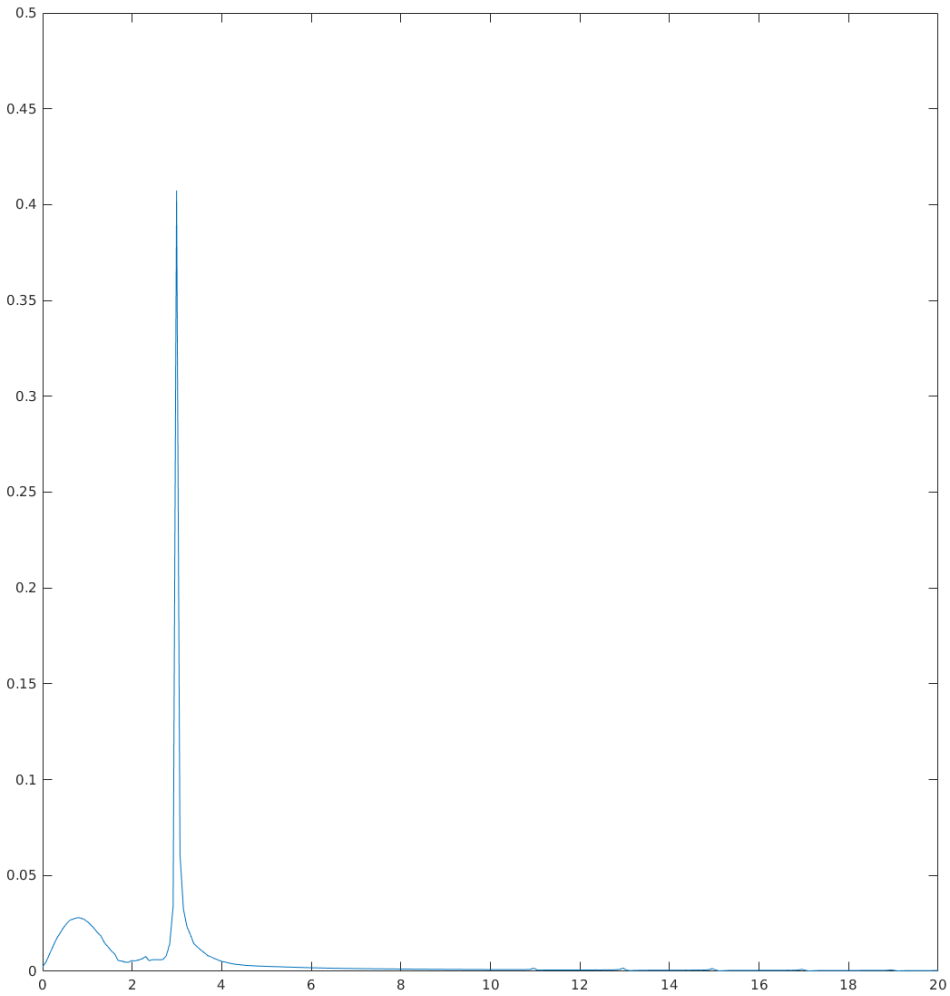
Figur 12: När x-signalen passerat systemet synns tydligt att frekvensvinkeln $\omega = 3$ och signalen är tydligt sinusformad.



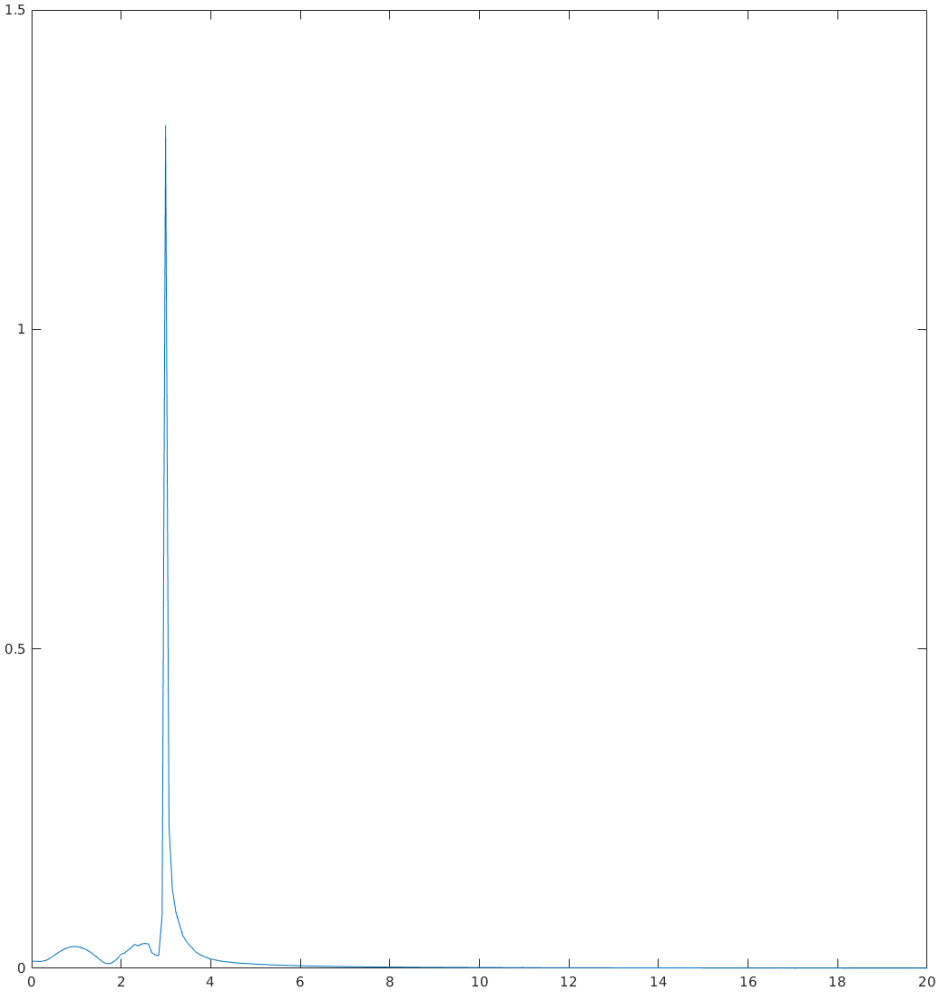
Figur 13: När y-signalen (fyrkantsvågen som först passerat G(s) och sedan vårt notchfilter) slutligen ritas upp återfinns den förväntade vinkelfrekvensen och sinuskarraktären.



Figur 14: Här ses tydligt att $\omega = 3$ släpps igenom.



Figur 15: Även här släpps $\omega = 3$ igenom som brukligt.



```
%% Uppgift 3.1 a)

% An = 0
% Bn = 4/(k*pi)
% Se fk.tex

% Uppgift 3.1 b)
clf
clc
syms n x k

T=1;
w=2*pi/T;
M=200;
x=0;
t=T*(0:M-1)/M;

for n=1:100
    Ak = 0;
    Bk = 4*mod(n,2)/(n*pi);
    x = x + Ak*cos(n*w*t) + Bk*sin(n*w*t);
end
plot(t,x)

%% Uppgift 3.2 a)
clf
clc

% y(t) = g(w)sin(wt+phi(w))
% num  = (s^2+10.1s+1)
% den  = (s^3+2*s^2+10*s+9)

num = [1 10.1 1];
den = [1 2 10 9];
Gs=tf(num, den);

% Uppgift 3.2 b)
F=100;
N=2.^13;
Ts=1/F;
Tmax = (N-1)*Ts;
t=0:Ts:Tmax;

% Insignals
x1 = sin(t);
x2 = sin(3*t);
x3 = sin(5*t);

%% Plots
% subplot(3,1,1)
% plot(t, x1)
% legend('sin(t)')
% axis([0 80 -1 1])
% subplot(3,1,2)
% plot(t, x2)
% legend('sin(3t)')
% axis([0 80 -1 1])
% subplot(3,1,3)
% plot(t, x3)
```

```
% legend('sin(5t)')
% axis([0 80 -1 1])

% Uppgift 3.2 c)

% y = x sent through sys using lsim
y1 = lsim(Gs,x1,t);
y2 = lsim(Gs,x2,t);
y3 = lsim(Gs,x3,t);

% y = correct amplitude, wrong phase
y1e = abs(evalfr(Gs,1j))*x1;
y2e = abs(evalfr(Gs,3j))*x2;
y3e = abs(evalfr(Gs,5j))*x3;

% y = correct amp and phase due to eq 2
phi1 = angle(evalfr(Gs,1j));
x1p = sin(t+phi1);
y1p = abs(evalfr(Gs,1j))*x1p;

phi2 = angle(evalfr(Gs,3j));
x2p = sin(3*t+phi2);
y2p = abs(evalfr(Gs,3j))*x2p;

phi3 = angle(evalfr(Gs,5j));
x3p = sin(5*t+phi3);
y3p = abs(evalfr(Gs,5j))*x3p;

%%% Plots
%% a)
% bode(sys)
% pzmap(sys)

% c)
subplot(3,1,1)
plot(t,y1,'--b', t,y1p,':r', t, x1,'k')
axis([0 30 -1 1.2])
legend('lsim1', 'ekv 2', 'x1'), title('w=1')

subplot(3,1,2)
plot(t,y2, '--b', t, y2p, ':r',t, x2,'k')
axis([0 30 -5 5])
legend('lsim2', 'ekv 2','x2'), title('w=3')

subplot(3,1,3)
plot(t,y3, '--b', t, y3p, ':r',t,x3,'k')
axis([0 30 -1.3 1.2])
legend('lsim3', 'ekv 2','x3'), title('w=5')

%% Uppgift 3.3 a)
clf
clc

N = 2.^13;
F = 100;
Ts = 1/F;
t = 0:Ts:40*pi;
x = square(t);
% x(t) != x(-t) => x ar udda
```

```
% Uppgift 3.3 b)
fprintf('3.3 b\n')
fprintf('FK enligt ekv 1:\n\n')
for n = 1:5;
    An = 0;
    Bn = 4*mod(n,2)/(n*pi);
    if(Bn)
        disp(Bn)
    end
end

% Uppgift 3.3 c)
k = 0:(N-1);
wk = (2*pi*F*k)/(N);
kf=@(wk) (N*wk)/(2*pi*F);
ffx=fft(x, N);

% Uppgift 3.3 d)
B1 = (2*abs(ffx(k+1)))/N;
fprintf('3.3 d\n')
fprintf('FK enligt fft (ekv 10):\n\n')
B1max1 = max((B1(1:ceil(kf(2)))));
B1max2 = max((B1(ceil(kf(2)):ceil(kf(4)))));
B1max3 = max((B1(ceil(kf(4)):ceil(kf(6)))));
disp(B1max1)
disp(B1max2)
disp(B1max3)

% Uppgift 3.3 e)
num = [1 10.1 1];
den = [1 2 10 9];
H=tf(num, den);

y = lsim(H,x,t);

%plot(t, y);

% Enligt ekv 8
fprintf('3.3 e\n')
fprintf('FK enligt ekv 8:\n\n')
disp(abs(evalfr(H,1j))*4/(pi*1))
disp(abs(evalfr(H,3j))*4/(pi*3))
disp(abs(evalfr(H,5j))*4/(pi*5))

% Enligt fft
ffy = fft(y, N);
B2 = (2*abs(ffy(k+1)))/N;

fprintf('FK enligt fft (ekv 10):\n\n')
B2max1 = max((B2(1:ceil(kf(2)))));
B2max2 = max((B2(ceil(kf(2)):ceil(kf(4)))));
B2max3 = max((B2(ceil(kf(4)):ceil(kf(6)))));
disp(B2max1)
disp(B2max2)
disp(B2max3)

% Plots
subplot(3,1,1)
plot(t, x, 'r', 'linewidth', 1.5)
axis([-0 10 -1.5 1.5])
```

```
grid on
subplot(3,1,2)
plot(wk, abs(ffx))
axis([0 6 0 6000])
grid on
subplot(3,1,3)
plot(wk, abs(B2))
axis([0 6 -0 1.5])
grid on

%% Uppgift 3.4 a)
clc
clf

syms s;

Ts = s*(s-1j)*(s+1j)*(s-5j)*(s+5j)*(s-7j)*(s+7j)*(s-9j)*(s+9j);
num = [1 0 156 0 7374 0 106444 0 99225 0];
Tp = tf(num, 1);
% % Plots
%bode(Tp);
%grid on

% Uppgift 3.4 b: n=10, c: n=11)
Np = 1;
for n = 1:12
    Np = Np*(s+4);
end
den = sym2poly(Np);
sys = tf(num,den);
w={1,5000};
% % Plots
%bode(sys,w);
%grid on

% Uppgift 3.4 d)
scale = abs(evalfr(sys, 3j));
sys2 = tf(num/scale, den);
abs(evalfr(sys2,3j));
% % Plots
% bode(sys2, 'r');
% grid on
% hold on
% bode(sys, '--b');

% Uppgift 3.4 e)
N=8192;
k = 0:(N-1);
wk = (2*pi*F*k)/(N);
F = 100;
Ts = 1/F;
t = 0:Ts:(N-1)*Ts;
x = square(t);

% x=square(t)
yx = lsim(sys2,x,t);
ffy = fft(yx, 8192);
By = (2*abs(ffy(k+1)))/N;
% % Plots
% plot(t,x, 'k', t, yx, 'b');
```

```
% legend('x(t)', 'y(t)')
% axis([0 30 -1.5 1.5])
% plot(wk, abs(By));
% axis([0 20 0 .5])


%% Amplitud hos Notchfiltrets utsignal
% fprintf('Notch-amp square:')
% disp(max(yx))
%% Amplitud genom DTF och FFT
% fprintf('FFT-amp square:')
% disp(max(By))

% ysignalen
num = [1 10.1 1];
den = [1 2 10 9];
H=tf(num, den);
y = lsim(H,x,t);
yy = lsim(sys2, y,t);
ffy = fft(yy, 8192);
By = (2*abs(ffy(k+1)))/N;
%% Plots
% plot(t,y, 'k', t, yy);
% legend('x(t)', 'y(t)')
% axis([0 30 -3 3])
plot(wk, abs(By));
axis([0 20 0 1.5])

%% Amplitud hos Notchfiltrets utsignal
% fprintf('Notch-amp yy:')
% disp(max(yy))
%% Amplitud genom DTF och FFT
% fprintf('FFT-amp yy:')
```