PART III: Design of various kinds of digital filters meeting the same criteria

- This part shows how the same filter criteria can be met by various kinds of digital filters.
- Futhermore, it is shown that after proper reasoning we are able to end up filters with a drastically reduced complexity.

PART III: Design of various kinds of digital filters meeting the same criteria

- This part shows how the same filter criteria can be met by various kinds of digital filters.
- Futhermore, it is shown that after proper reasoning we are able to end up filters with a drastically reduced complexity.

FISH RESEARCH

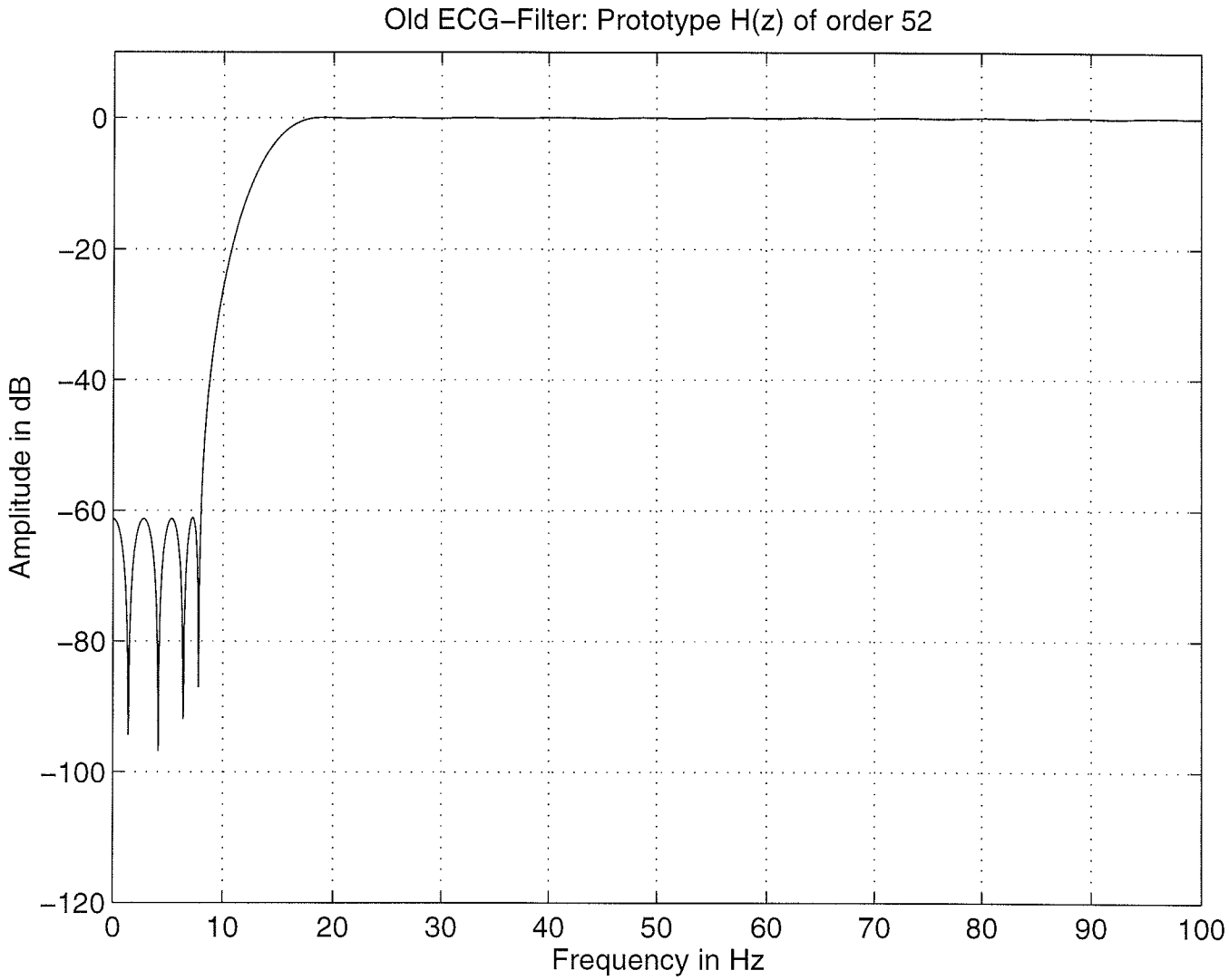
- In the end of this pile of lecture notes there is an article which considers the use of the brown trout to test the quality of the surface raw water.
- In this article, several filters have been designed.
- Here, we concentrate on the design of the ECG-filter and the total activity filter.
- Matlab-files, called fishi1.m and fishi2.m, used for designing these filters can also be found in the end of these lecture notes.

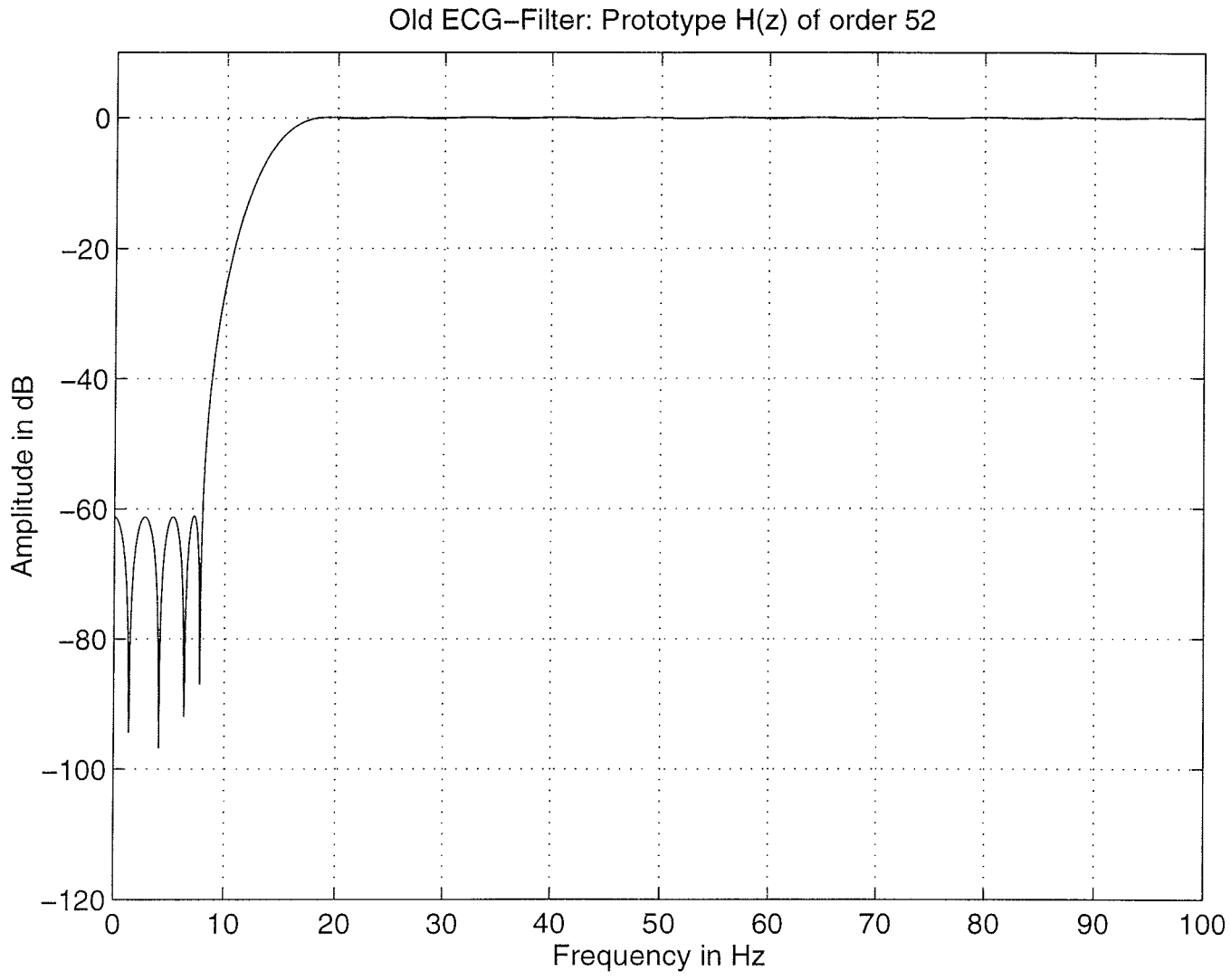
ECG FILTER: OLD DESIGN

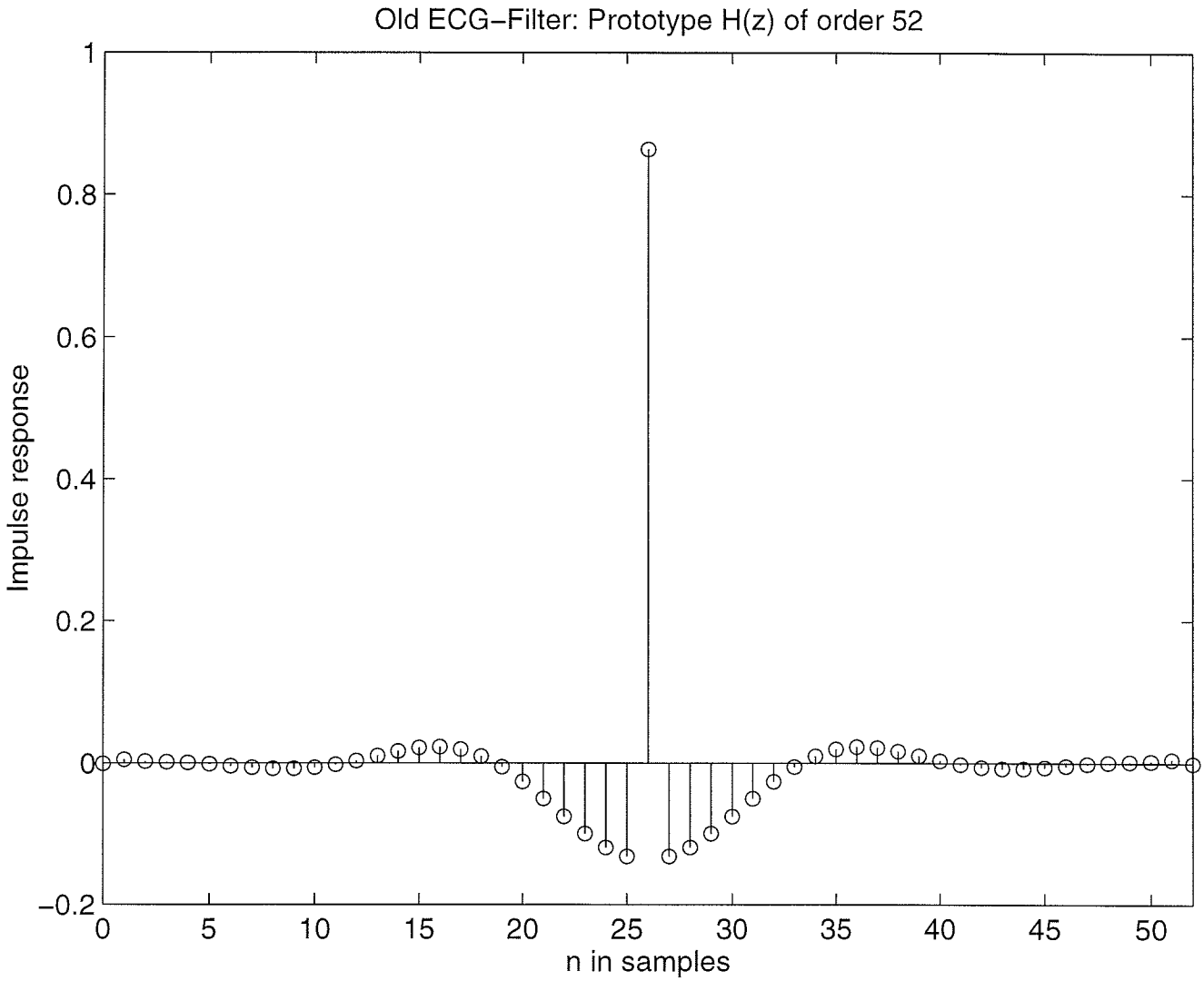
- It is desired to design for $f_s = 200$ Hz a filter having the stopband regions from 0 Hz to 2 Hz, from 48 Hz to 52 Hz, and from 98 Hz to 100Hz. The passband regions are from 4.5 Hz to 45.5 Hz and from 54.5 Hz to 95.5 Hz. The passband ripple is $\delta_p = 0.01$ and the stopband ripple is $\delta_s = 0.001$.
- In the article in the end of this pile, they first design a prototype highpass filter $H(z)$ for $f_s = 200$ Hz in such a way that the stopband region is from 0 Hz to $4 \cdot 2$ =8 Hz and the passband region is from $4 \cdot 4.5$ =18 Hz to 100 Hz. In the ω-scale, the stopband edge is 0.08π and the passband edge is 0.18π. Using the Remez algorithm (see Matlab-file fishi1.m), the minimum filter order is 52.
- Because of the periodicity of the overall specifications, the desired overall filter is then automatically $H(z^4)$, that is, a transfer function obtainable from $H(z)$ by using instead of a unit delay z^{-1} a delay block z^{-4}. The implementation of this filter is very

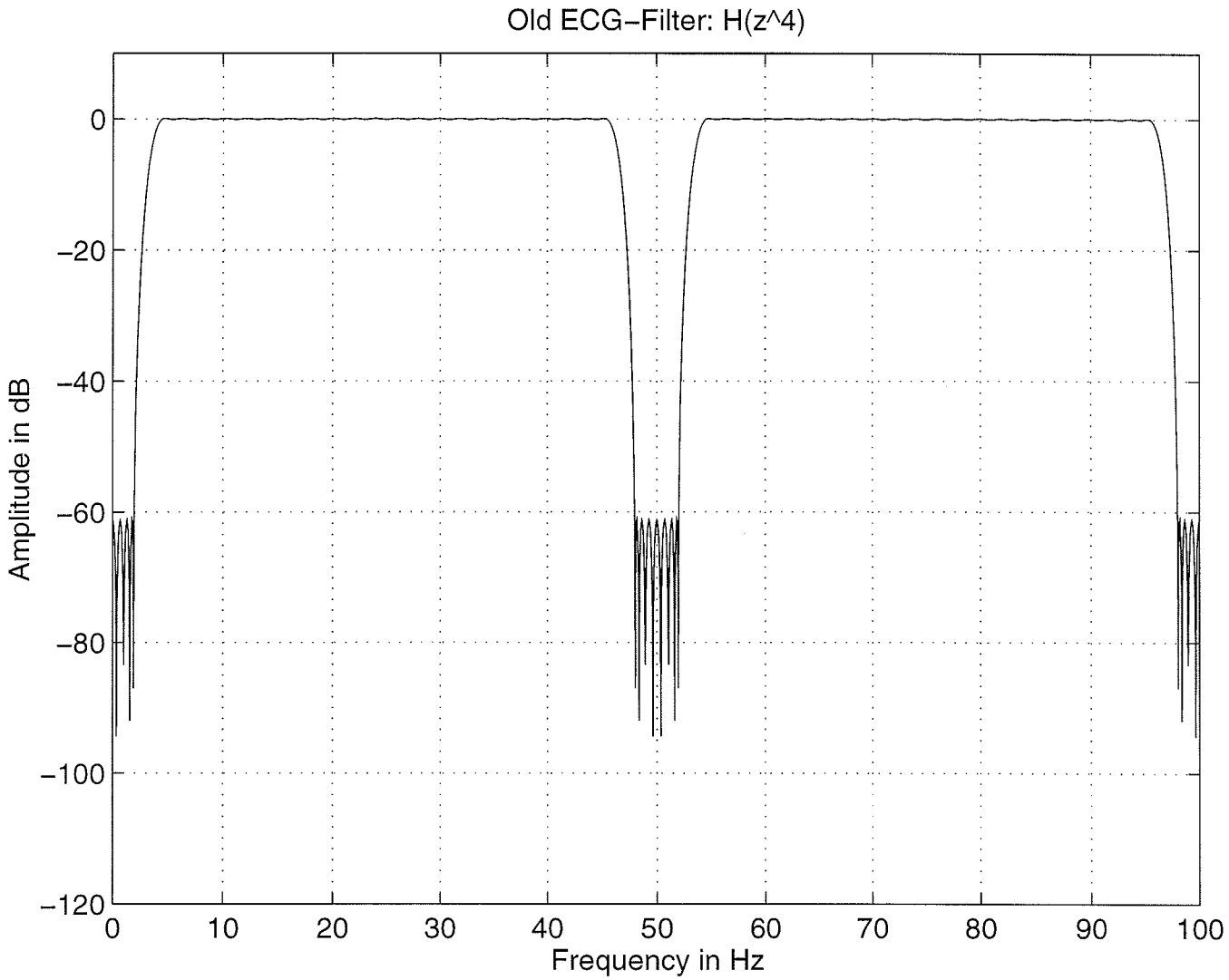
straightforward with the aid of some signal processors, where during the same instruction cycle it is possible to delay the signal by four unit samples instead of one unit sample.

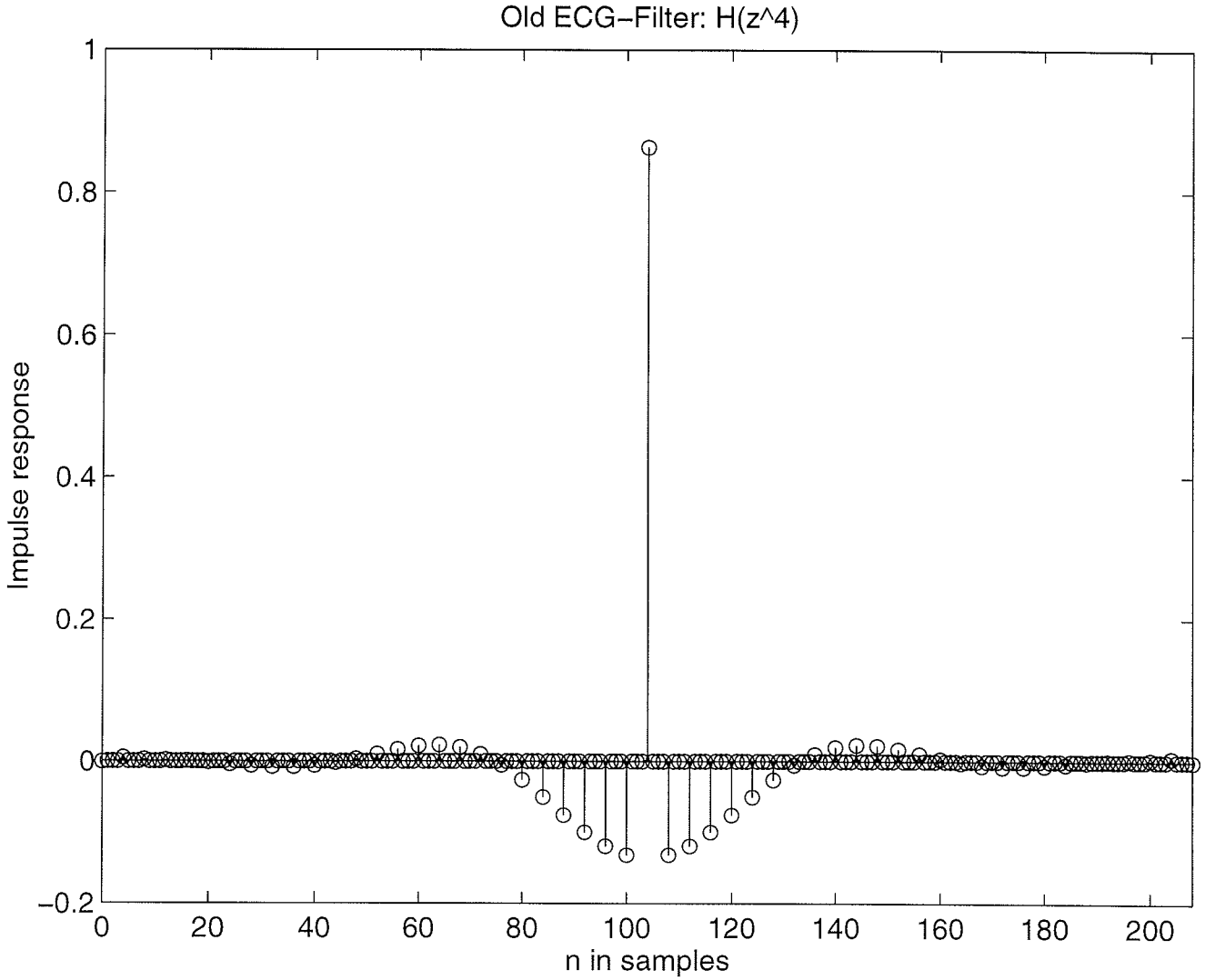
- Note that by replacing z^{-1} by z^{-4} the frequency axis is shrunk by by a factor of four and 400 Hz is mapped to 100 Hz so that there are several periods of the prototype filter in the frequency band of interest, that is, in the region from 0 Hz to 100 Hz.
- The number of multipliers required by this design is approximately one fourth compared to the conventional direct-form FIR filter whose order is approximately $4 \cdot 52 = 208$. Therefore, the code length is drastically reduced.
- In the following there are four transparencies illustrating the performance of this design.


Old ECG–Filter: Prototype H(z) of order 52
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Old ECG–Filter: Prototype H(z) of order 52
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Old ECG−Filter: Prototype H(z) of order 52
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
5
10
15
20
25
30
35
40
45
50

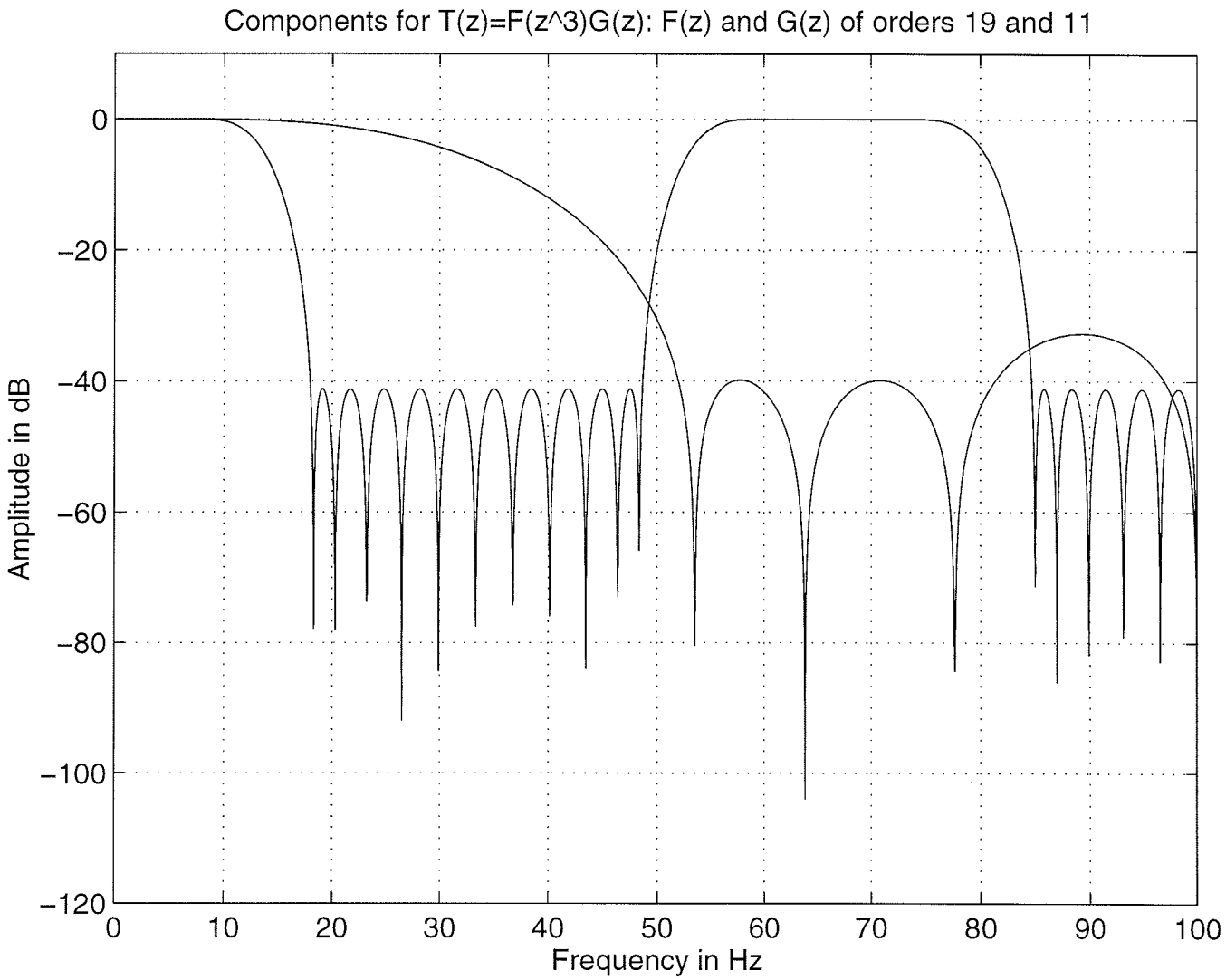

Old ECG–Filter: H(z^4)
Amplitude in dB
0
–20
–40
–60
–80
–100
–120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz

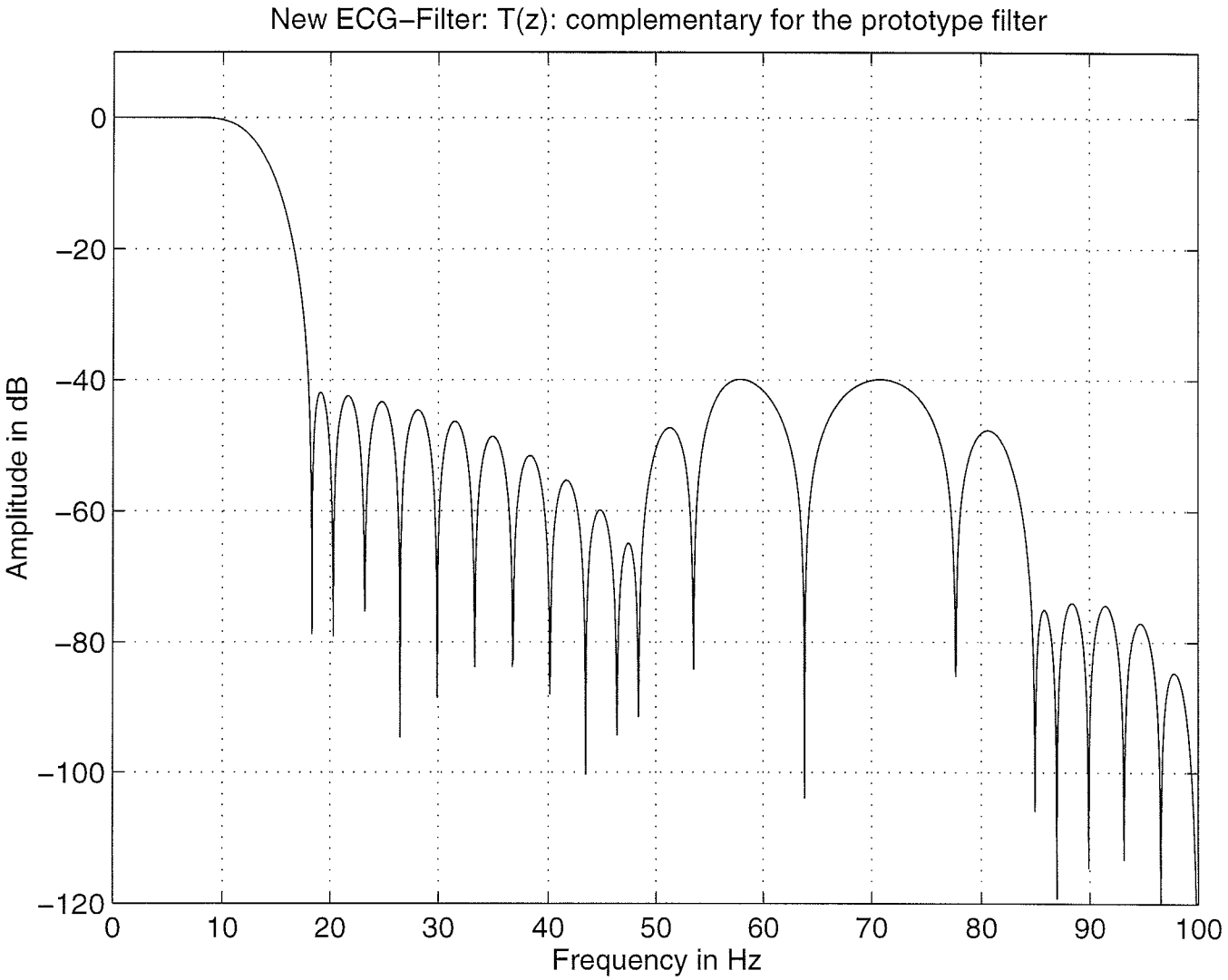

Old ECG–Filter: H(z^4)
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
20
40
60
80
100
120
140
160
180
200

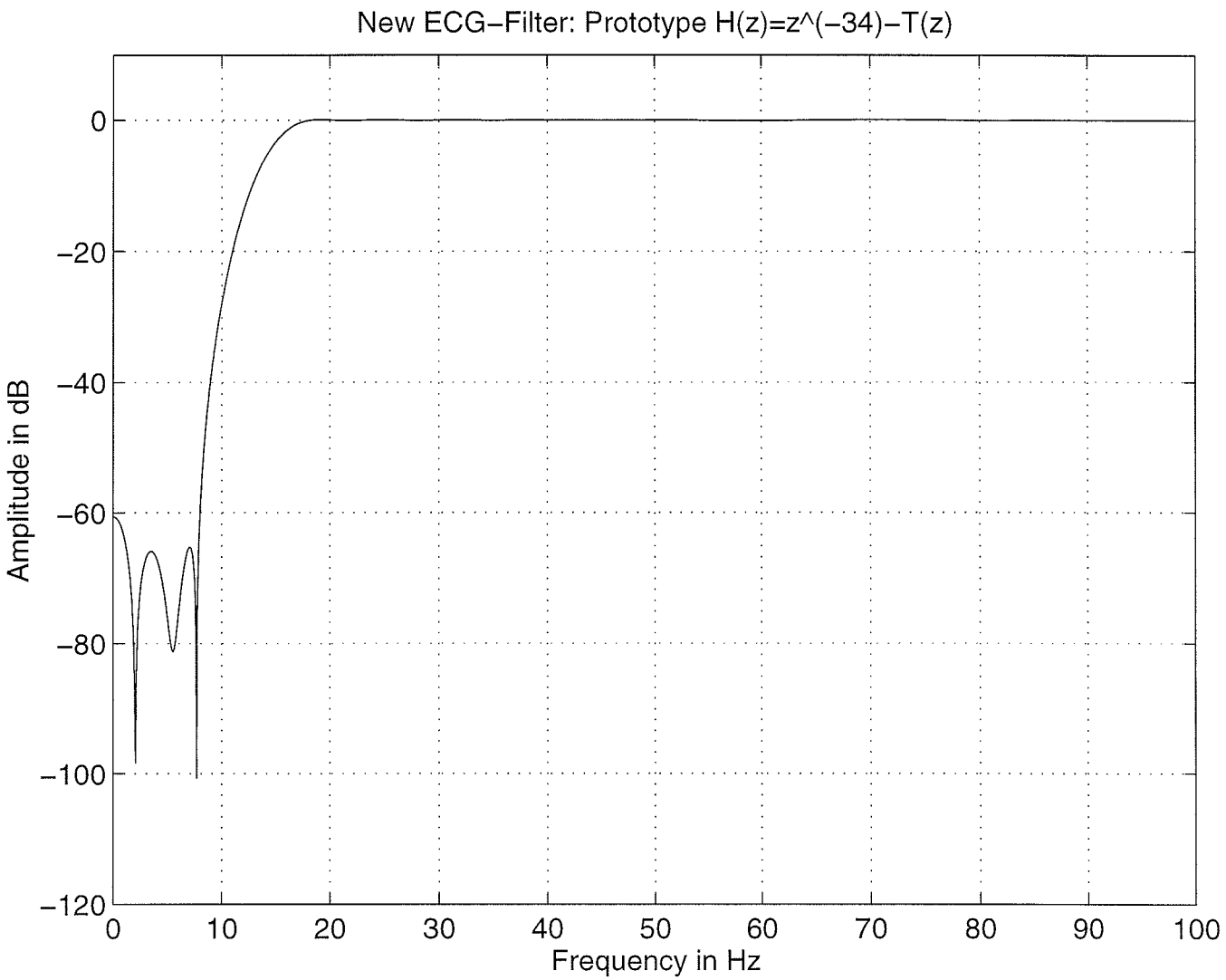
ECG FILTER: NEW DESIGN

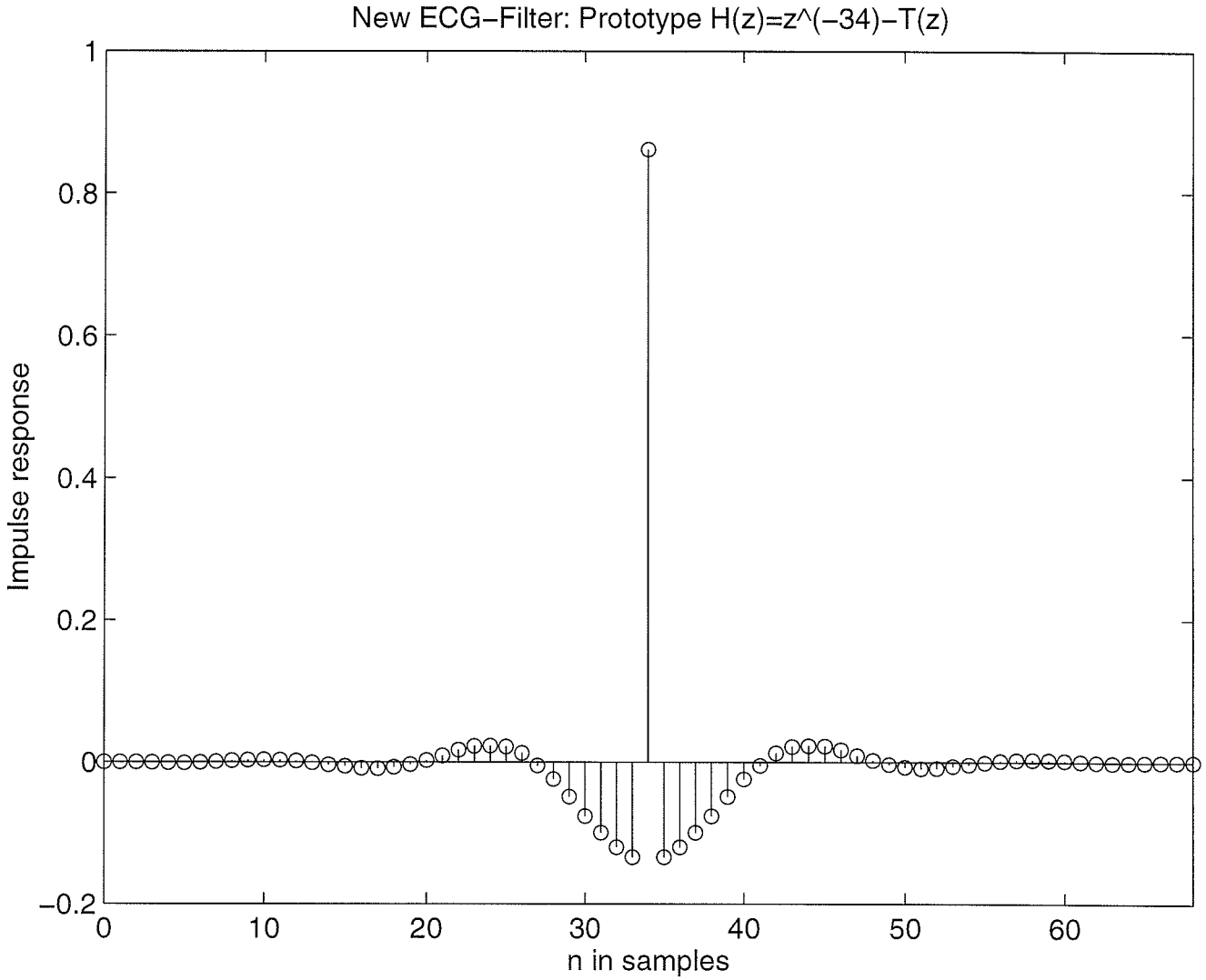
- Here we design the protype filter in the form $H(z) = z^{-M} - T(z)$, $T(z) = F(z^3)G(z)$, where M is half the order of $T(z)$. The complementary filter $T(z)$ is a lowpass filter with the passband and stopband edges in the ω-scale at $\omega_p = 0.08\pi$ and $\omega_s = 0.18\pi$. The passband and stopband ripples are $\delta_p = 0.001$ and $\delta_s = 0.01$.
- The desired overall filter $T(z)$ is obtained by designing $F(z)$ in such a way that its passband and stopband edges are located at $3\omega_p$ and $3\omega_s$. The passband and stopband ripples are $\delta_p/2$ and δ_s. For $G(z)$ the ripples are the same. The passband region is $[0, \omega_p]$, whereas the stopband region is $[\omega_{s1}, \omega_{s2}]$ with $\omega_{s1}, \omega_{s2} = 2\pi/3 \pm (2\omega_s + \omega_p)/3$.
- The given criteria are met by $F(z)$ of order 19 and $G(z)$ of order 11. The overall order of $T(z)$ is thus $3 \cdot 19 + 11 = 68$ so that $M = 34$.
- The prototype filter is thus $H(z) = z^{-34} - F(z^3)G(z)$ and the overall filter $H(z^4) = z^{-136} - F(z^{12})G(z^4)$.

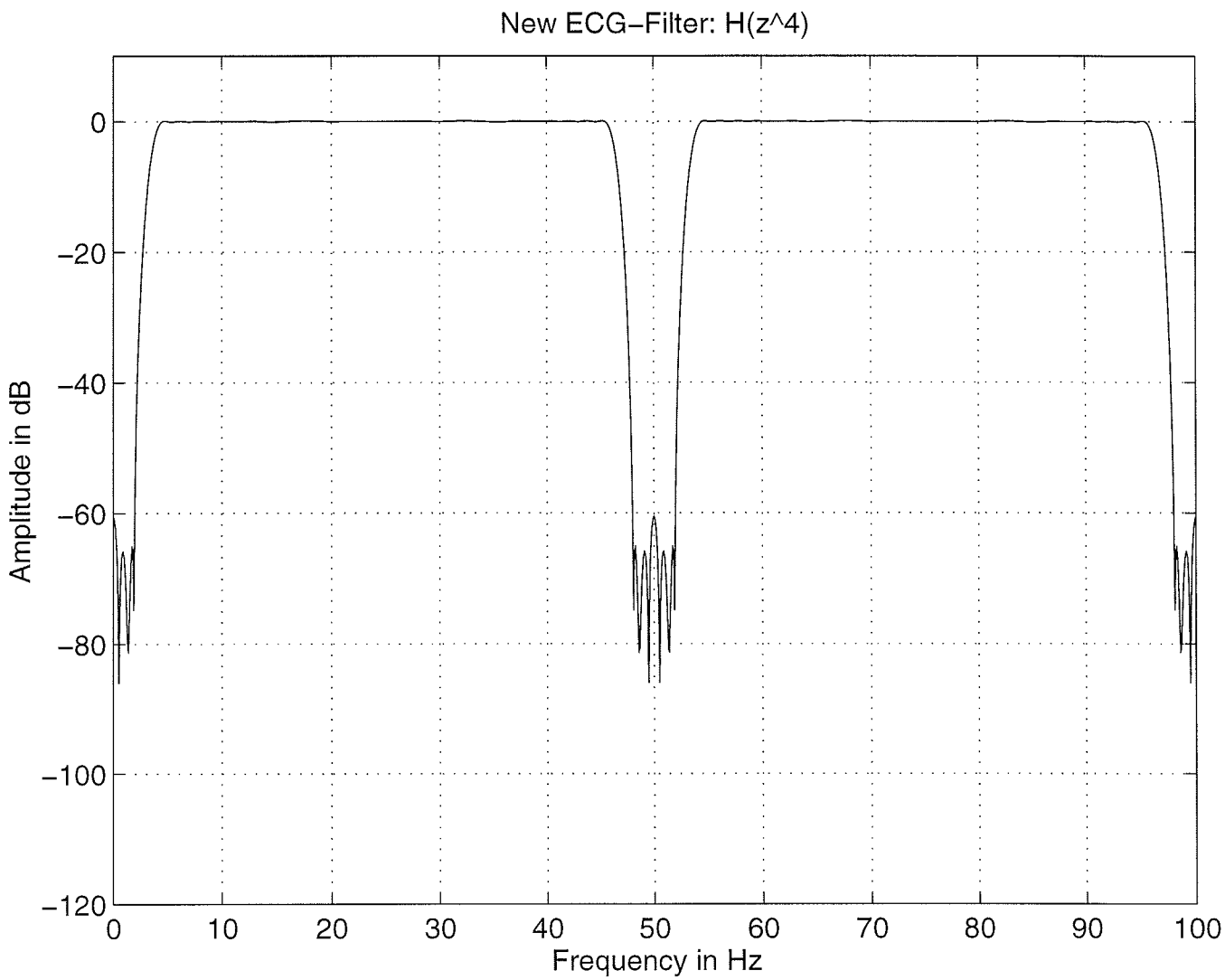
- In practical implementation using a signal processor, the delay z^{-136} can be shared with $F(z^{12})$.
- In normal signal processor implementations, the coefficient symmetry is not worth exploiting. The old design requires $52 + 1 = 53$ multipliers, whereas the new design requires $19 + 1 + 11 + 1 = 32$ multipliers at the expense of the increase in the overall filter order (from 208 to 272).
- In the following there are six transparencies illustrating the performance of this design.

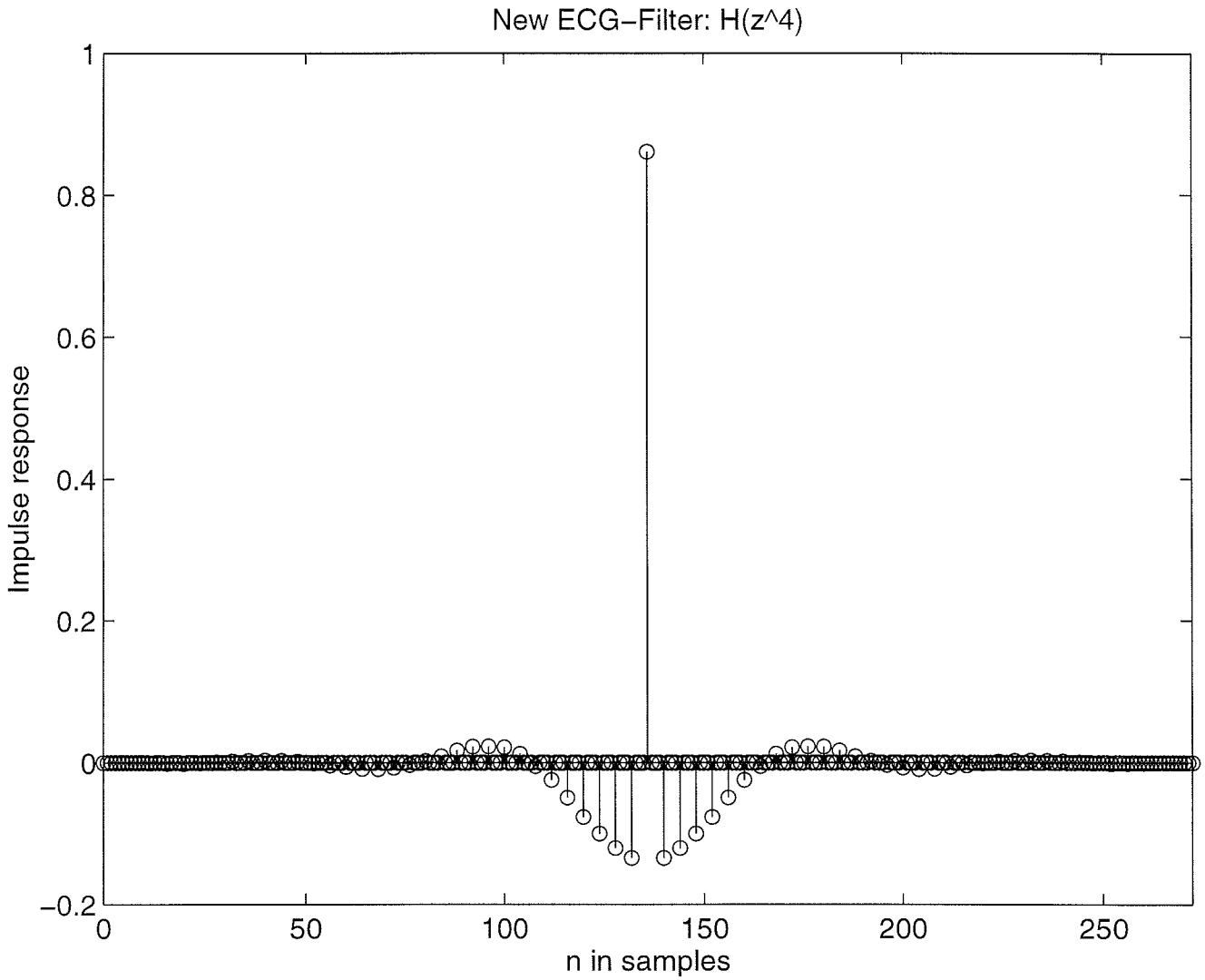

Components for T(z)=F(z^3)G(z): F(z) and G(z) of orders 19 and 11
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


New ECG–Filter: T(z): complementary for the prototype filter
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


New ECG–Filter: Prototype H(z)=z^(–34)–T(z)
Amplitude in dB
0
–20
–40
–60
–80
–100
–120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz

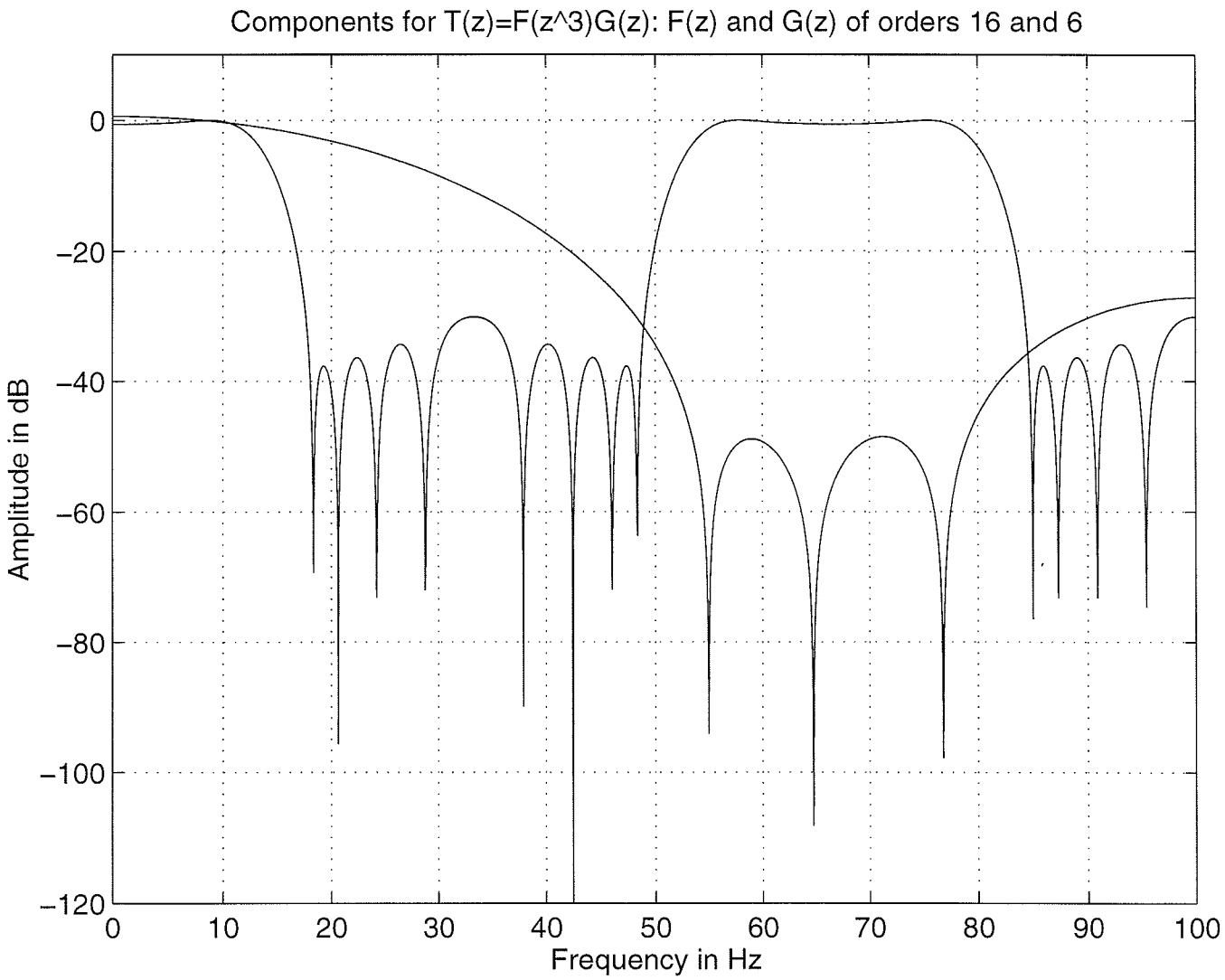

New ECG−Filter: Prototype H(z)=z^(−34)−T(z)
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
10
20
30
40
50
60

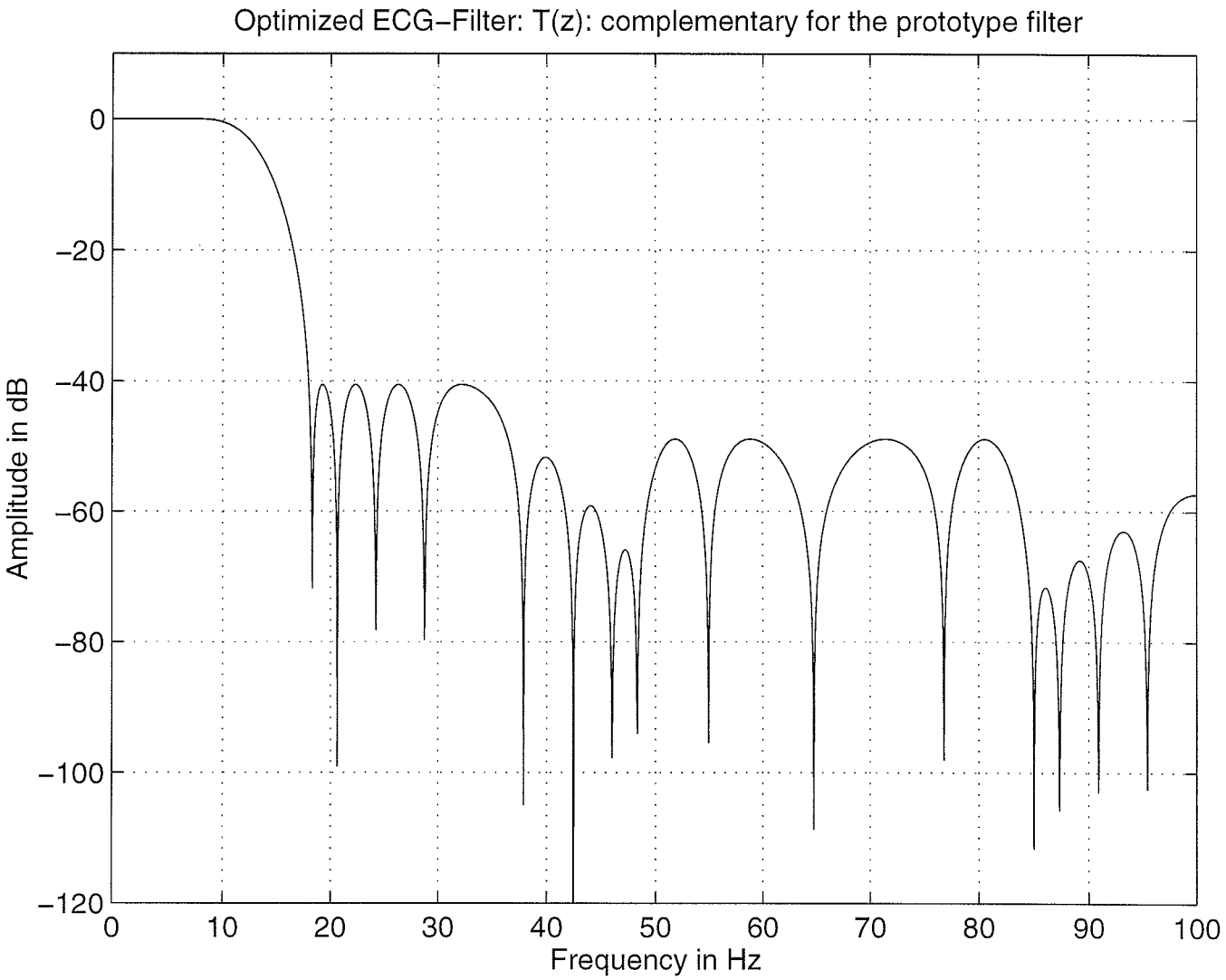

New ECG–Filter: H(z^4)
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz

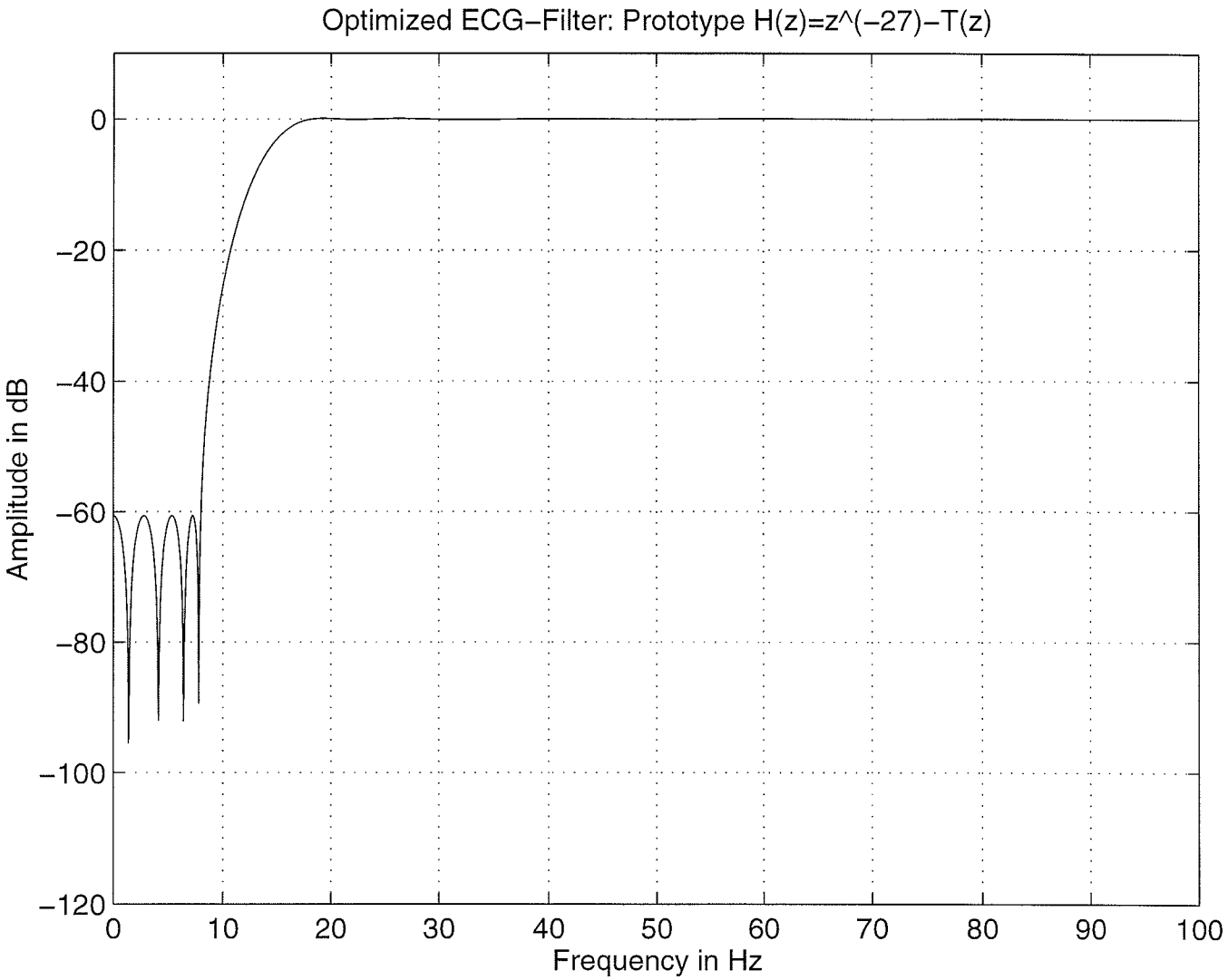

New ECG–Filter: H(z^4)
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
50
100
150
200
250

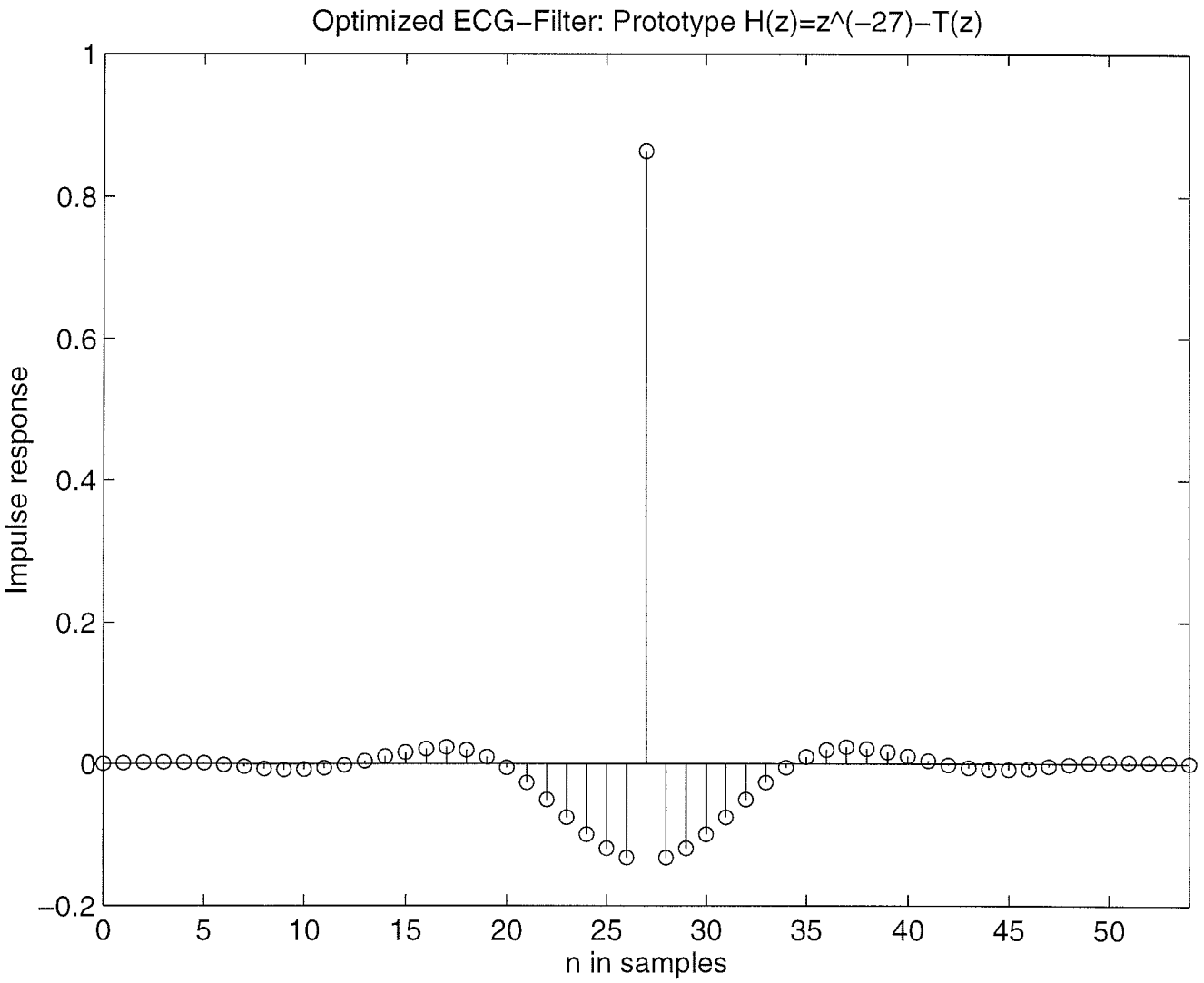
ECG FILTER: OPTIMIZED DESIGN

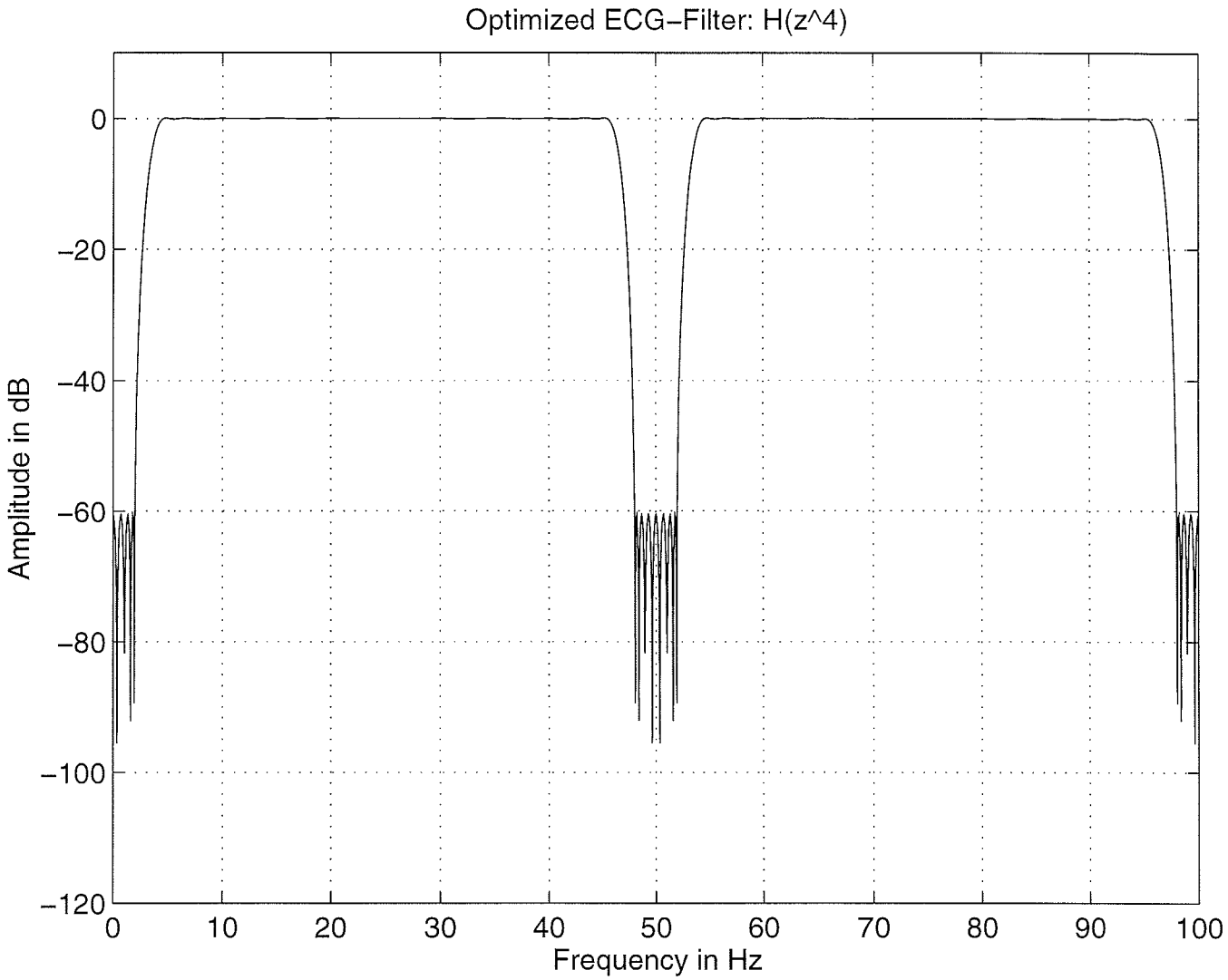
- Here, we design $F(z^3)$ and $G(z)$ using the algorithm described in T. Saramäki, "Finite impulse response Filter Design" in Handbook for Digital Signal Processing, S. K. Mitra and J. F. Kaiser, Eds., John Wiley & Sons, 1993, pp. 241-245.
- Using this algorithm, the orders of $F(z)$ and $G(z)$ reduce from 19 to 16 and from 11 to 6.
- In this case, $H(z^4) = z^{-108} - F(z^{12})G(z^4)$. The overall order is 216, which is only slightly larger than that (208) of the old design. The number of multipliers reduces to $16 + 1 + 6 + 1 = 24$, which is less than half that (53) required by the old design.
- In the following there are six transparencies illustrating the performance of this design.

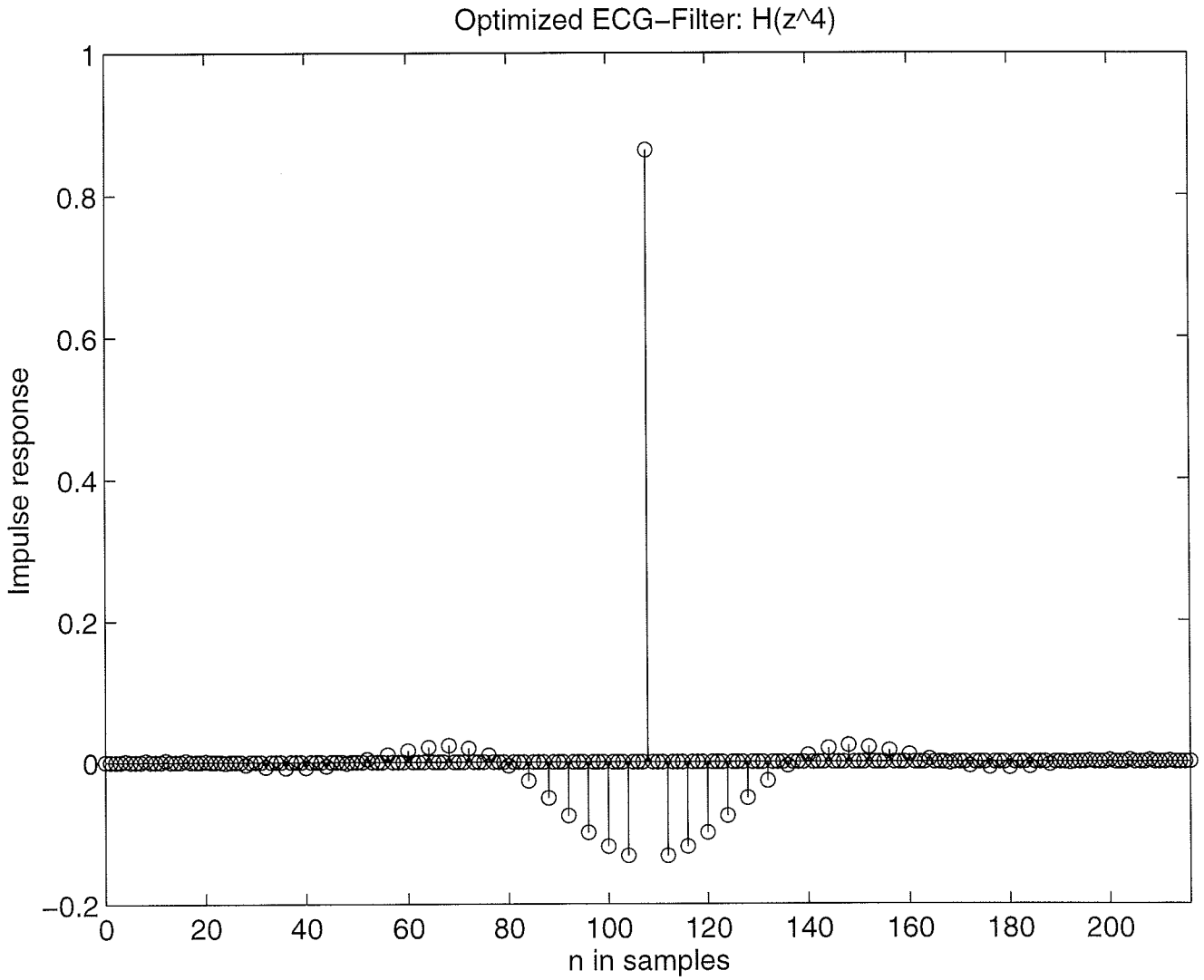

Components for T(z)=F(z^3)G(z): F(z) and G(z) of orders 16 and 6
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Optimized ECG–Filter: T(z): complementary for the prototype filter
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Optimized ECG–Filter: Prototype H(z)=z^(–27)–T(z)
Amplitude in dB
0
–20
–40
–60
–80
–100
–120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Optimized ECG−Filter: Prototype H(z)=z^(−27)−T(z)
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
5
10
15
20
25
30
35
40
45
50


Optimized ECG–Filter: H(z^4)
Amplitude in dB
0
−20
−40
−60
−80
−100
−120
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz

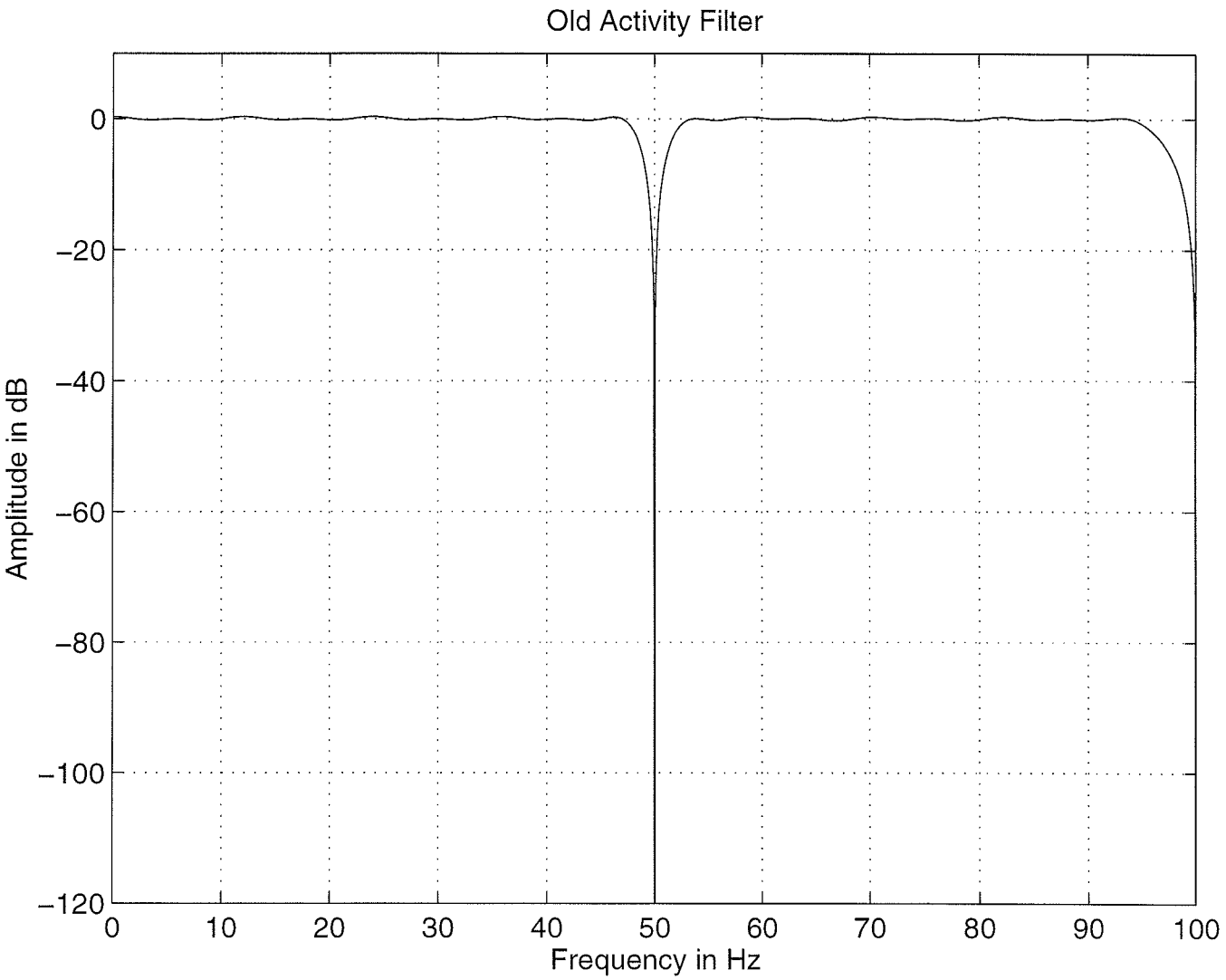

Optimized ECG–Filter: H(z^4)
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
20
40
60
80
100
120
140
160
180
200

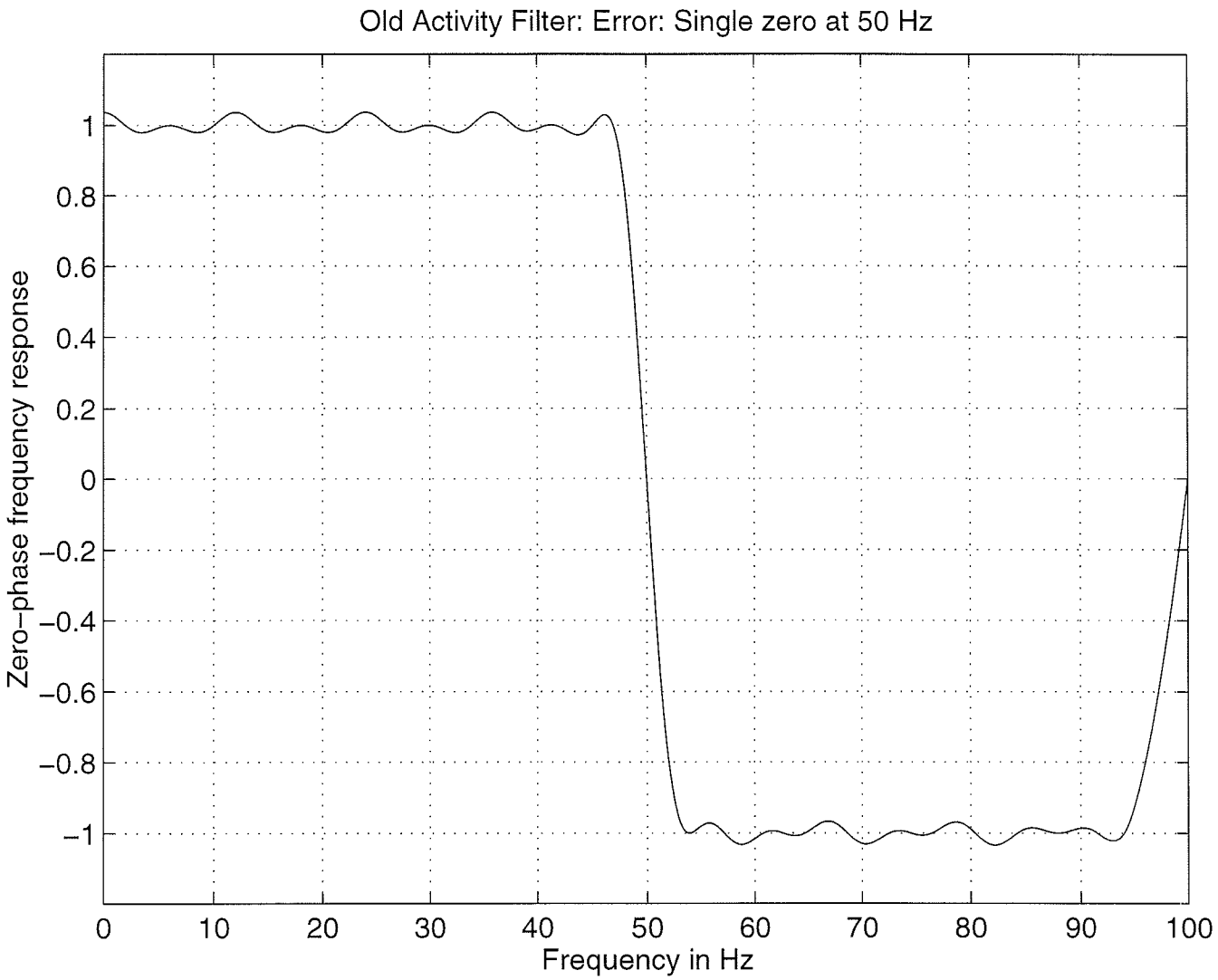
ACTIVITY FILTER: OLD DESIGN

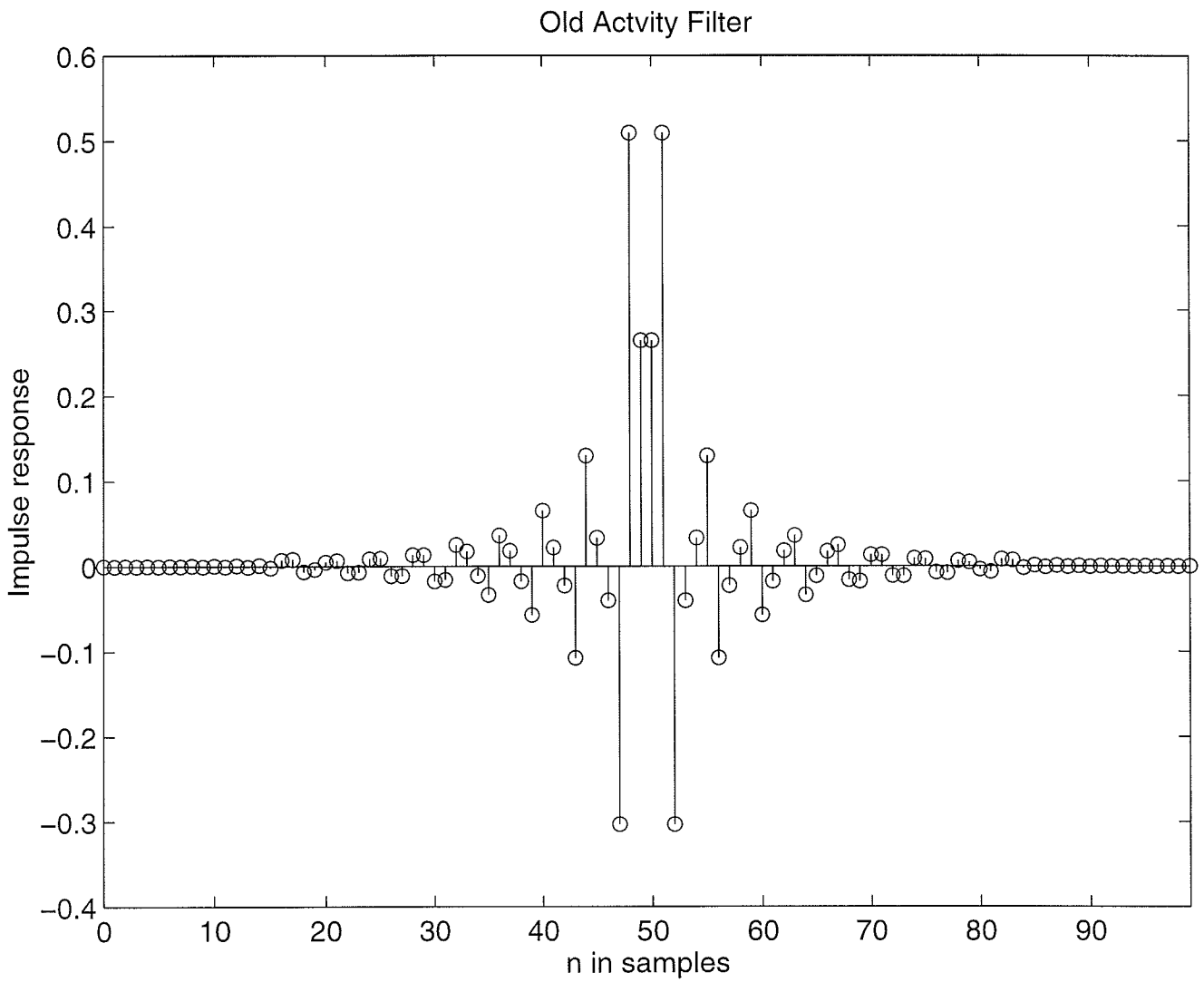
- It is desired to design for $f_s = 200$ Hz a filter $H(z)$ having zeros at 50 Hz and at 100 Hz to remove the line frequency interference. The passband regions are from 0 Hz to 47 Hz and from 53 Hz to 94 Hz. The passband ripple is $\delta_p = 0.01$.
- In the article in the end of this pile, they first design for $f_s = 200$ a filter $G(z)$ of order 33 such that the passband region is from 0 Hz to 94 Hz. For the Remez-algorithm, there is only one passband $[0, 0.94\pi]$ and the desired and weighting functions are unity (see Matlab-file fishi2.m). Because of an odd order, there is automatically a zero at $z = -1$ (at $\omega = \pi$ or at 100 Hz).
- The periodic filter $G(z^2)$ has then for $f_s = 200$ a zero at 50 Hz and passband regions from 0 Hz to 47 Hz and from 53 Hz to 100 Hz.
- Filter $H(z) = G(z^2)G(z)$ has the zeros at the right positions and the desired passband regions.
- In the following there are three transparencies illus-

trating the performance of this design.

- The overall design suffers, however, from a serious problem. As seen from the figure giving the zero-phase frequency response of the overall design, the zero-phase response changes its sign at 50 Hz.
- This means that the phase response has a jump of π at 50 Hz. Because of this jump, the phase delay is not constant in the interval from 0 Hz to 100 Hz. Therefore, the filter is useless in the cases where the waveforms are desired to be preserved.


Old Activity Filter
0
−20
−40
−60
−80
−100
−120
Amplitude in dB
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Old Activity Filter: Error: Single zero at 50 Hz
Zero−phase frequency response
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Old Actvity Filter
Impulse response
n in samples
0.6
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
0
10
20
30
40
50
60
70
80
90

ACTIVITY FILTER: NEW DESIGN

- The filter is designed in the following steps:

1) For $f_s = 200$ Hz, we design a filter $T(z)$ such that it achieves the value of unity at 0 Hz and its attenuation is at least 40 dB ($\delta_s = 0.01$) in the range from $4 \cdot 3 = 12$ Hz to 100 Hz. In the ω-scale, $\omega_s = 0.12\pi$. Our filter transfer function is of the form

$$T(z) = F(z^4)[RRS4(z)]^2,$$

where

$$RRS4(z) = 1/4 + 1/4z^{-1} + 1/4z^{-2} + 1/4z^{-3}.$$

For $F(z)$, the stopband edge is $4\omega_s = 0.48\pi$. The given criteria are met by $F(z)$ of order 7.

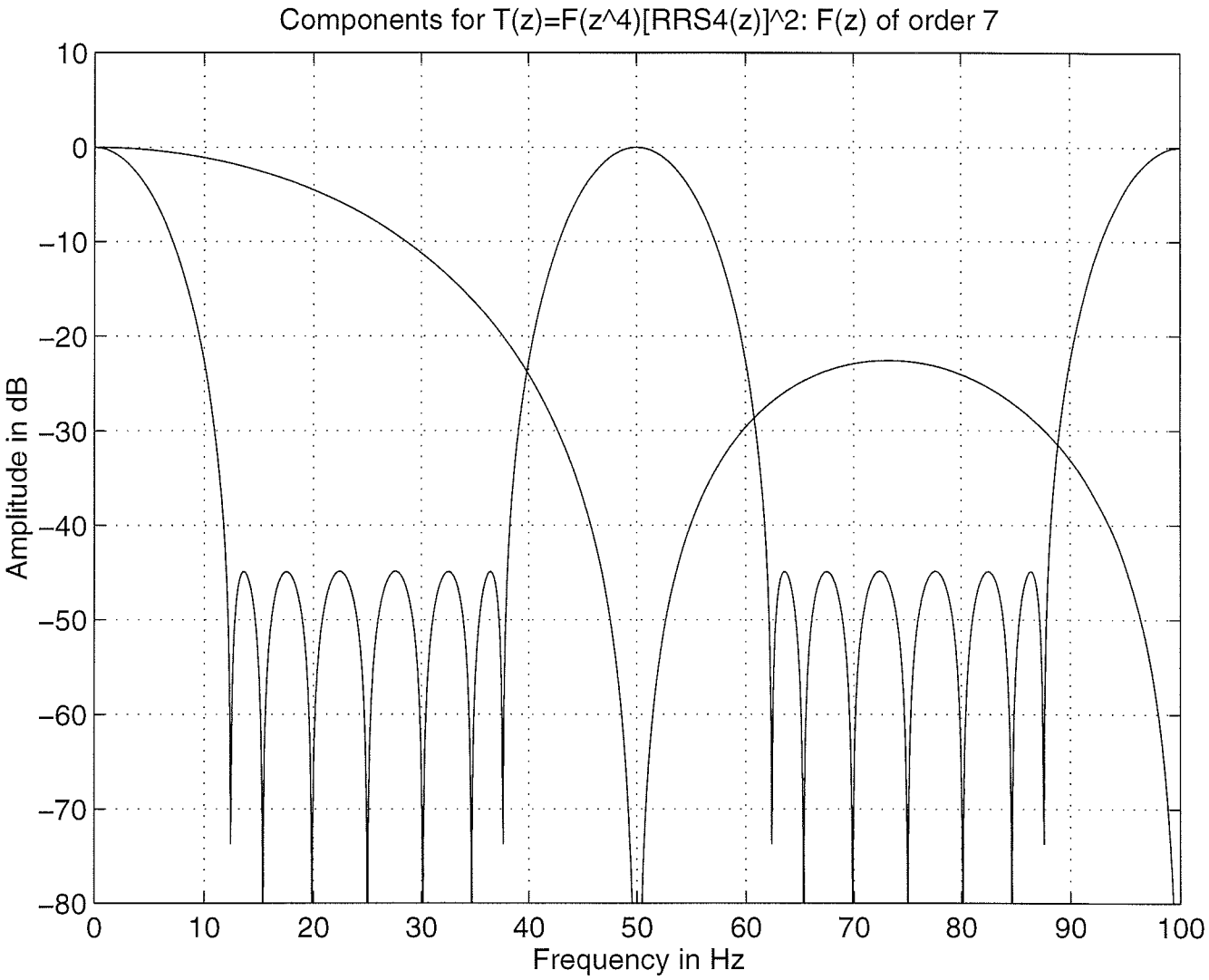
2) $T(z^4)$ is periodic achieving the value of unity for $f_s = 100$ Hz at 0, 50, and 100 Hz. The stopband regions are from 0 Hz to 47 Hz and from 53 Hz to 97 Hz.

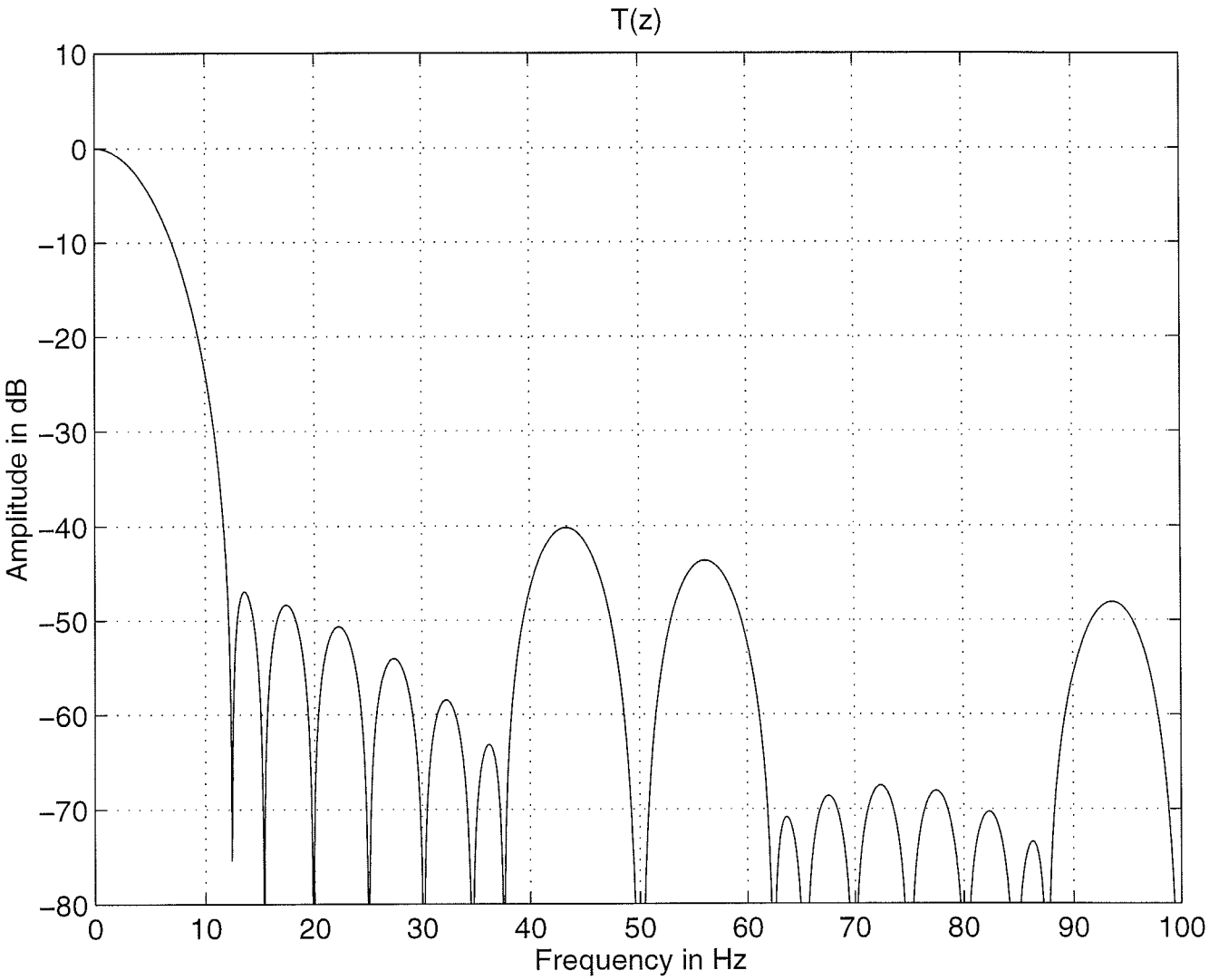
3) Cascade $T(z^4)$ with $z^{-3} - [RRS4(z)]^2$, giving $S(z) = T(z^4)[z^{-3} - [RRS4(z)]^2]$. Here, $z^{-3} - [RRS4(z)]^2$ attenuates the first passband region of $T(z^4)$ while

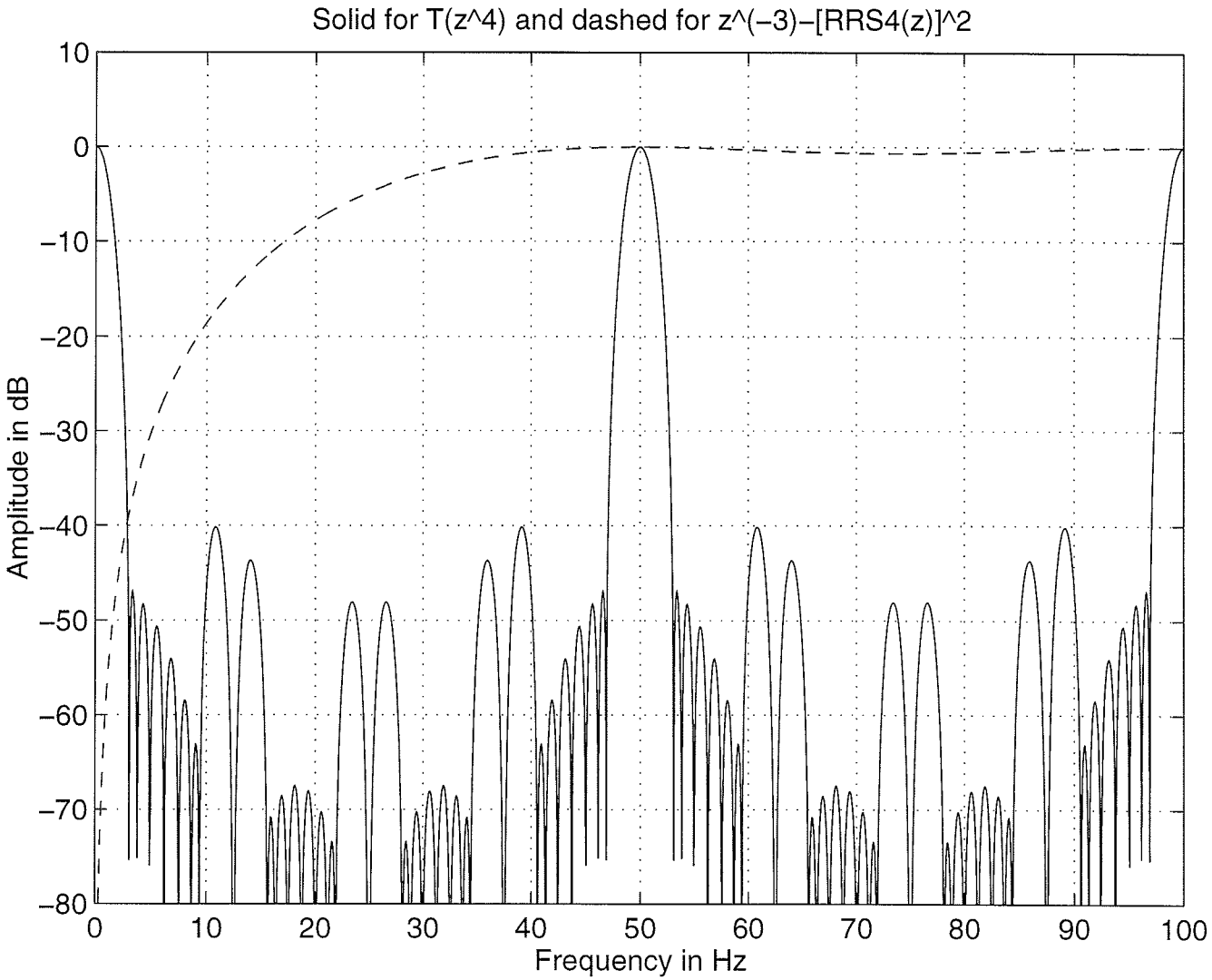
still keeping the value of unity at 50 and 100 Hz ($z^{-3} - [RRS4(z)]^2$ achieves the value of unity at these points.)

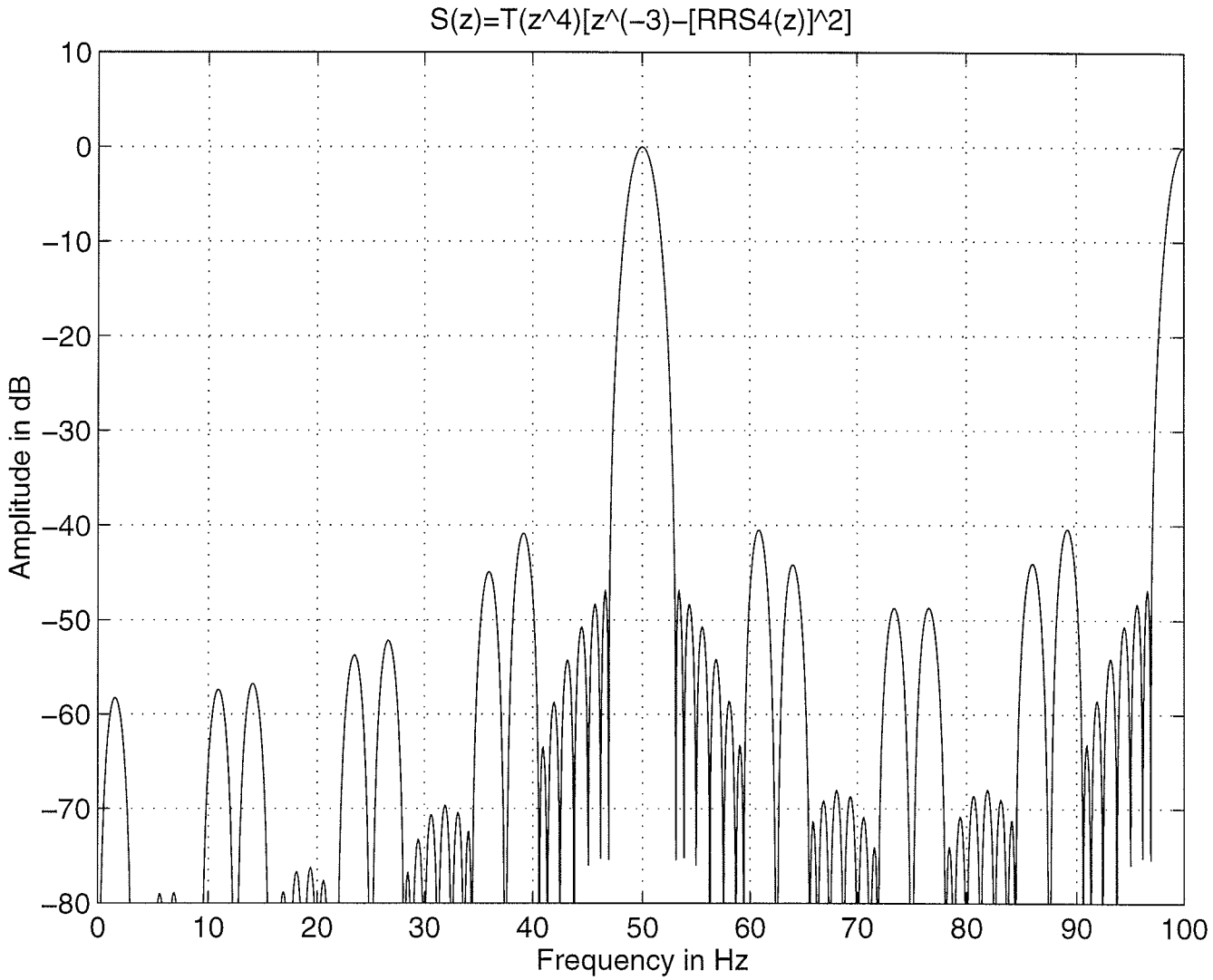
4) The desired overall filter is then $z^{-71} - S(z)$ providing zeros at 50 Hz and 100 Hz. The passband regions are from 3 Hz to 47 Hz and from 53 Hz to 97 Hz.

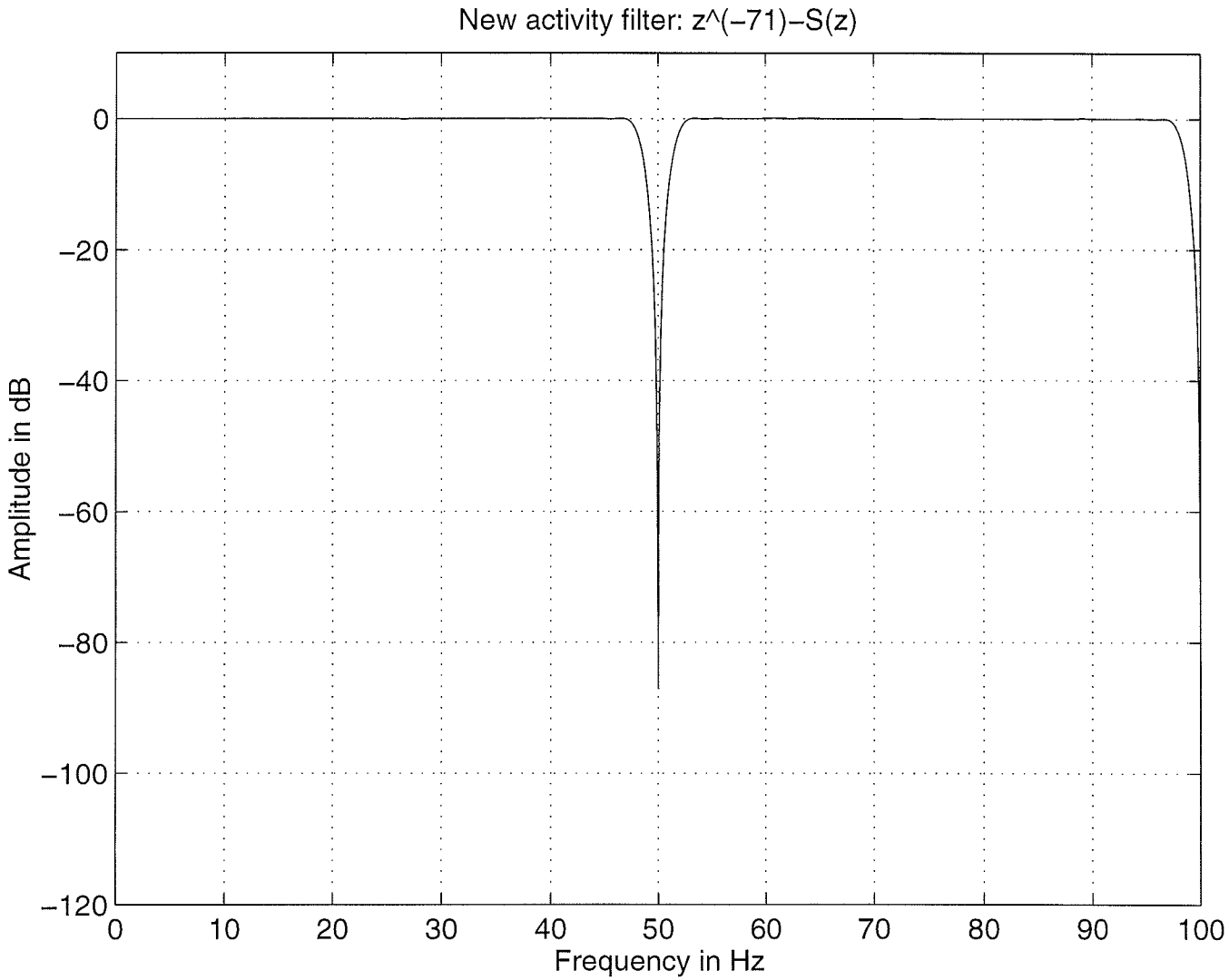
- In the following there are altogether 7 transparencies illustrating the performance of this design. One of them shows that the zero-phase frequency response of the overall filter. This response does not change its sign at 50 Hz so that the phase is linear, as is desired.

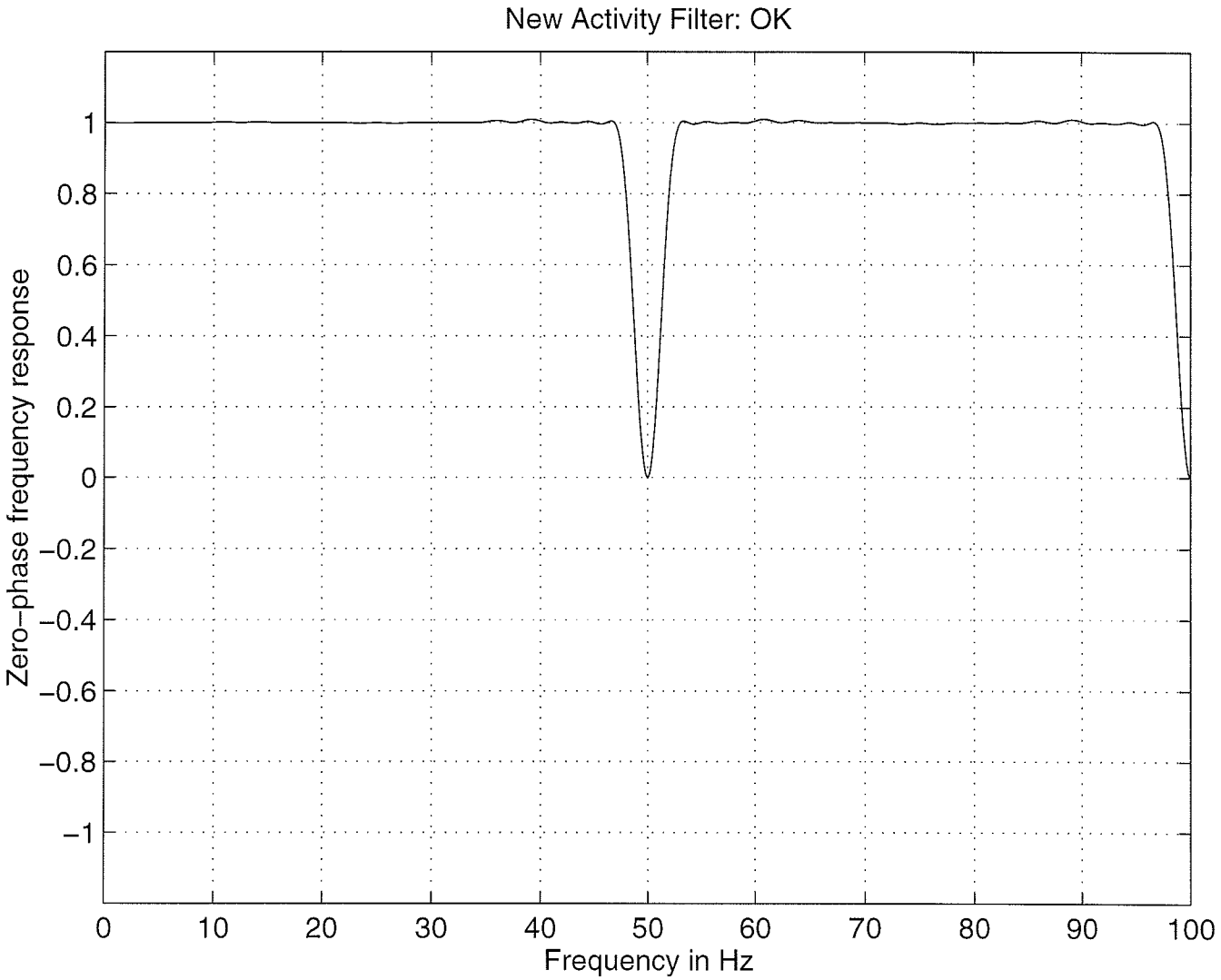

Components for T(z)=F(z^4)[RRS4(z)]^2: F(z) of order 7
Amplitude in dB
Frequency in Hz
10
0
−10
−20
−30
−40
−50
−60
−70
−80
0
10
20
30
40
50
60
70
80
90
100

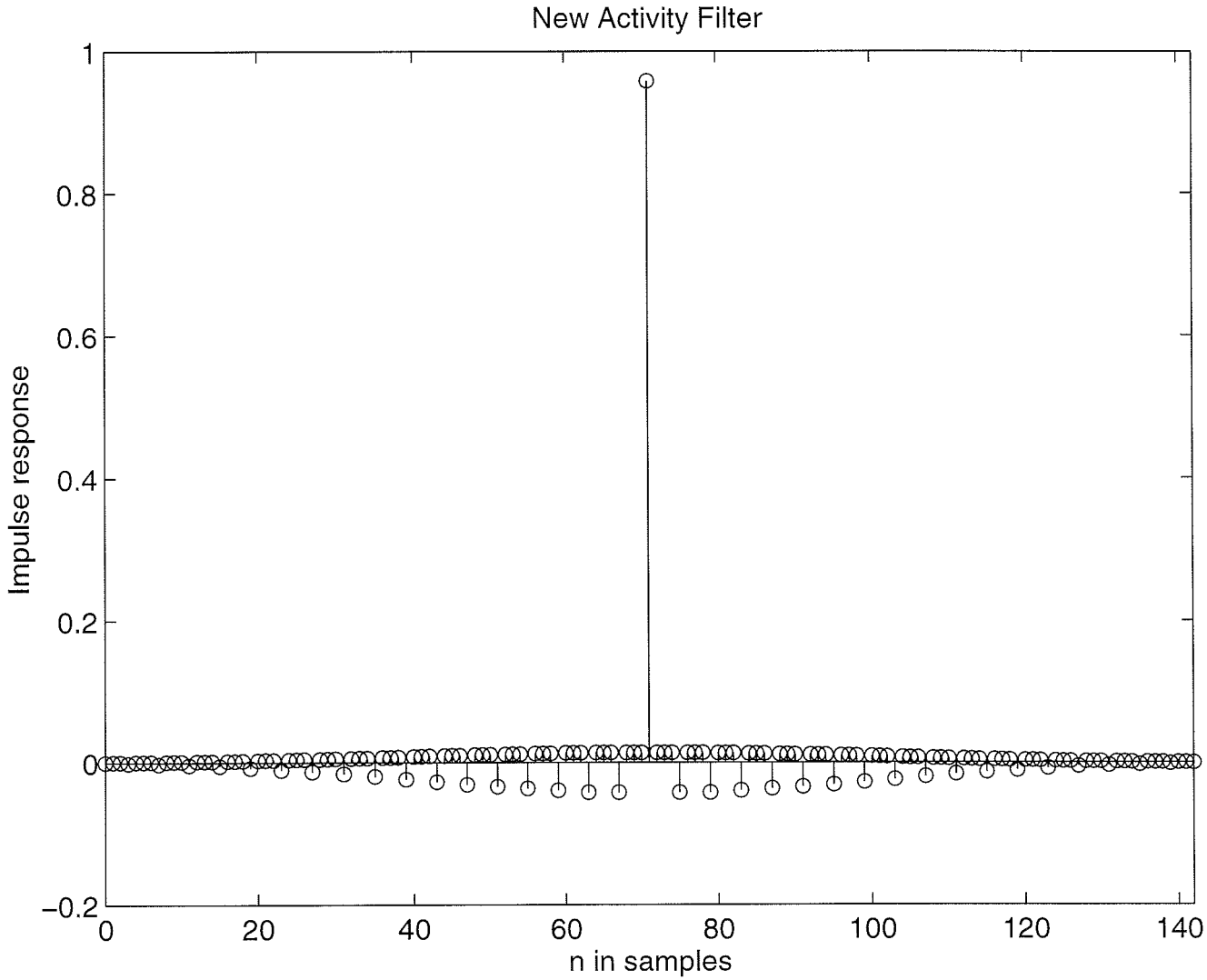

T(z)
10
0
−10
−20
−30
−40
−50
−60
−70
−80
Amplitude in dB
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Solid for T(z^4) and dashed for z^(−3)−[RRS4(z)]^2
Amplitude in dB
Frequency in Hz
10
0
−10
−20
−30
−40
−50
−60
−70
−80
0
10
20
30
40
50
60
70
80
90
100


S(z)=T(z^4)[z^(−3)−[RRS4(z)]^2]
10
0
−10
−20
−30
−40
−50
−60
−70
−80
Amplitude in dB
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz

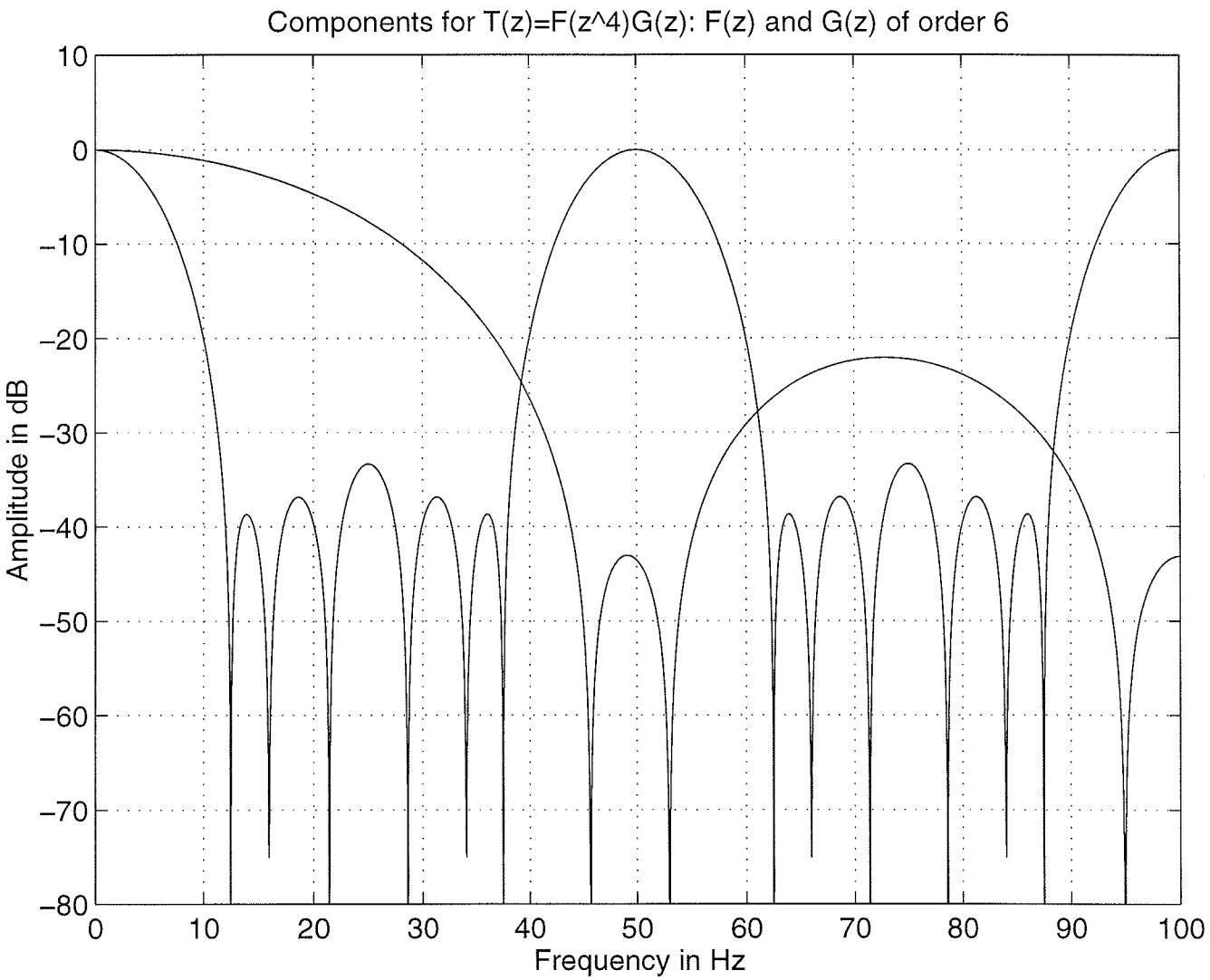

New activity filter: z^(−71)−S(z)
0
−20
−40
−60
−80
−100
−120
Amplitude in dB
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz

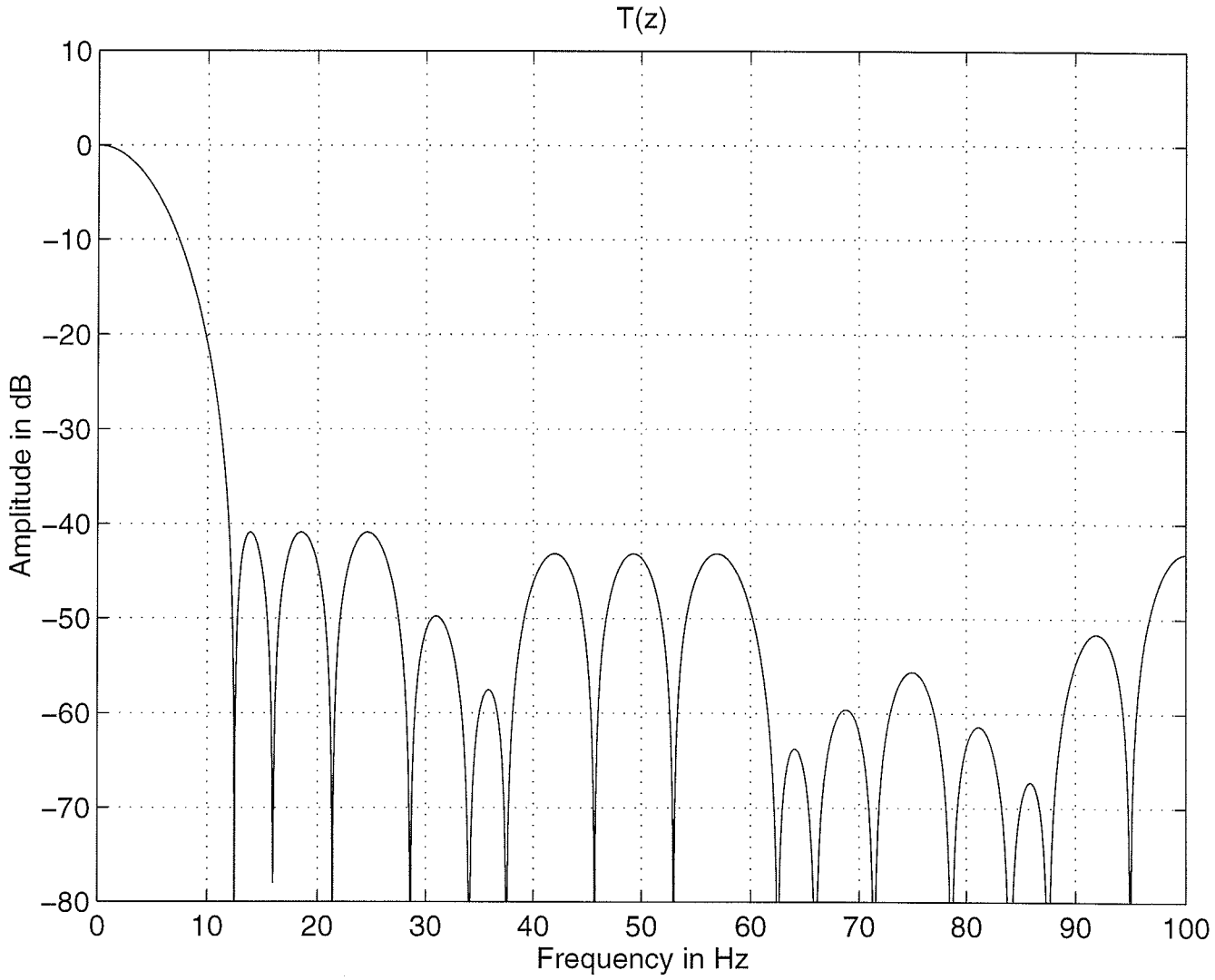

New Activity Filter: OK
Zero−phase frequency response
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz

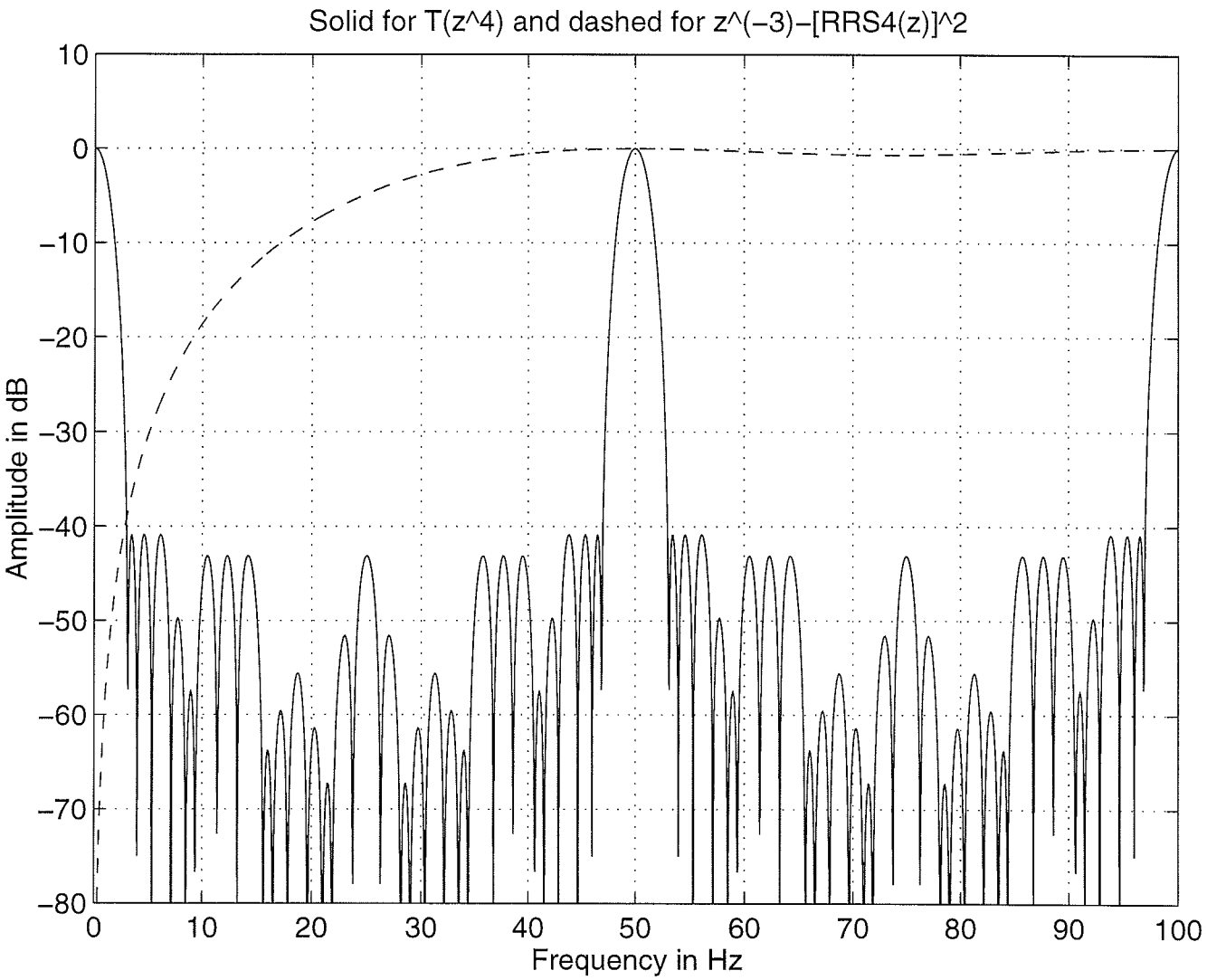

New Activity Filter
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
20
40
60
80
100
120
140

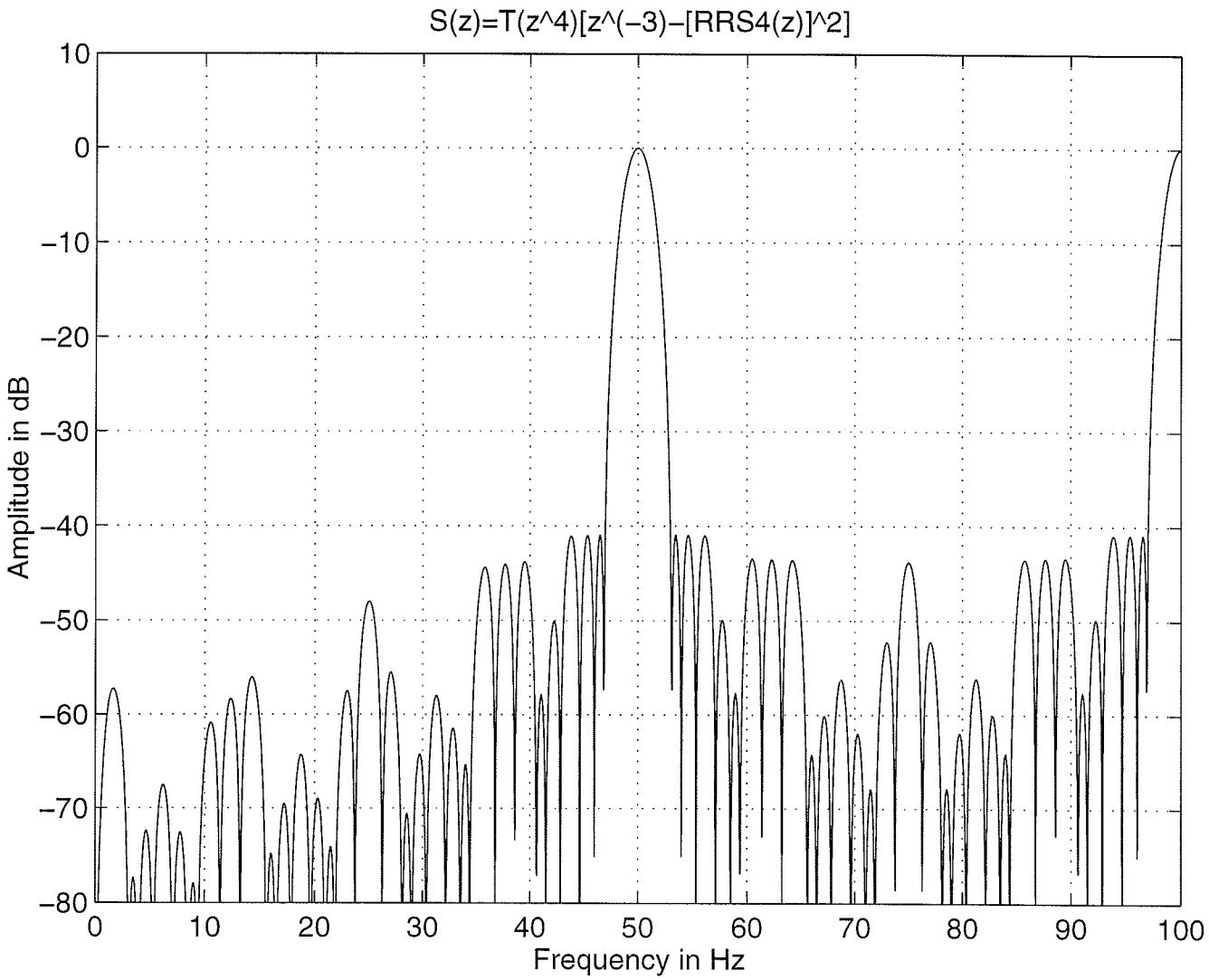
ACTIVITY FILTER: OPTIMIZED DESIGN

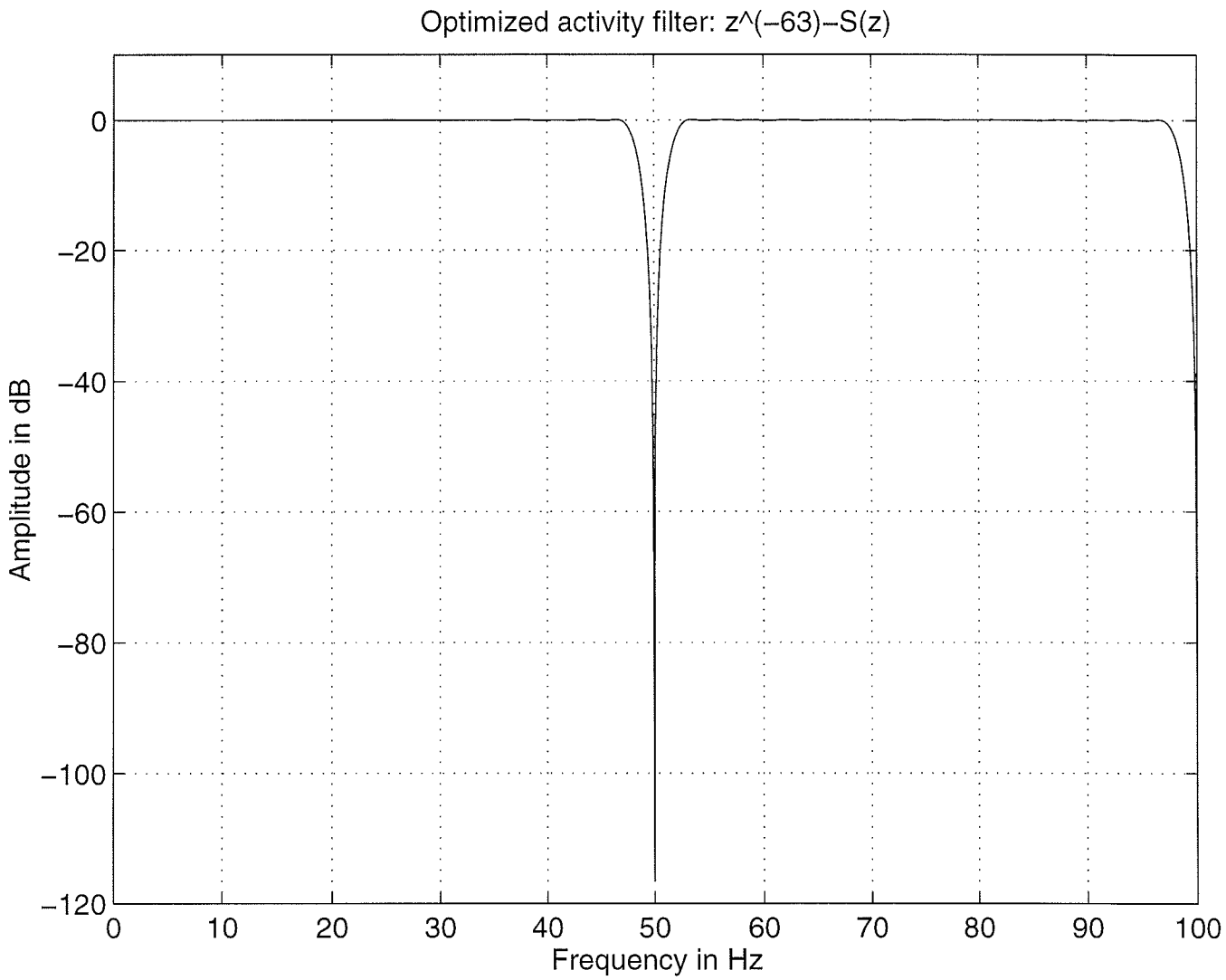
- Here, we design $T(z)$ in the form $T(z) = F(z^4)G(z)$ and $G(z)$ using the algorithm described in T. Saramäki, "Finite impulse response Filter S. K. Mitra and J. F. Kaiser, Eds, John Wiley & Sons, 1993, pp. 241-245. Otherwise the design is the same.
- Using this algorithm, the orders of both $F(z)$ and $G(z)$ are 6.
- In the following there are altogether 7 transparencies illustrating the performance of this design. One of them shows that the zero-phase frequency response of the overall filter. This response does not change its sign at 50 Hz so that the phase is linear, as is desired.

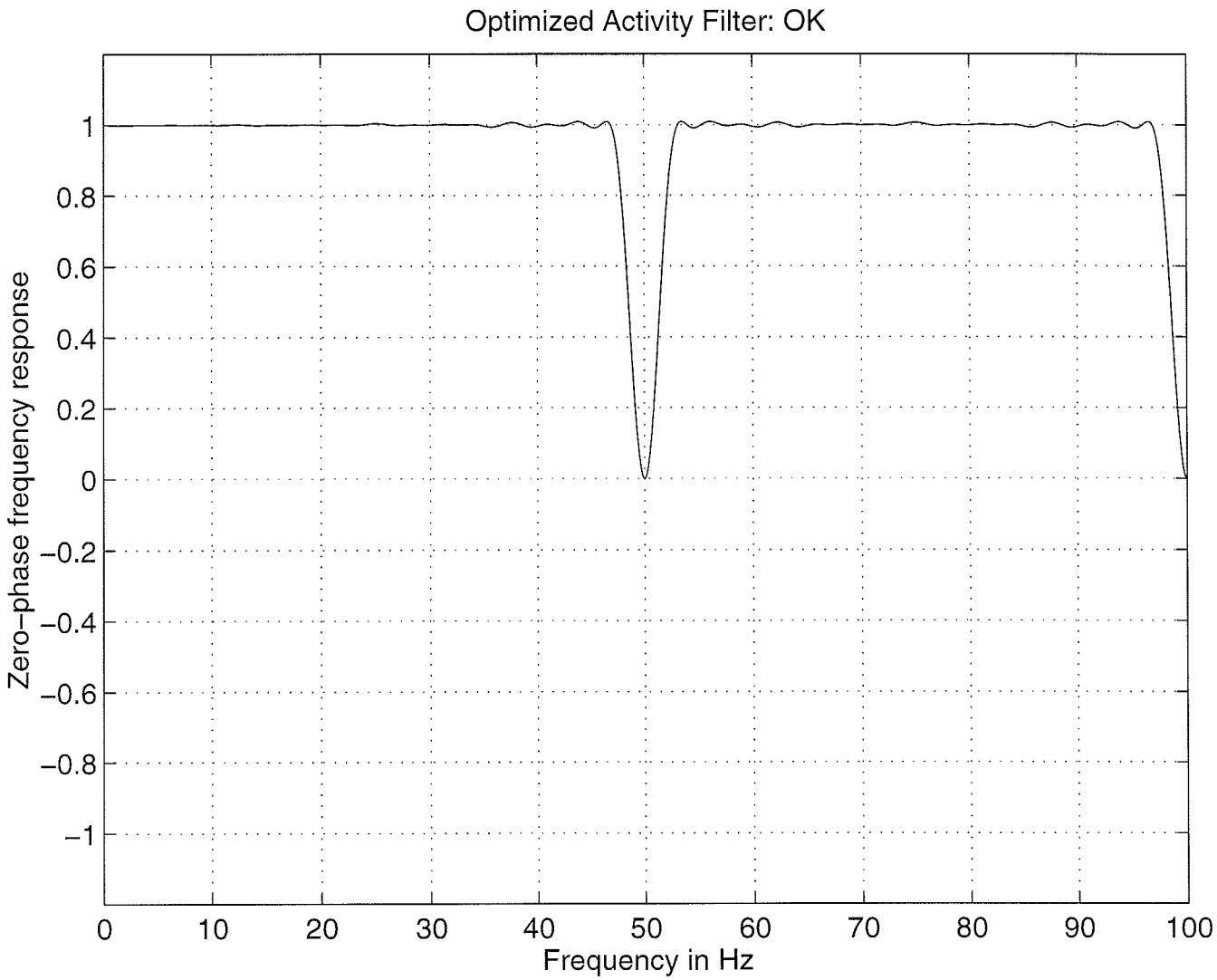

Components for T(z)=F(z^4)G(z): F(z) and G(z) of order 6
Amplitude in dB
Frequency in Hz
10
0
−10
−20
−30
−40
−50
−60
−70
−80
0
10
20
30
40
50
60
70
80
90
100

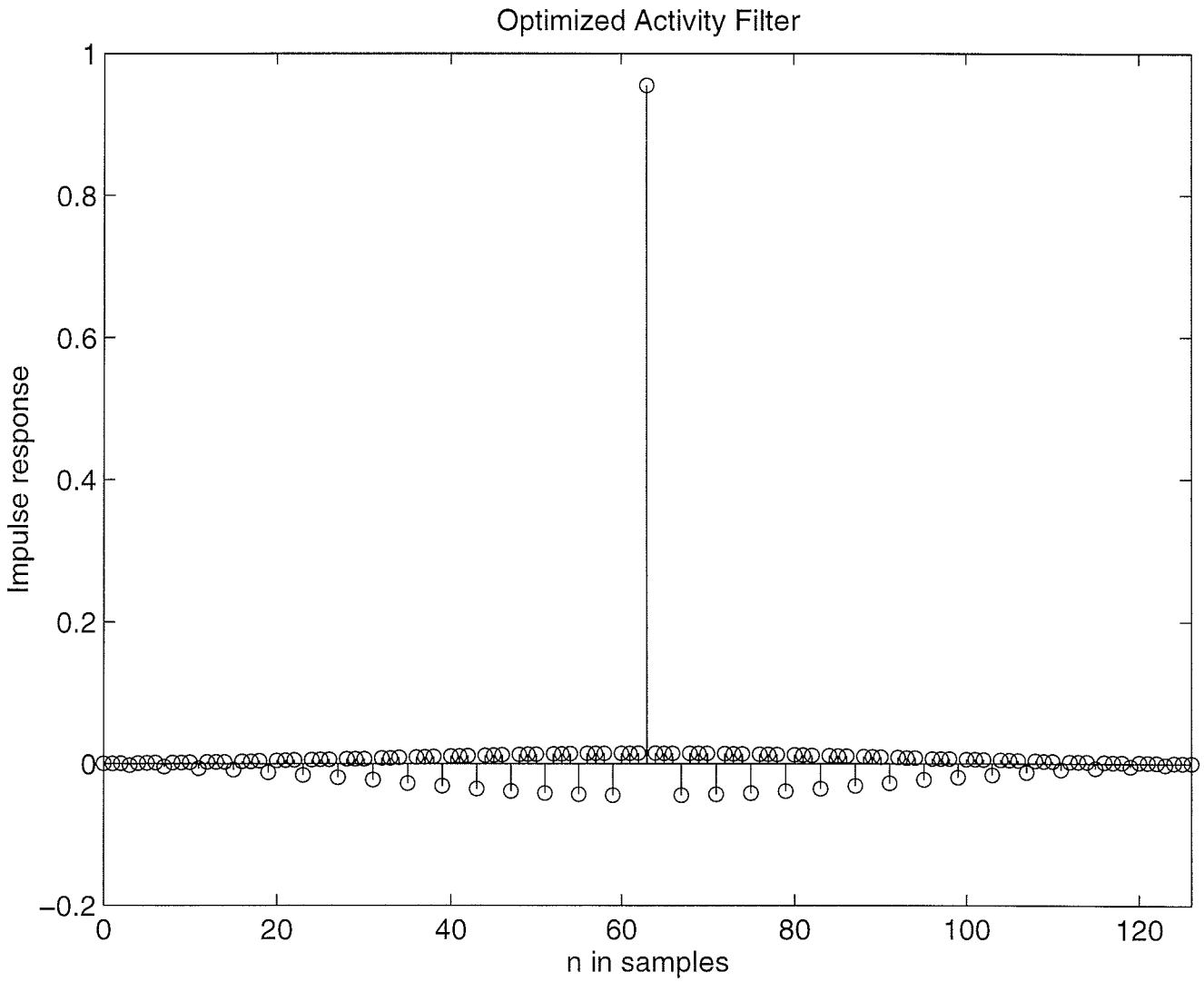

T(z)
10
0
−10
−20
−30
−40
−50
−60
−70
−80
Amplitude in dB
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Solid for T(z^4) and dashed for z^(−3)−[RRS4(z)]^2
Amplitude in dB
Frequency in Hz
10
0
−10
−20
−30
−40
−50
−60
−70
−80
0
10
20
30
40
50
60
70
80
90
100


S(z)=T(z^4)[z^(−3)−[RRS4(z)]^2]
10
0
−10
−20
−30
−40
−50
−60
−70
−80
Amplitude in dB
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Optimized activity filter: z^(−63)−S(z)
0
−20
−40
−60
−80
−100
−120
Amplitude in dB
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Optimized Activity Filter: OK
Zero−phase frequency response
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
10
20
30
40
50
60
70
80
90
100
Frequency in Hz


Optimized Activity Filter
Impulse response
n in samples
1
0.8
0.6
0.4
0.2
0
−0.2
0
20
40
60
80
100
120

```
% Matlab-file fishi1.m
%
%%%%%%%%%%%%%%%%%%%%%%%%%
%ECG-Filter: old version
%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%
%First ptototype H(z)
%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
f=[0. 8 18 100];
f=f/100;
m=[0 0 1 1];
w=[10 1];
h=remez(52, f, m, w);
figure(1)
[H,W]=zeroam(h,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Old ECG-Filter: Prototype H(z) of order 52')
figure(2)
impz(h)
xlabel('n in samples');
ylabel('Impulse response');
title('Old ECG-Filter: Prototype H(z) of order 52')
%%%%%%%%%%%%%%%%%%%%%%%%%
%Then H(z^4)
%%%%%%%%%%%%%%%%%%%%%%%%%
for k=1:53 hh(4*k-3)=h(k);end
figure(3)
[H,W]=zeroam(hh,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Old ECG-Filter: H(z^4)')
figure(4)
impz(hh)
xlabel('n in samples');
ylabel('Impulse response');
title('Old ECG-Filter: H(z^4)')
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%New ECG Prototype Filter: H(z)=z^(-M)-T(z)
%T(z)=F(z^3)G(z), M=half the order of T(z)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%For highpass H(z) the stopband edge is 8*pi/100=0.08pi
%and the passband edge is 18*pi/100=0.18pi
%stopband and passband ripples are 0.001 and 0.01
%T(z)=F(z^3)G(z) is a complementary lowpass filter
%It can be designed such that F(z) is a lowpass
%filter with edges at 0.24*pi and 0.54*pi
%and ripples of 0.0005 and 0.01
%G(z) has the same ripple values, passband region is
%[0, 0.08*pi] and stopband region is
%[2/3-(2*.18+.08)/3 2/3+(2*.18+.08)/3].
%
%Design F(z)
```

```
%
f=[0. .24 .54 1];
m=[1 1 0 0];
w=[20 1];
ff=remez(19, f, m, w);
%
%Design G(z)
%
f=[0. .08 2/3-(2*.18+.08)/3 2/3+(2*.18+.08)/3];
m=[1 1 0 0];
w=[20 1];
gg=remez(11, f, m, w);
%
%Impulse response of T(z)=F(z^3)G(z)
%
for k=1:20 fff(3*k-2)=ff(k);end
t=conv(fff,gg);
[H,W]=zeroam(fff,.0,1.,2000);
[HH,W]=zeroam(gg,.0,1.,2000);
figure(5)
plot(100*W/pi,20*log10(abs(H)),100*W/pi,20*log10(abs(HH)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Components for T(z)=F(z^3)G(z): F(z) and G(z) of orders 19 and 11')
figure(6)
[H,W]=zeroam(t,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('New ECG-Filter: T(z): complementary for the prototype filter')
%
%Prototype filter H(z)=z^(-34)-T(z)
%
h=-t;h(35)=1+h(35);
figure(7)
[H,W]=zeroam(h,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('New ECG-Filter: Prototype H(z)=z^(-34)-T(z)')
figure(8)
impz(h)
xlabel('n in samples');
ylabel('Impulse response');
title('New ECG-Filter: Prototype H(z)=z^(-34)-T(z)')
%%%%%%%%%%%%%%%%%%%%%%%%%%
%Then H(z^4)
%%%%%%%%%%%%%%%%%%%%%%%%%%
for k=1:69 hh(4*k-3)=h(k);end
figure(9)
[H,W]=zeroam(hh,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('New ECG-Filter: H(z^4)')
figure(10)
```

```
impz(hh)
xlabel('n in samples');
ylabel('Impulse response');
title('New ECG-Filter: H(z^4)')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Optimized ECG Prototype Filter: H(z)=z^(-M)-T(z)
%T(z)=F(z^3)G(z), M=half the order of T(z)
%F(z) and G(z) have been designed using the
%synthesis technique described in
%T. Saramaki, "Finite impulse response Filter
%Design" in Handbook for Digital Signal Processing,
%S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
%Sons, 1993,  pp. 241-245
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
clear all;
%Design F(z)
%
load ber11;
ff=rot90(ber11);
%
%Design G(z)
%
load ber12;
gg=rot90(ber12);
%
%Impulse response of T(z)=F(z^3)G(z)
%
for k=1:17 fff(3*k-2)=ff(k);end
t=conv(fff,gg);
[H,W]=zeroam(fff,.0,1.,2000);
[HH,W]=zeroam(gg,.0,1.,2000);
figure(11)
plot(100*W/pi,20*log10(abs(H)),100*W/pi,20*log10(abs(HH)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Components for T(z)=F(z^3)G(z): F(z) and G(z) of orders 16 and 6')
figure(12)
[H,W]=zeroam(t,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Optimized ECG-Filter: T(z): complementary for the prototype filter')
%
%Prototype filter H(z)=z^(-27)-T(z)
%
h=-t;h(28)=1+h(28);
figure(13)
[H,W]=zeroam(h,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Optimized ECG-Filter: Prototype H(z)=z^(-27)-T(z)')
figure(14)
impz(h)
xlabel('n in samples');
ylabel('Impulse response');
```

```
title('Optimized ECG-Filter: Prototype H(z)=z^(-27)-T(z)')
%%%%%%%%%%%%%%%%%%%%%%%%%%
%Then H(z^4)
%%%%%%%%%%%%%%%%%%%%%%%%%%
for k=1:55 hh(4*k-3)=h(k);end
figure(15)
[H,W]=zeroam(hh,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Optimized ECG-Filter: H(z^4)')
figure(16)
impz(hh)
xlabel('n in samples');
ylabel('Impulse response');
title('Optimized ECG-Filter: H(z^4)')
```

```
% Matlab-file fishi2.m
%
%%%%%%%%%%%%%%%%%%%%%%%%%
%Total Activity Filter: old version
%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%
%H(z)=G(z^2)G(z)
% For G(z), the passband edge
%is 94 Hz for f_s=200 Hz and it
%has a single zero at 100 Hz -->
%filter order must be odd
%Then H(z) contains this filter as well
%as the periodic G(z^2) having a single zero at
%50 Hz and passband edges 50+-3 Hz.
%%%%%%%%%%%%%%%%%%%%%%%%%
f=[0. 94/100];
m=[1 1];
w=[1];
g=remez(33, f, m, w);
for k=1:34 gg(2*k-1)=g(k);end
h=conv(gg,g);
figure(1)
[H,W]=zeroam(h,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Old Activity Filter')
figure(2)
plot(100*W/pi,H);
axis([0 100 -1.2 1.2]);grid;
ylabel('Zero-phase frequency response');
xlabel('Frequency in Hz');
title('Old Activity Filter: Error: Single zero at 50 Hz')
figure(3)
impz(h)
xlabel('n in samples');
ylabel('Impulse response');
title('Old Actvity Filter')
%%%%%%%%%%%%%%%%%%%%%%%%%
%New Solution
%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Step 1: Design
%T(z)=F(z^4)[RRS4(z)]^2 such that
%T(z) achieves the value of unity at the
%zero frequency and at least 40 dB attenuation
%at 4*3=12 Hz for f_s=200 Hz
%The impulse response of RRS4(z) is of length
%and all the values are equal to 1/4
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%[RRS4(z)]^2
rrs=[1 1 1 1];
rrs=rrs/4;
rrs=conv(rrs,rrs);
%
%Design F(z)
%The stopband edge for F(z) is 4*12 Hz
%so that the edge for F(z^4) is 12 Hz
```

```
%
f=[0. .001 4*12/100 1];
m=[1 1 0 0];
w=[100 1];
ff=remez(7, f, m, w);
%
%Impulse response of T(z)=F(z^4)[RRS4(z)]^2
%
for k=1:8 fff(4*k-3)=ff(k);end
t=conv(fff,rrs);
[H,W]=zeroam(fff,.0,1.,2000);
[HH,W]=zeroam(rrs,.0,1.,2000);
figure(4)
plot(100*W/pi,20*log10(abs(H)),100*W/pi,20*log10(abs(HH)));
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Components for T(z)=F(z^4)[RRS4(z)]^2: F(z) of order 7')
figure(5)
[H,W]=zeroam(t,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('T(z)')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Step 2: form T(z^4) and z^(-3)-[RRS4(z)]^2
%and cascade them as S(z)=T(z^4)[z^(-3)-[RRS4(z)]^2]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for k=1:length(t) tt(4*k-3)=t(k);end
rrs=-rrs;rrs(4)=rrs(4)+1;
[H,W]=zeroam(tt,.0,1.,2000);
[HH,W]=zeroam(rrs,.0,1.,2000);
figure(6)
plot(100*W/pi,20*log10(abs(H)),100*W/pi,20*log10(abs(HH)),'- -');
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Solid for T(z^4) and dashed for z^(-3)-[RRS4(z)]^2')
s=conv(rrs,tt);
figure(7)
[H,W]=zeroam(s,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('S(z)=T(z^4)[z^(-3)-[RRS4(z)]^2]')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Step 3 form z^(-71)-S(z)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ss=-s;ss(72)=1+ss(72);
figure(8)
[H,W]=zeroam(ss,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('New activity filter: z^(-71)-S(z)')
figure(9)
plot(100*W/pi,H);
```

```
axis([0 100 -1.2 1.2]);grid;
ylabel('Zero-phase frequency response');
xlabel('Frequency in Hz');
title('New Activity Filter: OK')
figure(10)
impz(ss)
xlabel('n in samples');
ylabel('Impulse response');
title('New Activity Filter')
clear all;
%%%%%%%%%%%%%%%%%%%%%%%%%
%Optimized Solution
%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Step 1: Design
%T(z)=F(z^4)G(z) such that
%T(z) achieves the value of unity at the
%zero frequency and at least 40 dB attenuation
%at 3*4=12 Hz for f_s=200 Hz
%F(z) and G(z) have been designed using the
%synthesis technique described in
%T. Saramaki, "Finite impulse response Filter
%Design" in Handbook for Digital Signal Processing,
%S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
%Sons, 1993,  pp. 241-245
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%Design of F(z)
%
load ber21;
ff=rot90(ber21);
%
%Design of G(z)
%
load ber22;
gg=rot90(ber22);
%
%Impulse response of T(z)=F(z^4)G(z)
%
for k=1:7 fff(4*k-3)=ff(k);end
t=conv(fff,gg);
[H,W]=zeroam(fff,.0,1.,2000);
[HH,W]=zeroam(gg,.0,1.,2000);
figure(11)
plot(100*W/pi,20*log10(abs(H)),100*W/pi,20*log10(abs(HH)));
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Components for T(z)=F(z^4)G(z): F(z) and G(z) of order 6')
figure(12)
[H,W]=zeroam(t,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('T(z)')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Step 2: form T(z^4) and z^(-3)-[RRS4(z)]^2
%and cascade them as S(z)=T(z^4)[z^(-3)-[RRS4(z)]^2]
%Use f_s=200 Hz
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for k=1:length(t) tt(4*k-3)=t(k);end
rrs=[1 1 1 1];
rrs=rrs/4;
rrs=conv(rrs,rrs);
rrs=-rrs;rrs(4)=rrs(4)+1;
[H,W]=zeroam(tt,.0,1.,2000);
[HH,W]=zeroam(rrs,.0,1.,2000);
figure(13)
plot(100*W/pi,20*log10(abs(H)),100*W/pi,20*log10(abs(HH)),'- -');
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Solid for T(z^4) and dashed for z^(-3)-[RRS4(z)]^2')
s=conv(rrs,tt);
figure(14)
[H,W]=zeroam(s,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -80 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('S(z)=T(z^4)[z^(-3)-[RRS4(z)]^2]')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Step 3 form z^(-63)-S(z)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ss=-s;ss(64)=1+ss(64);
figure(15)
[H,W]=zeroam(ss,.0,1.,2000);
plot(100*W/pi,20*log10(abs(H)));
axis([0 100 -120 10]);grid;
ylabel('Amplitude in dB');
xlabel('Frequency in Hz');
title('Optimized activity filter: z^(-63)-S(z)')
figure(16)
plot(100*W/pi,H);
axis([0 100 -1.2 1.2]);grid;
ylabel('Zero-phase frequency response');
xlabel('Frequency in Hz');
title('Optimized Activity Filter: OK')
figure(17)
impz(ss)
xlabel('n in samples');
ylabel('Impulse response');
title('Optimized Activity Filter')
```

Berlin '91

International Conference on DSP Applications and technology

October 28-31, 1991

Bioelectronic analyzer for raw water monitoring

J.Ranta[1], E. Aalto[2], T. Laakso[1], and I. Hartimo[1]

[1]Helsinki University of Technology
Laboratory of Signal Processing and
Computer Technology
Otakaari 5A
SF-02150 Espoo, Finland

[2]Helsinki City, Water and Wastewater Authority
Kuninkaantammentie 11
SF-00430, Helsinki, Finland

Abstract

In this paper, we present a method for monitoring surface raw water quality with the aid of a live fish. The behaviour of the fish is analyzed from the measured electromagnetic signals using digital signal processing methods. Useful variables for detecting stress responses have shown to be the total activity, coughing, breathing amplitude, breathing rate and heart rate (ECG). These variables are separated using digital filters implemented on the TMS320C25 signal processor. The obtained results are transferred to and displayed in real time on an IBM PC compatible computer. The data is stored in files in the PC for further analysis by spreadsheet programs etc.

1. Introduction

Surface raw water quality is measured by many physico-chemical analyses which indicate changes in water quality. When striving for faster and more subtle water control, one possibility is to use a bioelectronic water monitoring system, where the alterations in action potentials is measured from the musculature of a live fish.

It is known that fish, especially salmons, are very sensitive for alterations in the water quality. This is due to their highly developed smell and taste senses, of which smell is the more important for detection of alarming concentrations of harmful substances. The monitoring system is based on the principle that changes in the water quality result in changes of the physiological activity of the circulatory and respiratory systems, due to hormonal secretion. Variables for detecting stress responses are the total activity, coughing, breathing amplitude, breathing rate and heart rate.

1.1. System configuration

The water to be tested is run through 1 to 10 units of 94 liter aquariums with one fish in each. A symmetric pair of electrodes made of stainless steel are placed, the other under the fish and the other above the fish, to pick up the bioelectronic signals. The signals are amplified before filtering. The fish can freely move between the bottom and surface electrode without affecting the signal quality, although the fish usually lies on the bottom electrode.

The filters are implemented using an assembler program written for the TMS320C25 digital signal processor. The program can be configured to measure between one to ten fish by a change of a single parameter. Digital filters separate the breathing, coughing, heart muscle activity and total activity from the raw signal. In addition, the program measures the breathing amplitude, breathing rate and heart rate of each fish. The results obtained are fed to and displayed in real time on an IBM PC compatible computer. Data is written into files by the PC so that it can be statistically analyzed further by a spreadsheet program. See Figure 1 for system configuration.

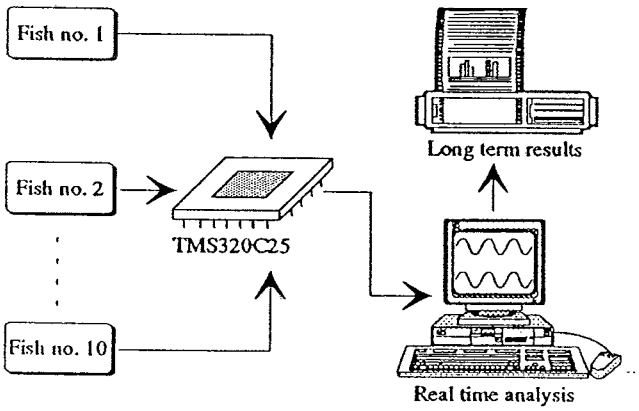


Figure 1. System configuration.

1.2. Characteristics of measured signals

Because the raw signal with useful information contains frequency components from 0.1 Hz to 100 Hz, the sampling rate was chosen as 200 Hz. The raw signal also contains a disturbing line frequency at 50 Hz and its harmonic at 100 Hz which have to be removed digitally.

This results in a signal that shows the total activity of the fish which is one of the desired parameters.

The ECG signal contains frequency components from 4.5 Hz up to 100 Hz. Figure 2 shows an example of an electrocardiogram (mean record of 2000 sweeps triggered from the raw signal /6/). The brown trout has a two-chambered heart, and the P-wave signals the depolarization of the atrium and QRS-complex the depolarization of the ventricle. The T wave signifies the repolarization of the ventricle and the relaxation of the ventricular muscle.

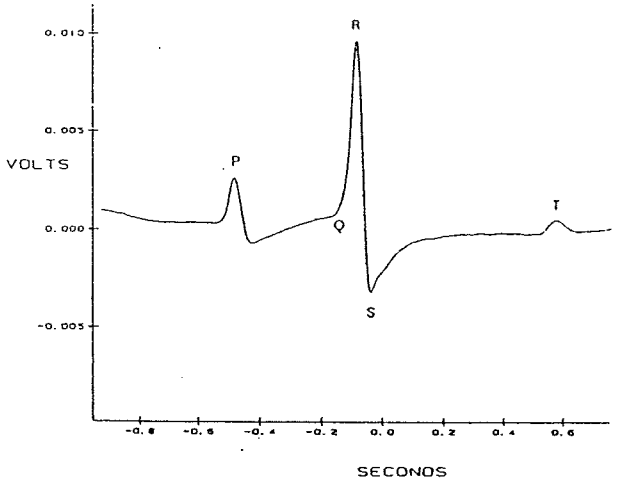


Figure 2. The 2000 sweep mean ECG.

Breathing and coughing signals occupy mainly low frequencies, 1.7 - 4.0 Hz for coughing and only 0.1 - 1.5 Hz for breathing. When these signals are filtered, sampling rate reduction is advantageous so that lower filter lengths can be used. From these specifications it can be seen that the transition band for ECG should be quite narrow in order to suppress coughing signals. Transition band starting at 2.0 Hz has been found sufficient. Because of the energy due to breathing and coughing at lower frequencies, the stopband attenuation should be at least 60 dB.

Since the signal shape is to be maintained, the filters should all have linear phase in the passband.

1.3 The overall filter structure

The 200 Hz sampling frequency is dropped in two stages (by 4 and 4) down to 12.5 Hz. Figure 3 shows the filter structure used. The filters shown in Figure 3 are explained in the following sections.

2. Methods used for solution

2.1. Problems encountered

When TMS320C25 is used for signal processing, its internal memory size (only 544 words) becomes the bottleneck of the system. Because of the strict requirements, the number of filter coefficients resulting from the straight forward design with the Parks-McClellan algorithm /8/ would require more than 400 words of program memory. Each fish needs also over 400 words of data memory for storage of the state variables. With 544 words of internal data memory, only one fish could be measured. Thus the filter state variables have to be stored in external data memory.

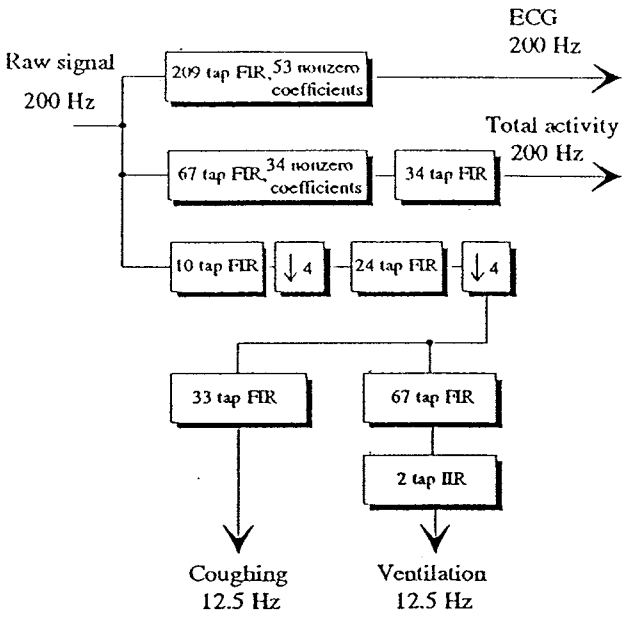


Figure 3. The overall filter structure used for measurements of one fish.

Additionally, the data move function (i.e., moving the processed data memory value to a next higher location) of instructions MACD, DMOV and LTD /7/ can only be performed in internal data memory. So, in order to perform the calculations, the processor has to transfer the filter delay lines from external memory to calculation area in internal data memory, do the calculations, and transfer the modified delay line back to external memory to wait for the next cycle. This results in a massive amount of traffic between the memories, and the processor will spend most of its time doing these memory transfers. Clearly, some method for reducing this traffic is needed.

2.2. FIR filters with sparse impulse response

Let us consider a FIR filter H(z) with impulse response h(n). If we insert L-1 zeros between the original samples of h(n), we obtain FIR filters with sparse impulse response /1/. This means that the delays of the original filter are replaced with L delays. In frequency domain this insertion results the frequency response to be

periodic with a period of $2\pi/L$. In addition, the passband and transition bandwidths are 1/L of the corresponding widths of the original filter. By designing the original transition band properly, one can benefit from this periodic frequency property a great deal. Figure 4 shows the frequency domain behaviour of a filter when three zeros are inserted between original samples.

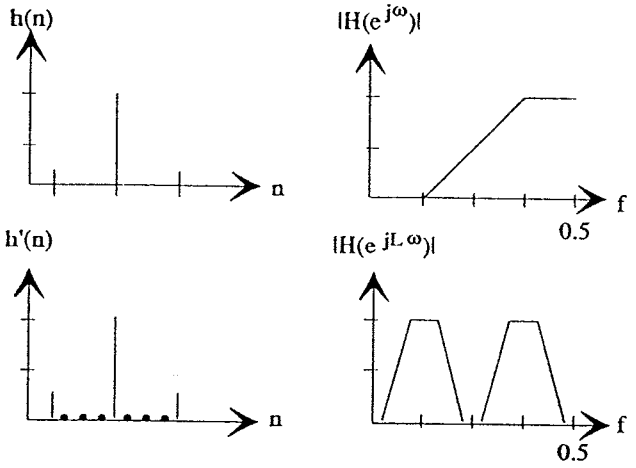


Figure 4. Impulse and frequency response representations of a sparse impulse response FIR filter when L = 4.

The amount of computations required are 1/Lth of the filter of the same length. Also the program memory needed to store the filter coefficients is reduced with the same factor. In this case however, the greatest advantage is that the processor now needs to transfer only 1/Lth of the delay line to calculation area in internal data memory.

2.3. The decimation process

Before coughing and breathing signals can be filtered, it is necessary to reduce the sampling rate in order to use smaller filter lengths. This sampling rate reduction is called *decimation* and it means that some of the input samples are simply discarded. Before this can be done, it is necessary to bandlimit the input signal with a lowpass filter to avoid *aliasing*. Aliasing means that higher frequencies fold over the lower frequencies thus corrupting the original signal.

The decimation factor needed is 16. Decimation should be done in stages so that the requirements for the decimation filters are less stringent. Dividing the decimation in stages can be done in many ways, either four, three, two or one stage could be used. In this particular case, two stages both with a decimation factor of four turned out to be the best choice. Two lowpass FIR filters of orders 9 and 23 satisfied the specifications.

More details on the efficient implementation of decimation filters can be found in /3/ and /4/.

3. Filters designed

3.1. The ECG filter

The measured raw signal contains disturbing 50 Hz and 100 Hz frequencies which have to be filtered out. By using a FIR filter with sparse impulse response, these disturbances can be eliminated efficiently. If we design a highpass filter with a transition band of 8 to 18 Hz, a stopband attenuation of 60 dB, and a passband ripple of 0.0864 dB, filter order of 52 will be sufficient. As already seen in Figure 4, by inserting three zeros between the impulse response of the original filter we obtain a filter with stopbands of 0 - 2 Hz, 48 - 52 Hz, and 98 - 100 Hz with the overall filter order of 208 but with only 53 nonzero coefficients. Figure 5 shows the specifications of the ECG filter. Now the DC, 50 Hz, and 100 Hz frequencies will be removed.

In Figure 10 the processor simulation results of this filter as well as the noise power measured by a program called *Noise and Distortion Measurement Program* (NDM) /2/ are shown.

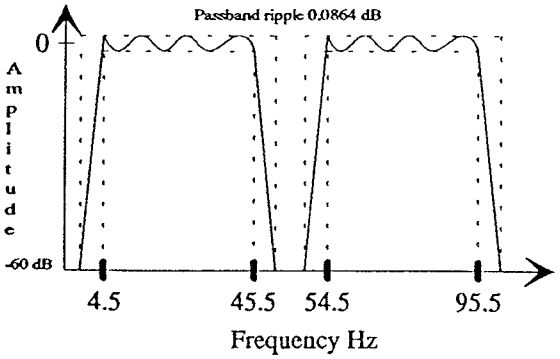


Figure 5. Specifications for the ECG filter.

In addition, the program calculates the time between two consecutive ECG spikes. This calculation is done by comparing each result to a threshold value. If the result is below this threshold, a counter is incremented, otherwise the counter value is multiplied by a time constant. (In this particular case, it is the inverse of the sampling frequency, namely 1/200 s). Figure 16 shows the raw signal used for testing, and the result of applying this filter to it is shown in Figure 17.

3.2. Total activity

When removing the 50 Hz net frequency and its harmonic at 100 Hz from the measured signal, we get the total activity of the fish. In this case it is necessary to use two filters in cascade. The original filter, which is used in the second stage, is a lowpass filter with a zero at 100

Hz and a passband edge at 94 Hz. The first filter is obtained from the original by inserting one zero between each coefficient so that it has a zero at the 50 Hz frequency and transition bands from 47 to 50 Hz and from 50 to 53 Hz. Figure 6 presents the specifications for the total activity filter. Figure 7 illustrates the effect in the frequency domain when one zero is inserted between each coefficient.

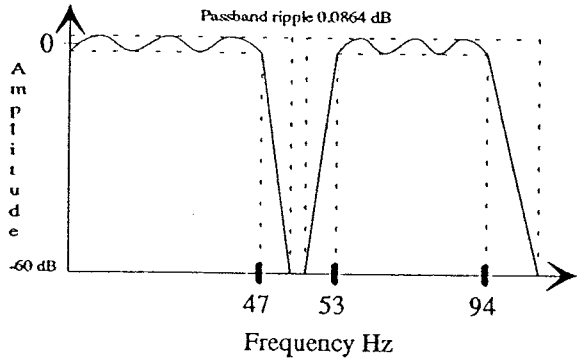


Figure 6. Specifications for the total activity filter.

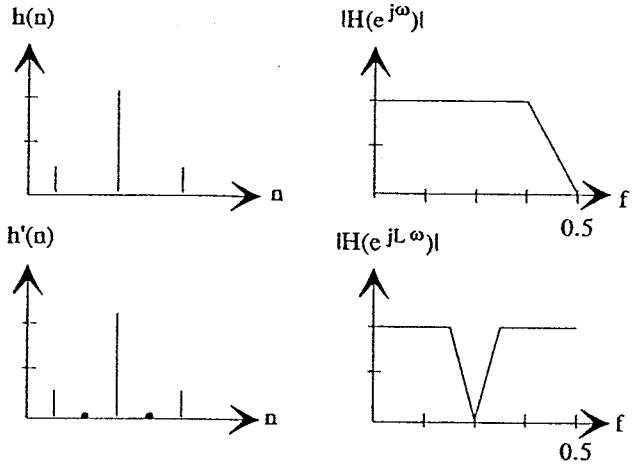


Figure 7. Impulse and frequency response representations of a sparse impulse response filter when L = 2.

Figure 11 contains the processor simulation results of the frequency response and noise power of this filter. In Figure 18 the result of applying this filter to the test signal of Figure 16 is shown. It should be noted that the test signals contained some noise over the whole frequency range which was due to poor recording. That is why the results, namely the ECG and the total activity do not look as good as they could.

3.3. Breathing

Breathing signal contains information in the range of 0.1 - 1.5 Hz. Stopband requirements are 60 dB in the upper stopband because of the coughing signals that appear nearby. Also the DC level at lower stopband must be removed completely so that the zero crossings can be found from the breathing signal. Filtering is done in two stages. First, a lowpass FIR filter is applied with transition band of 1.5 to 2.0 Hz. A filter order of 66 were found to be sufficient.

Because the lower edge of the filter is so close to zero frequency, FIR filters would need more than 500 taps to satisfy the requirements and this would introduce a delay of about 20 seconds into the output signal. Clearly this kind of huge delay cannot be tolerated in a real-time system.

Our solution is to use a first-order IIR filter with a zero on zero frequency and a pole on real axis as close to the zero as possible. By moving the pole close to the zero (which is on the unit circle in z-domain), group delay near the lower edge becomes nearly linear and thus the filter behaves almost like a FIR filter. There are disadvantages, however, because the impulse response is a slowly decaying exponential sequence. As the pole gets closer to the zero, the time required for the output signal to reach zero level will raise. On the other hand, if the pole is moved too far from zero, group delay will increase at the lower edge frequency so much that it will affect the output signal. Pole location must be selected so that the group delay at 0.1 Hz will be as small as possible. Figure 8 shows the specifications for the breathing filter.

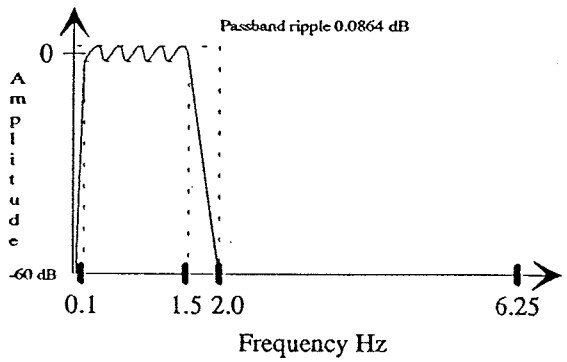


Figure 8. Specifications for the breathing filter.

In order to reduce the quantization noise of the recursive filter, first-order error feedback /5/ is applied to the quantization point of the IIR filter. Normally, when multiplying two sixteen bit numbers the result is 32 bits long. Only the upper sixteen bits are used and the lower part is ignored. Error feedback means that the lower part of the result is saved and at the next cycle it is added to the output. This has the effect of reducing the noise spike at low frequencies. The simulation results are shown in Figure 12 where the lower solid line represents the noise power with error feedback and the dotted line denotes the noise power without error feedback. In this case the noise

spike was removed completely. Figure 13 contains the group delay simulation result of this filter. It can be seen, the group delay is practically constant in the passband.

Figure 19 shows the result of applying this filter to the test signal. It should be noted that because the test signal contained only five breathing cycles, the output did not have enough time to reach the zero level. This result also contains the two decimation stages shown earlier.

3.4. Coughing

Frequency range from 1.7 to 4.0 Hz contains the energy of coughing signals. Also in this case it is necessary to have stopband attenuation of 60 dB at both sides. This is due to the fact that the upper stopband has some energy left from the ECG signal and the lower stopband contains breathing signals aswell as the DC component. A filter of the order of 33 satisfied the requirements when transition bands were chosen to be from 0.6 to 1.7 Hz and from 4.0 to 5.1 Hz. Figure 9 contains the specifications for the coughing filter.

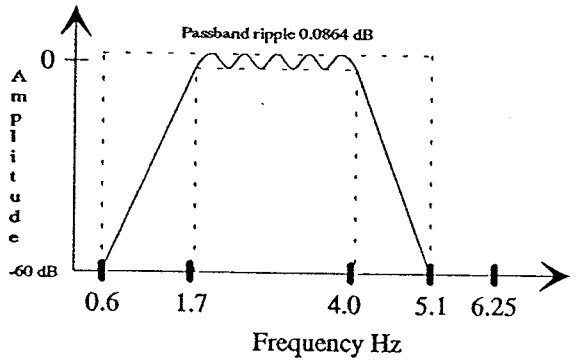


Figure 9. Specifications for the coughing filter.

Figure 14 shows the frequency response and noise power of the filter and Figure 20 shows the result when this filter was applied to the test signal of Figure 15. The test signal contains coughing between samples 600 and 700 and it appears as a wide drop in the figure. When filtered, coughing looks like it is shown in Figure 20 between samples 60 and 70. Note that the decimation stages are also included in Figure 20.

4. Transferring the measured results to PC

The results obtained with the TMS320C25 are delivered to an IBM PC compatible computer for displaying and further processing. The results are written into files by PC and processed statistically with some spreadsheet program, like Lotus or Microsoft Excel, for example. In order to do this, some method of communication must exist between the two processors. The most efficient way to accomplish this task is to share the TMS320C25's external data memory with PC so that no complicated handshaking methods or traffic control are needed. After TMS320C25 has finished computing, the results are written to data memory and the PC is signaled that new results are ready. TMS320C25 has a counter initialized so that an interrupt is caused once every 1/200th second, so after writing the results to data memory the TMS320C25 goes to an infinite loop until an interrupt occurs. Everything must be done within this time limit, otherwise the system gives false results because new data arrives before the calculation of the current data is finished.

Each measuring point has a table in external data memory where the results are written. The PC must naturally have some knowledge of where these tables reside. This information can be obtained in the initialization phase in a number of ways. The PC might be programmed to always read certain locations of TMS320C25's data memory to obtain these addresses or the addresses could be fixed in the program code.

1	Variable field
2	ECG result
3	ECG time counter
4	Time between ECG spikes
5	Total activity result
6	Coughing result
7	Breathing result
8	Breathing cycle time
9	Max breathing amplitude

Table 1. The table of results in TMS320C25's external data memory.

It should be noted that only some of the information is available at each time slot, namely ECG and total activity. Due to the decimation, other results are obtained every 0.08 second or even less frequently. Reading these same values many times is clearly unnecessary. This problem is solved by having a variable at the first location of the results table which contains the information of when these other results are ready. Table 1 shows the table format used.

When the first bit of the variable is one, it means that the table has in its 4th position a value telling the time elapsed between two consecutive ECG spikes. When the second bit is one, the table's 6th and 7th positions have values for coughing and breathing, respectively. When the third bit is one, the 8th location contains a value that gives the breathing cycle time and the 9th location has a value for the maximum amplitude that occurred during the last breathing cycle.

5. System performance

The following results are obtained from the program code when the TMS320C25 is clocked with 5 MHz and the external memory introduces no wait states. (A faster, 10 MHz version of TMS320C25 is available but it is more expensive and faster clocking rates also require quicker and more expensive memories.)

Table 2 shows the used processor power as computed from the program code assuming that each instruction requires only one clock cycle. This assumption is true in most cases because of pipelining. Calculation times for heart rate, maximum breathing amplitude and breathing cycle time are also included. It should be noted that the results are average values because the decimating filters are initialized so that they are all in a different phase. This means that only one fish at a time is measured at the lowest 12.5 Hz sampling rate. For example, in case of a single measuring point, the worst case cycle takes 912 instructions, but most of the time (fifteen out of sixteen) only 442 instructions are needed.

For comparison, Table 2 also contains the processor power used of the previous version of the program. This program used a 300 Hz sampling rate for the ECG and the total activity filters and a 50 Hz sampling rate for the breathing and coughing. That is why the filter lengths used by the program were different. Note that the previous version performed only filtering functions.

Measuring points	Previous version	Current version
1	27.1%	1.9%
2	54.3%	3.8%
3	81.4%	5.6%
4	—	7.5%
5	—	9.3%
6	—	11.2%
7	—	13.0%
8	—	14.9%
9	—	16.7%
10	—	18.6%

Table 2. Total processor power used, calculated from the program code.

Because delay lines for the decimation filters are stored in internal data memory, the upper limit for measuring points is ten. It is important to measure several fish so that reliable results can be obtained. For example, some sites might require the use of ten fish when monitoring drinking water intakes. By moving the delay line of the second stage decimation filter in external program memory, about 40 fish could be measured without problems. However, it is worth while to remember that each fish needs a tank which holds about 100 litres of raw water. These tanks require a lot of space and naturally the fish also demand some caring. Therefore, ten measuring points is quite sufficient in practice. Currently the number of fish used with the system is four.

When low sampling rates are used, it is clear that real-time performance will suffer. The advantages of the current version of the program over the old one can be seen from Table 3. Savings in filter coefficients are nearly 86%, mainly because of the use of sparse impulse response filters. The system delay is also reduced in every case but breathing. The longer delay of the breathing filter is caused by the use of decimation filters. The difference between the delays roughly equals the delay of these two decimation filters.

Current version	Filter length	Filter coeff.	Delay
ECG	209	53	0.52 s
Total Activity	67+34	68	0.25 s
Coughing	33	33	1.53 s
Breathing	67	67	2.97 s
Previous version			
ECG	512	512	0.85 s
Total Activity	512	512	0.85 s
Coughing	256	256	2.56 s
Breathing	256	256	2.56 s

Table 3. Filter lengths, number of filter coefficients and total delays calculated from the program code.

The filter lengths have also been reduced somewhat, partly because of the 100 Hz difference in the sampling rate. The decimation by 16 also reduces the lengths of the breathing and coughing filters. It should also be noted that the filter specifications were slightly tightened as compared to the previous version.

Processor simulations

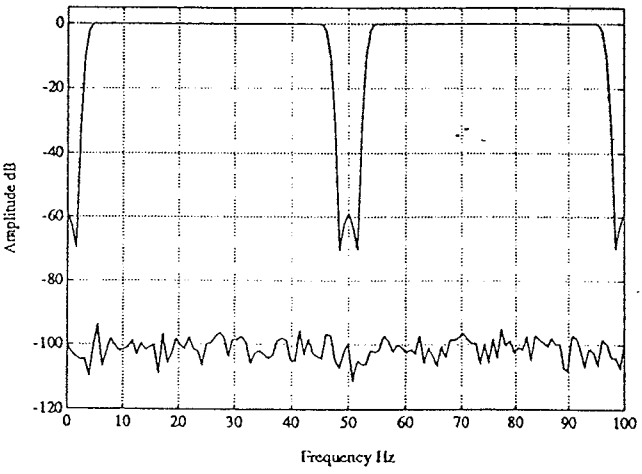


Figure 10. Processor simulation of the ECG filter. Average noise power -100.25 dB.

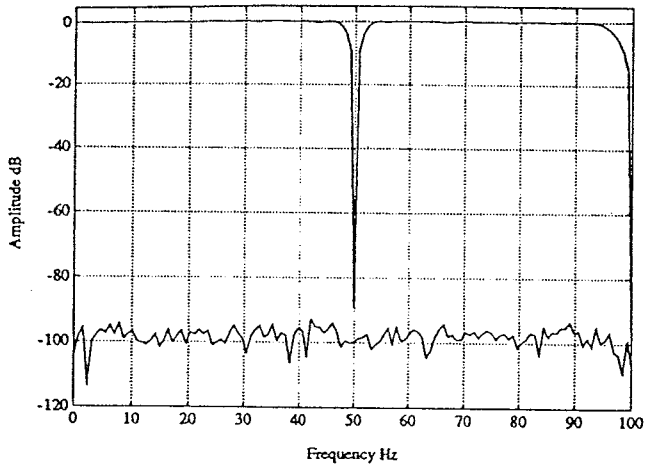


Figure 11. Processor simulation of the total activity filter. Average noise power -97.48 dB.

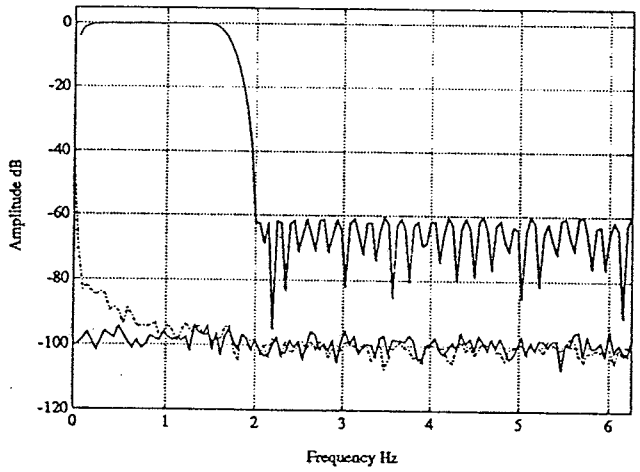


Figure 12. Processor simulation of the breathing filter. Average noise power (the lower solid line) with error feedback -98.65 dB and without error feedback (the dashed line) -65.03 dB.

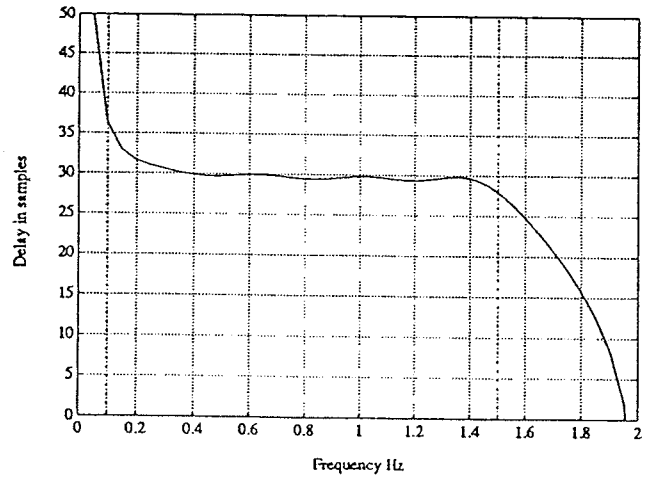


Figure 13. Group delay of the breathing filter.

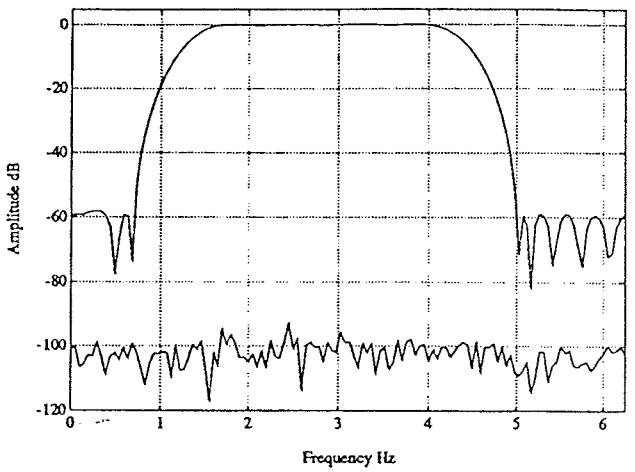


Figure 14. Processor simulation of the coughing filter. Average noise power -101.29 dB.

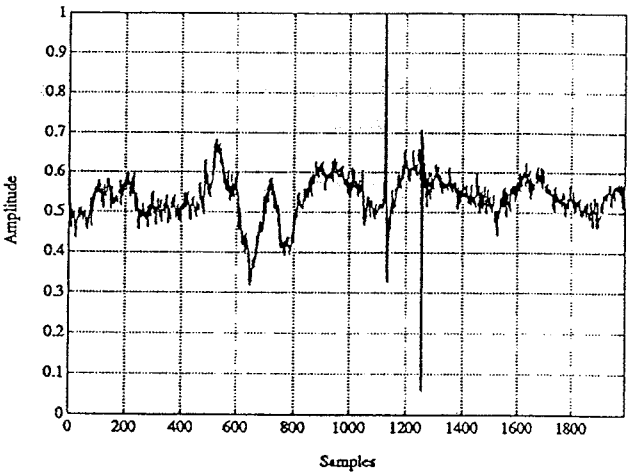


Figure 15. Raw signal that contains coughing.

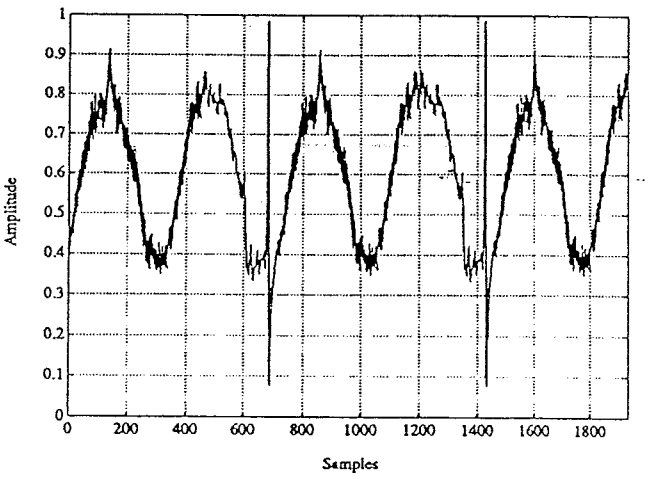


Figure 16. Raw signal that contains breathing and two ECG spikes.

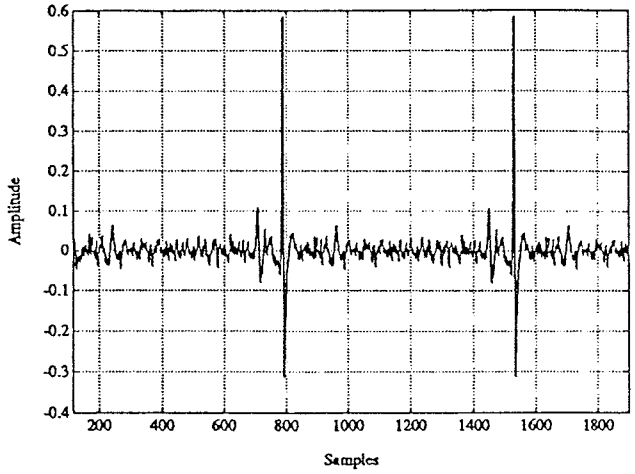


Figure 17. Simulator output when signal in Figure 16 is filtered with the ECG filter in Figure 10.

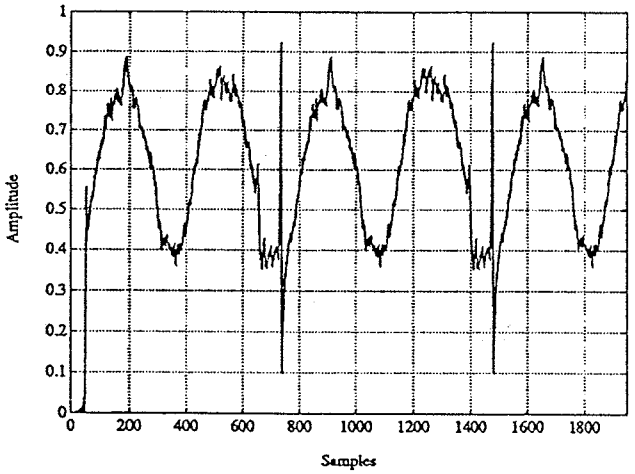


Figure 18. Simulator output when signal in Figure 16 is filtered with the total activity filter in Figure 11.

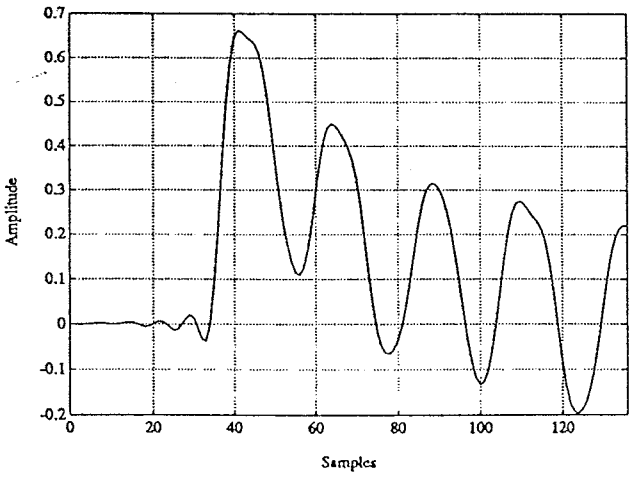


Figure 19. Simulator output when signal in Figure 16 is filtered with the breathing filter in Figure 12.

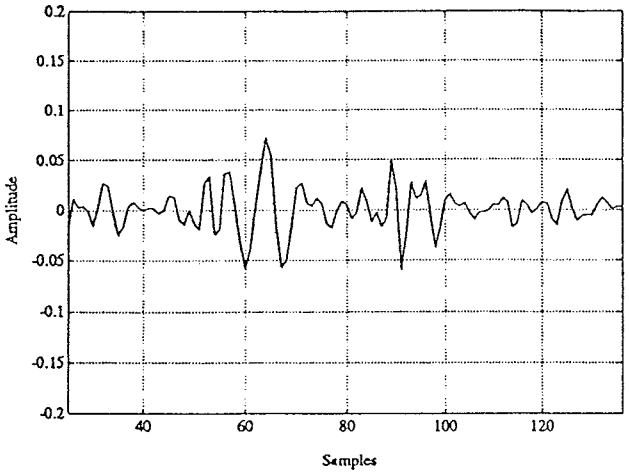


Figure 20. Simulator output when signal in Figure 15 is filtered with coughing filter in Figure 14.

References

/1/ Y. Neuvo, D. Cheng-Yu, and S. K. Mitra, "Interpolated finite impulse response filters", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-32, No. 3, pp. 563 - 573, June 1984.

/2/ H. W. Schüßler and P. Steffen, "A new method for measuring the performance of weakly nonlinear system", *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '89)*, pp. 2089-2092.

/3/ P. P. Vaidyanathan, "Multirate Digital Filters, Filter Banks, Polyphase Networks, and Applications: A tutorial", *Proceedings of the IEEE*, vol. 78, No. 1, January 1990.

/4/ R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1983, sec. 3.3, pp. 84-85.

/5/ T. Laakso, O. Hyvärinen, and M. Renfors, "Signal processor implementation of some recursive digital filter structures", *Proc. Int. Conf. on Digital Signal Processing*, pp. 220-224, Florence, Italy, Sept. 7 - 10, 1987.

/6/ E. Aalto and S. Smeds, "*Bioelectronic analyzer for raw water monitoring*", Demonstration at the XXXI International Congress of Physiological Sciences, Helsinki, 1989.

/7/ Texas Instruments, *Second Generation TMS320 User's Guide*, Digital Signal Processing Products, 1989.

/8/ T. W. Parks and C. S. Burrus, *Digital filter design*. John Wiley & Sons, Inc. New York 1987