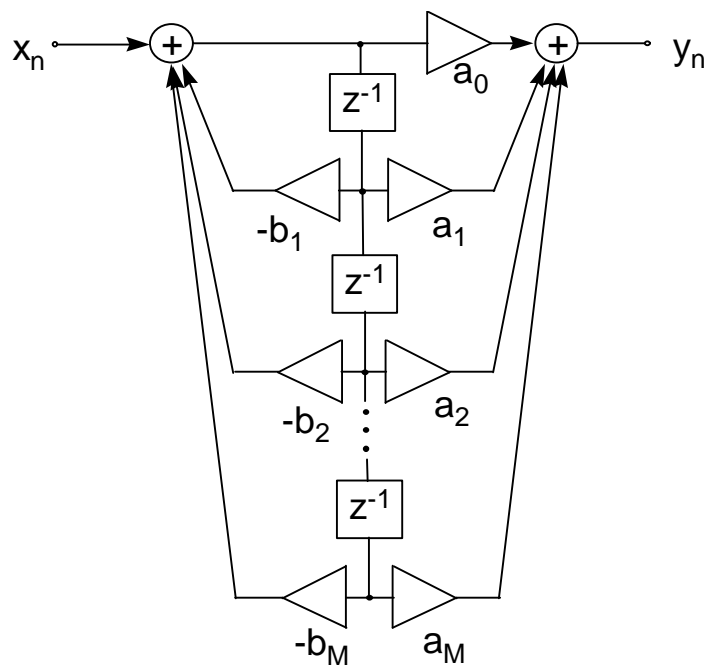


Filter Design Wizard (FIWIZ) User Manual



1. LICENSE AGREEMENT	4
2. GLOSSARY	5
3. INSTALLATION INSTRUCTIONS	5
3.1 Version which requires a Java virtual machine	5
3.2 Windows ® executable	6
4. INTRODUCTION	7
5. USAGE OF FIWIZ	10
5.1 Startup Screen	10
5.2 Magnitude Screen	11
5.3 Group Delay Screen	13
5.4 Prefilter Screen	15
5.5 Filter Realization Screen	16
5.5.1 Z-plane design	18
5.5.2 Design Control Screen	18
5.5.3 IIR filter design via analog prototypes	19
5.5.4 Linear phase FIR filter design via classical design methods	20
5.6 Design Process Screen	21
5.7 Output file	24
6. TROUBLE SHOOTING	25
6.1 Misconvergence	25
6.2 Constraint-violation	26
7. EXAMPLES	26
7.1 Notch filter	27
7.2 IIR-Differentiator	28
7.3 Multiband filter	29
7.4 Bandpass	30
7.5 Lowpass with quasi-constant group delay	31

7.6	Lowpass with quasi-constant group delay (allpass approach)	33
7.7	Lowpass with quasi-constant group delay (using zero radius constraint)	35
7.8	Lowpass with constant group delay (FIR-approach)	37
7.9	Equalization	39
7.10	Minimum delay filter	41
7.11	Examples for IIR-design according to analog prototypes	43
7.12	Examples for linear phase FIR-design	45
8.	REFERENCES	46

1. License Agreement

You should carefully read the following terms and conditions before using this software.

Registered Version

One registered copy of Fiwiz may either be used by a single person who uses the software personally on one or more computers, or installed on a single workstation used nonsimultaneously by multiple people, but not both.

You may access the registered version of Fiwiz through a network, provided that you have obtained individual licenses for the software covering all workstations that will access the software through the network. For instance, if 8 different workstations will access Fiwiz on the network, each workstation must have its own Fiwiz license, regardless of whether they use Fiwiz at different times or concurrently.

Disclaimer of Warranty

THIS SOFTWARE AND THE ACCOMPANYING FILES ARE SOLD "AS IS" AND WITHOUT WARRANTIES AS TO PERFORMANCE OR MERCHANTABILITY OR ANY OTHER WARRANTIES WHETHER EXPRESSED OR IMPLIED. NO WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE IS OFFERED.

Good data processing procedure dictates that any program be thoroughly tested with non-critical data before relying on it. The user must assume the entire risk of using the program.

2. Glossary

BP	Bandpass
BS	Bandstop
D/A	Digital to analog
dB	Decibel
DC	Direct current
DE	Differential Evolution
FIR	Finite Impulse Response
GUI	Graphical User Interface
HP	Highpass
IIR	Infinite Impulse Response
JRE	Java Runtime Environment
LP	Lowpass

JAVA® a registered trademark of SUN Microsystems Inc.
MATLAB® a registered trademark of The Math Works Inc.
WINDOWS® a registered trademark of Microsoft Corporation
DOS® a registered trademark of Microsoft Corporation

3. Installation Instructions

These installation instructions are given in terms of WINDOWS ®based computers. Since Fiwiz is written in Java it may run on any platform that supports the Java virtual machine, and the installation has to occur accordingly.

3.1 Version which requires a Java virtual machine

- 1) Fiwiz comes packaged in compressed .zip format, so you have to unpack it first. Do so in a directory named, for example, C:\Fiwiz. In C:\Fiwiz\Fiwiz you should then find the following directory tree:

Name	Size	Type
data		File Folder
de		File Folder
panel		File Folder
plot		File Folder
problem		File Folder
ptplot		File Folder
screen		File Folder
Fiwiz.class	1KB	Java Class File

Figure 3.1-1: Directory tree for Fiwiz.

Fiwiz.class is the class containing the main() method, and hence Fiwiz.class has to be called by Java to let the application run.

- 2) In order to do this get a Java 1.1.x runtime environment (JRE), for example, from <http://www.javasoft.com> or from <http://www.icsi.berkeley.edu/~storn/fiwiz.html> and install it on your computer if you haven't done so already. Note: If you want to run Fiwiz from a batch file later on, make sure that you install the JRE in a path that exhibits directory names with a maximum of 8 characters. This limitation is due to DOS®
- 3) Let's assume the Java runtime environment resides in C:\jre_1_1\bin, then from a DOS® command prompt you can run Fiwiz by typing

```
C:\jre_1_1\bin\jre -cp C:\Fiwiz Fiwiz.Fiwiz
```

3.2 Windows ® executable

For users running the WINDOWS ®operating system an executable Fiwiz20.exe is available. The installation requires simply to unzip the file FiwizExe.zip in a directory of your choice. Then just double-click on Fiwiz20.exe to work with the program. The dynamic link libraries snjawt11.dll and snjrt11.dll may also be copied into the Windows\System directory.

4. Introduction

FIWIZ is a constraint based design program for IIR as well as FIR digital filters which is geared towards features which are difficult if at all to find in other filter design programs. The highlights of FIWIZ's capabilities are:

- **Arbitrary magnitude constraints**
This feature allows for multiband filters, differentiators, hilbert filters, sinc compensated filters and others.
- **Arbitrary group delay constraints**
Applications are mainly classical lowpass, highpass, bandpass, and bandstop filters which should exhibit approximately linear phase in the passband(s) but not necessarily in the stopband(s). This way the filter degree can often be reduced considerably compared to exactly linear phase FIR filters. Another application area where group delay constraints become important are minimum delay and fractional delay filters.
- **Minimum phase filters**
Some applications don't require any specific phase response, and hence the filter degree can be minimized by using minimum phase filters. With FIWIZ minimum phase design can be enforced by restricting the radii of the filter zeros to lie inside or on the unit circle.
- **Allpass filters**
For IIR filters the phase can be linearized using allpass filters. FIWIZ offers the possibility to linearize the phase of an existing IIR design via allpass filters. Also fractional delay filter design can take advantage of allpass filters.
- **Filter design with quantized coefficients**
FIWIZ allows to include coefficient quantization in the filter design, i.e. quantization is incorporated into the design as opposed to quantizing the coefficients after the filter has been designed with high precision coefficients. The filter structures currently supported are the direct forms 1 and 2 [Mit93] as well as first and second order sections (biquads).
- **Definition of a prefilter with constant coefficients**
Defining a prefilter has many applications like presetting specific zeroes to suppress DC or the 50/60Hz powerline frequency, accomodating filters which are already in a design and cannot be removed, or setting a frequency response to equalize. A well-known example for the latter is sinc compensation needed for D/A-conversion.
- **IIR filters with quickly decaying impulse response**
For IIR filters it is sometimes desirable to have impulse responses that are quickly fading out. FIWIZ supports such designs by allowing to set an upper limit for the pole radii.

Some other convenient features of FIWIZ are:

- **Linear phase filters**
FIWIZ is able to design linear phase filters of up to 200 taps.
- **Output of poles and zeroes**
The results file of FIWIZ contains not only the filter coefficients of the direct form 1 (or 2) [Mit93] but also the pole and zero radii as well as angles. The poles and zeroes refer to the non-quantized filter coefficients.
- **MATLAB® friendly output format**
FIWIZ's output can be directly posted on to MATLAB's ® command line interface for further analysis.

- **Storage and retrieval of configuration files**
The settings of constraints and design parameters can be stored and retrieved so that there remains only little retyping if a previous filter design shall be altered.
- **Platform independence through JAVA® technology**
FIWIZ has been written completely in JAVA® and hence it runs on any platform which supports the JAVA® virtual machine.
- **Wizard based approach**
FIWIZ's wizard based approach makes using FIWIZ almost self-explanatory. The sequence of operations is evident.

In version 2.0 of FIWIZ the following design extensions have been made:

- **IIR-design via analog prototypes**
FIWIZ is able to design Butterworth, Chebyshev1, Chebyshev2, and Elliptic IIR-filters which make use of the bilinear Z-transform.

In version 2.3 of FIWIZ the following design extensions have been made:

- **Linear phase FIR-design via classical design methods**
FIWIZ is able to design equiripple and maxflat FIR filters with linear phase.

5. Usage of FIWIZ

5.1 Startup Screen

Figure 5.1-1 shows the startup screen of FIWIZ. ^{1.1} provides the possibility to load a properly formatted configuration file by which all important design parameters of a previous filter design can be loaded. These design parameters show up in the following screens. Whenever you want to save a certain parameter setting during a filter design session you have to go (back) to the startup screen and save the configuration using the Save As button ^{1.2}.

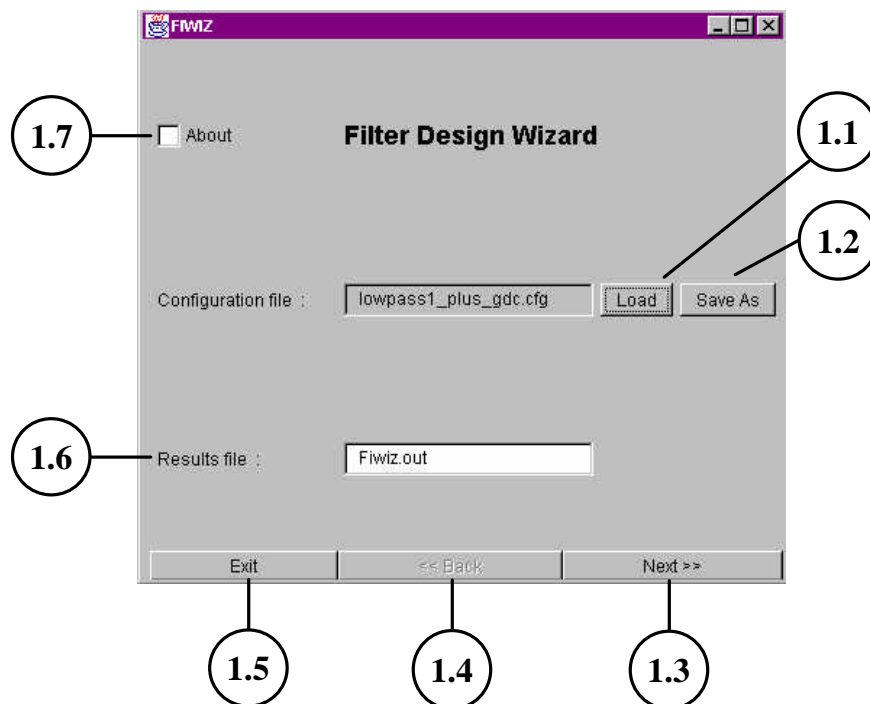


Figure 5.1-1: Startup Screen.

The results of your filter design will be poles and zeroes as well as the coefficients for a digital filter. In the results file textfield ^{1.6} you can type in the name of the file which you want your results to be stored in. The default filename is Fiwiz.out.

With the next button ^{1.3} you can proceed to the next screen, in this case the magnitude screen. The back button ^{1.4} is deactivated in the startup screen since there exists no previous screen. The exit button ^{1.5} serves to exit FIWIZ immediately without doing any save operation. The “About” check box ^{1.7} provides information about FIWIZ’s origin and the current version number.

5.2 Magnitude Screen

The magnitude screen shown in Figure 5.2-1 serves to define magnitude constraints in dB for

the filter under design. The constraints list 2.10 shows the currently active values for the upper and lower magnitude constraints. Whether the list for the upper or the one for the lower constraints is shown depends on the state of the radio buttons 2.1 and 2.2.

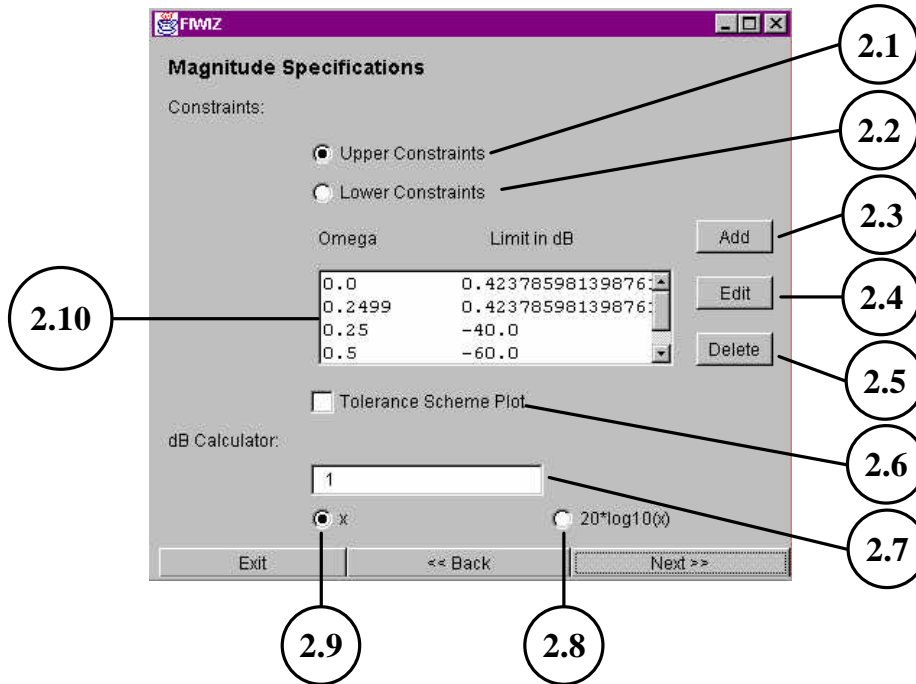


Figure 5.2-1: Magnitude Screen.

Thanks to the graphics capabilities provided by PtPlot [PtPlot00] the entire constraints can also be viewed graphically as shown in Figure 5.2-2. The graphics can be activated by using the

checkbox 2.6 and can be helpful when checking the list entries for correctness.

In order to add a new constraint to the list the add button 2.3 must be pressed. To modify an

existing entry the edit button 2.4 must be used after the list item has been selected with the mouse. For the deletion of an entry go to the appropriate line in the list and press the delete

button 2.5.

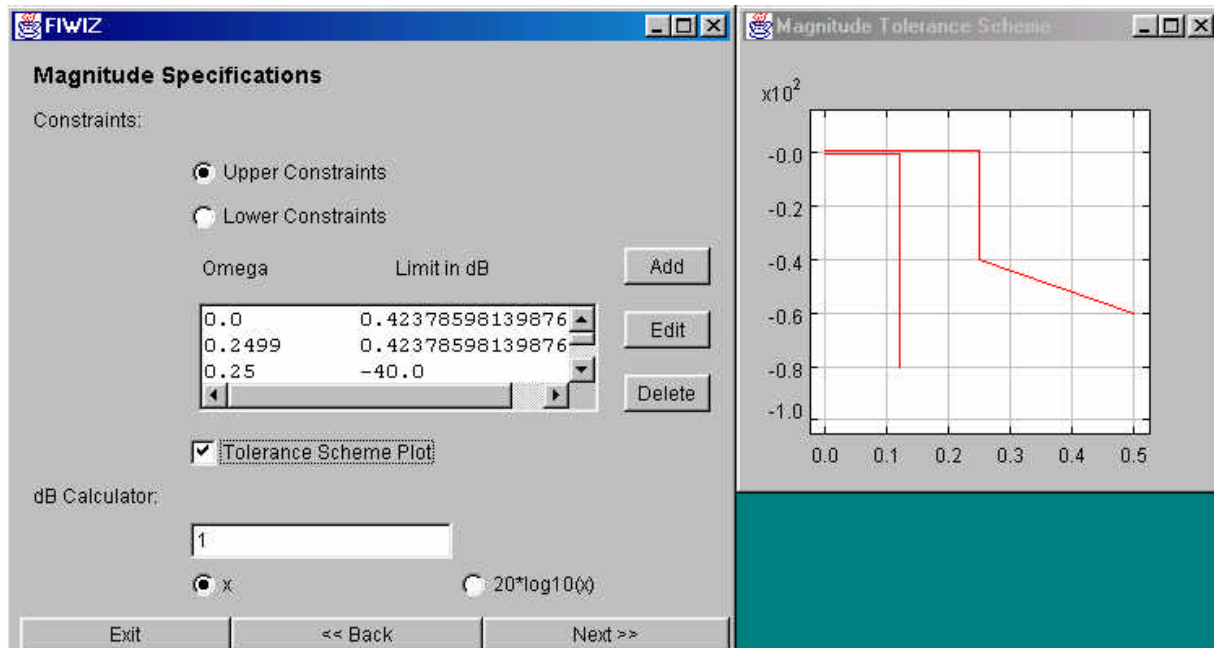


Figure 5.2-2: The magnitude constraints can also be viewed graphically.

Whenever the add or edit button is used the magnitude screen will change appearance according to Figure 5.2-3.

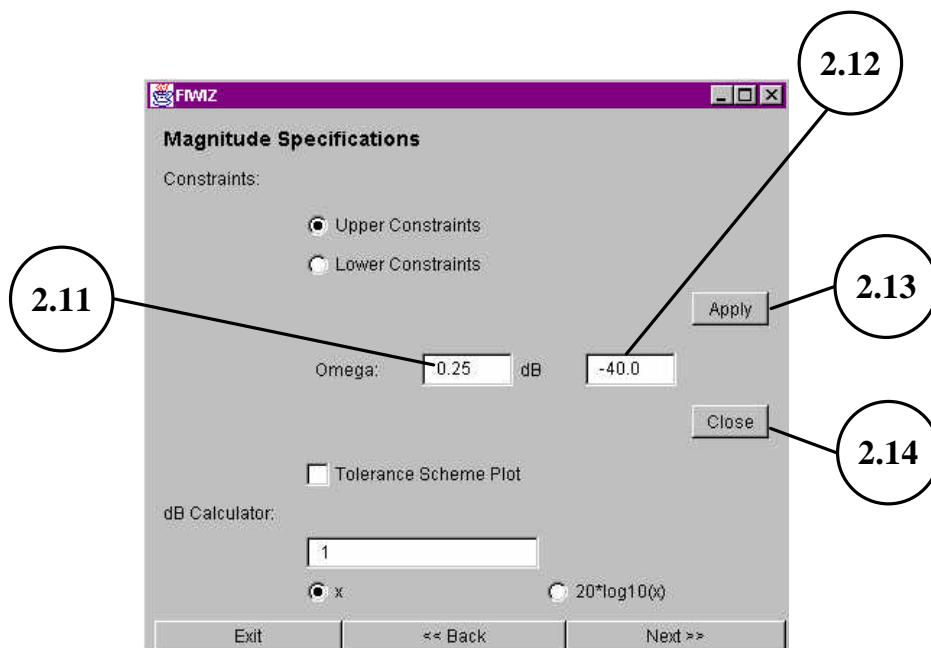


Figure 5.2-3: Magnitude screen after pressing either the add or the edit button.

In the omega text field ^{2.11} the normalized frequency

$$\Omega = \frac{w}{w_s} = \frac{2p \cdot f}{2p \cdot f_s} \in [0, 0.5]$$

can be entered. In this context f denotes the natural frequency and f_s represents the sampling

frequency. Text field 2.12 takes the constraint value in dB. If you decide to move your

constraint to the constraints list simply hit the apply button 2.13, otherwise hit the close

button 2.14. Note that no frequency value may appear twice in the constraints list, i.e. strictly vertical lines in the constraints graph are not possible. Of course, the difference between two adjacent frequency values in the list can be made very small in order to approximate a vertical line in the constraints graph.

Since all points in a constraints list are connected by straight lines arbitrary constraint shapes can be generated. Note, however, that it is prudent to keep the constraints list as short as possible since a long list will lead to longer filter design times. The longer design time results from the increased number of points along the frequency axis that has to be checked by FIWIZ.

As a convenience a dB calculator using the radio buttons 2.8 and 2.9 has been added to the magnitude screen so linear magnitude values can easily be transformed into dB values and vice versa.

The next button will lead to the group delay screen.

5.3 Group Delay Screen

The group delay shown in Figure 5.3-1 allows you to specify group delay constraints. Using group delay constraints for IIR filter design can be an interesting alternative to linear-phase FIR filters especially if the filter degree shall be kept low and quasi-linearity of the phase is required only in the passband(s) of a filter.

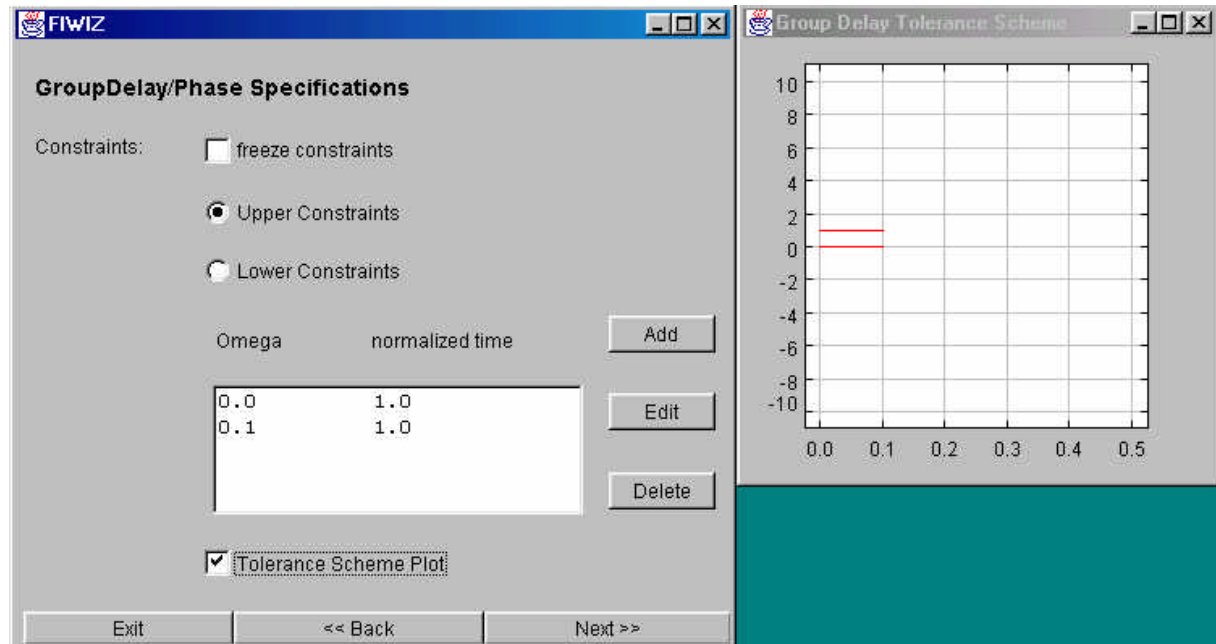


Figure 5.3-1: Group delay screen.

The operation of the group delay screen works just the way it worked for the magnitude screen. The constraints are provided in normalized time

$$T = \frac{t}{t_s} = t \cdot f_s$$

In contrast to magnitude constraints, however, the group delay constraints will be floating if the „freeze constraints“ check box is not checked. This means that an arbitrary but constant group delay T_0 might be added to the given constraints by the filter design procedure as indicated in Figure 5.3-2.

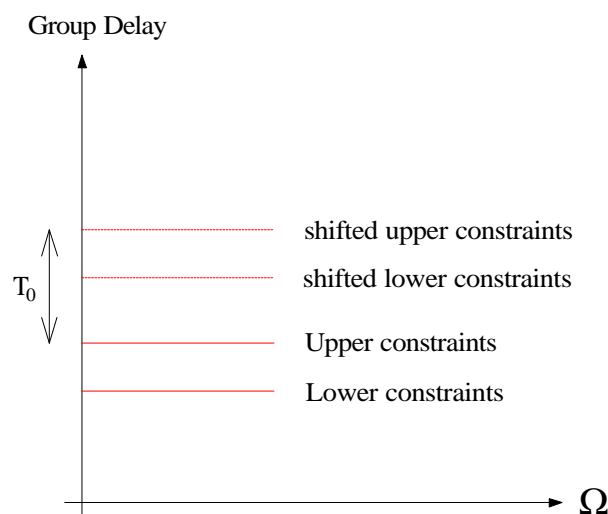


Figure 5.3-2: The filter design procedure has the freedom to shift the group delay tolerance scheme along the ordinate.

If, on the other hand, the „freeze constraints“ check box is checked the constraints are frozen, i.e. they are not allowed to float. This feature can be necessary if minimum delay filters or fractional delay filters shall be designed.

5.4 Prefilter Screen

The prefilter screen as shown in Figure 5.4-1 provides an entry form to define a prefilter with constant coefficients. Setting a prefilter has many applications like setting specific poles or zeroes, e.g. a zero at frequency 0 for DC removal, or setting a zero to filter out the 50/60Hz frequency of a powerline. The prefilter may also accommodate filters which occur in a given design and cannot be removed. Another application is equalization where the prefilter defines the frequency response to be compensated. Sinc compensation as required for D/A-conversion is a well-known and common application for equalization.

Figure 5.4-1: Prefilter Screen.

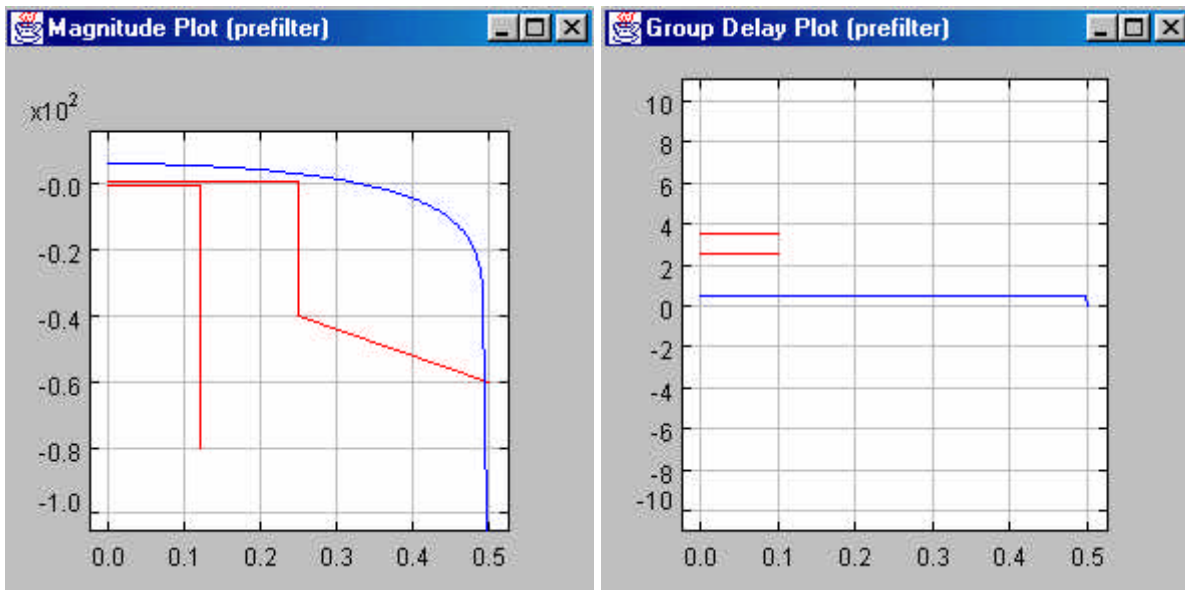


Figure 5.4-2: Magnitude and group delay plots for the prefilter.

5.5 Filter Realization Screen

The filter realization screen allows for the exact specification of the transfer function

$$H(z) = A_0 \frac{\prod_{n=0}^{N-1} (z - z_{0n})}{\prod_{m=0}^{M-1} (z - z_{pm})}, \quad z = e^{j\Omega}, \quad j = \sqrt{-1}$$

FIWIZ takes care that all zeroes z_{0n} and all poles z_{pm} are either real or have a conjugate

complex partner. Choice control **(4.0)** allows to select a design method for the tolerance scheme set up in the previous screens. Currently the only design method offered by FIWIZ is direct design in the Z-domain which is done via Differential Evolution (see section 5.5.2).

(4.1) takes the number of zeroes which are not subject to any phase constraint. **(4.2)** and

(4.3) specify the lower and the upper limit of the zero radii respectively. This can be very

helpful, e.g. when minimum phase filters shall be designed. In this case **(4.2)** may be set to 0

while **(4.3)** must be set to 1 to prevent the zeros from lying outside the unit circle. If, for some

applications all zeroes shall lie on the unit circle, both limits in **(4.2)** and **(4.3)** must be set to 1.

In an analogous way **(4.4)** specifies the maximum pole radius allowed. Any value >1 may result in unstable filters and hence is not useful in general. However, values smaller than 1 may prove efficient if the impulse response of an IIR-filter shall be as short as possible. The farther the poles are apart from the unit circle the shorter the impulse response gets.

(4.5) takes the number of poles, and **(4.6)** sets the number of linear-phase zero pairs, i.e. for the latter the corresponding filter degree is always twice the number entered. The following design rules should be adhered to:

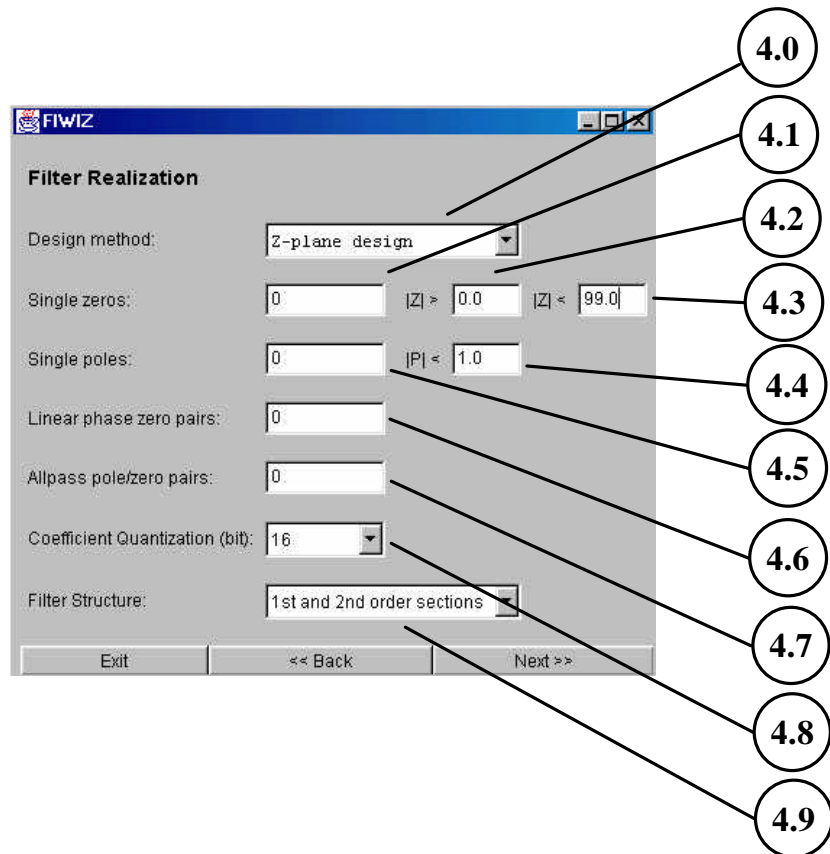


Figure 5.5-1: The filter realization screen defines the makeup of the filter transfer function.

- 1) When a minimum phase filter shall be designed (4.3) must be set to 1. In addition the entries in (4.6) and (4.7) must be zero.
- 2) In case of a linear phase FIR filter design (4.1), (4.5) and (4.7) should be set to zero.
- 3) In case of an IIR filter design with almost constant delay in the passband(s) it is sometimes helpful to not only use entries in (4.1) and (4.5) but also choose some linear phase zero pairs in (4.6). The linear phase zero pairs facilitate the design since some linearity is already introduced by design.
- 4) Another approach to quasi-linear phase IIR filter design is to first design a low degree IIR filter with magnitude constraints only. This result will then be input into the prefilter section. Afterwards an allpass filter is designed via (4.7) in order to linearize the phase.
- 5) Yet a third approach for quasi-linear phase IIR filter design is to prevent zeroes from being inside the unit circle by setting the lower limit of the zero radii (4.2) to 1. This way, the zeroes will be forced onto or outside the unit circle where their phase steering properties can be exploited. This strategy does not always work, but nevertheless is often worth a try.

Note that the maximum entry in any of the text fields of the filter realization screen is currently limited to 100.

Choice control **4.8** takes the number of bits (including the sign bit) which all coefficients of the filter structure shall be restricted to. Note that this information is incorporated into the design procedure and is not just done after a design with high precision coefficients.

Choice control **4.9** allows to choose the filter structure defining the set of coefficients. Currently the direct form structures (1 and 2) as well as first and second order sections are supported.

5.5.1 Z-plane design

Z-plane design is the primary focus of FIWIZ. It allows to specify arbitrary magnitude constraints, group delay constraints, and the inclusion of a prefilter. In order to cope with these multiple constraints FIWIZ employs a genetic algorithm called Differential Evolution or briefly DE [Sto00] for filter design. A few control parameters for DE are offered to enable a filter design with maximum success. These control parameters are accessible in the Design Control Screen.

5.5.2 Design Control Screen

The next screen is the design control screen which is depicted in Figure 5.5-2. The global optimization capability and flexibility of DE allows for much freedom with respect to the constraints in magnitude and group delay. In quite a few cases Fiwiz allows you to design filters which are very difficult if not impossible to design even with expensive filter design tools.

This flexibility and freedom have to be paid for by increased filter design times compared to traditional filter design methods. In addition, as with every genetic optimizer, convergence cannot be guaranteed. However, DE usually does a very good job when it comes to filter design.

Since genetic algorithms are population based there is a text field **5.1** which takes the number NP of population members. In contrast to most other genetic algorithms DE allows this number to be fairly small. In general

$$NP = 2 * (\text{number of all poles} + \text{number of all zeroes})$$

is a reasonable value to start with. In many cases an even lower number for NP suffices.

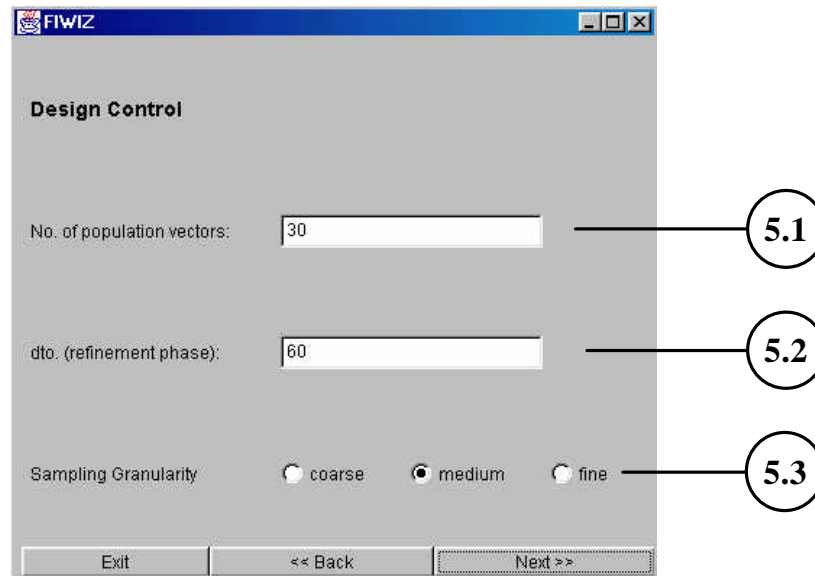


Figure 5.5-2: Design Control Screen.

The text field ^{5.2} takes the number NP2 of population members for the refinement phase of the design procedure. As a general rule NP2 should be the higher the smaller the number of coefficient-bits is chosen. A good estimate for NP2 is

$$NP2 = NP + 500 \cdot \left(\frac{32 - \text{bits}}{32} \right)$$

Also here NP2 may be chosen much smaller depending on the difficulty of the task at hand. In general NP as well as NP2 are restricted to 1000.

The last control variable in the design process is the sampling density ^{5.3} along the frequency axis. It is possible to choose from coarse, medium and high density where the default is coarse. A coarse sampling density tests the least number of points for compliance with the constraints chosen and hence the filter design proceeds faster as if medium or high were chosen. However, it might be that a coarse sampling grid misses out on some frequency values where the constraints are violated.

5.5.3 IIR filter design via analog prototypes

For many applications just the standard filter types lowpass (LP), highpass (HP), bandpass (BP), or bandstop (BS) need to be defined. In these cases the design can occur much faster if the design path is employing analog filter prototypes and the bilinear Z-transform [Rab75]. FIWIZ can design according to the following characteristics: Butterworth, Chebyshev1, Chebyshev2 (inverse Chebyshev), and Elliptic. These characteristics may also serve as an estimate for the filter degree that is going to be used for a Z-plane design. In order to design an LP, HP, BP, or BS via analog prototypes the passband(s) and the stopband(s) of the tolerance scheme must be totally flat. It is best to specify the tolerance schemes according to the templates indicated in Figure 5.5-3.

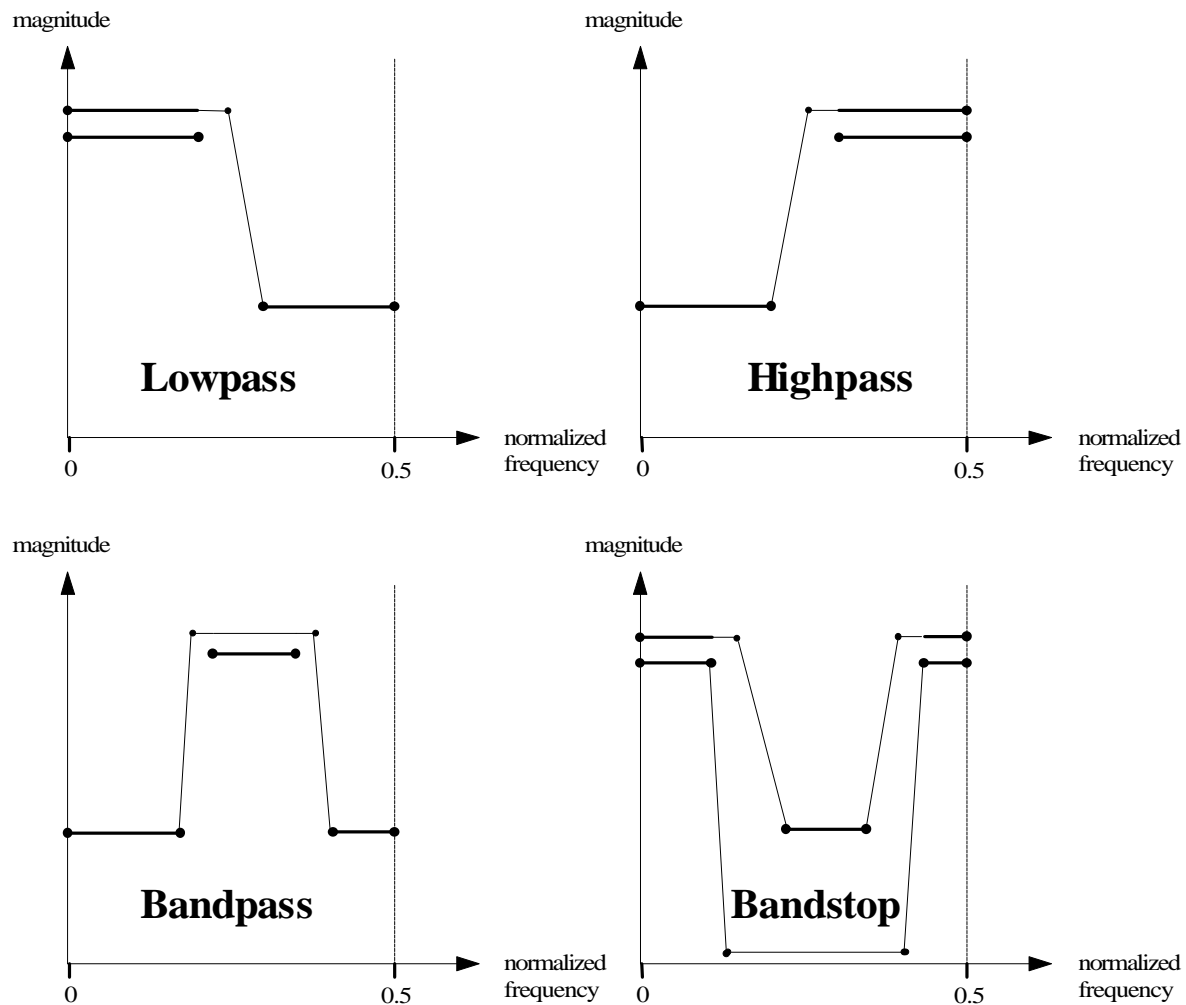


Figure 5.5-3: Tolerance scheme templates for the four standard filter types. Note that the passband and stopband has to be exactly horizontal and that the stopbands for the BP (and the passbands for the BS) must have the same gain values.

When designing filters using analog prototypes the following has to be noted:

- 1) All constraints entered in the group delay section will simply be disregarded.
- 2) All input in the prefilter section will simply be disregarded.
- 3) Only the thick lines in Figure 5.5-3 determine the filter specifications. The thin lines, however, still exist because of the tolerance schem input paradigm of FIWIZ (upper constraints must all be connected, lower constraints must all be connected). One can see that the transition bands are completely defined by the corner frequencies of passband and stopband, i.e. the slanted lines do not really define a constraint.
- 4) Passband and stopband constraints must be exactly horizontal.
- 5) Passbands for bandstop filters must have the same gain values.
- 6) Stopbands for bandpass filters must have the same gain values.



5.5.4 Linear phase FIR filter design via classical design methods

Linear phase FIR filters have a wide range of applications. Although FIWIZ is able to design these filters directly in the Z-domain, classical filter design methods like Parks-McClellan and linear programming [Mit93] yield these results in a much shorter time. Therefore FIWIZ now offers linear phase FIR filter design via classical methods. Currently these filters are restricted






to the classical types LP, HP, BP, and BS as described in chapter 5.5.3. The maximum number of coefficients is currently set to 256.


5.6 Design Process Screen

The last screen of Fiwiz is the design process screen shown in Figure 5.6-1. Beginning and end


of the filter design can be activated with the Start/Stop-Button  .  serves as a toggle button, i.e. once start is pressed the next press will stop the design. The filter design can also



be paused with the Pause/Resume-Button  which also is a toggle button. The monitor


values , , , , and  show the filter order, the elapsed number of DE-generations, elapsed number of filter function evaluations, current best cost function value, and state of the optimizer respectively. When the filter is designed successfully the cost function

value of the refinement phase will be smaller than 1.e-12 and the state  will return to “Ready”. As stated before even a cost value of zero does not guarantee that the filter meets all the constraints if the sampling grid along the frequency axis was not chosen dense enough.

If the cost value arrives at zero or the design process is stopped with the Start/Stop-Button the filter parameters achieved are printed into the results file specified in the startup screen of Figure 5.1-1. The printout is in a MATLAB ® friendly format, i.e. the results can be copied directly into MATLAB’s ® command line interface for further analysis. An online plot of

magnitude, zeroes and poles, and group delay can be invoked with the checkboxes ,

, and  respectively. This may be done during the design process or after its

completion. With  one can choose the refresh rate which defines the number of generations after which the plots will be updated. Note that magnitude and group delay plot comprise the entire filter (including prefilter with constant coefficients) while the pole/zero plot displays only the variable poles and zeroes.

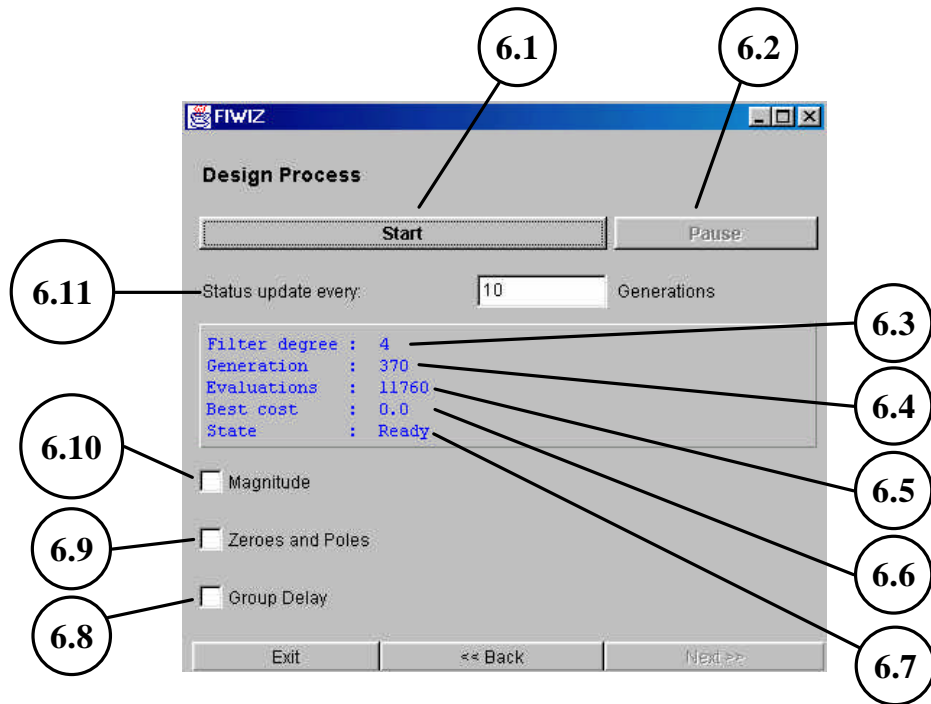


Figure 5.6-1: Design process screen steers the actual filter design process.

The plotters make use of PtPlot [PtPlot00] from the University of Berkeley, CA and allow for zooming in and out. Zooming in is done by mouse-drawing a rectangular box from left upper to right lower corner around the region of interest. Zooming out is done in a reverse manner by drawing a rectangular box from a lower right to an upper left corner. Figure 5.6-2 and Figure 5.6-3 give an example of the plotting capabilities provided by Fiwiz.

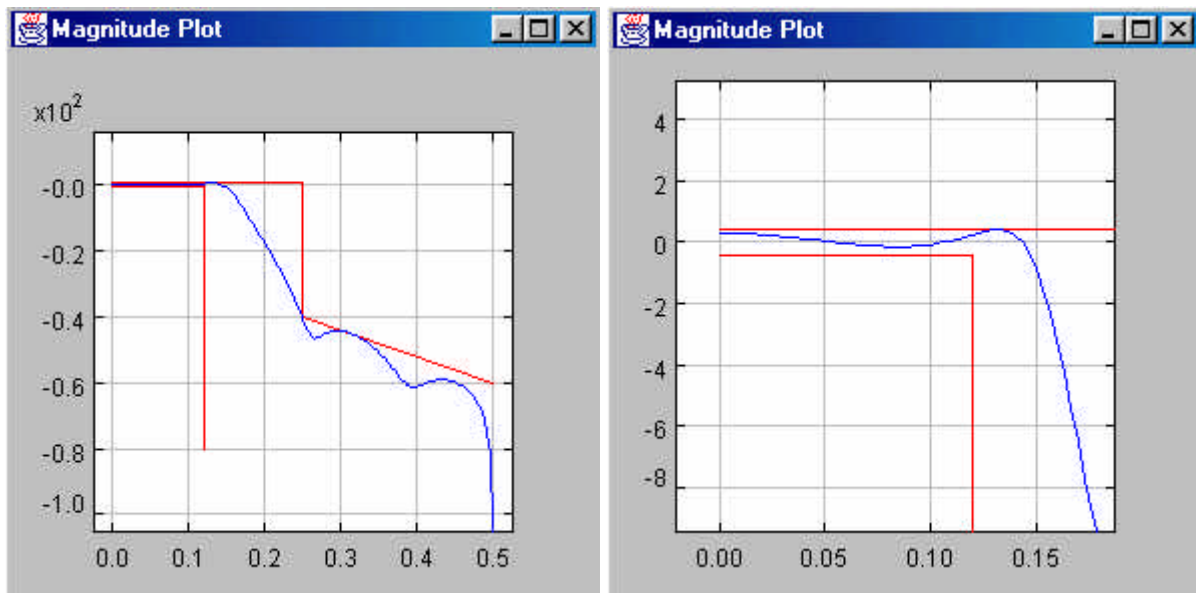


Figure 5.6-2: Example plot of magnitude in normal and zoomed-in version.

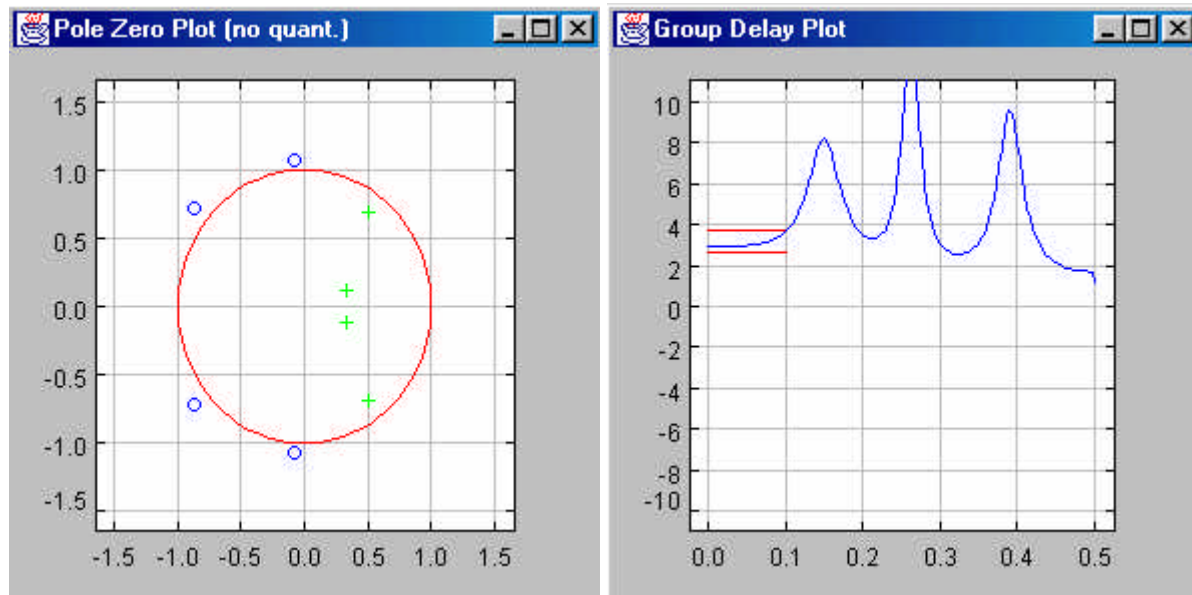


Figure 5.6-3: Example plots for zeroes and poles as well as group delay.

In the pole/zero-plot poles are indicated by + signs while zeroes are indicated by circles.

5.7 Output file

The format of the output file is designed such that the results can be readily copied into a MATLAB ® control window for further processing or verification. See below an example of the output file:

```
%=====Optimization characteristics=====
Number of function evaluations = 3600
Best cost function value      = 0.0

%=====Parameters for unquantized coefficients=====
a0 = 0.01845878824745417

%-----Zero radii-----
zr(0)=1.1300053080000325
zr(1)=1.0782123814583506

%-----Zero angles-----
za(0)=-21.304271406040076
za(1)=-17.201729188021265

%-----Pole radii-----
pr(0)=0.346671275605549
pr(1)=-0.847635342528437

%-----Pole angles-----
pa(0)=-19.18403167457911
pa(1)=-4.0822614371311445

%=====First and second order sections=====
a0=[1.0 1.74749755859375 1.276885986328125];
a1=[1.0 0.165924072265625 1.16253662109375];
k=0.01845878824745417;
b0=[1.0 -0.654937744140625 0.12017822265625];
b1=[1.0 -0.99896240234375 0.718475341796875];

%=====Direct form #2 coefficients=====
%-----numerator coefficients (MATLAB format)-----
a(1)=0.01845878824745417;
a(2)=0.03531944471213162;
a(3)=0.05038094628716659;
a(4)=0.04141037226939837;
a(5)=0.02740071849459553;

%-----denominator coefficients (MATLAB format)-----
b(1)=1.0;
b(2)=-1.653900146484375;
b(3)=1.49291174672544;
b(4)=-0.5906101455911994;
b(5)=0.08634508959949017;

%=====Direct form #2 coefficients for fixed filter=====
%-----numerator coefficients (MATLAB format)-----
af(1)=1.0;
af(2)=1.0;
```

Below is an example script file in MATLAB ® which can be used to verify the results of the above output file:


```
% script to plot FIWIZ's results

%====Direct form #2 coefficients for fixed filter=====
%----numerator coefficients (MATLAB format)-----
af(1)=1.0;
af(2)=1.0;

%====First and second order sections=====
%--Remark: pairing of poles and zeros can be chosen as-----
%--required, for example according to Jackson's minimum-----
%--noise scheme.-----
a0=[1.0 1.74749755859375 1.276885986328125];
a1=[1.0 0.165924072265625 1.16253662109375];
k=0.01845878824745417;
b0=[1.0 -0.654937744140625 0.12017822265625];
b1=[1.0 -0.99896240234375 0.718475341796875];

%---Compute resulting filter-----
a_final = k*conv(conv(af,a0),a1);
b_final = conv(b0,b1);

fg=0;
fg = fg+1;
figure(fg);
zplane(a_final,b_final);
zoom on;

fg = fg+1;
figure(fg);
freqz(a_final,b_final);
zoom on;

fg = fg+1;
figure(fg);
grpdelay(a_final,b_final);
zoom on;
```

6. Trouble Shooting

6.1 Misconvergence

As has been stated before Fiwiz doesn't guarantee to converge although convergence is likely if a solution exists. If the latter is assured, possible remedies are:

1. Simply rerun the design in the design process screen. Although DE can handle local optima fairly well it still might get trapped in one once in a while.
2. Increase the number of population members. Too low a number might prevent that the parameter space is sampled thoroughly enough.
3. Especially in IIR-filter design where quasi-linearity of phase (i.e. quasi constant group delay) in the passbands is desired, it is often advantageous for the optimizer if some of the zeroes are from the lin-phase zero pairs section in the filter realization screen.

If you are unsure about the feasibility of your design project try to get some estimate of the filter degree. If that is impossible work your way up from lower to higher degree filters. Fiwiz will yield a best fit in the least squares sense, so even when Fiwiz can't find a solution in the first few runs you get valuable information about the problem at hand.

6.2 Constraint-violation

If the filter design of Fiwiz converges but the resulting filter still violates the constraints set forth in the magnitude and/or group delay screens the sampling density in the design control screen has been chosen too low. If the violation still occurs with even the highest sampling density the constraints have to be divided into more segments. Figure 6.2-1 provides an example where a constant magnitude constraint of x dB between the normalized frequencies Ω_1 and Ω_2 is altered as the sequence of two constraints of x dB between the frequencies Ω_1 and Ω_3 as well as Ω_3 and Ω_2 respectively. As Fiwiz applies its sampling density to each segment individually the procedure outlined increases the overall sampling density.

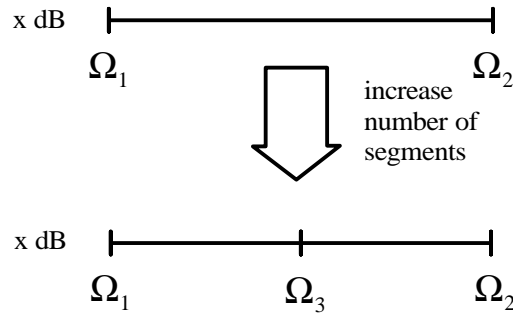


Figure 6.2-1: Example for increasing the number of constraint segments.

For the sake of shortening the design time it often is even advisable to resort to the technique shown in Figure 6.2-1 for critical frequency intervals rather than switching to the next higher sampling density.

7. Examples

The following examples shall provide a better idea of Fiwiz's capabilities and usage. The examples are by no means exhaustive.

7.1 Notch filter

Filter Realization

Design method: 2-plane design

Single zeros: 2 $|Z| =$ 1.0 $\angle Z =$ 90.0

Single poles: 2 $|P| =$ 1.0

Linear phase zero pairs: 0

Allpass pole/zero pairs: 0

Coefficient Quantization (bit): 16

Filter Structure: Direct Form

Exit
<< Back
Next >>

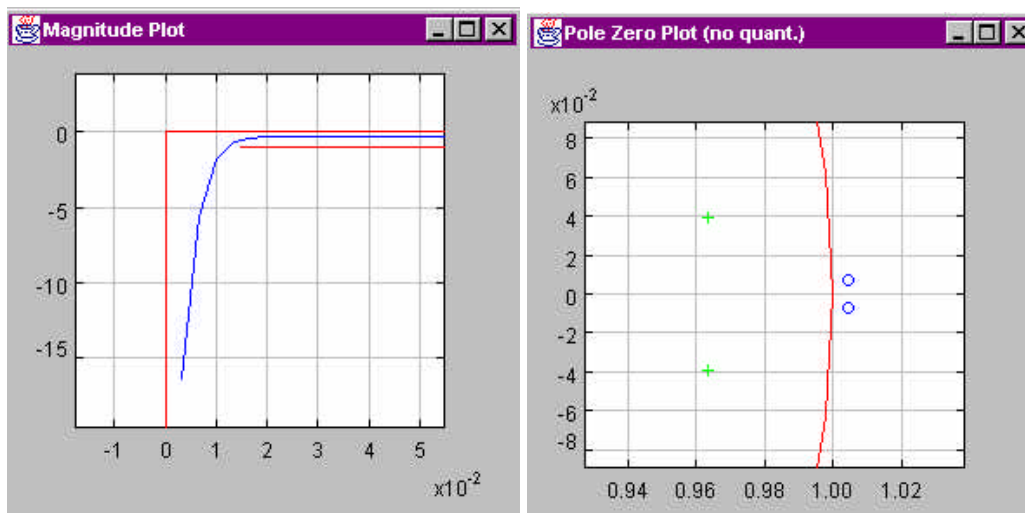
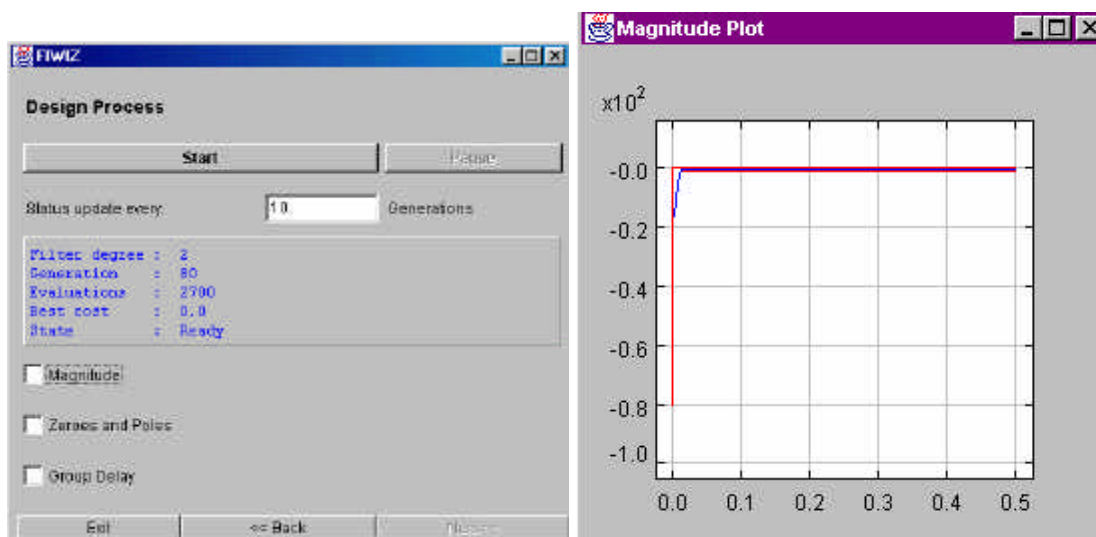
Design Control

No. of population vectors: 30

div. (refinement phase): 60

Sampling Granularity: ☒ coarse ☐ medium ☐ fine

Exit
<< Back
Next >>



7.2 IIR-Differentiator

Filter Realization

Design method: Z-plane design

Single zeros: 3 $|Z| <$ 0.0 $|Z| >$ 99.0

Single poles: 4 $|P| <$ 1.0

Linear phase zero pairs: 0

Allpass pole/zero pairs: 0

Coefficient Quantization (bit): 16

Filter Structure: Direct Form

Exit << Back Next >>

Design Control

No. of population vectors: 30

sto. (refinement phase): 50

Sampling Granularity: ☒ coarse ☐ medium ☐ fine

Exit << Back Next >>

Design Process

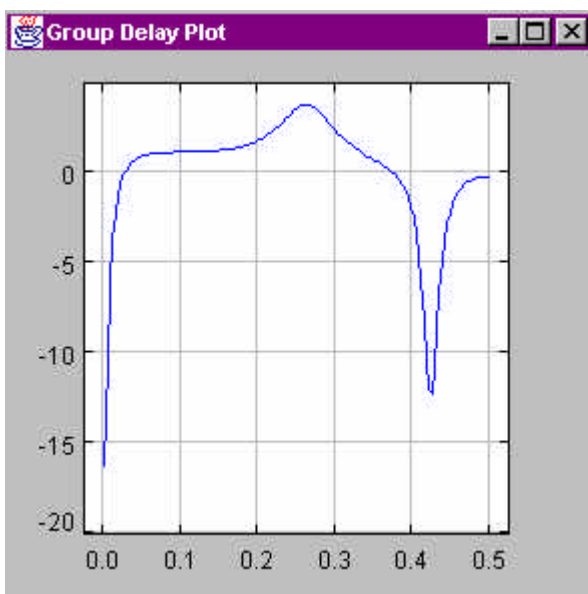
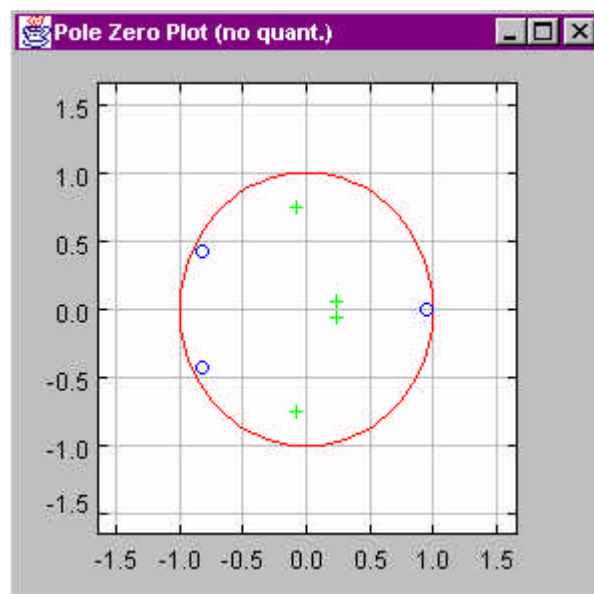
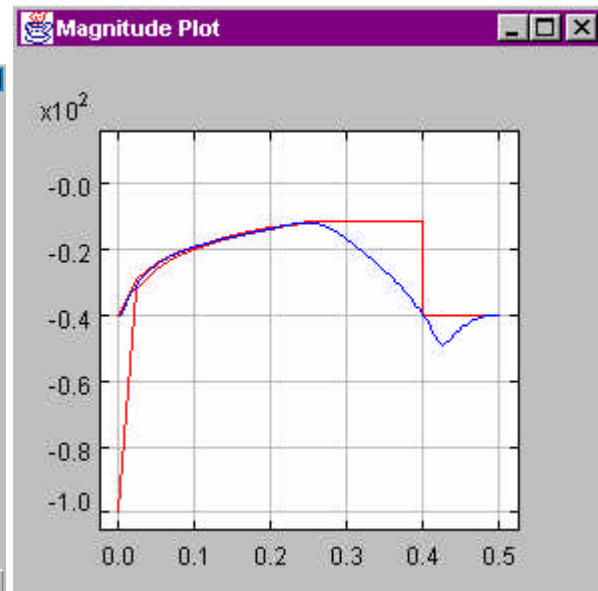
Start Pause

Status update every: 10 Generations

Filter degree : 4
Generation : 130
Evaluations : 4200
Best cost : 0.0
State : Ready

☐ Magnitude
☐ Zeroes and Poles
☐ Group Delay

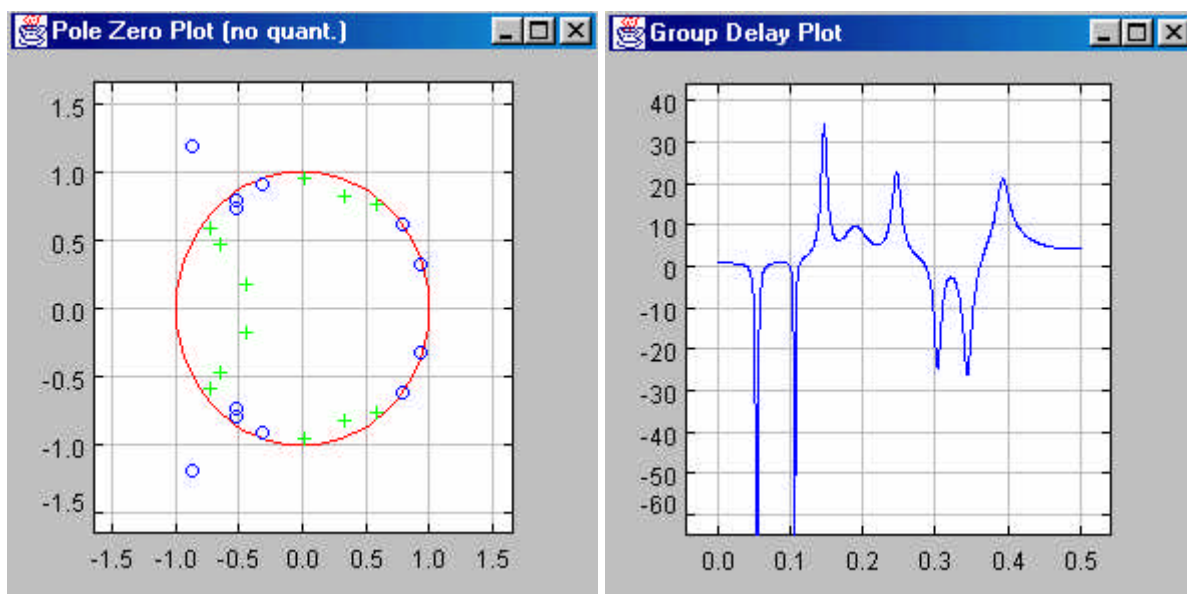
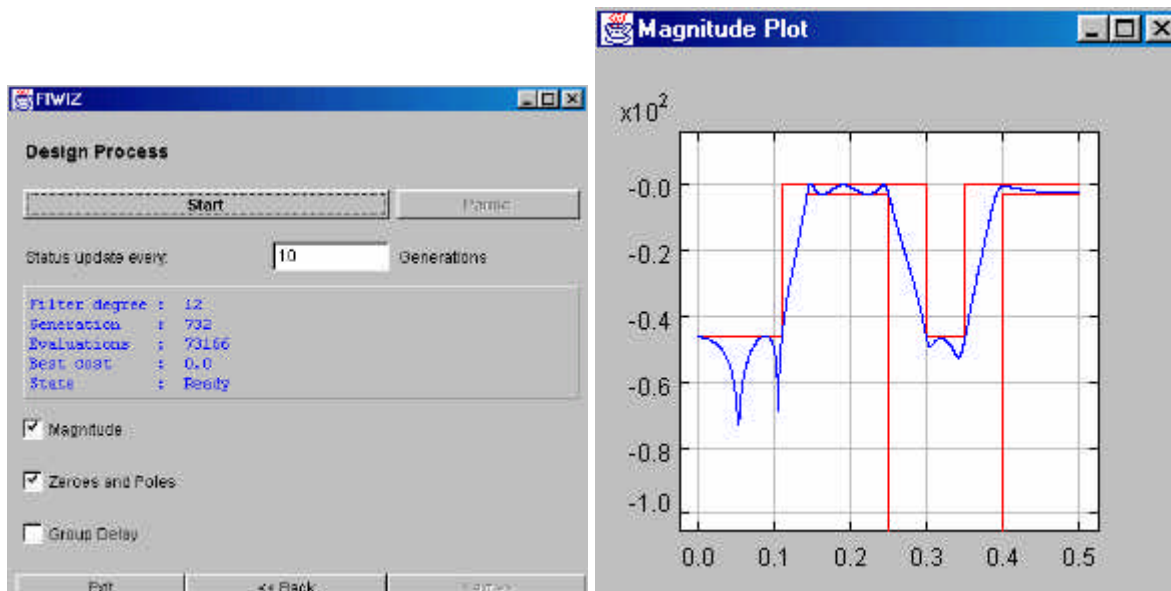
Exit << Back Next >>



7.3 Multiband filter

Filter Realization
Design method: Z-plane design
Single zeros: 12 $|z| > 0.0$ $|z| < 99.0$
Single poles: 12 $|p| = 1.0$
Linear phase zero pairs: 0
Allpass pole/zero pairs: 0
Coefficient Quantization (bit): 12
Filter Structure: Direct Form

Design Control
No. of population vectors: 100
Ita. (refinement phase): 100
Sampling Granularity: ☒ coarse ☐ medium ☐ fine



7.4 Bandpass

Filter Realization

Design method:

Single zeros: $|z| > 0.0$ $|z| < 99.0$

Single poles: $|p| < 1.0$

Linear phase zero pairs:

Allpass pole/zero pairs:

Coefficient Quantization (bit):

Filter Structure:

Exit << Back Next >>

Design Control

No. of population vectors:

cto. (refinement phase):

Sampling Granularity: ☒ coarse ☐ medium ☐ fine

Exit << Back Next >>

Design Process

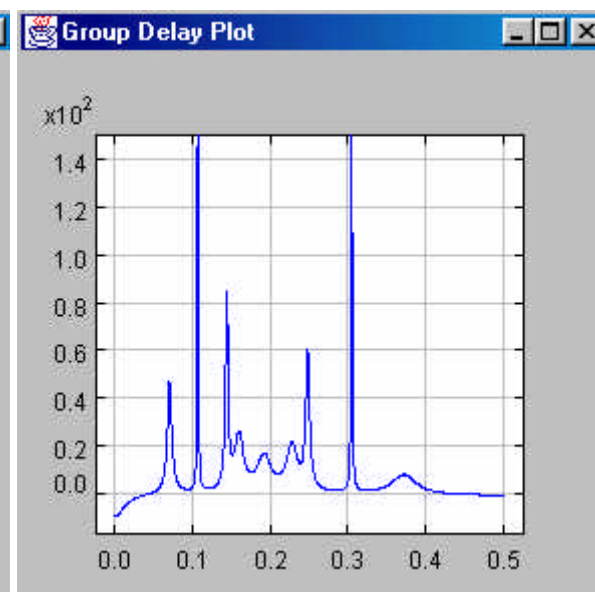
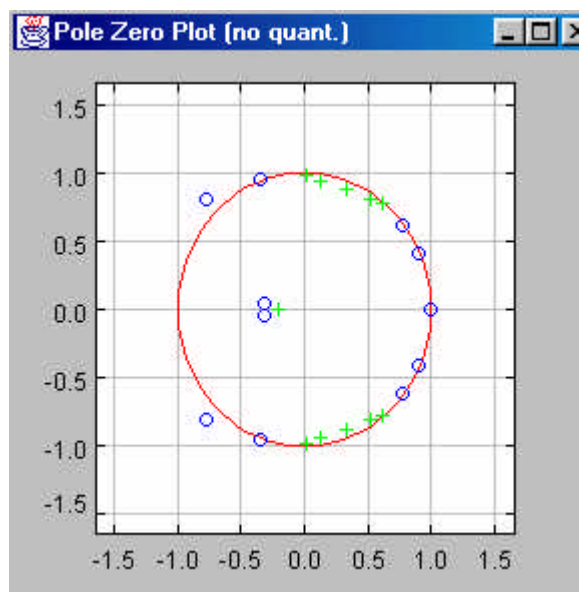
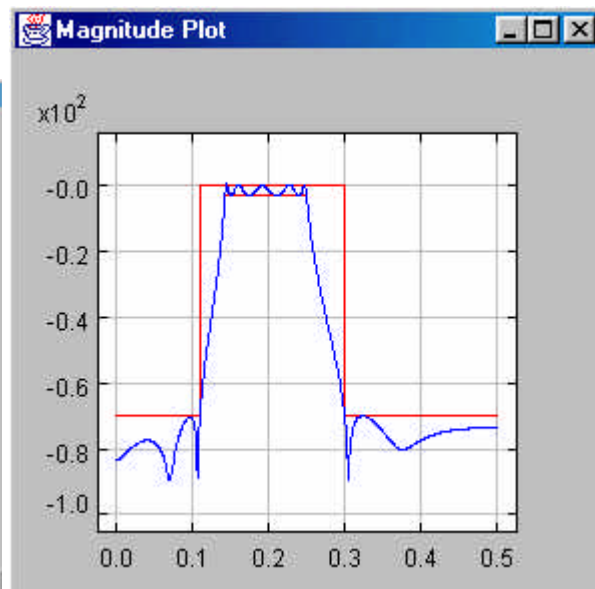
Start Pause

Status update every: Generations

Filter degree : 11
Generation : 2360
Evaluations : 283320
Best cost : 0.0
State : Ready

☐ Magnitude
☐ Zeros and Poles
☐ Group Delay

Exit << Back Next >>



7.5 Lowpass with quasi-constant group delay

Filter Realization
Design method:
Single zeros: $|Z| > 0.0$ $|Z| < 99.0$
Single poles: $|P| < 1.0$
Linear phase zero pairs:
Allpass pole/zero pairs:
Coefficient Quantization (bit):
Filter Structure:
Exit << Back Next >>

Design Control
No. of population vectors:
cto (refinement phase):
Sampling Granularity: ☐ coarse ☐ medium ☒ fine
Edit << Back Next >>

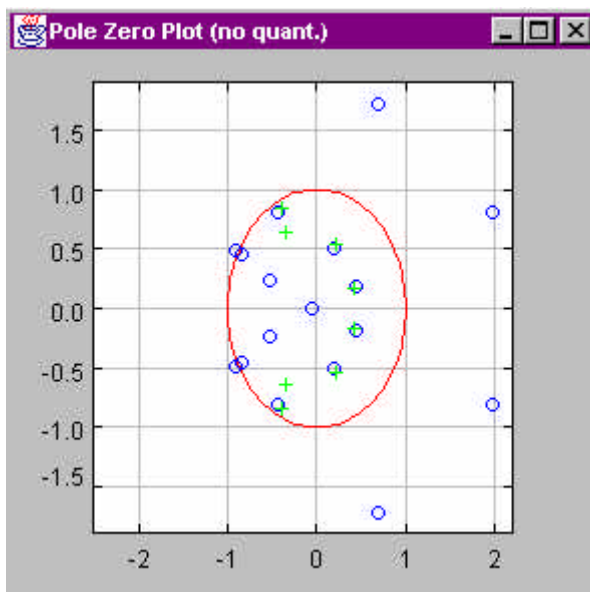
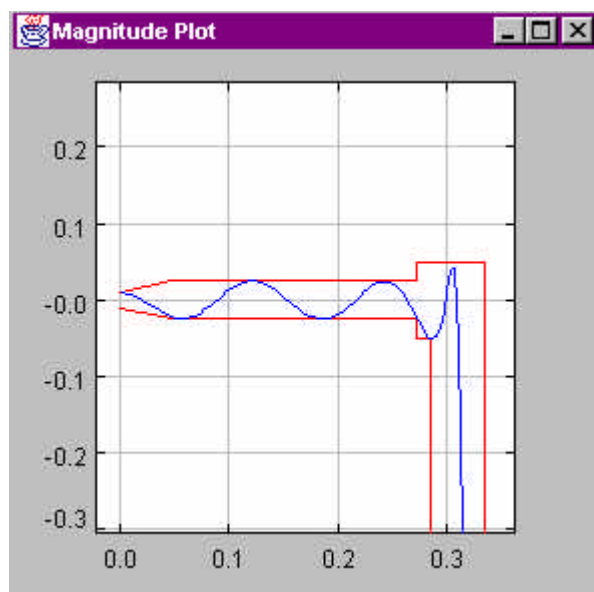
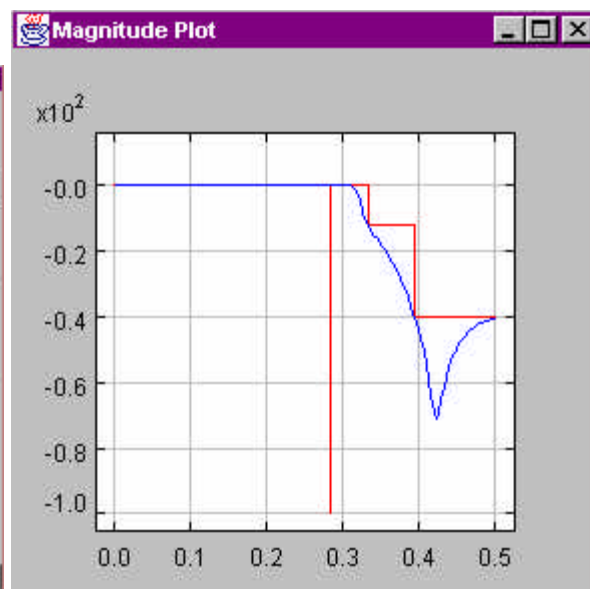
Design Process
Start Pause
Status update every: Generations

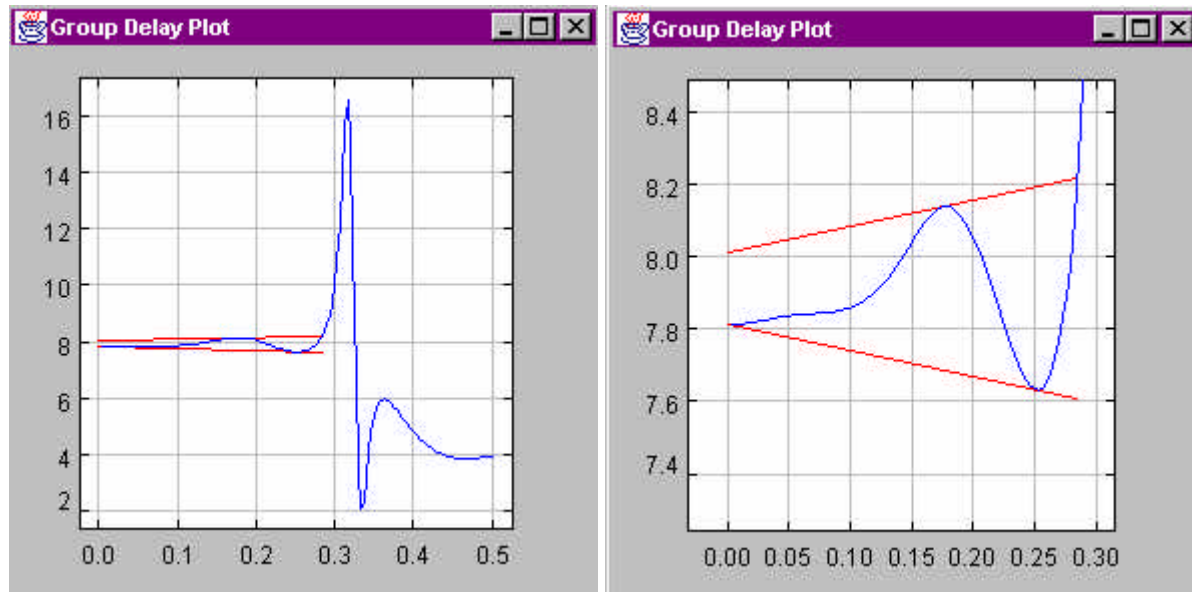
```

Generation : 6550
Evaluations: 197100
Best cost   : 0.0
State      : Ready

```

☐ Magnitude
☐ Zeros and Poles
☐ Group Delay
Exit << Back Next >>





Note: Screenshots have been done with FIWIZ version 1.6.

7.6 Lowpass with quasi-constant group delay (allpass approach)

Filter Realization

Design method: Z-plane design

Single zeros: 0 $|Z| > 0.0$ $|Z| < 99.0$

Single poles: 0 $|P| < 1.0$

Linear phase zero pairs: 0

Allpass pole/zero pairs: 6

Coefficient Quantization (bit): inf

Filter Structure: Direct Form

Exit
<< Back
Next >>

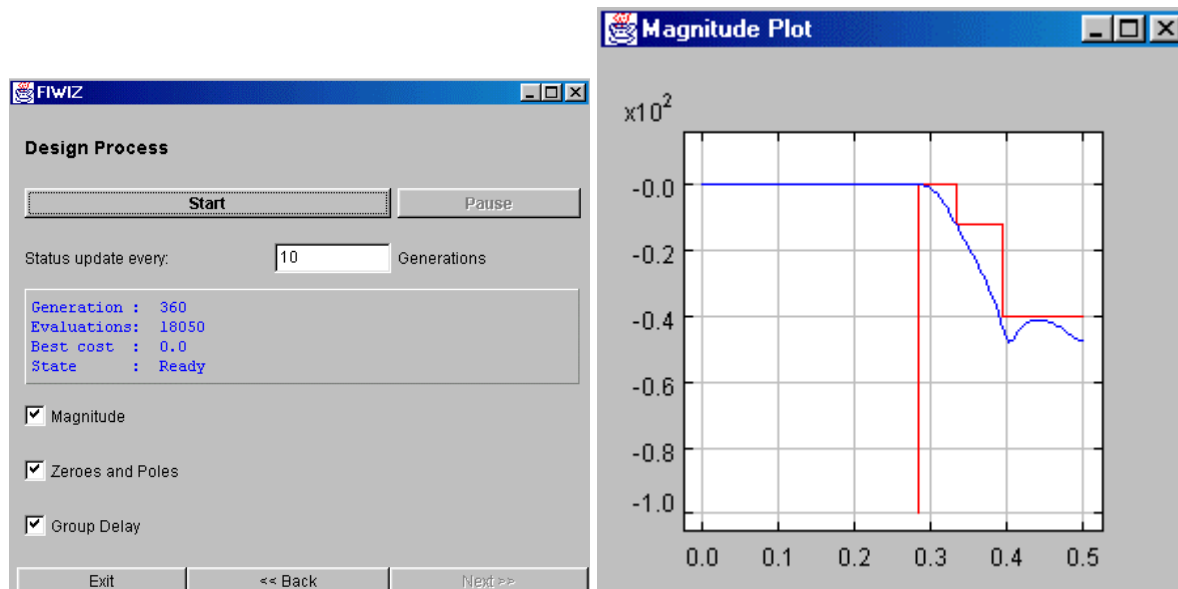
Design Control

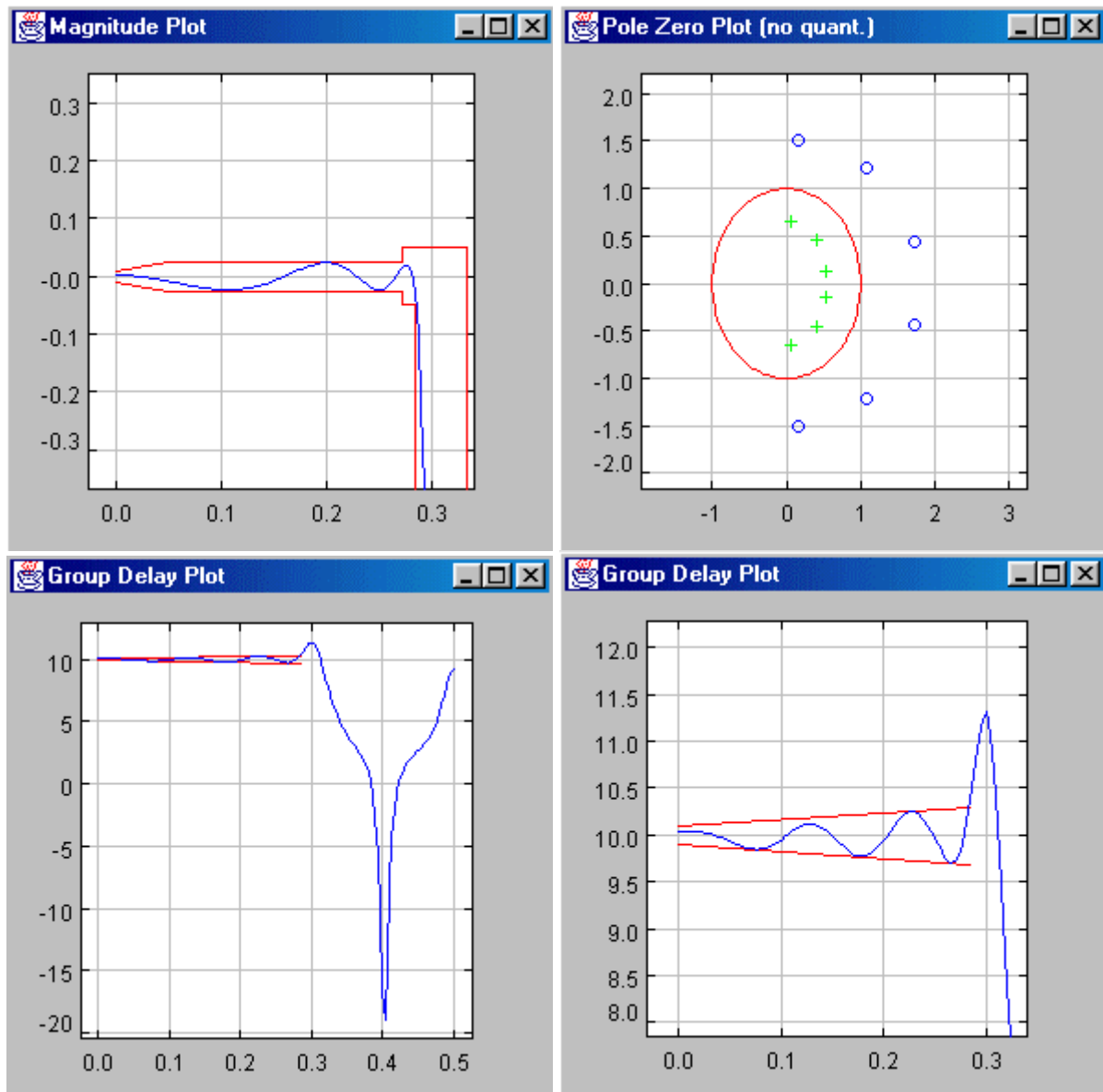
No. of population vectors: 50

dto. (refinement phase): 60

Sampling Granularity:
 ☐ coarse
☒ medium
☐ fine

Exit
<< Back
Next >>





Note: Screenshots have been done with FIWIZ version 1.6.

7.7 Lowpass with quasi-constant group delay (using zero radius constraint)

Filter Realization

Design method: 2-plane design

Single zeros: 12 $|Z| =$ 1.0 $|Z| =$ 99.0

Single poles: 6 $|P| =$ 1.0

Linear phase zero pairs: 0

Allpass pole/zero pairs: 0

Coefficient Quantization (bits): 24

Filter Structure: Direct Form

Exit << Back Next >>

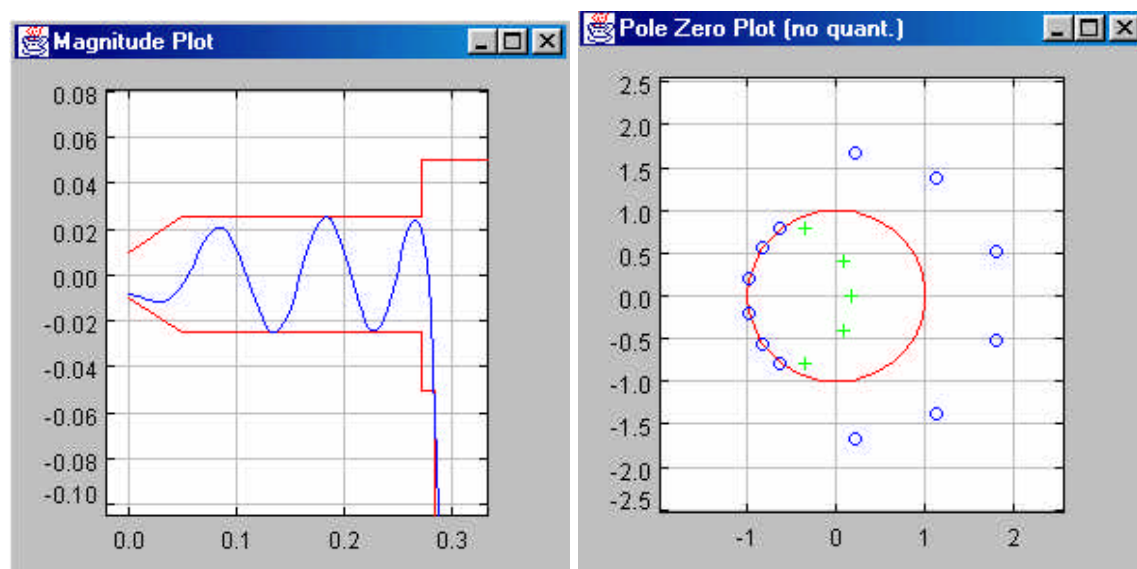
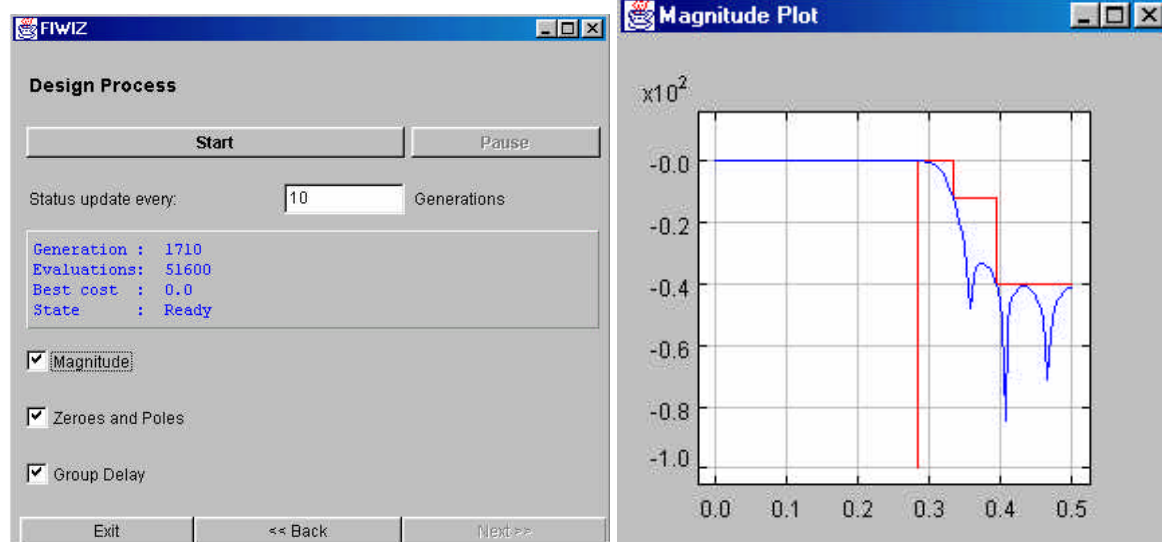
Design Control

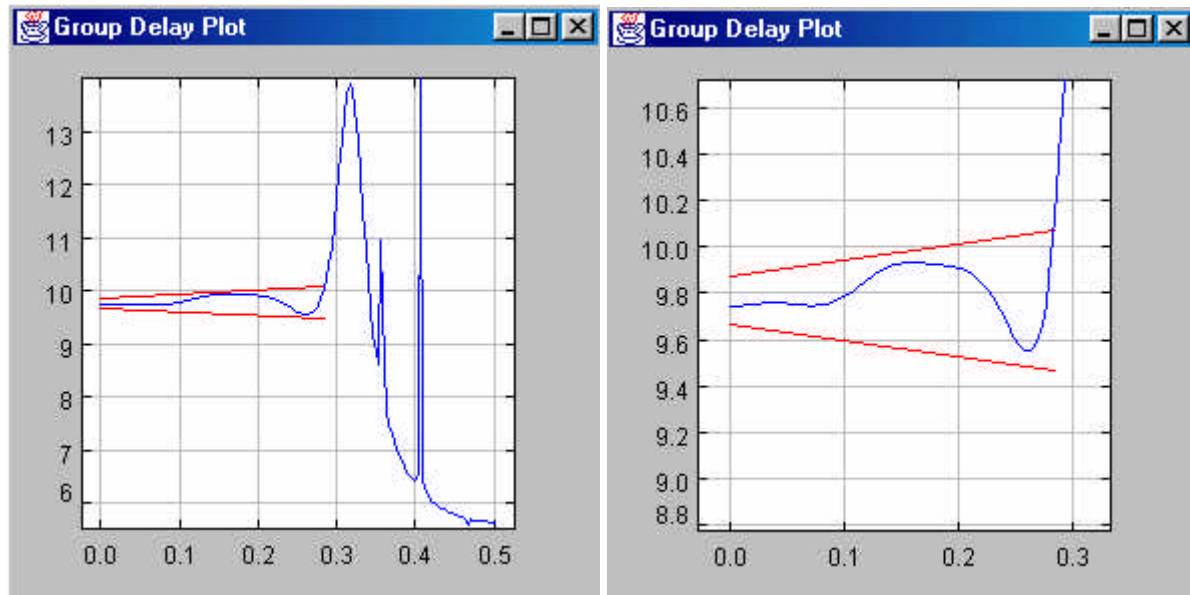
No. of population vectors: 30

db. (refinement phase): 60

Sampling Granularity ☐ coarse ☐ medium ☒ fine

Exit << Back Next >>





Note: Screenshots have been done with FIWIZ version 1.6.

7.8 Lowpass with constant group delay (FIR-approach)

Filter Realization

Design method:

Single zeros: $|Z| >$ $|Z| <$

Single poles: $|P| <$

Linear phase zero pairs:

Allpass pole/zero pairs:

Coefficient Quantization (bits):

Filter Structure:

Exit << Back Next >>

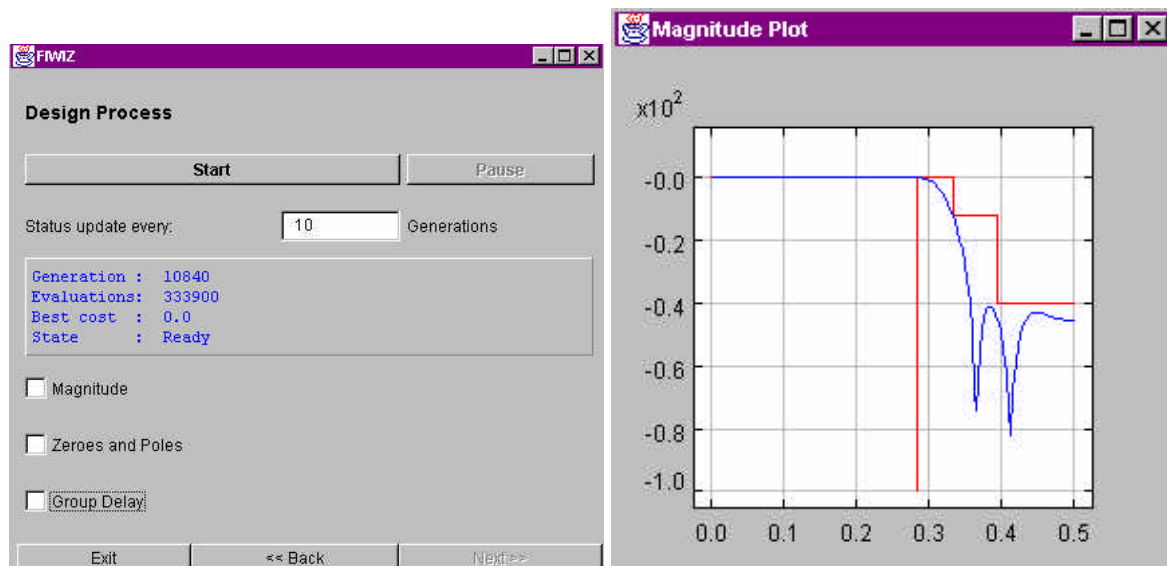
Design Control

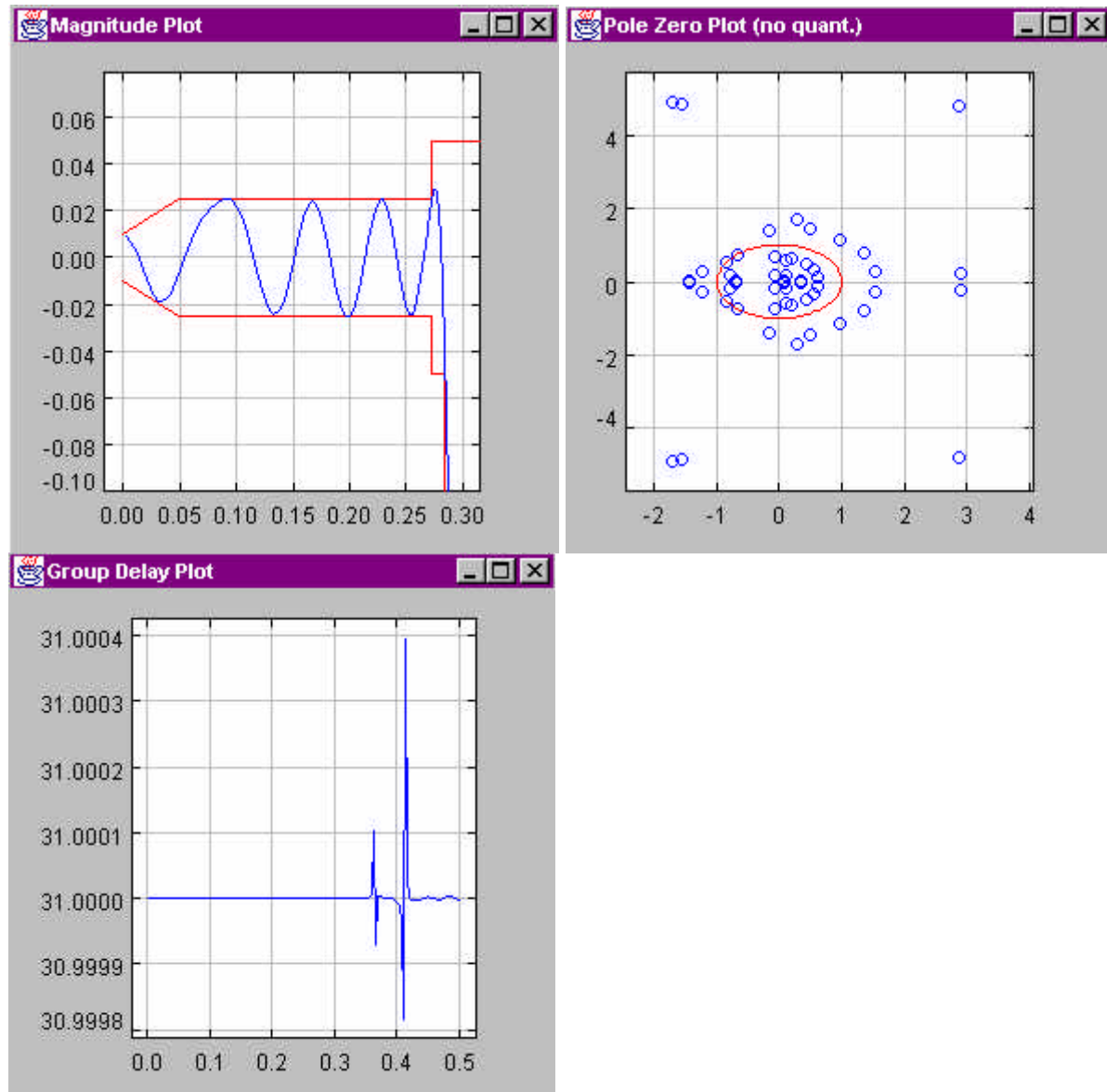
No. of population vectors:

dto (refinement phase):

Sampling Granularity: ☐ coarse ☐ medium ☒ fine

Exit << Back Next >>





Note: Screenshots have been done with FIWIZ version 1.6.

7.9 Equalization

FIWIZ

Prefilter Coefficients

Coefficients:

☒ Numerator Coefficients
☐ Denominator Coefficients

Coefficients: Add

1.0
1.1

Edit Delete

☒ Magnitude Plot
☒ Group Delay Plot

Exit << Back Next >>

FIWIZ

Prefilter Coefficients

Coefficients:

☐ Numerator Coefficients
☒ Denominator Coefficients

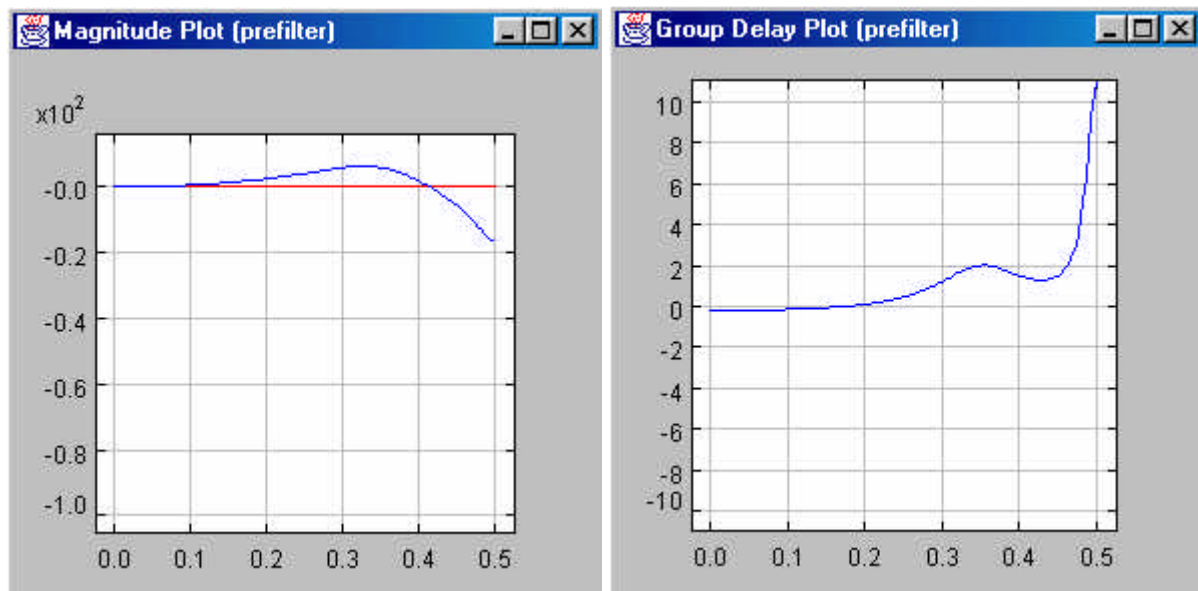
Coefficients: Add

1.0
0.75
0.4

Edit Delete

☒ Magnitude Plot
☒ Group Delay Plot

Exit << Back Next >>



FIWIZ

Filter Realization

Design method: 2-plane design

Single zeros: 10 $|Z| =$ 0.0 $|Z| <$ 99.0

Single poles: 2 $|P| =$ 1.0

Linear phase zero pairs: 0

Allpass pole/zero pairs: 2

Coefficient Quantization (bits): inf

Filter Structure: 1st and 2nd order sections

Exit << Back Next >>

FIWIZ

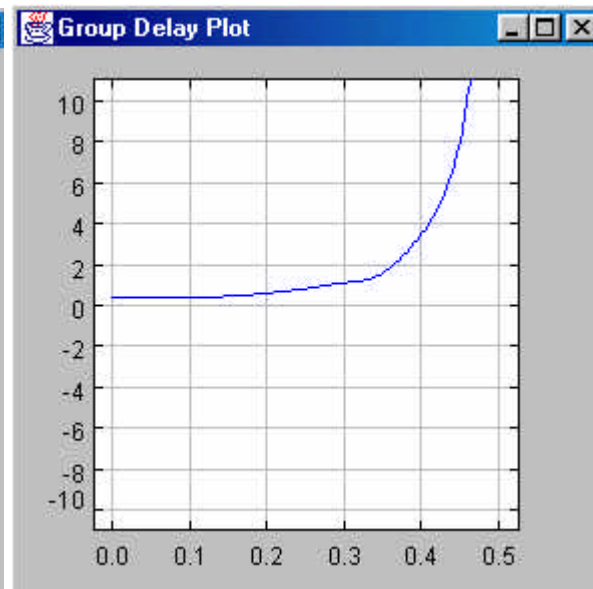
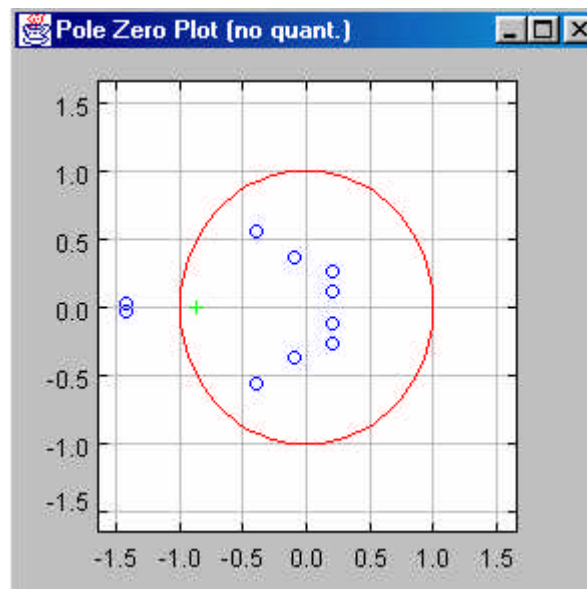
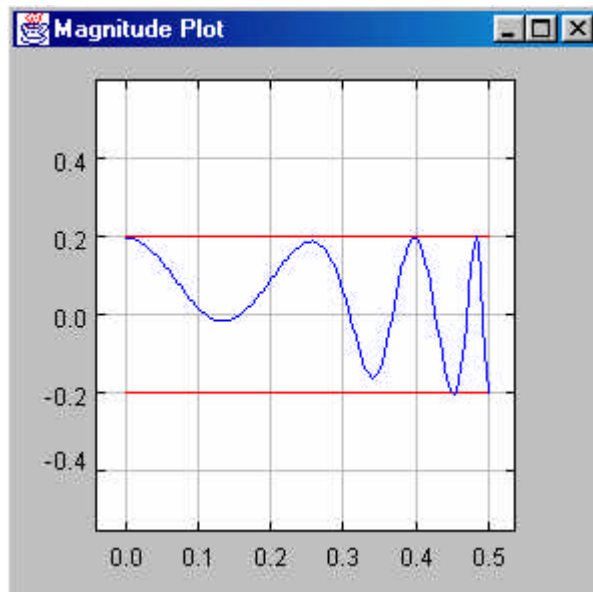
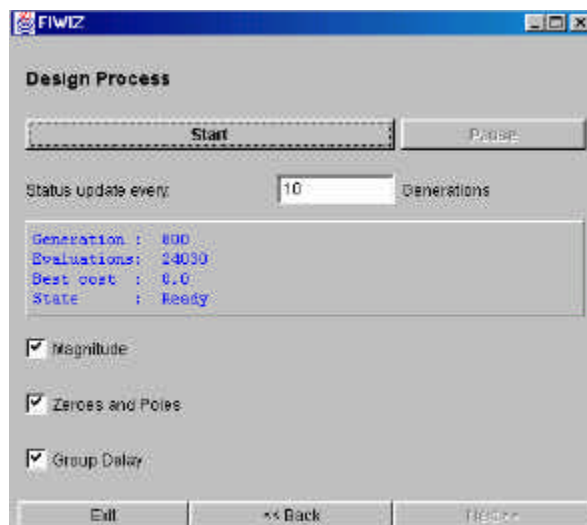
Design Control

No. of population vectors: 30

dfa. (refinement phase): 60

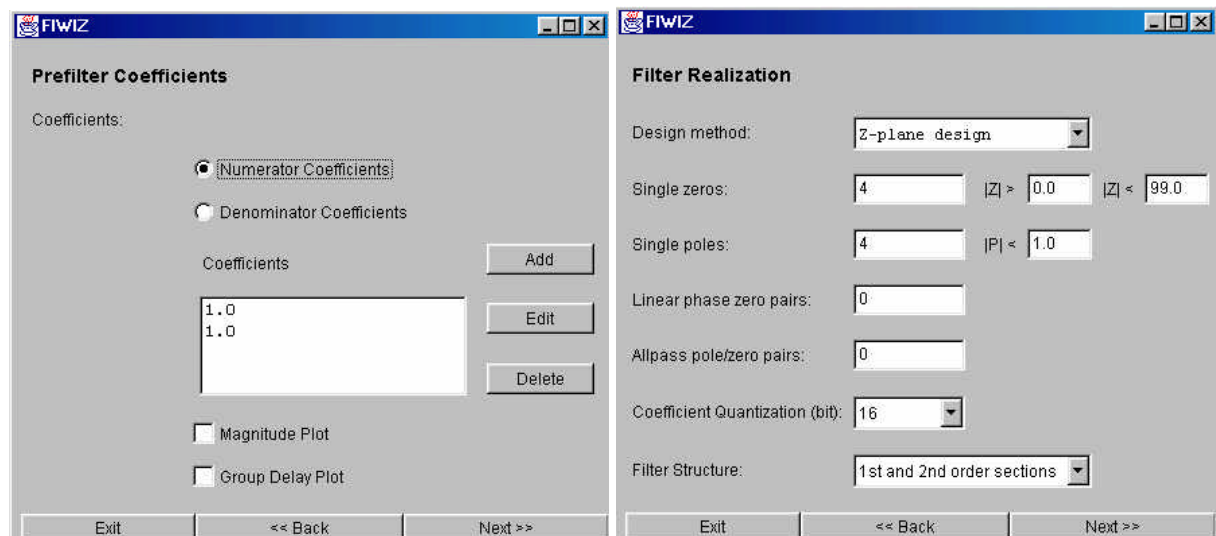
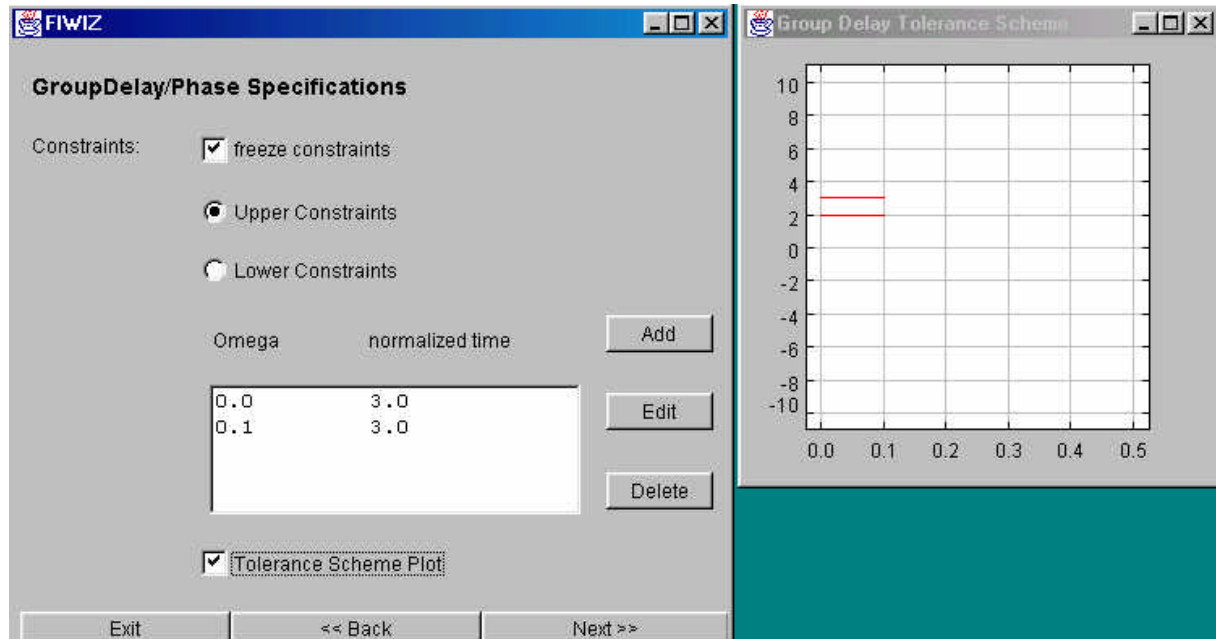
Sampling Granularity: ☐ coarse ☐ medium ☒ fine

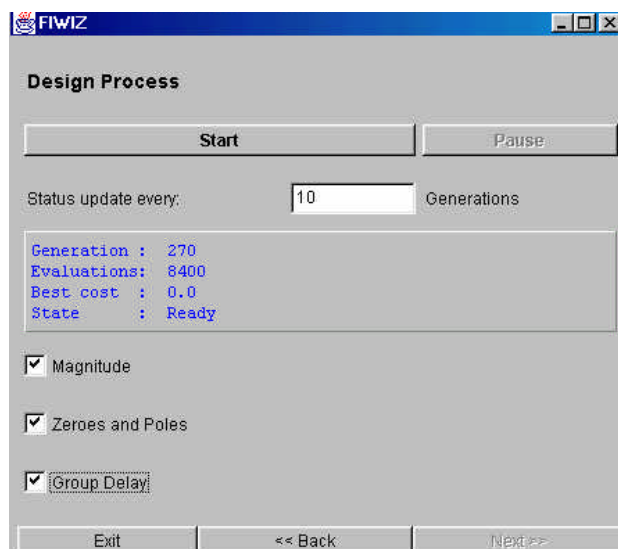
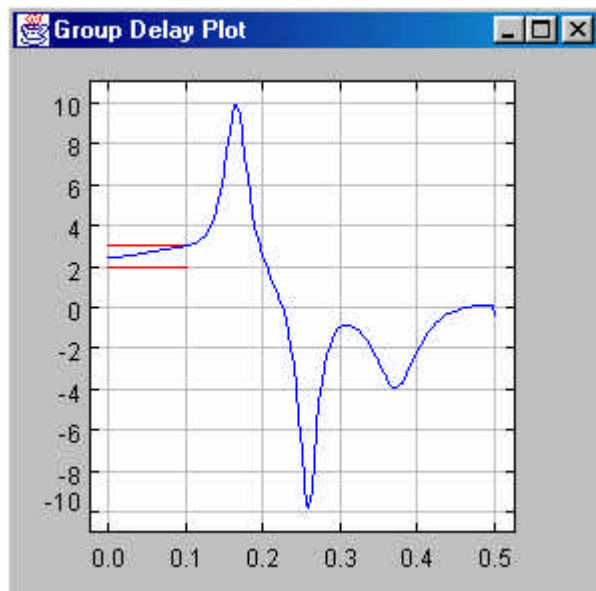
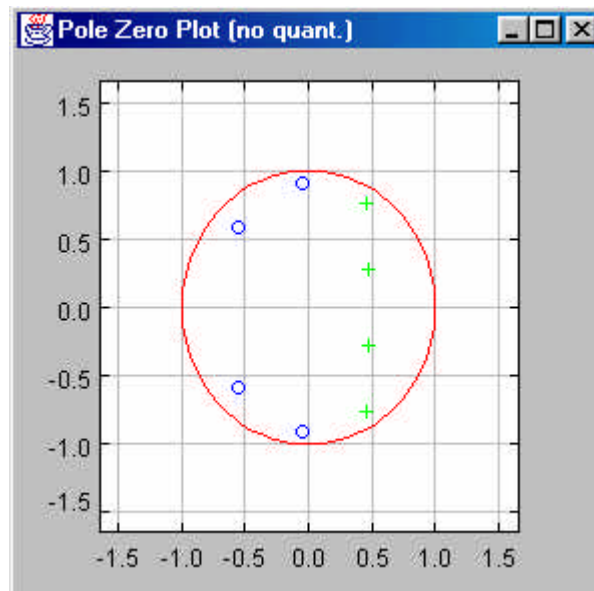
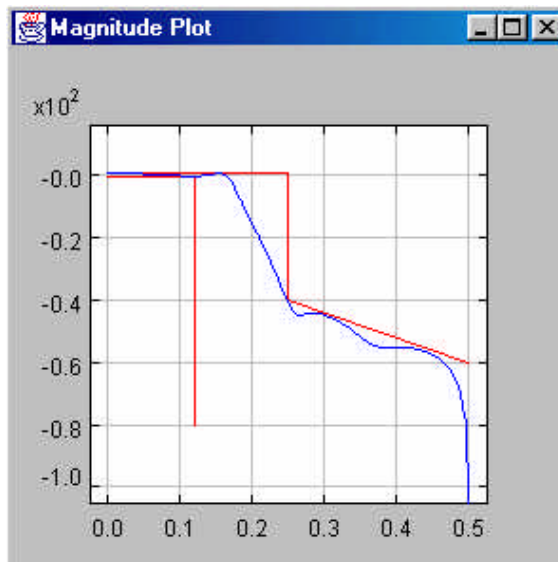
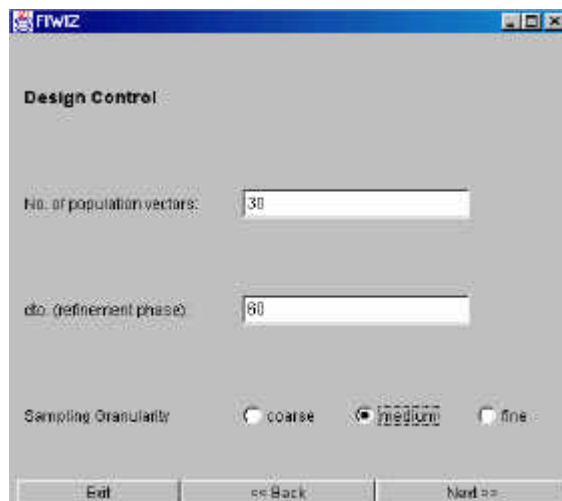
Exit << Back Next >>



Note: Screenshots have been done with FIWIZ version 1.6.

7.10 Minimum delay filter





Note: Screenshots have been done with FIWIZ version 1.6.

7.11 Examples for IIR-design according to analog prototypes

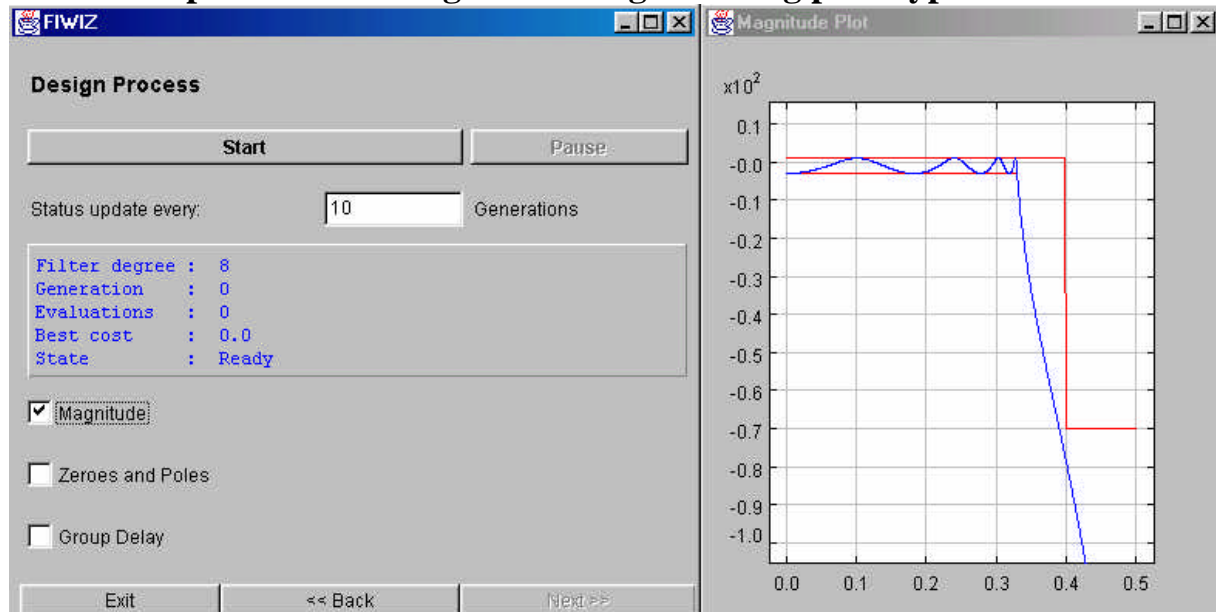


Figure 7.11-1: Chebyshev1 lowpass.

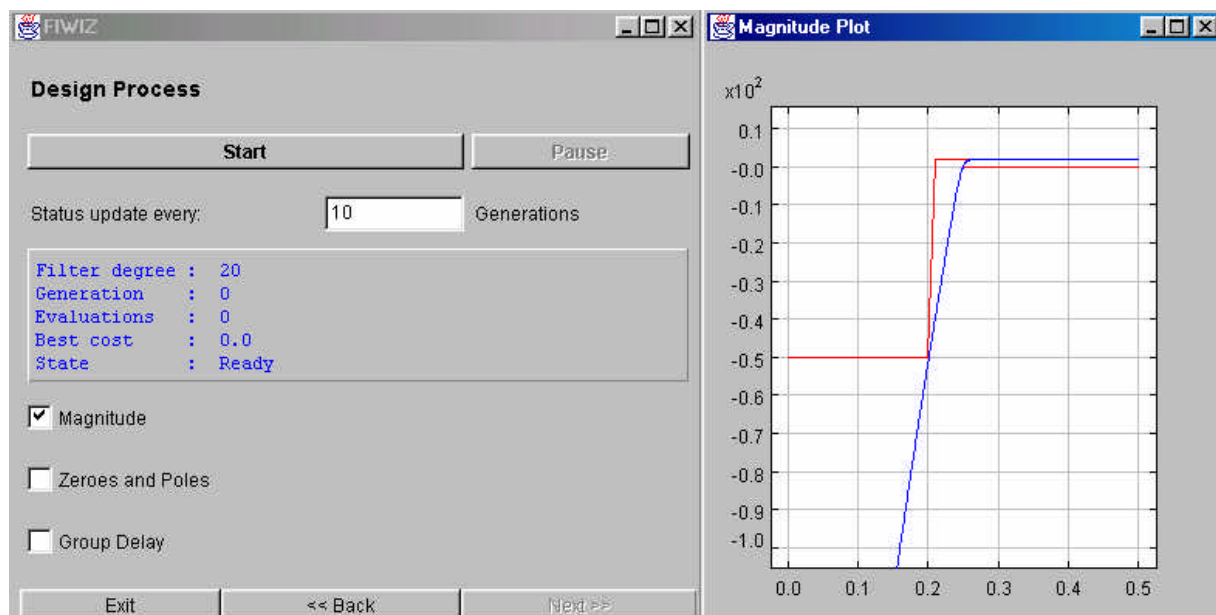


Figure 7.11-2: Butterworth highpass.

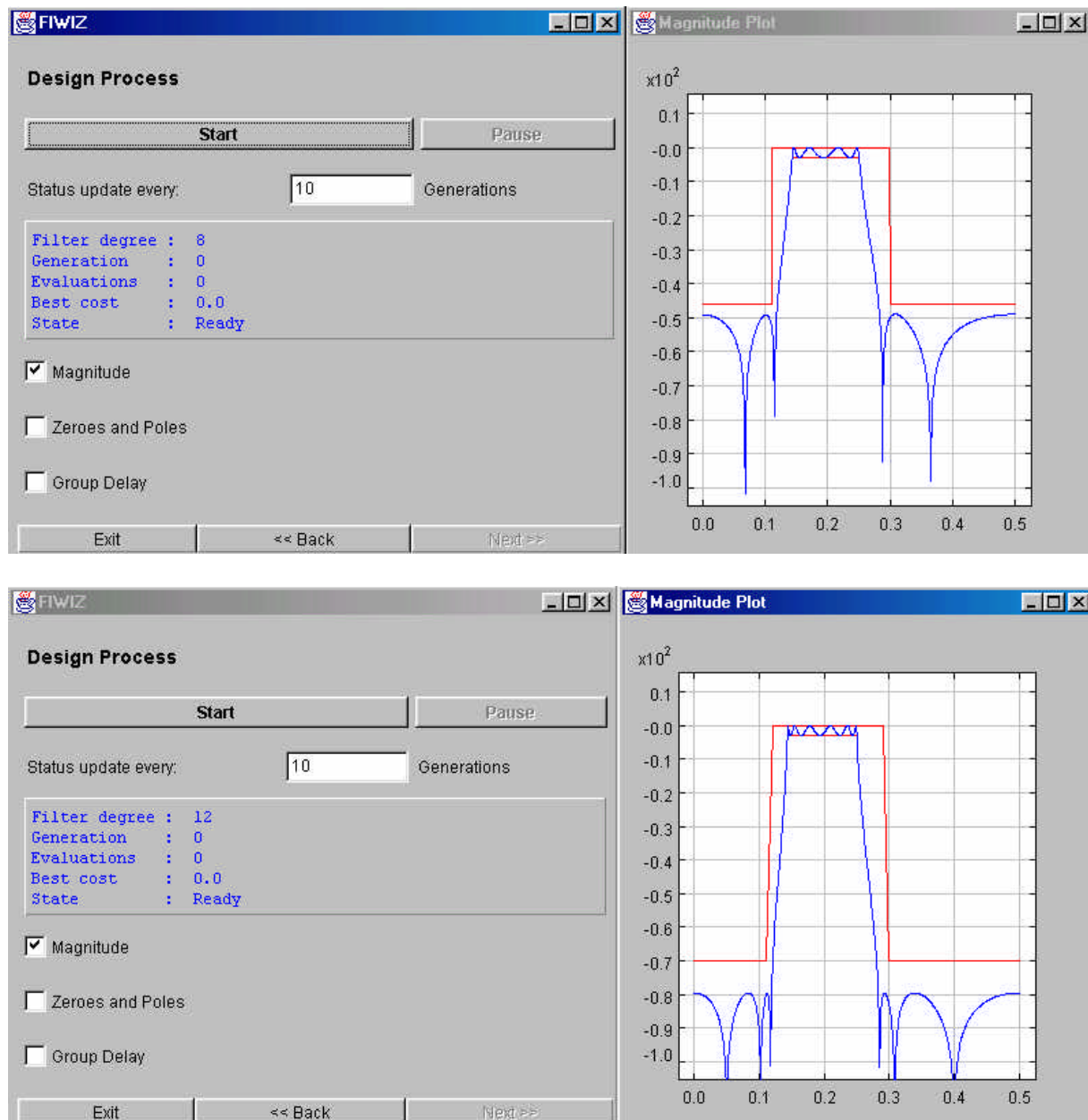


Figure 7.11-3: Elliptical bandpass. Compare the lower bandpass to chapter 7.4: the DE-based design has a lower degree because the elliptical design always has an even degree. This is due to the fact that the design is made by an equivalent lowpass and then doing a frequency transformation which doubles the degree.

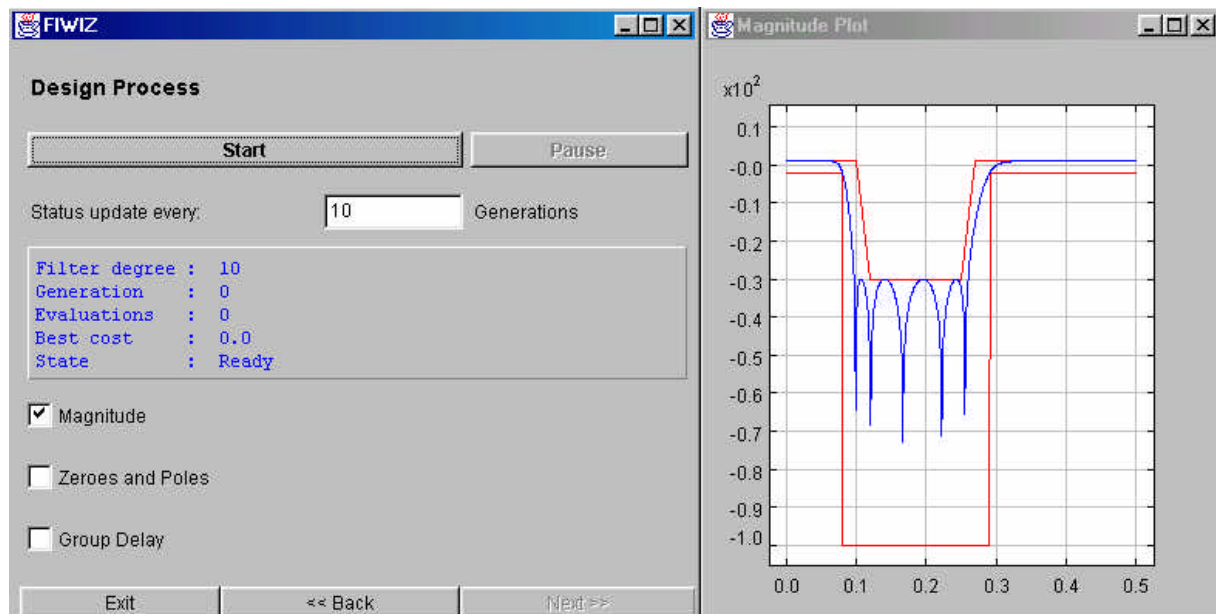


Figure 7.11-4: Chebyshev2 bandstop.

7.12 Examples for linear phase FIR-design

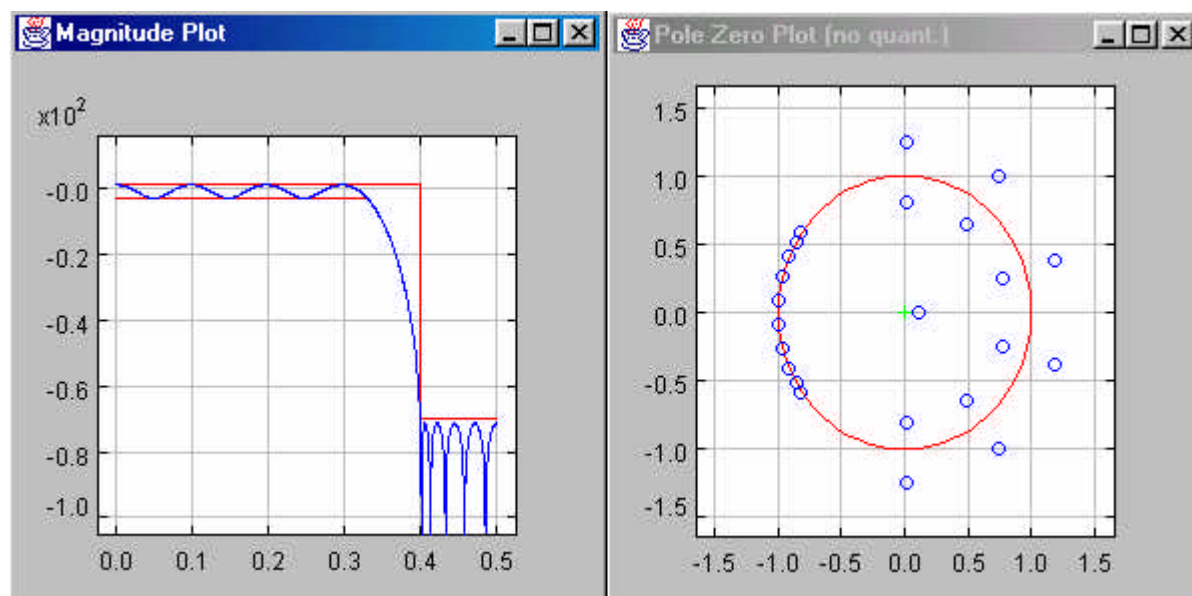


Figure 7.12-1: Equiripple lowpass.

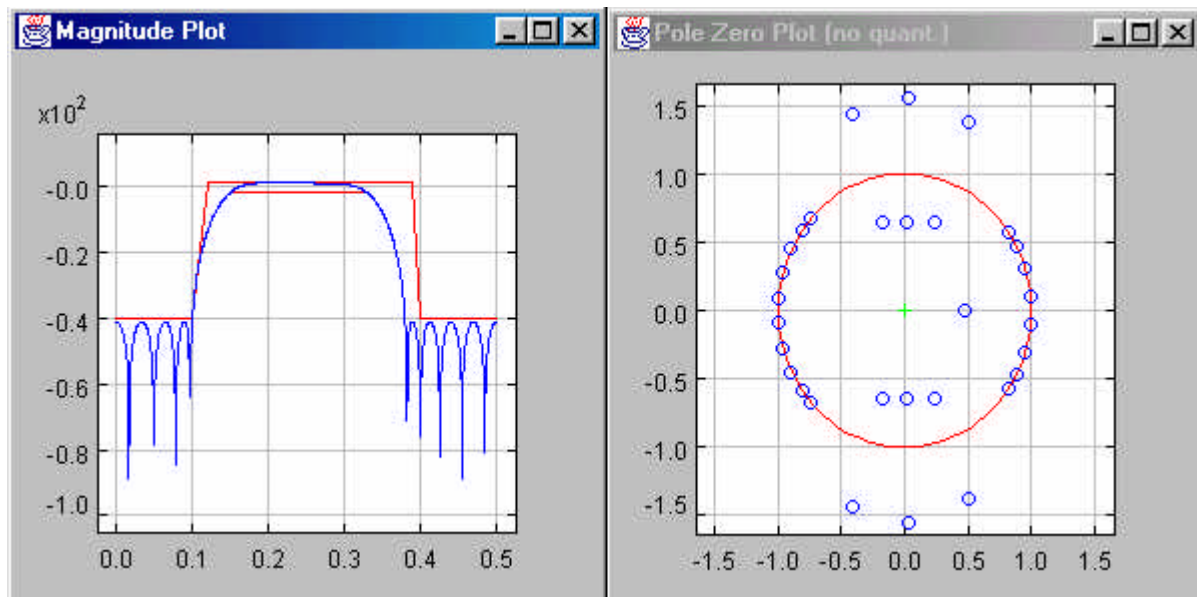


Figure 7.12-2: Bandpass with maximally flat passband response.

8. References

- [Mit93] Mitra, S.K., and Kaiser, J.F., Handbook for Digital Signal Processing, John Wiley, 1993.
- [PtPlot00] <http://ptolemy.eecs.berkeley.edu/ptolemyII/ptII0.3/ptII0.3/ptolemy/plot/doc/index.htm>.
- [Sto00] <http://www.icsi.berkeley.edu/~storn/code.htm>.
- [Rab75] Rabiner, L.R., and Gold, B., Theory and Application of Digital Signal Processing, Prentice-Hall, 1975.