Contents

- Start by constucting a design object
- Determine the available design FIR &/or IIR algorithms using the 'designmethods' function
- Vary the filter parameters
- Capture the design as a 'dfilt' object
- Enquire about the filter structure & it's properties
- Estimate computational complexity of the filter
- Use fvtool function to visualize the resulting designs and compare their properties
- Take the filter object and integrate it into SIMULINK by the System Design environment (optional)
- Generate HDL code of the Filter (optional)

```
% Applying specification-based filter design methodology
% -> using 'fdesign', 'designmethods', 'design', 'info' & 'cost' functions

% Remove all global variables from the current workspace
clear;
% Clear all input & output from the command window display
clc;
```

Start by constucting a design object

```
fd=fdesign.lowpass;
```

Determine the available design FIR &/or IIR algorithms using the 'designmethods' function

```
FIR_Filters = designmethods(fd,'fir')      % Ask for FIR
IIR_Filters = designmethods(fd,'iir')      % Ask for IIR
%All_Filters = designmethods(fd)           % Ask for FIR & IIR
```

```
FIR_Filters =

  4×1 cell array

    {'equiripple'}
    {'ifir'      }
    {'kaiserwin' }
    {'multistage'}


IIR_Filters =

  4×1 cell array

    {'butter'}
    {'cheby1'}
    {'cheby2'}
    {'ellip' }
```

Vary the filter parameters

```
for a cut-off frequency: 0.35, passband ripple: 1 & stopband attenuation: 50
```

```
fd.Fpass = 0.34;   %Passband frequency
fd.Fstop = 0.36;   %Stopband frequency
fd.Apass = 1;      %Passband ripple
fd.Astop = 50;     %Stopband attenuation

% Plot them separately
%design(fd,'equiripple');
%design(fd,'kaiserwin');
%design(fd,'ellip');
%design(fd,'butter');
```

Capture the design as a 'dfilt' object

Using the 'design' function, design eg equiripple, kaiser window FIR filters & elliptic, butterwoth IIR filter that meet the specifications

```
lpFIR1 = design(fd,'equiripple');
lpFIR2 = design(fd,'kaiserwin');
lpIIR1 = design(fd,'ellip');
lpIIR2 = design(fd,'butter');

% Further analysis can be performed such as :
% * Enquiring about the filter structure & it's properties
% * Computing the cost "estimating computational complexity" of implementing the filter
% * Viewing/Measuring the filter response characteristics
```

Enquire about the filter structure & it's properties

```
FIR1info = info(lpFIR1)
FIR2info = info(lpFIR2)
```

```
IIR1info = info(lpIIR1)
IIR2info = info(lpIIR2)
```

```
FIR1info =

  6×35 char array

    'Discrete-Time FIR Filter (real)    '
    '-------------------------------    '
    'Filter Structure  : Direct-Form FIR'
    'Filter Length     : 175            '
    'Stable            : Yes            '
    'Linear Phase      : Yes (Type 1)   '


FIR2info =

  6×35 char array

    'Discrete-Time FIR Filter (real)    '
    '-------------------------------    '
    'Filter Structure  : Direct-Form FIR'
    'Filter Length     : 294            '
    'Stable            : Yes            '
    'Linear Phase      : Yes (Type 2)   '


IIR1info =

  6×59 char array

    'Discrete-Time IIR Filter (real)                            '
    '-------------------------------                            '
    'Filter Structure    : Direct-Form II, Second-Order Sections'
    'Number of Sections  : 4                                    '
    'Stable              : Yes                                  '
    'Linear Phase        : No                                   '


IIR2info =

  6×59 char array

    'Discrete-Time IIR Filter (real)                            '
    '-------------------------------                            '
    'Filter Structure    : Direct-Form II, Second-Order Sections'
    'Number of Sections  : 46                                   '
    'Stable              : Yes                                  '
    'Linear Phase        : No                                   '
```

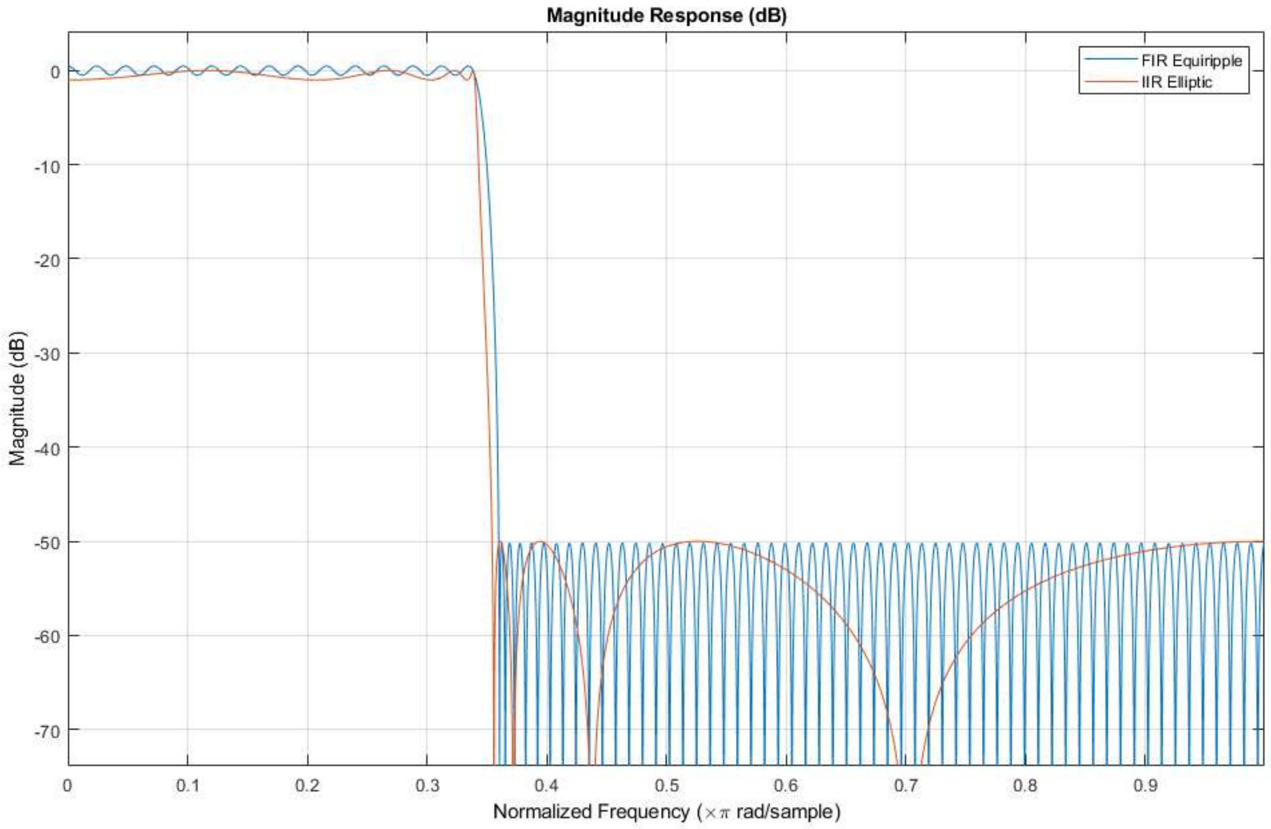
Estimate computational complexity of the filter

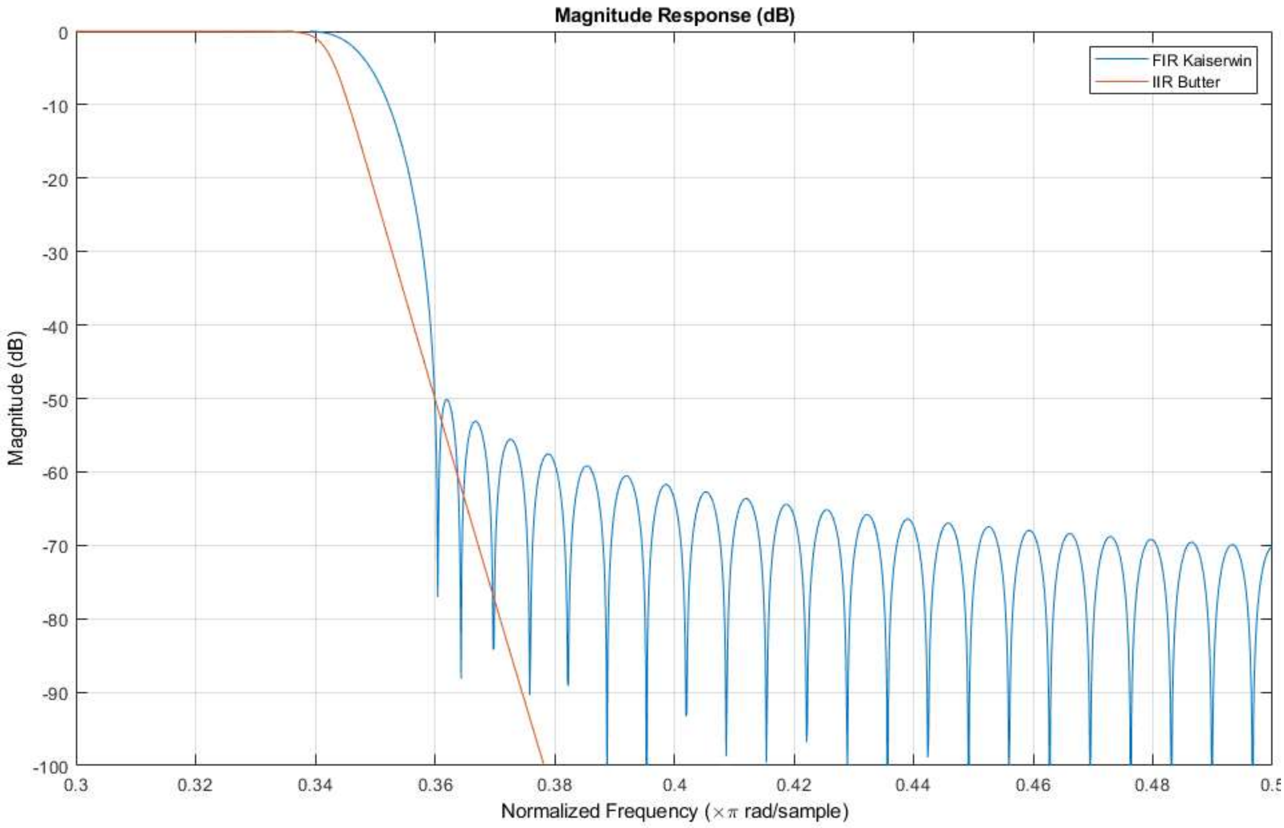
```
FIR1cost = cost(lpFIR1)
FIR2cost = cost(lpFIR2)
IIR1cost = cost(lpIIR1)
IIR2cost = cost(lpIIR2)
```

```
FIR1cost =

Number of Multipliers            : 175
Number of Adders                 : 174
Number of States                 : 174
Multiplications per Input Sample : 175
Additions per Input Sample       : 174

FIR2cost =

Number of Multipliers            : 294
Number of Adders                 : 293
Number of States                 : 293
Multiplications per Input Sample : 294
Additions per Input Sample       : 293

IIR1cost =

Number of Multipliers            : 16
Number of Adders                 : 16
Number of States                 : 8
Multiplications per Input Sample : 16
Additions per Input Sample       : 16

IIR2cost =

Number of Multipliers            : 184
Number of Adders                 : 184
Number of States                 : 92
Multiplications per Input Sample : 184
Additions per Input Sample       : 184
```

Use fvtool function to visualize the resulting designs and compare their properties

```
%fvtool(lpFIR1,lpFIR2,lpIIR1,lpIIR2);
%legend('FIR Equiripple','FIR Kaiserwin','IIR Elliptic','IIR Butter')

fvtool(lpFIR1,lpIIR1);
legend('FIR Equiripple','IIR Elliptic')

fvtool(lpFIR2,lpIIR2);
legend('FIR Kaiserwin','IIR Butter')
% Zoom-in to view the response better
xlim([0.3,0.5]);
ylim([-100,0]);
```





Take the filter object and integrate it into SIMULINK by the System Design environment (optional)

`[ Note: This will take some time because it needs to create a Simulink block and also open up SIMULINK ]`

```
realizemdl(lpFIR1);
%realizemdl(lpFIR2);
%realizemdl(lpIIR1);
%realizemdl(lpIIR2);
```

Generate HDL code of the Filter (optional)

```
[ Note: Need 'Filter Design HD coder' to execute the following command ]
```

```
%generatehdl(lpFIR1)
%generatehdl(lpFIR2)
%generatehdl(lpIIR1)
%generatehdl(lpIIR2)
```

Published with MATLAB® R2020b