Filter Design Using Scilab

Manas Das
Indian Institute of Technology, Bombay

March 1, 2012

Introduction

- What is a filter?
- A filter is a device or process that removes some unwanted component or feature from a signal.

Objective

In this presentation i will show how differnt types of filters can be designed using scilab.

This presentation is being divided into following parts:

- Different windowing techniques.
- Filter design by different in-built functions available in scilab.

In this slide i will be describing different windowing techniques.This can be performed by different window functions with window length by using the in-built command window().

Window Functions for FIR Filter Design

- `Hamming Window.`

Window Functions for FIR Filter Design

- Hamming Window.
  `win=window('hm',n)`
- Kaiser Window.

Window Functions for FIR Filter Design

- Hamming Window.
  `win=window('hm',n)`
- Kaiser Window.
  `win=window('kr',n,alpha)`
- Chebyshev Window.

Window Functions for FIR Filter Design

- `Hamming Window.`
  `win=window('hm',n)`
- `Kaiser Window.`
  `win=window('kr',n,alpha)`
- `Chebyshev Window.`
  `win=window('ch',n,par)`

Window based Linear Phase FIR filter

Calling Sequence

`[wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)`

Arguments

- `ftype:'lp','hp','bp','sb'`

Window based Linear Phase FIR filter

Calling Sequence

`[wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)`

Arguments

- `ftype:'lp','hp','bp','sb'`
- `wtype:'re','tr','hm','hn','kr','ch'`

Window based Linear Phase FIR filter

Calling Sequence

```
[wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)
```

Arguments

- `ftype:'lp','hp','bp','sb'`
- `wtype:'re','tr','hm','hn','kr','ch'`
- `cfreq:2-vector of cutoff frequencies`

Window based Linear Phase FIR filter

Calling Sequence

`[wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)`

Arguments

- `ftype:'lp','hp','bp','sb'`
- `wtype:'re','tr','hm','hn','kr','ch'`
- `cfreq:2-vector of cutoff frequencies`
- `fpar:2-vector of window parameters`

Window based Linear Phase FIR filter

Calling Sequence

`[wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)`

Arguments

- `ftype:'lp','hp','bp','sb'`
- `wtype:'re','tr','hm','hn','kr','ch'`
- `cfreq:2-vector of cutoff frequencies`
- `fpar:2-vector of window parameters`
- `wft:time domain filter coefficients`

Window based Linear Phase FIR filter

Calling Sequence

`[wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)`

Arguments

- `ftype:'lp','hp','bp','sb'`
- `wtype:'re','tr','hm','hn','kr','ch'`
- `cfreq:2-vector of cutoff frequencies`
- `fpar:2-vector of window parameters`
- `wft:time domain filter coefficients`
- `wfm:frequency domain filter response on the grid fr`

Window based Linear Phase FIR filter

Calling Sequence

`[wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)`

Arguments

- `ftype:'lp','hp','bp','sb'`
- `wtype:'re','tr','hm','hn','kr','ch'`
- `cfreq:2-vector of cutoff frequencies`
- `fpar:2-vector of window parameters`
- `wft:time domain filter coefficients`
- `wfm:frequency domain filter response on the grid fr`
- `fr:frequency grid`

Equiripple FIR Filter Design

Calling Sequence

`[hn]=eqfir(nf,bedge,des,wate)`

Arguments

- `nf`:number of output filter points desired

Equiripple FIR Filter Design

Calling Sequence

```
[hn]=eqfir(nf,bedge,des,wate)
```

Arguments

- `nf:number of output filter points desired`
- `bedge:Mx2 matrix giving a pair of edges for each band`

Equiripple FIR Filter Design

Calling Sequence

`[hn]=eqfir(nf,bedge,des,wate)`

Arguments

- `nf:number of output filter points desired`
- `bedge:Mx2 matrix giving a pair of edges for each band`
- `des:M-vector giving desired magnitude for each band`

Equiripple FIR Filter Design

Calling Sequence

`[hn]=eqfir(nf,bedge,des,wate)`

Arguments

- `nf`:number of output filter points desired
- `bedge`:Mx2 matrix giving a pair of edges for each band
- `des`:M-vector giving desired magnitude for each band
- `wate`:M-vector giving relative weight of error in each band

Equiripple FIR Filter Design

Calling Sequence

`[hn]=eqfir(nf,bedge,des,wate)`

Arguments

- `nf:number of output filter points desired`
- `bedge:Mx2 matrix giving a pair of edges for each band`
- `des:M-vector giving desired magnitude for each band`
- `wate:M-vector giving relative weight of error in each band`
- `hn:output of linear-phase FIR filter coefficients`

IIR Digital filter

Calling Sequence

```
[hz]=iir(n,ftype,fdesign,frq,delta)
```

Arguments

- `n:the filter order`

IIR Digital filter

Calling Sequence

```
[hz]=iir(n,ftype,fdesign,frq,delta)
```

Arguments

- `n:the filter order`
- `ftype:'lp','hp','bp','sb'`

IIR Digital filter

Calling Sequence

```
[hz]=iir(n,ftype,fdesign,frq,delta)
```

Arguments

- n:the filter order
- ftype:'lp','hp','bp','sb'
- fdesign:'butt','cheb1','cheb2' and 'ellip'

IIR Digital filter

Calling Sequence

```
[hz]=iir(n,ftype,fdesign,frq,delta)
```

Arguments

- `n:the filter order`
- `ftype:'lp','hp','bp','sb'`
- `fdesign:'butt','cheb1','cheb2' and 'ellip'`
- `frq:2-vector of discrete cut-off frequencies`

IIR Digital filter

Calling Sequence

```
[hz]=iir(n,ftype,fdesign,frq,delta)
```

Arguments

- `n:the filter order`
- `ftype:'lp','hp','bp','sb'`
- `fdesign:'butt','cheb1','cheb2' and 'ellip'`
- `frq:2-vector of discrete cut-off frequencies`
- `delta:2-vector of error values`

IIR Digital filter

Calling Sequence

```
[hz]=iir(n,ftype,fdesign,frq,delta)
```

Arguments

- `n:the filter order`
- `ftype:'lp','hp','bp','sb'`
- `fdesign:'butt','cheb1','cheb2' and 'ellip'`
- `frq:2-vector of discrete cut-off frequencies`
- `delta:2-vector of error values`

Filter

To design filter of any magnitude

Function- **remezb**

Calling Sequence

an=remezb(nc,fg,ds,wt)

- **nc:number of cosine functions**
- **fg:dense grid of frequency**
- **ds:derived magnitude values on this grid**
- **wt:error weighting vectors**
- **an:filter coeffficients**

Filter

Filtering of discrete signals by **flts** function

Function- **flts**

Calling Sequence

y,[x]=**flts(u,sl[,x0])**

- **u:the data to be filtered**
- **x0:initial state vector/matrix giving necessaty i/p-o/p.It allows for filtering of length signals**
- **x:optimal variable which gives the state sequence.**

Summary

In this presentation we have learnt:

- Different windowing techniques

Summary

In this presentation we have learnt:

- Different windowing techniques
- How to design linear phase FIR filter using `wfir()`

Summary

In this presentation we have learnt:

- Different windowing techniques
- How to design linear phase FIR filter using `wfir()`
- How to design linear phase FIR filter using `eqfir()`

Summary

In this presentation we have learnt:

- Different windowing techniques
- How to design linear phase FIR filter using `wfir()`
- How to design linear phase FIR filter using `eqfir()`
- How to design IIR filter using `iir()`

Summary

In this presentation we have learnt:

- Different windowing techniques
- How to design linear phase FIR filter using `wfir()`
- How to design linear phase FIR filter using `eqfir()`
- How to design IIR filter using `iir()`
- How to design filter of any magnitude using `remezb()`

Summary

In this presentation we have learnt:

- Different windowing techniques
- How to design linear phase FIR filter using wfir()
- How to design linear phase FIR filter using eqfir()
- How to design IIR filter using iir()
- How to design filter of any magnitude using remezb()
- How to filter discrete signals using flts()

Textbook Companion

- You already know Textbook Companion Project
- There are books on Signal Processing using Scilab under this project

Textbook Companion

- You already know Textbook Companion Project
- There are books on Signal Processing using Scilab under this project
- Refer to the link: http://www.scilab.in/Completed_Books