27/May/2020

* FFT fourier transform matlab

```
Fa = 1000;          % sampling frequency
Ta = 1/Fa;          % sampling period of time
dt = 0: Ta: 5-Ta;   % signal duration
 f1 = 10;
  f2 = 30;
  f3 = 70;
 y1 = 10 * sin (2* pi * f1 * dt);
 y2 = 10 * sin(2* pi * f2 * dt);
 y3 = 10 * sin (2* pi * f3 * dt);
 y4 = y1 + y2 + y3;
nfft = length (y4)
nfft2 = 2^ nextpower (nfft);
  ft = fft (y4, nfft2);
  fft = ft (1: nfft2 /2);
  plot (abs (fft));
```

* FIR & IIR filter

AN IIR filter: consider system described by transfor function

$$H(z) = \frac{b_3 z^3 + b_2 z^2 + b_1 z + b_0}{z^3 + a_2 z^2 + a_1 z + a_0}$$

corresponding difference Eq^n

$$y[k] = -a_2 y[k-1] - a_1 y[k-2] - a_0 y[k-3] + b_3 f[k] + b_2 f[k-1] + b_1 f[k-2] + b_0 f[k-3]$$

show current o/p is a function of current / past i/p & past o/p; it has a recursive nature

* IIR filter can have poles at Arbitary locations

* FIR filter: consider system by transfor function

$$H(z) = \frac{b_3 z^3 + b_2 z^2 + b_1 z + b_0}{z^3}$$

Vanishing moments

$$m_K = \int_{-\infty}^{\infty} f(x)\, x^k\, dx$$

moment k vanishes if the integral is zero

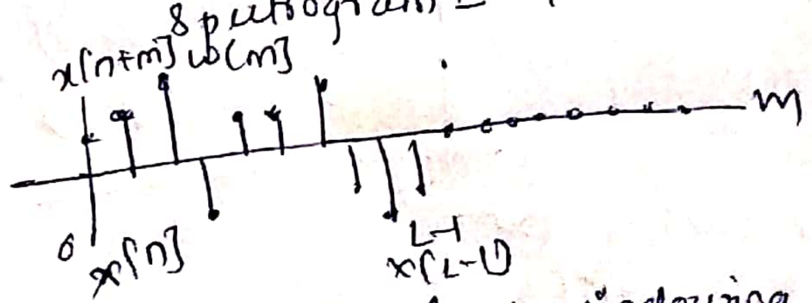
- higher number of vanishing moments = more complex wavelet, more accurate representation of complex signal
- higher number of vanishing moments = longer support
- Implementation of signal filtering singal using WT inmatlab

```
Clear all
[K, Fs]= audioread ('man_voice.wav');
K = k*0.5/rms(K);
K = acogn (K, 12, 'measured'),
[c, l] = wavedec [K, 3, 'db4');
   b = wthredh (c, 's', 0.25);
   y = wave rec (b, l, 'db4');
   y = y*0.5 / rms(y);
   sound (g, Fs);
```

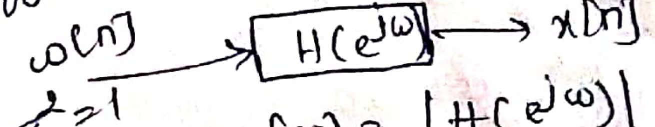
- short-time fourier transform & spectrogram

$$x[n,k] = \sum_{m=0}^{L-1} x[n+m]\, w[m]\, e^{jk\frac{2\pi}{N}m}$$

$$\text{Spectrogram} = |x[n,k]|^2$$



- Welch's method & windowing



$$S_{xx}(\omega) = |H(e^{j\omega})|^2$$

$$S_{xx}(f) = |H(e^{j2\pi f(Fs)})|^2$$

The corresponding difference Equation

$$y[k] = b_3 f[k] + b_2 f[k-1] + b_1 f[k-2] + b_0 f[k-3]$$

shows the current o/p is a function of current / past i/p

FIR filters only leave poles at the origin

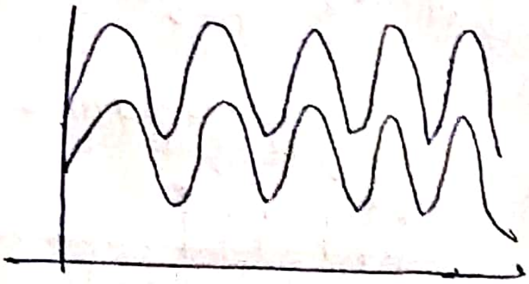
* study & andysis FIR & IIR using FDA tool in matlab

By writing the code in command window

```
>>fdatool
>> In fdatool ( line24)
>> filter designer
>> F = 500;
>> T = 1/F;
>> t = 0:T:1-T;
>> x = sin (2*pi*10*t) + 0.2 * randin(size(t));
>> y = filter (Num,1,x);
>> plot (t,x,t,y);
>> d = lpt

  d =  filterstructure: 'Direct - form FIR'
       Arithmatic    : 'double'
       Numerator     : [1x11double]
       persistent memory : false

>> yd = filter(d,x);
>> plot (t, y, t,yd);
>> plot (t,y, t, yd+1);
>> shg
```



* Introduction WT & CWT & DWT

$$FT \rightarrow X(F) = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi Ft}\, dt$$

wavelet transform:

$$X(a,b) = \int_{-\infty}^{\infty} x(t)\, \psi^{*}_{a,b}(t)\, dt$$

scale ↗   ^ time

- ECG signal analysis using matlab

```
beat_count = 0;
for k=2 : length(sig) - 1 :
    if (sig(k)> sig (k-1) & sig(k) > sig (k+1) & sig(k) > C)
    beat_count = beat_count +1;
    end
end
fs = 100;
N = length(sig);
duration_in_seconds = N/fs;
duration_in_minutes = duration_in_seconds / 60;
BPM = beat_count / duration_in_minutes.
```

Python

- Graphical use interfaces with Tkinter

* setting up a GUI with widgets

```
from tkinter import
window = Tk()
 b1 = Button (window.text = "Execute")
 b1 . pack()
[window . mainloop()
└> e1 = entry (window)
 el.grid (row=0, column=1)
t1 = Text (window, height =1, width = 20)
 t1.grid (row=0, column =0)
```

① Interfacing with database

→ connecting & inserting Data to sqlite via python

```
import sqlite3          . def (create - table()):
 Conn-qlite3. connect ("lite.db")
  cur= conn. cursor()
  Cur. Execute ("CREATE TABLE store ("item TEXT, quality
                                  INTEGER, price REAL)")
  conn. commit()
  Conn. close()
 def insert item, quantity, price):
    conn=Sqlite3. connect ("lite.db")
    cur = conn. cursor()
    cur. Execute ("INSERT INTO store VALUES ('(?,?)
                                   item, quantity, price)
   Conn. commit()
   conn. close()
insert ("coffee cup", 10,5)
 def view ():
    conn =sqlite3, connect ("lite.db")
    cur= conn. cursor()
    cur= Execute ("SELECT * FROM store")
    rows =(cur, fetchall()
     conn. close()
    return rows
 print (view())
```