DSP EXPT 1 - FILTERS

The following FIR filters were discussed in the lab:

- Moving Average Filter
- First-order difference Filter
- Three-point central difference Filter

MOVING AVERAGE FILTER

- The moving average is the most common filter in DSP, mainly because it is the easiest digital filter to understand and use.

- In spite of its simplicity, the moving average filter is optimal for a common task: **reducing random noise** while **retaining a sharp step response**.

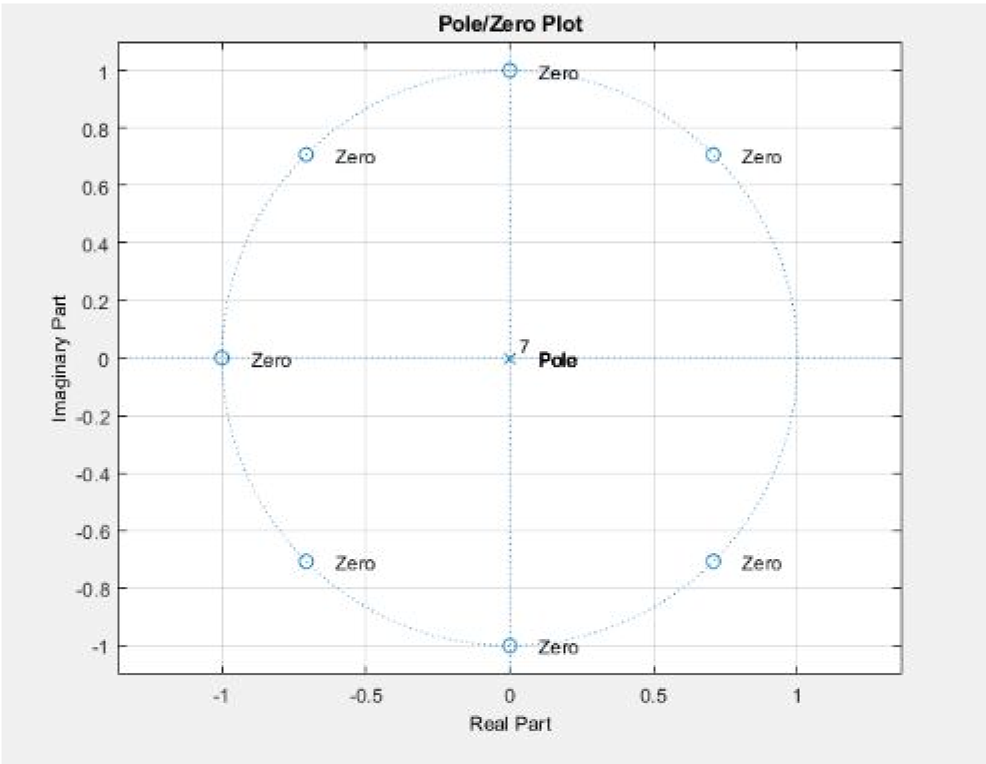
- The moving average filter is a simple **Low Pass FIR** (Finite Impulse Response) filter commonly used for smoothening of a signal.

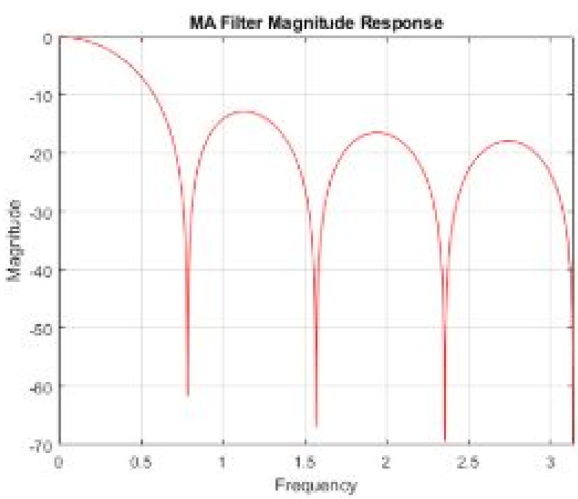
- However, the moving average is the worst filter for frequency domain encoded signals, with little ability to separate one band of frequencies from another.

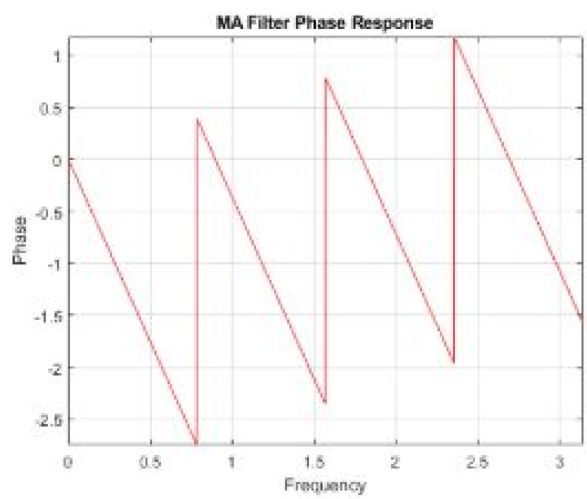
$$y[n] = \frac{1}{L}\sum_{k=0}^{L-1} x[n-k] \qquad H(z) = \frac{Y(z)}{X(z)} = \frac{1}{L}\sum_{k=0}^{L-1} z^{-k}$$

$$H(z) = \frac{1}{8}\frac{z^8 - 1}{z^7(z-1)} \qquad H[f] = \frac{\sin(\pi f M)}{M\sin(\pi f)}$$

If we take L = 8, the corresponding pole-zero plot and the Magnitude and phase response are as shown below:







A) MOVING AVERAGE FILTER Results

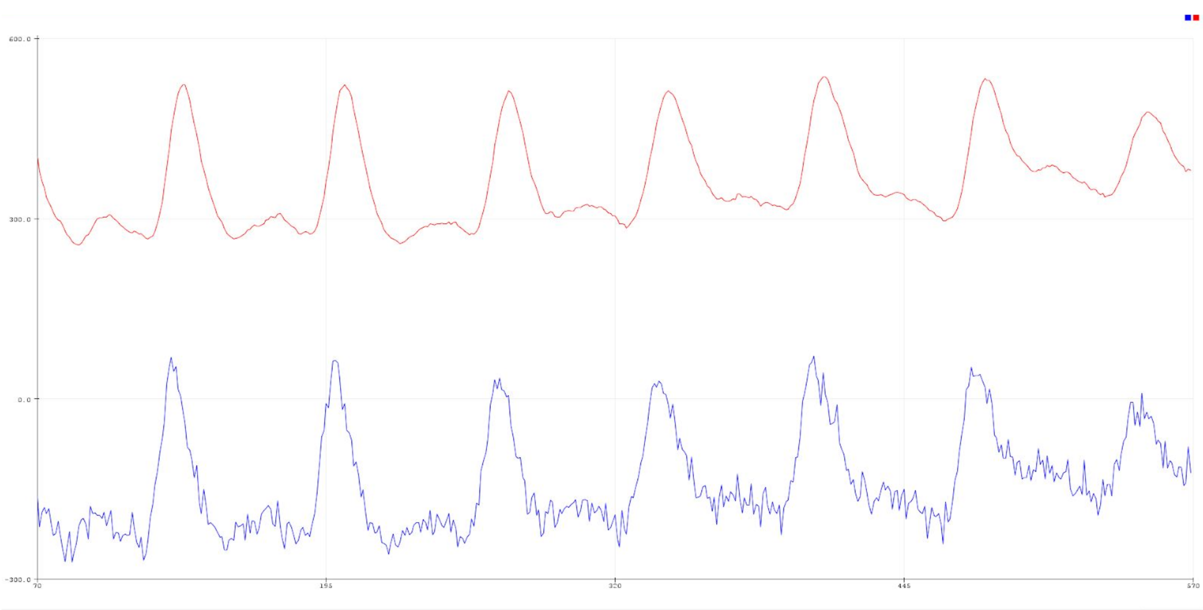
sketch_jan13a_MA §

```
float x[1000] = {-194.7293734,-228.7205774,-241.1012313,-274.3287909,-258.4301079,-244.645154,-267.9013166,-246.6050094,-242.4746091,-262.6250468,-238.1975294,-221.7994023,-235.1426569,-242.8932457,-247.5878307,-218.1253556,-237.4591843,
};
int k=8;
float y[1000];
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:

for ( int i=0; i<1000; i++)
{
  y[i] = 0;
  for (int j=0; j<k; j++)
  {
    y[i] = y[i] + x[i-j]/k;
  }
}
for (int i = 0; i<1000; i++)
{
Serial.print(x[i]);
Serial.print(',');
Serial.println(y[i]+500);
}
}
```

Done Saving.

```
done in 3.607 seconds
Set boot flash true
CPU reset.
```

27 Arduino Due (Programming Port) on COM3

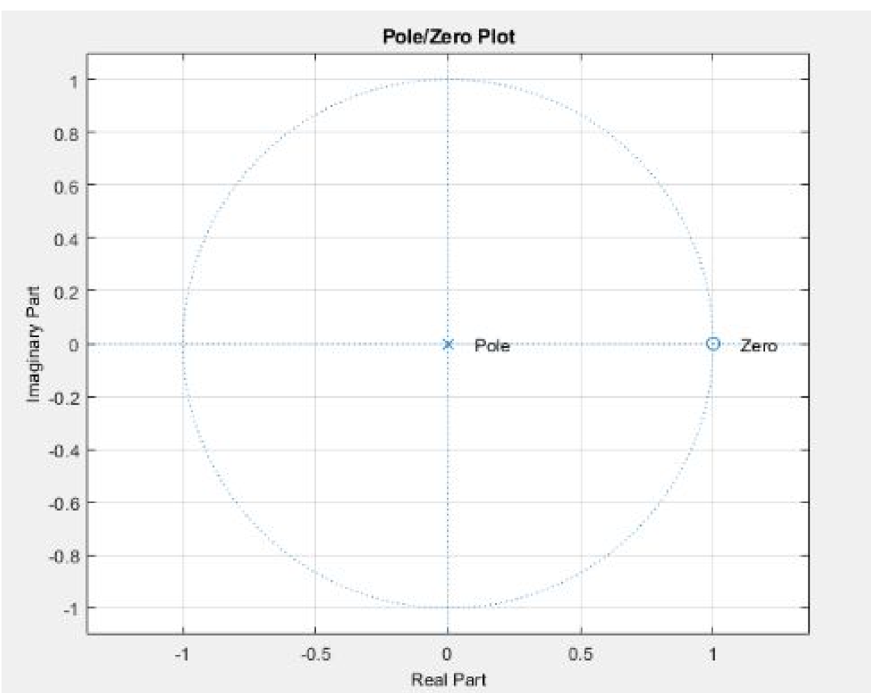


B) DERIVATIVE FILTER, first-order difference

- Derivative filter of first-order difference.
  - It is just a simple difference filter. It emphasizes the slope of the signal.
  - It acts as a high pass filter. The difference between the present input and the past input becomes the output point.
  - It's a backward difference filter.
    - Y[n] = X[n] - X[n-1], where **Y** is the output, **X** is the input and **n** is the sample number.

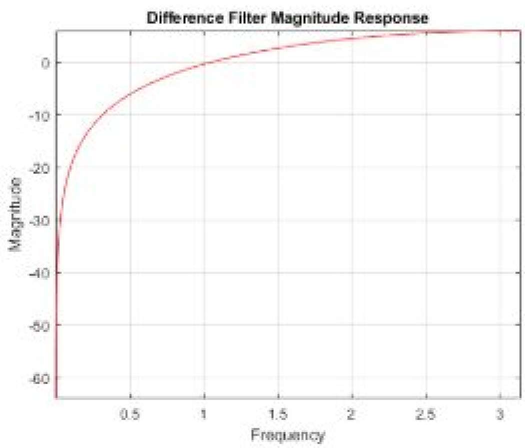
$$y[n] = x[n] - x[n-1]$$

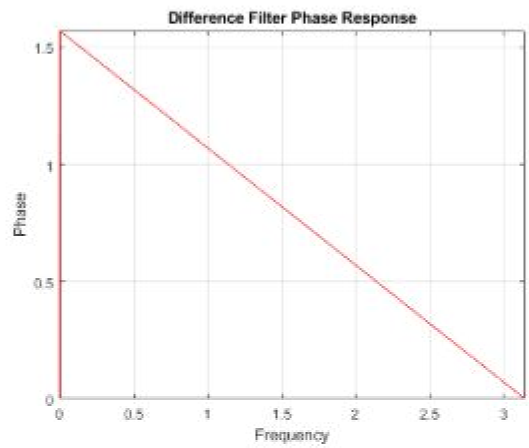
$$H(z) = \frac{Y(z)}{X(z)} = (1 - z^{-1})$$



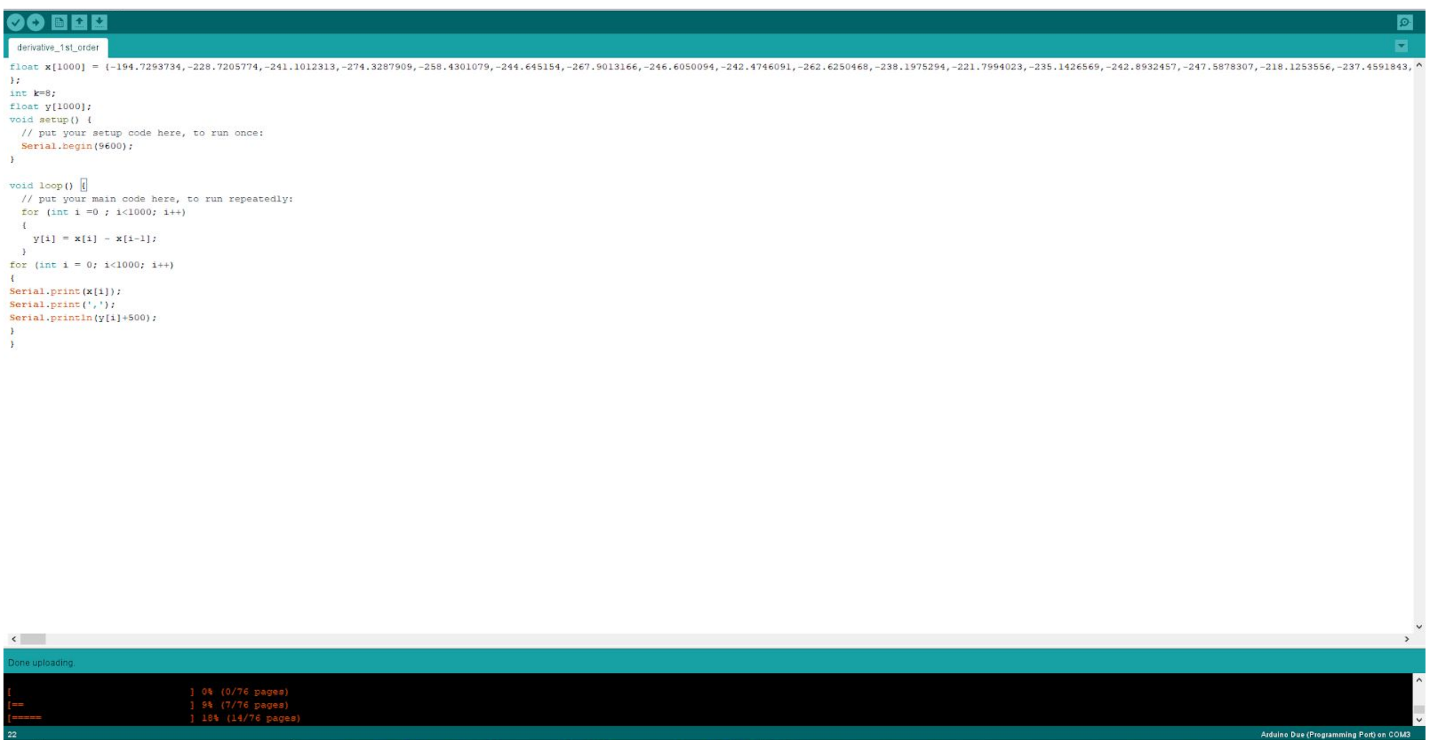
Pole-zero plot

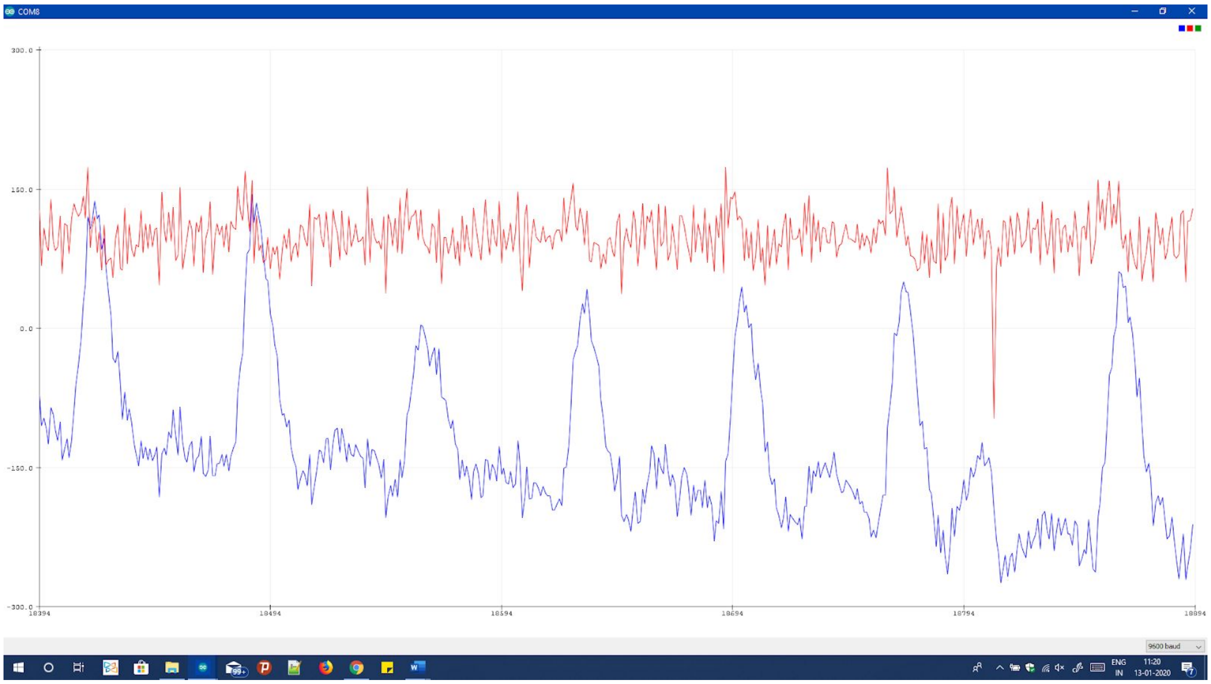
Magnitude and Phase Response of the derivative filter of the first order





B) Derivative filter results



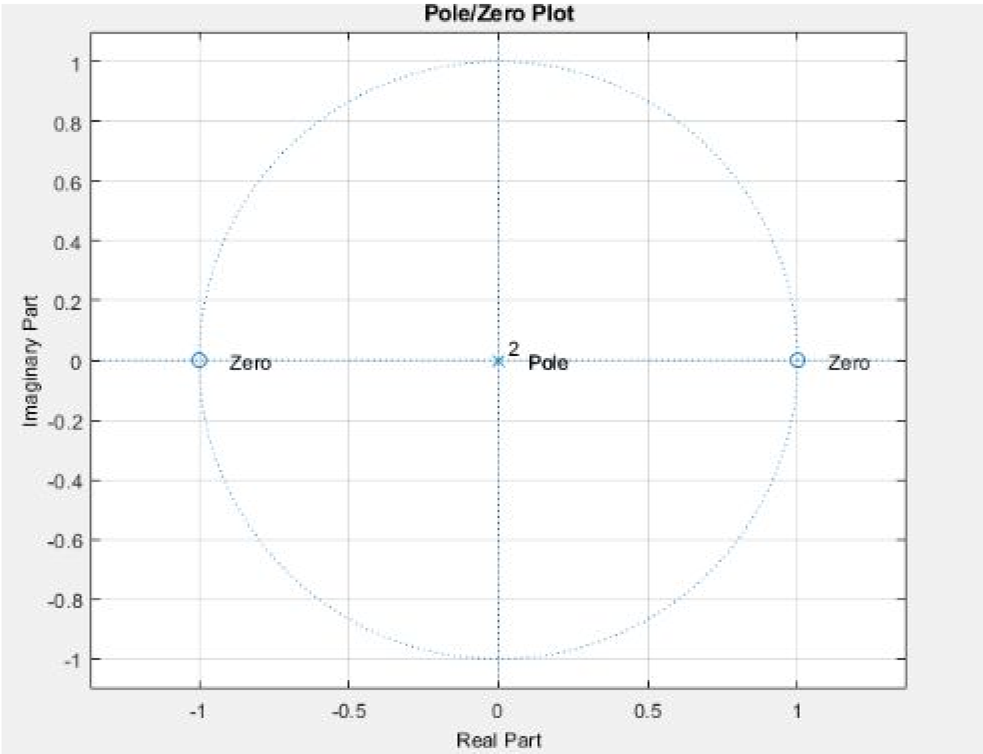


B) DERIVATIVE FILTER, 3 point central difference

- Derivative filter of three point central difference.
  - The difference between the present input and the second past input becomes the output point.
  - It's also a backward difference filter. This filtered output is also much more distorted.
    - Y[n] = X[n] - X[n-2], where **Y** is the output, **X** is the input and **n** is the sample number.

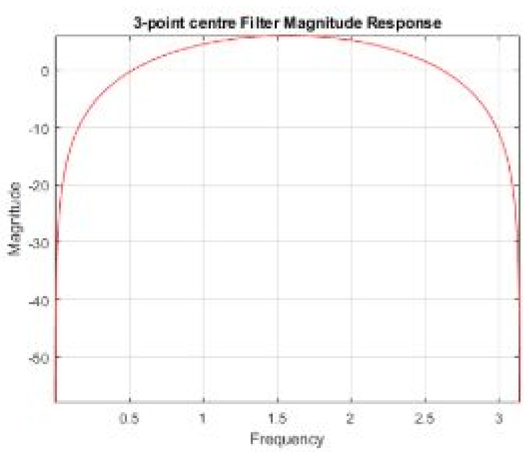
$$y[n] = x[n] - x[n-2]$$

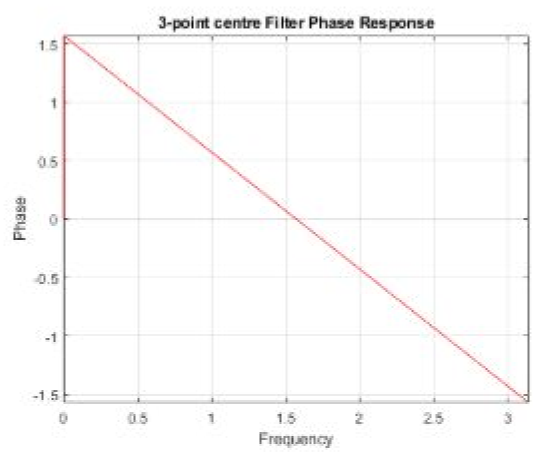
$$Y(z) = (1 - z^{-2})X(z) \qquad H(z) = \frac{Y(z)}{X(z)} = (1 - z^{-2})$$



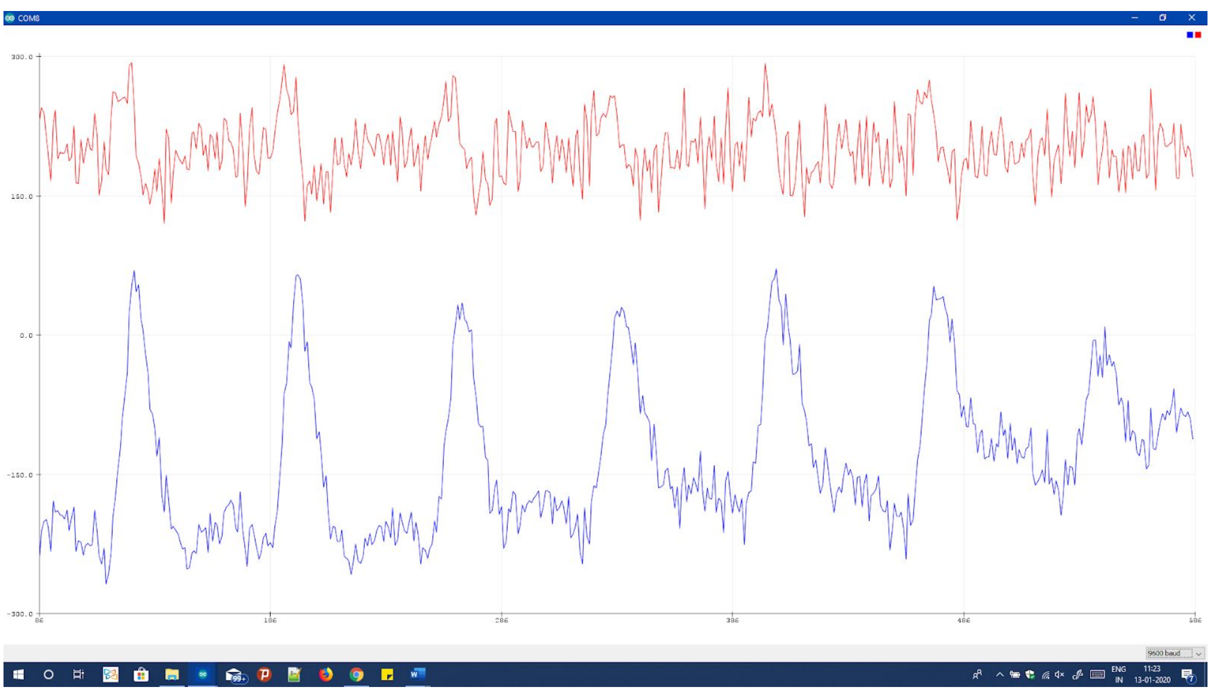
Pole-zero plot

Magnitude and Phase Response of the derivative filter of the first order





C) Derivative filter, 3 point central difference results



C) BASELINE DRIFT REMOVAL

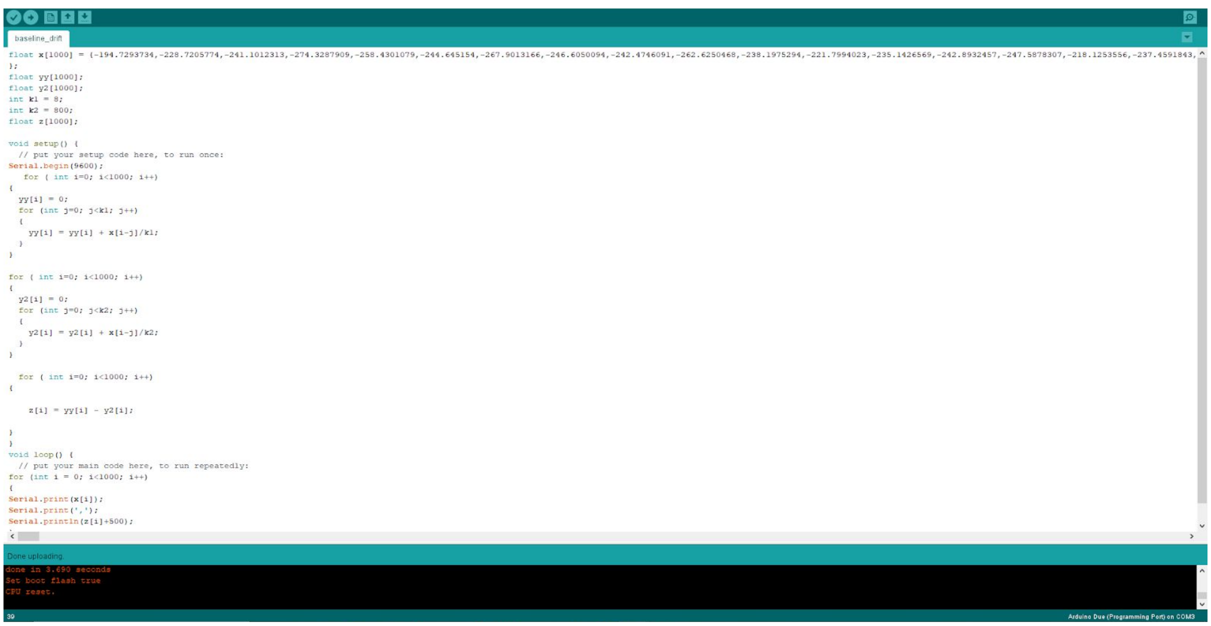
- **Given signal = Original PPG signal + Baseline-drift component + High Frequency components**

- **Original ppg signal** = **Given signal – (high-frequency components + baseline drift signal)**

- Moving average filter of order **8** removes high-frequency components.

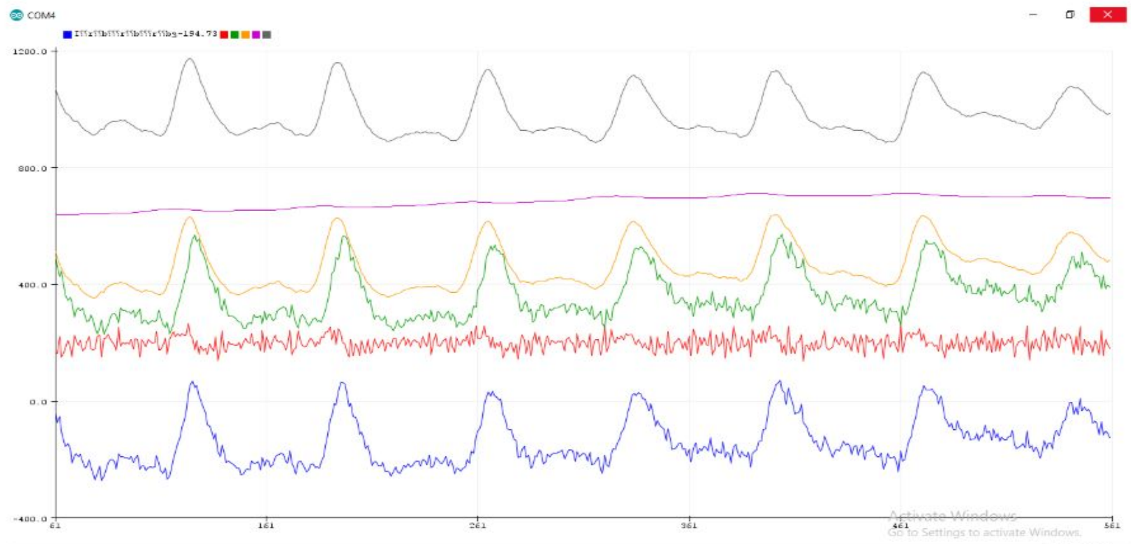
- Moving average filter of order **800** behaves as an LPF with a **very low cut-off frequency.** Thus, we can extract the **Baseline-drift** component.

- Removal of the baseline drift component is done by subtracting the extracted baseline signal from the original signal.

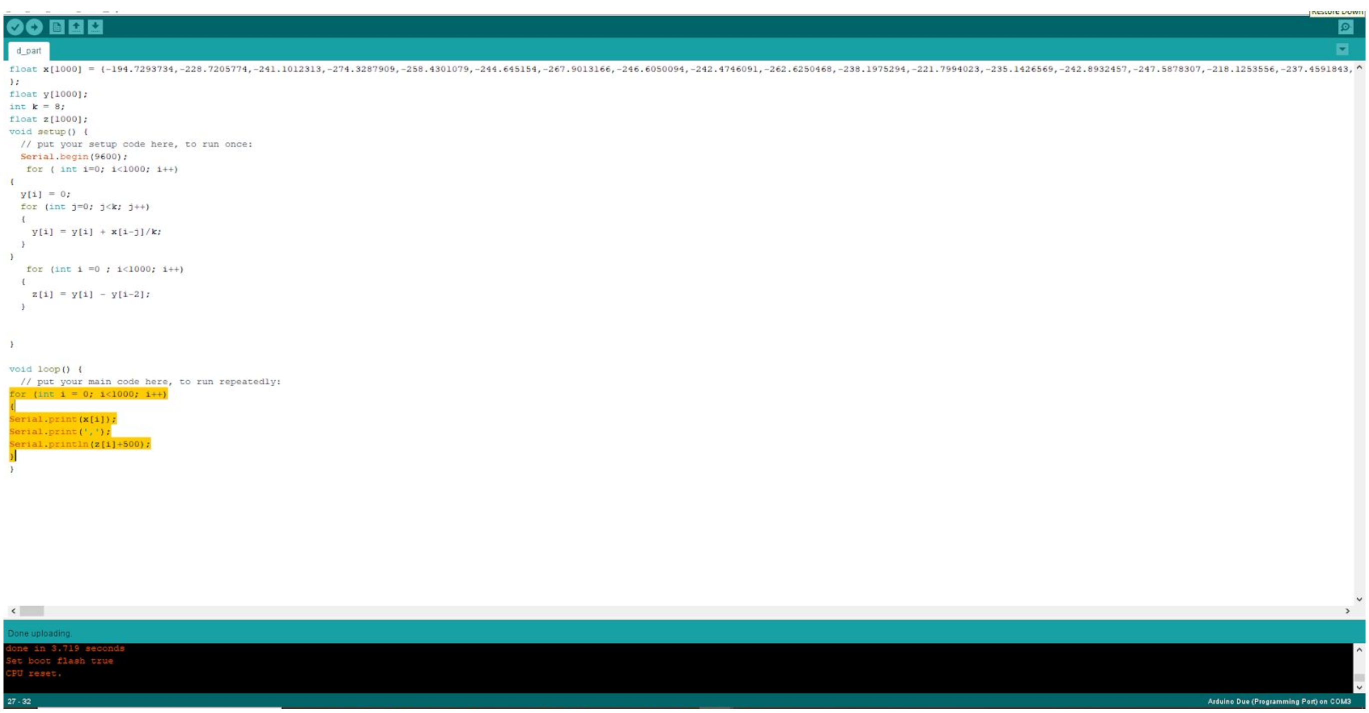


```
float x[1000] = {-194.7293734,-228.7205774,-241.1012313,-274.3287909,-258.4301079,-244.645154,-267.9013166,-246.6050094,-242.4746091,-262.6250468,-238.1975294,-221.7994023,-235.1426569,-242.8932457,-247.5878307,-218.1253556,-237.4591843,
};
float yy[1000];
float y2[1000];
int k1 = 8;
int k2 = 800;
float z[1000];

void setup() {
  // put your setup code here, to run once:
Serial.begin(9600);
   for ( int i=0; i<1000; i++)
{
  yy[i] = 0;
  for (int j=0; j<k1; j++)
  {
    yy[i] = yy[i] + x[i-j]/k1;
  }
}

for ( int i=0; i<1000; i++)
{
  y2[i] = 0;
  for (int j=0; j<k2; j++)
  {
    y2[i] = y2[i] + x[i-j]/k2;
  }
}

  for ( int i=0; i<1000; i++)
{

    z[i] = yy[i] - y2[i];

}
}
void loop() {
  // put your main code here, to run repeatedly:
for (int i = 0; i<1000; i++)
{
Serial.print(x[i]);
Serial.print(',');
Serial.println(z[i]+500);
```

- Removing Baseline Drift:



D) MA followed by DERIVATIVE FILTER



d_part

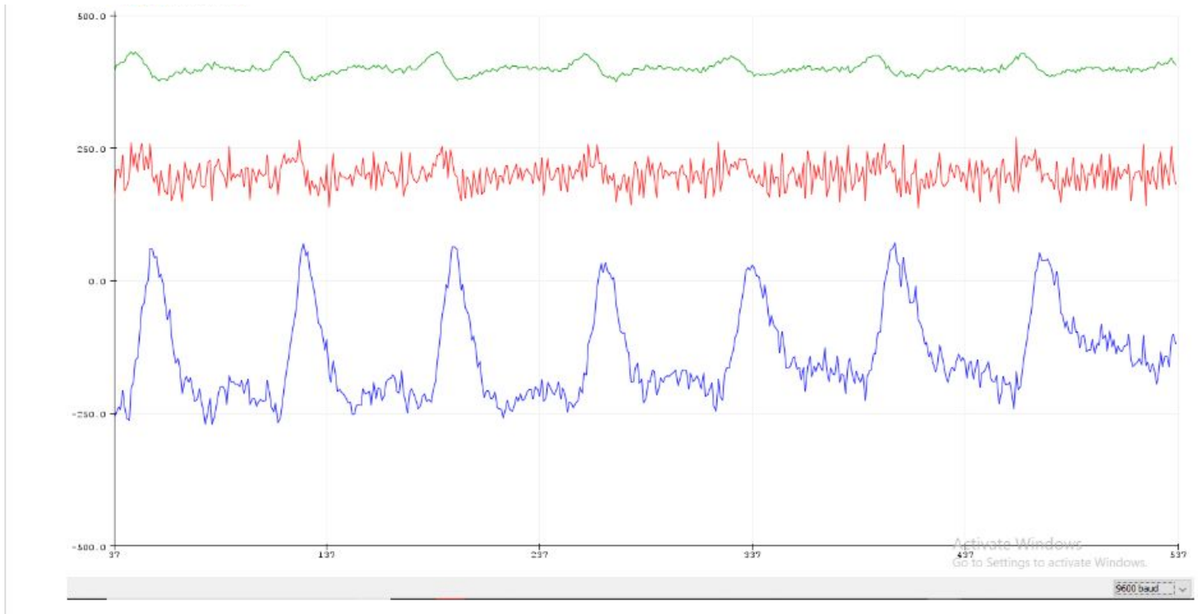
```
float x[1000] = {-194.7293734,-228.7205774,-241.1012313,-274.3287909,-258.4301079,-244.645154,-267.9013166,-246.6050094,-242.4746091,-262.6250468,-238.1975294,-221.7994023,-235.1426569,-242.8932457,-247.5878307,-218.1253556,-237.4591843,
};
float y[1000];
int k = 8;
float z[1000];
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
   for ( int i=0; i<1000; i++)
{
  y[i] = 0;
  for (int j=0; j<k; j++)
  {
    y[i] = y[i] + x[i-j]/k;
  }
}
   for (int i =0 ; i<1000; i++)
  {
    z[i] = y[i] - y[i-2];
  }


}

void loop() {
  // put your main code here, to run repeatedly:
for (int i = 0; i<1000; i++)
{
Serial.print(x[i]);
Serial.print(',');
Serial.println(z[i]+500);
}
}
```

Done uploading.

done in 3.719 seconds
Set boot flash true
CPU reset.

27 - 32

Arduino Due (Programming Port) on COM3



Conclusion:

A combination of filters can do a lot of wonders than a single filter. For example, peak detection, as discussed above, was made possible by the combination of the moving average filter and the difference filter. The MAF smoothens the waveform, removing the high frequency noises, which is followed by the action of derivative filter ie emphasising the high slope portions, thus detecting the peaks.