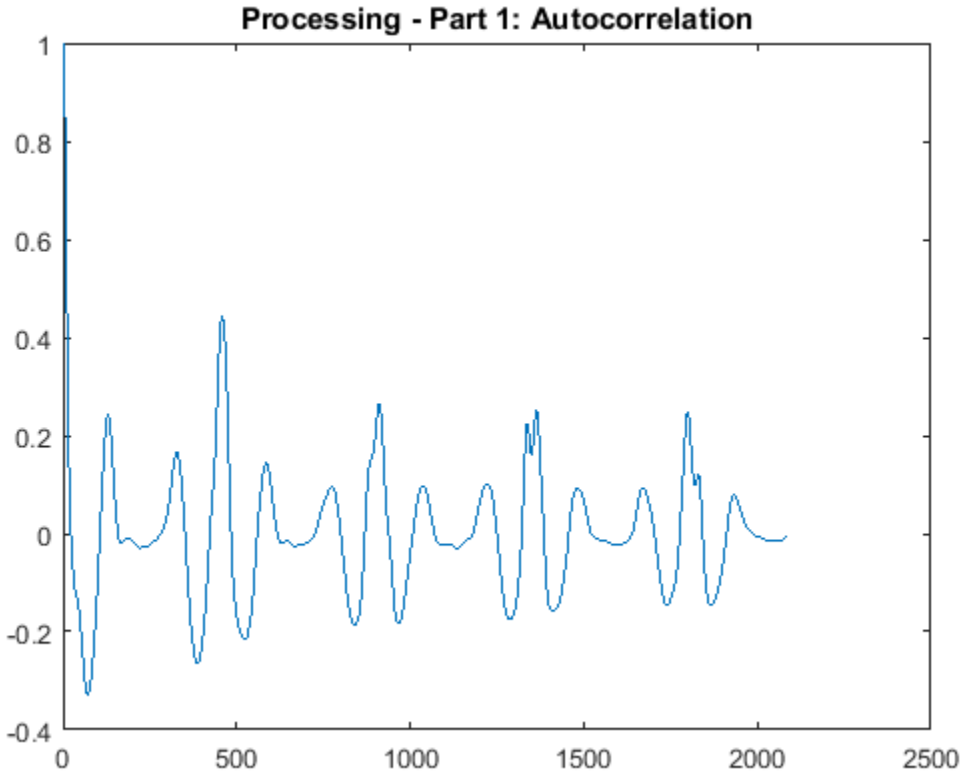
Processing

Table of Contents

Part 1 - Autocorrelation

We plot the autocorrelation of the signal using the autocorr() function. For better visualization, we increase the number of shifts to half of the number of the samples.

```
figure(1);
sig = snr_inc; % final preprocessed signal
ac = autocorr(sig, length(sig)/2);
plot(ac);
title('Processing - Part 1: Autocorrelation');
```



Part 2 - A Visual Guess

The number of samples between first two peaks visually is about 500.

```
nv = 500;
display(nv);
```

```
nv =

   500
```

Part 3 - Find Peaks

We can find the peaks of the signal using findpeak() function, but we do not want all the points. Since the maximum heart beat per minute for an average person is 200 bps, the minimun time between heart strokes is 0.3 second. Then the minimun number of samples between would be 0.3 * samplingFrequency. The result is still not what we want because there are points (peaks) that do not belong to a heart stroke. Heart stroke peaks have a higher value than ther adjacent peaks. As the result, we apply findpeaks() again to detain the desired peaks.

```
[sval, slag] = findpeaks(ac, 'MinPeakDistance', 0.3*fs); % finds peaks
 of the autocorrelation
[lval, llag] = findpeaks(sval);
pts = slag(llag); % heart stroke sample #
display(pts);
ns = [pts(1); diff(pts)]; % nummber of samples between adjacent peaks
display(ns);
```

```
pts =

         459
         911
        1364
        1799
```

```
ns =

   459
   452
   453
   435
```

Part 4 - Average Sample Numbers

We calculate the average number of samples between adjacent [heart stroke] peaks using mean() function.

```
avg_s = mean(ns);
```

```
display(avg_s);
```

```
avg_s =

  449.7500
```

Part 5 - Average Time

The average time between heart strokes are calculated by dividing the number of samples by the sampling frequency.

```
avg_t = avg_s./fs; % average time between heart strokes
display(avg_t);
```

```
avg_t =

    0.8995
```

Part 6 - Heart Rate (HR)

The Heart Rate (HR) can be obtained by dividing 60 by the average time between heart strokes.

```
hr = 60/avg_t; % caluclates heart rate
display(hr);
```

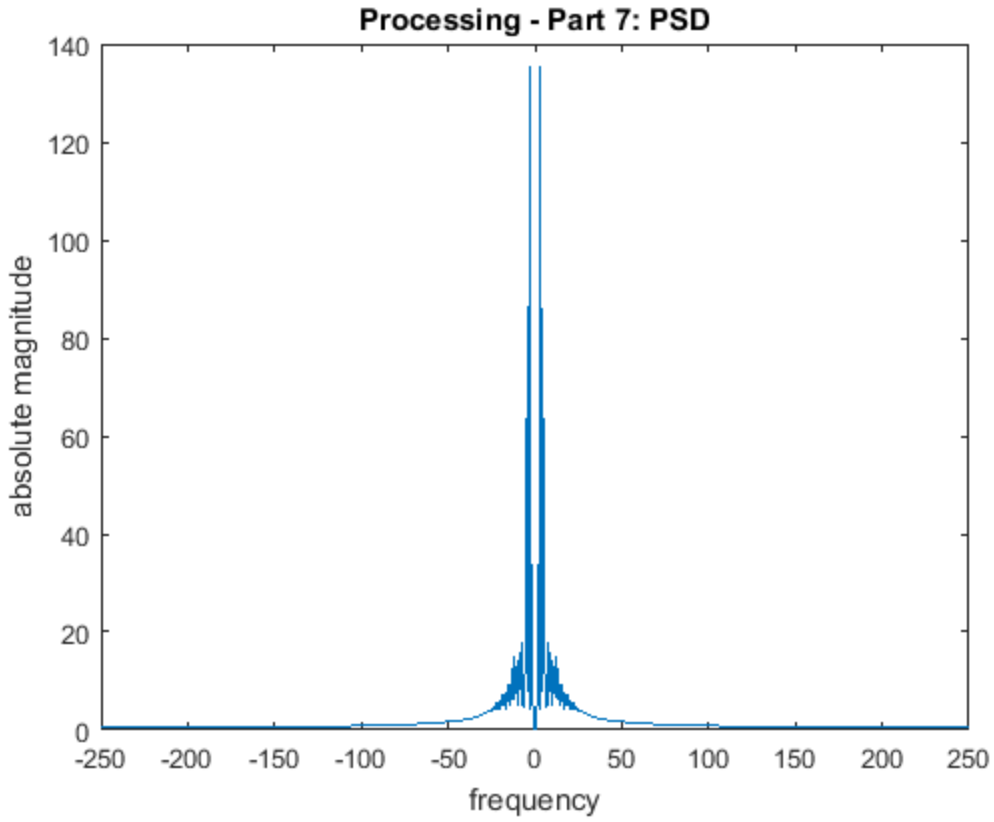
```
hr =

   66.7037
```

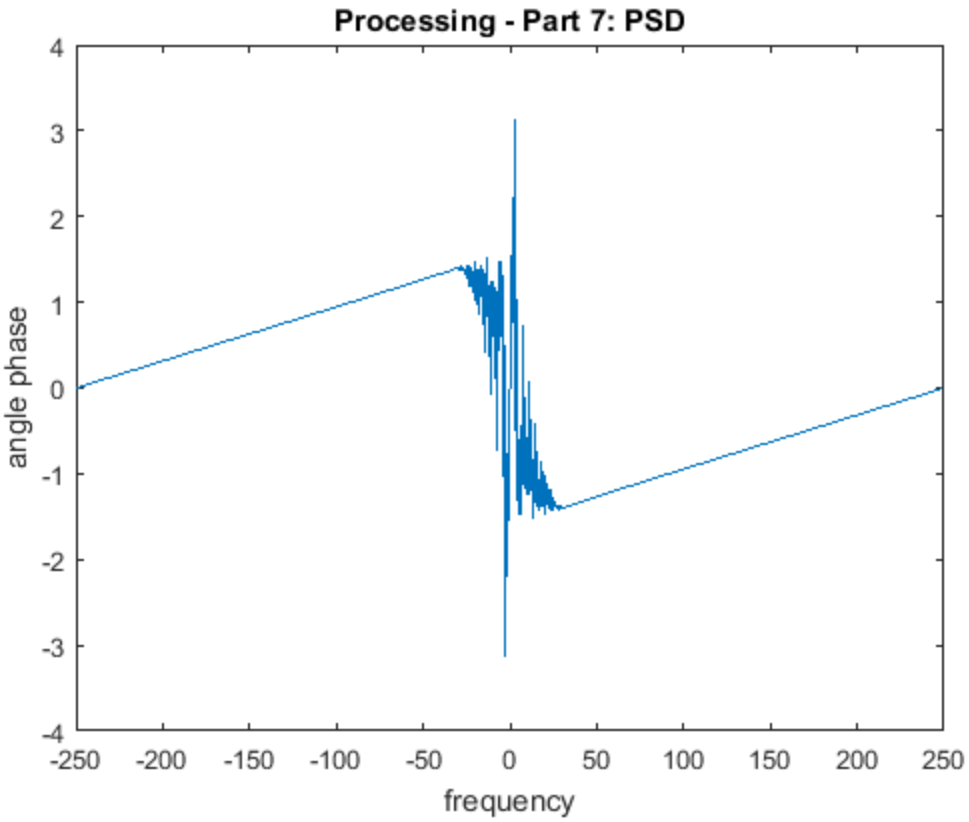
Part 7 - Power Spectrum Density (PSD)

Since PSD of the signal is its autocorrelation in frequency domain, we simply calculate the Fast Fourier Transform of the autocorrelation.

```
ac_f = fftshift(fft(ac));
n_ac = length(ac_f);
fac_f = (-n_ac/2:n_ac/2-1)*(fs/n_ac);
figure(2);
plot(fac_f, abs(ac_f));
xlabel('frequency');
ylabel('absolute magnitude');
title('Processing - Part 7: PSD');
figure(3);
plot(fac_f, angle(ac_f));
xlabel('frequency');
```

```
ylabel('angle phase');
title('Processing - Part 7: PSD');
```





Published with MATLAB® R2015a