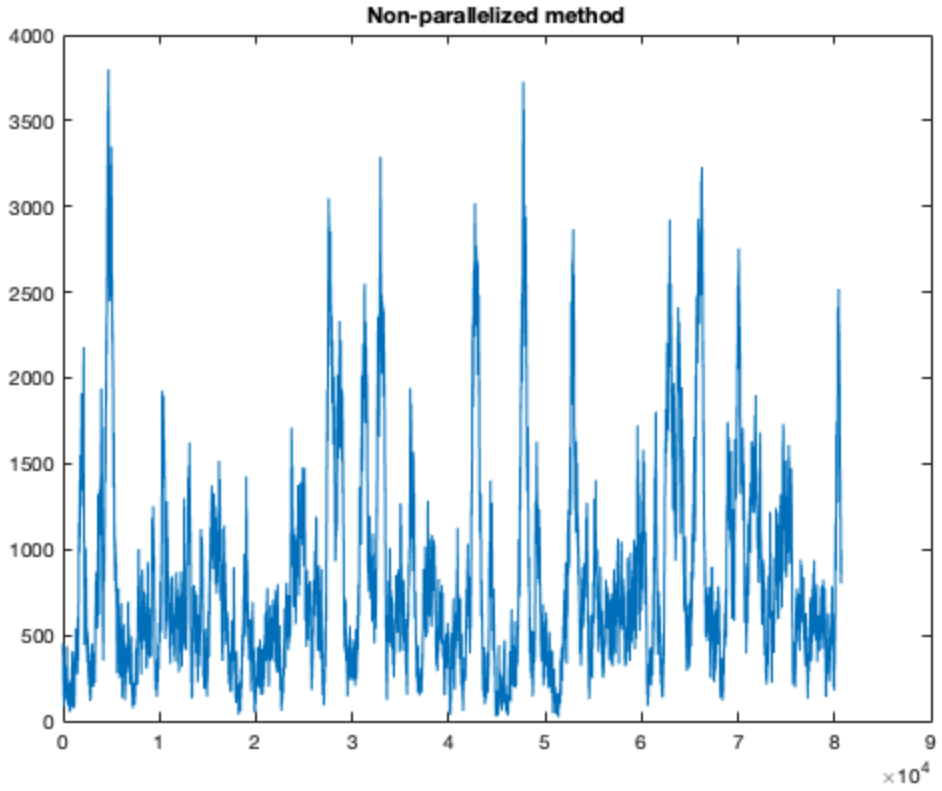
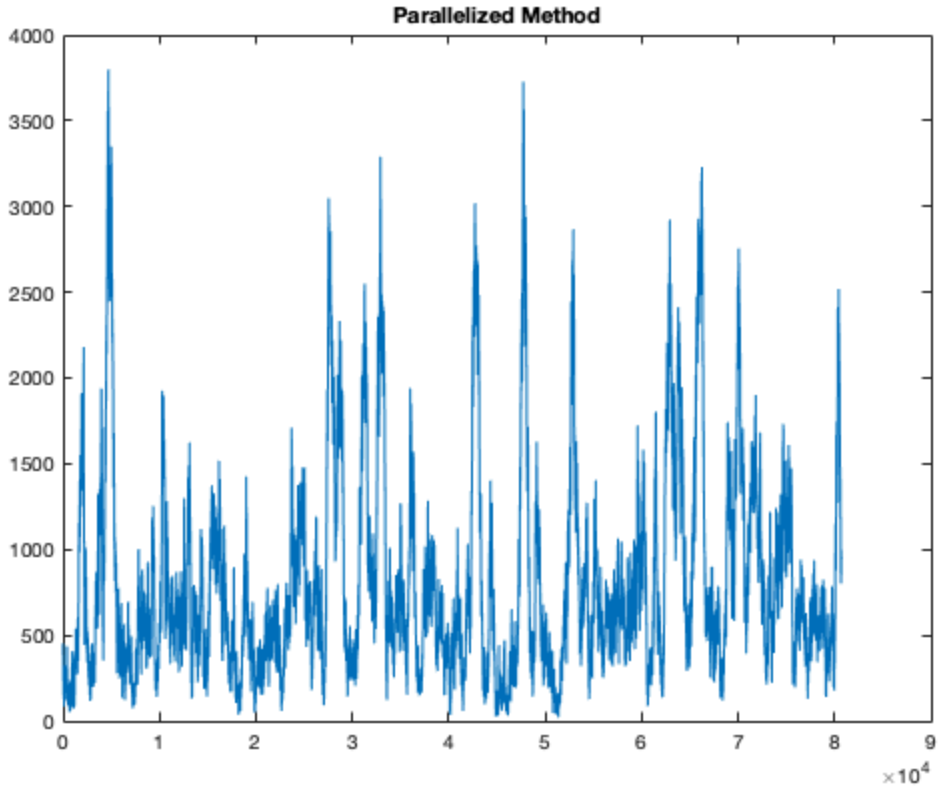
Lab 4 - Tyler Bradley

```
clc;close all;clear;


% load data
% load the DNA sequence we want to work with
hbb = genbankread('hbb_region_chr11.gb');
% display the method we are using
disp('Non-Parallelized Method:')
% use tic to mark a start time
tic
% run STFT function (provided on BBLEARN)
three_base_non_par = threebasefreq_stft(hbb.Sequence,1000,1024);

% use toc to mark an end time
% tic & toc will print the elapsed time for commands between tic and
 toc
% in our case, it print out how long it took to run STFT function
toc
% plot the result
figure(1)
plot(three_base_non_par)
title("Non-parallelized method")

% define the number of workers
worker_num = 2;

% initiate the parallel process
parpool(worker_num);


disp("Parallelized Method:")
tic

% define empty vector for parallel output
three_base_par = [];
parfor i=1:worker_num
  % you can run any command you want here.
  % in this sample code, I just print out the iterator index
  three_base_par_piece = threebasefreq_par(hbb.Sequence, 1000, 1024,
 worker_num, i);
  three_base_par = [three_base_par, three_base_par_piece];
end
toc

figure(2)
plot(three_base_par)
title("Parallelized Method")

% Test whether the values in the nonparallel and parallel are equel
ASE = sum(abs(three_base_par - three_base_non_par))
```

```
delete(gcp('nocreate'));

% Given non-parallel function
function Threebaseperiodicity_vs_position = threebasefreq_stft
 (DNA_SEQUENCE, WINDOW_LENGTH, NFFT)
DNA_len = length(DNA_SEQUENCE);
Threebaseperiodicity_vs_position=zeros(1,DNA_len-WINDOW_LENGTH+1);
coding = DNA_SEQUENCE;
coding_A = (upper(coding)=='A'); % find A bases and set them to 1
coding_T = (upper(coding)=='T'); % find T bases and set them to 1
coding_G = (upper(coding)=='G'); % find G bases and set them to 1
coding_C = (upper(coding)=='C'); % find C bases and set them to 1
for i = 1:DNA_len-WINDOW_LENGTH+1
    stc_A = coding_A(i:i+WINDOW_LENGTH-1);
    stc_T = coding_T(i:i+WINDOW_LENGTH-1);
    stc_G = coding_G(i:i+WINDOW_LENGTH-1);
    stc_C = coding_C(i:i+WINDOW_LENGTH-1);
    STFT = abs(fft(stc_A,NFFT)).^2+abs(fft(stc_T,NFFT)).^2 ...
    +abs(fft(stc_G,NFFT)).^2+abs(fft(stc_C,NFFT)).^2; % FFT of the
 sequence
    Threebaseperiodicity_vs_position(i)=STFT(floor(NFFT/3));
end
end

% Custom Parallel function
function output = threebasefreq_par(DNA_SEQUENCE, WINDOW_LENGTH, NFFT,
 division, dindex)
% Calculate the middle point of the sequence length
break_points = (length(DNA_SEQUENCE)-WINDOW_LENGTH)/division;

% Break the sequence into either the segment corresponding to dindex
seq = DNA_SEQUENCE((dindex
+break_points*(dindex-1)):break_points*dindex+WINDOW_LENGTH);

% Run the shortened sequence through the original function
DNA_len = length(seq);
output=zeros(1,DNA_len-WINDOW_LENGTH+1);
coding = seq;
coding_A = (upper(coding)=='A'); % find A bases and set them to 1
coding_T = (upper(coding)=='T'); % find T bases and set them to 1
coding_G = (upper(coding)=='G'); % find G bases and set them to 1
coding_C = (upper(coding)=='C'); % find C bases and set them to 1
for i = 1:DNA_len-WINDOW_LENGTH+1
    stc_A = coding_A(i:i+WINDOW_LENGTH-1);
    stc_T = coding_T(i:i+WINDOW_LENGTH-1);
    stc_G = coding_G(i:i+WINDOW_LENGTH-1);
    stc_C = coding_C(i:i+WINDOW_LENGTH-1);
    STFT = abs(fft(stc_A,NFFT)).^2+abs(fft(stc_T,NFFT)).^2 ...
    +abs(fft(stc_G,NFFT)).^2+abs(fft(stc_C,NFFT)).^2; % FFT of the
 sequence
    output(i)=STFT(floor(NFFT/3));
end
end
```

```
Non-Parallelized Method:
Elapsed time is 5.559215 seconds.
Starting parallel pool (parpool) using the 'local' profile ...
connected to 2 workers.
Parallelized Method:
Elapsed time is 4.643954 seconds.

ASE =

     0

Parallel pool using the 'local' profile is shutting down.
```





Published with MATLAB® R2018b