NKHBES001

EEE4120F Prac 1

Introduction

This report investigates the performance difference between optimized and non-optimized functions, analysing their speed. Additionally, it examines the correlation between shifted sine waves.

Methods

Part 3 step 4

In step 4 of part 3, generate noise using a for loop. Fill an array with random numbers, then normalize it to have an amplitude of 1:

```
tic()
% Generate white noise using a for loop
white_noise = zeros(1, num_samples);
for i = 1:num_samples
    white_noise(i) = randn(); % Generate a single random value
end
toc()
```

Figure 1: custom white noise generator.

Part 4

To implement the Pearson product moment correlation coefficient, was done by finding the mean of each data set, then the covariance of each data set, and then the standard deviation of each data set, divided the covariance by the product of the two standard deviations:

```
function correlation_coefficient = mycorr(data1, data2)
% CALCULATE_CORRELATION Calculates the correlation coefficient between two data sets
%   data1: First data set
%   data2: Second data set

% Check if the input data sets have the same length
if numel(data1) ~= numel(data2)
    error('Data sets must have the same length');
end

% Calculate the means of the two data sets
mean_data1 = mean(data1);
mean_data2 = mean(data2);

% Calculate the covariance
covariance = sum((data1 - mean_data1) .* (data2 - mean_data2)) / numel(data1);

% Calculate the standard deviations
std_data1 = std(data1);
std_data2 = std(data2);

% Calculate the correlation coefficient
correlation_coefficient = covariance / (std_data1 * std_data2);
end
```

Figure 2: custom Pearson product implementation

A quick validation of my method produces these differences from the inbuilt correlation calculator:

```
>> mycorr
   2.0833e-07

   2.0832e-07

    0.0329
```

Figure 3: differences between build in function and mine

The differences round out to 0, so my method is fairly accurate.

Part 4 step 3

A sinusoid was instatiated with an amplitude of 1 and a frequency of 1/10 and varied the sampling rate and the amount of time a sinusoid is shifted by.

```
amplitude = 1;       % Amplitude of the sine wave
frequency = 1/10;     % Frequency of the sine wave (in Hz)
phase = 0;           % Phase of the sine wave (in radians)
duration = 2 * 10;    % Duration of the sine wave to ensure at least 2 full periods
sampling_rate = 1000; % Sampling rate (number of samples per second)

% Create a time vector for the original sine wave
t_original = linspace(0, duration, duration * sampling_rate);

% Generate the original sine wave
sine_wave_original = amplitude * sin(2 * pi * frequency * t_original + phase);

% Create a time vector for the shifted sine wave
shift_amount = 10; % Shift by half a period
t_shifted = t_original + shift_amount;

% Generate the shifted sine wave
sine_wave_shifted = amplitude * sin(2 * pi * frequency * t_shifted + phase);
```

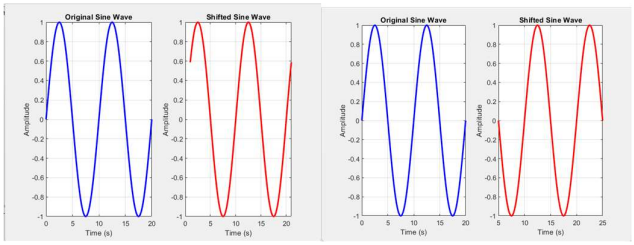

Figure 4: Examples of functions compared.

It is expected that the correlation coefficient will be sinusoidal, where in when the shift increases the correlation coefficient should decrease to a point, and then start increasing again when the wave goes back in phase, with the period set to 10 seconds, it should be expected that the correlation coefficient should go back to 1 when then sine is shifted by 10 seconds.

Results & Discussion

Part 3 step 4

Duration of the clip	Using a for loop	Built in method	Speed up
10	0.018382	0.01052	1.748
100	0.158877	0.04842	3.28123
1000	1.693087	0.51195	3.30713
10000	17.10276	5.2884	3.23402
15000	27.1384553	8.33977	3.2541

By varying the duration of each clip generated, it can be noted that using a for loop is generally slower than using the built in method, it is about 3 times slower for larger loads.

Part 4

Sample size	My Corr speed(ms)	Corr speed(ms)	speed up
100	1.625	1.076	0.662154
1000	0.606	0.307	0.506601
10000	0.695	0.316	0.454676

The built in correlation function is faster than then custom made one by a factor of about 2.

Part 4 step 3

Number of samples	Amount of time shifted(s)	Correlation co-efficient
100	1	0.8089
	2	0.3089
	5	-1
	7	-0.3089
	10	1
1000	1	0.8089
	2	0.3089
	5	-1
	7	-0.3089
	10	1
10000	1	0.8089
	2	0.3089
	5	-1
	7	-0.3089
	10	1

As expected, the correlation coefficient is itself be sinusoidal, where in when the time shift increases the correlation coefficient should decrease to a point, and then start increasing again when the wave goes back in phase.

Conclusion

In conclusion, it was found that the inbuilt methods tend to scale better than un optimized versions of the code.

It was found that shifting a sinusoid has the effect of creating a sinusoidal correlation with its original.