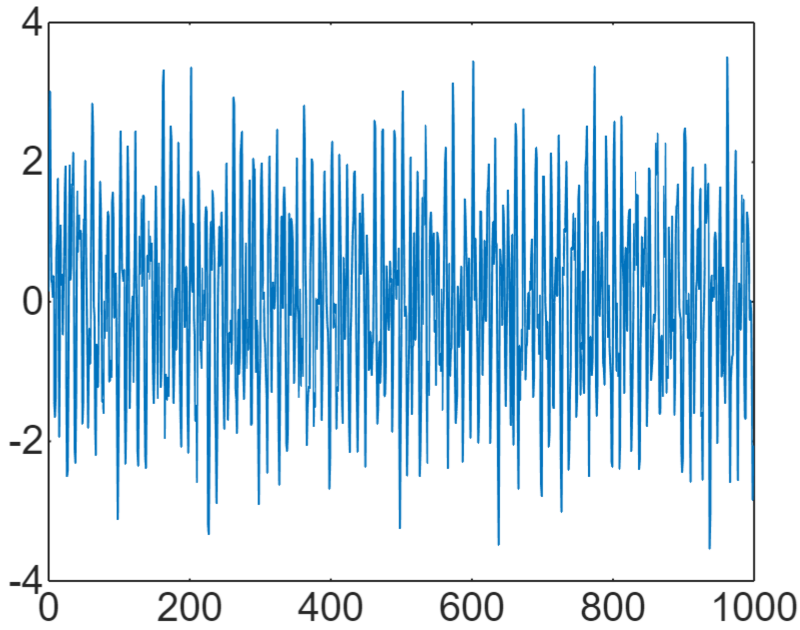
Fast Fourier Transform (FFT)

In MATLAB, the FFT (Fast Fourier Transform)[1] function is straightforward and commonly used for signal and data analysis. MATLAB's `fft` function is highly optimized and provides easy access to analyse signals in the frequency domain.

```
% Parameters
Fs = 1000;          % Sampling frequency (Hz)
T = 1/Fs;           % Sampling period (s)
L = 1000;           % Length of signal
t = (0:L-1)*T;      % Time vector

% Generate a signal with two frequencies
a1 = 0.6;           % Amplitude of first sine wave
f1 = 70;            % Frequency of first sine wave (Hz)
a2 = 1.3;           % Amplitude of second sine wave
f2 = 100;           % Frequency of second sine wave (Hz)
a3 = 1.0;           % Amplitude of second sine wave
f3 = 180;           % Frequency of third sine wave (Hz)
signal = a1* sin(2 * pi * f1 * t) + a2*sin(2 * pi * f2 * t) + a3*sin(2 * pi * f3 *
t);

% Add some noise (optional)
signal = signal + 0.5 * randn(size(t));
plot(L*t,signal);

% Compute the FFT
Y = fft(signal);

% Compute the two-sided spectrum, then one-sided spectrum
P2 = abs(Y / L);        % Two-sided spectrum
P1 = P2(1:L/2+1);       % Single-sided spectrum
P1(2:end-1) = 2 * P1(2:end-1);

% Define the frequency axis
f = Fs * (0:(L/2)) / L;

% Plot the single-sided amplitude spectrum
figure;
plot(f, P1);
title('Single-Sided Amplitude Spectrum of Signal');
xlabel('Frequency (Hz)');
ylabel('|P1(f)|');
```

[1] Fast Fourier Transform (FFT)

The input signal is made up of three sinusoidal functions at frequencies of 70, 100 and 180Hz with respective amplitudes of 0.6, 1.3 and 1.0. Random noise is then added to the signal. A graph of the input signal is given in the following figure.



The output frequency plot is shown in the following figure:

-