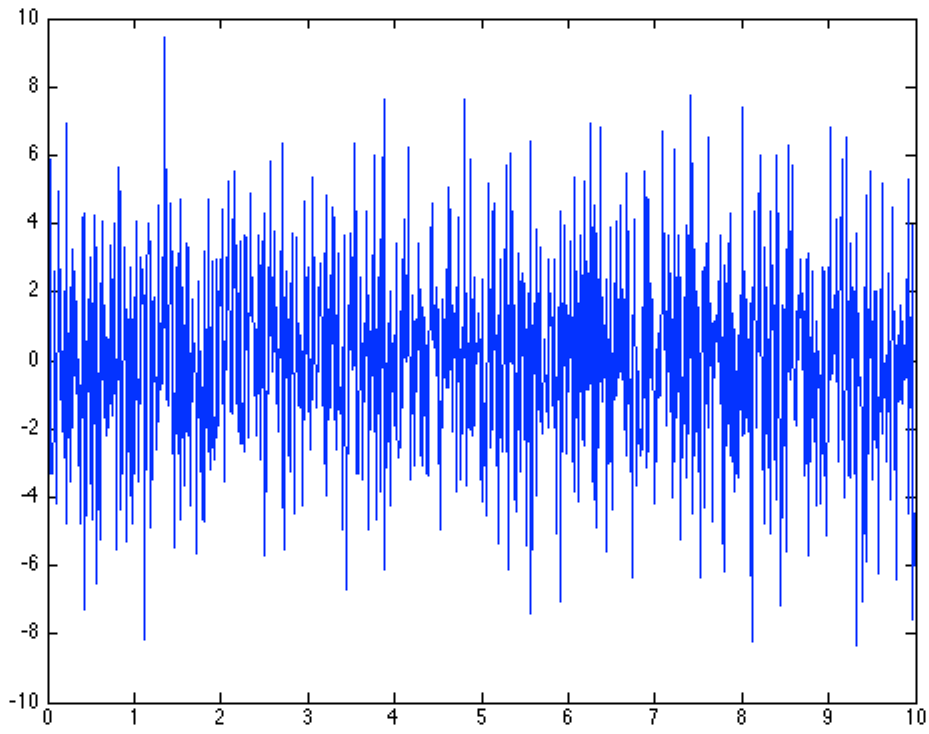
Contents

For example, consider the following data x with two component

frequencies of differing amplitude and phase buried in noise:

```
fs = 100;                           % Sample frequency (Hz)
t = 0:1/fs:10-1/fs;                 % 10 sec sample
x = (1.3)*sin(2*pi*15*t) ...        % 15 Hz component
  + (1.7)*sin(2*pi*40*(t-2)) ...    % 40 Hz component
  + (2.5)*randn(size(t));           % Gaussian noise;

figure(1)
plot(t,x)
```



Use fft to compute the DFT y and its power:

```
m = length(x);          % Window length
n = pow2(nextpow2(m));  % Transform length
```

```
y = fft(x,n);             % DFT
f = (0:n-1)*(fs/n);       % Frequency range
power = y.*conj(y)/n;     % Power of the DFT
```

nextpow2 finds the exponent of the next power of two greater

than or equal to the window length (ceil(log2(m))), and pow2 computes the power. Using a power of two for the transform length optimizes the FFT algorithm, though in practice there is usually little difference in execution time from using n = m.

To visualize the DFT, plots of abs(y), abs(y).^2, and log(abs(y))

are all common. A plot of power versus frequency is called a periodogram:

```
figure(2)
plot(f,power)
xlabel('Frequency (Hz)')
ylabel('Power')
title('{\bf Periodogram}')
```