```
%----------------------------------------------------------------------
%
%  MATLAB evaluation of DFT Interpolation Formulas from Rick Lyons
%  DSPrelated blog April 2018.
%
%  Accepts a non-integer line index (k) and returns the interpolated
%  spectral value from the DFT. The reason for this code is to demonstrate
%  an error in the calculation using Lyon's Eq. (3).
%
%  Uses a length 16 DFT, and compares results with a zero-padded length 128
%  DFT - which should give the same result (for the appropriate line when
%  the fractional part of k is a multiple of 1/8.
%
%  Note: The correction to the observed error using Lyons Eq. (3) is to
%  simply replace exp(j*2*pi*k) in the numerator with exp(-j*2*pi*k).
%
%  D.R. April 21, 2018
%
%-----------------------------------------------------------------------
clc;
% create two FFTs
f_16  = zeros(1,16);
f_128 = zeros(1,128);
for i=1:6
    f_16(i)  = 1.0;
    f_128(i) = 1.0;
end
N_16  = 16;
N_128 = 128;
F_16  = fft(f_16);
F_128 = fft(f_128);
%select a frequency for interpolation
%
k = 4.875;   % Corresponds to line 8*3.375 + 1 = 28 in the length 128 FFT
%
% Estimate interpolated component using Lyons & Rowell equations:
sum1 = 0.0;
sum1_corrected = 0.0;
sum2 = 0.0;
sum3 = 0.0;
sum4 = 0.0;
for m = 1:16
  m1 = m-1;
  sum1 = sum1 + F_16(m)*(1-exp(1i*2.0*pi*k))  /...
          (1.0 - exp(-1i*2.0*pi*(k-m1)/N_16));                          % Lyons  Eq.(3)
  sum1_corrected = sum1_corrected + F_16(m)*(1-exp(-1i*2.0*pi*k))  /...
          (1.0 - exp(-1i*2.0*pi*(k-m1)/N_16));                          % Lyons  Eq.(3) (corrected)
  sum2 = sum2 + F_16(m)*exp(-1i*2*pi*k/N_16*((N_16-1)/2))*sin(pi*k)/...
          (exp(1i*pi*m1/N_16)*sin(pi*(k-m1)/N_16));                     % Lyons  Eq.(2)
  sum3 = sum3 + F_16(m)*(1-exp(1i*2.0*pi*(m1-k)))/...
          (1.0 - exp(1i*2.0*pi*(m1-k)/N_16));                           % Rowell Eq.(3)
  sum4 = sum4 + F_16(m)*exp(-1i*pi*(k-m1)*(1-1/N_16))*sin(pi*(k-m1))/...
          sin(pi*(k-m1)/N_16);                                          % Rowell Eq.(4)
end
```

```
Lyons3  = sum1/N_16;
Lyons3a = sum1_corrected/N_16;
Lyons2  = sum2/N_16;
Rowell3 = sum3/N_16;
Rowell4 = sum4/N_16;
%
fprintf('Evaluation of DFT frequency domain interpolation formulas for k = %.3f\n', k);
fprintf('Lyons Eq(3) (as published):\t magnitude: %f,\t phase: %f,\t (%f + j%f)  <-- error!\n',...
    abs(Lyons3), angle(Lyons3), real(Lyons3),imag(Lyons3));
fprintf('Lyons Eq(3) (corrected):\t magnitude: %f,\t phase %f,\t (%f +j%f)\n',...
    abs(Lyons3a), angle(Lyons3a), real(Lyons3a),imag(Lyons3a));
fprintf('Lyons Eq(2):             \t magnitude: %f,\t phase %f,\t (%f +j%f)\n',...
    abs(Lyons2), angle(Lyons2), real(Lyons2),imag(Lyons2));
fprintf('Rowell Eq(3):            \t magnitude: %f,\t phase %f,\t (%f +j%f)\n',...
    abs(Rowell3), angle(Rowell3), real(Rowell3),imag(Rowell3));
fprintf('Rowell Eq(4):            \t magnitude: %f,\t phase %f,\t (%f +j%f)\n',...
    abs(Rowell4), angle(Rowell4), real(Rowell4),imag(Rowell4));
FFT_128_k = F_128(8*k+1);
fprintf('From length 128 FFT:     \t magnitude: %f,\t phase %f,\t (%f +j%f)\n',...
    abs(FFT_128_k), angle(FFT_128_k), real(FFT_128_k),imag(FFT_128_k));
```

MATLAB Output:

```
Evaluation of DFT frequency domain interpolation formulas for k = 4.875
Lyons Eq(3) (as published):  magnitude: 0.628807, phase: 0.711767, (0.476138 + j0.410719)  <-- error!
Lyons Eq(3) (corrected):  magnitude: 0.628807, phase -1.644427, (-0.046258 +j-0.627103)
Lyons Eq(2):              magnitude: 0.628807, phase -1.644427, (-0.046258 +j-0.627103)
Rowell Eq(3):             magnitude: 0.628807, phase -1.644427, (-0.046258 +j-0.627103)
Rowell Eq(4):             magnitude: 0.628807, phase -1.644427, (-0.046258 +j-0.627103)
From length 128 FFT:      magnitude: 0.628807, phase -1.644427, (-0.046258 +j-0.627103)
```