POWER SPECTRAL DENSITY CALCULATION VIA MATLAB Revision C

By Bob Light

June 22, 2000

Introduction

A power spectral density function can be calculated via the MATLAB PSD() command, which is part of the Signal Processing Toolbox. The PSD() command, however, is not available in every MATLAB software configuration. Thus, a more fundamental approach is needed.

This paper gives the source code for calculating the power spectral density using MATLAB based on the Fast Fourier transform (FFT).

Background theory is given in Reference 1. Additional notes on the MATLAB PSD() function are given in Appendix A.

Source Code

```
function [p,f,oarms] = psdfft(y,nfft,fsamp,wndw,novlap)
%
% Estimates Power Spectral Density by periodogram method using fft() function
%
% Usage: [p,f,oarms] = psdfft(y,nfft,fsamp,wndw,novlap)
%
%        Inputs:   y        Time history vector to analyze.
%                  nfft     Number of points in each ensemble,
%                           needs to be an even number, but does not
%                           have to be a power of two (MATLAB will use
%                           a slower DFT routine if not a power of two,
%                           see 'help fft' for details).
%                  fsamp    Sample rate of data, used to normalize output
%                           and generate frequency vector.
%                  wndw     A non-zero value will apply a Hanning window
%                           of length nfft to each ensemble (requires hann.m
%                            below).
%                  novlap   Number of points to overlap each ensemble,
%                           for example nfft=1024 with novlap=512 is
%                           a 50% overlap.
%
%        Outputs:  p       Power spectral density in units of [y units]
%                          squared
%                           per [fsamp units], for example g^2/Hz.
%                  f        Frequency vector.
%                  oarms   Overall rms value, square root of area under f-p
```

```
%                          curve.
%
% Reference: Numerical Recipes in C, Second Edition, Cambridge Press,pp 549-
% 556.
%
% Code for hann.m:
%
% function [w] = hann(n)
% w = 0.5*(1-cos((2*pi*[1:n])/(n+1)))';
%

% Note that this script was written to demonstrate the method. Several
% optimizations and enhancements are possible.

% Make sure that input is a column vector
argc = size(y) ;
if (argc(1)==1); y = y' ; end
% Calculate number of available ensembles
npts = length(y) ;

ensembles = floor((npts-nfft)/(nfft-novlap))+1 ;

% Initialize ensemble indexing variables
n1 = 1 ;
n2 = nfft ;
dn = nfft-novlap ;

% Initialize psd summation storage variable
arg_sum = zeros([nfft 1]) ;
% Main program loop
for k=1:ensembles
   arg_y = y(n1:n2) ;                   % Extract current ensemble points
   arg_y = arg_y - mean(arg_y) ;        % Remove mean
   if (wndw ~= 0)
      arg_y = arg_y.*hann(nfft) ;       % Apply window if required
   end
   arg_fft = fft(arg_y,nfft) ;          % FFT of ensemble
   arg_abs = abs(arg_fft).^2 ;          % Modulus squared of FFT
   arg_sum = arg_sum + arg_abs ;        % Accumulate in summation variable
   n1 = n1+dn ;                         % Increment ensemble index
variables
   n2 = n2+dn ;
end                                     % End of main loop
arg_sum = arg_sum/ensembles ;           % Average value of summed spectra

% Compute window function normalization factor
if (wndw ~= 0)
   wndw_sc = sum(hann(nfft).^2) ;
else
   wndw_sc = nfft ;
end
```

```
% Power spectrum is symmetric about Nyquist frequency, use lower half and
% multiply by 4.  Then divide by 2 to convert peak^2 to rms^2.  Thus scale
% factor is 2.
p = 2*arg_sum(1:nfft/2+1) ;
% Normalize to correct for window
p = p/wndw_sc ;
% Normalize spectral density for sampling units
p = p/fsamp ;
% Create frequency vector
df = fsamp/nfft ;
f = [0:df:fsamp/2]';
% Calculate overall rms level from area under PSD curve
oarms = sqrt(sum(p.*df)) ;
```

Reference

1. T. Irvine, An Introduction to Spectral Functions, Vibrationdata Publications, 1999.

APPENDIX A

Several engineers were surveyed regarding the use of MATLAB to calculate power spectral density functions.

Todd Dahling replied:

> I actually don't use the FFT (directly). There is a command called "PSD" and another called "PWELCH." The difference between the two is that "PWELCH" scales the FFT's properly (by the Sample Rate). Both functions use what is called the modified Welch periodgram method to convert a time history to PSD. The basis of this is simply to divide the time history into sections (overlapping or not), window each section with the specified window (Hanning=default), FFT each section and multiply by its complex conjugate, and average over the number of sections. One could do this with the FFT command, but these two functions bring all the "stuff" together into one command.

Note that Todd Dahling's method assumes availability of the MATLAB PSD() function.