cpsd

Cross power spectral density

Syntax

```
Pxy = cpsd(x,y)
Pxy = cpsd(x,y,window)
Pxy = cpsd(x,y,window,noverlap)
[Pxy,W] = cpsd(x,y,window,noverlap,nfft)
[Pxy,F] = cpsd(x,y,window,noverlap,nfft,fs)
[...] = cpsd(...,'twosided')
cpsd(...)
```

Description

`Pxy = cpsd(x,y)` estimates the cross power spectral density `Pxy` of the discrete-time signals `x` and `y` using the Welch's averaged, modified periodogram method of spectral estimation. The cross power spectral density is the distribution of power per unit frequency and is defined as

$$P_{xy}(\omega) = \sum_{m=-\infty}^{\infty} R_{xy}(m) e^{-j\omega m}$$

The cross-correlation sequence is defined as

$$R_{xy}(m) = E\{x_{n+m} y^*{}_n\} = E\{x_n y^*{}_{n-m}\}$$

where x_n and y_n are jointly stationary random processes, $-\infty < n < \infty$, and $E\{\cdot\}$ is the expected value operator.

For real `x` and `y`, `cpsd` returns a one-sided CPSD and for complex `x` or `y`, it returns a two-sided CPSD.

`cpsd` uses the following default values:

Parameter	Description	Default Value
`nfft`	FFT length which determines the frequencies at which the PSD is estimated For real `x` and `y`, the length of `Pxy` is (`nfft`/2+1) if `nfft` is even or (`nfft`+1)/2 if `nfft` is odd. For complex `x` or `y`, the length of `Pxy` is `nfft`. If `nfft` is greater than the signal length, the data is zero-padded. If `nfft` is less than the signal length, the segment is wrapped using datawrap so that the length is equal to `nfft`.	Maximum of 256 or the next power of 2 greater than the length of each section of `x` or `y`

`fs`	Sampling frequency	1
`window`	Windowing function and number of samples to use for each section	Periodic Hamming window of length to obtain eight equal sections of `x` and `y`
`noverlap`	Number of samples by which the sections overlap	Value to obtain 50% overlap

> **Note** You can use the empty matrix `[]` to specify the default value for any input argument except `x` or `y`. For example, `Pxy = cpsd(x,y,[],[],128)` uses a Hamming window, default `noverlap` to obtain 50% overlap, and the specified 128 `nfft`.

`Pxy = cpsd(x,y,window)` specifies a windowing function, divides `x` and `y` into overlapping sections of the specified window length, and windows each section using the specified window function. If you supply a scalar for `window`, `Pxy` uses a Hamming window of that length. The x and y vectors are divided into eight equal sections of that length. If the signal cannot be sectioned evenly with 50% overlap, it is truncated.

`Pxy = cpsd(x,y,window,noverlap)` overlaps the sections of `x` by `noverlap` samples. `noverlap` must be an integer smaller than the length of `window`.

`[Pxy,W] = cpsd(x,y,window,noverlap,nfft)` uses the specified FFT length `nfft` in estimating the CPSD. It also returns `W`, which is the vector of normalized frequencies (in rad/sample) at which the CPSD is estimated. For real signals, the range of `W` is [0, pi] when `nfft` is even and [0, pi) when `nfft` is odd. For complex signals, the range of `W` is [0, 2*pi).

`[Pxy,F] = cpsd(x,y,window,noverlap,nfft,fs)` returns `Pxy` as a function of frequency and a vector `F` of frequencies at which the CPSD is estimated. `fs` is the sampling frequency in Hz. For real signals, the range of `F` is [0, `fs`/2] when `nfft` is even and [0, `fs`/2) when `nfft` is odd. For complex signals, the range of `F` is [0, `fs`).

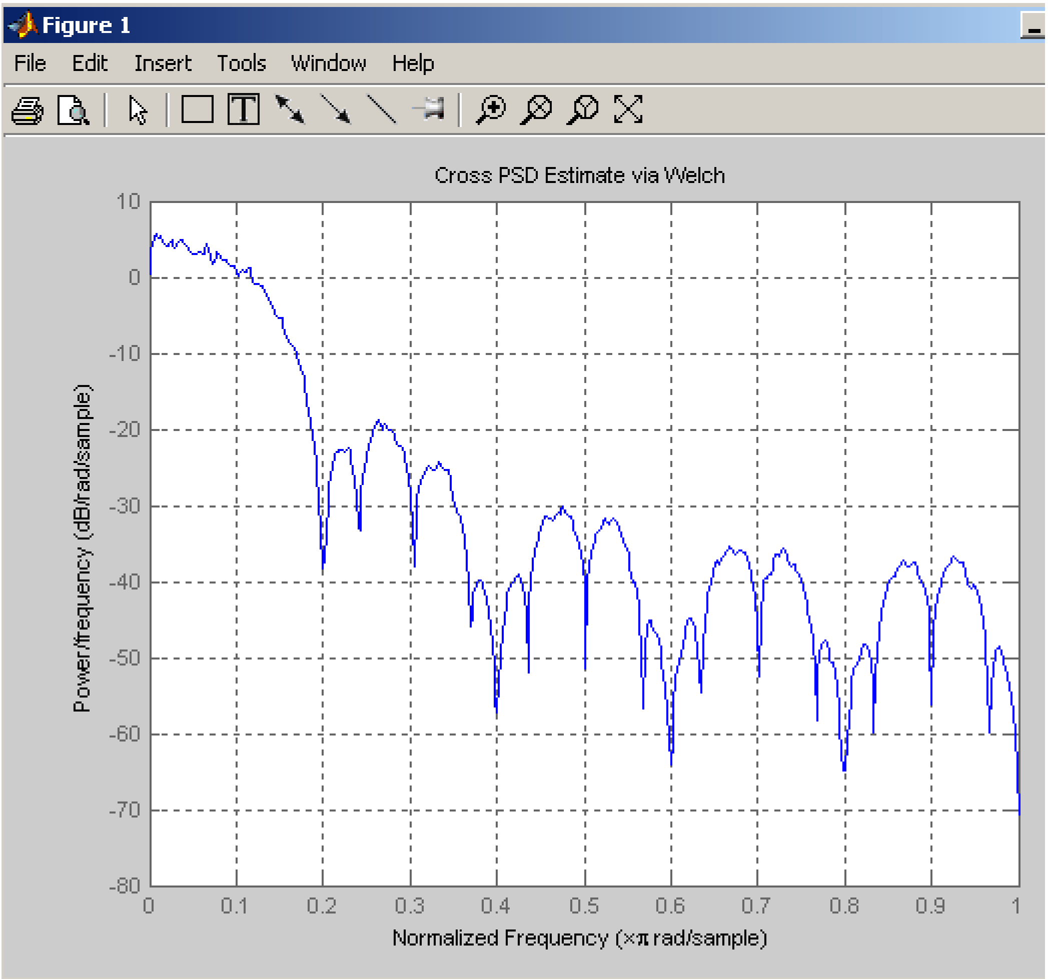
`[...] = cpsd(...,'twosided')` returns the two-sided CPSD of real signals `x` and `y`. The length of the resulting Pxy is nfft and its range is [0, 2*pi) if you do not specify fs. If you specify fs, the range is [0,fs). Entering'onesided'for a real signal produces the default. You can place the 'onesided' or 'twosided' string in any position after the `noverlap` parameter.

`cpsd(...)` plots the CPSD versus frequency in the current figure window.

Examples

Generate two colored noise signals and plot their CPSD with a confidence interval of 95%. Specify a length 1024 FFT, a 500 point triangular window with no overlap, and a sampling frequency of 10 Hz:

```
randn('state',0);
h = fir1(30,0.2,rectwin(31));
h1 = ones(1,10)/sqrt(10);
r = randn(16384,1);
x = filter(h1,1,r);
y = filter(h,1,x);
cpsd(x,y,triang(500),250,1024)
```



Algorithm

`cpsd` uses Welch's averaged periodogram method. See the references listed below.

References

[1] Rabiner, L.R., and B. Gold. *Theory and Application of Digital Signal Processing,* Englewood Cliffs, NJ: Prentice-Hall, 1975. Pgs.414-419.

[2] Welch, P.D. "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms." *IEEE® Trans. Audio Electroacoust,* Vol. AU-15 (June 1967). Pgs.70-73.

[3] Oppenheim, A.V., and R.W. Schafer.*Discrete-Time Signal Processing,* Upper Saddle River, NJ: Prentice-Hall, 1999, pp. 737.

See Also

`dspdata`, `mscohere`, `pburg`, `pcov`, `peig`, `periodogram`, `pmcov`, `pmtm`, `pmusic`, `pwelch`, `pyulear`, `spectrum`, `tfestimate`